

*EspressReport® ES*

Version *7.0*

## **EspressReport ES 7.0 User's Guide**



Quadbase Systems Inc.

EspressReport ES User's guide .....	ix
Q. QuickStart .....	1
Q.1. Overview .....	1
Q.1.1. Setup .....	1
Q.2. Getting Started .....	1
Q.2.1. Create a User .....	2
Q.2.2. Start the Organizer .....	3
Q.3. Setup Data Sources .....	7
Q.3.1. Create a Data Registry .....	8
Q.4. Report Designer .....	20
Q.4.1. Report Mapping .....	20
Q.4.2. Basic Report Formatting .....	35
Q.4.3. Advanced Reporting Features .....	44
Q.5. Chart Designer .....	58
Q.5.1. Chart Mapping .....	58
Q.5.2. Basic Chart Formatting .....	66
Q.5.3. Advanced Charting Features .....	71
Q.6. QuickDesigner Reports .....	74
Q.6.1. Create a Query .....	74
Q.6.2. Create a Report .....	79
Q.7. QuickDesigner Charts .....	84
Q.7.1. Create a Chart .....	85
Q.8. Online Maps .....	89
Q.8.1. Create Coordinates .....	89
Q.8.2. Create Online Map .....	93
Q.9. SVG Maps .....	100
Q.9.1. Create a Map .....	100
Q.9.2. Set SVG Map Thresholds .....	102
Q.9.3. Set SVG Map Drilldowns .....	105
Q.10. Publishing .....	106
Q.10.1. Dashboards .....	106
Q.10.2. URLs .....	118
Q.10.3. The Menu Page .....	120
Q.11. Alerts .....	120
Q.11.1. Alerts in Charts .....	120
Q.11.2. Alerts in Maps .....	122
Q.11.3. Alerts in Dashboards .....	123
Q.11.4. Alert Monitoring .....	125
Q.12. API .....	129
Q.12.1. Set Up Environment .....	129
Q.12.2. Run a Report .....	130
Q.12.3. Create a Report Programmatically .....	132
Q.12.4. Create a Chart Programmatically .....	135
Q.12.5. Modify Data Source of a Report .....	138
Q.12.6. Modify Report Elements .....	149
Q.12.7. Parameterized Reports .....	152
Q.12.8. Deploy/Export Drill-Down .....	159
Q.12.9. Launch Report Designer .....	163
1. Administration .....	165
1.1. Overview & Architecture .....	165
1.1.1. ERES Components .....	165
1.1.2. ERES Architecture .....	166
1.2. New Features List V7.0 .....	167
1.2.1. New EspressReport ES Features .....	167
1.3. Installation and Configuration .....	168
1.3.1. Installing ERES .....	168
1.3.2. Configuration .....	177
1.3.3. Backward compatibility patches .....	181

---

1.3.4. Run Applets As JNLP .....	183
1.4. Administration .....	184
1.4.1. The Console Interface .....	184
1.4.2. User Settings .....	212
1.A. ERES Details Log .....	213
1.B. Server Commands .....	218
2. Organizer .....	222
2.1. Using the Organizer .....	222
2.1.1. Starting the Organizer .....	222
2.1.2. The Organizer Interface .....	223
2.1.3. Projects and Folders .....	226
2.1.4. Files .....	227
2.1.5. URL Mapping .....	231
2.1.6. Repairing Broken Links .....	232
2.1.7. Update Directory .....	233
2.1.8. Searching in Organizer .....	236
2.1.9. Limiting Browse Directories .....	237
2.2. Scheduling & Archiving .....	237
2.2.1. Setting a Schedule .....	237
2.2.2. Setting an Archive .....	250
2.2.3. View Tasks .....	251
2.3. Security and Security Administration .....	256
2.3.1. Security Concepts .....	257
2.3.2. Setting User Privileges .....	258
2.3.3. Security Levels .....	261
3. Data Sources .....	263
3.1. Data in Organizer .....	263
3.1.1. Managing Data Registries .....	263
3.1.2. The Data Source Manager .....	265
3.1.3. Data from a Database .....	268
3.1.4. Data from XML and XBRL Files .....	303
3.1.5. Data from Text Files .....	307
3.1.6. Data from Class Files .....	308
3.1.7. Data from EJBs .....	309
3.1.8. Data from SOAP with WSDL support .....	311
3.1.9. Data from Salesforce .....	313
3.1.10. Data from Excel files .....	316
3.1.11. Using Data for Charts and Reports .....	318
3.1.12. Data Source Updating .....	321
3.1.13. CData JDBC drivers .....	323
3.2. Data in QuickDesigners and Maps .....	328
3.2.1. Select a Data Source .....	328
3.2.2. Queries .....	329
4. Designing Reports & Charts .....	341
4.1. Report Designer .....	341
4.1.1. Introduction to Report Designer .....	341
4.1.2. Report Types and Data Mapping .....	342
4.1.3. The Designer Interface .....	361
4.1.4. The Preview Window .....	415
4.1.5. Saving and Exporting Reports .....	420
4.1.6. Using Formulas & the Formula Builder .....	425
4.1.7. Scripting .....	440
4.1.8. Drill-Down .....	451
4.1.9. Sub-Reports .....	458
4.1.10. Template Security .....	462
4.2. Chart Designer .....	466
4.2.1. Introduction to Chart Designer .....	466
4.2.2. Charting Basics .....	470

---

---

4.2.3. Chart Types and Data Mapping .....	475
4.2.4. The Chart Designer Interface .....	505
4.2.5. Chart Drill-Down .....	572
4.2.6. Saving & Exporting Charts .....	577
4.3. QuickDesigner Reports .....	582
4.3.1. Start .....	582
4.3.2. Select a Data Source .....	582
4.3.3. Toolbar .....	584
4.3.4. Format Report Elements .....	584
4.3.5. Parameter Setting .....	591
4.3.6. Aggregation/Group .....	592
4.3.7. Data Formatting .....	594
4.3.8. Sorting .....	597
4.3.9. Data Filtering .....	597
4.3.10. Save the Report .....	598
4.3.11. Export the Report .....	598
4.3.12. Open the Saved Report .....	599
4.3.13. Exit .....	599
4.4. QuickDesigner Charts .....	599
4.4.1. Start .....	600
4.4.2. Select a Data Source .....	600
4.4.3. Data Mapping and Ordering .....	601
4.4.4. Toolbar .....	603
4.4.5. Data Formatting .....	604
4.4.6. Save the Chart .....	624
4.4.7. Export the Chart .....	625
4.4.8. Open the Saved Chart .....	626
4.4.9. Exit .....	627
5. Designing Maps .....	628
5.1. Introduction to ERES Maps .....	628
5.2. Online Maps .....	628
5.2.1. Generating Google Maps API Key .....	628
5.2.2. Data Sources .....	629
5.2.3. Start Online Maps .....	630
5.2.4. Coordinates Editor .....	631
5.2.5. Coordinates Mapping .....	640
5.2.6. Create Online Maps .....	641
5.2.7. Save Map .....	653
5.2.8. Open Saved Map .....	654
5.2.9. Exit .....	655
5.3. SVG Maps .....	655
5.3.1. Introduction to SVG Maps .....	655
5.3.2. Set Area IDs Using Inkscape Editor .....	655
5.3.3. Area ID Mapping .....	656
5.3.4. SVG Maps Designing .....	656
5.3.5. Dynamic SVG Maps .....	667
5.3.6. Save Map .....	671
5.3.7. Open Saved Map .....	672
5.3.8. Exit .....	672
5.4. Map Viewer .....	672
5.4.1. Writing Map Viewer URL .....	673
5.5. Migrating Maps .....	674
5.A. List of SVG Map Images .....	674
6. Designing Dashboards .....	676
6.1. Introduction to Dashboards .....	676
6.2. Create Dashboard .....	677
6.2.1. Toolbar .....	678
6.2.2. Responsive Dashboard .....	679

---

---

6.2.3. Add Charts, Reports and Maps .....	680
6.2.4. Shared Parameters .....	683
6.2.5. Insert Labels .....	698
6.2.6. Add Background .....	700
6.2.7. Additional Options .....	701
6.2.8. Drilldown .....	703
6.2.9. Folders .....	704
6.2.10. Template Linkage .....	708
6.2.11. Dashboard Preview .....	709
6.3. Dashboard Migration .....	709
6.3.1. Migrating from previous ERES versions .....	709
6.4. Save Dashboard .....	712
6.5. Open Saved Dashboard .....	712
6.6. Exit .....	713
7. Publishing (Menu & URLs) .....	714
7.1. The Menu Page .....	714
7.1.1. Launching the Menu Page .....	714
7.1.2. Using the Menu Page .....	714
7.1.3. Mobile MenuPage .....	722
7.2. Image URLs .....	726
7.2.1. Generating URLs in Organizer .....	726
7.2.2. Writing Image URLs .....	728
7.3. Report URLs .....	745
7.3.1. Generating URLs in Organizer .....	745
7.3.2. Writing Report URLs .....	747
7.4. Dashboard URLs .....	759
7.4.1. Generating URLs in Organizer .....	759
7.4.2. Writing Dashboard URLs .....	761
7.5. Menu Page Listener .....	762
7.5.1. ERES Listener Manager .....	762
7.5.2. Using the Menu Page Listener .....	763
7.6. Report Viewer .....	764
7.6.1. The Report Viewer Parameters .....	765
7.6.2. Specifying Data for Report Viewer .....	766
7.6.3. Using Report Viewer .....	767
7.6.4. Connecting to the ERES Server .....	769
7.6.5. Swing Version .....	769
7.6.6. Exporting from Viewer .....	769
7.7. Page Viewer .....	770
7.7.1. Launching Page Viewer .....	770
7.7.2. The Page Viewer Parameters .....	771
7.7.3. Using Page Viewer .....	772
7.7.4. Buffer Time .....	773
7.7.5. Connecting to the ERES Server .....	773
7.7.6. Swing Version Available .....	774
7.7.7. Exporting from Viewer .....	774
7.7.8. Unmapped Drilldown Parameter .....	774
7.8. Chart Viewer .....	775
7.8.1. The Chart Viewer Parameters .....	776
7.8.2. Specifying the Data Source for Chart Viewer .....	778
7.8.3. Using Chart Viewer .....	781
7.8.4. Axis Rulers .....	782
7.8.5. Pop-up Menu .....	782
7.8.6. Swing Version Available .....	783
7.9. Dashboard Viewer .....	783
7.9.1. Preview Toolbar .....	783
7.9.2. Preview Options .....	784
8. Programming .....	788

---

---

8.1. ERES Report API .....	788
8.1.1. Introduction and Setup .....	788
8.1.2. Recommended Approach for Using Report API .....	788
8.1.3. Interaction with ERES Server .....	789
8.1.4. Connecting to ERES Server .....	789
8.1.5. Using the API .....	790
8.1.6. Javadoc .....	845
8.1.7. Swing Version .....	845
8.1.8. Summary .....	845
8.2. ERES Chart API .....	845
8.2.1. Introduction and Setup .....	845
8.2.2. Recommended Approach for using Chart API .....	846
8.2.3. Interaction with the ERES Server .....	847
8.2.4. Connecting to ERES Server .....	847
8.2.5. Using the API .....	848
8.2.6. API Only Features .....	863
8.2.7. Changing Chart Viewer Options .....	879
8.2.8. Javadoc .....	879
8.2.9. Swing Version .....	879
8.2.10. Summary .....	880
8.3. Managing Users and Groups .....	880
8.3.1. Introduction .....	880
8.3.2. Users and Groups .....	880
8.3.3. Single Sign-On .....	882
8.3.4. Login Listener .....	883
8.3.5. Javadoc .....	884
8.3.6. Summary .....	884
8.4. ERES Menu API Overview .....	884
8.4.1. Introduction and Setup .....	884
8.4.2. Using the API .....	884
8.4.3. Javadoc .....	889
8.4.4. Summary .....	889
8.5. Menu JSP Programming .....	889
8.5.1. Menu Page Architecture .....	889
8.5.2. Login Code .....	890
8.5.3. Building the Main Page .....	890
8.5.4. Running Reports and Charts .....	894
8.5.5. Retrieving Scheduled Reports and Charts .....	896
8.5.6. Running Archived Reports and Charts .....	898
8.5.7. Getter Methods .....	899
8.6. Organizer .....	906
8.6.1. Introduction .....	906
8.6.2. Look and Feel .....	906
8.6.3. Inserting and Removing Items .....	907
8.6.4. Customizing Report and Chart Designer .....	908
8.6.5. Calling Scheduler .....	911
8.6.6. Javadoc .....	911
8.6.7. Summary .....	911
8.7. Scheduler .....	911
8.7.1. Introduction .....	911
8.7.2. Scheduling an Export .....	912
8.7.3. Scheduling Multiple Exports .....	917
8.7.4. Scheduling an Archive .....	919
8.7.5. Removing a Schedule .....	919
8.8. SOAP Interface .....	924
8.8.1. LookupService Running .....	924
8.9. Deployment .....	929
8.9.1. Introduction .....	929

---

---

8.9.2. Deploying with ERES Server .....	930
8.9.3. Deploying without ERES Server .....	932
8.9.4. Deploying in a non-Windows environment .....	934
8.9.5. Platform Specific Issues .....	935
8.A. Getting the Report Data .....	936
8.A.1. Data from a Database .....	937
8.A.2. Data from a Data File (TXT/DAT/XML) .....	938
8.A.3. Data from an XML Data Source .....	939
8.A.4. Data passed in an Array in Memory .....	940
8.A.5. Data passed in a Custom Implementation .....	941
8.A.6. Data from Enterprise Java Beans (EJBs) .....	944
8.A.7. Data from a SOAP Data Source .....	945
8.A.8. Data from Multiple Data Sources .....	946
8.B. Creating the Report .....	947
8.B.1. Simple Columnar .....	947
8.B.2. Summary Break .....	949
8.B.3. CrossTab .....	952
8.B.4. Master & Details .....	955
8.B.5. Mailing Labels .....	957
8.B.6. Formula .....	960
8.B.7. Parameterized Report .....	960
8.B.8. Sub-Report .....	963
8.B.9. Drill-Down .....	967
8.C. Getting the Chart Data .....	969
8.C.1. Data from a Database .....	970
8.C.2. Data from a Data file (TXT/DAT/XML) .....	971
8.C.3. Data from a XML Data Source .....	972
8.C.4. Data Passed in an Array in Memory .....	973
8.C.5. Data Passed in your Custom Implementation .....	974
8.C.6. Data from a Spreadsheet Model .....	977
8.C.7. Data from Enterprise Java Beans (EJBs) .....	978
8.C.8. Data from multiple Data Sources .....	979
8.C.9. Data in Spreadsheet Format .....	980
8.C.10. Transposing Data .....	981
8.D. Creating the Chart .....	983
8.D.1. Column, Bar, Line, Area, Pie and Overlay Charts .....	984
8.D.2. Radar Charts .....	986
8.D.3. XY(Z) Scatter Charts .....	987
8.D.4. Stack Column, Percentage Column, Stack Bar and Stack Area Charts .....	989
8.D.5. Dial Charts .....	991
8.D.6. Box Charts .....	992
8.D.7. Bubble Charts .....	994
8.D.8. High-Low and HLCO Charts .....	996
8.D.9. Surface Charts .....	998
8.D.10. Gantt Charts .....	1000
8.D.11. Polar Charts .....	1002
8.D.12. Date/Time Based Zoom Charts .....	1003
8.D.13. Parameterized Charts .....	1005
8.D.14. Drill-Down Charts .....	1008
8.D.15. Adding a Chart to a Report .....	1014
9. Configuration .....	1017
9.1. Using Other Databases .....	1017
9.1.1. Create Table Scripts .....	1017
9.1.2. Specifying the Database Connection .....	1018
9.1.3. Migrating the ERES Database .....	1019
9.2. Integrating Existing Users/Groups .....	1020
9.2.1. Using a Database .....	1020
9.2.2. Implementing UserSecurityProvider .....	1022

---

---

9.3. Using Other Application Servers .....	1027
9.3.1. Tomcat 4.1/5.x/6.0.x/7.0.x .....	1029
9.3.2. Resin .....	1029
9.3.3. WebLogic Server .....	1030
9.3.4. WebSphere .....	1036
9.3.5. JBoss 3.2.3 (with Tomcat 5.0) / JBoss 4.0.5/6.0.0 .....	1043
9.3.6. Oracle Containers for J2EE (OC4J) 10g (9.0.4.0/10.1.3.5) .....	1044
9.3.7. Oracle Application Server 10g R3 (10.1.3) .....	1046
9.3.8. GlassFish Server 3.0.1 .....	1046
9.4. Clustering/Load Balancing .....	1047
9.4.1. Tomcat 5.0 .....	1048
9.4.2. JBoss 3.2.5 (with Tomcat 5.0) .....	1054
9.4.3. JRun 4 (with Apache Web server) .....	1056
9.4.4. WebSphere 6.0 .....	1058
10. Internationalization .....	1060
10.1. Internationalizing ERES .....	1060
10.1.1. Specifying Locales .....	1060
10.1.2. Language and Encoding .....	1060
11. Alerts .....	1066
11.1. What is an Alert .....	1066
11.2. Specifying alerts .....	1066
11.2.1. Online Maps .....	1066
11.2.2. SVG Maps .....	1066
11.2.3. Charts .....	1067
11.2.4. Reports .....	1067
11.3. Dashboard alerts .....	1068
11.3.1. Visual alert display .....	1068
11.3.2. Alert messages .....	1068
11.3.3. Sound alert .....	1069
11.3.4. User interface .....	1069
11.4. Monitoring .....	1070
11.4.1. Alert notifications .....	1070
11.4.2. Failed emails .....	1070
11.4.3. User interface .....	1070
11.4.4. Monitoring log .....	1075
11.5. Handling special situations .....	1079
11.5.1. Change in script/control area/range .....	1079
11.5.2. Missed monitoring jobs .....	1080

---

# EspressReport<sup>®</sup> ES 7.0

## EspressReport ES User's guide

Welcome to the EspressReport ES Online User's Guide.

Here you will find information about all the functions and features in EspressReport ES. For additional API resources, please refer to the API JavaDoc for both the Report API and ERES API.

**EspressReport ES Users guide consists of several elements:**

<b>QuickStart</b>	This is a good starting point for EspressReport ES first time users. It explains some of the most commonly used features, and provides examples of administration, report and chart design, and publishing.
<b>Administration</b>	This section provides an overview of ERES along with installation instructions, and information for administering the reporting environment.
<b>Organizer</b>	This section explains how to use the Organizer interface, how to schedule and archive reports, security features, and provides information about using and connecting to different data sources.
<b>Designing Reports &amp; Charts</b>	This section explains how to use all the design tools including the Report Designer, Chart Designer, thin-client QuickDesigner Reports and QuickDesigner Charts.
<b>Designing Maps</b>	This section explains how to use map designers interface for creating Online Maps and SVG maps.
<b>Designing Dashboards</b>	This section explains how to use Dashboard Builder interface to create and deploy dashboards as well as migrating Dashboards from one ERES Server to another.
<b>Publishing (Menus &amp; URLs)</b>	This section covers the built-in report and publishing features available with ERES. API deployment features are covered in the Programming section.
<b>Programming</b>	This section explains how to use the Report API and the ERES API to create, modify, and deploy charts and reports programmatically.
<b>Configuration</b>	This section explains how to configure EspressReport ES to run with different application servers and databases.
<b>Internationalization</b>	Explains how to set ERES to create dashboards in virtually any language and locale-specific formatting.
<b>Alerts</b>	Explains how to monitor your KPI (Key Performance Indicator) alerts.

If you upgraded from an older ERES version, you might be interested in seeing the list of V7.0 new features.



### Tip

If you're reading the PDF version, enable *Bookmarks* toolbar in your PDF viewer.

**Please select an option from the menu on the left to begin.**

---

# QuickStart

## Q.1. Overview

Welcome to the EspressoReport ES Quick Start Guide. This guide is the recommended starting point for users who are new to ERES and would like to get up to speed fairly quickly. This guide can also help users to quickly evaluate the product capabilities.

This guide provides step-by-step tutorials for basic administration features, report and chart creation, publishing features and programming examples. Only a small portion of the features are described in this guide so for more information and detail, please refer to the other sections in the User's Guide.

### Q.1.1. Setup

The QuickStart assumes that you are working with ERES in default configuration. This means a clean installation using the Tomcat server that comes with ERES and running connected to the default HSQL database. For more information about installation options, please see Section 1.3.1 - Installing ERES. The information and exercises in this guide can be used in other configurations, however, references to certain options and file paths may not be applicable.

For more information about ERES components and architecture, see Section 1.1.1 - ERES Components.

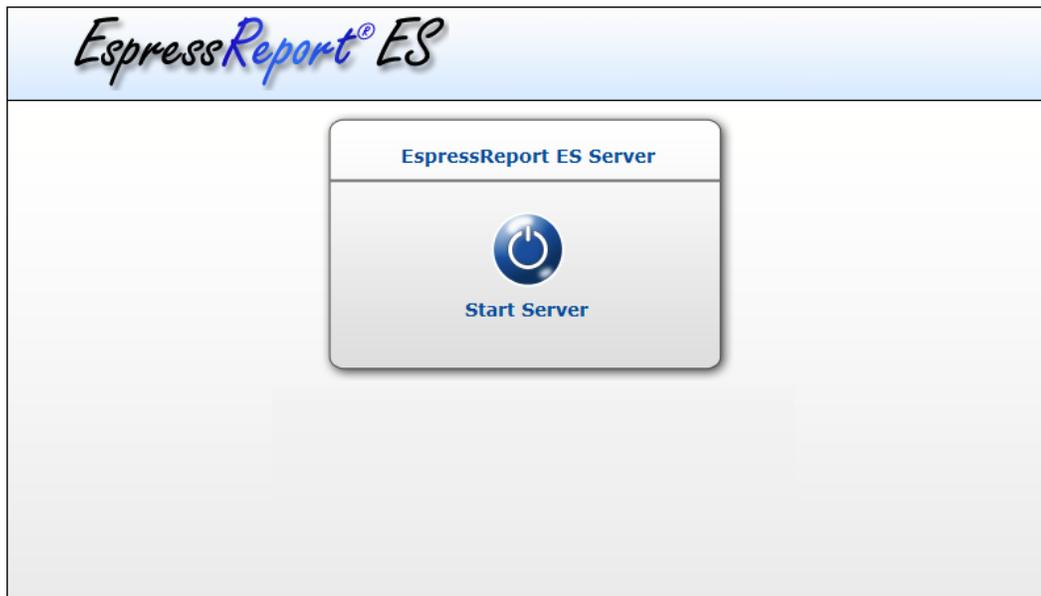
## Q.2. Getting Started

The first step to run ERES is to start the application server, if it didn't auto-start (installation default). In this case, it means starting the Tomcat server that comes with the ERES installation. To start the Tomcat, execute `start-up.bat/.sh` in the `/bin/` directory of your Tomcat installation. If you installed ERES on Windows, you can also start Tomcat from a shortcut in the Start Menu.

After the Tomcat is successfully started, you can load the Start page for ERES from the following location:

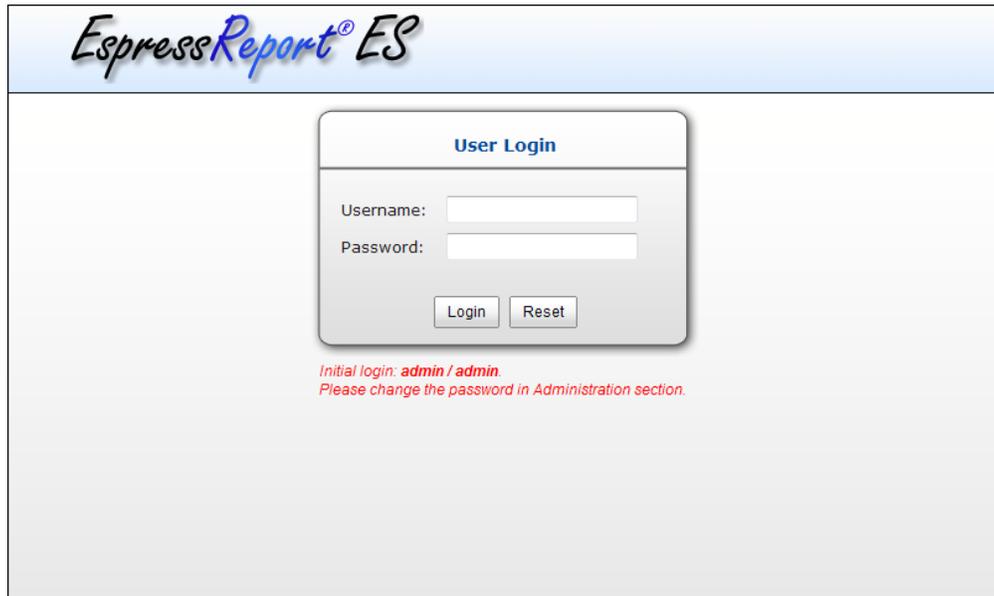
`http://<machinename>:<port>/ERES/index.jsp`

Replace `<machinename>` with your server hostname or IP address, and replace port with the port you specified for Tomcat during installation. The default port is 8080. If you installed ERES on Windows, you can launch the Start page from the Start Menu.



*ERES Start Page - ERES Server is OFF*

Before we start working with ERES, the ERES Server needs to be running. If the `Autostart` feature has been enabled during installation process (see Section 1.3.1 - Installing ERES for more details), the ERES Server should be now running. If the server is off, turn it on by clicking on the *Start Server* button.

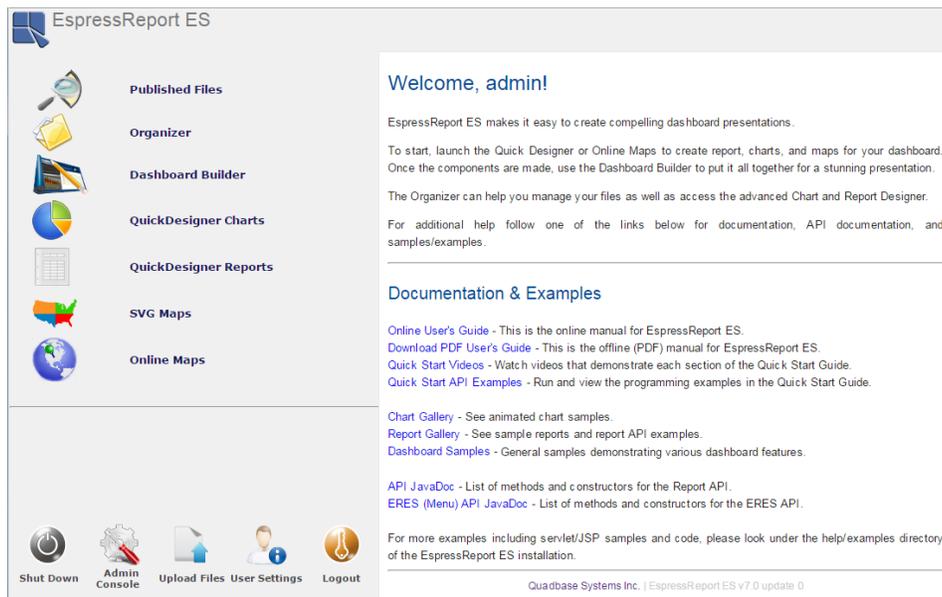


*ERES Server Started*

Now you need to first login before continuing.

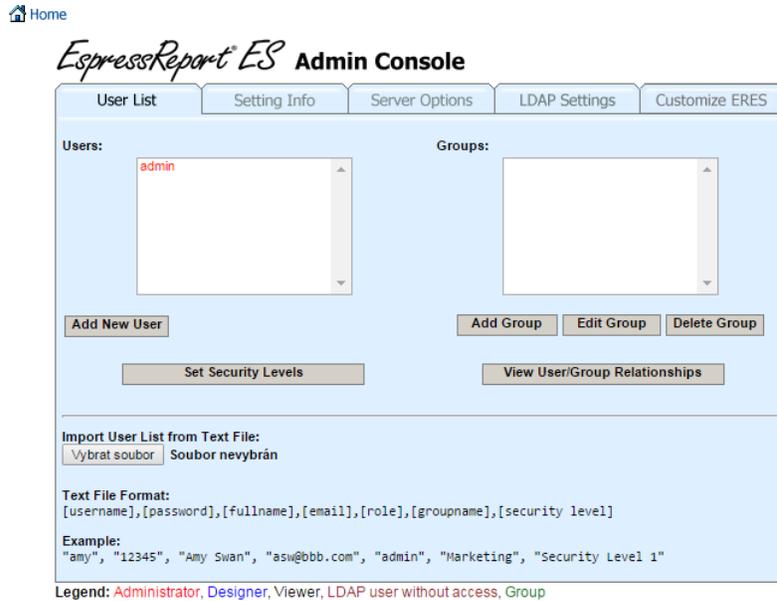
## Q.2.1. Create a User

With the server running, login as the system administrator. The default login for the administrator is username: **admin**, and password: **admin**. Once you have logged in, the Start page will be displayed.



*Admin Logged into Start Page*

To launch the Admin Console, click the *Admin Console* button in the lower left corner of the Start page.



Admin Console

The Admin Console opens the *User List* tab. By default, the administrator is the only defined user. To add a new user to the system, click the *Add New User* button. This will open a window allowing you to add details about a new user.

**User Details:**

Username: Tom

Full Name: Tom Weeks

Email Address: tom@quadbase.com

Password: \*\*\*\*\*

Re-Type Password: \*\*\*\*\*

Primary Role:  
 Administrator  Designer  Viewer

Ok Cancel

Add User Dialog

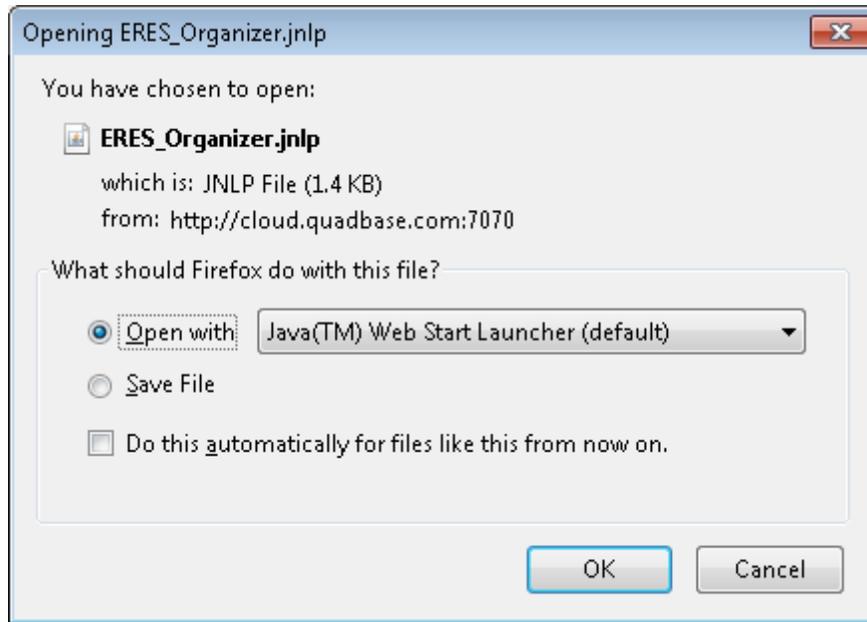
Enter details for a new user and make sure to select **Designer** as the primary role. Please note that if you select **Viewer** as the primary role, you will not have access to the core design/development tools like Chart Designer, Report Designer and Organizer. For more information about creating users, please visit Section 1.4.1.1 - User List.

When you're done, click the *Ok* button to create a new user. The dialog will close and the new user will be added to the user list in the Admin Console.

Now that you have entered a new user, return to the Start page by clicking on the  **Home** Home icon in the upper left corner of the Admin Console.

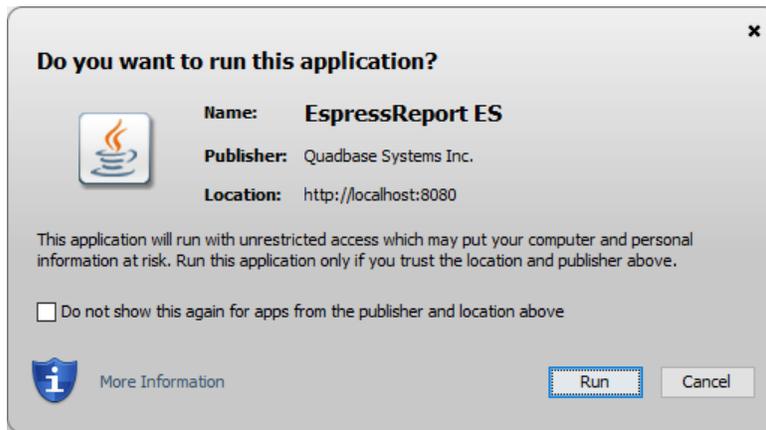
## Q.2.2. Start the Organizer

While we are still logged in as admin, let's do one more thing that will facilitate some of the exercises later. Click on the *Organizer* in the left menu of the Start page. This will open a pop-up window. Confirm `ERES_Organizer.jnlp` file open by Java Web Start Launcher and the Organizer will open.



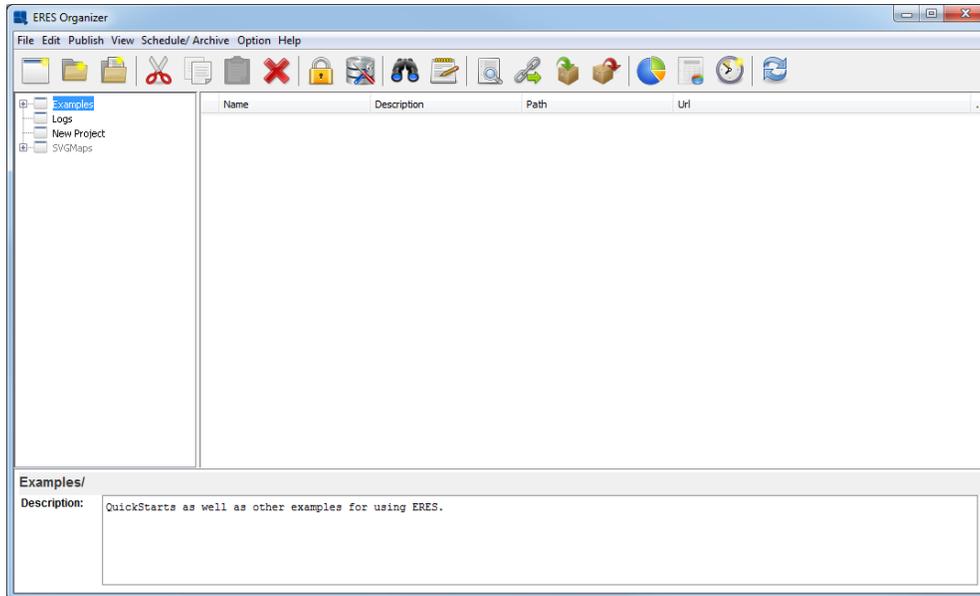
*Opening Organizer*

If you're using a different browser like Chrome, you have to save the `ERES_Organizer.jnlp` file first and then open it from your download directory. You will see a window with a question: "Do you want to run this application?" Click the *Run* button and the Organizer will open.



*Opening Organizer in Chrome*

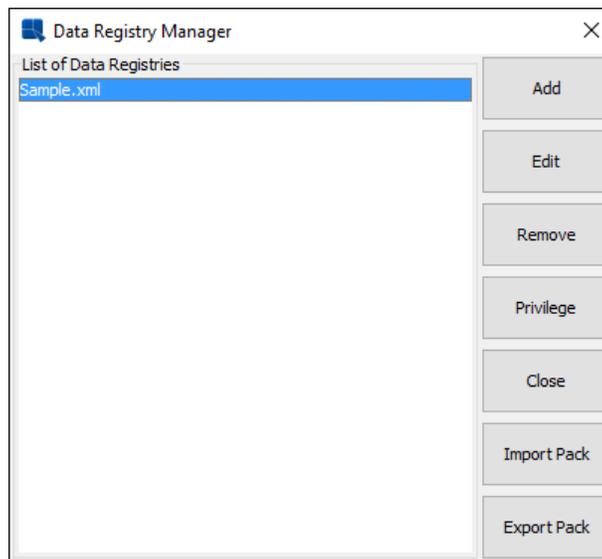
Once the Organizer is opened, you will see this window:



*Organizer Interface*

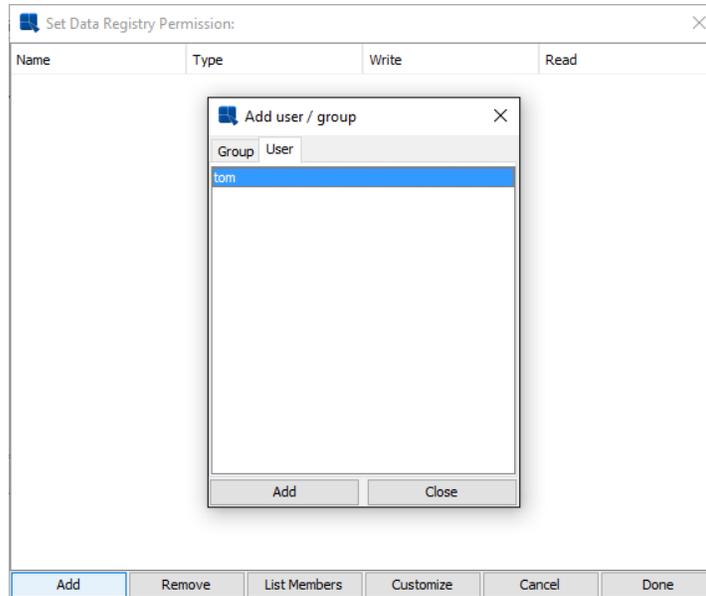
The Organizer is a virtual file management system that allows you to create, manage, publish, and schedule reports and charts. The Organizer is used in conjunction with most of the other interfaces in ERES and will be referenced extensively throughout this guide.

Once the Organizer is opened, click the *Manage Data Sources* button  and the Data Registry Manager window will pop-up. Highlight `Sample.xml` and click the *Privilege* button in the lower right corner of the window.



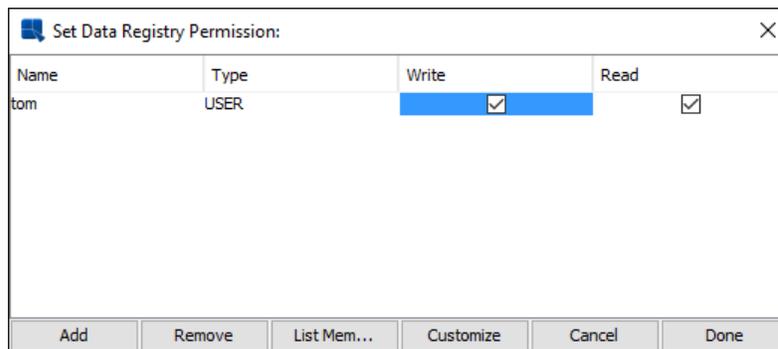
*Data Registry List*

A new dialog *Set Data Registry Permission* will be displayed. Click the *Add* button and switch to the *User* tab. Highlight the user you previously created via Admin Console and click the *Add* button. Close the window by clicking the *Close* button.



*Add Data Registry Privilege*

Now check the *Write* and *Read* checkboxes in the Set Data Registry Permission window and click the *Done* button to close it.



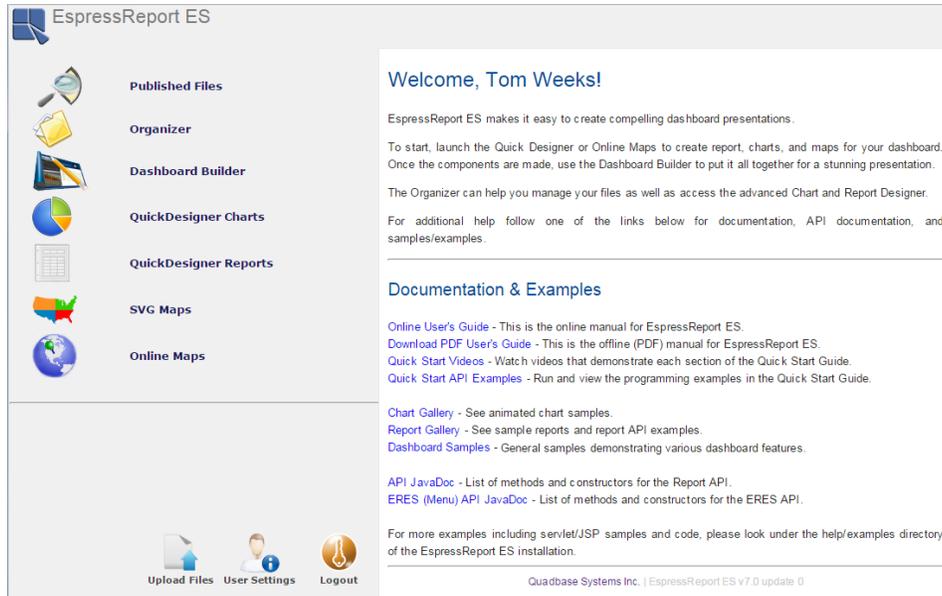
*Data Registry Read Write Privilege*

Close the Data Registry Manager window by clicking on the *Close* button and then close the Organizer. You should now be on the Start Page.

### Q.2.2.1. Login as User

We are done with all the administration activities, so you can now login as the designer user you created previously. On the Start page, click the *Logout* button. This will logout the administrator out of the system.

Now log in again using the user that you created in Section Q.2.1 - Create a User. After you login, notice that the administration functions are not available. Click on the *Organizer* on the left to open the Organizer.



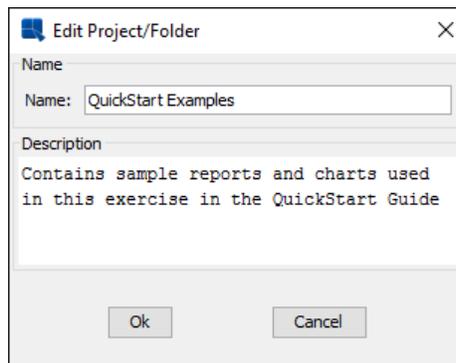
*User Logged into Start Page*

### Q.2.2.2. Add a Project

Reports and charts in the Organizer are grouped into projects and folders. Now we are going to create a new project in which to add reports and charts that we will use later in this guide. To add a new project, click the *New Project*

button on the toolbar . A new node will be added with the name *New project* or *New project(1)*, if a node of that name already exists.

To edit the new project, right click on the new node and select *Edit* from the pop-up menu. This will bring up a dialog allowing you to re-name the project and to specify a description.



*Edit Project Dialog*

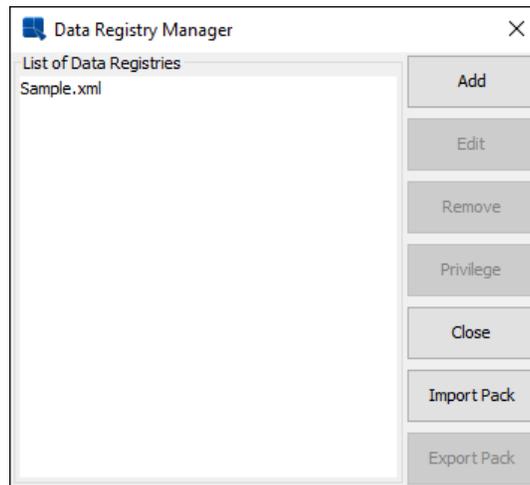
Specify a new name and description for the project and click *Ok*. The changes will be applied. When you select a project in the Organizer, you will see its description at the bottom of the page.

## Q.3. Setup Data Sources

Data sources in ERES are maintained in XML data registry files. These files are created and managed from within the Organizer. Detailed documentation about the data sources can be found in Section 3.1 - Data in Organizer.

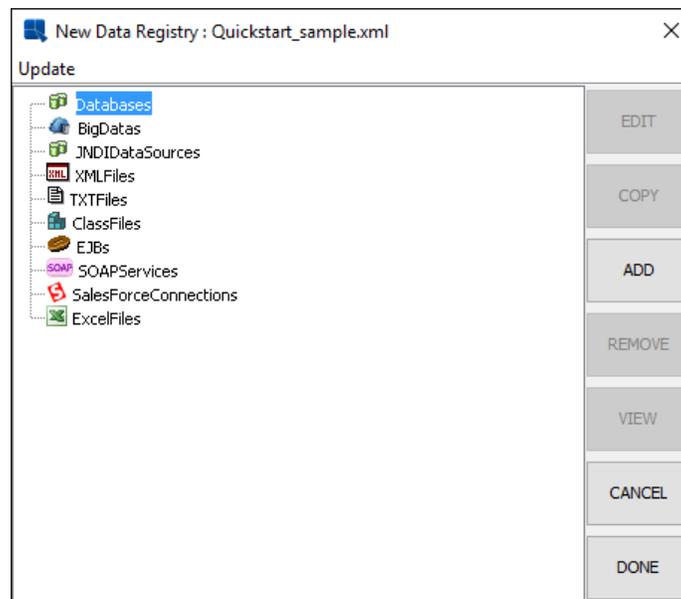
## Q.3.1. Create a Data Registry

To create a new data registry, open the Organizer and click the *Manage Data Sources* button . This will pop up a dialog containing all the registries defined in ERES.



*Registry List Dialog*

Click the *Add* button in this dialog to add a new data registry. A dialog will open prompting you to specify a name for the registry. Enter any name you like (e.g. **Quickstart\_sample**) and click the *Ok* button. The registry will be created and the Data Source Manager window for the new registry will open.



*Data Source Manager Window*

### Q.3.1.1. Setup Database Connections

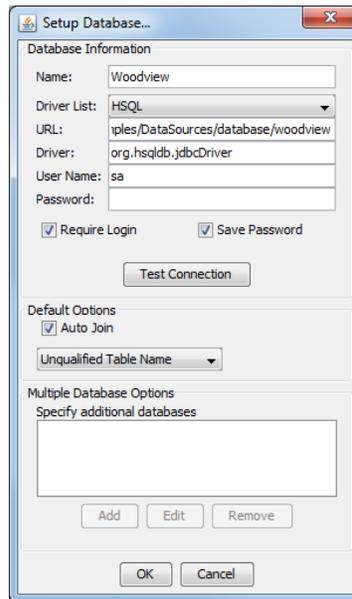
ERES allows you to connect to JDBC compliant data sources. Few examples are included in your installation.

#### Q.3.1.1.1. Setup a JDBC Connection

In this tutorial, we will set up a JDBC connection to the Woodview HSQL database that comes with the ERES Installation. If you are running ERES using the configuration described in Section Q.1.1 - Setup then the HSQL JDBC driver will already be in the classpath of your Tomcat server, as HSQL is also used as the default ERES database. If this is not the case, or if you are using a different database for ERES, you will need to make sure that

your database driver (hsqldb.jar for HSQL) is in the <ERESInstallDir>/WEB-INF/lib directory, or in Tomcat's classpath.

The classpath and Woodview sample database were already created during installation, so you can select the Woodview database in your registry. From the Data Source Manager window, click on the *Databases* node in the left panel, select Woodview and hit the *Edit* button to reach the following *Setup Database* panel, where HSQL is on the *Driver List*, `jdbc:hsqldb:help/examples/DataSources/database/woodview` for the *URL*, and `org.hsqldb.jdbcDriver` as the *Driver*. Click on both the *Require Login* and *Save Password* boxes. Next, enter `sa` for the *User Name* and leave the *Password* blank will connect to the built-in HSQL database.



*Add Database Dialog*

The procedure for connecting to other databases is very similar. First, select the type from the database list and then fill in the values. The JDBC Driver will be automatically provided so it is usually not necessary to modify this field. Here is an example connecting to a MySQL database.

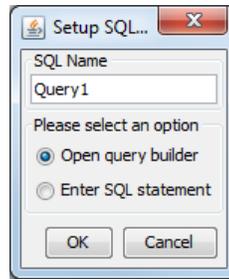
```
URL: jdbc:mysql://192.168.0.55:3306/woodview
Driver: com.mysql.jdbc.Driver
User Name: root
Password: *****
```

Leave the *Auto Join* and *Table Name* properties alone, and click the *Test Connection* button to make sure you've entered the information correctly. Click *OK* to bring up the Data Source Manager window, where there will be an existing node under *Databases* for Woodview.

### Q.3.1.2. Create a Query

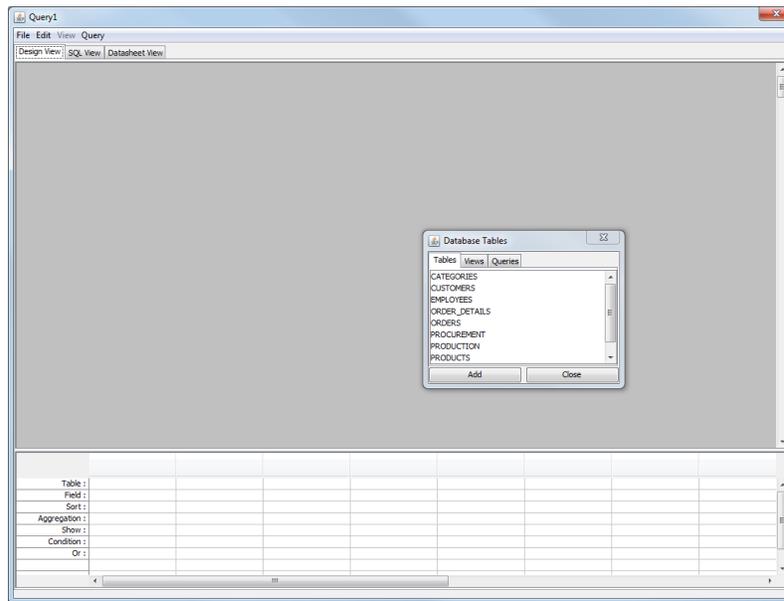
ERES provides a number of different interfaces to query a database to retrieve the report data. You can type a SQL statement, use the Query Builder, or use data views to create a query interface that insulates the end user from the database structure. In this example, we will use the Query Builder to create a query.

To create a new query, click to expand the *Woodview* node in the left-hand frame of the Data Source Manager. Two sub-nodes will appear, one called *Queries* and one called *Data Views*. Select the *Queries* node and click *Add*. A dialog will appear prompting you to specify a name for the query and to select whether to launch the Query Builder or to enter an SQL statement.



*Query Name Dialog*

Enter any name you would like, select *Open query builder*, and click on *OK*. The Query Builder will launch. You will see a separate window containing all of the tables for Woodview sitting over top of the main Query Builder window.

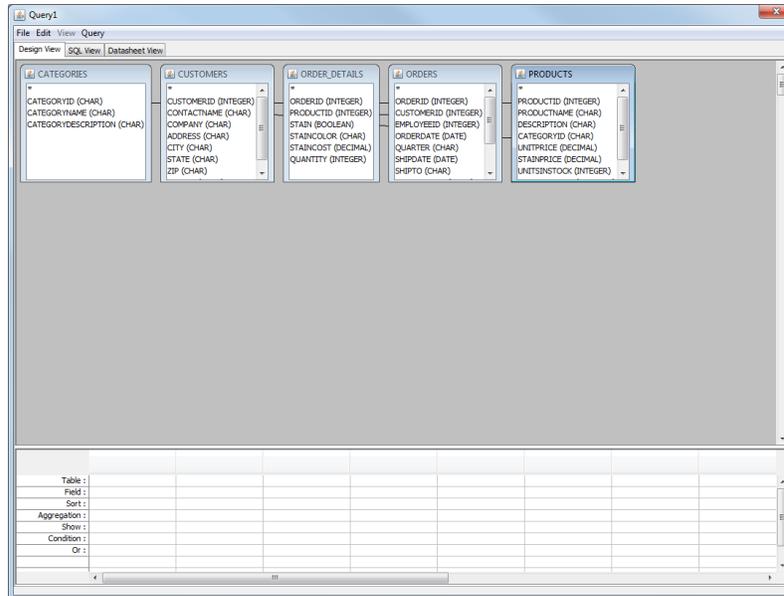


*Query Builder Dialog*

To add a table to the query, select a table in the Tables window and click the *Add* button. You can also double click on the table name. Using one of the two methods, add the following tables to the query:

CATEGORIES  
CUSTOMERS  
ORDER\_DETAILS  
ORDERS  
PRODUCTS

The tables will appear in the top half of the Query Builder window. You will see the join lines connecting various fields in the tables.

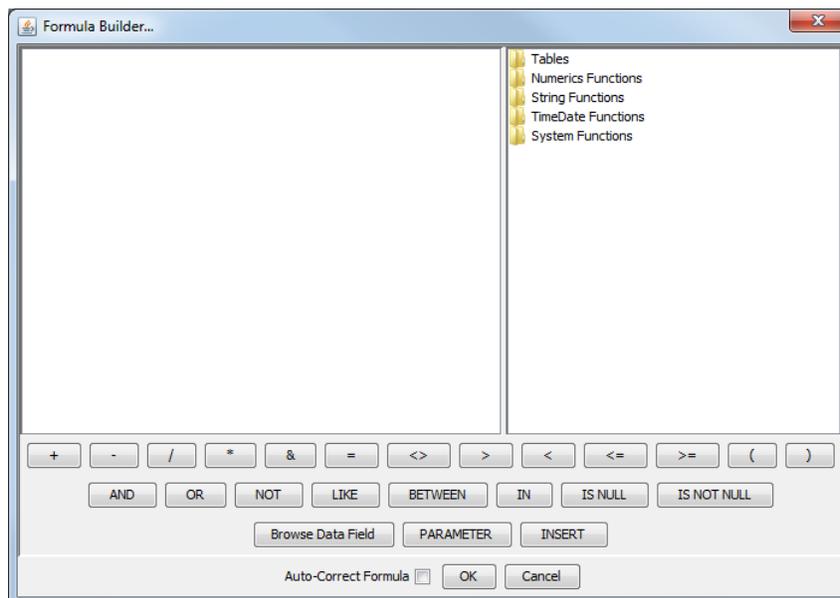


*Query Builder with Tables*

To add a field to the query, you can either double click on the field in the table window or double click on the *Table* and *Field* fields in the lower (QBE) portion of the Query Builder window and select the table and field from the drop-down menu. Using either method, add the following fields to the query.

ORDERID from ORDERS  
 COMPANY from CUSTOMERS  
 REGION from CUSTOMERS  
 CATEGORYNAME from CATEGORIES  
 PRODUCTNAME from PRODUCTS  
 UNITPRICE from PRODUCTS  
 QUANTITY from ORDER DETAILS

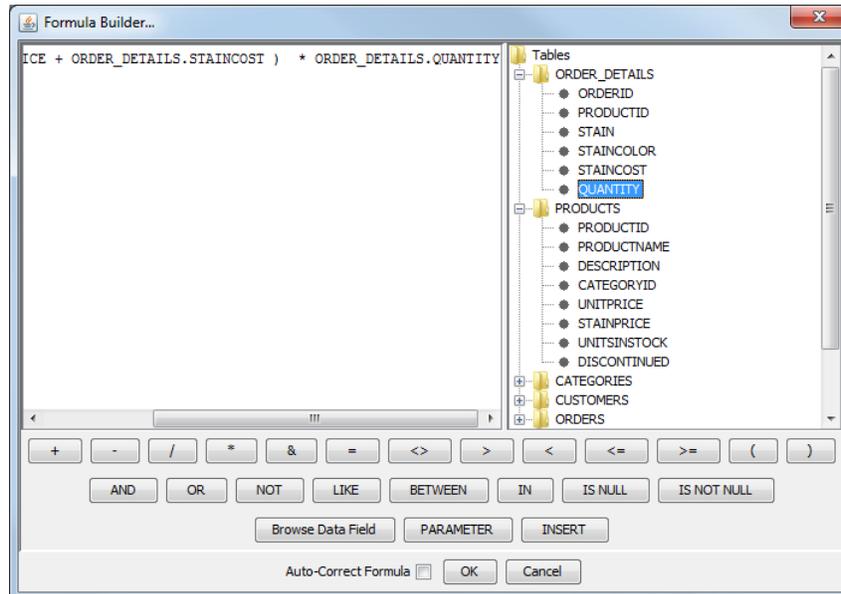
In the eighth column, which should be blank, right click in the *Field* field and select *Build* from the pop-up menu. This will open the Formula Builder interface allowing you to create a computed column.



*Formula Builder Window*

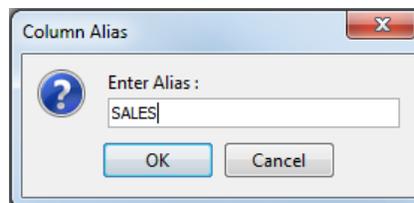
To build a column, first click the *left parenthesis* button. Then double click on the *Tables* folder. It will expand into five nodes, one for each table you selected for the query. Opening a table folder will list all of the column fields in that table. Open up the *PRODUCTS* folder, select *UNITPRICE* and click *Insert*. Then click the *add (+)* button. Next, insert *STAINCOST* from the *ORDER\_DETAILS* table. Then click the *right parenthesis* button. Click the *multiply (\*)* button and finally insert *QUANTITY* from the *ORDER\_DETAILS* column. The finished formula should look like this:

`(PRODUCTS.UNITPRICE + ORDER_DETAILS.STAINCOST) * ORDER_DETAILS.QUANTITY`



*Formula Builder Window with Formula*

Click the *OK* button and the built column will be added to the query. Next, we will give the column you built an alias. Right click on the column and select *Alias* from the pop-up menu. A window will appear asking you to enter a column alias. Enter "SALES" (without quotation marks) and click the *OK* button.



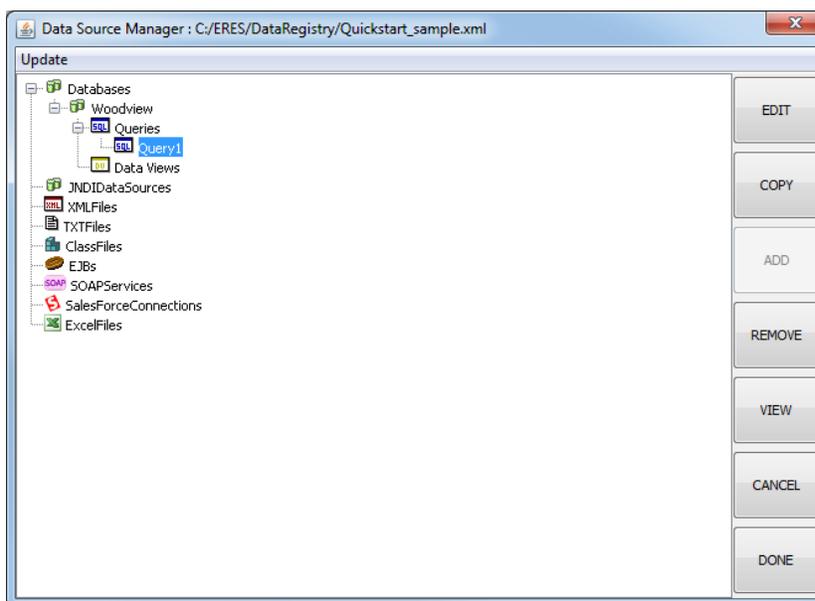
*Column Alias Dialog*

Click the *OK* button and you will see the column name change in the Query Builder. Now click on the *Datasheet View* tab in the Query Builder. Your query will run and you should see the first thirty records of the query results.

ORDERID (INTEGER)	COMPANY (CHAR)	REGION (CHAR)	CATEGORY (CHAR)	PRODUCTNAME (CHAR)	UNITPRICE (DECIMAL)	QUANTITY (INTEGER)	SALES (NUMERIC)
10001	P & S Furniture East	East	Side Chairs	Enfil Chair	450.00	12	5724.00
10001	P & S Furniture East	East	Single Dress...	Ra Dresser	1745.00	12	20940.00
10001	P & S Furniture East	East	Arm Chairs	Shimalya Chair	424.00	14	6440.00
10002	P & S Furniture East	East	Double Dres...	Set Dresser	1645.00	16	26320.00
10002	P & S Furniture East	East	Arm Chairs	Nisaba Chair	414.00	21	8694.00
10003	Alfano Furni...	East	Round Tables	Anzusa Table	1687.00	13	21931.00
10003	Alfano Furni...	East	Side Chairs	Nergal Chair	335.00	4	1340.00
10003	Alfano Furni...	East	Arm Chairs	Zabada Chair	312.00	12	4008.00
10003	Alfano Furni...	East	Arm Chairs	Skoupsuna...	445.00	41	18245.00
10004	P & S Furniture East	East	Round Tables	Apep Table	1587.00	15	23805.00
10004	P & S Furniture East	East	Oval Tables	Neith Table	1798.00	17	30766.00
10004	P & S Furniture East	East	Arm Chairs	Nusku Chair	425.00	22	9350.00
10005	Ebert Furnit...	Midwest	Single Dress...	Salt Dresser	1977.00	15	33665.00
10005	Ebert Furnit...	Midwest	Round Tables	Ningoda Table	1499.00	15	22485.00
10006	Room & Boa...	Midwest	Rectangular...	Bes Table	1141.00	17	23069.00
10006	Room & Boa...	Midwest	Round Tables	Anzusa Table	1687.00	22	43222.00
10007	Alfano Furni...	East	Side Chairs	Nemunag C...	369.00	14	5166.00
10007	Alfano Furni...	East	Arm Chairs	Cula Chair	468.00	16	7488.00
10007	Alfano Furni...	East	Oval Tables	Ma'at Table	1875.00	2	4506.00
10008	Design With...	South	Round Tables	Amon Table	1327.00	21	27867.00
10008	Design With...	South	Oval Tables	Isis Table	1785.00	12	21468.00
10009	Lou Ripponer...	South	Single Dress...	Ra Dresser	1745.00	18	31410.00
10009	Lou Ripponer...	South	Single Dress...	Salt Dresser	1977.00	12	28692.00
10009	Lou Ripponer...	South	Arm Chairs	Nusku Chair	425.00	11	4675.00
10009	Lou Ripponer...	South	Side Chairs	Zabada Chair	312.00	19	5928.00
10010	Munire Furni...	East	Arm Chairs	Ninguru Chair	478.00	4	2052.00
10010	Munire Furni...	East	Double Dres...	Setlet Dresser	1761.00	14	30450.00
10011	Room & Boa...	Midwest	Side Chairs	Ishtar Chair	339.00	18	6102.00
10011	Room & Boa...	Midwest	Side Chairs	Enfil Chair	425.00	18	7650.00
10011	Room & Boa...	Midwest	Side Chairs	Enfil Chair	450.00	18	8100.00

Query Builder Datasheet View

Now that you have finished designing the query, select *Done* from the File menu to save the changes. This will close the Query Builder window and return you to the Data Registry Manager window. There will now be a node under *Queries* for the query you have just designed.



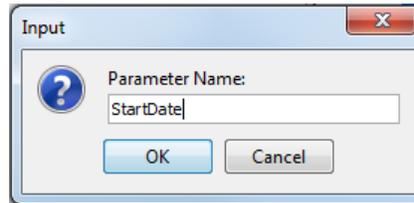
Data Source Manager With Query

### Q.3.1.2.1. Add Query Parameters

ERES allows you to easily parameterize queries, allowing report and chart data to be dynamically filtered at runtime. In this tutorial we will add parameters to the query created in Section Q.3.1.2 - Create a Query.

To open the query that you created, select it, click the *Edit* button in the Data Source Manager, and click the *OK* button. Your query will re-open in the Query Builder. The Tables window will open on top of the Query Builder. Click *Close* to close the Tables window and scroll down in the lower (QBE) portion of the Query Builder window until you see the *Condition* field. Right click in the *Condition* field under the *ORDERID* column and select *Build* from the pop-up menu. This will bring up the Formula Builder allowing you to construct a condition for the query.

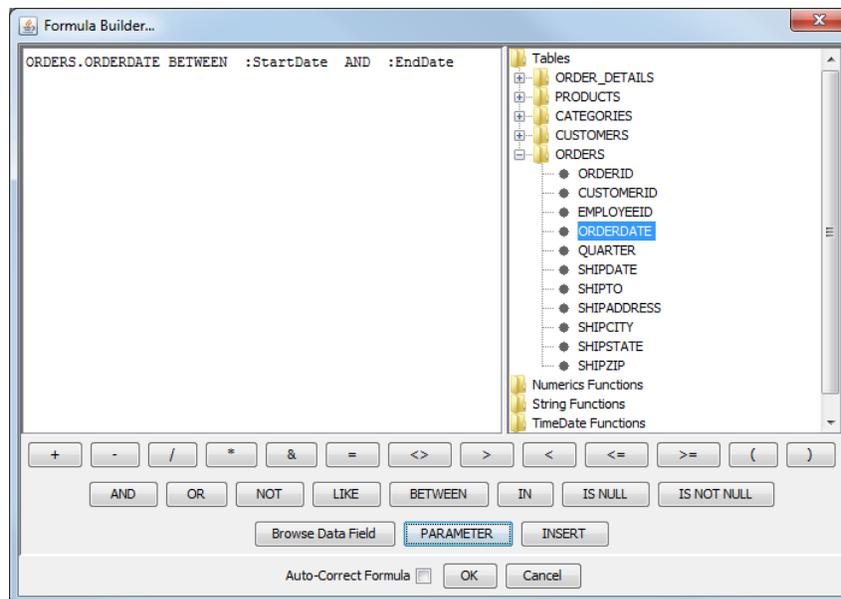
Double click on the *Tables* folder within the Formula Builder to expand it. Then expand the *ORDERS* node and double click on the *ORDERDATE* field. Next click the *Between* button and then click the *Parameter* button. This will bring up a dialog prompting you specify a name for the query parameter.



*Parameter Name Dialog*

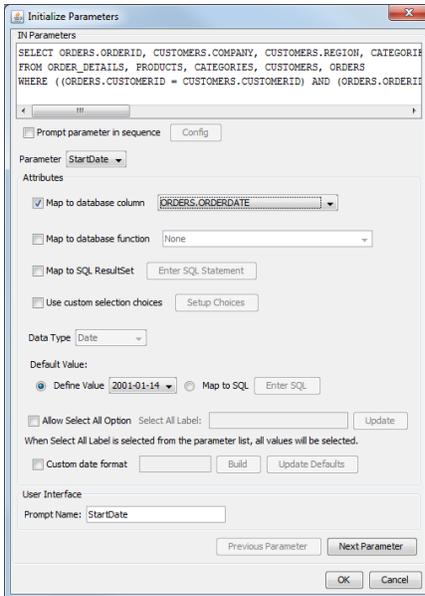
Enter **StartDate** as the parameter name and click the *OK* button. The parameter will be added to the query. Then click the *And* button. Click the *Parameter* button again. Enter **EndDate** as the second parameter name. The finished condition should look like this

`Orders.OrderDate BETWEEN :StartDate AND :EndDate.`



*Formula Builder with Conditions*

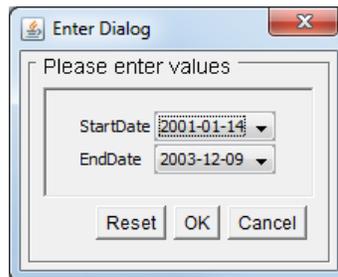
Click *OK* to close the Formula Builder and return to the Query Builder window. Now click on the *Datasheet View* tab. Because you have just added two parameters to the query, an initialization dialog will appear, asking you to specify a few properties for the query parameters.



*Parameter Initialization Dialog*

From this window, click on *Map to a database column* and select *ORDERS.ORDERDATE* from the drop-down menu. This will automatically fill the *Default Value* and *Data Type* options. Next, enter **Start Date** into the *Prompt Name*, then click the *Next Parameter* button and map the *EndDate* parameter to the same column. Click on the *Define Value* drop-down menu to select an end date. Select a date far enough from the start date that by default you will have more than a couple records to work with (this makes report design easier). Change its *Prompt Name* to **End Date**.

Click *OK* to close the initialization window once you have specified all the options. A new dialog will appear prompting you to select a date range by which to filter the result set.



*Parameter Selection Dialog*

Select the Start and End date that you would like and click *OK*. You will now see the filtered result in the datasheet window. Now, click *Done* from the File menu to save the changes you have made to the query.

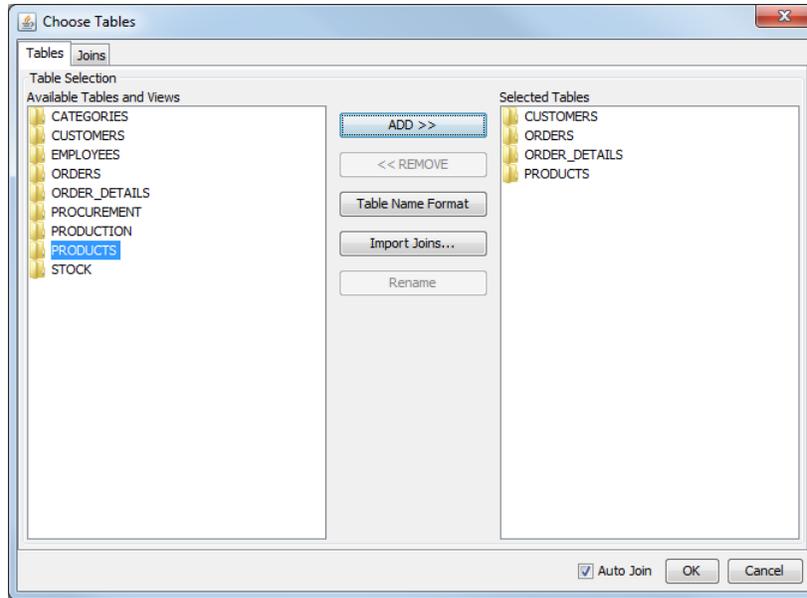
### Q.3.1.3. Create a Data View

A unique feature in ERES is the ability to create data views. Data views are local schemas/views that allow an administrator to pre-configure a group of tables and fields, so that the end users only need to select fields and define simple conditions to create a query. Data views are also used with the ad hoc QuickDesigner. To create a data view, select the *Data Views* node under *Woodview* and click *Add*.

This will open a new dialog asking you to select database tables you would like to use. Select the following tables and add them to the *Selected Tables* panel by clicking on the *ADD >>* button:

CUSTOMERS  
ORDERS  
ORDER\_DETAILS

## PRODUCTS

*Data View Tables Dialog*

Next, click on the *Joins* tab. You will see a representation of the tables like in the Query Builder. You can see the auto-join lines between the tables. This window can be used to join the tables or modify the auto-joins if necessary (for more about joins please see Section 3.1.3.2.1.2 - Joins). Click *Ok* to finalize the table selection. The next window allows you to select and group fields for the view. At the top of the window you can specify a name for the view. Name it **Invoicing**.

Next, double click on the *CUSTOMERS* folder to reveal the fields for that table. Add the following fields by double clicking them or selecting them and clicking on the *Add >>* button:

COMPANY  
CONTACTNAME  
ADDRESS  
CITY  
STATE  
ZIP

Now add fields from the other tables as follows:

ORDERS :

ORDERDATE  
SHIPDATE  
SHIPTO  
SHIPADDRESS  
SHIPCITY  
SHIPSTATE  
SHIPZIP

ORDER\_DETAILS :

ORDERID  
STAIN  
STAINCOLOR  
QUANTITY

PRODUCTS :

PRODUCTNAME  
UNITPRICE  
STAINPRICE

Now click the *Add Heading* button. At the prompt, specify the name **Customer Info**. Add two more headings in the same way, one called **Shipping Info** and one called **Order Info**. Once they are created, select the following fields (Using **CTRL+Click** or **SHIFT+Click** for multiple selection):

COMPANY  
CONTACTNAME  
ADDRESS  
CITY  
STATE  
ZIP

Once the fields are selected, click the *Group Fields* button and select *Customer Info* from the drop-down list. The fields will be moved under that heading. Next, select the following fields in the same way:

SHIPTO  
SHIPADDRESS  
SHIPCITY  
SHIPSTATE  
SHIPZIP  
SHIPDATE

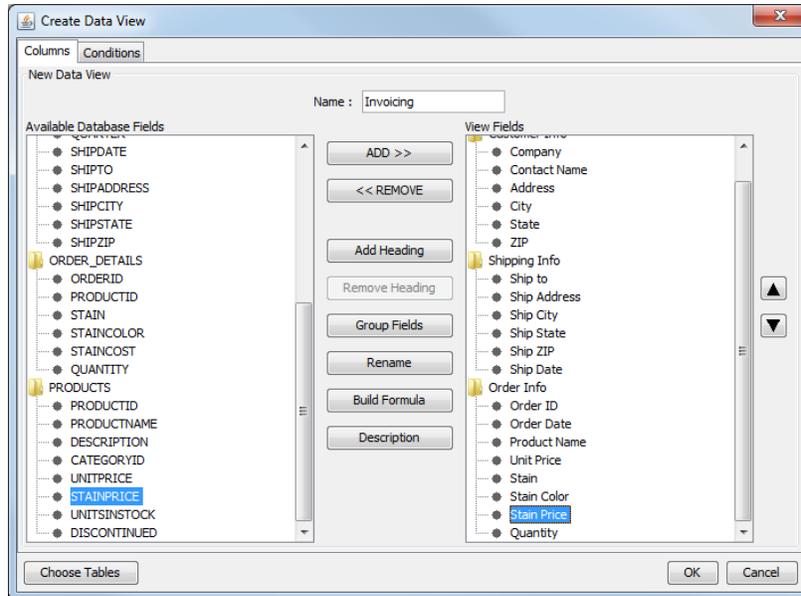
Add these fields to the *Shipping Info* group the same way as before. Next, select the following fields:

ORDERDATE  
ORDERID  
STAIN  
STAINCOLOR  
QUANTITY  
PRODUCTNAME  
UNITPRICE  
STAINPRICE

Add these fields to the *Order Info* group. Next, select the *CONTACTNAME* field on the right side and click the *Rename* button. In the dialog, specify the name **Contact Name**. Repeat this for every field in order to give it proper names.

Next, select the *Order ID* field on the right side and click on the up arrow button to move the field to the top of the *Order Info* heading. Use the arrows to arrange the items in the *Order Info* heading in the following order:

Order ID  
Order Date  
Product Name  
Unit Price  
Stain  
Stain Color  
Stain Price  
Quantity



*Data View Fields Window*

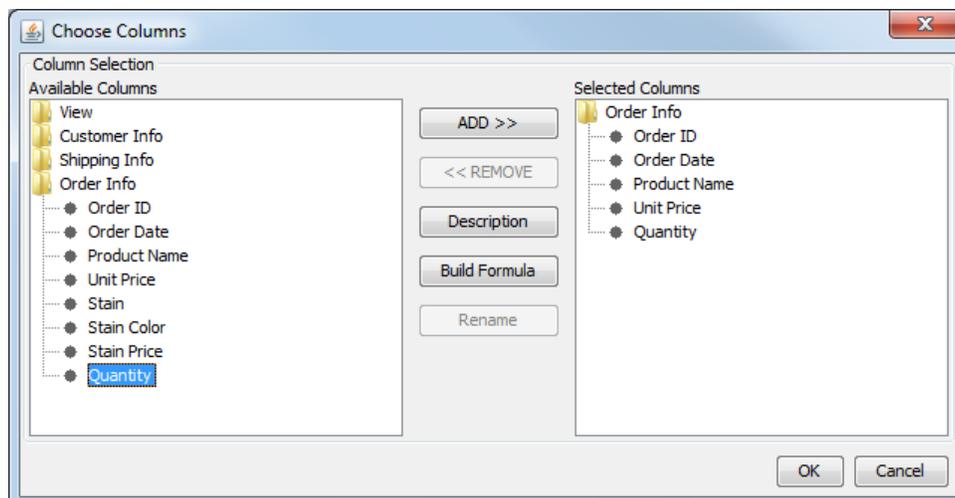
Now that the data view has been created, click the *OK* button in the fields window to save the view. It will be saved as a new node under *Data Views* in the Data Source Manager.

### Q.3.1.3.1. Query a Data View

Now that the data view has been created, you can write queries against the view. This allows users to develop queries without knowing the underlying structure of the database. It also allows administrators to limit which database elements the user has access to. In this tutorial we will create a query for the data view you created in. Section Q.3.1.3 - Create a Data View.

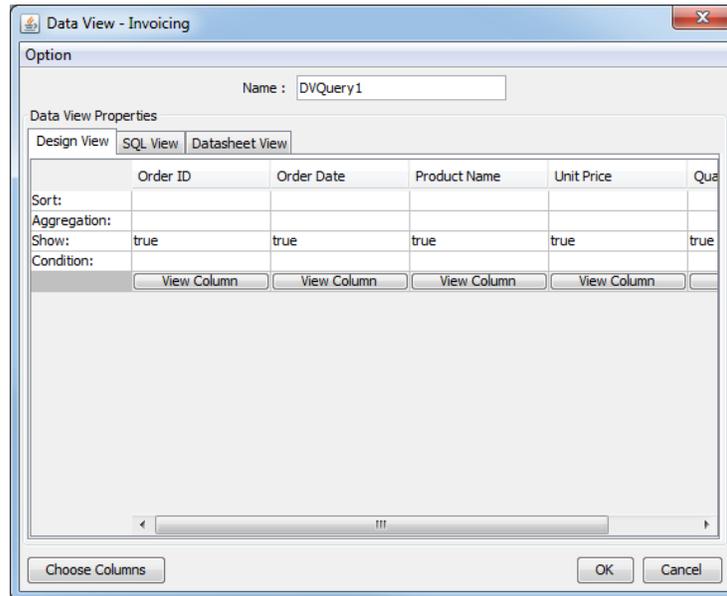
In the Data Source Manager, select the *Invoicing* data view. Then click the *View* button. This will open a dialog, prompting you to select fields from the view. To select fields, first double click on a heading to expand it. Add the following fields to the query by double clicking or selecting them and clicking on the *ADD* button:

Order ID  
 Order Date  
 Product Name  
 Unit Price  
 Quantity



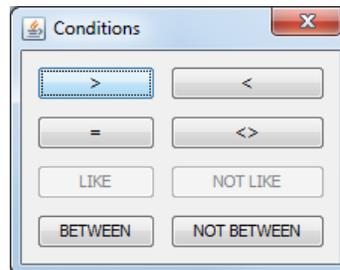
*Data View Query Field Selection Dialog*

Once you have finished adding the fields, click *OK*. This will bring up a new window allowing you to set conditions, grouping, and ordering for the query. Like the Query Builder this window also allows you to preview the query result with the *Datasheet View* tab.



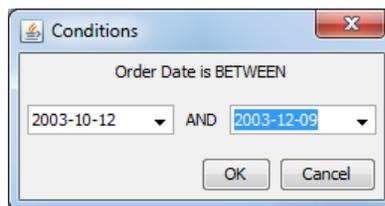
*Data View Conditions Dialog*

First specify a name for the query in the space provided at the top. Then double click on the *Condition* field for the *Order Date* column. This will bring up a dialog allowing you to specify a condition for the field.



*Specify Condition Dialog*

Click the *Between* button. A new dialog will appear prompting you to specify a start and end date with which to filter the results. Select **2003-10-12** as the first date and **2003-12-09** as the second.



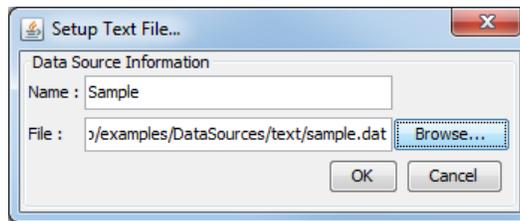
*Specify Condition Fields Dialog*

Click *OK* to close the dialog and add the condition. You will be taken back to the conditions window. Now you can click on the *Datasheet View* tab to preview the query. Click *OK* in the main window and the query will be saved using the name you provided. You will then be taken to a screen showing the result of the data view query. Options in this screen allow you to use the results to create a report or chart. Click *Cancel* in this dialog to close the interface and return to the Data Source Manager. There will now be a new node for your query under the *Invoicing* data view.

### Q.3.1.4. Setup a Text Data Source

In addition to database data, ERES can also draw data from flat files(XML and text). In this tutorial we will setup a text file data source in the registry. To add a new text file, select the node labeled *TXTFiles* and click the *ADD* button. This will bring up a dialog allowing you to specify a display name and the location for the text data source.

Enter any display name you would like (e.g. *Sample*). Then click the *Browse* button and browse to the `help/examples/DataSources/text` directory. Select the `sample.dat` file.



*Setup Text File Dialog*

After you have finished entering the information, click *OK*. You will see a new node in the Data Source Manager for the text file.

## Q.4. Report Designer

After setting up data sources and creating queries, the next step to create reports is to take the results of the data source and map them to the report. Depending on the type of report being created, the mapping options will vary. This section only covers basic mapping for certain report layout. Once you will finish data mapping for your report, you will be taken to the Report Designer interface where you can customize many different features/properties of the report. Here in QuickStart, we will use only some of the most commonly used features in the Report Designer. For more information, please see Section 4.1 - Report Designer.

### Q.4.1. Report Mapping

This section looks at several ways data can be mapped to the table structure of reports. This section will use the query created in Section Q.3.1.2 - Create a Query as the data source for the report.

#### Q.4.1.1. Simple Columnar Layout

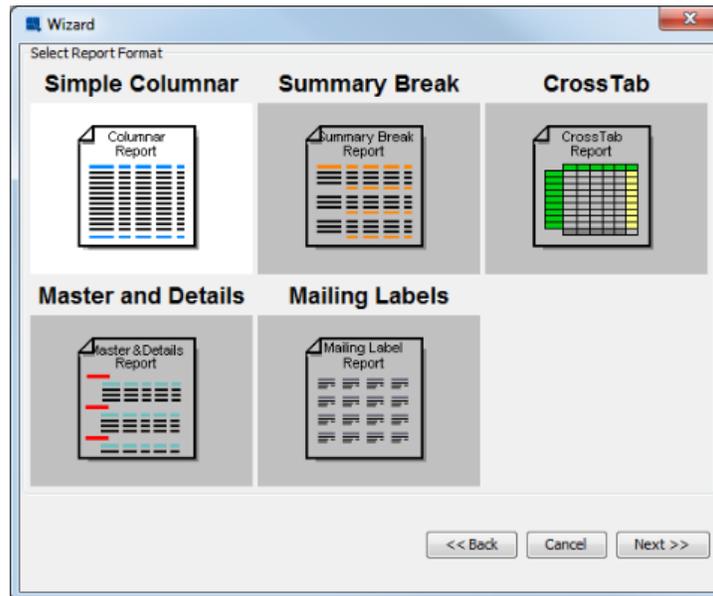
The simple columnar layout is the most straight forward type of report mapping. Columns from the data source are drawn in a straight table in the report without any grouping and breaks.

To begin creating a report, open your data registry and select the node for your query. Click the *VIEW* button. A new window will open, with a table containing the results of the query (first 20 records only). Notice that because your query contains a parameter, it initially runs with the default values that were specified when the parameters were initialized.

INDEX	TYPE	ORDERID	COMPANY	REGION	CATEGOR...	PRODUCT...	UNITPRICE	Q
1	Integer	10001	P & S Furniture East	East	Side Chairs	Enlii Chair	450.00	12
2	String	10001	P & S Furniture East	East	Single Dressers	Ra Dresser	1745.00	12
3	String	10001	P & S Furniture East	East	Arm Chairs	Shimaliya Chair	424.00	14
4	String	10002	P & S Furniture East	East	Double Dres...	Set Dresser	1645.00	16
5	String	10002	P & S Furniture East	East	Arm Chairs	Nisaba Chair	414.00	21
6	String	10003	Alfano Furnit... East	East	Round Tables	Anubis Table	1687.00	13
7	String	10003	Alfano Furnit... East	East	Side Chairs	Nergal Chair	335.00	4
8	String	10003	Alfano Furnit... East	East	Side Chairs	Zabada Chair	312.00	12
9	String	10003	Alfano Furnit... East	East	Arm Chairs	Sbuqamuma ...	445.00	41
10	String	10004	P & S Furniture East	East	Round Tables	Apep Table	1587.00	15
11	String	10004	P & S Furniture East	East	Oval Tables	Neith Table	1798.00	17
12	String	10004	P & S Furniture East	East	Arm Chairs	Nusku Chair	425.00	22
13	String	10005	Ebert Furnit... Midwest	Midwest	Single Dressers	Sati Dresser	1977.00	15
14	String	10005	Ebert Furnit... Midwest	Midwest	Round Tables	Ningizida Table	1499.00	15
15	String	10006	Room & Boar... Midwest	Midwest	Rectangular ...	Bes Table	1141.00	17
16	String	10006	Room & Boar... Midwest	Midwest	Round Tables	Anubis Table	1687.00	22
17	String	10007	Alfano Furnit... East	East	Side Chairs	Ninhursag C...	369.00	14
18	String	10007	Alfano Furnit... East	East	Arm Chairs	Cula Chair	468.00	16
19	String	10007	Alfano Furnit... East	East	Oval Tables	Ma'at Table	1875.00	2
20	String	10008	Design Withi... South	South	Round Tables	Amon Table	1327.00	21

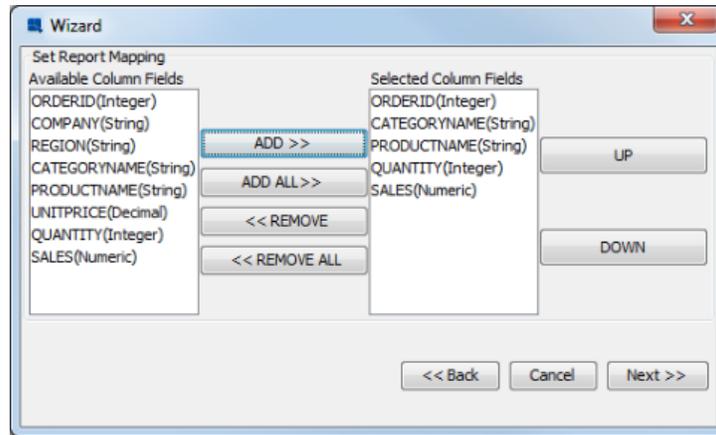
*Query Result Screen*

The bottom of the screen contains two buttons, *Create Chart* and *Create Report*, which allow you to use the query to design a chart or report. To continue designing the report, click *Create Report*. This will bring up a dialog asking you which report layout option you would like to use.



*Select Report Layout Dialog*

From this dialog, select *Simple Columnar* as the layout and click the *Next* button. The next step in the Report Wizard allows you to select which columns from the query you would like to use in the report, as well as re-arrange the column order. (Although the column order is not particularly important for the simple columnar layout, it can have a significant impact on other layouts depending on the mapping options selected).

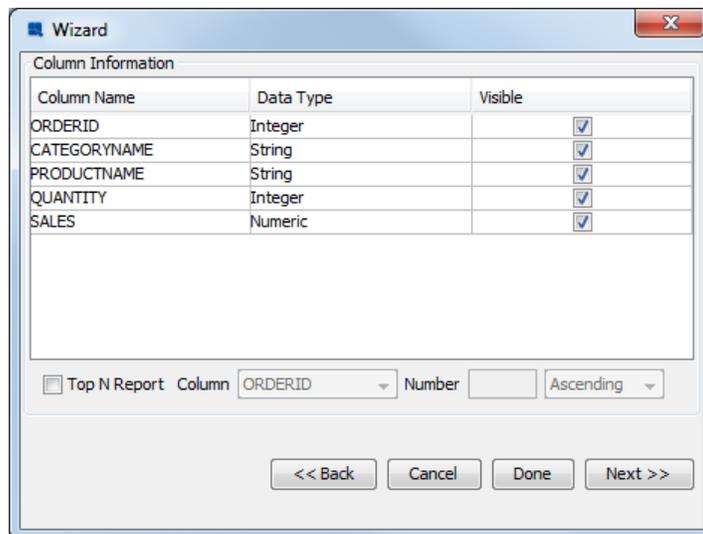


*Column Selection/Ordering Dialog*

In this dialog, select the following fields for the report:

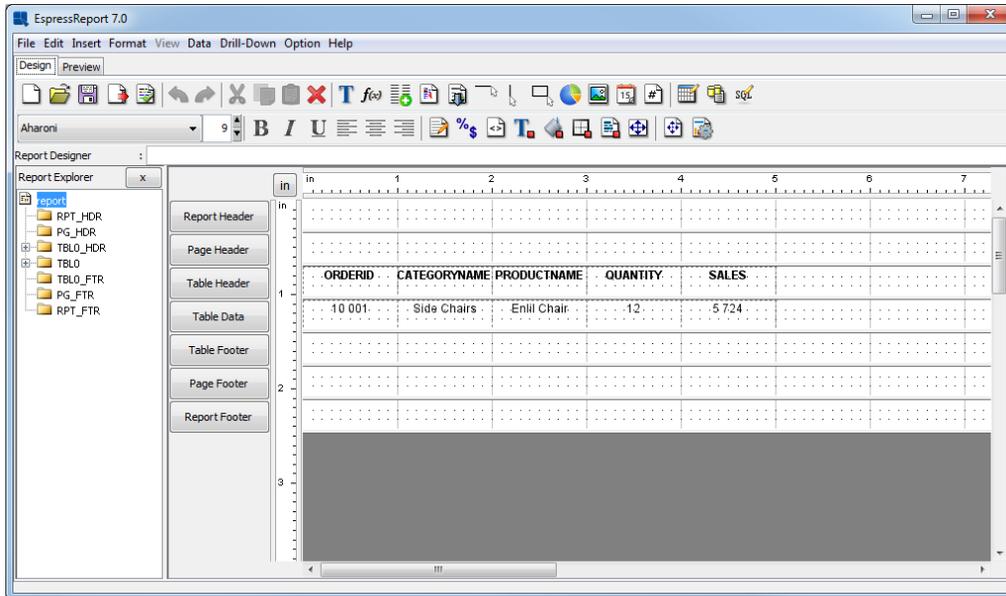
ORDERID  
 CATEGORYNAME  
 PRODUCTNAME  
 QUANTITY  
 SALES

Once you have selected the fields, click *Next* to continue on in the Wizard. The next dialog is the data mapping dialog. This is where you can select how to map the fields from the data source into the selected report layout. This is a simple columnar layout, so the only options are whether to set columns as visible or not and whether to generate a top N presentation.



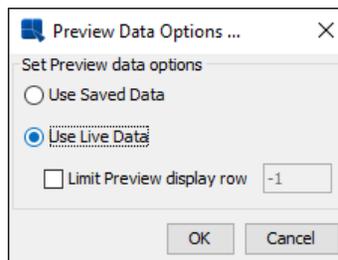
*Data Mapping Dialog*

In this dialog, select to leave all the columns visible and do not select the top N option. (For more about top N presentations, please see Section 4.1.2.1.1.1 - Top N Report). Next click the *Done* button. (There are some optional additional steps in the Wizard which will be explained later in this section). This will bring up the Report Designer interface with an unformatted version of your report.



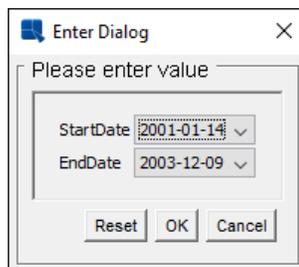
*Simple Columnar Report in Design Window*

Now click on the *Preview* Tab in the upper-left hand corner of the window. A dialog will open asking you whether you would like to preview the report with live or saved data.



*Preview Data Options*

Select the *Live Data* option and click *OK*. A parameter selection dialog will appear prompting you to select a start and end date by which to filter the report.



*Parameter Value Selection*

Specify a range that is large enough to produce enough records and click *OK*. You will then see the report output. Notice how the simple columnar layout places the columns from the result set in the report directly, without any grouping, sorting, or summaries.

ORDERID	CATEGORYNAME	PRODUCTNAME	QUANTITY	SALES
10 001	Side Chairs	Enlll Chair	12	5 724
10 001	Single Dressers	Ra Dresser	12	20 940
10 001	Arm Chairs	Shimaliya Chair	14	6 440
10 002	Double Dressers	Set Dresser	16	26 320
10 002	Arm Chairs	Nisaba Chair	21	8 694
10 003	Round Tables	Anubis Table	13	25 363
10 003	Side Chairs	Nergal Chair	4	1 340
10 003	Side Chairs	Zabada Chair	12	4 008
10 003	Arm Chairs	Sbuqamuma	41	18 245
10 004	Round Tables	Apep Table	15	23 805
10 004	Oval Tables	Neith Table	17	35 870

*Simple Columnar Report Preview*

### Q.4.1.2. Summary Break Layout

The summary break layout is similar to the columnar layout, except it adds the ability to group and aggregate the report columns. A summary break report must be grouped by at least one column. Using the report created in Section Q.4.1.1 - Simple Columnar Layout, we will convert it into a summary break layout.

Go to the design tab and select the *Change Data Mapping* Icon on the toolbar . This will return you to the Report Wizard, where we will select a different report layout. From the last data mapping dialog, hit the *Back* button twice until you get back to the *Select Report Format* window.

Change the report layout type to *Summary Break* and click *Next*. In the next screen, keep the column selection the same and click *Next* again to go to the data mapping window. You will notice that there are more options in this window than there were for the columnar layout.

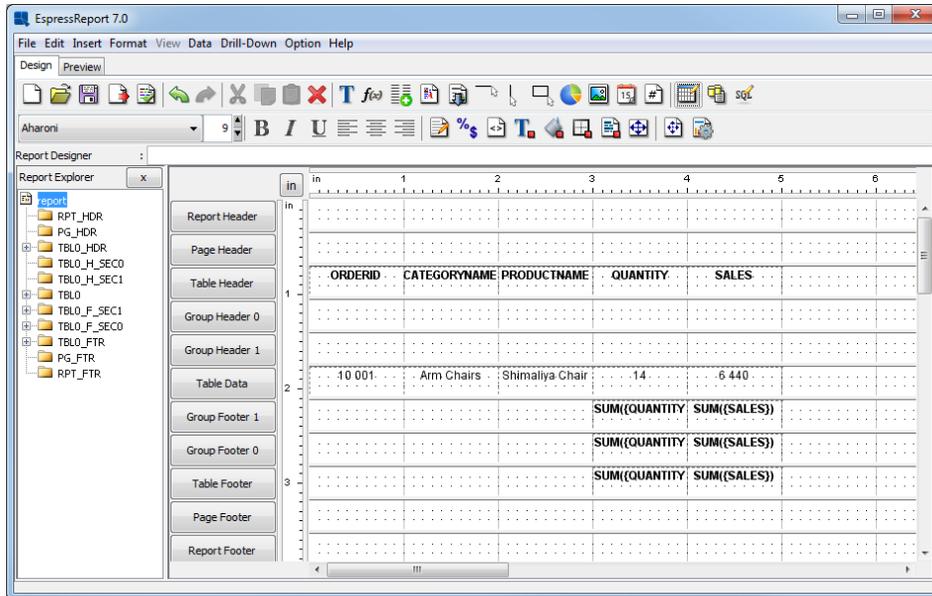
Column Name	Data Type	Visible	Row Break	Aggregation	Repeat break field
ORDERID	Integer	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	NONE	[ ]
CATEGORYNAME	String	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	NONE	[ ]
PRODUCTNAME	String	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	NONE	[ ]
QUANTITY	Integer	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	SUM	[ ]
SALES	Numeric	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	SUM	[ ]

Perform Column Aggregation  
 Keep Data Source Order  
 Top N Report: Column: ORDERID, Number: [ ], Ascending: [ ]

Apply Template  Apply Formula And Script  << Back Cancel Done Next >>

*Data Mapping Screen for Summary Break Layout*

In the data mapping dialog, check the option called *Row Break* for the first two columns. This will have the report group by those two columns. From the drop-down menus under *Aggregation*, select to *SUM* the *QUANTITY* and *SALES* columns. Also, uncheck the *Apply Template* option, as you do not need to carry over the formatting from the simple columnar layout. Once you have finished specifying options, again click the *Done* button. You will get a warning if you have not applied template. Click *Yes* to continue. You will be taken back to the Report Designer where the new mapping has taken effect.



*Summary Break Report in Design Window*

In the Designer you will notice that because there are two levels of nested grouping in the report there are now corresponding Group Header and Footer sections for each. For more about report sections and their behavior, please see Section 4.1.3.1 - Report Sections. Now click on the *Preview* tab to preview the report. Again, you will be prompted to specify parameter values. Once you see the report in the preview window, notice that the data is grouped by category name and order ID and that intermediate summaries are calculated for each group.

The screenshot shows the EspressoReport 7.0 Preview Window. The report is rendered with a dark border and a white background. The data is grouped by ORDERID and then by CATEGORYNAME. The table shows the following data:

ORDERID	CATEGORYNAME	PRODUCTNAME	QUANTITY	SALES
10 001	Arm Chairs	Shimaliya Chair	14	6 440
			<b>14</b>	<b>6 440</b>
	Side Chairs	Enlil Chair	12	5 724
			<b>12</b>	<b>5 724</b>
	Single Dressers	Ra Dresser	12	20 940
			<b>12</b>	<b>20 940</b>
		<b>38</b>	<b>33 104</b>	
10 002	Arm Chairs	Nisaba Chair	21	8 694
			<b>21</b>	<b>8 694</b>
	Double Dressers	Set Dresser	16	26 320
			<b>16</b>	<b>26 320</b>
		<b>37</b>	<b>35 014</b>	

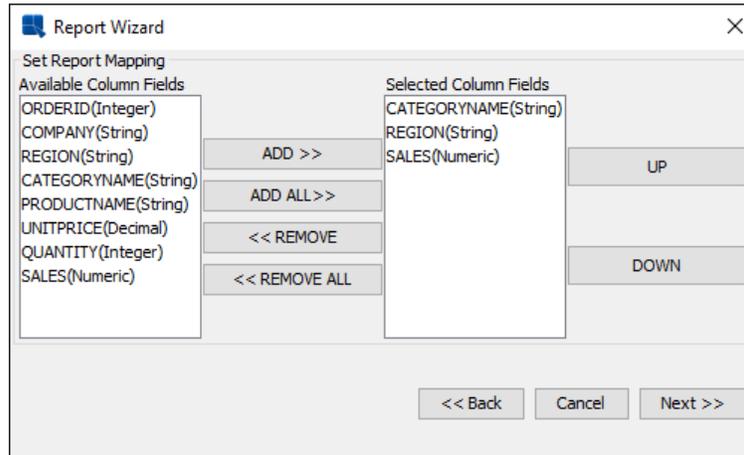
*Summary Break Report Preview*

### Q.4.1.3. Crosstab Layout

A crosstab report shows data in a matrix-like form, allowing multi-dimensional data to be displayed in a two-dimensional layout. In the current example, we will use the crosstab layout to breakdown the sales column by product category and region.

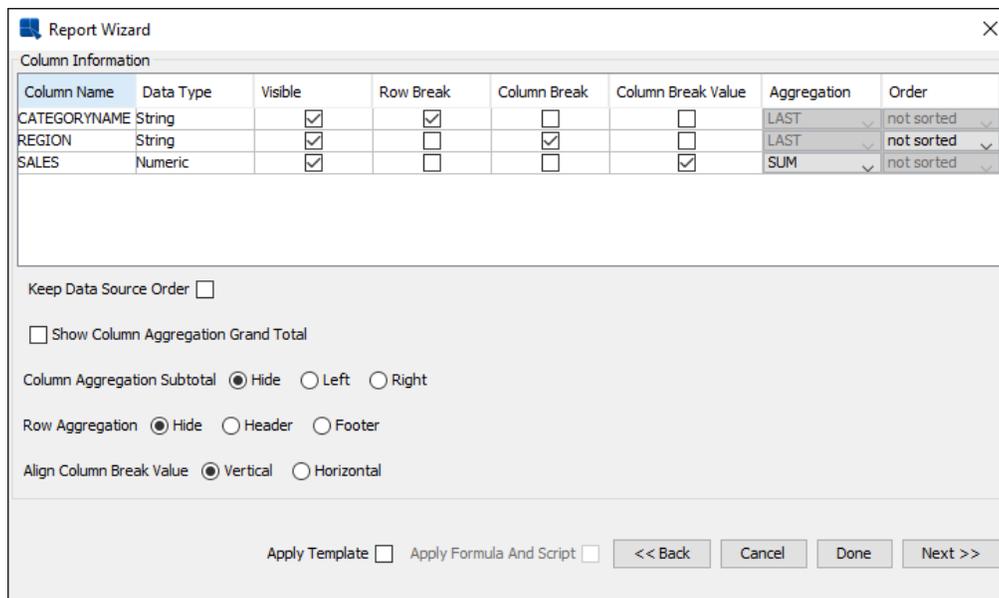
Go to the design window and select *Change Data Mapping* . When the Report wizard re-opens, click the *Back* button until you get back to the layout selection screen. From this screen, select to use a *CrossTab* layout and click *Next*. At the column selection/ordering screen (next in the Wizard), change the selection to be the following:

CATEGORYNAME  
 REGION  
 SALES



*Column Selection for Crosstab Report*

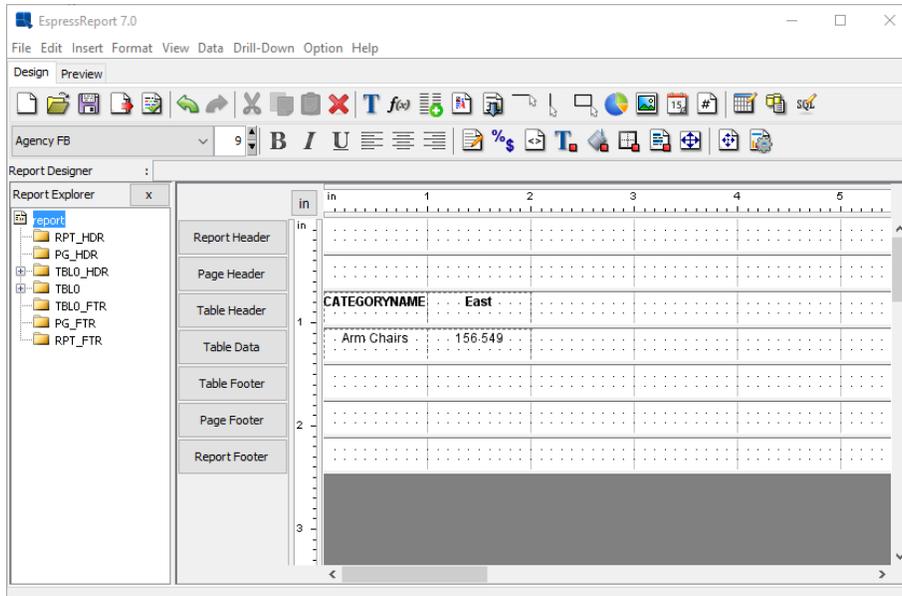
Once you have specified the columns in the correct order, click the *Next* button to bring up the data mapping option for the crosstab layout.



*Data Mapping Screen for Crosstab Layout*

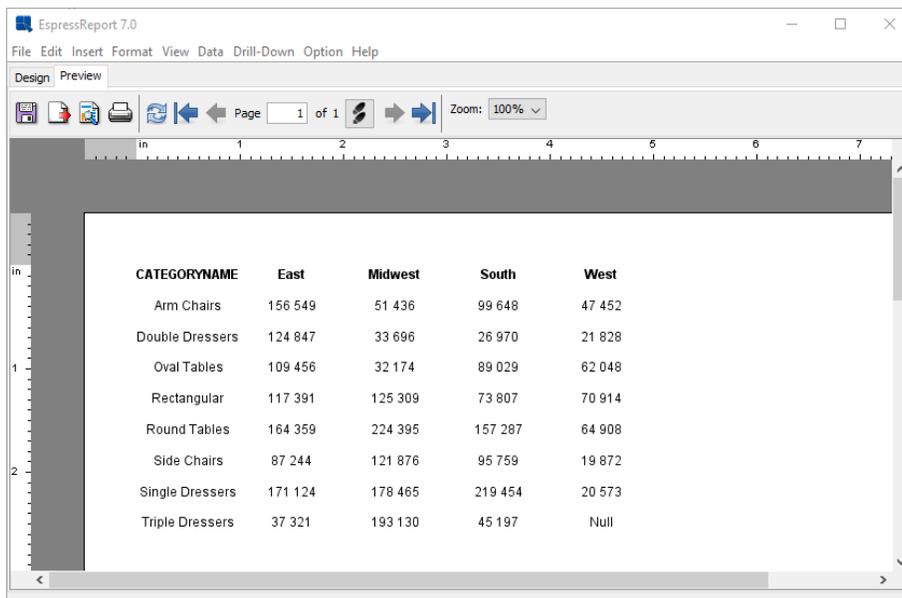
For the CATEGORYNAME column, check the option marked *Row Break*. This will create a row in the report data for each distinct category name. Next select the *Column Break* option for the REGION column. This will create a column in the report data for each distinct region. Leave the *Order* option as *not sorted*. Finally, select the *Column Break Value* option for the SALES column and click on the *Aggregation* menu to select *SUM*. This will give you the total sales for each category and region in the report.

Once you have finished specifying the options, click the *Done* button to go to the Report Designer. You will get a warning if no template was applied. Click *Yes* to continue.



*Crosstab Report in Design Window*

Now click on the *Preview* tab to preview the crosstab layout. As you can see a report column has been generated for each region, and has been automatically totaled both vertically (columns) and horizontally (rows).



*Crosstab Report Preview*

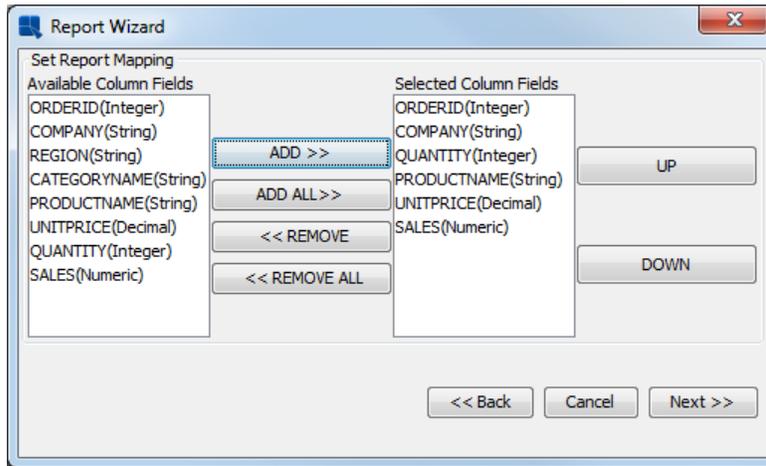
#### Q.4.1.4. Master & Details Layout

Like the summary break layout, the master & details layout also allows you to group the data. It also allows you to automatically add column fields to the Group Header section, creating a one to many layout that can also be configured in a side-by-side layout.

Go back to the design window and select *Change Data Mapping* from the Data menu. Again, navigate back to the layout selection screen. This time select the *Master and Details* layout. Click *Next* to get to the column selection screen. In the column selection screen, change the selection to include the following columns:

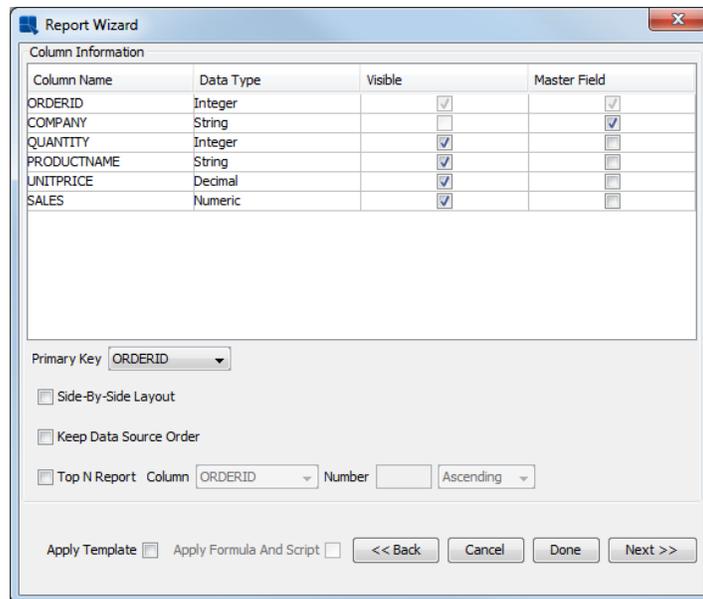
ORDERID  
COMPANY  
QUANTITY

PRODUCTNAME  
UNITPRICE  
SALES



*Column Selection for Master & Details Report*

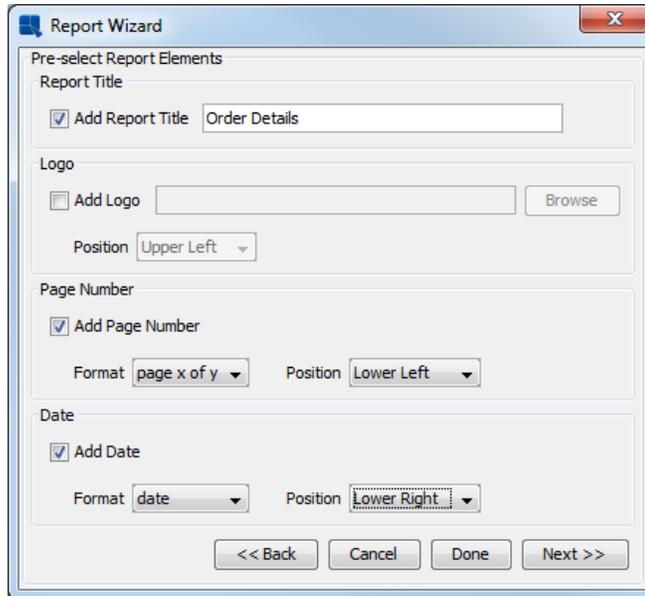
Once you have specified the columns in the correct order, click the *Next* button to bring up the data mapping dialog for the master & details layout.



*Data Mapping Screen for Master & Details Layouts*

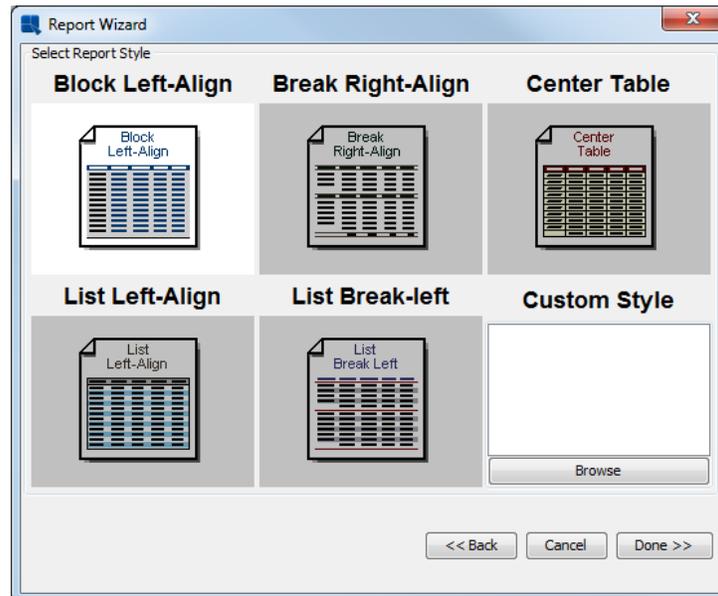
Select the ORDERID field as the *Primary Key* from the drop-down menu in the lower-left hand portion of the screen. This will group the data by the ORDERID field. Next, check the *Master Field* option for the COMPANY column. This will place the COMPANY field in the Group Header section of the report.

Instead of clicking *Done* to get to the Report Designer from this screen, click the *Next* button to invoke the additional pre-formatting options. You will get warning that all the formatting will be lost. Click *Yes* to continue to the dialog that allows you to add several elements to the report.



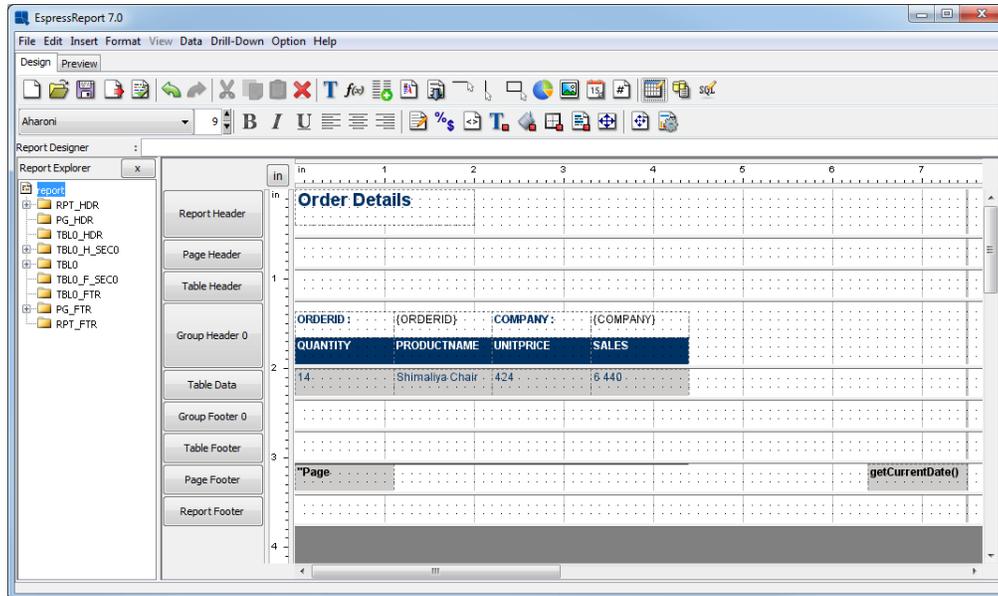
*Add Report Elements Dialog*

Check the boxes to add a *Report Title*, *Page Number* and *Date*. Enter the text you would like as the report title and specify the format and location of the date and time using the drop-down boxes for each option. Once you have finished setting the options, click *Next*. A new dialog will open allowing you to specify a style for the new report.



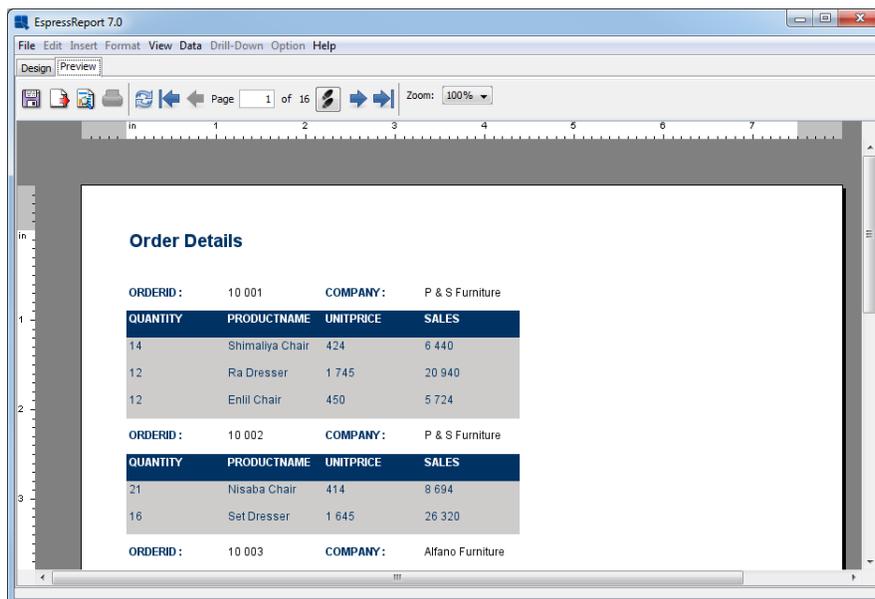
*Report Style Selection*

Select the *Block Left-Align* style and click the *Done* button. You will be taken to the Report Designer window, where the elements have been added. There is now some default formatting applied to the report.



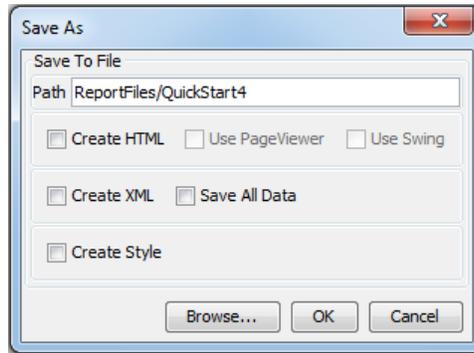
*Master & Details Report in Design Window (with pre-formatting)*

As you can see the ORDERID and COMPANY fields have been generated in the Group Header section. Now preview the report, and you will see a group with each order, as well as the page number and date in the section (header or footer) in which they were placed.

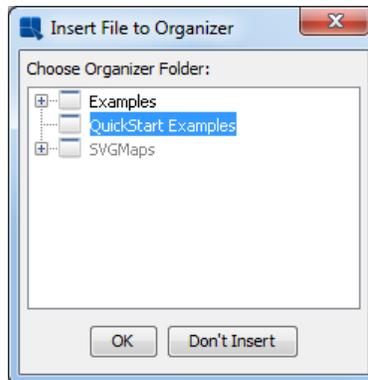


*Master & Details Report in Preview Window*

Now that you have finished creating a report, click the **Save** button on the toolbar to save the report . This will bring up a new dialog, allowing you to specify save options for the report.

*Save As Dialog*

Enter a name for the report and click *Ok*. By default the report will be saved in the `/ReportFiles/` directory under the ERES install directory. A dialog will open, asking you if you would like to insert the report in the Organizer, and what folder do you want to choose.

*Save to Organizer Prompt*

Click *Yes* and the report will be added to your project in the Organizer.

### Q.4.1.5. Mailing Labels Layout

A mailing labels layout is similar to a simple columnar layout; however, it prearranges data in a manner that allows you to create mailing labels.

Close the design window and go back to the data registry to start designing a report. Navigate to *Invoicing* data view. Click the *View* button to create a new dataview query. Select the following fields from *shipping info* to the query:

Ship To  
 Ship Address  
 Ship City  
 Ship State  
 Ship Zip

Specify a name for the query, then click *OK*. The query result show up:

Wizard

Query Result

INDEX	Ship To	Ship Address	Ship State	Ship City	Ship Zip
TYPE	String	String	String	String	String
1	P & S Furniture	49 Main Street	NH	Littleton	03561
2	Furniture Ro...	119 North 7...	NE	Omaha	68114
3	Alfano Furnit...	17 Memorial ...	NJ	Patterson	07505
4	Hive Modern...	820 Northwe...	OR	Portland	97209
5	Ebert Furnit...	3602 Broad...	PA	Allentown	18104
6	Room & Boar...	55 East Ohio...	IL	Chicago	60611
7	Alfano Furnit...	17 Memorial ...	NJ	Patterson	07505
8	Design With ...	200 West 2n...	TX	Austin	78701
9	Lou Rippners...	5025 Bloomf...	KS	Jefferson	70121
10	Munire Furni...	91 New Engl...	NJ	Piscataway	08854
11	Room & Boar...	55 East Ohio...	IL	Chicago	60611
12	Ebert Furnit...	3602 Broad...	PA	Allentown	18104
13	Design With ...	200 West 2n...	TX	Austin	78701
14	Alfano Furnit...	17 Memorial ...	NJ	Patterson	07505
15	Munire Furni...	91 New Engl...	NJ	Piscataway	07505
16	Lou Rippners...	5025 Bloomf...	KS	Jefferson	70121
17	Room & Boar...	55 East Ohio...	IL	Chicago	60611
18	Lou Rippners...	5025 Bloomf...	KS	Jefferson	70121
19	P & S Furniture	49 Main Street	NH	Littleton	03561
20	Alfano Furnit...	17 Memorial ...	NJ	Patterson	07505

Transpose Data    Show All Records        

*DataView Query Result for Creating Mailing Labels Report*

click the *Create Report* button and select the *Mailing Labels* layout. The next screen is to select columns to the report, click *ADD ALL*:

Wizard

Set Report Mapping

Available Column Fields	Selected Column Fields
Ship To(String)	Ship To(String)
Ship Address(String)	Ship Address(String)
Ship State(String)	Ship State(String)
Ship City(String)	Ship City(String)
Ship Zip(String)	Ship Zip(String)

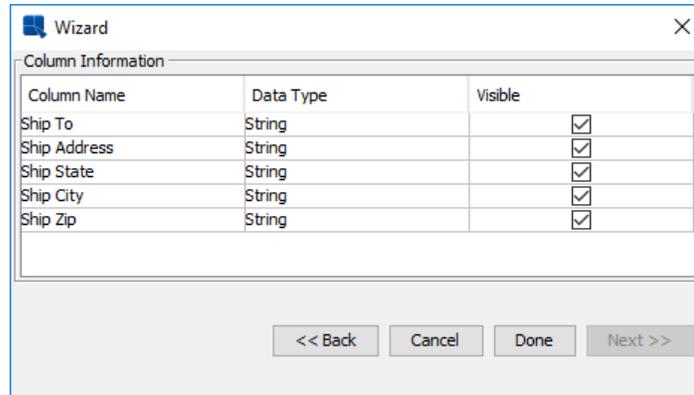
 

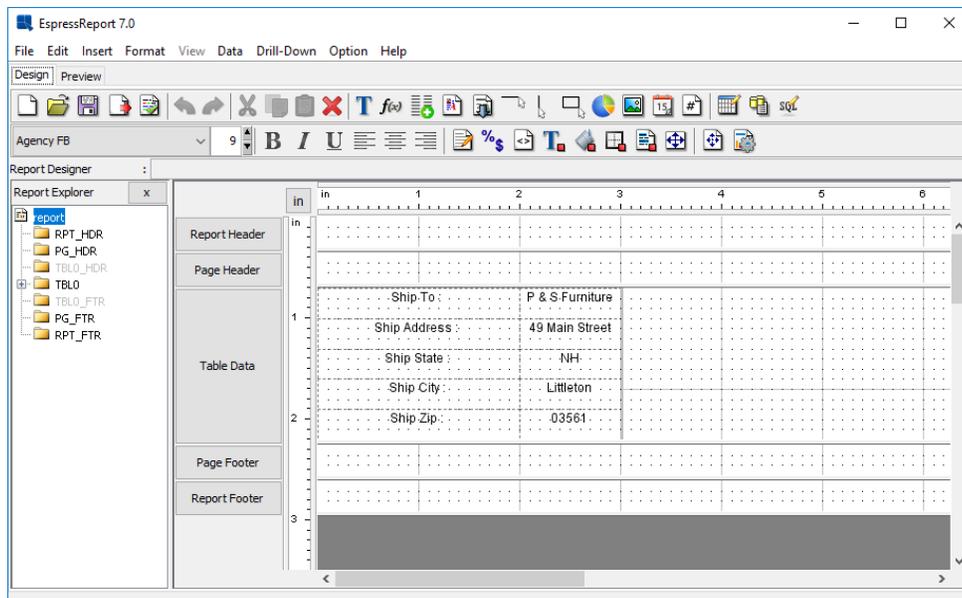
*Column Selection for Mailing Labels Report*

Once you have specified the columns in the correct order, click the *Next* button to bring up the data mapping dialog for the mailing labels layout.



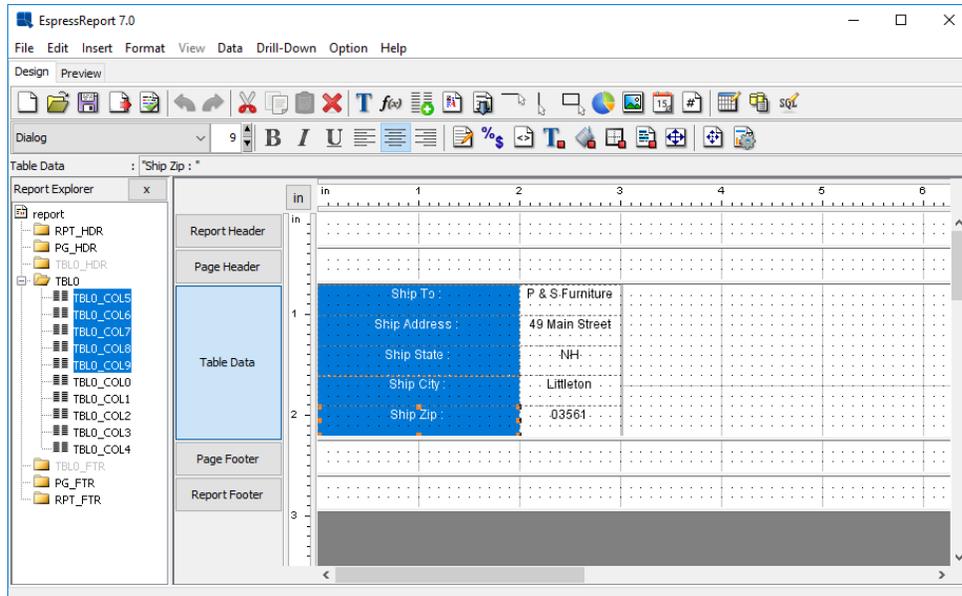
*Data Mapping Screen for Mailing Labels Layouts*

Click the *Done*, You will be taken to the Report Designer.



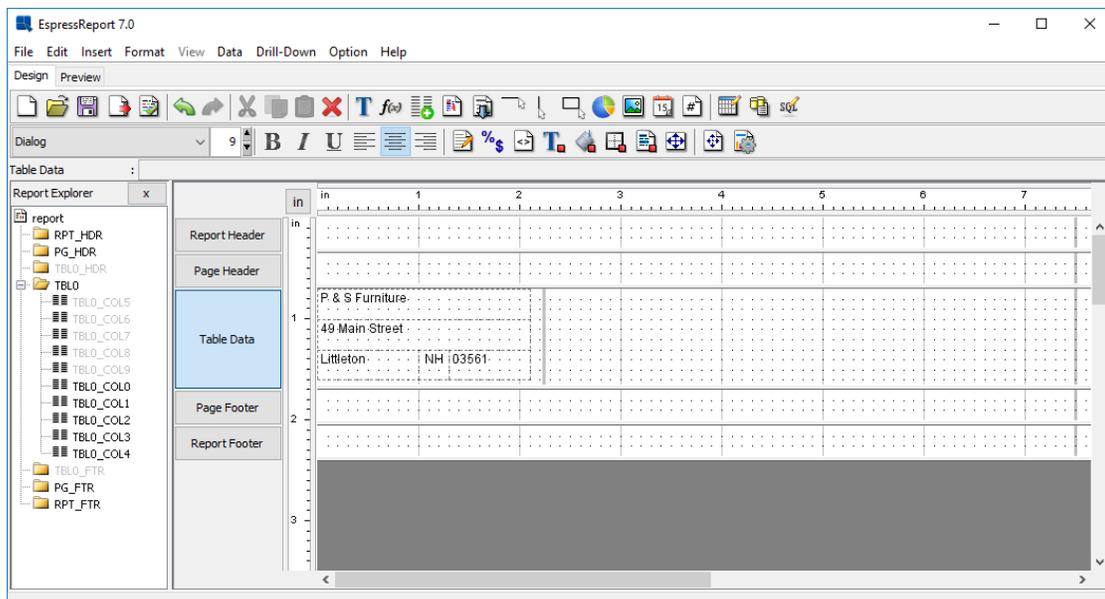
*Mailing Labels Report in Design Window*

We don't need the left columns in Mailing Label Report. We select all the left columns:



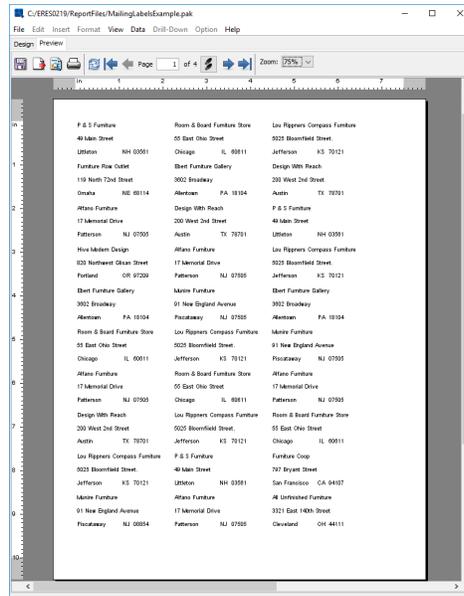
*Mailing Labels Report Edit in Design Window*

And remove them. We continue to edit the report to get general mailing label appearance:



*Mailing Labels Report Appearance in Design Window*

Now click on the *Preview* tab to preview the Mailing Labels layout.



*Mailing Labels Report in Preview Window*

Now go to the Report Designer window, and select *Exit* from the *File* menu to close the interface. We want to save the report, so select *Yes* in the prompt and specify the name for the new-created Mailing Label Report.

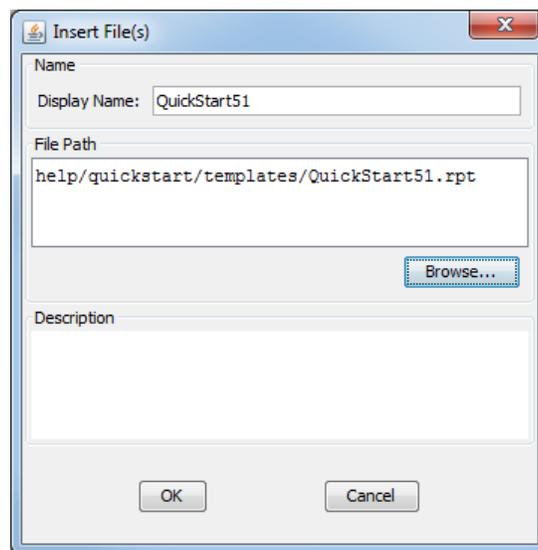
## Q.4.2. Basic Report Formatting

In this section, we will open an unformatted template and use some of the basic formatting features in EspressoReport to create a polished presentation. To begin formatting the report, you will first need to add the template to the Organizer.

### Q.4.2.1. Add a Report Template to Organizer

If you do not have the Organizer open, open it. Then select your project in the left-hand side. Next, click the *Insert*

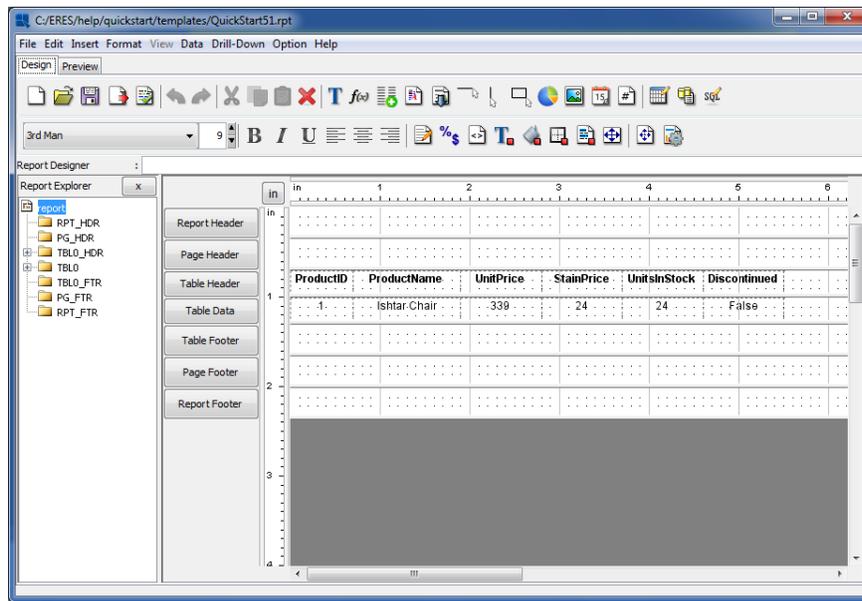
*File* button on the toolbar . A new dialog allowing you to select a file to insert into the Organizer.



*Insert File Dialog*

In the dialog, click the *Browse* button, navigate to `help/quickstart/templates/`, and select `QuickStart51.rpt`. The display name and corresponding URL should be automatically filled in if you have set the

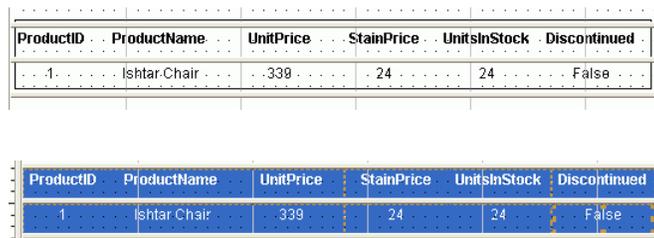
URL mapping correctly(Section 2.1.5 - URL Mapping). Click *OK* to add the report. The dialog will close and you will see an entry in the Organizer for the newly added report.



*QuickStart51.rpt in the Report Designer*

#### Q.4.2.2. Move and Align Report Elements

Report elements can be moved one by one, by click and dragging the cell. Report elements can also be moved as groups. To move a group of elements you must first select the group using a selection box. To activate the selection box, left-click on the report and drag the mouse to draw a box around the report columns. They should become highlighted when you release the mouse button. To add some more elements to your current selection, press the **CTRL** key and click again, or draw another selection box while holding the **CTRL** key.



*Drawing a Selection Box in Report Designer*

Once the fields in the report have been selected, click and drag to indent them about half an inch. Next, click the

*Left Alignment* button on the toolbar . This will align all of the cell text to the left edge of the cell.

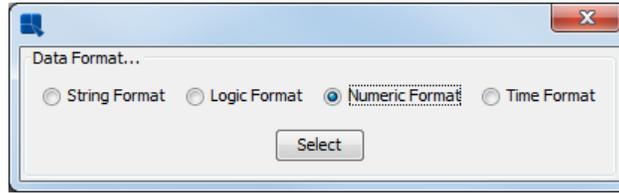
#### Q.4.2.3. Data Formatting

There are a number of options available that allow you to control how data will be displayed (date format, decimal places, rounding, etc). Using the selection box again, select the *UnitPrice*, and *StainPrice* columns (just the column fields not the headers).



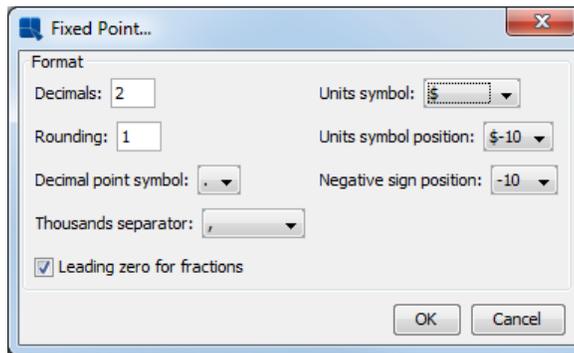
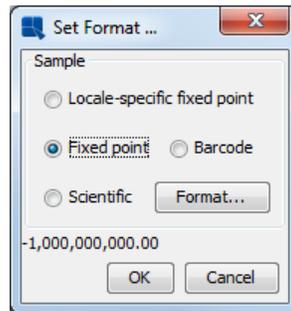
*Two Column Selection*

Once the columns have been selected, select the *Data Format* option from the *Format* menu. This will open a dialog prompting you to select which data type you would like to set the format for. Select *Numeric Format* and click *Select*.



*Data Type for Formatting Dialog*

This will open a new dialog. From this dialog select *Fixed Point* and click the *Format* button. At the next dialog select to have two decimal points and select the dollar sign as the *Units symbol*.



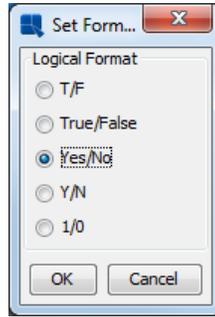
*Numeric Format Dialogs*

Click *OK* and *OK* again at the previous dialog. The selected fields will be converted to currency format.

ProductID	ProductName	UnitPrice	StainPrice	UnitsInStock	Discontinued
1	Ishtar Chair	\$339.00	\$24.00	24	False

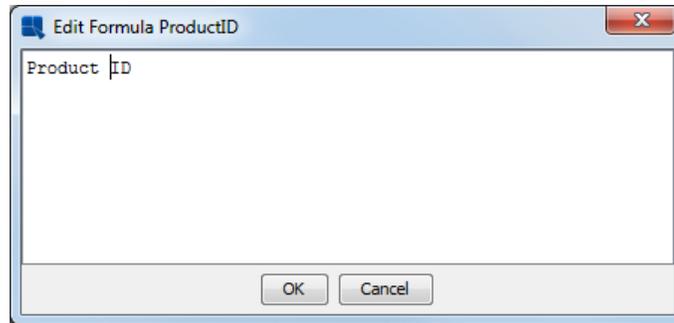
*Numeric Data in Currency Format*

Next, select the *Discontinued* column. The border outline will appear by clicking it. Again, click the *Data Format* button . This time a dialog will appear allowing you to select a format for the Boolean column. Select *Yes/No* and click *OK*. The data in the column will now change.



*Data Format Dialog for Boolean Data*

Now we will edit the label text. By default column headers will display the column names from the database. However, you can override these headers with a custom one. Double click on the `ProductID` cell and a dialog will appear allowing you to modify the column header. Insert a space between `Product` and `ID` and click *OK*. The change will appear in the design window.



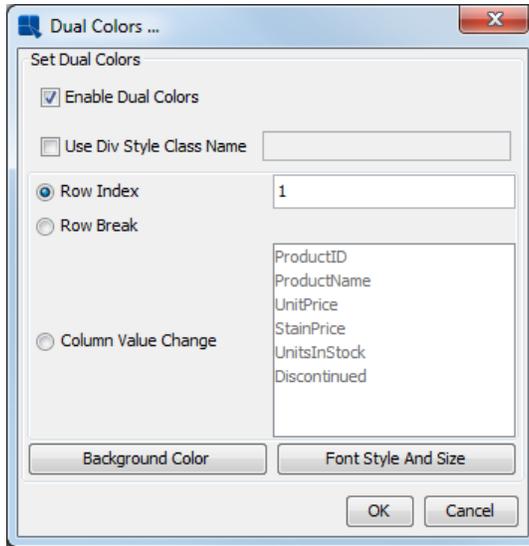
*Edit Column Header Dialog*

Repeat this for each column label except `Discontinued` so that you have inserted appropriate spaces in all of them.

#### Q.4.2.4. Set Dual Colors

The dual colors feature in EspressoReport allows users to differentiate different rows or groups of data by changing the background color, and/or font. Use the group selection tool to turn on dual colors and select all of the columns in the report (not headers).

Next, select the *Dual Colors* button on the toolbar . A dialog will appear prompting you to set dual colors for the columns. Click the checkbox labeled *Enable Dual Colors*. Then select the *Row Index* radio button to indicate that you would like to change color based on a row value. Enter **1** for the row index value.



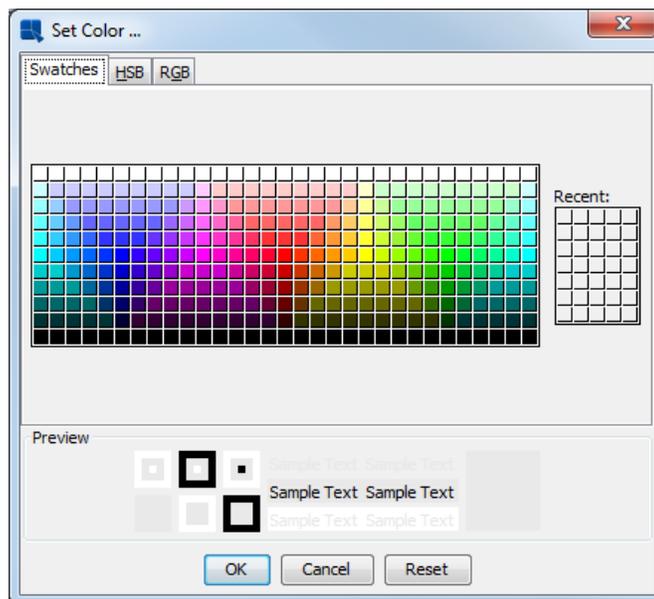
*Dual Colors Dialog*

Next, click the *Background Color* button. A dialog will appear giving you the option to set the background transparent and showing the current background color.



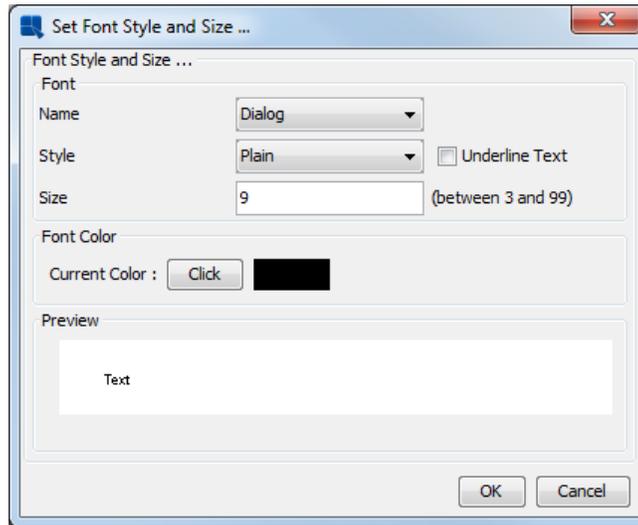
*Background Transparency Dialog*

Click the button labeled *Click* and a new dialog will appear with color swatches allowing you to select a new background color.



*Choose Color Dialog*

Select the new background color that you would like to use and click *OK*. You will return to the first dialog where the color selection will be reflected. Click the *OK* button again and you will return to the Dual Colors dialog. Once you are back at the Dual Colors dialog, click the *Font Style and Size* button. This will bring up a dialog allowing you to specify the font, font size, and font style for the alternating rows.



*Font Style and Size Dialog*

From this dialog, set the *Name* to **Dialog**, the *Style* to **Plain** and the *Size* to **9**. This will match the fonts for the alternating rows. Click *OK* to go back to the dual colors dialog and once more to return to the Report Designer. Now when you preview the report you will see alternating bands of color for each row.

Product ID	Product Name	Unit Price	Stain Price	Units In Stock	Discontinued
1	Ishtar Chair	\$339.00	\$24.00	24	No
2	Shamash Chair	\$449.00	\$26.00	4	No
3	An Chair	\$425.00	\$22.00	14	No
4	Enlil Chair	\$450.00	\$27.00	12	No
5	Enki Chair	\$425.00	\$24.00	45	No
6	Ninhursag Chair	\$369.00	\$25.00	12	Yes
7	Nargal Chair	\$335.00	\$17.00	25	No
8	Zabada Chair	\$312.00	\$22.00	40	No
9	Ninurta Chair	\$345.00	\$19.00	14	No
10	Marduk Chair	\$489.00	\$31.00	12	No
11	Nabu Chair	\$456.00	\$31.00	25	No
12	Cula Chair	\$468.00	\$33.00	4	Yes
13	Adad Chair	\$452.00	\$35.00	16	No
14	Nusku Chair	\$425.00	\$24.00	26	No

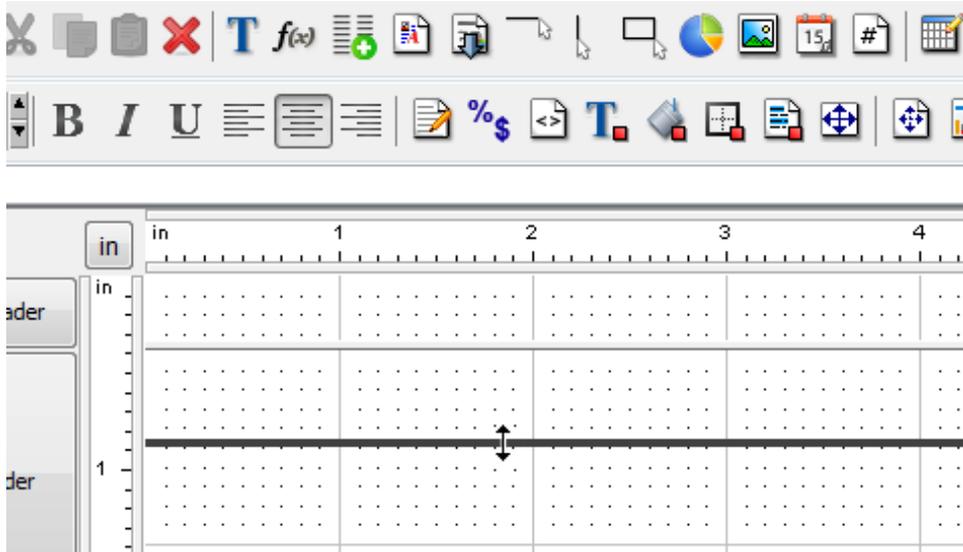
*Dual Colors in Preview*

## Q.4.2.5. Inserting Elements

To further customize reports, many different types of elements can be added to a report template.

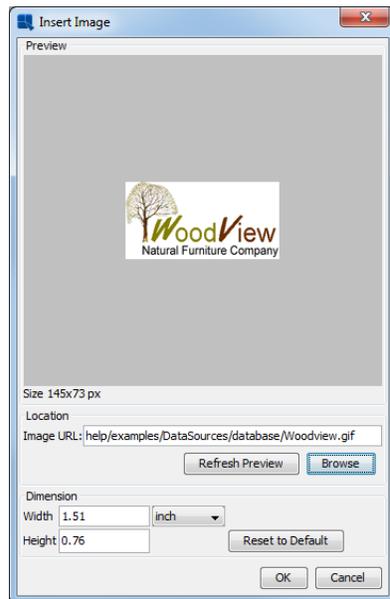
### Q.4.2.5.1. Insert an Image

To insert an image into the report header, first you will want to resize the section to fit the image. In the design window, place the mouse over the lower section divider of the Report Header section, click and drag down making the section about an inch taller.



*Re-Sizing a Report Section*

Once the section is resized, click the *Image* button on the toolbar . A small rectangle will follow your mouse pointer around the design window. Position the rectangle in the upper left-hand corner of the Report Header section and click. A dialog will appear prompting you to select an image to insert. Click on *Browse* and navigate to `help/examples/DataSources/database/Woodview.gif`. You should now see an image in the preview panel.

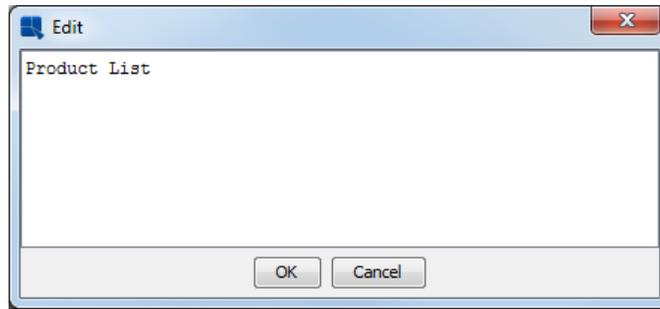


*Insert Image Dialog*

Click *OK* and the image will be inserted into the report. The image will be represented by a gray rectangle in the design view. You can see the image when you preview the report.

#### Q.4.2.5.2. Insert a Title

To insert a title, click the *Insert Label* button on the toolbar . A small rectangle will now follow your mouse pointer around the design window. Position the rectangle next to the image you are inserted in and click. A dialog will appear prompting you to enter the label text. Type in the text of your desired title.



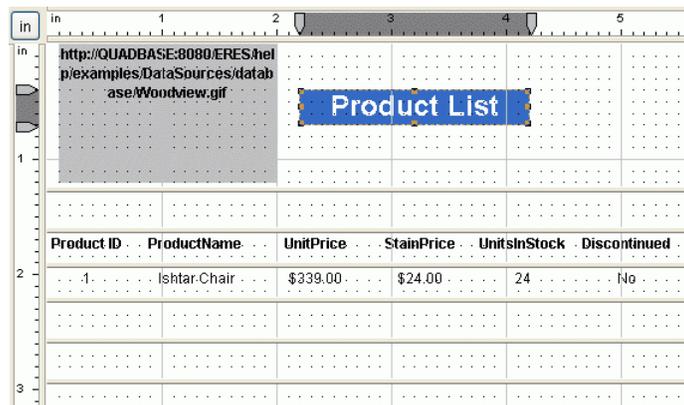
*Insert Label Dialog*

Click *OK* and the title will be added to the report. As you can see, the text is fairly small by default. To change this, change the font size dialog on the toolbar to be 18pt font.



*Toolbar Font Options*

When you do this you will notice that some of the text in the title cell has now disappeared. This is because the text is now larger than the space defined. To resize the cell, right click and drag until you can see the report title again. Then click the *Left Alignment* button on the toolbar to set the text alignment to the left as you did for the other report elements. Move and position the title cell so it is next to the inserted image.



*Report Header with Image and Title*

#### Q.4.2.5.3. Insert a Line

Next, we will add a horizontal line below the column headers. To do this, first resize the Table Header section slightly to provide some more space below the column headers. Next, select the *Insert Horizontal Line* button on

the toolbar . Your cursor will then change into a cross. Click below the `ProductID` header and drag across to the last column to draw the line.

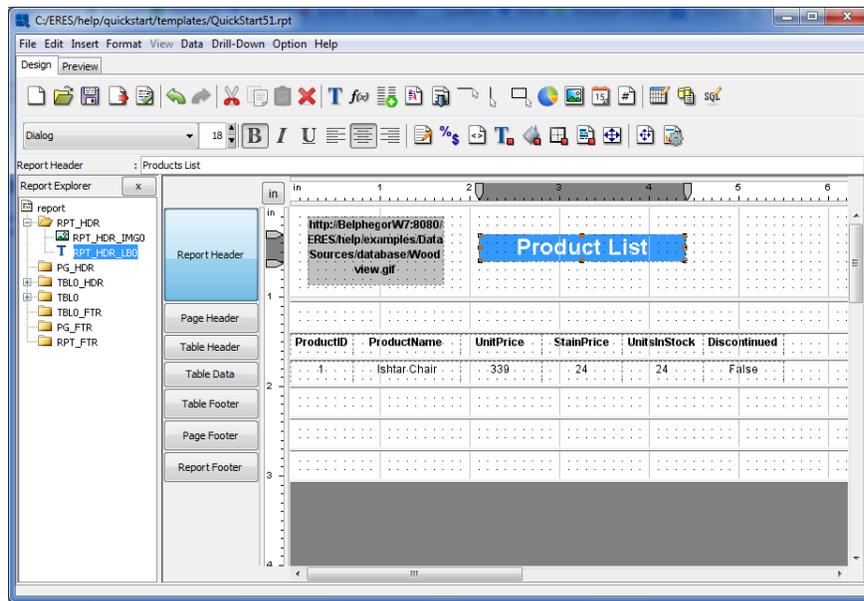
Product ID	ProductName	UnitPrice	StainPrice	UnitsInStock	Discontinued
1	Ishtar Chair	\$339.00	\$24.00	24	No

*Drawing a Line in Report Designer*

#### Q.4.2.6. View Report Elements in Report Explorer

*Report Explorer* from the *Option* menu is turned on by default. It is the panel on the left-hand side of the Report Designer, showing the report elements in a tree format. Click to expand some of the sections and you will see all

of the elements in the report represented in the tree. If you select one of the elements in the tree, the corresponding element in the report will be selected.



*Designer with Explorer Open*

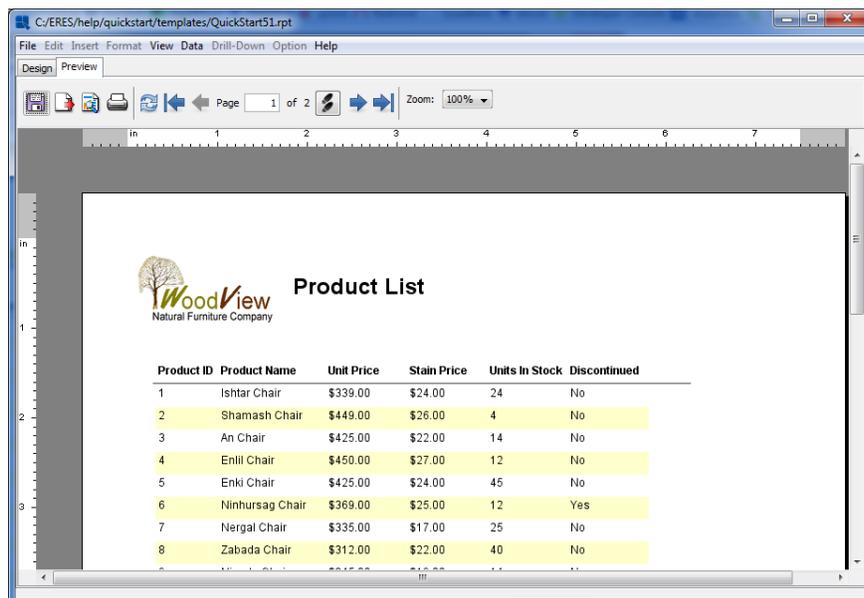
You can close the report explorer by selecting *Report Explorer* again from the *Option* menu.

#### Q.4.2.7. Set Section Options

In EspressReport, each of the report sections has a number of configurable options allowing you to display the data in the sections in any number of different ways. To invoke the options menu for a section, click the button for that section on the left-hand side of the design window. In this example, bring up the options menu for the *Table Header* section by clicking on the corresponding button.

From the pop-up menu select *Repeat On Every Page*. This will cause the Table Header to be drawn on each report page instead of only once (which is the default). For more about report section options, see Section 4.1.3.3 - Section Options.

Now that you have finished formatting the report, preview it to see the results.



*Finished Report*

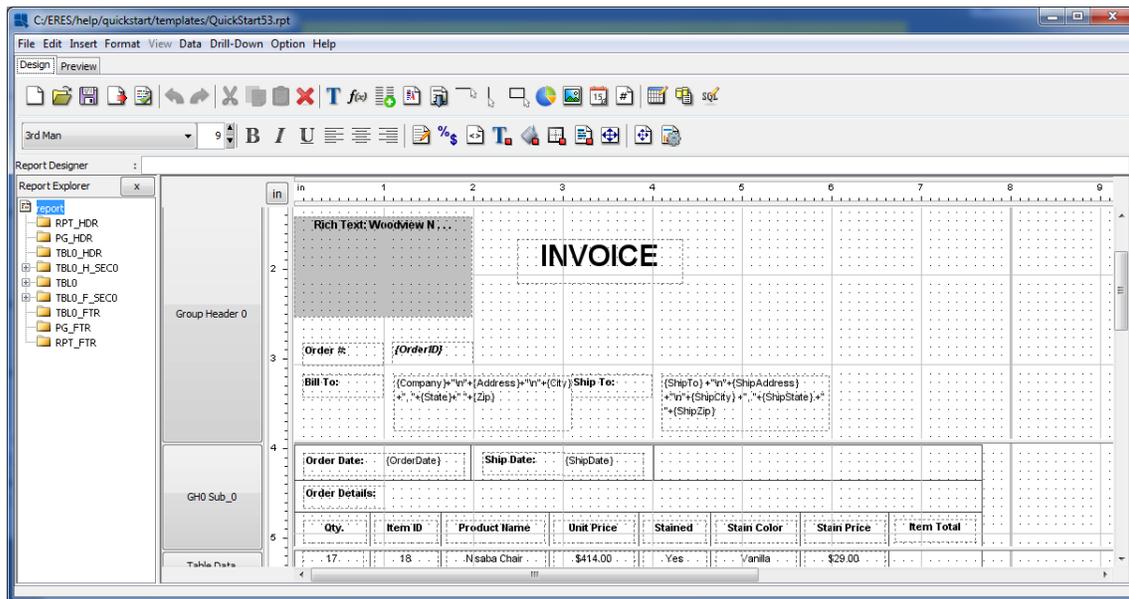
Now save the changes you have made to the report and exit from the Report Designer.

## Q.4.3. Advanced Reporting Features

### Q.4.3.1. Formulas & Scripting

EspressReport ES provides a large built-in formula and scripting library, giving you many ways to manipulate and analyze report data. In the following section we will take a template and use formulas and scripts to calculate/add some values to the report.

Following the same procedure as in Section Q.4.2.1 - Add a Report Template to Organizer, add the `QuickStart53.rpt` file under `help/quickstart/templates` into your project in the Organizer. Then right click on the entry for this file in Organizer and select *Open File* from the pop-up menu. This will open the report in Report Designer.

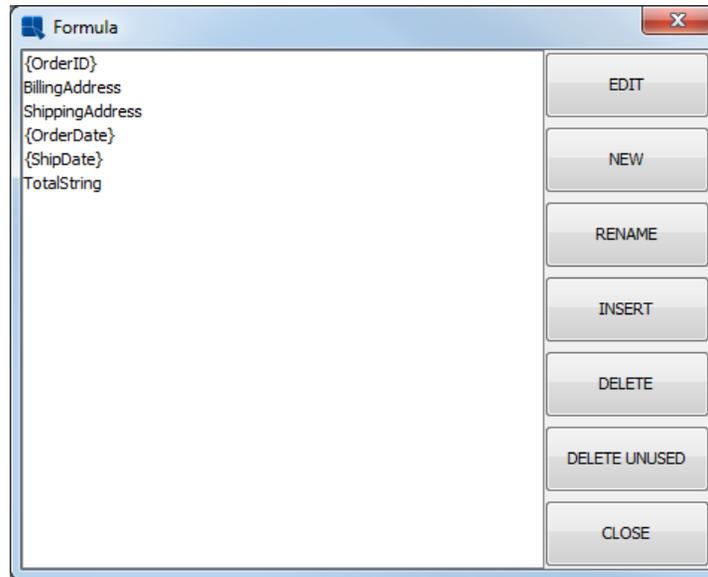


*QuickStart53.rpt in Report Designer*

The report is an invoice created using the Master & Details layout. Notice that the `Item Total`, as well as the `Sub-total` are blank. We will add in the formulas to calculate these values.

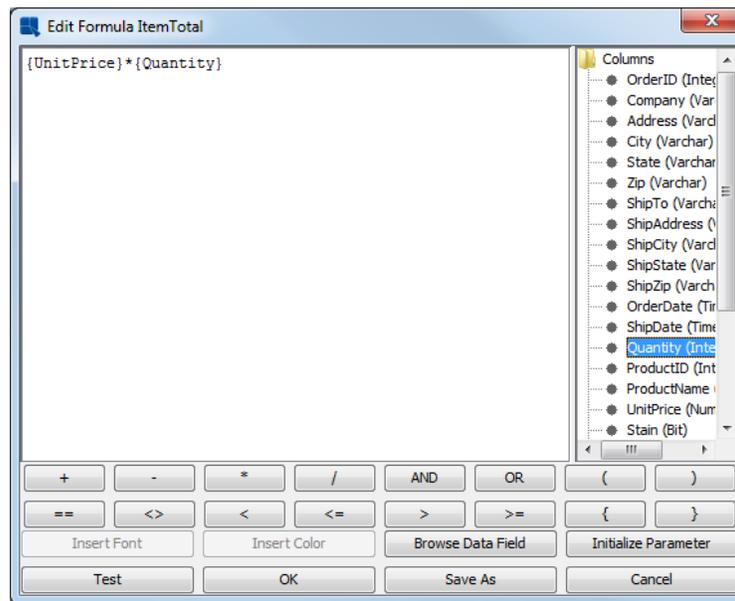
### Q.4.3.2. Add a Formula

To insert a formula, click the *Insert Formula* button on the toolbar . This will bring up a dialog containing all the formulas within the report. Notice that the template has several existing formulas. Click *New* to create a new formula. At the prompt, specify the name `ItemTotal` for the formula.



*Report Formula List*

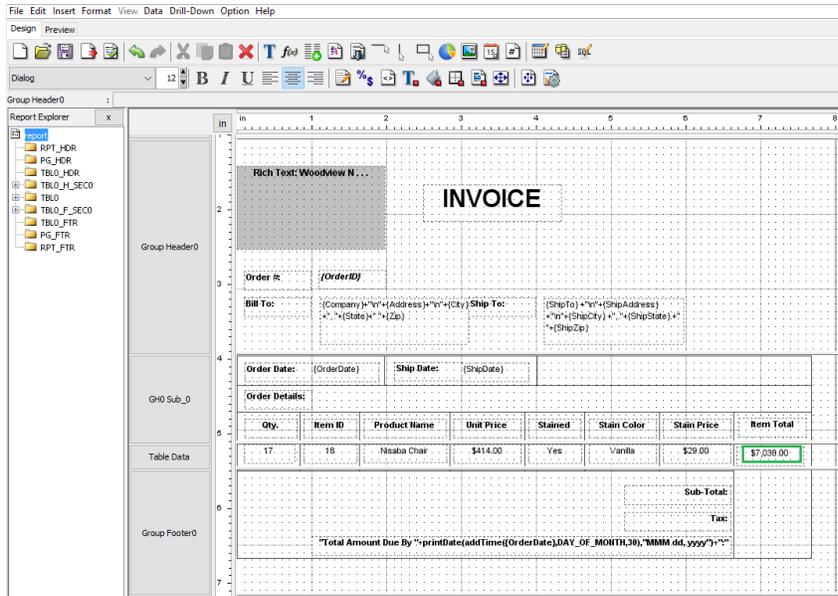
The Formula Builder window will then open. Double click on the *Columns* folder in the right-hand side to expand it. Then double click on the *UnitPrice* column to add it to the formula. Next, click the *multiply* ("\*") button. Next, double click on the *Quantity* column to add it. The finished formula should look like this: `{UnitPrice}*{Quantity}`.



*Formula Builder Window*

Click the *Test* button to ensure that the formula is entered correctly. Then click *OK*. You will be taken back to the formula list where your new formula has been added. From the formula list, select the *ItemTotal* formula that you just created and click the *Insert* button. The dialog will close and a small dotted rectangle will follow your pointer around the design window. Position the formula below the *Item Total* label and between the lines in the Table Data section and click. The formula will then be added to the report.

Next set the data format for the formula to be currency, like in Section Q.4.2.3 - Data Formatting.



Formula is Inserted

Now preview the report, notice that because the formula was added to the Table data section, it now computes for each row of data.

<b>Woodview Natural Furniture</b> 2855 Kifer Rd., Ste 203 Santa Clara, CA 95051 Tel: (480) 982-0835 Fax: (408) 982-0838		<h1>INVOICE</h1>					
<b>Order #:</b>	10059	<b>Ship To:</b>	Eastern Treasures 844 South Lake Drive Pitterson, NJ 09882				
<b>Bill To:</b>	Eastern Treasures 123 Summer Rd. Pitterson, NJ 09882	<b>Ship To:</b>	Eastern Treasures 844 South Lake Drive Pitterson, NJ 09882				
<b>Order Date:</b>	Mar 12, 2000	<b>Ship Date:</b>	Mar 19, 2000				
<b>Order Details:</b>							
<b>Qty.</b>	<b>Item ID</b>	<b>Product Name</b>	<b>Unit Price</b>	<b>Stained</b>	<b>Stain Color</b>	<b>Stain Price</b>	<b>Item Total</b>
17	18	Nisaba Chair	\$414.00	Yes	Vanilla	\$29.00	\$7,038.00
18	26	Geb Table	\$1,215.00	Yes	Vanilla	\$141.00	\$21,870.00
<b>Sub-Total:</b>							
<b>Tax:</b>							
<b>Total Amount Due By Apr 11, 2000:</b>							

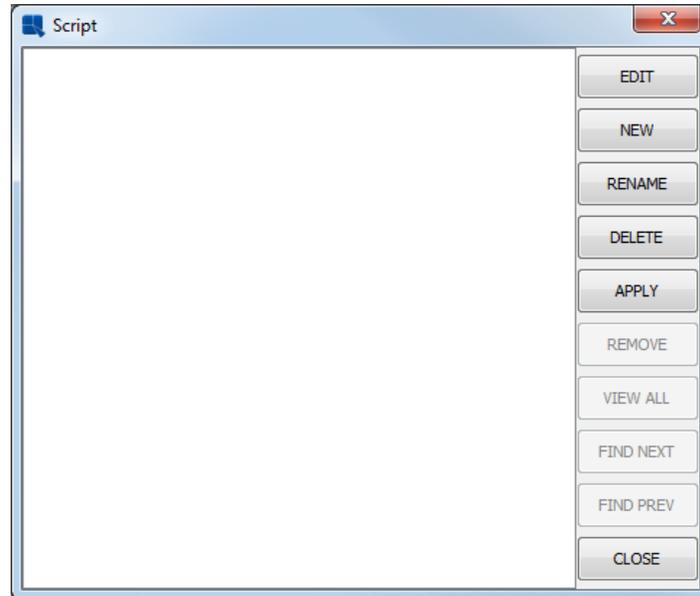
Report with Formula

### Q.4.3.3. Add a Script

Notice that the formula you added in the previous section does not correctly calculate the line total for the invoice. By only multiplying the unit price and the quantity the formula is ignoring whether an item was stained (which incurs an additional cost). To take care of this, we will use cell scripting.

To add a script, select the *ItemTotal* formula that you created in Section Q.4.3.2 - Add a Formula and click the

Scripting button on the toolbar . This will bring up a dialog containing all the scripts in the report. Since no scripts have been previously added the dialog will be blank.

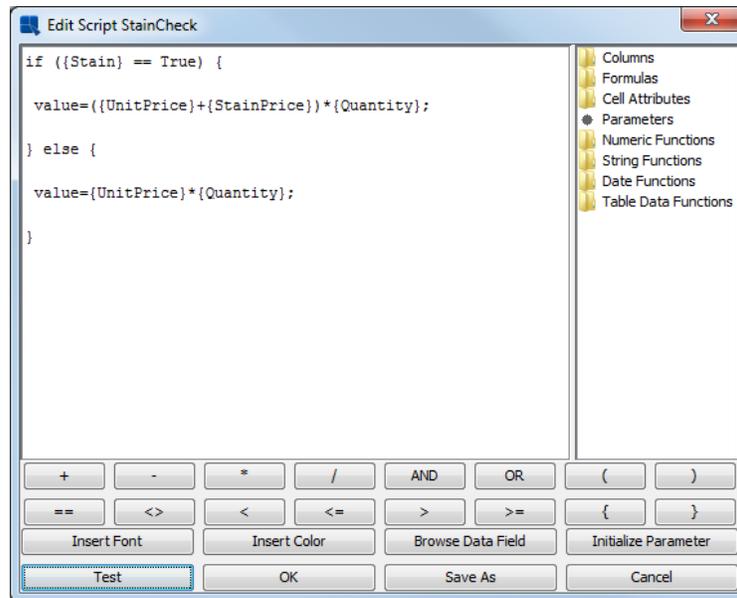


*Script List Window*

Click *New* to create a new script. In the prompt, type the name **StainCheck** for the script. Click *OK* and the Formula Builder will open allowing you to add the script. Enter the following script:

```
if ({Stain} == True) {  
value=({UnitPrice}+{StainPrice})*{Quantity};  
} else {  
value={UnitPrice}*{Quantity}; }  
}
```

The script dynamically modifies the `ItemTotal` price depending on whether the item has been stained or not. If the item has been stained (`{Stain} == True`) then the `ItemTotal` value includes the stain price. If the item has not been stained, the price is calculated in the same manner as before.



*Formula Builder for Cell Scripts*

Click *Test* to ensure that the script has been entered correctly. Then click *Ok*. You will return to the script list where the new script has been added. In the script list, select the script that you just created and click the *Apply* button to apply the script to the column. Notice that a check mark appears in the upper-left hand corner of the cell to which the script has been applied. Then click the *Close* button to close the dialog.

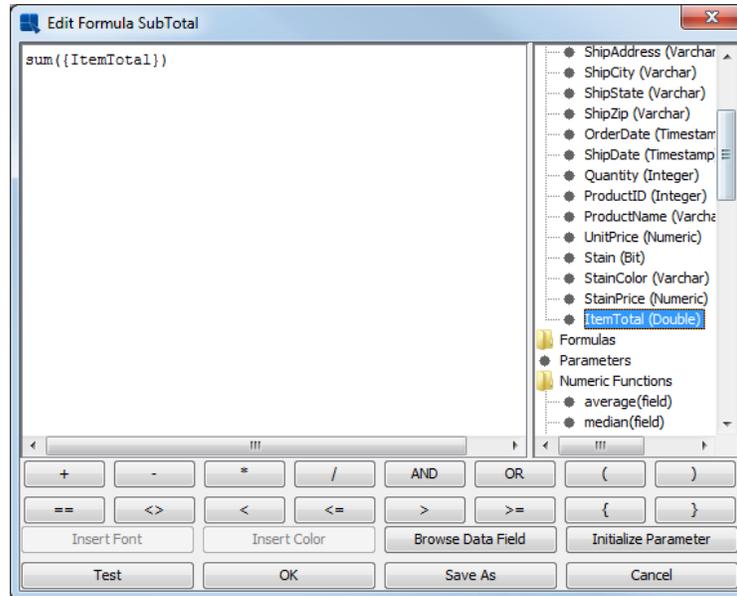
Stain Color	Stain Price	Rem Total
Vanilla	\$29.00	\$7,039.00

*Cell with Applied Script in Design Window*

#### Q.4.3.3.1. Add an Aggregation

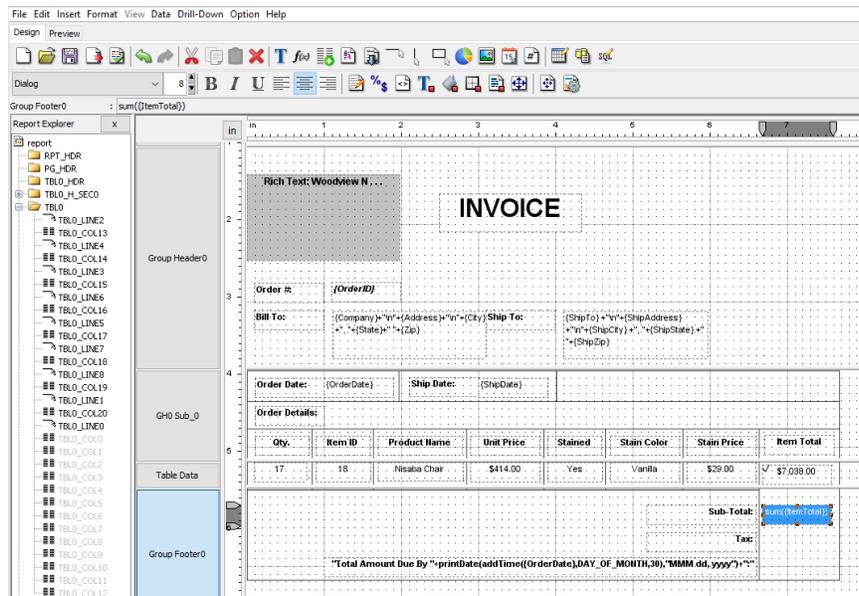
In ERES, formulas can also be used to aggregate report columns. Adding an aggregation is the same as adding a formula like in Section Q.4.3.2 - Add a Formula. Click on the *Insert Formula* icon and then click on the *New* icon to create a new formula. Name this formula **SubTotal**.

In the Formula Builder, double click on the *Numeric Functions* folder to expand it. Double click on the *sum(field)* function to insert it into the formula. Next use the cursor to highlight the *field* portion of the sum function. Double click on the *Columns* folder to expand it. At the end of the list of columns is one called *ItemTotal* that you created in Section Q.4.3.2 - Add a Formula. Double click it to add it to the formula. The finished formula should look like this: `sum( {ItemTotal} )`



*Formula Builder with Aggregation Formula*

Click *Test* to ensure that the formula is entered correctly. Then click *OK* to return to the formula list. In the formula list, select the formula that you have created and click the *Insert* button. The dialog will close and a small dotted rectangle will follow your pointer around the design window. Position the formula in the *Group Footer* section of the report, below the *Item Total* column, and click to add it.



*Formula Added*

Because the formula is in the *Group Footer* section, it will not calculate until the report is run, and it only displays the text of the formula. Format the formula as currency like in Section Q.4.2.3 - Data Formatting. Now preview the report. Notice that the aggregation reflects the values that are modified by the cell script.

The screenshot shows a report window titled 'C:/ERES/help/quickstart/templates/QuickStart53.rpt'. The report content is as follows:

**Woodview Natural Furniture**  
 2855 Kifer Rd., Ste. 203  
 Santa Clara, CA 95051  
 Tel: (408) 982-0835  
 Fax: (408) 982-0838

**INVOICE**

**Order #:** 10059

**Bill To:** Eastern Treasures  
 123 Summer Rd.  
 Pitterson, NJ 09882

**Ship To:** Eastern Treasures  
 844 South Lake Drive  
 Pitterson, NJ 09882

<b>Order Date:</b>	Mar 12, 2000	<b>Ship Date:</b>	Mar 19, 2000				
<b>Order Details:</b>							
Qty.	Item ID	Product Name	Unit Price	Stained	Stain Color	Stain Price	Item Total
17	18	Nisaba Chair	\$414.00	Yes	Vanilla	\$29.00	\$7,531.00
18	26	Geb Table	\$1,215.00	Yes	Vanilla	\$141.00	\$24,408.00
<b>Sub-Total:</b>							\$31,939.00

*Report With Formulas*

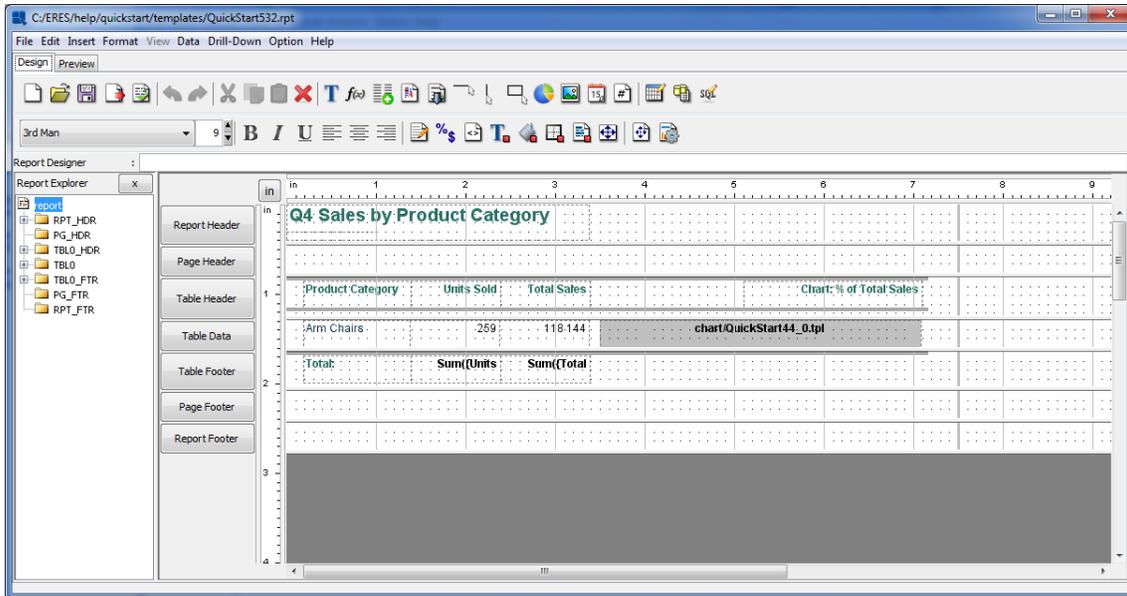
For additional tutorial, add two more formulas to the *Group Footer* section: one to calculate the sales tax and one to calculate a grand total for the order. Once you have finished, click the *Save* button to save the changes that you have made and close the Report Designer.

#### Q.4.3.4. Drill-Down

A unique feature of ERES is the ability to perform drill-down on reports automatically. Using drill-down, users can easily present summarized data in a top-layer report and can click through to view more detailed information. Using this feature, only one template has to be designed for each level of drill-down. For more information on this feature, see Section 4.1.8 - Drill-Down.

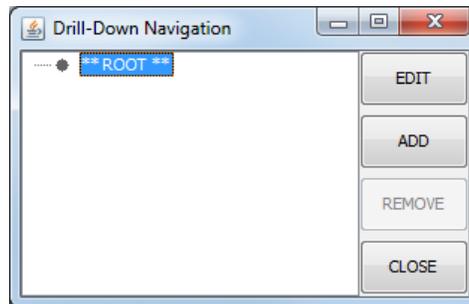
Because the drill-down features uses parameterized queries, you will have to have the Woodview database set up (detailed in Section Q.3.1.1.1 - Setup a JDBC Connection) in order to do this tutorial.

Following the same procedure as in Section Q.4.2.1 - Add a Report Template to Organizer, add the Quick-Start532.rpt file under help/quickstart/templates into your project in the Organizer. Right click on the entry for this file in Organizer and select *Open File* from the pop-up menu. This will open the report in Report Designer.



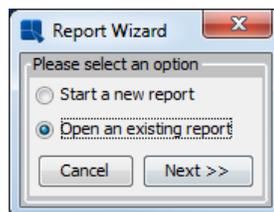
*QuickStart532.rpt in the Design Window*

The report shows aggregated sales data grouped by product category. Next, we will link this template to another parameterized template, so that users can drill into each category and look at the sales figures for products in each category. To add a layer of drill-down, select *Navigate* from the *Drill-Down* menu in the navbar. This will bring up a dialog showing the hierarchy of the any drill-down layers for the report. Only the root will show because there are not any layers defined.



*Drill-Down Navigation Dialog*

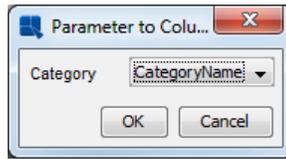
To add a new layer of drill-down, click the *ADD* button. This will bring up a dialog asking you if you would like to use an existing template or create a new one. If you select to create a new template, you will go back to the data registry where you can start designing a report. You can also use an existing report. In either case, the report you use must have a parameterized query or class as the data source. For more on this see Section 3.1.3.2.2 - Parameterized Queries and Section 3.1.6.1 - Parameterized Class Files. In this instance, select to use an existing report and click *Next*.



*Report Options Dialog*

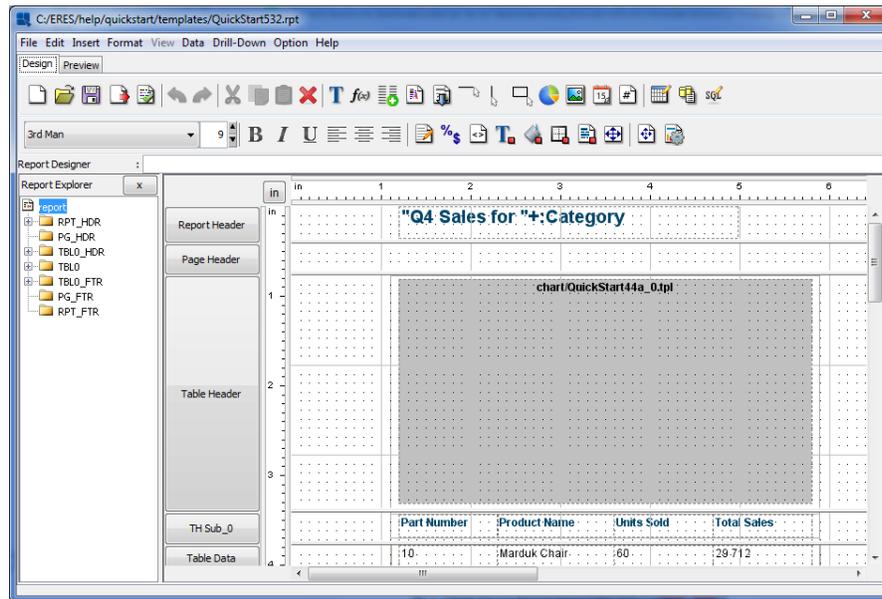
## QuickStart

At the prompt browse to the <ERESInstallDir>/help/quickstart/templates, and select the QuickStart532a.rpt file (or QuickStart532a\_Acc.rpt from the /Access/ directory). A dialog will then open prompting you to select a column from the primary report to map to the drill-down layer.



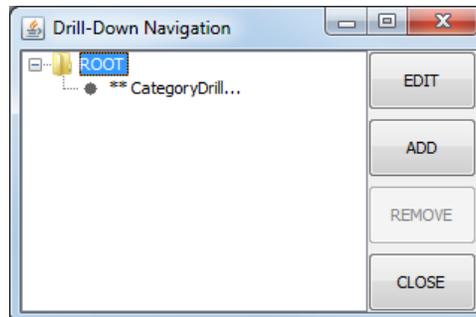
*Column - Parameter Mapping Dialog*

Select to map the CategoryName column to the parameter, and click *OK*. A dialog will open prompting you to specify a display name for the report. Enter a name and click *OK*. The drill-down layer will now open in the design window.



*First Drill-Down Level in Report Designer*

If you preview this level, you can see how the report is parameterized based on the Category Name. Back in the design window, again select *Navigate* from the *Drill-Down* menu. You will now see a new node for your added level under the ROOT node. The stars "\*\*" indicate which level is currently open in the designer.

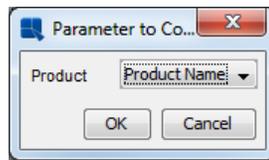


*Drill-Down Navigation Dialog with Additional Layer*

Now we will add one more layer of drill-down to this report, that allows users to drill through the product sales, to see the records of each order for a given product. To do this, select the node for the level you created, and click *Add*.

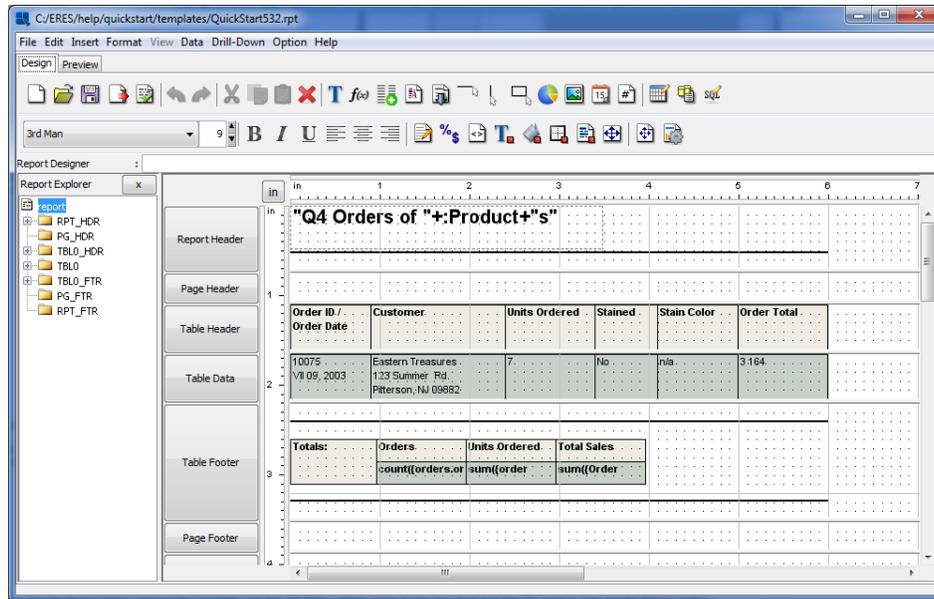
Again, select to use an existing template for the new drill-down layer. At the dialog, browse to the <ERESInstallDir>/help/quickstart/templates directory and select the Quick-

Start532b.rpt file (or QuickStart532b\_Acc.rpt from the /Access/ directory). This will bring up the parameter to column mapping dialog.



*Second Parameter Mapping Dialog*

Select to map the `Product Name` column to the parameter and click *OK*. Enter a display name for the new layer and click *OK* again to open it in the design window.

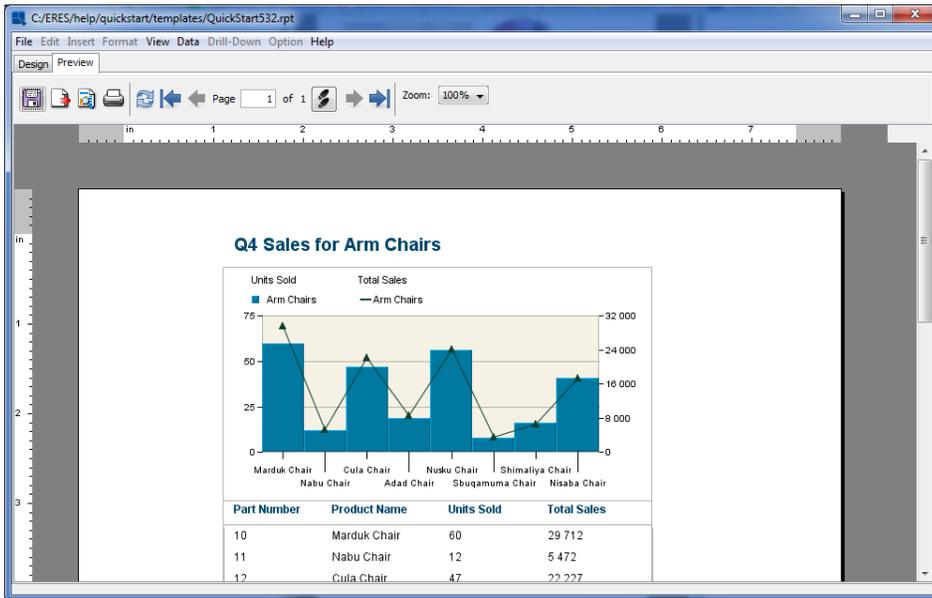
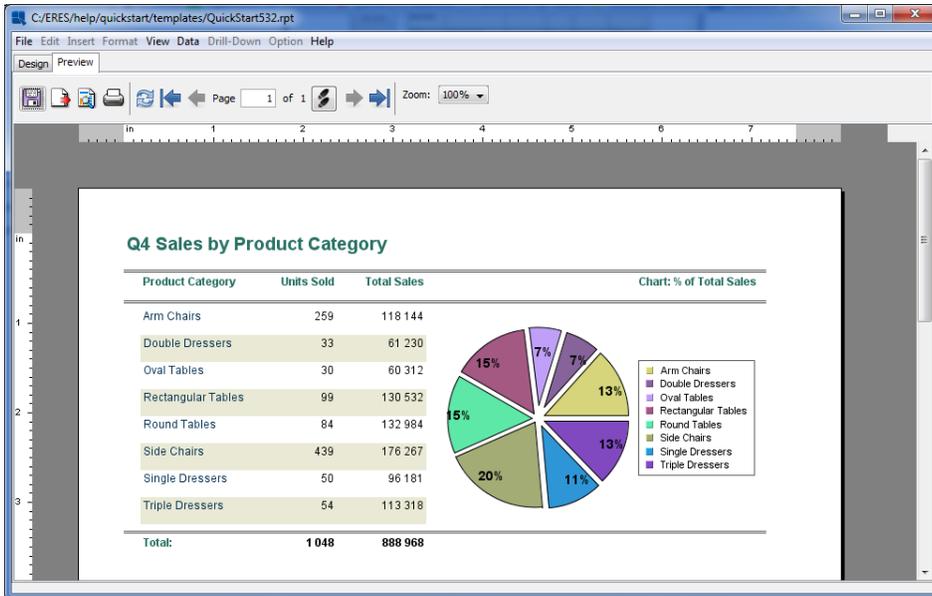


*Second Drill-Down Level in Report Designer*

Now, select *Navigate* from the *Drill-Down* menu again. Notice that there is a new node for the layer you just added. Select the root report, and click the *Edit* button to open it in the design window. You will see the first report open in the designer. Then click *Close* to dismiss the navigation dialog.

Now preview the report. Notice that the cursor changes when you mouse over a field in the `Category Name` column. Click on one and you will be taken to the next level that shows sales for each product in that category.

Within the product report you can click on a field in the `Product Name` column to go to the third level report showing the orders for that particular product.



Order ID / Order Date	Customer	Units Ordered	Stained	Stain Color	Order Total
10069 V 27, 2003	Allied Furniture Emporium 384 Broad St. Littletown, NY 18322	16	No	n/a	7 824
10071 V 09, 2003	All Unfinished Furniture 12 Woodpark Ave. Cleveland, OH 32343	12	Yes	Natural	6 240
10094 XII 02, 2003	Campbell Interiors 839 Via Gaty Rd. Long Island, NY 12334	18	No	n/a	8 802
10086 X 12, 2003	Gale Distributors 345 Sewgen Drive Boontown, NY 11234	14	No	n/a	6 846
<b>Totals:</b>		<b>Orders</b>	<b>Units Ordered</b>	<b>Total Sales</b>	

*Drill-Down Reports in Preview Window*

To navigate back to higher layers, right click and select *Back* from the pop-up menu. Now click the *Save* button to save the changes that you have made to the report.

### Q.4.3.5. Sub-Reports

Another powerful feature in ERES is the ability to use sub-reports to create more complex report layouts and combine data from multiple sources in a report. For detailed information about sub-reports, see Section 4.1.9 - Sub-Reports.

Following the same procedure as in Section Q.4.2.1 - Add a Report Template to Organizer, add the `QuickStart533.rpt` file under `help/quickstart/templates` into your project in the Organizer. Then right click on the entry for this file in Organizer and select *Open File* from the pop-up menu. This will open the report in Report Designer.

Quantity	Product Name	Unit Price	Stained	Stain Color	Total
12	An Chair	\$425.00	No	n/a	\$5,100.00
<b>Order Total: sum(Total)</b>					

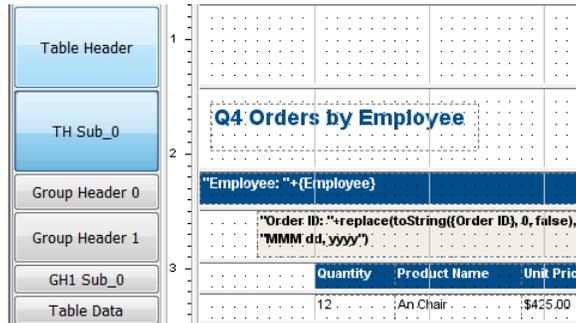
*QuickStart533.rpt in Report Designer*

The report uses two levels of nested grouping to show sales for each employee. We will now add a sub-report to the header that shows aggregated sales by category and employee (in a crosstab layout). Before adding the sub-report,

we will create a new report section in which to place the sub-report. Although a sub-report can be placed anywhere in a report, it often makes sense to give a sub-report its own section, especially if the size of the sub-report can vary.

To insert a nested section, click the *Table Header* button on the left-hand side to bring up the section options menu. Select *Insert Section*. This will spawn a nested section for the table header. For more about nested sections, see Section 4.1.3.1.1 - Nested Sections.

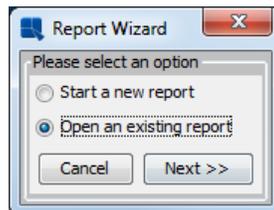
Next we will move the title into the new nested section. To do this, select the cell containing the report title and hit **CTRL+X** to cut it. Place the cursor in the new section and hit **CTRL+V** to paste. The cursor will turn to a cross. Position it where you would like the field and click to add it. You may want to resize the new section to fit the cell.



*New Report Section*

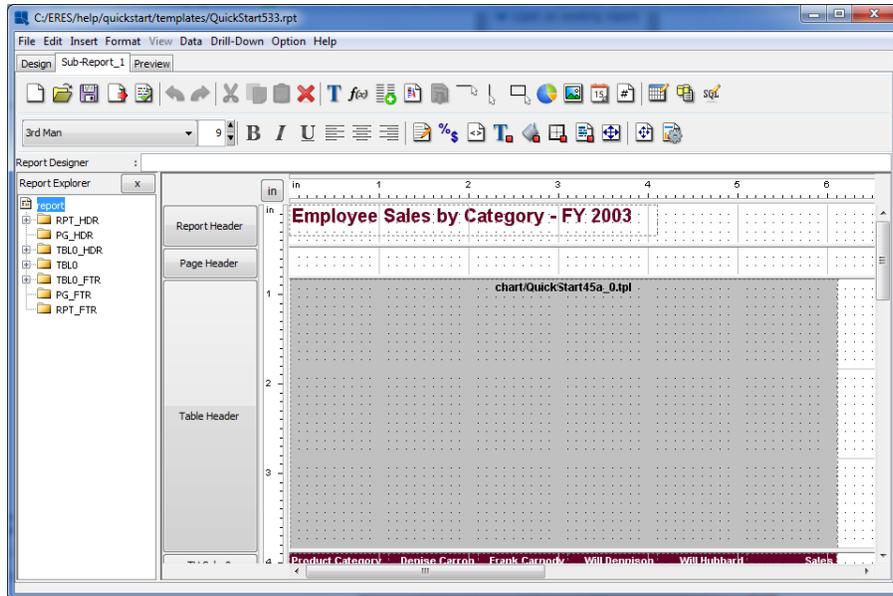
Next, click the *Insert Sub-Report* button on the toolbar . Your mouse pointer will change to the shape of the plus sign. Press the left mouse button in the top-left corner of the Table Header section to insert the SubReport.

As with drill-down, you have the option of creating a new report for the sub-report or using an existing template. Unlike drill-down, however, the template does not have to have a parameterized query as the data source (unless you want to create linked sub-reports. For more on this, please see Section 4.1.9.4 - Linked Sub-Reports). Select to use an existing template and click *Next*.



*Report Options Dialog*

At the prompt, browse to the /QuickStart/, select the QuickStart533a.rpt file, and click *OK*. The sub-report will then open in a new tab called Sub-Report\_1 in the Report Designer.

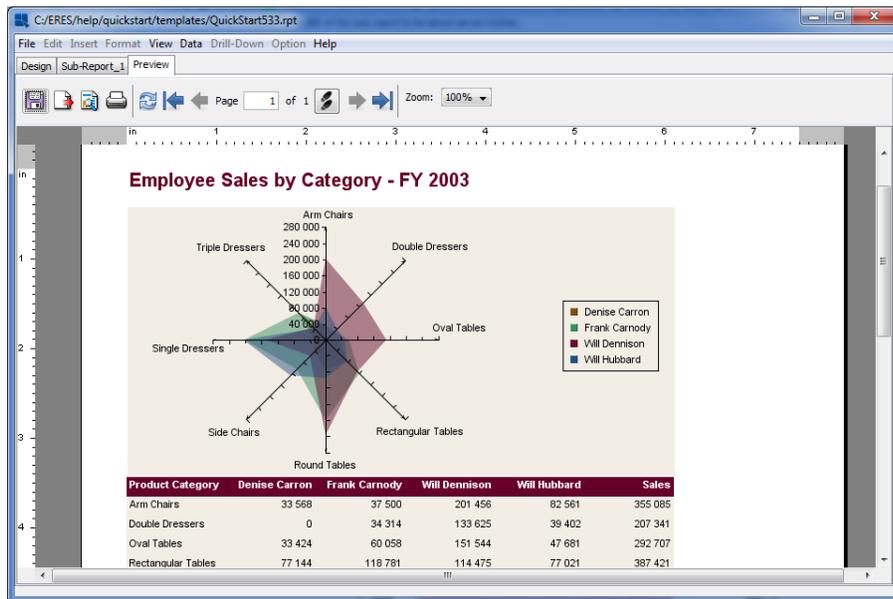


*Sub-Report in Designer*

You can preview just the sub-report by toggling between the *Preview* and *Sub-Report* tab. Now go back to the main report by clicking the *Design* tab. The sub-report will appear as a small gray rectangle. Move the rectangle to the upper left-hand corner of the section and click and drag on the horizontal ruler to increase the width of the sub report to be about seven inches.

Next, we will set the sub-report to resize dynamically. Right click on the sub-report and select *Resize To Fit Content* from the pop-up menu. This will turn the dynamic sizing on or off.

Now preview the report. You can see that the entire sub-report runs before the main report.



*Sub-Report with Main Report in Preview*

The screenshot shows a report designer window with a preview of a report. The report contains a table of furniture items, a sub-report titled "Q4 Orders by Employee" for Frank Carnody, and another table for order 10063.

	1	2	3	4	5	6
Arm Chairs	33 568	37 500	201 456	82 561	355 085	
Double Dressers	0	34 314	133 625	39 402	207 341	
Oval Tables	33 424	60 058	151 544	47 681	292 707	
Rectangular Tables	77 144	118 781	114 475	77 021	387 421	
Round Tables	75 885	200 239	238 331	96 494	610 949	
Side Chairs	41 570	102 652	59 646	126 408	330 276	
Single Dressers	31 410	203 154	149 437	205 615	589 616	
Triple Dressers	92 294	97 914	45 173	40 267	275 648	
<b>Total:</b>	<b>385 295</b>	<b>854 612</b>	<b>1 093 687</b>	<b>715 449</b>	<b>3 049 043</b>	

**Q4 Orders by Employee**  
Employee: Carnody, Frank

Order ID: 10063  
Order Date: IV 05, 2003

Quantity	Product Name	Unit Price	Stained	Stain Color	Total
12	An Chair	\$425.00	No	0	\$5,100.00
8	Enki Chair	\$425.00	No	0	\$3,400.00
2	Anubis Table	\$1,687.00	No	0	\$3,374.00
					<b>Order Total: \$11,874.00</b>

Order ID: 10065  
Order Date: IV 22, 2003

Quantity	Product Name	Unit Price	Stained	Stain Color	Total
----------	--------------	------------	---------	-------------	-------

*Sub-Report with Main Report in Preview*

Once you have finished modifying the report, click the *Save* button on the toolbar to save the changes you have made, and close the Report Designer.

## Q.5. Chart Designer

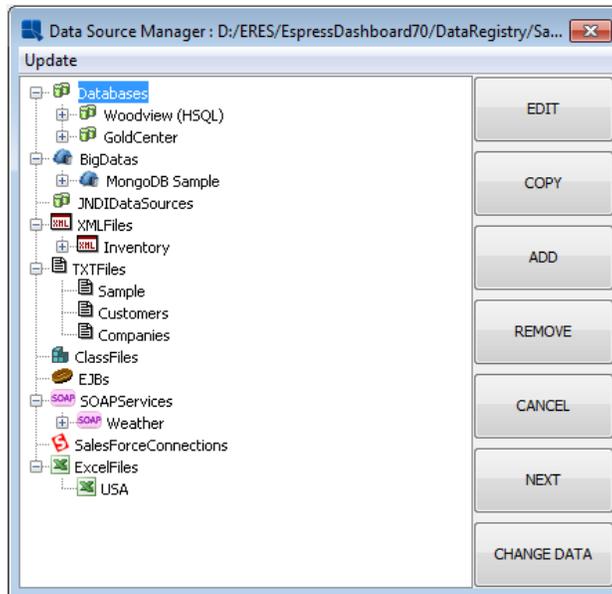
Chart Designer allows users to create and customize charts. This section shows data mapping, and how to use some of the most commonly used features in the Chart Designer. For more information about charts in Chart Designer, please refer to Section 4.2 - Chart Designer.

### Q.5.1. Chart Mapping

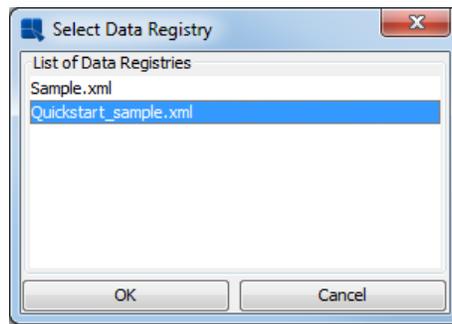
This section will look at several ways that data can be mapped to charts. This section will use the text file data source that was setup in Section Q.3.1.4 - Setup a Text Data Source.

#### Q.5.1.1. Column Chart

Column charts are a good starting point as the mapping for column charts is very similar to that of bar, area, and line charts. To begin creating a chart, click the *Chart Designer* button in the Organizer toolbar . This will launch the Chart Designer interface and open the *Data Source Manager*. Click *CHANGE DATA* option in the lower right corner, select *Quickstart\_sample.xml* data registry and click *OK*.



*Change Data Registry*



*Select Data Registry*

Now, select the *Sample* text data source and click the *NEXT* button. A dialog will open a table with the contents of the text file (first 20 records).

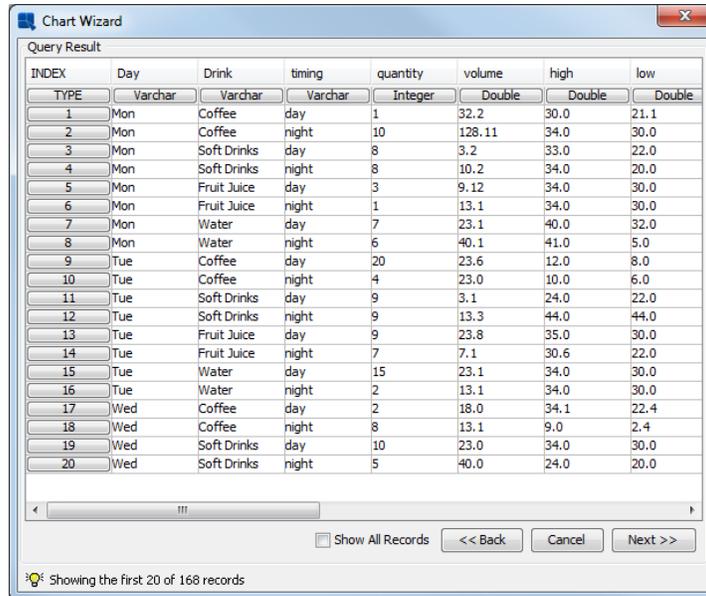


Chart Wizard

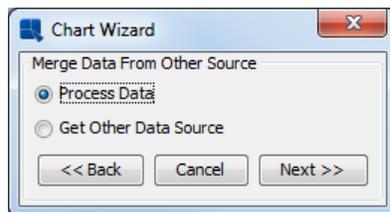
Query Result

INDEX	Day	Drink	timing	quantity	volume	high	low
TYPE	Varchar	Varchar	Varchar	Integer	Double	Double	Double
1	Mon	Coffee	day	1	32.2	30.0	21.1
2	Mon	Coffee	night	10	128.11	34.0	30.0
3	Mon	Soft Drinks	day	8	3.2	33.0	22.0
4	Mon	Soft Drinks	night	8	10.2	34.0	20.0
5	Mon	Fruit Juice	day	3	9.12	34.0	30.0
6	Mon	Fruit Juice	night	1	13.1	34.0	30.0
7	Mon	Water	day	7	23.1	40.0	32.0
8	Mon	Water	night	6	40.1	41.0	5.0
9	Tue	Coffee	day	20	23.6	12.0	8.0
10	Tue	Coffee	night	4	23.0	10.0	6.0
11	Tue	Soft Drinks	day	9	3.1	24.0	22.0
12	Tue	Soft Drinks	night	9	13.3	44.0	44.0
13	Tue	Fruit Juice	day	9	23.8	35.0	30.0
14	Tue	Fruit Juice	night	7	7.1	30.6	22.0
15	Tue	Water	day	15	23.1	34.0	30.0
16	Tue	Water	night	2	13.1	34.0	30.0
17	Wed	Coffee	day	2	18.0	34.1	22.4
18	Wed	Coffee	night	8	13.1	9.0	2.4
19	Wed	Soft Drinks	day	10	23.0	34.0	30.0
20	Wed	Soft Drinks	night	5	40.0	24.0	20.0

Showing the first 20 of 168 records

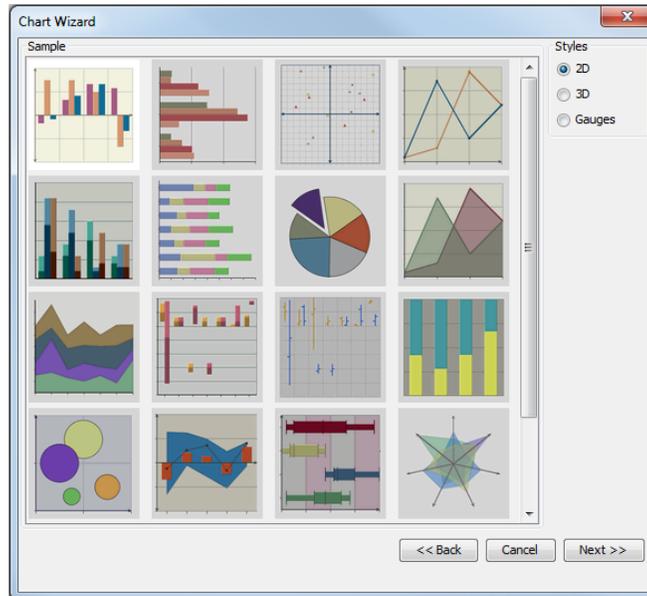
*Contents of Text File*

At the bottom of the dialog, click the *Next* button. A dialog will open asking you if you would like to select an additional data source for the chart.



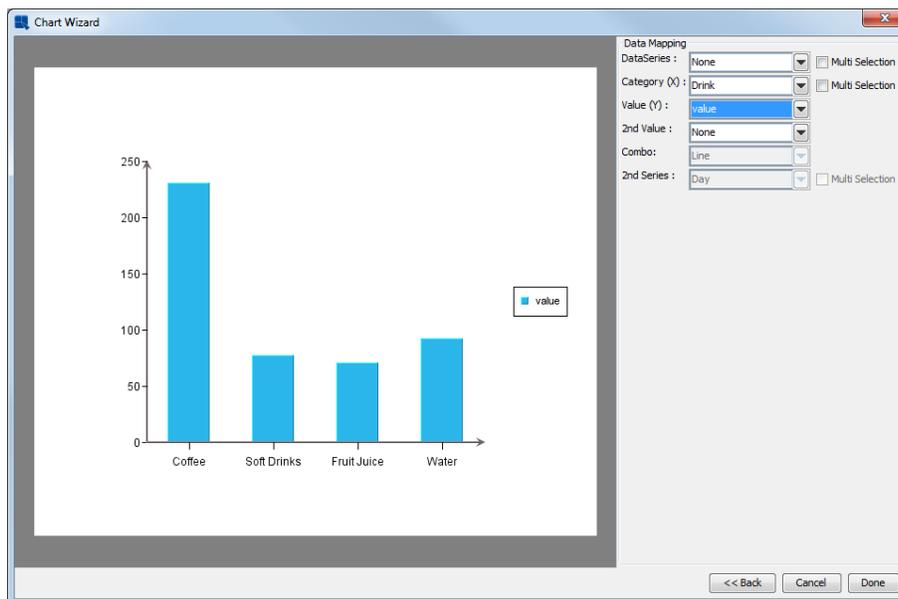
*Add Data Source Dialog*

Select the *Process Data* option and click *Next*. You will then be taken to a dialog prompting you to select which type of chart you would like to create.



*Chart Types Dialog*

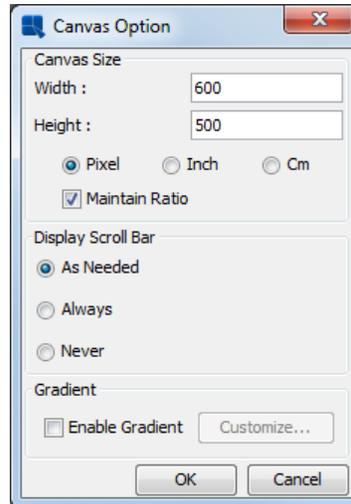
In this dialog, leave selected a *Column Chart* (the first image) and a 2D option (two-dimensional chart) as the data type and click *Next*. You will then be taken to the data mapping dialog which allows you to map columns from the data source to chart elements.



*Chart Data Mapping Dialog*

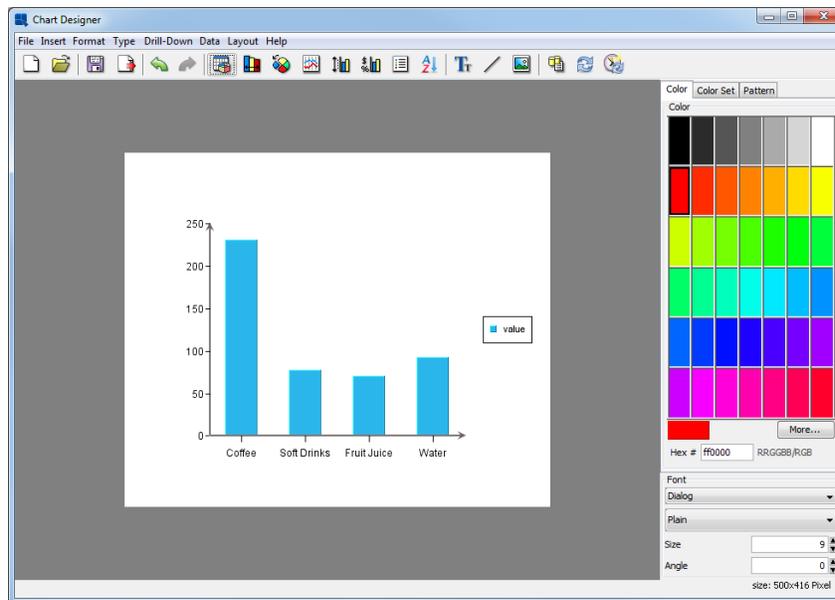
Set the *Data Series* to **None**, the *Category* to **Drink** and the *Value* to **value**. Then click *Done* to finish the Wizard and go to the Chart Designer interface.

If the generated chart does not fit in the viewport of the Chart Designer, select *Canvas* from the *Format* menu. This will bring up a dialog allowing you to set the size of the chart.



*Chart Canvas Dialog*

Enter a smaller size for the chart canvas and click *OK* to close the dialog. You should now see the whole chart in the viewport.



*Chart Designer Showing Column Chart*

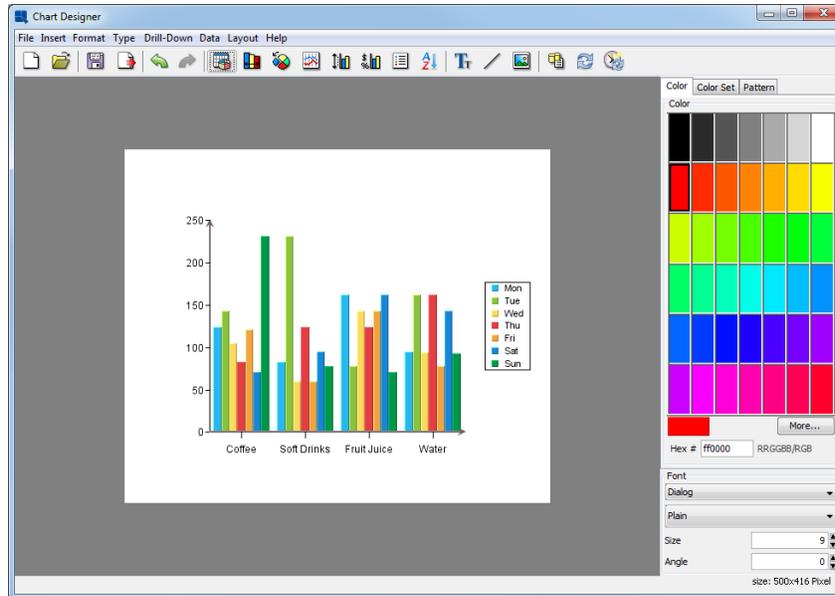
Notice that the column chart contains a column for each distinct value in the Drink column and shows the corresponding value.

#### Q.5.1.1.1. Add a Data Series

At this point, the column chart only contains data points for the categories Drink column. However, ERES supports adding another dimension to this data by way of a series. To add a series to the column chart, click the *Data Mapping*

button in the toolbar . This will return you to the data mapping window.

In the mapping window, change the *Data Series* option from **None** to **Day** and click *Done* to return to the Chart Designer.



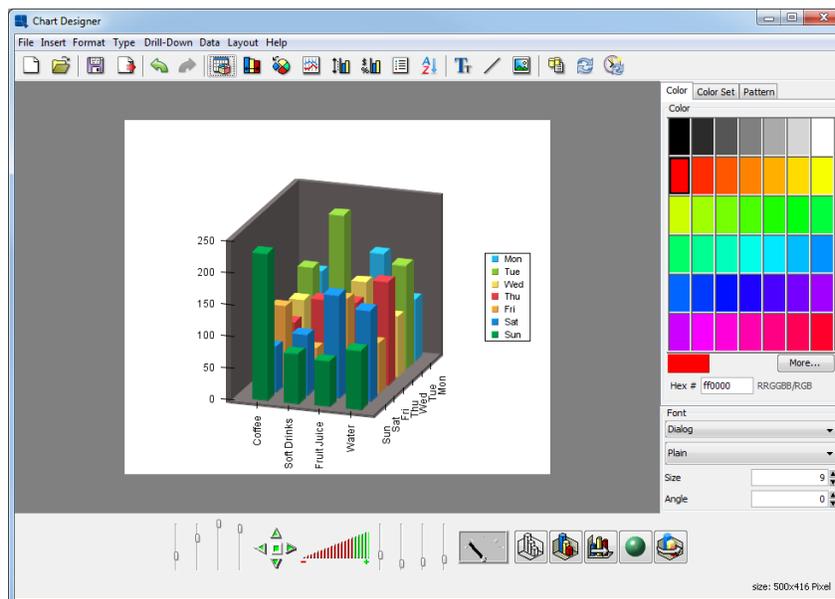
*Chart Designer Showing Column Chart with Series*

Notice now that instead of a single data point for each drink, the column for each drink is now comprised of seven small columns, one showing the data point for each day.

#### Q.5.1.1.2. 3D Column Chart

In a two-dimensional chart, the data series is represented in-line along the X axis. However, a three-dimensional representation provides another axis to work with.

In the Chart Designer select *3D Chart* from the *Type* menu to convert the column chart to a 3D representation.



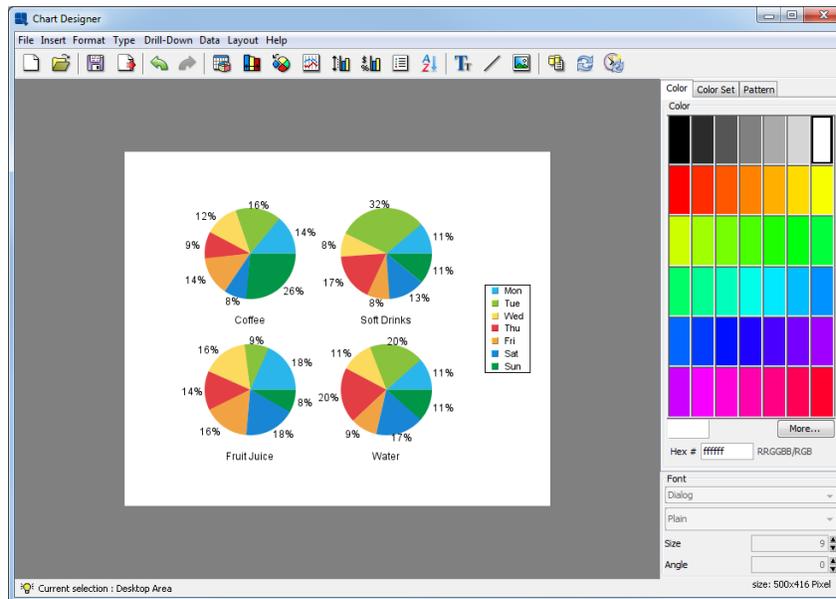
*Chart Designer Showing 3D Column Chart*

The 3D toolbar will automatically appear at the bottom of the Chart Designer when the chart is converted to 3D. The chart may appear squished when you convert it. You can use the sliders next to the zoom function to change the X, Y, and Z scaling for the 3D chart. You can also use the navigation buttons to position the chart in space. For more information about 3D features, see Section 4.2.4.4 - The Navigation Panel.

Notice that with the chart in 3D, the data series is now moved to the Z axis of the chart by default.

## Q.5.1.2. Pie Chart

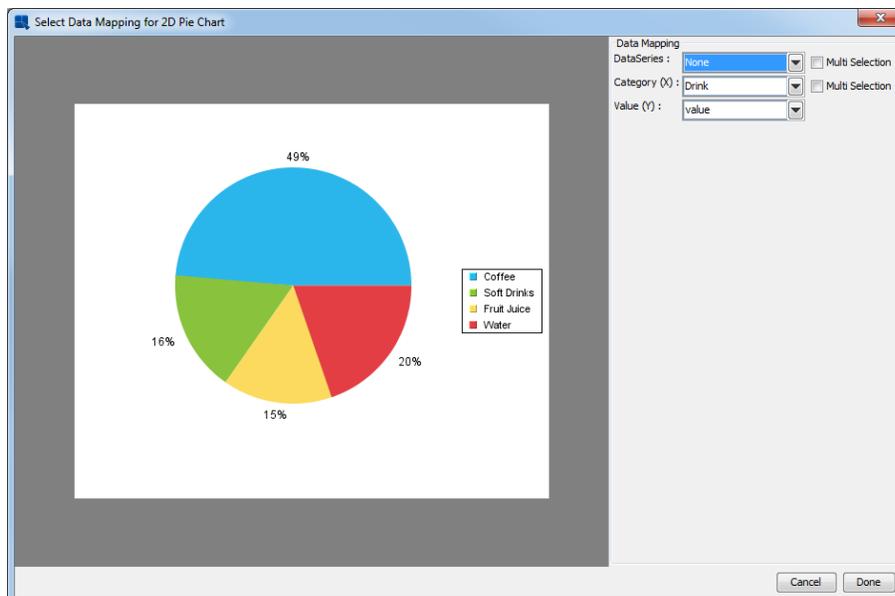
Pie charts are another commonly used chart type that shows values as a percentage of a whole. To convert your chart to a pie chart, first go to the *Type* menu and select *2D Chart* to convert your column chart back to two-dimensional form. Then select *Pie* from the *Type* menu. The chart will then be converted to a pie representation.



*Chart Designer Showing Pie Chart with Series*

Multiple pies are drawn when you have a data series; one for each category. Each category is broken down showing percentage contribution for the series elements.

In order to turn the chart into a single pie, we will remove the series. Click the *Change Data Mapping* button  on the toolbar to bring up the mapping options.



*Data Mapping Options for Pie Charts*

Change the *Data Series* option to **None** and click *Done*. Now in the Chart Designer you will see a single pie made up of your drink categories.

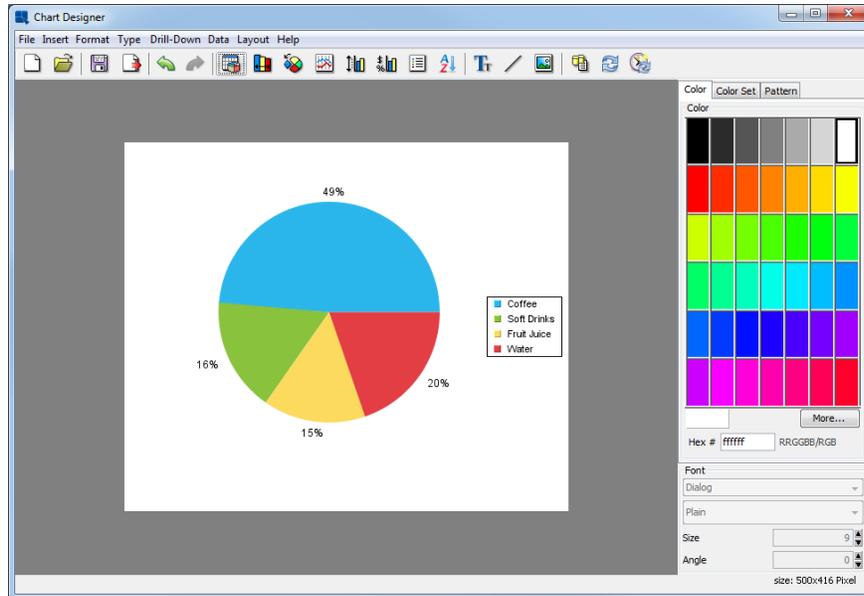


Chart Designer Showing Pie Chart

### Q.5.1.3. Stack Column Chart

Another way to display multi-dimensional data is to use a stack type representation to show contributions to a total. To convert your pie chart into a stack column chart, select *Stack Column* from the *Type* menu. Chart Designer will ask whether you want to redo data mapping for the new chart type. Click *Yes* and confirm the change by clicking *Done* in the “Select Data Mapping for 2D Stack Column Chart” dialog. The chart will then be converted to a stack column layout.

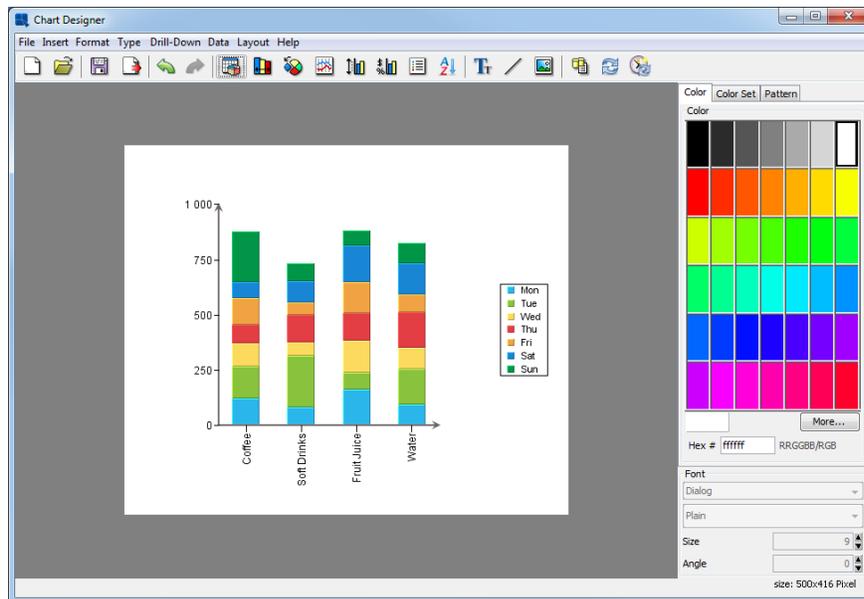
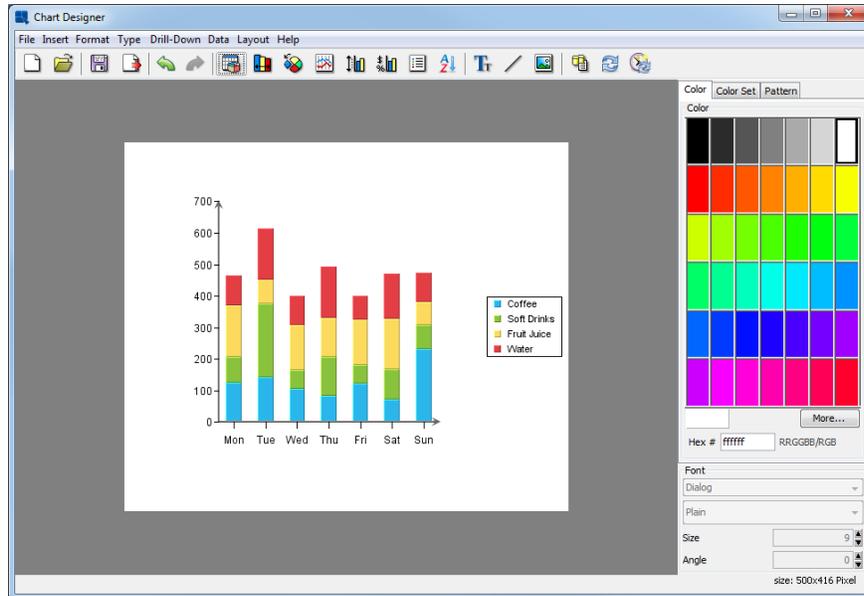


Chart Designer Showing Stack Column Chart

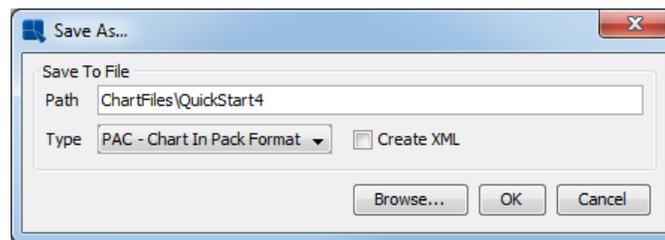
Notice that the chart now shows the columns as stacks made up of the values for each drink. Now click the *Data*

*Mapping* button  on the toolbar. In the data mapping window there is a new option called *Sum by*. This is set to the *Day* column. The *Sum by* column provides the individual stacks in these types of charts. Set *Sum by* to *Drink* and *Category* to *Day*. Click *Done* to return to the Chart Designer. You should see the following chart.



Stack Column Chart

Now that you have finished designing a chart, click the *Save* button on the Toolbar . A dialog will open prompting you to specify a location and file name for the chart.



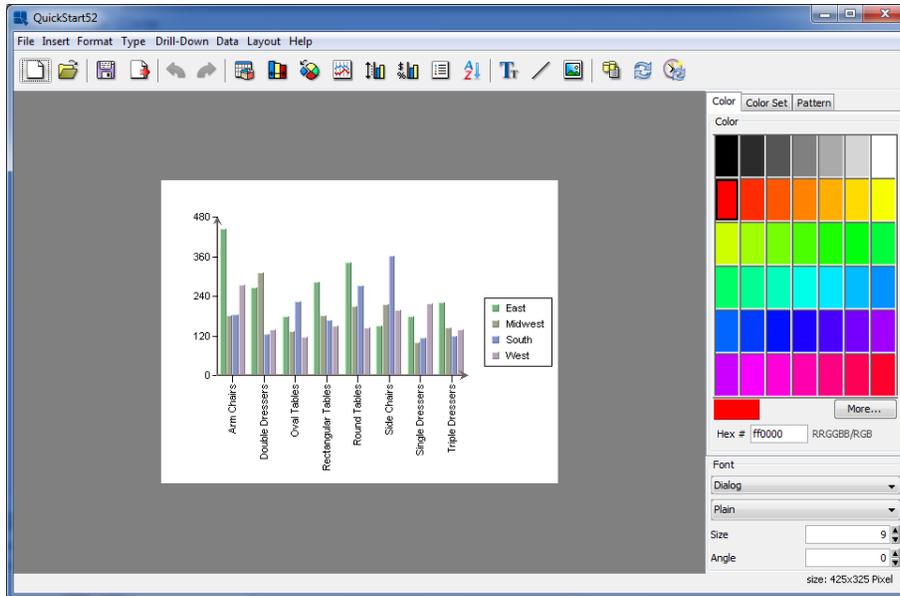
Save As Dialog

Enter a file name, and select to use either a PAC, CHT, or TPL format. For more about chart formats, see Section 4.2.6.2 - Saving Charts without Report Data. The chart will be saved in the /ChartFiles/ directory under the ERES installation by default. You will then be prompted to add the file to the Organizer. Select the *QuickStart Examples* Organizer folder and click *OK*. Close the Chart Designer.

## Q.5.2. Basic Chart Formatting

In this section, we will use some of the basic formatting features in the Chart Designer to change the look and feel of an unformatted chart. To begin you will need to add the chart to the Organizer.

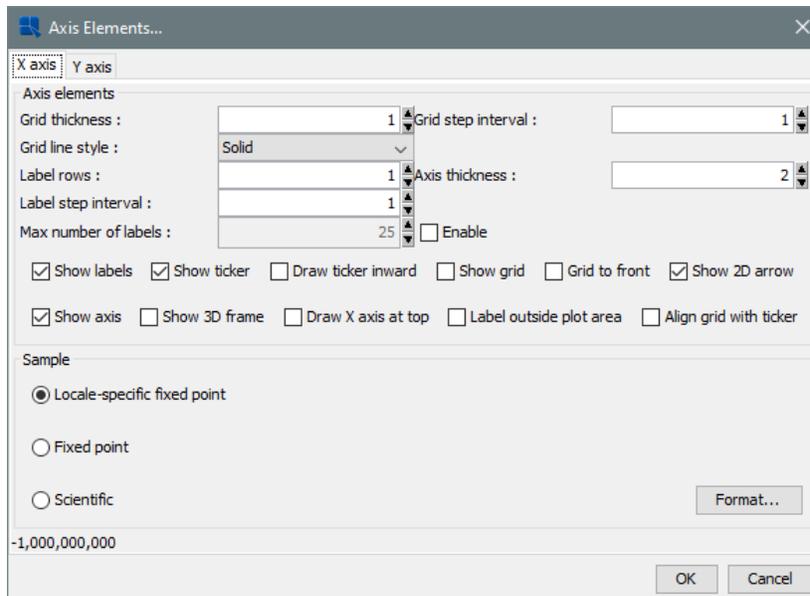
Following the same procedure as in Section Q.4.2.1 - Add a Report Template to Organizer, add the QuickStart52.tpl file under help/quickstart/templates into your project in the Organizer. Next, right click on the entry for this file in Organizer and select *Open File* from the pop-up menu. This will open the chart in the Chart Designer. As you can see its relatively un-formatted.



*Unformatted Chart in Designer*

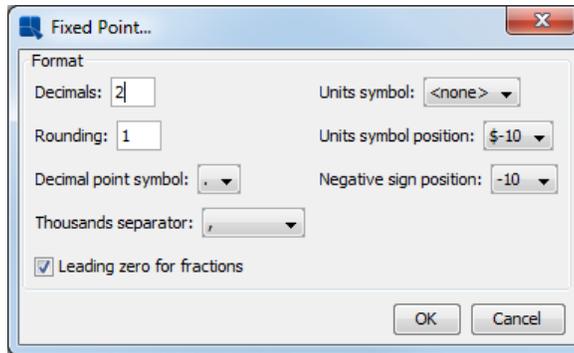
### Q.5.2.1. Axis Options

Many chart appearance options are controlled using the Axis Elements dialog. To bring up this dialog, click the *Format Value Elements* button on the toolbar . This will bring up a tabbed dialog allowing you to set different options for each chart axis. Click on the *Y Axis* tab to bring up options for the value axis.



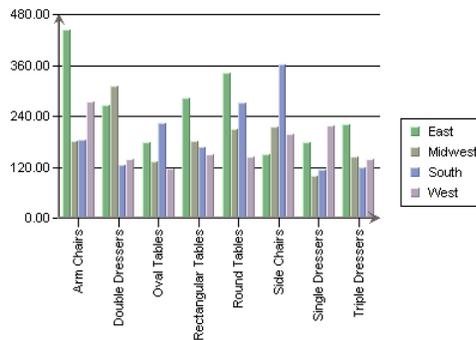
*Axis Elements Dialog*

Check the box marked *Show grid* to add grid lines to the Y Axis. Select *Fixed point* for the data format and click the *Format* button. This will bring up an additional dialog allowing you to set format options for the numeric data. Select the number of decimals as **2** and click *OK*.



*Numeric Format Dialog*

Click *OK* again to dismiss the axis elements dialog and you will see the specified changes reflected in the chart.

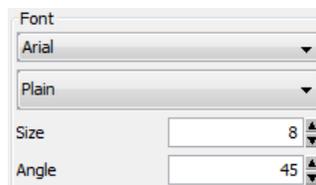


*Chart with new Axis Formatting*

### Q.5.2.2. Modify Color and Font

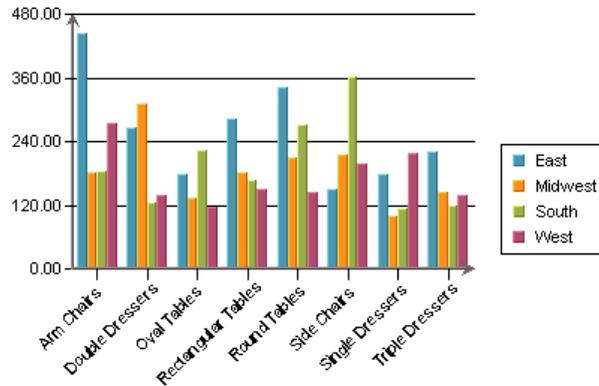
Chart colors can be selected individually or you can select predefined color set. In Chart Designer, click on the *Color set* tab on the right and select a color combination you like.

Now click on one of the X axis (category) labels. The lower left corner of the design window will reflect your selection. In the font dialog in the lower right corner, change the angle of the labels from 90 to 45 degrees and press Enter.



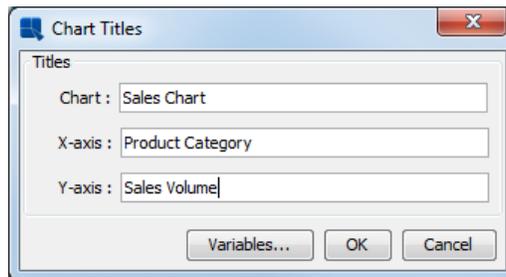
*Font Panel*

The chart labels will then rotate. You may need to adjust the position of the labels slightly. To do this, click and drag on a label. All the X axis labels will follow your cursor around the chart canvas.



### Q.5.2.3. Add Titles

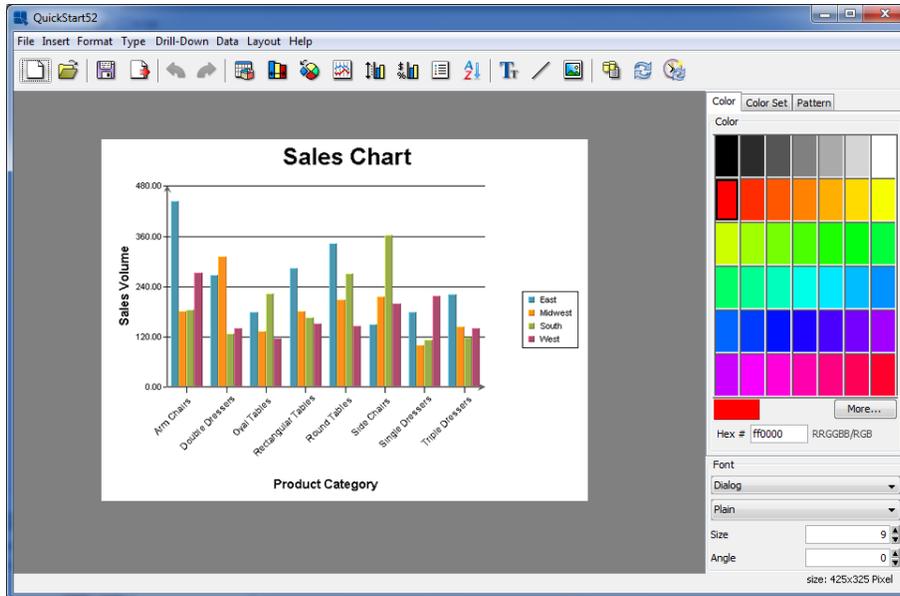
Next, we will add titles to the chart. To do this, select *Titles* from the *Insert* menu. This will bring up a dialog prompting you to enter a main title for the chart as well as titles for each of the axes.



*Chart Titles Dialog*

Enter any titles that you would like for the various elements and click *OK*. The titles will then be added to the chart. Titles are placed automatically but you can manually adjust their positions by clicking and dragging the text on the chart canvas.

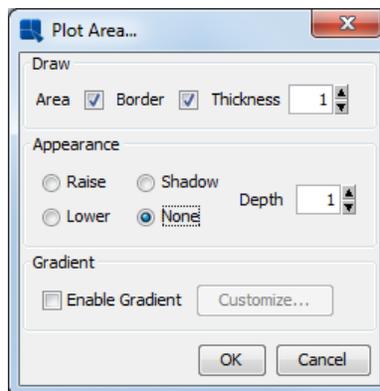
If you do not see chart titles, click and drag on the chart plot to move the chart. You should then be able to see the titles. You can adjust the font and font size of the chart titles by clicking them and changing the settings in the font panel.



*Chart With Titles*

#### Q.5.2.4. Customize Plot Area

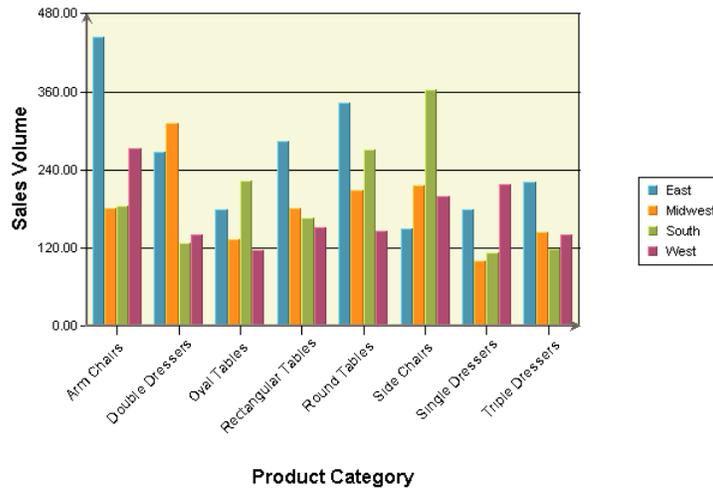
You can also customize the plot of two-dimensional charts, creating a separate background for the data points. To do this, select *Plot Area* from the *Format* menu. This will bring up a dialog allowing you to set display options for the chart plot.



*Plot Area Dialog*

Select to draw both the *Area* and the *Border* with a *Thickness* of **1**. Specify *None* for the *Appearance*. Once you have specified the options, click *OK* and the plot area for the chart will be modified. You can change the background color of the plot area by clicking to select it and then modifying the color in the color panel.

## Sales Chart



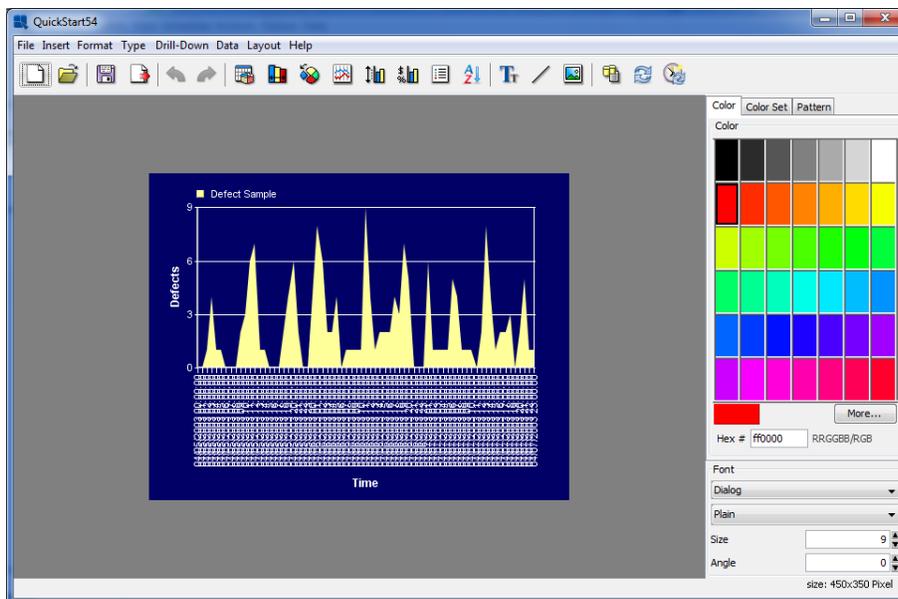
*Chart With Colored Plot Area*

Now save the changes you have made to the chart and exit the Chart Designer.

### Q.5.3. Advanced Charting Features

In this section we will use some of the charting features in ERES to provide a chart that quickly conveys salient information to the viewer. For detailed information about the features discussed in this section and other charting features, please see Section 4.2.4 - The Chart Designer Interface.

Following the same procedure as in Section Q.4.2.1 - Add a Report Template to Organizer, add the QuickStart54.tpl file under help/quickstart/templates into your project in the Organizer. Then right-click on the entry for this file in Organizer and select *Open File* from the pop-up menu. This will open the chart in Chart Designer.



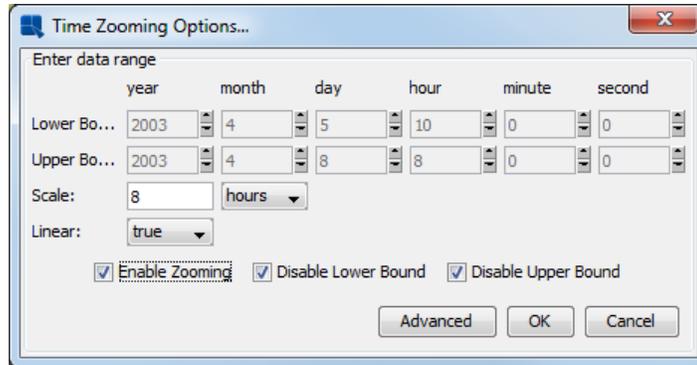
*QuickStart54.tpl in Chart Designer*

The chart shows hourly defect counts for a manufacturing process. Since the chart shows three days worth of data it is difficult to make out the individual points and the X axis labels are not legible because they are drawn on top of each other.

### Q.5.3.1. Time-Series Zooming

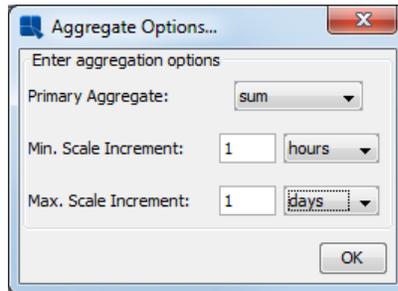
One way to improve the chart, is to use the time series zooming feature to aggregate the data into fewer data points. In this instance, let's look at the total number of defects for each eight hour shift, instead of plotting the data for each hour.

To turn on zooming, select *Time Zooming Options* from the *Format* menu. This will bring up the following dialog.



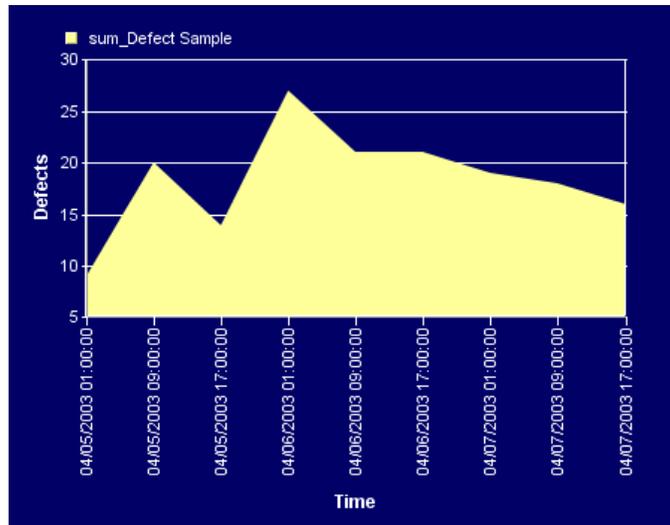
*Zoom Options Dialog*

In this dialog, set the *Scale* to **8 hours** and *Linear* option to **true**. Leave the *Disable Upper Bound* and *Disable Lower Bound* options checked. Check the *Enable Zooming* option. This will bring up a new dialog prompting you to select the aggregation for the zooming.



*Aggregation Options Dialog*

In this dialog, set the *Primary Aggregate* to be **sum**, the *Minimum Scale* to be **1 hour**, and the *Maximus Scale* to be **1 day**. Then click *OK* to return to the zoom options dialog, and *OK* again to return to apply the zoom setting. The chart now shows the total number of defects for each eight hour period.

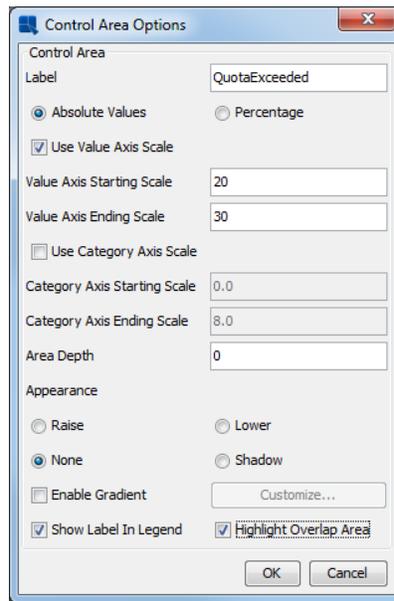


*Chart With Zooming Applied*

### Q.5.3.2. Control Areas

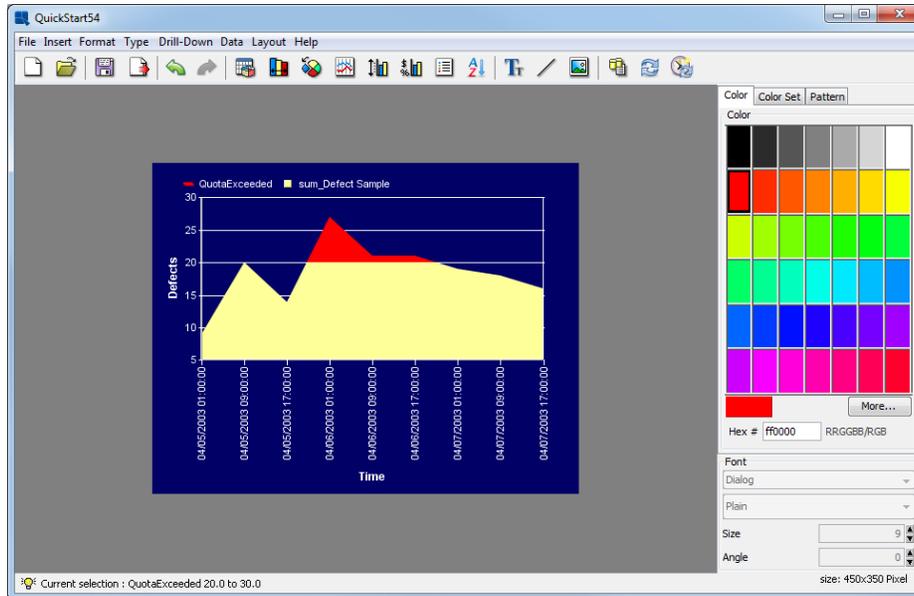
Next, we will use the control area feature to highlight any shift where there were more than 20 defects.

To add a control area, select *Control Area* from the *Insert* menu and then click *Insert* in the control range list. This will bring up a dialog allowing you to select options for the new control area.



*Control Area Dialog*

In this dialog, enter a name for your control area. Then check the option marked *Use Value Axis Scale*, and enter **20** for the *Starting Scale*, and **30** for the *Ending Scale*. Then at the bottom of the dialog check the options marked *Show Label In Legend* and *Highlight Overlap Area*. Then click *OK* to apply the control area. Notice how only the areas where the number of defects is colored. You can set the color, by selecting the overlapped area, and selecting a new color from the palette on the right-hand side.



*Chart With Control Areas Applied*

Once you have finished modifying the chart, click the *Save* button on the toolbar to save the changes you have made and close the Chart Designer.

## Q.6. QuickDesigner Reports

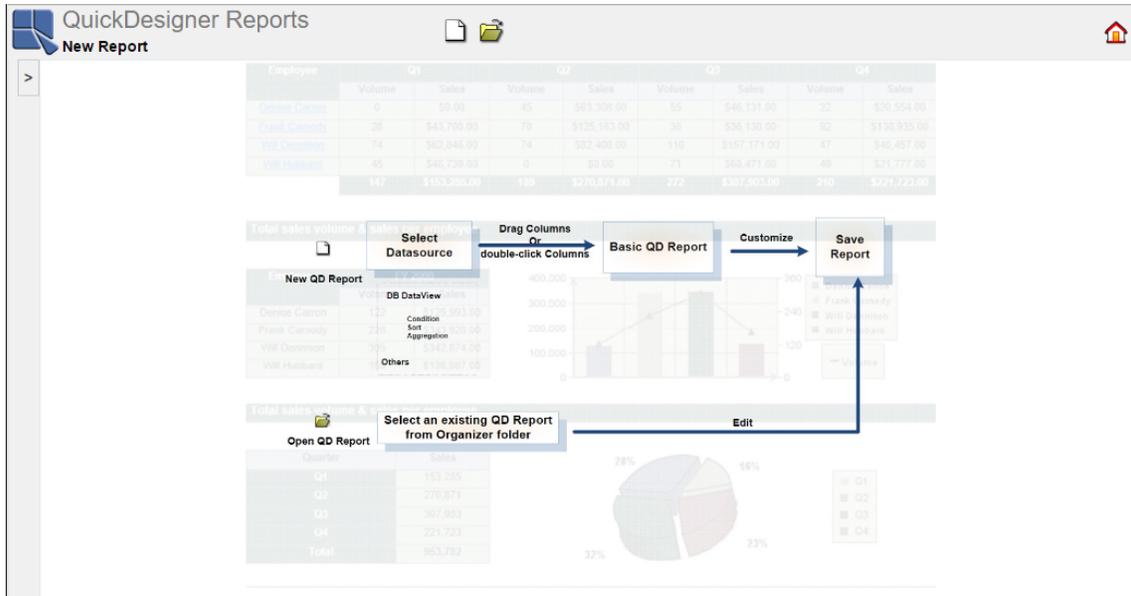
In the previous sections we looked at using Report Designer and Data Source Manager interfaces to query databases and create reports. ERES also provides the QuickDesigner Reports interface which provides a thin-client interface for creating ad hoc queries and reports. In this chapter, we will create a query and a report using QuickDesigner Reports.

To start QuickDesigner Reports, go to the ERES Start page and login using the user created in Section Q.2.1 - Create a User. When you login click the link labeled *QuickDesigner Reports* to launch QuickDesigner Reports.

### Q.6.1. Create a Query

QuickDesigner Reports allows you to build a report using any data source that you can with Report Designer. However, if you want to modify a data source, you can only query DataViews or modify existing DataView queries using QuickDesigner Reports.

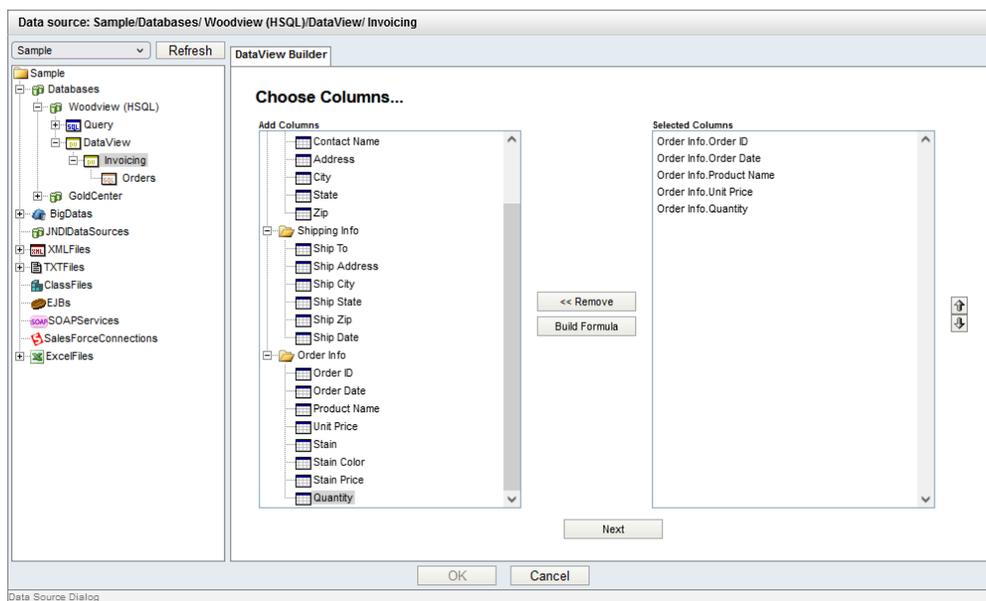
When QuickDesigner Reports opens, there are only two options accessible, to create a new report or open an existing one.



QuickDesigner Reports

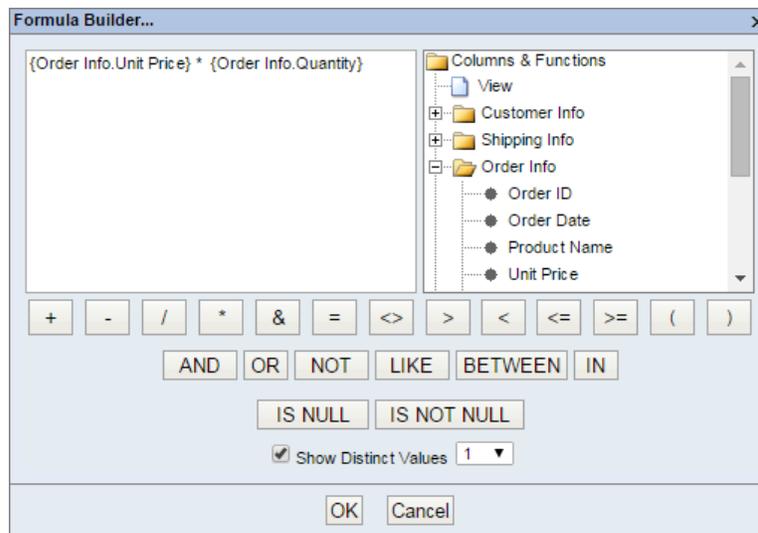
Click the  *New* button on the toolbar and the *Data Source Dialog* will open. Select the *Quickstart\_sample* data registry from the left top drop-down list (the data registry you created in Section Q.3.1 - Create a Data Registry). The tab below shows all of the defined data sources in the registry. Expand the nodes and select the *DataView Invoicing* that you created in Section Q.3.1.3 - Create a Data View. You will see all headings and fields of the view in *DataView Builder* in the right pane of *Data Source Dialog*. Add the following fields from the *Order Info* heading by clicking on them in the left side panel.

- Order ID
- Order Date
- Product Name
- Unit Price
- Quantity



Data Source Dialog

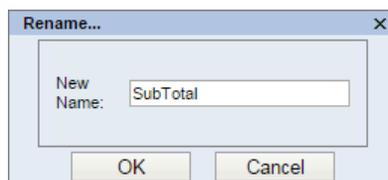
After adding the columns, click the *Build Formula* button to build a computed column. This will launch the Formula Builder in a new window.



*Formula Builder*

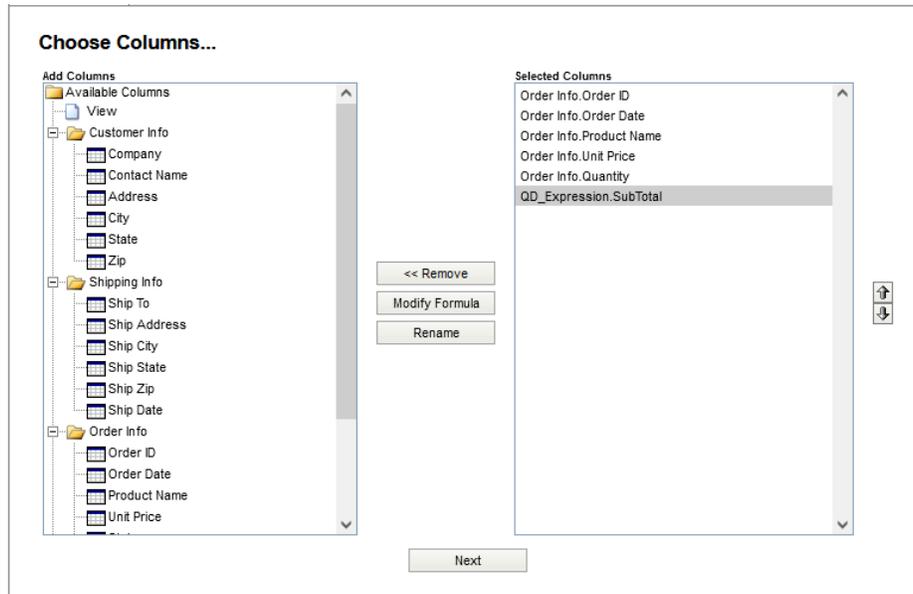
In the Formula Builder, expand the *View* node in the right-hand side. Then expand the *Order Info* node. Under this node click the *Unit Price* entry. The text will be added to the main formula window. Click the *multiply button* (\*). Finally, select the *Quantity* field. The finished formula should look like this: `{Order Info.Unit Price} * {Order Info.Quantity}`. Once you have finished, click *OK* to add the formula. A new column will be added in the column selection dialog.

To provide an alias for the expression column, select it and click the *Rename* button. This will bring up a dialog allowing you to specify the alias.



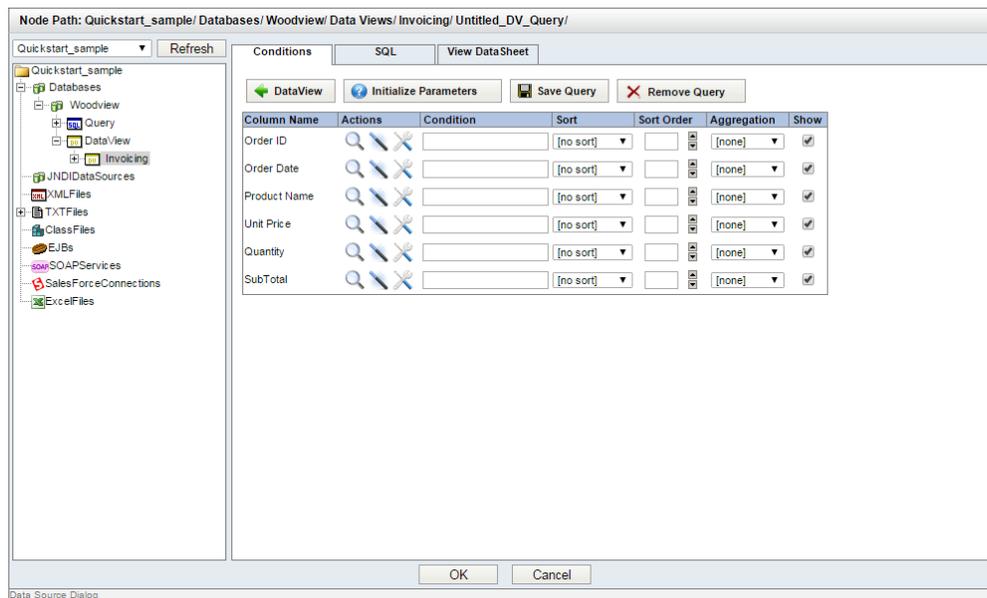
*Rename Dialog*

Enter **SubTotal** as an alias for the computed column and click *OK*. The computed column is renamed.



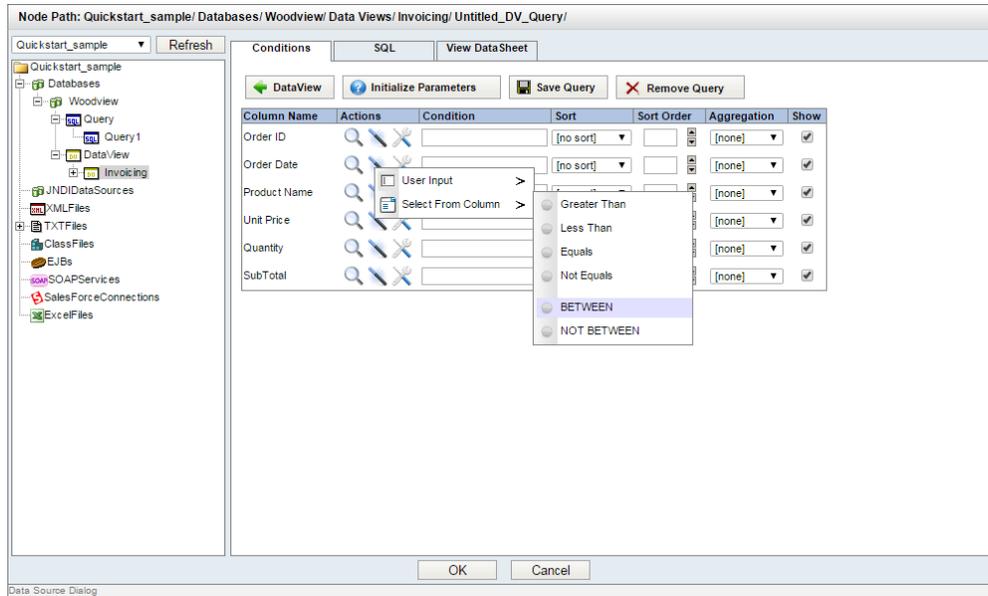
Renamed Column

Click the *Next* button to submit the column selection. The next dialog that opens allows you to set sorting, grouping, and conditions for the DataView query.



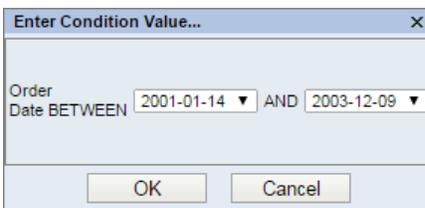
Conditions Dialog

Click the  *Condition Wizard* button for the Order Date field. Select the *Select From Column* option and then click the *Between* operator.



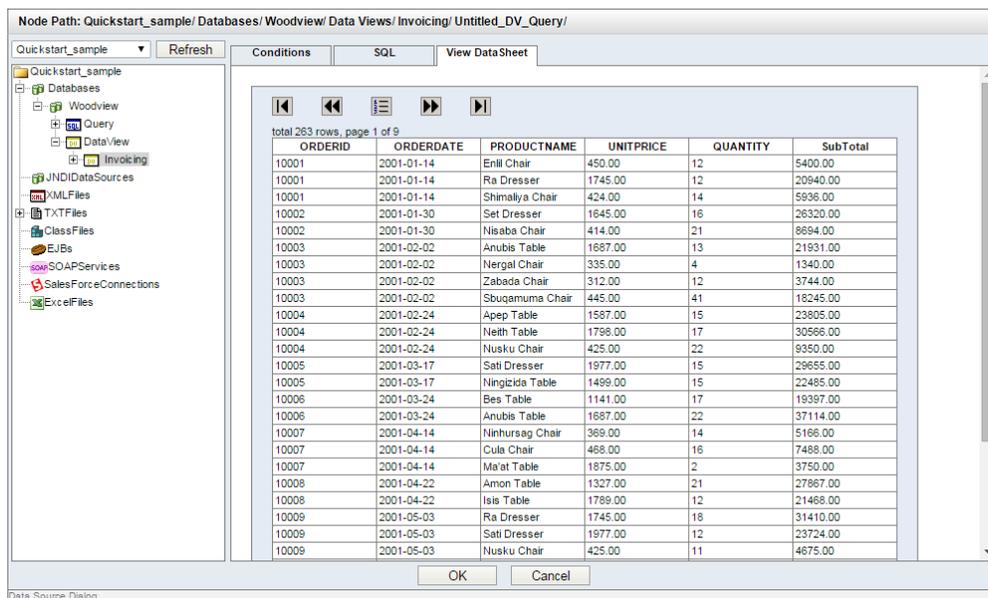
### Condition Operators

This will open a dialog allowing you to select a date range by which to filter the query.



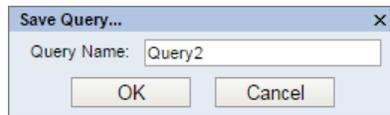
### Select Condition Value Dialog

Select the date range that you would like and click *OK*. The condition will be added for the Order Date field. Now you can preview the finished query by clicking the *View DataSheet* tab.



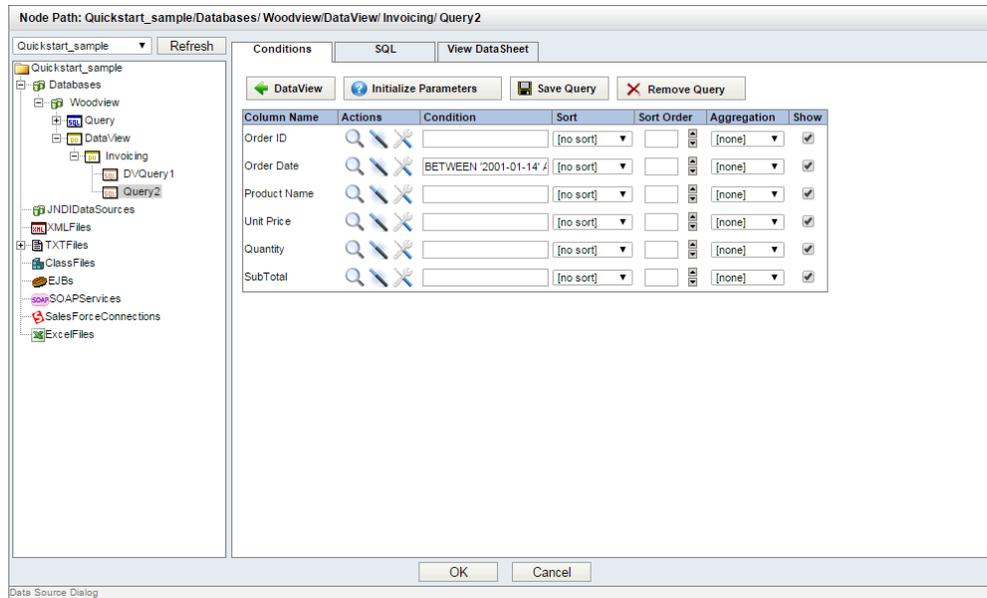
### Data Preview

Next, go back to the *Conditions* tab and click the *Save Query* button to save this *DataView* query back to the registry. This will bring up a dialog prompting you to select a name for the query.



*Save Query Dialog*

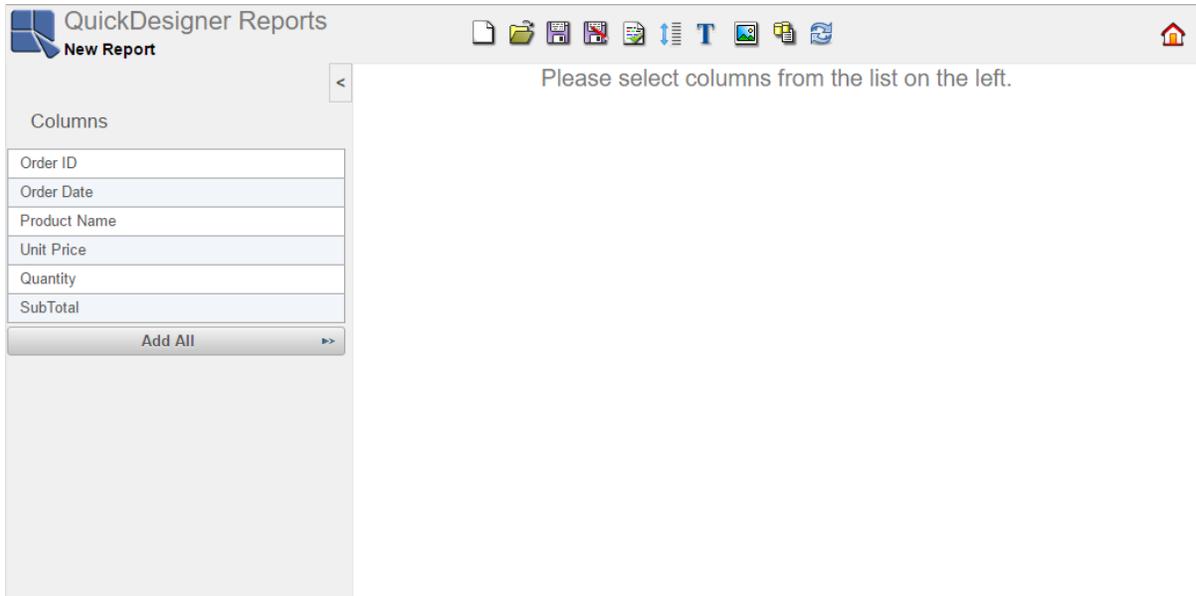
Enter the name you would like to use for the query and click *OK*. The new query appears under *DataView* node in the left pane of the *Data Source Dialog*.



*New DataView Query*

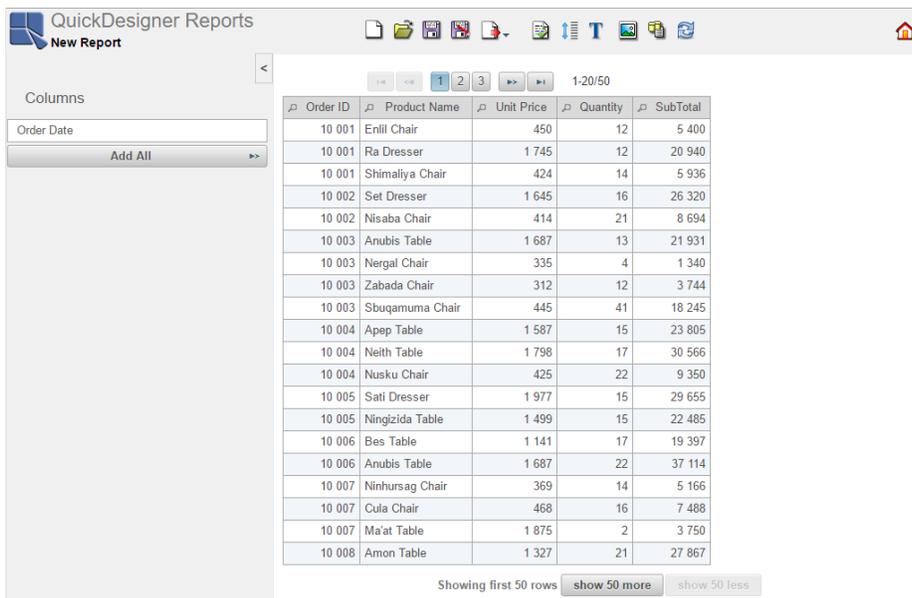
## Q.6.2. Create a Report

Now that you have completed the query, click *OK* to close the *Data Source Dialog* and to open the main page of QuickDesigner Reports.



*QuickDesigner Reports Main Page*

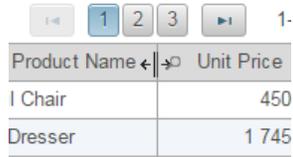
You can insert any column to the report by using a double-click on the column or by dragging it to the report. So, insert columns *Order ID*, *Product Name*, *Unit Price*, *Quantity*, *SubTotal*.



*Columns Selecting*

Collapse the left *Column* pane by clicking the  *Collapse* button and adjust the width of columns. To do this, move your mouse over the right side of the column header. You will see a double arrow. Then left click and drag it.

## QuickStart

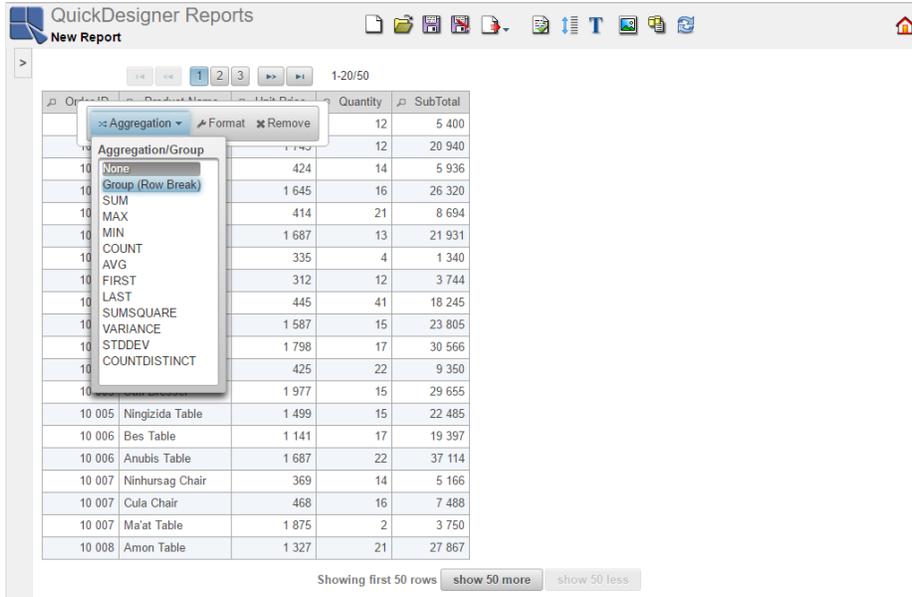


The screenshot shows a table with two columns: 'Product Name' and 'Unit Price'. The 'Product Name' column is wider than the 'Unit Price' column. Above the table, there are navigation buttons (1, 2, 3) and a page indicator '1-'.

Product Name	Unit Price
I Chair	450
Dresser	1 745

*Column Width Setting*

Right click on the *Order ID* header, select *Aggregation* and then *Group(Row Break)* option.

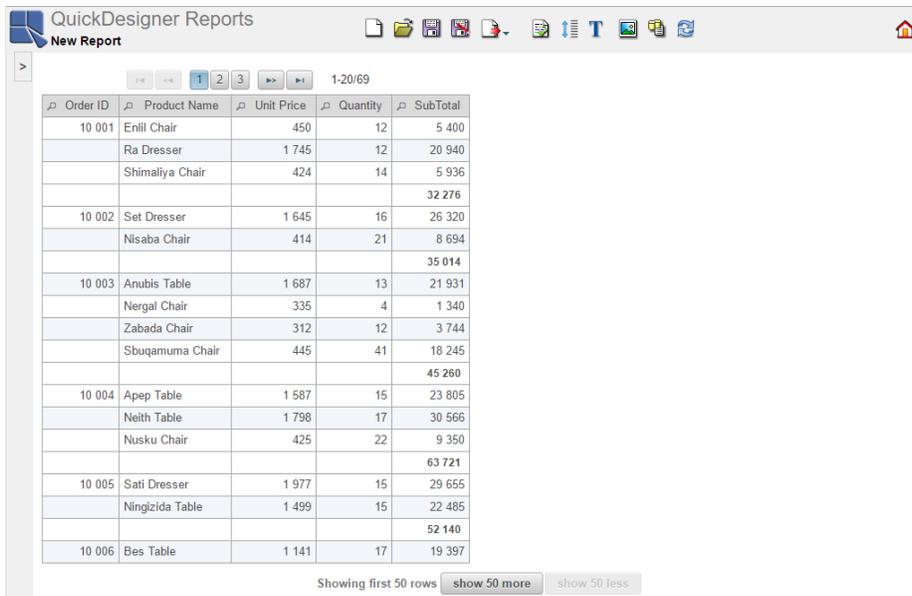


The screenshot shows the 'QuickDesigner Reports' interface with a table. A context menu is open over the 'Order ID' header, showing the 'Aggregation' option selected. The 'Aggregation/Group' sub-menu is also open, showing 'Group (Row Break)' as the selected option. The table below shows columns for Order ID, Product Name, Unit Price, Quantity, and SubTotal.

Order ID	Product Name	Unit Price	Quantity	SubTotal
10 001	Enlil Chair	450	12	5 400
	Ra Dresser	1 745	12	20 940
	Shimaliya Chair	424	14	5 936
				32 276
10 002	Set Dresser	1 645	16	26 320
	Nisaba Chair	414	21	8 694
				35 014
10 003	Anubis Table	1 687	13	21 931
	Nergal Chair	335	4	1 340
	Zabada Chair	312	12	3 744
	Sbuqamuma Chair	445	41	18 245
				45 260
10 004	Apep Table	1 587	15	23 805
	Neith Table	1 798	17	30 566
	Nusku Chair	425	22	9 350
				63 721
10 005	Sati Dresser	1 977	15	29 655
	Ningizida Table	1 499	15	22 485
				52 140
10 006	Bes Table	1 141	17	19 397

*Aggregation Dialog*

Right click on the *SubTotal* header, select *Aggregation* and then *SUM* option. Your report should look like this now:



The screenshot shows the 'QuickDesigner Reports' interface with a table. The 'SubTotal' column now shows the sum of the 'Quantity' column for each group. The table below shows columns for Order ID, Product Name, Unit Price, Quantity, and SubTotal.

Order ID	Product Name	Unit Price	Quantity	SubTotal
10 001	Enlil Chair	450	12	5 400
	Ra Dresser	1 745	12	20 940
	Shimaliya Chair	424	14	5 936
				32 276
10 002	Set Dresser	1 645	16	26 320
	Nisaba Chair	414	21	8 694
				35 014
10 003	Anubis Table	1 687	13	21 931
	Nergal Chair	335	4	1 340
	Zabada Chair	312	12	3 744
	Sbuqamuma Chair	445	41	18 245
				45 260
10 004	Apep Table	1 587	15	23 805
	Neith Table	1 798	17	30 566
	Nusku Chair	425	22	9 350
				63 721
10 005	Sati Dresser	1 977	15	29 655
	Ningizida Table	1 499	15	22 485
				52 140
10 006	Bes Table	1 141	17	19 397

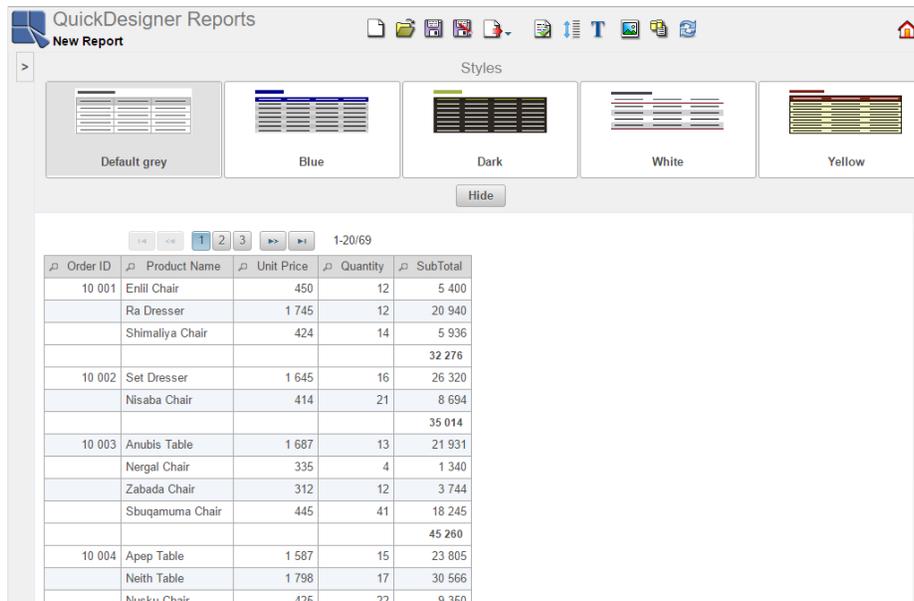
*Report with Aggregation*

## Q.6.2.1. Format Report Elements

QuickDesigner Reports allows you to change the style of the report by using several predefined styles. Click the

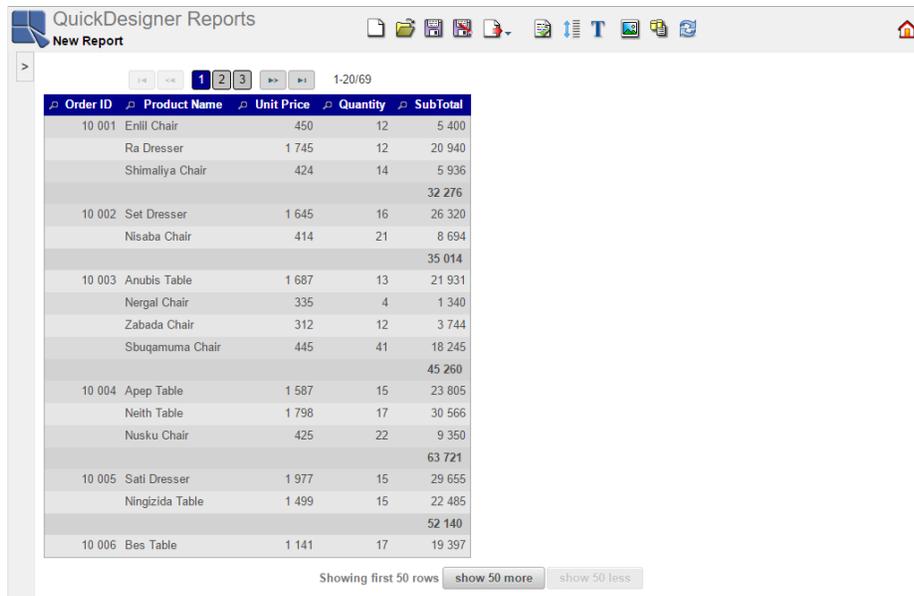


*Styles* button on the toolbar. The *Styles* dialog will open above the report.



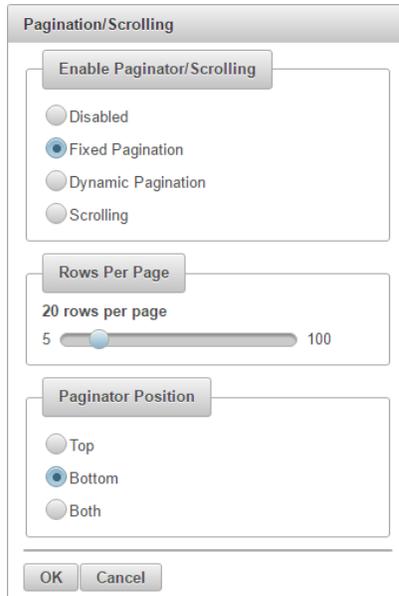
*Styles Dialog*

Select the style you like, e.g. the *Blue* style, and close the *Styles* dialog by clicking *Hide* button.



*Style Applied*

Next, place a pagination under the report. Click  *Pagination/Scrolling* button on the toolbar and *Pagination/Scrolling* dialog opens. Select *Bottom* option for *Paginator Position* and click *OK*.



*Pagination/Scrolling Dialog*

Now, add a report title. Click the **T** *Report Title* button on the toolbar and the *Report Title* dialog appears. Type **Order Information** as the title and click  *Apply* button.



*Report Title Dialog*

The report title appears on the left side of the report by default. Move it to the center of the report. Click on the title, hold the mouse button and move the mouse slightly. Dashed rectangles appear. Move the title to the middle rectangle and release the mouse button.



*Title Placement*

Your report should look like the report below. For more detail about all the formatting option available in Quick-Designer Reports, see Section 4.3.4 - Format Report Elements.

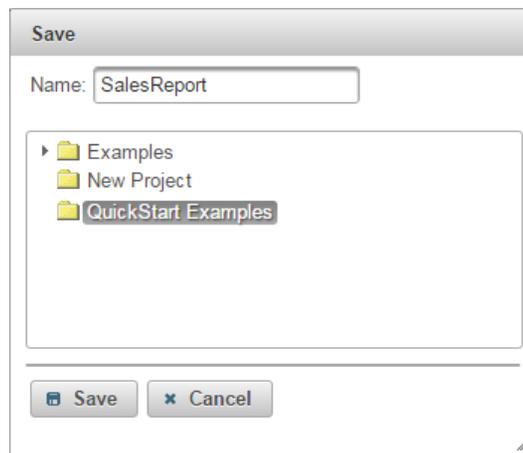
## Order Information

Order ID	Product Name	Unit Price	Quantity	SubTotal
10 001	Enil Chair	450	12	5 400
	Ra Dresser	1 745	12	20 940
	Shimaliya Chair	424	14	5 936
				32 276
10 002	Set Dresser	1 645	16	26 320
	Nisaba Chair	414	21	8 694
				35 014
10 003	Anubis Table	1 687	13	21 931
	Nergal Chair	335	4	1 340
	Zabada Chair	312	12	3 744
	Sbuqamuma Chair	445	41	18 245
				45 260
10 004	Apep Table	1 587	15	23 805
	Neith Table	1 798	17	30 566
	Nusku Chair	425	22	9 350
				63 721
10 005	Sati Dresser	1 977	15	29 655
	Ningizida Table	1 499	15	22 485
				52 140
10 006	Bes Table	1 141	17	19 397
				1 20/69

*Report With Changes Applied*

### Q.6.2.2. Save the Report

Now that you have finished formatting, you can save the report you have created back to Organizer so that other users can view it. To save the report, click the *Save* button on the toolbar . This will bring up a dialog allowing you to specify a name for the template.



*Save Report Dialog*

Enter a name for the report, and select the *QuickStart Examples* project that you created in Section Q.2.2.2 - Add a Project. Click *Save* button to save the report.

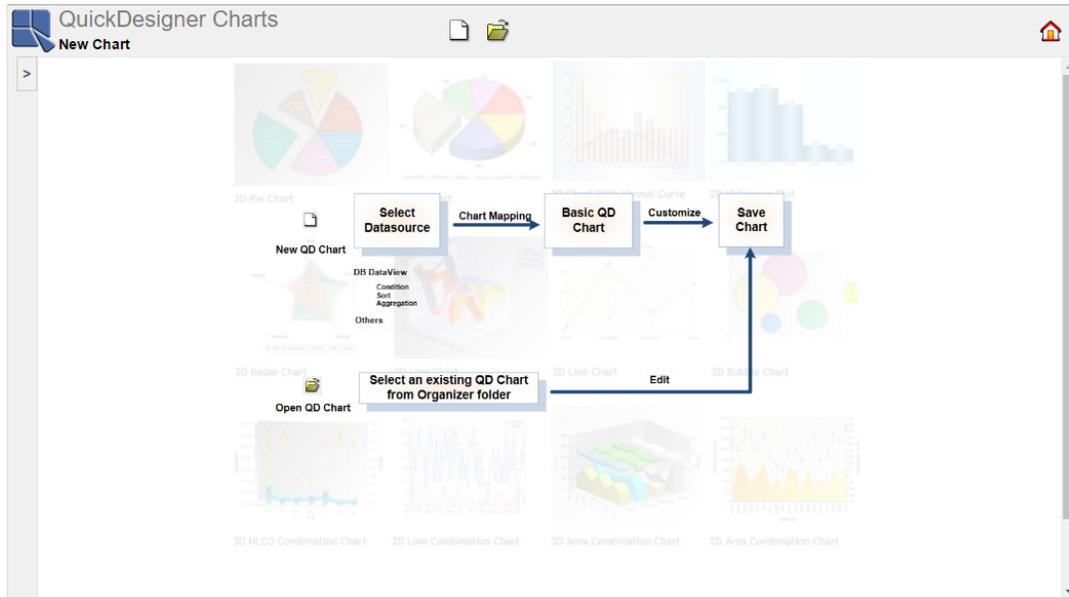
## Q.7. QuickDesigner Charts

QuickDesigner Charts is a thin-client ad-hoc charting interface. It provides users a scaled-down design tool to create charts. With QuickDesigner Charts, end users can easily select, filter, and present data without mastering database structures, all with zero client download.

For more detailed information about all the chart options available in QuickDesigner Charts, see Section 4.4 - QuickDesigner Charts.

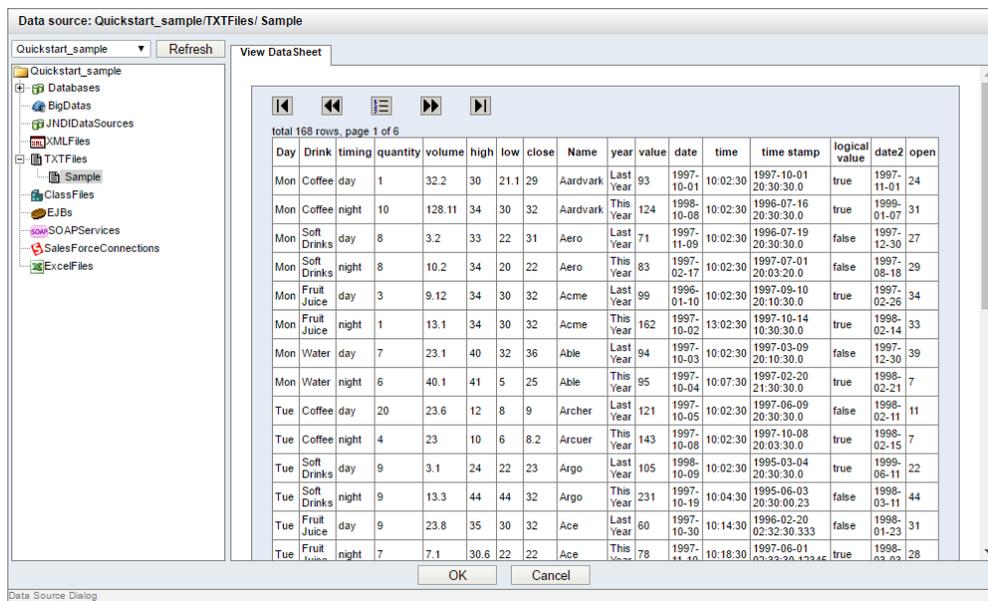
## Q.7.1. Create a Chart

To start QuickDesigner Charts, go to the ERES Start page and click the link labeled *QuickDesigner Charts*. QuickDesigner Charts should open, and you can create a new chart or open an existing one.



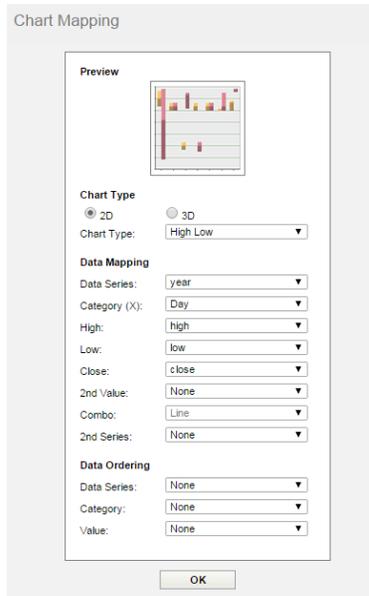
QuickDesigner Charts

Click the *Create New Chart* button on the toolbar . You will be taken to the *Data Source Dialog*, where you can select a data registry and a data source for a new chart. Select the *Quickstart\_sample* data registry from the top left drop-down list, and then select the *Sample.dat* text file data source that you created in Section Q.3.1.4 - Setup a Text Data Source. You can see records of the text file in the *View Data Sheet* window on the right side of the *Data Source Dialog*. Click the *OK* button to continue.



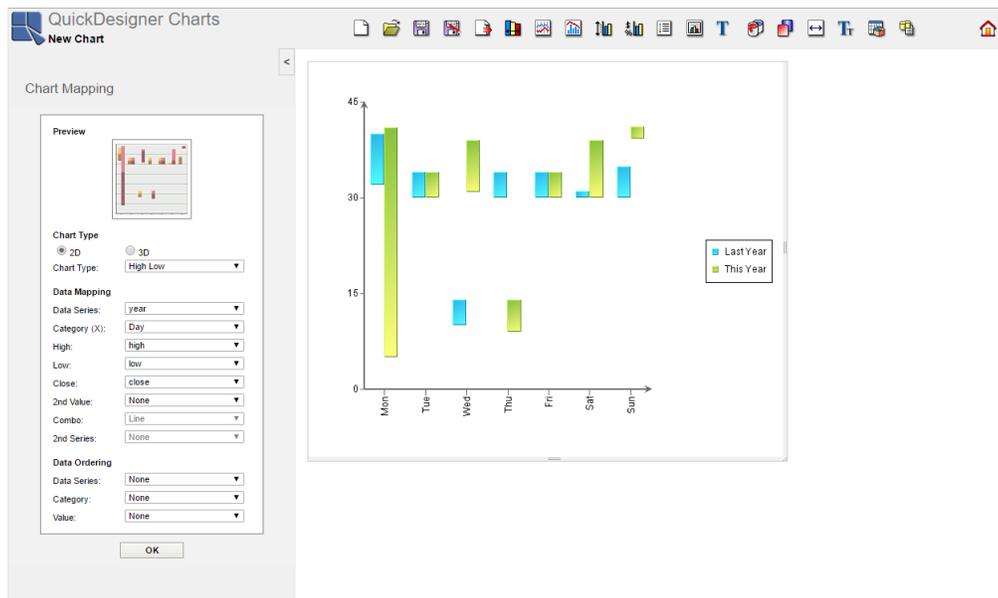
Data Source Dialog

You can see the *Chart Mapping* dialog that allows you to select the chart type and mapping options. Select to create a **2D High Low** chart using the first options in the dialog.



*Chart Mapping Dialog*

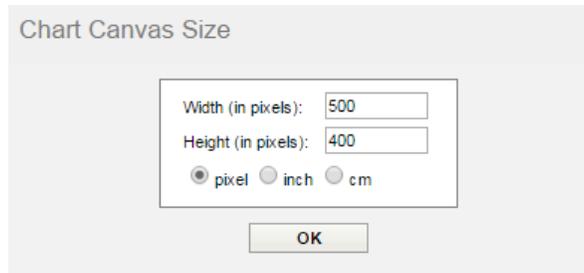
For the mapping options select **year** as the *Data Series*, **Day** as the *Category*, **high** as the *High*, **low** as the *Low*, and **close** as the *Close*. Click *OK* and the chart will open on the right side of QuickDesigner Charts.



*QuickDesigner Charts Interface*

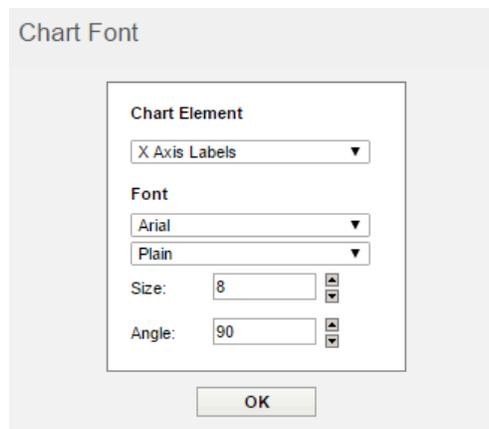
### Q.7.1.1. Format Chart Elements

Click the *Canvas Size* button on the toolbar . This will bring up a dialog allowing you to set the size of the chart canvas. Set the size to be **500** by **400** pixels and click *OK*.



*Chart Canvas Size Dialog*

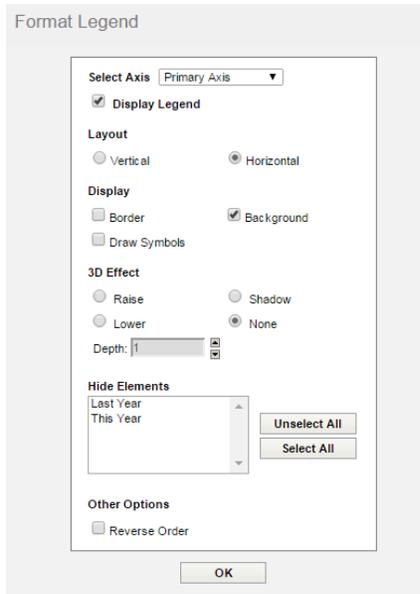
Next, click the *Font Settings* button on the toolbar . This will bring up a dialog allowing you to set the font options for the different text elements in the chart. From the drop-down list in the top of the dialog select **X Axis Labels**. Then set the font to be **Arial**, **8 pt**, and **Plain**.



*Chart Font Dialog*

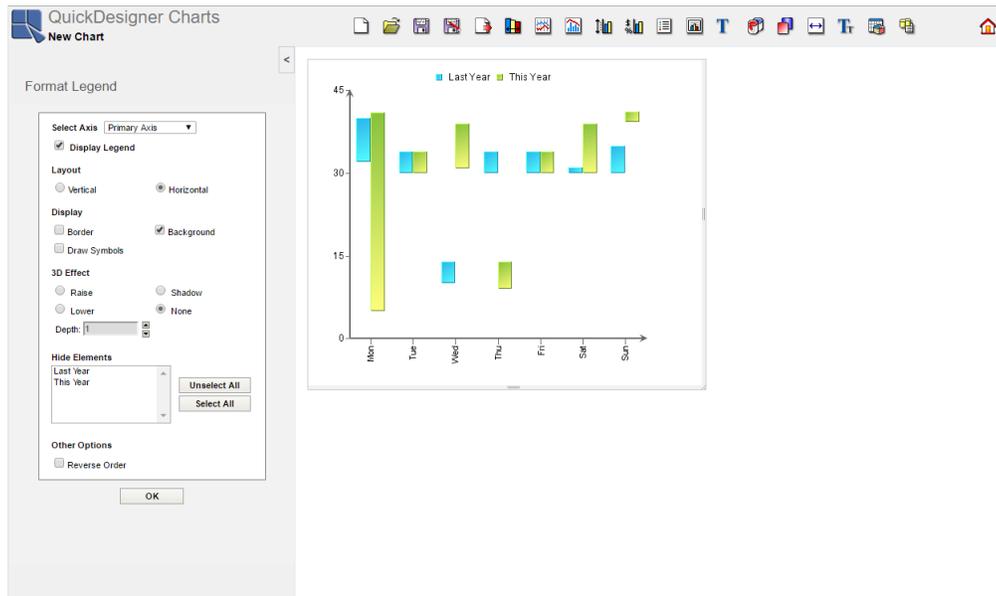
Now repeat this process for the Y axis labels and the legend text, setting them all to be **Arial**, **8pt** and **Plain**. Click *OK*.

Next, click the *Format Legend* button on the toolbar . This will bring up a dialog allowing you to set display options for the chart legend. Change the *Layout* option to **Horizontal**, uncheck the *Border* option, and change the *3D Effect* option to **None**. Click *OK* to apply the changes.



*Format Legend Dialog*

Now left click on the chart, hold the mouse button down and move the chart to keep the best position. Do the same with the legend. You can also set the plot area size by dragging the sides or dragging the corners of the chart. To do this, mouse over the side or the corner of the chart until you see the arrow. Then left click and drag.



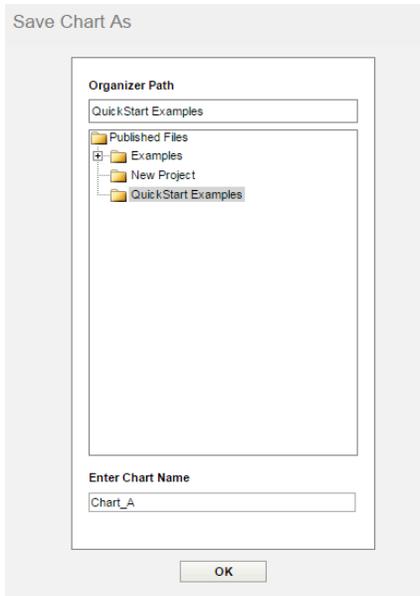
*Finished Chart*

For more detailed information about all the chart formatting options available in QuickDesigner Charts, see Section 4.4.5 - Data Formatting.

### Q.7.1.2. Save the Chart

Now that you have finished formatting the chart, you can save it back to the Organizer. To save the chart, click the

*Save* button on the toolbar . This will bring up a dialog allowing you to specify a name for the chart.



*Save Chart Dialog*

Select the *QuickStart Examples* project that you created in Section Q.2.2.2 - Add a Project and enter a name for the chart. Click *OK* to save the chart. The window will then give you a message to indicate that the chart was saved successfully. Click *OK* to close the dialog.

## Q.8. Online Maps

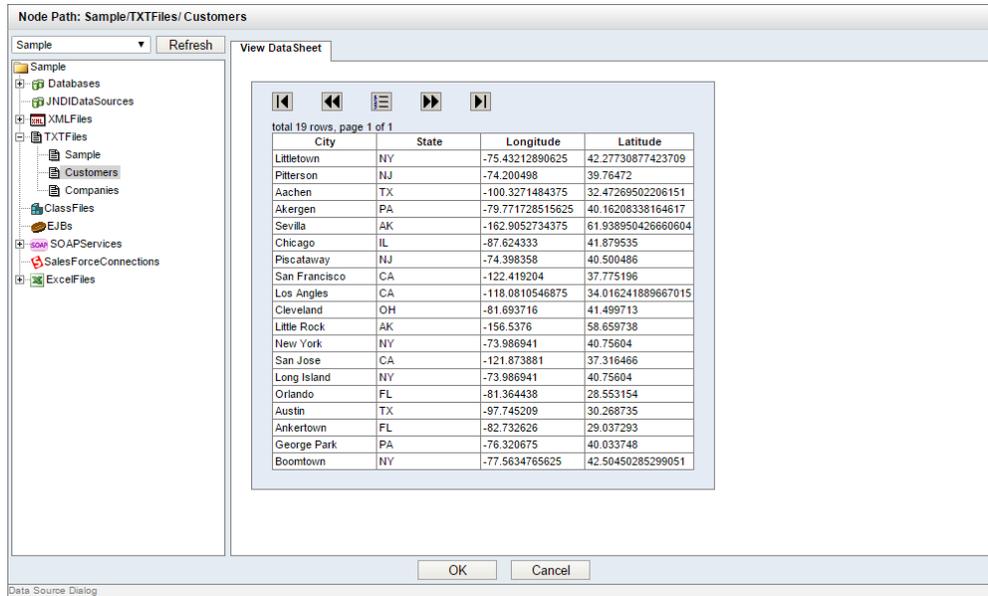
An alternative way of presenting your data is by using the ERES maps feature. ERES maps are designed to report geographical data from data sources. They fetch geographical data from a data source and mark them on a map. There are two types of maps: Online Maps and SVG Maps. We will work with Online Maps at first and then with SVG Maps. (SVG Maps are described in Section Q.9 - SVG Maps.)

### Q.8.1. Create Coordinates

Before creating an Online Map, you will need to create a coordinates file. Coordinates files contain coordinates of places that you want to mark on the map. For more details about coordinates, see section Section 5.2.4.2 - Create Coordinates.

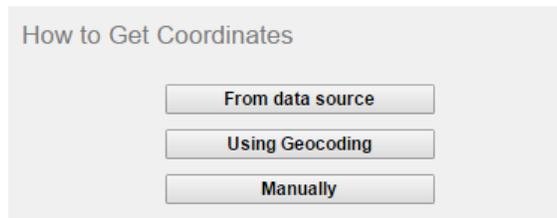
After launching Online Maps, you can create a new map, open some existing map, or create/edit coordinates.





*Selecting Data Source*

You will be prompted to choose a method of obtaining coordinates from the data source. There are three methods for this purpose (to learn more about how to choose the right method, see section Section 5.2.4.2 - Create Coordinates). The best method for our `Customers` data source is to obtain the coordinates directly from the data source, so click the *From data source* option.



*How To Get Coordinates*

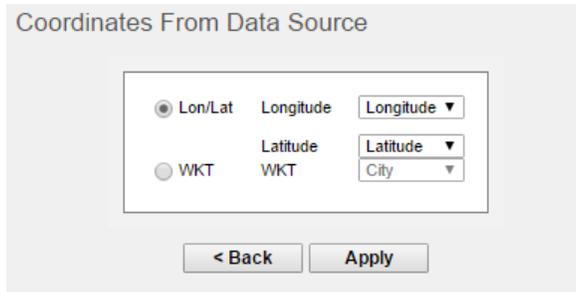
Next step is to select data source fields that contain the coordinates.



### Note

This step would be different if you chose some other method of obtaining coordinates (all possibilities described in Section 5.2.4.2 - Create Coordinates).

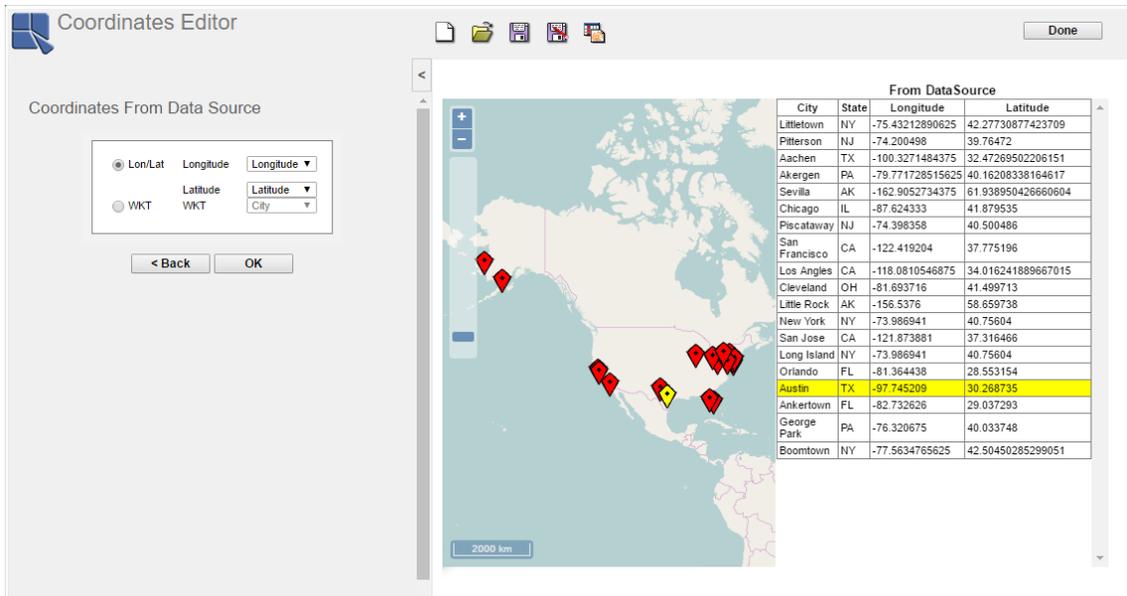
Each Coordinates data record consists of latitude and longitude (obtained automatically or manually) and a Point ID. The **Point ID** is an identifier of the place (e.g. city name, branch name, etc) and it is loaded from the data source you chose to create Coordinates from. The Point ID can consist of several **Point ID fields** - each data source column creates one field (except the fields, which contain longitude and latitude). Our data source has four columns `City`, `State`, `Longitude`, and `Latitude`. Therefore, the Point ID will consist of two fields - `City` and `State`. The upper option is labeled `Longitude` and the lower option is labeled as `Latitude`. Just set the upper field to data source field that contains information about longitude (in this case, it is the `Longitude` field) and the lower field to the data source field that contains latitude (`Latitude`). The following screenshot shows the correctly set dialog:



Select Fields With Coordinates

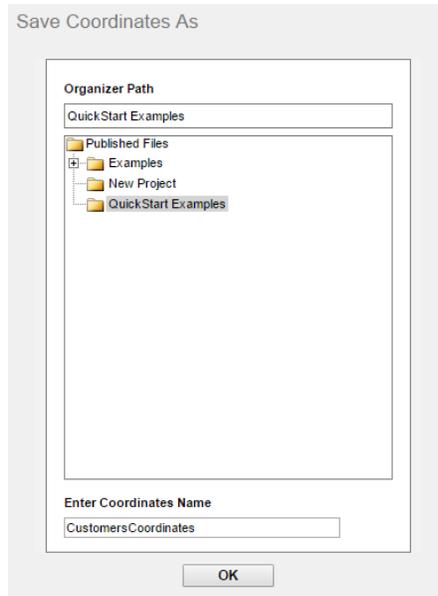
Your dialog should look like the one in the previous screenshot. If it does, click the *Apply* button.

You can see a map with coordinates table in the right pane of the Coordinates Editor. When you move the mouse arrow over one of the map markers or over a Point ID Table row, it will highlight the map marker and Point ID Table row that match together. (For more information see Section 5.2.4.3 - Coordinates Editor Interface).



Coordinates Editor

Now you may save the coordinates file. To save, click on the  *Save* icon on the toolbar.



*Save Coordinates*

Type **CustomersCoordinates** into the *Enter Coordinates Name* textfield and select the *QuickStart Examples* Organizer folder created in Section Q.2.2.2 - Add a Project. Click *OK* to save the coordinates file.

Close the Coordinates Editor and go back to the Online Maps designer by clicking the *Done* button on the right side of the toolbar.

## Q.8.2. Create Online Map

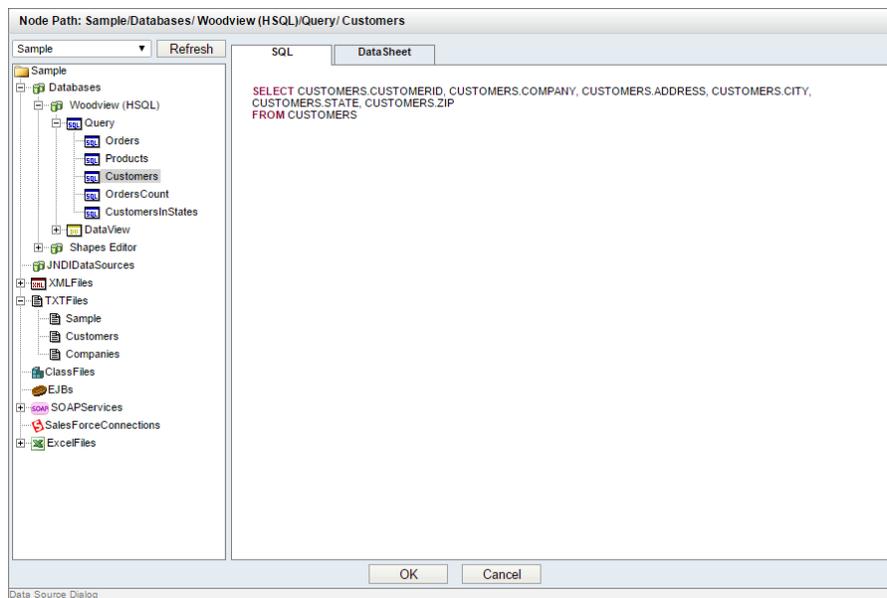


### Note

If you have not practiced creating coordinates, please do so. It may be extremely difficult to create an Online Map without proper coordinates.

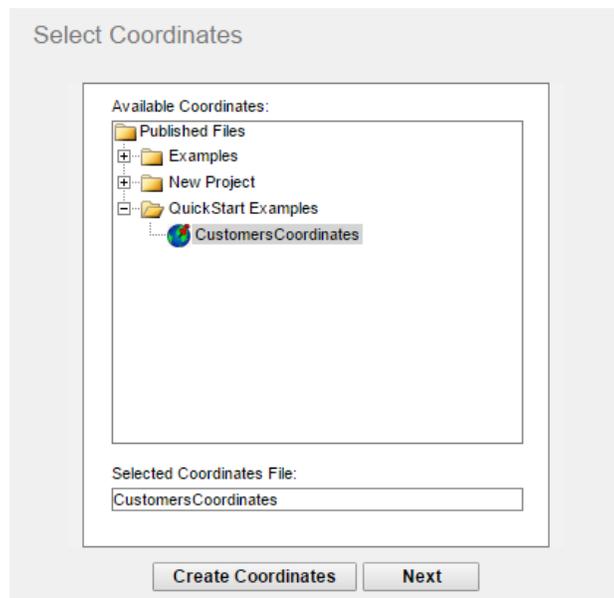
Open the *Data Source Dialog* by clicking the  *New Map* icon on the Online Maps toolbar. Select the *Sample* data registry.

We will use a different data source than the one we used for the coordinates in previous section. Open the *Databases* node, then the *Woodview (HSQL)* and the *Query* node and select the *Customers* query. Click *OK* to close the *Data Source Dialog*.



*Selecting Data Source*

Now, you will have to select a coordinates file from the Organizer. You will be able to see all projects, folders, and coordinates that were inserted into ERES Organizer. To select the coordinates file we created in previous section, open the `QuickStart Examples` project (click on it), then select the `CustomersCoordinates` coordinates and click *Next*.



*Selecting Coordinates File*

Next, you will need to set point mapping. To do this, you need to map at least one field from the ERES Online Maps data source to a Point ID field in the coordinates file data source. Both fields should contain the same type of information (for example, city name). But you can map more than one field if that is required to uniquely identify the location on the map (for example, city and state). When determining the map location of the ERES Online Maps data source record, the mapping function automatically searches for the same value of the mapped field(s) in ERES Online Maps data source and the mapped field(s) of the coordinates data source, and only the matched points will be visible on the map. In a nutshell, the coordinates data source contains information about location of data points on the map and ERES Online Maps data source contains the data that you want to report on the map. You may read more about point mapping in Section 5.2.5 - Coordinates Mapping.

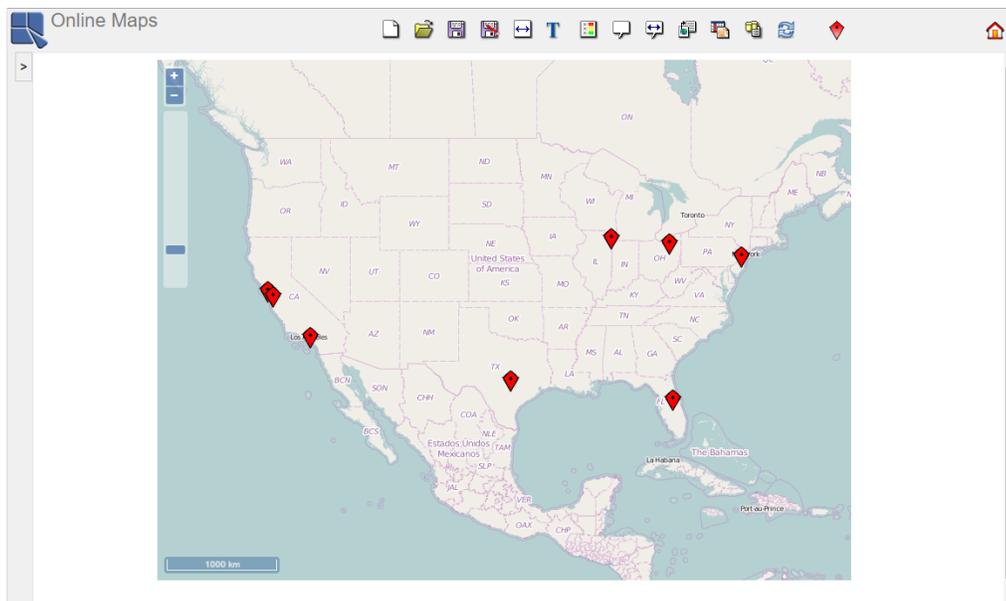
In this tutorial, we will map the `City` field to the `CITY` field and the `State` field to the `STATE` field. Just map the fields and click *Apply*.

**Coordinates to Map Data Mapping**

<u>Coordinates</u>	<u>Map Data</u>
City	CITY ▼
State	STATE ▼

### *Set Point Mapping*

Setting point mapping was the final step of the Maps Wizard. You should be able to see the Online Maps toolbar and the Online Map with some marker on it (the left pane is collapsed by clicking the  *Collapse* button on the screenshot).



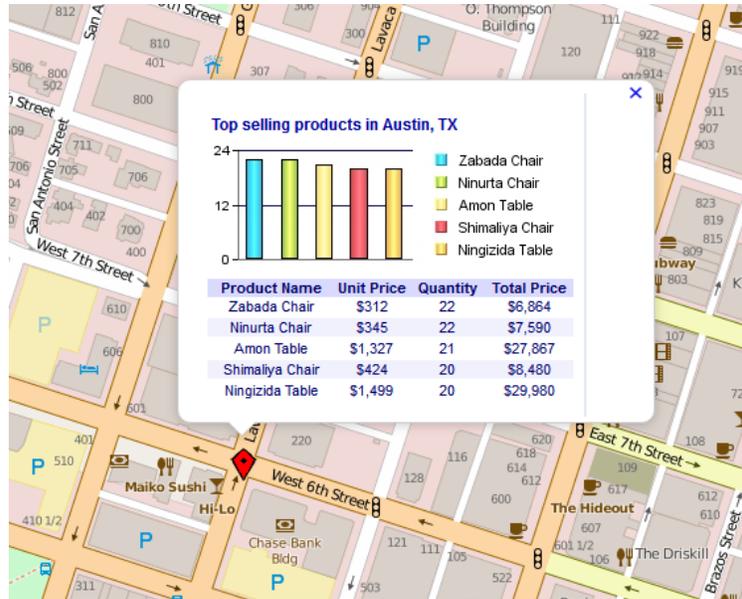
### *Online Maps Designer*

The Online Maps toolbar allows you to configure the Online Map. You can immediately see results of all changes you make on the Online Map.

Save the map by clicking on the  *Save* icon on the toolbar. Type **CustomersOnlineMap** into the *Enter Map Name* textfield. Select the **QuickStart Examples** Organizer folder and click *OK* to save the map.

## **Q.8.2.1. Configure Tooltips**

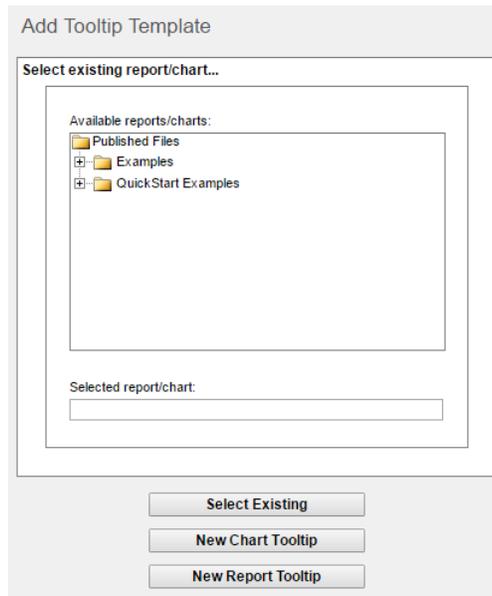
Tooltips show a brief report for a particular map marker. When the tooltips are enabled, each marker on the Online Map has its own tooltip. After setting the tooltip template (report or chart) for the Online Map, upon mouse over any map marker, the tooltip bubble will appear, displaying data relevant to the particular marker. Detailed information about the tooltips can be found in Section 5.2.6.7 - Tooltips.



Example: Google Map with Tooltip

Lets configure the tooltips for our CustomersOnlineMap that we created in the previous QuickStart section.

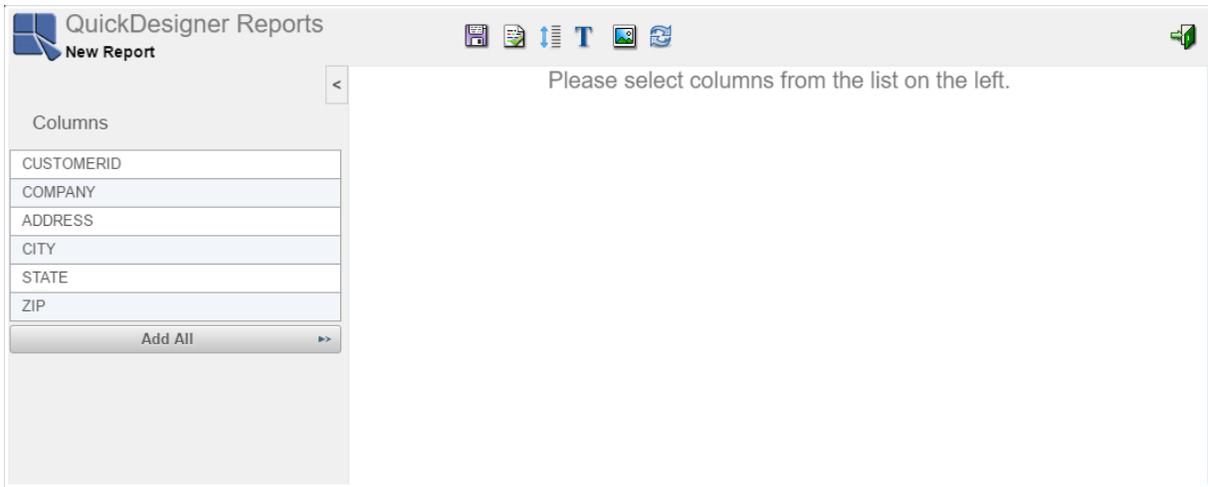
Click on the  *Tooltip template* icon on the toolbar. The *Add Tooltip Template* dialog will appear.



Add New Tooltip

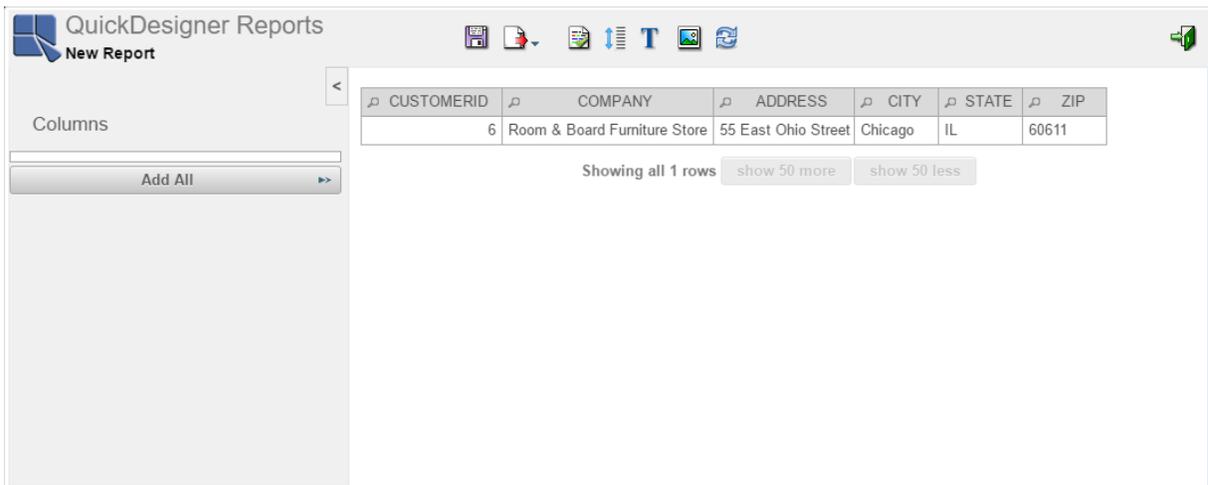
You can select an existing tooltip template or create a new chart or report tooltip template. When you choose to select new chart template, QuickDesigner Charts will launch. When you choose to select new report template, QuickDesigner Reports will launch. The launched QuickDesigner Report or QuickDesigner Charts will use the Map Data as a datasource to create a report or chart as a tooltip. In our example, Customer Query under Woodview(HSQL) Query node is the datasource used by Quick Designer to create tooltip. For now, we will just make a very simple report in QuickDesigner Reports. Visit Section Q.6 - QuickDesigner Reports, or the Section 4.3 - QuickDesigner Reports for information about how to use QuickDesigner Reports.

Click the *New Report Tooltip* option to open QuickDesigner Reports.



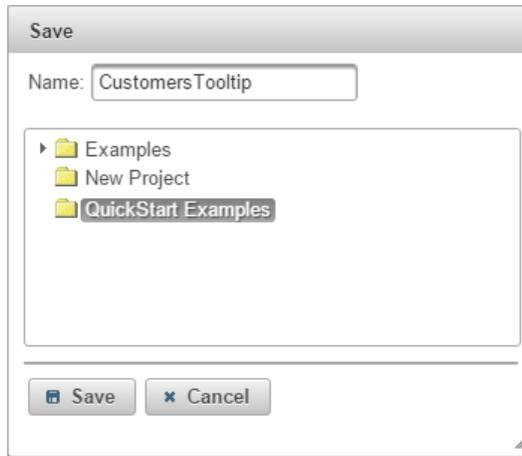
*QuickDesigner Reports*

Click *Add All* to add all columns to the report.



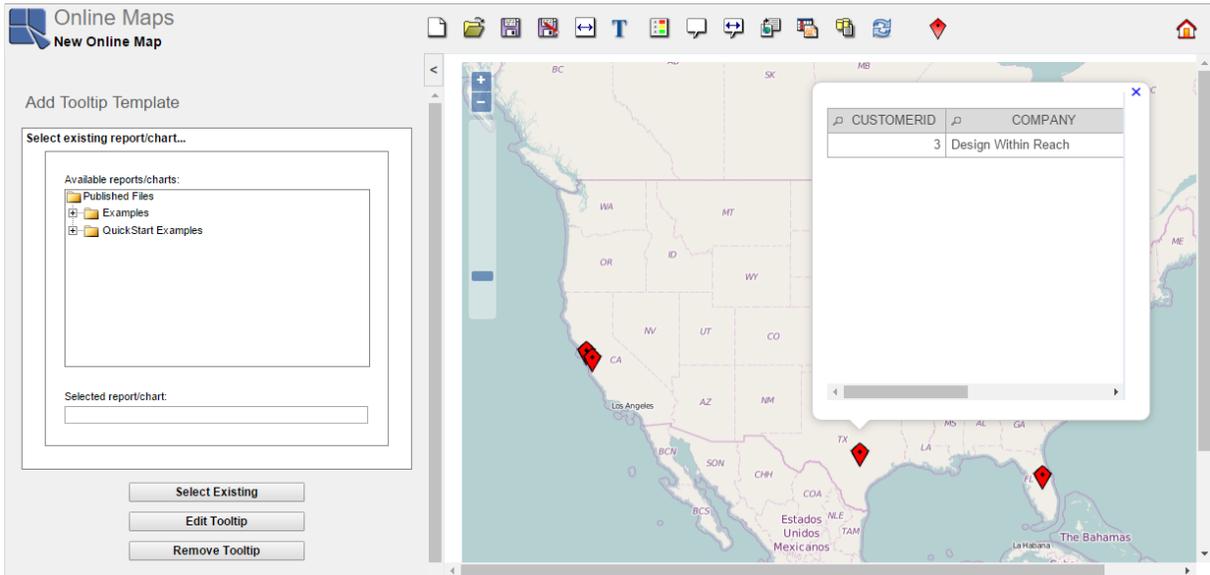
*Tooltip Report*

Click on the  *Save* icon on the QuickDesigner Reports toolbar. Enter **CustomersTooltip** as the *Name* and select the *QuickStart Examples Organizer* folder. Click *OK* to save the report.



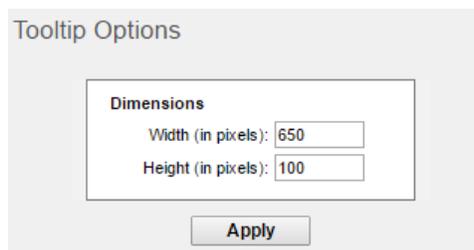
Save Report Dialog

Close QuickDesigner Reports by clicking the  Return to Map Designer icon on the toolbar.



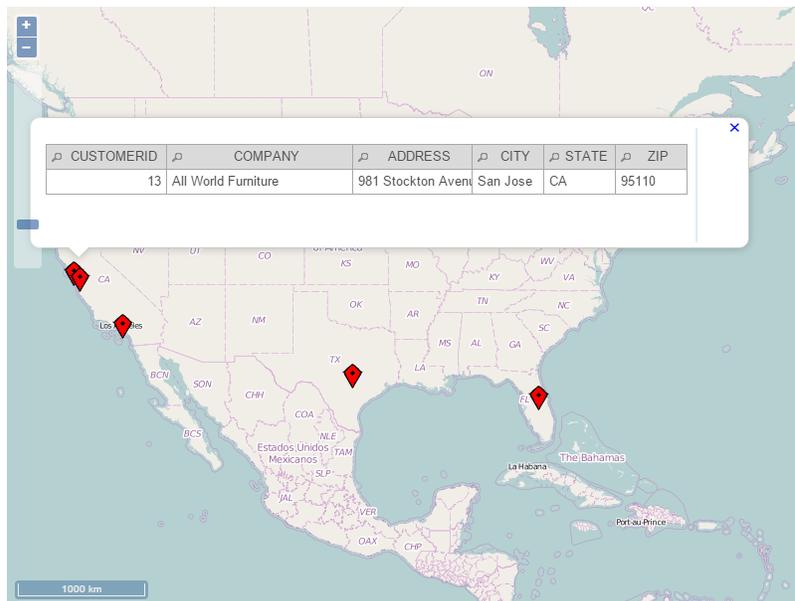
Adding Tooltip

Move the mouse arrow over some map marker and a tooltip bubble will appear. As you can see, the bubble has inconvenient dimensions for your report. To adjust the bubble dimensions, click on the  Tooltip Options icon on the toolbar. Set width to **650** and height to **100** and click **Apply**.



Tooltip Option Dialog

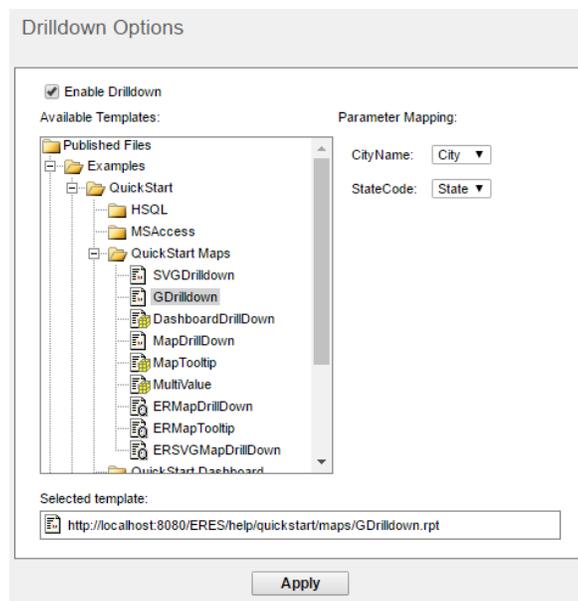
Close the currently open tooltip bubble (if any) and move mouse arrow over some marker. The tooltip should look better now.



*Online Map with Tooltip*

## Q.8.2.2. Setting Drilldowns

Click on the  *Drilldown Options* icon on the toolbar. Select the *Enable Drilldown* option. The *Available Templates* treeview will open, showing all projects, folders and parameterized reports, charts and maps inserted in the Organizer. Open the *Examples* project, then open the *QuickStart* and *QuickStart Maps* folder. Select the *GDrilldown* report. This report has two parameters: *CityName* and *StateCode*. You have to select the *Online Map* database fields containing corresponding data. In this case, map the *CityName* parameter to the *City* field and the *StateCode* parameter to the *State* field. Click *Apply*. The drilldowns are configured now. Click on a map marker, report will open in a new window. More info in Section 5.2.6.9 - Drilldowns.



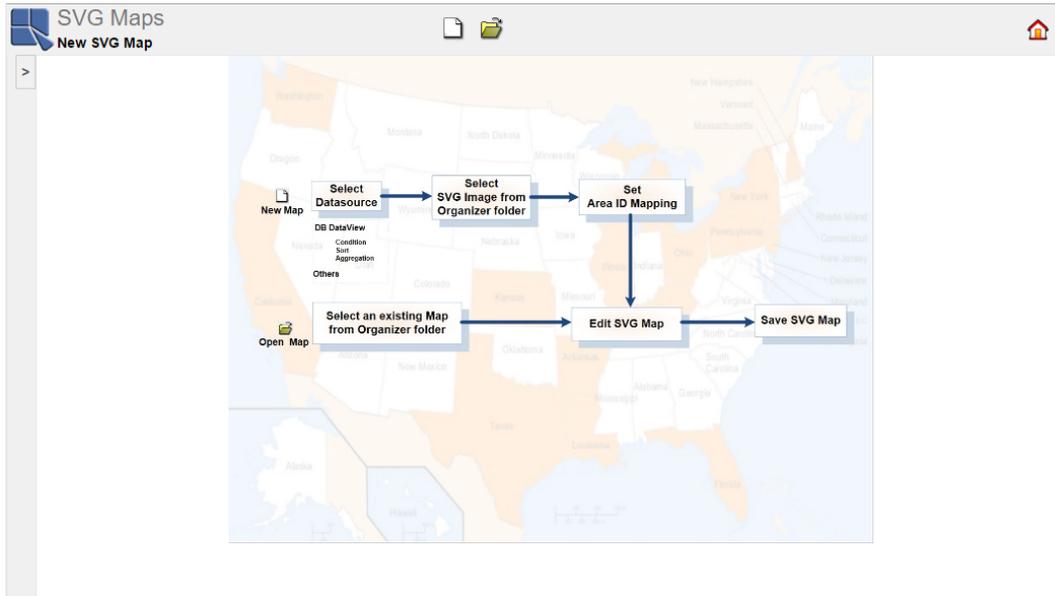
*Drilldown Options*

## Q.9. SVG Maps

Unlike Online Maps, SVG Maps do not use map points. They use **map areas**. The map areas can be colored according to some values from the map data source. Coordinates of the map areas are defined in **SVG map image**. For more information see Section 5.3 - SVG Maps.

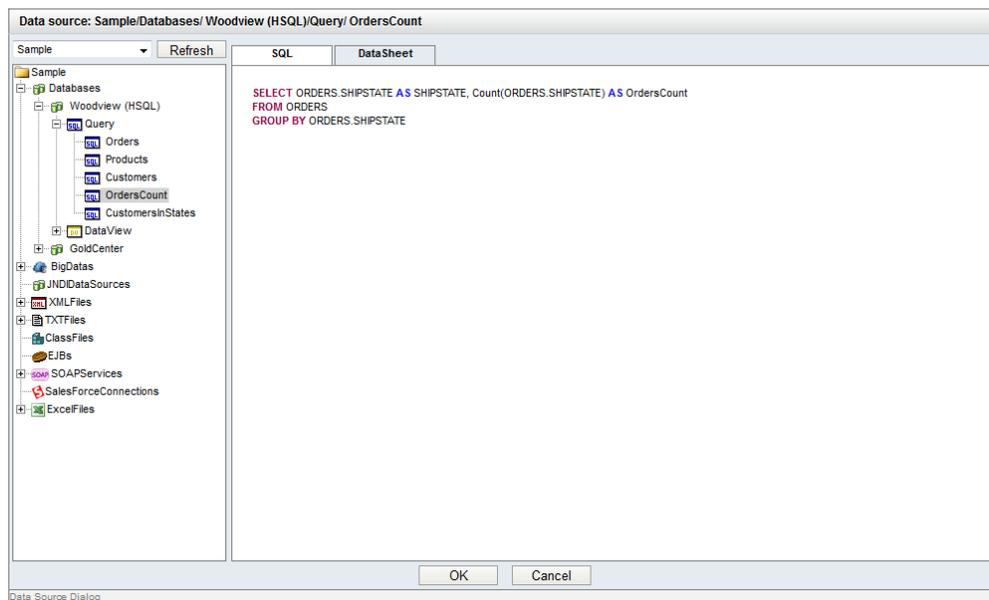
### Q.9.1. Create a Map

To create a SVG Map, go to the ERES Start page and click the link labeled *SVG Maps*. You can create a new SVG map or open some existing SVG map.



*New/Open Selection*

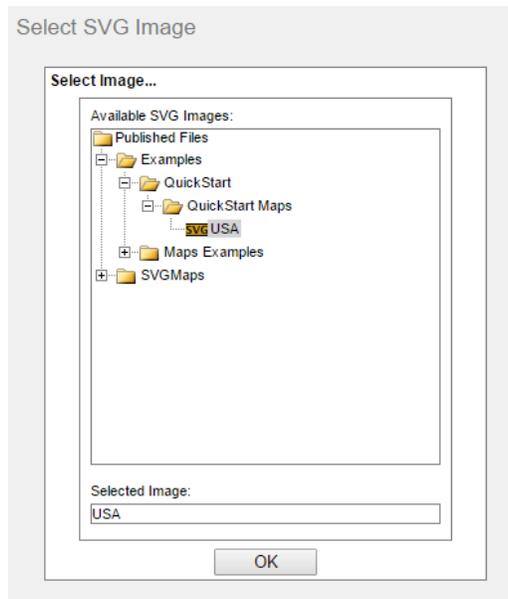
Click the  *Create New Map* button on the toolbar. You will be taken to the *Data Source Dialog*, where you can select a data registry and a data source for a new SVG Map. Select the *Sample* data registry (the left top corner of the window). Open the *Databases/Woodview (HSQL)/Query* node, and select the *OrdersCount* query.



*Data Source Dialog*

You can see records of the data source file in the *DataSheet* tab. Then click *OK* to close the *Data Source Dialog*.

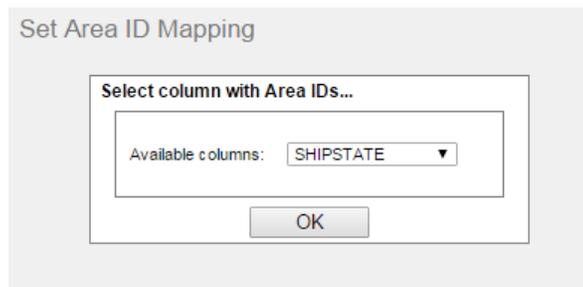
The next dialog allows you to select a Map Image. Map Images are SVG image files with data structures containing geographic data (see Section 5.3.3 - Area ID Mapping for more information about required image format), and must be inserted into the Organizer before they can be used.



*Select SVG Image Dialog*

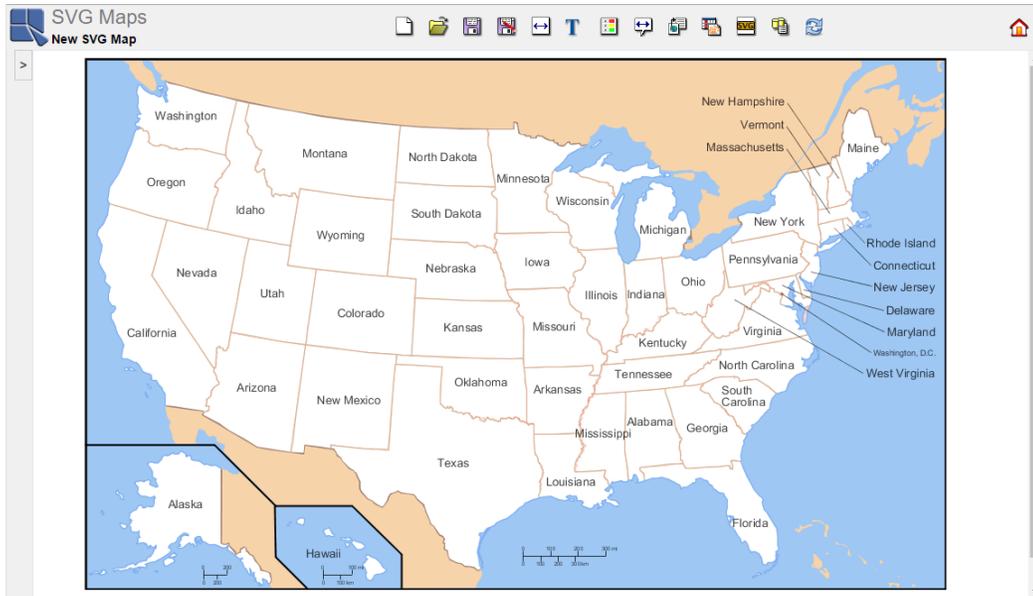
The *Select SVG Image* dialog contains a tree showing all Projects, Folders and SVG images inserted in the Organizer. Open the *Examples* project, then the *QuickStart* and the *QuickStart Maps* folder. You should be able to see the SVG image named *USA* in the *QuickStart Maps* folder. Select it and click *OK*.

Now, you can set the connection between the data source and the SVG image. This is done by assigning a data source field to the SVG map Area IDs (see Section 5.3.3 - Area ID Mapping to learn more about area ID mapping). In this case, we have a map of the USA with state name abbreviations in its Area IDs. So we want to assign a field containing the same type of data, which is the *SHIPSTATE* field. Set the *Available columns* drop-down menu to *SHIPSTATE* and click *OK*.



*Select Column with Area IDs*

SVG map will open (*Set Area ID Mapping* dialog is collapsed on the screenshot).

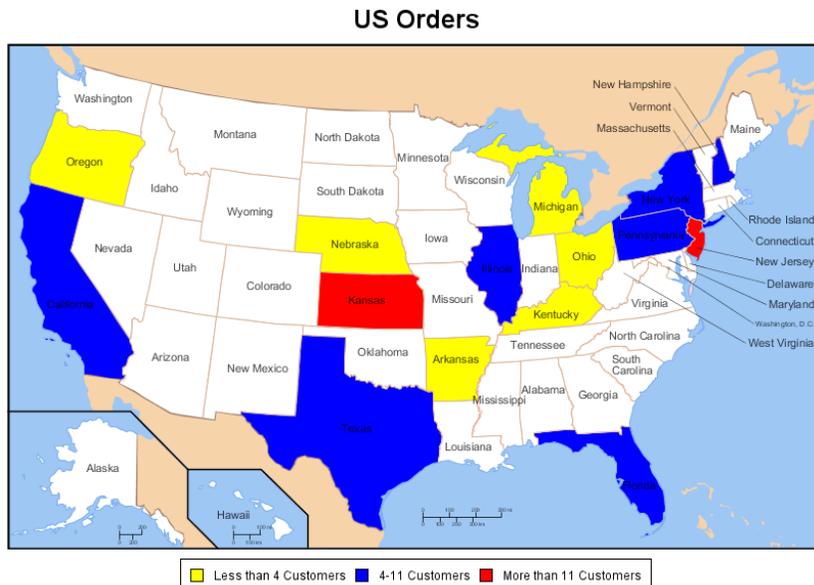


SVG Map Builder

As you can see, right now we have only the basic map with no data highlighted. There are two ways of adding data to the SVG Map: Thresholds and Drilldowns.

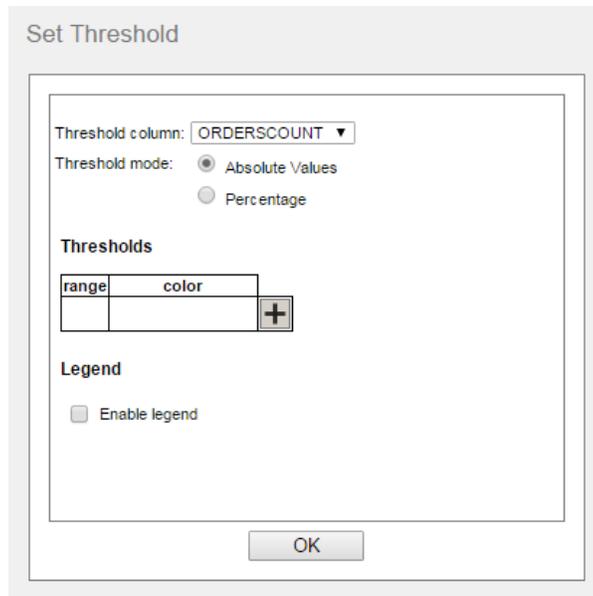
### Q.9.2. Set SVG Map Thresholds

Thresholds allow you to configure rules for coloring map areas according to its value in a particular data source field (see Section 5.3.4.5 - Thresholds).



Example: SVG Map with Thresholds

To set thresholds, click on the  *Set Thresholds* icon on the toolbar. First of all, you have to select a data source column containing numerical data that you want to report. We will use a column labeled as **ORDERSCOUNT**, which holds information about number of orders made in a particular state.

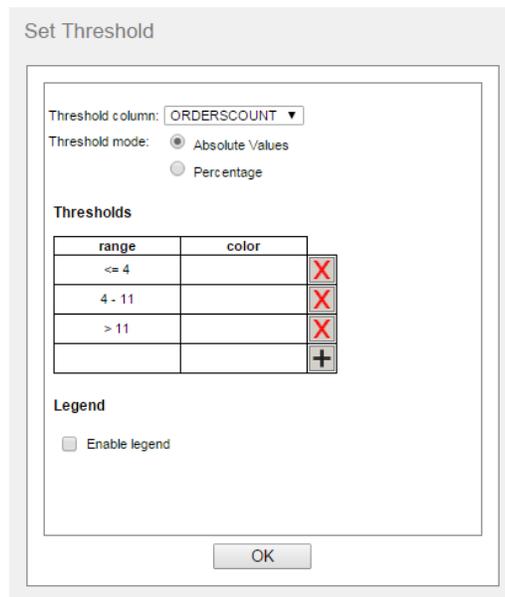


Set Threshold Dialog

Next, you can add some threshold value(s) in actual values, or percentage unit. We will use actual values for our

example. Click on the **+** Plus icon next to the blank field labeled as *color*. You will be prompted to enter threshold value, which can be done by two ways – you can type it manually into the text field or you can select the value from the drop-down menu. (There are three ways to enter threshold values for percentage units. The third way is to enter a number of intervals.) Select the value **4** from the drop-down menu and confirm the dialog by clicking

*OK*. Click on the **+** Plus icon once again, and choose the value **11** from the drop-down menu and click *OK*. ERES has automatically set the ranges between values you added.



Setting Range

You can set color for each of the intervals. Click on the white color field next to the  $\leq 4$  range. Click on some yellow field in the swatches color table and click *OK*. Color of the  $\leq 4$  range has changed to yellow. Click on the color field next to the 4-11 range, choose a blue color and click *OK*. Click on the color field next to the  $> 11$  range, choose a red color and click *OK*. The Set Thresholds dialog should look like the one on the following screenshot:

Set Threshold

Threshold column: ORDERSCOUNT ▼

Threshold mode:  Absolute Values  
 Percentage

Thresholds

range	color	
<= 4	Yellow	X
4 - 11	Blue	X
> 11	Red	X
		+

Legend

Enable legend

OK

Setting Color

Confirm the dialog by the *OK* button. Areas of some states will be filled with the colors you previously set. Without a legend, it can be unclear what the colors mean. So let's create one. Select the *Enable legend* option in the Set Threshold dialog. The *Thresholds* table has changed. It has one more column now labeled as *legend*. You should be able to write into the legend field. Fill in the column according to the following screenshot.

Set Threshold

Threshold column: ORDERSCOUNT ▼

Threshold mode:  Absolute Values  
 Percentage

Thresholds

range	color	legend	
<= 4	Yellow	Less than 4 Customers	X
4 - 11	Blue	4-11 Customers	X
> 11	Red	More than 11 Customers	X
			+

Legend

Enable legend

Legend layout: Horizontal ▼

Legend position: Bottom ▼

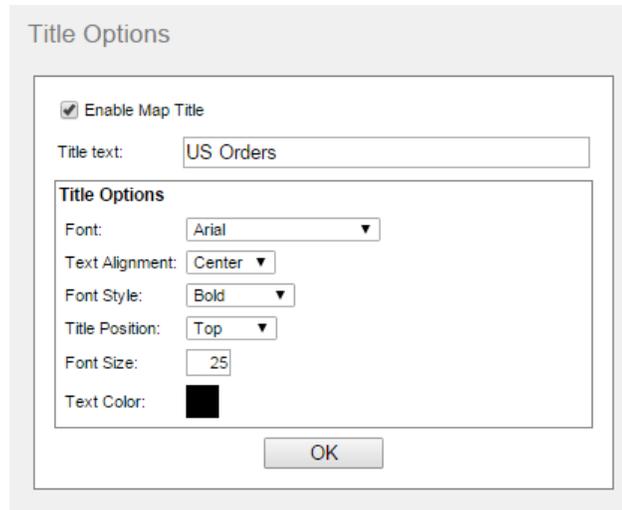
OK

Set Thresholds- Legend

Set the *Legend layout* drop-down menu to **Horizontal** and set the *Legend position* to **Bottom**. Click *OK*. Then, without map title, you have the same SVG Map image shown in the beginning of this part.

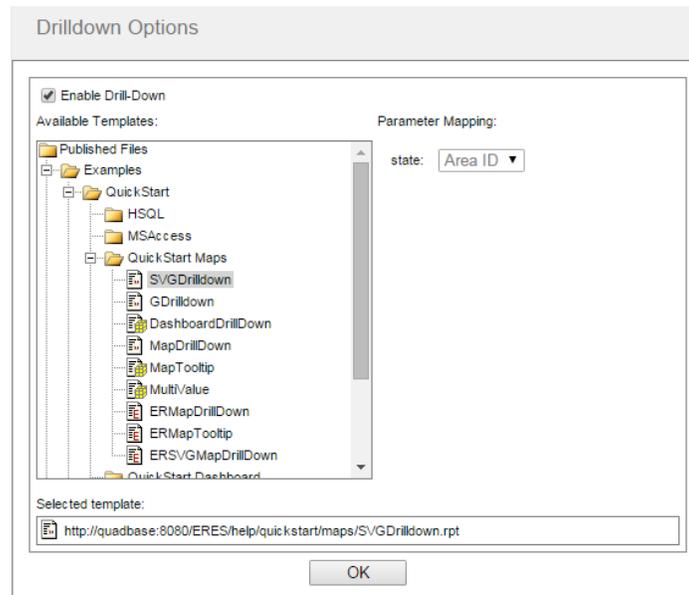
### Q.9.2.1. Add a Map Title

Click on the **T** *Map Title* icon on the toolbar. Select the *Enable Map Title* option. Type **US Orders** into the *Title text* field, set *Font Style* to **Bold** and confirm by the *OK* button.

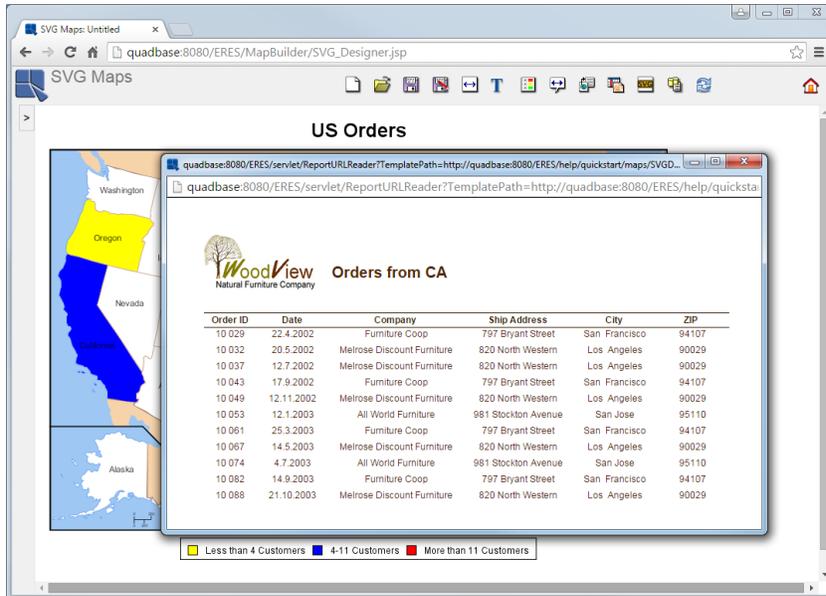
*Map Title*

### Q.9.3. Set SVG Map Drilldowns

Click on the  *DrillDown Options* icon on the toolbar. Select the *Enable Drill-Down* option. Organizer structure will show up in *Available Templates* treeview. You will be able to see only folders and parameterized reports, charts and maps inserted into Organizer. Open the `Examples/QuickStart/QuickStart Maps` folder and select the `SVGDrilldown` report. Click *OK*.

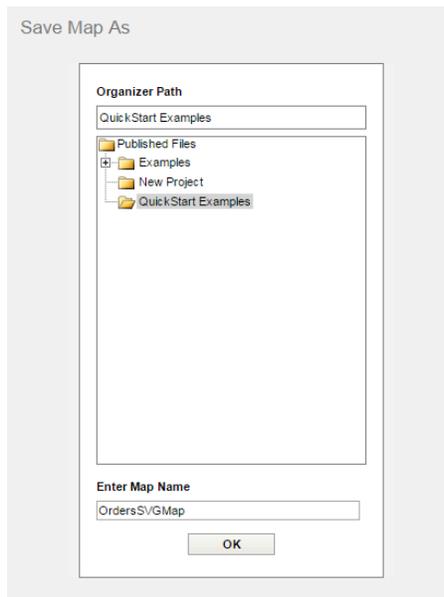
*DrillDown Options*

When you click on a colored state area, a drilldown report should open in a new window.



Map with Drilldown

To save the SVG Map, click on the  Save icon on the toolbar. Type **OrdersSVGMap** into the *Enter Map Name* text field and select the **QuickStart Examples** folder from the Organizer treeview.



Saving Map

## Q.10. Publishing

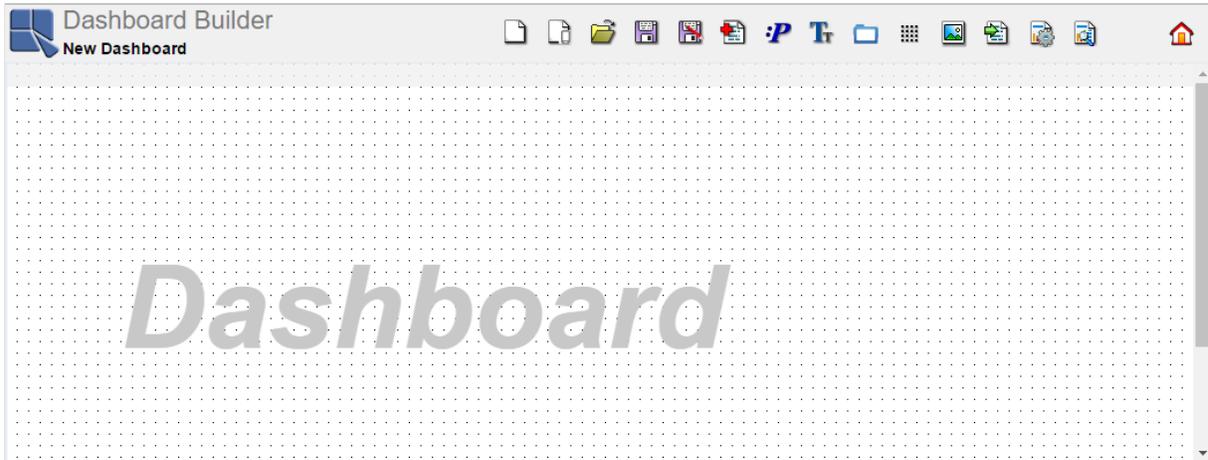
In the previous sections we looked at setting up ERES and creating reports, charts, and maps. This chapter discusses the automated publishing features in ERES.

### Q.10.1. Dashboards

In addition to URLs and the menu page, ERES also allows you to publish reports and charts using a dashboard interface. Dashboards can place multiple charts and report tables into a single presentation page. Users can define common filters for the dashboard items and set up drill-down for individual dashboard items.

### Q.10.1.1. Create a Dashboard

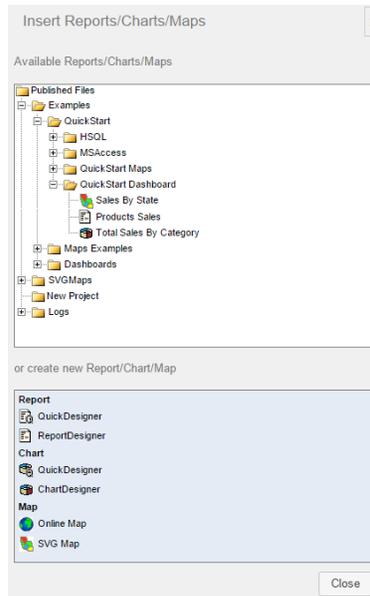
Dashboards are created in the thin-client Dashboard Builder interface. To launch the Dashboard Builder, click the *Dashboard Builder* link in the ERES start page. The interface will allow you to build a new dashboard.



*Dashboard Builder Interface*

In this section we will guide you through creating a dashboard from report, chart, and map templates that come with ERES installation. At first we need to add some templates to the dashboard. To add reports, charts, or maps to the

dashboard, click the *Add Report/Chart/Map* icon  on the Dashboard Builder toolbar. After you have clicked the button, the *Insert Reports/Charts/Maps* dialog will appear. The dialog contains a tree that mirrors the folder structure in the Organizer. Next, expand the *Examples* node and then the *QuickStart* and *QuickStart Dashboard* sub-nodes as shown on the image below.

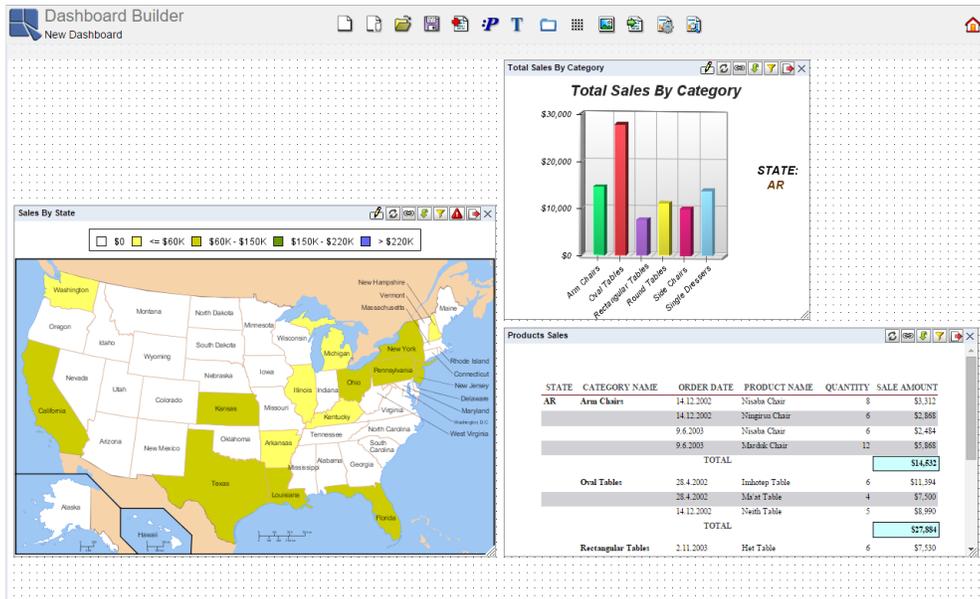


*Insert Reports/Charts/Maps Dialog*

From the dialog click on all the templates (*Sales By State* map, *Products Sales* report, and *Total Sales By Category* chart) in the *QuickStart Dashboard* node. This will insert the templates into the dashboard. Once you have inserted the templates, close the dialog by clicking *Close* button.

Now move the *Sales By State* map template by clicking to the map header and drag it to the lower left position in the dashboard. Then select the *Total Sales By Category* chart and drag it to the upper right position in the dashboard. Finally, move *Products Sales* report and drag it below the chart. You can also resize the chart/report/map by clicking

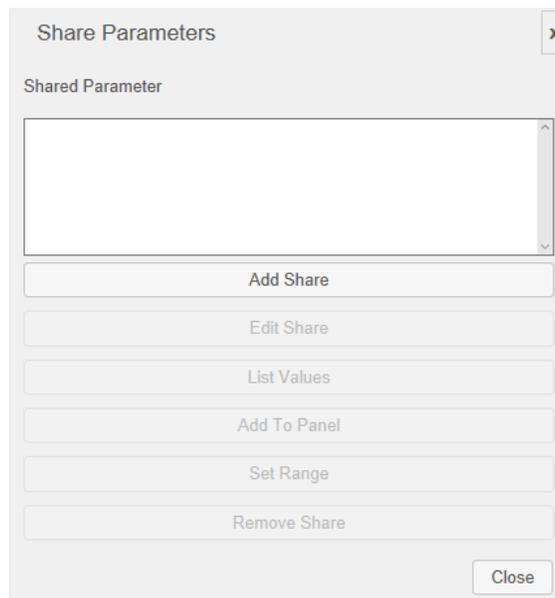
and dragging the sizing handle that appears in the lower right corner of the templates. After completing these steps, the dashboard should look like the following image:



Dashboard with Added Templates

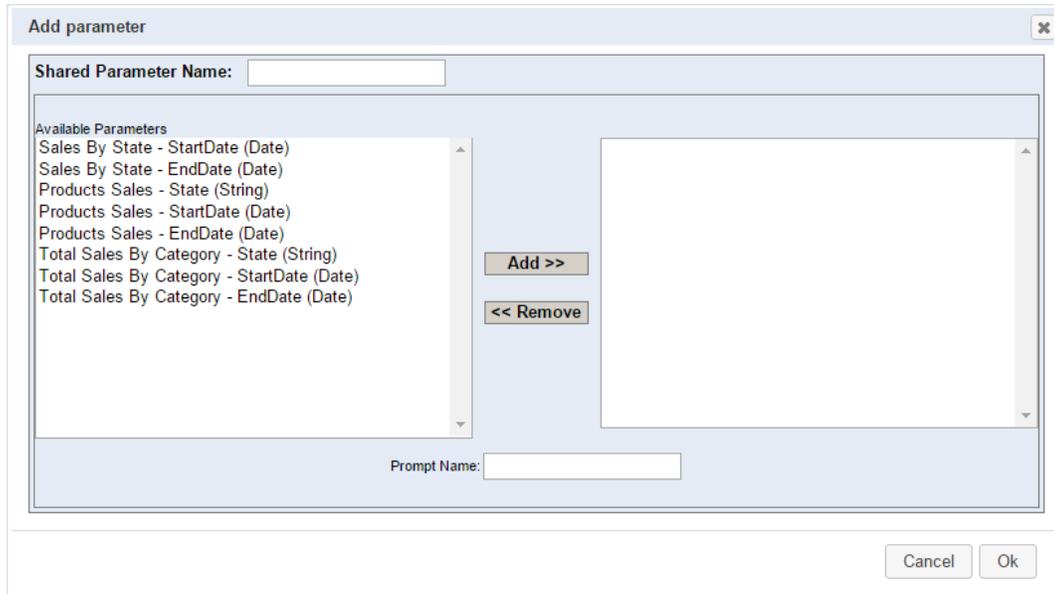
Next, we will add dashboard shared parameters that allow us to group common parameters from the chart, report, and map into a single parameter. First open the Shared Parameters dialog by clicking on the *Shared parameters*

icon  on the toolbar. This will open the Share Parameters dialog that allows you to add shared parameters to the dashboard.



Share Parameters Dialog

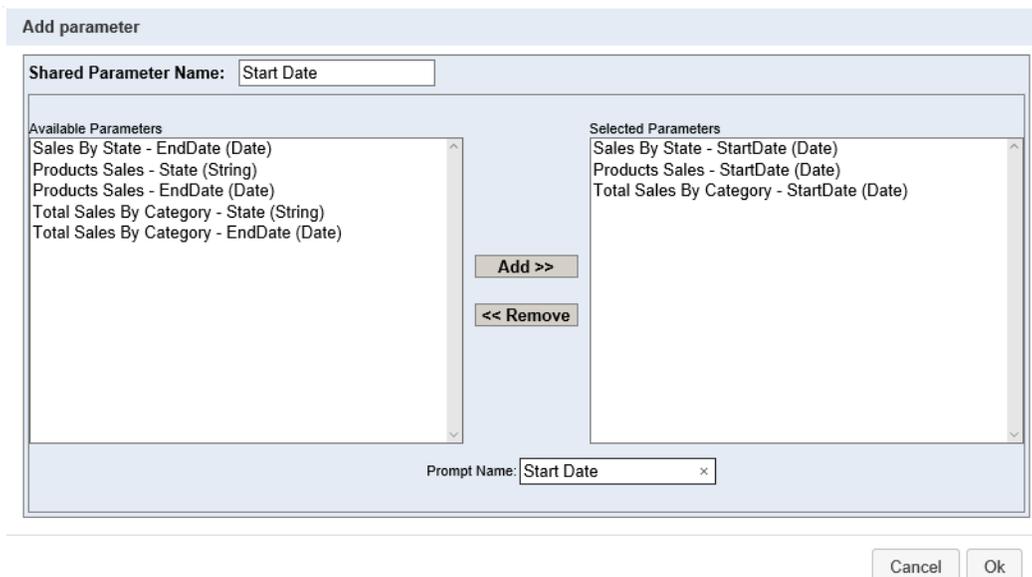
From the dialog, click *Add Share* button. This will open another dialog that allows you to specify a shared parameter.



*Shared Parameter Dialog*

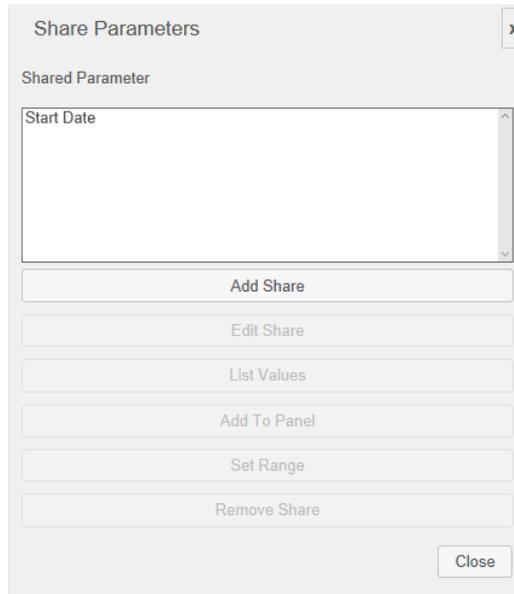
The left-hand side of the dialog shows all the available parameters from all the charts/reports/maps in the dashboard. You can select which parameters you would like to add to the selected parameters list by selecting them in the left-hand side and clicking the *Add* button. Note that all parameters in the selected parameters list must have the same data type.

In this example, we will create two shared parameters (*Start Date* and *End Date*) and put them into a date range panel. We will start with the first parameter (*Start Date*). First, enter the parameter name **Start Date** to the *Shared Parameter Name* text box at the dialog top. Enter the same text to the *Prompt Name* text field and then select all the *Start Date* parameters from the list on the left (you can select the parameters by **CTRL+Clicking** on the appropriate items in the list). After you have selected all the *Start Date* parameters from the list, click the *Add* button. This will add the selected parameters to the list on the right. At this moment the shared parameter dialog should look as follows:



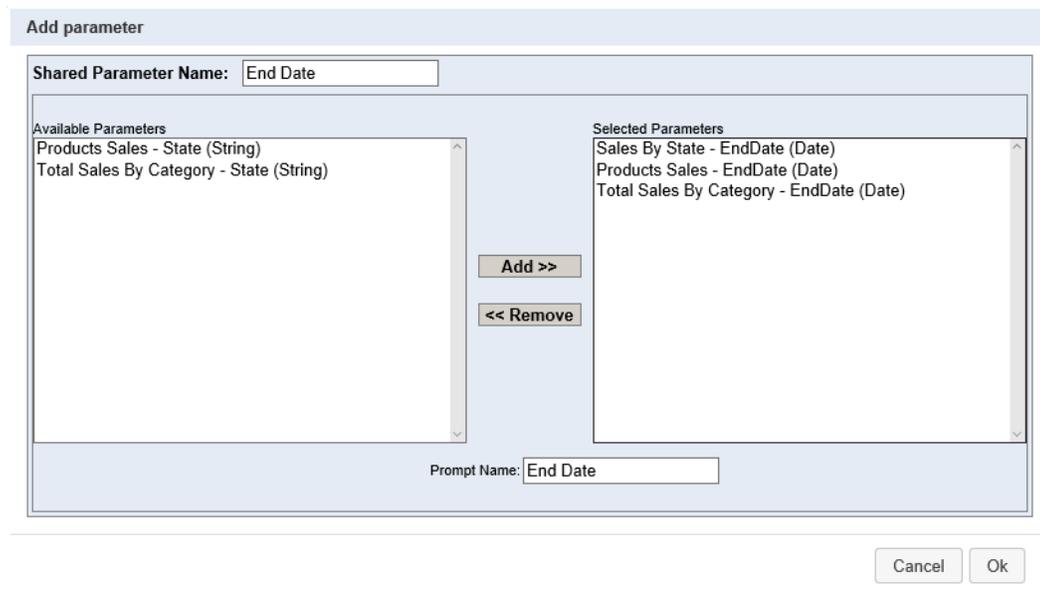
*Shared Parameter Dialog - Start Date Parameter*

Now click the *OK* button. This will put you back to the shared parameters dialog. As you can see the new shared parameter *Start Date* has been added to the shared parameters list.



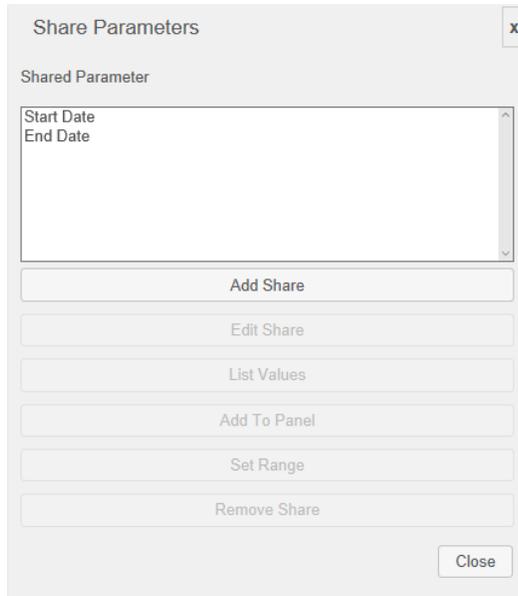
*Share Parameters Dialog - Start Date parameter added*

Next, we will add the second shared parameter `End Date` in the same way as mentioned above. From the dialog click the *Add Share* button again, which will open the shared parameter dialog. Enter the parameter name and the prompt name **End Date** and select all the `End Date` parameters from the list on the left. After you have selected all the parameters, click the *Add* button.



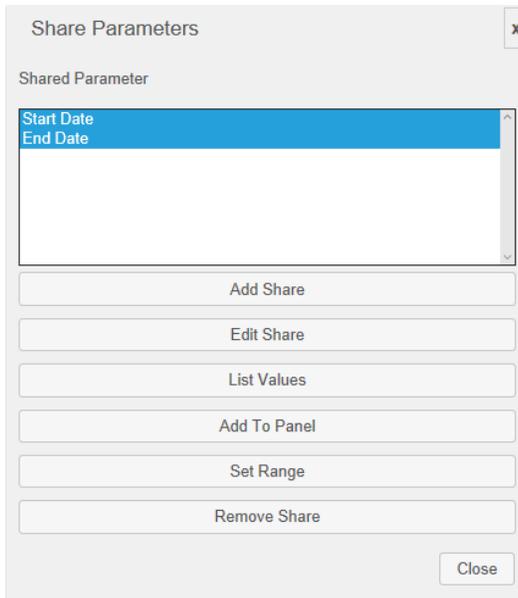
*Shared Parameter Dialog - End Date Parameter*

Click *OK* button. After adding both parameters you should see the same shared parameters dialog as follows:



*Shared Parameters Dialog - Added Parameters*

In the dialog, select both parameters by **CTRL+Clicking** on the parameters as shown in the image below.



*Shared Parameters Dialog - Selected Parameters*

After selecting the parameters, click *Set Range* button. This will open the *Range Param Attributes* dialog that allows you to specify ranges for the parameters.

*Set Date Range Dialog*

As you can see there are lot of options in the dialog. Please note that we will describe only the options necessary for the example. For more information about the rest of options, please see the Section 6.2.4.2 - Parameter Range.

First, specify the *Range Name* in the text box at the top of the dialog, e.g. **DateRange**. The name represents the title for the Date Range panel. Next, specify the *Prompt Text* to be **Date Range**. This text will be then displayed next to the selected date parameter drop-down menu in the Date Range panel. After entering the names, we will proceed with creating date ranges.

For simplicity we will add only two date ranges (*Year 2003* and *Year 2002*) in this example. The range *Year 2003* will include all the dates in year 2003 (dates from 2003-01-01 to 2003-12-31 in the Year-Month-Day format). Similarly, the range *Year 2002* will include all the dates in year 2002 (dates from 2002-01-01 to 2002-12-31).

Now specify the name for the first range in the *Option Name* text box to be **Year 2003**. Next, enter the expressions for the *Start Date* and *End Date* parameters to be **2003-01-01** and **2003-12-31**. Once you have entered the *Range Name* as well as the expressions, click *Add >>* button. This will add the date range to the list of selected ranges on the right.

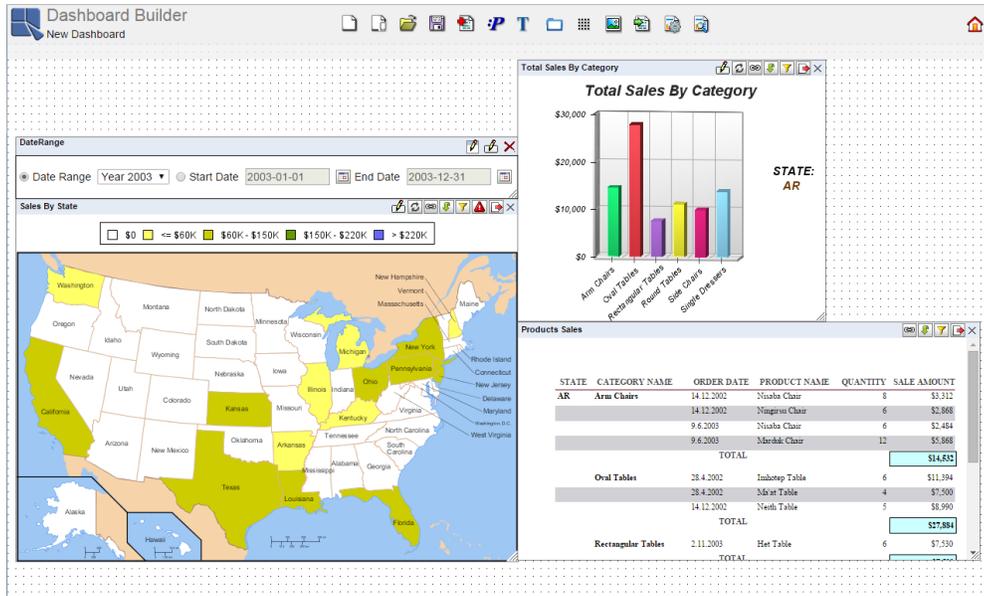
Similarly, add the second date range *Year 2002*. As mentioned above, first specify the range name to be **Year 2002** and enter the expressions for the *Start Date* and *End Date* parameters as **2002-01-01** and **2002-12-31**. After that, click the *Add >>* button to add the range to the list of selected ranges on the right.

*Set Date Range Dialog - Specified Ranges*

Once you have added both date ranges into the selected date ranges list on the right, click *OK* button. After that click somewhere to the Dashboard Builder, and the date range panel appears.

*Date Range Panel*

Now move the panel above the map as shown on the image below. To move the panel, just click on the panel header and drag it to the desired location. You can also resize the panel by clicking and dragging the lower right corner of the panel. The resizing rectangle will then appear, allowing you to adjust the size of the panel.



*Dashboard With The Date Range Panel*

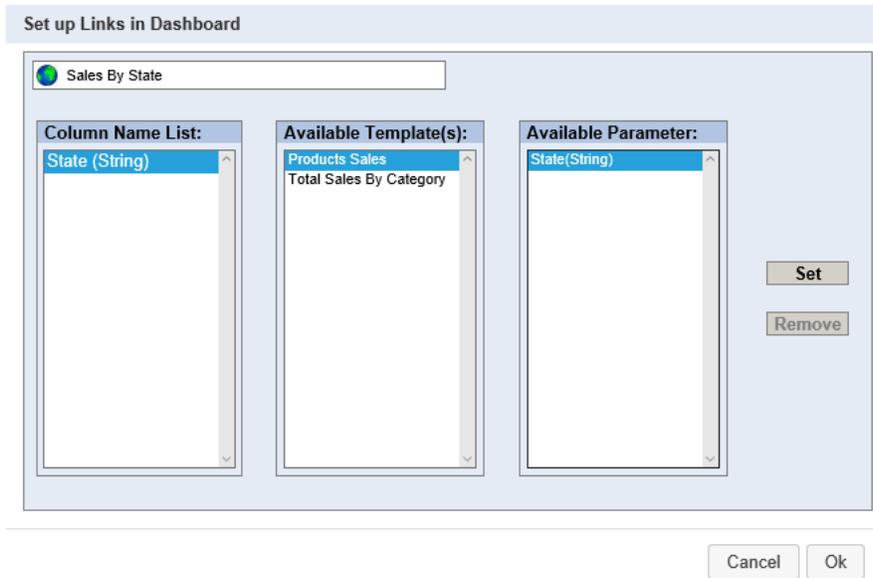
The last thing we will do in this example will be adding links from the map to the chart and report in the dashboard. This will enable sending the `State` parameter from the map data points to the chart and report.

To set up links from the map, click the *Add/Modify Link* icon  from the map header. This will open the Set Up Link dialog below. There are three lists in the dialog. The first list contains all the available data columns and their data types in the source template (in our case, `Sales By State` map). The second list shows all the available destination templates in the dashboard (in our case, `Product Sales` report and `Total Sales By Category` chart). Finally, the third column contains all the available parameters in the selected destination template. Please note that the third column will show the parameters only after selecting the destination template in the second list.

The screenshot shows the 'Set up Links in Dashboard' dialog box. It has a title bar and a main area with three lists. The first list, 'Column Name List', contains 'State (String)'. The second list, 'Available Template(s)', contains 'Products Sales' and 'Total Sales By Category'. The third list, 'Available Parameter:', is empty. To the right of the lists are 'Set' and 'Remove' buttons. At the bottom of the dialog are 'Cancel' and 'Ok' buttons.

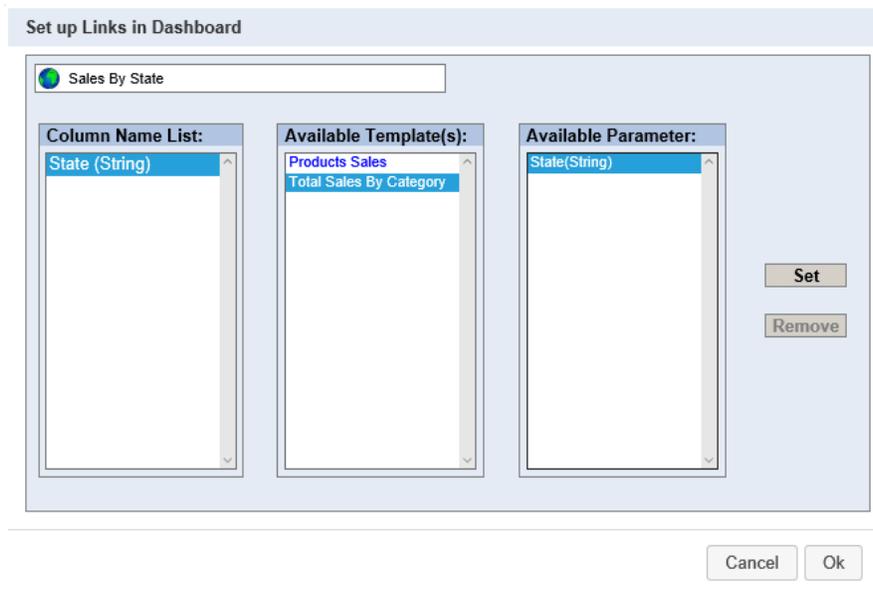
*Set Up Links Dialog*

To set up a link from the map to the `Product Sales` report, select `State (String)` item in the first list and the `Product Sales` item in the second list. This will cause showing the parameters from the report in the third list. After that select `State(String)` item from the third list. This should enable `Set` button as shown on the image below. Click the button to set up the first link.



*Set Up Links Dialog - Link From The Map To The Report*

Similarly, set up a link from the map to the Total Sales By Category chart. First, select *State (String)* item in the first list and the *Total Sales By Category* item in the second list. This will cause the parameters from the chart in the third list to be shown. From the third list select *State(String)* item. This should enable *Set* button as shown on the image below. Set up the second link by clicking the button.

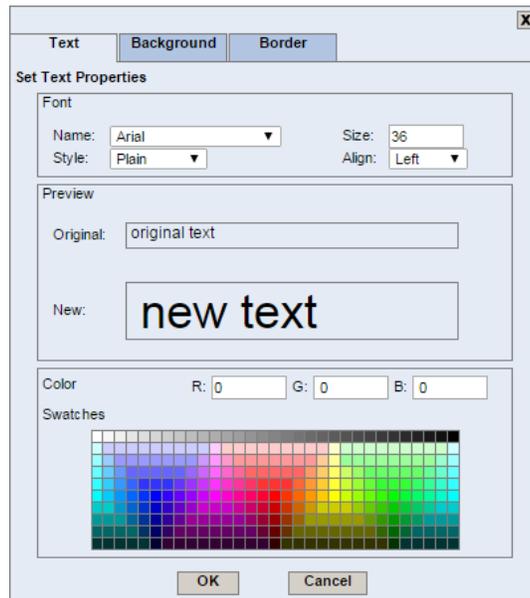


*Set Up Links Dialog - Link From The Map To The Chart*

Once you have set up both links, click *OK* button that will close the dialog. At this moment the links from the map should be set up. So if you now click on a data point in the map, the chart and report should refresh according to the data point parameter value.

To finish our dashboard, we will just add a dashboard title and choose a dashboard background color. To add the title, click the *Insert Label* icon **T** from the toolbar. A small rectangle will then follow your mouse cursor. Position the rectangle where you would like to insert the label and click. This should add the label panel into the dashboard. Next, double click to the label panel to edit the label text. Enter e.g. **SALES BY STATE DASHBOARD** text and click outside of the label panel. You may also want to resize the label because the label is now too small to be the

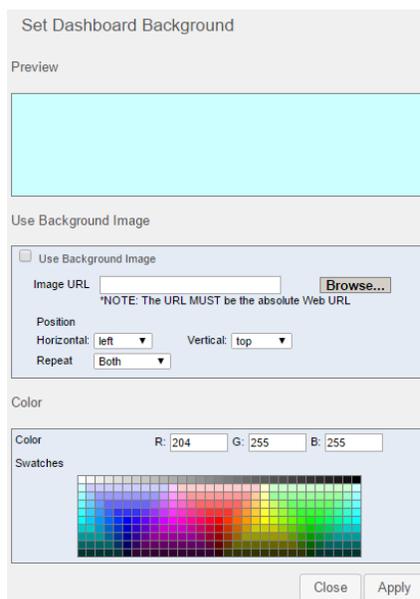
dashboard title. This can be done from the label properties dialog that opens after clicking on the *Edit* icon  on the panel header.



*Set Label Properties Dialog*

You can specify various properties for the label from this dialog. In the dashboard preview screen shot below, we just increased the label font size to **36px** in the *Text* tab, disabled the panel border in the *Border* tab (write **0** to the *Thickness* option), and set up the label background color to be transparent in the *Background* tab. Click *OK* to apply changes. Now, click and drag the bottom right corner of the label border to be the whole title in one line. For more information about the label properties, please see Section 6.2.5 - Insert Labels.

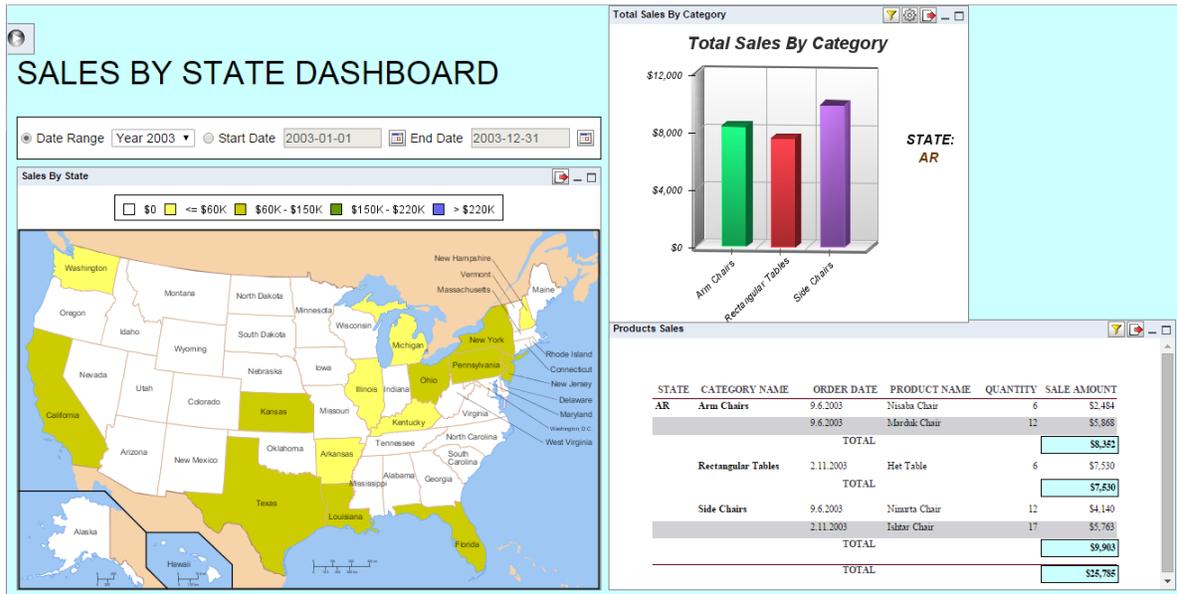
To change the dashboard background color, just click the *Add Dashboard Background* icon  on the toolbar. This will open the *Set Dashboard Background* dialog from which you can choose the dashboard background color. Select the color and click *Apply*.



*Set Dashboard Background Dialog*

Once you have chosen the dashboard background color, you may preview the dashboard by clicking the *Preview*

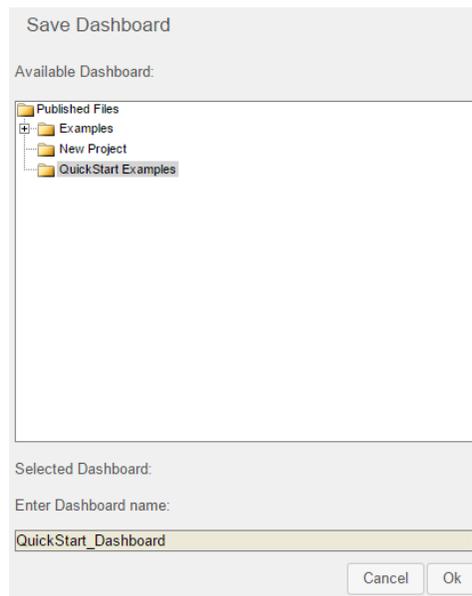
button  on the Dashboard Builder toolbar. The dashboard display will open in a new window. For more information about dashboard features, see Section 6.1 - Introduction to Dashboards.



*Dashboard Preview*

### Q.10.1.2. Save the Dashboard

Close the dashboard preview window to return to the main Dashboard Builder interface. Click the *Save* button  on the toolbar. A dialog will open prompting you to specify a name for the dashboard.



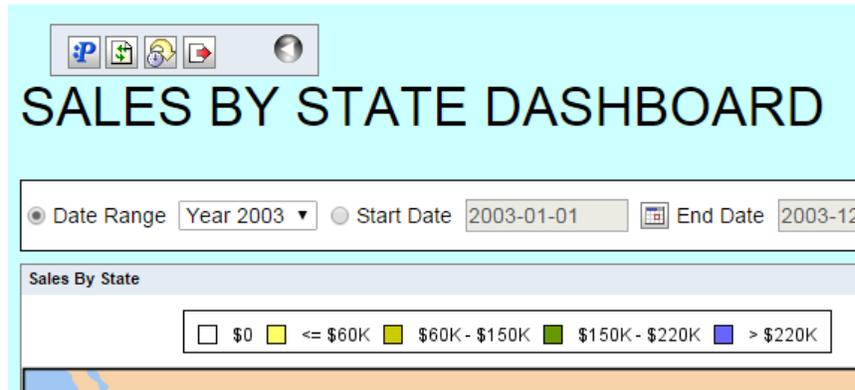
The "Save Dashboard" dialog box has a title bar and a main area. It contains a section for "Available Dashboard:" with a treeview showing a folder structure: "Published Files" (containing "Examples" and "New Project") and "QuickStart Examples". Below this is a "Selected Dashboard:" label, followed by an "Enter Dashboard name:" label and a text input field containing "QuickStart\_Dashboard". At the bottom right are "Cancel" and "Ok" buttons.

*Save Dashboard Dialog*

Enter a name for your dashboard and then from the treeview pane select the project that you created in Section Q.2.2.2 - Add a Project. Click *Ok* to save the dashboard. The window will then give you a message that the dashboard was saved successfully. Click *Ok* to close the dialog.

### Q.10.1.3. Export Dashboard to PDF

Click the *Options* button  on the Dashboard Builder toolbar and check the option *Show toolbar in preview* (in the *Other* section). Click *Apply* to apply the setting. Open the dashboard preview window by clicking the *Preview* button . Then click the *Unpack* button  to open the preview toolbar. You can export dashboard to PDF file by clicking *Export Dashboard* button  from the preview toolbar.



*Export Dashboard from Preview*

All the reports, charts, SVG Map and other objects in the Dashboard will be exported to PDF file with the exception of Google Map due to licence restrictions.

Alternatively, you may open the same .dsb file in Published Files (for more information please see Section 7.1 - The Menu Page ), open the Dashboard toolbar and Export PDF there.

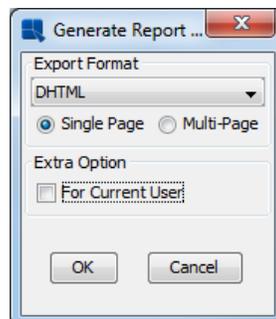
Note: There may be small differences in the appearance of elements in export (e.g. radio button shape can change from round to square).

### Q.10.2. URLs

One automated deployment provided in ERES is the ability to run reports and charts via URL calls to the ERES server. Report and chart URLs are documented in detail in Section 7.2 - Image URLs, and Section 7.3 - Report URLs.

To generate a URL, first log into the Organizer. With the Organizer interface open, select the project you created (in Section Q.2.2.2 - Add a Project) in the left-hand side. You will see a list of files added from the previous exercises. Select the `QuickStart532.rpt` file in the organizer (drill-down example that can be created in Section Q.4.3.4

- Drill-Down), and click the *Generate URL* button on the toolbar . A dialog will open prompting you to specify options for the generated URL.



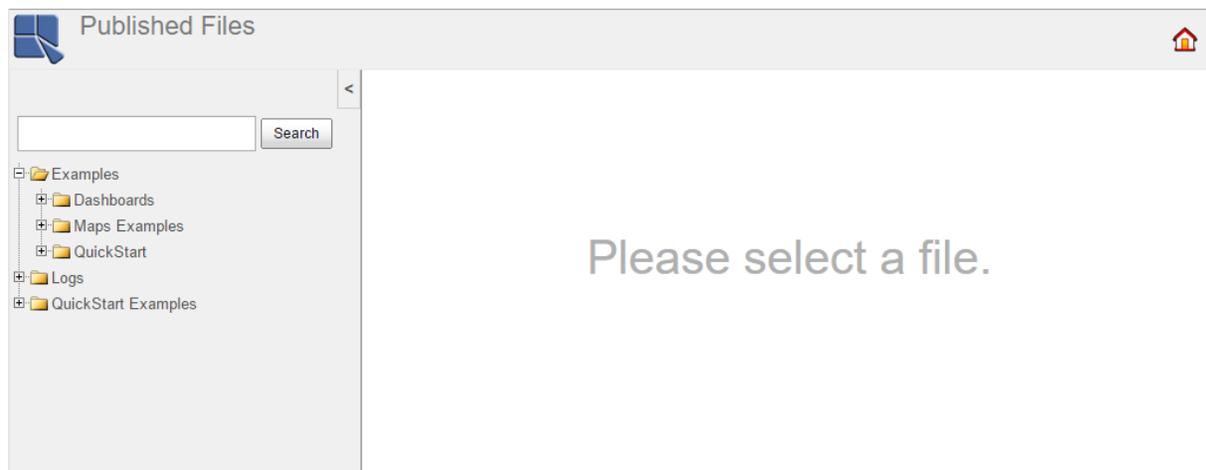
*URL Options Dialog*



## Q.10.3. The Menu Page

Another way that ERES automatically publishes reports, charts, maps, and dashboards is through the Menu Page. The Menu Page is a thin-client interface that allows users to run/view the reports, charts, maps, and dashboards to which they have access in the Organizer.

To launch the Menu Page, go to the ERES Start page and click the link labeled *Published Files*. The Menu Page will load.



*Menu Page*

The page contains a list of all the reports, charts, maps, and dashboards in your project and some examples. The only option available is to run the file because you do not have any active schedule or archive jobs. To open a file, expand respective project/folder nodes (in the left-hand tree-list) to locate the file and click on the file name. The file will load in the right-hand DHTML Viewer panel.

For more information about the options and functions available in the Menu Page, see Section 7.1 - The Menu Page. After you have finished viewing your report/chart/map/dashboard, close the window that contains it, and click the *Home* button in the menu interface to return to the start page.

## Q.11. Alerts

In previous sections, you learned how to create reports, charts, and maps (let's call them “objects”), and how to publish them in dashboards. This section describes how to add and manage alerts in objects, how to “watch” alerts in dashboards and how to monitor alerts automatically.

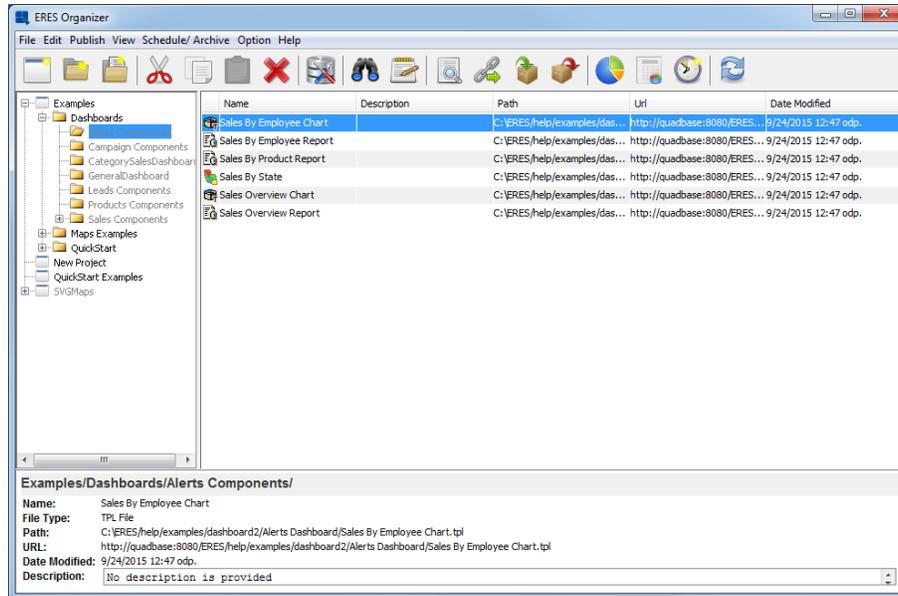
An alert is a range of some indicator (for example, profits/losses, sales, up-time, etc) you want or do not want the indicator to reach. If the indicator's value reaches into the alert range (in other words: if an alert was triggered), you will be instantly informed about that, so you can fix issues in a timely manner. More detailed information about alerts can be found in Section 11.1 - What is an Alert.

First of all, you have to create the alert. That means that you have to select which indicator you want to watch and what values are critical for you. Each object (report, chart, or map) can have its own alerts.

### Q.11.1. Alerts in Charts

In Section Q.5.3.2 - Control Areas, you learned how to add a “control area” into a chart. That is all you have to know to add alerts into charts – just open a chart in Chart Designer and add one or more control areas into it. Basically, you can think of a control area as an alert in a chart.

Launch ERES Organizer (as described in Section 2.1.1 - Starting the Organizer), expand *Examples/Dashboards/Alerts Components* and double-click on the *Sales By Employee Chart*.



*Sales By Employee Chart Location*

The *Enter Dialog* will show up. You do not have to change any parameter values, so just click *OK* to continue. You should be able to see a chart now. You may notice that at least one control area has been already added into the chart. The existing control area represents minimal sales limit – no employee should sell less than the limit. We will add another alert that will indicate that employee's personal sales are not optimal, although they are not critically low.

Open the *Insert* menu and choose the *Control Area* option. *Control Area Options* dialog has appeared.



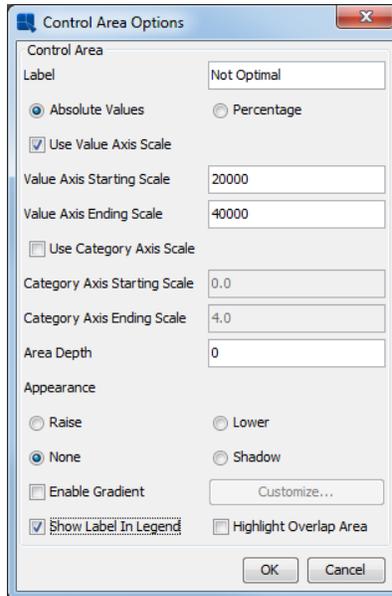
### Note

You may have noticed that the *Control Area Options* dialog contains two control areas, but you can see only one control area in the chart. That is because one control area (No sales, to be specific) is so small that it cannot be seen on current axis scale. This does not affect the alert in any way, it will still work as if the control area was visible.

To add a new control area, click the *Insert* button. Now, it is time to set up our new alert. The first option is called *Label*. The *Label* is set to **New Range** by default. Delete the default label and set it to **Not Optimal** instead.

Now we will set up the alert range. Set the *Value Axis Starting Scale* to **20000** and the *Value Axis Ending Scale* to **40000**.

To include the control area in the legend, choose the *Show Label In Legend* option.

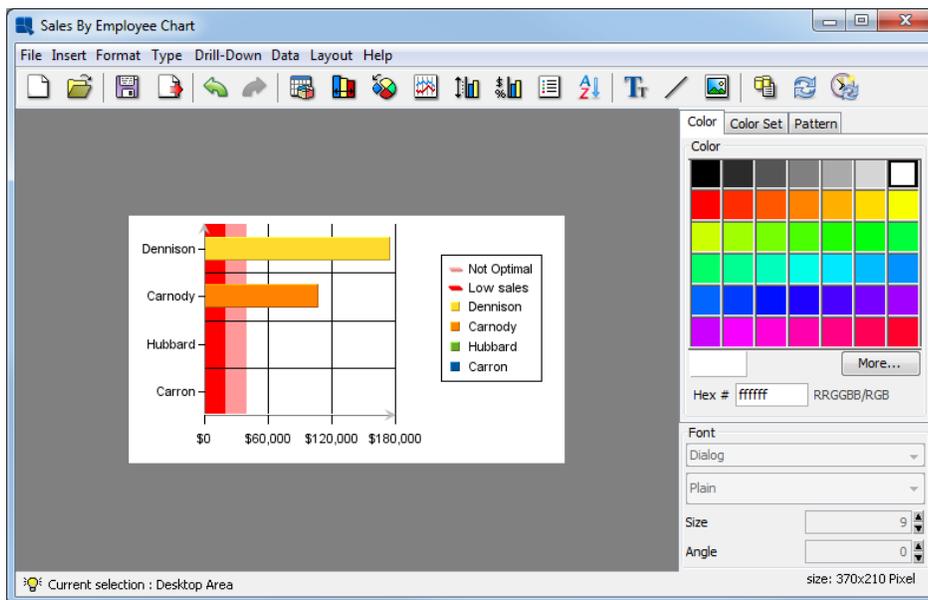


Control Area Options

**Note**

Complete description of this dialog can be found in Section 4.2.4.6.3 - Adding Control Areas

Click *OK* to close the dialog. As you can see, our new alert was created successfully. Now click the *OK* button again to go back to the Chart Designer. The new control area has appeared in the chart. It is highlighted by a random color. To change the color, click on the new area and then choose an indicative color (for example, light red) from the *Color* panel on the right side of the window. Save the chart and close the Chart Designer.



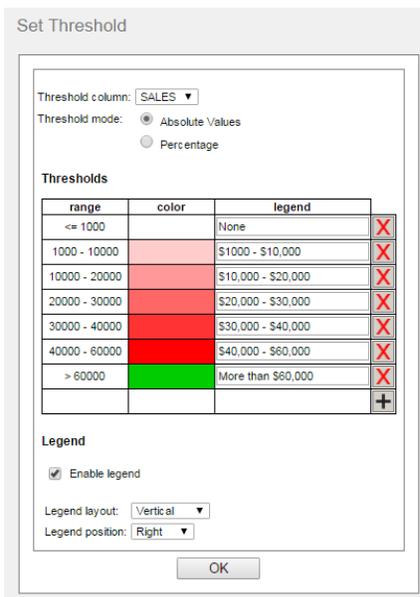
Sales By Employee Chart

**Q.11.2. Alerts in Maps**

In the Section Q.9 - SVG Maps, you created a SVG map with thresholds. This means that you have also created a map with alerts because thresholds are equivalent to alerts in maps.

Let's add a threshold into an existing SVG map. Open ERES main page and click on the *SVG Maps* link. Click on the  *Open Map* icon on the toolbar. Expand *Examples/Dashboards/Alerts* Components nodes in the *Open Map* tree-list, select the *Sales By State* map and click the *OK* button. Click *Submit* to use default parameter values. To manage thresholds, click on the  *Set Thresholds* icon. To add a new threshold (alert), click on the  *Add Threshold* icon. In the *Enter threshold value* dialog, type **60000** and click *OK*. As new thresholds are added, some legend labels are not right. The legend labels are important for alerts because they also double as alert names. Set the *40000 – 60000* legend to **\$40,000 – \$60,000**. Now set the *> 60000* legend label to **More than \$60,000**. Click on the *> 60000* color field. *Select Color* dialog should show up. Choose an appropriate color (for example, green) and click *OK*.

Now, the Set Thresholds dialog should look like this:



The dialog box is titled "Set Threshold" and contains the following elements:

- Threshold column: SALES (dropdown)
- Threshold mode:  Absolute Values,  Percentage
- Thresholds table:

range	color	legend	
<= 1000		None	X
1000 - 10000		\$1000 - \$10,000	X
10000 - 20000		\$10,000 - \$20,000	X
20000 - 30000		\$20,000 - \$30,000	X
30000 - 40000		\$30,000 - \$40,000	X
40000 - 60000		\$40,000 - \$60,000	X
> 60000		More than \$60,000	X
			+

- Legend section:
  - Enable legend
  - Legend layout: Vertical (dropdown)
  - Legend position: Right (dropdown)
- OK button

*Set Thresholds Dialog*

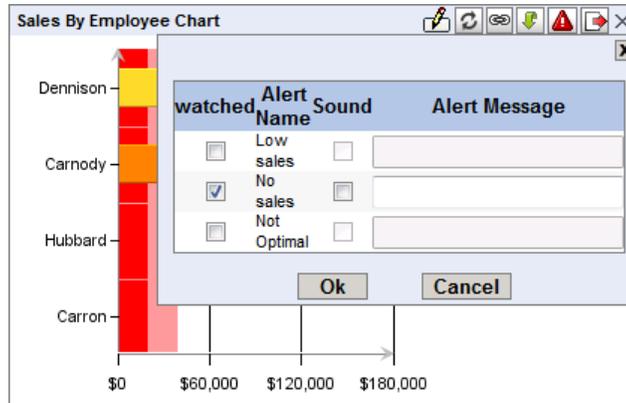
If it does, click *OK* to confirm the changes. The map has been changed since you added the threshold. Click on the  *Save* icon and click *OK* to save the changes. Click  icon in the top-right corner of the window to go back to the ERES main page.

### Q.11.3. Alerts in Dashboards

In previous paragraphs, you added alerts into few objects. Now we want to make use of those alerts. There are two ways to utilize alerts – alert watching and alert monitoring. This paragraph describes how to set up alert watching. Alert watching is set up in dashboards. The basic principle is quite easy: insert some objects (reports, charts or maps) into a dashboard and pick the alerts you want to watch.

Log on to the ERES main page and launch *Dashboard Builder*. Click  *Open* icon. Expand *Examples/Dashboards* nodes, select *Alerts Dashboard* dashboard, and click *Ok*. All dashboard objects have  *Set Alert* icons in their header bars. That means that all dashboard objects have some alerts that can be watched.

Locate the *Sales By Employee Chart* chart and click on its  *Set Alert* icon. The following dialog should appear.



*Sales By Employee - Watched Alerts*

There is the list of all alerts that you saw in Section Q.11.1 - Alerts in Charts. One alert is set to be watched.

The list also includes the *Not optimal* alert you created previously. Check the *Not optimal* checkbox and uncheck the *No sales* check-box.

Click *Ok* to close the dialog and then click on the  *Preview* icon in the main menu. Notice that the chart's border is not flashing. That means that the *Not optimal* alert was not triggered. Now locate following panel:

Year:   Quarter:

*Parameter Panel*

Change the *Quarter* parameter to **Q2**. The border started flashing, because the alert was triggered with current parameters. In other words: at least one *Sales By Employee Chart* chart bar ends in the *Not optimal* control area.

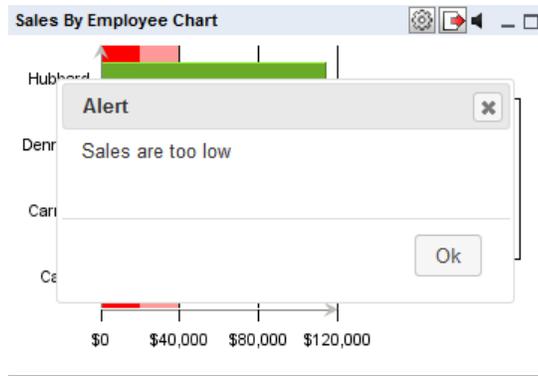
You may also notice that the *Sales Overview Report* header is flashing. That is because the report contains a watched alert which is triggered. You cannot see the report until you click on its tab header, but you can tell that an alert is triggered in the report.

Please note that alert highlighting, blinking and alert messages are visible only in *Preview* of the Dashboard Builder.

### Q.11.3.1. Sound alerts and alert messages

Border flashing is not the only way to alert you. You can also set alert messages and sound alerts. Close the *Preview* window and click again on the  *Set Alert* icon for the *Sales By Employee Chart*. For our watched *Not Optimal* alert, check *Sound* and enter any message into *Alert Message* textbox. Click on *OK* and use *Preview* button to preview the dashboard.

Trigger the alert by changing the *Quarter* to *Q2* in the *Parameter Panel* and an alert message with your entered text will appear and you will also hear a beep sound. To mute the sound, click on the  *Mute* icon in upper right corner of the chart. Please note that each template has its own *Mute* button. To close the alert message, simply click on *OK* button.



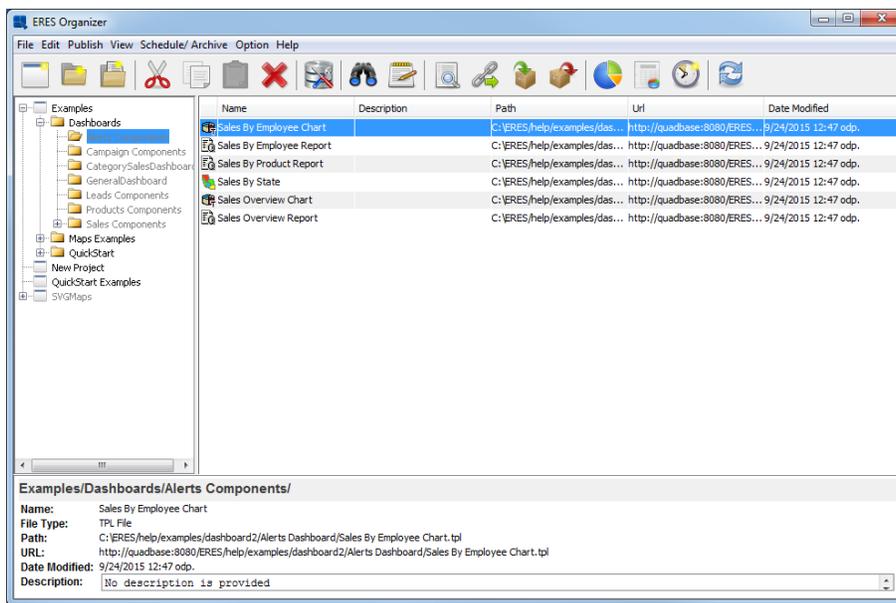
*Triggered alert message*

For more information about alerts, please refer to Section 11.3 - Dashboard alerts.

## Q.11.4. Alert Monitoring

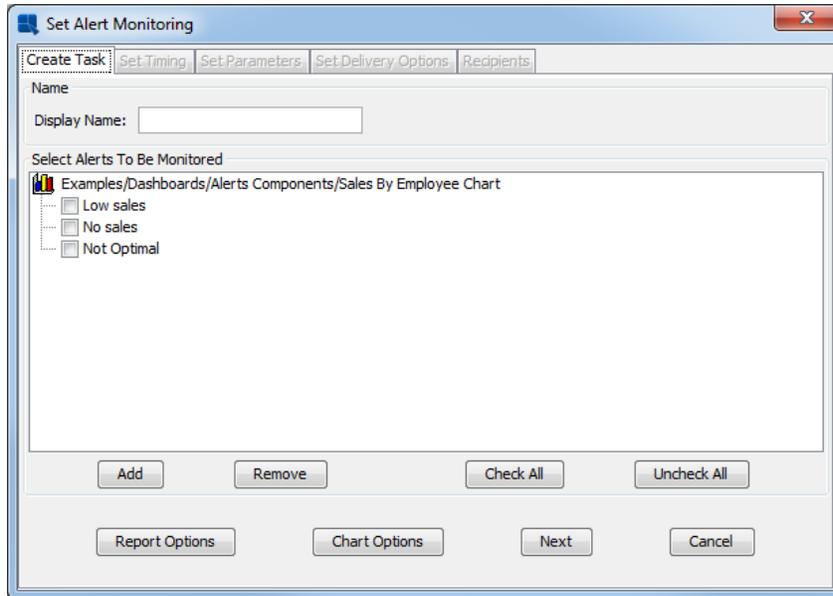
Another way of using alerts is alert monitoring. Unlike alert watching, alert monitoring checks alerts even when there is no dashboard running.

Lets set up a new alert monitoring job. Log on to the ERES main page and click the *Organizer* button. Expand Examples/Dashboards/Alerts Components nodes and select the Sales By Employee Chart file.



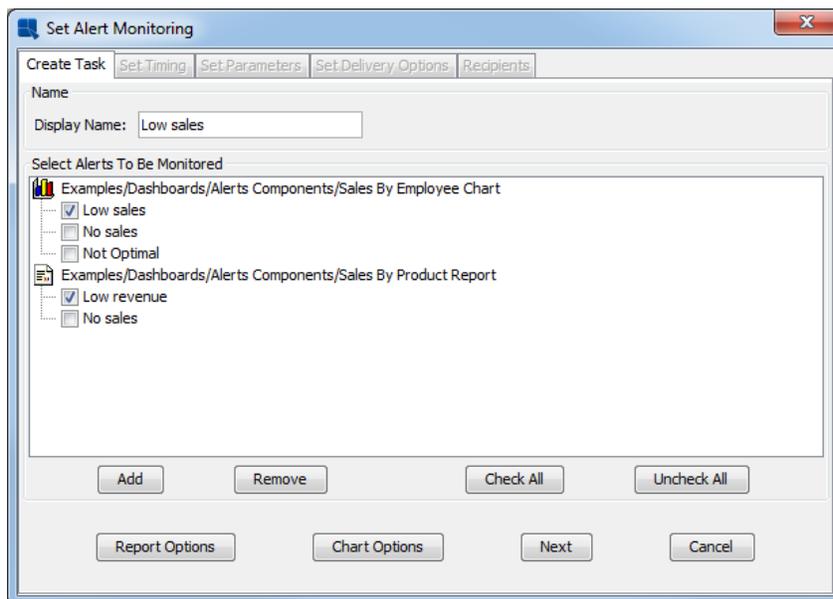
*Sales By Employee Chart Location*

Select *Set Alert Monitoring* from the *Schedule/Archive* menu. The following dialog should show up:



*Setup Alert Monitoring Dialog*

Click the *Add* button. Expand *Examples/Dashboards/Alerts Components* nodes, select the *Sales By Product Report* report and click *OK*. On the *Set Alert Monitoring* dialog select *Low sales* and *Low revenue* alerts. Type **Low sales** into the *Display Name* field. The dialog should look like this now:



*Setup Alert Monitoring Dialog*

If it does, click *Next*.

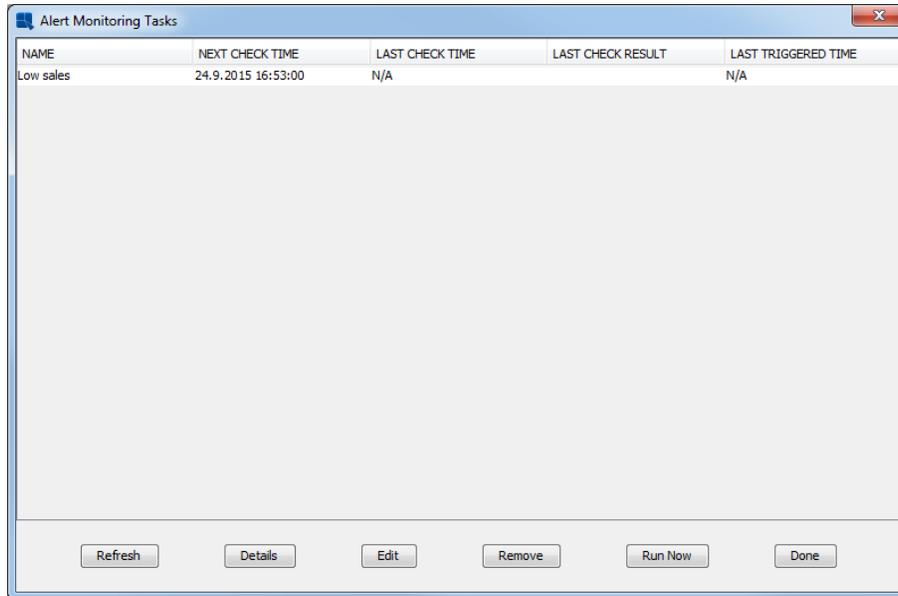
You should be on the *Set Timing* tab now. By default, the alerts would be checked just once, few minutes after configuring it. Select the *At a regular time interval* option. *Set Time Interval* dialog should show up. Default time interval is **1** minute, which is fine for us, so just click *OK* to confirm it. Uncheck the *Run Indefinitely* checkbox and click the *change* button right next to it. Set the end time one hour to the future (so you do not have to delete the job manually after you are done with this QuickStart chapter) and click *Ok*. The dialog should look like this:

*Set timing*

Click *Next*. Click *Add* to add new parameter set to the *Sales By Employee Chart* chart. Click *OK* to use default parameters. Click on the *Sales By Product Report* tab header, click *Add* and *OK* again and then click *Next*. The next step is configuring delivery options. This requires further knowledge and we will skip this step right now. If you are interested in setting delivery options, see Section 11.4.3.1 - Create/Edit monitoring dialog. Click *Next*. The following dialog should appear allowing you to check your settings before saving the alert monitoring job. Click *Confirm* to save the alert monitoring job.

*Alert Monitoring Summary*

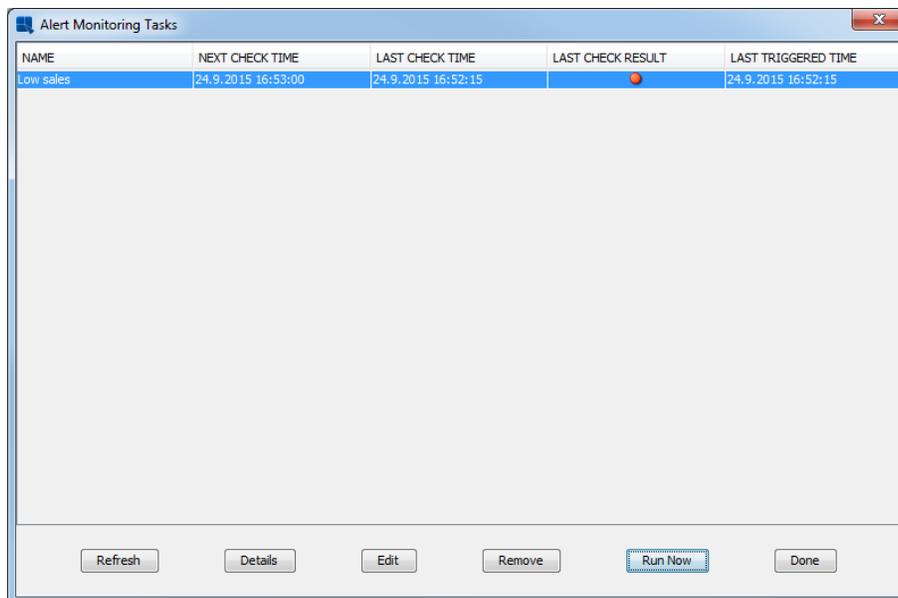
You should be back in Organizer now. Click on the *Schedule/Archive* pull-down option in the main Organizer menu and select *View Alert Monitoring Tasks* option. The following dialog should open:



*Alert Monitoring Tasks Dialog*

The `Low sales` task should be on the list. Check the *Last check time* value. If no check was performed until now, it will show `N/A`. You can either wait for next automatic check (look at the *Next check time* if you want to know when that's going to happen) and then click *Refresh*. You can also select the `Low sales` task and click *Run Now* and then click *Yes*.

After the task was checked, click *Refresh*. You should be able to see the results.

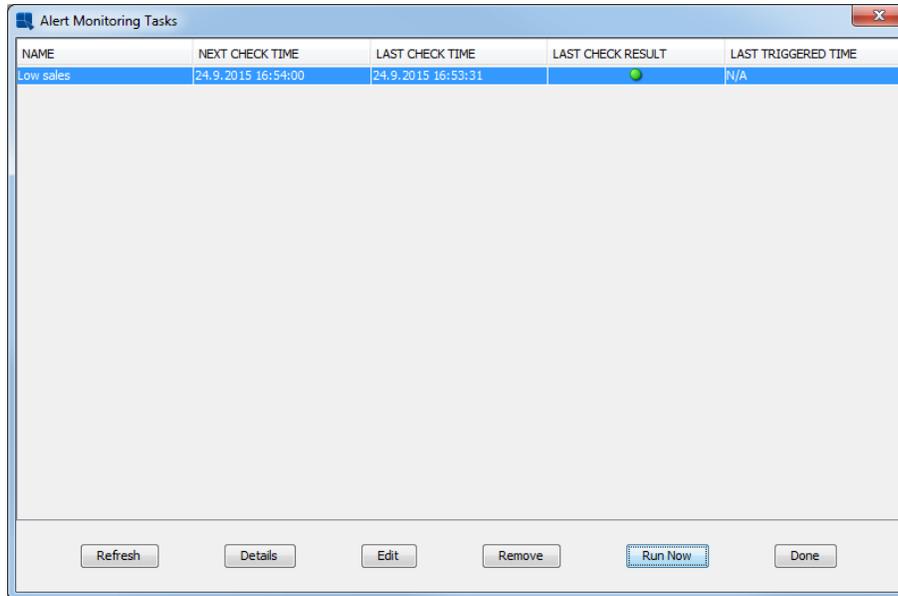


*Alert Monitoring Tasks Dialog*

The *Last check result* indicator is red. That means that at least one alert was triggered during last check. Also notice that the *Last triggered time* is now the same as *Last check time*.

Select the `Low sales` task and click *Edit*. Select the `Sales By Product Report` and then click the *Remove* button. Click *Done* and then click *Confirm* to save the changes.

Wait for the automatic check or click *Run Now* again. Click *Refresh*.



*Alert Monitoring Tasks Dialog*

Now, the *Last check result* indicator is green. That means that no alert was triggered during last check.



### Note

Please, keep in mind that this is just a quick guide which should guide you through basic steps of configuring alert monitoring. In reality, data in the database would be changing constantly; therefore, you would not have to change alert monitoring job properties in order to trigger/not trigger an alert.

## Q.12. API

This section contains a series of short exercises designed to illustrate some of the basic features of the Report API and Chart API. For more details about any of the features described in this section, please see the Programming guide portion of the documentation.

### Q.12.1. Set Up Environment

The code, given in this guide, has been created with the following in mind:

- ERES Server is up and running;
- Tomcat has been installed (with default configuration) along with ERES and is being used as the servlet container and the web-server;

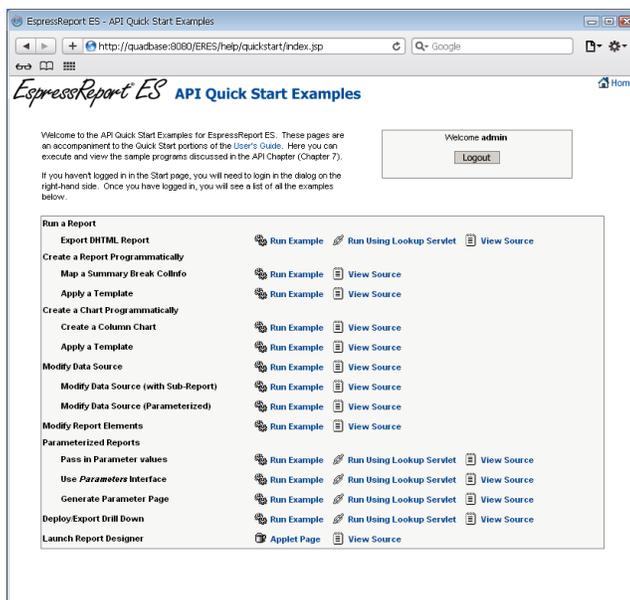
Any code (both source and compiled version) given in this guide is also under the `<ERES installation directory>/WEB-INF/classes/help/quickstart` directory. Any HTML and jsp files will be under the `<ERES installation directory>/help/quickstart` directory. Similarly, any templates would be under the `help/quickstart/templates` directory.

Most of the files need not be moved in order to run the exercises. In the few instances where the files have to be moved and/or modified, instructions will be given.

The templates (referenced in this guide), which uses a database as a data source, uses the HSQL java database. For windows users, there are alternate templates available, which use Access. These templates can be found under the `<ERES installation directory>/help/quickstart/templates/Access` directory. The templates will again follow the naming convention specified above along with “\_Acc” (without the double quotes) before the “.rpt” (without the double quotes) extension.

You can run the API QuickStart by going to `http://<machineName>:<portNumber>/<ERES context>/help/quickstart/index.jsp` (or `https://<machineName>:<portNumber>/<ERES context>/help/quickstart/index.jsp` if you are using HTTPS). Make sure that ERES Server

is running before running the examples. Once you connect to the URL, you should see the following document in the browser:



*API QuickStart Index Page*

Please note that while some of the API can be used without ERES Server and in other application servers, the code in this guide was designed with ERES Server and Tomcat in mind. Please refer to the Programming guide and Configuration guide to switch to other configurations.

## Q.12.2. Run a Report

The following sections show how to run a pre-existing template (`ExportDHTMLReport.pak` located in the `<ERES installation directory>/help/quickstart/templates` directory) in a jsp application using the Report API and also using the LookupServlet servlet.

Each section shows the code to generate the report and any steps necessary to deploy.

### Q.12.2.1. JSP Application

The following code shows how to display an existing report template (in this case `ExportDHTMLReport.pak`) in a jsp application:

```
// Connect to the ERES Server
QbReport.setEspressManagerUsed(true);
QbReport.useServlet(true);
QbReport.setServletRunner(protocol + "://" + host + ":" + port);

QbReport.setServletContext(quadbases.common.client.ServerMessage.getServletContext())
// Open the report and export it as DHTML, return the result as a String
QbReport report = new QbReport(null, help/quickstart/templates/
ExportDHTMLReport.pak);
ByteArrayOutputStream data = new ByteArrayOutputStream(2048);
OutputStream out = new BufferedOutputStream(data);
report.export(QbReport.DHTML, out);
out.flush();
```

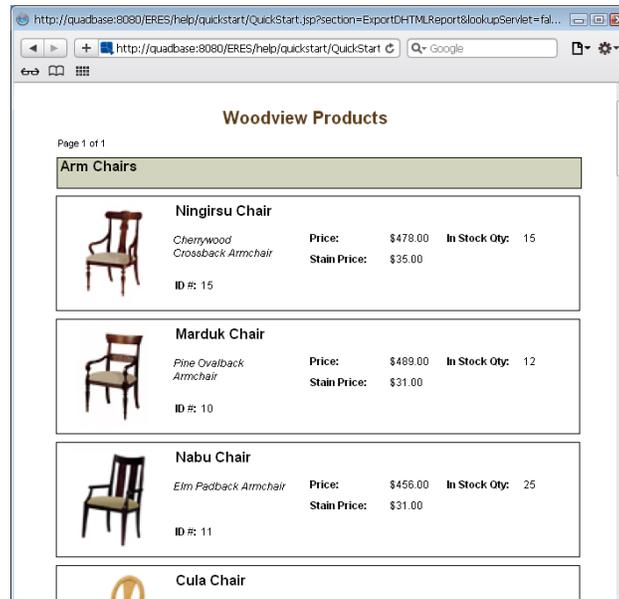


#### Note

The above piece of code is not complete. The code above is the core ERES Report API code needed to open a report template, export it to DHTML and stream the DHTML content back to the client browser.

The class file for the above source is located in the <ERES installation directory>/WEB-INF/classes/help/quickstart directory.

When the jsp application is run, by selecting the appropriate link from the main QuickStart API example page, the following report is shown:



*Report Generated*

The main part of the code is in the `getReport` method in the `ExportDHTMLReport` bean. There, a `QbReport` object called `report` is created using the `ExportDHTMLReport.pak` template. The `QbReport` is then exported, as DHTML content, to a `OutputStream`. The following constructor is used to create the `QbReport` object:

```
QbReport(Object parent, String reportTemplateName);
```

### Q.12.2.2. LookupServlet Servlet

In addition to the above approach, you can simply pass in the name of the template and the export format desired to the `LookupServlet` and let it do the work.

The following code shows how to display an existing report template (in this case `ExportDHTMLReport.pak`) using the `LookupServlet`:

```
String contextPath =
quadbase.common.client.ServerMessage.getServletContext();

// Based on the ERES context, get the http location of the files used in
this example
int lastSlash = contextPath.lastIndexOf(/);
String eresPath = contextPath.substring(0, lastSlash);
String domain = protocol + "://" + host + ":" + port;

return domain + contextPath + "/LookupServlet?USESESSION=TRUE&
URLTYPE=FORREPORT&" + "TemplatePath=" + domain + eresPath +
"/help/quickstart/templates/ExportDHTMLReport.pak&MultiPageExport=false";
```

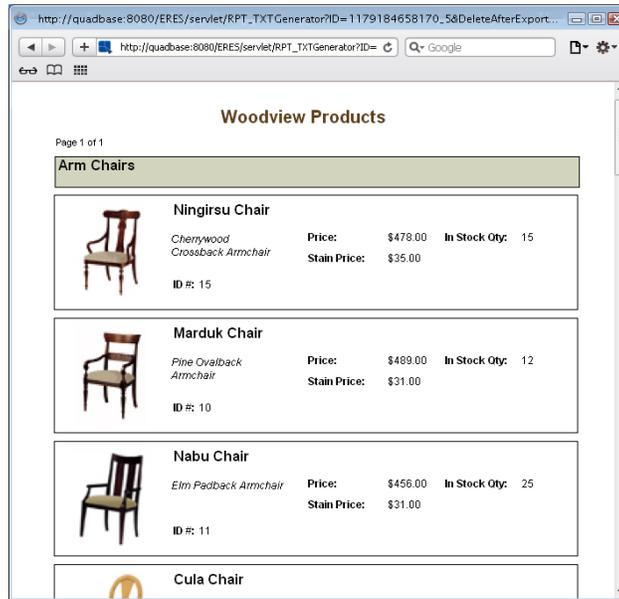


#### Note

The above piece of code is not complete. The code above is the core ERES API code needed to pass an existing report template and the export format to the `LookupServlet` servlet.

The class file for the above source is located in the <ERES installation directory>/WEB-INF/classes/help/quickstart directory.

When the jsp application is run, by selecting the appropriate link from the main QuickStart API example page, the following report is shown:



*Report Generated*

The main part of the code is in the `getReportUrl` method in the `ExportDHTMLReport` bean. There, the full path to the report template is created and passed to the `LookupServlet` servlet. The export format is also passed and the `LookupServlet` will then return the DHTML content.

## Q.12.3. Create a Report Programmatically

The following sections show how to create a report programmatically and apply a template, `ApplyTemplate.pak` (located in the <ERES installation directory>/help/quickstart/templates directory) to the report in an application.

Each section shows the code to generate the report and any steps necessary to deploy.

### Q.12.3.1. Map Summary Break ColInfo

The following code shows how to create a Summary Break report programmatically (using `MapSummaryBreak-ColInfo.txt` as the data source):

```
// Connect to ERES Server
```

```
QbReport.setEspressManagerUsed(true);
QbReport.useServlet(true);
QbReport.setServletRunner(protocol + "://" + host + ":" + port);
```

```
QbReport.setServletContext(quadbase.common.client.ServerMessage.getServletContext());
```

```
// Specify Column Mapping
```

```
ColInfo colInfo[] = new ColInfo[4];

colInfo[0] = new ColInfo(0);
colInfo[0].setRowBreak(true);
colInfo[1] = new ColInfo(1);
colInfo[1].setAggregation(false, ColInfo.NONE);
```

```

colInfo[2] = new ColInfo(2);
colInfo[2].setAggregation(false, ColInfo.SUM);

colInfo[3] = new ColInfo(3);
colInfo[3].setAggregation(false, ColInfo.SUM);
// Create the report and export it as DHTML, return the result as a String

QbReport report = new QbReport(null, QbReport.SUMMARY,
"help/quickstart/data/MapSummaryBreakColInfo.txt", colInfo, null);

ByteArrayOutputStream data = new ByteArrayOutputStream(2048);
OutputStream out = new BufferedOutputStream(data);

report.export(QbReport.DHTML, out);
out.flush();

```



### Note

The above piece of code is not complete. The code above is the core ERES Report API code needed to create a Summary Break report, export it to DHTML, and stream the DHTML content back to the client browser.

The class file for the above source is located in the <ERES installation directory>/WEB-INF/classes/help/quickstart directory.

When the jsp application is run, by selecting the appropriate link from the main QuickStart API example page, the following report is shown:

Product	Product Name	Orders	Units Shipped
Arm Chairs	Aded Chair	4	50
	Cula Chair	10	129
	Marbuk Chair	5	68
	Nabu Chair	4	53
	Ningiru Chair	4	27
	Nisaba Chair	15	173
	Nusku Chair	10	124
	Sruqanma	6	79
	Shimaliya Chair	7	81
		<b>65</b>	<b>784</b>
Double Dressers	Sekamet	1	6
	Serket Dresser	4	36
	Set Dresser	2	22
	Shu Dresser	1	12
	Tefnut Dresser	1	9
	Thoth Dresser	2	22
		<b>11</b>	<b>107</b>
Oval Tables	Imhotep Table	4	28
	Isis Table	5	50

*Report Generated*

Please note that when the above code is run, the report generated is of default look and feel without any additional formatting.

The main part of the code is in the `getReport` method in the `MapSummaryBreakColInfo` bean. There, a `QbReport` object called `report` is created using the specified Column Mapping, the specified Report Type, and the specified Data Source. The following constructor is used:

```
QbReport(Object parent, int reportType, String dataSource,
```

```
ColInfo[] columnMapping, String reportTemplate);
```

### Q.12.3.2. Apply Template

You can pass in a different formatting (i.e. different from the default look and feel) by specifying a template during the creation of the QbReport object.

The following code shows how to create a Summary Break report programmatically (using MapSummaryBreak-ColInfo.txt as the data source and ApplyTemplate.pak as the template):

```
// Connect to ERES Server
QbReport.setEspressManagerUsed(true);
QbReport.useServlet(true);
QbReport.setServletRunner(protocol + "://" + host + ":" + port);

QbReport.setServletContext(quadbase.common.client.ServerMessage.getServletContext());

// Specify Column Mapping

ColInfo colInfo[] = new ColInfo[4];
colInfo[0] = new ColInfo(0);
colInfo[0].setRowBreak(true);
colInfo[1] = new ColInfo(1);
colInfo[1].setAggregation(false, ColInfo.NONE);
colInfo[2] = new ColInfo(2);

colInfo[2].setAggregation(false, ColInfo.SUM);
colInfo[3] = new ColInfo(3);
colInfo[3].setAggregation(false, ColInfo.SUM);
// Create the report and export it as DHTML, return the result as a String

QbReport report = new QbReport(null, QbReport.SUMMARY,
"help/quickstart/data/MapSummaryBreakColInfo.txt", colInfo,
"help/quickstart/templates/ApplyTemplate.pak");

ByteArrayOutputStream data = new ByteArrayOutputStream(2048);

OutputStream out = new BufferedOutputStream(data);
report.export(QbReport.DHTML, out);
out.flush();
```



#### Note

The above piece of code is not complete. The code above is the core ERES Report API code needed to create a Summary Break report, apply a template, export it to DHTML, and stream the DHTML content back to the client browser.

The class file for the above source is located in the <ERES installation directory>/WEB-INF/classes/help/quickstart directory.

When the jsp application is run by selecting the appropriate link from the main QuickStart API example page, the following report is shown:

Product Category	Product Name	Orders	Units Shipped
Arm Chairs	Aided Chair	4	50
	Cula Chair	10	129
	Marduk Chair	5	68
	Nabu Chair	4	53
	Ningirsu Chair	4	27
	Nisaba Chair	15	173
	Nusku Chair	10	124
	Souqamuna Chair	6	79
	Shimaliya Chair	7	81
Total for Arm Chairs		65	784
Double Dressers	Sekhmed Dresser	1	6
	Serket Dresser	4	36
	Set Dresser	2	22
	Shu Dresser	1	12
	Tefnut Dresser	1	9
	Thoth Dresser	2	22
Total for Double Dressers		11	107
Oval Tables	Imhotep Table	4	28
	Isis Table	5	50
	Wahneset Table	1	6

Report Generated

The main part of the code is in the `getReport` method in the `ApplyTemplate` bean. There, a `QbReport` object called `report` is created using the specified Column Mapping, the specified Report Type, the specified Data Source, and the specified Report Template. The following constructor is used:

```
QbReport(Object parent, int reportType, String dataSource,
ColInfo[] columnMapping, String reportTemplate);
```

## Q.12.4. Create a Chart Programmatically

The following sections show how to create a chart programmatically and apply a template, `ApplyChartTemplate.tpl` (located in the `<ERES installation directory>/help/quickstart/templates directory`) to the report in an application.

Each section shows the code to generate the chart and any steps necessary to deploy.

### Q.12.4.1. Map Column Chart ColInfo

The following code shows how to create a two-dimensional Column chart programmatically (using `CreateColumnChart.txt` as the data source):

```
// Connect to ERES Server
QbChart.setEspressManagerUsed(true);
QbChart.useServlet(true);

QbChart.setServletRunner(protocol + "://" + host + ":" + port);

QbChart.setServletContext(quadbases.common.client.ServerMessage.getServletContext());

// Specify Column Mapping
ColInfo colInfo = new ColInfo();
colInfo.series = 1;
colInfo.category = 0;

colInfo.value = 2;
// Create the chart and export it as PNG
```

```
QbChart chart = new QbChart(null, QbChart.VIEW2D,
    QbChart.COL, "help/quickstart/data/CreateColumnChart.txt", false, null,
    colInfo, null);
```

```
chart.export(QbChart.PNG, out, 450, 350);
out.flush();
```

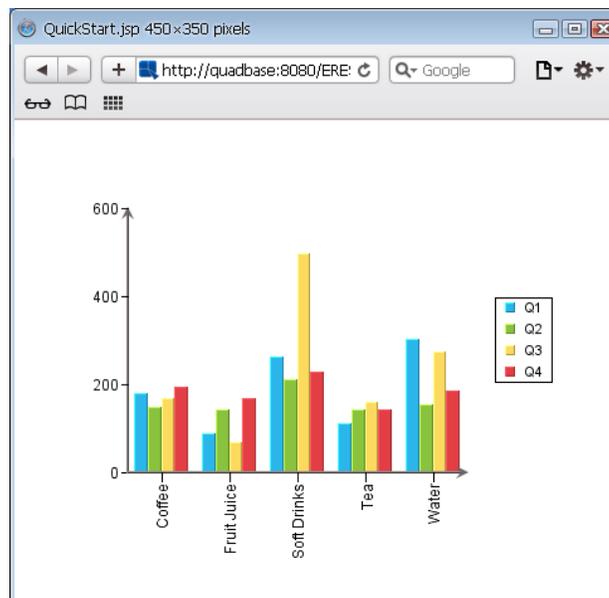


### Note

The above piece of code is not complete. The code above is the core ERES Chart API code needed to create a two-dimensional Column chart, export it to PNG, and stream the PNG content back to the client browser.

The class file for the above source is located in the <ERES installation directory>/WEB-INF/classes/help/quickstart directory.

When the jsp application is run, by selecting the appropriate link from the main QuickStart API example page, the following chart is shown:



*Chart Generated*

Please note that when the above code is run, the chart generated is of default look and feel without any additional formatting.

The main part of the code is in the `streamChart` method in the `CreateColumnChart` bean. There, a `QbChart` object called `chart` is created using the specified Column Mapping, the specified Chart Type and the specified Data Source. The following constructor is used:

```
QbChart(Applet applet, int dimensionType, int chartType,
    String dataSource, boolean doTranspose, int[] transposedColumns,
    ColInfo[] columnMapping, String reportTemplate);
```

### Q.12.4.2. Apply Template

You can pass in a different formatting (i.e. different from the default look and feel) by specifying a template during the creation of the `QbChart` object.

The following code shows how to create a two-dimensional Column chart programmatically (using `CreateColumnChart.txt` as the data source and `ApplyChartTemplate.tpl` as the template):

```
// Connect to ERES Server
QbChart.setEspressManagerUsed(true);

QbChart.useServlet(true);
QbChart.setServletRunner(protocol + "://" + host + ":" + port);

QbChart.setServletContext(quadbase.common.client.ServerMessage.getServletContext());

// Specify Column Mapping
ColInfo colInfo = new ColInfo();
colInfo.series = 1;

colInfo.category = 0;
colInfo.value = 2;

// Create the chart and export it as PNG
QbChart chart = new QbChart(null, QbChart.VIEW2D, QbChart.COL,
"help/quickstart/data/CreateColumnChart.txt", false, null, colInfo,
"help/quickstart/templates/ApplyChartTemplate.tpl");

chart.export(QbChart.PNG, out, 450, 350);
out.flush();
```

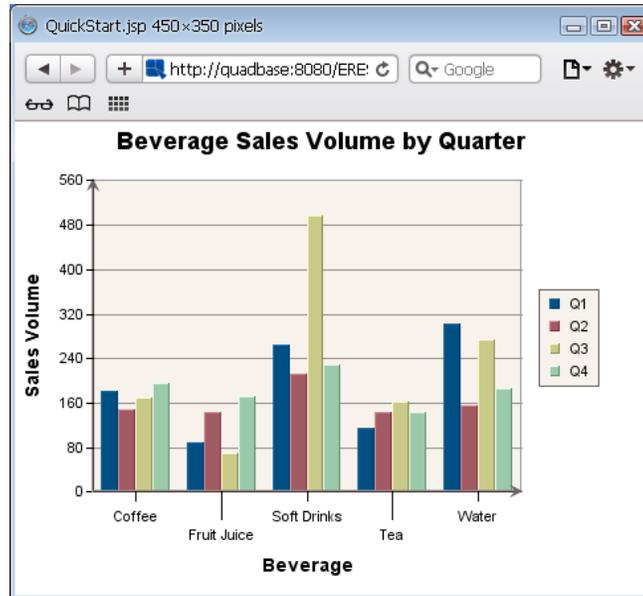


### Note

The above piece of code is not complete. The code above is the core ERES Chart API code needed to create a two-dimensional column chart, apply a template to it, export it to PNG, and stream the PNG content back to the client browser.

The class file for the above source is located in the `<ERES installation directory>/WEB-INF/classes/help/quickstart` directory.

When the jsp application is run by selecting the appropriate link from the main QuickStart API example page, the following chart is shown:



*Chart Generated*

The main part of the code is in the `streamChart` method in the `QuickStart1042` bean. There, a `QbChart` object called `report` is created using the specified Column Mapping, the specified Chart Type, the specified Data Source, and the specified Chart Template. The following constructor is used:

```
QbChart(Applet applet, int dimensionType, int chartType,
String dataSource, boolean doTranspose, int[] transposedColumns,
ColInfo[] columnMapping, String reportTemplate);
```

## Q.12.5. Modify Data Source of a Report

The following code shows how to modify a report's (and its accompanying sub-report's, drill-down's, and independent chart's) data source, without having to create a new `QbReport` object. The `QbReport` object (created from `ModifyDataSource.pak`) is opened with backup data (this is so that the database is not hit unnecessarily). The data source is then changed to the Access Woodview database.

```
QbReport.setEspressManagerUsed(true);
QbReport.useServlet(true);
QbReport.setServletRunner(protocol + "://" + host + ":" + port);

QbReport.setServletContext(quadbases.common.client.ServerMessage.getServletContext());

// Create the report using the two rows of back-up data
QbReport report = new QbReport(object, "help/quickstart/templates/
ModifyDataSource.pak",
false, false, false, true);

modifyDataSource(report);

ByteArrayOutputStream data = new ByteArrayOutputStream(2048);
OutputStream out = new BufferedOutputStream(data);

report.setServletDirectory(quadbases.common.client.ServerMessage.getServletContext());
```

```

    if (protocol.equalsIgnoreCase("https")) report.setHttpsDynamicExport(true,
host, port);
    else report.setDynamicExport(true, host, port);
    report.export(QbReport.DHTML, out);
    out.flush();

    static void modifyDataSource(QbReport report)
    throws Exception {

// Begin Code : Specification for new Database
String newDatabaseURL = "jdbc:odbc:Woodview";</br>
String newDatabaseDriver = "sun.jdbc.odbc.JdbcOdbcDriver";</br>
String newDatabaseUserid = "";</br>
String newDatabasePassword = "";</br>// End Code : Specification for new
Database

// Begin Code : Change the data source of the main
report and all ancillary templates
report.getInputData().setAllDatabaseInfo(newDatabaseURL, newDatabaseDriver,
newDatabaseUserid, newDatabasePassword);
// End Code : Change the data source of the main report and all
ancillary templates
    }

```

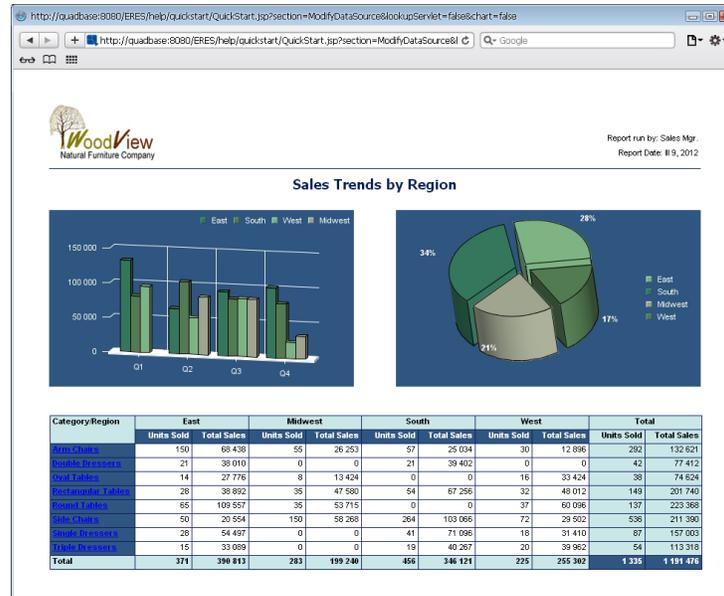


### Note

The above piece of code is not complete. The code above is the core ERES Report API code needed to modify the datasource of a report, export it to DHTML, and stream the DHTML content back to the client browser.

The class file for the above source is located in the <ERES installation directory>/WEB-INF/classes/help/quickstart/classes directory.

When the jsp application is run, by selecting the appropriate link from the main QuickStart API example page, the following report is shown:



Report Generated

The main part of the code is in the `getReport` method in the `ModifyDataSource` bean. A `QbReport` object called `report` is created there. The data source for the report, sub-report, drill-down, and chart (with independent data source) is changed using the following method in the `quadbase.reportdesigner.util.IInput-Data` interface:

```
setAllDatabaseInfo(String url, String driver,
String userid, String password);
```

### Q.12.5.1. Modify Data Source of a Report (with Sub-Report)

The following code shows how to modify a report's (and its accompanying sub-report's) data source, without having to create a new `QbReport` object. The `QbReport` object (created from `ModifyDataSourceWithSubReport_Acc.pak`) is opened with backup data (this is so that the database is not hit unnecessarily). The `datasource` is then changed from the Access Woodview database to the HSQLDB Woodview database.

```
// Connect to ERES Server

QbReport.setEspressManagerUsed(true);
QbReport.useServlet(true);
QbReport.setServletRunner(protocol + "://" + host + ":" + port);

QbReport.setServletContext(quadbase.common.client.ServerMessage.getServletContext());

// Create the report using the two rows of back-up data
QbReport report = new QbReport(null,
    "help/quickstart/templates/Access/ModifyDataSourceWithSubReport_Acc.pak",
    false, false,
    false, true);

modifyDataSource(report);
ByteArrayOutputStream data = new ByteArrayOutputStream(2048);
OutputStream out = new BufferedOutputStream(data);
```

```
report.setServletDirectory(quadbse.common.client.ServerMessage.getServletContext());
  if (protocol.equalsIgnoreCase("https")) report.setHttpsDynamicExport(true,
host, port);
  else report.setDynamicExport(true, host, port);

report.export(QbReport.DHTML, out);
out.flush();
```

```
static void modifyDataSource(QbReport report) throws
Exception {

    // Specification for new Database connection
    String newDatabaseURL =
        "jdbc:hsqldb:help/examples/DataSources/database/woodview";

    String newDatabaseDriver = "org.hsqldb.jdbcDriver";
    String newDatabaseUserid = "sa";
    String newDatabasePassword = "";

    String newDatabaseReportQuery = "select year(o.orderdate)
as \"Year\", month(o.orderdate) as \"Month\", count(o.orderid) as
\"Orders\",
    sum(od.quantity) as \"Units Sold\", sum((p.unitprice + od.staincost) *
od.quantity)
as \"Total Sales\" from orders o, order_details od, products p where
o.orderid = od.orderid
and p.productid = od.productid group by year(o.orderdate),
month(o.orderdate)
order by year(o.orderdate), month(o.orderdate);";

    String newDatabaseSubReportQuery = "select c.region
as \"Region\", year(o.orderdate) as \"Year\", sum((p.unitprice +
od.staincost)
* od.quantity) as \"Total Sales\" from customers c, orders o, products
p,
order_details od where c.customerid = o.customerid and o.orderid =
od.orderid
and od.productid = p.productid group by c.region, year(o.orderdate);";

    // Get a handle to the Sub-Report and change its data source

    SubReportObject subReportObject =
        (SubReportObject)report.getTable().getHeader().getData("TBL0_HDR_LB0");
    QbReport subReport =
        (QbReport)subReportObject.getSubReport(false, false, true, report);

    // Get the query and pass in new database info
```

```

DBInfo newSubReportDatabaseInfo = new DBInfo(newDatabaseURL,
newDatabaseDriver, newDatabaseUserid, newDatabasePassword,
newDatabaseSubReportQuery);

subReport.getInputData().setDatabaseInfo(newSubReportDatabaseInfo);

// Change the data source of the main report

// Get the query and pass in new database info
DBInfo newReportDatabaseInfo = new DBInfo(newDatabaseURL,
newDatabaseDriver, newDatabaseUserid, newDatabasePassword,
newDatabaseReportQuery);

report.getInputData().setDatabaseInfo(newReportDatabaseInfo);
}

```



### Note

The above piece of code is not complete. The code above is the core ERES Report API code needed to modify the datasource of a report (and its subreport), export it to DHTML, and stream the DHTML content back to the client browser.

The class file for the above source is located in the <ERES installation directory>/WEB-INF/classes/help/quickstart directory.

When the jsp application is run, by selecting the appropriate link from the main QuickStart API example page, the following report is shown:



*Report Generated*

The main part of the code is in the `getReport` method in the `ModifyDataSourceWithSubReport` bean. There, a `QbReport` object called `report` is created. The data source for the report and the sub-report is changed using the following method in the `quadbase.reportdesigner.util.IInputData` interface:

```
setDatabaseInfo(IDatabaseInfo db);
```

The sub-report is obtained by getting a handle to the cell containing the sub-report and then calling the sub-report. The sub-report is also opened using its backup data. This is done by:

```
SubReportObject subReportObject = (SubReportObject)<Handle
to desired Report Section>!.getData(String ID);

(QbReport)subreport = (QbReport)subReportObject.getSubReport(
boolean isEnterpriseServer, boolean optimizeMemory, boolean useBackupData,
IReport report);
```

Please note that if you wish to change the data source from the HSQLDB Woodview database to the Access Woodview database, you will need to change the following lines of code from:

```
QbReport report = new QbReport(object,
help/quickstart/templates/Access/ModifyDataSourceWithSubReport_Acc.pak,
false, false,
false, true);

// Begin Code : Specification for new Database
String newDatabaseURL = "jdbc:hsqldb:help/examples/DataSources/database/
woodview";

String newDatabaseDriver = "org.hsqldb.jdbcDriver";
String newDatabaseUserid = "sa";
String newDatabasePassword = "''";

String newDatabaseReportQuery = "select year(o.orderdate) as \"Year\",
month(o.orderdate) as \"Month\", count(o.orderid) as \"Orders\",
sum(od.quantity)
as \"Units Sold\", sum((p.unitprice + od.staincost) * od.quantity) as
\"Total
Sales\" from orders o, order_details od, products p where o.orderid =
od.orderid
and p.productid = od.productid group by year(o.orderdate),
month(o.orderdate)
order by year(o.orderdate), month(o.orderdate);";

String newDatabaseSubReportQuery = "select c.region as \"Region\",
year(o.orderdate) as \"Year\", sum((p.unitprice + od.staincost) *
od.quantity)
as \"Total Sales\" from customers c, orders o, products p, order_details
od where c.customerid = o.customerid and o.orderid = od.orderid and
od.productid
= p.productid group by c.region, year(o.orderdate);";

to

QbReport report = new QbReport(object,
help/quickstart/templates/ModifyDataSourceWithSubReport.pak, false, false,
false, true);

// Begin Code : Specification for new Database
```

```

String newDatabaseURL = "jdbc:odbc:Woodview";
String newDatabaseDriver = "sun.jdbc.odbc.JdbcOdbcDriver";

String newDatabaseUserid = "";
String newDatabasePassword = "";

String newDatabaseReportQuery = "select year(o.orderdate) as \"Year\",
month(o.orderdate) as \"Month\", count(o.orderid) as \"Orders\",
sum(od.quantity)
as \"Units Sold\", sum((p.unitprice + od.staincost) * od.quantity) as
\"Total
Sales\" from orders o, [order details] od, products p where o.orderid =
od.orderid and p.productid = od.productid group by year(o.orderdate),
month(o.orderdate)
order by year(o.orderdate), month(o.orderdate);";

String newDatabaseSubReportQuery = "select c.region as \"Region\",
year(o.orderdate) as \"Year\", sum((p.unitprice + od.staincost) *
od.quantity)
as \"Total Sales\" from customers c, orders o, products p, [order details]
od where c.customerid = o.customerid and o.orderid = od.orderid and
od.productid
= p.productid group by c.region, year(o.orderdate);";

```

### Q.12.5.2. Modify Data Source of a Parameterized Report (with Parameterized Sub-Report)

The following code shows how to modify a report's and its accompanying sub-report's parameterized data sources, without having to create a new QbReport object. The QbReport object (created from ModifyDataSourceParameterized\_Acc.pak) is opened with backup data (this is so that the database is not hit unnecessarily). The datasource is then changed from the Access Woodview database to the HSQLDB Woodview database.

```

// Connect to ERES Server
QbReport.setEspressManagerUsed(true);
QbReport.useServlet(true);
QbReport.setServletRunner(protocol + "://" + host + ":" + port);

QbReport.setServletContext(quadbase.common.client.ServerMessage.getServletContext());

// Open the report using the two rows of backup data
QbReport report = new QbReport(null,
"help/quickstart/templates/Access/ModifyDataSourceParameterized_Acc.pak",
false, false,
false, true);

changeDataSource(report);
report.setServletDirectory
(quadbase.common.client.ServerMessage.getServletContext());
if (protocol.equalsIgnoreCase("https")) report.setHttpsDynamicExport(true,
host, port);

```

```
else report.setDynamicExport(true, host, port);

ByteArrayOutputStream data = new ByteArrayOutputStream(2048);
OutputStream out = new BufferedOutputStream(data);

report.export(QbReport.DHTML, out);
out.flush();

static void changeDataSource(QbReport report) throws
Exception {

    <div class="dir">
    // Specification for the new Database
    String URL = "jdbc:hsqldb:help/examples/DataSources/database/woodview";
    String driver = "org.hsqldb.jdbcDriver";
    String userid = "sa";

    String passwd = "";

    String newDatabaseSubReportQuery =
    "select c.categoryname as \"Category\", p.productname as \"Product\",
    cu.region as \"Region\", sum(od.staincost + p.unitprice) * od.quantity)
    as \"Sales\" from categories c, products p, customers cu, orders o,
order_details od
    where c.categoryid = p.categoryid and p.productid = od.productid
    and cu.customerid = o.customerid and o.orderid = od.orderid and
    c.categoryname IN (:category) group by c.categoryname, p.productname,
cu.region";

    // Get a handle to the SubReport and change its datasource
    SubReportObject subReportObject =

(SubReportObject)report.getTable().getHeader().getData("TBL0_HDR_SRPT0");
    QbReport subReport = (QbReport)subReportObject.getSubReport(false,
false, true, report);

    // Get the parameter information of the subreport

    // and pass in the new database information along with the parameter
information
    IQueryInParam[] subReportParameters =

((IQueryFileInfo)subReport.getInputData().getDatabaseInfo()).getInParam();
    SimpleQueryFileInfo subReportInfo =
```

```

new SimpleQueryFileInfo(URL, driver, userid, passwd,
newDatabaseSubReportQuery);
subReportInfo.setInParam(subReportParameters);

subReport.getInputData().setDatabaseInfo(subReportInfo);

// Get the parameter information of the main report and pass
// in the new database information along with the parameter information
// Pass in the parameter values for the main report (which is
// then picked up by the subreport)
Vector paramValues = new Vector();
paramValues.addElement(new String("Arm Chairs"));
paramValues.addElement(new String("Double Dressers"));

paramValues.addElement(new String("Round Tables"));

// Get the parameter properties information and pass in the
// value of the parameter
IQueryInParam[] reportParameters =
((IQueryFileInfo)report.getInputData().getDatabaseInfo()).getInParam();
((IQueryMultiValueInParam)reportParameters[0]).setValues(paramValues);

// Get the query for the main report from the report template itself
// instead of passing in a new query
SimpleQueryFileInfo reportInfo = new SimpleQueryFileInfo(URL, driver,
userid, passwd, report.getInputData().getDatabaseInfo().getQuery());
reportInfo.setInParam(reportParameters);

report.getInputData().setDatabaseInfo(reportInfo);

</div>
}

```



### Note

The above piece of code is not complete. The code above is the core ERES Report API code needed to modify the datasource of a parameterized report (and its parameterized subreport), export it to DHTML, and stream the DHTML content back to the client browser.

The class file for the above source is located in the <ERES installation directory>/WEB-INF/classes/help/quickstart directory.

When the jsp application is run, by selecting the appropriate link from the main QuickStart API example page, the following report is shown:



---

```
QbReport report = new QbReport(object,
    help/quickstart/templates/Access/ModifyDataSourceParameterized_Acc.pak,
    false, false,
    false, true);
```

```
// Specification for the new Database
String newDatabaseURL = "jdbc:hsqldb:help/examples/DataSources/database/
woodview";
```

```
String newDatabaseDriver = "org.hsqldb.jdbcDriver";
String newDatabaseUserid = "sa";
String newDatabasePassword = "";
```

```
String newDatabaseSubReportQuery =
    "select c.categoryname as \"Category\", p.productname as \"Product\",
    cu.region as \"Region\", sum((od.staincost + p.unitprice) * od.quantity)
    as \"Sales\" from categories c, products p, customers cu, orders o,
    order_details od
    where c.categoryid = p.categoryid and p.productid = od.productid and
    cu.customerid = o.customerid and o.orderid = od.orderid and
    c.categoryname IN (:category) group by c.categoryname, p.productname,
    cu.region;";
```

to

```
QbReport report = new QbReport(object,
    help/quickstart/templates/ModifyDataSourceParameterized.pak, false, false,
    false, true);
```

```
// Specification for the new Database
String newDatabaseURL = "jdbc:odbc:Woodview";
String newDatabaseDriver = "sun.jdbc.odbc.JdbcOdbcDriver";
String newDatabaseUserid = "";
```

```
String newDatabasePassword = "";
```

```
String newDatabaseSubReportQuery =
    "select c.categoryname as \"Category\", p.productname as \"Product\",
    cu.region as \"Region\", sum((od.staincost + p.unitprice) * od.quantity)
    as \"Sales\"
    from categories c, products p, customers cu, orders o, [order details] od
    where c.categoryid = p.categoryid and p.productid = od.productid and
    cu.customerid = o.customerid and o.orderid = od.orderid and
    c.categoryname IN (:category) group by c.categoryname, p.productname,
    cu.region;";
```

You can also pass in a `java.sql.Connection` object, instead of passing in the URL, driver, userid and password of the database. For example, if you wish to pass in a `Connection` object, `conn`, you would need to change the following lines of code:

```
SimpleQueryFileInfo subReportInfo =
    new SimpleQueryFileInfo(URL, driver, userid, passwd,
    newDatabaseSubReportQuery);
```

```
SimpleQueryFileInfo reportInfo = new SimpleQueryFileInfo(URL, driver,
userid, passwd, report.getInputData().getDatabaseInfo().getQuery());
```

to

```
SimpleQueryFileInfo subReportInfo =
new SimpleQueryFileInfo(conn, newDatabaseSubReportQuery);
SimpleQueryFileInfo reportInfo = new SimpleQueryFileInfo(conn,
report.getInputData().getDatabaseInfo().getQuery());
```

## Q.12.6. Modify Report Elements

The following code shows how to modify certain elements of the report programmatically. The QbReport object is created from `ModifyReportElements.pak`.

```
// Connect to ERES Server
QbReport.setEspressManagerUsed(true);
QbReport.useServlet(true);

QbReport.setServletRunner(protocol + "://" + host + ":" + port);

QbReport.setServletContext(quadbase.common.client.ServerMessage.getServletContext());

QbReport report = new QbReport(null,
"help/quickstart/templates/ModifyReportElements.pak");

modifyElements(report);

ByteArrayOutputStream data = new ByteArrayOutputStream(2048);
OutputStream out = new BufferedOutputStream(data);

report.export(QbReport.DHTML, out);
out.flush();

static void modifyElements(QbReport report) {

// Set Dual colors
int numberOfColumns = report.getTable().getColumnCount();

for (int i = 0; i < numberOfColumns; i++) {

<div class="dir">
ReportColumn column = report.getTable().getColumn(i);
column.setAlternateRow(1);
column.setBgColor2(new Color(245,245,238));
column.setFontColor2(new Color(0,0,0));

column.setFont2(new Font("Dialog", Font.PLAIN, 8));
```

```

    </div>
  }

  // Add title to Report Header
  ReportCell title = new ReportCell();
  title.setText("Top 10 Customers");

  title.setBgColor(new Color(255,255,255));
  title.setFontColor(new Color(0,54,100));
  title.setFont(new Font("Dialog", Font.BOLD, 14));
  title.setAlign(IAalignConstants.ALIGN_LEFT);
  title.setWidth(2.1);
  title.setHeight(0.4);

  title.setX(0);
  title.setY(0);
  report.getReportHeader().addData(title);

  // Add a formula
  ReportCell formulaCell = new ReportCell();
  Formula formula = new Formula("totalSales", "sum({Total Sales})");

  report.addFormula(formula);
  formulaCell.setFormulaObj(formula);
  formulaCell.setBgColor(new Color(255,255,255));
  formulaCell.setFontColor(new Color(0,0,0));
  formulaCell.setFont(new Font("Dialog", Font.BOLD, 8));
  formulaCell.setAlign(IAalignConstants.ALIGN_LEFT);

  formulaCell.setWidth(1.0);
  formulaCell.setHeight(0.25);
  formulaCell.setX(4.1);
  formulaCell.setY(0);
  report.getReportFooter().addData(formulaCell);

  // Add a label

  ReportCell label = new ReportCell();
  label.setText("Total sales for top 10 customers:");
  label.setBgColor(new Color(255,255,255));
  label.setFontColor(new Color(0,54,100));
  label.setFont(new Font("Dialog", Font.BOLD, 8));
  label.setAlign(IAalignConstants.ALIGN_RIGHT);

  label.setWidth(2.1);
  label.setHeight(0.4);
  label.setX(1.9);
  label.setY(0);
  report.getReportFooter().addData(label);

```



### Note

The above piece of code is not complete. The code above is the core ERES Report API code needed to modify the various elements of a report, export it to DHTML, and stream the DHTML content back to the client browser.

The class file for the above source is located in the <ERES installation directory>/WEB-INF/classes/help/quickstart directory.

When the jsp application is run, by selecting the appropriate link from the main QuickStart API example page, the following report is shown:



Rank	Company	Orders	Units Ordered	Total Sales
1	Woodworks Furniture 88 Rio Grand Ave. Allerget, PA 38923	19	196	301 834
2	Specialty Retail 88 Spenser Drive Los Angeles, CA 88938	14	132	161 689
3	Southwest Style 21 Doger Sound Blvd. Austin, TX 20122	5	28	52 231
4	Sevilla Home & Garden 389 Jardin Rd. Sevilla, AK 62938	29	378	335 106
5	Les Chaise 77 Druguz Blvd. Little Rock, AK 12343	7	77	65 518
6	Imports & Leather Gallery 5 Baker's Lane Chicago, IL 39283	21	289	341 028
7	HomeWorld Furniture 71 Mulberry St. San Francisco, CA 98839	11	117	126 520
8	George Park Imports 300 Great Swamp Rd. George Park, PA 23445	8	97	60 661
9	Gale Distributors 345 Sewgen Drive Boontown, NY 11234	5	54	40 574
10	Furniture Pros, Inc. 44 Islander Lane Orlando, FL 63923	9	121	58 432

Total sales for top 10 customers: 1 543 593

*Report Generated*

The main part of the code is in `getReport` method in the `ModifyReportElements` bean. Report properties such as `Dual Color` are turned on and a formula, a label and a title are added to the `QbReport` object.

Dual colors are set using the following code. The dual color property is set for each table column and the alternate background color, font color and font are specified.

```
<Desired Report Column>.setAlternateRow(int
numberOfRowsBeforeAlternateColor);
<Desired Report Column>.setBgColor2(Color alternateBackgroundColor);
<Desired Report Column>.setFontColor2(Color alternateFontColor);
<Desired Report Column>.setFont2(Font alternateFont);
```

Adding a title to the Report Header is done using the following code. The `ReportCell` object is first created, the title text then set and the `ReportCell` properties such as `height`, `width`, `xPosition` and `yPosition` are specified before adding it to the Report Header section.

```
ReportCell title = new ReportCell();
title.setText(String text);
title.setBgColor(Color backgroundColor);
title.setFontColor(Color fontColor);

title.setFont(Font font);
title.setAlign(int alignment);
title.setWidth(double width);
title.setHeight(double height);
title.setX(double xPosition);
title.setY(double yPosition);

<Handle to desired Report Section>.addData(title);
```

A formula and a label are specified likewise. A `ReportCell` object is first created, the formula or label set and the `ReportCell` properties specified before adding the newly created formula or label `ReportCell` object to the appropriate section.

```
ReportCell formulaCell = new ReportCell();
Formula formula = new Formula(String formulaName, String formulaText);

report.addFormula(formula);
formulaCell.setFormulaObj(formula);
formulaCell.setBgColor(Color backgroundColor);
formulaCell.setFontColor(Color fontColor);
formulaCell.setFont(Font font);
formulaCell.setAlign(int alignment);

formulaCell.setWidth(double width);
formulaCell.setHeight(double height);
formulaCell.setX(double xPosition);
formulaCell.setY(double yPosition);
<Handle to desired Report Section>.addData(formulaCell);

ReportCell label = new ReportCell();

label.setText(String text);
label.setBgColor(Color backgroundColor);
label.setFontColor(Color fontColor);<br />
label.setFont(Font font);
label.setAlign(int alignment);
label.setWidth(double width);

label.setHeight(double height);
label.setX(double xPosition);
label.setY(double yPosition);
<Handle to desired Report Section>.addData(label);
```

## Q.12.7. Parameterized Reports

The following sections show how to run a pre-existing template (`PassInParameterValues.pak` located in the `<ERES installation directory>/help/quickstart/templates` directory) which uses a parameterized query as the data source.

Each section shows the code to generate the report and any steps necessary to deploy.

### Q.12.7.1. Pass in Parameter values

The following code shows how to display an existing report template (in this case `PassInParameterValues.pak`), which uses a parameterized query. The parameter values are passed when creating the report.

```
// Connect to ERES Server

QbReport.setEspressManagerUsed(true);
QbReport.useServlet(true);
QbReport.setServletRunner(protocol + "://" + host + ":" + port);

QbReport.setServletContext(quadbase.common.client.ServerMessage.getServletContext());

// Set query parameters
Vector vec = new Vector();

vec.addElement("CA");
vec.addElement("NY");
```

```
Object queryParams[] = new Object[3];
queryParams[0] = vec;
queryParams[1] = new Date(101, 0, 4);
queryParams[2] = new Date(103, 01, 12);

// Set formula parameter
Object formulaParams[] = new Object[1];
formulaParams[0] = "Ivan";

// Create new Report object using specified report and
parameter values
QbReport report = new QbReport (null, "help/quickstart/templates/
PassInParameterValues.pak" ,
queryParams, formulaParams);
ByteArrayOutputStream data = new ByteArrayOutputStream(2048);

OutputStream out = new BufferedOutputStream(data);
report.export(QbReport.DHTML, out);
out.flush();
```

**Note**

The above piece of code is not complete. The code above is the core ERES Report API code needed to open a parameterized report, pass the parameter values to the report. Export it to DHTML and stream the DHTML content back to the client browser.

The class file for the above source is located in the <ERES installation directory>/WEB-INF/classes/help/quickstart directory.

When the jsp application is run by selecting the appropriate link from the main QuickStart API example page, the following report is shown:

http://quadbase:8080/ERES/help/quickstart/QuickStart.jsp?section=PassInParameterValues&lookupSe...

### Sales Report

Report Run By: **nan**  
Orders From 04, 2001 To 12, 2003

Total Orders	Units Ordered	Total Sales	Average Sale
16	526	\$584,240.00	\$13,510.00

#### Order Details

Order ID: 10001      Ordered by: Allied Furniture Emporium  
Order Date: 114, 2001      384 Broad St.  
   Littletown, NY 18322

Quantity	Product Name	Unit Price	Stained	Stain Price	Sub-Total
12	Ra Dresser	\$1,745.00	No	\$425.00	\$20,940.00
12	Enli Chair	\$450.00	Yes	\$27.00	\$5,724.00
14	Shimafya Chair	\$424.00	Yes	\$36.00	\$6,440.00
					<b>Order Total: \$33,104.00</b>

Order ID: 10002      Ordered by: Allied Furniture Emporium  
Order Date: 130, 2001      384 Broad St.  
   Littletown, NY 18322

Quantity	Product Name	Unit Price	Stained	Stain Price	Sub-Total
16	Set Dresser	\$1,645.00	No	\$427.00	\$26,320.00
21	Nisaba Chair	\$414.00	No	\$29.00	\$8,694.00
					<b>Order Total: \$35,014.00</b>

Order ID: 10004      Ordered by: Allied Furniture Emporium  
Order Date: 124, 2001      384 Broad St.  
   Littletown, NY 18322

Report Generated

The main part of the code is in the `getReport` method in the `PassInParameterValues` bean. There, a `QbReport` object called `report` is created using the `PassInParameterValues.pak` template. The parameter values are passed using the following constructor:

```
QbReport(Object parent, String reportTemplateName,
Object[] queryParameterValues, Object[] formulaParamterValues);
```

Both query parameter and formula parameter values are declared in the same order as the parameters were defined. Query parameters which take in multiple values are declared as a `Vector` object which then contains the different multiple values.

In addition to the above approach, you can simply pass in the name of the template and the parameter values, then the export format desired to the `LookupServlet` and let it do the work.

The following code shows how to pass in parameter values to an existing parameterized report template (in this case `PassInParameterValues.pak`) using the `LookupServlet`:

```
String contextPath =
quadbase.common.client.ServerMessage.getServletContext();

// Based on the ERES context, get the http location
of the files used in this example
int lastSlash = contextPath.lastIndexOf('/');
String eresPath = contextPath.substring(0, lastSlash);
String domain = protocol + "://" + host + ":" + port;

return domain + contextPath +
"/LookupServlet?USESESSION=TRUE&URLTYPE=FORREPORT&"
+ "TemplatePath=" + domain + eresPath + "/help/quickstart/templates/"
+ "PassInParameterValues.pak&MultiPageExport=false" +
"&QueryParamName=State&QueryParamSize=2&QueryParamValue=CA&QueryParamValue=NY"
+ "&QueryParamName=StartDate&QueryParamSize=1&QueryParamValue=2001-01-04"
+ "&QueryParamName=EndDate&QueryParamSize=1&QueryParamValue=2003-02-12"
```

```
&FormulaParamName=Name&FormulaParamValue=Ivan";
```



### Note

The above piece of code is not complete. The code above is the core ERES API code needed to pass an existing report template and the export format to the LookupServlet servlet.

The class file for the above source is located in the <ERES installation directory>/WEB-INF/classes/help/quickstart directory.

The main part of the code is in the `getReportUrl` method in the `PassInParameterValues` bean. There, the full path to the report template is created and passed to the `LookupServlet` servlet. The export format is also passed and the `LookupServlet` will then return the DHTML content.

Please note that if you are using the Access templates, you will need to change the following lines of code from:

```
queryParams[1] = new Date(101, 0, 4);
queryParams[2] = new Date(103, 01, 12);
```

to

```
queryParams[1] = new Timestamp(101, 0, 4, 0, 0, 0, 0);
queryParams[2] = new Timestamp(103, 01, 12, 0, 0, 0, 0);
```

## Q.12.7.2. Pass in Parameter values using getAllParameters

In addition to the above, you can also pass in parameter values using the `getAllParameters` method in `QbReport`. The following code shows how to display an existing report template (in this case `PassInParameterValues.pak`), which uses a parameterized query, in an application. The report is opened with backup data (to avoid the initial hit to the database) and the parameter values are set. The report is then refreshed with the data.

```
// Connect to ERES Server
QbReport.setEspressManagerUsed(true);
QbReport.useServlet(true);
QbReport.setServletRunner(protocol + "://" + host + ":" + port);
```

```
QbReport.setServletContext(quadbase.common.client.ServerMessage.getServletContext());
```

```
// Set query parameters
```

```
Vector vec = new Vector();
vec.addElement("CA");
vec.addElement("NY");
```

```
Object queryParams[] = new Object[3];
queryParams[0] = vec;
queryParams[1] = new Date(101, 0, 4);

queryParams[2] = new Date(103, 01, 12);
```

```
// Set formula parameter
```

```
Object formulaParams[] = new Object[1];
formulaParams[0] = "Ivan";
```

```
// Create new Report object using backup data
QbReport report = new QbReport (null,
    "help/quickstart/templates/PassInParameterValues.pak", false, false,
    false, true);
```

```
report.getAllParameters().get(0).setValues((Vector)queryParams[0]);
report.getAllParameters().get(1).setValue(queryParams[1]);
report.getAllParameters().get(2).setValue(queryParams[2]);
report.getAllParameters().get(3).setValue(formulaParams[0]);
```

```
report.refreshWithOriginalData();
```

```
ByteArrayOutputStream data = new ByteArrayOutputStream(2048);
```

```
OutputStream out = new BufferedOutputStream(data);
report.export(QbReport.DHTML, out);
out.flush();
```



## Note

The above piece of code is not complete. The code above is the core ERES Report API code needed to open a parameterized report and to pass the parameter values to the report. Export it to DHTML and stream the DHTML content back to the client browser.

The class file for the above source is located in the <ERES installation directory>/WEB-INF/classes/help/quickstart directory.

When the jsp application is run, by selecting the appropriate link from the main QuickStart API example page, the following report is shown:

**Sales Report**

Report Run By: Ivan  
Orders From 1/04, 2001 To 1/12, 2003

Total Orders	Units Ordered	Total Sales	Average Sale
16	526	\$584,240.00	\$13,910.48

**Order Details**

Order ID: 10001      Ordered by: Allied Furniture Emporium  
Order Date: 1/14, 2001      384 Broad St.  
   Littleton, NY 18322

Quantity	Product Name	Unit Price	Stained	Stain Price	Sub-Total
12	Ra Dresser	\$1,745.00	No	\$425.00	\$20,840.00
12	Enli Chair	\$450.00	Yes	\$27.00	\$5,724.00
14	Shimalya Chair	\$424.00	Yes	\$36.00	\$6,440.00
<b>Order Total:</b>					<b>\$33,104.00</b>

Order ID: 10002      Ordered by: Allied Furniture Emporium  
Order Date: 1/30, 2001      384 Broad St.  
   Littleton, NY 18322

Quantity	Product Name	Unit Price	Stained	Stain Price	Sub-Total
16	Set Dresser	\$1,645.00	No	\$427.00	\$26,320.00
21	Nisaba Chair	\$414.00	No	\$29.00	\$8,694.00
<b>Order Total:</b>					<b>\$35,014.00</b>

Order ID: 10004      Ordered by: Allied Furniture Emporium  
Order Date: 1/24, 2001      384 Broad St.  
   Littleton, NY 18322

*Report Generated*

The main part of the code is in the `getReport` method in the `QuickStart1072` bean. There, a `QbReport` object called `report` is created using the `PassInParameterValues.pak` template. The parameter values are passed using the following interface in `QbReport`:

```
getAllParameters()
```

Both query parameter and formula parameter values are declared in the same order as the parameters were defined. Query parameters which take in multiple values are declared as a `Vector` object which then contains the different multiple values.

Please note that if you are using the Access templates, you will need to change the following lines of code from:

```
queryParams[1] = new Date(99, 0, 4);
queryParams[2] = new Date(101, 01, 12);
```

to

```
queryParams[1] = new Timestamp(99, 0, 4, 0, 0, 0, 0);
queryParams[2] = new Timestamp(101, 01, 12, 0, 0, 0, 0);
```

### Q.12.7.3. Use `getParameterPage`(and `ParamReportGeneratorServlet`)

The following code shows how to display an existing report template (in this case `PassInParameterValues.pak`), which uses a parameterized query, in a servlet. The servlet creates the `QbReport` object by opening the template using back-up data. An HTML page is then streamed asking for the parameter values. These values are passed to another servlet (the `ParamReportGeneratorServlet` servlet) and the `QbReport` object is created, from the given template, with the specified parameter values.

```
// Connect to ERES Server

QbReport.setEspressManagerUsed(true);
QbReport.useServlet(true);
QbReport.setServletRunner(protocol + "://" + host + ":" + port);

QbReport.setServletContext(quadbase.common.client.ServerMessage.getServletContext());

String reportLocation =
"help/quickstart/templates/PassInParameterValues.pak";
// Create the ObReport object using back-up data

QbReport report = new QbReport(null, reportLocation, false, false, false,
true, false);

// Specify the parameters for connecting to
ParamReportGeneratorServlet
if (protocol.equalsIgnoreCase("https")) report.setHttpsDynamicExport(true,
host, port);
else report.setDynamicExport(true, host, port);
report.setServletDirectory
(quadbase.common.client.ServerMessage.getServletContext());
// Specify report template location and export format desired

ParameterPage paramPage =
report.getParameterPage(reportLocation, null, QbReport.DHTML, null);
return paramPage.toHtmlString();
```



## Note

The above piece of code is not complete. The code above is the core ERES Report API code to open a parameterized report template, create a dynamic HTML page that asks for the input parameter values and stream that HTML page back to the client.

The class file for the above source is located in the <ERES installation directory>/WEB-INF/classes/help/quickstart directory.

When the jsp application is run, by selecting the appropriate link from the main QuickStart API example page, the following HTML page is shown:

Enter Your Name:

Select States:

Start Date:

End Date:

*HTML Prompt Page Generated*

Depending on what parameter values you pass in, a report similar to the following will be shown:

**Sales Report**

Report Run By: man  
Orders From 1/04, 2001 To 11/12, 2003

Total Orders	Units Ordered	Total Sales	Average Sale
16	526	\$584,240.00	\$13,910.48

**Order Details**

Order ID: 10001      Ordered by: Allied Furniture Emporium  
Order Date: 1/4, 2001      384 Broad St.  
   Littleton, NY 18322

Quantity	Product Name	Unit Price	Stained	Stain Price	Sub-Total
12	Rs Dresser	\$1,745.00	No	\$425.00	\$20,940.00
12	Enil Chair	\$450.00	Yes	\$27.00	\$5,724.00
14	Shimalya Chair	\$424.00	Yes	\$36.00	\$6,440.00
					<b>Order Total: \$33,104.00</b>

Order ID: 10002      Ordered by: Allied Furniture Emporium  
Order Date: 1/30, 2001      384 Broad St.  
   Littleton, NY 18322

Quantity	Product Name	Unit Price	Stained	Stain Price	Sub-Total
16	Set Dresser	\$1,645.00	No	\$427.00	\$26,320.00
21	Nisabe Chair	\$414.00	No	\$29.00	\$8,694.00
					<b>Order Total: \$35,014.00</b>

Order ID: 10004      Ordered by: Allied Furniture Emporium  
Order Date: 1/24, 2001      384 Broad St.  
   Littleton, NY 18322

*Report Generated*

The main part of the code is in the `getReport` method in the `QuickStart1073` bean. There, a `QbReport` object called `report` is created using the `PassInParameterValues.pak` template. The dynamic export is then called to use the `ParamReportGeneratorServlet` (provided with ERES) using the following lines:

```
<QbReport object>.setDynamicExport(
    boolean isDynamicExport, String serverName, int servletRunnerPort);

String htmlParamPage = <QbReport object>.getHTMLParamPage(
    String reportLocation, int exportFormat);
```



### Note

If you are using HTTPS protocol, you have to call `setHttpsDynamicExport` method instead of `setDynamicExport`.

The HTML file asking for the parameter values is then generated and passed to the client browser.

In addition to the above approach, you can simply pass in the name of the template and the export format desired to the `LookupServlet` and let it do the work.

The following code shows how to pass in an existing parameterized report template (in this case `PassInParameterValues.pak`) and have a HTML page asking for the input parameter values be generated, using the `LookupServlet`:

```
String contextPath =
    quadbase.common.client.ServerMessage.getServletContext();

// Based on the ERES context, get the http location
of the files used in this example
int lastSlash = contextPath.lastIndexOf(/);
String eresPath = contextPath.substring(0, lastSlash);
String domain = protocol + "://" + host + ":" + port;

return domain + contextPath +
    "/LookupServlet?USESESSION=TRUE&URLTYPE=FORREPORT&" +
    "TemplatePath=" + domain + eresPath +
    "/help/quickstart/templates/QuickStart54.rpt&MultiPageExport=false" +
    "&ForHTMLParamPage=TRUE";
```



### Note

The above piece of code is not complete. The code above is the core ERES API code needed to pass an existing report template and the export format to the `LookupServlet` servlet.

The class file for the above source is located in the `<ERES installation directory>/WEB-INF/classes/help/quickstart` directory.

The main part of the code is in the `getReportUrl` method in the `QuickStart1073` bean. There, the full path to the report template is created and passed to the `LookupServlet` servlet. The export format is also passed and the `LookupServlet` will then return the DHTML content.

## Q.12.8. Deploy/Export Drill-Down

The following code shows how to display an existing report template (in this case `DeployExportDrillDown.pak`), which is a drill-down report, in a servlet. The servlet creates the `QbReport` object by opening the template.

A DHTML page, showing the contents of the first level, is then streamed. The next level report is obtained by clicking on the links. These links point to the `DrillDownReportServlet`. The values of the link (clicked on) are passed to the `DrillDownReportServlet` and the next level report (based on those values) is then generated and then streamed to the client.

```
// Connect to ERES Server
QbReport.setEspressManagerUsed(true);
QbReport.useServlet(true);
QbReport.setServletRunner(protocol + "://" + host + ":" + port);

QbReport.setServletContext(quadbase.common.client.ServerMessage.getServletContext());

//Specify the template location

String reportLocation = "help/quickstart/templates/
DeployExportDrillDown.pak";

// Create the QbReport object using the template
QbReport report = new QbReport (null, reportLocation);

// Specify the parameters for connecting to DrillDownReportServlet

report.setServletDirectory(quadbase.common.client.ServerMessage.getServletContext());
if (protocol.equalsIgnoreCase("https")) report.setHttpsDynamicExport(true,
host, port);

else report.setDynamicExport(true, host, port);

// Export the report
ByteArrayOutputStream data = new ByteArrayOutputStream(2048);
OutputStream out = new BufferedOutputStream(data);

report.export(QbReport.DHTML, out);
```

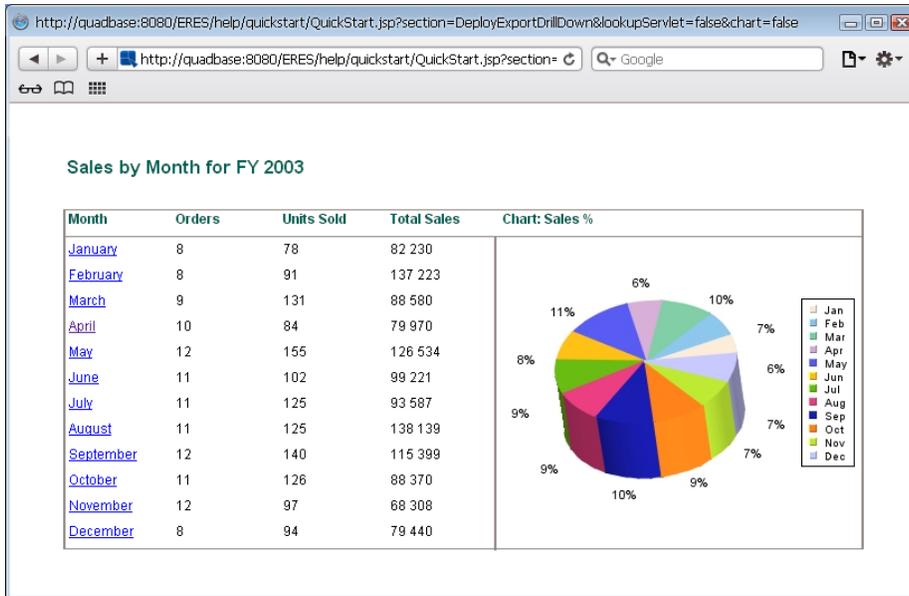


### Note

The above piece of code is not complete. The code above is the core ERES Report API code needed to open a drill-down report, export it to DHTML, and stream the DHTML content back to the client browser.

The class file for the above source is located in the <ERES installation directory>/WEB-INF/classes/help/quickstart directory.

When the jsp application is run by selecting the appropriate link from the main QuickStart API example page, the following report is shown:



*Report generated*

Depending on what link you click, a report similar to the following will be shown:

**Orders for April 2003**

Order ID	Customer	Quantity	Product Name	Sub-Total
10062	<a href="#">Southwest Style</a>	4	Asherat Dresser	8 008
		4	Anahita Dresser	7 948
<b>Order Total: 15 956</b>				
10063	<a href="#">Woodworks Furniture</a>	2	Anubis Table	3 374
		12	An Chair	5 100
		8	Enki Chair	3 400
<b>Order Total: 11 874</b>				
10064	<a href="#">Du Monde Furnishings</a>	6	Addad Dresser	15 006
		6	Set Dresser	12 432
<b>Order Total: 27 438</b>				
10065	<a href="#">Imports &amp; Leather Gallery</a>	18	Cula Chair	8 424
		6	Atun Table	8 196
		18	Shamash Chair	8 082

*Report Generated After Clicking on a Link*

Again, depending on what link you click, a report similar to the following will be shown:



```

return domain + contextPath +
"/LookupServlet?USESESSION=TRUE&URLTYPE=FORREPORT&" +
"TemplatePath=" + domain + eresPath +
"/help/quickstart/templates/
DeployExportDrillDown.pak&MultiPageExport=false";

```



### Note

The above piece of code is not complete. The code above is the core ERES API code needed to pass an existing report template and the export format to the LookupServlet servlet.

The class file for the above source is located in the <ERES installation directory>/WEB-INF/classes/help/quickstart directory.

The main part of the code is in the getReportUrl method in the DeployExportDrillDown bean. There, the full path to the report template is created and passed to the LookupServlet servlet. The export format is also passed and the LookupServlet will then return the DHTML content.

## Q.12.9. Launch Report Designer

The following code shows how to launch the Report Designer (in default mode) via the API:

```

String host = getParameter(host);

int port = Integer.parseInt(getParameter("port"));
String eresPath = getParameter("servletContext");

// Specify the connection information for Report Designer
to connect to ERES server.
QbReportDesigner.useServlet(true);
QbReportDesigner.setServletRunner(protocol + "://" + host + ":" + port);
QbReportDesigner.setServletContext(eresPath);

// Use toolbar icons in the JAR file
QbReportDesigner.setUseSysResourceImages(true);

// Create and display the Report Designer object
QbReportDesigner designer = new QbReportDesigner(this);
designer.setVisible(true);

```



### Note

The above piece of code is not complete. The code above is the core ERES Report API code needed to open the Report Designer.

The class file for the above source is located in the <ERES installation directory>/help/quickstart/classes directory.

When the jsp application is run, by selecting the appropriate link from the main QuickStart API example page, Report Designer (in default mode) will start.

The main part of the code is in the LaunchReportDesigner java code. There, a QbReportDesigner object called designer is created and shown:

```
QbReportDesigner(Object parent);  
<QbReportDesigner object>.setVisible(boolean b);
```

---

# Chapter 1. Administration

## 1.1. Overview & Architecture

Welcome to EspressoReport Enterprise Server (hereafter referred to as ERES). ERES is a powerful, server-based information presentation and delivery system. ERES provides a fully secure environment where users can create and deploy reports and charts.

With ERES, users can easily draw data from a variety of sources and create a wide range of charts and reports using the integrated Chart Designer and Report Designer interfaces. Ad-hoc reporting is supported through the thin-clients QuickDesigner Reports and QuickDesigner Charts. The Organizer interface provides a virtual file management and security system for reports and charts.

Deployment features allow users to automatically publish reports and charts through the integrated menu features, and by generating URL requests. ERES supports advanced scheduling and archiving features, allowing users to version reports, and schedule email, and batch printing jobs.

Advanced configuration and deployment is available through the Java API interfaces. Programmatically, users can get a handle to created reports and charts, and develop their own deployment vehicles for reports and charts, or integrate them within other application environments.

### 1.1.1. ERES Components

ERES is composed of 14 main components:

- |                               |  |
|-------------------------------|--|
| <b>ERES Server:</b>           | The server is the back-end to ERES. It deploys as a servlet collection within an application server or servlet container, and manages design and deployment for Reports and Charts. The server handles user authentication, scheduling, archiving, and provides the data access and file I/O for clients running Chart Designer and Report Designer.   |
| <b>Organizer:</b>             | The Organizer is a powerful graphical user interface that acts as a virtual file management system. It allows users to organize charts, reports, and other files in virtual folders, generate URLs deploying charts and reports, schedule/archive charts and reports, and set file-level privileges. It also provides users with an interface for managing data sources, and designing database queries. |
| <b>Report Designer:</b>       | The Report Designer is a graphical user interface, launched within the Organizer that allows users to create and customize reports. The simple drop and drag style interface and extensive editing/formatting capabilities, makes report design quick and easy.  |
| <b>Chart Designer:</b>        | The Chart Designer is a graphical user interface, launched within the Organizer that allows users to design and customize charts. Its point and click interface enables the swift and easy creation of charts with an almost infinite degree of flexibility.   |
| <b>QuickDesigner Reports:</b> | The QuickDesigner Reports is a thin-client ad-hoc reporting tool. It allows users to quickly design data view queries, to use existing data sources, and build reports. Limited formatting options allow users to customize reports and save them back to the Organizer.   |
| <b>QuickDesigner Charts:</b>  | The QuickDesigner Charts is a thin-client ad-hoc reporting tool. It allows users to quickly design data view queries, to use existing data sources, and build charts. Limited formatting options allow users to customize charts and save them back to the Organizer.  |
| <b>Dashboard Builder:</b>     | The Dashboard Builder is a thin-client tool that allows users to create custom dashboard presentations. The simple interface allows users to select charts, define parameters and drill-down using elements deployed in the Organizer.   |
| <b>Online Maps:</b>           | Online Maps is a thin-client tool that allows users to create online maps using OpenStreetMap or Google© maps. The easy to use interface allows users to   |

create elegant maps with parameters, titles, legends, and drilldowns. For Online Maps maps, reports and charts can be directly embedded into the tooltips.

**SVG Maps:**

SVG Maps is a thin-client tool that allows users to create SVG maps. The easy to use interface allows users to create SVG maps with thresholds, titles, legends, drilldowns, and tooltips. In SVG maps, regions can be color coded based on the user's own data.

**Report Viewer:**

Report Viewer is an applet that enables you to view and manipulate a report dynamically through a web browser. Within the Report Viewer, users can navigate from page to page in the report, export the report to a variety of static formats, and drill-down on report elements.

**Page Viewer:**

Page Viewer is an integrated applet like Report Viewer that uses page serving technology. With Page Viewer, pages in the report are only sent to the client when requested. This allows users to view/preview large reports.

**Chart Viewer:**

The Chart Viewer is a Java Web Start Application that allows charts to be viewed remotely. It enables viewers to rotate, resize, zoom and pan a chart. Viewers can click on individual data points to either recover the data associated with the data point, or link to another chart.

**Report API:**

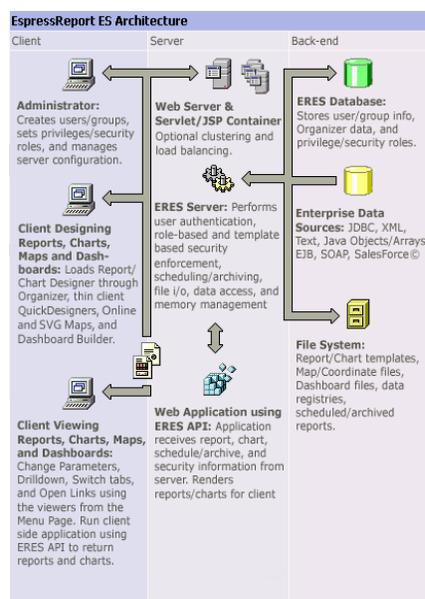
The Report API is an easy-to-use Java application programming interface that allows users to create, modify, and run reports and charts within applications, applets, servlets and JSPs. Users can easily get a handle to most chart or report elements making it easy to customize reports and charts on the fly.

**ERES API:**

The ERES API is a Java application programming interface that allows users to interface programmatically with the ERES Server and Organizer. Users can get a handle to the Organizer to retrieve information about files and permissions. Users can also get a handle to scheduled and archived reports.

## 1.1.2. ERES Architecture

The following diagram illustrates the basic architecture of an ERES implementation/installation:



*ERES Architecture Diagram*

The ERES server deploys as a servlet collection within an application server/servlet & JSP container. Other servlets/JSPs can be deployed in the server to connect to, and retrieve information from the ERES server (like file names, schedule information, and schedule reports/charts).

---

On the back end, information for ERES including user/group information, the Organizer structure and files, and security and privilege information are saved in a database. The server also performs data access and buffering for reports and charts, and file I/O on the server-side.

Clients can be administrators modifying users, privileges, and server settings from the remote Admin Console, or the Organizer. Clients can load the Report Designer and Chart Designer through the Organizer to create reports and charts, or load the thin-client QuickDesigners for ad-hoc reporting and analysis. Clients viewing reports and charts can access the menu, make URL calls to the ERES server, or connect to other applications to retrieve charts and reports.

## 1.2. New Features List V7.0

### 1.2.1. New EspressoReport ES Features

1. Unified data registry dialog in Quick Designers and Map Designers allows users to view query, query result, and test with different parameter values, so users can locate the desired query easily (Section 3.2 - Data in QuickDesigners and Maps).
2. Diagrams on start page of Quick Designers and Map Designers helps end-users to understand how to use the tools quickly (Section 5.2.3 - Start Online Maps).

#### 1.2.1.1. Dashboard Builder (Chapter 6 - Designing Dashboards)

1. Being able to change chart type in dashboard preview gives end-users more control on dashboard contents (Section 7.9.2 - Preview Options).
2. Directly opening report/chart/map designer from dashboard builder makes it much more convenient to edit existing components or add new components (Section 6.2.3 - Add Charts, Reports and Maps).
3. Besides original static dashboard, responsive dashboard is added as a new dashboard type. Responsive Dashboard optimizes utilization of the display without exceeding the display width preset by user. Therefore, it's useful for mobiles (Section 6.2.2 - Responsive Dashboard).
4. Refresh Template icon on each report/chart/map makes it easier to test dashboard with real data (Section 6.2.3.3 - Chart/Report/Map Toolbar).

#### 1.2.1.2. QuickDesigner Reports (Section 4.3 - QuickDesigner Reports)

1. Simplified UI includes only two steps: Select Datasource and Construct Report.
2. Constructing and formatting report in only one interface makes it intuitive and easy to see what you get and flexible to make changes (Section 4.3.4 - Format Report Elements).
3. Drag and drop to add report columns and order report column ease the work of report design (Section 4.3.4.1 - Add a Column, Section 4.3.4.5 - Column Order).
4. Right click on report column to do Aggregation/Format/Remove is straightforward (Section 4.3.6 - Aggregation/Group, Section 4.3.7 - Data Formating, Section 4.3.4.2 - Remove a Column).
5. 'Excel-like' search icon on each column header allows data filtering by column (Section 4.3.9 - Data Filtering).
6. Parameters pane on the right makes report testing easy (Section 4.3.5 - Parameter Setting).
7. Less items on Toolbar reduces confusion and saves time (Section 4.3.3 - Toolbar).
8. Collapse Sidebars allow more space for report design (Section 4.3.4.3 - Collapse Sidebars).

#### 1.2.1.3. QuickDesigner Charts (Section 4.4 - QuickDesigner Charts)

1. Simplified UI includes only two steps: Select Datasource and Construct Chart.

2. Setting pane and chart preview pane are on the same window, so all design steps can be done on one window (Section 4.4.3 - Data Mapping and Ordering).
3. Improved toolbar shows Chart-Specific Options based on current chart type and combines settings to reduce icons (Section 4.4.5.5 - Chart-Specific Options, Section 4.4.4 - Toolbar).
4. Parameters setting on Chart Preview pane makes chart testing easy.
5. Collapse Sidebar allows more space for chart design.

### 1.2.1.4. Online Maps (Section 5.2 - Online Maps)

1. Online Map is a newly designed Google Map. It is more straightforward to create a new map (Section 5.2.3 - Start Online Maps).
2. Displaying setting on left pane, instead of popup dialog, makes UI more user-friendly.
3. Save dialog shows Organizer node tree. It makes it easier to locate the desired organizer folder to save map file (Section 5.2.7 - Save Map).
4. Heatmap feature provides visualization of data intensity (Section 5.2.6.6 - Heatmap).
5. Being able to set parameter values in Map Designer makes testing easier.
6. More map types satisfies different needs (Section 5.2.6.5 - Map Options).
7. Users can draw various shapes instead of simple map points. The shapes are drawn in Coordinates Editor (Section 5.2.4.2 - Create Coordinates, Section 5.2.4.2.3 - Manually).

### 1.2.1.5. SVG Maps (Section 5.3 - SVG Maps)

1. Redesigned UI is more straightforward to create a new map.
2. Allow users to change SVG image after a map is created (Section 5.3.4.3.1 - Change SVG Image).

## 1.3. Installation and Configuration

There are many important installation and configuration options available in ERES, so it is important to refer to this section of documentation as you're starting out.

### 1.3.1. Installing ERES

There are four versions of the installation program, one for Windows 10/8/7/Vista/XP/2003/NT/2000, one for Solaris/Unix, one for Mac OS X, and a pure Java version.

**Windows 10/8/7/Vista/XP/2003/NT/2000:** To start the Windows installer, run the `installERES.exe` file and the installer will launch.

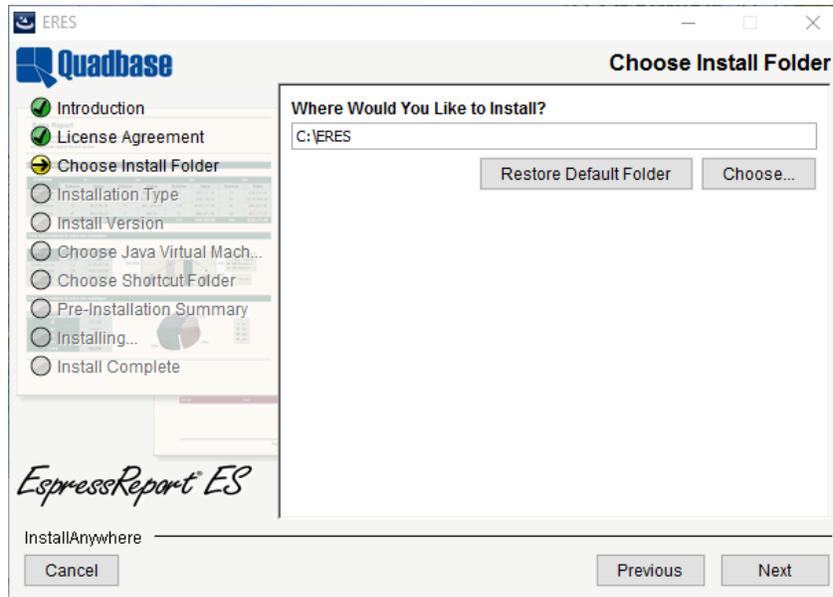
**Unix/Linux:** To start the Solaris/Unix installer, execute the `installERES.bin` file, and the installer will launch.

**Mac OS X:** To start the Mac installer, double click the `InstallERES.zip` file to extract the `InstallERES.app` file. Double-click on `InstallERES.app`, and the installer will launch.

**Pure Java:** To start the pure Java installer, a Java Virtual Machine, the equivalent of Java 1.8 or higher, must already be installed on the machine where ERES is to be installed. Make sure that the JVM is included in your path (or move the `installERES.jar` file to the same directory as the JVM). From a command prompt navigate to the directory where you have placed the `installERES.jar` file, and type the following command: `java -jar installERES.jar`. The installer will then launch.

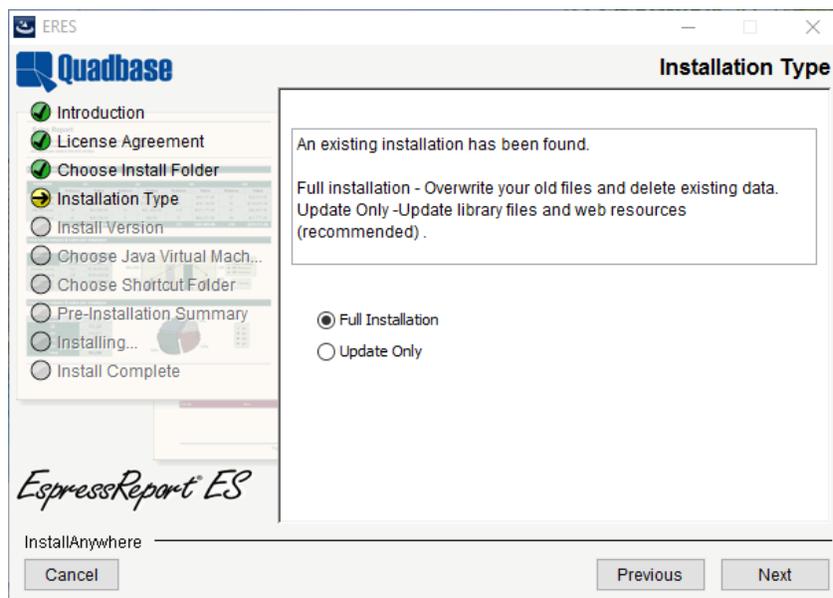
Each installer also comes with a console install mode where there is no graphical interface and the steps are performed through a terminal/console window. To launch any installer in console mode, add `-i console` to the end of the command. For example, for the Unix installer, you can execute the following command to run in silent mode: `./installERES.bin -i console`

Once the installation program has launched and you agree to the license agreement, the first option that is presented asks you to specify the directory into which you would like to install ERES.



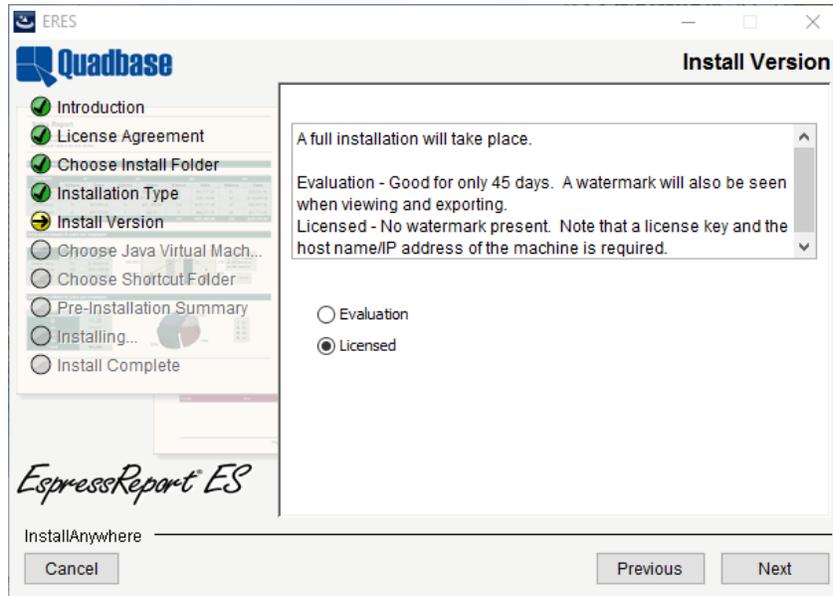
*Choose ERES Location Dialog*

If the directory already exists and contains an installation of ERES, the installer will give you the option to perform a full or update install. If the directory doesn't contain ERES, the installation will always be a full install. For more information about update installs please see Section 1.3.1.1 - Re-Install/Update.



*Install Type Dialog*

For full installs, the next step will ask you whether you would like to install the evaluation or release version. The evaluation version is fully functional. However, there will be a water mark on any report, chart, map, and dashboard and the license will expire in 45 days. For update installs, it will always use an evaluation license. Please visit <https://www.quadbase.com/register/> to get the full license.



*Install Version Dialog*

If you select to install a release version, the next dialog will prompt you to enter your license key and verify the host name of the machine on which you're installing ERES.



*Enter License Key Dialog*

After you have entered the information, the installer will attempt to register the license key with Quadbase. If the registration fails, you will be unable to continue installing the release version of ERES. You will have the option to continue installing the evaluation version. After the installation completes, you can register your key online at <https://www.quadbase.com/register/>, or contact Quadbase Sales for additional help.

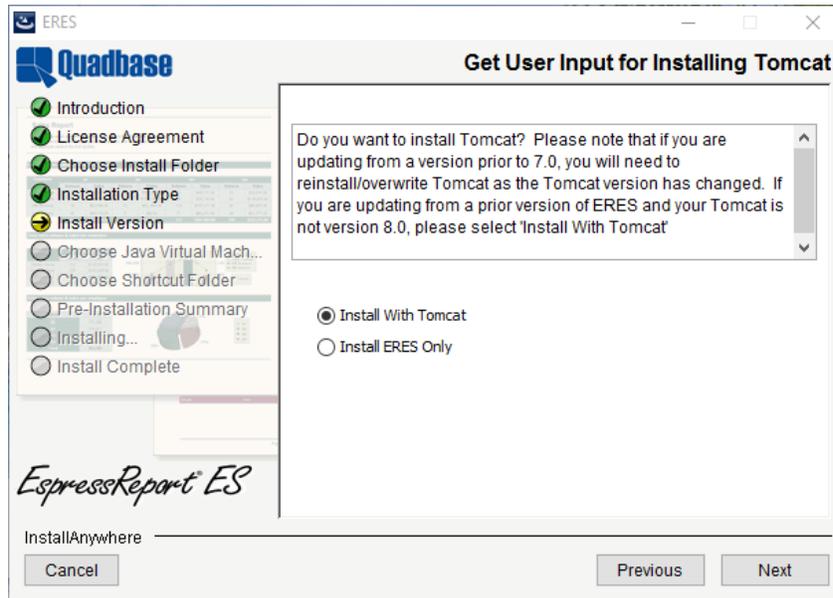


### Note

After the installation completes, the release version will only run for the hostname specified in this dialog, so double-check to make sure the host name is correct. You can also use the IP address of the machine if you prefer.

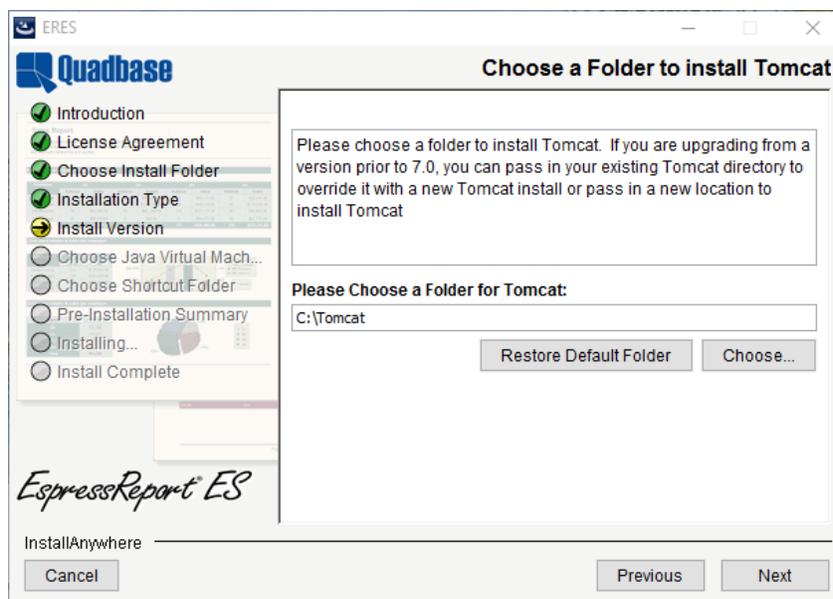
The next option that is presented is whether to install Tomcat with ERES or not. As described in the previous chapter, the ERES Server deploys as a servlet collection in an application server/servlet container. For out-of-the-box functionality, you can install the Apache Tomcat server with ERES already deployed. If you are installing ERES for the first time, or evaluating the product, it is recommended that you select this option.

If you select to install without Tomcat, the ERES files will be installed in your system and you will need to manually deploy it within your application server. Instructions for different server platforms are in Section 9.3 - Using Other Application Servers.



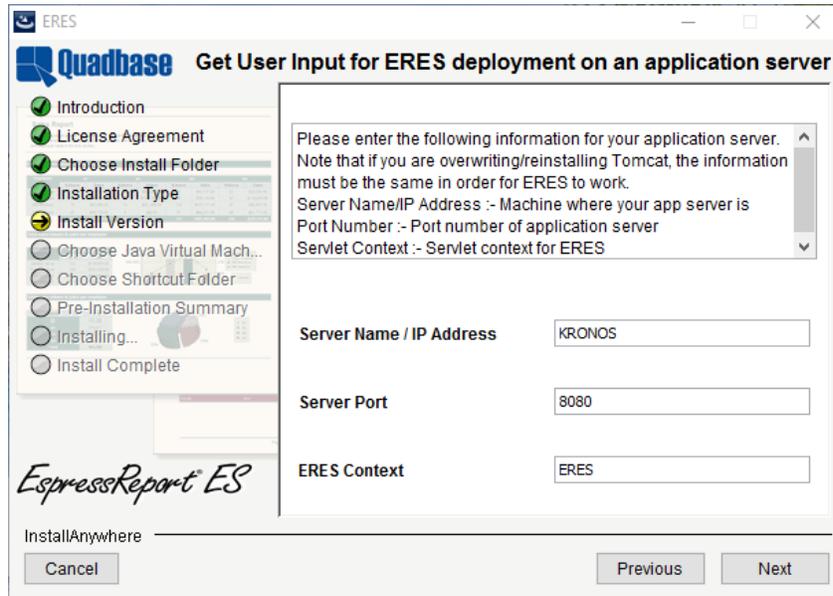
*Use Tomcat Dialog*

The next dialog prompts you to select the directory into which you would like to install Tomcat. If you select not to install Tomcat this option will be skipped. Tomcat and ERES will be installed in different directories on the system, and the ERES directory will be mapped as a virtual directory in Tomcat.



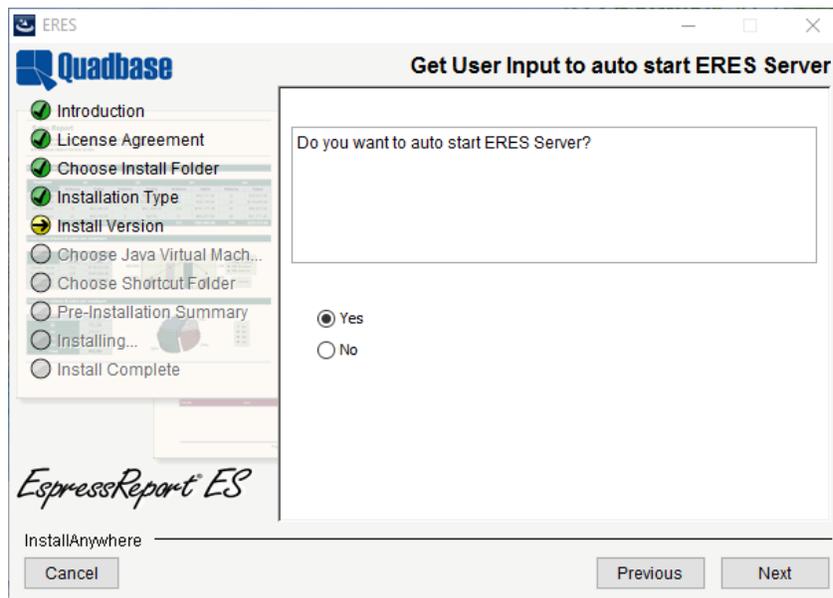
*Choose Tomcat Location Dialog*

The next dialog prompts you to specify the connection options for the application server. This dialog appears whether you choose to install ERES with Tomcat or not. You will need to specify the machine name or IP address, the port number, and the context for the application server on which ERES will be running.



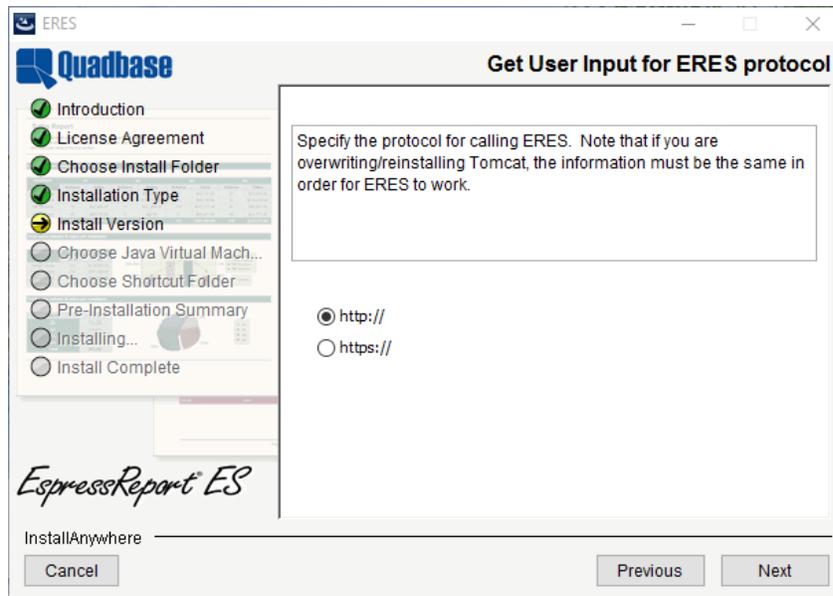
*Choose Tomcat Settings Dialog*

After specifying the connection details for the application server, the next dialog asks you to choose whether to auto start the ERES Server, i.e., have the ERES Server running whenever the application server is started.



*Autostart ERES Server*

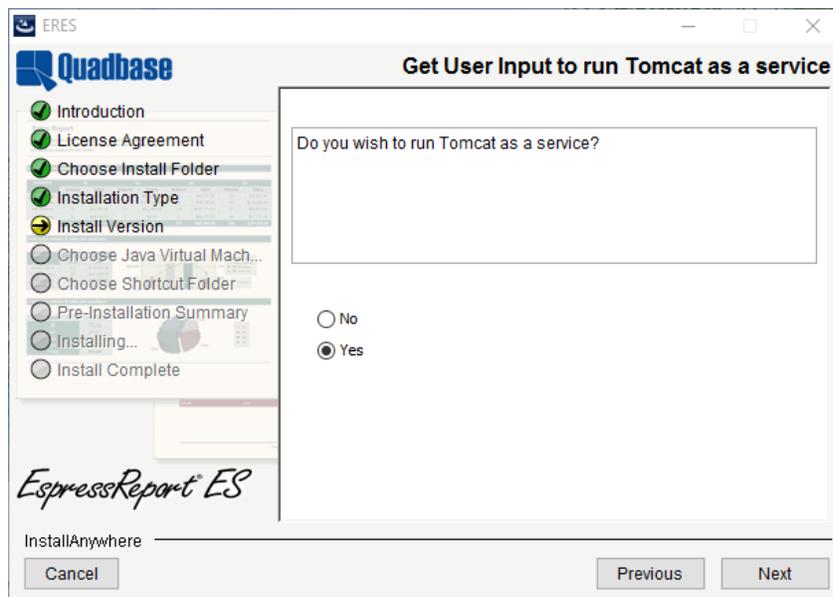
The next dialog asks you to choose the protocol you wish to use to access ERES, i.e., do you wish to use `http://` or `https://`?



*Specify Protocol for ERES*

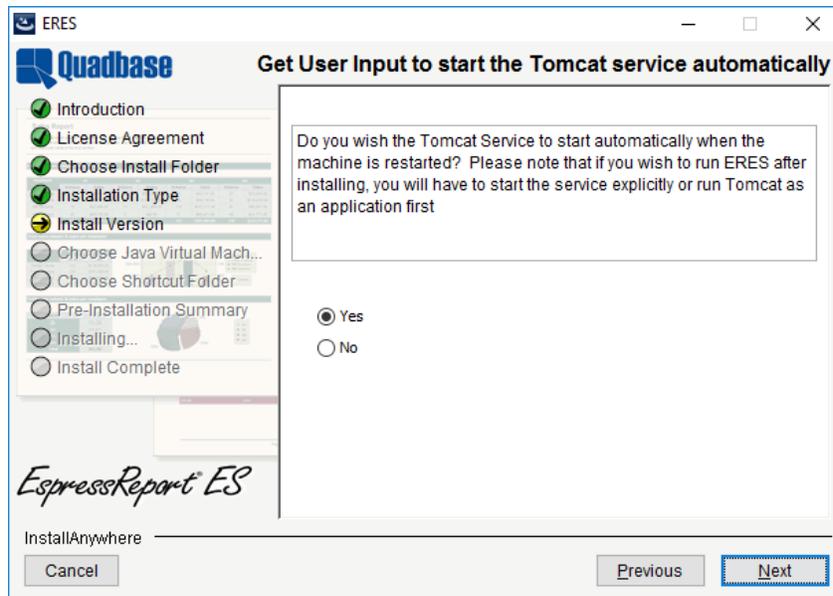
If you choose `https://` protocol and have selected to install Tomcat, the following two dialogs will appear. Note that if you have chosen not to install Tomcat, you are responsible for setting the https connection details and the certificate yourself.

The next dialog allows you to install Tomcat as a Windows service.



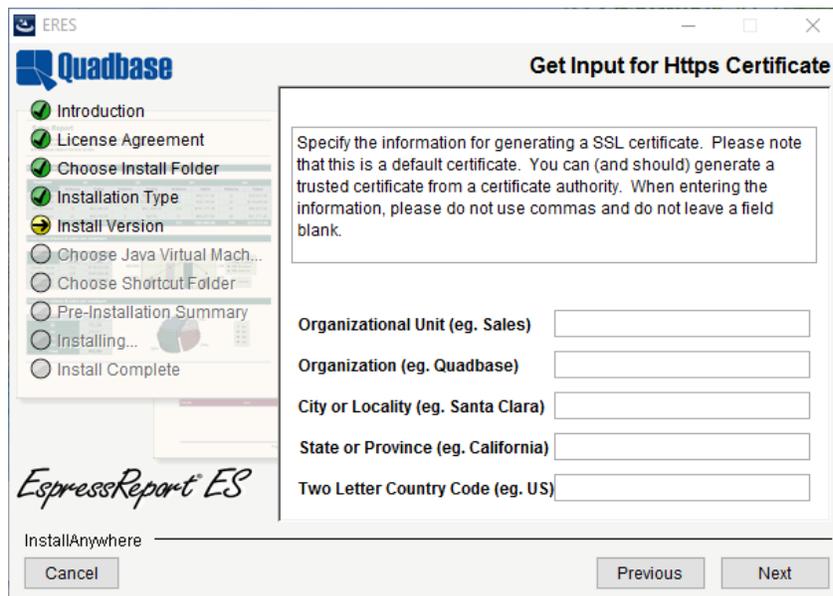
*Install Tomcat as a service*

If you choose to install Tomcat as a service, the next dialog will prompt you to select whether the service should start automatically when the machine is started.



*Start Tomcat service automatically*

The next dialog appears when the option to install Tomcat and the `https://` protocol is selected. This dialog obtains information to generate a generic SSL certificate for the installed Tomcat to use.



*Specify User and Organization Information*

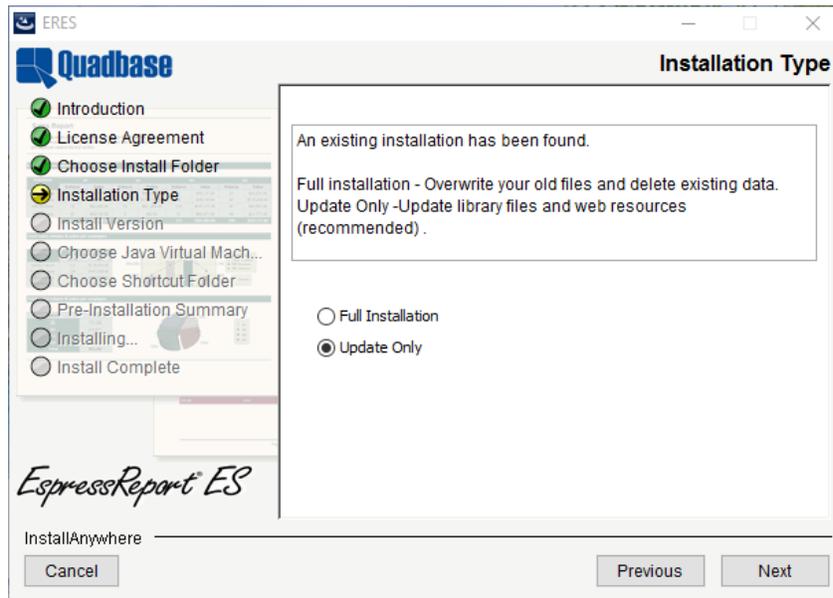
For this dialog, you cannot leave any information blank nor can you use a comma. If either of the two happens or if the two letter country code is not two characters in length, a error message is shown and you will be prompted to enter the information again.

The last option in the installer is only available for the Windows 10/8/7/Vista/XP/2003/NT/2000 installation. It allows you to specify where to create the program shortcuts in the Start Menu, on the Desktop, or both. Note that shortcuts will only be created if you select to install Tomcat.

After you complete the last option you will be shown a summary of the options you've selected. Next, the program will install.

### 1.3.1.1. Re-Install/Update

If you select a destination directory for the ERES files that already contains an ERES installation, you will be prompted as to whether you would like to update your existing installation, or create a new one.

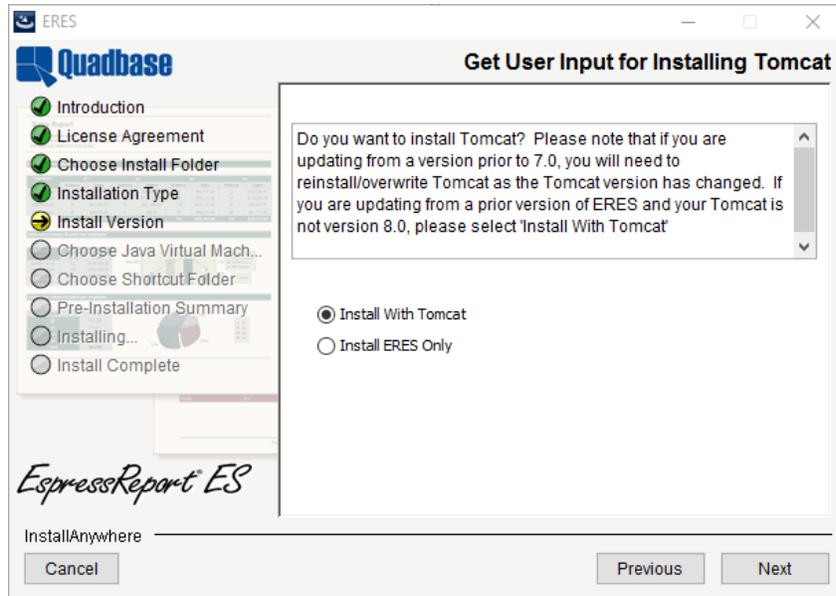


*Install Type Dialog*

If you select to create a new installation, then the entire existing installation will be overwritten (and all settings will be re-set to default values). If you select this option and click the *Next* button, the installation wizard will continue as if you were installing ERES for the first time (as described in the previous chapter).

If you select to update the current ERES installation, only program files will be updated (thus enabling new ERES features and fixing corrupted files), but all ERES settings and the ERES database will be kept as they are (i.e. with your current settings).

If you select to install an update and you are updating from a version prior to 7.0 (in other words: you are not re-installing an existing ERES 7.0 installation), you will be prompted whether you want to install ERES with bundled Tomcat 8.0. Please note that **ERES 7.0 requires Tomcat version 8.0 or higher**. ERES versions prior to 7.0 were distributed with Tomcat version 6 and lower, so if you are currently using ERES 6.6 or lower with bundled Tomcat, you will need to install ERES 7.0 with Tomcat, or update your Tomcat manually (not recommended). If you're just re-installing ERES 7.0, this step will be skipped, Tomcat will not be installed, and your deployment configuration will not be changed.



*Install Tomcat Dialog*

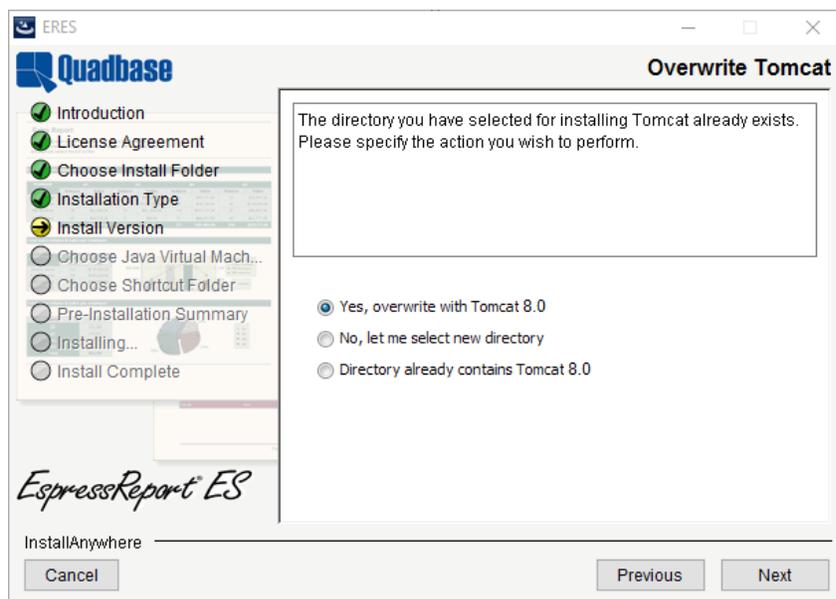
If you chose to install ERES with Tomcat 8.0, the next step will be to select a directory for Tomcat. If you choose a directory that doesn't contain an existing Tomcat installation and click *Next* (recommended), you will be taken straight to the Tomcat configuration wizard step (as described in the previous chapter).

If you choose a directory that already contains the Tomcat installation, the following dialog will appear giving you three options: to overwrite the existing Tomcat installation with a new one, to go back to the previous step, or to tell ERES to assume that the directory already contains Tomcat version 8.0 or higher.



### Tip

If you want to use an existing Tomcat 8.0 installation for running ERES, select the existing Tomcat installation directory and then select the *Directory already contains Tomcat 8.0* option.



*Install Version Dialog*

If you are upgrading from a previous version of ERES, make sure to follow the steps in Section 1.3.2.1.3 - Upgrading ERES Database from previous version of ERES to update your existing ERES database.

## 1.3.2. Configuration

After you have completed the installation of ERES there are several configuration options available.

### 1.3.2.1. The ERES Database

Information about users and groups, files in the Organizer, schedule/archive jobs, and security/privilege information is stored in a database. By default this is the HSQL Java application database which is included in the ERES installation. Connection to this database is transparent to users and will work out of the box. This database works fine for development or evaluation environments, however, it is generally insufficient for production environments, as it will not scale to large deployments and provides no failure/recovery features.

#### 1.3.2.1.1. Using a Different Database

ERES provides users the option of using a different database than the HSQL database provided with the installation. To run with a different database, you will need to create the tables used by ERES in your database. Create table scripts are provided in the ERES installation in the data directory, and are available for most major databases.

The database connection that the ERES server uses to connect to the database is specified in the Admin Console, under Setting Info → ERES Repository. You can change the connection information to provide a JDBC connection to the database that you would like to use. The JDBC driver for the database you're using will need to be added/available to the classpath of the application server/servlet container where you have deployed ERES. You can also make a connection to the ERES database using JNDI. You can pass in the JNDI connection details after pointing your application server to the ERES database.

Specific setup instructions for different databases are available in Section 9.1 - Using Other Databases.

#### 1.3.2.1.2. Running HSQL in Client-Server Mode

Normally the HSQL database runs as an application process on the server-side and needs no user interaction to start or stop the database process. HSQL can also be run in client-server mode. This mode can improve performance and scalability for the database when run in a multi-user environment. To run the HSQL database in client-server mode, open a console window and navigate to the /data/ directory of your ERES installation. In this directory run the following command:

```
java -classpath "../WEB-INF/lib/hsqldb.jar" org.hsqldb.Server -database
quadbasedb -port 2857 (or whatever port you would like the database to
listen on)
```

The database server process will then start. To configure the ERES Server to connect to HSQL running in this mode, log on as Admin and enter the Admin Console → Setting Info → ERES Repository and change the entry in Database URL to read `jdbc:hsqldb:hsql://machinename:port` where `machinename` is the name or IP address of the server, and the port number is the port you selected for the HSQL server.

#### 1.3.2.1.3. Upgrading ERES Database from previous version of ERES

In order to support new features, the ERES database structure generally needs to be modified.

If you are running the ERES database on HSQL, MySQL, Oracle, MS-SQL, Informix or Postgres database, the database will be updated automatically during the ERES installation process.

If you are using a different database, you need to run the upgrade program before you can start using the latest version of ERES. The upgrade programs are located in the `<ERES-INSTALL>/data/dbupgrade` directory.

Currently, the upgrade programs support MS-ACCESS and DB2 databases (besides the six databases that are also supported by the ERES installer). If your database is not in the above list, please contact our technical support staff for more details.

There are two versions of upgrade programs, namely, DBUpgradeGUI and UpgradeAppl.

The former is the graphic user interface version, the latter the command line version.

Please note that ERES log records are not preserved during the upgrade process.



### Warning

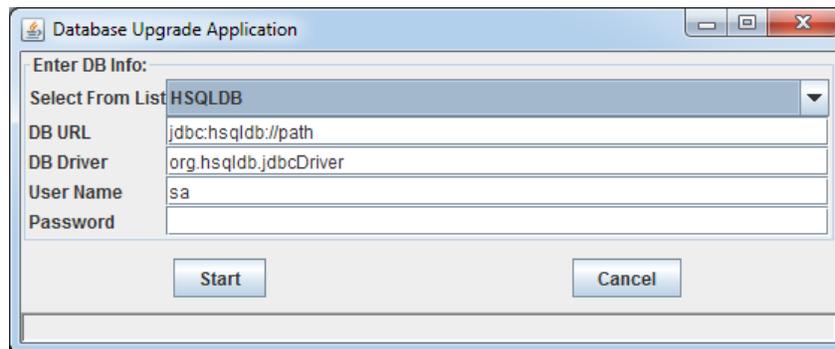
As a precaution, you are recommended to backup your ERES database before you proceed with the upgrade. This is to safeguard the unlikely event that any abnormal situation may occur during the upgrade process.

## Upgrade User Repository Database Using Graphic Interface

1. Change directory to <ERES-INSTALL>/data/dbupgrade/
2. Enter command:

```
java -classpath "<DB_Driver>;DBUpgrade.jar;." quadbase.DBUpgradeGUI
```

The upgrade program graphic interface is shown below:



*Upgrade DB Program GUI*

You need to specify the database information so that the program can proceed to do the upgrade.

Please note that the default database connection information is provided only as a guideline, you need to fill in the exact connection information for the upgrade program to run. Also, please make sure that the database driver is included in the `-classpath` flag in the command.

For example, if you are using Oracle, you may run the command with classpath as follows:

```
java -classpath "c:\ERES\WEB-INF\lib\jdbc_oracle.jar;DBUpgrade.jar;."
quadbase.DBUpgradeGUI
```

Press the *Start* button and the upgrade process will begin.

If you entered wrong database information, the program will prompt you with an error message complaining wrong database connection settings.

After the upgrade program is finished, the status bar at the bottom of the window will show *Done*.

If you click the *Cancel* button during the upgrade process, it will terminate the upgrade program. But it will NOT roll back to the original database. In such a case, you have to use your own backup data or use the backup SQL file generated by this program to restore your database.

### Upgrading ERES Database Using Command Line:

To upgrade using the command line version, simply type the command as follows:

```
java -classpath "<DB_Driver>;DBUpgrade.jar;." quadbase.UpgradeAppl
<ERES_config_file>
```

The program will use the file specified by <ERES\_config\_file> to obtain connection information. The config file should be written in the same format as the config.txt in <ERES\_INSTALL>/data/dpupgrade/.

### 1.3.2.2. Starting the Server

If you have selected the option to *Auto-Start Server*, then the ERES Server is started the moment the application server (Tomcat by default) is started. If the option was not selected, or you change the option in the Administration Console → Server Options → Auto-Start, then you will have to start the ERES Server manually.

The first step in running and administering ERES is to start the server. To start the ERES server, you will first need to start the application server/servlet container in which the server is deployed. If you installed ERES with Tomcat, go to the /bin/ directory of the Tomcat installation and execute startup.bat/.sh to start Tomcat.

Next, load the start page for ERES. This is the index.jsp page in the ERES install directory. If you installed ERES with Tomcat then the virtual directory mapped to the ERES install directory is /ERES/ so to reach the index page, you would use the following URL:

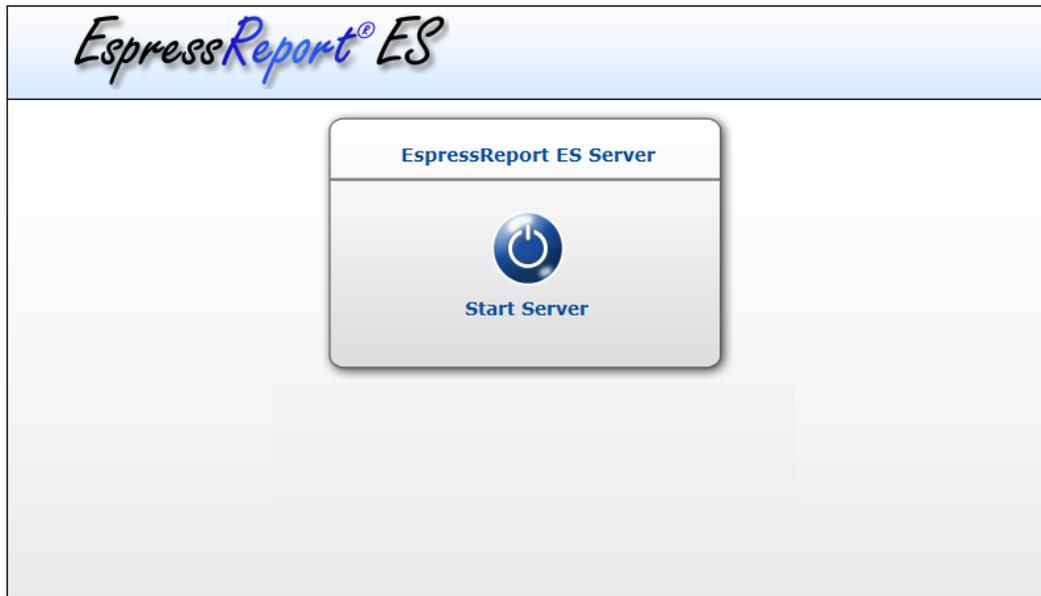
```
http://machinename:port/ERES/index.jsp
```



#### Note

You must have cookies and javascript enabled in your browser in order to use the ERES interfaces.

There are two versions of the ERES start page. If you performed a clean ERES install (i.e. to an empty directory), the following start page will be displayed:



*New ERES Start Page*

If you don't like the new ERES start page design, you can use the old design. This can be done in the admin console. See the Section 1.4.1.2 - Setting Info for more details.

If you updated from a previous ERES version, the ERES start page will be the same as before. The default ERES start page looked like this in previous versions:

*Old ERES Start Page*

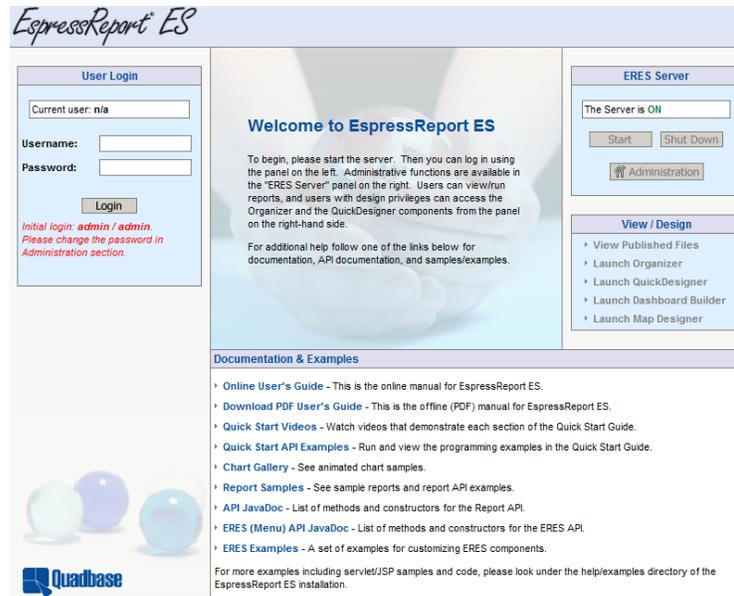
You can start the server directly in the start page. If the “Autostart” feature has been enabled (see Section 1.3.1 - Installing ERES for more details), the ERES server should be already running. If it's not, click the *Start Server* button. Once the server is started, you can log-in to access other EspressoReport ES functions (the default administrator login is username: **admin** password: **admin**).



## Note

Any user can start the server, but only the administrator can shut down the server. Even with the server running, the *Shut Down* button will only become active when you login as the system administrator.

*ERES server is running, Administrator is logged in*



Old ERES start page, ERES server is running

### 1.3.2.3. Increasing maximum memory heap size for applets

All applets have a maximum memory heap size of 16 megs, by default. In Windows, this can be increased by going to Control Panel and selecting Java (or Java Plugin). Click on the *Java* tab and then *View* under *Java Applet Runtime Settings*. Enter `-Xmx256M` under *Java Runtime Parameters* and click *OK*.

### 1.3.3. Backward compatibility patches

If you upgraded from an older version, you may notice some changes in the default behavior. Although the backward compatibility is kept as much as possible, sometimes a new behavior is preferred. The new behavior should be better for most users, but if you already have some charts or reports from an older version, you may want to keep the old behavior, so your charts and reports look exactly the same as before. That is why we provide backward compatibility patches.



#### Caution

Patches are for advanced users only. Please apply them only if you need them and if you know what you are doing. If you are not sure, please contact support.

The patches can be found in `<ERES_Installation_Directory>/lib/Patches` directory. They are stored in JAR archives. To apply a patch, you only need to add the appropriate JAR file to the classpath of your application as described below.

If you want to apply a patch to all designers (Report Designer, Chart Designer, Dashboard Builder, etc.) and viewers (Report Viewer, Chart Viewer, Dashboard Viewer, etc.), you have to copy the appropriate patch JAR file to `<ERES_INSTALLATION_DIRECTORY>/WEB-INF/lib/` directory. Then you have to edit `ERESOrganizer.bat` file (if you use Windows), or `ERESOrganizer.sh` file (if you use other OS) in your ERES installation directory and add relative path to the patch JAR file (e.g. `/lib/Patches/patch1.jar`) to the classpath parameter. Then you need to edit `ERESOrganizer.jnlp` file in the ERES installation directory and add relative path to the patch JAR file to the archive attribute of the applet tag.

If you are using API, you have to include the patch JAR file in the classpath of your application.

Below is a list of all available patches in the current version.

- patch1.jar** - turn off chart axis padding by default
- default behavior (without the patch) - axis padding is on by default
  - behavior with the patch - axis padding is off by default

- new behavior has been introduced in version 4.0
  - this feature can also be set using the `IAxis.setAxisPaddingAdded` method in API or in the Axis Scale dialog in the Chart Designer
- patch2.jar**
- add left margin for annotation text in charts
  - default behavior (without the patch) - annotation text does not have left margin
  - behavior with the patch - annotation text has left margin
  - new behavior has been introduced in version 5.0
  - this feature cannot be set by public API nor UI
  - annotation text is legend text, chart titles and any text inserted using Insert → Text in the Chart Designer
- patch3.jar**
- use 0 to 1 as min/max value when chart axis autoscale for axis pt less than 1
  - default behavior (without the patch) - maximum and minimum are always set according to data if autoscale is used
  - behavior with the patch - if max-min is < 1 and autoscale is used, min is set to 0 and max is set to 1
  - new behavior has been introduced in version 5.4
  - this feature cannot be set by public API nor UI
  - **It is not recommended to use this patch. The original behavior is a bug.**
- patch4.jar**
- turn off new pie chart label placement algorithm (calculate label placement based on pie sector position)
  - default behavior (without the patch) - new pie label placement algorithm is used
  - behavior with the patch - old pie label placement algorithm is used
  - new behavior has been introduced in version 6.0
  - this feature cannot be set by public API nor UI
- patch5.jar**
- always use integer value for chart axis auto scale
  - default behavior (without the patch) - integer is always used for axis auto scale
  - behavior with the patch - axis value data type is used for axis auto scale
  - new behavior has been introduced in version 6.0
  - this feature cannot be set by public API nor UI
- patch6.jar**
- disable minimum and maximum error check for chart axis scale
  - default behavior (without the patch) - the error check is enabled
  - behavior with the patch - the error check is disabled (so you can set maximum value lower than the one set in your dataset - same for minimum)
  - new behavior has been introduced in version 6.2
  - patch is for API only

- this feature cannot be set by public API nor UI
- patch7.jar** - turn off Single color for categories feature for columnar and bar charts by default
- default behavior (without the patch) - Single color for categories feature is turned on by default
  - behavior with the patch - Single color for categories feature is turned off by default
  - new behavior has been introduced in version 6.3
  - this feature can also be set using the `IDataPointSet.setSingleColorForCategories` method in API or in the Chart Options dialog in the Chart Designer
- patch8.jar** - line chart end to end revert single point data to display on left axis
- patch9.jar** - display stack label despite not having enough space in the stack to render it

### 1.3.4. Run Applets As JNLP

Trying to run Java Applets in newer browsers and/or with newer Java versions can lead to problems with Applets being deprecated or not supported any more. However, it is still possible to run Applets in a similar use case as before (open a Java application from a remote server via your browser) thanks to Java WebStart technology that's supported by Java 8.

To run a Java Applet via a Java Web Start JNLP file, you have to specify the `applet-desc` parameter in the JNLP file configuration file.

```
<applet-desc
  name="Chart Viewer"
  main-class="quadbase.chartviewer.Viewer"
  width="800"
  height="600">
<param name="filename" value="help/examples/ChartAPI/data/test.tpl"/>
<param name="preventSelfDestruct" value="true"/>
</applet-desc>
```

The `param` tags specify parameters for the applet. The parameters are different for each applet. The specific parameters are mentioned in the documentation chapter for each applet.

JNLP files can be either run locally when the required libraries (containing the compiled source code) are loaded as a local file or remotely when the libraries are loaded via HTTP/HTTPS.

When running JNLP files remotely via HTTP/HTTPS, the parameters `comm_protocol`, `comm_url` and `servlet_context` need to be added to the `applet-desc` element. You can either fill in the values manually (as the following code example shows) or you can have the values filled automatically in a JSP file.

```
<applet-desc
  name="Espress Report Designer"
  main-class="quadbase.reportdesigner.designer.ReportClient"
  width="380"
  height="160">
<param name="comm_protocol" value="servlet"/>
<param name="comm_url" value="http://127.0.0.1:8080"/>
<param name="servlet_context" value="Espress70/servlet"/>
</applet-desc>
```

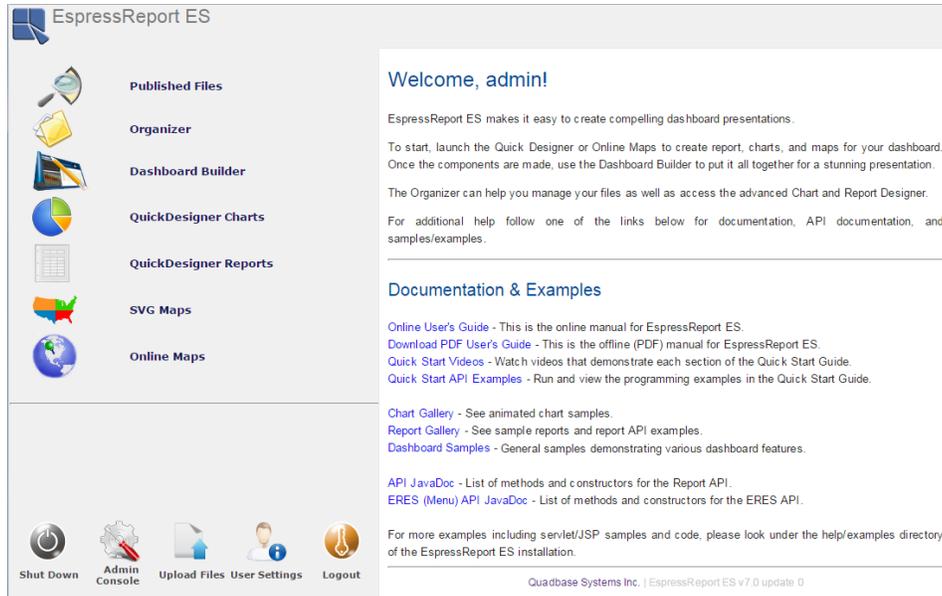


#### Note

If you want to pre-fill the values automatically in a JSP file, the actual values highly depend on your server configuration. The previous example is just illustrative.

## 1.4. Administration

Major administrative functions in ERES are handled in the Admin Console. The admin console is a thin-client interface that can be launched from the ERES start page. To start the Admin Console, start the server (see Section 1.3.2.2 - Starting the Server), and login as the administrator. The administrator username is **admin** and the default password is **admin**. This can be changed in the console.

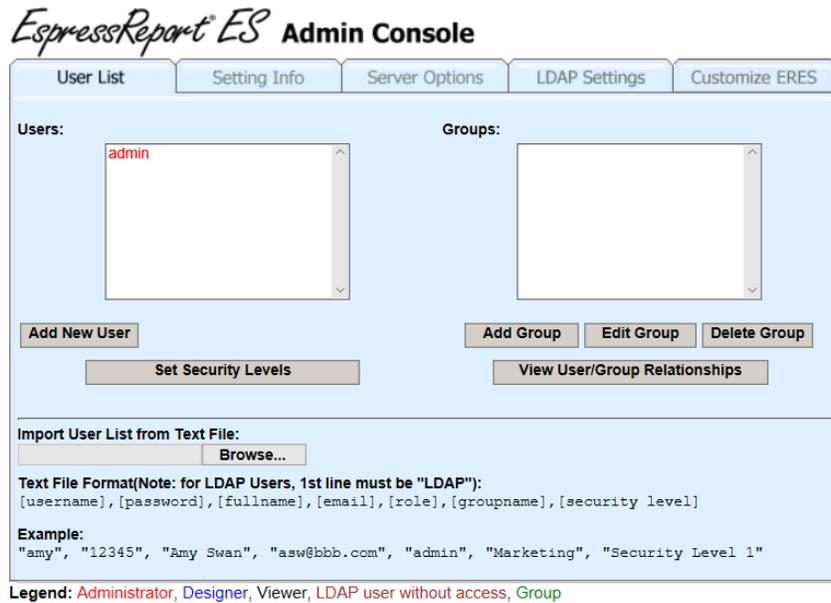


*Admin Logged into Start Page*

Once you login the administration button in the start page becomes active. Click this button to load the admin console. You can also access the admin console directly by going to the menu page login: `Admin_Login.jsp`. If you installed ERES with Tomcat the URL to the login is `http://machinename:port/ERES/Admin_Login.jsp`. This will bring up a page allowing you to login to the Admin Console directly.

### 1.4.1. The Console Interface

The top of the Admin Console interface contains five tabs. One is labeled *User List*, one *Setting Info*, one *Server Options*, one *LDAP Settings*, and one *Customization*.



Admin Console

### 1.4.1.1. User List

The *User List* tab of the Admin Console allows you to create and maintain users and groups that use ERES to design or view reports and charts. The left-hand side of the dialog contains a list of all the defined users, and the right-hand side a list of all the defined groups. Normally these users and groups are stored in the ERES database, however ERES can be configured to retrieve the users and groups from existing systems including LDAP servers. To add a new user, click the *Add New User* button. This will bring up a new dialog allowing you to specify information for the new user.

Add User Dialog

For each user, you need to supply a username, an email address (for schedule delivery), and a password. Each user also has a primary role of a viewer or a designer which is assigned in this dialog. Viewers can see and run reports as well as build ad hoc reports using the QuickDesigner, but they do not have access to the core design/development tools, like Report Designer, Chart Designer, and Organizer. Designers have access to all the interfaces, but are restricted by licensing and can't change server settings and user privileges. Each designer user must be licensed by

a developer seat. Administrator can view all the interfaces, change server settings, view detailed logs and modify user privileges. Each administrator must be licensed by a developer seat.

In the user list, you can quickly distinguish user roles by their color. There is a legend for that below the admin console.

Once you have finished adding information for the new user, click *Ok* and the new user will be added.

You can edit or remove a user by selecting it in the user list, and clicking either *Edit User* or *Delete User(s)* button below the user list. If you select to edit the user, a dialog will open allowing you to change the information for that user. If you select to delete the user, it will be removed. The original administrator user (*admin*) can't be removed.



## Note

You can't edit users that have been read from LDAP (see the Section 1.4.1.4 - LDAP Settings chapter to learn more about LDAP users). If you select a LDAP user and click the *Edit* button, the following dialog will open.

A screenshot of a Windows-style dialog box titled "Edit User". The dialog has a light blue background and a white border. At the top, it says "User Details from LDAP:". Below this, it displays "Username: patrik" and a red warning message: "Full name and email address are read from the LDAP and cannot be edited." There are three input fields: "Full Name:", "Email Address:", and "Primary Role:". The "Primary Role:" section has three radio buttons: "Administrator", "Designer" (which is selected), and "Viewer". At the bottom, there are "Ok" and "Cancel" buttons.

*Edit LDAP user dialog*

You are allowed to edit only the user attributes, that have not been read from LDAP. The rest of the user settings can be edited on the LDAP server.

Also, LDAP user accounts can't be fully deleted. If you select a user that has been read from an LDAP server and click the *Delete User(s)* button, the user will be deactivated, but it will not be removed from the user list - it will be marked as *LDAP user without access*.

To add a new group, click the *Add Group* button under the group list. This will pop-up a new dialog prompting you to specify information for the group and select members.

*Add Group Dialog*

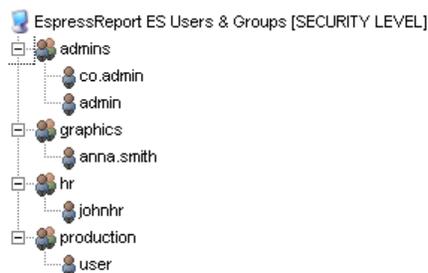
For each group, you need to name the group, and select members. Members can be individual users or other groups. To add group members, select users and/or groups from the left-hand side, and click the *Add* button. You can make multiple selections (or de-selections) by holding down the **CTRL** key when clicking on the user/group.

Once you have finished adding members to the group, click the *Ok* button and the group will be added.

You can edit or remove a group by selecting it in the group list, and clicking either *Edit Group* or *Delete Group* button below the group list. If you select to edit the group, a dialog will open allowing you to change the information for the group. If you select to delete the group, it will be removed.

There is also a button to set security levels for users and groups. Details for the security level dialog can be found in Section 1.4.1.1.3 - Setting User/Group Security Levels.

You can view all of the user and group relationships by clicking the *View User/Group Relationships* button. This will bring up a new dialog that shows all of the users and which groups they belong to in a tree format. If the user or group has a certain security level assigned to them, you will also see the name of the security level in brackets.



*User/Group Relationships Dialog*

#### 1.4.1.1.1. Importing Users and Groups

In addition to adding users and groups via the Admin Console manually, you can directly import users from a delimited file. To import users, you must have a CSV file with the following structure:

```
[username],[password],[fullname],[email],[role],[groupname],[security level]
```

`username` is the user login name, `password` is the user password, `fullname` is the user's full name, `email` is the user's email address, and `role` indicates the user's primary role either `designer`, `viewer` or `admin`.

The groupname field indicates which group the user belongs to. If that group has already been defined then the user will be added to the group. If the group has not been defined a new group will be created. A sample import file might look something like this:

```
"amy", "12345", "Amy Swansen", "aswansen@quadbase.com", "admin",
"Marketing", "Security Level 1"
"tom", "12345", "Tom Weeks", "tweeks@quadbase.com", "designer",
"Executive", "Security Level 2"
"sarah", "12345", "Sarah Jensen", "sjensen@quadbase.com", "viewer",
"Marketing", "Security Level 1"
```

In the case of importing user list for LDAP users, add one line with only "LDAP" at the beginning of the user-list text file, then all the contents of the password column will be ignored, and the password of the individual user will be obtained from LDAP Active Directory at run time.

If you have a convenient delimited file with users that you want to import into the ERES, log in as an administrator and then open the Admin Console. Make sure you are on the *User List* tab.

In the *Import User List from Text File* section, there will be a *Choose File* or *Browse* button allowing you to import the delimited file. Click on the button.

Legend: Administrator, Designer, Viewer, LDAP user without access, Group

*User List tab of the Admin Console*



## Note

The *Choose File* button may be different in other web browsers (the screen shot was taken in Safari), but it will always do the same thing - if you click the *Choose File* (or *Browse*) button, a browse dialog will pop up allowing you to select a file.

Select the delimited file and confirm the browse dialog. The following message should show up confirming that the users, groups and security levels were imported successfully.

Import of 3 new users was successful!

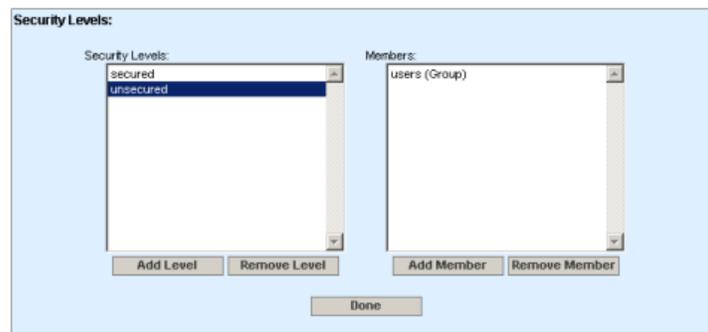
OK

### 1.4.1.1.2. Using Existing Users/Groups

Instead of creating new users and groups in ERES, you can re-use existing users and groups. ERES allows you to retrieve users and groups from an existing database, integrate with an LDAP server, or implement your own logic/methods to supply ERES with users and groups. The security provider interface provides an API that allows you to implement any custom logic to return user's, groups, and user/group relationships. For more information about configuring ERES to use existing users and groups, please see Section 9.2 - Integrating Existing Users/Groups. For more information about LDAP integration, please see Section 1.4.1.4 - LDAP Settings.

### 1.4.1.1.3. Setting User/Group Security Levels

On the User List page, there is also a button to *Set Security Levels*. When you click this button, a dialog will pop up allowing you to create, remove, and configure security levels within ERES. For more information regarding security levels please see Section 2.3.3 - Security Levels.



*Set Security Levels Dialog*

To add a new security level, click the *Add Level* button and provide a level name. You can add any number of users and groups into each security level. To do so, click on the *security level name* in the list box and click the *Add Member* button. This will present a dialog allowing you to select existing members and groups and add it to the security level. You can also remove members and security levels by selecting the item and using the *Remove Member* and *Remove Level* buttons respectively. Click *Done* when you are finished.

Please note that all security level changes will take effect immediately.

### 1.4.1.2. Setting Info

The *Setting Info* tab contains configurations and settings critical to the way ERES functions.

*Setting Info Dialog*

The first category, *System Settings*, presents the server information.

**ERES Server Name/IP Address:** This is the host name or IP address of the machine on which ERES is installed. This is generally configured during installation, but can be changed here if necessary.

**Web Root:** This is the web root for the server in which you've deployed ERES. If you installed ERES with Tomcat, this should be configured automatically. However, if you use a different database than the default HSQLDB, this field will need to be reconfigured. For tomcat, the default web root is `/webapps-ROOT/`.

**ERES Servlet Context:** This is the context in which the servlet collection for the ERES server, and deployment options are deployed. If you installed ERES with Tomcat, the default context is `ERES/servlet`. If you change this option, you will need to re-start the server for it to take effect.

**Google Map Key:** This is the key used for designing and viewing Google Maps. Please note that this must be changed to a key mapped for your machine, otherwise Google Maps will not appear. Please see Section 5.2 - Online Maps for Google Map details.

**Protocol:** This tells the system whether to use HTTPS with SSL or regular HTTP to connect to the sever.

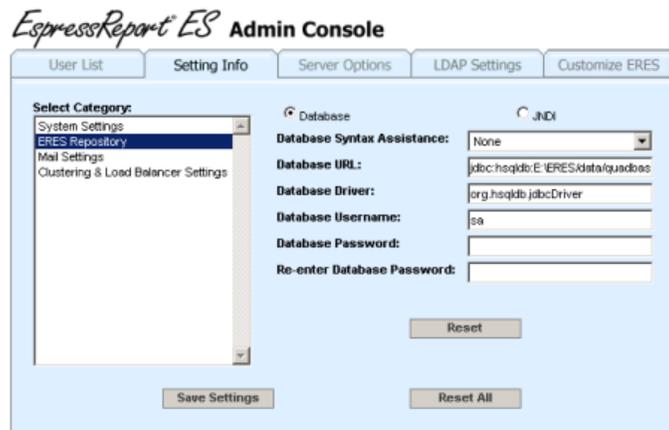
**Use Old Quick Designer:** Brings back the old ERES QuickDesigner (from version 6.6). You will see the QuickDesigner link on the ERES main page after restarting the ERES server. For information about QuickDesigner see Introduction to QuickDesigner [ <http://data.quadbase.com/Docs/eres/help/manual/IntroductionToQuickDesigner.html> ] from ERES 6.6 documentation.

Please note that changing protocol here is not enough to setup the system to work with SSL. You also have to configure your application server and update URLs of the existing files in Organizer. The recommended way is to re-install your product and use the HTTPS option in the installer that sets up everything automatically (see Section 1.3.1 - Installing ERES for more details). It is not enough to do upgrade install, you have to perform full installation. Be aware that full installation overwrites internal ERES database, so you loose all the files inserted in Organizer. Please back them up before making the full installation.

If you cannot or do not want to make the full installation, you can still configure SSL manually. To do that, please follow these steps:

1. Change the protocol in the Admin Console as described above.
2. Get (or generate) SSL certificate, install it on your application server (following instructions of your application server) and set up your application server to use HTTPS.
3. Start your application server and access ERES start page using the HTTPS (if you changed port number, do not forget to use the new one).
4. Start Organizer and update your URL Mapping to use the new protocol (and new port if it has changed). See Section 2.1.5 - URL Mapping for more details about URL Mapping.
5. Run Repair Broken Links to update URL Mapping for all files in Organizer. See Section 2.1.6 - Repairing Broken Links for more details about this feature.
6. If you use the `ERESHome.html` shortcut to access ERES start page, please also update the protocol (and port) in your `<ERES_INSTALLATION_DIRECTORY>/ERESHome.html`.

If you are switching from HTTPS to HTTP, use similar instructions. Just replace HTTPS with HTTP and skip generating the SSL certificate.



*Database Repository*

The second category ERES Repository describes the current database configuration options. You can use either a database connection or through JNDI. For database, you need to specify the following information.

- Database URL:** The JDBC URL to the database.
- Database Driver:** The JDBC Driver to the database.
- Database Username:** The username used to connect to the database.
- Database Password:** The password used to connect to the database. This field needs to be reentered in the next line to ensure validity.

You can use the Database Syntax Assistance drop down box to help in set up the connection by setting up a template of the URL and Driver.

Note that if you make a mistake while entering the database connection details, it may cause ERES not to run properly. In that event, the only way to correct it would be to be open `<ERES Install Directory>/WEB-INF/classes/QB.properties` file and edit the entries for DatabaseUrl, DatabaseDriver, DatabaseUserName and DatabasePassword. Depending on how the file was generated, the entries may not be grouped together.

The following is an image of the JNDI connection options. Not all components need to be completed in order to connect to the JNDI server.

The screenshot shows the 'ExpressReport ES Admin Console' with the 'Setting Info' tab selected. Under 'Select Category', 'ERES Repository' is chosen. The 'JNDI' section includes three input fields: 'JNDI Initial Context Factory', 'JNDI Provider URL', and 'JNDI Name'. A 'Reset' button is located below these fields. At the bottom of the console, there are 'Save Settings' and 'Reset All' buttons.

*JNDI Repository*

**JNDI Initial Context Factory:** The JNDI initial context factory specifies the class used to connect to the JNDI server. If you wish to use the application server's default context factory, you can leave this field blank.

**JNDI Provider URL:** The URL to the JNDI Provider. If the provider URL is already set up in the application server environment, you can leave this field blank.

**JNDI Name:** The name for the JNDI connection. Frequently, the name contains the prefix `java:comp/env/`. This field is required.

Mail Settings allow you to specify the SMTP information that ERES will use when sending scheduled e-mails to users.

The screenshot shows the 'ExpressReport ES Admin Console' with the 'Setting Info' tab selected. Under 'Select Category', 'Mail Settings' is chosen. The 'Mail Settings' section includes: 'SMTP Host' (text input with 'smtp.someserver.com'), 'Use SSL' (checkbox checked), 'Secured SMTP' (checkbox checked), 'SMTP Port' (text input with '25'), 'SMTP Username' (text input with 'username'), and 'Re-enter SMTP Password' (password input with masked characters). A 'Reset' button is located below these fields. At the bottom of the console, there are 'Save Settings' and 'Reset All' buttons.

*Mail settings*

**SMTP Host:** This is the host or ip address of the SMTP server.

**Use SSL:** Enable this if the SMTP server uses SSL/TLS.

**Secured SMTP:** Enable this if username and password are required to send mail from this server.

**SMTP Port:** This is the port for the secured SMTP server.

**SMTP Username:** Specify the username to use to log into the server.

**SMTP Password:** Specify the password for the email account. Enter again on the next line to verify.

Clustering & Load Balancer Settings are used when you want to run ERES in a Clustered environment.

The screenshot shows the 'ExpressReport ES Admin Console' interface. At the top, there are five tabs: 'User List', 'Setting Info', 'Server Options', 'LDAP Settings', and 'Customize ERES'. The 'Setting Info' tab is active. On the left, a 'Select Category:' dropdown menu is open, showing options: 'System Settings', 'ERES Repository', 'Mail Settings', and 'Clustering & Load Balancer Settings' (which is selected). The main area displays 'Load Balancer/Clustering Settings' with the following fields:

- Server Host:** 192.168.0.8
- Server Port Number:** 8080
- Cluster Member List:** 192.168.0.8:8080, 192.168.0.10:8080, 192.168.0.45:8080
- Member Host:** 192.168.0.45
- Member Port Number:** 8080

At the bottom of the form, there are three buttons: 'Save Settings', 'Reset', and 'Reset All'.

#### Clustering & Load Balancing

**Server Host:** This is the machinename or ip address of the load balancer.

**Server Port Number:** This is the port number for the load balancer.

**Cluster Member List:** This is the machinename or ip address of the cluster members. The first entry should always be the load balancer.

**Member Host:** This is the machinename or ip address of current node in the cluster.

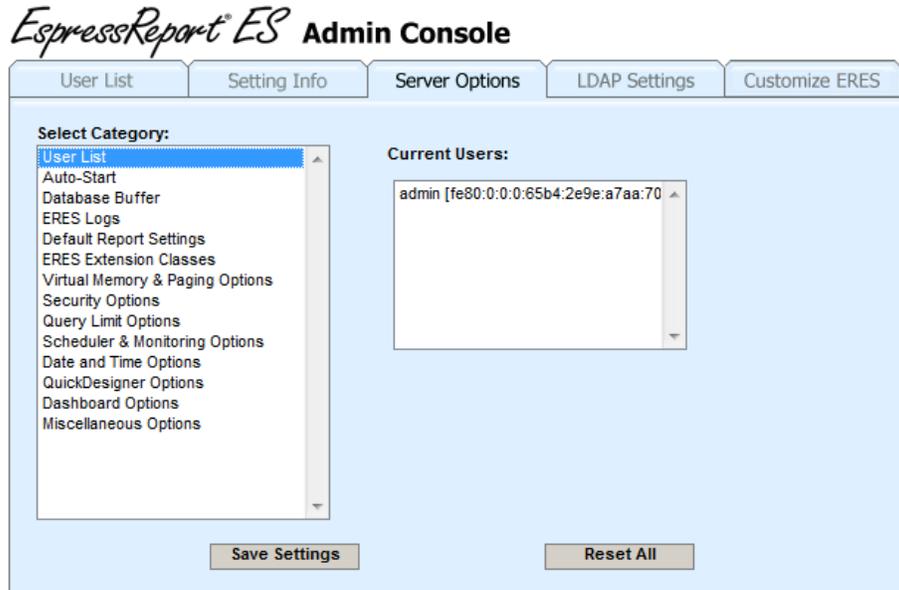
**Member Port Number:** This is the port number for current node in the cluster.

### 1.4.1.3. Server Options

The *Server Options* tab contains a number of server configuration options.

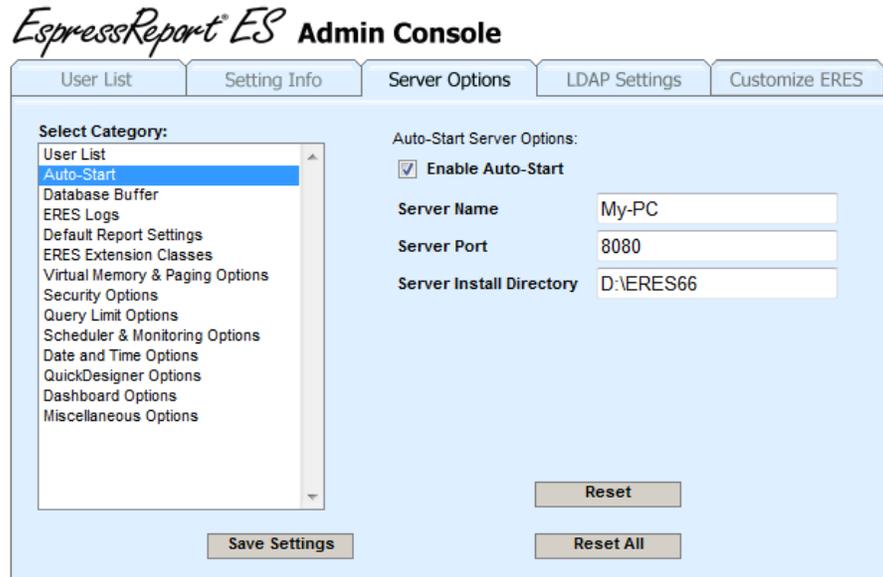
The following configuration categories are available from this tab:

**User List:** This category shows you which users are currently logged into ERES, either viewing or creating reports or charts.



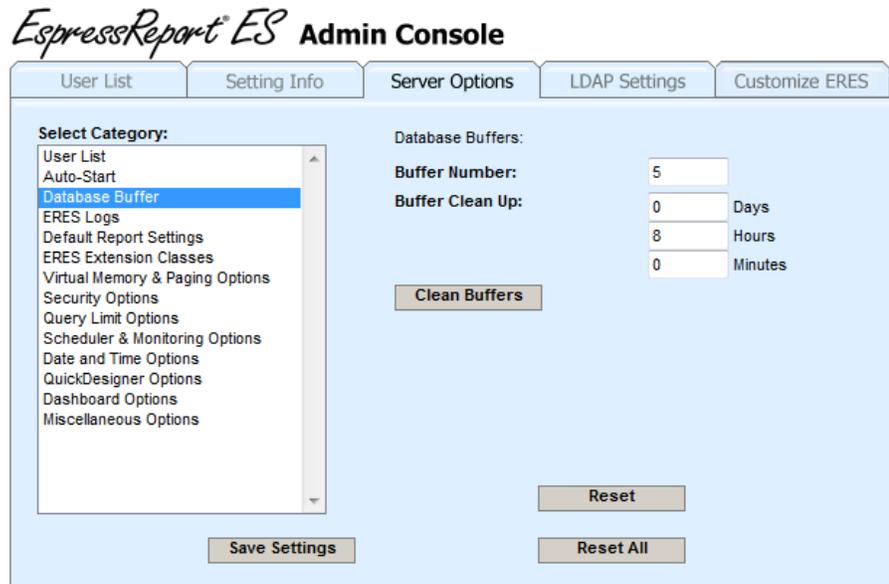
User list Dialog

**Auto-Start:** These options allow you to specify whether the server should be started automatically and what configuration should be used.



Auto-Start Options

**Database Buffer:** These options allow you to set the database connection buffering feature. If a database is being used as the data source for a report or chart, you can choose to buffer both the database connection and the data used for the report or chart using this feature. The number of connections and queries that will be stored, depends on the number of buffers from 1 to 999. The number of buffers is set in the *Buffer Number* dialog.



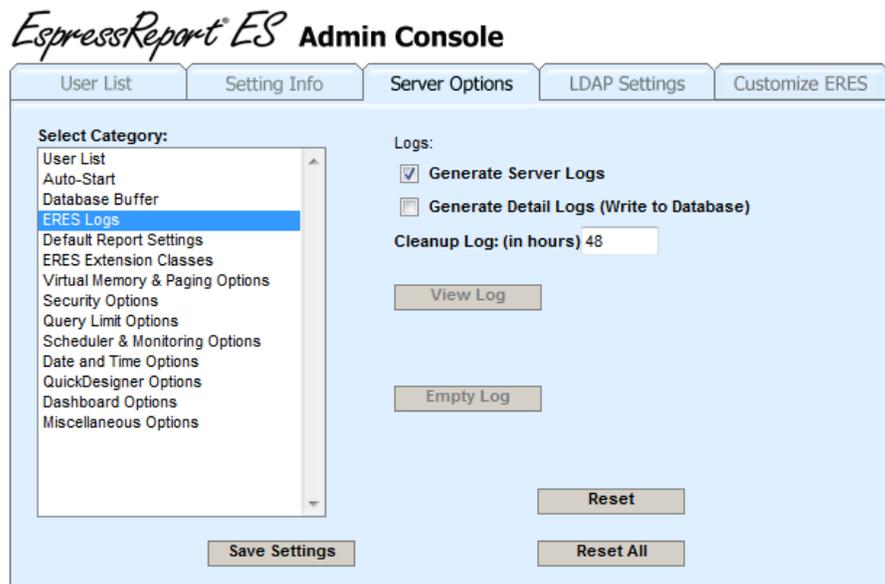
*Database Buffer Options*

**Buffer Number:** The number of buffers to use ranging from 1 to 999.

**Buffer Cleanup:** You can also specify the clean-up interval for database buffers in the Server Monitor. This interval indicates how often the buffers will be flushed, and the queries re-executed when reports/charts are run. You can specify the interval in days, hours, and minutes, by specifying the values in the appropriate dialogs.

**Clean Buffers:** You can immediately clean the database buffers by clicking the *Clean Buffers* button.

**ERES Logs:** ERES provides two levels of logging activities.



*ERES Logs Options*

**Generate Server Logs:**

The Generate Server Logs option allows you to turn on/off the ERES Server logs. When this option is enabled,

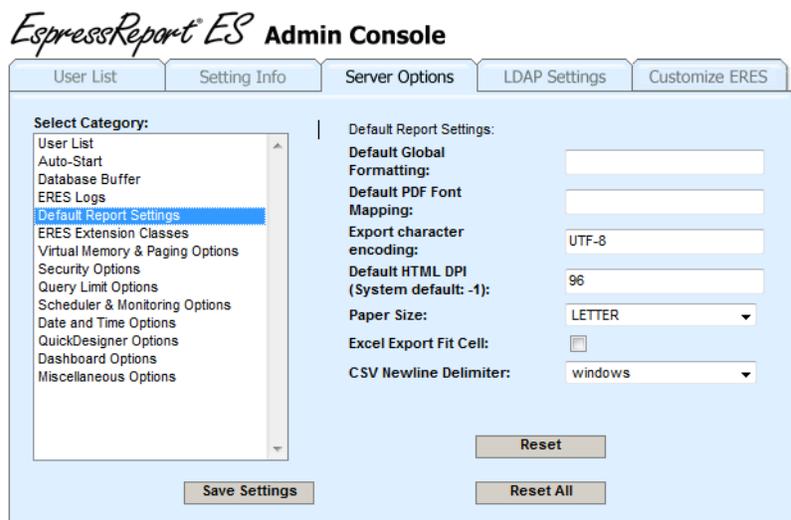
ERES writes a log file on the server-side that monitor the ERES installation. ERESserver.log monitors all of the client-server communication for the system, as well as the report generation. Any errors or exceptions are printed.

**Generate Detail Logs (Write to Database):**

The Generate Detail Logs (Write to Database) option creates an in depth report of all activities related to ERES. A full list of logged activities are described in Appendix 1.A - ERES Details Log. To view this log, click on the *View Log* icon. This will pop up a report of the recent ERES activities. You can adjust the refresh rate of this report by entering a number in minutes for the refresh interval. By default, the report will maintain the last 48 hours of events, but you can adjust this duration by changing the Cleanup value.

**Default Report Settings:**

These options allow you to specify global formatting, and font mapping XML files that will apply by default to new reports.



*Report Setting Options*

**Default Global Formatting:**

The global formatting properties will be applied any time a user elects to create a blank report in Report Designer or QuickDesigner.

**Default PDF Font Mapping:**

The font mappings will be applied by default anytime the report is exported to PDF. For more information about PDF font mapping, see Section 4.2.6.3.1 - PDF Font Mapping.

**Export character encoding:**

Character encoding that will be used for created reports. Default is *UTF-8*.

**Default HTML DPI (System default: -1):**

This argument allows you to set the screen resolution that should be used when reports are exported to various format. By default, the system resolution is used, however this can cause some discrepancies when reports are generated on a Linux or Unix server and viewed on a Windows client.

t. Generally in these scenarios, setting the DPI to 96 will produce a consistent export. Setting -1 will use the system default resolution.

**Paper Size:**

This argument allows you to set the default paper size. Available options are LETTER or A4

**Excel Export Fit Cell:**

This argument allows you to fit numeric values into single cells when exporting your report to Excel from the Menu Page, DHTML Viewer, or Dashboard. By default, this option is unchecked.

**CSV Newline Delimiter:**

This option allow you to set a default delimiter for all CSV exports.

**ERES Server Extension Classes:**

These options allow you to specify classes that invoke the server extension options for ERES.

*EspressReport ES* Admin Console

User List   Setting Info   **Server Options**   LDAP Settings   Customize ERES

Select Category:

- User List
- Auto-Start
- Database Buffer
- ERES Logs
- Default Report Settings
- ERES Extension Classes**
- Virtual Memory & Paging Options
- Security Options
- Query Limit Options
- Scheduler & Monitoring Options
- Date and Time Options
- QuickDesigner Options
- Dashboard Options
- Miscellaneous Options

ERES Server Extension Classes:

ERES Listener Class:

ERES Login Listener Class:

Scheduler Callback Class:

User Security Provider Class:

Reset

Save Settings   Reset All

*ERES Server Extension Options*

**ERES Listener Class:**

The ERES Listener Class allows you to implement listeners for server-driven report and chart execution. This includes reports and charts run from the Menu Page repository or via Dashboard URL calls. With these listeners in place, reports and charts will be passed to the custom code prior to final execution. For more information about the ERES Listener, see Section 7.5 - Menu Page Listener.

**ERES Login Listener Class:**

The ERES Login Listener Class allows you to modify the database users connect to when they run reports and charts in the dashboard. For more information please see Section 8.3.4 - Login Listener.

**The Scheduler Callback:**

The Scheduler Callback class allows you to provide a class that will be called when a schedule job completes. The class is called when the export/transmission of the report is successful, and if the schedule job fails. For more information about the scheduler callback, please see Section 8.7.5.1 - Getting Details of a Failed Schedule

**User Security Provider Class:**

The User Security Provider class allows you to specify the class which implements the methods to return your users, groups, and security relationships for use with ERES. For more information about the security provider interface, see Section 9.2.2 - Implementing UserSecurityProvider. Note that you do not have to setup the security provider if you are connecting to an LDAP server.

**Virtual Memory/Paging Options:**

For reports with large result sets, the paging feature reduces the amount of memory needed by using a virtual memory paging system. This system will write the data to temporary files on the hard drive once the specified memory usage is reached. You can specify the parameters for the paging feature using the options listed below. Please note that the older record set options (**Max Records in Memory**, **Max Characters Per Field**, and **Record Paging Buffer Size**) are no longer available.

*EspressReport ES* Admin Console

User List	Setting Info	Server Options	LDAP Settings	Customize ERES
<b>Select Category:</b> User List Auto-Start Database Buffer ERES Logs Default Report Settings ERES Extension Classes <b>Virtual Memory &amp; Paging Options</b> Security Options Query Limit Options Scheduler & Monitoring Options Date and Time Options QuickDesigner Options Dashboard Options Miscellaneous Options		<b>Virtual Memory/Paging Options:</b> <b>Enable Paging:</b> <input type="checkbox"/> <b>Paging Threshold: (in MB)</b> <input type="text" value="-1"/> <b>Max Field Size: (in Byte)</b> <input type="text" value="500"/> <b>Paging Buffer Size: (in MB)</b> <input type="text" value="20"/> <b>Total Paging Buffer Size: (in MB)</b> <input type="text" value="256"/> <b>Cleanup Paging Buffer: (in hour)</b> <input type="text" value="12"/> <b>Cleanup DHTMLViewer Buffer: (in hour)</b> <input type="text" value="2"/> <b>Cleanup PageViewer Buffer: (in hour)</b> <input type="text" value="2"/> <input type="button" value="Reset"/> <input type="button" value="Reset All"/>		
<input type="button" value="Save Settings"/>				

*Virtual Memory/Paging Options*

**Paging Threshold:**

This argument specifies when the paging feature will be activated. If this argument is set to “-1” then data will never be paged. This value is independent of Page Buffer Size option. If this value is larger than the Paging Buffer Size, then the system will begin paging when the thresh-

old is reached, not when the Paging Buffer Size is reached. Therefore each report or chart might use more memory than the amount specified in the Page Buffer Size. If this value is equal to or lower than the Paging Buffer Size, the paging feature will begin when the Paging Buffer Size is reached. This value is set in Megabytes.

**Max Field Size:**

This argument specifies the expected maximum field size for large field types such as varchars. Fields that are larger than this value are **not** necessarily truncated. The behavior depends on the current memory usage in relation to the Total Paging Buffer Size allowed. As the amount of available memory decreases the amount of data stored for these large fields will decrease until it reaches this value. For example, a user is running a report with a field size that is larger than the specified Max Field Size value. When the server is not busy and there is ample memory available, the report will generate the full field without any truncation. However, when the server is heavily loaded and the available memory is near 0, the field in the report will be truncated to the Max Field Size. For numeric, boolean, date, time, and character datatypes, the data will never be truncated. For those database fields that are defined with a size limit smaller than the Max Field Size (e.g. Max Field Size = 500, but the field is defined as varchar(20)), the database limit will be used in place of the Max Field Size. This value is set in Bytes.

**Paging Buffer Size:**

This argument allows you to set the amount of memory that each report or chart will use when the paging feature is invoked. The size of the buffer affects performance. The larger the buffer size, the faster the report or chart is generated, but more memory is used. When the amount of total memory used by the paging system approaches the **Total Paging Buffer Size**, the Paging Buffer Size provided to new reports will begin to diminish in order to avoid exceeding the specified total amount. This value is set in Megabytes.

**Total Paging Buffer Size:** This argument allows you to set the total amount of memory used by the paging feature across all reports and charts. The memory allocated to each report will diminish as the memory usage approaches this value. This value is set in Megabytes.

**Cleanup Paging Buffer:** This argument allows you to specify the cleanup interval for the temporary files used in the paging feature. Typically, these files are deleted when the reports are finished, however under rare circumstances (e.g. server crashed), these files are left on the server. The cleanup thread takes care of these rare scenarios. This value is set in hours.

**Cleanup DHTMLViewer Buffer:** In addition to the memory paging feature, the DHTMLViewer also provides a buffer to improve performance. Transparent to the user, the DHTMLViewer saves the exported pages to temporary files on the server when running reports. The report will be displayed as soon as the first page of the exported report is available. This means that loading a report and navigation between the pages will be very quick since there is no need to rerun the report at each page. The temporary files are deleted automatically upon user log out, however when the user doesn't log out properly, the files are left on the server. This argument, specified in hours, allows you to set the interval for the DHTMLViewer to delete these temporary files.

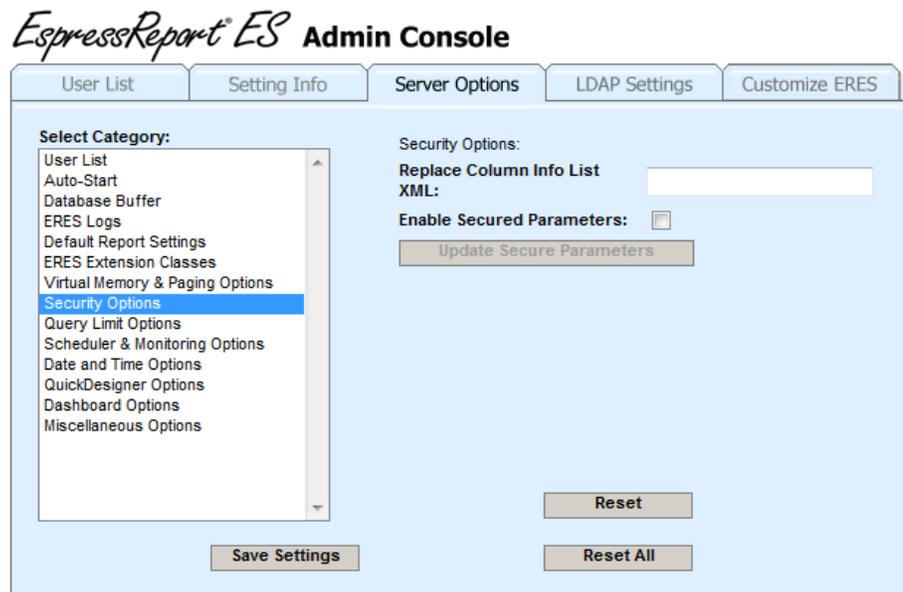
**Cleanup PageViewer Buffer:** This argument allows you to specify the cleanup interval for the files used in the PageViewer. Similar to the DHTMLViewer, these files help improve performance for users viewing reports in the Page Viewer. These files are typically cleaned up when the user logs out, but if the user does not log out properly, these files will be left behind and require clean up. This value is specified in hours.

**Best Practice:** Generally, if you have a lot of large reports, make sure to enable this feature by setting an adequate paging threshold. Specify the options based on the amount of memory available to your application server. For example, if you are running Tomcat with 512 MB memory, start with 256 MB for the Total Paging Buffer Size and around 10-20 MB for the Paging Threshold. This

way, between 10-20 users can be running the large reports without crippling the performance. The reason you should not use 100% of the Tomcat memory is because there are other components that will require memory as well. For instance, the POI libraries will require additional memory when exporting a report to Excel to generate the actual xls file. Each environment will be different, the number of large reports, the size of the result sets, and the number of concurrent users will all affect the optimized option settings. However, as a general starting point, shoot for around 50% of application server memory for the Total Paging Buffer Size. Then set the Paging Buffer Size to be around 2-10% of the Total Paging Buffer Size. The Paging Threshold should not be much greater than the Paging Buffer Size.

**Security Options:**

ERES also provides a set of sophisticated security options.



*Security Options*

**Replace Column Info List XML:**

This argument allows you to turn on the automatic decryption feature in QuickDesigners and menu page. The XML file must specify all the information for the column to be decrypted, including database information, table name, and column name. You can specify either an absolute path to the XML file or a relative path to the ERES install directory (i.e. `help/examples/DataDecryption/QBReplaceColumnInfoList.xml`). You can find more details as well as an example for this feature in Section 3.2.2.1.2 - Querying Encrypted Data.

**Enable Secured Parameters:**

This argument allows you to turn on the secured parameters feature. This feature activates a global security structure allowing you to specify the segments of data that each security level is entitled to view. You can find more details as well as interface for configuring Secured Parameters in Section 1.4.1.3.1 - Secured Parameter.

**Query Limit Options:**

These options allow you to set strict limits to the all query execution.

The screenshot shows the EspressoReport ES Admin Console interface. At the top, there are navigation tabs: User List, Setting Info, Server Options, LDAP Settings, and Customize ERES. The 'Setting Info' tab is active. On the left, a 'Select Category:' list includes options like User List, Auto-Start, Database Buffer, ERES Logs, Default Report Settings, ERES Extension Classes, Virtual Memory & Paging Options, Security Options, Query Limit Options (highlighted), Scheduler & Monitoring Options, Date and Time Options, QuickDesigner Options, Dashboard Options, and Miscellaneous Options. The main area displays 'Query Limit Options' with two input fields: 'Max Query Records: (Disable: -1)' and 'Query Timeout (Sec): (Disable: -1)', both set to '-1'. At the bottom, there are buttons for 'Save Settings', 'Reset', and 'Reset All'.

*Query Limit Options*

**Max Query Records:** This argument allows you to set a maximum limit for the number of records the server will retrieve from a database when executing a query. Once the maximum is reached, the server will stop running the query. If this argument is set to **-1** then no query record limit will be applied.

**Query Timeout (Sec):** This argument allows you to specify a timeout interval in seconds for report/chart queries. Once the query execution time passes the timeout argument, ERES will abort the query. This feature prevents users from accidentally creating run-away queries.

**Scheduler & Monitoring Options:**

These options allow you to set some options regarding scheduled reports.

The screenshot shows the EspressoReport ES Admin Console interface. At the top, there are navigation tabs: User List, Setting Info, Server Options, LDAP Settings, and Customize ERES. The 'Setting Info' tab is active. On the left, a 'Select Category:' list includes options like User List, Auto-Start, Database Buffer, ERES Logs, Default Report Settings, ERES Extension Classes, Virtual Memory & Paging Options, Security Options, Query Limit Options, Scheduler & Monitoring Options (highlighted), Date and Time Options, QuickDesigner Options, Dashboard Options, and Miscellaneous Options. The main area displays 'Scheduler & Monitoring Options' with three input fields: 'Max Scheduler Threads: (No Limit: -1)' set to '-1', 'Remove Schedule Icon Interval (in days):' set to '7', and 'Schedule Log Clean Up Interval (in days):' set to '30'. There is a checked checkbox for 'Re-run Missed Schedule Jobs'. Below these are buttons for 'View Scheduler Log' and 'View Alert Monitoring Log'. At the bottom, there are buttons for 'Save Settings', 'Reset', and 'Reset All'.

*Scheduler Options*

**Max Scheduler Threads:** The scheduler thread limit allows you to set the maximum number of sched-

ule jobs that can run concurrently. Additional jobs will be queued until one completes. If this argument is set to **-1** then no limit will be placed on the number of concurrent schedule jobs.

**Remove Schedule Icon Interval:** This value allows you to specify how long to retain the schedule icon in the menu page after a schedule has finished. After this duration is reached, the icon will be removed from the Menu Page. However, the exported file will remain in the export directory. The default interval is 7 days.

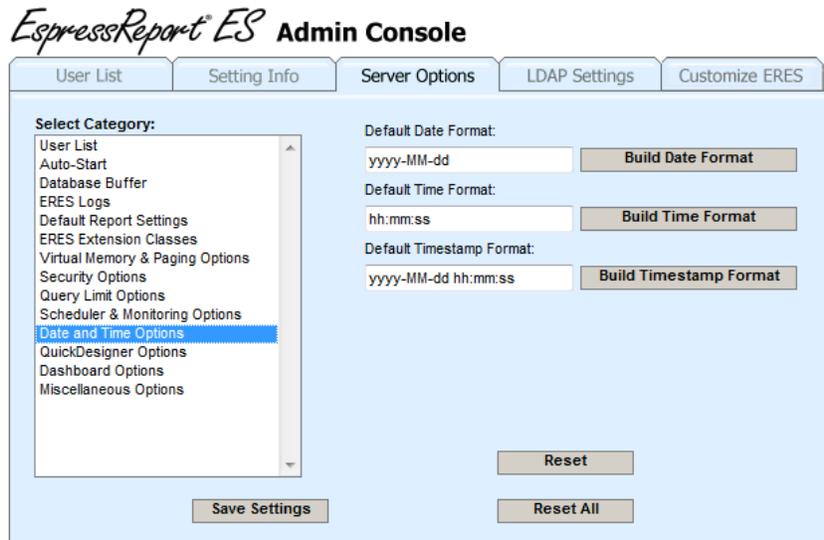
**Schedule Log Clean Up Interval:** This value allows you to specify how long to retain the finished schedule in scheduler log. The default interval is 30 days.

**Re-run Missed Schedule Jobs:** The re-run scheduled jobs option allows you to specify what should happen if a schedule job fails to execute due to server down-time. If this option is enabled, any missed schedule jobs will run when the server is restarted.

**View Scheduler Log:** Opens scheduler log. To learn more about scheduler log, visit Section 2.2.3.2 - Schedule and Archive Logs.

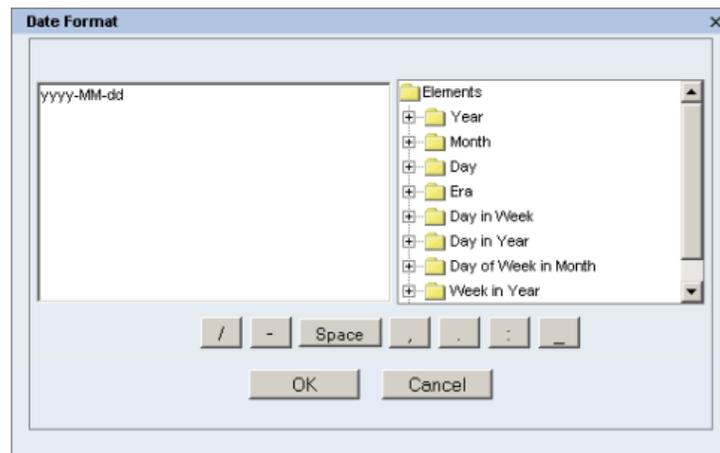
**View Alert Monitoring Log:** Opens alert monitoring log. To learn more about alert monitoring log, visit Section 11.4.4 - Monitoring log.

**Date and Time Options:** This section allows you to specify a default custom date, time, and timestamp format used when parameters are mapped to a column in a database. When the *Custom Date Format* option is checked, this is the default format presented to the user.



*Date and Time Options*

Click the *Build buttons* to build the custom format. The date/time representations are listed on the right and optional spacers and symbols are shown as a collection of buttons on the bottom. Once you have finished creating the format, click *OK* to save.

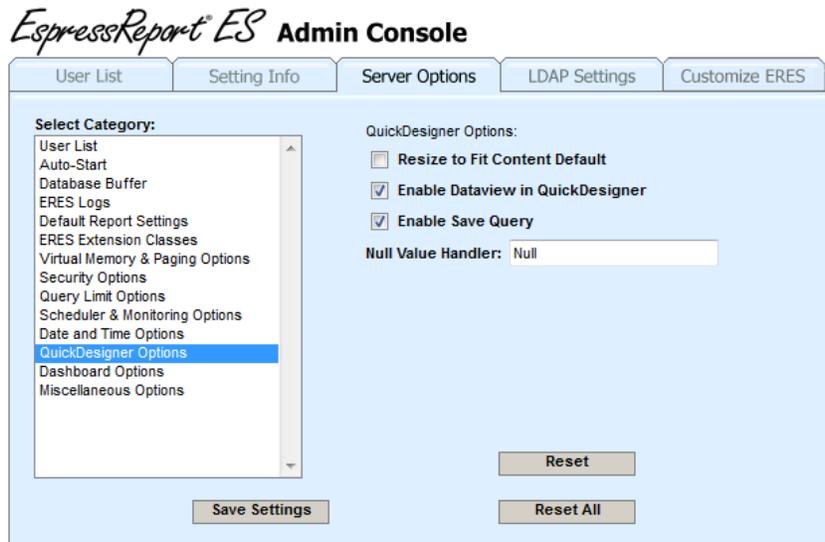


*Custom Timestamp Dialog*

#### **QuickDesigner Options:**

Please note, these settings apply only to the old QuickDesigner (from version 6.6). To see how to enable the old QuickDesigner, visit Section 1.4.1.2 - Setting Info.

This section provides several defaults for the QuickDesigner.



*QuickDesigner Options*

**Resize to Fit Content Default:**

While resize to fit content can be changed in any report within the QuickDesigner, this option allows you to set a default behavior. If you enable this option, all cells in the QuickDesigner will have Resize to Fit Content enabled by default.

**Enable Dataview in QuickDesigner:**

This option allows you to disable the data views in quick designers such that the user can only make ad hoc reports based on data view queries, but not from data views. This feature can prevent users from bypassing the data view secured parameters.

**Enable Save Query:**

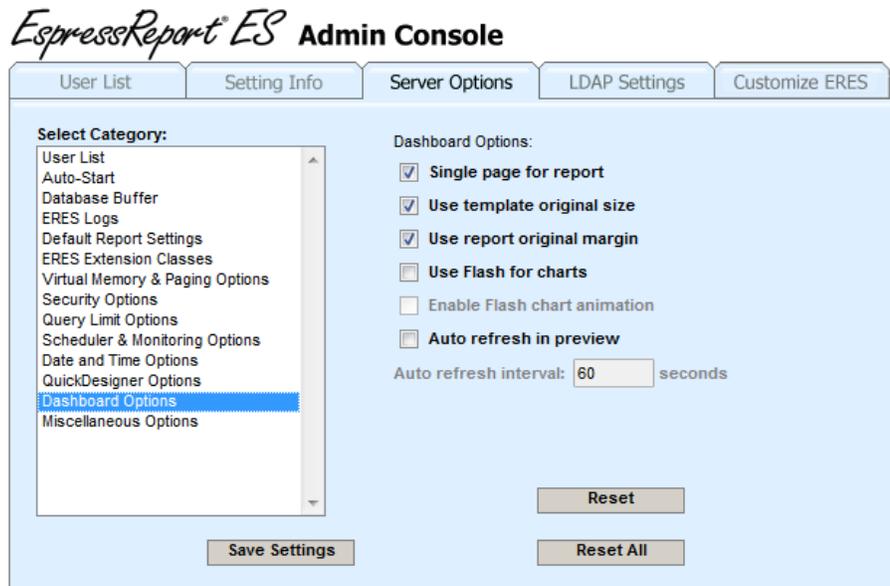
To prevent users from creating too many custom dataview queries, you can disable this option so that users are forced to work from the list of dataview queries that you have provided.

**Null Value Handler:**

This option allows you to control the display for null data in the QuickDesigner. The value can be any string that does not contain spaces. To insert a space into the null data handler, use the word *SPACE* in uppercase. For example: - *SPACE*- will replace all null values in the QuickDesigner with - -.

**Dashboard Options:**

This section provides several defaults for the Dashboard Builder.



*Dashboard Options*

**Single page for report:**

While the single-page or multi-page layout can be changed in any report within the Dashboard Builder, this option allows you to set a default behavior. If you enable this option, all reports in the Dashboard Builder will have the single-page layout enabled by default.

**Use template original size:**

This option allows you to configure the original size of the template to be used by default.

**Use report original margin:**

This option allows you to configure the original margin of the report to be used by default.

**Use Flash for charts:**

This option allows you to specify whether Flash should be used in charts by default.

**Enable Flash chart animation:**

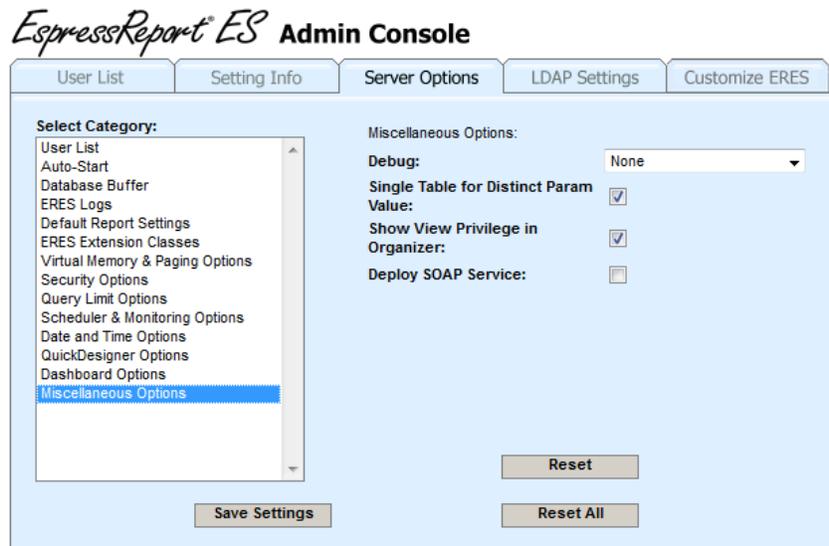
This option allows you to specify whether Flash animation should be turned on in charts by default.

**Auto refresh in preview:**

This option allows you to specify whether the previewed dashboard should be refreshed automatically, and if it should be refreshed automatically, you can configure the time interval for refreshing the dashboard preview.

**Miscellaneous Options:**

This section contains a number of general options available for ERES.



Miscellaneous Options

**Debug:**

When setting up the ERES system, you might need to enable the debug option for some components to troubleshoot problems. This option makes it easy to enable the debug statements for Secure Parameter, Replace Column, or Virtual Memory features. Once you restart the server, whenever the feature is invoked, informative outputs will be printed to the system console.

**Single Table for Distinct Param Value:**

When you enable this argument, the parameter dialog for parameterized charts and reports will be drawn differently. When you map the parameters to a database column, the distinct list will be drawn by running a select distinct on only the mapped field from the table that contains that field. The default behavior (without this argument) will use the joins and conditions in the original query to constrain the distinct parameter list.

**Show View Privilege in Organizer:**

By default, regular users (with designers role) will not be able to configure file privileges, but they can view them. Disable this option if you do not want users to see the privilege list for organizer files and folders.

**Deploy SOAP Service:**

This determines if ERES will deploy the SOAP Lookup Service on start up. The service allow developers to request reports and charts by sending relevant information using SOAP. Details can be found in Section 8.8 - SOAP Interface.

Once you've specified any changes to the options on this tab, click the *Save Settings* button to save the changes.



## Note

The changes will not take effect, nor will your changes display in the *Server Options* tab until the server is re-started.

These options can also be directly entered into a Server Command text file. For more information for the Server Commands, please see Appendix 1.B - Server Commands.

### 1.4.1.3.1. Secured Parameter

ERES contains a feature which allows you to filter data based on a user's security level within the QuickDesigners, Dashboard Builder, Dashboard Viewer and Published Files.

To set up and configure this feature, go to *Server Options* in the *Administration Console*, select the *Security Options* category, *Enable Secured Parameters* (if it is not checked), and click the *Update Secure Parameters* button. This will take you to the following dialog.

*Secured Parameter Dialog*

By default, there will be a secured parameter set up for *Customers.Region* in the HSQL database. You can add more secured parameters at any time by using the *Add* or *CopyDB* option at any time. The *Copy DB* option will copy the database connection information so you don't have to type it in again. Once you have created a new secured parameter or selected an existing one, the options on the right will enable. Fill in each of the options.

- Parameter Name:** This is just a name for the Secured Parameter, you can give it any name.
- Database Type:** This is a syntax assistance drop down box. Selecting the database will place default URL syntax and default JDBC Driver in the corresponding fields. You can then go into the URL and fill in the server specific information.
- Database URL:** Specify the URL for the database you would like to apply the Secured Parameter on. Use the Database Type drop down to assist in the syntax.
- Database Driver:** Specify the driver or use the Database Type drop down to use the default database driver.
- Table Name** Specify the table you would like to apply the Secured Parameter.
- Column Name:** Specify the column you would like to apply the Secured Parameter.
- Data Type:** This field will automatically be configured and is only displayed for verification.

<b>Allow Multiple Values:</b>	This will determine if security levels will each be able to access only one value or multiple values. If you select multiple values, the Values drop down box will become a list box allowing you to select multiple values.
<b>Security Level:</b>	Go through the security levels on the system and specify a value or values for each security level. The first time you select a security level, you will need to click the <i>Update Values List</i> button to retrieve a list of available values for that column.
<b>Values:</b>	The <i>Update Values List</i> button will launch a prompt for username and password to this database the first time it is run. Once you have supplied the information, it will retrieve a list of available values for the specified column. You can map a security level to one or more (if multiple is checked) values by clicking and highlighting the row of data. You can also use the <i>Grant All</i> option which allows that security level to view all data in that column.

Once you have set up mappings for secured parameters, click the *Done* button. That should bring you back to the Admin Console. Click the *Save Settings* button. You will be reminded to restart the ERES server. Confirm the message, go back to the ERES start page, shut down the server and then start it again.

After the server has been restarted, the secured parameters settings will carry through to the QuickDesigners, and Dashboards in ERES. To illustrate the concept, let us walk through a simple example.

Suppose you have a secured parameter set up for the *REGION* column from the *CUSTOMERS* table in the Woodview sample database. It specifies the behavior of the filter for six different security levels. For level *East* the query will be automatically filtered so that only rows where region is *East* will be returned. For level *South* the query will be automatically filtered so that only rows where region is *South* will be returned. For the level *Midwest* only rows where the region is *Midwest* will be returned. For the level *West* only rows for *West* region will be returned. The *Allow Multiple Values* attribute allows you to indicate whether multiple parameter values can be specified for the filter, like mapping the *EastSouth* security level to both *East* and *South*. For level *Executive*, *Grant All* is selected and all values from the region column will be available.

With the options set up, any time a user creates or runs a query, data view, or data view query in the QuickDesigners that includes the *CUSTOMERS* table, the filter defined for their security level will be applied. These settings reside on the server, and are not saved into the report, chart, or dashboard. Therefore if the user then creates and saves a report in QuickDesigner Reports and transfers the report to another machine, the security will not be apply. Instead, it will follow the security settings sepecified on the new Server.

For information about applying security filters to data views and data view queries, please see Section 3.1.3.3.3 - Data View Security.

### 1.4.1.4. LDAP Settings

The *LDAP Settings* tab in the Admin Console provides options that allow you to configure ERES to retrieve users from an LDAP server. To set the LDAP connection, select *Use LDAP Security Provider* option. The following dialog should open.



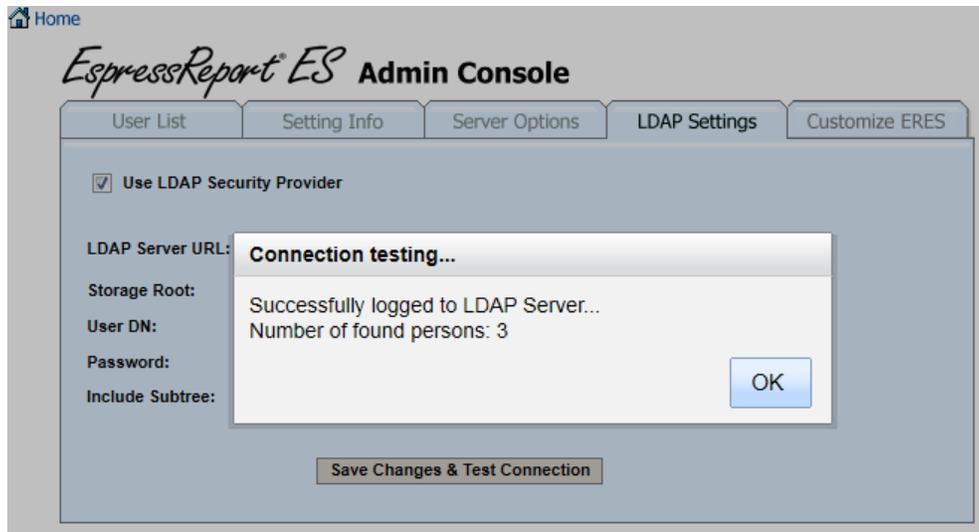
*LDAP Settings Dialog*

The above dialog contains example LDAP settings of Microsoft Active Directory.

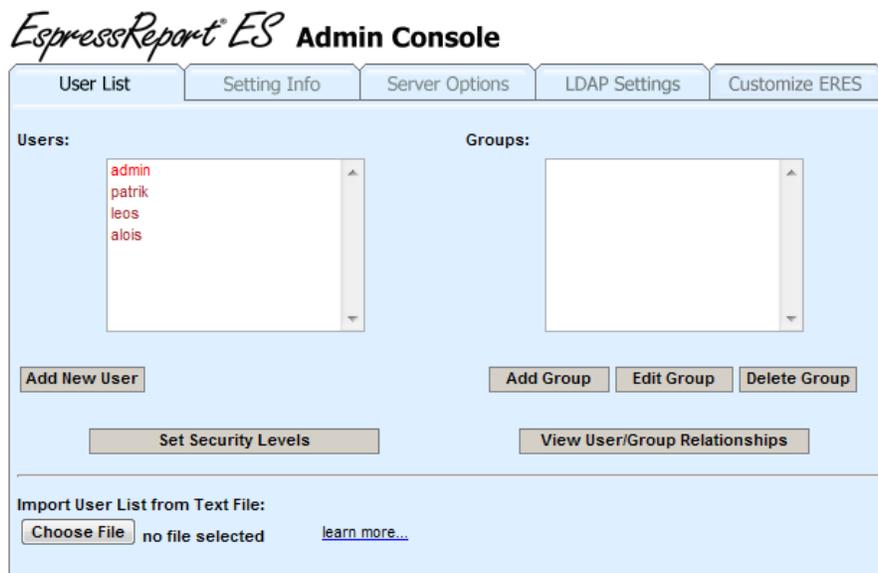
To the *Storage Root* field, enter the distinguished name of the AD node that contains all the users you want to import. By default, only the given AD node (i.e. the *Storage Root* node) will be searched for users, excluding it's sub-nodes. If you want to search for users in the *storage root* node and all of it's sub-nodes, select the *Include Subtree* option.

In order to connect with ERES you will need to have an user in your LDAP server that can read from the Active Directory. Enter the user's distinguished name in to the *User DN* field and then type the user's password to the *Password* field.

Click the *Save Changes & Test Connection* button. If you set the LDAP correctly, the following dialog will show up.



Click *OK*. The LDAP users were imported, but haven't been activated yet. To activate the users, click on the *Users List* tab.



Legend: Administrator, Designer, Viewer, LDAP user without access, Group

There are some users marked as *LDAP users without access* in the *Users* list (user roles are distinguished by their colors - there is a legend for that below the AdminConsole).

Select one or more LDAP users without access. The *Insert LDAP User(s)* button will show up!. Click on it.



### Note

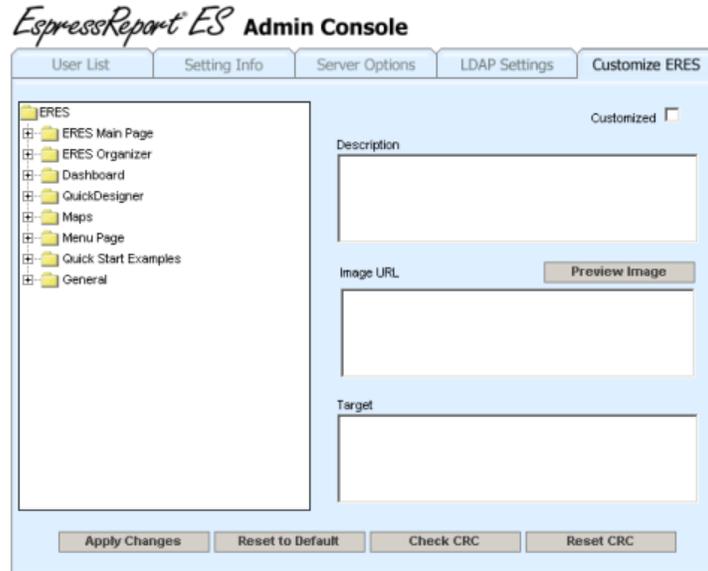
If you include a non-LDAP user in the selection, the 'Insert LDAP User(s)' button will disappear.

A new window dialog will show up allowing you to select user role for the selected user(s). To learn more about user roles, visit Section 1.4.1.1 - User List. Select a user role and click *Ok*.

The users will not be marked as *LDAP Users without access* any more. Their color changed from dark red to light red (if you gave them admin privileges), blue (if they are designers) or black (if you gave them viewer rights). They are valid ERES users and they can start using the ERES now.

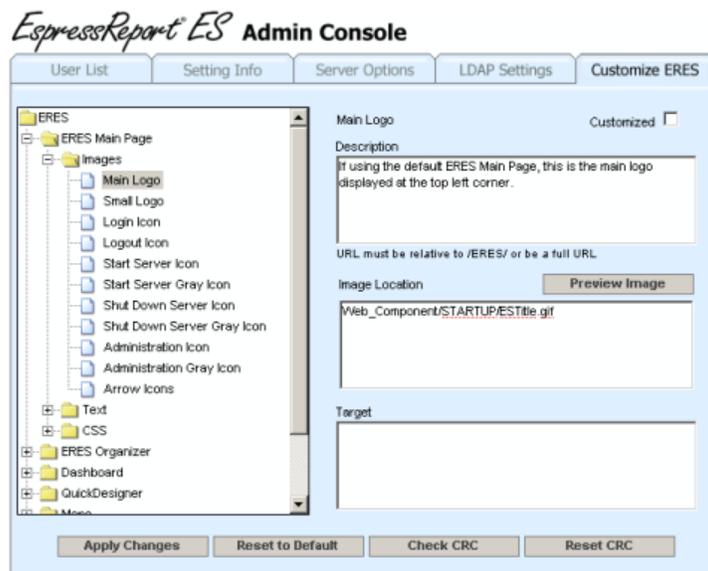
### 1.4.1.5. Customization

ERES can be customized to maintain a consistent look and feel for your web application. The customization tab allows the administrator to change the appearance of many components in ERES. When customizing, it is recommended that you save your images and files under different names rather than overwriting existing files in the ERES. This way, you will not have to worry about future updates overwriting your changes.



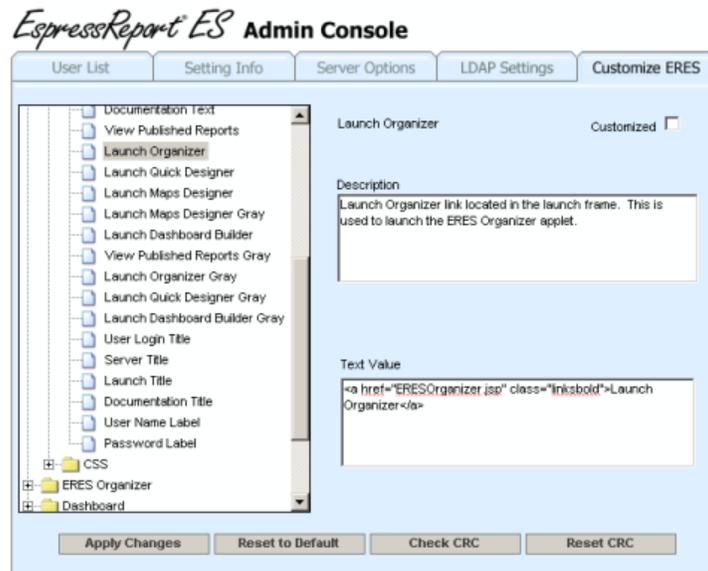
*The Customization Interface*

The left window provides a tree of the different elements that can be customized. The customizable elements are broken up into sections based on the components in the ERES application. Most sections contain three types of elements: images, text, and CSS files. By clicking on an image element, the page will display the description, an image URL, and a target URL if available. You can change the image file by providing a different URL. Clicking on the *Preview* button will display the image at the bottom of the screen. The preview window is only available for images, clicking on a text or CSS element will disable the preview window. The target URL is only available for certain elements. If the selected element allows for a target to be set, this text box will appear beneath the image URL box. Enter a URL in this window if you would like to add a hyperlink for this image. If this link is provided, when users click on the image, the browser will be redirected to the target URL.



*Preview Image*

If you select a text node, the page will display the description and a text value. Change the text value to adjust the message or link for this element. The text will be used directly in the ERES application, so you can enter any html tags in the tool. Certain elements also contain a link. You can utilize this to redirect the user to another URL if you are not using the default components. For example, if you are using a customized Menu Page, you can modify the link for the Published Files element to point to your customized page. Editing the text value also gives you the ability to add even more CSS to the page.



*Customizing Text*

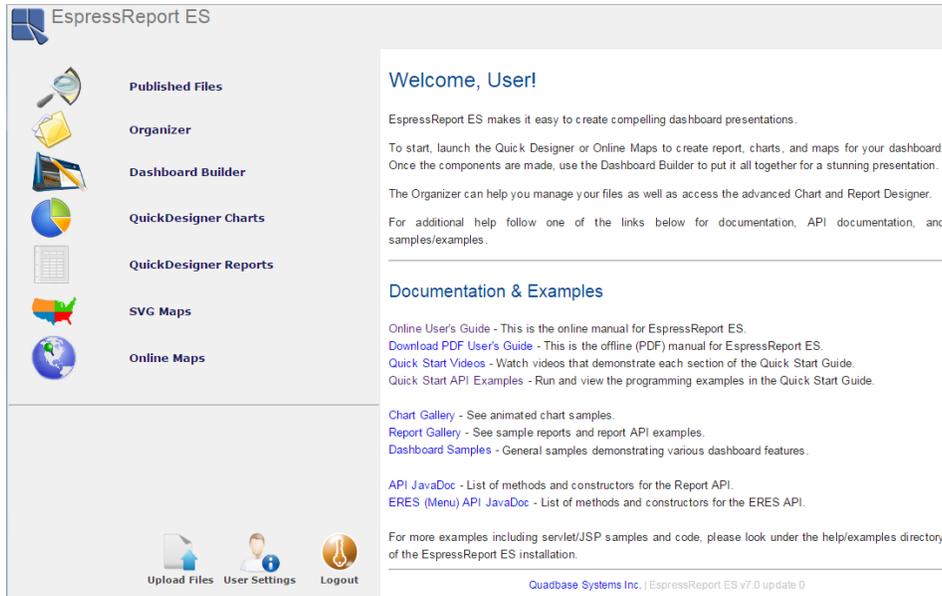
Changing the CSS file is quite straightforward, simply enter the url for the new CSS file. It is recommended that you make a copy of the original CSS file and edit the copied version so that future updates will not overwrite your changes. As mentioned earlier, you can specify new CSS elements in customized text elements and enter the style in your CSS file.

When you change the values for any element, the *Customized* box for that element will automatically check. If you wish to revert back to the default value for this element, simply uncheck the *Customized* box and choose *Ok* from the pop up dialog. If you wish to reset all elements to their default values, click on the *Reset to Default* icon at the bottom.

Once you are finished with your changes, make sure to click *Apply Changes* to save all changes. To see the changes appear in the ERES application, you will need to shut down the ERES Server and restart it.

## 1.4.2. User Settings

In addition to the administrator, users can modify some of their own information/settings in ERES. When a user other than the administrator logs in in the Start page, the *User Settings* link will appear at the bottom on the left side.



*User Logged into ERES Start Page*

When the link is clicked, a new window will open that allows the user to modify the email address and password associated with his/her account.

**User Settings:**

Username: **user**

Email Address:

Current Password:

New Password:

Confirm New Password:

*\*Leave password fields blank if not changing password.  
 \*Passwords must be alpha-numeric and between 4-45 characters long.*

*User Settings Dialog*

After making the appropriate changes, clicking *Ok* will save the user changes and close the window.

## 1.A. ERES Details Log

The ERES Details Log Report maintains a record of many key events that take place. Administrators can use this information to track what changes were made, which user made the changes, and when the changes took place. This feature is also useful for troubleshooting issues and backtracking through events.

Here is a list of all events recorded in the ERES Detailed Log Report:

### ERES Events

- Login
- Timeout
- Logout

- Set Webroot
- Set Server Name
- Set Protocol
- Set Port Number
- Set Web Server Port
- Set JSP Default Directory
- Set Servlet Context
- Set SMTP Hostname
- Repair Broken Links
- Replace Directory
- Add New Node
- Remove Node
- Copy Node
- Move Node
- Set Name
- Set Node Description
- Set Node Filename
- Update Node Owner Id
- Add Schedule Task
- Add Report Schedule Task
- Add Chart Schedule Task
- Remove Schedule Task
- Modify Schedule Task
- Run Schedule Task
- Add Data Registry
- Remove Data Registry
- Set Data Registry Permission
- Delete Data Registry File
- Set User Data File
- Set User Permission
- Set Group Permission
- Insert Security Level

- Remove Security Level
- Set Security Levels
- Set Node Permission
- Set File Permission
- Set Data Registry Invisible Node
- Table Setvalue
- Table Add Item
- Table Remove Item
- Table Create Item
- Urlmap Setvalue
- Urlmap Add Item
- Urlmap Insert Item
- Urlmap Move Item
- Urlmap Remove Item
- Urlmap Create Item
- Schedule Export
- Write Dataview
- Write Queryfile
- Menu Viewpage
- Menu Login
- Open Report
- Open Chart
- Export
- Set Cleanup Log Interval
- Get Cleanup Log Interval

### **Language Events**

- User Language

### **Misc Events**

- Get OS Name
- Data Registry

### **Report & Chart Designer Events**

- Default Files

- Export
- Pageviewerhtml
- Stream
- Stream Read Only
- Pageviewer Stream
- Load View Page Slave
- Load View Page Export
- File Info
- License Info
- Pageviewer File Info
- Fileinfo Root
- XML Report
- XML Report Data
- XML Data Parser
- XML Data Sheet
- XML Global Format
- XML Font Mapping
- Data File
- EJB Access
- EJB Metadata
- EJB Home Method Access
- XML Verify
- SQL
- Test Db Connection
- Url
- Rename
- Replace
- Export Barcode
- Server Directory
- Db Distinct
- Db First Value
- Db Param Meta
- Db Param Function Meta

- Db Param Type
- Db Query Meta
- Data File Meta

### **Query Builder Events**

- Qb Product Name
- Qb Numeric Function
- Qb System Function
- Qb String Function
- Qb Time Date Function
- Qb Table List
- Qb Table Info
- Qb SQL

### **Chart Generator Events**

- Chart Export
- Filter Data File
- XML Chart
- XML Template
- Creat Chart
- Filter PPSQL
- Img Export
- Cht Export
- Delete
- Delete Temp File
- Release File
- Copy File
- Working Directory
- Outputstream Initialize
- Outputstream Write
- Outputstream Close
- Classfile

Here is a list of the possible products represented in the log.

Loginpage

Adminpage

Menupage

Dashboard

Organizer

Report

Chart

Scheduler

Lookupservlet

Dhtmlviewer

Dashviewer

Soap

Designer

## 1.B. Server Commands

The configurations specified in the Server Options tab can also be set in a configuration text file. Under the `<ERESInstallDir>/WEB-INF/classes` directory there is a file called `QB.properties` that contains several arguments that control server settings. The `QB.properties` is a INI file. Server commands belong to the "ServerCommands" property. Arguments in the text file assume the format of a dash "-" followed by a command. Arguments should be added in a single line (starting with `ServerCommands=`) in the file separated by spaces.

- log:** When you type the `-log` argument, this turns on the basic server logging. This has the same effect as enabling the option in the Admin Console.
- writeLogToDatabase:** This turns on the detailed server logging feature. This has the same effect as enabling the *Generate Detail Logs* option in the Admin Console.
- cleanupIntervalForLog:hh:** This is the cleanup interval for the detail logs. The value following the colon is specified in Hours.
- recordLimit:nn:** This argument allows you to set a maximum limit for the number of records the server will retrieve from a database when executing a query. This has the same effect as setting the record limit in the Admin Console
- queryTimeout:sss:** This argument allows you to specify a timeout interval in seconds for report/chart queries. This has the same effect as setting the query timeout in the Admin Console.
- DBBuffer:nnn:** This allows you to set the number of database buffers for the server to use. This argument is the same as the buffering feature available in the Admin Console. The argument allows you to specify the number of buffers from 1 to 999.
- DBCleanAll:ddhhmm:** This allows you to set the clean up interval for the database buffers. This argument is the same as the clean up feature in the Admin Console. The value of `ddhhmm` means "dd" days, "hh" hours and "mm" minutes. You can also use omitted format, meaning that the

---

server will translate the value with the assumption that the omitted digits are the higher digits. For Example:

DBCleanAll:101010 means clean buffer every 10 days, 10 hours and 10 minutes

DBCleanAll:1010 means clean buffer every 10 hours and 10 minutes

DBCleanAll:10 means clean buffer every 10 minutes

**-RequireLogin:**

This argument is used in conjunction with the QbDesignerPassword argument to apply security to Report Designer and Chart Designer when they are launched via the API. Normally when Report or Chart Designer is called via the API there is no user authentication. This allows users to apply their own authentication to the programs, but can also allow unauthorized users access to the server. To prevent this, users can turn on this argument (values are true/false) to force authentication for Report and Chart Designer when they are called from the API. This option cannot be set in the Admin Console

**-QbDesignerPassword:**

This argument allows you to set the password to use when the *-RequireLogin* argument is turned on. The password specified here needs to be passed in via a method call when calling Report or Chart Designer via the API. This option cannot be set in the Admin Console.

**-globalFormat:<xmlfile>:**

This argument allows you to supply a global format XML file to set the default look and feel of report elements. This argument is the same as the default global formatting option in the Admin Console. For this argument specify a file path to the XML file relative to the ERES install directory i.e. (-globalFormat:ReportFiles/FormatFile.xml).

**-fontMapping:<xmlfile>:**

This argument allows you to supply a font mapping XML file to set the default font mapping for PDF export. This argument is the same as the default font mapping option in the Admin Console. For this argument specify a file path to the XML file relative to the ERES install directory i.e. (-fontMapping:Templates/FontFile.xml).

**-htmlDpi:nn:**

This argument allows you to hard-code the screen resolution that should be used when reports are exported to DHTML format. This argument is the same as the HTML DPI option in the Admin Console.

**-ListenerManagerClass:<classfile>:**

This argument allows you to specify a class for the ERES listener mechanism. This argument is the same as the ERES listener class option in the Admin Console.

**-LoginListenerClass:<classfile>:**

This argument allows you to specify a class for the ERES login listener mechanism. This argument is the same as the ERES login listener class option in the Admin Console.

**-ScheduleCallBackClass:<classfile>:**

This argument allows you to specify a class for the scheduler callback mechanism. This argument is the same as the scheduler callback class option in the Admin Console.

**-UserSecurityProviderClass:<classfile>:**

This argument allows you to specify a class file for the user security provider interface. This argument is the same as the user security provider class option in the Admin Console.

---

<b>-singleTableForDistinctParamValue:</b>	When you use <i>-singleTableForDistinctParamValue</i> argument, the parameter dialog for parameterized reports and charts will be drawn using a select distinct on a single table. This argument is the same as the <i>Single Table for Distinct Param Value</i> option in the Admin Console.
<b>-showViewPrivilege:true/false:</b>	This option determines if non-admin users will be able to view the file and folder privilege in the organizer. By default this option is true.
<b>-schedulerThreadLimit:nn:</b>	This argument allows you to set the maximum number of schedule jobs that can run concurrently. This argument is the same as the max scheduler threads option in the Admin Console.
<b>-runMissedScheduleJob:</b>	When you use the <i>-runMissedScheduleJob</i> argument, any schedule jobs that were missed due to server down-time will be run when the server is re-started. This argument is the same as the re-run missed schedule jobs in the Admin Console.
<b>-batchDeleteForScheduler:TRUE/ FALSE:</b>	This argument controls the process by which schedule job information is removed from the ERES database for expiring schedule jobs. Setting this option to true can improve schedule performance if you have large numbers of jobs that will expire at the same time. The effects of this feature may vary depending on the ERES database platform. This option cannot be set in the Admin Console.
<b>-xmlEncoding:encoding:</b>	This argument allows you to specify the encoding that ERES should use when writing XML. This includes data registry files, XML report and chart templates, XML exports, and XML global formatting, and font mapping files. This parameter needs to be set both in the server and in the Organizer. For more information about XML encoding, see Section 10.1.2.4 - XML Encoding.
<b>QuickDesignerSecuredParameters:xmlfile:</b>	This argument allows you to supply a data view security XML file to enable/apply security filters to data views and data view queries. You can specify either an absolute path to the XML file or a relative path to the ERES install directory i.e. ( <i>-QuickDesignerSecuredParameters:qDesigner/WoodviewParameters.xml</i> ). For more information about secured parameters, see Section 1.4.1.3.1 - Secured Parameter.
<b>-debug:DVW_SEC_PARAM:</b>	Use this argument to troubleshoot the <i>-QuickDesignerSecuredParameters</i> option. This debug statement will print multiple debug statements to the server console specifying whether there is a match for the column listed in the xml file. If you specify multiple debug options, only the last one will take effect.
<b>-disableDataViewInQuickDesigner:</b>	This argument allows you to disable the data views in quick designers such that the user can only make ad hoc reports based on data view queries, but not from data views. This feature can prevent users from bypassing the data view secured parameters.
<b>-disableSaveQueryInQuickDesigner:</b>	This argument allows you to disable the save query option in quick designers. This feature prevents users from creating and saving too many queries in the data registry.
<b>-ReplaceColumnInfoList:xmlfile:</b>	This argument allows you to turn on the automatic decryption feature in QuickDesigners and Menu Page. The XML file must specify all the information for the column to be decrypted, including database information, table name, and column name. You can specify either

---

- 
- an absolute path to the XML file or a relative path to the ERES install directory i.e. (`-ReplaceColumnInfoList:help/examples/DataDecryption/QBReplaceColumnInfoList.xml`). You can find more details as well as an example for this feature in Section 3.2.2.1.2 - Querying Encrypted Data.
- debug:REPLACE\_SQL\_COLUMN:** Use this argument to troubleshoot the `-ReplaceColumnInfoList` option. This debug statement will print multiple debug statements to the server console whether there is a match for the column listed in the xml file. If you specify multiple debug options, only the last one will take effect.
- debug:MEMORY\_PAGING:** Use this argument to troubleshoot the virtual memory/paging feature. This debug statement will print debug statements to the server console regarding the amount of memory used, and the paging process. If you specify multiple debug options, only the last one will take effect.
- ExcelExportFitCell:true/false:** Set this argument to true if you want to fit numeric values into single cells when exporting your report to Excel from the Menu Page, DHTML Viewer, or Dashboard. By default, this option is false. For more information on this feature, see Section 4.1.5.2 - Exporting Reports
- paperSize:LETTER/A4:** Set this argument to define the default paper size. If this option is not set, reports will default to Letter size.
- ExportNewlineDelimiter:windows/mac/others/system:** Set this argument to specify a default newline delimiter for CSV export. Use windows for any Windows version, it will use `\n\r` for the delimiter. Use mac for any version of Mac, it will use `\r` for the delimiter. Use others for any Unix/Linux variants, it will use `\n` for the delimiter. You can also use system to allow the server to pick up the system default value.
- QDResizeToFitContentDefault:true/false:** Please note, this command applies only to the old QuickDesigner (from version 6.6). To see how to enable the old QuickDesigner, visit Section 1.4.1.2 - Setting Info.  
When you set this argument to true, reports created in QuickDesigner will automatically have the *Resize to Fit Content option* turned on for all columns. By default this option is turned off.
- QDNullValueHandling:"<NullValue>":** Please note, this command applies only to the old QuickDesigner (from version 6.6). To see how to enable the old QuickDesigner, visit Section 1.4.1.2 - Setting Info.  
This argument allows you to control the display for null data in the QuickDesigner. `<NullValue>` can be any string that does not contain spaces. To insert a space into the null data handler, use the word "SPACE" in uppercase. For example:  
`-QDNullValueHandling: "-SPACE- "`  
will replace all null values in the QuickDesigner with " -".

---

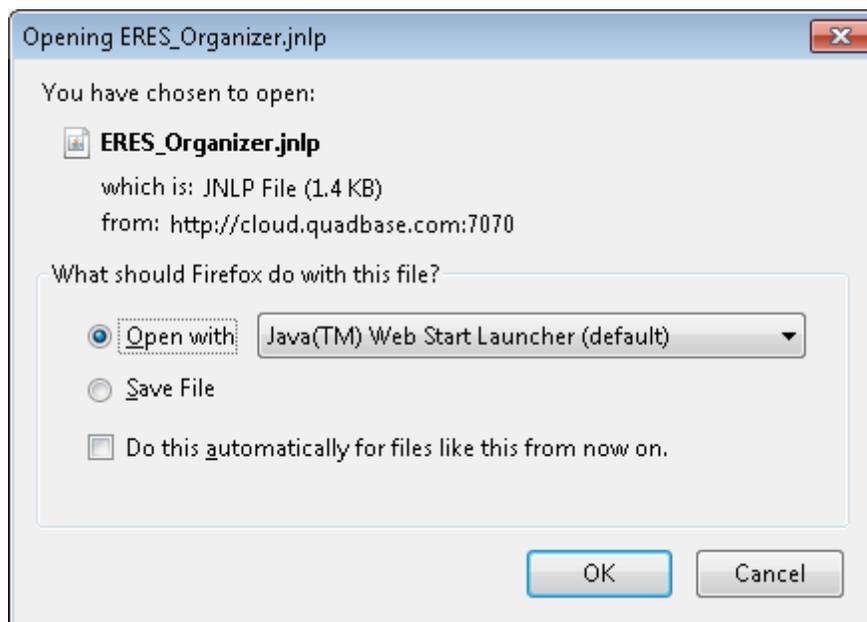
# Chapter 2. Organizer

## 2.1. Using the Organizer

The EspressoReport ES Organizer is a powerful graphical user interface that allows users to design charts and reports and organize them with other files in virtual folders. The Organizer contents are automatically published to the Menu page and users can automatically generate URLs to run reports and charts. In addition, schedule and archive jobs are created and maintained through the Organizer. The Organizer also includes the user permissions and security level setting options.

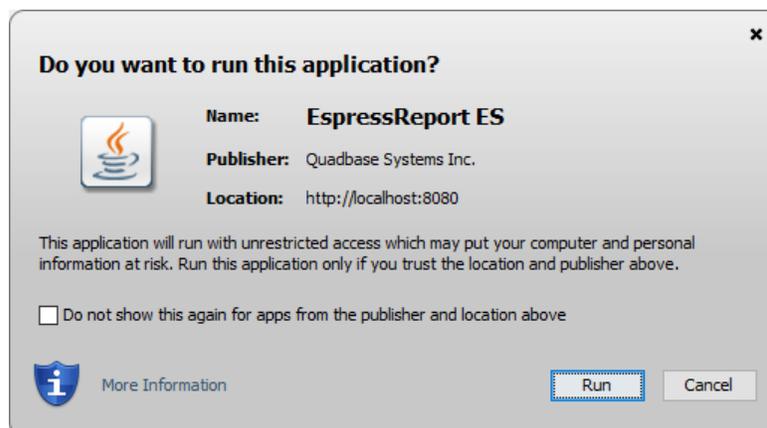
### 2.1.1. Starting the Organizer

To start the Organizer, click the link labeled *Organizer* in the Start page. This will open a pop-up window. Confirm `ERES_Organizer.jnlp` file open by Java Web Start Launcher. Then the Organizer will open.



*Opening Organizer*

If you use Chrome as a browser, you have to save `ERES_Organizer.jnlp` at first, and then to open it from your download directory. You will see the next dialog. Click *Run* and Organizer will open.

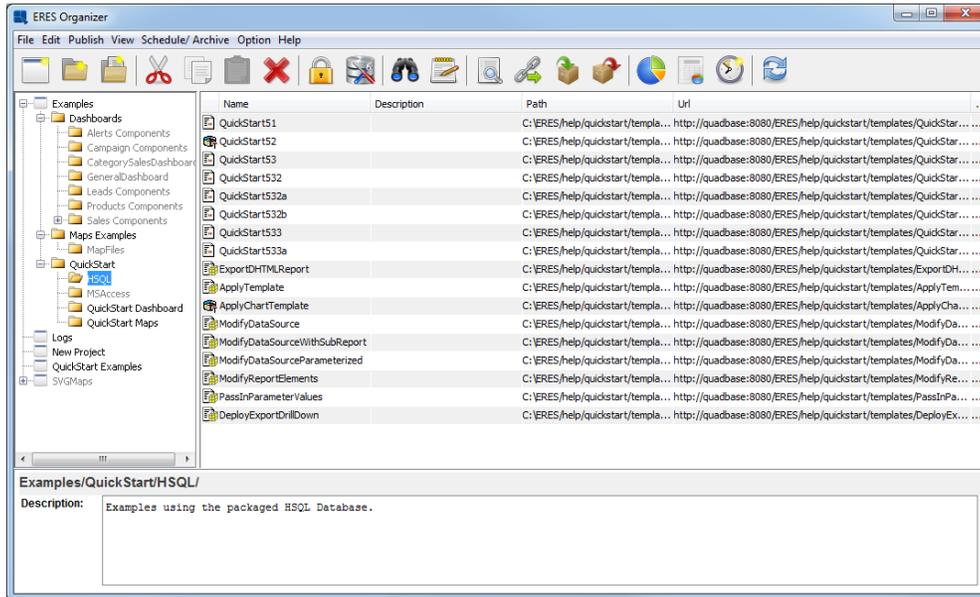


*Opening Organizer in Chrome*

If you are working on the same machine as the ERES Server, you can also start the Organizer as an application. To do this, go to the directory where EspressoReport ES is installed. With the server running, execute the `ERESOrganizer.bat/.sh` file. This will open a dialog allowing you to login to the Organizer.

## 2.1.2. The Organizer Interface

Since the Organizer is at heart a file management system, it is no coincidence that the application window resembles a file browser. The left-hand side of the window contains a list of projects and their associated sub-folders, while the right-hand side lists the contents of a folder in the upper pane and the project/folder descriptions in the lower pane.



*Organizer Interface*

### 2.1.2.1. The Organizer Menus

Most of the Organizer functions can be controlled using the drop-down menus located at the top of the Application window.

#### 2.1.2.1.1. File Menu

This menu performs most of the file operations. Options in this menu include:

- New** If you select this option, another menu pops up.
- Project** Creates a new project in Organizer.
- Folder** Creates a new folder in current project/folder.
- Chart** Launches ChartDesigner. (see Section 4.2.1 - Introduction to Chart Designer).
- Report** Launches ReportDesigner (see Section 4.1.1 - Introduction to Report Designer).
- Dashboard** Launches DashboardBuilder (see Chapter 6 - Designing Dashboards).

**Manage Data Sources:** This option allows you to set-up and manage data registries, their associated data files, and database queries.

**Open File:** This will open the currently selected file. If the file is a chart then the Chart Designer will open, allowing you to edit the chart. If the file is a report file then the Report Designer will open allowing you to edit the report.

<b>Create PAK:</b>	Pack a GXML (Online Map), SXML (SVG Map) or DSB (Dashboard) file and its components (Templates, Tooltips etc.) into a single file (see Section 2.1.4.5 - Dashboard and Map Packages).
<b>Unpack File:</b>	Unpack a GPAK (Online Map), SPAK (SVG Map) or DPAK (Dashboard) pack file into Organizer (see Section 2.1.4.5 - Dashboard and Map Packages).
<b>Delete:</b>	This will delete the currently selected project, folder, or file.
<b>Refresh:</b>	This will refresh your Organizer window, allowing it to reflect the latest changes/modifications performed by other users.
<b>Exit:</b>	This will close the Organizer.

### 2.1.2.1.2. Edit Menu

This menu allows you to edit, and cut/paste files, folders and projects. Options in this menu include:

<b>Insert File(s):</b>	This allows you to insert a report or chart into the selected project/folder.
<b>Edit:</b>	This allows you to edit the currently selected project, folder, or file. For projects and folders, you can change the project/folder name as well as its description. For files, you can change the display name, the location path or URL, and the description.
<b>Set User Privileges:</b>	This option is only available to the administrator. It allows you to set owner privileges for defined users and groups.
<b>Set Security Levels:</b>	This option is only available to the administrator. It allows you to setup and maintain user/group security levels that are applied to reports.
<b>Change Ownership:</b>	This option is only available to the administrator. It allows you to change the owner of the selected file, folder, or project.
<b>Cut:</b>	This removes the selected project, folder, or file, and places it on the clipboard.
<b>Copy:</b>	This makes a copy of the selected project, folder, or file, and places it on the clipboard.
<b>Paste:</b>	This pastes the current clipboard project, folder, or file into the currently selected project or folder.

### 2.1.2.1.3. Publish Menu

This menu allows you to setup and view published reports and charts. Options in this menu include:

<b>Preview Menu Page:</b>	This will load the ERES Menu page in a new browser window.
<b>Generate Image URL:</b>	This option generates an Image URL for the currently selected chart.
<b>Generate Report URL:</b>	This option generates a Report URL for the currently selected report.
<b>Generate Dashboard URL:</b>	This option generates a Dashboard URL for the currently selected dashboard.

### 2.1.2.1.4. View Menu

This menu allows you to search for files or edit charts and reports. Options in this menu include:

<b>Search:</b>	This allows you to search through projects, folders, and files.
<b>Chart Designer:</b>	This will open the Chart Designer, allowing you to design or modify charts.

---

**Report Designer:** This will open the Report Designer, allowing you to design or modify reports.

### 2.1.2.1.5. Schedule/Archive Menu

The menu allows you to set and view options related to chart/report scheduling and archiving. Options in this menu include:

**Set Schedule:** This allows you to set a schedule for the currently selected chart, report, project, or folder.

**Set Archive:** This allows you to set an archive for the currently selected chart, report, project, or folder.

**Set Alert Monitoring:** This allows you to set an alert monitoring for the currently selected chart, report, dashboard, svg map, project, or folder.

**View Schedule Tasks:** This will open the log allowing you to view all the of the currently scheduled and archived jobs.

**View Alert Monitoring Tasks:** This will open the log allowing you to view all the current alert monitoring jobs.

### 2.1.2.1.6. Option Menu

This menu allows you to set and view several of the Organizer options. Options in this menu include:

**Set Designer:** This allows you to specify whether to open every chart or report in the same Chart Designer or Report Designer window (thus closing the current chart or report), or to open a new Chart Designer or Report Designer window for each chart or report opened.

**Set URL Mapping:** This allows you to set up and modify virtual directories.

**Repair Broken Links:** This allows you to search for, repair, or delete any broken links to files in the Organizer.

**Update Directory:** This allows you to update the path of certain items in the Organizer. This is helpful if you are moving files between installations or moving complete installations.

### 2.1.2.1.7. Help Menu

This menu allows you to view a version of the program and to open a documentation. Options in this menu include:

**About:** This brings you an information about the version of the program.

**Contents:** This opens the EspressoReport ES User's Guide.

## 2.1.2.2. The Organizer Toolbar

The toolbar at the top of the Organizer window offers easy access to some of the most commonly used features. The buttons perform the following functions:



Create a new project



Insert a new folder into the current project



Insert a new file into the current project or folder

- 
-  Cut the currently selected project, folder, or file, and place on the clipboard
  -  Copy the currently selected project, folder, or file, and place on the clipboard
  -  Paste the current clipboard project, folder, or file into the currently selected project or folder
  -  Delete the currently selected project, folder, or file
  -  Set user privileges (Administrator only)
  -  Manage Data Registries, files, queries, etc.
  -  Search for projects, folders, or files
  -  Edit the currently selected project, folder, or file
  -  Preview the menu page
  -  Generate a URL for the currently selected file
  -  Pack a map or a dashboard and it's components into a single file (see Section 2.1.4.5 - Dashboard and Map Packages)
  -  Unpack a GPAK (Online Map), SPAK (SVG Map) or DPAK (Dashboard) pack file into Organizer (see Section 2.1.4.5 - Dashboard and Map Packages).
  -  Go to Chart Designer
  -  Go to Report Designer
  -  Set schedule for the currently selected chart, report, folder, or project
  -  Refresh the Organizer window

### 2.1.3. Projects and Folders

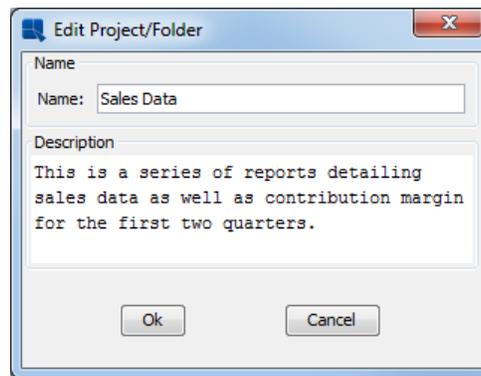
Projects and their associated folders are the central part of the Organizer interface. They provide the primary means of organizing charts and reports. Projects are the top level of organization. You can use different projects for different users, for different groups in your organization, or to group charts and reports representing different data or information. You can insert many layers of sub-folders into the top-level project, creating just about any organizational structure.

It is important to note that all folders in the Organizer are virtual folders, meaning that they do not necessarily have a physical location on the server machine or mirror an actual directory structure. Projects and folders in the Organizer are pointers to real directories and files that may be stored in a variety of locations.

### 2.1.3.1. Creating and Modifying Projects and Folders

To create a new project, you can do one of the following: click the *New Project* button on the toolbar, select *New Project* from the File menu, right click in the left-hand pane and select *New Project* from the pop-up menu, or press **Ctrl+P**. A node labeled *new project* will appear at the top of the file tree in the left-hand pane.

To edit the project name and description, first select it, then you can do one of the following: click the *Edit* button on the toolbar, select *Edit* from the Edit menu, press *F2* key, or right click, and select *Edit* from the pop-up menu. A dialog will then appear, prompting you to change the project name and description. Once you have entered your desired changes, click *Ok* and the new project name and description will be displayed in the lower right-hand pane.



*Edit Project/Folder Dialog*

To insert a sub-folder into a project or folder, first select the project or folder in which you would like to place the new sub-folder. Then you can either click the *New Folder* button on the toolbar, select *New Folder* from the File menu, or right-click in the left-hand pane and select *New Folder* from the pop-up menu. A node labeled *new folder* will appear under the currently selected project or folder.

To edit the folder name and description, first select it, and then you can either click the *Edit* button on the toolbar, select *Edit* from the Edit menu, press *F2* key, or right-click, and select *Edit* from the pop-up menu. A dialog will then appear prompting you to change the folder name and description. Once you have entered your desired changes, click *Ok*, and the new folder name and description will be displayed in the lower right-hand pane.

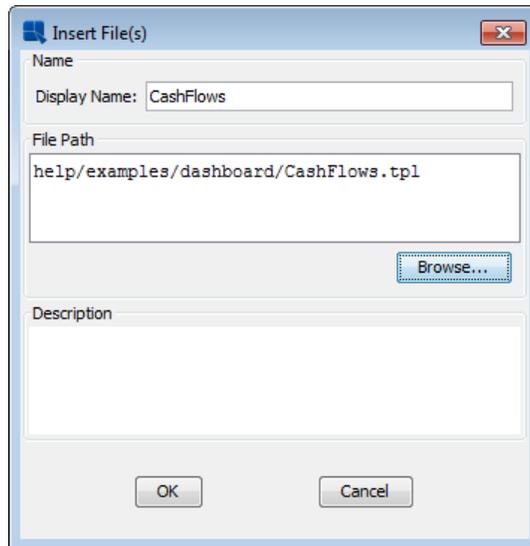
## 2.1.4. Files

You can insert files of any type into projects and folders. However, if you want your files to be accessible to other users or visible within the Menu, then files should be viewable within a web browser (i.e. HTML pages, image files, etc.), or be chart, report, map or dashboard files (CHT, TPL, RPT, EZR, PAK, PAC QPT, QCH, DSB, GXML, SXML, CXML or XML format). Files added to the Organizer must be accessible via http protocol, and may need to be mapped via URL mapping. For more about URL mapping in Organizer, see Section 2.1.5 - URL Mapping.

### 2.1.4.1. Adding and Modifying Files

To add a file, first select the folder or project in which you would like to place the file. Then you can either click the *Insert File* button on the toolbar, select *Insert File* from the Edit menu, or right click in the upper right-hand pane, and select *Insert File* from the pop-up menu. A dialog box will appear, prompting you to enter a display name for the file (by default the file name is used), the location of the file (a direct path) and a description of the file. (If you select a path that is not the server root or a mapped virtual directory, you will be prompted to map one.) Click *Ok* and the file will be inserted into the current directory or folder.

To insert more than one file at a time, click the *Browse* button on the insert file dialog, and use **CTRL+Click** to select multiple files in the browse dialog. Then when you click *OK* all of the files will be added to the Organizer.



*Insert File Dialog*

The list of files in the currently selected folder will be displayed in the upper right-hand pane of the Organizer. Each entry shows the file's display name, the file type, the file description, path, and URL.

To edit the file's display name, location, or description, first select the file that you would like to edit. Once you have selected a file, you can either click the *Edit* button on the toolbar, select *Edit* from the Edit menu, press *F2* key, or right click, and select *Edit* from the pop-up menu. A dialog box will appear allowing you to make changes to the display name, location, and description. Click *Ok* and the changes will be reflected in the file list.

To open a file, double click on it, or first select it, and then you can select *Open File* from the *File* menu, or right click and select *Open File* from the pop-up menu. If the file is a chart (PAC, CHT, TPL, or QCH) format, it will be opened in the Chart Designer. If the file is a report (PAK, RPT, QPT, or XML) format, it will be opened in the Report Designer. If the file is a dashboard, the Dashboard Builder interface will open in a new window. If the file is a map, Online Maps interface will open in a new window. All other file types, such as SVG, will be opened in a web browser window.

### 2.1.4.2. Arranging Files

To arrange files within folder or project, open the folder (click on it in the left-hand side tree-list) and then click on a column header in the right-hand side. Files in the folder/project will be arranged by the content of the selected column. To switch between ascending and descending order, click on the column header again.



*A folder arranged by the file names in descending order.*

### 2.1.4.3. File Identification Icons

Each file type can be easily identified by its icon. You can see these icons next to file names in Organizer and all dialogs in ERES designers where you need to select some file.



Report file created in Report Designer that contents supplementary files associated with it (.pak). For more information see Section 4.1.5.1 - Saving Reports.



Report template created in Report Designer (.rpt) in older versions of ERES (6.6 and older).



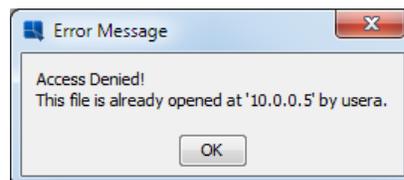
Chart file created in Chart Designer (.cht). For more information see Section 4.2.2.3.1 - Saving Chart Definitions.

-  Chart template created in Chart Designer (.tpl). For more information see Section 4.2.2.3.1 - Saving Chart Definitions.
-  Chart file created in Chart Designer that contents supplementary files associated with it (.pac). For more information see Section 4.2.2.3.1 - Saving Chart Definitions.
-  Report file created in QuickDesigner Reports (.qdr). For more information see Section 4.3 - QuickDesigner Reports.
-  Report file created in QuickDesigner version 6.6 and older (.qdr). This file is not possible to edit in QuickDesigner Reports, only in the old QuickDesigner. How to use QuickDesigner 6.6 see Section 1.4.1.2 - Setting Info.
-  Chart file created in QuickDesigner Charts (.qch). For more information see Section 4.4 - QuickDesigner Charts.
-  Map file created in Online Maps (.gxml). For more information see Section 5.2 - Online Maps.
-  Packed map file created in Online Maps and packed in Organizer (.gpak). For more information see Section 2.1.4.5 - Dashboard and Map Packages.
-  Coordinates file for Online Maps (.cxml). For more information see Section 5.2.2 - Data Sources.
-  SVG map file created in SVG Maps (.sxml). For more information see Section 5.3 - SVG Maps.
-  Packed SVG map file created in SVG Maps and packed in Organizer (.spak). For more information see Section 2.1.4.5 - Dashboard and Map Packages.
-  SVG map image (.svg). For more information see Section 5.3.1 - Introduction to SVG Maps.
-  Dashboard file created in Dashboard Builder (.dsb). For more information see Section 6.1 - Introduction to Dashboards.
-  Packed dashboard file created in Dashboard Builder and packed in Organizer (.dpak). For more information see Section 2.1.4.5 - Dashboard and Map Packages.

### 2.1.4.4. Report Locking

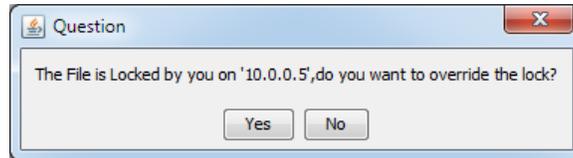
The ERES Organizer has an automatic report locking system designed to block multiple users from editing the same report in Report Designer at the same time. The system utilizes your IP address, your username, and a special token to determine whether you are allowed to open a specific file. Here is an example illustrating how the advanced file locking system operates.

Suppose you are logged in as userA and you are editing a report `Sales.pak`. One of your colleagues logs in as userB and attempts to edit the same `Sales.pak`. She would see the following error message, alerting her that the file is currently locked.



*File is Locked*

Suppose after opening the file, your browser crashed or you moved to a different computer. Now you log back in as userA and open the same file again. You will see the following message allowing you to override your own lock.



### Override Your Lock

If you moved to another computer and the report is still opened on the previous computer, you will not be able to save on the previous computer. You must close the report and re-open it in order to have write privilege again.



### Note

The admin has the ability to overwrite any lock.

## 2.1.4.5. Dashboard and Map Packages

A single map or a single dashboard usually consists of several files (for example, a dashboard needs several chart/report/map templates, a svg map needs a svg image, drill-down template, etc) which could make moving maps or dashboards to another machine very difficult, because you would have to move all the linked files as well. Fortunately, you do not have to move each file manually. A dashboard or a map and can be packed into a single file along with all the linked files (dashboard templates, drill-down templates, etc) without even knowing which exact files are needed for the dashboard or map to work. The packed files can be moved or archived quickly and easily.

To pack a dashboard or a map (both SVG and Online maps can be packed), select the dashboard (.dsb) or map

(.sxml or .gxml) file in Organizer and then click on the  *Create PAK* icon on the main Organizer toolbar. A dialog will pop up prompting you to select the pack file name. Enter a file name or use the default one and click *OK*. Another dialog should show up confirming that the package was created successfully and showing you the full file path. The packed file will be also automatically inserted into the current Organizer folder/project.

Packed files can be viewed in the Menu page without being unpacked, but they can not be modified in the SVG and Online Maps designers or in the Dashboard Builder. In order to modify a packed dashboard or map, you need to unpack the gpak, spak, or dpak file first.

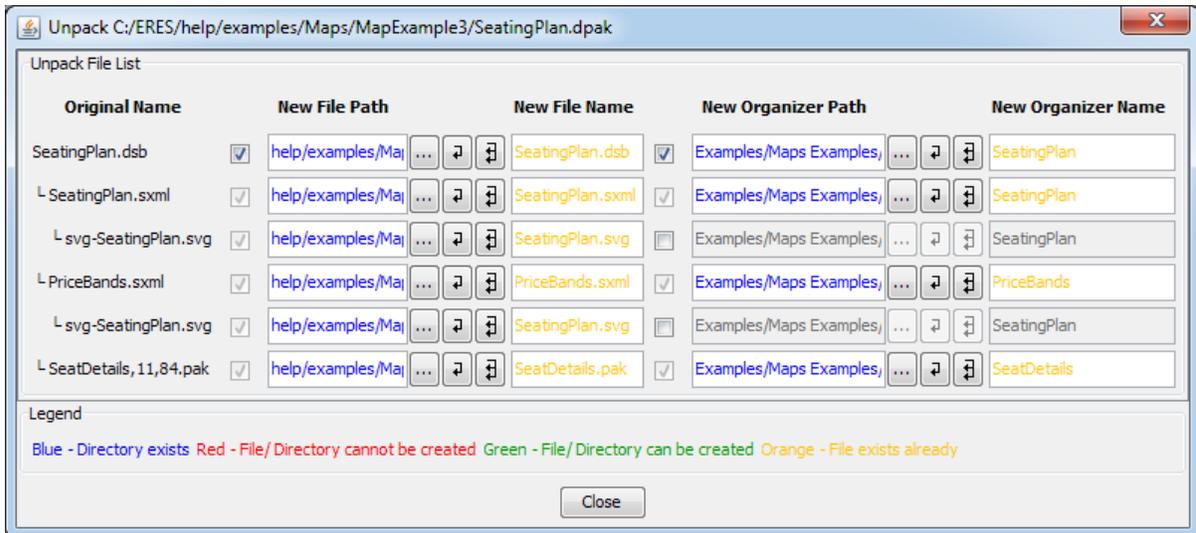
To un-pack a dpak, spak, or gpak file that has not been inserted into Organizer, click on the  *Unpack File* icon on the main Organizer toolbar. A dialog will pop up, allowing you to choose the file on your hard drive and to enter a name which will be used for the main unpacked file (.dsb, .sxml or .gxml file).

If you want to unpack a file that has been inserted into Organizer, select the file in Organizer and then click on the



*Unpack File*. The same dialog will pop up but the file path and the file name fields will be filled automatically.

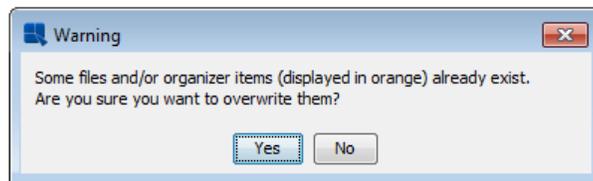
To view the list of all files that are stored in the package, click on the *Details* button. A new window will open.



*Unpack File List Dialog*

In this window, you can edit both physical and Organizer file path of all files that can be un-packed from the package (which also allows you to resolve potential file name conflicts). After you are done editing the file paths, click the *Close* and then click *OK* to unpack the package.

The ERES will now try to unpack all files from the packed file to their original locations. If there are any filename conflicts (i.e. there already is a file with the same name in the same path on your hard drive or in Organizer), the following dialog will show up:



If you click *Yes*, all existing files will be overwritten by the files from the package without no further questions. If you click *No*, the *Unpack File List* dialog will open again allowing you to edit the filename conflicts. Then click *Unpack* to unpack the files into their locations and to insert them into Organizer.

## 2.1.5. URL Mapping

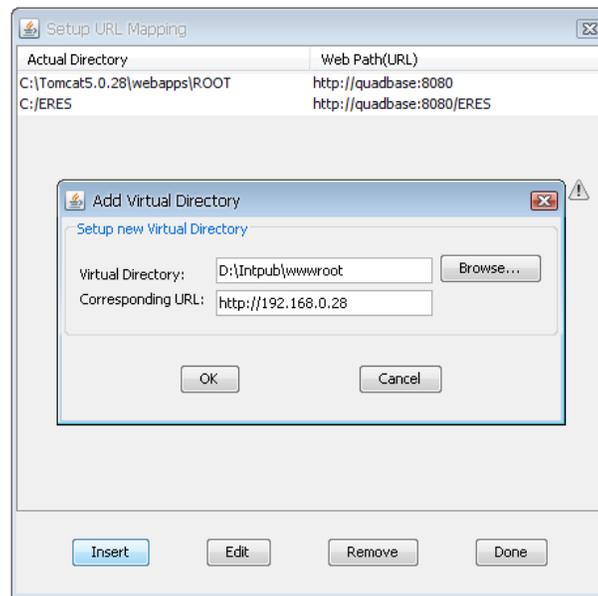
Any report or chart that is added to the Organizer needs to be accessible via URL. This is so that reports can be run/viewed through the menu page, using URLs, or retrieved through the API. In order for the Organizer to know the URL path to charts and reports, you need to set the URL mapping.

The first, and most important, thing to map is your ERES installation. You should have a virtual directory set-up in the app server/servlet container where you have deployed the server that points to your installation. If you selected to install Tomcat with ERES then a virtual directory has automatically been setup where `http://MachineName:Port/ERES` points to the ERES installation directory. In order for Organizer to read and write files to your installation directory you will need to make sure that this mapping is correctly set.

URL mapping can also be used to set the URL path for other servers or virtual directories on your server whose report/chart files you would like to import to the Organizer. Please note that only the admin can create and change the URL Mapping entries. You will have to log in as the admin user to make changes to the URL Mapping.

To set up a virtual directory, select *Set URL Mapping* from the Option menu. A dialog will appear listing all of the virtual directories that have been set up. (By default, the Web root for the app server/servlet container where you have deployed the server is mapped to the URL `http://machinename:port/` depending on your machine name and specified port number. To add a virtual directory, click on *Insert*. A dialog will appear allowing you to

enter or browse to a direct path. Once you have specified the path, then enter the corresponding URL. Click *Ok* and the virtual directory will be added. You can also edit and remove mapped URLs from the main dialog.



*URL Mapping List & Add Mapping Dialog*

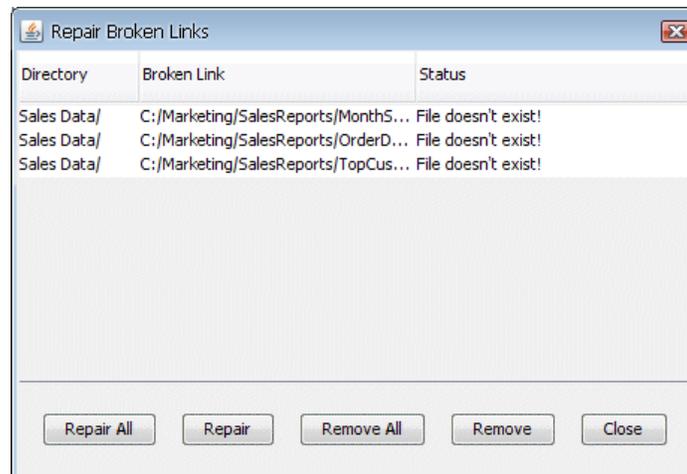
You will also be prompted to specify URL mapping any time you add a file from a new location that has not been mapped with a corresponding URL.

## 2.1.6. Repairing Broken Links

Because Organizer uses virtual directories, it is possible to have files in many different physical locations listed in the Organizer. While this makes it easy to publish reports and charts, it can be difficult to manually keep track of all the files. Specifically, if the files are moved or the directory structure is changed, then the Organizer will not be able to find the files and the links within the Menu page may no longer work.

You can use the *Repair Broken Links* feature in the Organizer to automatically check the status of the listed files. The Organizer will check all of the files within your projects and folders to check whether the path/URL specified is still correct.

To check for broken links, select *Repair Broken Links* from the Option menu. The Organizer will then check all of your files and pop up a list of any broken links found. You can then choose to repair any broken links by specifying a new path or file/filename, or you can remove the files from the Organizer.

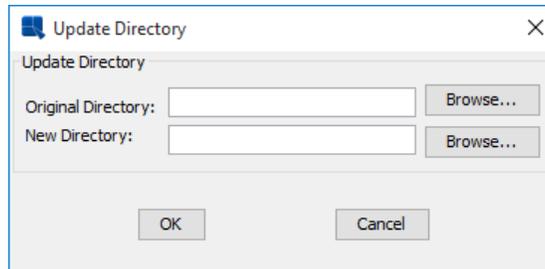


*Broken Links List*

## 2.1.7. Update Directory

Sometimes you may want to update the directory path of certain items in the Organizer. The *Update Directory* feature is helpful if you are moving files between installations or if you are moving complete installations.

In order to update the directory paths of items in the Organizer, select *Update Directory* from the Option menu. The update directory dialog will then open.



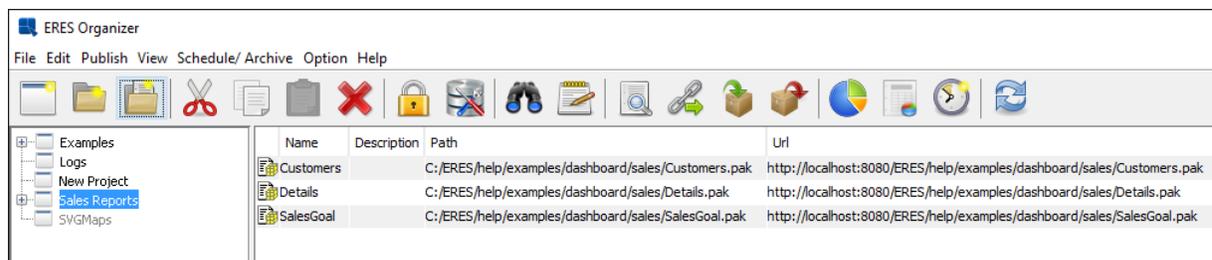
*Update Directory Dialog*

The dialog allows you to specify the original directory (the directory you want to update) and the new directory. You can either use the *Browse* buttons to select directories or just type directory paths manually.

Please note that when you are updating directories using the update directory feature, all the Organizer items that contain the original path in their directory paths will be updated. For example, if you specify the original path as `C:/ERES` and the new path as `D:/Install/ERES`, all the items in the Organizer that contain `C:/ERES` in their paths such as (`C:/ERES/Reportfiles/SalesReport.pak`, `C:/ERES/ChartFiles/MonthSales.cht`, `C:/ERES/DashboardFiles/Dashboards/SalesDashboard.dsb`, ...) will be updated to (`D:/Install/ERES/Reportfiles/SalesReport.pak`, `D:/Install/ERES/ChartFiles/MonthSales.cht`, `D:/Install/ERES/DashboardFiles/Dashboards/SalesDashboard.dsb`, ...). Also, note that the new directory (the updated directory) needs to be added to your URL mapping. Otherwise, you will not be able to repair broken URL links after the update, so the affected Organizer items will not work correctly. For more information about URL mapping and repairing broken URL links, please see Section 2.1.5 - URL Mapping and Section 2.1.6 - Repairing Broken Links.

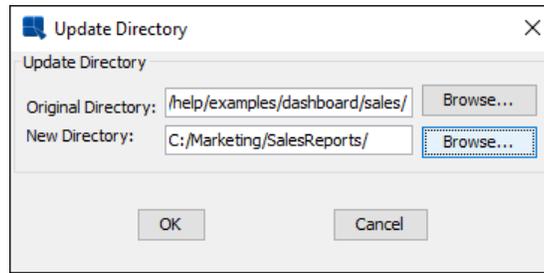
### Example:

Imagine that you have three reports `Customers.pak`, `Details.pak`, and `SalesGoal.pak` in the `C:/ERES/help/examples/dashboard/sales` directory, and assume that you have moved them to the `C:/Marketing/SalesReports` directory. The following image shows the reports in the Organizer before the updating process:



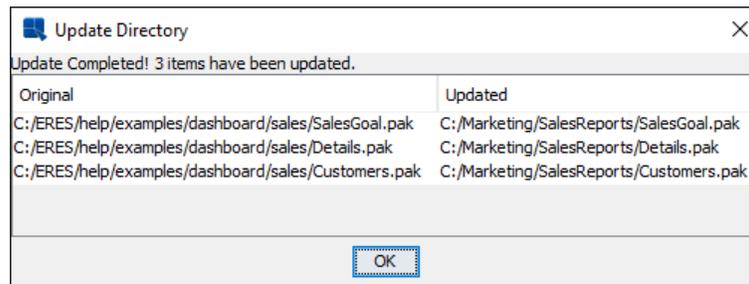
*Example - Reports in the Organizer - Before Directory Updating*

Select *Update Directory* from the *Option* menu in the Organizer to open the update directory dialog. In the dialog, specify the original directory as `C:/ERES/help/examples/dashboard/sales` and similarly the new directory as `C:/Marketing/SalesReports`.



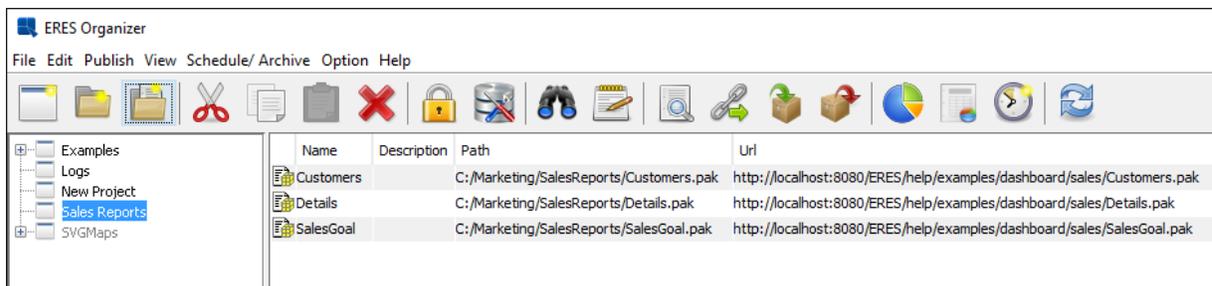
Example - Update Directory Dialog

Once you have properly specified the directories, press the *OK* button. The following dialog will then appear showing you the list of updated Organizer items as well as number of updated files.



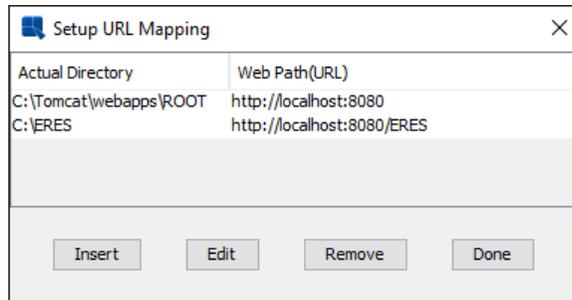
Example - Update Results Dialog

Next, click the *OK* button in the dialog. This will bring you back to the Organizer. The image below shows updated reports paths after the updating process.



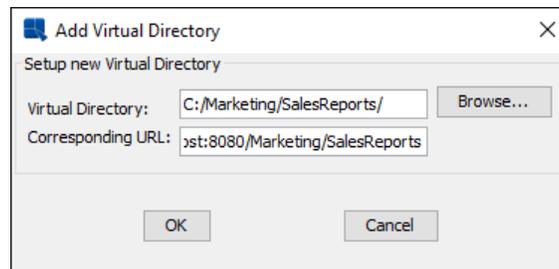
Example - Reports in the Organizer - After Directory Updating

Now you have to repair the broken URL links. Otherwise, URLs of the affected Organizer items will remain mapped to the original directory. In order to repair broken URL links properly, you have to add the new directory (the directory you have specified in the Update directory dialog C: /Marketing/ SalesReports) as new URL link to your URL mapping. To open the URL mapping dialog, select *Setup Url Mapping* from the *Option* menu in the Organizer. The dialog will then appear.

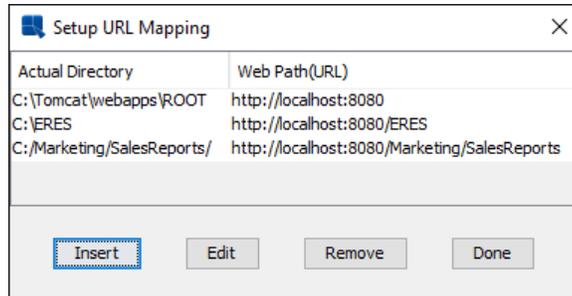


Example - Setup URL Mapping Dialog

In order to add a new URL link, click the *Insert* button, another dialog will pop-up prompting you to specify the *Virtual Directory* and *Corresponding URL*. Specify the virtual directory as `C:/Marketing/SalesReports/` and the corresponding URL e.g. as `http://<MachineName>:<Port>/Marketing/SalesReports`. After that, click *OK*. For more information about URL mapping, please see Section 2.1.5 - URL Mapping.

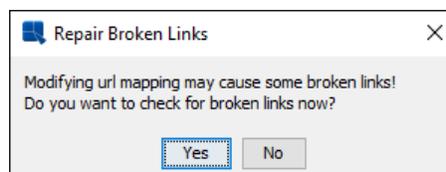


Example - Inserting New URL Link



Example - Setup URL Mapping Dialog - New URL Link Added

Once you have specified the URL link, click the *Done* button again. The repair broken links dialog will then appear prompting you whether you want to check for broken links now. Click *Yes*. Please note that if you click *No*, you may check for broken URL links by selecting *Repair Broken Links* from the Option menu in the Organizer. For more information about repairing broken URL links, please see Section 2.1.6 - Repairing Broken Links.



Example - Repair Broken Links Dialog

If you have correctly specified the URL mapping, you should get the following dialog that will inform you that broken links have been successfully fixed. Clicking the *OK* button will bring you back to the Organizer.

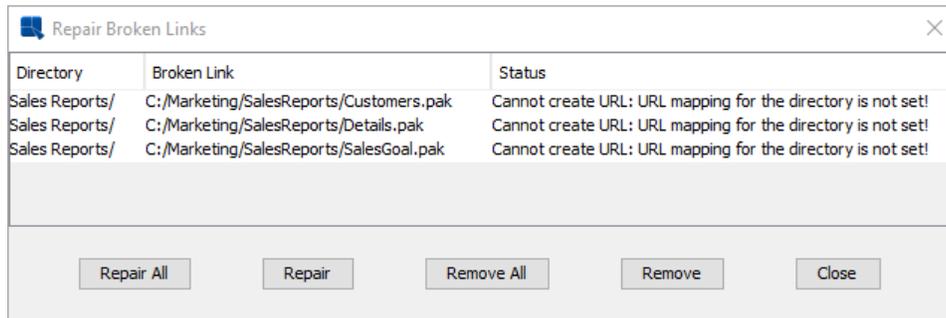


Example - Repair Broken Links - Links Successfully Fixed



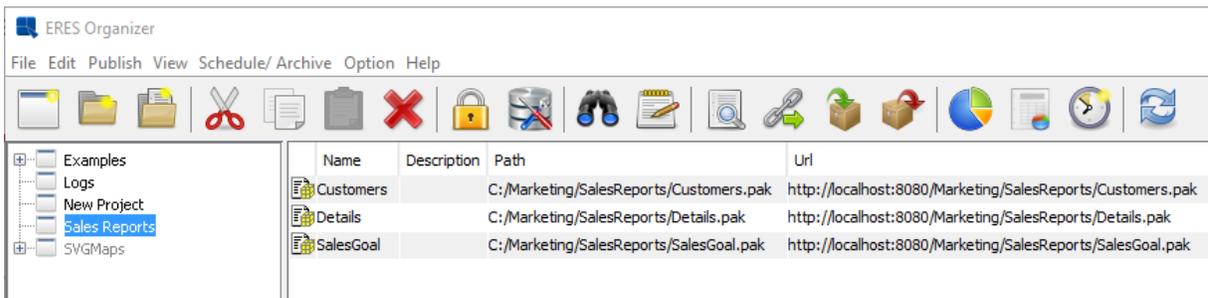
**Note**

If you did not specify the URL link in the URL mapping or did not specify the URL link correctly, the dialog below will appear informing you that broken URL links cannot be repaired. If you got this dialog, return back to the URL mapping dialog and add/modify the URL link as described in previous steps. After that try to repair broken links again.



Example - Repair Broken Links - Error During Links Fixing

Now you may check whether the URL fields of the organizer items have been correctly repaired. The URL fields should be similar as the ones shown on the image below.



Example - Reports in the Organizer - After Repairing URL Links

## 2.1.8. Searching in Organizer

Often, if you have many files in many folders in the Organizer, it can be hard to find a specific file. To locate files



or folders, you can use the search function. To search, you can either click the *Search* button on the toolbar, select *Search* from the View menu, press **Ctrl+F**, or right click in the upper right-hand panel and select *Search* from the pop-up menu. The search dialog will then appear.

*Search Dialog*

The search dialog prompts you to enter a search string in the project or folder in which you would like to search. By default, the currently selected project or folder is listed. If you leave the *Look In* field blank, it will search all projects.

Other options allow you to include sub-folders in your search, and to match the search string to filenames, descriptions, and URLs. You can also match case in your search by selecting the *Case Sensitive* option.

## 2.1.9. Limiting Browse Directories

In ERES, you can limit the open and insert dialogs in Organizer, Report Designer, and Chart Designer by adding a parameter to the Organizer. If launching the Organizer through the Java Web Start Launcher, edit `/ERESOrganizer.jnlp` and add the following line:

```
<argument>-BrowseRootDir:<Directory></argument>
```

For instance, the follow limits the browse directory to the ERES root directory located under `C:\ERES`

```
<argument>-BrowseRootDir:C:\ERES</argument>
```

If starting the Organizer using the bat file, edit `ERESOrganizer.bat` and add the following option at the end of the command.

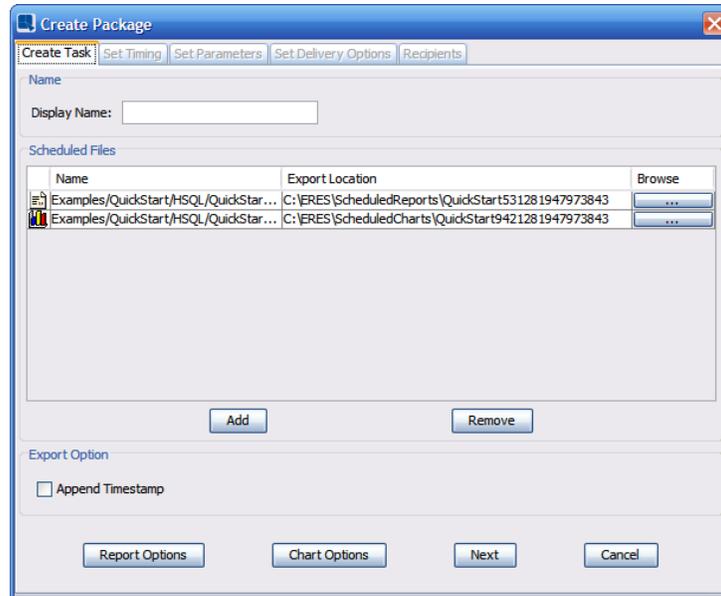
```
... quadbase.reportorganizer.manager.OrganizerClient -  
BrowseRootDir:<Directory>
```

## 2.2. Scheduling & Archiving

ERES is capable of scheduling/archiving any chart (`.pac`, `.cht` or `.tpl`) or report (`.pak`, `.rpt` or `.xml`) file that is placed in the Organizer. A schedule will generate an exported file for the report or chart at a regular interval specified by the user. An archive will generate a report or chart template with the current data stored in the template. The periodicity of schedule/archive can be set for specific time intervals, or for fixed days of the week or dates of the month.

### 2.2.1. Setting a Schedule

To schedule a chart or report, first, select the file in the Organizer that you would like to schedule. You can also select multiple files at once. Then click the *Set Schedule* button on the toolbar or right click on the selected file and select *Set Schedule* from the pop-up menu. The first dialog of the scheduling process will open.



*Create task dialog*

On this Wizard step, enter a *Display Name*.

In the *Scheduled files* list, you can see what files are being scheduled. You can also remove a file from the scheduled task or add a file.

The dialog has following objects:

**Display name:** scheduled task name.

**Scheduled files:**

- **Icon** - indicates the type of scheduled file.



- Chart



- Report

- **Name** - name of the file
- **Export location** - directory path. The file will be exported to the location at the time when the task is run. You can double click this field to alter the path manually.

You can access the exported file from MenuPage by clicking on the  *Schedule* icon (as described in Section 7.1.2.1 - Viewing Reports and Charts ).

- **Browse** - clicking the “...” field will bring up a browse dialog allowing you to change file export location. This has the same effect as double clicking the *Export location* field, only this option allows you to browse file structure.

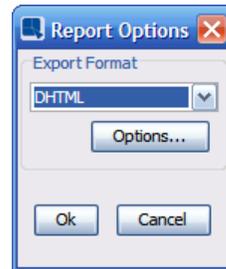
**Add** - this button allows you to add more files into the scheduled task.

**Remove** - removes selected file from the scheduled task.

**Append timestamp** - appends current time (i.e. time when the task is run) to the exported file's name. Also, if timestamp is appended to filename, the filename is always unique thus every time when the task is run, a new file is added into the export folder. If you disable the *Append timestamp* option, only one exported file will be kept in the export folder and it will be overwritten every time the task is run.

**Report options -**

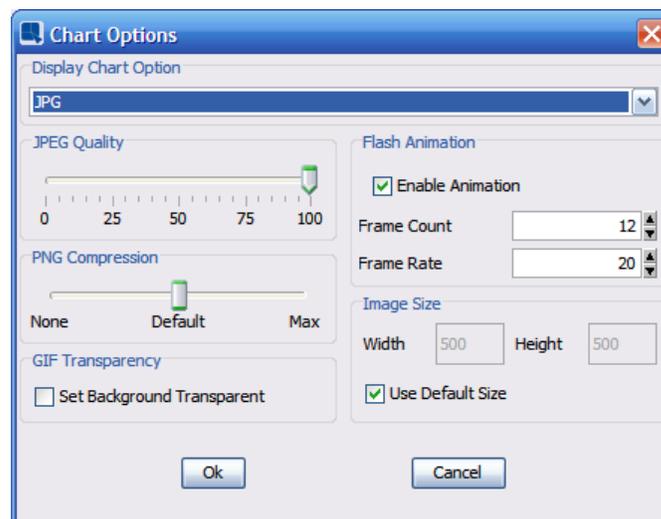
shows dialog allowing you to set report export format. For reports you can select DHTML, PDF, CSV, Excel (XLS), Excel 2007 (XLSX), text, rich text, or XML. Additional options allow you to select single page or multi-page exporting for DHTML and XML. The *Options...* button allows you to set format-specific options for the exported file. For DHTML you can set single or multi-page exporting. For DHTML you can also specify which CSS options to use. For PDF you can set encryption and for text you can specify the delimiter. For more information about report exporting options please see Section 4.1.5.2 - Exporting Reports.



*Report Options Dialog*

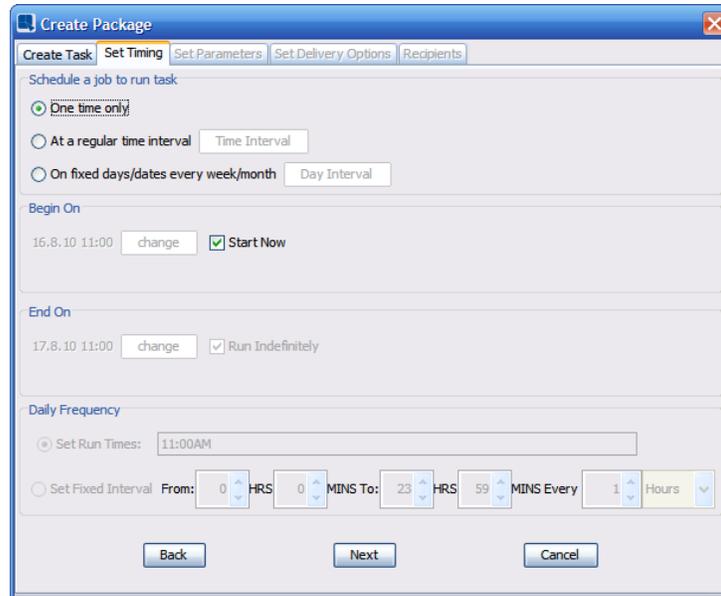
**Chart options -**

shows dialog allowing you to set chart export format. For charts you can select JPG, GIF, PNG, PDF, SVG, FLASH, or EXCEL. Other options allow you to specify PNG compression, set JPEG quality, and specify background transparency for GIFs. You can also enable and configure Flash animation if the image depiction should be animated. You can also specify the size of the exported image. For more information about chart exporting options please see Section 4.2.6.3 - Exporting Charts.



*Chart Options Dialog*

After you are done setting the first tab options, you can click *Next* button and proceed to the *Set timing* tab.



*Set Timing Dialog*

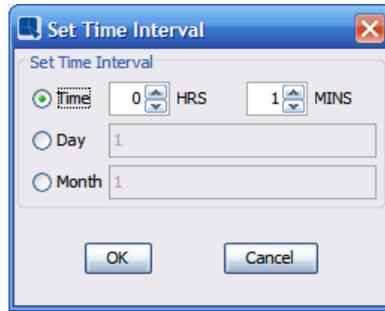
Set timing tab allows you to set a time pattern of the scheduled task.

You have following timing options:

- One time only:** task is run one time only at pre-defined time. You can choose when the task will be run using the *Begin on* option (will be described later).
- At regular time interval:** scheduled task will be run at fixed intervals for a pre-defined period of time. For example, you can set the task to be run every X hours (or minutes, days, months...) from *Begin on* to *End on* time and date. When you select this option, the *Set time interval* dialog pops up. This dialog will be described later.
- On fixed days/dates every week/month:** If you need to run your task irregularly, this option should do the trick. It allows you to run the task on certain week day(s) (for example, every Monday and Friday), or certain days of the month (for example, every the first of each month). This is set in the *Set fixed days/dates Interval* dialog which pops up when you select the option. Also, when you select this option, the *Daily frequency* section is enabled. These two dialogs will be described later.

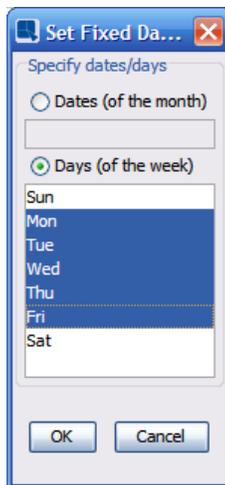
You also need to set when the task should start running and when it should stop. You can do this using the following options:

- Begin on:** allows you to set when the task will start running. By default, the *Start now* option is enabled. It means that the task start time and date will be set two minutes ahead of the time when the creation of the schedule job is finished. If you do not want the task to start immediately, uncheck this option to pick the start time and date from a calendar. You can access the calendar by clicking the *Change* button (which is available only when the *Start now* option is disabled).
- End on:** controls when the task will end. This option is very similar to the *Begin on* option. The only difference is that the *Start now* option is replaced by *Run indefinitely* option. If you choose this option, the task will never stop unless you stop it manually. You can also unselect this option and pick the *End on* time and date manually from a calendar.



*Set Time Interval Dialog*

This dialog is enabled only if the *At regular time interval* option is selected. First of all, choose a time unit (namely, hours and minutes, days or months) that suits your needs. Next, enter a number of the time units that should be waited for between each task runs.



*Set Fixed Days/Dates Dialog*

This dialog is enabled only if the *On fixed days/dates every week/month* option is selected. First, choose whether you want to run the task on a weekly basis or on a monthly basis. Next, select days of the week or month task should be run on. After you are done setting this dialog, click *OK* and proceed to the *Daily frequency* settings.

The screenshot shows the 'Create Package' dialog box with the 'Set Timing' tab selected. The 'Daily Frequency' section is highlighted, showing two options: 'Set Run Times' (selected) with a text input field containing '03:09PM', and 'Set Fixed Interval' with a 'From' field set to '0 HRS 0 MINS', a 'To' field set to '23 HRS 59 MINS', and an 'Every' field set to '1 Hours'. The 'Begin On' and 'End On' sections are also visible, with 'Start Now' and 'Run Indefinitely' checked.

*Daily Frequency Settings*

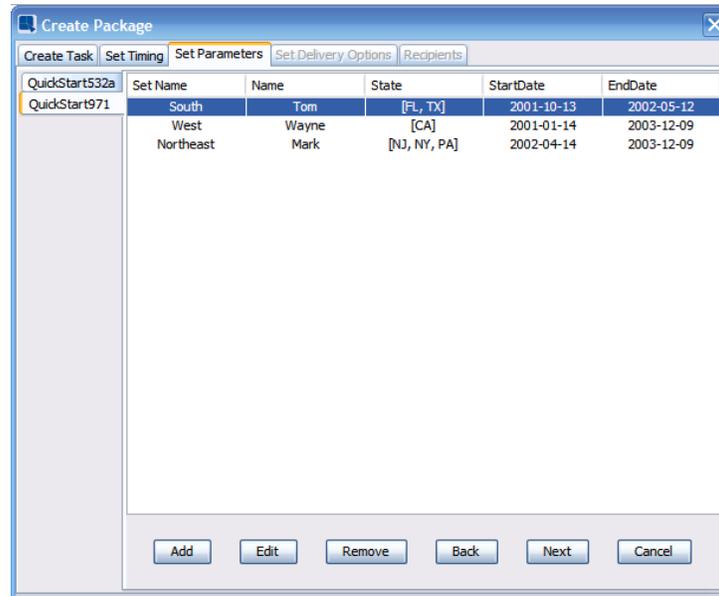
After you have set the days of the month/week you want the task to be run on, you have to set when exactly should it be run on those days. That can be done in the *Daily frequency* section. You have two basic options:

- Set run times -** allows you to manually enter the run times. You can add as many values as you want. You can use 12-hour or 24-hour format. Individual values are comma separated. **For example:** 08:05AM, 11:15PM, 22:35
- Set fixed interval:** use this option to run the task periodically from the *From:* time to the *To:* time every X hours/minutes. To set the times, you can use up and down arrows or you can enter the value manually (click in the field and type a number). If you enter an incorrect time (for example, 11:65), the value will be marked with red color. Correct values are marked with green color.

Once you have finished specifying the periodicity, click *Next* to continue with the Schedule Wizard.

### 2.2.1.1. Scheduling Parameterized Reports/Charts

If you have selected to schedule a report or chart that contains parameters, the next tab will appear allowing you to set the parameter values that you would like to use.



*Set parameters dialog*

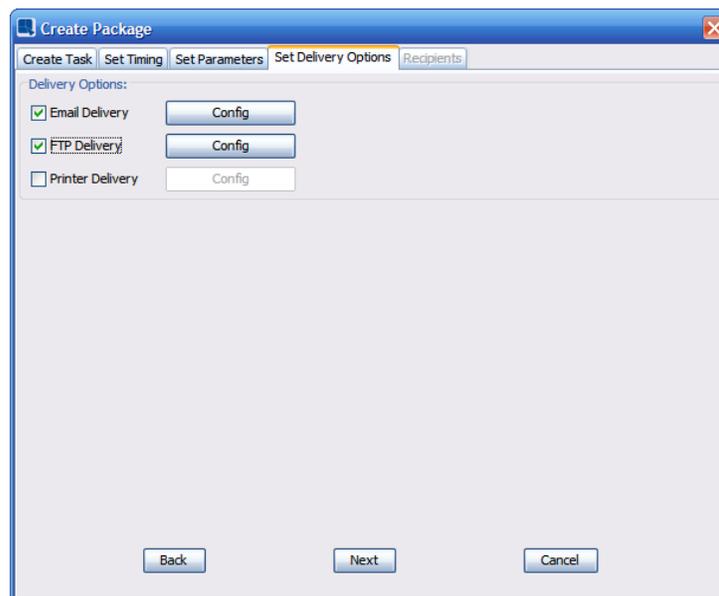
To add a set of parameters, click the *Add* button. This will bring up the parameter prompt dialog allowing you to select the set parameter values that you would like to use. Once you have selected a parameter set, your choices will appear in the dialog. You can add as many different combinations of parameter sets as you want. A separate file will be generated for each parameter set.

You can also specify a name for each parameter set by double clicking on the first column. The name specified here will be used in later dialogs to help you organize your recipients.

Once you have added all parameter sets, click the *Next* button to continue.

### 2.2.1.2. Additional Delivery Options

After you have finished setting up the schedule, you will be prompted to select delivery methods. These options allow you to email the generated files, print the exports, and upload the exported files to a FTP server. You do not have to select any additional delivery options.



*Delivery Options Dialog*

If you select a delivery option, click the *Config* button to configure the delivery method.

### 2.2.1.2.1. Email Delivery Options

When you click the 'Config' button for email delivery the following dialog will open allowing you to set an email message:

*Email Settings Dialog*

The first tab contains information for the email sent when the report or chart has successfully exported. The *General Email Information* portion of this tab allows you to specify the from addresses for the email as well as the subject.

The first tab is called *Successful email* and it allows you to set an email message which will be sent in case the scheduled task finished with no problems.

There are the following elements:

**From:** enter an email address into this field. This address will be used as an email sender's address. Please make sure that your SMTP server will not reject the address (depends on your SMTP server security settings).

**Subject:** email subject. Enter any convenient text in here.

**Email Body Text:** enter a message that will be sent on every task run. You can enter any text and also, you can use *Runtime variables* (will be described later).

**Attach exported files:** allows you to attach all exported reports and charts to the email.

**As HTML:** if this option is enabled, the email body text will be replaced by the scheduled report or chart in HTML format. This option is available only if the *Attach exported files* is enabled and there is only one scheduled report/chart in the scheduled task. Also, PDF, SVG and Flash charts cannot be added into HTML emails and only DHTML reports can be added into HTML emails. You can set the export format on the *Create task* tab, as described in Section 2.2.1 - Setting a Schedule.

**Insert Runtime Variables -** allows you to insert certain variables into the email body text. Variables are replaced by real values at the task run time. All variables also have a "default value". If no convenient data can be inserted into the variable, the default value is used (for example, when user has not provided his/her name). You can choose between following variables:

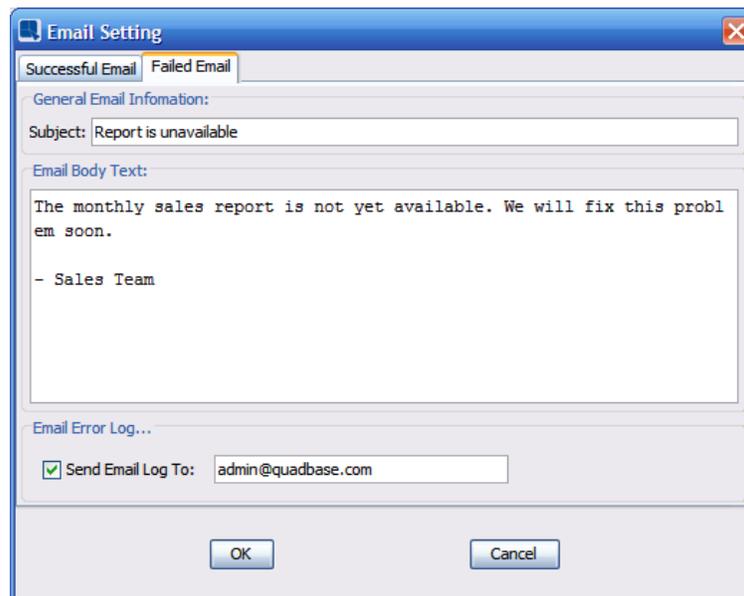
<USERNAME> -	recipient's user name. For example: you can use this variable in a salutation like “Hello <USERNAME>”. This variable works only for email recipients that were chosen from the ERES database (will be described later). If you add an email address to recipients list manually, no user-name is displayed because no username was provided.
<FULLNAME> -	recipient's full name. Just like the <USERNAME> variable, this variable also does not work for manually entered recipients email addresses.
<EMAIL> -	recipient's email address.
<TIMESTAMP> -	current time and date. Shows when the email was sent.
<PARAMETERS> -	list of all parameter sets and parameter values.
<SAVED_LINKS> -	links of all scheduled files. Users can access the exported files through these links.
<LIVE_LINKS> -	links of all scheduled files. Unlike the <SAVED_LINKS>, these links lead to the DHTML Viewer which allows users to access current version of the scheduled files.



### Note

In order to send email correctly, the SMTP host needs to be set up. How to configure a SMTP server is described in Section 1.4.1.2 - Setting Info.

The *Failed Email* tab allows you to send a different email (subject and body) to your recipients if an error has occurred during the task running (for example: one of the scheduled reports has been removed).

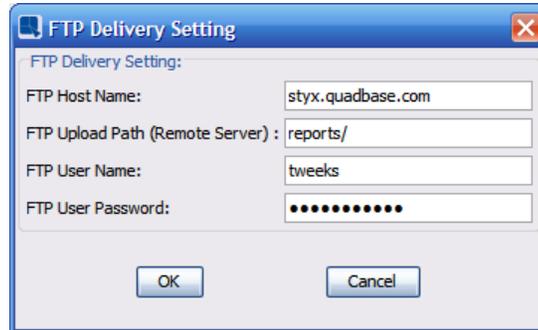


*Failed email settings dialog*

This tab also allows you to send the error logs to a specific user. Typically, this would be the admin or technical user who is able to diagnose and troubleshoot the problem.

### 2.2.1.2.2. FTP Delivery Options

When you click the *Config* button for FTP delivery, the following dialog will open, allowing you to set options for FTP schedule delivery.



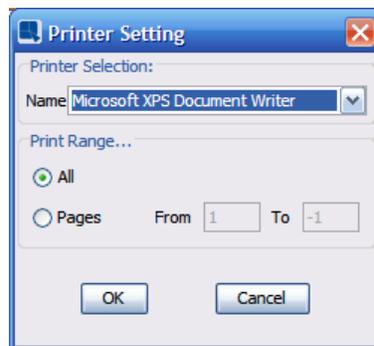
*FTP Delivery Options Dialog*

The following information is required for FTP schedule delivery:

- FTP Host Name:** This is the host name of the FTP server to which you would like to upload the reports. If you have a Secure FTP Server (SSH FTP), enter **sftp://** in front of the hostname.
- FTP Upload Path:** This is the path or directory (relative to the FTP root) where you would like to write the exported report files.
- FTP User Name:** This is the user name for the FTP server.
- FTP Password:** This is the password for the FTP server.

### 2.2.1.2.3. Printer Delivery Options

When you click the *Config* button for printer delivery, the following dialog will open allowing you to set options for the print schedule delivery.

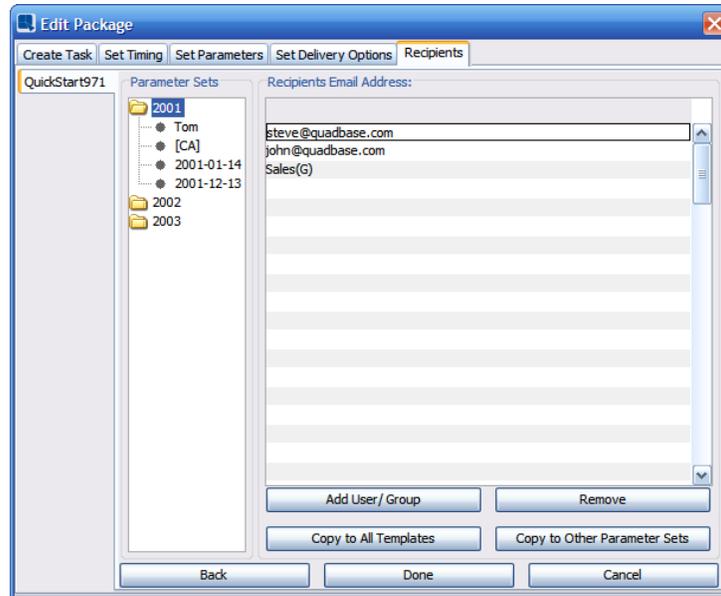


*Printer Delivery Options Dialog*

The first drop-down dialog contains a list of all the available system and network printers. You can select which one you would like to use. The second option allows you to select a page range for printing. You can print all of the pages or set a specific range. -1 represents the last page. **Example**, if you want to print all of the pages except for the first one, set the range to *From 2 , To -1*.

### 2.2.1.3. Specifying Email Recipients

If you selected the email delivery, the next tab allows you to specify the recipient list. The recipient list is set up differently depending on the type of report or chart you are scheduling. If the report or chart contains parameters, you will see the following dialog. This allows you to send different parameter sets to different users or groups.



*Set Recipients Dialog for Parameterized Reports*

Entering recipients into the list is simple, click on the *Add User/Group* icon to select users and groups from the ERES database. You can also add regular email addresses by double clicking any row in the recipients list and typing it in manually. You may also click on *Add User/Group* and view the *Prev. List* tab for the previously listed recipients.

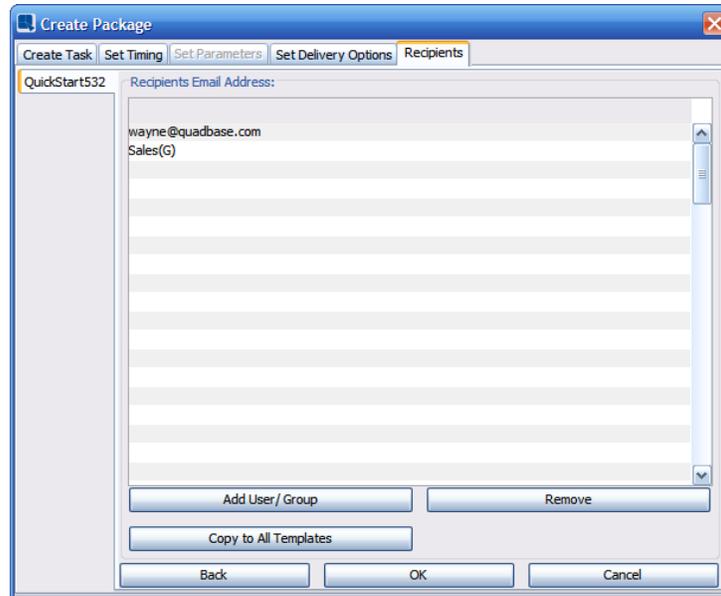
If you want to send all parameter sets or all templates to the same recipients, you do not have to add the recipients manually into all parameter sets and templates. Just add the recipients into one recipients list and then use one of the following buttons:

**Copy to all templates:** click this button to copy current recipient list into all scheduled templates (reports, charts) recipient lists.

**Copy to other parameter sets:** allows you to copy current recipient list to the rest of the parameter sets.

If you are scheduling a grouped report (such as a summary break report) and the report is using a database as a datasource, you will see the option for report bursting as well. Screenshots and detailed information for report bursting can be found in the Section 2.2.1.3.1 - Report Bursting.

For all other cases, you will see the following dialog containing only the recipients list.

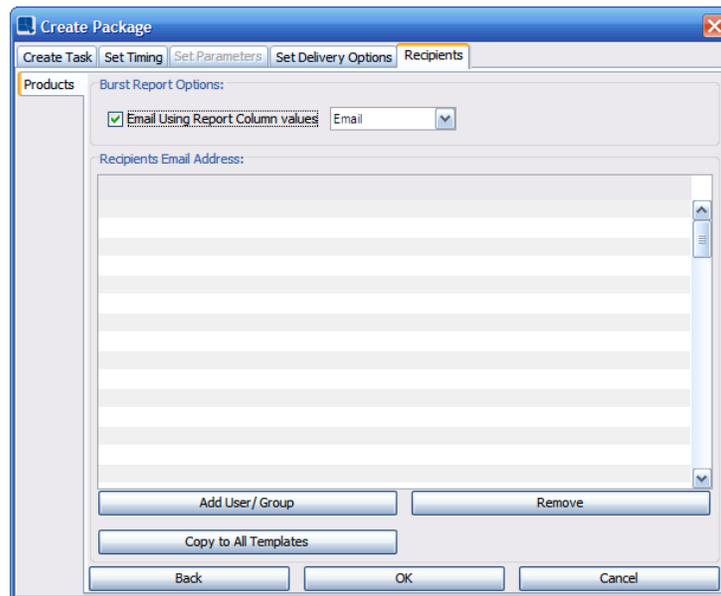


*Set Recipients Dialog*

### 2.2.1.3.1. Report Bursting

If the scheduled report contains grouped data (like a summary break report), you can enable “report bursting”. When a report is burst, the main report is divided into smaller reports and each group report is then sent to a different recipient. The recipients are taken from a database column so you do not have to enter them manually for each group of data. This feature offers significant performance and security improvements over running many small reports. The report burst option is available only for database data sources (MySQL, MS SQL, Oracle etc...).

If the report contains grouped data, the recipients dialog will contain a bursting option at the top.



*Report Bursting*

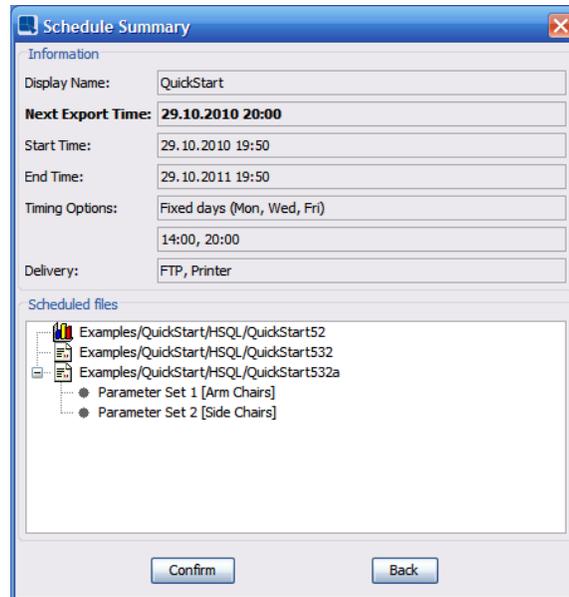
To set up the burst report feature, the report has to have a column that contains email addresses that correctly correspond to the group by (row break) column. For example, if the report is an account statement grouped by a customer id column, the report must also contain a column that contains the email address for each customer. Note that the email column does not need to be visible in the report in order to use the bursting feature.

If the report meets the aforementioned requirements, you can enable report bursting by selecting the option *Email Using Report Column Values* at the top of the dialog. This will run a single query for the report, break the report up by groups, export each group, and deliver the exports to the email address listed for each group.

### 2.2.1.4. Schedule Summary

After you confirm the last step of the *Create Package Wizard*, the *Schedule summary* dialog pops up. It offers you a quick overview of the scheduled task you have just finished configuring. If you are satisfied with the task's settings, click *Confirm*. If not, click *Back* to go back to the *Create Package* wizard.

It is recommended to always check the *Next Export Time*. If you are not satisfied with the next export time, it indicates that the timing has been set wrong. You can click *Back* and revise the timing settings.

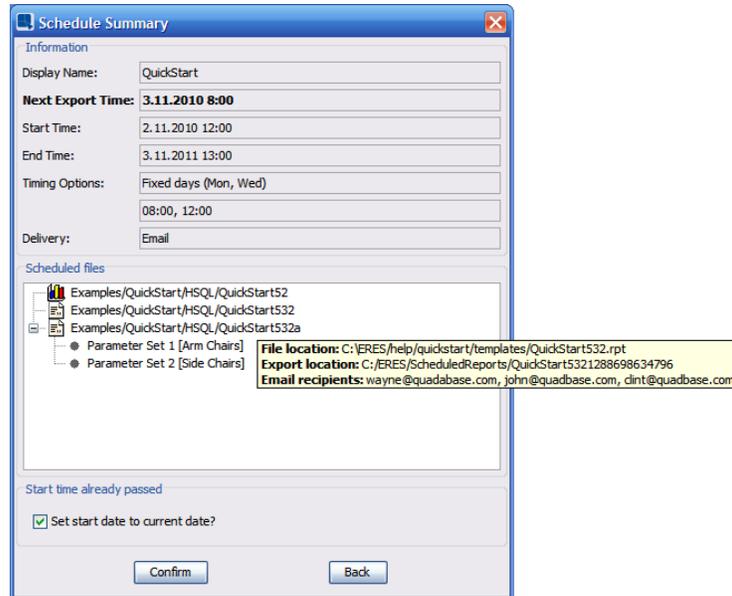


*Schedule Summary*

The *Schedule Summary* dialog has only one setting - *Set start date to current date*. This setting appears only when the start time has already passed. If you select this option, start time will be automatically set to current date/time.

Move the mouse over a scheduled object (report, chart, etc) in the *Scheduled files* list to display file and export locations.

If the *email delivery* is enabled, the tooltip also displays email recipients for the scheduled object. If the object has some parameters, there is a tooltip for each parameter set displaying email recipients.

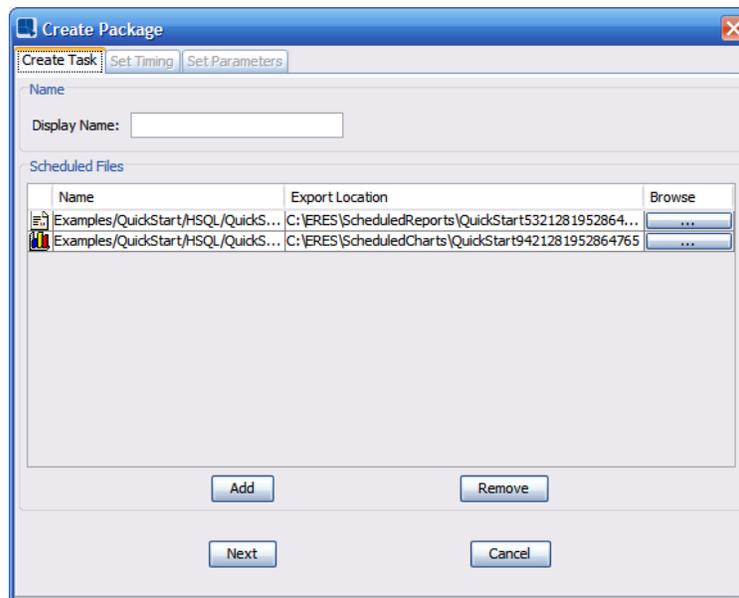


Schedule Summary

## 2.2.2. Setting an Archive

In addition to scheduling, which periodically exports the chart or report to a static file, ERES also supports archiving. An archive is similar to a schedule job except that instead of exporting a static copy of the report or chart, a version of the chart or report template is saved. The archived template contains all the data for that report or chart, thus saving a historical snapshot of the data that can be run/viewed at a later date.

To setup an archive, select the report or chart that you would like to use in the Organizer, and select *Set Archive* from the *Schedule/Archive* menu, or right click and select *Set Archive* from the pop-up menu. The following dialog will appear allowing you to set the name and location of the archive:



Set Archive Dialog

From this screen you are prompted to specify a *Display Name* for this archive. You can also select to specify the location and file name (without extensions) for the saved archive or allow the server to assign a location and file name. Note that each time the archive runs a new file will be created (the file name is appended with a timestamp)

creating a distinct file for each version. Archived files can also double as backup that can be used to restore old report/chart template version.

*Set archive* wizard is just a simpler version of the *Create scheduled task* wizard that was described in the previous chapters.

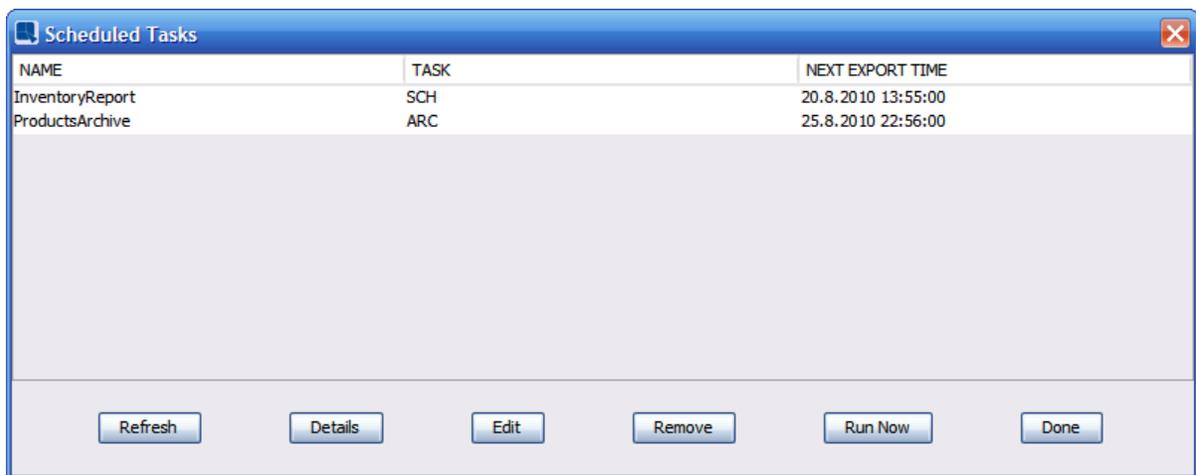
Once you have set the initial options for the archive, the next two tabs allow you to set the periodicity for the archive, and supply parameter sets (if any). These operations are exactly the same as for setting a schedule discussed in Section 2.2.1 - Setting a Schedule.

Archives cannot be delivered since they are in templates rather than exported formats. They can be accessed from the menu page, which will show a special icon  for files that are archived (as described in Section 7.1.2.1 - Viewing Reports and Charts).

## 2.2.3. View Tasks

You can see a list schedules or archives, by selecting *View schedule tasks* from the *Schedule/Archive* menu. A window will open with a list of all of the currently active schedule and archive jobs. The list displays the schedule name, task type (*SCH* for scheduled tasks, and *ARC* for archive tasks) and the next export time. This window also allows you to edit, remove, and immediately run the schedule or archive at the moment.

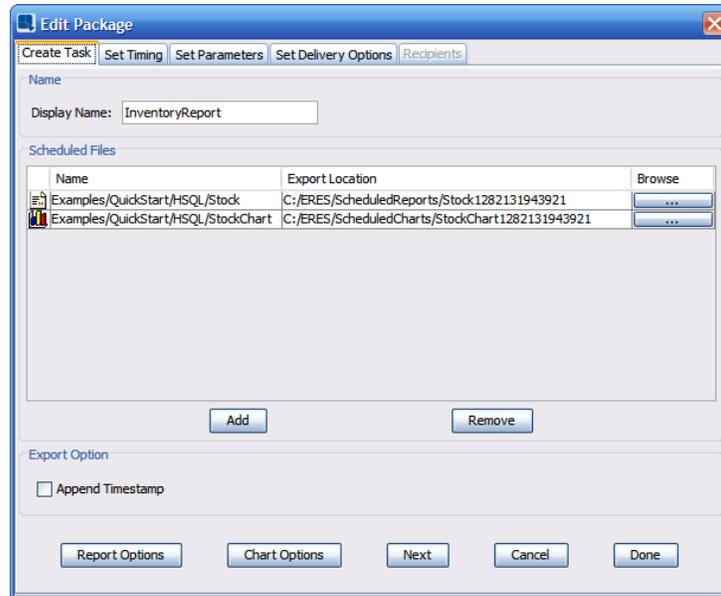
The *Schedule/Archive* menu also contains a very similar option called *View alert monitoring tasks*. This is useful when you are working with alerts monitoring. To learn more about alerts monitoring, please visit Section 11.4 - Monitoring.



*View Scheduled Tasks*

### 2.2.3.1. Editing Schedules and Archives

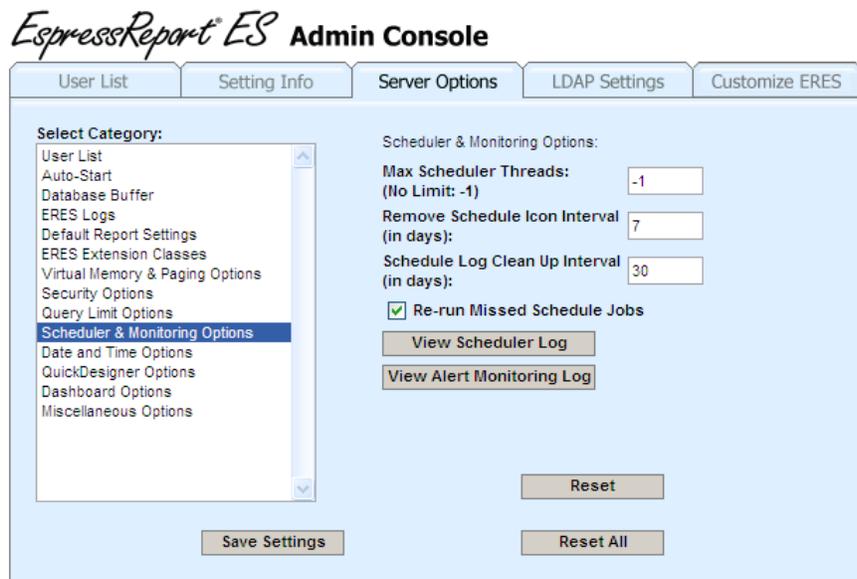
The *Edit* option in the *View Tasks* window allows you to modify the existing schedule or archive. The edit dialog looks very similar to the *Create scheduled task* dialog. However, rather than having to go through the steps in sequence, you can skip to any section by clicking on the tab names. There is also a *Done* button, which allows you to exit the dialog whenever you are satisfied with your changes.



*Edit Scheduled Task Dialog*

### 2.2.3.2. Schedule and Archive Logs

The admin also has the ability to view a log report containing detailed information about previously run schedules and archives. To view the detailed Scheduler Log, log in as the **admin**, and go into the administration console. In the *Server Options* tab, under the *Scheduler & Monitoring Options* section, click on the *View Scheduler Log* icon.



*Admin Console*

This will pop up a window containing the Scheduler Log. The root report displays information regarding the schedule list. This not only includes the currently active schedules and archives, it also includes any completed schedules and archives.

Name	Created by	Task type	Last export	Next export	Email delivery	FTP delivery	Printer delivery	Successful
ScheduledTask2	admin	Schedule	24.11.2010 13:24:00	Finished	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NO
		Name		Parameter set name		Parameter values		
Examples/QuickStart/HSQL/QuickStart53								
ScheduledTask1	admin	Schedule	24.11.2010 13:20:00	24.12.2010 13:20:00	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	OK
		Name		Parameter set name		Parameter values		
Examples/QuickStart/HSQL/QuickStart1042								
Examples/QuickStart/HSQL/QuickStart52								
Examples/QuickStart/HSQL/QuickStart52a								
		Parameter Set 1		Arm Chairs				
		Parameter Set 2		Side Chairs				
ArchiveTask	admin	Archive	24.11.2010 13:18:00	25.11.2010 13:18:00	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	OK
		Name		Parameter set name		Parameter values		
Examples/Dashboards/Alerts Components/Sales Overview Chart								
Examples/QuickStart/HSQL/QuickStart108								

Scheduler Log

**Note**

The list contains only tasks that have been run at least once in the past. If there are no such tasks, *Scheduler log* will not be visible. A “No tasks found!” warning message will be displayed instead.

All tasks in the Scheduler Log have *Task type* indicator which tells whether the task is a schedule or an archive.

If you do not want the completed schedules and archives to be on the list, you can filter them out. Click on the  *Filter* icon, set the *Show finished tasks?* parameter to *false* and click *Ok*.

Scheduler Log Filter

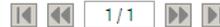
The *successful* field indicates whether the task was run successfully during the last export time. If one of scheduled templates could not be exported or one of delivery methods could not be performed, the *successful* field will look like this: **NO**. If that happens, you can get more detailed information in the task's drill-down. This will be discussed later.

Under each task's header, there is a list of its files. If there are several parameter sets added to a single template, the file will take several rows in the list, but its name will be displayed only once (as displayed in the following picture).

Name	Created by	Task type	Last export	Next export	Email delivery	FTP delivery	Printer delivery	Successful
ScheduledTask3	admin	Schedule	24.11.2010 13:38:00	Finished	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	OK
Name		Parameter set name			Parameter values			
Examples/QuickStart/HSQL/QuickStart532b		Parameter Set 1			Adad Chair			
		Parameter Set 2			Hapi Table			
		Parameter Set 3			Marduk Chair			

*Parameterized Report*

The list contains one drill-down. If you click on a scheduled task name, another report will open, providing detailed information about the particular task.



## ScheduledTask3 log

21.11.2010 14:26:26

Created by:	admin						
Next export time:	This task has finished						
Delivery settings:	<table border="1"> <tr> <td>Email delivery:</td> <td>Yes</td> </tr> <tr> <td>FTP delivery:</td> <td>No</td> </tr> <tr> <td>Printer delivery:</td> <td>No</td> </tr> </table>	Email delivery:	Yes	FTP delivery:	No	Printer delivery:	No
Email delivery:	Yes						
FTP delivery:	No						
Printer delivery:	No						

Export time: 21.11.2010 14:20:00			
Name:	Examples/QuickStart/HSQL/QuickStart532	Export success:	OK
Template name:	C:\ERES\help\quickstart\templates\QuickStart532.rpt	Email success:	<b>NO</b>
Export location:	C:\ERES\ScheduledReports\QuickStart5321290345447281		
Parameter set name:			
Parameter value:			

*Scheduler Log - Second Level*

All tables between two consecutive blue headers show details for files that were run at the same run time.

Each scheduled file and parameter set has its own table. In the left part of the table, you will find file details, namely: *name* (also doubles as organizer path), *template name* (also doubles as disk path), *export location*, and *parameter settings*.

In the right part of the table, there are export and delivery methods results. All files have the *Export success* indicator. It indicates whether the file was successfully exported. This is the most important indicator, because if the file was not exported successfully, there is no file to be delivered by the delivery methods. If the export fails, be sure to check whether the template can be accessed by the ERES server (it could have been removed or renamed), if the export location can be accessed (there is enough free disk space and the ERES server has write permission to the directory) and if the file's data source can be accessed.

Each delivery method has its own indicator. If some delivery method was not enabled for the scheduled task, its indicator will not be visible in this table. If a delivery indicator is visible, you can click on it to drill-down.

Export time: 21.11.2010 14:20:00			
Name:	Examples/QuickStart/HSQL/QuickStart532	Export success:	OK
Template name:	C:\ERES\help\quickstart\templates\QuickStart532.rpt	Email success:	<b>NO</b>
Export location:	C:\ERES\ScheduledReports\QuickStart5321290345447281	FTP success:	OK
Parameter set name:		Print success:	OK
Parameter value:			

*Export and Delivery Success*

This log level can be also filtered. The log can either contain all exports or just failed exports. To enable the filter,

click on the  *Filter* icon and set the *Failed exports only* parameter to *true*.

To return back to the root level, click on the  *Back* icon.

## Email Delivery Log

If you click on an *Email delivery* indicator in the second Scheduler Log level, email delivery log will open. It contains details for one file and one run time only. Each email recipient has one table which shows whether the email was sent successfully to the recipient's email address. If the delivery fails, an error message will be displayed.

For example:

## Email delivery log

Task name:	ScheduledTask3
Run time:	21.11.2010 14:20:00
From:	clint@quadbase.com
Subject:	Monthly report

To: george2@quadbase.com			
Name	Parameter name	Parameter value	Success
Examples/QuickStart/HSQL/QuickStart532			NO
<b>Error message</b>			
<a href="#">Sending failed:</a> <a href="#">nested exception is:</a> <a href="#">class javax.mail.MessagingException: Could not connect to SMTP host: localhost, port: 25;</a> <a href="#">nested exception is:</a> <a href="#">java.net.ConnectException: Connection refused: connect</a>			

### *Failed Email Delivery Details*

In the previous picture, there is a failed email delivery. The log shows that an email from `clint@quadbase.com` to `george2@quadbase.com` (not actual email addresses) couldn't be sent because the ERES server wasn't able to connect to SMTP host at `localhost:25`. It could be caused by a network problem or by incorrect SMTP settings (as described in Section 1.4.1.2 - Setting Info). This is probably the most common problem with email delivery.

## FTP delivery

FTP delivery log is much simpler – there is only one “recipient” so there is only one table. The table contains information about the scheduled file, ftp settings, run time, and *success* indicator and in case the delivery has failed – an error message. You can see such situation in following picture. The problem is that the ERES server was not able to connect to `gamma.quadbase.com` FTP host (not an actual FTP host).

## FTP Delivery log

Name:	Examples/QuickStart/HSQL/QuickStart532a
Run time:	21.11.2010 13:11:00
FTP Host:	gamma.quadbase.com
FTP Path:	/uploads
FTP User name:	jamie
Parameter name:	Parameter Set 2
Parameter values:	Single Dressers
Success:	<b>NO</b>
Error message	
gamma.quadbase.com	

*Failed FTP Delivery*

### Printer Delivery Log

Printer delivery log is very similar to the FTP delivery log. It shows file and printer details and an error message (only in case the delivery has failed).

## Printer delivery log

Task name:	ScheduledTask2
Name:	Examples/QuickStart/HSQL/QuickStart532
Run time:	21.11.2010 14:14:00
Printer:	LaserPrinter1
Print from page:	1
Print to page:	All pages
Parameter name:	
Parameter value:	
Success:	<b>OK</b>

*Successful Printer Delivery*

To go back to the second log level, click on the  *Back* button.

## 2.3. Security and Security Administration

This chapter covers security concepts in ERES and the administration of user privileges. Other security features for data registries and deployment (menu page, URLs, and the API) are discussed in later sections.

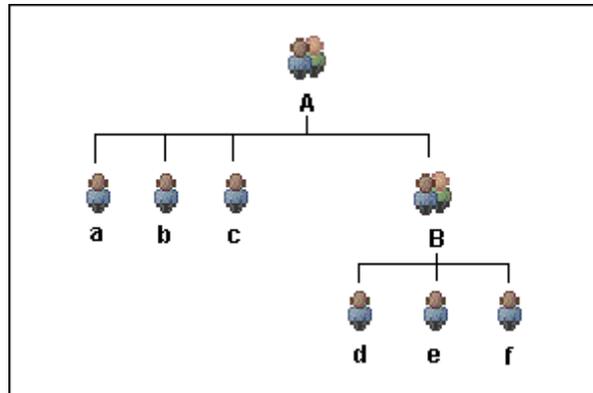
There are two security concepts in ERES:

- User privileges** This is the default security concept. User privileges are derived from relations between users and groups (will be described later). Relations between users/groups can be modified.
- File permissions** You can disable the “User Privileges” mode for certain files/folders and set custom permissions for those files/folders.

## 2.3.1. Security Concepts

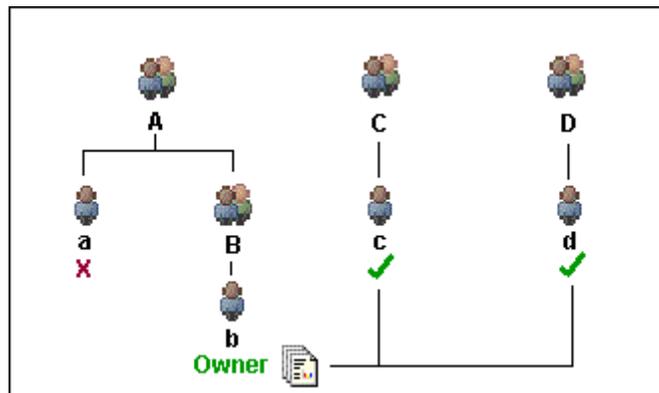
In EspressReport ES, privileges are based on the concept of ownership by default. This means that the privileges or access to a particular folder, report or chart, stem from who is the owner or creator of that folder, report or chart. Each user (or group) has a set of privileges that apply to other users or groups when accessing reports or charts created by that user.

In this ownership approach, users belonging to the same group or subgroups of the same group have full access privilege to items (folders, reports, charts) created by each other. For example, group A (group structure A) has users a,b,c and sub-group B. Group B (as a sub-group of A) has users d,e ,and f. The default behavior is such that users a,b,c,d,e,f will have full access privilege to items created by each other. At the same time, users outside of the group structure A have no access to the items created by users in this group structure.



*Users from group A and B will have access to each other's files.*

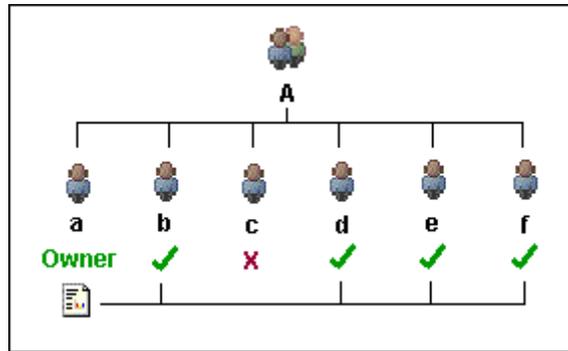
With the “user privileges” setting functionality, you (as the administrator) can modify the default privilege granting scheme. For example, you can grant users from group C and D read or read/write access to group B users' items whereas group A's users have no permission to access group B users' items. Meanwhile, group B users still have same access right to other group A users' items if group A is still using default privileges.



*After setting group permission, groups C and D have access to group B's files, but group A does not.*

In addition to the “user privileges” ownership model, the privileges for individual items can be modified to further enhance the security settings. This function is called the “File permissions”. In this scenario, individual items can be made accessible to groups and users independent of the ownership assigned scheme.

For example, an item created by user a is automatically fully accessible to users b,c,d,e, and f. But you can specifically modify the permission setting such that user c has no access right to that particular item while the rest of the users in the group structure maintain their access rights.

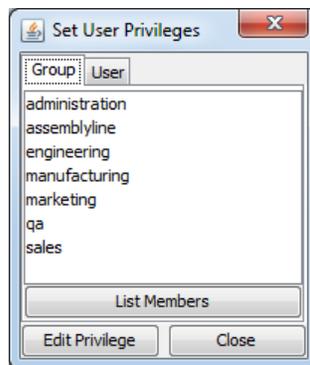


After setting file permission, user c does not have access to this particular report.

## 2.3.2. Setting User Privileges

Privileges are set by the administrator in the Organizer interface (with the exception of the basic designer/viewer role for users set in the Admin Console). To set user privileges in Organizer, select Edit → Set Privileges or click

the  *Set Privileges* button on the toolbar. This will bring up a dialog, allowing you to set owner permissions for users and groups.



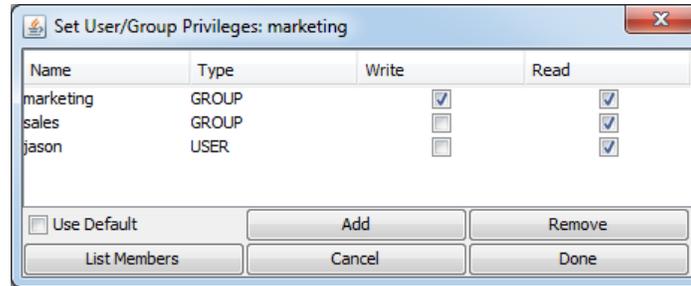
*Permission Setting Dialog*

The dialog contains two tabs one listing all of the groups currently defined for ERES and the other listing all the users. From either of these tabs you can select a user or group for which you would like to set owner permissions by selecting it and clicking the *Edit Privilege* button.



### Note

When you select a user or group, you are setting privileges for folders, reports, and charts that this user, or users in this group create (or become owners of). You are not setting access levels for that particular user or group, rather you are setting privileges for other groups or users who access reports created by this user or group.



*Group/User Permissions*

Every group or user will automatically have read and write permissions for their own report and charts. You can give other groups and users access, by clicking the *Add* button. This will bring up a list of users and groups allowing you to select which users and/or groups you would like to add. For each additional user or group, you can set their access to read-only, or read-write.

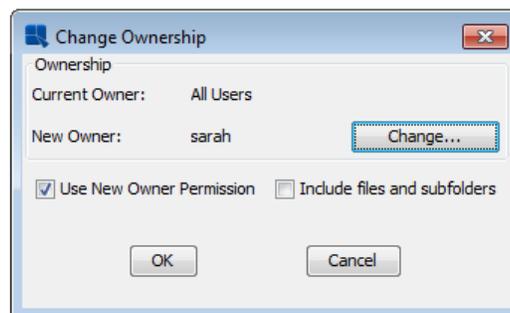
Read-only access allows the user or group to view the file using various ERES tools. For example, when the user opens the file from the ERES Organizer, the program will launch the corresponding viewer (Chart Viewer for charts, Page Viewer for reports, Dashboard Viewer for dashboards) rather than opening the Designer. The user can also create schedules and archives for this file and access the file through the ERES menu page. With read-only permission, the user will not be able to edit the file using the Designers, nor will they be able to generate URL for the file. In addition, the user cannot see the file and URL path in the Organizer.

Read-write access grants full privilege to the user. They can edit the file using the corresponding designer as well as generate URL and see the file and URL path in the organizer.

If the *Use Default* checkbox is checked then the user or group will inherit permissions from the group to which he/it belongs. If the user or group does not belong to any other groups then they will have read and write permissions for their reports but no other users will have access. This option is used by default in EspressoReport ES. There users will automatically inherit group permissions unless settings for that user are specifically modified.

### 2.3.2.1. Changing Ownership

Generally, a user is the owner of any folder, chart, or report he/she creates. However, the administrator can change the owner of any folder or file in the Organizer. To change the owner of a folder or file, right click on it, and select *Change Ownership* from the pop-up menu. This will bring up a dialog showing the current owner of the file or folder.



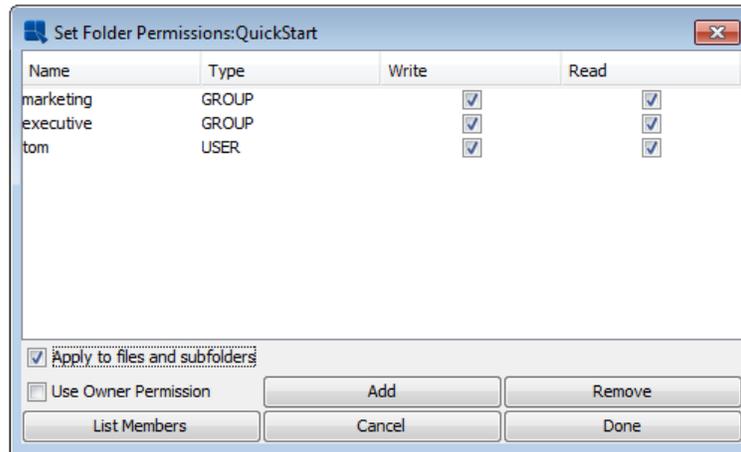
*Change Ownership Dialog*

To change ownership, click the *Change* button. This will bring up a list of all the defined users. Select the user to which you would like to assign ownership and click *Ok*. The new owner will then be reflected in the dialog. Click *Ok* again to apply the changes. Note that changing the owner of the file will generally effect which users can access and edit the file.

### 2.3.2.2. Folder/File Permissions

By default, user privileges for files and folders in the Organizer are derived from the owner of that file or folder. However, in certain circumstances you may want to set privileges for specific files or folders to be different from

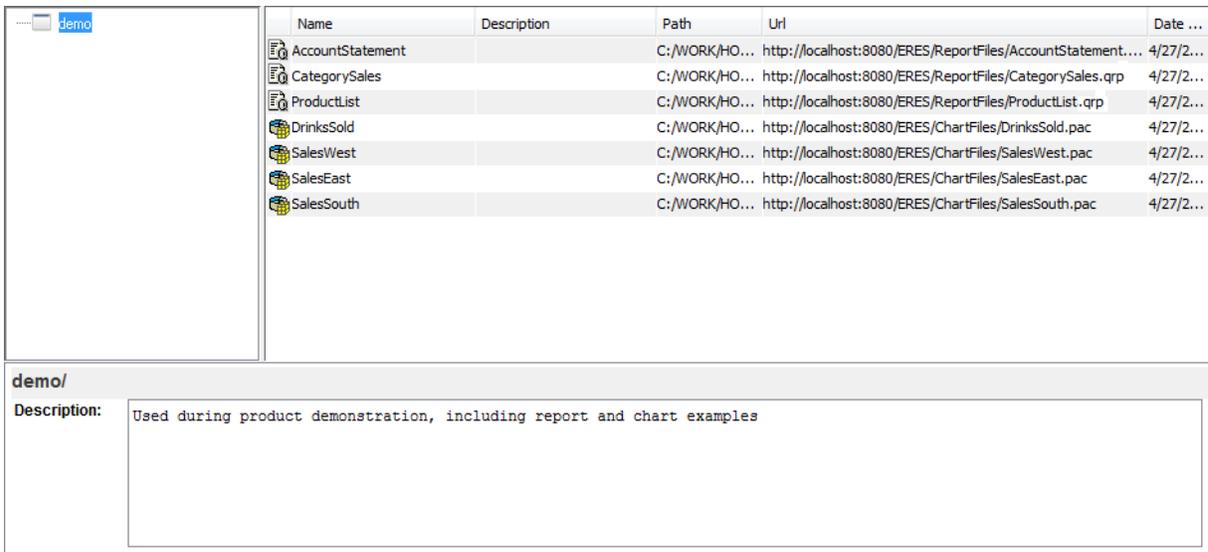
those associated with the owner. To apply a different set of permissions to a file or folder, right click on it and select *Change Folder/File Permissions* from the pop-up menu.



*File/Folder Permissions Dialog*

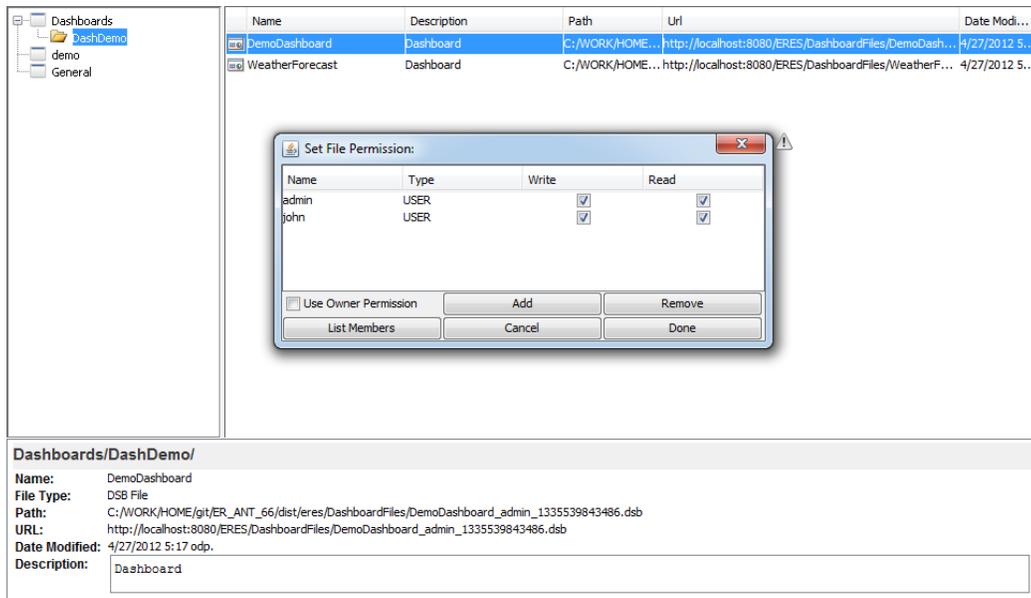
The *Use Owner Permission* checkbox indicates, if checked, that the folder or file is currently using owner permissions. To add different levels of access for that particular object, first un-check this option, and click the *Add* button. This will bring up a list of users and groups, allowing you to select which users and/or groups you would like to add. For each additional user or group, you can set their access to read-only, or read-write. Once you have finished adjusting access for the object, click *Ok* and the changes will be applied.

Here is a practical example that illustrates how the file/folder permissions function. Suppose John, a member of the marketing group, can only see the demo project, which is used during project demonstrations. All other existing projects that John does not have access to are automatically hidden from view.



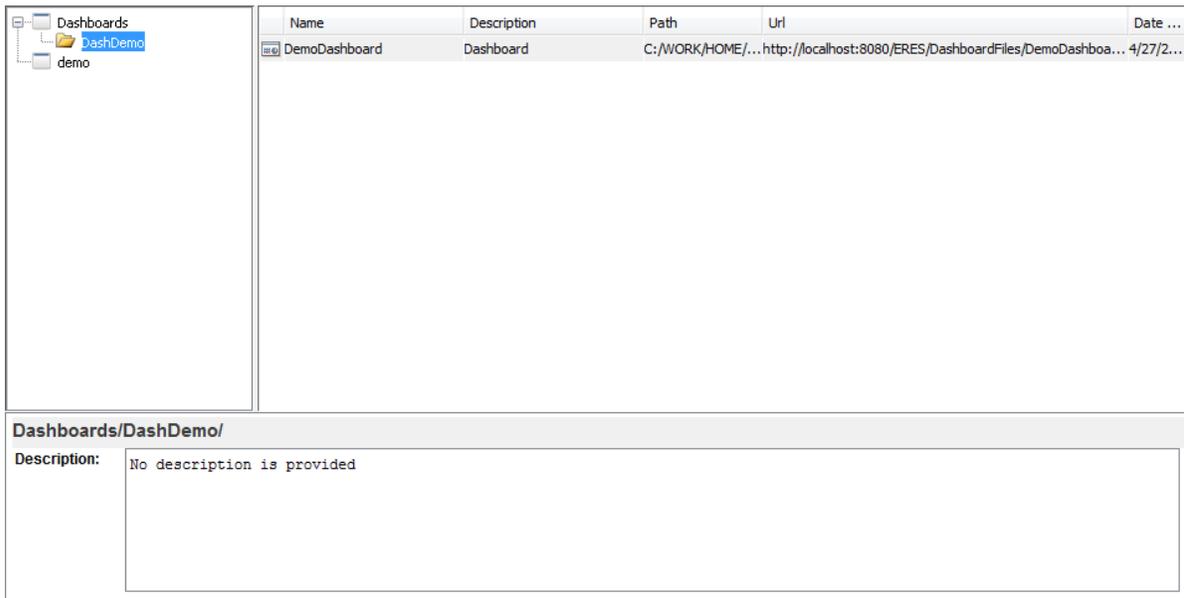
*John's Organizer*

Suppose the administrator would like for John to edit and incorporate a dashboard into his presentations. To do so, the administrator gives John read and write permissions for one dashboard in the *Dashboard/DashDemo/* project. It is not necessary for administrator to set folder privileges since the change will be propagated through the parent folders.



*Setting File Permission*

The next time John launches the Organizer, he will see the dashboard file that he has been given permission to view and edit. However, John will not be able to see any other files in this folder nor any files in the Dashboard folder (the parent folder).



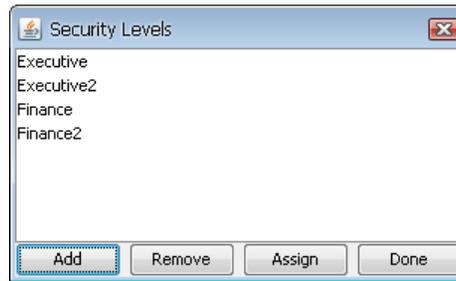
*Dashboard File is Accessible*

### 2.3.3. Security Levels

In addition to setting permissions and access for different reports, the administrator can also apply security settings within reports, such that users and groups can get different views of the same report when accessing it. This security feature is administered through security levels.

Security levels are settings which can be applied to reports. For different security levels, elements (cells, and columns) in a report can be modified, and filters can be automatically applied to the report. For more information about how to apply security settings for reports in the Report Designer, please see Section 4.1.10 - Template Security.

Security levels are created by the administrator in Organizer and applied to reports in the Report Designer. To create or modify security levels, select *Set Security Levels* from the *Edit* menu. This will bring up a dialog listing all the security levels that have been defined.

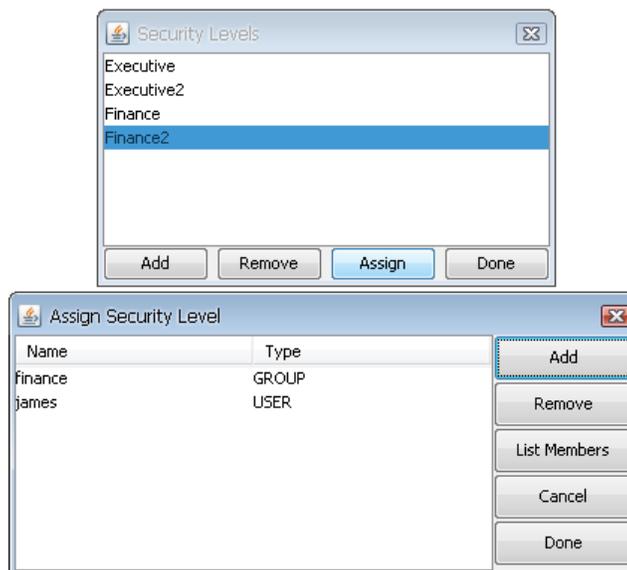


*Security Levels Dialog*

To add a new security level, click the *Add* button. This will bring up a dialog prompting you to specify a name for the new security level. There is no limit to the number of different security levels that can be defined. Defined security levels are then imported into the Report Designer allowing you to set/modify report behavior for different levels.

### 2.3.3.1. Associating Security Levels with Users and Groups

In ERES, users and groups can be associated with defined security levels to provide end-to-end security. When a user accesses a report, the security level to which they are associated will be applied, and the view defined by that security level will be presented to the user. To associate users and groups with security levels, select a security level in the list and click the *Assign* Button in the Security Level Dialog.



*Assign Security Level Dialog*

In the dialog that opens, click the *Add* button to associate a user or group with the selected security level. This will bring up a dialog that contains a list of all the defined users and groups. From this dialog you can select groups and users to associate with the security level. When you have finished, click *Done* to return to the security levels dialog.

You can assign more than one user or group to a security level, but users cannot be assigned to more than one security level at a time. Security level associates, like permissions are applied at the lowest level. This means that if an individual user is associated with a security level, it will take precedence over settings for their group.

# Chapter 3. Data Sources

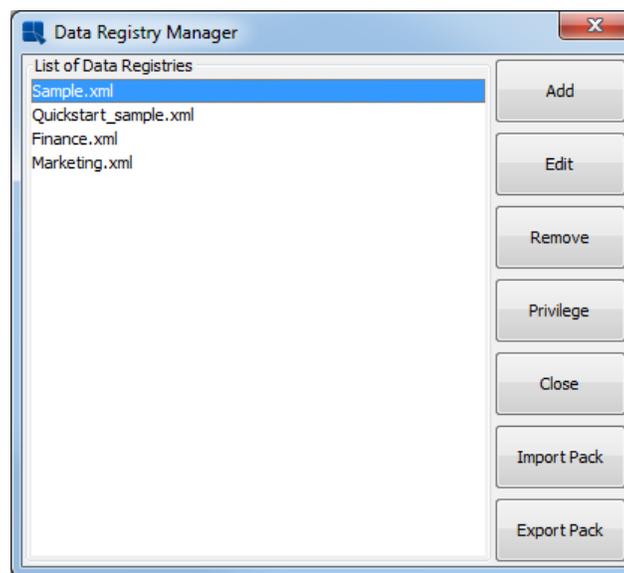
## 3.1. Data in Organizer

EspressReport ES can draw data from JDBC compliant databases, text files, XML files, EJBs, and even bring in object/array data through class files. Prior to creating reports or charts, users must first set up data sources they want to use. Data source information, including database connection information, database queries, text file location, XML file/DTD/Schema location, Java class location, and EJB connection information is stored in XML registry files that are set up and defined within the Organizer interface.

### 3.1.1. Managing Data Registries

To edit or create a data registry, you can either select the *Manage Data Sources* option from *File* menu, or click on

the  *Manage Data Sources* icon on the toolbar. This will launch the Data Registry Manager dialog.



*Data Registry Manager Dialog*

From this dialog, you can add, remove, and edit registries, as well as assign user privileges for data registries.

To add a registry, click the *Add* button and a dialog will appear prompting you to specify a filename for the new registry. Click the *OK* button and a new Data Source Manager window will open, allowing you to set up data sources within the registry.

To edit a registry, first select it from the list on the left side and click the *Edit* button. This will bring up the Data Source Manager window for the selected registry file.

#### 3.1.1.1. Exporting/Importing Data Registries

A data registry consists of several files such as database queries, text files, xml files, etc. If you export a data registry, all files will be packed in a single *rpack* file that can be archived or imported to another ERES installation/server.

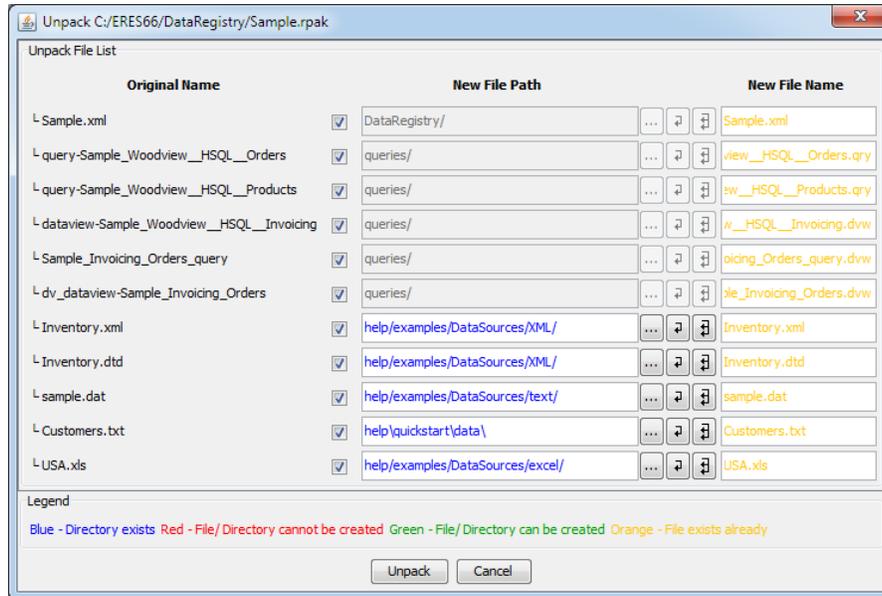
To export a data registry, select it and then click the *Export Pack* button. Enter a file name and click the *OK* button.



#### **Note**

Only administrators can export or import data registries.

To import a data registry, click the *Import Pack* button and then select a *rpack* file. You can insert the data registry using its original name or you can change it in the *New registry name* field. If you want to choose which files should be unpacked or change their path or file name, click the *Details* button.



Unpack dialog

In this dialog, you can choose which files you want to unpack and you can also change their names and paths.

If you don't want to unpack any file, uncheck 'unpack file to disk' checkbox.

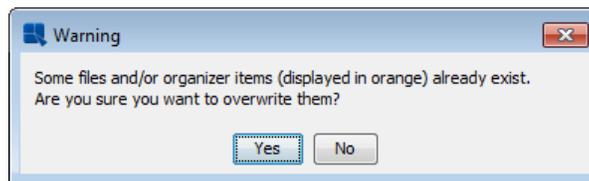


### Note

File paths of some type of files (for example: sql queries) can't be changed. In such cases, the *New file path* field and buttons will be disabled.

Click the *Cancel* button to get back to the *Import Data Registry* dialog.

Click the *Unpack* button. The data registry will now unpack all contained files. It is possible that some files from the *rpak* file might have the same file names and paths as other existing files on your hard drive. If there are any filename conflicts, the following dialog will pop up.



Data Registry Manager Dialog

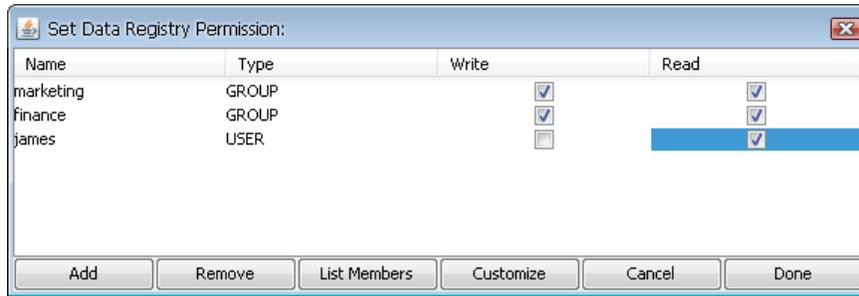
If you click *Yes*, the conflicting files on your hard drive will be overwritten with the files from the *rpak* file.

If you click *No*, the *Unpack dialog* will open again allowing you to solve the filename conflict.

### 3.1.1.2. Data Registry Privileges

As with other objects in EspressoReport ES, the administrator can also assign privileges to Data Registries. These include access permissions (read/write) as well as content filtering (restricting which sources in the registry users or groups can see).

To set privileges for a registry, select it in the Data Registry Manager and click the *Privilege* button. This will bring up a dialog listing permissions for the file.



*Registry Privileges Dialog*

To give users or groups access to the registry, click the *Add* button. This will bring up a list of all the defined users and groups. From this dialog, select the groups and users you want to add. You can set access rights for each user or group.

For each user or group that has access to the data registry, you can also customize their view of the registry. This allows you to restrict access of a group of users to specific data sources or types of data sources. To customize the registry display, select one of the users or groups in the registry privileges dialog and click the *Customize* button. This will bring up a dialog showing all the data sources that have been defined in the registry.



*Customize Registry Dialog*

Each node (data source) in the registry can be set to visible or invisible by selecting it and checking the option at the bottom of the dialog. Invisible nodes will be grayed out. Note that if you render a parent node invisible, all the sub-nodes will also be invisible, so you can't make Queries node invisible and individual queries visible at the same time.

Once you finish, click the *Ok* button and the settings will be associated with the user or group. When they access the registry, only the visible nodes will appear.

### 3.1.2. The Data Source Manager

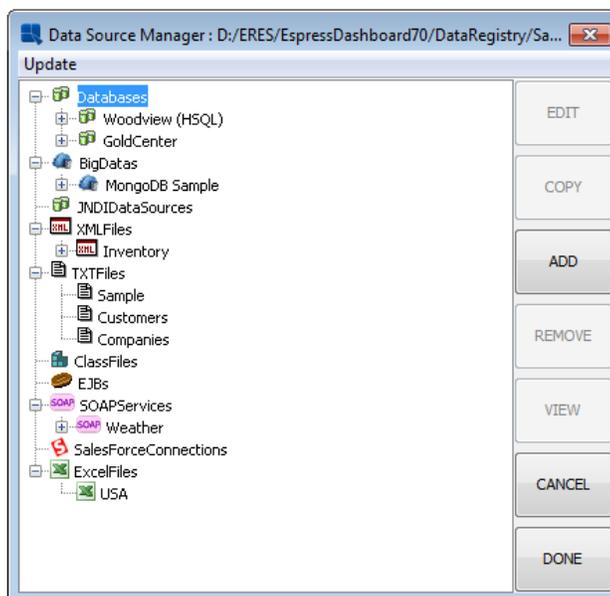
When you edit or create a new data registry, the Data Source Manager window will open allowing you to add and edit data sources. The Data Source Manager is a visual representation of the registry XML file.



#### Note

The XML data source repository only stores location and connection information, not the actual data. It is not an XML file that contains data to be used in a report or chart.

### 3.1.2.1. Using Data Source Manager



*Data Source Manager*

The left side of the window contains a tree listing all of the data sources in the registry file. Grouped under *Databases* are individual databases and their associated queries and data views. Grouped under *JNDIDataSources* are database sources that use JNDI (Java Naming and Directory Interface) name to connect instead of JDBC. Grouped under *XMLFiles* are all XML files and their associated queries. Grouped under *TXTFiles* are all specified Text files. Grouped under *ClassFiles* are all specified class files, and grouped under *EJBs* are all specified EJB connections.

Right side of the window contains a series of buttons controlling all of the functions of the Data Source manager. Each button performs the following function:

- Edit:** This option allows you to modify attributes of a data source. For a database, it allows you to change the connection information and modify queries/data views. For XML files, it allows you to change the file and its location, and modify XML queries. For text files, it allows you to change the display name and file location. For class files, it allows you to change the display name and modify the location. And for EJBs, it allows you to change the display name as well as parameter values.
- Copy:** This option is not only available for database nodes and bigdata nodes but also for the specified query or data view.
- Add:** This option allows you to add a data source. It will create a new source depending on which node is selected on the left side of the window. Hence, if you select *TXTFiles* and click the *Add* button, you will be prompted to add a new text file data source.
- Remove:** This option will remove the selected data source.
- View:** This will open a new window showing the tabular data from the selected data source, either query, XML file, text file, class file or EJB. For data views, the data view query interface will start. From this window, you can select to create a new report or a new chart by selecting the appropriate button.

INDEX	City	State	Longitude	Latitude
1	Littletown	NY	-75.4321289...	42.2773087...
2	Pitterson	NJ	-74.200498	39.76472
3	Aachen	TX	-100.327148...	32.4726950...
4	Akergen	PA	-79.7717285...	40.1620833...
5	Sevilla	AK	-162.905273...	61.9389504...
6	Chicago	IL	-87.624333	41.879535
7	Piscataway	NJ	-74.398358	40.500486
8	San Francisco	CA	-122.419204	37.775196
9	Los Angles	CA	-118.081054...	34.0162418...
10	Cleveland	OH	-81.693716	41.499713
11	Little Rock	AK	-156.5376	58.659738
12	New York	NY	-73.986941	40.75604
13	San Jose	CA	-121.873881	37.316466
14	Long Island	NY	-73.986941	40.75604
15	Orlando	FL	-81.364438	28.553154
16	Austin	TX	-97.745209	30.268735
17	Ankertown	FL	-82.732626	29.037293
18	George Park	PA	-76.320675	40.033748
19	Boomtown	NY	-77.5634765...	42.5045028...

View Data Source Dialog

**Cancel:** This will cancel the wizard process.

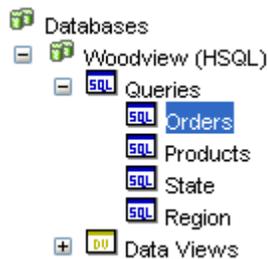
**Done:** This will save any changes you've made in the registry and return you to the registry list.

### 3.1.2.2. Data Source Node Locking

The data registry has an automatic node locking system designed to block multiple users from editing related nodes at the same time. The system utilizes your IP address, username, and a special security token to determine whether you are allowed to add, remove, or edit a specific node. If you are currently editing a node, other users will not be able to edit it. However, if you disconnect from the server or move to another computer and try to open the same node from the same username, an option box will appear allowing you to override your previous lock.

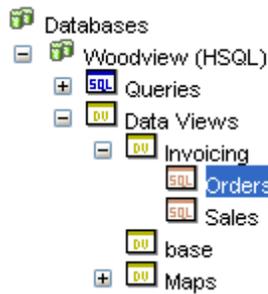
Please note that administrator is capable of unlocking any node, even if they did not open it in the first place.

The data registry is a tree where data type is the root and queries and files are the leaves. Changing any node within a particular branch will lock all parent nodes and all child nodes in that branch. Here are some examples of how the node locking system works:



Locking Query

In the above image, you can see that the database *Woodview (HSQL)* and query *Orders* are part of the same branch. So if you edit *Orders* query, it will lock both *Orders* and database *Woodview (HSQL)*. If you edit *Woodview (HSQL)*, the database will be locked and all child nodes connected to this database will be locked as well, which includes *Orders*, *Products*, *State*, and *Region*.



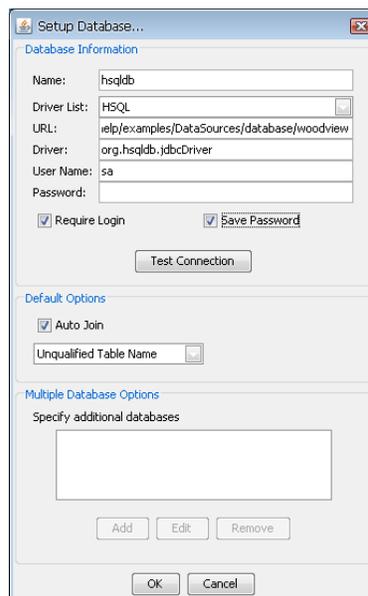
*Locking Data Views*

In the case of data views, there are three levels in the tree. In the above image, editing *Orders* data view query will lock the query, data view *Invoicing*, and database *Woodview (HSQL)*. If the data view *Invoicing* is being edited, both queries *Orders* and *Sales* will be locked, as well as database *Woodview (HSQL)*. However, in both of these scenarios, other data views and queries can still be edited. If the database *Woodview (HSQL)* is being edited, all data views as well as their queries will be locked.

### 3.1.3. Data from a Database

ERES can draw data from any JDBC compliant database. In order to connect to a database via a 3rd party driver (other than the JDBC bridge), you will need to add the classes for that driver to the classpath of the application server/servlet runner where you have deployed the ERES server. If you installed ERES with Tomcat, then you need to modify `setclasspath.bat / .sh` file in `/bin/` directory of your Tomcat installation to add the driver classes, or copy them to `<ERESInstallDir>/WEB-INF/lib` directory. Note that JDBC drivers for MS SQL Server, MySQL, Oracle, Informix and PostgreSQL databases are included as a convenience. Other database JDBC jar files are not included due to licensing, multiple drivers, and/or other concerns, although support for these databases exist and the jar files can be explicitly added.

The first step in using a database as the data source is to set up a database in the registry and specify the connection information. To add a database, click on the *Databases* node and click the *Add* button. This will bring up a window prompting you to specify the connection information for that database. You can choose a database to connect to from the Driver List or specify the information manually. Fields to enter are database name, URL, and driver. You can also select whether the database requires a login and whether you want to save username and password information. If you select to save login and password information, you can then enter these informations in *User Name* and *Password* textboxes. Then click the *Ok* button and the new database will be added to the Data Source Manager window.



*Add Database Dialog*

In order for ERES to create a connection to the database, the following information must be provided:

**URL:** This JDBC URL specifies the location of the database to be used. A standard JDBC URL has three parts, which are separated by colons:

**`jdbc:<subprotocol>:<subname>`**

The three parts of a JDBC URL are broken down as follows:

1. `jdbc` - the protocol. The protocol in a JDBC URL is always `jdbc`.
2. `<subprotocol>` - the name of the driver or the name of a database connectivity mechanism, which may be supported by one or more drivers. A prominent example of a subprotocol name is `odbc`, which has been reserved for URLs that specify ODBC data source names. For example, to access a database through a JDBC-ODBC bridge, you have to use a following URL:

**`jdbc:odbc:Northwind`**

In this example, the subprotocol is `odbc` and the subname `Northwind` is a local ODBC data source, i.e. `Northwind` is specified as a system DSN under ODBC.

3. `<subname>` - a way to identify the database. The subname can vary, depending on the subprotocol, and it can have a subsubname with any internal syntax the driver writer chooses. The function of a subname is to give enough information to locate the database. In the previous example, `Northwind` is enough because ODBC provides the remainder of the information.

Databases on a remote machine require additional information to be connected to. For example, if a database is to be accessed over your company Intranet, the network address should be included in the JDBC URL as part of the subname and should follow the standard URL naming convention of

**`//hostname:port/subsubname`**

Assuming you use a protocol called `VPN` for connecting to a machine on your company Intranet, the JDBC URL you need to use can look like this:

**`jdbc:vpn://dbserver:791/sales`** (similar to `jdbc:dbvendorname://machine-Name/SchemaName`)

It is important to remember that JDBC connects to a database driver, not the database itself. The JDBC URL that identifies the particular driver is determined by the database driver vendor. Usually, your database vendor also provides you with the appropriate drivers. It is highly recommended that users contact their database driver vendor for the correct JDBC URL that is needed to connect to the database driver.

**Driver:** This is the appropriate JDBC driver that is required to connect to the database. If you are using the JVM included within the installation (or Oracle's J2SE), use the following driver specification to connect to an ODBC data source:

**`sun.jdbc.odbc.JdbcOdbcDriver`**

You can also specify a JDBC driver name specific to your database if you are NOT using the JDBC-ODBC bridge. For example, the Oracle database engine will require `oracle.jdbc.OracleDriver`.

**User Name:** This is the login used for the database.

**Password:** Password for the above user.

Once you specify the connection information, you can test the database connection by clicking the *Test Connection* button. This will test the connection using the information you've provided and report any problems.

The *Default Options* portion of the dialog allows you to specify some properties for queries generated through the Query Builder interface or data views. You can specify whether to auto-join selected tables. Auto-join will attempt to join primary and foreign keys defined in the database. You can specify the table name format that should be used for queries either unqualified (only table name), or 2-part or 3-part qualified. Properties specified here will become the default setting for new queries and data views. They can also be modified for individual queries.

The *Multiple Database Options* portion of the dialog allows you to specify additional databases (i.e. additional database URLs) to obtain data from within the query. This option is only available when the database (original and

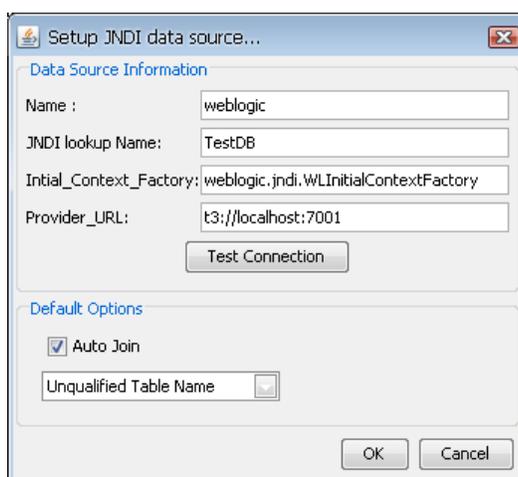
any additional database) is MS SQL Server and 3-Part Qualified Table Name option is chosen. Note that the same login details as well as the same driver (as defined in the original connection) are used to connect to the specified additional databases as well. The query can obtain data by referencing a column in the additional database using a 3-Part table nomenclature.

There are two sample databases included within the ERES installation. One is a HSQL (a pure Java application database) database and the other one is a MS Access database. Both contain the same data and are located in `help/examples/DataSources/database` directory. For details about how to set up connections to these sample databases, please see Section Q.3.1.1 - Setup Database Connections of the Quick Start.

### 3.1.3.1. JNDI Data Sources

In addition to connecting to databases via JDBC, ERES lets you use the JNDI (Java Naming and Directory Interface) to connect to data sources. In ERES, JNDI data sources are treated just like database data sources and support the same functionality (queries, parameters, data views, etc.). The advantage of using a JNDI data source is that it potentially makes it easier to migrate reports between environments. If data sources in both environments are set up with the same lookup name, reports can be migrated without any changes.

To connect to a JNDI data source in ERES, you must have a data source deployed in the same Web application environment as you are running the ERES server. To set up a JNDI data source, select the *JNDIDataSources* node in the Data Source Manager and click the *Add* button. This will bring up a dialog allowing you to specify the connection information.



*JNDI Setup Dialog*

The first option allows you to specify a display name for the data source. The second option allows you to specify the JNDI lookup name for the data source. The third option allows you to specify the initial context factory for the data source, and the last option allows you to specify the provider URL. This information will vary depending on the application server you're using as different vendors implement JNDI data sources differently. You can test the connection by clicking the *Test Connection* button.

### 3.1.3.2. Queries

Once you add a database, a new node for your database will appear in the Data Source Manager window. When you expand the node, you will see two more nodes, one called *Queries* and one called *Data Views*. These are the two ways to retrieve data from your database. To create a new query, select the *Queries* node and click the *Add* button. A dialog will come up prompting you to specify a query name and select whether you would like to enter SQL statement or launch the Query Builder.

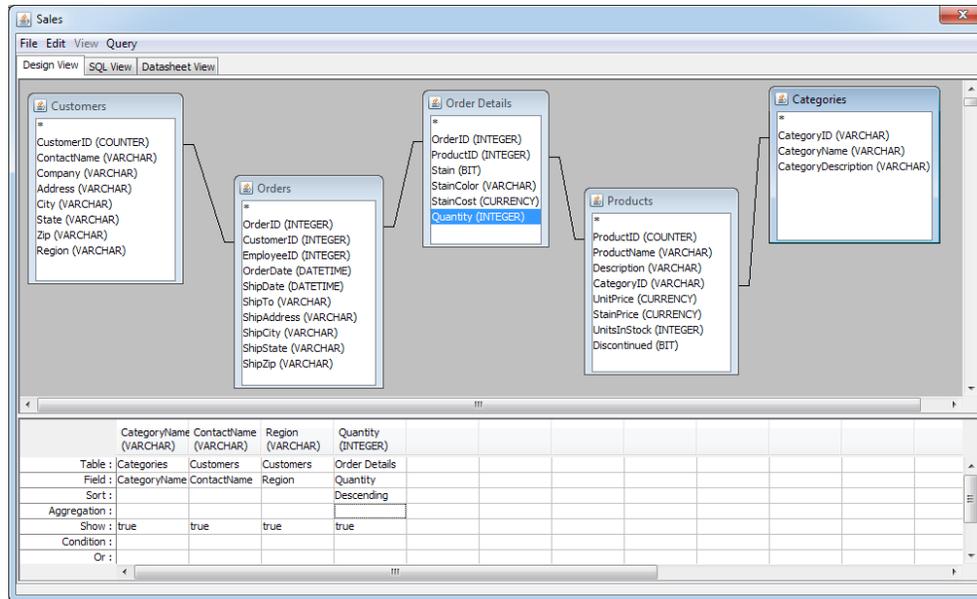
If you select to enter an SQL statement, a dialog box will come up allowing you to type in your SQL statement. From this dialog, you can also load a QRY or text file containing SQL text, or execute a stored procedure. If you select to launch the Query Builder, the Query Builder will open in a new window, allowing you to construct the query visually. After you finish building or entering the query, you will return to the Data Source Manager window and the query will appear as a new entry under the *Queries* node for your database.

#### 3.1.3.2.1. Using Query Builder

The Query Builder is an integrated utility that allows you to construct queries against relational databases in a visual environment. To launch the query builder, add a new query within the Data Source Manager and select the *Open*

*Query Builder* option. The Query Builder will then open in a new window. You can also launch the query builder to modify an existing query by double clicking the query name in the Data Source Manager.

The main Query Builder window consists of two parts. The top half of the window contains all the database tables selected for the queries and their associated columns. The top window also shows what joins have been set up between column fields. The lower half of the main window or QBE (query by example) window contains columns that have been selected or built for the query and their associated conditions.



*Query Builder Window*

There are three tabs at the top of the Query Builder window. These allow you to toggle between different views. The *Design View* tab is the main designer window described above, the *SQL View* tab shows the SQL statement that is generated by the current query and the *Datasheet View* tab shows the query result.

When you finish constructing the query, select *Done* from the File menu to return to the Data Source Manager.

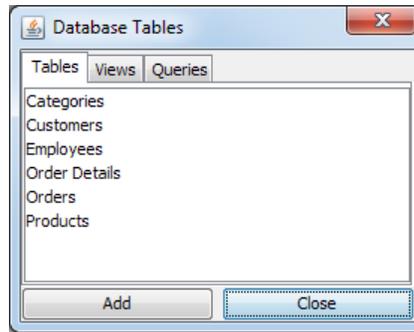
### 3.1.3.2.1.1. Tables

When the Query Builder launches for the first time, a tabbed window will appear containing a list of all the tables within the database. A second tab contains a list of all the views in the database, and the third tab contains a list of other queries you have designed for the database under a heading called *Queries*. From this window, you can select the tables/views/queries from which you would like to build the query. You can also load a previously designed query as a table. To add a table, select it and click the *Add* button or double click on the table name. When a table is added, it will appear in the main Query Builder window and will display all columns within that table. To remove a table, right click within the table and select *Delete* from the pop-up menu. You can also specify a table alias and sort the fields alphabetically from this menu. You can close the tables window by clicking the *Close* button. To reopen it, select *Show Tables* from the *Query* menu.



#### Note

By default, the tables will appear using the name format you specified when setting up the database connection. You can change the naming by selecting *Table Name Format* from the *Query* menu.



*Query Builder Tables Window*

### 3.1.3.2.1.2. Joins

When you select database tables for the query, the Query Builder can auto-detect joins between column fields based on primary key-foreign key relationships in the database. Auto-joins will be added depending on which option you selected when setting up the database connection. Auto-joins will create a standard join between tables. A join is represented by a line drawn between two fields in the top half of the design window. To remove a join or edit its properties, right click on the line and select your choice from the pop-up menu. To add a join, click and drag one column field to another in a different table. A join will then appear. You can change the auto-join settings by selecting *Auto Join* from the Query menu.

#### **Join Properties:**

Selecting *Join Properties* from the pop-up menu will bring up three options allowing you to select the type of join used between the column fields. Query Builder only supports equi-joins. Inequality joins can be easily accomplished using the *conditions* field. You can specify inner joins, left outer joins, and right outer joins. See the examples below for an explanation of the different join types.

Suppose you have the following two tables: Customers and Orders

CustomerID	CustomerName
1	Bob
2	Ivan
3	Sarah
4	Randy
5	Jennifer

OrderID	CustomerID	Sales
1	4	\$2,224
2	3	\$1,224
3	4	\$3,115
4	2	\$1,221

An inner join on **CustomerID** on the two tables will result in combining rows from the **Customers** and **Orders** tables in such a way that each row from the **Customers** table will be “joined” with all the rows in the **Orders** table with the matching **CustomerID** value. Rows from the **Customers** table with no matching **CustomerID** fields from the **Orders** table will not be included in the query result set.

Now suppose you create a query by selecting the **OrderID**, **CustomerName**, and **Sales** fields with an inner join on the **CustomerID** field. The select statement generated by the Query Builder would look like this:

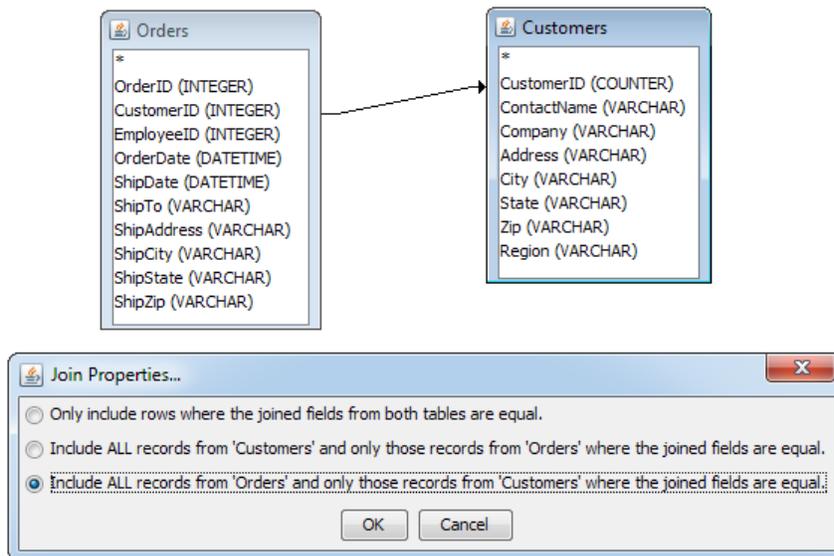
```
Select Orders.OrderID, Customers.CustomerName, Orders.Sales
From Customers, Orders
Where Customers.CustomerID = Orders.CustomerID
Order by Orders.OrderID;
```

The result of the query is shown below:

OrderID	CustomerName	Sales
1	Randy	\$2,224
2	Sarah	\$1,224
3	Randy	\$3,115
4	Ivan	\$1,221

As you can see, the **CustomerName** entries “Bob” and “Jennifer” do not appear in the result set. This is because neither customer has placed an order. There are situations where you may want to include all the records (in this example customer names) regardless whether matching records exist in the related tables(s) (in this case the **Orders** table). You can achieve this result using outer joins.

The Query Builder gives you the option of either right or left outer joins. The keywords “right” and “left” are not significant. It is determined by the order that the tables are selected in the Query Builder. If the outer table (the one that will have all records included regardless of matching join condition) is selected first, then Query Builder will use a right outer join. If the outer table is selected after the other join table, a left outer join is used. In our example, the **Customers** table has been selected before the **Orders** table, hence to select all of the records from the **CustomerName** field, the Query Builder will use a right outer join on the **CustomerID** fields.



*Join Properties Dialog*

Now, using the previous example, suppose you create the same query as before, except this time you specify to include all records from the **Customers** table. The select statement generated by the Query Builder would look like this:

```
Select Orders.OrderID, Customers.CustomerName, Orders.Sales
From Orders right outer join Customers on Orders.CustomerID =
Customers.CustomerID
Order by Orders.OrderID;
```

The result of the new query is shown below:

OrderID	CustomerName	Sales
	Jennifer	
	Bob	
1	Randy	\$2,224

OrderID	CustomerName	Sales
2	Sarah	\$1,224
3	Randy	\$3,115
4	Ivan	\$1,221

As you can see, all of the customer names have now been selected and null values have been inserted into the result set where there are no corresponding records. If you specify an outer join, the join line connecting the two tables in the Query Builder will become an arrow in the direction of the join.

### 3.1.3.2.1.3. Columns

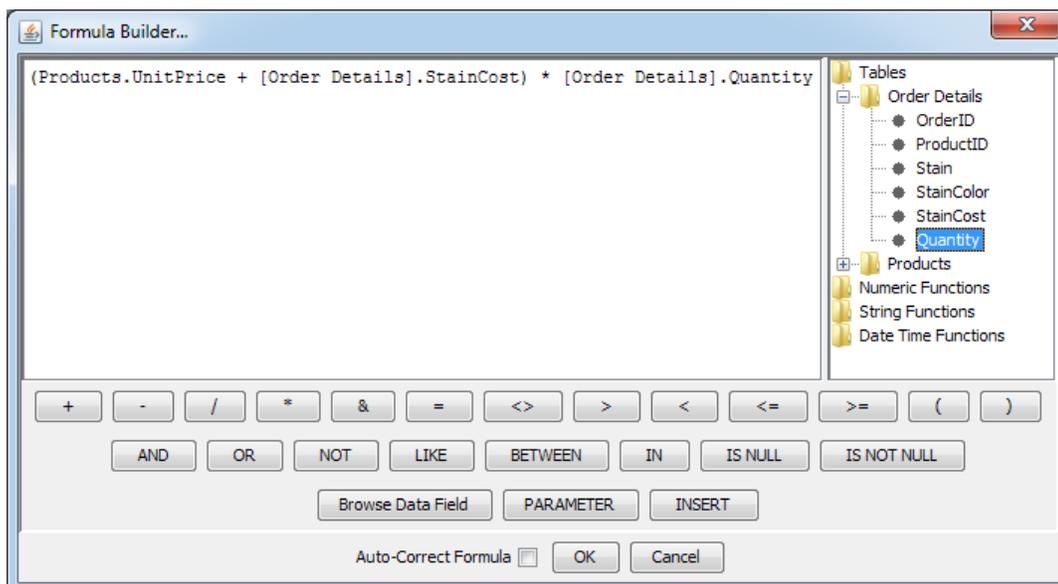
The QBE window contains information on column fields selected for the query, as well as any conditions for the selection.

#### Selecting Column Fields:

You can add column fields to the query from any table that has been selected in one of two ways. You can double-click on a field name within a table to add it to the query, or you can double-click on the *Table* or *Field* fields to bring up a drop-down menu with field choices. You can remove a column from the query by right clicking in the lower window and selecting *Delete Column* from the pop-up menu, or by selecting *Delete Column* from the Edit menu. Once you select a column field, you can specify how you want to sort the column, in ascending or descending order, by double clicking on the *Sort* field. You can also specify group by or column aggregation by double clicking on the *Aggregation* field. Aggregation options include: *Group By*, *Sum*, *Average*, *Min*, *Max*, *Count*, *Standard Deviation*, *Variance*, *First*, *Last*, and *Where*. If you select group by for one column, then you are required to specify group by (or aggregation) for all of the other columns. To specify a column alias, right click on the column and select *Alias* from the pop-up menu. You can perform a `SELECT DISTINCT` operation by selecting the *Select Distinct* option from the query menu.

#### Building Columns:

To build your own column, right click on a blank column in the QBE window and select *Build* from the pop-up menu. This will launch the Formula Builder. The Formula Builder allows you to construct columns in a visual environment using the tables that you have selected and the formula library for the database that you are using. You can click the *Browse Data Field* button to see the first few records of data for any field in your query.



Formula Builder Window

**Conditions:** You can place conditions on the query selection by entering them in the *Condition* or *Or* fields. A condition placed in the *Condition* field creates an AND clause within the generated SQL. A condition placed in the *Or* field creates an OR clause within the SQL. Right clicking in either field and selecting *Build* from the pop-up menu will bring up the Formula Builder. In the Formula Builder, you can specify standard conditions, =, <, >, BETWEEN, LIKE, NOT, etc., as well as construct formulas to filter the query. You can also specify a query parameter here.



### Note

ERES can auto-correct items entered as query conditions by appropriately appending the field name and encasing string arguments in quotes. For example, if you enter = ARC, ERES will change the query condition to `Categories.CategoryName='ARC'`. If you're using complex functions (i.e. database functions that take multiple string arguments), ERES may not be able to properly parse the function. You can turn off the auto-correct feature by unchecking the box at the bottom of the formula builder window.

#### 3.1.3.2.1.4. Using Database Functions

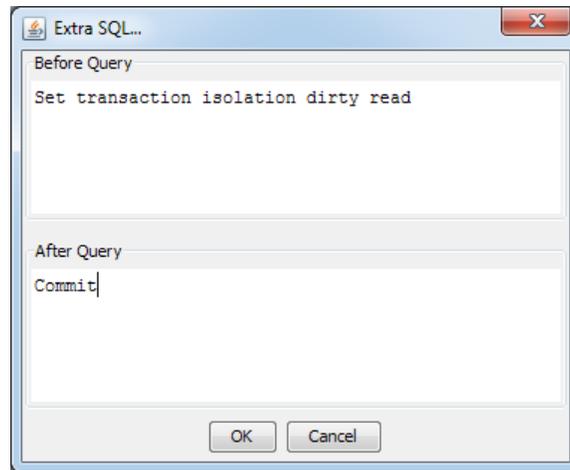
The formula builder component in the query builder allows you to use database specific functions when building a column or condition for the query. You can use the functions that are supplied or add your own to the interface.

ERES comes with the function libraries for Oracle, Access, MS SQL, and DB2 pre-loaded. They are stored in XML format in `DatabaseFunctions.xml` file in `userdb` directory. For databases with functions not stored in XML, ERES will use default ones. You can specify different database functions by editing the XML file or creating a new one based on `DatabaseFunctions.dtd` file in `userdb` directory. A sample database functions file might look like this:

```
<DatabaseFunctions>
  <Database ProductName="ACCESS">
    <FunctionSet Name="Numeric Functions">
      <Function>Abs(number)</Function>
      <Function>Atn(number)</Function>
    </FunctionSet>
  </Database>
</DatabaseFunctions>
```

#### 3.1.3.2.1.5. Adding Extra SQL

Sometimes it is necessary to add extra SQL statements to run before or after a query. For example, you may need to set a transaction level or call a stored procedure before executing a query and/or commit a transaction or drop a temporary table after executing a query. The query builder allows you to specify these extra SQL statements by selecting *Extra SQL* from the Query menu. This will bring up a window allowing you to write statements to execute before and/or after a query.



*Extra SQL Dialog*

You can enter any SQL statements you would like to run before and/or after the query in the appropriate boxes. Once you finish, click the *Ok* button and the statements will be added to the query.

### 3.1.3.2.1.6. Query Output

The *SQL View* and *Datasheet View* tabs allow you to see two different views of the query.

#### **SQL View:**

The *SQL View* tab shows you the SQL statement generated by the query in the design view. It allows you to see how the Query Builder is translating different operations into SQL. You can edit the generated SQL if you want, however, if you change the SQL and then return to the *Design View*, all changes you made will be lost. If you save a query after changing the SQL, the query will re-open in the *SQL View* tab if you select to edit it.

#### **Datasheet View:**

The *Datasheet View* tab shows you the query result in data table form (this tab is also available in the Enter SQL dialog). The datasheet view will show you all the data that is drawn as a result of executing the query. Going to the datasheet view will also test the query to check for design errors. You can navigate the query result by using the toolbar at the bottom of the window.



Go to the first page of the data table



Go to the previous page of the data table



Go to a specific row of data



Go to the next page of the data table



Go to the last page of the data table



Set number of rows to display per page (default is 30)

### Exporting Queries:

You can export queries in one of two ways. You can output the SQL statement as a text or you can output the query result as CSV file. To export a query, select *Export* from the File menu. A second menu will appear giving you the option to *Generate SQL* or *Generate CSV*. Select the desired option and a dialog box will appear prompting you to specify the filename and location.



### Note

To save the query and exit the Query Builder, select *Done* from the File menu.

### 3.1.3.2.2. Parameterized Queries

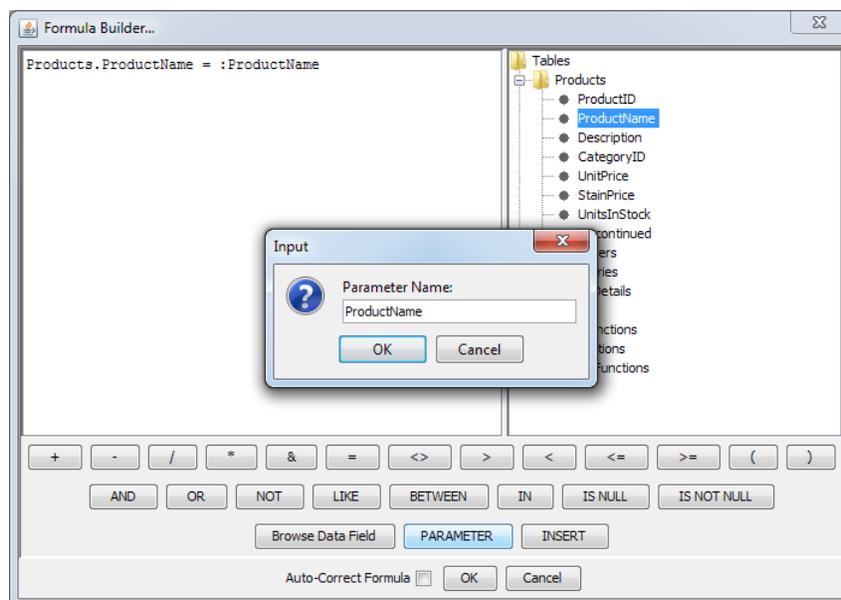
You can also use the Query Builder to design parameterized queries. This feature allows you to filter the data at run-time.

Query parameters can be defined when typing an SQL statement or using the Query Builder. They can also be defined when running data views (this is covered in the next section). A parameter is specified within an SQL statement by the " : " character. Generally, the parameter is placed in the WHERE clause of an SQL Select statement. For example, the following SQL statement

```
Select * From Products Where ProductName = :Name
```

specifies a parameter called Name. You can then enter a product name at run-time and only retrieve data for that product.

Within the Query Builder, you can specify a query parameter by right clicking on the *Condition* field and selecting *Build* from the pop-up menu. The Formula Builder will open, allowing you to place a condition on the column.



*Specifying a Parameter in the Formula Builder*

You can insert a parameter by clicking the *PARAMETER* button. A second dialog will appear prompting you to specify a name for the parameter. Type the parameter name, click *OK* and then click *OK* again to close the formula builder. You can specify as many different parameters for query as you like.

### 3.1.3.2.2.1. Multi-Value Parameters

ERES supports a special kind of parameter that takes an array of values as input rather than a single value. Multi-value parameters are useful when you want to filter the result set based on an unknown number of values. For example, say a report is run to return a list of customers for a specific state/province. Users could select as many different states/provinces as they wanted and return the relevant information.

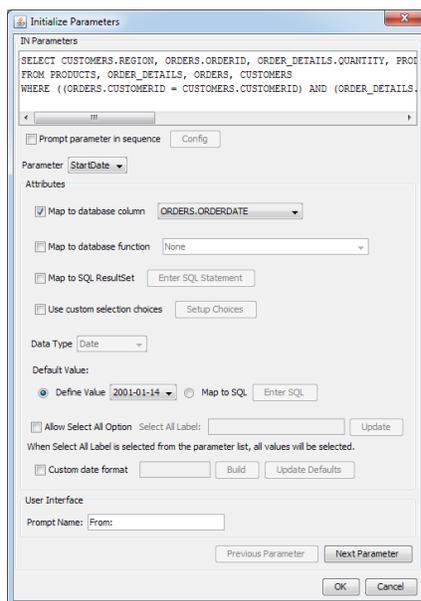
To create a multi-value parameter, place a parameter within an `IN` clause in an SQL statement. For example, the following query

```
Select Customers.Company, Customers.Address, Customers.City,
       Customers.State, Customers.Zip
From Customers
Where Customers.State IN (:State);
```

will create a multi-value parameter named `State`. Multi-value parameters will only be created if there is only one parameter in the `IN` clause. If you place more than one parameter in the `IN` clause, i.e. `Customers.State IN (:State1, :State2, :State3)`, it will create three single value parameters instead.

### 3.1.3.2.2.2. Initializing Query Parameters

When you attempt to save (by selecting *Done* from the File menu) or preview (by clicking the *Datasheet View* tab) a parameterized query, you will first be prompted to initialize the parameter. You can also initialize it by selecting *Initialize Parameters...* from the Query menu or by clicking the *Initialize Parameters* button in the Enter SQL Dialog.



*Initialize Parameter Dialog*

From this dialog you can specify the following options:

#### **Map to database column:**

This option allows you to specify a column from the database whose values will be used for the parameter input. Selecting this option modifies the parameter prompt that the end user will get when previewing or running the report in the Report Viewer. If you map the parameter to a database column, then the user will be prompted with a drop-down list of distinct values from which to select a parameter value. If you do not map, the user will have to type in a specific parameter value.

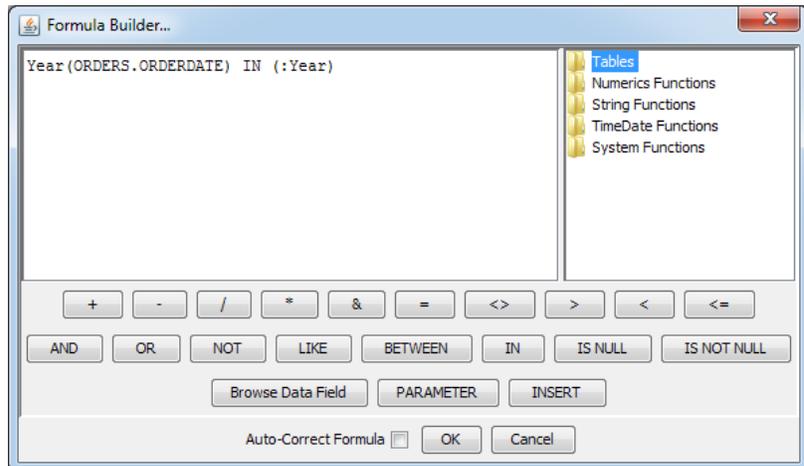


## Tip

Normally, this drop-down list is populated by running a select distinct on the column while applying the joins and conditions from the query. If you prefer to get all data from the column without constraints (sometimes this can improve performance of the parameter prompts), you can set the *Distinct Parameter List Selection* option in the Admin Console. This will result in selecting the distinct values of the mapped column from the table that contains the column. For more information about ERES server options, see Section 1.4.1.3 - Server Options.

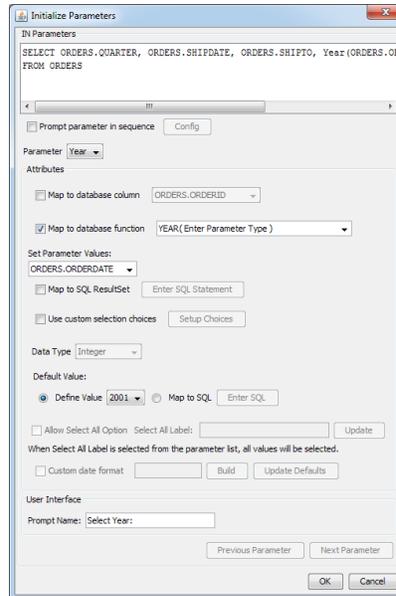
### Map to database function:

The *map to database column* feature is very handy if you want to enter a valid value for a parameter from a list box, but sometimes you rather want a computed value or a derived value from a database column. For example, you want to find all orders from year 2007. However, OrderDate is a date. What you want is to apply the *Year* function to the OrderDate column. This is the impetus behind this feature. Mapping a parameter to a database function is very similar to mapping to a column. In the formula builder, enter a condition comparing a function result to a parameter as shown below:



*Condition for Mapping to Database Function*

In the initialize parameter dialog, check the *Map to database* function checkbox and the values will be automatically filled in.



*Map Parameter to Database Function*

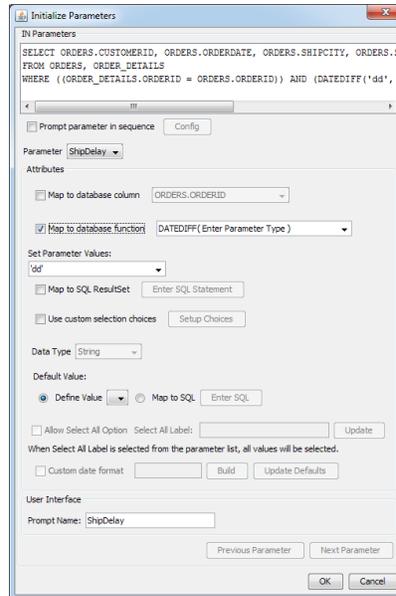
The list of custom functions is extracted from DatabaseFunctions.xml file located in <ERES Install>/userdb/ directory. Modify the .xml file if you wish to add a new database or custom functions. The new functions will appear in this list after you restart the program.

If your database is not listed in the .xml file, the function list will be populated with functions listed in the JDBC driver. However, the function parameters are not provided. For example, the HSQL database will not be listed in the .xml file.

An interesting example using the HSQL database is as follows. Suppose you would like to create a report for orders that were delayed. You can utilize the HSQL DateDiff function to find the number of days for the order to ship.

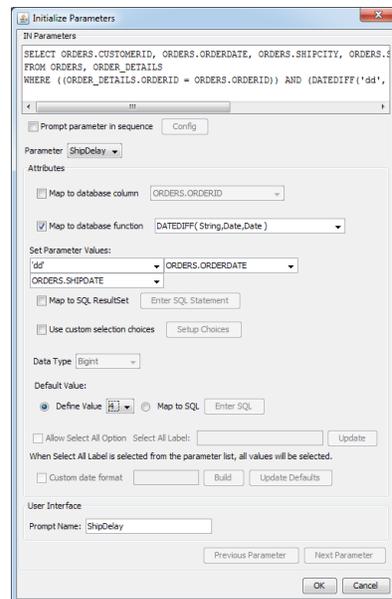
```
DATEDIFF('dd', ORDERS.ORDERDATE, ORDERS.SHIPDATE)
  >= :ShipDelay
```

This function finds the difference between the order date and the ship date and displays the result in terms of days. If you initialize the parameter and check the map to database function, the following window will be displayed:



*No Parameter Types for HSQL Function*

The `DateDiff` function requires a string and two date values for the parameters. Enter these parameter types in the parentheses. This will bring up three set parameter value lists. Enter `dd` (day) for the first parameter, select `Orders.OrderDate` from the list for the second parameter, and select `Orders.ShipDate` from the list for the third parameter. The default values will be updated with the function results.



*Map Parameter to HSQL Function*

### Map to SQL ResultSet:

A parameter mapped to a database column will give you a list of distinct values in a drop-down list box for the user to choose from when running the report. However, to produce the list of values, a select distinct on the column with the joins and conditions from the query will be run. In some cases, this can be a time-consuming process. To obviate this problem, and in fact gain complete control as to what and how to populate the drop-down list box, you can write your own select statement to populate the drop-down

list. An added bonus is that parameters that are in the query can be included in this query. With proper joins and parameters included, you can use this feature to facilitate cascading parameters (See Section 3.1.3.2.2.3 - Cascading Parameters). Below is an example:

Suppose you have two parameters in the query. So, your query is as follows:

```
SELECT CATEGORIES.CATEGORYID,
       PRODUCTS.PRODUCTNAME, PRODUCTS.UNITPRICE,
       PRODUCTS.UNITSINSTOCK
FROM PRODUCTS, CATEGORIES
WHERE ((PRODUCTS.CATEGORYID =
        CATEGORIES.CATEGORYID))
AND (((CATEGORIES.CATEGORYID =:category) AND
        (PRODUCTS.PRODUCTNAME =:product)))
```

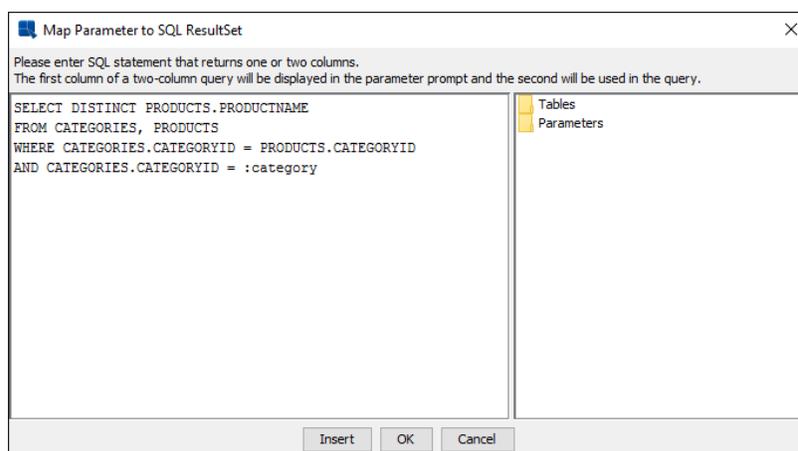
In the config prompt in `initialize` parameter, set the order for parameter prompting to `category` first, then `product`.

The select statement for parameter `category` can be the following:

```
SELECT DISTINCT CATEGORIES.CATEGORYID
FROM CATEGORIES
```

The select statement for parameter `product` will be as shown below:

```
SELECT DISTINCT PRODUCTS.PRODUCTNAME
FROM CATEGORIES, PRODUCTS
WHERE CATEGORIES.CATEGORYID = PRODUCTS.CATEGORYID
AND CATEGORIES.CATEGORYID = :category
```



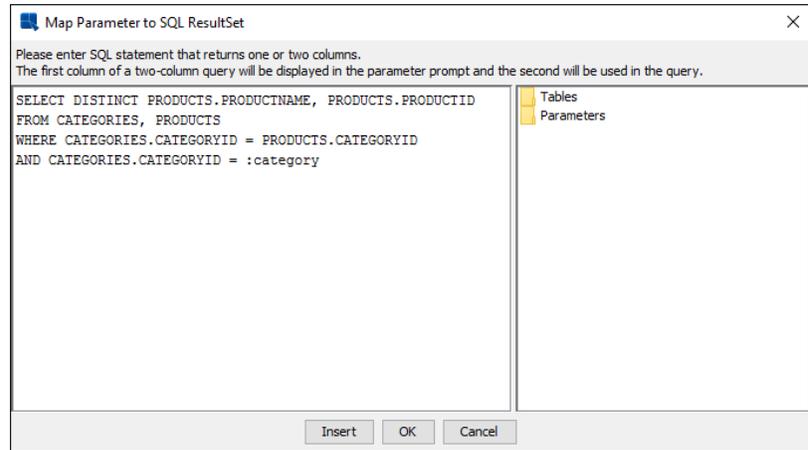
*Select Statement for Product*

When the user runs the template, `category` will be prompted first. Then the value of `category` chosen will be used to filter for `product`.

The select statement mapped to a parameter can have either one or two columns in the select list. It is clear that if one column is in the select list, it must be the column that supplies list of distinct values for the parameter. Another useful feature provided here is that you can actually select two columns

in the select list such that the first one of the columns will supply values for the drop-down list while the second column will be the actual parameter value for the filter condition. Consider the following example.

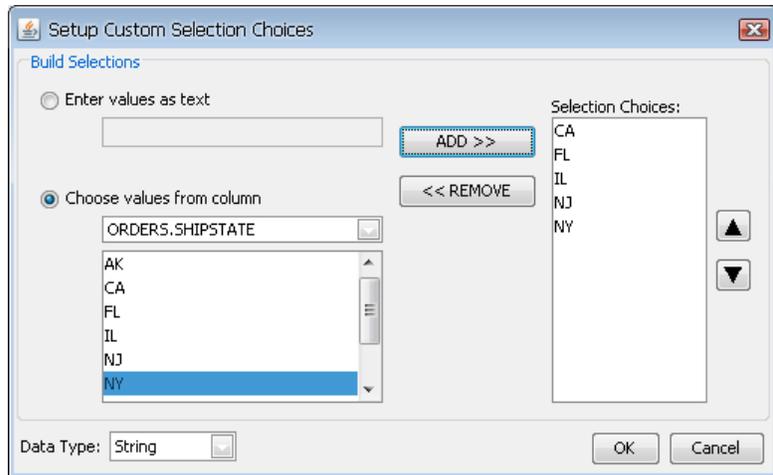
Suppose your database has a table with product ID as the primary key. When your end user wants to search for products from the database, they want to use the *product name* as parameter (therefore it is listed in the query as first) since a *product ID* could be just a cryptic code (therefore it is listed in the query as second). Using this feature, you can choose product name for the values in the drop-down list and product ID as the actual value filter condition.



*Select Statement with Two Columns*

**Use custom selection choices:**

Rather than having a drop-down menu with all the distinct column values, you can also create a custom list of parameter values. To create this list, select this option and click the *Setup Choices* button. This will launch a new dialog allowing you to create a list of choices.



*Custom Parameter List Dialog*

In this dialog, you can either enter custom values or select values from the distinct values of a column in the database. Once you finish specifying the values for the list, click the *OK* button and the choices will be saved.

**Default Value:**

This allows you to specify a default value for the parameter. Although you don't have to specify a default value, it is recommended that you do so. If

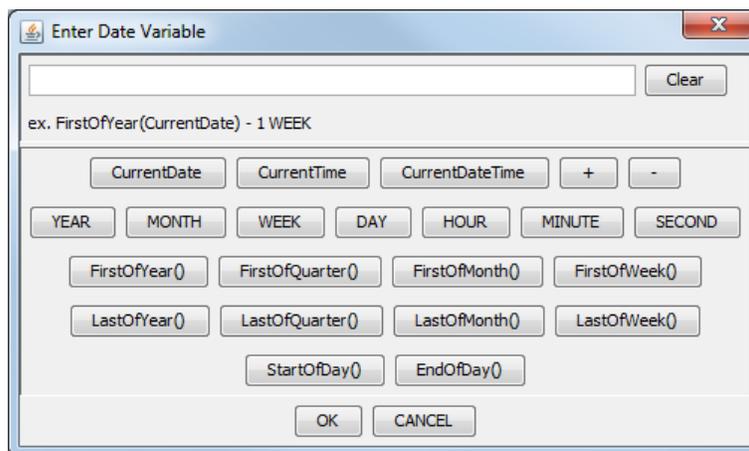
you do not supply a default value, you cannot open or manipulate the report template without the data source present.

You can either select a single value manually (either choose it from a list or type it manually, it depends on the mapping method you chose) or map the default value to a SQL query.

For multi-value parameters (see Section 3.1.3.2.2.1 - Multi-Value Parameters), the SQL query can return more than one value. In such case, several values will be chosen as default parameter values.

### Date Variable:

This option is only available when the parameter is not mapped to a database column or function, or it's mapped to a SQL resultset and not set to a custom selection choice. This option is only intended for parameters with variable type date/time. When you click this button, the following panel will pop up, listing all the supported keywords.



*Enter Date Variable Dialog*

This dialog allows you to select one of the three keywords: `CurrentDate`, `CurrentTime`, and `CurrentDateTime`. You can add or subtract units of time from the current date/time, allowing you to have a dynamic date range. For example, a report can have the following default values:

```
StartDate: CurrentDate - 1 WEEK
EndDate: CurrentDate
```

This would indicate that every time the report is run, the default prompt should be one week ago to the current date. Other supported time units are `YEAR`, `MONTH`, `DAY`, `HOUR`, `MINUTE`, and `SECOND`. This feature only supports a single addition or subtraction, it does not support multi-value parameters.

You can also use functions to define the parameter value:

#### **FirstOfYear()**

Argument format: `CurrentDate`, `CurrentDateTime`,  
e.g.  
`FirstOfYear(CurrentDate)`

This function returns a date of the first day of the year from the argument. For example, when the argument evaluates to `2012-08-14`, the function returns `2012-01-01`.

---

<b>LastOfYear()</b>	Argument format: CurrentDate, Current-DateTime, e.g. LastOfYear(CurrentDate)  This function returns a date of the last day of the year from the argument. For example, when the argument evaluates to 2012-08-14, the function returns 2012-12-31.
<b>FirstOfQuarter()</b>	Argument format: CurrentDate, Current-DateTime, e.g. FirstOfQuarter(CurrentDate)  This function returns a date of the first day of the quarter which includes the date from the argument. For example, when the argument evaluates to 2012-08-14, the function returns 2012-07-01.
<b>LastOfQuarter()</b>	Argument format: CurrentDate, Current-DateTime, e.g. LastOfQuarter(CurrentDate)  This function returns a date of the last day of the quarter which includes the date from the argument. For example, when the argument evaluates to 2012-08-14, the function returns 2012-09-30.
<b>FirstOfMonth()</b>	Argument format: CurrentDate, Current-DateTime, e.g. FirstOfMonth(CurrentDate)  This function returns a date of the first day of the month from the argument. For example, when the argument evaluates to 2012-08-14, the function returns 2012-08-01.
<b>LastOfMonth()</b>	Argument format: CurrentDate, Current-DateTime, e.g. LastOfMonth(CurrentDate)  This function returns a date of the last day of the month from the argument. For example, when the argument evaluates to 2012-08-14, the function returns 2012-08-31.
<b>FirstOfWeek()</b>	Argument format: CurrentDate, Current-DateTime, e.g. FirstOfWeek(CurrentDate)  This function returns a date of the first day of the week which includes the date from the argument. For example, when the argument evaluates to 2012-08-14, the function returns 2012-08-12 (Sunday is considered as the beginning of the week).
<b>LastOfWeek()</b>	Argument format: CurrentDate, Current-DateTime, e.g. LastOfWeek(CurrentDate)  This function returns a date of the last day of the week which includes the date from the ar-

---

gument. For example, when the argument evaluates to 2012-08-14, the function returns 2012-08-18 (Saturday is considered as the end of the week).

**StartOfDay()**

Argument format: CurrentTime, Current-DateTime, e.g. StartOfDay(CurrentDateTime)

This function returns a time of the start of the day from the argument. For example, when the argument evaluates to 2012-08-14 12:15:03, the function returns 2012-08-14 00:00:00.0.

**EndOfDay()**

Argument format: CurrentTime, Current-DateTime, e.g. EndOfDay(CurrentDateTime)

This function returns a time of the end of the day from the argument. For example, when the argument evaluates to 2012-08-14 12:15:03, the function returns 2012-08-14 23:59:59.999.

**Data Type:**

This allows you to specify data type for the parameter value(s). If you mapped the parameter to a column, the data type is set automatically.

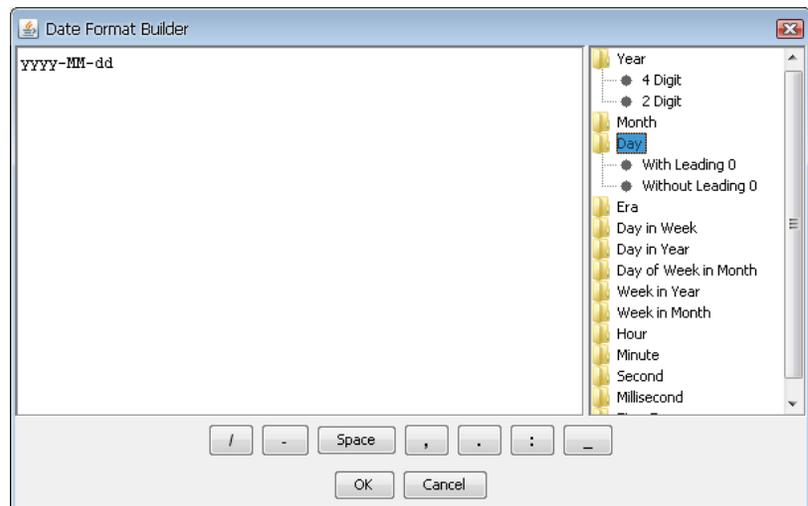
**Allow Select All Option:**

Use this to add an option to the parameter prompt dialog that will allow users to select all parameter values even for single-value parameters. See the Section 3.1.3.2.2.4 - All Parameters for more details.

**Custom Date Format:**

This allows you to set the format in which the date parameter should be entered. This option is only available when the data type is either date, time, or timestamp and the parameter is either mapped to column or not mapped at all.

When you check this option, the default custom date format is shown. These defaults can be changed in the admin console (See Section 1.4.1.3 - Server Options). Please note that the default date format only applies to database data sources. The date format is built using a combination of characters that represent date/time elements. You can build the format easily using the date format builder by clicking the *Build* button.



*Date/Time Format Builder*

The builder contains a list of elements available on the right. You can mouse over the elements to see an example of each presentation. The bottom section contains a set of separators available for use.

You can also type in the format by hand either in the builder or directly in the Custom Date Format input box. For more information about characters and their formatting, please refer to the documentation for the **printDate()** report function in Section 4.1.6.2.8.3 - Date/Time Functions. Formatting for this option is the same as for the format argument of the function.

**Prompt Name:**

This allows you to specify the prompt that is given to the user in the parameter dialog.

If you map the parameter, the user will see either a drop down box (single value parameter) or a list box (multi-value parameter) containing various options. If you choose not to map the parameter, the user will see a textbox to enter their own value. In case of a multi-value parameter, it is recommended to let the user inform in the parameter prompt that this parameter accepts multiple values. Users can separate multiple values using a comma (e.g. ARC, DOD, TRD). If the text requires the use of comma, the user can use quotes to include the comma within the filter string (e.g. "Doe, John", "Smith, Mike").

Clicking the *Previous Parameter* and the *Next Parameter* buttons allows you to initialize each of the parameters that have been defined in the query.

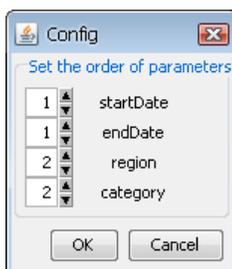
When you select to use a parameterized query to design a report, or open a report that uses a parameterized query, the report will load/start with the default values. You will be prompted to provide parameter values when you preview the report.

### 3.1.3.2.3. Cascading Parameters

By default, the user is prompted to enter all report parameters at once in the prompt dialog. This configuration, however, may not be the best approach if some parameters are mapped to database columns with a significant number of distinct values. It can be difficult to select from a very large list and depending on the parameter combination, users may be able to select parameters that do not return any data.

To assist with these problems, ERES provides a feature that allows the user to configure the order in which the parameters should be entered. With this feature enabled, the user enters parameters in the dialog in a pre-defined order. As such, each selection will be applied as a filter to the next parameter prompt(s). Using cascading parameters can limit the number of distinct values presented to the user and can prevent the user from selecting invalid parameter combinations.

To enable cascading parameters, check the option marked *Prompt parameter in sequence* in the parameter initialization dialog. Then click the *Config* button to set the order of the parameter prompts. A dialog will open showing all the parameters defined in the query.



*Parameter Sequence Dialog*

Using the spin boxes, you can set the sequence for the query parameters. User will be prompted to start with the lowest numbered parameter and work his/her way up to the highest one. If two or more parameters share the same number, the user will be prompted to enter those parameters at the same time (in the same dialog).

By default, the parameter values for the next level are generated by rerunning the entire query with the previously prompted parameters filled in. If the original query is slow to execute, you can improve performance by mapping

higher order parameters to SQL Queries. You can even include previously selected parameter values in the mapped query. For more information, please see Section 3.1.3.2.2.2 - Initializing Query Parameters.

Please note that this feature can only be used with reports.

### 3.1.3.2.2.4. All Parameters

Sometimes you want to select all parameter values at once. The *All Parameters* feature allows you to do so.

#### Single-value parameters

It is possible to select all parameter values at once, even for parameters that do not allow multi-value selection.



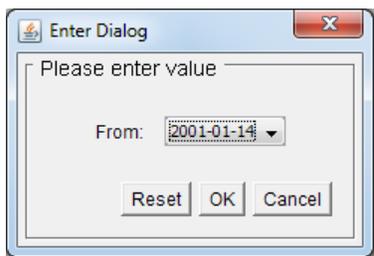
**Note**

There is a difference between multi-value selection and all-value selection. See the Inner Workings chapter to learn more.

**For example:** Let's assume you have a condition like this:

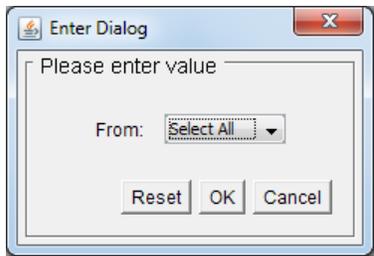
```
WHERE column = :Parameter
```

In such case, the parameter prompt dialog will not allow you to select more than one value.



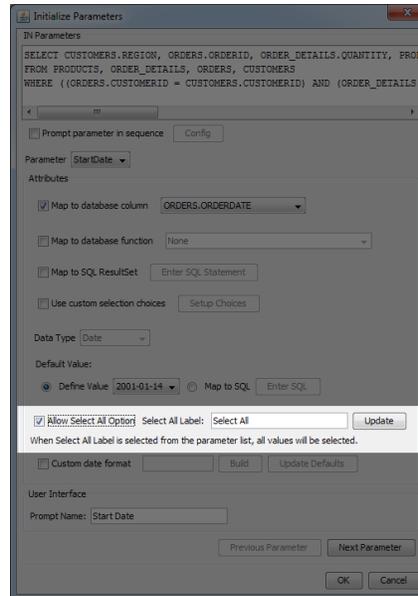
*Typical single-value parameter*

But you can use the *Select All Values* feature to add an option to the parameter value list that will allow viewers to select all parameter values at once (even if the parameter does not allow multi-value selection).



*Single-value parameter with the Select All functions enabled*

The *Select all* feature can be enabled in the *Initialize Parameters* dialog (see Section 3.1.3.2.2.2 - Initializing Query Parameters for more details) by selecting the *Allow Select All Option* checkbox.



This option is only available for parameters that meet the following requirements:

1. The parameter uses one of the following operators. If there are multiple occurrences of this parameter in the query, all parameter comparison operators have to be one of these:

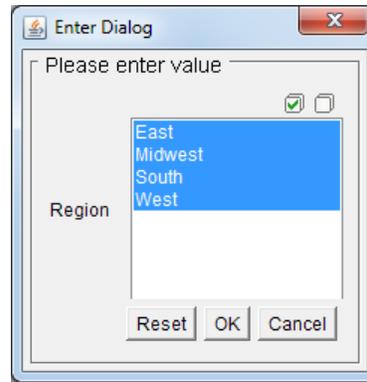
- < less than
- <= less than or equal to
- > greater than
- >= greater than or equal to
- = equal to

2. The parameter is mapped to the same column as the column from the parameter condition **or** the parameter isn't mapped to anything.

After you select the *Allow Select All* option, the *Select All Label* field activates, allowing you to enter a text that will be used for selecting all data from the query. For parameters that are mapped to a column, this text will be displayed in the parameter value list in 1st place. For parameters that aren't mapped to anything, entering this text as the parameter value will result in selecting all data.

### Multi-value parameters

Unlike single-value parameters, the *Select All* feature is enabled for all multi-value parameters by default (in fact, it can't be disabled, because disabling it for multi-value parameters would make no sense). All you have to do to use this feature is to click on the  *Select All* icon in the parameter prompt.



However, multi-value parameters can work in two modes:

1. If the parameter meets the conditions from the previous paragraph and the parameter is on the first cascading level (i.e. parameter cascading is disabled or the parameter is on the first cascading level), it is parsed by the SQL parser and the parameter condition is nullified. Nullifying the parameter optimizes the query and prevents it from causing performance issues or even errors. See the Inner Workings to learn more about how it works.
2. If the parameter doesn't meet the conditions from the previous paragraph or if it's not on the first cascading level, selected values will be injected to the query as a list of values separated by comma. If there is a large amount of values injected to the query as a list, the query can become quite long. Long queries can cause performance issues or even errors, so it is **not recommended** to use this option for parameters with many values.

### Inner Workings

If a report/chart/map viewer chooses to select all parameter values for a single-value parameter or for a multi-value parameter that meets the conditions for parameter disabling, the query is then automatically parsed and a special condition is added to the parameter which basically disables the parameter.

**For example:** The following query

```
select *
from table
where column > parameter_value
```

Would be parsed and passed to the database as:

```
select *
from table
where ((column > parameter_value) OR (1 = 1))
```

This example also demonstrates another important thing: selecting all values for the < (less than) or > (greater than) operators returns all values from the table (if there are no other conditions) rather than returning no data at all (because condition like WHERE <all data from the Date column> > Date would return no data...).

Because ERES allows you to use many database systems, parsing may fail for certain complex queries in certain databases. In such case a warning dialog will be displayed.

In such situations, you have the following three options:

1. Try to modify the query so it can be parsed by our parser.
2. Add your own *Select all parameters* condition to the query.

**For example:**

```
WHERE ((column = :Parameter) OR (:Parameter
LIKE 'selectall'))
```



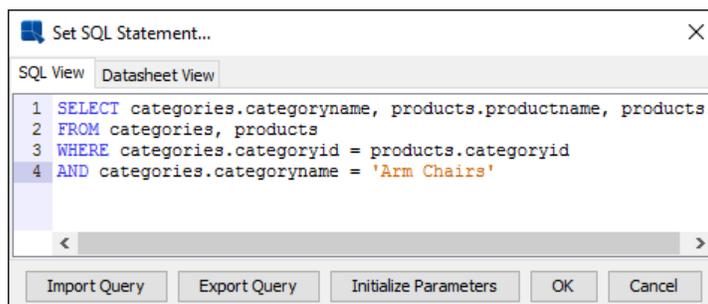
### Note

If you embed all parameters directly to the query, leave the *Allow Select All Option* option **disabled**.

3. Contact Quadbase support [ <https://www.quadbase.com/support-overview/> ].

### 3.1.3.2.3. Entering SQL Statements

Typically, the Query Builder is recommended for creating queries. However, there are cases when it is necessary to enter SQL statements directly: for example if the query is already created in a QRY file, if the query is built into a stored procedure/function, or if the query requires commands not supported by the Query Builder. In these situations, select *Enter SQL statement* to open the Set SQL Statement window. Here you can enter SQL statements directly into the text area as shown below or you can load an existing QRY File.



*Enter SQL Statement Dialog*

To preview the result set, click on the *Datasheet View* tab.

#### 3.1.3.2.3.1. Calling Oracle Stored Procedures

Compared to other database systems, Oracle uses a different approach when it comes to stored procedures and functions. For example, on MS SQL Server, using the EXEC command will return a result set. However, Oracle requires the use of an OUT parameter with a REF CURSOR type to return the result set. In addition, Oracle will not accept multiple statements from a single query. Therefore, it is necessary to store the query within a stored function and use special syntax to access the existing Oracle stored procedures.

To access your Oracle stored procedures, the first step is to define a weakly typed REF CURSOR using the following PL/SQL statement:

```
CREATE OR REPLACE PACKAGE types
AS

    TYPE ref_cursor IS REF CURSOR;

END;
```

This `ref_cursor` type will be used to store the query result set and return as an OUT parameter. The next step is to create a function which calls your stored procedure and executes your query. The following skeleton code will return a simple query using the `ref_cursor` type.

```
CREATE OR REPLACE FUNCTION my_function()

    RETURN types.ref_cursor

AS

    result_cursor types.ref_cursor;

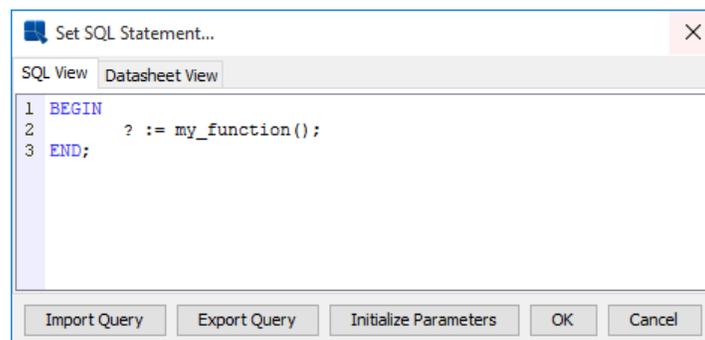
BEGIN

    do_stored_procedure();
    OPEN result_cursor FOR
        SELECT * FROM Categories

    RETURN result_cursor;

END;
```

Now that the Oracle stored function is set up, it can be easily called from ReportDesigner using a special PL/SQL like syntax. In the Set SQL Statement window, enter the following syntax to call the Oracle stored function:



*Calling simple Oracle stored function*

The `BEGIN . . . END;` syntax alerts the system that the user is trying to access an Oracle stored function. And the “?” notifies the ReportDesigner that a variable is reserved for the OUT parameter. The JDBC syntax for calling Oracle stored procedures is as follows:

```
( call ? := my_function() )
```

However, ERES does not support this format. Preview the results by clicking the *Datasheet View* tab.

Here is a more practical example to illustrate how stored procedures can be used with ERES to develop useful solutions. Suppose you have a table called *employee\_table* that stores an organization's location hierarchy such as the one shown below:

ID	NAME	PARENT	EMPLOYEE
1	All	NULL	0
2	America	1	0
3	Europe	1	0
4	New York	2	20

## Data Sources

ID	NAME	PARENT	EMPLOYEE
5	Santa Clara	2	30
6	Dallas	2	12
7	London	3	14
8	Paris	3	11

The table lists the various corporate locations in a tree structure. The numbers of employees are stored in the leaf nodes (e.g. New York, London, etc.) and each node contains information about its immediate parent. Suppose you want to create a report that displays the number of employees in a certain region and information about the separate branches within that region. For example, if the user inputs ID = 2 (America), you want the report to display the total number of employees in America along with the branch locations. Using Oracle's CONNECT BY and START WITH clauses, the problem can be solved with two simple Oracle Stored Functions:

```
CREATE OR REPLACE FUNCTION sum_employees(locID IN NUMBER)

    RETURN NUMBER

AS

    sum_emp NUMBER;

BEGIN

    SELECT sum(employee) INTO sum_emp
    FROM employee_table
    CONNECT BY PRIOR id = parent
    START WITH id = locID;

    RETURN sum_emp;

END;

CREATE OR REPLACE FUNCTION regional_employees (locID IN NUMBER)

    RETURN types.ref_cursor

AS

    result_cursor types.ref_cursor;

BEGIN

    OPEN result_cursor FOR
        SELECT id, name, sumEmployees(id) AS Employees
        FROM employee_table
        CONNECT BY prior id = parent
        START WITH id = locID;

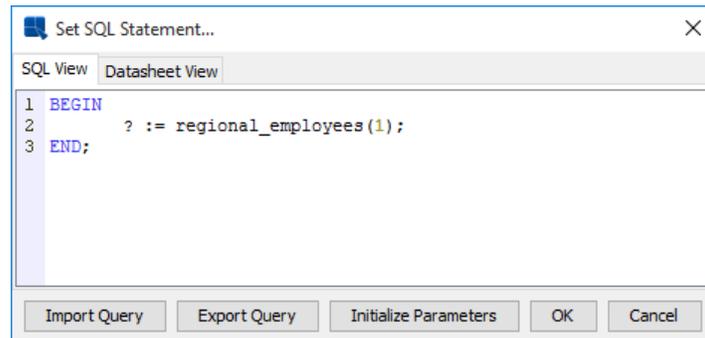
    RETURN result_cursor;

END;
```

The function `sum_employees` takes the starting node as an argument and finds the sum of all leaf nodes that are descendants of that node. For example, `sum_employees ( 3 )` returns 25 because there are 25 employees in Europe (14 in London, 11 in Paris). The second function, `regional_employees`, traverses through the tree structure

starting with the locID and builds a result set from the ID, Name and the result from the sum\_employees function. The result set is then returned through a REF CURSOR.

To call a stored function that requires an argument, enter the following statements in the Set SQL Statement window:



*Calling regional\_employees function*

Preview the results by clicking the *Datasheet View* tab.

ID	NAME	EMPLOYEES
1	All	87
2	America	62
4	New York	20
5	Santa Clara	30
6	Dallas	12
3	Europe	25
7	London	14
8	Paris	11

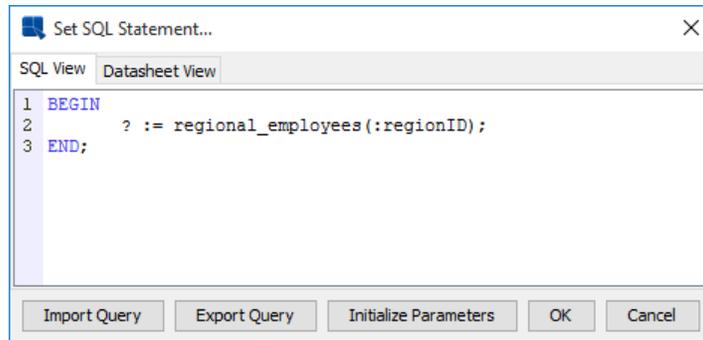
*Result set from regional\_employees*

As seen in the results, the CONNECT BY clause traverses the tree recursively listing the American nodes together before listing the European nodes. If the user is only interested in European locations, they can enter 3 for the parameter and the following result set would return this:

ID	NAME	EMPLOYEES
3	Europe	25
7	London	14
8	Paris	11

*Result set from regional\_employees in Europe*

To create a parameterized report, use the :param\_name syntax. The SQL parser in ERES will be able to differentiate between the colon used for parameters and the one used for the assignment operator ( := ). Here is an example of using the parameters:



*Calling Oracle Stored Function using Parameter*

When using IN parameters, it is necessary to initialize the parameters prior to executing the query. It is especially important to set the correct default data type for executing stored procedures because the parameters cannot be mapped to existing columns. You can find more information about initializing parameters in Section 3.1.3.2.2 - Parameterized Queries.

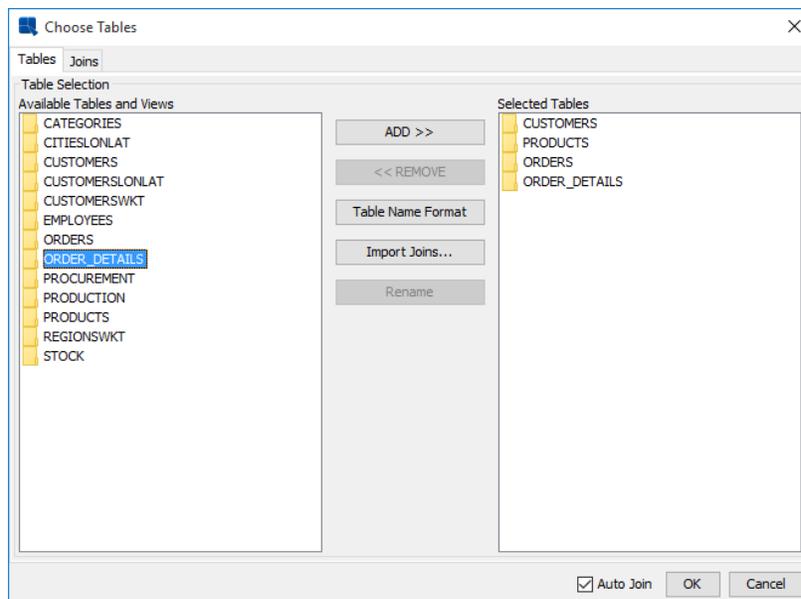
To try this example, <ERESInstall>\help\examples\DataSources\database\locationHierarchyExample.sql contains SQL commands to create employee\_table as well as two stored functions.

### 3.1.3.3. Data Views

In addition to the query interfaces, ERES provides another way of retrieving database data - data views. Data views provide a simplified view of the database in which users can design queries by simply selecting fields without using the Query Builder or having any knowledge of the underlying database structure. Using data views, administrators can predefine tables, joins, and fields, creating a local schema for the user to select from.

For example, an administrator could set up a data view for the sales department. The appropriate database tables and fields are pre-selected and grouped in a manner congruent with business users' logic. For example a group called *invoices* would have the appropriate customer and order fields. End users would then select this data view, pick the pertinent fields, specify a date range, and then begin designing a report or chart.

To create a data view, select the *Data Views* node in the Data Source Manager window and click the *Add* button. A new window will open allowing you to select the database tables you want to use for the data view.

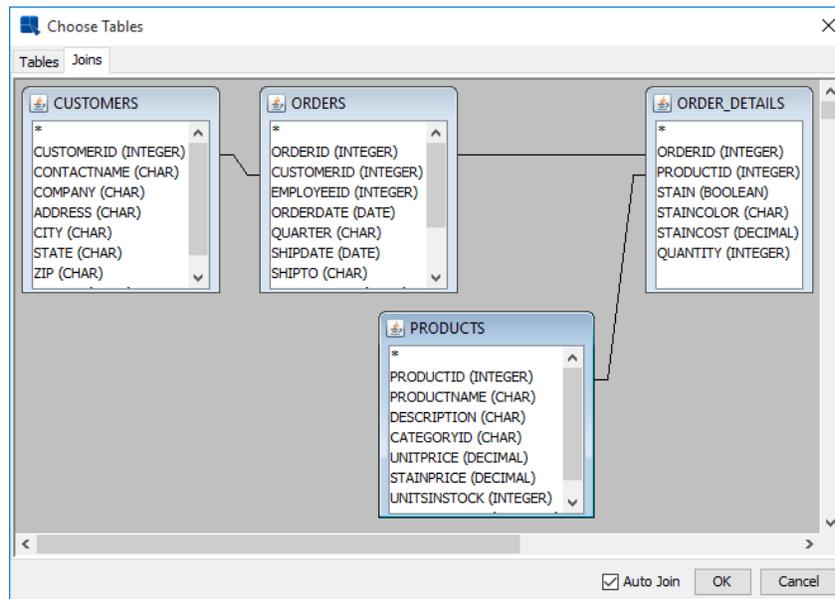


*Data View Choose Tables Dialog*

Left window contains all available database tables and views. You can add a table by selecting it in the left window and clicking the *ADD>>* button. By default, the data view will use the name format you specified when setting up the database connection. You can change the naming by clicking the *Table Name Format* button, or specify a

table alias by clicking the *Rename* button. You can also import selected tables and joins from another data view by clicking the *Import Joins...* button.

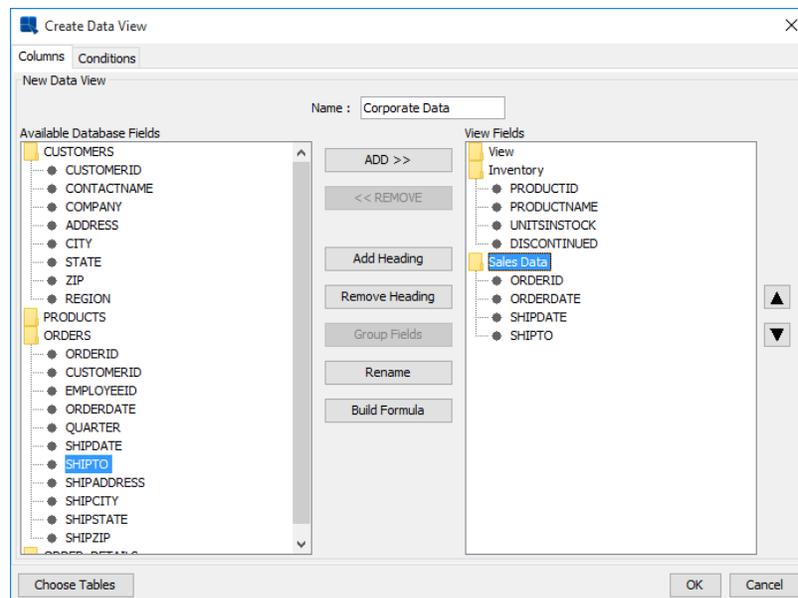
The *Joins* tab of this window allows you to specify the joins between the selected tables.



*Data View Joins Dialog*

The *Joins* tab shows all selected tables and their associated fields. The tables will be auto-joined depending on which option you selected when setting up the database connection. These auto-joins create a standard join between tables represented by a line. To remove a join or edit join properties, right click on the line and select your choice from the pop-up menu. To add a join, click and drag one column field to another in a different table. A join will then appear. Data views use the same join properties as the Query Builder. For more information about join properties, please see Section 3.1.3.2.1.2 - Joins.

After you finish selecting and joining tables, click the *OK* button and a new window will open allowing you to construct the data view.



*Create Data View Dialog*

The left window contains a list of tables you have selected and their associated fields. Each folder represents a table and can be opened and closed by double clicking. The right window contains fields that have been selected

for the data view. To add a field to a data view, select it in the left window and click the *ADD>>* button. Fields can be removed from the data view in the same way by selecting a field in the right window and clicking the *<<REMOVE* button. You can create a calculated column by clicking the *Build Formula* button. This will open the formula builder, allowing you to build the column. You can also define an alias by selecting any of the view fields in the right window and clicking the *Rename* button.

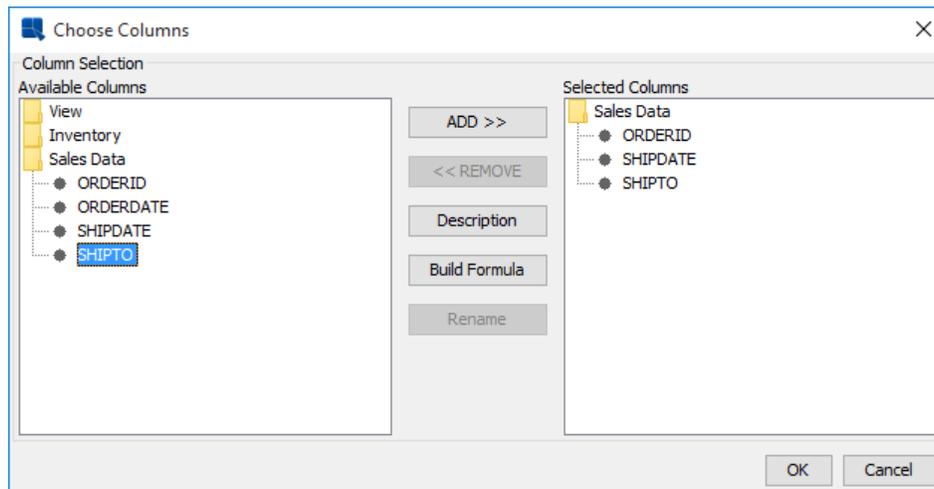
You can also group fields within the data view by adding headings. This allows you to create your own organizational structure of *virtual tables* that group data from different database tables under one heading. To create a heading, click the *Add Heading* button. You will then be prompted to specify a name for the heading. The new heading will then appear as a folder in the right window. To add fields under a heading, first select the fields you want to add from the right window and click the *Group Fields* button. You will then be presented with a drop-down menu, allowing you to select the heading under which you would like to add the fields.

The *Conditions* tab contains a formula builder window that allows you to specify certain filtering criteria for end users. Anything added in this window will be added to the *Where* clause of the generated SQL. For more information about using the formula builder, please see Section 3.1.3.2.1.3 - Columns.

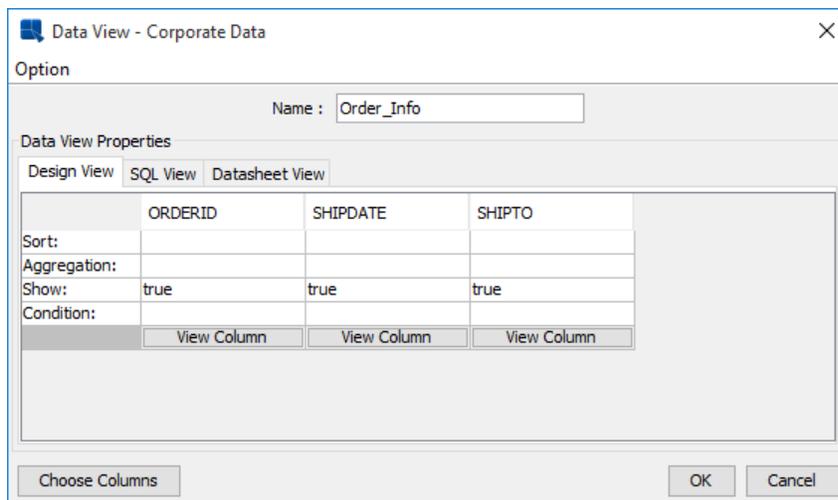
Once you finish creating the data view, click the *OK* button and the data view will be added to the Data Source Manager. Users can now use this view to construct ad-hoc queries.

When you design a report or chart using a data view as the data source (by selecting the data view and clicking the *Create Report* or *Create Chart* button), a window will open allowing you to select which fields in the view you want to use for the report. From this dialog, you can also build computed fields based on the available view columns.

After you select the fields, click the *OK* button and a new window will open allowing you to specify sorting, aggregation, and filtering conditions for the data view.



*Data View Choose Fields Dialog*



*Data View Conditions Window*

You can specify sorting, aggregation, and conditions for every field in the data view by double clicking on the respective field. Sorting and aggregation can be selected from drop-down menus. Double clicking on the *Conditions* field brings up a new window that allows you to specify simple selection criteria like  $>$ ,  $<$ ,  $=$ , and *between*. Users can build more advanced filtering criteria by right clicking on the *Conditions* field and selecting *Build* from the pop-up menu. This will open the Formula Builder window allowing you to build a condition. You can also display all of the unique values in the column by double clicking on the *View Column* button.

The Option menu in the upper left corner of the conditions window allows you to select a vertical/horizontal view for the conditions window, initialize any parameters in the data view, or save the query.

The selection set and conditions that you specify will be saved as a data view query with the name that you specify in the name field. Data view queries are saved under the node for the data view. A report created from the data view will refer to the data view query for updating/modification.

### 3.1.3.3.1. Data View Parameters

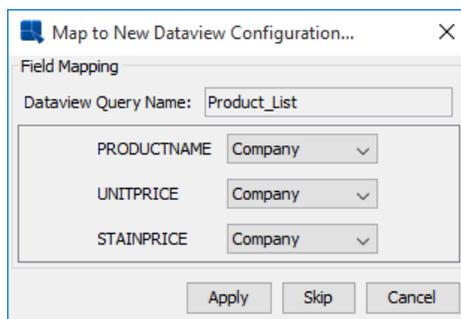
As with Query Builder, users can specify query parameters in Data Views. To add a parameter to a data view, select a data view in the Data Source Manager and click *View* to run the data view. After you select fields for the data view and you are in the conditions window, right-click in the *Condition* field for a column and select *Build* from the pop-up menu. This will bring up the formula builder, allowing you to specify a parameter in the same way as in Query Builder. For more information about this, please see Section 3.1.3.2.2 - Parameterized Queries.

Once you enter the parameter, you will be prompted to initialize it if you go to the *Datasheet View* tab and then click *Ok* to continue with the report wizard, or if you save the selections as a query. You can also initialize the parameter by selecting *Initialize Parameters* from the Option menu.

### 3.1.3.3.2. Updating Data View Queries

Sometimes you may need to make changes to the structure/make-up of the data view as your data model or requirements change. Changes could include adding/removing fields or re-naming them. You can propagate changes from the data view to its associated queries by selecting it in the data source manager and selecting *Data View Queries* from the Update menu.

All of the queries associated with the view will be scanned and any inconsistencies in fields or field names will be presented for you to update.



*Update Query Fields Dialog*

For each query, you will be prompted to change any fields that no longer match the data view structure. For each field, you can select a field from the data view to map it to, or remove the field from the query. If you do not want to change anything in the query, you can click the *Skip* button. The query will continue to run, but it will refer to the old data view structure. Click the *Apply* button to save the changes to the data view query.

### 3.1.3.3.3. Data View Security

EspressReport ES allows you to filter data based on a user's security level when viewing a pre-built report/chart. But what if the user is doing ad hoc querying and reporting and you want to restrict data access based on user's security level? For this, EspressReport ES allows you to apply security filters to data views. This feature will automatically apply preset filters to data views and data view queries when certain users run them through QuickDesigner Report-s/QuickDesigner Charts. These options can be configured via XML file and tie into the report-based secured parameter features. For more information about secured report parameters, see Section 4.1.10.2.1 - Security Parameters.

To illustrate the concept, here is a simple example. A sample data view security XML file is displayed here:

```
<?xml version="1.0"?>
<!DOCTYPE SecuredParamInfoList SYSTEM "QuickDesignerSecuredParameters.dtd">
<SecuredParamInfoList>

  <SecuredParamInfo>
    <Driver>org.hsqldb.jdbcDriver</Driver>
    <URL>jdbc:hsqldb:<ERESInstallDir>\help\examples\DataSources
\database\woodview</URL>
    <TableName>CUSTOMERS</TableName>
    <ColumnName>REGION</ColumnName>
    <DataType>VARCHAR</DataType>
    <IsMulti state="true" />
    <SecLevel Name="EastSouth" GrantAll="false">
      <Data>East</Data>
      <Data>South</Data> </SecLevel>
    <SecLevel Name="Midwest" GrantAll="false">
      <Data>Midwest</Data> </SecLevel>
    <SecLevel Name="West" GrantAll="false">
      <Data>West</Data> </SecLevel>
    <SecLevel Name="East" GrantAll="false">
      <Data>East</Data> </SecLevel>
    <SecLevel Name="South" GrantAll="false">
      <Data>South</Data> </SecLevel>
    <SecLevel Name="Executive" GrantAll="true">
      </SecLevel> </SecuredParamInfo>

</SecuredParamInfoList>
```

This XML file defines a filter for the *REGION* column from the *CUSTOMERS* table in the Woodview sample database. It specifies behavior of the filter for six different security levels. For level *East*, the query will be automatically filtered so that only rows where region is *East* will be returned. For level *South*, the query will be automatically filtered so that only rows where region is *South* will be returned. For the level *Midwest*, only rows where the region is *Midwest* will be returned. For the level *West*, only rows for *West* region will be returned. The *IsMulti* attribute allows you to indicate whether multiple parameter values can be specified for the filter, like in the case of the *EastSouth* level. For level *Executive*, all values from the region column will be available.

The security levels specified in the XML file should match those that are defined in the Organizer. For more information about setting up security levels, see Section 2.3.3 - Security Levels. The sample XML file — `SampleQD-SecuredParameters.xml` and the DTD file - `QuickDesignerSecuredParameters.dtd` that defines all of the specifications for the XML file can be found under `<ERESInstallDir>/qDesigner`.

To deploy a security filter XML file, you will need to modify `QB.properties` file under `<ERESInstallDir>/WEB-INF/classes` directory and add a new server option (add it to the `ServerCommands=` line) -`QuickDesignerSecuredParameters`: followed by the relative or absolute path to your XML file. For more information about setting server options, see Section 1.4.1.3.1 - Secured Parameter. For this tutorial, please add `-QuickDesignerSecuredParameters:qDesigner/SampleQDSecuredParameters.xml` to `QB.properties`.

With the XML file setup, any time a user creates or runs a data view or data view query that uses the field/table defined in the XML file, the filter defined for their security level will be applied. If the user then creates and saves a report/chart in QuickDesigner Reports/QuickDesigner Charts, this filter will be converted to a secured query parameter in the finished report/chart. Any changes to the security permissions after this time can be made by administrator through the Report Designer/Chart Designer. For more information about report-level security, see Section 4.1.10 - Template Security. If the user does not belong to a security level, or their security level is not defined in the XML file, the user will not be allowed to run the query. Therefore, it is important to have security levels assigned to all groups or users that are allowed to use QuickDesigner Reports/QuickDesigner Charts.

Sign in as **admin** and create 6 users, namely, **exec, easternmanager, southernmanager, midwesternmanager, westernmanager, eastsouthmanager**. Assign them with **Design** privilege. Then open the data registry and highlight `sample.xml`. Click the *Privilege* button to assign privilege to allow read and write access to all these users.

Create 6 security levels in organizer (**Note: security levels are case-sensitive**):

```
Executive (assigned to the user exec)
East (assigned to easternmanager)
South (assigned to southernmanager)
Midwest (assigned to midwesternmanager)
West (assigned to westernmanager)
EastSouth (assigned to eastSouthmanager)
```

If you are running this example with HSQL, open `SampleQDSecuredParameters.xml` and change `<ERESInstallDir>` in the line

```
<URL>jdbc:hsqldb:<ERESInstallDir>\help\examples\DataSources\database
\woodview</URL>
```

to your install directory. For example, if you installed ERES under `c:\ERES`, the line between the `<URL>` tags would read `jdbc:hsqldb:c:\ERES\help\examples\DataSources\database\woodview`. Make sure that under *Manage Data Sources* in the Organizer, the URL for this database reads the same as the above.

To prepare a data view for our example, open the Organizer, click the *Manage Data Sources* button on the toolbar, select `Sample.xml` data registry and click the *Edit* button. Select *Databases/Woodview/Data Views* node and click the *ADD* button. Choose *CUSTOMERS* table in the *Choose Tables* dialog, click the *ADD* button to add it to

the *Selected Tables* pane and then click *OK* to close this dialog. You are in the *Create Data View* dialog now. Enter **Customers** name for the view, select the *CUSTOMERS* table on the left, click *ADD* to add it to the *View Fields* pane and click *OK* to close the dialog. The data view is prepared. Close the *Data Source Manager* and the Organizer.

To create your own report, shut down and restart Tomcat, then sign in as one of the Manager logins you just created (not the executive level). Open Quick Designer Reports and click the *New* button on the toolbar to open the *Data Source Dialog*. Select Databases/Woodview/DataView/Customers data view. In the *DataView Builder*, select columns *REGION*, *COMPANY*, *ADDRESS*, *CITY*, *STATE*, *ZIP* and click the *Next* button. Do not add any conditions in the *Conditions* dialog and click *OK*. Finally, add all columns to the report. Save your report to make sure that all your users have access to the file and the database (see Section 2.3.2 - Setting User Privileges).

Log out and login again as **easternmanager**:

From Published Files, open the file you have just created.

You should see the report with all Eastern values.

Log out and login again as **eastsouthmanager**:

Open the file you have just created from Published Files.

You should see the report with both values for East and South.

Log out and login again as **exec**:

Click on the report you have created from *Published Files*.

You will only see one value, but if you click on the *filter*, you can change between the different security levels by typing in the region you want to see.

When viewing this report from QuickDesigner Reports, you will be automatically prompted for the value you want to see and prompted again every time you refresh, as you would be with any other parameter.

If you were to create a report using this query in QuickDesigner Reports while logged in as a user with *GrantAll* privileges (the Executive level in the example), the report would be saved by default without any secured parameters and all users would be able to view all parameter values from Published Files, QuickDesigner Reports and Report Designer. Any security of a report created from a user with *GrantAll* privileges must be done from Report Designer with template security (see Section 4.1.10 - Template Security).

If you want to create your own XML file, please follow the DTD spec in `<ERESInstallDir>/qDesigner/QuickDesignerSecuredParameters.dtd`.



### Tip

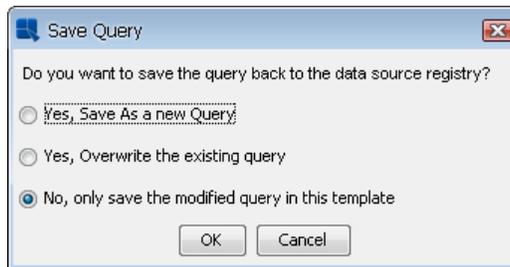
In case you think your query is not being filtered properly, you can add the flag `-debug:DVW_SEC_PARAM` to the `ServerCommands=` line of the `QB.properties` file; shut down the ERES Server, shut down your Application server (Tomcat), and run again. The Applications server's console or log file should show diagnostic statements for you to see possible problems.

### 3.1.3.4. Editing Queries

If you have selected to build a report using database data either by designing a query in the Query Builder, writing an SQL statement, or running a data view, you can modify the query directly from the Report Designer or the Chart Designer without having to go back to the Data Source Manager.

To modify a report or chart's query, select *Modify Query* from the Data menu in Report Designer or Chart Designer. If you have designed a query in the Query Builder, then the Query Builder interface will re-open allowing you to modify the query. If you have entered an SQL statement, a text box will open allowing you to modify the SQL. If you have used a data view, the data view conditions window will re-open allowing you to change the filters or pick additional fields.

Once you specify all the changes, you will be given an option to modify the query in the data registry, save a new query in the data registry, or modify only the query in the template.

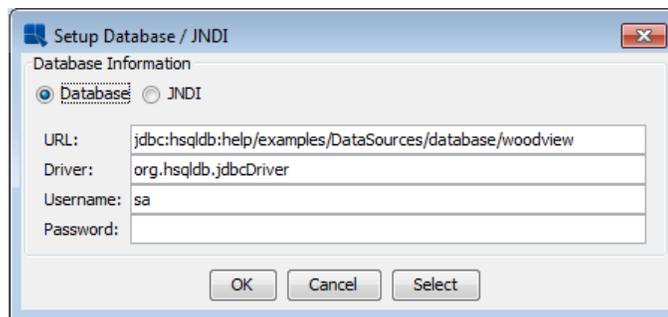


*Saving Query Options*

Once you specify the save options, the modified query will be applied to the report or chart. Note that if you made significant changes to the query, you may need to perform the data mapping again.

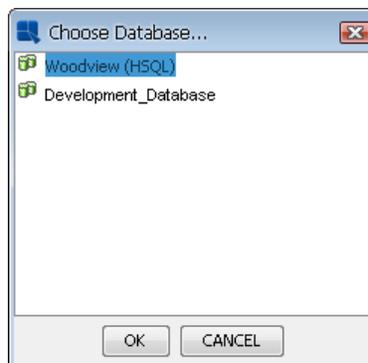
### 3.1.3.5. Editing Database Connections

If you have selected to build a report or chart using database data, you can also directly edit a template's database connection information from within the Report Designer or the Chart Designer. To modify the connection information in the report or chart, select *Modify Database* from the Data menu. This will bring up a dialog allowing you to specify a different URL, Driver, Username, and/or password for the report to use when connecting to the database.



*Change Database Connection Dialog*

In addition to manually entering the database information, you can retrieve the database connection information from a data registry. To do this, click the *Select* button on the Database Connection dialog. This will allow you to browse to XML registry file from which you want to pull the database connection. When you select a registry file, you will be presented with a list of databases defined in the registry.



*Select Database from Registry*

Select the database that you want to use and click the *Ok* button. The connection information for that database will be automatically applied to the connection dialog. After you set the connection information for the template, click the *Ok* button to apply the changes. A dialog will open asking you if you would like to verify the new connection information.



*Verify Connection Dialog*

Unless you know that data source isn't present, it's generally a good idea to check that the supplied connection information is correct. If you're changing the database connection information in a report, all the sub-reports, charts, and drill-down layers in the report that use the same connection information will be automatically updated as well.

Unlike the modify query feature, changes to a template's database connection will only be saved to the template. It will not be saved to the data registry.

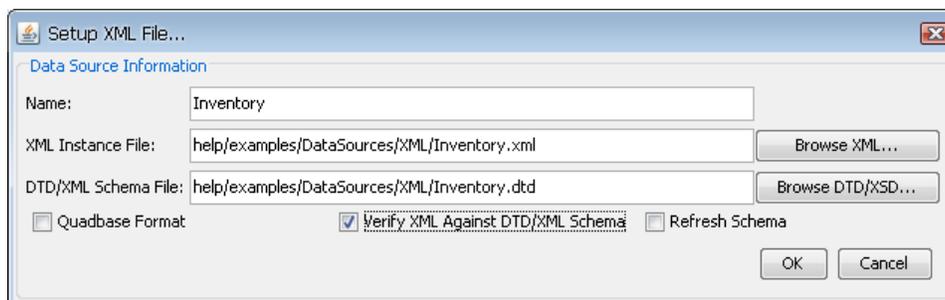
### 3.1.3.6. Troubleshooting Database Connections

If you're having difficulty connecting to your database via JDBC, ERES provides a small utility you can use to help troubleshoot the problem. To launch the utility, point your Web browser to `http://machinename:port/ERES/TestConnection.jsp`. This page opens a small form that allows you to enter database connection information as well as sample query.

The utility will directly contact the database (outside of any ERES components) and execute the query, if provided. Any connection/query errors and suggested fixes will be printed on the page.

### 3.1.4. Data from XML and XBRL Files

In addition to relational databases, ERES allows you to retrieve data and query XML files. XML data can be in virtually any format, but you need to specify a DTD file or an XML Schema (XSD) along with the XML data. To set up an XML data source, select *XMLFiles* node in the Data Source Manager and click the *Add* button. A dialog will open prompting you to specify options for the new XML source.



*Setup XML Data Source Dialog*

The first option allows you to specify a display name for the XML data source. The second option allows you to specify the location of the XML file from which you want to retrieve data. Note that you can also specify an XBRL file in this field. You can set up a data source that retrieves XML data from an HTTP server here as well, by adding the appropriate URL as the file location. The third option allows you to specify the location of a valid DTD or XML schema file for the XML file.

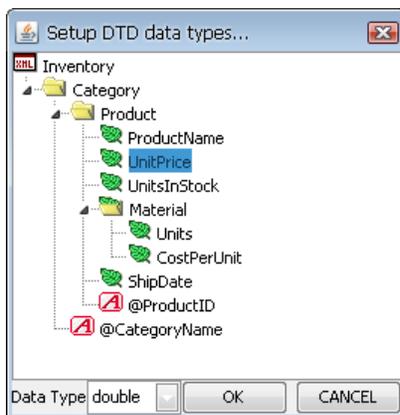
The *Quadbase Format* checkbox allows you to indicate whether the XML file is in the form of an XML export from ERES. For example, if you choose to export a report's data in XML format, you can read it back in using this format. When you use a file in this format, you do not have to specify a DTD or XML Schema.

The *Verify XML against DTD/XML Schema* checkbox will make sure that the supplied XML file/source complies with the layout specified in the DTD or XML schema file. Because queries are designed based on the structure of the DTD/XML Schema file, a non-conforming XML source could produce unexpected results. If the XML does not conform to the DTD or XML schema, you will be given a warning. You can, however, continue setting up the data source.

The *Refresh Schema* checkbox will only appear if you choose to edit an existing XML data source. Checking this option will reload the schema or DTD definition to incorporate any changes to the structure in the XML data source

in the registry. This option is only necessary if the DTD or schema definition has changed since the data source was first created.

Once you finish setting up the XML file and the DTD/XML Schema, a new dialog will open allowing you to specify the data type for all the selectable elements in the XML data source. The dialog will be different depending on whether you are using a DTD file or XML Schema.



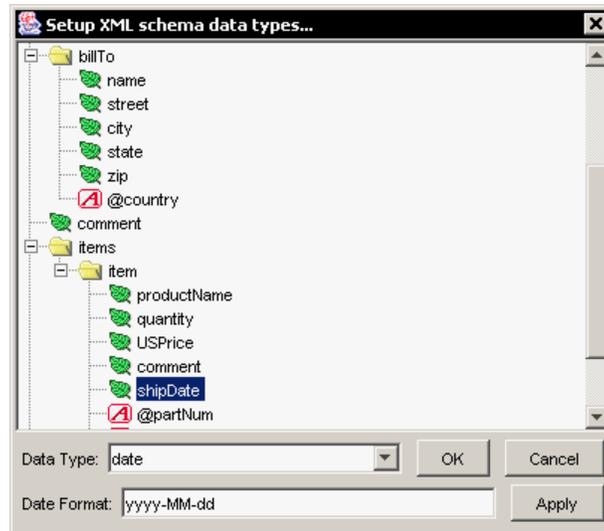
*DTD Data Type Selection Dialog*

Because DTD files do not specify a correct data type for elements, all elements are considered to be Strings by default. To change the data type of an element, you have to select the element and pick a data type from the drop-down window at the bottom of the dialog. To ensure proper results when you query the XML file, you should set the data type for all selectable elements. This includes leaf nodes, parent nodes that contain data, and attribute elements. The following data types are supported:

- String
- Integer
- Double
- Date (If you specify date as the data type, you will also be required to specify the date format.)
- Boolean

Once you finish specifying the data types, click the *OK* button and the XML source will be added to the Data Source Manager.

If you're using an XML schema, a different data types dialog will open.

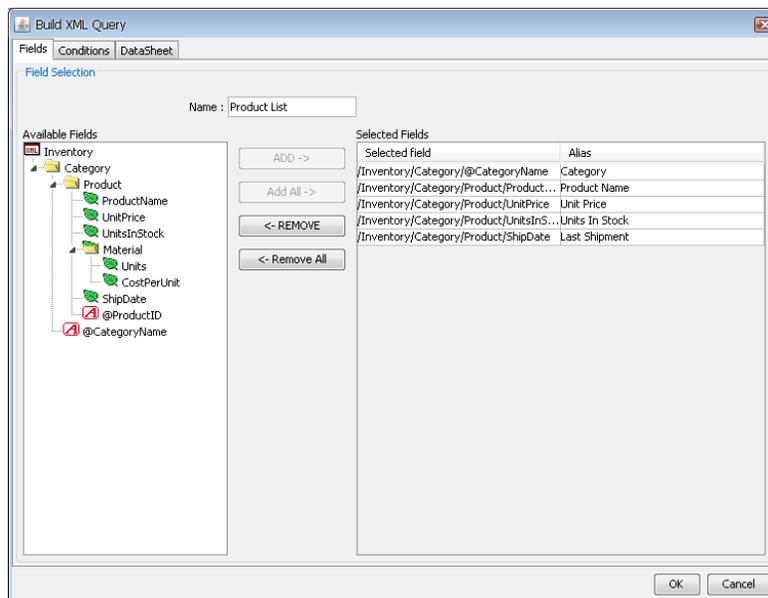


*XML Schema Data Type Selection Dialog*

Generally, the data types should already be defined in the XML schema file, but you can make any changes in this dialog. Once you finish specifying the data types, click the *OK* button and the XML source will be added to the Data Source Manager.

### 3.1.4.1. XMLQueries

Once you set up an XML data source, you can then create queries to select nodes, specify filtering conditions, and transform the tree structure into the tabular form used by ERES. To add a query, select the node for your XML source and click the *Add* button. This will launch the XML query builder interface that allows you to construct a query.



*XML Query Field Selection Tab*

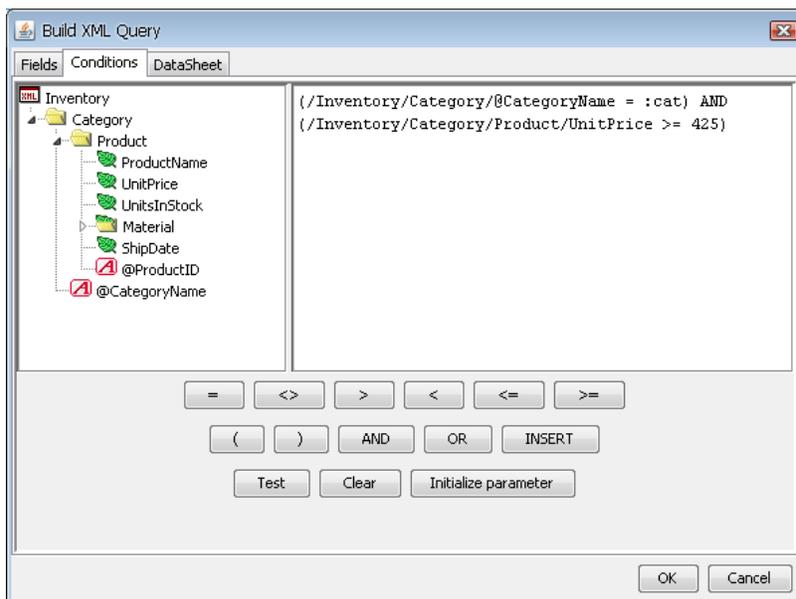
The first tab in the XML query builder allows you to select the fields/nodes from the XML file that you would like to use in the report. The left side of the window contains the tree structure from the DTD or XML schema file. You can pick any selectable elements and add them to the query by clicking the *Add* button. Selected fields will appear in the right side of the window. You can specify an alias for any field by double clicking the *Alias* field for a column and typing the new column alias.



## Note

Each selected element will become a column in the report or chart data. For results where a one-to-one relationship cannot be determined, the tabular structure is built using all available permutations in the data (similar to a cross-join in SQL). For best results, it is recommended that you select fields for a query where a clear hierarchical relationship is present.

The *Conditions* tab of the XML query builder allows you to specify some basic filtering criteria for your selection.



*XML Query Conditions Tab*

You can specify an equal, not equal, greater than, less than, less or equal to, or greater or equal to condition for any selectable element in the XML file. You can also use the AND and OR operators to build compound conditions. Fields are specified using a direct path down the XML tree. Currently, only direct path is supported. You cannot use more complex XPath expressions. To add a field, you can double click on it in the left side, or you can select it and click the *Insert* button.

After you finish writing the conditions, click the *Test* button to verify that the syntax is correct.

The *DataSheet* tab allows you to preview the query result and see how the XML data is converted to tabular form. You can navigate through the result set the same way as in Query Builder (Section 3.1.3.2.1.5 - Adding Extra SQL).

Once you select the fields and specify the appropriate conditions, click the *OK* button. The query will then be added as a new node under your XML source in the Data Source Manager and can now be used to create a report or chart.

There is a sample XML file with DTD and XML schema descriptions included in the ERES installation. The files are located in `help/examples/DataSources/XML` directory of your installation and are called `Inventory.xml`, `Inventory.dtd` and `Inventory_XSD.xml`, `Inventory.xsd` for the XML schema example. There is also a sample servlet that allows you to stream XML data to the Chart Designer. The servlet code and instructions are located in `help/examples/DataSources/XML/servlet` directory.

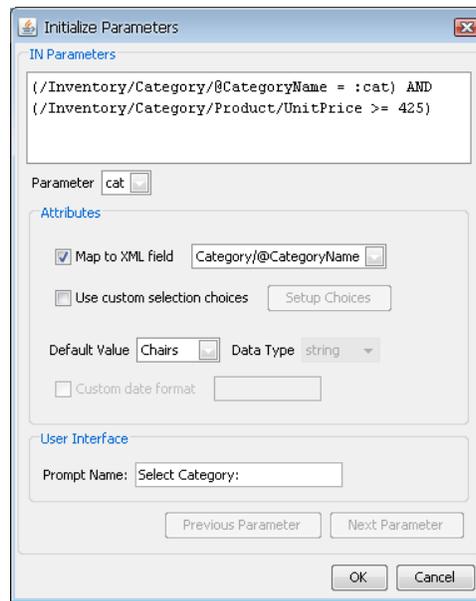
### 3.1.4.1.1. XML Parameters

As with database queries, you can also specify parameters for XML queries. The same syntax “:” is used to denote a parameter in the XML condition as it is in a query condition. So the following XML condition:

```
/Inventory/Category/@CategoryName = :category
```

would place a dynamic filter on the query for the *CategoryName* attribute. XML parameters are initialized in the same way as query parameters. The initialization dialog will appear if you try to preview or close the query, or

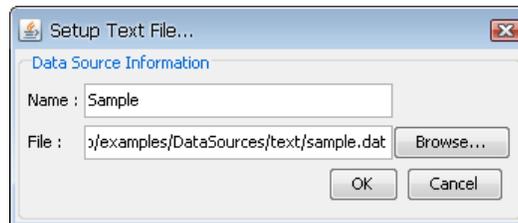
you can trigger it by clicking the *Initialize Parameters* button. The only difference is that instead of mapping to a database field, the parameter prompt can be mapped to a node in the XML file.



*XML Parameter Initialization Dialog*

### 3.1.5. Data from Text Files

ERES also allows you to retrieve data from flat text files. To add a text file as a data source, select the *TXTFiles* node in the Data Source Manager and click the *Add* button. A dialog will open prompting you to specify a display name and the location of the text file you want to use.



*Add Text Data Source Dialog*



#### Tip

The text file can be also retrieved from URL. To do so, enter the URL to the *File* text field. You have to enter a full URL with protocol etc. (e.g. `http://www.quadbase.com/textfile.txt`).

After you specify the information, click the *OK* button and the text source will appear under the *TXTFiles* node of the Data Source Manager window.

#### 3.1.5.1. Formatting Requirements for Text Files

There are certain formatting requirements for the data within a text file in order make it readable by ERES. Generally, data is expected to be in a form similar to the following:

```
String,date,decimal
Name,day,Volume
"John", "1997-10-3", 32.3
"John", "1997-4-3", 20.2
"Mary", "1997-9-3", 10.2
```

"Mary" , "1997-10-04" , 18.6

The above data file is a plain text file. The first row specifies the data types and the second row specifies the field names. The third row and so on are the records. Every text file must consist of these three parts. There are four records, with three fields each in the example data file. The delimiter between the fields may be one of the following characters: " , " , " ; " , or " " (that is comma, semi-colon, or space). Each field can be put in quotes (single or double).

### 3.1.5.2. Data Types and Format for Text Files

In text data files, the data type is specified using a keyword. The following is a list of recognized keywords and their corresponding JDBC type and Java type.

Data File Keywords (Not Case Sensitive)	JDBC Type	Java Type in ERES
Boolean, logical, bit	BIT	Boolean
tinyint	TINYINT	byte
smallint, short	SMALLINT	short
int, integer	INTEGER	int
long, bigint	BIGINT	long
float	FLOAT	double
real	REAL	float
double	DOUBLE	double
numeric	NUMERIC	java.math.BigDecimal
decimal	DECIMAL	java.math.BigDecimal
date	DATE	java.sql.Date
time	TIME	java.sql.Time
timestamp	TIMESTAMP	java.sql.Timestamp
string	CHAR	String
varchar	VARCHAR	String
longvarchar	LONGVARCHAR	String

For certain data types, the data in a text file must be presented in a specific format. The following is a list of the data types that require specific formatting:

Data Type	Format	Example
Date	yyyy-mm-dd or yyyy-mm	2001-06-12 or 2000-06
Time	hh:mm:ss	12:17:34
Timestamp	yyyy-mm-dd hh:mm:ss	2001-06-12 12:17:34
Boolean	true/false, t/f, 1/0 (case insensitive)	true

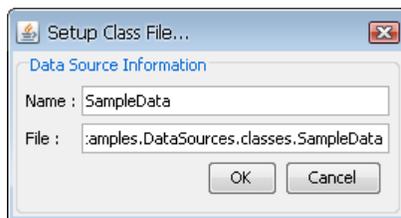
There is a sample text file included in the ERES installation. The file is located in `help/examples/Data-Sources/text` directory of your installation and is called `Sample.dat`.

### 3.1.6. Data from Class Files

For maximum flexibility, ERES allows you to design reports and charts using object or array data by providing an interface to pass data to the Report Designer as an argument. Using the API, you can implement `IDataSource` to return an `IResultSet` object similar to the `java.sql.ResultSet` interface used for JDBC result sets. Users can provide their own implementation of `IResultSet` or use one provided by ERES.

For more information about this feature, please see Appendix 8.A.5 - Data passed in a Custom Implementation.

To add a class file as a data source, select the *ClassFiles* node in the Data Source Manager and click the *Add* button. A dialog will open prompting you to specify a display name and the location of the class file you want to use.



*Add Class File Dialog*

There is sample code available in `help/examples/DataSources/classes` directory of the installation. The compiled code will generate a class file that passes the data array into Report Designer or Chart Designer. Note that in order to use a class file as a data source, you must have the file or directory containing the package in the classpath of your application server/servlet runner.

**Here is a simple guide to use `SampleData.class` file available as datasource in ERES Data Registry:**

The most simple solution is to copy `SampleData.class` file from `<ERES install dir>/help/examples/DataSources/classes` into `<ERES install dir>/WEB-INF/classes/help/examples/DataSources/classes` directory (you will have to create `help/examples/DataSources/classes` under `<ERES install dir>/WEB-INF/classes/` before copying the file).



**Note**

You can also change the app server CLASSPATH to include `<ERES install dir>` instead. (Note that in Tomcat, you have to include all the jars, as setting the CLASSPATH in Tomcat will not pick up the jars under `ERES/WEB-INF/lib`.)

Start or restart your Tomcat server and open ERES Organizer - DataRegistry

Select *ClassFiles* and add the new classfile by filling in the dialog (as showed in the picture above):

**Name:** SampleData

**File:** help.examples.DataSources.classes.SampleData

Now you can view your ClassFile data and use it as datasource for reports and charts.

### 3.1.6.1. Parameterized Class Files

ERES provides an additional API interface, `IParameterizedDataSource`, that allows you to define report/chart parameters within the context of a class file data source.

Parameters that are defined within the context of a class file work in the same way as query parameters. For more information about setting up a parameterized class file data source, please see Appendix 8.A.5 - Data passed in a Custom Implementation.

### 3.1.7. Data from EJBs

Using Enterprise JavaBeans™ technology, developers can simplify the development of large enterprise applications. With EJB technologies, developers can rely on building business logic and allow the application server (EJB container) to manage all system level services.

When working in the Java EE™ environment, persistent data interfaces are provided by entity beans. Although the underlying storage mechanism might be a relational database, the application data model is the EJB and it may not be desirable to have a reporting tool making redundant database connections. For this situation, ERES allows users to query data directly from an entity bean.

To add an EJB as a data source, the EJB must first be deployed in the application server and the client JAR file containing the appropriate stub classes must be added to your application server/servlet runner's classpath. Select the *EJBs* node in the Data Source Manager and click the *Add* button. This will bring up a dialog allowing you to specify a display name and the connection information for the bean.

*Add EJB Dialog*

To connect to an EJB data source, you must provide the JNDI lookup name for the bean (this is specified when you deploy the bean). For EJB 1.1 users, specify the name for the home interface and the remote interface. For EJB 2.0 users, specify the local home interface and the local interface. Once you specify all the information, click the *Import* button, which will analyze the home interface and retrieve all of the finder methods. Any methods found will be populated in the *List of Methods* section in the dialog.

The same dialog can be used to filter the data being retrieved based on parameters that are present in the finder methods. When you select a method in the left dialog, any parameters present will appear in the *Parameter List* section. You can then click on a parameter to set its value.

*Specifying EJB Parameter Values Dialog*

When you select a parameter, the data type of the parameter will appear in the lower portion of the window. Below that you will be able to specify a value for the parameter. Be sure to enter a correct value for the data type, and then click the *Set Value* button. This will fix the parameter values. Once you finish setting up all parameter values, click the *Environment* button. This will bring up a new dialog allowing you to specify environment properties for your application server. This information is necessary for the ERES Server to connect to the EJB.

Please enter environment information:	
APPLET	
AUTHORITATIVE	
BATCHSIZE	
DNS_URL	
INITIAL_CONTEXT_FACTORY	weblogic.jndi.WLInitialContextFactory
LANGUAGE	
OBJECT_FACTORIES	
PROVIDER_URL	t3://192.168.0.4:7001
REFERRAL	
SECURITY_AUTHENTICATION	
SECURITY_CREDENTIALS	
SECURITY_PRINCIPAL	
SECURITY_PROTOCOL	
STATE_FACTORIES	
URL_PKG_PREFIXES	

*EJB Environment Setup*

The fields here are the available environment properties for the JNDI context interface. You don't have to specify all values, only the information necessary for your environment (application server). Once you finish specifying the environment variables, click the *OK* button. You will be returned to the EJB setup window. Click *OK* again to finish setting up the EJB data source. A new node will appear under *EJBs* with your EJB.

You can modify the parameter values by selecting your EJB source, and clicking the *Edit* button.

There is a sample EJB data source included in the installation in `help/examples/DataSources/EJB` directory. The directory contains `sales.ear` and `salesClient.jar` files, as well as the source code for the Sales entity bean. The `sales.ear` file is designed to be deployed in Java 2 Reference Implementation and uses the Cloudscape database as the underlying storage mechanism. You can use `SalesClient.java` program to populate the Cloudscape database. The `salesClient.jar` file contains the stub classes to connect to the deployed Sales EJB and needs to be in the classpath.

### 3.1.8. Data from SOAP with WSDL support

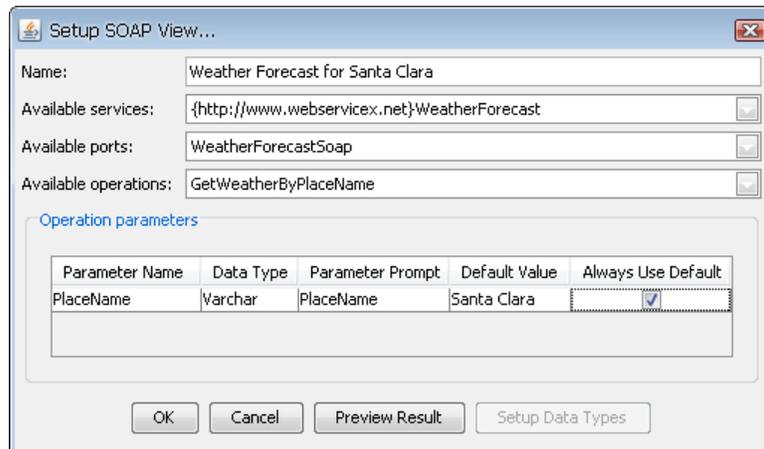
ERES also allows you to retrieve data using SOAP (Service Oriented Architecture Protocol). To connect to a SOAP data source using WSDL, you don't need to know any URLs for the services, SOAP actions, operation names or parameters. All you need to know is a location of WSDL file, which contains all the necessary information.

To set up a SOAP data source, select the *SOAPServices* node in the Data Source Manager and click the *Add* button. A dialog will open prompting you to specify options for the new SOAP data source.

*SOAP data source setup*

The first option allows you to specify a display name for the data source. The second option allows you to specify a location of the WSDL file. The location can be either absolute path on the server or path relative to your ERES installation directory or URL. Once you specify the connection information, you can test the connection to the WSDL file by clicking the *Test* button. This will test retrieving the WSDL file from the URI you've provided, check the file for any supported SOAP operations and report any problems. After clicking *OK*, a new SOAP data source node will be added to the data registry.

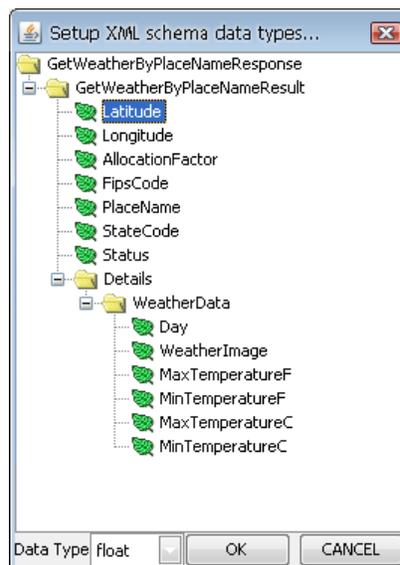
To add a new SOAP View, select an existing SOAP data source and click the *Add* button. A dialog will open prompting you for all the parameters necessary to make a SOAP query.



*Setup SOAP View dialog*

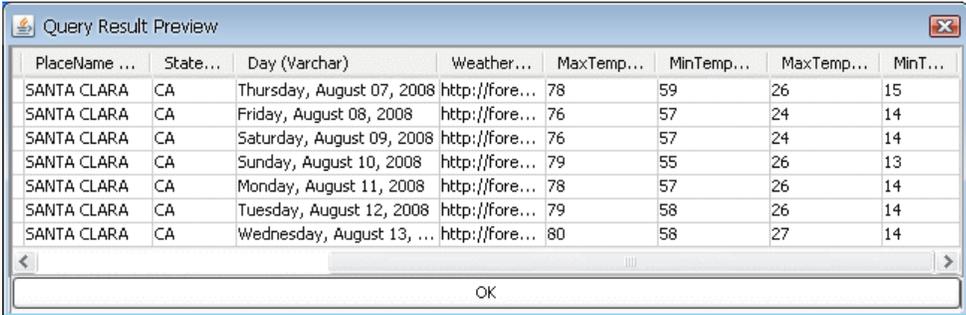
The first option in the dialog allows you to specify a display name in the Data Source Manager. Next, there are three drop-down menus in the dialog. The first drop-down menu contains all the SOAP services described in the WSDL file. Once you specify a service, the second drop-down list will be populated with all the ports of this service. Selecting a port will populate the last drop-down list with all the operations of this port. If you move the mouse over any drop-down list, a hint will appear with documentation for the service/port/operation (if the documentation is provided in the WSDL file). After specifying the service, port and operation, all the parameters of the operation will be read. If there are some parameters, they will be displayed in a table. The first two columns of the table are not editable. They are read from the WSDL file. The next 2 columns are editable and allow you to specify parameter prompts and default values. The last column contains a checkbox which allows you to choose whether the default parameter value will always be used or not. This means that this parameter value will be fixed and you will not be prompted for it. All the parameters that don't have this checkbox checked will be used as report/chart parameters.

The *Setup Data Types* button is only available when editing the SOAP View from the Data Source Manager and allows you to adjust data types when necessary. In order to verify result from the SOAP response, click the *Preview Result* button. All the default values will then be tested to see if they have proper data type or not. In case they do not match, you will be prompted to adjust them. After that, you will get the setup data types dialog (the same as for XML data source). If the data source is parameterized, you will get the parameter prompt dialog before specifying data types. Please note that in order to generate the XML schema properly, you have to specify existing parameter values.



*Setup XML Schema Data Types dialog*

You can setup data types from this dialog. The behavior is exactly the same as for XML data source with DTD schema (see Section 3.1.4 - Data from XML and XBRL Files). Once you finish specifying data types, click the *OK* button. A dialog will open showing you the result preview.



PlaceName ...	State...	Day (Varchar)	Weather...	MaxTemp...	MinTemp...	MaxTemp...	MinT...
SANTA CLARA	CA	Thursday, August 07, 2008	http://fore...	78	59	26	15
SANTA CLARA	CA	Friday, August 08, 2008	http://fore...	76	57	24	14
SANTA CLARA	CA	Saturday, August 09, 2008	http://fore...	76	57	24	14
SANTA CLARA	CA	Sunday, August 10, 2008	http://fore...	79	55	26	13
SANTA CLARA	CA	Monday, August 11, 2008	http://fore...	78	57	26	14
SANTA CLARA	CA	Tuesday, August 12, 2008	http://fore...	79	58	26	14
SANTA CLARA	CA	Wednesday, August 13, ...	http://fore...	80	58	27	14

*Query Result Preview dialog*

Clicking the *OK* button in this dialog will take you back to the Setup SOAP View dialog. Once you finish specifying all necessary information, click the *OK* button in the dialog. A new node will be added under your SOAP data source in the Data Source Manager and can now be used to create a report or chart.

### 3.1.9. Data from Salesforce

The Salesforce data source is designed for existing Salesforce users who want to display their Salesforce data in ERES. The connection to the Salesforce server is established via SOAP using Salesforce Partner WSDL (version 13.0). Users communicate with Salesforce server by SOQL (Salesforce Object Query Language) queries. Please note that users must have valid Salesforce accounts with username and password to work with this data source. Moreover, users who use the ERES Salesforce data source must have access to Salesforce account from trusted networks. To add your IP address to the trusted IP list, you have to activate your computer as described below.

For more information about SOQL queries and activating Salesforce user's accounts from trusted networks, please visit the following Salesforce sites:

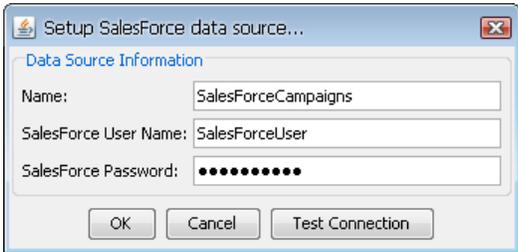
SOQL queries

[https://developer.salesforce.com/docs/atlas.en-us.soql\\_sosl.meta/soql\\_sosl/sforce\\_api\\_calls\\_soql\\_sosl\\_intro.htm](https://developer.salesforce.com/docs/atlas.en-us.soql_sosl.meta/soql_sosl/sforce_api_calls_soql_sosl_intro.htm)

Activating Salesforce user's accounts

[https://help.salesforce.com/s/articleView?id=sf.security\\_networkaccess.htm](https://help.salesforce.com/s/articleView?id=sf.security_networkaccess.htm)

To set up a Salesforce data source, select the *SalesForce* node in the Data Source Manager and click the *Add* button. A dialog will open prompting you to specify a display name for the data source, user name and password to your Salesforce account. Once you specify the connection information, you can test the connection to your Salesforce account by clicking the *Test Connection* button. This will test the connection using the information you've provided and report any problems.



Setup Salesforce data source...

Data Source Information

Name: SalesforceCampaigns

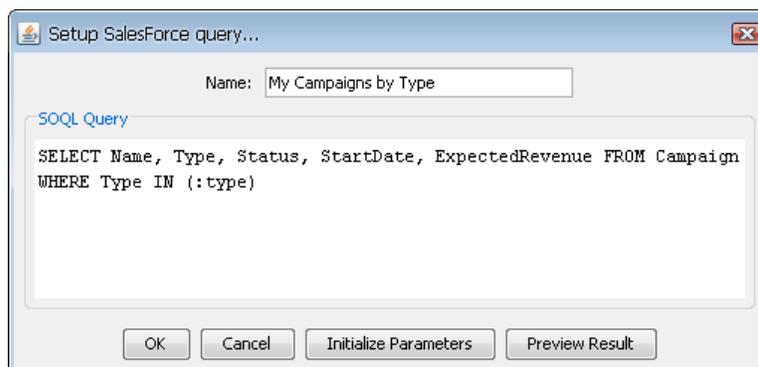
SalesForce User Name: SalesforceUser

SalesForce Password: ●●●●●●●●

OK Cancel Test Connection

*Setup Salesforce Data Source Dialog*

Once you add a Salesforce data source, a new node will appear in the Data Source Manager window. To add a new Salesforce query, click the *Add* button. A new dialog will open prompting you to specify a query name and SOQL query.



*Setup Salesforce Query Dialog*

Please note that only child-to-parent relationship queries are supported in the current ERES version. You cannot use parent-to-child queries (using nested SOQL queries). For more information about Salesforce relationship queries and their syntaxes, please visit the following Salesforce site: [https://developer.salesforce.com/docs/atlas.en-us.soql\\_sosl.meta/soql\\_sosl/sforce\\_api\\_calls\\_soql\\_relationships.htm](https://developer.salesforce.com/docs/atlas.en-us.soql_sosl.meta/soql_sosl/sforce_api_calls_soql_relationships.htm)

Moreover, this dialog allows you to initialize query parameters in case that your query contains single value or multi value parameters. A parameter is specified within an SOQL statement using the ":" character. Generally the parameter is placed in WHERE clause of an SOQL Select statement. For example, the following SOQL statement

```
Select Name, Type, Status, ActualCost From Campaign Where Name
= :CampaignName
```

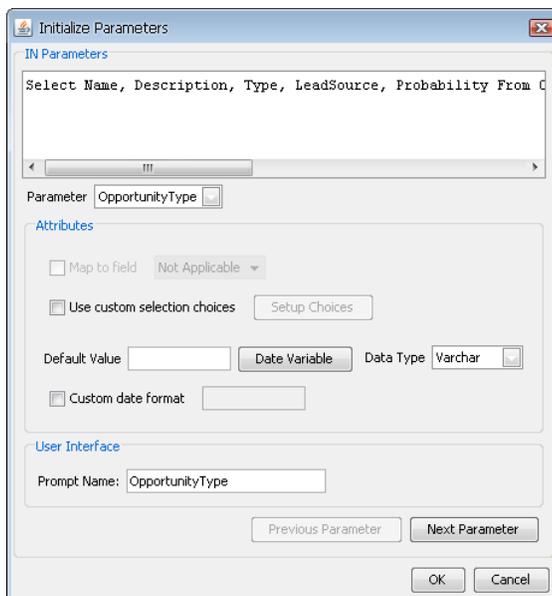
specifies a single value parameter called *CampaignName*. You would then be able to enter a campaign name at run-time and only retrieve data for that campaign.

Another example of SOQL statement shows using of multi value parameters that take an array of values as the input rather than single values.

```
Select Name, Description, Type, LeadSource, Probability From Opportunity
Where Type IN (:OpportunityType) And LeadSource IN (:OppLeadSource)
```

The statement specifies two multi value parameters called *OpportunityType* and *OppLeadSource*. You would then be able to specify opportunity types and lead sources at run-time and you will only retrieve data according to specified parameters values.

In order to initialize SOQL query parameters, click the *Initialize Parameters* button. The initialize parameters dialog will then appear allowing you to specify parameters mapping.



*Initialize Parameters Dialog*

From this dialog, you can specify the following options:

**Map to field:**

This allows you to specify a field from the Salesforce data source whose values will be used for the parameter input. Selecting this option modifies the parameter prompt that you will get when previewing or running the report/chart. If you map the parameter to a Salesforce field, you will be prompted with a drop-down list of distinct values from which you can select a parameter value. If you do not map, you will have to type in specific parameter value.

**Use custom selection choices:**

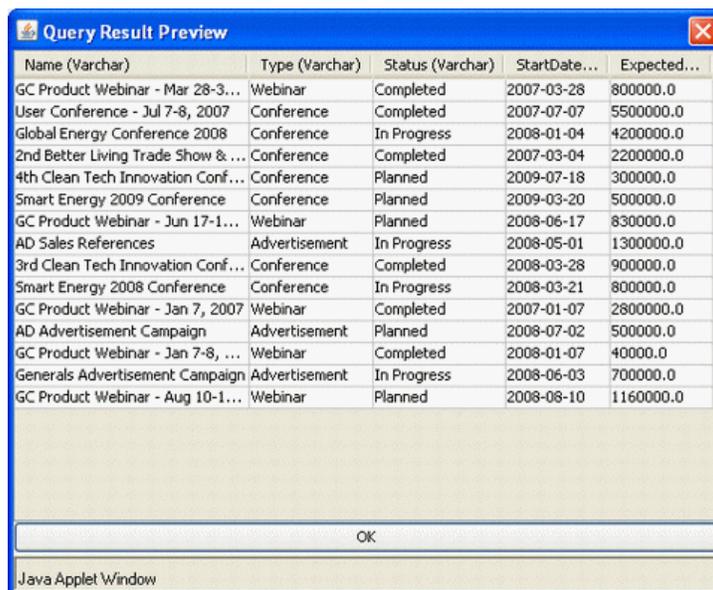
Rather than having a drop-down menu with all the distinct column values, you can build a custom list of parameter values. To set up the list, select this option and click the *Setup Choices* button. This will launch a new dialog allowing you to create a list of choices.

The rest of the options are basically same as for database query parameters. For further information about initializing database query parameters, see Section 3.1.3.2.2.2 - Initializing Query Parameters. Once you specify mapping for all available parameters, click the *OK* button and you will be taken back to the Setup Salesforce Query dialog.

From the Setup Salesforce Query dialog, you can also preview the query result using the *Preview Result* button to verify output from your query. In case you have a parameterized query, the parameter prompt dialog will appear prompting you to specify parameter values. Once you specify the parameter values, click the *OK* button and the query result preview dialog will appear.



Parameter Prompt



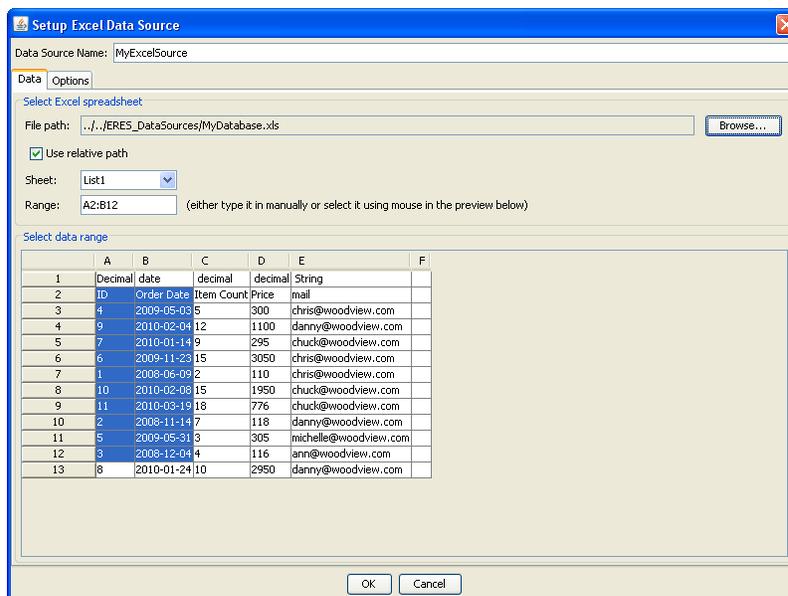
Query Result Preview Dialog

From this dialog, you can verify the query output. Clicking the *OK* button will take you back to the Setup Salesforce Query Dialog.

Once you specify the query, click the *OK* button. The query will then be added as a new node under your Salesforce data source in the Data Source Manager and can now be used to create a report or chart.

### 3.1.10. Data from Excel files

ERES also allows you to design reports and charts using data retrieved from Excel files. To add an Excel file as a data source, select the *ExcelFiles* node in the Data Source Manager and click the *Add* button. A dialog will open prompting you to specify the data source name and to select the Excel file from which the data should be imported.



Setup Excel Data Source dialog - Data

After selecting the Excel file, the imported data will be previewed in the dialog. If the checkbox *Use Relative Path* is checked, the file path to the selected Excel file will be set as relative to the ERES installation directory. Otherwise the full path will be used. In case the Excel file is stored on a different disk drive from the one which ERES is installed on, this option is not available. You can select the sheet (if there is more than one in the file) and the cells which are relevant for the data source being designed using your mouse or by specifying the range in the *Range* box (for instance, you can specify that your source will use the data from the columns A and B and from the rows 2 to 12 by typing **A2:B12** in the *Range* box; this is the format which is also used for ranges in MS Excel). You can also select the data by clicking the row header or the column header. Hold **Ctrl** or **Shift** to select more rows or columns. Click the top left corner to select all cells. Again, this behavior is similar to MS Excel.

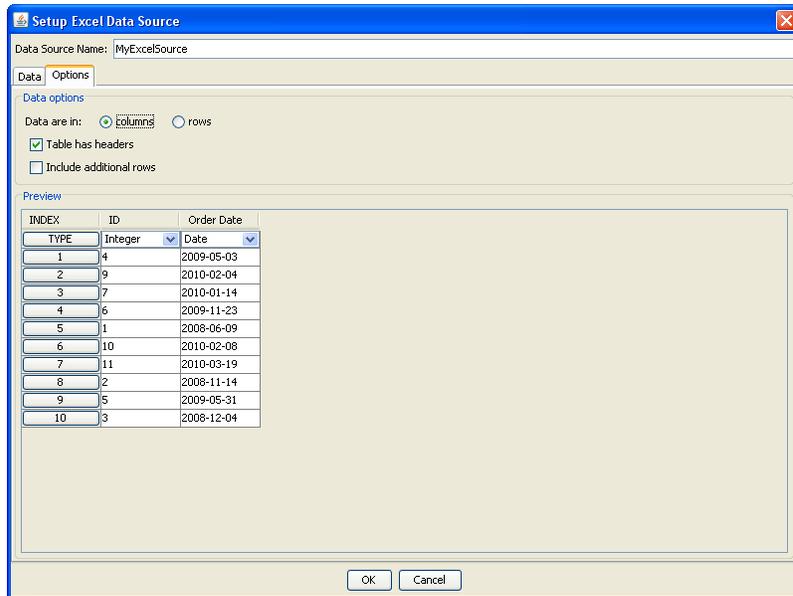
Please note that ERES can process both \*.xls files and \*.xlsx files. The \*.xlsx files are used in Microsoft Excel 2007 and newer and they are based on Open XML.

ERES can also process basic Excel formulas. If it is not able to process the entered formula, an error message will be displayed.

Empty rows (in case the data are in columns) or columns (in case the data are in rows) at the end of the sheet are automatically removed from the data source, even in case they have been selected.

Click on the *Options* tab to specify whether the data are in columns or in rows to get the data structure correctly. You can also specify whether table headers are included in your data selection. The option *Include additional rows* or *Include additional columns* (depending on whether the data are in columns or rows) allows you to automatically include data rows or columns into the Excel file after the data source has been created without the need to change the data source in the Data Source Manager manually.

The bottom part of the dialog on the *Options* tab shows you the data source content following the current configuration. You can change the data type for each column of the data source there, if desired. The data types are detected automatically, therefore changing the type should not be necessary in most cases.



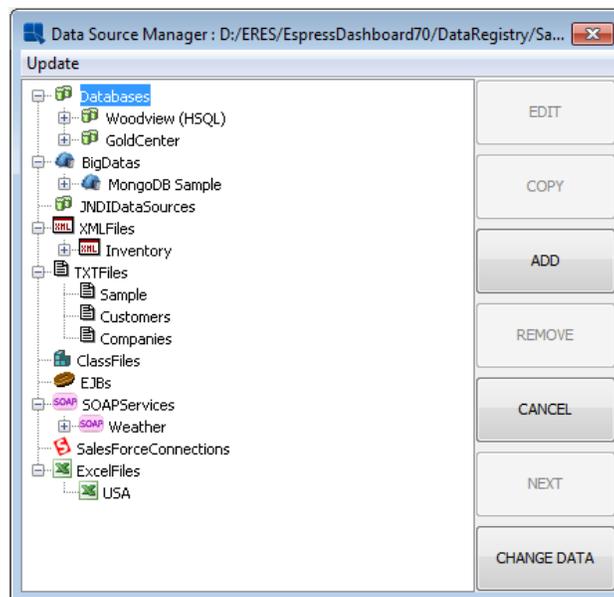
*Setup Excel Data Source dialog - Options*

Click *OK* to save the data source when the configuration is finished.

### 3.1.11. Using Data for Charts and Reports

Once you set up a data registry, the sources defined in the registry can be used for charts and reports. The first time you select to start a new report or chart, you will be prompted to select a registry you want to use. If there are currently no available registries (meaning that you haven't created any, or you don't have privileges to view any), you will be prompted to go the data registry manager to create one.

Once you select a registry, a data source manager window will open allowing you to select, add, or modify a data source that you want to use for the chart or report.



*Data Source Manager Window for Chart Designer*

After you first select a registry, the next time you design a report or chart, it will automatically connect to that registry and open the data source manager. You can change registries by clicking the *Change Data* button.

### 3.1.11.1. Using Multiple Data Sources

Once you select the data source you want to use for a report or chart and click the *Next* button, the next screen will display first twenty records from the data source (With the exception of data views, which will require you to select fields and set conditions first). You can display all of the records by checking the *Show All Records* box.

ERES allows you to construct reports and charts from multiple data sources. Once you have selected your first source, and click *Next* from the data table window, you will be presented with a dialog asking if you would like to process the current data or select another data source.

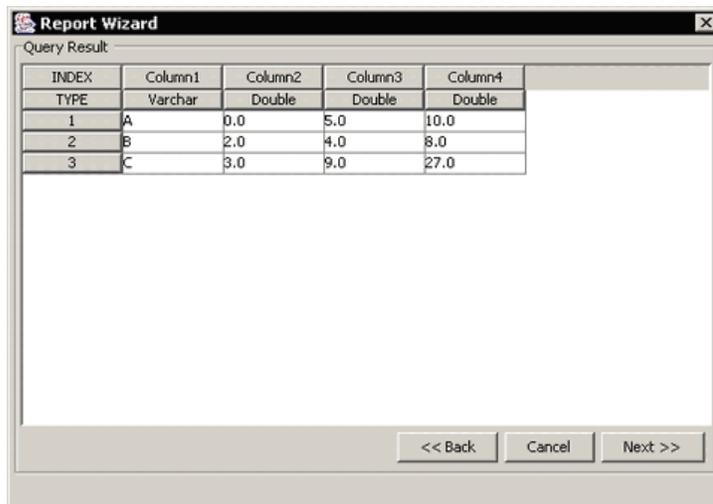


*Additional Data Source Dialog*

If you select *Process Data* and click the *Next* button, you will continue to the next step in the report or chart wizard. If you select *Get Other Data Source*, you will return to the Data Source Manager to select another data source for the report. You can repeat this process to select as many sources as you want.

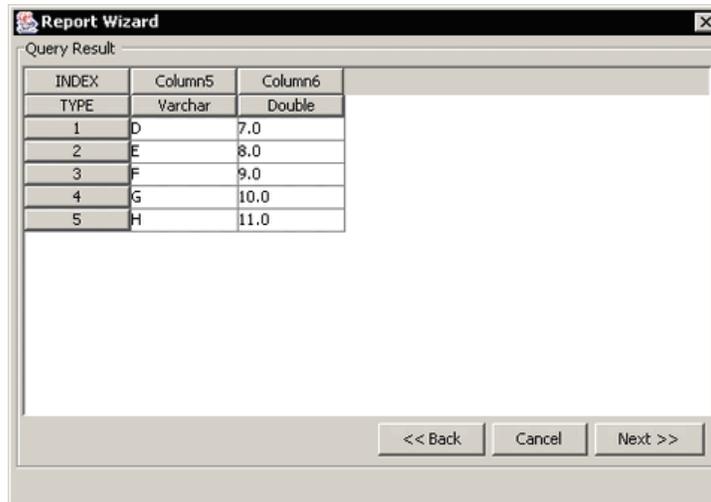
Multiple data sources are combined sideways in the data table. This means that the columns from the second (or third, fourth, etc.) data sources are placed to the right next to the columns from the first data source. If the columns from one data source have more rows than the other ones, null values will be placed in extra rows.

For example, assuming you want to use two data sources to create a report. The data table generated by the first source will look like this:



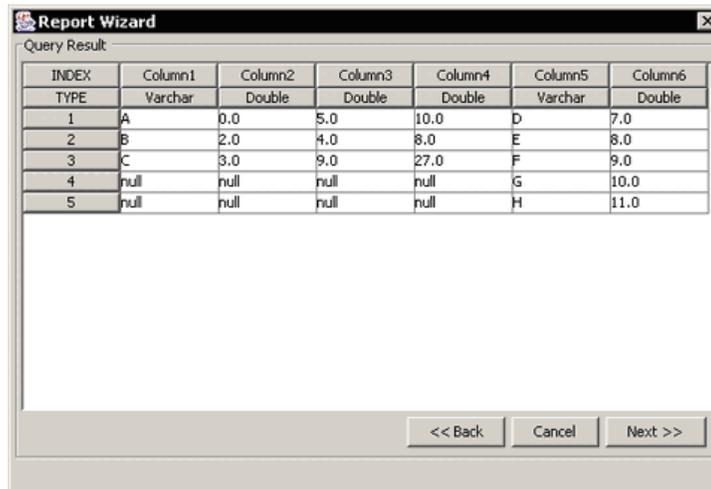
*Data From the First Data Source*

The data from the second source will look like this:



*Data From the Second Data Source*

When you combine two data sources, you will get the following data table:



*Data From Combined Sources*

As you can see, the two data sources have been placed side by side. Since the second source has more rows of data than the first, additional rows of null data were added.



### Note

You cannot use parameterized data sources to create a multiple data source.

You can also use multiple data sources in reports by adding sub-reports. For more information about sub-reports, please see Section 4.1.9 - Sub-Reports.

### 3.1.11.2. Change Data Source

At any point during report or chart design, you can select to change the template's data source. Since report and chart templates are saved with their data source information, you must use the option within Report Designer or Chart Designer to change the template's data source. Simply altering a data source in the registry will not affect the template unless you use the data source updating feature (For more information about this, see Section 3.1.11 - Using Data for Charts and Reports). To change a template's data source, select *Change Data Source* from the Data menu, or click the *Change Data Source* button on the toolbar. This will bring up the Data Source Manager allowing you to select a new data source or modify an existing one.

If the new data source is significantly different from the previous data source in number of columns or data types, you might need to re-map the data to the report or chart. This is necessary because some layouts or computations may no longer work correctly with major changes to the structure of the data. For more information about data mapping, please see Section 4.1.2 - Report Types and Data Mapping for reports and Section 4.2.3 - Chart Types and Data Mapping for charts.

### 3.1.12. Data Source Updating

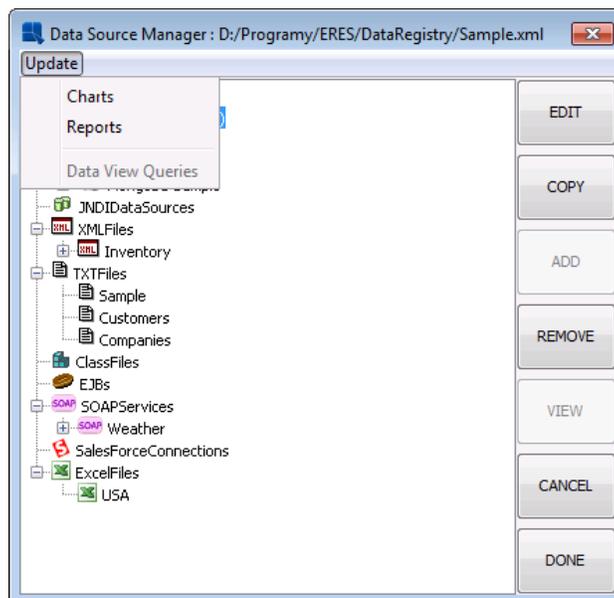
There are many circumstances where you will want to move a group of templates or a complete installation of ERES from one location to another. For example an application can move from a development to a production environment. In each environment, the location and connection information for data sources may be different. In this scenario, updating report templates one by one as detailed in the previous section isn't a feasible way to change the connection information for a large number of templates. Instead, ERES allows you to quickly update a group of templates based on information in the data registry.

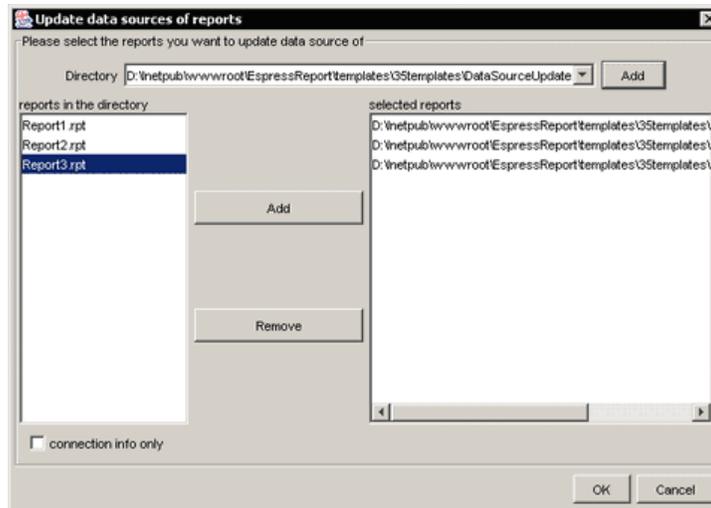
Although report and chart templates maintain an internal copy of the data source information (allowing them to be deployed independently), they also maintain information (location & source) about the data registry from which they were created. Hence, when you modify a report or chart's query (as detailed in Section 3.1.3.4 - Editing Queries), you have the option to save the changes back to the data registry. In order to use this feature, you will have to keep your data registry up to date. This means that query changes should be saved back to the registry and changes in data view structure should be propagated to data view queries.

To use this feature, first make any modifications to the data registry that you want to propagate to the templates. These modifications can include database connection information, file locations for text, XML data files, and even changes to data views or queries that you want to pass to the templates.

Note that if you're moving reports or charts between installations, the data registry file will need to be moved to the same relative location in the new installation. In addition, the associated query files for that registry (`.qry/`, `.dvw/`, `.ddt`) will need to be moved to `/queries/` directory of the new installation (Query file names begin with the name of the registry).

From the data source manager, select *Reports* or *Charts* from the *Update* menu depending on whether you want to update charts or reports. This will bring up a dialog allowing you to select which files you want to update.

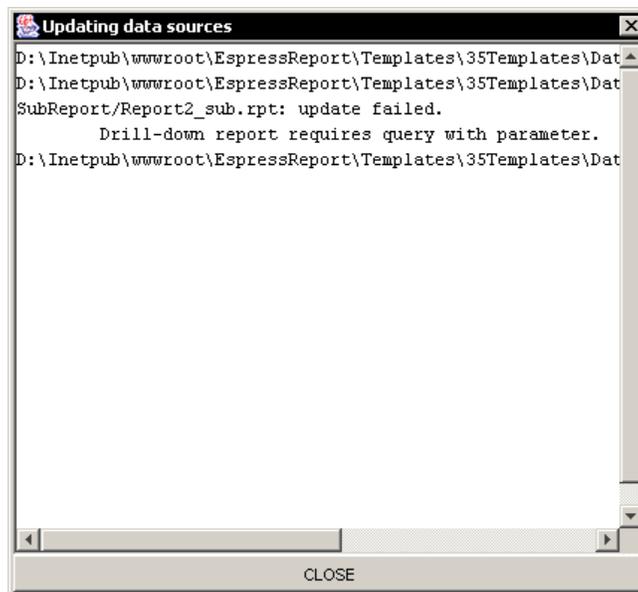




*Select Reports for Data Source Updating*

To select files to update, first browse to the directory that contains the reports/charts. After you select a directory, any report/chart templates will appear in the left side of the dialog. You can select any templates you want like to update and click the *Add* button. You can navigate to as many different directories as you want to select templates. Selecting the *Connection Info Only* option will only update the connection information (database URL, driver, username, password, and locations for XML files, text files and Java classes). Queries and data view information will not be updated in the templates.

Once you finish selecting the templates you want to update, click *OK* and the updating process will begin. A dialog will display the current progress and any errors.



*Updating Data Sources Progress/Log Dialog*

A log file named `UpdateDataSources` will also be written in the root directory of the installation with the contents of the progress screen. Reports that fail the updating for various reasons can be updated manually using the option in Report Designer or Chart Designer.



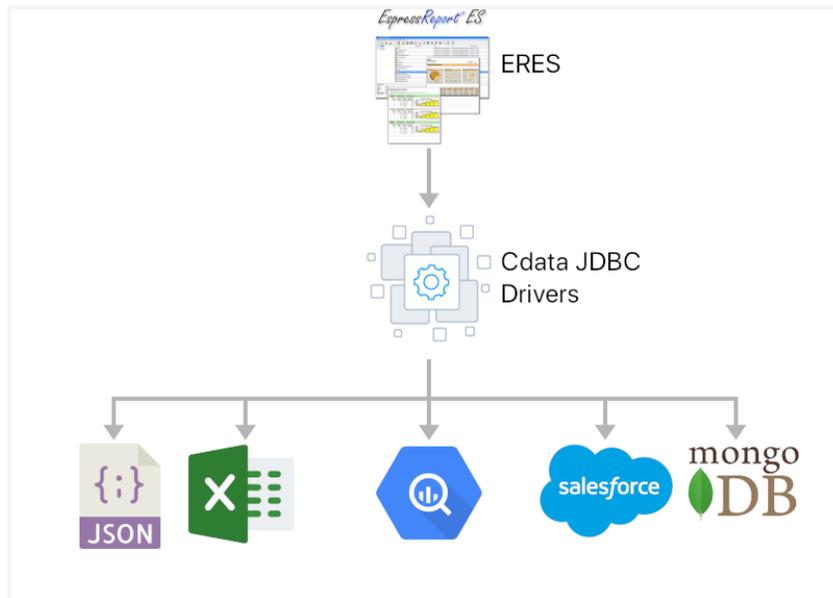
### Note

Only reports or charts for the current data registry will be modified. If you select templates that do not retrieve their data from a source in the current registry, they will be ignored.

## 3.1.13. CData JDBC drivers

If you want to connect to a data source driver that doesn't ship with ExpressReport ES, CData JDBC drivers are an interesting option.

CData JDBC drivers can be also used to increase capabilities of non-database data sources like Excel or JSON.



### Tip

Although CData JDBC drivers are a 3rd party commercial product, you can install a free trial version to test the product before purchasing.



### Note

CData JDBC drivers are a 3rd party product. If you want to use the drivers, you have to purchase them at <https://www.cdata.com>

### 3.1.13.1. Supported CData Drivers

Currently, we support the following CData JDBC drivers:

- Salesforce
- BigQuery
- Excel
- JSON
- MongoDB
- Kintone

Other CData JDBC drivers might work too but we can not guarantee full functionality for unsupported drivers. If you require a connection to a driver that is not listed here, please contact us at <[support@quadbase.com](mailto:support@quadbase.com)>

### 3.1.13.1.1. Excel

Download: <https://www.cdata.com/drivers/excel/jdbc>

Official CData Documentation (for the JDBC driver only): <https://cdn.cdata.com/help/RXF/jdbc>

After downloading the driver, see the following chapters: Section 3.1.13.2 - CData JDBC driver installation, Section 3.1.13.3 - Deploying the CData JDBC Driver in EspressoReport ES and Section 3.1.13.4 - Using the CData JDBC drivers in DataSource Manager

The CData Excel driver allows you to run SQL queries on top of an Excel file just like it was a database. This allows you to use the QueryBuilder, build Data Views in a graphical user interface as well as write SQL queries manually (in Query Builder).



#### Tip

You can also write queries with parameters and multi-value parameters using Excel files as the data source.

Since Excel does not have data types set for columns as a normal database has, determining the data type for the columns can be a bit tricky at times. By default, we added the following parameters to the CData Excel driver connection URL:

```
TypeDetectionScheme=RowScan;
RowScanDepth=10;
```

This makes the CData Excel driver scan the first ten rows when loading the selected Excel file and detect the data source for each column automatically based on the data in the first ten rows.

However, there are multiple options of how to determine the data type for each column.

For more options, read the CData JDBC Excel driver documentation about the TypeDetectionScheme parameter: [https://cdn.cdata.com/help/RXH/jdbc/RSBExcel\\_p\\_TypeDetectionScheme.htm](https://cdn.cdata.com/help/RXH/jdbc/RSBExcel_p_TypeDetectionScheme.htm)

You can change the TypeDetectionScheme value in the Setup Database... dialog in Data Source Manager in the URL field (the field where you entered the Excel file path).

### 3.1.13.1.2. JSON

Download: <https://www.cdata.com/drivers/json/jdbc>

Official CData Documentation (for the JDBC driver only): <https://cdn.cdata.com/help/DJF/jdbc>

After downloading the driver, see the following chapters: Section 3.1.13.2 - CData JDBC driver installation, Section 3.1.13.3 - Deploying the CData JDBC Driver in EspressoReport ES and Section 3.1.13.4 - Using the CData JDBC drivers in DataSource Manager

The CData json driver allows you to run SQL queries on top of a json file just like it was a database. This allows you to use the QueryBuilder, build Data Views in a graphical user interface as well as write SQL queries manually (in Query Builder).



#### Tip

You can also write queries with parameters and multi-value parameters using JSON files as the data source.

### 3.1.13.1.3. Salesforce

To connect to Salesforce:

Download: <https://www.cdata.com/drivers/salesforce/jdbc>

Official CData Documentation (for the JDBC driver only): <https://cdn.cdata.com/help/RFF/jdbc>

After downloading the driver, see the following chapters: Section 3.1.13.2 - CData JDBC driver installation, Section 3.1.13.3 - Deploying the CData JDBC Driver in EspressoReport ES and Section 3.1.13.4 - Using the CData JDBC drivers in DataSource Manager

### 3.1.13.1.4. MongoDB

To connect to MongoDB:

Download: <https://www.cdata.com/drivers/mongodb/jdbc>

A bug fix was added to newer MongoDB driver from [https://cdatabuilds.s3.amazonaws.com/support/DGR-JV\\_8598.exe](https://cdatabuilds.s3.amazonaws.com/support/DGR-JV_8598.exe)

Official CData Documentation (for the JDBC driver only): <https://cdn.cdata.com/help/DGF/jdbc>

After downloading the driver, see the following chapters: Section 3.1.13.2 - CData JDBC driver installation, Section 3.1.13.3 - Deploying the CData JDBC Driver in EspressoReport ES and Section 3.1.13.4 - Using the CData JDBC drivers in DataSource Manager

### 3.1.13.1.5. BigQuery

To connect to BigQuery:

Download: <https://www.cdata.com/drivers/bigquery/jdbc>

Official CData Documentation (for the JDBC driver only): <https://cdn.cdata.com/help/DBF/jdbc>

After downloading the driver, see the following chapters: Section 3.1.13.2 - CData JDBC driver installation, Section 3.1.13.3 - Deploying the CData JDBC Driver in EspressoReport ES and Section 3.1.13.4 - Using the CData JDBC drivers in DataSource Manager

### 3.1.13.1.6. Kintone

To connect to Kintone:

Download: <https://www.cdata.com/drivers/kintone/jdbc>

Official CData Documentation (for the JDBC driver only): <https://cdn.cdata.com/help/DBF/jdbc>

After downloading the driver, see the following chapters: Section 3.1.13.2 - CData JDBC driver installation, Section 3.1.13.3 - Deploying the CData JDBC Driver in EspressoReport ES and Section 3.1.13.4 - Using the CData JDBC drivers in DataSource Manager

## 3.1.13.2. CData JDBC driver installation

The installation process of all CData JDBC drivers is basically the same for all the supported data sources. We'll guide you through the process of installing the CData Excel JDBC driver for example.

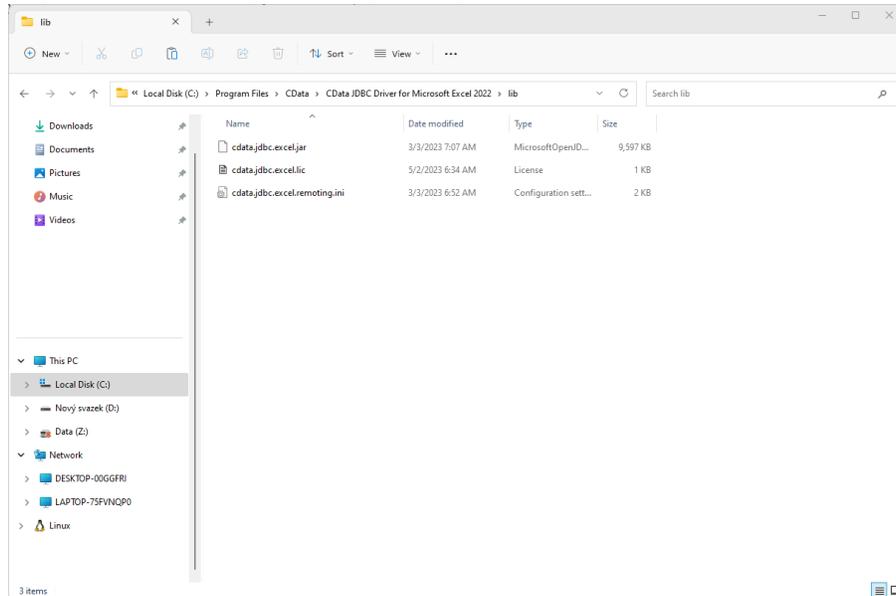
First, install the CData JDBC Driver of your choice. You will find the links to each supported CData JDBC driver in the chapters below.

After downloading the installer, proceed with the setup according to the dialogs.



In the following dialog, you can choose to either install a paid version of the product or a free 30-day trial version. In this example, we'll use the trial version.

When the setup is complete, JDBC Driver will be created in "CData JDBC Driver for Microsoft Excel 2022\lib".



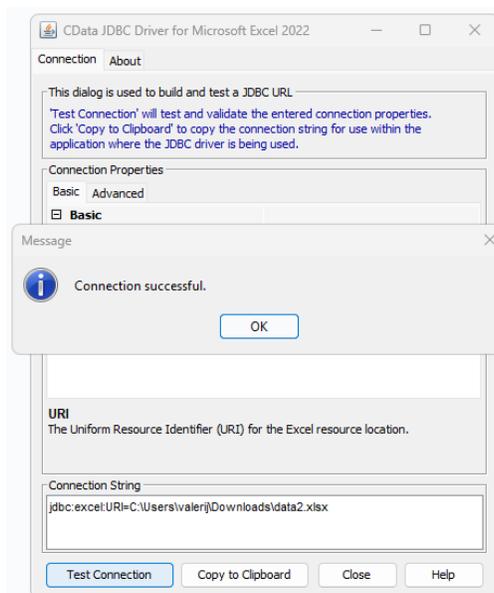
### Tip

Executing "cdata.jdbc.excel.jar" will launch the connection test tool. You can use this tool to test or troubleshoot the CData driver.



### Tip

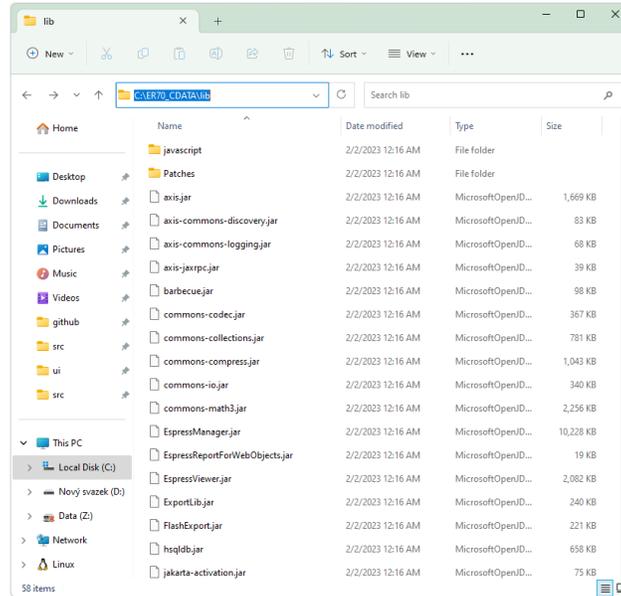
The connection test tool also displays you the connection URL that can be used in EspressoReport ES Data Source Manager.



### 3.1.13.3. Deploying the CData JDBC Driver in EspressoReport ES

Locate the "CData Installation Directory"\CData JDBC Driver for Microsoft Excel 2023\lib (for example: C:\Program Files\CData\CData JDBC Driver for Microsoft Excel 2023\lib) on your hard drive. The directory should contain three files: cdata.jdbc.excel.jar, cdata.jdbc.excel.lic, cdata.jdbc.excel.remoting

Copy the three files to ERES/WEB-INF/

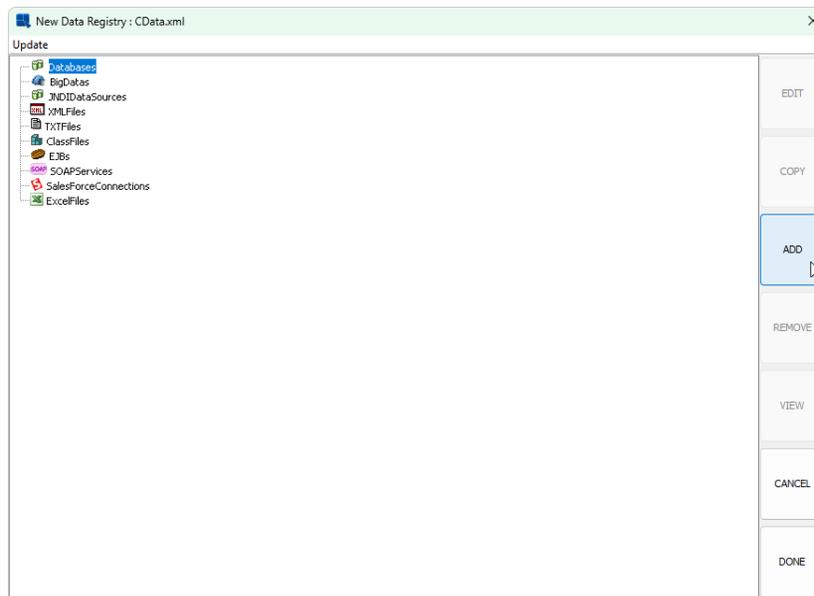


After you've copied the files, restart The EspressManager (if it is running).

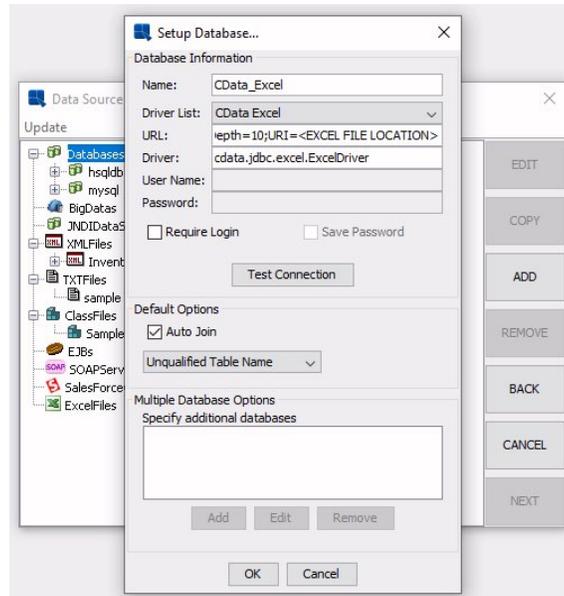
### 3.1.13.4. Using the CData JDBC drivers in DataSource Manager

Launch the DataSource Manager and open a data registry in it (existing one or a new one).

Select the “Databases” option in the tree-list and press the “ADD” Button.



In the “Driver List:” drop-down menu, select “CData Excel” (or any other CData JDBC driver you might be installing).



In the “URL:” text field, replace the placeholders (like “EXCEL FILE LOCATION”) with real values.



### Tip

Alternatively, you can replace the text in the “URL:” text field with the connection string obtained by the CData Test Connection tool described in the previous chapter.

Click OK. You’re done. You can start using Excel files as if they were a database.

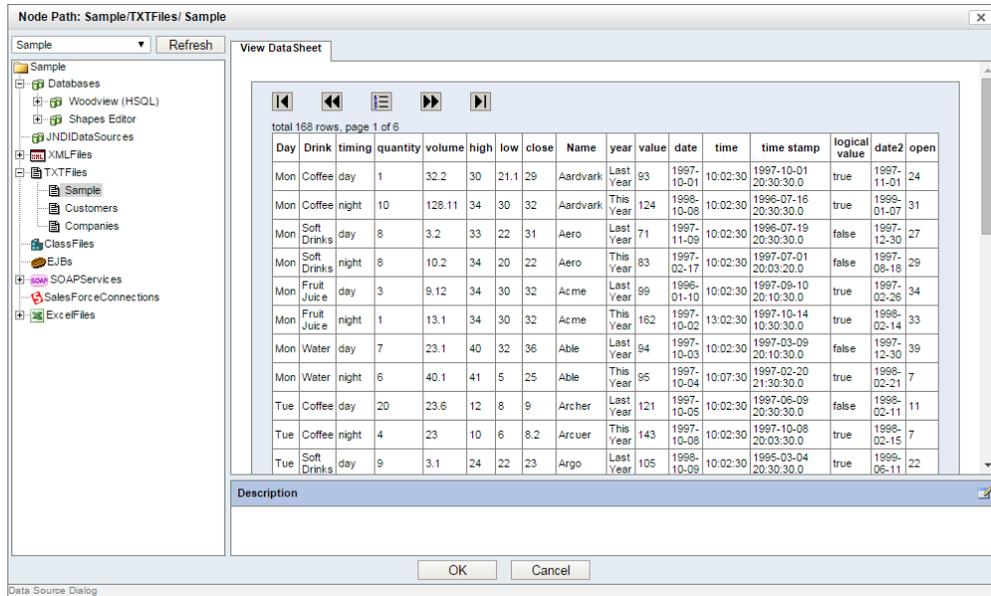
## 3.2. Data in QuickDesigners and Maps

### 3.2.1. Select a Data Source

QuickDesigner Reports, QuickDesigner Charts, Online Maps and SVG Maps use the same interface to manage data - *Data Source Dialog* . The *Data Source Dialog* allows end users to select, filter, and present data without mastering database structures, and all with zero client download. For full control of data sources, please see Section 3.1 - Data in Organizer.

The *Data Source Dialog* prompts you to select the data registry and the data source that you would like to use. A user must have read privileges to one of the registries defined in the Organizer. For more about creating and managing data registries, please see Section 3.1.1 - Managing Data Registries.

Select the data registry from the drop-down menu in the upper part on the left. The pane below displays the content of the selected registry. Select a data source you want to use.



Data Source Dialog

Once a data source is selected, you can see records of the source in the right-hand side *View DataSheet* pane. If you select a *DataView* as the data source, you will be able to create a new *DataView* query in the *DataView Builder* in the right pane. You can also select an existing *DataView* query and modify it in the right pane.

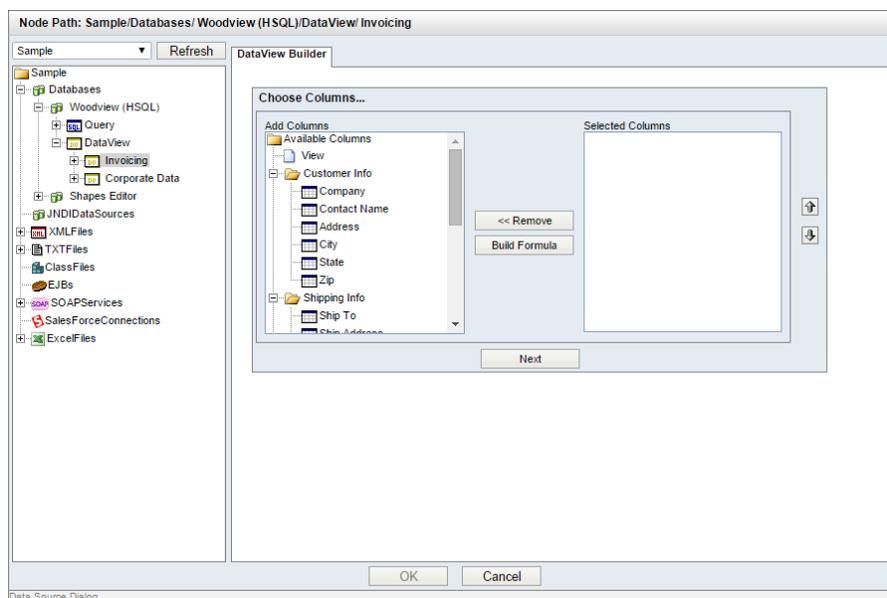
## 3.2.2. Queries

Queries work in conjunction with Data Views. Data Views allow an administrator to pre-select tables and fields from a database and create a local view from which end users can run queries. For more information about data views and setup, please see Section 3.1.3.3 - Data Views.

If you select either a data view or a data view query as the data source, you will be taken to *Data Source Dialog* query interface in the right-hand pane where you can create or modify a query based on the data view.

### 3.2.2.1. Select Fields

If you select a data view as the data source, the *DataView Builder* will open in the right pane allowing you to select fields from the view for your query.

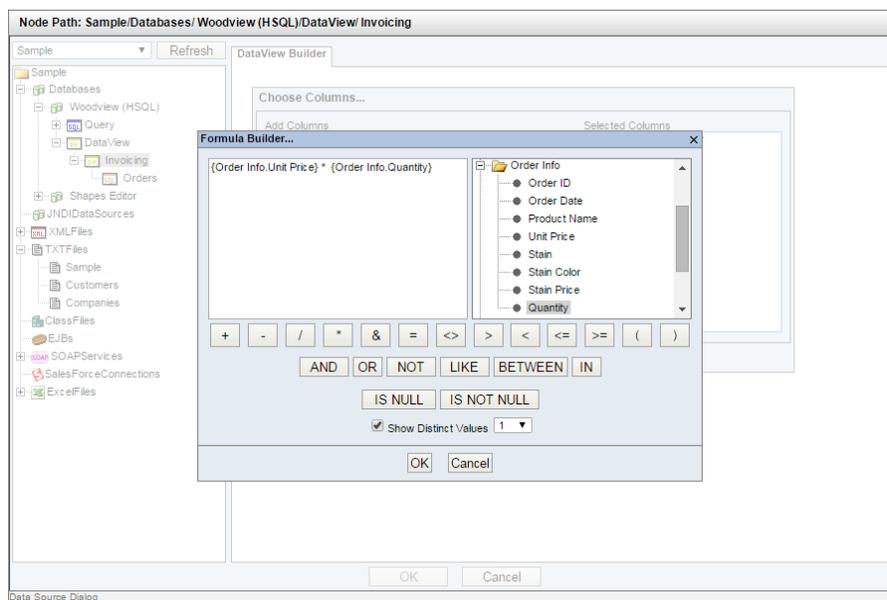


*Select Columns Dialog*

A select list for each heading in the view is generated on the left-hand side of the *DataView Builder*. To select fields for the query, click on a field on the left-hand side. The selected column will be added to the dialog on the right-hand side. To remove fields, you can select the columns from the right-hand side and click the *Remove* button.

### 3.2.2.1.1. Build Columns

You can also create computed columns from the *Choose Columns* dialog. To create a computed column, click the *Build Formula* button. This will launch the Formula Builder interface in a new window.

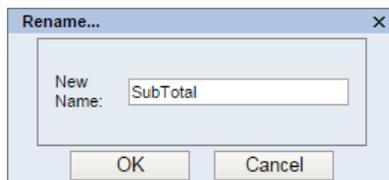


*Formula Builder Window*

The right-hand side of the Formula Builder contains a list of fields in the view, as well as the database functions. The buttons at the bottom are available operators and conditionals. The Boolean and conditional operators are more commonly used for building conditions which is covered in Section 3.2.2.2 - Set Conditions.

Once you have finished constructing the field, click *OK* to close the Formula Builder and return to the *Choose Columns* window where the computed field is added to the *Selected Columns* list. You can edit the formula by selecting it in the list and clicking the *Modify Formula* button.

You can also create an alias for any expression column by selecting it and clicking the *Rename* button. This will bring up a new window for you to specify the alias.



*Rename Dialog*

Once you have finished selecting the columns you would like to use, click the *Next* button to set conditions.

### 3.2.2.1.2. Querying Encrypted Data

Suppose you have a database that has encrypted fields, e.g. social security numbers, tax ID, passport numbers, etc in a tax payer database or immigration database. How do you allow authorized users of your (ERES) system to make ad hoc reports or view reports online so that they can see the actual field values? The other concern is that you do not want any users of your system to know the encryption/decryption key but you want them to be able to see the decrypted data.

ERES provides an advanced feature that allows you to run QuickDesigner Reports and Menu Page/report URL with automatic decryption enabled.

To do this, you need to do two things:

1. You need to create an XML file that gives the database URL, database driver, name of column to be decrypted, and the function to be applied when the data is being retrieved. An example file is as follows:

```
<?xml version="1.0"?>
<!DOCTYPE ReplaceColumnInfoList SYSTEM "QBReplaceColumnInfoList.dtd">
<ReplaceColumnInfoList>
  <ReplaceColumnInfo>
    <Driver>com.mysql.jdbc.Driver</Driver>
    <URL><![CDATA[ jdbc:mysql://prodigy:3306/CAIT ]]></URL>
    <TableName>user</TableName>
    <Pair>
      <ColumnName>user.SSN</ColumnName>
      <ReplaceValue><![
CDATA[cast(AES_DECRYPT(user.SSN,'1111111111111111') as CHAR)]]></
ReplaceValue>
    </Pair>
  </ReplaceColumnInfo>
</ReplaceColumnInfoList>
```

The DTD file used is shown below.

```
<?xml encoding="US-ASCII"?>
<!ELEMENT ReplaceColumnInfoList (ReplaceColumnInfo*)>
<!ELEMENT ReplaceColumnInfo (Driver, URL, TableName, Pair*)>
<!ELEMENT Pair (ColumnName, ReplaceValue)>
<!ELEMENT Driver (#PCDATA)>
<!ELEMENT URL (#PCDATA)>
<!ELEMENT TableName (#PCDATA)>
<!ELEMENT ColumnName (#PCDATA)>
<!ELEMENT ReplaceValue (#PCDATA)>
```

In this example, we are using a MySQL database, the table name is *user*, the encrypted column is *SSN*. When a query is run, the function `AES_DECRYPT(user.SSN, '1111111111111111')` will replace `user.SSN` wherever it may appear in the SQL statement. The function `AES_DECRYPT` is the decryption function in MySQL. The key for decrypting the data is `'1111111111111111'` in this example.

2. You need to set the path to the XML file in the Admin Console (*Server Options* tab, *Security Options* category - see Section 1.4.1.3 - Server Options for more details) and then restart the ERES server.

Once this is set up, when you run a report in Menu Page, using generated report URL or generating reports in QuickDesigner Reports, you will be able to see the real (decrypted) values of the encrypted columns.

You can find the example XML and DTD files shown above under the `<ERESInstallDir>/help/examples/DataDecryption/` directory. You will also find a script that you can use to generate some sample data in MySQL.

When you run a report (in Menu Page, QuickDesigner Reports or with report URL) with the following query,

```
select user.firstName, user.lastName, user.SSN from user;
```

you will see the following result set:

firstName	lastName	SSN
John1	Smith1	1111
John2	Smith2	2222
John3	Smith3	3333
John11	Smith11	11111

This query below,

```
select user.firstName, user.lastName, user.SSN from user where SSN LIKE '1%';
```

should produce the following result in your report.

firstName	lastName	SSN
John1	Smith1	1111
John11	Smith11	11111

You can specify multiple fields in the xml file. To add another field from the same table, simply add another `<pair>` element. To add a field from another table or datasource, add a full `<ReplaceColumnInfo>` element. This feature also supports sub-reports and drill-downs. Any field in these components matching the information in the xml will also be decrypted.

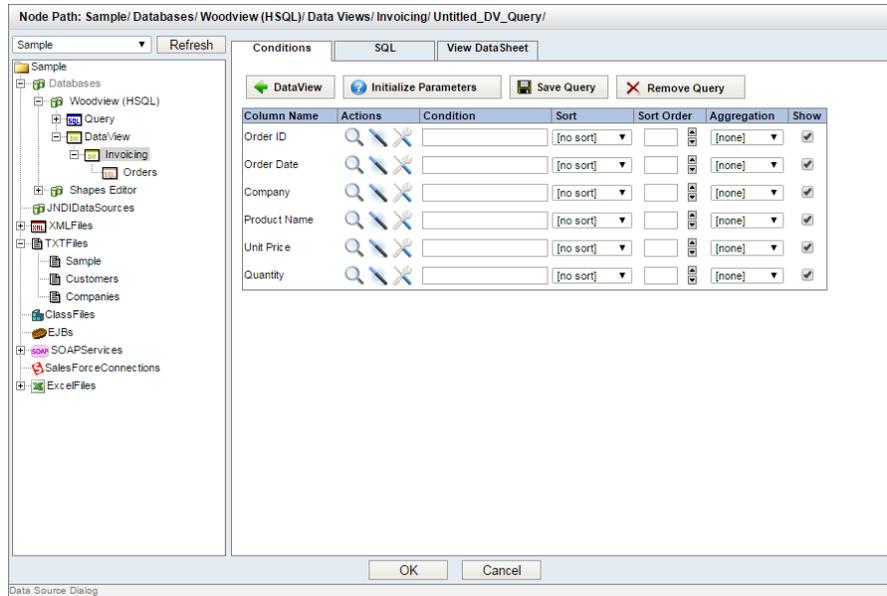


### Note

If you are entering SQL for your query, you must use the full name for the field in order for this feature to work. For example, `SELECT user.SSN FROM user` works, but `SELECT SSN FROM user` will not.

## 3.2.2.2. Set Conditions

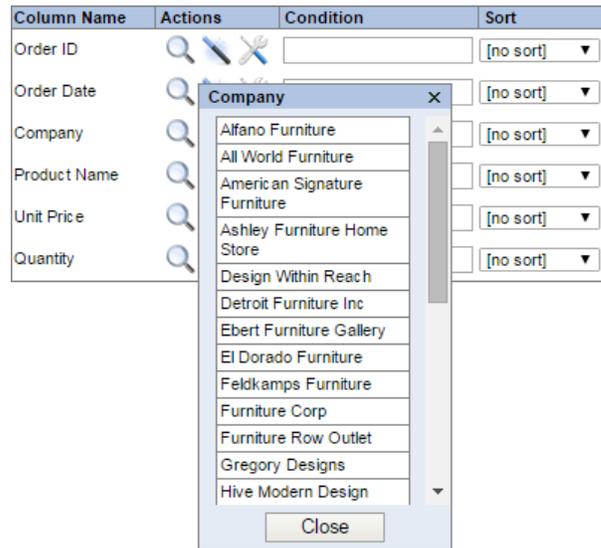
After selecting fields, or if you select a data view query as the data source, the conditions window opens allowing you to set aggregation, grouping, and filtering for the selected columns.



Conditions Dialog

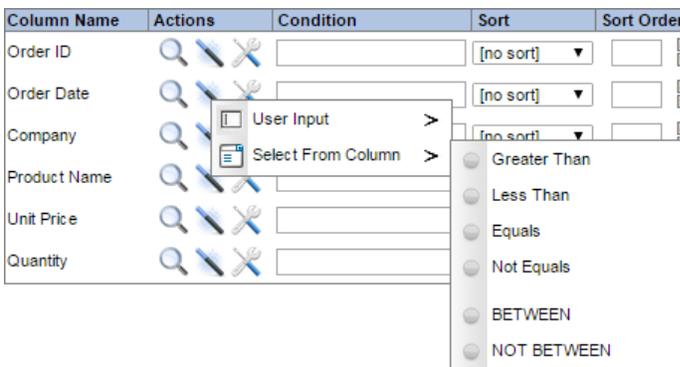
Each selected column is listed along with several options:

**Actions:**  **Show Column Data:** This will bring up a new window with column data.



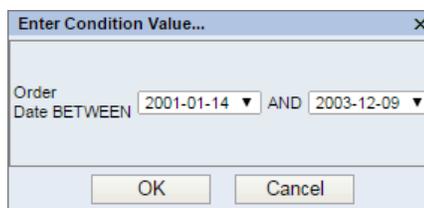
Show Column Data

 **Condition Wizard:** This will bring up a dialog allowing you to set basic conditions from a field. The first dialog allows you to select whether you would like to enter values or select them from a list.



Condition Wizard

Once you have selected the condition, a new dialog opens allowing you to set the values for the condition.



Condition Value Dialog

Select the values that you would like and click *OK*. The condition will automatically be added to the *Condition* field.

 **Build Condition:** This will bring back up the Formula Builder, allowing you to construct an expression for the condition.

**Condition:** This allows you to type a condition for the column.

**Sort:** This allows you to indicate if the column should be sorted. You can select whether to sort in ascending or descending order.

**Sort Order:** This allows you to specify the priority (order) for sorting columns.

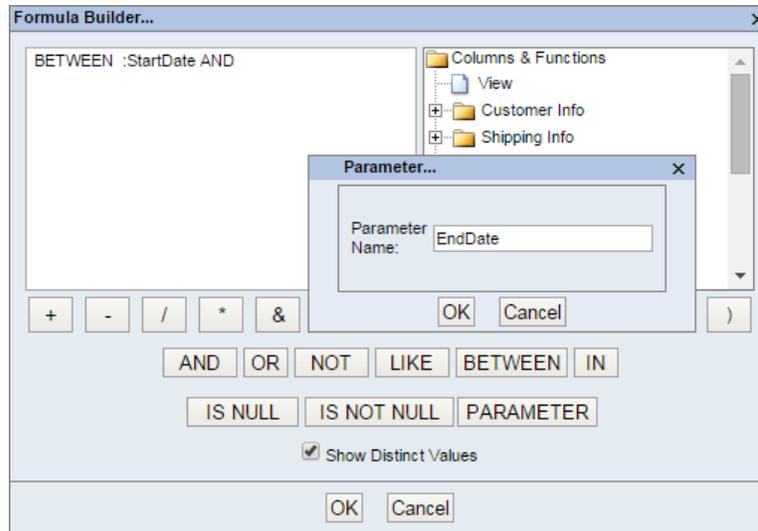
**Aggregation:** This allows you to specify any grouping or aggregation in the query.

**Show:** This option indicates whether the column should be visible or not.

### 3.2.2.2.1. Parameterized Queries

*Data Source Dialog* also allows you to define query parameters. Parameters are defined in the same way as with other query interfaces - using the ":" character. You can define parameters directly in the conditions field, using the convention of :ParameterName to define a parameter.

Parameters can also be defined in the Formula Builder when building conditions. To define a parameter this way, click the  *Build Condition* button to open the Formula Builder, click the *Parameter* button to insert a parameter. This will bring up a new dialog allowing you to enter the parameter name.

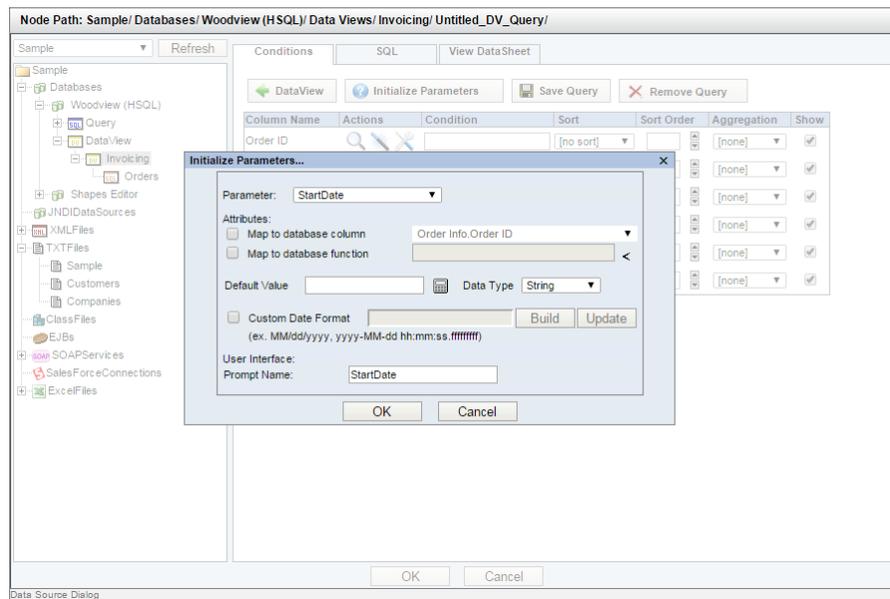


*Specifying a Parameter in Formula Builder*

For more information about query parameters, please see Section 3.1.3.2.2 - Parameterized Queries.

### 3.2.2.2.1.1. Initialize Query Parameters

Once you have specified parameters in the query, they will need to be initialized. You can initialize parameters by clicking the *Initialize Parameters* button. The initialization dialog will also open if you preview the query by clicking on the *View DataSheet* tab or if you click on the *OK* button closing *Data Source Dialog*.



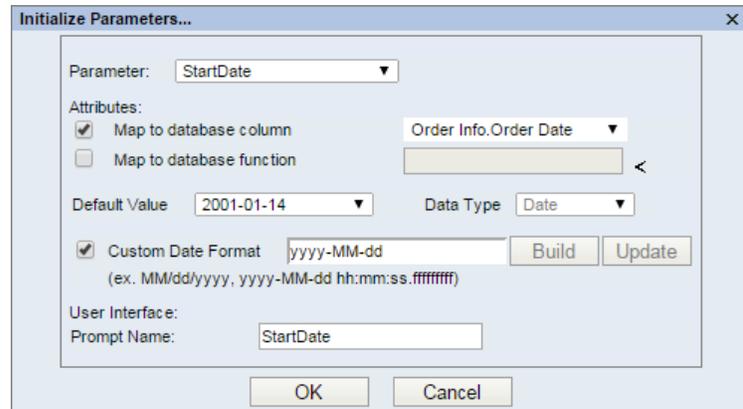
*Initialize Parameter Dialog*

The drop-down list at the top of the dialog indicates which parameter is currently being initialized. You can access and set the options for each parameter by selecting it from this list.

The following options are available for each parameter:

#### **Map to database column:**

This allows you to specify a column from the database whose values will be used for the parameter input. Selecting this option modifies the parameter prompt that you will get when running the report or chart. If you map the parameter to a database column, you will be prompted with a drop-down list of distinct values from which to select a parameter value. If you do not map, you will have to type in the specific parameter value.



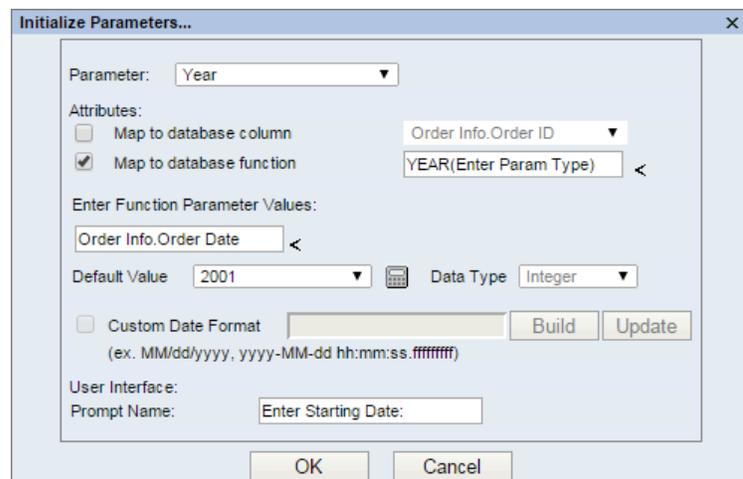
*Initialize Parameter Dialog*

**Map to database function:**

Mapping a parameter to a database function is very similar to mapping to a column. This feature allows you to map to the results of any function from your database. For example, suppose you have the following condition in your query:

```
Year({Orders.OrderDate}) IN (:OrderYear)
```

In the initialize parameter dialog, check the *Map to database function* box, select the function from the list and enter the parameter value for your function. The default value and data type will be automatically updated



*Initialize Parameter Dialog*

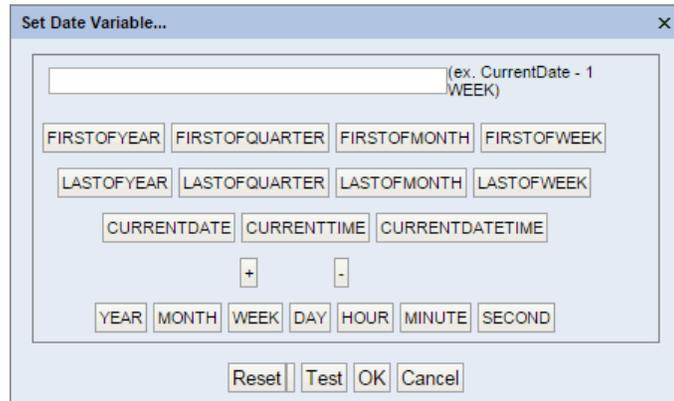
For certain databases, you may notice that the functions available do not specify the parameter types for functions. In this case, there will only be one function parameter field and it will be longer than normal. Selecting a column from the drop down list will be appended at the end of the field rather than replacing the existing text. You must enter all parameter values for the selected function, separating values with commas. Once you click away from this field, the default value field will be filled with function results and the data type will be automatically determined.

**Default Value:**

This allows you to specify a default value for the parameter. Although you do not have to specify a default value, it is recommended that you do so.

**Date Variable:**

This option is only available when the parameter is not mapped to a database column or function, and is only intended for parameters with variable type date/time. When you click this button, the following panel will pop up, listing all the supported keywords.



*Enter Date Variable Dialog*

This dialog allows you to select one of the three keywords: *CurrentDate*, *CurrentTime*, and *CurrentDateTime*. You may add or subtract units of time from the current date/time, allowing you to have a dynamic date range. For example, a report may have the following default values:

```
StartDate: CurrentDate - 1 WEEK
EndDate: CurrentDate
```

This would indicate that every time the report is run, the default prompt should be one week ago to the current date. Other supported measures are *YEAR*, *MONTH*, *DAY*, *HOUR*, *MINUTE*, and *SECOND*. This feature only supports a single addition or subtraction. This feature does not support multi-value parameters. Note: QuickDesigner Reports date values (in the right-hand panel) correspond to the variable values entered here, not the keywords themselves as in Report Designer.

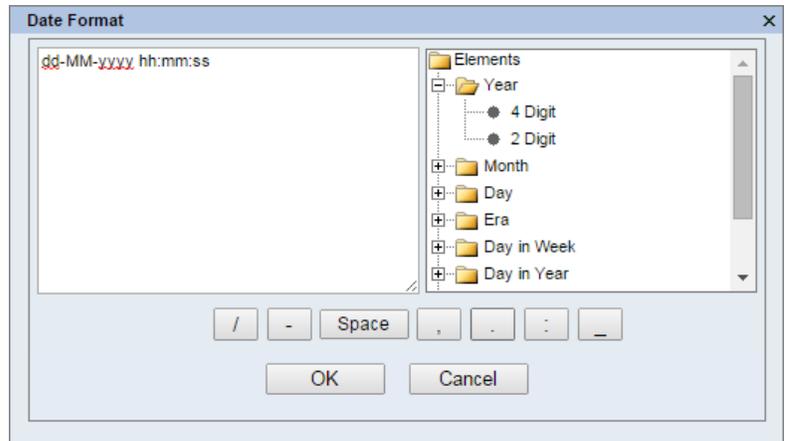
You can also use functions to define the parameter value.

**Data Type:**

This allows you to specify the data type for the parameter value(s). If you have mapped the parameter to a column, the data type is set automatically.

**Custom Date Format:**

This allows you to set the format in which the date parameter should be entered. This option is only available when you have mapped the parameter to a date/time column or when you not mapped the parameter at all and the parameter data type is *date*, *time*, or *timestamp*. When you check this option you can specify a date format by either using the builder interface or by entering the date format in a combination of characters that represent time elements.



*Date Format Dialog*

If you click the *Build* button, the above dialog will appear. The date/time representations are listed on the right and optional spacers and symbols are shown as a collection of buttons on the bottom. Once you have finished creating the format, click *OK* to save.

The date and/or time format is a series of characters and delimiters. Letters are used to represent different elements of date/time data. For more on the characters and their formatting, please refer to the documentation for the `printDate()` report function in Section 4.1.6.2.8.3 - Date/Time Functions.

If the parameter is mapped to a column, once you have finished designing the format, click on the *update* button to refresh the parameter prompt with the new format.

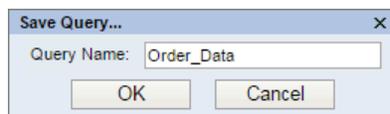
**Prompt Name:**

This allows you to specify the prompt that is given to the user in the parameter dialog.

Once you have finished setting values for all the parameters, click *Ok* to close the dialog and save the settings.

### 3.2.2.2.2. Save Queries

You can save the created query, or save the changes to the query if you selected a data view query to begin with by clicking the *Save Query* option in the conditions window. This will bring up a dialog prompting you to specify a name for the query.

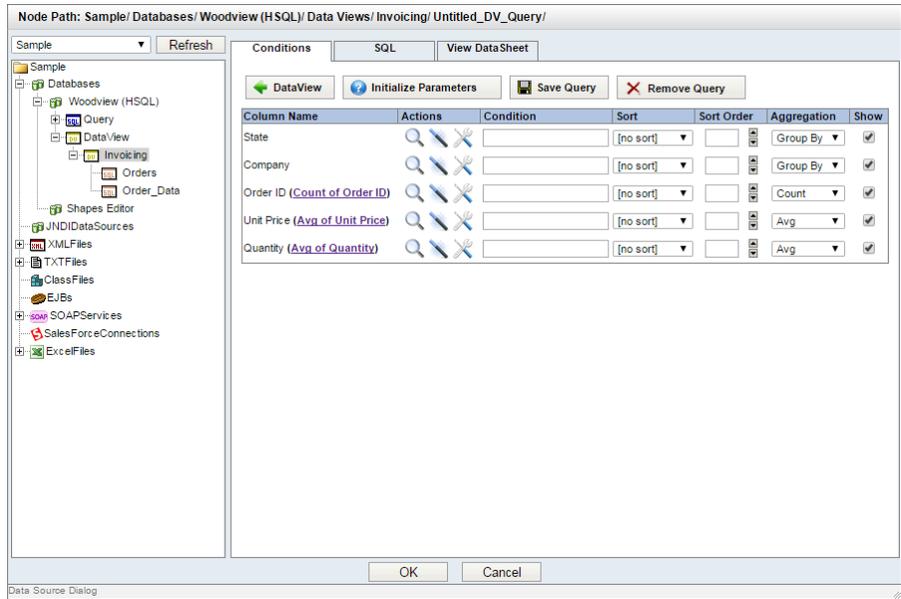


*Save Query Dialog*

Once you have specified the name, click *OK*. The query will be saved in the data registry as a new node under the data view.

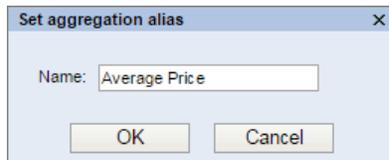
### 3.2.2.2.3. Aggregation Alias

If you add aggregation and group bys to the query, you are also able to specify an alias for the aggregated field.



Set Alias

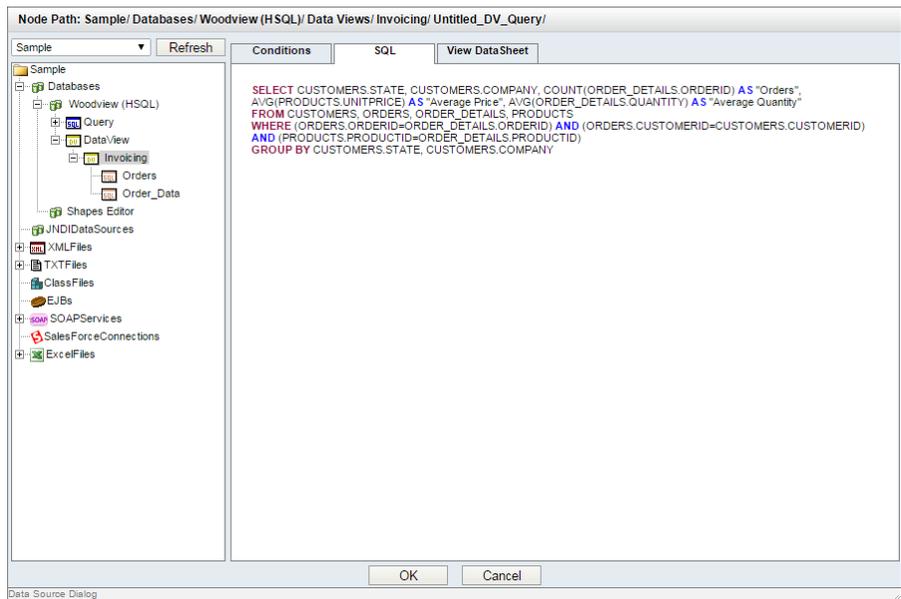
Clicking on the underlined option next to the column name will pop up a dialog allowing you to specify a name for the field. The name you specify here will replace the actual field name.



Alias Dialog

### 3.2.2.3. View SQL Query

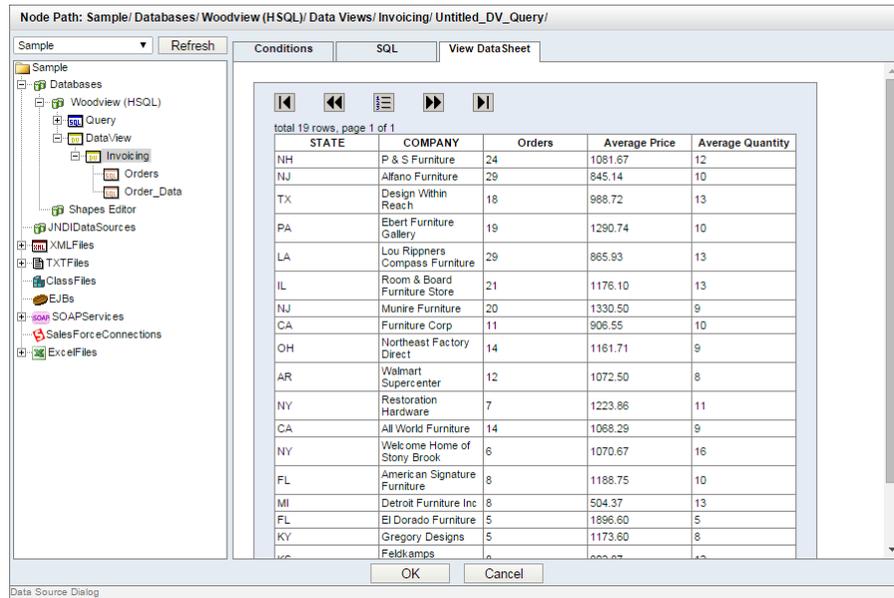
You can view the constructed query in the *SQL* tab of the *Data Source Dialog*.



SQL View

### 3.2.2.4. Preview Query Results

You can preview the query at anytime by clicking the *View DataSheet* tab in the *Data Source Dialog*.



*Query Results*

You can navigate through the result set using the icons at the top of the window.

 Go to the first page of the data table.

 Go to the previous page of the data table.

 Set the number of records to display per page. This will open a new dialog prompting you to specify the record number.

 Go to the next page of the data table.

 Go to the last page of the data table.

---

# Chapter 4. Designing Reports & Charts

## 4.1. Report Designer

### 4.1.1. Introduction to Report Designer

The Report Designer is a graphical user interface launched within the Organizer that allows users to create and customize reports. The simple drag and drop style interface and extensive editing/formatting capabilities makes the report design quick and easy.

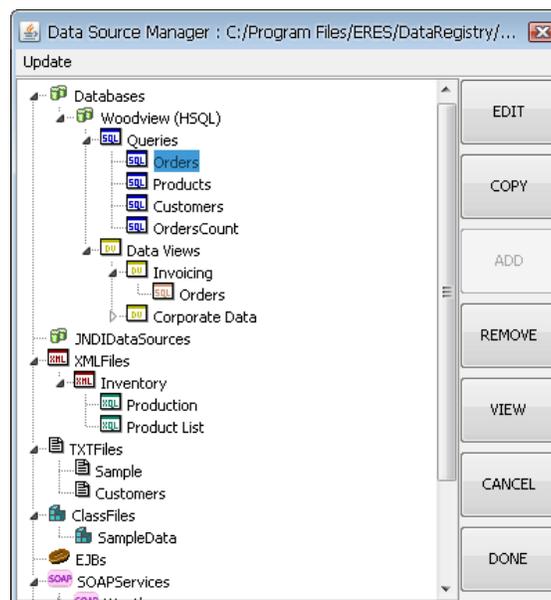
#### 4.1.1.1. Starting Report Designer

The Report Designer interface is always loaded from within the Organizer. To start a new report, you can click the *Report Designer* button on the toolbar or select *Report Designer* from the *View* menu. You can also start Report Designer when creating or editing data sources in the registry. The *View* button allows you to preview a data source and select to build a chart or report. In addition, you can open the Report Designer to edit a report template file in the Organizer. To do this, first select the file that you would like to open, then select *Open File* from the *File* menu, or press **Ctrl+O**. You can also right click on the file and then select *Open File* from the pop-up menu, or simply double click the file.

#### 4.1.1.2. Selecting a Data Source

The first step in designing a report is to select the data source from which the report is to be drawn. The first time you select to start a new report you will be prompted to select the data registry that you would like to use. If there are not currently any available registries (meaning that you have not created any or you do not have privileges to view any), you will be prompted to go the Data Registry Manager to create one. For more on data sources, please see Section 3.1 - Data in Organizer.

Once you have selected a registry, the Data Source Manager window will open allowing you to select, add, or modify a data source that you would like to use for the chart or report. Note that the registry will not open if you are building a report from the registry in the *Modify Data Sources* dialogs.



*Data Source Manager Window for Report Designer*

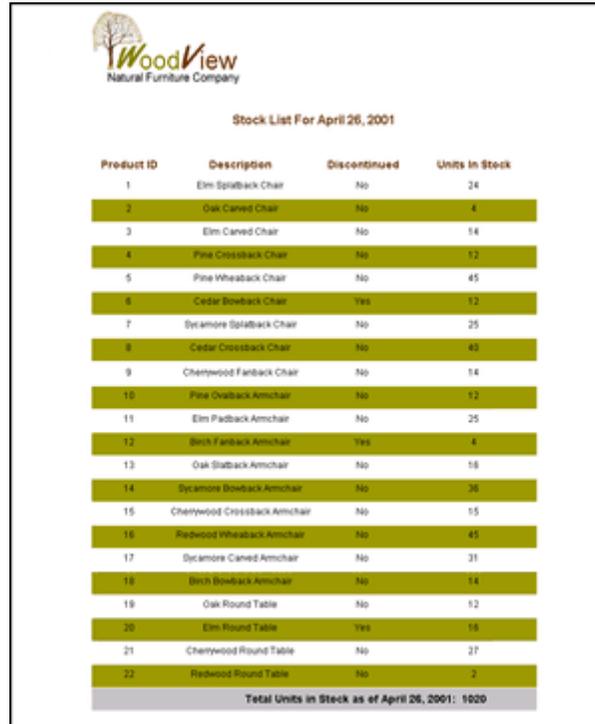
To use a particular data source, select the query, XML query, text file, data view, or data view query that you would like to use, and click the *NEXT* button. With the exception of data views, which will require you to select fields and set conditions first, the next screen will present the first twenty records from the data source. From this screen you can see all of the records returned by the selected source, by checking the *Show All Records* box.

## 4.1.2. Report Types and Data Mapping

ReportDesigner supports five basic report types: simple columnar, summary break, crosstab, master & details, and mailing label. Each report type has slightly different mapping options and can be used to generate a different style of report.

### 4.1.2.1. Simple Columnar Report

The simple columnar report is the most basic report type supported by ReportDesigner. It displays columnar data in a single table without any grouping or breaks.



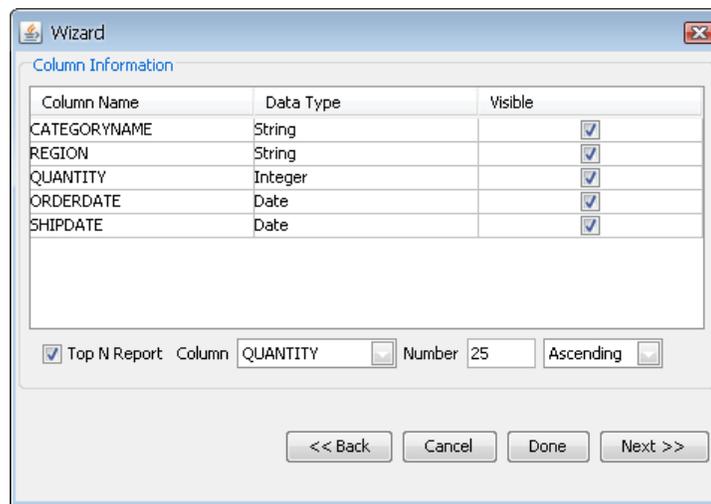
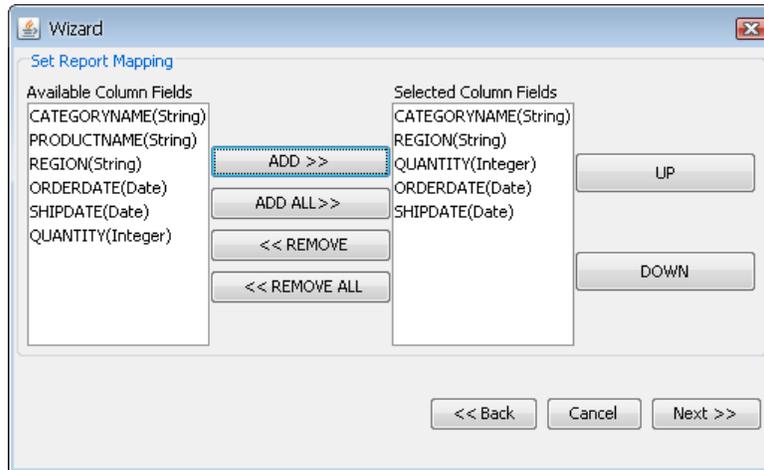
Product ID	Description	Discontinued	Units in Stock
1	Elm Splatback Chair	No	24
2	Oak Carved Chair	No	4
3	Elm Carved Chair	No	16
4	Pine Crossback Chair	No	12
5	Pine Wheelback Chair	No	45
6	Cedar Bowback Chair	Yes	12
7	Sycamore Splatback Chair	No	25
8	Cedar Crossback Chair	No	40
9	Cherrywood Fanback Chair	No	14
10	Pine Ovalback Armchair	No	12
11	Elm Padback Armchair	No	25
12	Birch Fanback Armchair	Yes	4
13	Oak Splatback Armchair	No	18
14	Sycamore Bowback Armchair	No	30
15	Cherrywood Crossback Armchair	No	15
16	Redwood Wheelback Armchair	No	45
17	Sycamore Carved Armchair	No	31
18	Birch Bowback Armchair	No	14
19	Oak Round Table	No	12
20	Elm Round Table	Yes	16
21	Cherrywood Round Table	No	27
22	Redwood Round Table	No	2
Total Units in Stock as of April 26, 2001:			1020

*Simple Columnar Report*

#### 4.1.2.1.1. Data Mapping

The first step in data mapping for this type is to select the columns you want to include in your report from the *Set Report Mapping* window in the Report Wizard. The left side of the window lists all available columns from your query or data file and the right side lists columns that will be added to the report. Click on a column name to select it (**Shift+Click** for multiple selections) and press the **ADD** or **REMOVE** buttons to add it or to remove it from the report. The order in which you select columns in this window is the order in which they will appear in your report. You can change the order of the selected columns by highlighting the column you want to move and clicking the **UP** or **DOWN** buttons. Fields can also be moved by clicking and dragging the selected field.

The second step for report mapping is to set the column options. For simple columnar reports there is only one available option. The window displays a table showing all the columns you have selected for the report and all available options. The first field displays column names, the second one displays data type, and the third one displays a check box called *Visible*. Checking or un-checking this box will make the column visible or invisible within the body of the report (All columns are visible by default).



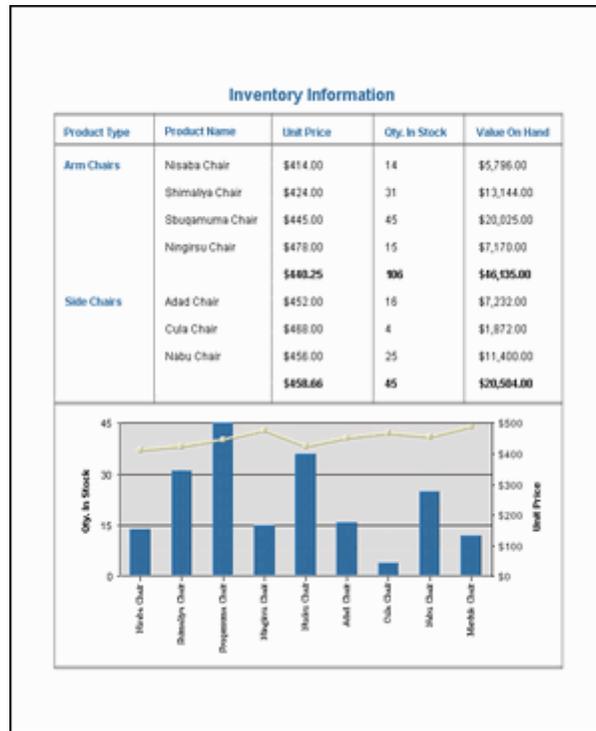
*Simple Columnar Report Mapping*

#### 4.1.2.1.1.1. Top N Report

You can also create a Top N report using the columnar format. A Top N report will order and show you the highest set of values based on a particular column in the report. For example, you want to show top five customers based on total sales. To do this, check the *Top N Report* checkbox at the bottom of the data mapping window. You can then specify which column you want to use as the measure (for the above example, it would be total sales column), the number you want to retrieve (i.e. 10 or 20, etc), and whether you want to show the columns in ascending or descending order. The columnar report will then return the number of records specified sorted by the highest value for the specified column.

#### 4.1.2.2. Summary Break Report

Like the simple columnar report, the summary break report type takes columnar data and presents it in a tabular form. However, unlike the simple columnar report, it allows you to break data into sections and insert summary fields.



Summary Break Report

#### 4.1.2.2.1. Data Mapping

The first step in data mapping for this type is to select the columns you want to include in your report from the Set Mapping window in the Report Wizard. The left side of the window lists all available columns from your query or data file and the right side lists the columns that will be displayed in the report. Click on a column name to select it (**SHIFT+Click** for multiple selections) and press the *ADD* or *REMOVE* buttons to add it or remove it from the report. The order in which you select columns in this window is the order in which they will appear in your report. You can change the order of the selected columns by highlighting the column you want to move and clicking the *UP* or *DOWN* buttons. Fields can also be moved by clicking and dragging the selected field.

The second step in data mapping is to set column options. There are five column options available for summary break reports. The window displays a table showing all of the columns that you have selected for your report and all available options. The first field displays column names, the second one displays data type, and the third one displays a check box marked *Visible*. Checking or un-checking this box will make the column visible or invisible within the body of the report (All columns are visible by default). The fourth field is the row break field. Checking this box indicates that the report will break and insert column summaries every time the selected column field changes. By default, the first column selected for the report is the break field. The fifth field allows you to select aggregation. If the *Perform Column Aggregation* checkbox at the bottom of the window is checked, the selected aggregation will be performed on the entire column and the report will only contain summarized data. If the box is not checked, the report will display all data and only insert aggregations as summaries after each row break.

The drop-down menu allows you to select the aggregation operation to be performed for that column. The aggregation operations that can be performed are: none, sum, maximum, minimum, count, average, first, last, sum of squares, variance, standard deviation, and count distinct. If you select to perform column aggregation, you must select an aggregation option for each column. Columns selected as row break fields cannot be aggregated.

For example, suppose we have the following data table:

Order#	Product	Quantity
12	Chair	2
12	Table	3

Order#	Product	Quantity
14	Cabinet	2
14	Table	5

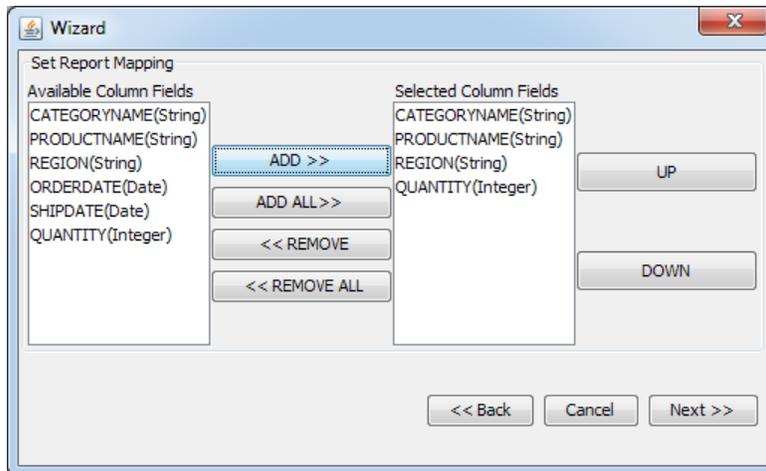
Setting Order # as the row break field and the aggregation on Quantity to Sum without checking *Perform Column Aggregation* will produce the following report:

Order#	Product	Quantity
12	Chair	2
	Table	3
		<b>5</b>
14	Cabinet	2
	Table	5
		<b>7</b>

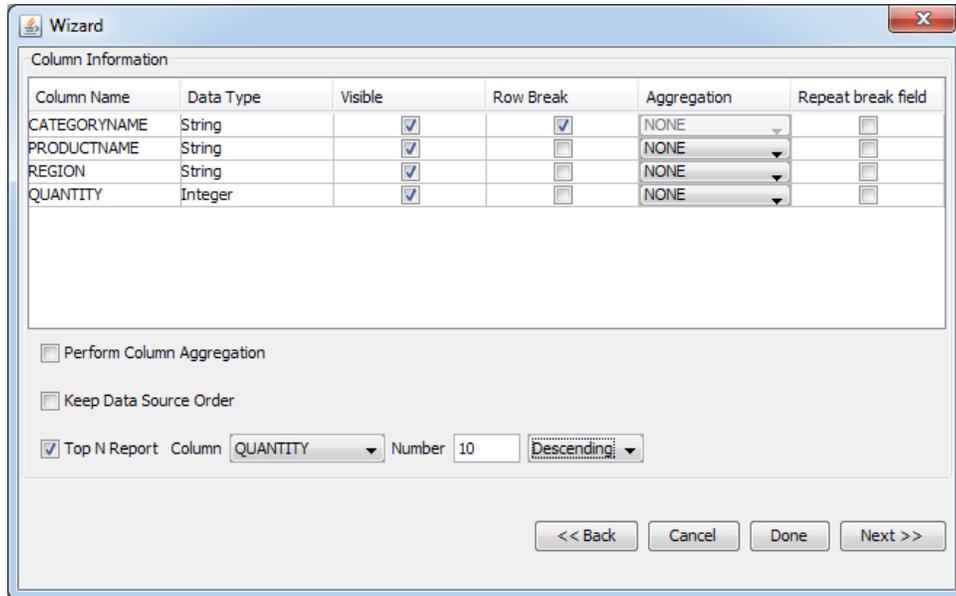
However, selecting *Perform Column Aggregation* and changing the Product aggregation to Count will produce the following report:

Order#	Product	Quantity
12	2	5
14	2	7
	<b>2</b>	<b>5</b>

The sixth field allows you to select whether to repeat a break field or not. By default, a row break column will only print each distinct value in the column once (i.e. once for each group). Selecting this option will cause the break fields to repeat for each row of data in the group.



*Summary Break Report Mapping*



*Summary Break Report Mapping*

If you want to keep the data ordering specified in the data source, select the *Keep Data Source Order* option. To learn more about this feature, see Section 4.1.2.9 - Keep Data Source Order.

#### 4.1.2.2.1.1. Top N Report

You can also create a Top N report using the summary break format. A Top N report will order and show you to highest set of values based on a particular column in the report. For summary break reports, you can also specify to show the highest values for each group. For example, you may want to show the top five customers in each region based on total sales. To do this, check the *Top N Report* checkbox at the bottom of the data mapping window. You can then specify which column you want to use as the measure (for the above example, it would be total sales column), the number that you want to retrieve (i.e. 10 or 20, etc), and whether you want to display the columns in ascending or descending order. The summary break report will then return the number of records specified (or number of records specified for each row break) sorted by the highest value for the specified column.

#### 4.1.2.3. Crosstab Report

A crosstab report is a report format that shows and summarizes columnar data in a matrix-like form. Crosstab reports often resemble spreadsheets. Both rows and columns are summarized, allowing multi-dimensional data to be displayed in a two-dimensional format.

SALES						
Category	Product	East	South	Midwest	West	Quantity
Side Chairs	Ishtar Chair	32	26	18	Null	76
	Shamash Chair	18	28	18	4	68
	Enlil Chair	30	32	40	8	110
	Ninurta Chair	18	34	21	Null	73
	An Chair	18	12	34	Null	64
	Enki Chair	26	39	90	16	171
	Ninursag	39	Null	21	13	73
	Nergal Chair	29	30	34	Null	93
	Zabada Chair	12	49	34	7	102
			222	250	310	48
Arm Chairs	Marduk Chair	24	30	Null	14	68
	Cula Chair	58	16	47	8	129
	Nusku Chair	44	31	37	12	124
	Shimaliya	33	41	Null	7	81
	Nisaba Chair	82	39	3	49	173
	Adad Chair	38	Null	12	Null	50
	Ningirsu Chair	12	15	Null	Null	27
	Sbuqamuma	54	25	Null	Null	79
	Nabu Chair	Null	23	14	16	53
		345	220	113	106	784
Round Tables	Apep Table	25	23	31	6	85
	Anubis Table	13	21	36	4	74
	Bast Table	34	Null	18	Null	52
	Ningizida Table	Null	20	23	31	74
	Amen Table	Null	33	Null	Null	33

Crosstab Report

The actual report is only constructed during running or preview. Since the crosstab table is essentially constructed from scratch every time the report is run, it is easier to create a smoothly collapsing and expanding crosstab table.

#### 4.1.2.3.1. Data Mapping

Report mapping for crosstab reports is more complicated than other report types and may require some planning to be able to execute it correctly.

The first step in data mapping for this type is to select the columns you want to include in your report from the *Set Report Mapping* window in the Wizard. The left side of the window lists all available columns from your query or data file and the right side lists the columns that will be displayed in the report. Click on a column name to select it (**Ctrl+click** for multiple selections) and press the *ADD* or *REMOVE* buttons to add it or remove it from the report. The order in which you select columns in this window is the order in which they will appear in your report. Once you select the columns, click the *Next* button.

The second step in data mapping is to set column options.

The top of the column options window displays a table showing all of the columns that you have selected for your report and all available options.

- The first field displays the *Column Name*.
- The second field displays the *Data Type*.
- The third field displays a checkbox marked *Visible*. Checking or un-checking this box will make the column visible or invisible within the body of the report.
- The fourth field is the *Row Break* field. Selecting a column as the row break field will cause the report to insert a new crosstab row for each unique entry in the selected column.
- The fifth field is the *Column Break* field. Selecting a column as a column break field will cause the report to create a new column for each unique entry in the database.
- The sixth field is the *Column Break Value* field. Columns that you select as column break values become the fields that are summarized in the report.
- The seventh field allows you to select column *Aggregation*. A drop-down menu allows you to select the aggregation to be performed for that column. The aggregation operations that can be performed are: *sum*, *maximum*,

*minimum, count, average, first, last, sum of squares, variance, standard deviation, and count distinct.* Columns that have been selected as *Row Break* or *Column Break* fields cannot be aggregated.

- The eighth field *Order* allows you to specify ordering for the column break column. This order will determine how the crosstab columns are drawn from left to right. Ordering options are *not sorted, ascending, and descending.*

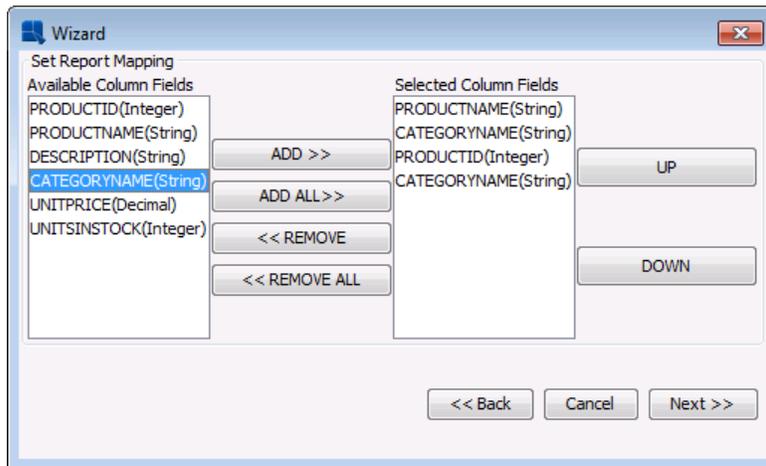


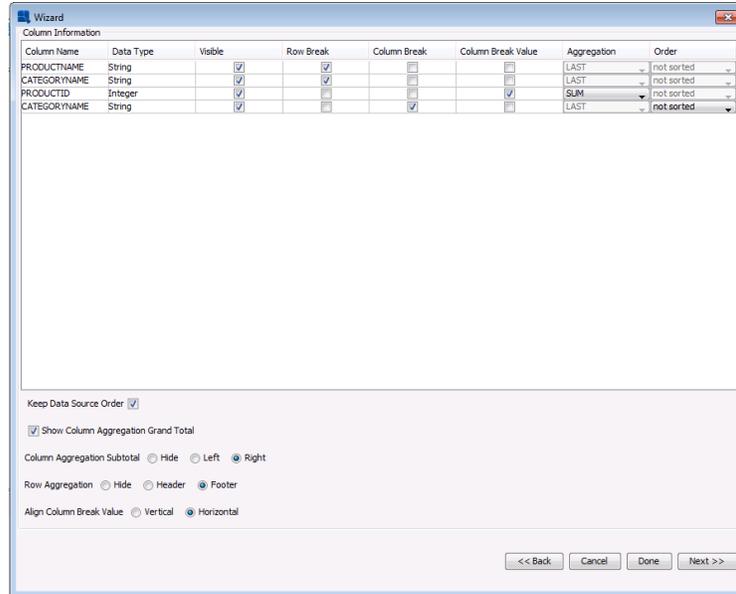
**Note**

The aggregation that will be performed depends on the aggregation of the *Column Break Value* column(s). For example: if you use the *MAX* aggregation, the greatest value from the row will be displayed in the *Grand Total* column.

- Keep Data Source Order:** If you want to keep the data ordering specified in the data source, select this option. To learn more about this feature, see Section 4.1.2.9 - Keep Data Source Order.
- Show Column Aggregation Grand Total:** If this option is enabled, a column will be added into the report. Each row of the column will aggregate all values from the corresponding row into a single value.
- Column Aggregation Subtotal:** Subtotal column is very similar to the *Grand Total* column, except it doesn't aggregate the whole row, but each one of the *Column Break* group. This option allows you to disable the subtotal column or set its position (right or left to the *Column Break* group).
- Row Aggregation:** This is also very similar to the *Grand Total* column. The main difference is that this option doesn't aggregate rows, but columns. Also, it doesn't add a column to the report, but it adds a row instead. Use this options to disable the *Row Aggregation* row, or to adjust it's position (on top of the table or below the table).
- Align Column Break Value:** If you select more than one *Column Break Value* columns, use this option to choose whether the values will be aligned in rows or columns.

For example, assume we have following crosstab mapping:





*Crosstab Fixed-Field Mapping*

Setting CATEGORYNAME and PRODUCTNAME as **Row Break** fields, PRODUCTID as a **Column Break Value** field with an aggregation of **SUM**, and CATEGORYNAME as a **Column Break** field with the *Show Column Aggregation Grand Total* option enabled will produce a report from the beginning of the *Crosstab Report* section.

You can easily change the crosstab settings using the *Change Data Mapping* option under the *Data* menu. This will open the crosstab report mapping dialog again. To learn more about this feature, see Section 4.1.2.8 - Change Data Mapping.

The following table shows various formatting in crosstab reports:

	<b>Column Aggregation Subtotal</b>	<b>Row Aggregation</b>	<b>Align Column Break Value</b>	<b>Template</b>
Report A [ <a href="http://data.quadbase.com/Docs70/help/manual/code/export/Fixed_field_Crosstab_A.pdf">http://data.quadbase.com/Docs70/help/manual/code/export/Fixed_field_Crosstab_A.pdf</a> ]	Right	Footer	Horizontal	Template A [ <a href="http://data.quadbase.com/Docs70/help/manual/code/templates/Fixed_field_Crosstab_A.pak">http://data.quadbase.com/Docs70/help/manual/code/templates/Fixed_field_Crosstab_A.pak</a> ]
Report B [ <a href="http://data.quadbase.com/Docs70/help/manual/code/export/Fixed_field_Crosstab_B.pdf">http://data.quadbase.com/Docs70/help/manual/code/export/Fixed_field_Crosstab_B.pdf</a> ]	Right	Header	Horizontal	Template B [ <a href="http://data.quadbase.com/Docs70/help/manual/code/templates/Fixed_field_Crosstab_B.pak">http://data.quadbase.com/Docs70/help/manual/code/templates/Fixed_field_Crosstab_B.pak</a> ]
Report C [ <a href="http://data.quadbase.com/Docs70/help/manual/code/ex-">http://data.quadbase.com/Docs70/help/manual/code/ex-</a>	Left	Footer	Horizontal	Template C [ <a 482="" 513="" 951="" 967"="" data-label="Page-Footer" href="http://data.quadbase.com/Docs70/help/manual/code/tem-&lt;/a&gt;&lt;/td&gt; &lt;/tr&gt; &lt;/tbody&gt; &lt;/table&gt; &lt;/div&gt; &lt;div data-bbox="> <p>349</p> </a>

	Column Aggregation Subtotal	Row Aggregation	Align Column Break Value	Template
port/Fixed_field_Crosstab_C.pdf ]				plates/Fixed_field_Crosstab_C.pak ]
Report D [ <a href="http://data.quadbase.com/Docs70/help/manual/code/export/Fixed_field_Crosstab_D.pdf">http://data.quadbase.com/Docs70/help/manual/code/export/Fixed_field_Crosstab_D.pdf</a> ]	Left	Header	Vertical	Template D [ <a href="http://data.quadbase.com/Docs70/help/manual/code/templates/Fixed_field_Crosstab_D.pak">http://data.quadbase.com/Docs70/help/manual/code/templates/Fixed_field_Crosstab_D.pak</a> ]
Report E [ <a href="http://data.quadbase.com/Docs70/help/manual/code/export/Fixed_field_Crosstab_E.pdf">http://data.quadbase.com/Docs70/help/manual/code/export/Fixed_field_Crosstab_E.pdf</a> ]	Left	None	Vertical	Template E [ <a href="http://data.quadbase.com/Docs70/help/manual/code/templates/Fixed_field_Crosstab_E.pak">http://data.quadbase.com/Docs70/help/manual/code/templates/Fixed_field_Crosstab_E.pak</a> ]

#### 4.1.2.3.2. Transposing Data

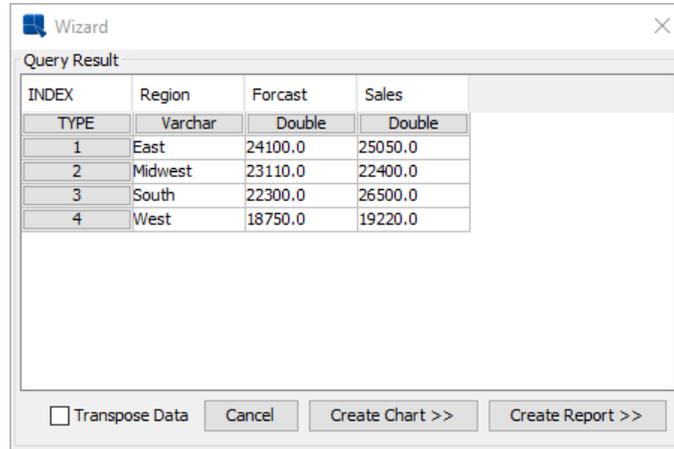
The crosstab report also provides a unique feature that allows users to create a transposed report. Data transposition allows users to create a report presentation where the input data is essentially rotated by 90 degrees. Columns become rows and rows become columns. For example, you have a following set of data:

Region	Forecast	Sales
East	24100	25050
Midwest	23110	22400
South	22300	26500
West	18750	19220

The data transposition feature can be used to create a report that will look like this:

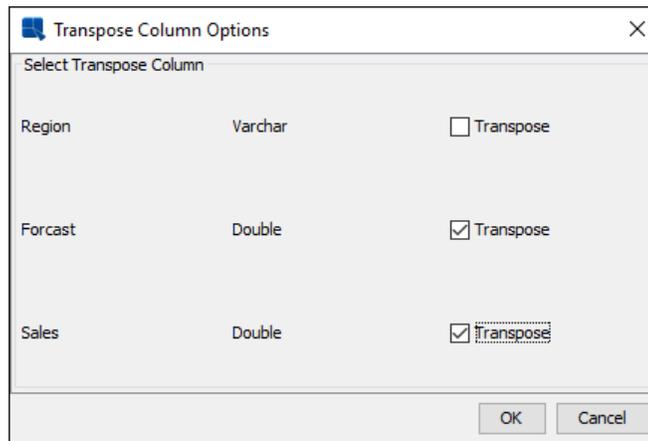
	East	Midwest	South	West
Forecast	24100	23110	22300	18750
Sales	25050	22400	26500	19220

To transpose the data, click the *Transpose Data* option at the bottom of the result screen that first appears when you select a data source for the report. The following dialog shows the initial result set from the sample data in the example above.



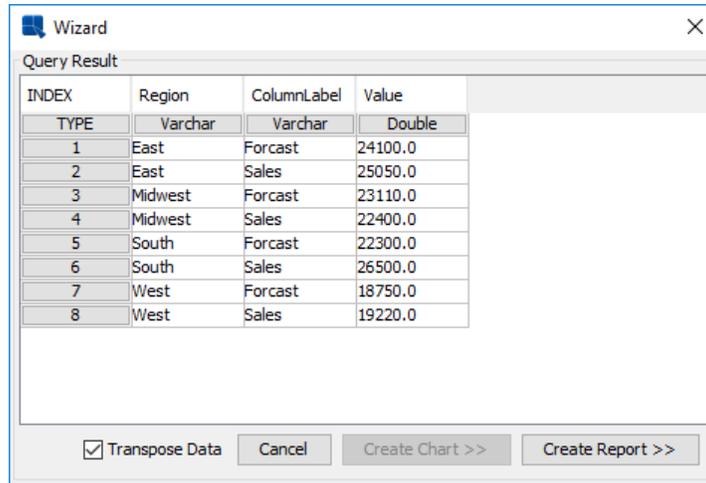
*Data Table Dialog*

When you check the option at the bottom, a new dialog will appear allowing you to select the columns that you want to transpose. In this example, you would select the `Forecast` and `Sales` columns. Note that in order to perform transposition, the selected columns must have the same data type.



*Select Transpose Dialog*

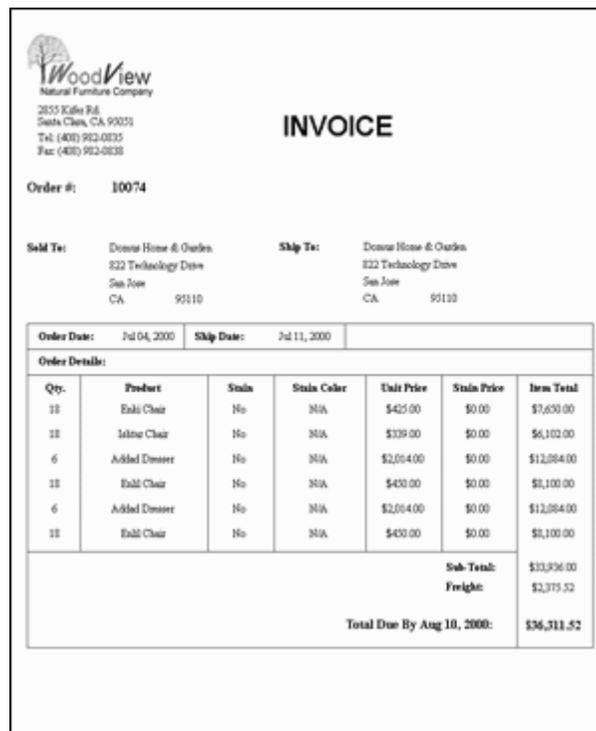
After making your selections, click the *OK* button to apply the changes. You will see the transposition in the data table dialog. The `Sales` and `Forecast` headers are transposed into a column called `ColumnLabel` and their values are merged into a column called `Value`. Now you can map this result set using a crosstab report where `ColumnLabel` is the Row Break, `Region` is the Column Break, and `Value` is the Column Break Value to create the final transposed report layout.



Data After Transposition

### 4.1.2.4. Master & Details Report

A Master & Details report is a set of tabular data that is grouped according to a master field. This report type is most commonly used when you have fields in your data table that have a one to many relationship. A good example of this is an invoice. For each order number in a database there will be customer information and several items with pricing information. A Master & Details report would be used to group the information according to order number.



Master & Details Report

#### 4.1.2.4.1. Data Mapping

The first step in data mapping for this type is to select the columns you want to include in your report from the *Set Mapping Window* in the Report Wizard. The left side of the window lists all available columns from your query or data file and the right side lists columns that will be displayed in the report. Click on a column name to select it (**Ctrl+click** for multiple selections) and press the *ADD* or *REMOVE* buttons to add it or remove it from the report. The order in which you select columns in this window is the order in which they will appear in your report.

The second step in data mapping is to set the column options. There are three column options available for Master & Details reports. There is a drop-down menu at the bottom of the column options window. It is labeled `primary key`. The drop-down menu contains all of the columns that you have selected for the report. Selecting a column as the `primary key` will group the report according to that column. A new group will be created every time the value in the selected column changes.

The top of the column options window displays a table showing all of the columns that you have selected for your report and the available options. The first field displays column names, the second one displays data type, and the third one displays a checkbox marked *Visible*. Checking or un-checking this box will make the column visible or invisible within the body of the report. The fourth field is the `Master field` field. Selecting a column as a Master field will place the column value in the column header instead of the data section of the report.

A checkbox at the bottom of the data mapping window allows you to change the layout of the report. Checking the *Side-By-Side Layout* box will arrange the report so that the master section is displayed next to the details section rather than above it.

For example, suppose we have the following data table:

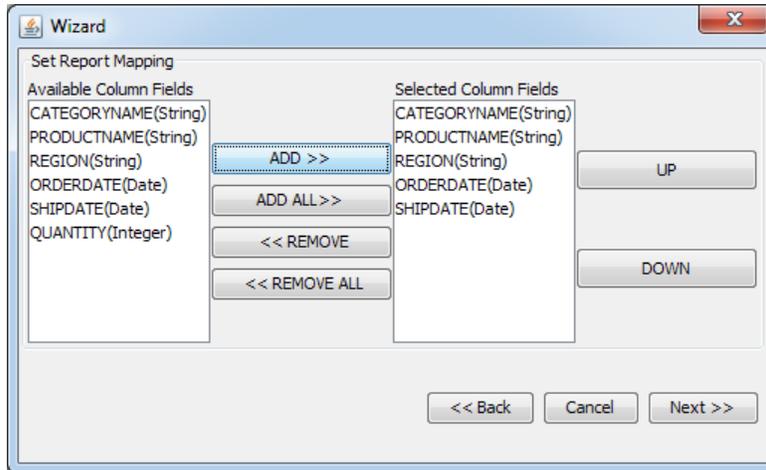
Order #	Customer Name	Product	Unit Price	Quantity
12	Paul Campbell	Chair	\$24.95	4
12	Paul Campbell	Table	\$127.50	1
14	Sally Hayes	Cabinet	\$227.25	2
14	Sally Hayes	Chair	\$24.95	2
14	Sally Hayes	Table	\$127.50	1

Setting `Order #` as the **primary key** and `Customer Name` as a **Master field** without using side-by-side layout will result in the following report:

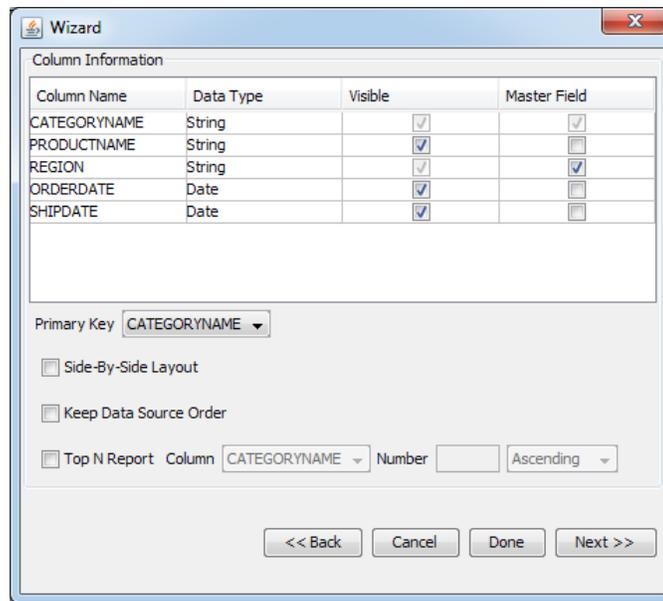
<b>Order #</b>	<b>12</b>	
<b>Customer Name</b>	<b>Paul Campbell</b>	
<b>Product</b>	<b>Unit Price</b>	<b>Quantity</b>
Chair	\$24.95	4
Table	\$127.50	1
<b>Order #</b>	<b>14</b>	
<b>Customer Name</b>	<b>Sally Hayes</b>	
<b>Product</b>	<b>Unit Price</b>	<b>Quantity</b>
Cabinet	\$227.25	2
Chair	\$24.95	2
Table	\$127.50	1

However, the same report with side-by-side layout would look like this:

		<b>Product</b>	<b>UnitPrice</b>	<b>Quantity</b>
<b>Order #:</b>	<b>12</b>	Chair	\$24.95	4
<b>Customer Name:</b>	<b>Paul Campbell</b>	Table	\$127.50	1
<b>Order#:</b>	<b>14</b>	Cabinet	\$227.25	2
<b>Customer Name:</b>	<b>Sally Hayes</b>	Chair	\$24.95	2
		Table	\$127.50	1



*Master & Details Report Mapping*



*Master & Details Report Mapping*

If you want to keep the data ordering specified in the data source, select the *Keep Data Source Order* option. To learn more about this feature, see Section 4.1.2.9 - Keep Data Source Order.

#### 4.1.2.4.1.1. Top N Report

You can also create a Top N report using the Master & Details format. A Top N report will order and show you the highest set of values based on a particular column in the report. For Master & Details reports you can also specify to show the highest values for each group. For example, you may want to show the top five customers in each region based on total sales. To do this, check the *Top N Report* checkbox at the bottom of the data mapping window. You can then specify which column you want to use as the measure (for the above example it would be total sales column), the number that you want to retrieve (i.e. 10 or 20, etc), and whether you want to display the columns in ascending or descending order. The master & details report will then return the number of records specified (or number of records specified for each row break) sorted by the highest value for the specified column.

#### 4.1.2.5. Mailing Label Report

A mailing label report is similar to a simple columnar report; however, it pre-arranges data in a way that allows you to create mailing labels. Data is arranged within the data table vertically and is wrapped automatically, making it easy to create a report showing an address list for mailing labels.



**Customer Addresses**

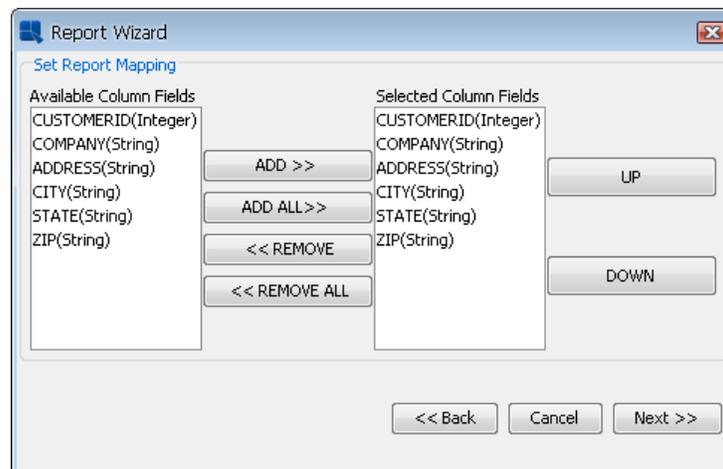
Malinda Gladwiler Allied Furniture Emporium 384 Broad St Lilbourn NY 18322	Lillian Antolini Specialty Retail 88 Spenser Drive Los Angeles CA 88938	Gary Cooper Du Monde Furnishings 8300 Cleveland St Ankerdown FL 29183
Frances Polk Eastern Treasures 123 Summer Rd Piterson NJ 09892	Arthur Childs All Unfinished Furniture 12 Woodpark Ave Cleveland OH 32343	Ernest Morrow George Park Imports 300 Great Swamp Rd George Park PA 23445
Jackie Benson Benson Imports 23 Tassaga Ave Aachen TX 03887	Ed Banley Les Chase 77 Dropuz Blvd Little Rock AK 12343	Herb Gale Gale Distributors 345 Seagen Drive Boontown NY 11234
Sally Hayes Woodworks Furniture 88 Rio Grand Ave Aerpen PA 38923	Eddie Bedsett Furniture Gallery 90 First Ave New York NY 11223	
Phoebe Caulfield Savilla Home & Garden 389 Jardim Rd Savilla AK 82938	Mal Grossard Domus Home & Garden 822 Technology Drive San Jose CA 95110	
Seymour Glass Imports & Leather Gallery 5 Baker's Lane Chicago IL 39283	Paul Campbell Campbell Interiors 839 Via Oady Rd Long Island NY 12334	
Muriel Fedder Furniture Palace 1230 Hoos Lane Piscataway NJ 08954	James Castle Furniture Pros, Inc 44 Islander Lane Orlando FL 83823	

*Mailing Label Report*

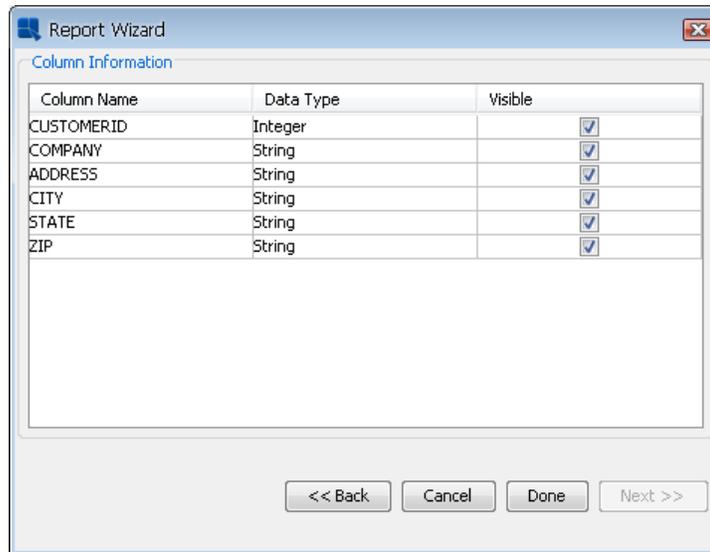
#### 4.1.2.5.1. Data Mapping

Data Mapping for this type is exactly the same as for simple columnar report. The first step is to select the columns you want to include in your report from the *Set Report Mapping* window in the Report Wizard. The left side of the window lists all available columns from your query or data file and the right side lists columns that will be displayed in the report. Click on a column name to select it (**SHIFT+Click** for multiple selections) and press the *ADD* or *REMOVE* buttons to add it or remove it from the report. The order in which you select columns in this window is the order in which they will appear in your report.

The second step for data mapping is to set the column options. For mailing label reports there is only one available option. The window displays a table showing all of the columns you have selected for the report and all available options. The first field displays column names, the second one displays data type, and the third one displays a checkbox marked *visible*. Checking or un-checking this box will make the column visible or invisible within the body of the report (all columns are visible by default).



*Mailing Label Report Mapping*

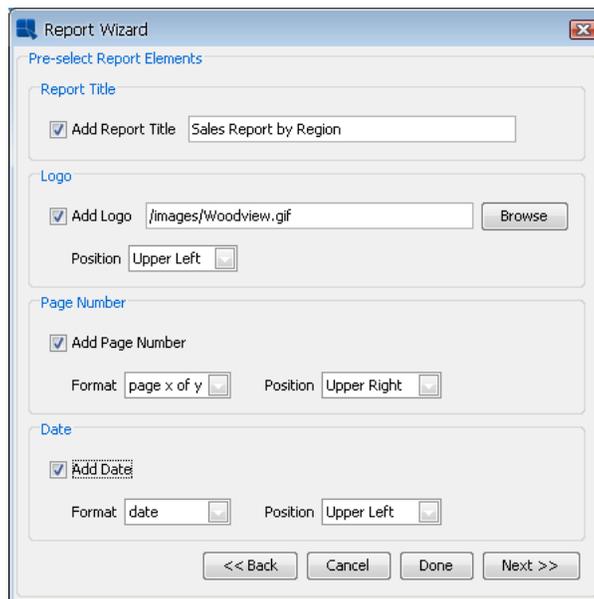


*Mailing Label Report Mapping*

### 4.1.2.6. Additional Formatting Options

After you finish specifying mapping options, you can dismiss the Wizard and begin editing/modifying the report. To exit the Wizard, click the *Done* button in the last data mapping window. This will take you back to the main designer window where a blank (unformatted) report will be generated based on your mapping specifications.

Instead of generating an unformatted report, ReportDesigner also provides several additional options in the Wizard that allows you to automatically place some elements in the report, as well as provide a default style for the report elements. To continue on in the Wizard, click the *Next* button in the last data mapping window. This will bring up a dialog prompting you to add several elements to the report.

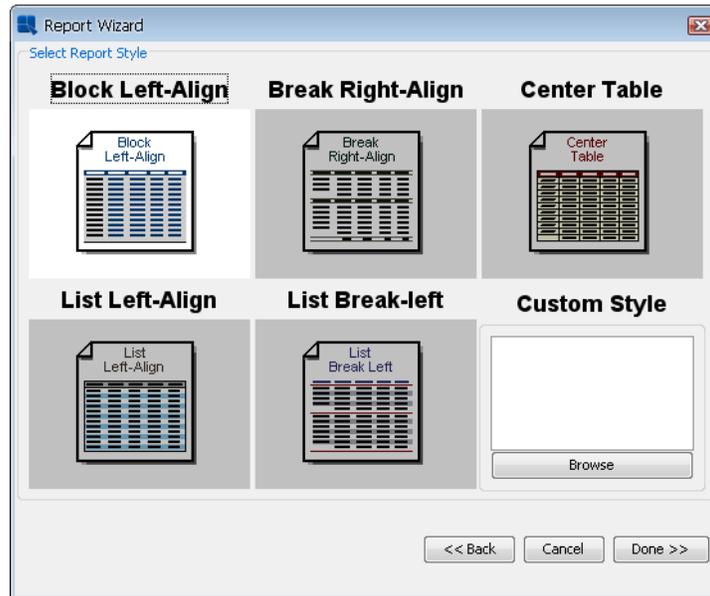


*Add Elements Dialog*

This dialog allows you to specify a report title and a logo for the report header. If you select to add a report title, you will be able to type in the title text. If you specify to add a logo, you can specify the location of the image file either using a file or URL path. You can also specify whether to place the logo in the upper right or upper left corner of the page.

You can also specify to add page numbers and date to the page headers and footers. You can select format and position of the elements either in the page header or footer and align them to the right, left, or center of the page.

After you finish specifying the elements, you can again click the *Done* button to dismiss the Wizard and begin editing/modifying the report with the added elements. If you click *Next*, you will be taken back to the last dialog in the Report Wizard. This dialog allows you to select a style for the report.



*Report Style Dialog*

From here you can pick a pre-defined style or a custom style you want to use for the report. Selecting a style will apply formats to the report elements, as well as change the default attributes of new report elements. Below are examples of each of the pre-defined report styles when applied to the same summary break report:

**Block Left-Align**

Order ID	Product Name	Unit Price	Stain Price	Stain	Quantity
<b>10 011</b>	Ishtar Chair	339	24	False	18
	Enlil Chair	450	27	False	18
	Enki Chair	425	24	False	18
<b>10 023</b>	Ishtar Chair	339	24	False	14
	Ninurta Chair	345	19	False	18
<b>10 057</b>	Ishtar Chair	339	24	False	9
	Het Table	1 255	140	False	12
	Nun Dresser	1 548	414	False	19
<b>10 074</b>	Ishtar Chair	339	24	False	18
	Enlil Chair	450	27	False	18
	Enki Chair	425	24	False	18
	Addad Dresser	2 014	487	False	6

*Block Left-Align*

This style sets text alignment to the left and draws the report on a gray background with blue headers.

**Break Right-Align**

Order ID	Product Name	Unit Price	Stain Price	Stain	Quantity
10 011	Ishtar Chair	339	24	False	18
	Enlil Chair	450	27	False	18
	Enki Chair	425	24	False	18
Order ID	Product Name	Unit Price	Stain Price	Stain	Quantity
10 023	Ishtar Chair	339	24	False	14
	Ninurta Chair	345	19	False	18
Order ID	Product Name	Unit Price	Stain Price	Stain	Quantity
10 057	Ishtar Chair	339	24	False	9
	Het Table	1 255	140	False	12
	Nun Dresser	1 548	414	False	19
Order ID	Product Name	Unit Price	Stain Price	Stain	Quantity
10 074	Ishtar Chair	339	24	False	18
	Enlil Chair	450	27	False	18
	Enki Chair	425	24	False	18
	Addad Dresser	2 014	487	False	6

*Break Right-Align*

This style sets text alignment to the right and draws lines to demarcate the headers.

**Center Table**

Order ID	Product Name	Unit Price	Stain Price	Stain	Quantity
<b>10 011</b>	<b>Ishtar Chair</b>	<b>339</b>	<b>24</b>	<b>False</b>	<b>18</b>
	<b>Enlil Chair</b>	<b>450</b>	<b>27</b>	<b>False</b>	<b>18</b>
	<b>Enki Chair</b>	<b>425</b>	<b>24</b>	<b>False</b>	<b>18</b>
<b>10 023</b>	<b>Ishtar Chair</b>	<b>339</b>	<b>24</b>	<b>False</b>	<b>14</b>
	<b>Ninurta Chair</b>	<b>345</b>	<b>19</b>	<b>False</b>	<b>18</b>
<b>10 057</b>	<b>Ishtar Chair</b>	<b>339</b>	<b>24</b>	<b>False</b>	<b>9</b>
	<b>Het Table</b>	<b>1 255</b>	<b>140</b>	<b>False</b>	<b>12</b>
	<b>Nun Dresser</b>	<b>1 548</b>	<b>414</b>	<b>False</b>	<b>19</b>
<b>10 074</b>	<b>Ishtar Chair</b>	<b>339</b>	<b>24</b>	<b>False</b>	<b>18</b>
	<b>Enlil Chair</b>	<b>450</b>	<b>27</b>	<b>False</b>	<b>18</b>
	<b>Enki Chair</b>	<b>425</b>	<b>24</b>	<b>False</b>	<b>18</b>
	<b>Addad Dresser</b>	<b>2 014</b>	<b>487</b>	<b>False</b>	<b>6</b>
<b>10 090</b>	<b>Ishtar Chair</b>	<b>339</b>	<b>24</b>	<b>False</b>	<b>17</b>
	<b>Het Table</b>	<b>1 255</b>	<b>140</b>	<b>False</b>	<b>6</b>
<b>10 016</b>	<b>Shamash Chair</b>	<b>449</b>	<b>26</b>	<b>False</b>	<b>4</b>

*Center Table*

This style centers text and draws table borders around the report cells.

**List Left-Align**

Order ID	Product Name	Unit Price	Stain Price	Stain	Quantity
10 011	Ishtar Chair	339	24	False	18
	Enlil Chair	450	27	False	18
	Enki Chair	425	24	False	18
10 023	Ishtar Chair	339	24	False	14
	Ninurta Chair	345	19	False	18
10 057	Ishtar Chair	339	24	False	9
	Het Table	1 255	140	False	12
	Nun Dresser	1 548	414	False	19

*List Left-Align*

This style sets text alignment to the left and draws alternating color for each row.

**List Break-Left**

Order ID	Product Name	Unit Price	Stain Price	Stain	Quantity
10 011	Ishtar Chair	339	24	False	18
	Enlil Chair	450	27	False	18
	Enki Chair	425	24	False	18
10 023	Ishtar Chair	339	24	False	14
	Ninurta Chair	345	19	False	18
10 057	Ishtar Chair	339	24	False	9
	Het Table	1 255	140	False	12
	Nun Dresser	1 548	414	False	19
10 074	Ishtar Chair	339	24	False	18
	Enlil Chair	450	27	False	18
	Enki Chair	425	24	False	18
	Addad Dresser	2 014	487	False	6

*List Break-Left*

This style sets text alignment to the left, adds lines to demarcate column headers, and draws alternating color for each row.

Report styles will only effect the appearance properties of the report elements and will not change the report type or data mapping options that were selected earlier. Once you specify a style, click the *Done* button to dismiss the Wizard and continue to the main designer window.

#### 4.1.2.6.1. Custom Styles

In addition to the five pre-defined styles included with ReportDesigner, users can create their own style definitions. Style definitions are special versions of report templates that are saved with a `.stl` extension. Users can create a style definition using any report template.

#### 4.1.2.6.1.1. Creating a Custom Style

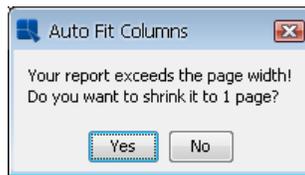
The first step in creating a custom style is to set the global formats. This allows you to control the default look and feel for all elements the user inserts into the report. For more information about global formats, see Section 4.1.3.9.1 - Global Formatting.

The second step in creating a custom style is to select the look and feel for the elements in each section of the report. To do this, select the report element whose formats you want to apply to all elements in the section when the style is applied and select *Set Attributes as Section Style* from the pop-up menu.

Once you finish setting the attributes, save the report template as a custom style by checking the *Create Style* option in the save as dialog.

#### 4.1.2.7. Fit Columns to Page Width

Often when you finish with the Report Wizard, your created report will be wider than the default page width (8.5"). When this happens, you will be presented with a warning.

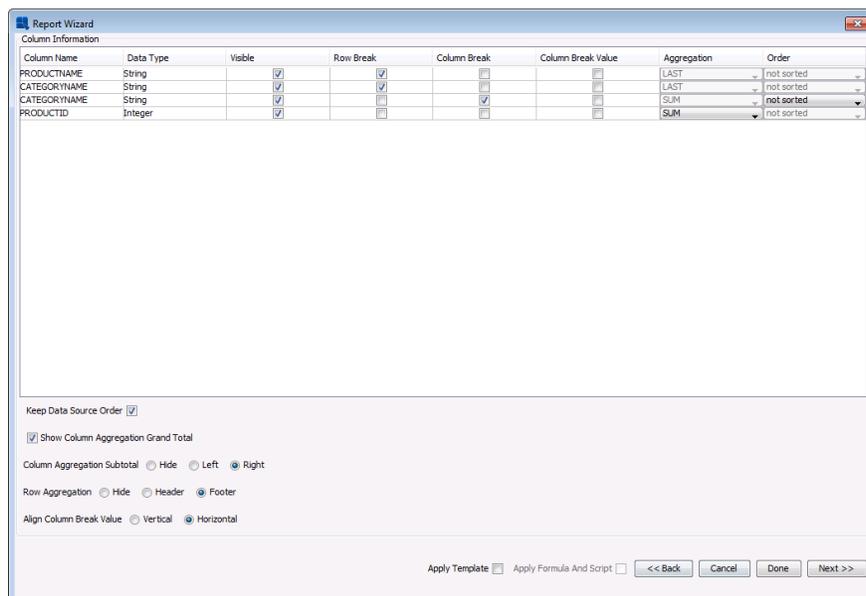


*Auto-Fit Columns Dialog*

The dialog gives you an option to shrink the report to fit within the page. If you select *Yes*, all columns will automatically shrink to fit the width of the page. The page boundary is indicated by a vertical bar within the design window. If you select to keep the columns overlapping the page boundary, the overflow will be printed on a new page when you preview the report. You can auto-fit the columns anytime you are editing the report by selecting *Auto Fit Columns* from the Format menu. The page width can also be adjusted by selecting *Page Setup* from the Option menu.

#### 4.1.2.8. Change Data Mapping

Once you complete the data mapping, a rough version of the report will be generated. If it is not correct, you can change the data mapping by selecting *Change Data Mapping* from the *Data* menu, or clicking the *Change Data Mapping* button on the toolbar. This will take you back to the Report Wizard (starting with data mapping), allowing you to go back to change the report type and data mapping.

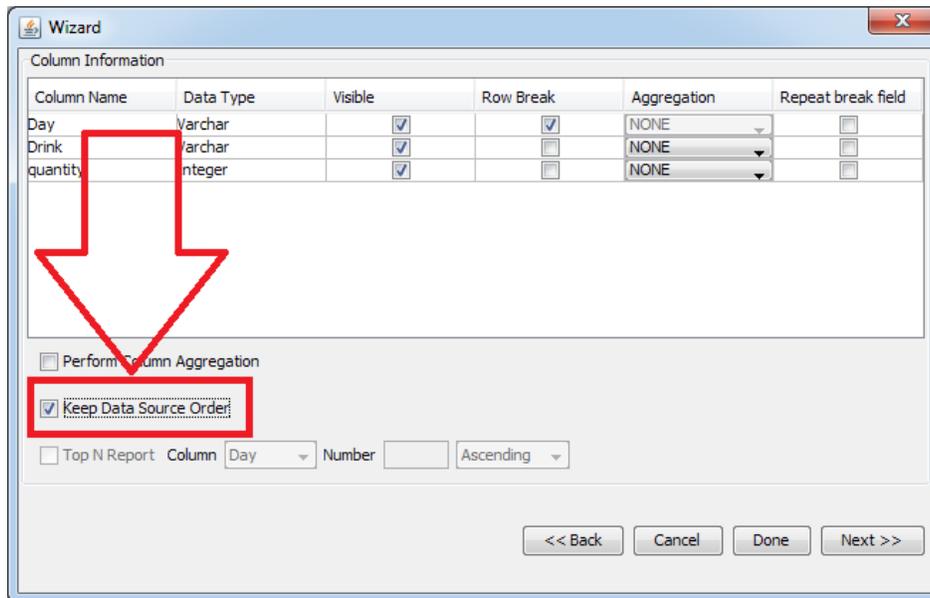


*Change Data Mapping window*

When you change the data mapping, there will be two checkboxes at the bottom of the last screen in the Report Wizard. One is marked *Apply Template* and the other *Apply Formula and Script*. Checking the first box will retain the formatting of your report; however, the labels and formulas may not be correct. Un-checking this box will create a blank report with the new data mapping. If you did not format lot of the report objects, it is recommended that you un-check this box. This will ensure that the new data mapping is completely accurate. The second box allows you to include functions and scripts placed in the data table section to be applied (formulas and scripts in other sections are always applied), when you select the *Apply Template* option. Please note that if you have changed the number of columns or the data types of certain columns, the functions and/or scripts may no longer work.

#### 4.1.2.9. Keep Data Source Order

Summary Break, Cross-tab and Master&Details reports use memory-optimized query processing which can unfortunately change the order of the data. In other words: the data in the report may be displayed in different order than the data in the data source. To prevent this from happening, choose the *Keep Data Source Order* option in the report mapping dialog. This will disable the memory-optimized query processing which means that the data will be passed to the report without any optimization (i.e. the original data ordering will be preserved).



*Keep Data Source Order option*

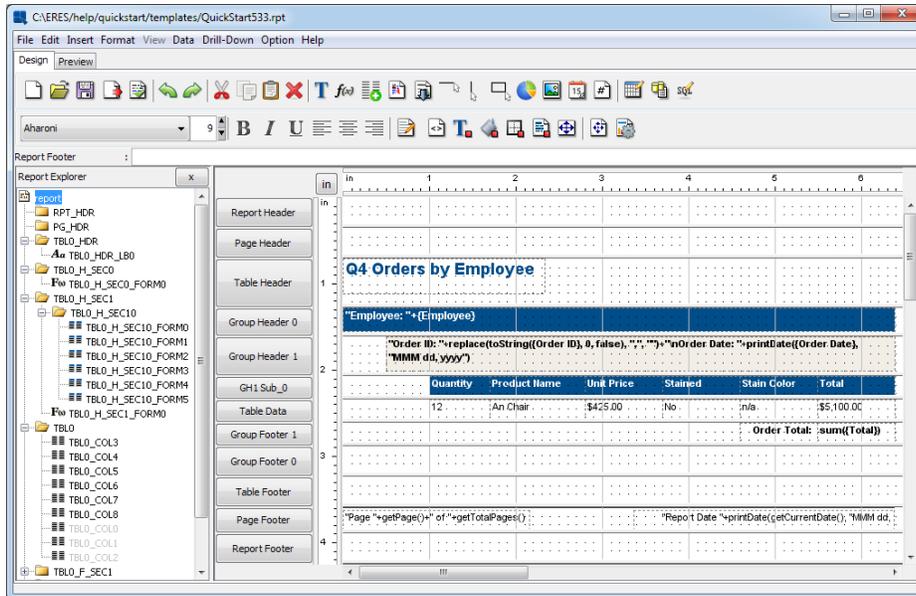


#### Note

If you choose the *Keep Data Source Order* option, the *Top N Report* option for Master&Details and Summary Break reports will be disabled.

#### 4.1.3. The Designer Interface

Users who have some familiarity with other report writers will recognize the banded-style interface of the Report Designer. Report elements can be added and modified within the Designer window and then the resulting report can be previewed in the preview window.



*Report Designer Interface*

### 4.1.3.1. Report Sections

Each group within the designer interface represents a different report section. The section names are listed on the buttons to the left of each report group.

**Report Header:** This section is like the title of the report. It appears only once at the top of the report.

**Page Header:** Any element placed in this section will appear at the top of every page of the report.

**Table Header:** This section serves as a header to the detail or data section of the report. By default, it only appears once in the report.

**Group Header:** This section appears in reports with grouped data (i.e. master & details report), or data with row breaks inserted (i.e. summary break report). It repeats at the top of each grouping within the report.

**Table Data:** This is the main section of the report that contains most of the data. Data columns that have been selected for the report are placed in this section and are repeated for each entry in the column.

**Group Footer:** This section appears on reports with grouped data (i.e. Master & Details report) or data with row breaks inserted (i.e. summary break report). It repeats at the bottom of each grouping within the report.

**Table Footer:** This section serves as the footer or data section of the report. By default, it appears only once in the report.

**Page Footer:** Any element placed in this section will appear at the bottom of every page of the report.

**Report Footer:** This is the last summary of footer section of the report. It only appears once at the end of the report.

#### 4.1.3.1.1. Nested Sections

There can also be nested sections in addition to the Report Header, Table Header, Group Header, Group Footer, Table Footer, and Report Footer sections. Nested sections are useful for placing sub-reports and rich text fields with variable lengths. This allows these type of elements to resize without overlapping any elements below. Nested sections inherit most of the section options from their parent sections, including page-breaking and repeating options.

To add a nested section, select *Insert Section* from the section options pop-up menu for the parent section. For more information about section options, options see Section 4.1.3.3 - Section Options. Note that if you select to insert a section from a nested section, it will inherit the same parent section (i.e. nested sections cannot be a parent).

### 4.1.3.2. Resizing Sections

To resize section height, place your mouse over the section divider within the design window. The pointer will change to a resize pointer. Click and drag the mouse to expand or decrease the section height.

### 4.1.3.3. Section Options

There are several formatting and display options available for each report section. You can access the section options menu by clicking the button with the section name at the left side of the Designer or by right clicking on a blank portion of the section field.

**Background Color:**

This option is available for every section of the report. It allows you to set background color for the entire section. Selecting this option will bring up a dialog box prompting you to specify whether or not to set the background transparent. If you do not want a transparent background, uncheck the checkbox and click the button. This will bring up a dialog prompting you to specify the background color. You can select font color from swatches or enter HSB/RGB values. You can also directly enter a HEX color value to the *Hex #* field.



**Note**

If there are no elements placed in a section, that section will not display. Hence, changing a section background color will have no effect on the finished report if the section is blank.



*Background Color Dialog*

**Section Height:**

This option is available for every section of the report. It allows you to set the section height. It has the same effect as dragging the section's bottom edge, but unlike the drag&drop method, this option allows you to set the section height precisely in inches.

**Insert Section:**

This option is available for Report Header, Table Header, Group Header, Group Footer, Table Footer, and Report Footer sections. Selecting this option will add a nested section to the currently selected report section.

**Remove Section:**

This option is only available for nested sections that you have added to the report (regular sections can only be rendered invisible). This option will remove the nested section from the report and delete its contents.

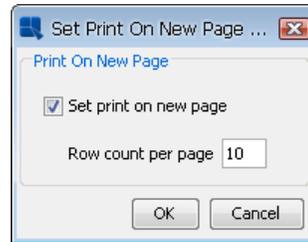
**Repeat On Every Page:**

This option is available for Table Header and Group Header sections. Selecting this option causes the section to repeat again after every page break (group headers will only repeat if the data for that group exceeds one page).

**Print On a New Page:**

This option is available for Table Header, Group Header, Table Data, Group Footer, Table Footer, and Report Footer sections. For the Table Data section,

a new dialog will pop-up allowing you to select the number of rows to display on each page. This number will be set as a new maximum of table data rows for the page. After this amount of rows a page break will be inserted. This count does not include any formulas in the footer sections. For example, if the report contained row breaks and an aggregation calculated for each group, this element will not count towards the number of rows.



*Print on New Page Dialog*

Selecting this option for any other section inserts a page break each time the section appears.

**New Excel Sheet:**

This option is only available for Group Header section(s). Selecting this option causes each group to be printed on a separate worksheet when the report is exported to Excel (XLS) or Excel 2007 (XLSX) format.

**Reset Page Number:**

This option is only available for Group Header section(s). Selecting this option causes the report pagination to reset each time the section is run. Hence, each time the Group Header section runs, the current page number goes to 1. Note that setting this option will automatically enable the print on a new page option.

**Fit Group On Page:**

This option is only available for Group Header section(s). It allows you to control some of the pagination for grouped data. Selecting this option will cause ReportDesigner to try to fit the entire group on a page. If the entire group cannot print on a page, it will print on a new page. If the group cannot fit on any page, (the fields are longer than the page height), it will be broken up into multiple pages.

**Skip First Value:**

This option is available for Table Header, Group Header, Group Footer, and Table Footer sections. This option is a subset of the print on a new page option. Selecting this option will ignore the page break for the very first time a section is run. For example, setting this option for the Group Header section will cause the report not to insert a page break for the first time the section is run. The page break will only occur when the report runs the Group Header for a second time.

**Skip First Group Value:**

This option is only available for Group Header section. This option is a subset of the print on a new page option. Selecting this option will ignore the page break for the first value of each group of data. This option is used for nested grouping in a report where more than one group is set to print on a new page. To prevent a blank page for records where more than one grouped column value changes, the 'Skip First Group Value' can be set for the inner Group Header section.

**Resize Cells Proportionally:**

This option only applies if you have set cells within the section to resize to fit content. If this option is selected, all cells in the section will resize with the variable height cells.

**Invisible:**

This option is available for every section of the report. Selecting this option hides the entire section. To make the section visible again, click on the *Format* menu at the top of the designer and select *Make Section Visible* option.

**HTML Border:**

This option is available for every section and only takes effect if the report is exported to HTML. Since tabular HTML (not DHTML CSS formatting) does not allow selective cell borders within a table, users can select to turn on or off a border for all cells in a section when the report is rendered in HTML. If you select this option, the following dialog will appear allowing you to set options for the HTML border:



*HTML Border Options*

In this dialog, you can set thickness and border color by clicking the *Border Color* button. This will open the color palette dialog allowing you to pick a color for the HTML border.

**Scripting:**

This option will bring up scripting interfaces allowing you to write, modify, or apply a script to dynamically modify certain section attributes. For more information about this feature, please see Section 4.1.7.2.7 - Section Scripts.

**4.1.3.4. Rulers**

The Report Designer has two rulers in the top left corner of the Design window. The rulers measure how the report will look in terms of the physical page dimensions. Please note that the dimensions do not include page margins. The rulers helps you to get a good idea of how the printed report will look like. You can also use the rulers to adjust the bounds of elements in your report. Selected element bounds will appear as shaded areas within the rulers and can be adjusted by clicking on and dragging the markers.



*Rulers and in/cm button*

**4.1.3.4.1. Toggling Between Inches and Centimeters**

The button in the upper left corner of the Design window allows you to toggle between inches and centimeters. By default, ReportDesigner uses inches for measurement. Depressing the button will change the ruler measures to centimeters. If centimeter is selected, all of the manually entered values like element bounds and page margins will also be entered and displayed in centimeters.

**4.1.3.5. Designer Menus**

Most of the Report Designer's features can be controlled using the drop-down menus at the top of the designer.

**4.1.3.5.1. File Menu**

This menu performs most of the file operations.

<b>New:</b>	Creates a new report.
<b>Open:</b>	Opens an existing report (.pak/.rpt/.xml/.qrp/.stl format).
<b>Close:</b>	Closes the current report.
<b>Apply Template:</b>	Imports and applies a template (.pak/.rpt/.qrp or .xml) file to the current report. All of the report attributes are changed to those of the template.
<b>Save:</b>	Saves the current report.
<b>Save As:</b>	Allows you to name and save the current report as a template (.pak) file. A checkbox within the <i>Save As</i> window allows you to generate an applet page with the Report Viewer embedded, as well as generate an XML template or a report style (.stl).
<b>Export:</b>	This option exports the report to a selected format. Options include DHTML, PDF, CSV (data file), Excel (XLS), Excel 2007 (XLSX), RTF, TXT, and XML.
<b>Print:</b>	Prints the current report (This option is only executable from Preview tab.)
<b>Exit:</b>	Closes the application.

#### 4.1.3.5.2. Edit Menu

This menu allows you to edit and cut/paste report elements.

<b>Undo:</b>	Cancels the last operation performed in the Designer and reverts back to the previous state. The Designer will remember last 10 actions made.
<b>Redo:</b>	Reverses the action of the undo command. For example, if you change the font color from black to red, you can undo this command to change it back to black and then redo this command to have it changed back to red again.
<b>Edit:</b>	This option is available for labels, formulas, images, and charts. For labels, it will prompt you to change the label text. For formulas, it will prompt you to change the formula. For images, it will prompt you to change the image file URL. For charts, it will bring up Chart Designer to edit your chart.
<b>Copy:</b>	This option makes a copy of the selected element and places it on the clipboard.
<b>Cut:</b>	This option removes the selected element and places it on the clipboard.
<b>Paste:</b>	This option pastes the current clipboard element into the report.
<b>Delete:</b>	This will delete the selected element, except for elements in the Table Data section. It will not delete these, but render them invisible.
<b>Remove Sub-Report:</b>	Removes a sub-report from the current report.

#### 4.1.3.5.3. Insert Menu

This menu allows you to insert elements into the current report.

<b>Insert Label:</b>	Inserts a label element into the current report.
<b>Insert Formula:</b>	Inserts a formula element into the current report.
<b>Insert Column Field:</b>	Inserts a column field into the report from the available fields (Available fields are those that have been selected for the report. See data mapping: Section 4.1.2 - Report Types and Data Mapping).

<b>Insert Database Field:</b>	This option is only available if the template's data source is a database. It allows you to insert a field from a database that is not part of the original query.
<b>Insert Column Header:</b>	Inserts a column header for one of the report fields.
<b>Insert Parameter Value:</b>	Inserts a formula that returns the supplied value for a report parameter.
<b>Insert Chart:</b>	Inserts a new chart into the current report.
<b>Insert Image:</b>	Inserts an image from a file into the report. Images can be in JPEG, GIF, or PNG format.
<b>Insert Rich Text Field:</b>	Inserts a new rich text into the current report.
<b>Insert Table of Contents:</b>	Inserts a table of contents into the report.
<b>Insert Line:</b>	Inserts a vertical or horizontal line into the report.
<b>Insert Rectangle:</b>	Inserts a grid rectangle.
<b>Insert Guideline:</b>	Inserts a guideline into the report. Guidelines can help you move and position elements in the report. For more information about guidelines, please see Section 4.1.3.7.14 - Guidelines.
<b>Today's Date:</b>	Inserts the current date into the report in one of three ways: <code>Date</code> , <code>Date &amp; Time</code> , or <code>Time</code> . The date function is actually a formula and the format can be adjusted after it is inserted. For more information about formula formatting, please see Section 4.1.3.7.3 - Data Formatting for Formulas and Column Fields.
<b>Page Number:</b>	Inserts the page number into the report in one of two ways: <code>Page Number</code> or <code>Page Number of Total Pages (i.e. page X of Y)</code> . The page number function is actually a formula and it can be formatted after it is inserted. For more information about formula formatting, please see Section 4.1.3.7.3 - Data Formatting for Formulas and Column Fields.
<b>Insert Sub-Report:</b>	Inserts a new or pre-existing sub-report into the current report. For more information about sub-reports, please see Section 4.1.9 - Sub-Reports.

#### 4.1.3.5.4. Format Menu

The format menu contains formatting options for the currently selected element within the report. Some menu options will not work depending on the type of element which is currently selected.

<b>Data Format:</b>	This option is available for formulas and column fields. It will bring up the data format editor for the data type of the selected element, either <code>string</code> , <code>numeric</code> , <code>date/time</code> , or <code>Boolean</code> .
<b>Chart Export Format:</b>	This option is only available for charts. When reports are exported to DHTML format, charts are converted to static images. This option allows you to specify the image properties for the chart upon export.
<b>Edit Attributes:</b>	This option is available for every report element. It will bring up a dialog box that allows you to adjust attributes depending on the selected element (e.g. bounds, background color, alignment, font style and size, rotation, border/round corners, data format).
<b>Font Style and Size:</b>	This option is available for labels, formulas, and column fields. It brings up a dialog box prompting you to change the font appearance and the size of the selected element.

<b>Background Color:</b>	This option is available for labels, formulas, and column fields. It allows you to adjust the background color of the selected element.
<b>Dual Colors:</b>	This option is only available for elements in the Table Data section of the report. It allows you to specify alternating background colors, font colors, and font styles for the selected column, as well as the number of rows between changes or on which column field to base the change.
<b>Line Color &amp; Thickness:</b>	This option is only available for lines. It allows you to set the color and the thickness for the selected line(s).
<b>Border:</b>	This option is available for every report element. It allows you to specify thickness (in pixels), and color and shape of corners (sharp or round) of the border drawn around the selected element. The border thickness/corners can be set individually for the each side/corner of the element.
<b>HyperLink:</b>	This option is available for all elements except column fields. It allows you to hyperlink an element to a web page or another report.
<b>Scripting:</b>	This option is available for all labels, formulas, and column fields. It allows you to create a script and apply conditional formatting to the selected element.
<b>Rotation:</b>	This option is available for all labels, formulas, and column fields. It allows you to rotate the element 90 degrees clockwise or counter-clockwise.
<b>Z-Index:</b>	This option allows you to set the z index for the selected report element. The z index controls how report elements behave (i.e. which one appears first) when placed on top of each other.
<b>Bounds:</b>	This option is available for every report element. This feature allows you to manually enter the boundaries of an element. The measurements will be in inches or centimeters depending on which unit is selected with the toggle button in the upper left corner of the Designer window where the rulers meet.
<b>Alignment:</b>	This option is available for labels, formulas, and column fields. It allows you to align the selected element horizontally to the left, center, right, as well as vertically to the top, middle, or bottom.
<b>Wordwrap:</b>	This option is available for labels, formulas, and column fields. It allows users to enable or disable word wrapping within report elements.
<b>Resize To Fit Content:</b>	This option is available for labels, formulas, and column fields. It causes the height of the selected element to adjust dynamically to fit its content.
<b>Make Column Visible:</b>	This option is only available if you deleted an element in the Table Data section, or if you selected an invisible data column from the column options screen of the Report Wizard. It allows you to render any invisible column visible.
<b>Make Section Visible:</b>	This option is only available if you rendered a report section invisible (See Section 4.1.3.3 - Section Options). It will make the section visible again.
<b>Invisible:</b>	This option is available for elements in the Table Data section or for report sections. It will make the selected element invisible.
<b>Column Wrap:</b>	This option allows column fields to be wrapped, so rather than breaking to the next page at the end of the page, they will continue next to the original column field within the current page.
<b>Auto Fit Columns:</b>	This option will shrink all elements within the report to fit the report within the page width.

**Swap Columns:** This option allows the position of two elements to be swapped in the report. Elements must first be selected using **CTRL+Click**.

#### 4.1.3.5.5. View Menu

The view menu contains options allowing you to navigate through the report in the Preview window and sort the data. It is only active when you are previewing the report.

**First Page:** This option will navigate to the first page of the report within the Preview window.

**Previous Page:** This option will scroll back one page within the Preview window.

**Next Page:** This option will scroll forward one page within the Preview window.

**Last Page:** This option will navigate to the last page of the report within the Preview window.

**Go To Page...:** This option will prompt you to enter a page number and then navigate to that page of the report within the Preview window.

**Sort by (ascend):** This option opens up a second menu allowing you to select one of the report columns to sort the report in ascending order (either numeric or alphabetical).

**Sort by (descend):** This option opens up a second menu allowing you to select one of the report columns to sort the report in descending order (either numeric or alphabetical).

**Sort by...:** This opens a dialog which allows you to select nested sorting in a report (by more than one column). The direction of the sorting can be set for each column.

**Launch Page Viewer:** This will load the entire report in the page viewer window. For more information about this feature, see Section 4.1.4.2.4 - Using Page Viewer.

#### 4.1.3.5.6. Data Menu

The data menu contains options that allow you to see and manipulate the data used in the current report and its properties.

**Refresh:** This option will re-query the data source used for the current report and then update the report.

**Change Data Mapping:** This will re-open the Report Wizard, allowing you to change data mapping for the current report.

**Change Data Source:** This will re-launch the Data Source Manager, allowing you to modify or change the report's data source.

**Modify Database:** If the report uses a database as the data source, this option allows you to modify the connection information that the report uses to connect to the database. For more information about this feature, please see Section 3.1.3.5 - Editing Database Connections.

**Modify Query:** If the report uses a database as the data source, this option allows you to modify the query used to retrieve the report data. For more information about this feature, please see Section 3.1.3.4 - Editing Queries.

**View Column Mapping:** This option will bring up a window that displays column mapping for the current report. The window displays the column index, name, data type, options, and whether it is currently visible or invisible.

**View Table:** This option will bring up a window containing the data table from which the current report is generated. The table will initially display only the first 20 records. Clicking on the *Show All Records* checkbox will display all of the records in the data table.

---

<b>View Data Source Info:</b>	This option will bring up a window containing information about the data source that was used to create the template. The data source type, location, and the data registry location will be displayed.
<b>Sub-Report Parameter Sharing:</b>	This allows you to link parameters defined within sub-reports to parameters in the main report or to the parameters defined in other sub-report. With this feature enabled, reports can share a user supplied parameter value automatically without having to pass the value to each sub-report. For more information about sub-report, see Section 4.1.9 - Sub-Reports.
<b>Sub-Report Parameter Mapping:</b>	This allows you to link parameterized sub-reports to column fields in the primary report. For more information about sub-reports, see Section 4.1.9 - Sub-Reports.
<b>Chart Parameter Mapping:</b>	This allows you to link parameterized charts (with an independent data source) to column fields in the report. For more information about this features, see Section 4.2.1.3.1 - Chart Parameter Linking.
<b>Select Multiple Drill-Down Values:</b>	This option is only available when you preview a multi-value drill-down report. It allows you to select values to use when drilling to the next level. For more information about drill-down reports, please see Section 4.1.8 - Drill-Down.
<b>Security:</b>	This will bring up the security settings dialog for the currently selected cell. For more information about the template security, please see Section 4.1.10 - Template Security.
<b>Cell Properties:</b>	This option is available for all elements in the report. It will bring up a window displaying the properties of the selected element including its ID value, as well as column index, name, and data type for column fields.
<b>Set Preview Security Level:</b>	This option allows you to specify which defined security level you want to use when previewing the template. For more information about the template security, please see Section 4.1.10 - Template Security.
<b>Set Preview Data Options:</b>	This option allows you to turn on/off live data for the preview window as well as set the maximum number of records that should be returned when previewing the report.
<b>Secured Query Parameters:</b>	This will bring up the parameter security dialog, allowing you to assign parameter values to security levels. For more information about the template security, please see Section 4.1.10 - Template Security.
<b>Preview Parameter Prompt:</b>	This option allows you to enable/disable parameter prompting when you preview the report.

#### 4.1.3.5.7. Drill-Down Menu

The drill-down menu contains options that allow you to add, remove, and navigate layers of drill-down within the report. For more information about drill-downs, please see Section 4.1.8 - Drill-Down.

<b>Navigate:</b>	This option brings up the drill-down navigation window allowing you to edit, add, and remove layers of drill-down.
<b>Drill-Down Link:</b>	This option allows you to link the currently selected element to a drill-down layer.

#### 4.1.3.5.8. Option Menu

This menu contains display and printing options

<b>Report Explorer:</b>	This option turns on/off the report explorer display. For more information about the explorer interface, see Section 4.1.3.8 - The Report Explorer.
-------------------------	---

<b>Page Setup:</b>	This option allows you to customize the page size and margins. The measurements will be in inches or centimeters depending on which unit is selected with the toggle button in upper left corner of the designer window where the rulers meet.
<b>HTML Page Title:</b>	This option allows you to specify the page title that is generated when the report is exported to DHTML format.
<b>Background Color:</b>	This option allows you to set the page background color for the report. Specifying a color here will color the entire page in Report Viewer and when exporting to DHTML or PDF. Any non-transparent section color or report element background color will draw over the background color.
<b>Background Image:</b>	This option allows you to specify a background image for the report.
<b>Snap to grid:</b>	This option allows you to control the settings for the snap to grid feature. You can turn the feature on/off, as well as control the step size.
<b>Font Mapping:</b>	This allows you to map system (true type) font files for the PDF export. For more information about this feature, please see Section 4.1.5.2.1 - PDF Font Mapping.
<b>Viewer Font Setup:</b>	This option allows the font size for text in the Report Designer and Report Viewer to be relative to the screen resolution. Turned on by default, this feature allows for more precise conversions of reports to various export formats and between installations.
<b>Export Style Sheet:</b>	This option allows you to export the style sheet (.css) file with the style definitions for the elements in the report. This style sheet file can be used to apply the DHTML export settings onto other reports. For more information about working with CSS, please see Section 4.1.5.2.2 - CSS Options.
<b>Show Cell Outline:</b>	This option draws the outline for all the elements in the report. It allows you to see the boundaries of all elements, as well as see elements where the text may not be visible.
<b>Shift On/Off:</b>	This option allows you to turn the cell shifting mode on or off. If the shift mode is on, resizing and moving report elements will cause the surrounding elements to shift: accommodating the resized or repositioned element.
<b>Show Formula Name:</b>	This option allows you to enable/disable displaying the function name, rather than the text of the function in the Design window. For more information about working with formulas, see Section 4.1.6 - Using Formulas & the Formula Builder.
<b>Enable Table of Contents:</b>	This option allows you to show/hide a table of contents (if it is inserted).
<b>Global Format:</b>	This option allows you to set formatting for each of the different type of report elements. It will apply changes to elements currently in the report, as well as change the default properties.
<b>Null Data Handler:</b>	This option allows you to specify the treatment of null data within the report.

#### 4.1.3.5.9. Help Menu

This menu allows you to view program version and open the User's Guide.

**About:** This option shows you information about the program version.

**Contents:** This option opens the EspressoReport ES User's Guide.

### 4.1.3.6. Designer Toolbar

The toolbar at the top of the designer offers easy access to ReportDesigner's most commonly used features. The first row of buttons perform the following functions:

-  Start a new report
-  Open an existing report
-  Save the current report
-  Export the current report
-  Apply a template to the current report
-  Undo the last change
-  Redo the last undone change
-  Cut the selected element and place it on the clipboard
-  Copy the selected element
-  Paste the current clipboard object into the report
-  Delete the selected element
-  Insert a label
-  Insert a formula
-  Insert a column field
-  Insert a rich text field
-  Insert a sub-report
-  Insert a horizontal line
-  Insert a vertical line
-  Insert a grid rectangle

-  Insert a chart
-  Insert an image
-  Insert the current date
-  Insert page number
-  Change Data Mapping
-  Change Data Source
-  Edit Query

The first two boxes in the second row of the toolbar allows you to select font face and size for the currently selected element. Buttons in the second row perform the following functions:

- B** Set the font style to bold
- I* Set the font style to italic
- U Underline the text
-  Set the horizontal alignment to left
-  Set the horizontal alignment to center
-  Set the horizontal alignment to right
-  Edit the selected element
-  Set data format for the current element
-  Apply a script to the current element
- T** Set font style, size and color for the current element
-  Set background color for the current element
-  Set border thickness, color and corner rounding for the current element
-  Set dual colors for the current element



Set bounds for the current element



Page setup



Limit preview display rows

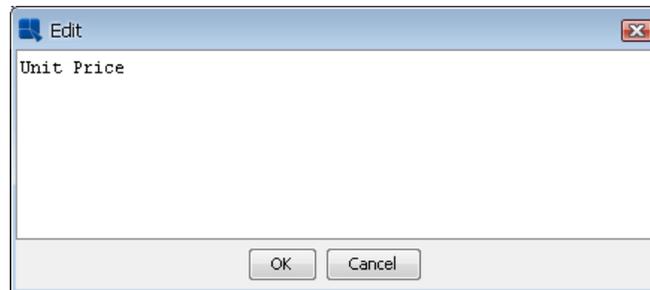
### 4.1.3.7. Inserting and Manipulating Report Elements

When you finish with the Report Wizard, a rough version of your report will be generated based on the mapping and it will display the options you selected. To polish the report, you may want to insert new report elements, as well as format the existing ones.

#### 4.1.3.7.1. Inserting Elements

##### Labels:

You can insert a label in one of two ways: by selecting *Insert Label* from the *Insert* menu or by clicking the  *Insert Label* button on the toolbar. After you select the *Insert Label* option, a small box will follow your mouse pointer around the Design window. Position the box where you want to insert the label and click. A dialog box will appear prompting you to enter the label text. Click the *OK* button and the label will appear in your report.



*Insert Label Dialog*

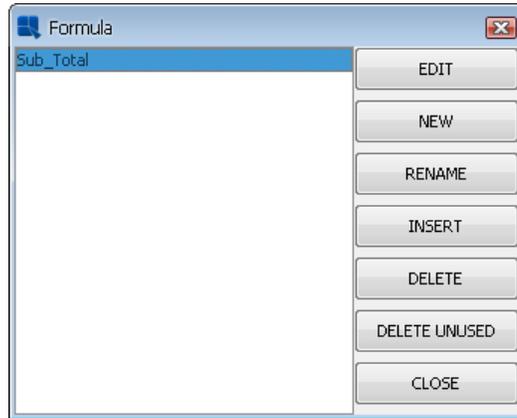
##### Formulas:

You can insert a formula in one of two ways: by selecting *Insert Formula* from the *Insert* menu or by clicking the  *Insert Formula* button on the toolbar. After you select the *Insert Formula* option, a dialog box will open allowing to select a formula to insert. To insert a formula, select a formula from the list and click the *Insert* button. A small box will follow your mouse pointer around the Design window. Position the box where you want to insert the formula and click. The formula will be added. You can use an existing formula or create a new one. To create a new formula, click the *New* button in the formula list dialog. After entering a name for the new formula, the Formula Builder window will open allowing you to construct the formula (See Section 4.1.6 - Using Formulas & the Formula Builder for more information about using the formula builder and formula syntax). Any formula is automatically anchored to the report section in which it is inserted. This means that the formula will reset each time the section repeats.

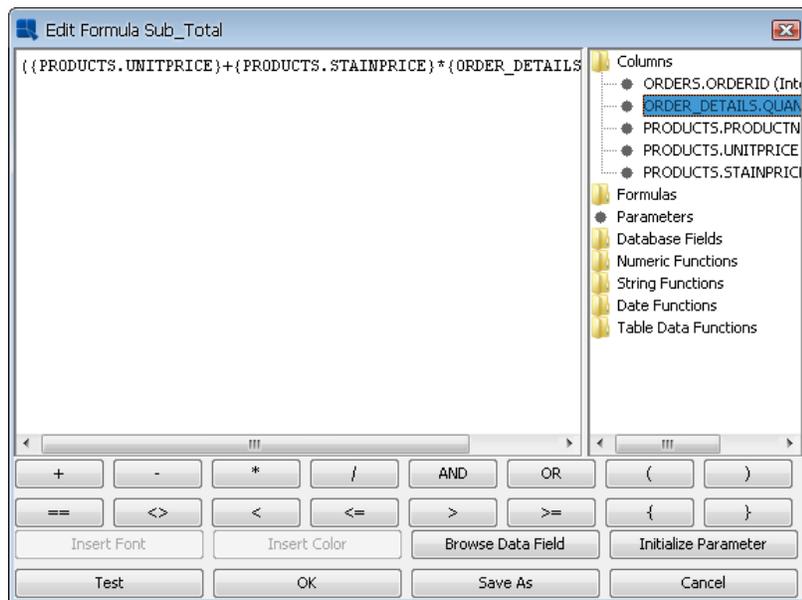


#### Note

Only the text of the formula will appear in the design window (unless the formula is inserted in the Table Data section). You can choose to display the formula names instead of the text by selecting *Show Formula Name* from the Option menu.



Formula List



Formula Editor

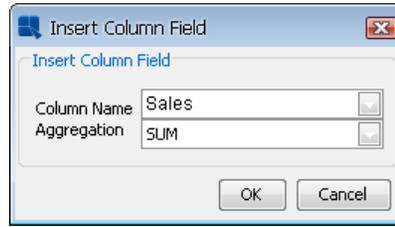
**Column Fields:**

You can insert a column field in one of two ways: by selecting *Insert Column Field* from the *Insert* menu or by clicking the  *Insert Column Field* button on the toolbar. After you select the *Insert Column Field* option, a dialog box will appear with two drop-down menus. The first one allows you to select the column field you want to insert into the report. The second one allows you to select aggregation (if any) for the column field. After you select the desired column field, click the *OK* button and a small box will follow your mouse pointer around the design window. Position the box where you want to place the column field and click. The column field will appear in the report. The column field is automatically anchored to the report section in which it is inserted. This means that each time the section repeats, the column field will display the next row in the data column.



**Note**

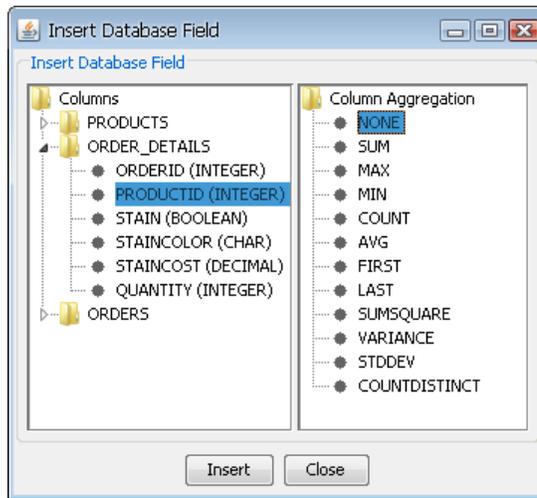
The column value will not appear in the Design window (unless it is placed in the Data Table section). All that will show is the column name {name}. You can see the column field value in the preview window.



*Insert Column Field*

**Database Field:**

This option is only available if the current report uses a database as the data source. It allows you to add a field from the database into the report that may not have been initially selected by the query or as part of the data mapping. To insert a database field, select *Insert Database Field* from the *Insert* menu. This will open a dialog with all of the tables you have included in your query and their respective fields, allowing you to select the field you want to add. Another option allows you to select the aggregation (if any) that you want to perform on the field.



*Insert Database Field Dialog*

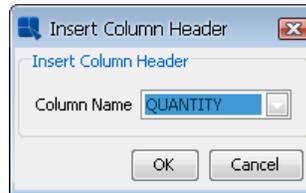


**Note**

If the template's query contains aggregation or a *Group by* clause, a third option will appear in the dialog prompting you to select *Database Aggregation* for the field. This can either be group by or any aggregation supported by the database.

**Column Header:**

Column headers are a unique report element that will dynamically display column header from the data source. You can add a column header by selecting *Insert Column Header* from the *Insert* menu. This will bring up a dialog prompting you to select the column for which you want to retrieve the header.

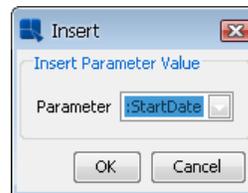


*Insert Column Header Dialog*

Select a desired column and click the *Ok* button. A box will then allow you to position the header wherever you want in the report. Click to add the column header.

**Parameter Values:**

You can insert user supplied values to query or formula parameters directly into the report. To insert a parameter value, select *Insert Parameter Value* from the *Insert* menu. This will bring up a dialog prompting you to select which parameter you want to display the value for.



*Insert Parameter Value Dialog*

Parameter values are also accessible through formulas. For more information about formulas, please see Section 4.1.6 - Using Formulas & the Formula Builder.

**Images:**

You can insert an image in one of two ways: by selecting *Insert Image* from the *Insert*

menu or by clicking the  *Insert Image* button on the toolbar. After you select this option, a small box will follow your mouse pointer around the Design window. Position the box where you would like to place the image and click. A new window will appear prompting you to enter the URL of the image you want to insert.

There are two ways of inserting images: either locate the image on your hard drive or retrieve it from an URL.

To locate an image on your hard drive, click the *Browse* button. If you want to use an image from a different location, insert its URL.

To retrieve image from an URL, enter the URL in the *Image URL* text field. After that, click the *Refresh Preview* button to verify the URL. If the image from the URL appears in the *Preview* section, the URL is correct.



**Note**  
You have to insert complete URL with protocol (e.g. `http://`).

If you save the report as PAK, the image will be stored in the PAK file along with the report.

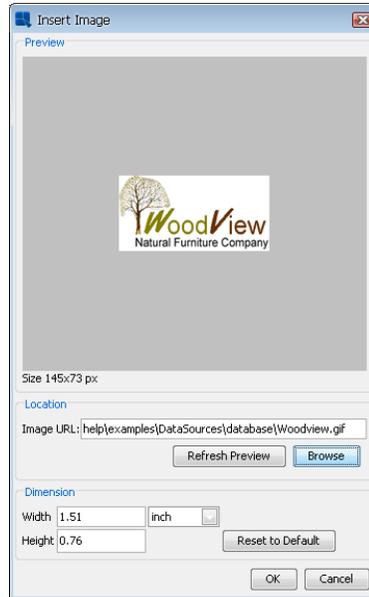
If you want to re-open a RPT file (from an older ERES version) with a background image, please note that the image itself is not stored within the report. Only the path or URL is saved. If you move a RPT report, you need to be sure that it can still access the image using the specified path.

You can manually set the image dimensions in pixels, inches, or millimeters in the *Dimension* section.

**Note**

If the image dimension exceeds the page dimension, the image will not be shown (although space will be allocated for that image).

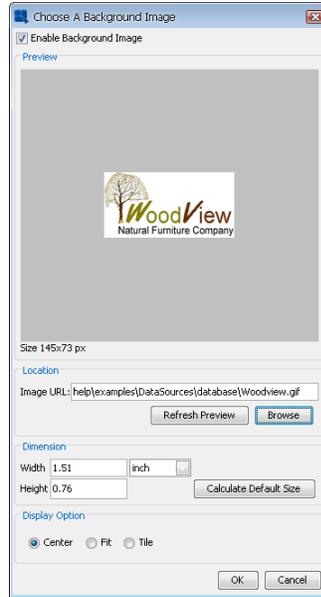
ReportDesigner also supports database images. You can add images from a database (BLOB format) by selecting the BLOB field as part of the query for the report. The images will be added as a column in the report. You can also retrieve images in the Table Data section via URL instead of using a BLOB field. For more information about this, please see Section 4.1.3.7.3 - Data Formatting for Formulas and Column Fields.



*Insert Image Dialog*

**Background Images:**

ReportDesigner allows you to add background images to a report. Background images will underlay all other report elements and will repeat on every page. To add a background image, select *Background Image* from the *Option* menu. This will bring up a dialog prompting you to specify location of the image, as well as several display options.



*Background Image Dialog*

To insert a new image, select the *Enable Background Image* option. If you want to remove an existing background image and use simple background color instead, unselect this option.

This dialog also allows you to choose one of the *Display options (Center, Fit and Tile)*.

The rest of this dialog is the same as the *Insert image* dialog described in the previous section.

**Lines/Rectangles:**

To insert a line or a grid rectangle, select either *Insert Line* or *Insert Rectangle* from the *Insert* Menu. If you select a line, you can further select whether you want to insert a vertical or horizontal line. You can also select the corresponding icons on the toolbar. After you select a line or rectangle, a cross or a box will follow the mouse pointer around the Design window. Position the cross/box where you would like to insert the line/rectangle and click, then drag the cross to draw the line or rectangle. A line or grid will then be inserted into the report.

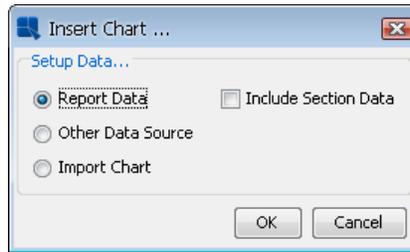


**Note**

The line or grid will appear thicker than it actually is in the Design window.

**4.1.3.7.1.1. Inserting Charts**

You can insert a chart in one of two ways: by selecting *Insert Chart* from the *Insert* menu or by clicking the  *Insert Chart* button on the toolbar. After you select this option, a small box will follow your mouse pointer around the design window. Position the box where you want to place the chart and click. You will see the following dialog box:



*Insert Chart Dialog*

You have an option to use data from the current report or use an external data source. If you choose to use the data from the report, you will be shown a preview of your dataset and you will be taken directly to the chart selection window in the Chart Designer. Otherwise, you will be given an option to either select a different data source for your chart or import a chart file created before.

Once you select your options, the Chart Designer will open in a new window, allowing you to design and customize your chart. Please see the Section 4.2.1 - Introduction to Chart Designer for more information about using the Chart Designer.



**Note**

The actual chart will not appear in the Design window. Instead, you will see a gray rectangle with the chart URL. You can see the actual chart in the preview window.

If your report contains grouping (Summary Break, CrossTab, or Master & Detail Reports), you will also have the option to *Include Section Data*. If you select this option, you will be able to use group aggregations that are located within any section that the chart encompasses. This means that you can create charts using data from any aggregation located in the same section as the chart or in any inner sections.

For example, if you were to create a Summary Break Report with two row break columns, this will result in two group header and two group footer sections. Assuming we have a group aggregation for a column in your report, the report might look like this:

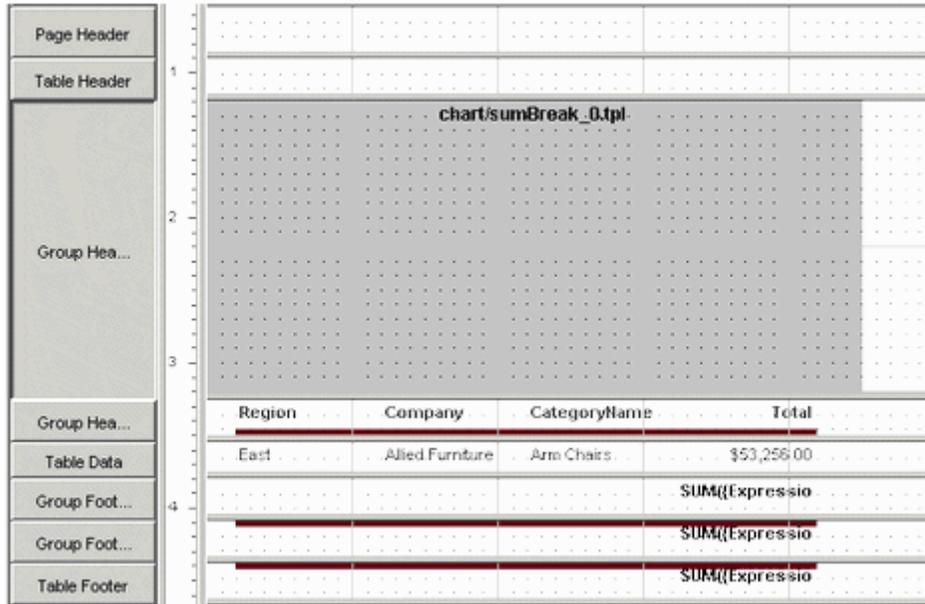
Report Designer						
	in	1	2	3	4	5
Report Hea...	in					
Page Header						
Table Header	1	Region . . . . .	Company . . . . .	CategoryName . . . . .	Total . . . . .	
Group Hea...		Group Header 0 . . . . .				
Group Hea...		Group Header 1 . . . . .				
Table Data	2	East . . . . .	Allied Furniture . . . . .	Arm Chairs . . . . .	\$53,256.00	
Group Foot...		Group Footer 1 . . . . .			SUM(Expression)	
Group Foot...		Group Footer 0 . . . . .			SUM(Expression)	
Table Footer	3				SUM(Expression)	
Page Footer						
Report Foo...						

*Summary Break Report with Two Row Break Columns*

If you position the chart in the inner group header (Group Header 1) or inner group footer (group footer 1), you will only be able to use the group aggregation for the inner section in your chart (the aggregation in Group Footer

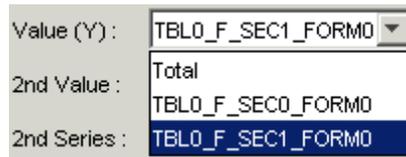
1). If you positioned the chart in the outer group header (Group Header 0) or outer group footer (Group Footer 0), you will be able to use the aggregation for the outer section as well as the inner section in your chart because the outer section encompasses the inner section. Positioning the chart in the table header or table footer sections will allow you to use any group aggregation in the report.

Suppose you moved the column headers to Group Header 1 and positioned the chart in Group Header 0 as shown below.



*Insert Chart in Outer Group Header*

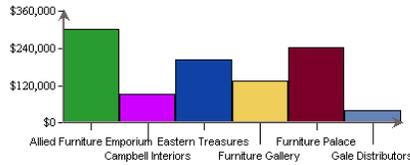
When you see the data mapping window in Chart Designer, you will be able to use the summary data from both Group Footer 0 (TBL0\_F\_SEC0\_FORM0) and Group Footer 1 (TBL0\_F\_SEC1\_FORM0).



*Chart Mapping with Section Data*

To make the aggregation easier to recognize, you can use Custom IDs. For more information about using Custom IDs, see Section 4.1.6.2.1 - Using Column Field Data.

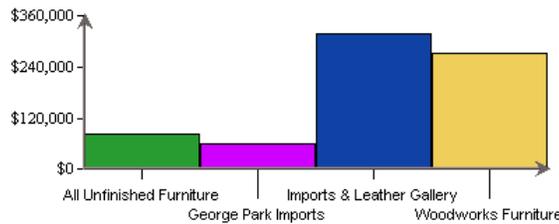
The resulting report will look like the image below. Notice that the chart plots the aggregation for each company. If you used the aggregation from Group Footer 0, the chart would have shown the aggregation for each region instead.



Region	Company	CategoryName	Total
East	Allied Furniture	Arm Chairs	\$53,256.00
		Double Dressers	\$58,144.00
		Oval Tables	\$52,142.00
		Round Tables	\$23,805.00
		Side Chairs	\$23,540.00
		Single Dressers	\$92,039.00
		<b>Total</b>	<b>\$302,926.00</b>
	Campbell Interiors	Arm Chairs	\$15,609.00
		Oval Tables	\$11,394.00
		Rectangular Tables	\$7,290.00
Round Tables		\$42,916.00	
Triple Dressers		\$15,936.00	
<b>Total</b>	<b>\$93,145.00</b>		
Eastern Treasures	Arm Chairs	\$73,505.00	
	Oval Tables	\$3,750.00	
	Rectangular Tables	\$41,511.00	

Report with Chart Using Section Data (East)

Since the chart is placed in Group Header 0, a new chart will be displayed for every region. The above image shows the chart for the East region only, but the report will contain charts corresponding to each of the other regions as well. Here is the chart for the Midwest region:



Region	Company	CategoryName	Total
Midwest	All Unfinished	Arm Chairs	\$14,532.00
		Oval Tables	\$27,884.00
		Rectangular Tables	\$7,530.00
		Round Tables	\$11,109.00
		Side Chairs	\$9,903.00
		Single Dressers	\$13,839.00
		<b>Total</b>	<b>\$84,797.00</b>
	George Park	Arm Chairs	\$5,424.00
		Oval Tables	\$13,424.00
		Rectangular Tables	\$5,416.00

Report with Chart Using Section Data (Midwest)

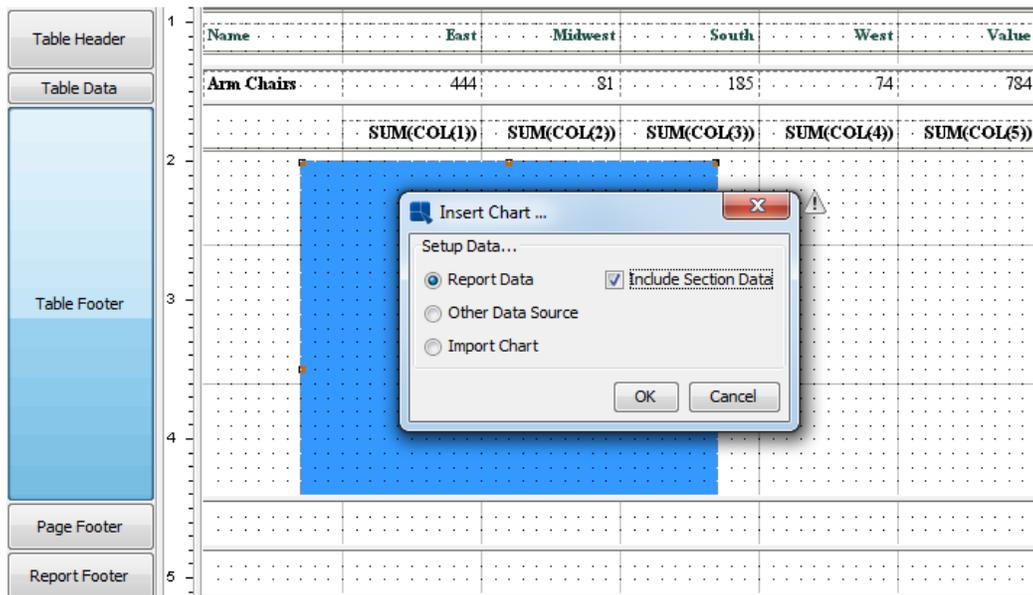
Keep in mind that if you select the *Include Section Data* option, the data available for the chart will vary between sections, so moving the chart between sections is not recommended. If you need to move the chart after it is made, make sure that the data mapped to the chart is also available from the location where you are moving the chart to.

Here is another example using *Include Section Data*. Suppose you had the following CrossTab report and you wanted to use the aggregation for each column as data points in your chart.

Name	East	Midwest	South	West	Value
Arm Chairs	444	81	185	74	784
Double Dressers	68	12	27	0	107
Oval Tables	79	33	24	16	152
Rectangular	85	82	67	52	286
Round Tables	143	109	72	46	370
Side Chairs	150	216	377	100	843
Single Dressers	79	100	114	18	311
Triple Dressers	21	45	19	41	126
	<b>1 069</b>	<b>678</b>	<b>885</b>	<b>347</b>	<b>2 979</b>

*CrossTab Report with Group Aggregation*

To make the aggregations easier for later use, give each group aggregation a Custom ID that matches their column header. For example, give  $SUM(COL(1))$  the ID **SumEast**, give  $SUM(COL(2))$  **SumMidwest** and so on. For more information about Custom IDs, see Section 4.1.6.2.1 - Using Column Field Data. Then insert the chart in the Table Footer Section using Report Data and Include Section Data.



*Insert Chart in Table Footer*

The formulas with custom IDs will be visible in the data source preview (if you selected the *Include section data* option).

INDEX	Name	East	Midwest	South	West	Value	SumEast	SumMidwest	SumSouth	SumWest	TBL0_FTR...
TYPE	Varchar	Integer	Integer	Integer	Integer	Integer	Double	Double	Double	Double	Double
1	Arm Chairs	444	81	185	74	784	1069.0	678.0	885.0	347.0	2979.0
2	Double Dres...	68	12	27	0	107	1069.0	678.0	885.0	347.0	2979.0
3	Oval Tables	79	33	24	16	152	1069.0	678.0	885.0	347.0	2979.0
4	Rectangular ...	85	82	67	52	286	1069.0	678.0	885.0	347.0	2979.0
5	Round Tables	143	109	72	46	370	1069.0	678.0	885.0	347.0	2979.0
6	Side Chairs	150	216	377	100	843	1069.0	678.0	885.0	347.0	2979.0
7	Single Dressers	79	100	114	18	311	1069.0	678.0	885.0	347.0	2979.0
8	Triple Dressers	21	45	19	41	126	1069.0	678.0	885.0	347.0	2979.0

Data source preview with section data

Since the data points are in different aggregations, you will need to use the transpose feature. To learn more about data transposition in charts, see the Section 4.2.3.1.1 - Data Transposition chapter.

In the data mapping window, select the *Multi Selection* option for the Category (X). This option allows you to select multiple columns for the category. Select all the formulas with custom IDs. The Value (Y) should be automatically set to *Value*.

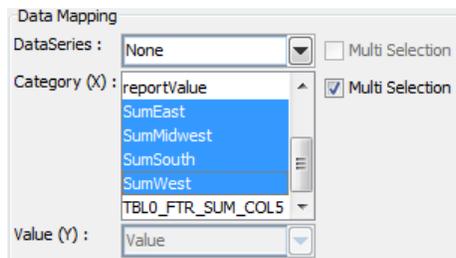
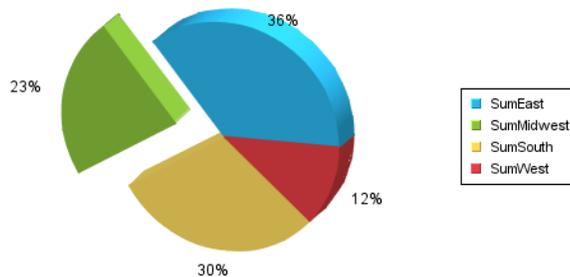


Chart Mapping for Transposed Data

Depending on the type of chart you select, the finished report might look like this:

Name	East	Midwest	South	West	Value
Arm Chairs	444	81	185	74	784
Double Dressers	68	12	27	0	107
Oval Tables	79	33	24	16	152
Rectangular	85	82	67	52	286
Round Tables	143	109	72	46	370
Side Chairs	150	216	377	100	843
Single Dressers	79	100	114	18	311
Triple Dressers	21	45	19	41	126
	<b>1 069</b>	<b>678</b>	<b>885</b>	<b>347</b>	<b>2 979</b>

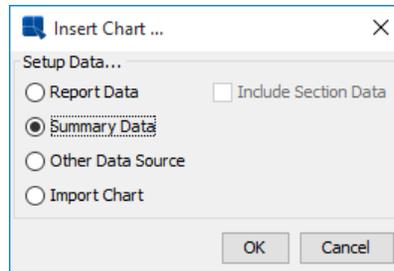


Report with Chart Using Section Data

#### 4.1.3.7.1.1.1. Inserting Summary Charts into Crosstab Reports

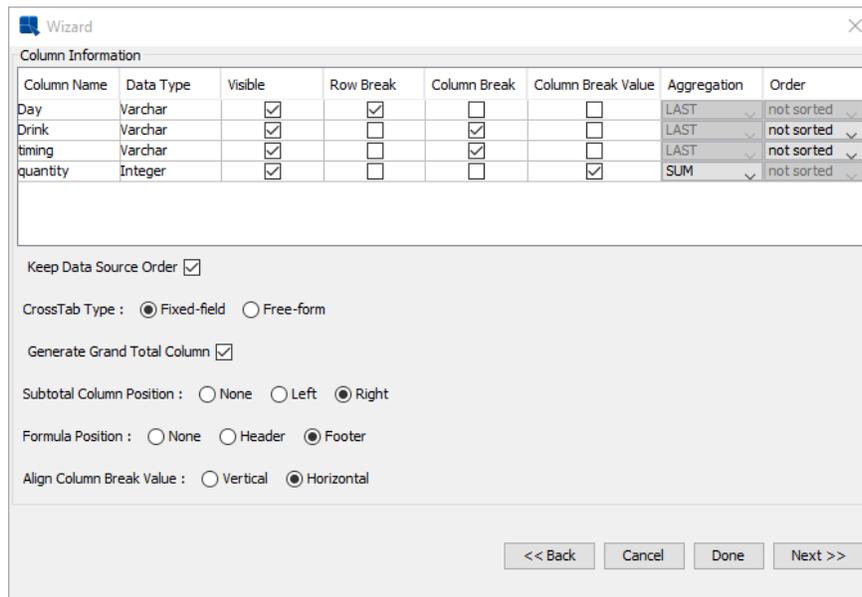
You can also insert summary charts into crosstab reports. A summary chart is a type of chart that is created from summarized data of a fixed-field crosstab report. For more information about crosstab reports, please see Section 4.1.2.3 - Crosstab Report.

When you are going to insert a chart into a crosstab report, you will still have the option to use data from the current report or use an external data source. Otherwise, you will be given an option to select a summary data source for your chart or import a chart file created before. Note that a summary chart can only be inserted into the Table Header/Footer sections, depending on the Formula Position. For example, if you created a fixed-field crosstab report with the formula position set to be drawn in the Header section, you can only insert a summary chart into the Table Header section. If you try to insert a chart to other report sections, the *Summary Data* option will not be available in the Insert Chart dialog.



*Insert Chart Dialog*

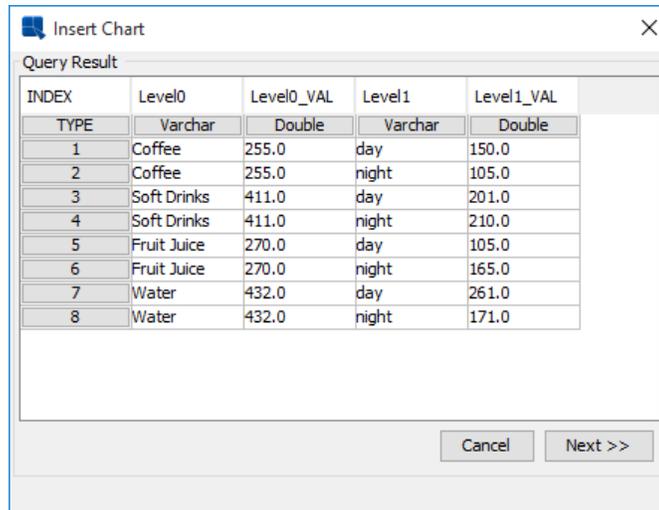
For example, say we have the following data mapping for a crosstab report:



*Data Mapping*

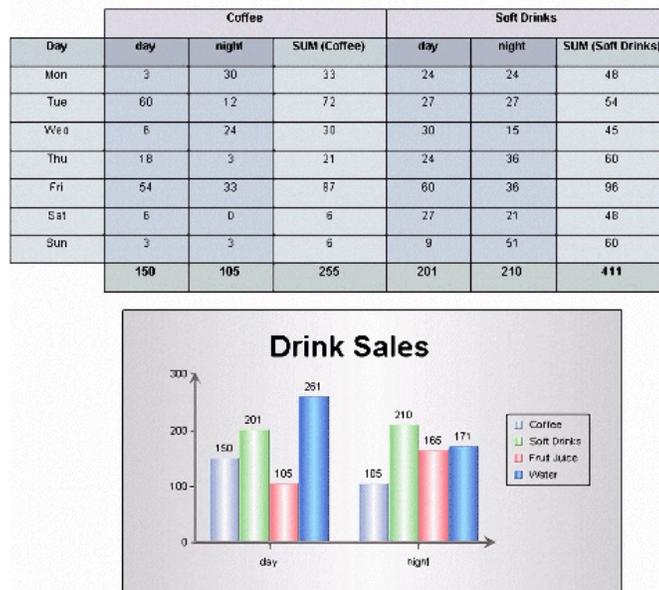
We selected Day as a *Row Break* field, Drink and Timing as *Column Break* fields, Quantity as a *Column Break Value* field with an aggregation of **Sum**. We also chose the *Formula Position* to be drawn in the Table Footer section of the report, which means we have to insert the summary chart into the Table Footer section in which data will be summarized.

Inserting a summary chart into the table footer section and selecting the **Summary Data** as datasource in the *Insert Chart* dialog will show the following Query Result dialog. The dialog shows data that will be used for a summary chart.



Query Result

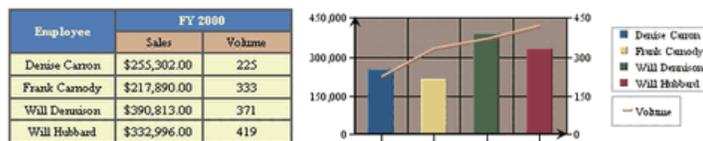
Depending on the type of chart and formatting you select, the finished report might look like this:



Report Preview

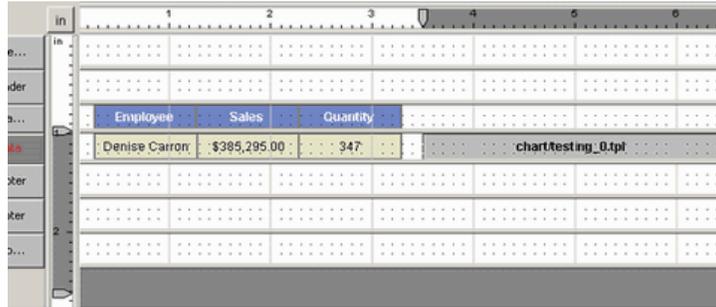
#### 4.1.3.7.1.2. Inserting Elements into the Table Data Section

Generally, when elements are inserted into a report section, they will repeat each time that the report section repeats. However, this does not apply when certain elements are inserted into the Table Data section. Charts, images, and sub-reports do not repeat for each row of data (labels, column fields, and formulas do) when inserted into the Table Data section. This allows you to design a report in a side-by-side configuration where you might have a chart or in sub-report placed next to a data table as pictured below.



Side-by-Side Data Table & Chart

Normally, when you insert a chart, image, or sub-report into a report section, you will want to resize the section so that the entire element fits within the section (otherwise it is truncated). This is not the case for charts, images, or sub-reports inserted into the Table Data section. In this situation, you will want to keep the section the same height as your column fields regardless of the size of the element you're placing in the section.



Side-By-Side Data Table & Chart (Designer)

You can resize an element in the Table Data section using the rulers or by temporarily resizing the section to adjust the element size.

#### 4.1.3.7.2. Editing Elements

You can edit elements in one of four ways: by selecting *Edit* from the *Edit* menu, clicking on the  *Edit* button in the toolbar, right clicking on the element, and then selecting *Edit* from the pop-up menu, or by double clicking on the element.

##### Editing Labels:

When you select the *Edit* option for a label a dialog box will appear prompting you to change the label text. Click on *OK* and the label will be changed.

##### Editing Formulas:

When you select the *Edit* option for a formula, the Formula Builder will appear prompting you to change the formula. Make any changes you want, then click the *OK* button and the formula will be changed.



#### Note

If you use the same formula in more than one place in the report, modifying the formula will change it everywhere. You can avoid this by clicking the *Save As* button in the Formula Builder. This will allow you to specify a new name for the modified formula and modify it only for the particular element.

##### Editing Charts:

When you select the *Edit* option for a chart, the Chart Designer will open, allowing you to edit and format the chart.

##### Editing Images:

When you select the *Edit* option for an image, a new window appears prompting you to select a new image or directory URL. After selecting the new image, click the *OK* button and the new image will replace the old one.

##### Editing Column Headers:

Column header is a unique type of report element that is generated when you first create the report. By default, column headers are dynamic and will display the name of the column even when the column changes (by changing data mapping, source, or applying a template). When you select to edit a column header, a dialog box will appear prompting you to change the text of the header.



*Edit Column Header Dialog*

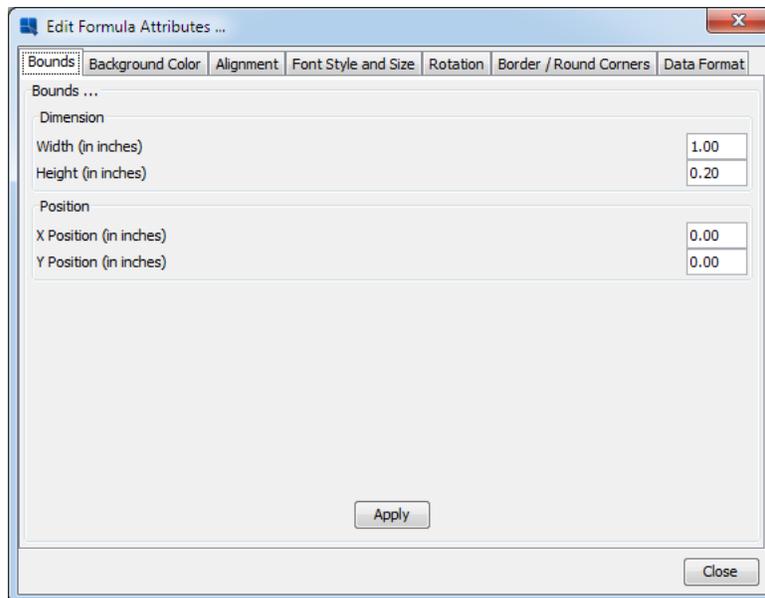
When you edit the header, a new text will be displayed. However, the header will no longer be dynamic. You can return the header to display the (dynamic) column name by right clicking on it and selecting *Original Column Header* from the pop-up menu.

You can also copy and paste cell attributes between report elements. This will apply all attributes including data formatting (assuming the data types are the same), alignment, font, color, and border attributes, as well as the bounds of a cell.

To copy the attributes of an element, right click on the element and select *Element Appearance* from the pop-up menu. This will expand into two additional choices. Select *Copy* from the second menu. To apply the copied attributes to another cell, repeat the same steps as before and select *Paste* from the secondary menu.

#### 4.1.3.7.2.1. Editing Element Attributes

In addition to directly editing an element, you can directly access all of the element's attributes by selecting *Edit Attributes* option from the *Format* menu, or by right clicking on an element and selecting *Edit Attributes* from the pop-up menu. This will bring up a tabbed dialog that allows you to set a number of properties for the element at once.

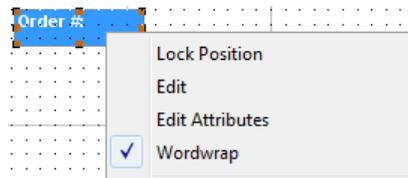


*Multi-Attribute Editing Dialog*

The options will vary depending on what type of element you select. For each tab in the dialog, you can set the option you want and then click the *Apply* button to set the changes for that attribute.

#### 4.1.3.7.2.2. Word Wrapping

By default, any text within report elements, including labels, formulas, and columns will wrap to the next line if the cell boundaries are not wide enough to fit the text. However, this behavior can be disabled. To set text wrapping, select an element and select *Format* → *Wordwrap*, or right click on an element and select the option from the pop-up menu. This will disable/enable text wrapping.

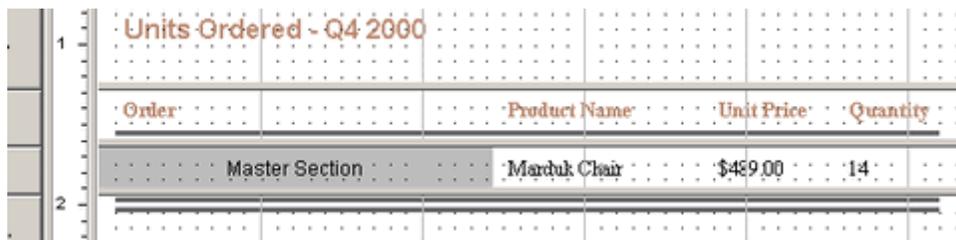


*Word Wrapping Dialog*

To disable word wrapping, un-check the option. Note that when word wrapping is disabled, the entire contents of the cell will be printed regardless of the cell boundaries. This can overlap and obscure other report elements. Also, word wrapping cannot be disabled for rich text fields as the rich text and Excel-based exports will still display the wrapped text.

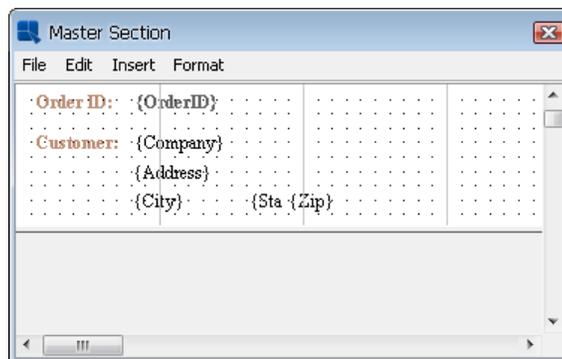
#### 4.1.3.7.2.3. Editing Side-By-Side Master & Details Reports

As noted in Section 4.1.2.4 - Master & Details Report of this guide, you can specify a side-by-side layout for master & details reports. This will print the master field (group header) next to the details section (table data), rather than its traditional position above the details section. When you select this report format, the master section will appear as a gray rectangle next to the column fields within the table data section of the report.



*Side-by-Side Master & Details*

The master section can be moved in free-form like any other report element. You can edit the contents of the master field in one of three ways: selecting it and clicking the *Edit* button on the toolbar, right clicking it and selecting *Edit* from the pop-up menu, or double clicking it. The master section will open in a new window, allowing you to modify its contents.



*Edit Master Section Dialog*

The elements in this section can be moved, resized, and edited in the same way as other elements in the report. The master section acts the same as a group header section. It repeats for each unique value in the primary key column. After you finish editing the contents of the master section, it can be closed by selecting *Close* from the *File* menu.



#### **Note**

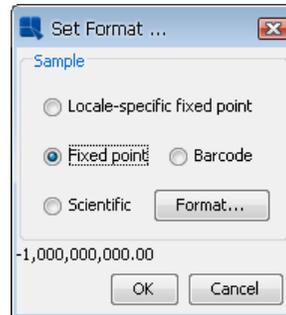
After you finish editing the master section, it will automatically resize to fit its contents.

### 4.1.3.7.3. Data Formatting for Formulas and Column Fields

You can change the output of formulas and column fields in one of three ways: clicking the  *Data Format* icon in the toolbar, selecting *Data Format* from the *Format* menu, or right clicking on a formula or column field and selecting *Edit Attributes* from the pop-up menu. When you select the format option, a dialog box will appear (or the format tab of the multi-attribute editing dialog). The box that appears depends on what type of data is present in the selected element: numeric, string, date/time, or logical/boolean.

#### Formatting Numeric Data:

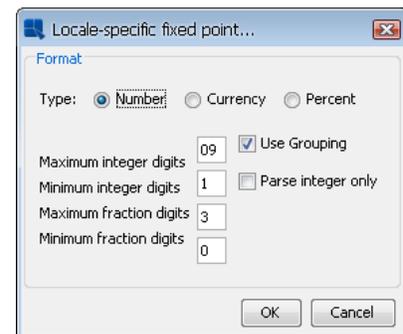
The dialog box for numeric data contains four primary options for the data: locale-specific fixed point, fixed point, bar code, and scientific. You can select the option you want and then click on format for additional options.



*Numeric Format Dialog*

#### Locale-Specific Fixed Point:

This will change the data format depending on the locale in which it is being viewed. Additional formatting for this option allows you to specify whether the data should be displayed as a number, currency, or percentage. Additionally, you can set the maximum and minimum number of integer digits and fraction digits. Other display attributes will vary depending on locale.

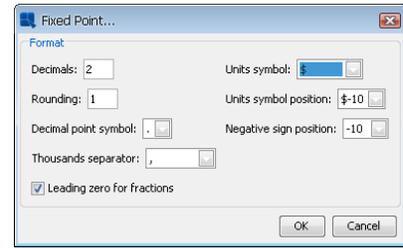


*Locale-Specific Formatting*

#### Fixed Point:

This will keep the data format consistent, regardless of locale. Additional formatting for this option allows you to set the number of decimals, rounding for digit number, unit symbols, negative sign positions, deci-

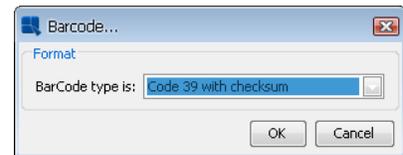
mals and thousands separator, and specify leading zeroes for fractions.



*Fixed Point Formatting*

**Bar Code:**

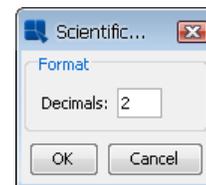
This will convert the data into a bar code. Additional formatting for this option allows you to select the symbology to use for the bar code. Supported symbologies are Code 39 with and without checksum, Interleave 2 of 5, UPC A, EAN 13, EAN 128, Standard 2 of 5 with and without check digit, Code 128, Code 128A, Code 128B, Code 128C, USD 3 with/without checksum, Code 3 of 9 with and without checksum, Global Trade Item Number, Random Weight UP-CA, SCC 14 Shipping Code, Shipment Identification Number, SSCC 18, and US Postal Service.



*Bar Code Formatting*

**Scientific:**

This will display the data in scientific notation. Additional formatting for this option allows you to set the number of decimals.

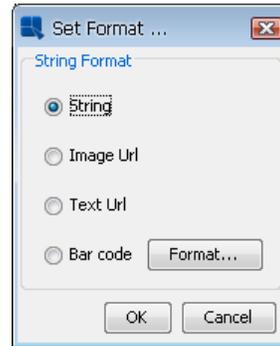


*Scientific Formatting*

After you finish selecting additional options, click the *OK* button to return to the main dialog box. Click the *OK* button and the data format will be changed.

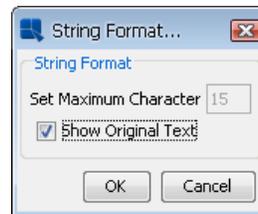
**Formatting String Data:**

The dialog box for string data contains four primary options: string, image URL, text URL, and bar code. You can select the option you want and in the case of string or bar code, click *Format* for additional options.



*String Data Format*

**String:** This dialog allows you to format the appearance of the string. The checkbox labeled *Show Original Text* controls whether the complete string for each data entry will be displayed. If you un-check it, you can then specify the maximum number of characters to be displayed. Click the *OK* button to return to the previous dialog.

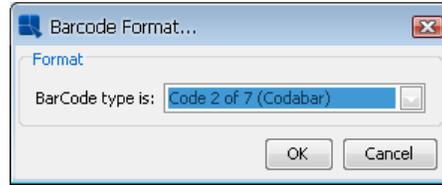


*String Formatting*

**Image URL:** This will convert the string into an image. This is used in situations where instead of storing an image as an object in a database (BLOB), you have stored URLs that point to image files on a server. Selecting this option will read the URL and retrieve the images to be placed in the report.

**Text URL:** This will convert the string into a large text object. This is used in situations where instead of storing a large text file as an object in a database (CLOB), you have stored URLs that point to files on a server. Selecting this option will read the URL and retrieve the text files to be placed in the report.

**Bar Code:** This will convert the data into a bar code. Additional formatting for this option allows you to select the symbology to use for the bar code. Available symbologies for string data are Code 39 with/without checksum, Code 2 of 7 (Codabar), Codabar, EAN 128, Code 128, Code 128A, Code 128B, Code 128C, Global Trade Item Number, Monarch, NW 7, SCC 14 Shipping Code, Shipment Identification Number, SSCC 18, US Postal Service, and USD 4.



*Bar Code Formatting*

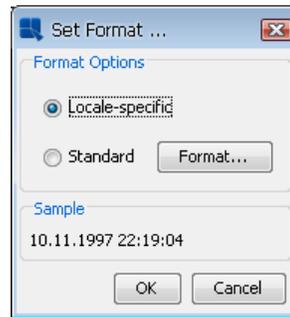


**Note**

Codabar will not accept start/stop characters in the data. If the input data is incorrect, the bar code may not be readable.

**Formatting Date/Time Data:**

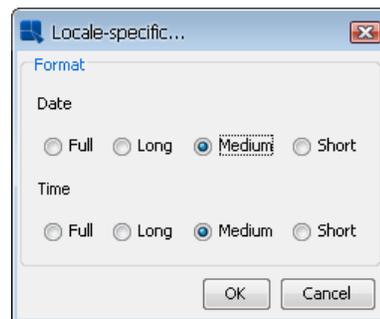
The dialog box for date/time data contains two primary options for the data: locale-specific and standard. You can select the option you want and click the *Format* button for additional options. The available additional options will vary depending on the nature of your data. Date, time, and timestamp data will bring up date, time, and date & time options respectively.



*Date/Time Data Format*

**Locale Specific:**

This will change the format of the data depending on the locale in which it is being viewed. Additional formatting for this option allows you to select full, long, medium, or short notations for date and time information. Other display attributes will vary depending on locale.

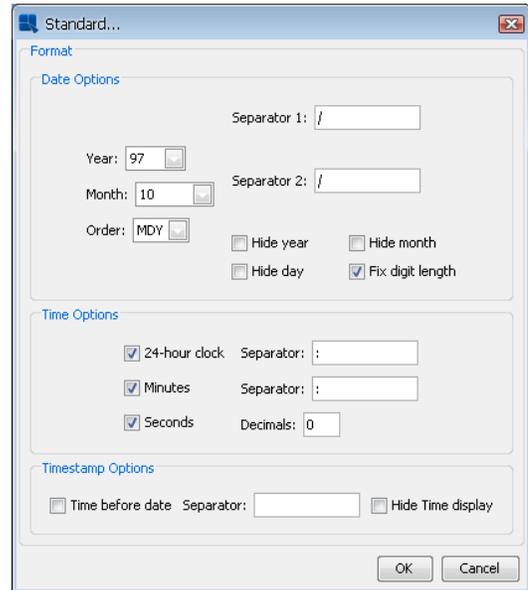


*Locale-Specific Formatting*

**Standard:**

This will keep the data format consistent, regardless of locale. Additional formatting for this option allows you to select year and month displays,

as well as the order in which month, day, and year information is presented. You can also select the characters to be used as separators. Time options allow you to display hours, minutes, and/or seconds, and select the separators between them. For timestamp data, you can select to display the time before or after the date and the separator to be used between them.

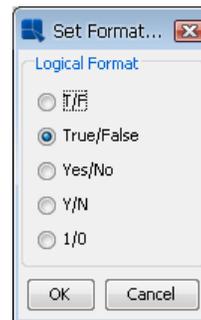


*Standard Formatting*

After you finish selecting additional options, click the *OK* button to return to the main dialog box. Click the *OK* button and the data format will be changed.

**Formatting Logical Data:**

The dialog box for Logical or Boolean data contains five options for displaying the data. They are: T/F, True/False, Yes/No, Y/N, and 1/0. Select the format you want to use and then click the *OK* button to change the data format.



*Boolean Data Format*

**4.1.3.7.3.1. Formatting Null Data**

By default, null data will display in a report as `Null`. You can change the appearance of null data by selecting *Null Data Handler* from the *Option* menu. This will bring up a dialog, prompting you to specify a string to be displayed for nulls.



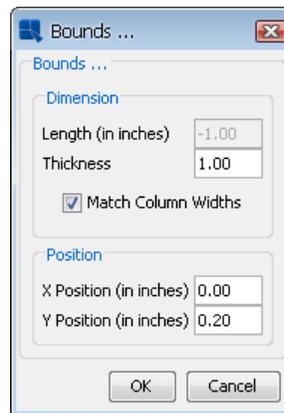
*Null Data Handler*

Enter a value you want and click the *OK* button. All the null values in the report will be displayed as the value you specified.

#### 4.1.3.7.4. Line/Rectangle Format

You can format the appearance of lines and rectangles in the report by right clicking on a line or rectangle element. The pop-up menu will contain a list of available options.

**Lines:** There are three formatting options for lines. *Line Color & Thickness* allows you to set the color and thickness of the line. *Line Style* allows you to select the style of line - either solid, dotted, or double. *Set Bounds* allows you to specify the length and thickness of the line, as well as the X and Y coordinates of its origin. In addition, you can set the width of horizontal lines to match the aggregate width of columns in the report. This feature is useful for crosstab reports where the number of columns can vary and also for user-defined report styles. For vertical lines, you can set the length to match the section height.



*Bounds Dialog for Horizontal Lines*

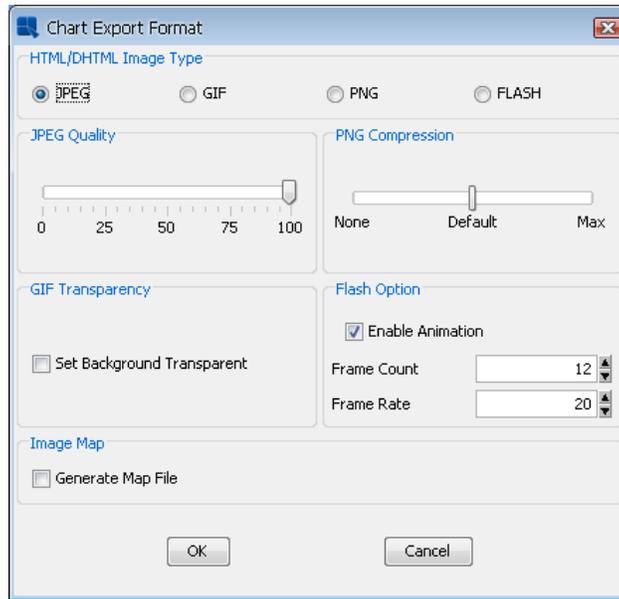
**Rectangles:** There are four formatting options for grid rectangles. *Border* allows you to set the border thickness and color for the grid rectangle. *Background* allows you to set a background color for the fill area of the rectangle. *Set Bounds* allows you to specify the width and height of the grid rectangle, the X and Y coordinates of its origin, as well as the degree of rounding for the corners. *Border Style* allows you to select the line style of the border - either solid, dotted, or double.

#### 4.1.3.7.5. Chart Export Format

This option allows you to specify attributes for charts exported to DHTML. You can format the chart export properties by selecting *Chart Export Format* from the *Format* menu or by right clicking on a chart and selecting *Export Format* from the pop-up menu. When you select this option, a dialog box will appear. You can select GIF, PNG, JPEG, or FLASH for the image type and set some options for each type. For GIF images, you can set the background to be transparent. For PNG images, you can set the compression. For JPEG images, you can set the quality (Image quality is directly proportional to file size). For FLASH images, you can enable animation and set the frame count and frame rate.

You can also specify to create an image map when the report is exported. If you create an image map, the hyperlinks defined in the chart will be active when the report is exported. If no hyperlinks are defined, the map will contain

display pop-up labels for the chart containing data point information. Note that FLASH images do not support image maps.



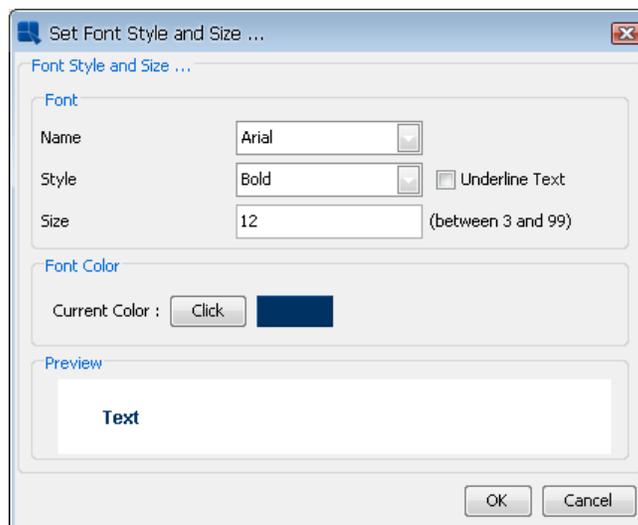
*Chart Export Format*

#### 4.1.3.7.6. Font, Color, and Border Options

You can change the appearance of report elements by changing the font, color, and border properties.

##### Font Style and Size:

The font style and size can be adjusted for labels, formulas, and column fields in one of three ways. You can directly modify the font style and size using the options in the Report Designer toolbar. In addition, you can also select *Font Style and Size* from the *Format* menu or right click on a report element and select *Edit Attributes* from the pop-up menu. The latter two options will invoke a separate dialog prompting you to select the font style and font size or in the attribute editing dialog, where font options can be set in the 'Font Style and Size' tab.



*Font Style and Size Dialog*



### Note

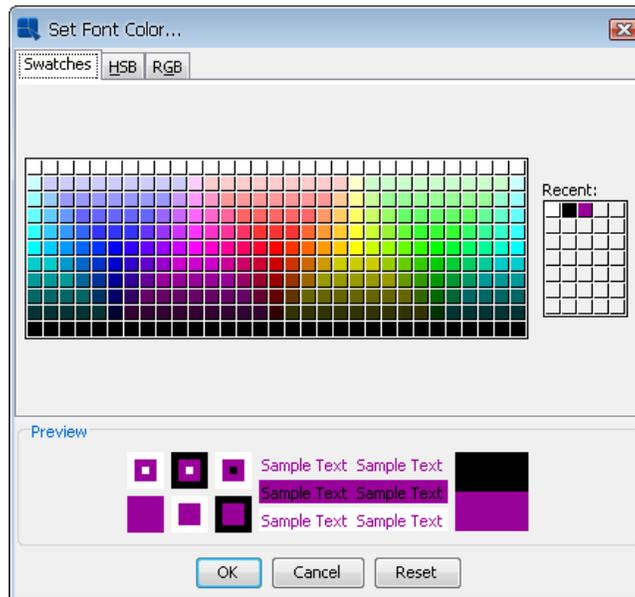
You can use any fonts from the system when designing a report. However, these fonts may no longer be present when you move reports between platforms. Also, you will have to explicitly map font files (.ttf) to font styles when exporting a report to PDF. For more information about this, please see Section 4.1.5.2.1 - PDF Font Mapping.

#### Alignment:

The alignment for labels, formulas and column fields can be adjusted in one of three ways. You can directly adjust the horizontal alignment for an element by selecting the option(s) from the toolbar. You can also select *Alignment* from the *Format* menu, or right click on the element and select *Edit Attributes* from the pop-up menu. The latter two options also allow you to select vertical as well as horizontal alignment for the element text.

#### Font Color:

The font color can be adjusted for labels, formulas, and column fields in one of three ways: selecting *Font Style And Size* from the *Format* menu, clicking the  *Font Style And Size* button on the toolbar, or right clicking on the element and selecting *Edit Attributes* from the pop-up menu. Selecting this option will bring up a dialog box (or the multi-attribute editing dialog where options can be set in the *Font Style And Size* tab). To change the font color, click the *Click* button next to the *Current Color:* field. The Set Font Color dialog will appear allowing you to select font color from swatches. You can also enter HSB, HEX, or RGB values.

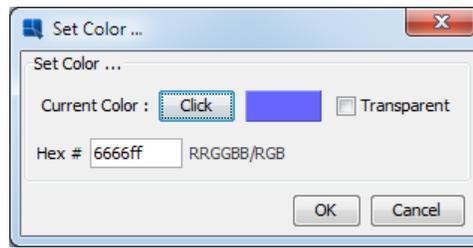


*Font Color Dialog*

#### Background Color:

The background color can be adjusted for labels, formulas, and column fields in one of three ways: selecting *Background Color* from the *Format* menu, clicking the  *Background Color* button on the toolbar, or right clicking on the element and selecting *Edit Attributes* from the pop-up menu. Selecting this option will bring up a dialog box (or the multi-attribute editing dialog where options can be set in the *Background Color* tab) prompting you to specify whether or not to set the background transparent. If you do not want a transparent background, uncheck the checkbox and click the button. This will bring up a dialog prompting you to specify

the background color. You can select font color from swatches, or enter HSB or RGB values. You can also directly enter HEX color value to the *Hex #* field.



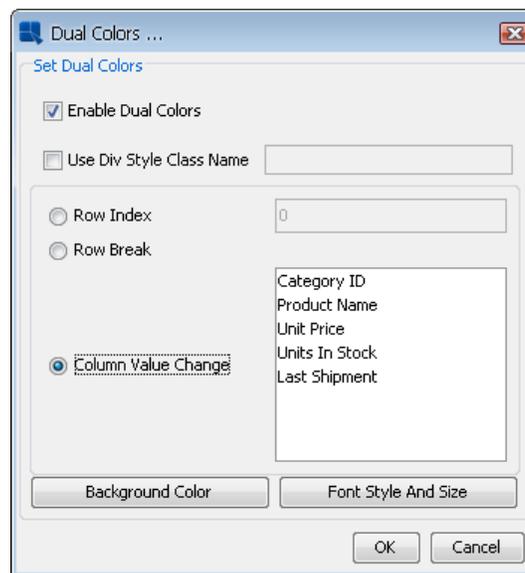
*Background Color Dialog*

**Dual Colors:**

You can specify alternating color for labels, formulas, or column fields that are in the data table section of the report in one of three ways: selecting *Dual Colors*

from the *Format* menu, clicking the  *Dual Color* button on the toolbar, or right clicking on the element and selecting *Edit Attributes* from the pop-up menu. Selecting this option will bring up a dialog box (or the multi-attribute editing dialog where options can be set in the *Dual Colors* tab), allowing you to specify the number of rows between alternating colors. Instead of specifying alternate row numbers, you can also set dual colors to change on the row break (for summary break or crosstab reports) or as a particular column value changes. Using this feature, you can set the cell attributes to change when you reach a new group in a summary break report.

For alternate rows, you can set the background color, the font color, as well as the font style and size. Set the primary attributes as you would for any other report element.



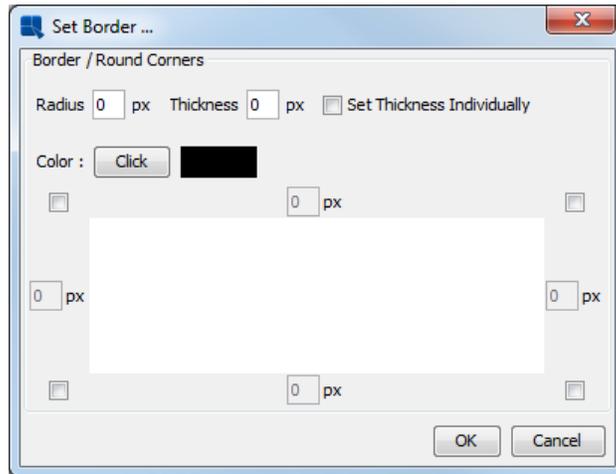
*Dual Colors Dialog*

**Border:**

You can specify the thickness and color of the border to be drawn around any report element in one of three ways: selecting *Borders* from the *Format* menu,

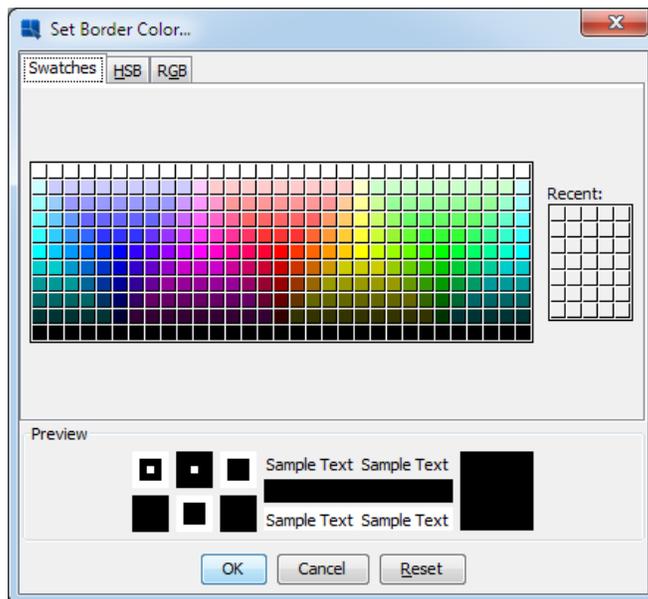
selecting the  *Border* button on the toolbar, or right clicking on the element and selecting *Edit Attributes* from the pop-up menu. Selecting this option will bring up a dialog box (or the multi-attribute editing dialog where options can be set in the

*Border/Round Corners* tab) prompting you to enter the border thickness in pixels and to choose a color. Entering 0 as the border thickness value will remove the border.



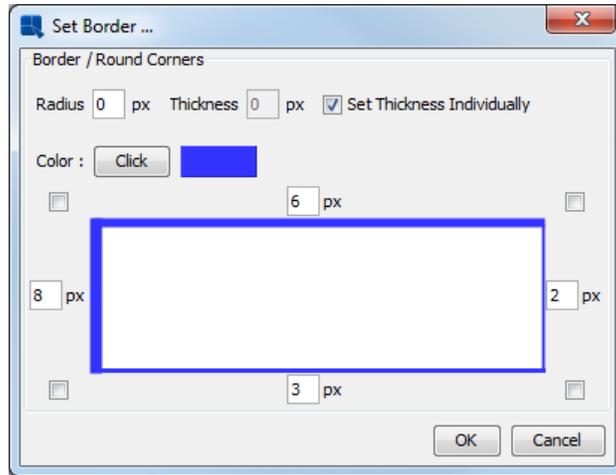
*Border Dialog*

To change the border color, click the *Click* button next to the *Color:* field. The *Set Border Color* dialog will appear allowing you to select border color from swatches. You can also enter HSB, HEX, or RGB values.



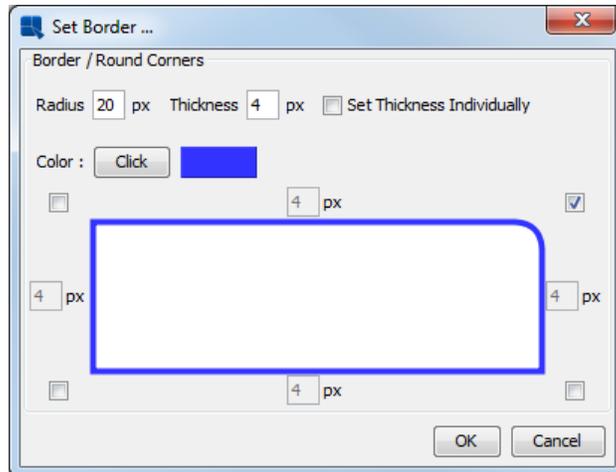
*Border Color Dialog*

To set the border thickness of the each side of the report element, check the *Set Thickness Individually* option. Then type the thickness (in pixels) in the text fields near the sides of the rectangle representing a selected element.



*Border / Individual Thickness*

To set round corners to an element, the elements need to have either non-transparent background color or a visible border, otherwise you will not see any changes. There are four checkboxes near the corners and each checkbox represents an element corner. To make a corner round, check its checkbox (you can also select several corners at once), then type a radius (in pixels) into the *Radius* text field.



*Border / Round Corners Dialog*



**Note**

If you plan on exporting reports with round corners, please note that round corners are available for DHTML and PDF export formats only.

**4.1.3.7.6.1. Show Cell Outline**

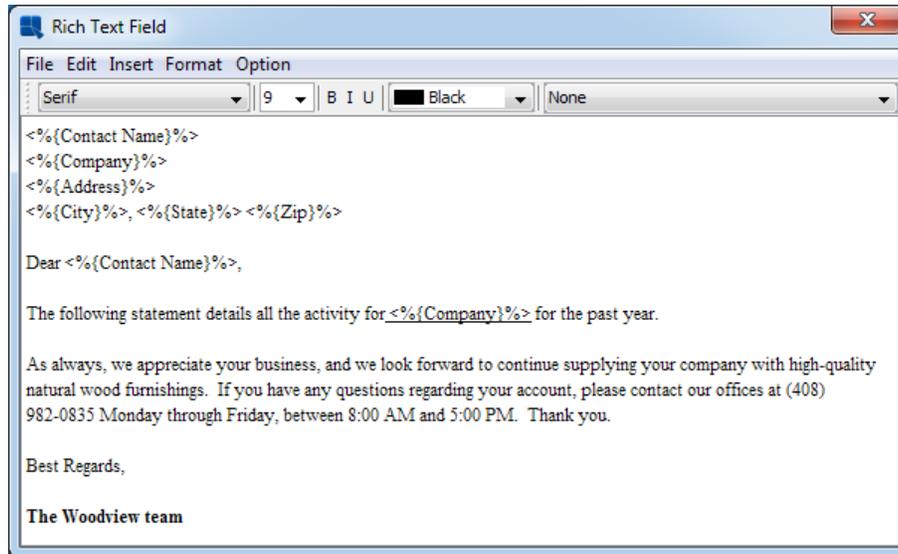
Sometimes you may want to show the boundaries or borders of report elements in the Design window, without actually turning on the borders. This is especially important if you have blank place-holder cells or columns where the first record is empty. To turn on the cell boundaries, select *Show Cell Outline* from the Option menu. This will draw a gray dotted border around each report element in the Design window.

**4.1.3.7.7. Rich Text Fields**

One of the common reporting needs is the ability to create form letters or other blocks of formatted text to a report and merge in data fields and functions. This functionality is available using rich text fields. Rich text fields, unlike

labels or string function fields, allow for complex paragraph, font, and color formatting within the cell. They also allow the ability to embed column fields and functions directly in the text flow.

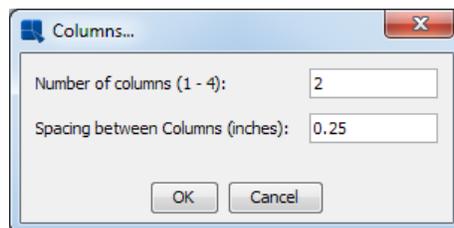
To add a rich text field to a report, select *Insert Rich Text Field* from the Insert menu, or click the  *Rich Text Field* button on the toolbar. After you select this option, a small box will follow your mouse pointer around the design window. Position the box where you want to insert the field and click. A new window will open, allowing you to enter the rich text field.



*Rich Text Editor*

The rich text editor works like a small word processor. The first drop-down box allows you to select the font and the font size. The three buttons allows you to specify bold, italic, or underlined text. The next drop-down-box allows you to specify font color. The last one allows you to select function or column field formats.

You can specify a columnar layout for the text by selecting *Columns* from the *Option* menu. This will bring up a dialog allowing you to specify the number of columns for the rich text field, as well as the spacing between the columns.



*Rich Text Column Options Dialog*

You can add in-line images to the text flow in rich text fields as well. To add an image, select *Insert Image* from the *File* menu. This will bring up a dialog prompting you to specify the image file you want to import. After you specify it, a new dialog will open, allowing you to specify the size for the in-line image in pixels. After you set the size, click the *Ok* button and the image will be added to the rich text field.

You can import any rich text file (.rtf format) into a rich text field. To do this, select *Import* from the *File* menu in the editor. This will bring up a dialog prompting you to specify the file you want to import.



**Note**

Certain paragraph settings will be lost when you import a rich text file.

#### 4.1.3.7.7.1. Adding Formulas

You can add any formula directly into the text flow in a rich text field. It is also possible to add column fields and parameter values. To insert a formula, column field or a parameter value, click on the *Insert* menu, select a category (column, formula or parameter), and choose an item from the list of all available fields/parameters/formulas. The following syntax will be added into the text: `<% formula name/parameter name/column field name %>`.

The *Insert* → *Formula* list contains all formulas from the report. You can also create a custom formula directly in the rich text. To do so, simply type the `<%Formula Syntax%>` syntax (replace the *Formula Syntax* by the actual formula syntax). For more information about formula syntax, please see Section 4.1.6 - Using Formulas & the Formula Builder. This approach is recommended for simple formulas only. If you want to insert a more complex formula, you can create it in the report using the Formula Builder and then add it to the rich text field by selecting it from the *Insert* → *Formula* menu. See Section 4.1.6 - Using Formulas & the Formula Builder chapter to learn how to create formulas in the Formula Builder.

To format formulas and column fields, you must first pre-define a format. To do this, select *New* from the editor *Format* menu and select the data type for which you want to create a format. This will bring up the formatting dialog for that data type, allowing you to define the format. After you close the formatting dialog, you will be prompted to specify a name for the formatting. For example, you could define a format called *Currency* that uses locale-specific currency formatting for numbers.

To apply a pre-defined format of a formula, select the formula and then select the format that you want to apply from the drop-down box on the right side of the rich text editor.

#### 4.1.3.7.8. Hyperlinks

You can apply a hyperlink to any report element except column fields. To apply a hyperlink to an element, select it and then select *HyperLink* from the *Format* menu or right click on the element and select *HyperLink* from the pop-up menu. A dialog box will then pop-up asking you to enter the link, hint, and target. You can enter any URL as the link or another .pak file. The hint will display text in a hint box on mouse over when the report is viewed.

If you are running ReportDesigner in stand-alone mode, then hyperlinks to URLs will not work from the Preview window. If you export to DHTML or PDF, links to .pak files will no longer work.



*Hyperlink Dialog*

You can also apply dynamic hyperlinks to report elements, including column fields, by using cell scripting. For more information about cell scripting, please see Section 4.1.7 - Scripting.

#### 4.1.3.7.9. Column Wrapping

This feature is useful for reports that have few or narrow columns of data that are longer than one page. Rather than have the data take up only a portion of the page width and extend over multiple pages in length, you can 'wrap' the columns so that they will continue to the right of the original columns on the first page of the report.

For example, say you have the following report listing product names, and the number of units ordered:

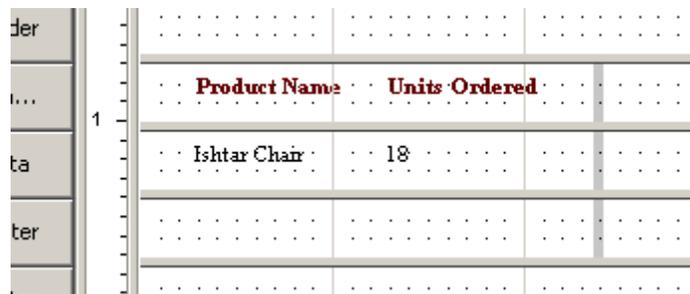
Product Name	Units Ordered
Adad Chair	4
Addad Dresser	5
Amon Table	2
An Chair	5

Product Name	Units Ordered
Anahita Dresser	3
Anubis Table	7
Apep Table	10
Apsu Dresser	3
Asherat Dresser	2
Atun Table	4
Bast Table	4
Bes Table	3
Cula Chair	10
Enki Chair	12
Enlil Chair	7
...	...
...	...

Assuming this report is long enough to encompass multiple pages, it is ideal to use column wrapping because it uses fewer pages and fills the blank space in the page width. The report will look like this with column wrapping:

Product Name	Units Ordered	Product Name	Units Ordered
Adad Chair	4	Bast Table	4
Addad Dresser	5	Bes Table	3
Amon Table	2	Cula Chair	10
An Chair	5	Enki Chair	12
Anahita Dresser	3	Enlil Chair	7
Anubis Table	7	...	...
Apep Table	10	...	...
Apsu Dresser	3		
Asherat Dresser	2		
Atun Table	4		

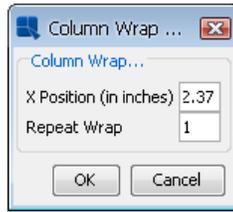
Column wrapping is available for simple columnar and summary break report types. To implement column wrapping, select *Column Wrap* from the *Format* menu. A gray vertical bar will be drawn across the middle of the Design window indicating the column wrapping placement.



Column Wrapping in Design Window

Column wrapping begins immediately to the right of the bar, so be sure to allow adequate space. Also, anything to the right of the column wrap bar will be truncated.

You can edit the column wrap properties by first selecting the column wrap bar and then selecting *Column Wrap* from the Format menu. You can also right click on the column wrap bar and then select *Column Wrap* from the pop-up menu. A dialog box will then appear allowing you to edit the properties of the column wrapping.



*Column Wrapping Options*

From this dialog, you can specify the X position of the column wrapping, as well as the number of times you want the columns to wrap in the page. The X position is the distance (in inches or centimeters) from the right side of the page where column wrapping is to begin. By default, column wrapping occurs once. Setting *Repeat Wrap* to **-1** will cause the wrapping to occur as many times as can fit within the page width.



**Note**

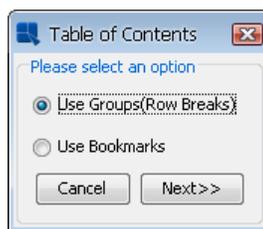
You can not specify column wrapping to occur more times than can fit within the page width. For example, if you specify three times, and the resultant report would be wider than the page, it will only repeat the wrapping twice.

**4.1.3.7.10. Table of Contents**

ERES has an ability to generate a table of contents for your report. The table of contents can either show the defined groups in the report, or a list of user-defined bookmarks. The TOC will display in Report Viewer, Page Viewer, DHTML, and PDF exports.

**4.1.3.7.10.1. Adding a Table of Contents**

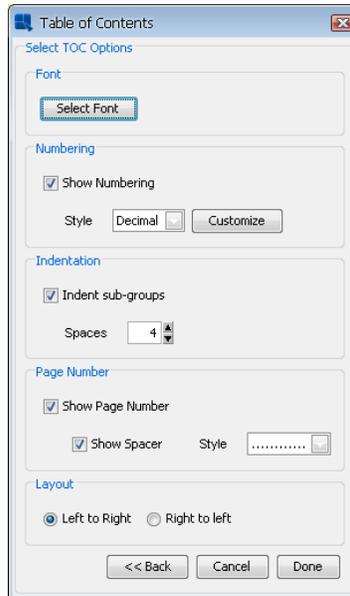
To add a table of contents to a report, select *Insert Table of Contents* from the *Insert* menu. Your cursor will turn into a cross. Click to place the table of contents in either the Report Header or Table Header sections. You can not add it to any other section. A dialog will open asking you if you would like to use groups or bookmarks for the table of contents.



*TOC Type Dialog*

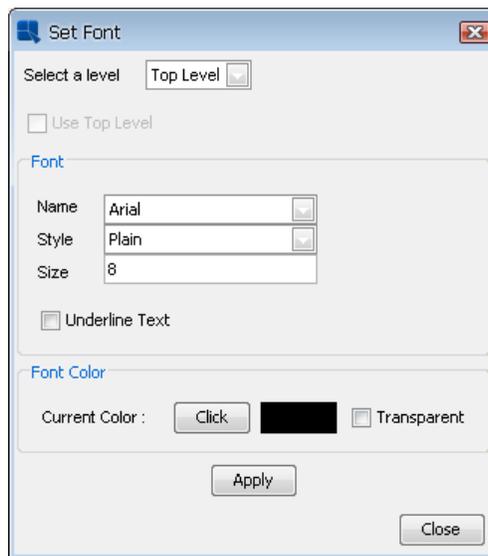
If you select to use groups, the table of contents will generate an entry for each group (and sub-group) in the report. In order to use this option, your report must contain grouped data (i.e. summary break, crosstab, master & details reports) with detail records. A crosstab report with one row break or a summary break report with column aggregation may be grouped, but because there are no detail records, you cannot create a table of contents for this presentation. If you select to use bookmarks, a one-level table of contents will be generated for each bookmark that is defined in the report. Bookmarks are defined using cell scripts. For more information about setting bookmarks, please see Section 4.1.7.2.1 - Formatting Actions.

The next dialog that appears allows you to specify options for the table of contents.



*TOC Options Dialog*

The first option in this dialog allows you to set the font for each level of the table of contents. When you click the *Select Font* button, a new dialog will open that allows you to specify font options.

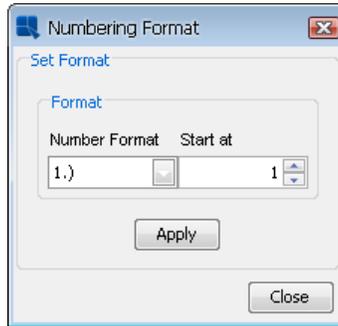


*TOC Font Dialog*

For each level you can set the font face, style, size, and color. Note that a level corresponds with a level of nested grouping in the report. If you select to create a report with bookmarks, only the top level will be available. Sub-levels can inherit the font from the parent level. After you make changes to the font for a particular level, click the *Apply* button to save the changes.

The next option allows you to select whether to show numbering for the table of contents and which style to use. You can customize each style by clicking the *Customize* button. The following styles are supported:

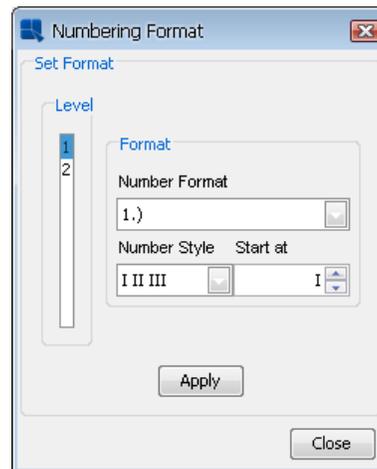
**Decimal:** This will use a numbered configuration for the table of contents entries with whole numbers for the outermost group and decimals for each sub-group. If you click the *Customize* button for this option, the following dialog will appear:



*Customization Options for TOC Decimal Format*

The first drop-down list allows you to select how the number should be formatted. The second option allows you to select the starting number. After you make your choices, click the *Apply* button to apply the changes.

**Outline:** This option will use an outline format allowing you to select numbers, letters, or Roman numerals for each level of the table of contents. If you click the *Customize* button for this option, the following dialog will appear:



*Customization options for TOC Outline Format*

The list on the left side allows you to select the level for which you would like to apply formatting. The first drop-down allows you to select the number format (this will apply whether you select numbers letters or Roman numerals for the level). The second drop-down allows you to pick the format for that level - either numbers, capital letters, lower-case letters, capital Roman numerals, or lower-case Roman numerals. The last option allows you to select the starting point for the selected style. After you make your choices, click the *Apply* button to apply the changes.

**Bulleted:** This option will use bullets for each entry. There are now additional customization options for this format.

The next option allows you to specify indentation for sub-groups. You can turn on/off the indentation and specify the number of spaces to use.

The next option allows you to specify whether to show the page number in the table of contents and whether to draw a spacer to the page number. You can draw a dotted line or a solid line for the spacer.

The last option in the dialog allows you to set the layout for the table of contents, either right to left, or left to right. After you finish setting options for the table of contents, click the *Done* button and it will be added to the report.

In the report, the table of contents will appear as a small grey rectangle. By default, the *resize to fit content* option will be turned on, but the width of the table of contents will be based on whatever size you allot for it in the section. Generally, you want to make this at least as wide as your report.

The screenshot shows a report design window titled 'RPT\_HDR'. The report is titled 'Sales Report' and includes a 'Table Of Contents' section. The table of contents is a table with columns for 'Region', 'Category', 'Product', 'Orders', 'Units Sold', and 'Total Sales'. The data is as follows:

Region	Category	Product	Orders	Units Sold	Total Sales
East	Arm Chairs	-Shimaliya Chair	3	33	13,992
<b>Category Total:</b>			<b>SUM((Orders))</b>	<b>SUM((Units</b>	<b>SUM((Total</b>

*TOC in Design Window*

You can see the complete table of contents when you preview the report. The table of contents is only supported for the DHTML and PDF exports. It will not appear if you export the report to other formats.

The screenshot shows a report preview window titled 'EspressReport 6.6'. The report is titled 'Sales Report' and includes a 'Table Of Contents' section. The table of contents is a table with columns for 'Region', 'Category', 'Product', 'Orders', 'Units Sold', and 'Total Sales'. The data is as follows:

Region	Category	Product	Orders	Units Sold	Total Sales						
1 East	11 Arm Chairs	12 Double Dressers	13 Oval Tables	14 Rectangular Tables	15 Round Tables	16 Side Chairs	17 Single Dressers	18 Triple Dressers	2 Midwest	21 Arm Chairs	22 Double Dressers

*TOC in Preview Window*

#### 4.1.3.7.11. Moving and Resizing Report Elements

There are several ways to move and resize report elements. You can use rulers, mouse, or you can manually enter the element bounds.

##### **Rulers:**

You can resize and move elements using the rulers located in upper left corner of the Design window. When you select a report element by clicking on it with your mouse, a shaded area will appear on each ruler, marking the element's horizontal and vertical bounds.

On the ruler at the top of the Design window, clicking and dragging the left-hand marker will move the element horizontally. Clicking and dragging the right-hand marker will stretch the element horizontally.

On the ruler on the left side of the Design window, clicking and dragging the top marker will move the element vertically. Clicking and dragging the bottom marker will stretch the element vertically.

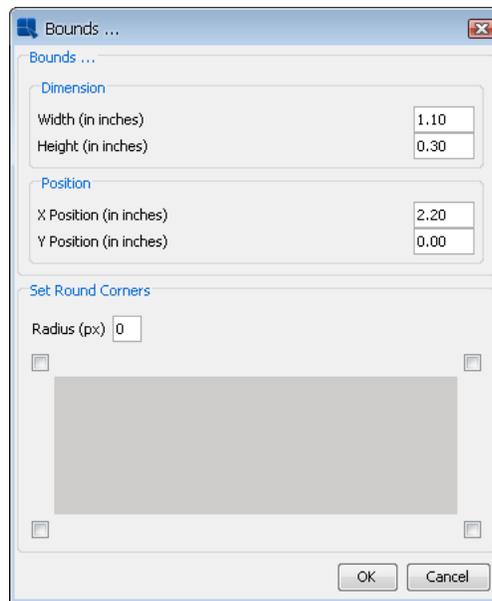
**Mouse:**

You can resize and move elements using your mouse. To move an element, simply click and drag it. To resize an element, click and drag on any of the sizing handles that appear around the edge of an object when it is selected. You can also right click and drag within the cell to resize it.

**Manually Set Bounds:**

You can manually set the bounds for any report element by clicking the  *Bounds* icon on the toolbar, or by selecting *Bounds* from the *Format* menu (You can also set bounds via the *Edit Attributes* dialog that you can open from the *Format* menu or by right-clicking an object and selecting it from the pop-up menu). A dialog box will appear prompting you to enter the new element bounds. The measurements will be in inches or centimeters, depending on which unit is selected with the toggle button in the upper left corner of the designer window where the rulers meet. The X and Y coordinates are for each specific report section. The point that marks an element's X and Y position is in the upper left corner of the element.

From the same dialog, you can also set round corners for the selected element. To learn more about round corners, please see Section 4.1.3.7.6 - Font, Color, and Border Options.

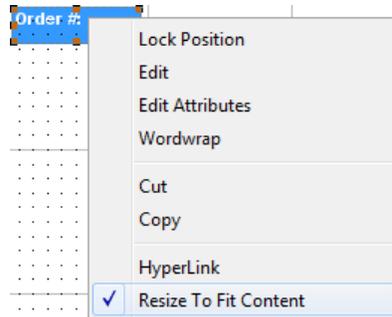


*Set Bounds Dialog*

**Resizing To Fit Contents:**

Often, you may have a column field that contains data of varying length. In this case the data may be truncated, or there may be a great deal of blank space within the cell when the field is short. To alleviate this problem, you can set the element to resize its height dynamically to encompass its contents.

To do this, first select the cell you want to resize and then select *Format* → *Resize To Fit Content*. The element will dynamically resize to fit its content. Note that only the height of the cell will resize. The width will remain the same.



*Dynamic Resize Dialog*

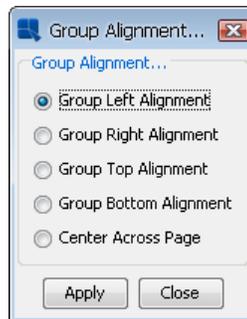
You can also invoke the section option called *Resize Cells Proportionally* to cause all of the other cells in the section to adjust their height with the resized cell. For more information about section options, please see Section 4.1.3.3 - Section Options.

**Group Move/Resize:**

You can move or resize a group of elements by selecting the elements first. Multiple elements in a report can be selected by drawing a selection box around them or by selecting them using **CTRL+Click**. You can draw a selection box by clicking and dragging on empty space in a report section. Once the elements are selected, click and drag on any element in the group to move that group. Click and drag on one of the re-sizing handles to adjust the size of elements in the group.

**Group Alignment:**

You can align a group of elements to the left side, right side, top, or bottom of a group. Select a group of elements and then right click on it. The *Group Edit* dialog box will appear giving you the option to align the group. Select *Group Left Alignment*, *Group Right Alignment*, *Group Top Alignment*, or *Group Bottom Alignment* and click the *Apply* button. All of the elements in the group will move to the left, top, or bottom edge of the group. You can also select the *Center Across Page* option. This will take the selected group of cells and center them in the page.



*Group Options/Alignment Pop-up Dialog*

**Moving Elements Between Sections:**

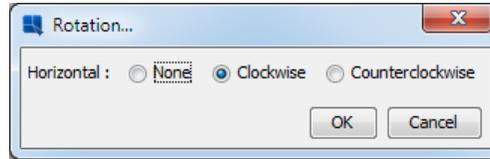
You can move a single element or group of elements between report sections by dragging them over the section boundaries. Note that elements cannot be dragged into the Table Data section.

**Column Swap:**

You can swap the position of two elements within a report section. This is primarily used to rearrange the position of column fields in the report. To swap two elements, select the elements using **CTRL+Click** or the selection box and then select *Swap Columns* from the Format menu. The positions of the two elements will be switched and the surrounding elements will be moved to accommodate the swap.

**Rotate:**

Text elements like labels, formulas, column headers, and column fields can be rotated 90 degrees clockwise or counter-clockwise. There are two ways to rotate elements. The first way to rotate an element is to right click on it and then move the mouse pointer over the *Rotate* option. A sub-menu will pop-up containing three options - *None*, *Clockwise* and *CounterClockwise*. The second way to rotate an element is to select it and then choose *Rotation* option from *Format* menu.



*Rotation dialog*

**Note**

Text rotation is available for DHTML, PDF, XSLX, and XLS export formats only.

**4.1.3.7.11.1. Controlling Element Overlapping**

In ReportDesigner, each element in a report has an associated Z index. This Z index number determines how elements behave when placed in the same space (on top of each other). When elements are placed in the same space, the element with the highest Z index will appear on top and all subsequent elements with lower Z indexes will be drawn successively below each other. To set the Z index for a report element, right click on it and select *Set Z-Index* from the pop-up menu. This will bring up a dialog allowing you to specify a number for the index of that element.



*Z Index Dialog*

**Note**

Certain export formats that require a tabular layout, like HTML and Excel, may not appear correctly if the report has elements that overlap each other.

**4.1.3.7.11.2. Locking Element Position**

In Report Designer, you can lock a cell position by right clicking on the element and selecting *Lock Position* from the pop-up menu. When this feature is enabled, the report element cannot be moved with mouse, rulers, or any of the group formatting or alignment features. It can only be moved when the lock position option is turned off.

**4.1.3.7.12. Snap to Grid**

By default, Report Designer operates in *Snap to Grid* mode. This forces report elements to move only in set increments. The grid layout is represented by small dots drawn in the Design window. By default, the grid step size is 0.1 in or 0.25 cm (depending on which measurement is selected).

To modify snap to grid options, select *Snap To Grid* from the Option menu. A dialog will open allowing you to set grid properties.



*Snap To Grid Options*

In this dialog, you can change the grid step size or disable the snap to grid feature. If you disable it, the report elements can be moved in free-form around the Design window.

#### 4.1.3.7.13. Shift Mode

Shift mode is toggled on and off by selecting *Shift On/Off* from the *Option* menu. When shift mode is enabled, report elements will move to accommodate changes in other report elements. Hence, if you resize the width of one report element from 1" to 2", the elements next to the current element will shift to the right. If shift mode is not selected, other report elements will not move, even if the resized element ends up overlapping another.

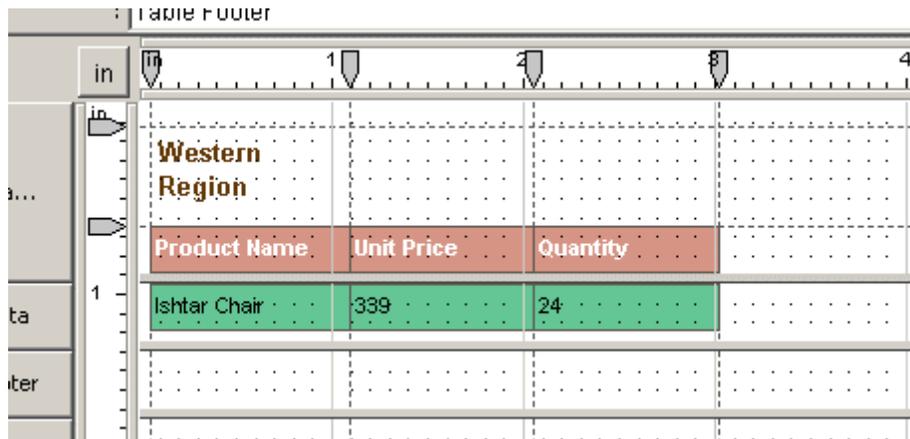


#### Note

Shift mode works best if you only resize elements or move them by a small amount. If you move elements a pronounced distance with shift mode enabled, it will displace other report elements by a pronounced distance as well, giving the report an odd appearance.

#### 4.1.3.7.14. Guidelines

Another way to position and move groups of elements is to use guidelines. Guidelines allow you to place arbitrary position lines within the Design window and snap report elements to those lines. Guidelines allow you to precisely line up and position report elements without worrying about performing precise movements.



*Elements Positioned with Guidelines*

##### 4.1.3.7.14.1. Inserting Guidelines

You can insert both horizontal and vertical guidelines by selecting *Insert Guideline* from the *Insert* menu. This option will open a sub-menu, prompting you to select whether you want a vertical or horizontal guideline. After you select your desired option, click in the design panel where you would like the guideline to be drawn. The guideline will appear as a dotted line with a marker in the upper or left ruler. You can move the guideline by clicking and dragging the marker in the ruler.

You can remove any guideline from a report by right clicking on the guideline (or the guideline marker in the ruler) and selecting *Delete* from the pop-up menu.



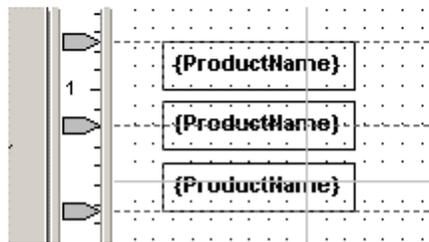
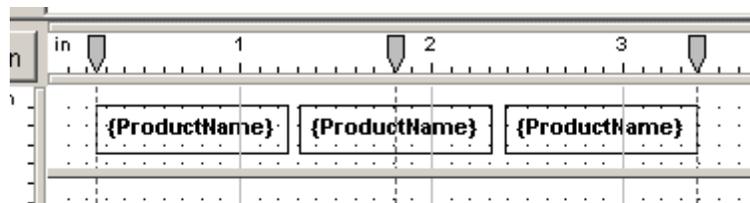
**Note**

Horizontal guidelines cannot be moved between report sections. If you wish to move your guideline to a new report section, you will need to insert a new guideline.

**4.1.3.7.14.2. Positioning Elements with Guidelines**

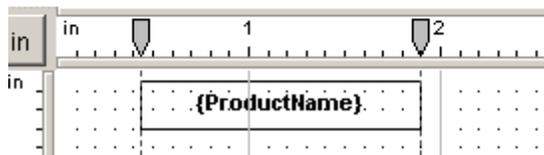
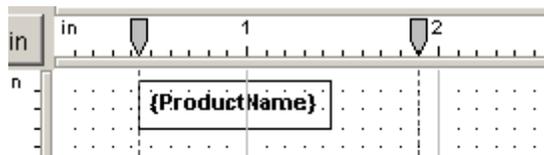
There are two ways in which objects can be snapped to guidelines. The first option is to snap all the elements that are near the guideline. To do this, right click on the guideline (or the guideline marker in the ruler) and select *Snap Elements in Range*. You can then further select to snap the cells left edge, right edge, or center. This will snap all the cells that are up to 0.1 inches away from the guideline.

The other option is to selectively snap elements to guidelines. To do this, first select the elements you want to snap to the guideline using **CTRL+Click** (you can select only one element, but you must use **CTRL+Click** to select it), then right click on the guideline (or the guideline marker in the ruler) to which you want to snap it. You can specify to snap the cell's left edge, right edge, or center to the guideline for both horizontal and vertical guidelines. Select the option you want from the pop-up menu and the cell(s) will snap to the guideline.



*Guideline Positioning Options*

Once an element is attached to a guideline, moving the guideline will cause the element to move as well. You can also snap an element to multiple guidelines. Snapping an element to two guidelines will cause it to stretch to fit.



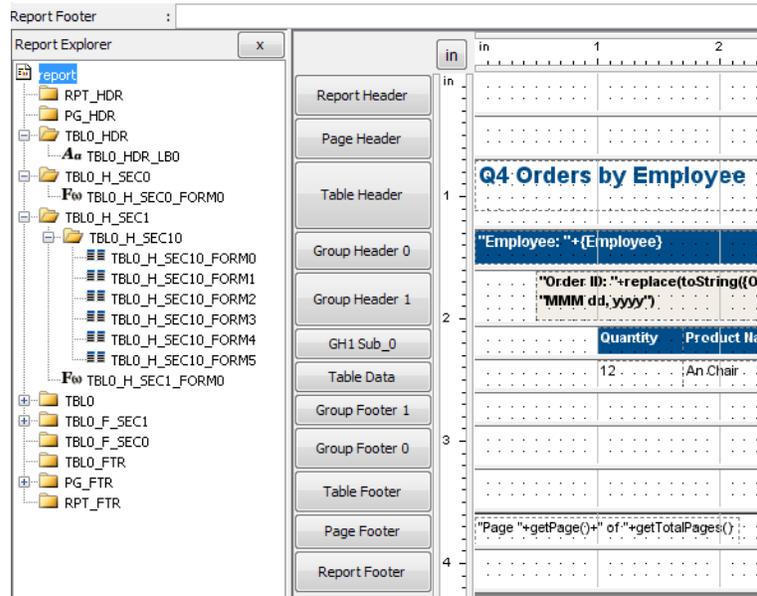
*Element Snapped to Two Guidelines*

When you have an element attached to two guidelines, moving either guideline will cause the element to resize (either stretch or shrink).

To release an element from a guideline, right click on it and select *Separate From Guidelines* from the pop-up menu. This will release the element from any guidelines to which it is attached. You can also release all of the elements attached to a guideline by right clicking on the guideline (or the guideline marker in the ruler) and selecting *Separate All* from the pop-up menu.

### 4.1.3.8. The Report Explorer

In addition to the main Design window, ReportDesigner provides another interface to select and edit the elements in a report - the Report Explorer. The Report Explorer displays all elements in the report as a tree structure on the left side of the Designer. It can be turned on and off by selecting *Report Explorer* from the *Option* menu.



Report Explorer Window

Each parent node in the tree represents a section in the report. Inside each section, an icon indicates element type and element ID. If you specified a custom ID for the element, it will display instead. For more information about custom IDs, see Section 4.1.6.2.1 - Using Column Field Data.

You can also edit report elements from within the Explorer. When you select an element in the tree, the element will also be selected in the Design window and the Design window will scroll in order to display that element on your screen. You can then use toolbars, menus, or pop-up menu to set any of the element properties. The pop-up menu can also be activated by right clicking on any element in the tree.

### 4.1.3.9. Global Formatting, Group Formatting, and Templates

To this point, you have only seen how to edit and format the properties of individual report elements. However, sometimes you may want to format multiple elements at once. To do this, you can use global formatting, group formatting, and templates.

#### 4.1.3.9.1. Global Formatting

The global formatting feature allows you to give all the elements in your report a consistent look and feel. To set the global formats, select *Global Format* from *Option* menu. This will bring up a second menu with each of the different report element types listed: chart, column, formula, image, label, line, rectangle, column header, and title. Selecting one of these will bring up a tabbed dialog box prompting you to set the formats for those elements. Each tab contains formatting options for an element attribute. The formatting options are as follows:

- Chart:** border color, border thickness, set bounds.
- Column:** data format, font style and size, font color, background color, border color, border thickness, dual colors, set bounds, alignment, rotation, script.
- Formula:** data format, font color, font style and size, background color, border color, border thickness, set bounds, alignment, rotation, script.
- Image:** border thickness, border color, set bounds, script.
- Label:** font color, font style and size, background color, border color, border thickness, set bounds, alignment, rotation, script.

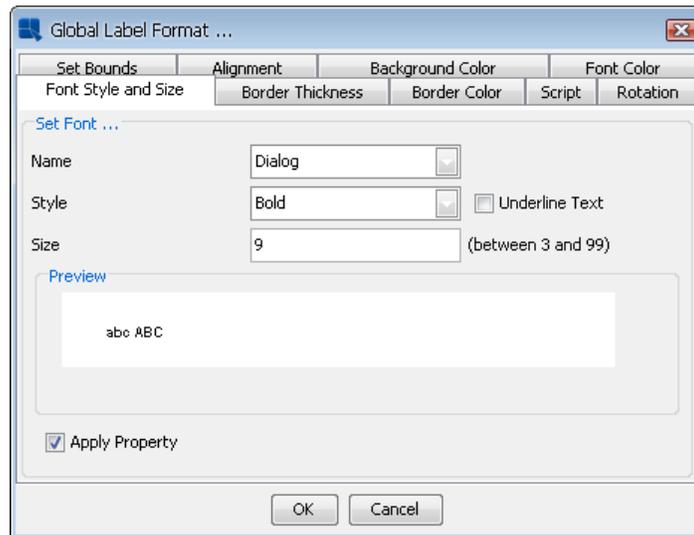
<b>Line:</b>	line color, set bounds, script.
<b>Rectangle:</b>	background color, border color, border thickness, set bounds, script.
<b>Column Header:</b>	font color, font style and size, background color, border color, border thickness, set bounds, alignment, rotation, script.
<b>Title:</b>	font color, font style and size, background color, border color, border thickness, set bounds, alignment.

At the bottom of each tab is a checkbox marked *Apply Property*. Checking this box will apply the property to the global formats when you click *OK*. The box automatically becomes checked when you change a property. After you make all the property changes, clicking the *OK* button will change the properties of all elements of that type in the entire report. It will also become the default attribute for any additional elements placed in the report.



**Note**

You have an option to insert images in the original size (default attribute). This can be done using the checkbox labeled *Default Size* in the *Set Bounds* tab of the image global format dialog.



*Global Format Dialog*

**4.1.3.9.1.1. Global Format Import/Export**

You can pass global formats from one report to another using the Import/Export feature. You can export global formats by selecting *Export* from the *Global Format* sub-menu. This will bring up a dialog box prompting you to specify a filename. The global formats will then be saved as an XML file. You can load a global format XML file by selecting the *Import* option from the *Global Format* sub-menu. This will bring up a dialog box prompting you to specify the XML file you want to import. Click the *OK* button and the formats stored in the XML file will be applied to the current report.

You can also specify this global format XML export as the default look and feel for blank reports by setting the corresponding server option. For more information about this feature, see Section 1.4.1.3 - Server Options.

**4.1.3.9.1.2. Global Formatting & Crosstab Reports**

Global formatting is very important when dealing with crosstab reports. Due to the nature of the report, the number of columns can increase or decrease depending on changes in the data, data source, or filtering criteria. When new columns are generated in a crosstab report, they will inherit the default formatting. In order to control the appearance of new columns in a crosstab report, you will need to set the default formatting using global formats.

Global formatting is also important because it controls the intervals in which new columns appear in the report. New columns will appear in intervals defined by the column width in the global formats. Therefore, if you set the

default width to be inch, new columns in the crosstab report will appear at one inch intervals. Because of this, it is recommended that you maintain consistent widths for the data columns in a crosstab report. Setting different widths can result in unexpected behavior when new columns are added to the report.

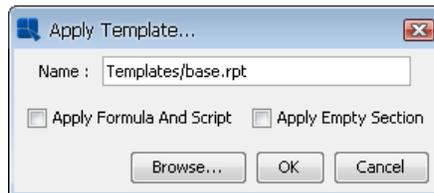
#### 4.1.3.9.2. Group Formatting

To apply formatting to a group of elements, first select a group of elements either by drawing a selection box around them or using **CTRL+Click**. You can also select a row or column of cells by right clicking on a cell and selecting the *Select Column* or *Select Row* option from the pop-up menu.

Once you select the group, format the properties as you would for a single element. Formatting will only take effect on elements where it is applicable. For example, if you select four elements, two labels, a formula, and a chart, and then change the font size, it will have no effect on the chart.

#### 4.1.3.9.3. Applying a Template

The features and elements of any report (.pak, .grp, .rpt or .xml) file can be applied to another report. This is accomplished using the apply template feature. To apply a template to the current report, select *Apply Template* from the *File* menu or click on the *Apply Template Button* on the toolbar. A dialog box will appear, prompting you to select the file you want to apply. When you apply a template, all of the elements and their respective formatting will be applied to the current report. Only the data source information is not carried over (this includes functions and scripts in the table data section of the report).



*Apply Template Dialog*

There are two additional options available in the dialog.

##### **Apply Formula and Scripts:**

This option will apply the formulaic columns and cell scripts from the template onto the report. By default, they do not apply. Note that formulas and scripts from the template may not work correctly if the data types of the columns in the new report are different from those in the template.

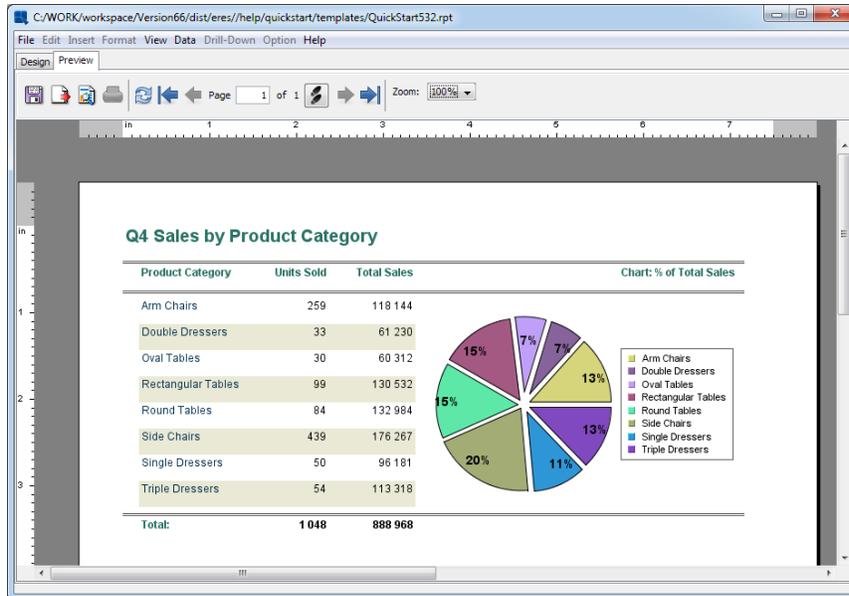
##### **Apply Empty Section:**

This option indicates whether or not to apply the blank sections from the template to the report. If you apply blank sections, they will overwrite sections in your report, essentially removing any elements you may have in the section. This option is generally used if you want to completely replicate the template that you are applying with the new report. If you do not apply blank sections, those sections in your report will not be overwritten. Only sections in the template with elements will be applied to your new report. This feature can be used if you have a certain default layout (headers/footers) that you want to pass among a group of reports. You can define a template where only the page headers/footers have defined elements and apply those headers/footers to other reports by selecting not to apply empty sections.

If you would like to have a report that has the same look and feel as another one without replacing the labels, formulas, etc, you can accomplish this by using the global format import/export feature rather than applying an entire template.

## 4.1.4. The Preview Window

After inserting or manipulating report objects, you can view the results in the Preview window. You can switch back and forth between the Design and Preview windows by using the tabs in the upper left corner of the Report Designer. The Preview window gives you an accurate picture of what the report will look like if it is printed or exported.



Preview Window

The Preview shows the page dimensions in inches or centimeters (depending on which unit is selected with the toggle button at the upper left corner of the designer window where the rulers meet). The drop-down menu on the right side of the toolbar allows you to set zooming for the preview.

#### 4.1.4.1. Navigating the Report

You can navigate around the report using the toolbar or the *View* menu. The toolbar buttons perform the following functions:

-  Go to the first page of the report
-  Go to the previous page of the report
-  Go to a specific page (that you enter in the *Page* text box)
-  Go to the next page of the report
-  Go to the last page of the report

These navigation functions can also be performed by selecting *First Page*, *Previous Page*, *Next Page*, *Last Page*, or *Go To Page...* from the *View* menu.

#### 4.1.4.2. Other Preview Window Options

Other options in the Preview window toolbar perform the following functions:

-  Save the file
-  Export the file
-  Load the entire report in the page viewer window. For more information about this, please see Section 4.1.4.2.4 - Using Page Viewer.
-  Print the report (this option is not available if the Report Designer is run as applet)



Refresh Data

In addition, the *File*, and *Data* menus remain active, allowing you to manipulate the file and data.

#### 4.1.4.2.1. Preview Data Options

You will be presented with the data options dialog the first time you preview a report. This dialog allows you to set several options related to how the Preview window displays report data.



*Preview Data Options Dialog*

The first option allows you to select whether or not to use live data when previewing the report. If you select live data, the report will connect to the data source and retrieve data every time the report is previewed. Note that if the data cannot be obtained (for example, if the database connection is down), the preview will show an error. You will then need to reload the report (once the connection has been reestablished) or go back to the *Preview Data Options* and select *Use Saved Data*. If you select to use saved data, the report will show whatever data it has when you preview the report. This could be either two records of back-up data that are stored in the template, the first twenty records that are retrieved when a report is first created, or the full dataset from the previous preview (if the option had been set to use live data).

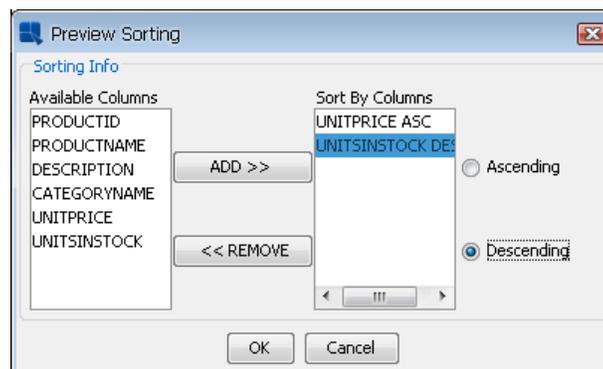
If you select to use live data to preview the report, a second option can be set that allows you to limit the number of records that should be retrieved from the data source. If your report uses a large dataset, this option allows you to see the report with data while limiting the processing time and preserving client memory.

Once you set this option the first time a report is previewed, you will not be prompted again when you preview the report. If you want to change these settings, select *Set Preview Data Options* from *Data* menu (this option is active only in *Design* tab). This will bring up the dialog again, allowing you to change the preview settings.

#### 4.1.4.2.2. Sorting Data

You can sort the data within the Preview window based on any column in the report in ascending or descending order. To do this, select '*Sort by (ascend)*' or '*Sort by (descend)*' from the *View* menu. Each option will bring up a secondary menu containing all of the report column fields. Selecting a column field will cause the report to sort by that column in either ascending or descending order. This feature is also available in the Report Viewer applet.

To sort by multiple columns, select *Sort By...* from the *View* menu. This will bring up a dialog allowing you to select which columns to sort by and the direction.



*Preview Sorting Dialog*

To sort by a column, select it in the left side dialog and click the *Add* button. You can set the sorting direction by selecting the column on the right side and clicking either the *Ascending* or *Descending* button.



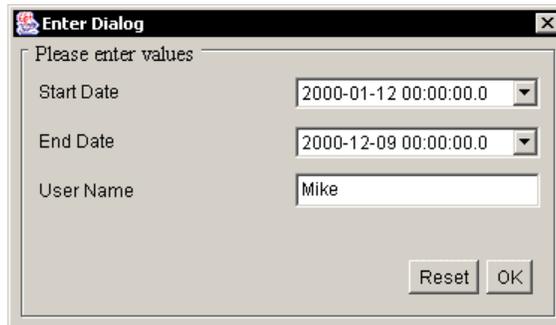
**Note**

Any sorting performed will not be saved with the template. When the template is re-opened (run), the data will use the original order. However, you can export the report after sorting and the exported file will reflect the re-ordered data.

It is not recommended that you use this feature with reports containing a large amount of data, as this will use a large amount of memory and can compromise performance.

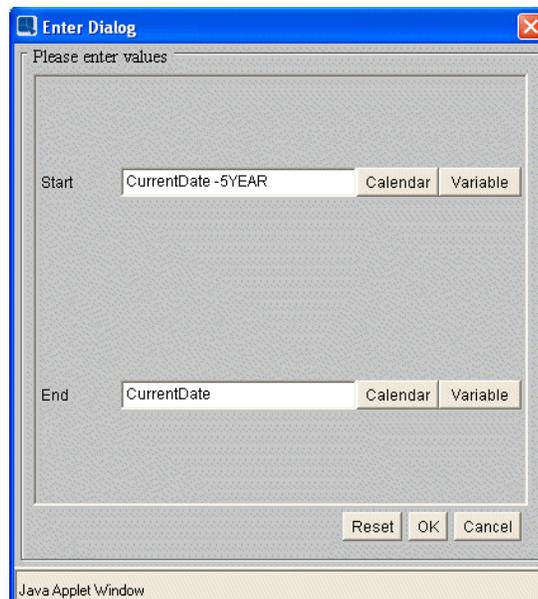
**4.1.4.2.3. Parameter Prompts**

If your report uses a parameterized query or if you have defined any formula parameters in the report, you will be prompted to select parameter values when you preview the report.



*Parameter Prompt Dialog*

You are prompted to type in or select parameter values depending on how the parameters are mapped. Clicking the *Ok* button will generate the report using the specified values. You can disable parameter prompting by toggling the *Preview Parameter Prompt* option in the Data menu. If date variables are not mapped to a database column, the parameter prompt will appear as below, with the options of entering a date from the calendar or a date variable.



*Date Parameter Prompt Dialog*

For more information about query parameters, see Section 3.1.3.2.2 - Parameterized Queries. For more information about formula parameters, see Section 4.1.6.2.6 - Formula Parameters.

#### 4.1.4.2.4. Using Page Viewer

Generally when you preview the report, the entire report with its data is kept in memory. This allows you to quickly preview and navigate through the generated report. However, for large reports, previewing the entire report could significantly affect system performance as the entire report is loaded into memory. To prevent this problem, you can limit the number of records that are fetched during preview as detailed in Section 4.1.4.2.1 - Preview Data Options.

On the other hand, you may want to preview the entire report without loading it into memory when creating an ad-hoc report. To do this, ReportDesigner allows you to load the report in the Page Viewer. To do this, select *Launch*

*Page Viewer* from the *View* menu or click the  *Launch Page Viewer* icon on the toolbar. This will open a Page Viewer window containing the report.

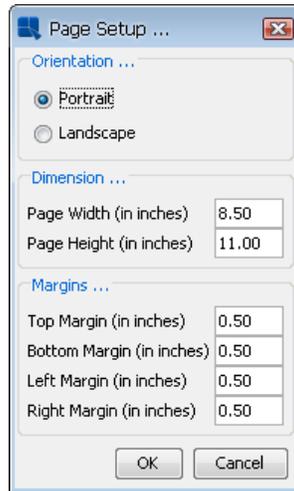


Report Shown in Page Viewer Window

The Page Viewer window will show you the entire report. You can navigate around the different pages by clicking the buttons on the toolbar at the bottom of the window or by right clicking and selecting to change pages from the pop-up menu. Because the Page Viewer is a static format, the window will not automatically reflect any subsequent changes made to the report. You will need to close the window and re-launch the Page Viewer. For more information about the Page Viewer, please see Section 7.7 - Page Viewer.

#### 4.1.4.3. Setting Page Properties

The output you see in the Preview window is based on the page dimensions and margins that are set in the Design window. The page dimensions are marked by the rulers, with the shaded areas indicating the page margins. To adjust these properties, select *Page Setup* from the *Option* menu. This will bring up a dialog box prompting you to set the page orientation (landscape or portrait), height, width, and margins. Measurements are in inches or centimeters, depending on which metric system is currently selected. Changing the orientation will also change the default print properties.



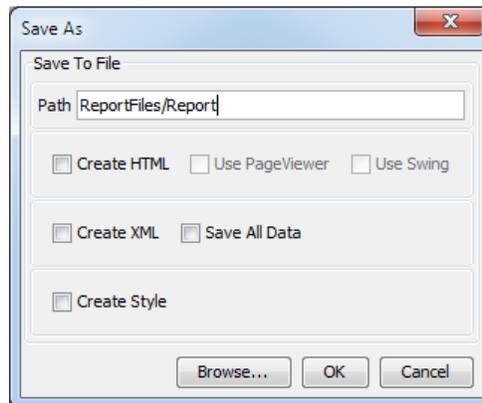
Page Properties Dialog

## 4.1.5. Saving and Exporting Reports

After you finish editing the report, you can save it as a template or export it to DHTML, PDF, CSV, Excel (XLS), Excel 2007 (XLSX), Text, XML, or Rich Text format.

### 4.1.5.1. Saving Reports

You can save the current report by selecting *Save* or *Save As* from the *File* menu, or by clicking the  *Save* button on the toolbar. All reports are saved as templates either in `.pak` (binary) or XML format. Selecting the save option will simply overwrite the existing file, unless you are working on a new report that has yet to be saved. Selecting *Save As* option will bring up a dialog box prompting you to create a file name. The *Save As* dialog also has several checkboxes.



Save As Dialog

**Create HTML:** This option will create a HTML page with the Report Viewer or Page Viewer applet embedded. The file will have the same name as your `.pak` file and it will be placed in HTML directory. By default, the Report Viewer applet is used. For more information about the Report Viewer, please see Section 7.6 - Report Viewer.

**Use Page Viewer:** Checking this option indicates that you would like to generate the applet page using the Page Viewer applet instead of the Report Viewer. For more information about the Page Viewer, please see Section 7.7 - Page Viewer.

**Use Swing:** This checkbox specifies whether or not to use the swing version of the Report Viewer or the Page Viewer depending on which viewer is selected for the applet page.

- Create XML:** This checkbox specifies whether or not to save the report in XML format. By default, reports are saved in binary (.pak) format. When this checkbox is checked, ReportDesigner will generate the template in XML format. The XML file can be saved and re-opened by the Designer. It can also be applied as a template for other reports.
- Create Style:** This checkbox specifies whether or not to save the report as a custom style. Reports saved in this format can be applied as a default look and feel to new reports. However, they cannot be used as a report anymore. For more information about this feature, see Section 4.1.2.6.1 - Custom Styles.
- Save All Data:** This option will save all of the report data in the template. By default, only two records are saved. This feature is useful if you want to preserve the structure of a crosstab when changing data sources or editing the report without the datasource. It also allows users to share a complete report when the data source is not available. Reports are always opened with backup data in the Report Designer. Users can get the latest data by selecting the "Live Data" option when previewing the report.

#### 4.1.5.1.1. XML Encoding

Both data registry files and report templates in ERES can be saved in XML format. By default, both types of files use the Western European encoding. In order to save data registries (including queries) and report templates correctly in XML, you will need to set the XML encoding to use a different character set. For more information about using this feature, please see Section 10.1.2.4 - XML Encoding.

#### 4.1.5.2. Exporting Reports

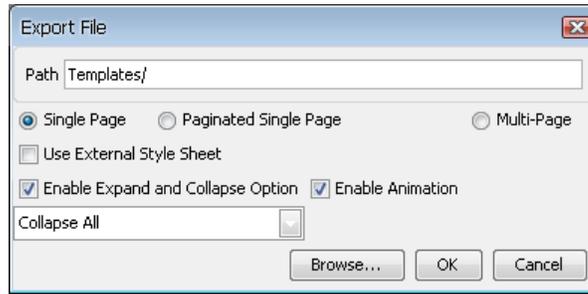
You can export the current report to multiple formats directly from the Report Designer. To export a file, select

*Export* from the *File* menu or click the  *Export* button on the toolbar. This will bring up a second menu allowing you to select the type of file you want to export to: DHTML, PDF, CSV, Excel (XLS), Excel 2007 (XLSX), text, XML, or rich text. You can also export it by clicking the *Export File* button on the toolbar.

**DHTML:** If you select DHTML as the desired export format, you will be prompted to enter the file name and location of the new DHTML file. Because DHTML is a text only format, charts will be saved as separate image files in the same directory as the DHTML file. If your report includes a chart, be sure to set the chart export format before exporting to DHTML, to specify the desired image format and attributes. If your report has images, the image files will be referenced in the DHTML document. DHTML will produce a much more accurate output than HTML; however, not all of the report features will translate accurately in older browsers. Internet Explorer 11, Chrome, Firefox, or Edge are recommended for viewing.

You can specify to export the report to one page, multiple pages, or as a paginated single page. If you specify one page, the entire report will be exported into one DHTML file. If you specify multiple pages, each page of the report will be generated in a new DHTML file. The output pages are linked by a navigation bar on top of each page. If you specify to export as a paginated single page, only one DHTML file will be generated. However, the page size, spacing, and margins will be retained, allowing the report to be printed out of the browser and appearing the same as if printed from a PDF or Report Designer.

You can also specify to use internal or external style sheet (CSS) definitions to apply formatting to the exported DHTML instead of generating style definitions for each report element. For more information about this option, see Section 4.1.5.2.1.1 - PDF Font Mapping Import/Export.



*Export to DHTML Dialog*

For reports with grouped data (except for Side-By-Side Master & Details reports), you can select to add expand and collapse controls for the grouping. This allows for a more compact presentation of the report. Each grouping can be expanded and/or collapsed depending on the viewing requirement. When selecting this option, you can also choose the initial presentation, i.e. show the report completely collapsed, completely expanded or expanded to a particular group level. You can also choose to enable animation wherein the grouping that is being expanded or collapsed fades in or out of view. Note that the expand and collapse controls can only be selected for single page exports and the controls will not appear if the Section Header or Footer is empty.

You can specify a title for the exported DHTML file. The title will appear as `<title>` meta elements in the export. To add a title, select *Html Page Title* from the *Option* menu. This will bring up a dialog allowing you to enter the title.

**PDF:**

If you select PDF as the desired export format, you will be prompted to enter the file name and location of the new PDF file. Because PDF format supports images, there will be no separate files created by the export.

You can also set encryption for PDF files. When you check the *Encrypt PDF Export* box in the export dialog, two new options will appear, allowing you to specify a user and an owner password for the generated document. More encryption options are available when exporting files through the API.

For more information about exporting to PDF, please see Section 8.1.5.7.2 - PDF Exporting Options.

**CSV:**

CSV export creates a text file of the data used in the current report with a comma field delimiter. If you select CSV as the desired export format, you will be prompted to enter the file name and location of the new CSV file, the *Delimiter* for the file (either Tab, Space, Double Space, Comma, or Semi-Colon), and the *Newline Delimiter* (either Windows(\r\n), Mac(\r), Others(\n), System).



**Note**

CSV export will only export the data associated with the report and none of the objects and attributes of the report itself.

**Excel (XLS):**

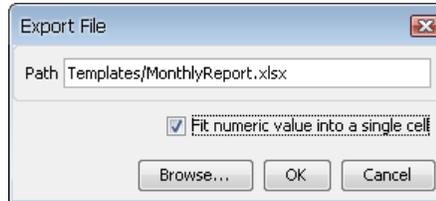
If you select an Excel spreadsheet as the desired export format, you will be prompted to enter the file name and the location of the new spreadsheet. ReportDesigner generates a formatted Excel output where spacing, fonts, colors, lines, and grids will be applied. Excel exports also include charts and images. Note that charts will be exported as images and thus are not editable in Excel. Any gif, jpeg, and/or png image will be shown in the Excel export. You can use the CSV export to get a plain data dump to Excel.

**Excel 2007 (XLSX):**

Excel 2007 (XLSX) export creates an XLSX file used in Microsoft Excel 2007 or newer. If you select an Excel 2007 as the desired export format, you will be prompted to enter the file name and the location of the new spreadsheet. ReportDesigner gen-

erates a formatted Excel 2007 output where spacing, fonts, colors, lines, and grids get applied. Excel 2007 exports also include charts and images. Again, the charts will be exported as images and thus are not editable in Excel.

If you plan to use Excel functions in the report, you will want to select the option *Fit numeric value into a single cell*. This option guarantees that each numeric data is exported to its own cell allowing you to use these values in Excel functions. Note that this option will sometimes affect the cell alignment of the exported report.



*Export to Excel Dialog*

For reports with grouped data, you can select to print each group on a different sheet in the exported file. This option is set in the Section menu for the Group Header section. For more information, please see Section 4.1.3.3 - Section Options.



**Note**

Since Excel spreadsheets are a fixed field format, not all report elements may translate correctly.

**TXT:**

Unlike CSV, which creates a comma delimited data file, the text export feature generates the entire report as a text file. Because the report is exported as pure text, no formatting information is retained. If you select text as the desired export format, you will be prompted to enter the file name and the location of the new text file, the *Delimiter* for the file (either Tab, Space, Double Space, Comma, or Semi-Colon), and the *Newline Delimiter* (either Windows(\r\n), Mac(\r), Others(\n), System).

**XML:**

There are two XML export versions. One is a data file and the other exports report data with formatting information.

**Pure Data:**

This XML export option exports the data used by the report in XML format. The format of the XML file is the same as the formatting requirements for XML input data.

**Data + Format:**

This XML export creates an XML version of the whole report. All of the report data as well as formatting information are included. This file is similar to the XML version of the DHTML export and can be used in conjunction with style sheets to display HTML content. This export allows you to export the entire report as one XML file or as a separate file for each page.

**Rich Text:**

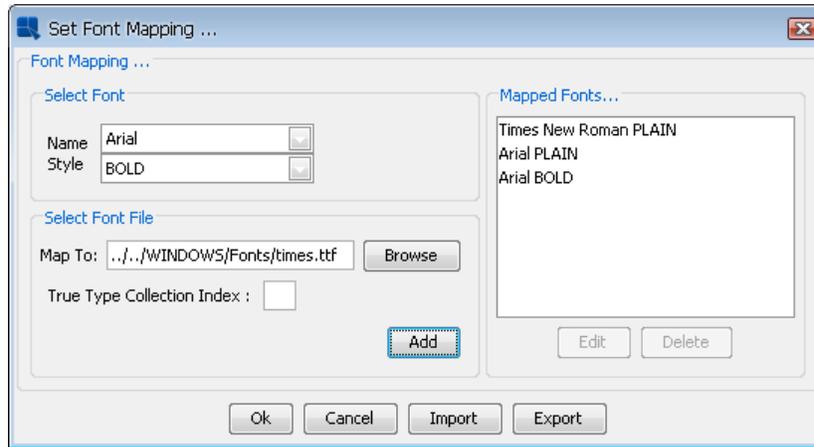
If you select rich text as the desired format, you will be prompted to enter the file name and the location for the new rich text file. ReportDesigner uses the full capability of the RTF 1.5 specifications to generate a rich text output very similar to the actual report. The rich text export may not work in all viewers. By default, ReportDesigner will export all the colors in the RTF export directly into the output. However, older versions of Microsoft Word only support 16 colors. To support these viewers, you can export the RTF using 16 colors. This option will convert the colors used in the report into the basic color palette for Microsoft Word 97.

**4.1.5.2.1. PDF Font Mapping**

ReportDesigner allows you to use any font on the system for reports. For most formatted exports (DHTML/Rich Text/Excel/Excel 2007) system fonts will translate automatically in the generated output (For Rich Text and Ex-

cel/Excel 2007, if the fonts cannot be found, they will default to Arial). However, for PDF you will have to manually specify a .ttf (true type font), .ttc (true type collection), .pfb, or .afm file for any system font that you would like to use in the report.

To set font mapping for the PDF export, select *Font Mapping* from the *Option* menu. This will bring up a dialog allowing you to specify font files.



*Font Mapping Dialog*

You can select a specific .ttf, .ttc, .pfb, or .afm file for each font and style combination. You can either type the full path or browse to the font file. If you are using a .ttc file, you will need to specify the font index in the box provided (.ttc files contain more than one font). Once you specify the correct file, click the *Add* button to save the mapping to the list. You can edit or delete existing mappings by selecting them in the list and clicking the appropriate button.

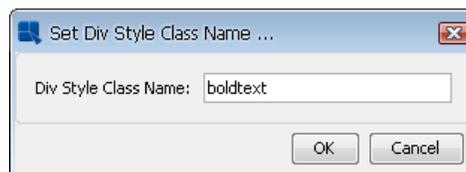
#### 4.1.5.2.1.1. PDF Font Mapping Import/Export

You can pass the font mapping from one report to another using the Import/Export feature. You can export the font mapping by clicking the *Export* button on the font mapping dialog. This will bring up a dialog box prompting you to specify a file name. The font mapping will then be saved as XML file. You can load a font mapping XML file by selecting the *Import* option from the dialog. This will bring up a dialog box prompting you to specify the XML file that you want to import. Click the *OK* button and the mapping stored in the XML file will be applied to the current report.

#### 4.1.5.2.2. CSS Options

When exporting a report to DHTML, the style definitions are created for each element in the report by default. This makes the export faster. However, it can result in a very large file size for large reports. To limit the size of the generated DHTML file, users can select to generate an internal list of global style definitions. This option is slightly slower, but it produces a significantly smaller file. In addition, ReportDesigner provides an option that allows users to apply their own style definitions to report elements by selecting a .css file when exporting the report.

If you are going to use external style sheet definitions, you will need to first select the class name from the external style definitions that should be associated with elements in the report. To set the class for a report element, right click on the cell and select *Set Style Class Name* from the pop-up menu. This will bring up a dialog allowing you to specify a style name for that report element.

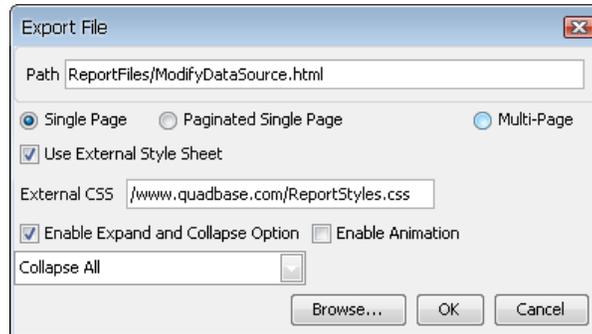


*Set Style Class Dialog*

Elements for which you do not select a style class will generate their own style based on the settings in the report when the report is exported to DHTML.

Users can also export a .css file for the formatting defined in the report. These style definitions are the same as generated by the internal style sheet option. This feature can be used to apply formatting created in the Report Designer to other reports or other web pages. To export a .css file from a report, select *Export Style Sheet* from the *Option* menu. A dialog will open, prompting you for the name and location for the generated file.

CSS options can be set in the export dialog for the DHTML export.



*DHTML Export Dialog*

The *Use External Style Sheet* option allows you to specify the location (either relative path or full HTTP path) of a .css file which contains the definitions for the style classes that you have assigned to report elements.



### Note

The formatting from an external style sheet file will only be applied to the DHTML export. Other export formats will use the formatting defined in the report template.

## 4.1.6. Using Formulas & the Formula Builder

ReportDesigner supports a wide variety of formulas, giving you an important mechanism for manipulating and displaying report data. Using formulas, users can add summaries and aggregations, perform basic calculations, and build complex expressions using one of over 70 built-in functions.



### Note

Some of the syntax for ReportDesigner formulas has changed greatly between v2.51 and v3.0. In most cases, the deprecated syntax will continue to work correctly. However, it is recommended that you check older templates to make sure the functions still return correctly if you have upgraded to v3.0 or higher from v2.51 or lower.

### 4.1.6.1. Creating a Formula

When you select to insert a new formula into the report (either by selecting *Insert Formula* from the *Insert* menu or clicking the  *Insert Formula* button on the toolbar), a list of all the defined formulas in the report will appear.



*Formula List Dialog*

To insert a formula into the report, select a formula from the list and click the *Insert* button. The formula list dialog will then close and you will be allowed to place the formula.

The report section in which you place the formula is extremely important, since the formula will reset each time that section repeats in the report. For example, if you place a formula in the Group Footer of a summary break report, the formula will recalculate for each group in the report. If the formula references a column value, it will only use data within each break group. If you place the same formula in the Report Footer section, it would only calculate once and it would use all of the data in the column field, regardless of breaks and grouping.

Formulas and labels that are placed in the Table Data section of the report become computed columns and can be treated as column fields as well as formulas.

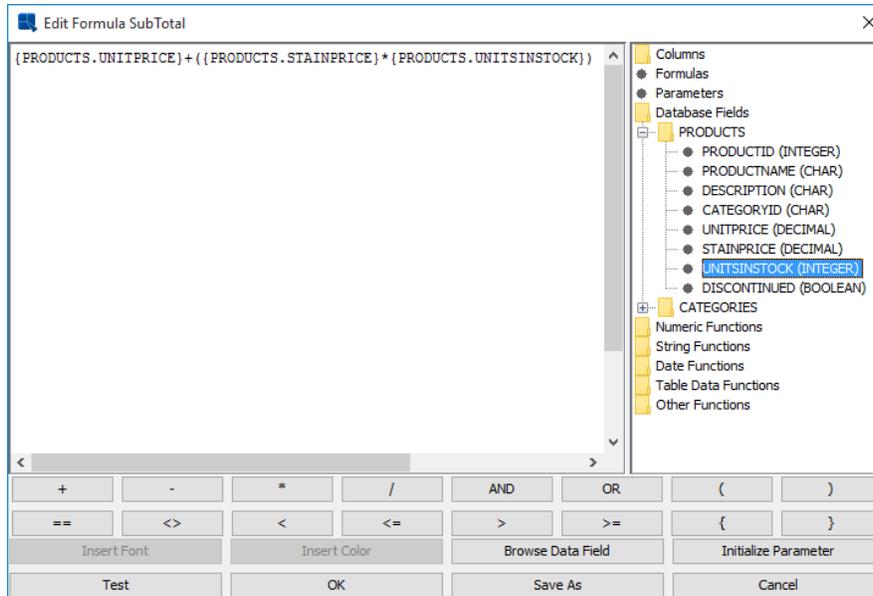


### Note

Labels are automatically treated as Strings and can be edited by double clicking on them (i.e. to use a number as integer in a formula, open the formula builder and remove double quotes from the number).

#### 4.1.6.1.1. The Formula Builder

Formulas are constructed using the Formula Builder interface. To launch the Formula Builder, you can select to create a new formula by selecting *New* from the formula list dialog. This will prompt you to specify a name for the new formula. Once you specify a name, the Formula Builder will open, allowing you to construct a formula. You can also use the Formula Builder to edit an existing formula. To edit a formula, select it from the list and click the *Edit* button or double-click on a formula cell in the Design window.



*Formula Builder Window*

The main window of the Formula Builder contains the text of the formula. The folders on the right side contain various elements that can be added to the formula, including column fields, other formulas, parameter values, database fields, and built-in functions. The first two rows of buttons contain the most common arithmetic and Boolean operators (operators and functions are discussed in the next section). The last two rows of buttons serve as command buttons and perform the following functions:

- |                              |   |
|------------------------------|---|
| <b>Insert Font:</b>          | This option is not available for formulas, it is only available for scripting.  |
| <b>Insert Color:</b>         | This option is not available for formulas, it is only available for scripting.  |
| <b>Browse Data Field:</b>    | This option is for database fields. It will query the database and return the first 20 values for the column.   |
| <b>Initialize Parameter:</b> | This option will bring up the parameter initialization dialog for any parameters defined in the formula. Parameters are discussed in Section 4.1.6.2.6 - Formula Parameters.  |
| <b>Test:</b>                 | This option will check the current formula and see if it is correct.  |
| <b>Ok:</b>                   | This option will close the formula builder and return to the formula list dialog, allowing you to insert the formula into the report.   |
| <b>Save As:</b>              | This option allows you to save the current formula under a different name. Because you can re-use a formula in different places in the report, using the save as option allows you to modify a formula in one place, without changing it for every place where the formula is referenced. |
| <b>Cancel:</b>               | This option will close the Formula Builder without saving the formula/changes.  |

## 4.1.6.2. Formula Syntax

The following section details various operators and functions available in ReportDesigner and how to use them.

### 4.1.6.2.1. Using Column Field Data

You will often want to use data from column fields for your formulas. This can be easily accomplished in formulas using the following syntax: {<Field Name>}. The braces delimit the field and the field name is the name of the column field you're referencing. So in a basic example: {UnitPrice} \* {Quantity} would multiply two column fields together.

In most cases, the column name is the name specified in the data source (i.e. the database column name or alias). If you are uncertain about the correct name, you can simply select the column field you want to use from the *Columns* node in the right panel in the formula builder.

There are two other functions you can use to retrieve column data:

**id()** This option will allow you to retrieve the value of any cell in the report, including column fields. The syntax is `id( <Object ID>)`. You can retrieve an object's ID by right clicking on the cell and selecting *Properties* from the pop-up menu. You can also retrieve the ID for column fields by selecting *View Column Mapping* from the Data menu.

You can also assign a custom ID to elements in the report, instead of using the default ID. To assign a custom ID to an element, right click on it in the Report Designer and select Custom ID from the pop-up menu. This will bring up a dialog allowing you to enter an ID for that element. The argument for the `id()` function can either be the original ID for the element or the defined custom ID.

Element IDs can also be used to get a handle to an element in the Report API. You can pass in either the original ID or the custom ID into the `getData()` method to get a handle to an element.

The `id()` function also serves another important purpose. It can be used to delay the calculation of computed columns. Normally, formulas in the Table Data section of the report will compute prior to any aggregations in the group or table footers. However, sometimes you may want to calculate the aggregations prior to the column fields. For example, you want to create a column that calculates each row's percentage of a total. In order to get the correct result, the total has to be calculated before the column. To do this, simply refer to the aggregation in the Table/Group Footer using the `id()` function. The finished function in the Table Data section would look like this: `{Quantity} / id(TBL0_FTR_FORM0)`.

**col()** This option is a deprecated function that is used in earlier version of ReportDesigner to retrieve data from column fields. Although this function is still valid, it is recommended that you use the new field notation. The syntax for this function is `col( Column Index)`. To view the column index, select *View Column Mapping* from the Data menu. This will display the assigned index number for each column field. Index values are assigned based on the order in which the columns are selected for the report. The first column has an index of 0.

#### 4.1.6.2.1.1. Column Aggregation

You can aggregate any column field (including computed columns) using one of the aggregation functions provided with ReportDesigner. The following aggregation options are available:

**average()** This option returns the mean value of the specified column. The syntax is `average(<Column Field>)`. To return the average value of a column field named `UnitPrice`, you can use the following formula: `average( {UnitPrice} )`.

**median()** This option returns the median value for the specified column. The syntax is `median(<Column Field>)`. If the median of the column is two values (for an even number of rows), the aggregation will return the average of the two values.

**count()** This option returns a count of the values in the specified column. The syntax is `count(<Column Field>)`.

**countdistinct()** This option returns a count of all the distinct values in the specified column. The syntax is `countdistinct(<Column Field>)`.

**max()** This option returns the maximum value for the specified column. The syntax is `max(<Column Field>)`.

**min()** This option returns the minimum value for the specified column. The syntax is `min(<ColumnField>)`.

**stddev()** This option returns the standard deviation for the specified column. The syntax is `stddev(<Column Field>)`.

**sum()** This option returns the sum of the specified column. The syntax is `sum(<Column Field>)`.

---

<b>sumsquare()</b>	This option returns the sum of squares for the specified column. The syntax is <code>sumsquare(&lt;Column Field&gt;)</code> .
<b>variance()</b>	This option returns the variance for the specified column. The syntax is <code>variance(&lt;Column Field&gt;)</code> .

It is important to note that aggregation functions can only take column fields as an argument. You cannot use an expression or another function as an argument. Therefore, `sum({UnitPrice}*{Quantity})` is not valid. To make this calculation, you would first need to add `{UnitPrice}*{Quantity}` as a formula in the Table Data section of the report. Then you can sum the computed column created by the first formula using `sum(@formulaname)`.

#### 4.1.6.2.2. Using Formulas

ReportDesigner allows you to easily re-use report formulas by plugging them directly into new formulas. You can refer to a formula using the following syntax: `@<FormulaName>`. The formula name is the name that was assigned to the formula the first time it was created. So for example if you created a formula `{UnitPrice}*{Quantity}` called `Total`, you could retrieve the result of this formula by typing `@Total`.

If you are uncertain about the correct name, you can simply select the formula you want to use from the *Formulas* node in the right panel of the Formula Builder.

You can also use this notation to aggregate computed columns if the formula has been placed in the Data Table section of the report.

#### 4.1.6.2.3. Using Parameter Values

In addition to column field and formula values, you can also access parameter values as part of a formula. Parameter values are the user supplied values to either query or formula parameters. For more information about query parameters, see Section 3.1.3.2.2 - Parameterized Queries. For more information about formula parameters, see Section 4.1.6.2.6 - Formula Parameters.

To refer a parameter value, use the following syntax: `<ParameterName>` for query parameters and `?<ParameterName>` for formula parameters. For example, if the report has a query parameter named `StartDate` that prompts the user to supply a date, `:StartDate` will return the date that the user has supplied when the report is run.

If you are uncertain about the correct name, you can simply select the parameter you want to use from the *Parameters* node in the right panel of the Formula Builder. Also, if you type a formula parameter name wrong, the Formula Builder will assume that you are trying to define a new parameter.

#### 4.1.6.2.4. Using Database Fields

If your report uses a database as the data source, you can also use database fields that were not selected for the report in a formula. Database fields are referenced in a similar manner to column fields and use the syntax `{<TableName>.<FieldName>}`. If your database requires three part names, the fields should be referenced accordingly.

You can select database fields to add from the *Database Fields* node in the right panel of the Formula Builder. In addition, the Formula Builder allows you to preview database fields. To do this, first select a field in the right panel and then click the *Browse Data Field* button. This will bring up a new dialog showing the first few rows from the selected column.

#### 4.1.6.2.5. Constants

When using constants in formulas, there are certain formats that are required for each data type.

**Numeric Data:** To represent a numeric constant, you simply have to type the number. Both integers and decimals can be accepted; however, you cannot type in thousands separators. So to specify the number 12.28 as a constant in a formula, type `12.28`.

**String Data:** To represent a string constant, type the string delimited with double quotes. For example, to specify a string Hello, you would type `"Hello"`.

**Boolean Data:** To represent a Boolean constant, type either `true` or `false` without any delimiters. To specify a false Boolean value, you would type `false`. Note that these are not case sensitive.

- Date Data:** To represent a date constant, type the date in the following format without delimiters: MM/dd/yyyy. To specify the date August 5th, 2002, you would type 08/05/2002.
- Time Data:** To represent a time constant, type the time in the following format without delimiters: hh:mm:ss. To specify the time 2:30 PM, you would type 14:30:00. Note that time must be in 24-hour format.
- Timestamp Data:** To represent a timestamp constant, type the date and time in the following format without delimiters: MM/dd/yyyy hh:mm:ss. To specify the time stamp August 5th, 2002 at 2:30 PM, you would type 08/05/2002 14:30:00.

#### 4.1.6.2.6. Formula Parameters

Rather than adding constants into a formula, you can specify a parameter. Using a parameter, you can collect constant values from the user (or elsewhere) at run-time and dynamically compute the formula based on those values. To specify a parameter in the formula, use a question mark “?” followed by the parameter name.

For example, the following formula `{Quantity}+?Value` would add a user supplied value to the `Quantity` field in a report.

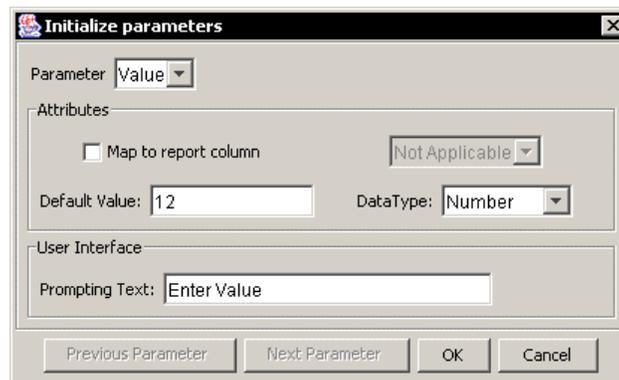


#### Note

If you want to define a new parameter within a formula, you have to use a unique name. If you specify a parameter name of a parameter that is already defined in another formula within the report, the new formula will return the value from the previous parameter.

#### 4.1.6.2.6.1. Initializing Formula Parameters

Like query parameters (detailed in Section 3.1.3.2.2.2 - Initializing Query Parameters), you must initialize a formula parameter before a formula can be used in a report. To initialize any parameters defined in the current formula, click the *Initialize Parameter* button in the Formula Builder. This will bring up a dialog prompting you to specify options for the parameter.



*Initialize Formula Parameter Dialog*

From this dialog, you can specify a default value for the parameter, map the parameter to a report column (this will give the user a drop-down list of distinct values, rather than having them type in the parameter), specify the data type for the supplied parameter, as well as specify the prompt for the end user. Clicking the *Previous Parameter* and *Next Parameter* buttons allows you to initialize each of the parameters that have been defined in the formula. Click *OK* when you finish setting up the options for all parameters.



#### Note

For formula parameters, you must specify a default value for each parameter. Also, if the parameter is mapped to a report column, the first time the report is previewed, the dialog will only contain the first twenty values of that column. This is because the full dataset is not retrieved when the report is first created. At run-time, the dialog will contain all the available values in the report column.

### 4.1.6.2.7. Operators

ReportDesigner provides several arithmetic and Boolean operators that allow you to create expressions in formulas. Operators use in-fix notation that places the operator between arguments in the expression, i.e. `<argument1> <operator> <argument2>`.

You can automatically insert operators into a formula by clicking one of the operator buttons in the formula builder.

#### 4.1.6.2.7.1. Arithmetic Operators

Four basic arithmetic operators are supported: +, -, \*, and, /. These will add, subtract, multiply, and divide two objects respectively. For example `1 + 2` will return 3.

Generally, arithmetic operators will take numbers as arguments and will return numbers with two exceptions. + can be used to concatenate strings, for example `Hello + "World"` would return `Hello World`. Also, - can be used to subtract two date objects and return the difference in days, for example `8/2/2002 - 4/7/2002` would return 117.

#### 4.1.6.2.7.2. Boolean Operators

Eight Boolean operators are supported: AND, OR, ==, <, <=, >, >=, and <>. AND and OR take Boolean arguments and return Boolean values. For example `true AND true` would return `true` or `1 < 2 OR 4 < 3` would return `true`.

==, <, <=, >, >=, and <> are comparison operators that signify equal to, less than, less than equal to, greater than, greater than equal to, and not equal respectively. Comparison operators compare two objects of the same data type and will return a Boolean value, for example `3 < 4` would return `true` and `Yes == "No"` would return `false`.

Boolean operators are more commonly used in cell scripting rather than formulas. Cell scripting is covered in Section 4.1.7 - Scripting.

### 4.1.6.2.8. Functions

In addition to the basic operators, ReportDesigner provides a number of built-in functions. Function syntax generally takes the form of a function name outside of a set of parenthesis enclosing the comma-separated arguments `function(<argument>, <argument>)`.

You can insert functions into the formula by selecting the function you want under the *Numeric Functions*, *String Functions*, or *Date Functions* nodes in the Formula Builder. Functions are inserted with hints indicating the type and number of arguments it should take. In order for the function to return properly, these hints have to be removed and replaced with valid arguments.

#### 4.1.6.2.8.1. Numeric Functions

The following functions take numeric arguments.

<b>abs()</b>	This option will return the absolute value of a number. The syntax is <code>abs(&lt;Number&gt;)</code> . For example <code>abs(-12)</code> would return 12.
<b>acos()</b>	This option will return the arc cosine of an angle, in the range of 0 through pi. Syntax for this is <code>acos(&lt;radians&gt;)</code> . For example, <code>acos(-sqrt(2)/2)</code> would return <code>3*pi/4</code> .
<b>asin()</b>	This option will return the arc sine of an angle in the range of -pi/2 through pi/2. The syntax is <code>asin(&lt;radians&gt;)</code> . For example, <code>asin(sqrt(2)/2)</code> would return <code>pi/4</code> .
<b>atan()</b>	This option will return the arc tangent of an angle in the range of -pi/2 through pi/2. The syntax is <code>atan(&lt;radians&gt;)</code> . For example, <code>atan(1)</code> would return <code>pi/4</code> .
<b>atan2()</b>	This option will convert rectangular coordinates (b, a) into polar coordinates (r, theta). This function returns theta by taking the arc tangent of b/a in the range of -pi to pi. The syntax is <code>atan2(&lt;y_coordinate&gt;, &lt;x_coordinate&gt;)</code> . For example <code>atan2(1, 0)</code> would return <code>pi/2</code> .
<b>ceil()</b>	This option will return the lowest integer value that is greater than the specified number. The syntax is <code>ceil(&lt;Number&gt;)</code> . For example, <code>ceil(15.3)</code> would return 16.

<b>cos()</b>	This option will return the cosine of an angle in radians. The syntax is <code>cos(&lt;radians&gt;)</code> . For example <code>cos(3*pi()/4)</code> would return $-\sqrt{2}/2$ .
<b>e()</b>	This option will return the value e, the natural logarithm base. The syntax is <code>e()</code> without any arguments.
<b>exp()</b>	This option will return e raised to the nth power. The syntax is <code>exp(&lt;Number&gt;)</code> . For example <code>exp(3)</code> would return $e^3$ or 20.086.
<b>factorial()</b>	This option will return the factorial of the specified object. The syntax is <code>factorial(&lt;Number&gt;)</code> . For example <code>factorial(3)</code> would return 6.
<b>floor()</b>	This option will return the highest integer value that is less than the specified argument. The syntax is <code>floor(&lt;Number&gt;)</code> . For example <code>floor(5.52)</code> would return 5.
<b>getColumnCount()</b>	This option will return an integer indicating the total number of columns in the report. The syntax is <code>getColumnCount()</code> without any arguments. This function returns the total number of columns whether they are visible or not.
<b>getRowIndex()</b>	This option will return an integer indicating the index value of the current row (of Table Data) of the report. The syntax is <code>getRowIndex()</code> without any arguments.
<b>getTotalRowIndex()</b>	This option will return a count of all the rows (in Table Data) in the report. The syntax is <code>getTotalRowIndex()</code> without any arguments.
<b>getRowIndexOfCurrentTable()</b>	This option will return the current row in the current group (table) in a report. This function is generally only relevant in reports that have grouped data like the summary break and master & details layout. For example, if the current row is the third row in a group of data, <code>getRowIndexOfCurrentTable()</code> will always return 2 regardless of what the row index is for the total report (which you could retrieve using the <code>getRowIndex()</code> function). The syntax is <code>getRowIndexOfCurrentTable()</code> without any arguments.
<b>getTotalRowIndexOfCurrentTable()</b>	This option will return a count of all the rows in the current group (table) in a report. This function is generally only relevant in reports that have grouped data like the summary break and master & details layout. The syntax is <code>getTotalRowIndexOfCurrentTable()</code> without any arguments.
<b>getSiblingCount()</b>	This option will return a count of the number of sibling groups within a nesting level. This function is generally only relevant in reports that have grouped data like the summary break and master & details layout. For example, you have a summary break report with one level of grouping. There are three distinct values in your row break column creating three groups at the level. Within the group (i.e. Group Header, Table Data, or Group Footer sections) <code>getSiblingCount()</code> would return 3. The syntax is <code>getSiblingCount()</code> without any arguments.
<b>getChildCount()</b>	This option will return a count of the number of child groups inside a nesting level. This function is generally only relevant in reports that have grouped data like the summary break and master & details layout. The function will return the number of child group in outer most group when running in a table header or a table footer; It will return the number of child in the related group when running in a group footer/header. For example, you have the same example as described for the previous function. Within the group, the <code>getChildCount()</code> function would return 0 as there is only one level of grouping. However, outside the group (i.e. Table Header or Table Footer

	sections) <code>getChildCount()</code> would return 3. The syntax is <code>getChildCount()</code> without any arguments.
<b>getGroupIndex()</b>	This option will return an integer indication the index value of the current group. This function is only relevant in reports that have grouped data like the summary break and master & details layout. This function will only return the index of the inner most group for reports with nested groups.
<b>log()</b>	This option will return the natural logarithm of the specified number. The syntax is <code>log(&lt;Number&gt;)</code> . For example <code>log(10)</code> would return 2.303.
<b>mod()</b>	This option will return the remainder after dividing two numbers. The first argument is the numerator and the second is the denominator. The syntax is <code>mod(&lt;Number&gt;, &lt;Number&gt;)</code> . For example <code>mod(12, 7)</code> would return 5.
<b>pi()</b>	This option will return the value pi. The syntax is <code>pi()</code> without any arguments.
<b>pow()</b>	This option will return the value of the first argument raised to the power of a second argument. The syntax is <code>pow(&lt;Number&gt;, &lt;Number&gt;)</code> . For example <code>pow(2, 3)</code> would return 8.
<b>random()</b>	This option will return a random value greater than or equal to 0 and less than 1. The syntax is <code>random()</code> without any arguments.
<b>rint()</b>	This option will round the specified argument to the nearest integer. The syntax is <code>rint(&lt;Number&gt;)</code> . For example <code>rint(5.6)</code> would return 6.
<b>sin()</b>	This option will return the sine of an angle in radians. The syntax is <code>sin(&lt;radians&gt;)</code> . For example <code>sin(pi()/2)</code> would return 1.
<b>sqrt()</b>	This option will return the square root of the specified number. The syntax is <code>sqrt(&lt;Number&gt;)</code> . For example <code>sqrt(64)</code> would return 8.
<b>tan()</b>	This option will return the tangent of an angle in radians. The syntax is <code>tan(&lt;radians&gt;)</code> . For example, <code>tan(pi()/4)</code> would return 1.
<b>toDegrees()</b>	This option will convert an angle measured in radians to degrees. The syntax is <code>toDegrees(&lt;Number&gt;)</code> . For example, <code>toDegrees(pi())</code> would return 180.
<b>toRadians()</b>	This option will convert an angle measured in degrees to radians. The syntax is <code>toRadians(&lt;Number&gt;)</code> . For example, <code>toRadians(180)</code> would return 3.142.
<b>toString()</b>	This option will convert a number into a string. The syntax is <code>toString(&lt;Number&gt;, &lt;Number of Decimals&gt;, &lt;Round Up(True/False)&gt;)</code> . For example <code>toString(12.216, 2, True)</code> would return "12.22" as a string.

#### 4.1.6.2.8.2. String Functions

The following functions take string arguments:

<b>getAllRowData()</b>	This option returns whole row of data in a format like <code>( COLUMN1_NAME ) VALUE1 ( COLUMN2_NAME ) VALUE2 . . .</code> Does not require any arguments.
<b>getHeader()</b>	This option will return the column name for a column field. The syntax is <code>getHeader(&lt;Column Field&gt;)</code> . For example, if you have a column named UnitPrice, <code>getHeader({UnitPrice})</code> would return UnitPrice as a string.
<b>getPage()</b>	This option will return the current page number. The syntax is <code>getPage()</code> without any arguments.

---

<b>getTotalPages()</b>	This option will return the total number of pages for the current report. The syntax is <code>getTotalPages()</code> without any arguments.
<b>getTotalSections()</b>	This option will return the total number of sections (i.e. the number of pages needed horizontally) for the current report. The syntax is <code>getTotalSections()</code> without any objects.
<b>indexOf()</b>	<p>This option will return the first index value where a specified pattern within a string occurs. There are two syntaxes for this function.</p> <ol style="list-style-type: none"> <li><code>indexOf(&lt;String&gt;, &lt;Pattern&gt;)</code>. For example <code>indexOf("Banana", "an")</code> would return 1.</li> <li><code>indexOf(&lt;String&gt;, &lt;Pattern&gt;, &lt;Starting Index&gt;)</code>. For example, <code>indexOf("Banana", "an", 2)</code> would return 3.</li> </ol>
<b>insert()</b>	This option allows you to insert new characters into a string. The syntax is <code>insert(&lt;String&gt;, &lt;Character Number&gt;, &lt;New Characters&gt;)</code> . For example <code>insert("Wood Natural Furniture", 4, "View")</code> would return <code>WoodView Natural Furniture</code> .
<b>lastIndexOf()</b>	<p>This option will return the last index value where a specified pattern within a string occurs. There are two syntaxes for this function:</p> <ol style="list-style-type: none"> <li><code>lastIndexOf(&lt;String&gt;, &lt;Pattern&gt;)</code>. For example <code>lastIndexOf("Banana", "an")</code> would return 3.</li> <li><code>lastIndexOf(&lt;String&gt;, &lt;Pattern&gt;, &lt;Starting Index&gt;)</code>. For example, <code>lastIndexOf("Banana", "an", 2)</code> would return 1.</li> </ol>
<b>replace()</b>	<p>This option allows you to replace one set of characters in a string with another. There are two syntaxes for this function.</p> <ol style="list-style-type: none"> <li><code>replace(String, Character Start Number, Character End Number, New Characters)</code>. For example, <code>replace("Today is a rainy day", 11, 16, "sunny")</code> would return <code>Today is a sunny day</code>.</li> <li><code>replace(&lt;String&gt;, &lt;Old Characters&gt;, &lt;New Characters&gt;)</code>. For example, <code>replace("Today is a rainy day", "rainy", "sunny")</code> would return <code>Today is a sunny day</code>.</li> </ol>
<b>setMaxLength()</b>	This option allows you to set the maximum length of a string. The syntax is <code>setMaxLength(&lt;String&gt;, &lt;Maximum Number of Characters&gt;)</code> . For example, <code>setMaxLength("You're a firefighter", 13)</code> would return <code>You're a fire</code> .
<b>strcmp()</b>	This option compares two strings and returns the lexicographical difference between them. If the first argument is larger than the second one, the result is positive. If the first argument is smaller than the second one, the result is negative. If the strings are same, the result is zero. The syntax is <code>strcmp(&lt;String&gt;, &lt;String&gt;)</code> . For example, <code>strcmp("a", "c")</code> would return -2.
<b>strcmpIgnoreCase()</b>	This option compares two strings and returns the lexicographical difference between them while ignoring case. The syntax is <code>strcmpIgnoreCase(&lt;String&gt;, &lt;String&gt;)</code> . For example, <code>strcmpIgnoreCase("a", "A")</code> would return 0.
<b>strcat()</b>	This option concatenates multiple strings together. The syntax is <code>strcat(&lt;String&gt;, &lt;String&gt;, &lt;String&gt;...)</code> . For example <code>strcat("Wood", "View")</code> would return <code>WoodView</code> . You can also use the "+" operator to concatenate strings.

---

<b>strlen()</b>	This option will return an integer indicating the number of characters in a specified string. The syntax is <code>strlen(&lt;String&gt;)</code> . For example <code>strlen("ReportDesigner")</code> would return 14.
<b>substring()</b>	This option will return a portion of a string as specified by an argument. There are two syntaxes for this function: <ol style="list-style-type: none"> <li>1. <code>substring(&lt;String&gt;, &lt;Character Start Number&gt;)</code>. For example, <code>substring("unhappy", 2)</code> would return happy.</li> <li>2. <code>substring(&lt;String&gt;, &lt;Character Start Number&gt;, &lt;Character End Number&gt;)</code>. For example <code>substring("smiles", 1, 5)</code> would return mile.</li> </ol>
<b>trim()</b>	This option will remove any leading or trailing spaces from the specified string. The syntax is <code>trim(&lt;String&gt;)</code> . For example, <code>trim(" Hello ")</code> would return Hello.
<b>toLowerCase()</b>	This option will render any uppercase letters within the specified string to lowercase. The syntax is <code>toLowerCase(&lt;String&gt;)</code> . For example <code>toLowerCase("ABCdef")</code> would return abcdef.
<b>toNumeric()</b>	This option will turn a string into a double. The syntax is <code>toNumeric(&lt;String&gt;)</code> . For example <code>toNumeric("425.52")</code> would return 425.52. For this formula to work correctly, the string argument must contain numeric characters.
<b>toUpperCase()</b>	This option will render any lowercase letters within the specified string to uppercase. The syntax is <code>toUpperCase(&lt;String&gt;)</code> . For example <code>toUpperCase("ABCdef")</code> would return ABCDEF.

#### 4.1.6.2.8.3. Date/Time Functions

The following functions use date/time arguments. Some of the date/time functions use a special argument that indicates a calendar field. To specify a calendar field argument, type one of the following without any delimiters.

ERA	DAY_OF_- MONTH	HOUR
YEAR	DAY_OF_YEAR	HOUR_OF_DAY
MONTH	DAY_OF_WEEK	MINUTE
WEEK_OF_YEAR	DAY_OF_WEEK_IN_- _MONTH	SECOND
WEEK_OF_- MONTH	AM_PM	MILLISECOND

<b>addTime()</b>	This option adds a specified amount of time to a given date/time. The syntax is <code>addTime(&lt;Date/Time&gt;, &lt;Calendar Field&gt;, &lt;Number&gt;)</code> . The number specifies the amount that the calendar field should be added to the date/time object. For example, <code>addTime(12/5/1998, MONTH, 5)</code> would return May 5, 1999, and <code>addTime(12/5/1998, DAY_OF_MONTH, -25)</code> would return Nov, 10 1998.
<b>getAmPm()</b>	This option will return AM or PM as a string for a given time. The syntax is <code>getAmPm(&lt;Time&gt;)</code> . For example, <code>getAmPm(13:24:00)</code> would return PM.
<b>getCurrentDate()</b>	This option will return the current date from the system. The syntax is <code>getCurrentDate()</code> without any arguments.
<b>getCurrentDateTime()</b>	This option will return the current date and time from the system. The syntax is <code>getCurrentDateTime()</code> without any arguments.

- getCurrentTime()** This option will return the current time from the system. The syntax is `getCurrentTime()` without any arguments.
- getDateTime()** This option will return the value of a specified calendar field for a given date/time. The syntax is `getDateTime(<Date/Time>, <Calendar Field>)`. For example, `getDateTime(12:24:00, MINUTE)` would return 24, and `getDateTime(10/10/2001, DAY_OF_WEEK)` would return 4 (meaning Wednesday).
- getDayDifference()** This option will return the difference in days as an integer between two dates. The syntax is `getDayDifference(<Date>, <Date>)`. For example, `getDayDifference(5/1/2001, 3/1/2001)` would return 61. You can also use the "-" operator to return the difference between dates.
- getDayOfWeek()** This option will return the day of the week for the specified date. The syntax is `getDayOfWeek(<Date>)`. For example, `getDayOfWeek(10/10/2001)` will return Wednesday as a string.
- getWeekOfYear()** This option will return the week of the year for the specified date. The syntax is `getWeekOfYear(<Date>)`. For example, `getWeekOfYear(10/10/2001)` will return 41.
- getEra()** This option will return the era for the specified date. The syntax is `getEra(<Date>)`. For example, `getEra(10/10/2000)` would return AD as a string.
- getMonth()** This option will return the month for the specified date. The syntax is `getMonth(<Date>)`. For example, `getMonth(10/10/2000)` would return October as a string.
- rollTime()** This option is a time rolling function. The syntax is `rollTime(<Date/Time>, <Calendar Field>, <Number>)`. Unlike the `addTime()` function, this function will only adjust the specified calendar field by the specified amount and will have no effect on the other fields. For example `rollTime(12/5/1998, MONTH, 5)` would return May 5, 1998, and `rollTime(12/5/1998, DAY_OF_MONTH, -25)` would return Dec 11, 1998. For the latter example, when the count reaches the beginning of the month, it starts over at the end without changing the month field.
- printDate(), printDateTime(), printTime()** These three functions allow you to set the way in which date/time information is displayed beyond what is capable using the data formatting options. These functions will return date and/or time information as a string. Their syntax is as follows:

```
printDate(Date, Date Format)
printDateTime(Date, Date & Time Format)
printTime(Time, Time Format)
```

The date and/or time format is entered as a series of characters and delimiters. Letters are used to represent different elements of date/time data. The characters and what each of them represent are listed below:

Character	Represents	Output (text/number)	Example
G	era	text	AD
y	year	number	1996, 96
M	month in year	text or number (dependence length)	July, Jul, 07
d	day in month	number	10

Character	Represents	Output (text/number)	Example
h	hour am/pm (1-12)	number	1
H	hour 24 hr. (0-23)	number	18
m	minute in hour	number	30
s	second in minute	number	55
S	millisecond	number	978
E	day in week	text	Tuesday, Tue
D	day in year	number	189
F	day of week in month	number	2 (as in 2 nd Wed. in July)
w	week in year	number	27
W	week in month	number	2
a	am/pm marker	text	AM, PM
k	hour 24 hr (1-24)	number	24
K	hour am/pm (0-11)	number	0
z	time zone	text	Pacific Standard Time, PST

You can piece together almost any combination of these characters to produce a date expression in any format you want. The count of groups of characters determines the form that the element will take. For text elements with four or more characters in a group, the full form of the element will be used. If less than four characters are used, the short form will be used, if it exists. For example EEEE would return Monday and EE would return Mon. For month M which can be displayed as either text or a number, four or more in a group will display the full version, three will display the abbreviation, and two or less will display the number form.

For numeric elements, the count of characters is the minimum number of digits that the element will take. Shorter numbers will implement leading zeros. For example if the day of the date is 2, dd would return 02 and d would return 2.

Any character that is not a-z or A-Z like “;”, “:”, “@”, etc can be inserted anywhere within the string expression and will display as they are entered. You can also insert words and expressions by enclosing them within single quotes (type two single quotes to insert an apostrophe as text). Several examples are listed below:

```
printDateTime(10/10/2001 21:15:12, "MMMM dd, yyyy 'at' hh:mm a z") would return October 10, 2001 at 9:15 PM PDT
```

```
printDate(10/10/2001, "EEE MMM dd'th', yyyy") would return Wed Oct 10th, 2001
```

```
printTime(13:22:12, "h 'o''clock' m 'minutes and' s 'seconds'") would return 1 o'clock 22 minutes and 12 seconds
```

```
printDateTime(10/10/2001 13:22:12, "MM/dd/yy HH:mm:ss") would return 10/10/01 13:22:12
```

If you do not enter any information for the date/time format, the date will display as yyyy-mm-dd and the time will default to HH:mm:ss.S. So `printDateTime(10/10/2001 13:22:12)` would return `2001-10-10 13:22:12.0`.

**toDate()** This option will convert a long integer into a timestamp. The syntax is `toDate(<Number>)`. For example, `toDate(1025039526306)` would return `Jun 25, 2002 2:12:06 PM`.

#### 4.1.6.2.8.4. Table Data Functions

The following functions take numeric arguments:

**.row()** This option will return a value of a column field according to the row index. To do this, you can append a column field with the following syntax: `.row(Row Index)`. The row index is the index value associated with each row in a report starting with 0. For example `{ProductName}.row(3)` would return the fourth value of the `ProductName` field.

**.getScriptValue()** This option will return value of a column field according to the row index after applying script. If there is no script applied on the column, it returns the same value as `.row()`. See Section 4.1.7.2.9 - Accessing Values Computed by Other Scripts for more details.

#### 4.1.6.2.8.5. Other Functions

The following functions take string arguments:

**first()** This option will return the first value of a column field. The syntax is `first(<Column Field>)`. For example `first({UnitPrice})` would return the first value of the `UnitPrice` field.

**last()** This option will return the last value of a column field. The syntax is `last(<Column Field>)`. For example `last({UnitPrice})` would return the last value of the `UnitPrice` field.

#### 4.1.6.2.8.6. Custom Functions

In addition to the functions provided with ReportDesigner, it is also possible to include your own custom functions in the Formula Builder when launching the Report Designer via API. Every Java function is supported. For more information about this feature, please see Section 8.1.5.8.6 - Open Report Designer with Custom Functions.

### 4.1.6.3. Subreport Formula Access

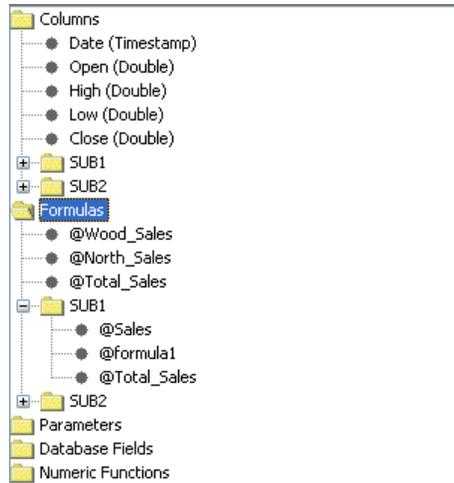
A main report can access its sub-reports' columns and formulas and use them as its own by using the prefix `SUB#` where “#” is the number of the subreport, beginning with 1. For example, the syntax to access the column `UnitPrice` from sub-report 1 is: `SUB1.{UnitPrice}` or `SUB1.COL(i)` where “i” is the column number of `UnitPrice`. Syntax to access the formula `Sales` from sub-report 1 is: `SUB1.@Sales`.



#### Note

Accessing a subreport column (i.e. `SUB1.{UnitPrice}`) returns a single value - the value from the first row of the column. You can get the value at a particular row by appending `.row(index)` at the end where “index” is the row number beginning with 0, but it will always only return a single value.

In the formula builder's help panel, under both Columns and Formulas, there is a folder for each subreport:



Sub-report columns and formulas can be used to build more complex expressions or scripts (i.e. `SUB1.@Total + SUB2.@Total`).

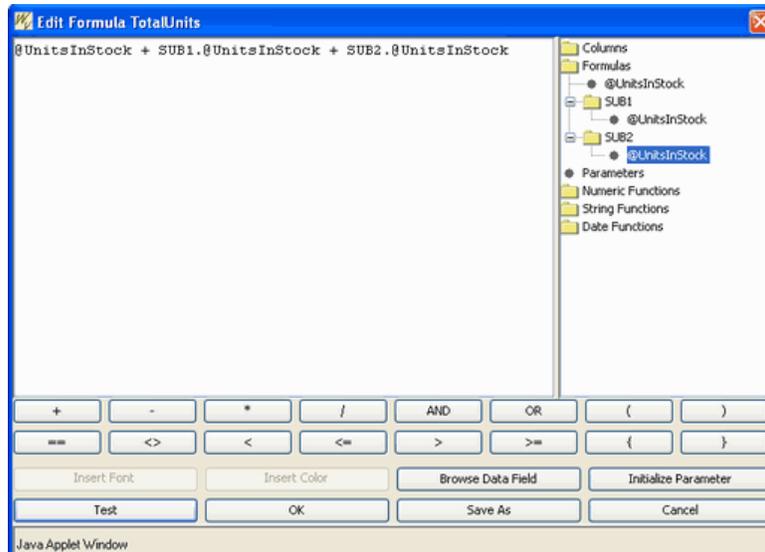
Although sub-report columns and formulas can be used in complex expressions, sub-report columns and formulas cannot be used in aggregation functions (SUM, COUNT, MAX, AVERAGE, etc). For example, `SUM({UnitPrice})` is a valid formula, but `SUM(SUB1.{UnitPrice})` is not. To accomplish this, instead create a formula `SUB1.{UnitPrice}` and add it as a column into the main report. Then create a second formula which applies aggregation to the first formula (i.e. `SUM(@sub_total)`).

Although the main report can access any of the sub-report's columns or folders, sub-reports cannot access the main report's or any other sub-report's columns or formulas.

If a sub-report formula or column that is used by the main report is deleted, the formula in the main report will display a NULL value when running the report and script will not be applied.

For example, if product information is spread across several databases, but you would like to sum up all the units in stock across the three databases, you could create one main report, add several sub-reports and then create a formula to insert to the report footer adding up all the units in stock. To demonstrate this feature, open a report created from the Inventory XML file with the fields Category Name, Product ID, Product Name, Unit Price, and Units In Stock. Proceed to add subreports to it, one from the Woodview Access database, and one from the Woodview HSQL database, containing the same fields.

Within each subreport, as well as within the main report itself, create a formula called `UnitsInStock`, `SUM({UnitsInStock})`. Then you can create an ultimate formula within the main report using these three functions:



*Creating a Formula Using Sub-report Formulas*

This formula will take the formula values from each of the three reports and add them together to calculate the final value.

This feature is also applicable to linked sub-reports, although the application of it is different. For example, if you create a summary break report containing customer addresses with a sub-report in the group footer detailing all purchases made by that particular customer, you can sum up all the sales info within the main report.

First, create a formula within the sub-report entitled `Sum_Sales` which sums up all the sales in the sub-report. Return to the main report and create a formula `sub_total`, which is simply `SUB1 . @Sum_Sales`. Insert it into the Table Data Section. It can be made invisible if you want to, but it must be in the Table Data Section. Then create a second formula in the main report, `total`, `SUM(@sub_total)`. This formula is added to the report footer and will sum up all the sales columns from all of the sub-reports.

## 4.1.7. Scripting

Cell scripting or conditional formatting allows you to dynamically modify certain object properties when specific conditions are met. Scripting allows you to highlight certain data values and to present a report that is easier to read. A simple example of this would be a report showing financial results or cash flows. Scripting could be applied to a column field so that negative numbers are shown in red, dynamically highlighting areas of revenue or cash shortages.

In ReportDesigner, scripts are element specific, meaning that in order to dynamically change the properties of a report cell, a script must be applied to that specific cell. Scripts can be applied to labels, column fields, formulas, or report sections.



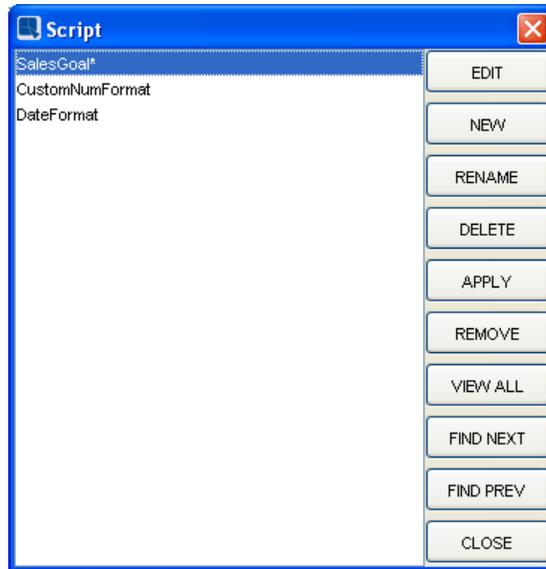
### Note

Some of the syntax for ReportDesigner cell scripts has changed greatly between v2.51 and v3.0. In most cases, the deprecated syntax will continue to work correctly. However, it is recommended that you check older templates to make sure the scripts still function correctly if you have upgraded to v3.0 or higher from v2.51 or lower.

### 4.1.7.1. Creating a Script

To create a script, first select a cell you want to apply the script to and then select *Scripting* from the *Format* menu

or click the  *Scripting* button on the toolbar. This will bring up a list of all the scripts that have been defined within the report.



*Script List Dialog*

Like formulas, scripts are maintained on a report wide basis and can be re-used in multiple places in the report. However, unlike formulas, scripts cannot be inserted into the report, they can only be applied to existing report elements. A star next to a script name in the list indicates that the script is applied to the current cell.

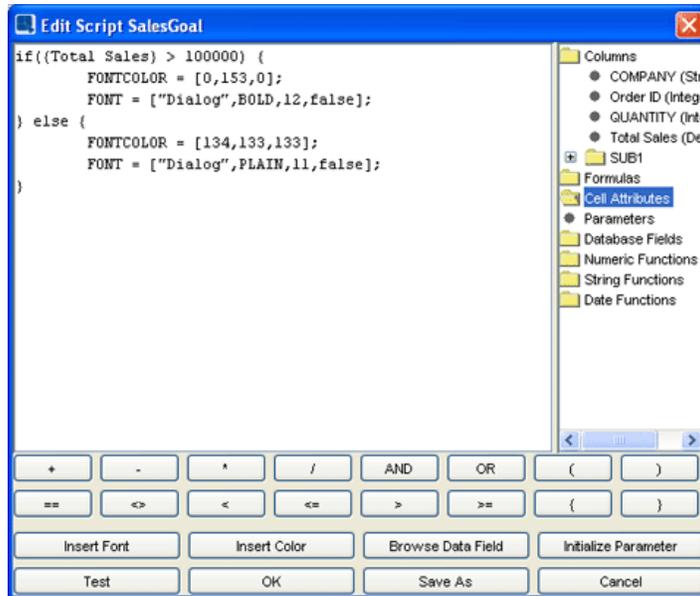
From this dialog, you can edit an existing script, create a new script, rename a script, delete a script, apply a script to the current element, and remove a script from the current element. If a script is being used, it can still be deleted (a message will appear asking if it is ok to proceed) and the script is first removed from any cell before being deleted. If the script is used by a security level, the script has to be removed manually first before it can be deleted. To apply a script to the currently selected report element, select a script from the list and click the *Apply* button. The script list will close and the script will be applied.

Any report element that has a script applied will have a small check mark in the upper left corner of the cell. Scripts will not execute until you preview or export the report.

Months	Orders	Sales	Qty
↑	18	196,341	203

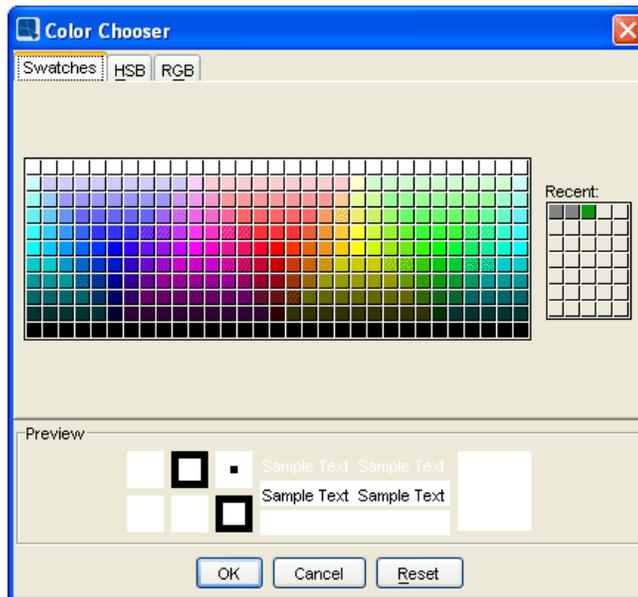
*Script Indicators in Design View*

If you select to create a new script by clicking the *New* button from the script list dialog, you will be prompted to specify a name for the new script. Once you specify a name, the Formula Builder will open, allowing you to construct a script.



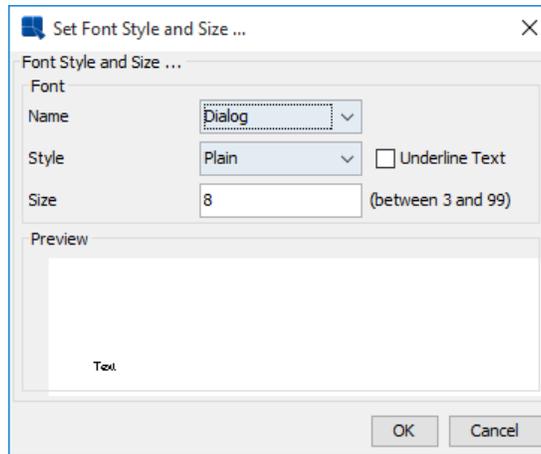
*Formula Builder (Scripting)*

Scripts are developed in the same formula builder interface as formulas. Although scripts use a different syntax, all of the formula fields, operators, and functions are also accessible in scripts. For more information about formulas and the Formula Builder, please see Section 4.1.6 - Using Formulas & the Formula Builder. In addition to the options available for formulas, there are two additional options available when creating scripts. The *Cell Attributes* folder on the right panel contains a list of the cell attributes (in addition to the value) that can be modified with a script. Attributes and their use are discussed in Section 4.1.7.2.1 - Formatting Actions. The *Insert Color* button will bring up a color selection dialog that allows you to visually pick a color for a cell attribute. Selecting a color will automatically insert an array of RGB values into the script.



*Script Color Dialog*

The *Insert Font* button will bring up a set font dialog for you to pick from the available fonts and allow you to set styles such as bold and underline.



*Script Font Dialog*

### 4.1.7.2. Script Syntax

For most scripts the syntax is fairly simple and consists of two basic components: condition and action.

```
if (Condition) {Action(s);}
```

The condition is generally a Boolean expression derived using one of the Boolean operators discussed in Section 4.1.6.2.7.2 - Boolean Operators. For example, (`{Quantity} < 5`) would return `true` when the value of the `Quantity` field is lower than 5. The `AND` and `OR` operators can be used to check multiple conditions. For example, (`{Quantity} < 5`) `AND` (`{InStock} == "Yes"`) would return `true` when the value of the `Quantity` field is lower than 5 and the `InStock` field equals "Yes".

So a simple script that changes the font color to red when negative values are present in the report field would look something like this:

```
if ({SubTotal} < 0) {
    FONTCOLOR=[255,0,0];
}
```

More complex scripts can be created by specifying multiple conditions for multiple actions, using `if else` syntax.

```
if (Condition 1) {Action 1;}
else if (Condition 2) {<i>Action 2</i>;}
else {<i>Action 3</i>;}
```

There is no limit to the number of nesting levels (`if else`) that can be specified. Multiple actions for a single condition can also be specified.

```
if (Condition 1) {
    Action 1;
    Action 2;
    Action 3;
}
```

This will perform multiple actions when the condition is met. Note that each line of the script must end with a semicolon.

#### 4.1.7.2.1. Formatting Actions

The following is a list of element attributes that can be modified using scripts.

**VALUE:** The action syntax that allows you to change the value of the element to which the script is applied is `VALUE=New Value;`. The new value does not necessarily have to be

of the same data type as the report element (for example you can replace number with string). However, changing string to number or number to date can cause formatting irregularities.

- BGCOLOR:** The action syntax that allows you to change the background color of the element is `BGCOLOR=[ R , G , B ]`; where R is a number for the red value of the new color, G is a number for the green value of the new color and B is a number for the blue value of the new color. You can automatically generate the RGB array using the *Insert Color* button in the Formula Builder.
- FONTCOLOR:** The action syntax that allows you to change the font color of the element is `FONTCOLOR=[ R , G , B ]`; where R is a number for the red value of the new color, G is a number for the green value of the new color and B is a number for the blue value of the new color. You can automatically generate the RGB array using the *Insert Color* button in the Formula Builder.
- FONT:** The action syntax that allows you to change the font, font style, and font size of the element is `FONT=[ Font Name , Font Style , Font Size ]`; . The font name is the name of the font you want to use (i.e. "Dialog", "Serif", etc.). The options for font style are `BOLD`, `ITALIC`, or `BOLD+ITALIC`. The font size is a numeric value indication of the font size. For example, `FONT=[ Dialog , BOLD , 12 ]`; would change the font to 12 point bold Dialog.
- ALIGN:** This action allows you to change the horizontal alignment of the data within the cell. The syntax is `ALIGN=Position`; . The position options are `left`, `right`, and `center`.
- BORDERCOLOR:** The action syntax that allows you to change the border color of the element is `BORDERCOLOR=[ R , G , B ]`; where R is a number for the red value of the new color, G is a number for the green value of the new color, and B is a number for the blue value of the new color. You can automatically generate the RGB array using the *Insert Color* button in the Formula Builder.
- BORDERTHICKNESS:** The action syntax that allows you to change the thickness of the element's border is `BORDERTHICKNESS=Thickness`; . The thickness is an integer indicating the number of pixels. For example, `BORDERTHICKNESS=2`; would set the border thickness to two pixels.
- HYPERLINK:** The action syntax that allows you to set the hyperlink properties of the element is `HYPERLINK=[ URL , "Hint " , "Target " ]`; . The URL is the location you want the link to point to. The hint is the mouse over hint that will appear and the target is the specified target for the link. If you do not want to specify the hint or target, specify an empty string " ". Note that you will not be able to click on the hyperlinks in the preview window if you launched the ERES Organizer using `bat` file, but the links will still work if you export the report to DHTML or PDF format.
- BOOKMARK:** This action allows you to specify a named location within the report. When you export to DHTML, it will become an anchor tag. When you export to PDF, it will become a bookmark within the generated PDF file. The syntax is `BOOKMARK=Name`. The bookmark can be any string value.
- XPOSITION:** The action syntax that allows you to change the X position of a cell within a section is `XPOSITION=Position`; (in inches or centimeters depending on which measurement is selected). For example, `XPOSITION=1.2`; would set the left edge of the report element 1.2 inches/centimeters away from the left edge of the section.
- YPOSITION:** The action syntax that allows you to change the Y position of a cell within a section is `YPOSITION=Position`; (in inches or centimeters depending on which measurement is selected). For example, `YPOSITION=0.3`; would set the top edge of the report element 0.3 inches/centimeters away from the top of the section.

<b>WIDTH:</b>	The action syntax that allows you to set the width of a report element is <code>WIDTH=Size;</code> (in inches or centimeters depending on which measurement is selected). For example, <code>WIDTH=1.5;</code> would set the width of the element to 1.5 inches/centimeters.
<b>HEIGHT:</b>	The action syntax that allows you to set the height of a report element is <code>HEIGHT=Size;</code> (in inches or centimeters depending on which measurement is selected). For example, <code>HEIGHT=0.25;</code> would set the height of the element to 0.25 inches/centimeters.
<b>ROTATION:</b>	The action syntax that allows you to set the rotation of a report element is <code>ROTATION=Angle;</code> (in degrees, where negative value brings counterclockwise rotation, applicable values are <code>-90</code> , <code>0</code> and <code>90</code> ). For example, <code>ROTATION=-90;</code> would set the angle of the element to 90 degrees counterclockwise.
<b>VISIBLE:</b>	This action allows you to control whether a cell is visible or not. The syntax is <code>VISIBLE=True/False;</code> . Setting visible to false will make the report element invisible.

There is also a special action called **ALERT**. To learn more about alerts in report scripts, please visit Section 11.2.4 - Reports.

In addition to modifying these properties for the current cell, you can also retrieve these properties from other report elements. You can retrieve the value from a report element in the same manner that you do in formulas either `{Column Field}`, `@Formula`, `:Parameter`, etc, or using the `id()` function for other cells or labels. For more information about this, please see Section 4.1.7.2.3 - Using Formulas.

You can retrieve all other attributes using the following syntax `ATTRIBUTE(Report Element)`. For example, `FONTCOLOR(id(RPT_HDR_LB0))` would return an integer array `[R,G,B]` representing the font color of a label in the Report Header section. You can use this to run comparisons based on attributes or to match the attributes of the current cell with those of another report element. For example, `BGCOLOR=BGCOLOR({ProductName});` would change the current cell's background color to match that of the `ProductName` column.

Depending on which type of element the script is being applied to, certain formatting actions may have no effect. For example, you can set scripts on lines and rectangles, but you can only modify their bounds and colors.

#### 4.1.7.2.1.1. Scripting Image URLs

Utilizing the `if else` conditionals and the Image URL formatting option, you can create image URL scripts that change images dynamically based on data in your report. Suppose you want to include two simple images to help readers determine if the sales for a particular category has achieved the desired goal. Let's say that the sales goal for each category is 100 units sold. Using this information, you can write the following script to display the success image when the goal is met and the failed image when the goal is not met.

```
if(@SUM_Quantity > 100) {
    value = "http://www.quadbase.com/images/green.jpg";
} else {
    value = "http://www.quadbase.com/images/red.jpg";
}
```

Create a new formula cell that has a string data type (you can simply use `" "` for the content) and apply this script to the cell. Then set the data formatting for this cell to be `"Image Url"`. More information about data formatting can be found in Section 4.1.3.7.3 - Data Formatting for Formulas and Column Fields. Here is how the report will look like using this script:

Region	CategoryName	ProductName	UnitPrice	Quantity
East	Arm Chairs	Adad Chair	\$452.00	38
		Cula Chair	\$468.00	58
		Marduk Chair	\$489.00	56
		Nabu Chair	\$456.00	12
		Ningirsu Chair	\$478.00	12
		Nisaba Chair	\$414.00	122
		Nusku Chair	\$425.00	56
		Shugamurra	\$445.00	57
		Shimsaliya	\$424.00	33
			<b>\$443.35</b>	<b>444</b>
	Double Dressers	Sekhmet	\$1,877.00	6
		Serket	\$1,761.00	20
		Set Dresser	\$1,645.00	16
		Tefnut Dresser	\$1,798.00	9
	Thoth Dresser	\$1,872.00	17	
		<b>\$1,785.67</b>	<b>68</b>	

Scripted Image URL

#### 4.1.7.2.2. Variables & Arrays

In addition to the if else statements, you can also define variables and arrays in a script. Variables and arrays are defined at the beginning of the script (before any statements).

##### 4.1.7.2.2.1. Variables

If you have a constant or expression that you intend to use in multiple places in a script, it can often be easier to define a variable rather than re-typing the constant or expression in every place where it is used. A variable can be declared at the beginning of a script using the following syntax: `DataType VariableName ;`. The data type is the type for the variable value and can be either Number, Boolean, String, or Date.



#### Note

DateVar is used as the data type declaration for Date variables. This avoids confusion with the date() function. The variable name can be anything except an operator or function name.

The following example uses variables which contain an expression.

```
String Line1;
String Line2;
Line1 = {Address1} + "\n";
Line2 = {City} + ", " + {State} + " " + {Zip};

if ({Address2} == NULL) {

    VALUE = Line1 + Line2;
} else VALUE=Line1 + {Address2} + "\n" + Line2;
```

This script pieces together several report fields to create an address entry. The first two lines of the script declare the variables. The script then concatenates fields and strings depending on whether there is an Address2 entry or not. Using the variables prevents having to re-type the expressions. Note that “\n” is used to break the text to a new line.

You can also assign a value to a variable as part of an action in a script. You can use this to modify the variable value based on some conditions and then return the variable value to the cell. This is useful if you only want to write one value statement. The following example assigns a value to a variable:

```
if ({Flags} == "M") {
    LinkAppend = "Main";
} else if ({Flags} == "R") {
    LinkAppend = "Remote";
} HYPERLINK = ["StatPage.html#" + LinkAppend, "", "];
```

In this example, a variable is defined without an initial value. The value of the variable is assigned based on a string flag in the report. At the end of the script a hyperlink is defined for the element that uses the variable value to specify a page location.

#### 4.1.7.2.2. Arrays

In addition to variables, you can also declare arrays at the beginning of a script. Specific values from the array can be retrieved later. This is useful if you have a number of fixed values that you will be using in different places in the script like months or days of the week. An array declaration is similar to a variable declaration and uses the following syntax: `DataType[] ArrayName = [Value1, Value2, Value3, ...];`. Data types for arrays are the same as for variables and can be one of Number, Boolean, String, or Date. The variable name can be anything except an operator or function name. The initial value assignment is optional.

You can retrieve a value from an array using the following syntax: `ArrayName[Index]`. The following example uses an array to replace the numeric values for a Month column (1-12) and replaces them with the proper month names.

```
String[] MonthNames = [ "January", "February", "March", "April",
    "May", "June", "July", "August", "September", "October",
    "November", "December" ];
```

```
VALUE = MonthNames[ ( {Month} - 1 )];
```

Notice that the index specified is `{Month}-1`. This is because the first month value is 1 and the first index value for the array is 0.

Like variables, you can also assign or overwrite values in an array as part of an action. The syntax for this is `ArrayName[Index]=Value;`.

#### 4.1.7.2.3. Using Formulas

You can use any of the ReportDesigner formulas from the main report or any sub-reports (for more on accessing sub-report formulas, see Section 4.1.6.3 - Subreport Formula Access) in both the condition and action components of the script, as well as for variable values. Most commonly formulas are used to retrieve values either from the current element or other report elements. You can retrieve the value from column fields using `{Column Field}`, the value from existing report formulas using `@FormulaName` and the user supplied values for query and formula parameters using `:ParamName` and `?ParamName` respectively.

In addition to these methods, there is one additional function that can be used in scripts.

**this() -** This function returns the value of the current element (i.e. the cell to which the script is applied). The syntax is `this()` without any arguments. The function can be useful if you're reusing scripts or if you want to ensure that the script is referring to the cell to which it is applied. For example `If (this() < 0) {FONTCOLOR=[255,0,0];}` would turn the font red, then the value of the current cell is less than zero.

You can also use the formulas to build expressions for comparison or to change an element's value. For more information about ReportDesigner formulas and their syntax, please see Section 4.1.6 - Using Formulas & the Formula Builder.

#### 4.1.7.2.4. Checking for Nulls

Using cell scripts you can check for null values within report columns. To specify a null as a constant, simply use the word `null` without any delimiters. So a simple script that checks for nulls would look like this:

```
if (this() == null) {
    VALUE="yes";
}
```



#### Note

Even if you have set the null data handler in Report Designer, you can still use script in this manner. This is because the script will execute before the null handler has a chance to convert the content of the cell.

#### 4.1.7.2.5. Row-Specific Options

Using scripts, you can retrieve and modify values for specific rows of column fields. Normally when you retrieve or replace the value of a column field you will get/replace the first value from the column or the current row if the script is applied to a field in the Table Data section. However, you also have the option of specifying which row you would like to get/replace. To do this, you can append a column field with the following syntax: `.row(Row Index)`. The row index is the index value associated with each row in a report starting with 0. For example `{ProductName}.row(3)` would return the fourth value of the ProductName field.

You can obtain the current row index as well as the total number of rows in the report using the `getRowIndex()` and `getTotalRowIndex()` functions respectively. The following example uses this feature to hide repeated values in a column field.

```
if (({Category}.row(getRowIndex()) == {Category}.row(getRowIndex()-1)) AND
    (getRowIndex() <> 0)) {
    FONTCOLOR=[ 255 , 255 , 255 ];
}
```

This example assumes that the background color for the report is white and hides the field by adjusting the font color accordingly.

#### 4.1.7.2.6. Loops

In addition to conditional if else statements, ReportDesigner also allows you to add loops to scripts. Used in conjunction with variables and row-specific options/formatting, loops are useful if you need to perform running totals/calculations independent of the current row or section of the report. The basic syntax for loops is:

```
while (Condition) {Action(s);}
```

The specified actions will be performed as long as the condition (Boolean expression) is true.

For example, say you have the following report:

Product	Quantity	Price
Chair	6	\$327.00
Chair	4	\$218.00
Chair	12	\$418.00
Chair	5	\$221.00
Chair	18	\$248.00
Table	4	\$875.00
Table	2	\$1,024.00
Table	8	\$967.00
Table	7	\$1,106.00

Adding a formula that calculates the total quantity to the Table Footer (`sum({Quantity})`) would return 66. However, say you only wanted to show the total quantity for chairs in the Table Footer. To do this, you write a loop. Applying the following script to the aggregation.

```
Number i;
Number s;
i = 0;
s = 0;

while (i < getTotalRowIndex()) {
```

```

if ({Product}.row(i) == "Chair") {
    s = s + {Quantity}.row(i);
}
i = i + 1;
}
VALUE = s;

```

will return 45 - the sum of the quantity column for Chair. The script loops through each row in the report and checks to see if the value for the Product column is "Chair". If it is, the Quantity value for that row is added to the variable s.

#### 4.1.7.2.7. Section Scripts

In addition to individual cells, scripts can also be applied to report sections. Based on conditions within the report data, individual section options can be turned on/off and section background colors can be set. To add a script to a report section, click on the button with the section name on the left side of the Designer. You can also right click on a blank portion of the section field and select *Scripting* from the pop-up menu.

Writing section scripts is exactly the same as writing scripts for report elements as they use the same syntax. The only difference is that instead of *Cell Attributes* folder in the Formula Builder, you will see *Section Attributes* folder containing various formatting actions that can be performed with section scripts.

##### 4.1.7.2.7.1. Formatting Actions for Section Scripts

The following is a list of section attributes that can be modified using scripts (not all options are available for every report section).

<b>BGCOLOR:</b>	The action syntax that allows you to change the background color of the element is <code>BGCOLOR=[R,G,B]</code> ; where R is a number for the red value of the new color, G is a number for the green value of the new color and B is a number for the blue value of the new color. You can automatically generate the RGB array using the <i>Insert Color</i> button in the Formula Builder.
<b>PRINT_ON_NEW_PAGE:</b>	This action allows you to turn on/off the print on new page section option. The syntax is <code>PRINT_ON_NEW_PAGE=True/False</code> ; For more information about the print on new page feature, please see Section 4.1.3.3 - Section Options.
<b>RESET_PAGE_NUMBER:</b>	This action allows you to turn on/off the reset page number section option. The syntax is <code>RESET_PAGE_NUMBER=True/False</code> ; For more information about the reset page number feature, please see Section 4.1.3.3 - Section Options.
<b>VISIBLE:</b>	This action allows you to control whether a section is visible or not. The syntax is <code>VISIBLE=True/False</code> ; Setting visible to false will make the section invisible. Using this attribute coupled with the nested section feature (Section 4.1.3.1.1 - Nested Sections) allows you to create different headers/footers to display depending on the data.
<b>HEIGHT:</b>	This action allows you to set the height of a section. The syntax is <code>HEIGHT=Size</code> ; (in inches or centimeters depending on which measurement is selected). For example <code>HEIGHT=0.25</code> ; would set the height of the section to 0.25 inches/centimeters.



#### Note

Scripts that set or refer to section attributes can only be applied to sections. They cannot be applied to other report elements.

#### 4.1.7.2.8. Commenting Scripts

You can also add comments to a script to explain its design and operation. A comment is preceded by two slashes `//`. Anything following this on the same line is treated as a comment. You can break comments into multiple lines, but each line must begin with `//`.

#### 4.1.7.2.9. Accessing Values Computed by Other Scripts

If you have scripts applied to cells in your report, you can access these values using the syntax `.getScriptValue()`. For example, you may have this data:

```
String, int
Month, Revenue
January, 3567
"February", 6854
"March", 5421
"April", 8425
"May", 5611
"June", 2356
"July", 6543
"August", 2563
"September", 4432
"October", 8841
"November", 1023
"December", 2265
```

If you wish to take this data and create a year to date revenue column, you could use the following script, which would display the scripted value from the previous row added to the revenue from the current row. This script is written to be put on a formula consisting of the number 0 as a column in the table data section, referred to within this script as `{formula1}`.

```
if(getRowIndex()==0)
    VALUE={revenue};
else
    VALUE={revenue} + {formula1}.getScriptValue(getRowIndex()-1);
```

This would yield a report like this:

Month	Revenue	Year to Date
January	3,567	3,567
February	6,854	10,421
March	5,421	15,842
April	8,425	24,267
May	5,611	29,878
June	2,356	32,234
July	6,543	38,777
August	2,563	41,340
September	4,432	45,772
October	8,841	54,613
November	1,023	55,636
December	2,265	57,901

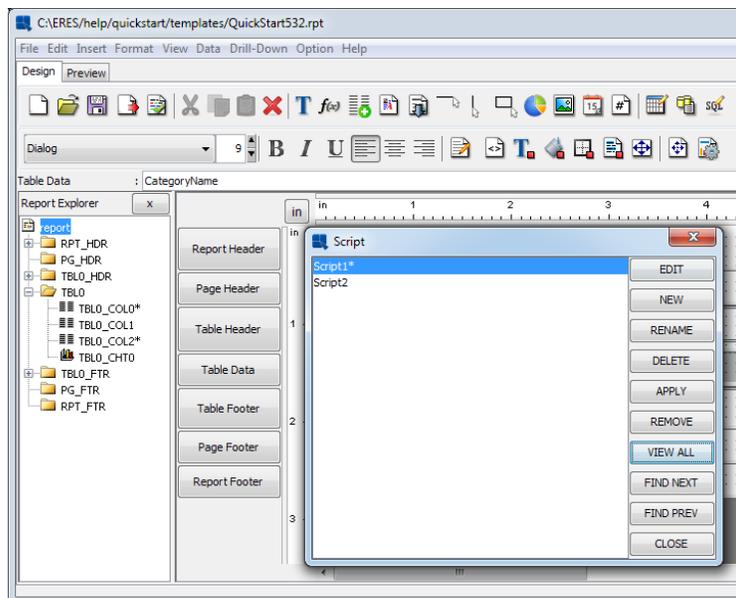
#### 4.1.7.3. Managing Scripts

For reports that contain a number of scripts or scripts applied to a number of elements, there are several options available to help you manage and find the elements that the scripts are applied to. In the script dialog, you will notice that in addition to the standard options such as remove, delete, and rename, there are also buttons to view all, find next, and find previous elements. Make sure that you select a script before using these buttons.

##### VIEW ALL:

This button will open the Report Explorer if it is not already open and highlight every element that the selected script has been applied to with an as-

terisk. The tree will automatically expand the sections that have highlighted cells. Clicking on one of these elements will center the designer view on that element allowing you to change or remove the applied script. The highlights will be changed if you select another script and click on VIEW ALL again. Closing the script dialog box will remove all the highlights and the explorer tree will collapse.



View All Result

**FIND NEXT and FIND PREVIOUS**

These buttons are used to cycle through elements that the selected script has been applied to. If you already have an element selected in the Designer when clicking on these buttons, the search will begin at the current element. If no elements are selected, the search will begin at the beginning of the report. If you select another script and click FIND NEXT, the asterisk will shift to the newly selected script and the first element that has the script applied will be highlighted in the designer. FIND NEXT and FIND PREVIOUS only cycle through visible elements. If you want to see invisible elements with the script, use VIEW ALL which shows all invisible elements as well as visible elements.

For more information about the Report Explorer, please see Section 4.1.3.8 - The Report Explorer

**4.1.8. Drill-Down**

Often reports are used to display large amounts of data. However, with large data sets (particularly if the data is fairly detailed) displaying everything in one report may not be the best way to present the information. One solution to this problem is to create a top-level report that displays only summarized data and allow users to click through to see underlying data. This is the concept of drill-downs.

In ReportDesigner, users can easily display data in this way. Links can be placed on column fields in the primary (top-level) report to secondary (sub-level) reports. The sub-level reports use parameterized queries that accept the value from the primary report as the input. Hence, sub-level reports will only display data related to the particular value on which the user has clicked.

For example, say you are designing a report to show sales data for an entire year. Rather than creating a report that displays the entire year's worth of data, you could design a top-level report that shows aggregated data by month.

Month	Units Sold	Total Sales
January	5	\$10,224.12
February	4	\$12,286.14

Month	Units Sold	Total Sales
March	6	\$14,415.16
April	5	\$12,316.11
May	2	\$9,482.64
June	3	\$10,116.12
July	7	\$17,624.18
August	5	\$10,361.54
September	4	\$10,116.82
October	3	\$8,612.41
November	5	\$10,552.24
December	11	\$22,614.89

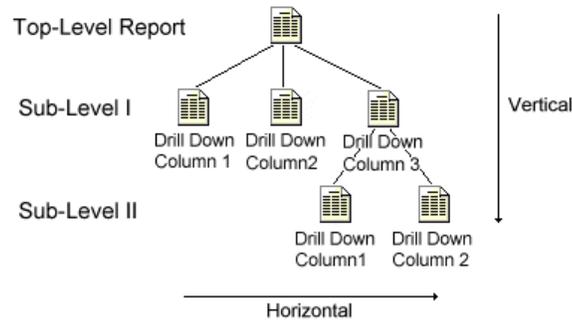
Users viewing the report could then drill-down to see more data for each month. In this case, say the user clicks on *March*. They would then be taken to an underlying report that offers detailed sales information for that month.

Sales Data For March 2001				
Order #	Customer	Product	Quantity	Total
10042	Allied Furniture Emporium	Chair	2	\$6,425.16
10051	Furniture World	Table	1	\$3,114.12
		Dresser	1	\$2,116.83
10068	Domus Home & Garden	Chair	1	\$800.65
		Dresser	1	\$1,958.40
<b>Total:</b>			<b>6</b>	<b>\$14,415.16</b>

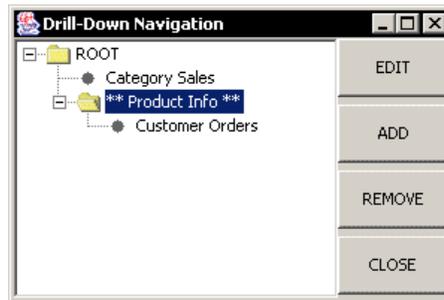
From this report, another layer of drill-down can be added, allowing users to see more detailed customer information. In this case, say the user clicks on *Allied Furniture Emporium*. They would be taken to a third report that shows more information about the customer.

Customer Information		
		<b>Address:</b>
<b>Company:</b>	Allied Furniture Emporium	384 Broad St. Littletown, NY 18322
<b>Contact Name:</b>	Matilda Gladwaller	
<b>Order History:</b>		
<b>Year</b>	<b>Orders</b>	<b>Total Sales</b>
1999	14	\$52,614.18
2000	18	\$68,115.89
2001	12	\$38,812.12

Using drill-down in this manner allows large amounts of information to be presented in a way that is easy to understand and navigate. ReportDesigner allows horizontal and vertical layers of drill-down. This means that different columns in that same report can have drill-down (horizontal), or that sub-level reports can have subsequent drill-down reports (vertical). Users can also add drill-down layers to sub-reports.



Within the Report Designer, drill-down is controlled from the Drill-Down menu. Selecting the *Navigate* option from this menu brings up the Navigation window. From this window you can add and edit drill-down reports.



*Drill-Down Navigation Dialog*

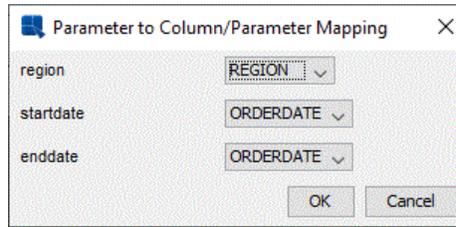
The left side of the Navigation window displays the hierarchy of drill-down reports (see the above diagram). The ROOT is the top-level report. The level that you are currently editing is marked with “\*\*”. To edit a different report, select it and click the *EDIT* button on the right side. That report will then open in the Designer, allowing it to be customized. Reports can also be removed by selecting the report and then clicking the *REMOVE* button.

### 4.1.8.1. Creating a Drill-Down Report

The first step in creating a drill-down report is to create the top-level report. Because drill-down relies on relationships between data sets, only reports that use parameterized queries, XML queries, or class files can be used for drill-down layers. When designing the top-level report, bear in mind that at least one of the columns or parameters will be passed as a parameter to any sub-level reports. After you finish designing the top-level report, go to the Drill-Down menu and select *Navigate*. The navigation window will open. Select the report under which you would like to add the drill-down layer (if the report has no drill-down, only the ROOT node will be visible), and click the *ADD* button. If you are adding drill-down to a sub-report, first go to the sub-report tab and then load the navigation dialog.

You will then be prompted to create a new report or use an existing report for the drill-down layer. You can use any existing report; however any report for a drill-down layer must have a parameterized query or class file as the data source. If the existing report contains drill-down levels of its own, all drill-down levels will be imported as well. If you select to create a new report, the Report Wizard will open, allowing you to select a data source, report type, and column mapping.

The next step is to map the columns or parameters in the top-level report, to the parameters in the sub-level report. You will be prompted to do this when you select an existing report for the drill-down layer, or when you select the data source for a new report for the drill-down layer. In either case, a dialog will appear prompting you to map the columns or parameters from the top-level report.



*Parameter to Column/Parameter Mapping Window*

The options that are available in the drop-down menu are based on data type. For example, if your parameterized query takes a string as a parameter, only columns containing string data can be mapped. If there are multiple parameters for the sub-level query, there will also be the option *None*. This will allow you to create unmapped drill-down reports. For more information, see Section 4.1.8.5 - Unmapped Drill-Down. Once you specify the drill-down mapping, you will be prompted to specify a name for the sub-level report. Now the sub-level report will open in the Design window, allowing you to format and customize it.

#### 4.1.8.1.1. Drill-Down Links

By default, a link will be placed on the column in the top-level report which is mapped to the sub-level report. Instead of using the default link, you can place the link on a different column, or multiple columns using the *Drill-Down Link* option from the Drill-Down menu.

To add or remove a drill-down link from a report column, first make sure that you are currently editing the correct template (top or higher level). Then select a column field and select *Drill-Down Link* from the Drill-Down menu. This will bring-up a drop-down list allowing you to select which sub-level report you would like to link to. To remove a drill-down link, select the blank option at the top of the drop-down list.



*Drill-Down Link Dialog*

##### 4.1.8.1.1.1. Linking from Charts

In addition to placing links on a column, you can also set drill-down links on a chart. With this feature, users can click on the data point in a chart and traverse to the next level of drill-down. To set a drill-down link on a chart, select the chart object in the Report Designer and select *Drill-Down Link* from the Drill-Down menu. Next, select the sub-level report just as you would for a column.

When you set a drill-down link in the chart depending on the elements in the chart and the number of parameters in the sub-level report, the category, data series, and/or sum by value for the data point that is selected are passed as the parameter value to the sub-level report. In order for this feature to function correctly, the chart will need to use the same data source as the report. The chart should also have the categories mapped to the column that was selected to match the parameter in the sub-level report. For more information about column mapping in charts, please see Section 4.2.2.2 - Basic Data Mapping and Section 4.2.3 - Chart Types and Data Mapping.

#### 4.1.8.2. Multi-Value Drill-Down

In addition to drilling on one field at a time, ReportDesigner supports the concept of multi-value drill-down. Using this feature, you can drill into several different values from a top-level report at once.

For example say you have a report detailing sales information for 100 different customers. Using the multi-value drill-down, users could select the customers that most interest them and drill into their records at once instead of looking at each customer individually.

##### 4.1.8.2.1. Creating a Multi-Value Drill-Down Report

Adding layers of multi-value drill-down to a report is almost exactly same as adding regular drill-down layers. The only difference is in the query in the lower-level report. To create a multi-value drill-down layer, the lower-level report must have a multi-value parameter in its query that is mapped to a column in the higher-level report.

Using the previous example, say the query for the lower level has a multi-value parameter for a customer id - something like `Where Customers.CustomerID IN (:CustID)`. If the `CustID` parameter is mapped to the `Customer ID` column in the top-level report, this will automatically create a multi-value drill-down layer. For more information about multi-value parameters, see Section 3.1.3.2.2.1 - Multi-Value Parameters. If you do not wish to create a multi-value drill-down layer, simply use a single value parameter in the lower-level report.



**Note**

You can have only one report per level when using multi-value drill-down. It does not support linking different columns from the same report to different reports.

You can place column links in multi-value drill-down reports; however, a dialog or form is used to allow users to select multiple column values and drill to the next level. For more information about this, please see Section 4.1.8.4 - Viewing Drill-Down Reports.

**4.1.8.3. Crosstab Drill-Down**

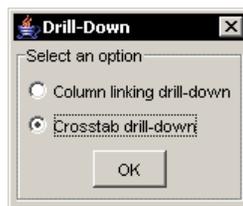
ReportDesigner provides a special implementation of the drill-down features when the top level report is a crosstab report. Using the crosstab drill-down features, users can click on a cell in the crosstab matrix and drill into information for that particular row or column. For example, assume you have the following report showing sales volume by product and region.

	East	South	Midwest	West
Chair	144	208	131	108
Cabinet	208	114	158	206
Dresser	100	101	112	109

Using standard drill-down configuration, if a user clicks on one of the hyperlinked cells, only the values from that row could be passed to the lower-level report. However, with crosstab drill-down, the column break and row break values for that cell are passed to the lower-level report. Therefore, if the user clicks on the cell value *144*, the values "Chair" and "East" would be passed to the parameters in the lower-level report.

**4.1.8.3.1. Creating a Crosstab Drill-Down Report**

When you select to add a drill-down layer to a crosstab report, you will be prompted whether you would like to create a standard (column linking) drill-down presentation or use crosstab drill-down.



*Crosstab Drill-Down Options*

If you select to use crosstab drill-down, the setup will continue just as it would for any other drill-down presentation. The only difference comes with parameter mapping. Instead of mapping parameters from the lower level report to columns in the main report, they are mapped to the row break and column break values for the top-level report.



*Parameter-Column Mapping for Crosstab Drill-Down*

Once you add the drill-down layer, the cells (column break values) in the crosstab matrix for the top level report will automatically be linked to the layer you have added.

#### 4.1.8.4. Viewing Drill-Down Reports

You can directly navigate and view drill-down reports within the Report Viewer Applet (and the Preview window). When you open a report containing drill-down, you can click on any entry in a column with a drill-down link. You will be taken to the sub-level report, which will take the value for that row as the parameter.

For multi-value drill-down reports, you can select *Select Multiple Drill-Down Values* from the *Data* menu in the Preview window or from the Viewer pop-up menu. This will bring up a dialog containing all the values in the mapped column. You can then select the values you want and click the *Ok* button to go to the next level.



*Multi-Value Drill-Down Selection Dialog*



#### Note

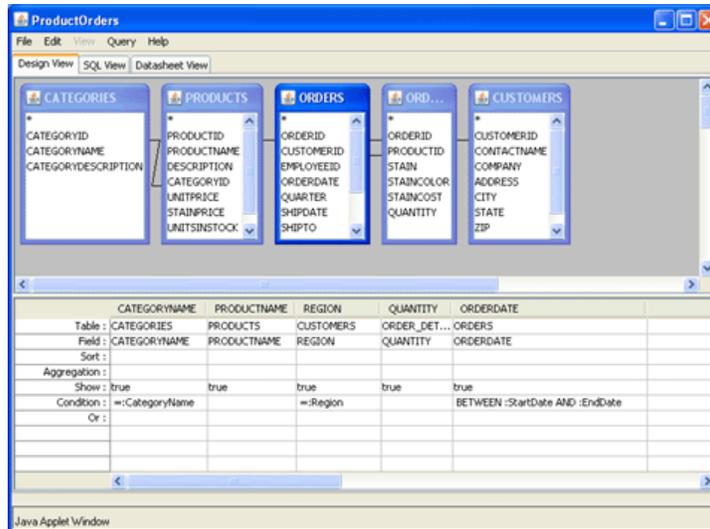
The templates for sub-level reports are saved in DrillDown directory. If you are viewing reports on another machine, the Report Viewer will look for the templates in that directory, so they have to be moved as well.

Drill-down reports can be exported to any format; however, only the DHTML and PDF formats can retain the drill-down functionality (multi-value drill-down reports only support DHTML). Any other format will only export the top-level report.

#### 4.1.8.5. Unmapped Drill-Down

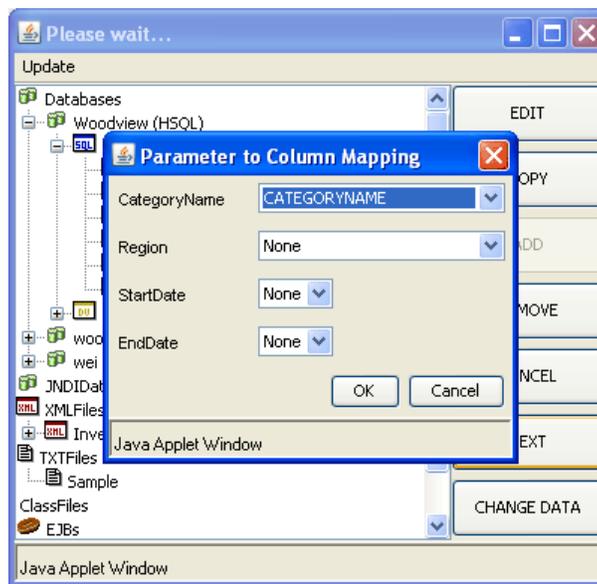
When you create a sub-level query with multiple parameters, you have the option of leaving some of these parameters unmapped, so that they can be prompted after clicking a drill-down link. To use this feature when selecting mapping for parameters, select **none** for the parameters you wish to remain unmapped. The sub-level report also contains a *Data* menu option *Preview Parameter Prompt* which determines if the parameter prompt will be displayed when you drill-down from the root level. If this level has never been previewed before, it will be turned off and use the default parameters. Otherwise, it will save the last used parameter values for future use.

Each sub-level drill-down report requires at least one mapped parameter. Thus, in order to create a report with unmapped parameters, the sub-level query must have at least two parameters. For example:



*Drill-Down Level Query*

Map at least one parameter to a database column and select **none** for the other(s).



*Creating Unmapped Parameters*

The remainder of the creation of the drill-down report is like any other.

When you preview the report, the root report will be displayed as normal (with a parameter prompt if the root report contains parameters).

**Category Listing**

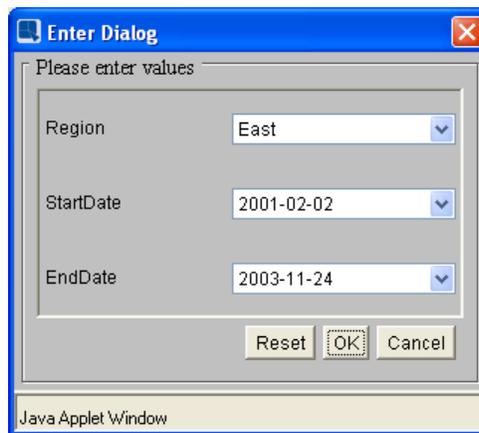
CATEGORYID :	ARC
CATEGORYNAME	Arm Chairs
CATEGORYDESCRIPTION	Formal Arm Chair

CATEGORYID :	ROT
CATEGORYNAME	Round Tables
CATEGORYDESCRIPTION	Circular Table 56" Diameter

CATEGORYID :	DOO
CATEGORYNAME	Double Dressers
CATEGORYDESCRIPTION	Dresser with 2 sets of drawers

CATEGORYID :	SIC
CATEGORYNAME	Side Chairs
CATEGORYDESCRIPTION	Formal Chair Without Arms

If you click on a link and the *Preview Parameter Prompt* menu option for the drill-down report is checked, you will see the following prompt. Notice that the `CategoryName` parameter is not shown:



*Parameter Prompt for Unmapped Drill-down*

The resulting report will look like this:

**Product Orders**

CATEGORYNAME	PRODUCTNAME	REGION	QUANTITY	ORDERDATE
Round Tables	Atun Table	East	21	Jul 11, 2003
	Apep Table	East	6	Nov 24, 2003
	Anubis Table	East	16	Feb 14, 2003
	Atun Table	East	6	Feb 21, 2003
	Ningizida Table	East	4	Jan 16, 2003
	Amon Table	East	12	Feb 14, 2003
	Ningizida Table	East	18	Jun 22, 2002
	Anubis Table	East	4	Jun 22, 2002
	Bast Table	East	16	Dec 4, 2001
	Bast Table	East	12	Mar 12, 2002
	Anubis Table	East	13	Feb 2, 2001
	Apep Table	East	15	Feb 24, 2001

From the DHTML Viewer or in an exported file, the default values will be used, but they can be changed from the filter options.

**4.1.9. Sub-Reports**

A sub-report is a report within a report. The sub-report is nested entirely within a section of a primary report. Each sub-report has its own data source and design. Using sub-reports you can create reports that use un-related data. You can also create a report that has more than one data table.

In ReportDesigner, sub-reports are generally unrelated to the primary report (however parameterized sub-reports can be linked to column fields in the primary report, see Section 4.1.9.4 - Linked Sub-Reports). Each sub-report uses its own data source and it can be designed independently from the primary report. It also runs entirely within the section in which it is placed in the primary report. You can either use an existing report (.pak, .qrp, .rpt or .xml) template or create a new one for the sub-report.

### 4.1.9.1. Adding a Sub-Report

To add a sub-report to an existing report, click the  *Insert Sub-Report* button on the toolbar or select *Insert Sub-Report* from the *Insert* Menu. A second menu will open giving you the option of adding a new sub-report or re-using an existing one. To insert a new sub-report, select *New...*

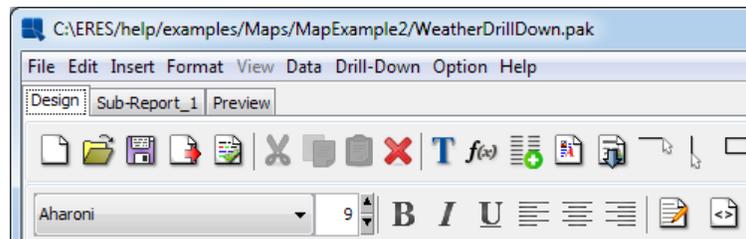
The cursor will turn to cross. Position it where you want to insert the sub-report and click. A dialog box will appear asking you whether you want to use an existing report or create a new one.

You can use any existing report in either .pak, .rpt or .xml format as the sub-report. If you select to use an existing report, you will be prompted to specify the location of the report file. After you specify the file, it will be inserted as a sub-report.

If you select to create a new report for the sub-report, the Report Wizard will launch, allowing you to go through the process of designing the report from data source selection to report mapping. Once you finish with report mapping, the new report will be inserted as a sub-report.

### 4.1.9.2. Editing a Sub-Report

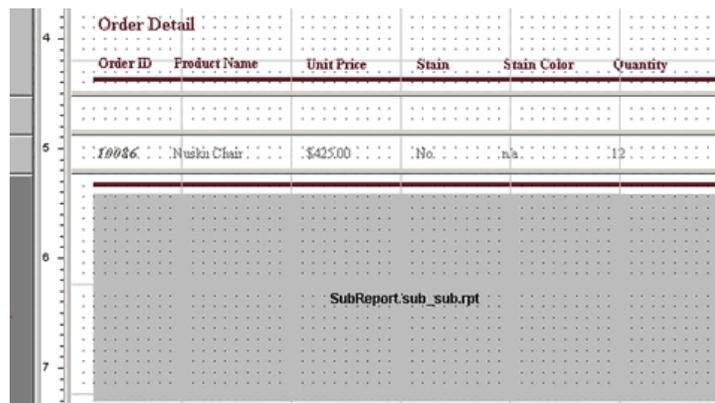
After a sub-report has been added, a new tab will appear at the top of the Report Designer window.



*Sub-Report Tab*

Selecting the *Sub-Report* tab will open a new design pane that allows you to modify the sub-report. You can modify the sub-report in the same way as you modify any report. For each sub-report you add, a new tab will appear. When you have a sub-report, the *Preview* tab becomes context sensitive. If you click on the *Preview* tab from the *Sub-Report* tab, the preview pane will only contain the sub-report. If you click on the *Preview* tab from the *Design* tab, you will preview the main report with the sub-report embedded.

Within the Design window, the sub-report will appear as a gray rectangle. It can be moved and resized like any object. You can also edit a sub-report by double-clicking on the corresponding rectangle in the Design window.



*Sub-Report in Design Window*

Sub-reports run entirely in the section in which they are placed and will generally repeat each time the report section repeats. This means that if the sub-report is placed in the table header section, it will only run once (unless the *Print on Every Page* option is enabled). However, if the sub-report is placed in the Group Header section, the sub-report will repeat for each group in the report.

The exception to this is the table data section. Sub-reports inserted into the table data section will not repeat for each row, but will rather run alongside the data in the primary report. This allows you to create a report with the effect of two data tables running next to each other.

#### 4.1.9.2.1. Sub-Report sizing

The size of the gray rectangle in the main report Design window represents the size of the sub-report, so if the sub-report is larger than the space you allow in the Design window, portions of the sub-report may be truncated.

Fixing a sub-report's size within the main report may not be the best solution if the number of records in the sub-report vary. You can solve this by setting the sub-report to have variable height. Dynamic sizing for sub-reports is set in the same way as for other report cells using the resize to fit option. For more information about this feature, please see Section 4.1.3.7.11 - Moving and Resizing Report Elements.

You can also set the width of the sub-report to be dynamic. This feature is most useful for crosstab type reports where the number of columns in the report can vary as the data or filters change. This sets the sub-report to always draw wide enough to fit all the visible columns in the sub-report. To set dynamic width for the sub-report, right click on the sub-report in the Design window and select *Auto Resize Width* from the pop-up menu (To disable this feature, open the pop-up menu and click on this option again).



#### Note

If you set a sub-report to have variable height, any objects placed directly below the sub-report in the same section of the primary report will not shift to accommodate the resized sub-report. Hence, these objects may be overlapped by the sub-report.

To ensure that objects are not overlapped, it is recommended that you place sub-reports with dynamic height in their own report sections. If necessary, you can add nested sections to the report. For more information about this, see Section 4.1.3.1.1 - Nested Sections.

#### 4.1.9.3. Removing a Sub-Report

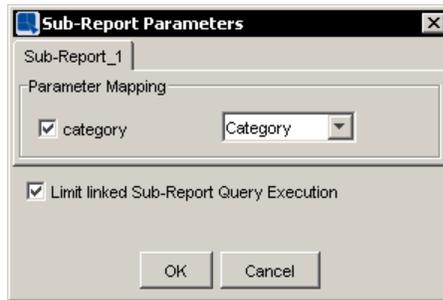
There are two ways to remove a sub-report from the primary report. You can simply select the sub-report object in the primary report and delete it. This will remove the sub-report from the primary report but it will keep it available. This means that the sub-report tab is still active and the sub-report can be re-inserted into the primary report.

The second method is to select *Remove Sub-Report* from the *Edit* menu. A second menu will open with a list of all the sub-reports embedded in the current report. Select the sub-report you want to remove. The sub-report tab will be removed and the sub-report will no longer be available.

#### 4.1.9.4. Linked Sub-Reports

Sometimes you may want sub-reports to be coordinated with the data in the primary report. For example, if you have grouped data, you may want to filter a sub-report so that it shows data pertinent to each group. You can use linked sub-reports to do this.

Sub-reports are linked to primary reports using query parameters. So to create a linked sub-report, the sub-report must have a parameterized query. After you create the sub-report, you can link it by selecting *Sub-Report Parameter Mapping* from the *Data* menu. This will bring up a dialog allowing you to map column fields from the primary report to query parameters in the sub-report. This operation is similar to the one used in drill-down.



*Sub-Report Parameter Mapping Dialog*

The dialog contains a tab for each parameterized sub-report. Each parameter in the sub-report can be mapped to a different column field in the primary report.

Normally when linked sub-reports execute, the sub-report will issue its query each time it runs using the current value of the report column. However, if your report is configured such that the sub-report executes a large number of times, these repeated queries can have an impact on performance. If you check the last option in this dialog called *Limit Sub-report query execution*, ReportDesigner will try to merge the sub-report queries and limit the number of calls to the database when the sub-report runs. Note that this option will only take effect if the sub-report uses a database query to retrieve the data. This will not effect sub-reports that use parameterized classes or xml files as the data source.

When using linked sub-reports, it is recommended that you do not use the cascading parameters functionality, as this may potentially cause incorrect data to appear (depending on the cascading parameters and the order in which they are presented). The default value of the parameter will always be selected instead of the value passed by a column in the main report.

#### 4.1.9.4.1. Using Linked Sub-Reports

Although linked sub-reports will run in any configuration and in any report section, it is important to be aware of the behavior of sub-reports when creating a linked presentation. As noted in Section 4.1.9.2 - Editing a Sub-Report, sub-reports will only run as many times as the section in which they are placed and will only run once when placed in the Table Data section.

Therefore, if you place a linked sub-report in a section that runs only once (Table Header, Report Header, Table Data, etc), the sub-report will only run once and it will use the first value from the column to which it is linked to filter the report. To generate a linked presentation where the sub-report runs for each value in the column to which it is linked, you should use a summary break layout. By setting the column as a row break and placing both the column and the sub-report in the Group Header section, you will get the desired result.



#### Note

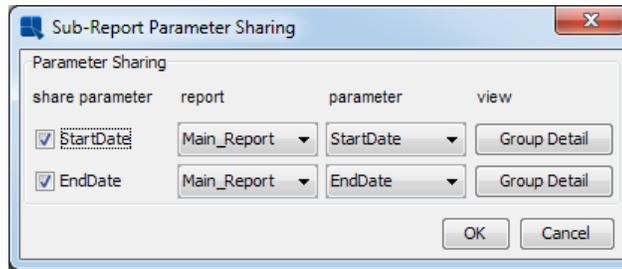
If a relationship between the column and the sub-report already exists in the database, you may want to generate a side-by-side master & details layout instead of linked sub-reports. Using the master & details layout will result in better performance. For more information about this, please see Section 4.1.2.4.1 - Data Mapping.

#### 4.1.9.5. Parameter Linking

In addition to taking a value from a column in the main report, a sub-report can also share parameter values with the main report or with other sub-reports. In this arrangement, a user can enter values to filter the main report's data and have the same filters apply to the sub-report data.

Parameter linking is only available if parameters share the same data type. By default, if you create a sub-report that has parameters with the same name and data type as those in the main report, they will be linked. When parameter values are supplied to the main report, they are also automatically passed to the sub-report.

The parameter linking behavior can be modified by navigating to the sub-report that you want to modify and selecting *Sub-Report Parameter Sharing* from the Data menu (in the Sub-Report tab). This will bring up a dialog allowing you to modify the linking behavior for the sub-report parameters.



Parameter Sharing Dialog

For each parameter defined in the sub-report, you can enable/disable the linking. You can also set which parameter in the main report or in another sub-report you want to link to. The *Group Detail* button will launch a dialog showing the current relationships for that parameter.

Once you specify the settings you want for all parameters, click the *Ok* button to apply the changes.

For more about query parameters and initialization, please see Section 3.1.3.2.2 - Parameterized Queries of this guide.



**Note**

Parameter linking will be overridden if you select to map a column field from the main report to the sub-report parameter as described in Section 4.1.9.4 - Linked Sub-Reports.

### 4.1.10. Template Security

One of the most common needs with a reporting tool is to keep the number of different report templates at a minimum. In ERES, the administrator can grant access to specific reports for users and groups in the system. However, simply granting or denying access may not be enough to facilitate template re-use. To help with this, ERES provides a template security features that allow the administrator to configure different levels of access to the same report template for different users/groups in the system.

For example, say you have a group in ERES called *Sales*. The group all has access to a report that shows total sales grouped by sales region. Sales executives viewing the report need to see the totals as well as the contributions by each region, while individual account managers would only need to see the data for their region. Rather than creating a different report for each sub-group within the *Sales* group, the same template can have security settings applied. The base template for this scenario would look like this:

Sales Report				
<b>Total Units Sold: 29 Total Sales: \$24,449.69</b>				
Eastern Region				
Order #	Quantity	Product	Unit Price	Total Sales
1001	2	Chair	\$325.16	\$650.32
	1	Table	\$1,114.18	\$1,114.18
<b>Order Total:</b>				<b>\$1,764.50</b>
1004	3	Table	\$1,114.18	\$3,342.54
	2	Dresser	\$1,518.60	\$3,037.20
<b>Order Total:</b>				<b>\$6,379.74</b>
<b>Total for Eastern Region:</b>				<b>\$8,144.24</b>
Midwestern Region				
Order #	Quantity	Product	Unit Price	Total Sales
1003	4	Chair	\$325.16	\$1,300.25

Sales Report				
	2	Table	\$1,114.18	\$2,228.36
	1	Dresser	\$1,518.60	\$1,518.60
			<b>Order Total:</b>	<b>\$5,047.21</b>
<b>Total for Mid-western Region:</b>				<b>\$5,047.21</b>
Southern Region				
Order #	Quantity	Product	Unit Price	Total Sales
1005	2	Table	\$1,114.18	\$2,228.35
	1	Dresser	\$1,518.60	\$1,518.60
			<b>Order Total:</b>	<b>\$3,746.95</b>
1007	3	Chair	\$325.16	\$975.48
	1	Table	\$1,114.18	\$1,114.18
			<b>Order Total:</b>	<b>\$2,089.66</b>
<b>Total for Southern Region:</b>				<b>\$5,836.61</b>
Western Region				
Order #	Quantity	Product	Unit Price	Total Sales
1006	2	Dresser	\$1,518.60	\$3,037.20
	4	Chair	\$325.16	\$1,300.25
	1	Table	\$1,114.18	\$1,114.18
			<b>Order Total:</b>	<b>\$5,451.63</b>
<b>Total for Western Region:</b>				<b>\$5,451.63</b>

Using the security features, this one template could be used to meet the requirements of all the members of the *Sales* group. In this example, assume that the report has a query parameter that filters based on region.

For executives, the view above is what they would want to see when viewing the report, so a security level called *Management* could be created that retrieves data for all regions. This is accomplished by mapping all the available values for the region parameter to the *Management* level. In this example the values for the Eastern, Midwestern, Southern, and Western regions would be assigned to the security level. Note that in ReportDesigner, in order to supply more than one value to a query parameter, you need to define a multi-value parameter. For more information about this, please see Section 3.1.3.2.2.1 - Multi-Value Parameters.

For the account managers, different security levels could be created for each region. The above example has four regions, so security levels *East*, *Midwest*, *South*, and *West* would be added. Since account managers should not see the aggregate data, cells containing the summaries in the header would be hidden for the region levels using the cell-level security features discussed in Section 4.1.10.2 - Cell-Level Security. To make sure that each level returns data for the appropriate regions the parameter value for the Eastern region would be mapped to the "East" security level, the parameter value for the Midwestern region would be mapped to the *Midwest* security level, the parameter value for the Southern region would be mapped to the *South* security level, and the parameter value for the Western regions would be mapped to the *West* security level. For more information about setting security levels to apply parameter values, see Section 4.1.10.2.1 - Security Parameters.

In the Organizer, the members of the *Sales* group can be associated with the defined security levels. Then when the report is run, the security level for the user who has requested the report is applied and the parameter values associated with that level are automatically supplied to the query without prompting the user. For example, here is the same report run by an account manager in the eastern region:

Sales Report				
Eastern Region				
Order #	Quantity	Product	Unit Price	Total Sales

Sales Report				
1001	2	Chair	\$325.16	\$650.32
	1	Table	\$1,114.18	\$1,114.18
			<b>Order Total:</b>	<b>\$1,764.50</b>
1004	3	Table	\$1,114.18	\$3,342.54
	2	Dresser	\$1,518.60	\$3,037.20
			<b>Order Total:</b>	<b>\$6,379.74</b>
<b>Total for East-ern Region:</b>				<b>\$8,114.24</b>

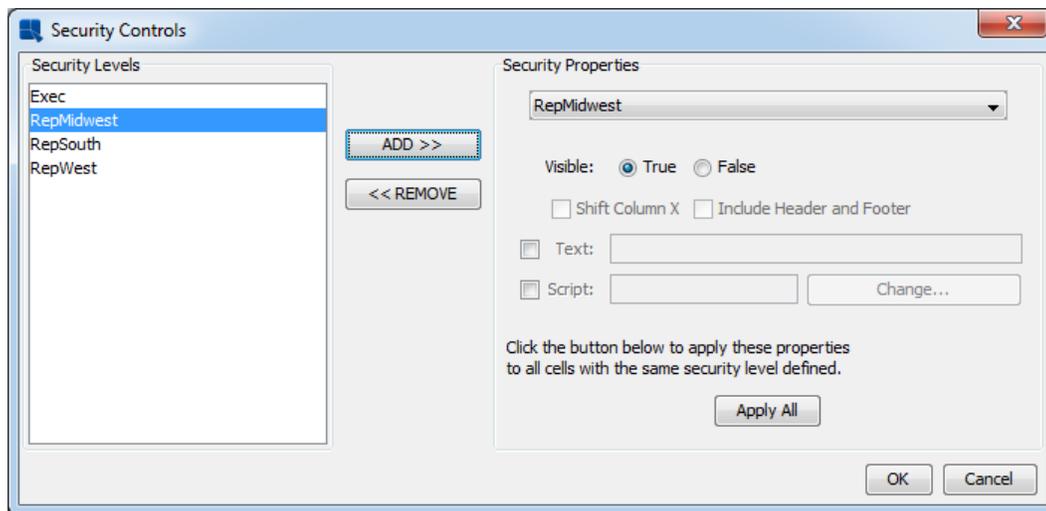
### 4.1.10.1. Security Levels

Security levels are unique groups of settings that can control cell and parameter behavior in reports. Levels are created by the administrator in the Organizer interface. For more information about setting security levels, see Section 2.3.3 - Security Levels. The administrator can also associate users and groups with specific security levels. For more information about this feature, please see Section 2.3.3.1 - Associating Security Levels with Users and Groups.

Within a report, different cell behaviors and filtering can be associated with a specific security level. At run-time, based on the user login, the security level assigned to that user is applied to the report.

### 4.1.10.2. Cell-Level Security

For any cell/visible column in the report, you can set different behaviors for different security levels. To apply security settings to a cell, either select the cell and select *Security* from the *Data* menu, or right click on the cell and select *Security* from the pop-up menu. This will bring up a dialog allowing you to specify security settings for that cell.



Cell Security Settings Dialog

The dialog contains a list of all the defined security levels in the Organizer.

Security levels can be added/removed/modified in the Organizer. See the Section 2.3.3 - Security Levels chapter for more details.

To add security settings for a report cell, select the level for which you want changes to take effect and click the *Add* button. The security level will then appear in the drop-down list on the left side of the cell security dialog. Now you can modify the cell for that security level. There are three basic controls available for cells. You can render the cell invisible for a security level. If the cell is visible, you can either provide custom text for the cell or you can apply a cell script. For more information about the cell scripts, please see Section 4.1.7 - Scripting. If you select to make a column (cell in Table Data section) invisible, make sure to select the column header along with the data cell so they both contain the same security setting, then you can check the *Shift Column X* option to make the other columns shift to the left when the security level is applied.



security level, you can select which parameters you want to secure using the check box next to the parameter. For each secured parameter, you can select the value(s) that you want to be associated with this security level. Once you finish assigning parameter values, click the *Ok* button to return to the design window.

### 4.1.10.3. Security for Sub-Reports & Drill-Down Layers

Template security settings can be automatically applied to sub-reports as well as drill-down layers. In both cases, you need to edit the sub-report or drill-down layer (by clicking the sub-report tab, or navigating to the drill-down layer) to add security. In the sub-report or drill-down layer, you can add security settings just as you would for the main report. If you want to apply security to both the main report and the sub-report/drill-down layer, you will need to assign the security levels with the same name(s) as in your main report. When a report is run with a security level, the security level will be automatically applied to the sub-report(s) and/or drill-down layer(s).

For more information about drill-down and sub-reports, please see Section 4.1.8 - Drill-Down and Section 4.1.9 - Sub-Reports.

### 4.1.10.4. Viewing Secured Templates

You can preview the effects of your security settings in the Preview window of the Report Designer. To do this, select *Set Preview Security Level* from the *Data* menu. This will bring up a dialog prompting you to select the security level that you want to use to view the report.



*Preview Security Level Dialog*

Select the level you want and then preview the report. You will see the report with security settings applied for the level you selected. This level will remain selected until you change the settings. Selecting the blank entry at the top of the list in this dialog will set the report to run without applying any security settings.

#### 4.1.10.4.1. Secured Report Designer

Only the administrator has access to the report security features. When other users access the Report Designer, it will run in secure mode. When the Report Designer is running in secure mode, users will only be able to preview reports at the level assigned to them. In addition, they will be unable to modify report cells with defined security settings or cell scripts that have been applied as security settings.

You can also call Report Designer in secure mode when launching it through the API. For more information about launching Report Designer through the API, see Section 8.1.5.8 - Calling Report Designer from Report API.

## 4.2. Chart Designer

### 4.2.1. Introduction to Chart Designer

The Chart Designer is a graphical user interface, launched within the Organizer that allows users to create and customize charts. The simple drop and drag style interface and extensive editing/formatting capabilities, makes chart design quick and easy.

#### 4.2.1.1. Working with Charts

It is important to note that in ERES there are two ways that you can use charts. Charts can either be placed within reports or designed and run as stand-alone entities. A unique feature of charts placed in reports is that they can use the report data as their data source. In this instance, the chart cannot be deployed outside of the report.

Charts that use their own data sources can be deployed independently from reports, either in menu pages or using image URLs to embed them in web pages.

#### 4.2.1.2. Starting Chart Designer

The Chart Designer interface is always loaded from within the Organizer. To start a new chart, you can click the *Chart Designer* button on the toolbar or select *View → Chart Designer*. You can also start Chart Designer when creating or editing data sources in the registry. The *View* button allows you to preview a data source and to build

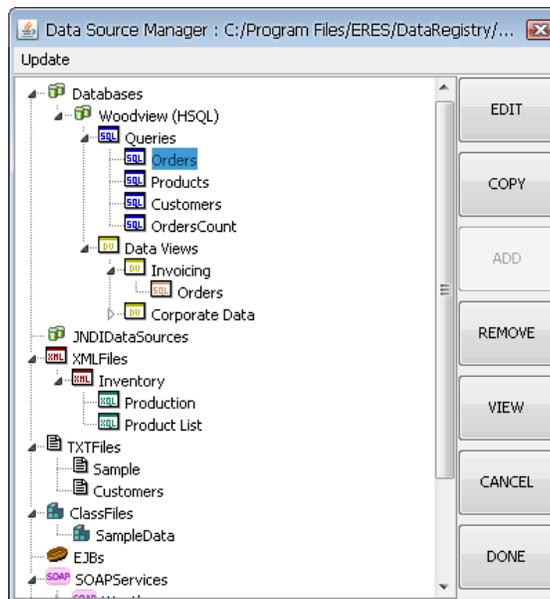
a chart or report. In addition, you can open the Chart Designer to edit a chart template file in the Organizer. To do so, first, select the file that you would like to open and then you can select File → Open File, right click on the file and select *Open File* from the pop-up menu, or double-click on the file.

If you are creating or adding an embedded chart within a report, the Chart Designer is launched from within Report Designer by either clicking on the *Insert Chart* button on the toolbar or by selecting Insert → Insert Chart.

### 4.2.1.3. Selecting a Data Source

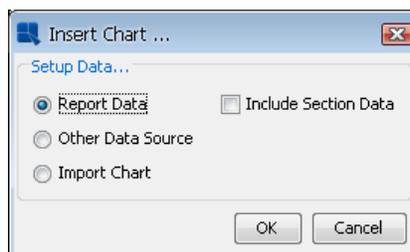
The first step in designing a chart is to select the data source from which the chart is to be drawn. The first time you select to start a new chart you will be prompted to select the data registry that you would like to use. If there are not currently any available registries (meaning that you have not created any or you do not have privileges to view any), you will be prompted to go the Data Registry Manager to create one. For more on data sources, please see Section 3.1 - Data in Organizer

Once you have selected a registry, a Data Source Manager window will open allowing you to select, add, or modify a data source that you would like to use for the chart or report. Note that the registry will not open if you are building a chart from the registry in the *Modify Data Sources* dialogs.



*Data Source Manager Window for Chart Designer*

If you are launching Chart Designer from within Report Designer (using the *Insert Chart* option), the following dialog will be presented:



*Chart Data Options*

The following options are available:

**Report Data:** This will plot the chart using the data contained in the report. This data includes all of the report columns and computed columns that have been added using the Report functions. When using report data for the chart, you also have the option to include section data. This will take any formulas or labels from the report section in which

the chart is placed and make them available to the chart. Data for labels and formulas is imported as additional columns to the chart input data. The cell ID is used as the column name.

However, when you use report data, the chart cannot be deployed independently from the report, and certain charting features like time-based zooming and histograms are not available.

**Other Data Source:**

This will allow you to select a different data source than the report data. If you select this option, you will go to the Data Source Manager window where you can create or select a data source for the chart.

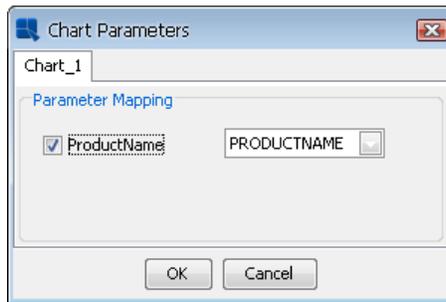
When you use an independent data source the chart can later be deployed independently from the report. Also, all of the charting features are available. Note that some features like drill-down are only functional when the chart is deployed independently from the report.

**Import Chart:**

Choose this option to import an existing chart to the report.

**4.2.1.3.1. Chart Parameter Linking**

If you are launching Chart Designer from within Report Designer (embedded chart) and you select to use an independent data source that is a parameterized query or class file, by default it will run with default values when the chart is run in the report. However, you can use the chart parameter linking feature for the chart to retrieve its parameter values based on column values in the report. This feature works in the same manner as the sub-report linking feature described in Section 4.1.9.4 - Linked Sub-Reports. To link a parameterized chart to a column, first, insert a parameterized chart into a report. Then, select *Chart Parameter Mapping* from the *Data* menu in Report Designer. This will bring up a dialog allowing you to map column fields from the report to query parameters in the chart.



*Chart Parameter Mapping Dialog*

The dialog contains a tab for each parameterized chart. Each parameter in the chart can be mapped to a different column field in the report.

**4.2.1.3.2. Transposing Data**

If you preview a data source in the Data Source Manager (see Section 3.1.2.1 - Using Data Source Manager to learn more about the Data Source Manager), you can transpose the data source. This is useful when you want to see what the transposition will do to your data source before creating a chart. There is a better way to configure data transposition for charts (see Section 4.2.3.1.1 - Data Transposition); however, if you enable the transposition in the data source preview and choose to create a chart, the transposition settings will **not** be lost and the chart will use the transposed data (and you will be able to modify the transposition later in the *Data Mapping* dialog).



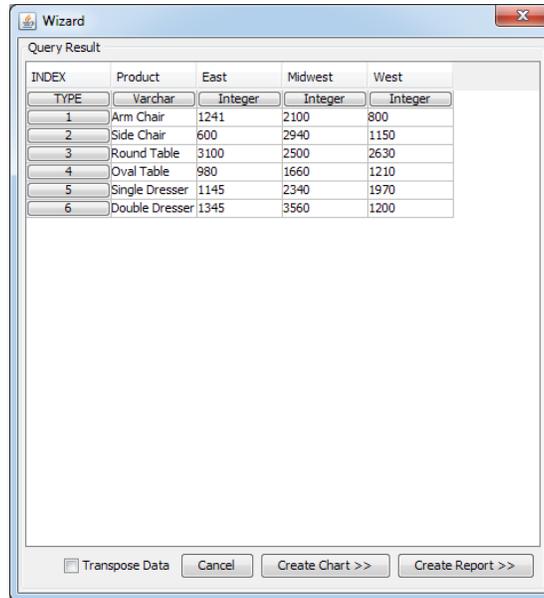
**Note**

This transposition method is available only in the data source preview dialog (that means that you have to open the Data Source Manager, select a data source and then click the *View* button). The other chart data transposition method is available in almost all charts (see Section 4.2.3.1.1 - Data Transposition for more details).

Once you preview your chart's data source, the next screen will present the first twenty records from the data source (With the exception of data views, which will require you to select fields and set conditions first). From this screen

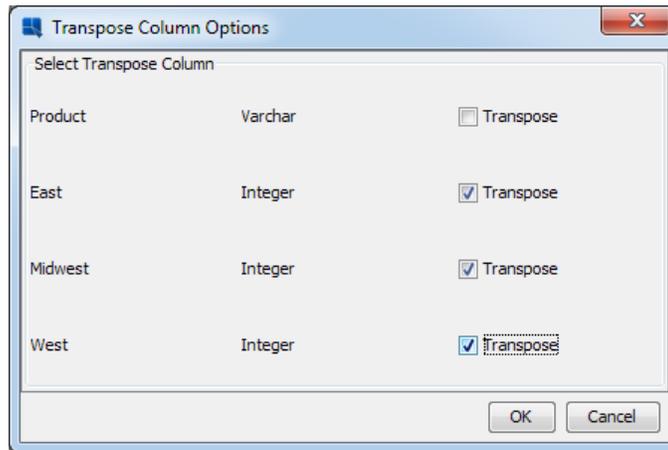
you can see all of the records returned by the selected source by checking the *Show All Records* box. Please note that the *Show All Records* button is hidden when the data source returns less than 20 rows of data.

Often, if you Are drawing data in spreadsheet (pivot table) format or from a crosstab report, you may want to use the column headers from the data source either as the categories or as a data series. For example, the following table is shown when a crosstab report is used as the data source.



*Data Table Dialog*

In order to be able to plot the regional column headers (East, Midwest, & West) in the chart, you will first need to transpose the data. To transpose the data, click the checkbox at the bottom of the data table window. A new window will open prompting you to select the columns that you would like to transpose.



*Select Transpose Dialog*

Check the columns that you would like to transpose and click *Ok*. You will return to the data table where you will see the results of the data transposition. Columns that you have selected are removed from the data table and the transposed columns are placed at the end of the table as new columns.



**Note**

Columns that you select must have the same data type.

INDEX	Product	ColumnLabel	Value
1	Arm Chair	East	1241
2	Arm Chair	Midwest	2100
3	Arm Chair	West	800
4	Side Chair	East	600
5	Side Chair	Midwest	2940
6	Side Chair	West	1150
7	Round Table	East	3100
8	Round Table	Midwest	2500
9	Round Table	West	2630
10	Oval Table	East	980
11	Oval Table	Midwest	1660
12	Oval Table	West	1210
13	Single Dresser	East	1145
14	Single Dresser	Midwest	2340
15	Single Dresser	West	1970
16	Double Dresser	East	1345
17	Double Dresser	Midwest	3560
18	Double Dresser	West	1200

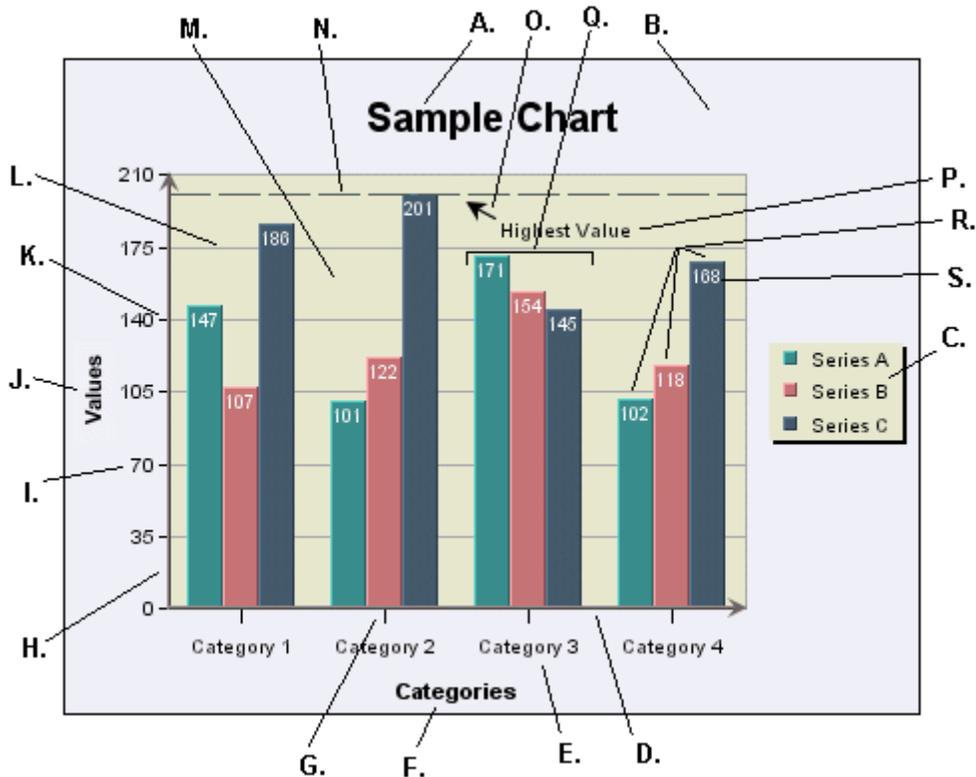
*Data After Transposition*

## 4.2.2. Charting Basics

This section covers some of the basics of charts including the different parts of a chart and the way in which tabular data either from the report or another data source is mapped to a chart.

### 4.2.2.1. What is a Chart

The following diagram illustrates the various components that make up a chart. Also, this diagram refers to various terms that will be used throughout this guide.



<b>A. Main Title</b>	This is the main title for the chart.
<b>B. Chart Canvas</b>	This is the background on which the chart is drawn. The canvas serves as the size/boundary for the chart and it is generally the same size as any exported image. You can modify the canvas color or place/add an image file as a background.
<b>C. Legend</b>	This is the chart legend. The legend shows your category or series names along with a color point. Secondary values, as well as added trend/control lines and control areas, can also be displayed in the legend.
<b>D. Category (X) Axis</b>	This is the X or category axis of the chart. Generally, the category axis plots the distinct entries from the dataset for which you want to plot values in the chart. (The values are generally plotted on the Y-axis.) Certain chart types such as bar and Gantt reverse this by drawing the categories on the Y-axis, and plotting the values on the X-axis. Other chart types like scatter and bubble plots, plot values on each axis to create a point in 2D or 3D space.
<b>E. X-Axis Labels</b>	These are the labels for the X-axis elements or categories.
<b>F. X-Axis Title</b>	This is the X-axis title.
<b>G. X Ticker</b>	These are the X-axis tickers. By default the tickers match each data point in the chart.
<b>H. Value (Y) Axis</b>	This is the Y or value axis of the chart. Generally, the Y-axis plots the values for each of the categories. By default the scale of the Y axis is generated to provide a best fit for the dataset; however, it can be manually adjusted. For combination charts, the second value axis is drawn at the right-hand side of the plot.
<b>I. Y-Axis Labels</b>	These are the labels for the Y-axis values.
<b>J. Y-Axis Title</b>	This is the Y-axis title.
<b>K. Y-Axis Ticker</b>	These are the Y-axis tickers.
<b>L. Y Grid</b>	These are grid lines drawn along each scale step in the Y-axis. Grid lines can also be drawn for the points on the X-axis (and Z-axis for 3D charts).
<b>M. Plot Area</b>	This is the area, bounded by the axes, where all the data points are plotted. You can fill the area with color and/or draw a border around the area, along with other options. The plot area can be moved and resized on the chart canvas.
<b>N. Control Line</b>	This is one of the special types of lines that can be added to a chart. In this instance, it is a control line that follows the highest value in a series. Control lines can also be drawn for averages and multiples of standard deviation. Users can also add a variety of trend lines to charts.
<b>O. Floating Line</b>	A floating line is an arbitrary line added to a chart. In this case it is being used as a pointer with an arrow. Floating lines move in relative position with the chart plot. They can also be used to create filled shapes.
<b>P. Annotation Text</b>	This is a piece of arbitrary text added to a chart (not labels or titles). You can place text anywhere in the chart canvas. Like floating lines, annotation text moves in relative position to the chart plot.
<b>Q. Category Elements</b>	This is the plot for a category element. There are three points plotted for each category because this chart has a data series.

**R. Data Series Elements**

These are the individual points that make up a category. A series allows groups of data to be plotted on a single chart. For more about information about categories and series, see Section 4.2.2.2 - Basic Data Mapping

**S. Data Top Labels**

These are labels placed at each data point in the chart that display the exact value for each point.

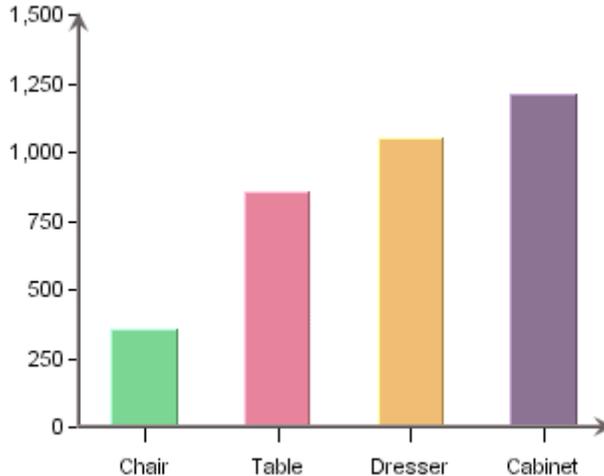
**4.2.2.2. Basic Data Mapping**

Data mapping is the way that raw data is rendered in the chart. Although data can be drawn from many sources, a chart looks for the basic structure of the data to be in the form of a table. Hence, data passed in as arguments, from the report or from XML files is converted to a table structure before mapping.

A basic set of data might look something like this:

Product	Sales
Chair	362
Table	862
Dresser	1052
Cabinet	1211

To plot this data in a chart, you would want to plot the **Sales** value for each entry in the **Product** column. Hence, the products are your category and the sales numbers are your values. In a chart you would map the **Product** column to the X (category) axis and the **Sales** column to the Y (value) axis. The resulting plot would look like this in a column chart:

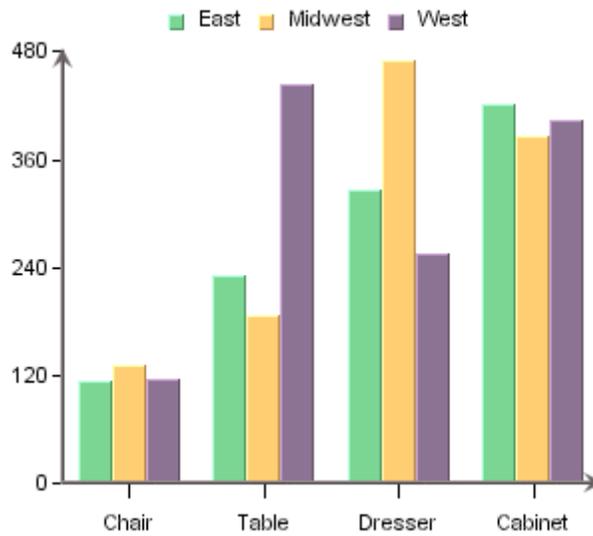


Here, a column is drawn to show the value for each distinct element in the category column. On top of the basic category values, additional information can be displayed in the form of a data series. For example, say that there is another element to our data that shows sales data not only for product, but also over a sales region. Our adjusted table would look like this:

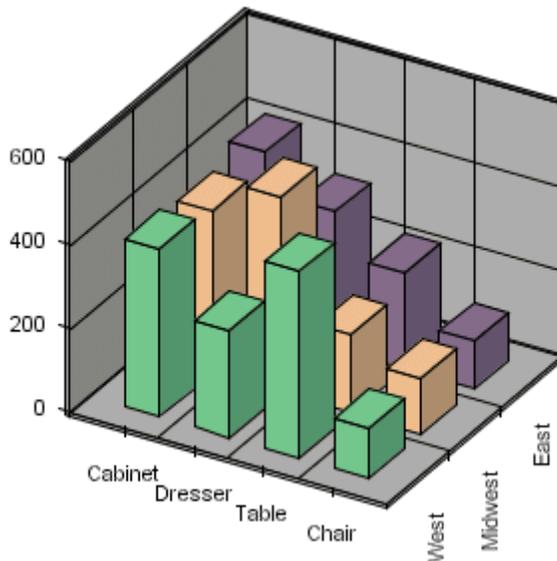
Product	Region	Sales
Chair	East	114
Chair	Midwest	131
Chair	West	117
Table	East	231

Product	Region	Sales
Table	Midwest	187
Table	West	444
Dresser	East	327
Dresser	Midwest	469
Dresser	West	256
Cabinet	East	422
Cabinet	Midwest	386
Cabinet	West	403

In order to show the value for each region per product we could add the **Region** column to the data mapping as a data series. Doing this gives us the following chart:



Now each category has three data points, one for each region. For two-dimensional charts the series is always displayed in-line. In three-dimensional charts, the series is drawn on the Z-axis by default, although it can be drawn in-line as well. Below, is the same chart show in 3D.



In this chart, the data series is drawn along the Z-axis. Note that the order of the categories has been changed to provide a better view of the data. This is the basic concept behind data mapping. Most chart types use this mapping technique or mapping options similar to this. Detailed data mapping instructions for each chart type are available in Section 4.2.3 - Chart Types and Data Mapping.

### 4.2.2.3. Saving and Exporting Charts

There are several options available both for saving chart definitions and exporting charts as image files. More information on how to save and export charts in the Chart Designer can be found in Section 4.2.6 - Saving & Exporting Charts.

#### 4.2.2.3.1. Saving Chart Definitions

There are two primary methods which you can use to store chart definitions created in ChartDesigner: either as chart or as template files.

**Chart files** Chart files save the chart in a binary file called `filename.cht`. A chart file stores both the definitions of the chart (type, dimension, etc.) as well as the data that was used to create the chart. Hence, chart files are portable. Any time you open a chart file, it will open with the original data that was used to create the chart. After opening the chart, you can refresh the data from the source or change the chart's data source entirely.

**Template files** Template files save the chart in a binary file called `filename.tpl`. A template file stores only the chart definitions and stores only 10 records of data with the chart. Hence, any time a template file is opened it will try to connect to the original data source to retrieve the data. Because of this, template files can be less portable than chart files. Template files can also be used to pass chart attributes from one chart to another. To do this, you can apply a template to a chart. This will carry over many of the attributes of the template to the current chart. For more information about applying templates, please see Section 4.2.6.2.1 - Working with Templates.

**PAC files** PAC files save the chart the way that it is ready for deployment. A `.PAC` file takes the chart and all the supplementary files associated with it - background images, dynamic drill-downs, and parametric drill-downs and places them in a single binary file. This configuration makes it easy to deploy charts and move them between ERES installations. If you reopen a `.PAC` file in Chart Designer, it will expand all the files back to their original location. Therefore, the target directories will need to exist for a `.PAC` file to be opened successfully. Refrain from renaming the `.PAC` file after its creation because this will lead to problems during expansion.

In addition to saving chart definitions in binary format, chart files and template files can also be saved in XML format. These XML chart definition files can be modified outside of Chart Designer or Chart API.

#### 4.2.2.3.2. Generating Image Files

Generally, charts can be deployed via the Web in one of two ways - either as applets or by generating image files. Applets can directly load chart definitions, either template or chart files. For more about using applets see the Chart Viewer section in Section 7.8 - Chart Viewer. For image files, ERES can render charts in the following formats:

**GIF** ERES can generate GIF images using one of two compression methods - RLE or LZW. The LZW method is faster and produces smaller files; however, its use is protected by patent. You must obtain a license from Unisys in order to unlock the LZW compression. By default, GIF files are generated using RLE compression.

**JPEG** JPEG is another popular image format. It is a higher resolution image format than GIF and it is not patent protected. When generating a JPEG file you can specify the quality and compression of the file. The higher the quality, the larger the file.

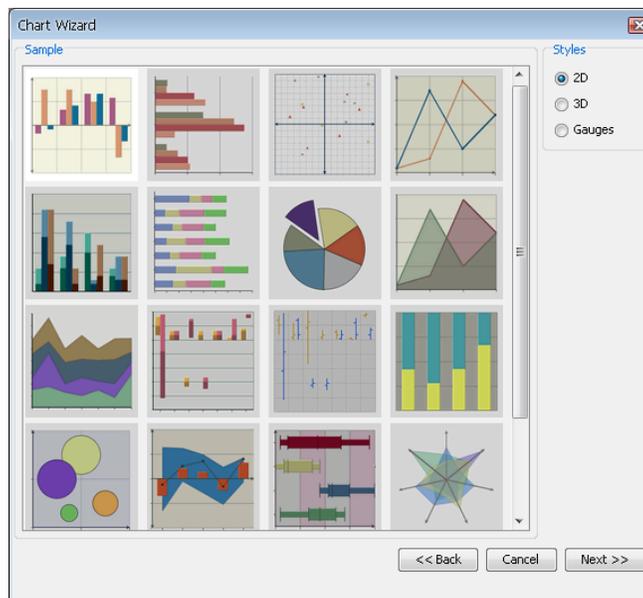
**PNG** PNG is an image format that is less popular but it can be displayed in most browsers. It is a high-quality image with a smaller file size than JPEG. There are three different compression options available for this format.

- SVG** SVG (Scalable Vector Graphics) is a relatively new image format that saves the image as vectors in an XML-based text format. Generally, you will need a browser plug-in to view these images.
- SWF** SWF is an Adobe Flash file. The flash format is vector based and it allows the chart to be resized after export. Also, flash allows for high-resolution printing and produces a small file size.
- BMP** This is a Windows bitmap format.
- WMF** WMF is the Windows Meta File format. This can be used for import/export into MS Office documents.

In addition to static images, ChartDesigner also allows the chart data to be exported as a text file, PDF, or an XML file.

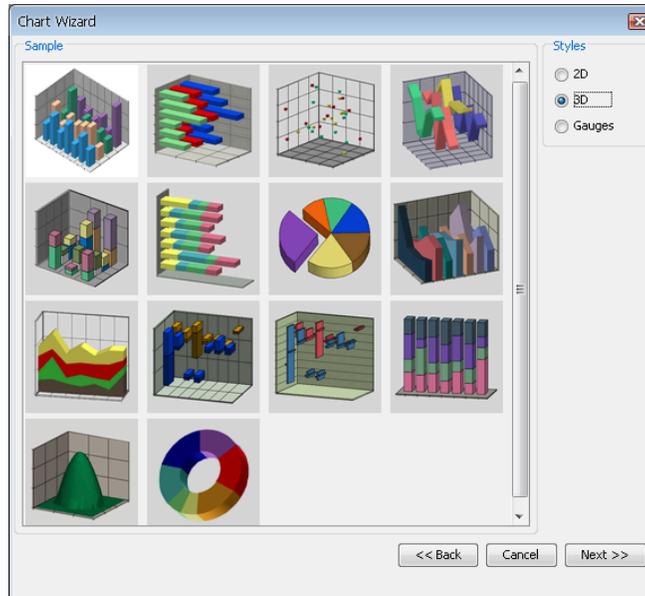
### 4.2.3. Chart Types and Data Mapping

Once you have selected the data that you would like to use for the chart, the next step in the Chart Wizard is to select the chart type that you would like to use. You will be presented with a dialog that allows you to specify the chart type.



*Two-Dimensional Chart Types Selection Dialog*

Each chart type represents a different way in which the data points are plotted to give a pertinent representation to all kinds of data. The different types of charts are broken down by dimension. In addition to basic chart types, users can create many different types of composite/combination charts by adding secondary values/series to the chart. You can toggle between the chart categories by selecting either *2D*, *3D* or *Gauges* in the right-hand side of the chart types dialog.



*Three-Dimensional Chart Types Selection Dialog*

This dialog enables you to select your chart type by either selecting the image and clicking *Next* or by double clicking on the chart image. You can select from one of the following chart types:

- Column
- Bar
- XY(Z) Scatter
- Line
- Stack Column
- Stack Bar
- Pie
- Area
- Stack Area
- High Low
- HLCO
- Percentage Column
- Doughnut
- Surface (Three-Dimensional Only)
- Bubble (Two-Dimensional Only)
- Overlay (Two-Dimensional Only)
- Box (Two-Dimensional Only)
- Radar (Two-Dimensional Only)
- Dial (Two-Dimensional Only)
- Gantt (Two-Dimensional Only)

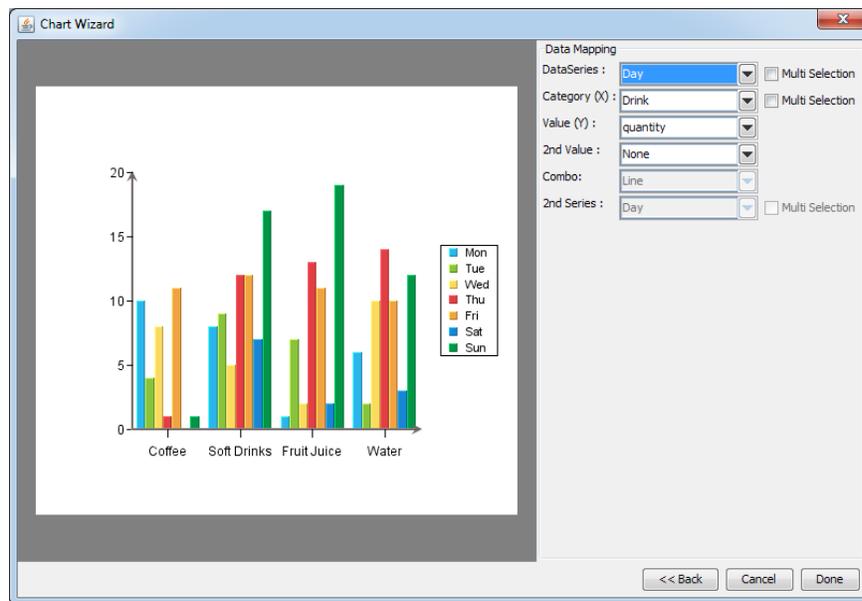
- Polar (Two-Dimensional Only)
- Circular Gauge
- Square Gauge
- Semi Circular Gauge
- Square Gauge
- Quarter Circular Gauge

Each of the chart types is described in detail later in this chapter, starting with Section 4.2.3.2 - Column Charts

For information on gauges, see Section 4.2.3.20.2 - Gauges

### 4.2.3.1. Data Mapping

After you have selected the chart type, the next step is to specify the data mapping for the chart. Data mapping is the way that the selected data source is rendered in the chart. The basics of data mapping are described in Section 4.2.2.2 - Basic Data Mapping. The data mapping screen in the Chart Wizard allows you to set the mapping options and also preview the results.



*Data Mapping Dialog*

The left-hand side of the dialog shows a preview of your chart. The right-hand side shows which columns from your data source have been selected to plot in the different chart elements (series, category, value, etc). By default, ERES will select the first available columns, based on the data type, to plot. Changing the data mapping is easy - click on the down-facing arrow next to a data field and select a different field. The chart preview in the left-hand part of the data mapping dialog will be updated immediately.

The specific mapping options vary for each chart type and are discussed later in this chapter starting with Section 4.2.3.2 - Column Charts.

#### 4.2.3.1.1. Data Transposition

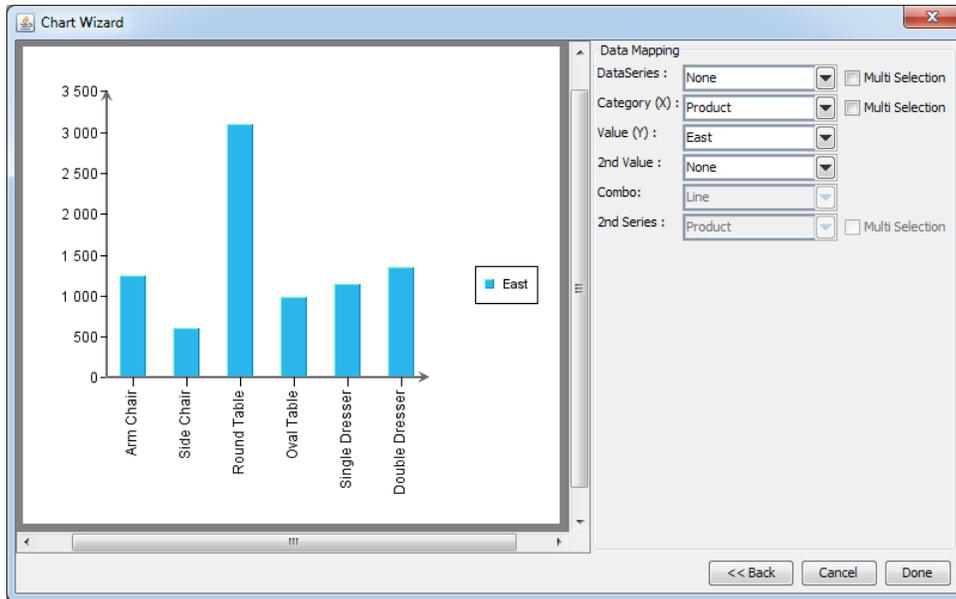
You can also set data transposing from the data mapping dialog. Transposing allows you to plot multiple data source columns in one chart without modifying the data source.

For example, we have a data source that looks like this:

INDEX	Product	East	Midwest	West
TYPE	Varchar	Integer	Integer	Integer
1	Arm Chair	1241	2100	800
2	Side Chair	600	2940	1150
3	Round Table	3100	2500	2630
4	Oval Table	980	1660	1210
5	Single Dresser	1145	2340	1970
6	Double Dresser	1345	3560	1200

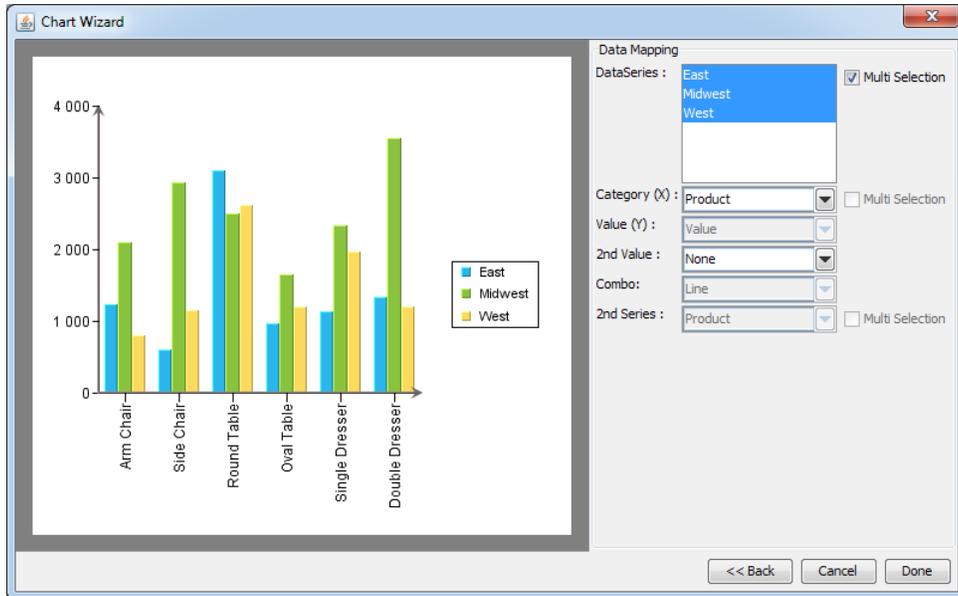
*Example Data Source*

We would like to plot data from all regions in one chart. The default mapping without data transposing does not allow us to do that. We would have to plot only one region in the chart or modify the data source, so we will have to transpose the data source.



*Default Chart Data Mapping*

To transpose the data, select the *Multi Selection* option next to the *Data Series* field. The field will change to a list allowing us to select several columns at the same time. We will just select all region columns and the chart will look like this:



Default Chart Data Mapping

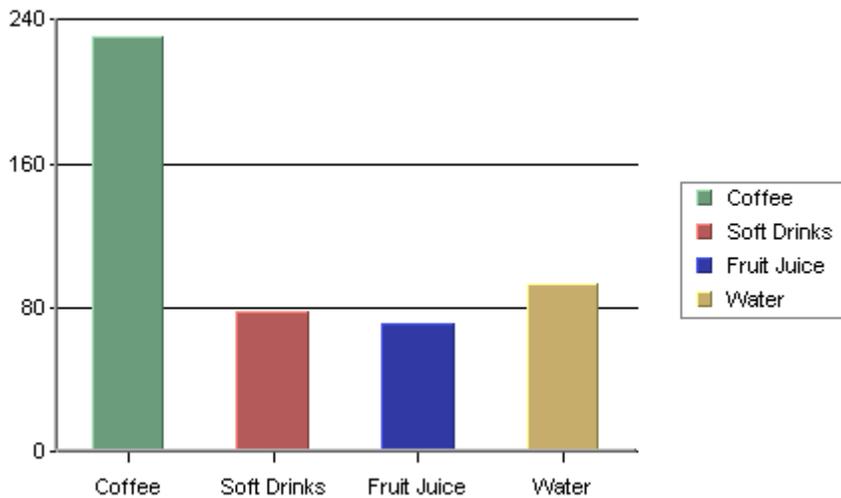


**Note**

Only one chart element (such as data series, category, value, or second series) can be transposed at a time. If you select the *Multi Selection* checkbox for one chart element, the *Multi Selection* checkboxes will be deactivated for other chart elements.

Transposing is not available for drill-down charts.

**4.2.3.2. Column Charts**



Column Chart (Without Data Series)

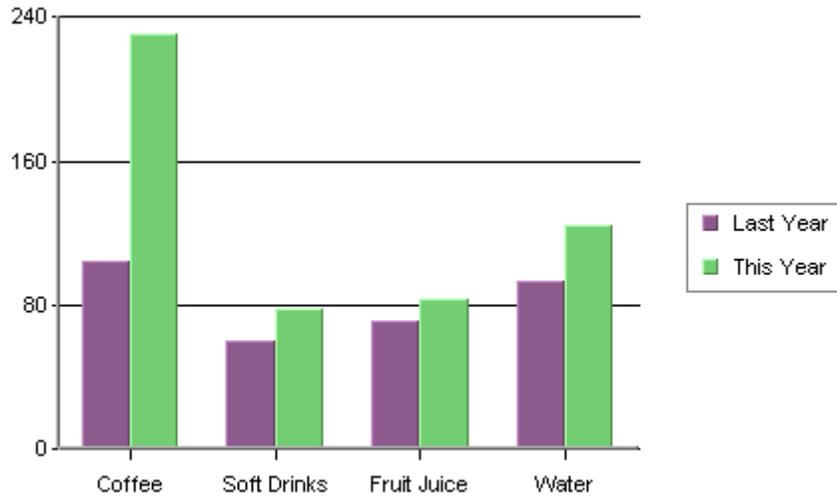
A column chart displays each row in the data table as a vertical bar (or column). The categories are listed along the X-axis and the values plotted along the Y-axis. Column charts are good for comparing discrete values for different groups. Each group is represented by a different color.

In a two-dimensional chart, if a data series has been selected then the entire series for a single category will be displayed in the XY plane. If a three-dimensional column chart is being used then all the vertical bars in a given data series are drawn using the same color along the Z-axis. If a data series is not present in the chart, the categories will be represented by the same color by default. Different colors for the categories are possible to set by clicking the



Change *chart options* icon on the toolbar (or selecting Format → Chart Options), and unchecking the *Single Color For All Categories* option.

In the examples given, we have chosen Drink as the category variable and Value as the value variable (most examples use the `Sample.dat` file included in the ERES installation). Year has been selected as the data series variable for the chart below. Therefore, each name has two columns shown: one for *last year* and one for *this year* which are the only two values present in the data series column.



Column Chart with Data Series

#### 4.2.3.2.1. Data Mapping

Data mapping for column charts is fairly straightforward. It is similar to the examples first presented in Section 4.2.2.2 - Basic Data Mapping. For column charts, the following options are available:

**Data Mapping**

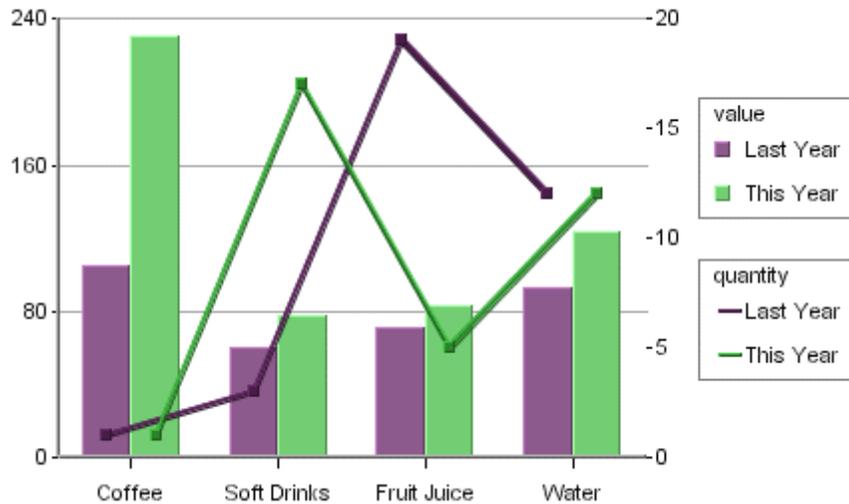
- DataSeries : year  Multi Selection
- Category (X) : Drink  Multi Selection
- Value (Y) : value
- 2nd Value : quantity
- Combo: Line
- 2nd Series : Day  Multi Selection

Mapping Options for Column Charts

- Data Series:** Choose a data column whose distinct values will determine the number of data series in the chart. Each element in a data series is drawn using the same set of colors and other attributes.
- Category (X):** Choose a data column whose distinct values will determine the categories.
- Value (Y):** Choose a data column to provide (Y) values for each category.
- 2nd value:** Add a second value to create a combination chart.
- 2nd Series:** Choose another column to be series for the secondary chart. This option is applicable only if the secondary chart is an overlay chart.
- Combo:** Choose the chart type for the secondary chart. For column charts the combo options are *Line* and *Overlay*.

The data mapping also allows you to transpose the data (in other words: to select several columns for a single category). To learn more about data transposition, please see Section 4.2.3.1.1 - Data Transposition.

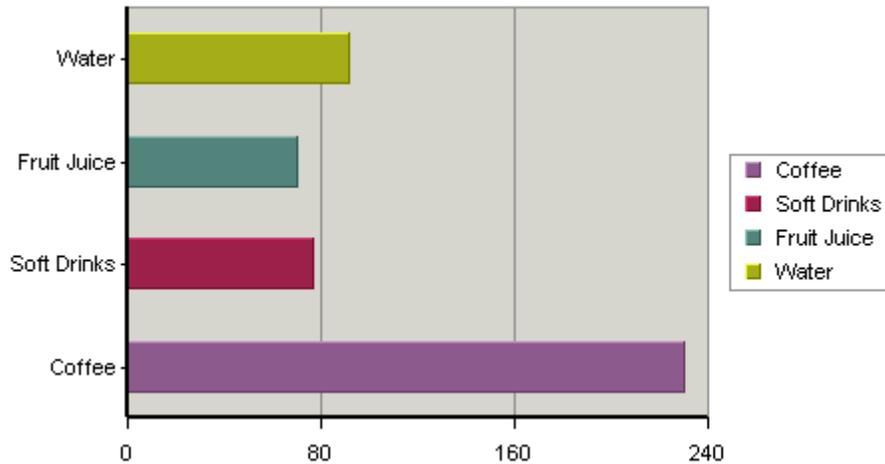
The last three data mapping options allow you to add a second value to the chart. ChartDesigner supports secondary values for all chart types except pie, radar, bubble, dial, surface, and scatter charts. In the example below, a second value of Quantity has been added to our column chart.



Column Chart With 2nd Value

As you can see, the second axis labels are drawn on the right-hand side of the plot area (you can choose to make this axis visible). Usually, the second value will share the same categories and series as the primary value. However, for two-dimensional column, stack column, stack area, high low, HLCO, and percentage column charts you can specify an overlay combination, which allows you to specify a second series. For more on overlay charts, please see Section 4.2.3.17 - Overlay Charts.

### 4.2.3.3. Bar Charts



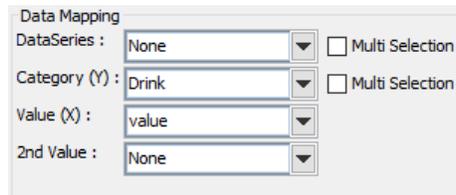
Bar Chart

A bar chart is essentially the same as a column chart, except that horizontal bars are drawn in the chart as opposed to the vertical bars which are used in a column chart. In a bar chart, the categories are plotted along the Y-axis and the values along the X-axis.

Just as for a column chart, if a data series has been selected then the entire series for a single category is displayed in the XY plane. If a three-dimensional bar chart is being used, all the horizontal bars in a data series are drawn using the same color. Each category is drawn using a different color. If the data series is not present in the chart, the categories will be represented by the same color by default. Different colors for the categories are possible to set by

clicking the  *Change chart options* icon on the toolbar (or selecting Format → Chart Options), and unchecking the *Single Color For All Categories* option.

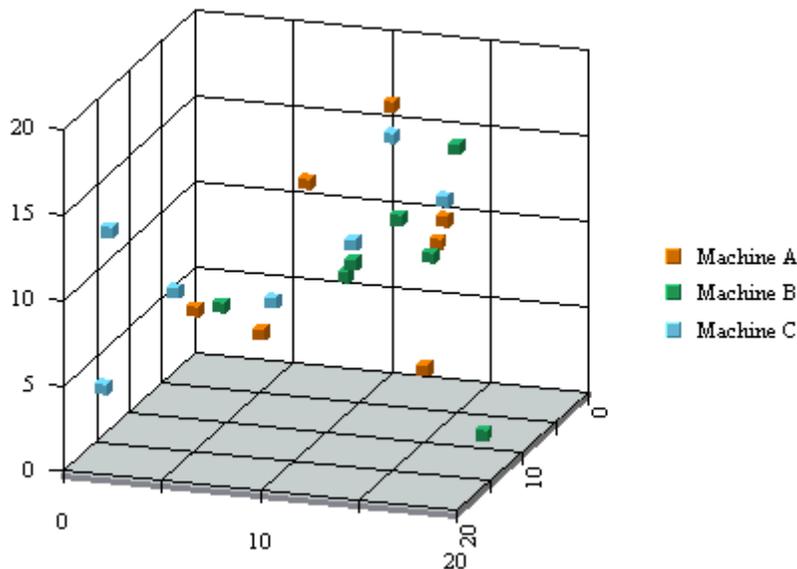
#### 4.2.3.3.1. Data Mapping



*Mapping Options for Bar Charts*

The mapping for this chart type is similar to that of a column chart, except the category (X) and values (Y) under column chart become category (Y) and values (X) respectively. This is because values are represented vertically in a column chart, but horizontally in a bar chart. Please note that the *2nd Series* and *Combo* options are not available for bar charts. This is because the only combination available with bar charts is a line.

#### 4.2.3.4. XY(Z) Scatter Charts

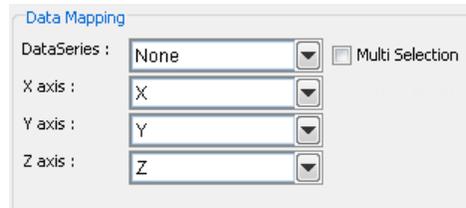


*XY(Z) Scatter Chart*

In an XY(Z) scatter chart, each selected row in the data table defines a point in two or three-dimensional space. Thus each column must contain either numeric or date/time values. A marker represents each point. The data columns that are in each row of the data table determine the spatial position of the marker.

Optionally, another data table column can be chosen to separate the markers into groups. Elements of each group have the same value on this column which is referred to as the data series column. Markers in the same group are drawn using the same drawing attributes; in other words, using the same shapes and colors. The X-axis scale of a scatter chart is linear. This means that unlike other chart types, the data points may or may not be evenly spaced along the X-axis.

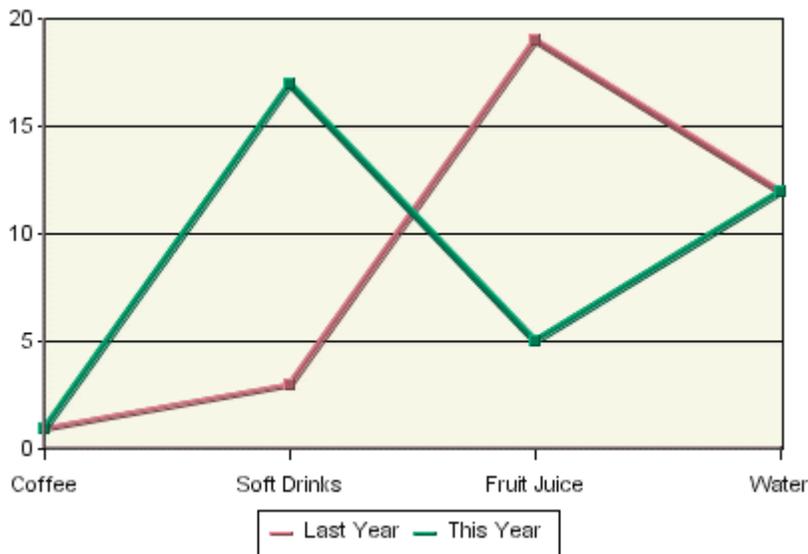
### 4.2.3.4.1. Data Mapping



*Mapping Options for Scatter Charts*

In a scatter chart, the X-axis, Y-axis, and Z-axis values determine the X, Y, and Z coordinates of a point respectively. The data series box allows you to choose a data column whose distinct values will determine the number of data series in the chart. Each element in a data series is drawn using the same set of drawing attributes( e.g., colors and markers).

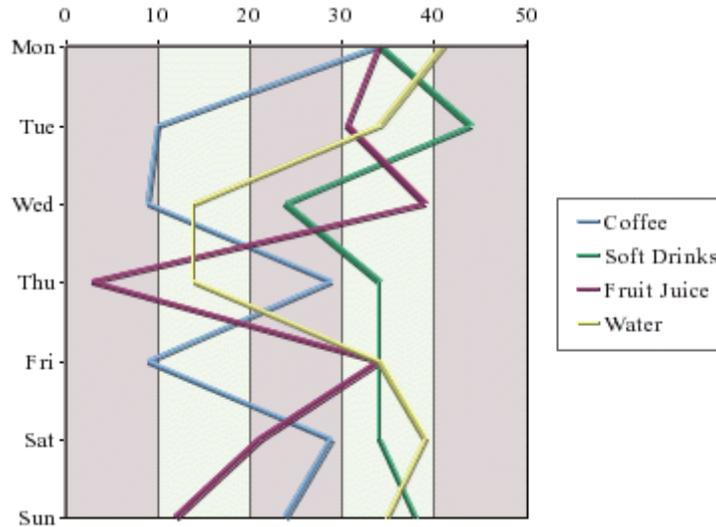
### 4.2.3.5. Line Charts



*Line Chart*

Line charts present data in a manner similar to column charts. For a two-dimensional line chart, a marker denotes a data point for each category. The value column, which must be numeric, determines the height of a marker. Each marker is joined with a line. If the chart has a data series, a separate line is drawn for each element in the series. Please note, the markers (points) are not shown by default. You can enable showing markers (points) in the *Line and Point* dialog. This dialog can be opened from the *Format* menu or by clicking the  *Format line and points* icon on the toolbar.

ChartDesigner allows 2D line charts to be displayed vertically in addition to the default horizontal setting. To use this feature in Chart Designer, create a line chart and then select *Format* → *Chart Options*. Next, select *Vertical*. The chart is rotated clockwise 90 degrees.



*Vertical Line Chart*

A three-dimensional line chart is an extension of its two-dimensional counterpart. It contains no additional information. The markers disappear (they are not available for 3D line charts) and thicker lines, spaced apart on the Z-axis, replace the thin lines.

#### 4.2.3.5.1. Data Mapping

**Data Mapping**

DataSeries :   Multi Selection

Category (X) :   Multi Selection

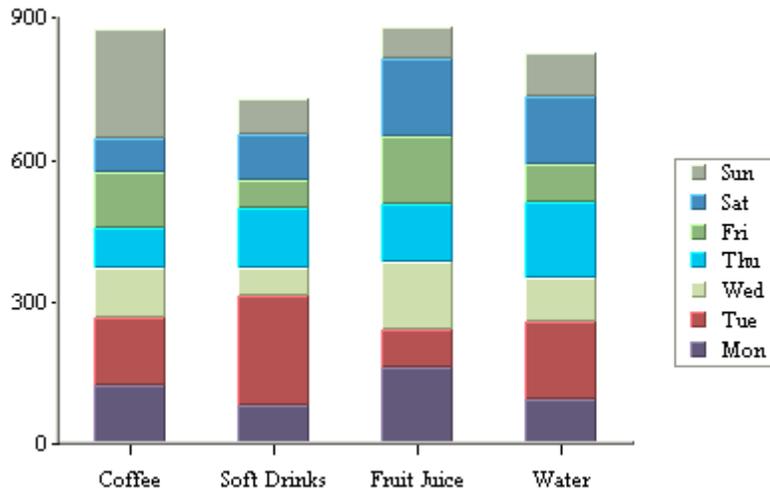
Value (Y) :

2nd Value :

*Mapping Options for Line Charts*

Data mapping for line charts is almost exactly the same as for column charts (discussed in Section 4.2.3.2.1 - Data Mapping). Line charts, however, do not have the *2nd Series* and *Combo* options. This is because the only combination available with line charts is a line. To create line combinations with other chart types (bar, column, stack column, etc) select the other chart type as the primary chart, and select *Line* as the *Combo* option.

### 4.2.3.6. Stack Column Charts



Stack Column Chart

A stack column chart is a derivative of a column chart in which each vertical column comprises a stack of bars. Each selected row in the data table is represented by a component of a chart column. For a two-dimensional stack column chart, the categories are plotted on the X-axis and the values on the Y-axis. The value column, which must be numeric, determines the length of each bar component. A third column, known as the sum-by column, represents a group of values for each category. A stack for a given category value is built up stacking the sum-by values for that category value. Each stack component of a chart column has a distinct sum-by value denoted by a distinct color. Each row in the data table must have a unique pair of values on both the category and sum-by columns. Components with negative values are separated from components with positive values and they are stacked up in the “negative” direction below the X-axis.

In our example above, we have chosen types of drink as category value and day representing the sum-by values. Another independent category (to be displayed on the Z-axis) may be optionally specified based on a fourth column as a data series. In such a case, each row in the data table must have a unique triplet of values on the category, sum-by and data series columns. In a two-dimensional chart, the entire data series for a single category is displayed side-by-side in the XY plane. In a three-dimensional stack column chart, the data series is displayed along the Z-axis.

#### 4.2.3.6.1. Data Mapping

In stack column charts, each category is further subdivided into components. Hence, there is an additional sum-by option used to determine the sum-by values for the chart.

The Data Mapping panel includes the following settings:

- DataSeries : None
- Category (X) : Drink
- Sum by : Day
- Value (Y) : volume
- 2nd Value : None
- Combo: Line
- 2nd Series : drink

Mapping Options for Stack Column Charts

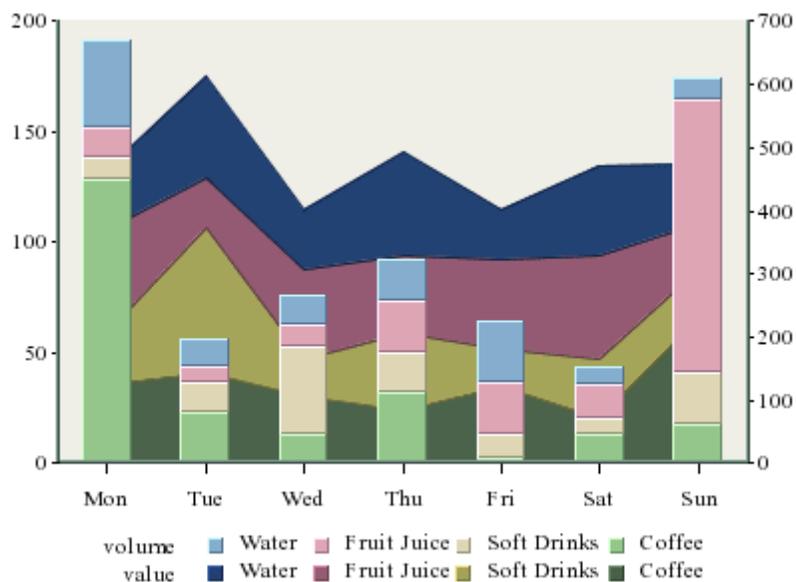
The data mapping options are as follows:

**Data Series:** Allows you to choose a data column whose distinct values will determine the number of data series in the chart. Each element in a data series is drawn using the same color.

- Category (X):** Allows you to choose a data column whose distinct values determine the categories. Values from this data column are used to calibrate the X-axis. Each category in a data series is drawn as a distinct column in the chart.
  
- Sum-by:** Allows you to choose a data column whose distinct values determine the components in each category. Each distinct value in this column determines a distinct stack component of a column.
  
- Value (Y):** Choose a data column to provide values for each category.
  
- 2nd value:** Add a second value to create a combination chart.
  
- 2nd Series:** Choose another column to be series for the secondary chart. This option is applicable only if the secondary chart is an overlay chart.
  
- Combo:** Choose the chart type for the secondary chart. For stack column charts the combo options are *Line*, *Stack Area*, and *Overlay*.

The data mapping also allows you to transpose the data (in other words: to select several columns for a single category). To learn more about data transposition, please see Section 4.2.3.1.1 - Data Transposition.

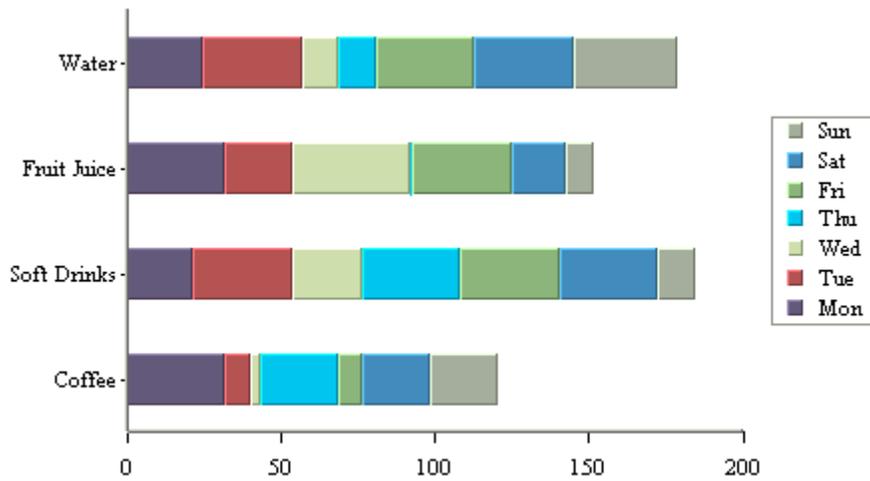
Stack column charts have a unique combination option which allows users to plot a second value as a stack area chart. Both values share the same category and sum-by values. This chart can compare component based categories for different values in the data.



*Stack Column-Stack Area Combination Chart*

With this combination chart you can specify to hide particular sum-by component in the chart to achieve a strip area effect.

### 4.2.3.7. Stack Bar Charts



Stack Bar Chart

Like a stack column chart, the stack bar chart displays the chart categories as a sum of different stacked values - the sum-by column in the data source. The difference for a stack bar chart is that the categories are plotted on the Y-axis and the values are plotted along the X-axis.

#### 4.2.3.7.1. Data Mapping

**Data Mapping**

DataSeries :   Multi Selection

Category (Y) :   Multi Selection

Sum by :   Multi Selection

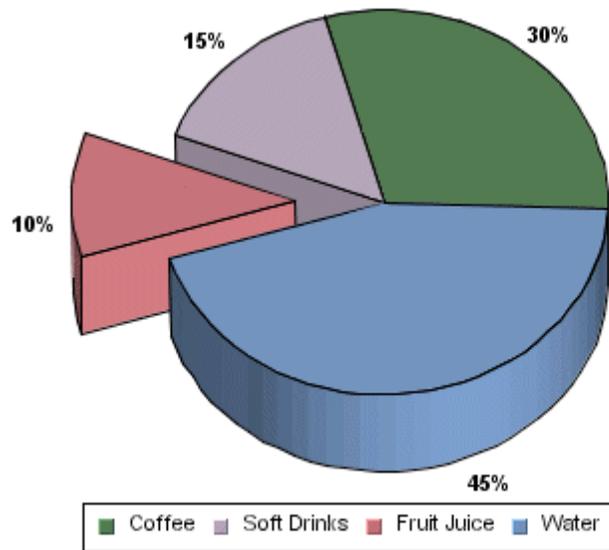
Value (X) :

2nd Value :

Data Mapping Options for Stack Bar Charts

The mapping for this chart type is similar to that of a stack column chart, except the category (X) and values (Y) under column chart become category (Y) and values (X), respectively. This is because values are represented vertically in a column chart, but horizontally in a bar chart. Please note that the *2nd Series* and *Combo* options are not available for stack bar charts. This is because the only combination available with stack bar charts is a line.

### 4.2.3.8. Pie Charts

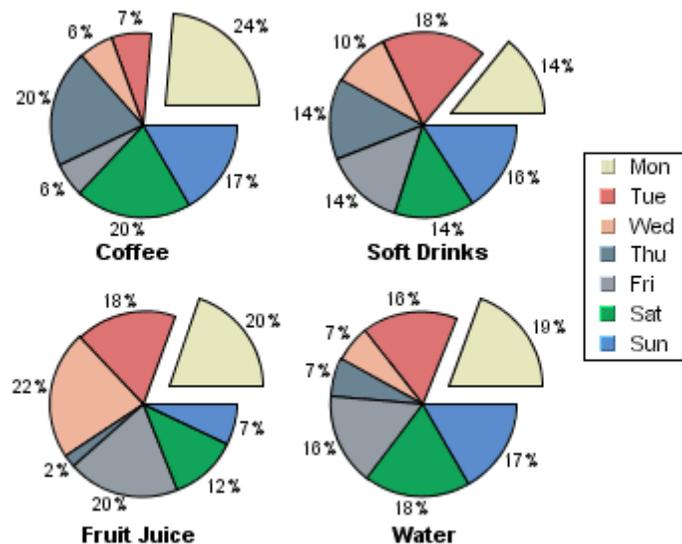


*Pie Chart*

In a pie chart, each row in the data is represented as a pie sector. The pie chart requires a category column and a value column. The value column must be numeric. The number of distinct values in the category column determines the number of pie sectors in the chart. All the values are summed up and each slice is assigned an angle according to its share in the total. Thus, the value column for each category determines the size of the slice representing that category.

A three-dimensional pie chart is simply an extension of its two-dimensional counterpart and contains no extra information.

Pie charts can also have a data series. If you specify a series in the data mapping, a separate pie will be drawn for each category element and it will be comprised of the series elements. A pie chart with a data series is displayed below:



*Pie Chart With Data Series*

When a series is present, all the separate pies are drawn on a single plot area and resized in proportion with the plot. You can choose to either stack the different pies or draw them in a line.

### 4.2.3.8.1. Data Mapping

**Data Mapping**

DataSeries :   Multi Selection

Category (X) :   Multi Selection

Value (Y) :

*Mapping Options for Pie Charts*

For pie charts, the mapping is as follows:

**Data Series:** Allows you to choose a data column whose distinct values will determine the number of data series in the chart. Each element in a data series is drawn using the same color.

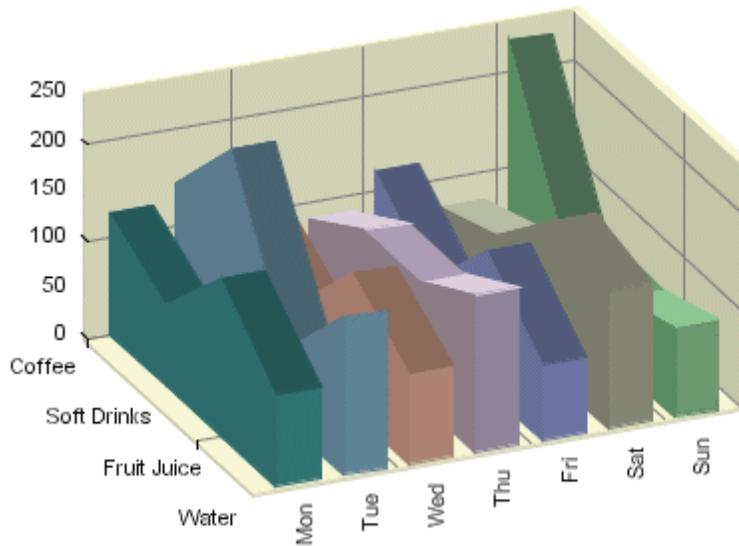
**Category (X):** Allows you to choose a data column whose distinct values determine the various categories.

**Value (Y):** Allows you to choose a data column to provide values for each category.

There is no *2nd Value* option, as you cannot make any combination charts with a pie chart.

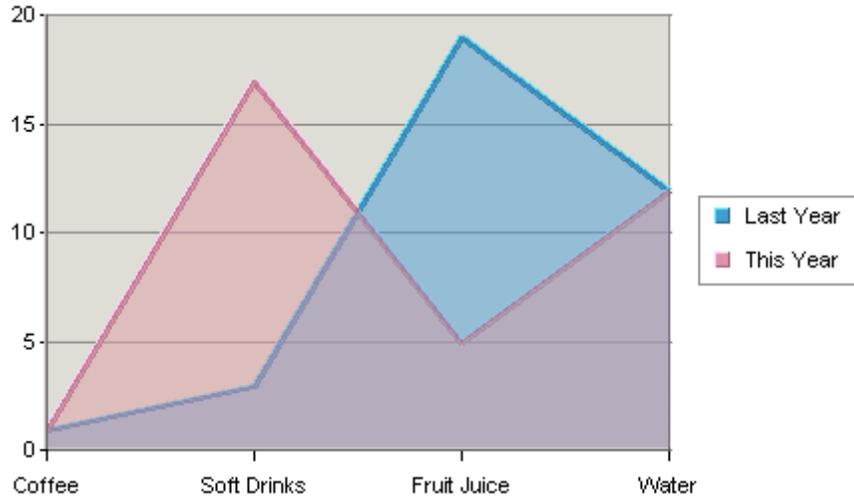
The data mapping also allows you to transpose the data (in other words: to select several columns for a single category). To learn more about data transposition, please see Section 4.2.3.1.1 - Data Transposition.

### 4.2.3.9. Area Charts



*Three-Dimensional Area Chart*

A three-dimensional area chart may be viewed as a derivative of a column chart. A three-dimensional area chart may be constructed from a three-dimensional column chart in the following manner. For each data series (Z-axis) value, the tops of all the columns are joined together by a thick line. The columns are then removed and the area (in the XY plane) under each line is filled with a distinct color.



*Two-Dimensional Area Chart*

A two-dimensional area chart is essentially a projection of its three-dimensional counterpart viewed along the Z-axis. As a result, a two-dimensional area chart may sometimes hide a data series value altogether. Therefore, it must be used with caution. The chart above illustrates this point: in a two-dimensional display some parts of each series are concealed by a series in front. To ameliorate this problem, you can change the ordering of the data series (see Section 4.2.4.8.3 - Data Ordering) or set the area translucent (see Section 4.2.4.1.3 - Format Menu) as in the example above.

#### 4.2.3.9.1. Data Mapping

**Data Mapping**

DataSeries :   Multi Selection

Category (X) :   Multi Selection

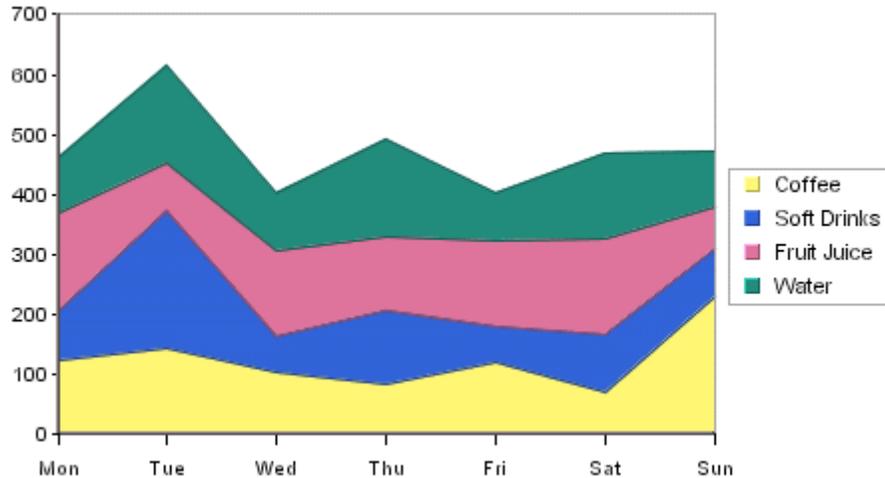
Value (Y) :

2nd Value :

*Mapping Options for Area Charts*

Data mapping for area charts is almost exactly the same as for column charts (discussed in Section 4.2.3.2.1 - Data Mapping). However, area charts do not have the *2nd Series* and *Combo* options. This is because the only combination available with area charts is a line.

### 4.2.3.10. Stack Area Charts



Stack Area Chart

A two-dimensional stack area chart may be viewed as a derivative of a two-dimensional stack column chart but without a data series. This chart may be constructed from a stack column chart as follows: The tops of each stack component that have the same color are joined together by lines. The stack columns are removed and the area between each set of lines is filled with a distinct color.

A three-dimensional stack area chart can have a data series and can also be derived from a three-dimensional stack column chart using the process stated above. In this case, the stack columns for each distinct data series value (a fixed Z-axis value) are joined separately, giving rise to multiple stacks.

Both two-dimensional and three-dimensional stack area charts may be constructed using a similar set of data as needed for their respective stack column chart counterparts. Note, as stated earlier, that a two-dimensional stack area chart cannot have a data series.

#### 4.2.3.10.1. Data Mapping

**Data Mapping**

DataSeries :   Multi Selection

Category (X) :   Multi Selection

Sum by :   Multi Selection

Value (Y) :

2nd Value :

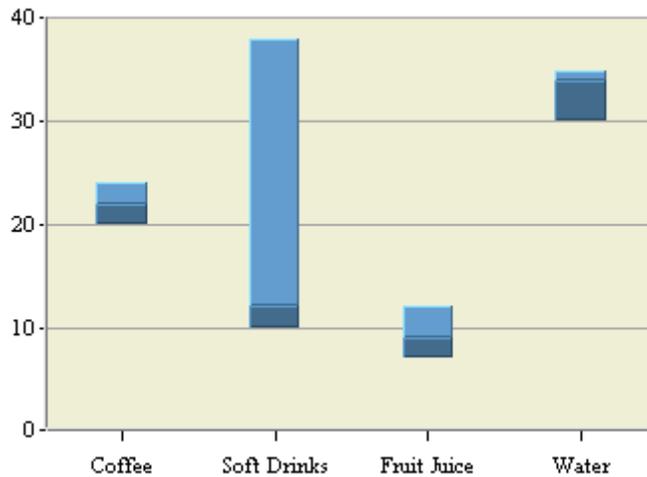
Combo:

2nd Series :   Multi Selection

Mapping Options for Stack Area Charts

Data Mapping for stack area charts is almost exactly the same as for stack column charts (covered in Section 4.2.3.6.1 - Data Mapping). However, two-dimensional stack charts cannot have a data series, and the combo options are *Line* and *Overlay*.

### 4.2.3.11. High-Low Charts



*High-Low Chart*

A high-low chart is also a derivative of a column chart, only that instead of one value column it uses two columns for high and low bounds of values. The data for the chart must contain at least three columns: category, high, and low. The category column can contain values of any type, while the high and low columns must be numeric.

Another option for a high-low chart is a data column called the *close* column. If a close column is also used, then the close value will lie somewhere between the high and low values. The portion of the bar between the low point and the close point is rendered in a darker form of the same color as the portion between the close point and the high point. As for many other types of charts, a high-low chart may include a data series column.

#### 4.2.3.11.1. Data Mapping

**Data Mapping**

DataSeries :   Multi Selection

Category (X) :   Multi Selection

High :

Low :

Close :

2nd Value :

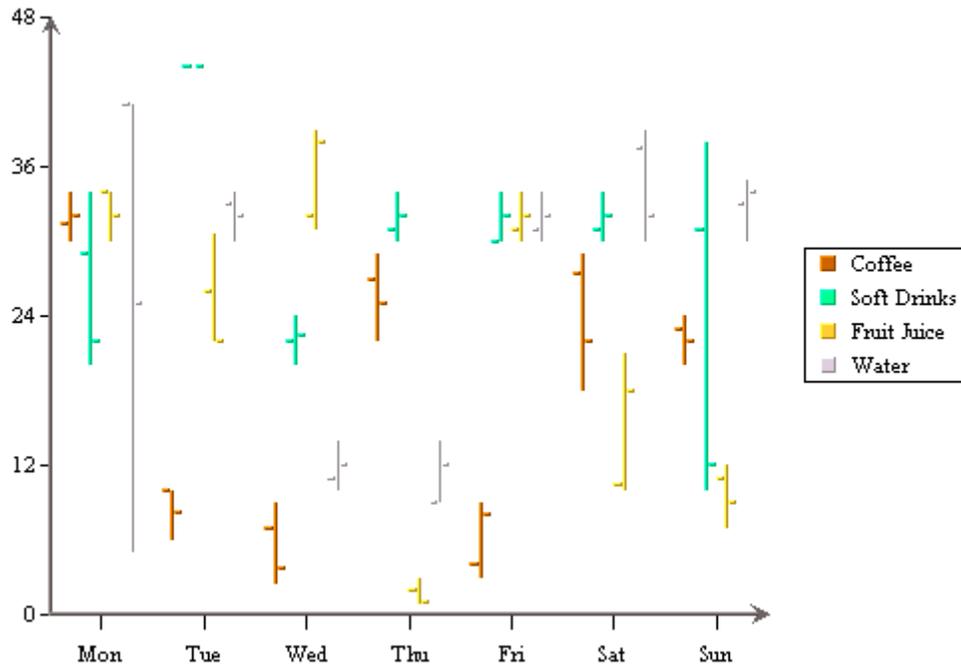
Combo:

2nd Series :   Multi Selection

*Mapping Options for High-Low Charts*

Data mapping for high-low charts is similar to column charts. You can select series and category columns in the same manner. The difference for high-low charts is that you need to specify at least two value columns - a high and a low value. These are the two points that are plotted for each category. A third value called *Close* can also be specified. Combo options for high-low charts are *Line*, *Column*, and *Overlay*.

### 4.2.3.12. HLCO Charts



HLCO Chart

A HLCO (High Low Close Open) chart is similar to the high-low chart described above, except that it also contains an *open* column. It is useful in presenting data that fluctuates over discrete periods of time, such as stock prices or inter-day temperatures. The HLCO chart can also be displayed in a Candlestick format for both 2D and 3D charts. This feature is described in Section 4.2.4 - The Chart Designer Interface.

#### 4.2.3.12.1. Data Mapping

The Data Mapping interface includes the following settings:

- DataSeries : Drink
- Category (X) : Day
- High : high
- Low : low
- Open : open
- Close : close
- 2nd Value : None
- Combo: Line
- 2nd Series : Day

Mapping Options for HLCO Charts

Data mapping for HLCO charts is similar to that for High-Low charts. The only difference is that instead of specifying two different value columns, you must specify four different columns - *High*, *Low*, *Open* and *Close*. Combo options are *Line*, *Column* and *Overlay*.

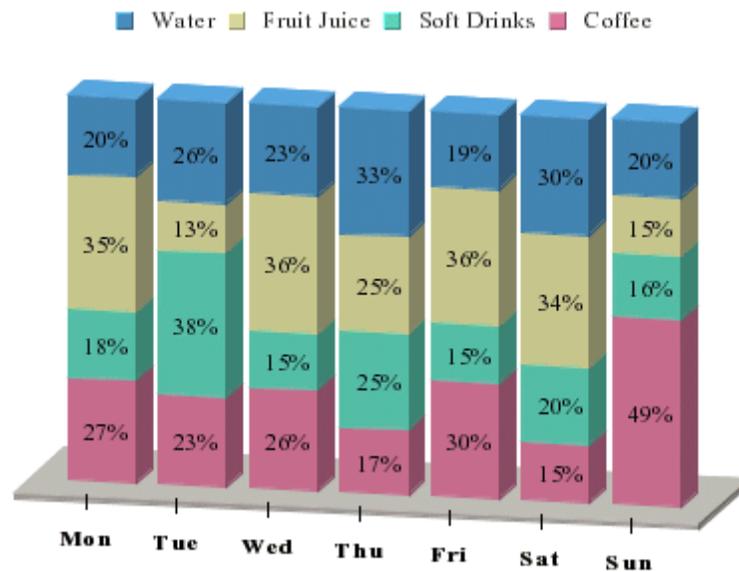
A common kind of chart is a one that plots both stock price and trading volume in the same chart. This can be accomplished using an HLCO-column combination in ChartDesigner.



HLCO-Column Combination Chart

This example plots stock price and volume data over a three month period.

### 4.2.3.13. Percentage Column Charts



Percentage Column Chart

A percentage column chart may be viewed as a derivative of pie and column charts together. Each column in the chart corresponds to one pie. A percentage column chart has a category column corresponding to the X-axis value. A sum-by column represents the category in this case and the value column is the same as that of a pie chart.

### 4.2.3.13.1. Data Mapping

**Data Mapping**

DataSeries :   Multi Selection

Category (X) :   Multi Selection

Sum by :   Multi Selection

Value (Y) :

2nd Value :

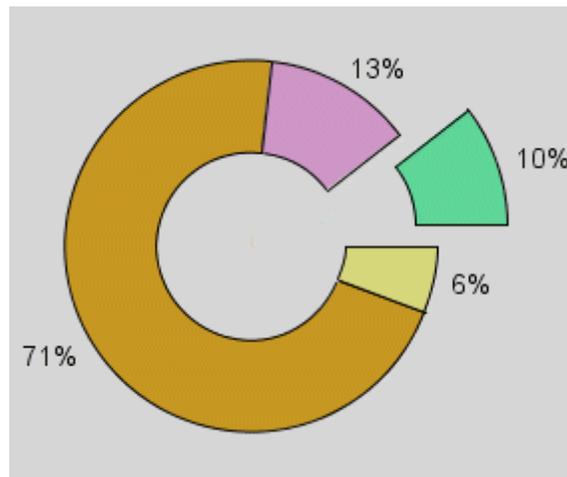
Combo:

2nd Series :   Multi Selection

*Mapping Options for Percentage Column Charts*

Data mapping for percentage column charts is the same as for stack column charts (covered in Section 4.2.3.6.1 - Data Mapping). The only difference is that the sum-by components are plotted as a percentage of the Value column instead of discrete values. Combo options for percentage column charts are *Line* and *Overlay*.

### 4.2.3.14. Doughnut Charts



*Doughnut Chart*

A doughnut chart is the same as a pie chart, except for the “hole” in the center.

Just as for a pie chart, if a data series is selected, a separate doughnut will be drawn for each category element and each doughnut will be comprised of the series elements. When a series is present, all the separate doughnuts are drawn on a single plot area and resized in proportion with the plot. You have an option to either stack the different doughnuts or draw them in a line.

#### 4.2.3.14.1. Data Mapping

**Data Mapping**

DataSeries :   Multi Selection

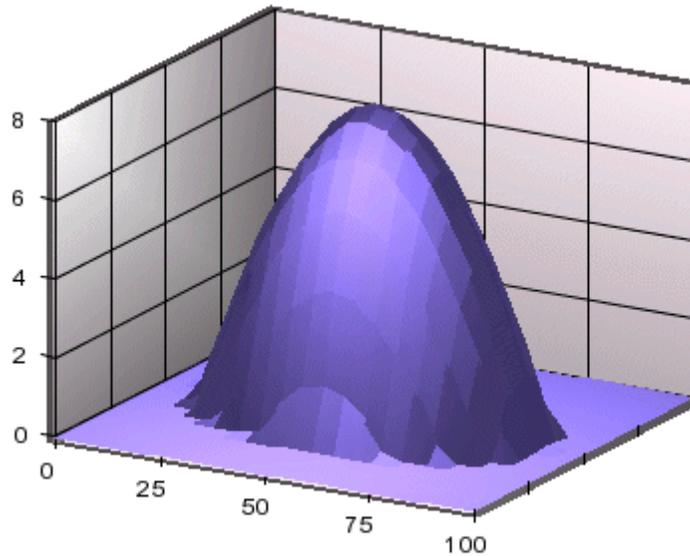
Category (X) :   Multi Selection

Value (Y) :

*Mapping Options for Doughnut Charts*

Data mapping for doughnut charts is exactly the same as for pie charts (discussed in section Section 4.2.3.8.1 - Data Mapping).

### 4.2.3.15. Surface Charts



Surface Chart

A 3D surface chart is a scientific/business chart, which uses three sets of numeric data values to form a smooth surface in a three dimensional space. For each data point, two of the three values represent the coordinates on the horizontal plane and the third value is used to determine the vertical position of a data point. Input data can be in a form of a rectangular matrix (as shown below) or arranged in three columns where each column represents an axis. In the following sample data, the top row (column header) and the left column (row header) represent the coordinate values of the axes that form the plane on which the surface will be drawn. When the chart is viewed from above the plane, you can see a grid formed by joining every four adjacent points together. A given set of data cannot plot a surface chart if this grid cannot be drawn.

Sample surface chart data:

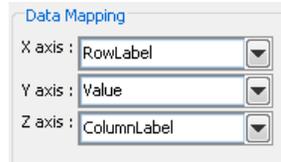
	0	20	40	60	70	80	100
20	0	0	0	0	0	0	0
30	0	10	10	10	10	10	10
40	0	10	25	25	25	2	10
80	0	10	25	30	30	30	25
90	0	10	25	30	30	30	25
100	0	10	10	25	25	25	10

Note: Sample data for creating a surface chart can be found in `surface.dat` file, which is located in `<ERES install dir>/help/examples/DataSources/text/` directory.

The values of the column header and the row header are not required to be equally separated because a surface chart can properly scale the horizontal distances. This dataset can create a distorted inverted cone. A surface chart can be very impressive when animation is turned on.

A surface chart uses a set of X, Y, and Z coordinates to plot a 3D chart. The vertical axis is called Y-axis and each Y value is represented by a function of X and Z,  $f(X, Z)$ . Data on XY plane (horizontal plane) simply looks like a square matrix. Surface charts do not support data series and cannot be converted to a 2D chart.

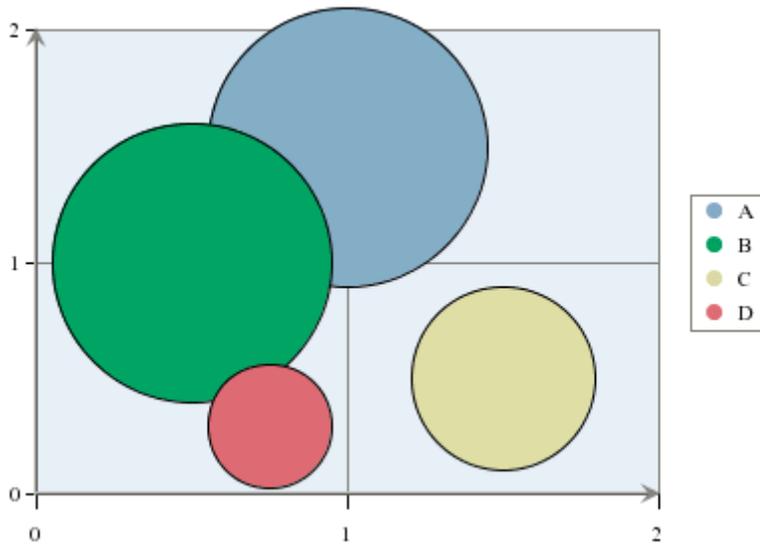
### 4.2.3.15.1. Data Mapping



*Mapping Options for Surface Charts*

The data mapping for a surface chart is similar to a three-dimensional scatter chart; the X-axis, Y-axis, and Z-axis values determine the X, Y, and Z coordinates of a point respectively. However, unlike a scatter chart, surface charts do not support data series and cannot be converted to a two-dimensional chart.

### 4.2.3.16. Bubble Charts



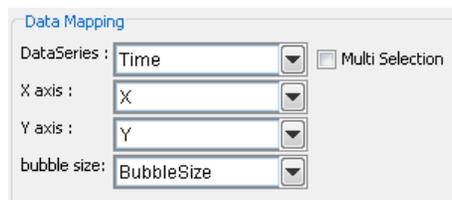
*Bubble Chart*

A bubble chart is used to represent data wherein the size of the bubble also provides information. A data point is represented by an XY coordinate and a third point, which is the radius of a circle or bubble.

This chart is available in a two-dimensional form only. For more information about the bubble charts and their display options, please see Section 4.2.4.9.1 - Bubble Charts.

#### 4.2.3.16.1. Data Mapping

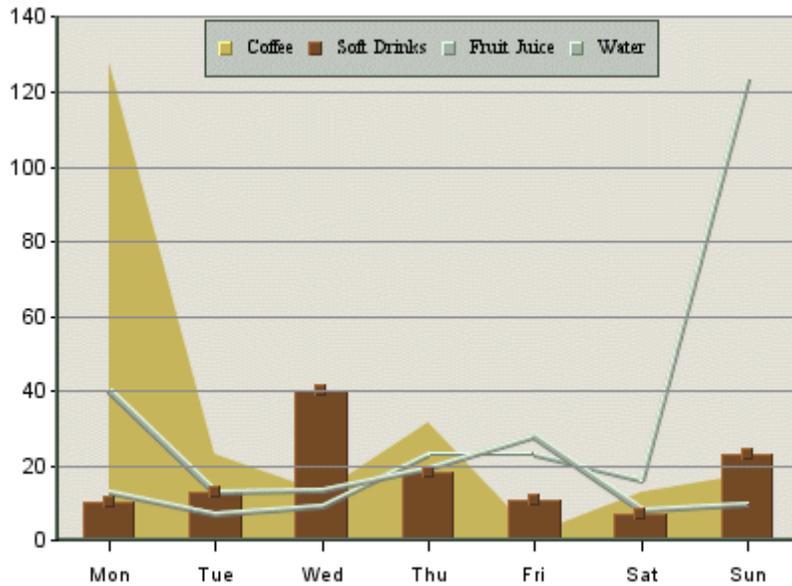
The data mapping for bubble charts is similar to three-dimensional scatter charts; however, instead of plotting a Z-axis position, the third value determines the size of the bubble (specifically the radius).



*Mapping Options for Bubble Charts*

Note that similarly to a scatter chart, the columns have to be numeric in order to be mapped successfully.

### 4.2.3.17. Overlay Charts



*Overlay Chart*

ChartDesigner supports a special chart type called overlay chart, which allows you to super-impose more than two charts with a common category axis. A different chart can be used to represent each element of a data series. This allows for more freedom while creating the chart and also allows more information to be represented. The chart types supported in an overlay chart are column, area, and line charts. Only two-dimensional charts can be included in an overlay chart.

#### 4.2.3.17.1. Data Mapping

**Data Mapping**

DataSeries :   Multi Selection

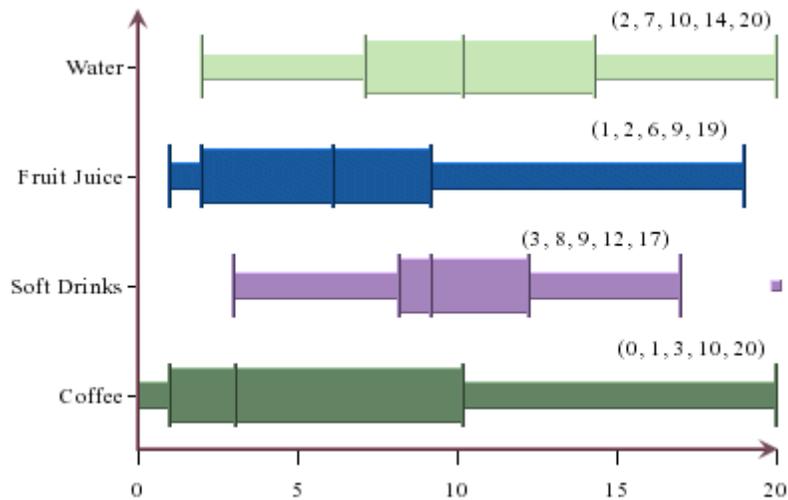
Category (X) :   Multi Selection

Value (Y) :

*Mapping Options for Overlay Charts*

The data mapping for an overlay chart is similar to a column chart (covered in Section 4.2.3.2.1 - Data Mapping), except that the overlay chart plots each element in the data series as a separate chart. Please note that the *2nd Value*, *2nd Series* and *Combo* options do not apply to overlay charts. Overlay charts do not support secondary values. However, the different series elements can be plotted on different axes. For more about plotting options for overlay charts, see Section 4.2.4.9.3 - Overlay Charts.

### 4.2.3.18. Box Charts

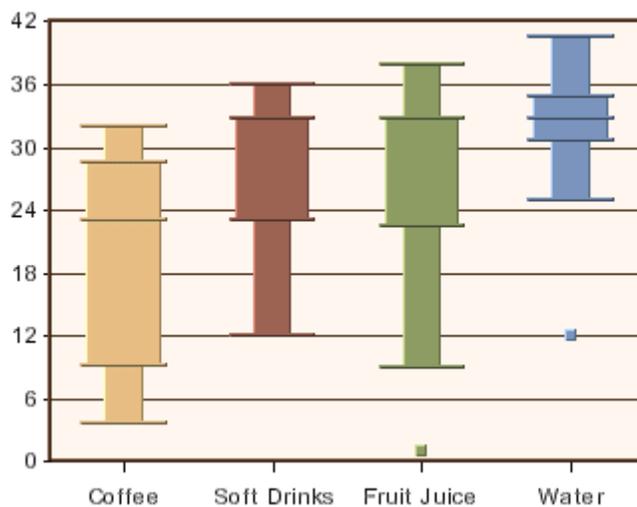


Box Chart

ChartDesigner provides a statistical chart type, the Box Chart, also known as Box and Whiskers Chart. Basically, the box chart provides a quick visual realization of summary statistics. A histogram shows the distributions of observed values so you can identify the peak(s), minimum, maximum values and clustering of data. A box chart, on the other hand, displays a summary of data distribution in quarterly percentiles (Minimum data point, 25th percentile, median, 75th percentile, and Maximum data point).

The middle line in the box represents the median. The other lines in each direction represent the 25th percentile increment (decrement). The minimum and maximum points are the observed minimum and maximum which are not outliers. The lines that extend from the edge of the box to the minimum/maximum are sometimes called whiskers. Hence, the name box and whisker chart. Outliers are any values that are 1.5 times (or more) the length of the box, that being the difference between the 75th and 25th percentiles. Outliers are calculated and shown as dots away from the box. The other nice feature about box plot is that you can stack up the boxes in one chart to compare summaries of different data sets.

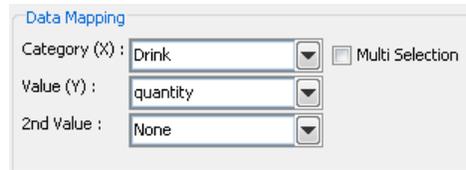
ChartDesigner allows box charts to be displayed vertically in addition to the default horizontal setting. To use this feature in Chart Designer, create a box chart and then select Format → Chart Options, and select *Vertical*.



Vertical Box Chart

This chart is available in two-dimensional form only.

### 4.2.3.18.1. Data Mapping



*Mapping Options for Box Charts*

For box charts the mapping is as follows:

**Category (X):** Allows you to choose a data column whose distinct values determine the various categories.

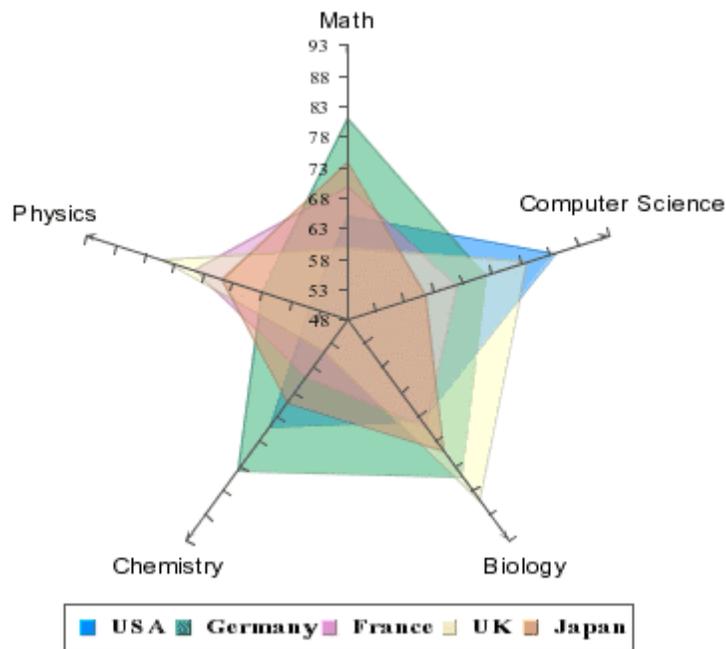
**Value (Y):** Allows you to choose a data column to provide values for each category. These values are used to calculate the minimum data point, the 25th percentile, the median, the 75th percentile, and the maximum data point.

**2nd value:** Add a second value to create a combination chart.

Please note that the *2nd Series* and *Combo* options are not available for box charts. This is because the only combination available with box charts is a line.

The data mapping also allows you to transpose the data (in other words: to select several columns for a single category). To learn more about data transposition, please see Section 4.2.3.1.1 - Data Transposition.

### 4.2.3.19. Radar Charts

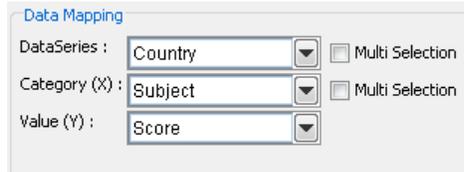


*Radar Chart*

Radar charts are useful when you need to compare performance/measurement results, statistics, etc from different sources. For example, we can compare median test scores of children in the industrial nations on subjects such as math, computer science, and biology. A radar chart has multiple axes along which data can be plotted. Each axis is a category. The data is shown as points on the axis. The points belonging to one data series can be either joined or the enclosed area filled. A point close to the center on any axis indicates a low value while a point near the end represents a high value. In some cases, low values may be more desirable than high values, e.g. price/earning, price/sales, price/book of stocks.

This chart is available in a two-dimensional form only.

### 4.2.3.19.1. Data Mapping



*Mapping Options for Radar Charts*

For radar charts the mapping is as follows:

**Data Series:** Allows you to choose a data column whose distinct values will determine the number of data series in the chart. Each element in a data series is drawn using the same color along each axis.

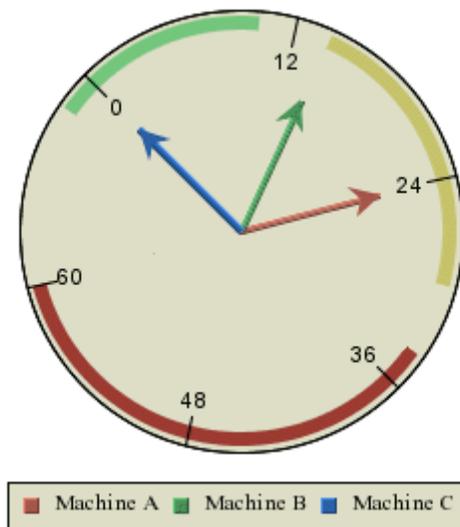
**Category (X):** Allows you to choose a data column whose distinct values determine the categories. The number of unique elements in this column determines the number of axes in the radar chart.

**Value (Y):** Allows you to choose a data column to provide the numeric value to plot against the category.

Please note that the *2nd Value*, *2nd Series*, and *Combo* options do not apply for radar charts. Radar charts do not support secondary values.

The data mapping also allows you to transpose the data (in other words: to select several columns for a single category). To learn more about data transposition, please see Section 4.2.3.1.1 - Data Transposition.

### 4.2.3.20. Dial Charts



*Dial Chart*

Dial Charts are used to build circular charts (such as temperature gauges, speedometers, and clocks), create dashboards, or balanced scorecard applications. Here the data is represented as a “hand” on a “dial”. You can either use the whole dial (for instance for a clock) or just part of it (a semi-circular chart such as a gasoline gauge). Dial Charts support pop-up labels, drill-down, and user interactions such as rotating the hands or sectors of a dial.

This chart type is available in a two-dimensional form only.

### 4.2.3.20.1. Data Mapping

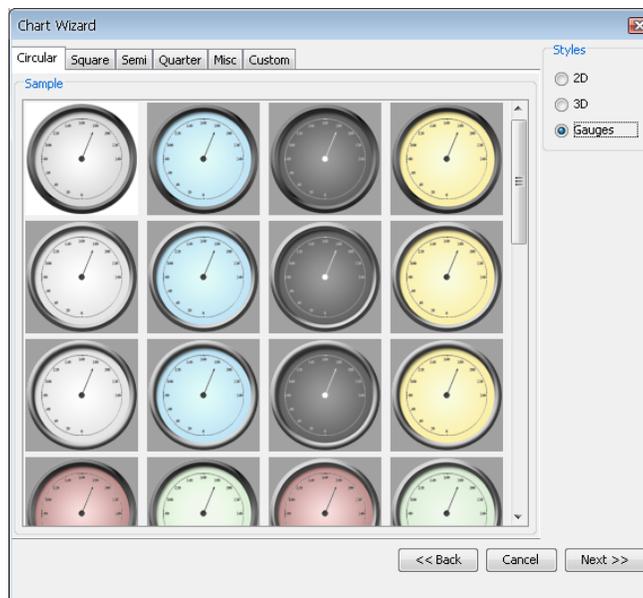


*Mapping Options for Dial Charts*

The data mapping of a dial chart is similar to a pie chart (detailed in Section 4.2.3.8.1 - Data Mapping). However, instead of pie wedges, the categories become dial hands. Also, dial charts do not support data series or secondary values.

### 4.2.3.20.2. Gauges

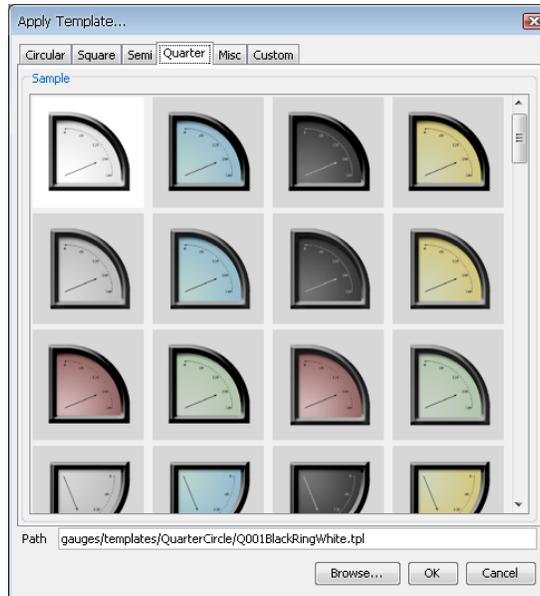
Gauges are special types of dial charts containing a plot background image and/or plot foreground image. To create a new gauge, select the *Gauges* radio button on the *Chart Type Selection* dialog.



*Selecting a Gauge Template*

You can also create your own gauge by creating a template, saving the tpl file in the <ERES Install>/gauges/templates/Custom/ folder (please note that the Custom folder is not created automatically by the ERES installer so you will have to create it manually in your favorite file manager). To add this template to the Custom tab, you will also need to provide two images, a screenshot of the template placed in <ERES Install>/gauges/screenshots/selected/Custom/, and a dimmer version of the same template placed in <ERES Install>/gauges/screenshots/unselected/Custom/. An easy way to create the screenshot is to export the template to gif for the regular screenshot and resize it to 100px by 100px. Then change the background to a darker color, export and resize again for the dimmer version.

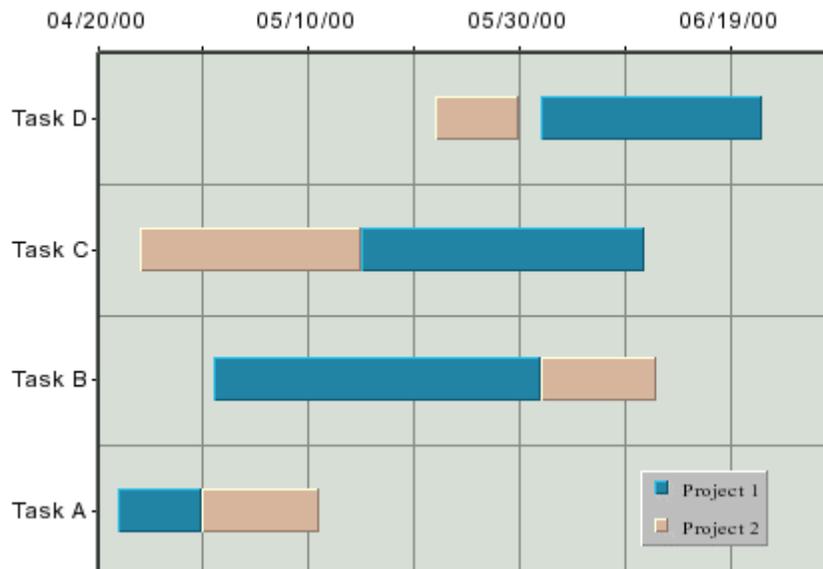
It is also possible to apply a gauge template onto an existing dial chart. Selecting apply template when the current chart is a dial chart will display the gauge tabs just like when creating a new chart.



*Applying a Gauge Template*

For more information regarding the dial background and foreground images, see Section 4.2.4.9.2 - Dial Charts.

### 4.2.3.21. Gantt Charts



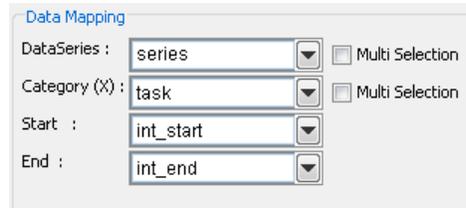
*Gantt Chart*

A Gantt or time chart is used to represent data where a time factor is being measured. This is often used in project or time management situations. A Gantt chart resembles a bar chart with the category axis on the Y-axis and the value axis on the X-axis. The value axis uses date, time, or timestamp data (numeric values can also be used). Each data point has a start and end time associated with it.

This chart type is available in a two-dimensional form only.

#### 4.2.3.21.1. Data Mapping

The data mapping for Gantt charts is similar to high-low charts. However, instead of selecting a column for the high and low values, you select the start and end values.



*Mapping Options for Gantt Charts*

The mapping is as follows:

**Data Series:** Allows you to choose a data column whose distinct values will determine the number of data series in the chart.

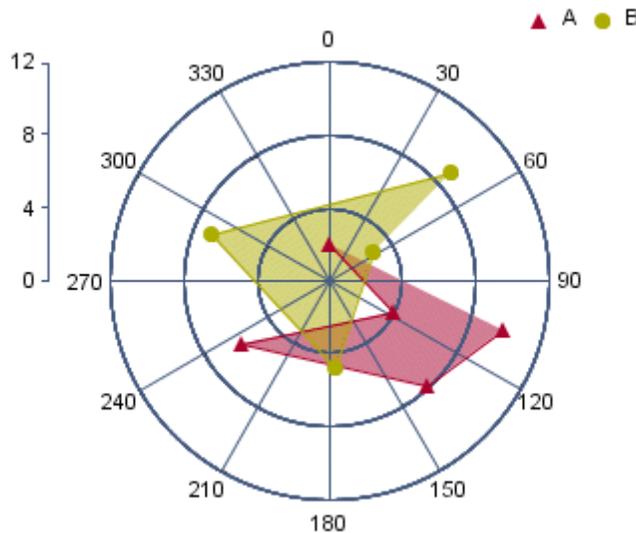
**Category (X):** Allows you to choose a data column whose distinct values will determine the categories.

**Start:** Allows you to choose a data column whose distinct values represent the starting time interval for the category.

**End:** Allows you to choose a data column whose distinct values represent the ending time interval for the category.

The data mapping also allows you to transpose the data (in other words: to select several columns for a single category). To learn more about data transposition, please see Section 4.2.3.1.1 - Data Transposition.

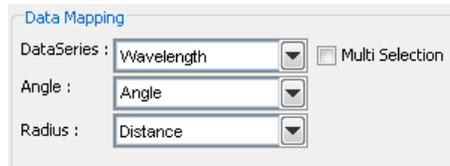
### 4.2.3.22. Polar Charts



*Polar Chart*

Like a two-dimensional scatter chart, a polar chart plots points on a plane. However, instead of rectangular coordinates, a polar chart plots points using polar coordinates  $(r, \theta)$ , where  $r$  represents the distance from the center of the circular plot and  $\theta$  represents the angle of a ray emanating from the center of the plot and passing through the point. Like scatter charts, the data for polar chart coordinates must be numeric. The  $\theta$  values can be supplied in degrees or radians. A third data series column can be used to separate the data points into groups.

### 4.2.3.22.1. Data Mapping



*Mapping Options for Polar Charts*

The data mapping for polar charts is similar to scatter charts. The angle and radius options allow you to select the columns whose values will make up the polar coordinates. These must both be numeric. The data series box allows you to choose a data column whose distinct values will determine the number of data series in the chart. Each element in a data series is drawn using the same set of drawing attributes, e.g., colors and markers.

### 4.2.3.23. Changing Data Mapping or Data Source

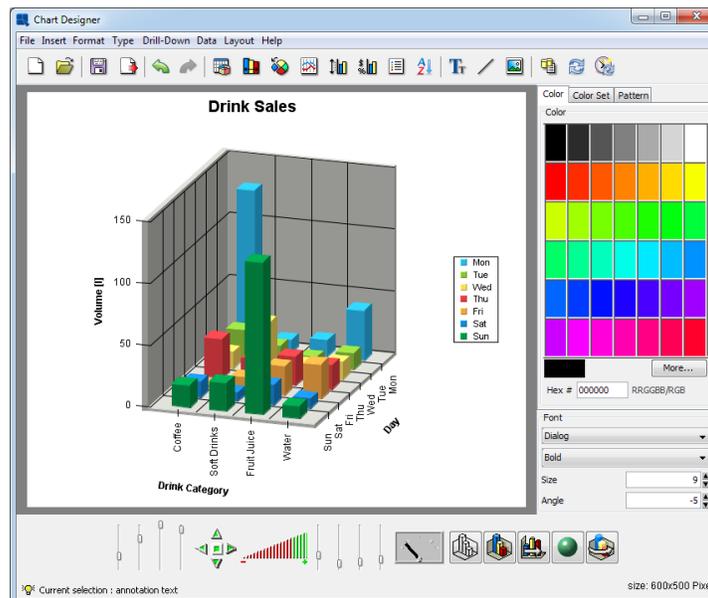
Once you have finished setting the mapping options you will be taken to the main Chart Designer interface. Once you are in the Chart Designer you can return to the Data Mapping window by selecting Data → Modify Data

Mapping, or by clicking the  *Change data mapping* icon. This will return you to the data mapping screen, allowing you to adjust the mapping for the chart.

You can also change a data source directly from the Chart Designer by selecting Data → Modify Data Source, or by clicking the  *Modify data source* icon on the toolbar. This will open the Data Source Manager.

## 4.2.4. The Chart Designer Interface

Once you have completed all the steps in the Wizard, the main Chart Designer interface will appear. Here you can customize and modify the appearance of the chart by changing properties and adding new elements.



*Chart Designer Interface*

### 4.2.4.1. The Designer Menus

Most of Chart Designer's functions can be controlled from the menu bar at the top of the designer window. This section provides a brief overview of the available options. All of the features here are discussed later in this chapter.

#### 4.2.4.1.1. File Menu

This menu performs basic file operations such as opening, closing, and saving files. Note that saving and opening options are not available when you use the report data as the data source for an embedded chart. For more information about file options, please see Section 4.2.6 - Saving & Exporting Charts.

- New:** This will return you to the Data Source Manager to begin creating a new chart. If you have not saved your current chart, you will be prompted to do so.
- Open:** This allows you to open a saved chart definition. Files that can be loaded by Chart Designer include .cht, .tpl, .pac, and .xml chart definition files.
- Close:** This closes the current chart.
- Apply Template:** This option allows you to apply a template to the current chart. You can apply any chart template in either .tpl or .xml format. For more information about working with chart templates, please see Section 4.2.6.2.1 - Working with Templates.
- Save:** This saves the current chart.
- Save As:** This allows you to save the current chart. You can save the chart as a .cht, .pac or .tpl (a chart, a complete packed chart or a template file). The
- Create XML
- check-box allows you to create an XML chart definition file.
- Export:** This allows you to export the chart to a number of static image formats. The chart data can also be exported as XML file.
- Exit:** This closes the Chart Designer with the possibility to save the current file.

#### 4.2.4.1.2. Insert Menu:

This menu allows you to add various elements to a chart.

- Titles:** This option allows you to automatically add titles to the chart. A main title can be specified as well as titles for each of the axes. Unlike annotation text, titles will size and position automatically with the chart.
- Text:** This allows you to add text or annotation to a chart. Text can be added anywhere on the chart and can have a number of different formatting properties. Variables can be added to the text for run-time substitution. For more on adding text to charts, see Section 4.2.4.6.1 - Adding Text
- Background:** This allows you to select an image to use as the chart background. Background images can be tiled, centered, or stretched to fit the entire canvas.
- Dial Foreground:** This option is only available for dial charts. This allows you to select an image to use as the dial chart foreground. The image can be stretched.
- Dial Background:** This option is only available for dial charts. This allows you to select an image to use as the dial chart background. The image can be stretched.
- Link:** This allows you to add a hyperlink to a data point or a collection of data points in a chart. To use hyperlinks in a chart, you will need to export a map file along with the image.
- Line:** This allows you to add an arbitrary floating line to a chart. These lines can be placed anywhere and can be used to draw enclosed shapes as well. Floating lines are often used as pointers and can be generated with arrowheads.
- TrendLine:** This allows you to add a trend line to the chart. ChartDesigner allows you to draw many different types of trend lines including: linear, a polynomial of any degree, a power, exponential, logarithmic, a moving average, exponential moving average, triangular moving average, cubic B-spline, and a normal distribution curve.

**Horz/Vert Line:** This allows you to add a fixed horizontal or vertical line to a chart. You have the choice of either adding a constant line (one that draws at a fixed value on the X or Y-axis) or a control line (draws lines based on a certain value range either average, minimum, maximum, or multiples of standard deviation).

**Control Area:** This allows you to draw a fixed area (either on the plot area of a 2D chart or on the face of a dial chart) to compare against the data values of the chart.

#### 4.2.4.1.3. Format Menu

This menu allows you to edit and modify the properties of many different chart components.

**Undo:** Cancels the last operation performed in the designer and reverts back to the previous state. The designer will remember the last 10 actions made.

**Redo:** Reverses the action of the undo command. For example, if you change the font color from black to red, you can undo this command to change it back to black, and then redo this command to have it change to red again.

**Data Properties:** This option allows you to control several options for how the data is displayed. You can control the thickness of columns/bars, how null data is displayed, whether to draw data top labels or not, as well as enable and select a color for negative top labels.

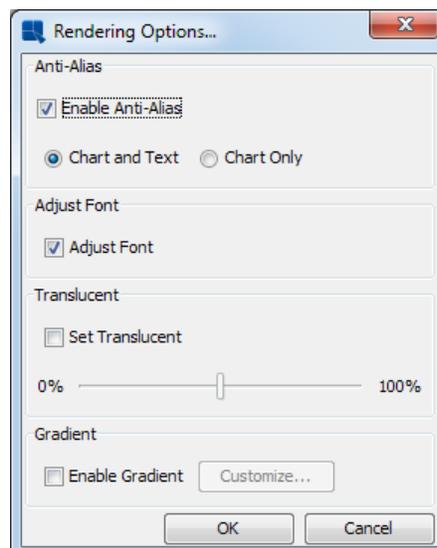
**Histogram Options:** This allows you to specify if you want to draw the chart as a histogram and display additional options to specify the frequency count.

**Aggregation Options:** This allows you to specify whether to aggregate the data before drawing the chart and display additional options to specify type of aggregation.

**Zoom Options** This allows you to enable/disable and set options for time based zooming. This option only applies if you have date/time data mapped to the category axis.

**3D Display Options:** This allows you to set several options that control the display of 3D charts. You can specify an inline series for 3D column (or similar) charts, as well as specify rendering approximation for 3D scatter and surface charts. (This improves performance for charts with a lot of data points.)

**Rendering Options:** This allows you to specify various rendering options for the chart for a better presentation.



*Rendering Options Dialog*

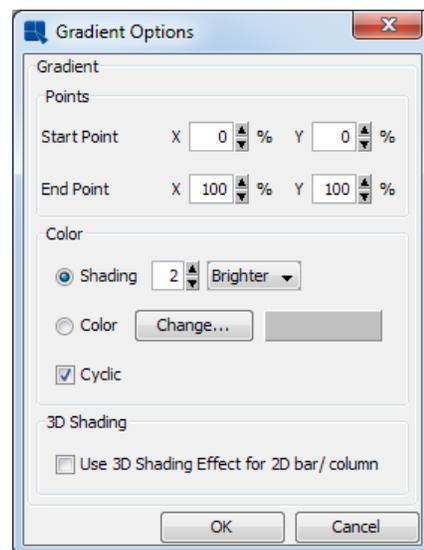
Among the options available are:

**Enable Anti-Alias:** This allows you to specify anti-aliasing for the chart. Anti-aliasing will smooth text and lines in the chart, creating a smoother appearance. This feature can be applied to either the entire file or just to the chart, which will leave the text unaffected.

**Adjust Font:** This allows you to specify the font size for text in charts to be relative to the screen resolution. This feature allows for more precise conversions of charts to various export formats and between installations.

**Translucent** This allows you to specify whether the columns/bars/areas should be translucent. This allows any columns/bars/areas “hidden” behind to show through. You can also specify the opacity by adjusting the slider, where 0% is completely opaque and 100% is completely transparent. This option is available for all 3D charts and for 2D Area, Radar and Gantt charts with data series.

**Gradient:** This allows you to enable gradients. The gradient can be applied either across the entire canvas or to chart data points only. You can change the gradient options by clicking on Customize to show the following dialog:



*Gradient Options Dialog*

This dialog allows you to specify the start and end points for the gradient (based on the x-y plane), the gradient color scheme, and whether the gradient should be cyclic. The start and end points are represented as percentages with (0,0) being the top left corner of the canvas and (100, 100) being the bottom right corner of the canvas. You can also specify whether the gra-

dient change is shading (becoming darker or brighter) or a different color. Lastly, you can specify the gradient to be cyclic i.e., toggle between the two opposites in the gradient. Note that the start and end points represent the first segment if the gradient is cyclic.

You can also enable 3D shading for certain 2D charts (Column, Bar, Stack Column, Stack Bar, 100% Column and Overlay) by selecting the *Use 3D Shading Effect for 2D bar/column* option (please note that this option is available for 2D bar/column charts only).

**Font Mapping:**

This allows you to map system (true type) font files for the PDF export. For more information about this feature, please see Section 4.2.6.3.1 - PDF Font Mapping

**Chart Options:**

This brings up some specific options for the chart type that you are using. Options vary depending on chart type.

**Axis Scale:**

This allows you to adjust the scale for any value axes. Automatic (best fit) scaling is used by default.

**Axis Elements:**

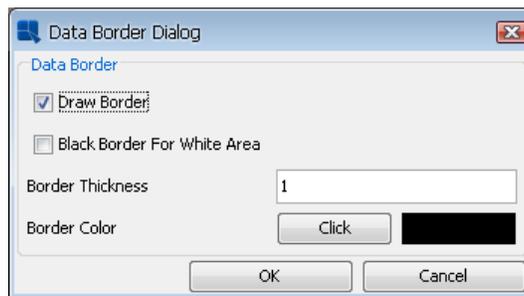
This allows you to modify the appearance of the axes and axis labels. Options here include axis thickness, grid lines, label steps, and data formatting.

**Canvas:**

This allows you to adjust the background canvas size. You can also specify whether to use scroll bars when the canvas size exceeds the view port.

**Data Border:**

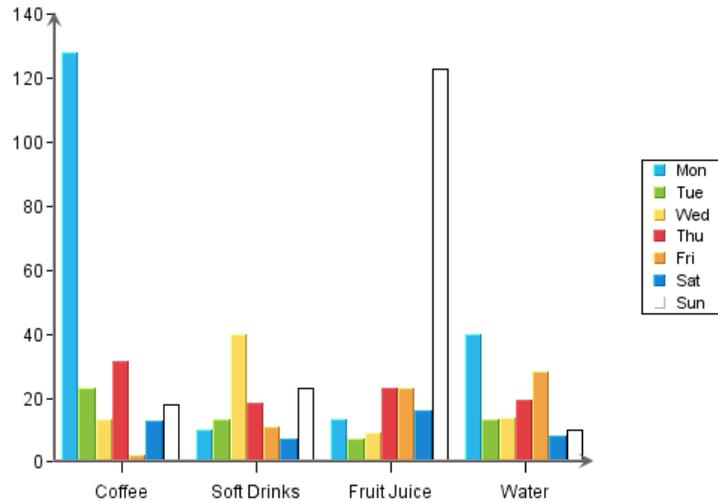
Add a border to data elements (columns, bars, etc).



*Data Border Dialog*

If you select the *Draw border* option, the *Border thickness*, and the *Border color* fields will be enabled allowing you to configure the border properties.

The *Black border for white area* option can be selected even when the *Draw border* is disabled. If you do so, a simple black border will be added only to white data elements.



Column chart with the “Black border for white area” enabled



**Note**

This option is not available for Surface, Scatter, Line, Bubble, Box, Radar, Dial, and Polar charts.

- Legend:** This allows you to modify the display properties of the chart legend.
- Lighting Model:** This allows you to modify some of the lighting options for three-dimensional charts. You can modify both the light ambient color and the intensity.
- Line and Point:** This allows you to modify the display properties of any lines in a chart. For two-dimensional charts you can choose to display lines and points for any dataset as well as customize their appearance. You can also use this menu item to modify the display properties for any trend, floating, or horizontal/vertical lines.
- Plot Area:** This option allows you to customize appearance of the area bounded by the X and Y axes for two-dimensional charts.
- No Data To Plot Message:** This option allows you to set a message that appears if there is insufficient data to plot the chart. By default, the “No Data To Plot” message appears.
- Flash Hintbox Customization:** This option allows you to specify the font properties as well as the border and background color for the hint box in the flash export.
- NULL Data Properties:** This option allows you to show any category axis point that contains Null data and replace it with a different string. By default, any Null data category point is skipped.
- Table:** This allows you to add and configure a table displaying the chart data.
- Text Properties:** This option allows you to set a resize ratio for any text in the chart. From this option you can also specify to use Java 2D rotate text. This option gives rotated text a cleaner appearance. You can also specify any text replacement options.
- Viewer Options:** This menu item allows you to specify some configuration options for the Chart Viewer JNLP (More about Applets in JNLP: Section 1.3.4 - Run Ap-

plets As JNLP) when the chart is viewed. You can control what options are available in the Viewer pop-up menu. These options can also be controlled through HTML parameters.

#### 4.2.4.1.4. Type Menu

This menu allows you to change the current chart and its dimension. You can switch between 2D and 3D for chart types that support representations in each dimension. You can also change chart types. Note that as you switch between chart types, some formatting information will be lost. Also, you cannot switch between Gantt charts and other chart types.

#### 4.2.4.1.5. Drill-Down Menu

This menu contains options allowing you to add and navigate drill-down layers in a chart. The drill-down features are explained in Section 4.2.5 - Chart Drill-Down. Note that drill-down is only available (functional) when charts are deployed independently from reports.

<b>Add:</b>	This allows you to add a layer of data drill-down.
<b>Remove This:</b>	This removes the current level of data drill-down.
<b>Remove All:</b>	This removes all levels of data drill-down.
<b>Previous</b>	This navigates to the previous layer of data drill-down.
<b>Next:</b>	This navigates to the next layer of data drill-down.
<b>Go To Top Level:</b>	This navigates to the top-level chart for data drill-down.
<b>Dynamic:</b>	This allows you to enable dynamic data for drill-down.
<b>Parameter Drill-Down:</b>	This brings up the parameter drill-down navigation window allowing you to edit, add, and remove layers of parameter drill-down.

#### 4.2.4.1.6. Data Menu

This menu contains options that allows you to refresh, re-order or completely change the chart data.

<b>Modify Data Mapping:</b>	This will bring back the Data Mapping Window, allowing you to change the data mapping for the chart.
<b>Modify Data Source:</b>	This will take you back to the Data Source Manager, allowing you to select a new data source for the chart.
<b>Modify Database:</b>	If the chart uses an independent data source that is from a database, this feature, like in Report Designer, will allow you to modify the database connection that the chart use. For more information about this feature, please see Section 3.1.3.5 - Editing Database Connections.
<b>Modify Query:</b>	If the chart uses an independent data source that is from a database, this feature, like in Report Designer, will allow you to modify the query that retrieves the chart data. For more information about this feature, please see Section 3.1.3.4 - Editing Queries.
<b>Query Parameters:</b>	This will allow you to re-initialize query parameters and change parameter values for the current chart. This option is only available if the chart uses a parameterized query as the data source.
<b>Ordering:</b>	This option allows you to re-order the data points in a chart. You can arbitrarily change the order of any of the category elements or you can sort the categories by value.
<b>Refresh:</b>	This will update the current chart with the latest data. The original data source must be available for this option to work.

<b>Schedule Refresh:</b>	This allows you to set a schedule to refresh the data. This option is for deploying charts in the Chart Viewer JNLP (More about Applets in JNLP: Section 1.3.4 - Run Applets As JNLP). You can set a periodic refresh interval so that the chart will update itself with the latest data.
<b>View Table:</b>	This option will bring up a window containing the data table from which the current chart is generated. The table will initially display only the first 20 records. Clicking on the <i>Show All Records</i> checkbox will display all of the records.
<b>View Chart Data:</b>	Rather than viewing the entire data table you can just view the data points plotted in the current chart.
<b>View Data Source Info:</b>	If the chart contains an independent data source, this option will bring up a dialog containing information about the data source that was used to create the chart. The data source type and location are displayed.
<b>Go Back:</b>	This will go back to the original chart if you have traversed a link.

#### 4.2.4.1.7. Help Menu

This menu allows you to view a version of the program and to open a documentation.

**About:** This shows you information about the version of the program.

**Contents:** This opens the EspressoReport ES User's Guide.

#### 4.2.4.1.8. Layout Menu

The options in this menu allow you to toggle various elements in the Chart Designer interface. From this menu you can turn on and off the font and color panel, the Chart Designer toolbar, and the navigation panel for 3D charts.

### 4.2.4.2. The Designer Toolbar

The toolbar at the top of the Chart Designer window offers easy access to the Chart Designer's most commonly used features and functions. The buttons perform the following functions:

-  Start a new chart
-  Open an existing chart
-  Save the current chart
-  Export the current chart
-  Undo the last change
-  Redo the last undone change
-  Change data mapping for the current chart
-  Modify data display properties
-  Modify chart-specific options
-  Modify line and point attributes

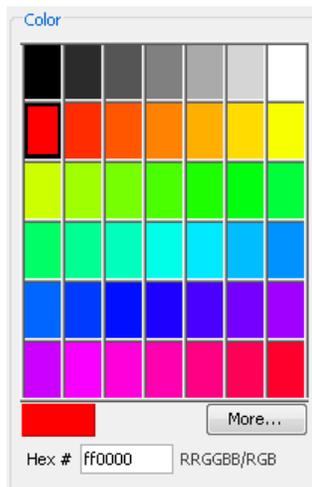
-  Change axis scale
-  Modify axis elements/display properties
-  Modify legend display
-  Change data ordering
-  Insert annotation text
-  Insert floating line
-  Add/change background image
-  Modify/change chart data source
-  Refresh chart data
-  Schedule periodic data refresh

### 4.2.4.3. Color, Color set, Pattern, and Font Panels

The color, color set, and font panels on the right-hand side of the Chart Designer window allow you to modify the color of any chart object, as well as modify the font size and style for any text/labels in the chart. You can choose not to display these panels by toggling the Layout → Show font/color panel option.

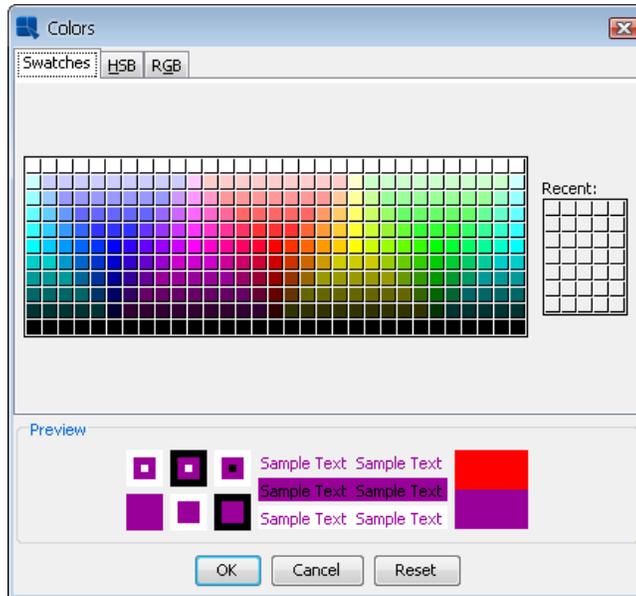
#### 4.2.4.3.1. Color Panel

You can use the color panel to change the color of any element in the chart. To modify an element's color, first, click on it. The status bar at the bottom of the Chart Designer window will indicate which element has been currently selected. After you have selected the object, click on one of the panels in the color panel and the color of the object will change to reflect the color you selected.



*Color Panel*

To create a custom color for the object, first select it, then click the *More* button. This will bring up a new dialog allowing you to pick a color from a larger palette or to create a new color.

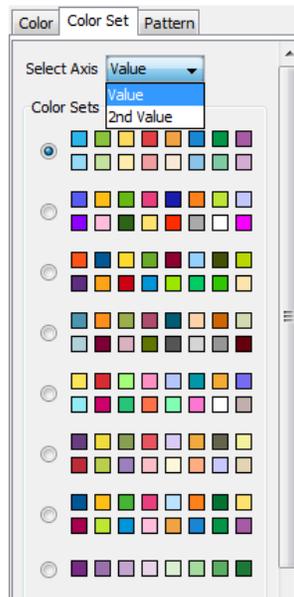


*Additional Colors Dialog*

From this dialog you can pick a new color from swatches or configure a custom color using HSB values or RGB values.

#### 4.2.4.3.2. Color Set Panel

The color set panel allows you to choose a color scheme for your chart. It contains predefined color sets that can be applied to chart data points on the value or the second value axis.



*Color Set Panel*

In order to change a color set for the value or the second value axis, first select the *Color Set* tab. Then select the axis (*Value* or *2nd Value*) from the *Select Axis* select box and click the appropriate color set radio button. After that chart data points will get colors from the selected color set. Note that the *Select Axis* select box is only visible when the chart has the second value axis. By default, the value axis uses the first color set while the second value axis uses the seventh.

Please note that if you change a color of a data point manually, it will automatically unselect the selected color set. This is because of the color set that no longer corresponds to colors of data points in the chart. It is also important

to note that if chart data points have more different colors than there are colors in the color set, it will automatically use colors from the beginning of the next color set, and so on. If there is no next color set available, it will use the first one instead.

#### 4.2.4.3.2.1. Save Colors for Categories feature

Data points colors are closely related to the *Save Colors for Categories* feature that is available in the *Data Properties* dialog. The dialog can be opened by clicking the  *Change data properties* icon on the toolbar, or by using *Format* → *Data Properties*. (Please note that this feature is not available for Stack charts and it is disabled if the chart has *Single Color for All Categories*.) If the feature is turned on, colors of chart data points are assigned to names of categories (or series for charts with series). If such categories (or series) then appear in the chart, it will automatically use their assigned colors. If the feature is turned off (default), colors are assigned by data points order (i.e. the first data point will get the first available color from the color set, the second will get the second color, etc.). *Save Colors for Categories* setting is automatically saved in the .pac file. The following example shows scenarios with the feature on and off:

Assume that a chart has three categories (“A”, “B”, and “C”) with colors taken from the following color set (blue, green, yellow, red).

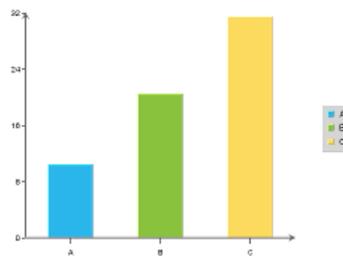


Image 1 - Example Chart

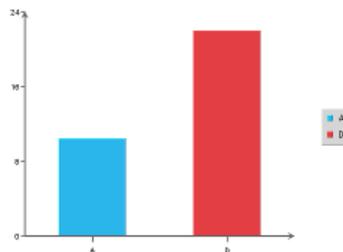


Image 2 - “Save Colors for Categories” Feature On

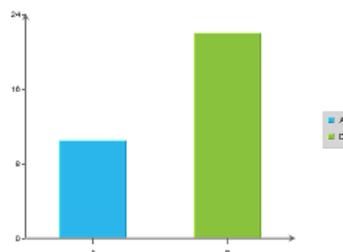


Image 3 - “Save Colors for Categories” Feature Off

#### Save Colors for Categories feature on

We saved the chart with the “Save Colors for Categories” feature enabled (see Image 1), so the following categories and colors were saved to the list of saved categories:

category A ... blue color

category B ... green color

category C ... yellow color

Now if you open the chart and data changes (e.g. there will be only categories A and D in the data - see Image 2), categories will have the following colors:

category A ... blue color (category A has blue color, because the category name A is present in the list of saved categories)

category D ... red color (category D has the next available color in the color set, because the category name D isn't present in the list of saved categories. In this case, the category will have red color, because blue, green, and yellow colors are already assigned to categories A, B, and C.

For this scenario, the color set will not be selected under the *Color Set* tab because colors of data points do not correspond to the color set.

**Save Colors for Categories feature off**

Here is the same situation, but assume that the chart has been saved with the "Save Colors for Categories" feature off (see Image 3). The list of saved categories will be empty this time.

After opening the chart, the categories will have the following colors:

category A ... blue color (category A has blue color, as it is the first available color from the color set)

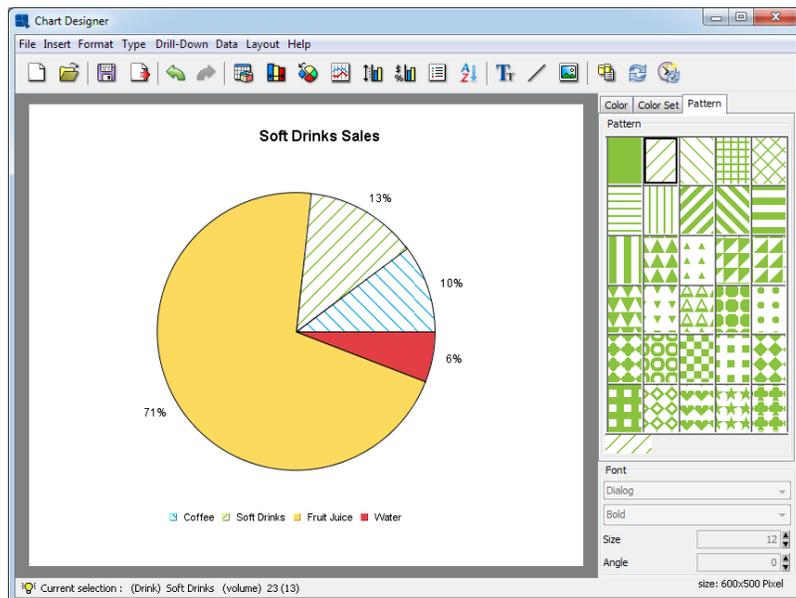
category D ... green color (category D has the second color from the color set, because the first is already assigned to the category A)

For this scenario, the color set will be selected under the *Color Set* tab because colors of data points correspond to the color set.

**4.2.4.3.3. Pattern Panel**

Unlike color and font panels, the pattern panel can only be applied to data points. There is a predefined pattern palette available for you to use. Similarly to how the color panel is used, you will need to first pick the data point you would like to change and then pick any pattern from the pattern palette. The pattern will be applied to the data point directly.

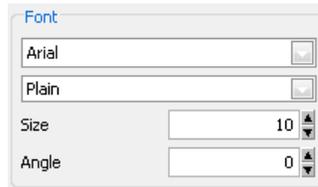
If a pattern has already been defined for a data point, the user can still change the color by selecting the color panel tab and picking a different color from the color palette. The color on the pattern will be changed to the new color immediately. The patterns shown on the pattern palette will change to the new color as well.



*Pattern Example*

#### 4.2.4.3.4. Font Panel

You can use the font panel to modify the font, the font style, font size, and the font angle for labels, titles, or other text in a chart. To modify the font, you must first select the text object whose font you would like to change by clicking on it. The status bar at the bottom of the Chart Designer window will indicate which is the currently selected object.



*Font Panel*

The first drop-down box allows you to select the font that you would like to use. The second drop-down box allows you to select the font style either plain, bold, italic, or bold and italic. The third box allows you to select the font size. The last box allows you to specify the angle of the text.

Certain groups of text (i.e. axis labels or data top labels) will all have the same properties. Hence, if you select one of text and modify the font properties it will apply to all of them.

ChartDesigner allows you to use the Java graphics libraries to give your text a cleaner appearance. For regular text you can use the chart anti-alias feature by selecting Format → Rendering Options. For rotated text (i.e. text not a 0 degrees), you can use the Java 2D rotate text feature by selecting Format → Text Properties. Note that these methods require Java 1.2 or higher.

#### 4.2.4.4. The Navigation Panel

The navigation panel provides options for controlling a number of the properties specific to three-dimensional charts. It does not appear for two-dimensional charts and it can be hidden for three-dimensional charts by toggling the Layout → Show Navigation Panel option.



*Navigation Panel*

There are six controls and five buttons in the Navigation panel. Starting from the left, the six controls are step size, light position for X, Y and Z axes, navigation, zoom, scale and navigation speed. The buttons on the right control wire frame/solid mode, on/off border drawing, on/off inline series (for columnar and bar charts with series), on/off Gouraud shading, and on/off animation.

**Step size:** This slide bar allows you to set the incremental amount used by the navigation control to rotate or move the 3D chart. The higher the position of the horizontal bar, the larger is the increment for each step.

**Light position for X, Y, and Z:** User-defined light position is useful to control the position of the light and therefore the direction from which it will shine on each axis.

**Navigation:** This control allows you to rotate or translate the chart. The center button is a toggle switch. When it is in a depressed state (denoted by the red color of the button), clicking on any of the four triangular buttons moves the chart in the direction indicated by that button. When the center button is in an elevated state, clicking on any of the four triangular buttons rotates the chart in the direction indicated by that button. The speed of navigation may be controlled by the Navigation Speed control.

**Zoom:** This control allows you to effectively move the chart closer to or farther away from the viewer. To operate, you point the mouse to the control and

click on the left mouse button. Then, while holding down the button, you drag the mouse to the left or right. When you drag the mouse to the right, the red area is extended to the right, and the chart will appear to be closer to you (zoom in to blow up the chart). When you drag the mouse to the left, the red area will move to the left, and the chart will appear to move farther away from you (zoom out to shrink the chart).

**Scale:**

This is a set of four slide bars. The first three bars allow you to change the scale factors for the X, Y, and Z axes respectively. The fourth bar determines the thickness of a three-dimensional column, bar, line, or pie (depending upon the chart type). The higher the position of the horizontal bar, the larger the value. For two-dimensional charts, you can adjust the bar width for bar, column, stack bar, stack column, high-low, and HLCO charts by selecting Format → Data Properties.

**Animation speed:**

This allows you to set the speed for chart navigation. This control is useful with the Navigation control and the Animation On/Off switch. It determines how fast the chart moves or rotates. To operate, click and drag the indicator needle to the desired position. Moving the indicator needle to the right will speed up the translation or rotation and moving the needle to the left will slow it down.

In addition, the five buttons do the following:



**Wire frame/ solid mode On/Off:** This toggle switch allows you to view the three-dimensional chart as a wire frame when the switch is on or as a solid object when the switch is off.



**Border Drawing On/Off:** This toggle switch will draw a black outline on each edge of the chart when the switch is on.



**Inline Series On/Off:** This toggle switch will draw the series column in the same XY plane. This option is only available for columnar and bar charts with a data series and it does not appear on the navigation panel if the chart type is not applicable.



**Gouraud Shading On/Off:** Gouraud shading is a sophisticated and very realistic shading feature for three-dimensional charts. When the switch is on it will begin rendering the chart to shade each individual surface.



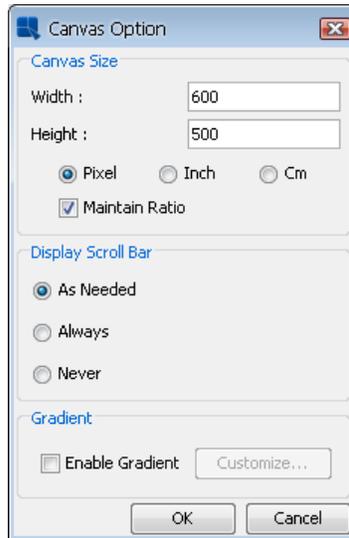
**Animation On/Off:** This toggle switch allows you to start/stop the animation of a three-dimensional chart. The speed of animation may be set using the Navigation Speed control. During animation, all panel controls except the animation speed control are disabled.

## 4.2.4.5. The Viewport

The viewport comprises the central portion of the Chart Designer window. Within the viewport you can select, move, and size all of the various chart elements on the canvas.

### 4.2.4.5.1. The Chart Canvas

The chart canvas is the background on which all of the chart elements are drawn. Its dimensions are the size of the finished chart. You can modify the size of the chart canvas by selection Format → Canvas. This will bring up a dialog prompting you to specify the new canvas dimensions.



*Canvas Formatting Dialog*

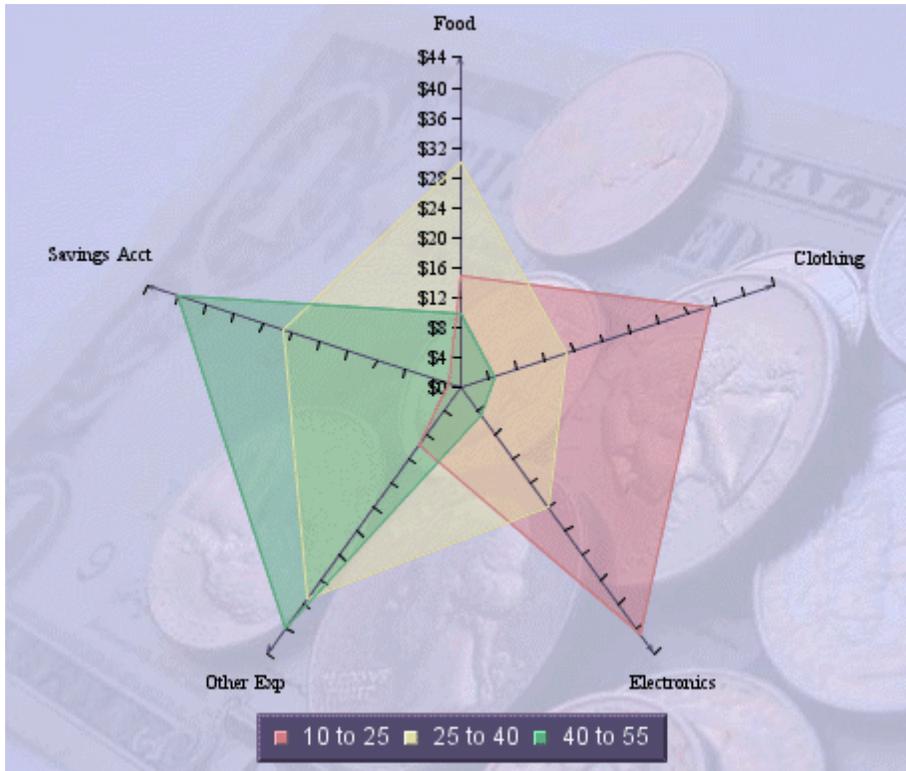
You can specify the canvas size in pixels, inches, or centimeters. From this dialog you can also specify when to use scroll bars in the viewport. By default the viewport will display scroll bars when the canvas is larger than the window. When the canvas is smaller than the viewport window, a dark gray area will appear around it.

If you're creating an embedded chart in a report, the canvas will resize to match the space that you have defined for the chart in the Report Designer. For example, if you set the space in the report to be 3 inches by 4 inches, the next time you edit the chart, the canvas will automatically size to 3 inches by 4 inches.

On this dialog, you can also set up gradient background for the canvas. The gradient settings are the same as in the *Rendering options* described in the Section 4.2.4.1.3 - Format Menu.

#### **4.2.4.5.1.1. Background Images**

You can add an image as the background of the chart instead of having a plain or a colored canvas.



*Radar Chart with Background Image*

You can add a background image by selecting Insert → Background or by clicking the  *Background* button on the toolbar. This will bring up a dialog allowing you to specify the background image for the chart.



*Add Background Image Dialog*

To insert a new image, select the *Enable Background Image* option. If you want to remove an existing background image and use simple background color instead, unselect this option.

There are two ways of inserting background images: either locate the image on the hard drive or retrieve it from an URL.

To locate an image on the hard drive, click on the *Browse* button.

To retrieve image from an URL, enter the URL in the *Image URL* text field. After that, click on the *Refresh Preview* button to verify the URL. If the image from the URL appears in the *Preview* section, the URL is correct.

If you add a background image and save the chart as TPL or CHT, the image itself is not stored with the chart. Only the path or URL is saved. If you move a TPL or CHT chart, you need to be sure that it can still access the image along the path specified. If you save the chart as PAC, the background image will be stored in the PAC file along with the chart.

#### 4.2.4.5.2. Moving and Sizing Chart Elements

You can select any element in the chart by clicking on it. The status bar at the bottom of the Chart Designer window will indicate which object has been currently selected. Clicking and dragging on an object will move it around the chart canvas. Note that some objects like axis or data top labels move in tandem, while other objects like legends move independently.

You can move the entire chart plot by clicking in the plot area and dragging the mouse. This will move the entire chart around the canvas. To resize a chart click the plot area and 8 points will appear in the corners and on the sides of the plot area. Drag any of those points using the mouse to enlarge or shrink the plot area. You can also resize three-dimensional charts by using the zoom control in the navigation panel.

#### 4.2.4.6. Adding Chart Elements

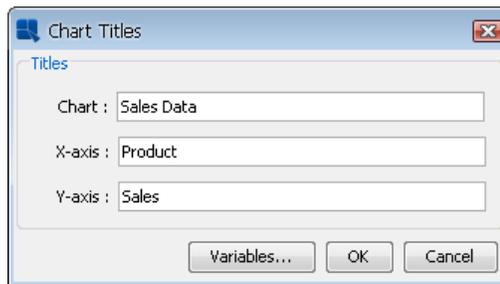
In addition to the default chart elements, ChartDesigner provides a number of additional elements that you can add to a chart.

##### 4.2.4.6.1. Adding Text

There are two ways to add text to a chart: as titles or as plain text elements.

###### **Adding Titles:**

To add titles to a chart, select *Insert* → *Titles*. This will bring up a dialog prompting you to enter titles for the chart.

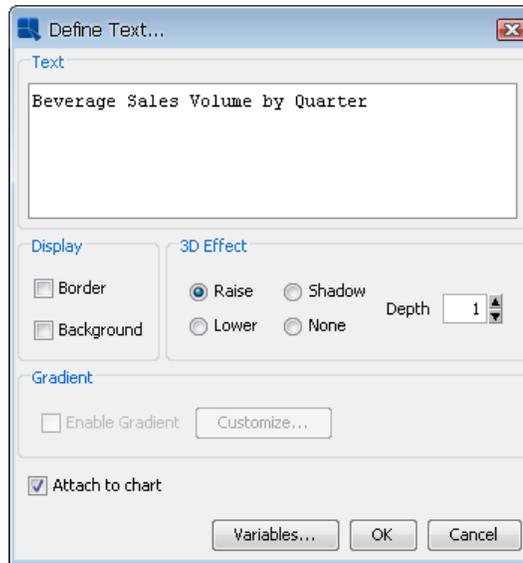


*Add Titles Dialog*

The dialog allows you to specify a main title for the chart and a title for each of the axes. Pie and dial charts, which do not have axes, prompt you for the chart title only. After you have finished specifying the titles click *OK* and they will be added to the chart. Titles are placed and sized automatically. However, they can be moved and the fonts can be changed.

###### **Adding Text:**

To add individual text fields to the chart, select *Insert* → *Text* or click the  *Add Text* button on the toolbar. This will bring up a dialog prompting you to add text to the chart.



*Add Text Dialog*

From this dialog you can specify the text and configure some display options. You can specify whether to draw a border around the text or around a background. You can also specify what effect you want to apply to the background.

On this dialog, you can also set up gradient background for the text label. The gradient settings are the same as in the *Rendering options* described in the Section 4.2.4.1.3 - Format Menu.

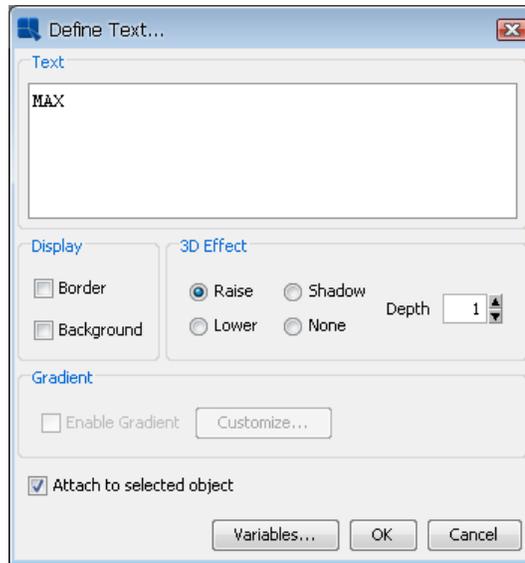
Once you have finished specifying the text, click *OK*. You will then be able to place the text in the chart. A small rectangle will follow your pointer around the chart canvas. Click the mouse where you would like to place the text.

#### **Annotation Text:**

ChartDesigner also supports annotation. Annotation allows you to attach labels or text fields to a particular element, such as a line or the chart plot. For example, you can insert a control line showing the maximum value to a chart and attach a text label called **MAX** to this control line. Each time the maximum value changes, the label will adjust its position along with the control line. For more information about the control lines, please see Section 4.2.4.6.2.3 - Fixed Horizontal/Vertical Lines.

You can specify text to be annotation in two ways. To attach the text to the chart plot, select the *Attach to Chart* option when adding text. To attach the text to a specific object

like a control line, first select the object, and then select *Insert* → *Text* or click the  *Add Text* button on the toolbar. The option for *Attach to selected object* will be automatically checked. Leave it checked and any text you add will be automatically attached to the object.

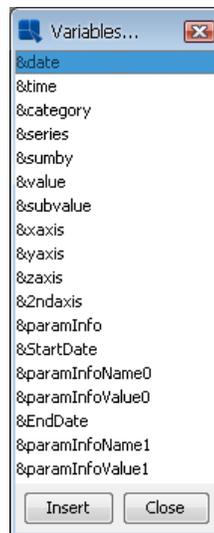


*Add Text (Annotation) Dialog*

#### 4.2.4.6.1.1. TextVariables

ChartDesigner allows you to specify certain variables within text that allow for run time substitution based on certain values/objects in the chart. For example, if your chart uses a parameterized query as the data source, you could use the **&paramInfo** variable to display which parameter value(s) were selected at runtime.

Both the insert titles dialog and the add text dialog have a button marked *Variables* at the bottom. This will bring up a dialog with a list of variables you can use and it will allow you to select one to add to the title or to the text.



*Variables Dialog*

The following text variables are supported:

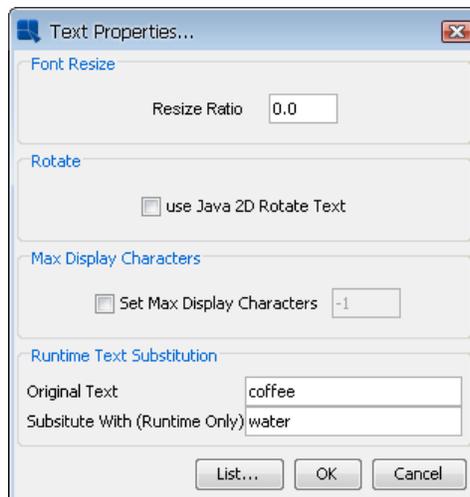
- &drillInfo:** This displays which data point is being drilled on for drill-down charts. This variable does not work for parameter drill-down.
- &paramInfo:** This displays the parameter value(s) that were selected. You can use this variable instead of &drillInfo for parameter drill-down charts.
- &date:** This displays the date when the chart was last drawn/redrawn.
- &time:** This displays the time when the charts were last drawn/redrawn.

<b>&amp;category:</b>	This displays the name of the category column.
<b>&amp;series:</b>	This displays the name of the data series column.
<b>&amp;sumby:</b>	This displays the name of the sum-by column.
<b>&amp;value:</b>	This displays the name of the value column
<b>&amp;subvalue:</b>	This displays the name of the secondary value column
<b>&amp;xaxis:</b>	This displays the name of the column that is mapped to the X-axis. This is for charts that map a value instead of a category to the X-axis like scatter or bubble charts.
<b>&amp;yaxis:</b>	This displays the name of the column that is mapped to the Y-axis. This is for scatter, bubble, and surface charts.
<b>&amp;zaxis:</b>	This displays the name of the column that is mapped to the Z-axis. This is for scatter, bubble, and surface charts.
<b>&amp;2ndaxis:</b>	This displays the name of the column that is mapped to the 2nd-axis.
<b>&amp;paramInfoName&lt;index&gt;:</b>	If the chart contains parameters, this displays the name of the parameter for the selected index.
<b>&amp;paramInfoValue&lt;index&gt;:</b>	If the chart contains parameters, this displays the supplied parameter value for the selected index.
<b>&amp;&lt;paramName&gt;&gt;:</b>	If the chart contains a parameter with this name, this will display the value selected for that parameter.

#### 4.2.4.6.1.2. Text Replacement

ChartDesigner allows you to overwrite a particular piece of text in a chart. This can be useful if the data source does not use particularly intuitive column names. Note that this feature will replace all instances of the text. For example, if you have a column chart without a series that displays a column name for both the X-axis label and the legend item, you cannot use text replacement to change only the label and not the legend item. The text replacement feature will also change only whole strings and not instances where there is a partial match.

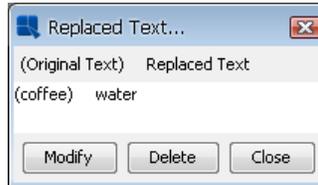
To use text replacement, select Format → Text Properties. This will bring up a dialog allowing you to specify replaced text.



*Text Properties Dialog*

Please note that when making successive changes to the same text, the original text must be used. For example, assume you replaced the word “coffee” with “water”. Now if you want to change “water” to “soft drink” the text replacement should have the original text, which is “coffee” and then “soft drink” as the replacement. To remove any text replacement, simply replace the original string with itself. Hence, using the same example, you would replace “coffee” with “coffee”.

You can see a list of all the original text and the replacements by clicking on the *List* button. This will bring up a dialog listing all of the text replacement in the chart.

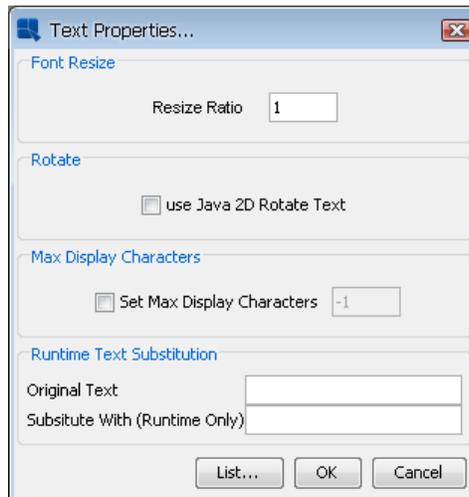


*Replaced Text List*

From this dialog you can select any of the replaced text and modify or undo the replacement by clicking the buttons at the bottom of the dialog.

#### 4.2.4.6.1.3. Automatic Text Resizing

ChartDesigner has the ability to automatically adjust the font size of the text in a chart as it is resized. This is useful if you're using the same chart template to produce a number of charts in different sizes. You can specify a ratio for the font size to adjust based on changes in the chart canvas. To specify a resize ratio, select Format → Text Properties.

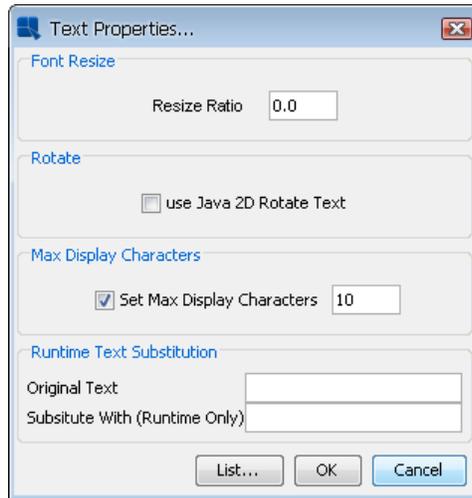


*Text Properties Dialog*

The ratio dictates the relative percent that the font should resize in regards to the canvas. For example, say you resize a chart from 500 x 500 pixels to 250 x 250. With a resize ratio of 1 then text with 12 point font would decrease to 6 point, decreasing by the same percent as the canvas. However, with a resize ratio of 0.5 the font would decrease half as much as the canvas so our 12 point font would only decrease to 9 point.

#### 4.2.4.6.1.4. Text Cropping

Long labels or text in a chart can sometimes take up too much space in the chart plot, leaving little room for the actual chart. For situations like this, ChartDesigner offers a text cropping feature for chart text. Text that is longer than a user-supplied threshold will be truncated with “...”. The hint box for the chart will show the whole label. To specify text cropping, select Format → Text Properties.



*Text Properties Dialog*

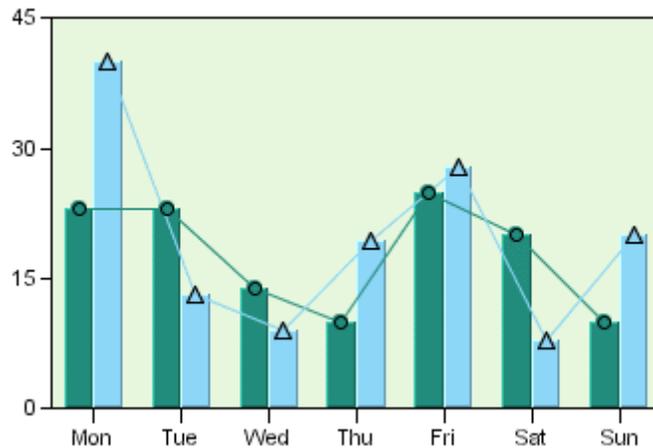
To enable text cropping, check the *Set Max Display Characters* option and specify the maximum character length in the dialog. Any text longer than the specified number of characters will be truncated.

#### 4.2.4.6.2. Adding Lines

ChartDesigner allows you to add and format a number of different types of lines for charts.

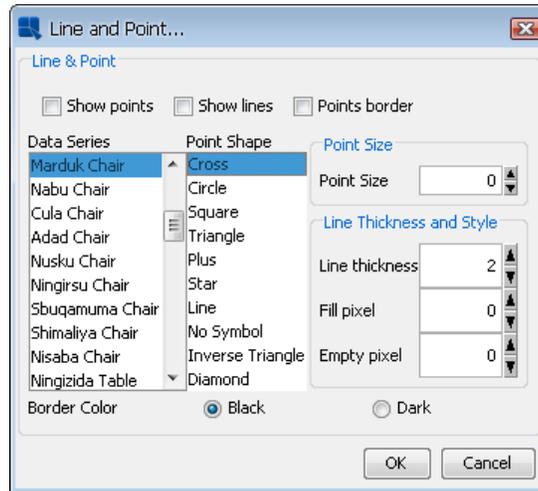
##### 4.2.4.6.2.1. Line and Point Formatting

You can choose to display lines and points for all the data points in the chart for any two dimensional chart type. Note that some chart types already use this representation (i.e. line or scatter charts).



*Column Chart with Lines and Points Defined*

Line and point display is controlled by selecting Format → Line and Point, or click the  *Line and Point* button on the toolbar. This will bring up a dialog presenting several options.



*Line and Point Dialog*

The first three options allow you to specify whether you would like to show lines, points, and a points border for the chart. For radar charts you also have the option of showing areas. The remaining options allow you to customize the line and point displays for each element in the data series.

For each data series element, you can specify the point shape that you would like to use. You can also control the size of the points. The default point size is 0. You can specify point sizes of -1, -2, & -3 which represent sizes of 0.75, 0.5, and 0.25 respectively. At -3 (0.25), the point will be drawn as a dot regardless of the selected point shape.

For lines you can specify the line thickness, as well as customize a dash pattern. The dash pattern is created by specifying the number of filled pixels and the number of empty pixels (between 0 and 255). The line is then drawn by dividing into segments - the number of filled pixels followed by the number of empty pixels. Setting 0 for both will result in a solid line. Setting 255 for both will result in no line being drawn.

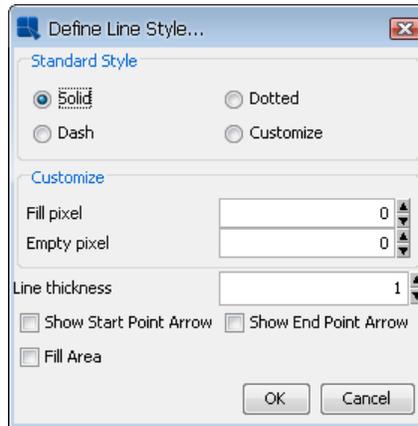
The last option allows you to change data point symbol border color to black or darker shade of symbol color.

#### 4.2.4.6.2.2. Floating Lines

Floating lines are free-form lines that can be arbitrarily added to any place on the chart canvas. Often floating lines are used to point to a specific element in a chart. To add a floating line select Insert → Line, or click the  *Insert Line* button on the toolbar.

When you select this option, your mouse pointer will change to a cross. Click within the chart canvas where you would like the line to begin. Each additional click will add a point to the line, allowing you to add another segment. This way you can use floating lines to draw shapes as well. Once you have finished, right-click to stop drawing. The line will then be added.

Once a line has been placed on the canvas, it cannot be moved individually. It will move with the chart plot, like annotation text. To specify options for a floating line, first select it, and then select Format → Line and Point or click the *Line and Point* button on the toolbar. This will bring up a dialog allowing you to format various properties for the floating line.



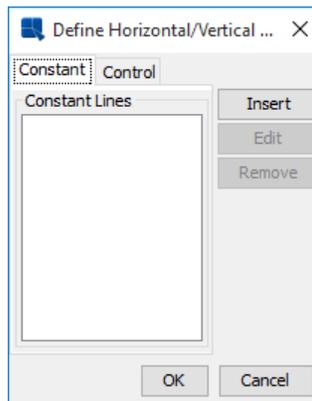
*Line and Point Dialog for Floating Lines*

The dialog allows you to specify a standard line style or to create a custom dash pattern in the same manner as line and point formatting. You can also specify the line thickness in pixels. The checkboxes at the bottom of the dialog allow you to place an arrowhead at the start and/or end of the line, as well as fill the area enclosed by the line to create a solid shape.

Floating lines can be deleted by selecting them and hitting the delete key.

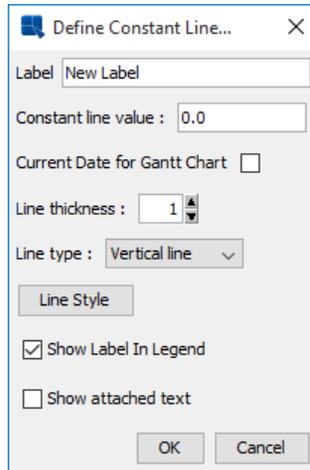
#### 4.2.4.6.2.3. Fixed Horizontal/Vertical Lines

Fixed horizontal or vertical lines are lines that are drawn on one of the chart axes. These lines can also be drawn on three-dimensional charts where they appear as planes. There are two types of fixed lines: constant lines and control lines. Constant lines are lines that are fixed to a certain value in an axis. Control lines are drawn based on computed values that allow you to spot data points that are outside of certain value ranges. To add either type of line to a chart, select *Horz\Vert Line* from the *Insert* menu. This will bring up a dialog showing the list of existing horizontal/vertical lines, allowing you to edit the selected line, remove the selected line, and/or create a new line.



*Define Horizontal/Vertical Lines Dialog*

Clicking on *Insert* or clicking on *Edit* when an existing line is selected, respectively, will bring up a dialog allowing you to configure the selected line.



Constant Line Dialog

For constant lines you need to specify a label for the line, as well as the numeric value to use for the line. Note that for the category axis, the data points start with 0.5. You can also specify the line thickness and whether the line is horizontal or vertical. The last two options allow you to add an item to the chart legend for the line and whether to display any annotation for the line.

For Gantt charts, there is one extra option called *Current Date for Gantt Chart*. If you choose this option, the *Constant line value* field will deactivate and the current date will be used as the line value (the line position will be updated every time you run the chart).

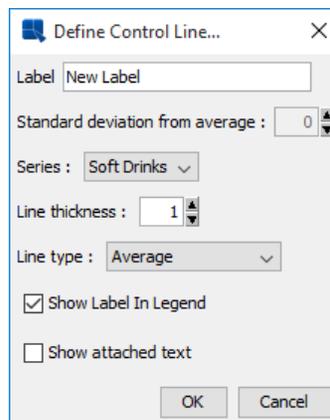
For radar charts, the horizontal/vertical option is disabled. Radar charts draw constant/control lines at the same point around all the chart axes (in a similar manner to the radar grid). In addition, for radar charts, an additional option named *Circular Style* is present. By default, lines in radar charts are drawn in a segmented fashion - straight lines connect the points on each axis. Selecting this option will draw the constant/control as a circle.



**Note**

If you have not specified any annotation for the line then none will appear if you select the last option. For more on adding annotation to a chart, please see Section 4.2.4.6.1 - Adding Text.

To add a control line, click on the *Control Line* tab in the dialog.

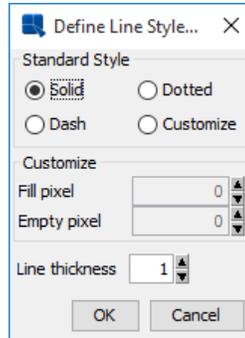


Control Line Dialog

For control lines you need to specify which series element you want to compute the value for (this option does not appear if no series is present) and how to compute the value. Options for control lines are average, minimum, maximum, and multiples of standard deviation.

After you have specified all of the options, the line will be added to the chart or the selected line will be changed, respectively. To edit any of the properties specified in the previous dialogs, you can select Insert → Horz/Vert Line again and select the line from the list. You can also double-click on the line that you want to modify.

You can change the appearance of fixed lines by first selecting the line and then selecting Format → Line and Point, or clicking the *Line and Point* icon on the toolbar. This will bring up a dialog allowing you to customize the line.



*Line and Point Dialog for Fixed Lines*

This dialog allows you to specify a standard line style or to create a custom dash pattern in the same manner as line and point formatting.

#### 4.2.4.6.2.3.1. Multiple control lines for Stack Type Charts

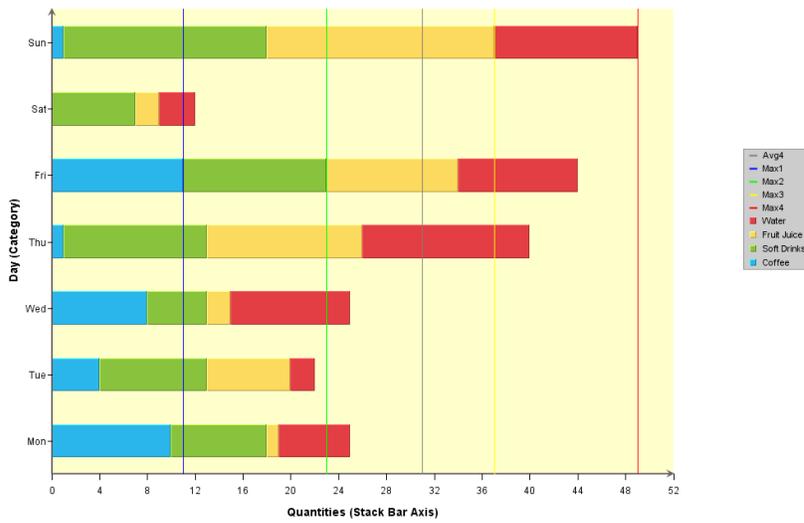
For stack type chart, i.e., Stack Column, Stack Bar and Stack Area, we have API function:

```
...
newControlLine(int linetype, String label, int level);
...
```

can draw control line with indicated stack level. For more information about this option, see Section 8.2.6.2.5 - Adding Multiple Control Lines to Stack Type Chart. If a stack column chart with the combo chart of stack area, the API can only works on the main axis data, i.e., stack column data; In this case, the API code cannot get the secondary value to draw control lines on stack area as combo.

You can also set the control line color with this API code. The following image shows the muliple control lines in different color, and display the meaning of them in the legend of the chart.

Drink Quantities by Day



Stack Chart with Multiple Control Lines

#### 4.2.4.6.2.4. Trend Lines

A powerful feature of ChartDesigner is the ability to add trend lines to charts. Trend lines can help to show more details of a chart's data by exposing and highlighting certain trends.

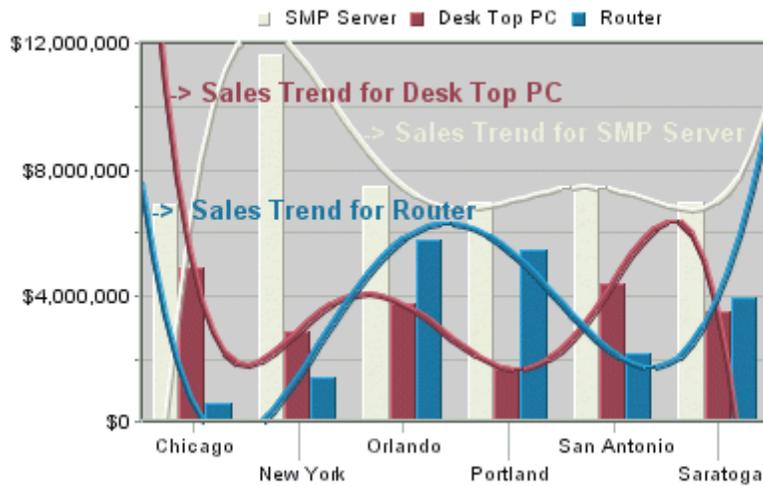
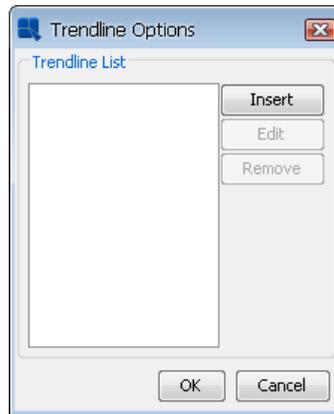


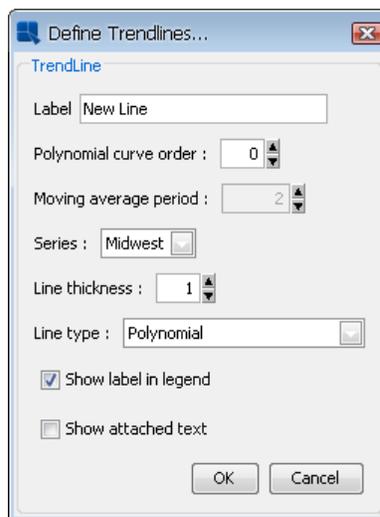
Chart with Trend Lines

To add a trend line to a chart, select Insert → Trendline. This will bring up a dialog showing the list of existing trend lines, allowing you to edit the selected trend line, remove the selected trend line, and/or create a new trend line.



*Trend Line Options Dialog*

Clicking on *Insert* or clicking on *Edit* when an existing trend line is selected, respectively, will bring up a dialog allowing you to configure the selected trend line.



*Define Trend Lines Dialog*

In this dialog you can specify a label for the line, as well as what element of the data series to base the calculation on. The following types of trend lines are supported: a polynomial of any degree (please note that a linear trend line is a polynomial trend line of the 1st degree, i.e. the *Polynomial curve order* option has to be set to 1), a power, exponential, logarithmic, a moving average, an exponential moving average, a triangular moving average, cubic B-spline, and a normal distribution curve. For moving averages you will need to specify the average period and for a polynomial you will need to specify the curve order. You can also specify the thickness of the line and configure whether a label in legend and the attached text should be shown. In case the chart has data series, you can configure the trendline for a specific series.

After you have specified all of the options, the trend line will be added to the chart or the selected trend line will be changed, respectively. To edit any of the properties specified in the previous dialog, you can select 'TrendLine' from the Insert menu again, and select the line from the list. You can change the appearance of the trend line by first selecting it, and then selecting Format → Line and Point, or clicking the *Line and Point* button on the toolbar. This will bring up a dialog allowing you to customize the lines.

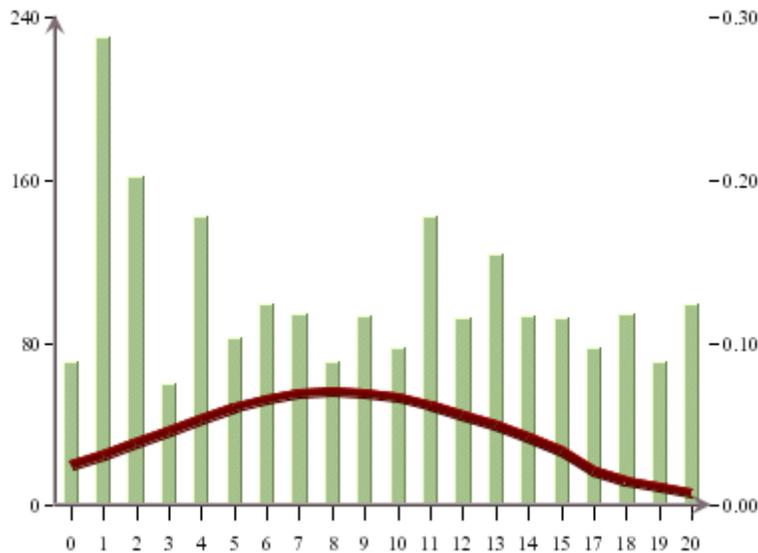


*Line and Point Dialog for Trend Lines*

This dialog allows you to create a custom dash pattern in the same manner as line and point formatting.

#### 4.2.4.6.2.4.1. Normal Distribution Curve

A special type of trend line that allows you to draw a normal distribution curve for the data in the chart. In order to plot a normal distribution curve, the chart must either be a two-dimensional column or bar chart, it cannot have a data series, and the category should be numeric. Assuming these conditions are met, you can specify a normal distribution curve as one of the trend line options.

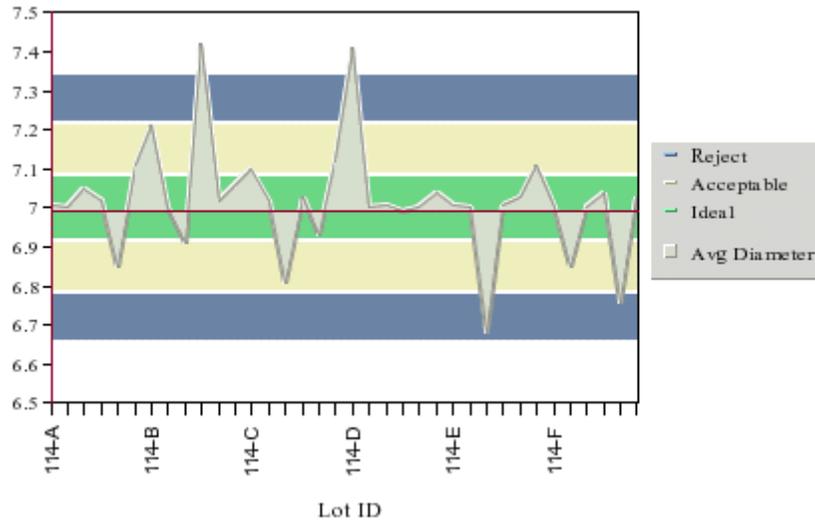


*Chart with Normal Distribution Curve*

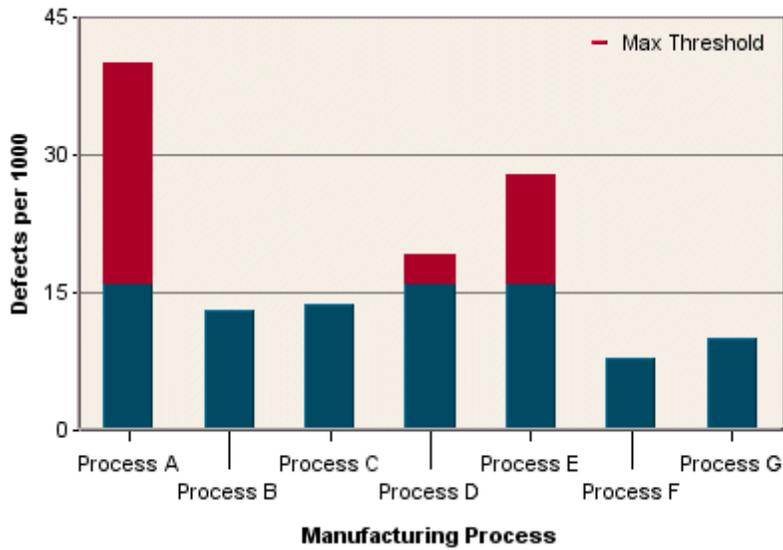
Since the scale for the curve is usually different than the scale for the value axis, the curve is shown on a secondary axis. You can modify the scale by changing the scale for the secondary axis.

#### 4.2.4.6.3. Adding Control Areas

Control areas are useful for comparing the chart data against a certain range of data. For most two-dimensional charts, control areas are drawn as filled areas on the chart plot between a range of values on the chart's value axis and/or category axis. The data points are then drawn over top of the control areas giving you a quick visual reference to see which data points fall within the designated range. Instead of drawing a background area on the plot, the control areas can also be shown only where the data points intersect the control area. This feature gives users a clear visual reference when specific threshold values are reached.

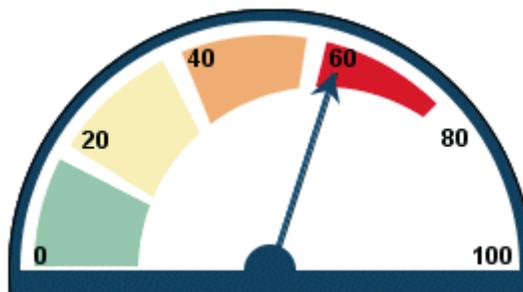


*Two-Dimensional Area Chart with Control Areas*



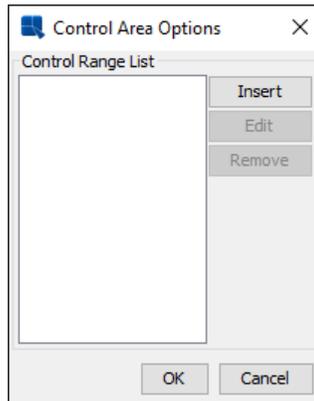
*Column Chart with Control Area Drawn for Data Points*

A special instance of control areas can be used for dial charts. For dial charts control areas are drawn as arcs on the face of the dial, allowing you to see if the dial hands (data points) fall within the range. Note that control areas are not available for radar and pie charts.



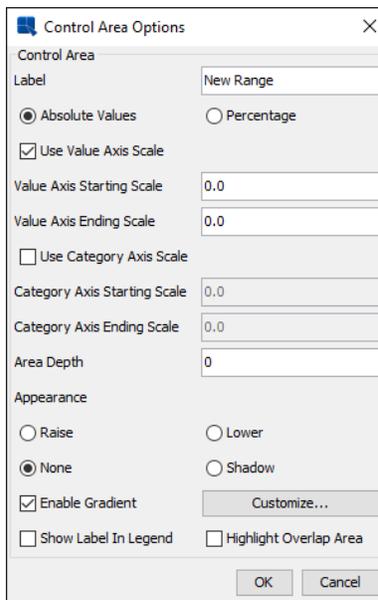
*Dial Chart with Control Areas*

To add a control area to a chart, select **Insert** → **Control Area**. The following dialog will appear showing the list of existing control areas, allowing you to edit the selected control area, remove the selected control area, and/or create a new control area.



*Control Area List*

Clicking on *Insert* or clicking on *Edit* when an existing control area is selected, respectively, will bring up a dialog allowing you to configure the selected control area. If your chart is not a dial chart, the following dialog will open.



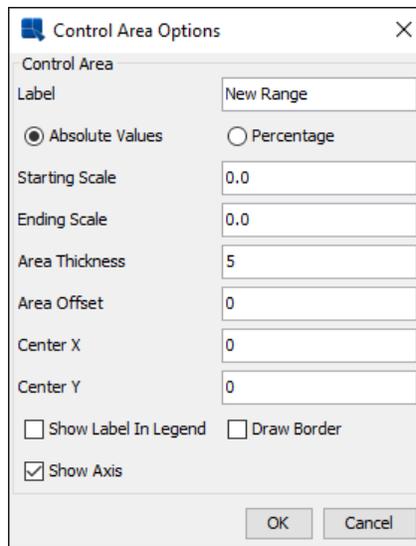
*Control Area Configuration Dialog*

The following options are provided for control areas:

- Label:** This option allows you to specify a label for the control area.
- Absolute Values:** This option allows you to specify the scale in absolute values.
- Percentage:** This option allows you to specify the scale in percentage.
- Use Value Axis Scale:** This option allows you to specify whether the control area should be bounded by values on the value axis of the chart.
- Value Axis Starting Scale:** This is the lower bound for the control area on the value axis.
- Value Axis Ending Scale:** This is the upper bound for the control area on the value axis.

<b>Use Category Axis:</b>	This option allows you to specify whether the control area should be bounded by values on the category axis of the chart.
<b>Category Axis Starting Scale:</b>	This is the lower bound for the control area on the category axis.
<b>CategoryAxis Ending Scale:</b>	This is the upper bound for the control area on the category axis.
<b>Area Depth:</b>	This option specifies the depth for any of the appearance styles. If no style is selected, the depth will have no effect.
<b>Appearance:</b>	This option allows you to specify a 3D or shadow effect for the control area. If the area depth is specified as zero, the appearance will not take effect.
<b>Enable Gradient:</b>	Enable color gradient for the control. Gradient settings are described in Section 4.2.4.1.3 - Format Menu
<b>Show Label In Legend:</b>	This option specifies whether or not to show the control area label in the chart legend.
<b>Highlight Overlap Area:</b>	This option will only show the control area where the data points overlap the control area boundaries.

If your chart is a dial chart, then a different dialog will appear when you select Insert → Control Area and then click on the *Insert* or *Edit* button.



*Control Area Dialog for Dial Charts:*

The following options are provided for dial chart control areas:

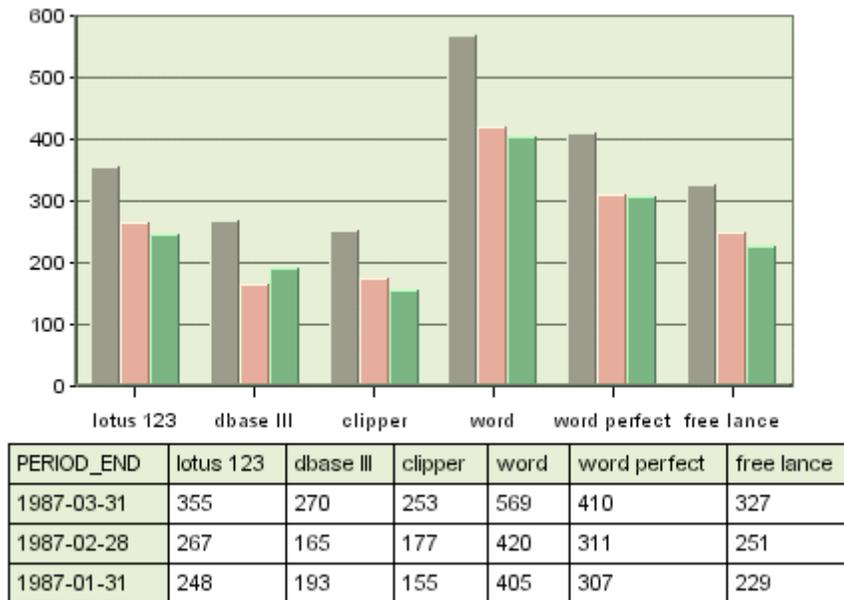
<b>Label:</b>	This option allows you to specify a label for the control area.
<b>Absolute Values:</b>	This option allows you to specify the scale in absolute values.
<b>Percentage:</b>	This option allows you to specify the scale in percentage.
<b>Starting Scale:</b>	This is the value where the control area begins.
<b>Ending Scale:</b>	This is the value where the control area ends.
<b>Area Thickness:</b>	This option allows you to set the thickness for the control area
<b>Area Offset:</b>	This option allows you to specify the offset in pixels from the edge of the dial chart

- Center X:** This sets the X coordinate for the center of the control area. 0 shares the same center point as the dial face. You can specify a new number (either negative or positive) pixels to specify an offset position from the center of the dial.
- Center Y:** This sets the Y coordinate for the center of the control area. It works in the same way as the previous option.
- Show Label in Legend:** Specifies whether to show the control area label in the chart legend.
- Draw Border:** This option allows you to draw a border around the control area.
- Show Axis:** This option allows you to show or hide the axis for the remaining area not covered by the control range.

After you have specified all of the options, the control area will be added to the chart or the selected control area will be changed, respectively. To edit any of the properties specified in the previous dialog, you can select Insert → Control Area again, and select the control area from the list. You can also double-click on the control area that you want to modify.

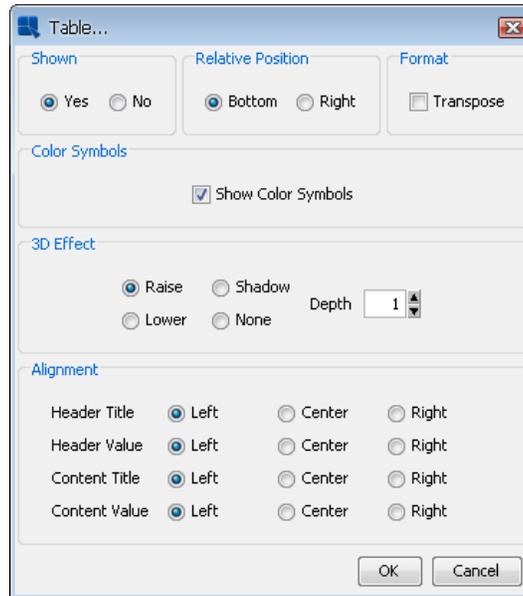
#### 4.2.4.6.4. Adding Tables

In addition to displaying charts, you can also display a table showing the data points displayed in the chart. The table can be placed below or to the right of the chart plot.



*Chart with Table*

To add a table to a chart, or to modify the various display options for a table, select Format → Table. This will bring up a dialog allowing you to customize various options for the table display.



*Format Table Dialog*

From this dialog you can specify whether or not to display the table, as well as what relative position to give the table either to the bottom or right-hand side of the chart plot. You can also specify a 3D effect for the table and its depth.

The *Transpose* checkbox allows you to swap the columns and rows of the table. By default, the category elements are drawn as columns and the data series elements as rows.

The *Show Color Symbols* option allows you to show/hide color boxes for data series in the table.

quarter/drink	Coffee	Fruit Juice	Soft Drinks	Tea	Water
Q1	181	89	264	114	303
Q2	149	144	212	144	156
Q3	169	70	498	162	275
Q4	195	171	230	144	186

*Chart table with color boxes*

The *Alignment* options allows you to specify horizontal alignment of the text in the table cells, either left, center, or right. The alignment can be set for row headers, column headers, and inner table cells.

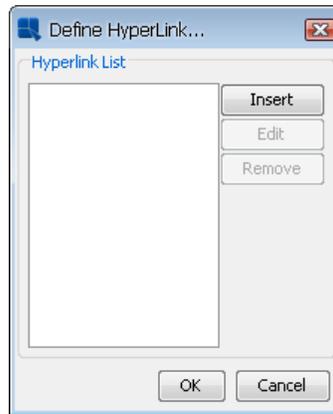


### Note

If there isn't enough room in the chart canvas, not all data points will be displayed in the table. The table size adjusts with the canvas size and also with the font size in the table cells.

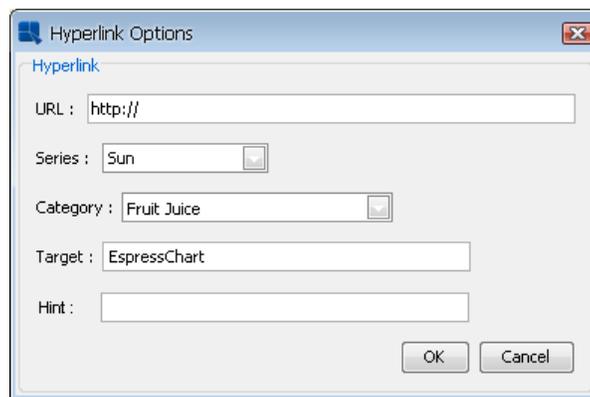
#### 4.2.4.6.5. Adding Hyperlinks

ChartDesigner has an capability to add hyperlinks to any data point in a chart. Links can be specified for either single data points or multiple elements. Any added hyperlinks will be applied to both the data points on the chart and to their respective fields on the legend. To add a hyperlink to a chart, select *Insert* → *Link* or right-click on a data point and select *Insert Link* from the pop-up menu. This will bring up a dialog showing a list of existing hyperlinks which you can configure, remove, and/or create new ones.



*Insert Link Dialog*

Clicking on *Insert* or clicking on *Edit* when an existing hyperlink is selected will bring up a dialog allowing you to configure the selected hyperlink.



*Define Link Dialog*

The URL field allows you to specify a Web page that you want to link.

The *Series* and *Category* drop down menus allows you to select an element in the data series and category elements for the hyperlink. You can also link to all data series elements or all category elements.

You can specify the `Target` parameter recognized by HTML when specifying a hyperlink to be attached to a data point or data series. This can be used to determine whether the new HTML page should open in a new browser window, in the same browser window, or whether the new page should occupy the same portion of the page as the current page.

The *Hint* field allows you to enter text that will pop-up when you move your mouse cursor over a data point. If you want to create pop-up labels without hyperlinks, you can leave the URL field blank and only specify the hint.

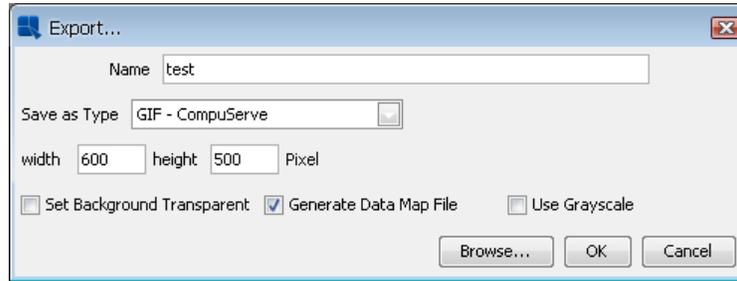
#### 4.2.4.6.5.1. Viewing Hyperlinks

If you specify hyperlinks for charts, they will be active only when the chart is exported to Flash format\*. For most image formats such as PNG, JPG, GIF, etc, an image map file containing information for the link will be automatically generated when you export the chart. You can insert the image and image map into an HTML file to view the image with clickable links.



### Note

\* For Flash export, when the user clicks on the hyperlink, there will likely be a warning message prompted by Flash saying that there was a potentially unsafe operation. You can turn this warning off by clicking on settings and adding the chart into the list of trusted locations.



*Export Dialog*

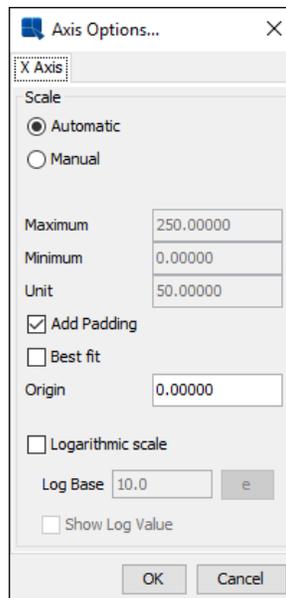
## 4.2.4.7. Formatting Chart Axes

ChartDesigner provides a number of extensive formatting capabilities for the chart axes. Users can customize everything from the axis scale to the way how axis labels should be displayed.

### 4.2.4.7.1. Axis Scale

By default, the scale of any value axes in the chart is calculated to provide a 'best fit' for the data being plotted. This is often a useful feature if the data being displayed can change radically. However, you may often want to set

the scale of the axes manually. To modify the axis scale, select Format → Axis Scale, or click the  *Axis Scale* button on the toolbar. This will bring up a dialog allowing you to format the scale for any value axes in the chart.



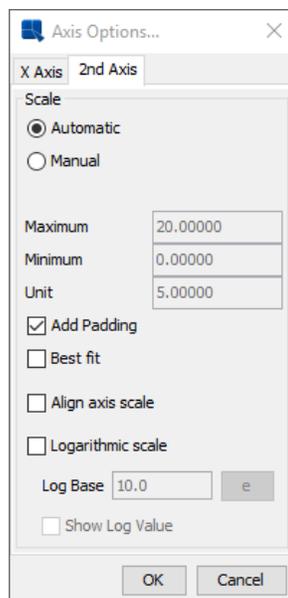
*Axis Scale Dialog*

The following options are provided:

- Automatic:** This turns on automatic scaling for the axis. This is the default option.
- Manual:** This turns on manual scaling, allowing you to set the axis scale to your preference.
- Maximum:** This is the highest value on the axis scale.
- Minimum:** This is the lowest value on the axis scale.
- Unit:** This is the step interval between successive labels.
- Add Padding:** This will raise the highest value of the axis to create a cushion between the max value of the data and the top of the chart.

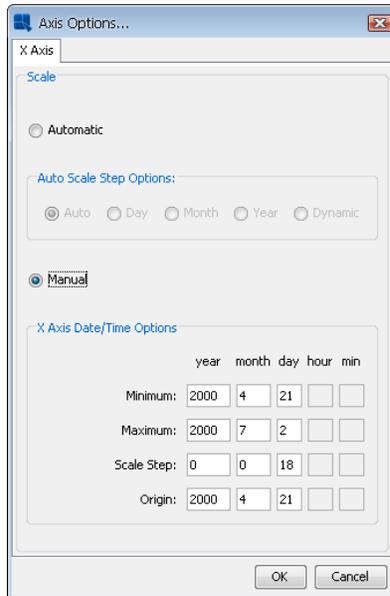
- Best fit:** This will automatically place the origin of the chart based on the minimum and maximum values of the data.
- Origin:** This allows you to specify where the X and Y axes should intersect. This is usually set to zero.
- Logarithmic Scale:** This option creates a logarithmic scale for the given axis. It's valid only if the axis in question contains positive values.
- Log Base:** This allows you to specify the base for the log value.
- Show Log Value:** This specifies whether to show log values in the axis labels or not.

The axis scale dialog will have a tab for each axis in the chart. There's a unique option available for secondary axes which allows you to align the axis scale. It will apply all options from the primary axis to the secondary axis, giving both axes the exact same scale.



*Axis Scale Dialog for Secondary Axis*

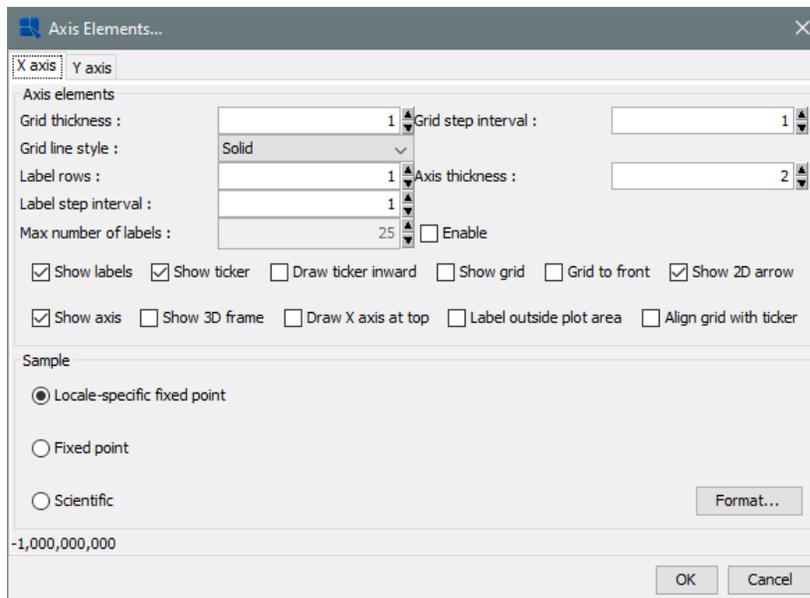
The axis scale dialog is different for a Gantt chart. You can still select the *Automatic* option to have the scale configured automatically or the *Manual* option to set the scale manually. The *Automatic* axis scale has a few Auto Scale Step Options: Auto, Day, Month, Year, and Dynamic. The *auto* option always finds a best fit. The *dynamic* option also finds a best fit, but unlike the *auto* option, it uses standard step intervals only (for example: 1 month, 1 year etc...). When the *Manual* axis scale is chosen, the *Maximum*, *Minimum*, *Unit* and *Origin* are replaced with *Maximum Date*, *Minimum Date*, *Scale Step* and *Origin Date* respectively and these new settings take in a Date/Time (represented by yyyy, MM, dd, hh, mm).



*Axis Scale Dialog for Gantt Chart*

#### 4.2.4.7.2. Axis Elements

The appearance properties of the axes and the axis labels are controlled through the axis elements dialog. To invoke the axis elements dialog, select Format → Axis Elements or click the  *Format Value Elements* button on the toolbar. This will bring up the following dialog, allowing you to customize elements in all chart axes. You can also customize the appearance of dial and pie chart labels from this dialog.



*Axis Elements Dialog*

A tab will appear in this dialog for each axis in the chart. The dialog allows you to perform the following options. Note that some options are only available for certain chart types, certain data types, and on certain axes.

**Grid thickness:** This allows you to specify the thickness of any grid lines along the axis.

**Grid step interval:** This option allows you to set the grid step interval for any grid lines along the axis.

<b>Grid line style:</b>	This option allows you to select the grid line style (solid, dotted, dash).
<b>Label rows:</b>	This option allows labels to be displayed in alternating rows. This can prevent overlapping. This option is only available for X-axis.
<b>Axis thickness:</b>	This option allows you to set the thickness of the axis in pixels. Note that this setting is applied to all axes in the chart.
<b>Label step interval:</b>	This option allows you to set the label step interval for the data. For example, setting this to 2 will draw the label for every other data point in the chart.
<b>Label interval unit:</b>	This option allows you to select the unit to be used when sorting and representing time-based data (date, time, or timestamp). Selecting tickers will use the data as it is read by Chart Designer. You can also select Dynamic for time-based data and that will choose an appropriate scale (depending on number of data points and range of data).
<b>Max number of labels:</b>	This option allows you to select the maximum number of labels and tickers displayed on the axis. If the number of labels exceeds the max count, the label step will be re-calculated.
<b>Ascending/Descending:</b>	This option allows you to order and filter time-based data. You can sort the data in ascending or descending order, as well as specify the starting (or ending) point for the data.
<b>Show labels:</b>	This option allows you to remove or display the labels for each axis.
<b>Show ticker:</b>	This option allows you to remove or display the axis tickers.
<b>Draw ticker inward:</b>	This option will draw the axis tickers inside the plot area instead of outside (default).
<b>Show sub-tickers:</b>	This feature is only available for the value axis when the axis scale is set to logarithmic with a log base of 10. This feature will draw interval tickers (non-uniform) between the points on the value axis.
<b>Show grid:</b>	This option allows you to remove or display the grid for each axis.
<b>Grid to front:</b>	This option allows you to draw the grid lines on top of the data elements in the chart. By default the data points are drawn on top of the grid.
<b>Show 2D arrow:</b>	This option allows you to remove or display the arrowhead at the end of the axis. Note that this option applies to all chart axes and it is only available for two-dimensional charts.
<b>Show axis:</b>	This option allows you to remove or display the axis (for two-dimensional charts) or the wall (for three-dimensional charts).
<b>Show 3D frame:</b>	This option allows you to remove or display a frame around the chart. Note that this option applies to all chart axes and it is only available for three-dimensional charts.
<b>Label outside plot area:</b>	This option sets the labels to be placed outside of the plot area, irrespective of where the axis is. This feature can be useful for category axis labels if you're plotting data with both positive and negative values.
<b>Align grid with ticker:</b>	This option aligns the grid line with the ticker instead of placing it between tickers. This places the ticker and the corresponding grid line along the same line. This option only applies to the category axis of column-type charts.
<b>Swap Y-axis position:</b>	This option will swap the primary and secondary value axes. This option can only be found under the 2nd Axis tab.

**Draw X-axis at top:**

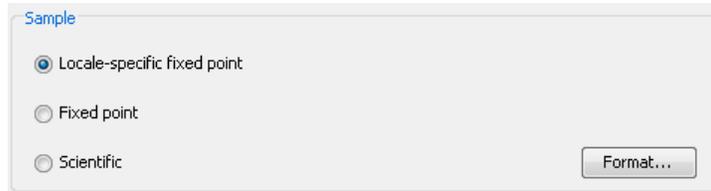
This option allows the X-axis to be positioned at the top of the chart instead of the default bottom position. This option is available for two-dimensional column, bar, scatter, high-low, HLCO, bubble, and Gantt charts.

**4.2.4.7.2.1. Axis Label Formatting**

The axis elements dialog also allows you to format the appearance of the axis labels, depending on what type of data is plotted on the axis. The *Format Options* portion of the dialog contains the label formatting dialog.

**Formatting Numeric Data**

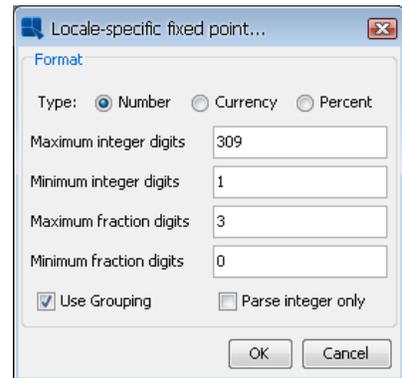
For numeric data there are three primary options for display formatting: locale-specific fixed point, fixed point, and scientific. Additional options will be displayed if you click on the *Format* button.



*Numeric Data Format Options*

**Locale-Specific Fixed Point**

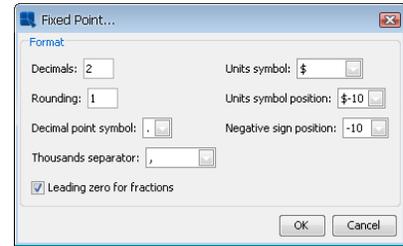
This will change the format of the data depending on the locale in which it is being viewed. Additional formatting for this option allows you to specify whether the data should be displayed as a number, currency, or percentage. In addition, you can set the maximum and minimum number of integer digits and fraction digits. Other display attributes will vary depending on locale.



*Locale-Specific Formatting Options*

**Fixed Point:**

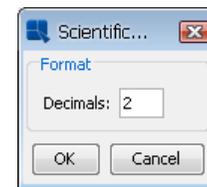
This will keep the data format consistent, regardless of locale. Additional formatting for this option allows you to set the number of decimals, rounding for digit number, unit symbols, negative sign position, decimal and thousands separator, and enable leading zeros for fractions



*Fixed Point Formatting Options*

**Scientific:**

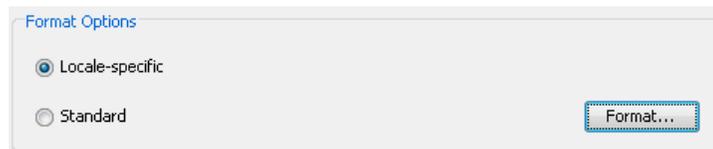
This will display the data in scientific notation. Additional formatting for this option allows you to set the number of decimals.



*Scientific Formatting Options*

**Formatting Date/Time Data**

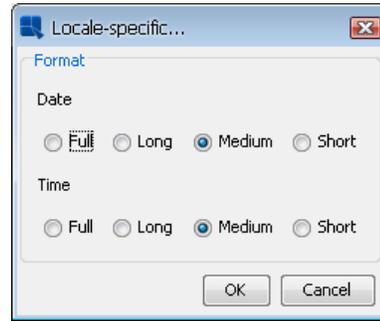
For date/time data there are two primary options for display formatting: locale specific and standard. Additional options will be displayed if you click on the *Format* button. The available options will vary depending on the nature of your data. Date, time, and timestamp data will bring up date, time, and date & time options respectively.



*Date/Time Data Format Options*

**Locale-Specific:**

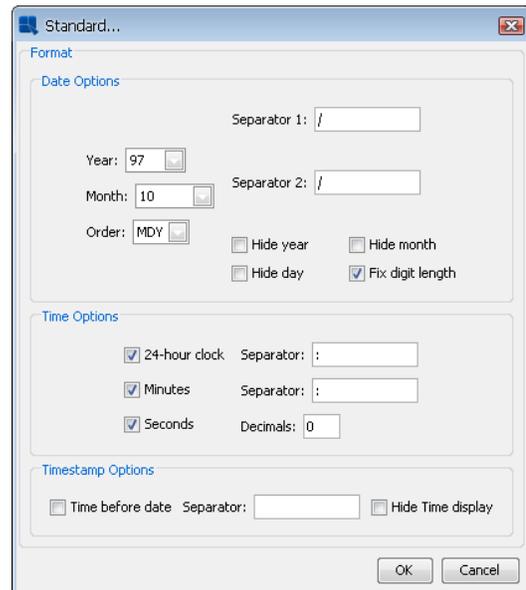
This will change the format of the data depending on the locale in which it is being viewed. Additional formatting for this option allows you to select full, long, medium, or short notations for date and time information. Other display attributes will vary depending on locale.



*Locale-Specific Formatting Options*

**Standard:**

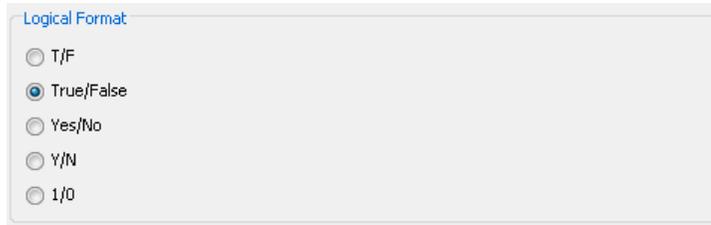
This will keep the data format consistent, regardless of locale. Additional formatting for this option allows you to select year and month displays, as well as the order in which month, day, and year information is presented. You can also select the characters that you want to be used as separators. Time options allow you to display hours, minutes, and/or seconds and also to select the separators between them. For timestamp data, you can select to display the time before or after the date, as well as the separator to be used between them.



*Standard Formatting Options*

**Formatting Logical Data:**

There are five options available for displaying logical or Boolean data: T/F, True/False, Yes/No, Y/N, and 1/0.



*Logical Data Formatting Options*

Any changes you make to the data formatting will take effect after you click on the *OK* button in the axis elements dialog. Note that there are no additional formatting options for string data.

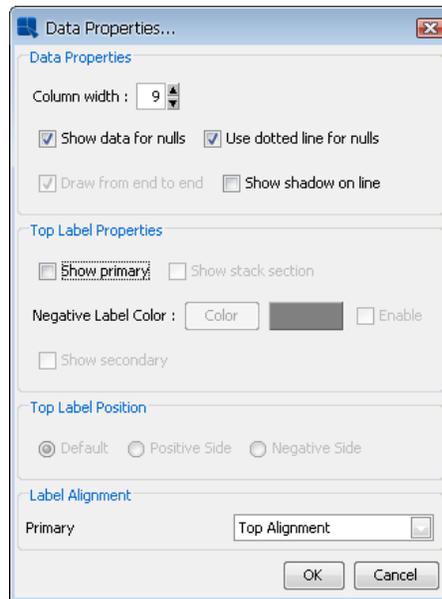
### 4.2.4.8. Formatting Plot/Data Elements

ChartDesigner provides a number of ways to customize and configure the way data points are drawn and annotated on the chart, as well as the chart plot itself.

#### 4.2.4.8.1. Data Properties

Many of the data display options are controlled through the data properties dialog. From this dialog you can control the size of bars/columns, set display options for null values, and specify options for data labels. To invoke the data

properties dialog, select *Format* → *Data Properties* or click the  *Data Properties* button on the toolbar. This will bring up the following dialog:



*Data Properties Dialog*

This data properties dialog contains the following options:

**Column width:**

This specifies the ratio of the bar/column width with respect to the gap between successive bars in the chart. Each unit represents 1/10<sup>th</sup> of the space between data points. Therefore, entering **9** would leave 10% of the space between data points blank, while **10** would eliminate all space between bars/columns. This option only pertains to two-dimensional bar, column, stack bar, stack column, high-low, HLCO, and Gantt charts. To control the column thickness in three-dimensional charts, you can use the thickness of shape slider in the navigation panel.

**Show data for nulls:**

This option will connect lines when null data is present. For example, if you have three points and the value of point 1 is 4, point 2 is null, and point 3

is 6, then a line will be drawn from 4 to 6. This option is only available for line charts and other two-dimensional charts with lines. All other chart types will not plot null data.

**Use dotted lines for nulls:**

You can use this option to replace the full line with a dotted line. Like the show data for nulls option, this property is only available for lines.

**Draw from end to end:**

This option allows you to draw two-dimensional line and area charts across the entire plot area, rather than offsetting to the first and last data points on the chart.

**Show shadow on line:**

This option allows you to use shading on two-dimensional lines. However, the line must be thicker more than one pixel.

**Show primary:**

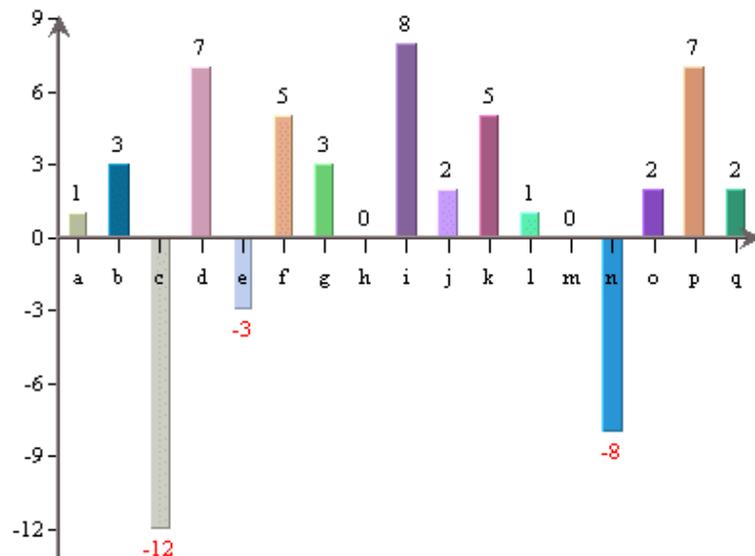
This option will display data top labels for the primary values in the chart.

**Show stack section:**

This option will display individual labels for each stack section for stack bar, stack column, and stack area charts.

**Negative Label Color:**

This option will display the top labels with a value smaller than that of the origin in a different color that can be selected using the *Color* button after enabling the feature.



*Chart with Colored Negative Top Labels*

**Show secondary:**

This option will display data top labels for the secondary values in the chart.

**Top label position:**

This option allows you to specify where the data top labels should be drawn. By default, they are drawn above data points if they are positive and below data points if they are negative. Other options allows you to draw the labels to either positive or negative side.

**Label Alignment:**

This option allows you to set the alignment for the data top labels. You can draw them at the top, bottom, or middle of the data points. In addition, you can select to draw the label inside the data point at the top or bottom. An additional option stack charts offers you to set the alignment for stack section labels.

**4.2.4.8.2. Date/Time Based Zooming**

For charts displaying date or time data on the category axis, ChartDesigner provides a unique feature allowing users to perform date/time based zooming. Using this feature, you can group the category elements into user-defined

intervals and aggregate the points in each group. You can also filter the data by specifying upper and lower bounds for the results.

For example, suppose your data contains daily sales volume for the past two years. Using zooming, you could aggregate the data to look at average volume per month, quarter, or year. Using the upper and lower bounds, you could narrow the range to look at weekly sales volume within a specific quarter.

Zooming is available for all chart types except scatter, surface, box, dial, polar, radar, bubble, and Gantt.

#### 4.2.4.8.2.1. Adding Zooming

When you create a new chart with date, time, or timestamp data in the category axis, you can specify zooming options by selecting **Format** → **Time Zooming Options**.

*Zoom Options Dialog*

This dialog allows you to specify a lower and upper bound for the data, as well as the interval by which you would like to group the data. The scale specified here must be within the maximum and minimum scale specified in the aggregation options dialog.

This dialog also allows you to preserve a linear scale for the chart. By setting the *Linear* option to true, the chart will always display points for the grouped intervals, even if there is no data associated with a particular group. For example, say again that you are measuring sales volume over a three month period - April, May, and June. If the input data has no records for May and you set the *Linear* option to true, a point will be drawn for May with a value of zero. If you set the *Linear* option to false, the data point for April will be immediately followed by June.

You can disable/enable zooming, as well as the lower and upper bound restrictions by using the checkboxes at the bottom of the dialog.

If you enable zooming (if you check *Enable Zooming* option), the dialog *Aggregate Options* will appear, prompting you to specify aggregation options for the grouped data points.

*Aggregation Options Dialog*

In this dialog, you can specify the Primary Aggregation, as well as the maximum and minimum scale increments that can be used when zooming the data. After you have specified your desired options, click on the *OK* button to return to zooming options.

Once you have finished specifying all the options, click on the *OK* button and the zooming will be applied to the chart.

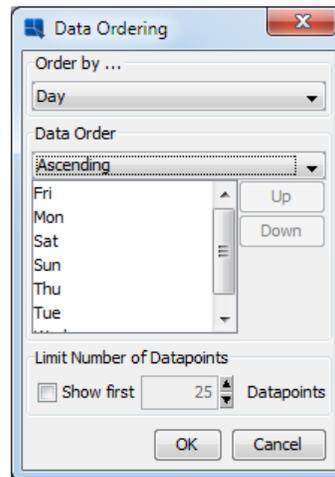
#### 4.2.4.8.2.2. Zooming In Chart Viewer

When deploying charts using Chart Viewer, end users can perform dynamic zooming. To perform a time-series zoom in the Chart Viewer, **Ctrl+Click** on a point on the chart and drag it to another point in the chart. This will automatically zoom in based on the lower and upper bounds selected using the mouse. The aggregation is performed according to the options that were set during design time. You can undo the zoom by **Ctrl+Right-Click**.

The scale interval is chosen automatically, depending on the data and chosen bounds (as long as minimum 2 data points can be shown). The scale interval can also be changed in the Chart Viewer by pressing **Alt+Z**. This will bring up a dialog allowing the user to change the zoom settings.

#### 4.2.4.8.3. Data Ordering

ChartDesigner allows you to change the order of the category and series elements. To modify the ordering, select Data → Ordering or click the  *Change Data Ordering* button on the toolbar. This will bring up the following dialog:

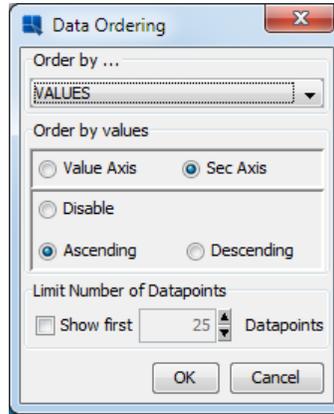


*Data Ordering Dialog*

There is an *Order By* list which contains category element, series element, and an option marked *VALUES*. You have the following options for the category and series elements:

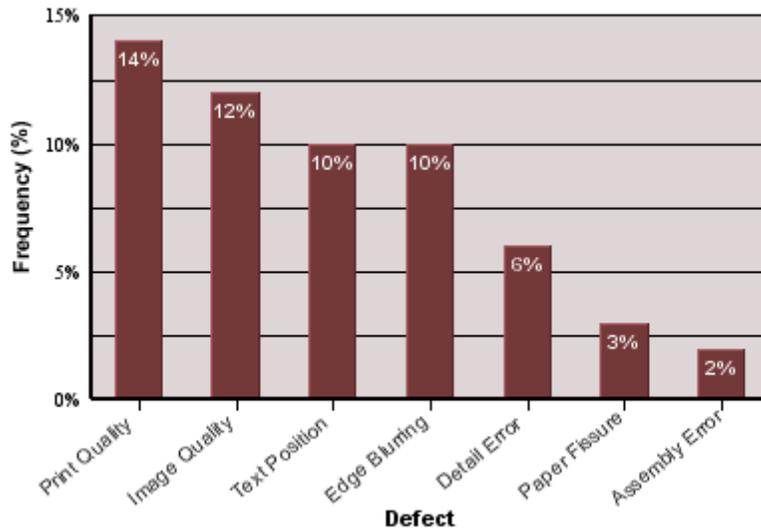
- DataSource Order:** This option turns the ordering off. The categories/series order will depend on the data source only and will not be altered by the ERES at all.
- Ascending:** This option will arrange the categories/series elements in ascending order. For example, if the category elements are strings, they will be arranged alphabetically.
- Descending:** This option will arrange the categories/series elements in descending order.
- Customize:** This option allows you to customize the categories/series order. To customize the order, select an item from the list of Categories/Series items and then move it upwards or downwards in the list by clicking on the *Up* or *Down* button (the buttons are inactive until you select the *Customize* option).

You can also sort the category elements based on their corresponding values. To do this, select the *VALUES* option in the data ordering dialog.



*Value Data Ordering Dialog*

If you choose the *VALUES* option, the entire dialog changes. From this dialog, you can specify to sort the category elements based on their corresponding values in the value, or secondary value axis. You can also specify whether to sort them in ascending or descending order. This type of ordering is called a *Pareto* chart and is often used in process control applications.



*Pareto Chart*

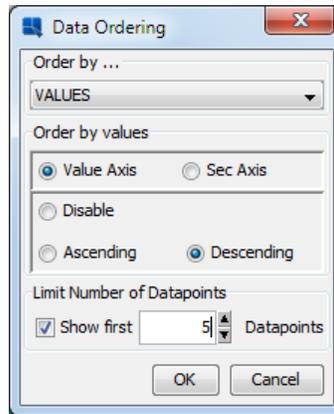
Please note that any sorting set will be re-applied if the chart is refreshed and/or if the data changes.

#### 4.2.4.8.3.1. Top/Bottom N Charts

Sometimes you want to plot only a few highest or lowest values. To do that, you can use the *Top/Bottom N* function.

To enable this feature, choose Data → Ordering, or click the  *Change Data Ordering* button on the toolbar.

If the chart doesn't have any data series, the *Ordering* dialog will show the *Limit Number Of DataPoints* option.



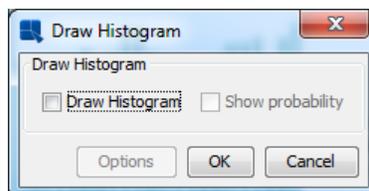
This option can only be enabled if you sort the chart by categories or values, or by the values in ascending or descending order. If you have such chart, you can enable this function by selecting the *Show first* option. Then you can specify the maximum number of items that will be shown in the chart. If the data source returns more items than you specify in this option, excessive items will not be shown in the chart as if they didn't exist.

#### 4.2.4.8.4. Histograms

Histogram is a useful analysis tool that allows you to track how often events occur and when and how a set of data falls into specific ranges. ChartDesigner allows you to plot histograms based on the category elements in a chart. You can plot histograms for all category data types except time-based data (date, time, or timestamp).

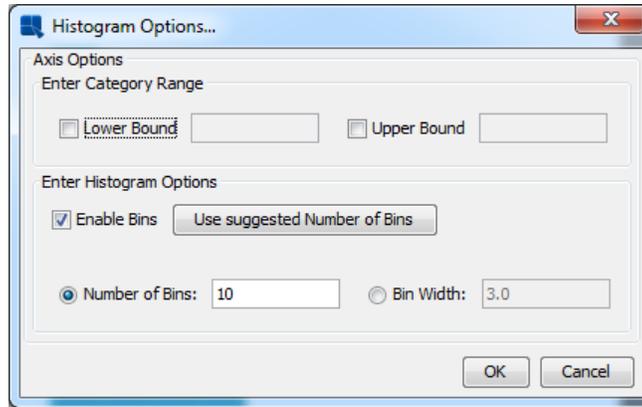
Histograms are calculated by counting data points or instances of each category element. For numeric categories, you can further specify upper and lower bounds, as well as the number of bins or bin width to create ranges for the frequency counts.

To create a histogram, you must start with a 2D column chart, bar chart, line chart or area chart. In the Data Mapping dialog, make sure *DataSeries* is set to *None* and the field you want to plot in the histogram is set in the Category axis. Once you click *Done* in the Data Mapping dialog and the chart is shown on the canvas, select *Format* → *Histogram Options*. A dialog will appear allowing you to select a histogram plot. By default, the histogram is displayed as frequency count. You can choose to change it to display probability by checking the *Show probability* check box.



*Select Histogram Dialog*

If the values in the Category axis is numeric, you will see that the *Options* button is enabled. When you click *Options* button, another dialog will appear, allowing you to specify options for the histogram plot.



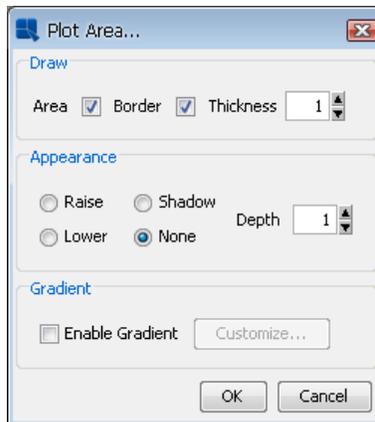
*Histogram Options Dialog*

From this dialog, you can set a lower or upper bound for that data being plotted. When you place bound restrictions, the histogram will not count data that falls outside of the range specified by the upper and lower bounds. If you select *Enable Bins*, you can specify the number of bins or set width of each bin. The default number of bins is 10. You can also click *Use suggested Number of Bins* if you wish to use a value calculated by the system. If *Enable Bins* is deselected, the frequency count will be performed on each Category value instead of a range of values for a bin.

If you enable scale, you can specify the number of bins or set width of each bin.

#### 4.2.4.8.5. Formatting Plot Area

The plot area is the plane on which the data points are drawn for two-dimensional charts. You can customize the appearance of the plot area by selecting Format → Plot Area. Assuming the current chart is a two-dimensional chart, the following dialog will appear.



*Plot Area Dialog*

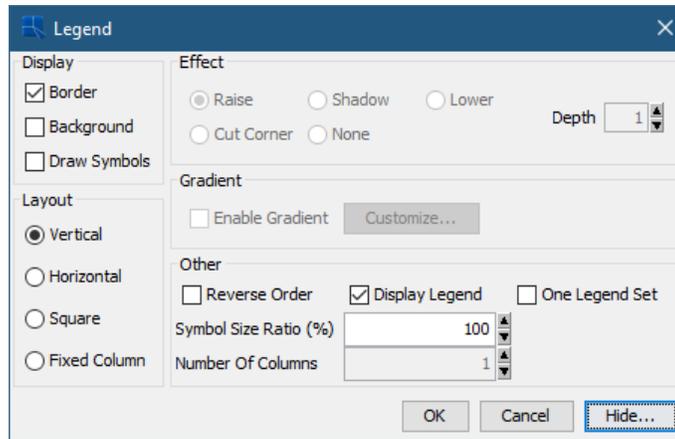
This dialog allows you to draw a border around the plot area, or fill it with a background color. If you fill the area, you can also specify certain 3D effects like raising, lowering or shadow.

On this dialog, you can also set up gradient background for the plot area. The gradient settings are the same as in the *Rendering options* described in the Section 4.2.4.1.3 - Format Menu.

#### 4.2.4.8.6. Formatting Chart Legend

You can control and modify the display of the chart legend either by selecting Format → Legend, or by clicking on

the  *Format Legend* button on the toolbar, or by selecting *Legend properties* from legend pop-up menu (which pops up when you right-click on the legend). This will bring up the following dialog, allowing you to customize the legend properties.



*Format Legend Dialog*

The dialog contains the following options:

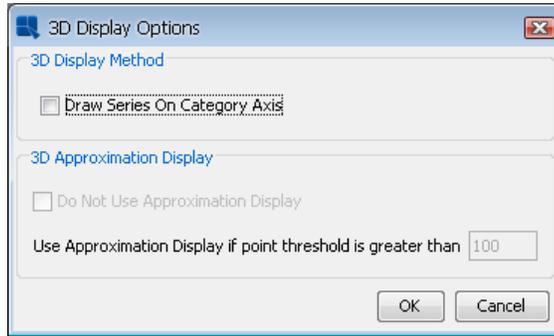
- Display:** These options allow you to turn on or off the legend border and background. This also allows you to display the point symbols instead of lines or blocks in the legend.
- Effect:** This allows you to add a 3D effect to the legend. You can raise it, lower it, or draw a shadow. In addition to the 3D effects, you can also display the legend with cut corners.
- Layout:** This allows you to change the legend from vertical, horizontal, square, or fixed column layout.
- Gradient:** Allows you to configure gradient for the legend background. The gradient settings are the same as in the *Rendering options* described in the Section 4.2.4.1.3 - Format Menu.
- Other:** This allows you to choose whether or not to display the legend, or to draw the legend in reverse order. You can set the fixed number of columns in legend in the *Number of columns* field. This field will be active only if you choose the *Fixed columns* layout option in the *Layout* section. You can set trend/control lines and chart data legend drawn as *One Legend Set*. You can also change the size of the symbols in the legend.

Additionally, you can remove specific category/series elements from the legend by clicking on the *Hide* button. This will bring up a list of the legend items, where you can select which elements you would like to hide.

#### 4.2.4.8.7. 3D Display Options

ChartDesigner renders three-dimensional charts in true 3D, allowing light source modification, panning, zooming, and rotation. However, 3D rendering can be very memory and CPU intensive. When charts have a lot of data points (like 3D scatter and surface charts), it's possible to run out of memory when generating the chart. To solve this problem, a rendering approximation feature is provided. Using this algorithm the chart is not rendered perfectly, but it's usually acceptable when a lot of points have to be shown.

By default, approximation is turned on at a threshold value of 100 points. This means that if a 3D chart has more than 100 data points, the approximation will be used. You can turn this feature off, or change the threshold value by selecting Format → 3D Display Options. This will bring up the following dialog.

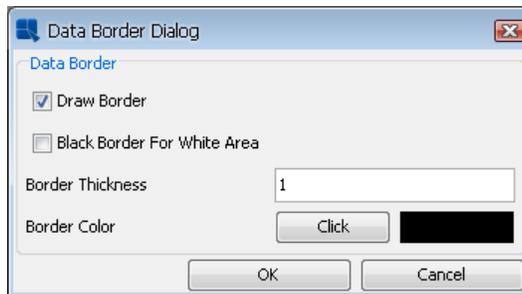


*3D Display Options Dialog*

The two options for 3D approximation allows you to turn on/off the approximation and set the threshold value. The other option in this dialog allows you to draw the series in-line (the same option is in navigation panel).

#### 4.2.4.8.8. Data Border

For column, bar, stack column, stack bar and HLCO charts, ChartDesigner allows you to configure a border around the columns. To set the border option, select Format → Data Border. This will bring up a dialog allowing you to set border options.



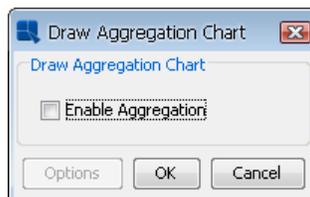
*Data Border Dialog*

The first option allows you to turn on/off the data border. The second option allows you to set a black border for any white areas in the chart. Please note that the border is black only if the first option is unchecked and will only appear around white areas in the chart. The third and fourth options allows you to set border thickness and border color. If you click on the *Click* button, a new dialog will appear allowing you to select or enter a new color.

#### 4.2.4.8.9. Aggregation

ChartDesigner allows you to aggregate data if there is more than one data point associated to a given category (and its series and/or stack, if a series and/or stack is present). This allows for a broader look at the data rather than just a single data point (out of many).

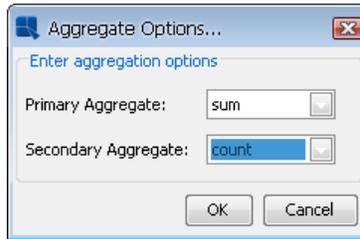
To aggregate the data, select Format → Aggregation Options. A dialog will appear allowing you to enable aggregation.



*Select Aggregation Dialog*

When you select *Enable Aggregation*, a second dialog box will appear, asking you which type of aggregation should be applied. You can choose from minimum, maximum, average, sum, count, first, last, sumsquare, variance, stddev,

and countdistinct for the aggregates. You can specify a primary aggregate (aggregate applied to the column mapped to the primary axis) as well a secondary aggregate (aggregate applied to the column mapped to the secondary axis), if a secondary axis exists.



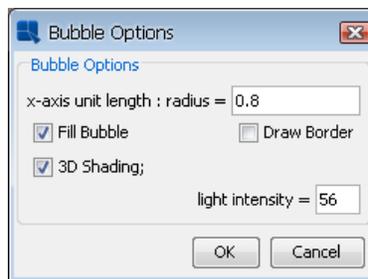
*Aggregate Options Dialog*

### 4.2.4.9. Chart-Specific Options

There are a number of formatting options that are unique to certain chart types. These options can be modified by selecting Format → Chart Options, or clicking the  *Chart Options* button on the toolbar. This will bring up a dialog that varies depending on the type of the current chart. Some chart types have no additional options.

#### 4.2.4.9.1. Bubble Charts

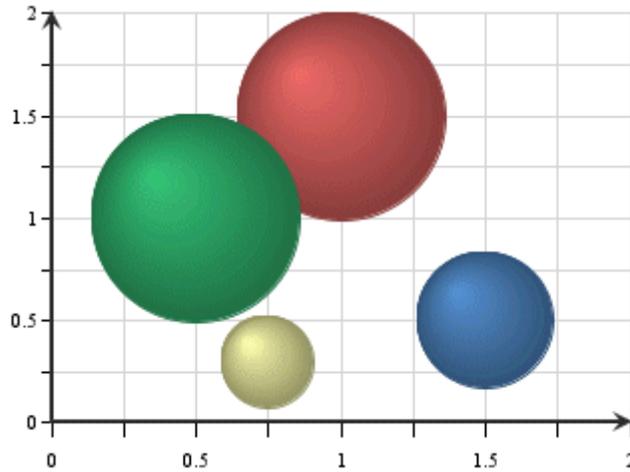
For bubble charts, the following dialog is displayed:



*Bubble Options Dialog*

The following options are available for bubble charts:

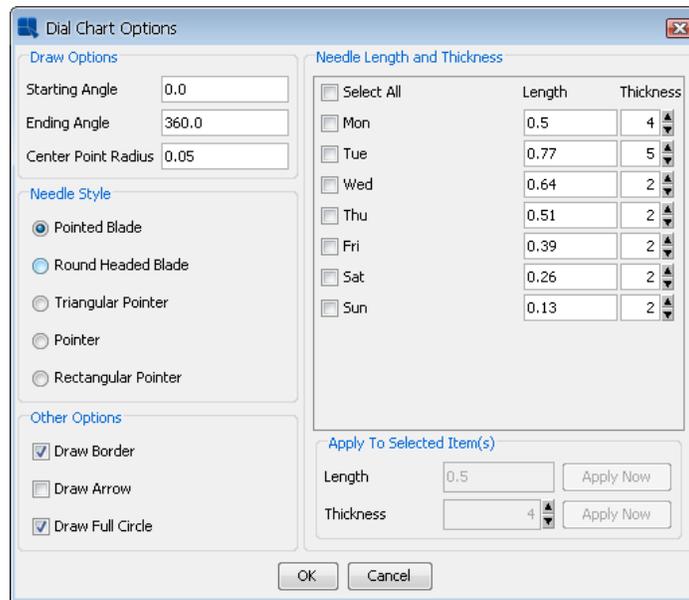
- |                                   |  |
|-----------------------------------|--|
| <b>x-axis unit length: radius</b> | This option specifies the ratio of X-axis unit length to the radius of the bubble.                                 |
| <b>Fill Bubble:</b>               | This option allows you to specify whether or not to fill the bubble area.  |
| <b>Draw Border:</b>               | This option allows you to specify whether or not to draw a border around the bubbles.                              |
| <b>3D Shading:</b>                | This option allows you to add 3D shading to the bubbles. You can also specify the light intensity for the shading. |



Bubble Chart with 3D Shading

#### 4.2.4.9.2. Dial Charts

For dial charts, the following dialog is displayed.



Dial Options Dialog

The following options are available for dial charts:

**Starting Angle:**

This option specifies the angle where the first axis label is to be set. This property also determines where the border and dial area will start if the *Draw Full Circle* option is unchecked. The angle is represented in degrees and is 0 by default. Assuming the dial chart is a clock face, 0 degrees is 12 o'clock.

**Ending Angle:**

This option specifies the angle where the last axis label should be set. This property also determines where the border and dial area will end if the *Draw Full Circle* option is unchecked. The angle is represented in degrees and is 360 by default. Hence by default the labels (and data points) encompass the entire circumference of the dial.

**Center Point Radius:**

This option specifies the radius for an inner circle, which starts from the center of the dial chart. The radius is specified as a ratio to the radius of the

dial plot. Hence, a value of 1 will make the inner circle encompass the entire dial. If the *Draw Full Circle* option is unchecked, only the portion of the center point that is within the starting and ending angles will be shown.



*Dial Chart with Center Point Radius*

**Needle Style:**

This option specifies the type of needle to draw.



The Pointed Blade is a smooth line with a thick base. It becomes slightly thinner as it extends outwards, but finishes with a very sharp pointed tip.



The Round Headed Blade is similar to the Pointed Blade except for a round tip.



The Triangular Pointer is sharp and resembles a very thin triangle.



The Pointer is a step ladder pointer with three segments.



The Rectangular Pointer (as shown above on the picture) is a simple straight needle.

Default needle is Pointed Blade.

**Draw Border:**

This option specifies whether or not to draw a border around the dial.

**Draw Arrow:**

This option specifies whether or not to draw arrowheads at the end of the dial hands.

**Draw Full Circle:**

This option specifies whether to draw the dial as a complete circle (360 degrees) or only draw the portion of the circle determined by the starting and ending angles.

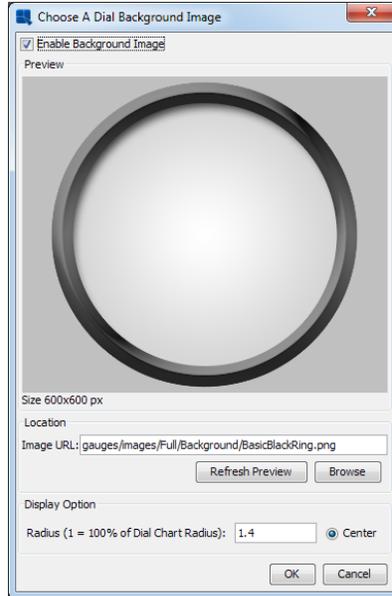
**Needle Length and Thickness:**

This option specifies the length and thickness of the needle. The needle length is measured from the center of the dial. The range is from 0 (center of the dial) to 1 (the end of the dial). The thickness determines the width of the needle, larger values results in a wider pointer. When creating a chart, the needle length is randomly generated. Default thickness is 2. Each category element is represented by a different needle. You can either change properties individually or change multiple categories at once. To change the property of each needle individually, directly change the values to the right of the category. To make changes to multiple needles, check each category or check the *Select All* option, set the properties in the lower right corner

and then click on the *Apply Now* button for each property changed. This will modify all checked categories to new values.

#### 4.2.4.9.2.1. Gauge Images

Dial charts have an additional option to display a foreground or background image for the dial plot area. To add a foreground or background image, select *Insert* → *Dial Foreground...* or *Insert* → *Dial Background...*. These options are only enabled for dial charts.

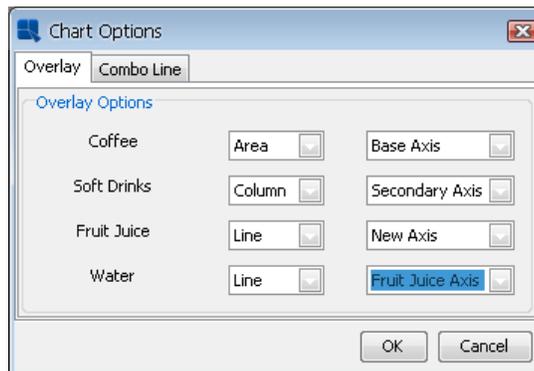


*Dial Chart Background Image Dialog*

Selecting an image works in the same way as the background image dialog, see Section 4.2.4.5.1.1 - Background Images. In the dial chart image dialog, there is also an option to specify the radius of the image. Specifying 1 for the radius will make the image the same width and height as the plot area. Increasing or decreasing this value will enlarge or shrink the images.

#### 4.2.4.9.3. Overlay Charts

For overlay charts, the following dialog is displayed:

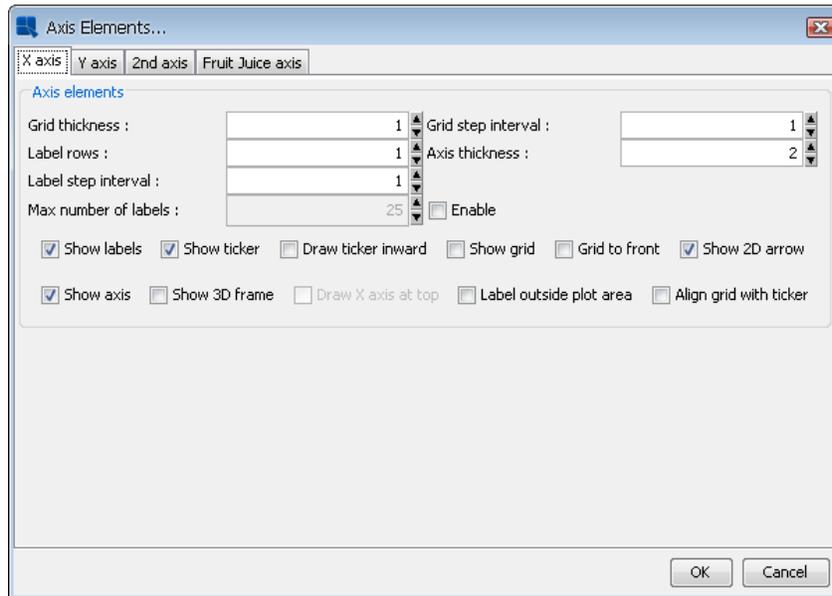


*Overlay Options Dialog*

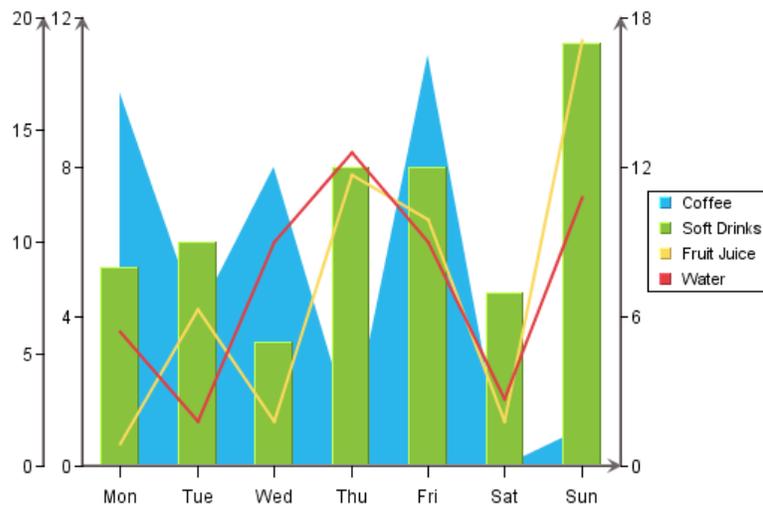
From this dialog, you can specify which chart type you would like to use for each element of the data series. Available chart types for the series elements in an overlay chart are column, area, and line. You can also choose not to display certain series elements.

From this dialog, you can also specify which axis you would like to use to plot a series element. You can place elements on the primary or secondary axes, or you can create new value axes for the series elements. To create a new value axis, select *New Axis* from the drop-down menu. Once you specify to use a new axis, a new option will be added to the drop-down menus for the other data series elements, allowing them to be drawn on the same axis

that you previously specified. Using a variety of axes allows you to precisely tune the scale that the different series elements use. Each of these axes will have its own tab in the Axis Elements window, where you can change the label step interval for each axis independently of the others.

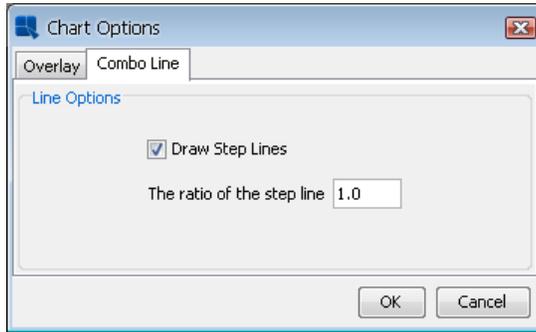


*Axis Options Dialog for Multiple Value Axes*



*Overlay Chart with Multiple Value Axes*

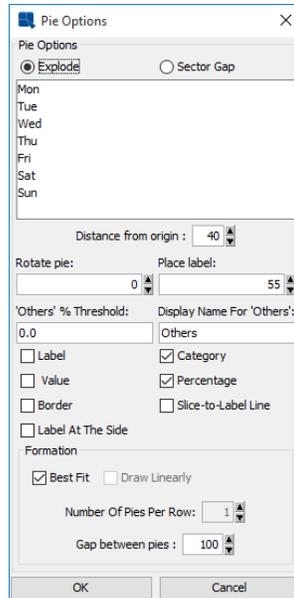
If one of the chart types used by the overlay chart is a line, you can specify whether or not to draw the line as a step line using the *Combo Line* tab. For more about step lines see Section 4.2.4.9.5 - Line Charts:



*Combo Line Options for Overlay Charts*

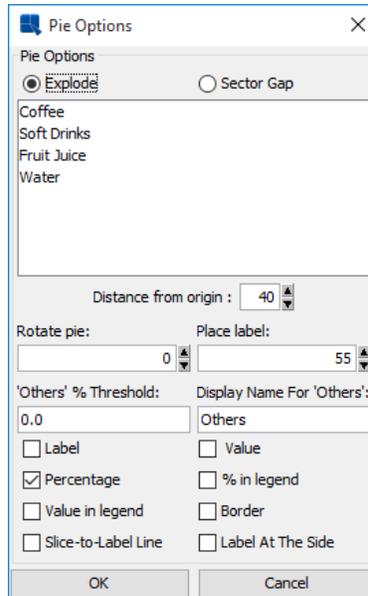
#### 4.2.4.9.4. Pie Charts

For pie charts, the following dialog is displayed. (Note that different options will appear/disappear depending on whether the chart has a series, and if it's a 2D or 3D chart.) Here are the options available for a 2D chart with series:



*Pie Options Dialog (With Series)*

Here are the options available for a 2D chart without series. Notice that the formation options are removed and the *% in legend* and *Value in legend* options are added:

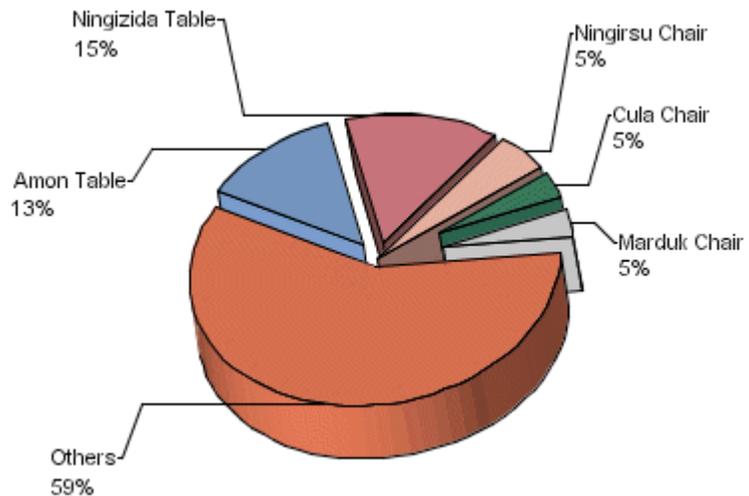


*Pie Options Dialog (Without Series)*

The following options are available for pie charts:

- Explode:** This option allows you to pick one or more category/series elements whose sections are to be drawn at a certain distance away from the center of the pie.
- Sector Gap:** This option allows you to pick one or more category/series elements whose sections are drawn at a certain distance away from the center of the pie and still maintain the same distance between pie slices and a circular boundary.
- Distance from origin:** This option allows you to specify how far the exploded/sector gap sections are to be drawn away from the center. This number, represented as a percentage of the radius, indicates the distance between the center and the tip of the pie slice to be exported.
- Rotate pie:** This option allows you to specify the number of degrees that the chart should be rotated in a clockwise direction. Available values are between 0 and 360
- Place label:** This option indicates the distance of the labels from the center of the pie. The position of an individual label can also be adjusted by dragging the text.
- “Others” % Threshold:** This feature is useful for pie charts that have a large number of small categories. Rather than draw a slice for each category, users can select a threshold value. Any category whose percentage of the value column is less than the threshold value will be lumped into an “Others” slice.
- Display Name for “Others”:** This option allows you to set the display name for the “Others” slice that is created for categories that fall below the supplied threshold value. This label will appear in the legend, and/or for the slice label.
- Label:** This option determines whether a category/series label should be drawn for each pie slice. By default, these only appear as legend items. Note that the label will not appear if the data for the slice is 0 or null.
- Value:** This option allows you to specify whether to display the actual value of each pie slice. Note that the value will not appear if the data for the slice is 0 or null.
- Category:** This option allows you to specify whether to display the category of each pie slice.

- Percentage:** This option allows you to display the percentage for each pie slice. The percentages are calculated by dividing the value of each section by the sum of all the values. Note that the percentage will not appear if the data for the slice is 0 or null.
- % in legend:** This option allows you to display the percentage represented by each slice in the pie in the legend. This can be a preferable presentation if the pie slices become too thin. This option is only available if the pie chart does not have a data series.
- Value in legend:** This option allows you to display the value represented by each slice in the pie in the legend. This can be a preferable presentation if the pie slices become too thin. This option is only available if the pie chart does not have a data series.
- Border:** This option specifies whether to draw a border around each pie slice. This option is only available for two-dimensional pie charts. For three-dimensional pies, you can use the border drawing option on the navigation panel. Note that the border will not appear if the data for the slice is 0 or null.
- Slice-to-Label Line:** This option will draw a line from any label(s) to its corresponding pie slice. Note that the slice-to-label line will not appear if the data for the slice is 0 or null.
- Label at the Side:** This option will place labels for the pie chart away from the plot around the outside of the chart. When used with the *Slice-to-Label Line* option, it gives users a way to display the pie labels for charts with many small categories without any text overlapping. Note that the label will not appear if the data for the slice is 0 or null.



*Pie Chart with Side Labels and Lines*

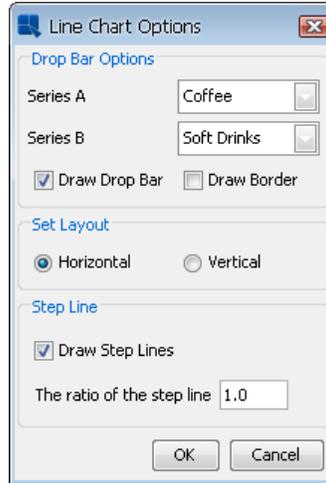
- Best Fit:** This option will arrange multiple pies in best configuration to fit the chart canvas. It's only available for pies with data series.
- Draw Linearly:** This option will arrange multiple pies in a straight horizontal line. It's only available for pies with data series.
- Number of Pies Per Row:** This option allows you to create a custom arrangement of multiple pies, by specifying the number of pies to draw in each row of the arrangement.

**Gap between pies:**

This option allows you to specify the gap between the multiple pies. The number is a multiple of the pie radius, so the gap will adjust with the size of the chart plot.

**4.2.4.9.5. Line Charts:**

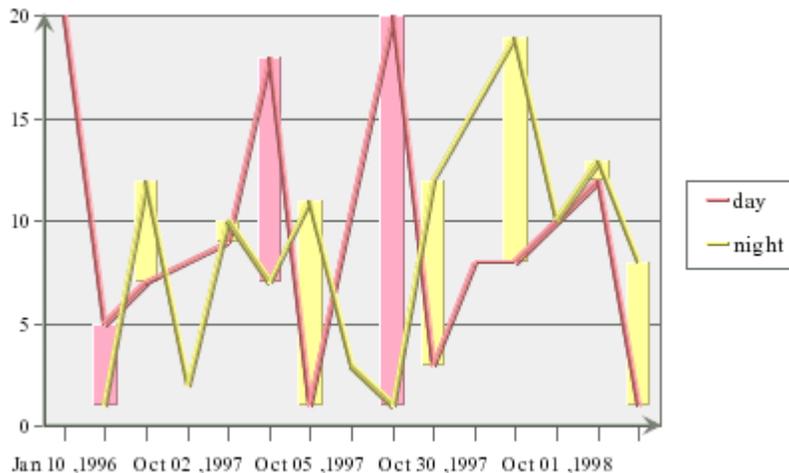
For two-dimensional line charts, one of two different dialogs will be displayed depending on whether the chart has a data series or not. If the chart has a data series then the following dialog is displayed.



*Line Options Dialog (with series)*

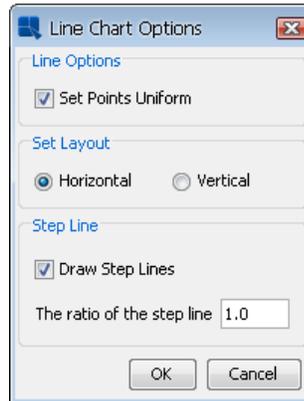
Line charts with data series have a specific option that allows you to draw drop bars between two series elements. The dialog options are as follows:

- Series A:** This option specifies the first series element for the drop bar.
- Series B:** This option specifies the second series element for the drop bar.
- Draw Drop Bar:** This option specifies whether or not to draw drop bars.
- Draw Border:** This option specifies whether or not to draw a border around drop bars.
- Set Layout:** This option specifies whether to draw a line chart in vertical or horizontal orientation.
- Step Line:** This option allows you to draw line chart as a step line. You can also specify the step line ratio to use.



*Line Chart with Drop Bars*

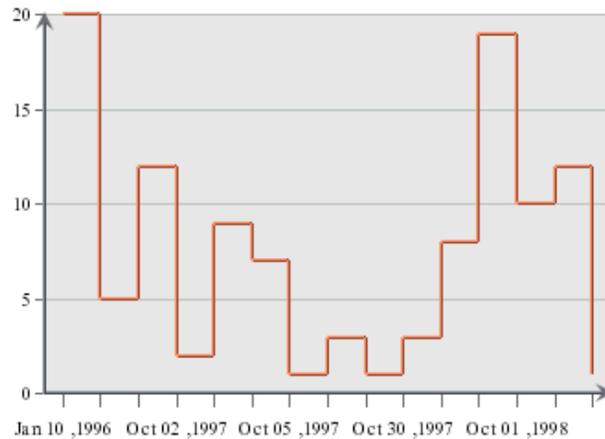
Note that the color of the drop bar will vary depending on which series has the higher value for a given point. If the line chart does not have a series, then the following dialog will appear.



*Line Options Dialog (without series)*

The dialog options are as follows:

- Set Points Uniform:** This option specifies whether the point shapes and colors are uniform or not. Un-checking this option allows you to set multiple colors and point shapes for the data points. The points can be customized in the line and point dialog.
- Set Layout:** This option specifies whether to draw the line chart in vertical or horizontal orientation.
- Step Line:** This option allows you to draw the line chart as a step line. You can also specify the step line ratio to use.



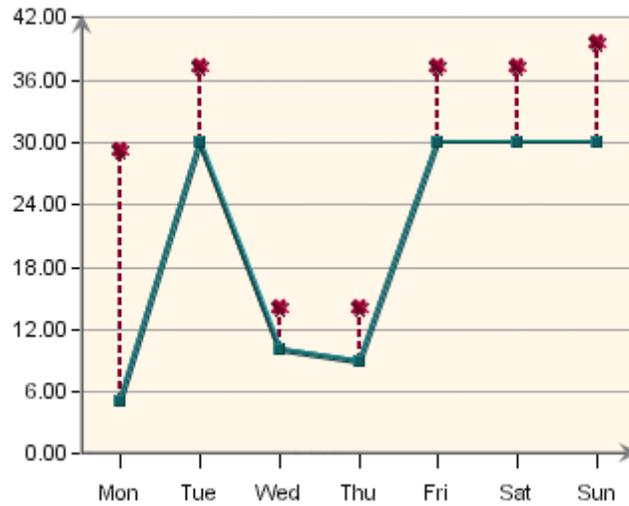
*Line Chart with Step Lines*

The step line ratio allows you to specify how far between points the horizontal portion of the step line should be drawn. A ratio of 1 draws the line horizontally to the next point on the chart and then draws vertically to that point, while a ratio of 0.5 draws the horizontal portion halfway between the two points. A ratio of 0 will result in the vertical portion of the line drawn first and then connect to the next point horizontally. Values for the ration are between 0 and 1.

There are no additional options for three-dimensional line charts.

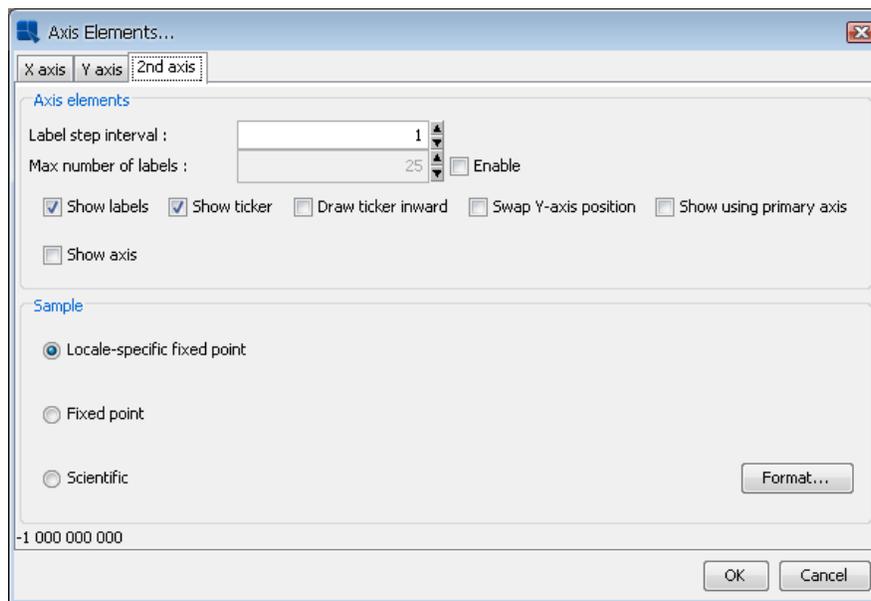
#### 4.2.4.9.5.1. Double Value Line Charts

ChartDesigner contains a special option for line charts that allows you to have two values shown for the same line. Here the secondary axis is used to plot the second value (as in a line-line combination) and then combined with the line on the primary axis.



*Double Value Line Chart*

To create a double value line chart, design a line-line combination chart (a line chart with primary and secondary values). Then select Format → Axis Elements. This will bring up the axis elements dialog.



*Axis Elements Dialog for Line-Line Combination Charts*

Under the *2nd Axis* tab, there is a checkbox marked *Show using primary axis*. Check this box and click on the *OK* button. Your chart will now be drawn as a double value line chart.

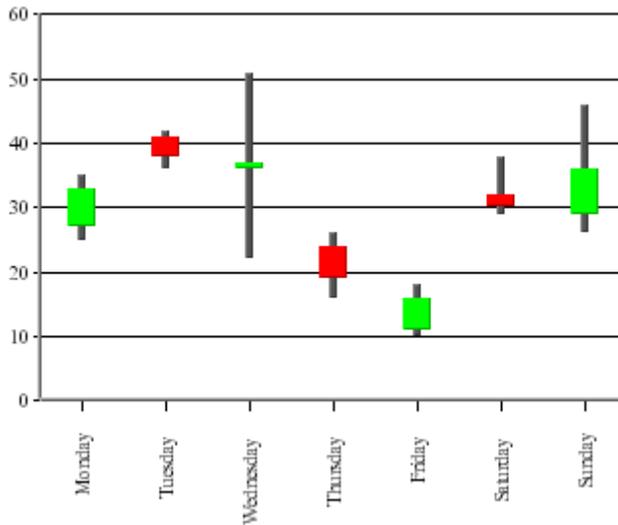
#### 4.2.4.9.6. HLCO Charts

For HLCO charts, the following dialog is displayed:



*HLCO Options Dialog*

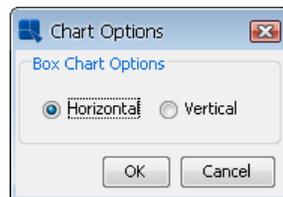
The *Show Hi-Low As Candle Stick* option will turn the HLCO chart into a candle representation. A candle HLCO chart blends high, low, close, and open data into a single object that resembles a candlestick.



*HLCO Candlestick Chart*

#### 4.2.4.9.7. Box Charts

For box charts, the following dialog is displayed:

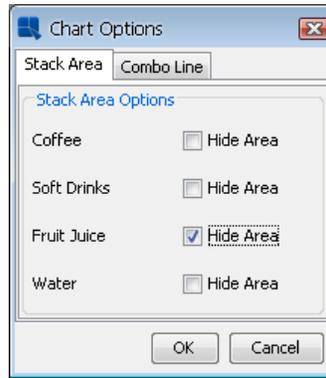


*Box Options Dialog*

This dialog allows you to specify whether to display the box chart in a horizontal or vertical orientation.

#### 4.2.4.9.8. Stack Area Charts

For stack area charts, the following dialog is displayed:

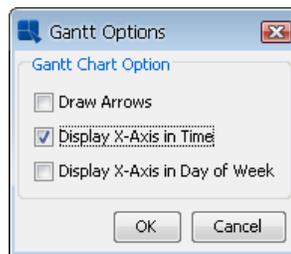


*Stack Area Options Dialog*

You can choose to hide any of the stacks in the chart by clicking on the corresponding check box. The *Combo Line* tab allows you to specify step lines if the chart is a line stack area combination. There are no additional options for three-dimensional stack area charts.

#### 4.2.4.9.9. Gantt Charts

For Gantt charts, the following dialog is displayed:

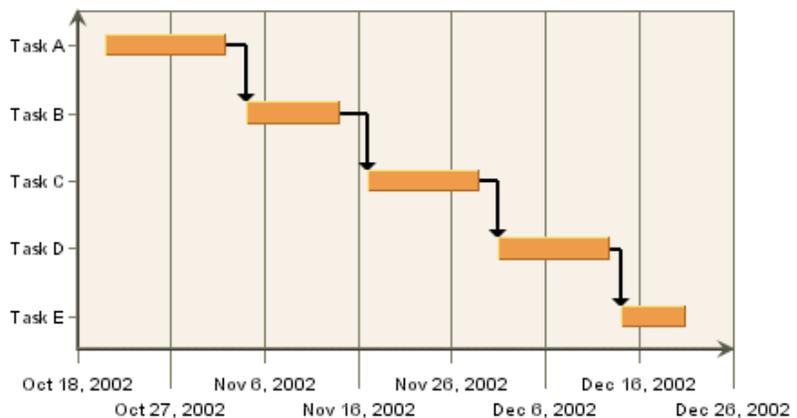


*Gantt Options Dialog*

The following options are available for Gantt charts:

#### **Draw Arrows:**

This option will draw connecting arrows between category elements of the Gantt chart. This allows you to illustrate a sequence between scheduled events. The arrows are drawn in the order the category elements appear in the data source.



*Gantt Chart with Arrows*

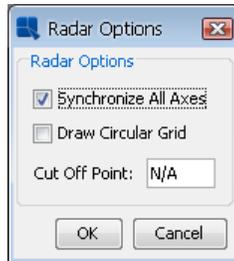
#### **Display X-Axis in Time:**

This option shows the ticker labels as time values instead of numeric values for the X-axis.

**Display X-Axis in Day of Week:** This option shows the ticker labels as days of the week with the date for each Sunday shown as well.

#### 4.2.4.9.10. Radar Charts

For radar charts, the following dialog is displayed:



*Radar Options Dialog*

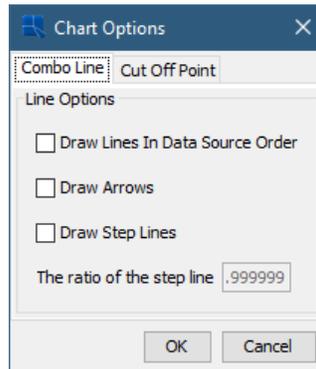
By default, the scale is same for all axes in the radar chart. Unchecking the *Synchronize All Axes* option will allow each axis in the radar chart to be scaled independently. You can select to use auto-scaling for each axis, or you can set the scales manually by invoking the axis scale dialog.

The second option allows you to set how the grid is drawn for the radar chart. By default, if the grid is enabled, it is drawn in straight lines that connect the tickers on each axis. Enabling the *Draw Circular Grid* option will draw the grid in a circle, similar to the polar chart grid.

The third option allows you to specify a cut-off point for the data points (areas) in the radar chart. You can enter a maximum value that should be shown in the chart. The areas bounded by the data points will not be drawn beyond the specified cut-off point.

#### 4.2.4.9.11. Scatter Charts

For scatter charts, the following dialog is displayed:



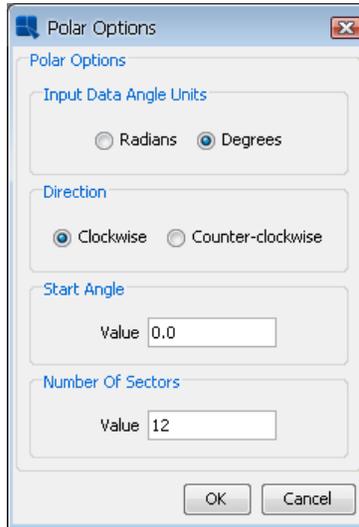
*Scatter Options Dialog*

The *Combo Line* tab allows you to set Draw Lines In Data Source Order, Draw Arrows on lines or Draw the connecting lines as a Step Lines, as well as specify the step lines ratio.

The *Cut Off Point* tab allows you to specify a maximum value for the Y point of the scatter coordinates. Any coordinates that fall beyond this threshold will not be plotted. Connecting lines will draw up to the edge of the threshold and continue to the next data point.

#### 4.2.4.9.12. Polar Charts

For polar charts, the following dialog is displayed:



*Polar Options Dialog*

**Scale:** This option allows you to specify whether the input data for the angle ( $\theta$ ) portion of the data points is in radians or degrees. The chart will always display angles from 0 to 360. If the input data is in radians, it will be displayed as degrees.

**Direction** This option allows you to specify whether the circular plot should be drawn clockwise or counter-clockwise.

**Start Angle:** By default, the top of the polar chart plot is 0 degrees. This option allows you to specify a different angle for the top of the plot. The argument for this angle is supplied in degrees or radians, depending on the scale you have chosen.

**Number of Sectors:** This option allows you to select number of sectors you'd like to show in the chart. Sectors are created by drawing additional polar axis lines at specified angle intervals. By default, four sectors are shown.

#### 4.2.4.9.13. Column Charts with Series

For column charts with data series, the following dialog is displayed:



*Column Options Dialog*

Normally, when column charts have data series, each series has its own color that is applied for every category in the chart. If you want to assign different colors to the columns in the chart regardless of the series, you can enable the *Unique Color Column* option in this dialog. When it's turned on, you can set color for each column in the chart individually.

#### 4.2.4.9.14. Column/Bar Charts without Series

For column/bar charts without data series, the following dialog is displayed:

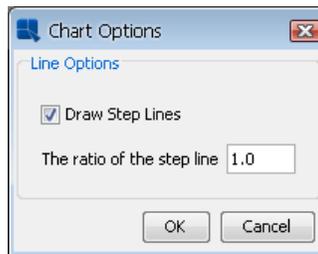


*Column/Bar Options Dialog*

Normally, when column/bar charts don't have data series, all categories in the chart have single color. If you want to assign different colors to the categories in the chart, uncheck the *Single Color For All Categories* option in this dialog.

#### 4.2.4.9.15. Two-Dimensional Line Combination Charts

For any other two-dimensional line combination chart, the following dialog is displayed:

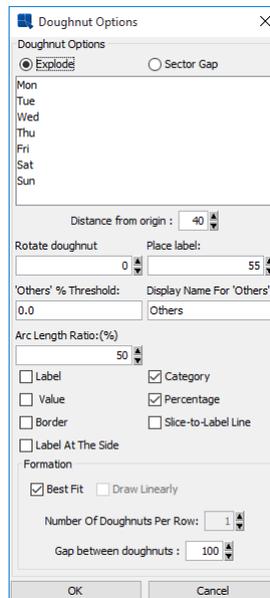


*Line Combination Chart Options*

This dialog allows you to specify whether to draw the combo line as a step line, as well as specify the step line ratio.

#### 4.2.4.9.16. Doughnut Charts

The chart options for a doughnut chart are almost exactly the same as the options for a pie chart. You can refer to the options under Section 4.2.4.9.4 - Pie Charts for more details.



*Doughnut Chart Options*

The only option unique to doughnut charts is the Arc Length Ratio (%) which specifies the size of the hole at the center of the chart. A higher number results in a smaller hole.

## 4.2.5. Chart Drill-Down

Like with reports, it is also possible to add layers of drill-downs to charts. All of the drill-downs added will be applied to both data points on the plot area and their respective fields in the legend. Using chart drill-down, you can create a top-level chart that displays summarized data and allows users to click through specific data points to see underlying details.

The charting engine in ERES contains several different built-in drill-down mechanisms that allows you to create only one chart template for each level of drill-down. All of the drill-down options are available from the Drill-Down menu in Chart Designer.

It is important to note that you can only use drill-downs in charts when the chart is deployed independently from the report (in the Chart Viewer, its own servlet, etc). If a chart template that includes drill-down is placed in a report, only the top-level chart will display. Users will not be able to click through to the lower-level charts.

### 4.2.5.1. Data Drill-Down

Data drill-down allows you to group and display information that is based on a single data source. The advantage to this form of drill-down is that it works with any data source. However, it does not allow you to display loosely related information because all levels of drill-down will share the same value column from the input data.

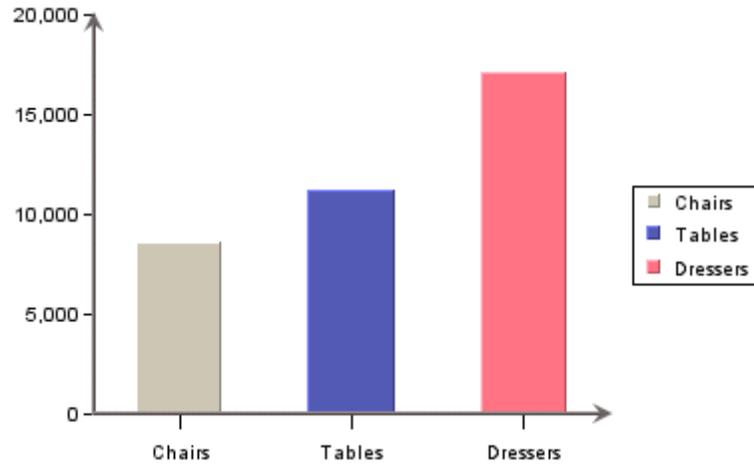
For example, assume you have the following table as the chart data:

Category	Product	Sales
Chairs	Elm Arm Chair	\$8,216
Chairs	Pine Side Chair	\$7,611
Chairs	Redwood Arm Chair	\$8,625
Tables	Elm Round Table	\$10,241
Tables	Pine Oval Table	\$9,663
Tables	Oak Oval Table	\$11,261
Dressers	Oak Single Dresser	\$16,442
Dressers	Elm Double Dresser	\$17,148

Using this data, you could create a top-level chart that displays total sales for each distinct product category and then create a lower-level chart that shows individual sales for each product in a category. The number of drill-down levels available depends on the number of groupings that are present in the input data.

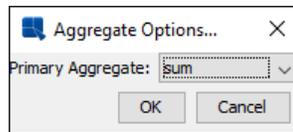
#### 4.2.5.1.1. Adding Data Drill-Down

To add layers of data drill-down to a chart, you must first create a top-level chart. In the above example, we would create a chart with “Category” mapped to the category axis and “Sales” mapped to the value axis. In a column chart, it would look like this:



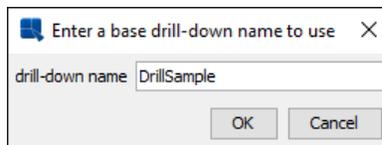
*Top-Level Chart*

To add a layer of drill-down, select Drill-Down → Add. This will bring up a dialog prompting you to specify the aggregation you want to use for the value axis. For example, selecting sum will display the total for each category in the top-level chart. Available aggregate functions are minimum, maximum, average, sum, count, first, last, sumsquare, variance, stddev, and countdistinct.



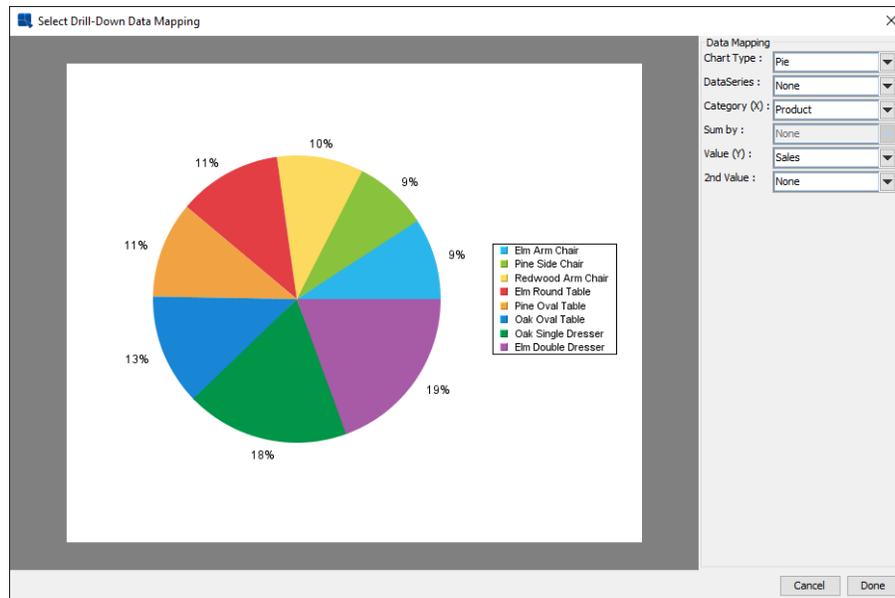
*Aggregation Dialog*

After you select the aggregation you want to use, click on the *OK* button. A new dialog will appear prompting you to specify a name for the drill-down chart. This name will be assigned to the chart templates that are created by the drill-down process. Note that all sub-level drill-down templates are saved in the `/drilltemplates/` directory.



*Drill-Down Name Dialog*

Once you have specified a name you want to use, click on the *OK* button. You will then be prompted to specify the chart type and data mapping for the sub-level chart.



*Data Drill-Down Mapping Dialog*

The options in the dialog are similar to those for normal data mapping. The major difference is that the first option allows you to select a chart type. Available chart types for data drill-down are column, bar, line, stack column, stack bar, pie, area, doughnut, overlay, radar, and dial. The data mapping options will change depending on the type of chart that you select.

Once you have finished specifying the mapping options, click on the *Done* button and you will go back to the Chart Designer where you can customize and modify your sub-level chart. You can add additional layers of drill-down by selecting Drill-Down → Add again.

You can navigate to a particular drill-down chart by selecting Drill-Down → Previous or Drill-Down → Next or by double clicking a data point. The *Previous* selection takes you up a level, while the *Next* selection goes down a level using the left-most category as the data to be drilled on. You can also go from any drill-down chart to the top-level chart by selecting Drill-Down → Go To Top Level. You can customize (i.e. assign colors, add title, axis labels, background image, ...etc.) the drill-down chart in the same way as the top-level chart. Everytime you change and leave a level, you will be prompted to save the chart. Make sure that you answer *yes* if you want the attributes of the drill-down level to be saved, otherwise you will lose all the changes.

You can insert a drill-down chart between two other drill-down charts by going to the higher level drill-down chart and selecting Drill-Down → Add. The drill-down chart that you create will be inserted between the two previous drill-down charts.

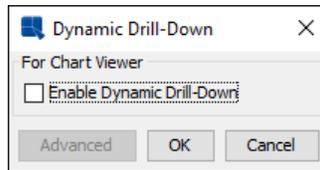
You can remove a level of the drill-down by navigating to that level and selecting Drill-Down → Remove This. The entire drill-down chart can also be deleted by selecting Drill-Down → Remove All. Selecting Drill-Down → Previous brings the drill-down chart to a higher level, which is the same as right clicking and selecting *Back* from the pop-up menu. The *Next* selection brings the drill-down chart to a lower level, which has the same function as double clicking on an item in the chart.

Once you have finished creating and editing all the levels of the drill-down chart, you can save it by navigating to the top-level chart and selecting File → Save or File → Save as.

#### 4.2.5.2. Dynamic Data Drill-Down

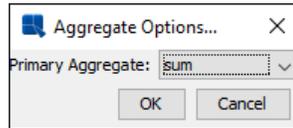
Usually, mapping and drill options for data drill-down are fixed during design time. Dynamic drill-down is an additional option that allows you to select the mapping. The only thing specified during design time is the top-level chart and aggregation.

To create a chart with dynamic drill-down, first create a chart you want to use as the top-level chart (for example, you can create the same top-level chart as before) a then select Drill-Down → Dynamic. A dialog will appear allowing you to enable dynamic drill-down.



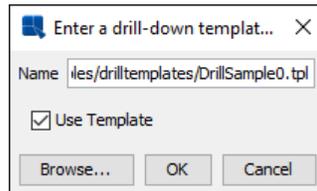
*Enable Dynamic Drill-Down Dialog*

Check the *Enable Dynamic Drill-Down* box and a new dialog will pop-up prompting you to select the aggregation.



*Aggregation Dialog*

Options for this dialog are the same as for regular data drill-down. Once you specify the aggregate you want to use, click on the *OK* button. Another dialog will pop-up, allowing you to specify a template you want to use for the sub-level charts.



*Select Dynamic Drill-Down Template Dialog*

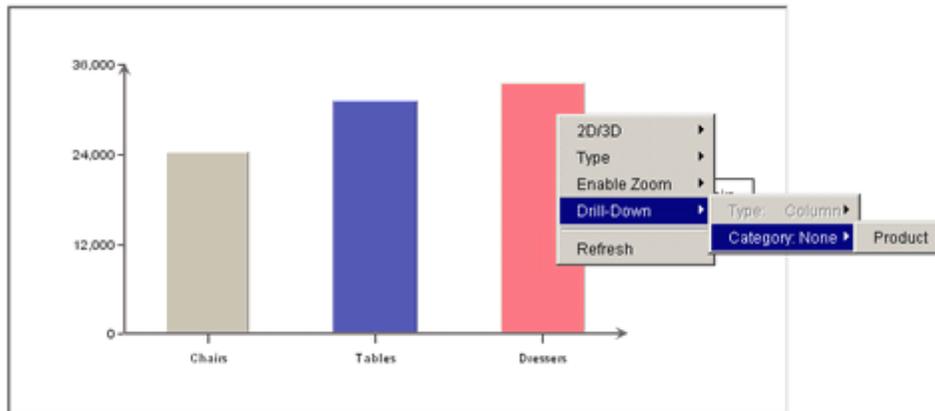
If you do not select to use a template, the sub-level charts will be generated using default appearance properties. After you finish selecting these options, your chart will change to display aggregated data on the value axis. You can change the option by selecting *Drill-Down* → *Dynamic* again and then clicking on the *Advanced* button.

Dynamic drill-down charts can only be viewed in the Chart Viewer JNLP (More about Applets in JNLP: Section 1.3.4 - Run Applets As JNLP). There you can configure setting of the next drill-down chart by right clicking on the desktop area which will bring up a pop-up menu. If the dynamic drill-down is enabled and the chart still has some unused fields (columns) to plot, the item *Drill-Down* will be displayed in the pop-up menu. Upon selecting *Drill-Down* in the pop-up menu, you can view the current setting for next drill-down. If *Category* is **None**, it means you have not configured the next drill-down chart yet. To configure the drill-down chart for next level, you must select *Category* (*Type*, *Series* and *SumBy* can be selected once *Category* has been set). After you create a drill-down chart, you can navigate to different levels in the Chart Viewer by either left clicking on a data point if you want to go down one level, or by right clicking and selecting *Back* from the pop-up menu if you want to go up one level, or by repeating the previous step to create another drill-down chart for next level.



**Tip**

After traversing to a lower-level of drill-down, right clicking on a data point will navigate you back to the top-level chart. To bring up the pop-up menu, right click on the chart canvas away from the chart data points.



*Dynamic Data Drill-Down in Chart Viewer*

### 4.2.5.3. Parameter Drill-Down

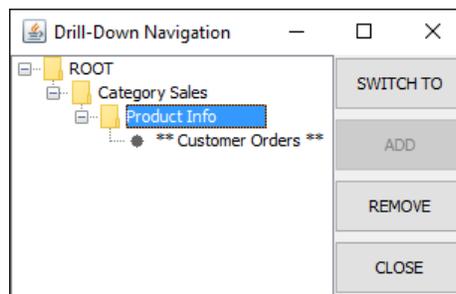
The third type of a drill-down you can use for charts is a parameter drill-down. Parameter drill-down is the most flexible implementation because you can relate the data between drill-down levels in any way you like. Specifically, you do not need to use the same value element for each level. Instead, parameter drill-down uses parameterized query feature to relate various chart levels and since it uses queries, the data source for the sub-level charts must be a database or parameterized class file.

For example, using our previous scenario, rather than always looking at sales, you want your top level chart to look at aggregated sales by category (same as before), then on the next level, you want to look at sales volume for each product, and from there, you want to look at inventory levels for each product by region. This can be accomplished with parameter drill-down.

With parameter drill-down, the category value of an element you click to drill on will be passed to the sub-level chart as the parameter value. Hence, if you click on the “Chairs” column, the value of “Chairs” would be passed to the query. Therefore, anything in the database that could be filtered by category name could be retrieved.

#### 4.2.5.3.1. Adding Parameter Drill-Down

To add and edit various layers of a parameter drill-down, select Drill-Down → Parameter Drill-Down. This will bring up a navigation window that shows various levels of drill-down.

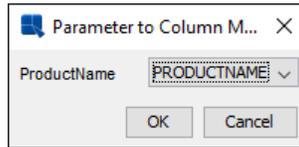


*Parameter Drill-Down Navigation Window*

Left-hand side of the navigation window displays the hierarchy of drill-down levels. The “ROOT” node is the top-level chart. The level you are currently editing is marked with \*\*. To edit a different level, select it and click the *SWITCH TO* button on the right-hand side. The chart will then open in the Designer. Levels of drill-down can also be removed by selecting the node in the Navigation window and then clicking on the *REMOVE* button.

To add a new level of drill-down, select the chart under which you would like to add the layer (if you have only the top-level chart, then only the “ROOT” node will be visible), and click *ADD*. You will then be prompted to create a new chart or to use an existing chart for the drill-down layer. You can use any existing chart; however, any chart for a drill-down layer must have a parameterized query as the data source. If you select to create a new chart, the Data Source Manager will reopen, allowing you to select a data source for the chart and to continue through the Chart Wizard steps.

The next step is to map the category and/or series columns from the top-level chart to the query parameter(s) in the sub-level chart. You will be prompted to do this when you select an existing chart for the drill-down layer or when you select the data source for a new chart for the drill-down layer. In either case, a dialog will appear prompting you to map the fields.



*Parameter Mapping Window*

Available options in the drop-down menus are based on data type. For example, if your parameterized query has string as the parameter, only fields containing string data can be mapped. Hence, it is important to consider what type of data will be passed from the top-level chart to the lower-level charts because if your category or series are not of the correct data type or if you do not have enough fields to pass to the lower-levels, the drill-down will not work correctly.

Once you correctly specify the parameter mapping, you will be prompted to specify a display name for the drill-down level. After you select a name, the sub-level chart will appear in the Designer, allowing you to customize it. You can navigate through the layers of drill-down using the Navigation window, or like data drill-down, you can double click on a data point to go down a level and right click and select *Back* from the pop-up menu to go up a level.

## 4.2.6. Saving & Exporting Charts

After you finish designing a chart, you can save the definition as a chart or as a chart template, and provide XML definitions for both. You can also generate a number of static image exports of the chart or create a JSP/JNLP page with the Chart Viewer JNLP (More about Applets in JNLP: Section 1.3.4 - Run Applets As JNLP).

### 4.2.6.1. Saving Charts with Report Data

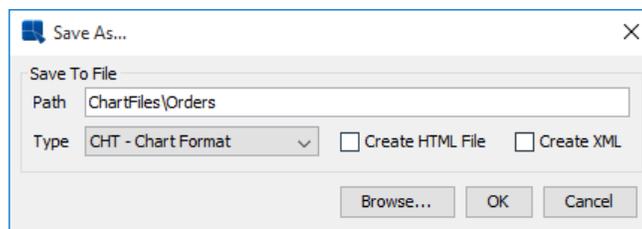
When you design a chart using report data, the chart cannot be run/viewed any place except within the report. Because of this, most of the file operations are disabled in the Chart Designer. When you opt to save a chart it will be automatically saved with the same name as the report in the `/chart/` directory of the installation. For example, if your report is named *SalesReport*, any charts in the report that use report data would be saved in the following convention `SalesReport_0.tpl`, `SalesReport_1.tpl`, etc

Although most of the file options are disabled when you use report data for the chart, you can apply a template to carry over the appearance properties of another chart. Chart templates are covered in Section 4.2.6.2.1 - Working with Templates.

### 4.2.6.2. Saving Charts without Report Data

If your chart uses an independent data source from the report, it can be saved in any place with any filename. To

save the current chart, select `File` → `Save` or `File` → `Save As`, or click on the  *Save* button on the toolbar. Assuming you have not saved the chart before, or selected `File` → `Save As`, the following dialog will appear:



*Save As Dialog*

The first option allows you to specify a name and file path for the saved chart. You can browse to the appropriate file path by clicking on the *Browse* button at the bottom of the dialog. The second option allows you to specify which format you would like to use when saving the chart. As detailed in Section 4.2.2 - Charting Basics, there are two principle ways in which chart definitions can be saved.

<b>Chart format:</b>	Chart files save the chart in a binary file called <code>filename.cht</code> . A chart file stores both the definitions of the chart (type, dimension, etc), as well as the data that was used to create the chart.
<b>Template format:</b>	Template files save the chart in a binary file called <code>filename.tpl</code> . A template file only stores chart definitions. It doesn't store any chart data. Hence, any time a template file is opened, it will try to connect to the original data source to retrieve the data.
<b>PAC format:</b>	PAC files save the chart in such a way that makes them ready for deployment. A <code>.PAC</code> file takes the chart and all the supplementary files associated with it, such as background images, dynamic drill-downs, or parametric drill-downs, and places them in a single binary file.

Once you select a format you want to use, click on the *OK* button and the chart will be saved.



### Note

When your report is run, ERES will try to load the chart along a relative path from where ever the chart is saved. If the chart cannot be found, it will not display. However, you can specify a different path for the chart using options available in the API.

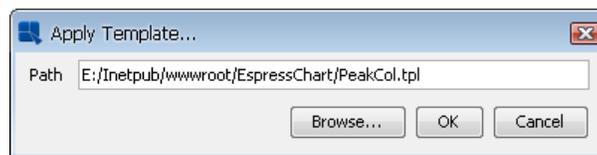
#### 4.2.6.2.1. Working with Templates

Chart templates are a special format in which chart definitions can be saved. Unlike chart files (`.cht` format) which saves both the chart definitions as well as the data for the chart, templates only save the definitions (i.e. the chart attributes and the layout of the individual components), and the data source information. This means that templates store the location/connection information for the data source that was used to create the chart, but they store none of the actual data in the file. (Templates do keep 10 records of back-up data, allowing them to be opened when the data source is not present).

Template files can be used in two ways. First, it can be opened, viewed, or exported, allowing you to see a chart with fresh data. This way data for the chart are retrieved based on the data source specified in the file. If the original data source is not accessible, this method cannot be used. The second way in which templates can be used is to apply its attributes to other charts. In this scenario, the chart to which the template is applied will inherit appearance properties of the template (including colors, fonts, size of chart components, position of legends, etc). This feature allows you to produce a consistent look and feel among charts.

The second approach is very useful if you are using the API to generate charts programmatically, using JDBC code or some other data source. You can then use a pre-defined template to control the look and feel of the generated chart without having to define the appearance properties in code or relying on the default chart properties. For more on applying templates in the API, please see Section 8.2.5.2 - Applying a Chart Template.

You can apply a template in Chart Designer by selecting *File* → *Apply Template*. This will bring up a dialog prompting you to specify the template file you want to use.



*Apply Template Dialog*

The dialog allows you to specify the template file and its location. You can browse to a file by clicking on the *Browse* button.

Note that when the chart and the template being applied are different sizes (i.e. chart canvas size), the resulting chart may not display correctly. This is because the text size will not change between template and chart to which it is applied. While the other components will adjust to the size of the new canvas, the font will not. To keep a consistent look, the size of your template should be close to the size of the chart to which it will be applied.

Hyperlinks, floating lines, floating text, and axis scales defined in the template files carry over. You may have to redefine them in the chart, if necessary. Chart type and dimension are not modified by the template. For example, a

three-dimensional chart will not be changed to a two-dimensional chart if the template is a two-dimensional chart. Likewise, a pie chart remains a pie chart when a template of a bar chart is applied. Although the chart type does not change when you apply a template, some appearance properties will not translate very well from a template with another chart type. For best results, try to apply templates of the same type and dimension to a chart.

#### 4.2.6.2.2. Saving XML Templates

In addition to the two binary chart representations, you can also save the chart definitions in XML format. This option allows you to have a text-based chart template that can be used to modify chart properties outside of Chart Designer or the API.

To create an XML file when saving the chart, check the *Create XML File* box in the save as dialog when saving the chart. This will create an XML file for the chart. Note that the XML file is a representation of the `.tpl` format and not the `.cht` format.

#### 4.2.6.2.3. Creating a Viewer Page

The other option to specify is whether you would like to create an HTML page to view the chart. The HTML page will redirect to a JSP which contains the Chart Viewer JNLP, which allows end users to view and manipulate the chart. The features of the Chart Viewer are discussed in detail in Section 7.8 - Chart Viewer

To create an HTML page when saving the chart, check the *Create HTML File* box in the save as dialog when saving the chart. This will create the HTML page with the same name as the chart file under the `html//` directory of your ERES installation. Before the file is written, you will be prompted to select which version of the Viewer that you would like to use.



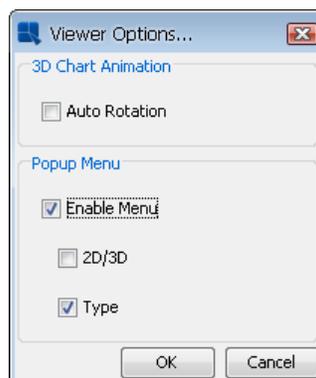
*Select Viewer Dialog*

ERES supports AWT and swing versions of Viewer. Note that if you use Viewer, the client will need to have the Java plug-in.

Once you have created the HTML page, you can point your browser to the page to view the chart. Note that in order for the chart to appear you must load the HTML page via http protocol. If you attempt to load it over a file protocol (i.e. browsing to the file and opening it) this will not work.

#### 4.2.6.2.3.1. Viewer Options

There are a number of Chart Viewer options that you can configure from within Chart Designer. These are properties that are saved with the chart, which will appear when the chart is displayed in the viewer. To modify the viewer options, select Format → Viewer Options. This will bring up the following dialog.

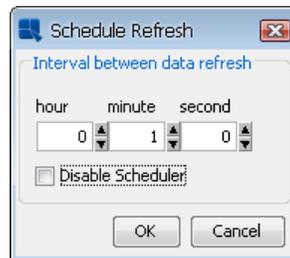


*Viewer Options Dialog*

The following options are available:

- Auto Rotation:** This will enable automatic rotation for three-dimensional charts. The charts will begin to rotate when the applet is loaded. The rotation can be stopped using the appropriate button on the navigation panel.
- Enable Menu:** This allows you to turn on or off the pop-up menu when users are viewing the chart in Chart Viewer.
- 2D/3D.** This allows you to turn on or off the dimension toggle in the pop-up menu. If this is turned off, users will not be able to change the chart dimension.
- Type:** This allows you to turn on or off the type sub-menu in the pop-up menu. If this is turned off, users will not be able to change the chart type.

For .cht files, you can also set a scheduled refresh. This will allow the chart to periodically refresh the data when the chart is being viewed in Chart Viewer. To schedule a refresh rate, select Data → Schedule refresh. This will bring up a dialog, allowing you to set the schedule options.



*Schedule Refresh Dialog*

From this dialog you can set the hour, minute, and second for the refresh interval. If you check the *Disable Scheduler* box, the Scheduler will be turned off.

### 4.2.6.3. Exporting Charts

From the Chart Designer, you can generate a number of static image exports for your chart. To export the current

chart, select File → Export, or click the  *Export* button on the toolbar. This will bring up a dialog allowing you to specify options for the generated file.



*Export Chart Dialog*

The first option allows you to specify a name and the file path for the generated file. You can browse to the appropriate file path by clicking the *Browse* button at the bottom of the dialog. The second option allows you to select what type of export you would like. The following options are available:

- GIF** ERES can generate GIF images. GIF has 256 color limit which keeps image file sizes small.
- JPEG** JPEG is another popular image format. It is a higher resolution image format than GIF and it is not patent protected. When generating a JPEG file, you can specify quality/compression of the file. The higher the quality, the larger the file.

If you select JPEG as the export format, after clicking *OK* you will be prompted to specify the quality. The higher quality image you select, the larger the generated file size. It is recommended that with the JPEG export, you use specify a high-quality image, as the low-quality results will most likely be undesirable.

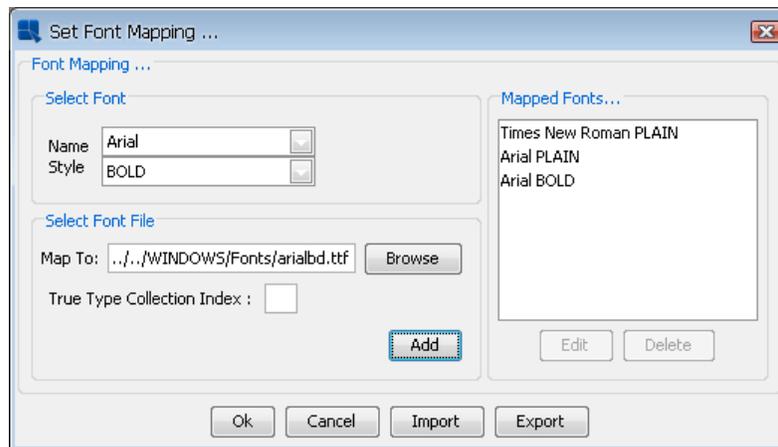
- PNG** PNG is an image format that is less popular, but can be displayed in most browsers. It is a high-quality image with a smaller file size than JPEG. It also provides transparent image background option.
- SVG** SVG (Scalable Vector Graphics) is a relatively new image format, which saves the image as vectors in an XML-based text format. Generally, you will need a browser plug-in to view these images.
- SWF** SWF is an Adobe Flash file. The flash format is vector based and allows the chart to be resized after export. Also, flash allows for high-resolution printing and produces a small file size. When selecting this export type, you can also specify the frame count (number of frames in the animation) and the frame rate (number of frames shown per second) or choose to disable the animation.
- BMP** This is a Windows bitmap format.
- WMF** WMF is the Windows Meta File format. This can be used for import/export into MS Office documents.
- PDF** This will generate the chart in Adobe Portable Document Format. The chart will be generated as a one-page PDF document.
- XML** This will generate an XML data file containing the chart's data. This will not generate an XML chart attribute file, only the data will be written into XML format.
- XLS** Will generate an XLS (MS Excel) file and insert the chart as an image to the first XLS sheet.
- TXT** This will generate a text data file containing the chart's data.

In addition, if your chart contains hyperlinks, you can choose to export a map file along with any of the generated images. The map file contains an HTML image map with the links for the generate chart. Map files are generated in the same location as the image file, and have the same file name. To generate a map file, check the *Generate Data Map File* box prior to exporting.

#### 4.2.6.3.1. PDF Font Mapping

As with reports, you can use any font in the system for the labels, text, and titles in a chart. For most image exports, the text is written in the image and no additional configuration is necessary. For PDF export, however, you will need to specify a *.ttf* (true type font), *.ttc* (true type collection), *.pfb*, or *.afm* file for any system font you want to use in the chart.

To set font mapping for PDF export, select *Format* → *Font Mapping*. This will bring up a dialog allowing you to specify font files.



*Font Mapping Dialog*

For each font and style combination, you can select a specific *.ttf*, *.ttc*, *.pfb*, or *.afm* file for that font. You can either type the full path or browse to the font file. If you are using a *.ttc* file you will need to specify the

font index in the box provided (.ttc files contain more than one font). Once you have specified the correct file, click the 'Add' button to save the mapping to the list. You can edit or delete existing mappings by selecting them in the list and then clicking the appropriate button.



**Note**

You only need to apply font mapping when exporting the chart by itself. If the chart is in a report, it will inherit the font mapping defined in the report.

**4.2.6.3.1.1. PDF Font Mapping Import/Export**

You can pass the font mapping from one chart to another using the Import/Export feature. You can export the font mapping by clicking on the *Export* button on the font mapping dialog. This will bring up a dialog box prompting you to specify a file name. The font mapping will then be saved as an XML file. You can load a font mapping XML file by selecting the *Import* option from the dialog. This will bring up a dialog box prompting you to specify the XML file that you would like to import. Click on *OK* and the mapping stored in the XML file will be applied to the current chart.

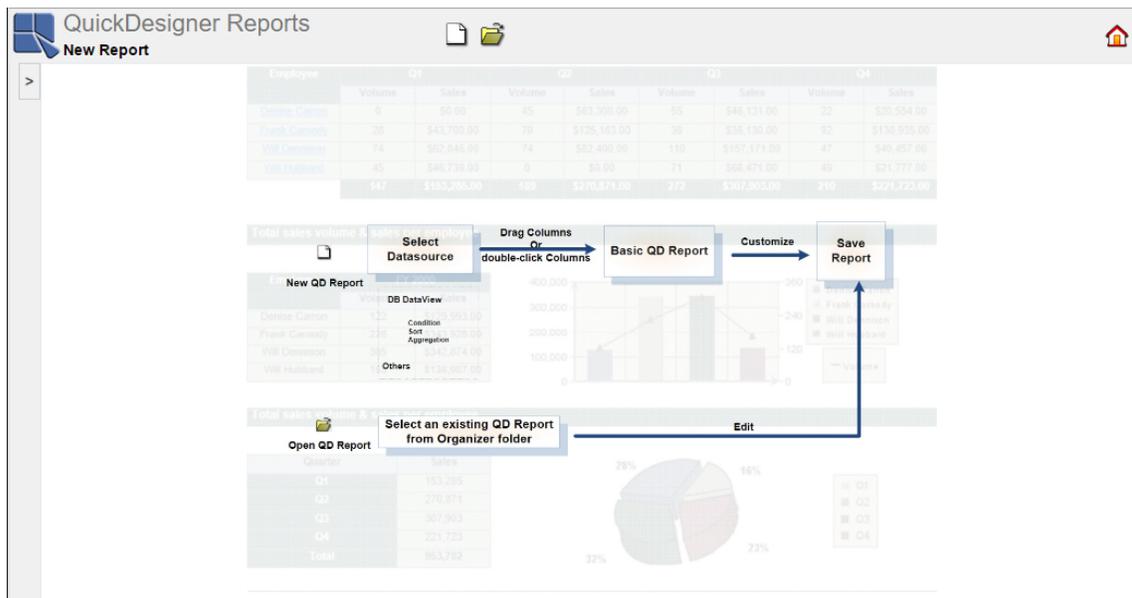
## 4.3. QuickDesigner Reports

QuickDesigner Reports is a thin-client ad-hoc reporting interface. It provides users a scaled-down design tool to create reports. With QuickDesigner Reports, end users can easily select, filter, and present data without mastering database structures, all with zero client download.

### 4.3.1. Start

QuickDesigner Reports can be started from the ERES Start page. If you have logged in as a user with design privileges, then you can follow QuickDesigner Reports link to begin using QuickDesigner Reports.

When the QuickDesigner Reports launches, the first page that appears prompts you to select between opening an existing report, or creating a new one.



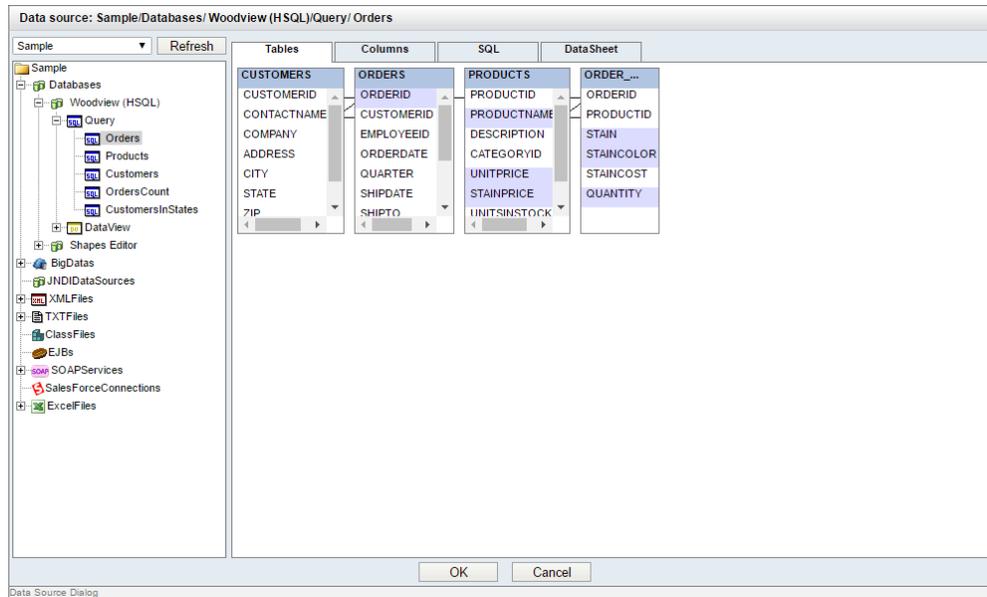
*QuickDesigner Reports Start Options*

### 4.3.2. Select a Data Source

If you want to create a new report, click on the  *Create New Report* icon on the toolbar. The *Data Source Dialog* will appear and prompt you to select data registry and data source you want to use with QuickDesigner Reports. In order to use QuickDesigner Reports, a user must have read privileges to one of the registries defined

in the Organizer. For more about creating and managing data registries, please see Section 3.1.1 - Managing Data Registries.

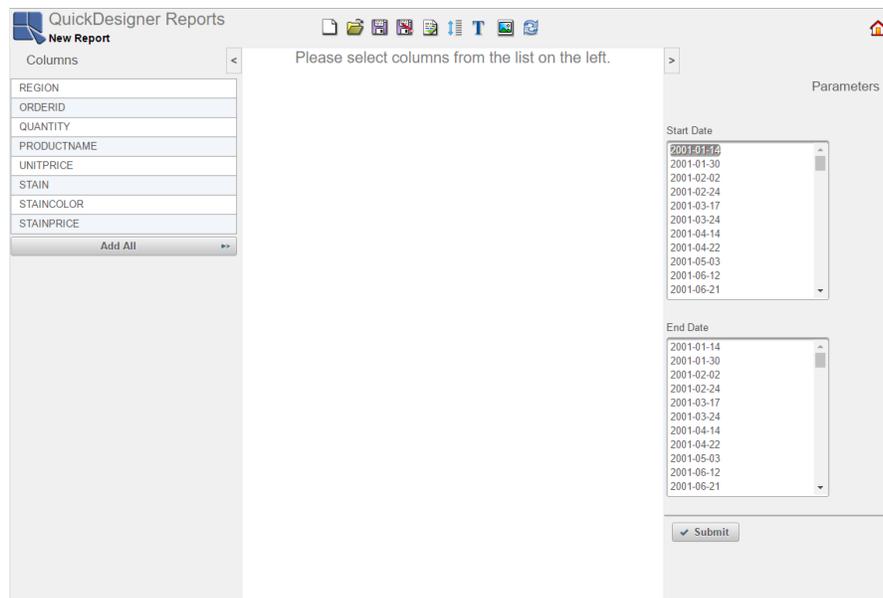
Select data registry from the drop-down menu in the upper left corner. The tree-list bellow displays the content of the selected registry. There are all of the data sources that have been defined in the registry that the user has access to. Select a data source you want to use for the report.



*Data Source Dialog*

Unlike Report Designer or Chart Designer where the user can select to create new data sources or modify the existing ones, the only option is to select a data source for the report. The only exception to this is for Data Views or Data View Queries. If data sources of this type are selected, you will see the *DataView Builder* on the right side of the Data Source Dialog. The *DataView Builder* allows you to build or modify queries against the View. For more about managing data sources in QuickDesigner Reports, please see Section 3.2 - Data in QuickDesigners and Maps.

Once you finish selecting a data source, click on the *OK* button to close the *Data Source Dialog*.



*QuickDesigner Reports*

You can see columns of the data source on the left side, the central part for creating reports, and parameters pane on the right side (this pane is visible only if the data source has some parameter(s)).

### 4.3.3. Toolbar

A lot of formatting actions for reports in QuickDesigner Reports are accessed through the toolbar. The icons perform the following actions:



Start a new report



Open an existing report



Save the current report



Save report as



Export the current report



Set style for the report



Pagination/Scrolling options



Insert/Edit report title



Insert report logo



Modify DataView Query - This option allows user to modify the DataView and DataView query used for the report. The icon is only visible in toolbar when a DV query is used as data source for the report (otherwise it's hidden).

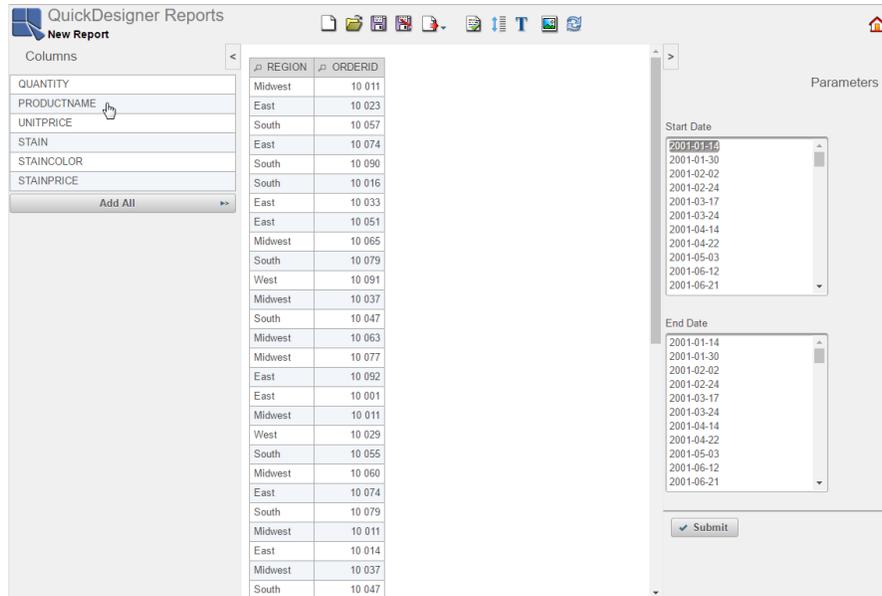


Refresh the report

### 4.3.4. Format Report Elements

#### 4.3.4.1. Add a Column

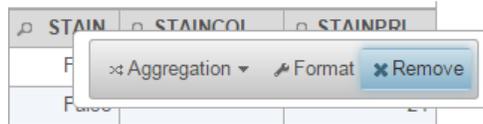
To insert a column to the report, double-click on the column in the left *Columns* pane, or click on the column and drag it to the report. You can also add all columns to the report in one step by clicking the *Add All* button.



*Add a Column*

### 4.3.4.2. Remove a Column

To remove a column from the report, right click on the column header and select *Remove* or drag the column back to the *Columns* pane. The column is removed from the report.

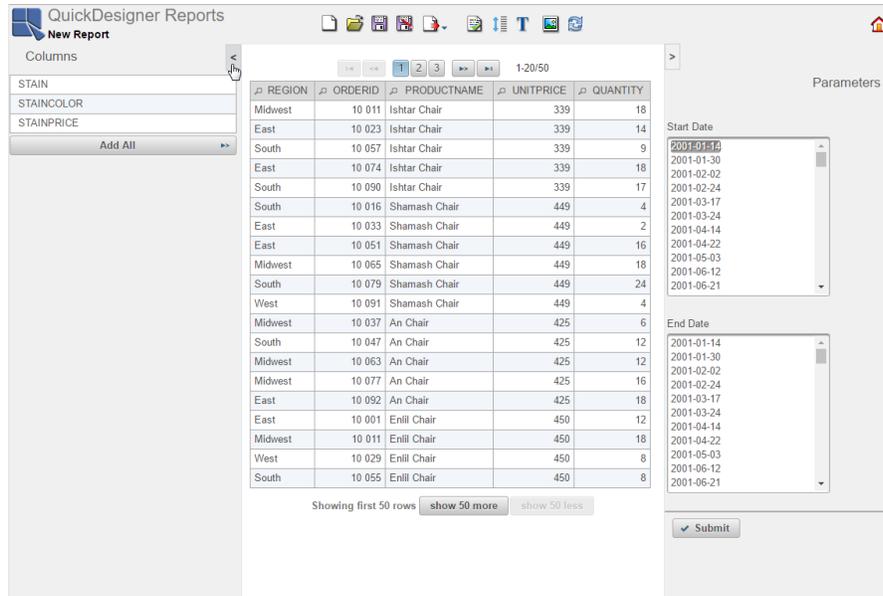


*Remove a Column*

### 4.3.4.3. Collapse Sidebars

When you add columns to the report, you can collapse the *Columns* pane to have more space for designing the report.

To do this, click on the  *Collapse* button at the top of the *Columns* pane. You can also collapse the *Parameters* pane on the right side (if the report has some parameters).



*Collapse Sidebars*

#### 4.3.4.4. Column Width

You can simply resize a column width to fit the content or to see the whole column header. Move the mouse over the right side of the column header to see double arrow, left click and drag to set the column width.



*Column Width*

#### 4.3.4.5. Column Order

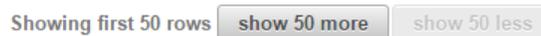
You can also reorder the columns. Left click on the column header and drag it. When you see the arrows, release the mouse button.



*Column Order*

#### 4.3.4.6. Show More Rows

By default, only the first 50 rows will be displayed in the report. You can display more or less rows by clicking on the *show 50 more* or *show 50 less* buttons under the report.



*Show More Rows*

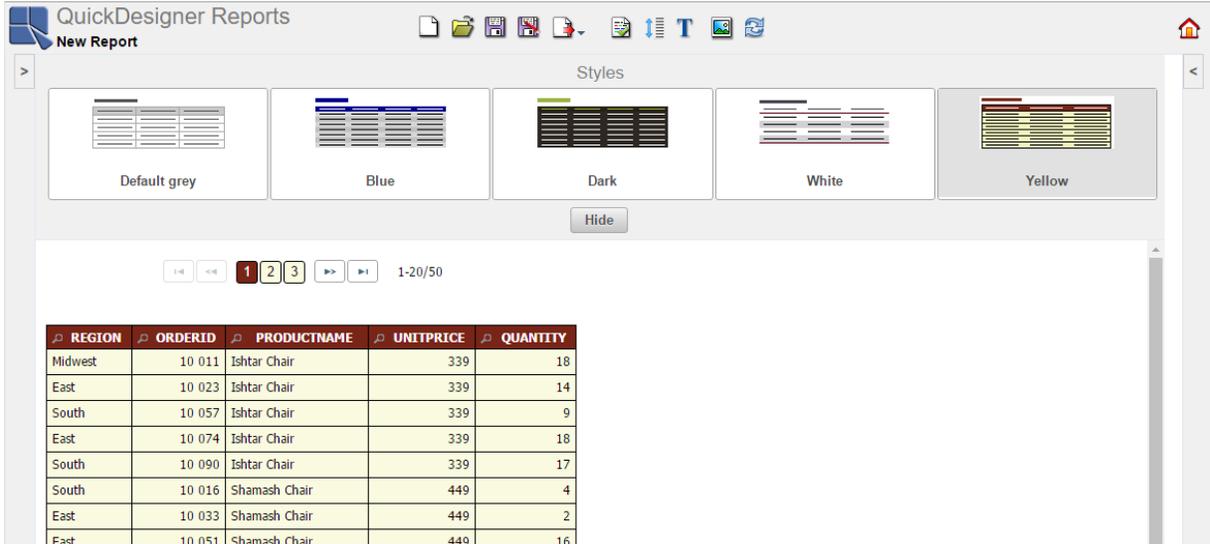
When all rows are displayed, you will also see the total number of rows.

Showing all 263 rows show 50 more show 50 less

Show Less Rows

### 4.3.4.7. Report Style

You can choose from five pre-defined report styles. Click on the  *Styles* icon on the toolbar to open the *Styles* dialog. Select the style you would like to use and close the *Styles* dialog by clicking on the *Hide* button.



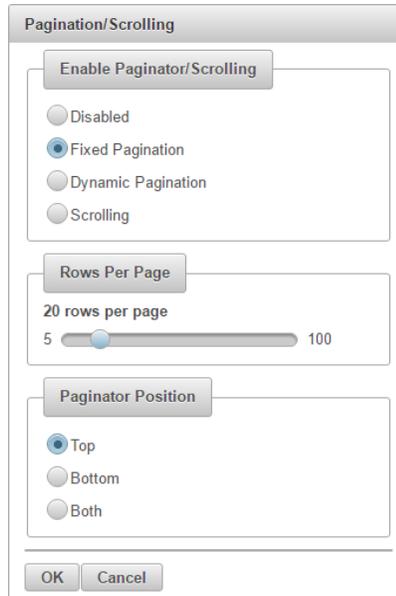
Styles Dialog

### 4.3.4.8. Pagination/Scrolling

Click the *Pagination/Scrolling* icon on the main toolbar  to open the *Pagination/Scrolling* dialog. You can choose fixed pagination, dynamic pagination that automatically keeps the height of your screen, scrolling, or disable this feature.

#### Fixed Pagination

For fixed pagination, you can select a number of rows per page and a paginator position on the page.



*Pagination Dialog*

Paginator divides the report data into separate pages. Once you set pagination to be active, you can switch to any page of the report using buttons with page numbers.

You can also switch to the first page by clicking on the  *First Page* button, to the last page by clicking on the  *Last Page* button, or to the previous and next page by clicking on the  *Previous Page* and the  *Next Page* buttons. Next to the buttons you can see which rows are displayed/total number of rows.



REGION	ORDERID	PRODUCTNAME	UNITPRICE	QUANTITY
Midwest	10 060	Enlil Chair	450	22
East	10 074	Enlil Chair	450	18
South	10 079	Enlil Chair	450	24

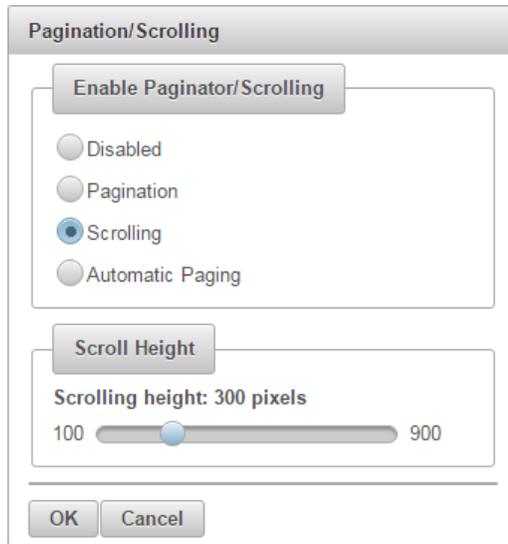
*Report with Pagination Buttons*

**Dynamic Pagination**

Dynamic pagination does not require any further setting. It will automatically adjust number of rows per page to the height of your screen. As a result, no scroll bar will appear in QuickDesigner Reports, Dashboard Builder, and Published Files. The number of rows per page may be different for different machines.

**Scrolling**

Scrolling allows you to display all the report data on a single page while not taking up too much space. All the displayed data is placed into a scrollable table with scrolling height of your choice. The scrolling height in pixels can be set in the *Scroll Height* section of the *Pagination/Scrolling* dialog.



*Scrolling Dialog*

REGION	ORDERID	PRODUCTNAME	UNITPRICE	QUANTITY
Midwest	10 011	Ishtar Chair	339	18
East	10 023	Ishtar Chair	339	14
South	10 057	Ishtar Chair	339	9
East	10 074	Ishtar Chair	339	18
South	10 090	Ishtar Chair	339	17
South	10 016	Shamash Chair	449	4
East	10 033	Shamash Chair	449	2
East	10 051	Shamash Chair	449	16
Midwest	10 065	Shamash Chair	449	18

*Report with Scrolling*

#### 4.3.4.9. Report Title

You can add a report title. Click the  *Report Title* icon on the toolbar, and the *Report Title* dialog appears. Write a title of the report and click on the  *Apply* button or press *Enter* to insert the title.



*Report Title Dialog*

The report title is on the left by default. You can move it to the center or right side of the report. Click on the title, hold down the mouse button and move the mouse slightly. You can see three dashed rectangles that represent three positions of the title - left, center, right. Move the title to the position you want to use and release the mouse button.

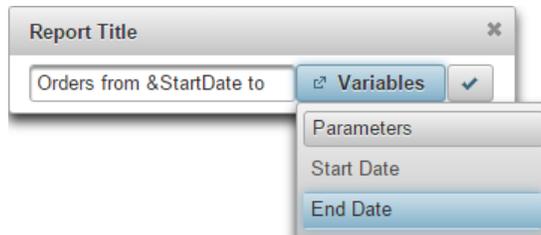


Report Title Moving

You can edit or remove the report title via the pop up *Edit* button when the cursor is pointed over the title, or via the **T** *Report Title* icon on the toolbar. Both will open the *Report Title* dialog. There you can rewrite the title and apply it with the  *Apply* button, or remove it by clicking on the  *Remove* button. Then close the *Report Title* dialog by clicking on *X* in the upper right corner. You can also remove the report title directly via the pop up *X* button when the cursor is pointed over the title.

#### 4.3.4.9.1. Report Title with Variables

If the data source is parameterized, you can add variables to the report title. Variables display the current parameter value in the report title and change dynamically according to which parameter value is selected.



Variables for Report Title

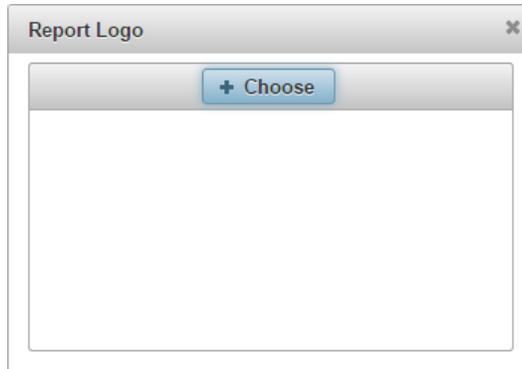
To add a variable, click on the *Variables* button in the *Report Title* dialog, select a parameter from the list, then it will be displayed in this format *&<parameterName>* in the textfield.



Report Title with Variables

### 4.3.4.10. Report Logo

You can add a logo to your report. To do this, click on the  *Report Logo* icon on the toolbar to open the *Report Logo* dialog. This dialog allows you to choose a logo from your local directory. Once a logo is selected, it is inserted to the top left.



*Report Logo Choose*

You can move the logo to the center or right side of the report. Click on the logo, hold down the mouse button and move the mouse slightly. You can see three dashed rectangles that represent three positions of the logo - left, center, right. Move the logo to the position you would like to use and release the mouse button.

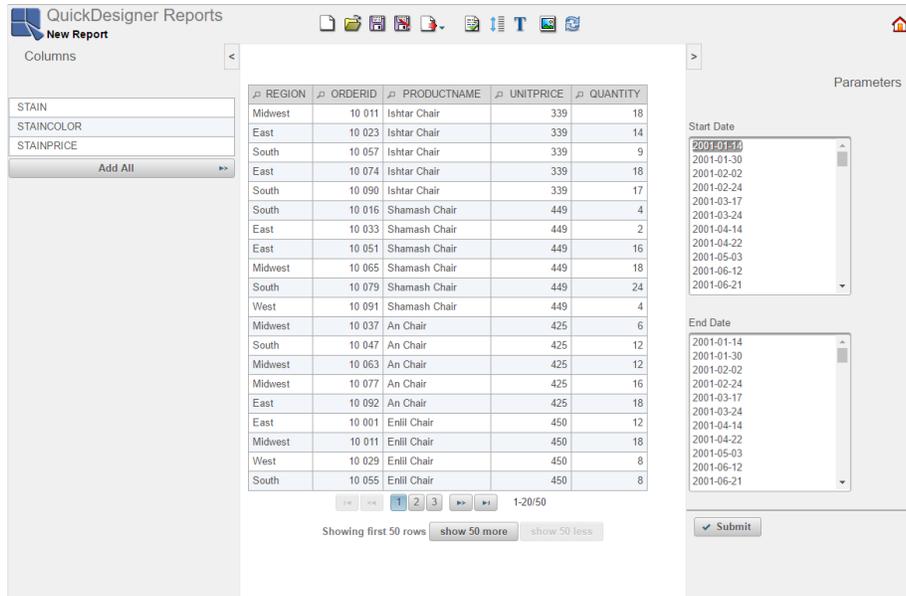


*Report Logo Placement*

You can remove the report logo via the pop up  button when the cursor is pointed over the logo. If you want to change the logo, just click on the  *Report Logo* icon to choose a new image. The new image will replace the old one.

### 4.3.5. Parameter Setting

If the report uses a parameterized data source, you can see the *Parameters* pane on the right side of the designer window. You can select parameter(s) value(s) and apply it (them) by clicking on the *Submit* button.

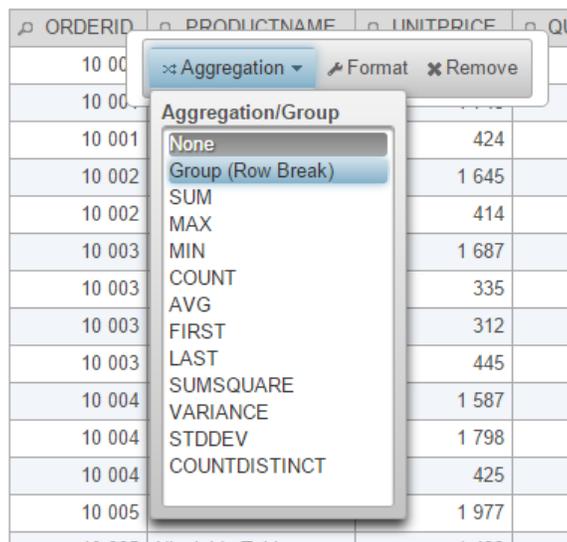


Parameters Pane

You can collapse or open this pane by clicking on the  or  Collapse buttons.

### 4.3.6. Aggregation/Group

You can add an aggregation or group to each column of the report. To do this, right click on the column header and move the mouse cursor over *Aggregation* button. The *Aggregation/Group* list will appear. Select an aggregation by clicking on it in the list. The aggregation will be assigned to the appropriate column. After you finish this setting, an extra line of the aggregation value will appear after each group (row break) and after the last aggregation value line, an extra line with a grand total value will appear.



Aggregation/Group Setting

Once the aggregations are set, you can set label for the row break and grand total. The grand total label is called Report Footer label. To set a label, right click on an empty cell next to the value you want to describe and then click on the pop up *Label* button.

	QUANTITY	SALES
0	12	5 400
5	12	20 940
4	14	5 936
		32 276
5	16	6 784
4	21	8 694
		35 014

Aggregation Label Button

The *Footer Labels* dialog will appear. You can insert a text and decide how to align it in a cell - left or right. Then click on the *OK* button to apply the label.



Footer Labels Dialog

To edit or remove a label, right click on it and then click on the pop up *Label* button and edit the text or click on the *Remove* button in the *Footer Labels* dialog.

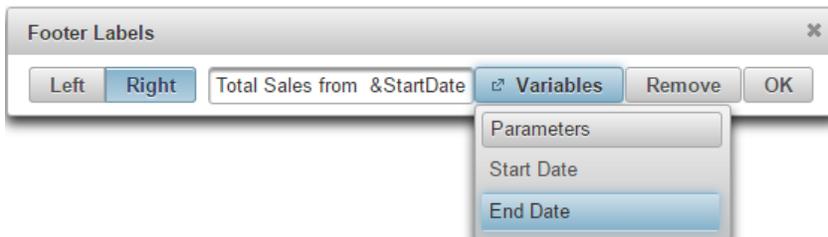
The next image shows a sample of the report with *Row Break*, *SUM* aggregation, group footer label, and report footer label.

⏪ ⏩ 34 35 36 ⏪ ⏩ 351-360/360

ORDE..	PRODUCTNAME	UNITPRICE	QUANTITY	SALES
	Apsu Dresser	1 992	8	15 936
			Order Total	30 210
10 095	Shimaliya Chair	424	16	6 784
	Bes Table	1 141	16	18 256
			Order Total	25 040
10 096	Zabada Chair	312	6	1 872
	Ninurta Chair	345	6	2 070
	Horus Table	1 354	12	16 248
			Order Total	20 190
			Total Sales	2 850 764

Aggregation and Label

If the data source is parameterized, you can add variables to the footer labels. Variables display the current parameter value in the selected footer label(s) and change dynamically according to the selected parameter value.



Footer Labels Variables

To add a variable, click on the *Variables* button in the *Footer Labels* dialog and select a parameter from the list. It will be displayed as `&<parameterName>` in the textfield.

rus Table	1 354	12	16 248
		Order Total	20 190
Total Sales from 2001-01-14 to 2003-12-09			2 850 764

*Report Footer Label with Variables*

If you add a variable to the label, the text may not fit the cell. In this case it is advisable to use text alignment.

### 4.3.7. Data Formatting

Right click on the column header and select *Format* from the pop-up menu.



*Format Button*

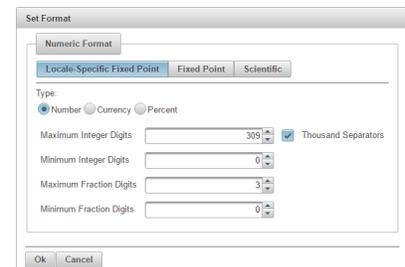
When you select the *Format* option, the *Set Format* dialog will appear. The *Set Format* dialog is dependent on what type of data is present in the selected column: numeric, string, date/time, or logical/boolean.

#### Formatting Numeric Data:

The dialog for numeric data contains three primary options for the data: *Locale-Specific Fixed Point*, *Fixed Point*, and *Scientific*. Each option opens a different tab.

#### Locale-Specific Fixed Point:

This will change the format of the data depending on the locale in which it is being viewed. Additional formatting for this option allows you to specify whether the data should be displayed as a number, currency, or percentage. In addition, you can set the maximum and minimum number of integer digits and fraction digits. Other display attributes will vary depending on locale.

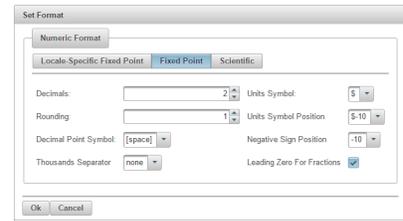


*Locale-Specific Formatting*

#### Fixed Point:

This will keep the data format consistent, regardless of locale. Additional formatting for this option allows you to set the number of decimals, rounding for digit number, unit symbols, negative sign position, decimal

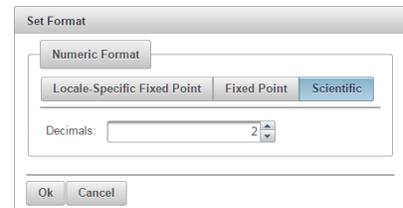
and thousands separator, and specify leading zeroes for fractions.



*Fixed Point Formatting*

**Scientific:**

This will display the data in scientific notation. Additional formatting for this option allows you to set the number of decimals.

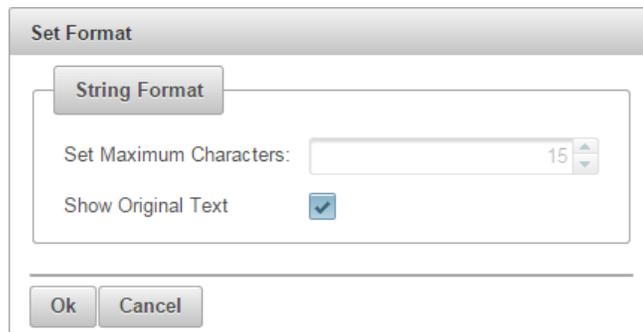


*Scientific Formatting*

After you finish selecting options, click on the *Ok* button to apply formatting and close the *Set Format* dialog.

**Formatting String Data:**

In this dialog, you can change maximum characters for the string format. To do this, uncheck the *Show Original Text* option and then set maximum characters.



*String Data Format*

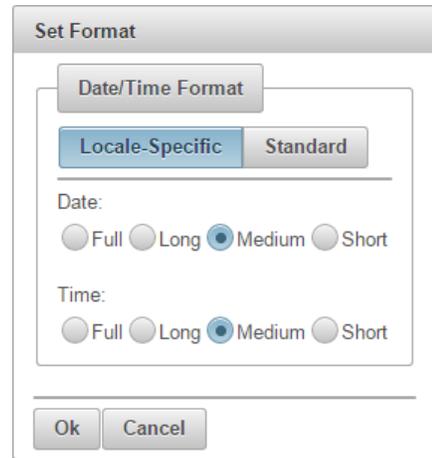
Click on the *Ok* button to apply changes and close the *Set Format* dialog.

**Formatting Date/Time Data:**

The dialog for date/time data contains two tabs: *Locale-Specific* and *Standard*.

**Locale Specific:**

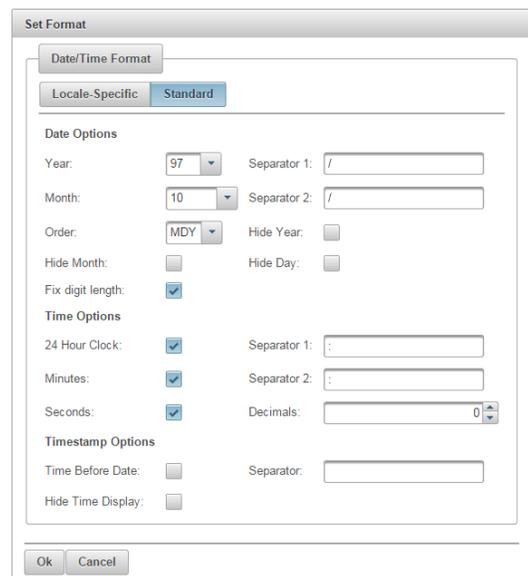
This will change the format of the data depending on the locale in which it is being viewed. Additional formatting for this option allows you to select full, long, medium, or short notations for date and time information. Other display attributes will vary depending on locale.



*Locale-Specific Formatting*

**Standard:**

This will keep the data format consistent, regardless of locale. Additional formatting for this option allows you to select year and month displays, as well as the order in which month day and year information is presented. You can also select which characters should be used as separators. Time options allow you to display hours, minutes, and/or seconds, and select the separators between them. For timestamp data, you can select to display the time before or after the date and the separator to be used between them.

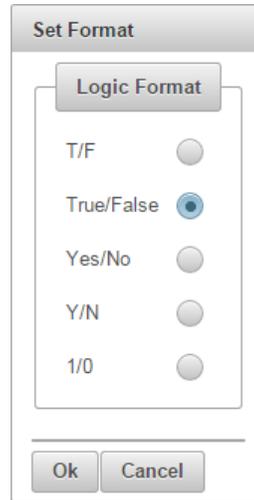


*Standard Formatting*

After you finish selecting options, click on the *Ok* button to apply formatting and close the *Set Format* dialog.

**Formatting Logical Data:**

The dialog for Logical or Boolean data contains five options for displaying the data: T/F, True/False, Yes/No, Y/N, and 1/0.



Logic Format

Select a format you want to use and then click on the *Ok* button to change the data format and close the *Set Format* dialog.

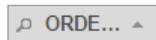
### 4.3.8. Sorting

When you hover your mouse cursor over the column header, you will see a label *Sorting: NONE*. Left click on the column header to apply the sorting. First click causes ascending sorting, second click causes descending sorting and third click cancels the sorting.

REGION	ORDERID	PRODUCTNAME
Midwest	10011	Lehter Chair
		Enlli Chair
		Enki Chair

Sorting

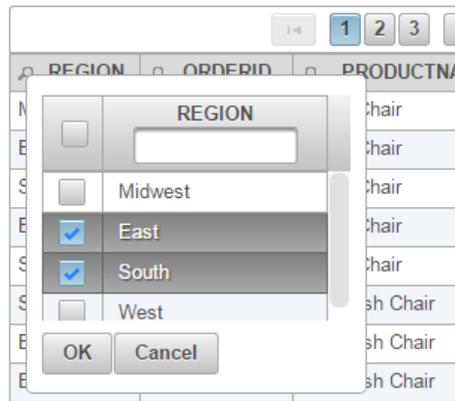
Sorted columns have arrow icons on the column header. The *Up Arrow* icon indicates ascending order and the *Down Arrow* icon indicates descending order.



Sorting Arrow

### 4.3.9. Data Filtering

You can filter data by clicking on the *Filter Column* icon on the column header. A pop-up dialog will appear, which will allow you to select data for filtering. To select values which you want to be displayed in the report, check off the box next to the desired value(s). You can narrow down the values shown in the *Filter* dialog or search for particular values by typing them into the text field at the top of the dialog. To select all available values within the column, check the box in the upper left corner (next to the search field). Click on the *OK* button to apply the filtering and close the dialog.

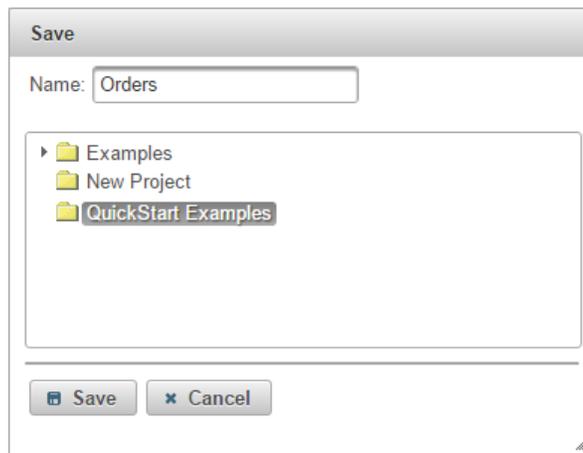


*Filter Column*

If there are several columns applying data filters, the report will display all filter conditions, i.e. different column filters has “AND” relationship.

### 4.3.10. Save the Report

You can save the report by clicking on the *Save* button on the toolbar . This will bring up a dialog allowing you to specify a name for the report. Enter a name for the report, select the project where you want to save it and click on the *Save* button.

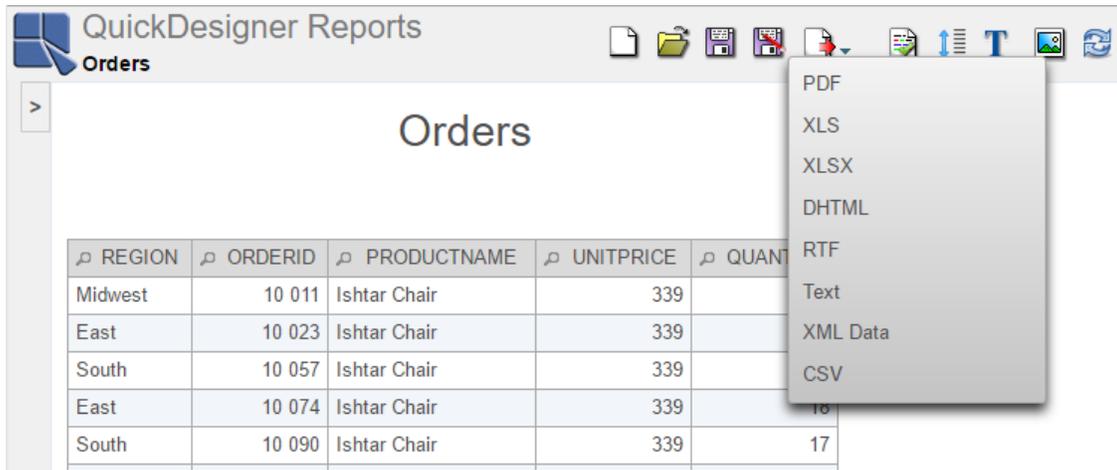


*Save Dialog*

After saving or opening existing report, the  *Save As* button allows you to save the existing or modified report under a different name or into a different location.

### 4.3.11. Export the Report

You can also export the report in several file formats. To do so, click on the  *Export* icon on the toolbar. A pop-up dialog will appear, prompting you to specify the export format. Available options are PDF, XLS, XLSX, DHTML, RTF, Text, XML Data, and CSV.

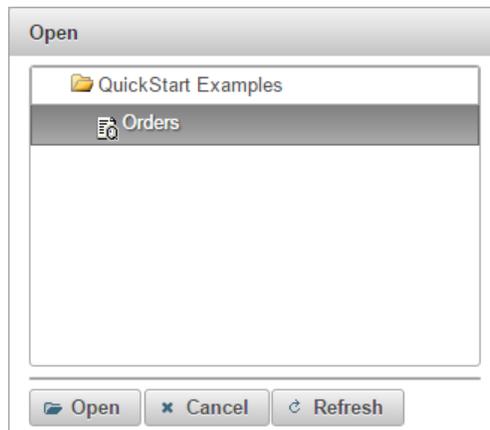


*Export Options*

Select the format and then you can save the generated file to your local system.

### 4.3.12. Open the Saved Report

You can open the saved report by clicking on the  *Open* icon on the main toolbar. The *Open* dialog will appear.



*Open the Report*

Here you can see all reports from Organizer created in QuickDesigner Reports. Select a report and click on the *Open* button to open it.

### 4.3.13. Exit

You can exit QuickDesigner Reports either by clicking on the  *Home* icon in the upper right corner, or by clicking on the *QuickDesigner Reports* title, or by clicking on the  *Logo* icon in the upper left corner. Before closing, you will be prompted if you want to save an unsaved report.

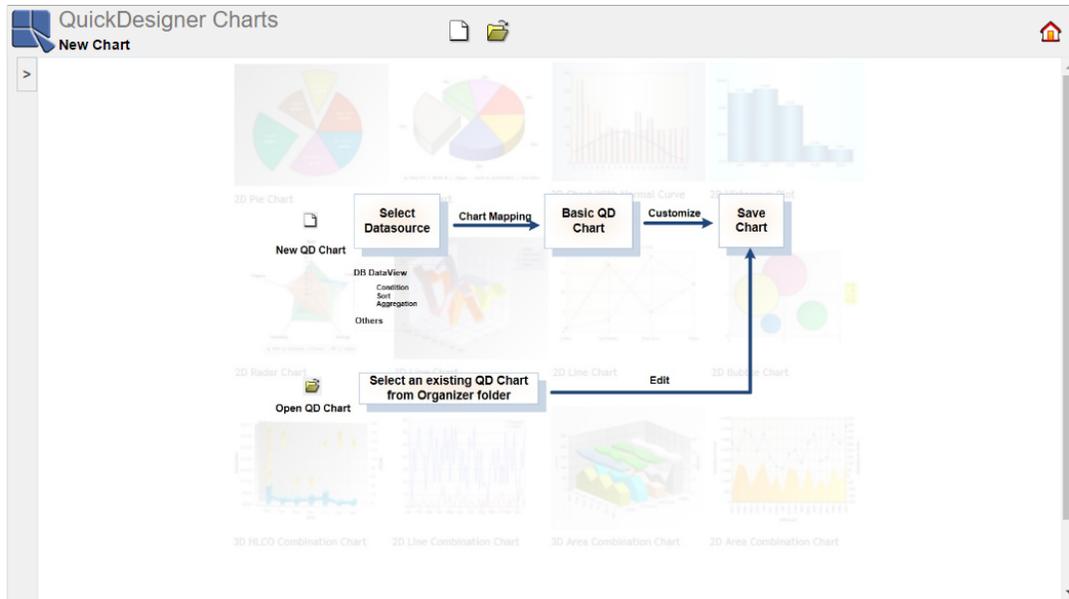
## 4.4. QuickDesigner Charts

QuickDesigner Charts is a thin-client ad-hoc charting interface. It provides users a scaled-down design tool to create charts. With QuickDesigner Charts, end users can easily select, filter, and present data without mastering database structures, all with zero client download.

## 4.4.1. Start

QuickDesigner Charts can be started from the ERES Start page. If you have logged in as a user with design privileges, you can follow QuickDesigner Charts link to start using QuickDesigner Charts.

When QuickDesigner Charts launches, the first page that appears prompts you to select between opening an existing chart or creating a new one.

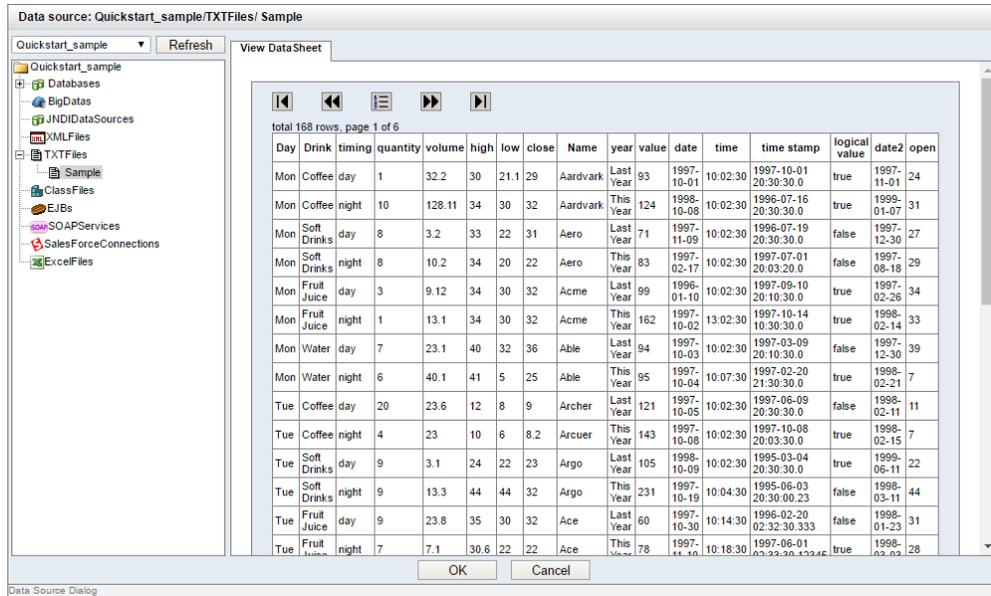


*QuickDesigner Charts Start Options*

## 4.4.2. Select a Data Source

If you want to create a new chart, click on the  *Create New Chart* icon. The *Data Source Dialog* appears and prompts you to select data registry and data source you want to use with QuickDesigner Charts. In order to use QuickDesigner Charts, a user must have read privileges to one of the registries defined in the Organizer. For more about creating and managing data registries, please see Section 3.1.1 - Managing Data Registries.

Select data registry from the drop-down menu in the upper left corner. The pane below displays the content of the selected registry. Select a data source you want to use for the chart.



Data Source Dialog

You can see the records of the data source in *View DataSheet* pane on the right. If you select *DataView* as the data source, you will be able to create a new *DataView* query in the *DataView Builder* in the right pane. You can also select an existing *DataView* query and modify it in the right pane. For more about managing data sources in QuickDesigner Charts, please see Section 3.2 - Data in QuickDesigners and Maps.

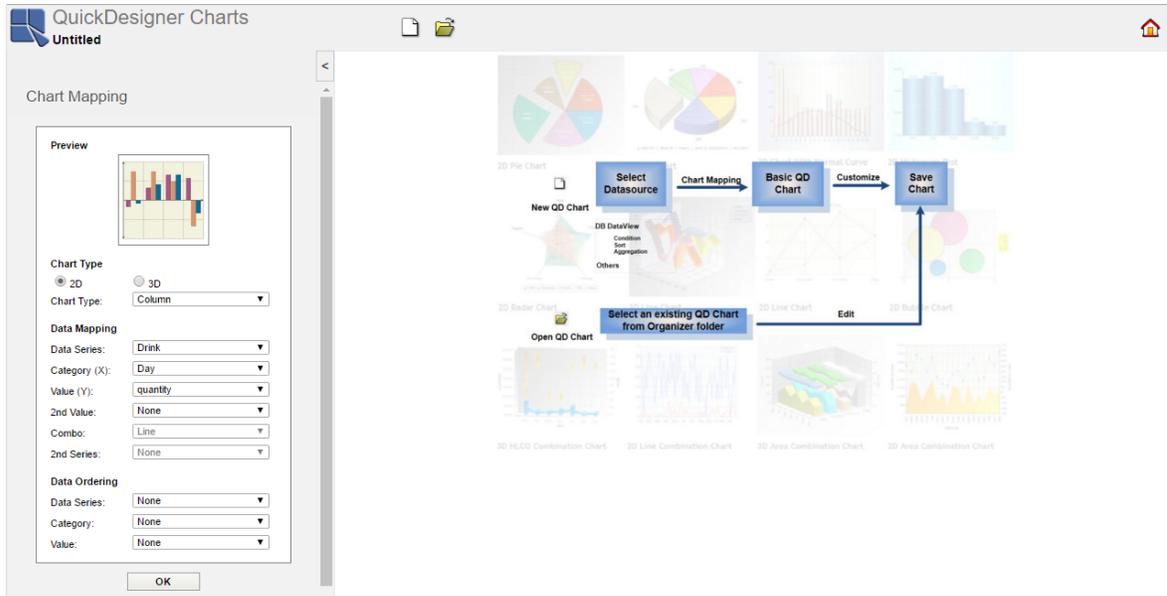
Once you have finish selecting a data source, click on the *OK* button to close the *Data Source Dialog*.

#### 4.4.2.1. Change a Data Source

After you return to the main QuickDesigner Charts window (after you created a new chart or opened an existing one), you can change the chart's data source by clicking on the  *Change Data Source* button on the toolbar. This will take you back to the *Data Source Dialog* and allow you to pick a new data source.

#### 4.4.3. Data Mapping and Ordering

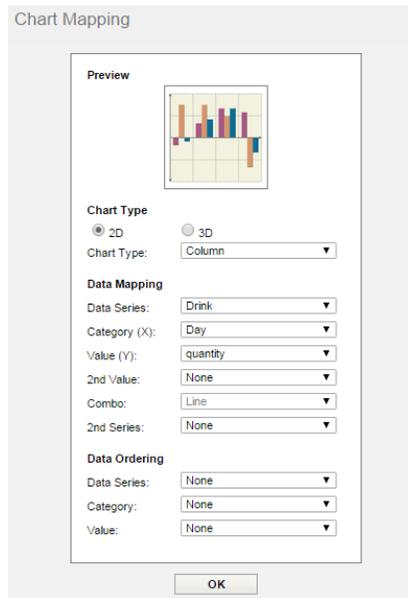
Once you select a data source or finish building a query and close the *Data Source Dialog* by clicking on the *OK* button, you will see *Chart Mapping* dialog in the left pane of QuickDesigner Charts.



*Chart Mapping Dialog*

### 4.4.3.1. Mapping and Ordering Options

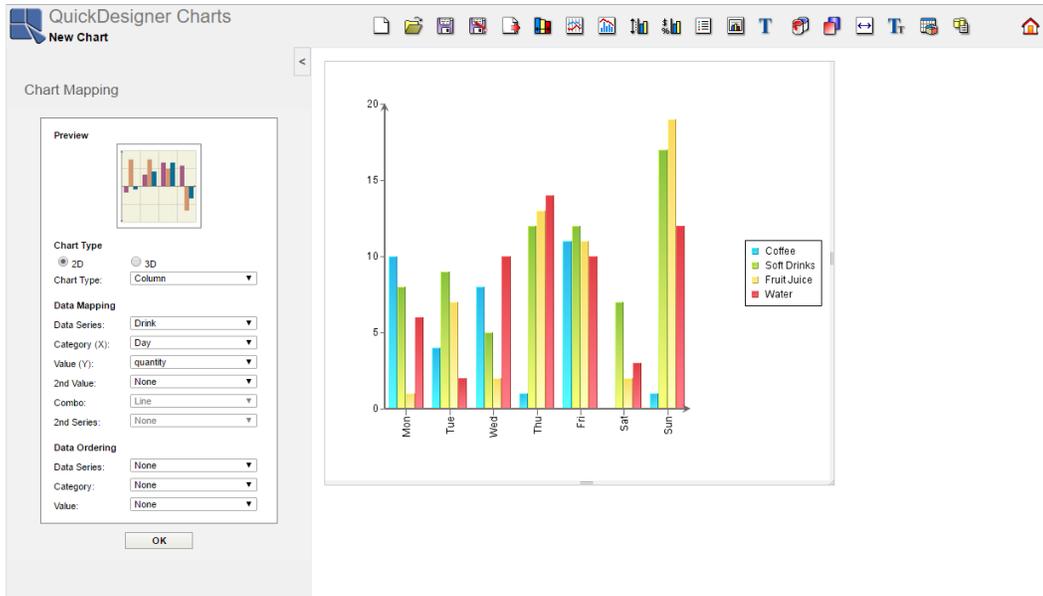
The *Chart Mapping* dialog allows you to select which chart type you want to create and which fields from the data source you want to map to the chart elements.



*Chart Mapping Options*

The *Chart Type* options allows you to select between two-dimensional and three-dimensional chart types, and the chart type you want to use. The *Data Mapping* options allows you to select data mapping for the chart. The *Data Ordering* options allows you to sort data in ascending or descending order. For a detailed description of all available charts and mapping options, see Section 4.2.3 - Chart Types and Data Mapping.

Once you select the type and mapping options, click on the *OK* button and the chart will appear in the right pane, as well as all icons.



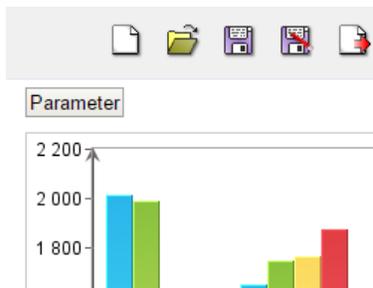
*QuickDesigner Charts Main Page*

### 4.4.3.2. Change Data Mapping

You can change the mapping options and the chart type by clicking on the  *Change Mapping/Chart Type* icon on the toolbar. This will open the *Chart Mapping* dialog in the left pane.

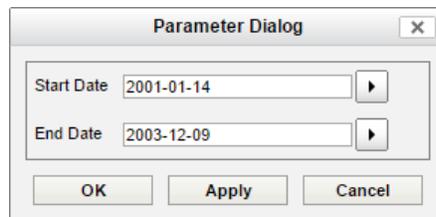
### 4.4.3.3. Parameter Setting

If the data source is parameterized, you can see *Parameter* button above the chart.



*Parameter Button*

After clicking on it, the *Parameter Dialog* will open allowing you to set parameter values. To apply a parameter value, click on the *Apply* button (the dialog stays open) or *OK* button (the dialog will be closed).



*Parameter Dialog*

### 4.4.4. Toolbar

All of the formatting actions for charts in QuickDesigner Charts are accessed through the toolbar. The icons perform the following actions:

-  Create a new chart
-  Open an existing chart
-  Save the chart
-  Save chart as
-  Export the chart
-  Set data properties for the chart
-  Chart-specific options (This option is only available for Pie, HLCO, Dial, Gantt, Polar, and Doughnut chart.)
-  Line and point settings
-  Trend line settings
-  Format axis scale
-  Format axis elements
-  Format chart legend
-  Format plot area
-  Add/Edit chart titles
-  Set chart colors
-  Set color gradient
-  Set chart canvas size
-  Set label/legend font
-  Change chart type/mapping
-  Change chart data source

## 4.4.5. Data Formatting

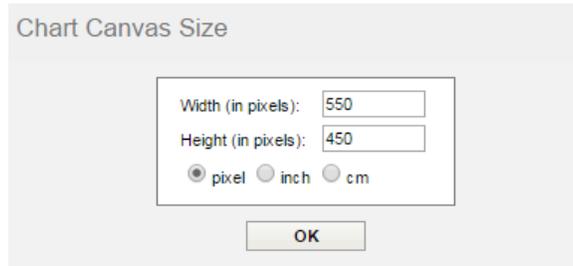
Once you finish specifying all the mapping options, you can format chart elements.

### 4.4.5.1. Position and Size

There are several options available in QuickDesigner Charts that allows you to control the size and position of the chart.

#### 4.4.5.1.1. Canvas Size

The chart canvas size can be adjusted by clicking on the  *Canvas Size* button on the toolbar. This will bring up a dialog in the left pane allowing you to specify the size of the chart.

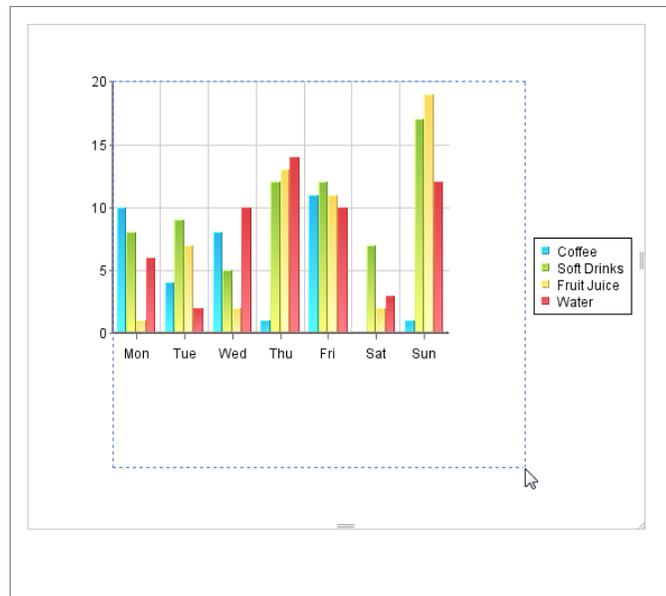


*Chart Canvas Dialog*

The dialog allows you to specify the width and height of the canvas in pixels, inches, or centimeters. Once you specify the new dimensions, click on the *OK* button and all changes will be applied. You can also change the canvas size freely using drag & drop. In this case, you can see the exact size in the *Chart Canvas Size* dialog.

#### 4.4.5.1.2. Chart Position and Size

You can set the position of the chart within the canvas freely using drag & drop. You can set the canvas size or plot area size by dragging the sides or dragging the corner.



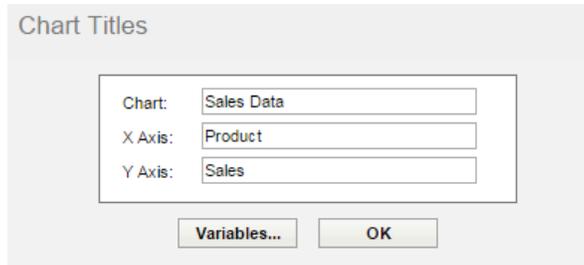
*Setting Size*

### 4.4.5.2. Font and Color Options

QuickDesigner Charts interface allows you to add/edit chart titles, set title/label font, and chart colors.

#### 4.4.5.2.1. Chart Titles

To add or edit the chart titles, click on the  *Add/Edit Chart Titles* button on the toolbar. This will bring up a dialog in the left pane.

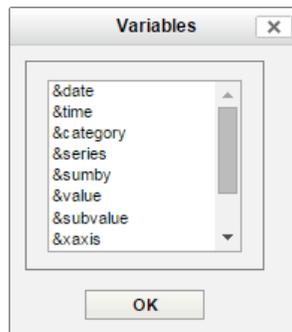


*Insert/Edit Titles Dialog*

You can add a main title to the chart, as well as one for each of the chart axes. Once you finish adding titles, click on the *Ok* button to apply the changes.

#### 4.4.5.2.1.1. Text Variables

As with the Chart Designer, you can also add text variables that will provide run-time substitution based on certain values in the chart. To bring up a list of variables, click on the *Variables* button in the *Chart Titles* dialog. The variables list will open in a new window.



*Text Variables Dialog*

The following text variables are supported:

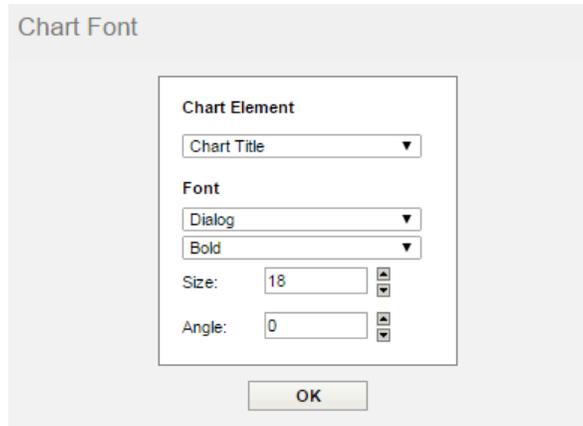
<b><i>&amp;date</i></b>	This variable displays the date when the chart was last drawn/redrawn.
<b><i>&amp;time</i></b>	This variable displays the time when the chart was last drawn/redrawn.
<b><i>&amp;category</i></b>	This variable displays the name of the category column.
<b><i>&amp;series</i></b>	This variable displays the name of the data series column.
<b><i>&amp;sumby</i></b>	This variable displays the name of the sum-by column.
<b><i>&amp;value</i></b>	This variable displays the name of the value column.
<b><i>&amp;subvalue</i></b>	This variable displays the name of the secondary value column.
<b><i>&amp;xaxis</i></b>	This variable displays the name of the column that is mapped to X-axis. This is for charts that map a value instead of a category to the X-axis like scatter or bubble charts.
<b><i>&amp;yaxis</i></b>	This variable displays the name of the column that is mapped to Y-axis. This is for scatter, bubble, and surface charts.
<b><i>&amp;zaxis</i></b>	This variable displays the name of the column that is mapped to Z-axis. This is for scatter, bubble, and surface charts.
<b><i>&amp;2ndaxis</i></b>	This variable displays the name of the column that is mapped to 2nd-axis.

- &paramInfoValue***      If the chart contains a parameter, this displays the parameter value(s) that were selected.
- &paramInfoName***      If the chart contains a parameter with this name, this will display the name of that parameter.

To add a text variable to a title, select it from the list and click on the *OK* button.

#### 4.4.5.2.2. Chart Fonts

To set the font for chart labels and titles, click on the  *Font Settings* button on the toolbar. This will open a dialog in the left pane.

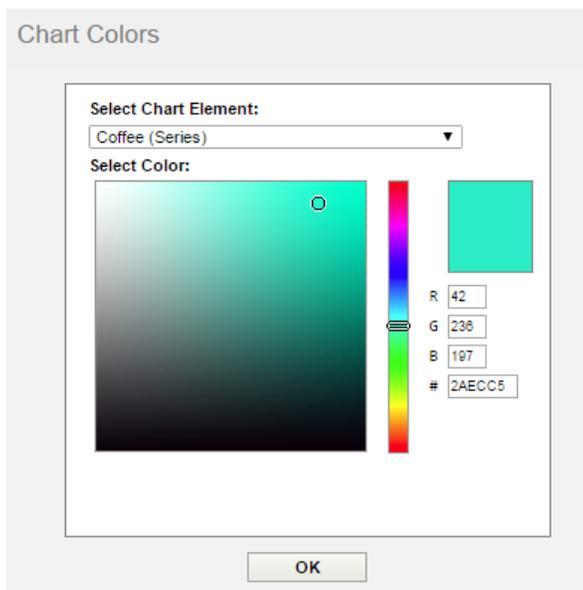


*Font Settings Dialog*

To set the font for a title or label, select it in the drop-down list at the top of the dialog. Then select font face, style, size, and angle using the options in the lower dialog. Once you finish setting up fonts, click on the *OK* button to apply the changes.

#### 4.4.5.2.3. Chart Colors

You can set color for all chart elements by clicking on the  *Set Chart Colors* button on the toolbar. The *Chart Colors* dialog will open in the left pane.



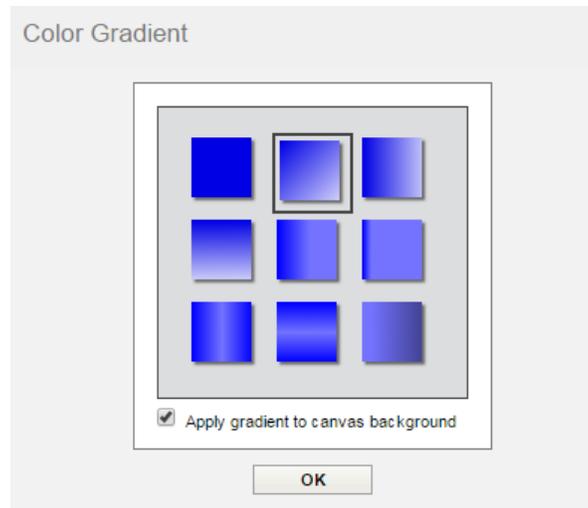
*Chart Colors Dialog*

To set the color for a chart element, first select it from the drop-down list at the top of the dialog. Then select a new color either by clicking on one of the swatches, entering RGB values for the color, or entering HEX code to the # field. Once you finish setting up the colors, click on the *OK* button to apply the changes.

Please note that some elements are not visible by default (e.g. data border). In this case, first make the element visible (see Section 4.4.5.4.1 - Data Properties) and then use *Chart Colors* dialog for setting a color of the element.

#### 4.4.5.2.4. Color Gradient

You can set the color gradient for the chart by clicking on the  *Set Color Gradient* button on the toolbar. The *Color Gradient* dialog will open in the left pane.



*Chart Colors Dialog*

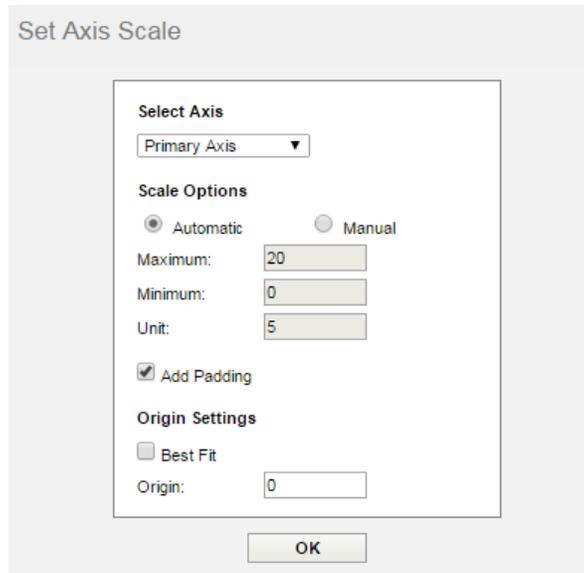
Select a gradient you want to use and click on the *OK* button to apply the changes. You can apply the same gradient to the canvas background.

#### 4.4.5.3. Chart Axes

In QuickDesigner Charts, you have several formatting options for the chart axes. You can customize the axis scale as well as label/axis appearance options.

##### 4.4.5.3.1. Axis Scale

By default, the scale of any value axes in the chart is calculated to provide the *best fit* for the plotted data. This is especially useful if the data change often. However, there are many times when you may want to set the scale of the axes manually. To modify the axis scale, click on the  *Axis Scale* button on the toolbar. A new dialog will open in the left pane.



*Axis Scale Dialog*

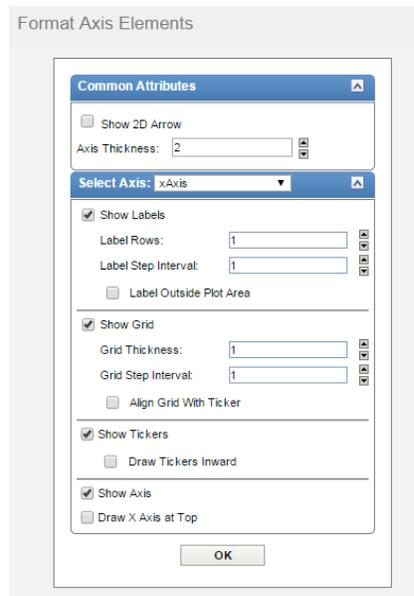
You can select the axis for which you would like to set the scale by selecting it in the drop-down list at the top of the dialog. The following scaling options are provided for each value axis:

- Automatic:** This option turns on automatic scaling for the axis. This is the default setting.
- Manual:** This option turns on manual scaling to set the axis scale to the options provided.
- Maximum:** This is the highest value on the axis scale.
- Minimum:** This is the lowest value on the axis scale.
- Unit:** This is the step size of the axis scale.
- Add Padding:** This option will add padding to the top of the automatic scale. If this option is disabled, the maximum scale will match the largest data point.
- Best Fit:** This option will automatically place the origin of the chart based on minimum and maximum values of the data.
- Origin:** This option allows you to specify where the X and Y axes should intersect. This is usually at zero.

Once you finish specifying the axis scale options, click on the *OK* button to apply the changes.

#### 4.4.5.3.2. Axis Elements

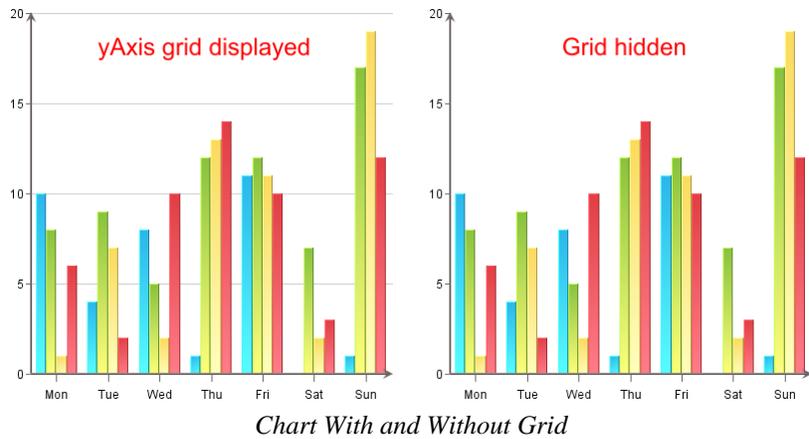
Additional appearance properties for the axes and axis labels can be set in the *Format Axis Elements* dialog. To change these settings, click on the  *Axis Options* button on the toolbar. This will open the dialog in the left pane.



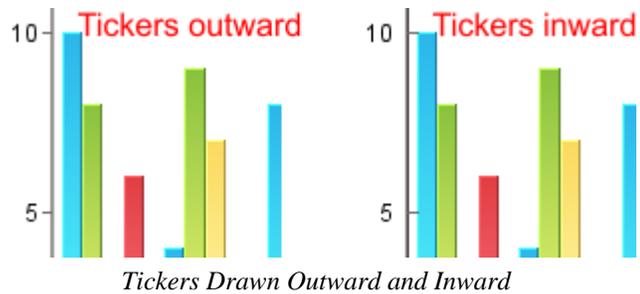
*Axis Elements Dialog*

To change the settings for an axis, first select it in the drop-down *Select Axis* list and then set the available options below. The dialog allows you to perform the following options. Note that some options are only available for certain chart types, certain data types, and on certain axes.

- Show 2D Arrow:** This option allows you to remove or display arrowhead at the end of the axis. Note that this option applies to all chart axes and is only available for two-dimensional charts.
- Axis Thickness:** This option allows you to set the thickness of the axis in pixels. Note that this setting is only available for two-dimensional charts and it is applied to all axes in the chart.
- Show Labels:** This option allows you to remove or display the labels for each axis.
- Label Rows:** This option allows labels to be displayed in alternating rows. This can prevent overlapping. This option is only available for the X-axis.
- Label Step Interval:** This option allows you to set the label step interval for the data. For example setting this to 2 would draw the label for every other data point in the chart.
- Label Outside Plot Area:** This option sets the labels to be placed outside of the plot area, regardless of where the axis is. This feature can be useful for category axis labels if you are plotting data with both positive and negative values.
- Label Interval Unit:** This option allows you to select the unit to be used when sorting and representing time-based data (date, time, or timestamp). Selecting tickers will use the data as it is read by Chart Designer.
- Show Grid:** This option allows you to remove or display the grid for each axis.



- Grid Thickness:** This option allows you to specify the thickness of any grid lines along the axis.
- Grid Step Interval:** This option allows you to set the grid step interval for any grid lines along the axis.
- Align Grid With Ticker:** This option aligns the grid line with the ticker instead of placing it between tickers. This places the ticker and corresponding grid line along the same line. This option only applies to the category axis of column-type charts.
- Show Tickers:** This option allows you to remove or display the axis tickers and draw them inward or outward.



- Draw Tickers Inward:** This option will draw the axis tickers inside the plot area instead of outside (default).
- Show Axis:** This option allows you to remove or display the axis (for two-dimensional charts) or the wall (for three-dimensional charts).
- Draw X Axis at Top:** This option allows the X-axis to be positioned at the top of the chart instead of the default bottom position. This option is available for two-dimensional column, bar, scatter, high-low, HLCO, bubble, and Gantt charts.
- Swap Y Axis Position:** This option will swap the primary and secondary value axes. This option can only be found under 2nd Axis tab.

**4.4.5.3.2.1. Axis Label**

The *Format Axis Elements* dialog also allows you to format the appearance of the axis labels, depending on what type of data is plotted on the axis.

- **Formatting Numeric Data:** For numeric data, there are three primary options for display formatting: locale-specific fixed point, fixed point, and scientific. Additional options will be displayed when you click on the *Format* button.

*Numeric Data Format Options*

**Locale-Specific:**

This will change the format of the data depending on the locale in which it is being viewed. Additional formatting for this option allows you to specify whether the data should be displayed as a number, currency, or percentage. In addition, you can set the maximum and minimum number of integer digits and fraction digits. *Use Grouping* brings space as the thousands separator. Other display attributes will vary depending on locale.

Please note that if you select *Currency* as *Type*, local-specific format will automatically take currency symbol from your system settings (e.g. it will add “\$” to the label if you're in U.S.A.).

*Locale-Specific Formatting Options*

**Fixed Point:**

This will keep the data format consistent, regardless of locale. Additional formatting for this option allows you to set the number of decimals, rounding for digit number, unit symbols, negative sign position, decimal and thousands separator, and to use *Leading Zero For Fractions* which determines whether it should display 0 before decimals for numbers from -1 to 1 (e.g. 0.15/ .15).

*Fixed Point Formatting Options*

**Scientific:**

This will display the data in scientific notation. Additional formatting for this option allows you to set the number of decimals.

Scientific

Decimals: 2

*Scientific Formatting Options*

• **Formatting Date/Time Data:**

For date/time data, there are two primary options for display formatting: locale specific and standard. Additional options will be displayed if you click on the *Format* button. The available options will vary depending on the nature of your data. Date, time, and timestamp data will bring up date, time, and date & time options.

Date Format

Locale-Specific

Standard

*Date/Time Data Format Options*

**Locale-Specific:**

This option will change the format of the data depending on the locale in which it is being viewed. Additional formatting for this option allows you to select full, long, medium, or short notations for date, and time information. Other displayed attributes will vary depending on your locale.

Locale-Specific

Date:

Full Long Medium Short

Locale-Specific

Time:

Full Long Medium Short

*Locale-Specific Formatting Options*



*Example of Long/Medium/Short Time Format*

**Standard:**

This option will keep the data format consistent, regardless of locale. Additional formatting for this option allows you to select year and month displays, as well as the order in which month, day, and year information is presented. You can also select the characters to be used as separators. Time options allow you to display hours, minutes, and/or seconds, and select the separators between them. For timestamp data, you can select to display the time before or after the date and the separator to be used between them. *Fix Digit Length* option adds zeros to days and months between 1 – 9 to keep the length of the date consistent (two digits for both day and month).

Standard

**Date Options:**  
 Year: 97 Month: 10  
 Order: MDY  
 Separator 1: /  
 Separator 2: /  
 Hide Year  Hide Month  
 Hide Day  Fix Digit Length

**Time Options:**  
 24 Hour Clock Separator 1: :  
 Minutes Separator 2: :  
 Seconds Decimals: 0

**Timestamp Options:**  
 Time Before Date Separator: :  
 Hide Time Display

*Standard Formatting Options*

• **Formatting Logical Data:**

There are five options available for displaying logical or Boolean data: T/F, True/False, Yes/No, Y/N, and 1/0.

Logical Format

T/F  
 True/False  
 Yes/No  
 Y/N  
 1/0

*Logical Formatting Options*

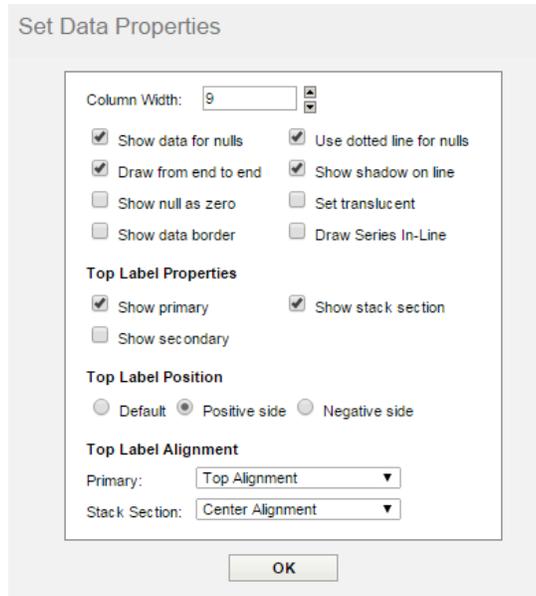
**4.4.5.4. Plot/Data Elements**

In QuickDesigner Charts, there are many ways to configure how data points and chart plot should be drawn.

**4.4.5.4.1. Data Properties**

Many of the data display options are controlled through the data properties dialog. From this dialog, you can control the size of bars/columns, set display options for null values, and specify options for data labels. To invoke this

dialog, click on the  *Data Properties* button on the toolbar. The dialog will open in the left pane.

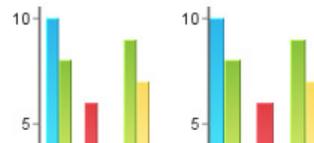


*Data Properties Dialog*

This data properties dialog contains the following options:

**Column Width:**

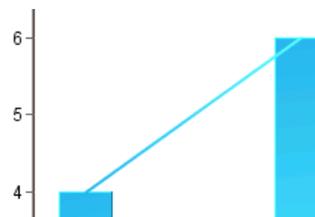
This option specifies the ratio of the bar/column width relative to the gap between successive bars in the chart. Each unit represents 1/10th of the space between data points. Therefore, entering **9** would leave 10% of the space between data points blank, while **10** would eliminate all space between bars/columns. This option only pertains to two-dimensional bar, column, stack bar, stack column, high-low, HLCO, and Gantt charts.



*Column width 9 vs 10*

**Show data for nulls:**

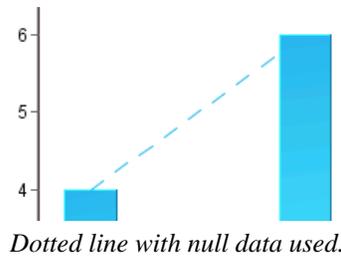
This option will connect null data with lines. For example, if you have three points and the value of point 1 is 4, point 2 is null, and the value of point 3 is 6, then a line will be drawn from 4 to 6 for the three points. This option is only available for line charts or other two-dimensional charts with lines. All other chart types will not plot null data.



*Connected line with null data used.*

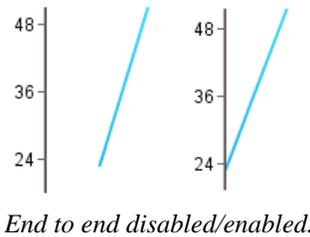
**Use dotted lines for nulls:**

Instead of drawing a whole line through null values in the chart, you can use this option to draw a dotted line. Like the show data for nulls option, this option is only available for lines.



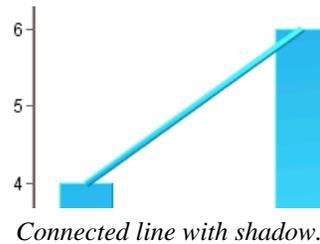
**Draw from end to end:**

This option allows you to draw 2D line and area charts across the entire plot area, rather than offsetting to the first and last data points on the chart.



**Draw shadow for lines:**

This option specifies whether to draw shadow on 2D lines. In order for shading to apply, the line must be thicker than one pixel.



**Show null as zero:**

This option will display 0 for null data.

**Set translucent:**

This option will draw all data points in translucent color, allowing overlapped elements to be visible. This is useful for radar charts and two-dimensional area charts with data series. This option is also available for bubble and Gantt charts. Note that because this feature requires Java 1.2 or higher, the translucent elements will not display if you are deploying charts in applets and the client is not using the Java plugin.

**Show data border:**

This option will draw a border around columns and areas.

**Draw Series In-Line:**

This option will display series in one line. It only pertains to three-dimensional column, stack column, percent column, and bar charts.

**Show primary:**

This option will display data top labels for the primary values in the chart.

**Show secondary:**

This option will display data top labels for the secondary values in the chart.

**Show stack section:**

This option will display individual labels for each stack section for stack bar, stack column, and stack area charts.

**Top label position:**

This option allows you to specify where the top labels should be drawn. By default, they are drawn above data points if they are positive and below data

points if they are negative. Using these options, you can force the data points to always draw to the positive or negative side of the data.

**Label Alignment:**

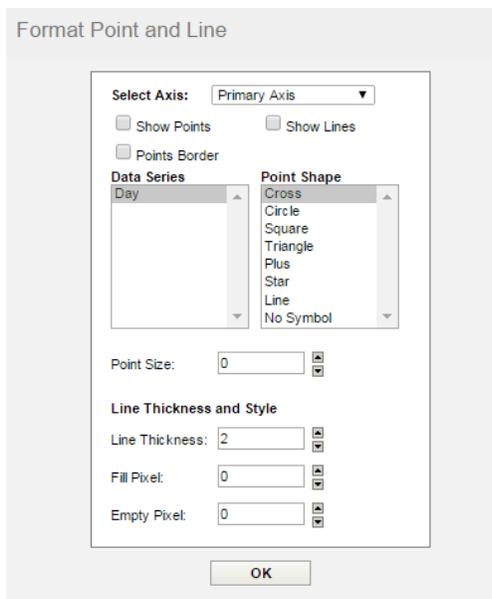
This option allows you to set the alignment for the data top labels. You can draw them at the top, bottom, or middle of the data points. In addition, you can select to draw the label inside the data point at the top or bottom. An additional option for stack charts allows you to set the alignment for stack section labels.

Once you finish setting up the options, click on the *OK* button to apply the changes.

**4.4.5.4.2. Line And Point**

For any two dimensional chart type, you can choose to display lines and points for all data points in the chart. Note that some chart types already use this representation (i.e. line or scatter charts). Line and point settings can be

adjusted by clicking on the  *Line and Point Settings* button on the toolbar. The dialog will open in the left pane.



*Line and Point Dialog*

The first three options allows you to specify whether you want to show lines, points, and points border for the chart. For radar charts, you also have the option of showing areas. The remaining options allows you to customize the line and point displays for each element in the data series.

For each data series element, you can specify the point shape you want to use. You can also control the size of the points. The default point size is 0. You can specify point sizes of -1, -2, & -3 which represent sizes of 0.75, 0.5, and 0.25. At -3 (0.25), the point will be drawn as a dot regardless of the selected point shape.

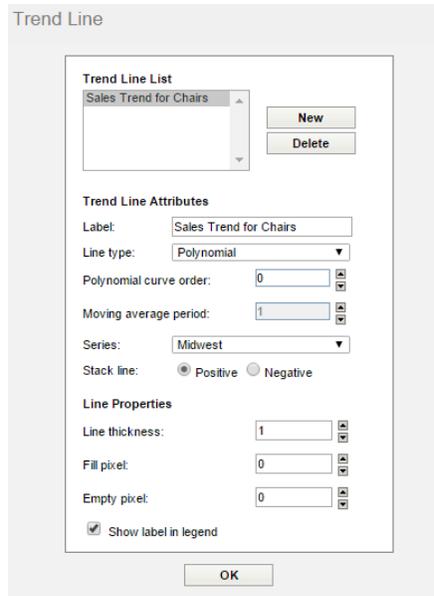
For lines you can specify the line thickness as well as customize a dash pattern. The dash pattern is created by specifying the number of filled pixels and empty pixels (between 0 and 255). The line is then drawn by dividing into segments - the number of filled pixels followed by the number of empty pixels. Setting 0 for both will result in a solid line. Setting 255 for both will result in no line being drawn.

Once you finish setting up the options, click on the *OK* button to apply the changes.

**4.4.5.4.3. Trend Line**

A powerful feature in QuickDesigner Charts is the ability to add trend lines to charts. Trend lines can help you show more details of a chart's data by exposing and highlighting certain trends. Trend lines settings can be adjusted by

clicking on the  *Trend Line* button on the toolbar. The dialog will open in the left pane.



*Trend Line Dialog*

There is a list of existing trend lines at the top of the dialog. You can select existing trend line and edit it, remove the selected trend line, and/or create a new trend line.

You can specify a label for the line as well as on which element of the data series the calculation should be based. The following types of trend lines are supported: polynomial of any degree, exponential, logarithmic, power, moving average, exponential moving average, triangular moving average, and cubic B-spline. For moving averages you will need to specify the average period and for a polynomial you will need to specify the curve order. You can also specify the thickness of the line and configure whether a label in legend and the attached text should be shown. In case the chart has data series, you can configure the trendline for a specific series.

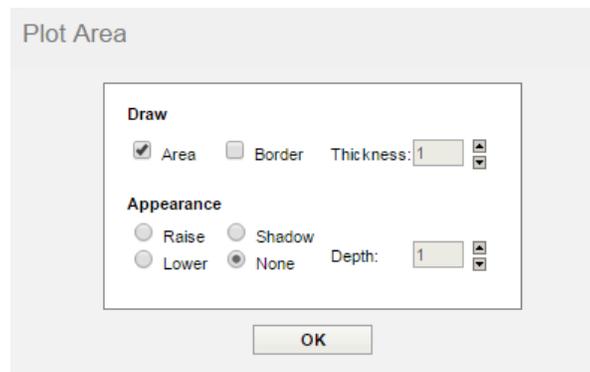
Please note that there is a *Stack line* option in the *Trend Line* dialog. This option is only available for Stack Column, Stack Bar, and Stack Area chart. You can select whether the line will be drawn for positive or negative values.

Once you finish setting up the options, click on the *OK* button to apply the changes.

#### 4.4.5.4.4. Plot Area

The plot area is the plane on which the data points are drawn for two-dimensional charts. You can customize the

appearance of the plot area by clicking on the  *Format Plot Area* button on the toolbar. The dialog will open in the left pane.



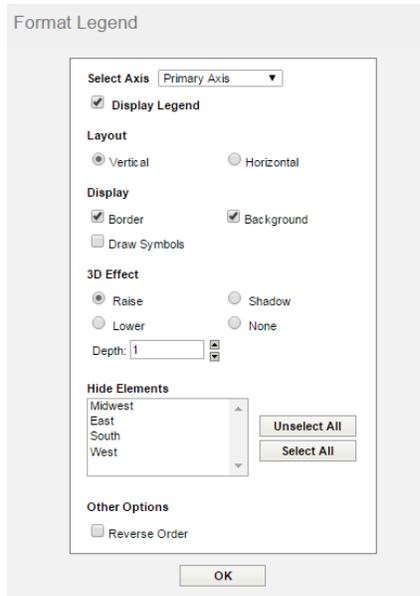
*Plot Area Dialog*

This dialog allows you to draw a border around the plot area or fill it to provide a background color. If you fill the area, you can also specify several 3D effects like raising, lowering, or drawing a shadow. You can emphasize the 3D effect by increasing the *Depth* option.

Once you finish setting up the options, click on the *OK* button to apply the changes.

#### 4.4.5.4.5. Chart Legend

You can control and modify the display of the chart legend by clicking on the  *Format Legend* button on the toolbar. The dialog will open in the left pane.



*Format Legend Dialog*

The drop-down list at the top of the dialog allows you to select which legend (primary or secondary) you want to modify. The formatting options are as follows:

- Display Legend:** This option allows you to turn on or off the legend.
- Layout:** This option allows you to change the legend from vertical to horizontal layout.
- Display:** This option allows you to display border, background, and symbols for the legend.
- 3D Effect:** This option allows you to add a 3D effect to the legend. You can raise it, lower it, or draw a shadow. The 3D effect can be emphasized by increasing the *Depth* option.
- Hide Elements:** This option allows you to select certain category/series elements to hide in the legend.
- Other Options:** This option allows you to choose whether or not to draw the legend in reverse order.

Once you have finish seting up the options, click on the *OK* button to apply the changes.

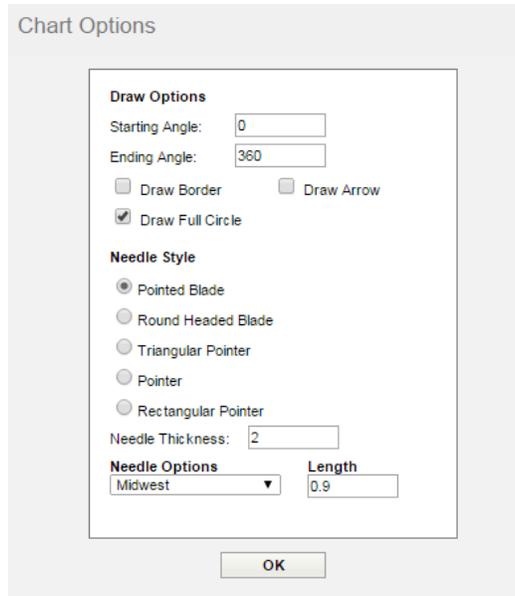
#### 4.4.5.5. Chart-Specific Options

There are a number of formatting options that are unique to certain chart types - Pie Chart, Dial Chart, HLCO

Chart, Gantt Chart, Polar Chart, Doughnut Chart. These options can be modified by clicking on the  *Chart Type Options* button on the toolbar in these types of charts. This will bring up a dialog in the left pane that varies depending on the type of the current chart. Besides the chart types listed here, other chart types have no additional options.

##### 4.4.5.5.1. Dial Charts

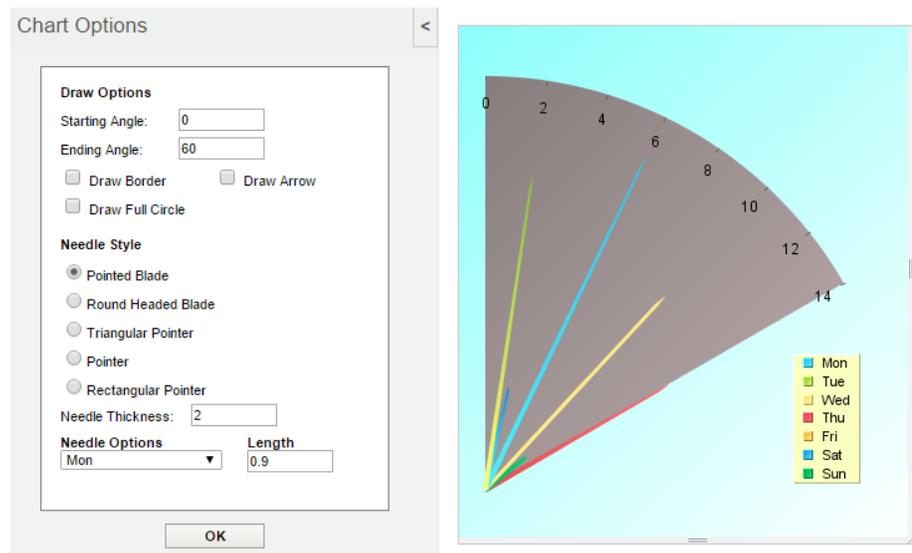
For dial charts, the following dialog is opened:



*Dial Chart Options Dialog*

The following options are available for dial charts:

- Starting Angle:** This option specifies the angle of the first axis label. The angle is represented in degrees and is set to 0 by default. Assuming the dial chart is a clock face, 0 degrees is 12 o'clock.
- Ending Angle:** This option specifies the angle of the last axis label. The angle is represented in degrees and is set to 360 by default. By default, the labels (and data points) encompass the entire circumference of the dial.
- Draw Border:** This option specifies whether or not to draw a border around the dial.
- Draw Arrow:** This option specifies whether or not to draw arrowheads at the end of the dial hands.
- Draw Full Circle:** This option specifies whether to draw the dial as a complete circle (360 degrees) or only draw the portion of the circle determined by the starting and ending angles.



*Circular Sector*

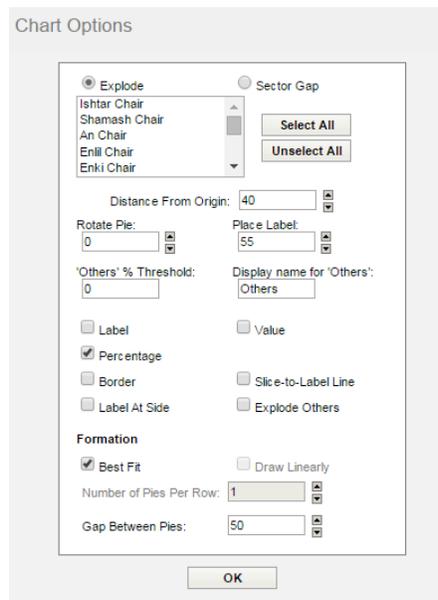
- Needle Style:** This option allows you to set style of the needle. The default one is *Pointed Blade*.

**Needle Options:** This option specifies the distance of the hand for each needle (if multiple needles exist) from the center of the dial. The range is from 0 (center of the dial) to 1 (end of the dial). You can adjust the needle length for each category by selecting the category from the drop-down list.

#### 4.4.5.5.2. Pie and Doughnut Charts

Please note that doughnut chart specific options are exactly same as for pie charts, so even though the next section will only mention pie charts, the same options also apply for doughnut charts.

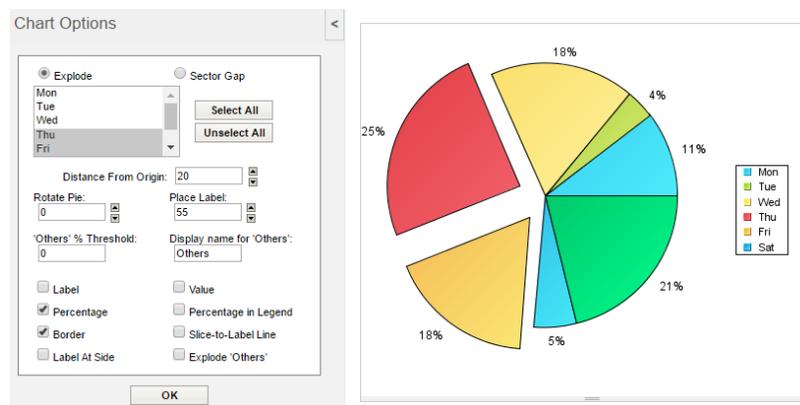
The following dialog is opened for pie charts:



*Pie Chart with Series Options Dialog*

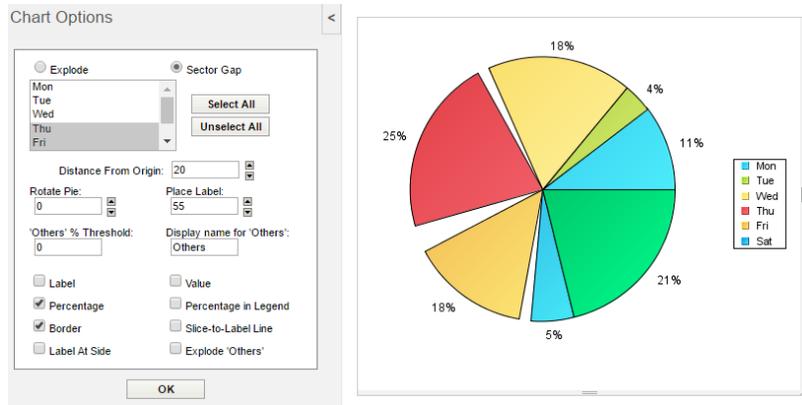
Different options will appear in this dialog depending on whether the pie chart is two-dimensional or three-dimensional and whether or not it has a series. The following options are available for pie charts:

**Explode:** This option allows you to pick one or more category/series elements whose sections are to be drawn at a certain distance away from the center of the pie.



*Exploded Slices*

**Sector Gap:** This option allows you to pick one or more category/series elements whose sections maintain the same distance from the center of the pie. This option creates a gap between the slices.



*Sector Gap Applied on Slices*

**Distance From Origin:**

This option allows you to specify how far the exploded sections are to be drawn away from the center. This number, represented as a percentage of the radius, indicates the distance between the center and the tip of the pie slice to be exploded. For sector gap, this option is used to define the gap between the slices representing selected categories/series.

**Rotate Pie:**

This option allows you to specify the number of degrees that the chart should be rotated. To rotate the pie in a clockwise direction, use negative values (i.e. -90). To rotate it in a counter-clockwise direction, use positive values (between 1 - 360).

**Place Label:**

This option indicates the distance of the labels from the center of the pie.

**'Others' % Threshold:**

This feature is useful for pie charts that have a large number of small categories. Rather than drawing a slice for each category, you can select a threshold value. Any category with less than the threshold value will be moved into "Others" slice.

**Display Name for 'Others':**

This option allows you to set the display name for the "Others" slice that is created for categories that fall below the set threshold value. This label will appear in the legend and/or for the slice label.

**Label:**

This option determines whether a category/series label should be drawn for each pie slice. By default, it only appears as legend.

**Value:**

This option allows you to specify whether to display the actual value of each pie slice.

**Percentage:**

This option allows you to display the percentage for each pie slice. The percentages are calculated by dividing the value of each section by the sum of all the values.

**Percentage in Legend:**

This option allows you to display the percentage in the legend represented by each slice in the pie. In case a pie slice is too thin, this can be a preferable presentation. This option is only available if the pie chart does not have a data series.

**Border:**

This option specifies whether to draw a border around each pie slice.

**Slice-to-Label Line:**

This option will draw a line from any label(s) to its corresponding pie slice.

**Label At Side:**

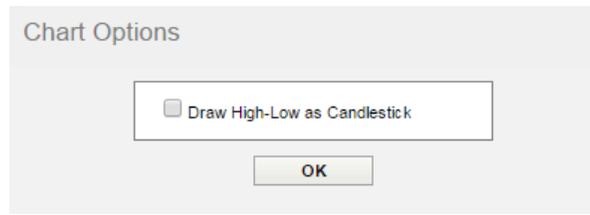
This option will place the labels for a pie chart away from the plot around the outside of the chart. When used with the "Slice-to-Label Line" option,

it gives you a way to display the pie labels for charts with many small categories without any text overlapping.

- Explode Others:** This option allows you to draw the “Others” segment at a certain distance from the center of the pie chart.
- Best Fit:** This option will arrange multiple pies in best configuration to fit the chart canvas. It's only available for pie charts with data series.
- Draw Linearly:** This option will arrange multiple pies in a straight horizontal line. It's only available for pie charts with data series.
- Number of Pies Per Row:** This option allows you to create a custom arrangement of multiple pies by specifying the number of pies to be drawn in each row. It's only available for pie charts with data series.
- Gap Between Pies:** This option allows you to specify a gap between multiple pies. The number represent multiple of the pie radius, so the gap will adjust with the size of the chart plot. It's only available for pie charts with data series.

#### 4.4.5.5.3. HLCO Charts

The following dialog is opened for HLCO charts:

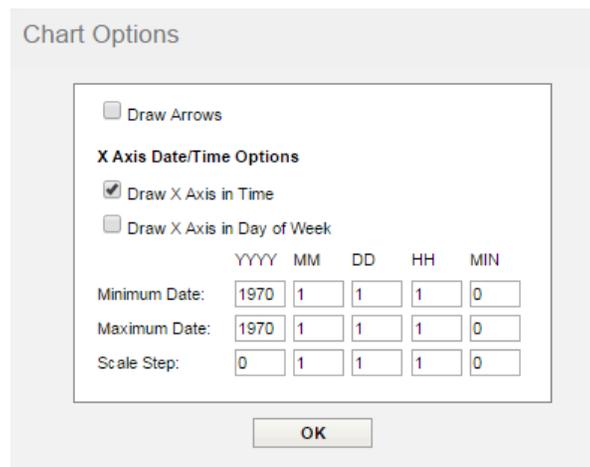


*HLCO Chart Options Dialog*

The *Draw Hi-Low As Candle Stick* option will turn the HLCO chart into a candle representation. A candle HLCO chart blends high, low, close, and open data into a single object that resembles a candlestick.

#### 4.4.5.5.4. Gantt Charts

The following dialog is opened for Gantt charts:



*Gantt Options Dialog*

The following options are available for Gantt charts:

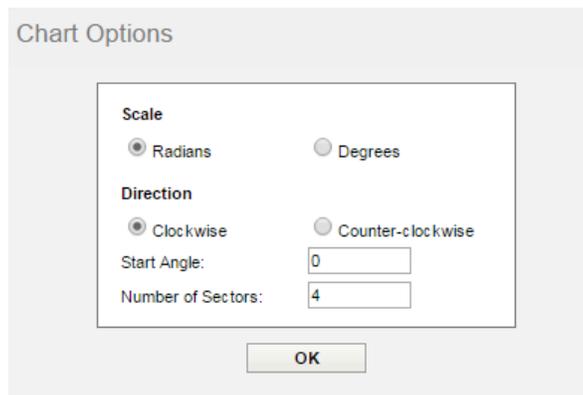
- Draw Arrows:** This option will draw connecting arrows between category elements of the Gantt chart. This allows you to illustrate a sequence between scheduled

events. The arrows are drawn in the order the category elements appear in the data source.

- Display X-Axis in Time:** This option shows the ticker labels as time values instead of numeric values for the X-axis.
- Display X-Axis in Day of Week:** This option shows the ticker labels as days of the week with the date for each Sunday shown as well.
- Minimum Date:** This option specifies the beginning date for the X-axis. The format is year, month, day, hour, and minute.
- Maximum Date:** This option specifies the end date for the X-axis. The format is year, month, day, hour, and minute.
- Scale Step:** This option specifies the scale step for the X-axis. The format is year, month, day, hour, and minute.

#### 4.4.5.5. Polar Charts

For polar charts, the following dialog is opened:



*Polar Chart Options Dialog*

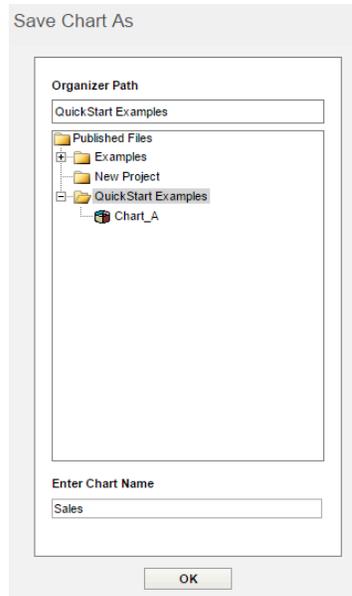
The following options are available for polar charts:

- Scale:** This option allows you to specify whether the input data for the angle ( $\theta$ ) portion of the data points is in radians or degrees. The chart will always display angles from 0 to 360. If the input data is in radians, it will be displayed as degrees.
- Direction:** This option allows you to specify whether the circular plot should be drawn clockwise or counter-clockwise.
- Start Angle:** By default, the top of the polar chart plot is 0 degrees. This option allows you to specify a different angle for the top of the plot. It can be displayed either in degrees or radians, depending on the scale you chose.
- Number of Sectors:** This option allows you to select the number of sectors you want to show in the chart. Sectors are created by drawing additional polar axis lines at specified angle intervals. By default, four sectors are shown.

#### 4.4.6. Save the Chart

Once you create a chart in QuickDesigner Charts, you can share it with other users by saving it on the server and inserting it into the Organizer.

To save a chart, click on the  *Save* (or  *Save Chart As* for saving the file under a different name that is already saved) button on the toolbar. A dialog will open in the left pane.

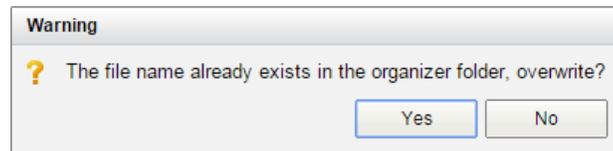


*Save Chart Dialog*

Here you can see all Organizer projects and folders you have access to. Select the project or folder you want to save the chart into. Specify a name for the chart at the bottom of the dialog. The chart will be saved with .qch extension.

Once you finish specifying the options, click on the *OK* button to save the chart. The new file will be saved in the ChartFiles folder in the ERES installation directory as ChartNameInOrganizer\_Username\_TimeStamp.qch. This should help to avoid unintentional overwriting of chart files.

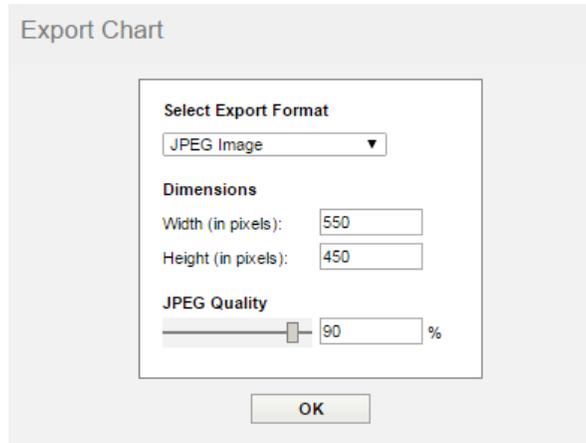
If you try to save a file with the same name as another file, you will be asked to confirm overwriting it or using a different name. If you wish to save the changes in the currently opened chart, do not change the chart name and the Organizer folder in the Save Chart dialog. Click *Yes* in the dialog and the chart file will be updated.



*Chart Overwrite Dialog*

## 4.4.7. Export the Chart

You can export a chart in a number of different formats. To do so, click on the  *Export* button on the toolbar and a dialog will open in a new window prompting you to specify the export options.



*Export Chart Dialog*

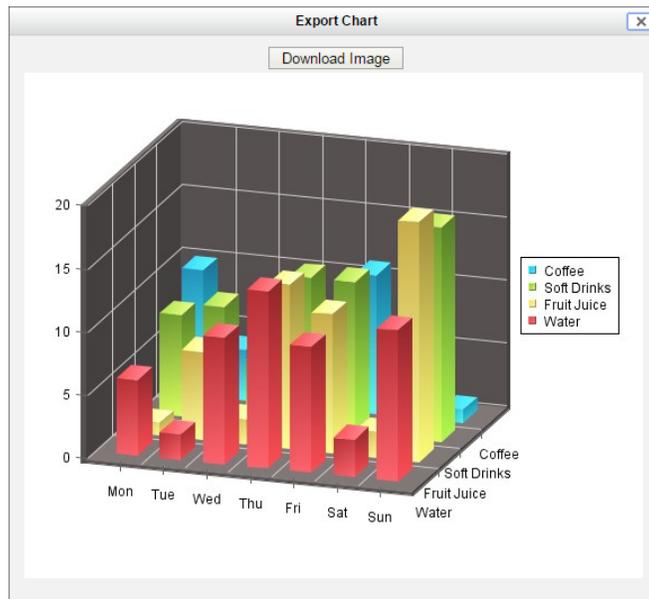
The first option allows you to specify the format in which to export the chart. The available formats are GIF, JPEG, PNG, BMP, PDF, SVG, SWF (Flash), Text, XML, and Chart Image Map.

The second option allows you to set the dimensions of the exported image. By default, these will match the canvas dimensions of your chart.

The last options allow you to set some image type-specific options. (The options appear only for appropriate format.) You can set the background transparent for GIF files (please note that the result is not good for more than 256 colors - e.g. with Color Gradient), set the quality for JPEG files, as well as specify compression for PNG files.

For more information about the chart export options, please see Section 4.2.6.3 - Exporting Charts.

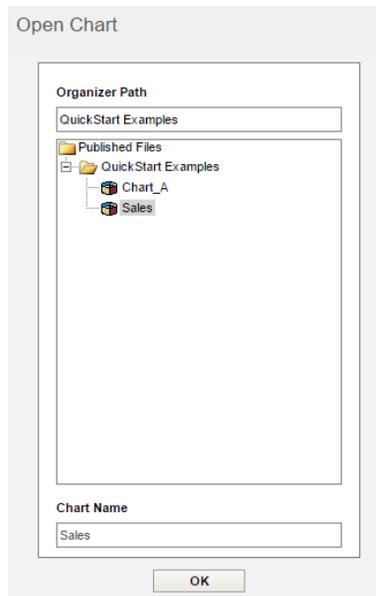
Once you finish specifying the options for the exported chart, click on the *OK* button and a new window will open containing the exported chart. You can save the generated file to your local system by clicking on the *Download Image* button.



*Exported Chart*

### 4.4.8. Open the Saved Chart

You can open a saved chart by clicking on the  *Open* icon on the main toolbar. The *Open Chart* dialog will appear.



*Open Chart Dialog*

Here you can see all charts created in QuickDesigner Charts. Select a chart and then click on the *OK* button to open it.

## 4.4.9. Exit

You can exit QuickDesigner Charts either by clicking on the  *Home* icon in the upper right corner, clicking on the *QuickDesigner Charts* title, or by clicking on the  *Logo* icon in the upper left corner. Before closing, you will be asked if you want to save an unsaved chart.

---

# Chapter 5. Designing Maps

## 5.1. Introduction to ERES Maps

ERES Maps are designed to report geographical data from data sources. They fetch geographical data from data source and mark them on a map. There are two types of maps: Online Maps and SVG Maps.

**Online Maps** The **Online Maps** feature in ERES displays geographical data using high quality on-line maps made by Google and Open Street Maps. Online Maps provides street maps and satellite maps, spanning over the entire world. By using high quality satellite imagery, a very wide zoom range is possible, making it possible to display every imaginable area of the world. Online Maps are useful for displaying exact spots on the maps. These spots are called **map points** and they are marked with **map markers**. A powerful tool provided by ERES Online Maps is **Geocoding**. Geocoding can locate buildings/cities on map by knowing only their addresses. Usage of Online Maps is limited by Online Maps Terms of Service (To see the current version, visit Google Maps API Terms of Service [ <https://developers.google.com/maps/terms.html> ], Open Street Maps API Terms of Service [ <https://www.openstreetmap.org/copyright/en> ]).

Online Maps can contain tooltips and drilldowns. **Tooltips** are small reports or charts which are displayed on mouse over a map marker. They display brief summary information related to the selected map point. **Drilldowns** are links to parameterized charts, reports or other maps that open when you click on a map marker. They are usually used to display more detailed information about the selected map point.

**SVG Maps** **SVG Maps** display data on SVG (Scalable Vector Graphics) maps. They differ from ERES Online Maps in two ways. First, SVG Maps work with geographical **areas** (e.g. a state is marked by the whole territory, not just by a point in the middle of the state). These areas can be colored according to set criteria. Second, the SVG Maps feature requires **SVG images** as the map source and do not use any online map source. The SVG Maps feature may be used to display spatial (non-geographic) and customized maps (e.g. maps of buildings, floor plans, mine maps, etc.).

## 5.2. Online Maps

ERES Online Maps use two types of files: Coordinates and Map Files. **Coordinates** contain location of individual map points (e.g. cities, company branches). Coordinates have to be created before creating a map. One set of Coordinates can be shared by several maps, so you do not have to store the same file several times. It is enough to add a map point in the Coordinates and it is added to all maps that use these Coordinates. For more information about Coordinates, see Section 5.2.4.2 - Create Coordinates. **Map File** displays map markers based on Coordinates and defines Drilldowns, Tooltips, and Heatmaps.

### 5.2.1. Generating Google Maps API Key

It is possible to use ERES Maps without Google API Key if you don't select either Google Street Map, Google Satellite Map or Geocoding.



#### Tip

You can use Open Street Maps even without the API key

Open Street Maps are a free alternative to Google Maps. Select Open Street Map in the Map Options dialog if you don't need Google Maps.

Since the time Google Maps were embedded into our products, Google has change their ToS. To keep the ERES Maps compatible, our developers added the option to use your Google API Key in ERES.

To use Google geocoding and Google Maps, you have to get the correct Google API key set up.

All the steps necessary for enabling Google Street Map, Google Satellite Map or Google Geocoding:

1. If you don't have a Google Cloud Platform account, create one <https://cloud.google.com/console/google/maps-apis/overview>
2. Log in To Google Cloud Platform here <https://cloud.google.com/console/google/maps-apis/overview>
3. Create a new project there
4. Select the project
5. Select APIs & Services → Credentials
6. Create credentials → API key

Then enable the services you need.

For using Google Maps without geocoding, you have to have “Maps JavaScript API” enabled

For using Geocoding, you have to have both Maps “JavaScript API” and “Geocoding API” enabled

### **To enable Maps JavaScript API**

1. Click *APIs*
2. Click *Maps JavaScript API*
3. Click *ENABLE*

### **To enable Geocoding API**

1. Click *APIs*
2. Click *Geocoding API*
3. Click *ENABLE*

### **To check quotas and billing:**

- Click *APIs* again
- In the list of enabled APIs, select the *Geocoding API*
- Click *Quotas*
- Scroll down to *Requests*
- Find “Requests per day” and click the pencil *Edit* icon
- As you can see, you cannot increase the limit for more than 1 request per day unless you enable billing
- Click *Enable billing for this project*.
- Then you can either use an existing credit card information saved in Google or create a new one.

## **5.2.2. Data Sources**

Each ERES Online Map uses data from two data sources - the first one is used for Coordinates and the second one is for the map itself. This chapter explains where and how these data sources are used.

**The Coordinates data source** is used when you are creating Coordinates. It is used for obtaining all the Point IDs. The **Point ID** is an identifier of a place (e.g. city name, branch name etc.) and it is loaded from the data source where you selected to create the Coordinates. The Point ID can consist of several **Point ID fields** - each data source column creates one field (except the fields which contain longitude, latitude, or WKT). For example, if your data source contains two columns - City and State, the Point ID will consist of two fields - City and State.

Each map point should have its unique Point ID, so any map point can be distinguished from the others. The data source should be chosen wisely to fulfill this condition (each row of data should be unique, there must not be duplicates).

It is recommended not to use parameterized data source for Coordinates, because there is no way to set parameter values for Coordinates. If you use a parameterized data source for Coordinates, the default parameter values are always used. This does not apply to multivalued parameters which use all the values (e.g. if you use CustomersInStates HSQL query (Databases/Woodview/Query/CustomersInStates) to create coordinates using geocoding - all param values (states) will be used).

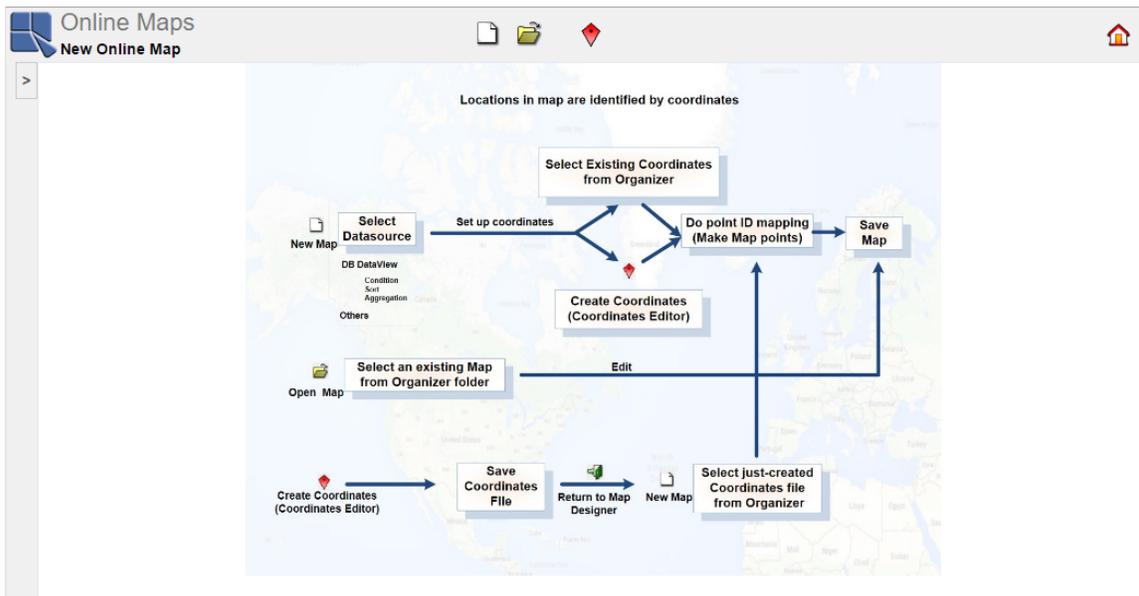
The Point ID is used for mapping map markers to the map data source in the Map File, so it should contain data which occurs in the map data source and which uniquely identifies map points. For example, if you want to display cities where your company has some customers, you should use City name and State name as the Point ID fields. The State name is useful in case when there are more cities with the same name in several different states. Optionally, you can add more Point ID fields for more detailed specification of city. For example if there are two cities with the same name in the same state, you should use more Point ID field(s) which will distinguish between these two cities, a ZIP code for instance.

The Coordinates data source can also contain coordinates (longitudes, latitudes, or WKT) of the map points. The longitudes and latitudes have to be stored in two separated columns of double data type, while WKT is a text field. These columns are not used as Point ID fields. If the coordinates are read from the data source, they are fetched every time the map is generated, so if there is a change in the Coordinates data source, it will be reflected in the map immediately. If the coordinates are not read from the data source, they are stored directly in the Coordinates file. In this case, the Coordinates data source is accessed only when you are creating or editing the Coordinates in the Coordinates Editor. When you are creating or displaying a map that is using this Coordinates, the data source is not accessed at all, because all the necessary information is in the Coordinates file, so it is not necessary for this Coordinates data source to be available when the map is created/displayed.

**The map data source** has two purposes. The first one is that it contains data that are used for Tooltips and Heatmaps. The second purpose is filtering. The map data source is mapped to the Coordinates as described in the Section 5.2.5 - Coordinates Mapping and only those map points from the Coordinates that have at least one associated row in the map data source are displayed. The map data source can be parameterized. In this case, the whole map is parameterized. If you create/open map that is using parameterized data source, there will be a filter icon in the map which opens parameters dialog that allows you to enter parameter values.

### 5.2.3. Start Online Maps

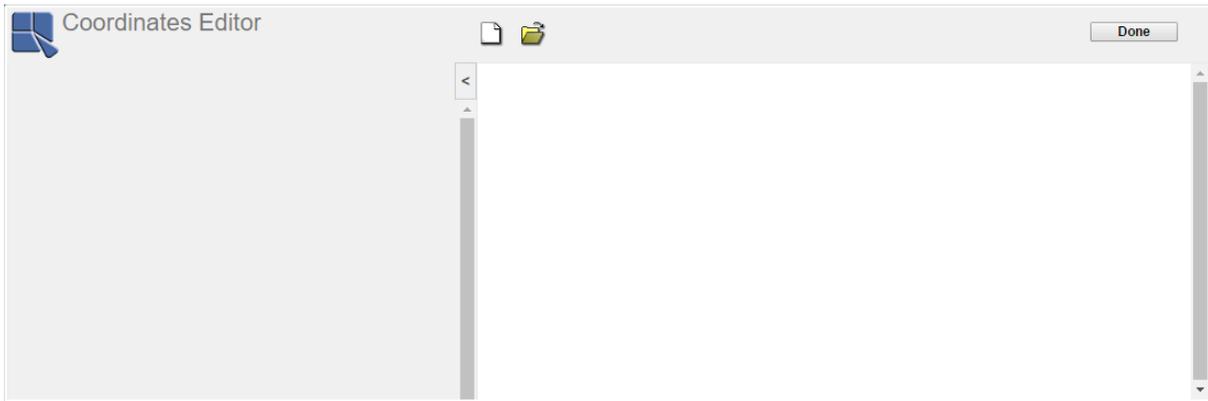
Online Maps can be started by clicking on the *Online Maps* link on the ERES Main page. After launching Online Maps, you will see a diagram that shows how to work with Online Maps. You can choose to create a new map, open an existing one, or open the Coordinates Editor where you can create/edit Coordinates.



Online Maps Start Options

## 5.2.4. Coordinates Editor

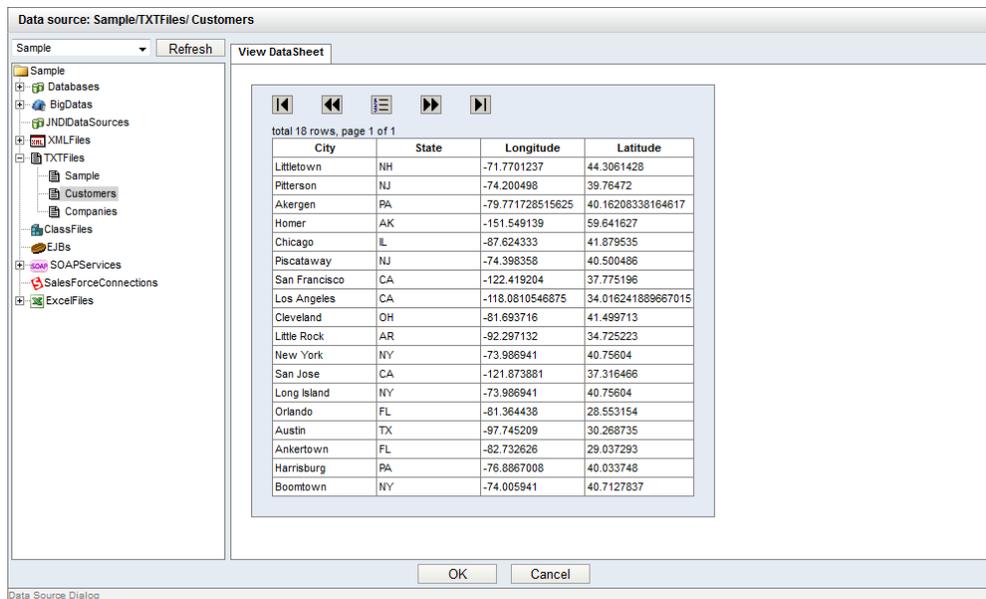
Click on the  *Coordinates Editor/Edit Coordinates* icon on the toolbar. The *Coordinates Editor* will open.



*Coordinates Editor Start Options*

### 5.2.4.1. Select Data Source

Open the *Data Source Dialog* by clicking on the  *New Coordinates* icon on the toolbar. Select the data registry in the upper left corner of the dialog and then select the desired data source (For more information about managing data registries, visit Section 3.1.1 - Managing Data Registries). A view of the data will appear in the right pane, which will allow you to check whether you chose the right data source. Click on the *OK* button to close the *Data Source Dialog*.

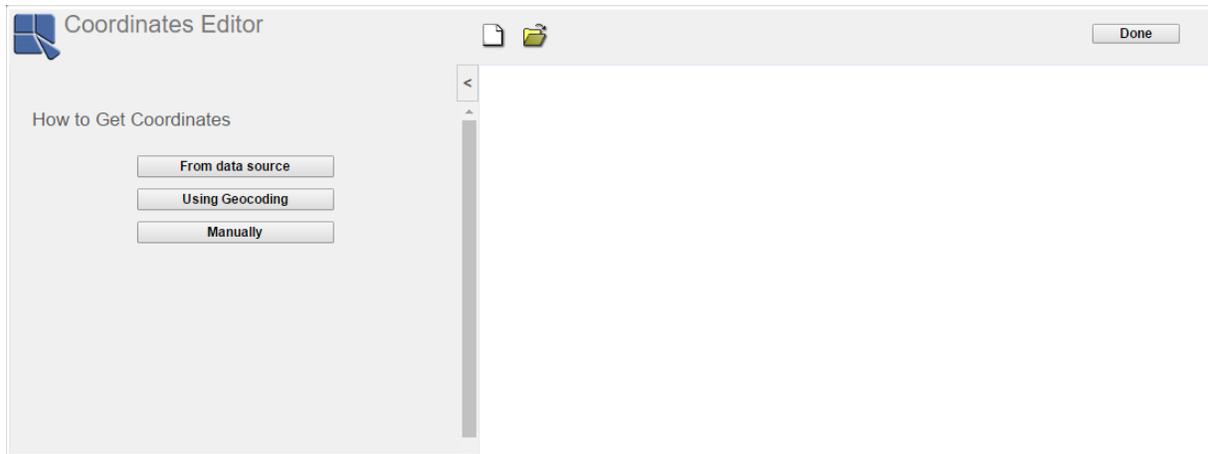


*Data Source Dialog*

### 5.2.4.2. Create Coordinates

The **Coordinates** file contains the coordinates of map points. Each Coordinates data record consists of WKT (obtained automatically or manually) and Point ID (for more information about Point ID, see Section 5.2.2 - Data Sources). Note: Existing coordinates files or data sources with longitude and latitude columns are still supported, but only WKT will be generated by geocoding or manual entry.

After selecting the data source, you will be prompted to select a method for obtaining Coordinates from the data source.



*Get Coordinates*

There are three ways how to get Coordinates, which are described below. Status of each pointID is determined by colors in the pointID table (For more information about colors, see Section 5.2.4.3 - Coordinates Editor Interface):

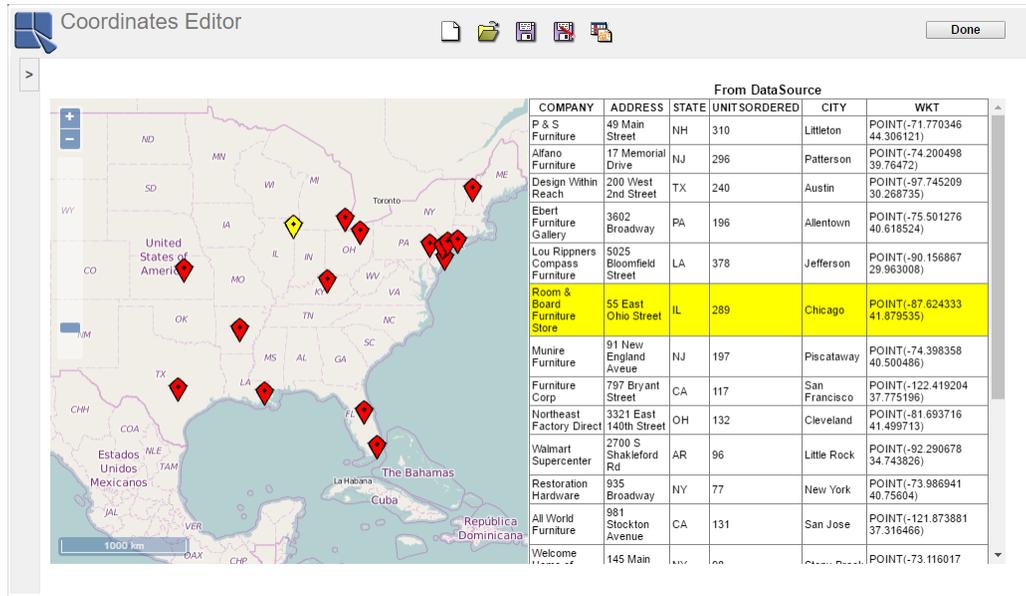
#### 5.2.4.2.1. From Data Source

This option requires two numeral data source columns containing information about the latitude and longitude or WKT of particular map point. Latitude and longitude have to be in different columns. The data type of these columns should be double and they should contain latitude and longitude in decimal degree notation. Positive longitude means East longitude, negative means West. Positive latitude means North latitude, negative means South.

In this case, map point latitudes and longitudes are read from the data source every time a map, which is using this Coordinates, is open. If the data source changes, the change is reflected in the map (e.g. if a new map point is added, it will be displayed on the map). It is not necessary to edit the Coordinates, updating the data source is good enough.

If Geocoding or Manual insertion of the map point coordinates is used, it is necessary to edit the Coordinates every time you want to add or modify (move) a map point.

If the Coordinates are obtained from a data source, all map points will be placed automatically according to the data from the data source. These rows will have white color. In this case, the Coordinates are read-only - you can only check which map marker belongs to which Point ID by holding mouse over a map marker/Point ID. The associated Point ID/map marker will be highlighted with yellow color. If you want to change the position of a map point, you have to modify the data source directly (i.e. change the longitude and latitude or WKT in the appropriate data source column).



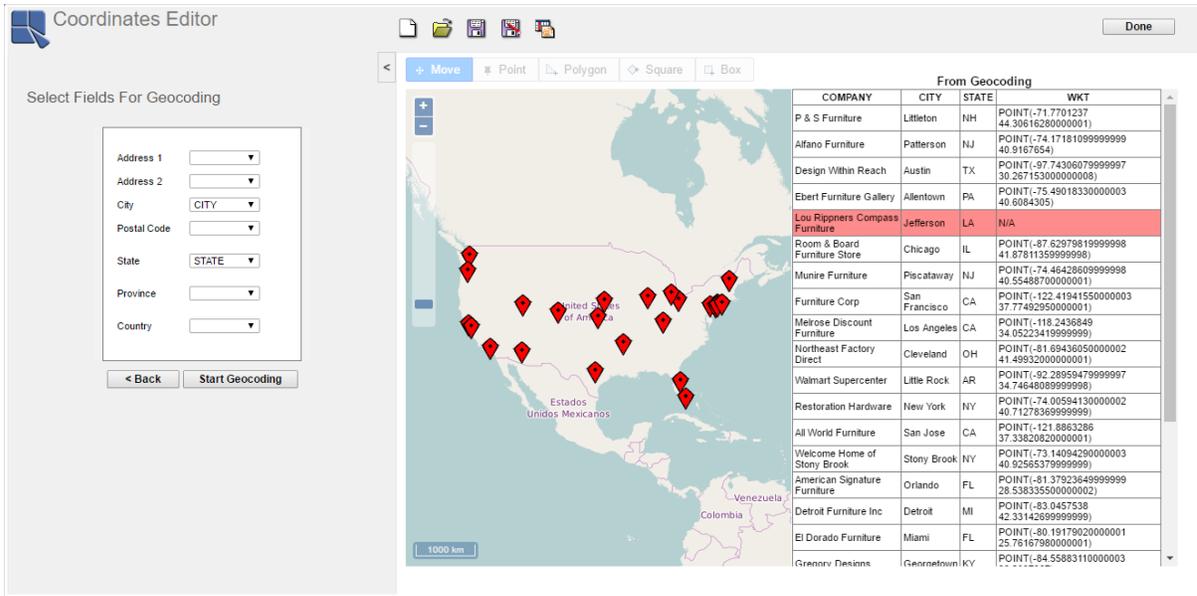
Coordinates from Data Source

### 5.2.4.2.2. Geocoding

Positions will be determined automatically using point addresses. The addresses can consist of several data source columns. You can select columns, which contain Address 1, Address 2, City, Postal Code, State, Province, and Country. Some of the fields may already be pre-selected when there is a convenient column in your data source (e.g. if the data source contains a column called *City*, it will be pre-selected in *City* field). You do not have to fill all the fields. If you do not fill any field, you will be automatically switched to manually option (see the next chapter). The name of the Geocoding fields are only illustrative, they do not have to be filled with exactly this information. The only rule you should keep in mind is to use the more specific information (e.g. street address) before the less specific (e.g. state). If your data source column contains more than just one field, use the first suitable field (e.g. if there is column with full address, you should use Address 1 field and leave the rest empty). Geocoding is very adaptive as it does not require any specific format of addresses and is able to find even inaccurate or incomplete addresses.

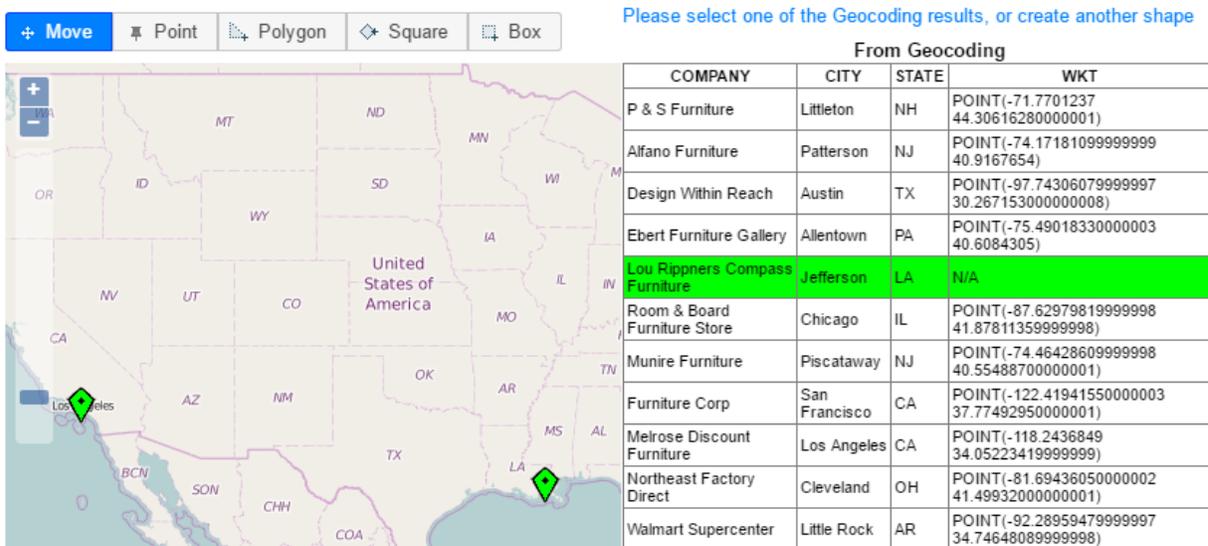
Please note that Geocoding may not be supported for all countries. To see the list of currently supported countries, visit Google Maps Coverage [ <https://developers.google.com/maps/coverage> ] and find your country in the list.

If you selected Geocoding, the Geocoding process is started automatically. It goes through all Point IDs and tries to determine positions of the corresponding points from their addresses. If it finds exactly one result, the map point is inserted automatically. The new point and the corresponding Point ID will turn yellow for a moment and then it will turn white, which indicates that the position is already determined. The toolbar is disabled during Geocoding, so you have to wait until it is finished before performing other actions. You can watch the Geocoding progress on the status bar. After the Geocoding is done, you should insert points that were not inserted automatically. These points had either no Geocoding results or more than one Geocoding result. The process is same as inserting the point manually and it is described below.



Coordinates by Geocoding

If you used Geocoding and you insert the rest of the points manually, the Point ID row may turn green instead of dark red after you click on it. This occurs when there is more than one Geocoding result. These results are displayed on the Online map with green markers so you can select the right one by clicking on it. If the desired map point is not among the Geocoding results, you can still add the map point manually on the correct place in the same way as manual inserting. If you hold mouse over a Geocoding result (green map marker), the full address of this map point is displayed to help you identify the correct place.



Selecting Geocoding Result

Please note that the Geocoding is not 100% accurate, so you should check all results that were inserted automatically.

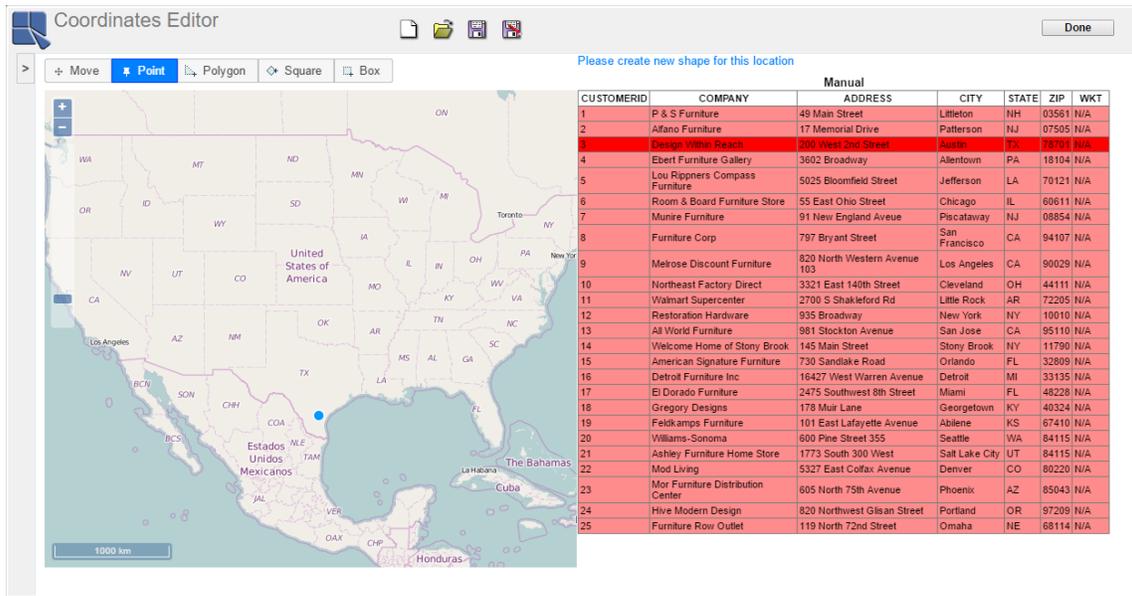
### 5.2.4.2.3. Manually

This option does not require any geographical data at all. Coordinates of map points are determined manually by clicking on the map.

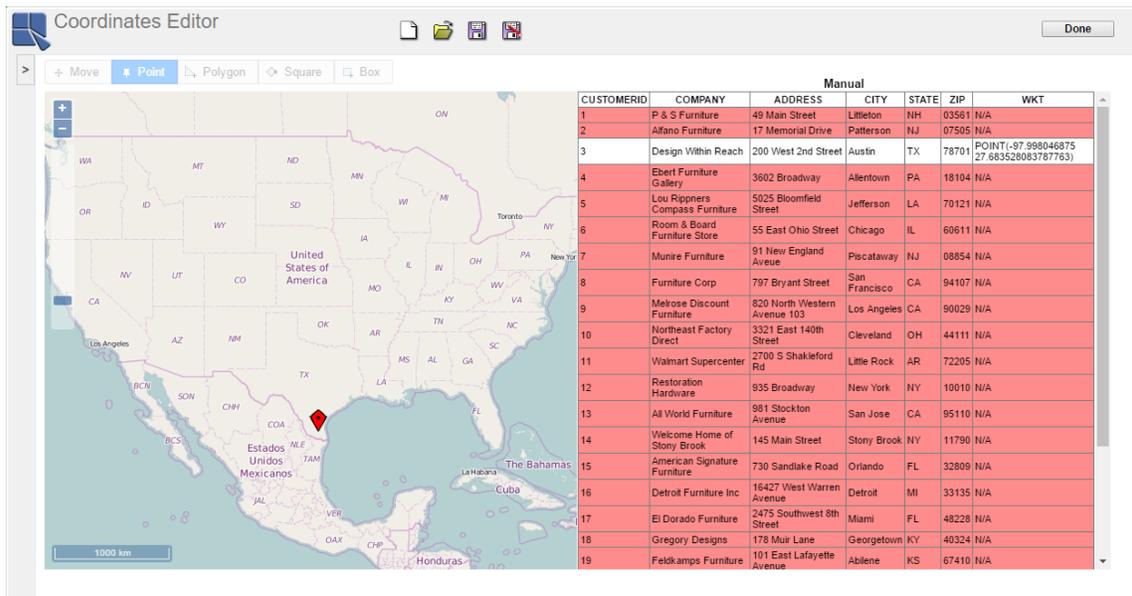
In the *Manually* case, all the Point IDs are light red at the beginning, which means that their position is not determined. To place a map point, click on a Point ID (row in the table) first. The Point ID row will now turn dark red. *Point* option should be active as default on the map toolbar (*Point* is blue). Mouse over the map and you will see a small blue circle under your mouse pointer. Then click on the location where you want to place your map point.

## Designing Maps

A new map marker will be placed on the selected map point and the appropriate row will become white. Then you can place another map point in the same way. To freely navigate across the map, select *Move* option on the toolbar. Now you can move and zoom the map any way you need. You can also use another shapes to define a location on the map - *Polygon*, *Square*, and *Box* on the map toolbar. To use these options, select the shape, place the first point on the map by clicking the mouse, move the mouse and place another point(s).



*Selecting Location*



*Selected Location*

The following images show the formation of shapes (blue color) and their final form (red color).



**Creating Box**

Please create new shape for this location

COMPANY	CITY	STATE	WKT
P & S Furniture	Littletown	NH	POINT(-71.78887367248535 44.30887438271947)
Aziano Furniture	Patterson	NJ	POINT(-74.17181099999999 40.9167654)
Design Within Reach	Austin	TX	POINT(-97.74336079999997 30.267153000000008)
Ebert Furniture Gallery	Abertown	PA	POINT(-75.49018330000003 40.6084305)
Low Rippers Compass Furniture	Jefferson	LA	POINT(-90.1531298 29.966037099999994)
Room & Board Furniture Store	Chicago	IL	POINT(-87.6297104358673 41.87795208845544)
Munire Furniture	Piscataway	NJ	POINT(-74.46428099999998 40.55488700000001)
Furniture Corp	San Francisco	CA	POINT(-122.41941550000003 37.77482050000001)
Makros Discount Furniture	Los Angeles	CA	POINT(-118.2436849 34.05223419999999)
Northeast Factory Direct	Cleveland	OH	POINT(-81.69436500000002 41.49932000000001)
Walmart Supercenter	Little Rock	AR	POINT(-92.28989479999997 34.74448099999999)

**Finished Box**

COMPANY	CITY	STATE	WKT
P & S Furniture	Littletown	NH	POLYGON((-71.78887367248535 44.30887438271947, -71.78887367248535 44.2964138927863, -71.74338340759276 44.2964138927863, -71.74338340759276 44.31887418271947, -71.78887367248535 44.31887418271947))
Aziano Furniture	Patterson	NJ	POINT(-74.17181099999999 40.9167654)
Design Within Reach	Austin	TX	POINT(-97.74336079999997 30.267153000000008)
Ebert Furniture Gallery	Abertown	PA	POINT(-75.49018330000003 40.6084305)
Low Rippers Compass Furniture	Jefferson	LA	POINT(-90.1531298 29.966037099999994)
Room & Board Furniture Store	Chicago	IL	POINT(-87.6297104358673 41.87795208845544)
Munire Furniture	Piscataway	NJ	POINT(-74.46428099999998 40.55488700000001)

*Box*

You can move any of the already existing map points by dragging the appropriate map marker with mouse. You can also resize a shape by dragging the edge of a side to any direction.

### 5.2.4.3. Coordinates Editor Interface

After choosing a method of obtaining coordinates, you will get a map with a point ID table.

City	State	Longitude	Latitude
Littletown	NY	-75.43212890625	42.27730877423709
Pitterson	NJ	-74.200498	39.76472
Aachen	TX	-100.3271484375	32.47269502206151
Akergen	PA	-79.771728515625	40.16208338164617
Sevilla	AK	-162.9052734375	61.938950426660604
Chicago	IL	-87.624333	41.879535
Piscataway	NJ	-74.398358	40.500486
San Francisco	CA	-122.419204	37.775196
Los Angeles	CA	-118.0810546875	34.016241889667015
Cleveland	OH	-81.693716	41.499713
Little Rock	AK	-156.5376	58.659738
New York	NY	-73.986941	40.75604
San Jose	CA	-121.873881	37.316466
Long Island	NY	-73.986941	40.75604
Orlando	FL	-81.364438	28.553154
Austin	TX	-97.745209	30.268735
Ankertown	FL	-82.732626	29.037293
George Park	PA	-76.320675	40.033748
Boontown	NY	-77.5634765625	42.50450285299051

*Coordinates Editor*

The following buttons are on the toolbar:



**New:** Creates new Coordinates.



**Open:** Opens existing Coordinates.



**Save:** Saves current Coordinates to a file and Organizer. Note that all saved files will include userID and timestamp in addition to the name you provided. If an Organizer item with the supplied name already exists in the folder, you will be given a choice to override the existing item or create a new entry.



**Save As:** Allows you to save the existing or modified Coordinates to a different file or location.



**Change Point Mapping:** Allows you to change point mapping.

The left part of the window displays the Online Map, which displays individual map points using the map markers. The Online Map can be scrolled and zoomed to find a specific object.

Point ID table in the right part of the window shows all the Point IDs, which were read from the data source, and longitude and latitude of map points. Each row represents one Point ID and can have one of the following background colors:



**White:** Location of the corresponding map point has been successfully determined.



**Light red:** Location of the corresponding map point has not been determined yet. There are multiple locations for the point. This is not static status color.



**Grey:** Location of the corresponding map point has not been determined yet. There is no location for the point. When you click this grey-colored row, it turns into dark red. This is not static color.



**Dark red:** Marks the point ID, which is currently under mouse cursor. If you click on it, it will either change into green (multiple locations), or grey (No location found), waiting for users to enter the location manually.



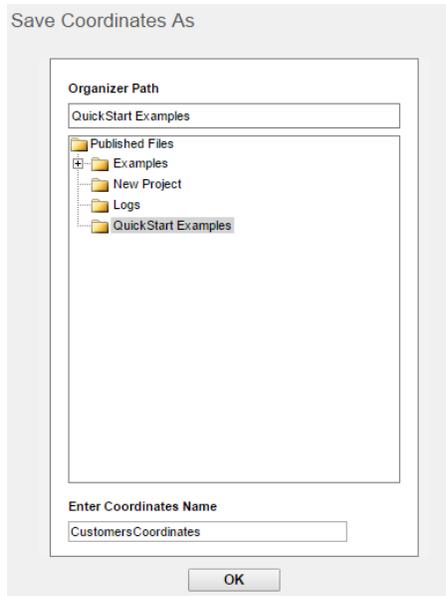
**Yellow:** Highlights the row with Point ID of the map point under mouse cursor on the map (this is used to determine which Point ID belongs to which map marker).



**Green:** Indicates that there are Geocoding matches for the selected Point ID.

#### 5.2.4.4. Save Coordinates

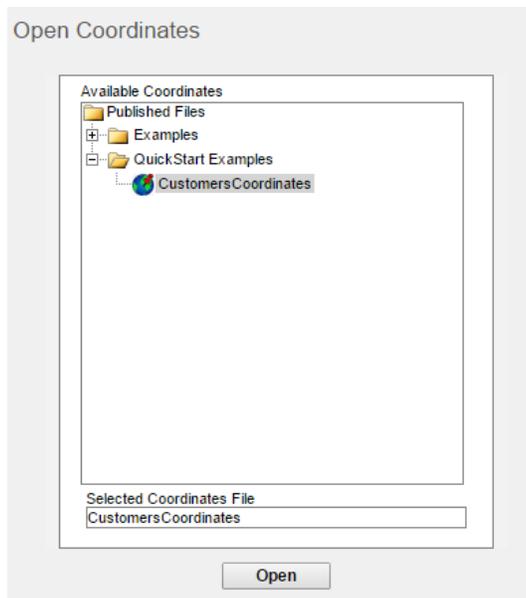
If you want to use the Coordinates for creating maps, be sure to save it to Organizer (click the  *Save* icon, enter a Coordinates name, and select any Organizer folder). Then click the *Done* button on the right side of the toolbar to close the Coordinates Editor and go back to Online Maps main window.



*Save Dialog*

### 5.2.4.5. Open, Edit Coordinates

To open a saved coordinates file, first open Coordinates Editor by clicking the  *Coordinates Editor/Edit Coordinates* icon on the main toolbar of the Online Maps designer. Then click the  *Open* icon on the main toolbar of Coordinates Editor.

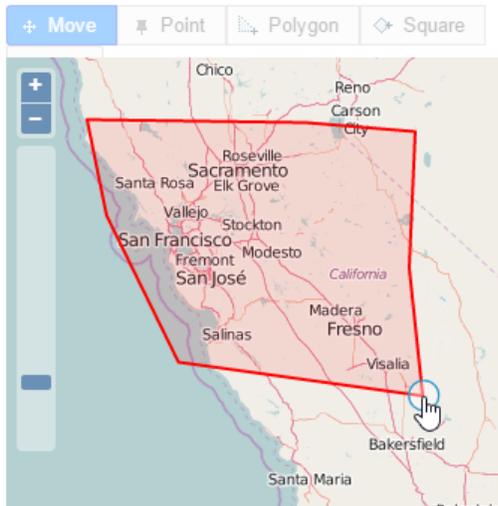


*Open Coordinates*

You can see all coordinates files from Organizer. Select a coordinates file and click the *Open* button.

Now you can edit coordinates obtained via geocoding or manual insertion. Coordinates obtained from data source are not possible to edit. You can change a point when you click the respective row in the pointID table - then you can either select again one of the geocoding results (applies only to geocoding coordinates) or replace a current point/shape with a new one in the map.

You can directly edit a map point using the drag and drop feature or edit a shape by stretching its sides.



Shape Editing

Please note that if the coordinates obtained via geocoding are not complete (not all rows are white), the geocoding process starts each time the file is opened.

Once you finish editing the coordinates, save the file and click the *Done* button to close the Coordinates Editor.

### 5.2.5. Coordinates Mapping

ERES Online Maps needs Coordinates to map the data source records to map points. When creating Online Maps, you have to map the Point ID fields from the Coordinates file to the map data source columns. According to this mapping, data source rows are associated with map points. There can be more data source rows associated with one map point but not vice versa (because Point IDs must be unique). The rows associated with a map point are used to generate Tooltips (see Section 5.2.6.7 - Tooltips) for the appropriate map marker.

Not all the Point ID fields have to be mapped. For example, if your Point ID has three fields - City, State, and Country, you can map only Country and State. But you have to keep in mind that any map point should be uniquely determined by the mapped Point ID fields. So if there are two Point IDs in your Coordinates, which differ only in City, but have equal Country and State, you have to use the City as well to distinguish between those two map points. You will get no error message if you use wrong mapping, but the data can be displayed incorrectly.

Some map points from the Coordinates may not have any associated rows. These map points are not displayed on the map using map markers. Only the map markers, for which some data was found in the data source, are displayed. So you can have Coordinates containing all the major cities in USA, but you can display only the one you are interested in. And you can use the same Coordinates for different map, which will show different set of cities, because the data source will contain data for different cities, so you do not have to maintain too many Coordinates files if you choose suitable Point ID fields.

**Example:**

Your data source contains the following data and you want to display data grouped by cities on a map:

Row #	City	State	Customer Name	Sales
1	San Francisco	CA	Company A	\$152,560
2	New York	NY	Company B	\$240,468
3	Los Angeles	CA	Company C	\$335,256
4	San Francisco	CA	Company D	\$80,381

Row #	City	State	Customer Name	Sales
5	Boston	MA	company E	\$23,540
6	Chicago	IL	Company F	\$124,532

You will have to create the Coordinates first. In this case, only the City name could be used as the Point ID because it is unique, but we will use two Point ID fields instead - City and State, in case that there will be more cities with the same name in different states in the future. So we will create a data source with two columns - City and State. This data source has to contain all the listed cities (and could contain some more). We can either write a suitable query to obtain these two columns from the original data source (if using a database for the data source) or we can, for example, create a simple text file data source with these two columns.

Next, we can start Coordinates Editor and create the Coordinates. Assuming we have a data source that does not contain data for longitudes and latitudes or WKT, so we cannot use coordinates from data source but we know part of an address, we will use Geocoding. Using Geocoding, we will assign WKT to every city, so the table in the Coordinates Editor will look like this (note that there is one extra city - Orlando, which is not in the map data source):

City	State	WKT
San Francisco	CA	POINT(-122.41941550000003 37.77492950000001)
New York	NY	POINT(-74.00594130000002 40.712783699999999)
Los Angeles	CA	POINT(-118.2436849 34.052234199999999)
Chicago	IL	POINT(-87.629798199999998 41.878113599999998)
Orlando	FL	POINT(-81.379236499999999 28.538335500000002)

Now you can create a map from the data source using these Coordinates. You map City Point ID field to the City data source column and State Point ID field to the State data source column. The Point IDs will be paired with the data source rows - San Francisco with the first and fourth, and Chicago with the sixth. Orlando does not have any matching row in the data source, so it will not be displayed at all. If you add a Tooltip, the associated rows will be used to generate it. For example, the Tooltip for San Francisco will be created using data from the first and fourth row of the data source.

If you add a new city to the data source and you want to display it too, you have to edit the Coordinates data source first and add the city there. Next, you can open the Coordinates in the Coordinates Editor and there will be one new row in the table with this new city. This enables you to add the new map point for this city. You do not have to modify anything in your map. Therefore, if this Coordinates is shared among several maps, the map point is added automatically to all these maps.

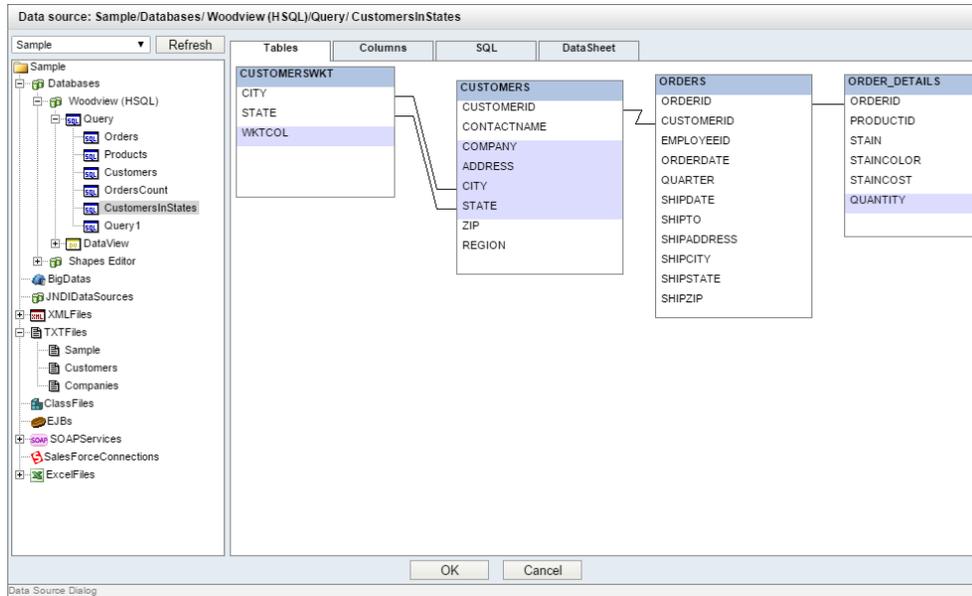
## 5.2.6. Create Online Maps

ERES Online map is a special kind of report that allows displaying geographical data on map. It points out the reported geographical data on a map using map markers. Markers can be used as drill-down links to parameterized reports, charts, or maps. Markers may also have tooltips. **Tooltips** look like bubbles, containing a report or chart. When tooltips are enabled, every map marker has its own tooltip bubble, which displays data associated to this map point. Only one tooltip bubble can be displayed at the time. A tooltip appears when you move your mouse cursor over a map marker. It closes automatically when you open a different tooltip, or it can be closed by clicking on the X sign in the upper right corner of the tooltip bubble.

To create a Online Map, launch Online Maps from the ERES main page.

### 5.2.6.1. Select Data Source

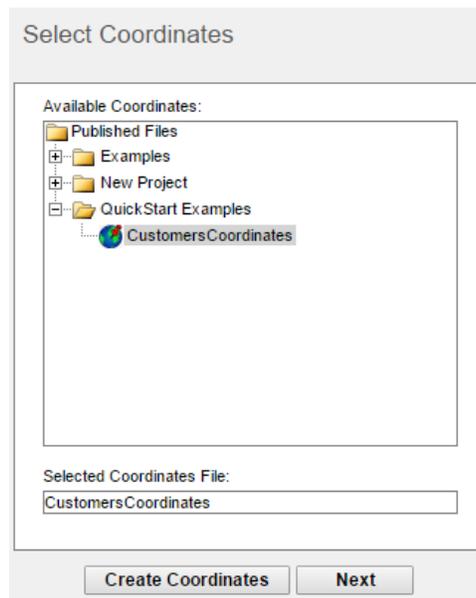
To create a new Online Map, click on the  *New Map* icon on the toolbar. The *Data Source Dialog* will appear. Select a data source for your map and click *OK*.



Data Source Dialog

### 5.2.6.2. Select Coordinates

Once you select a data source for your map, you will be prompted to select Coordinates file.



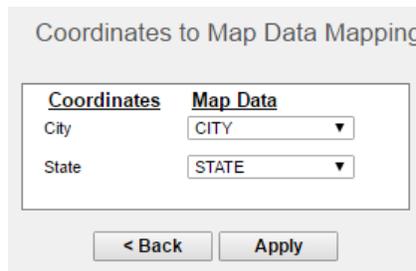
Select Coordinates

Select a Coordinates file and click the *Next* button. You can also launch the Coordinates Editor from this dialog by clicking the *Create Coordinates* button and creating a new coordinates file. Once the new coordinates file is saved, click the *Done* button in the upper right corner of the Coordinates Editor. You will be brought directly to *Coordinates Mapping* which is described in the next chapter.

### 5.2.6.3. Coordinates Mapping

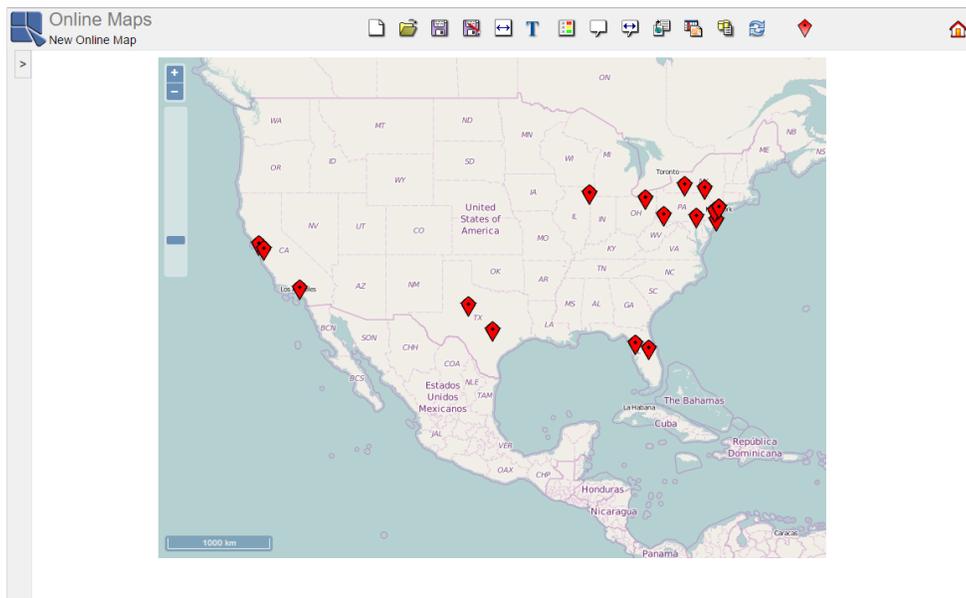
You will be prompted to select point mapping, i.e. you have to pair columns that will enable ERES Online Maps to assign records from maps record's data source to suitable Point ID fields from the Coordinates (e.g. if you want to show cities with your affiliates on map, your Coordinates may contain a column called *City* which is mapped to *Affiliates\_City* from the map's data source). For more information, see Section 5.2.5 - Coordinates Mapping. Please note that Online Maps designer automatically recognizes which columns are the most suitable for Coordinates

mapping. Therefore, some of the fields may already be pre-selected when there is a convenient column in your data source (e.g. if the data source contains a column called *City*, it will be pre-selected in *City* field).



*Coordinates Mapping*

You have to map at least one column. Then click *Apply* and the Online Map will appear.



*Online Maps Designer*

You can change the Coordinates mapping by clicking the  *Change Point Mapping* icon on the toolbar.

### 5.2.6.4. Toolbar

These buttons are on the Online Maps toolbar:



**New Map:** Creates a new Map.



**Open Map:** Opens an existing Map.



**Save Map:** Saves the current Map to a file and Organizer. Note that all saved files will include userID and timestamp in addition to the name you provided. If an Organizer item with the supplied name exists in the folder, you will be given a choice to override the existing item or create a new entry.



**Save Map As:** Allows you to save the existing or modified Map to a different file or location.



**Set Map Options:** Allows you to change the way the Online Map is displayed.



**Map Title Options:** Allows you to set map title and its options.



**Set Heatmap:** Allows you to color-code map markers.



**Tooltip Template:** Adds/Modifies report/chart templates used as tooltips.



**Tooltip Options:** Allows you to change dimensions of tooltip bubbles.



**Drilldown Options:** Configures drill-downs.



**Change Point Mapping:** Changes the point mapping.



**Change Data Source:** Changes the data source of current map. Note that changing data source will result in losing tooltip and drill-down settings.



**Refresh:** Refreshes the map data (if the data source changed).



**Coordinates Editor/Edit Coordinates:** Opens Coordinates Editor that allows you to create/edit coordinates.



**Home:** Closes Online Maps designer. Before closing, you will be asked if you want to save an unsaved map.

## 5.2.6.5. Map Options

To set map options, click on the  *Set Map Options* icon on the toolbar.

**Map Options**

**Dimensions**

Width (in pixels):

Height (in pixels):

**Zoom**

Minimal Zoom:

Maximal Zoom:

**Map Controls**

Zoom Control:

Map Type:

Map Scale Control:

*Map Options*

**Dimensions:** Allows you to set width and height of the map (in pixels).

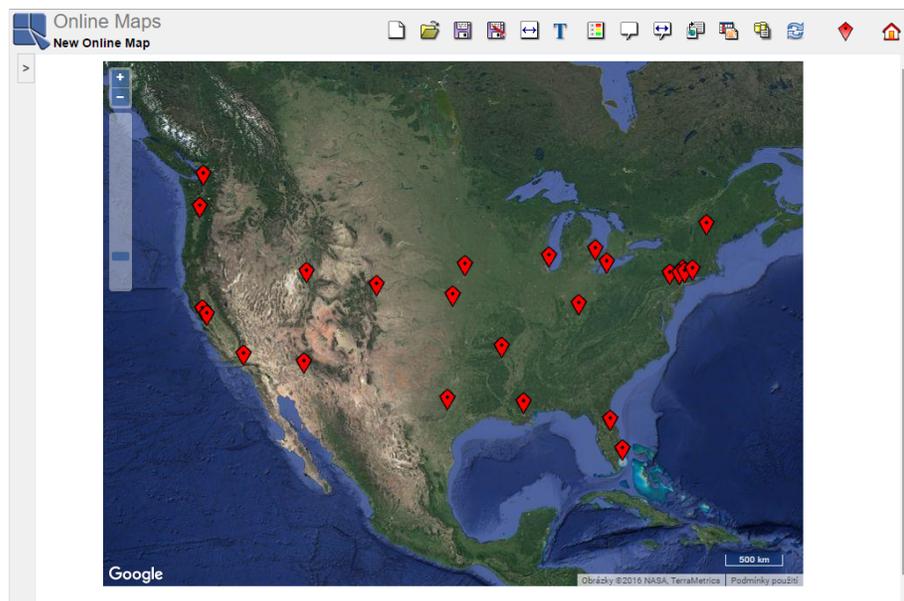
**Zoom:** Allows you to set minimal and maximal zoom of the map (in the range 1-17).

**Map Controls:** **Zoom Control** - shows/hides the zoom controls in the upper left corner (large, small, none).



*Zoom Control Large*

**Map Type** - allows you to set map type (Open Street Map, Google Street Map, Google Satellite).



*Google Satellite*

**Map Scale Control** - shows/hides the scale control in the lower right-hand corner.



*Scale Control*

### 5.2.6.6. Heatmap

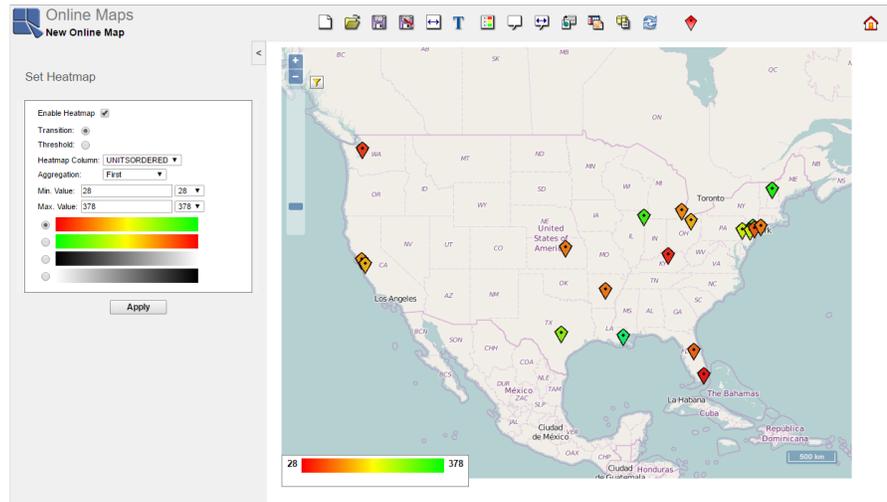
Heatmaps allows you to display map points in different colors based on values from a column in the data source. You can choose between two heatmap types - transition or thresholds.

Please note that some of the data sources have multiple records belonging to single map points in coordinates file. The *Aggregation* option allows users to choose a preferred way on how to aggregate values in the given heatmap column. The default aggregation value is set to *First* and this value is recommended only for maps where multiple records for one point in the map are not expected. In other cases, it is better to choose a more appropriate aggregation.

To set a heatmap, click on the  *Set Heatmap* icon on the toolbar. The *Set Heatmap* dialog appears in the left pane. At first check off the *Enable Heatmap* option and select a type of the heatmap - *Transition* or *Threshold*. The *Set Heatmap* dialog will be adjusted accordingly:

**Transition Heatmap:**

Select a *Heatmap Column* from the drop-down list, select an *Aggregation*, and set minimal and maximal value of the selected column values. You can also choose a value from the drop down lists which contain the precise values from the selected column. Select a color gradient and click *Apply* to apply your settings on the map.

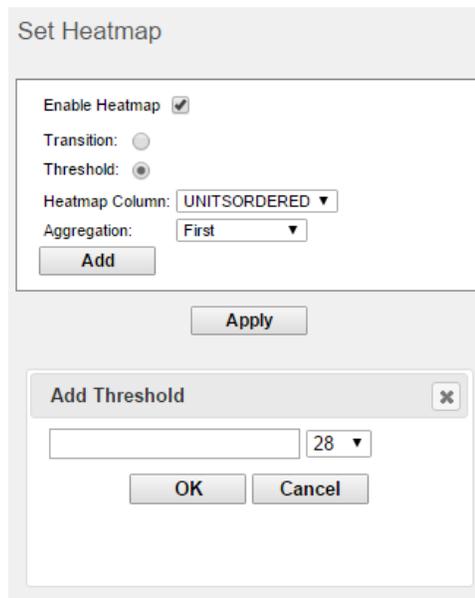


*Set Transition Heatmap*

You can enable/disable a heatmap by checking/unchecking *Enable Heatmap* option.

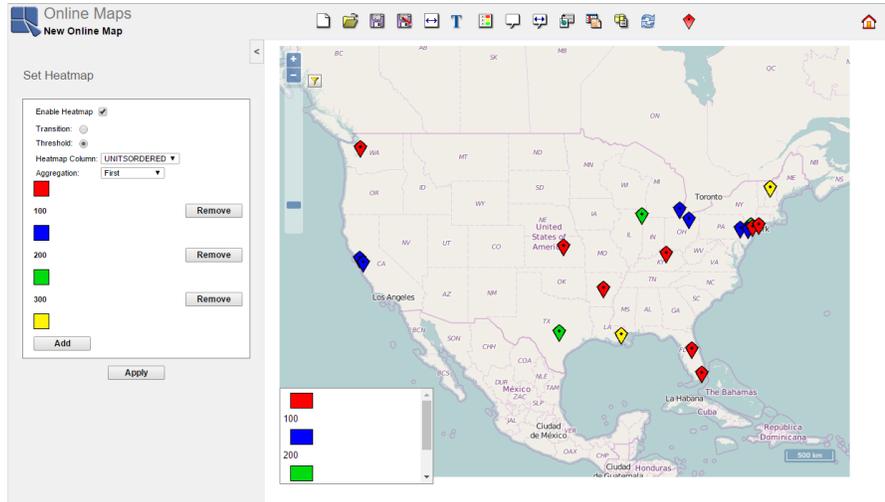
**Threshold Heatmap:**

Select a *Heatmap Column* from the drop-down list, select an *Aggregation*, and click the *Add* button to add a threshold. *Add Threshold* dialog will appear.



*Set Threshold Heatmap*

Type a threshold value or choose a value from the drop-down list which contains precise values from the selected column and click the *OK* button. Set another thresholds in the same way and set colors for each threshold by clicking on the color square and selecting the color. Then click *Apply* to apply the heatmap.

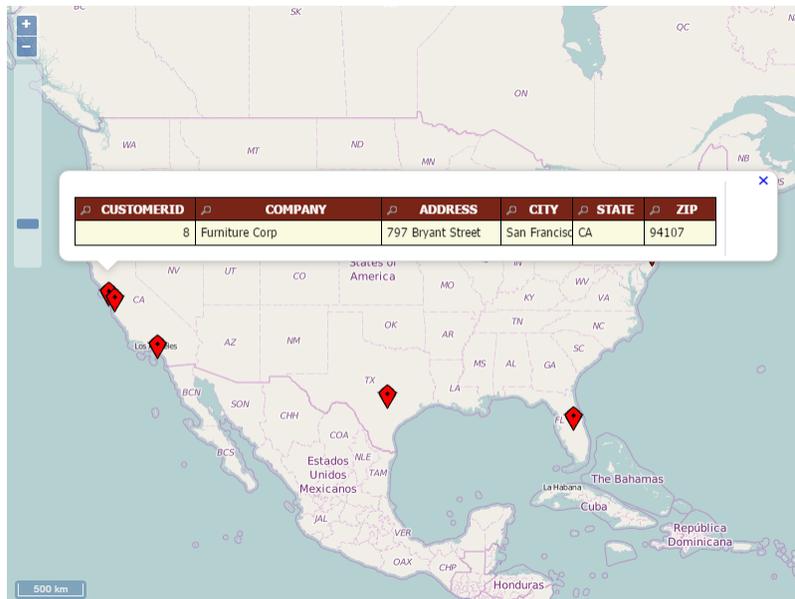


*Set Threshold Heatmap*

You can remove each threshold by clicking the *Remove* button next to the threshold. To enable/disable a heatmap, check/uncheck *Enable Heatmap* option.

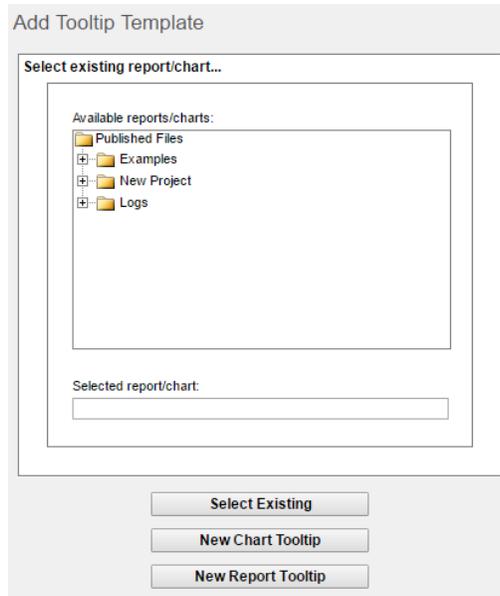
### 5.2.6.7. Tooltips

Tooltips may contain a report or chart that shows up when you move your mouse cursor over any map marker on the Online Map. Tooltips display single or multiple-row data that is related to the particular map point. As an example, a report tooltip will display data from the map datasource using parameter(s) from the *Map Data* of the corresponding map point. The *Map Data* is the same as the list in the *Change Point Mapping* dialog box. The report or chart file that is displayed in the tooltip bubble is called **Tooltip Template**.



*Online Map with Tooltip*

To enable tooltips for ERES Online Map, open the map in Online Maps and click on the  *Tooltip Template* icon. The *Add Tooltip Template* options will appear in the left pane.



*Add Tooltip Template*

You can select an existing tooltip, create chart tooltip, or create report tooltip.

**Select Existing:**

To select an existing Tooltip Template, select it from *Available reports/charts* list and click the *Select Existing* button. Using this option, you can also use the report/chart templates created in the Report Designer/Chart Designer, so you are not limited to use just the QuickDesigner Charts or QuickDesigner Reports to design Tooltip Templates. In this case, it is recommended to use exactly the same data source for creating this Tooltip Template report/chart as for the map, otherwise there might be a problem with data mapping.

**New Chart Tooltip:**

Launches QuickDesigner Charts that will allow you to create a new chart file. Cre-

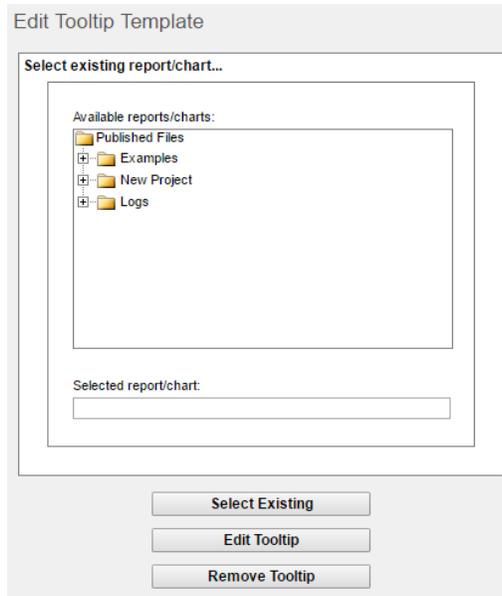
ate a chart, save it, and leave QuickDesigner Charts by clicking the  *Return to Map Designer* icon on the toolbar. (For more information about working in QuickDesigner Charts, see Section 4.4 - QuickDesigner Charts.)

**New Report Tooltip:**

Launches QuickDesigner Reports that will allow you to create a new report file.

Create a report, save it, and leave QuickDesigner Charts by clicking the  *Return to Map Designer* icon on the toolbar. (For more information about working in QuickDesigner Reports, see Section 4.3 - QuickDesigner Reports.)

To edit, select, or remove the current tooltip, click on the  *Tooltip Template* icon. The *Edit Tooltip Template* options will appear in the left pane. Please note that from this dialog, you can only directly edit files created in QuickDesigner Reports and QuickDesigner Charts (.qdr or .qch).



*Edit Tooltip Template*

**Select Existing:** Allows you to select other Tooltip Templates.

**Edit Tooltip:** Launches QuickDesigner Report or QuickDesigner Chart where you can edit the current tooltip template if it's a .qdr or .qch file (i.e. files from one of the QuickDesigners).

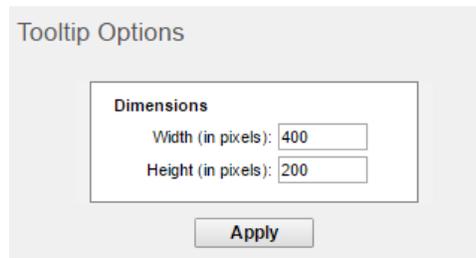
**Remove Tooltip:** Removes current Tooltip Template.



**Note**

To create a new chart/report tooltip you have to remove current tooltip at first and then will be *New Chart Tooltip* and *New Report Tooltip* options available.

To configure dimensions of the tooltip bubbles, click on the  *Tooltip Options* icon on the toolbar. Set new dimensions and click *Apply*.



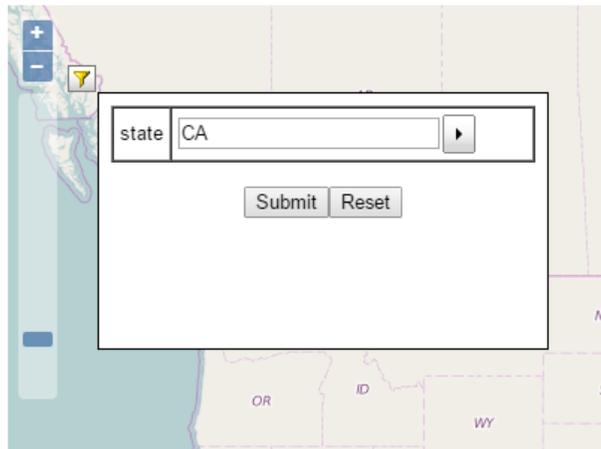
*Tooltip Options*

Changes will take effect after closing the current tooltip bubble (if a bubble is open, otherwise the changes will take effect immediately). It is not necessary to close the current tooltip bubble, you can just mouse over some other map point.

**5.2.6.8. Parameter Setting**

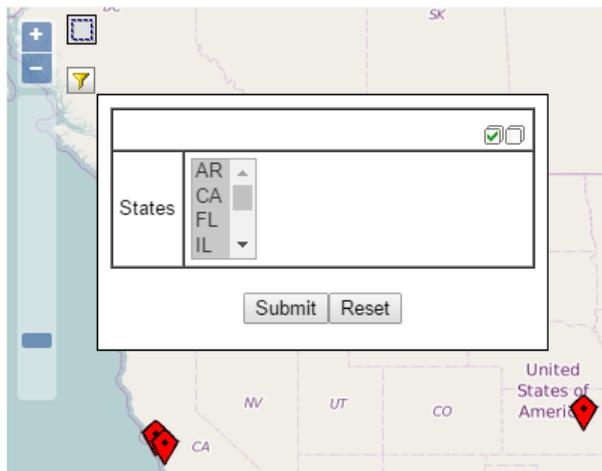
You can change parameters directly on the map by clicking the  *Set parameters* icon in the upper left corner of the map (The icon is displayed only if the data source is parameterized). A new dialog will open where you can select parameter value(s).

If the parameter is single-value, the next dialog will display a drop-down list. Select a parameter value and click the *Submit* button to apply it on the map. (You can reset your selection by clicking the *Reset* button.)



*Parameter Setting*

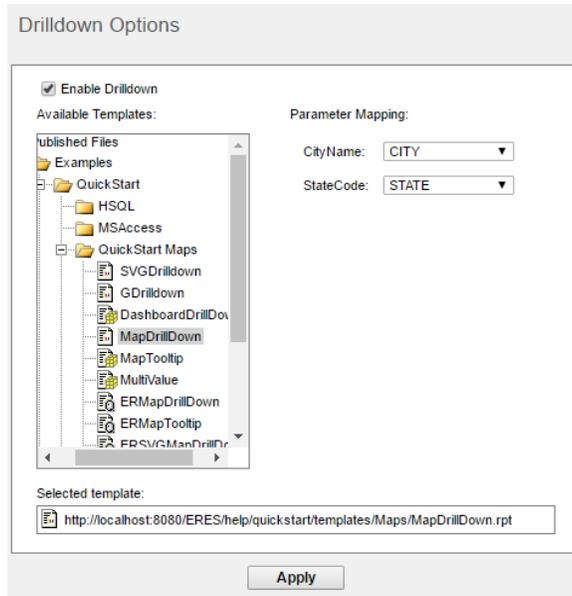
If the parameter is multi-value, the next dialog will display a list of parameter values. You can select parameter values by clicking on them (use **Ctrl+click** for multiple selection), select all parameter values by clicking the , deselect all parameter values by clicking the , and reset your selection by clicking the *Reset* button. Then apply your selection to the map by clicking the *Submit* button.



*Parameter Setting*

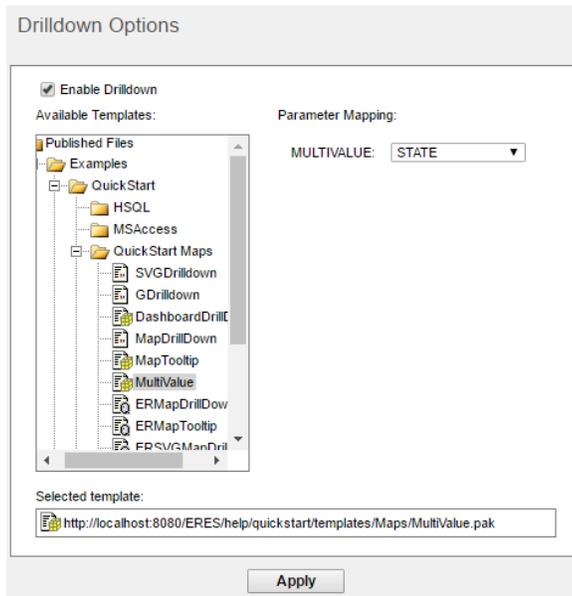
### 5.2.6.9. Drilldowns

To enable drilldowns in a map, click on the  *DrillDown Options* icon. Check the *Enable Drilldown* checkbox which will show you all parameterized reports, charts and maps saved in Organizer. Click on a suitable parameterized report, chart, or map and select parameter mapping. All drilldown parameters have to be mapped to some Point ID field from the Coordinates. Mapping in this case means that the Point ID values of the selected map point will be used as drilldown parameter values (each map marker has unique Point ID). When you click on a map marker on the map, the drilldown report/chart/map will open in a new browser tab.



*Drilldown Options*

If all the drilldown parameters are multi-value (see Section 3.1.3.2.2.1 - Multi-Value Parameters), you will be allowed to select multiple map markers at once. For example, you can use Examples/QuickStart/QuickStart Maps/MultiValue data source, map *MULTIVALUE* parameter to *STATE* column, and then click *Apply*.



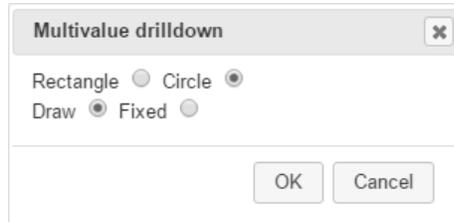
*Multi-value Data Source*

Selecting multiple map markers can be done by clicking the *multi-value Drilldown* button (see the image below; the button is displayed only if the map contains multi-value Drilldown).



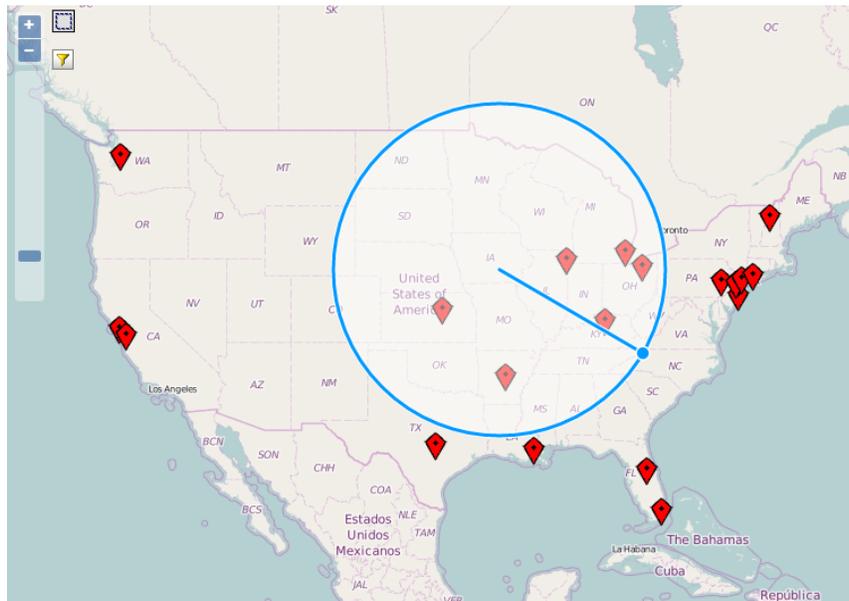
*Multi-value Drilldown Button*

After clicking the *Multi-value drilldown* button, the *Multivalue drilldown* dialog will appear.



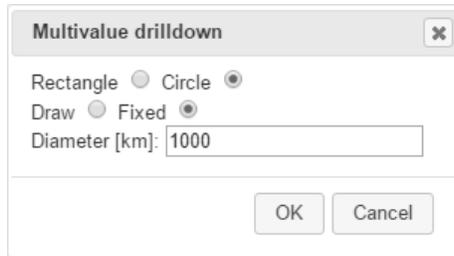
*Multi-value Drilldown Dialog*

You can choose a shape of the bounding box (rectangle or circle). The circle can be fixed or drawn. Once you select a shape, click the *OK* button. Then you can select multiple map markers by dragging using bounding box. Left click on the map (this determines rectangle corner or circle center), hold mouse button down and drag it, and then release the mouse button to determine size of the bounding shape.



*Bounding Box*

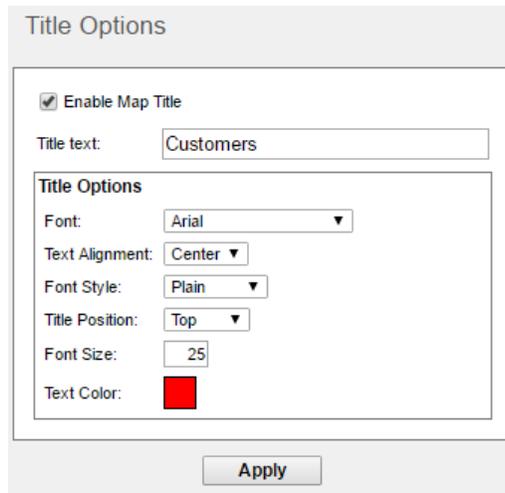
If you select fixed circle as a bounding box, you can set a diameter of the circle. Write a diameter value and click the *OK* button. You will see a circle with the diameter you just set. Then click to the map. Your click is a center of the circle bounding box. A new tab with a drilldown of your selection (markers within the circle) will appear.



*Fixed Circle Bounding Box*

### 5.2.6.10. Map Title

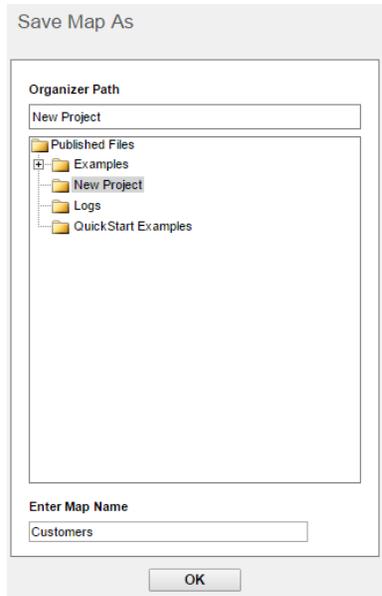
To enable map title, click on the **T** *Map Title Options* icon and check the *Enable Map Title* checkbox. Then you can insert the map title text and select various options - font, text alignment, font style, title position, font size (positive integers only), and text color. The color can either be chosen from swatches or inserted manually by entering red, green and blue components (all the values has to be integers from 0 to 255).



*Map Title Options*

### 5.2.7. Save Map

You can save the Online map by clicking the  *Save* button on the toolbar. This will open a dialog allowing you to specify a name for the map. Enter a name for the map, select a project where you want to save it, and click the *OK* button.

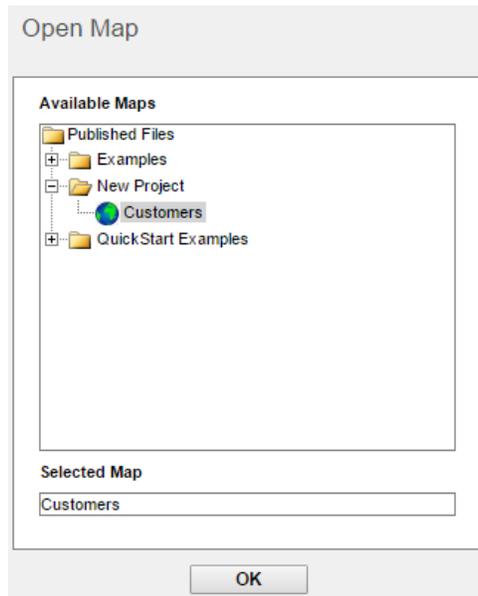


*Save Dialog*

After saving or opening an existing Online map, the  *Save As* button allows you to save the existing or modified map into a different name or location.

## 5.2.8. Open Saved Map

You can open a saved map by clicking the  *Open* icon on the main toolbar. The *Open Map* dialog will appear.



*Open a Map*

All Online maps created in Online Maps designer are visible in Organizer. Select a map and click the *OK* button to open it.

## 5.2.9. Exit

You can exit Online Maps designer either by clicking the  *Home* icon in the upper right corner, clicking the  *Logo* icon in the upper left corner. Before closing, you will be asked if you want to save an unsaved map.

## 5.3. SVG Maps

### 5.3.1. Introduction to SVG Maps

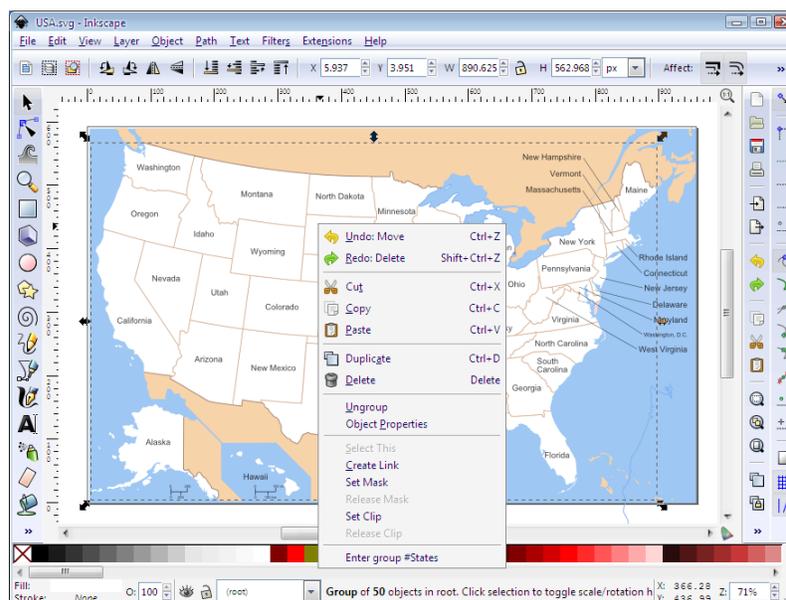
Unlike Online Maps, SVG Maps do not use map points. They use **map areas**. The map areas can be colored based on values from the map data source. Coordinates of the map areas are defined in **SVG map image**. Virtually any SVG image can be used for SVG Maps, but it usually requires some modifications, which can be done in any graphical editor with SVG support. ERES does not contain any integrated graphical editor, so you have to use third party software. The recommended software is **Inkscape**, which can be downloaded at [www.inkscape.org](http://www.inkscape.org) [ <https://inkscape.org/> ]. Inkscape is released under GPL license version 2 [ <http://www.gnu.org/licenses/old-licenses/gpl-2.0.html> ], so it is free for commercial and non-commercial use.

SVG (Scalable Vector Format) images consist of objects. These objects can be used as map area. To distinguish among individual map areas, ERES uses **Area ID**. Each map area must have an Area ID, but unlike Point ID, Area ID does not take up more than one field (it always consists of only one field). The Area ID has to be unique for every area and they are read from the SVG Image object ID attribute. This attribute can be set in most graphical editors. Please see the next chapter for a step-by-step guide for the recommended tool - Inkscape.

ERES is distributed with several SVG images that are already prepared to be used for creating SVG Maps without any modification. These images can be found in `<ERES_Installation_directory>/MapFiles/SVG`. They are also included in Organizer in SVGMaps project. You can find the list of all these maps in the Appendix 5.A - List of SVG Map Images.

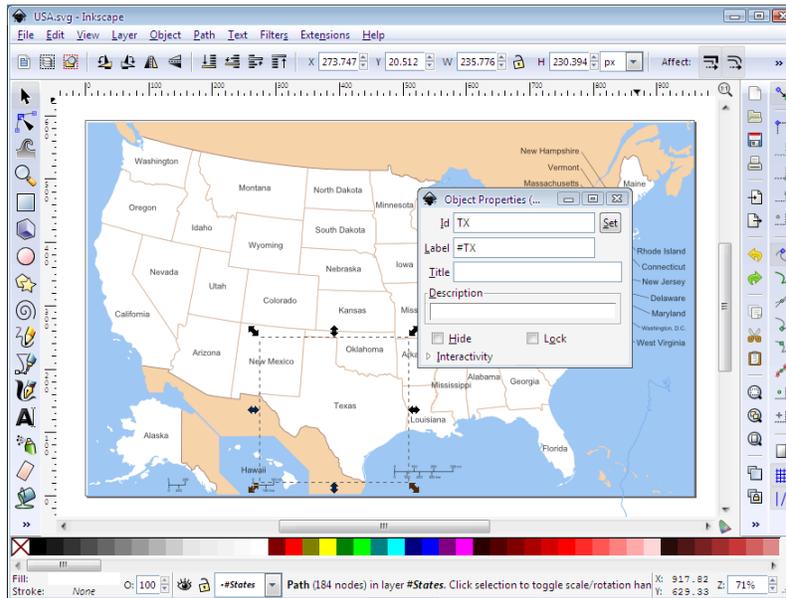
### 5.3.2. Set Area IDs Using Inkscape Editor

Download the Inkscape editor (<http://www.inkscape.org/> [ <https://inkscape.org/> ]) and install it on your computer. Open an SVG image. Complex SVG Images may consist of groups, each consisting of several elements (areas). Right click on the area you want to change the ID of. At the bottom of the pop-up menu, notice an option called *Enter group [name of the group]* (see the screenshot below) and click on it to enter the group.



*Inkscape - Entering Object Group*

Right click on the area you wish to change the ID of again and select the *Object properties* option. Fill in the new Area ID in the ID field and confirm it by pressing the *Set* button. Close the *Object properties* dialog. You can now set Area IDs for other objects in a similar way.



*Inkscape - Editing ID Attribute*

If your map area consists of several unconnected SVG objects (like Alaska and Hawaii for USA), you can group them in one group and assign the Area ID to the whole group. To do that, do not enter the group as described above, but instead go to the Properties dialog for the whole group and assign the ID there. If a group has Area ID assigned and there are objects inside this group that also have Area IDs assigned, there could be conflicts if the both Area IDs (ID of the whole group and ID of an object from this group) are present in the map data source. In this case, the behavior is undefined, so you can get unexpected results. It is OK to have groups with Area IDs that contains objects with Area IDs, unless you use both IDs simultaneously in your data source. It can make sense in some cases (e.g. you can have a world map with continent names as well as country names. You can safely use this map if you use either continent names or country names separately).

To save the changes, open File menu (upper left corner) and select Save. Note that SVG images that you want to use for creating SVG Maps have to be inserted in Organizer.

### 5.3.3. Area ID Mapping

SVG Map needs an SVG image that defines the map areas. All SVG objects in the SVG image can be used as map areas. The only requirement is that they must have an Area ID assigned as described in the previous section. These Area IDs are used for mapping the map data source rows to map areas (SVG objects). Unlike the Point IDs, Area IDs cannot have multiple fields, so the mapping is much simpler. It is enough to select just one data source column, which will be mapped to the Area ID. A map area is associated with the data source row, which has the value of the mapped column equal to the Area ID (comparison is case sensitive).

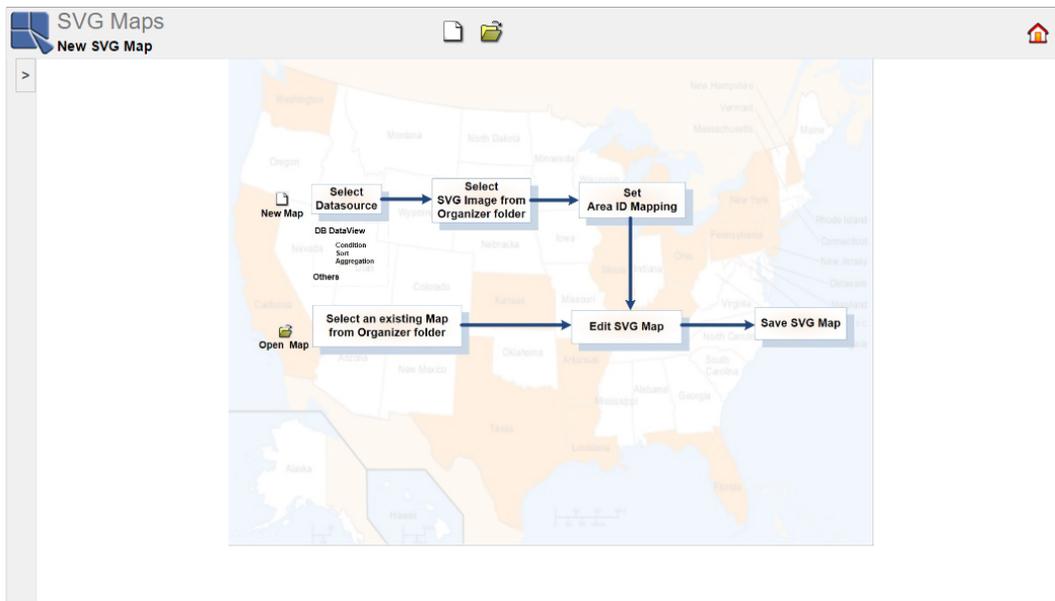
In the SVG Maps, the map data source is used for coloring areas according to numerical value of a data source column. It brings one more limitation for the map data source compared to Online Maps. There has to be at least one matching data source row for one Area IDs. If there is no matching row, the area will have original (unchanged) color. If there is exactly one, the fill color of the matching map area will be changed according to Thresholds (see Section 5.3.4.5 - Thresholds). If there is more than one matching row, the behavior is undefined. There will be no error message, but the data displayed may not make sense.

### 5.3.4. SVG Maps Designing

#### 5.3.4.1. Start

SVG Maps can be started from the ERES Start page. If you have logged in as a user with design privileges, then you can follow SVG Maps link to begin using SVG Maps.

When the SVG Maps designer launches, the first page that appears prompts you to select between opening an existing map, or creating a new one.

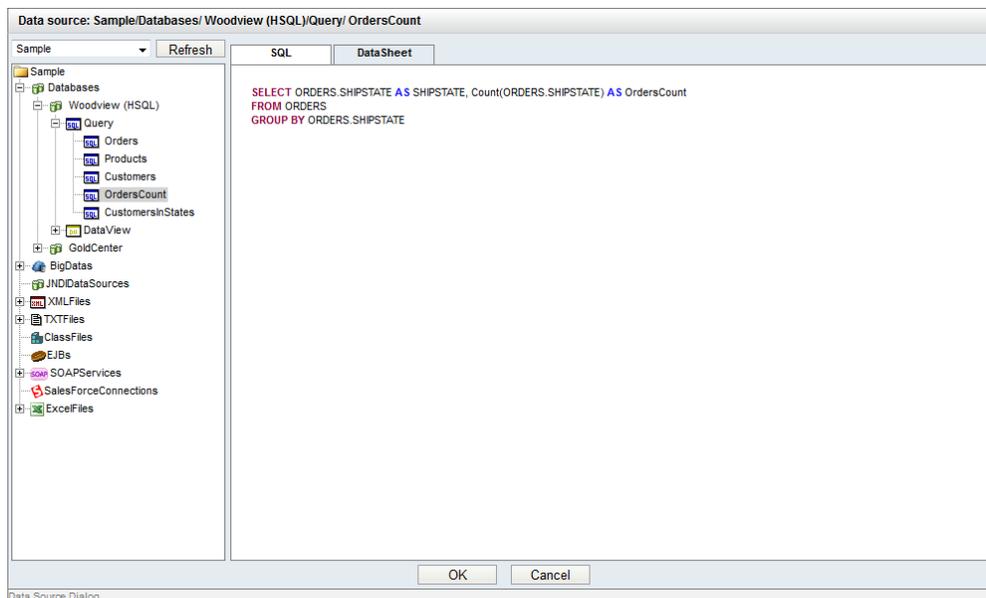


SVG Maps Start Options

### 5.3.4.2. Select Data Source

If you want to create a new SVG map, click on the  *Create New Map* icon on the toolbar. The *Data Source Dialog* appears and prompts you to select data registry and data source you want to use with SVG Maps. In order to use SVG Maps, a user must have read privileges to one of the registries defined in the Organizer. For more about creating and managing data registries, please see Section 3.1.1 - Managing Data Registries.

Select data registry from the drop-down menu in the upper left corner. The tree-list below displays the content of the selected registry. There are all of the data sources that have been defined in the registry that the user has access to. Select a data source you want to use for the SVG map.



Data Source Dialog

Unlike Report Designer or Chart Designer where the user can select to create new data sources or modify existing ones, the only option is to select a data source for the map. The only exception to this is for Data Views or Data

View Queries. If data sources of this type are selected, you will see the *DataView Builder* on the right side of the Data Source Dialog. The *DataView Builder* allows you to build or modify queries against the View. For more information about managing data sources, please see Section 3.2 - Data in QuickDesigners and Maps.

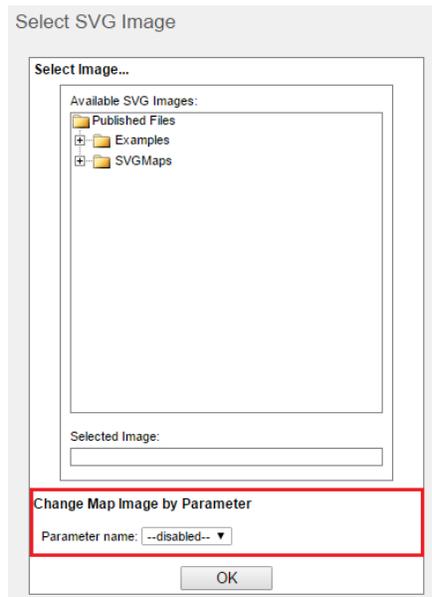
Once you finish selecting a data source, click the *OK* button to close the *Data Source Dialog*.

### 5.3.4.2.1. Change Data Source

To change a data source (for SVG maps that have already been created), click the  *Change Data Source* icon on the toolbar. The *Data Source Dialog* will open and you will see current data source. Select other data source from the tree list and then click the *OK* button to apply it.

### 5.3.4.2.2. Parameterized Data Source

You can use parameterized data source for SVG maps. For more information about parameterized data source in SVG maps, please see Section 5.3.5 - Dynamic SVG Maps.

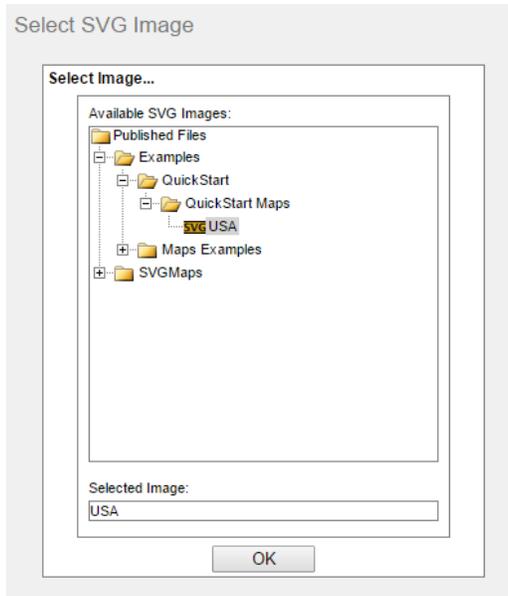


*Selecting Parameterized Data Source*

To change a parameter from the toolbar of the SVG Maps designer, click the  *Refresh/Set Parameters* icon on the toolbar. You will see available parameters. Set the parameter(s) and click *Submit* to apply this setting.

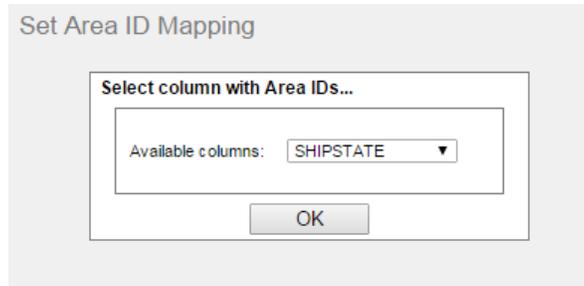
### 5.3.4.3. Select SVG Image

Now you can see the *Select SVG Image* dialog on the left side of the SVG Maps designer. This dialog prompts you to select a Map Image. Map Images are SVG image files with data structures containing geographic data (see Section 5.3.3 - Area ID Mapping for more information about required image format) and must be inserted into the Organizer before they can be used.



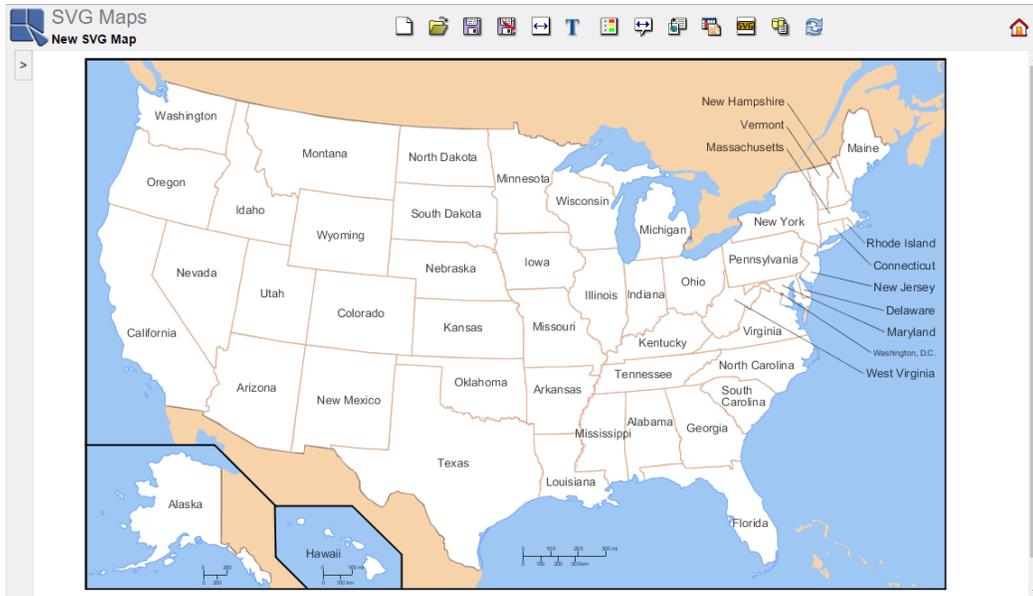
*Select SVG Image Dialog*

The *Select SVG Image* dialog contains a tree-list showing all projects, folders, and SVG images inserted in the Organizer. Select an SVG image you want to use as the map image and click the *OK* button. The *Set Area ID Mapping* dialog appears. Select a data source column that should be mapped to the Area IDs and then click the *OK* button.



*Select Column with Area IDs*

SVG map will open (*Set Area ID Mapping* dialog is collapsed on the screenshot).

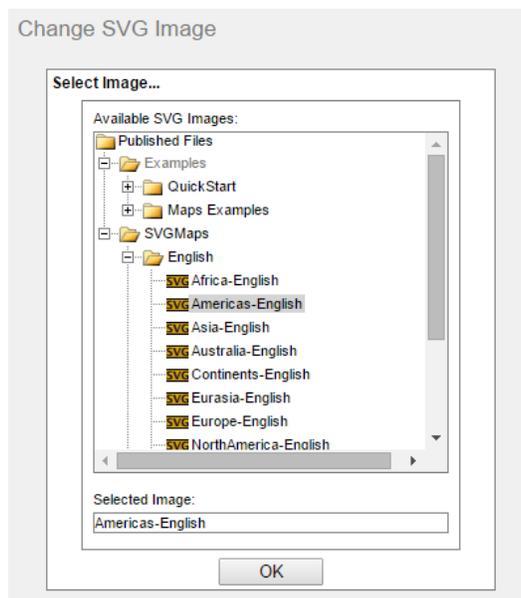


SVG Maps Interface

As you can see, right now we have only the basic map with no data highlighted. There are two ways of adding data to the SVG Map: Thresholds and Drilldowns.

### 5.3.4.3.1. Change SVG Image

To change an SVG image (for SVG maps that have already been created), click the  *Change SVG Image* icon on the toolbar. Select an image from the tree list and then click the *OK* button to apply the image.



Changing SVG Image

### 5.3.4.4. Toolbar

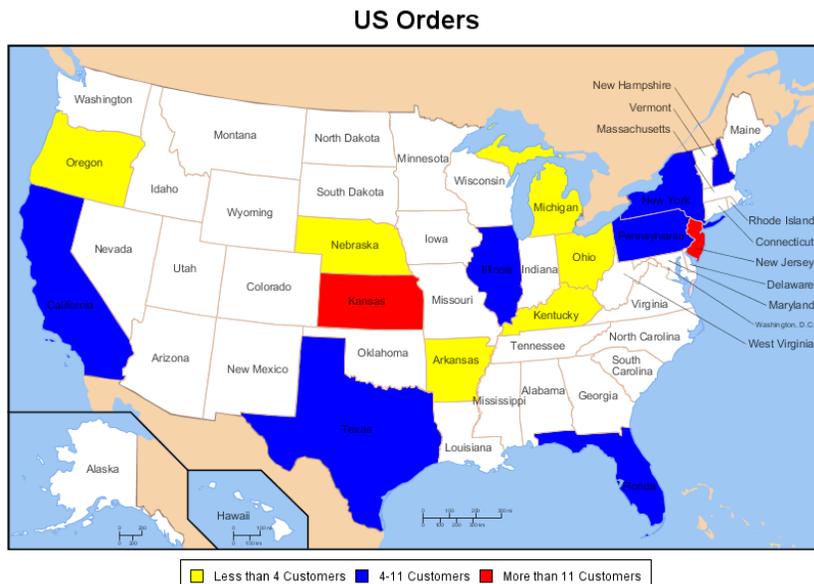
A lot of formatting actions for reports in SVG Maps are accessed through the toolbar. The icons perform the following actions:

-  Start a new SVG map

-  Open an existing SVG map
-  Save the current SVG map
-  Save the SVG map as
-  Set a map size
-  Insert/Edit a map title
-  Set thresholds
-  Insert/Edit a tooltip
-  Insert a drilldown
-  Change area ID mapping
-  Change SVG image
-  Change data source
-  Refresh/Set parameters

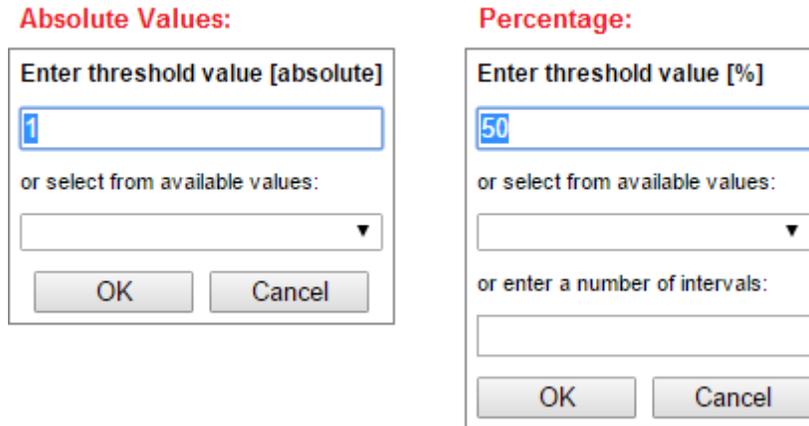
### 5.3.4.5. Thresholds

Thresholds allow you to differentiate areas by color based on value in a particular data source column. If a threshold is set, all the map areas (i.e. objects with a record in the map data source) are colored with color assigned to the appropriate value range.



*Example: SVG Map with Thresholds*

To set the thresholds, click on the  *Set Thresholds* icon on the toolbar and select a threshold column (i.e. column that will be compared to the ranges defined below). Only numerical column can be selected. You can add a threshold value(s) in actual values or percentage unit. To add a threshold value, click the  button next to the empty table field. A dialog box will open prompting you for the new threshold value.

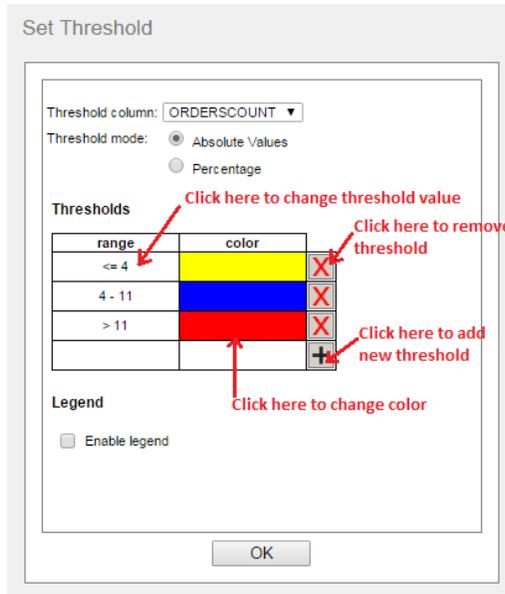


*Entering Threshold Values*

You can either enter it manually or select one of the suggested values from the drop-down list at the bottom of the dialog. This list contains all the values from the selected threshold column rounded to integers (There is a third way to enter threshold values for percentage units. It is an option to enter a number of intervals). After adding the first threshold value, two ranges are added - one for values below and one for values above the given threshold. Any threshold value added after that will create only one new range by splitting one of the old ranges in two. You can edit any of the existing thresholds by clicking on the appropriate range (the first column in the table).

To change the color assigned to a value range, click on the empty table field next to the value range you want to change the color of. You can then select one of the predefined colors or choose any color by entering its red, green and blue component values (each value is an integer within the 0-255 range). To remove a threshold value, click on the red  button next to the value you want to remove. Changes will take effect after you confirm the *Set Threshold* dialog by clicking the *OK* button.

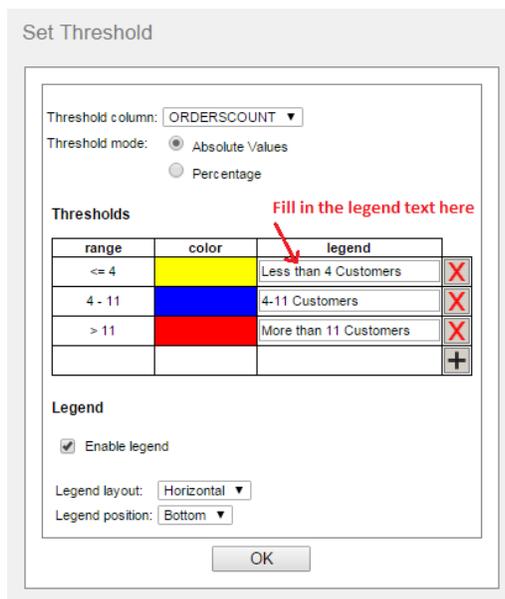
To remove all thresholds and disable coloring of areas, delete all threshold values or select an empty field for the threshold column.



*Set Threshold Dialog*

### 5.3.4.5.1. Legend

The *Set Threshold* dialog also allows you to add a legend to describe the value ranges. To enable it, check the *Enable Legend* checkbox. Then you can select a layout and position of the legend. If the legend is enabled, legend input fields next to the value ranges will appear. Insert a description of the ranges in these fields. If the legend is too large to fit one column (for vertical layout) or row (for horizontal layout), it will be split into several columns/rows. If it still does not fit the map canvas even after splitting, it will not be displayed at all. If this ever happens, you can try to resize the map or change the position or layout of the legend.

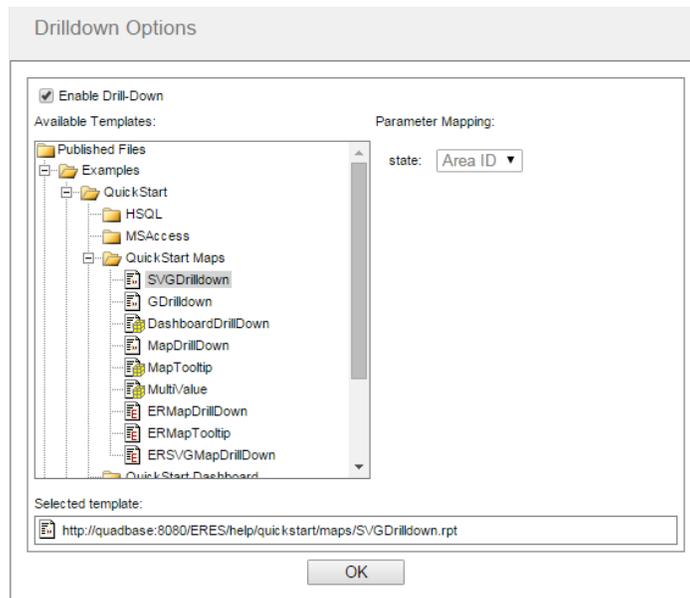


*Setting Legend*

### 5.3.4.6. Drilldowns

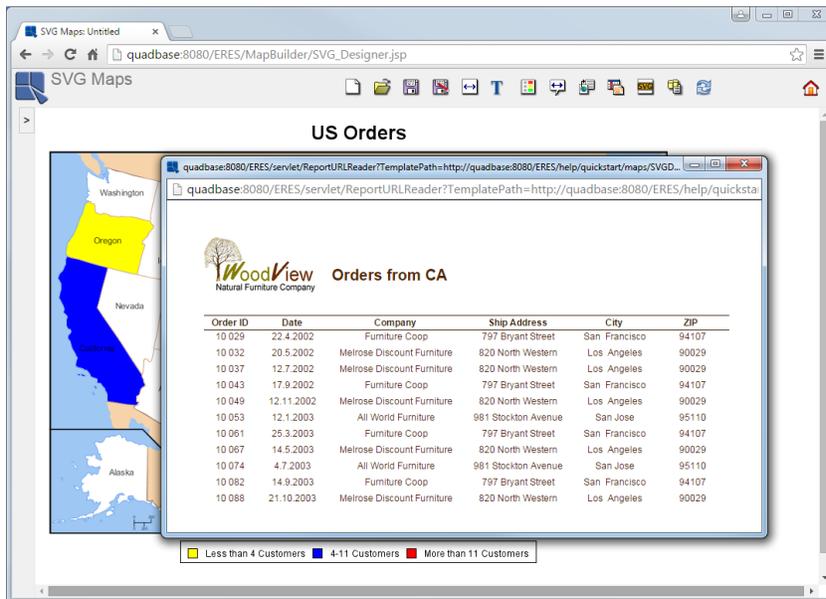
Click on the  *Drilldown Options* icon on the toolbar and select *Enable Drilldown* option. The Organizer structure will show up in the *Available Templates* treeview. Select a parameterized report, chart, or map. All parameters will be automatically mapped to the Area ID. This means that all parameters of the drilldown report, chart or map will be set to the Area ID of the selected area. It is not possible to change the mapping. SVG map also does not

support multi-value drilldowns (multi-value parameters are treated as single-value). Once you select a file, click the *OK* button to open it.



*Drilldown Options*

When you click in an area matching the ID mapping (e.g. a colored state), a drilldown report will open in a new window.



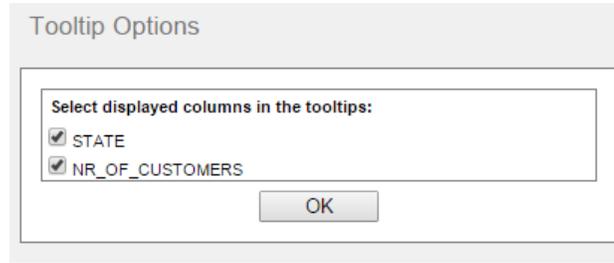
*Map with Drilldown*

### 5.3.4.7. Tooltips

Tooltips in SVG maps are different than tooltips in Online maps. In SVG maps, there are no reports in tooltips, just data from data source fields associated with particular map area.



To set up tooltips in a SVG map, click on the *Tooltip Options* icon on the main toolbar. The following dialog will appear:



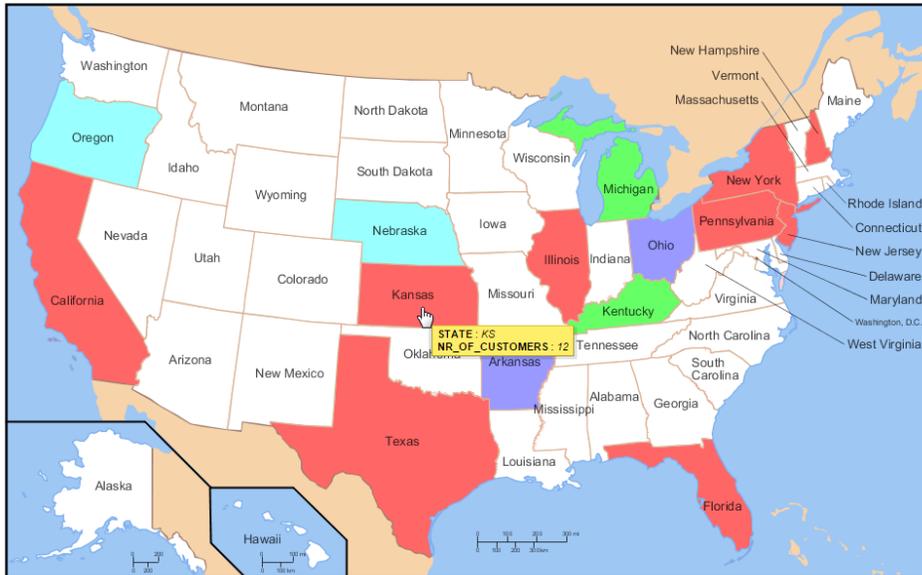
*Tooltip options dialog*

On this dialog, you can choose which data source fields will be displayed in tooltips, which will pop up in yellow background when you move your mouse cursor over a map area.



**Note**

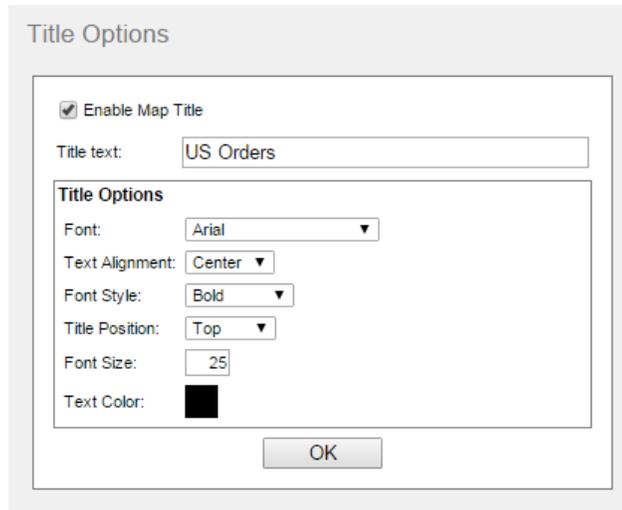
If you deselect all options on the *Tooltip options* dialog, default browser tooltips will be displayed (in grey background), showing SVG area IDs.



*SVG Map with a Tooltip*

**5.3.4.8. Map Title**

To insert a map title, click the **T** *Map Title* icon on the toolbar. Check the *Enable Map Title* checkbox. Then you can insert the map title text and select various options - font, font style, font size (positive integers only), text alignment, title position, and text color. The color can be either choosed from swatches or inserted manually by entering red, green and blue components (all values has to be integers from 0 to 255). To apply your settings, click the *OK* button.



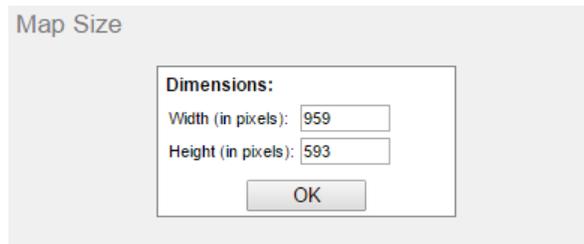
The 'Title Options' dialog box contains the following elements:

- Enable Map Title
- Title text:
- Font:
- Text Alignment:
- Font Style:
- Title Position:
- Font Size:
- Text Color:
- OK button

Map Title

### 5.3.4.9. Map Size

To set a map size, click the  *Set Map Size* icon on the toolbar. Enter dimensions and then click the *OK* button to apply your setting.



The 'Map Size' dialog box contains the following elements:

- Dimensions:
  - Width (in pixels):
  - Height (in pixels):
- OK button

Map Size Setting

### 5.3.4.10. Change Parameter Values

To change parameter values (for maps with parameterized data source) or refresh the map, click the  *Refresh* icon on the toolbar and the Parameter setting dialog will appear. Select a parameter value(s) and then click the *Submit* button to apply parameter values on the map (You can reset your selection by clicking the *Reset* button).

Start Date	<input type="text" value="2001-01-14"/>	<input type="button" value="▶"/>
End Date	<input type="text" value="2003-12-09"/>	<input type="button" value="▶"/>

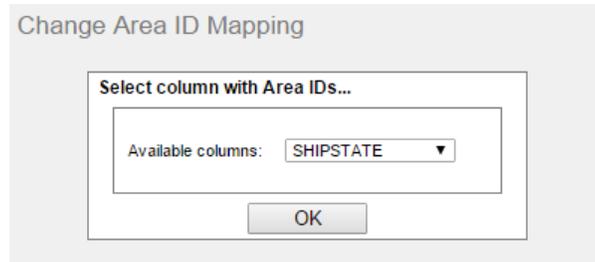
Parametr Setting Dialog for Single-value Parameters

States	<input type="text" value="AR"/> <input type="text" value="CA"/> <input type="text" value="FL"/> <input type="text" value="IL"/>	<input type="button" value="▶"/>
--------	--	----------------------------------

Parameter Setting Dialog for Multi-value Parameters

### 5.3.4.11. Change Area ID Mapping

To change area ID mapping, click the  *Change Area ID Mapping* icon on the toolbar. Select a column from the drop down menu and then click the *OK* button to apply your setting (For more information about area ID mapping, please see Section 5.3.3 - Area ID Mapping).



*Changing Area ID Mapping*

### 5.3.5. Dynamic SVG Maps

ERES Dynamic SVG Maps allow changing SVG image according to a parameter, i.e. you can use several SVG images in a single SVG map. Dynamic SVG map can be used also as a drilldown, so you can create useful SVG map with drilldown - for example continent to country, country to state, state to county, etc.

There are some rules and recommendations for Dynamic SVG Maps:

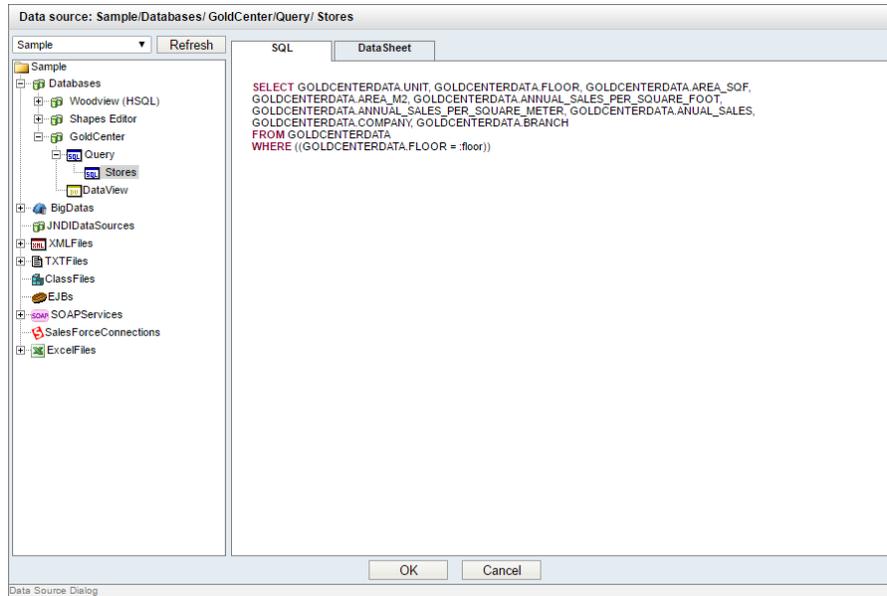
- SVG images have to be in format: `imageName_PARAMETER.svg`, where `imageName` is the same for all SVG images and `PARAMETER` is used as the parameter.(e.g. `worldmap_political`, `worldmap_physical`, `worldmap_topographic`,...)
- Data source has to be parameterized where one of parameters takes the same values at the end of the SVG images file names (after the underscore character).
- SVG images have to be saved inside a single folder.
- One of the SVG images has to be in Organizer. This image is used as default for creating a map and it is also used in case the SVG image is not available due to a wrong parameter, missing file, etc.
- It is recommended to have all SVG images in the same size.

It is possible to change SVG Images by clicking the  *Refresh/Set Parameters* icon on the toolbar. You will see a drop down menu for selecting a parameter. Select a parameter and click the *Submit* button. The corresponding image will be used for the map. All settings of the map (thresholds, legend, title, tooltips, drilldowns) will be applied to each SVG image.



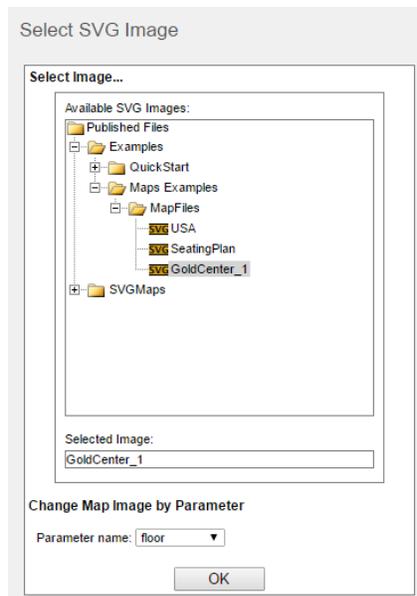
*Parameter Selection*

Let's take a look at the following example of a shopping center with three floors. We want to display the status of annual sales per square foot of stores. Open SVG Maps designer (from the ERES main page) and click the  *Create New Map* icon on the toolbar. *Data Source Dialog* will appear. Select *Sample* data registry and then select *Databases/GoldCenter/Query/Stores* query as a data source. Click the *OK* button to close the *Data Source Dialog*.



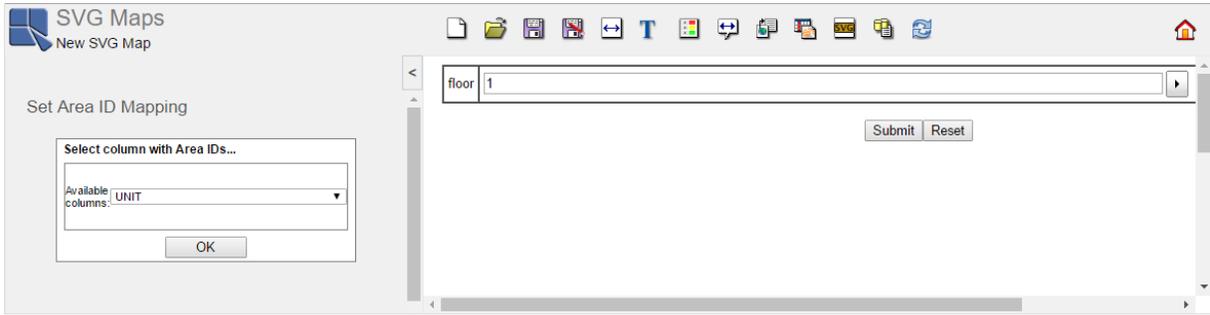
Select Data Source

Now you are prompted to select an SVG image. Select *Examples/Maps Examples/MapFiles/GoldCenter\_1* SVG image. We have three SVG images for this example (*GoldCenter\_1.svg*, *GoldCenter\_2.svg*, and *GoldCenter\_3.svg* in the *ERES/help/examples/Maps/MapExample4* folder), but you can see only one of them in the *Select SVG Image* dialog because only one was inserted into Organizer (it is not necessary to insert all SVG images to Organizer). Next select a parameter for changing SVG images in the *Change Map Image by Parameter* dropdown menu. Select the parameter *floor* and click the *OK* button.



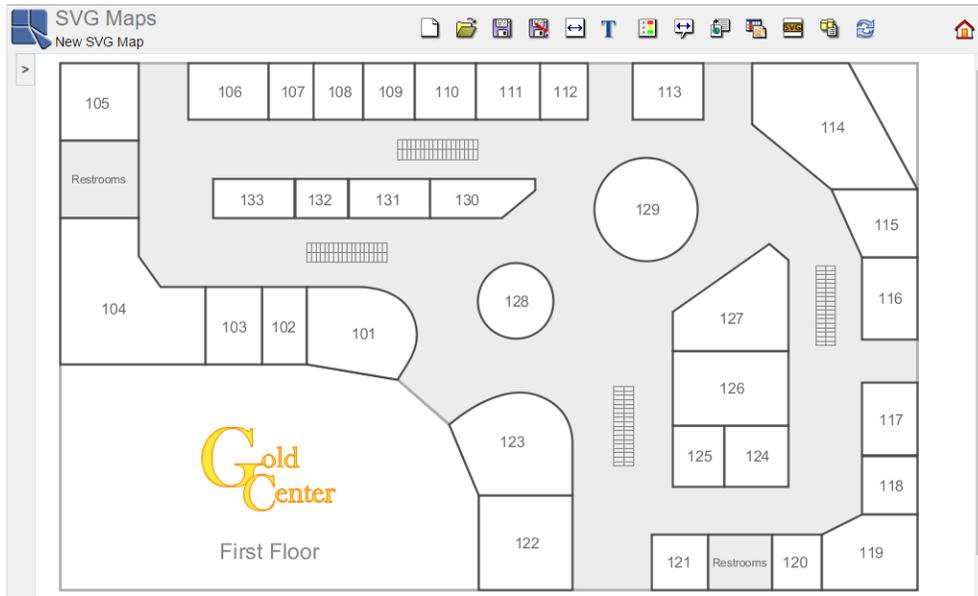
Select SVG Image

Next step is to map IDs to the data source column. Select *UNIT* and click the *OK* button. You can also select a parameter, a number of floor in our case, in this step. Leave the first floor selected and click the *Submit* button to open a map.



*Set Mapping and Parameter*

You can now see a map of the first floor (Left pane is collapsed in the next image).



*Set Mapping and Parameter*

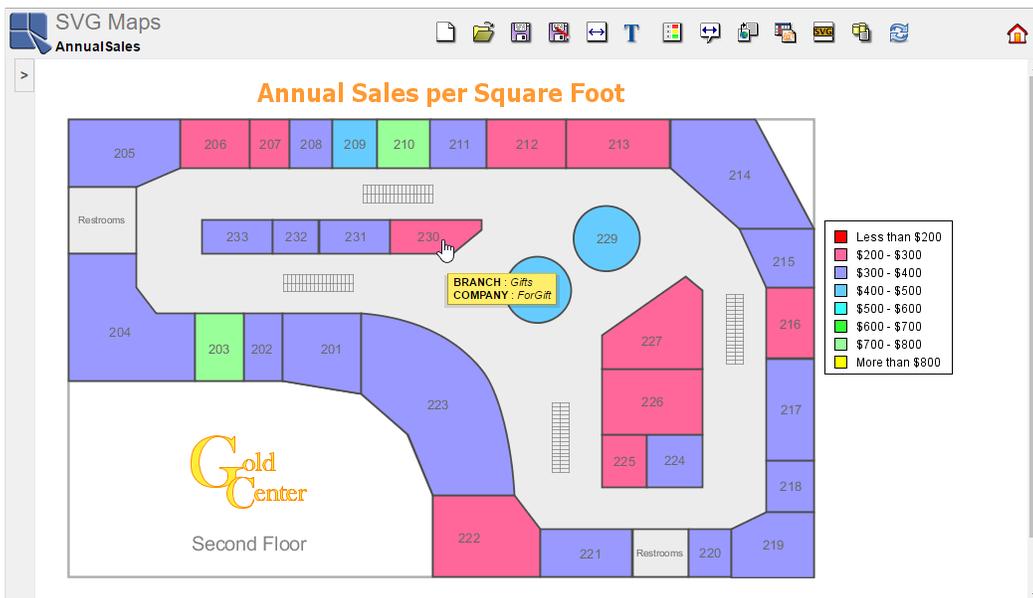
To change the SVG image (the other floor of the shopping centre in our case), you only need to change the parameter value by clicking the  *Refresh/Set Parameters* icon and selecting parameter value from the dropdown menu.

The next image shows the finished SVG map (you can find this SVG map in `Examples/Maps Examples/MapFiles/AnnualSales`). There are set thresholds (for more information, see Section 5.3.4.5 - Thresholds), a legend (for more information, see Section 5.3.4.5.1 - Legend), a title (for more information, see Section 5.3.4.8 - Map Title), and a tooltip (for more information, see Section 5.3.4.7 - Tooltips).



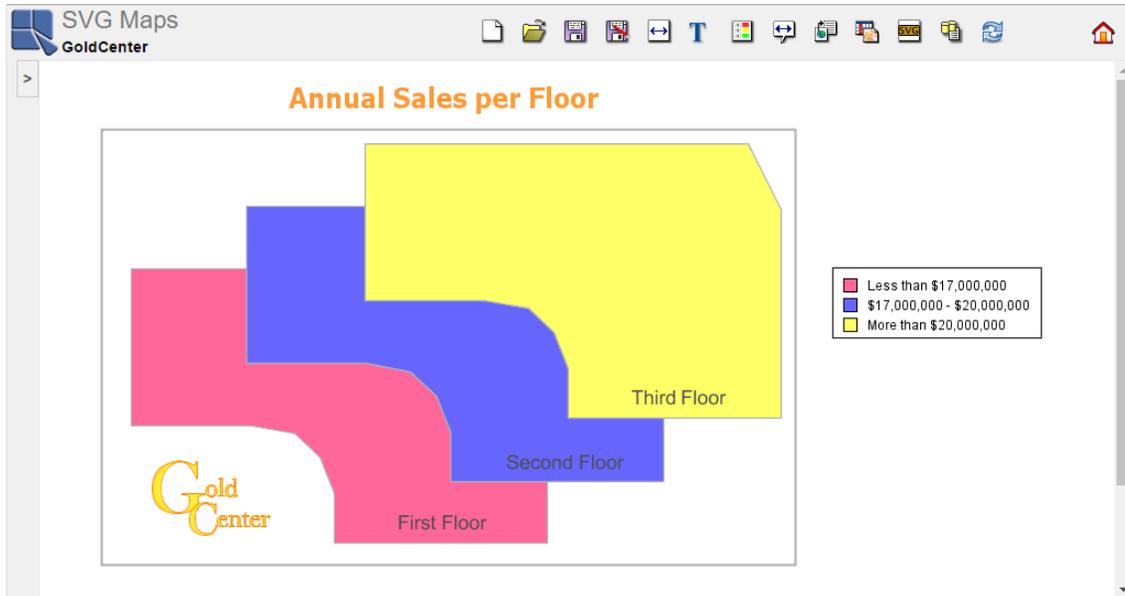
*Finished Map*

The next image shows the SVG map for the parameter value 2, which is the second floor of the shopping center. All adjustments are also valid for the SVG image of the second floor.



*Changed SVG Image*

Dynamic SVG map can also be used as a drilldown. For example, open GoldCenter SVG map from Examples/Maps Examples/MapFiles directory. You should see this map:



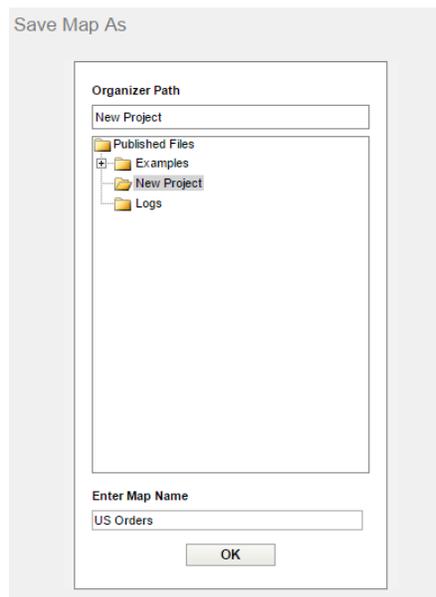
*Basal SVG Map*

Add a drilldown by clicking the  *Drilldown Options* button, select `Examples/Maps Examples/MapFiles/AnnualSales` file in the *Drilldown Options* dialog and click the *OK* button. Then you can click on the area of some floor to open the drilldown.

You can also open the finished SVG map with drilldown `GoldCenterWithDrilldown` under `Examples/Maps Examples/MapFiles` node.

### 5.3.6. Save Map

You can save the SVG map by clicking the  *Save* button on the toolbar. This will open a dialog allowing you to specify a name for the map. Enter a name for the map, select the project where you want to save it and click the *OK* button.

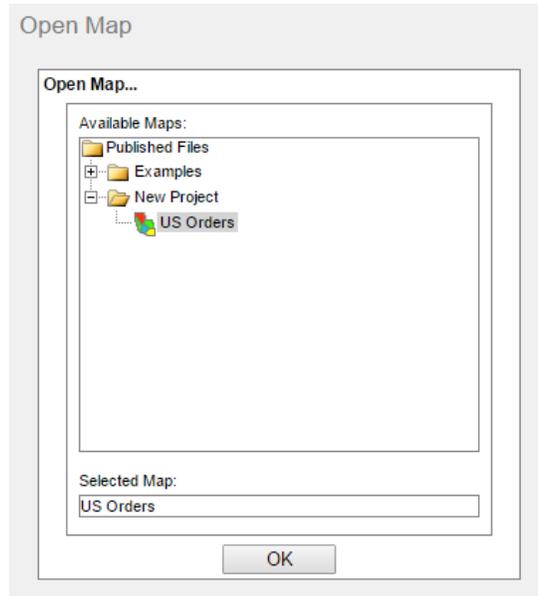


*Save Dialog*

After saving or opening an existing SVG map, the  *Save As* button allows you to save the existing or modified map into a different name or location.

### 5.3.7. Open Saved Map

You can open a saved map by clicking the  *Open* icon on the main toolbar. The *Open Map* dialog will appear.



*Open the SVG Map*

All SVG maps created in SVG Maps designer are visible in Organizer. Select a map and click the *OK* button to open it.

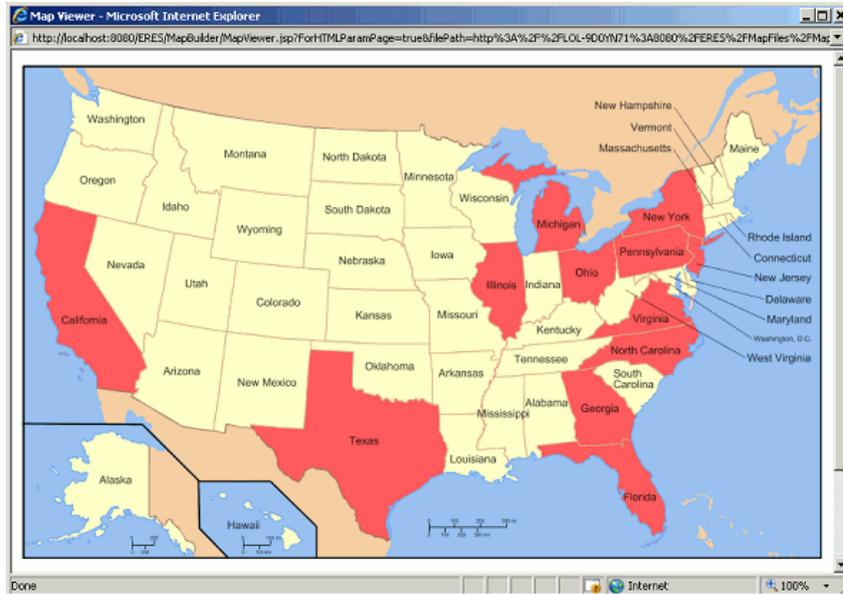
### 5.3.8. Exit

You can exit SVG Maps designer either by clicking the  *Home* icon in the upper right corner, clicking the *SVG*

*Maps* title, or clicking the  *Logo* icon in the upper left corner. Before closing, you will be asked if you want to save an unsaved map.

## 5.4. Map Viewer

Map Viewer is a JSP page used for viewing maps. It is located in `http://machine:port/<ERES_CONTEXT>/MapBuilder/MapViewer.jsp` and it can be used for both Online Maps and SVG Maps. If you open a map from Menu Page, it is automatically opened in the Map Viewer (If you open a map from Organizer, it is opened in corresponding map designer). Another way to use the Map Viewer is to write your own map URL. This is described in the next chapter.



*ERES Maps Viewer (showing SVG Maps with threshold)*

### 5.4.1. Writing Map Viewer URL

You can write your own Map Viewer URL. The URL always begins with the location of the Map Viewer JSP: `http://<machine>:<port>/<ERES_CONTEXT>/MapBuilder/MapViewer.jsp?`. Following the question mark you need to specify Map Viewer parameters. Parameters are separated by the “&” character.

The most important Map Viewer parameter is **filePath**. It specifies the path to the map you want to display. The value can be either URL or file path on the server. The file path can be absolute or relative to the ERES installation directory. This parameter is mandatory and must always be specified.

The remaining Map Viewer parameters are used for setting map parameters. These parameters have no function in non-parameterized maps, so they are ignored if the map is not parameterized. The parameters are:

**ForHTMLParamPage:** This is a Boolean flag that indicates whether or not to display parameter page before displaying the map. If this argument is set to true then the URL will return the parameter page and ignore any parameter values passed into the URL. This argument is set to false by default.

**QueryParamName:** This parameter is used to specify parameter for parameterized map. QueryParamName specifies the name of the parameter you will be supplying a value for (Parameter names are specified when you create the query. For more information about this, see Section 3.1.3.2.2 - Parameterized Queries). It is always followed by QueryParamSize and QueryParamValue parameters.

**QueryParamSize:** This parameter is used to specify the number of values that will be passed into a particular map parameter. As described in Section 3.1.3.2.2.1 - Multi-Value Parameters, some queries can have multi-value parameters. This URL parameter allows you to specify if the parameter (indicated by the previous QueryParamName parameter) takes multiple values. This parameter is always followed by one or more QueryParamValue arguments for each of the parameter values for a particular parameter. If you do not specify this parameter, it will assume that the number of the values to be passed in is one.

**QueryParamValue:** This parameter is used to specify a parameter value you want to pass to the query parameter specified by the QueryParamName argument. For multi-value parameters, a separate QueryParamValue argument must be supplied for each distinct parameter value that you are passing in. For more information about creating parameterized queries, please see Section 3.1.3.2.2 - Parameterized Queries.

Example:

To open GoogleMapParam (one of the map examples - parameterized Online Map) with parameter State set to CA, use this URL:

```
http://machine:port/<ERES_CONTEXT>/MapBuilder/MapViewe.r.jsp?filePath=help/
examples/Maps/MapExam-
ple1/GoogleMapParam.gxml&QueryParamName=State&QueryParamSize=1&QueryParamValue=CA
```

Please replace machine with your machine name or IP, port with the port Tomcat is running on (8080 by default) and <ERES\_CONTEXT> with your ERES context (ERES by default).

## 5.5. Migrating Maps

Often you may need to move maps from one location to another. For example, maps have to be moved between the server and the develop machine. On the develop machine the tooltips/drill-downs/svg images locations can change and may not be located from the data stored in the map .gxml or .sxml file. Therefore, ERES uses SPAK (for SVG Maps) and GPAK (for Online Maps) files to move or archive maps along with its components. To learn how to create a SPAK or a GPAK file, see Section 2.1.4.5 - Dashboard and Map Packages.



### Note

The SPAK and GPAK files can be viewed in the MenuPage, but they cannot be opened in the SVG or Online map designers. To edit a map from a SPAK or a GPAK file, you have to unpack it first (to learn how to unpack DPAK or SPAK files, see Section 2.1.4.5 - Dashboard and Map Packages).

## 5.A. List of SVG Map Images

The following SVG map images are distributed with ERES. They are prepared to be used for creating SVG Maps. They all have uniform look - the land is gray and borders and oceans are white. You can change the colors in any SVG editor (e.g. Inkscape) or you can edit them directly in the SVG files (this requires at least basic knowledge of the SVG format).

The maps are split in two directories according to the Area IDs naming convention. The maps in the <ERES\_Installation\_directory>/MapFiles/SVG/English directory use English country (continent) names, while the maps in the <ERES\_Installation\_directory>/MapFiles/SVG/ISO directory use ISO 3166-1-alpha-2 [ <https://www.iso.org/iso-3166-country-codes.html> ] code. All Area IDs in these maps are in UPPER CASE. The English names are exactly the names from this table [ <https://www.iso.org/obp/ui/> ], only the spaces are replaced with underscores (\_), because spaces are not supported in the SVG objects IDs. However, underscores are automatically converted to spaces before comparing them to the data source columns, so you should still use spaces in your data source.

These SVG map images are already inserted in the Organizer in the SVGMap project. Please note that this project is hidden in the Menu Page, but it can still be displayed and modified in Organizer and the files from this project can be used for creating maps.

This is the list of all SVG map images distributed with ERES with short description. The <NAMING> placeholder can be one of “English” or “ISO”, depending on the naming convention used.

<b>Africa-&lt;NAMING&gt;.svg</b>	Map of all countries in Africa
<b>Americas-&lt;NAMING&gt;.svg</b>	Map of all countries in North and South America
<b>Asia-&lt;NAMING&gt;.svg</b>	Map of all countries in Asia
<b>Australia-&lt;NAMING&gt;.svg</b>	Map of all countries in Australia and Oceania
<b>Continents-English.svg</b>	Map of the whole world divided by continents. The Area IDs are English names of the continents. This map is not available with ISO naming convention because there are no codes for the continents.

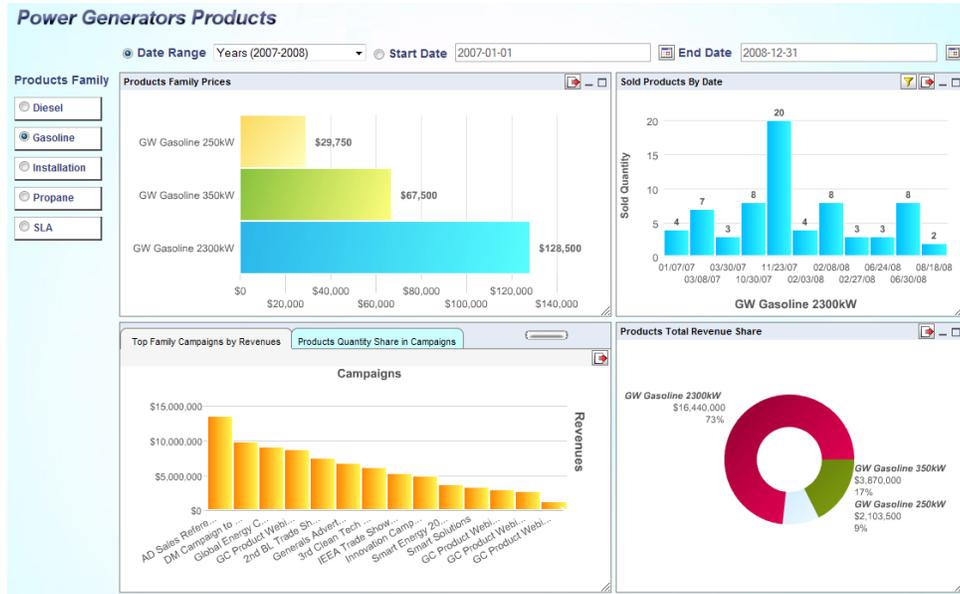
<b>Eurasia-&lt;NAMING&gt;.svg</b>	Map of all countries in Europe and Asia
<b>Europe-&lt;NAMING&gt;.svg</b>	Map of all countries in Europe
<b>NorthAmerica-&lt;NAMING&gt;.svg</b>	Map of all countries in North America
<b>SouthAmerica-&lt;NAMING&gt;.svg</b>	Map of all countries in South America
<b>WorldCompact-&lt;NAMING&gt;.svg</b>	Map of all countries in the world except Antarctica.
<b>WorldCompact-Small-Countries-&lt;NAMING&gt;.svg</b>	Same as WorldCompact-<NAMING>.svg, but very small countries are represented by circles, so they are more visible.
<b>World-&lt;NAMING&gt;.svg</b>	Map of all countries in the world including Antarctica.
<b>World-SmallCountries-&lt;NAMING&gt;.svg</b>	Same as World-<NAMING>.svg, but very small countries are represented by circles, so they are more visible.

# Chapter 6. Designing Dashboards

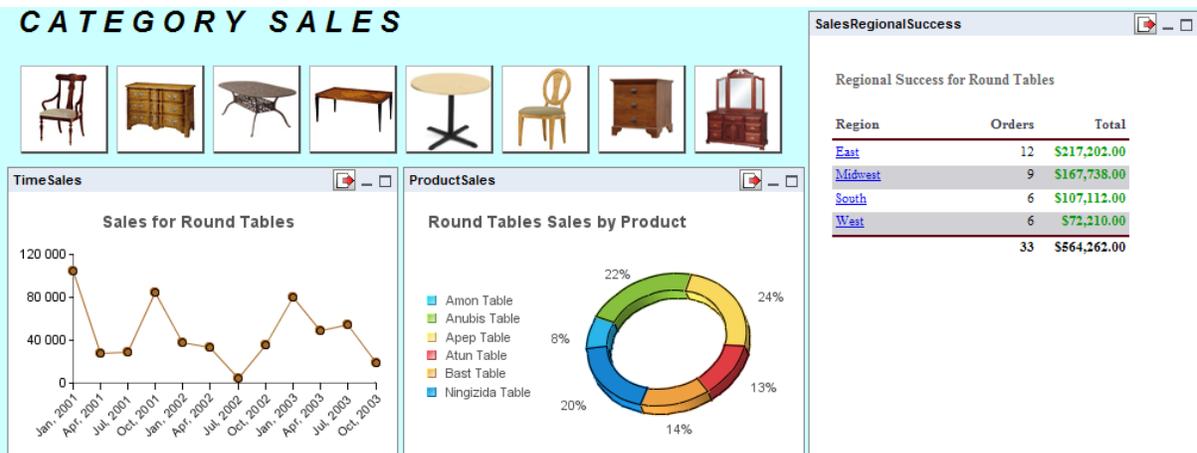
## 6.1. Introduction to Dashboards

Another way the ERES allows users to publish reports, charts and maps is through dashboards. Dashboards allow users to easily consolidate summarized information in a single page or briefing. Dashboards in ERES require no special skill or administrative support. Simple thin-client interfaces allow all users to easily create customized information presentations.

Here are some sample dashboard presentations.

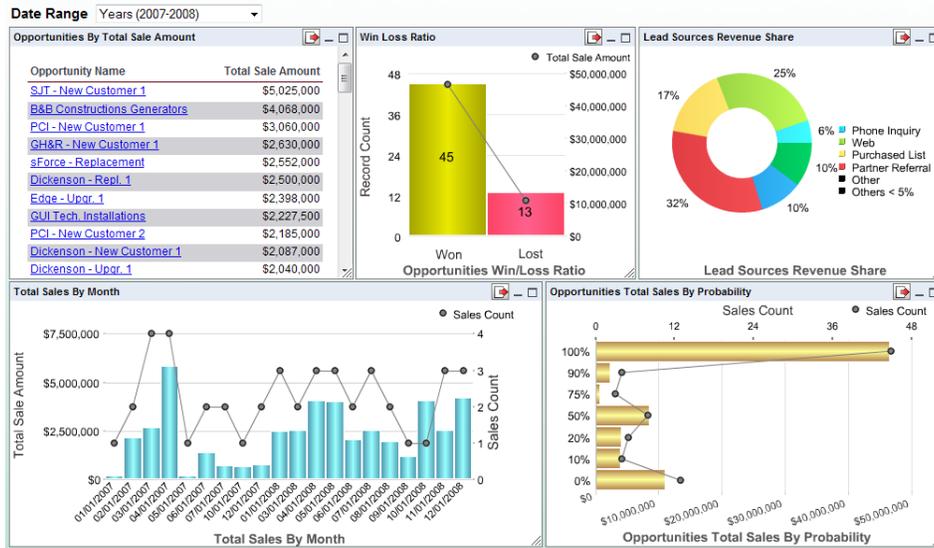


Sales Dashboard



Summary Dashboard

## SALES DASHBOARD



Products Dashboard

The top-level or main presentation consists of a set of charts, reports or maps arranged on the page. You can add shared parameters in a panel, or list the values of parameters so that the dashboard users can apply filters to refresh the applicable charts/reports/maps.

Individual options for each chart or report allow users to apply additional filters as well as export the charts, reports and maps to various formats.

Users can also add layers of drilldown to each chart/report/map in the main presentation. The drilldown can point to a report, chart or map. This allows each piece of summarized information in the presentation to be immediately actionable.

## 6.2. Create Dashboard

Dashboards in ERES can be created using the Dashboard Builder interface. This easy-to-use thin-client interface allows you to easily create dashboard presentations using charts, reports and maps. You can simply insert dashboard components (charts, reports and maps) that are already deployed in the Organizer to the dashboard. Alternatively, you can create them from scratch directly in the Dashboard Builder. The newly created item will be automatically inserted in the dashboard. One convenient feature is that you can edit a chart, report or map in the Dashboard Builder interface without leaving it.

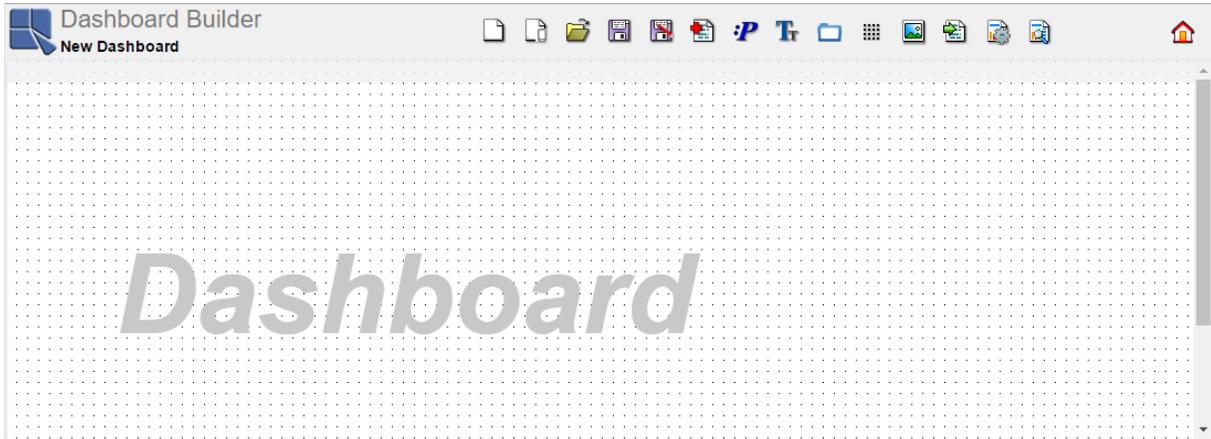
The dashboard layout can be either static or responsive. As you may know, responsive layout is great for displaying content on small devices such as mobile phones. In a static dashboard, the width is dependent on the size and layout of its components. If the dashboard is wider than the screen, all the components stay in place and a horizontal scrollbar appears. Unlike the static dashboard, the responsive layout, using Responsive Dashboard feature, optimizes utilization of the display without exceeding the display width. Therefore the layout is automatically rearranged. The width of the responsive dashboard can also be set manually by dragging a limiter, a vertical line on the right side of the Dashboard Builder working space (in case you want the dashboard to be narrower than the screen).



### Note

It is not possible to display static dashboards as responsive and it is not possible to change them to responsive and vice versa.

The Dashboard Builder can be launched from the ERES Start page. If you've logged in as a user with design privileges, you can follow the Dashboard Builder link to open the interface.



*Dashboard Builder Interface*

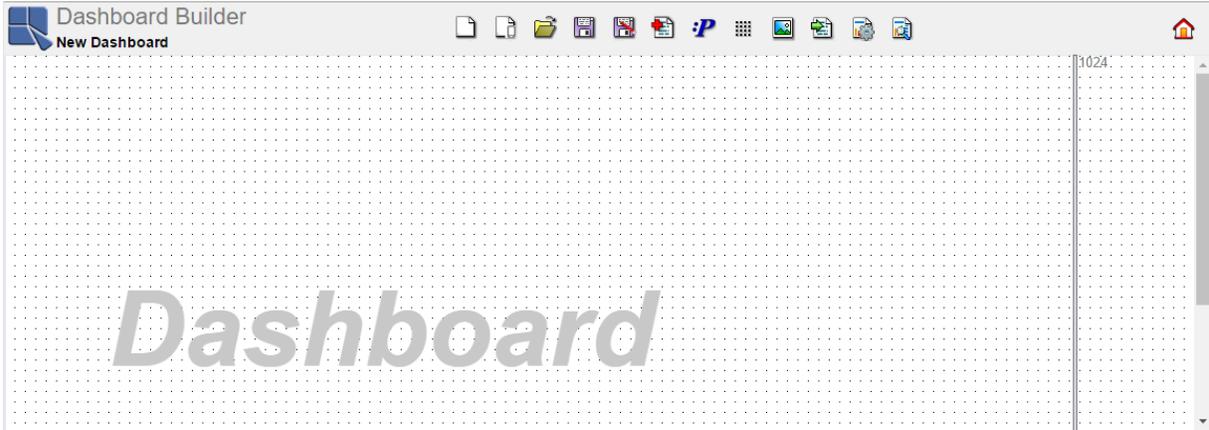
### 6.2.1. Toolbar

The top part of the Dashboard Builder interface contains a small toolbar that allows you to initiate the following actions:

-  Start a new static dashboard
-  Start a new responsive dashboard
-  Open an existing dashboard
-  Save the current dashboard
-  Add report/chart/map to the current dashboard
-  Add share parameter(s)
-  Insert a label into the current dashboard
-  Insert Folder
-  Hide/Show Grid
-  Add dashboard background
-  Migration
-  Dashboard additional options
-  Preview the current dashboard

## 6.2.2. Responsive Dashboard

To create a new responsive dashboard, click the  *New Responsive Dashboard* icon on the toolbar.



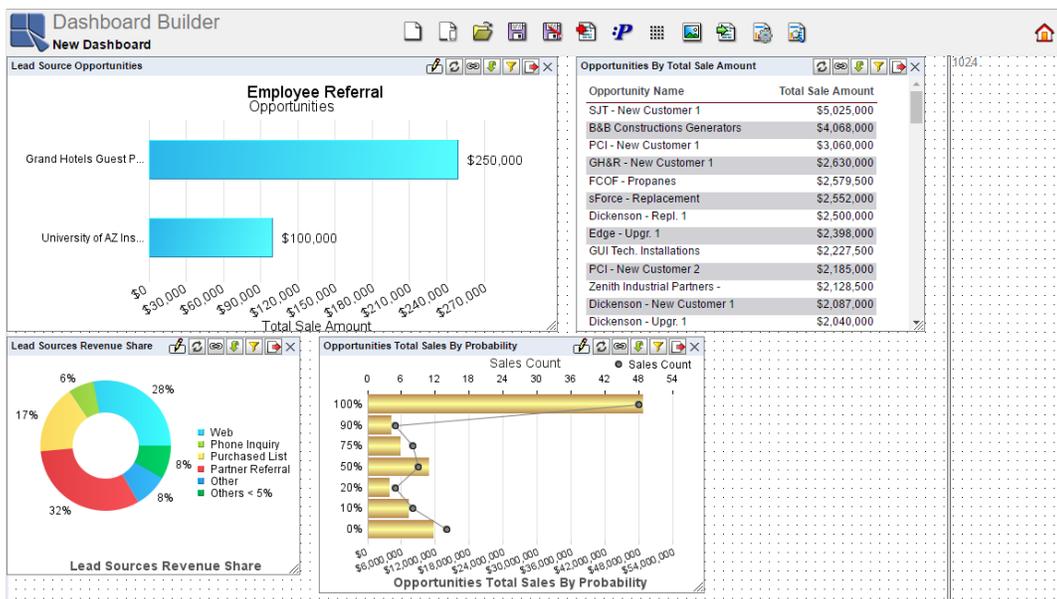
*New Responsive Dashboard*

You can click and drag the limiter to set a dashboard width.



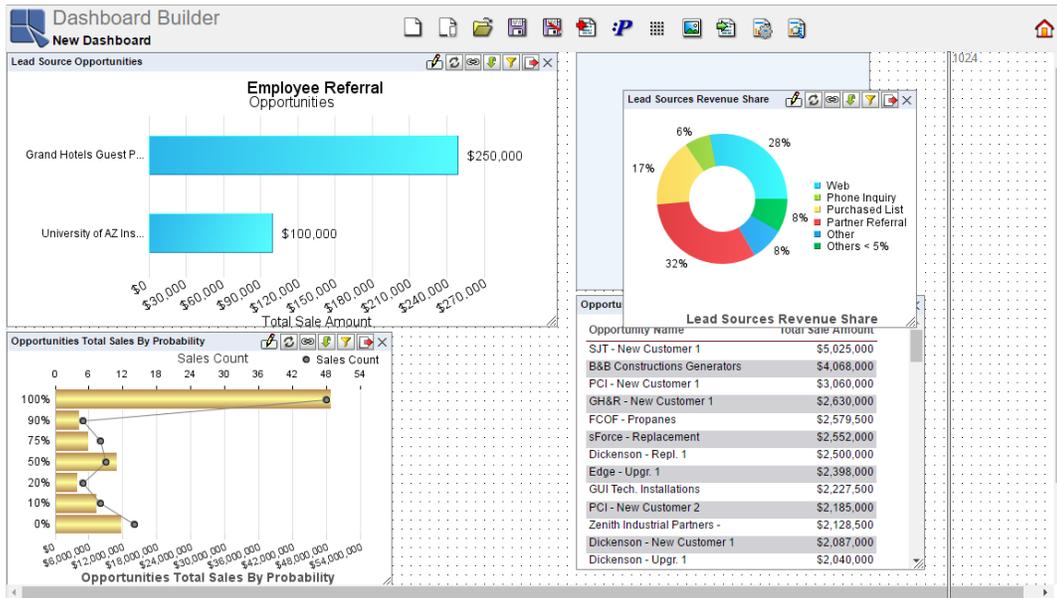
*Dashboard Limiter Setting*

Now you can add reports/charts/maps and resize them. The limiter (or the screen width) keeps the width of the dashboard. The layout of reports/charts/maps is arranged automatically.



*Dashboard With Limiter*

To change the layout manually, click on the header of the report/chart/map, hold the mouse button and move it. If there is enough space for the component, a blue rectangle will appear. Release the mouse button. The moved component is placed instead of the blue rectangle and the other components are arranged as best as possible.



Dashboard Layout Changing

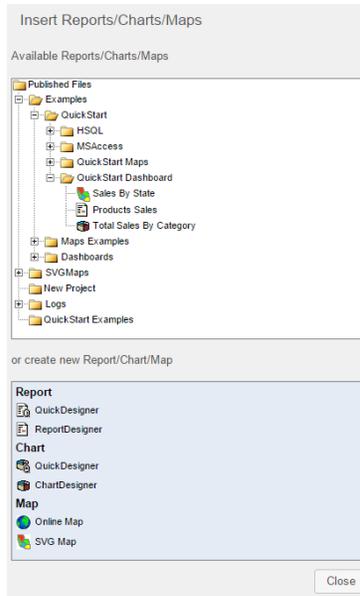


**Note**

It is not possible to insert labels and folders (see Section 6.2.9 - Folders) into responsive dashboards. Also, you cannot show component header bar (see Section 6.2.7 - Additional Options). Hence dynamic charts are not supported in preview and published dashboards.

### 6.2.3. Add Charts, Reports and Maps

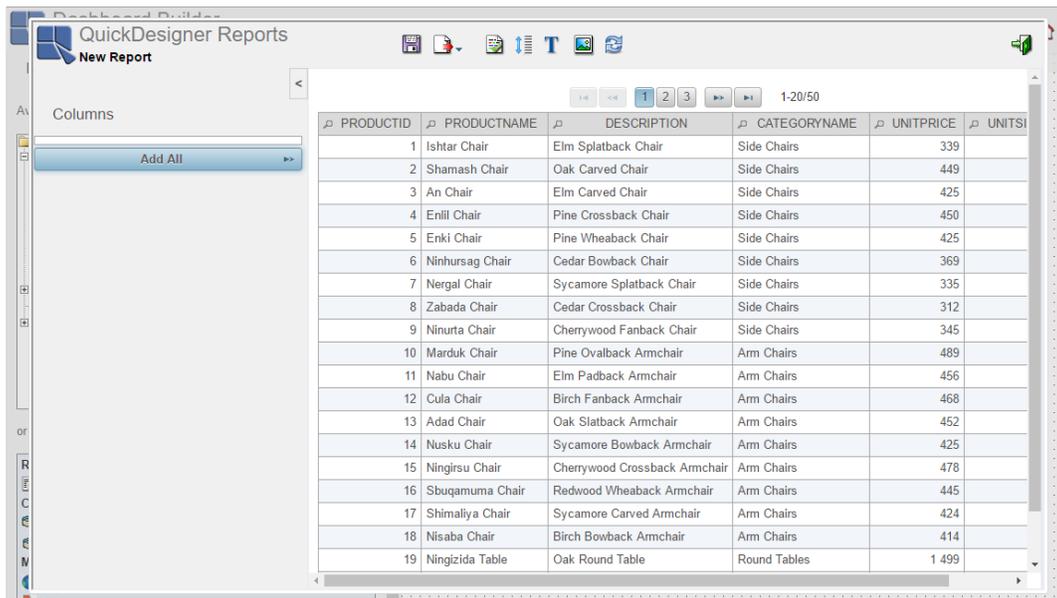
To add a chart, report or map to the dashboard, click the  *Add Report/Chart/Map* button on the toolbar. After you do so, the *Insert Reports/Charts/Maps* dialog will appear in the left pane. The dialog contains a tree that mirrors the folder structure in the Organizer. All charts, reports and maps to which you have access to are listed here. In order to insert a chart, report or map into the dashboard, simply click on the chart or report entry in the tree. The chart, report or map will then be added to the dashboard.



*Insert Report/Chart/Map Dialog*

You can also create a new report/chart/map directly from the Dashboard Builder. To do this, click the relevant link in the lower pane of the *Insert Reports/Charts/Maps* dialog (titled *create new Report/Chart/Map*) and you will be taken to the corresponding designer. Create a new report/chart/map in the designer and save it. Then close

the designer or click the exit icon  to return to the Dashboard Builder. Your new component is added to the dashboard. For more information about working in designers, please see Section 4.1 - Report Designer, Section 4.2 - Chart Designer, Section 4.3 - QuickDesigner Reports, Section 4.4 - QuickDesigner Charts, Section 5.2 - Online Maps, Section 5.3 - SVG Maps.



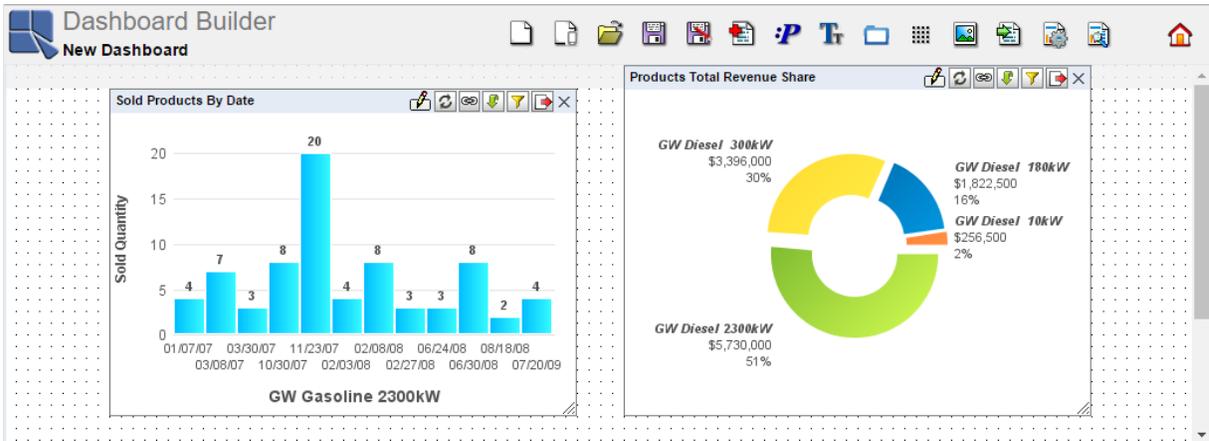
*QuickDesigner Reports Opened from Dashboard Builder*

### 6.2.3.1. Move and Resize Charts, Reports and Maps

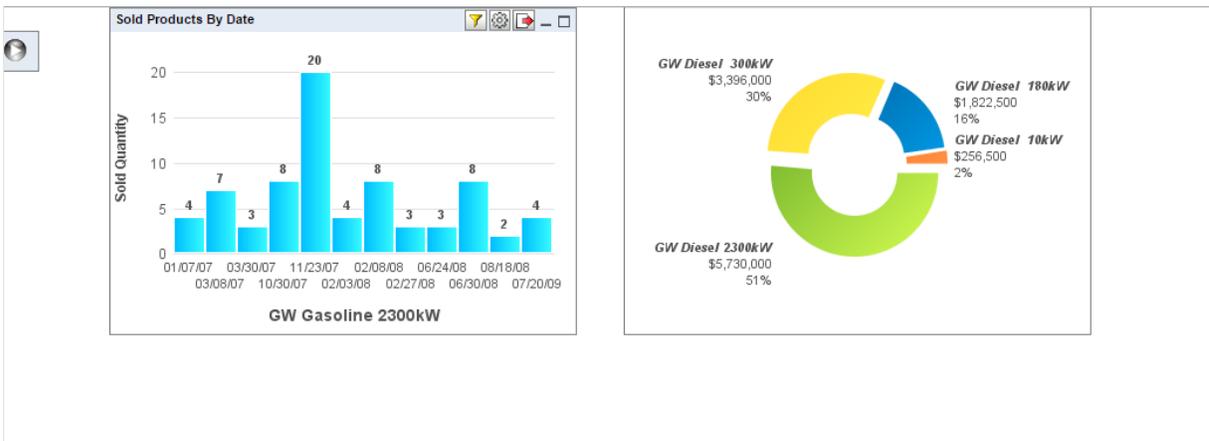
Charts, reports and maps in the dashboard can be moved or resized in free-form. You can resize and move charts, reports and maps using the mouse. To move a chart/report/map, simply click on the header bar of the chart/re-

port/map window and drag it. To resize a chart/report/map, click on the lower right corner of the chart/report/map window and drag on the sizing handle that appears in the lower right corner of the window.

You can see a gray stripe under the Dashboard Builder toolbar. This stripe allows you to place a report/chart/map so that its header bar isn't visible in the preview window. You can see the difference in two images below.



Report/Chart/Map Placement

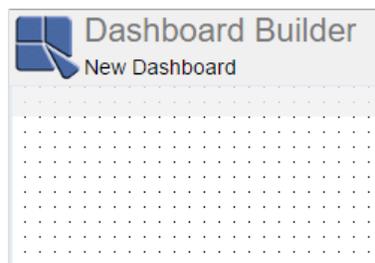


Report/Chart/Map in Preview

If a report, chart or map contains alerts, *Watch alerts* icon  appears in the header bar. To learn more about alerts in dashboards, please visit Section 11.3 - Dashboard alerts.

### 6.2.3.2. Snap to Grid

You can see a grid of dots in Dashboard Builder that helps you to align charts, reports and maps in the dashboard. The grid is visible by default and it snaps all dashboard elements (reports, charts, maps).



Dashboard Grid

You can hide/show the grid by clicking the  *Hide/Show* icon on the toolbar. To position charts/reports/maps completely free, open the *Dashboard Options* dialog by clicking the  *Options* icon and disable the option *Snap to Grid* under *Other* section.

### 6.2.3.3. Chart/Report/Map Toolbar

Each chart/report/map in the dashboard has its own toolbar that allows you to initiate the following actions:

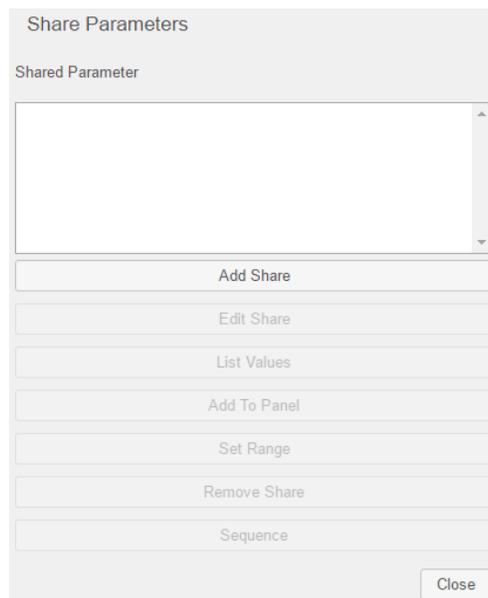
-  **Edit Template** - This button opens appropriate designer that allows you to edit a template.
-  **Refresh Template** - This button refreshes a template.
-  **Add/Modify Link** - This button allows you to use data from one chart/report/map as a parameter in another. (see Section 6.2.10 - Template Linkage)
-  **Add/Modify Drilldown** - This button allows you to add/modify drilldowns. (see Section 6.2.8 - Drilldown)
-  **Filter Data** - This button allows you to specify filters/parameters for a template. (see Section 7.9.2 - Preview Options)
-  **Export** - This button allows you to export chart/report/map. (see Section 7.9.2 - Preview Options)

### 6.2.4. Shared Parameters

Shared parameters allows you to group common parameters from all charts/reports/maps into a single parameter. You can apply a common filter to some or all of the dashboard items at the same time. All of the dashboard's shared parameters are listed in the *Shared Parameter* list in the *Share Parameters* dialog that can be opened by clicking

the  *Share Parameters* icon on the Dashboard Builder toolbar.

For example, create a dashboard by adding all files from the *Examples/Dashboards/CategorySales-Dashboard* node (*OrdersCatRegion*, *ProductSales*, *SalesRegionalSuccess*, *TimeSales*). Then click the *Share Parameters* icon. You will now see *Share Parameters* dialog in the left pane.



*Share Parameters Dialog*

To create a shared parameter, click the *Add Share* button in the Share Parameters dialog. This will open a dialog that allows you to set options for the share.

The screenshot shows a dialog box titled "Add parameter". It contains a "Shared Parameter Name:" text input field at the top. Below this is a list of "Available Parameters" with five entries: "OrdersCatRegion - Category (String)", "OrdersCatRegion - Region (String)", "ProductSales - category (String)", "SalesRegionalSuccess - category (String)", and "TimeSales - category (String)". To the right of this list is an empty "Selected Parameters" list. Between the two lists are two buttons: "Add >>" and "<< Remove". At the bottom of the dialog is a "Prompt Name:" text input field. At the very bottom right of the dialog are "Cancel" and "Ok" buttons.

*Shared Parameter Dialog*

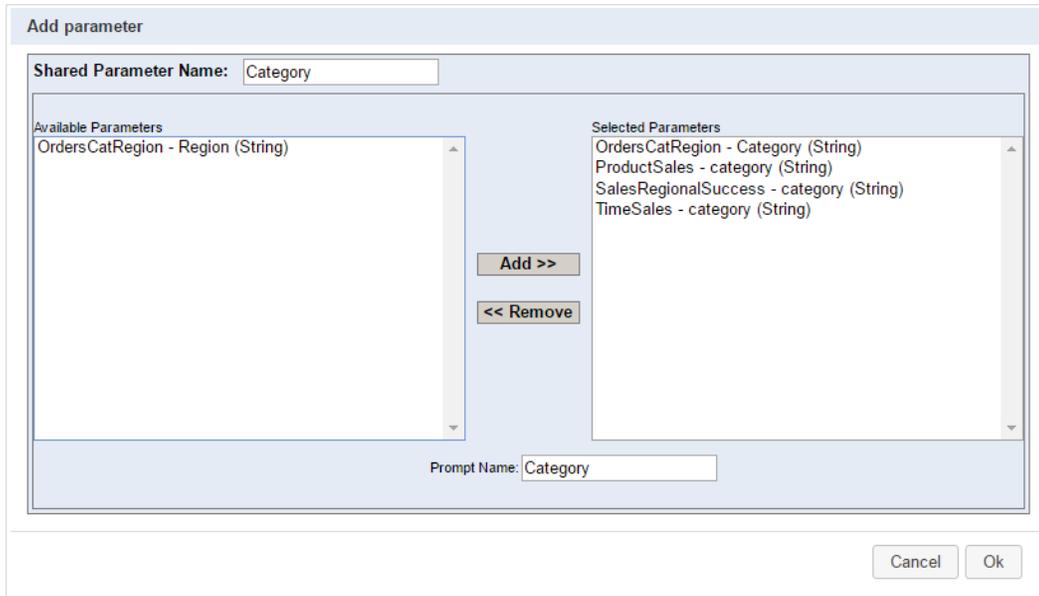
The left side of the dialog shows all parameters for all the selected charts/reports/maps in the dashboard. You can select which parameters you want to add to the share by selecting them in the left side window and clicking the *Add* button. Note that all parameters in the share must have the same data type.

The parameter mapping options (whether the parameter is mapped to a database column) is based on the first parameter selected for a share. For more information about parameter mapping, see Section 3.1.3.2.2.2 - Initializing Query Parameters.

You can enter a prompt name for each shared parameter. Once you finish entering all informations for the shared parameter, be sure to specify a name for it in the field at the top and then click the *Ok* button to save the changes. The parameter will then be added to the *Shared Parameters* list in the share parameters dialog.

You can edit or remove a shared parameter by selecting it in the list and clicking the *Edit Share* or *Remove Share* buttons.

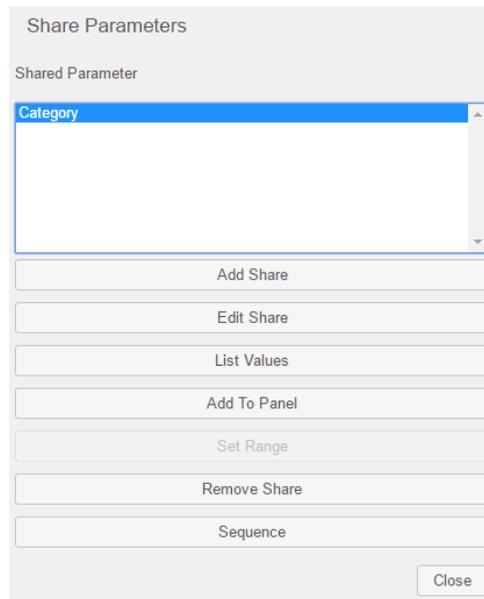
For our example, write *Category* as the *Shared Parameter Name* and the *Prompt Name* and select all available parameters *Category*. Click the *Ok* button to create a shared parameter.



*Shared Parameter Dialog*

### 6.2.4.1. Arrange Shared Parameters

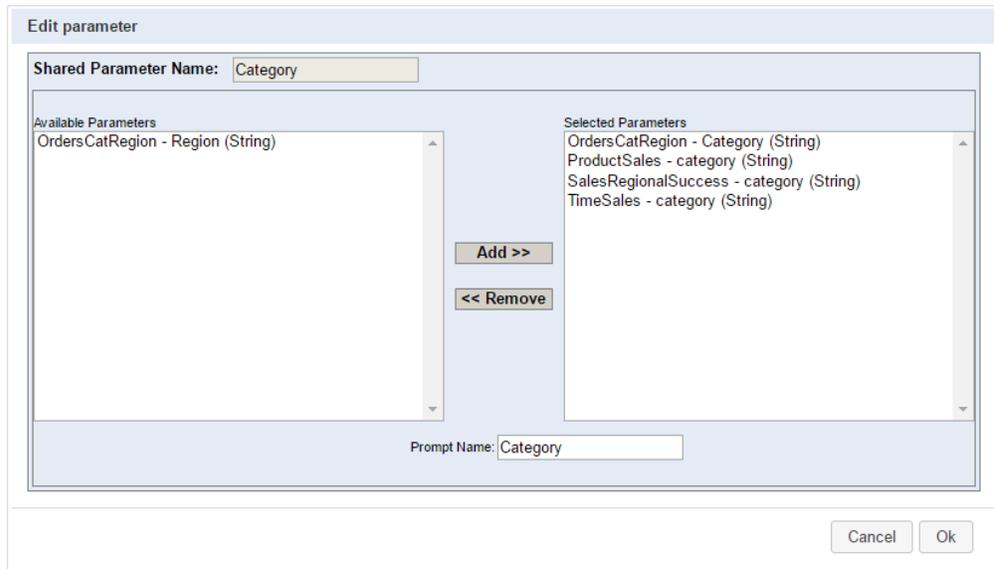
In addition to adding/editing/removing shared parameters, you can also list values of a shared parameter into a set of buttons in the dashboard or arrange a shared parameter to be displayed in the panel that allows you to enter parameter value right from the dashboard. Please note that in order to list the values of a parameter, the shared parameter can be either single or multi-valued and must have predefined selectable values. To add the shared parameters to the dashboard, simply click the *List Values* or *Add to Panel* button in the *Share Parameters* dialog.



*Share Parameters Dialog*

**Edit Share**

Allows you to edit a shared parameter. To do this, select a shared parameter and click the *Edit Share* button. This will open *Edit parameter* dialog.



*Edit Shared Parameter*

**List Values**

If the parameter has a small number of values, you can choose to list the values as buttons or images. For example, let's say your dashboard shows sales data for your products and you have a relatively small number of products. You can select this option to make a list which makes it easier for the users to pick a value for the filter. To list the values for a shared parameter, first select it from the shared parameters list and click the *List Values* button. This will open the parameter value-list panel that contains the parameter values in buttons.



*Parameter Value-list Panel*

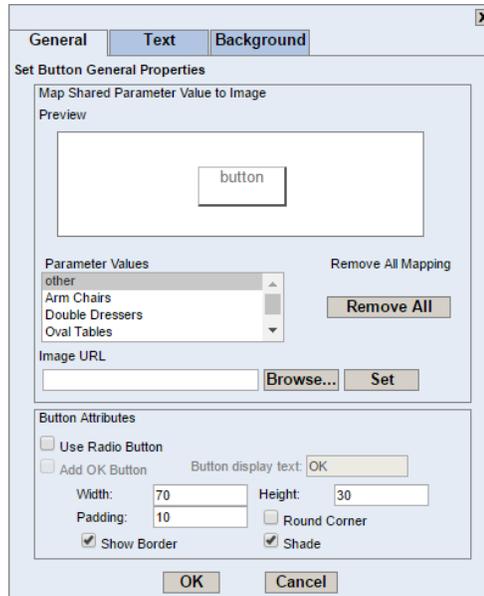
There are four buttons available in the panel header bar.



**Modify Panel Attributes:** This button will open a new dialog that allows you to modify panel attributes such as background color and panel border. For more information about the dialog, please see Section 6.2.5 - Insert Labels.



**Modify Button Attributes:** This button will open the following dialog that allows you to modify button attributes.



*Set Button Properties Dialog*

The first option in this tab allows you to map shared parameter value to an image. To map a shared parameter value, first select it either from the left panel, specify the image URL or browse for it using the *Browse* button and click the *Set* button. In order to remove image URL mapping, select appropriate parameter value, clear the image URL text box and press the *Set* button or use the *Remove All* button to remove all mapping. The other options in the dialog allows you to specify button width, height and padding. You can also choose whether to show the buttons border or whether the buttons should be shaded or not. Another option allows you to display the boxes with a radio button. This option is not available for multi-valued parameters, which are automatically given checkboxes.

The other tabs of the Set Button Properties dialog allows you to set the button text and background. The tabs are basically the same as for labels and they are described in the following Section 6.2.5 - Insert Labels.

The next screenshot shows our example with a modified view of buttons.



*Modified Buttons*



**Delete:** This button will delete the parameter value-list panel.



**Set Buttons Alignment:** These buttons allows you to set vertical or horizontal alignment of buttons. By default, buttons are aligned horizontally and you will only see

the  *Vertical Align* button. When you click on it, the alignment will be changed

to vertical and the  *Horizontal Align* button will appear.



Vertical Buttons Alignment

**Add To Panel** To arrange the layout of shared parameters by adding them to a panel, first select it from the shared parameters list and click the *Add to Panel* button. This will open a parameter panel with the parameter's list box added to the panel.



Parameter Panel

In the Dashboard Builder, you can move/resize the panel or move the small parameter box inside the panel. If you close the panel, you can still use the shared parameter with the preview toolbar (see Section 7.9.1 - Preview Toolbar). If you change the parameter value, the dashboard will be automatically refreshed.

There are three buttons available in the panel header bar:

 **Disable Auto Rearrangement:** This button will disable/enable auto rearrangement.

 **Modify Detach Panel Attributes:** This button will open a new dialog that allows you to modify the parameter panel attributes such as adding an *OK* button, changing panel layout, text label properties, background color and border for the panel. For more information about the dialog, please see Section 6.2.5 - Insert Labels.

 **Delete:** This button will delete the parameter panel.

**Set Range** This option allows you to define a range across two shared parameters. For more information, please see Section 6.2.4.2 - Parameter Range.

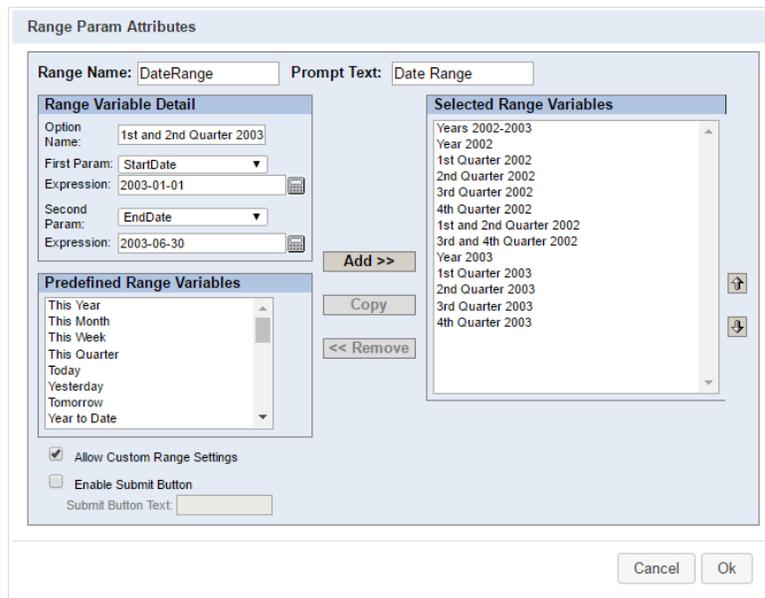
**Remove Share** This option allows you to remove shared parameter. To do this, select the shared parameter and click the *Remove Share* button.

**Sequence** This option allows you to set an order of shared parameters. For more information, please see Section 6.2.4.3 - Cascading Parameters.

### 6.2.4.2. Parameter Range

This function allows you to define a range across two shared parameters. To use this function, select two related shared parameters (such as start and end date, or integer objects representing a range) and then click the *Set Range* button. This will bring you to the *Range Param Attributes* dialog. To create a user defined range variable, simply fill out the *Range Variables Detail* dialog box. To see an example, please see the QuickStart Guide (Section Q.10.1.1 - Create a Dashboard). The next screenshot shows other setting of parameter range for the example created in QuickStart Guide.

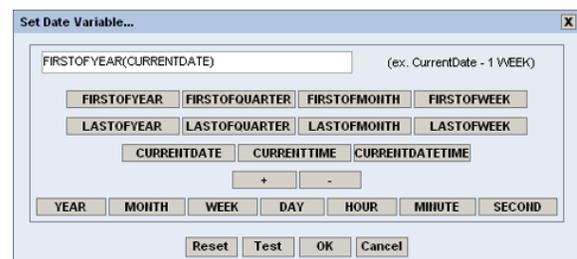
In addition to creating your own range variables, you can also create range variables using *Predefined Range Variables* as described below.



*Parameter Range Dialog*

The dialog has these following options:

- Range Name:** This field allows you to enter a title for the parameter range window.
- Prompt Text:** This text is displayed next to the selected combo parameter dropdown menu.
- Range Variable Detail:** This window has several sub-fields:
  - Option Name:** This is the display name in the *Selected Range Variables* list.
  - First Param:** This is the first parameter in the combined parameter, usually the starting value.
  - Second Param:** This is the last parameter in the combined parameter, usually the ending value.
  - Expression:** There are two expression fields, one for each parameter.  
For date parameters, you can click on the *Date calculator* button  next to the expression field.



In this screen, you can enter one of the three keywords: *CurrentDate*, *CurrentTime*, and *Current-*

DateTime, or one of the function names, such as FIRSTOFTHISYEAR. Details about functions are listed in the list of functions.

You can add or subtract time units from the current date/time, allowing you to have a dynamic date range. For example, a report may have the following default values:

StartDate: CurrentDate - 1 WEEK

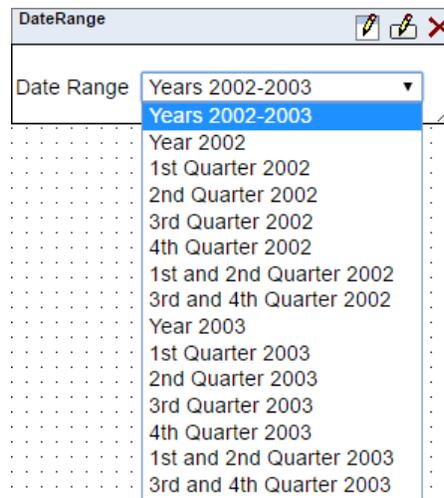
EndDate: CurrentDate

This would indicate that every time the report is run, the default prompt should be one week ago of the current date. Other supported options are YEAR, MONTH, DAY, HOUR, MINUTE, and SECOND. This feature only supports single addition or subtraction and it does not support multi-value parameters.

**Selected Range Variables:**

This window displays all range variables that have been defined. Clicking on any range variable in this list allows you to modify it, delete it or copy it. Once the variable is selected, you can see its detail in the *Range Variable Detail* window where you can modify it. You can move the selected range variable up or down in the list by clicking the   arrows right next to the window.

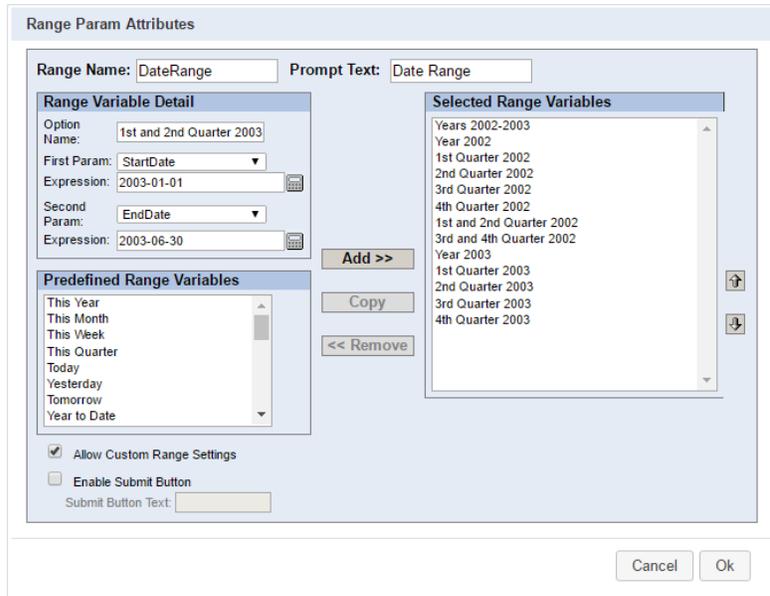
The next screenshot shows a dropdown menu in the dashboard that contains defined range variables (range variables from the *Selected Range Variables* window).



*Range Variables in Dashboard*

**Add:**

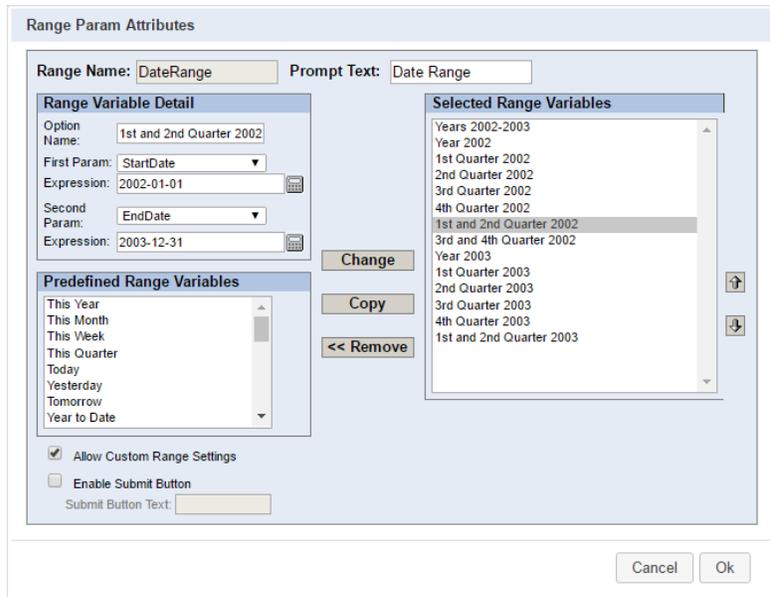
This button has two modes: *Change* or *Add*. When the text of the button reads *Add*, the parameter information in the *Range Variable Detail* window will be added into the *Selected Range Variables* list.



Range Variable Adding

**Change:**

This button has two modes: *Change* or *Add*. Clicking on an item in the *Selected Range Variables* list will change the text to *Change*. When in this mode, clicking the button will save any changes made in the *Range Variable Detail* window to the currently selected range variable.

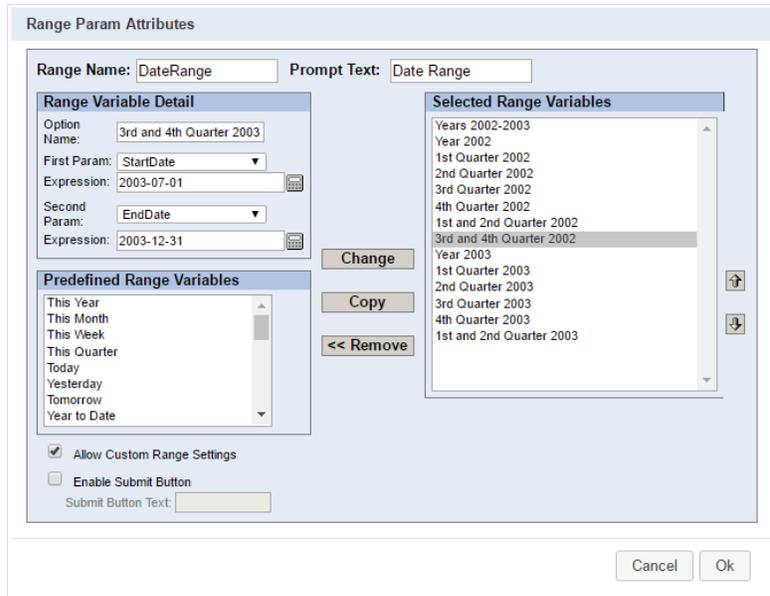


Range Variable Change

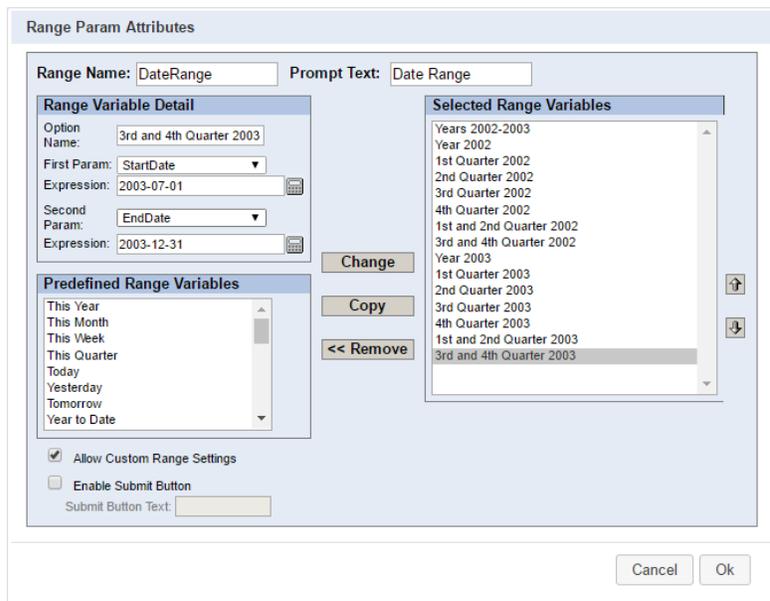
**Copy:**

This button will copy the selected item and add it to the *Selected Range Variables* list under a new name. The new name is created automatically by adding *\_0, \_1, \_2, etc.* or you can change it in the *Range Variable Detail* window. To copy a variable under your name, select it, change a name in the *Range Variable Detail* window and then click the *Copy* button.

*Copy* can help you to create range variables more quickly. For example, you can select a range variable, change only the year and then click the *Copy* button.



*Range Variable before Copy*



*Range Variable Copied*

**Remove:**

This button will remove the selected item from the *Selected Range Variables* list.

**Predefined Range Variables:**

This window contains predefined date range variables for your convenience. It contains commonly used date range expressions such as *This Year*, *This Month*, *This Week*, *This Quarter*, and many more. To use a predefined variable, simply select the needed predefined range variable and click the *Add* button. You can add more than one expression and all of them will be listed under the *Selected Range Variables* window.

**Allow Custom Range Settings:**

This option allows you to specify a parameter range by using calendar and parameter input boxes in the parameter range panel. The following screenshots show the difference between with and without custom range.



*Enabled Custom Range Settings*



*Disabled Custom Range Settings*

**Enable Submit Button**

If you allow *Custom Range Settings*, you can add *Submit Button*. This button allows you to apply custom range when both parameters are set. Without this button, each parameter is applied immediately when it is set. You can also determine a text of this button.

Once you finish setting up the parameter range in the *Range Param Attributes* dialog, click the *Ok* button to close the dialog. A rectangle will follow your mouse pointer around the dashboard design window. Position the rectangle and click. A parameter range panel will appear.



*Parameter Range Panel with Submit Button*

You can select range variables from the dropdown menu where are all variables from the *Selected Range Variables* window, or use customer text boxes (text boxes are available only when *Allow Custom Range Settings* option was checked). The text boxes represent the first and second parameters in the combined parameter. To use customer text boxes, check the radio button next to them and insert parameter values. Note that you can use the calendar to specify the date parameters. To open the calendar, simply click the *Calendar* button  next to the textbox.

There are three buttons available in the panel header bar:

 **Modify Panel Attributes:** This button will open a new dialog that allows you to modify panel attributes such as text label properties, background color and border for the panel. For more information about the dialog, please see Section 6.2.5 - Insert Labels.

 **Modify Range Param Attributes:** This button will open the *Range Param Attributes* dialog that allows you to edit range variables.

 **Delete:** This button will delete the range parameter panel.



**Note**

If you delete this panel, all range variables that were defined in the *Range Param Attributes* dialog will be lost.

**6.2.4.3. Cascading Parameters**

By default, you are prompted to enter all report parameters at once in the prompt dialog. However, this configuration may not be the best solution in case some parameters are mapped to database columns with a significant number

of distinct values. It can be difficult to go through a very large list and you may select values that don't return any data, depending on the parameter combination.

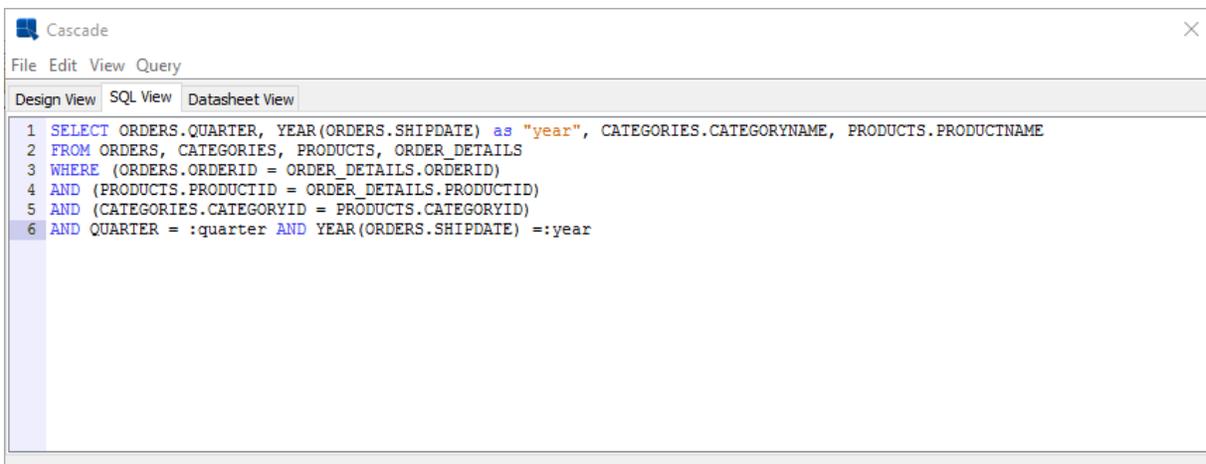
To assist with these problems, ERES provides a feature that allows you to configure the order in which the parameters should be entered. With this feature enabled, you can enter parameters in the dialog in a pre-defined order. As such each selection will be applied as a filter to the next parameter prompt(s). Using cascading parameters can limit the number of distinct values presented and can prevent selecting invalid parameter combinations.

First, create a query named e.g. *Cascade* with parameter sequence in Woodview HSQL database that is in ERES.



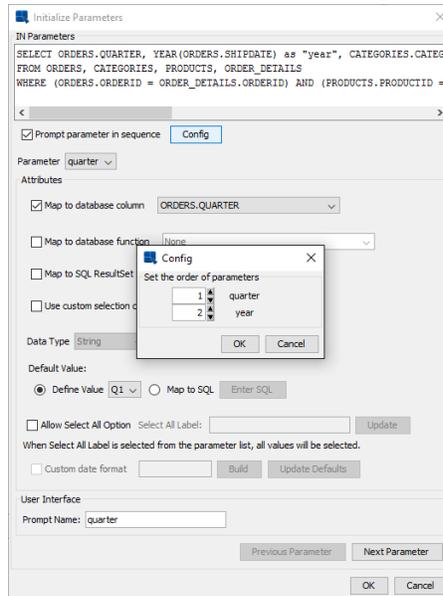
To do this, open Organizer, click the *Manage Data Sources* icon to open the *Data Registry Manager* dialog, select a data registry and click the *Edit* button. Select *Queries* under *Databases/Woodview (HSQL)* node and click the *ADD* button to create a new query. Type *Cascade* as a name and for example, use the following query:

```
SELECT ORDERS.QUARTER, YEAR(ORDERS.SHIPDATE) as "year",
       CATEGORIES.CATEGORYNAME, PRODUCTS.PRODUCTNAME
FROM ORDERS, CATEGORIES, PRODUCTS, ORDER_DETAILS
WHERE (ORDERS.ORDERID = ORDER_DETAILS.ORDERID)
AND (PRODUCTS.PRODUCTID = ORDER_DETAILS.PRODUCTID)
AND (CATEGORIES.CATEGORYID = PRODUCTS.CATEGORYID)
AND QUARTER = :quarter AND YEAR(ORDERS.SHIPDATE) =:year
```



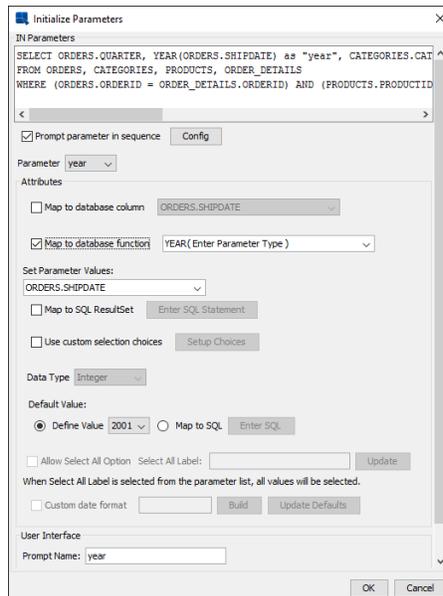
*Cascading Parameters Query*

Now initialize query parameters. Check *Prompt parameter in sequence* and click the *Config* button. Set the parameter sequence according to the image below, i.e. the *quarter* parameter will be in the first level and *year* in the second level. Then map the first parameter *quarter* to database column *ORDERS.QUARTER*.



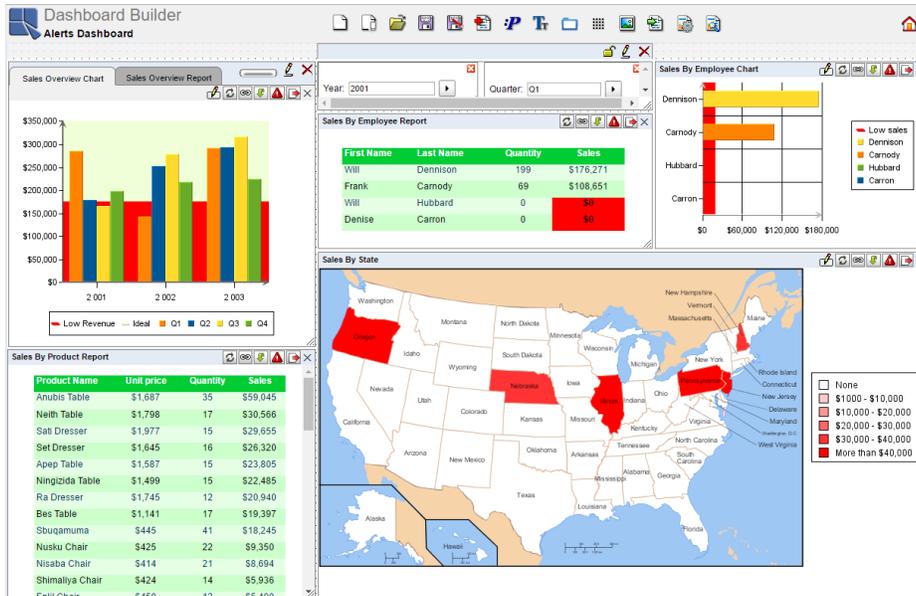
*Initialize First Parameter*

Map the second parameter *year* to database function *YEAR* and *Set Parameter Values* to *ORDERS.SHIPDATE*. Then click *OK* to save the query.



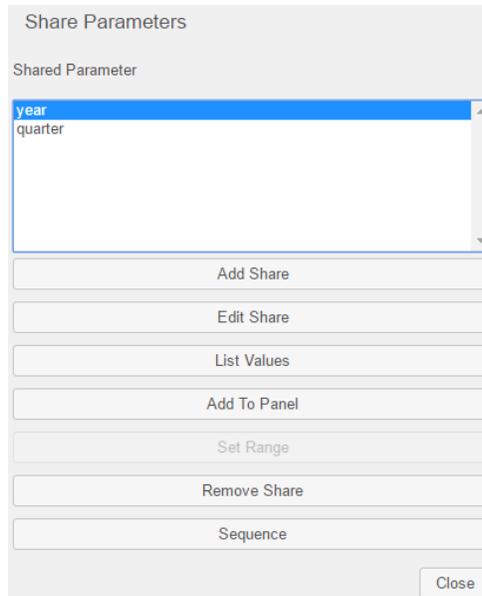
*Initialize Second Parameter*

Go to Dashboard Builder and open the *Alerts* dashboard (Published Files/Examples/Dashboard-s/Alerts Dashboard).



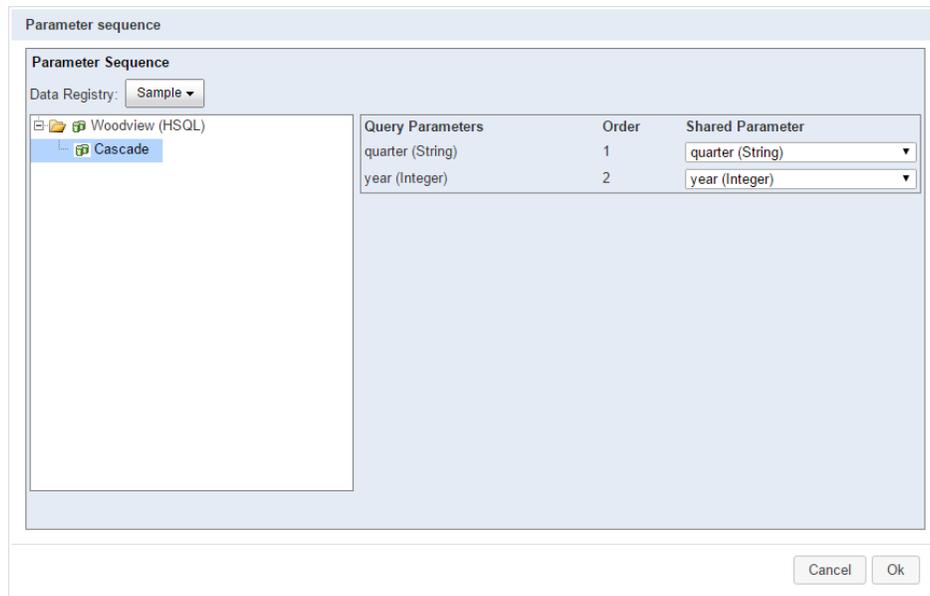
Alerts Dashboard

Then click the  *Share Parameters* icon on the Dashboard Builder toolbar to show the list of shared parameters in the dashboard (Notice that there are two shared parameters *quarter* and *year* already added in a dashboard. You can remove them to make room for the cascading parameters). From the dialog, select one of parameters to activate options and click the *Sequence* button at the bottom.



Shared Parameters Setting

The *Parameter Sequence* dialog will then appear. From the query tree on the left, select *Cascade* query you have created in previous steps. It should automatically map query parameters to shared parameters in the dashboard.



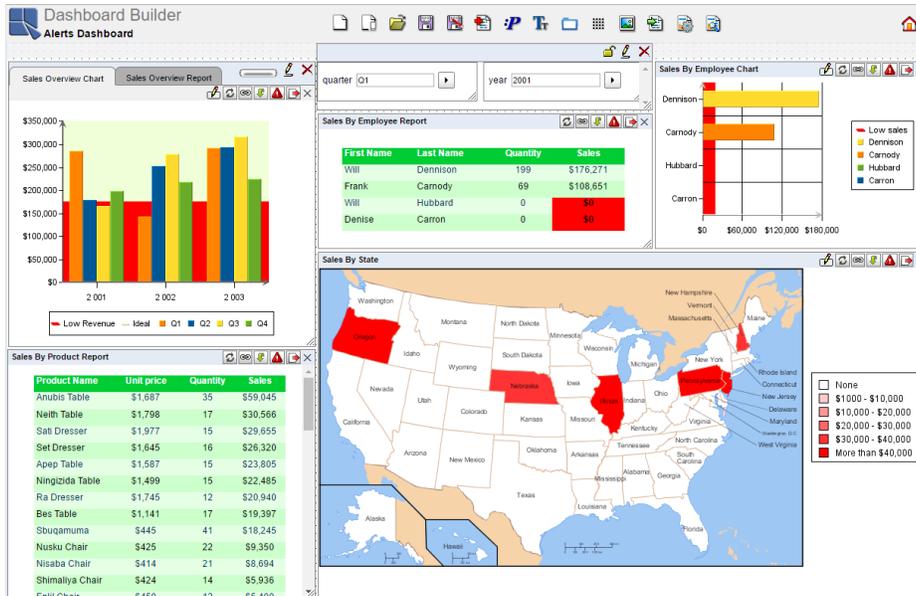
*Parameter Sequence Dialog*

Click the *OK* button and place the parameter panel into a dashboard.



*Parameters Panel*

The image below shows inserted cascade parameters panel in the dashboard. Now you can specify a parameter value for the first and second level. Once you specify the last parameter in the sequence, the dashboard templates should refresh according to the selected values.



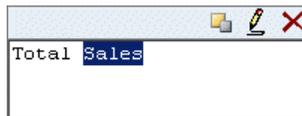
Dashboard with Cascade Parameters Panel

## 6.2.5. Insert Labels

You can insert a label into a dashboard by clicking the *Insert Label* button  on the dashboard builder toolbar (Please note that labels are not available for responsive dashboards). After you click the button, a small rectangle will follow your mouse pointer around the builder interface. Position the rectangle where you want to insert the label and click. The label will be inserted into the dashboard. You can edit the label text by double-clicking on the label body.



Inserting a Label



Editing a Label

Labels can be moved or resized in the same way as charts, reports and maps. To move a label, simply click on the header bar of the label window and drag it. To resize a label, click on the lower right corner of the label window and drag on the sizing handle that appears around the lower right edge of the window.

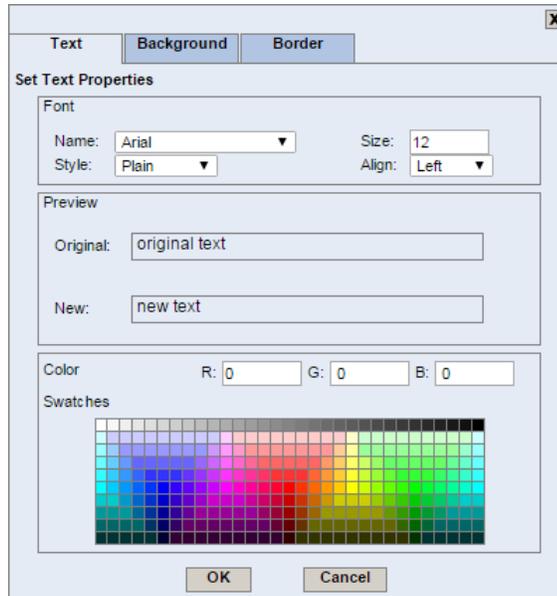
There are three additional buttons available for a label in the label header bar.

 **Move to Back:** This button will move the label to back.

 **Edit:** This button will open a new dialog that allows you to specify text label properties, background color and border for the label.

### Text Label Properties:

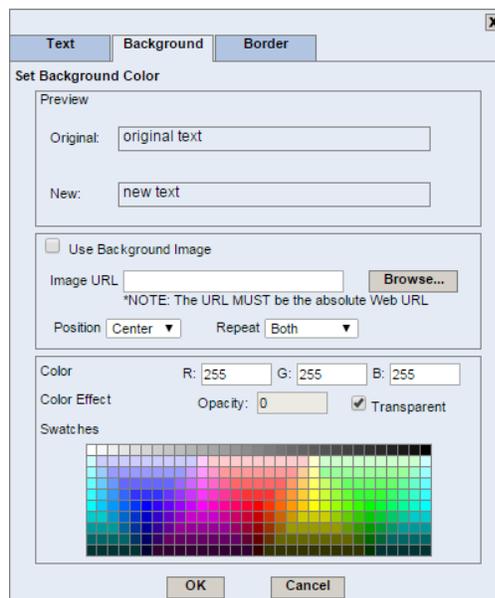
The first tab of the edit window allows you to adjust text label properties. You can adjust the font type, font style, font size and alignment of the label as well as changing font color using swatches or RGB values. In the preview section you can still see the original and new text.



*Label Text Properties*

**Label Background Color:**

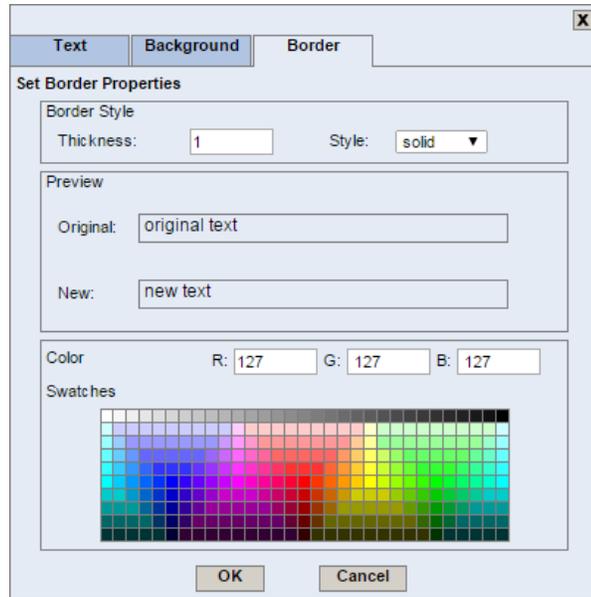
The second tab of the edit window allows you to set label background color or to use background image. The background color can be adjusted using swatches or RGB values. You can also specify the label opacity or choose whether or not to set the label background transparent. If you want to use a label background image, simply check the *Use Background Image* checkbox and enter the image URL or use the *Browse* button to browse to the appropriate background image. Please note that the URL must be absolute Web URL. In addition to adding background images, you can also specify the background image position (available positions are: center, left, right, top, bottom and fixed) or choose whether to set the dashboard background image to be repeated (available options are: both, horizontal, vertical and none).



*Label Background Color*

**Border Properties:**

The third tab of the edit window allows you to set label border style and border color. The label border can be adjusted using swatches or RGB values. If you want to use border style, specify the border thickness and select appropriate border style (available options are: solid, dotted, dashed, double and groove). In the preview section you can still see the original and new text and border.



*Label Border Properties*

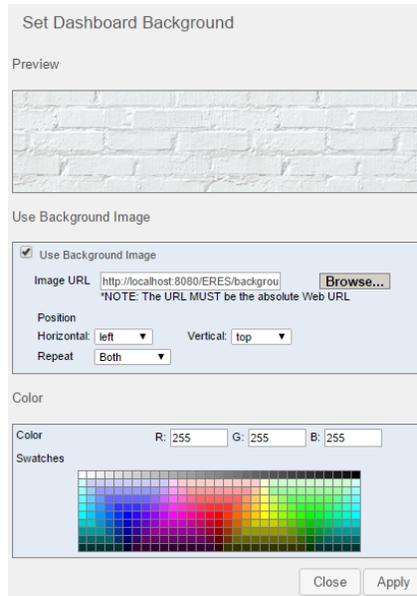
**✗ Delete:** This button will delete the label.

Please note that the labels can also contain any HTML tags, which may override external settings from the dialog.

For example, let's say you have a label with the font type set to Arial. However, once you enter HTML tag `<font face="Garamond">Label content</font>` into the label, Arial font will be overridden with Garamond font for the Label content text.

## 6.2.6. Add Background

You can also add dashboard background by clicking the *Add Dashboard Background* button  on the dashboard builder toolbar. This will open a dialog that allows you to set the dashboard background.



*Dashboard Background Dialog*

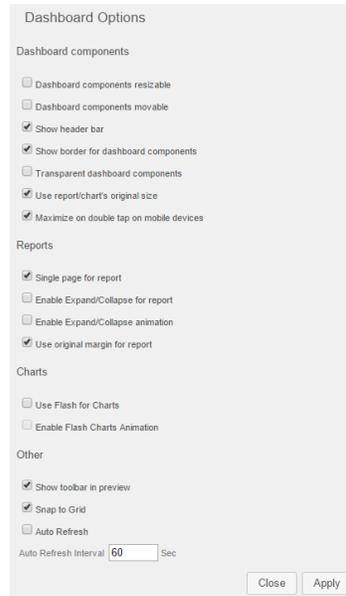
The first option in the dialog allows you to use a background image for a dashboard. If you want to use a background image for the dashboard, simply check the *Use Background Image* checkbox and enter the image URL or use the *Browse* button to browse to the appropriate background image. Please note the URL must be the absolute Web URL. In addition to adding background images, you can also specify the background image position (available positions are: center, left, right, top and bottom) or choose whether to set the dashboard background image to be repeated (available options are: both, horizontal, vertical and none). In the preview you can still see updated dashboard background.

The second option in the dialog allows you to set dashboard background color. The background color can be adjusted using swatches or RGB values.

Once you finish setting up the dashboard background options, click the *Apply* button to apply the changes. You can close this dialog by clicking the *Close* button or by clicking the *X* sign in the upper right corner.

## 6.2.7. Additional Options

You can also specify additional dashboard options by clicking the *Options* button  on the dashboard builder toolbar. This will open a dialog that allows you to set the dashboard components resizable/movable and other options.



*Dashboard Options Dialog*

**Dashboard components resizable:**

This option allows you to resize the dashboard components. It doesn't affect the Dashboard Builder as it works in dashboard viewer (i.e. when you open a dashboard from MenuPage etc.) and dashboard preview only.

**Dashboard components movable:**

This option allows you to move the dashboard components. Like the previous option, this one also affects deployed dashboards only.

**Show header bar:**

Shows/hides header bars for templates in the dashboard. Affects deployed dashboards only. In Dashboard Builder, headers are always visible.

**Show border for dashboard components:**

Shows/hides borders for templates in the dashboard. Affects deployed dashboards only. In Dashboard Builder, borders are always visible.

**Transparent dashboard components:**

Enable transparent background of dashboard components (e.g. transparent background of chart/report/map title, legend).

**Use report/chart's original size:**

This option only affects charts/reports at the moment you add them to the dashboard. If this option is enabled, charts/reports will be added to the dashboard using their original size settings (i.e. canvas size for charts and page size for reports). If this option is disabled, all reports/charts will be resized to standard size and their original size settings will be ignored. This option doesn't affect charts/reports after they have been added to the dashboard so you can resize them as you like.

**Maximize on double tap on mobile devices:**

Enable this option to maximize a dashboard template by double tapping on mobile devices.

**Single page for report:**

This option allows you to set whether reports will be displayed in single page or multi-page format

**Enable Expand/Collapse for report?**

Enables Expand and Collapse function for reports. To learn more about this feature, see Section 4.1.5.2 - Exporting Reports chapter.

**Enable Expand/Collapse animation:**

Enable animation for Expand and Collapse function.

**Use original margin for report:**

This option will set reports to be displayed with their original margins as defined in the Page Setup in the Report Designer. If this option is disabled, report margins are set to 0.2 inches.

- Use Flash for Charts:** Use Flash for charts instead of the default PNG format.
- Enable Flash Charts Animation:** Enable/disable animation for Flash charts.
- Show toolbar in preview:** Hides/displays toolbar in deployed dashboards. For more information about the preview toolbar, please see Section 7.9.1 - Preview Toolbar.
- Snap to Grid:** Snaps components to grid when resizing/moving.
- Auto Refresh:** Enables auto refresh for deployed dashboards (this option doesn't affect the Dashboard Builder - it affects dashboard viewer and dashboard preview only). In already deployed dashboards, auto-refresh can be also enabled/modified from the preview toolbar.

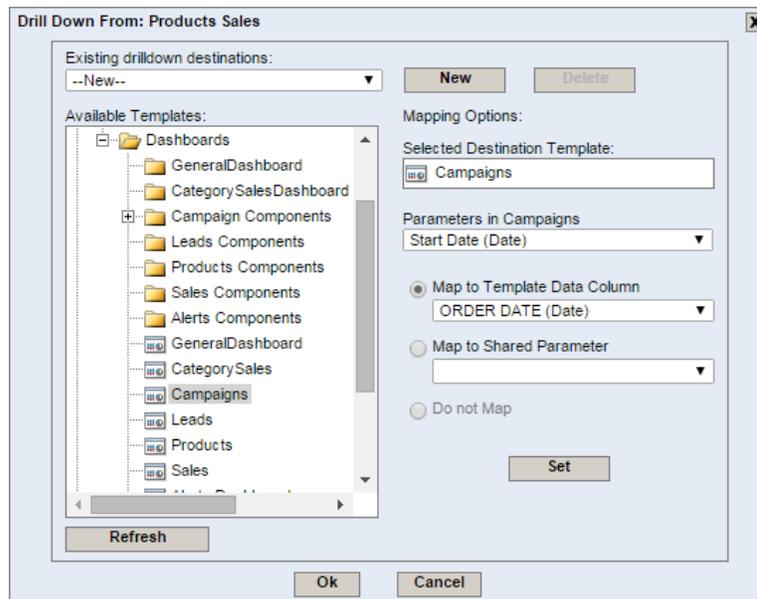
Once you finish setting up the dashboard options, click the *Apply* button to apply the settings. You can close this dialog by clicking the *Close* button or by clicking the *X* sign in the upper right corner.

## 6.2.8. Drilldown

You can add a drilldown to any chart/report/map in the dashboard. The drilldown feature allows you to map the chart data points, report columns, and/or shared parameters to the lower-level chart or report parameters. This creates a relationship where an end-user can either click on a data point in a dashboard chart, or row in a table and be taken to a lower-level report or chart that shows detailed information for the selected point.

You can specify more than one drilldown relationship within a template (chart/report/map) although you can only have a maximum of two drilldowns for charts and SVG maps. For a report, any column can be used as a drilldown anchor.

To add a layer of drilldown, click the  *Add/Modify Drilldown* button in the chart/report/map header bar. A dialog will open that allows you to configure drilldown options for the chart, report or map.



*Drilldown Options Dialog*

The first option in the dialog, Existing drilldown destinations, is a dropdown selection box that will contain a list of the drilldowns that have been set in the template. The first element is always *--New--*. You can select a drilldown from the list and the component information will be filled automatically. If you choose *--New--*, all dialogs will be cleared and new entries into the dialogs can be entered. The items in the list are names of the drilldown targets with an icon that can help you to identify the type of the file (see Section 2.1.4.3 - File Identification Icons). The *Delete* button will only be enabled when you select an existing drilldown from the *Existing drilldown destinations*. When you hit this button, the current selected drilldown will be removed from the list and the whole dialog will be reset.

The field on the left side of the dialog contains a tree that mirrors the Organizer folder structure. All of the reports/charts/maps and dashboards to which you have access to will be listed. Please note that drilldowns to dashboards will be discussed in the next section.

To select a chart, report or map as the drilldown destination (lower-level), click on it in the list. The dialog will update to show the name of the currently selected drilldown report/chart/map. The first drop-down list below the name will be populated with all of the parameters in the selected report/chart/map.

Each parameter in the destination report/chart/map can be unmapped (i.e. it does not pass the parameter value and shows the drillable link no matter where you click on the report/chart) or mapped to either a column (data element) in the top-level chart/report/map, or one of the shared parameters defined in the dashboard. To set an unmapped link, simply select the *Do not Map To Column*. Note that this option is only available if the next drillable link is a dashboard. To set the parameter, first select the destination parameter in the drop-down list. Then select the radio button to indicate whether it should be mapped to the top-level chart/report/map or a shared parameter. Then select the element/field or shared parameter from the appropriate drop-down list.

Note that a column selected for the drilldown in the *Map to Template Data Column* can only be used once, i.e. you can have only one drilldown per column. A shared parameter in the *Map to Shared Parameter* can be used multiple times in the dashboard. This is because a shared parameter isn't tied to a specific template in the dashboard and it behaves like a parameter value provider for the drilldown destination template.

If all destination template parameters are mapped to shared parameters and if the parent template is a report, the first column of the report is used as the anchor for the drilldown link, although it won't pass the element value to the destination template. If the parent template is a chart or a SVG map, the data points become the anchor for the drilldown links.

You can click the *Set* button to save the drilldown settings that is currently being worked on. If the drilldown is new, a new entry in the *Existing Drilldown destinations* will be added, else the drilldown settings are updated in the browser memory. To make the saved settings permanent, click the *OK* button.

The *Refresh* button reloads the folder tree with latest organizer folder list. Note that all folders will be shown in a collapsed view.

Once you finish selecting the options, click the *Ok* button to save the changes. You will be returned back to the Dashboard Builder interface.

### 6.2.8.1. Drilldown to Dashboard

In addition to drilldown to any chart/report/map in the dashboard, you can also map the chart data points, report columns, and/or shared parameters to another dashboard. The dashboard builder supports both parameterized and non-parameterized dashboard as the destination of the drilldown. If the target dashboard has no parameter or you don't setup any parameter mapping, the template from which the drilldown link starts will become clickable and the result of the click will popup the target dashboard in a new window without any default parameter setup.

In case you want to set a parameterized dashboard to be a drilldown destination, you will need to map shared parameter(s) from the target (child) dashboard to chart data points/report columns or shared parameters in the mother dashboard. It will behave like setting up a drilldown link for a report, chart or map, as mentioned before. The corresponding field(s) will then be changed to a hyperlink and when you click the link, the target dashboard will pop up and the shared parameter of the target dashboard will be updated.

You can set up a drilldown link without any parameter mapping to another dashboard. If the template has both unmapped and mapped drilldowns and is a report or chart, the drilldown that appears depends on which point on the report/chart was clicked. If it is a datapoint that is used as a drilldown link for a mapped drill-down, a mapped drillable link will appear. If not, an unmapped drillable link will appear.

The other drilldown functionalities are basically the same as mentioned in the previous section. For more details about setting drilldowns and the drilldown options dialog, please navigate to the previous Section 6.2.8 - Drilldown.

### 6.2.9. Folders

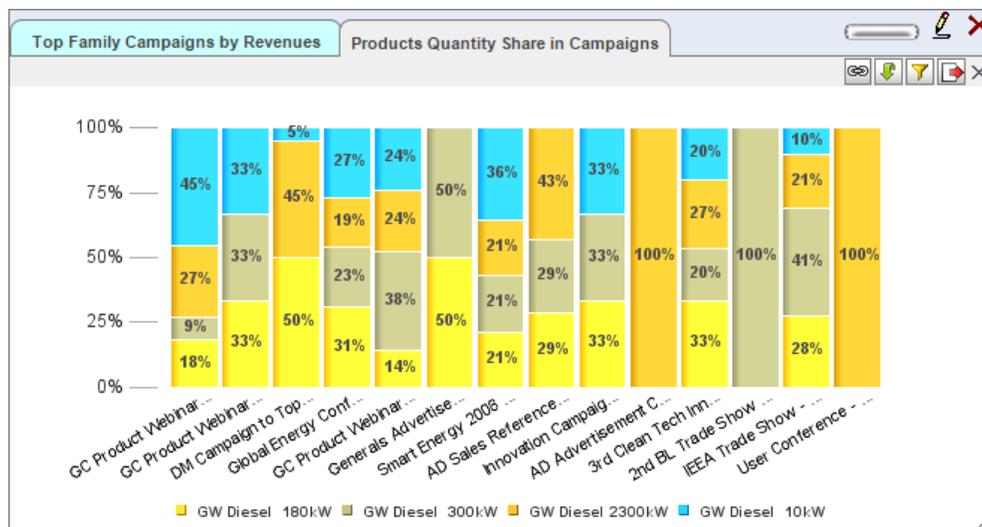
Folders allow you to place tabbed objects into your dashboard. Folders can contain charts, reports or maps, and use a tabbed interface that allows you to switch between current objects to display. You can insert a folder into

a dashboard by clicking the *Insert Folder* button  on the dashboard builder toolbar. After you do so, a small rectangle will follow your mouse pointer around the builder interface. Position the rectangle where you want to insert the folder and click. The folder will be inserted into the dashboard. To move a folder, click and drag your mouse over the right side of the folder's header bar.



*New Folder*

After adding a folder to the dashboard, you can drag and drop reports, charts and maps into it. After you add an object to a folder, a new tab will be visible inside the folder, showing the names of the objects within it. To switch between objects, simply click on the tab of the object you wish to view. Objects can be removed from the folder by clicking and dragging on the object's title bar.



*Folder with objects and style formatting*

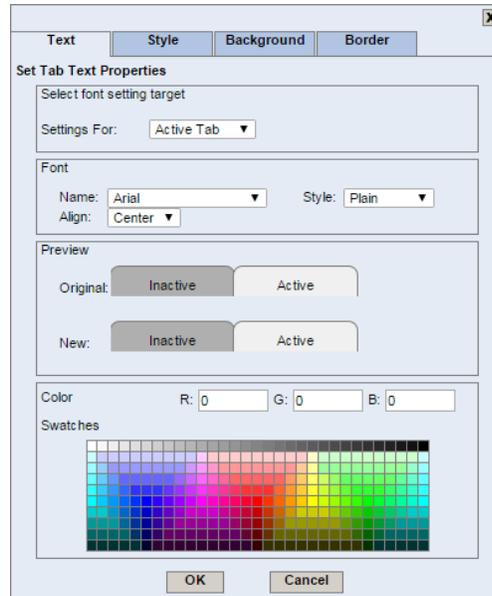
There are two additional buttons available for a folder in the folder header bar:



**Edit:** This button will open a new dialog that allows you to specify text label properties, style, background color and border for the folder tabs.

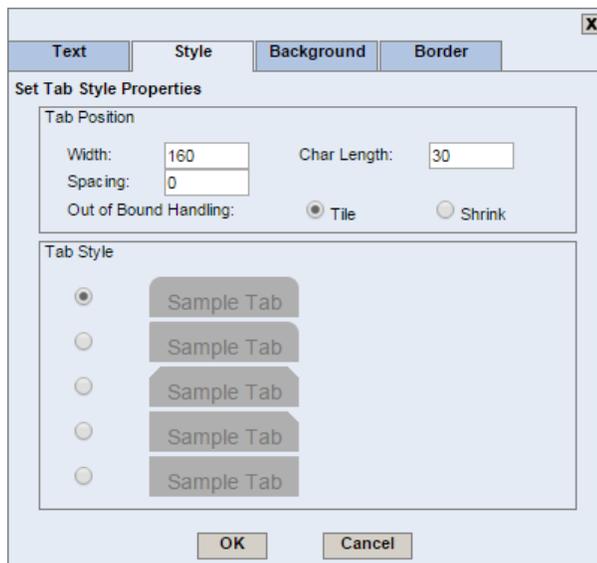
**Text Properties:**

The first tab of the edit window allows you to adjust text label properties. You can adjust the font type, font style, font size and alignment of the label as well as changing font color using swatches or RGB values.



**Style Properties:**

The second tab of the edit window allows you to set style properties. You can adjust tab shape and position.



**Width:** This option allows you to set a width of tabs in pixels.

**Char Length:** This option allows you to set character length. Shortened file names will end with three dots (included in the length).

**Spacing:** This option allows you to set a space between tabs in pixels.

**Out of Bound Handling:** **Tile** - The width of the tabs will be kept even if the

whole tabs are not visible.



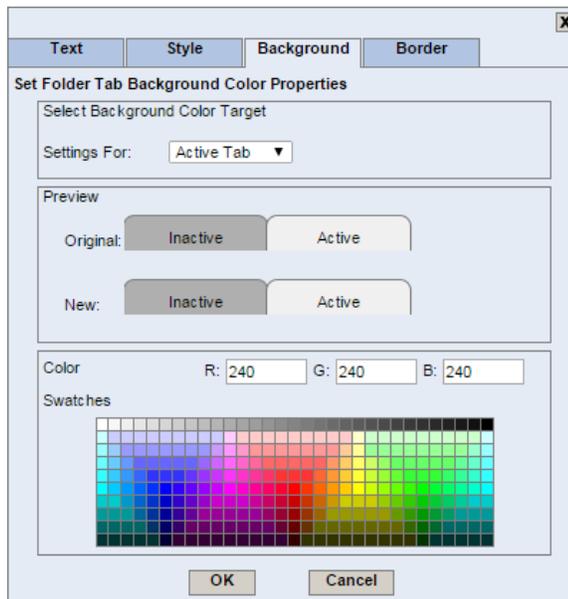
**Shrink -**

The width of the tabs will adjust to make the whole tabs visible.



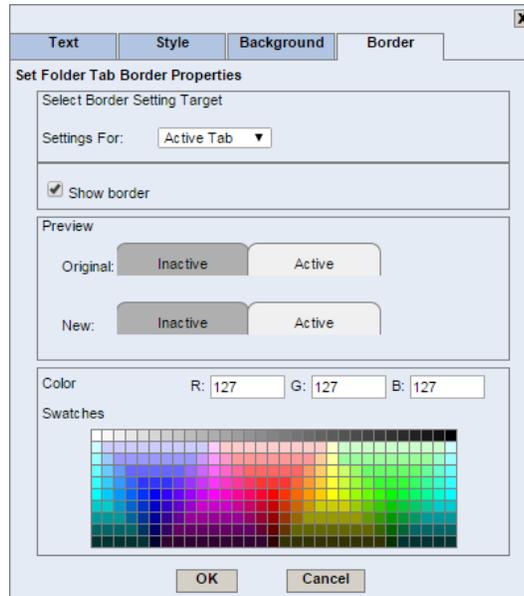
**Background Properties:**

The third tab of the edit window allows you to set the background color for either active or inactive tabs. The background color can be adjusted using swatches or RGB values.



**Border Properties:**

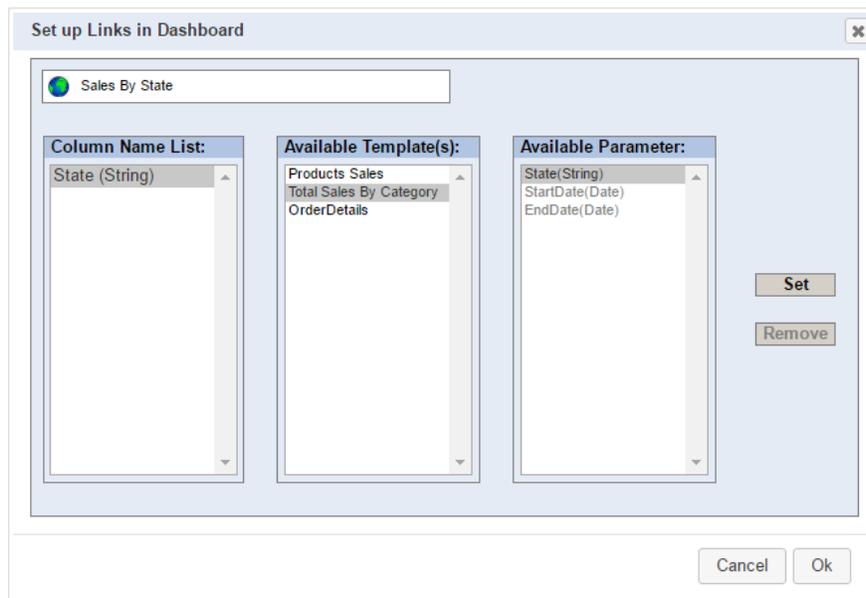
The fourth tab of the edit window allows you to set the tab border color. The tab border can be adjusted using swatches or RGB values.



**✗ Delete:** This button will delete the folder.

## 6.2.10. Template Linkage

Template Linkage allows you to use data from one chart, report or map as a parameter in another. To use this feature, click the *Add/Modify link* button  on the source chart, report or map to bring up the *Set up Links in Dashboard* screen.



On this screen, there are three columns:

**Column Name List -** This is a list of available columns and their data type in the source template. Click on a value from this column to use it as a parameter in another template.

<b>Available Template(s) -</b>	This lists the other templates that are open in the current dashboard. Click on a template name to select it. While a template is selected, it will be highlighted in the dashboard.
<b>Available Parameter -</b>	After selecting a column name and template, this list of available parameters will appear. Select a parameter you want to map to the selected column name in the first column.

When you are done selecting values in each column, click the *Set* button. You can then select new fields if you wish to add additional links. Once a link has been established, clicking on a value in the specified source column (or chart object) will cause the target templates to be refreshed using data from that column.

To remove a link, select it and then click the *Remove* button.

## 6.2.11. Dashboard Preview

At any point in the design process, the dashboard can be previewed by clicking the *Preview* button  on the dashboard builder toolbar. This will open a new window showing the dashboard. Dashboards that have been deployed to the Organizer can also be viewed in the Menu Page.

For more information about dashboard preview options, please see Section 7.9 - Dashboard Viewer.

## 6.3. Dashboard Migration

Often you may need to move dashboards from one location to another. For example dashboards may move between the server and the develop machine. On the develop machine the reports/charts/maps locations can change and may not be located from the data stored in the dashboard .dsb file. Therefore, ERES uses the DPAK files to move or archive dashboards along with it's components. To learn how to create a DPAK file, see the Section 2.1.4.5 - Dashboard and Map Packages.



### Note

The DPAK files can be viewed in Published Files, but they can't be open in the Dashboard Builder. To edit a dashboard from a DPAK file, you have to unpack it first (to learn how to unpack a DPAK file, see the Section 2.1.4.5 - Dashboard and Map Packages).

### 6.3.1. Migrating from previous ERES versions

DPAK files can't be created in ERES version 6.3 and older. If you want to move a dashboard from an old ERES version (6.3 or older) to a new ERES version (6.6 or newer), you will have to use the "Dashboard Migration" feature.

To use this feature, you will need to copy the following files from source ERES server to the corresponding files on the destined ERES server :

- dashboard file from <InstallDirectory>/DashboardFiles
- report files from <InstallDirectory>/ReportFiles (can be a stand-alone report or behave as a drilldown target)
- chart files from <InstallDirectory>/ChartFiles (can be a stand-alone chart or behave as a drilldown target)
- drilldown reports/charts from <InstallDirectory>/DrillDown (for drilldown templates that were created in the drilldown wizard)
- sub-reports from <InstallDirectory>/SubReports (for sub-reports that were created in the sub-report wizard)
- charts embedded in reports from <InstallDirectory>/chart
- map files from <InstallDirectory>/MapFiles with all the files used by this map (i.e. Coordinates file, Tooltip template and DrillDown template for Google Maps or SVG Image and DrillDown template for SVG Maps). If

you are not sure about what files are used by your map, you can view the map file and search for filenames. Map files are XML files, so they are human-readable. Please note that all the files used by your map should be copied in the same subdirectories of the ERES installation directory on the destined ERES server. Otherwise, you will have change the file links in the map file manually.

- necessary images which are used in dashboard, but not available from the destined machine

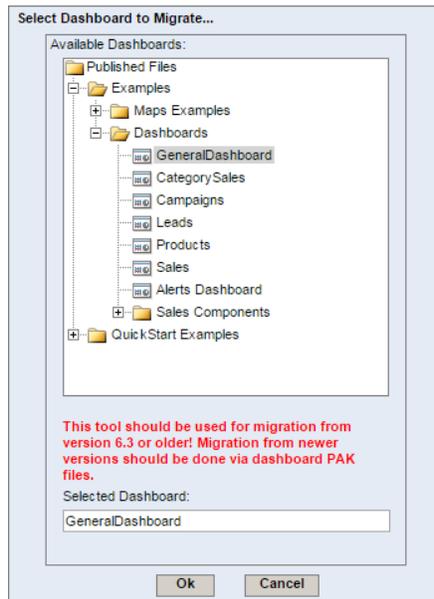


**Note**

The directories that are mentioned in the above list are the default directories that are used for saving certain file types. If you saved a template to a different directory or if you inserted some files into Organizer from a custom directory, you will have to search for the files in their original locations.

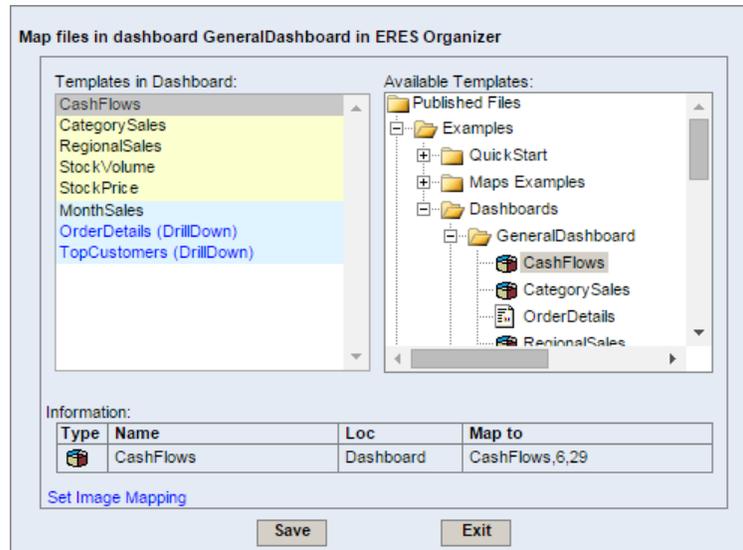
Next, you will need to add previously mentioned files to the destined Organizer and then launch the dashboard builder.

To migrate a dashboard from the Dashboard Builder, click the *Migration* button  on the Dashboard Builder toolbar. After you have clicked the button, the following dialog will then appear.



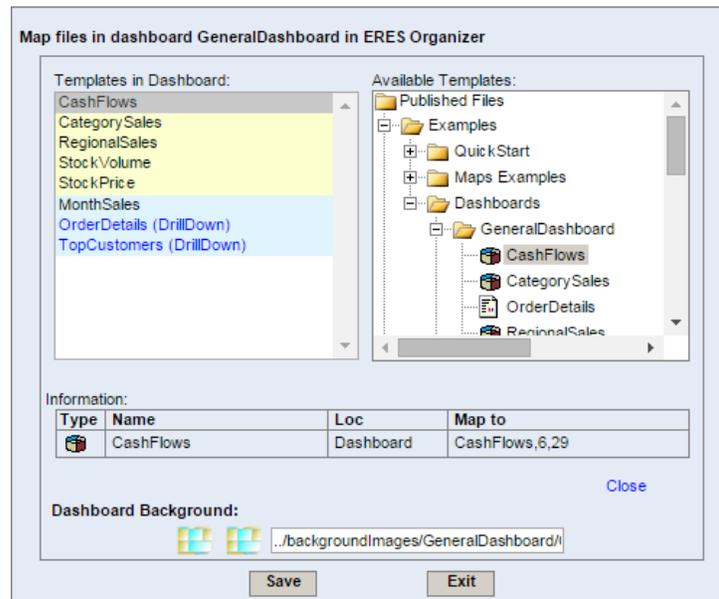
*Dashboard migration dialog*

The dialog contains a tree that mirrors the folder structure in the Organizer. All of the dashboards to which you have access are listed. In order to select a dashboard to migrate, simply select it in the tree and click *Ok*. The migration mapping dialog will then appear.



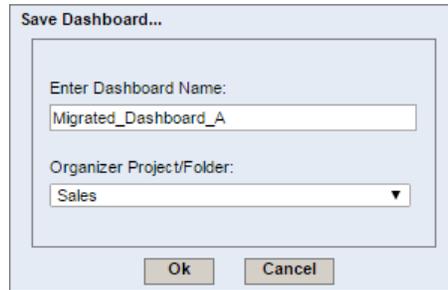
Dashboard migration dialog

From this dialog you can map the original reports/charts/maps to the reports/charts/maps in the destined Organizer. Please note when your dashboard uses images (dashboard background, shared param panels background, shared param value list button icons, etc.), which are not available from the destined machine, you need to set image mapping as well. To set the image mapping click *Set Image Mapping* link at bottom of the dialog. The dialog will then extend to allow you specifying/changing the image mapping.



Dashboard migration dialog

All the images that were used in the original dashboard will be listed with their URLs. You can see the preview of the original (the first image) and current mapped image (the second image). If an image is not correctly mapped, the second preview image will be blank. Once you have finished specifying the mapping for the dashboard files/images click *Save*. It will open the following dialog prompting you to enter the name for the migrated dashboard.

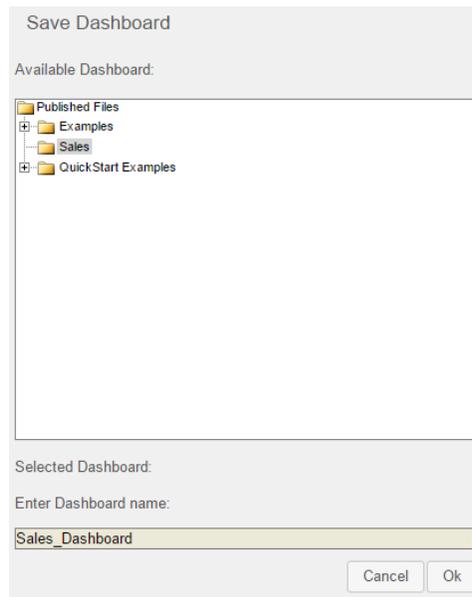


*Dashboard migration dialog*

Once you have finished specifying the options click *Ok* and the dashboard will then be saved.

## 6.4. Save Dashboard

Once you have finished setting up all the dashboard definitions, you can save the dashboard by clicking the *Save* button  on the toolbar. A new dialog will open prompting you to specify the name for the dashboard.



*Save Dashboard Dialog*

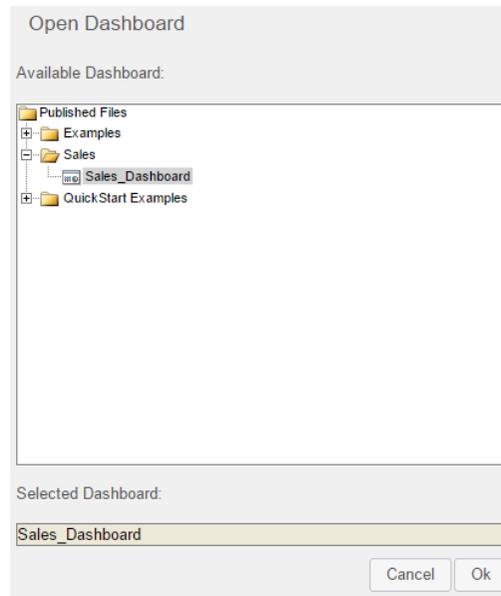
The first option allows you to specify a name for the dashboard. The dashboard will be saved with a `.dsb` extension in the `/DashboardFiles/` directory of your ERES installation.

The second option allows you to insert the dashboard into the Organizer. The drop-down list contains all of the projects and folders in Organizer to which you have access. Select the project or folder in Organizer where you'd like to place the dashboard.

Once you have finished specifying options, click the *Ok* button to save the file.

## 6.5. Open Saved Dashboard

You can open the saved dashboard by clicking the  *Open* icon on the main toolbar. *Open Dashboard* dialog will appear in the left pane.



*Open the Dashboard*

You can see all dashboards from Organizer created in Dashboard Builder. Select a dashboard and click *Ok* to open it.

## 6.6. Exit

You can exit Dashboard Builder by clicking the  *Home* icon in the upper right corner, by clicking the *Dashboard*

*Builder* title, or by clicking the  *Logo* icon in the upper left corner. Before closing, you will be asked if you want to save an unsaved dashboard.

# Chapter 7. Publishing (Menu & URLs)

## 7.1. The Menu Page

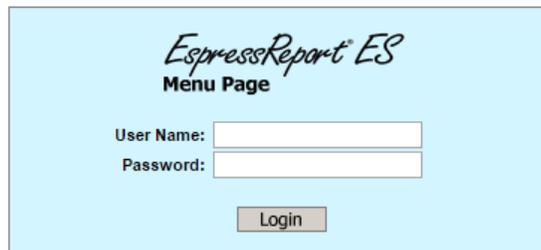
In ERES, all reports, charts, maps and dashboards created or added to the Organizer are automatically published. By logging into the menu page, users can access and run any files to which they have privileges in the Organizer. The menu page is included with ERES and deploys without any code.

### 7.1.1. Launching the Menu Page

The menu page can be launched from the ERES start page. Once you've successfully logged in, click the *Published Files* link.

The menu page can be also accessed directly by going to the menu page login: `Menu_Login.jsp`. If you installed ERES with Tomcat, the URL to the login is `http://machinename:port/ERES/Menu_Login.jsp`. This will bring up a page allowing you to login to the menu page directly.

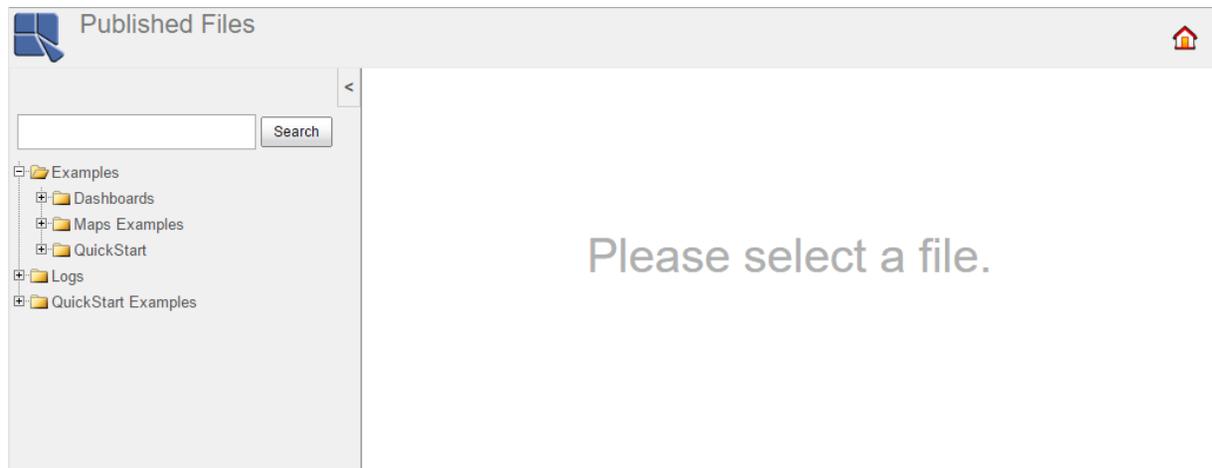
 Home



*Direct Access to Menu Page*

### 7.1.2. Using the Menu Page

When the menu page first loads, it will show a collapsed tree-list of all the reports, charts, maps, dashboards, and other files in the Organizer which you have privileges to see in a tabular format.



*Menu Page*

#### 7.1.2.1. Viewing Reports and Charts

To open a file, expand respective project/folder nodes (in the tree-list on left side) to locate the file and click on the filename. The file will load in the right *DHTML Viewer* panel.

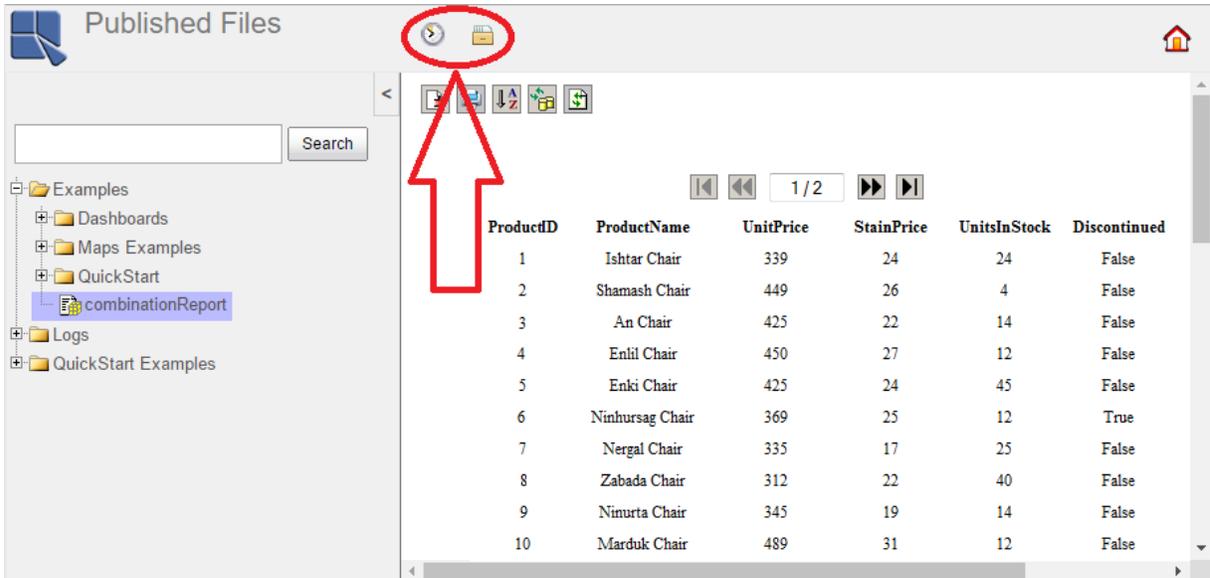
You can also search for specific files. To do so, enter search terms in the upper left dialog and click the *Search* button. The tree-list will reload, showing only relevant files. To cancel the search filter, click the  *Close Search*

icon next to the search field. Note that the *Close Search* icon will not be visible if there is no filter applied to the list at the moment.

You can hide the left panel by clicking on the  *Collapse* icon located in the top-right corner of the navigation panel.

To close the MenuPage and to return to the ERES start page, click the  button in the top-right corner of the MenuPage.

If you open a report or a chart that was scheduled or archived (i.e. the task was run at least one time, it doesn't matter whether the task has already finished or whether it is still active), the following options will be available.

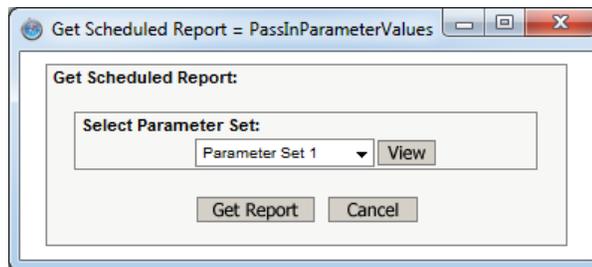


Location of the View Scheduled/Archived version icons



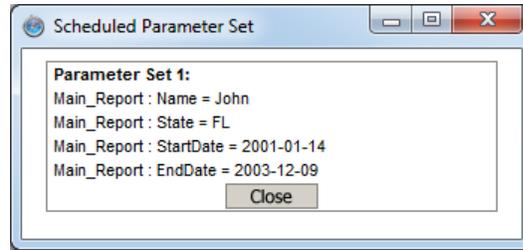
**View Scheduled Report/Chart:**

This option allow you to view the latest scheduled version for the report or chart with file extensions as .pac, .cht, .pak, .rpt, .tpl and .xml. This dialog is only available if an active schedule job is defined for the report or chart. If the report contains parameters, clicking this will bring up a new dialog prompting you to select a parameter set to view for the report.



Select Parameter Set Dialog

Here you select which parameter set you would like to use to see the latest scheduled version. Parameter sets are defined when the schedule job is created. For more about setting schedules see Section 2.2 - Scheduling & Archiving. You can see which values are included in a parameter set by clicking the *View* button. This will open a new window showing the parameter values.



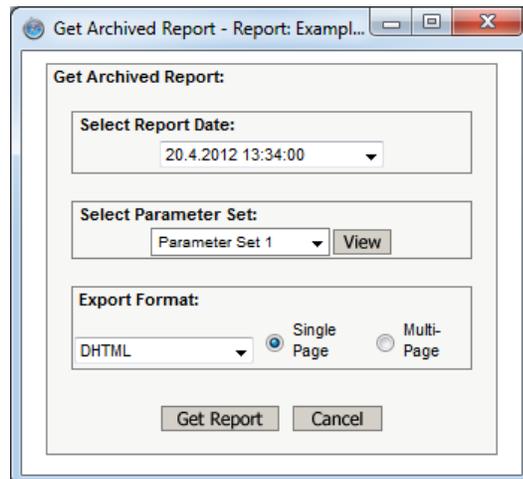
*Parameter Values Dialog*

Once you've selected the parameter set that you would like to use, click *Get Report* (or *Get Chart*) and the scheduled report/chart will open in a new window. If the report or chart contains no parameters, the scheduled version will open directly when you click the *View Scheduled Report/Chart* button.



#### **View Archived Report/Chart:**

This option allows you to view an archived chart or report. This option is only available if an active archive is defined for the report or chart with file extensions as `.pac`, `.cht`, `.pak`, `.rpt`, `.tpl` and `.xml`. Clicking this option will bring up a dialog in a new window allowing you to specify options for the generated report or chart.



*View Archive Dialog*

The first option allow you to select the version of the report or chart you would like to run. The drop-down list contains dates and times for each version. If the report or chart is parameterized, then the second option will allow you to select a parameter set, in the same manner as for scheduled reports and charts. The last option allow you to select the export format. These options are the same as when running a report or chart.

Once you have finished setting options for the report or chart, hit the *Get Report* or *Get Chart* button. The exported report or chart will open in a new window.

#### **7.1.2.1.1. Viewing Reports with Encrypted Data**

If your report contains encrypted data, you need to do two things in order to view the data.

1. You need to create an XML file that gives the database URL, database driver, name of column to be decrypted, and the function to be applied when the data is being retrieved.
2. You need to include the *ReplaceColumnInfoList* option in the command line, i.e. `servercommand.txt` before you start ERES server.

For more details and an example for viewing encrypted data, please see Section 3.2.2.1.2 - Querying Encrypted Data.

## 7.1.2.2. Using the DHTML Report Viewer

If you click on a report in the menu page tree-list, the report will open in the DHTML report viewer. This interface is a thin-client component that allows you to view and interact with the report. In the DHTML report viewer, the first page will show as soon as it is ready, while the rest of the report continues to be processed on the server side. This means that for larger reports, navigating to a page that is not yet exported will result in delay. Before a report is finished processing, the toolbar options will not be available and clicking on any of them will result in a warning message. The main window of the interface provides HTML view of the report with a small toolbar in the top-left corner of the page. Each option on the toolbar will bring up a dialog allowing you to make changes to the report.

The main window also contains a page navigator that contains first page, previous page, next page, last page buttons and a textbox that allows you to jump to any specific page number. The textbox displays the current page number in the form <Current Page Number> / <Total Page Number>. To jump to a specific page, simply type the page number into the textbox and press **Enter**. You can also enter the value in the same fractional format that is used in the display. For example, to jump to page 7 of a 22 page report, you can enter 7 or 7 / 22.



### Q4 Orders by Employee

Employee: Carnody, Frank						
Order ID: 10063						
Order Date: IV 05, 2003						
Quantity	Product Name	Unit Price	Stained	Stain Color	Total	
12	An Chair	\$425.00	No	n/a	\$5,100.00	
8	Enki Chair	\$425.00	No	n/a	\$3,400.00	
2	Anubis Table	\$1,687.00	No	n/a	\$3,374.00	
					<b>Order Total: \$11,874.00</b>	
Order ID: 10065						
Order Date: IV 22, 2003						
Quantity	Product Name	Unit Price	Stained	Stain Color	Total	
18	Shamash Chair	\$449.00	No	n/a	\$8,082.00	
18	Cula Chair	\$468.00	No	n/a	\$8,424.00	
6	Atun Table	\$1,366.00	No	n/a	\$8,196.00	
					<b>Order Total: \$24,702.00</b>	

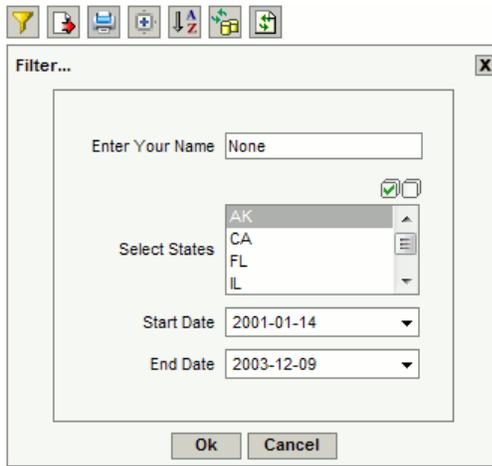
### DHTML Viewer Interface

#### 7.1.2.2.1. The DHTML Report Viewer Toolbar

The options in the DHTML report viewer toolbar are as follows:



**Filter Report:** This option is only available if the report is parameterized. It will bring up a dialog containing all parameters defined in the report.

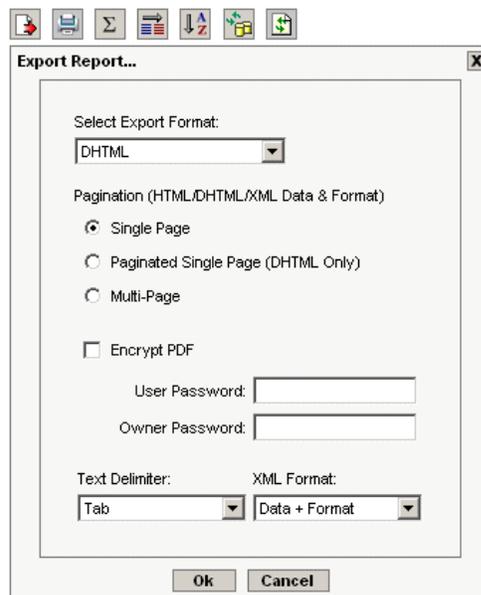


*Filter Dialog*

Select new values and click the *Ok* button to update the report. There is a validation system for unmapped parameters that will check entered values against expected datatype and alert you if it does not match.



**Export Report:** This option allow you to export the report in a variety of file formats. It will bring up a dialog prompting you to select export options.



*Export Dialog*

The first option in the dialog allows you to specify in which format you want to export the report. Available options are HTML, PDF, CSV, Excel (XLS), Excel 2007 (XLSX), DHTML, text, XML and rich text.

Other options allow you to set single or multi-page export for HTML, DHTML, and XML exports. PDF format allows you to enable encryption. For text export, you can select delimiter to be used, and for XML export, you can select whether to export only report data or XML description of the report and data.

For more information about the export formats and options, please see Section 4.1.5.2 - Exporting Reports.

Once you finish specifying options for the exported report, click the *Ok* button. A new window will open containing the exported report. The toolbar will be active on this window, allowing you to save the generated file to your local system or print the report locally.



**Print Current Report:** This option allows you to print current page of the report on user's local printer, as displayed inside the right-side panel.

1. By default, all browsers don't print background colors and images to save ink. You have to change print settings to allow that.

How to set it in different browsers:

Chrome: Check the *Background colors and images* option. It is in the *Print* dialog that opens automatically.

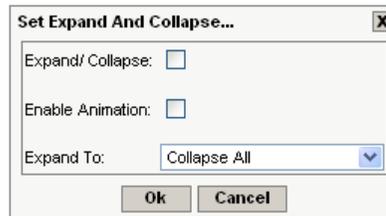
IE 11: Click the gear icon in the upper right corner → *Print* → *Page Setup* → check *Print Background Colors and Images*

Firefox: *Print* → *Page Setup* → check *Print Background (colors & images)*

2. Reports wider than page might be cut off. If you need to print wider reports, please export them to PDF and print the PDF.



**Expand and Collapse View:** This option allows you to add expand and collapse controls to any grouped report.



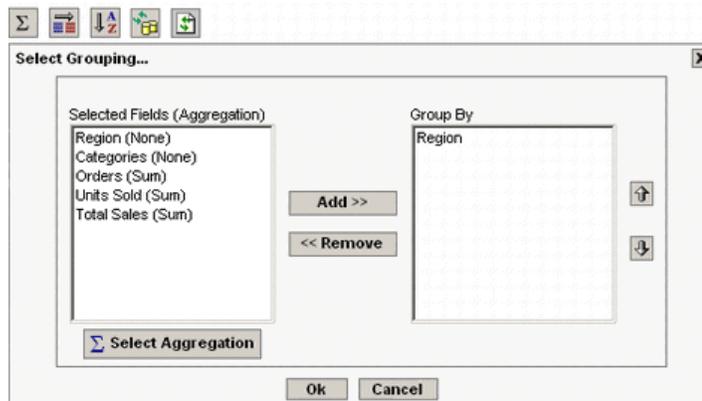
*Expand and Collapse Dialog*

The first option in the dialog allows you to turn the control for expanding and collapsing on. Enable Animation allows the grouping to fade in and out as it is expanded and collapsed. Expand To specifies the initial presentation of the report, i.e. collapsed completely, expanded completely or expanded to a specific group level.

Once you finish specifying the options, click the *Ok* button. The controls will now be added to the report.



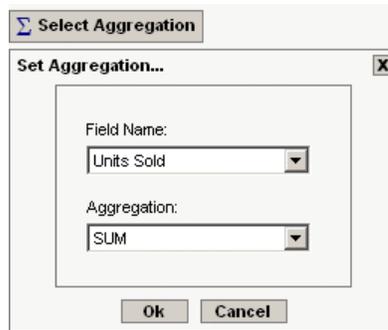
**Change Grouping/Summaries:** This option is only available if the report was created using QuickDesigner. It opens a dialog that allows you to edit the report grouping and aggregation.



*Grouping Dialog*

To add or remove grouping, select a column and click the *Add* or *Remove* button. If you select more than one level of grouping, you can change the nesting order using the arrow buttons on the right side of the dialog.

To set column aggregation, click the *Select Aggregation* button below the column list on the left side. This will open a new dialog allowing you to change the aggregation for the columns in the report.

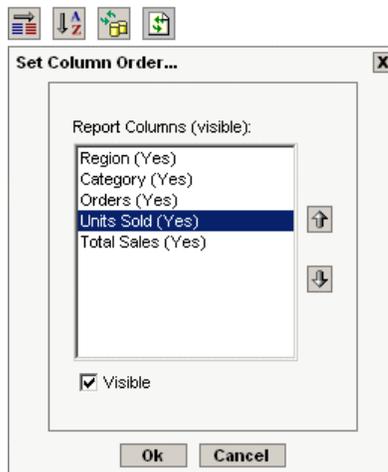


*Aggregation Dialog*

The top drop-down list contains all selected columns for the report and the lower list contains all available aggregation options. To set aggregation for a column, select it in upper list and set the desired aggregation in lower list. Once you finish setting up the aggregation, click the *Ok* button and the dialog will close. The new aggregations will be reflected in the column list.



**Change Column Order/Visibility:** This option is only available if the report was created using Quick-Designer. It opens a dialog that allows you to change the column order in the report as well as show/hide columns.



*Column Order Dialog*

The dialog contains a list of all columns in the report. To change the column order, select a column in the list and click on the *up* or *down arrow* buttons to change the order. The order from top to bottom in the dialog represents the column order from left to right in the report. To show or hide a report column, check or uncheck *Visible* option in the bottom of the dialog.



**Change Sorting Options:** This option allows you to change the sorting in the report. It will bring up a dialog prompting you to set sorting options.



*Sorting Dialog*

The left side of the dialog contains all columns in the report. To sort it by a column, select it in the left side and click the *Add* button. You can control the sort priority by selecting a sort-by column on the right side and clicking on the *up* or *down arrow* buttons to change its position. You can control the sort direction by selecting a sort-by column on the right side and clicking on the *Asc* or *Desc* radio buttons to indicate the direction.



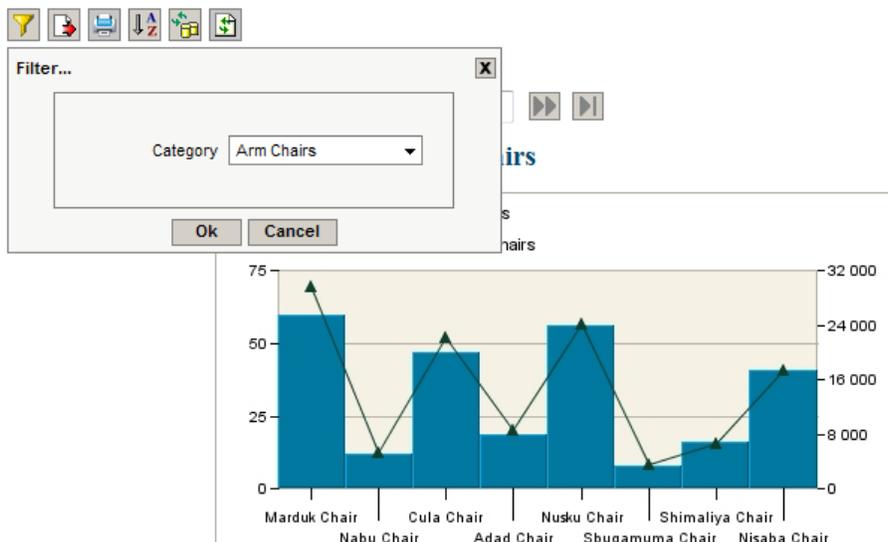
**Refresh Data:** The DHTML Viewer caches the report and therefore when you close and reopen the same report, the data is not updated. This option will refresh the data in the report.



**Reload Report:** When you run a report in the DHTML Viewer, make some changes to this report and then try to open this report again in DHTML Viewer, you'll notice that the report does not reflect the new changes. This is because the DHTML Viewer caches the report to improve performance. To see the changes, use the *Reload Report* button to have the DHTML Viewer read the report structure again and also refresh the data.

### 7.1.2.2.2. Parameterized Reports and Reports with Unmapped Drilldown Parameters

When viewing a parameterized report in the DHTML Viewer, the parameters are initially set to their default values. You can then change the parameter value using the *Filter* icon described above. The DHTML Viewer can also be used to view drill-down reports with unmapped parameters. Similar to a regular parameterized report, the drilldown report will display using default values, but you can adjust them by using the *Filter* icon.



*Drilldown with Unmapped Parameter*

### 7.1.2.3. Using the DHTML Chart Viewer

If you click on a chart in the menu page tree-list, the chart will open in the DHTML chart viewer. This interface is a thin-client component that allows you to view and interact with the chart. The DHTML chart viewer displays the chart using default parameter value (if the chart is parameterized). The parameter value can be changed using the toolbar options. The main window of the interface provides you with HTML view of the chart with a small toolbar in the top-left corner of the page. Each option on the toolbar will bring up a dialog allowing you to modify the chart's look and feel, and navigate through the chart.

#### 7.1.2.3.1. The DHTML Chart Viewer Toolbar

The options in the DHTML chart viewer toolbar are as follows:



**Filter Parameter :** This option is only available when the chart is parameterized. Clicking on the icon will bring up a dialog that allows you to specify parameter values for the chart. You can choose/select the values and then click the *OK* button to update the chart. For unmapped parameter, the value is validated against the expected data type and alerts you in case of a mismatch.



**Export Chart :** This option allows you to export the chart to a variety of formats such as PDF, PNG, GIF, JPEG and SVG. The exported chart is then displayed in a new window from where it can be saved to your local system or printed.



**3D Display Option :** This option is only available when the chart is three dimensional. Three-dimensional charts have several additional options. These options can be accessed by clicking the *3D Display Options* button on the toolbar. This will open a dialog in a new window. Available options include:

- Arrow buttons at the top of the dialog allows you to adjust the viewing angle of the chart. The 3D chart can be rotated horizontally and vertically.
- The first three sliders allows you to control the chart scaling of the chart in the X, Y, and Z directions. The scaling features control the proportion of the chart in each direction. The final slider controls the thickness of the data points.
- The remaining options allow you to toggle wire frame mode for the chart, show/hide the border around the chart, and draw the series inline (instead of on the Z axis) for charts with data series. The last option also enables 3D approximation for the chart. 3D approximation is useful when dealing with charts with a large number of data points. The approximation can improve 3D rendering performance when a large amount of data points exists on the chart. This feature is automatically enabled when the chart contains more than 100 data points.

Once you finish setting the options, click the *Ok* button to save the changes.



**Zoom Option :** The Zoom option dialog is only available for a chart with zoom feature enabled. You can modify the zoom properties such as lower bound, upper bound and zoom scale, as well as change the period layout to linear. Then click the *Ok* button to apply the new zoom changes.



**Refresh Data :** You can use the refresh data option to fetch the latest data for the chart.



**Reload Chart :** Changes in the chart are not always reflected in the DHTML chart viewer. This is because the DHTML chart viewer caches the chart to improve performance. You can click on the *Reload* icon to display the changes and refresh the data.



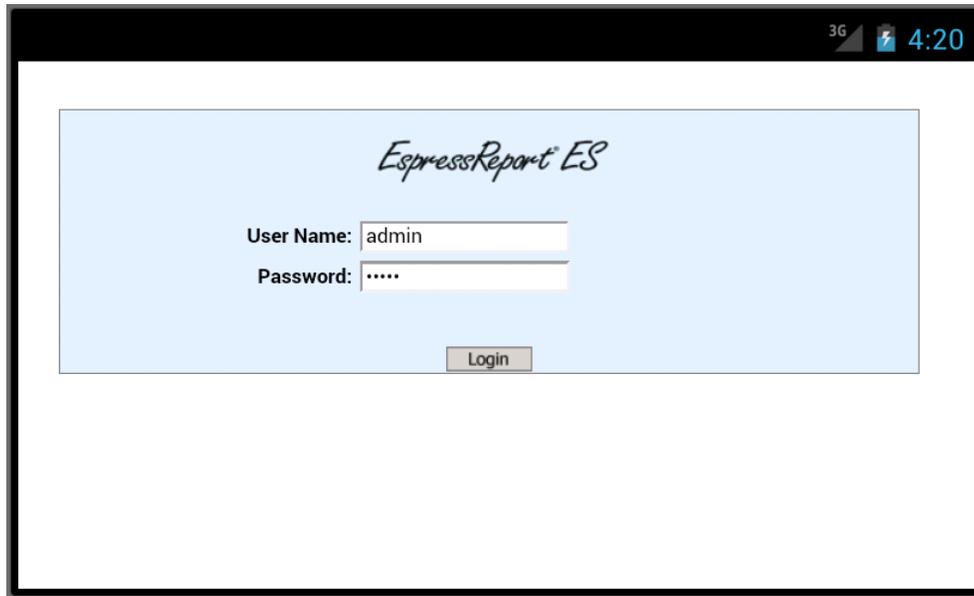
**Back Button :** The *Back* button appears if you are in a sub-level chart. Clicking the *Back* button will take you back to the previous level.

The DHTML chart viewer also supports drill-down. You can drill-down to the next level chart (if the chart is a drill-down chart) by clicking on the desired data point.

The DHTML chart viewer also pops up a hintbox that displays the data information/hyperlink information for the specific data point. If a hyperlink is embedded, clicking on the data point will direct the user to the linked page.

### 7.1.3. Mobile MenuPage

If you access ERES main page from a mobile device, a special simplified ERES start page will be displayed.



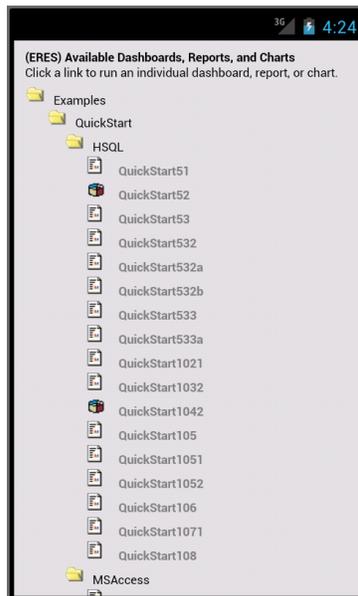
*ERES mobile start page*



**Note**

The mobile features don't work in so-called *PC view*, which can be enabled in some mobile web browsers. In *PC view*, the standard ERES start page and MenuPage will be displayed.

Enter your user name and password and tap on the *Login* button. You will be redirected directly to a special mobile MenuPage which allows you to view reports, charts, maps and dashboards.



*Mobile MenuPage*

The mobile MenuPage is easy to use - scroll/swipe up or down to locate a template or a dashboard and tap on its name to open it.

**Mobile Report Viewer**

If you open a report from the mobile MenuPage, standard DHTML Viewer will open. To learn more about the DHTML Viewer, see Section 7.1.2.2 - Using the DHTML Report Viewer.

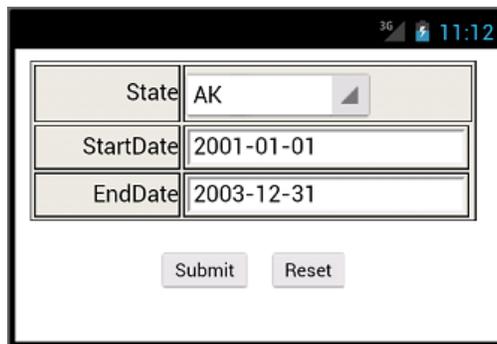


*DHTML Viewer*

**Mobile Chart and Map Viewer**

Chart/map viewer has no control elements.

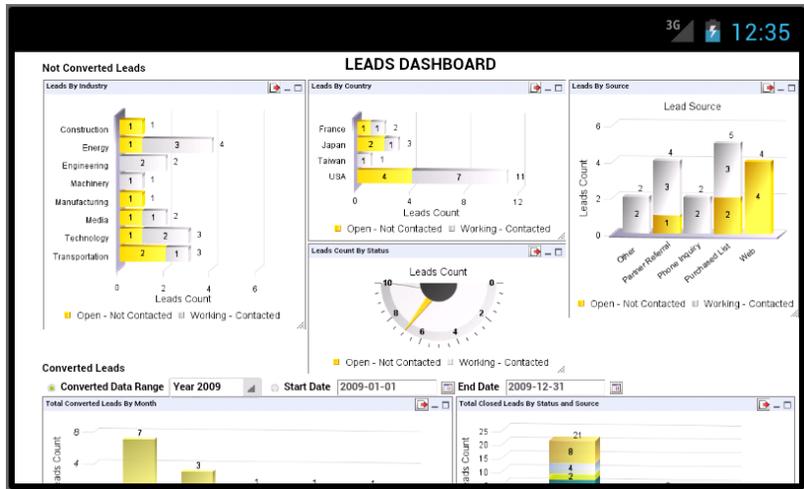
If the chart/map has any parameters, a parameter prompt will be displayed first. Enter any parameter values and tap on the *Submit* button to display the chart/map.



*Parameter prompt*

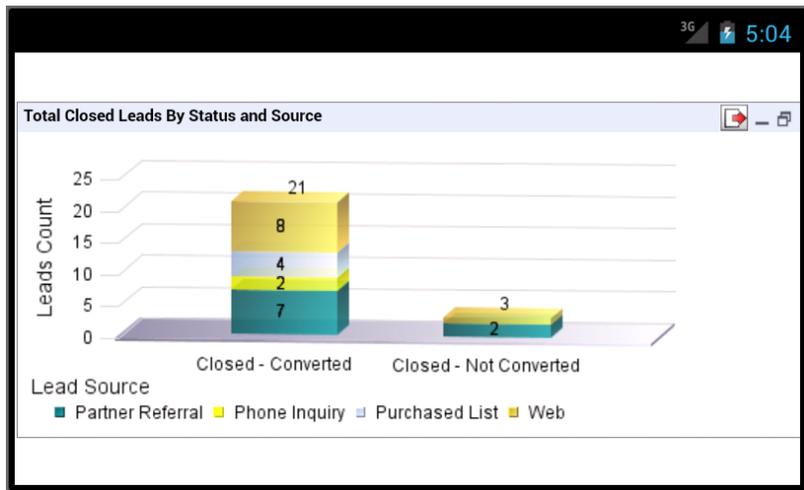
**Dashboards**

If you open a dashboard on a mobile device, you can swipe the dashboard to navigate in it.



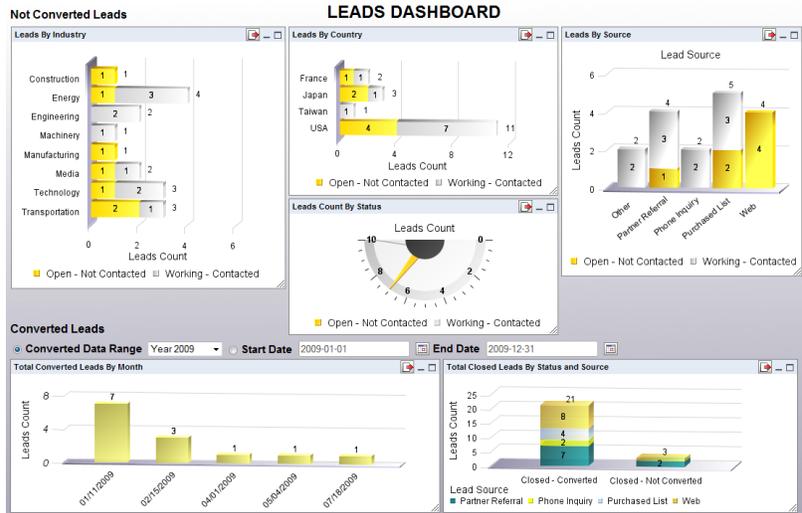
However, on smaller devices, navigating through large dashboards may require extensive zooming and scrolling. In such cases, you can maximize a dashboard template to display the dashboard in a more mobile-friendly way.

To maximize a template, click on its  maximize icon, or double-tap/pinch on the template.

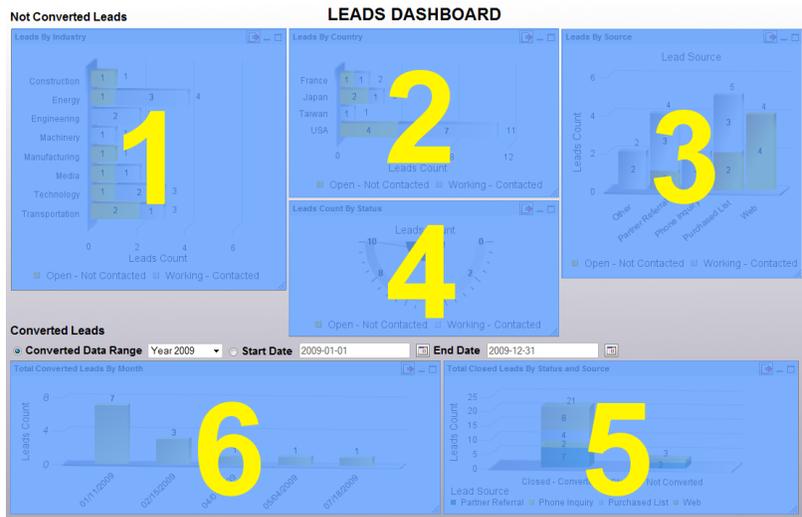


When navigating in maximized-mode, all dashboard templates will be arranged in a single loop, so you don't have to remember the dashboard layout at all. If you swipe on a maximized template to the left, the next template will be displayed.

**For example:** Templates from the following dashboard:



Would be displayed in the following order:

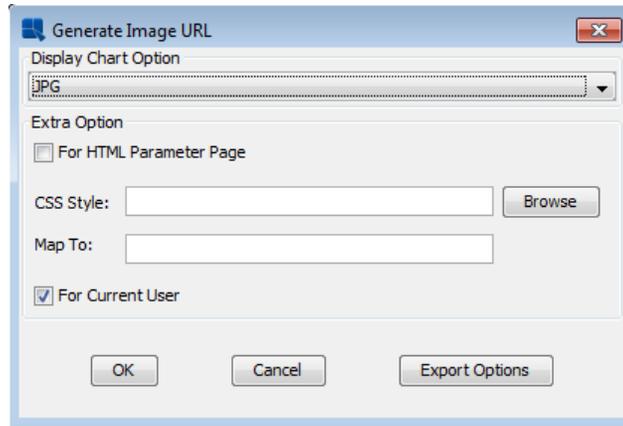


## 7.2. Image URLs

In addition to the menu and the API, EspressReport ES also allows you to deploy charts using image URLs (Reports can also be deployed using URLs as described in Section 7.3 - Report URLs). Image URLs are `http` calls to the server that can either return a chart as a complete `html` page or return an image to be embedded in an existing Web page in an `<img src>` or `<embed>` tag (SVG/PDF). You can use an URL to call a pre-defined chart (`.cht` file) or chart template (`.tpl` file), or to create a chart using default appearance properties. Parameters entered into the URL can control various chart properties.

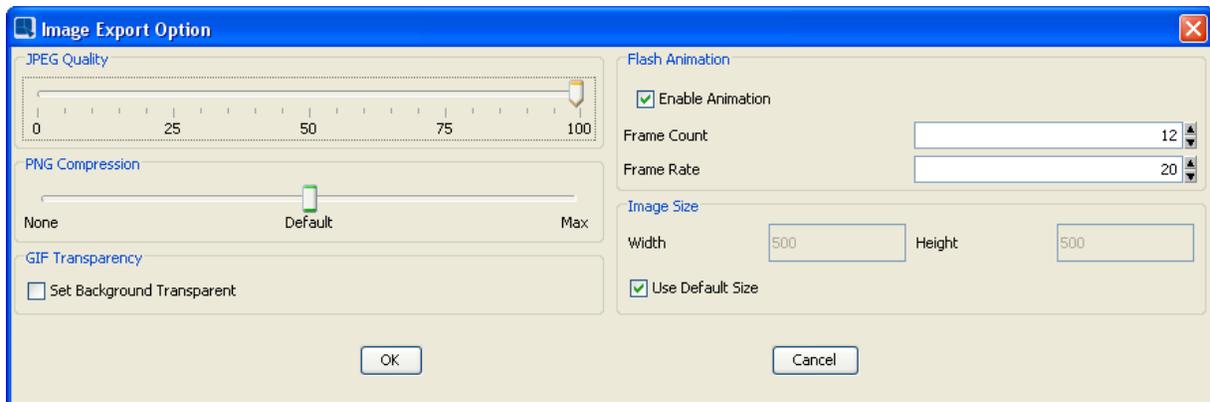
### 7.2.1. Generating URLs in Organizer

The easiest way to generate image URLs for your Web pages is to have the Organizer do it for you. To generate an image URL in the Organizer, first select a chart file (`.cht` or `.tpl`) that you want to use. Then you can either select *Generate Image URL* from the Publish menu, click the *Generate URL* button on the toolbar, or right-click and select *Generate Image URL* option from the pop-up menu. A dialog will appear prompting you to specify attributes for the generated image.



*Image URL Options Dialog*

The first option in the dialog allows you to select the export format for the chart - GIF, JPEG, PNG, PDF, SWF (Flash), or SVG. If the chart contains parameters, you will see options for the HTML Parameter Page (shown above). Unchecking the *For Current User* button will prompt viewers to login first in order to view the chart. You can set additional export options by clicking the *Export Options* button. This will bring up a new dialog with several options for the different chart formats.



*Export Options Dialog*

Options in this dialog allow you to set the JPEG quality. The higher the quality setting, the larger the generated image. You can also set PNG compression and whether to make the exported GIF image transparent. The export method allows you to select either frame or buffer method for generating the chart. Each method can result in faster performance depending on the system, the size of the generated image, and the number of data points in the chart. You can also enable and configure Flash animation if the image depiction should be animated. The last option allows you to specify the image size. You can specify the image size in pixels, or select default size. Clicking the *Use Default Size* button causes the image to be the same size as the chart canvas.

### 7.2.1.1. URLs for Parameterized Charts

If your chart is parameterized, then there are two ways in which the chart can be run. You can either pass parameters directly into the URL as arguments, or have the URL generate a parameter prompt. If the chart contains parameters, then several options will appear in the image URL options dialog as pictured above. To generate an HTML parameter prompt check the *For HTML Parameter Page* option.

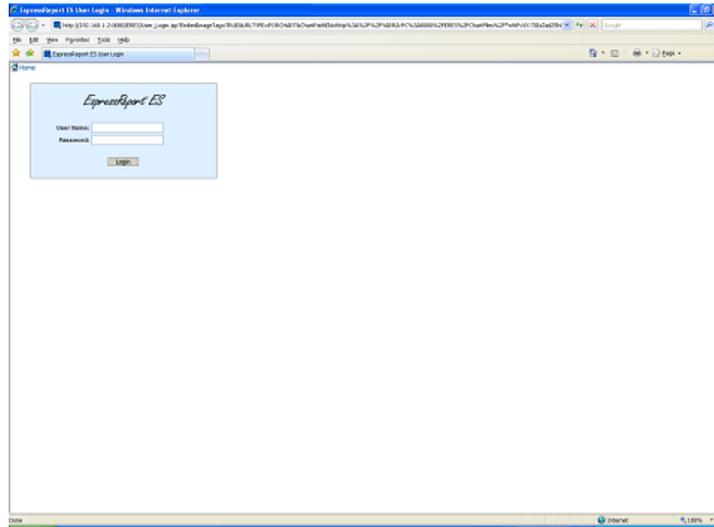
If you select to generate a parameter prompt, you can also specify a CSS file to use to format the prompt. The dialog has options allowing you to specify the file and http path to the CSS file (An example of a parameter page CSS file is available under `<ERESInstallDir>/help/examples/URL`).

If you select not to generate a parameter prompt, then a dialog will open asking you to select parameter values which will be added as arguments to the generated URL.

For information on running the chart URL, see Section 7.2.1.3 - Running Image URLs.

### 7.2.1.2. For Current User

Checking this option will embed the current user in the URL. If you uncheck this option, the user will be re-directed to a login page before viewing the chart.

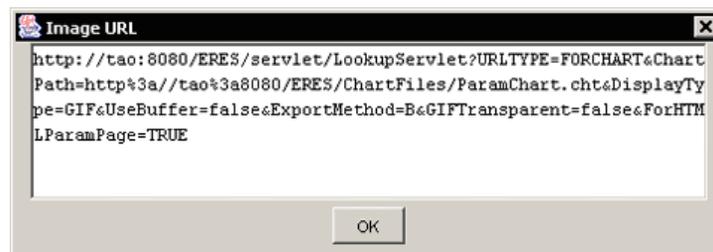


*User Login Dialog*

By default, the URL will redirect to the login page under the ERES context (i.e. `http://machinename:port/ERES`). If you have deployed ERES in a different context, you can use the `JSPCONTEXT` parameter to pass in a different location.

### 7.2.1.3. Running Image URLs

Once you finish specifying options for the image URL, a new dialog will open containing the generated URL string. If you're running Organizer as an applet, the URL will load in a new browser window (allowing you to copy and paste).



*Generated URL*

Image URLs can be run in one of two ways. The image URL can either generate a whole (HTML) page containing the generated chart or it can be used to stream the chart image into an existing page via an image tag (i.e. ``). To generate the chart as a complete page, you need to add the following parameter to the URL: `EmbedImageTag=true`. Additionally, if the URL re-directs to the login page (no username and password supplied) or if you select to generate a parameter prompt for the chart, it will be returned in a complete page.

To embed the chart in an existing page, you must pass an username and password in with the URL or have it retrieve user information from the session (unless you're creating a new chart with the URL) and pass any parameter values into the chart in the URL.

## 7.2.2. Writing Image URLs

Rather than having the Organizer generate the image URL, you can write your own. You can use an existing chart or template file, or you can generate a chart on the fly using default properties. You can control various chart attributes using URL parameters. You can also modify an URL generated by the Organizer.

### 7.2.2.1. URL Syntax

The syntax for image URLs is fairly simple. Every URL begins with a call to the Server: `http://<machinename>:<port>/<context>/LookupServlet?`. The machinename is either the URL, machinename, or ip address of the server machine. The port is the port number for the application server, the default Tomcat for instance uses port 8080. The context is the context for the ERES servlets. By default, the context is `ERES/servlet/`, but this can be changed by the administrator. Following the question mark users need to specify whether the URL will return a chart or report using the `URLTYPE` parameter. Following the `URLTYPE` users can add parameters to the URL. Parameters are separated by the “&” character.

The following structure will generate a simple column chart in a Web page.

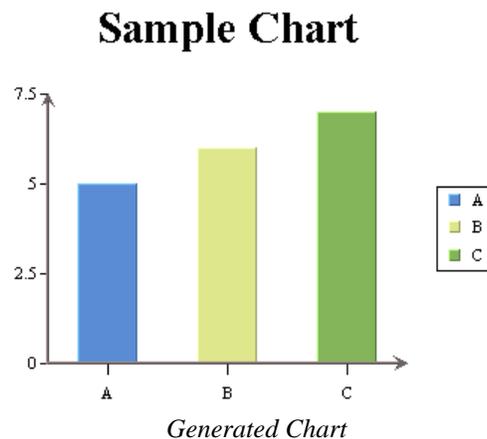
```

```

For example, if the server was set up on `www.quadbase.com` with port 8080 and the context is `ERES/servlet/`, then the image tag should look like this.

```

```



As you can see, this URL generates a column chart with three points on the category (X) axis, and three data points in 1 series on the value (Y) axis.

In order for an URL to run, you must at minimum specify a chart or template file, or specify data, either directly within the URL or using a file or database. Also any input parameter string should use the “+” character in place of a space. An actual space character may disrupt the creation of the image file.

The same URL could be used to generate a Web page with the chart embedded. Simply add the following parameter to the end: `EmbedImageTag=true`. Then the URL can be used as a direct link, and doesn't need to be included in an existing Web page.

### 7.2.2.2. URL Parameters

URL parameters allow you to specify which chart or template file to use, as well as data or data source information, and chart appearance such as chart type and image dimensions.

#### 7.2.2.2.1. Login Parameters

Login parameters allow you to pass username and password into image URLs. A login is required if the URL references a chart or template file (using the `ChartPath` or `TemplatePath` parameters). If login information is not specified in the URL, then the URL will re-direct to a login page before showing the chart.

**USERNAME:** This allows you to specify an username. If you generate an URL in Organizer and select to use an user, the username will be encoded in the generated URL. You can also specify the username as plain text.

```
&USERNAME=user
```

**PASS:** This allows you to specify the user password. If you generate an URL in Organizer and select to use an user, the password will be encoded in the generated URL. The password can also be specified using plain text.

```
&PASS=password
```

**USESESSION:** This allows you to have the server read the username and password from the session. If this option is enabled the information will be retrieved from session parameters `USERNAME` and `PASS`. This option is only available if you will be deploying the URL within a servlet or JSP application.

```
&USESESSION=true
```

**JSPCONTEXT:** This allows you to specify the location (context) of the ERES installation. This location is used when the URL re-directs to the login page. The default value is `/ERES`, and the URL will work without setting this parameter in most deployments. You only need to specify this parameter if ERES is deployed in a context of a different name.

```
&JSPCONTEXT=/MyReports
```

#### 7.2.2.2.2. Data Input Parameters

**TemplatePath:** This allows you to specify the `.tpl` or template file that you would like to use to create the chart. Because the `.tpl` format is saved without data, you can specify different data when using templates. If you do not specify different data ERES will try to reload data from the template's original source.

```
&TemplatePath=http://machinename:port/ChartFiles/  
templatefile.tpl
```

If `USERNAME` and `PASSWORD` is not provided, a login screen will appear prompting for one. If the user provided does not have permission for the template, a chart will not appear.



#### Note

You can use both absolute and relative paths for the template file location.

**ChartPath:** This allows you to specify the `.cht` or chart file that you would like to use to create the chart. Because the `.cht` file is saved with the chart data, you cannot specify different data when using chart files.

```
&ChartPath=http://machinename:port/ChartFiles/  
chartfile.cht
```



### Note

You can use both absolute and relative paths for the chart file location. If either a chart or template file is specified in the URL, then a login is required to access the file.

#### DataFilePath:

This allows you to specify a data file that will be used to plot the chart, either text or XML. For more on data file specifications, please see Section 3.1.5 - Data from Text Files.

```
&DataFilePath=http://machinename:port/chartdata.txt
```



### Note

You can use both absolute and relative paths (relative to ERES installation directory) for the data file location.

From XML files you can use only those, that are in Quadbase format ( please see Section 3.1.4 - Data from XML and XBRL Files).

#### DataSourceClass:

This allows you to specify a class file that will retrieve data to plot the chart. For more on using class files please see Section 3.1.6 - Data from Class Files.

```
&DataSourceClass=package.class
```

#### DBSourceInfo:

This allows you to specify a database and SQL query to be used to retrieve chart data. The parameter takes five input values: database URL, database driver, username, password, and query. Each input value is separated by a semi-colon. For more on connecting to a database, please see Section 3.1.3 - Data from a Database.

```
&DBSourceInfo=url;driver;username;password;query
```

If you do not wish to enter an username or password use the space character “+” instead. Do not drop the input value. The parameter will not run with insufficient inputs. In your SQL statement be sure to substitute the “+” character for any spaces. Also, substitute the word `equal` or `equals` for the “=” sign.

#### QueryParamName:

This parameter is used if the chart or template file you're using contains a parameterized query. `QueryParamName` specifies the name of the query parameter you will be supplying a value for. (Parameter names are assigned when you initialize a query parameter. For more on this, see Section 3.1.3.2.2 - Parameterized Queries) It is always followed by the `QueryParamSize`, and the `QueryParamValue` parameter.



### Caution

You must specify all query parameters (or none of them), else your chart will not be displayed.

---

```
&QueryParamName=customerID&QueryParamSize=1&QueryParamValue=4
```

- QueryParamSize:** This parameter is used to specify the number of values that will be passed into a particular chart parameter. As detailed in Section 3.1.3.2.2.1 - Multi-Value Parameters, some queries can have multi-value parameters. This URL parameter allows you to specify if the parameter (indicated by the previous `QueryParamName` parameter) takes multiple values. This parameter is always followed by one or more `QueryParamValue` arguments for each of the parameter values for a particular parameter. If you do not specify this parameter then it will assume that the number of values to be passed in is one.
- QueryParamValue:** This parameter is used if the chart or template file you're using contains a parameterized query. `QueryParamValue` specifies the value that you would like to pass to the query parameter specified by the `QueryParamName` argument. For multi-value parameters a separate `QueryParamValue` argument must be supplied for each distinct parameter value that you're passing in. For more on creating parameterized charts, please see Section 3.1.3.2.2 - Parameterized Queries.
- ColumnMapping:** If you have specified a data source for the generated chart, this argument allows you to map columns from your data file, or database query result set to the chart. Column mapping is specified using index values. Index values are assigned based on the order they appear in your data file from left to right, or the order in which they are selected in your SQL statement. Index values start with **0**, so the first column is **0** and the second is **1**, etc. To perform column mapping, specify the column index for the respective axes for the chart. If you do not want to map one of the axes use the value **-1**. Do not leave it blank. Mapping varies for different chart types.



### Caution

You must specify this parameter along with `DataFilePath` parameter for proper chart generation.

For Scatter, Surface, and Bubble Charts:

```
&ColumnMapping=series;xvalue;yvalue;zvalue;subValue
```

For High-Low, and HLCO Charts:

```
&ColumnMapping=series;category;high;low;open;close;subValue
```

For Gantt Charts:

```
&ColumnMapping=series;category;end;start;-1;-1;subValue
```

For all other chart types:

```
&ColumnMapping=series;category;sumby;value;subValue
```

For more detailed information about column mapping for all chart types, please see Section 4.2.3.1 - Data Mapping.

---

**TransposeData:** This is a Boolean flag that denotes whether the data is to be transposed or not. It is followed by the `SelectTranspose` parameter which dictates which columns to transpose. By default this is false.

```
&TransposeData=true
```

**SelectTranspose:** This allows you to specify the columns that you would like to transpose. To select columns, specify the column indices separated by a semi-colon. If you do not specify any columns then all of them will be transposed.

```
&SelectTranspose=ColumnIndex;ColumnIndex...
```

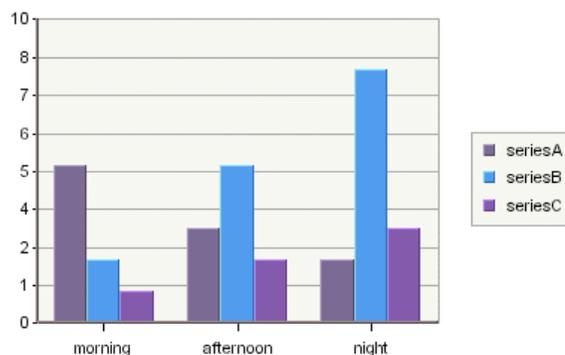
For an explanation of how data transposition works, please see Section 4.2.1.3.2 - Transposing Data.

**ChartData:** This parameter allows you to supply data for the chart within the URL string. The `ChartData` parameter specifies the data points and series for the chart. It is generally used in conjunction with the `CategoryName` parameter which specifies the individual category names for the data points, and the `DataType` parameter which specifies that data type for your chart data. Stack-type charts with a sum by value, can have the sum by names specified using the `SumbyName` parameter.

The structure of the `ChartData` parameter is a series name followed by data points. Each series is separated by the `|` character, and each data point is separated with a semi-colon. For chart types that require more than one value per data point, the sub-values are separated by a comma. The following examples illustrate how the `ChartData` parameter is used for various chart types.

For Column, Bar, Area, Line, Pie, and Dial charts, the structure resembles the following:

```
&ChartData=seriesA;5;3;2|seriesB;2;5;8|seriesC;1;2;3
&CategoryName=morning;afternoon;night&DataType=integer
```

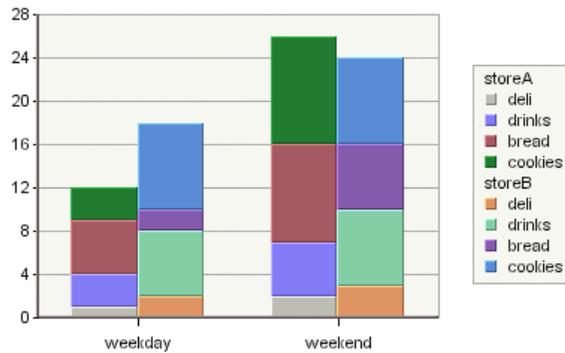


This generates a chart that has three series, `seriesA`, `seriesB`, and `seriesC`. The chart also has three integer data points, `morning`, `afternoon`, and `night` within each of the series.

For Stack Column Stack Bar, Stack Area, and Percent Column charts the `SumbyName` parameter has to be specified in addition to the `CategoryName` and `DataType` parameters. The structure for these charts resembles the following:

```
&ChartData=storeA;1,3,5,3;2,5,9,10|storeB;2,6,2,8;3,7,6,8
```

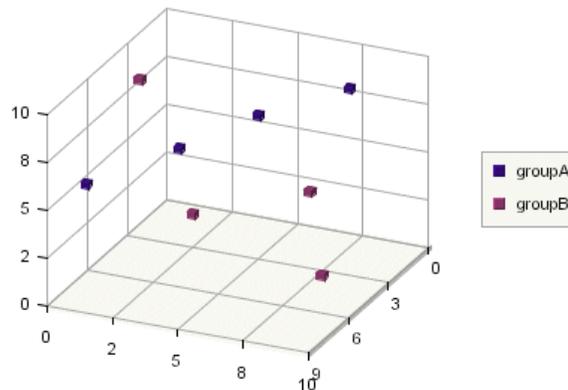
```
&CategoryName=weekday;weekend&SumbyName=deli;drinks;bread;cookies
&DataType=integer
```



This generates a chart with two series, *storeA*, and *storeB*, which have two respective points. The individual data points have four sum by segments separated by a comma.

For Bubble, Scatter and Surface Charts, the *ChartData* parameter value structure is similar to that of the Stack-type charts, however, rather than using the *CategoryName* parameter for the categories, it uses *CategoryLength*, an integer value that specifies the maximum length of the series. The structure resembles the following:

```
&ChartData=groupA;2,5,3;4,6,1;1,6,8;8,9,2|
groupB;1,9,4;7,4,3;9,2,6;2,1,2&CategoryLength=4&DataType=integer
```



For Surface charts, there are some additional restrictions. First, surface charts cannot contain more than one series. Second, each data point must be of the format *x, z, y*, the *x* and *z* values signify the position on the horizontal plane, while the *y* value determines the height. And finally, every unique *x* value must be paired with every unique *z* value, for example the following is an acceptable dataset

```
0,1,5;
0,2,6;
0,3,5;
1,1,5;
1,2,6;
1,3,5
```

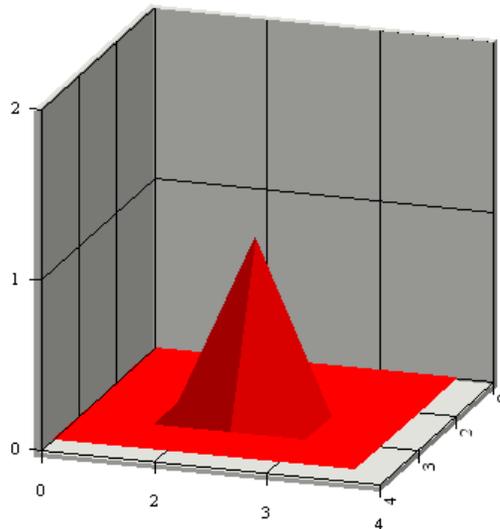
Every unique  $x$  value (0 and 1) were paired with every unique  $z$  value (1, 2 and 3). While the following dataset is not acceptable

```
0,1,2;
0,2,4;
0,3,2;
1,1,2;
1,3,2
```

because when  $x = 1$ , there is no corresponding  $z$  value which equals 2. Also note that there is no restriction on the  $y$  value. It is not necessary to use the `CategoryLength` parameter since there can only be one series, for example:

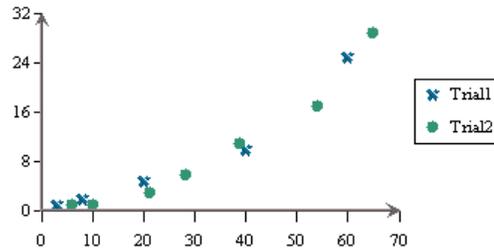
```
&ChartData=A;
0,0,0;0,1,0;0,2,0;0,3,0;0,4,0;
1,0,0;1,1,0;1,2,0;1,3,0;1,4,0;
2,0,0;2,1,0;2,2,1;2,3,0;2,4,0;
3,0,0;3,1,0;3,2,0;3,3,0;3,4,0;
4,0,0;4,1,0;4,2,0;4,3,0;4,4,0
```

Yields the following graph:



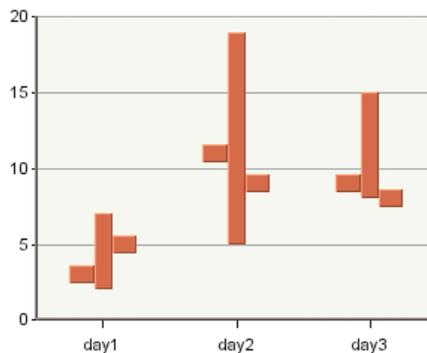
For two-dimensional Scatter Charts, each data point has two sub data points. All other charts have three sub data points separated by a comma. In addition, if the number of data points is not the same for all series, the maximum number of data points in a series must be specified. Use the `CategoryLength` parameter to specify the maximum number of data points, for example:

```
&ChartData=Triall;3,1;8,2;20,5;40,10;60,25|
Trial2;6,1;10,1;21,3;28,6;39,11;54,17;65,29
&CategoryLength=7
```



For HLCO, High-Low, and Gantt Charts, the ChartData parameter specification is similar to that of the Stack-type charts. For HLCO Charts, 4 sub data points within each individual data point specify the High, Low, Open, and Close values. For High-Low Charts, 2 sub data points specify the High, and Low values. For Gantt Charts, 2 sub data points specify the Start and End values. The ChartData structure resembles the following:

```
&ChartData=priceA;7,2,3,5;19,5,11,9;15,8,9,8
&CategoryName=day1;day2;day3&DataType=integer
```



This data generates a HLCO chart with 1 series priceA, and three integer data points day1, day2, and day3. Each data point has a High, Low, Open, and Close value.

For more information about the various chart types, and data mapping, please see Section 4.2.3 - Chart Types and Data Mapping.

**CategoryName:**

This parameter allows you to specify the individual category names of the data points. Each category name is separated by a semi-colon. The data points are specified within the ChartData parameter.

```
&CategoryName=day1;day2;day3...
```

**CategoryLength:**

This parameter allows you to specify the maximum length of the series in scatter, and bubble charts. The argument is specified as a single integer. This needs to be specified when the individual series are not the same length. If CategoryLength is not specified, then the chart will be plotted assuming that the series length is never longer than the first series specified. Hence, if the second series is longer than the first, the additional data points will not be plotted unless the CategoryLength parameter is used.

---

```
&CategoryLength=4
```

**SumbyName:** This parameter allows you to specify the individual sumby names for the data points. This parameter is used when you're entering data for Stack-type charts.

```
&SumbyName=deli;drinks;bread;cookies...
```

**DataType:** This parameter allows you to specify the type of data values that are used by the chart. The default type is `float`. The arguments that can be specified for this parameter are `int`, `integer`, `float`, `double`, `boolean`, `date`, `time`, and `timestamp`. The values are not case sensitive. `Date`, `time`, and `timestamp` types can only be used for Gantt charts.

```
&DataType=integer
```



### Note

`DataType` is only specified for the value axis. You can use different types of data for the category axis, however they are all treated as strings. To use numeric, or date data without having it converted to string, design the chart using a database or data file.

#### 7.2.2.2.3. Appearance Parameters

**MainTitle:** This allows you to specify the main title of the chart. By default, the chart has no title. However, if you're using a template file the titles will carry over, unless you specify different titles using these parameters.

```
&MainTitle=Sample+Chart
```

**XTitle:** This allows you to specify the X-axis title.

```
&XTitle=Time+of+Day
```

**YTitle:** This allows you to specify the Y-axis title.

```
&YTitle=Sales
```

**ZTitle:** This allows you to specify the Z-axis title.

```
&ZTitle=seriesA
```

**Width:** This allows you to specify the width of the generated image in pixels. If you do not specify a width, then the image will have the same width as the chart canvas.

```
&Width=400
```

**Height:** This allows you to specify the height of the generated image in pixels. If you do not specify a height, then the image will have the same height as the chart canvas.

```
&Height=444
```

**FitOnCanvas:** This is a Boolean flag that will cause the chart (not including labels, legend, etc.) to be shrunk and/or repositioned to fit on the canvas. By default, this is `false`.

```
&FitOnCanvas=true
```

**ChartRelativePosition:** This allows you to specify the X and Y position of the lower left-hand corner of the chart to the lower left-hand corner of the chart canvas or image. The two arguments, numbers between 0 and 1 separated by a semi-colon, specify the X and Y positions respectively. `0;0` places the chart at the lower left-hand edge of the image, and `1;1` places the chart at the upper right-hand edge. By default, the chart relative position is `0.2;0.2`.

```
&ChartRelativePosition=0.23;0.18
```

**ChartRelativeSize:** This allows you to specify the size of the chart (relative width and height) relative to the chart canvas or image. The two arguments, numbers between 0 and 1 separated by a semi-colon, specify the width and height respectively. `1;1` causes the chart to be the same size as the image, and `0;0` will have no chart. By default, the chart relative size is `0.6;0.6`.

```
&ChartRelativeSize=0.63;0.7
```

**ChartDimension:** This allows you to specify whether the chart is two-dimensional or three-dimensional. By default the chart is two-dimensional. The argument for this parameter is either 2D or 3D.

```
&ChartDimension=3D
```

**ChartType:** This parameter allows you to select which chart type to use. By default, a Column Chart type is used. The arguments for this parameter can be: `Column`, `Bar`, `Line`, `Pie`, `Area`, `Overlay`, `Box`, `Radar`, `Dial`, `StackColumn`, `StackBar`, `StackArea`, `PercentColumn`, `Scatter`, `Bubble`, `Surface`, `HLCO`, `HiLow`, and `Gantt`.

```
&ChartType=StackArea
```



### Note

For `Overlay`, `Box`, and `Radar` charts, you can only open existing charts through the `ChartPath` parameter, you will not be able to specify `ChartData` for these chart types.

**3DDrawMode:** This allows you to specify the way in which three-dimensional charts should be drawn. There are five options that are specified with an integer argument from 1 to 5. 1 draws the chart as a wire frame without filling in the surfaces. 2 draws a border around the chart frame. 3 draws a border and applies Gouraud shading. 4 draws the chart with Gouraud shading, but doesn't draw a border. 5 draws the 3D chart as standard (without borders or specialized shading). (This is also the default).

```
&3DDrawMode=2
```

For more information about three-dimensional rendering options, see Section 4.2.4.4 - The Navigation Panel.

#### 7.2.2.2.4. Export Parameters

**DisplayType:** This parameter allows you to select the image format in which you would like to render the chart. Options are GIF, JPG, PNG, PDF, and SVG. The default type is JPG.

```
&DisplayType=GIF
```



#### Note

If you're generating a parameter page for the chart (see Section 7.2.2.2.5 - Parameter Page Parameters) you can specify this argument more than once in an URL to give the user a drop-down list from which to select the image format. If you specify more than one format without generating a parameter page, the chart will be generated using the format specified in the first argument.

**ExportOnServer:** This is a Boolean flag that allows you to specify whether the chart should be written as a file on the server-side, or streamed back to the client. If this argument is set to true, then the `ExportPath` argument should also be used to indicate where the file should be written.

```
&ExportOnServer=true&ExportPath=C:\ERES\ExportFiles
\chart.gif
```

**ExportPath:** This allows you to specify a file path to export the chart to. This argument should always be used with the `ExportOnServer` argument. When a file path is specified, the chart will be exported in the selected format to the server-side.



#### Caution

When using `ExportPath` parameter and the file with the same name already exists, it is replaced without any warnings.

**UseBuffer:** This is a Boolean flag that allows you to specify if the image buffer is used.

```
&UseBuffer=true
```

---

**ExportMethod:** This allows you to specify which of the two available image generation methods the server should use. You can specify either `B` for the buffer method or `F` for the frame method. Each method can result in faster performance depending on the system, the size of the generated image, and the number of data points in the chart.

`&ExportMethod=B`

**GIFTransparent:** This is a Boolean flag that allows you to set the background of generated GIF images transparent. This parameter only works if the display type is set to `GIF`.

`&GIFTransparent=true`

**PNGCompression:** This parameter allows you to specify the compression of PNG images. The lower the compression is set, the faster the file will be generated. Lower compressions produce much larger files. There are four compression settings that can be specified using an integer argument. 0 is default compression. This level is the mid-level compression. 1 is fast compression. This generates the PNG file with minimal compression, causing fast file generation with a fairly large file. 2 is maximum compression. This generates a fully compressed PNG file, causing the file to be very small, but taking more generation time. 3 is zero compression. This generates a PNG file without any compression. It is generated very quickly, but has a very large file size.

`&PNGCompression=1`

**JPEGQuality:** This parameter allows you to specify the quality of generated JPEG images. As quality increases, so does the generated file size. The argument for this parameter is an integer between 1 and 99. 1 is minimum quality, and 99 is the maximum. By default the quality is set to 99.

`&JPEGQuality=99`

#### 7.2.2.2.5. Parameter Page Parameters

**ForHTMLParamPage:** This is a Boolean flag that indicates whether or not to return a parameter page instead of a chart. This option only works if the chart contains query parameters. If this argument is set to `true` then the URL will return the parameter page, and ignore any parameter values passed into the URL. By default this argument will be `false`.

`&ForHTMLParamPage=true`

**ParamPageCssStyle:** This allows you to specify a CSS file to format the generated parameter page.

`&ParamPageCssStyle=http://machinename:port/files/  
ParamPage.css`

**Note**

You can use both absolute and relative paths to the CSS file.

<b>ParamPageTitle:</b>	<p>This parameter allows you to specify a title for the parameter page.</p> <pre>&amp;ParamPageTitle=Select+Chart+Options</pre>
<b>ParamPageTitleFontName:</b>	<p>This parameter allows you to specify the font to use for the parameter page title.</p> <pre>&amp;ParamPageTitleFontName=Arial</pre>
<b>ParamPageTitleFontSize:</b>	<p>This parameter allows you to specify the font size for the Parameter page title. The size is specified as HTML size (i.e. <code>&lt;font size=3&gt;</code>) rather than point size. It ranges from 1 to 7, where 1 is the smallest and 7 is the largest.</p> <pre>&amp;ParamPageTitleFontSize=5</pre>
<b>ParamPageTitleFontStyle:</b>	<p>This allows you to apply a font style to the text of the parameter page title. The style is specified as an integer from 0 - 3. The numbers indicate the following styles: 0 - Plain, 1 - Bold, 2 - Italic, 3 - Bold + Italic.</p> <pre>&amp;ParamPageTitleFontStyle=1</pre>
<b>ParamPageTitleFontColor:</b>	<p>This parameter allows you to specify the font color for the parameter page title. The color is specified with a six digit hexadecimal number like in HTML.</p> <pre>&amp;ParamPageTitleFontColor=404040</pre>
<b>ParamPageTextFontName:</b>	<p>This parameter allows you to specify the font to use for the prompt text in the parameter page.</p> <pre>&amp;ParamPageTextFontName=Arial</pre>
<b>ParamPageTextFontSize:</b>	<p>This parameter allows you to specify the font size for the Parameter page prompt text. The size is specified as HTML size (i.e. <code>&lt;font size=3&gt;</code>) rather than point size. It ranges from 1 to 7, where 1 is the smallest and 7 is the largest.</p> <pre>&amp;ParamPageTextFontSize=3</pre>
<b>ParamPageTextFontStyle:</b>	<p>This allows you to apply a font style to the parameter page prompt text. The style is specified as an integer from 0 - 3. The numbers indicate the following styles: 0 - Plain, 1 - Bold, 2 - Italic, 3 - Bold + Italic.</p>

`&ParamPageTextFontStyle=0`

**ParamPageTextFontColor:**

This parameter allows you to specify the font color for the parameter page prompt text. The color is specified with a six digit hexadecimal number like in HTML.

`&ParamPageTextFontColor=000000`

**ParamPageBorderThickness:**

This parameter allows you to specify the thickness of the border drawn around the prompts in the parameter page. The thickness is specified in pixels. Specifying 0 will draw no border around the prompts.

`&ParamPageBorderThickness=1`

**ParamPageBorderColor:**

This parameter allows you to specify the color for the border drawn around the prompts in the parameter page. The color is specified with a six digit hexadecimal number like in HTML.

`&ParamPageBorderColor=C6C6C6`

**ParamPageTableBackgroundColor:**

This parameter allows you to specify the background color for the table containing the prompts in the parameter page. The color is specified with a six digit hexadecimal number like in HTML.

`&ParamPageTableBackgroundColor=FFFFFF`

**ParamPageAlignment:**

This parameter allows you to specify the alignment in the page for the table containing the prompts. Options are LEFT, RIGHT, and CENTER. The default alignment is CENTER.

`&ParamPageAlignment=LEFT`

**ParamPageLayout:**

This parameter allows you to specify the layout of the parameter page. You can either align the parameter prompts vertically (default) or horizontally. The options for this parameter are HORIZONTAL and VERTICAL.

`&ParamPageLayout=HORIZONTAL`

**ParamPageResetEnabled:**

This is a Boolean flag that allows you to show/hide the reset button for the parameter form. By default the button is shown.

`&ParamPageResetEnabled=false`

**ParamPageOuterBorder:**

This is a Boolean flag that allows you to draw only the outer border for the parameter prompt table. This parameter will have no effect if the border thickness is set to zero.

```
&ParamPageOuterBorder=true
```

### 7.2.2.3. Examples

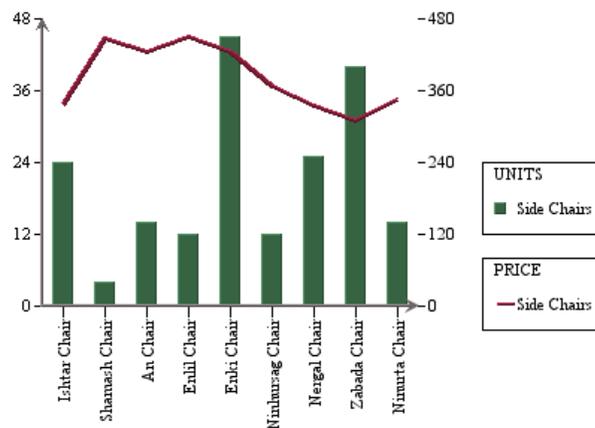
The following examples are sample image URLs that use one or more of the parameters listed in the preceding section. To use these, simply change `machinename` to your host name, `port` to the server port that you are using, and `context` to your servlet context. Note also that these examples all use the `EmbedImageTag` parameter to return a whole Web page containing the chart. If you want to use these examples in an existing Web page, set this parameter to `false`.

These URL examples are also included in a file named `ImageURL.txt` under `help/examples/URL`.

#### Example 1: Query data from a database, and plot a column chart

This example uses the Woodview HSQL database that is included with the ERES installation under `help/examples/DataSources/database`. To run this example you will need to have the HSQL JDBC driver in your classpath.

```
http://machinename:port/context/LookupServlet?URLTYPE=FORCHART
&DBSourceInfo=jdbc:hsqldb:help/examples/DataSources/database/woodview;
org.hsqldb.jdbcDriver;sa;+;select+c.categoryname+as+category,+
p.productname+as+product,+p.unitsinstock+as+units,+p.unitprice+
as+price+from+categories+c,+products+p+where+c.categoryid=equal+
p.categoryid+and+c.categoryid=equal+'SIC'&ColumnMapping=0;1;-1;2;3
&ChartRelativePosition=0.1;0.1&Width=420&Height=320
&EmbedImageTag=true&DisplayType=GIF
```

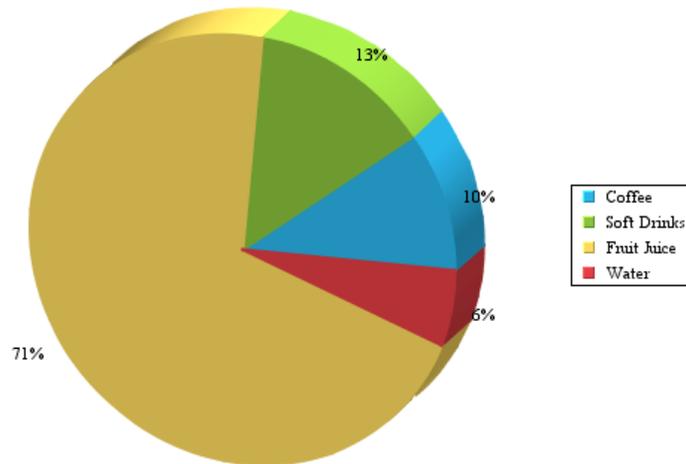


The query retrieves four columns from the database. The `UnitPrice` column is plotted as a sub-value which is the line. Note that the third argument in the `ColumnMapping` parameter is a `-1`. This is because column charts have no Sumbly mapping.

#### Example 2: Draw data from a text file, and plot a 3D pie chart

This example draws data from the `sample.dat` file which is located in the `help/examples/DataSources/text` directory under the server root. You may need to modify the URL for the text file depending on how you have ERES deployed.

```
http://machinename:port/context/LookupServlet?URLTYPE=FORCHART
&DataFilePath=http://machinename:port/ERES/help/examples/DataSources
/text/sample.dat&ColumnMapping=-1;1;-1;4;-1&ChartType=Pie
&ChartDimension=3D&3DDrawMode=2&ChartRelativePosition=0.1;0.2;
&ChartRelativeSize=0.7;0.7&Width=420&Height=320
&EmbedImageTag=true&DisplayType=GIF
```

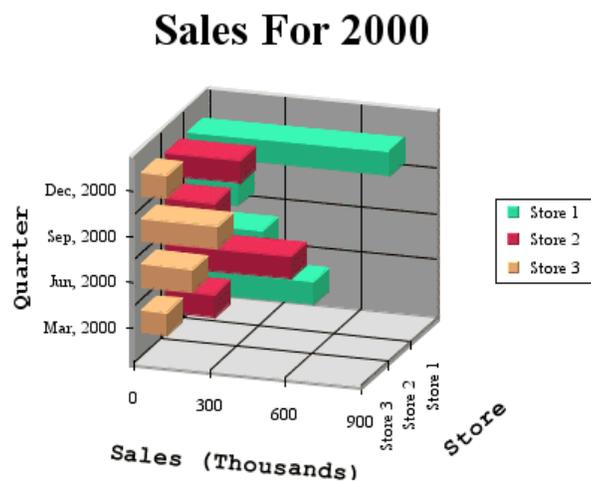


Since pie charts only have category, and value mappings, the column mapping uses -1 for the series, sum by, and sub-value mappings. The 3DdrawMode parameter is used to specify a border around the chart frame.

### Example 3: Create a 3D Bar Chart, and add chart titles

This example uses the ChartData parameter to pass data within the URL string into the chart.

```
http://machinename:port/context/LookupServlet?URLTYPE=FORCHART
&ChartData=Store+1;500;300;200;800|Store+2;200;500;200;300|
Store+3;100;200;300;100&CategoryName=Mar,+2000;Jun,+2000;Sep,+2000;
Dec,+2000&DataType=integer&ChartType=Bar&ChartDimension=3D
&ChartRelativePosition=0.3;0.22&ChartRelativeSize=0.50;0.50&Width=420
&Height=320&MainTitle=Sales+For+2000&YTitle=Quarter
&XTitle=Sales+(Thousands)&ZTitle=Store&EmbedImageTag=true
&DisplayType=PNG
```



The chart is a 3D chart, which by default plots the series on the Z axis. X, Y, and Z axis labels have been added, as well as the chart title.

#### Example 4: Use a template file to apply appearance properties to a new dataset

This example uses the `Example4.tpl` file which is included in the `help/examples/URL` directory of the ERES installation. Because this example includes an existing file, security roles apply. To run this example, you will need to add this file in the Organizer (see Section 2.1.4.1 - Adding and Modifying Files) and set privileges for it (see Section 2.3.2 - Setting User Privileges).

The URL includes the `USERNAME` and `PASS` parameters. You can either supply these parameters, or remove them. If they're removed, you will be re-directed to the login page before viewing the chart.

```
http://machinename:port/context/LookupServlet?URLTYPE=FORCHART
&USERNAME=username&PASS=password&TemplatePath=http://machinename:port
/ERES/help/examples/URL/Example4.tpl&ChartData=Store+A;4;8;16;10|
Store+B;17;9;6;7&DataType=Integer&CategoryName=Computer;Printer;
Monitor;Mouse&ChartType=Area&EmbedImageTag=true&DisplayType=PNG
```



As you can see, the template file has applied all of the appearance properties, including many that cannot be controlled using an Image URL. You main need to modify the URL for the template file depending on how you have ERES deployed.

Image URL parameters can only control some of the chart properties, and most of the properties are displayed using default values. For greater control over the appearance of the chart, you should create a chart or template file using the Chart Designer, and then call or apply that file with an image URL.

## 7.3. Report URLs

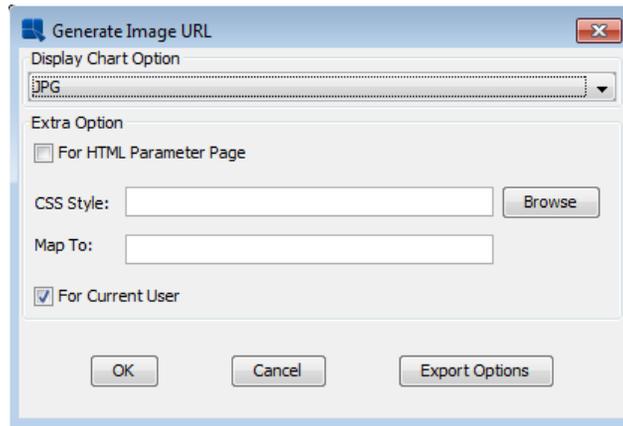
In addition to the menu and the API, EspressReport ES also allows you to deploy reports using URLs. Report URLs, like image URLs, allow reports to be called via an http call to the server. However unlike chart images generated by image URLs, report URLs cannot embed in an `<img src>` or other HTML tag. Report URLs are whole URLs that are run by pointing the web browser to the URL address, or by specifying a hyperlink to the report URL in another Web page.

Like Image URLs, report URLs can be used to generate reports from scratch, or to run existing report templates. However due to the number of available controls in reports, URL parameters are generally related to data input, and exporting.

### 7.3.1. Generating URLs in Organizer

The easiest way to generate report URLs is to have the Organizer do it for you. To generate a report URL in the Organizer, first select the report template (`.pak`, `.rpt` or `.xml`) that you would like to use. Then you can select

*Generate Report URL* from the Publish menu, Click the *Generate URL* button on the toolbar, or right-click and select *Generate Report URL* from the pop-up menu. A dialog will appear prompting you to specify some of the attributes for the generated report.



*Report URL Options Dialog*

The first option allows you to select whether to use the optimized memory exporting. This feature is useful for reports that use a large amount of data. Rather than fetching all of the data from the source at once. This feature will only export 5000 records at a time. This allows reports that use a lot of data to be run without running out of system memory. This feature is only applicable if the report draws its data from a database.

The second option allows you to select the format in which you would like to generate the report. Available formats include: HTML, DHTML, PDF, Excel (XLS), Excel 2007 (XLSX), CSV, text, and rich text.

The third option allows you to select single page or multi page export. This option is only applicable to HTML or DHTML reports. Selecting single page will generate the entire report within a single html file. Selecting multi page will generate a new html file for each page of the report.

### 7.3.1.1. URLs for Parameterized Reports

If your report is parameterized, then there are two ways in which the report can be run. You can either pass parameters directly into the URL as arguments, or have the URL generate a parameter prompt. If the report contains parameters, then several options will appear in the report URL options dialog as pictured above. To generate an HTML parameter prompt check the *For HTML Parameter Page* option.

If you select to generate a parameter prompt, you can also specify a CSS file to use to format the prompt. The dialog has options allowing you to specify the file and http path to the CSS file. (An example of a parameter page CSS file is available under <ERESInstallDir/help/examples/URL>).

If you select not to generate a parameter prompt, then a dialog will open asking you to select parameter values which will be added as arguments to the generated URL.

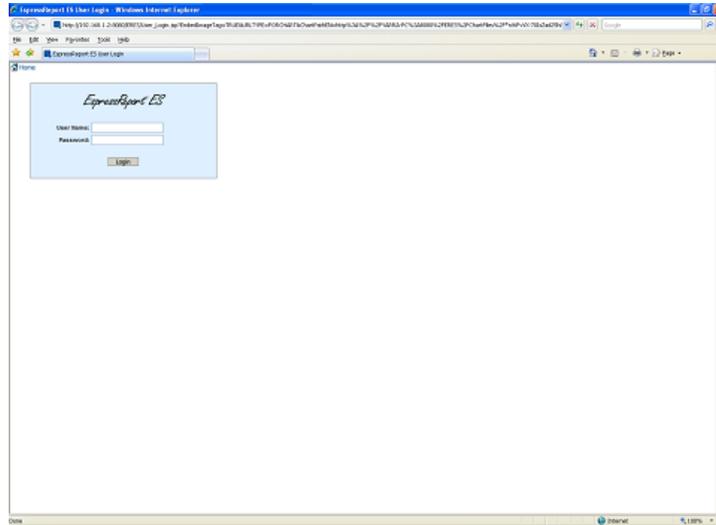
### 7.3.1.2. User Options

Report URLs can have an username and password passed in as arguments in the URL. If the administrator is generating the URL then an option will appear in the report URL options dialog as pictured above that allows you to select the user for which the URL is being generated. Clicking the *Select* button will bring up a list of defined users.

The URL can also be configured to read the username and password from the session of the URL is going to be run in a JSP or servlet environment. If this option is enabled the URL will try to read the username and password from session parameters USERNAME and PASS.

If an user other than the administrator generates the URL, then an option named *For Current User* will appear. Checking this option will embed the current user in the URL.

In either case if an user is selected for the URL the encoded username and password are added as arguments to the URL. If no username or password is supplied in the URL then when it is run, the user will be re-directed to a login page before viewing the report.

*User Login Dialog*

By default, the URL will redirect to the login page under the ERES context (i.e. `http://machinename:port/ERES`). If you have deployed ERES in a different context, you can use the `JSPCONTEXT` parameter to pass in a different location.

### 7.3.1.3. Running Report URLs

Once you have finished specifying options for the report URL a new dialog will open containing the generated URL string. If you're running Organizer as an applet, then the URL will load in a new browser window (allowing you to copy and paste).

*Generated URL*

When run, URLs will generate the report in the specified format and stream it back to the browser. For some formats like PDF or Excel, you will need to have the appropriate plug-in installed on the client in order for the report to display correctly.

### 7.3.1.4. Running Reports with Encrypted Data

To run a report contains encrypted data, you need to do two things in order to view the data.

1. You need to create an XML file that gives the database URL, database driver, name of column to be decrypted, and the function to be applied when the data is being retrieved.
2. You need to include the `ReplaceColumnInfoList` option in the command line, i.e. `servercommand.txt` before you start ERES server.

For more details and an example for viewing encrypted data, please see Section 3.2.2.1.2 - Querying Encrypted Data.

## 7.3.2. Writing Report URLs

Rather than having the Organizer generate the report URL, you can write your own. You can run an existing report template, or you can pass a new data source to a report template, or even create a new report (un-formatted). You can

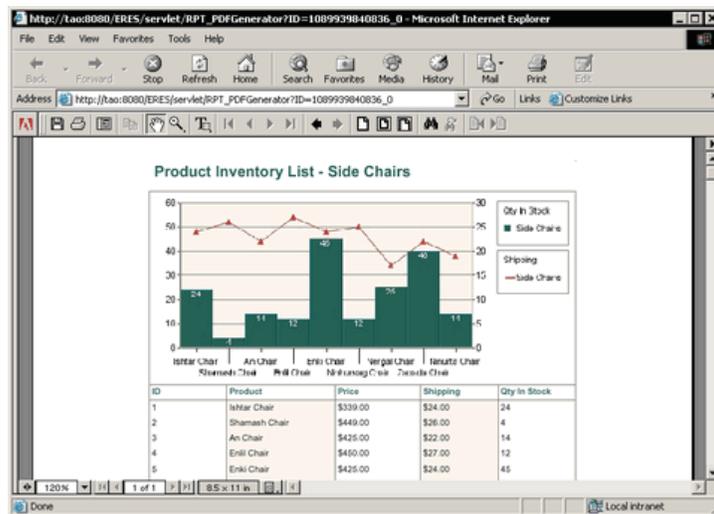
control various report data and output properties using URL parameters. You can also modify an URL generated by the Organizer.

### 7.3.2.1. URL Syntax

The syntax for report URLs is fairly simple. Every URL begins with a call to the Server: `http://machinename:port/context/LookupServlet?`. Following the question mark users need to specify whether the URL will return a chart or report using the `URLTYPE` parameter. Following the `URLTYPE` users can add parameters to the URL. Parameters are separated by the “&” character.

For example the following URL will run the `ProductList.rpt` template in the `help/examples/URL` directory of the ERES installation and return it in PDF format.

```
http://machinename:port/context/LookupServlet?URLTYPE=FORREPORT
&USERNAME=username&PASS=password&TemplatePath=http://machinename:port
/ERES/help/examples/URL/ProductList.rpt&ExportFormat=PDF
```



*Generated Report*

As you can see the URL streams a PDF file back to the browser.

To run this example, you will need to add this file in the Organizer (see Section 2.1.4.1 - Adding and Modifying Files) and set privileges for it (see Section 2.3.2 - Setting User Privileges). The URL includes the `USERNAME` and `PASS` parameters. You can either supply these parameters, or remove them. If they're removed, you will be re-directed to the login page before viewing the report. You may also need to modify the URL for the template file depending on how you have deployed ERES.

In order for an URL to run, you must at minimum specify a report template, or specify data source - either a file or database. Also, any input parameter string should use the “+” character in place of a space. An actual space character may disrupt the creation of the report.

### 7.3.2.2. URL Parameters

URL parameters allow you to specify which report template to use, as well as data source information, and output format appearance such as PDF or HTML.

#### 7.3.2.2.1. Login Parameters

Login parameters allow you to pass username and password into report URLs. A login is required if the URL references a template file (using the `TemplatePath` parameters). If login information is not specified in the URL then the URL will re-direct to a login page before showing the report.

**USERNAME:** This allows you to specify an username. If you generate an URL in Organizer and select to use an user, the username will be encoded in the generated URL. You can also specify the username as plain text.

```
&USERNAME=user
```

**PASS:** This allows you to specify the user password. If you generate an URL in Organizer and select to use an user, the password will be encoded in the generated URL. The password can also be specified using plain text.

```
&PASS=password
```

**USESESSION:** This allows you to have the server read the username and password from the session. If this option is enabled the information will be retrieved from session parameters `USERNAME` and `PASS`. This option is only available if you will be deploying the URL within a servlet or JSP application.

```
&USESESSION=true
```

**JSPCONTEXT:** This allows you to specify the location (context) of the ERES installation. This location is used when the URL re-directs to the login page. The default value is `/ERES`, and the URL will work without setting this parameter in most deployments. You only need to specify this parameter if ERES is deployed in a context of a different name

```
&JSPCONTEXT= "/MyReports"
```

### 7.3.2.2.2. Data Input Parameters

**TemplatePath:** This allows you to specify the report template you would like to run. A report template is saved with its data source information, so if a different data source is not specified, it will retrieve data from the source with which it was designed.

```
&TemplatePath=http://machinename:port/ReportFiles/  
reportfile.rpt
```



#### Note

You can use both absolute and relative paths for the template file location.

**DataFilePath:** This allows you to specify either a text or XML file from which you would like to retrieve data for the report. For more on data file specifications, please see Section 3.1.5 - Data from Text Files.

```
&DataFilePath=http://machinename:port/reportdata.txt
```



#### Note

You can use both absolute and relative paths for the data file location. Also, only XML files in the Quadbase format can be specified (The for-

mat generated when you export XML data from EspressoReport). Other XML formats will not work in Report or Image URLs.

**DataSourceClass:** This allows you to specify a class file that will retrieve data to plot the chart. For more on using class files please see Section 3.1.6 - Data from Class Files.

```
&DataSourceClass=package.class
```

**DBSourceInfo:** This allows you to specify a database and SQL query to be used to retrieve report data. The parameter takes five input values: database URL, database driver, username, password, and query. Each input value is separated by a semi-colon. For more on connecting to a database, please see Section 3.1.3 - Data from a Database.

```
&DBSourceInfo=url;driver;username;password;query
```

If you do not wish to enter an username or password use the space character “+” instead. Do not drop the input value. The parameter will not run with insufficient inputs. In your SQL statement be sure to substitute the “+” character for any spaces. Also, substitute the word `equal` or `equals` for the “=” sign.

**QueryParamName:** This parameter is used if the report you're using contains a parameterized query. `QueryParamName` specifies the name of the query parameter you will be supplying a value for. (Parameter names are specified when you create the query. For more on this, see Section 3.1.3.2.2 - Parameterized Queries) It is always followed by the `QueryParamSize`, and the `QueryParamValue` parameter.

```
&QueryParamName=customerID&QueryParamSize=1&QueryParamValue=4
```

**QueryParamSize:** This parameter is used to specify the number of values that will be passed into a particular report parameter. As detailed in Section 3.1.3.2.2.1 - Multi-Value Parameters, some queries can have multi-value parameters. This URL parameter allows you to specify if the parameter (indicated by the previous `QueryParamName` parameter) takes multiple values. This parameter is always followed by one or more `QueryParamValue` arguments for each of the parameter values for a particular parameter. If you do not specify this parameter then it will assume that the number of values to be passed in is one.

**QueryParamValue:** This parameter is used if the report you're using contains a parameterized query. `QueryParamValue` specifies a value that you would like to pass to the query parameter specified by the `QueryParamName` argument. For multi-value parameters a separate `QueryParamValue` argument must be supplied for each distinct parameter value that you're passing in. For more on creating parameterized queries, please see Section 3.1.3.2.2 - Parameterized Queries.

**FormulaParamName:** This parameter is used if the report you're running contains a formula parameter. `FormulaParamName` specifies the name of the formula parameter for which you will be supplying a value. (Parameter names are specified when you write the formula, for more on this, see Section 4.1.6.2.6 - Formula Parameters). It is always followed by the `FormulaParamValue` parameter.

```
&FormulaParamName=UserName&FormulaParamValue=Jeff
```

**FormulaParamValue:** This parameter is used if the report you're running contains a formula parameter. `FormulaParamValue` specifies a value that you would like to pass to the formula parameter indicated by the preceding `FormulaParamName` argument. For more on formula parameters, see Section 4.1.6.2.6 - Formula Parameters.

**ColumnMapping:** If you have specified a new data source for the report, this argument allows you to map columns from your data file, or database query result set to the report. Column mapping is specified using index values. Index values are assigned based on the order they appear in your data file from left to right, or the order in which they are selected in your SQL statement. Index values start with 0, so the first column is 0 and the second is 1, etc. To perform column mapping, Specify the index values (separated by a semi-colon) from the data source that you would like to include in the report (in the order you want them to appear in the report). You can also provide names for the column in your data source with this parameter as well. This is optional, but setting names allows you to refer to columns by name for the other data mapping parameters. To add column names place them after the column index separated by a comma.

```
&ColumnMapping=3,Category;1,Product;4,Price;5,Sales;10,Employee
```

This example will generate a report that has the fourth, second, fifth, sixth, and eleventh columns from the data source as the first through fifth columns in the report respectively.



### Note

Column order appears from left to right in the generated report.

**RowBreak:** This argument allows you to set row break columns for a summary break or crosstab report. Columns are specified by index number separated by semi-colons. You can also refer to column names that you established in the `ColumnMapping` parameter. For more information about report mapping, please see Section 4.1.2 - Report Types and Data Mapping.

```
&RowBreak=0;1 or &RowBreak=Category;Product
```



### Note

The numbers in this argument refer to the report columns and not the data source columns. Hence the above example sets the first two columns of the report to be row breaks. These columns would be the fourth and second columns from the data source (using the `ColumnMapping` example).

**ColumnBreak:** This argument allows you to set column break columns for a crosstab report. Columns are specified by index number, and separated by semi-colons. You can also refer to column names that you established in the `ColumnMapping` parameter. For more information about report mapping, please see Section 4.1.2 - Report Types and Data Mapping.

```
&ColumnBreak=4 or &ColumnBreak=Employee
```

**Note**

The numbers in this argument refer to the report columns and not the data source columns.

**ColumnBreakValue:**

This argument allows you to set column break value columns for a crosstab report. Columns are specified by index number, and separated by semi-colons. You can also refer to column names that you established in the `ColumnMapping` parameter. For more information about report mapping, please see Section 4.1.2 - Report Types and Data Mapping.

```
&ColumnBreakValue=3 or &ColumnBreakValue=Sales
```

**Note**

The numbers in this argument refer to the report columns and not the data source columns.

**Aggregation:**

This argument allows you to specify column aggregation for non-row break columns in summary break reports, and column break value columns in crosstab reports. Columns are specified by index number followed by a comma and then the aggregation type. You can also refer to column names that you established in the `ColumnMapping` parameter. Available aggregations are `SUM`, `COUNT`, `SUMSQUARE`, `AVG`, `VARIANCE`, `MAX`, `FIRST`, `STDDEV`, `MIN`, `LAST`, `COUNTDISTINCT`.

```
&Aggregation=2,AVG;3,SUM or
&Aggregation=Price,AVG;Sales,SUM
```

By default, non-row break columns in summary break reports have no aggregation, and column break value columns for crosstab reports use `SUM`. For more information about report mapping, please see Section 4.1.2 - Report Types and Data Mapping.

**Note**

The numbers in this argument refer to the report columns and not the data source columns.

**PrimaryKey:**

This argument allows you to set the primary key column for a master & details report. It only accepts one parameter - the column index of the primary key column. You can also refer to column names that you established in the `ColumnMapping` parameter.

```
&PrimaryKey=1 or &PrimaryKey=Product
```

**Note**

The number in this argument refer to the report columns and not the data source columns.

**MasterField:** This argument allows you to specify which columns should be included in the master field for a master & details report. Columns are specified by index number, and separated by semi-colons. You can also refer to column names that you established in the `ColumnMapping` parameter.

```
&MasterField=0;2 or &MasterField=Product;Price
```

The numbers in this argument refer to the report columns and not the data source columns.

**ReportType:** This argument allows you to specify which report type to use for the generated report. Available types are "Columnar", "CrossTab", "MailingLabels", "MasterDetails", and "Summary". Report types are discussed in Section 4.1.2 - Report Types and Data Mapping. The default report type is "Columnar".

```
&ReportType=MasterDetails
```

### 7.3.2.2.3. Export Parameters

**Export Format:** This argument allows you to specify the format in which you would like to display the report. Available formats are DHTML, PDF, TXT, CSV, XLS and XLSX. The default format is DHTML.

```
&ExportFormat=PDF
```



#### Note

If you're generating a parameter page for the report (see Section 7.3.2.2.4 - Parameter Page Parameters) you can specify this argument more than once in an URL to give the user a drop-down list from which to select the export format. If you specify more than one format without generating a parameter page, the report will be generated using the format specified in the first argument.

**ExportOnServer:** This is a Boolean flag that allows you to specify whether the report should be written as a file on the server-side, or streamed back to the client. If this argument is set to `true`, then the `ExportPath` argument should also be used to indicate where the file should be written.

```
&ExportOnServer=true&ExportPath=C:\ERES\ExportFiles
\report.pdf
```

**ExportPath:** This allows you to specify a file path to export the report to. This argument should always be used with the `ExportOnServer` argument. When a file path is specified, the report will be exported in the selected format to the server-side.

**MultiPageExport:** This is a Boolean flag that allows you to specify whether or not to use multi page exporting for HTML and DHTML export formats. By default multi page exporting is not used.

```
&MultiPageExport=false
```

**Note**

This parameter will not work if you use a different data source than the one specified in the report template.

**OptimizeMemory:** This is a Boolean flag that allows you to specify whether or not to use memory optimized exporting when generating the report. This feature is explained in Section 7.3.1 - Generating URLs in Organizer. By default memory optimized exporting is not used.

```
&OptimizeMemory=true
```

### 7.3.2.2.4. Parameter Page Parameters

**ForHTMLParamPage:** This is a Boolean flag that indicates whether or not to return a parameter page instead of a report. This option only works if the Report contains query or formula parameters. If this argument is set to `true` then the URL will return the parameter page, and ignore any parameter values passed into the URL. By default this argument will be `false`.

```
&ForHTMLParamPage=true
```

**ParamPageCssStyle:** This allows you to specify a CSS file to format the generated parameter page.

```
&ParamPageCssStyle=http://machinename:port/files/ParamPage.css
```

**Note**

You can use both absolute and relative paths to the CSS file.

**ParamPageTitle:** This parameter allows you to specify a title for the parameter page.

```
&ParamPageTitle=Select+Report+Options
```

**ParamPageTitleFontName:** This parameter allows you to specify the font to use for the parameter page title.

```
&ParamPageTitleFontName=Arial
```

**ParamPageTitleFontSize:** This parameter allows you to specify the font size for the Parameter page title. The size is specified as HTML size (i.e. `<font size=3>`) rather than point size.

```
&ParamPageTitleFontSize=5
```

---

<b>ParamPageTitleFontStyle:</b>	This allows you to apply a font style to the text of the parameter page title. The style is specified as an integer from 0 - 3. The numbers indicate the following styles: 0 - Plain, 1 - Bold, 2 - Italic, 3 - Bold + Italic.
	<code>&amp;ParamPageTitleFontStyle=1</code>
<b>ParamPageTitleFontColor:</b>	This parameter allows you to specify the font color for the parameter page title. The color is specified with a six digit hexadecimal number like in HTML.
	<code>&amp;ParamPageTitleFontColor=404040</code>
<b>ParamPageTextFontName:</b>	This parameter allows you to specify the font to use for the prompt text in the parameter page.
	<code>&amp;ParamPageTextFontName=Dialog</code>
<b>ParamPageTextFontSize:</b>	This parameter allows you to specify the font size for the Parameter page prompt text. The size is specified as HTML size (i.e. <code>&lt;font size=3&gt;</code> ) rather than point size.
	<code>&amp;ParamPageTextFontSize=3</code>
<b>ParamPageTextFontStyle:</b>	This allows you to apply a font style to the parameter page prompt text. The style is specified as an integer from 0 - 3. The numbers indicate the following styles: 0 - Plain, 1 - Bold, 2 - Italic, 3 - Bold + Italic.
	<code>&amp;ParamPageTextFontStyle=0</code>
<b>ParamPageTextFontColor:</b>	This parameter allows you to specify the font color for the parameter page prompt text. The color is specified with a six digit hexadecimal number like in HTML.
	<code>&amp;ParamPageTextFontColor=000000</code>
<b>ParamPageBorderThickness:</b>	This parameter allows you to specify the thickness of the border drawn around the prompts in the parameter page. The thickness is specified in pixels. Specifying 0 will draw no border around the prompts.
	<code>&amp;ParamPageBorderThickness=1</code>
<b>ParamPageBorderColor:</b>	This parameter allows you to specify the color for the border drawn around the prompts in the parameter page. The color is specified with a six digit hexadecimal number like in HTML.
	<code>&amp;ParamPageBorderColor=C6C6C6</code>

---

---

**ParamPageTableBackgroundColor:** This parameter allows you to specify the background color for the table containing the prompts in the parameter page. The color is specified with a six digit hexadecimal number like in HTML.

```
&ParamPageTableBackgroundColor=FFFFFF
```

**ParamPageAlignment:** This parameter allows you to specify the alignment in the page for the table containing the prompts. Options are `LEFT`, `RIGHT`, and `CENTER`. The default alignment is `CENTER`.

```
&ParamPageAlignment=LEFT
```

**ParamPageLayout:** This parameter allows you to specify the layout of the parameter page. You can either align the parameter prompts vertically (default) or horizontally. The options for this parameter are `HORIZONTAL` and `VERTICAL`.

```
&ParamPageLayout=HORIZONTAL
```

**ParamPageResetEnabled:** This is a Boolean flag that allows you to show/hide the reset button for the parameter form. By default the button is shown.

```
&ParamPageResetEnabled=false
```

**ParamPageOuterBorder:** This is a Boolean flag that allows you to draw only the outer border for the parameter prompt table. This parameter will have no effect if the border thickness is set to zero.

```
&ParamPageOuterBorder=true
```

### 7.3.2.3. Examples

The following examples are sample report URLs that use one or more of the parameters listed in the preceding section. To use these, simply change `machinename` to your host name, `port` to the server port that you are using, and `context` to your servlet context. All of these examples use templates that are in the `help/examples/URL` directory, so you'll need to make sure the templates have been added to the Organizer and that you have permission to view them.

The URLs include the `USERNAME` and `PASS` parameters. You can either supply these parameters, or remove them. If they're removed, you will be re-directed to the login page before viewing the reports.

These URL examples are also included in a file named `ReportURL.txt` under `help/examples/URL`.

#### Example 1: Draw data from a text file

This example draws data from the `Sample.dat` file under the `help/examples/DataSources/text` directory, and creates a simple columnar layout. To run this example, you will need to add `help/examples/URL/Example1.rpt` to the Organizer (see Section 2.1.4.1 - Adding and Modifying Files) and set privileges for it (see Section 2.3.2 - Setting User Privileges).

---

```
http://machinename:port/context/LookupServlet?URLTYPE=FORREPORT
```

```
&USERNAME=username&PASS=password&DataFilePath=http://machinename:port/ERES/help/examples/DataSources/text/sample.dat&ColumnMapping=3;1;4;5;10&TemplatePath=http://machinename:port/ERES/help/examples/URL/Example1.rpt&ExportFormat=DHTML
```

quantity	Drink	volume	high	value
1	Coffee	32.2	30	93
10	Coffee	128.11	34	124
8	Soft Drinks	3.2	33	71
8	Soft Drinks	10.2	34	83
3	Fruit Juice	9.12	34	99
1	Fruit Juice	13.1	34	162
7	Water	23.1	40	94
6	Water	40.1	41	95
20	Coffee	23.6	12	121
4	Coffee	23	10	143
9	Soft Drinks	3.1	24	105
9	Soft Drinks	13.3	44	231
9	Fruit Juice	23.8	35	60
7	Fruit Juice	7.1	30.6	78
15	Water	23.1	34	93

*Generated Report*

This report draws data from fourth, second, fifth, sixth, and eleventh columns of the `sample.dat` file. You may need to modify the URL for the data and template files depending on how you have ERES deployed.

**Example 2: Draw data from a database and generate a summary break report**

This example uses the Woodview HSQL database that is included with the ERES installation under `help/examples/DataSources/database` also names the columns in the data mapping parameters. To run this example you will need to have the HSQL JDBC driver in your classpath. To run this example, you will need to add `help/examples/URL/Example2.rpt` to the Organizer (see Section 2.1.4.1 - Adding and Modifying Files) and set privileges for it (see Section 2.3.2 - Setting User Privileges).

```
http://machinename:port/context/LookupServlet?URLTYPE=FORREPORT
&USERNAME=username&PASS=password&DBSourceInfo=jdbc:hsqldb:help/
examples/DataSources/database/woodview;org.hsqldb.jdbcDriver;sa;+;
select+c.categoryname,+p.productname,+p.unitprice,+p.stainprice,+
p.unitsinstock+from+categories+c,+products+p+where+c.categoryid+
equal+p.categoryid+and+(c.categoryname+equal+'Side+Chairs'+or+
c.categoryname+equal+'Arm+Chairs')&ColumnMapping=0,Category;1,Product;
2,Price;3,StainPrice;4,UnitsInStock&RowBreak=Category
&Aggregation=Product,NONE;Price,AVG;StainPrice,AVG;UnitsInStock,SUM
&ReportType=Summary&TemplatePath=http://machinename:port/ERES/
/help/examples/URL/Example2.rpt&ExportFormat=PDF
```

CATEGORYNAME	PRODUCTNAME	UNITPRICE	STAINPRICE	UNITSINSTOCK
Arm Chairs	Nisabu Chair	414	29	14
	Sibuganuma Chair	445	32	45
	Shimalya Chair	424	36	31
	Nusku Chair	425	31	36
	Ningnuu Chair	478	35	15
	Cula Chair	468	33	4
	Adad Chair	452	35	16
	Marduk Chair	489	31	12
	Nabu Chair	456	31	25
			450.111	32.556
Side Chairs	Zabada Chair	312	22	40
	Ninurta Chair	345	19	14
	Ninhursag Chair	369	25	12
	Nergal Chair	335	17	25
	Enki Chair	450	27	12
	Enki Chair	425	24	45

Generated Report

You may need to modify the URL for the template file depending on how you have ERES deployed.

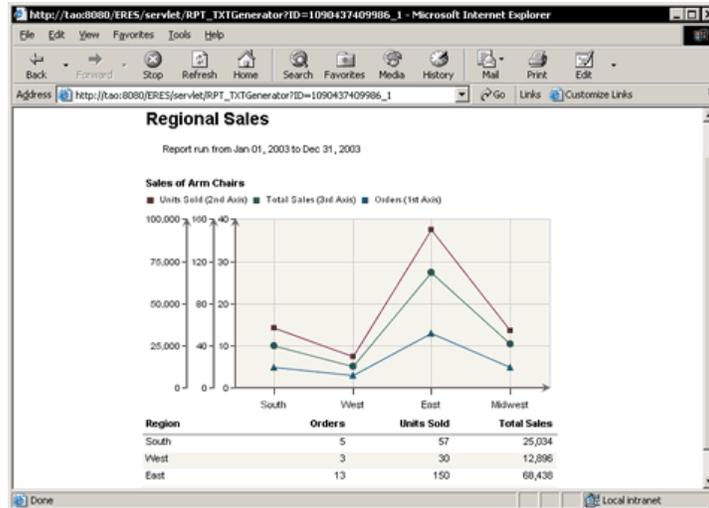
**Example 3: Open a parameterized report and generate an HTML parameter prompt**

To run this example, you will need to add help/examples/URL/RegionalSales.rpt to the Organizer (see Section 2.1.4.1 - Adding and Modifying Files) and set privileges for it (see Section 2.3.2 - Setting User Privileges).

```
http://machinename:port/context/LookupServlet?URLTYPE=FORREPORT
&USERNAME=admin&PASS=admin&TemplatePath=http://machinename:port
/ERES/help/examples/URL/RegionalSales.rpt&ForHTMLParamPage=true
&ParamPageCssStyle=http://machinename:port/ERES/help/examples/URL
/ParamPage.css&ParamPageBorderThickness=1&ParamPageTitle=Please+
Select+Parameters&ParamPageOuterBorder=true
&ParamPageTableBackgroundColor=F7F7EF&ParamPageBorderColor=6C6464
&ExportFormat=DHTML
```

Generated Parameter Prompt

The URL first returns a parameter prompt based on the parameters specified in the URL and formatted with the .css file under help/examples/URL. Clicking *Submit* in this page returns the report with the specified parameters.



Generated Report

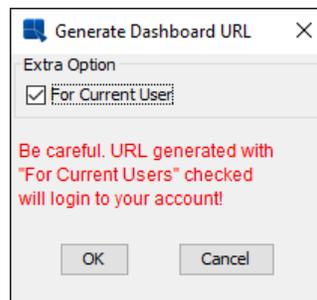
You may need to modify the URL for the template and CSS file depending on how you have ERES deployed.

## 7.4. Dashboard URLs

In addition to the menu and the API, ERES also allows you to deploy dashboards using URLs. Dashboard URLs allow dashboards to be called via http call to the server. Dashboard URLs are whole URLs that are run by pointing the web browser to the URL address, or by specifying a hyperlink to the dashboard URL in another Web page.

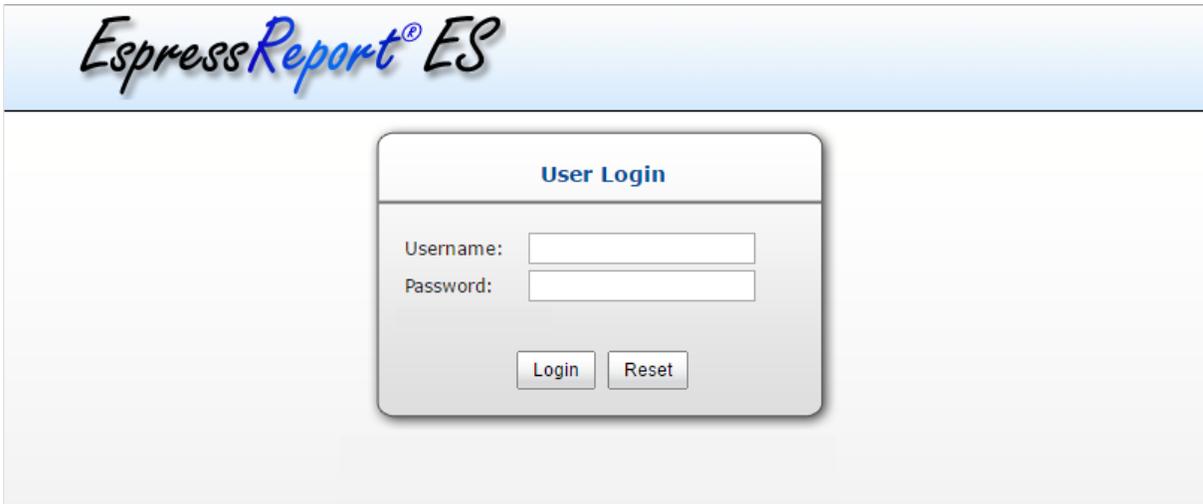
### 7.4.1. Generating URLs in Organizer

The easiest way to generate dashboard URLs is to have the Organizer do it for you. To generate a dashboard URL in the Organizer, first select the dashboard file (.dsb) that you want to use. Then you can select *Generate Dashboard URL* from the Publish menu, click the *Generate Dashboard URL* button on the toolbar, or right-click and select *Generate Dashboard URL* from the pop-up menu. A new dialog will appear allowing you to specify user option for the generated dashboard URL.



Generate Dashboard URL Dialog

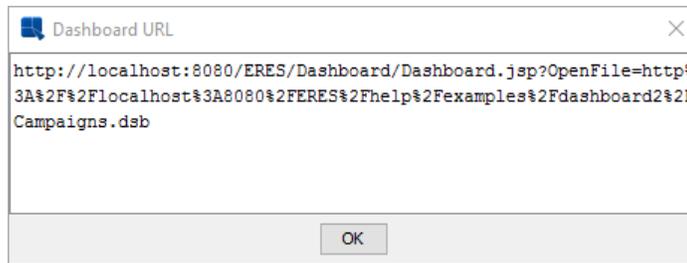
Dashboard URLs can have username and password passed in as arguments in the URL. If the *For Current User* option is checked, the current user will be embedded in the URL. If no username or password is supplied in the URL (*For Current User* option is unchecked) when it is run, the user will be re-directed to a login page before viewing the dashboard.



User Login Page

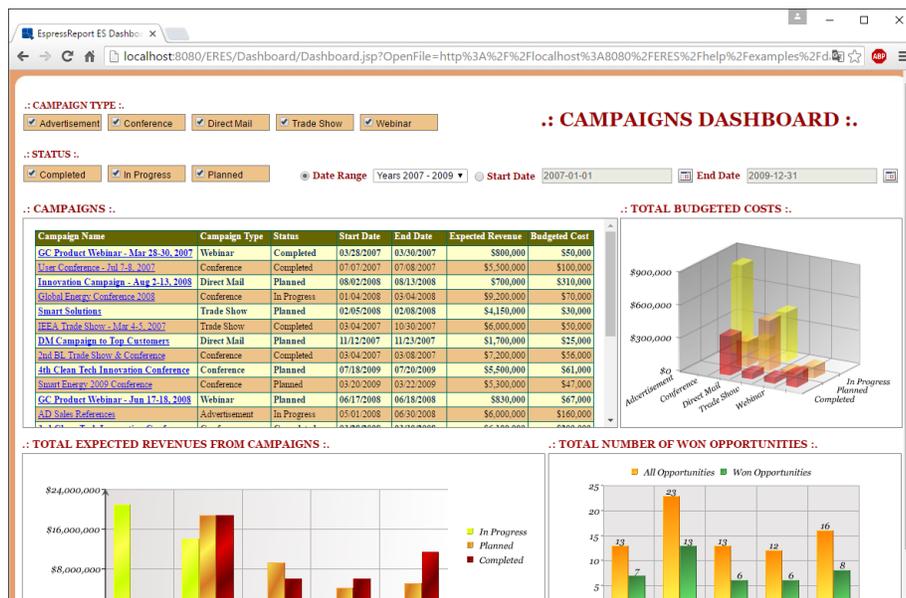
### 7.4.1.1. Running Dashboard URLs

Once you specify the *For Current User* option and click the *OK* button, a new window will open containing the generated URL string. From this window, you can copy the dashboard URL.



Generated Dashboard URL

In order to view a dashboard, just copy and paste the URL into the browser window.



View Dashboard URL

## 7.4.2. Writing Dashboard URLs

Rather than having the Organizer generate the dashboard URL, you can write your own by using the URL syntax below.

### 7.4.2.1. URL Syntax

The syntax for dashboard URLs is fairly simple. Every URL begins with a call to the Server: `http://<MACHINE_NAME>:<PORT>/<CONTEXT_ROOT>/Dashboard/Dashboard.jsp?`. The `MACHINE_NAME` is either the URL, machine name, or ip address of the server machine. The `PORT` is the port number for the application server, the default for Tomcat is 8080. The `CONTEXT_ROOT` is the context root for the ERES. By default, the context root is ERES, but this can be changed by administrator. Following the question mark, users need to specify either the `USERNAME` and `PASS` parameters, or the `USESESSION` parameter instead, as well as the dashboard file location.

The following URL will open the `Population.dsb` dashboard file for the specified user.

```
http://<MACHINE_NAME>:<port>/<CONTEXT_ROOT>/Dashboard/Dashboard.jsp?
USERNAME=qMFq-5dab27h&PASS=qMFq-5dab27h&OpenFile=http://
<MACHINE_NAME>:<port>/<CONTEXT_ROOT>/help/examples/Dashboard/Population.dsb
```

For example, if the server was set up on `www.quadbase.com` with port 8080 and the context root is ERES, then the dashboard URL will look like this.

```
http://www.quadbase.com:8080/ERES/Dashboard/Dashboard.jsp?
USERNAME=qMFq-5dab27h&PASS=qMFq-5dab27h&OpenFile=http://
www.quadbase.com:8080/ERES/help/examples/Dashboard/Population.dsb
```

### 7.4.2.2. URL Parameters

URL parameters allows you to specify the dashboard file you want to run as well as specify whether to use username and password that are stored in the browser session or not.

**OpenFile:** This allows you to specify the dashboard file that you want to run.

```
&OpenFile=http://machinename:port/ERES/DashboardFiles/
TestDashboard.dsb
```



#### Note

You can use both absolute and relative paths for the dashboard file location.

**USERNAME:** This option allows you to specify a username. If you generate an URL in Organizer and select to use an user, the username will be encoded in the generated URL. You can also specify the username as plain text.

```
&USERNAME=user
```

**PASS:** This option allows you to specify a user password. If you generate an URL in Organizer and select to use an user, the password will be encoded in the generated URL. The password can also be specified using plain text.

```
&PASS=password
```

**USESESSION:** This allows you to have the server read the username and password from the session. If this option is enabled, the information will be retrieved from session parameters `USERNAME` and `PASS`. This option is only available if you will be deploying the URL within a servlet or JSP application.

```
&USESESSION=true
```

## 7.5. Menu Page Listener

The menu page and URL publishing options provide a convenient deployment vehicle for reports and charts. Using these interfaces, users can easily deploy charts and reports to the Web without any coding. However, the interfaces themselves do not provide any significant run-time customization capability. Report and chart templates are run in a pretty much “as is” format.

For users that would like to provide some additional logic, or run-time customization to reports and charts, ERES provides the Menu Page Listener interface which is part of the ERES extension package. Using this method, users can implement listeners that will intercept the report or chart prior to export, and use the APIs to modify it.

### 7.5.1. ERES Listener Manager

The ERES Listener Manager is a user-implemented class that is used to manage several listeners that monitor events on the server, including the Menu Page Listener. In order to implement the Menu Page Listener, users must implement the Listener Manager. Below is a sample Listener Manager class:

```
package extensionClasses;

import quadbase.reportorganizer.ext.*;

public class MyEresListenerManager extends DefaultListenerManager {

    public MyEresListenerManager() {}

    public EresSchedulerListener getSchedulerListener() {

        return new MyEresSchedulerListener();

    }

    public MenuPageListener getMenuPageListener() {

        return new MyMenuPageListener();

    }

}
```

Users can implement the `ERESListenerManager` interface or extend `DefaultListenerManager`. The above code implements two listeners, the Scheduler Listener and the Menu Page Listener. The Scheduler Listener takes effect when a schedule job executes. For more information about this interface, see Section 8.7.5.2 - `ICallBackScheduler` Interface

#### 7.5.1.1. Deploying the Listener Manager

You can specify the Listener Manager class as a server option for ERES. You can set this in one of two places. The first option is the *Admin Console*. You can specify the class in the *Server Options* tab. For more information about server configuration options, see Section 1.4.1.3 - Server Options. You can also specify the class by modi-

fyng the QB.properties under <ERESInstallDir>/WEB-INF/classes. However, editing configuration files directly is not recommended and should be done only in case when the ERES server cannot be started because incorrect values have been provided through the Admin Console.

## 7.5.2. Using the Menu Page Listener

The following code shows a sample implementation of the Menu Page Listener:

```
package extensionClasses;

import java.lang.*;
import java.awt.*;
import quadbase.reportorganizer.ext.*;
import quadbase.reportdesigner.ReportAPI.*;
import quadbase.reportdesigner.ReportElements.*;
import quadbase.reportdesigner.util.*;
import quadbase.ChartAPI.*;
import quadbase.reportdesigner.report.Formula;

public class MyMenuPageListener implements MenuPageListener {

    public QbReport modifyBeforeRun(QbReport report, String username) {

        System.out.println("Calling Menu Page Listener...");

        try {

            // Create a new label to show user that is running the
report
            ReportCell userLabel = new ReportCell();
            userLabel.setText("Report Run By: " + username);
            userLabel.setFont(new Font("Arial", Font.PLAIN, 8));
            userLabel.setAlign(IAAlignConstants.ALIGN_RIGHT);
            userLabel.setHeight(0.2);
            userLabel.setWidth(1.6);
            userLabel.setY(0);
            userLabel.setX(5.90);

            // Create a new formula to show the report run date/time
            ReportCell runDate = new ReportCell();
            Formula runDateFormula = new
Formula("ReportRunDate", "\"Report Run Date: \" +
printDateTime(getCurrentDateTime(), \"MMM dd, yyyy h:mm a\")");
            report.addFormula(runDateFormula);
            runDate.setFormulaObj(runDateFormula);
            runDate.setFont(new Font("Arial", Font.PLAIN, 8));
            runDate.setAlign(IAAlignConstants.ALIGN_RIGHT);
            runDate.setHeight(0.2);
            runDate.setWidth(2.5);
            runDate.setY(0.2);
            runDate.setX(5.0);

            // Add new cells to page header section
            report.getPageHeader().addData(userLabel);
            report.getPageHeader().addData(runDate);

            // Adjust section height if necessary
            if (report.getPageHeader().getHeight() < 0.4)
```

```

report.getPageHeader().setHeight(0.4);

    } catch (Exception ex) {

        ex.printStackTrace();

    }

    return report;

}

public QbChart modifyBeforeRun(QbChart chart, String username) {

    System.out.println("Calling Menu Page Listener");

    try {

        // Define an array of chart colors
        Color gold = new Color(227,215,130);
        Color salmon = new Color(199,85,90);
        Color burgandy = new Color(125,18,66);
        Color slate = new Color(81,119,156);
        Color teal = new Color(130,203,217);
        Color blue = new Color(26,42,103);
        Color beige = new Color(188,204,177);
        Color[] chartColors = {gold, salmon, burgandy, slate,
teal, blue, beige};

        // Apply new colors to the chart
        chart.getDataPoints().setColors(chartColors);

    } catch (Exception ex) {

        ex.printStackTrace();

    }

    return chart;

}

}

```

This example implements to modifyBeforeRun methods, one for charts and one for reports. With this code deployed, everytime a report is run, the username and the run time is added to the report header. Everytime a chart is run, the chart colors are modified to use the palette defined in the method. Note that this is only one example. Any report/chart API code can be used in these interfaces to modify the reports and charts prior to export. For more information about the Report API, see Section 8.1 - ERES Report API.

## 7.6. Report Viewer

The Report Viewer is an applet that allows you to view a report dynamically through a Web browser. The Viewer can read report template files (.rpt, and .xml). The Report Viewer can be automatically embedded in an HTML page or manually placed in a Web page.

Inside Report Viewer, you go to different sections and pages of the report. There are built-in callback mechanisms that would allow users to click on a data element to jump to a related URL. Report Viewer also supports scheduled refresh (whereby a report's data is updated at regular intervals specified by the designer).

To create a stand-alone HTML page with the Viewer embedded, check the *Create HTML* option when saving a report in the Report Designer.

In addition you can embed the Viewer with your own Web page using the following syntax:

```
<applet codebase= ".." code =
  "quadbase.reportdesigner.ReportViewer.Viewer.class" width= 100% height=100%
  archive="ReportViewerWithChart.jar">
<PARAM name="filename" value="yourReport.pak">
</applet>
```

The parameter value of `filename` specifies the name of the file that contains the report. The filename value can also be specified by a URL; for example, you can prefix it by `http://` for accessing a remote data file. When viewed in the Report Viewer, the `.pak` file will connect to the data source specified, dynamically fetch the data, and generate the report.

## 7.6.1. The Report Viewer Parameters

Without using the Report Designer, you can also pass data via parameters to the Report Viewer applet. That is, you can use the Report Viewer to directly view a data file, or pass the data directly in the form of lines of data, in the HTML code. The following is a list of parameters:

### Report Parameters

<b>comm_protocol:</b>	The protocol to be used, to connect to the ERES Server
<b>comm_url:</b>	The URL to connect to the ERES Server
<b>servlet_context:</b>	The context in which the ERES Server is running
<b>RefreshInterval:</b>	Specifies the interval in seconds to refresh the data (i.e., getting the data from the data source)
<b>FileName:</b>	Specifies the name of the <code>.pak</code> file
<b>ReportData:</b>	Specifies the report as a string
<b>SourceDB:</b>	Specifies the database information
<b>SourceFile:</b>	Specifies the data file information
<b>SourceData:</b>	Specifies the data information
<b>ReportType:</b>	Specifies the report type
<b>Aggregation:</b>	Specifies the Aggregation Method
<b>ColInfo:</b>	Sets the column mapping
<b>RowBreak:</b>	Specifies the column(s) to be used as Row Breaks
<b>ColumnBreak:</b>	Specifies the column to be used as Column Break
<b>ColumnBreakValue:</b>	Specifies the column to be used as the values for the Column Break columns
<b>PrimaryKey:</b>	Specifies the column to be used as Primary Key
<b>MasterKey:</b>	Specifies the column(s) to be used as Master Key
<b>RefreshData:</b>	If false, report cannot be refreshed. The default is <code>true</code>

**ExportEnabled:** If false, report cannot be exported. The default is `true`

Example: With the parameter `RefreshInterval` you can insert the following line:

```
<PARAM name="RefreshInterval" value="60">
```

In the above example, Report Viewer applet will fetch data from the specified data source (with the help of the ERES Server) and redraw the report every 60 seconds - all transparently. It is useful for accessing databases, which are updated frequently. Note that when specifying parameters that can have multiple values (or needs to be have multiple values), the separator is a space (" " without the double quotes). For example:

```
<PARAM name="RowBreak" value="0 1 2">
```

In the above example, the Row Break is set on Column 0, Column 1 and Column 2 of the Column Mapping (i.e., the first three columns). The values for the parameter have a space (" " without the double quotes) separating them.

## 7.6.2. Specifying Data for Report Viewer

Using parameters, you can specify a data source for the Report Viewer in order to display different data with a report template.

### 7.6.2.1. Data From a Database

The following is a piece of HTML code that uses the Report Viewer to view a report using data extracted from a database.

```
<applet code="quadbase.reportdesigner.ReportViewer.Viewer.class" width=640
height=480>

    <PARAM name="sourceDB" value="jdbc:odbc:DataSource,
sun.jdbc.odbc.JdbcOdbcDriver, username, password, select * from products">
    <PARAM name="ColInfo" value="0 1 3">
    <PARAM name="ReportType" value="SummaryBreak">
    <PARAM name="RowBreak" value="0">
    <PARAM name="Aggregation" value="SUM">

</applet>
```

The arguments of **ColInfo** specify the mapping for the report.

The argument **ReportType** specifies the type of report to be displayed and it can be one of

- SimpleColumnar
- SummaryBreak
- CrossTab
- MasterDetails

The argument **RowBreak** specifies the column on which the row break is applied.

The argument **Aggregation** specifies the aggregation method applied and it can be one of:

- NONE

- SUM
- MAX
- MIN
- COUNT
- AVG
- FIRST
- LAST
- SUMSQUARE
- VARIANCE
- STDDEV
- COUNTDISTINCT

### 7.6.2.2. Data From a File

This HTML code generates a report using data from a data file.

```
<applet code = "quadbase.reportdesigner.ReportViewer.Viewer.class" width=640
height=480>

    <PARAM name="sourceFile" value="http://.../test.dat">
    <PARAM name="ColInfo" value="1 0 2 3">
    <PARAM name="ReportType" value="SimpleColumnar">

</applet>
```

### 7.6.2.3. Data From an Argument

It is possible to have the Report Viewer read in data directly from the HTML file itself, rather than from a data file or database.

```
<applet code = "quadbase.reportdesigner.ReportViewer.Viewer.class" width=640
height=480>

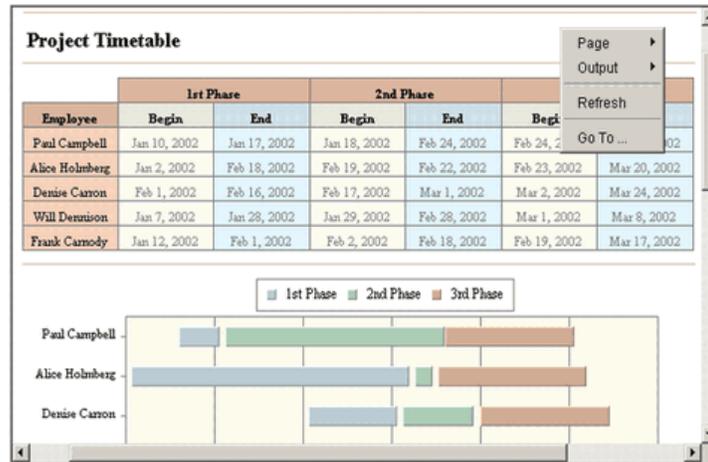
    <PARAM name="sourceData" value="int, string, int | value, name, vol |
10, 'John', 20 | 3, 'Mary', 30 | 8, 'Kevin', 3 | 9, 'James', 22">
    <PARAM name="ColInfo" value="1 0 2">
    <PARAM name="ReportType" value="SimpleColumnar">

</applet>
```

The format for the parameter `SourceData` is the same as the data file format, except that each line is ended by a vertical bar “|”.

## 7.6.3. Using Report Viewer

Using the Report Viewer, you can browse through the report using the Pop-Up Menu. Right-clicking on the report displays a Pop-Up Menu which can be used to traverse the report.



*Report Viewer Window with Pop-Up Menu*

The Pop-Up Menu can show the following options:

- **Back:** Go back to the previous report. This option is shown when a hyperlink is clicked on.
- **Section:** Contains various options to navigate the various sections
  - First Section:** Go to the first section
  - Previous Section:** Go to the previous section
  - Next Section:** Go to the next section
  - Last Section:** Go to the last section
- **Page:** Contains various options to navigate the various pages
  - First Page:** Go to the first page
  - Previous Page:** Go to the previous page
  - Next Page:** Go to the next page
  - Last Page:** Go to the last page
- **Output:** Contains various formats to output the report to
  - Generate DHTML (Single Page):** Export the report to DHTML onto a single file
  - Generate DHTML (Multiple Pages):** Export the report to DHTML onto multiple files, one file per page.
  - Generate PDF:** Export the report to PDF
  - Generate CSV:** Export the report to CSV
  - Generate EXCEL (XLS):** Export the report to Excel (xls file)
  - Generate EXCEL 2007 (XLSX):** Export the report to Excel 2007 (xlsx file)
  - Generate TXT:** Export the report to TXT
  - Generate RTE:** Export the report to RTE
  - Generate XML (Data + Format):** Export the complete report to XML

---

**Generate XML (Pure Data):** Export the report data only to XML

**Print:** Print the report

- **Refresh:** Refresh the data from the data source
- **Go To:** Go to the specified page and section
- **Sort by (ascend):** Sort the report based on a selected column's ascending values
- **Sort by (descend):** Sort the report based on a selected column's descending values
- **Show/Hide Report Toolbar:** This either shows or hides the navigation toolbar at the bottom of the viewer window.

In addition to the Pop-Up Menu, you can also

- Left single click to jump to the hyperlink associated with the cell. (Note that this creates a new browser window if the associated URL is **NOT** a .pak file. Users cannot use the *Back* button on the browser to return to the previous report)
- Run the mouse over a cell containing a hyperlink to see the Hint text associated with the link.

## 7.6.4. Connecting to the ERES Server

In order for the Report Viewer applet to connect to the ERES Server (to fetch the report data, and save exported files) you will need to specify the following parameters in your HTML code:

```
<PARAM name="comm_protocol" value="servlet">
<PARAM name="comm_url" value="http://machine:port">
<PARAM name="servlet_context" value="ERES/servlet">
```

Adding the above params to your HTML code will make Viewer connect to the server at `http://machine:port/ERES/servlet`.

## 7.6.5. Swing Version

A JFC/Swing version of the Report Viewer can also be used by referring to `SwingReportViewer.jar` instead of `ReportViewer.jar` in the `lib` directory under the ERES install directory. Call the following class when using the Swing viewer: `quadbase.reportdesigner.ReportViewer.swing.Viewer.class`. You can also specify to use the Swing Viewer when creating an applet page in the Report Designer.

## 7.6.6. Exporting from Viewer

This section pertains to exporting from the viewer in an Applet only. When using the Report Viewer API to create a `Component` that displays a report, the user can right-click on the `Component` to launch a pop-up menu with various options. One of the options (`Output` → `Server` → `Generate...`) is for exporting the report to different file formats. This will result in creating the exported file on the server side. However, there is an API feature that utilizes a server-side Java Servlet to stream back the exported content to the client's browser. The following describes how to use this feature.

In the code that retrieves the report `Component`, two lines of code need to be added:

```
Viewer viewer = new quadbase.reportdesigner.ReportViewer.Viewer();
Component comp = viewer.getComponent(report);
viewer.setExportServlet("http://host:port/ERES/servlet/ViewerExportServlet");
viewer.setDynamicExport(true, "host", port, "ERES/servlet/");
```

The argument for the `setExportServlet` method is the url location of your deployed `ViewerExportServlet`. Please see the javadoc Specification for more details about the arguments of these two methods.

Then, when the user views the report Component, there will be new options under Output → Client → (Generate...). All of these options will result in a pop-up browser window that contains the streamed content of the report.

## 7.7. Page Viewer

The Report Viewer component offers users the capability of viewing and manipulating a report through a Web browser. The viewer loads the report template on the client and shows it to the user with the latest data. Report Viewer, however, is not the ideal deployment solution when the report contains a large amount of data. In these situations, loading the entire report on the client would cause the client to quickly run out of memory.

For situations like this, EspressoReport provides another viewer called Page Viewer. Instead of loading the entire report on the client, Page Viewer employs a page serving technology that allows the client to load only one page of the report at a time. This allows the report to load faster, and conserve the client memory.

Page Viewer works by exporting the report into a series of page (`.page`) files that contain the drawing information for a set of pages in the report. The `.page` files are then loaded in the viewer on the client. Users running with EspressoManager, can directly pass in a report template as part of Page Viewer code. The report will be exported on the server with the latest data, and the client will load the `.page` files. Users can export the report in “view” format either from the API or Report Designer. These exported files can be loaded directly in Page Viewer.

### 7.7.1. Launching Page Viewer

The Page Viewer can be automatically embedded in a menu page or be manually embedded in an applet using a `.pak`, `.rpt` file or an exported view (`.view`) file. Page Viewer can also be used to load a report directly from the Preview window in Report Designer.

To embed the Page Viewer within a menu page simply select RPT as the export format when viewing the menu page. For more information about the menu page, please see Section 7.1 - The Menu Page.

To embed Page Viewer in an applet, simply use the following code:

```
<applet codebase=".." code="quadbase.reportdesigner.PageViewer.Viewer.class"
  width=100% height=100% archive="PageViewer.jar">
  <PARAM name="filename" value = "yourReport.pak">
</applet>
```

The parameter value of `filename` specifies the name of the file that contains the report. You can specify the name of a report template or an exported view file for this parameter. When using a view file with Page Viewer, you do not have to connect to the ERES Server. The filename value can also be specified by a URL; for example, you can prefix it by `http://` for accessing a remote data file.

You may also use Page Viewer in an application, simply use the following code to get a `java.awt.Component` Object.

```
quadbase.reportdesigner.PageViewer.Viewer.getComponent
(java.awt.Frame frame, java.lang.String filename, long bufferTimeInSec)
```

In addition to passing in the template file (or view file), you can also pass in a `QbReport` object. Note that instead of using the Viewer class, you would be using the ViewerAPI class.

---

```
quadbase.reportdesigner.PageViewer.ViewerAPI.getComponent( java.awt.Frame
    frame, QbReport report, long bufferTimeInSec, java.lang.String
    securityLevel);
```

The above methods are the simplest of all methods for getting Page Viewer as a Component. For more information about the parameters for this constructor or information on other methods related to Page Viewer, please consult the Report API Specification.

Note that on the server side, you may need to copy the `PageViewer.jar` file from the `lib/` directory of the ERES installation to some directory of your web server that is accessible by the web client. Please also note that if you are having troubles with refreshing reports, see Section 7.7.4 - Buffer Time that deals with the `BufferTime` parameter.

### 7.7.1.1. Launching from Report Designer

Page Viewer can also be invoked from Report Designer. When previewing a report with a large amount of data, the report can be loaded in a Page Viewer window. This allows users to preview/view the entire report without loading all the data in memory. For more information about this feature, please see Section 4.1.4.2.4 - Using Page Viewer.

## 7.7.2. The Page Viewer Parameters

Without using Report Designer, you can also pass data via parameters to Page Viewer applet. That is, you can use Page Viewer to directly view a data file, or pass the data directly in the form of lines of data, in the HTML code. The following is a list of parameters:

### Page Parameters

<b>comm_protocol:</b>	The protocol to be used to connect to the ERES Server
<b>comm_url:</b>	The URL to connect to the ERES Server
<b>servlet_context:</b>	The context in which the ERES Server is running
<b>Filename:</b>	Specifies the name of the <code>.pak</code> file
<b>BufferTime:</b>	The time, in seconds, that controls how often to get new pages from the server. It looks at the local version of the <code>.page</code> file (if available), and if the time since it is last updated exceeds the <code>bufferTimeInSec</code> parameter, then new version of the page is requested again from the server. Otherwise it uses the local version of the <code>.page</code> file. If the local version of the <code>.page</code> file does not exist, it gets the page from the server
<b>DataHintOffsetX:</b>	When the user moves the mouse over a data point (a bar on a bar graph or a point on a line graph) on a chart, a hint box that shows the details of the data point is shown. This parameter sets the <i>horizontal</i> offset position of the hint box relative to the data point
<b>DataHintOffsetY:</b>	When the user moves the mouse over a data point (a bar on a bar graph or a point on a line graph) on a chart, a hint box that shows the details of the data point is shown. This parameter sets the <i>vertical</i> offset position of the hint box relative to the data point
<b>DataHintBgColor:</b>	When the user moves the mouse over a data point (a bar on a bar graph or a point on a line graph) on a chart, a hint box that shows the details of the data point is shown. This parameter sets the <i>background</i> color of the hint box
<b>DataHintFontColor:</b>	When the user moves the mouse over a data point (a bar on a bar graph or a point on a line graph) on a chart, a hint box that shows the details of the data point is shown. This parameter sets the <i>font</i> color of the hint box
<b>DataHintFont:</b>	When the user moves the mouse over a data point (a bar on a bar graph or a point on a line graph) on a chart, a hint box that shows the details of the data point is shown. This parameter sets the <i>font type</i> of the hint box

---

<b>Printing:</b>	Valid values: <code>true</code>   <code>false</code> . Sets whether the print option is shown in the pop-up menu
<b>RefreshData:</b>	Valid values: <code>true</code>   <code>false</code> . If false, report cannot be refreshed. The default is <code>true</code>
<b>PopupMenu:</b>	Valid values: <code>true</code>   <code>false</code> . Sets whether the pop-up menu is shown
<b>EnableExport:</b>	Valid values: <code>true</code>   <code>false</code> . If false, report cannot be exported. The default is <code>true</code>
<b>ShowDataHint:</b>	Valid values: <code>true</code>   <code>false</code> . Sets whether to show a data hint box when mouse over a data point in a chart
<b>ShowLinkHint:</b>	Valid values: <code>true</code>   <code>false</code> . Sets whether to show a data hint box when mouse over a hyperlink

Example: With the parameter `BufferTime` you can insert the following line:

```
<PARAM name="BufferTime" value="3600">
```

In the above example, when Page Viewer applet opens, it will fetch data from the specified data source (with the help of the server) if more than 1 hour (3600 seconds) has passed since the local existing `.page` file has been updated. Otherwise Page Viewer will display the local existing `.page` file, if it exists. If it doesn't exist, it also fetches data from the specified data source.

Note that when specifying color parameters Use the Hexadecimal format for RGB colors that is standard in HTML tags. That means the first two hexadecimal digits are for red, the next two are for green, the last two are for blue. A total of six hexadecimal digits has to be specified for a color. For example:

```
<PARAM name="DataHintBgColor" value="ff00aa">
```

### 7.7.3. Using Page Viewer

Using Page Viewer, you can browse through the pages of the report using the Pop-Up Menu. Right clicking on the report displays a Pop-Up Menu which can be used to traverse the report. The Pop-Up Menu can show the following options:

- **Back:** Go back to the previous report. This option is shown when a hyperlink is clicked on.
- **Section:** Contains various options to navigate the various sections
  - First Section:** Go to the first section
  - Previous Section:** Go to the previous section
  - Next Section:** Go to the next section
  - Last Section:** Go to the last section
- **Page:** Contains various options to navigate the various pages
  - First Page:** Go to the first page
  - Previous Page:** Go to the previous page
  - Next Page:** Go to the next page
  - Last Page:** Go to the last page
- **Output:** Contains various formats to output the report to

**Generate DHTML (Single Page):** Export the report to DHTML onto a single file

**Generate DHTML (Multiple Pages):** Export the report to DHTML onto multiple files, one file per page.

**Generate PDF:** Export the report to PDF

**Generate CSV:** Export the report to CSV

**Generate EXCEL (XLS):** Export the report to Excel (xls file)

**Generate EXCEL 2007 (XLSX):** Export the report to Excel 2007 (xlsx file)

**Generate TXT:** Export the report to TXT

**Generate RTF:** Export the report to RTF

**Generate XML (Data + Format):** Export the complete report to XML

**Generate XML (Pure Data):** Export the report data only to XML

**Print:** Print the report

- **Refresh:** Refresh the data from the data source
- **Go To:** Go to the specified page and section
- **Sort by (ascend):** Sort the report based on a selected column's ascending values
- **Sort by (descend):** Sort the report based on a selected column's descending values
- **Show/Hide Report Toolbar:** This either shows or hides the navigation toolbar at the bottom of the viewer window.

In addition to the Pop-Up Menu, you can also

- Left single click to jump to the hyperlink associated with the cell. (Note that this creates a new browser window if the associated URL is **NOT** a .pak or a .rpt file. Users cannot use the Back button on the browser to return to the previous report)
- Run the mouse over a cell containing a hyperlink to see the Hint text associated with the link.

## 7.7.4. Buffer Time

A problem working with the view (.view) and page (.page) files associated with Page Viewer is that while saving memory and boosting performance by caching the report page by page on a file system, Page Viewer files' data might become obsolete in time. In other words, they might not reflect the latest report data. The solution to this is to have a mechanism for updating the Page Viewer files during certain time intervals.

Page Viewer provides the mechanism to adjust the time elapsed until a new view file needs to be fetched from the server. This time interval, also known as the **Buffer Time**, can be adjusted by setting the parameter `BufferTime` in Page Viewer applet. Simply pass in the buffer time in seconds as the value to the parameter `BufferTime`. The Page Viewer will then fetch new pages whenever the current version of the VIEW file becomes older than the specified buffer time.

Also, Page Viewer has a built-in mechanism that automatically eliminates old view and page files when they have not been updated for 3 days or more.

## 7.7.5. Connecting to the ERES Server

In order for the Page Viewer applet to connect to the ERES Server (to fetch the page files and save exported files) you will need to specify the following parameters in your HTML code:

```
<PARAM name="comm_protocol" value="servlet">
<PARAM name="comm_url" value="http://machine:port">
<PARAM name="servlet_context" value="ERES/servlet">
```

Adding the above parameters to your HTML code will make Viewer connect to the server at `http://machine:port/ERES/servlet`.

## 7.7.6. Swing Version Available

A JFC/Swing version of the Page Viewer can also be used by referring to `SwingPageViewer.jar` instead of `PageViewer.jar` in the `ERES/lib` directory.

## 7.7.7. Exporting from Viewer

This section pertains to exporting from the viewer in an Applet only. When using the Page Viewer API to create a Component that displays a report, the user can right-click on the Component to launch a pop-up menu with various options. One of the options (Output → Server → Generate...) is for exporting the report to different file formats. This will result in creating the exported file on the server side. However, there is an API feature that utilizes a server-side Java Servlet to stream back the exported content to the client's browser. The following describes how to use this feature.

In the code that retrieves the report Component, two lines of code need to be added:

```
Viewer viewer = new quadbase.reportdesigner.PageViewer.Viewer();
Component comp = viewer.getComponent((Applet)null, reportTemplate, 0);
viewer.setExportServlet("http://host:port/ERES/servlet/
ViewerExportServlet");
viewer.setDynamicExport(true, "host", port, "ERES/servlet/");
```

where `reportTemplate` is a string containing the name and path of the report template file. You can also pass in a `QbReport` object using the following code:

```
ViewerAPI viewer = new quadbase.reportdesigner.PageViewer.ViewerAPI();
Component comp = viewer.getComponent((Applet)null, report, 0, null);
viewer.setExportServlet("http://host:port/ERES/servlet/
ViewerExportServlet");
viewer.setDynamicExport(true, "host", port, "ERES/servlet/");
```

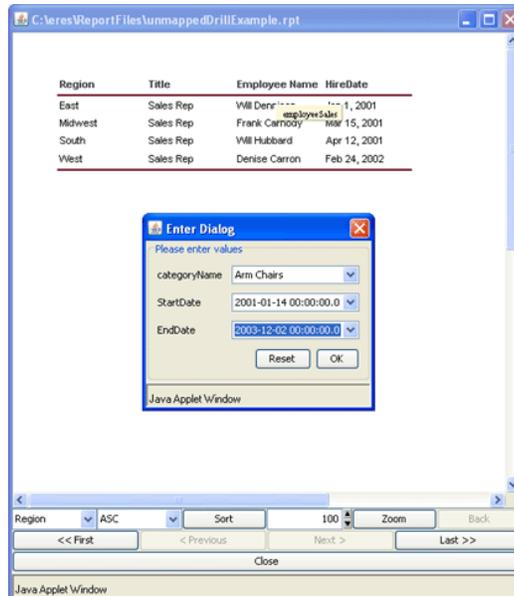
where `report` is a `QbReport` object.

The argument for the `setExportServlet` method is the url location of your deployed `ViewerExportServlet`. Please see the javadoc Specification for more details about the arguments of these two methods.

Then, when the user views the `reportComponent`, there will be new options under Output → Client → (Generate...). All of these options will result in a pop-up browser window that contains the streamed content of the report.

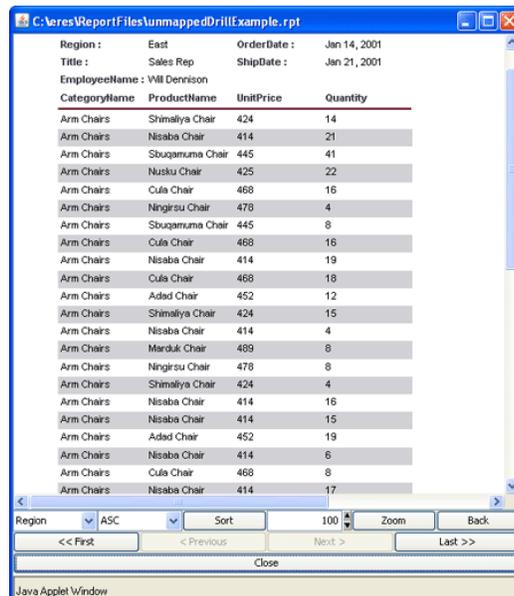
## 7.7.8. Unmapped Drilldown Parameter

When viewing a report with unmapped drilldown parameters, clicking on the links in the main report will cause a parameter selection dialog to pop up allowing you to fill in the unmapped parameters.



*Parameter Selection Dialog*

Once the values are entered, click *OK* and the drill report will be shown. To return to the previous level, click on the *back* button in the page viewer toolbar.



*Drill-Down Level*

## 7.8. Chart Viewer

All charts created by Chart Designer can be saved in a range of file formats that may be pasted into documents. These formats include BMP, JPG, PNG, PDF, SVG, SWF, WMF, and GIF. In addition, Chart Designer provides the option of saving a chart as a *.cht*, *.tpl*, or *.xml* file.

Chart Viewer is a JNLP (More about Applets in JNLP: Section 1.3.4 - Run Applets As JNLP) that enables you to view and manipulate a chart dynamically through a web browser. Viewer reads the file (in *.cht* or *.tpl* format) as outputted by Chart Designer or the API, and then displays the chart. The small size of the data file makes it suitable to distribute the chart image over the web. The data is encrypted while being transferred from EspressoManager to Chart Viewer and so lends a degree of security to sensitive information.

Files in the `.tpl` format can also be viewed using Chart Viewer. When a web page that contains a `.tpl` file is viewed using a browser, fresh data will be obtained automatically by Chart Viewer. Thus, a single chart template can be used to supply up-to-the-minute charts to users in real time.

Inside Chart Viewer, you can drag the chart, legend, title, or label to position the object. You can also resize the chart and drill-down on data points or a series of data. For three-dimensional charts, users can use the navigation panel to pan, zoom, rotate in each direction, and translate. Also, individual x, y, and z-axis scaling, thickness ratio adjustment, real time three-dimensional animation, etc can all be preformed easily. There are built-in callback mechanisms that let a user click on a data element to view the underlying data or to jump to a related URL. Chart Viewer is written in pure Java that runs on all platforms that support Java. Chart Viewer also supports scheduled refresh (where a chart's data is updated at regular intervals specified by the designer) and parameter serving where a chart's parameters are provided at load time.

Chart Viewer is an applet that allows you to view and manipulate a chart dynamically through a web browser. The Viewer can read chart files (`.cht`), template files (`.tpl`) and chart XML files. Chart Viewer can be automatically embedded in a menu page, in an HTML page, or manually placed in a web page.

Inside the applet, you can drag the chart, legend, title, or label to position the object. You can resize the chart and drill-down on data points or a series of data. For three-dimensional charts, users can use the navigation panel to pan, zoom, rotate in each direction, and translate. Also, individual x, y, and z-axis scaling, thickness ratio adjustment, real time three-dimensional animation, etc can be preformed easily. There are built-in callback mechanisms that let a user click on a data element to view the underlying data or to jump to a related URL. Chart Viewer also supports scheduled refresh (whereby a chart's data is updated at regular intervals specified by the designer).

To embed Chart Viewer within a menu page, simply select *CHT* as one of display options when viewing the menu page. For more information about the menu page, please see Section 7.1 - The Menu Page.

To create a stand-alone HTML page with the Viewer embedded, check the *Create HTML File* option when saving a chart in the Chart Designer.

In addition you can embed the Viewer with your own Web page using the following syntax:

```
<applet-desc
  name="Chart Viewer"
  main-class="quadbase.chartviewer.Viewer"
  width="800"
  height="600">
  <param name="filename" value="help/examples/ChartAPI/data/test.tpl"/
>
  <param name="preventSelfDestruct" value="false"/>
</applet-desc>
```

The parameter `filename` specifies the file name of the file that contains the chart data and you can prefix it by `http://` for accessing a remote data file. When viewed by Chart Viewer, a chart saved in the chart format (`.cht`) will use the data stored in that file for plotting.

A chart saved in the template format (`.tpl`) allows Chart Viewer to dynamically fetch the data from a database or a data file depending on where you specify the data source of chart to be when using Chart Designer to create the template (the database name, user name, password, etc are all stored in the `.tpl` file).

## 7.8.1. The Chart Viewer Parameters

You can also pass data and chart viewing control information via parameters to the Chart Viewer JNLP (More about Applets in JNLP: Section 1.3.4 - Run Applets As JNLP) without using Chart Designer. That is, you can use the Chart Viewer to directly view a data file or pass the data directly in the form of lines of data, along with other control information, in the HTML code. By default, the parameter is true if the parameter is of a true/false type. The following is a list of parameters:

Chart Parameters (Common to Two and Three-Dimensional Charts):

<b>mainTitle</b>	the main title of the chart
<b>xTitle</b>	x-axis title

---

<b>yTitle</b>	y-axis title
<b>zTitle</b>	z-axis title
<b>RefreshInterval</b>	scheduled refresh interval in seconds
<b>DragLegend</b>	if false, the legend(s) can not be moved
<b>DragChart</b>	if false, the chart can not be repositioned or resized
<b>ShowDataHint</b>	if false, data information box will not be shown when left mouse click on chart data
<b>ShowLinkHint</b>	if false, hyperlink information box will not be shown when right mouse click on chart data
<b>DataHintBgColor</b>	set background color of data information box
<b>LinkHintBgColor</b>	set background color of hyperlink information box
<b>DataHintFontColor</b>	set font color of data information box
<b>LinkHintFontColor</b>	set font color of hyperlink information box
<b>DataHintFont</b>	set font of data information box
<b>LinkHintFont</b>	set font of link information box
<b>DataHintOffsetX</b>	set x offset of the data information box
<b>DataHintOffsetY</b>	set y offset of the data information box
<b>LinkHintOffsetX</b>	set x offset of the link information box
<b>LinkHintOffsetY</b>	set y offset of the link information box
<b>Printing</b>	if false, this will disable the ability to export the chart (using <b>Ctrl+P</b> and/or <b>Ctrl+J</b> ) in a browser
<b>filename</b>	name of the template file to be applied to the chart
<b>xAxisRuler</b>	if true, show the x-axis ruler (for 2D charts only)
<b>yAxisRuler</b>	if true, show the y-axis ruler (for 2D charts only)
<b>sAxisRuler</b>	if true, show the secondary-axis ruler (for 2D charts only)
<b>ResizeChart</b>	if false, the chart cannot be resized
<b>ResizeCanvas</b>	if false, the canvas cannot be resized
<b>comm_protocol</b>	the protocol to be used, to connect to the ERES Server
<b>comm_url</b>	the URL to connect to the ERES Server
<b>servlet_context:</b>	the context in which the ERES Server is running
<b>RefreshData</b>	if false, the chart data cannot be refreshed
<b>PopupMenu</b>	if false, the pop-up menu will not be displayed
<b>TypeMenu</b>	if false, the type sub-menu will not be displayed in the pop-up menu
<b>DimensionMenu</b>	if false, the dimension sub-menu will not be displayed in the pop-up menu
<b>For Three-Dimensional Charts only</b>	
<b>Toggle3Dpanel</b>	if false, the navigation panel can not be toggled to be visible or invisible

---

---

<b>Drawmode</b>	set different mode of drawing 3D chart. Available draw modes are <b>Flat</b> (default), <b>WireFrame</b> , <b>Flat Border</b> (which draws a black border around the flat shading model), <b>Gouraud</b> , and <b>Gouraud Border</b>
<b>NavColor</b>	set navigation panel color
<b>navpanel</b>	if false, the Navigation Panel is not displayed when a 3D chart is being viewed. The Navigation Panel is never displayed when a 2D chart is being viewed
<b>GouraudButton</b>	if false, the Gouraud shading button in the navigation panel is hidden
<b>AnimateButton</b>	if false, the animation speed control in the navigation panel is hidden
<b>SpeedControlButton</b>	if false, the speed control button in the navigation panel is hidden

**Data Input Parameters**

<b>sourceDB</b>	set the database information in order to generate the chart
<b>sourceData</b>	set the data information in order to generate the chart
<b>sourceFile</b>	set the datafile information in order to generate the chart
<b>datamap</b>	set the column mappings for the chart
<b>TransposeData</b>	set the data to be transposed before using it to generate the chart
<b>chartType</b>	set the chart type for the generated chart
<b>ParameterServer</b>	update the data in the chart dynamically
<b>transposeData</b>	if true, transpose the data

*Example:* the parameters `mainTitle`, `xTitle`, `yTitle`, and `zTitle` are used to specify the main title and axis title of the chart, they will override the ones defined in the template:

```
<PARAM name="mainTitle" value="This is the main Title">
<PARAM name="xTitle" value="x axis title">
<PARAM name="yTitle" value="y axis title">
<PARAM name="zTitle" value="z axis title">
```

*Example:* With the parameter `RefreshInterval` you can specify:

```
<PARAM name="RefreshInterval" value="60">
```

In the above example, the applet will fetch data from a database or data file (with the help of the server) and redraw the chart every 60 seconds - all transparently. It is useful for accessing databases in which the data changes frequently.

## 7.8.2. Specifying the Data Source for Chart Viewer

Using parameters, you can specify a data source for the Chart Viewer in order to display different data with a chart template or create a chart from scratch.

### 7.8.2.1. Data Read From a Database

This is some sample JSP/JNLP code that uses the Chart Viewer to view a chart drawn using data extracted from a database.

```
<?xml version="1.0" encoding="utf-8"?>
<jnlp spec="1.0+" href="EspressoViewer.jnlp">
<information>
```

```
<title>Espress Viewer</title>
<vendor>Quadbase Systems Inc.</vendor>
<offline-allowed/>
</information>
<resources>
<j2se version="1.8+" max-heap-size="1024m"/>
<jar href="lib/EspressViewer.jar"/>
</resources>
<security>
<all-permissions/>
</security>
<applet-desc
name="Chart Viewer"
main-class="quadbase.chartviewer.Viewer"
width="640"
height="480">

<PARAM name="sourceDB" value="jdbc:odbc:DataSource,
  sun.jdbc.odbc.JdbcOdbcDriver, username, password, select * from products">
<param name="dataMap" value="0 1 -1 3">
<param name="chartType" value="3D Column">

</applet-desc>
<update check="always" policy="always"/>
</jnlp>
```

The arguments of `dataMap` specify how the chart utilizes different columns from the input data to plot the chart. In case of a scatter chart, they are series, x-value, y-value, and z-value. For a high low open close or high low chart the numbers are series, category, high, low, open, and close. For all other charts, the arguments are series, category, sumBy, and value. If you would like more information, please see the chapter on Column Mapping in the Chart API reference. The argument `chartType` specifies the type of chart to be displayed and it can be one of:

- 2D Column
- 3D Column
- 2D bar
- 3D bar
- 2D stack bar
- 3D stack bar
- 2D stack column
- 3D stack column
- 2D area
- 3D area
- 2D stack area
- 3D stack area
- 2D line
- 3D line
- 2D pie
- 3D pie

- 2D scatter
- 3D scatter
- 2D High Low
- 3D High Low
- 2D HLCO
- 3D HLCO
- 2D 100% Column
- 3D 100% Column
- 3D Surface
- 2D Bubble
- 2D Overlay
- 2D Box
- 2D Radar
- 2D Dial
- 2D Gantt
- 2D Polar

### 7.8.2.2. Data Read From a Data File

This HTML/jsp code draws a chart using data from a data file:

```
<applet-desc code = "quadbase.chartviewer.Viewer.class" width=640
height=480>

    <param name="sourceFile" value="http://.../test.dat">
    <param name="dataMap" value="-1 0 -1 1">
    <param name="chartType" value="3D Pie">
</applet-desc>
```

### 7.8.2.3. Data Read From an Argument

It is possible to have Chart Viewer read in data directly from the HTML/jsp file itself rather than from a data file or from a database.

```
<applet-desc code = "quadbase.chartviewer.Viewer.class" width=640
height=480>

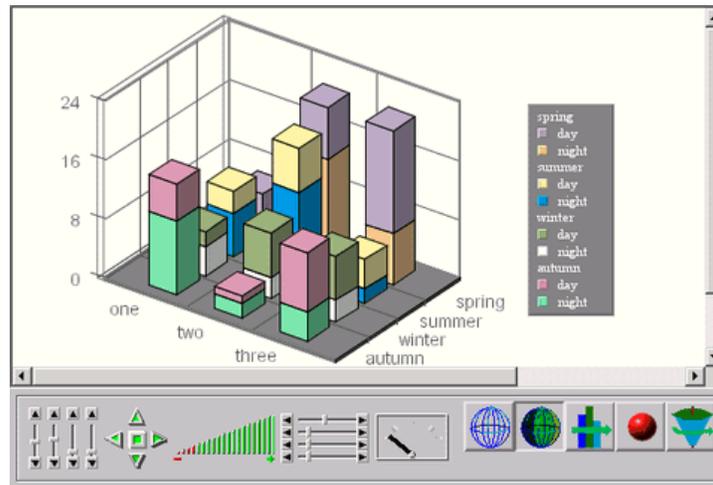
    <param name="sourceData" value="int, string, int | value, name, vol |
10, 'John', 20 | 3, 'Mary', 30 | 8, 'Kevin', 3 | 9, 'James', 22">
    <param name="dataMap" value="-1 1 -1 2">
    <param name="chartType" value="3D Bar">

</applet-desc>
```

The format for the parameter sourceData is the same as the data file format, except that each line is ended by a vertical bar “|”.

### 7.8.3. Using Chart Viewer

You can manipulate the chart appearance simply by using the mouse in Chart Viewer. For three-dimensional charts, Chart Viewer screen comprises of two sections: the *drawing panel* and the *navigation panel*. Only the *drawing panel* is visible for two-dimensional charts. Within the *drawing panel*, the chart itself is located on the *plot area*.



*A Sample Chart Viewer Display*

Below is a list, not by order or importance, of chart manipulations that are possible when using Chart Viewer:

- Using the navigation panel, you can:
  - Change light position
  - Rotate the chart
  - Translate the chart
  - Zoom in/out on the chart
  - Scale the chart in x, y, z dimension
  - Adjust the thickness ratio of bar/line/point/pie
  - Start the chart animation and control the speed of the animation
  - Toggle between wireframe and solid mode
  - Draw a black outline on all edges of the chart
  - Perform Gouraud shading
- Drag the left mouse button on the chart, main title, x, y, and z-labels, or legend to move the item
- Drag the right mouse button on plot area to resize the chart
- Press the **Alt** key and drag the right mouse button on plot area to resize the canvas
- To toggle the navigation panel (show and hide), **double click** the left mouse button on the drawing panel
- Using the mouse you can also query data points individually
- **Left single click** on a data point to view the data associated with that point
- A **right single click** on data provides the name of the hyperlink associated with the data point

- **Left double click** to jump to the hyperlink associated with the data point. (Note that this creates a new browser window. Users can not use the *Back* button on the browser to return to the previous chart)
- **Right double click** to jump back to the previous chart (if appropriate)
- Use **Alt+Z** to specify the zoom-in parameters
- Use **Ctrl+R** to refresh data manually
- Use **Ctrl+Left Click** to select the bounds to zoom in and **Ctrl+Right Click** to zoom out

## 7.8.4. Axis Rulers

There is now an option to show Axis Rulers in Chart Viewer (by setting a parameter `{axisRuler}` in the applet-desc tag) and Chart API (Please refer to the API Documentation `quadbase.util.IAxisRuler` [ <https://data.quadbase.com/Docs70/eres/help/apidocs/quadbase/util/IAxisRuler.html> ]). This provides a reference point when scrolling is enabled and the chart is moved such that the chart's own axes are not visible. This feature is for 2D charts only.

## 7.8.5. Pop-up Menu

If the pop-up menu option was selected during the chart creation/editing, the chart can be modified. The menu is opened by right clicking on the chart and the options there can be selected by highlighting and left clicking on it. Options available for viewing, when enabled, are dynamic data drill-down, changing chart types, axis zooming, and time-series data zooming.

### 7.8.5.1. Changing Chart Dimension and Type

The chart dimension and chart type can be changed by using the pop-up menu and selecting either the *2D/3D* option or the *Type* option.

Please note that *Overlay*, *Box* and *Dial* chart options do not appear if the chart is three-dimensional.

### 7.8.5.2. Axis Zooming

In the case of a large chart whose dimensions exceed that of the viewport (i.e. if a chart is large enough that it cannot be completely seen within window), an option exists to allow for axis zooming. This allows you to move all the axis and zoom in and out of the axis.

This option is only available for two-dimensional charts. Please note that the bubble, dial, pie, polar, radar, scatter, stack area, and vertical box charts do not support this feature.

You can enable this option by opening the pop-up menu and selecting *Enable Zoom*. You can choose from the zooming in on the x-axis, the y-axis or both.

To zoom in, left click on and drag out the desired area. If only the y-axis zooming is selected, you need not worry about the length or position in the x-axis.

Scrolling is enabled only for axes that have been zoomed. There will be scrollbars visible on the particular axis when axis zooming is enabled. You can left click and drag this scrollbar to shift the viewport of the applet.

Holding down the control key while left clicking will take the applet back to the previous zoom. Please note that only one previous zoom is stored in memory and thus you can only go back one step. To go all the way back to the default x and y scale, press the **Home** key on your keyboard.

### 7.8.5.3. Zooming

If the chart being shown is a zoom enabled chart, the zoom option will be available from the pop-up menu. Using the zooming option, you can group the category elements into user-defined intervals and aggregate the points in each group. For details on date/time based zooming, see Section 4.2.4.8.2 - Date/Time Based Zooming. Enabling zoom will allow you to input various zoom options such as lower bound, upper bound (you can also disable the bounds in which case it will go from the first data point to the last), the time scale, and whether you want the x-axis to be linear or not.

### 7.8.5.4. Dynamic Data Drill-Down

You can configure the next drill-down by selecting *drill-down* from the pop-up menu. This option is available only if dynamic data drill-down is enabled for the chart. After selecting the *Drill-Down* option in the pop-up menu, you can view the current setting for the next drill-down chart. If the *Category* is *None*, the next level has not yet

been configured. If there are unused columns in the data used to generate the chart, you can select a column for the *Category*. After the *Category* has been selected, the *Type* of the chart in the next level can be set (along with *Series* and *SumBy*). The data points can now be left clicked to move down a level and right clicked to move up a level.

### 7.8.5.5. Query Parameter

When viewing a parameterized chart, you can enter in a different value for the parameter (or parameters depending on the number) by opening the pop-up menu and selecting the *Query Parameter* option. The chart is then redrawn with the new data based on the parameters selected.

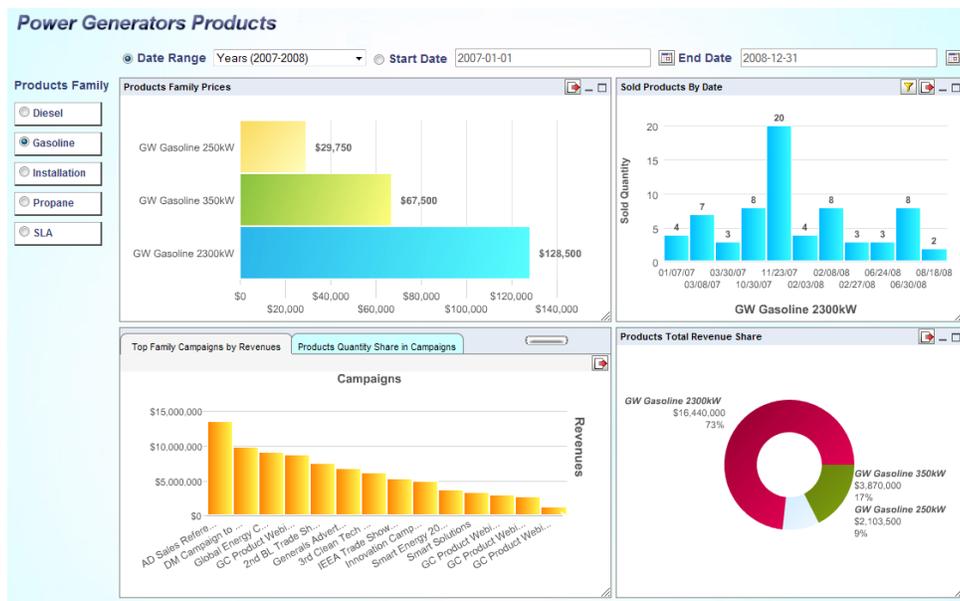
### 7.8.6. Swing Version Available

A JFC/Swing version of the Chart Viewer can also be used by referring to `SwingEspressViewer.jar` instead of `EspressViewer.jar` in the `ERES/lib` directory. Call the following class when using the Swing viewer: `quadbase.chartviewer.swing.Viewer.class`.

## 7.9. Dashboard Viewer

Dashboard Viewer is an applet that enables you to view and manipulate a dashboard dynamically through a web browser. Viewer reads the file (in `.dsb` format) as outputted by Dashboard Builder and then displays the dashboard.

The dashboard can be previewed by clicking the *Preview* button  on the Dashboard Builder toolbar. This will open a new window showing the dashboard.



Dashboard

### 7.9.1. Preview Toolbar

The top part of the Preview interface contains a small toolbar that allows you to initiate the following actions:



Set shared parameter(s)



Refresh a dashboard



Set auto refresh

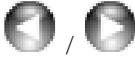


Export Dashboard to PDF (Note: All objects in dashboard including Online maps are exported, however, if map type is set to Google street map or Google satellite, map imagery will be missing in the exported file due to its license restrictions.)



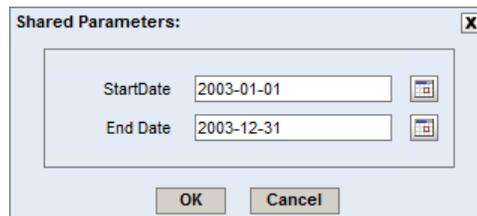
Pack/Unpack the preview toolbar

## 7.9.2. Preview Options

This section gives you description of options available from the preview toolbar and additional options available from the chart/report/map header bar. You can hide/show the toolbar by clicking the  Pack/Unpack icons.

### Set Shared Parameters:

To set shared parameters, click the *Set Shared Parameters* button  on the preview toolbar. After you click the button, the shared parameters dialog will then appear. The dialog allows you to specify values for shared parameters.



The dialog box titled "Shared Parameters:" contains two text input fields. The "StartDate" field is set to "2003-01-01" and the "End Date" field is set to "2003-12-31". Each field has a small calendar icon to its right. At the bottom of the dialog are "OK" and "Cancel" buttons.

*Set Shared Parameters Dialog*

Note that you can use calendar for specifying date parameters. To open the calendar, simply click the *Calendar* button  in the shared parameters dialog.



A calendar widget showing the month of December for the year 2009. The days of the week are listed as S M T W T F S. The date 30 is highlighted with a blue border. Below the calendar is a "Today" link.

*Calendar*

Once you finish specifying the shared parameters, click the *Ok* button to save the changes. You will be taken back to the Preview.

### Auto Refresh:

If you want your dashboard to be refreshed periodically, click the *Auto Refresh* button  on the preview toolbar. After you do so, the *Auto Refresh* dialog will then appear. You have to check off the *AutoRefresh* checkbox to enable the auto refresh feature. The dialog allows you to set the refresh interval in seconds. For example, if you specify the refresh interval for 10 seconds, the dashboard will refresh every 10 seconds.



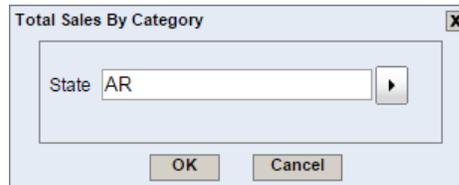
The dialog box titled "Auto Refresh:" contains a checkbox labeled "AutoRefresh" which is currently unchecked. Below it is a text input field labeled "Interval(sec):" with the value "5" and a small spinner control to its right. At the bottom are "OK" and "Cancel" buttons.

*Auto Refresh Dialog*

Once you finish setting up the auto refresh options, click the *Ok* button to save the changes. You will be taken back to the Preview.

In addition to the options on the preview toolbar, there are additional options available for a chart/report/map that can be triggered by clicking the small buttons in the chart/report/map header bars.

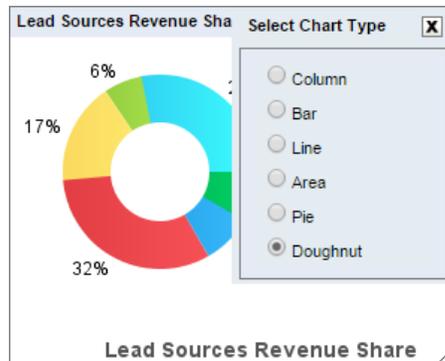
 **Filter:** This option will open a new dialog that allows you to specify additional filters/parameters for the chart/report/map, if it has parameters that aren't associated with one of the dashboard's shared parameters.



*Dashboard Filter Chart/Report/Map Dialog*

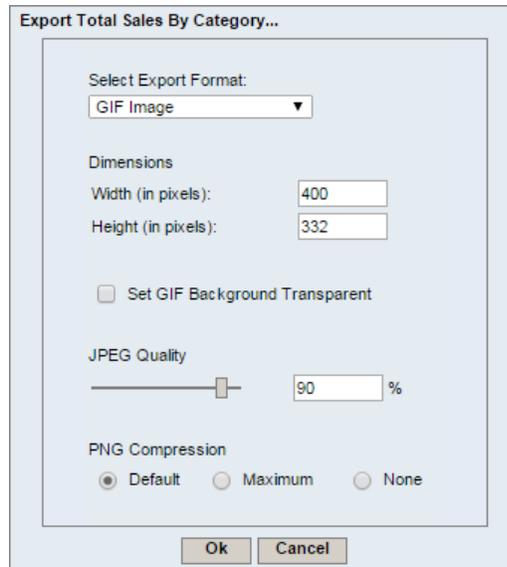
You can select parameter values in this dialog. Once you complete your selection, click the *Ok* button. The dashboard will refresh and the chart/report/map will reflect the new parameter values.

 **Change Chart Type:** This option will open a new dialog that allows you to change a chart type. The chart type is changed only in the Preview. It is not possible to save the changed view of the chart, but you can export it.



*Select Chart Type Dialog*

 **Export:** This option will open a new dialog that allows you to export the chart/report/map in a variety of formats. For charts the following dialog is displayed:



*Dashboard Export Chart Dialog*

The first option allows you to specify the format in which to export the chart. The available formats are GIF, JPEG, PNG, PDF, SVG, Excel Image (XLS), Flash, Text Data File, XML Data File and MAP Data File (HTML Image Map).

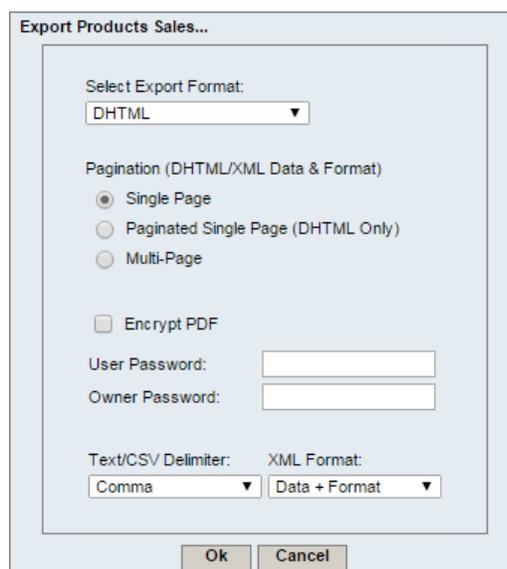
The second option allows you to set dimensions of the exported image. By default, these will match the canvas dimensions of your chart.

The last set of options allows you to set some image type-specific options. You can set the background transparent for GIF files, quality for JPEG files, and specify encoding for PNG files.

For more information about chart export options, please see Section 4.2.6.3 - Exporting Charts.

Once you finish specifying the options for exporting the chart, click the *Ok* button. A new window will open containing the exported chart.

For reports the following dialog is displayed:



*Dashboard Export Report Dialog*

---

The first option in the dialog allows you to specify in which format you want to export the report. Available options are PDF, CSV, Excel (XLS), Excel 2007 (XLSX), DHTML, text, XML and rich text.

Other options allows you to set single or multi-page export for DHTML and XML exports. You can enable encryption for PDF format. For text export you can select to use the delimiter and for XML export you can select whether to export only report data, or XML description of the report and data.

For more information about the export formats and options, please see Section 4.1.5.2 - Exporting Reports.

Once you finish specifying the options for exporting the report, click the *Ok* button. A new window will open containing the exported report.

Online Maps and SVG Maps support only one export format, PDF, so no export format settings dialog is displayed for maps. Please note that Online Maps cannot be exported if Google street map/Google satellite is used as map type, due to license restrictions. In this case, map imagery will be missing in the exported file.

- Maximize:** This option maximizes the report/chart/map.
  
- Minimize:** This option minimizes the report/chart/map (only template header bar is visible).
  
- Restore:** This option resizes the report/chart/map to its original size.

---

# Chapter 8. Programming

## 8.1. ERES Report API

### 8.1.1. Introduction and Setup

ERES provides an easy-to-use application programming interface (API) that enables users to create and customize reports within their own applications (and applets) on either the server-side or on the client-side. It is written in 100% Pure Java and thus can be run on any platform with little or no changes. Any and every part of the report is customizable using the API. You can use as little as a single line of code to generate a report.

The main class, `QbReport`, extends `java.awt.Component`. Associated with this component is a set of auxiliary classes consisting of five packages: `quadbase.reportdesigner.ReportAPI`, `quadbase.reportdesigner.ReportElements`, `quadbase.reportdesigner.ReportViewer`, (and its swing counterpart `quadbase.reportdesigner.ReportViewer.swing`), `quadbase.reportdesigner.lang` and `quadbase.reportdesigner.util`. The remainder of this document explains the constituents of the API and their use.



#### Note

The complete API documentation is located at [help/apidocs/index.html](http://help/apidocs/index.html).

To use the API, add `ERESOrganizer.jar` (located in the `ERES/lib` directory) and `ERESServer.jar` (located in the `ERES/WEB-INF/lib` directory) to your `CLASSPATH`. Please note that if you are also using XML (to read or write data), you will also need to add `xercesImpl.jar` and `xml-apis.jar` (also located in the same directory) to the `CLASSPATH` as well. To export the report to an Excel file (i.e. a `.xls` file), you must include `poi.jar` in your `CLASSPATH`. If you want to export to the MS Excel 2007 format (OOXML - extension `.xlsx`), you must also include these files in your `CLASSPATH`: `poi-ooxml.jar`, `poi-ooxml-schemas.jar`, `commons-codec.jar`, `commons-collections.jar`, `commons-compress.jar`, `commons-io.jar`, `commons-math3.jar`, `log4j-api.jar`, `SparseBitSet.jar`, `xmlbeans.jar`. If you want to use parameterized database queries as your data sources, add `jsqlparser.jar` to your `CLASSPATH`. Please note that you will also need to include `qblicense.jar` in your `CLASSPATH`. If your application is on a Windows or Solaris machine, you will have to add the following environment variable (depending on the platform):

```
(For Windows) set PATH=%PATH%;<path to ERES root directory>\lib
```

```
(For Solaris) export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:<path to ERES root directory>/lib
```

Please note that third-party jar files may also be required, depending on what your application does. For example, if you make a JDBC connection to a database, you will need to include the JDBC jar files in the `CLASSPATH` as well.

### 8.1.2. Recommended Approach for Using Report API

ERES provides API through which reports can be generated programmatically from scratch. However, this generally involves a significant amount of code. We recommend using Report Designer and creating a Report File (`.PAK` file) first, which can function as a template. This template can then be passed in the `QbReport` constructor and the template's look and feel can be applied to the newly created report object. Thus, at the time of the report generation, a major portion of the look and feel will have been set. You can also open a Report File (either a `.RPT` or a `.PAK` file) using the API. Using either approach, you can then write code to modify, add, or remove the properties. This approach will save you coding time and improve performance as well.

When using ERES Report API, you have the option of connecting to the ERES Server or not. While a connection is required when using Report Designer, you do not need to connect to the ERES Server when running any application

---

that utilizes ERES Report API. We generally recommend that you do not connect to the ERES Server and thereby avoid another layer in your architecture. For more details, please refer to the next section.

All examples and code given in the manual follow the above two recommendations, i.e. a template based approach and no connection to ERES Server. Unless otherwise noted, all code examples will use a template (templates can be downloaded from corresponding chapters) and will not connect to ERES Server.

Also note that if you have applets that use ERES Report API, the browser must have at least a 1.5 JVM plugin.

### 8.1.3. Interaction with ERES Server

Before we go into the details of how to create and use reports, let's explore the options of using ERES Server in an application. ERES is generally used in conjunction with ERES Server. The report component connects to ERES Server in order to read and write files, to access databases, and to perform data pre-processing required for certain advanced features (such as aggregation). However, the report component can also be used in a stand-alone mode, in which it performs file I/O and database access directly, without the use of ERES Server.

Both approaches have their own advantages. If the report is contained within an applet, security restrictions prevent it from directly performing file input/output. Database access is also difficult without the presence of a JDBC driver on the client. In such cases, ERES Server provides the above services to the report. On the other hand, if the report is used in an application, there are no such restrictions, and performance can be improved by direct access. The application can be run without the necessity of having ERES Server running at a well-known location.

For instance, the applet or application may be running on machine **Client** and may require data from a database on machine **DbMachine**. However, machine **DbMachine** may be behind a firewall or a direct connection may not be allowed to machine **DbMachine** from machine **Client** due to security restrictions. The ERES Server can be run on machine **Server** and the applet/application can connect to the ERES Server on machine **Server**. JDBC can then be used to connect to machine **DbMachine** from machine **Server** and get the data. The data is then delivered to machine **Client** and the report is generated. This is useful when you want to keep the data secure and non-accessible from your client machines and make all connections come through a server machine (a machine running ERES Server). You can also utilize this option to keep a log of all the clients accessing the data through ERES (you can have a log file created when starting ERES Server. The log file is called `espressmanager.log`). Note that this functionality comes at a cost. You will face a slight performance overhead because your code is connecting to the data through ERES Server (i.e. another layer has been added).

By default, a report component requires the presence of ERES Server. To change the mode, use the `QbReport` class static method at the beginning of your applet/application (before any `QbReport` objects are created):

```
static public void setEspressManagerUsed(boolean b)
```

Both applications and applets can be run with or without accessing ERES Server. Communication is done using http protocol. The location of the server is determined by an IP address and port number passed in the API code. Below are instructions on how to connect to the ERES Server.

### 8.1.4. Connecting to ERES Server

If you wish to use Report API to connect to ERES Server, you will have to use the following methods:

```
public static void useServlet(boolean b);  
public static void setServletRunner(String comm_url);  
public static void setServletContext(String context);
```

For example, the following lines of code:

```
QbReport.useServlet(true);  
QbReport.setServletRunner("http://someMachine:somePortNumber");  
QbReport.setServletContext("ERES/servlet");
```

will connect to ERES Server running at `http://someMachine:somePortNumber/ERES/servlet`.

Please note that these methods exist in the `QbReport`, `QbChart`, `QbReportDesigner`, and `QbOrganizer` classes.

## 8.1.5. Using the API

The following section details how to utilize the API. Again, the API examples and code is designed using the above recommendation (template based and no ERES Server).

Note that unless otherwise noted, all examples use the Woodview HSQL database, which is located in the `<ERESInstall>/help/examples/DataSources/database` directory. In order to run the examples, you will need to add database HSQL JDBC driver (`hsqldb.jar`) to your classpath. The driver is located in the `<ERESInstall>/WEB-INF/lib` directory.

Also, all the API examples will show the core code in the manual. To compile the examples, make sure the `CLASS-PATH` includes `ERESOrganizer.jar`, `ERESServer.jar` and `qblicense.jar`.

For more information on the API methods, please refer the API documentation [ <http://data.quadbase.com/Docs70/eres/help/apidocs/index.html> ].

### 8.1.5.1. Loading a Report

Reports can be saved to a file using the PAK format (a proprietary format) or XML format. A PAK/XML file stores all report information except actual data (although an option exists to save the entire data within a PAK file). This format can be used to reconstruct a report object. The data is automatically reloaded from the original data source each time the PAK/XML file is opened.

It is important to note that the PAK/XML file, by default, does **NOT** contain the data. It contains, along with the report template information (i.e. the look and feel of the report), the specified data source. So, when loading a PAK/XML file, the data for the report is obtained by querying the data source. This format can be obtained by using Report Designer or the `export()` method provided in the `QbReport` class. You can also choose to include the complete data in the PAK file (along with the data source information).

The following example, which can run as an applet or application, reads a PAK file and reconstructs a report:

```
Component doOpeningTemplate(Object parent) {
    // Do not use ERES Server
    QbReport.setEspressManagerUsed(false);

    // Open the template
    QbReport report = new QbReport (parent, // container
                                   "OpeningTemplate.rpt"); // template

    // Show report in Viewer

    return (new Viewer().getComponent(report));
}
```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/OpeningTemplate.zip> ]

Exported Results [ <http://data.quadbase.com/Docs70/help/manual/code/export/OpeningTemplate.pdf> ]

The constructor in this example is:

```
public QbReport(Object parent, String file);
```

where `parent` is the container and `file` is a PAK/XML file name.

**Note**

File names may be specified either as URL strings or using relative/absolute paths. Path names are interpreted as being relative to the current directory of ERES Server, or to the current application if ERES Server is not used.

**8.1.5.1.1. Sub-Reports, Charts ,and Drill-Down Reports**

A report template may contain charts, sub-reports, and/or drill-down reports. These ancillary templates are saved separately from the main report template. Chart templates are saved in the `chart` directory, sub-report templates in the `SubReport` directory and drill-down templates in the `DrillDown` directory. A relative URL (relative to the ERES installation directory) and the template name is then specified and put in the main report template.

When using the API and not connecting to ERES Server, the code looks for the template relative to the working directory. However, methods are available in the API that allow you to specify the location of the chart, drill-down, image, and sub-report directories.

The following example, which can run as an applet or application, reads a RPT file and sets the Chart Path:

```
Component doSetChartPath(Object parent) {

    // Do not use ERES Server
    QbReport.setEspressManagerUsed(false);

    // Open the template
    QbReport report = new QbReport (parent,           // container
        "setChartPath.rpt"); // template

    // Set Chart Path
    report.setChartPath(".");

    // Show report in Viewer
    return (new Viewer().getComponent(report));
}
```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/SetChartPathERES.zip> ]

Exported Results [ <http://data.quadbase.com/Docs70/help/manual/code/export/SetChartPath.pdf> ]

The above code sets the location of the chart template file, thus avoiding the need to have a Chart subdirectory under the working directory of your class file.

The following methods work similarly:

```
QbReport.setDrillDownPath(String directory);
QbReport.setSubReportPath(String directory);
QbReport.setImagePath(String directory);
```

**8.1.5.1.2. PAK File**

You can also pack your report in PAK format and deploy it as a PAK file. This alternative simplifies the deployment procedures by including all the chart, sub-report, drill-down, and image files that are associated with your main report, into one `.pak` file. Your code can use the `.pak` file name (instead of the `.rpt` / `.xml`) and creates the report. With the `.pak` file approach, you do not need to specify the directories where the chart, sub-report and/or drill-down templates are located.

**8.1.5.1.3. Backup Data**

By default, report templates are always saved with two rows of backup data. This is to ensure that any template can be used to create the `QbReport` object even if the data source is not present. When you create a `QbReport` object based on an existing template, it looks for the data source, specified in the template, and if the data source

is not found, the backup data is used. However, you can force the API to use the backup data instead of searching for the data source using the following constructor:

```
public QbReport(Object parent, String file, boolean isEnterpriseServer,
               boolean optimizeMemory, boolean multiPageExp,
               boolean useBackupData);
```

To force the API to use the backup data, only the last argument in the above constructor has to be set to `true`. The values of the other boolean arguments do not matter.

#### 8.1.5.1.4. Parameterized Reports

Reports can also contain parameters to either limit the data in some form or to provide additional information. Typically, query (or IN) parameters are used by the data source to limit the data while parameterized formula are used to include more data within the report.

Query parameters can be both single value and multi-value parameter types while formula parameters are single value only.

When a parameterized template is opened using following constructor:

```
QbReport(Object parent, String templateName);
```

a dialog box appears, asking for the value(s) of the parameter(s). This dialog box is the same as the one that appears in Designer when the report is previewed.

##### 8.1.5.1.4.1. Object Array

A parameterized report can also be opened without the dialog box prompting for any value(s). This can be done by passing in two object arrays, one for the query parameters and another for the formula parameters. Each element in the array represents the value for that particular parameter.

The order of the array must match the order in which the parameters were created in Designer. For correct results, the data type of the value must also match the data type of the parameter.

Query parameters can also be multi-value parameter types. For multi-value parameters, a vector is passed to the object array. The vector contains all the values for the multi-value parameter.

The following example, which can run as an applet or application, passes in the parameter values when opening a template:

```
Component doObjectArray(Object parent) {

    // Do not use ERES Server
    QbReport.setEspressManagerUsed(false);

    // Object array for Query Parameters
    Vector vec = new Vector();
    vec.add("CA");
    vec.add("NY");

    GregorianCalendar beginDate = new GregorianCalendar(2001, 0, 4);
    GregorianCalendar endDate = new GregorianCalendar(2003, 1, 12);

    long beginLong = beginDate.getTimeInMillis();
    long endLong = endDate.getTimeInMillis();

    Date beginDateTime = new Date(beginLong);
    Date endDateTime = new Date(endLong);

    Object queryParams[] = new Object[3];
```

```

queryParams[0] = vec;
queryParams[1] = beginDateTime;
queryParams[2] = endDateTime;

// Object array for Formula Parameter
Object formulaParams[] = new Object[1];
formulaParams[0] = "Sarat";

// Open the template
QbReport report = new QbReport(parent, // container
    "ObjectArray.rpt", // template
    queryParams, // Query Parameters
    formulaParams); // Formula Parameters

// Show report in Viewer
return (new Viewer().getComponent(report));
}

```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/ObjectArrayERES.zip> ]

Exported Results [ <http://data.quadbase.com/Docs70/help/manual/code/export/ObjectArray.pdf> ]

#### 8.1.5.1.4.2. getAllParameters method

In addition to the above, you can also pass in the parameters using the `getAllParameters` method. The `getAllParameters` method returns a list of all parameters in the report (this includes any parameters from the sub-report that are not shared). Each parameter is obtained and the value is then set.

The following example, which can be run as an applet or application, take the same report as above and passes in the parameters using the `getAllParameters` method:

```

Component doGetAllParameters(Object parent) {

    // Do not use EspressoManager
    QbReport.setEspressoManagerUsed(false);

    // Object array for Query Parameters
    Vector vec = new Vector();
    vec.addElement("CA");
    vec.addElement("NY");

    GregorianCalendar beginDate = new GregorianCalendar(2001, 0, 4);
    GregorianCalendar endDate = new GregorianCalendar(2003, 1, 12);

    long beginLong = beginDate.getTimeInMillis();
    long endLong = endDate.getTimeInMillis();

    Date beginDateTime = new Date(beginLong);
    Date endDateTime = new Date(endLong);

    Object queryParams[] = new Object[3];
    queryParams[0] = vec;
    queryParams[1] = beginDateTime;
    queryParams[2] = endDateTime;

    // Object array for Formula Parameter
    Object formulaParams[] = new Object[1];
    formulaParams[0] = "Sarat";

    // Open the template with backup data

```

```

QbReport report = new QbReport(parent, // container
    "ObjectArray.rpt", // template
    false, false, false, true);

// Pass in parameters using getAllParameters
report.getAllParameters().get(0).setValues((Vector) queryParams[0]);
report.getAllParameters().get(1).setValue(queryParams[1]);
report.getAllParameters().get(2).setValue(queryParams[2]);
report.getAllParameters().get(3).setValue(formulaParams[0]);

try {
    // Get the data for the report
    report.refreshWithOriginalData();
} catch (Exception ex)
{
    ex.printStackTrace();
}

// Show report in Viewer
return (new Viewer().getComponent(report));
}

```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/GetAllParameters.zip> ]

Exported Results [ <http://data.quadbase.com/Docs70/help/manual/code/export/GetAllParameters.pdf> ]

The parameter prompt dialogs in Report Designer, Page Viewer, and Menu Page will display the parameters in ordered sequence. To get a list of ordered parameters use the overloaded method with the parameter value set to true.

```
report.getAllParameters(boolean ordered)
```

The results of this method will maintain two qualities. First, formula parameters will always be return first. Second, cascading parameter ordering will be maintained. For more information regarding cascading parameters, please see Section 3.1.3.2.2.3 - Cascading Parameters.

### 8.1.5.1.5. Secure Reports

Information on the report can be changed by passing in a `Security` parameter, when creating the report. The security levels are created in Designer and the appropriate security level is passed, using the API.

The following example, which can be run as an applet or application, opens a secure report that shows sales information for every region except West:

```

Component doSecurity(Object parent) {

    // Do not use EspressoManager
    QbReport.setEspressoManagerUsed(false);

    // Open the template
    QbReport report = new QbReport(parent, // container
        "Security.rpt"); // template

    try {
        // Set Security Level
        report.setSecurityLevel("NoWest");
    } catch (Exception ex)
    {

```

```

    ex.printStackTrace();
}

// Show report in Viewer
return (new Viewer().getComponent(report));
}

```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/Security.zip> ]

Exported Results [ <http://data.quadbase.com/Docs70/help/manual/code/export/Security.pdf> ]

You can also pass in the security level using a `Properties` object. This is especially useful when query/formula parameters are secure.

The following example, which can be run as an applet or application, opens a secure report that shows sales information for every region except West:

```

Component doSecurityProperties(Object parent) {

    // Do not use EspressoManager
    QbReport.setEspressoManagerUsed(false);

    // Set up Properties
    Properties props = new Properties();
    props.put("security level", "NoWest");

    // Open the template
    QbReport report = new QbReport(parent, // container
        "Security.rpt", // template
        props); // properties

    // Show report in Viewer
    return (new Viewer().getComponent(report));
}

```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/SecurityProperties.zip> ]

Exported Results [ <http://data.quadbase.com/Docs70/help/manual/code/export/SecurityProperties.pdf> ]

### 8.1.5.2. Applying a Report Template

When a `QbReport` object is created from scratch (see Appendix 8.B - Creating the Report), the report is created using default attributes. However, you can use a report template (either `.rpt` or `.xml`) to specify user defined attributes during report construction. Almost all the attributes (except for data source and report type) are extracted from the template and applied to the `QbReport` object. The template name usually appears as the last argument in the `QbReport` constructors.

You can also specify the template name using the `applyTemplate(String fileName)` method in the `QbReport` class.

The following example, which can be run as an applet or application, applies a template onto the `QbReport` object:

```

Component doApplyingTemplate(Object parent) {

    // Do not use ERES Server
    QbReport.setEspressoManagerUsed(false);

    String templateLocation = "..";

    // Apply the template

```

```

        QbReport report = new QbReport (parent, // container
        QbReport.SUMMARY, // report type
        data, // data
        columnMapping, // column mapping
        templateLocation); // template

        // Show report in Viewer
        return (new Viewer().getComponent(report));
    }

```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/ApplyingTemplate.zip> ]

Exported Results [ <http://data.quadbase.com/Docs70/help/manual/code/export/ApplyingTemplate.pdf> ]

Please note that the above code is not complete and is there as a guide. However, the link contains a complete application code that can be run.

You can also take the column mapping from the template and have it applied on the `QbReport` object being created. This is done by passing in `null` instead of a `ColumnInfo[]` object.

In addition to the column mapping, you also obtain the database connection information from the template (assuming the template uses a database as the datasource). This is done by passing in `null` for any of the `DBInfo` or `SimpleQueryFileInfo` parameters.

By default, when you apply templates, formulas in the table data section are not applied. To apply formulas and/or scripts from the template `.rpt/.xml` file to the `QbReport` object, you will need to use the following method:

```

QbReport.applyTemplate(String templateName, boolean applyFormula);

```

### 8.1.5.3. Modifying Data Source

You can create report templates in Designer and open those templates using the API. The `QbReport` object created uses the same data source as the template and attempts to fetch the data. However, it may be that while the template has all the look and feel needed, the data source may be an incorrect one. The following sections show how to open the template with backup data and switch the data source, without recreating the entire report.

Please note that for best results, the number of columns and the data type of each column must match between the two data sources (i.e., the one used to create the template in Designer and the new data source).

After switching the data source, the `QbReport` object must be forced to fetch the new data. This can be done by calling the refresh method in the `QbReport` class.

#### 8.1.5.3.1. Data from a Database

Switching the data source to point to a database is simple. All you would need to do is provide the database connection information as well as the query to be used and pass that to the `QbReport` object. You can provide the database connection information (as well as the query) in a `DBInfo` object (for more information on creating a `DBInfo` object, please refer to Appendix 8.A.1 - Data from a Database).

The following example, which can be run as an applet or application, switches the data source of the `QbReport` object to a database:

```

Component doSwitchToDatabase(Object parent) {

    // Do not use EspressoManager
    QbReport.setEspressoManagerUsed(false);

    // Data Source
    DBInfo newDatabaseInfo = new DBInfo(...);

    // Open the template with backup data

```

```

QbReport report = new QbReport(parent, // container
    "SwitchToDatabase.rpt", // template
    false, false, false, true);

try {
    // Switch data source
    report.getInputData().setDatabaseInfo(newDatabaseInfo);

    // Refresh report
    report.refresh();
} catch (Exception ex)
{
    ex.printStackTrace();
}

// Show report in Viewer
return (new Viewer().getComponent(report));
}

```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/SwitchToDatabase.zip> ]

Exported Results [ <http://data.quadbase.com/Docs70/help/manual/code/export/SwitchToDatabase.pdf> ]

Please note that the above code is not complete and is there as a guide. However, the link contains a complete application code that can be run.

The above approach changes the data source for the current report level only (it can be either the main report or the sub-report).

If your report template has any sub-reports, drill-down reports, or independent charts (i.e. charts that do not use the report data as the data source) and you wish to change the data sources for all, then you can use the `setAllDatabaseInfo` method under the `IInputData` interface. Please note that with this approach, a new query cannot be specified and the original query will be used. Only the database connection information will be changed.

The following example, which can be run as an applet or application, uses the `setAllDatabaseInfo` method to switch the data source:

```

Component doSwitchToDatabaseSetAll(Object parent) {

    // Do not use EspressoManager
    QbReport.setEspressoManagerUsed(false);

    // Open the template with backup data
    QbReport report = new QbReport(parent, // container
        "SwitchToDatabaseParam.rpt", // template
        false, false, false, true);

    try {
        // Switch data source
        report.getInputData().setAllDatabaseInfo(...);

        // Refresh report
        report.refresh();
    } catch (Exception ex)
    {
        ex.printStackTrace();
    }

    // Show report in Viewer
    return (new Viewer().getComponent(report));
}

```

---

 }

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/SwitchToDatabaseSetAll.zip> ]

Exported Results [ <http://data.quadbase.com/Docs70/help/manual/code/export/SwitchToDatabaseSetAll.pdf> ]

Please note that the above code is not complete and is there as a guide. However, the link contains a complete application code that can be run.

#### 8.1.5.3.1.1. Parameterized

Just as with a regular query, you can switch the data source to a parameterized query. With a parameterized query, the parameter(s) properties as well as the database connection information and the query must be specified.

The following example, which can be run as an applet or application, switches the data source of the QbReport object to a parameterized query:

```
Component doSwitchToDatabaseParam(Object parent) {

    // Do not use EspressoManager
    QbReport.setEspressoManagerUsed(false);

    // New Database connection and parameter information
    // Parameter information
    SimpleQueryInParam param = new SimpleQueryInParam(...);

    SimpleQueryInParam[] paramSet = { param };

    // Database information
    SimpleQueryFileInfo newDatabaseInfo = new SimpleQueryFileInfo(...);
    newDatabaseInfo.setInParam(paramSet);

    // Open the template with backup data
    QbReport report = new QbReport(parent, // container
        "SwitchToDatabaseParam.rpt", // template
        false, false, false, true);

    try {
        // Switch data source
        report.getInputData().setDatabaseInfo(newDatabaseInfo);

        // Refresh report
        report.refresh();
    } catch (Exception ex)
    {
        ex.printStackTrace();
    }

    // Show report in Viewer
    return (new Viewer()).getComponent(report);
}
```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/SwitchToDatabaseParam.zip> ]

Exported Results [ <http://data.quadbase.com/Docs70/help/manual/code/export/SwitchToDatabaseParam.pdf> ]

Please note that the above code is not complete and is there as a guide. However, the link contains a complete application code that can be run.

Again, just like with regular queries, you can use the `setAllDatabaseInfo` method to switch to the new data source. In this approach, the original query and parameter information is used while the database connection

information is altered. After switching the database information, the parameter value(s) must be specified before refreshing the report.

The following example, which can be run as an applet or application, uses the `setAllDatabaseInfo` method to switch the data source:

```
Component doSwitchToDatabaseParamSetAll(Object parent) {

    // Do not use ERES Server
    QbReport.setEspressManagerUsed(false);

    // Open the template with backup data
    QbReport report = new QbReport(parent, // container
        "SwitchToDatabaseParam.rpt", // template
        false, false, false, true);

    try {
        // Switch data source
        report.getInputData().setAllDatabaseInfo(...);

        // Pass in parameter value
        report.getAllParameters().get(0).setValue("TRD");

        // Refresh the report
        report.refresh();

    } catch (Exception ex)
    {
        ex.printStackTrace();
    }

    // Show report in Viewer
    return (new Viewer().getComponent(report));
}
```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/SwitchToDatabaseParamSetAll.zip> ]

Exported Results [ <http://data.quadbase.com/Docs70/help/manual/code/export/SwitchToDatabaseParamSetAll.pdf> ]

Please note that the above code is not complete and is there as a guide. However, the link contains a complete application code that can be run.

#### 8.1.5.3.1.2. JNDI

You can also change the data source to a JNDI data source. This is done by specifying the JNDI connection information in a `DBInfo` object and then passing it to the `QbReport` object.

The following example, which can be run as an applet or application, switches the data source of the `QbReport` object to a JNDI database:

```
Component doSwitchToDatabaseJNDI(Object parent) {

    // Do not use ERES Server
    QbReport.setEspressManagerUsed(false);

    // Open the template with backup data
    QbReport report = new QbReport(parent, // container
```

```

"SwitchToDatabaseJNDI.rpt", // template
false, false, false, true);

    // New database connection information
    DBInfo newDatabaseInfo = new DBInfo(...);

    try {
        // Switch data source
        report.getInputData().setDatabaseInfo(newDatabaseInfo);

        // Refresh the report
        report.refresh();

    } catch (Exception ex)
    {
        ex.printStackTrace();
    }

    // Show report in Viewer
    return (new Viewer().getComponent(report));
}

```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/SwitchToDatabaseJNDI.zip> ]

Exported Results [ <http://data.quadbase.com/Docs70/help/manual/code/export/SwitchToDatabaseJNDI.pdf> ]

Please note that the above code is not complete and is there as a guide. However, the link contains a complete application code that can be run. Note that for the application code to run, the Woodview database needs to be set up as a JNDI data source in the Tomcat environment and the application code changed to match the connection information.

### 8.1.5.3.2. Data from a Data File (TXT/DAT/XML)

You can switch the data source to a text file as long as the text file follows the Quadbase guidelines (for more details, please refer to Appendix 8.A.2 - Data from a Data File (TXT/DAT/XML)).

The following example, which can be run as an applet or application, switches the data source of the QbReport object to a text file:

```

Component doSwitchToDataFile(Object parent) {

    // Do not use EspressoManager
    QbReport.setEspressoManagerUsed(false);

    // Open the template with backup data
    QbReport report = new QbReport(parent, // container
        "SwitchToDataFile.rpt", // template
        false, false, false, true);

    try {
        // Switch data source
        report.getInputData().setDataFile("sample.dat");

        // Refresh report
        report.refresh();
    } catch (Exception ex)
    {
        ex.printStackTrace();
    }
}

```

```

}

// Show report in Viewer
return (new Viewer().getComponent(report));
}

```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/SwitchToDataFile.zip> ]

Exported Results [ <http://data.quadbase.com/Docs70/help/manual/code/export/SwitchToDataFile.pdf> ]

Please note that the above code is not complete and is there as a guide. However, the link contains a complete application code that can be run.

You can also specify whether the data is sorted (this improves performance) and/or the encoding for the text file. This can be done using the following method in `IInputData`:

```
setDataFile(String dataFile, boolean sortedData, String encoding);
```

### 8.1.5.3.3. Data from an XML Data Source

You can switch the data source to your custom XML data as long as there is a `.dtd` or `.xml` schema accompanying your data. The XML data information is specified (for more details, please refer to Appendix 8.A.3 - Data from an XML Data Source) and then passed to the `QbReport` object.

The following example, which can be run as an applet or application, switches the data source of the `QbReport` object to XML data:

```

Component doSwitchToXMLData(Object parent) {

    // Do not use ERES Server
    QbReport.setEspressManagerUsed(false);

    // Open the template with backup data
    QbReport report = new QbReport(parent, // container
        "SwitchToXMLData.rpt", // template
        false, false, false, true);

    // XML data source information

    XMLFileQueryInfo newData = new XMLFileQueryInfo(...);

    try {
        // Switch data source
        report.getInputData().setXMLFileQueryInfo(newData);

        // Refresh the report
        report.refresh();
    } catch (Exception ex)
    {
        ex.printStackTrace();
    }

    // Show report in Viewer
    return (new Viewer().getComponent(report));
}

```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/SwitchToXMLData.zip> ]

Exported Results [ <http://data.quadbase.com/Docs70/help/manual/code/export/SwitchToXMLData.pdf> ]

Please note that the above code is not complete and is there as a guide. However, the link contains a complete application code that can be run.

You can also specify whether the data is sorted (this improves performance) by using the following method in `IInputData`:

```
setXMLFileQueryInfo(XMLFileQueryInfo xmlInfo, boolean sortedData);
```

#### 8.1.5.3.4. Data from Custom Implementation

In addition to the regular data sources, you can also pass in your own custom data. The custom data is passed to the `QbReport` object using either the `IDataSource` [ <http://data.quadbase.com/Docs70/eres/help/apidocs/quadbase/reportdesigner/util/IDataSource.html> ] or `IParameterizedDataSource` [ <http://data.quadbase.com/Docs70/eres/help/apidocs/quadbase/reportdesigner/util/IParameterizedDataSource.html> ] interfaces (for more details, please refer to Appendix 8.A.5 - Data passed in a Custom Implementation).

The following example, which can be run as an applet or application, switches the data source to a custom implementation:

```
Component doSwitchToCustomData(Object parent) {

    // Do not use ERES Server
    QbReport.setEspressManagerUsed(false);

    // Open the template with backup data
    QbReport report = new QbReport(parent, // container
        "SwitchToCustomData.rpt", // template
        false, false, false, true);

    try {
        // Switch data source
        report.getInputData().setClassFile(...);

        // Refresh the report
        report.refresh();
    } catch (Exception ex)
    {
        ex.printStackTrace();
    }

    // Show report in Viewer
    return (new Viewer().getComponent(report));
}
```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/SwitchToCustomData.zip> ]

Exported Results [ <http://data.quadbase.com/Docs70/help/manual/code/export/SwitchToCustomData.pdf> ]

Please note that the above code is not complete and is there as a guide. However, the link contains a complete application code that can be run.

You can also specify whether the data is sorted (this improves performance) and/or to prompt the parameter dialog by using the following method in `IInputData`:

```
setClassFile(String classname, boolean sortedData, boolean
showPromptDialog);
```

#### 8.1.5.3.4.1. Parameterized

You can have a custom implementation, that requires parameter values, to be the new data source. In this scenario, the custom implementation must use the `IParameterizedDataSource` [ <http://data.quadbase.com/Docs70/eres/help/apidocs/quadbase/reportdesigner/util/IParameterizedDataSource.html> ] interface (for more details, please refer to Appendix 8.A.5 - Data passed in a Custom Implementation). After switching the data source information, the parameter value(s) must be specified before refreshing the report.

The following example, which can be run as an applet or application, switches the data source to a parameterized custom implementation:

```
Component doSwitchToCustomDataParam(Object parent) {

    // Do not use ERES Server
    QbReport.setEspressManagerUsed(false);

    // Open the template with backup data
    QbReport report = new QbReport(parent, // container
    "SwitchToCustomDataParam.rpt", // template
    false, false, false, true);

    try {
        // Switch data source
        report.getInputData().setClassFile(..., false, false);

        // Pass in parameter value
        report.getAllParameters().get(0).setValue(...);

        // Refresh the report
        report.refresh();

    } catch (Exception ex)
    {
        ex.printStackTrace();
    }

    // Show report in Viewer
    return (new Viewer().getComponent(report));

}
```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/SwitchToCustomDataParam.zip> ]

Exported Results [ <http://data.quadbase.com/Docs70/help/manual/code/export/SwitchToCustomDataParam.pdf> ]

Please note that the above code is not complete and is there as a guide. However, the link contains a complete application code that can be run.

#### 8.1.5.3.5. Data Passed in an Array in Memory

You can also pass in data using arrays. The array data is usually stored in memory and passed to the `QbReport` object (for more details, please refer to Appendix 8.A.4 - Data passed in an Array in Memory).

The following example, which can be run as an applet or application, switches the data source to an array in memory:

```
Component doSwitchToArrayData(Object parent) {
```

```

// Do not use ERES Server
QbReport.setEspressManagerUsed(false);

// Open the template with backup data
QbReport report = new QbReport(parent, // container
    "SwitchToArrayData.rpt", // template
    false, false, false, true);

// Create array data
DbData newData = new DbData(...);

try {
    // Switch data source
    report.getInputData().setData(newData);

    // Refresh the report
    report.refresh();

} catch (Exception ex)
{
    ex.printStackTrace();
}

// Show report in Viewer
return (new Viewer().getComponent(report));
}

```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/SwitchToArrayData.zip> ]

Exported Results [ <http://data.quadbase.com/Docs70/help/manual/code/export/SwitchToArrayData.pdf> ]

Please note that the above code is not complete and is there as a guide. However, the link contains a complete application code that can be run.

You can also specify whether the data is sorted (this improves performance) by using the following method in `IInputData`:

```
setData(IResultSet rs, boolean sortedData);
```

#### 8.1.5.3.6. Drill-Down with DrillDownReportServlet

When changing the data source for a drill-down report, you use the `setAllDatabaseInfo` method in `IInputData`. However, if you are exporting the report and using the `DrillDownReportServlet` (for more details, please see Section 8.1.5.7.10.2 - `DrillDownReportServlet`), you need to get a handle to the session and use the method `setDrillDownDatabaseInfo` in `QbReport` so that the drill-down layers will know what the new data source is.

The following example, which can be run as an applet or application, switches the data source of a drill-down report:

```

public void doGet(HttpServletRequest req, HttpServletResponse res) throws
    ServletException,
    IOException
{
    HttpSession session = req.getSession(true);

```

```

// Set the "content type" header of the response
res.setContentType("text/html");

// Get the response's OutputStream to return content to the client.
OutputStream toClient = res.getOutputStream();

try
{
    // Do not use EspressoManager
    QbReport.setEspressoManagerUsed(false);

    // Open report with backup data (data source will be switched later)
    QbReport report = new QbReport(null, "SwitchDrillDownServlet.pak",
        false, false, false, true);

    // New database connection information
    String newDatabaseURL = "jdbc:hsqldb:woodview";
    String newDatabaseDriver = "org.hsqldb.jdbcDriver";
    String newDatabaseUID = "sa";
    String newDatabasePassword = "";

    // Switch data source
    report.getInputData().setAllDatabaseInfo(newDatabaseURL,
newDatabaseDriver,
        newDatabaseUID, newDatabasePassword);

    // Put new data source information in session for DrillDownReportServlet
    report.setDrillDownDatabaseInfo(session, newDatabaseURL,
newDatabaseDriver,
        newDatabaseUID, newDatabasePassword);

    report.setDynamicExport(true, "localhost", 8080);

    ByteArrayOutputStream tempStream = new ByteArrayOutputStream();

    report.refresh();

    // Export the report to DHTML
    report.export(QbReport.DHTML, tempStream);

    tempStream.writeTo(toClient);
} catch (Exception e) {
    e.printStackTrace();
}

// Flush the outputStream
toClient.flush();

// Close the writer; the response is done.
toClient.close();
}

```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/SwitchDrillDownServlet.zip> ]

Exported Results Root [ <http://data.quadbase.com/Docs70/help/manual/code/export/SwitchDrillDownServlet-Root.html> ]

Exported Results Drill-Down Level [ <http://data.quadbase.com/Docs70/help/manual/code/export/SwitchDrill-DownServletDrill.html> ]

Please note that the above code is not complete and is there as a guide. However, the link contains a complete application code that can be run. You would have to change the location of the template and deploy the servlet in order to run the code successfully.

You can also specify a connection object for the database by using the following method in `QbReport`

```
setDrillDownConnection(Object session, Connection conn);
```

### 8.1.5.4. Modifying Column Mapping

Just as the data source of a report can be changed, the column mapping can be modified using the API as well. However, this is not recommended as the report may have formulas and/or scripts that are data dependent. If the number of columns and/or the data of the columns do not match the original mapping, certain formulas and/or scripts may not work. While this section shows how to switch the mapping, it is recommended that in such a scenario a new `QbReport` object be created (see Appendix 8.B - Creating the Report) and a template applied on it.

The following example, which can be run as an applet or application, modifies the column mapping of the report:

```
Component doModifyColumnMapping(Object parent) {

    // Do not use EspressoManager
    QbReport.setEspressoManagerUsed(false);

    // Open the template with backup data
    QbReport report = new QbReport(parent, // container
        "ModifyColumnMapping.rpt", // template
        false, false, false, true);

    ColInfo[] newColumnMapping = new ColInfo[...];

    try {
        // Switch mapping
        report.getInputData().setMapping(newColumnMapping);

        // Refresh report
        report.refreshWithOriginalData();

        // Reapply template
        report.applyTemplate("ModifyColumnMapping.rpt");
    } catch (Exception ex)
    {
        ex.printStackTrace();
    }

    // Show report in Viewer
    return (new Viewer().getComponent(report));
}
```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/ModifyColumnMapping.zip> ]

Exported Results [ <http://data.quadbase.com/Docs70/help/manual/code/export/ModifyColumnMapping.pdf> ]

Please note that the above code is not complete and is there as a guide. However, the link contains a complete application code that can be run.

### 8.1.5.5. Report Components

An ERES report is made of different segments. Each segment can be set and modified independent from each other. Listed below are the various parts of the report :

### 8.1.5.5.1. ReportElement

A `ReportElement` object forms the core of objects in Report API. A `ReportElement` object provides a way to manipulate the contents as well as the look and feel of the individual elements. All the other report objects (such as `ReportCell`, `ReportSection`, `ReportColumn` etc.) extend the `ReportElement` class. Every part of the report is comprised of a `ReportElement` object, or an extension of it. Within each `ReportElement` object, properties such as font, color, data format, etc can be manipulated. You can also copy `ReportElement` objects or just apply the template of one `ReportElement` object to another. When a template is applied, the data of the object does not change. Only the look and feel of the `ReportElement` object is modified.

When modifying, you do not get a handle to the `ReportElement` but rather use the methods of the subclass (`ReportCell`, `ReportSection`, `ReportColumn` etc.)

### 8.1.5.5.2. ReportCell

A `ReportCell` object is used to insert labels, formulas, text, charts or even images into different sections of the report (i.e. in the `ReportSection(s)` of the report). You can also use `ReportCell` objects to help with the formatting of the cells in the various parts of the report.

You can get a handle to a cell by using its numeric index (i.e. the position of the cell in the section), its ID (which is a string) or its custom ID (assuming the cell has one). You can use the following methods, in `ReportSection`, to get the desired cell:

```
getData(int i);
getData(String ID);
```

You can also get all the report cells, in a given section, by using the following method in `ReportSection`:

```
getData();
```

### 8.1.5.5.3. ReportSection

A report is divided into many sections. The following shows the different sections of a report:

<b>Report Header:</b>	This section is like the title of the report. It only appears once at the top of the report.
<b>Page Header:</b>	This section serves as a header to the page. It appears at the top of every page of the report.
<b>Table Header:</b>	This section serves as a header to the detail or data section of the report. By default it only appears once in the report.
<b>Group Header:</b>	This section appears in reports with grouped data (i.e. Master & Details report), or data with row breaks inserted (i.e. Summary Break report). It repeats at the top of each grouping within the report.
<b>Table Data:</b>	This is the main section of the report that contains most of the data. Data columns that have been selected for the report are placed in this section and repeated for each entry in the column.
<b>Group Footer:</b>	This section appears on reports with grouped data (i.e. Master & Details report) or data with row breaks inserted (i.e. Summary Break report). It repeats at the bottom of each grouping within the report.
<b>Table Footer:</b>	This section serves as the footer to the detail or data section of the report. By default it appears only once in the report.
<b>Page Footer:</b>	This section serves as a footer to the page. It appears at the bottom of every page of the report.
<b>Report Footer:</b>	This is the last summary or footer section of the report. It only appears once at the end of the report.

---

You can get a handle to the Page Footer, Page Header, Report Footer, Report Header, and Report Table sections using the following methods in `QbReport`:

```
getPageFooter();
getPageHeader();
getReportFooter();
getReportHeader();
getTable();
```

and you can get a handle to the Group Footer, Group Header, Table Header, and Table Footer sections using the following methods in `ReportTable`:

```
getRowBreakFooter(int breakLevel);
getRowBreakHeader(int breakLevel);
getFooter();
getHeader();
```

#### 8.1.5.5.4. ReportColumn

A `ReportColumn` object is an array of `ReportCell` objects that appears in the column selected. Using `ReportColumn` objects, formatting can be done through the entire column once, instead of applying the formatting one cell at a time.

You can get a handle to a `ReportColumn` by using the following method in `ReportTable`:

```
getColumn(int index);
```

#### 8.1.5.5.5. ReportTable

This is the main part of the report. This is the section that contains the data columns that have been selected for the report. This section contains the data that is used for the Group Footers and for the various summaries.

You can get a handle to the `ReportTable` section by using the following method in `QbReport`:

```
getTable();
```

#### 8.1.5.5.6. ReportImage

`ReportImage` objects are used to add images to the report. The formats supported by ERES are GIF, JPEG, and PNG. The `ReportImage` object can be considered as a cell that contains a image. Care is to be taken to define the dimensions of the cell so that the image is clearly visible and is not truncated.

You can get a handle to all the `ReportImage` objects by using the following method in `QbReport`:

```
getReportImages();
```

#### 8.1.5.5.7. ReportChartObject

`ReportChartObjects` are used to add charts of type TPL or CHT (Quadbases's proprietary formats) to the report. Adding in the `ReportChartObject` is different from adding in a `ReportImage` object. The location of the TPL or template file is given here. ERES then takes the template and creates a chart using the data from the relevant section. Thus, using the same template, you can have different charts at different points of the report, all of which share the same look and feel even though the data might be different.

---

The chart template files are first created in Report Designer. Thus, the mapping of the chart is based on the mapping of the data in the report. We recommend using chart templates from the same type of report containing similar data mapping (i.e. the same kind of data and data type) as the report being generated using the API.

`ReportChartObjects` can also be used to add stand-alone charts (i.e. charts whose data is not from the report). Stand-alone charts are created from the API and the data source for the chart can be independent from the report.

Again, a `ReportChartObject` object can be considered as a cell that contains a chart. Similar care should be taken to define the dimensions of the cell so that the chart is clearly visible and not truncated.

You can get a handle to all the `ReportChart` objects by using the following method in `QbReport`:

```
getReportChartObjects();
```

#### 8.1.5.5.8. ChartObject

`ChartObjects` are intermediate objects created before adding a chart, created completely through the API, to the report. The chart obtains its data from the report and the mapping for the chart is based on the columns of the report. These charts are different from stand-alone charts as the data for these charts is the report itself whereas the data for stand-alone charts can be independent from the report. Other chart properties such as dimension, chart type and mapping are also specified and the `ChartObject` created.

The `ChartObject` is then added to a `ReportChartObject`. This `ReportChartObject` is then added to the report.

For more details on how to use `ChartObject`, please refer to the Section 4.2.1 - Introduction to Chart DesignerCharting Guide.

#### 8.1.5.5.9. ReportDocument

`ReportDocument` is used to represent a clob (character large object) when adding a clob to the report. Depending on the length of the clob, it is either stored as a string in the `ReportDocument` or is stored in a file and the filename is stored in the `ReportDocument`.

Again, a `ReportDocument` object can be considered as a cell that contains a clob. Similar care should be taken to define the dimensions of the cell so that the content is clearly visible and not truncated.

We recommend that the `ReportDocument` object not be modified using the API and that all changes be done in the template.

#### 8.1.5.5.10. ReportRTFObject

`ReportRTFObject` is used to add content, from a RTF file, to the report. The file content is streamed to a `ReportRTFObject` and this object is later added to the report.

Again, a `ReportRTFObject` object can be considered as a cell that contains RTF content. Similar care should be taken to define the dimensions of the cell so that the content is clearly visible and not truncated.

We recommend that the `ReportRTFObject` object not be modified using the API and that all changes be done in the template.

#### 8.1.5.5.11. SubReportObject

`SubReportObject` is used to add subreports to the report. The subreport content can either be from a file or created completely from the API.

Again, a `SubReportObject` object can be considered as a cell that contains a subreport. Similar care should be taken to define the dimensions of the cell so that the content is clearly visible and not truncated.

You can get a handle to all the `SubReportObject` objects by using the following method in `QbReport`:

```
getSubReports();
```

You can also get a handle to the subreport from the `SubReportObject` by using the following method in `SubReportObject`:

---

```
getSubReport(IReport qbReport);
```

The `SubReport` object obtained can then be casted to `QbReport`.

#### 8.1.5.5.12. ReportGrid

`ReportGrid` objects are used to draw grids within different sections of the table. The grid will encompass each row and each column (and each cell in the section, if applicable). Currently, the line styles available are dash, double, and solid line styles.

We recommend that the `ReportGrid` object not be modified using the API and that all changes be done in the template.

#### 8.1.5.5.13. ReportLine

`ReportLine` objects are used to draw a line, or lines, in different sections of a table. Lines can be inserted in any part of a report. As with `ReportGrid` objects, `ReportLine` objects have a choice of three styles: dash, double, and solid line styles.

We recommend that the `ReportLine` object not be modified using the API and that all changes be done in the template.

### 8.1.5.6. Modifying Report Attributes

ERES has hundreds of properties, which provide a fine control over the various elements of a report. As a developer, you can customize the look and feel of a report dynamically at run-time. In order to facilitate ease-of-use, most properties have been categorized into groups and exposed in the form of interfaces. An application first obtains a handle to a group interface using a `getXXX` method and then manipulates the report's properties directly by calling methods on that interface. Most interfaces are contained in the package `quadbase.reportdesigner.util`.

#### 8.1.5.6.1. Adding New Cells

ERES allows you to create new cells and position them where you want them in the report.

To add a cell to the report, you will need to create a `ReportCell` object (though it doesn't have to be a `ReportCell` object. It can also be a `ReportImage` object, a `ReportChartObject`, a `ReportLine` object or a `ReportGrid` object depending on the type of information you are adding). After creating and specifying the `ReportCell` properties, you can add it to any part of the report.

In the following example, a cell is added as the Page Footer:

```
ReportCell cell = new ReportCell("Inventory Report");
cell.setWidth(7);
cell.setAlign(IAAlignConstants.ALIGN_RIGHT);
report.getPageFooter().addData(cell);
```

You can also apply another cell's look and feel onto the newly created `ReportCell` object by using the following method in `ReportCell`:

```
applyTemplate(ReportCell templateCell, boolean applyScript);
```

#### 8.1.5.6.2. Adding Images/Charts

Both images and charts can be included in practically any part of the report. Images of type GIF, JPEG, PNG, and charts of type TPL (Quadbases's proprietary format) are supported by ERES. Adding charts/images to a report generally allows for better presentation and imparts more information and makes the report easier to read and understand.

To add an image, you will need to define a `ReportImage` object. Within the `ReportImage` object, you will have to pass in the URL for the image, either as an `http://` or a `file://` url. It is recommended that you use an `http://` url so that if the report is exported to a DHTML format, the DHTML report picks up the image consistently. You can also specify the image type, dimensions, and alignment of the `ReportImage` object and then add it to the location desired in the report.

In the following example, an image called logo3 of type gif is added to the Page Header:

```
ReportImage reportImage = new ReportImage();
try {

    reportImage.setImageType(IExportConstants.GIF);
    java.net.URL imageUrl =
        new java.net.URL ("http://someMachineName/Gifs/logo3.gif");
    reportImage.setImageURL(imageUrl);
    reportImage.setWidth(7);
    reportImage.setHeight(1);
    reportImage.setAlign(IAlignConstants.ALIGN_LEFT);

} catch (Exception ex) {

    ex.printStackTrace();

}

report.getPageHeader().addData(reportImage);
```

You can also specify a relative link to be used instead of the complete URL specified in the code (or in Report Designer). This can be done by passing in the relative path in the following method in ReportImage:

```
public void setImagePath(String path);
```

For example, passing `../..../logo3.gif` in the above method would result in the DHTML export having a relative link in the `<img src>` tag rather than the absolute URL.

To add a chart, please refer to Appendix 8.D - Creating the Chart.

### 8.1.5.6.3. Adding Hyperlinks

You can also add hyperlinks to a cell or several cells in the report and have different hyperlinks for different cells. Hints to the link and targets can be set up in the cell and the link is then preserved when the report is exported. Note that links to PAK files can also be inserted in a report. However, the link to such files will not work outside of an applet environment. To view RPT files in DHTML format, we recommend you to create separate DHTML files and link to those files instead.

In the following example, a cell is created which contains a link to a page and is added to the Page Header:

```
ReportCell cell = new ReportCell("ABC Incorporated");
cell.setLink(new String("http://www.quadbase.com"));
cell.setHint(new String("Click here to go to Quadbase's Homepage"));
cell.setTarget(new String("ABC Home Page"));
report.getPageHeader().addData(cell);
```

### 8.1.5.6.4. Adding GridLines and Lines

While you can add a ReportGrid and ReportLine to a report using the API, we recommend that all grids and lines be inserted into a report using Designer. To add grid lines to a report, you must define a ReportGrid object. Within the ReportGrid object, you can define various properties such as color, line style, thickness, width etc before specifying a section to be encompassed by the grid.

In the following example, a blue grid line is added around the Report Table:

```
ReportGrid reportGrid = new ReportGrid();
reportGrid.setBorderColor(Color.blue);
reportGrid.setGridStyle(ReportGrid.DOUBLE);
reportGrid.setBorder(1);
reportGrid.setWidth(7);
report.getTable().addImage(reportGrid);
```

To add lines to the report, you must define a `ReportLine` object. Here, you can also define its various properties such as color, line style, thickness, width, etc before specifying the section it is added to.

In the following example, a line is added to the Page Header:

```
ReportLine reportLine = new ReportLine(false);
reportLine.setBorderColor(Color.red);
reportLine.setLineStyle(ReportLine.DOUBLE);
reportLine.setBorder(1);
reportLine.setWidth(7);
report.getPageHeader().addData(reportLine);
```

### 8.1.5.6.5. Adding Rich Text Field Objects

To add rich text fields to a report, you must define a `ReportRTFObject` object. Within the `ReportRTFObject` object, you can specify the location of the `.rtf` file and then specify the other properties of the cell before specifying a section to add the object to. Please note that a `.rtf` file must exist in order to add a `ReportRTFObject` to the report.

However, we recommend that you do not include RTF content using this approach but rather add it to the template in the Designer.

In the following example, the content of the file `RText1.rtf` is added to the table data section:

```
ByteArrayOutputStream fileOne = new ByteArrayOutputStream();
FileInputStream fileIn = new FileInputStream("RText1.rtf");

int b = fileIn.read();
while (b != -1) {

    fileOne.write(b);
    b = fileIn.read();

}

ReportRTFObject rtfObjectOne = new ReportRTFObject(fileOne.toByteArray());
rtfObjectOne.setWidth(1);
rtfObjectOne.setHeight(.3);
rtfObjectOne.setResizeToFitContent(true);
rtfObjectOne.setX(report.getTable().getColumn(0).getWidth() +
    report.getTable().getColumn(1).getWidth() +
    report.getTable().getColumn(2).getWidth()); // Set Object at the
end of table data

report.getTable().addRTFObject(rtfObjectOne);
// report is an object of type QbReport
```

The RTF file content can also be displayed in multiple column format. Thus, instead of the default column count of one, you can show the content in two or more columns. The column count and the spacing between the columns can be set using the following methods:

```
ReportRTFObject.setColumnCount(int columnC);
ReportRTFObject.setColumnSpacing(double space);
```

For example, in the above example, if the following lines of code were added:

```
rtfObjectOne.setColumnCount(3);
rtfObjectOne.setColumnSpacing(.3);
```

before adding the `ReportRTFObject` to the report, the report would now show the RTF portion in a three column format with .3 inches separating two adjacent columns.

For more details on rich text field content, please refer to Section 4.1.3.7.7 - Rich Text Fields in Section 4.1.3 - The Designer Interface.

### 8.1.5.6.6. Adding Nested Sections

To add in a nested section (or add nested sections) in a report, you must get a handle to the parent section and created a child section. Nested sections are extremely useful when using in conjunction with any cell or object that need resizing without the overlapping any objects below. Please note that nested sections inherit most of the section options from their parent sections except the page-breaking option.

In the following example, two nested sections are being created within the table header section and a `ReportCell` object (`reportCell`) is being added to one of the sections:

```
ReportSection tableHeader = report.getTable().getHeader();
tableHeader.addSection();
tableHeader.insertSection(0);
tableHeader.getSection(0).setHeight(.5);
tableHeader.getSection(1).setHeight(.5);
tableHeader.getSection(0).addData(new ReportCell("Test Cell"));
```

The index used for nested sections follows a vector index. Please note that a section will not appear even if the height is set, if there are no cells within the section.

### 8.1.5.6.7. Modifying Background Color, Font, Etc

Properties for any section of the report can be modified by getting the appropriate handle and calling the methods. Since all parts of the report extends `ReportElement`, almost all the properties (which can be applied to a single `ReportElement` object) are here and can be applied one at a time, onto a group, or in groups.

For instance, the code below will set the background color of the `ReportCell` object to red:

```
ReportCell cell;
cell.setBgColor(Color.red);
```

The code fragment below would set the background color of the first column of the report to be blue:

```
ReportTable table = report.getTable();
table.getColumn(0).setBgColor(Color.blue);
```

To set all the columns of the table to blue, you need to run through a for-loop, such as one given below:

```
for (int i = 0; i < table.getColumnCount(); i++)
```

```
table.getColumn(i).setBgColor(Color.blue);
```

Note that writing the following code

```
table.setBgColor(Color.blue);
```

does NOT set the columns in the table to have the color blue. It merely sets the color of the table section (i.e. the area behind the cells in the column) to be blue.

Similarly, the code below:

```
ReportCell cell;
cell.setFont(new Font("Arial", Font.BOLD, 14));
table.getColumn(0).setFont(new Font(table.getFont().getName(), Font.BOLD,
    16));
```

sets the font for the cell and for the first column of the report. Note that setting the font for any part of the report other than a `ReportCell` or a `ReportColumn` object (such as a `ReportTable` or a `ReportSection`) involves setting the font for each individual cell in the section. You can create your own method, which sets the font for any section of the report. For instance, the code below:

```
void setFont(Font font, ReportElement elts[]) {
    if (elts == null) return;
    for (int i = 0; i < elts.length; i++) {
        elts[i].setFont(font);
    }
    setFont(new Font(table.getFont().getName(), Font.BOLD, 16),
        rowBreakHeaderZero.getData());
    setFont(new Font(table.getFont().getName(), Font.BOLD, 18),
        tableHeader.getData());
}
```

creates your own method and uses it to set the font for the Row Break Zero Header and for the Table Header.

If you are planning to export your report in PDF format, True Type Fonts can also be mapped and used with the following method:

```
report.setFontMapping(String fontName, int style, String ttf);
```

where `fontName` is the name of the font, `style` is a `QbReport` FONT constant, and `ttf` is the path and filename to the installed `.ttf` font. For example:

```
report.setFontMapping("Dialog", QbReport.BOLDITALIC, "C:/ERES/help/examples/
fonts/bookosbi.ttf");
```

For a more detailed description of PDF Font Mapping, please see Section 4.1.5.2.1 - PDF Font Mapping.

Similarly, other properties can be set using the methods in the interfaces.

### 8.1.5.6.8. Modifying the Format

You can also change the format of any data (be it Numeric, boolean, or String) to conform to your requirements. The format can be set for individual `ReportCell` objects or for `ReportColumn` objects.

For example, the code below:

```
NumericFormat contentFormat = new NumericFormat();
contentFormat.decimal = 2;
contentFormat.currencySymbol = '$';

for (int i = table.getStartOfColumnBreakColumn(); i <
    table.getColumnCount(); i++ )

    table.getColumn(i).setDataFormat(contentFormat);
```

sets the format of the numeric data of the Column Break columns, in the Cross Tab report. It adds a '\$' symbol and sets the cells to show two decimal places.

Similarly, the properties for string or boolean data can also be modified to fit your requirements.

### 8.1.5.6.9. Modifying a Column to Show Bar Codes

You can also represent the data (string and numeric) in a column as bar codes. This is helpful as most data sources do not have the capacity to store bar codes. As such, only the information imprinted on the bar codes is saved, either as string or numeric data. This data can be added to the report as a column and then the data format modified to show bar codes. The format can be set for individual `ReportCell` objects or for `ReportColumn` objects. The bar code symbologies supported are Code 39, UPC A, EAN 13, Interleaved 2 of 5, and Codabar.

For example, the code below:

```
BarcodeFormat barCode = new BarcodeFormat(BarcodeFormat.UPCA);
table.getColumn(0).setDataFormat(barCode);
```

encodes the data in Column 0 as bar codes using the UPC A symbology.

### 8.1.5.6.10. Modifying a Report to be a Top N Report

You can modify the report to show a set of the highest values or the lowest values within the group. This is helpful when you want to show a report with the highest revenues or the lowest incidences of errors.

The number of records as well as the ordering is specified using the following method:

```
QbReport.createTopNReport(int colIndex, int topN, boolean ascending);
```

The `colIndex` argument refers to the column in the report, the `topN` argument specifies the number of records and the `ascending` argument specifies whether to show the highest or the lowest records.

For example, the code below:

```
report.createTopNReport(3, 20, true);
```

takes the highest 20 records within Column 3 (depending on the grouping of the report) and displays them in order.

Note that only Columnar, Summary Break, and Master & Details report can be modified to a TopN report.

### 8.1.5.6.11. Cell Scripts

You can also develop your custom cell scripts. These scripts can be assigned to the cells or columns and can be run when certain conditions or requirements are met. These scripts can change the format of the cell(s) so that it looks

different from its surrounding cells. Please note that cell scripting using the API is different from cell scripting in Report Designer.

The following example, which can be run as an application or as an applet, sets up a script that changes the font of any data that is less than 0:

```
public class myScript implements ICellScript {

    // Format the cells according to the specified parameters
    public ReportCell formatCell(int rowIndex, ReportCell cell, Object
originalData, IFormat dataFormat) throws Exception {
        if (originalData instanceof Double)
            if (((Double)originalData).intValue() < 0)
                cell.setFontColor(java.awt.Color.red);

        return cell;
    }
}
```

You can use the following method in ReportColumn to apply the script to a particular column:

```
setCellScript(ICellScript script);
```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/CellScript.zip> ]

Exported Results [ <http://data.quadbase.com/Docs70/help/manual/code/export/CellScript.pdf> ]

Note that you can have different scripts on different columns although a column can have only one script.

### 8.1.5.7. Exporting the Report

ERES API has the capability to export reports in a variety of formats. These include PDF, DHTML, TXT, and CSV formats etc. In addition, any charts included in the report can be exported into JPEG, GIF, and PNG formats as well; although, by default, the charts in the report are exported as JPEG's. The format for the chart can be changed, when creating the ReportChartObject object, by specifying the image type (use the method setImageType(int)).

A report may also be exported to the proprietary PAK format. An PAK file stores all information except actual data. The PAK file can then be used to construct a report object. For an PAK file, the data is automatically loaded from the original data source at the time the report object is constructed. (Note PAK files can be directly viewed using a Report Viewer applet.)

To export the report, use the following method

```
public export(int format, String filename)
```

In the above method, format is one of the format constants listed below and filename is the output filename (with or without an extension).

The following list shows the format constants available for exporting the report components

<b>QbReport.CSV</b>	Comma delimited text file
<b>QbReport.TXT</b>	Delimited text file
<b>QbReport.DHTML</b>	Dynamic Hyper Text Markup Language (DHTML)
<b>QbReport.HTML</b>	Hyper Text Markup Language (HTML)

---

<b>QbReport.PDF</b>	Portable Document Format, with password protection option (PDF)
<b>QbReport.PAK</b>	Report Pack Format (PAK)
<b>QbReport.XML_DATA_AND_FORMAT</b>	Extensible Markup Language, Data and Report Template (XML)
<b>QbReport.XML_PURE_DATA</b>	Extensible Markup Language, Data Only (XML)
<b>QbReport.XML_TEMPLATE</b>	Extensible Markup Language, Report Template Only (XML)
<b>QbReport.EXCEL</b>	Excel Format (XLS)
<b>QbReport.EXCEL_OOXML</b>	Excel 2007 Format (XLSX)
<b>QbReport.RTF</b>	Rich Text Format (RTF)
<b>QbReport.VIEW</b>	View File (VIEW)

In addition, any charts included in the report can be exported to one of the following image formats (format given with its corresponding constant):

<b>QbReport.GIF</b>	GIF
<b>QbReport.JPEG</b>	JPEG
<b>QbReport.PNG</b>	PNG

The following code, which can be run as an applet or application, shows how to construct and export a report:

```
Component doExportReport(Object parent) {
    QbReport.setEspressManagerUsed(false);
    ColInfo colInfo[] = new ColInfo[4];
    for (int i = 0; i < colInfo.length; i++) {
        // Map data column to the Report column
        colInfo[i] = new ColInfo(i);
    }

    QbReport report = null;
    try {
        report = new QbReport
            (parent, // applet
             "ExportReport.rpt"); // template

        ReportChartObject chartObject = new ReportChartObject();
        String chartLocation0 = new String("ExportReport0.tpl");
        chartObject.setText(chartLocation0);
        chartObject.setWidth(7);
        chartObject.setHeight(3);
        chartObject.setImageType(QbReport.JPEG);

        report.getReportFooter().addData(chartObject);

        report.export(QbReport.DHTML, "ExportReport");

    } catch (Exception ex) {
        System.out.println("Cannot create the report");
        ex.printStackTrace();
    }

    return (new Viewer().getComponent(report));
}
```

---

```
}
```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/ExportReport.zip> ]

Exported Results [ <http://data.quadbase.com/Docs70/help/manual/code/export/ExportReport.html> ]

Please note that when you export the report to a text file, the default delimiter is a tab space. However, you can set another delimiter (available delimiters are “;”, “;” and “ ”) by using the following code:

```
report.setExportDelimiter(IDelimiterConstants.COMMA);    // where report is
    an Object of type QbReport
```

When exporting the report to an Excel or Excel 2007 file, you can also set a method so that each numeric value only occupies one cell. This may be helpful if the end user intends to use Excel functions on the exported report. To use this feature, set the following method to `true` before exporting the report.

```
QbReport.setExcelExportFitCell(boolean b)
```

You can also export the `QbReport` object to a byte array using the method

```
public byte[] exportReportToByteArray();
```

This will give the `.pak` equivalent in the form of a byte array. This is useful if you need to serialize the `QbReport` object. Please note that you will still need to specify the directories for any Sub-Reports, Drill-Down and/or charts (if you are not using ERES Server) when recreating the `QbReport` object from the byte array.

### 8.1.5.7.1. Multiple Page Exporting

You can also export the report into several “pages”, instead of a single “page” i.e., the report can be exported to various files and then shown piecemeal. This option is only available for DHTML and XML (Data + Format) export.

The following code is used to export the report into multiple files:

```
report.setExportToMultiPages(true);
```

When using this option, several files are generated (the number of files generated is equal to the number of pages in the report). The first file has the same name as the filename specified in the export method. The subsequent files have the corresponding page number attached at the end of the filename, with the file containing the last page having `LAST` in the filename instead of the page number. For example, if a three page report is exported to `SalesSummary`. The resulting files that are created are `SalesSummary.html` (contains the first page), `SalesSummary_2.html` (contains the second page) and `SalesSummary_LAST.html` (contains the last page).

You can also export a specific page at a time (instead of the complete report) and export the page to a specific file. This is done using the method:

```
export(int format, OutputStream out, int pageNumber);
```

For example, to export page three of a report into the html file `ThirdPage`, the following code is used:

```
FileOutputStream dout3 = new FileOutputStream("ThirdPage.html");
report.export(QbReport.HTML, dout3, 3);
dout3.close();
```

Note that you can pass in `-1` as the argument for the `pageNumber` to export the last page. You also do not need to set `setExportToMultiPages` to `true` in order to export a specific page.

### 8.1.5.7.2. PDF Exporting Options

In certain constructors of the `QbReport` class, the parameters `userPass`, `ownerPass`, and `permissions` are available. These parameters are used to set the user password, owner password, and permissions for the user of the PDF document, respectively. All permissions will be granted to the owner of the PDF document (who uses the owner password to open the PDF document) while only the specified permissions in the `permissions` argument will be available to the user (who uses the user password to open the PDF document). Please refer to the ERES Java API Documentation for more detail about exporting to PDF with options.

The export method that sets PDF options is listed here for the reader's convenience.

```
public void export(int format,
                 java.io.OutputStream out,
                 int exportPage,
                 java.lang.String userPass,
                 java.lang.String ownerPass,
                 int permissions);
```

Additionally, you can also pass embed java scripts in the PDF export so that when the PDF content is streamed to a client browser, the java script is run. The following method allows java script to be embedded to a PDF content

```
public void export(int format,
                 java.io.OutputStream out, // Or java.lang.String
                 specifying the file Name
                 java.lang.String userPass,
                 java.lang.String ownerPass,
                 int permissions,
                 java.lang.String javaScript);
```

For example, the following method sets the PDF to automatically print when viewed in a client browser.

```
QbReport report = new QbReport(.....);
.....
.....
String javaScript = "this.print(true);\r";
report.export(QbReport.PDF, someOutputStream, null, null, QbReport.AllowAll,
             javaScript);
```

The above call would export the report as PDF content (with all permissions) to a stream (`someOutputStream`) passing data to a client browser. When the PDF is loaded on the client browser, the javascript is automatically run (which in this case, tells the browser to print the content).

### 8.1.5.7.3. Exporting DHTML Content with Style Sheets

You can specify a style sheet to be used while exporting the report to DHTML or HTML format. You can specify an internal style sheet or an external style sheet before exporting the report.

To export the report using an internal style sheet, you would use the following method in `QbReport`:

```
public void setUseStyleSheet(boolean state);
```

To export the report using an external style sheet, you first need to specify the style to be used for the specified `ReportElement` object using the following method in `ReportElement`:

```
public void setStyleName(String newStyleName);
```

You then specify the name of the external style sheet file using the following method in QbReport:

```
public void setExternalStyleSheetName(String css);
```

For example, the following code:

```
report.getTable().getColumn(0).setStyleName("style_1"); // where report is
  an object of type QbReport
report.setExternalStyleSheetName("http://someMachine/someDirectory/
styles.css");
```

will set the first column of the report to use `style_1` from the specified style sheet file (in this case, it is `http://someMachine/someDirectory/styles.css`).

#### 8.1.5.7.4. HTML Block Export

You can export reports as a block of HTML code rather than a complete HTML file. This allows for custom content to be added to the generated HTML report.

The code given below:

```
report.setHeadTagIncluded(false); // where report is an object of
  type QbReport
```

generates the report as a DHTML table when the report is exported. This block can then be included in another page with more content than just the report.

#### 8.1.5.7.5. Custom Links for Multi-page DHTML Export

You can also create your own links at the top of the page or remove links generated at the top of the pages for single page and multi page exports. This allows custom links to be added to DHTML reports.

Custom links can be generated by implementing `IHTMLLinksProvider` [ <http://data.quadbase.com/Docs70/eres/help/apidocs/quadbase/reportdesigner/util/IHTMLLinksProvider.html> ] interface. For example, the code below, which can be run as an application or an applet, demonstrates creating custom links:

```
Component doCustomLinks(Object parent) {
  QbReport.setEspressManagerUsed(false);
  QbReport report = new QbReport(parent, "CustomLinks.rpt");
  report.setHTMLLinksProvider(this);
  QbReport.setExportToMultiPages(true);

  try
  {
    report.export(IExportConstants.DHTML, "CustomLinks.html");
  } catch (Exception ex) {
    ex.printStackTrace();
  }

  return (new Viewer().getComponent(report));
}
```

```

public HTMLBlock getLinksForDHTML(int cPage, int tPage, String
    prefix, boolean top)
{
    String linksText = "<CENTER><font face=\\\\"verdana\\\\" size=\\\\"2\\\\"> " +
        "<a href=\\\\" + HTMLBlock.getFirstFileName(prefix) + "\\\">FIRST</a> |
    ";
    if (cPage <= 2) {
        linksText += "<a href=\\\\" + HTMLBlock.getFirstFileName(prefix) + "\\\"
    \>PREV</a> | "
        + "<a href=\\\\" + HTMLBlock.getNextFileName(cPage, tPage, prefix)
        + "\\\">NEXT</a> | " + "<a href=\\\\" +
    HTMLBlock.getLastFileName(prefix)
        + "\\\">LAST</a> [" + "<a href=\\\\"
        + HTMLBlock.getCurrentFileName(cPage, tPage, prefix)
        + "\\?x=x\\\" target=\\\\"print\\\">Print Version</a>]" + "</font></
    CENTER>";
    } else {
        linksText += "<a href=\\\\" + HTMLBlock.getPreviousFileName(cPage, tPage,
    prefix)
        + "\\\">PREV</a> | " + "<a href=\\\\"
        + HTMLBlock.getNextFileName(cPage, tPage, prefix) + "\\\">NEXT</a> | "
        + "<a href=\\\\" + HTMLBlock.getLastFileName(prefix) + "\\\">LAST</a> ["
        + "<a href=\\\\" + HTMLBlock.getCurrentFileName(cPage, tPage, prefix)
        + "\\?x=x\\\" target=\\\\"print\\\">Print Version</a>]" + "</font></
    CENTER>";
    }
}

String text = "<SCRIPT>\n" + "document.write(\"" + linksText + "\");\n"
+ "</SCRIPT>";
return new HTMLBlock(text, 100);
}

public HTMLBlock getLinksForHTML(int cPage, int tPage, String
    prefix, boolean top)
{
    return getLinksForDHTML(cPage, tPage, prefix, top);
}

public HTMLBlock getTOCLinkForDHTML(int cPage, int tPage, String body) {
    return null;
}

```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/CustomLinks.zip> ]

Exported Results [ <http://data.quadbase.com/Docs70/help/manual/code/export/CustomLinks.html> ]

Similarly other links can be generated and set to the report.

### 8.1.5.7.6. Setting the Pixels Per Inch Ratio for DHTML Export

ERES allows a “pixels per inch” ratio to be specified while exporting DHTML content. This is especially useful when you are moving and deploying reports between platforms or when exporting reports in a system without any graphics. You can specify the ratio using the following method in QbReport:

```

public void setPixelPerInchForExport(int pixelPerInchRatio);

```

### 8.1.5.7.7. Pre-load Charts

ERES allows charts with independent data sources to be pre-loaded before the report is exported. The pre-loading is simultaneous, so it improves export performance (especially if there are multiple charts in the report). You can specify to pre-load the charts by calling the following method before exporting the report:

```
public void preloadChartObjects();
```

### 8.1.5.7.8. IExportThreadListener for .view files

Reports can be exported to .view files. These files are used by Page Viewer to show the report a few pages at a time (as opposed to loading the entire report in Report Viewer). Exporting to .view files is similar to exporting to multiple pages in that the one export method results in multiple pages (the more the number of pages in the report, the more files that are generated). However, Page Viewer cannot be started unless the .view file is generated (the pages of the report are generated as .page files). ERES provides an interface, IExportThreadListener [ <http://data.quadbase.com/Docs70/eres/help/apidocs/quadbase/reportdesigner/util/IExportThreadListener.html> ], which you can implement to perform actions when the first page is exported and when the export is completed.

Given below is an example using the IExportThreadListener:

```
public class ExportView extends Applet implements IExportThreadListener {
    static final long serialVersionUID = 1;

    public static void main(java.lang.String[] args) {

        try {

            ExportView report = new ExportView();
            report.exportView(null);

        } catch (Exception ex) {

            ex.printStackTrace();

        }

    }

    // creates report and return it
    void exportView(Object parent) throws Exception {

        // Turn off ReportServer as it is not needed.
        QbReport.setEspressoManagerUsed(false);

        // Create the colinfo array to be used in the QbReport constructor
        QbReport report = new QbReport(parent, "ExportView.rpt");

        report.setMultiPageExp(true);
        report.export(QbReport.VIEW, "ExportView", new Properties(), this);
        System.out.println("DONE!");

    }

    public void endAction() {
        System.out.println("END ACTION");
    }

    public void firstPageFinishedAction() {
```

```

System.out.println("FIRST PAGE FINISHED ACTION");
quadbase.reportdesigner.PageViewer.Viewer.main(new String[]
{ "ExportView.view" });
}
}
}

```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/ExportView.zip> ]

### 8.1.5.7.9. Virtual Memory/Paging

In addition to memory optimized exporting, ERES also has a memory paging feature to handle large amounts of data. In memory paging, you specify the amount of memory and a temp directory. When the amount of memory used exceeds the number specified, the data is compacted and then stored on disk in the temp directory specified.

To use memory paging export, the following code must be added before calling the QbReport constructor:

```

QbReport.setTempDirectory
    (<specify the temp directory to store data. Default is ./temp>);
QbReport.setMaxFieldSize
    (<specify the maximum field size length.
    Value is specified in bytes and the default is 500>);
QbReport.setPagingThreshold
    (<specify the threshold to turn on the memory paging feature.
    Value is specified in Megabytes (MB) and the default is -1
    i.e., disable the memory paging feature>);
QbReport.setPageBufferSize
    (<specify the amount of data kept in memory for the paging feature
    (excess data will be written to disk). Value is specified in
    Megabytes (MB) and the default is 20>);
QbReport.setTotalPageBufferSize
    (<specify the total amount of data kept in memory for the paging
    feature.
    This can be used to control the total memory usage for multiple
    reports running at the same time.
    Value is specified in Megabytes (MB) and the default is 256>);

```

For more information regarding these memory paging options, see Section 1.4.1.3 - Server Options.

### 8.1.5.7.10. Streaming Reports

In addition to exporting to local drives, reports can also be exported as a byte stream and streamed directly to a web browser. However, in order for charts, drill-downs, and parameters to function correctly, this export method requires that several support servlets be available. The three servlets that does this are RPTImageGenerator, DrillDownReportServlet, and ParamReportGeneratorServlet, and they are located in the <ERES>/WEB-INF/classes/ directory. The three files must be copied to the servlet directory of your servlet runner. Use the following code to connect to the servlets.

```

report.setDynamicExport(true, "Machine Name or IP Address", Port);
report.setServletDirectory("Servlet Directory");

```

These methods sees to it that any call to these servlets are pointing to the correct machine and port. The servlets get called using the URL, `http://<MachineName>:<Port>/<ServletDirectory>/<Servlet>.` You must set dynamic export when streaming your report, but setting the servlet directory is optional. If you do not specify which servlet directory to use, it will be automatically set to `servlet/.`

Given below is an example that exports a report to DHTML and streams it to the browser. In order to run the example, you will need to configure and compile the source code, then deploy the class file in the servlet direc-

tory of your servlet runner. Replace the `reportTemplate` variable with either an absolute path or a path relative to the working directory of your application server. Remember that the working directory is usually not the same as the servlet directory. In addition, make sure to add `ERESServer.jar`, `ERESOrganizer.jar` and `qblicense.jar` to the classpath of your application server. These jar files can be found in `<ERES>/lib/` and `<ERES>WEB-INF/lib`.

```
public class StreamingReport extends HttpServlet {
    static final long serialVersionUID = 1;

    @Override
    public void doGet(HttpServletRequest req, HttpServletResponse res) throws
        ServletException,
        IOException
    {
        // Do not use EspressoManager
        QbReport.setEspressoManagerUsed(false);

        // Location of report template
        String reportTemplate = "StreamingReport.rpt";

        // Create QbReport object
        QbReport report = new QbReport(null, reportTemplate);

        // Set up connection information for RPTImageGenerator and
        DrillDownReportServlet
        report.setDynamicExport(true, "localhost", 8080);
        report.setServletDirectory("servlet/");

        // Export the report to DHTML and stream the bytes
        ByteArrayOutputStream reportBytes = new ByteArrayOutputStream();

        try {
            report.export(QbReport.DHTML, reportBytes); // where report is an object
            // report.export(QbReport.PDF, reportBytes); // where report is an
            // object of type QbReport
        } catch (Exception ex)
        {
            ex.printStackTrace();
        }

        res.setContentType("text/html"); // where res is the HttpServletResponse
        // res.setContentType("application/pdf"); // where res is the
        // HttpServletResponse
        res.setContentLength(reportBytes.size());

        OutputStream toClient = res.getOutputStream();
        reportBytes.writeTo(toClient);
        toClient.flush();
        toClient.close();
    }
}
```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/StreamingReport.zip> ]

To run the example, make sure to copy the report templates to the location accessed by the code. If using a relative path (as shown in the example), remember that the current directory is the working directory of your application server. For example, since the working directory in Tomcat is `<Tomcat>/bin`, you will need to create the di-

rectory <Tomcat>/templates/ and copy the report templates there to run the code. The resulting report is shown below.

### Woodview Sales By Region

CategoryName	ProductName	East	Midwest	South	West	Total Sales
Arm Chairs	Adad Chair	\$963,414.00	\$292,896.00	0	0	\$1,256,310.00
	Cula Chair	\$1,526,364.00	\$644,274.00	\$960,336.00	\$216,432.00	\$3,347,406.00
	Marduk Chair	\$1,478,736.00	\$336,960.00	0	0	\$1,815,696.00
	Nabu Chair	\$295,488.00	0	\$270,864.00	\$762,156.00	\$1,328,508.00
	Ningiro Chair	\$317,304.00	\$154,872.00	\$232,308.00	0	\$704,484.00
	Nisaba Chair	\$2,827,656.00	\$389,448.00	\$538,900.00	\$201,204.00	\$3,977,208.00
	Nusku Chair	\$1,311,984.00	\$206,550.00	\$895,050.00	\$459,000.00	\$2,872,584.00
	Srogantusa Chair	\$1,374,894.00	0	\$563,220.00	0	\$1,938,114.00
	Shimaliya Chair	\$782,784.00	0	\$977,616.00	\$173,880.00	\$1,934,280.00
	Category Total:	\$10,878,624.00	\$2,025,000.00	\$4,458,294.00	\$1,812,672.00	\$19,174,590.00
Double Dressers	Sekinet Dresser	\$608,148.00	0	0	0	\$608,148.00
	Seiket Dresser	\$2,214,864.00	0	\$1,655,640.00	0	\$3,870,504.00
	Set Dresser	\$1,421,280.00	0	\$671,328.00	0	\$2,092,608.00
	Shu Dresser	0	\$1,148,256.00	0	0	\$1,148,256.00
	Tefrut Dresser	\$873,828.00	0	0	0	\$873,828.00
	Thoth Dresser	\$2,097,630.00	0	\$505,440.00	0	\$2,603,070.00
	Category Total:	\$7,215,750.00	\$1,148,256.00	\$2,832,408.00	\$8.00	\$11,196,414.00

### Streaming Report

#### 8.1.5.7.10.1. RPTImageGenerator

The `RPTImageGenerator` is used when you export reports containing charts or BLOB images to DHTML format. DHTML files are purely textual; therefore, charts and images must be stored separately. When you export a report locally, the chart is stored as an image file and the file path is appended to the `IMG` tag of the HTML page.

```
<IMG SRC="stream_0.jpg" ALIGN="CENTER" VALIGN="TOP" BORDER="0" HEIGHT="288"
WIDTH="576"></IMG>
```

Similarly, when you stream the DHTML file, only the textual component is sent to the browser. Charts are handled in the `RPTImageGenerator` servlet. Instead of pointing the `IMG` tag to the image file, the tag is set to the `RPTImageGenerator` servlet and appends the ID of the chart image as a parameters.

```
<IMG SRC="http://localhost:8080/servlet/RPTImageGenerator?
ID=1166125348416_0" ALIGN="CENTER" VALIGN="TOP" BORDER="0" HEIGHT="288"
WIDTH="576"></IMG>
```

Reports containing images that are stored locally, do not require the servlet. The `IMG` source will point directly to the image file. However, images retrieved from the datasource (BLOBs) are treated similar to charts and does require the aid of the `RPTImageGenerator` servlet.

#### 8.1.5.7.10.2. DrillDownReportServlet

Regardless if you are exporting the report to a byte stream or to a file, reports with drill-downs require that the `DrillDownReportServlet` be accessible in order to function properly. When you export drill-down reports to either DHTML or PDF format, you must have `DrillDownReportServlet` in the servlet directory of your servlet runner. This servlet provides the link to the next level.

```
<a href="http://localhost:8080/servlet/DrillDownReportServlet?
FILENAME=DrillDown%2Fstream_lvl111.rpt&FORMAT=4&PARAM0=112+Ishtar+Chair"
title="tt" alt="tt">
Ishtar Chair<BR></a>
```

Since drill-down levels must contain a parameterized query, you need to make sure that the `ParamReportGeneratorServlet` is in the servlet directory as well. More information on the `ParamReportGeneratorServlet` are available in the next several sections.

If you export the report to DHTML, the column linked to the drill-down report will display as a hypertext link. In PDF format, the linked column will not look any different from the others; however, when you hover the mouse over that column, the cursor will change to a hand.



CategoryName	ProductName	UnitPrice	StainPrice	UnitsInStock
<a href="#">Arm Chairs</a>	Adad Chair	452	35	16
	Cula Chair	468	33	4
	Marduk Chair	489	31	12
	Nabu Chair	456	31	25
	Ningirsu Chair	478	35	15
	Nisaba Chair	414	29	14
	Nusku Chair	425	31	36
	Sbuqammua Chair	445	32	45
	Shimaliya Chair	424	36	31
<a href="#">Double Dressers</a>	Sekhmet Dresser	1,877	412	14
	Serket Dresser	1,761	414	4
	Set Dresser	1,645	427	17

*Root Level*

Depending on the link clicked, the servlet delivers the appropriate next level report.

CategoryName : [Arm Chairs](#)

ProductName	OrderDate	Quantity
Marduk Chair	Mar 21, 2002	8
Marduk Chair	May 27, 2003	16
Marduk Chair	Jun 9, 2003	12
Marduk Chair	Oct 12, 2003	14
Marduk Chair	Dec 2, 2003	18
Nabu Chair	Feb 12, 2002	11
Nabu Chair	Sep 17, 2002	16
Nabu Chair	Nov 12, 2002	14
Nabu Chair	Dec 2, 2003	12
Cula Chair	Apr 14, 2001	16
Cula Chair	Sep 17, 2001	16
Cula Chair	Nov 16, 2001	18
Cula Chair	Apr 22, 2002	8
Cula Chair	Oct 24, 2002	16
Cula Chair	Dec 4, 2002	8
Cula Chair	Apr 22, 2003	18
Cula Chair	Jul 21, 2003	3

*The Drill-Down Level*

Note that drill-down reports exported to formats other than DHTML or PDF will only show the current level.

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/StreamingDrillDown.zip> ]

### 8.1.5.7.10.3. Generating HTML Parameter Page

When streaming the report, a powerful alternative to setting parameter values using the API (discussed in Section 8.1.5.1.1 - Sub-Reports, Charts, and Drill-Down Reports) is to use an automatically generated HTML form that contains text or selection inputs. After the user submits the form, the parameter values are passed to the `ParamReportGeneratorServlet` that processes them and streams back a report initialized with these parameter values.

The main classes for this purpose are `ParameterPage`, `ParameterPageWriter`, `HtmlParameterPageWriter`, and `CssHtmlParameterPageWriter`. The typical method of generating the HTML Parameter Page involves the following code:

```

OutputStream out = ...; //response.getOutputStream();
ParameterPage paramPage = QbReport.getParameterPage();
Writer writer = new PrintWriter(out);
HtmlParameterPageWriter paramPageWriter = new
    HtmlParameterPageWriter(paramPage, writer);
paramPageWriter.writePage();
writer.flush();
writer.close();
    
```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/HTMLParamPage.zip> ]

When you use the above functionality, you must have the `ParamReportGeneratorServlet` in the `servlet` directory of your servlet runner. This servlet generates the report in the specified format using the parameters specified in the HTML page.

When you run the example, the following page is automatically generated:

*Generated Parameter Page*

Depending on the values you select, clicking on the *Submit* button may result in the following report:

Region	Company	ContactName	State	Orders
East	Furniture Gallery	Eddie Birdsell	NY	5
	Eastern Treasures	Frances Polk	NJ	10
	Gale Distributors	Herb Gale	NY	2
	Allied Furniture Emporium	Miltida Gladwaller	NY	9
	Furniture Palace	Muriel Fedder	NJ	8
	Campbell Interiors	Paul Campbell	NY	3

Region	Company	ContactName	State	Orders
Midwest	All Unfinished Furniture	Arthur Childs	OH	4
	George Park Imports	Ernest Morrow	PA	2
	Woodworks Furniture	Sally Hayes	PA	7
	Imports & Leather Gallery	Seymour Glass	IL	8

*Parameterized Report*

Although it is convenient to automatically generate the entire parameter page, there are times when you want more control over the appearance of the page. One way to achieve this is to only generate the main form, allowing you to design the contents around the main form to provide your own look and feel. Another way is to use CSS to set the desired format. A third way is to generate each component within the form individually giving you control over every element on the page. These various methods are discussed in greater detail in the next two sections.

#### 8.1.5.7.10.4. Utilizing Cascading Style Sheets (CSS)

Two powerful ways to alter the HTML parameter page are to add content around the form and to utilize cascading style sheets. These methods allow you to maintain a consistent look and feel to your web application without the necessity to write an excessive amount of code.

The example below shows how these methods can be used to present the parameter page using your predefined colors, fonts, style, and format. In order to run this example, you need to modify the template path. Keep in mind

that if you use a relative path, it will be relative to the working directory of your application server. You will also be required to copy the CSS file to the root directory of your web application so that it can be accessible by the CSS link in the code (alternatively, you can position the CSS file elsewhere and modify the link). The examples require that both ParamReportGeneratorServlet and DrillDownReportServlet be available in the servlet directory since the example includes a drill-down report.

```
CssHtmlParameterPageWriter cssParamPageWriter =
    new CssHtmlParameterPageWriter(paramPage,
        tempStringWriter);

// Specify the css file to be used
cssParamPageWriter.setCssFile("http://localhost:8080/CssExample.css");

// Create head of html page, adding title and css specification
cssParamPageWriter.write("<HEAD>\n" +
    "<link REL=\"stylesheet\" TYPE=\"text/css\" " +
    "href=\"" + cssParamPageWriter.getCssFile() + "\">" +
    "\n<TITLE>Welcome to Woodview</TITLE>\n" +
    "</HEAD>");

// Body section - Most of the formatting is done in the css file,
// very little is needed here
cssParamPageWriter.write("\n<body><p> </p>");

// Add title and fieldset
cssParamPageWriter.write("<TABLE class=\"outer\"><TR><TD>" +
    "<p class=\"heading2\">Welcome to the Order History
    Database." +
    "</TD></TR><TR><TD>&nbsp;</TD></TR>" +
    "<TR><TD><FIELDSET><LEGEND>Select</LEGEND>");

// Add the parameter forms
cssParamPageWriter.writeBodyBody();

// Finish it off
cssParamPageWriter.write("</FIELDSET></TD></TR>" +
    "</TABLE></body>\n</html>");
```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/CSSParamPage.zip> ]

In the above segment of code, the <HEAD> is written manually so that a title can be inserted. In the body section, only the form is generated. Using this method, users can add titles, images, and other components to maintain their own look and feel. Depending on the CSS file provided, the result might look like this:

**Welcome to the Order History Database.**

*Parameter Page using CSS*

Depending on the parameter options you select, the result may look like the following:

---



---

*Woodview Furniture*


---



---

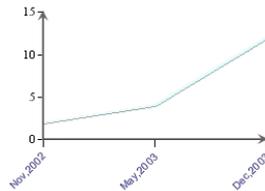
**Product Order History**

ProductID : 30

ProductName : Horus Table      Description : Redwood

OrderID	OrderDate	UnitPrice	Quantity
<a href="#">10,049</a>	Nov 12, 2002	1,354	2
<a href="#">10,068</a>	May 22, 2003	1,354	4
<a href="#">10,096</a>	Dec 9, 2003	1,354	12

Total Sold: 18

*Example Results***8.1.5.7.10.5. Customize the Parameter Page**

Another way to customize the parameter page is to write a java class that extends `HtmlParameterPageWriter` [ <http://data.quadbase.com/Docs70/eres/help/apidocs/quadbase/common/param/HtmlParameterPageWriter.html> ] or `CssHtmlParameterPageWriter` [ <http://data.quadbase.com/Docs70/eres/help/apidocs/quadbase/common/param/CssHtmlParameterPageWriter.html> ]. This approach gives you access to the protected methods available in these classes. For information on the full set of methods available in these classes, please see the APIDocs [ <http://data.quadbase.com/Docs70/eres/help/apidocs/index.html> ].

The following is an example extending the `HtmlParameterPageWriter` class. To run this example, you will need to copy the two html pages to the root directory of your application server. The template and java class files must be copied to the working directory and servlet directory of your application server respectively. The example also requires that `ParamReportGeneratorServlet` and `DrillDownReportServlet` be placed in the servlet directory.

```
private class MyHtmlParameterPageWriter extends HtmlParameterPageWriter{

    public MyHtmlParameterPageWriter(ParameterPage pp, StringWriter sw) {
        super(pp, sw); }

    public void printCustom() {
        try {
            write("<BODY bgcolor=#FDFAED>");

            write("<FORM action = \"" + servletName + "\" target=\"main
|\" method = GET>\n");
            write("<TABLE ALIGN=\"CENTER\"><TR><TD><IMG src=http://
www.quadbase.com/FurnitureImages/Woodview.gif></TD>");
            write("<TD WIDTH=50></TD><TD>");
            writeParamTable();

            write("</TD><TD WIDTH=50></TD><TD>");
            writeSubmitButton();
            write("</TD>");
            write("</TR></TABLE>");

            writeHiddenParamValue("ReportFilePath", reportLocation);
```

```

write("\n");

writeHiddenParamValue("ReportExportFormat", ""+format);
write("\n");

writeHiddenParamValue("ServerName", servletAddress);
write("\n");

writeHiddenParamValue("ServletRunnerPort", ""+servletPort);

write("\n");
writeHiddenParamValue("ServletDirectory",
servletDirectory);

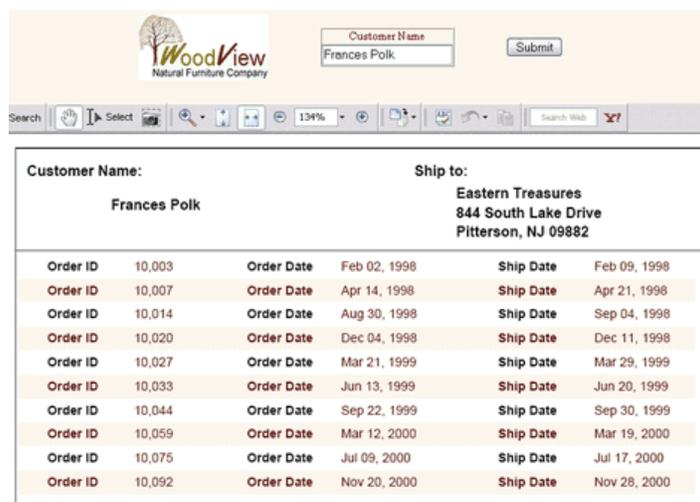
write("\n</FORM>");
html.bodyEnd();
html.htmlEnd();
} catch (Exception e) {
    e.printStackTrace();
}
}
}

```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/CustomParamPage.zip> ]

Notice that by using the protected methods, you are required to add a number of details in your code, such as FORM tags and hidden parameters. However, this also grants your the ability to add other elements such as company logos into the form section of the page and arrange the elements to your preference. This approach also allows you to eliminate certain elements from the page such as the *reset* button.

Once the files have been copied to the correct locations, you can run the example by entering the following url in your browser: `http://<IP Address/localhost>:<Port>/CustomParamPage.html`. You will see a page with two frames. The top frame is the parameter page and the bottom frame initially displays a simple message. To view the report, enter the full name of a customer (e.g. Francis Polk).



*Order History Report*

This example exports to PDF instead of DHTML. Although there are no links displayed in PDF format, if you hover the mouse over the *Order ID* column, you will see that the mouse pointer turns into a hand alerting you that you can drill-down from this column. Depending on the *Order ID* you selected, the follow report may be shown:

Order Summary for ID:10,007

Side Chairs						
Product Name	Description	Unit Price	Stain Price	Stain	Quantity	SubTotal
Ninhursag Chair	Cedar Bowback Chair	\$369.00	\$25.00	No	14	\$5,166.00

Arm Chairs						
Product Name	Description	Unit Price	Stain Price	Stain	Quantity	SubTotal
Cula Chair	Birch Fanback Armchair	\$468.00	\$33.00	No	16	\$7,488.00

Oval Tables						
Product Name	Description	Unit Price	Stain Price	Stain	Quantity	SubTotal
Ma'at Table	Cedar Oval Table	\$1,875.00	\$378.00	Yes	2	\$4,506.00

<b>Total:</b>	\$17,160.00
<b>Tax (7.25%):</b>	\$1,244.10
<b>Grand Total:</b>	\$18,404.10

Order History Detail (Drill-Down)

### 8.1.5.8. Calling Report Designer from Report API

Report Designer can be called from Report API in either an application or an applet. Depending on the code, you can pass in parameters to open up Report Designer with a specified .rpt file or a specified data source or other parameters.

To call Report Designer from Report API, you must:

- make sure that ERES Server is up and running;
- add ERESOrganizer.jar, ERESserver.jar and qblicense.jar to the CLASSPATH;
- make sure that the information to connect to ERES Server is specified using the relevant API calls (This is especially important if the ERES Server is on a different machine and/or if it started with a port number other than 22071);
- copy the images, reportimages and backgroundimages directories to the working directory or use QbReportDesigner.setUseSysResourceImages(true) in your code to use the images from the jar files;

Depending on the -RequireLogin and -QbDesignerPassword flags for ERES Server, you may need to pass in a userid and/or password to connect to ERES Server. If the -RequireLogin flag is set for ERES Server, you need to add the following line of code before calling setVisible() or any getDesigner methods:

```
public login(String userName, String password); // Method found in
QbReportDesigner class
```

If the -QbDesignerPassword is specified for ERES Server, you will need to add the following line of code before calling setVisible() or any getDesigner methods:

```
public login(String password); // Method found in QbReportDesigner class
```

You can also specify a look and feel to Report Designer (Report Designer will use the system's look and feel by default). This is done by using the following method in QbReportDesigner:

```
public static void setLookAndFeel(javax.swing.LookAndFeel newLookAndFeel);
```

Given below are the different ways Report Designer can be called via Report API. Note that if you are running the example code as an applet, you need to change the ERES Server machine from 127.0.0.1 (or localhost) to the ERES Server machine's name or IP address.

### 8.1.5.8.1. Specify a Report Template

You can open Report Designer with a specified report template file. This allows the end users to create their own custom reports in Report Designer GUI and then view the finished report.

The following constructor is used:

```
QbReportDesigner(Object parent, String templateFileName);
```

Given below is an example of calling Report Designer with a specified .rpt file:

```
public void doReportDesignerApplet(Applet applet) {
    // Connect to ERES Server
    QbReportDesigner.useServlet(true);
    QbReportDesigner.setServletRunner(ServletRunner);
    QbReportDesigner.setServletContext(ServletContext);

    // Use images from jar file
    QbReportDesigner.setUseSysResourceImages(true);

    // Create a new QbReportDesigner instance
    designer = new QbReportDesigner(applet, "RDWithTemplateFile.rpt");

    // Overwrites default saveFile method
    designer.setReportIO(this);

    // Start Designer
    designer.setVisible(true);
}

// Save the file to a temp directory
public void saveFile(byte[] data, String fileLocation) {

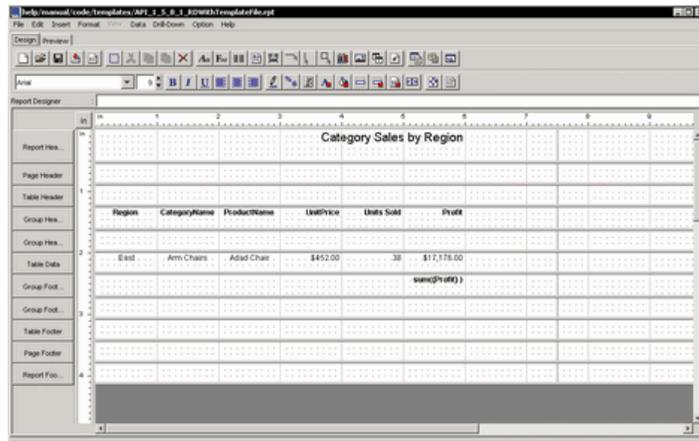
    System.out.println("OLD LOCATION: " + fileLocation);
    String newLoc = fileLocation.replace('/', '\\');
    int idx = newLoc.lastIndexOf("\\");
    newLoc = "temp/" + newLoc.substring(idx + 1);
    try {
        designer.writeFile(newLoc, data);
        System.out.println("NEW LOCATION: " + newLoc);
    } catch (Exception ex) {
        ex.printStackTrace();
    }
    designer.getDesigner().setTitle(newLoc);
    designer.getDesigner().repaint();
}
```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/RDWithTemplateFileERES.zip> ]

Exported Results [ <http://data.quadbase.com/Docs70/help/manual/code/export/RDWithTemplateFileEndUser.pdf> ]

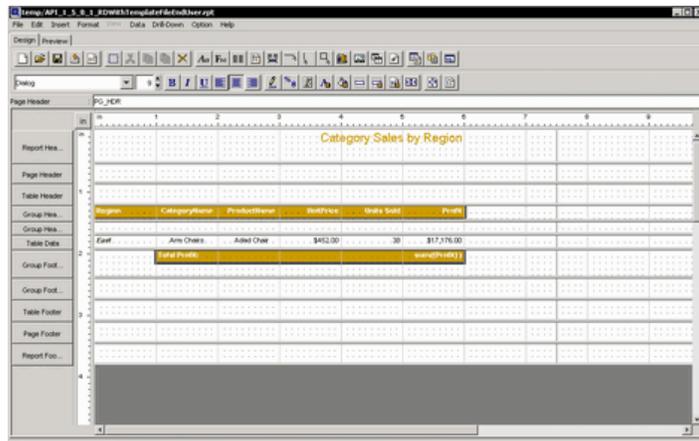
Note that the above code can be used as both an application and an applet. The above code also changes the *Save* functionality in Report Designer so that the .rpt file is always saved in the temp directory.

When the user runs this code, the Report Designer is launched with the report template you specified.



*Report Designer with Template*

The user can then customize the report and save the results.



*End User Customization*

You can also specify the registry to use when running the above code. This is achieved by adding in the following line of code before setting the designer visible:

```
designer.setDataRegistry("DataRegistry/Data_Registry.xml");
```

The above line specifies the Report Designer to use Data\_Registry.xml as the Data Registry when choosing the data sources for a new report.

### 8.1.5.8.2. Specify a Data Registry

You can open Report Designer and have it starting with a Data Source Manager (with a specified .xml file for the Data Registry). This allows the end users to choose the data source and create their own custom reports in Report Designer GUI and then view the finished report.

The following constructor is used:

```
QbReportDesigner(Object parent, String nameOfXMLFile, boolean  
doNotStartWithOpenNewReportDialogWizard);
```

Given below is an example of calling Report Designer with a specified .xml file:

```

public void doReportDesignerWithDataRegApplet(Applet applet) {

    // Connect to ERES Server
    QbReportDesigner.useServlet(true);
    QbReportDesigner.setServletRunner("http://localhost:8080");
    QbReportDesigner.setServletContext("ERES/servlet");

    // Use images from jar file
    QbReportDesigner.setUseSysResourceImages(true);

    // QbReportDesigner (component, name of XML file, boolean);
    // true = start from Data Source Manager false = start from Create a
new report
    // or open existing report choice before going to Data Source Manager
    designer = new QbReportDesigner(applet, "DataRegistry/Sample.xml",
true);

    designer.setVisible(true);

}

```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/RDWithDataRegistryERES.zip> ]

Note that the above code can be used as both an application and an applet.

You can also control what the user does at the Data Registry by enabling or disabling the options available. For example, you can remove complete or parts of data sources in the Data Registry and you can disable the options for adding, copying, editing, and deleting of data sources. Note that the nodes are not hidden; they are removed. Therefore, when the Data Registry is opened (without any restrictions), the contents of the removed nodes are lost. You can create a backup of the Data Registry .xml file and use the backup to enable/disable options.

Given below is an example of calling Report Designer with a specified data registry and setting it up so that only the database data sources are available and no queries or data views can be added.

```

public void doReportDesignerWithDataRegOptionApplet(Applet applet) {

    // Connect to ERES Server
    QbReportDesigner.useServlet(true);
    QbReportDesigner.setServletRunner("http://localhost:8080");
    QbReportDesigner.setServletContext("ERES/servlet");

    // Use images from jar file
    QbReportDesigner.setUseSysResourceImages(true);

    // QbReportDesigner (component, name of XML file, true = start from
// Data Source Manager false = start from Create a new report or open
// existing report choice before going to Data Source Manager
    designer = new QbReportDesigner(applet,
        "help/manual/code/templates/Sample.xml", true);
    designer.addDataSourceManagerListener(this);
    designer.setVisible(true);

}

public JTree modifyDataSourceTree(JTree tree) {

```

```

    // The following code will remove all nodes from the tree except for

```

---

```

    // the database nodes.
    DefaultDataSourceNode root =
    (DefaultDataSourceNode)tree.getModel().getRoot();
    for (int i=root.getChildCount()-1; i>=1; i--)
    {
        // System.out.println("removing");
        root.remove(i);
    }

    // The following code will prevent the user from adding any query or
    dataview.
    DefaultDataSourceNode databaseHeading =
    (DefaultDataSourceNode)root.getChildAt(0);
    for (int i=0; i<databaseHeading.getChildCount(); i++)
    {
        ((DefaultDataSourceNode)databaseHeading.getChildAt(i)
        .getChildAt(0)).setAddEnabled(false);
        ((DefaultDataSourceNode)databaseHeading.getChildAt(i)
        .getChildAt(1)).setAddEnabled(false);
    }

    return tree;
}

```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/RDWithDataRegistry2ERES.zip> ]

Note that the above code can be used as both an application and an applet.

### 8.1.5.8.3. Specify a DBInfo Object (Database Information)

You can open Report Designer and have it starting at the *Select Report Type* wizard window (with a specified DBInfo object). This allows the end users to create their own custom reports in Report Designer GUI and then view the finished report.

The following constructor is used:

```

QbReportDesigner(Object parent, DBInfo databaseInformation, boolean
doNotStartWithOpenNewReportDialogWizard);

```

Given below is an example of calling Report Designer with a specified DBInfo object:

```

public QbReportDesigner designer;
String url = "jdbc:hsqldb:help/examples/DataSources/database/woodview";
String driver = "org.hsqldb.jdbcDriver";
String username = "sa";
String password = "";
String query = "SELECT * FROM Order_Details";

public void doReportDesignerWithDBInfoApplet(Applet applet) {

    // Connect to ERES Server
    QbReportDesigner.useServlet(true);
    QbReportDesigner.setServletRunner("http://localhost:8080");
}

```

```

QbReportDesigner.setServletContext("ERES/servlet");

// Use images from jar file
QbReportDesigner.setUseSysResourceImages(true);

// QbReportDesigner (component, Database Information, true = start
from Data Source Manager
// false = start from Create a new report or open existing report
choice before
// going to Data Source Manager
DBInfo dbInfo = new DBInfo(url, driver, username, password, query);
designer = new QbReportDesigner(applet, dbInfo, true);

designer.setVisible(true);
}

```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/RDWithDBInfoERES.zip> ]

Note that the above code can be used as both an application and an applet.

#### 8.1.5.8.4. Open Query Builder with a Specified DBInfo Object (Database Information)

You can open Query Builder with a specified DBInfo object. This allows the end users to create their own custom SQL queries in Query Builder GUI and save them.

Given below is an example of calling Query Builder with a specified DBInfo object:

```

public void doQueryBuilderApplet(Applet applet, String sqlName) {

    QbReportDesigner.useServlet(true);
    QbReportDesigner.setServletRunner("http://localhost:8080");
    QbReportDesigner.setServletContext("ERES/servlet");

    queryMain = new QbQueryBuilder(applet, this);
    if (sqlName != null) {
        // sqlName .qry file name (without extension)
        // System.out.println("OPEN QUERY - " + sqlName);
        queryMain.openQuery(sqlName, false, null, null, driver, url,
username, password);
    } else {
        // System.out.println("NEW QUERY ");
        queryMain.newQuery("Setup Query", false, null, null, driver,
url, username, password);
    }
    frame = queryMain.getBuilder();
    frame.setVisible(true);
    try { queryMain.showTablesWindow();
    } catch (Exception ex) {};
}

public void back() {};

public void cancel() {};

```

```

public void next() {

    queryMain.getBuilder().setVisible(false);
    DBInfo dbInfo = new DBInfo(url, driver, username, password,
queryMain.getSQL());
    designer = new QbReportDesigner(applet, dbInfo, queryMain.getInSet(),
true);
    designer.setVisible(true);

}

```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/QBWithDBInfoERES.zip> ]

Note that the above code can be used as both an application and an applet.

You can also open Query Builder by passing in the database connection (schema) as a java object, rather than as a DBInfo object. Given below is an example of calling Query Builder by passing in the schema information:

```

public QBWithDBInfo2() {

    QbReportDesigner.useServlet(true);
    QbReportDesigner.setServletRunner("http://localhost:8080");
    QbReportDesigner.setServletContext("ERES/servlet");
    QbReportDesigner.setUseSysResourceImages(true);
    try {
        Class.forName(driver);
        conn = DriverManager.getConnection(url, username, password);
        metaData = conn.getMetaData();

    } catch (Exception ex) { ex.printStackTrace(); }

    queryMain = new QbQueryBuilder(null, this);
    queryMain.newQuery("Setup Query", this);
    queryMain.setVisible(true);

    try { queryMain.showTablesWindow();
    } catch (Exception ex) {};

}

public void back() {};
public void cancel() {};

public void next() {

    // System.out.println("EXIT");

}

public String getDatabaseProductName() throws Exception {

    return metaData.getDatabaseProductName();

}

public ResultSet getTables(String catalog, String schemPattern, String
tableNamePattern,

```

---

```

String[] types) throws Exception {

    return metaData.getTables(catalog, schemPattern, tableNamePattern,
types);

}

public String getNumericFunctions() throws Exception {

    return metaData.getNumericFunctions();

}

public String getStringFunctions() throws Exception {

    return metaData.getStringFunctions();

}

public String getTimeDateFunctions() throws Exception {

    return metaData.getTimeDateFunctions();

}

public String getSystemFunctions() throws Exception {

    return metaData.getSystemFunctions();

}

public ResultSet executeQuery(String sql) throws Exception {

    Statement stmt = conn.createStatement();
    return stmt.executeQuery(sql);

}

```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/QBWithDBInfo2ERES.zip> ]

Note that the above code can be used as both an application and an applet.

#### 8.1.5.8.5. Open Report Designer with a Specific Class File Data Source

You can open Report Designer with a specified class file data source. This allows the end users to create their own custom reports in Report Designer GUI and then view the finished reports. The class file data source can be non-parameterized or parameterized.

The following constructor is used:

```

QbReportDesigner(Object parent, String classFile,
quadbase.common.paramquery.QueryInParameterSet inset, boolean newReport, int
displayRow, String[] imagesPath);

```

Given below is an example of calling Report Designer with a specified class file data source:

```

public void doReportDesignerWithQueryApplet(Applet applet) {

    QbReportDesigner.useServlet(true);
    QbReportDesigner.setServletRunner("http://localhost:8080");
    QbReportDesigner.setServletContext("ERES/servlet");
    QbReportDesigner.setUseSysResourceImages(true);

    // QbReportDesigner (component, class file location, parameter
    information,
    // true = start from Data Source Manager false = start from Create a
    new
    // report or open existing report choice before going to Data Source
    // Manager, number of rows to display (-1 means show all rows), path
    to
    // Report Designer images
    designer = new QbReportDesigner(applet, "ParamClassFile",
        null, true, -1, null); // Parameterized class file

    QbReportDesigner.setUseSysResourceImages(true);
    designer.setVisible(true);

    JFrame frame = designer.getDesigner();

}

```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/RDWithClassDataERES.zip> ]

For more information on creating your custom class file data source, see Appendix 8.A.5 - Data passed in a Custom Implementation.

### 8.1.5.8.6. Open Report Designer with Custom Functions

You can also include your own functions with the Formula and Scripts dialog boxes that users can use when calling Report Designer from the API. You can in effect create functions that handle complex computations and allow any user to use that same function in Report Designer.

Given below is an example code that shows you how to customize Report Designer by assigning custom functions to the Formulas and Scripts dialog boxes. The code takes in a number and converts it to an IP address by appending the number at the end of 192.168.0.

```

public void doRDWithCustomFunctions(Applet applet) {
    QbReportDesigner.useServlet(true);
    QbReportDesigner.setServletRunner(ServletRunner);
    QbReportDesigner.setServletContext(ServletContext);
    QbReportDesigner.setUseSysResourceImages(true);

    designer = new QbReportDesigner(applet, "RDWithCustomFunctions.rpt");
    designer.setCustomDefinedFunctions(this);
    designer.setVisible(true);
}

public String[] getAllFunctionNames() {
    return new String[] { "inet_ntoa" };
}

public int getReturnType(String functionName) {
    if (functionName.equals("inet_ntoa")) {
        return STRING;
    }
}

```

```

    }
    return -1;
}

public int[] getParamTypes(String functionName) {
    if (functionName.equals("inet_ntoa")) {
        return new int[] { NUMERIC };
    }
    return null;
}

public Object getValue(String functionName, Object[] args) throws Exception
{
    if (functionName.equals("inet_ntoa")) {
        return "192.168.0." + ((Double) args[0]).intValue();
    }
    return null;
}

```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/RDWithCustomFunctionsERES.zip> ]

Note that the above code can be used as both an application and an applet.

The cell containing the function will initially display as “Null” since the function is not part of the core set. However, the function will compute correctly when previewing the data. The value will then be shown correctly thereafter.

### 8.1.5.8.7. Open Report Designer with Pre-Set Directories

You can also set the directories that reports load from and save to. This feature will restrict users from navigating to any level above the specified directory, giving you the ability to control which files the user is able to see.

Given below is an example that shows you how to customize Report Designer by specifying different directories for the load folder and the save folder:

```

public void doReportDesignerApplet(Applet applet) {
    QbReportDesigner.useServlet(true);
    QbReportDesigner.setServletRunner(ServletRunner);
    QbReportDesigner.setServletContext(ServletContext);
    QbReportDesigner.setUseSysResourceImages(true);

    designer = new QbReportDesigner(applet,
        "RDWithPreSetDirectories.rpt");
    designer.setRootDirectoryForBrowse("templates");
    designer.setReportIO(this);

    designer.setVisible(true);
}

public void saveFile(byte[] data, String fileLocation) {

    System.out.println("OLD LOCATION: " + fileLocation);
    String newLoc = fileLocation.replace('/', '\\');
    int idx = newLoc.lastIndexOf("\\");
    newLoc = "temp\\" + newLoc.substring(idx + 1);
    try {
        designer.writeFile(newLoc, data);
        System.out.println("NEW LOCATION: " + newLoc);
    } catch (Exception ex) {

```

```

    ex.printStackTrace();
}
designer.getDesigner().setTitle(newLoc);
designer.getDesigner().repaint();

}

```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/RDWithPreSetDirectoriesERES.zip> ]

Note that the above code can be used as both an application and an applet. Also in the above example, you can choose *Save As* and specify a directory to save the `.rpt` file. However, the `.rpt` file will always be saved in the `temp` directory.

In addition to the above approach, you can also use the following method to get the default directories used by Report Designer:

```
public BrowseDirectories getBrowseDirectories();
```

You can then use the methods in `BrowseDirectories` [ <http://data.quadbase.com/Docs70/eres/help/apidocs/quadbase/common/util/BrowseDirectories.html> ] to get the default location of the directories for the different browse dialogs.

#### 8.1.5.8.8. Open Report Designer with Modified Menubar and Toolbar

You can also add items to the menu and remove items from the toolbar. Given below is an example illustrating that:

```

public void doCustomizeDesignerMenuApplet(Applet applet) {
    QbReportDesigner.useServlet(true);
    QbReportDesigner.setServletRunner(ServletRunner);
    QbReportDesigner.setServletContext(ServletContext);
    QbReportDesigner.setUseSysResourceImages(true);

    designer = new QbReportDesigner(applet, "RDWithModifiedBars.rpt");

    // Add a new menu item
    JMenuBar menuBar = designer.getReportMenuBar();
    JMenu fileMenu = menuBar.getMenu(0);
    newItem = new JMenuItem("Hello World");
    newItem.addActionListener(this);
    fileMenu.insert(newItem, 7);

    // Remove toolbar buttons
    JToolBar designBar = designer.getDesignerToolBar();
    designBar.remove(5); // Remove Separator
    designBar.remove(4); // Remove Apply Template Button
    designer.setSaveOnExitEnabled(false); // Do not prompt to save the report
    if unsaved on exiting Designer

    designer.setVisible(true);
}

public void actionPerformed(ActionEvent e) {

    /*** save report *****/
    designer.save("RDWithModifiedBars_Temp.rpt");
}

```

```

/**** create new testing frame *****/
JPanel contentPane = new JPanel();
contentPane.setLayout(new BorderLayout());
contentPane.add(new JLabel("Hello World!"), "Center");
JFrame frame = new JFrame();
frame.setContentPane(contentPane);
frame.pack();
frame.setVisible(true);
}

```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/RDWithModifiedBarsERES.zip> ]

Note that the above code can be used as both an application and an applet.

### 8.1.5.8.9. Open Report Designer with Skipped Wizard Steps

You can open Report Designer with some of the Wizard Steps skipped. Simply call the following functions to skip the *query result*, *multiple data source*, and *pre-defined templates* steps:

```

qbReportDesigner.setSkipQueryResultStep(true);
qbReportDesigner.setSkipMultiDataSourceStep(true);
qbReportDesigner.setSkipPredefinedTemplatesStep(true);

```

### 8.1.5.8.10. Open Report Designer with Customized Messages and Chart Designer

You can open Report Designer with customized warnings when navigating between sub-reports and drill-down reports. In addition, you can also override the default *Save As* behavior for the Report Designer. The following example demonstrates ways to alter these various components. The example has been broken down into fragments to make it easier to read. The full source code can be found at the end of this section.

```

public class RDWithCustomMessage {

    public static void main(String[] args) {
        try {
            startDesigner(); } catch (Exception ex) {
            ex.printStackTrace(); } }

    static void startDesigner() {
        QbReportDesigner.useServlet(true);
        QbReportDesigner.setServletRunner("http://localhost:8080");
        QbReportDesigner.setServletContext("ERES/servlet");
        QbReportDesigner.setUseSysResourceImages(true);

        QbReportDesigner designer = new QbReportDesigner((Frame)null);
        // modify warning message before adding drill-down layer

        designer.modifyWarningMessage(QbReportDesigner.SAVE_RPT_BEFORE_DRILLDOWN,
            "The report designer will save the current report for you,
            continue?");

        // modify warning message before inserting Sub-Report

        designer.modifyWarningMessage(QbReportDesigner.SAVE_RPT_BEFORE_SUBREPORT,
            "The report designer will save the current report for you,
            continue?");
    }
}

```

```

        // by pass save as dialog when submitting save location
        designer.setReportIO(new ReportIO(designer));

        // by pass save as dialog when user try to navigate to next
level
        designer.setByPassSaveAsIO(new ByPassSaveAsForReport());

        // modify chart designer from report
        designer.setChartDesignerHandle(new ChartDesignerHandle());

        // REMOVE "SAVE AS" option for Report Designer
        JMenu fileMenu = designer.getReportMenuBar().getMenu(0);
        fileMenu.remove(6);
        designer.setVisible(true); }
}

```

The above portion of the code modifies the warning message that is shown when adding a sub-report or a drill-down level. It also calls `setReportIO`, `setByPassSaveAsIO`, and `setChartDesignerHandle` to by pass the *Save As* dialogs (details are discussed below). The code also removes the *Save As* option from the menu bar in Report Designer.

```

public static class ReportIO implements IReportIO {

    String fileName = null;
    QbReportDesigner designer = null;

    public ReportIO(QbReportDesigner designer) { this.designer =
designer; }

    public void saveFile(byte[] data, String fileName) {
        System.out.println("SAVE REPORT FILE - " + fileName);
        try {
            FileOutputStream fout = new FileOutputStream(fileName);
            fout.write(data, 0, data.length); } catch (Exception ex) {
            ex.printStackTrace(); }
        this.fileName = fileName; }
}

```

The above fragment implements the `IReportIO` interface. By setting the `ReportIO` in the previous code fragment, Report Designer will call the `saveFile` method here when the user tries to save the report. The file is passed to this method as a byte array and the filename as a string. This method creates an output stream using the filename and writes the report byte array to that stream.

```

public static class ByPassSaveAsForReport implements IByPassSaveAsForReport
{

    public ByPassSaveAsForReport() {};

    public String getFileName(String originalFileName) {
        System.out.println("BY PASS REPORT SAVE AS OPTION...");
        if (originalFileName == null) return "TEMP_REPORT_FILE.rpt";
        else return originalFileName; }
}

```

```

    public Properties getSaveAsProperties(String originalFileName) {
        return new Properties(); }
}

```

When you by pass the *Save As* option, the `saveFile` method from the previous code fragment will obtain the filename from the `getFileName` method in the above class. Here, we are simplifying the process by hardcoding the name of the file. Typically, you will want to manipulate the filenames so that users do not overwrite each other's files. The `getSaveAsProperties` method returns a `Properties` instance containing any save options. Possible options include: `CREATE_STL`, `SAVE_ALL_DATA`, `CREATE_HTML`, `USE_SWINGVIEWER`, `CREATE_XML`, `CREATE_PACK`, and `USE_PAGEVIEWER`. Here, we choose to not use any.

```

public static class ChartDesignerHandle implements IChartHandle {

    public ChartDesignerHandle() {};

    public void processDesigner(QbChartDesigner designer) {
        System.out.println("PROCESS CHART....");
        // REMOVE "SAVE AS" option for Chart Designer
        JMenu fileMenu = designer.getChartMenuBar().getMenu(0);
        fileMenu.remove(6);
        designer.setChartIO(new ChartIO());
        designer.setByPassSaveAsIO(new ByPassSaveAsForChart()); }

}

public static class ChartIO implements IChartIO {

    public ChartIO() {};

    public String saveChartFile(byte[] data, String fileName) {
        System.out.println("SAVE CHART FILE - " + fileName);
        try {
            FileOutputStream fout = new FileOutputStream(fileName);
            fout.write(data, 0, data.length);
            fout.close(); } catch (Exception ex) {
                ex.printStackTrace(); }
        return fileName; }

}

public static class ByPassSaveAsForChart implements IByPassSaveAsForChart {

    public ByPassSaveAsForChart() {};

    public String getFileName(String originalFileName) {
        System.out.println("BY PASS CHART SAVE AS OPTION...");
        if (originalFileName == null) return "TEMP_CHART_FILE.cht";
        else return originalFileName; }

    public Properties getSaveAsProperties(String originalFileName) {
        return new Properties(); }

}

```

The above three classes makes the same changes to the chart designer side. The `ChartDesignerHandler` class removes the *Save As* option from the menu bar. The `ChartIO` class saves the file given the filename and byte array. The `ByPassSaveAsForChart` class sets a fixed filename for the chart.

### 8.1.5.9. Changing Report Viewer and Page Viewer Options

At times, you may want to configure what users can or cannot do when they are viewing the report using the Report Viewer or Page Viewer.

When the user is using the Viewers to view the report, right clicking on the report causes a menu to pop up. Using this menu, the user can navigate through different pages of the report, as well as perform other tasks on the report, such as exporting it as a DHTML or PDF file. This can easily be done using the Report Viewer and Page Viewer APIs. However, these powerful features of the Report Viewer can be too complicated or overly confusing for the average user. Therefore, a few API methods have been introduced to control what options are available in the pop-up menu of the Report/Page Viewer.

These API calls are:

```
viewer.setMenuVisible(boolean b);
viewer.setPageMenuVisible(boolean b);
viewer.setPageMenuItemVisible(String[] items, boolean b);
viewer.setOutputMenuVisible(boolean b);
viewer.setOutputMenuItemVisible(String[] items, boolean b);
viewer.setRefreshMenuItemVisible(boolean b);
viewer.setGoToMenuItemVisible(boolean b);
viewer.setSortMenuVisible(boolean b);
```

For more detailed information on these API methods, please consult the APIDocs [ <http://data.quadbase.com/Docs70/eres/help/apidocs/index.html> ].

### 8.1.6. Javadoc

Javadoc for the entire API is provided along with ERES. The API covers both Report and Charting API. It is located here (Reports and charts) [ <http://data.quadbase.com/Docs70/eres/help/apidocs/index.html> ] and here (ES functions) [ <http://data.quadbase.com/Docs70/eres/help/ERESapidocs/index.html> ].

### 8.1.7. Swing Version

1.1 JFC/Swing versions of Report API are also available. For more details, please refer to `quadbase.reportdesigner.ReportViewer.swing` package [ <http://data.quadbase.com/Docs70/eres/help/apidocs/quadbase/reportdesigner/ReportViewer/swing/package-summary.html> ].

### 8.1.8. Summary

The ERES API provides an easy-to-use, yet powerful application programming interface for business applications. Combined with Report Designer, programming is as simple as adding one line of code to your applet. All of the attributes of a report may be set in a template file, which can be created with the Report Designer. The ERES API has been tested with Netscape's Communicator (4.06 and above), Microsoft's Internet Explorer (4.x and above), and Sun's Appletviewer (1.2 and above) on the Windows 95, Windows NT/2000/XP, Solaris, Linux, AIX, and HP platforms.

## 8.2. ERES Chart API

### 8.2.1. Introduction and Setup

In addition to designing and creating chart templates in the ERES Designer, ERES also provides an easy-to-use application programming interface (API) that enables users to create and customize 2D and 3D charts within their

own applications (and applets). It is written in 100% Pure Java and thus can be run on any platform with little or no modifications necessary. The chart template is completely customizable using the API. You can use as little as four lines of code to add a chart to a report.

The data for the chart can come from two sources:

- The data can come from the report section to which the chart template is set. For instance, a Report Footer, which has a chart template in it, will show the entire report's data. If however, the same template is assigned to the Group Footer, only the partial data within the group is shown in the chart. Therefore, care must be taken to ensure that the correct data is shown with the chart and that the chart templates are set in the proper locations in the report.
- The data can come from an independent data source. This data source can be the same as the data source used to create the report or from a different data source. Please note that the chart data will not change depending on its placement within the report. However, multiple charts may appear depending on which section of the report the chart has been added to.

In addition to adding charts to reports, you can also generate and deploy stand-alone charts, independent of a report. You do not necessarily have to add a chart to a report (even a blank one) to show the chart.

The main classes, `QbChart` and `ChartObject` (which extends `quadbase.ChartAPI.QbChart`) are used for creating and adding a chart object to the report. A set of auxiliary classes contained within two packages is associated with this component: `quadbase.ChartAPI` and `quadbase.util`. The remainder of this document explains the constituents of the API and their usage.



### Note

The complete API documentation is located here [ <http://data.quadbase.com/Docs70/eres/help/api-docs/index.html> ] and here [ <http://data.quadbase.com/Docs70/eres/help/ERESapidocs/index.html> ].

To use the API, add `ERESOrganizer.jar` (located in the `ERES/lib` directory) and `ERESServer.jar` (located in the `ERES/WEB-INF/lib` directory) to your `CLASSPATH`. Please note that if you are also using XML (to output the data, or to read in data), you will also need to add `xercesImpl.jar` and `xml-apis.jar` (also located in the same directory) to the `CLASSPATH` as well. If you wish to use to export the chart to SVG or to Flash, you will need to add `SVGExport.jar` or `FlashExport.jar` (also located in the same directory) to the `CLASSPATH`. If you want to use parameterized database queries as your data sources, add `jsqlparser.jar` to your `CLASSPATH`. Please note that you will also need to include `qblicense.jar` in your `CLASSPATH`. If your application is on a Windows or Solaris machine, you will have to add the following environment variable (depending on the platform):

```
(For Windows) set PATH=%PATH%;<path to ERES root directory>\lib
```

```
(For Solaris) export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:<path to ERES
root directory>/lib
```

## 8.2.2. Recommended Approach for using Chart API

ERES provides API through which charts can be generated programmatically from scratch. However, this generally involves a significant amount of code. We recommend using Designer and creating chart templates (`.TPL` file) first. The chart template can be passed in the `ChartObject` constructor and thus have its look and feel applied to the newly created chart object. Therefore, most of the look and feel will have been set when the chart is generated. You can then write code to modify, add, or remove the properties. This approach will save you coding time and improve performance as well.

When using the ERES Chart API, you have the option of connecting to the ERES Server or not. While a connection is required when using Designer, you do not need to connect to the ERES Server when running any application that utilizes the ERES Chart API. We generally recommend that you do not connect to the ERES Server and thereby avoid another layer in your architecture. For more details, please refer to the next section.

All examples and code given in the manual follow the above two recommendations, i.e., a template based approach and no connection to ERES Server. Unless otherwise noted, all code examples will use a template (templates can be downloaded from corresponding chapters) and will not connect to ERES Server.

Also, note that if you have applets that use the ERES Chart API, the browser must have at least a 1.5 JVM plugin.

### 8.2.3. Interaction with the ERES Server

ERES is generally used in conjunction with the ERES Server. The chart component connects to the ERES Server in order to read and write files, to access databases, and perform data pre-processing required for certain advanced features (such as aggregation). However, the chart component can also be used in a stand-alone mode, in which it performs file I/O and database access directly, without the use of the ERES Server.

Both approaches have their own advantages. If the chart is contained within a JNLP (More about Applets in JNLP: Section 1.3.4 - Run Applets As JNLP), security restrictions prevent it from directly performing file input/output. Database access is also difficult without the presence of a JDBC driver on the client. In such cases, the ERES Server provides the above services to the chart. On the other hand, if the chart is used in an application, there are no such restrictions and performance can be improved by direct access. The application can be run without the necessity of having the ERES Server running at a well-known location.

For instance, the applet or application may be running on machine **Client** and may require data from a database on machine **DbMachine**. However, machine **DbMachine** may be behind a firewall or a direct connection may not be allowed to machine **DbMachine** from machine **Client** due to security restrictions. The ERES Server can be run on machine **Server** and the applet/application can connect to ERES Server on machine **Server**. JDBC can then be used to connect to machine **DbMachine** from machine **Server** and get the data. The data is then delivered to machine **Client** and the chart generated. This is useful when you want to keep the data secure and non-accessible from your client machines and make all connections come through a server machine (a machine running ERES Server). You can also utilize this option to keep a log of all the clients accessing the data through ERES (you can have a log file created when starting the ERES Server. The log file is called `espressmanager.log`). Note that this functionality comes at a cost. You will face a slight performance overhead because your code is connecting to the data through the ERES Server (i.e., another layer has been added).

By default, a chart component requires the presence of the ERES Server. To change the mode, use the `QbChart` class static method at the beginning of your applet/application (before any `QbChart` objects are created):

```
static public void setEspressManagerUsed(boolean b)
```

Both applications and applets can be run with or without accessing the ERES Server. Communication is done using http protocol. The location of the server is determined by an IP address and port number passed in the API code. Given below are instructions on how to connect to the ERES Server.

### 8.2.4. Connecting to ERES Server

If you wish to use Chart API to connect to the ERES Server, you will have to use the following methods:

```
public static void useServlet(boolean b);  
public static void setServletRunner(String comm_url);  
public static void setServletContext(String context);
```

For example, the following lines of code:

```
QbChart.useServlet(true);  
QbChart.setServletRunner("http://someMachine:somePortNumber");  
QbChart.setServletContext("ERES/servlet");
```

will connect to ERES Server running at `http://someMachine:somePortNumber/ERES/servlet`.

Please note that these methods exist in the `QbReport`, `QbChart`, `QbReportDesigner`, and `QbOrganizer` classes.

## 8.2.5. Using the API

The following section details how to utilize the API. Again, the API examples and code is designed using the above recommendation (template based and no ERES Server).

Note that unless otherwise noted, all examples use the Woodview HSQL database, which is located in the `<ERESInstall>/help/examples/DataSources/database` directory. In order to run the examples, you will need to add database HSQL JDBC driver (`hsqldb.jar`) to your classpath. The driver is located in the `<ERESInstall>/WEB-INF/lib` directory.

Also, all the API examples will show the core code in the manual. To compile the examples, make sure the `CLASSPATH` includes `ERESOrganizer.jar`, `ERESServer.jar` and `qblicense.jar`.

For more information on the API methods, please refer the API documentation [ <http://data.quadbase.com/Docs70/eres/help/apidocs/index.html> ].

### 8.2.5.1. Loading a Chart

Charts can be saved to a file using the CHT or TPL formats (both proprietary formats). A TPL file stores all chart information except actual data while a CHT file stores the actual data as well. These formats can be used to reconstruct a chart object. The data is automatically reloaded from the original data source each time the TPL file is opened. When a CHT file is opened, the data used to create it is shown (although the option to fetch data from the data source exists as well).

It is important to note that the TPL file, by default, does **NOT** contain the data. It contains the specified data source along with the chart template information (i.e. the look and feel of the chart). Therefore, when loading a TPL file, the data for the chart is obtained by querying the data source. The CHT file, on the other hand, does **NOT** query the data source when it is opened. It shows the data used to create the chart to begin with. Both formats can be obtained by using Designer or the `export()` method provided in the `QbChart` class.

The following example, which can run as a JNLP (More about Applets in JNLP: Section 1.3.4 - Run Applets As JNLP) or application, reads a CHT file and reconstructs a chart:

```
Component doOpeningChartTemplate(Applet parent) {
    // Do not use EspressoManager
    QbChart.setEspressoManagerUsed(false);

    // Open the template
    QbChart chart = new QbChart(parent, // container
        "OpeningChartTemplate.cht"); // template

    // Show chart in Viewer
    return chart;
}
```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/OpeningChartTemplate.zip> ]

Exported Results [ <http://data.quadbase.com/Docs70/help/manual/code/export/OpeningChartTemplate.png> ]

The constructor in this example is:

```
public QbChart(Applet applet, String file);
```

where applet is the applet container and file is a CHT/TPL file name.



### Note

File names may be specified either as URL strings or using relative/absolute paths. Path names are interpreted as being relative to the current directory of ERES Server or to the current application if ERES Server is not used.

#### 8.2.5.1.1. Parameterized Charts

Charts can also contain parameters to either limit the data in some form. Typically, `query` (or `IN`) parameters are used by the data source to limit the data.

Query parameters can be both single value and multi-value parameter types.

When a parameterized template is opened using following constructor:

```
QbChart(Applet parent, String templateName);
```

a dialog box appears, asking for the value(s) of the parameter(s). This dialog box is the same as the one that appears in Designer when the template is opened.

##### 8.2.5.1.1.1. Object Array

A parameterized chart can also be opened without the dialog box prompting for any value(s). This can be done by passing in an object arrays. Each element in the array represents the value for that particular parameter.

The order of the array must match the order in which the parameters were created in Designer. For correct results, the data type of the value must also match the data type of the parameter.

Query parameters can also be multi-value parameter types. For multi-value parameters, a vector is passed to the object array. The vector contains all the values for the multi-value parameter.

The following example, which can run as a JNLP (More about Applets in JNLP: Section 1.3.4 - Run Applets As JNLP) or application, passes in the parameter values when opening a chart template:

```
Component doObjectArray(Applet parent) {

    // Do not use EspressoManager
    QbChart.setEspressoManagerUsed(false);

    // Object array for Query Parameters
    Vector vec = new Vector();
    vec.add("CA");
    vec.add("NY");

    GregorianCalendar beginDate = new GregorianCalendar(2001, 0, 4);
    GregorianCalendar endDate = new GregorianCalendar(2003, 1, 12);

    long beginLong = beginDate.getTimeInMillis();
    long endLong = endDate.getTimeInMillis();

    Date beginDateTime = new Date(beginLong);
    Date endDateTime = new Date(endLong);
```

```

Object queryParams[] = new Object[3];
queryParams[0] = vec;
queryParams[1] = beginDateTime;
queryParams[2] = endDateTime;

// Open the template
QbChart chart = new QbChart(parent, // container
    "ObjectArray.cht", // template
    queryParams); // Query Parameters

// Show chart in Viewer
return chart;
}

```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/ObjectArray.zip> ]

Exported Results [ <http://data.quadbase.com/Docs70/help/manual/code/export/ObjectArray.png> ]

### 8.2.5.2. Applying a Chart Template

When a `QbChart` object is created from scratch (see Appendix 8.D - Creating the Chart), the chart is created using default attributes. However, you can use a chart template (`.tpl`) to specify user defined attributes during chart construction. Almost all the attributes (except for data source and chart type) are extracted from the template and applied to the `QbChart` object. The template name usually appears as the last argument in the `QbChart` constructors.

You can also specify the template name using the `applyTemplateFile(String fileName)` method in the `QbChart` class.

The following example, which can be run as an applet or application, applies a template onto the `QbChart` object:

```

Component doApplyingTemplate(Applet parent) {

    // Do not use ERES Server
    QbChart.setEspressManagerUsed(false);

    String templateLocation = "..";

    // Apply the template
    QbChart chart = new QbChart (parent, // container
        QbChart.VIEW2D, // chart dimension
        QbChart.BAR, // chart type
        data, // data
        columnMapping, // column mapping
        templateLocation); // template

    // Show chart in Viewer
    return chart;

}

```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/ApplyingChartTemplate.zip> ]

Exported Results [ <http://data.quadbase.com/Docs70/help/manual/code/export/ApplyingChartTemplate.png> ]

Please note that the above code is not complete and is there as a guide. However, the link contains a complete application code that can be run.

You can also take the column mapping from the template and have it applied on the `QbChart` object being created. This is done by passing in `null` instead of a `ColumnInfo` object.

### 8.2.5.3. Modifying Data Source

You can create chart templates in Designer and open those templates using the API. The `QbChart` object created uses the same data source as the template and attempts to fetch the data. However, it is possible to use a template with the correct look and feel with an incorrect data source. The following sections show how to switch the data source, without recreating the entire chart.

Please note that for best results, the number of columns and the data type of each column must match between the two data sources (i.e. the one used to create the template in Designer and the new data source).

After switching the data source, the `QbChart` object must be forced to fetch the new data. This can be done by calling the refresh method in the `QbChart` class.

#### 8.2.5.3.1. Data from a Database

Switching the data source to point to a database is simple. All you would need to do is provide the database connection information as well as the query to be used and pass that to the `QbChart` object. You can provide the database connection information (as well as the query) in a `DBInfo` object (for more information on creating a `DBInfo` object, please refer to Appendix 8.A.1 - Data from a Database).

The following example, which can be run as an applet or application, switches the data source of the `QbChart` object to a database:

```
Component doChartSwitchToDatabase(Applet parent) {

    // Do not use EspressoManager
    QbChart.setEspressoManagerUsed(false);

    // Data Source
    DBInfo newDatabaseInfo = new DBInfo(..);

    // Open the template
    QbChart chart = new QbChart(parent, // container
        "ChartSwitchToDatabase.tpl"); // template

    try {
        // Switch data source
        chart.getInputData().setDatabaseInfo(newDatabaseInfo);

        // Refresh chart
        chart.refresh();
    } catch (Exception ex)
    {
        ex.printStackTrace();
    }

    // Show chart in Viewer
    return chart;
}
```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/ChartSwitchToDatabase.zip> ]

Exported Results [ <http://data.quadbase.com/Docs70/help/manual/code/export/ChartSwitchToDatabase.png> ]

Please note that the above code is not complete and is there as a guide. However, the link contains a complete application code that can be run.

**8.2.5.3.1.1. JNDI**

You can also change the data source to a JNDI data source. This is done by specifying the JNDI connection information in a DBInfo object and then passing it to the QbChart object.

The following example, which can be run as an applet or application, switches the data source of the QbChart object to a JNDI database:

```
Component doChartSwitchToDatabaseJNDI(Applet parent) {

    // Do not use EspressoManager
    QbChart.setEspressoManagerUsed(false);

    // Data Source.  Replace comp with computer and env with environment.
    // The environment hashtable is empty for tomcat.
    // If other application server is used, need to set INITIAL_CONTEXT_FACTORY
    // and PROVIDER_URL.
    DBInfo newDatabaseInfo = new DBInfo(...);

    // Open the template
    QbChart chart = new QbChart (parent, // container
    "ChartSwitchToDatabaseJNDI.cht"); // template

    try {
        // Switch data source
        chart.getInputData().setDatabaseInfo(newDatabaseInfo);

        // Refresh chart
        chart.refresh();
    } catch (Exception ex)
    {
        ex.printStackTrace();
    }

    // Show report in Viewer
    return chart;
}
```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/ChartSwitchToDatabaseJNDI.zip> ]

Exported Results [ <http://data.quadbase.com/Docs70/help/manual/code/export/ChartSwitchToDatabaseJNDI.png> ]

Please note that the above code is not complete and is there as a guide. However, the link contains a complete application code that can be run. Note that the Woodview database needs to be set up as a JNDI data source in the Tomcat environment and the application code changed to match the connection information in order for the application code to run.

**8.2.5.3.2. Data from a Data File (TXT/DAT/XML)**

You can switch the data source to a text file as long as the text file follows the Quadbase guidelines (for more details, please refer to Appendix 8.A.2 - Data from a Data File (TXT/DAT/XML)).

The following example, which can be run as an applet or application, switches the data source of the QbChart object to a text file:

```
Component doChartSwitchToDataFile(Applet parent) {

    // Do not use EspressoManager
```

---

```

QbChart.setEspressManagerUsed(false);

// Open the template
QbChart chart = new QbChart(parent, // container
    "ChartSwitchToDataFile.cht"); // template

try {
    // Switch data source
    chart.gethInputData().setDataFile("surface.dat");

    // Refresh chart
    chart.refresh();
} catch (Exception ex)
{
    ex.printStackTrace();
}

// Show chart in Viewer
return chart;
}

```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/ChartSwitchToDataFile.zip> ]

Please note that the above code is not complete and is there as a guide. However, the link contains a complete application code that can be run.

### 8.2.5.3.3. Data from an XML Data Source

You can switch the data source to your custom XML data as long as there is a .dtd or .xml schema accompanying your data. The XML data information is specified (for more details, please refer to Appendix 8.C.3 - Data from a XML Data Source) and then passed to the QbChart object.

The following example, which can be run as an applet or application, switches the data source of the QbChart object to XML data:

```

Component doChartSwitchToXMLData(Applet parent) {

    // Do not use EspressManager
    QbChart.setEspressManagerUsed(false);

    // Open the template
    QbChart chart = new QbChart(parent, // container
        "ChartSwitchToXMLData.cht"); // template

    // XML data source information
    String xmlfilename = "Inventory.xml";
    String xmlcondition = "";

    XMLFileQueryInfo xmlInfo = new XMLFileQueryInfo(...);

    try {
        // Switch data source
        chart.gethInputData().setXMLFileQueryInfo(xmlInfo);

        // Refresh chart
        chart.refresh();
    } catch (Exception ex)
    {

```

```

    ex.printStackTrace();
}

// Show chart in Viewer
return chart;
}

```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/ChartSwitchToXMLData.zip> ]

Exported Results [ <http://data.quadbase.com/Docs70/help/manual/code/export/ChartSwitchToXMLData.png> ]

Please note that the above code is not complete and is there as a guide. However, the link contains a complete application code that can be run.

#### 8.2.5.3.4. Data from Custom Implementation

In addition to the regular data sources, you can also pass in your own custom data. The custom data is passed to the `QbChart` object using the `IDataSource` interface (for more details, please refer to Appendix 8.A.5 - Data passed in a Custom Implementation).

The following example, which can be run as an applet or application, switches the data source to a custom implementation:

```

Component doChartSwitchToCustomData(Applet parent) {

    // Do not use EspressoManager
    QbChart.setEspressoManagerUsed(false);

    // Open the template
    QbChart chart = new QbChart(parent, // container
        "ChartSwitchToCustomData.cht"); // template

    try {
        // Switch data source
        chart.getInputData().setClassFile("Furniture_Chart");

        // Refresh chart
        chart.refresh();
    } catch (Exception ex)
    {
        ex.printStackTrace();
    }

    // Show chart in Viewer
    return chart;
}

```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/ChartSwitchToCustomData.zip> ]

Exported Results [ <http://data.quadbase.com/Docs70/help/manual/code/export/ChartSwitchToCustomData.png> ]

Please note that the above code is not complete and is there as a guide. However, the link contains a complete application code that can be run.

#### 8.2.5.3.5. Data passed in an Array in Memory

You can also pass in data using arrays. The array data is usually stored in memory and passed to the `QbChart` object (for more details, please refer to Appendix 8.A.4 - Data passed in an Array in Memory).

The following example, which can be run as an applet or application, switches the data source to an array in memory:

```
Component doChartSwitchToArrayData(Applet parent) {

    // Do not use EspressoManager
    QbChart.setEspressoManagerUsed(false);

    DbData newData = new DbData(...);

    ColInfo colInfo = new ColInfo();
    colInfo.category = 0;
    colInfo.value = 1;
    colInfo.subvalue = 2;

    // Open the template
    QbChart chart = new QbChart(parent, // container
        "ChartSwitchToArrayData.cht"); // template

    try {
        // Switch data source
        chart.getInputData().setData(newData);

        // Refresh chart
        chart.refresh();
    } catch (Exception ex)
    {
        ex.printStackTrace();
    }

    // Show chart in Viewer
    return chart;
}
```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/ChartSwitchToArrayData.zip> ]

Exported Results [ <http://data.quadbase.com/Docs70/help/manual/code/export/ChartSwitchToArrayData.png> ]

Please note that the above code is not complete and is there as a guide. However, the link contains a complete application code that can be run.

## 8.2.5.4. Modifying Chart Attributes

ERES has hundreds of properties, which provide a fine control over various elements of a chart. In order to facilitate ease-of-use, most properties have been categorized into groups and exposed in the form of interfaces. An application first obtains a handle to a group interface using a `gethXXX` method, and then manipulates the chart's properties directly by calling methods on that interface. Most interfaces are contained in the `quadbase.util` [ <http://data.quadbase.com/Docs70/eres/help/apidocs/quadbase/util/package-summary.html> ] and `quadbase.ChartAPI` packages [ <http://data.quadbase.com/Docs70/eres/help/apidocs/quadbase/ChartAPI/package-summary.html> ].

### 8.2.5.4.1. Modifying Color, Font, Etc

A chart is comprised of different elements (for instance, the canvas, the axis, the legend, the chart data, etc). Each element can be modified independently of the other by getting the appropriate interface and changing the properties by calling the methods therein.

For instance, the following code sets the background color of the chart to white:

```
chart.gethCanvas().setBackgroundColor(Color.white);    // chart being an
object of type QbChart
```

while the code given below changes the color of the X axis to black:

```
chart.getAxis().setColor(Color.black);           // chart being an
object of type QbChart
```

You can also set the color of the data elements of the chart based on the series or based on the category (if no series is defined). For example, if the chart is a Column chart with five elements in the series, the following code sets the columns colors in the series, to be yellow, orange, red, green, and blue.

```
Color dataColors[] = {Color.yellow, Color.orange, Color.red, Color.green,
    Color.blue};
chart.getDataPoints().setColors(dataColors);     // chart being an
object of type QbChart
```



### Note

The number of colors defined **must** match the number of unique elements in the series (or the category if the series is not defined).

Similarly, the font styles can be set by calling the appropriate interface and using the method. The following code sets the text used in the legend to be of Arial, bold type, and size 15.

```
Font legendFont = new Font("Arial", Font.BOLD, 15);
chart.getLegend().setFont(legendFont);          // chart being an object
of type QbChart
```

while the following code sets the font and color of the labels used in the Y Axis:

```
Font YAxisFont = new Font("Helvetica", Font.ITALIC, 15);
chart.getAxis().getLabel().setFont(YAxisFont);  // chart being an
object of type QbChart
chart.getAxis().getLabel().setColor(Color.blue); // chart being an
object of type QbChart
```

Similarly, other properties can be set using the methods in the interfaces.

#### 8.2.5.4.2. Setting Predefined Patterns

The class `QbPattern` is a subclass of `java.awt.Color`, so there is no new method introduced to set patterns explicitly. You create a `QbPattern` object first, and then use it as if it is a `Color` object. The pattern feature is only applicable to data points. For other chart elements (such as axis, canvas), ERES will ignore the `QbPattern` properties, and use it just like a `Color`.

The following example, which can be run as an applet or application, shows how to set a pattern to a data point:

```
Color[] dataColors = chart.getDataPoints().getColors();
QbPattern dataPattern = new QbPattern(colors[2],
    QbChart.PATTERN_THICK_FORWARD_DIAGONAL);
```

```
dataColors[2] = dataPattern;
chart.gethDataPoints().setColors(dataColors);
```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/ChartSetPredefinedPattern.zip> ]

Exported Results [ <http://data.quadbase.com/Docs70/help/manual/code/export/ChartSetPredefinedPattern.png> ]

There are totally 30 predefined patterns available for you to use. The pattern ID range from 0 to 29. If the user passes an integer beyond this range, it will be considered as ID = 0, which is a solid color block. The Pattern ID is defined in `quadbase.ChartAPI.IMiscConstants` [ <http://data.quadbase.com/Docs70/eres/help/apidocs/quadbase/ChartAPI/IMiscConstants.html> ] class as well as in `quadbase.common.swing.color.PatternImage` class. Since `QbChart` implements the `IMiscConstants` interface, you can obtain these IDs directly from `QbChart`. You can also see what each pattern looks like in the pattern palette Designer.

#### 8.2.5.4.3. Setting Customized Patterns

You can also set up your own pattern (texture) images. This feature is only available via API and the modification will not be saved to the template. This feature is mainly for those who want to modify the chart during the run time.

The following example, which can be run as an applet or application, shows how to set a custom pattern to a data point:

```
Color[] dataColors = chart.gethDataPoints().getColors();
QbPattern dataPattern = new QbPattern(dataColors[2]);
File customImageFile = new File("Quadbase_Logo.png");
BufferedImage customImage = ImageIO.read(customImageFile);
dataPattern.setPatternImage(customImage);
dataColors[2] = dataPattern;
chart.gethDataPoints().setColors(dataColors);
```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/ChartSetCustomPattern.zip> ]

Exported Results [ <http://data.quadbase.com/Docs70/help/manual/code/export/ChartSetCustomPattern.png> ]

In the above code, a `QbPattern` object is created without specifying the pattern ID. The newly created object is equivalent to a `java.awt.Color` object. The developer is responsible for providing an image so as to avoid the `NullPointerException` prompting in the procedure. An existing image (from a file) is then passed as the pattern for data point two.

#### 8.2.5.4.4. Modifying Size

There are two sizes to be considered when creating a chart:

**Relative Size:** This size is defined relative to the chart canvas. For instance, the chart plot height might be set as `.7` and the chart plot width set as `.8`. In the case where the canvas is 300 by 300 pixels, the height of the chart plot becomes 210 (`.7 x 300`) pixels and the width becomes 240 pixels (or `.8 x 300`). If the canvas size is increased to 500 by 600 pixels, the height of the chart plot becomes 420 pixels (`.7 * 600`) and the width becomes 400 pixels (`.8 * 500`). Relative sizes depend on the canvas height and width.

**Absolute Size:** This size denotes an absolute size of the canvas in pixels.

Every element of the chart (where the size parameter is used) is defined relative to the chart canvas. A float is used to represent the relative size of the chart element. For instance, the following code sets the chart plot height to `.85` and the chart plot width to `.65`:

```
chart.gethChartPlot().setRelativeHeight(.85f); // chart being an object
of type QbChart
```

```
chart.getChartPlot().setRelativeWidth(.65f); // chart being an object
of type QbChart
```

The code given below sets the canvas size of the chart:

```
Dimension canvasSize = new Dimension(500, 450);
chart.getCanvas().setSize(canvasSize); // chart being an object
of type QbChart
```

Similarly, the sizes of the other elements can be set using the methods in the interfaces.

#### 8.2.5.4.5. Modifying Date/Time Zoom Charts

You can modify the properties (such as scale and starting/ending time period) of a Date/Time Zoom template created in the Designer. This is done by getting a handle to the Zoom properties (using the `IZoomInfo` interface) and using the various methods there to change the presentation.

The following example, shows how to modify a Date/Time Zoom chart and can be run as an applet or application:

```
// Modify starting and ending period and scale
IZoomInfo zoomInfo = chart.getZoomInfo();

// Begin Date
Calendar beginDate = new GregorianCalendar(2001, 0, 1);

// End Date
Calendar endDate = new GregorianCalendar(2003, 11, 31);

zoomInfo.setLowerBound(beginDate.getTime());
zoomInfo.setUpperBound(endDate.getTime());

zoomInfo.setScale(6, IZoomInfo.MONTH);

chart.refresh();
```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/ZoomChart.zip> ]

Exported Results [ <http://data.quadbase.com/Docs70/help/manual/code/export/ZoomChart.png> ]

In the above code, `chart` is an object of type `QbChart`.



#### Note

After modifying the chart's zoom properties, the chart **must** be refreshed for the modified properties to take effect.

#### 8.2.5.4.6. Modifying Chart in Report

While stand-alone charts can be modified directly using the API, any charts in a report (whether the chart is independent or not) cannot be modified as easily. You can change the look and feel of a chart (in a report) by implementing `IChartModifier` interface to create a class that invokes the desired changes. Note that the actual chart template is not changed, merely the chart's appearance within the report.

The following example, which can be run as an applet or application, shows how to modify a chart in a report:

```

ReportChartObject[] reportCharts = report.getReportChartObjects();
for (int i = 0; i < reportCharts.length; ++i)
{
    reportCharts[i].setChartModifier(new ModifyChart());
}

public class ModifyChart implements IChartModifier {
    public IChart modifyChart(Object chartInfo) {
        ChartObject chart = new ChartObject(chartInfo); // Get actual
        ChartObject
        chart.getCanvas().setBackgroundColor(Color.white);
        return chart; }
}

```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/ExampleIChartModifier.zip> ]

Exported Results Before Modification [ [http://data.quadbase.com/Docs70/help/manual/code/export/IChartModifier\\_Before.pdf](http://data.quadbase.com/Docs70/help/manual/code/export/IChartModifier_Before.pdf) ]

Exported Results After Modification [ [http://data.quadbase.com/Docs70/help/manual/code/export/IChartModifier\\_After.pdf](http://data.quadbase.com/Docs70/help/manual/code/export/IChartModifier_After.pdf) ]

### 8.2.5.5. Exporting the Chart

Any charts included in the report can be exported into JPEG, GIF and PNG formats as well; although by default, the charts in the report are exported as JPEG's. The format for the chart can be changed, when creating the `ReportChartObject` object, by specifying the image type use the method `setImageType(int)`.

The ERES API has the capability to export stand-alone charts in a variety of formats. These include GIF, JPEG, PNG, BMP, and PDF formats. In addition, a chart may be exported to the proprietary `.cht` or `.tpl` formats. A `.cht` stores all the chart information and can be considered the ERES equivalent of a static image file. A `.cht` file differs from a static image file in that its data can be refreshed from the data source and it allows additional functionality such as zooming, drill-down, etc. A `.tpl` file is identical to a `.cht` file except it does not store actual data. For a `.tpl` file, the data is automatically reloaded from the original data source at the time of chart reload. Both `.cht` and `.tpl` files can be viewed using the ERES Viewer or ERES API. When using stand-alone charts, you can specify the format by creating a `QbChart` object and using the various export methods within the class.

There are different methods available for exporting a stand-alone chart in a variety of formats with various options. The simplest method would be:

```
public export(int format, String filename)
```

In the above method, `format` is one of the format constants and `filename` is the output filename (with or without an extension).

The following is a list of the constants used for exporting the chart components:

<b>Bitmap:</b>	<code>QbChart.BMP</code>
<b>GIF:</b>	<code>QbChart.GIF</code>
<b>JPEG:</b>	<code>QbChart.JPEG</code>
<b>PNG:</b>	<code>QbChart.PNG</code>

---

<b>PDF:</b>	QbChart.PDF
<b>Flash:</b>	QbChart.FLASH
<b>Scalable Vector Graphics:</b>	QbChart.SVG
<b>Text Data:</b>	QbChart.TXTFORMAT
<b>XML Data:</b>	QbChart.XMLFORMAT
<b>Windows Meta File:</b>	QbChart.WMF

Depending on the format selected, additional options are also available. For example, exporting a chart object as a jpeg gives you the choice of the image quality (represented as number from 0 - 99) while export a chart object as a PNG gives you the choice of specifying the compression used. The options can be specified using the following method:

```
public export(int format, String filename, int option)
```

The following code, which can be run as an applet or application, shows how to construct and export a chart:

```
// Open the template
QbChart chart = new QbChart(parent, // container
    "ExportChart.cht"); // template

try {
    chart.export(QbChart.PNG, "ExportChart");
} catch (Exception ex){
    ex.printStackTrace();
}
}
```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/ExportChart.zip> ]

Exported Results [ <http://data.quadbase.com/Docs70/help/manual/code/export/ExportChart.png> ]

Please note that when you export the chart to a text file, the default delimiter is a tab space. However, you can set another delimiter (available delimiters are “;”, “:” and “”) by using the following method:

```
chart.exportDataFile("TextData", QbChart.COMMA, QbChart.TXTFORMAT); //
    where chart is an object of type QbChart
```

If you plan on exporting the chart object without viewing it, setting the following static method to true would improve performance slightly:

```
QbChart.setForExportOnly(boolean);
```

Calling this method before constructing the QbChart object would result in better performance.

You can also export the QbChart object to a byte array using the following method:

---

```
public byte[] exportChartToByteArray();
```

This will give the `.cht` equivalent in the form of a byte array. This is useful if you need to serialize the `QbChart` object.

### 8.2.5.5.1. Record File Exporting

ERES also has record file exporting to handle large amounts of data. In record file exporting, you specify the number of records to be held in memory and a `temp` directory. When the number of records in the data exceeds the number specified to be held in memory, the data is stored on disk in the `temp` directory specified.

There are certain conditions that have to be met before you can use this feature. You must:

- make sure that ERES Server is not used;
- make sure that the data is NOT from a XML source;
- make sure that the data is sorted;

To use record file export, the following code must be added before calling the `QbChart` constructor:

```
QbChart.setEspressManagerUsed(false);
QbChart.setTempDirectory(<specify the temp directory to store data. Default
is ./temp>);
QbChart.setMaxCharForRecordFile(<specify the maximum character length.
Default is 40>);
QbChart.setMaxRecordInMemory(<specify the maximum number of rows to be kept
in memory. Default is -1 i.e., store all records in memory>);
QbChart.setFileRecordBufferSize(<specify the number of rows to be retrieved
at one time from disk. Default is 10,000>);
```

### 8.2.5.5.2. Streaming Charts

In addition to exporting to local drives, charts can also be exported as a byte stream and streamed directly to a web browser. Note that server-side code typically exports the image only in a binary format. You are responsible for creating a wrapper (i.e. DHTML content) around the exported image.

An example that exports a chart to PNG and streams it to the browser is given below. In order to run the example, you will need to configure and compile the source code, then deploy the class file in the servlet directory of your servlet runner. Replace the `chartTemplate` variable with either an absolute path or a path relative to the working directory of your application server.

```
public void doGet(HttpServletRequest req, HttpServletResponse resp) throws
ServletException,
IOException {

    // Do not use EspressManager
    QbChart.setEspressManagerUsed(false);

    String chartTemplate = "StreamingChart.cht";

    // Open template
    QbChart chart = new QbChart((Applet) null, chartTemplate);

    // Export the chart to PNG and stream the bytes
    ByteArrayOutputStream chartBytes = new ByteArrayOutputStream();
```

```

try {
    // Export chart
    chart.export(QbChart.PNG, chartBytes);
} catch (Exception ex){
    ex.printStackTrace();
}

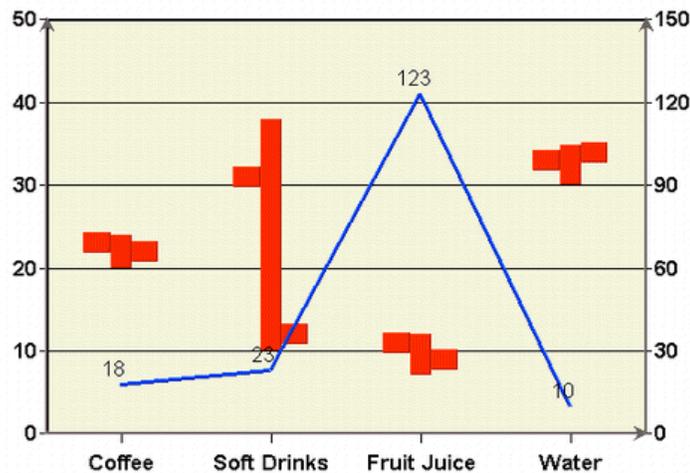
resp.setContentType("image/x-png");
resp.setContentLength(chartBytes.size());

OutputStream toClient = resp.getOutputStream();
chartBytes.writeTo(toClient);
toClient.flush();
toClient.close();
}

```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/StreamingChart.zip> ]

To run the example, make sure to copy the chart template to the location accessed by the code. If using a relative path (as shown in the example), remember that the current directory is the working directory of your application server. For example, since the working directory in Tomcat is `<Tomcat>/bin`, you will need to create the directory `<Tomcat>/templates/` and copy the chart template there to run the code. The resulting chart is shown below.



*Streaming Chart*

### 8.2.5.6. Changing Chart Viewer Options

At times, you may want to configure what users can or cannot do when they are viewing the chart by using the Chart Viewer.

When the user is using the Viewers to view the report, right clicking on the report causes a menu to pop up. Using this menu, the user can navigate through different pages of the report, as well as performing other tasks on the report, such as exporting it as a DHTML or PDF file. This can easily be done using the Report Viewer and Page Viewer APIs. However, these powerful features of the report viewer can be too complicated or overly confusing for the average user. Therefore, a few API methods have been introduced to control what options are available in the pop-up menu of the Report/Page Viewer.

These API calls are:

```
viewer.setMenuVisible(boolean b);
viewer.setPageMenuVisible(boolean b);
viewer.setPageMenuItemVisible(String[] items, boolean b);
viewer.setOutputMenuVisible(boolean);
viewer.setOutputMenuItemVisible(String[] items, boolean b);
viewer.setRefreshMenuItemVisible(boolean b);
viewer.setGoToMenuItemVisible(boolean b);
viewer.setSortMenuVisible(boolean b);
```

For more detailed information on these API methods, please consult the Javadoc.

## 8.2.6. API Only Features

While ERES API can reproduce all the functionality of Chart Designer, the API also has additional functionalities. These additional features will be described below. Please note that some of the features are for stand-alone charts only.

Note that in the snippets of code provided, chart is an object of type `QbChart`.

### 8.2.6.1. Visual

The features, shown below, deal with the presentation of the chart and its components. Each feature below visually changes the chart in some manner. These changes are also carried forth when exported to a static image.

#### 8.2.6.1.1. Canvas/Plot

The following features describe the various visual changes that can be made to the chart plot and/or canvas using the API. Note that the features below deal with general changes to the chart plot and/or canvas. Any component specific change is described in its own section.

##### 8.2.6.1.1.1. Customizable Message for No-Data-In-Plot

ERES allows the message, which gets displayed when there is no data or not enough data to be plotted, to be changed. This can be done by getting a handle to the `INoDataToPlotMessage` and specifying the new message.

```
INoDataToPlotMessage noData = chart.getNoDataToPlotMessage();
noData.setMessage("Not enough data to plot chart");
```

Please refer to the online API documentation for more information (`quadbase.util.INoDataToPlotMessage` [ <http://data.quadbase.com/Docs70/eres/help/apidocs/quadbase/util/INoDataToPlotMessage.html> ]).

##### 8.2.6.1.1.2. Set Chart to Fit Canvas

ERES allows charts to be resized and fit into the canvas correctly, taking into account all the text and labels associated with the chart. To correctly fit a chart onto its canvas, use the method `setFitOnCanvas(boolean b)` in the `ICanvas` interface.

```
ICanvas canvas = chart.getCanvas();
canvas.setFitOnCanvas(true);
```

Please refer to the online API documentation for more information (`quadbase.util.ICanvas` [ <http://data.quadbase.com/Docs70/eres/help/apidocs/quadbase/util/ICanvas.html> ]).

##### 8.2.6.1.1.3. Set Chart Invisible

A chart can be made invisible so that only the plot data in table form is shown. This is accomplished by getting a handle to the `ICanvas` interface and using the `setChartVisible` method.

```
ICanvas canvas = chart.getCanvas();
canvas.setChartVisible(false);
```

Please refer to the online API documentation for more information (quadbase.util.ICanvas [ <http://data.quadbase.com/Docs70/eres/help/apidocs/quadbase/util/ICanvas.html> ]).

#### 8.2.6.1.1.4. Applying Different Graphics Rendering

ERES allows you to apply different rendering techniques (such as anti-aliasing) to the chart. This allows you to draw the chart to your specifications.

To utilize this feature in the API, call the `setRenderingHint` method in `QbChart` and pass in the hint key and hint value parameters.

```
chart.setRenderingHint( java.awt.RenderingHints.KEY_ANTIALIASING,
    java.awt.RenderingHints.VALUE_ANTIALIAS_ON );
```

You can also specify the horizontal text to be anti-aliased only.

```
chart.forceApplyAntiAliasToHorizontalText(true);
```

Please refer to the online API documentation for more information (quadbase.ChartAPI.QbChart [ <http://data.quadbase.com/Docs70/eres/help/apidocs/quadbase/ChartAPI/QbChart.html> ]).

This functionality is only available for stand-alone charts.

#### 8.2.6.1.1.5. Chart Plot Position

ERES allows the chart position to be placed down to  $-0.5$  (X and Y position relative to the canvas). This allows the chart to be placed right along the edge of the canvas.

To utilize this feature using the API, get a handle to `IPlot` and use the `setPosition` method.

```
IPlot chartPlot = chart.getChartPlot();
chartPlot.setPosition(new Position(-0.5, -0.5));
```

Please refer to the online API documentation for more information (quadbase.util.IPlot [ <http://data.quadbase.com/Docs70/eres/help/apidocs/quadbase/util/IPlot.html> ]).

#### 8.2.6.1.1.6. Drawing Multiple Charts in same Plot

ERES allows multiple charts to be overlapped and shown one above the other (using the primary chart's axes). The canvas of the primary chart is used and therefore 2D charts cannot be added to 3D charts and vice versa. All charts overlapped onto the primary chart will have any text and canvas information removed. Please note that since resulting chart will use the primary chart's category and scale, add on charts must also use the similar categories and scales to be displayed correctly. For Scatter and Surface charts, both X and Y axis scales should be taken into consideration.

To utilize this feature using the API, you would use the `setAddOnChart` method in `QbChart`.

The following code, which can be run as an applet or application, shows how to overlap three charts in one plot:

```
// Open the template
QbChart primaryChart = new QbChart(parent, // container
    "AddOnChart1.cht"); // template
```

```
// Open other two templates
QbChart chart2 = new QbChart(parent, "AddOnChart2.cht");
QbChart chart3 = new QbChart(parent, "AddOnChart3.cht");

primaryChart.setAddOnChart(new QbChart[] { chart2, chart3 });
```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/AddOnChart.zip> ]

Exported Results [ <http://data.quadbase.com/Docs70/help/manual/code/export/AddOnChart.png> ]

Please refer to the online API documentation for more information (quadbase.ChartAPI.QbChart [ <http://data.quadbase.com/Docs70/eres/help/apidocs/quadbase/ChartAPI/QbChart.html> ]).

### 8.2.6.1.2. Hint Box

The following features describe the various visual changes that can be made to the chart hint box using the API. These include modifying the content as well as look of the hint box.

#### 8.2.6.1.2.1. Modify Hint Box

ERES allows the Hint Box to be modified, i.e. you can dictate what the Hint Box says when it is viewed. This is done by creating a class that implements `IHintBoxInfo` and assigning that class using the method `setHintBoxInfo` in `IHint`.

The following code, which can be run as an applet or application, shows how to modify the hint box info:

```
// Open the template
QbChart chart = new QbChart(parent,
    // container
    "../templates/ModifyHintBox.cht"); // template

IHintBoxInfo hintBoxInfo = new Hint();
chart.gethDataPoints().gethHint().setHintBoxInfo(hintBoxInfo);

...

class Hint implements IHintBoxInfo {

    public Vector getHint(PickData pickData) {
        Vector vec = new Vector();

        if (pickData.series != null)
            vec.addElement("(" + pickData.seriesName + ") " +
                pickData.s_series);

        if (pickData.category != null)
            vec.addElement("(" + pickData.categoryName + ") " +
                pickData.s_category);

        if (pickData.s_value != null)
            vec.addElement("(" + pickData.valueName + ") " +
                pickData.s_value + (pickData.value > 7 ? ":Good" : ":Restock"));

        return vec;
    }
}
```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/ModifyHintBox.zip> ]

Exported Results [ <http://data.quadbase.com/Docs70/help/manual/code/export/ModifyHintBox.png> ]

Please refer to the online API documentation for more information (quadbase.util.IHintBoxInfo [ <http://data.quadbase.com/Docs70/eres/help/apidocs/quadbase/util/IHintBoxInfo.html> ]).

This functionality is only available for stand-alone charts.

#### 8.2.6.1.2.2. Data and Hyperlink Hint Box Offset

ERES allows an offset to be set for the Data and the HyperLink hint box so that it does not overlap the chart or any other component in the chart.

To utilize this feature in the API, get a handle to `IHint` (from either the chart object or the `IHyperLinkSet` object) and then use the `setOffset` method.

```
IHint dataHints = chart.gethDataPoints().gethHint();
dataHints.setoffset(new Dimension (30, 20));
```

Please refer to the online API documentation for more information (quadbase.util.IHint [ <http://data.quadbase.com/Docs70/eres/help/apidocs/quadbase/util/IHint.html> ]).

#### 8.2.6.1.2.3. Hint Box Border Color

ERES allows the color of the Hint box border to be set using the API. This can be done by getting a handle to `IHint` and using the `setBorderColor` method.

```
IHint chartHintBox = chart.gethHint();
chartHintBox.setBorderColor(Color.red);
```

Please refer to the online API documentation for more information (quadbase.util.IHint [ <http://data.quadbase.com/Docs70/eres/help/apidocs/quadbase/util/IHint.html> ]).

#### 8.2.6.1.2.4. Customize Image Map Hint Box

ERES allows the customization of the hint box text when exporting the chart to a map file. When the map file is included in a DHTML file, the customized hint box is visible when the mouse is moved over the data points of the chart. You can create a customized image map hint box by creating a class that implements `ICustomized-ImageDataMapHintBox` and assigning that class to the `setImageMapDataHintBoxHandle` method in `QbChart`.

The following code, which can be run as an applet or application, shows how to customize the image map hint box:

```
// Open the template
QbChart chart = new QbChart(parent, // container
    "CustomizeImageMapHintBox.cht"); // template

chart.setImageMapDataHintBoxHandle(new customizeImageMap());

try {
    // Export chart to image and map file
    chart.export(QbChart.PNG, "CustomizeImageMapHintBox", "CustomizeImageMapHintBox",
        0, 0,
        true);
} catch (Exception ex)
```

```

{
    ex.printStackTrace();
}

...

class customizeImageMap implements ICustomizeImageMapDataHintBox {

    public String customizeHintBox(String str) {
        return str.substring(8, 11) + " " + str.substring(str.length() - 3);
    }
}

```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/CustomizeImageMapHintBox.zip> ]

Exported Map [ <http://data.quadbase.com/Docs70/help/manual/code/export/CustomizeImageMapHintBox.map> ]

Exported Image [ <http://data.quadbase.com/Docs70/help/manual/code/export/CustomizeImageMapHintBox.html> ]

Please refer to the online API documentation for more information ([quadbase.util.ICustomizeImageMapDataHintBox](http://data.quadbase.com/Docs70/eres/help/apidocs/quadbase/util/ICustomizeImageMapDataHintBox.html) [ <http://data.quadbase.com/Docs70/eres/help/apidocs/quadbase/util/ICustomizeImageMapDataHintBox.html> ]).

### 8.2.6.1.3. Legend/Annotation

The following features describe the various visual changes that can be made to the chart legend and/or any annotation using the API. These include modifying the content as well as look of the chart legend/annotations.

#### 8.2.6.1.3.1. Annotation with Symbol

ERES allows symbols to be included with an `Annotation` thus allowing symbols and strings to be placed in the chart. One of the applications of this feature is to create your own legend in place of the default legend created by ERES.

A new constructor has been created for `IAnnotation`, which can be used for adding symbols and text.

The following code, which can be run as an applet or application, shows how to combine symbols with an `Annotation`:

```

// Open the template
QbChart chart = new QbChart(parent,
    // container
    "../../../../templates/AnnotationWithSymbol.cht"); //

template

// Create custom legend
IAnnotationSet set = chart.getAnnotations();
String[] text = {"New Legend", "Hello World", "ABC", "I got It"};
int[] shape = {QbChart.PLUS, QbChart.NOSYMBOL, QbChart.SQUARE,
    QbChart.DASH};
Color[] color = {Color.red, Color.black, Color.blue, Color.white};
IAnnotation anno = set.newAnnotation(text, shape, color);
Point_2D newPosition = new Point_2D(.7f, .7f);
anno.setRelativePosition(newPosition);
set.addAnnotation(anno);

```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/AnnotationWithSymbol.zip> ]

Exported Image [ <http://data.quadbase.com/Docs70/help/manual/code/export/AnnotationWithSymbol.png> ]

Please refer to the online API documentation for more information (quadbase.util.IAnnotationSet [ <http://data.quadbase.com/Docs70/eres/help/apidocs/quadbase/util/IAnnotationSet.html> ] and quadbase.util.IAnnotation [ <http://data.quadbase.com/Docs70/eres/help/apidocs/quadbase/util/IAnnotation.html> ]).

#### 8.2.6.1.3.2. Set Relative Shift of Annotation Border

ERES allows shift of Annotation border from Relative Position of Annotation and Annotation text. The following methods allows you to set Shift in x,y axis in pixels.

```
void setxShift(int x);
void setyShift(int y);
```

Please refer to the online API documentation for more information (quadbase.util.IAnnotation [ <http://data.quadbase.com/Docs70/eres/help/apidocs/quadbase/util/IAnnotation.html> ]).

#### 8.2.6.1.3.3. Set Reference Position of Legend and Annotation Text

ERES allows the reference positions of legend and any annotation texts to be set at either the upper left corner or default position (lower left corner). To change the reference position to the upper left position, you would use the setReferenceAtTop method in ICanvas.

```
ICanvas canvas = chart.getCanvas();
canvas.setReferenceAtTop(true);
```

Please refer to the online API documentation for more information (quadbase.util.ICanvas [ <http://data.quadbase.com/Docs70/eres/help/apidocs/quadbase/util/ICanvas.html> ]).

#### 8.2.6.1.4. Misc.

The following features describe the various visual changes that can be made to the different components of the chart. These include modifying both the content as well as the look of the chart and its elements.

##### 8.2.6.1.4.1. Ticker Label Replacement

ERES allows ticker labels to be replaced by user-defined strings. This enables the users to put in their own ticker values.

To replace ticker labels, use the method setTickerLabels in IAxis.

```
IAxis hXAxis=chart.gethXAxis();
hXAxis.setTickerLabels(new String[] {new String("a"), new String("b"), new
String("c")});
```

Please refer to the online API documentation for more information (quadbase.util.IAxis [ <http://data.quadbase.com/Docs70/eres/help/apidocs/quadbase/util/IAxis.html> ]).

##### 8.2.6.1.4.2. Customizable Data Top Label

ERES allows data top labels for both primary and secondary charts to be customized. You can do this by creating a class that implements IDataLabelInfo and passing that class using the setDataLabelInfo method in IDataPointSet.

The following code, which can be run as an applet or application, shows how to combine symbols with an Annotation:

```
// Open the template
QbChart chart = new QbChart(parent, // container
"CustomizeDataTopLabel.cht"); // template
```

```

chart.gethDataPoints().setDataLabelInfo(new customizeDataTopLabel());

...

class customizeDataTopLabel implements IDataLabelInfo {

    public String getDataLabel(PickData pickData, String
originalDataLabel) {
        return (pickData.toString());
    }
}

```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/CustomizeDataTopLabel.zip> ]

Exported Image [ <http://data.quadbase.com/Docs70/help/manual/code/export/CustomizeDataTopLabel.png> ]

Note that any customization to the data top labels will not be evident unless they are visible.

Please refer to the online API documentation for more information (quadbase.util.IDataLabelInfo [ <http://data.quadbase.com/Docs70/eres/help/apidocs/quadbase/util/IDataLabelInfo.html> ]).

#### 8.2.6.1.4.3. Show Value Axis as Date/Time/Timestamp

ERES allows the value axis to be shown in units of time/date (instead of numeric units) for any chart, which has a value axis. Note that the data for the value axis must still be numeric.

To utilize this feature using the API, get a handle to IAxis and use the `setDisplayLabelAsDate` method.

```

IAxis chartYAxis = chart.gethXAxis();
chartXAxis.setDisplayLabelAsDate(true);

```

Please refer to the online API documentation for more information (quadbase.util.IAxis [ <http://data.quadbase.com/Docs70/eres/help/apidocs/quadbase/util/IAxis.html> ]).

#### 8.2.6.1.4.4. Selective String Rendering

Strings, which are drawn horizontally or vertically, look better when not anti-aliased. However, strings at other angles look better when anti-aliased. ERES allows certain text to be anti-aliased and other strings (0 and 90 degree angles) to be drawn without anti-aliasing.

To utilize this feature using the API, get a handle to IDataPointSet and use the `setDisableJava2DForStraightText` method.

```

IDataPointSet dataPoints = chart.gethDataPoints();
dataPoints.setDisableJava2DForStraightText(true);

```

Please refer to the online API documentation for more information (quadbase.util.IDataPointSet [ <http://data.quadbase.com/Docs70/eres/help/apidocs/quadbase/util/IDataPointSet.html> ]).

#### 8.2.6.1.4.5. Drawing Data Points above Horizontal/Vertical/Trend Lines

ERES allows the data points to be drawn on top of any horizontal/vertical/trend lines, i.e. it appears like the data points are resting on top of the line instead of being overshadowed by the line.

To utilize this feature using the API, get a handle to ILinePropertySet and use the method `setDataDrawnOnTop`.

---

```

ILineStyleSet lineProperties = chart.getLineProperties();
lineProperties.setDataDrawnOnTop(true);

```

Please refer to the online API documentation for more information (quadbase.util.ILineStyleSet [ <http://data.quadbase.com/Docs70/eres/help/apidocs/quadbase/util/ILineStyleSet.html> ]).

## 8.2.6.2. Data

The features, shown below, deal with the data in the chart and its components. Each feature changes data shown (typically the presentation of the data shown) or obtains the data in the chart. These changes are also carried forth when exported to a static image.

### 8.2.6.2.1. Getting the Coordinates

ERES allows you to get the information of a datapoint based on a specified pixel position. This returns the category, value, series, and sumby (if they are there) of the data point at the specified pixel location and otherwise returns null.

To get `pickData`, use the method `getPickData(width, height)` in `IHint`.

```

IDataPointSet dataPoints=chart.getDataPoints();
IHint hint=dataPoints.getHint();
PickData pickdata=hint.getPickData(200, 300)           // pixel at width=200
and height=300

```

Please refer to the online API documentation for more information (quadbase.util.IHint [ <http://data.quadbase.com/Docs70/eres/help/apidocs/quadbase/util/IHint.html> ]).

### 8.2.6.2.2. Set Data Limit at Axis Scale

When a manual axis is applied, ERES gives you the option to truncate data points that are beyond the maximum value on the axis. You can set the data limit by using the method `setLimitAtAxisScale` in `IDataPointSet`.

```

IDataPointSet dataPoints = chart.getDataPoints();
dataPoints.setLimitAtAxisScale(true);

```

Please refer to the online API documentation for more information (quadbase.util.IDataPointSet [ <http://data.quadbase.com/Docs70/eres/help/apidocs/quadbase/util/IDataPointSet.html> ]).

### 8.2.6.2.3. Set Null Data as Zero

ERES can now represent any null data as zero data (i.e. datapoints with an associated 0 value). To accomplish this, use the `setNullDataAsZero` method in `IDataPointSet`.

```

IDataPointSet dataPoints = chart.getDataPoints();
dataPoints.setNullDataAsZero(true);

```

Please refer to the online API documentation for more information (quadbase.util.IDataPointSet [ <http://data.quadbase.com/Docs70/eres/help/apidocs/quadbase/util/IDataPointSet.html> ]).

### 8.2.6.2.4. Additional Trend Line Options

ERES allows you to get the minimum, the maximum, the probability and the inverse normal. In addition, you can get the mean value for the normal curve trend line and draw standard deviation trend lines by calling `ITrendLine` and using the appropriate method.

To utilize this feature in the API, call `ITrendLine` and use the appropriate methods.

Note that you can only draw standard deviation trend lines for a normal curve if the chart is a histogram chart with `setLinearScale` and `setRounded` true.

The following code, which can be run as an applet or application, shows how to get information from a normal curve trendline in the chart:

```
ITrendLine normalCurve =
    (ITrendLine)chart.getDataLines().elements().nextElement();
System.out.println("Mean = " + normalCurve.getMean());
System.out.println("St. Dev = " + normalCurve.getStandardDev());
System.out.println("Min = " + normalCurve.getMin());
System.out.println("Max = " + normalCurve.getMax());
System.out.println("Dev of 1.0, Prob = " + normalCurve.getProbability(1.0));
System.out.println("Back Calculated Dev = " +
    normalCurve.getInverseNorm(normalCurve.getProbability(1.0)));
```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/AdditionalTrendLine.zip> ]

Exported Image [ <http://data.quadbase.com/Docs70/help/manual/code/export/AdditionalTrendLine.png> ]

Please refer to the online API documentation for more information (quadbase.util.ITrendLine [ <http://data.quadbase.com/Docs70/eres/help/apidocs/quadbase/util/ITrendLine.html> ]).

### 8.2.6.2.5. Adding Multiple Control Lines to Stack Type Chart

ERES allows you to draw the minimum, the maximum, the average in addition to the stand deviation lines of any level existing in the stack data together, by calling `newControlLine(int linetype, String label, int level)` method with proper parameters setting.

To utilize this feature in the API, you need to set the right parameter value, for the parameter of `linetype`: `MINIMUM = 12`, `MAXIMUM = 13`, `CONTROL_AVERAGE = 10`, `STANDARD_DEVIATION = 11`.

The parameter of `label` is the text display in the legend.

The last parameter of `level` is the level of the data display in the stack chart.



#### Note

This feature only works for Stack Type Chart, i.e., stack column, stack bar and stack area. For stack column chart with combo type is stack area, this feature only works for the main axis data, that is, stack column chart.

The following code, which can be run as an applet or application, shows how to draw multiple control lines to a stack area chart:

```
QbChart chart = new QbChart(this, QbChart.VIEW2D,
    QbChart.STACKAREA, "sample.dat", colInfo);

IDataLineSet hDataLines = chart.getDataLines();

IControlLine clLine1 = hDataLines.newControlLine(13, "Max4", 4);
IControlLine clLine2 = hDataLines.newControlLine(13, "Max3", 3);
IControlLine clLine3 = hDataLines.newControlLine(13, "Max2", 2);
IControlLine clLine4 = hDataLines.newControlLine(13, "Max1", 1);
IControlLine clLine5 = hDataLines.newControlLine(10, "Avg4", 4);
clLine1.setColor(Color.RED);
clLine2.setColor(Color.YELLOW);
clLine3.setColor(Color.GREEN);
```

```

clLine4.setColor(Color.BLUE);
clLine5.setColor(Color.GRAY);
hDataLines.add(clLine1);
hDataLines.add(clLine2);
hDataLines.add(clLine3);
hDataLines.add(clLine4);
hDataLines.add(clLine5);

```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/MultipleControlLines.zip> ]

Exported Image [ <http://data.quadbase.com/Docs70/help/manual/code/export/MultipleControlLines.png> ]

Please refer to the online API documentation for more information (quadbase.util.IControlLine [ <http://data.quadbase.com/Docs70/eres/help/apidocs/quadbase/util/IControlLine.html> ]).

## 8.2.6.3. Chart Specific

The features, shown below, deal with the specific chart types. Each feature shows additional functionality available to the chart, depending on the chart type. These changes are also carried forth when exported to a static image.

### 8.2.6.3.1. Column/Bar Charts

The following features describe the various changes that can be made to column/bar charts using the API. These include modifying the content as well as look of the chart.

#### 8.2.6.3.1.1. Color Separator

ERES allows different colors to be shown for the columns based on defined category values. To use color separator, use the method `setColorSeparators` in `IDataPointSet`.

The following code, which can be run as an application or applet, shows how to set up the color separator:

```

IDataPointSet hDataPoints=chart.gethDataPoints();
hDataPoints.setColorSeparators(new Color[]{Color.green, Color.red,
    Color.blue},
                                new Integer[]{new Integer((int)Math rint(9)),
    new Integer((int)Math rint(14))},
                                QbChart.ASCENDING);

```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/ColorSeparator.zip> ]

Exported Image [ <http://data.quadbase.com/Docs70/help/manual/code/export/ColorSeparator.png> ]

Please refer to the online API documentation for more information (quadbase.util.IDataPointSet [ <http://data.quadbase.com/Docs70/eres/help/apidocs/quadbase/util/IDataPointSet.html> ]).

#### 8.2.6.3.1.2. Disabling Shadow

ERES allows the shadow that appears for the columns/bars to be rendered visible or invisible. You can use this feature by getting a handle to `IDataPointSet` and using the `setShowShadowOnPoint` method.

```

IDataPointSet dataPoints = chart.gethDataPoints();
dataPoints.setShowShadowOnPoint(false);

```

Please refer to the online API documentation for more information (quadbase.util.IDataPointSet [ <http://data.quadbase.com/Docs70/eres/help/apidocs/quadbase/util/IDataPointSet.html> ]).

### 8.2.6.3.2. Pie Charts

The following features describe the various changes that can be made to pie charts using the API. These include modifying the content as well as look of the chart.

### 8.2.6.3.2.1. Drawing Pie Slices Clockwise/Counter Clockwise

ERES now allows the slices in a pie chart to be drawn in clockwise as well as counter clockwise order. To set the clockwise/counter clockwise option, use the `reverseOrder` method in `IDataPointSet`.

```
IDataPointSet dataPoints = chart.getDataPoints();
dataPoints.reverseOrder(QbChart.CATEGORY);
```

Please refer to the online API documentation for more information (`quadbase.util.IDataPointSet` [ <http://data.quadbase.com/Docs70/eres/help/apidocs/quadbase/util/IDataPointSet.html> ]).

### 8.2.6.3.2.2. Pie Border for 0% and 100% Slices

ERES offers the option of removing the pie border for slices that are 0% or 100% of the whole pie. You can do this by getting a handle to `IPiePropertySet` and using the `setRadialBorderDrawnForZero` method.

```
IPiePropertySet pieProperties = chart.getPieProperties();
pieProperties.setRadialBorderForZero(true);
```

Please refer to the online API documentation for more information (`quadbase.util.IPiePropertySet` [ <http://data.quadbase.com/Docs70/eres/help/apidocs/quadbase/util/IPiePropertySet.html> ]).

### 8.2.6.3.2.3. Customize Separator between Category and Percent Value Strings in Pie Legend

ERES allows user-defined separators to be placed between the `Category` and `Percent Value Strings` in the legends for Pie Charts. This can be done by getting a handle to `IPiePropertySet` and use the `setSepSymbol` method.

```
IPiePropertySet pieProperties = chart.getPieProperties();
pieProperties.setSepSymbol(" , ");
```

Please refer to the online API documentation for more information (`quadbase.util.IPiePropertySet` [ <http://data.quadbase.com/Docs70/eres/help/apidocs/quadbase/util/IPiePropertySet.html> ]).

### 8.2.6.3.2.4. Pie Border Color Customizable

ERES allows the user to specify the color for the pie border. This is accomplished by using the `setBorderColor` method in `IPiePropertySet`.

```
IPiePropertySet pieProperties = chart.getPieProperties();
pieProperties.setBorderColor(Color.red);
```

Please refer to the online API documentation for more information (`quadbase.util.IPiePropertySet` [ <http://data.quadbase.com/Docs70/eres/help/apidocs/quadbase/util/IPiePropertySet.html> ]).

## 8.2.6.3.3. Line Charts

The following features describe the various changes that can be made to line charts using the API. These include modifying both the content and the look of the chart.

### 8.2.6.3.3.1. Line Area

ERES allows you to create line areas between a horizontal line and the data line to denote the change. For example, you could have a horizontal line at 25 and a data line. All areas enclosed by the data line and horizontal line and above the horizontal line can be one color and all areas enclosed by the horizontal line and data line and below the horizontal line can be another color. Please note that this feature is only available for 2D Line charts with without a series. Also note that you need to set the color above and below the horizontal line in order to use this feature.

---

To use this feature in the API, you must get a handle to `ILinePropertySet` and use the `setAreaVisible` and `setAreaColors` methods.

The following code, which can be run as an applet or application, shows how to use line area:

```
ILinePropertySet lineProperties = chart.gethLineProperties();
lineProperties.setAreaVisible(true);
lineProperties.setAreaColors(Color.green, Color.yellow);
```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/LineArea.zip> ]

Exported Image [ <http://data.quadbase.com/Docs70/help/manual/code/export/LineArea.png> ]

Please refer to the online API documentation for more information (`quadbase.util.ILinePropertySet` [ <http://data.quadbase.com/Docs70/eres/help/apidocs/quadbase/util/ILinePropertySet.html> ]).

### 8.2.6.3.4. Scatter Charts

The following features describe the various changes that can be made to scatter charts using the API. These include modifying the content as well as look of the chart.

#### 8.2.6.3.4.1. Show Series in Top Label

ERES allows series to be shown in the top labels for scatter charts. This can be done by using the method `showSeriesInTopLabel` in `IDataPointSet`.

```
IDataPointSet dataPoints=chart.gethDataPoints();
dataPoints.showSeriesInTopLabel(true);
```

Please refer to the online API documentation for more information (`quadbase.util.IDataPointSet` [ <http://data.quadbase.com/Docs70/eres/help/apidocs/quadbase/util/IDataPointSet.html> ]).

#### 8.2.6.3.4.2. Drawing Order

ERES allows the connecting lines for a Scatter chart to be drawn in the order of the dataset. For example, if the data contains the following points (0, 2), (3, 4), (1, 2), (2, 5), the default presentation for the connecting lines would generate a line from (0, 2) to (1, 2) to (2, 5) and finally (3, 4). However, you can generate the connecting lines in the order of the data.

To utilize this feature using the API, get a handle to `IDataPointSet` and use the `setConnectLinesInOriginalOrder` method.

```
IDataPointSet dataPoints = chart.gethDataPoints();
dataPoints.setConnectLinesInOriginalOrder(true);
```

Please refer to the online API documentation for more information (`quadbase.util.IDataPointSet` [ <http://data.quadbase.com/Docs70/eres/help/apidocs/quadbase/util/IDataPointSet.html> ]).

#### 8.2.6.3.4.3. Scatter Chart Cube Width

ERES allows the cube width of 3D Scatter charts to be changed to any size. To modify the cube width, first get a handle to `IDataPointSet` and use the `setScatterCubeWidth` method.

```
IDataPointSet dataPoints = chart.gethDataPoints();
dataPoints.setScatterCubeWidth(15);
```

Please refer to the online API documentation for more information (quadbase.util.IDataPointSet [ <http://data.quadbase.com/Docs70/eres/help/apidocs/quadbase/util/IDataPointSet.html> ]).

### 8.2.6.3.5. Overlay Charts

The following features describe the various changes that can be made to overlay charts using the API. These include modifying the content as well as look of the chart.

#### 8.2.6.3.5.1. Multiple Axes Titles

ERES allows the setting of a different title for each axis in an overlay chart. This can be done by getting a handle to each individual axis and using the `getTitle().setValue` method. You get the handle to the individual axis by specifying the layer.

```
IAxis axis0 = chart.gethYAxis();
axis0.getTitle().setValue("XYZ");

IAxis axis1 = chart.gethAxis(1); // Get axis of Layer 1
axis1.getTitle().setValue("ABC");

IAxis axis2 = chart.gethAxis(2); // Get axis of Layer 2
axis2.getTitle().setValue("DEF");
```

Please note that before you set the titles, you need to draw the chart in the background.

Please refer to the online API documentation for more information (quadbase.util.IAxis [ <http://data.quadbase.com/Docs70/eres/help/apidocs/quadbase/util/IAxis.html> ]).

### 8.2.6.3.6. Dial Charts

The following features describe the various changes that can be made to dial charts using the API. These include modifying the content as well as look of the chart.

#### 8.2.6.3.6.1. Control Area Scale Labels

ERES allows the starting and ending scale of a control area to be shown as labels. This can be done by obtaining a handle to a control area and use the `setShowLabel` method.

```
ControlRange cr1 = chart.gethControlRanges().elementAt(0);
cr1.setShowLabel(true);
```

Please refer to the online API documentation for more information (quadbase.util.ControlRange [ <http://data.quadbase.com/Docs70/eres/help/apidocs/quadbase/util/ControlRange.html> ]).

### 8.2.6.3.7. HLCO Charts

The following features describe the various changes that can be made to HLCO charts using the API. These include modifying the content as well as look of the chart.

#### 8.2.6.3.7.1. Changing Candle Stick Color

ERES allows you to change the candlestick color for HLCO charts using the API only. To do so, you will need to get a handle to `IDataPointSet` and use `setCandleStickColors` method.

```
IDataPointSet dataPoints = chart.gethDataPoints();
// Up color is green, Down color is red
dataPoints.setCandleStickColors(Color.green, Color.red);
```

Please refer to the online API documentation for more information (quadbase.util.IDataPointSet [ <http://data.quadbase.com/Docs70/eres/help/apidocs/quadbase/util/IDataPointSet.html> ]).

### 8.2.6.3.7.2. Changing CandleStick Wicker Width

ERES allows you to change the candlestick wicker width (the upper and lower extensions of candlesticks) for HLCO charts using the API only. To do so, you will need to get a handle to `IDataPointSet` and use the method `setCandleStickWidth`. The number passed is the ratio of the candle wicker width to the candle width.

```
IDataPointSet dataPoints = chart.getDataPoints();
// Set width to 0.5
dataPoints.setCandleStickWidth((float)0.5);
```

Please refer to the online API documentation for more information (`quadbase.util.IDataPointSet` [ <http://data.quadbase.com/Docs70/eres/help/apidocs/quadbase/util/IDataPointSet.html> ]).

### 8.2.6.3.8. Surface Charts

The following features describe the various changes that can be made to surface charts using the API. These include modifying the content as well as look of the chart.

#### 8.2.6.3.8.1. Heat Map

ERES allows users to draw a surface chart like a contour map. Basically, the surface chart can be drawn in sections, with different colors according to the threshold values specified.

The following code, which can be run as an applet or application, shows how to create a surface chart with a heat map:

```
double [] heatMapValues = {3, 6};
Color [] heatMapColors = { Color.green, Color.yellow, Color.red};
ColorSpectrum heatMapColorSpectrum = new ColorSpectrum(heatMapColors,
    heatMapValues);
I3DPropertySet set = chart.get3DProperties();
set.setColorSpectrum(heatMapColorSpectrum);
```

The code above creates a 3D surface chart with three color sections: green, yellow and red. The threshold values determining the colors are three and six.

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/HeatMap.zip> ]

Exported Image [ <http://data.quadbase.com/Docs70/help/manual/code/export/HeatMap.png> ]

Please refer to the online API documentation for more information (`quadbase.util.ColorSpectrum` [ <http://data.quadbase.com/Docs70/eres/help/apidocs/quadbase/util/ColorSpectrum.html> ] and `quadbase.util.I3DPropertySet` [ <http://data.quadbase.com/Docs70/eres/help/apidocs/quadbase/util/I3DPropertySet.html> ]).

### 8.2.6.4. Performance

The features, shown below, deal with improving the performance of the chart generation and the export. Each feature below shows additional functionality available to the chart, to decrease memory resources and time needed to generate the chart.

#### 8.2.6.4.1. BufferedImage or Frame

ERES allows the choice of using either `java.awt.image.BufferedImage` or `java.awt.Frame` during export to improve performance. By default, ERES uses `java.awt.Frame` to create the chart object. Using `java.awt.Frame` gives better performance as the number of data points increases while using `java.awt.image.BufferedImage` yields better performance on larger chart dimensions. This is done by using the `setBufferedImageUsed` method in `QbChart`.

```
chart.setBufferedImageUsed(true);
```

Please refer to the online API documentation for more information (quadbase.ChartAPI.QbChart [ <http://data.quadbase.com/Docs70/eres/help/apidocs/quadbase/ChartAPI/QbChart.html> ]).

This functionality is only available for stand-alone charts.

### 8.2.6.4.2. Chart Generation Order

ERES allows you to choose the order in which the chart is generated and even add elements in the generation of the chart using the API only. For example, you can draw a background and a circle and then have the chart generated in the center of the circle to create a chart. You can do this by creating a class that implements the `IChartGraphics` interface and then assigning the class to the `setChartGraphics` method in `QbChart`. In essence, the `IChartGraphics` interface allows you to add or modify any graphics information before or after drawing the chart.

The following code, which can be run as an applet or application, shows how to generate charts and graphics in a specific order:

```
// Open the template
QbChart chart = new QbChart(parent, // container
    "ChartGeneration.cht"); // template

chart.setChartGraphics(new chartGenerationGraphics());

...

class chartGenerationGraphics implements IChartGraphics {

    public void initializeGraphics(Graphics g, int w, int h) {
        g.setColor(Color.red);
        g.fillOval(50, 50, 400, 400);
    }

    public void finalizeGraphics(Graphics g, int w, int h) {
        g.setColor(Color.white);
        g.fillOval(125, 225, 50, 50);
        g.setColor(Color.orange);
        g.drawString("HELLO WORLD", 150, 250);
    }
}
```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/ChartGeneration.zip> ]

Exported Image [ <http://data.quadbase.com/Docs70/help/manual/code/export/ChartGeneration.png> ]

Please refer to the online API documentation for more information (quadbase.util.IChartGraphics [ <http://data.quadbase.com/Docs70/eres/help/apidocs/quadbase/util/IChartGraphics.html> ]).

This functionality is only available for stand-alone charts.

### 8.2.6.5. Viewer

The features, shown below, deal with the Viewer when viewing the chart in either a java application or java applet. Each feature below shows additional functionality available to Chart Viewer.

### 8.2.6.5.1. Call Back Mechanism

ERES has a call back mechanism for higher levels to handle the event. An action event is generated when a data object in the chart is selected by the viewer. The event argument contains an instance of `PickData`, which provides information of the series, category, value, etc, of the data point selected.

The following code, which can be run as an applet or application, shows how to capture an event:

```

static TextField textField;

// Open the template
QbChart chart = new QbChart(parent, // container
    "CallBack.cht"); // template

chart.addActionListener(new callBackActionListener());

...

class callBackActionListener implements ActionListener {

    public void actionPerformed(ActionEvent e) {
        Object arg = ((QbChart) e.getSource()).getArgument();

        String click;

        switch (e.getModifiers()) {

            case QbChart.LEFT_SINGLECLICK:
                click = "Left single click";
                break;

            case QbChart.LEFT_DOUBLECLICK:
                click = "Left double click";
                break;

            case QbChart.RIGHT_SINGLECLICK:
                click = "Right single click";
                break;

            case QbChart.RIGHT_DOUBLECLICK:
                click = "Right double click";
                break;

            default: // shall not happen
                click = "Error !";

        }

        if (arg instanceof PickData)
            textField.setText(((PickData) arg).toString() + " " +
click);

        else
            textField.setText((String) arg + " " + click);

    }

}

```

---

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/CallBack.zip> ]

Exported Image [ <http://data.quadbase.com/Docs70/help/manual/code/export/CallBack.png> ]

This functionality is only available for stand-alone charts.

### 8.2.6.5.2. Disable/Enable Tools-Tips Text

ERES allows the tool-tips text for the navigation panel to be enabled and disabled. This can be done by using the method `setToolTipEnabled` in `I3DControlPanel`. Note that this option is for 3D charts only.

```
I3DControlPanel controlPanel = chart.get3DControlPanel();
controlPanel.setToolTipEnabled(true);
```

Please refer to the online API documentation for more information (`quadbase.util.I3DControlPanel` [ <http://data.quadbase.com/Docs70/eres/help/apidocs/quadbase/util/I3DControlPanel.html> ]).

### 8.2.6.5.3. Canvas Area

ERES allows the viewpanel containing the canvas to be selected so that more event properties (i.e. user defined event properties) can be added. You can do this by getting a handle to `ICanvas` and using the `getCanvasArea()` method to return the component.

```
ICanvas chartCanvas = chart.getCanvas();
Component chartCanvasComponent = chartCanvas.getCanvasArea();
```

Please refer to the online API documentation for more information (`quadbase.util.ICanvas` [ <http://data.quadbase.com/Docs70/eres/help/apidocs/quadbase/util/I3DControlPanel.html> ]).

## 8.2.7. Changing Chart Viewer Options

At times, you may want to configure what users can or cannot do when they are viewing the chart using the Chart Viewer.

When the user is using Chart Viewer to view the chart, right clicking on the chart causes a menu to pop up. The user can use this menu select chart options such as changing chart type, changing chart dimension etc. The user can also choose to export the chart and the type of the static image. While the default pop-up menu lists all available choices, API methods exist that control what options are available in the pop-up menu of Chart Viewer.

These API calls are available in `IPopupMenu`:

```
IPopupMenu popupMenu = chart.getPopupMenu();
popupMenu.setDimMenuEnabled(boolean b);
popupMenu.setPopupMenuEnabled(boolean b);
popupMenu.setTypeMenuEnabled(boolean b);
```

Please refer to the online API documentation for more information (`quadbase.util.IPopupMenu` [ <http://data.quadbase.com/Docs70/eres/help/apidocs/quadbase/util/IPopupMenu.html> ]).

## 8.2.8. Javadoc

Javadoc for the entire API is provided along with ERES. The API covers both the Report and the Charting API. It is located at Quadbase website [ <http://data.quadbase.com/Docs70/eres/help/apidocs/index.html> ].

## 8.2.9. Swing Version

1.1 JFC/Swing versions of the ERES charting API is also available. For more details, please refer to `quadbase.ChartAPI.swing` package [ <http://data.quadbase.com/Docs70/eres/help/apidocs/quadbase/ChartAPI/swing/package-summary.html> ].

## 8.2.10. Summary

The ERES API provides an easy-to-use, yet powerful reporting library for business applications. Combined with Chart Designer, programming is as simple as adding one line of code to your applet/application. All of the attributes of a chart may be set in a template file, which can be created with the ERES Designer. The ERES API has been tested with Netscape's Communicator (4.06 and above), Microsoft's Internet Explorer (4.x and above), and Sun's Appletviewer (1.2 and above) on the Windows 95, Windows NT/2000, Solaris, Linux, AIX, and HP platforms.

## 8.3. Managing Users and Groups

### 8.3.1. Introduction

User and group administrator can be done programmatically, via the API (instead of always going to the ERES home page). API calls are available to create, edit, and delete users and groups.

Both `ERESOrganizer.jar` (located in the `ERES/lib` directory) and `ERESServer.jar` (located in the `ERES/WEB-INF/lib` directory) need to be added to the `CLASSPATH` for any code using ERES Organizer. The following snippet shows how to connect to ERES Organizer:

```
QbOrganizer.setServletRunner("http://localhost:8080");
QbOrganizer.setServletContext("ERES/servlet");
QbOrganizer organizer = new QbOrganizer(null, "admin", "admin");
```

The above code sets the connection information to connect to ERES Server and provides the username and password (in the above example, `admin` and `admin`) for ERES Organizer.

All examples and code given in the manual assume that ERES server is running locally (i.e., on your local machine) and on port 8080. You can change this by going to the source code (you can download the source code by clicking on the *Full Source Code* link in the corresponding chapter), editing the code to enter the new connection information, and recompiling the code.

Also note that if you have applets that use ERES Report API, the browser must have at least a 1.5 JVM plugin.

### 8.3.2. Users and Groups

You do not have to use the ERES Admin console to create, edit, and/or delete user and group information. You can modify any user and/or group information via the API. Note that only the `admin` user can call the API methods.

To modify the user and group information, you must get a handle to `UserGroupProperties` using the following call:

```
UserGroupProperties userGroupProperties = new
    UserGroupProperties(organizer);
```



#### Note

The code must log to ERES as the `admin` user otherwise the user/group information will remain unchanged. Any username and/or group name must be in lowercase. Uppercase and mixed case names are converted to lower case automatically.

#### 8.3.2.1. Creating Users and Groups

You would use the following method to create a user:

```
public void createUser(String userName, String fullName, String email,  
String password, String role, String securityLevel)
```

For example:

```
userGroupProperties.createUser("jdoe", "John Doe", "jDoe@somedomain.com",  
"123", IUser.ROLE_DESIGNER, "manager");
```

While the following method to create a group:

```
public void createGroup(String newGroup, String description, String[] users)
```

For example:

```
userGroupProperties.createGroup("testinggroup", "For testing purpose only",  
new String[]{"admin", "jDoe"});
```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/CreatingUserGroup.zip> ]

The above code creates a new user `jdoe` along with a new group `testinggroup`. The user profile information as well as group information is also passed via the code.

### 8.3.2.2. Deleting Users and Groups

You can also delete any users/groups through the code in the same manner.

You would use the following method to delete a user:

```
public void deleteUser(String[] users)
```

For example:

```
userGroupProperties.deleteUser(new String[] {"jdoe"});
```

While the following method to delete a group:

```
public void deleteGroup(String[] groups)
```

For example:

```
userGroupProperties.deleteGroup(new String[] {"testinggroup"});
```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/DeletingUserGroupERES.zip> ]

Result Screenshot [ <http://data.quadbase.com/Docs70/help/manual/images/DeletingUserGroup.gif> ]

The above code deletes `jdoe` user and `testinggroup` group. Note that the user and group have to exist before they can be deleted.

### 8.3.3. Single Sign-On

You can also have your own custom login page (for example, login to your portal) and then pass the required information to ERES (rather than having to login twice, once for your application and the other for ERES).

To pass in the login information, you would need to pass the following parameters to `authenticate.jsp` (located in the root ERES install directory):

- origPage :** Page to be redirected to if login is successful.
- loginPage :** Page to be redirected to if login is not successful.
- username :** username to be passed.
- password :** password to be passed.

The following example shows the possible contents of a jsp page that is passing the information to `authenticate.jsp`. Note that in the example, username and password is inputted although you can specify them before calling `authenticate.jsp`:

```
<html>

    <head>
    </head>
    <body>

        <form name="Login" action="authenticate.jsp" method="POST">

            <input type="hidden" name="origPage" value="MenuPage.jsp">
            <input type="hidden" name="loginPage"
value="myLogin.html">

            <table width=100% cellpadding=2 cellspacing=0>

                <tr>
                    <td width=80 valign="middle" align="right">User
Name: </td>
                    <td align="left"><input type="text"
name="username"></td>
                </tr>

                <tr>
                    <td width=80 valign="middle" align="right">Password:
</td>
                    <td align="left"><input type="password"
name="password"></td>
                </tr>

            </table>

            <input type="image" src="Web_Component/STARTUP/Login.gif"
border=0></td>

        </form>

    </body>

</html>
```

The recommended way to run the above code is to go to the root ERES directory and copy the contents to a empty .jsp page. Note that myLogin.html has to be created as well.

### 8.3.4. Login Listener

You can also create additional code that changes the datasource (only if the datasource is a database) of the report/chart, based on the user logged onto ERES at the time. This code extends the LoginListener class and specifies the database connection information, based on the user.

Please note that after creating the code you need to change QB.properties (in the <ERES-installation-directory>/WEB-INF/classes directory) and add the following argument to the ServerCommands= line:

```
-LoginListenerClass:<name of class file extending LoginListener>
```

The following code shows how to use the LoginListener class:

```
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpSession;

import quadbase.auth.WebLogin;
import quadbase.auth.bean.Login;
import quadbase.reportorganizer.ext.LoginListener;

public class ExampleLoginListener extends LoginListener {

    public void setUserDatabaseConnection(HttpServletRequest req,
String userName) throws Exception {

        HttpSession session = req.getSession();
        WebLogin wl =
(WebLogin)session.getAttribute(Login.WEB_LOGIN);
        if (wl == null || wl.getUser() == null) {

            return;

        }

        String user = wl.getUser();
        if (user.equalsIgnoreCase("user1"))
        {

session.setAttribute(USER_DBURL, "someDatabaseURL1");

session.setAttribute(USER_DBDRIVER, "someDatabaseDriver1");

session.setAttribute(USER_DBUSERNAME, "someDatabaseUsername1");

session.setAttribute(USER_DBPASSWORD, "someDatabasePassword1");

        } else if (user.equalsIgnoreCase("user2"))
        {

session.setAttribute(USER_DBURL, "someDatabaseURL2");

session.setAttribute(USER_DBDRIVER, "someDatabaseDriver2");
```

```
session.setAttribute(USER_DBUSERNAME, "someDatabaseUsername2");

session.setAttribute(USER_DBPASSWORD, "someDatabasePassword2");

    }

}

}
```

To use the above code, add the following to the `ServerCommands=` line of the `QB.properties` INI file:

```
-LoginListenerClass:ExampleLoginScheduler
```

Make sure that `ExampleLoginScheduler` is in the `CLASSPATH`. The code will switch the datasource information to a different database, depending on whether `user1` or `user2` has logged on. If it is a different user, then the original datasource will be used.

### 8.3.5. Javadoc

Javadoc for the entire API is provided along with ERES. It is located here (Reports and charts) [ <http://data.quadbase.com/Docs70/eres/help/apidocs/index.html> ] and here (ES functions) [ <http://data.quadbase.com/Docs70/eres/help/ERESapidocs/index.html> ].

### 8.3.6. Summary

ERES API provides an easy-to-use and powerful API that can be used in your custom code to administer and manage users and groups in ERES.

Please note that the API requires a JDK 1.8 or above. The ERES API has been tested on Windows, Solaris, Linux, AIX, HP, and Mac platforms.

## 8.4. ERES Menu API Overview

### 8.4.1. Introduction and Setup

A Menu is typically a jsp page that shows all the published reports and charts. Depending on the options available, a menu may allow the user to choose the format in which the report and/or chart are to be presented. The menus discussed here are analogous to those described in the Menu chapter.

ERES provides an easy-to-use application programming interface (API) that allows users to create and customize menus to show in their servlet/jsp applications. This API is written in 100% Pure Java and can be run on any platform with no or few minimal changes. Any and every part of the menu is customizable using the API.

The main package for ERES Menu API is the `quadbase.reportorganizer.MenuAPI` package. Associated with this package are five main classes: `MenuNode`, `MenuTableRow`, `QbMenu`, `ScheduleJob` and `Util`. The remainder of this chapter will give you an overview of these various classes and their usage. Please note that the complete documentation is located at [help/ERESapidocs/index.html](http://data.quadbase.com/Docs70/eres/help/ERESapidocs/index.html).

The jar files for using ERES Menu API are located in the `ERES/lib` directory. These jar files should already be present in your application's `CLASSPATH` when you set up ERES.

Currently, you can only use ERES Menu API in a servlet/jsp environment. The servlets/jsps must also run in the same servlet context as ERES server.

### 8.4.2. Using the API

This section discusses the usage of ERES Menu API with code examples and contains descriptions of main API classes.

## 8.4.2.1. Creating and Customizing an ERES Menu

The sections below describe the various steps involved in creating and customizing an ERES Menu:

### 8.4.2.1.1. Connecting to ERES Server

The first step is to connect to ERES server. After successfully connected to the server, a `QbMenu` object can be allocated. `QbMenu` is the main class containing contents for a Menu page.

The ERES server is simply an object that resides in the `ServletContext` of the web application. Since you will want to put most Java code in your Java bean, you will need to pass the `HttpServletRequest` and `ServletContext` objects from the jsp to the bean. The following jsp and bean code illustrates this:

- Example jsp (`Menu_login.jsp`):

```
<!-- import the bean class -->
<%@page language="java" import="help.examples.Menu.exampleMenuLogin"%>

<!-- include/instantiate the bean -->
<jsp:useBean id="exampleMenuLogin" scope="session"
  class="help.examples.Menu.exampleMenuLogin" />

<!-- pass the HttpServletRequest and ServletContext objects to the bean -->
<% exampleMenuLogin.processRequest(request, application); %>
```

where the jsp passes the request (`HttpServletRequest`) and application (`ServletContext`) to the Java bean `MenuLogin`'s `processRequest` method. Next, you will need to implement code in the bean that allocates the `QbMenu` object from the `ServletContext`:

- Example bean (`MenuLogin.java`):

```
public void processRequest(HttpServletRequest request, ServletContext
context) throws IOException {

    QbMenu qbMenu = new
QbMenu(context.getAttribute(QbMenu.QBMENUDATA));

}
```

where `context.getAttribute(QbMenu.QBMENUDATA)` is set when ERES server started.

### 8.4.2.1.2. Getting User Menu Page

The second step uses the `QbMenu` object to authenticate the user. This is easy since all security and permission checks are performed by ERES server. If login is successful, you can store the `QbMenu` object for the session so that for the duration of the session, the user will not have to login again. The following code illustrates this:

```
boolean loginSuccess = qbMenu.login(userName, password);
if (loginSuccess) {

    //store the QbMenu object in session

    session.setAttribute("QBMENUOBJ", qbMenu);

}
```

where `"QBMENUOBJ"` is the name that you can use later for retrieving the stored `QbMenu`. This means that after storing the `QbMenu` in the session, in other jsp pages by doing `session.getAttribute("QBMENUOBJ")`.

The `QbMenu.login` method returns `true` only if the user name and password match. If login failed, the user can be prompted again.



### Note

Login **MUST** succeed for the `QbMenu` object to return a valid Menu.

#### 8.4.2.1.3. Getting all Visible Nodes

After authenticating the user, you can get a list of authorized Menu nodes, which contain reports or charts, from the `QbMenu`. A node is either a project or a folder in ERES Organizer.

```
MenuNode[] MenuNode = qbMenu.getMenuNodeList(null);
```

Passing in `null` to the `getMenuNodeList` method results in getting all the available Menu nodes. If you pass in a string, the method will return only those nodes items that contain the string.



### Note

The path for the Menu node will be a complete one, i.e. `Project1/Folder1/SubFolder2`

#### 8.4.2.1.4. Getting the Node Details

Once you have retrieved the Menu nodes, the next step is get the node details, namely, the files contained in the node. Conceptually, each item in the node is a row and is represented by the `MenuTableRow` object. The following code iterates through each file in all nodes:

```
for (int i = 0; i < MenuNode.length; i++) {
    for (int j = 0; j < MenuNode[i].getRowItemCount(); j++) {
        MenuTableRow MenuTableRow = MenuNode[i].getMenuTableRowAt(j);
    }
}
```

#### 8.4.2.1.5. Getting the File Details

You can obtain all the information of the file (such as file name, description, whether it is a chart or report etc) using the `MenuTableRow` class.

The following methods are used to obtain the file details:

```
String getDescription()
Returns the description for the row item
```

```
String getName()
Returns the name for the row item
```

```
String getPath()
Returns the path for the row item
```

```
ScheduleJob getScheduleJob()
```

Returns the schedule job information for the row item

ScheduleJob getArchiveJob()

Returns the archive job information for the row item

String getSecurityLevel()

Returns the security level for the row item

String getURL()

Returns the URL for the row item

boolean isArchiveOptionAvail()

Returns whether the archive option is available for the row item

boolean isScheduleOptionAvail()

Returns whether the schedule option is available for the row item

boolean isChart()

Returns whether the row item is a chart

boolean isReport()

Returns whether the row item is a report

boolean isDashboard()

Returns whether the row item is a dashboard

Using the above methods, you can get the various file details.

#### **8.4.2.1.6. Scheduling/Archiving**

After getting the file details (see previous section), you can also get all the details for any scheduled or archived jobs for the particular file. This can be done by getting the ScheduleJob object from MenuTableRow.

The following methods are used to obtain any scheduling/archiving information:

long getEndDate()

Returns the ending date for the row item's schedule job. If no end date is specified, -1 is returned

long getStartDate()

Returns the starting date for the row item's schedule job

String getName()

Returns the name of the job for the row item

long getNextExportTime()

Returns the next scheduled export time for the row item

int getParamSetCount()

Returns the number of parameter sets for the row item's schedule/archive job

Object[] getParamSet(int setNumber)

Returns the selected parameter value(s) for the specified parameter set for the row item's schedule/archive job

String[] getParamNames()

Returns the names of all the parameters for the row item

```
String getLocation()
Returns the location of the report (if the row item happens to be a report)

String getChartLocation()
Returns the location of the chart (if the row item happens to be a chart)

String getScheduledFileLocation()
Returns the location of the exported file created by the row item's schedule

String getScheduledFileLocation(int paramIndex)
Returns the location of the exported file, for the specified parameter set,
    created by the row item's schedule

String getSecurityLevel()
Returns the security level for the row item's schedule

boolean isChart()
Returns whether the row item's job is for a chart

boolean isReadable()
Returns whether the user has permission to the row item's job

boolean isReport()
Returns whether the row item's job is for a report

boolean isArchiveJob()
Returns whether the row item's job is an archived job

boolean isScheduleJob()
Returns whether the row item's job is a scheduled job

int getArchiveFilesCount()
Returns the total number of archived files for the row item

long getArchiveDate(int index)
Returns the date of the specified archived file for the row item

String getArchiveFileLocation(int index)
Returns the location of the specified archive file for the row item

String getArchiveFileLocation(int index, int paramIndex)
Returns the location of the specified archive file, using the specified
    parameter set, for the row item
```

You can get the various details for any scheduled/archived jobs for the particular row item by using the above methods.

### 8.4.2.2. Exporting Reports/Charts using the LookupServlet Servlet

Depending on your needs and architecture, you can setup the Menu to show the reports and charts in various static formats (such as DHTML, PDF, JPEG, GIF, PNG, etc) or in an interactive applet. You can code your own servlet using the APIs provided in the ERES Reporting API Overview and ERES Charting API Overview chapters to get the templates and export them to the format desired.

ERES also comes with a ready-made servlet that can take different parameters to produce the report and/or chart in the format desired without you having to write any additional servlet code. This servlet is the `LookupServlet` servlet and is generally in the `ERES/servlet` context (for example, `http://192.168.0.1:8080/ERES/servlet/LookupServlet`). For more details on this servlet and the various parameters it requires, please refer to the Designer guide.

### 8.4.3. Javadoc

Javadoc for the entire API is provided along with ERES. It is located here (Reports and charts) [ <http://data.quadbase.com/Docs70/eres/help/apidocs/index.html> ] and here (ES functions) [ <http://data.quadbase.com/Docs70/eres/help/ERESapidocs/index.html> ].

### 8.4.4. Summary

ERES already provides a menu for any user-created reports and/or charts. This menu can be reached by logging into the Organizer and choosing *Published Files*. In addition to this, ERES also provides an easy-to-use, yet powerful library to create your own menu in your own styles to display any reports and/or charts.

The jsp code discussed in this chapter is also available in the `ERES/help/examples/Menu` directory (this directory contains both the jsp pages and the css styles). The `ERES/WEB-INF/classes/help/examples/Menu` directory contains the source and class files for the beans. This example demonstrates the steps given above.

Please note that the Menu API requires a JDK 1.8 or above. The ERES Menu API and jsp pages have been tested with Mozilla (1.3 and above), Firefox (0.91 and above), and Microsoft Internet Explorer (5.x and above) on the Windows NT/2000, Solaris, Linux, AIX, and HP platforms.

## 8.5. Menu JSP Programming

As detailed in Section 7.1 - The Menu Page, the Menu Page in ERES provides a portal that allows users to automatically publish reports and charts that have been deployed in Organizer. Although a standard format for the Menu Page is provided with ERES, users can completely customize this presentation to fit with their existing web content. All of the Menu Page formatting and functions are written on the client using the Java Standard Tag Library, HTML, and JavaScript while the core functionality (processing login, retrieving reports, charts, and schedule/archive information) can be accessed using the JSP beans provided. These features allow web developers to easily create their own reporting portals without developing any server-side code.

This chapter describes how the Menu Page works in the standard format. There is also a more simplified example available in the ERES installation under `<ERESInstallDir>/help/examples/menu/exampleMenu-JSTL.jsp`. Note that if you do wish to modify the standard Menu Page, it is recommended that you create your own JSP files. The standard files described in this chapter will be modified during an upgrade installation of ERES.

### 8.5.1. Menu Page Architecture

The standard format menu page consists of the following JSP pages and Java beans:

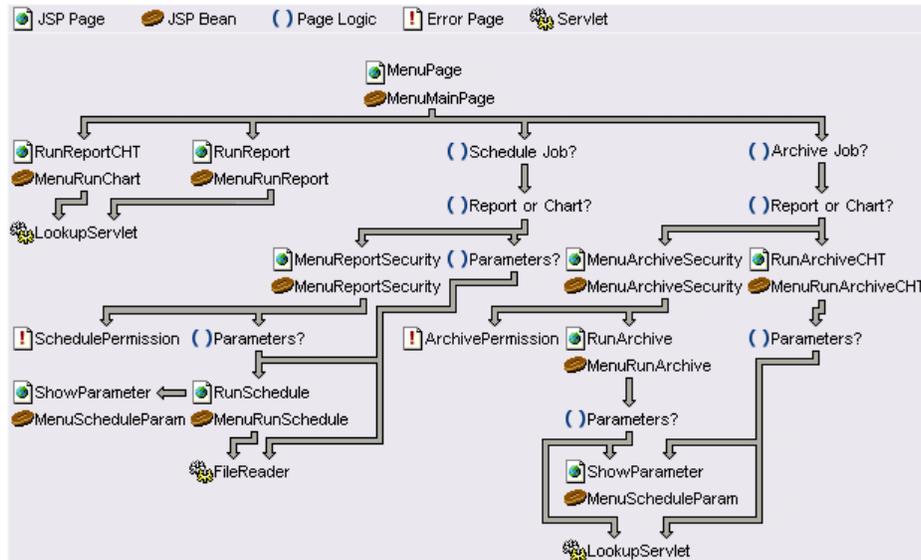
```
MenuArchiveSecurity.jsp
MenuError.jsp
MenuPage.jsp
RunArchive.jsp
RunArchiveCHT.jsp
RunReport.jsp
RunReportCHT.jsp
RunSchedule.jsp
ShowParameters.jsp

MenuScheduleParam.class
MenuRunSchedule.class
MenuRunReport.class
MenuRunChart.class
MenuRunArchiveCHT.class
MenuRunArchive.class
MenuReportSecurity.class
MenuMainPage.class
MenuLogin.class
MenuArchiveSecurity.class
ItemScheduleJob.class
ItemNode.class
ItemArchiveJob.class
```

FolderNode.class

In the ERES Start-up page, when you click on the *View Published Files* option, you will see the menu page which lists all the reports, charts, and files to which you have access privileges. The page is displayed by `MenuPage.jsp`. `MenuPage.jsp` uses the Java bean `MenuMainPage.class`.

The relationship among the JSP pages and Java beans is as follows:



*Menu Page JSP Architecture*

## 8.5.2. Login Code

In all of the JSP pages, except `MenuError.jsp` there is a code scriptlet similar to the following:

```
<%
WebLogin wl = (WebLogin) session.getAttribute("eres-web-login");
String redirect = MenuPageBean.getLoginPage();
if (wl == null || wl.getUser() == null) {

    response.sendRedirect(redirect+"?errorMsg=Not+Logged+In");

} else if (!wl.canAccess(WebLogin.MENU)) {

    response.sendRedirect(redirect+"?errorMsg=Permission+Denied.+You+must
+have+Menu+privilege+to+access+this+page.");

} else {

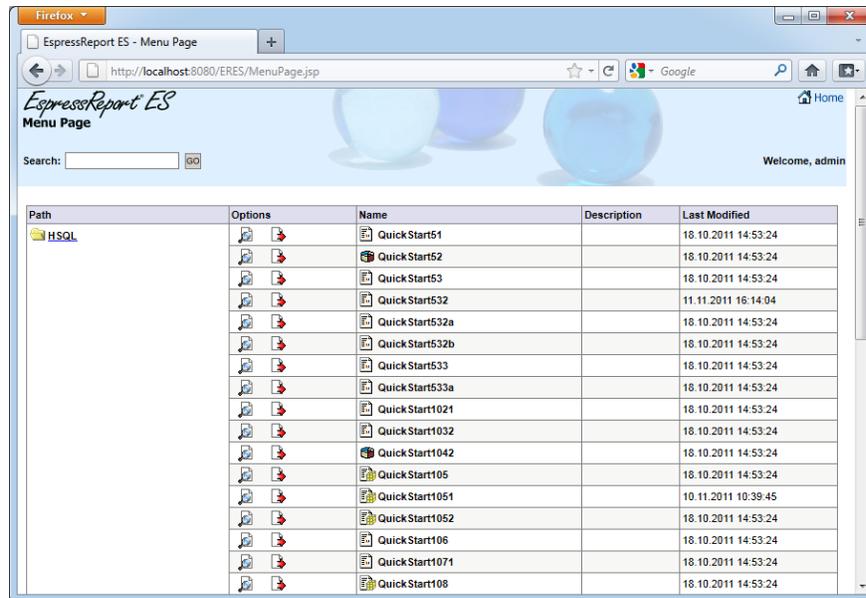
    MenuPageBean.processRequest(request, application);

}%>
```

This is trying to ensure that you are properly logged in. If not, you will be directed to a login page. Otherwise, some initialization code will be run when the method `processRequest(request, application)` is called. Each bean has a method by this name for initializing the bean.

## 8.5.3. Building the Main Page

`MenuPage.jsp` retrieves the items (reports, charts, files) that you have right to access and displays them in the tabular format as shown in the following figure:



Menu Main Page

Items are grouped under folders, as you see in the Organizer. However, in the menu page, folder hierarchy is not displayed. If you have nested folders and you want to display them as such, you can refer to `exampleMenu-JSTL.jsp` under the `<ERESInstallDir>/help/examples/menu` directory.

In the following block of tags, the first `forEach` starts the for loop to get the folders. The second one retrieves the individual items in the folder.

```

<c:set var="rowColor" value="innertablewhite" />
<c:forEach items="{MenuPageBean.folderList}" var="current">

<c:set var="firstTime" value="true" />
<c:forEach items="{current.items}" var="curItem" >
<tr>
<c:if test="{firstTime}" >
<td rowspan=<c:out value="{current.rowCount}" /> valign="top"
class="innertablewhite">

    <table width=100% cellpadding=0 border=0
class="innertablewhite">
        <tr>
            <td width=20></td>
            <td valign="middle" class="text"><c:out
value="{current.nodeName}" /> </td>
        </tr>
    </table>

</td>

<c:set var="firstTime" value="false" />
</c:if>
...
...

```

The `firstTime` variable is true if the current item is the first one in the folder. `rowspan` is set to total item count for the folder. `rowColor` is set to alternate values to achieve dual color.

The following code sets up the link to run the report, chart, or simply display the referenced file:

```

<c:set var="archiveObj" value="{curItem.archiveJob}" />
<!-------test for archive object ----->
<c:choose>
<c:when test="{curItem.archiveJobAvailable}">

<:c:set var="argv" >
.jsp?NODEID=<c:out value="{current.nodeID}" />&TABLEID=<c:out
value="{curItem.rowID}" />&ARCHID=<c:out value="{archiveObj.id}" />
</c:set>
</c:when>
<c:otherwise>
<c:set var="argv" >
.jsp?NODEID=<c:out value="{current.nodeID}" />&TABLEID=<c:out
value="{curItem.rowID}" />
</c:set>
</c:otherwise>
</c:choose>

<c:choose>
<c:when test="{curItem.chart || curItem.report}">
<td width=20 align="center"><a href=# onClick="O('RunReport<c:if
test= "{curItem.chart}">CHT</c:if><c:out value="{argv}" /
>', 'RunReport', 350, 370); return false;">
</a></td>

</c:when>
<c:otherwise>
<td width=20 align="center"><a href=# onClick="O('<c:out
value="{curItem.documentURL}" />', 'RunReport', 700, 450); return false;">
</a></td>

</c:otherwise>
</c:choose>

```

Depending on whether the item is a chart, report, a file, RunReportCHT.jsp, RunReport.jsp or just the file URL will be invoked “onClick”. The argv variable is used to construct the request objects for the RunReport.jsp and RunReportCHT.jsp. If the report or chart has archive job turned on, the archive job's ID will be included. In such case, when you run the report or chart, you have the option to save the report or chart as archive.

If the item has a scheduled job, the following code will be executed:

```

<c:set var="scheduledRptURL" >
MenuReportSecurity<c:out value="{argv}" escapeXml="false" />&SCHEDID=<c:out
value="{curItem.scheduleJob.id}" />&SCHEDPARAMSETCOUNT=<c:out
value="{curItem.scheduleJob.paramSetCount}" />
</c:set>
<c:choose>
<c:when test="{curItem.scheduleJob.chart &&
curItem.scheduleJob.paramSetCount==0}" >
<c:set var="width" value="{750}" />
<c:set var="length" value="{450}" />
<c:set var="scheduledRptURL" >
<c:out value="{MenuPageBean.contextPath}" />/FileReader?SCHEDID= <c:out
value="{curItem.scheduleJob.id}" />
</c:set>

```

```

</c:when>
<c:otherwise>
<c:if test="\${curItem.scheduleJob.chart}">
<c:set var="scheduledRptURL">

RunSchedule<c:out value="\${argv}" />&SCHEDID=<c:out
  value="\${curItem.scheduleJob.id}" />&SCHEDPARAMSETCOUNT=<c:out
  value="\${curItem.scheduleJob.paramSetCount}" />
</c:set>
</c:if>
</c:otherwise >
</c:choose>
<td width=20 align="center"><a href=# onClick="0('<c:out
  value="\${scheduledRptURL}" />', 'RunSchedule', <c:out value="\${width}" /
>, <c:out value="\${length}" />); return false;"><
</a></td>

```

The variable `scheduleRptURL` will be set to the correct value (link) based on whether the item is a chart with parameters, report with parameters or just report or chart without parameters. As such, “onClick”, if the item is a chart without parameters, the `FileReader` servlet will be run; if the item is a chart with parameters, `RunSchedule.jsp` will be run; if the item is a report, action will be directed to `MenuReportSecurity.jsp`. `Web_Component/MENU/ViewSchedule.png` is the schedule object icon.

If the item has an archive job, the following code will be executed:

```

<c:choose>
<c:when test="\${archiveObj.chart}" >
<c:set var="archiveURL" >
RunArchiveCHT.jsp?ARCHIVEID=<c:out value="\${archiveObj.id}" />
</c:set>
</c:when>
<c:otherwise>
<c:set var="archiveURL" >
MenuArchiveSecurity.jsp?ARCHIVEID=<c:out value="\${archiveObj.id}" />
</c:set>
</c:otherwise>
</c:choose>
<td width=20 align="center"><a href=# onClick="0('<c:out
  value="\${archiveURL}" />', 'RunArchive', 350, 370); return false;">
</a></td> </c:when>
<c:otherwise>
<td width=20 align="center"></td>
</c:otherwise>
</c:choose>

```

The variable `archiveURL` will be set to the correct value (link) based on whether the item is a chart or report. As such, “onClick”, `RunArchiveCHT.jsp` will be run if it is a chart; if the item is a report, `MenuArchiveSecurity.jsp` will be run. `Web_Component/MENU/ViewArchive.gif` is the icon for an archive object.

The remaining (right-most) three columns show the name of the file, description if any, and the date it was last modified. The JSTL tags used are mainly `<c:if test...> </c:if>` tags and the `<c:choose> ... </c:choose>` tags. They are quite self-explanatory.

Lastly, the following code completes the dual/alternate color scheme for the table:

```

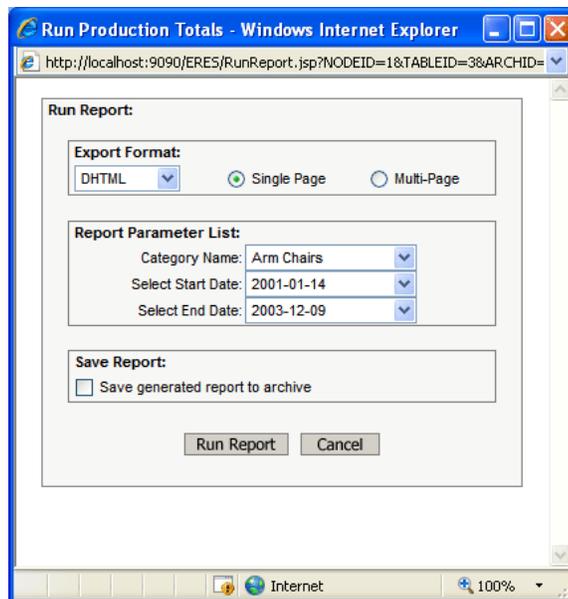
<c:choose>
<c:when test="\${rowColor=='innertablewhite'}" >
<c:set var="rowColor" value="innertablegrey" />
</c:when>
<c:otherwise>
<c:set var="rowColor" value="innertablewhite" />
</c:otherwise>
</c:choose>

```

## 8.5.4. Running Reports and Charts

For each report and chart in the main menu page, users will have the option to run that chart or report on-demand as discussed in the previous section. If the item is a report, then `RunReport.jsp` will be opened. If the item is a chart then `RunReportCHT.jsp` will be opened.

`RunReport.jsp` which uses Java bean `MenuRunReport`, is invoked when you click the run report/chart icon. The following window will be shown:



*Run Report Dialog*

In the following block of tags, `<c:when test="\${menuRunReport.paramReport}" >` checks to see if the report has run-time parameters. If so, the parameter prompts will be displayed by the tag `<c:out value="\${menuRunReport.paramTable}" escapeXml="false" />`.

```

<c:choose>
<c:when test="\${menuRunReport.paramReport}" >
    <c:set target="\${menuRunReport}" property="languageEncodedText"
value="Report Parameter List:" />
    <tr>
    <td>
    <center>
    <table width=90% cellspacing=1 cellpadding=0 class="outertable">
    <tr>
    <td>
    <table width=100% cellspacing=0 cellpadding=1 border=0
class="innertablegrey">
    <tr>

```

```

        <td colspan=2 class="textbold">&nbsp;<c:out
value="\${menuRunReport.languageEncodedText}" /> </td>
        </tr>
        <tr><td>
        <c:out value="\${menuRunReport.paramTable}" escapeXml="false" />
        </td></tr>
        </table>
        </td>
        </tr>
        </table>
        </center></td>
        </tr>
        <tr>
        <td>&nbsp;<
        </tr>
</c:when>
<c:otherwise>
<tr><td> </tr></td>
</c:otherwise>
</c:choose>

```

The tag, `<c:set target="\${menuRunReport}" property="languageEncodedText" value="Report Parameter List:" />` basically allows you to display the content in value in the language you are working with. You can do this with scriptlet as well, i.e. `<%= MenuRunReport.getEncodedText("Report Parameter List")%>`.

If there is an active archive job, you will see the *save generated report to archive* checkbox as shown in the screen shot. The following code generates the option in the window:

```

<c:set target="\${menuRunReport}" property="languageEncodedText" value="Save
Report:" />

        <tr>
        <td>
        <center>
        <table width=90% cellspacing=1 cellpadding=0 class="outertable">
        <tr>
        <td>
        <table width=100% cellspacing=0 cellpadding=1 border=0
class="innertablegrey">
        <tr>
        <td colspan=2 class="textbold">&nbsp;<c:out
value="\${menuRunReport.languageEncodedText}" /></td>
        </tr>
        <c:set target="\${menuRunReport}" property="languageEncodedText"
value="Save generated report to archive" />
        <tr>
        <td valign="middle" width=10>
        <input type="checkbox" name="BACKUPTOARCHIVE"></td>
        <td valign="middle" class="text"><c:out
value="\${menuRunReport.languageEncodedText}" /></td>
        </tr>
        </table>
        </td>
        </tr>
        </table>
        </center>

```



RunSchedule.jsp uses the Java bean MenuRunSchedule.class. The drop-down list box allows you to view the parameters in a given parameter set, as well as get the scheduled report output of the selected parameter set. The list box is constructed with the following forEach loop tags:

```
<td align="right" valign="middle" width=65%>&nbsp;  <select name="PARAMSET"
class="search">

    <c:forEach begin="0" end="\${param.SCHEDPARAMSETCOUNT-1}"
var="curIndex" >
    <c:choose>
    <c:when test="\${curIndex ==0}" >
    <option value="\<c:out value="\${curIndex}" />" selected<Parameter Set
1</option>
    </c:when>
    <c:otherwise>
    <option value="\<c:out value="\${curIndex}" />">Parameter Set <c:out
value="\${curIndex+1}" /> </option>
    </c:otherwise>
    </c:choose>
    </c:forEach>
    </select>
    <INPUT TYPE="hidden" NAME="SCHEDID" VALUE="\<c:out
value="\${param.SCHEDID}" />" />>

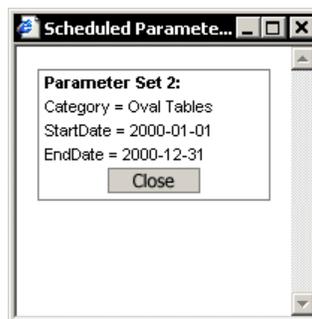
</td>
```



### Note

Here we use the begin/end version of the forEach syntax so that we can use an index to retrieve the item in the bean.

If you click on the *view* button, you will see the following window that shows the parameter sets parameter name/value pairs:



*Parameter Values Dialog*

The following Java script invokes ShowParameters.jsp to create this window:

```
function viewParamSet() {

    paramOption = document.showschedule.PARAMSET.value;
    windowLocation = "ShowParameters.jsp?SCHEDID=<c:out
value="\${param.SCHEDID}" /><c:forEach items="\${menuRunSchedule.paramNames}"
var="currentName"><c:out value="\&PARAMNAME=" escapeXml="false"/><c:out
value="\${currentName}" /></c:forEach&PARAMSET=" + paramOption;
```

```

window.open(windowLocation, "ShowParameters", "top=30, left=30,
toolbar=no, directories=no, location=no, status=no, menubar=no,
resizable=yes, scrollbars=yes, width=200, height=180");
}

```



### Note

`document.showschedule.PARAMSET.value` refers to `<select name="PARAMSET" class="search">` from the drop-down list above. The `forEach` loop tag in this Java script function creates the parameter set name/value pair by calling the getter method `paramNames`.

`ShowParameters.jsp` uses the Java bean `MenuScheduleParam.class`. The following `forEach` loop displays the parameter set values:

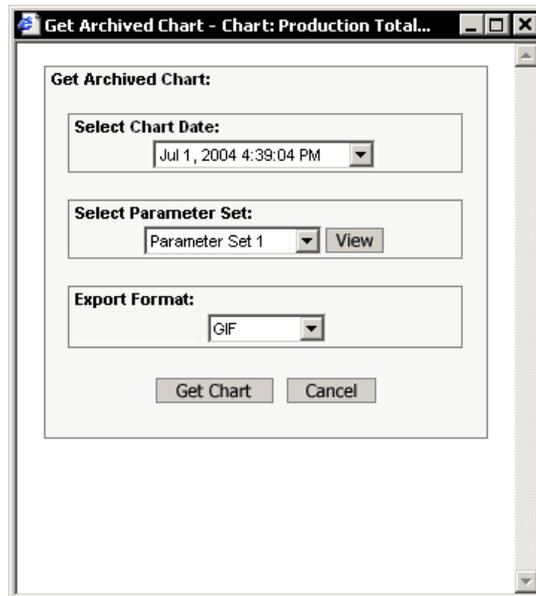
```

<c:forEach items="{menuScheduleParam.paramNameValue}" var="curItem" >
<tr>
<td class="text">&nbsp;<c:out value="{curItem}" /></td>
</tr>
</c:forEach>

```

## 8.5.6. Running Archived Reports and Charts

If you click on the archive icon of a chart, `RunArchiveCHT.jsp` will be invoked. The Java bean used is `MenuRunArchiveCHT.class`. A window similar to the following screen shot will pop up. The parameter set selection list box will be displayed or hidden depending on whether the chart has parameters or not.



*View Archive Dialog*

The parameter set selection code is similar to that in `RunSchedule.jsp` and will be not repeated here. The code that generates the archive selection box is shown below:

```

<c:set var="ct" value="{menuRunArchiveCHT.archiveFilesCount}" />
<c:forEach begin="0" end="{ct-1}" var="curIndex" >
<OPTION value="{c:out value="{menuRunArchiveCHT.reportName[curIndex]}" />"

```

```

<c:if test="{curIndex == 0}" >selected</c:if> ><c:out
  value="{menuRunArchiveCht.archiveDate[curIndex]}" />
</OPTION>
</c:forEach>
</select></td>

```

As for parameter set selection, the `forEach` tag syntax here uses `begin/end` to allow for indexing. Based on the archive date you select, the corresponding archive chart will be sent to the servlet, `RPT_Generator` to render the chart.

If you select to view the archive of a report, `MenuArchiveSecurity.jsp` will be invoked. It uses the Java bean `MenuArchiveSecurity.class`. The line of code of interest is `<%= menuRunArch.getOnLoad() %>`. The bean checks to see if you are of the same security level as the security level saved in the archive. You will be allowed to retrieve the archive on the following conditions: (1) you have no security level assigned to you and (2) you have the same security level as the one saved in the archived file. So, you will be directed to `ArchivePermission.html`, or run `RunArchive.jsp`. `RunArchive.jsp` uses the Java bean `MenuRunArchive.class`. You will notice that the code in `RunArchive.jsp` is very similar to that in `RunArchiveCht.jsp`. The only difference is the export format options and that you need to supply a security level in the request object for the servlet, `LookupServlet`, to render the report. The code to get and pass security level to the request object is as follows:

```

<c:if test="{menuRunArchive.securityLevel != null &&
  menuRunArchive.securityLevel != ''}">

    <tr>
    <td>
    <INPUT TYPE="hidden" NAME="SecurityLevel" VALUE="{c:out
  value="{menuRunArchive.securityLevel}" />" />
    </td>
    </tr>
  </c:if>

```

## 8.5.7. Getter Methods

The following section details all of the getter methods that are available in the Menu Java Bean classes:

### 8.5.7.1. MenuMainPage

The following getter methods are available in the class `MenuMainPage`:

```

//returns the path "/ERES/servlet" where ERES is the ERES install directory
  (without ").
public String getContextPath()

//get the array of folder nodes
public ArrayList getFolderList()

// if user not login yet, return "onload=\"top.location='Menu_Login.jsp'\"",
  otherwise return "".
public String getOnLoad()

//get the text "Welcome " + user name
public String getMenuTitle()

//for international language encoding, these two methods can be used in JSTL
  tag, instead of getEncodeText()
public void setLanguageEncodedText(String text)

```

---

```
public String getLanguageEncodedText()  
  
//for international language encoding. Text will be translated to language  
in the language translation file.  
public String getEncodeText(String str)
```

### 8.5.7.2. FolderNode

The following getter methods are available in the class FolderNode:

```
//get array of items in current menu folder node  
public ArrayList getItems()  
  
//get the name of current folder node  
public String getNodeName()  
  
// get ID of current folder node  
public int getNodeID()  
  
// get number of rows in the current folder node  
public int getRowCount()  
  
//get the tabs, one for each level down in the node hierarchy. You can use  
this to display folder nodes in staggered fashion  
public String getBufferTab()
```

### 8.5.7.3. ItemNode

The following getter methods are available in the class ItemNode:

```
//returns true if current item is a chart  
public boolean isChart()  
  
//returns true if current item is a report  
public boolean isReport()  
  
//returns true if current item has scheduled job  
public boolean isScheduleJobAvailable()  
  
//returns true if current item has archive job  
public boolean isArchiveJobAvailable()  
  
//get the schedule object of the current item  
public ItemScheduleJob getScheduleJob()  
  
//get the archive object of the current item  
public ItemArchiveJob getArchiveJob()  
  
//get current item's name  
public String getDocumentName()  
  
//get current item's url  
public String getDocumentURL()  
  
//get current item's file path  
public String getDocumentPath()
```

```
//get current item's description  
public String getDocumentDescription()  
  
//get the row ID of current item  
public int getRowID()  
  
//get security level of current item  
public String getSecurityLevel()  
  
//get the last modified date of current item  
public String getLastModified()
```

#### 8.5.7.4. ItemScheduleJob

The following getter methods are available in the class ItemScheduleJob:

```
//get the end date of item's schedule job  
public Date getEndDate()  
  
//get the start date of item's schedule job  
public Date getStartDate()  
  
//get the ID of the schedule job for current item  
public int getId()  
  
//get name of schedule job  
public String getJobName()  
  
// get the number of parameter set for this schedule job  
public int getParamSetCount()  
  
//get the next export time  
public Date getNextExportTime()  
  
//returns true if item is a chart  
public boolean isChart()  
  
//returns true if item is a report  
public boolean isReport()  
  
//returns location of report file  
public String getReportLocation()  
  
//return location of chart file  
public String getChartLocation()
```

#### 8.5.7.5. ItemArchiveJob

The following getter methods are available in the class ItemArchiveJob:

```
//returns true if archive item is a chart  
public boolean isChart()  
  
//returns true if archive item is a report  
public boolean isReport()
```

---

```

//get the number of parameter set for this archive job
public int getParamSetCount()

//get ID of archive job
public int getId()

//get the name of the archive job
public String getJobName()

```

### 8.5.7.6. MenuRunReport

The following getter methods are available in the class MenuRunReport:

```

//if user not login yet, returns the string: "onload=
\"top.location='Menu_Login.jsp'\
//use it to force users to login
//otherwise return "".
public String getOnLoad()

//returns the string: "Run " + name of item
public String getMenuTitle()

// get the path of servlet ReportURLReader to generate report
public String getReportURL()

//for international language encoding
//these two methods can be used in JSTL tag, instead of getEncodeText()
public void setLanguageEncodedText(String text)
public String getLanguageEncodedText()

//get the parameter prompts for the report/chart
public String getParamTable()

//get security level of report
public String getSecurityLevel()

//get the archive object ID
public int getArchID ()

//returns true if item has parameters
public boolean isParamReport ()

//for international language encoding. Text will be translated to language
in the language translation file.
public String getEncodeText(String str)

```

### 8.5.7.7. MenuRunChart

The following getter methods are available in the class MenuRunChart:

```

//if user not login yet, returns the string: "onload=
\"top.location='Menu_Login.jsp'\
//use it to force users to login
//otherwise return "".
public String getOnLoad()

//returns the string: "Run " + name of item

```

---

```

public String getMenuTitle()

// returns servlet path for RPT_Generator to generate chart
public String getChartURL()

// get parameter prompts for chart
public String getParamTable()

//URL for chart file
public String getDocumentURL()

//archive ID
public int getArchID()

//returns true if chart has parameters
public boolean isParamChart()

//for international language encoding
//these two methods can be used in JSTL tag, instead of getEncodeText()
public void setLanguageEncodedText(String text)
public String getLanguageEncodedText()

//for international language encoding. Text will be translated to language
in the language translation file.
public String getEncodeText(String str)

```

### 8.5.7.8. MenuRunSchedule

The following getter methods are available in the class MenuRunSchedule:

```

// if user did not login, return "onload=\"top.location='Menu_Login.jsp'\""
// otherwise return ""
public String getOnLoad()

//REPORT TITLE -- get name of item
public String getMenuTitle()

//get names of parameters of report
public String[] getParamNames()

//GENERATE FILE READER URL
public String getFileReader()

//for international language encoding. Text will be translated to language
in the language translation file.
public String getEncodeText(String str)

//for international language encoding
//these two methods can be used in JSTL tag, instead of getEncodeText()
public void setLanguageEncodedText(String text)
public String getLanguageEncodedText()

```

### 8.5.7.9. MenuReportSecurity

The following getter methods are available in the class MenuReportSecurity:

```

//if user has permission to view item and item has no parameters returns
"onload=\"runSchedule()\""
//else if user does not have permission to view item returns //"onload=
\"window.location='Web_Component/MENU/SchedulePermission.html'
//else if item has parameters returns "onload=
\"window.location='RunSchedule.jsp?' + queryString +"
//else returns ""
public String getOnLoad()

//get url of scheduled export file
public String getFileURL()

//for international language encoding. Text will be translated to language
in the language translation file.
public String getEncodeText(String str)

```

### 8.5.7.10. MenuRunArchive

The following getter methods are available in the class MenuRunArchive:

```

//if user did not login, return "onload=\"top.location='Menu_Login.jsp'\""
//otherwise return ""
public String getOnLoad()

//REPORT TITLE - returns "Report: " + archive object's name
public String getMenuTitle()

//get names of parameter names of report
public String[] getParamNames()

//get url of servlet ReportURLReader to generate report
public String getReportURL()

//get name of report
public ArrayList getReportName()

//get date of archive file
public ArrayList getArchiveDate()

//get number of archive files for current item
public int getArchiveFilesCount()

//get number of parameter sets
public int getParamSetCount()

//get security level of report
public String getSecurityLevel()

//for international language encoding. Text will be translated to language
in the language translation file.
public String getEncodeText(String str)

//for international language encoding
//these two methods can be used in JSTL tag, instead of getEncodeText()
public void setLanguageEncodedText(String text)
public String getLanguageEncodedText()

```

### 8.5.7.11. MenuRunArchiveCHT

The following getter methods are available in the class MenuRunArchiveCHT:

```
//if user did not login, return "onload=\"top.location='Menu_Login.jsp'\""
//otherwise return ""
public String getOnLoad()

//returns "Chart: " + archive object name
public String getMenuTitle()

//Get url for the servlet RPT_Generator to generate chart
public String getChartURL()

//get the names of archived files
public ArrayList getReportName()

//get the archive dates of the archive items
public ArrayList getArchiveDate()

//get the archive file count
public int getArchiveFilesCount()

//get the number of parameter sets
public int getParamSetCount()

//for international language encoding. Text will be translated to language
//in the language translation file.
public String getEncodeText(String str)

//for international language encoding
//these two methods can be used in JSTL tag, instead of getEncodeText()
public void setLanguageEncodedText(String text)
public String getLanguageEncodedText()
```

### 8.5.7.12. MenuArchiveSecurity

The following getter methods are available in the class MenuArchiveSecurity:

```
//if user has different security level as that save in the
//report returns //"onload=\"window.location='Web_Component/MENU/
//ArchivePermission.html'
//otherwise returns "onload=\"window.location='RunArchive.jsp?' +
//queryString +"
public String getOnLoad()

//for international language encoding. Text will be translated to language
//in the language translation file.
public String getEncodeText(String str)
```

### 8.5.7.13. MenuScheduleParam

The following getter methods are available in the class MenuScheduleParam:

```
//if user did not login, return "onload=\"top.location='Menu_Login.jsp'\""
//otherwise return ""
public String getOnLoad()
```

```

//get "Parameter Set " + (paramIndex+1)
public String getName()

//get parameter name/value pairs for selected parameter set
public ArrayList getParamNameValue()

//for international language encoding. Text will be translated to language
in the language translation file.
public String getEncodeText(String str)

//for international language encoding
//these two methods can be used in JSTL tag, instead of getEncodeText()
public void setLanguageEncodedText(String text)
public String getLanguageEncodedText()

```

## 8.6. Organizer

### 8.6.1. Introduction

ERES Organizer can also be called, using the API, with custom choices instead of the default options provided. ERES Organizer can be customized by using the API, both in terms of look and feel and function. Report and Chart Designer, called from the Organizer, can also be customized via the API.

Both `ERESOrganizer.jar` (located in the `ERES/lib` directory) and `ERESServer.jar` (located in the `ERES/WEB-INF/lib` directory) need to be added to the `CLASSPATH` for any code using ERES Organizer. The following snippet shows how to connect to ERES Organizer:

```

QbOrganizer.setServletRunner("http://localhost:8080");
QbOrganizer.setServletContext("ERES/servlet");
QbOrganizer organizer = new QbOrganizer(null, "admin", "admin");

```

The above code sets the connection information to connect to ERES Server and provides the username and password (in the above example, `admin` and `admin`) for ERES Organizer.

ERES Organizer can then be shown by getting a handle to the Organizer user interface and setting it visible.

```

JFrame frame = organizer.getOrganizerUI();
frame.setVisible(true);

```



#### Note

You need to copy the contents of the `images`, `orgimages`, `reportimages`, and `backgroundImages` directories to the working directory of the `.class` file.

All examples and code given in the manual assume that ERES server is running locally (i.e. on your local machine) and on port 8080. You can change this by going to the source code (you can download the source code by clicking on the *Full Source Code* link in the corresponding chapter), editing the code to enter the new connection information and recompiling the code.

Also, note that if you have applets that use ERES Report API, the browser must have at least a 1.5 JVM plug in.

### 8.6.2. Look and Feel

You can specify a look and feel to ERES Organizer before getting a handle to the user interface. By default, ERES Organizer uses the system's look and feel. The look and feel can be set using the following method in `QbOrganizer`:

```
public static void setLookAndFeel(javax.swing.LookAndFeel newLookAndFeel);
```

### 8.6.3. Inserting and Removing Items

You do not have to show ERES Organizer in order to insert and remove items. You can add and delete items from Organizer via the API.

You would use the following method to add an item to the Organizer:

```
public void insertNewItem(String name, String description, String location,
    String url, String folderName);
```

For example:

```
organizer.insertNewItem("Orders_Breakdown",
    "This is an example report added to Organizer",
    "OrdersBreakdown.qrp",
    "http://localhost:8080/ERES/OrdersBreakdown.qrp",
    "/New Project");
```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/InsertingItemERES.zip> ]

Result Screenshot [ <http://data.quadbase.com/Docs70/help/manual/images/InsertingItem.gif> ]

The above code inserts a new item called `Account_Report` in the Organizer under the `/ReportFiles` project folder. The URL and physical location of the file is also passed along with a short description. Please note that to run the above code successfully, you will need to create a project called `ReportFiles` in Organizer. The recommended way to run the above code is to compile it, then go to the root ERES directory and enter the following command:

```
java -classpath ".../src/er/bin;./lib/ERESOrganizer.jar;./
lib/WEB-INF/ERESServer.jar;./lib/qblicense.jar;./lib/
hsqldb.jar;&lt;PATH_TO_COMPILED_CLASS_FILE&gt;" InsertingItem
```

You can also remove items from the Organizer using the following method:

```
public void removeItem(String name, String folderName);
```

For example:

```
organizer.removeItem("Orders_Breakdown", "/New Project");
```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/RemovingItem.zip> ]

Result Screenshot [ <http://data.quadbase.com/Docs70/help/manual/images/RemovingItem.gif> ]

The above code deletes `Account_Report` from the `/ReportFiles` project folder in ERES Organizer. The recommended way to run the above code is to compile the source code, then go to the root ERES directory and enter the following command:

```
java -classpath ".../src/er/bin;./lib/ERESOrganizer.jar;./
lib/WEB-INF/ERESServer.jar;./lib/qblicense.jar;./lib/
hsqldb.jar;&lt;PATH_TO_COMPILED_CLASS_FILE&gt;" RemovingItem
```

## 8.6.4. Customizing Report and Chart Designer

You can also customize the properties of Report Designer and Chart Designer invoked from the custom Organizer. You can add or remove items from the menu bars and also override the Save function. Please note that when the Save function is overridden, any files saved with a relative path are not relative to the ERES root directory but relative to the working directory of the .class file.

The following example, which can be run as a Java Web Start Application, shows a custom Organizer whose Report Designer's and Chart Designer's Save function are overridden:

```
QbOrganizer.setServletRunner("http://localhost:8080");
QbOrganizer.setServletContext("ERES/servlet");
QbOrganizer organizer = new QbOrganizer(null, "admin", "admin");

// modify organizer's reportdesigner
organizer.setReportDesignerHandle(new ReportDesignerHandle());

// modify organizer's chartdesigner
organizer.setChartDesignerHandle(new ChartDesignerHandle());

// set organizer visible
JFrame frame = organizer.getOrganizerUI();
frame.setVisible(true);

public static class ReportDesignerHandle implements IReportHandle {

    public ReportDesignerHandle() {}

    public void processDesigner(QbReportDesigner designer) {

        System.out.println("PROCESS REPORT...");

        // modify warning message before inserting drill-down layer
        designer.modifyWarningMessage(designer.SAVE_RPT_BEFORE_DRILLDOWN, "The
report designer will save the current report for you, continue?");

        // modify warning message before inserting subreport
        designer.modifyWarningMessage(designer.SAVE_RPT_BEFORE_SUBREPORT, "The
report designer will save the current report for you, continue?");

        // customize report IO
        designer.setReportIO(new ReportIO(designer));

        // by pass SAVE AS IO
        designer.setByPassSaveAsIO(new ByPassSaveAsForReport());

        // modify reportdesigner's chartdesigner
        designer.setChartDesignerHandle(new ChartDesignerHandle());

        // remove "SAVE AS" option for Report Designer
        JMenu fileMenu = designer.getReportMenuBar().getMenu(0);
        fileMenu.remove(6);

    }
}
```

```

}

public static class ReportIO implements IReportNavigationIO {

    String fileName = null;
    QbReportDesigner designer = null;

    public ReportIO(QbReportDesigner designer) { this.designer =
designer; }

    public void saveFile(byte[] data, String fileName) {

        System.out.println("SAVE REPORT FILE - " + fileName);
        try {

            FileOutputStream fout = new FileOutputStream(fileName);
            fout.write(data, 0, data.length);
            fout.close();

        } catch (Exception ex) {

            ex.printStackTrace();

        }

        this.fileName = fileName;

    }

    // if parent report is saved, return saved report file location
    // else return null;
    public String isParentReportSaved() {

        System.out.println("IS PARENT REPORT SAVED = NO...");
        return null;

    }

    // overridden save parent report method and return parent report
location
    public String saveParentReport() {

        System.out.println("RETURN PARENT REPORT LOCATION");
        return fileName;

    }

}

public static class ByPassSaveAsForReport implements IByPassSaveAsForReport
{

    public ByPassSaveAsForReport() {};

    public String getFileName(String originalFileName) {

        System.out.println("BY PASS REPORT SAVE AS OPTION...");
        if (originalFileName == null) return "TEMP_REPORT_FILE.rpt";
    }
}

```

```

        else return originalFileName;

    }

    public Properties getSaveAsProperties(String originalFileName) {

        return new Properties();

    }

}

public static class ChartDesignerHandle implements IChartHandle {

    public ChartDesignerHandle() {};

    public void processDesigner(QbChartDesigner designer) {

        System.out.println("PROCESS CHART....");

        // remove "SAVE AS" option for Chart Designer
        JMenu fileMenu = designer.getChartMenuBar().getMenu(0);
        fileMenu.remove(6);

        designer.setChartIO(new ChartIO());
        designer.setByPassSaveAsIO(new ByPassSaveAsForChart());

    }

}

public static class ChartIO implements IChartIO {

    public ChartIO() {};

    public String saveChartFile(byte[] data, String fileName) {

        System.out.println("SAVE CHART FILE - " + fileName);
        try {

            FileOutputStream fout = new FileOutputStream(fileName);
            fout.write(data, 0, data.length);
            fout.close();

        } catch (Exception ex) {

            ex.printStackTrace();

        }

        return fileName;

    }

}

public static class ByPassSaveAsForChart implements IByPassSaveAsForChart {

```

```

public ByPassSaveAsForChart() {};

public String getFileName(String originalFileName) {

    System.out.println("BY PASS CHART SAVE AS OPTION...");
    if (originalFileName == null) return "TEMP_CHART_FILE.cht";
    else return originalFileName;

}

public Properties getSaveAsProperties(String originalFileName) {

    return new Properties();

}

}

```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/CustomOrganizerERES.zip> ]

The recommended way to run the above code is to compile the source code, then go to the root ERES directory and enter the following command:

```

java -classpath "../src/er/bin;./lib/ERESOrganizer.jar;./lib/WEB-INF/ERESserver.jar;./lib/qblicense.jar;./lib/hsqldb.jar;&lt;PATH_TO_COMPILED_CLASS_FILE&gt;" CustomOrganizer

```

## 8.6.5. Calling Scheduler

You can also get a handle to ERES Scheduler using the following method:

```

Scheduler scheduler = organizer.getScheduler();

```

You can add schedules to and remove schedules from the Scheduler via the API. For more on this, please refer to Section 8.7 - Scheduler.

## 8.6.6. Javadoc

Javadoc for the entire API is provided along with ERES. It is located here (Reports and charts) [ <http://data.quadbase.com/Docs70/eres/help/apidocs/index.html> ] and here (ES functions) [ <http://data.quadbase.com/Docs70/eres/help/ERESapidocs/index.html> ].

## 8.6.7. Summary

ERES API provides an easy-to-use and powerful API to customize and show Organizer interface as well as customize Report Designer and Chart Designer interfaces. You can also write code to insert and delete items in Organizer.

Please note that the API requires a JDK 1.8 or above. The ERES API has been tested on Windows, Solaris, Linux, AIX, HP, and Mac platforms.

# 8.7. Scheduler

## 8.7.1. Introduction

ERES has a scheduler interface through which charts and reports can be archived/exported at a specific times or at specific intervals. This allows ERES to handle mundane tasks such as exporting and archiving charts and reports

rather than having a user log in at a specific time and explicitly perform the operation. For more information on how to use Scheduler, please see Section 2.2 - Scheduling & Archiving.

Both `ERESOrganizer.jar` (in the `<ERES>/lib` directory) and `ERESServer.jar` (in the `<ERES>/WEB-INF/lib` directory) need to be added to the `CLASSPATH` for any code using the Scheduler. The following snippet shows how to connect to Scheduler:

```
QbOrganizer.setServletRunner("http://localhost:8080");
QbOrganizer.setServletContext("ERES/servlet");
QbOrganizer organizer = new QbOrganizer(null, "admin", "admin");
```

The above code sets the connection information to connect to ERES Server and provides the username and password (in the above example, `admin` and `admin`) for Scheduler.

All examples and code given in the manual assume that ERES server is running locally (i.e. on your local machine) and on port 8080. You can change this by going to the source code (you can download the source code by clicking on the *Full Source Code* link in the corresponding chapter), editing the code to enter the new connection information and recompiling the code.

## 8.7.2. Scheduling an Export

You can schedule a chart/report to be exported using Scheduler. The following code, given below, shows how to do this:

```
QbOrganizer.setServletContext(ServletContext);
QbOrganizer.setServletRunner(ServletRunner);
QbOrganizer organizer = new QbOrganizer(null, "admin", "admin");

QbSchedulePackage sPack = new QbSchedulePackage("Report_OneTime",
    organizer);
sPack.setReportExportType(IExportConstants.DHTML);
sPack.setTaskOption(QbSchedulePackage.ONE_TIME);
Calendar calendar = Calendar.getInstance();
calendar.add(Calendar.MINUTE, 1);
sPack.setStartDate(calendar.getTimeInMillis());
sPack.setSendEmail(false);

QbScheduleObject sObj = new QbScheduleObject("report1",
    QbScheduleObject.TEMPLATE_TYPE_REPORT, sPack);
sObj.setFileLocation("Account.rpt");
String exportLocation = sObj.pickDefaultExportLocation();

organizer.getScheduler().addSchedulePackageTask(sPack);
```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/SchedulingExportReportOne-TimeERES.zip> ]



### Note

Any references to a data source or a report file using the relative URL reference (e.g. `help/examples/ReportGallery/templates/Account.rpt`) are relative to the directory from where ERES is running in.

The above code schedules a report (`Account.rpt`) to be exported to DHTML, in the default export directory and sets the export to take place once and in one minute.

You can specify a chart file by changing the `QbScheduleObject.TEMPLATE_TYPE_REPORT` constant in the `QbSchedulerObject` constructor to `QbScheduleObject.TEMPLATE_TYPE_REPORT`. Please note that

you will also need to specify a .cht or .tpl file as well as a valid export type (PNG, JPEG, GIF, etc) using the setChartExportType(int exportType) method. The following code shows how to modify the above code to schedule a chart:

```
QbSchedulePackage sPack = new QbSchedulePackage("Chart_OneTime", organizer);
sPack.setChartExportType(IExportConstants.GIF);
sPack.setTaskOption(QbSchedulePackage.ONE_TIME);
Calendar calendar = Calendar.getInstance();
calendar.add(Calendar.MINUTE, 1);
sPack.setStartDate(calendar.getTimeInMillis());
sPack.setSendEmail(false);

QbScheduleObject sObj = new QbScheduleObject("chart1",
    QbScheduleObject.TEMPLATE_TYPE_CHART, sPack);
sObj.setFileLocation("col2d.tpl");
String exportLocation = sObj.pickDefaultExportLocation();

organizer.getScheduler().addSchedulePackageTask(sPack);
```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/SchedulingExportChartOne-TimeERES.zip> ]

Both examples, given above, show how to schedule a chart or a report for a one-time export. The examples below show how to set up a periodic schedule:

```
QbSchedulePackage sPack = new QbSchedulePackage("Report_Periodically",
    organizer);
sPack.setReportExportType(IExportConstants.DHTML);
sPack.setTaskOption(QbSchedulePackage.TIME_INTERVAL);

/** every 5 mins ***/
sPack.setIntervalType(QbSchedulePackage.TIME);
sPack.setTimeInterval(5); // export every 5 mins

/** every day ***/
//sPack.setIntervalType(QbSchedulePackage.DAYS);
//sPack.setDayInterval(1); // export everyday

Calendar calendar = Calendar.getInstance();
calendar.add(Calendar.MINUTE, 5);
sPack.setStartDate(calendar.getTimeInMillis());
Calendar calendar2 = Calendar.getInstance();
calendar2.add(Calendar.MINUTE, 26);
sPack.setEndDate(calendar2.getTimeInMillis());
sPack.setSendEmail(false);

QbScheduleObject sObj = new QbScheduleObject("report1",
    QbScheduleObject.TEMPLATE_TYPE_REPORT, sPack);
sObj.setFileLocation("Account.rpt");
String exportLocation = sObj.pickDefaultExportLocation();

organizer.getScheduler().addSchedulePackageTask(sPack);
```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/SchedulingExportReportPeriodically-ERES.zip> ]

The above code snippet shows how to schedule a job to run every five minutes (or everyday depending on which section of the code you comment).

The following code snippet shows to schedule a job to run at certain time periods:

```

QbSchedulePackage sPack = new QbSchedulePackage("Report_Time", organizer);
sPack.setReportExportType(IExportConstants.PDF);
sPack.setTaskOption(QbSchedulePackage.FIXED_DAYS);

/** export every 2 hrs. Monday through Friday */
sPack.setSpecifyDays(new int[] { 1, 2, 3, 4, 5 });
sPack.setStartTime(9 * 60); // 9:00AM in minutes
sPack.setEndTime(22 * 60); // 10:00PM in minutes
sPack.setTimeInterval(2 * 60); // export every 2 hours

/** export at specific times on specific dates */
//sPack.setSpecifyDates(new int[]{26,27,28,29}); // export on specific dates
//sPack.setSpecifyTime(new int[]{16 * 60, 17 * 60, 18 * 60}); // 4PM, 5PM
// and 6PM in minute

Calendar calendar = Calendar.getInstance();
calendar.add(Calendar.MINUTE, 5);
sPack.setStartDate(calendar.getTimeInMillis());
sPack.setEndDate(-1); // run indefinitely
sPack.setSendEmail(false);

QbScheduleObject sObj = new QbScheduleObject("report1",
    QbScheduleObject.TEMPLATE_TYPE_REPORT, sPack);
sObj.setFileLocation("Account.rpt");
String exportLocation = sObj.pickDefaultExportLocation();

organizer.getScheduler().addSchedulePackageTask(sPack);

```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/SchedulingExportReport-TimeERES.zip> ]

You can also specify the parameters when scheduling a parameterized chart or report to be exported. The following code shows how:

```

QbSchedulePackage sPack = new QbSchedulePackage("ParamReport_OneTime",
    organizer);
sPack.setReportExportType(IExportConstants.DHTML);
sPack.setTaskOption(QbSchedulePackage.ONE_TIME);
Calendar calendar = Calendar.getInstance();
calendar.add(Calendar.MINUTE, 5);
sPack.setStartDate(calendar.getTimeInMillis());
sPack.setSendEmail(false);

QbScheduleObject sObj = new QbScheduleObject("report1",
    QbScheduleObject.TEMPLATE_TYPE_REPORT, sPack);
//Single Param Report
sObj.setFileLocation("EmployeeDetails.pak");
String exportLocation = sObj.pickDefaultExportLocation();
//1st param set {"Denise Carron"}
Object tmp1[] = { "Denise Carron" };
//2nd param set {"Frank Carnody"}
Object tmp2[] = { "Frank Carnody" };

```

```

Vector paramList = new Vector();
paramList.addElement(tmp1);
paramList.addElement(tmp2);
sObj.setParamList(paramList);

organizer.getScheduler().addSchedulePackageTask(sPack);

```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/SchedulingExportParamReportOne-TimeERES.zip> ]

You can also specify the schedules to be sent to one or more email addresses, either as a link or an attachment. The following code shows how to send a scheduled job which contains a non-parameterized report to multiple recipients:

```

QbSchedulePackage sPack = new QbSchedulePackage("Report_Email", organizer);
sPack.setReportExportType( IExportConstants.DHTML );
sPack.setTaskOption(QbSchedulePackage.ONE_TIME);
Calendar calendar = Calendar.getInstance();
calendar.add(Calendar.MINUTE, 10);
sPack.setStartDate(calendar.getTimeInMillis());
sPack.setSendEmail(true);
sPack.setFromAddress("user1@quadbase.com");
sPack.setSubject("Report Email Account Report");
sPack.setBodyText("Scheduled Export of Account from user1");
sPack.setEmailType(QbSchedulePackage.ASATTACHMENT);

QbScheduleObject sObj = new QbScheduleObject("report1",
    QbScheduleObject.TEMPLATE_TYPE_REPORT, sPack);
sObj.setFileLocation("help/examples/ReportGallery/templates/Account.rpt");
String exportLocation = sObj.pickDefaultExportLocation();
sObj.setToAddresses(new String[]{"user2@quadbase.com",
    "user3@quadbase.com"});

organizer.getScheduler().addSchedulePackageTask(sPack);

```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/SchedulingExportReportEmail-ERES.zip> ]

Similarly, emails can be sent for each parameter set chosen for a parameterized report/chart when scheduling an export. Note that at least one email address must be specified for each parameter set and the same recipient can be used for multiple parameter sets.

```

QbSchedulePackage sPack = new QbSchedulePackage("ParamReport_Email",
    organizer);
sPack.setReportExportType( IExportConstants.DHTML );
sPack.setTaskOption(QbSchedulePackage.ONE_TIME);

Calendar calendar = Calendar.getInstance();
calendar.add(Calendar.MINUTE, 5);
sPack.setStartDate(calendar.getTimeInMillis());
sPack.setSendEmail(true);
sPack.setFromAddress("user1@quadbase.com");
sPack.setSubject("Param Report Email Account Report");
sPack.setBodyText("Scheduled Export of Employee Details from user1");
sPack.setEmailType(QbSchedulePackage.ASATTACHMENT);

QbScheduleObject sObj = new QbScheduleObject("report1",

```

```

    QbScheduleObject.TEMPLATE_TYPE_REPORT, sPack);
//Single Param Report
sObj.setFileLocation("EmployeeDetails.pak");
String exportLocation = sObj.pickDefaultExportLocation();
//1st param set {"Denise Carron"}
Object tmp1[] = { "Denise Carron" };
//2nd param set {"Frank Carnody"}
Object tmp2[] = { "Frank Carnody" };
Vector paramList = new Vector();
paramList.addElement(tmp1);
paramList.addElement(tmp2);
sObj.setParamList(paramList);
Vector paramNameList = new Vector();
paramNameList.addElement("Denise");
paramNameList.addElement("Frank");
sObj.setParamList(paramNameList, paramList);

Hashtable toAddr = new Hashtable();
toAddr.put("Denise", new String[] { "user2@quadbase.com" });
toAddr.put("Frank", new String[] { "user3@quadbase.com" });
sObj.setParamAddresses(toAddr);

organizer.getScheduler().addSchedulePackageTask(sPack);

```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/SchedulingExportParamReportEmail-ERES.zip> ]

You can also set up the schedule to send the export to a FTP location by providing the details. The following code shows how:

```

QbSchedulePackage sPack = new QbSchedulePackage("FTPReport_OneTime",
    organizer);
sPack.setReportExportType(IExportConstants.DHTML);
sPack.setTaskOption(QbSchedulePackage.ONE_TIME);
Calendar calendar = Calendar.getInstance();
calendar.add(Calendar.MINUTE, 1);
sPack.setStartDate(calendar.getTimeInMillis());
sPack.setFtpDelivery(true);
sPack.setFtpHost("someMachine.com");
sPack.setFtpFilePath("somePath");
sPack.setFtpUserName("user1");
sPack.setFtpPassword("password1");

QbScheduleObject sObj = new QbScheduleObject("report1",
    QbScheduleObject.TEMPLATE_TYPE_REPORT, sPack);
sObj.setFileLocation("Account.rpt");
String exportLocation = sObj.pickDefaultExportLocation();

organizer.getScheduler().addSchedulePackageTask(sPack);

```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/SchedulingFTPReportOne-TimeERES.zip> ]

You can also request the schedule to send the export to a printer. The following code shows how:

```

QbSchedulePackage sPack = new QbSchedulePackage("PrintReport_OneTime",
    organizer);

```

```

sPack.setReportExportType(IExportConstants.DHTML);
sPack.setTaskOption(QbSchedulePackage.ONE_TIME);
Calendar calendar = Calendar.getInstance();
calendar.add(Calendar.MINUTE, 10);
sPack.setStartDate(calendar.getTimeInMillis());
sPack.setPrinterDelivery(true);
sPack.setSelectedPrinter("somePrinter");
sPack.setPrintFromPage(0);
sPack.setPrintToPage(-1);

QbScheduleObject sObj = new QbScheduleObject("report1",
    QbScheduleObject.TEMPLATE_TYPE_REPORT, sPack);
sObj.setFileLocation("Account.rpt");
String exportLocation = sObj.pickDefaultExportLocation();

organizer.getScheduler().addSchedulePackageTask(sPack);

```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/SchedulingPrinterReportOneTimeERES.zip> ]

### 8.7.3. Scheduling Multiple Exports

Every schedule job can contain any combination of reports and/or charts. The following code, shows how to schedule a report and a chart in a single schedule:

```

QbSchedulePackage sPack = new QbSchedulePackage("Package_OneTime",
    organizer);
sPack.setReportExportType(IExportConstants.PDF);
sPack.setChartExportType(IExportConstants.PNG);
sPack.setTaskOption(QbSchedulePackage.ONE_TIME);
Calendar calendar = Calendar.getInstance();
calendar.add(Calendar.MINUTE, 1);
sPack.setStartDate(calendar.getTimeInMillis());
sPack.setSendEmail(false);

QbScheduleObject rptObj = new QbScheduleObject("new project/Account",
    QbScheduleObject.TEMPLATE_TYPE_REPORT, sPack);
rptObj.setFileLocation("Account.rpt");
String exportLocation1 = rptObj.pickDefaultExportLocation();

QbScheduleObject chtObj = new QbScheduleObject("new project/col2d",
    QbScheduleObject.TEMPLATE_TYPE_CHART, sPack);
chtObj.setFileLocation("col2d.tpl");
String exportLocation2 = chtObj.pickDefaultExportLocation();

organizer.getScheduler().addSchedulePackageTask(sPack);

```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/SchedulingPackageOneTimeERES.zip> ]



#### Note

Any references to a data source or a report file, using the relative URL reference (e.g. `help/examples/ReportGallery/templates/Account.rpt`) are relative to the directory from where ERES is running in.

The above code schedules a package containing a report (`Account.rpt`) and a chart (`col2d.tpl`) to be exported to PDF (report) and PNG (chart), in the default export directory and sets the export to take place once and in one minute.

Each report and chart in a schedule can have different email recipients. If the report/chart is parameterized, each parameter set can have different email recipients. The following code shows how to set email recipients for both a non-parameterized report and a parameterized report in a single schedule:

```
QbSchedulePackage sPack = new QbSchedulePackage("ParamPackage_Email",
    organizer);
sPack.setReportExportType( IExportConstants.PDF );
sPack.setTaskOption(QbSchedulePackage.ONE_TIME);
Calendar calendar = Calendar.getInstance();
calendar.add(Calendar.MINUTE, 10);
sPack.setStartDate(calendar.getTimeInMillis());
sPack.setSendEmail(true);
sPack.setFromAddress("user1@quadbase.com");
sPack.setSubject("Schedule Param Package Email");
sPack.setBodyText("Schedule Package export containing a report and a
    parameterized report\n"
    +
    "<SAVED_LINKS></SAVED_LINKS>");
sPack.setEmailType(QbSchedulePackage.NONE);

QbScheduleObject rptObj1 = new QbScheduleObject("new project/Account",
    QbScheduleObject.TEMPLATE_TYPE_REPORT, sPack);
rptObj1.setFileLocation("Account.rpt");
String exportLocation1 = rptObj1.pickDefaultExportLocation();
rptObj1.setToAddresses(new String[] { "user2@quadbase.com" });

QbScheduleObject rptObj2 = new QbScheduleObject("new project/
EmployeeDetails",
    QbScheduleObject.TEMPLATE_TYPE_REPORT, sPack);
rptObj2.setFileLocation("EmployeeDetails.pak");
String exportLocation2 = rptObj2.pickDefaultExportLocation();
//1st param set {"Denise Carron"}
Object tmp1[] = { "Denise Carron" };
//2nd param set {"Frank Carnody"}
Object tmp2[] = { "Frank Carnody" };
Vector paramList = new Vector();
paramList.addElement(tmp1);
paramList.addElement(tmp2);

Vector paramNameList = new Vector();
paramNameList.addElement("Denise");
paramNameList.addElement("Frank");
rptObj2.setParamList(paramNameList, paramList);

Hashtable toAddr = new Hashtable();
toAddr.put("Denise", new String[] { "user2@quadbase.com" });
toAddr.put("Frank", new String[] { "user3@quadbase.com" });
rptObj2.setParamAddresses(toAddr);

organizer.getScheduler().addSchedulePackageTask(sPack);
```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/SchedulingParamPackageEmail-ERES.zip> ]

The code above also shows how to use a runtime variable (<SAVED\_LINKS>) in the email body. For more information about runtime variables see Section 2.2.1.2.1 - Email Delivery Options.

## 8.7.4. Scheduling an Archive

Along with scheduling exports, you can also schedule archiving of reports and charts. The following code shows how to do that:

```
QbSchedulePackage sPack = new QbSchedulePackage("ArchiveReport_OneTime",
    organizer);
sPack.setArchive(true);
sPack.setTaskOption(QbSchedulePackage.ONE_TIME);
Calendar calendar = Calendar.getInstance();
calendar.add(Calendar.MINUTE, 5);
sPack.setStartDate(calendar.getTimeInMillis());
sPack.setSendEmail(false);

QbScheduleObject sObj = new QbScheduleObject("report1",
    QbScheduleObject.TEMPLATE_TYPE_REPORT, sPack);
sObj.setFileLocation("Account.rpt");
String exportLocation = sObj.pickDefaultExportLocation();

organizer.getScheduler().addSchedulePackageTask(sPack);
```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/SchedulingArchiveReportOne-TimeERES.zip> ]

An archive job can also contain any combination of reports and/or charts. The following code shows how to schedule archiving of a report and a chart:

```
QbSchedulePackage sPack = new QbSchedulePackage("ArchivePackage_OneTime",
    organizer);
sPack.setArchive(true);
sPack.setTaskOption(QbSchedulePackage.ONE_TIME);
Calendar calendar = Calendar.getInstance();
calendar.add(Calendar.MINUTE, 1);
sPack.setStartDate(calendar.getTimeInMillis());
sPack.setSendEmail(false);

QbScheduleObject rptObj = new QbScheduleObject("new project/Account",
    QbScheduleObject.TEMPLATE_TYPE_REPORT, sPack);
rptObj.setFileLocation("Account.rpt");
String exportLocation1 = rptObj.pickDefaultExportLocation();

QbScheduleObject chtObj = new QbScheduleObject("new project/col2d",
    QbScheduleObject.TEMPLATE_TYPE_CHART, sPack);
chtObj.setFileLocation("col2d.tpl");
String exportLocation2 = chtObj.pickDefaultExportLocation();

organizer.getScheduler().addSchedulePackageTask(sPack);
```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/SchedulingArchivePackageOne-TimeERES.zip> ]

## 8.7.5. Removing a Schedule

In addition to adding schedules, you can also use the API to remove any schedules. This is done by getting a list of all the jobs scheduled and then deleting a particular job.

The following code shows how to delete a particular job:

```
Vector schedList = organizer.getScheduler().getSchedulePackageList();

QbSchedulePackage sPack = (QbSchedulePackage) schedList.elementAt(0);
organizer.getScheduler().removeSchedulePackageTask(sPack);
```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/RemovingScheduleERES.zip> ]

Note that the above code removes the first job in the schedule list. The job may not be visible if it has been already marked for deletion. The recommended approach is to go through the vector and search each `QbScheduleObject` by the name and then to delete the correct one.

### 8.7.5.1. Getting Details of a Failed Schedule

Sometimes, a schedule job may fail for some reason or another. ERES Server keeps a track of all the failed jobs so that its details can be obtained later. The following code shows how to obtain the details:

```
Vector vec = organizer.getScheduler().getFailedScheduledJob();
for (int i = 0; i < vec.size(); i++) {
    FailedScheduledJob obj = (FailedScheduledJob) vec.elementAt(i);
    System.out.println("FAIL # " + i + ": Name = " +
        obj.getScheduledJobName());
    System.out.println("File Location = " + obj.getFileLocation());
    System.out.println("Time = " + obj.getExportTime());
    System.out.println("StackTrace = " + obj.getStackTrace());
}
```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/FailedScheduleERES.zip> ]

You can also set up an email to be sent in case the schedule fails. The following code illustrates how:

```
QbSchedulePackage sPack = new QbSchedulePackage("Package_FailedEmail",
    organizer);
sPack.setReportExportType(IExportConstants.PDF);
sPack.setChartExportType(IExportConstants.PNG);
sPack.setTaskOption(QbSchedulePackage.ONE_TIME);
Calendar calendar = Calendar.getInstance();
calendar.add(Calendar.MINUTE, 10);
sPack.setStartDate(calendar.getTimeInMillis());

sPack.setSendEmail(true);
sPack.setFromAddress("user1@quadbase.com");
sPack.setSubject("Schedule Package Email");
sPack.setBodyText("Schedule Package export containing a report and chart");
sPack.setFailSubject("Package_FailedEmail schedule failed");
sPack.setFailBodyText("Cannot send report and/or charts from
    Package_FailedEmail");
sPack.setFailToAddress("user1@quadbase.com");
sPack.setEmailType(QbSchedulePackage.ASATTACHMENT);

QbScheduleObject rptObj = new QbScheduleObject("new project/Account",
    QbScheduleObject.TEMPLATE_TYPE_REPORT, sPack);
rptObj.setFileLocation("Account.rpt");
String exportLocation1 = rptObj.pickDefaultExportLocation();
rptObj.setToAddresses(new String[] { "user2@quadbase.com" });
```

```

QbScheduleObject chtObj = new QbScheduleObject("new project/col2d",
    QbScheduleObject.TEMPLATE_TYPE_CHART, sPack);
chtObj.setFileLocation("col2d.tpl");
String exportLocation2 = chtObj.pickDefaultExportLocation();
chtObj.setToAddresses(new String[] { "user3@quadbase.com" });

organizer.getScheduler().addSchedulePackageTask(sPack);

```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/SchedulingPackageFailedEmail-ERES.zip> ]

### 8.7.5.2. ICallbackScheduler Interface

You can also create an additional code that performs certain actions based on the Scheduler performing a job (regardless of if it was successful). This code implements the `ICallbackScheduler` interface and specifies what to do when a job has been successfully completed or not.

Please note that after creating the code you need to change `QB.properties` (in the `<ERES-installation-directory>/WEB-INF/classes` directory) and add the following argument to the `ServerCommands=` line:

```
-SchedulerCallbackClass:<name of class file implementing ICallbackScheduler>
```

The following code shows how to use the `ICallbackScheduler` interface:

```

import quadbase.reportorganizer.organizerAPI.ICallbackScheduler;
import quadbase.reportorganizer.organizerAPI.QbScheduleObject;

public class CallbackScheduler implements ICallbackScheduler {

    public CallbackScheduler() {};

    public void exportSucceeded(QbScheduleObject obj, String path) {

        System.out.println(obj.getName() + "- EXPORT SUCCEEDED");
        System.out.println("PATH = " + path);

    }

    public void exportFailed(QbScheduleObject obj, String path,
        Throwable e) {

        System.out.println(obj.getName() + "- EXPORT FAILED");
        System.out.println("PATH = " + path);
        System.out.println("ERROR = " + e.toString());

    }

}

```

To use the above code, add the following to the `ServerCommands=` line of the `QB.properties` INI file:

```
-SchedulerCallbackClass:CallbackScheduler
```

Make sure that `CallBackScheduler` is in the CLASSPATH. The code will then print messages at the end of each job saying whether the job was run successfully or not.

### 8.7.5.3. Scheduler Listener

ERES allows scheduled reports to be modified via custom code using server extensions. These custom classes will intercept reports before they are exported and allow users to implement additional business logic to the scheduling process.

To use the Scheduler Listener, you will have to write your own custom class that implements `EresSchedulerListener`. Given below is an example:

```
import quadbase.ChartAPI.QbChart;
import quadbase.reportdesigner.ReportAPI.QbReport;
import quadbase.reportorganizer.organizerAPI.*;
import quadbase.reportorganizer.ext.*;

public class MyEresSchedulerListener implements EresSchedulerListener
{
    public QbReport modifyBeforeExport(QbReport report,
    QbScheduleObject so, String exportPath, String username) {
        System.out.println("modifyBeforeExport(" + report + "," +
so + "," + exportPath + "," + username + ")");
        return report;
    }

    public QbChart modifyBeforeExport(QbChart chart,
    QbScheduleObject so, String exportPath, String username) {
        System.out.println("modifyBeforeExport(" + chart + "," +
so + "," + exportPath + "," + username + ")");
        return chart;
    }
}
```

The above example prints a simple `System.out.println` statement before exporting either the report or chart.

To use any custom class implementing `SchedulerListener`, you will have to implement another custom class implementing `DefaultListenerManager`. For example:

```
import quadbase.reportorganizer.ext.*;

public class MyEresListenerManager extends DefaultListenerManager {
    public MyEresListenerManager() {}

    public EresSchedulerListener getSchedulerListener() {
        return new MyEresSchedulerListener();
    }
}
```

```

    public MenuPageListener getMenuPageListener() {

        return new MyMenuPageListener();

    }

}

```

To use the above code, add the following to the `ServerCommands=` line of the `QB.properties` INI file:

```
-ListenerManagerClass:MyEresListenerManager
```

Make sure that both `MyEresListenerManager` and `MyEresSchedulerListener` are in the `CLASS-PATH`.

More information about the `MenuPageListener` can be found in Section 7.5 - Menu Page Listener.

### 8.7.5.4. Report Bursting

You can also use the Report Bursting feature in Scheduler by using the API. For more details on report bursting, please refer to Section 2.2.1.3.1 - Report Bursting. Note that the report must use a database as the data source and must be grouped in order for the report to be bursted.

Given below is an example on how to use report bursting:

```

QbSchedulePackage sPack = new QbSchedulePackage("PackageBursting_All",
    organizer);
sPack.setReportExportType(IExportConstants.PDF);
sPack.setChartExportType(IExportConstants.PNG);
sPack.setTaskOption(QbSchedulePackage.ONE_TIME);
Calendar calendar = Calendar.getInstance();
calendar.add(Calendar.MINUTE, 10);
sPack.setStartDate(calendar.getTimeInMillis());
sPack.setSendEmail(true);
sPack.setFromAddress("user1@quadbase.com");
sPack.setSubject("Test Package Bursting");
sPack.setBodyText("You should find a report of one group");
sPack.setEmailType(QbSchedulePackage.ASATTACHMENT);

QbScheduleObject rptObj = new QbScheduleObject("new project/
InventoryInformation",
    QbScheduleObject.TEMPLATE_TYPE_REPORT, sPack);
rptObj.setFileLocation("help/examples/ReportGallery/templates/
InventoryInformation.rpt");
String exportLocation1 = rptObj.pickDefaultExportLocation();
rptObj.setBurstReport(QbScheduleObject.ALLBURSTING);
rptObj.setEmailColumnIndex(16);

QbScheduleObject chtObj = new QbScheduleObject("new project/col2d",
    QbScheduleObject.TEMPLATE_TYPE_CHART, sPack);
chtObj.setFileLocation("help/examples/ChartGallery/data/col2d.tpl");
String exportLocation2 = chtObj.pickDefaultExportLocation();

organizer.getScheduler().addSchedulePackageTask(sPack);

```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/PackageBurstingAllERES.zip> ]

The above code bursts all groups and sends out an email using a report column as the source for the addresses. As you can see, bursting can be set individually for each scheduled report. The above example contains a report that is bursted and a chart that is not bursted (because bursting is not supported for charts).

Note that in the above code, column 16 does not exist in the template. The method is provided in the example to show how to pass in the email column for bursting.

### 8.7.5.5. Javadoc

Javadoc for the entire API is provided along with ERES. It is located here (Reports and charts) [ <http://data.quadbase.com/Docs70/eres/help/apidocs/index.html> ] and here (ES functions) [ <http://data.quadbase.com/Docs70/eres/help/ERESapidocs/index.html> ].

### 8.7.5.6. Summary

ERES API provides an easy-to-use and powerful API to query the scheduler interface as well as add and remove schedules. You can also write code to perform your own action when a job has been completed successfully (or not) in Scheduler.

Please note that the API requires a JDK 1.8 or above. The ERES API has been tested on Windows, Solaris, Linux, AIX, and HP platforms.

## 8.8. SOAP Interface

ERES provides SOAP service for communication with other applications. You can run and export a report or a chart by sending an XML SOAP request message to the ERES SOAP service. In this chapter, you will find instructions for writing your own application that will be able to communicate with ERES using SOAP.

The SOAP service that runs a report/chart is called **LookupService**. The functionality of LookupService is similar to the LookupServlet, which is used in the Report/Image URLs. The main difference is that the request and response are both in XML format and are being transported in an envelope over HTTP/SMTP.

LookupService takes a SOAP envelope, processes the request, generates report/chart and exports it to the requested format. It returns the result in a SOAP envelope back to the client. It uses MIME attachments for sending data files and receiving the generated reports/charts.

The Lookup Service is running at this URL by default:

```
http://machine:port/ERES_CONTEXT/services/LookupService
```

(Please, replace machine, port, and ERES\_CONTEXT by actual values used in your system. If you used the default setup with Tomcat, the port number is 8080 and ERES\_CONTEXT is ERES.)

The name of the SOAP method to use is **lookup**.

### 8.8.1. LookupService Running

Before you can use the LookupService, the member host and member port number need to be set up. They can be added in the Admin Console → Setting Info → Clustering & Load Balancer Setting. The member host is usually the machine ERES is running on and the member port number is usually the port ERES is running on.

LookupService can be started automatically when ERES Server is started. To do this, you need to add the `-deploySOAPService` parameter to the `ServerCommands=` line of the `QB.properties` (in the `ERES/WEB-INF/classes` directory). LookupService will be deployed the next time you start the ERES Server.

#### 8.8.1.1. Request Message

The request message is a standard XML SOAP message. It looks like this:

```
<s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/">
  <s:Header>
    <!--Here belongs the Header elements (see below)--> </s:Header>
  <s:Body>
    <lookup xmlns:er="LookupService">
```

```

        <!--Here belongs the Body elements (see below)--> </
lookup> </s:Body>

</s:Envelope>

```

After you create the SOAP message, you have to send it to the LookupService and wait for the response.

### 8.8.1.1.1. Header Elements

Possible header elements are listed in this table.

Element name	Required	Content description
UserName	Yes	Username used to log in to ERES Server
Password	Yes	Password used to log in to ERES Server
URLType	Yes	<b>FORREPORT</b> - generate report or <b>FORCHART</b> - generate chart
ReturnOption	Yes	Return Option: <b>CONTENT</b> - Return the exported file in a byte array. Dynamic export is used. Only the root report is returned, drill-down is generated using DrilldownReportServlet. For DHTML export, chart is generated using RPTImageGeneratorServlet. <b>LINK</b> - Store the exported file in ERES repository and return a link in the envelope. The user can use the link to retrieve the exported file later. <b>EMAIL</b> - Notify the user with an email after the report is exported. This asynchronous approach is especially useful when generating large reports that takes a long time. Can return the result as link, attachment, or HTML.
TemplateType	Only if the template is passed in attachment	Type of the template stored in attachment: <b>PAK, RPT</b> or <b>XML</b> for reports; <b>CHT, TPL</b> or <b>XML</b> for charts
XMLDataType	Only if the XML data are passed in attachment	Type of the XML data stored in attachment: <b>QUAD-BASE</b> - XML data in Quadbase format, <b>DTD</b> - XML data with DTD schema, <b>XSD</b> - XML data with XML Schema

### 8.8.1.1.2. Body Elements

Possible body elements are listed in this table.

Element name	Required	Content description
Refresh	No	Refresh report/chart data before export: <b>true</b> or <b>false</b>
TemplatePath	Only if the template is not passed in attachment	Path to the report/chart template file (on server side). The path can be either absolute or relative to the ERES installation directory or it can be an URL.
QueryParamName	No	Name of a query parameter. See Section 8.8.1.1.5 - Passing Query Parameters for details about passing parameters.
QueryParamSize	No	Number of values for multi-value parameters. On for single-value parameters. See Section 8.8.1.1.5 - Passing Query Parameters for details about passing parameters.
QueryParamValue	No	Single parameter value. For multi-value parameters, use this element several times. See Section 8.8.1.1.5 -

Element name	Required	Content description
		Passing Query Parameters for details about passing parameters.
EmailFromAddress	Only if the EMAIL Return Option is used.	From address used in the email.
EmailToAddress	Only if the EMAIL Return Option is used.	To address used in the email. This element can be used several times to specify more recipients.
EmailSubject	Only if the EMAIL Return Option is used.	Subject of the email.
EmailBodyText	Only if the EMAIL Return Option is used and Email type is 0 or 1.	Body of the email.
EmailType	Only if the EMAIL Return Option is used.	Type of the email: <b>0</b> - report/chart is sent as email attachment <b>1</b> - link to report/chart is append to the end of the email body <b>2</b> - the report is sent as the HTML email body. Only available for reports and (D)HTML export formats
MultiPageExport	No	Is multiPage export used? <b>true</b> or <b>false</b> . Default is false. This option is useful only for reports.
NewDHTML	No	Is new DHTML rendering used? <b>true</b> or <b>false</b> . Default is true. This option is useful for DHTML export format only. The new DHTML is displayed correctly in FireFox and IE 6.0 and above. If you want to use IE 5.5 or lower, set it to false.
OptimizeMemory	No	Is optimized memory used? <b>true</b> or <b>false</b> . Default is false. This option is useful only for reports.
ExportFormat	Yes	Export format: For reports: <b>DHTML, PDF, RTE, TXT, CSV, XLS, XLSX, or RPT</b> For charts: <b>GIF, JPEG, JPG, PNG, PDF, SVG, BMP, or CHT</b>
IsGIFBGTransparent	No	Is GIF background transparent? <b>true</b> or <b>false</b> . Default is false. This option is useful for GIF display type only.

### 8.8.1.1.3. Request Message Attachments

All the necessary files are sent with SOAP message as MIME attachments. The attachments are sent as byte arrays. The following attachments can be sent:

**template file -**

File with report/chart template. Only single file is allowed. If you want to send report with subreport, chart, image or drilldown, use PAK template. If this attachment is added, the `TemplateType` must be specified. This attachment must be the first attachment of the SOAP message.

<b>XML data file -</b>	File used as XML datasource. This can be used only for the reports with XML data-source. If this attachment is added, the <code>XMLDataType</code> must be specified. This attachment must be the second attachment after the template file. This attachment must be the first if the template file attachment is not used. If the <code>XMLDataType</code> is DTD or XSD, there must be another attachment added just after this one. This attachment must contain the DTD or XML Schema of the XML.
<b>DTD or XML Schema -</b>	File with DTD or XML Schema of the XML data file. This attachment must be added, if the <code>XMLDataType</code> is DTD or XSD. This attachment is always the last attachment of the SOAP message and can be used only together with the XML data file attachment.

#### 8.8.1.1.4. Example of Request Message

Here you can see an example of the request message. This message will generate a report from the template located at `c:/ERES/ReportFiles/report.rpt`. It then exports it to DHTML to the default ERES repository and returns a link, which allows accessing the exported report.

```
<s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/">
  <s:Header>
    <UserName>admin</UserName>
    <Password>admin</Password>
    <URLType>FORREPORT</URLType>
    <ReturnOption>LINK</ReturnOption> </s:Header>
  <s:Body>
    <lookup xmlns:er="LookupService">
      <TemplatePath>c:/ERES/ReportFiles/report.rpt</
TemplatePath>
      <ExportFormat>DHTML</ExportFormat> </lookup> </s:Body>
</s:Envelope>
```

#### 8.8.1.1.5. Passing Query Parameters

If your report/chart is parameterized you may want to enter parameter values. This can be done in a similar way as in the Report/Image URL. The `QueryParamName`, `QueryParamSize` and `QueryParamValue` elements are used to specify parameters. All three elements must be used for each parameter in this exact order: `QueryParamName` followed by `QueryParamSize` followed by one or more `QueryParamValue`. This can be followed by other parameter specification. If some parameter value is specified, the default parameter value is used.

The example below shows how to set parameters for a report with a single value parameter (`Category`) and a multi-value parameter (`Products`).

```
<QueryParamName>Category</QueryParamName>
<QueryParamSize>1</QueryParamSize>
<QueryParamValue>Arm Chairs</QueryParamValue>

<QueryParamName>Products</QueryParamName>
<QueryParamSize>3</QueryParamSize>
<QueryParamValue>Cula Chair</QueryParamValue>
<QueryParamValue>Marduk Chair</QueryParamValue>
<QueryParamValue>Nisaba Chair</QueryParamValue>
```

#### 8.8.1.2. Response Message

The message, you will get back from the `LookupService` is again a standard XML SOAP message. The response message has no header and looks like this:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/
envelope/" xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://
www.w3.org/2001/XMLSchema-instance">

    <soapenv:Body>
        <LookupResponse xmlns="">
            <!--Here is a message from the LookupService (see below)--
            > </LookupResponse> </soapenv:Body>

</soapenv:Envelope>
```

### 8.8.1.2.1. Messages from the LookupService

You can get one of the following responses from the LookupService:

- Info:** these messages begin with `Info:`. They contain various information for the user. For example, you will get this message if you use the `EMAIL ReturnOption`. In this case, the message says that the report is processed and will be emailed to the specified user.
- Error:** these messages begin with `Error:`. You will get this kind of message if there were some errors. You should always check the response message if it starts with `Error`. If error occurs, the attachments may be empty.
- Content:** this message is always `CONTENT`. It says that an exported report/chart can be found in attachment (see the next chapter).
- Link:** if the message is not anything from above, it is link to the generated report (when the `LINK ReturnOption` is used).
- SOAP fault -** if the response message does not contain the `LookupResponse` element at all, it is because of a SOAP fault. In this case, you will find the `Fault` element instead of the `LookupResponse` in the message. This element contains description of the SOAP fault.

### 8.8.1.2.2. Response Message Attachments

The response message can contain only one attachment- the exported report or chart. This is in case you have used the `CONTENT ReturnOption` and you have gotten the `CONTENT` response message. Then you can find the attachment, which is containing the file as byte array.

Only single file is supported as attachment. Therefore, it is not possible to export to multi-page (D)HTML using the `CONTENT ReturnOption`.

### 8.8.1.3. SimpleSOAPMessenger

You can find an example SOAP client application in `help/example/soap/SimpleSOAPMessenger.java`. The example is using Apache AXIS as its SOAP engine.

`SimpleSOAPMessenger` is a convenient class for user to easily send SOAP request and to receive response from ERES LookupService. The most important method in this class is `SendAndRecieve()`. It first creates a SOAP Request Message (as described in Section 8.8.1.1 - Request Message). It then sends the message to the LookupService on the server (at the specified URL). Finally, it will receive a response message, process it and return a byte array to the user. The byte array can be a string containing error or link or actual content of the exported report/chart depending on the `ReturnOption`.

User can set the request message parameters using the following methods:

```
public void setURL(String s)- set the LookupService URL
public void setUsername(String s)
public void setPassword(String s)
public void setURLType(String s)
```

```

public void setTemplatePath(String s)
public void setTemplateType(String s)
public void setTemplateByteArray(byte[] b)
public void setReturnOption(String s)
public void setXmlDataType(String s)
public void setXmlData(byte[] data)
public void setDTDDData(byte[] data)
public void setRefresh(boolean b)
public void setMultiPageExport(boolean b)
public void setNewDHTML(boolean b)
public void setOptimizeMemory(boolean b)
public void setExportFormat(String s)
public void addQueryParamNameAndValue(String name, Object[]value)
public void setDisplayType(String s)
public void setGIFBGTransparent(boolean b)
public void setChartPath(String s)
public void setEmailFromAddress(String s)
public void setEmailToAddresses(String[] s)
public void setEmailSubject(String s)
public void setEmailBodyText(String s)
public void setEmailType(int type)

```

### 8.8.1.3.1. SOAP Client Example

Here is an example of using the SimpleSOAPMessenger class.

```

public class TestLookupService{

    public static void main(java.lang.String[] args) {
        SimpleSOAPMessenger ssm = new SimpleSOAPMessenger();
        ssm.setURL("http://localhost:8080/ERES/services/LookupService");
        ssm.setUsername("admin");
        ssm.setPassword("admin");
        ssm.setURLType("FORREPORT");
        ssm.setTemplatePath("help/quickstart/templates/
QuickStart51.rpt");
        ssm.setExportFormat("DHTML");
        ssm.setReturnOption("LINK");
        try {
            byte[] byteArray = ssm.SendAndRecieve();
            String link = new String(byteArray);
            System.out.println("Generated report is at : "+link);
        }catch (Exception ex) {
            ex.printStackTrace();
        }
    }
}

```

## 8.9. Deployment

### 8.9.1. Introduction

ERES consists of several components: Report Designer, Report Viewer, Page Viewer, Chart Viewer, Scheduler, EspressoManager, and Report API. Report Designer is used to create reports in a GUI environment. Report Viewer is an applet used to view already created reports (saved in .rpt / .xml / .pak format). Page Viewer is an applet

that is used to view a report page by page (saved in `.page` format). Chart Viewer is an applet that is used to view a chart (saved in `.cht/.tpl` format). Report API is used to create reports programmatically. Scheduler is used to schedule exports/archives for reports/charts. Finally, the ERES Server serves as a user administrator and handles data and data buffering.



### Note

`qblicense.jar` **must** be included in the CLASSPATH or in the HTML page as an archive (along with any other additional jars) in order to run any code.

While the Report Designer and Scheduler **must** be used in conjunction with ERES Server, an option is provided for Report Viewer, Page Viewer, Chart Viewer, and Report API to work without connecting to the ERES Server.

Further, you must provide information about how you would like to connect to the ERES Server. You can specify this by using the following three API methods that are available in `QbReport`, `QbChart`, `QbReportDesigner`, and `QbChartDesigner`:

```
static void useServlet(boolean b);
static void setServletRunner(String comm_url);
static void setServletContext(String context);
```

For Page Viewer and Report Viewer, you can set the connection information to the ERES Server by passing in the following parameters:

```
comm_protocol (servlet), comm_url (machine name/IP address and port number),
servlet_context (servlet context)
```

Described, in sections below, are the various deployment scenarios that you may encounter and the consequences of each scenario.

## 8.9.2. Deploying with ERES Server

This section deals with deploying report components that work in conjunction with ERES Server. To learn more about the interaction, please refer to the Interaction with ERES Server section under the ERES Report API Overview chapter.

Note that any references to a data source, or a report file, using the relative URL reference (i.e. `help\examples\data\sample.dat`) are relative to the directory from where ERES is running in.

### 8.9.2.1. Report Designer

As mentioned before, Report Designer must be used in conjunction with ERES Server. `ReportDesigner` can be called using the API. Note that in this scenario, you will have to add `ReportDesigner.jar` in your CLASSPATH and place a copy of the `reportimages` (`ERES/reportimages`), `images` (`ERES/images`), and the `backgroundImages` (`ERES/backgroundImages`) directories under the working directory of the `.class` file.

Instead of copying the directories relative to the working directory when invoking the Report Designer from API, an option to set the location `images/`, `reportimages/` and `backgroundImages/` directory is also available. Simply use a `QbReportDesigner` constructor with the parameter `imagesPath`. For more detail about the `QbReportDesigner` constructor or the `imagesPath` parameter, please refer to the ERES Java API Documentation.

To connect to ERES Server, you use the following three API methods in `QbReportDesigner` to set the connection information:

```
static void useServlet(boolean b);
```

```
static void setServletRunner(String comm_url);  
static void setServletContext(String context);
```

Please note that specifying the connection information to ERES Server must be made before calling any `QbReportDesigner` constructors.

### 8.9.2.2. Report Viewer

Report Viewer can be used in an applet or application environment to show a report saved in a `.pak` file (either generated by Report Designer or Report API).

To deploy Report Viewer in an applet environment, the `ReportViewerWithChart.jar` must be included in the HTML file as the archive file.

To use Report Viewer, construct an HTML page with the proper applet code (for more details, please refer to the Report Viewer chapter). Note that any relative references to the report location and/or data are relative to the directory from where the ERES Server has been started.

To connect to the ERES Server, you pass the following parameters to the Report Viewer applet:

```
comm_protocol (servlet), comm_url (machine name/IP address and port number),  
servlet_context (servlet context)
```

### 8.9.2.3. Page Viewer

Page Viewer can be used in an applet or application environment to show a particular page of a report at a time. These PAGE files are of `.page` file extension and are generated by Report Designer or by the Report API.

To deploy Page Viewer in an applet environment, the `PageViewer.jar` must be included in the HTML file as the archive.

To use Page Viewer, construct an HTML page with the proper applet code (for more details, please refer to the Page Viewer chapter). Note that any relative references to the report location and/or data are relative to the directory from where ERES Server has been started.

To connect to ERES Server, you pass the following parameters to the Page Viewer applet:

```
comm_protocol (servlet), comm_url (machine name/IP address and port number),  
servlet_context (servlet context)
```

### 8.9.2.4. Chart Viewer

Deploying Chart Viewer is similar to deploying Report Viewer. Chart Viewer can be used in an JNLP (More about Applets in JNLP: Section 1.3.4 - Run Applets As JNLP) or application environment to show a chart saved in a `.cht` or `.tpl` file (either generated by Designer or API).

To deploy Chart Viewer in an JNLP (More about Applets in JNLP: Section 1.3.4 - Run Applets As JNLP) environment, `EspressoViewer.jar` must be included in the HTML file as the archive.

To use Chart Viewer, construct an HTML page redirects to a JSP file with the proper JNLP code. (for more details, please refer to start the Chart Viewer chapter) Note that any relative references to the chart location and/or data are relative to the directory from where ERES Server has been started.

To connect to ERES Server, you pass the following parameters to the Chart Viewer JNLP (More about Applets in JNLP: Section 1.3.4 - Run Applets As JNLP):

```
comm_protocol (servlet), comm_url (machine name/IP address and port number),  
servlet_context (servlet context)
```

### 8.9.2.5. Report API

Report API can be used in either a JNLP (More about Applets in JNLP: Section 1.3.4 - Run Applets As JNLP) environment or as an application. Report API can also be used in a servlet or jsp environment to generate server-side reports.

To deploy Report API, `ERESOrganizer.jar`, `ERESServer.jar` and `ExportLib.jar` must be in the CLASSPATH (for application and servlet/jsp environment) or included in the JSP file, for a JNLP (More about Applets in JNLP: Section 1.3.4 - Run Applets As JNLP)

To connect to ERES Server, you would use the following methods in `QbReport` to set the connection information:

```
static void useServlet(boolean b);
static void setServletRunner(String comm_url);
static void setServletContext(String context);
```

Please note that specifying the connection information to ERES Server must be made before calling any `QbReport` constructors.

### 8.9.2.6. Scheduler

A scheduler has also been included with ERES. This scheduler can be used to export and archive reports and charts from their source files at specific intervals. For more information on the Scheduler, please look in the Scheduler chapter.

Like Report Designer, Scheduler too must be used in conjunction with the ERES Server. Scheduler can be called using the API. Note that in this scenario, you will have to add `ERESServer.jar` and `ERESOrganizer.jar` in your CLASSPATH.

To connect to the ERES Server, you would use the following methods in `QbOrganizer` to set the connection information:

```
static void setServletRunner(String comm_url);
static void setServletContext(String context);
```

Please note that specifying the connection information to the ERES Server must be made before calling any `QbOrganizer` constructors.

## 8.9.3. Deploying without ERES Server

This section deals with deploying components that work independent of ERES. To learn more about the interaction, please refer to the Interaction with ERES Server section under the ERES API Overview chapter.

Generating reports/charts independently of ERES Server results in a slightly better performance as the interaction between the ERES Server and the report/chart component is removed.

An important consideration is that any references to a data source or a report source, that is relative (for example, `help\examples\data\sample.dat`) are relative to the working directory from where the HTML file (in the case of an applet) or the class file (in an application) is being executed. In a servlet/jsp environment, the working directory typically differs from the directory where the class resides. To find out the working directory, have the following line of code in your servlet/jsp:

```
System.out.println("The working directory is " +
    System.getProperty("user.dir") + ". ");
```

By having the above line of code, and executing the servlet/jsp, the working directory is ascertained and the data files and report templates can be moved to locations relative to the working directory, as dictated by the code.

### 8.9.3.1. Report Viewer

To deploy Report Viewer, `ReportViewerWithChart.jar` must be included in the HTML file as the archive. Depending on what functionality you are using, you have to include additional jars as necessary.

Report Viewer can be used independent of the ERES Server with the addition of one more parameter in the applet code:

```
<param name="EspressManagerUsed" value="false">
```

Note that any relative references to the report location and/or data are relative to the directory where the HTML file is located. For example, if the data source specified in the `.pak` file is `help\examples\data\sample.dat` and if the HTML file is located in `D:\ERES\TestApplet`, then for the report to be displayed successfully, `help\examples\data\sample.dat` must exist within `D:\ERES\TestApplet` (i.e., `D:\ERES\TestApplet\help\examples\data\sample.dat` must exist).

### 8.9.3.2. Page Viewer

To deploy Page Viewer, `PageViewer.jar` must be included in the HTML file as the archive. Depending on what functionality you are using, you have to include additional jars as necessary.

Page Viewer can be used independently from ERES Server with the addition of one more parameter in the applet code:

```
<param name="EspressManagerUsed" value="false">
```

Note that any relative references to the report location and/or data are relative to the directory where the HTML file is located. For example, if the data source specified in the `.pak` file is `help\examples\data\sample.dat` and if the HTML file is located in `D:\ERES\TestApplet`, then for the report to be displayed successfully, `help\examples\data\sample.dat` must exist within `D:\ERES\TestApplet` (i.e., `D:\ERES\TestApplet\help\examples\data\sample.dat` must exist).

Also note that when using Page Viewer without connecting to ERES Server, you cannot pass in a template directly. You must pass in the page file instead.

### 8.9.3.3. Chart Viewer

To deploy Chart Viewer, `EspressViewer.jar` must be included in the HTML file as the archive. You have to include additional jars as necessary depending on what functionality you are using.

Chart Viewer can be used independently from ERES Server with the addition of one more parameter in the applet code:

```
<param name="EspressManagerUsed" value="false">
```

Note that any relative references to the report location and/or data are relative to the directory where the HTML file is located. For example, if the data source specified in the `.pak` file is `help\examples\data\sample.dat` and if the HTML file is located in `D:\ERES\TestApplet`, then for the report to be displayed successfully, `help\examples\data\sample.dat` must exist within `D:\ERES\TestApplet` (i.e., `D:\ERES\TestApplet\help\examples\data\sample.dat` must exist).

### 8.9.3.4. Report API

To deploy Report API, `ERESorganizer`, `ERESServer.jar` and `ExportLib.jar` must be in the CLASS-PATH (for application and servlet/jsp environment) or included in the HTML file (for an applet).

Report API can also be used without connecting to ERES Server by adding the line of code below:

---

```
QbReport.setEspressManagerUsed(false);
```

This method **must** be called prior to any `QbReport` instantiation.

As with Report Viewer, any relative references to the report location and/or data are relative to the working directory from where class file is being executed. For example, if the data source specified is `help\examples\data\sample.dat` and if the class file is located in `D:\ERES\TestApplication`, then for the report to be displayed successfully, `help\examples\data\sample.dat` must exist within `D:\ERES\TestApplication` (i.e., `D:\ERES\TestApplication\help\examples\data\sample.dat` must exist).

Note that any charts created in Report Designer will contain a **relative** path to the template. Therefore, the directory structure must be mirrored under the working directory (i.e. create a `chart` sub-directory with the template under the working directory) or open the template and change the path with an absolute path using the following code:

```
// Get the Chart from the footer
ReportChartObject obj =
    (ReportChartObject)report.getTable().getFooter().getData(0);
String originalLocation = obj.getText();
String newLocation = new String("http://128.0.0.1/EspressEnterprise/ERES/
test/chart/chart/ERES1.1_0.tpl");
obj.setText(newLocation);
```

In a servlet/jsp environment, the location of the directory with the class file and the working directory might differ. In that case, any relative references are with respect to the working directory. For example, if the servlet class file was in `D:\<some servlet engine>\webapp\examples\quadbase\someservlet.class` and the data source was `help\examples\data\sample.dat` and the working directory was `D:\<some servlet engine>\webapp`, then `help\examples\data\sample.dat` must exist under `D:\<some servlet engine>\webapp` (i.e., `D:\<some servlet engine>\webapp\help\examples\data\sample.dat` must exist) for the report to be generated successfully.

Note that in the API, methods are also available that allow you to specify the location of the chart, drill-down, image, and sub-report directories where certain files may be located. For instance, given the existence of a template, `ERES1.1_0.tpl`, in the report, you can specify the location this template by using the following code:

```
report.setChartPath("d:/ERES/Chart/");           // where report is an object
of type QbReport
```

The above code sets the location of the template file thus avoiding the need to have a chart subdirectory under the working directory of your class file.

The following methods work similarly:

```
QbReport.setDrillDownPath(String directory);
QbReport.setImagePath(String directory);
QbReport.setSubReportPath(String directory);
```

## 8.9.4. Deploying in a non-Windows environment

ERES is a Pure Java tool for generating reports. As such, it does not contain graphical libraries for generating colors and fonts and other AWT information. For that, Java relies on the system's (on which the reports are being generated) libraries for providing that information. Thus, an environment capable of providing AWT information and a graphics card (for exporting to static formats) are required.

In a Windows environment, nothing extra needs to be done to set up such an environment as it already exists. A GUI interface is already running and a graphics card already exists.

---

This is usually not the case for non-Windows environments (such as Unix and Linux). You need to have X or some form of X running on such systems and point the display to the machine running X (such as running the command `export DISPLAY=192.168.0.16:0.0` in a shell). For the best performance, Quadbase recommends running X on the machine (or setting the `DISPLAY` to point to another machine running X). However, if that is not an acceptable solution, there are alternative solutions available.

### 8.9.4.1. Xvfb (X Virtual Frame Buffer)

Xvfb is an X server that can run on machines with no display hardware and no physical input devices. It emulates a dumb terminal framebuffer using virtual memory.

The primary use of this server is intended to be server testing. Xvfb is also limited in the number of colors and fonts it can handle.

Xvfb can be downloaded (there are different version available depending on the system) and then run. After Xvfb is running, the `DISPLAY` variable needs to be set and then passed to the application (or the servlet engine).

### 8.9.4.2. JVM 1.5+ in Headless Mode

You can use a 1.5+ JVM run-time parameter to run applications using ERES. The run-time parameter is `java.awt.headless` and setting it to `true` results in Java not making an X connection for any graphics information. For example, if you have an application called `chartExport` that generates charts as jpg's, ordinarily running it would involve making a connection to X. However, using the following command:

```
java -Djava.awt.headless=true exportChart
```

the same application is now run without a connection to X.

## 8.9.5. Platform Specific Issues

### 8.9.5.1. AS/400

#### 8.9.5.1.1. Running Under NAWT

You must do the following to run any application using ERES in an AS/400 environment using NAWT:

- Set the `DISPLAY` environment variable to the system name and display number. The display number is the display number of the VNC server.
- Set the `XAUTHORITY` variable to `/home/VNCProfile/.Xauthority`, where `VNCProfile` is the profile that started the VNC server. Please note that `XAUTHORITY` does not need to be set if both the VNC server and the Java virtual machine is running under the same user profile.
- Set the Java system property before running java

```
os400.awt.native=true
```

#### 8.9.5.1.2. Running Under Headless Mode

you must do the following to run any application using ERES in an AS/400 environment using headless mode:

- Set the Java system property before running java

```
java.awt.headless=true
```

- Make sure that your code include the following before calling any `QbChart` constructor:

```
QbChart.setForExportOnly(true);
```

## 8.9.5.2. Linux/Unix

### 8.9.5.2.1. Running under X

You must do the following to run any application using ERES in a Linux/Unix environment using X:

- Set the `DISPLAY` environment variable to a X client system name and display number.

### 8.9.5.2.2. Running Under Headless Mode

You must do the following to run any application using ERES in a Linux/Unix environment using headless mode:

- Set the Java system property before running java

```
java.awt.headless=true
```

- Make sure that your code include the following before calling any `QbChart` constructor:

```
QbChart.setForExportOnly(true);
```

## 8.A. Getting the Report Data

The two appendices in this Chapter (Appendix 8.A - Getting the Report Data and Appendix 8.B - Creating the Report) contain information to help you build a report from scratch (without using the Designer). Keep in mind that this method is typically not recommended as it will make the report more difficult to deploy and maintain. However, under certain circumstances, it may be necessary to construct reports in this manner.

The following is an example of a `QbReport` constructor. Although there are numerous variations available, the typical `QbReport` constructor requires at least three parameters. To create a new report, you must specify the report type, the input data source information, and a mapping of the data columns to the respective columns of the report.

```
QbReport(java.lang.Object parent, int reportType, IResultSet data, ColInfo[]  
mapping, java.lang.String template)
```

The report template is not required (can be null) and the parent object is only required if the report is used in an applet, but the other three parameters must always contain meaningful information. This appendix takes an in depth look at the data parameter and the methods for connecting to different data sources. Appendix 8.B - Creating the Report discusses the report type, the column mapping, and the actual creation of the report. Appendix 8.B - Creating the Report also contains working examples for you to try.

The first step to creating any report is to obtain the data. Data may be retrieved from one of several different types of sources. These sources include:

- Fetch the data from a local or remote database using a JDBC driver. All you need to do is to provide the information of how to connect to the database and the exact form of the query. The report would then fetch the result automatically.
- Read the data from a plain text file, which contains database records in plain text (ASCII) format or XML format. Files of this format can be created by most database programs.
- Pass your own dynamic data, either as an EJB or a class file, through API.
- Pass the dataset as an array in memory.

In the following sections, we will take a close look at each of these data sources and the methods used to obtain data from them.

## 8.A.1. Data from a Database

One of the powerful features of Report API is its ability to fetch data from a database directly through JDBC. With this approach, all you need to do is to specify the information necessary to connect to the database and the precise form of the SQL statement. Thus, your program can connect to virtually any database provided that a JDBC driver is available.

The database and query information must be stored in a `DBInfo` object prior to constructing the report. Use the following code to instantiate the `DBInfo` object.

```
DBInfo dbinfo = new DBInfo(
    "jdbc:odbc:woodview",           // URL
    "sun.jdbc.odbc.JdbcOdbcDriver", // JDBC driver
    "myName",                       // username
    "myPassword",                   // password
    "select * from sales");         // SQL Query
```

Since `DBInfo` implements the interface `IDatabaseInfo`, you can use the `DBInfo` object in the following `QbReport` constructor:

```
QbReport(java.lang.Object parent, int reportType, IDatabaseInfo dbinfo,
    ColInfo[] mapping, java.lang.String template)
```

In some cases, you may not wish to pass the entire database information (such as userid, password, location, driver and the query) to ERES. You may want to make the connection yourself and just provide the resultset of the query directly to the API. This can be done by creating a `QueryResultSet` object from the `ResultSet` object you have generated and passing that `QueryResultSet` object as the data source, instead of the `DBInfo` object.

```
// Create a QueryResult object from the ResultSet object resultSet
QueryResultSet queryResultSet = new QueryResultSet(resultSet);

// QueryResultSet implements IResultSet, use it in the following QbReport
// constructor
QbReport(java.lang.Object parent, int reportType, IResultSet data, ColInfo[]
    mapping, java.lang.String template)
```

In the above example, an instance of class `QueryResultSet` is created from the `ResultSet` object passed to the API and this instance is passed to the `QbReport` constructor to create the report object. Please note that in this scenario, the report is **not** refreshable. To update the report, you will need to make the connection to the database again and pass the `ResultSet` object to the report and refresh it yourself.

For a working example using data from a database, see Appendix 8.B.4.2 - Creating the Report.

### 8.A.1.1. Data from JNDI

ERES also allows data to be obtained from a JNDI source. JNDI data sources are treated like database data sources and support the same functionalities. Using a JNDI data source makes it easier to migrate reports between different environments. If the data sources in both environments are setup with the same lookup name, reports can be migrated without any changes.

You must have a data source deployed in your application server to connect to a JNDI data source. You must also provide the `INITIAL_CONTEXT_FACTORY` and `PROVIDER_URL` to successfully make the connection. Please

note that when connecting to a JNDI data source deployed in Tomcat, the `INITIAL_CONTEXT_FACTORY` and `PROVIDER_URL` need not be provided (although ERES Server must be running under the Tomcat environment).

```
// create a DBInfo object with JNDI lookup name and query
String JNDIName = "java:comp/env/jdbc/TestDB";
String query = "select * from testdata";

// The environment hashtable is empty for tomcat because ERES Server is
// running inside Tomcat context. If other application server is used,
// need to set INITIAL_CONTEXT_FACTORY and PROVIDER_URL.
Hashtable env = new Hashtable();
DBInfo dbInfo = new DBInfo(JNDIName, query, env);
```

In the above example, an instance of class `DBInfo` provides the information that the program needs to connect to a JNDI data source and retrieve data. Use the following constructor containing `IDatabaseInfo` to construct the report:

```
public QbReport(Object parent, int reportType, IDatabaseInfo dbinfo,
    ColInfo[] colMap, String templateFile);
```

## 8.A.2. Data from a Data File (TXT/DAT/XML)

Data for generating a report can also be imported from a data file, which can be a text file as well as an XML formatted file. A report data file (with a `.dat` extension) is a plain text file where each line represents one record, except the first two lines, which contain the data types and field names, respectively. Here is an example of a data file, which contains four records and where each record has three fields. The first line specifies the data type of each field and the second line specifies the field name.

```
string, date, decimal
Name, Day, Volume
"John", "1997-10-3", 32.3
"John", "1997-4-3", 20.2
"Mary", "1997-9-3", 10.2
"Mary", "1997-10-04", 18.6
```

The use of the comma character is optional, but most database programs can output a file in this format (except for the first two lines) when exporting records from a table in text format. As a result, even if you do not have a JDBC driver for your database, you can still quickly produce reports. You can simply export the data from your database in text format and then your program can read it using Report API.

For XML format, the specification is as follows:

```
<EspressData>

    <DataType>string</DataType>
    <DataType>date</DataType>
    <DataType>decimal</DataType>

    <FieldName>Name</FieldName>
    <FieldName>Day</FieldName>
    <FieldName>Volume</FieldName>

    <Row>
        <Data>John</Data>
```

```

        <Data>1997-10-3</Data>
        <Data>32.3</Data> </Row>

    <Row>
        <Data>John</Data>
        <Data>1997-4-3</Data>
        <Data>20.2</Data> </Row>

    <Row>
        <Data>Mary</Data>
        <Data>1997-9-3</Data>
        <Data>10.2</Data> </Row>

    <Row>
        <Data>Mary</Data>
        <Data>1997-10-4</Data>
        <Data>18.6</Data> </Row>

</EspressData>

```

For more details about XML Data, please refer to Section 4.1.5.1.1 - XML Encoding.

Specifying the text file to use is very straight forward. Use the following constructor and replace the variable `dataFile` with the path (relative or full path) of the data file. If you are connecting to the server, relative paths should be in respect to the ERES root directory. Otherwise, the path will be relative to the current working directory.

```

QbReport(java.lang.Object parent, int reportType, java.lang.String dataFile,
    ColInfo[] mapping, java.lang.String template)

```

For a working example using data from a text file, see Appendix 8.B.2.2 - Creating the Report.

### 8.A.3. Data from an XML Data Source

In addition to the above, ERES allows you to retrieve data and query XML files. XML data can be in virtually any format, but you need to specify a DTD file or an XML schema along with the XML data. The following code demonstrates how to set up an XML query:

```

// Set up the XML Data Source
String xmlfilename = "Inventory.xml";
String dtdfilename = "Inventory.dtd";
String xmlcondition = "/Inventory/Category/Product/ProductID < 45";

XMLFieldInfo[] fields = new XMLFieldInfo[5];
fields[0] = new XMLFieldInfo(new String[] {"Inventory", "Category",
    "Product"}, "ProductID");
fields[0].setAttributeDataType(DTDDatatype.INT);

fields[1] = new XMLFieldInfo(new String[] {"Inventory", "Category",
    "Product", "ProductName"});

fields[2] = new XMLFieldInfo(new String[] {"Inventory", "Category",
    "Product", "UnitPrice"});
fields[2].setElementDataType(DTDDatatype.DOUBLE);

fields[3] = new XMLFieldInfo(new String[] {"Inventory", "Category",
    "Product", "UnitsInStock"});

```

```

fields[3].setElementDataType(DTDDatatype.INT);

fields[4] = new XMLFieldInfo(new String[] {"Inventory", "Category",
    "Product", "ShipDate"});
fields[4].setElementDataType(DTDDatatype.DATE);
fields[4].setDateFormat(XMLDataTypeUtil.YYYY_MM_DD);

XMLFileQueryInfo xmlInfo = new XMLFileQueryInfo(xmlfilename, fields,
    xmlcondition, fields, dtdfilename, false, null);

```

The `XMLFieldInfo` instance is created using one of two constructors. The first constructor, used to select xml fields, contains one parameter. For this constructor, you need to pass in a `String` array that specifies each xml tag in the hierarchy leading to the target field. In the above example, `fields[1-4]` are created using the first constructor. The second constructor, used to select xml attributes, contains two parameters. In addition to the `String` array parameter, the second constructor also requires another string for the attribute name. In the above example, `field[0]` is created using this constructor because `ProductID` is an attribute of `Product`.

You may have also noticed that for any non-String field, you must explicitly set the data type. Once you have created the `XMLFileQueryInfo` instance, you can use the following constructor to create the `QbReport`.

```

public QbReport(Object parent, int reportType, XMLFileQueryInfo xmlInfo,
    ColInfo[] colMap, String templateFile, boolean sideBySideLayout);

```

You can also pass in an XML stream instead of an XML file, when using XML data as a data source. To pass in an XML stream, you would pass in the byte array containing the XML data instead of the XML data file name.

In the above example, you can pass in a XML stream via a byte array (for example, a byte array called `xml-ByteArray`) in the `XMLFileQueryInfo` constructor:

```

XMLFileQueryInfo xmlInfo = new XMLFileQueryInfo(xmlByteArray, fields,
    xmlCondition, fields);

```

For a working example using data passed in from an xml file source, see Appendix 8.B.5.2 - Creating the Report.

## 8.A.4. Data passed in an Array in Memory

The API allows input data to be passed directly in memory, as an array. This is made possible by the interface `IResultSet` [ <http://data.quadbase.com/Docs70/eres/help/apidocs/quadbase/reportdesigner/util/IResultSet.html> ] (defined in `quadbase.reportdesigner.util` package [ <http://data.quadbase.com/Docs70/eres/help/apidocs/quadbase/reportdesigner/util/package-summary.html> ]). This interface is used to read data in tabular form and is quite similar to the `java.sql.ResultSet` interface used for JDBC result sets (but is much simpler). Users can provide their own implementation of `IResultSet`, or use one provided by ERES. The simplest implementation is the class `DbData` (Other classes that provide an `IResultSet` implementation are `QueryResultSet` [ <http://data.quadbase.com/Docs70/eres/help/apidocs/quadbase/reportdesigner/util/QueryResultSet.html> ] and `StreamResultSet` [ <http://data.quadbase.com/Docs70/eres/help/apidocs/quadbase/reportdesigner/util/StreamResultSet.html> ]). If you can fit all the data you need for the report in memory, you can simply pass the array as an argument in `DbData` with one line of code. There are three constructors for `DbData`:

```

DbData(java.lang.String s)
    Construct DbData by parsing the data value argument from an HTML
page

```

```

DbData(java.lang.String[] fieldName, java.lang.Object[][] records)
    Construct a new DbData class

```

```
DbData(java.lang.String[] dataType, java.lang.String[] fieldName,
java.lang.String[][] records)
```

Construct a new DbData class

Note that when passing in data, using this approach, for a Summary Break or Cross Tab report, make sure that the data is applicable and is already grouped and sorted to improve performance and generate a report successfully.

We will use the following constructor in the example here.

```
public DbData(String dataType[], String fieldName[], String records[][])
```

Here, the first argument presents the data types (the first line in the data file) and the second argument presents the field names (the second line). The third argument, `records[ ][ ]`, provides an array of records, `records[ i ]` being the *i*th record. The following shows how it works:

```
String dataType[] = {"varchar", "decimal"};
String fieldName[] = {"People", "Sales"};
String records[][] = {{"Peter", "93"}, {"Peter", "124"},
                      {"John", "110"}, {"John", "130"},
                      {"Mary", "103"}, {"Mary", "129"}};
```

```
DbData data = new DbData(dataType, fieldName, records);
```

To create the report, use the following QbReport constructor:

```
public QbReport(Object parent, int reportType, IResultSet data,
                ColInfo[] colMap, String templateFile);
```

You can pass sorted data to ERES to improve performance considerably. If the data is already sorted, you can set a flag to `true` in the constructor to avoid ERES sorting the data again. The constructor is given below:

```
public QbReport(Object parent, int reportType, IResultSet data,
                ColInfo[] colMap, String templateFile, boolean
                sideBySideLayout, boolean isDataSorted);
```

This is important especially when you are generating a summary break report with multiple break levels. If data in memory is already sorted (e.g. sorted by a database engine), you can take advantage of the fact and use this feature to prevent the report engine from unnecessarily building extra internal data structures and performing sorting, which consumes a large amount of memory and processor resources. Hence, performance can be greatly improved.

For a working example using data passed in an array in memory, see Appendix 8.B.1.2 - Creating the Report.

## 8.A.5. Data passed in a Custom Implementation

For maximum flexibility, you can retrieve and prepare the dataset in any way you want and pass it to the report engine. To pass in your class file as the data source, your class file must implement the `IDataSource` [ <http://data.quadbase.com/Docs70/eres/help/apidocs/quadbase/reportdesigner/util/IDataSource.html> ] interface. Given below is a simple example that implements `IDataSource`:

```
public class CustomClassData extends Applet implements IDataSource {
    // Setting DbData for passing data as arguments
```

```

String dataType[] = {"string", "String", "double"};
String fieldName[] = {"Destination", "Time", "Price"};
String records[][] = {"Mayfair", "13:43", "3.50"},
    {"Bond Street", "13:37", "3.75"},
    {"RickmansWorth", "13:12", "5.25"},
    {"Picadilly", "13:24", "3.00"};
DbData data = new DbData(dataType, fieldName, records);

public IResultSet getResultSet()
{
    return data; }
}

```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/CustomClassDataERES.zip> ]

The example above creates data (DbData instance) and stores it in memory. When the `getResultSet()` method is called, it returns the DbData object which implements IResultSet. Keep in mind that it is not necessary to create your data in this manner. As long as you are able to return an object that implements IResultSet, you can obtain the data from any data source. Use the following constructor to create your report:

```

QbReport(java.lang.Object parent, int reportType, int fileType,
    java.lang.String filename, ColInfo[] mapping, java.lang.String template)

```

For custom class files, set the `fileType` to `QbReport.CLASSFILE` and the `filename` to the name of the class-file.

Please note that if you are passing in your own class file as the data source and you are using the ERES Server, the class file must be accessible from the CLASSPATH of the ERES Server.

You can also pass in a parameterized class file as the data source for the report. The parameter is obtained at run-time from the user and the data is then fetched and used to generate the report. Here is an example for a parameterized class file, this is the same file used in Section 8.1.5.8.5 - Open Report Designer with a Specific Class File Data Source.

```

public class ParamClassFile implements IParameterizedDataSource {

    String url = "jdbc:odbc:woodview";
    String driver = "sun.jdbc.odbc.JdbcOdbcDriver";
    String username = "";
    String password = "";

    // Specify what the parameter properties are.
    public IQueryInParam[] getParameters() {
        SimpleQueryInParam[] params = new SimpleQueryInParam[2];
        // SimpleQueryInParam(name, promptString, mapToColumn,
        // defaultValue, actualValue)
        params[0] = new SimpleQueryInParam("price", "Max price:", false,
        null, null,
        Types.DOUBLE, new Double(500.00), null);
        params[1] = new MySimpleQueryInParam("popular", "Popular Items
        Only:", false, null,
        null, Types.VARCHAR, new String("NO"), null);
        return params; }

    private class MySimpleQueryInParam extends SimpleQueryInParam

```

```

        implements IQueryParamValuesProvider {
    public MySimpleQueryInParam(String paramName, String promptName,
        boolean mapToColumn, String tableName, String columnName,
        int sqlType, Object defaultValue, Object value) {
    super(paramName, promptName, mapToColumn, tableName,
columnName,
        sqlType, defaultValue, value); }

    public Vector getSelectionChoices() {
        Vector choices = new Vector();
        choices.add(new String("Yes"));
        choices.add(new String("No"));
        return choices; } }

    // Specify what data is to be returned
    public IResultSet getResultSet(IQueryInParam[] params) {
        double price = 500.00;
        int units = 999;

        if ((params != null) && (params.length >= 1)) {
            Object obj = params[0].getValue();
            if ((obj != null) && (obj instanceof Double))
                price = ((Double)obj).doubleValue();
            obj = params[1].getValue();

            if((obj != null) && (obj instanceof String) &&
                ((String)obj).equalsIgnoreCase("yes"))
                units = 15; }

        try{
            Class.forName(driver);
            Connection conn = DriverManager.getConnection(url,
username, password);
            String query = "SELECT ProductName, Description,
UnitPrice, UnitsInStock
FROM Products WHERE UnitPrice < " + price + " AND
UnitsInStock < " + units;
            Statement stmt = conn.createStatement();

            ResultSet rs = stmt.executeQuery(query);
            QueryResultSet qry = new QueryResultSet(rs);

            return qry; } catch(Exception e) {
                e.printStackTrace(); }

        return null; }
}

```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/ParamClassFile.zip> ]

The parameterized class file must implement `IParameterizedDataSource` [ <http://data.quadbase.com/Docs70/eres/help/apidocs/quadbase/reportdesigner/util/IParameterizedDataSource.html> ]. The above example obtains data from the Woodview database.

The query contains two parameters. The first parameter allows the user to set the maximum price for the query results. The parameter is of type `double` and the default is set to `500.00`. The second parameter is a custom parameter and its purpose is to give the user a yes or no option to determine whether or not to show popular items only. To construct a custom parameter, create a class that extends the `SimpleQueryInParam` [ [943](http://data.quadbase.com/Doc-</a></p>
</div>
<div data-bbox=)

s70/eres/help/apidocs/quadbases/reportdesigner/util/SimpleQueryInParam.html ] class, implements the IQuery-ParamValuesProvider [ <http://data.quadbases.com/Docs70/eres/help/apidocs/quadbases/reportdesigner/util/IQuery-ParamValuesProvider.html> ] interface and overwrites the getSelectionChoices() method. If the user selects yes, then in getResultSet() only items with less than 15 UnitsInStock will be returned.

You can use the same constructor as before to create a report using this parameterized class. When using parameterized class file as the data source, keep in mind that you can not map it to a column directly (i.e. set the table name and column name in the SimpleQueryInParam constructor). In order to map a parameter to a column, you will have to use a custom parameter similar to the one shown above. However, the getSelectionChoices() method will look as follows:

```
public Vector getSelectionChoices() {
    System.out.println("getSelectionChoices called");
    try {
        Class.forName("org.hsqldb.jdbcDriver");

        String url = "jdbc:hsqldb:help/examples/DataSources/database/woodview";
        Connection conn = DriverManager.getConnection(url, "sa", "");
        Statement stmt = conn.createStatement();
        String query = "SELECT DISTINCT " + getColumnName() + " FROM " +
getTableName();
        ResultSet rs = stmt.executeQuery(query);
        Vector v = new Vector();

        while (rs.next()) {
            switch (getSqlType()) {
                case Types.INTEGER:
                    v.add(new Integer(rs.getInt(1)));
                    break;

                case Types.VARCHAR:
                    v.add(rs.getString(1));
                    break;
            }
        }

        stmt.close();
        conn.close();

        return v;
    } catch (Exception ex) {
        ex.printStackTrace();
    }
    return null;
}
```

Full Source Code [ <http://data.quadbases.com/Docs70/help/manual/code/src/ClassDataMapToColumn.zip> ]

In addition to mapping the parameter to a column, this example also uses a multi-value parameter. Although the process is similar to using a single value parameter, there are some places where the code deviates from the norm. Please see the comments in the above source code for details. Also, more information on using parameters from the API can be found in Appendix 8.B.7 - Parameterized Report.

For a working example using data passed in from a custom class file, see Appendix 8.B.3.2 - Creating the Report.

## 8.A.6. Data from Enterprise Java Beans (EJBs)

Data can be passed from an EJB data source to the report by allowing users to query data directly from an entity bean. To add an EJB as a data source, the EJB must first be deployed in the application server and the client JAR

file containing the appropriate stub classes must be added to your classpath (or the `-classpath` argument of the ERES Server batch file when using the API in conjunction with ERES Server).

To construct a report using an EJB as a data source, first you must construct a `EJBInfo` object with information for connecting to the EJB. The constructor is shown below:

```
public EJBInfo(java.lang.String jndiName,
              java.lang.String homeName,
              java.lang.String remoteName,
              java.lang.String selectedMethodName,
              java.lang.Object[] selectedMethodParamVal)
```

For example, the following lines create an `EJBInfo` instance that connects to the `ProductEJB` data source.

```
Object[] vals = new Object[1];
vals[0] = new String("Printer");

EJBInfo.ejbInfo = new EJBInfo("ProductEJB", "ejb.ProductHome",
                              "ejb.Product", "findByCategory", vals);
```

Finally, use the following constructor to create the `QbReport` object:

```
public QbReport(Object parent, int reportType, EJBInfo.ejbInfo, ColInfo[]
               mapping, String template);
```

## 8.A.7. Data from a SOAP Data Source

ERES allows to retrieving data from SOAP services. To use a SOAP data source, you need to provide a location of WSDL file, which contains all the necessary information. The location can be either absolute path on the server or path relative to your ERES installation directory or URL.

Constructor for SOAP data source looks like this:

```
SOAPQueryFileInfo(String wsdlURI, QName serviceName, String portName, String
                  operationName, XMLFieldInfo[] xmlFieldInfo, SOAPParam[] parameters)
```

The `wsdlURI` is URL or absolute/relative path to the WSDL file. The `serviceName` of type `QName` is name of SOAP service. The following constructor is used:

```
QName(String namespaceURI, String localPart),
```

where the `namespaceURI` is URI namespace (a part of the `serviceName` string in curly brackets) and the `localPart` is a service name (the rest of the string). For example, in the following servicename `{http://www.webservicex.net}WeatherForecast`. The `http://www.webservicex.net` is `namespaceURI` and the `WeatherForecast` is a service name.

The `portName` and `operationName` in the `SOAPQueryFileInfo` constructor are names of a port and operation. The `XMLFieldInfo` array is the same as for XML data sources (see Appendix 8.A.3 - Data from an XML Data Source). Finally, the `SOAPParam` field is dealing with parameters and uses the following constructor:

---

```
SOAPParam(String paramName, int sqlType, String paramPrompt, Object
    defaultValue, Object value, boolean alwaysUseDefault),
```

where the `paramName` is a parameter name, the `sqlType` represents parameter data type, which is a constant from `java.sql.Types`. The `paramPrompt` is the parameter prompt string, the `defaultValue` is the parameter default value, the `value` is the parameter value and the `alwaysUseDefault` of boolean type says whether the default value will be always used or not. This means that this parameter value will be fixed and users will not be prompted for it.

To create the `QbReport/QbChart` object use the constructors below:

```
QbReport(Object parent, int reportType, SOAPQueryFileInfo soapInfo,
    ColInfo[] mapping, String template, Properties props)
```

```
QbChart(Applet applet, int dimension, int chartType, SOAPQueryFileInfo
    soapInfo, boolean doTransposeData, int[] transposeCol, IColumnMap cmap,
    String template)
```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/SOAP.zip> ]

## 8.A.8. Data from Multiple Data Sources

ERES also provides the functionality to merge multiple data sources together. You can create a `DataSheet` object from any combination of data sources, for example, data file, database or `IResultSet`. You can use a `QbReport` constructor to create a report object from an array of `DataSheet` objects.

The following example program fragment demonstrates how to combine the data from the examples in the above sections:

```
// Declaration of DataSheet object to be used to merge data
DataSheet dataSheet[] = new DataSheet[3];

// Declaration For Database (Data from Database section)
DBInfo dbinfo = new DBInfo("jdbc:odbc:woodview",
    "sun.jdbc.odbc.JdbcOdbcDriver", "", "", "select * from Orders");

//Declaration For Data Passed in from Memory (Data Passed in from Memory
    section)
String dataType[] = {"varchar", "decimal"};
String fieldName[] = {"People", "Sales"};
String records[][] = {{"Peter", "93"}, {"Peter", "124"},
    {"John", "110"}, {"John", "130"},
    {"Mary", "103"}, {"Mary", "129"}};
DbData data = new DbData(dataType, fieldName, records);

// Create DataSheet from data file (Data from a Text File section)
// DataSheet(Applet applet, String dataFile)
dataSheet[0] = new DataSheet(this, "../data/1_A_7_TextData.dat");

// Create DataSheet from database (Data from a Database section)
// DataSheet(Applet applet, IDatabaseInfo dbInfo)
dataSheet[1] = new DataSheet(this, dbinfo);

// Create DataSheet from IResultSet (Data Passed in from Memory section)
// DataSheet(Applet applet, IResultSet data)
dataSheet[2] = new DataSheet(this, data);
```

Then, use the following `QbReport` constructor to create the report:

```
QbReport(java.lang.Object parent, int reportType, DataSheet[] dataSheet,
         ColInfo[] mapping, java.lang.String template)
```

Note that after you have created a `DataSheet` object, you can modify it (add, delete, or update row values) by using `DataSheet` API. Data is not refreshable if merging data from `IResultSet`.

## 8.B. Creating the Report

To create a new report, you must specify the report type, the input data source information, and a mapping of the data columns to the respective columns of the report. In this appendix, we look at the various report types and the methods used to map the columns. There are also a number of fully functional examples in this appendix.

Please note that unless otherwise specified, all examples use the Woodview Access database and the data is obtained using a System ODBC DSN called *Woodview*.

There are five basic report types: *Simple Columnar*, *Summary Break*, *Cross Tab*, *Master & Details*, and *Mailing Labels* reports. Every report has a column mapping defined as well as properties specific to the type of the report.

The column mapping is done in the API using the `ColInfo` [ <http://data.quadbase.com/Docs70/eres/help/apidocs/quadbase/reportdesigner/util/ColInfo.html> ] class (located in the `quadbase.reportdesigner.util` [ <http://data.quadbase.com/Docs70/eres/help/apidocs/quadbase/reportdesigner/util/package-summary.html> ] package). You define the column mapping by declaring a `ColInfo` array object. Each element in the array represents the column's respective column. You can also assign specific properties for each element of the `ColInfo` array object, depending on the type of the report desired, and create the report.

The `QbReport` class contains numerous constructors. However, most constructors can be broken down into three sets of parameters. The first set is only the parent object. The parent object must be set if the report is to be used in an applet context, it can be null otherwise. The second set of parameters are always required and they determine the type of report you are constructing, the data source, and the mapping. There is a great deal of information in the two Appendices regarding this group of parameters. Finally, the third set of parameters are options and properties. For example, most constructors have the option to specify a template, although it is not necessary. Some options are specific to certain report types, for example the boolean parameter `sideBySideLayout` is only relevant if you are creating a *Master & Details* report. Some constructors contain a `Properties` parameter that allows you to set multiple properties and group them into one properties object. For more information on the various options and properties, please see the APIDocs [ <http://data.quadbase.com/Docs70/eres/help/apidocs/index.html> ].



### Caution

As you may know, creating objects in java is a resource intensive operation. It is always recommended that you do not create too many objects. One way to conserve resource is to reuse `QbReport` objects whenever possible. For example, if a lot of your users request for a *Simple Columnar Report* in your website, instead of creating a new `QbReport` object when each such request is received, you can have one (or a limited number of) such `QbReport` object(s) created and reuse the object(s) by simply modifying the data and attributes of the report for each particular request.

### 8.B.1. Simple Columnar

The *Simple Columnar* report is the most basic of all the types supported by ERES. It presents columnar data in a single table without any groupings, or breaks.

Here is an example:

Product Type	Product Name	Price
12	Chair	\$20

Product Type	Product Name	Price
12	Table	\$30
14	Cabinet	\$20
14	Table	\$50

### 8.B.1.1. Column Mapping

With *Simple Columnar* reports, all you need to do is define the `ColInfo` array object with respect to the data source. For instance, given the dataset below:

```
String dataType[] = {"String", "String", "String"};
String fieldName[] = {"Product Type", "Product Name", "Price"};
String records[][] = {{ "12", "Chair", "$20"},
                      { "12", "Table", "$30"},
                      { "14", "Cabinet", "$20"},
                      { "14", "Table", "$50"}};
```

The column mapping is set so that the columns of the data source are mapped to the report. Thus Column Zero of the report corresponds to Column Zero of the data source, Column One of the report to Column One of the data source and so on and so forth. Thus, the following report is created:

Product Type	Product Name	Product Price
12	Chair	\$20
12	Table	\$30
14	Cabinet	\$20
14	Table	\$50

*Report after mapping and setting the Table Header Names*

To generate the aforementioned report, the following column mapping has to be set using the ERES API:

```
ColInfo[] colInfo = new ColInfo[3];
colInfo[0] = new ColInfo(0);
colInfo[0].setName("Product Type");
colInfo[1] = new ColInfo(1);
colInfo[1].setName("Product Name");
colInfo[2] = new ColInfo(2);
colInfo[2].setName("Product Price");
```

In the above code, the names for the columns are also changed to `Product Type`, `Product Name` and `Product Price` respectively rather than keeping the Header names defined in the original data.

The column mapping need not be in order or follow the same order as the data. You can also selectively choose the columns you desire. For instance, given a data source that has multiple columns, you can write the following code:

```
ColInfo[] colInfo = new ColInfo[3];
colInfo[0] = new ColInfo(2);
colInfo[0].setName("Product Name");
colInfo[1] = new ColInfo(7);
colInfo[1].setName("Units Sold");
```

```
colInfo[2] = new ColInfo(4);
colInfo[2].setName("Unit Price");
```

Here, the Column 0 of the report is mapped to column 2 of the data source, Column 1 of the report is mapped to column 7 of the data source and Column 2 of the report is mapped to column 4 of the data source. In addition to setting the column mappings, the above example also sets the Table Header for each column.

### 8.B.1.2. Creating the Report

Constructing a *Simple Columnar* report is relatively straight forward. We have already discussed how to set the `ColInfo` array and how to obtain the data in previous sections. The following code demonstrates how to create a *Simple Columnar* report.

```
// Data passed in an array in memory
DbData data = new DbData(dataType, fieldName, records);

// Set Column Mapping
ColInfo colInfo[] = new ColInfo[3];
colInfo[0] = new ColInfo(0);
colInfo[0].setName("Product Type");
colInfo[1] = new ColInfo(1);
colInfo[1].setName("Product Name");
colInfo[2] = new ColInfo(2);
colInfo[2].setName("Product Price");

// Create Report
QbReport report = new QbReport
    (parent, // Parent
     QbReport.COLUMNAR, // Type of Report
     data, // Data
     colInfo, // Column Mapping - use the column mapping in the template
     "SimpleColumnar.rpt"); // Template Name
```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/SimpleColumnar.zip> ]

Exported Unformatted Results [ <http://data.quadbase.com/Docs70/help/manual/code/export/SimpleColumnarNF.html> ]

Exported Formatted Results [ <http://data.quadbase.com/Docs70/help/manual/code/export/SimpleColumnar.html> ]

The Exported Unformatted Results link shows how the report appears when it was first created without any formatting or templates. The Exported Formatted Results presents the outcome after applying the template.

Even though this example creates a report from scratch, we still recommend that you apply a template. If you look at the source code, notice the large commented section after the `QbReport` constructor. This section modifies the visual appearance of the report in exactly the same way as applying the template. As you can imagine, writing the code takes much more effort since there is no visual aid to help you gauge position and dimensions.

When you apply a template, you have the option of using the column mapping stored in the template. In the above example, you could replace the `colInfo` in the constructor with `null` and it would still generate the same report. This is because the column mapping in the template is exactly the same. If you set the `ColInfo` and apply a template, the constructed report will use the `ColInfo` you supplied and ignore the column mapping in the template.

### 8.B.2. Summary Break

For a *Summary Break* report, you need to define a Row Break. The Row Break defines when to break the report and insert column summaries. The report essentially gets “broken” everytime the data in the “break” column changes. An aggregation must be set for any columns that are not specified as Row Breaks, even if the columns are non-

numeric (set the aggregation to `NONE`). This is so that column summaries can be generated based on the aggregation. For instance, given the dataset below:

Order #	Product	Quantity
12	Chair	2
12	Table	3
14	Cabinet	2
14	Table	5

*Original Data*

If Column 0 (Order #) is assigned as Row Breaks, the Aggregation for Column 1 (Product) is set to `NONE`, and the Aggregation for Column 2 (Quantity) is set to `SUM`, we get the following report.

Order #	Product	Quantity
12	Chair	2
	Table	3
		5
14	Cabinet	2
	Table	5
		7

*Report after applying Row Break and Aggregation*

### 8.B.2.1. Column Mapping

Setting the column mapping for *Summary Break* reports requires that two more properties to be set than the *Simple Columnar* report. First, the row break columns must be specified for row group columns. Second, columns that are not row break columns must specify the aggregation type. For example, to generate the report in the previous section, you need the set the following lines of code:

```
ColInfo[] colInfo = new ColInfo[3];
colInfo[0] = new ColInfo(0);
colInfo[0].setRowBreak(true);
colInfo[1] = new ColInfo(1);
colInfo[1].setAggregation(false, ColInfo.NONE);
colInfo[2] = new ColInfo(2);
colInfo[2].setAggregation(false, ColInfo.SUM);
```

Again, the first step is to map the column in the report with the column from the data source. Then, Column Zero is set to be a row break column. Since Columns One and Two are not set as row break columns, you must set an aggregation for them even if it is `NONE`. There are two parameters in the `setAggregation` method, the first parameter determines if the report uses column aggregation. If column aggregation is used, the individual rows of data will be hidden and only the aggregation will be displayed. If you set the first parameter to `true`, the other non row break column must also be set to use column aggregation. The second parameter determines the type of aggregation. Here is a list of all available aggregations in the `ColInfo` class:

AVG	MAX	STDDEV
COUNT	MEDIAN	SUM
COUNT-DISTINCT	MIN	SUMSQUARE

FIRST	NONE	VARIANCE
LAST		

## 8.B.2.2. Creating the Report

The following example shows how to create a *Summary Break* report:

```
ColInfo[] colInfo = new ColInfo[5];
colInfo[0] = new ColInfo(1);
colInfo[0].setRowBreak(true);
colInfo[1] = new ColInfo(9);
colInfo[1].setRowBreak(true);
colInfo[2] = new ColInfo(4);
colInfo[2].setAggregation(true, ColInfo.SUM);
colInfo[3] = new ColInfo(5);
colInfo[3].setAggregation(true, ColInfo.SUM);
colInfo[4] = new ColInfo(6);
colInfo[4].setAggregation(true, ColInfo.SUM);

QbReport report = new QbReport
    (parent, // parent
     QbReport.SUMMARY, // Sumamry Break Report
     "sample.dat", // filename
     colInfo, // column information
     null);
try {
    // Apply template including scripts and formula
    report.applyTemplate("SummaryBreak.rpt", true);
} catch (Exception e) {
    e.printStackTrace();
}
```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/SummaryBreak.zip> ]

Exported Unformatted Results [ <http://data.quadbase.com/Docs70/help/manual/code/export/SummaryBreakNF.html> ]

Exported Formatted Results [ <http://data.quadbase.com/Docs70/help/manual/code/export/SummaryBreak.html> ]

The Exported Unformatted Results link shows how the report appears when it was first created without any formatting or templates. The Exported Formatted Results presents the outcome after applying the template.

First thing you may notice is that the column mapping is different from the previous examples. This is often the case when using a text file as the data source. In this example, we are mapping column 0 from the report with column 1 in the text file, column 1 from the report with column 9 in the text file, and so on.

Also, notice that the example uses a template to format the appearance of the report eliminating the need to write excessive amounts of code. In this example, we cannot include the template in the constructor because by default, adding the template to the constructor will not import the formula and scripts. The way to import both the formula and the scripts is by calling the method `applyTemplate` after constructing the `QbReport` object. The two arguments that you will need to pass in are the file path of the template and a boolean specifying whether you want to import formula and scripts.

An alternate way to create a summary break report is by specifying that your data source is already sorted. That way, ERES will build a summary break report in the same manner as for database data. This feature is available for all data sources including text, XML, and class data. To do this, simply call the `QbReport` constructor with the boolean parameter `isDataSorted` set to `true`:

```
new QbReport(parent, reportType, data, mapping, template, sideBySideLayout,
  isDataSorted);
```

## 8.B.3. CrossTab

A *Crosstab* report is a report format that shows and summarizes columnar data in a matrix-like form. *Crosstab* reports often resemble spreadsheets. Both rows and columns are summarized, allowing multi-dimensional data to be displayed in a 2 dimensional format.

In addition to defining *Row Breaks*, you will also need to specify a column that serves as the *Column Break* as well as a column that serves as the *Column Break Value*. The *Column Break* column gets included in the *Table Header* with a separate column for each unique entry in the selected column while the *Column Break Value* column represents the field that is summarized in the report. You will also need to specify the type of *Aggregation* for the *Column Break Value* column. For instance, given the dataset below:

Region	Product	Total Sales
East	Chair	14500
Midwest	Chair	13250
South	Chair	15252
East	Table	10550
Midwest	Table	9150
South	Table	11250

### *Original Data*

If *Column 0 (Region)* is assigned as *Row Break*, *Column 1 (Product)* is assigned as the *Column Break*, and *Column 2 (Sales)* assigned as the *Column Break Value* with a summation aggregation, we get the following report:

Region	Chair	Table	Total Sales
East	14500	10550	25050
Midwest	13250	9150	22400
South	15252	11250	26502
	43002	30950	73952

### *Report after applying Column Break and Column Break Value with Aggregation*

This example demonstrates several qualities frequently exhibited by *CrossTab* reports. Notice that the distinct *Product* names are transformed to become column headers. This means that the number of columns in the report will vary based on the data source. Also, the numeric values represent *Total Sales* for a particular combination of *Product* and *Region*. The cells to the right and bottom of the data are aggregations cells. The *Total Sales* at the end of each row sums up the *Total Sales* for each region, while the values at the bottom of each column sums up the *Total Sales* of each product.

### 8.B.3.1. Column Mapping

To generate the aforementioned report, the following column mapping has to be set using the ERES API:

```
ColInfo[] colInfo = new ColInfo[3];
colInfo[0] = new ColInfo(0);
// use this column as the row break. The number of
// new groups would be equal to the number of unique
```

---

```
// fields present.
colInfo[0].setRowBreak(true);

colInfo[1]=new ColInfo(1);
// use this column as the column break. The number of
// new columns would be equal to the number of unique
// column fields present.
colInfo[1].setColumnBreak(true);

colInfo[2] = new ColInfo(2);
// use this column as the value fields for the
// newly created chair and table columns
colInfo[2].setColumnBreakValue(true);
// aggregate by adding up the individual fields to
// get the summary field.
colInfo[2].setAggregation(ColInfo.SUM);
```

In the above code, the names for the columns are not set as the original names of the columns from the data source sufficed.

Note that in the example given above, only a single Column Break and a single Column Break Value was specified. However, just like Row Breaks, multiple Column Breaks and Column Break Values can be specified.

For example:

```
ColInfo[] colInfo = new ColInfo[7];
colInfo[0] = new ColInfo(0);
colInfo[0].setRowBreak(true);

colInfo[1]=new ColInfo(1);
colInfo[1].setRowBreak(true);

colInfo[2]=new ColInfo(2);
colInfo.setColumnBreak(true);

colInfo[3]=new ColInfo(3);
colInfo[3].setColumnBreak(true);

colInfo[4]=new ColInfo(4);
colInfo[4].setColumnBreakValue(true);
colInfo[4].setAggregation(ColInfo.SUM);

colInfo[5]=new ColInfo(5);
colInfo[5].setColumnBreakValue(true);
colInfo[5].setAggregation(ColInfo.SUM);

colInfo[6]=new ColInfo(6);
colInfo[6].setColumnBreakValue(true);
colInfo[6].setAggregation(ColInfo.SUM);
```

In the above code, Columns 2 and 3 of the data source are set to be Column Break columns while Columns 4, 5 and 6 are set to be Column Break Value columns.

### 8.B.3.2. Creating the Report

The following example shows how to create a *CrossTab* report.

```

ColInfo[] colInfo = new ColInfo[4];
colInfo[0] = new ColInfo(0);
colInfo[0].setRowBreak(true);

colInfo[1] = new ColInfo(1);
colInfo[1].setColumnBreak(true);

colInfo[2] = new ColInfo(2);
colInfo[2].setColumnBreakValue(true);
colInfo[2].setAggregation(ColInfo.AVG);

QbReport report = new QbReport
    (parent, // Parent
     QbReport.CROSSTAB, // Type of Report
     QbReport.CLASSFILE, // Data
     "ClassFile",
     colInfo, // Column Mapping
     "CrossTab"); // Template Name

```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/CrossTab.zip> ]

Exported Unformatted Results [ <http://data.quadbase.com/Docs70/help/manual/code/export/CrossTabNF.html> ]

Exported Formatted Results [ <http://data.quadbase.com/Docs70/help/manual/code/export/CrossTab.html> ]

The Exported Unformatted Results link shows how the report appears when it was first created without any formatting or templates. The Exported Formatted Results presents the outcome after applying the template.

The example uses a class file as the data source. The class file generates the temperatures everytime you run the program so the results will vary. For more information on class file data sources, see Appendix 8.A.5 - Data passed in a Custom Implementation.

### 8.B.3.3. Fixed-field Crosstab Report

The fixed-field crosstab option improves some of the limitations of the free-form crosstab implementation, by making it easier for users to design crosstab reports that can expand and contract with changing data.

The key difference between a fixed field and a free-form crosstab is that in the design view, you are not able to position or control the individual columns for the crosstab report. Instead you can set formats for groups of elements (row break, headers, footers and formulas). The actual report is only constructed during running/preview. Since the crosstab table is essentially constructed from scratch every time the report is run it is easier to create a smoothly contracting and expanding crosstab table.

#### 8.B.3.3.1. Creating the Report

The following example shows how to create a multi-dimensional (extra row/column breaks) fixed-field crosstab report with 3 column breaks.

```

ColInfo colInfo[] = new ColInfo[4];

colInfo[0] = new ColInfo(0);
colInfo[1] = new ColInfo(1);
colInfo[2] = new ColInfo(2);
colInfo[3] = new ColInfo(3);

colInfo[0].setRowBreak(true);
colInfo[1].setColumnBreak(true);
colInfo[2].setColumnBreak(true);

```

```
colInfo[3].setColumnBreakValue(true);
colInfo[3].setAggregation(ColInfo.SUM);

Properties props = new Properties();
props.put("crossTabFreeForm", "false"); // Using fixed-field form
props.put("crossTabSummaryPositionL", "false"); // Draw summary column to
    the right of the details matrix
props.put("crossTabFormulaOnHeader", "true"); // Draw formula column in the
    header section
props.put("crossTabColBkValAlignH", "false"); // Align Column Break Value
    vertically

QbReport report = new QbReport
    (parent, // Parent
    QbReport.CROSSTAB, // Report Type
    QbReport.DATAFILE, // Data Type
    "sample.dat", // Filename
    colInfo, // Column Mapping
    "field.rpt", // Template
    props); // Properties
```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/FixedFieldCrosstab.zip> ]

Exported Results [ [http://data.quadbase.com/Docs70/help/manual/code/export/Fixed\\_field\\_Crosstab.pdf](http://data.quadbase.com/Docs70/help/manual/code/export/Fixed_field_Crosstab.pdf) ]

In this example, we are mapping column 0 from the report with column 0 in the text file, column 1 from the report with column 1 in the text file, and so on. In the above code, columns 1, 2 and 3 of the data source are set to be Column Break columns and column 4 is set to be Column Break Value column with an sum aggregation. There are different crosstab options, because you cannot individually format or position crosstab options in the fixed-field layout. In this example the summary column is set to be drawn to the right of the report data, the formula column is set to be drawn in the header section and the column break value is set to be aligned vertically.

For more information about crosstab, see Section 4.1.2.3 - Crosstab Report.

## 8.B.4. Master & Details

A *Master & Details* report is a set of tabular data that is grouped according to a *Master* field. This report type is most commonly used when some fields in your data table are related to many other fields. A good example of this is an invoice. For each order number, there will be customer information, and several items with pricing information. A *Master & Details* report would be used to group the information according to order number.

In a *Master & Details* report, the data is grouped based on a “Primary” and/or “Master” column(s). Here, you **must** define a *Primary Key* column. The *Primary Key* column (order number in the invoice example) determines the grouping of the report. A new group will be created every time there is a value in the selected column (i.e. the *Primary Key* column). Please note that only one column can serve as the *Primary Key*. A *Master* field or several *Master* fields can also be selected in addition. Any column selected as the *Master* field (customer information) will result in the placing of the column value in the Group Header instead of the Data Section of the report. By definition, the *Primary Key* value must be unique for each grouping of the *Master* field column(s). For instance, given the dataset below:

Order #	Customer Name	Product	Unit Price	Quantity
12	Paul Campbell	Chair	\$24.95	4
12	Paul Campbell	Table	\$127.50	1
14	Sally Hayes	Cabinet	\$227.25	2

Order #	Customer Name	Product	Unit Price	Quantity
14	Sally Hayes	Chair	\$24.95	2
14	Sally Hayes	Table	\$127.50	1

*Original Data*

If Column 0 (Order #) is assigned as the Primary Key and Column 1 (Customer Name) is assigned as the Master field, we get the following report:

<b>Order #</b>	<b>12</b>	
<b>Customer Name</b>	<b>Paul Campbell</b>	
<b>Product</b>	<b>Unit Price</b>	<b>Quantity</b>
Chair	\$24.95	4
Table	\$127.50	1
<b>Order #</b>	<b>14</b>	
<b>Customer Name</b>	<b>Sally Hayes</b>	
<b>Product</b>	<b>Unit Price</b>	<b>Quantity</b>
Cabinet	\$227.25	2
Chair	\$24.95	2
Table	\$127.50	1

*Report after applying Primary Key and Master field***8.B.4.1. Column Mapping**

To generate the aforementioned report, the following column mapping has to be set using the ERES API:

```
ColInfo[] colInfo = new ColInfo[5];
colInfo[0] = new ColInfo(0);
colInfo[0].setPrimaryKey(true);
colInfo[1]=new ColInfo(1);
colInfo[1].setMaster(true);
colInfo[2] = new ColInfo(2);
colInfo[3] = new ColInfo(3);
colInfo[4] = new ColInfo(4);
```

In the above code, the names for the columns are not set as the original names of the columns from the data source sufficed.

You can select any number of columns to be in the master section. The only restriction is that the primary key is automatically placed in the master section. If this is undesired, you can set that column to be invisible.

**8.B.4.2. Creating the Report**

The following example shows how to create a *Master & Details* report.

```
String url = "jdbc:hsqldb:database/woodview";
String driver = "org.hsqldb.jdbcDriver";
String query = "...";
```

```

DBInfo dbInfo = new DBInfo(url, driver, "sa", "", query);

ColumnInfo[] colInfo = new ColInfo[16];

for(int i = 0; i < colInfo.length; i++){
    colInfo[i] = new ColInfo(i);
    if(i < 8)
        colInfo[i].setMaster(true);
}
colInfo[0].setPrimaryKey(true);
colInfo[15].setVisible(false);

QbReport report = new QbReport
    (parent,                // Parent
    QbReport.MASTERDETAILS, // Type of Report
    dbInfo,                // Database Info
    colInfo,               // Column Mapping
    null);                 // Template Name
try {
    // Apply Template with Formula and Script
    report.applyTemplate("MasterDetails", true);
} catch( Exception e ) {
    e.printStackTrace();
}

```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/MasterDetails.zip> ]

Please see the full source code for the query details. The Exported Unformatted Results link shows how the report appears when it was first created without any formatting or templates. The Exported Formatted Results presents the outcome after applying the template.

This example uses the WoodView HSQL, so you will need to add database HSQL JDBC driver (`hsqldb.jar`) to your classpath. The driver is located in the `<ERESInstall>/WEB-INF/lib` directory. For more information on using a database data source, see Appendix 8.A.1 - Data from a Database.

Since there are a total of 16 columns in this example, we use a for-loop to set the column info. The first eight columns are set as master columns and will be shown in the Group Header section. Column 0, `OrderID`, is set as the primary key.

We apply the template after constructing the `QbReport` object in order to import the formula and scripts stored in the template.

You can also position the master section to be side by side with the rest of the columns. To do so, use the following `QbReport` constructor and pass in `true` for the `sideBySideLayout` parameter.

```

QbReport(java.lang.Object parent, int reportType, IDatabaseInfo dbinfo,
    ColInfo[] mapping, java.lang.String template, boolean sideBySideLayout)

```

## 8.B.5. Mailing Labels

The *Mailing Labels* report presents data in a columnar format without any groupings or breaks. It is similar to the *Simple Columnar* report, all you need to do is merely define the `ColumnInfo` array with respect to the data source. For instance, given the dataset below:

Name	Company	Region
Mary	ABC Inc.	New York

Name	Company	Region
Peter	GHI Ltd	London

*Original Data*

The column mapping is set so that the columns of the data source are mapped to the report. Thus Column 0 of the report corresponds to column 0 of the data source, Column 1 of the report to column 1 of the data source and so on and so forth. Thus the following report is created:

<b>Name:</b>	Mary
<b>Company:</b>	ABC Inc.
<b>City:</b>	New York
<b>Name:</b>	Peter
<b>Company:</b>	GHI Ltd
<b>City:</b>	London

*Report after mapping and setting the Table Header Names*

### 8.B.5.1. Column Mapping

To generate the aforementioned report, the following column mapping has to be set using the ERES API:

```
ColInfo[] colInfo = new ColInfo[3];
colInfo[0] = new ColInfo(0);
colInfo[1] = new ColInfo(1);
colInfo[2] = new ColInfo(2);
colInfo[2].setName("City");
```

In the above code, the name for the third column was also changed to `City` rather than keeping the Header names defined in the original data.

The column mapping need not be in order or follow the same order as the data. You can also selectively choose the columns you desire. For instance, given a data source that has multiple columns, you can write the following code:

```
ColInfo[] colInfo = new ColInfo[3];
colInfo[0] = new ColInfo(2);
colInfo[1] = new ColInfo(7);
colInfo[2] = new ColInfo(4);
```

Here, the Column 0 of the report is mapped to column 2 of the data source, Column 1 of the report is mapped to column 7 of the data source, and Column 2 of the report is mapped to column 4 of the data source.

### 8.B.5.2. Creating the Report

The following example shows how to create a *Mailing Labels* report.

```
String xmlfilename = "Inventory.xml";
String dtdfilename = "Inventory.dtd";
String xmlcondition = "/Inventory/Category/Product/UnitsInStock < 15";

XMLFieldInfo[] fields = new XMLFieldInfo[8];
fields[0] = new XMLFieldInfo(new String[] { "Inventory", "Category" },
    "CategoryName");
```

```

fields[1] = new XMLFieldInfo(new String[] { "Inventory", "Category",
    "Product" },
    "ProductID");
fields[2] = new XMLFieldInfo(new String[] { "Inventory", "Category",
    "Product",
    "ProductName" });

fields[3] = new XMLFieldInfo(
    new String[] { "Inventory", "Category", "Product", "UnitPrice" });
fields[3].setElementDataType(DTDDDataType.DOUBLE);

fields[4] = new XMLFieldInfo(
    new String[] { "Inventory", "Category", "Product", "Material" });
fields[5] = new XMLFieldInfo(new String[] { "Inventory", "Category",
    "Product", "Material",
    "Units" });
fields[5].setElementDataType(DTDDDataType.INT);

fields[6] = new XMLFieldInfo(new String[] { "Inventory", "Category",
    "Product", "Material",
    "CostPerUnit" });
fields[6].setElementDataType(DTDDDataType.DOUBLE);

fields[7] = new XMLFieldInfo(new String[] { "Inventory", "Category",
    "Product",
    "UnitsInStock" });
fields[7].setElementDataType(DTDDDataType.INT);

XMLFileQueryInfo xmlInfo = new XMLFileQueryInfo(xmlfilename, fields,
    xmlcondition, fields,
    dtdfilename, false, null);

ColumnInfo[] colInfo = new ColumnInfo[8];
for (int i = 0; i < 8; i++) {
    colInfo[i] = new ColumnInfo(i);
}

QbReport report = new QbReport
    (parent, // Parent
    QbReport.MAILINGLABELS, // Type of Report
    xmlInfo, // XML Info
    colInfo, // Column Mapping
    null, // Template Name
    false); // Side By Side

try {
    // Apply Template with Formula and Script
    report.applyTemplate("MailingLabels.rpt", true);
} catch (Exception e) {
    e.printStackTrace();
}

```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/MailingLabels.zip> ]

Exported Unformatted Results [ <http://data.quadbase.com/Docs70/help/manual/code/export/MailingLabelsNF.html> ]

Exported Formatted Results [ <http://data.quadbase.com/Docs70/help/manual/code/export/MailingLabels.html> ]

The Exported Unformatted Results link shows how the report appears when it was first created without any formatting or templates. The Exported Formatted Results presents the outcome after applying the template. This example uses a XML file as the data source, for more information on using an XML data source, see Appendix 8.A.3 - Data from an XML Data Source.

We apply the template after constructing the `QbReport` object in order to import the formula and scripts stored in the template.

## 8.B.6. Formula

You are not restricted to just using the original data in generating the columns for the report. You may create your own columns with your own value or use formula to calculate over one or more columns. For instance, given the dataset below:

Units Sold	Product	Unit Price
12	Chair	\$25
2	Table	\$38
18	Cabinet	\$21
4	Sofa	\$57

### *Original Data*

If Column 0 (`Units Sold`) and Column 2 (`Unit Price`) were multiplied, you can get a new column, which would have the total revenue for that product, thus producing the following report:

Product	Units Sold	Unit Price	Total Revenue
Chair	12	\$25	\$300.00
Table	2	\$38	\$76.00
Cabinet	18	\$21	\$378.00
Sofa	4	\$57	\$228.00

### *Report after applying formula and creating a new column*

To generate the aforementioned report, the following column mapping has to be set using the ERES API:

```
ColInfo[] colInfo = new ColInfo[4];
colInfo[0] = new ColInfo(1);
colInfo[1] = new ColInfo(0);
colInfo[2] = new ColInfo(2);
IObject salesColumn =
    NumericObject.multiply(NumericObject.getColumnValue(1),
        NumericObject.getColumnValue(2));
colInfo[3] = new ColInfo(salesColumn, "Total Revenue", Types.DOUBLE);
```

In the above code, the names for the columns are not set as the original names of the columns from the data source sufficed. For the newly created column, `Total Revenue` was specified as the header.

To learn more about formula, please see Section 4.1.6 - Using Formulas & the Formula Builder.

## 8.B.7. Parameterized Report

In addition to regular queries, you can pass in queries that have parameters and have the report prompt the user for values for the parameters, before generating the report. There is no limit to the number of parameters allowed in each query.

To use a parameterized query as your data source for your report, you must use the `SimpleQueryFileInfo` [ <http://data.quadbase.com/Docs70/eres/help/apidocs/quadbase/reportdesigner/util/SimpleQueryFileInfo.html> ] class, which implements `IQueryFileInfo` [ <http://data.quadbase.com/Docs70/eres/help/apidocs/quadbase/reportdesigner/util/IQueryFileInfo.html> ], to pass in the query information. Since `SimpleQueryFileInfo` is a subclass of `DBInfo`, you can use the same parameters to create a `SimpleQueryFileInfo` object.

```
SimpleQueryFileInfo(java.lang.String url, java.lang.String driver,
    java.lang.String username, java.lang.String password, java.lang.String
    query)
```

In addition to creating the query object, you must also create the parameters using either an array of `SimpleQueryInParam` for single valued parameters or an array of `SimpleQueryMultiValueInParam` for multi valued parameters. Finally, you must pass an instance of the parameter set to the query object using the method `setInParam(ParamSet)`.

The constructors for the single valued parameter and multi valued parameter are identical since one is a subclass of the other.

```
SimpleQueryInParam(java.lang.String paramName, java.lang.String promptName,
    boolean mapToColumn, java.lang.String tableName, java.lang.String
    columnName, int sqlType, java.lang.Object defaultValue, java.lang.Object
    value)
```

The `paramName` is the name for the parameter, having two parameters with the same name is permitted provided that they are created in separate parameter instances. The `promptName` is the message the user will see when they are asked to input the value for the parameter.

The parameters `mapToColumn`, `tableName` and `columnName` determines if you would like to specify a column from the database whose values will be used for the parameter input. Selecting `true` modifies the parameter prompt that the end user will see when previewing or running the report in the Report Viewer. If you map the parameter to a database column, the user will be prompted with a drop-down list of distinct values from which to select a parameter value. If you do not map, set the `tableName` and `columnName` to `null`, the user will have to type in the specific parameter value. If you are querying the database from a parameterized class file and using the class file as the datasource, keep in mind that you can not set `mapToColumn` to `true`. For an alternative method of mapping to a column in a parameterized class file, see Appendix 8.A.5 - Data passed in a Custom Implementation.

The `sqlType` determines the datatype of the parameter. Select the constant that matches the datatype from the `java.sql.Types` class. The `defaultValue` is displayed in the input box when the prompt first appears. If you selected to map to column and provide a default value that does not exist, the first element of the drop down list will be displayed. The last parameter is of type object and if this parameter is not `null`, the parameter prompt will not be displayed. Instead of allowing the user to specify the value, it will use the value in this parameter for the query.

### 8.B.7.1. Creating the Report with Single Value Parameter

The following example shows how to create a report with a single value parameter.

```
SimpleQueryInParam inParam = new SimpleQueryInParam(
    "param", "Please select", true, "Categories",
    "CategoryName", Types.VARCHAR, "Arm Chairs", null);
SimpleQueryInParam[] paramSet = { inParam };

SimpleQueryFileInfo reportInfo = new SimpleQueryFileInfo(
    "jdbc:hsqldb:woodview",
    "org.hsqldb.jdbcDriver",
    "sa", "",
```

```

    "...");
reportInfo.setInParameter(paramSet);
// End Code : Adding Parameter Info

// Begin Code : Setting up Column Mapping
ColumnInfo colInfo[] = new ColumnInfo[4];
for (int i = 0; i < colInfo.length; i++) {
    colInfo[i] = new ColumnInfo(i);
}
colInfo[0].setPrimaryKey(true);
colInfo[1].setName("Product Name");
colInfo[2].setName("Price");
colInfo[3].setName("Units In Stock");
// End Code : Setting up Column Mapping

QbReport report = new QbReport(parent,
    QbReport.MASTERDETAILS,
    reportInfo,
    colInfo,
    "SingleValueParameter.rpt");

```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/SingleValueParameter.zip> ]

Exported Results [ <http://data.quadbase.com/Docs70/help/manual/code/export/SingleValueParameter.html> ]

This example uses the WoodView HSQL database, so you will need to add database HSQL JDBC driver (hsqldb.jar) to your classpath. The driver is located in the <ERESInstall>/lib directory. For more information on using a database data source, see Appendix 8.A.1 - Data from a Database.

This example uses the Woodview ODBC database, for more information on using a database data source, see Appendix 8.A.1 - Data from a Database.

Here the parameter is mapped to the CategoryName column of the Category table. The prompt will display "Please Select" and the values will be presented in a drop down list for the user to select.

## 8.B.7.2. Creating the Report with Multi Value Parameter

You can also assign a parameter to have multiple values, for example, in the case where a user wants to check against a range of values rather than just a single value. The range of values is usually specified within the IN clause of a SQL query. Note that ERES only considers parameters within the IN clause to be multi-value.

The following example shows how to create a report with a multi-value parameter.

```

SimpleQueryMultiValueInParameter inParam = new
SimpleQueryMultiValueInParameter("param",
    "Please select", true, "products", "productid", Types.INTEGER, new
Integer(1), null);
SimpleQueryMultiValueInParameter[] paramSet = { inParam };

SimpleQueryFileInfo rootInfo = new SimpleQueryFileInfo(
    "jdbc:hsqldb:woodview",
    "org.hsqldb.jdbcDriver",
    "sa", "",
    "select productid, productname " +
    "from Products where productid in (:param)");
rootInfo.setInParameter(paramSet);
// End Code : Adding Parameter Info

```

```
// Begin Code : Setting up Column Mapping
ColumnInfo rootColumnInfo[] = new ColumnInfo[2];
rootColumnInfo[0] = new ColumnInfo(0);
rootColumnInfo[0].setName("Product ID");
rootColumnInfo[1] = new ColumnInfo(1);
rootColumnInfo[1].setName("Product Name");

// End Code : Setting up Column Mapping

QbReport rootReport = new QbReport(
    parent,
    QbReport.MAILINGLABELS,
    rootInfo,
    rootColumnInfo,
    "MultiValueParameter.rpt");
```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/MultiValueParameter.zip> ]

Exported Results [ <http://data.quadbase.com/Docs70/help/manual/code/export/MultiValueParameter.html> ]

## 8.B.8. Sub-Report

You can also add sub-reports to the main report. This helps in presenting more data as two reports can be shown in one. For more information on sub-reports, please refer to Section 4.1.9 - Sub-Reports.

To create a report containing a sub-report, you need to create the sub-report object and then insert that sub-report object as a report cell object in the main report. The sub-report can be placed anywhere in the main report. To create a `SubReportObject`, use the following static method.

```
SubReport.createSubReport(QbReport parentReport, int reportType,
    IDatabaseInfo dbinfo, ColumnInfo[] mapping, java.lang.String template)
```

Similar to the `QbReport` constructors, there are numerous variations for the parameters to compensate for different data sources. The above method retrieves the data from a database. The sub-report can also contain parameters and you can set up parameter sharing from the API as well. Once the `SubReportObject` is created, it can be inserted into the report using the `addData(subReportObject)` method.

### 8.B.8.1. Creating the Report

The following example shows how to create a report that contains a sub-report.

```
// Begin Code : Creating the sub report
DBInfo subReportInfo = new DBInfo(
    "jdbc:hsqldb:woodview",
    "org.hsqldb.jdbcDriver",
    "sa",
    "",
    "SELECT Employees.FirstName + ' ' + Employees.LastName AS Employee, "
    +
    "Count(Order_Details.OrderID) AS Orders, "
    +
    "Sum( ( Products.UnitPrice + Order_Details.StainCost ) *
Order_Details.Quantity) AS Sales "
    +
    "FROM Employees, Products, Orders, Order_Details " +
    "WHERE ((Orders.EmployeeID = Employees.EmployeeID) " +
```

```

"AND (Orders.OrderID = Order_Details.OrderID) " +
"AND (Products.ProductID = Order_Details.ProductID)) " +
"AND (((Orders.OrderDate BETWEEN '2003-01-10' AND '2003-31-12')))" +
"GROUP BY Employees.FirstName + ' ' + Employees.LastName;");

ColInfo subReportColInfo[] = new ColInfo[3];
for (int i = 0; i < subReportColInfo.length; i++) {
    subReportColInfo[i] = new ColInfo(i);
}

try {
    // createSubReport(root report, report type, sub report data source, sub
    report column mapping, template)
    SubReportObject reportCell = quadbase.reportdesigner.ReportAPI.SubReport
        .createSubReport(
            rootReport,
            QbReport.COLUMNAR,
            subReportInfo,
            subReportColInfo,
            "SubReportSub.rpt");
    reportCell.setWidth(rootReport.getTable().getHeader().getWidth() - 2);
    reportCell.setHeight(2);
    // Or use the following method if the height of the subreport may change
    // reportCell.setResizeToFitContent(true);

    reportCell.setY(0);
    reportCell.setX(2);

    // Add SubReport to the Table Header
    rootReport.getTable().getHeader().addData(reportCell);
    rootReport.getTable().getHeader().setHeight(1.8);
} catch (Exception ex) {
    ex.printStackTrace();
}
// End Code : Creating the sub report

```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/SubReport.zip> ]

Exported Results [ <http://data.quadbase.com/Docs70/help/manual/code/export/SubReport.html> ]

The main report in this example obtains the data using a `DBInfo` object to create a summary break report. Both reports apply a template to reduce the amount of code in the example. The sub-report is created and inserted to the table header section of the main report. Notice that several lines of code are necessary to adjust the dimensions of the `SubReportObject` and the table header section in order to see the entire sub-report.

You can also include sub-reports that take a parameter from the main report and use it to populate the data within the sub-report. This helps to put in relevant data in sections within the main report. A parameterized sub-report is created by passing in a parameterized query for the sub-report and then specifying the column from the main report that provides the parameter for the sub-report's query. The parameter passed to the sub-report is the first value of a given column.

Depending on the placement of the Sub-Report, you can have a linked sub-report for each distinct value of the column. For example, in a *Summary Break* report, you can have a sub-report whose parameter is linked to the row break column. If this sub-report is then placed in the Group Header section, a sub-report is generated in the main report for each distinct value in the row break column (since the column breaks on each distinct value).

A linked sub-report will always take the first value in the column mapped to its parameter. However, depending on the placement of the sub-report, it can appear multiple times within the report, each time taking the first value of the column within its grouping.

Given below is an example of a sub-report placed in the group header that takes in a parameter from the main report.

```

QbReport.setEspressManagerUsed(false);

// Begin Code : Creating the root Report
DBInfo rootInfo = new DBInfo("jdbc:hsqldb:woodview",
    "org.hsqldb.jdbcDriver",
    "sa",
    "",
    "SELECT Customers.Company, Orders.OrderID, " +
    "Products.ProductName, Products.UnitPrice, " +
    "Order_Details.Quantity " +
    "FROM Order_Details, Products, Orders, Customers " +
    "WHERE (Orders.CustomerID = Customers.CustomerID) " +
    "AND (Order_Details.OrderID = Orders.OrderID) " +
    "AND (Products.ProductID = Order_Details.ProductID);");

ColInfo rootColInfo[] = new ColInfo[5];
rootColInfo[0] = new ColInfo(0);
rootColInfo[0].setRowBreak(true);
rootColInfo[1] = new ColInfo(1);
rootColInfo[1].setRowBreak(true);
rootColInfo[2] = new ColInfo(2);
rootColInfo[2].setAggregation(false, ColInfo.NONE);
rootColInfo[3] = new ColInfo(3);
rootColInfo[3].setAggregation(false, ColInfo.AVG);
rootColInfo[4] = new ColInfo(4);
rootColInfo[4].setAggregation(false, ColInfo.SUM);

QbReport rootReport = new QbReport(
    parent,
    QbReport.SUMMARY,
    rootInfo,
    rootColInfo,
    "ParameterizedSubReport.rpt");
// End Code : Creating the root Report

int numberOfTableHeaderCells =
    rootReport.getTable().getHeader().getData().length;

// Transferring Table Header cells to Row Break Zero Header
for (int i = 0; i < numberOfTableHeaderCells; i++)
{
    rootReport.getTable().getRowBreakHeader(0)
        .addData(rootReport.getTable().getHeader().getData(i));
    rootReport.getTable().getRowBreakHeader(0).getData(i).setY(1);
}

// Deleting Table Header cells
for (int i = 0; i < numberOfTableHeaderCells; i++) {
    rootReport.getTable().getHeader().removeData(0);
}

// Begin Code : Creating the SubReport
// SimpleQueryInParam(name of Parameter, String to be displayed,
// MapToColumn?, tableName, ColumnName, SQL Type, DefaultValue, value)
SimpleQueryInParam inParam = new SimpleQueryInParam(

```

```

"Company", "Please select", true, "Customers",
"Company", Types.VARCHAR, "All Unfinished Furniture",
"All Unfinished Furniture");
SimpleQueryInParam[] paramSet = { inParam };

SimpleQueryFileInfo subReportInfo = new SimpleQueryFileInfo(
    "jdbc:hsqldb:woodview",
    "org.hsqldb.jdbcDriver",
    "sa", "",
    "SELECT Customers.CustomerID, Customers.ContactName, " +
    "Customers.Address, Customers.City, " +
    "Customers.State, Customers.Zip " +
    "FROM Customers " +
    "WHERE (Customers.Company =:Company);");
subReportInfo.setInParam(paramSet);

// Begin Code : Setting up Column Mapping
ColumnInfo subReportColumnInfo[] = new ColumnInfo[6];
for (int i = 0; i < subReportColumnInfo.length; i++) {
    subReportColumnInfo[i] = new ColumnInfo(i);
}

// End Code : Setting up Column Mapping

try {

    // createSubReport(root report, report type, sub report data source, sub
    report column mapping, template)
    SubReportObject reportCell = SubReport.createSubReport(
        rootReport,
        QbReport.COLUMNAR,
        subReportInfo,
        subReportColumnInfo,
        null);
    ((QbReport) reportCell.getSubReport()).applyTemplate(
        "ParameterizedSubReportSub.rpt", true);

    // Get Parameter from root report column
    reportCell.setParameterMap(new String[]
    { rootReport.getTable().getColumn(0).getID() });

    // Set SubReport cell's height and width
    reportCell.setWidth(7.2);
    reportCell.setResizeToFitContent(true);
    reportCell.setY(.3);
    reportCell.setX(0);

    // Add SubReport to the Group Zero Header
    rootReport.getTable().getRowBreakHeader(0).addData(reportCell);

} catch (Exception ex) {
    ex.printStackTrace();
}
// End Code : Creating the sub report

return (new Viewer().getComponent(rootReport));

```

Exported Results [ <http://data.quadbase.com/Docs70/help/manual/code/export/ParameterizedSubReport.pdf> ]

You do not need to link a column from the main report to a sub-report's parameter if both the main report and the sub-report have a parameter with the same name.

## 8.B.9. Drill-Down

With the help of parameterized queries or classes, *Drill-Down* reports can be created. Instead of having reports with large amounts of data in it, you can show a top level report showing the minimum data required and then delve deeper on the selected data. For more information on *Drill-Down* reports, please refer to Section 4.1.8 - Drill-Down.

To create a *Drill-Down* report, you need to create the various `QbReport` objects (please note that all `QbReport` objects, other than the root report object, will have parameterized queries as their data source) and then specify the order of the drill-down as well as the column of the report to attach the next level of the *Drill-Down* report. To create a *Drill-Down* report use the following method:

```
rootReport.createDrillDownReport(java.lang.String name, int reportType,
    IDatabaseInfo dbInfo, ColInfo[] mapping, java.lang.String template, int[]
    columnMapping)
```

Unlike Sub-Reports, *Drill-Down* reports must be parameterized queries, so there are fewer construction methods for *Drill-Down* reports.

### 8.B.9.1. Creating the Report

The following example shows how to create a report that contains a *Drill-Down* report.

```
Component doDrillDownReport(Object parent) {
    QbReport.setEspressManagerUsed(false);

    // Begin Code : Creating Report 1 - the Root Report
    DBInfo rootInfo = new DBInfo("jdbc:hsqldb:woodview",
        "org.hsqldb.jdbcDriver",
        "sa",
        "",
        "select Categories.CategoryName, sum(Products.UnitsInStock) " +
        "from Categories, Products " +
        "where Categories.CategoryID = Products.CategoryID " +
        "group by Categories.CategoryName " +
        "order by Categories.CategoryName;"
    );

    ColInfo rootColInfo[] = new ColInfo[2];
    for (int i = 0; i < rootColInfo.length; i++) {
        rootColInfo[i] = new ColInfo(i);
    }

    rootColInfo[0].setName("Product Name");
    rootColInfo[1].setName("Total Units In Stock");
    QbReport rootReport = new QbReport(
        parent,
        QbReport.COLUMNAR,
        rootInfo,
        rootColInfo,
        "DrillDown.rpt");
    // End Code : Creating Report 1 - the Root Report
    // Begin Code : Creating Report 2 - the drill-down Report
    SimpleQueryInParam inParam = new SimpleQueryInParam("param", "Please
    select", true,
```

```

"Categories", "CategoryName", Types.VARCHAR, "Arm Chairs", null));

SimpleQueryInParameter[] paramSet = { inParam };
SimpleQueryFileInfo levelOneReportInfo = new SimpleQueryFileInfo(
    "jdbc:hsqldb:woodview",
    "org.hsqldb.jdbcDriver",
    "sa",
    "",
    "select Categories.CategoryName, Products.ProductName,
    Products.UnitPrice, Products.UnitsInStock from Categories,
    Products where Categories.CategoryID=Products.CategoryID and
    Categories.CategoryName=:param order by Categories.CategoryName,
    Products.ProductName;");
levelOneReportInfo.setInParameter(paramSet);
ColumnInfo levelOneReportColumnInfo[] = new ColumnInfo[4];
for (int i = 0; i < levelOneReportColumnInfo.length; i++) {
    levelOneReportColumnInfo[i] = new ColumnInfo(i);
}

levelOneReportColumnInfo[0].setName("Product Type");
levelOneReportColumnInfo[1].setName("Product Name");
levelOneReportColumnInfo[2].setName("Price");
levelOneReportColumnInfo[3].setName("Units In Stock");

try {
    // createDrillDownReport(Name, Type of Report, database information,
    // column mapping, template, column of root report to be mapped to)
    rootReport.createDrillDownReport("TestDrillDownReport",
        QbReport.COLUMNAR,
        levelOneReportInfo,
        levelOneReportColumnInfo,
        "lvl1.rpt", new int[] { 0 });
} catch (Exception ex) {
    ex.printStackTrace();
}

rootReport.getTable().getColumn(0).setDrillDownName("TestDrillDownReport");
// End Code : Creating Report 2 - the drill-down Report
return (new Viewer().getComponent(rootReport));

```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/DrillDown.zip> ]

Exported Results [ <http://data.quadbase.com/Docs70/help/manual/code/export/DrillDown.html> ]

When you generate *Drill-Down* reports without using the ERES Server, you MUST have a sub directory called DrillDown under the working directory of the .class file.

The above exported results, only contain the root report and the *Drill-Down* report for Oval Tables.

When you export *Drill-Down* reports, to either DHTML or PDF format, you must have the DrillDownReportServlet servlet provided. This servlet provides the link in the report to the next level. Depending on the link clicked, this servlet delivers the appropriate next level report. To use this servlet, you must make it accessible from your servlet runner/application server. The URL to access this servlet MUST be `http://<machine name>:<port number>/servlet/DrillDownReportServlet`. Then in your application code, you must use the following method:

```

QbReport.setDynamicExport(boolean state, String serverName, int
    servletRunnerPort);

```

---

For example, given a *Drill-Down* report and the following method call:

```
report.setDynamicExport(true, "Aphrodite", 8080);
```

When the report is exported, the above code sees to it that the links are pointing to machine *Aphrodite* and the port 8080. The `DrillDownReportServlet` servlet gets called using the URL, `http://Aphrodite:8080/servlet/DrillDownReportServlet` and thus computes the next level based on the link clicked and shows the next level.

You can also change the url for this servlet by using the `setServletDirectory(String)` method in the `QbReport` class. For instance, given the example above, if the following line of code were added:

```
report.setServletDirectory("ERES/Test/");
```

then the links will show `http://Aphrodite:8080/ERES/Test/DrillDownReportServlet` instead of the default `servlet/` in the link.

Note that *Drill-Down* reports exported to formats other than HTML, DHTML, or PDF will only show the current level.

## 8.C. Getting the Chart Data

The two appendices in this Chapter (Appendix 8.C - Getting the Chart Data and Appendix 8.D - Creating the Chart) contain information to help you build a chart from scratch (without using the designer). Keep in mind that this method is typically not recommended as it will make the chart more difficult to deploy and maintain. However, under certain circumstances, it may be necessary to construct charts in this manner.

The following is an example of a `QbChart` constructor. Like reports, there are numerous chart constructors available. The typical `QbChart` constructor requires at least four parameters. To create a new chart, you must specify the chart dimensions, chart type, the input data source information, and a mapping of the data columns to the respective columns of the chart.

```
QbChart(java.applet.Applet applet, int dimension, int chartType,  
        IDatabaseInfo dbinfo, IColumnMap cmap, java.lang.String template)
```

The template is not required (can be null) and the applet is only required if the chart is to be displayed in an applet, but the other four parameters must always contain meaningful information. This appendix takes an in depth look at the data source parameter and the methods used to connect to different data sources. Appendix 8.D - Creating the Chart discusses the dimensions, the chart type, the column mapping, and the actual creation of the chart. Appendix 8.D - Creating the Chart also contains working examples for you to try.

The data for a chart can be obtained from a report (if the chart is inserted into a report) or from an independent data source. The following sections deal with connecting to an independent data source in order to obtain data to generate a chart.

The data used to create a chart may be fetched from one of several different types of sources. These sources include:

- Fetch the data from a local or remote database using a JDBC driver;
- Read the data from a plain text file, which contains database records in plain text (ASCII) format. Files of this format can be generated by most database programs;
- Read the data from a spreadsheet model;
- Pass the dataset as an array in memory;
- Pass your own data through API;
- Fetch the data from an EJB data source;
- Merge from multiple data sources;

In the remainder of this section, we discuss the above methods of data extraction in greater detail.

## 8.C.1. Data from a Database

One of the powerful features of Chart API is its ability to fetch data from a database directly through JDBC. With this approach, all you need to do is to specify the information necessary to connect to the database and the precise form of the SQL statement. Thus your program can connect to virtually any database provided that a JDBC driver is available.

The database and query information must be stored in a `DBInfo` object prior to constructing the chart. Use the following code to instantiate the `DBInfo` object.

```
DBInfo dbinfo = new DBInfo(
    "jdbc:odbc:ODBCDatabase",           // URL
    "sun.jdbc.odbc.JdbcOdbcDriver",    // JDBC driver
    "myName",                          // username
    "myPassword",                      // password
    "select * from sales");           // SQL
```

Since `DBInfo` implements the interface `IDatabaseInfo`, you can use the `DBInfo` object in the following `QbChart` constructor:

```
public QbChart(Applet parent, int dimension, int chartType, IDatabaseInfo
    dbinfo, ColInfo colMap, String templateFile);
```

In some cases, you may not wish to pass the entire database information (such as `userid`, `password`, `location`, `driver`, and the query) to ERES. You may want to make the connection yourself and just provide the result set of the query directly to the API. This can be done by creating a `QueryResultSet` object from the `ResultSet` object you have generated and passing that `QueryResultSet` object as the data source, instead of a `DBInfo` object.

```
// Create a QueryResult object from a java.sql.ResultSet object resultSet
QueryResultSet queryResultSet = new QueryResultSet(resultSet);

// use queryResultSet in place of IResultSet data in the following
// constructor
QbChart(java.applet.Applet applet, int dimension, int chartType, IResultSet
    data, IColumnMap cmap, java.lang.String template)
```

In the above example, an instance of class `QueryResultSet` is created from the `ResultSet` object passed to the API and this instance is passed to the `QbChart` constructor to create the chart object. Please note that in this scenario, the chart is not refreshable. To update the chart, you will need to make the connection to the database again and pass the result set object to the chart and refresh it yourself.

### 8.C.1.1. JNDI

ERES also allows data to be obtained from a JNDI source. JNDI data sources are treated like database data sources and support the same functionalities. Using a JNDI data source makes it easier to migrate charts between different environments. If the data sources in both environments are setup with the same lookup name, charts can be migrated without any changes.

To connect to a JNDI data source, you must have a data source deployed in your application server. You must also provide the `INITIAL_CONTEXT_FACTORY` and `PROVIDER_URL` to successfully make the connection. Please note that when connecting to a JNDI data source deployed in Tomcat, the `INITIAL_CONTEXT_FACTORY` and `PROVIDER_URL` need not be provided (although ERES Server must be running as a servlet under the Tomcat environment).

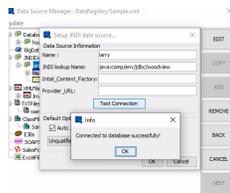
```
// create a DBInfo object with JNDI lookup name and query
String JNDIName = "java:comp/env/jdbc/TestDB";
String query = "select * from testdata";
```

```
// The environment hashtable is empty for tomcat because ERES Server is
// running inside Tomcat context. If other application server is used,
// need to set INITIAL_CONTEXT_FACTORY and PROVIDER_URL.
Hashtable env = new Hashtable();
DBInfo dbInfo = new DBInfo(JNDIName, query, env);
```

In the above example, an instance of class DBInfo provides the information that the program needs to connect to a JNDI data source and retrieve data. To construct the chart, use the following constructor containing IDatabaseInfo:

```
public QbChart(Applet parent, int dimension, int chartType, IDatabaseInfo
    dbinfo, ColInfo colMap, String templateFile);
```

### 8.C.1.2. JNDI example using Tomcat and EspressoManager running from servlet



Woodview database connection

### 8.C.2. Data from a Data file (TXT/DAT/XML)

Data for generating a chart can also be imported from a data file, which can be a text file as well as an XML formatted file. A chart data file (with a .dat extension) is a plain text file where each line represents one record, except the first two lines, which contain the data types and field names, respectively. Here is an example of a data file, which contains four records and where each record has three fields. The first line specifies the data type of each field, and the second line specifies the field name.

```
string, date, decimal
Name, Day, Volume
"John", "1997-10-3", 32.3
"John", "1997-4-3", 20.2
"Mary", "1997-9-3", 10.2
"Mary", "1997-10-04", 18.6
```

The use of the comma character is optional, but most database programs can output a file in this format (except for the first two lines) when exporting records from a table in text format. As a result, even if you do not have a JDBC driver for your database, you can still quickly produce charts. You can simply export the data from your database in text format and then your program, using the ERES API, can read it.

For XML format, the specification is as follows:

```
<EspressoData>
    <DataType>string</DataType>
    <DataType>date</DataType>
    <DataType>decimal</DataType>
    <FieldName>Name</FieldName>
    <FieldName>Day</FieldName>
    <FieldName>Volume</FieldName>
</Row>
```

```

        <Data>John</Data>
        <Data>1997-10-3</Data>
        <Data>32.3</Data> </Row>
    </Row>

    <Data>John</Data>
    <Data>1997-4-3</Data>
    <Data>20.2</Data>
</Row>
<Row>
    <Data>Mary</Data>
    <Data>1997-9-3</Data>
    <Data>10.2</Data>
</Row>
<Row>
    <Data>Mary</Data>
    <Data>1997-10-4</Data>
    <Data>18.6</Data>
</Row>
</EspressData>

```

For more details about XML Data, please refer to the Section 3.1.4 - Data from XML and XBRL Files.

Specifying the text file to use is very straight forward. Use the following constructor and replace the variable filename with the file path (relative or full path) of the data file. If you are connecting to the server, relative paths should be in respect to the ERES root directory. Otherwise, the path will be relative to the current working directory. Also replace the **fileType** parameter with one of the following options: **QbChart.DATAFILE**, **QbChart.QUERYFILE**, or **QbChart.XMLFILE**

```

public QbChart(Applet parent, int dimension, int chartType, int fileType,
    String filename, boolean doTransposeData, ColInfo colMap, String
    templateFile);

```

The same constructor can also be used for passing in XML data generated by a servlet. You can specify the file-type as **QbChart.XMLFILE** and the filename as the URL to your servlet (e.g. **http://localhost:8080/servlet/XMLDataGenerator**).

### 8.C.3. Data from a XML Data Source

In addition to the above, ERES allows you to retrieve data and query XML files. XML data can be in virtually any format, but you need to specify a DTD file or an XML schema along with the XML data. The following code demonstrates how to set up an XML query:

```

// Set up the XML Data Source
String xmlfilename = "Inventory.xml";
String xmlcondition = "/Inventory/Category/Product/ProductID < 45";

XMLFieldInfo[] fields = new XMLFieldInfo[5];
fields[0] = new XMLFieldInfo(new String[]
    {"Inventory", "Category", "Product", "ProductID"});
fields[0].setAttributeDataType(DTDDataType.INT);

fields[1] = new XMLFieldInfo(new String[]
    {"Inventory", "Category", "Product", "ProductName"});

fields[2] = new XMLFieldInfo(new String[]
    {"Inventory", "Category", "Product", "UnitPrice"});
fields[2].setElementDataType(DTDDataType.DOUBLE);

```

```

fields[3] = new XMLFieldInfo(new String[]
    {"Inventory", "Category", "Product", "UnitsInStock"});
fields[3].setElementDataType(DTDDatatype.INT);

fields[4] = new XMLFieldInfo(new String[]
    {"Inventory", "Category", "Product", "ShipDate"});
fields[4].setElementDataType(DTDDatatype.DATE);
fields[4].setDateFormat(XMLDataTypeUtil.YYYY_MM_DD);

XMLFileQueryInfo xmlInfo = new XMLFileQueryInfo(xmlfilename, fields,
    xmlcondition, fields);

```

The `XMLFieldInfo` instance is created using one of two constructors. The first constructor, used to select xml fields, contains one parameter. For this constructor, you need to pass in a String array that specifies each xml tag in the hierarchy leading to the target field. In the above example, fields[1-4] are created using the first constructor. The second constructor, used to select xml attributes, contains two parameters. In addition to the String array parameter, the second constructor also requires another string for the attribute name. In the above example, field[0] is created using this constructor because “ProductID” is an attribute of “Product”.

You may have also noticed that for any non-String field, you must explicitly set the data type. Once you have created the `XMLFileQueryInfo` instance, you can use the following constructor to create the `QbChart`.

```

public QbChart(Applet applet, int dimension, int chartType, XMLFileQueryInfo
    xmlInfo, boolean doTranspose, int[] transposeColumns, IColumnMap colMap,
    String templateFile);

```

You can also pass in an XML stream instead of an XML file, when using XML data as a data source. To pass in an XML stream, you would pass in the byte array containing the XML data instead of the XML data file name.

In the above example, you can pass in a XML stream via a byte array (for example, a byte array called `xmlByteArray`) in the `XMLFileQueryInfo` [ <http://data.quadbase.com/Docs70/eres/help/apidocs/quadbase/common/util/internal/XMLFileQueryInfo.html> ] constructor:

```

XMLFileQueryInfo xmlInfo = new XMLFileQueryInfo(xmlByteArray, fields,
    xmlCondition, fields);

```

## 8.C.4. Data Passed in an Array in Memory

The API allows input data to be passed directly in memory, as an array. This is made possible by the interface `IResultSet` [ <http://data.quadbase.com/Docs70/eres/help/apidocs/quadbase/util/IResultSet.html> ] (defined in `quadbase.util` [ <http://data.quadbase.com/Docs70/eres/help/apidocs/quadbase/util/package-summary.html> ] package). This interface is used to read data in the tabular form, and is quite similar to the `java.sql.ResultSet` interface used for JDBC result sets (but is much simpler). Users can provide their own implementations of `IResultSet` (which is discussed in the next section), or use one provided by ERES. The simplest implementation is provided by the class `DbData` (Other classes that provide an `IResultSet` implementation are `QueryResultSet` [ <http://data.quadbase.com/Docs70/eres/help/apidocs/quadbase/util/QueryResultSet.html> ] and `StreamResultSet` [ <http://data.quadbase.com/Docs70/eres/help/apidocs/quadbase/util/StreamResultSet.html> ]). If you can fit all the data you need for the chart in memory, you can simply pass the array as an argument in `DbData` with one line of code. There are three constructors for `DbData`:

- `DbData(java.lang.String s)` - Construct `DbData` by parsing the data value argument from an HTML page
- `DbData(java.lang.String[] fieldName, java.lang.Object[][] records)` - Construct a new `DbData` class
- `DbData(java.lang.String[] dataType, java.lang.String[] fieldName, java.lang.String[][] records)` - Construct a new `DbData` class

We will use the following constructor in the example here.

```
public DbData(String dataType[], String fieldName[], String records[][]);
```

This is a similar construction to reading in data from a data file. Here, the first argument presents the data types (the first line in the data file) and the second argument presents the field names (the second line). The third argument, records[], provides an array of records, records[i] being the ith record. The following shows how it works.

```
String dataType[] = {"varchar", "decimal"};
String fieldName[] = {"People", "Sales"};
String records[][] = {{"Peter", "93"}, {"Peter", "124"},
                      {"John", "110"}, {"John", "130"},
                      {"Mary", "103"}, {"Mary", "129"}};
```

```
DbData data = new DbData(dataType, fieldName, records);
```

To create the chart, use the following QbChart constructor:

```
public QbChart(Applet parent, int dimension, int chartType, IResultSet data,
               ColInfo colMap, String templateFile);
```

## 8.C.5. Data Passed in your Custom Implementation

For maximum flexibility, you can retrieve and prepare the dataset in any way you want and pass it to the charting engine. To pass in your class file as the data source, your class file must implement the IDataSource [ <http://data.quadbase.com/Docs70/eres/help/apidocs/quadbase/util/IDataSource.html> ] interface. Given below is an example of code that implements IDataSource and passes in a class file as the data source:

```
public class CustomClassData extends Applet implements IDataSource {

    // Setting DbData for passing data as arguments
    String dataType[] = {"string", "String", "double"};
    String fieldName[] = {"Destination", "Time", "Price"};
    String records[][] = {{"Mayfair", "13:43", "3.50"},
                          {"Bond Street", "13:37", "3.75"},
                          {"RickmansWorth", "13:12", "5.25"},
                          {"Picadilly", "13:24", "3.00"}};
    DbData data = new DbData(dataType, fieldName, records);

    public IResultSet getResultSet()
    {
        return data; }

}
```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/CustomClassData.zip> ]

The example above creates data (DbData instance) and stores it in memory. When the getResultSet() method is called, it returns the DbData object which implements the IResultSet [ <http://data.quadbase.com/Docs70/eres/help/apidocs/quadbase/util/IResultSet.html> ] interface. Keep in mind that it is not necessary to create your data in this manner. As long as you are able to return an object that implements IResultSet, you can obtain the data from any data source. Use the following constructor to create your chart:

```
public QbChart(Applet parent, int dimension, int chartType, int fileType,
               String filename, ColInfo colMap, String templateFile);
```

For custom class files, set the fileType to **QbChart.CLASSFILE** and the filename to the name of the classfile.

Please note that if you are passing in your own class file as the data source and you are using the ERES Server, the class file must be accessible from the CLASSPATH of the ERES Server.

You can also pass in a parameterized class file as the data source for the chart. The parameter is obtained at run-time from the user and the data is then fetched and used to generate the chart. Note that this will only work for a stand-alone chart configuration.

```
public class ParameterizedClassFile implements IParameterizedDataSource {
    public IQueryInParam[] getParameters()
    {
        SimpleQueryInParam[] params = new SimpleQueryInParam[1];
        params[0] = new SimpleQueryInParam("param2", "Enter the price:", false,
            null, null, Types.INTEGER, new Integer(2), null);
        return params;
    }

    public IResultSet getResultSet(IQueryInParam[] params) {
        double price = 3.5;
        if ((params != null) && (params.length >= 1))
        {
            Object obj = params[0].getValue();
            if ((obj != null) && (obj instanceof Integer)) price =
                ((Integer)obj).intValue();
        }
        String dataType[] = {"string", "String", "double"};
        String fieldName[] = {"Destination", "Time", "Price"};
        String records[][] = {"Mayfair", "13:43", price+""},
            {"Bond Street", "13:37", price+""},
            {"Rickmansworth", "13:12", price+""},
            {"Picadilly", "13:24", price+""};
        return new DbData(dataType, fieldName, records);
    }
}
```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/ParameterizedClassFile.zip> ]

When using a parameterized class file as the data source, the parameter dialog box is usually a text box with no choices available. However, you can specify what the selection choices can be (available via a drop-down box) by implementing the IQueryParamValuesProvider [ <http://data.quadbase.com/Docs70/eres/help/apidocs/quadbase/util/IQueryParamValuesProvider.html> ] interface.

```
public class CustomParamClassFile implements IParameterizedDataSource {

    public IQueryInParam[] getParameters() {
        mySimpleQueryMultiValueInParam[] params = new
        mySimpleQueryMultiValueInParam[1];
        params[0] = new mySimpleQueryMultiValueInParam("region", "Select
        Region(s):", true, "Customers", "Region", Types.VARCHAR, "East", null);
        return params;
    }

    public IResultSet getResultSet(IQueryInParam[] params) {
        QueryResultSet data = null;
        ResultSet rs = null;

        String paramValue = "'East'";
        if ((params != null) && (params.length >= 1)) {
            Vector selectedValues = null;
            if (params[0] instanceof IQueryMultiValueInParam)
                selectedValues = ((IQueryMultiValueInParam)params[0]).getValues();
        }
    }
}
```

```

    for (int i = 0; i < selectedValues.size(); i++) {
        if ((selectedValues.get(i) != null) && (selectedValues.get(i) instanceof
String)) {
            if (i == 0) paramValue = "'" + (String)selectedValues.get(i) + "'";
            else paramValue += "','" + (String)selectedValues.get(i) + "'";
        }
    }
}
String myQuery = "select cu.region, c.categoryname, count(o.orderid),
sum(od.quantity), sum(p.unitprice * od.quantity) from customers cu,
categories c, products p, orders o, order_details od where cu.customerid =
o.customerid and c.categoryid = p.categoryid and p.productid = od.productid
and o.orderid = od.orderid and cu.region in (" + paramValue + ") group by
cu.region, c.categoryname";

try {
    Class.forName("org.hsqldb.jdbcDriver");

    String url = "jdbc:hsqldb:help/examples/DataSources/database/woodview";
    Connection conn = DriverManager.getConnection(url, "sa", "");
    Statement stmt = conn.createStatement();

    rs = stmt.executeQuery(myQuery);
    data = new QueryResultSet(rs);
    // conn.close();
} catch (Exception ex) {
    ex.printStackTrace();
}
return data;
}

public class mySimpleQueryMultiValueInParam extends
SimpleQueryMultiValueInParam implements IQueryParamValuesProvider {

    public String paramName, promptName, tableName, colName;
    boolean mapToColumn;
    int sqlType;
    Object defaultValue;
    Vector values;
    public mySimpleQueryMultiValueInParam(String paramName, String
promptName, boolean mapToColumn, String tableName, String colName, int
sqlType, Object defaultValue, Vector values) {
        super(paramName, promptName, mapToColumn, tableName, colName, sqlType,
defaultValue, values);
    }

    public Vector getSelectionChoices() {
        System.out.println("getSelectionChoices called");
        try {
            Class.forName("org.hsqldb.jdbcDriver");

            String url = "jdbc:hsqldb:help/examples/DataSources/database/woodview";
            Connection conn = DriverManager.getConnection(url, "sa", "");
            Statement stmt = conn.createStatement();
            String query = "SELECT DISTINCT " + getColumnName() + " FROM " +
getTable();
            ResultSet rs = stmt.executeQuery(query);
            Vector v = new Vector();

```

```

while (rs.next()) {
    switch (getSqlType()) {
        case Types.INTEGER:

            v.add(new Integer(rs.getInt(1)));
            break;

        case Types.VARCHAR:

            v.add(rs.getString(1));
            break;
    }
}
stmt.close();
conn.close();

return v;
} catch (Exception ex) {
    ex.printStackTrace();
}
return null;
}
}
}

```

Full Source Code [ <http://data.quadbases.com/Docs70/help/manual/code/src/CustomParamClassFile.zip> ]

## 8.C.6. Data from a Spreadsheet Model

A chart can function as a view to a spreadsheet model in a Model-View-Controller (MVC) architecture. It automatically reads the spreadsheet data and plots itself. The chart registers itself as a listener to the spreadsheet model and updates itself when notified of any changes to the spreadsheet data. A spreadsheet (Java) object is provided by the user, which is an instance of a class that implements the `ISpreadSheetModel` [ <http://data.quadbases.com/Docs70/eres/help/apidocs/quadbases/util/ISpreadSheetModel.html> ] interface. The event class `SpreadSheetModelEvent` is used by the model to notify its listeners of any changes to data. The following example shows how a spreadsheet model can be used for a chart. It uses the utility class `SimpleSpreadSheet` [ <http://data.quadbases.com/Docs70/eres/help/apidocs/quadbases/util/SimpleSpreadSheet.html> ] (defined in the `quadbases.util` [ <http://data.quadbases.com/Docs70/eres/help/apidocs/quadbases/util/package-summary.html> ] package), which implements the `ISpreadSheetModel` interface.



### Note

This method is applicable only to live Java program spreadsheet objects that contain data to be plotted, and hence complements the methods for reading data from a database or data file in spreadsheet format.

```

String[] columnVals = {"quantity", "high"};
String[] rowVals = {"coffee", "Soft Drinks", "Fruit
    Juice", "Water", "beer"};
Double[][] vals = { {new Double(1), new Double(30)},
                    {new Double(3), new Double(33)},
                    {new Double(7), new Double(34)},
                    {new Double(8), new Double(40)},
                    {new Double(8), new Double(40)} };

```

```
// Please see quadbase.util.SimpleSpreadSheet
sss = new SimpleSpreadSheet(rowVals, columnVals, vals);
```

Here the data is given in a spreadsheet format and looks like the table below:

	quantity	high
coffee	1	30
Soft Drinks	3	33
Fruit Juice	7	34
Water	8	40
beer	8	40

*Data in Spreadsheet Format*

ERES then transposes the data (this is done internally) so that the data is changed to the following:

coffee	quantity	1
coffee	high	30
Soft Drinks	quantity	3
Soft Drinks	high	33
Fruit Juice	quantity	7
Fruit Juice	high	24
Water	quantity	8
Water	high	40
beer	quantity	8
beer	high	40

*Data After Transpose*

The data mapping for the chart is then done with respect to the transposed data.

Use the following constructor to create the QbChart object:

```
public QbChart(Applet applet, int dimension, int chartType,
    ISpreadSheetModel spreadsheet, IColumnMap colMap, String templateFile);
```

## 8.C.7. Data from Enterprise Java Beans (EJBs)

Data can be passed from an EJB data source to the chart by allowing users to query data directly from an entity bean. To add an EJB as a data source, the EJB must first be deployed in the application server and the client JAR file containing the appropriate stub classes must be added to your classpath (or the `-classpath` argument of the ERES Server batch file when using the API in conjunction with ERES Server).

Unlike reports, it is not necessary to create an `EJBInfo` object to store the connection information. Provide the names directly into the following constructor to create a `QbChart` object.

```
public QbChart(Applet applet,
    int dimension,
    int chartType,
    String jndiName,
    String homeName,
    String remoteName,
    String selectedMethodName,
    Object[] selectedMethodParamVal,
```

```

        IColumnMap cmap,
        String template);

```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/DataFromEJB.zip> ]

The above code can be run both as an application and as a JNLP (More about Applets in JNLP: Section 1.3.4 - Run Applets As JNLP).

The constructor for this class is:

```

public QbChart(Applet applet, int dimension, int chartType, String jndiName,
    String homeName, String remoteName, String selectedMethodName, Object[]
    selectedMethodParamVal, IColumnMap cmap, String template);

```

## 8.C.8. Data from multiple Data Sources

ERES also provides a functionality to merge multiple data sources together. You can create a `DataSheet` object from any combination of data sources, for example, data file, database or `IResultSet` (see “Data Passed as an Argument” section). You can use a `QbChart` constructor to create a chart object from an array of `DataSheet`.

The following example program fragment demonstrates how to combine the data from the examples in the above sections.

```

// Declaration of DataSheet object to be used to merge data
DataSheet dataSheet[] = new DataSheet[3]; // Declaration For Database (Data
    From a Database section)
DBInfo dbinfo = new DBInfo("jdbc:quadbase:/machine/schema",
    "quadbase.jdbc.QbDriver", "myName", "myPassword", "select * from sales"); //
    Declaration For Data Passed as an Argument (Data Passed as an Argument
    section)
String dataType[] = {"varchar", "decimal"};
String fieldName[] = {"People", "Sales"};
String records[][] = {{"Peter", "93"}, {"Peter", "124"},
    {"John", "110"}, {"John", "130"},
    {"Mary", "103"}, {"Mary", "129"}};
DbData data = new DbData(dataType, fieldName, records); // Create DataSheet
    from data file (Data From a Text File section)
// DataSheet(Applet applet, String dataFile)
dataSheet[0] = new DataSheet(this, "help/examples/data/Columnar1.dat"); //
    Create DataSheet from database (Data From a Database section)
// DataSheet(Applet applet, IDatabaseInfo dbInfo)
dataSheet[1] = new DataSheet(this, dbinfo); // Create DataSheet from
    IResultSet (Data Passed as an Argument section)
// DataSheet(Applet applet, IResultSet data)
dataSheet[2] = new DataSheet(this, data); // create QbChart
QbChart chart = new QbChart
    (this, // applet
    QbChart.VIEW2D, // Two-Dimensional
    QbChart.PIE // Pie Chart
    dataSheet, // DataSheet
    colInfo, // column information
    null); // No template

```



### Note

After you have created a `DataSheet` object, you can modify it (add/delete/update row values) by using `DataSheet` API. Data is not refreshable if merging data from `IResultSet`.

## 8.C.9. Data in Spreadsheet Format

Data obtained using the above methods may also be interpreted as a spreadsheet. A spreadsheet has a grid structure, much like a table. The leftmost column and first row in a spreadsheet contain labels (or headings). Each cell represents a distinct data point comprising its row label, column label, and the cell value (in contrast to the normal tabular notation, where each row represents a distinct data point).

Consider the following example:

Date	Nasdaq	Dow	SP500
"12/04/2000"	2304	10503	1240
"12/05/2000"	2344	10486	1239
"12/06/2000"	2344	10458	1224
-----	-----	-----	-----

Suppose the data in your database is arranged as four columns as shown. You can plot the three indices (Nasdaq, Dow, and SP500) as three lines with Date as the category axis if you treat the data as in spreadsheet format.

Several `QbChart` constructors are provided to deal with this issue. The three main ones are listed below.

```
QbChart(java.applet.Applet applet, int dimension, int chartType, IResultSet
data, boolean
```

```
doTransposeData, IColumnMap cmap, java.lang.String template);
```

```
QbChart(java.applet.Applet applet, int dimension, int chartType, int
fileType,
```

```
java.lang.String filename, boolean doTransposeData, IColumnMap cmap,
java.lang.String
template);
```

```
QbChart(java.applet.Applet applet, int dimension, int chartType,
IDatabaseInfo dbinfo, boolean
```

```
doTransposeData, IColumnMap cmap, java.lang.String template);
```

To specify that data is in spreadsheet format, you need to set the `doTransposeData` flag to true.

The above constructors transpose all the columns from the second column to the last column into a 3-column table. Thus, the data type from the second column onwards must be numeric otherwise the transpose will not be successful.

ERES also allows transposing of selective columns. The columns to be transposed must share the same data type. After transposing, the original columns are removed and the new columns inserted at the end of the table data. Selective transposing can be done only once; you cannot transpose certain numeric columns and then try to transpose other numeric or string columns again.

As with the complete transposing, `QbChart` has several constructors to allow selective transposing. The three main ones are listed below:

```
QbChart(java.applet.Applet applet, int dimension, int chartType, IResultSet
data, boolean
```

```
doTransposeData, int[] transposeCol, IColumnMap cmap, java.lang.String
template);
```

```

QbChart(java.applet.Applet applet, int dimension, int chartType, int
    fileType,

        java.lang.String filename, boolean doTransposeData, int[]
    transposeCol, IColumnMap cmap,
        java.lang.String template);

QbChart(java.applet.Applet applet, int dimension, int chartType,
    IDatabaseInfo dbinfo, boolean

        doTransposeData, int[] transposeCol, IColumnMap cmap, java.lang.String
    template);

```

To specify the columns to be transposed, you need to pass in an array containing the column indices for `transposeCol` and set the `doTransposeData` flag to true.

## 8.C.10. Transposing Data

ERES allows data to be obtained from various data sources. You can also “transpose” the data (i.e., transform the data so that the column names become part of the data) before passing the data to the desired chart type.

When the data is transposed, the original data columns (used in the transpose) are removed from the dataset and two new columns are added at the end. These two new columns would contain the transposed column names as well as the values of the original columns. For example, if the original data set has five columns and three are selected for transpose, the new data set would have five minus three (the number of transposed columns) plus two (the new columns added) or four columns.

When transposing data columns they are two things to note:

- The data columns must all have the same datatype;
- The column index passed to the chart's column mapping will refer to the new dataset (and not the original data set).

The sections below describe the various ways the data can be transposed:

### 8.C.10.1. Non-Selective Transposing

In this scenario, all the columns, except for the very first column (Column 0) is transposed. For example, given the data below:

Product	January	February	March
Chairs	\$3872.35	\$3962.21	\$4218.57
Tables	\$6534.98	\$6018.43	\$5928.71

the transposed data will appear as follows:

Product	ColumnLabel	Value
Chairs	January	\$3872.35
Chairs	February	\$3962.21
Chairs	March	\$4218.57
Tables	January	\$4218.57
Tables	February	\$6018.43
Tables	March	\$5928.71

To non-selectively transpose the data using the API, you use any `QbChart` constructor that has a `doTransposeData` boolean parameter, such as the following constructor:

```
QbChart(java.applet.Applet applet, int dimension, int chartType, int
  fileType, java.lang.String filename, boolean doTransposeData, IColumnMap
  cmap, java.lang.String template)
```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/NonTranspose.zip> ]

Exported Results [ <http://data.quadbase.com/Docs70/help/manual/code/export/NonTranspose.png> ]

Please note that the column index passed in `IColumnMap` would refer to the new dataset.

### 8.C.10.2. Selective Transposing

In this scenario, you choose which columns are to be transposed. You can only transpose those columns that share the same data type. With selective transposing, you can choose the very first column to be transposed as well. For example, given the data below:

Category	Product	January	January	March
Chairs	Side Chairs	\$3872.35	\$3962.21	\$4218.57
Chairs	Arm Chairs	\$2654.84	\$1924.83	\$2543.24
Tables	Round Tables	\$6534.98	\$6018.43	\$5928.71
Tables	Rectangu- lar Tables	\$10227.32	\$9721.83	\$11748.93

After transposing the numeric columns, the transposed data will appear as follows:

Category	Product	ColumnLabel	Value
Chairs	Side Chairs	January	\$3872.35
Chairs	Side Chairs	February	\$3962.21
Chairs	Side Chairs	March	\$4218.57
Chairs	Arm Chairs	January	\$2654.84
Chairs	Arm Chairs	February	\$1924.83
Chairs	Arm Chairs	March	\$2543.24
Tables	Round Tables	January	\$6534.98
Tables	Round Tables	February	\$6018.43
Tables	Round Tables	March	\$5928.71
Tables	Rectangu- lar Tables	January	\$10227.32
Tables	Rectangu- lar Tables	February	\$9721.83
Tables	Rectangu- lar Tables	March	\$11748.93

To selectively transpose the data using the API, you use any `QbChart` constructor that has a `doTransposeData` boolean parameter and integer array that takes the indices of the columns to be transposed, such as the following constructor:

```
QbChart(java.applet.Applet applet, int dimension, int chartType, int
  fileType, java.lang.String filename, boolean doTransposeData, int[]
  transposeCol, IColumnMap cmap, java.lang.String template)
```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/Transpose.zip> ]

Exported Results [ <http://data.quadbase.com/Docs70/help/manual/code/export/Transpose.png> ]

Please note that the column index passed in `IColumnMap` would refer to the new dataset.

## 8.D. Creating the Chart

To create a new chart, you must specify the chart type, the dimension of the chart, the input data source information, and the column mapping for the chart template. In this appendix, we look at the various chart types and the methods used to map the columns. There are also a number of fully functional examples in this appendix. Note that unless otherwise noted, all examples use the Woodview HSQL database, which is located in the `<ERESInstall>/help/examples/DataSources/database` directory. In order to run the examples, you'll need to add database HSQL JDBC driver (`hsqldb.jar`) to your classpath. The driver is located in the `<ERESInstall>/WEB-INF/lib` directory.

Charts can be either two or three-dimensional. Given below are the dimensions and the constant to select that dimension:

**Two-Dimensional**      `QbChart.VIEW2D`

**Three-Dimensional**    `QbChart.VIEW3D`

Charts have one of the following chart types. Given below are the different dimensions and the constants to select them:

<b>Column Chart</b>	<code>QbChart.COL</code>
<b>Bar Chart</b>	<code>QbChart.BAR</code>
<b>Line Chart</b>	<code>QbChart.LINE</code>
<b>Area Chart</b>	<code>QbChart.AREA</code>
<b>Pie Chart</b>	<code>QbChart.PIE</code>
<b>XY(Z) Scatter Chart</b>	<code>QbChart.SCATTER</code>
<b>Stack Column Chart</b>	<code>QbChart.STACKCOL</code>
<b>Stack Bar Chart</b>	<code>QbChart.STACKBAR</code>
<b>Stack Area Chart</b>	<code>QbChart.STACKAREA</code>
<b>High Low Chart</b>	<code>QbChart.HILOW</code>
<b>HLCO Chart</b>	<code>QbChart.HLCO</code>
<b>Percentage Column Chart</b>	<code>QbChart.PERCENTCOL</code>
<b>Surface Chart</b>	<code>QbChart.SURFACE(Three-Dimensional Only)</code>
<b>Bubble Chart</b>	<code>QbChart.BUBBLE(Two-Dimensional Only)</code>
<b>Overlay Chart</b>	<code>QbChart.OVERLAY(Two-Dimensional Only)</code>
<b>Box Chart</b>	<code>QbChart.BOX(Two-Dimensional Only)</code>
<b>Radar Chart</b>	<code>QbChart.RADAR(Two-Dimensional Only)</code>
<b>Dial Chart</b>	<code>QbChart.DIAL(Two-Dimensional Only)</code>

**Gantt Chart** `QbChart . GANTT(Two-Dimensional Only)`

**Polar Chart** `QbChart . POLAR(Two-Dimensional Only)`

For more information on the different chart types, please refer to the Section 4.2.3 - Chart Types and Data Mapping.

Each chart type requires two (or more) data columns to be mapped in order to create a chart object. For more detail on the mapping (and for a definition of the terms used here), please refer to the Section 4.2.3 - Chart Types and Data Mapping.

To define the Column Mapping for the chart template, the `ColInfo` class (located in the `quadbase.report-designer.util` package) is used.

The columns (from the data source) are numbered from left to right, starting with Column "0". The column positions, passed to the `ColInfo` object, are based on the data table. Note that the parameters for `ColInfo` objects are "-1" by default. A negative column position indicates that the particular parameter is not being used.



### Caution

As you may know, creating objects in java is a resource intensive operation. It is always recommended that you do not create too many objects. One way to conserve resource is to reuse `QbChart` objects whenever possible. For example, if a lot of your users request for a simple Columnar Chart in your web site, instead of creating a new `QbChart` object when each such request is received, you can have one (or a limited number of) such `QbChart` object(s) created and reuse the object(s) by simply modifying the data and attributes of the chart for each particular request.

## 8.D.1. Column, Bar, Line, Area, Pie and Overlay Charts

Column/Bar/Line/Area/Pie/Overlay charts need a *category* and a *value* axis. Those are the minimum requirements for constructing such types of charts. You can also add in a *series* and a *subvalue* if needed. The *subvalue* refers to the secondary axis i.e., it contains the column that gets mapped to the secondary axis.

### 8.D.1.1. Column Mapping

For instance, given the data shown below:

Order #	Product	Units Ordered	Units Shipped
12	Chair	15	12
12	Table	34	26
14	Chair	8	8
14	Table	23	14

*Original Data*

The following code sets the category to be Column 1 (Product), the series to be Column 0 (Order #), the value to be Column 3 (Units Shipped) and the subvalue to be Column 2 (Units Ordered):

```
ColInfo colInfo = new ColInfo();
colInfo.category = 1;
colInfo.series = 0;
colInfo.value = 3;
colInfo.subvalue = 2;
```

### 8.D.1.2. Creating the Chart

Constructing a Column chart is relatively straight forward. We have already discussed how to set the `ColInfo` array and how to obtain the data in previous sections. The following code demonstrates how to create the aforementioned Column chart.

```

Component doDataFromArguments(Applet applet) {

    // Do not connect to EspressoManager
    QbChart.setEspressoManagerUsed(false);

    // Column Mapping
    ColInfo colInfo = new ColInfo();
    colInfo.category = 1;
    colInfo.series = 0;
    colInfo.value = 3;
    colInfo.subvalue = 2;

    String dataType[] = {"integer", "varchar", "decimal", "decimal"};
    String fieldName[] = {"Order #", "Product", "Units Ordered", "Units
Shipped"};
    String records[][] = {{ "12", "Chair", "15", "12"},
{"12", "Table", "34", "26"},
{"14", "Chair", "8", "8"}, {"14",
"Table", "23", "14"} };
    DbData data = new DbData(dataType, fieldName, records);
    QbChart chart = new QbChart
        (applet, // Applet
        QbChart.VIEW2D, // Two-Dimensional
        QbChart.COL, // Column Chart
        data, // Data
        colInfo, // Column information
        null); // No specified

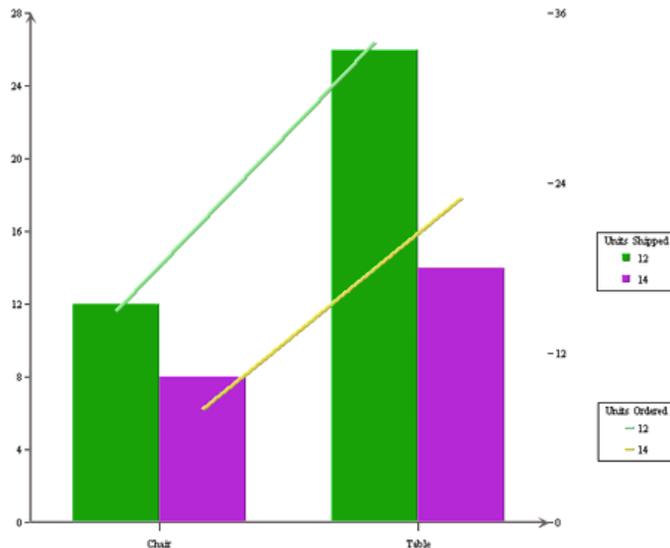
    template
    return chart;

}

```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/ColumnChart.zip> ]

When the above code is run as a Java Web Start Application, it will produce a two-dimensional Column chart shown below:



Generated Column Chart

Note that the chart created is a default chart without any formatting done to it.

## 8.D.2. Radar Charts

The `ColInfo` parameters for a Radar chart are the same as the parameters for a Column/Bar/Line/Area chart **except** the subvalue. A secondary axis cannot be defined.

### 8.D.2.1. Column Mapping

For instance, given the data shown below:

Order #	Product	Units Ordered
12	Chair	15
12	Table	34
12	Cabinet	21
12	Dresser	24
14	Chair	23
14	Table	23
14	Cabinet	16
14	Dresser	19

*Original Data*

The following code sets the category to be Column 1 (Product), the series to be Column 0 ("Order #") and the value to be Column 2 ("Units Ordered"):

```
ColInfo colInfo = new ColInfo();
colInfo.category = 1;
colInfo.series = 0;
colInfo.value = 2;
```

### 8.D.2.2. Creating the Chart

The following code demonstrates how to create the aforementioned Radar chart.

```
Component doDataFromArguments(Applet applet) {

    // Do not connect to ERES Server
    QbChart.setEspressManagerUsed(false);

    // Column Mapping
    ColInfo colInfo = new ColInfo();
    colInfo.category = 1;
    colInfo.series = 0;
    colInfo.value = 2;

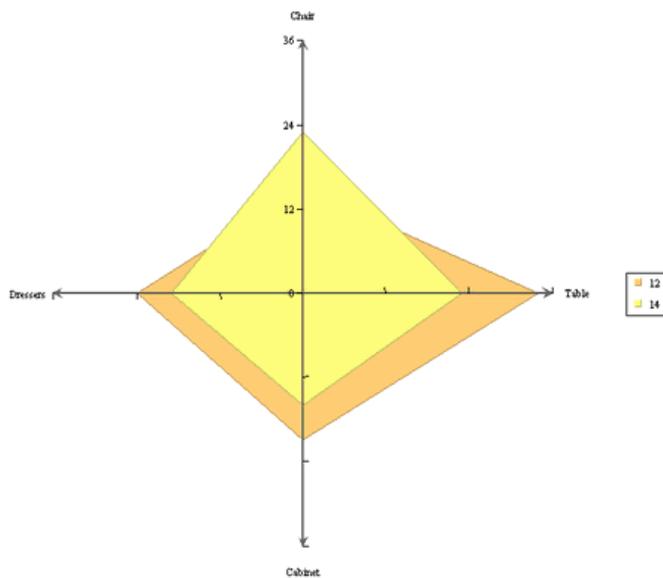
    String dataType[] = {"integer", "varchar", "decimal"};
    String fieldName[] = {"Order #", "Product", "Units Ordered"};
    String records[][] = {"12", "Chair", "15"},
{"12", "Table", "34"},
{"12", "Cabinet", "21"}, {"12"
, "Dresser", "24"},
{"14", "Chair", "23"}, {"14"
, "Table", "23"},
```

```

        {"14" , "Chair", "23"}, {"14"
, "Table", "23"}];
        DbData data = new DbData(dataType, fieldName, records);
        QbChart chart = new QbChart
        (applet,
        QbChart.VIEW2D,
        QbChart.RADAR,
        data,
        colInfo,
        null);
        // Applet
        // Two-Dimensional
        // Radar Chart
        // Data
        // Column information
        // No specified template
        return chart; }
    
```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/RadarChart.zip> ]

When the above code is run as a Java Web Start Application, it will produce a two-dimensional Radar chart shown below:



Generated Radar Chart

Note that the chart created is a default chart without any formatting done to it.

### 8.D.3. XY(Z) Scatter Charts

Scatter charts need a `xvalue` and a `yvalue` axis. Those are the minimum requirements for constructing such types of charts. You can also add in a series and a `zvalue` (for three-dimensional Scatter charts) axis if needed.

#### 8.D.3.1. Column Mapping

For instance, given the data shown below:

Season	High Temperature	Average Temperature	Low Temperature
Summer	110	101	96
Fall	103	85	78

Season	High Temperature	Average Temperature	Low Temperature
Winter	85	75	67
Spring	93	88	81

#### Original Data

The following code sets the xvalue to be Column 2 (Average Temperature), the yvalue to be Column 1 (High Temperature), and the zvalue to be Column 3 (Low Temperature):

```
ColInfo colInfo = new ColInfo();
colInfo.xvalue = 2;
colInfo.yvalue = 1;
colInfo.zvalue = 3;
```

### 8.D.3.2. Creating the Chart

The following code demonstrates how to create the aforementioned Scatter chart.

```
Component doDataFromArguments(Applet applet) {

    // Do not connect to ERES Server
    QbChart.setEspressManagerUsed(false);

    // Column Mapping
    ColInfo colInfo = new ColInfo();
    colInfo.xvalue = 2;
    colInfo.yvalue = 1;
    colInfo.zvalue = 3;

    String dataType[] =
    {"varchar", "integer", "integer", "integer"};
    String fieldName[] = {"Season", "High Temperature", "Average
Temperature", "Low Temperature"};
    String records[][] = {"Summer", "110", "101", "96"},
    {"Fall", "103", "85", "78"},
    {"Spring", "93", "88", "81"};
    DbData data = new DbData(dataType, fieldName, records);
    QbChart chart = new QbChart

    (applet,
    QbChart.VIEW3D,
    QbChart.SCATTER,
    data,
    colInfo,
    null);

    // Applet

    // Three-Dimensional

    // Scatter Chart

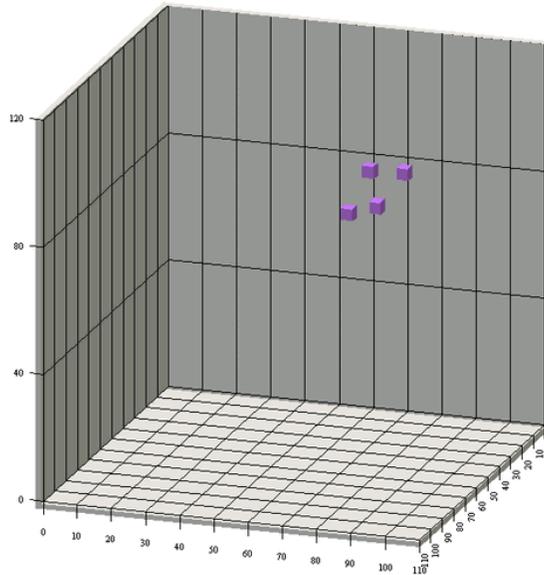
    // Data

    // Column information

    // No specified template
    return chart; }
```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/XYZScatterChart.zip> ]

When the above code is run as a Java Web Start Application, it will produce a three-dimensional Scatter chart shown below:



*Generated Scatter Chart*

Note that the chart created is a default chart without any formatting done to it.

## 8.D.4. Stack Column, Percentage Column, Stack Bar and Stack Area Charts

In addition to the parameters for a Column/Line/Area chart, a Stack Column/Percentage Column/Stack Bar/Stack Area also requires the `SumBy` parameter to be set. The minimum requirements for creating a Stack Column/Percentage Column/Stack Bar/Stack Area chart are the `category`, the `value` and the `sumby` parameters. You can also add in a `series` and a `subvalue` if needed.

### 8.D.4.1. Column Mapping

For instance, given the data shown below:

Day	Drink	Total	Average
Monday	Water	101	5.4
Monday	Coffee	85	2.4
Tuesday	Water	143	6.7
Tuesday	Coffee	92	2.5
Wednesday	Water	186	7.6
Wednesday	Coffee	121	4.2
Thursday	Water	173	6.3
Thursday	Coffee	75	1.1
Friday	Water	88	3.6
Friday	Coffee	193	5.9
Saturday	Water	152	7.3
Saturday	Coffee	57	1.6
Sunday	Water	194	8.8
Sunday	Coffee	25	0.6

*Original Data*

The following code sets the category to be Column 0 (“Day”), the value to be Column 2 (“Total”), the sumby to be Column 1 (“Drink”) and the subvalue to be Column 3 (“Average”):

```
ColInfo colInfo = new ColInfo();
colInfo.category = 0;
colInfo.value = 2;
colInfo.subvalue = 3;
colInfo.sumBy = 1;
```

### 8.D.4.2. Creating the Chart

The following code demonstrates how to create the aforementioned Stack Area chart.

```
Component doDataFromArguments(Applet applet) {

    // Do not connect to ERES Server
    QbChart.setEspressManagerUsed(false);

    // Column Mapping
    ColInfo colInfo = new ColInfo();
    colInfo.category = 0;
    colInfo.value = 2;
    colInfo.subvalue = 3;
    colInfo.sumBy = 1;

    String dataType[] = {"varchar", "varchar", "integer", "double"};
    String fieldName[] = {"Day", "Drink", "Total", "Average"};
    String records[][] = {"Monday", "Water", "101", "5.4"},
        {"Monday", "Coffee", "85", "2.4"},

        {"Tuesday", "Water", "143", "6.7"}, {"Tuesday", "Coffee", "92", "2.5"},

        {"Wednesday", "Water", "186", "7.6"}, {"Wednesday", "Coffee", "121", "4.2"},

        {"Thursday", "Water", "173", "6.3"}, {"Thursday", "Coffee", "75", "1.1"},
        {"Friday", "Water", "88", "3.6"},
        {"Friday", "Coffee", "193", "5.9"},

        {"Saturday", "Water", "152", "7.3"}, {"Saturday", "Coffee", "57", "1.6"},
        {"Sunday", "Water", "194", "8.8"},
        {"Sunday", "Coffee", "25", "0.6"};

    DbData data = new DbData(dataType, fieldName, records);
    QbChart chart = new QbChart

        (applet,

        QbChart.VIEW2D,

        QbChart.STACKAREA,

        data,

        colInfo,

        null);

    // Applet

    // Two-Dimensional

    // Stack Area Chart

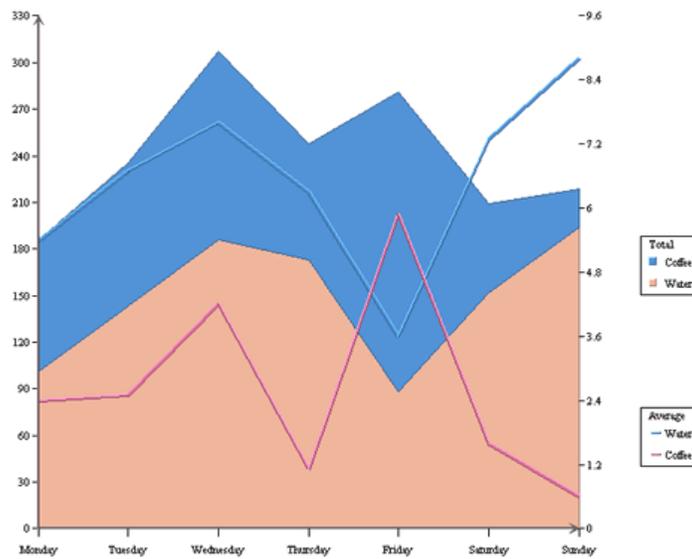
    // Data

    // Column information

    // No specified template
    return chart; }
```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/StackAreaChart.zip> ]

When the above code is run as a Java Web Start Application, it will produce a two-dimensional Stack Area chart shown below:



Generated Stack Area Chart

Note that the chart created is a default chart without any formatting done to it.

## 8.D.5. Dial Charts

The `ColInfo` parameters for a Dial chart are the same as the parameters for a Radar chart **except** the `series`. A `series` column cannot be defined.

### 8.D.5.1. Column Mapping

For instance, given the data shown below:

Product	Units Ordered
Chair	15
Table	34
Cabinet	21
Dresser	24

Original Data

The following code sets the category to be Column 0 (“Product”) and the value to be Column 1 (“Units Ordered”):

```
ColInfo colInfo = new ColInfo();
colInfo.category = 0;
colInfo.value = 1;
```

### 8.D.5.2. Creating the Chart

The following code demonstrates how to create the aforementioned Dial chart.

```
Component doDataFromArguments(Applet applet) {
    // Do not connect to ERES Server
```

```

QbChart.setEspressManagerUsed(false);

// Column Mapping
ColInfo colInfo = new ColInfo();
colInfo.category = 0;
colInfo.value = 1;

String dataType[] = {"varchar", "integer"};
String fieldName[] = {"Product", "Units Ordered"};
String records[][] = {"Chair", "15"}, {"Table", "34"},
                    {"Cabinet", "21"},
{"Dresser", "24"};

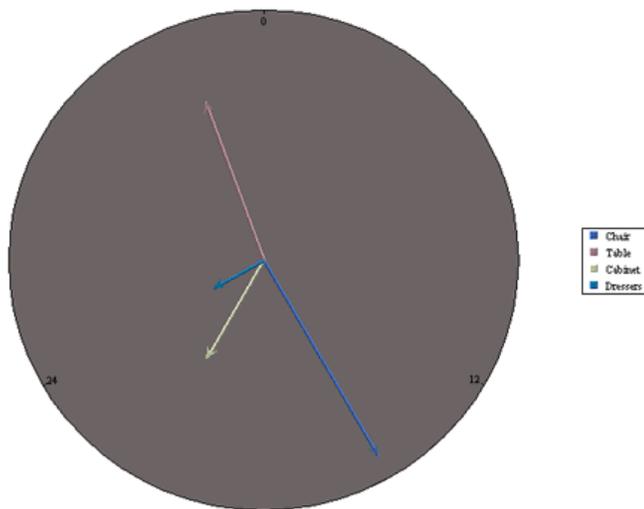
DbData data = new DbData(dataType, fieldName, records);
QbChart chart = new QbChart

// Applet
// Two-Dimensional
// Dial Chart
// Data
// Column information
// No specified template
return chart; }
        (applet,
        QbChart.VIEW2D,
        QbChart.DIAL,
        data,
        colInfo,
        null);

```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/DialChart.zip> ]

When the above code is run as a Java Web Start Application, it will produce a two-dimensional Dial chart shown below:



Generated Dial Chart

Note that the chart created is a default chart without any formatting done to it.

## 8.D.6. Box Charts

The ColInfo parameters for a Box chart are the same as the parameters for a Column/Bar/Line/Area/Pie/Overlay chart **except** the series. A series column cannot be defined.

## 8.D.6.1. Column Mapping

For instance, given the data shown below:

Subject	Student	Score
Math	A.S.	65
Math	T.E.	75
Math	V.Q.	83
Math	X.C.	87
Math	I.Z.	93
Science	A.S.	86
Science	T.E.	90
Science	V.Q.	73
Science	X.C.	95
Science	I.Z.	84

### *Original Data*

The following code sets the category to be Column 0 (“Subject”) and the value to be Column 2 (“Score”):

```
ColInfo colInfo = new ColInfo();
colInfo.category = 0;
colInfo.value = 2;
```

## 8.D.6.2. Creating the Chart

The following code demonstrates how to create the aforementioned Box chart.

```
Component doDataFromArguments(Applet applet) {,

    // Do not connect to ERES Server
    QbChart.setEspressManagerUsed(false);

    // Column Mapping
    ColInfo colInfo = new ColInfo();
    colInfo.category = 0;
    colInfo.value = 2;

    String dataType[] = {"varchar", "varchar", "integer"};
    String fieldName[] = {"Subject", "Student", "Score"};
    String records[][] = {"Math", "A.S.", "65"},
{"Math", "T.E.", "75"},
{"Math", "V.Q.", "83"},
{"Math", "X.C.", "87"},
{"Math", "I.Z.", "93"},
{"Science", "A.S.", "86"},
{"Science", "T.E.", "90"},
{"Science", "V.Q.", "73"},
{"Science", "X.C.", "95"},
{"Science", "I.Z.", "84"};

    DbData data = new DbData(dataType, fieldName, records);
    QbChart chart = new QbChart

    (applet,
```

```
// Applet
```

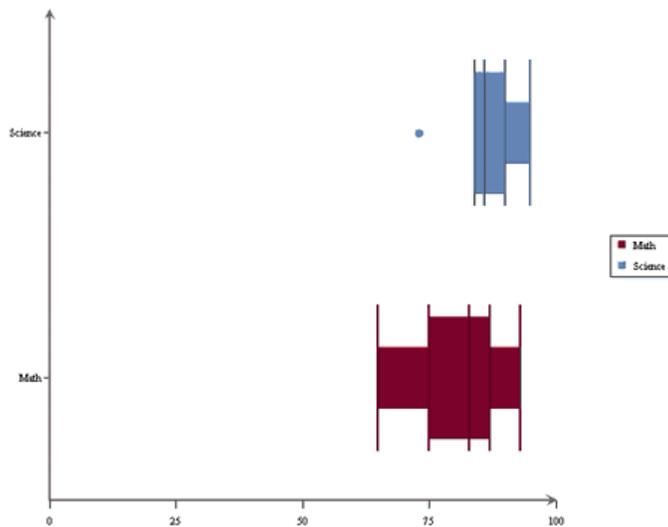
```

// Two-Dimensional
// Box Chart
// Data
// Column information
// No specified template
return chart; }
QbChart.VIEW2D,
QbChart.BOX,
data,
colInfo,
null);

```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/BoxChart.zip> ]

When the above code is run as a Java Web Start Application, it will produce a two-dimensional Box chart shown below:



Generated Box Chart

Note that the chart created is a default chart without any formatting done to it.

## 8.D.7. Bubble Charts

The ColInfo parameters for a Bubble chart are the same as the parameters for a three-dimensional Scatter Chart. The xvalue, yvalue and zvalue parameters are necessary to draw a Bubble Chart. Note that for a bubble chart, the zvalue refers to the Bubble Size.

### 8.D.7.1. Column Mapping

For instance, given the data shown below:

Drink	High	Average	Low
Water	3	2	2
Soda	1	1	0
Coffee	5	2	1
Tea	3	1	1

Original Data

The following code sets the series to be Column 0 (“Drink”), xvalue to be Column 1 (“High”), the yvalue to be Column 3 (“Low”), and the zvalue to be Column 2 (“Average”):

```
ColInfo colInfo = new ColInfo();
colInfo.xvalue = 1;
colInfo.yvalue = 3;
colInfo.zvalue = 2;
colInfo.series = 0;
```

## 8.D.7.2. Creating the Chart

The following code demonstrates how to create the aforementioned Bubble chart.

```
Component doDataFromArguments(Applet applet) {

    // Do not connect to ERES Server
    QbChart.setEspressManagerUsed(false);

    // Column Mapping
    ColInfo colInfo = new ColInfo();
    colInfo.xvalue = 1;
    colInfo.yvalue = 3;
    colInfo.zvalue = 2;
    colInfo.series = 0;

    String dataType[] = {"varchar", "integer", "integer", "integer"};
    String fieldName[] = {"Drink", "High", "Average", "Low"};
    String records[][] = {"Water", "3", "2", "2"},
        {"Soda", "1", "1", "0"},
        {"Coffee", "5", "2", "1"},
        {"Tea", "3", "1", "1"};
    DbData data = new DbData(dataType, fieldName, records);
    QbChart chart = new QbChart

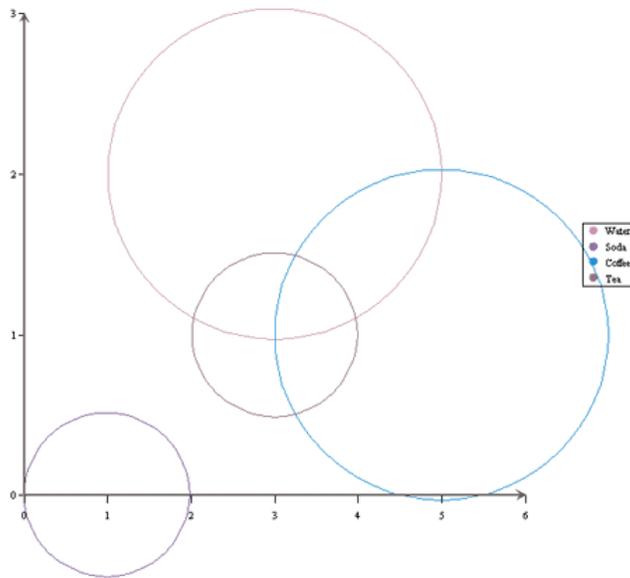
        (applet, //
        QbChart.VIEW2D, //
        QbChart.BUBBLE, //
        data, //
        colInfo, //
        null); // No
        specified template

    return chart; }

```

Full Source Code [ <http://data.quadbases.com/Docs70/help/manual/code/src/BubbleChart.zip> ]

When the above code is run as a Java Web Start Application, it will produce a two-dimensional Bubble chart shown below:



Generated Bubble Chart

Note that the chart created is a default chart without any formatting done to it.

## 8.D.8. High-Low and HLCO Charts

The `ColInfo` parameters for a High-Low/HLCO chart differ from the parameters of the other chart types. Here the value part of the chart is further subdivided into an `open` value, a `close` value, a `high` value and a `low` value. The minimum requirements for a High-Low chart are the `category`, the `high` value and the `low` value parameters (for a HLCO, the `close` value and `open` value parameters are also required). You can also additionally add in the series and subvalue parameters as well.

### 8.D.8.1. Column Mapping

For instance, given the data shown below:

Day	Company	High	Low	Close	Open	Volume
2001-01-01	ABC	1.33	1.18	1.22	1.23	43723
2001-01-01	DEF	9.24	8.74	9.16	8.89	18478
2001-01-01	GHI	2.20	1.82	2.14	1.97	46743
2001-01-02	ABC	1.87	0.79	1.63	1.12	33605
2001-01-02	DEF	9.48	8.12	8.93	8.66	16758
2001-01-02	GHI	2.47	2.32	2.44	2.34	60671
2001-01-03	ABC	2.47	0.22	0.44	0.63	45211
2001-01-03	DEF	9.94	9.92	9.93	9.93	10697
2001-01-03	GHI	2.48	2.40	2.46	2.44	45238
2001-01-04	ABC	1.8	1.38	1.79	1.44	50224
2001-01-04	DEF	9.49	8.87	8.94	8.93	11868
2001-01-04	GHI	2.06	1.45	1.96	1.46	62053
2001-01-05	ABC	1.23	0.58	0.79	0.79	37285
2001-01-05	DEF	9.94	8.61	8.08	8.94	10476
2001-01-05	GHI	2.8	1.58	2.45	1.96	47117

Original Data

The following code sets the category to be Column 0 (Day), the series to be Column 1 (Company), the High to be Column 2 (High), the Low to be Column 3 (Low), the Close to be Column 4 (Close), the Open to be Column 5 (Open) and the subvalue to be Column 6 (Volume):

```
ColInfo colInfo = new ColInfo();
colInfo.category = 0;
colInfo.series = 1;
colInfo.high = 2;
colInfo.low = 3;
colInfo.close = 4;
colInfo.open = 5;
colInfo.subvalue = 6;
```

### 8.D.8.2. Creating the Chart

The following code demonstrates how to create the aforementioned HLCO chart.

```
Component doDataFromArguments(Applet applet) {

    // Do not connect to ERES Server
    QbChart.setEspressManagerUsed(false);

    // Column Mapping
    ColInfo colInfo = new ColInfo();
    ColInfo colInfo = new ColInfo();
    colInfo.category = 0;
    colInfo.series = 1;
    colInfo.high = 2;
    colInfo.low = 3;
    colInfo.close = 4;
    colInfo.open = 5;
    colInfo.subvalue = 6;

    String dataType[] =
{"date", "varchar", "double", "double", "double", "double", "integer"};
    String fieldName[] =
{"Day", "Company", "High", "Low", "Close", "Open", "Volume"};
    String records[][] =
{{"2001-01-01", "ABC", "1.33", "1.18", "1.22", "1.23", "43723"},
{"2001-01-01", "DEF", "9.24", "8.74", "9.16", "1.23", "18478"},
{"2001-01-01", "GHI", "2.20", "1.82", "2.14", "1.23", "46743"},
{"2001-01-02", "ABC", "1.87", "0.79", "1.63", "1.23", "33605"},
{"2001-01-02", "DEF", "9.48", "8.12", "8.93", "1.23", "16758"},
{"2001-01-02", "GHI", "2.47", "2.32", "2.44", "1.23", "60671"},
{"2001-01-03", "ABC", "1.12", "0.22", "0.44", "1.23", "45211"},
{"2001-01-03", "DEF", "9.94", "9.92", "9.93", "9.93", "10697"},
{"2001-01-03", "GHI", "2.48", "2.40", "2.46", "2.44", "45238"},
{"2001-01-04", "ABC", "1.80", "1.38", "1.79", "1.44", "50224"},
```

```

{"2001-01-04", "DEF", "9.49", "8.87", "8.94", "8.93", "11868"},
{"2001-01-04", "GHI", "2.06", "1.45", "1.96", "1.46", "62053"},
{"2001-01-05", "ABC", "1.23", "0.58", "0.79", "0.79", "37285"},
{"2001-01-05", "DEF", "9.94", "8.61", "8.08", "8.94", "10476"},
{"2001-01-05", "GHI", "2.80", "1.58", "2.45", "1.96", "47117"};
    DbData data = new DbData(dataType, fieldName, records);
    QbChart chart = new QbChart

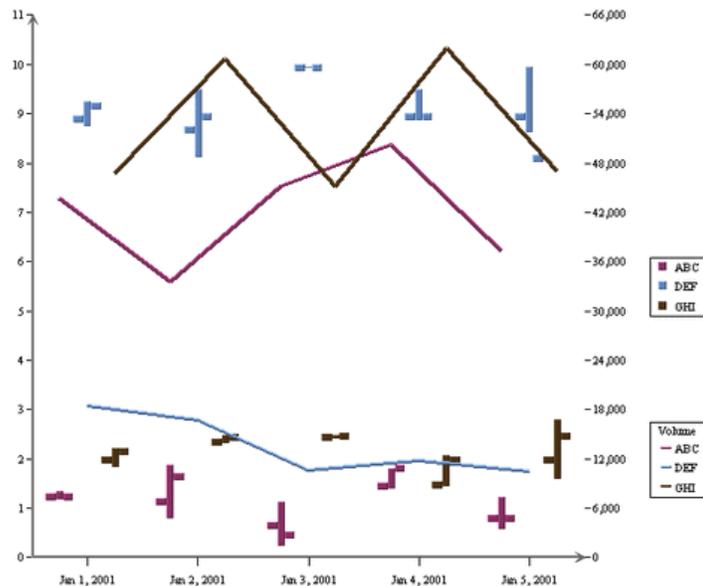
                                                                    (applet,
                                                                    QbChart.VIEW2D,
                                                                    QbChart.HLCO,
                                                                    data,
                                                                    colInfo,
                                                                    null);

// Applet
// Two-Dimensional
// HLCO Chart
// Data
// Column information
// No specified template
    return chart; }

```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/HLCOChart.zip> ]

When the above code is run as a Java Web Start Application, it will produce a two-dimensional HLCO chart shown below:



Generated HLCO Chart

Note that the chart created is a default chart without any formatting done to it.

## 8.D.9. Surface Charts

The ColInfo parameters for a Surface chart are similar to those of a three-dimensional Scatter chart. The xvalue, yvalue and zvalue parameters are required to create a surface chart. However, a surface chart does not support the series parameter. For a surface chart, the ColInfo object defined is always the same no matter what the data is.

### 8.D.9.1. Column Mapping

For instance, given the data shown below:

	0	10	20	30	40	50	60	70	80	90	100
0	0	0	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	1	0	0	0	0	0
20	0	0	0	0	1	2	1	0	0	0	0
30	0	0	0	1	2	3	2	1	0	0	0
40	0	0	1	2	3	4	3	2	1	0	0
50	0	1	2	3	4	5	4	3	2	1	0
60	0	0	1	2	3	4	3	2	1	0	0
70	0	0	0	1	2	3	2	1	0	0	0
80	0	0	0	0	1	2	1	0	0	0	0
90	0	0	0	0	0	1	0	0	0	0	0
100	0	0	0	0	0	0	0	0	0	0	0

*Original Data*

The following code sets the mapping for the surface chart. Note that this mapping is constant for surface charts:

```
int map[] = {0, 2, 1};
ColumnInfo colInfo = new ColumnInfo(-1, map);
```

### 8.D.9.2. Creating the Chart

The following code demonstrates how to create the aforementioned Surface chart.

```
Component doDataFromArguments(Applet applet) {

    QbChart.setEspressoManagerUsed(false);

    int map[] = { 0, 2, 1 };
    ColumnInfo colInfo = new ColumnInfo(-1, map);

    String inputFileName = "surface.dat";

    QbChart chart = null;

    try {

        chart = new QbChart(applet, // Applet
            QbChart.VIEW3D, // Three-Dimensional
            QbChart.SURFACE, // Surface Chart
            QbChart.DATAFILE, // Type of text file
            inputFileName, // Data
            false, // Do not transpose data
            colInfo, // Column information
            null); // No specified template

    } catch (Exception ex)
    {
        ex.printStackTrace();
    }
}
```

```

}

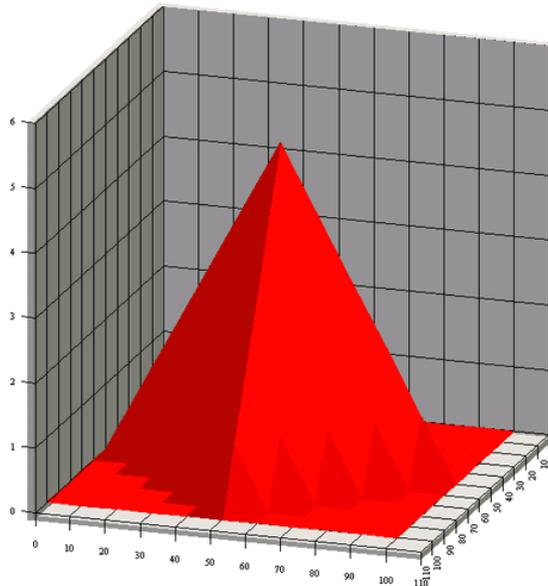
return chart;

}

```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/SurfaceChart.zip> ]

When the above code is run as a Java Web Start Application, it will produce a three-dimensional Surface chart shown below:



*Generated Surface Chart*

Note that the chart created is a default chart without any formatting done to it.

## 8.D.10. Gantt Charts

The ColInfo parameters for a Gantt chart are similar to those of a High-Low chart. Here the category, high (or start date) and low (end date) parameters are required to create a Gantt chart. You can also add a series parameter if needed.

### 8.D.10.1. Column Mapping

For instance, given the data shown below:

Project	Task	Starting Date	Ending Date
Project 1	Task A	1998-01-15	1998-03-01
Project 1	Task B	1998-02-25	1998-05-18
Project 1	Task C	1998-06-11	1998-09-29
Project 2	Task A	1998-03-15	1998-09-29
Project 2	Task B	1998-04-25	1998-08-18
Project 2	Task C	1998-08-11	1998-12-29

*Original Data*

The following code sets the category to be Column 1 (“Task”), the High to be Column 2 (“Starting Date”), the Low to be Column 3 (“Ending Date”) and the series to be Column 0 (“Project”).

```
ColInfo colInfo = new ColInfo();
colInfo.category = 1;
colInfo.high = 2;
colInfo.low = 3;
colInfo.series = 0;
```

## 8.D.10.2. Creating the Chart

The following code demonstrates how to create the aforementioned Gantt chart.

```
Component doDataFromArguments(Applet applet) {

    QbChart.setEspressManagerUsed(false);

    // Column Mapping
    ColInfo colInfo = new ColInfo();
    colInfo.category = 1;
    colInfo.high = 2;
    colInfo.low = 3;
    colInfo.series = 0;

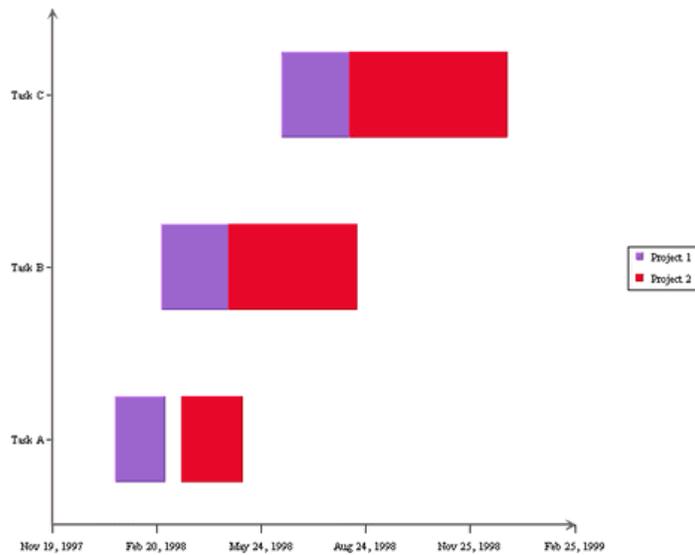
    String dataType[] = {"varchar", "varchar", "date", "date"};
    String fieldName[] = {"Project", "Task", "Starting Date", "Ending Date"};
    String records[][] = {{"Project1", "Task A", "1998-01-15", "1998-03-01"},
                          {"Project1", "Task B", "1998-02-25", "1998-05-18"},
                          {"Project1", "Task C", "1998-06-11", "1998-09-29"},
                          {"Project2", "Task A", "1998-03-15", "1998-05-08"},
                          {"Project2", "Task B", "1998-04-25", "1998-08-18"},
                          {"Project2", "Task
C", "1998-08-11", "1998-12-29"}}};
    DbData data = new DbData(dataType, fieldName, records);
    QbChart chart = new QbChart
        (applet, // Applet
        QbChart.VIEW2D, // Two-Dimensional
        QbChart.GANTT, // Gantt Chart
        data, // Data
        colInfo, // Column information
        null); // No specified

    template
    return chart;

}
```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/GanttChart.zip> ]

When the above code is run as a Java Web Start Application, it will produce a two-dimensional Gantt chart shown below:



Generated Gantt Chart

Note that the chart created is a default chart without any formatting done to it.

## 8.D.11. Polar Charts

The `ColInfo` parameters for a Polar chart are similar to those of a Scatter Chart. Here the `radius`, and `angle` parameters are required to create a Polar chart. You can also add a `series` parameter if needed.

### 8.D.11.1. Column Mapping

For instance, given the data shown below:

Radius	Angle	Series
0	2	A
90	4	A
180	6	A
270	8	A
360	10	A
45	3	B
135	5	B
225	7	B
315	9	B

Original Data

The following code sets the `radius` to be Column 0 (“Radius”), the `Angle` to be Column 1 (“Angle”), and the `series` to be Column 2 (“Series”).

```
ColInfo colInfo = new ColInfo(2, new int[] {0, 1});
```

### 8.D.11.2. Creating the Chart

The following code demonstrates how to create the aforementioned Polar chart.

```
Component doDataFromArguments(Applet applet) {
```

```

QbChart.setEspressManagerUsed(false);

// Column Mapping
ColumnInfo colInfo = new ColumnInfo(2, new int[]{0, 1});

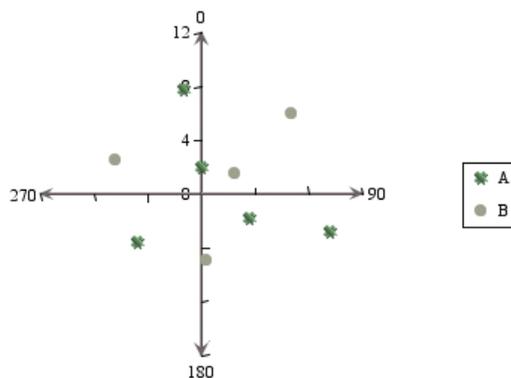
String dataType[] = {"integer", "integer", "varchar"};
String fieldName[] = {"Radius", "Angle", "Series"};
String records[][] = {{"0", "2", "A"}, {"90", "4", "A"},
{"180", "6", "A"}, {"270", "8", "A"}, {"360", "10", "A"},
{"45", "3", "B"}, {"135", "5", "B"}, {"225", "7", "B"},
{"315", "9", "B"};
DbData data = new DbData(dataType, fieldName, records);
QbChart chart = new QbChart
    (applet, // Applet
    QbChart.VIEW2D, // Two-Dimensional
    QbChart.POLAR, // Polar Chart
    data, // Data
    colInfo, // Column information
    null); // No specified template

return chart;
}

```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/PolarChart.zip> ]

When the above code is run as a Java Web Start Application, it will produce a two-dimensional Polar chart shown below:



Generated Polar Chart

Note that the chart created is a default chart without any formatting done to it.

## 8.D.12. Date/Time Based Zoom Charts

ERES allows date/time based zooming in charts. The date/time based zooming can be applied to the chart of almost any type (except high-low, HLCO, scatter, Surface, Box, Dial, Radar, Bubble and Gantt charts). The only major requirement is that the data along the category axis be of date, time or timestamp type.

Zooming can be achieved by setting the attributes and then refreshing the chart. The attributes are set using the `quadbase.util.IZoomInfo` interface.

The following example, which can run as example or application, shows a chart that incorporates zooming. Here the default zooming shows 5 days at a time while the maximum zoom-out allowed is 1 month and the maximum zoom-in allowed is 1 day.

```

// Data passed in an array in memory
DbData chartData = new DbData(dataTypes, fieldNames, data);

// Set Column Mapping
ColumnInfo colInfo = new ColumnInfo();
colInfo.category = 1;
colInfo.value = 0;

// Create Chart
QbChart chart = new QbChart(
    (Applet)null, // Applet
    QbChart.VIEW2D, // Dimension
    QbChart.COL, // Chart Type
    chartData, // Data
    colInfo, // Column Mapping
    null); // Template

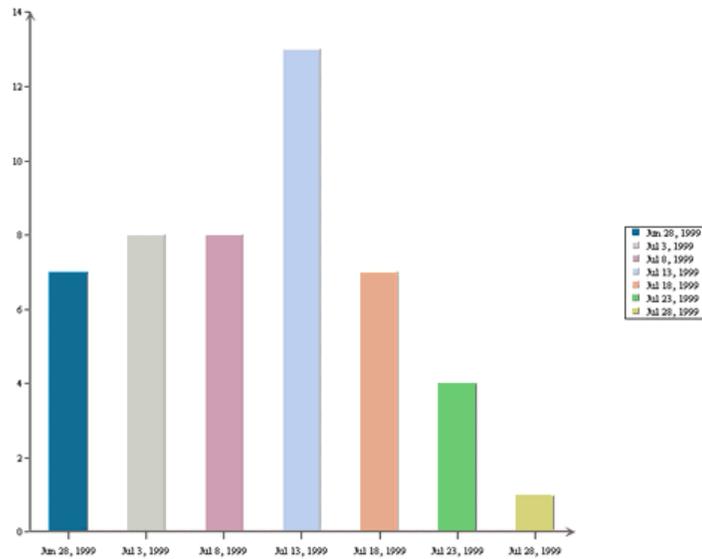
// Get a handle to the Zooming interface
IZoomInfo zoomInfo = chart.getZoomInfo();
zoomInfo.setAggregateOperator(IZoomInfo.SUM); // Specify the aggregation
zoomInfo.setMaxScale(1, IZoomInfo.YEAR); // Specify max zoom out
zoomInfo.setMinScale(1, IZoomInfo.DAY); // Specify max zoom in
zoomInfo.setScale(5, IZoomInfo.DAY); // Specify current zooming
zoomInfo.setLinearScale(true); // Turn on Linear Scale
zoomInfo.setZoomEnabled(true); // Turn on Zooming

try
{
    chart.refresh(); // Refresh the chart with
    zooming
} catch (Exception ex)
{
    ex.printStackTrace();
}
return chart;

```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/ZoomChartAppendix.zip> ]

When the above code is run as a Java Web Start Application, it will produce a top-level chart shown below:



Generated Chart

You can then zoom in or zoom out on this chart, depending on the selections you make in the pop-up menu (which you can see by right-clicking on the chart canvas).

Please note that this is a default chart that is generated without formatting of any kind.

## 8.D.13. Parameterized Charts

In addition to regular queries, you can pass in queries that has one, or more, parameters and have the chart prompt the user for values for the parameters, before generating the chart. Note that only stand-alone charts can be parameterized.

To use a parameterized query as your data source for your chart, you must create a class that implements `IQueryFileInfo` (you can use the implementation already provided, `SimpleQueryFileInfo`) and use that class to pass in the parameter information.

Given below is an example of a chart that uses a parameterized query. The database against which the query is run is WoodView HSQL, so you will need to add database HSQL JDBC driver (`hsqldb.jar`) to your classpath. The driver is located in the `<ERESInstall>/WEB-INF/lib` directory. Please note that you have to have ERES server running to run this example.

```
Component doParameterizedChart(Applet applet) {

    // Connect to ERES Server
    QbChart.useServlet(true);
    QbChart.setServletRunner("http://localhost:8080");
    QbChart.setServletContext("ERES/servlet");

    // Begin Code : Adding Parameter Info
    // SimpleQueryInParam(name of Parameter, String to be displayed,
    MapToColumn?,
                                // tableName, ColumnName, SQL
    Type, DefaultValue, value)
    SimpleQueryInParam inParam = new SimpleQueryInParam("param",
                                                         "Please
select",true, "Categories",
Types.VARCHAR,
                                                         "CategoryName",
                                                         "Arm Chairs",
null);
```

```

SimpleQueryInParam[] paramSet = {inParam};

SimpleQueryFileInfo chartInfo = new
SimpleQueryFileInfo("jdbc:hsqldb:help/examples/DataSources/database/
woodview",

"org.hsqldb.jdbcDriver", "sa", "",

"select
Products.ProductName, Products.UnitsInStock
from Categories,
Products where Categories.CategoryID=Products.CategoryID
and
Categories.CategoryName=:param order by Categories.CategoryName,
Products.ProductName;");
chartInfo.setInParameter(paramSet);
// End Code : Adding Parameter Info

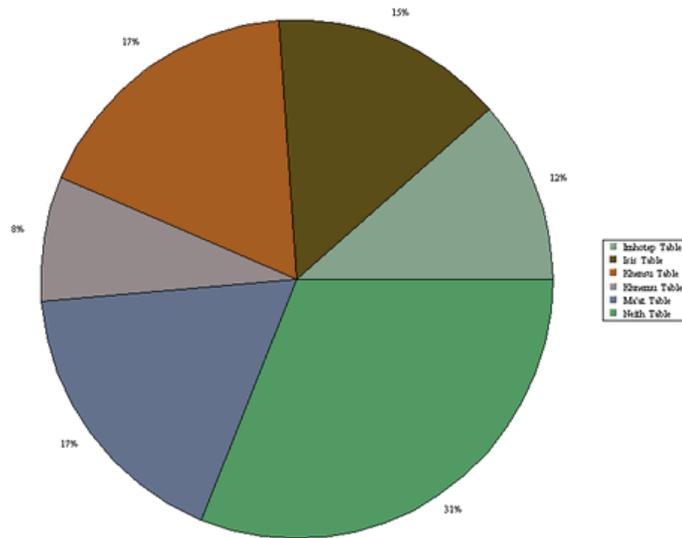
// Begin Code : Setting up Column Mapping
ColumnInfo colInfo = new ColumnInfo();
colInfo.category = 0;
colInfo.value = 1;
// End Code : Setting up Column Mapping

QbChart chart = new QbChart(applet, QbChart.VIEW2D, QbChart.PIE,
chartInfo, colInfo, null);
return chart; }

```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/ParameterizedChartERES.zip> ]

When the above code is run as a Java Web Start Application, it will produce a chart (depending on the parameter selected) similar to the one shown below:



Generated Chart

Please note that this is a default chart that is generated without formatting of any kind.

If you add a parameterized chart to a report, then you must link a column from the main report to provide the values for the parameters defined in the chart. You do this by using the following method in ReportChartObject:

```
public void setParameterMap(String[] columnID);
```

The string array contains the IDs of the columns that will provide the values for the parameters in the chart.

You can also assign a parameter to have multiple values, for example, in the case where a user wants to check against a range of values rather than just a single value. The range of values is usually specified within the “IN” clause of a SQL query. Note that ERES only considers parameters within the “IN” clause to be multi-value.

To use a multi-value parameterized query as your data source for your chart, you must create a class that implements IQueryFileInfo and use that class to pass in the parameter information.

Given below is an example of a chart that uses a multi-value parameterized query. The database against which the query is run is WoodView HSQL, so you will need to add database HSQL JDBC driver (hsqldb.jar) to your classpath. The driver is located in the <ERESInstall>/WEB-INF/lib directory. Please note that you have to have ERES server running to run this example.

```

Component doMultiValueParameter(Applet applet) {

    // Connect to ERES Server
    QbChart.useServlet(true);
    QbChart.setServletRunner("http://localhost:8080");
    QbChart.setServletContext("ERES/servlet");

    // Begin Code : Adding Parameter Info
    SimpleQueryMultiValueInParam inParam = new
SimpleQueryMultiValueInParam("param",
                                "Please select", true,
                                "Categories", "CategoryName", Types.VARCHAR,
                                "Arm Chairs", null);

    SimpleQueryMultiValueInParam[] paramSet = {inParam};
    SimpleQueryFileInfo chartInfo = new
SimpleQueryFileInfo("jdbc:hsqldb:help/examples/DataSources/database/
woodview",
                    "org.hsqldb.jdbcDriver", "sa", "",
                    "select
Products.ProductName,
Products.UnitsInStock
from Categories,
Products where
Categories.CategoryID=Products.CategoryID
and
Categories.CategoryName in(:param)
order by
Categories.CategoryName,
Products.ProductName;");
    chartInfo.setInParam(paramSet);

    // End Code : Adding Parameter Info

    // Begin Code : Setting up Column Mapping
    ColInfo colInfo = new ColInfo();
    colInfo.category = 0;
    colInfo.value = 1;
    // End Code : Setting up Column Mapping

```

```

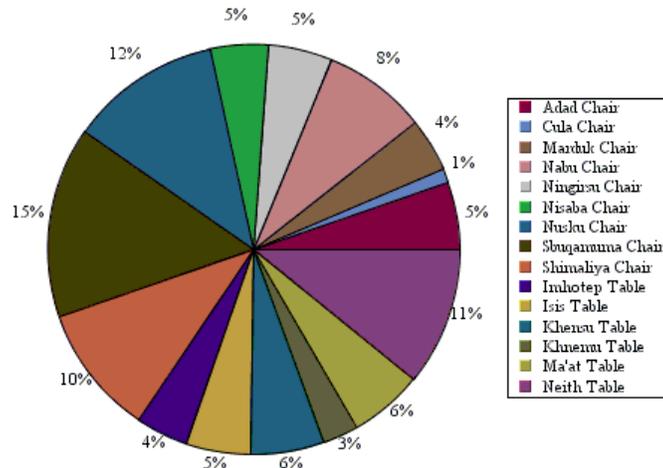
        QbChart chart = new QbChart(applet, QbChart.VIEW2D, QbChart.PIE,
        chartInfo, colInfo, null);

        return chart;
    }
}

```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/MultiValueParameterizedChartERES.zip> ]

When the above code is run as a Java Web Start Application, it will produce a chart (depending on the parameter selected) similar to the one shown below:



*Generated Chart*

Please note that this is a default chart that is generated without formatting of any kind.

Note that If there are more than one parameters in the IN clause, each of them is considered single-value. For example, in the query:

```

Select * From Products
Where ProductID IN (:param1, :param2, :param3)

```

:param1, :param2, and :param3 are all single-value parameters.

Initializing a multi-value parameter is the same as initializing a single-value parameter. The only difference is in the value selection dialog. While single-value parameters will translate into a drop down box, multi-value parameters will be given a multi-selection list box.

## 8.D.14. Drill-Down Charts

ERES supports different drill-down capabilities. They are:

- Parameter drill-down
- Data drill-down
- Dynamic drill-down

Like parameterized charts, drill-down charts are available only when generating stand-alone charts.

Constructing the different types of drill-down charts are described in the sections below.

### 8.D.14.1. Parameter Drill-Down Charts

With the help of parameterized queries, parameter drill-down charts can be created. Instead of having a chart with large amounts of data in it, you can show a top level chart showing the minimum data required and then delve

deeper on the selected data. For more information on parameter drill-down charts, please refer to the Section 4.2.5.3 - Parameter Drill-Down

To create a parameter drill-down chart, you need to create the various chart objects (please note that all chart objects, other than the root chart object, will have parameterized queries as their data source) and then specify the order of the drill-down as well as the column/bar/point/line/slice of the chart to attach the next level of the parameter drill-down chart.

Given below is an example of a parameter drill-down chart. The database against which the query is run is Wood-View HSQL, so you will need to add database HSQL JDBC driver (hsqldb.jar) to your classpath. The driver is located in the <ERESInstall>/WEB-INF/lib directory. Please note that you have to have ERES server running to run this example.

```
Component doDrillDownChart(Applet applet) {

    // Connect to ERES Server
    QbChart.useServlet(true);
    QbChart.setServletRunner("http://localhost:8080");
    QbChart.setServletContext("ERES/servlet");

    // Begin Code : Creating Chart 1 - the Root Chart
    DBInfo rootInfo = new DBInfo("jdbc:hsqldb:help/examples/DataSources/
database/woodview", "org.hsqldb.jdbcDriver",
        "sa",
        "",
        "SELECT Employees.LastName, Count(Orders.OrderID) AS
Count_Of_OrderID FROM Employees, Order_Details, Orders WHERE
(Orders.EmployeeID = Employees.EmployeeID) AND (Orders.OrderID =
Order_Details.OrderID) GROUP BY Employees.LastName");

    ColInfo rootColInfo = new ColInfo(-1, 0, -1, 1);
    QbChart rootChart = new QbChart(applet, QbChart.VIEW2D, QbChart.PIE,
rootInfo, false, rootColInfo, null);

    SimpleQueryInParam inParam = new SimpleQueryInParam("LastName", "Please
select", true,
        "Customers", "Company", Types.VARCHAR,
        "All Unfinished Furniture", null);

    SimpleQueryInParam[] paramSet = {inParam};

    SimpleQueryFileInfo levelChartInfo = new
SimpleQueryFileInfo("jdbc:hsqldb:help/examples/DataSources/database/
woodview",
        "org.hsqldb.jdbcDriver", "sa", "",
        "SELECT Categories.CategoryName, Employees.LastName,
Sum(Order_Details.Quantity) AS Sum_Of_Quantity FROM Products,
Employees, Categories, Order_Details, Orders WHERE (Orders.OrderID =
Order_Details.OrderID) AND (Employees.EmployeeID = Orders.EmployeeID)
AND (Products.CategoryID = Categories.CategoryID) AND (Products.ProductID
= Order_Details.ProductID) GROUP BY Categories.CategoryName,
Employees.LastName HAVING (Employees.LastName =:LastName)");

    levelChartInfo.setInParam(paramSet);

    ColInfo levelOneChartColInfo = new ColInfo(-1, 0, -1, 2);

    try {
```

```

rootChart.createDrillDownChart("TestDrillDownChart", QbChart.VIEW2D,
QbChart.COL, levelChartInfo, false, null,
levelOneChartColInfo, null, new int[] {0});
rootChart.updateDrillDownCharts();

} catch (Exception ex) { ex.printStackTrace(); }

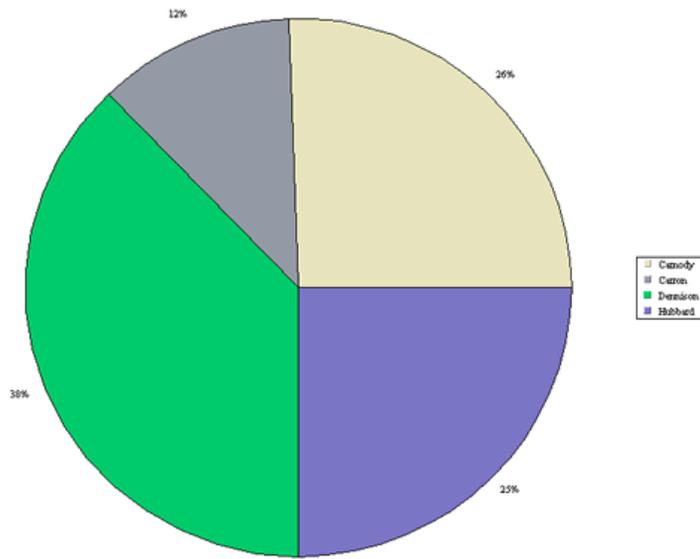
return rootChart;

}

```

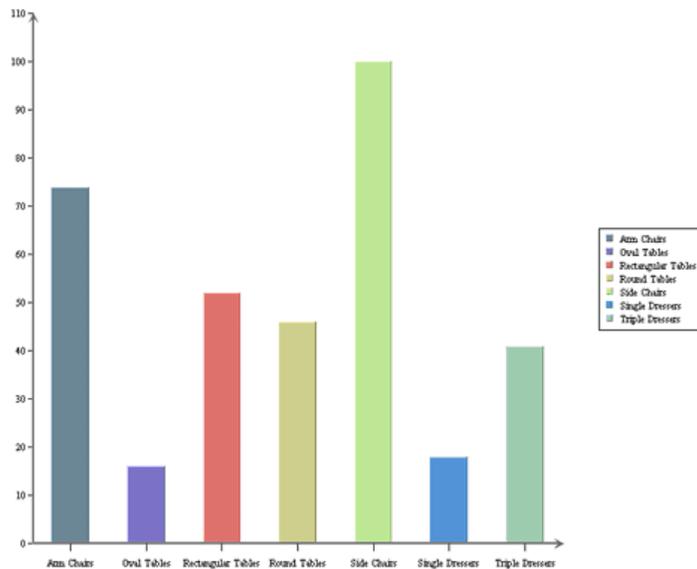
Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/ParameterDrillDownChartERES.zip> ]

When the above code is run as a Java Web Start Application, it will produce a top-level chart shown below:



*Generated top-level chart*

Depending on the link clicked, you will see a chart similar to the one shown below:



*Generated chart*

Please note that this is a default chart that is generated without formatting of any kind.

When you generate parameter drill-down charts without using the ERES Server, you **MUST** have a sub directory called `drillTemplates` under the working directory of the `.class` file.

## 8.D.14.2. Data Drill-Down Charts

Parameter drill-down charts allow charts to be generated from different data sources. With data drill-down, the same data source is used throughout the different levels of the charts. The top-level presents a summary of the data. You can click on a data point to bring up another chart showing some detailed information about that particular data point.

Only the column, bar, line, pie, area, overlay, radar, dial, stack column and stack bar supports the data drill-down functionality.

To create a data drill-down chart, you need to create a chart object first before specifying the drill-down properties.

Given below is an example of a data drill-down chart. The database against which the query is run is WoodView HSQL, so you will need to add database HSQL JDBC driver (`hsqldb.jar`) to your classpath. The driver is located in the `<ERESInstall>/WEB-INF/lib` directory. Please note that you have to have ERES server running to run this example.

```

Component doDataDrillDownChart(Applet applet) {

    // Connect to ERES Server
    QbChart.useServlet(true);
    QbChart.setServletRunner("http://localhost:8080");
    QbChart.setServletContext("ERES/servlet");

    // Begin Code : Creating Chart 1 - the Root Chart
    DBInfo rootInfo = new DBInfo("jdbc:hsqldb:help/examples/
DataSources/database/woodview",
                                "org.hsqldb.jdbcDriver",
                                "sa",
                                "",
                                "select Categories.CategoryName,
Products.ProductName,
Categories,
Products.UnitsInStock from
Products where
Categories.CategoryID = Products.CategoryID
order by
Categories.CategoryName;");

    ColInfo rootColInfo = new ColInfo();
    rootColInfo.category = 0;
    rootColInfo.value = 2;
    QbChart chart = new QbChart(applet, QbChart.VIEW2D, QbChart.BAR,
rootInfo, rootColInfo, null);

    // End Code : Creating Chart 1 - the Root Chart

    try {

        // Begin Code : Creating Chart 2 - the sub Chart
        IDrillDown chartDrillDown = chart.getDrillDown();
        chartDrillDown.setAggregateOperator(IDrillDown.SUM);
        // Set the Aggregation
        chartDrillDown.addDrillDown(QbChart.AREA, 1, -1, -1,
true); // addDrillDown(chartType, category, series, sumby, is2DChart)

```

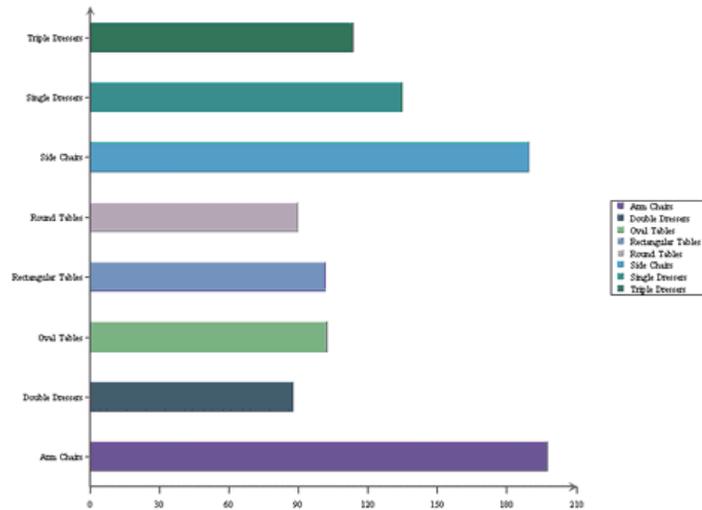
```

        chartDrillDown.setDrillName("DrillDownChart");
// Set the drill template name
    } catch (Exception ex)
    {
        ex.printStackTrace(); }
// End Code : Creating Chart 2 - the sub Chart
return chart;}

```

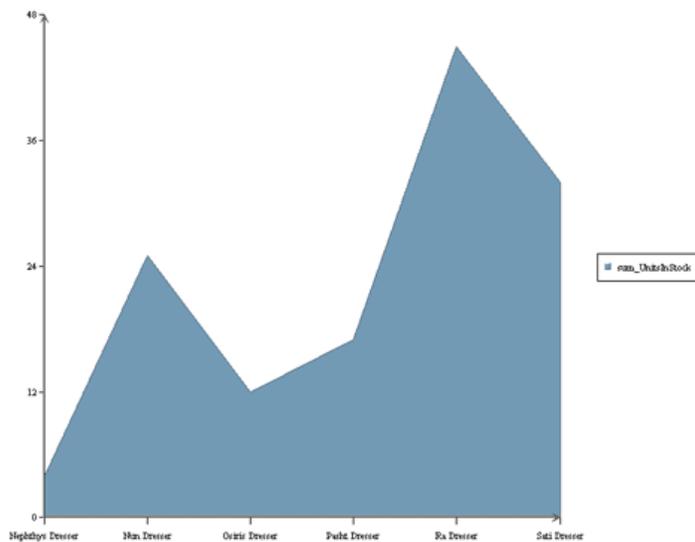
Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/DataDrillDownChartERES.zip> ]

When the above code is run as a Java Web Start Application, it will produce a top-level chart shown below:



Generated top-level chart

Depending on the link clicked, you will see a chart similar to the one shown below:



Generated chart

Please note that this is a default chart that is generated without formatting of any kind.

### 8.D.14.3. Dynamic Data Drill-Down Charts

In data drill-down charts, you have to pre-define the column-to-axis mapping for each drill-down level. Dynamic data drill-down gives you the flexibility to select the column-to-axis mapping for drill-down charts when you are

viewing the chart. Only the top-level summary chart needs to be built and the aggregation specified for the drill-down.

Only the column, bar, line, pie, area, overlay, radar, dial, stack column and stack bar supports the dynamic data drill-down functionality.

To create a dynamic data drill-down chart, you need to create a chart object first before specifying the drill-down properties.

Given below is an example of a data drill-down chart. The database against which the query is run is WoodView HSQL, so you will need to add database HSQL JDBC driver (`hsqldb.jar`) to your classpath. The driver is located in the `<ERESInstall>/WEB-INF/lib` directory. Please note that you have to have ERES server running to run this example.

```
Component doDynamicDataDrillDownChart(Applet applet) {

    //Do not use EspressoManager
    QbChart.setEspressoManagerUsed(false);

    // Begin Code : Creating Chart 1 - the Root Chart

    DBInfo rootInfo = new DBInfo(
        "jdbc:hsqldb:woodview",
        "org.hsqldb.jdbcDriver",
        "sa",
        "",
        "select Categories.CategoryName, Products.ProductName,
        Products.UnitsInStock from Categories, Products where Categories.CategoryID
        = Products.CategoryID order by Categories.CategoryName;");

    ColInfo rootColInfo = new ColInfo();
    rootColInfo.category = 0;
    rootColInfo.value = 2;
    QbChart chart = new QbChart(applet, QbChart.VIEW2D, QbChart.BAR, rootInfo,
        rootColInfo,
        null);

    // End Code : Creating Chart 1 - the Root Chart

    try {

        // Begin Code : Creating Chart 2 - the sub Chart

        IDrillDown chartDrillDown = chart.getDrillDown();
        chartDrillDown.setAggregateOperator(IDrillDown.SUM); // Set the
        Aggregation
        chartDrillDown.setDynamicDrillDown(true); // Turn on dynamic data drill-
        down
        chartDrillDown.setDrillName("DynamicDrillDownChart"); // Set the drill
        template name

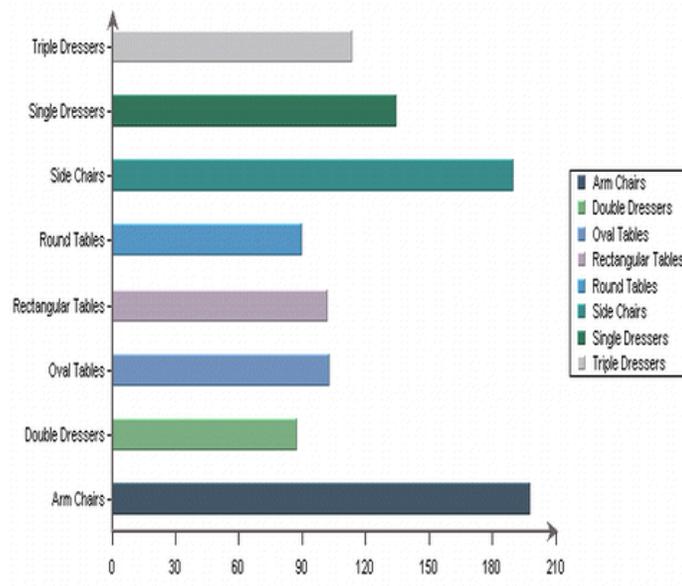
    } catch (Exception ex) {
        ex.printStackTrace();
    }

    // End Code : Creating Chart 2 - the sub Chart

    return chart;
}
```

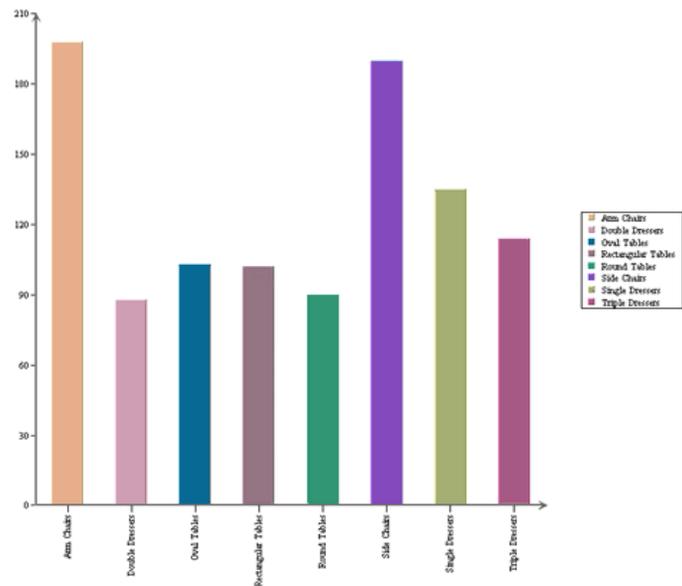
Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/DynamicDataDrillDownChart.zip> ]

When the above code is run as a Java Web Start Application, it will produce a top-level chart shown below:



Generated chart

Depending on the link clicked, you will see a chart similar to the one shown below:



Generated chart

Please note that this is a default chart that is generated without formatting of any kind.

## 8.D.15. Adding a Chart to a Report

A chart can be programmatically added to the report in one of three ways :

- Creating a chart template in ERES Designer and adding the template to the report component;
- Creating a chart (that uses report data as its data source) using the API and adding that chart to the report component;
- Creating a `QbChart` object using the API and adding that object to the report component;

### 8.D.15.1. Adding a Chart Template

To add a chart, you will need to define a `ReportChartObject` object. Within the `ReportChartObject` object, you will have to pass in the location of the template (TPL) file that you are using. You can also specify the dimensions and alignment. If the report is exported to a static format (i.e. DHTML), you can also specify the image type (JPEG, GIF, PNG, etc) of the exported chart. The default alignment is the center of the cell and default image type is JPEG. You can then add the object to the desired location in the report.



#### Note

You can only use a TPL template file and the chart uses report data (even if the template uses an independent data source) as its data source.

The following example, which can be run as a Java Web Start Application, shows how to add an existing chart template to a report:

```
ReportChartObject chartObject = new ReportChartObject();
String chartLocation0 = new String("../templates/AddingChartTemplate.tpl");
chartObject.setText(chartLocation0);
chartObject.setWidth(7.40);
chartObject.setHeight(4.90);
chartObject.setY(0.10);
chartObject.setImageType(QbReport.PNG);
report.getTable().getFooter().addData(chartObject);
```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/AddingChartTemplate.zip> ]

Exported Results [ <http://data.quadbase.com/Docs70/help/manual/code/export/AddingChartTemplate.pdf> ]

### 8.D.15.2. Adding a Chart that uses Report Data

To add a chart using this method, you must create the chart object using the `ChartObject` class. A `ChartObject` object contains the chart properties (i.e. the chart mapping, the dimension and chart type). Then a `ReportChartObject` object is instantiated/created, and the newly created `ChartObject` object is passed to it using the `setChart(IChart)` method. The `ReportChartObject` is then finally added to the desired location in the report.

The following example, which can be run as an applet or application, shows how to add a chart, that uses report data as its data source, to a report:

```
ColumnInfo chartColumnInfo = new ColumnInfo();
chartColumnInfo.category = 0;
chartColumnInfo.value = 1;

ChartObject chartObj = new ChartObject(report, // QbReport

    ChartObject.VIEW2D, // Dimension
    ChartObject.BAR, // Chart Type
    false, // Do Transpose
    null, // Transpose Columns
    chartColumnInfo, // Chart Mapping
    "AddingChart.tpl"); // Template

ReportChartObject chartObject = new ReportChartObject();
chartObject.setChart(chartObj);
chartObject.setWidth(7.40);
chartObject.setHeight(4.90);
chartObject.setY(0.10);
```

```
chartObject.setImageType(QbReport.PNG);
report.getTable().getFooter().addData(chartObject);
```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/AddingReportDataChart.zip> ]

Exported Results [ <http://data.quadbase.com/Docs70/help/manual/code/export/AddingReportDataChart.pdf> ]

### 8.D.15.3. Adding a Chart that uses an Independent Data Source

To add a chart using this method, you must create the chart object using the `QbChart` class. A `QbChart` object contains the chart properties (i.e., the chart mapping, the dimension and chart type). Then a `ReportChartObject` object is instantiated/created, and the newly created `QbChart` object is passed to it using the `setChart(IChart)` method. The `ReportChartObject` is then finally added to the desired location in the report.

This scenario differs from the one in the previous section in that the data for the chart is not coming from the report. The chart data is independent of the report's.

The following example, which can be run as a Java Web Start Application, shows how to add a chart, that uses report data as its data source, to a report:

```
ColInfo chartColInfo = new ColInfo();
chartColInfo.category = 0;
chartColInfo.value = 1;

// Data Source for chart
String chartQuery = "SELECT ProductName, Sum(Order_Details.Quantity) " +
    "FROM Products, Order_Details " +
    "WHERE (Order_Details.ProductID = Products.ProductID) " +
    "GROUP BY Products.ProductName " +
    "ORDER BY Sum(Order_Details.Quantity) DESC";

DBInfo chartDatabaseInfo = new DBInfo("jdbc:hsqldb:database/
woodview", "org.hsqldb.jdbcDriver",
    "sa", "", chartQuery);

QbChart chartObj = new QbChart((Applet)null, QbChart.VIEW2D, QbChart.COL,
    chartDatabaseInfo,
    false, chartColInfo, "AddingChartTemplate.tpl");

// Add ChartObject to report
ReportChartObject chartObject = new ReportChartObject();
chartObject.setChart(chartObj);
chartObject.setWidth(7.40);
chartObject.setHeight(4.90);
chartObject.setReportDataUsed(false);
chartObject.setY(0.10);
chartObject.setImageType(QbReport.PNG);
report.getTable().getFooter().addData(chartObject);
```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/AddingIndependentDataChart.zip> ]

Exported Results [ <http://data.quadbase.com/Docs70/help/manual/code/export/AddingIndependentDataChart.pdf> ]

---

# Chapter 9. Configuration

## 9.1. Using Other Databases

As discussed in Section 1.3.2.1 - The ERES Database, EspressoReport ES uses a database as a persistent storage mechanism for users, groups, permissions, data sources, and published reports and charts. By default, EspressoReport ES uses an HSQL database that comes with the installation. However, since this database is generally inappropriate for production environments, users can easily configure ERES to use a different one.

### 9.1.1. Create Table Scripts

The first step in configuring ERES to run with another database is to setup the tables and fields that are used to store application information. Several scripts for different databases are available in the `<ERESInstallDir>/data` directory. The scripts contain the create table statements for the database.

In addition to database specific scripts, a generic script is also provided that can be modified to run with other databases. The `quadbasedb_jdbc.sql` file contains default scripts that are used to create the HSQL database.

#### 9.1.1.1. Database Specific Notes

For each of the supported databases, there can be a couple additional configuration steps that needs to be taken to correctly setup the tables.

##### 9.1.1.1.1. DB2

The script to create the tables for DB2 is named `quadbasedb_db2.sql`. For DB2, you will need to first create a TABLESPACE for the new tables. To do so, use the following steps from the command line:

1. connect to your database using db2admin using this command:

```
db2 connect to <DBNAME> user <DB2USER> using <Password>
```

Be sure to substitute your database username and password when making the connection.

2. Create buffer pool with the command below:

```
db2 CREATE BUFFERPOOL BP32k SIZE 100 PAGESIZE 32k
```

3. Create tablespace with the command below:

```
db2 "CREATE REGULAR TABLESPACE DATASPACE1 PAGESIZE 32 K MANAGED BY
DATABASE USING ( FILE 'c:\db2\db2data\dataspace1_01.dbf' 1000 )
EXTENTSIZE 4 OVERHEAD 10.5 PREFETCHSIZE 4 TRANSFERRATE 0.33 BUFFERPOOL
BP32k"
```

##### 9.1.1.1.2. MS SQL Server

The script to create the tables for Microsoft SQL Server is named `quadbasedb_sqlserver.sql`. There is no additional setup needed to run the script for this database. However, the database connection cannot be established to SQL Server with the default settings. The requirements may vary with different SQL Server versions.

###### MS SQL Server 2005

Download the latest MS SQL Server 2005 JDBC Driver from: [msdn.microsoft.com \[ https://learn.microsoft.com/en-us/sql/connect/jdbc/download-microsoft-jdbc-driver-for-sql-server?view=sql-server-ver16 \]](https://learn.microsoft.com/en-us/sql/connect/jdbc/download-microsoft-jdbc-driver-for-sql-server?view=sql-server-ver16)

Unpack the driver and copy the `sqljdbc.jar` file into the `<ERES>/WEB-INF/lib` directory.

Recommended *Admin Console* entries (see Section 9.1.2 - Specifying the Database Connection):

```
Database URL: jdbc:sqlserver://hostname:port;databaseName=<databaseName>;
```

```
Database Driver: com.microsoft.sqlserver.jdbc.SQLServerDriver
```

ERES will connect to the SQL server instance using TCP/IP protocol, so you will have to determine the TCP port number of your SQL Server instance. To do that, launch *SQL Server Configuration Manager*, expand the *SQL Server 2005 Network Configuration* node and click *Protocols for <YourSQLServerInstanceName>* (for example *Protocols for MSSQLSERVER*). Double click on the *TCP/IP* protocol. First of all, make sure that the TCP/IP protocol is enabled. Look at the *Enabled* option in the *Protocol* tab, it should be set to *Yes*, if it is not, please do so. Then look at the *Listen All* option, there are two possibilities:

1. **Listen All is set to Yes** (default) : Go to *IP Addresses* tab and scroll down to the *IP All* section. The port number you are looking for is below the *IP All* section in the *TCP Dynamic Port* option.
2. **Listen All is set to No** : Go to *IP Addresses* tab and find the section that has the *IP Address* option set to the IP address you will be connecting to. The port number can be found one row below the particular IP address option (*TCP Dynamic Ports* option).

ERES connects to the SQL database using SQL Server Authentication mode, so you have to have this function enabled in your MS SQL Server. You can do that in MS SQL Server Management Studio. Right click on the server name in *Object Explorer* and click *Properties*. Go to the *Security* page and select the *SQL Server and Windows Authentication mode* option. Also the user that you will use to connect to the SQL Server has to have the SQL Server authentication mode set. Be sure that the user has sufficient right to use the database.

### MS SQL Server 2000

Download the latest tar version of MS SQL Server 2000 JDBC Driver from: [download.microsoft.com \[ https://learn.microsoft.com/en-us/sql/connect/jdbc/download-microsoft-jdbc-driver-for-sql-server?view=sql-server-ver16 \]](https://learn.microsoft.com/en-us/sql/connect/jdbc/download-microsoft-jdbc-driver-for-sql-server?view=sql-server-ver16)

Unpack the `mssqlserver.tar` file. Go to the unpacked directory and unpack the `msjdbc.tar` file. Go to the `msjdbc/lib` directory. You should see three `jar` files in the `lib` folder. Copy them into the `<ERES>/WEB-INF/lib` directory.

Recommended *Admin Console* entries (see Section 9.1.2 - Specifying the Database Connection):

```
Database URL: jdbc:microsoft:sqlserver://
hostname:port;databaseName=<databaseName>;
```

```
Database Driver: com.microsoft.jdbc.sqlserver.SQLServerDriver
```

#### 9.1.1.1.3. MySQL

The script to create the tables for MySQL is named `quadbasedb_mysql.sql`. There is no additional setup needed to run the script for this database.

Please make sure that your MySQL database uses case-insensitive identifiers (see MySQL Identifier Case Sensitivity [ <https://dev.mysql.com/doc/refman/8.0/en/identifier-case-sensitivity.html> ]).

#### 9.1.1.1.4. Oracle

The script to create the tables for Oracle is named `quadbasedb_oracle.sql`. There is no additional setup needed to run the script for this database.

## 9.1.2. Specifying the Database Connection

Once you have successfully created the tables in your database, you can configure ERES to use that database for storage. To setup the database connection, log in as *Admin* and enter the *Admin Console*. The database information can be entered under Setting Info → ERES Repository.

Replace the *URL*, *driver*, *username*, and *password* with values of your database. ERES uses JDBC to connect to the database, so you will need to make sure to enter the correct URL and JDBC driver class for the connection.

Once you finish making the changes, save on Save Settings and restart the ERES Server. Note that you may need to copy the JDBC driver for your database (typically one or several Jar files) to the `<ERESInstallDir>/WEB-INF/lib` directory before restarting the ERES Server. For application servers other than Tomcat, please refer to the instructions in Section 9.3 - Using Other Application Servers.

You can also specify a connection to the ERES database using JNDI. To do so, you must first edit `<ERESInstallDir>/WEB-INF/web.xml` and uncomment the JNDI tag (remove the `<!--` and `-->` tags). You can also change the JNDI reference name (in the `<res-ref-name>` tag) from `jdbc/ERESdb` to another name. Then login as *Admin*, enter *Admin Console* and pass in the JNDI details (such as JNDI context factory, JNDI provider URL and JNDI name) under Setting Info → ERES Repository. The values can be entered after the JNDI radio button is selected. Note that depending on the application server and the mapping used, the JNDI context factory and JNDI provider URL may not be necessary. For example, if ERES is installed on a Tomcat or WebSphere Application Server, connecting to the JNDI server only requires supplying the JNDI name. For Oracle 10g, you have the choice of either omitting the context factory and provider URL to use the default environment, or you can specify one of context factories they provide: `oracle.j2ee.rmi.RMIInitialContextFactory`, `oracle.j2ee.naming.ApplicationClientInitialContextFactory`, or `oracle.j2ee.iiop.IIOPInitialContextFactory`. Due to the wide range of application servers and JNDI implementations, please see the documentation for your JNDI server for configuration options and details. When the JNDI name is specified, the database information (URL, driver, username, password) is ignored.

With the connection information set, restart the server. The ERES server should start connected to the new database. Note that this new database will be blank, so any information about old users, privileges, or reports and charts will not be available. In addition, several key pieces of information will need to be manually setup in order to successfully deploy and run reports in ERES with the new database.

**Web Root:** You will need to enter the correct Web Root for your application server/servlet container. This information is supplied in the *Admin Console*. For more information, see Section 1.4.1.2 - Setting Info.

**Servlet Context:** You may need to re-set the servlet context information. For more on this, please see Section 1.4.1.2 - Setting Info.

**URL Mapping:** You will need to add a virtual directory in Organizer that maps the Web path to your ERES installation. For more information, see Section 2.1.5 - URL Mapping.

### 9.1.3. Migrating the ERES Database

Sometimes it may be necessary to migrate the ERES database from one platform to another. For example, initial development on an implementation may be done using ERES in default configuration with the HSQL database. However, since HSQL is not recommended for production environments, users may need to migrate data to another database. To accommodate this, ERES provides a database migration utility for all supported databases.



#### Warning

Since version 6.3, this tool can be used for database migration only. Database upgrade using this tool is no longer supported. Please use `dbupgrade` (`DBUpgradeGUI` or `UpgradeAppl`) application instead (Section 1.3.2.1.3 - Upgrading ERES Database from previous version of ERES).

Before starting the database migration process, you will have to prepare the destination database, which must contain empty ERES 6.6 tables. To create the tables, run the appropriate SQL script from the `<ERESInstallDir>/data` directory. To learn more about the scripts, see the Section 9.1 - Using Other Databases chapter.

Please make sure that both the source and the destination databases belong to ERES version 6.6. If not, use the `dbupgrade` application to fix this problem.

The program for the utility is in `<ERESInstallDir>/data/converter` directory. It is a Java application that is run from the command line (the source code is also provided). To run the converter, modify `FromDB.properties` file to use the JDBC connection information for your source database. Then modify `ToDB.properties` file to use the JDBC connection information for your destination database (sample connection files are provided for different databases in the directories under `<ERESInstallDir>/data/converter`).

At the command line, navigate to `<ERESInstallDir>/data/converter` directory and execute the following command:

```
java -classpath ".;driver1;driver2" CopyERESDB
```

For example:

```
java -classpath ".;hsqldb.jar;jdbc_mysql.jar" CopyERESDB
```

Within the `-classpath` argument, you will need to specify the archives/classes that contain the JDBC drivers for the source and destination databases.

Running this program will copy all data from the source (FROM) ERES database tables to the destination (TO) ERES database tables.

Once the conversion is complete, you need to modify the database connection information in the *Admin Console* as described in Section 9.1.2 - Specifying the Database Connection, before restarting the server.

Please note that ERES log records can not be transferred during the database migration.

## 9.2. Integrating Existing Users/Groups

By default, the ERES database provides persistent storage for users, groups, and their relationships. However, many users already have a set of users and groups, as well as persistent storage mechanism in place. In these cases, duplicating users and groups in the new reporting environment is inefficient, time-consuming, and requires extra administration overhead.

To prevent these issues, ERES provides three different ways on how users can integrate their existing user/group framework into the reporting system. The first option is to retrieve users and groups from an LDAP server. This connection and setup is detailed in Section 1.4.1.4 - LDAP Settings. The second option is to retrieve users and groups from an existing database and the third option is to use the `UserSecurityProvider` interface which provides an open API allowing users to retrieve/implement users and groups from any source.

### 9.2.1. Using a Database

If your existing users are in a database, ERES provides a simple implementation of the `UserSecurityProvider` that can be configured using a properties file. This implementation allows you to supply the connection information for the database as well as SQL statements for all the user/group related actions. Note that this implementation will not work in all circumstances. If your schema cannot retrieve/set the proper information with a single SQL statement, you may need to provide your own implementation of `UserSecurityProvider`.

The properties file is named `user-database.properties`, and should be placed in the `<ERESInstallDir>/WEB-INF/orguserdb` directory. A sample properties file is shown below:

```
#####
#Whether to use custom database for user/group
custom.user.database=true
#####

#####
#If custom.user.database=true, then use these information to
#connect to the database.
custom.user.database.url=jdbc:hsqldb:C:/ERES/data/quadbasedb
custom.user.database.jdbcDriver=org.hsqldb.jdbcDriver
custom.user.database.login=sa
custom.user.database.password=
#####

#####
#Whether each of the methods in UserSecurityProvider
#is enabled.
#-----
getUsers.enable=true
```

## Configuration

---

```
getUsers.sql=Select username, fullname, email, password, user_role, id,
  securitylevel From users Order
  By username
#-----
getGroups.enable=true
getGroups.sql=Select name, description, id, securitylevel From groups Order
  By name
#-----
getGroupRelations.enable=true
getGroupRelations.sql=SELECT parent_g.name, child_g.name FROM
  group_relations r, groups parent_g, groups
  child_g WHERE r.parent_id = parent_g.id AND r.child_id = child_g.id
#-----
getGroupUserRelations.enable=true
getGroupUserRelations.sql=Select g.name, u.username From
  group_user_relations gr, groups g, users u
Where gr.group_id=g.id And gr.user_id=u.id Order By g.name
#-----
getUserParents.enable=true
getUserParents.sql=SELECT r.group_id, g.name FROM group_user_relations r,
  groups g WHERE r.group_id =
  g.id AND r.user_id=?
#-----
getGroupParents.enable=true
getGroupParents.sql=Select parent_id From group_relations Where child_id=?
#-----
deleteUser.enable=true
deleteUser.sql=Delete From users Where username=?
#-----
deleteGroup.enable=true
deleteGroup.sql=Delete From groups Where name=?
#-----
createUser.enable=true
createUser.sql=Insert Into users (id, username, fullname, email, password,
  user_role, securitylevel)
Values (?, ?, ?, ?, ?, ?, ?)
#-----
createGroup.enable=true
createGroup.sql1=Insert Into groups (id, name, description, securitylevel)
  Values (?, ?, ?, ?)
createGroup.sql2=Delete From group_relations Where parent_id=?
createGroup.sql3=Delete From group_user_relations Where group_id=?
createGroup.sql4=Insert Into group_relations (parent_id, child_id) Values
  (?, ?)
#-----
updateUser.enable=true
updateUser.sql=Update users Set username=?, fullname=?, email=?, password=?,
  user_role=?,
  securitylevel=? Where username=?
#-----
updateGroup.enable=true
updateGroup.sql1=Update groups Set name=?, description=?, securitylevel=?
  Where name=?
updateGroup.sql2=Delete From group_relations Where parent_id=?
updateGroup.sql3=Delete From group_user_relations Where group_id=?
updateGroup.sql4=Insert Into group_relations (parent_id, child_id) Values
  (?, ?)
#-----
```

```

addGroupUserRelation.enable=true
addGroupUserRelation.sql=Insert Into group_user_relations (group_id,
  user_id) Values (?, ?)
#-----
setSecurityLevels.enable=true
setSecurityLevels.sql1=Update users Set securitylevel=? Where username=?
setSecurityLevels.sql2=Update groups Set securitylevel=? Where name=?
#-----
resetSecurityLevel.enable=true
#-----
getGroupsBySecurityLevel.enable=true
getGroupsBySecurityLevel.sql=Select name From Groups Where securitylevel=?
#-----
getUsersBySecurityLevel.enable=true
getUsersBySecurityLevel.sql=Select username From users Where securitylevel=?
#-----
#####

```

The question marks in the SQL statements will be supplied by the ERES engine when it runs queries. For any method that isn't enabled, ERES will save that information in the ERES database. Depending on your configuration, saving some information in the ERES database and some in your own may not be able to work correctly.

## 9.2.2. Implementing UserSecurityProvider

For maximum flexibility, ERES allows users to provide their own implementation of users and groups. The UserSecurityProvider interface provides open APIs into all the methods that set/retrieve user and group information. Implementing UserSecurityProvider allows you to retrieve user/group information from any source, and allows you to integrate your user database if the schema prevents you from retrieving the correct information with a single query. Below is a sample UserSecurityProvider implementation:

```

import java.util.*;

import quadbase.reportorganizer.data.*;
import quadbase.reportorganizer.ext.*;

/**
 * A sample implementation of UserSecurityProvider that provides hard-coded
 * initialized users/groups/security that does not have any data persistent
 * mechanism.
 */
public class MyUserSecurityProvider implements UserSecurityProvider {

    User[] users;
    Group[] groups;
    GroupRelation[] gr;
    GroupUserRelation[] gur;

    /**
     * Initializes the provider with dummy users/groups/security
     */
    public MyUserSecurityProvider() {
        users = new User[6];
        users[0] = createUser("admin");
        users[1] = createUser("jason");
        users[2] = createUser("ricky");
        users[3] = createUser("jordan");
        users[4] = createUser("oneal");
        users[5] = createUser("jerry");
    }
}

```

```

groups = new Group[6];
groups[0] = createGroup("nba");
groups[1] = createGroup("sports");
groups[2] = createGroup("testers");
groups[3] = createGroup("managers");
groups[4] = createGroup("marketing");
groups[5] = createGroup("engineers");

gr = new GroupRelation[3];
gr[0] = new GroupRelation("sports", "nba");
gr[1] = new GroupRelation("managers", "marketing");
gr[2] = new GroupRelation("engineers", "testers");

gur = new GroupUserRelation[6];
gur[0] = new GroupUserRelation("managers", "jason");
gur[1] = new GroupUserRelation("nba", "jordan");
gur[2] = new GroupUserRelation("marketing", "ricky");
gur[3] = new GroupUserRelation("nba", "oneal");
gur[4] = new GroupUserRelation("testers", "jerry");
gur[5] = new GroupUserRelation("testers", "jason");
}

public User[] getUsers() throws Exception {
    return users; }

public Group[] getGroups() throws Exception {
    return groups; }

public GroupRelation[] getGroupRelations() throws Exception {
    return gr; }

public GroupUserRelation[] getGroupUserRelations() throws Exception {
    return gur; }

public String[] getUserParents(String username) throws Exception {
    Vector v = new Vector();
    for (int i = 0; i < gur.length; i++)
        if (gur[i].getUserName().equals(username))
            v.add(gur[i].getGroupName()); return
toStringArr(v); }

public String[] getGroupParents(String groupname) throws Exception {
    Vector v = new Vector();
    for (int i = 0; i < gr.length; i++)
        if (gr[i].getChildName().equals(groupname))
            v.add(gr[i].getParentName()); return
toStringArr(v); }

public void deleteUser(String username) throws Exception {
    if (users.length == 0) return;
    User[] users2 = new User[users.length-1];
    int j = 0;
    for (int i = 0; i < users.length; i++) {
        if (j == users.length-1 && !
users[i].getName().equals(username))
            return; if (!users[i].getName().equals(username)) {

```

```

        users2[j] = users[i];
        j += 1; } }
    users = users2; }

/**
 * Delete a Group. Also deletes the group relationships and
 * group-user relationships
 * that references this GROUPNAME.
 * @param groupname the group name
 */
public void deleteGroup(String groupname) throws Exception {
    if (groups.length == 0) return;
    Group[] groups2 = new Group[groups.length-1];
    int j = 0;
    for (int i = 0; i < groups.length; i++) {
        if (j == groups.length-1 && !
groups[i].getName().equals(groupname))
            return; if (!groups[i].getName().equals(groupname))
{
            groups2[j] = groups[i];
            j += 1; } }
    groups = groups2;

    Vector v = new Vector();
    for (int i = 0; i < gr.length; i++) {
        if (gr[i].getParentName().equals(groupname) ||
            gr[i].getChildName().equals(groupname)) continue;
v.add(gr[i]); }
    GroupRelation[] gr2 = new GroupRelation[v.size()];
    for (int i = 0; i < gr2.length; i++)
        gr2[i] = (GroupRelation) v.get(i); gr = gr2;

    v = new Vector();
    for (int i = 0; i < gur.length; i++) {
        if (gur[i].getGroupName().equals(groupname)) continue;
        v.add(gur[i]); }
    GroupUserRelation[] gur2 = new GroupUserRelation[v.size()];
    for (int i = 0; i < gur2.length; i++)
        gur2[i] = (GroupUserRelation) v.get(i); gur = gur2; }

public void createUser(User user) throws Exception {
    User[] users2 = new User[users.length+1];
    System.arraycopy(users, 0, users2, 0, users.length);
    users2[users2.length-1] = user;
    users = users2; }

public void updateUser(User user) throws Exception {
    for (int i = 0; i < users.length; i++) {
        if (users[i].getName().equals(user.getName())) {
            users[i] = user;
            return; } } }

/**
 * Create a new Group.
 * @param parentGroup create this group
 * @param childUsers set the group to have these child users
 * @param childGroups set the group to have these child groups
 */

```

```

    public void createGroup(Group parentGroup, User[] childUsers, Group[]
childGroups) throws Exception {
        addGroup(parentGroup);

        if (childUsers != null && childUsers.length > 0) {
            for (int i = 0; i < childUsers.length; i++) {
                addGroupUserRelation (parentGroup.getName(),
childUsers[i].getName()); } }

        if (childGroups != null && childGroups.length > 0) {
            GroupRelation[] gr2 = new GroupRelation[gr.length
+childGroups.length];
            System.arraycopy(gr, 0, gr2, 0, gr.length);
            int j = 0;
            for (int i = gr.length; i < gr2.length; i++) {
                gr2[i] = new GroupRelation(parentGroup.getName(),
childGroups[j].getName());
                j++; }
            gr = gr2; } }

/**
 * Update a Group. This simple implementation simply calls
 * deleteGroup(parentGroup.getName()) and calls
 * createGroup(parentGroup, childUsers, childGroups);
 *
 * @param parentGroup update this group
 * @param childUsers set the group to have these child users
 * @param childGroups set the group to have these child groups
 */
    public void updateGroup(Group parentGroup, User[] childUsers, Group[]
childGroups) throws Exception {
        deleteGroup(parentGroup.getName());
        createGroup(parentGroup, childUsers, childGroups); }

    public void addGroupUserRelation(String groupname, String
username) throws Exception {
        GroupUserRelation[] gur2 = new GroupUserRelation[gur.length+1];
        System.arraycopy(gur, 0, gur2, 0, gur.length);
        gur2[gur2.length-1] = new GroupUserRelation(groupname,
username);
        gur = gur2; }

    // SECURITY
    LEVEL //////////////////////////////////////

/**
 * Sets the security level(s) for a group or user of the specified
name(s)
 * Also sets the security level(s) for group or user that is not in
NAME
 * but with the same SECLEVEL to -1 (Can simply call
 * resetSecurityLevel(secLevel) in the first line of this method
 * to achieve this).
 * @param secLevel the result security level number
 * @param name an array of name(s) to change
 * @param isGroup whether the name[i] is a group (isGroup[i]==true)
 * or a user (isGroup[i]==false).
 */

```

```

    public void setSecurityLevels(int secLevel, String[] name, boolean[]
isGroup) throws Exception {
        resetSecurityLevel(secLevel);
        for (int i = 0; i < name.length; i++) {
            if (isGroup[i]) {
                for (int j = 0; j < groups.length; j++)
                    if (groups[j].getName().equals(name[i])) {
                        groups[j].setSecurityLevel(secLevel);
                        break; } } else {
                for (int j = 0; j < users.length; j++)
                    if (users[j].getName().equals(name[i])) {
                        users[j].setSecurityLevel(secLevel);
                        break; } } } }

/**
 * Resets (set to -1) the security level number of all groups
 * and users whose security level number is SECLEVEL.
 * @param secLevel the security level number of the group/user
 */
    public void resetSecurityLevel(int secLevel) throws Exception {
        for (int i = 0; i < users.length; i++)
            if (users[i].getSecurityLevel() == secLevel)
                users[i].setSecurityLevel(-1); for (int i = 0; i <
groups.length; i++)
            if (groups[i].getSecurityLevel() == secLevel)
                groups[i].setSecurityLevel(-1); }

/**
 * @param secLevel the security level number of the group
 * @return the Group names of all groups with SECLEVEL
 */
    public String[] getGroupsBySecurityLevel(int secLevel) throws
Exception {
        Vector v = new Vector();
        for (int i = 0; i < groups.length; i++)
            if (secLevel == groups[i].getSecurityLevel())
                v.add(groups[i].getName()); String[] result = new
String[v.size()];
        for (int i = 0; i < result.length; i++)
            result[i] = (String) v.get(i); return result; }

/**
 * @param secLevel the security level number of the user
 * @return the User names of all users with SECLEVEL
 */
    public String[] getUsersBySecurityLevel(int secLevel) throws Exception
{
        Vector v = new Vector();
        for (int i = 0; i < users.length; i++)
            if (secLevel == users[i].getSecurityLevel())
                v.add(users[i].getName()); String[] result = new
String[v.size()];
        for (int i = 0; i < result.length; i++)
            result[i] = (String) v.get(i); return result; }

    User createUser(String name) {

```

```

if(name.equalsIgnoreCase("admin")){
return new User(name, //name
name + " full name", //full name
name, //password
name + "@example.com", //email
User.ADMIN, //role
-1); //security level
}else{
return new User(name, //name
name + " full name", //full name
name, //password
name + "@example.com", //email
User.DESIGNER, //role
-1); //security level
}
}

Group createGroup(String name) {
return new Group(name, //name
name + " desc", //description
-1); //security level
}

void addGroup(Group g) {
Group[] g2 = new Group[groups.length+1];
System.arraycopy(groups, 0, g2, 0, groups.length);
g2[g2.length-1] = g;
groups = g2; }

static String[] toStringArr(Vector v) {
String[] p = new String[v.size()];
for (int i = 0; i < p.length; i++)
p[i] = (String) v.get(i); return p; }
}

```

In the previous sample, the users and groups are defined as simple arrays in the code. As with the database properties file, any method that you do not implement in the `UserSecurityProvider` interface will use the default ERES database as the storage mechanism.

### 9.2.2.1. Deploying UserSecurityProvider

The `UserSecurityProvider` class can be set as a server option in ERES. You can set this in one of two places. The first option is the *Admin Console*. You can specify the class in the *Server Options* tab. For more information about server configuration options, see Section 1.4.1.3 - Server Options. You can also specify the class by modifying the `QB.properties` under `<ERESInstallDir>/WEB-INF/classes`. However, editing configuration files directly is not recommended and should be done only in case when the ERES server cannot be started because incorrect values have been provided through the *Admin Console*. If you still want to edit the files manually: edit `user-database.properties` to use the "users/groups" other than ERES database, `QB.properties` to set ERES database not only "users/groups", but also ERES setting, report/chart/etc/file paths, schedules, email settings, extension classes. It has higher priority than the database defined by `user-database.properties`. Compiled, `MyUserSecurityProvider.class`, if used, should be located in `<ERESInstallDir>/WEB-INF/classes` directory.

## 9.3. Using Other Application Servers

EspressReport ES deploys as servlet and JSP collection in an application server. As detailed in Section 1.3.1 - Installing ERES, ERES comes bundled with the Tomcat servlet container. Users have an option to install Tomcat

(with ERES automatically deployed) when running the installer. This option is recommended for new users and evaluators because ERES will work “out-of-the-box” in this configuration.

However, ERES can also be deployed on most popular application servers. This chapter provides step-by-step setup instructions for Tomcat, Resin, WebLogic, WebSphere, JBoss, Oracle, and Sun servers.

Throughout this chapter, the following naming conventions will be used.

- <ERES\_INSTALL\_DIR> :** This is the location where ERES is installed.
- <ERES\_DEPLOY\_DIR> :** This is the final deployment directory of ERES. It might not be same as your installation directory (<ERES\_INSTALL\_DIR>).
- <ERES\_CONTEXT> :** This is the context path used to access ERES web application components.
- <ERES\_SERVLET\_CONTEXT> :** This is the servlet context of ERES. ERES uses servlets for many of the back-end functions. There is a `web.xml` file under the `<ERES_INSTALL_DIR>/WEB-INF` directory that maps all servlet classes packaged with ERES. By default, every servlet is mapped to `/servlet/<servlet name>`. After deploying ERES, you may need to change the servlet context.
- <server> :** This is your application/web server name. It can be the name of the server machine or IP address.
- <port> :** The port number on which your application/web server is listening.
- ERES Start Page :** This is the main/home page for ERES. It should be accessed by the following URL: `http://<server>:<port>/<ERES_CONTEXT>/index.jsp`.

In addition, instructions in this chapter refer to starting and stopping the ERES server, as well as features/options in the *Admin console*. If you are using *AUTOSTART* feature, you need to set correct port in *Server Options - Auto-Start* in *Admin Console* page. For details about these actions, please see Section 1.3.2.2 - Starting the Server and Section 1.4 - Administration.

As the result of moving to a different application server, the `<ERES_INSTALL_DIR>` location may move or the name of the ERES context may be changed. When this happens, you will need to modify the URL mapping in Organizer in order to create/modify/run reports and charts. For more information about setting URL mapping, see Section 2.1.5 - URL Mapping.



### Note

These instructions assume you are running ERES with the default (HSQL) database. If you are using a different database, you need to add the JDBC driver classes for your database to your servlet container's classpath. All JAR files are under `<ERES_INSTALL_DIR>/WEB-INF/lib`, which is used by the servlet container to load classes required by the ERES web application. Therefore, if you are adding JDBC driver that is needed by the ERES web application, you can simply copy the JAR file to `<ERES_INSTALL_DIR>/WEB-INF/lib`.

You can also deploy ERES as a war file. To make an ERES war file, you must:

- Go to `<ERES_INSTALL_DIR>/WEB-INF` directory and edit `web.xml`. Remove the two instances of the `<!-- WAR Deployment and WAR Deployment -->` references and then save the file.
- Open command prompt and `cd` to `<ERES_INSTALL_DIR>` directory. Then run following command (make sure `jdk/bin` is in your `PATH`):

```
jar -cvf0 ERES.war *
```

- The `ERES.war` file is now available under the `<ERES_INSTALL_DIR>` and can be deployed to an application server.

Please note that when deploying the war file, the `<ERES_INSTALL_DIR>` is still being used to store all templates and user files, so it must be valid and accessible. Also, the instructions, given above, must be carried out again when an update is performed on the `<ERES_INSTALL_DIR>` directory and the newly created war file should be deployed again. Note that certain application servers requires additional files (such as JBoss needing a `jboss-web.xml` file) before the war file can be successfully deployed.

### 9.3.1. Tomcat 4.1/5.x/6.0.x/7.0.x

The following instructions show how to deploy EspressoReport ES under Tomcat 4.1/5.x/6.0.x/7.0.x. The instructions assume that Tomcat has been installed on the system. The location of the Tomcat installation is referenced as `<TOMCAT_INSTALL_DIR>`.

For reference, please note that Tomcat's default port is 8080, the default working directory is `<TOMCAT_INSTALL_DIR>/bin` and the default Web root is `<TOMCAT_INSTALL_DIR>/webapps/ROOT`.

1. Modify the `<TOMCAT_INSTALL_DIR>/config/server.xml` file and add the following entry: `<Context path="/ERES" docBase="<ERES_INSTALL_DIR>" debug="0"/>`. Path is the context path of ERES web application and docBase is the absolute path or relative path of the ERES installation. Please substitute `<ERES_INSTALL_DIR>` with your actual path.



#### Note

Please use double quotes.

2. Now start Tomcat from `<TOMCAT_INSTALL_DIR>/bin/startup.bat / .sh`.
3. With Tomcat running, go to ERES Start page `http://<server>:<port>/<ERES_CONTEXT>/index.jsp` With the above configuration, `ERES_CONTEXT` is `ERES`. Start the ERES Server.
4. From the ERES Start page, login as administrator and launch the *Admin Console*.
5. From *Admin Console* page, go to *Setting Info* tab and change the ERES Servlet Context entry to `/ERES/servlet`. This change requires you to restart the ERES Server. You can also change the webroot entry to `<TOMCAT_INSTALL_DIR>/webapps/ROOT`, but it is not necessary.
6. Stop and restart the ERES Server.

### 9.3.2. Resin

#### 9.3.2.1. Resin 3.x/4.x

Deploying EspressoReport ES to Resin 3.x/4.x can be accomplished using the instructions provided below. The location of the Resin installation is referenced as `<RESIN_INSTALL_DIR>`.

For reference, please note that Resin's default port is 8080, the default working directory is `<RESIN_INSTALL_DIR>` and the default Web root is `<RESIN_INSTALL_DIR>/docs`.

1. You may need to move `tools.jar` file from your Java development kit to the `<RESIN_INSTALL_DIR>/lib` directory (for JSP compilation) or add a library path entry for the file.
2. Modify `<RESIN_INSTALL_DIR>/conf/resin.conf`. Add the following entry for context path `/ERES` `<web-app id="/ERES" document-directory="<ERES_INSTALL_DIR>" />` between `<host-default>` and `</host-default>` tags. The `id` in the entry is the context path of ERES, and `document-directory` is the absolute path of the ERES installation. Please substitute `<ERES_INSTALL_DIR>` with your actual ERES installation directory.

3. In Resin 3.X, start your Resin server by running `<resin-installation-dir>/bin/httpd`. In Resin 4.X, first navigate to the `<RESIN_INSTALL_DIR>` directory and then start the server by running the following command: `java -jar lib/resin.jar start`.
4. Now from your web browser, browse to the ERES Start page to start the server (`http://<server>:<port>/<ERES_CONTEXT>/index.jsp`). For the above configuration, the context path is ERES.
5. From the ERES Start page, login as the administrator and launch the *Admin Console*.
6. From *Admin Console* page, go to the *Setting Info* tab. Change the ERES Servlet Context entry to `/ERES/servlet`. You can also change the webroot to `<RESIN_INSTALL_DIR>/doc`, but it is not necessary.
7. Stop the ERES Server.
8. Stop and re-start Resin if necessary.
9. Go to the ERES Start page and start the ERES server.

### 9.3.2.2. Resin 2.1.12

The following instructions show how to deploy EspressoReport ES under Resin 2.1.12. The instructions assume that Resin has been installed on the system. The location of the Resin installation is referenced as `<RESIN_INSTALL_DIR>`.

For reference, please note that Resin's default port is 8080, the default working directory is `<RESIN_INSTALL_DIR>` and the default Web root is `<RESIN_INSTALL_DIR>/docs`.

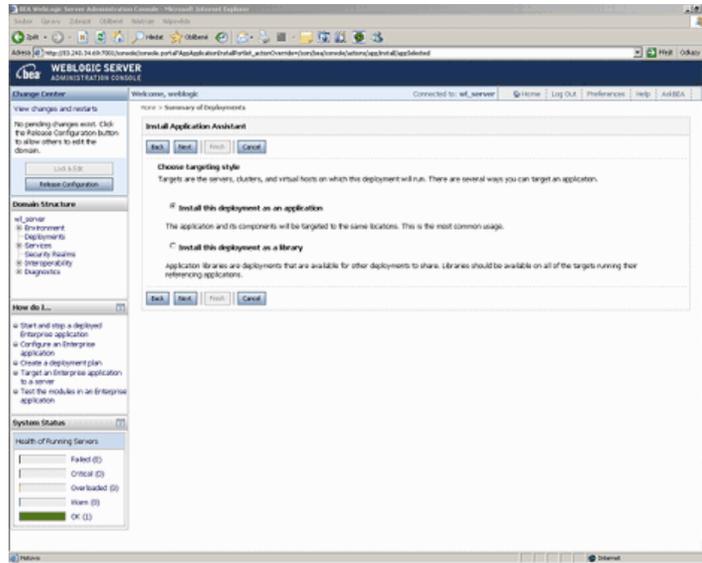
1. You may need to move `tools.jar` file from your Java development kit to the `<RESIN_INSTALL_DIR>/lib` directory (for JSP compilation), or add a library path entry for the file.
2. Modify `<RESIN_INSTALL_DIR>/conf/resin.conf`. Add the following entry for context path `/ERES` `<web-app id="/ERES" app-dir="<RESIN_INSTALL_DIR>"/>` Where `id` is the context path of ERES, and `app-dir` is the absolute path of the ERES installation. Please substitute `<RESIN_INSTALL_DIR>` with your actual ERES installation directory.
3. Start your Resin server by running `<resin-installation-dir>/bin/httpd`.
4. Now from your web browser, browse to the ERES Start page to start the server (`http://<server>:<port>/<ERES_CONTEXT>/index.jsp`). For the above configuration, the context path is ERES.
5. From the ERES Start page, login as administrator and launch the *Admin Console*.
6. From *Admin Console* page, go to *Setting Info* tab and change the ERES Servlet Context entry to `/ERES/servlet`. You can also change the webroot to `<RESIN_INSTALL_DIR>/doc`, but it is not necessary.
7. Stop the ERES Server.
8. Stop and restart Resin if necessary.
9. Go to the ERES Start page and start the ERES server.

## 9.3.3. WebLogic Server

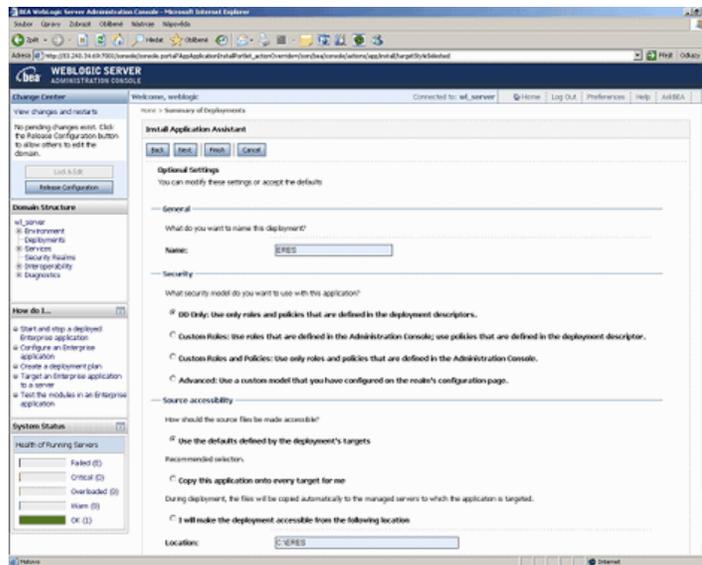
### 9.3.3.1. WebLogic Server 9.2/10.3

The following instructions show how to deploy EspressoReport ES under WebLogic Server 9.2/10.3. The instructions assume that you have WebLogic 9.2/10.3 Server installed on the system and will be deploying ERES under the `wl_server` that comes with WebLogic. The location of the WebLogic installation will be referenced as `<WL_INSTALL_DIR>`.





6. This will show up the following panel that will allow you to specify optional settings of the configuration procedure. From this panel, enter the information for ERES and then press the *Finish* button.

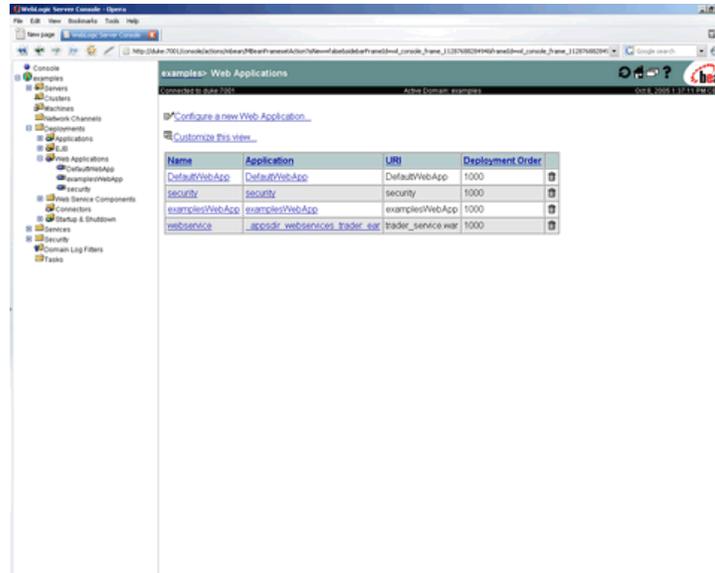


7. If the deployment has been installed and added to the list of pending changes successfully, press the *Activate Changes* button from the Change Center dialog to activate the pending changes.

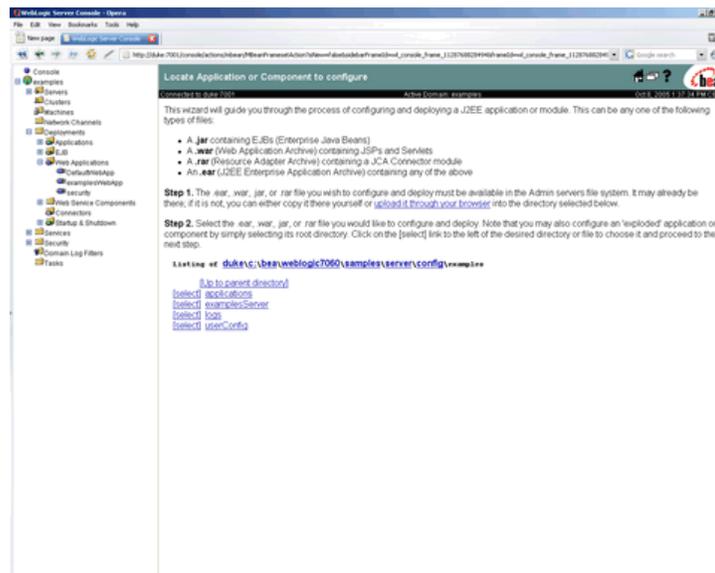
8. Activate the ERES application by checking the checkbox to the left of the ERES link and press the "Start" button.



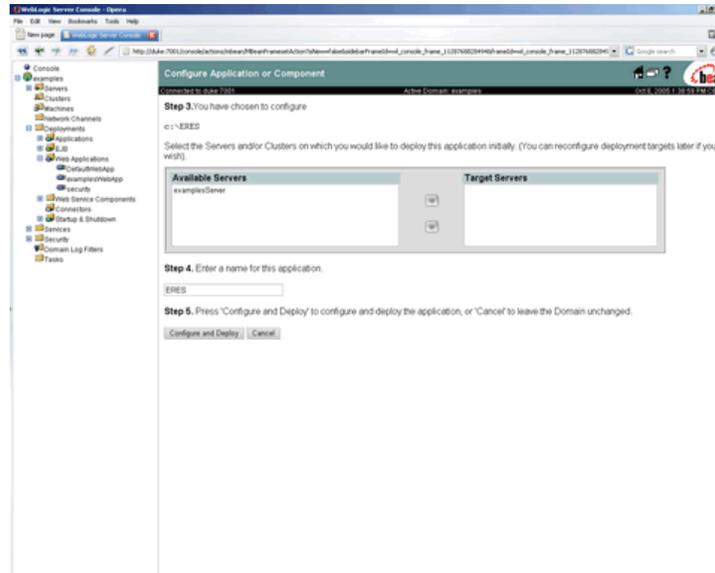
1. Start WebLogic example server by executing `<WL_INSTALL_DIR>/samples/server/config/examples/startExamplesServer.cmd`
2. When Example Server web page is loaded, click the *Administration console* link in the home page.
3. Login to the *administration console* page and click examples → Deployment → Web Applications from the left panel. This will bring up the Web Applications panel on the right.



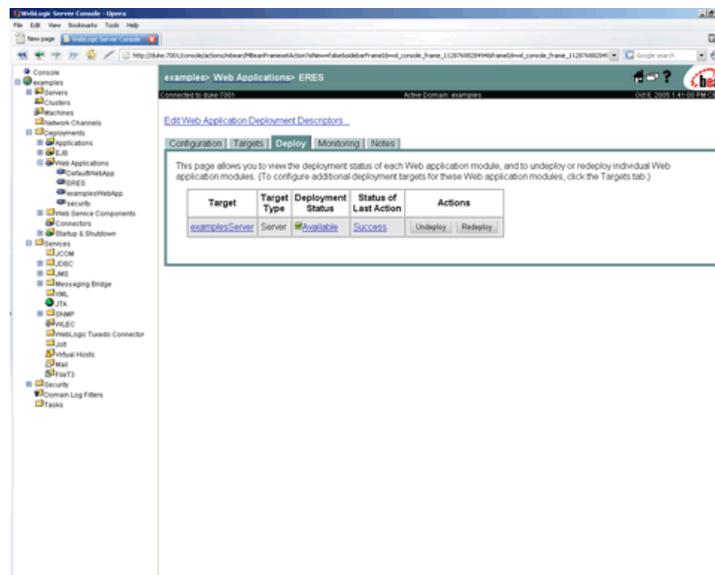
4. Click the *Configure a new web application* link from the above page. This will direct you to install and configure the ERES web application.



5. From the right panel, browse to the expanded application directory of ERES, which in this case will be the installation directory of ERES `<ERES_INSTALL_DIR>`, and then click the *Select* link to the left of the ERES directory. This will bring you to the following panel to continue with Steps 3, 4 and 5 of the configuration procedure. From this panel, enter the information for ERES and choose the targeted server which in this case will be the *exampleServer*.



6. Then press the *Configure and Deploy* button to deploy ERES.



7. If hot deployment is not enabled in your WebLogic application server, stop and restart the exampleServer.

8. Now go to the ERES Start page `http://<server>:<port>/<ERES_CONTEXT>/index.jsp`. From the start page, start the ERES server.

9. From the ERES Start page, login as administrator and launch the *Admin Console*.

10. From Admin Console page, go to *Setting Info* tab and change ERES Servlet Context entry to `/<ERES_CONTEXT>/servlet`. Please substitute `<ERES_CONTEXT>` with your actual context for ERES.

11. Stop and restart the ERES Server.

### 9.3.3.1.3. WebLogic Server 7.0 - Quick Deployment

These instructions will accomplish the same effect - deploying EspressoReport ES under the WebLogic *exampleServer* - as the previous section. This section explains how to deploy ERES without using the WebLogic administration console. Users who are not very familiar with WebLogic should use the instructions in the previous section.

1. Modify the `<WL_INSTALL_DIR>/samples/domains/examples/config.xml` file to add the following entry:

```
<Application Deployed="true" Name="ERES" Path="<PARENT of
ERES_INSTALL_DIR>" TwoPhase="true">

    <WebAppComponent Name="ERES" Targets="exampleServer" URI="ERES"/>

</Application>
```

Path is the parent directory of `<ERES_INSTALL_DIR>`. Application name is the ERES installation directory name. For example, if you install ERES under `C:/ERES`, this should be "ERES". `WebAppComponent Name` is the name of this component. These instructions show how to deploy ERES with `exampleServer`. This is declared by the `Target` attribute in the `config.xml` file. URI is how to access this web component. This is the context path that will be used to access this web application. "ERES" is used as the default context path.

2. Start the WebLogic 7.0 `exampleServer` by executing the following command: `<WL_INSTALL_DIR>/samples/server/config/examples/startExampleServer.cmd`
3. Now go to the ERES Start page `http://<server>:<port>/<ERES_CONTEXT>/index.jsp`. From the start page, start the ERES server.
4. From the ERES Start page, login as administrator and launch the *Admin Console*.
5. From Admin Console page, go to *Setting Info* tab and change the ERES Servlet Context entry to `<ERES_CONTEXT>/servlet`. Please substitute `<ERES_CONTEXT>` with your actual context for ERES.
6. Stop the ERES Server.
7. Stop and restart Weblogic if necessary.
8. Restart the ERES Server.

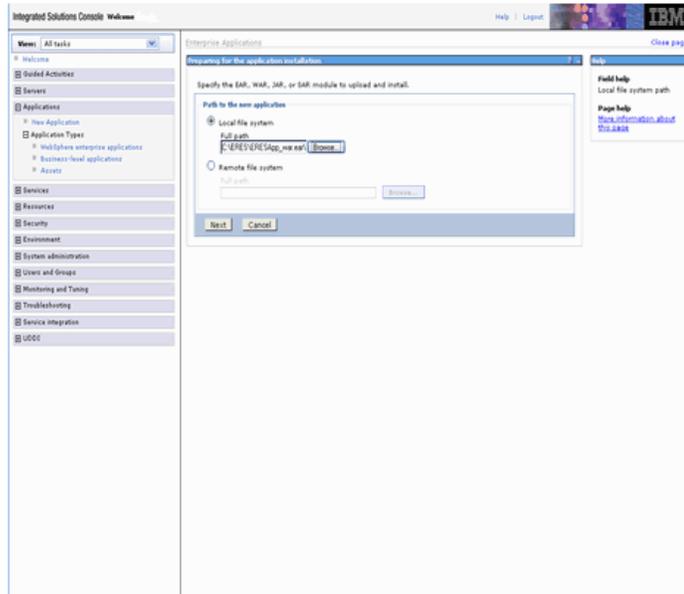
## 9.3.4. WebSphere

### 9.3.4.1. WebSphere 7.1

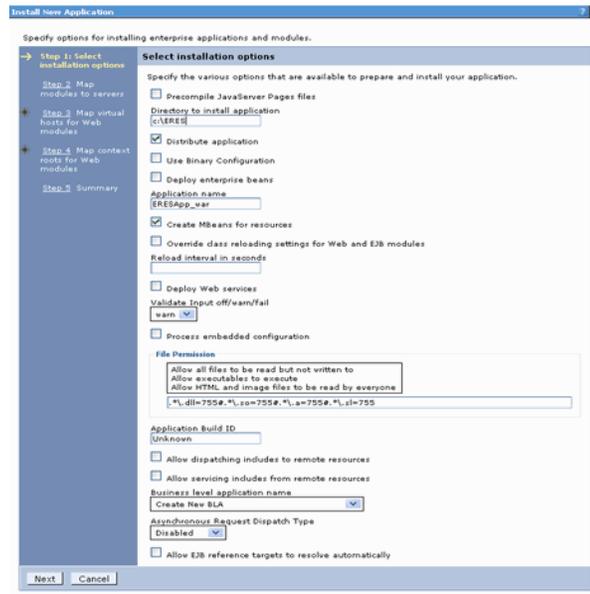
While EspressoReport ES can be deployed in WebSphere 7.1 using several ways, the following instructions show how to do so with minimal changes to WebSphere and ERES. The instructions assume WebSphere has been installed, started, and running on the same machine as ERES.

1. Reinstall ERES under the following directory `ERES/ERESApp_war.ear/ERESApp.war` (for example, `C:\ERES\ERESApp_war.ear\ERESApp.war`), then enter the host name/IP of the WebSphere machine and the port it is running on when prompted in the ERES installer. Leave the default ERES as the servlet context.
2. In order to deploy ERES under WebSphere, you will need to convert the ERES web application into a WAR file, called `ERESApp.war`, using the Java's JAR tool. To do this, make sure that the `java/bin` directory is in your path and navigate to `<ERES_INSTALL_DIR>` in a command window. Then execute the following command:  

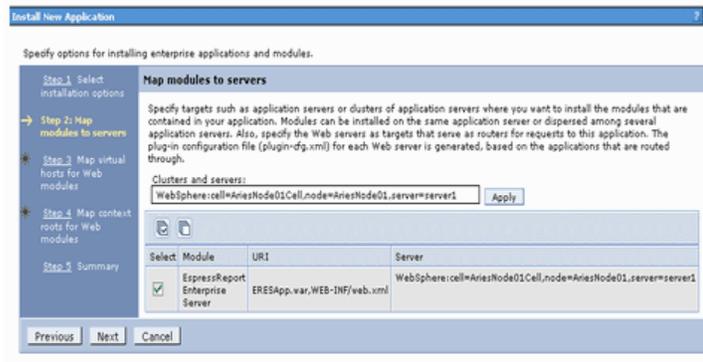
```
jar -cvf ERESApp.war *
```
3. Start the administration console for WebSphere. From the left panel of the administration console, select the *Applications* node and then click the *Install New Applications* link.
4. Select *New Enterprise Application* and then browse to the created war file (which should be `ERES/ERESApp_war.ear/ERESApp.war/ERESApp.war`). After selecting `ERESApp.war`, click on *Next*.



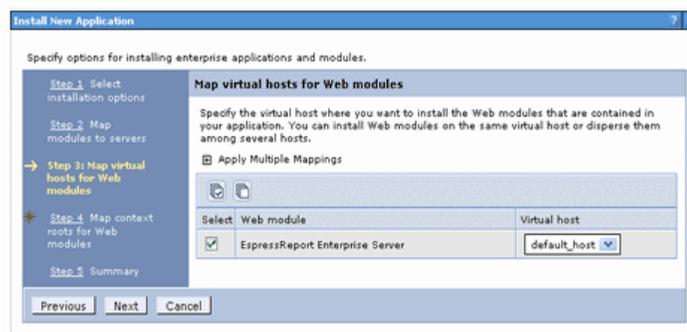
5. When you get asked about how to install the application, select *Fast Path* and click the *Next* button.
6. Under *Directory to install application*, enter the path to /ERES. For example, if you installed ERES under c : \ERES\ERESApp\_war . ear \ERESApp . war, enter c : \ERES. Click *Next*.



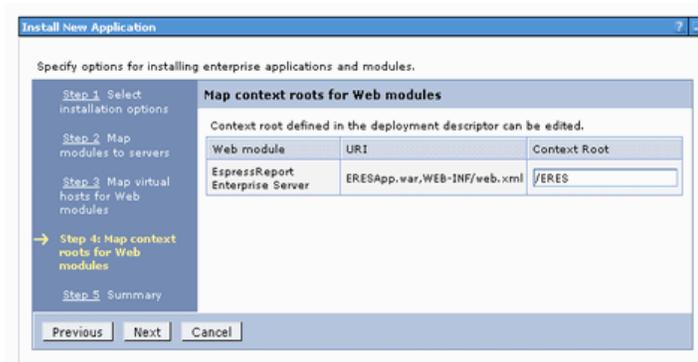
7. Under *Map modules to application servers*, select *Module* and click on *Next*.



8. Under *Map virtual hosts for Web modules*, select *Web module* and click on *Next*.



9. For *Map context root for Web modules*, enter */ERES* under *Context Root* and click on *Next*.



10. A summary of the installation options is then shown. Click on *Finish* and the WebSphere will now deploy the ERES web application.

11. If the deployment was successful, finish it by clicking the *save* link on this page.

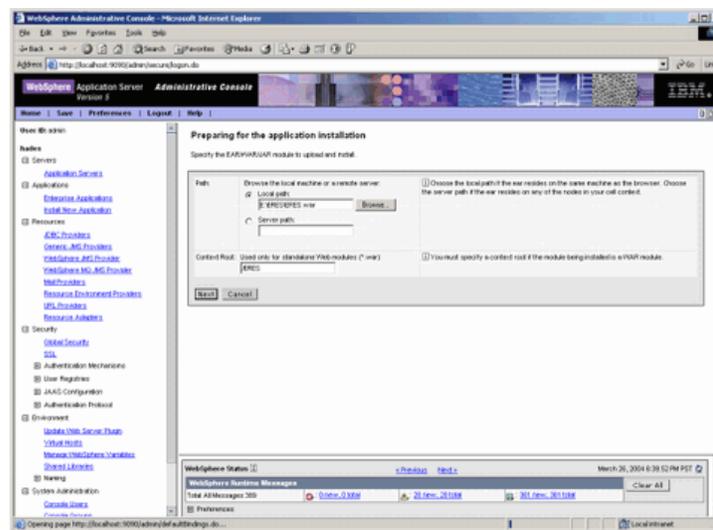
12. Expand *Applications* on the right side and select *WebSphere enterprise applications*. Then select *ERESApp\_war* and click on *Start*.

13. Now go to the ERES Start page `http://<server>:<port>/ERES/index.jsp`. Note that port is 9080 by default. From the start page, start the ERES server.

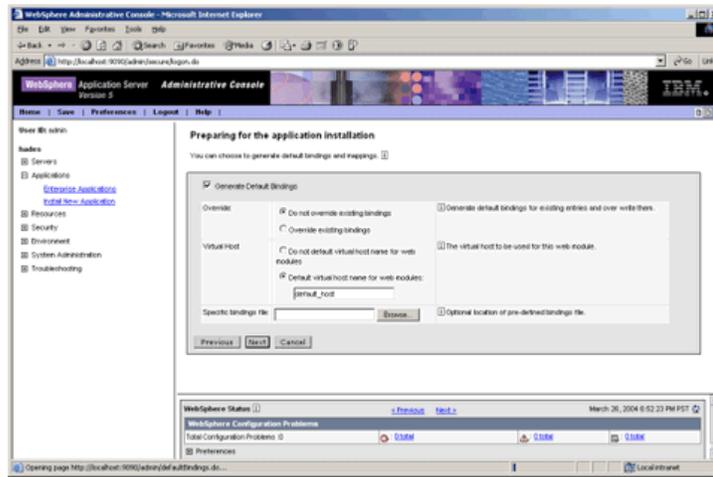
### 9.3.4.2. WebSphere 5.0/6.1

EspressReport ES can be deployed in WebSphere in a number of ways. The following instructions show how to deploy ERES under WebSphere Application Server 5.0 or 6.0/6.1 with minimal configuration changes. The instructions assume that WebSphere Application Server has been installed on the system. The location of the WebSphere installation is referenced as <WS\_INSTALL\_DIR>. WebSphere is also assumed to be started and running.

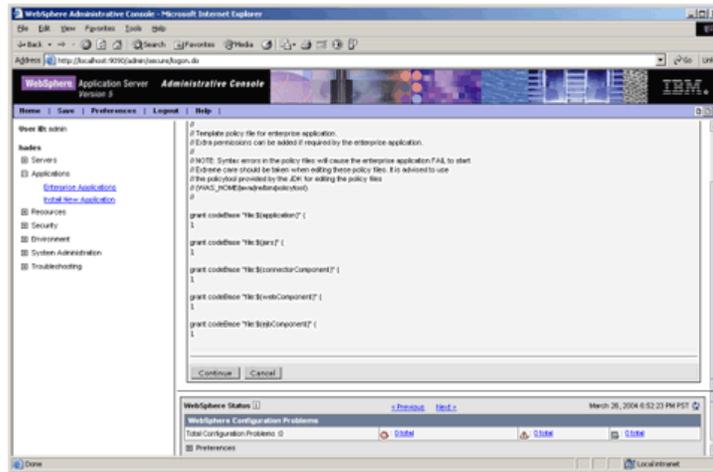
1. Reinstall ERES under the following directory `ERES/ERESApp_war.ear/ERESApp.war` (for example, `c:\ERES\ERESApp_war.ear\ERESApp.war`), then enter the host name/IP of the WebSphere machine and the port it is running on when prompted in the ERES installer. Leave the default ERES as the servlet context.
2. In order to deploy ERES under WebSphere, you will need to convert the ERES web application into a WAR file, called `ERESApp.war`, using the Java's JAR tool. To do this, make sure that the `java/bin` directory is in your path and navigate to <WS\_INSTALL\_DIR> in a command window. Next, execute the following command: `jar -cvf ERESApp.war *`
3. Start the administration console for WebSphere. From the left panel of the administration console, select the *Applications* node and then click the *Install New Applications* link.
4. You will be redirected to the following window for preparing the application installation. From the right panel, provide the war file path (which should be `ERES/ERESApp_war.ear/ERESApp.war/ERESApp.war`), enter **ERES** for the context path and then click the *Next* button. Please note that if you are deploying ERES under WebSphere 6.1, you can skip steps 5 and 6 and continue with step 7.



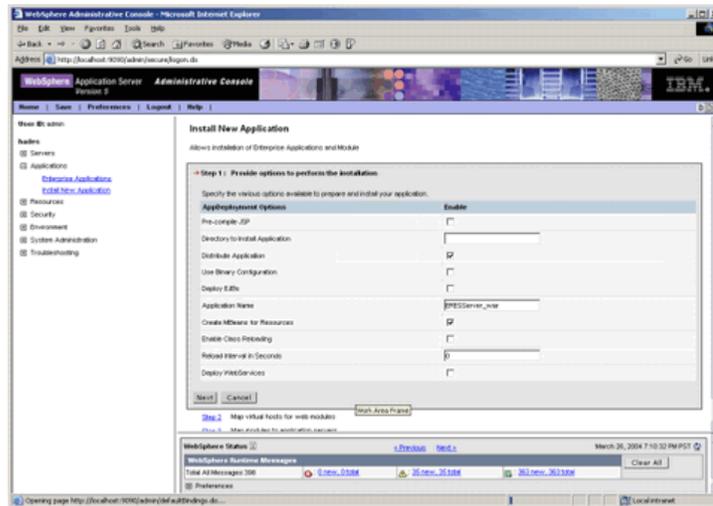
5. In case you do not have any binding specified, select *Generate Default Bindings* in the next page.



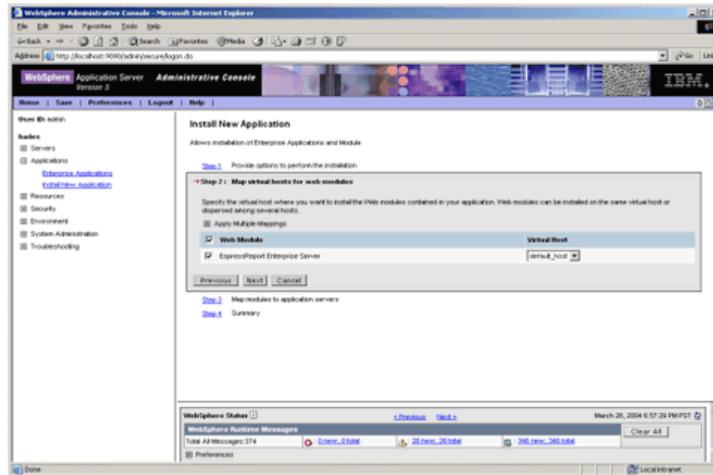
6. Click the *Next* button.



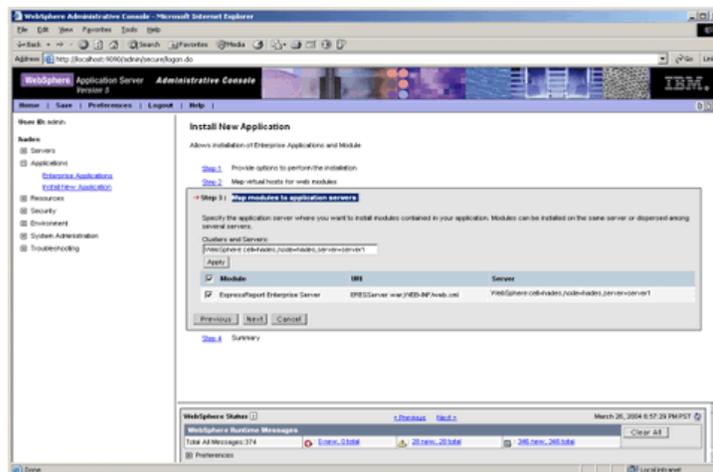
7. Click the *Continue* button. Now you are redirected to the step 1 of the installation: *Provide options to perform the installation*. Under *Directory to install application*, enter the path to /ERES. For example, if you installed ERES under `c:\ERES\ERESApp_war.ear\ERESApp_war`, enter `c:\ERES`.



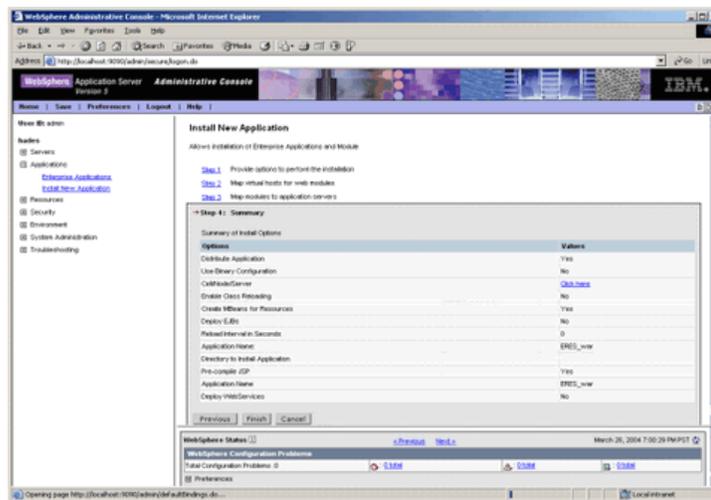
8. Click the *Next* button to go to step 2: *Map virtual hosts for web modules*. Select *Web Modules*.



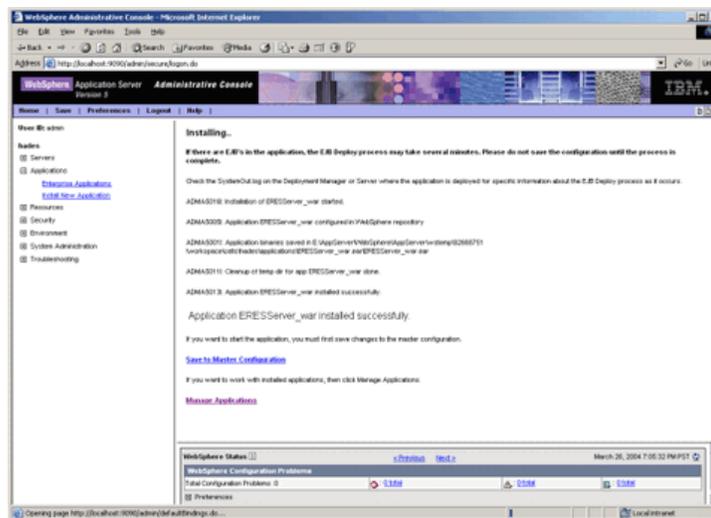
9. Click the *Next* button to go to step 3: *Map modules to application servers*. Select *Module*.



10. Click the *Next* button to go to step 4: *Summary of this web application*. Please make sure that all information is correct. If it's not, go back to the previous pages to edit the information.



11. After verifying all informations, click the *Finish* button. WebSphere will now try to deploy the ERES Web application.



12. If the deployment was successful as shown above, save the changes either by clicking the *save* link in this page, or clicking on the *save* link on top of the administration console. Note that you may have to go under *Applications* in the right frame, expand *Enterprise Applications* and start *ERESApp\_war*.

13. Now go to the ERES Start page `http://<server>:<port>/ERES/index.jsp`. Note that port is 9080 by default. From the start page, start the ERES server.

### 9.3.4.3. WebSphere on z/OS

Deploying EspressoReport ES to WebSphere for z/OS has a few additional steps:

1. Download `ERES.war` and deploy it on WebSphere (see above section for general WebSphere deployment). You will need to use `viacsi` to configure text files that are mentioned in the instructions. Also, since the required jar files are in the `WEB-INF/lib` directory inside the war file, you can skip the steps for adding them as classpath to your servlet container.
2. ERES requires Java graphics support to run. See Section 8.9.4.2 - JVM 1.5+ in Headless Mode for more detail about using graphics on headless platform. In brief, if you are using JDK 1.5 or higher, the solution is to set it

to run in headless mode by specifying the `-Djava.awt.headless=true` flag for the java command that starts your WebSphere. For JDK 1.3 or lower, you can use PJA graphics emulation. This is discussed in the deployment section of the Programming Guide. However, if the PJA causes Java not to find its fonts:

```
-Djava2d.font.usePlatformFont=false -Djava.awt.fonts=TrueTypeFontsPath
```

try using this flag instead:

```
-Djava2d.font.usePlatformFont=true
```

When using `Djava2d.font.usePlatformFont=true`, you will also need to copy `Arial*.ttf` fonts (can be found in Windows platform) to the native `jre/lib/fonts` directory. Also, note that you will not need the flag `-Djava.awt.fonts=TrueTypeFontsPath`.

3. Stop and restart WebSphere if necessary. ERES is now set up and accessible using URL: `http://host:port/ERES/index.jsp`.

### 9.3.5. JBoss 3.2.3 (with Tomcat 5.0) / JBoss 4.0.5/6.0.0

The following instructions show how to deploy EspressoReport ES under JBoss 3.2.3 with Tomcat 5.0 / JBoss 4.0.5 /6.0.0. The instructions assume you have JBoss installed on the system. The location of the JBoss installation is referenced as `<JBoss_INSTALL_DIR>`.

For reference, the default port is 8080 and the default working directory is `<JBoss_INSTALL_DIR>/bin`.

1. You need a `jboss-web.xml` deployment descriptor to deploy ERES. This file should be added to the `<ERES_INSTALL_DIR>/WEB-INF` directory. This is the same location as `web.xml` file. This deployment descriptor is to specify the ERES context. The content should look like this:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE jboss-web PUBLIC "-//JBoss//DTD Web Application 2.3//EN"
"http://www.jboss.org/j2ee/dtds/jboss-web_3_0.dtd">

<jboss-web>

    <context-root>/ERES</context-root>

</jboss-web>
```

2. Rename the ERES installation directory `<ERES_INSTALL_DIR>` to `ERES.war`, then move the entire `ERES.war` directory to `<JBoss_INSTALL_DIR>/server/default/deploy`. This new directory is referenced as `<ERES_DEPLOY_DIR>`. Next time when JBoss starts, server will automatically deploy the ERES web application.
3. Open `<ERES_DEPLOY_DIR>/WEB-INF/classes/QB.properties` file and change the `AutoStartServerInstallDir` entry to point to `<ERES_DEPLOY_DIR>`. Example:

Original entry

```
AutoStartServerInstallDir=<ERES_INSTALL_DIR>
```

New entry

```
AutoStartServerInstallDir=<ERES_DEPLOY_DIR>
```

Now you have to change Database URL. If the `DatabaseUrl` entry already exists in your `QB.properties` file, just change it to point to the `hsqldb` database under `<ERES_DEPLOY_DIR>`. Example:

Original entry

```
DatabaseUrl=jdbc\:hsqldb\:<ERES_INSTALL_DIR>/data/quadbasedb
```

New entry

```
DatabaseUrl=jdbc\:hsqldb\:<ERES_DEPLOY_DIR>/data/quadbasedb
```

If the entry does not exist in your file, save the changes and open <ERES\_DEPLOY\_DIR>/WEB-INF/orguserdb/config.txt file. In the file, modify [database url] entry to point to the hsql database under <ERES\_DEPLOY\_DIR>. Example:

Original entry

```
[database url] jdbc:hsqldb:<ERES_INSTALL_DIR>/data/quadbasedb
```

New entry

```
[database url] jdbc:hsqldb:<ERES_DEPLOY_DIR>/data/quadbasedb
```

4. Execute run.bat / .sh in the <JBOSS\_INSTALL\_DIR>/bin directory to start the JBoss Server. The ERES application will be automatically deployed. The deployment information should be printed out on the JBoss console.
5. Go to the ERES Start page [http://<server>:<port>/<ERES\\_CONTEXT>/index.jsp](http://<server>:<port>/<ERES_CONTEXT>/index.jsp) to start the ERES server. ERES\_CONTEXT is defined in your jboss-web.xml file. In this case, it will be ERES.
6. From the ERES Start page, login as administrator and launch the *Admin Console*.
7. From Admin Console page, go to *Setting Info* tab and change ERES Servlet Context entry to /<ERES\_CONTEXT>/servlet. Please substitute <ERES\_CONTEXT> with your actual context for ERES.
8. Stop and restart the ERES server.

## 9.3.6. Oracle Containers for J2EE (OC4J) 10g (9.0.4.0/10.1.3.5)

The following instructions show how to deploy EspressoReport ES under OC4J 10g (9.0.4.0/10.1.3.5). The instructions assume that you have OC4J installed on the system. The location of the OC4J installation is referenced as <OC4J\_INSTALL\_DIR>

For reference, the default port is 8888 and the default working directory is <OC4J\_INSTALL\_DIR>/j2ee/home.

1. The easiest way to deploy ERES is to put ERES under default-web-app. Copy the entire ERES directory <ERES\_INSTALL\_DIR> to <OC4J\_INSTALL\_DIR>/j2ee/home/default-web-app. This new directory is referenced as <ERES\_DEPLOY\_DIR>.
2. You will need to copy the tools.jar file from your Java development kit to the <OC4J\_INSTALL\_DIR>/j2ee/home/applib directory (for JSP compilation), or add a library path entry for the file. Check your <OC4J\_INSTALL\_DIR>/j2ee/home/config/application.xml file. The following entry should exist in the file: <library path=" ../applib"/>

3. Open `<ERES_DEPLOY_DIR>/WEB-INF/classes/QB.properties` file and change the `AutoStartServerInstallDir` entry to point to `<ERES_DEPLOY_DIR>`. Example:

Original entry

```
AutoStartServerInstallDir=<ERES_INSTALL_DIR>
```

New entry

```
AutoStartServerInstallDir=<ERES_DEPLOY_DIR>
```

Now you have to change Database URL. If the `DatabaseUrl` entry already exists in your `QB.properties` file, just change it to point to the hsql database under `<ERES_DEPLOY_DIR>`. Example:

Original entry

```
DatabaseUrl=jdbc\:hsqldb\:<ERES_INSTALL_DIR>/data/quadbasedb
```

New entry

```
DatabaseUrl=jdbc\:hsqldb\:<ERES_DEPLOY_DIR>/data/quadbasedb
```

If the entry does not exist in your file, save the changes and open `<ERES_DEPLOY_DIR>/WEB-INF/orguserdb/config.txt` file. In the file, modify `[database url]` entry to point to the hsql database under `<ERES_DEPLOY_DIR>`. Example:

Original entry

```
[database url] jdbc:hsqldb:<ERES_INSTALL_DIR>/data/quadbasedb
```

New entry

```
[database url] jdbc:hsqldb:<ERES_DEPLOY_DIR>/data/quadbasedb
```

4. Backup the `web.xml` file under `<OC4J_INSTALL_DIR>/j2ee/home/default-web-app/WEB-INF`, then move all directories and files (except for the `/orguserdb/` directory) under the `<ERES_DEPLOY_DIR>/WEB-INF` to the `<OC4J_INSTALL_DIR>/j2ee/home/default-web-app/WEB-INF` directory. This will make all the class files and servlets accessible from OC4J. If necessary, merge the old `web.xml` file with the `web.xml` file for ERES to create a new `web.xml` file.
5. Start the Oracle server by executing the command `java -jar oc4j.jar` in the `<OC4J_INSTALL_DIR>/j2ee/home` directory.
6. Go to the ERES Start page `http://<server>:<port>/<ERES_CONTEXT>/index.jsp` to start the ERES server. The `ERES_CONTEXT` will be the directory name of `<ERES_DEPLOY_DIR>`.
7. From the ERES Start page, login as administrator and launch the *Admin Console*.
8. From Admin Console page, go to *Setting Info* tab and change the ERES Servlet Context entry to `/servlet/`.
9. Stop and restart the ERES server.

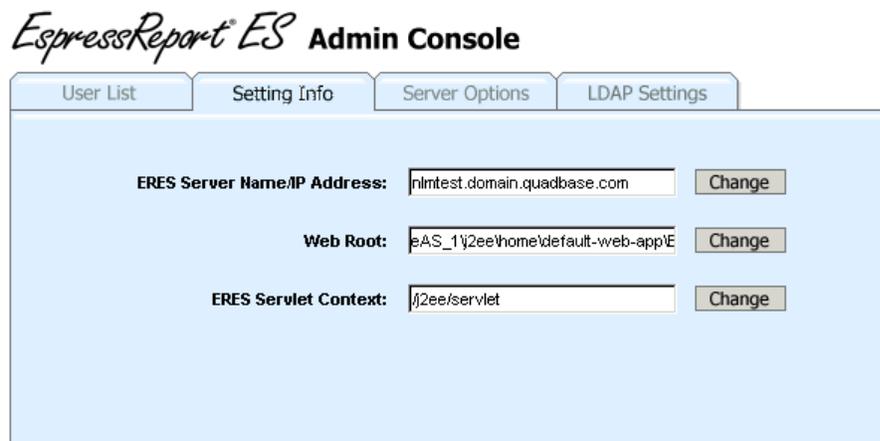
### 9.3.7. Oracle Application Server 10g R3 (10.1.3)

The following instructions show how to deploy EspressoReport ES under Oracle Application Server 10g R3. The instructions assume that you have Oracle AS installed on the system with standard installation. The location of the Oracle AS installation is referenced as <O10g\_INSTALL\_DIR>

1. Install ERES to the <O10g\_INSTALL\_DIR>/j2ee/home/default-web-app/ or
2. Copy the ERES directory to <O10g\_INSTALL\_DIR>/j2ee/home/default-web-app/.
3. The new location for the ERES directory is referenced as <ERES\_HOME>.
4. If the database is set up with direct path, open <ERES\_HOME>/WEB-INF/classes/QB.properties and modify the database info as shown below.

```
DatabaseUrl=jdbc:hsqldb:<ERES_HOME>/data/quadbasedb
```

5. Move all directories and files (except for the /orguserdb/ directory) from <ERES\_HOME>/WEB-INF to <O10g\_INSTALL\_DIR>/j2ee/home/default-web-app/WEB-INF/ directory.
6. Add <ERES\_HOME>/lib/EREServer.jar to the CLASSPATH user variable.
7. Go to <http://<machine>:80/j2ee/ERES> and start ERES server.
8. In the admin console, change the servlet context to /j2ee/servlet.

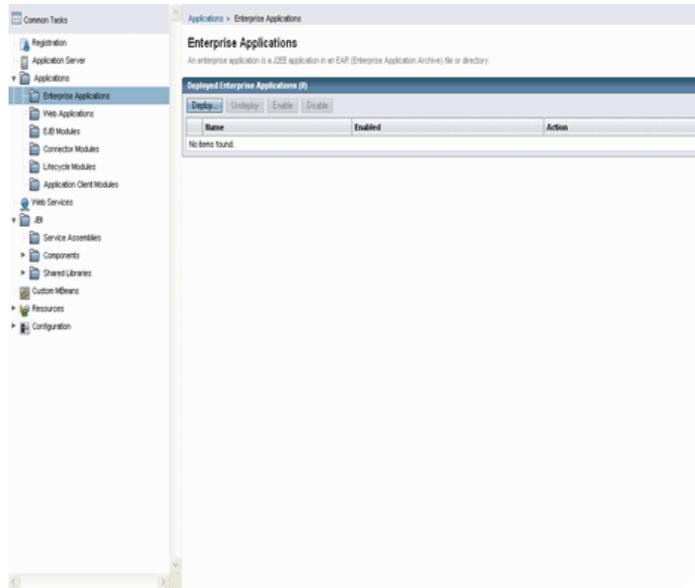


9. Restart ERES Server

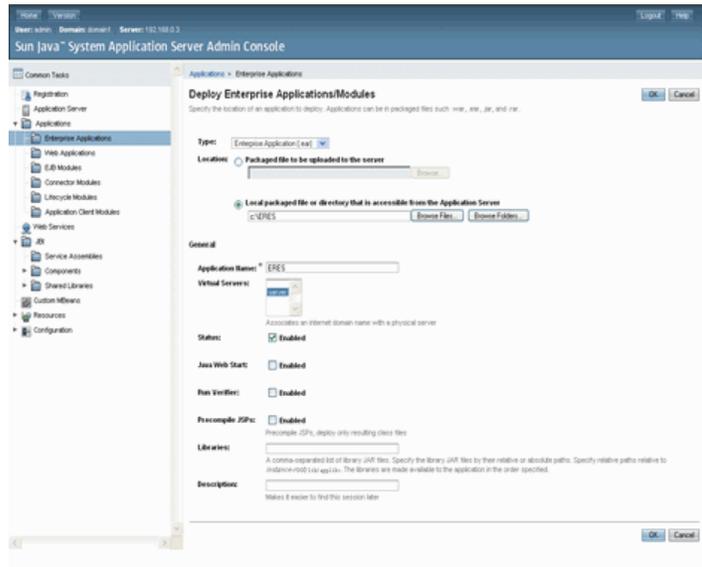
### 9.3.8. GlassFish Server 3.0.1

The following instructions show how to deploy EspressoReport ES under GlassFish Server 3.0.1. The instructions assume that the application server and ERES are installed on the same machine and that GlassFish is up and running.

1. Log in to the GlassFish Admin Console. Click on *Applications* or (in Glasfish v2) expand *Applications* on the left and then click on *Enterprise Applications*.



2. Click on *Deploy* to deploy a new enterprise application.
3. Select *Browse Folders* and select <ERES\_INSTALL\_DIR>.
4. Enter **ERES** under *Application Name* and select the server under *Virtual Server*.

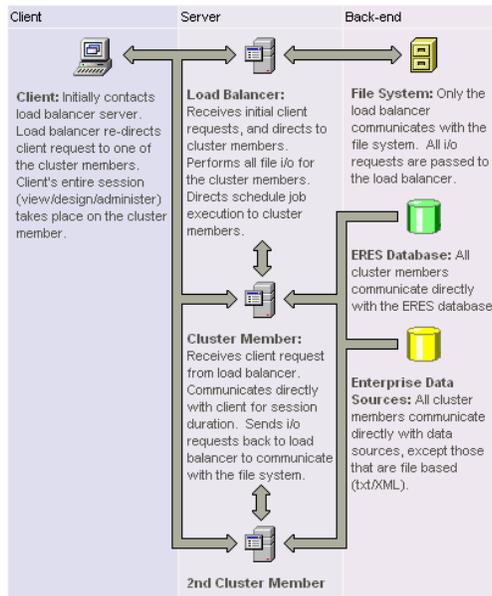


5. Click on the *OK* button to finish deployment.
6. In the Enterprise Application window, select the ERES application you have created and then select *Launch*. This will enable the enterprise application (i.e. start it) and launch it in a separate browser window. You can now start using ERES.

## 9.4. Clustering/Load Balancing

Scalability is an important concern for enterprise reporting. One of the best ways to easily scale an application while eliminating server down-time is to deploy the application in a clustered environment. To provide for maximum scalability, ERES can be deployed in a clustered environment.

When running ERES in a cluster, one server acts as the load balancing machine, receiving incoming requests and routing them to other servers in the cluster where they are processed. This following diagram illustrates ERES running in a clustered environment.



*ERES Clustering Diagram*

All incoming requests are handled by the load balancing server, which will redirect the requests to one of the cluster members. The client's entire session will then take place on the cluster member. Users have an option to either make the load balancer one of the cluster members, or use it solely as the balancer. Each of the cluster members will directly access data sources and read/write information from the ERES Server, but only the load balancer will access the file system. All i/o requests from the cluster members are routed through the load balancer. This ensures that users are working with the same set of deployed reports and charts, regardless of which cluster member they are currently using.

The following chapter details how to deploy ERES cluster in different application server environments.

### 9.4.1. Tomcat 5.0

In this section, you will be modifying the following:

- <TomcatInstallDir>/conf/server.xml
- Admin Console
- <TomatInstallDir>/webapps/balancer/WEB-INF/web.xml
- <TomcatInstallDir>/webapps/balancer/WEB-INF/config/rules.xml
- <ERESInstallDir>/index.html
- <ERESInstallDir>/WEB-INF/web.xml

The first step in setting up ERES cluster is to deploy ERES to all the servers in the cluster. You will need to install ERES on all the machines and deploy it in the Tomcat instance. You can follow the instructions in Section 9.3.1 - Tomcat 4.1/5.x/6.0.x/7.0.x.

Next, modify the server.xml under <TomcatInstallDir>/conf on the load balancing machine. Add the following configuration to this file to enable clustering.

```
<Cluster className="org.apache.catalina.cluster.tcp.SimpleTcpCluster"
```

```
managerClassName="org.apache.catalina.cluster.session.SimpleTcpReplicationManager"
```

```

expireSessionsOnShutdown="false"
useDirtyFlag="true">

<Membership
  className="org.apache.catalina.cluster.mcast.McastService"
  mcastAddr="228.0.0.4"
  mcastPort="45564"
  mcastFrequency="500"
  mcastDropTime="3000"/>

<Receiver
  className="org.apache.catalina.cluster.tcp.ReplicationListener"
  tcpListenAddress="auto"
  tcpListenPort="4000"
  tcpSelectorTimeout="100"
  tcpThreadCount="6"/>

<Sender
  className="org.apache.catalina.cluster.tcp.ReplicationTransmitter"
  replicationMode="pooled"/>

  <Valve className="org.apache.catalina.cluster.tcp.ReplicationValve"
    filter=".*\.gif;.*\.js;.*\.jpg;.*\.htm;.*\\.html;.*\\.txt;"/>

</Cluster>

```

There is a default clustering implementation in the `server.xml` file, but it is commented out. For the load balancer server, you will need to use this slightly different implementation instead.

For all the other servers in the cluster, you can use the default cluster configuration in the `server.xml` file. You just need to uncomment the default configuration. In the default configuration, the default `tcpListenPort` is 4001.

If any of the cluster members have more than one IP address, it may be necessary to add `mcastBindAddress` parameter to the `<Membership>` tag. This argument allows you to specifically set an IP for the cluster service.

If you are using HSQR as the ERES database, you will need to change the configuration of this database on the load balancer machine. HSQR usually runs as application process and cannot be accessed from other machines. In order to make it available to other servers in the cluster, it needs to be configured to run in client-server mode. For setup instructions, please see Section 1.3.2.1.2 - Running HSQR in Client-Server Mode.

Next, you will need to log in as **Admin** and enter the *Admin Console*. Go to Setting Info → Clustering & Load Balancer Settings and add the entries for `Server Host`, `Server Port Number` and `Cluster Member List`.

Server Host: 192.168.0.8

Server Port Number: 8080

Cluster Member List: 192.168.0.8:8080  
 192.168.0.10:8080  
 192.168.0.16:8080

The `Server Host` and `Server Port Number` entries should be the address and port number for the load balancing server. The first entry under the `Cluster Member List` heading should also be the load balancer server. It needs to be listed under this argument even if the load balancer is not acting as a cluster member. The order of other servers should be same in the Admin Console for each ERES instance.

Next, you will need to deploy the balancer web application to the load balancer server. This is a standalone Web application for Tomcat that is included in the ERES installation under `<ERESInstallDir>/Clustering/Tomcat/balancer`. To deploy the Web application, copy the contents of the `/balancer/` directory to the load balancer server and place it under `<TomcatInstallDir>/webapps/`. The `web.xml` file for the balancer application, now under `<TomcatInstallDir>/webapps/balancer/WEB-INF`, maps `LoadBalancer` to a `BalancerFilter` and gives the definition of the `BalancerFilter` with its class file and `rules.xml` file.

```
<web-app>

    <!-- BalancerFilter definition -->

<filter>

    <filter-name>BalancerFilter</filter-name>
    <filter-class>org.apache.webapp.balancer.BalancerFilter</filter-class>
    <init-param>
        <param-name>configUrl</param-name>
        <param-value>/WEB-INF/config/rules.xml</param-value> </init-
param>

</filter>

<!-- BalancerFilter mapping -->

<filter-mapping>

    <filter-name>BalancerFilter</filter-name>
        <url-pattern>/LoadBalancer</url-pattern> </filter-mapping>

</web-app>
```

`Rules.xml` defines how the `BalancerFilter` should redirect requests to server instance. The following is an example of `ruler.xml` file and it is based on `RoundRobinRule`. `RoundRobinRule.class` and several other rule classes are included in the balancer package.

```
<rules>

    <!-- Redirect to server instance based on RoundRobinRule -->
    <rule className="org.apache.webapp.balancer.rules.RoundRobinRule"
        serverInstance="1"
        maxServerInstances="3"
        tcpListenAddress="127.0.0.1"
        tcpListenPort="4000"
        redirectUrl="http://192.168.0.8:8080/ERES/index.jsp" />

    <rule className="org.apache.webapp.balancer.rules.RoundRobinRule"
        serverInstance="2"
        maxServerInstances="3"
        tcpListenAddress="127.0.0.1"
        tcpListenPort="4001"
        redirectUrl="http://192.168.0.10:8080/ERES/index.jsp" />

    <rule className="org.apache.webapp.balancer.rules.RoundRobinRule"
        serverInstance="3"
        maxServerInstances="3"
        tcpListenAddress="127.0.0.1"
```

```

tcpListenPort="4001"
redirectUrl="http://192.168.0.16:8080/ERES/index.jsp" />

</rules>

```

Each defined server instance in the `rules.xml` file should point to one of the ERES cluster members. If you want the load balancer server to also function as a cluster member, include it as a server instance here. You need to make sure that the `tcpListenPort` in the `rules.xml` file matches the port specified in the `server.xml` file for that server. The `RoundRobinRule`, as depicted above, routes each incoming request to the next cluster member in sequence using the default `tcpListenPort`. The balancer application also includes `RandomRedirectRule` option. This will randomly choose a cluster member for each incoming request.

Finally, you need to create a new `index.html` (instead of the current `index.jsp`) page in the `<ERESInstallDir>` directory on the load balancer server. The page can be blank, however, it needs to have the following meta tag at the top of the page:

```

<meta http-equiv="refresh" content="0; URL=http://servername:8080/balancer/
LoadBalancer" />

```

This will redirect all incoming requests to the balancer application, where it will be routed to one of the cluster members following the logic defined in the `rules.xml` file. Replace `servername` with the domain or IP address of your load balancer server in the URL. In order to make the new `index.html` page the default start page for the load balancer machine, you have to edit the `web.xml` file under `<ERESInstallDir>/WEB-INF` and change the following entry:

```

<welcome-file-list>

    <welcome-file>index.jsp</welcome-file>

</welcome-file-list>

to

<welcome-file-list>

    <welcome-file>index.html</welcome-file>

</welcome-file-list>

```

### 9.4.1.1. Running ERES in a Tomcat Cluster

Once you complete the deployment, start Tomcat on your load balancing server and then do the same on all other servers in the cluster. To get to the ERES home page, call the `index.html` file that you created on the load balancer server. You will then be redirected to the Start page on one of the cluster machines. When you start the ERES server on this cluster machine, it will automatically start the ERES server on all other machines in the cluster.

Once you have been redirected to a particular server through the load balancer, your entire session will take place on that machine. However, all file I/O will take place on the load balancing machine. This ensures that all users will interact with the same set of reports and charts which are stored in the file system. All other system information is stored in the ERES database, which can be accessed by any server in the cluster.

### 9.4.1.2. Set up ERES clustering using Apache Web Server and Tomcat

The following instructions will help you setup ERES using Apache Web Server as the load balancer and Tomcat Clusters. In this section, you will be modifying the following files:

- <ApacheInstallDir>/conf/httpd.conf
  - <ApacheInstallDir>/conf/workers.properties
  - <TomcatInstallDir>/conf/server.xml
1. Setup ERES cluster with Tomcat using the instructions in Section 9.4.1 - Tomcat 5.0.
  2. Apache HTTP server:
    - a. Download Apache HTTP server 2.2.4 from The Apache HTTP Server Project (<https://httpd.apache.org/> [ <https://httpd.apache.org/> ])
    - b. Run the installer and select standard installation.



**Note**

Do not install to the default Program Files directory as space in directory name could cause problems later. Install directory name must not have space in it, for example "Apache2.2" is fine.

- c. Open the Apache Server Monitor and start the web server if it's not already running.
  - d. Point your browser to <http://localhost/> to verify that Apache is running on port 80.
  - e. Stop Apache.
3. Apache-Tomcat connector
    - a. Download Mod JK Tomcat connector from <https://tomcat.apache.org/download-connectors.cgi>.



**Note**

You want to download the binary - click on JK 1.2 Binary Releases → win32 → jk-1.2.21 → mod\_jk-apache-2.2.4.so

- b. Copy the mod\_jk-apache-2.2.4.so to the modules directory in your Apache installation.
- c. Rename it to mod\_jk.so
- d. Open httpd.conf in the conf directory of your Apache installation in a text edit and add the following line at the end of the LoadModule statements: LoadModule jk\_module modules/mod\_jk.so
- e. Create a file called workers.properties in the conf directory. Add these lines to it:

```
workers.tomcat_home=<tomcat_install_directory>
workers.java_home=<jdk_install_directory>
worker.list=balancer

worker.worker0.port=8009
worker.worker0.host=192.168.0.8
worker.worker0.type=ajp13
worker.worker0.lbfactor=1

worker.worker1.port=9009
worker.worker1.host=192.168.0.8
worker.worker1.type=ajp13
worker.worker1.lbfactor=1

worker.worker2.port=8009
```

```
worker.worker2.host=192.168.0.10
worker.worker2.type=ajp13
worker.worker2.lbfactor=1

worker.worker3.port=9009
worker.worker3.host=192.168.0.10
worker.worker3.type=ajp13
worker.worker3.lbfactor=1

worker.balancer.type=lb
worker.balancer.balance_workers=worker0,worker1,worker2,worker3
worker.balancer.method=B
worker.balancer.sticky_session=True
```

We have specified the worker list to a single worker called balancer and specified that the worker type of balancer is 'lb' or load balancer. The workers it manages are worker0, worker1, worker2 and worker3. And we set the balance method to 'B' or balance by busy factor. Apache will delegate the next request to the Tomcat instance which is the least busy. Please note that there are several options for the balance method - consult the Apache/Tomcat documentation, which lists the options for workers properties, to help you decide the best method for your type of application.



### Note

Port numbers must match the Connector port (with protocol="AJP/1.3") defined in Tomcat's server.xml. 8009 is the original port and 9009 for the second instance of Tomcat on the same machine.

- f. Edit the server.xml for each instance of Tomcat and add a jvmRoute attribute to the Engine element:

```
<Engine name="Catalina" defaultHost="localhost" jvmRoute="worker0">
```

for the first instance and

```
<Engine name="Catalina" defaultHost="localhost" jvmRoute="worker1">
```

for the second.

```
<Engine name="Catalina" defaultHost="localhost" jvmRoute="worker2">
```

for the third.

```
<Engine name="Catalina" defaultHost="localhost" jvmRoute="worker3">
```

for the last.

- g. Specify the worker properties in httpd.conf:

Add these lines just after the LoadModule definitions:

```
JkLogFile c:/apache2.2/mod_jk.log
JkLogLevel info
JkLogStampFormat "[%a %b %d %H:%M:%S %Y] "
JkOptions +ForwardKeySize +ForwardURICompat -ForwardDirectories
JkRequestLogFormat "%w %V %T"
#use apache balancer
JkMount /ERES balancer
JkMount /ERES/* balancer
```

#### 4. Start up

- a. Startup Tomcat50, Tomcat51, Tomcat52 and Tomcat53
- b. Start Apache Server
- c. Go to <http://192.168.0.8:8080/ERES/index.html> and start the ERES server
- d. Go to <http://192.168.0.8/ERES/index.jsp>



#### Note

The internal Tomcat server (<http://192.168.0.8:8080/ERES>) is for Administrator to start and stop ERES server and should not be available to other users. Other users must go through Apache web server (<http://192.168.0.8/ERES/>) and their requests will be forwarded to Tomcat cluster which is invisible to them. Apache HTTP server will serve loadbalancer.

The above example creates four instances of ERES on two separate machines (192.168.0.8 and 192.168.0.10). To change the number of clusters and machines, modify the worker configuration in step 3.

## 9.4.2. JBoss 3.2.5 (with Tomcat 5.0)

In this section, you will be modifying the following files:

- Admin Console
- `<JBossInstallDir>/server/default/deploy/loadbalancer.sar/META-INF/jboss-service.xml`
- `<ERESInstallDir>/index.html`

The first step in setting up an ERES cluster is to deploy ERES to all servers in the cluster. You will need to install ERES on all machines and deploy it in the JBoss instance. You can follow the instructions in Section 9.3.7 - Oracle Application Server 10g R3 (10.1.3).

If you are using HSQL as the ERES database, you will need to change the configuration of this database on the load balancer machine. HSQL usually runs as an application process and cannot be accessed from other machines. In order to make it available to other servers in the cluster, it needs to be configured to run in client-server mode. For setup instructions, please see Section 1.3.2.1.2 - Running HSQL in Client-Server Mode.

Next, you will need to log in as **Admin** and enter the *Admin Console*. Go to Setting Info → Clustering & Load Balancer Settings and add the entries for Server Host, Server Port Number and Cluster Member List.

Server Host: 192.168.0.8

Server Port Number: 8080

Cluster Member List: 192.168.0.8:8080  
192.168.0.10:8080

The `Server Host` and `Server Port Number` entries should be the address and port number for the load balancing server. The first entry under the `Cluster Member List` heading should also be the load balancer server. It needs to be listed under this argument even if the load balancer is not acting as a cluster member. The order of other servers should be same in the *Admin Console* for each ERES instance.

Next, you will need to deploy the balancer web application to the load balancer server. This is a standalone web application for JBoss that is included in the JBoss installation under `<JBOSS_INSTALL_DIR>\docs\examples\varia\loadbalancer`. To deploy the web application, copy the contents of `/loadbalancer.sar/` directory to the load balancer server and place it under JBoss deploy directory (`<JBOSS_INSTALL_DIR>\server\default\deploy` if you follow the instructions from this manual).

Edit `loadbalancer.sar/META-INF/jboss-service.xml` to match your configuration. Modify `<host-url>` and add new `<host>` so the configuration file contains all cluster members. The example below shows the configuration for two clusters. Value of `<lb-factor>` isn't important because it is not used in this JBoss version.

```
<hosts>

  <host>
    <host-url>http://192.168.0.8:8080/ERES</host-url>
    <lb-factor>1</lb-factor> </host>
  <host>
    <host-url>http://192.168.0.10:8080/ERES</host-url>
    <lb-factor>2</lb-factor> </host>

</hosts>
```

Uncomment the first Monitor service in the `jboss-service.xml`.

```
<!-- Monitor Services -->
<!-- A monitor that only checks that a given path is reachable -->
<mbean code="org.jboss.web.loadbalancer.monitor.SimpleMonitorService"
  name="jboss.web.loadbalancer:service=Monitor">

  <depends optional-attribute-
name="Scheduler">jboss.web.loadbalancer:service=Scheduler</depends>
  <attribute name="Interval">15000</attribute>
  <attribute name="Timeout">20000</attribute>
  <attribute name="Path">/</attribute>

</mbean>
```

You can find a sample `jboss-service.xml` in `<ERES_DEPLOY_DIR>/Clustering/JBoss`. Or you can take a look at `<JBOSS_INSTALL_DIR>\docs\examples\varia\loadbalancer\loadbalancer.-doc` for more detailed description of the load balancer application and its configuration files.

If you use Tomcat 5.0 plugin, you must remove the `<JBOSS_DEPLOY_DIR>/jbossweb-tomcat50.sar/ROOT.war` directory because the load balancer also wants to be root.

Finally, you need to modify `index.html` page in `<ERES_DEPLOY_DIR>` directory on all cluster members. The page can be blank, but it needs to have the following meta tag at the top:

```
<meta http-equiv="Refresh" content="0; url=http://machine:port/ERES/
index.jsp">
```

Replace machine/port with IP address/port of the machine the cluster member is running on. ERES is ERES context on the cluster member. Make the `index.html` page the default start page for all cluster members. You can do this by editing the `web.xml` file under `<ERES_DEPLOY_DIR>/WEB-INF` on all cluster members. Change the following entry:

```
<welcome-file-list>
    <welcome-file>index.jsp</welcome-file>
</welcome-file-list>
```

to

```
<welcome-file-list>
    <welcome-file>index.html</welcome-file>
</welcome-file-list>
```

### 9.4.2.1. Running ERES in JBoss Cluster

Once you complete the deployment, start JBoss on all servers in the cluster and then start JBoss on your load balancing server. Do not forget to start the HSQL database server first (if you are using it). To get to the ERES home page, open `http://machine:port`, where machine is your load balancing server and port is the port JBoss is running on (8080 by default). You will then be redirected to the Start page on one of the cluster machines. When you start the ERES server on this cluster machine, it will automatically start the ERES server on all other machines in the cluster.

Once you have been redirected to a particular server through the load balancer, your entire session will take place on that machine. However, all file I/O will take place on the load balancing machine. This ensures that all users will interact with the same set of reports and charts which are stored in the file system. All other system information is stored in the ERES database, which can be accessed by any server in the cluster.

### 9.4.3. JRun 4 (with Apache Web server)

In this section, you will be modifying the following files:

- Admin Console
- `<JRUNInstallDir>\lib\wsconfig\1\jrunserver.store`
- `<ERESInstallDir>/index.html`
- `<ERESInstallDir>/WEB-INF/web.xml`

The first step in setting up an ERES cluster is to deploy ERES to all servers in the cluster. You will need to install ERES on all machines and deploy it in the JRun instance. You can follow the instructions in Section 9.3.4 - WebSphere. ERES needs to be installed in the same context on all cluster members.

If you're using HSQL as the ERES database, you will need to change the configuration of this database on the load balancing machine. HSQL usually runs as application process and cannot be accessed from other machines. In order to make it available to the other servers in the cluster, it needs to be configured to run in client-server mode. For setup instructions, please see Section 1.3.2.1.2 - Running HSQL in Client-Server Mode.

Next, you will need to log in as **Admin** and enter the *Admin Console*. Then go to Setting Info → Clustering & Load Balancer Settings and add the entries for Server Host, Server Port Number, and Cluster Member List.

Server Host: 192.168.0.8

Server Port Number: 8080

Cluster Member List: 192.168.0.8:8080  
 192.168.0.10:8080  
 192.168.0.16:8080

The `Server Host` and `Server Port Number` entries should be the address and port number for the load balancing server. The first entry under the `Cluster Member List` heading should also be the load balancing server. It needs to be listed under this argument even if the load balancer is not acting as a cluster member. The order of other servers should be same in the Admin Console for each ERES instance.

Use the following steps to complete the JRun deployment:

1. This configuration uses Apache web server as a load balancing server. You will need to install it on the load balancing machine.
2. Next, go to the JRun admin console on the load balancing server and log in. The default port of admin console is 8000.
3. Click *Register Remote Server* at the top of the page. Enter the IP address, server, and JNDI port of the cluster member which is hosted on a machine other than the load balancer. Repeat this step for all cluster members which are hosted on other machines.
4. Then click *Create New Cluster* at the top of the page. Enter the name of the cluster and click *next*. Select all cluster members on the next page. Then click *next* and *done*.
5. Now run all cluster members and run the JRun Web Server Configuration Tool. Here click *Add*. In the following dialog, select the previously created cluster in the *JRun server* field. Select Apache web server and insert Apache configuration directory (`<APACHE_INSTALL_DIR>\conf`). Then click *OK* and close the Web Server Configuration.
6. Open the `<JRUN_INSTALL_DIR>\lib\wsconfig\1\jrunserver.store` file. There will be a similar entry to the one below:

```
proxyservers=192.168.0.8:51000
```

Add all cluster members to this file (separate them with semicolon “;”). The port is the cluster member proxy port (51000 by default). You can find this port in the admin console.

```
proxyservers=192.168.0.8:51000;192.168.0.10:51000
```

Finally, you need to modify `index.html` page in `<ERES_INSTALL_DIR>` directory on all cluster members. The page can be blank, but it needs to have the following meta tag at the top of the page:

```
<meta http-equiv="Refresh" content="0; url=http://machine:port/ERES/index.jsp">
```

Replace `machine/port` with `IP address/port` of the machine the cluster member is running on (port is 8100 by default). ERES is the ERES context on the cluster member. Make the `index.html` page the default start page for all cluster members. You can do this by editing the `web.xml` file under `<ERES_INSTALL_DIR>/WEB-INF` on all cluster members. Change the following entry:

```
<welcome-file-list>

    <welcome-file>index.jsp</welcome-file>

</welcome-file-list>

to

<welcome-file-list>

    <welcome-file>index.html</welcome-file>

</welcome-file-list>
```

### 9.4.3.1. Running ERES in a JRun Cluster

Once you complete the deployment, start JRun on all servers in the cluster and then start Apache on your load balancing server. Do not forget to start the HSQL database server first (if you are using it). To get to the ERES home page, open `http://machine:port`, where `machine` is your load balancing server and `port` is the port Apache is running on (80 by default). You will then be redirected to the Start page on one of the cluster machines. When you start the ERES server on this cluster machine, it will automatically start the ERES server on all other machines in the cluster.

Once you have been redirected to a particular server through the load balancer, your entire session will take place on that machine. However, all file I/O will take place on the load balancing machine. This ensures that all users will interact with the same set of reports and charts which are stored in the file system. All other system information is stored in the ERES database, which can be accessed by any server in the cluster.

### 9.4.4. WebSphere 6.0

In this section, you will be modifying the following files:

- Admin Console

To deploy ERES in a WebSphere cluster, you will first need to have a WebSphere cluster setup (using HTTP Server). We will call your deployed WebSphere cluster `<Cluster_Name>`

Next, you will need to deploy ERES to each of the WebSphere cluster members. To do this, you will need to WAR the `/ERES/` directory and deploy it using the WebSphere administration console. or more information about deploying ERES in WebSphere, see Section 9.3.6 - Oracle Containers for J2EE (OC4J) 10g (9.0.4.0/10.1.3.5). Follow the steps in this section until you get to *Map modules to servers* step. In this step, map the ERES Web application to the `<Cluster_Name>` and your HTTP server.

Make sure that you deploy ERES on all cluster members using the same context.

Next, you will need to log in as **Admin** and enter the *Admin Console*. Then go to Setting Info → Clustering & Load Balancer Settings and add the entries for Server Host, Server Port Number and Cluster Member List.

```
Server Host: 192.168.0.8
```

```
Server Port Number: 8080
```

```
Cluster Member List: 192.168.0.8:8080
                    192.168.0.10:8080
                    192.168.0.16:8080
```

The `Server Host` and `Server Port Number` entries should be the address and port number for the load balancing server. The first entry under the `Cluster Member List` heading should also be the load balancing server. It needs to be listed under this argument even if the load balancer isn't acting as a cluster member. The order of other servers should be same in the Admin Console for each ERES instance.

Next, you will need to log in as **Admin** and enter the *Admin Console*. Go to *Setting Info* → *Clustering & Load Balancer Settings* and add the entries for `Server Host`, `Server Port Number`, `Cluster Member List`, `Member Host` and `Member Port Number`.

`Server Host`: 172.26.35.12

`Server Port Number`: 9080

`Cluster Member List`: 172.26.35.12:9080  
                          172.26.35.13:9080  
                          172.26.35.14:9080  
                          172.26.35.15:9080

`Member Host`: 172.26.35.15

`Member Port Number`: 9080

The `Server Host` and `Server Port Number` entries should be the address and port number for the load balancing server. The first entry under the `Cluster Member List` should also be the load balancer server. It needs to be listed under this argument even if the load balancer is not acting as a cluster member. The order of the other servers should be the same in the Admin Console for each ERES instance. The `Member Host` entry should be the IP address of the current machine, and the `Member Port Number` entry should be the server port number for the current machine. These entries will change for each cluster member.

In the WebSphere administration console, click on *Servers* → *WebServers* → *Generate Plug-in* and *Propagate Plug-in*. The re-start the HTTP Server.

Next, start the WebSphere cluster. Navigate to the `index.jsp` page on the load balancer machine, and start the ERES Server. Note that because the IBM HTTP server does the balancing, you do not need to change the start page for ERES. Users can simply start their session by pointing to the `index.jsp` page on the load balancer machine. The HTTP server on that machine will re-direct their requests to the cluster members.

---

# Chapter 10. Internationalization

## 10.1. Internationalizing ERES

ERES provides many different features that allows you to generate reports for just about any locale and language. Different internationalization features can require different system or setting configurations depending on your specific requirements.

### 10.1.1. Specifying Locales

ERES allows different time zones and locales reports and charts. They are not limited to the locale on the machine where they are created. The locale can be set through API right before the report or chart is run.

#### 10.1.1.1. Locale-Specific Formatting

For data formatting (date and numeric), ERES allows you to set locale-specific formatting for report elements. Locale-specific formatting allows data elements to be displayed in the correct format for the particular locale that is being used for the report. Locale-specific options are available in the data formatting dialog for Report Designer (see Section 4.1.3.7.3 - Data Formatting for Formulas and Column Fields).

The locale for report and charts, as well as the time zone, can be set at run-time through the API. Note that this only effects the date, the time, and the data formatting.

#### 10.1.1.2. Setting Time Zones and Locales Using the API

To set the time zone or locale, you can use the methods in QbReport/QbChart (applies to the entire report/chart) or use `LocaleDateTimeFormat` and `LocaleNumericFormat` objects (applies to a specific object in the report/chart). The following code fragments shows how to do this with two different approaches.

```
report.setLocale(Locale.UK);
report.setTime zone(time zone.getTime zone("GMT"));
```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/LocaleReport.zip> ]

Exported Results [ <http://data.quadbase.com/Docs70/help/manual/code/export/LocaleReport.pdf> ]

Here, the formats are applied to the entire report. Alternatively, you can apply the format to specific data columns as follows:

```
LocaleNumericFormat currencyFormat =
    LocaleNumericFormat.getCurrencyInstance();
currencyFormat.setLocale(Locale.UK);
int columnIndex = 2; //column 2 is the "Unit Price" column
report.getTable().getColumn(columnIndex).setDataFormat(currencyFormat);

LocaleDateTimeFormat dateFormat = LocaleDateTimeFormat.getDateInstance();
dateFormat.setTime zone(time zone.getTime zone("GMT"));
report.getPageFooter().getData(1).setDataFormat(dateFormat);
```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/LocaleReportObject.zip> ]

Exported Results [ <http://data.quadbase.com/Docs70/help/manual/code/export/LocaleObject.pdf> ]

### 10.1.2. Language and Encoding

ERES includes a set of features that removes limitations caused by language differences. By utilizing a simple to use interface, it is possible to translate all text, buttons, and menus within ERES to a different language. Through

some basic configuration, you will be able to import, view, and type characters in a foreign language, as well as save and export reports in that language.

### 10.1.2.1. ERES Language Translation

Internationalization is supported through the use of an `.xml` file, `ERES_Language.xml` (located in the `<ERES installation directory>`). A GUI is provided to work with the `ERES_Language.xml` file and replace the lines of English with those of another language.

The `ERES_Language.xml` file has the following structure:

```
<Product name="REPORTDESIGNER" dir="quadbase/reportdesigner/designer">

    <File name="ReportMenubar.java">
        <CODE>File</CODE>
        <TEXT>File</TEXT>
        <CODE>New</CODE>
        <TEXT>New</TEXT>
        ... </File>

</Product>
```

The file has the following tree structure: Product (Directory) → File → text. You can easily search the product, file, or text you want to translate and simply replace the Text between `<Text>` and `</Text>` with the translation. ERES will replace `<Code>` with `<Text>` in the Java Swing UI and/or in the ERES web application. The translation GUI will list all products and each product will list codes that need translation. A same code (for example, `File`) may be listed under several products to maintain the completeness of each product. However, the translation for a code is duplicated across all products. The same code cannot have two different translations for different products.

The `ERES_Language.xml` file is located in the install directory. It is recommended that you make a copy (of `ERES_Language.xml`) and then use the GUI provided (on the copy) to put in the translation without touching the xml file directly. To utilize the user interface, navigate to the ERES root directory and enter the following command:

```
java -classpath "./lib/ERESOrganizer.jar;."
quadbase.internationalization.TranslateWizard -file:<fileName> -
enc:<encoding>
```

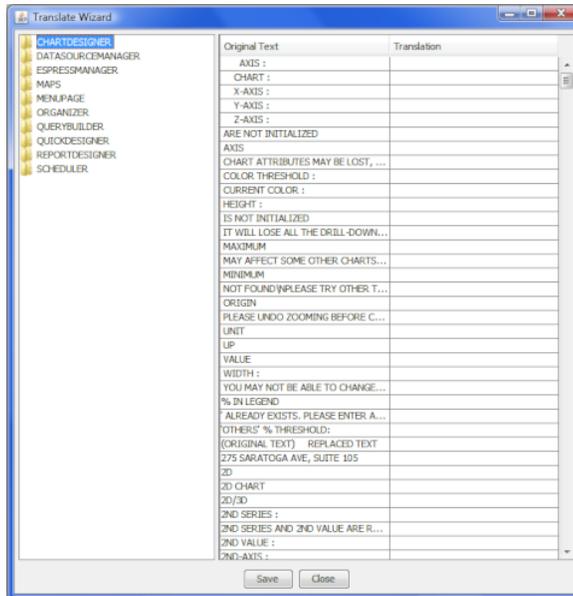
where `<fileName>` represents the xml file that will store the translation and `<encoding>` represents the encoding for the translation. For correct results, the proper encoding must be specified. If the file name and encoding are not provided, the default file name (`ERES_Language.xml`) and default encoding (`Cp1252`) will be used.

An example of the above command is given below:

```
java -classpath "./lib/ERESOrganizer.jar;."
quadbase.internationalization.TranslateWizard -file:Chinese.xml -enc:gbk
```

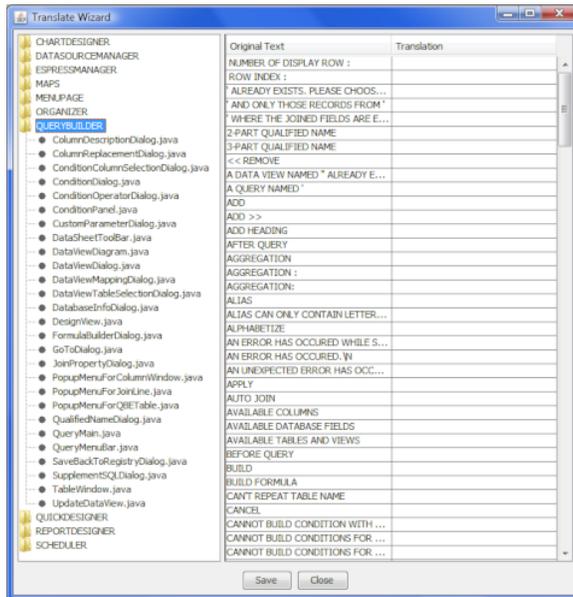
Where `gbk` is the encoding used to save `Chinese.xml` file.

When the Translate wizard is started, it appears as below:



*Translate Wizard*

You can see the various codes (available for translation) by selecting appropriate Product. Products are shown in a tree on the left that contains all the files available for translation. For example, on the image below you can see expanded *QUERYBUILDER* product. You can either translate entire product by selecting the product node or just select a file that you want to translate.



*Insert Translation*

The English texts are listed on the left and you can enter the translation on the right. The translation can be entered by clicking on the cell and typing in the text. The translation can be saved (to the file specified by the command that started the wizard) by clicking on the *SAVE* button. You can navigate to the previous screen by clicking on the *PREVIOUS* button. Note that if a translation for a code has already been set, it will appear for every instance of the code.

You can use *ERES\_Language.xml* (or the copy you modified) in *ERES Organizer* by adding the *-file* and *-enc* arguments (e.g. *-file:ERES\_Language.xml* and *-enc:UTF-8*) to the java web start (jnlp) file that starts *Organizer* (in *ERESOrganizer.jnlp*).

For the web application, edit *index.jsp* page as follows:

- Add an import for the `LanguageEncoder` class:

```
<%@page
  language="java" import="quadbase.common.util.internal.LanguageEncoder"
  errorPage = "MenuError.jsp" contentType="text/html; charset=UTF-8" %>
<jsp:useBean id="index" scope="session" class="quadbase.auth.bean.Index" /
>
<jsp:setProperty name="index" property="*" />
```

- Add a few lines near the beginning of the jsp to load the language file.

```
<%
  quadbase.common.util.internal.LanguageEncoder.load("<language
  file>", "<encoding>");
  index.setRequest(request, application);
%>
```

where `<language file>` is the language file (for example, `Chinese.xml`) and `<encoding>` is the encoding (for example, `gbk`).

#### 10.1.2.1.1. Upgrading Language File

When the user upgrades to a newer version, the `ERES_Language.xml` file needs to be upgraded as well. The language upgrade program will copy over the translations from the previously customized `ERES_Language.xml` file and append the additional entries in the new version. To use the language upgrade program, navigate to the `ERES` directory from a console window and use the following command:

```
java -classpath ".;\WEB-INF\lib\EREServer.jar"
  quadbase.internationalization.UpgradeLanguageXMLFile -from:oldfile -
  to:newfile -enc:encoding
```

It may be necessary to replace the semicolon with colon and backslash with slash on non-Windows environments.

`Oldfile` refers to the customized `ERES_Language.xml` from a previous version of `ERES`, `newfile` refers to the `ERES_Language.xml` included in the upgraded `ERES` installation, and the `enc` is the language encoding used in the translation. After running the program, the resulting file will be named `ERES_Language.xml` and you can open it with `Translate Wizard` to further translate any new entries.

#### 10.1.2.2. Displaying Foreign Characters

Foreign characters can be easily displayed in `Designer` and various `Viewers`. To display foreign characters in a report/chart, you will need to have fonts for that language installed in your system. Then, in the report/chart, you can set the font for the object that contains foreign characters to the appropriate system font.

Another option is to modify the `font.properties` file in the JVM so that foreign characters are supported in the default JVM fonts. For Sun JVMs, the `font.properties` file is located under the `jre/lib` directory. The different language files have names like `font.properties.ru` for Russian, `font.properties.zh` for Chinese, etc. To change the language settings for the JVM, rename the current `font.properties` file to back it up and change the name of the desired language file to `font.properties`. With the language settings changed in the JVM, the default fonts in `ERES` (`Dialog`, `Serif`, `Monospaced`, etc.) will display with foreign characters.

#### 10.1.2.3. Entering Foreign Characters

In order to enter foreign characters into any `ERES` interface (for example, `Organizer` and `Designer`), the following changes to the system settings are required:

- The *default locale* of your system must be set to the region for the language you want to use.

- The *input locale* for the system must also be set to the region for the language you want to use

Note that for Windows the settings can be accessed through *Regional Options* in the *Control Panel*.

In addition, the font settings in the JVM must be adjusted to the desired language following the instructions discussed in Section 10.1.2.2 - Displaying Foreign Characters.

Once these settings have been applied, foreign characters can be entered in labels, queries, formulas, and parameters in reports/charts.

#### 10.1.2.4. XML Encoding

By default, ERES use UTF-8 character set for encoding when writing to XML. This includes data registry files, XML report templates, XML exports, and XML representations of global format information and font mapping. Please note that for most of users it is not necessary to change character set encoding, because UTF-8 fully supports all languages.

This encoding can be changed for other languages by adding a run-time parameter to Organizer and ERES Server. For Organizer, you can do this by modifying the `ERESOrganizer.bat/sh` file, or modifying the JSP source of the applet page used to launch the Organizer.

To change the XML encoding, add the following argument to the `<ERESInstallDir>\ERESOrganizer.bat` or `.sh` file: `-xmlEncoding:Encoding`. For example `-xmlEncoding:ISO-2022-JP` would set the encoding for the Japanese character set.

If you are running Organizer through an applet, you will need to add the following parameter to the applet JSP page: `<PARAM NAME="xml_encoding" VALUE="ISO-2022-JP">`.

This parameter also needs to be set for the server. For more information about server settings, please see Section 1.4.1.3.1 - Secured Parameter.

#### 10.1.2.5. Exporting With Foreign Characters

Most of the ERES export formats will be automatically generated with the UTF-8 character encoding. In order to view the exported reports/charts, the client will need to have the appropriate system fonts installed. The only exception to this is the PDF export. PDF format does not depend on client fonts for viewing. Therefore, foreign language fonts will need to be embedded into the PDF document in order for characters to display properly. To do this, you will need to use the PDF Font Mapping feature. The Font Mapping can be set in Designer and in API.

If you wish to export the report/chart to a locale different from the one that is set on the machine, you can specify a different encoding before doing the export. This is done using the following method in `QbReport`:

```
public void setExportEncoding(String encoding);
```

where `encoding` is a character encoding supported by the Java 2 platform. For example, the character encoding for English (Windows Latin-1) is `Cp1252`. Please note that you have to use the canonical name used by the `java.io` and `java.lang` APIs.

You can also specify the character set used in the DHTML exports using the following method in `QbReport`:

```
public void setHTMLCharSet(String charset);
```

where `charset` is a valid character set. For example, a character set for English used in DHTML documents is `UTF-8`.

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/ExportForeign.zip> ]

Exported Results [ <http://data.quadbase.com/Docs70/help/manual/code/export/ExportForeign.html> ]

#### 10.1.2.6. Specifying Encoding for Text Data Files

By default, ERES use system encoding for any text data files. The encoding for input text data files can be set via API using the following block of code:

```
report.getInputData().setDataFile("sample.dat", false, "ASCII");
```

Full Source Code [ <http://data.quadbase.com/Docs70/help/manual/code/src/DataFileEncoding.zip> ]

Exported Results [ <http://data.quadbase.com/Docs70/help/manual/code/export/DataFileEncoding.pdf> ]

The above block use the following method found in `quadbase.reportdesigner.util.IInputData`:

```
public void setDataFile(String fileName, boolean isDataSorted, String  
encoding);
```

# Chapter 11. Alerts

## 11.1. What is an Alert

An alert is basically an exception handling mechanism that notifies user when certain data is out of a pre-defined range. Typically, the user is looking at some KPIs (key performance indicators) that are important to his/her organization. Some example KPIs on which a user may want to set alerts are inventory level, profit margin, sales growth rate, percentage of dropped calls in a mobile network, number of intrusion attempts on a company network, response time on customer support/service center etc..

ERES supports two types of alerts - dashboard alerts and monitoring alerts. Dashboard alerts are evaluated only on currently opened dashboards. They are not triggered when the dashboard is not open. Monitoring alerts are alerts in scheduled objects (reports, charts, maps, dashboards). This is similar to scheduling (described in Section 2.2 - Scheduling & Archiving), except that at the time when alert monitoring task is run, alert conditions are evaluated and the monitoring task actions (i.e. Send email, upload to FTP) are run only if an alert was triggered. If no alerts in the alert monitoring task are triggered, no action is performed.

## 11.2. Specifying alerts

Alerts are specified in a displayable object, namely maps, charts and reports.

### 11.2.1. Online Maps

There is currently no support for alerts in Online Maps.

### 11.2.2. SVG Maps

For SVG Maps, alerts can be specified on all thresholds in the map. You can select ranges that are critical for you. If any of the areas on the map would be in that range, alert will trigger.

If the map doesn't have any thresholds, alerts will not be available for the map.

**Example:**



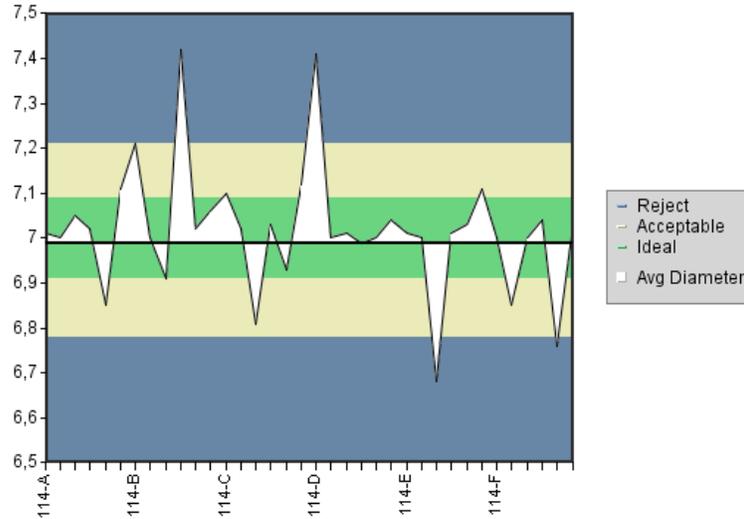
*SVG map with thresholds*

You can for example say that ranges <= \$90000 and \$90000 - \$330000 are critical and you will get alert if there are any yellow or light green areas in the map.

## 11.2.3. Charts

For Charts, you can specify alerts on control areas. Similarly to SVG maps, you can choose the ranges that are critical for you. If there are no control ranges, alerts will not be available for the chart.

**Example:**



*Chart with control areas*

You can say that the Reject control area is critical and you will get alert whenever there is at least one data point in this (blue) control area.

Control area ranges are evaluated as left-closed, right-open intervals [start, end). In other words, the alert will trigger if the following condition is true:  $\text{start} \leq \text{ActualValue} < \text{end}$ .

## 11.2.4. Reports

For reports, you can specify alerts in scripts. There is a special Formatting action called ALERT. You can assign any two strings to this action. First string will indicate name of the alert that should be triggered. Second string represents alert details. Alert details can be any sentence explaining the meaning of the alert or a convenient function result (for example: `getAllRowData()` - explained in Section 4.1.6.2.8.2 - String Functions). Several scripts can trigger single alert, or you can specify several different alerts in one report. For more information about scripting visit Section 4.1.7.1 - Creating a Script.

In the alert UI, you will get names of all possible alerts that can be triggered in the report and you can choose which ones you want to watch.

**Example:**

There are these two scripts in a report:

```
if ({Profit} < 0){
  FONTCOLOR = [255, 0, 0];
  ALERT = ["Negative Profit",getAllRowData()];
} else {
  FONTCOLOR = [0, 255, 0];
}
```

and

```

if ({AccountBalance} < 0){
  FONTCOLOR = [255, 0, 0];
  ALERT = ["Negative Account Balance", "You're in debt!"];
}

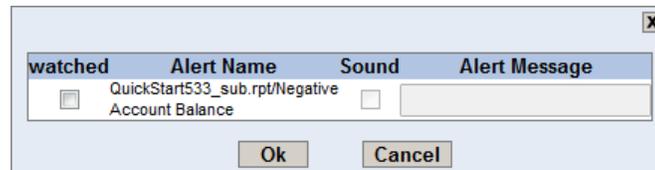
```

If there is any row where `AccountBalance` is negative, the alert called `Negative Account Balance` will trigger. If there is any row where `Profit` is negative, the alert called `Negative Profit` will trigger. If both conditions are satisfied at once, both alerts will trigger.

The named alerts are analogical to ranges in maps and control areas in chart. They also have a name and you can select which ones you want to watch (you do not have to watch all of them). How to watch an alert will be described later.

### 11.2.4.1. Subreports

Subreports support alerts as well. The alert names are propagated to the main report. The alert names are prepended with subreport name to keep them unique in the main report (main report can have alerts with the same names and we don't want to mix them).



*Alert in subreport*

### 11.2.4.2. Charts in reports

For charts in reports, names of the control ranges are propagated to the report. They will also be prepended with the chart name to avoid naming conflicts.

## 11.3. Dashboard alerts

Dashboard alerts are set up for individual templates (reports, charts, maps) in the dashboard using the user interface in the dashboard builder. Alert (exception) conditions are evaluated when the templates are exported, i.e. during initial loading, auto refresh, manual refresh of the dashboard, changing parameters and resizing the template. Auto refresh is enabled every time you preview or view a dashboard. Default auto refresh interval is 60 seconds.

### 11.3.1. Visual alert display

If an alert was triggered, color of template border will change and the border will start blinking. For templates in folders, tab header color will change and blink.

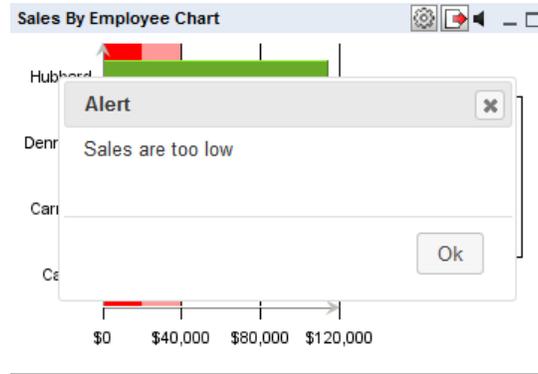
In addition, the template will have its own “native” highlighting of the values:

- coloring of areas in the SVG maps
- control ranges in charts
- for reports, visual changes that can be done by scripting to change text color, background color, border color, change image, change value, etc. (complete list of all formatting options can be found in Section 4.1.7.2.1 - Formatting Actions).

Please note that alert highlighting, blinking and alert messages are visible only in *Preview* of the Dashboard Builder.

### 11.3.2. Alert messages

If you also set up an alert message, a message with your text will pop-up.



*Triggered alert message*

### 11.3.3. Sound alert

In some cases, sound alerts may be preferred over the visual alerts. When an alert is triggered, the template will continuously make beeping sound. If the alert is muted and you change a parameter in the template and the watched alert is triggered again, the alert sound will be active again. However, if the same watched alert is triggered after auto refresh, the alert sound will not reset, but if another watched alert is triggered, the alert sound will be active again. If the alert is disabled during auto refresh, it will reset back to active state.

### 11.3.4. User interface

Alerts dialog can be accessed using this button  in the header of every template in the dashboard. This button is only visible for templates that can have some alerts. It will not be displayed for templates that cannot have alerts (i.e. all Google maps, SVG maps without thresholds, charts without control ranges and reports without scripts with ALERT variable).

After clicking the button, you will get a dialog with all possible alerts that can be watched (i.e. names of all control ranges for charts, all ranges for SVG maps or all possible values of the ALERT variable for reports).

watched	Alert Name	Sound	Alert Message
<input type="checkbox"/>	<1000	<input type="checkbox"/>	
<input type="checkbox"/>	1000-10000	<input type="checkbox"/>	
<input type="checkbox"/>	10000-20000	<input type="checkbox"/>	
<input type="checkbox"/>	20000-30000	<input type="checkbox"/>	
<input type="checkbox"/>	30000-40000	<input type="checkbox"/>	
<input type="checkbox"/>	>40000	<input type="checkbox"/>	

*Watch alerts dialog*

All the *watched* checkboxes will be unchecked by default, which means that no alert is watched. After checking a *watched* checkbox, you enable *Sound* checkbox and *Alert message* textbox, which will allow you to use sound alerts and alert messages. The text you fill in alert message textbox will then display as a pop-up message when the watched alert is triggered. If you leave it empty, no message will appear. You can then check the checkboxes and start watching the alerts. The alert name cannot be changed. It will always be read from the template.

All alerts will be joined using logical OR. If any of these alerts trigger, the dashboard alert will trigger. The AND operation only makes sense for reports and this kind of conditions can always be set directly in the script in the report.

---

## 11.4. Monitoring

Monitoring is supported for dashboards and also for individual reports, charts and maps (let's call them **monitored objects**). Monitoring is similar to scheduling. The monitored objects are checked in given time intervals and notifications are sent if alert occurs in some of the monitored objects. This is done in background, so the notifications are sent even if the monitored object is not viewed.

The user interface to set up monitoring is similar to that of scheduling (scheduling described in Section 2.2 - Scheduling & Archiving). If the monitored object has parameters, user can set up several parameter sets (just like for scheduling). Monitoring has the same options for testing intervals as scheduling (i.e. one time, time interval, fixed days, etc.). Monitoring is not supported for drilldowns.

Administrators and designers are the only users who can set up monitoring (only for objects they have read access to). They can also select recipients of notification emails. Viewers are not allowed to set up monitoring.

### 11.4.1. Alert notifications

Alert notifications are sent via emails. Exported templates can also be uploaded to FTP. Alert notification can be sent every time an alert occurs or it can be send only when an alert starts (i.e. if alert wasn't triggered during last check, but it is triggered during current test).

You can set email subject and body. They have to be the same for the entire monitoring (for all templates, alerts and parameter sets). Body of the notification email can contain some variables that will be substituted during runtime. List of these variables will be detailed later.

You can set different recipients for different templates and different parameter sets (just like in scheduling). It is not possible to set different recipients for different alerts in one template. If you want that, you have to create another alert monitoring task (you can add as many tasks as you want).

### 11.4.2. Failed emails

If some of the monitored objects are not available, or their data sources are not available, or if there are any other problems with exporting the monitored objects or alert evaluation (e.g. watched control area was removed), the user can be notified with failed emails.

The UI is exactly the same as the Failed emails dialog in scheduling (described in Section 2.2.1.2.1 - Email Delivery Options). You can set failed email subject and body. Recipients are the same as for the notifications. Error log (with stack traces) can be sent to a selected user (typically admin).

### 11.4.3. User interface

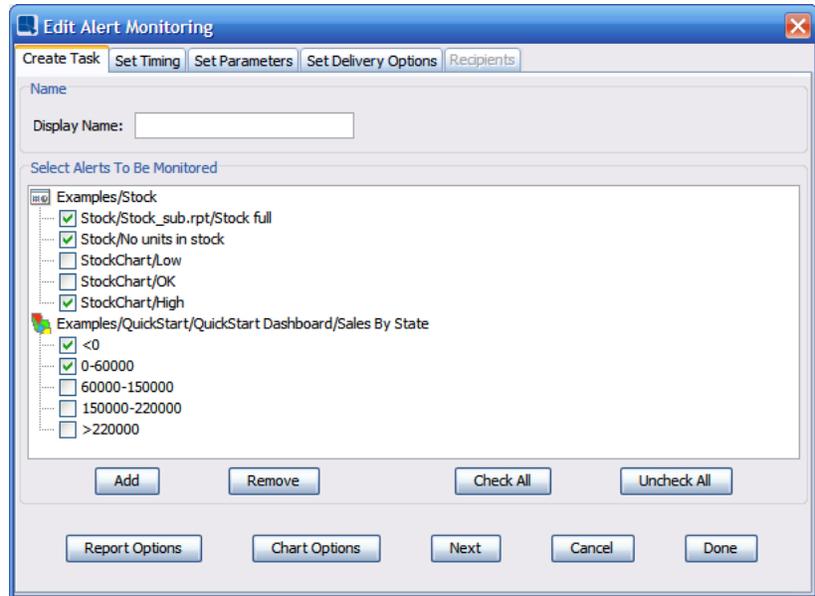
#### 11.4.3.1. Create/Edit monitoring dialog

Monitoring is set up in the Organizer in a dialog similar to the Schedule dialog (see Section 2.2.1 - Setting a Schedule for more details).

To set up, please use the following steps.

**Select alerts -**

This step is similar to the first step of the Schedule Package wizard. Select the templates that you want to monitor in this monitoring task. You can select any dashboard, stand-alone report, chart or map.



*Create monitoring task dialog*

Unlike the list in schedule package, the monitored template list is a tree.

The first level of the tree is the selected templates (before the dialog was opened) or templates added using the *Add* button. The leaf nodes are available alerts (for their parent node). You can check alerts that should be monitored. You have to select at least one alert for each template. If you don't want to monitor any alert of a monitored object, remove the object by selecting it and clicking the *Remove* button.

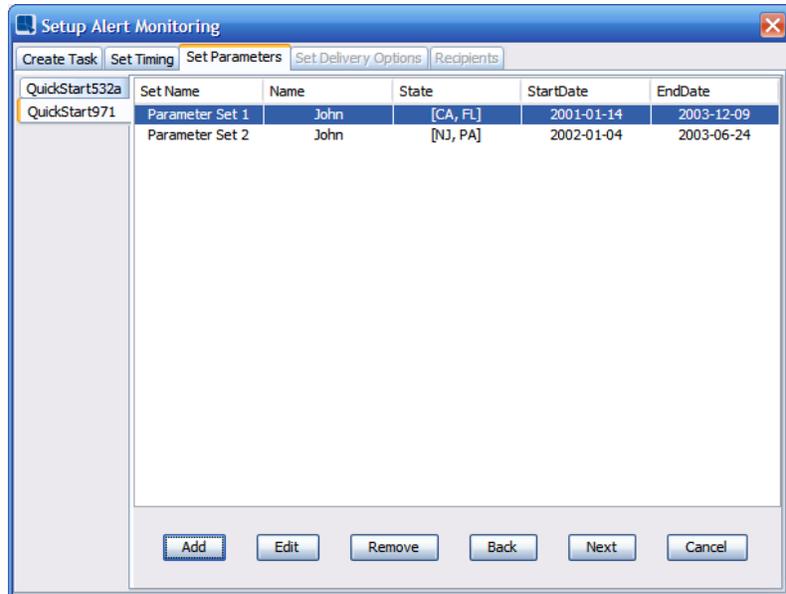
If the selected template is dashboard that already have some dashboard alerts defined, the checkboxes are initialized according to the dashboard alerts (the alerts that are watched in the dashboard are checked). If the selected template is not dashboard, all the checkboxes are unchecked by default.

**Set timing –**

this step is exactly the same as the Set timing in Scheduling (see Section 2.2.1 - Setting a Schedule for more details).

**Set parameters –**

this step is skipped if there are no parameters in the selected templates. If you have selected to monitor a report, chart, map or dashboard that contains parameters, the next tab will appear allowing you to set the parameter values you want to use.



*Set parameters dialog*

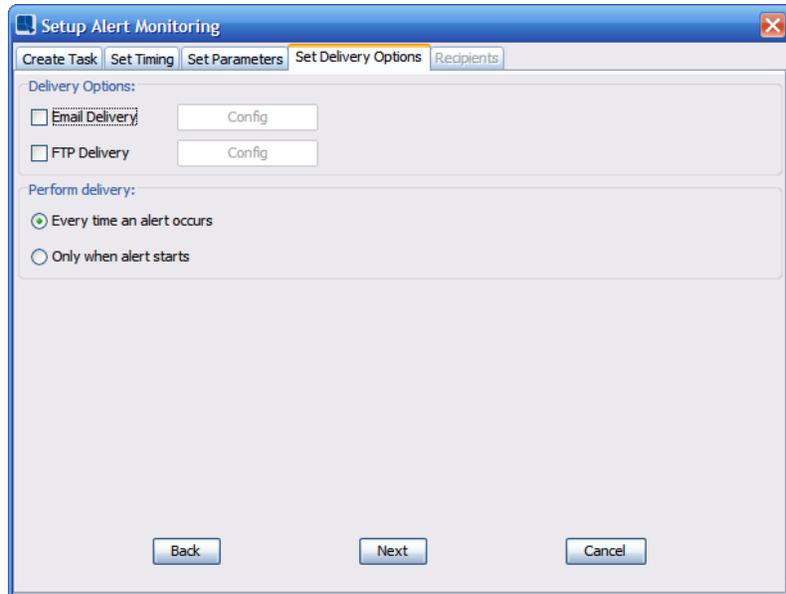
To add a set of parameters, select the *Add* button. This will bring up the parameter prompt dialog allowing you to select the set of parameter values you want to use. Once you have selected a parameter set, your choices will appear in the dialog. You can add as many different combinations of parameter sets as you like. A separate file will be generated for each set of parameters you specify. Alerts will be evaluated separately for each parameter sets. Only the templates that triggered some alerts will be sent in the notification email or uploaded to FTP.

You can also specify a name for each parameter set by double clicking on the first column. The name specified here will be used in later dialogs to help you organize your recipients.

Once you specify all parameters, click the *Next* button to continue.

#### **Set delivery options -**

This dialog allows you to configure the delivery options for the alert notifications. It is almost the same as Set Delivery Option tab in the Scheduling dialog (described in Section 2.2.1.2 - Additional Delivery Options).



*Delivery options dialog*

Printer delivery is not available for alert monitoring.

There is an extra option that is not available for scheduling. You can select if you want to perform the delivery (send emails or upload files to FTP) every time an alert occurs or if you only want to perform it when alert starts (i.e. if alert wasn't triggered during last check, but it is triggered during current test).

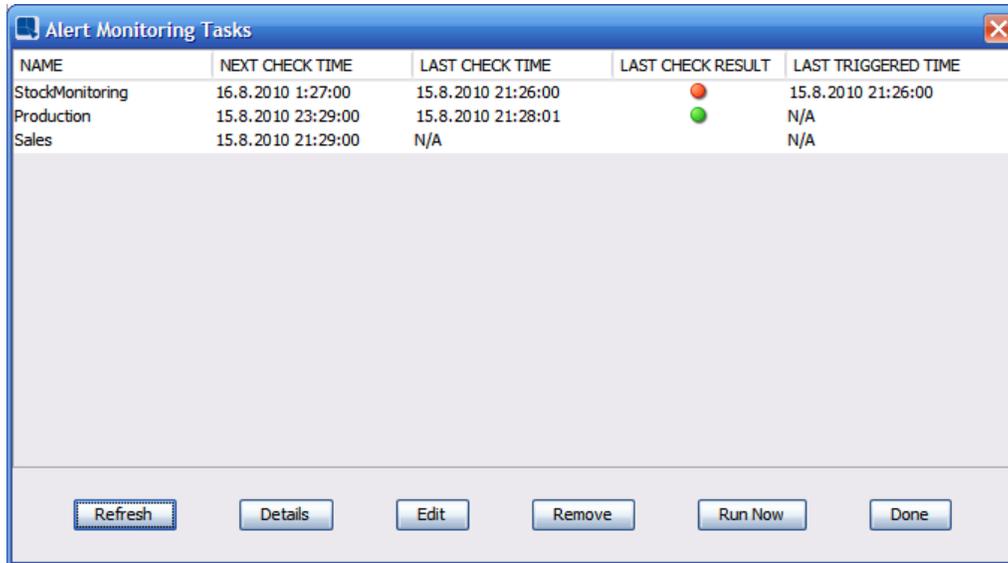
There are also two extra runtime variables that can be used in email body. It is `<ALERTS>` and it is replaced by names of all alerts that have been triggered during the export and `<ALERT_DETAILS>`. For reports, alert details are replaced by alert details specified in the `alert details` parameter (see Section 11.2.4 - Reports to learn more). For charts, it shows values of the tooltips that would be displayed on the data points that triggered alerts. For maps, it shows all values from the data source rows associated to the areas that triggered alert.

#### **Email Recipients -**

This dialog allows you to set the recipients of the emails. It is almost the same as the Recipients dialog for scheduling (described in Section 2.2.1.3 - Specifying Email Recipients. The only difference is that alert monitoring doesn't support report bursting. Specifying Email Recipients). You can assign different recipients for individual templates and parameter sets.

### **11.4.3.2. Monitoring list**

You can view monitoring list from pull-down menu Schedule/Archive → View Alert Monitoring Tasks. The dialog allows users to view, edit and remove current monitoring tasks.



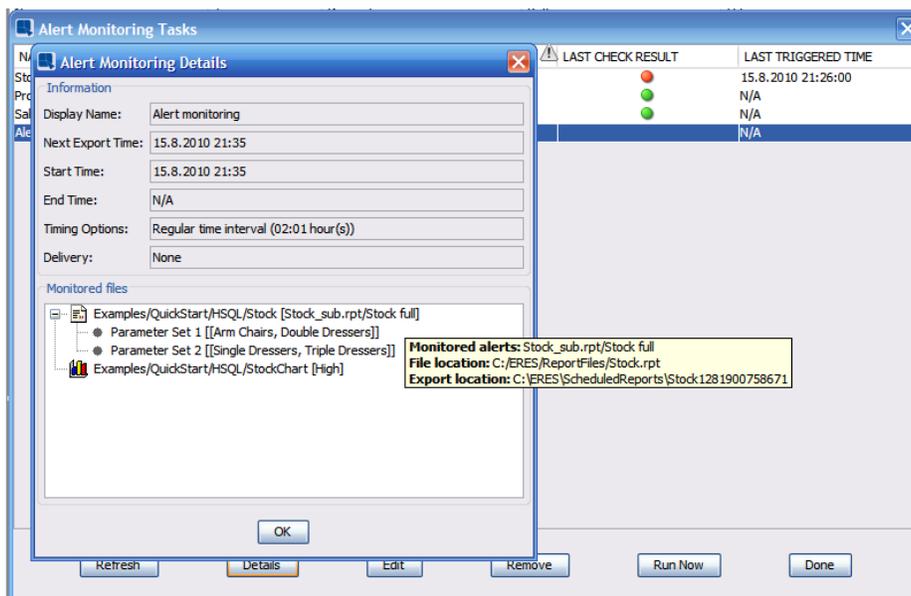
*View alert monitoring tasks dialog*

It also displays the following info about each monitoring task.

- name
- next check time
- last check time
- last check result (alert was triggered or not) – red light means that an alert was triggered, green light means no alerts
- last triggered time

The *Check Now* button is similar to the *Run Now* button in the Schedule list dialog. It checks selected alerts immediately (and sends notifications if necessary).

You can use the *Details* button to view more details about selected monitoring.

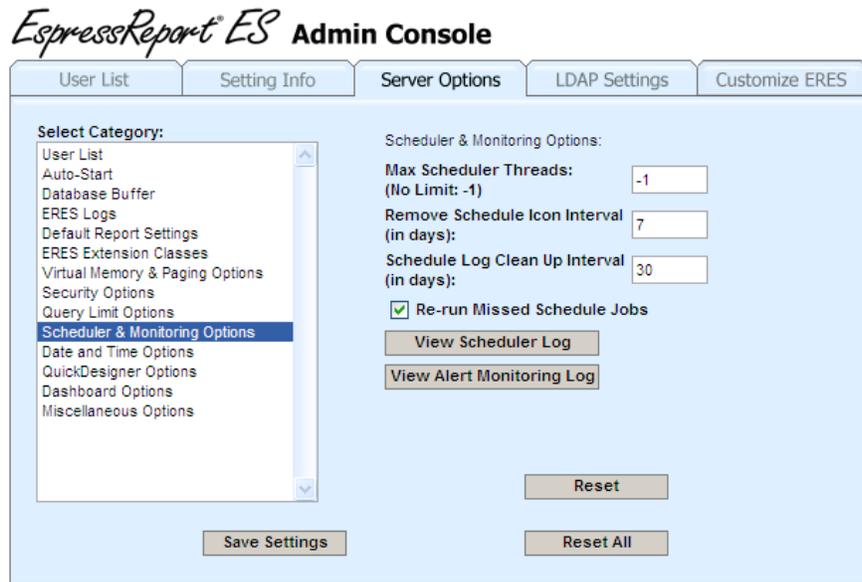


*Alert monitoring details dialog*

The Monitored files list contains list of all monitored files with all monitored alerts (in the square brackets). It also contains all parameter sets if the template is parameterized. If you hold cursor over a template or parameter set in the Monitored files list, you will get a tooltip with more details.

## 11.4.4. Monitoring log

To open the monitoring log, log in as **admin** and click *Administration* to open the Admin Console. Then go to the *Server option* tab, select *Scheduler & Monitoring Options* category and click the *View Alert Monitoring Log* button.



View alert monitoring log button

The monitoring log will pop up in a new window. The log contains monitoring tasks that have been checked at least once in the past.

Name	Created by	Last checked	Next check	Last triggered	Email del.	FTP del.	Success
QuickStart2	admin	11/25/10 16:33:00	12/25/10 16:33:00	At last check!	<input type="checkbox"/>	<input type="checkbox"/>	OK
Template name			Parameter set name	Parameter values	Alert name		
Examples/Dashboards/Alerts Components/Sales By Employee Chart			Parameter Set 1	2002.Q2	No sales		
Name	Created by	Last checked	Next check	Last triggered	Email del.	FTP del.	Success
QuickStart3	admin	11/25/10 16:38:00	12/09/10 16:38:00	Never	<input type="checkbox"/>	<input type="checkbox"/>	OK
Template name			Parameter set name	Parameter values	Alert name		
Examples/Dashboards/Alerts Components/Sales By Employee Chart			Parameter Set 1	2002.Q2	Low sales		
Name	Created by	Last checked	Next check	Last triggered	Email del.	FTP del.	Success
QuickStart4	admin	11/25/10 17:19:00	Finished!	At last check!	<input type="checkbox"/>	<input type="checkbox"/>	NO
Template name			Parameter set name	Parameter values	Alert name		
Examples/Dashboards/Alerts Components/Sales Overview Report			Parameter Set 1	2001.Q1	Low sales		
			Parameter Set 2	2002.Q3	Low sales		
			Parameter Set 3	2001.Q1	Low sales		
					Low revenue		

Alert monitoring log

The most important indicator is whether an alert was triggered at the *last check*. Basically, triggered alerts are highlighted by red color.

There are several possibilities:

Task that doesn't contain any triggered alerts looks like this:

## Alerts

Name	Created by	Last checked	Next check	Last triggered	Email del.	FTP del.	Success
<b>Stock</b>	admin	11/25/10 16:44:00	Finished!	11/25/10 16:43:00	<input type="checkbox"/>	<input type="checkbox"/>	OK
Template name			Parameter set name	Parameter values	Alert name		
	New Project/Stock				Low stock		

*Task with no triggered alerts*

Task that contains at least one triggered alert looks like this:

Name	Created by	Last checked	Next check	Last triggered	Email del.	FTP del.	Success
<b>Quick Start4</b>	admin	11/25/10 17:19:00	Finished!	At last check!	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	NO
Template name			Parameter set name	Parameter values	Alert name		
	Examples/Dashboards/Alerts Components/Sales By Employee Chart		Parameter Set 1	2001,Q1	Low sales		
			Parameter Set 2	2002,Q3	Low sales		
			Parameter Set 3	2001,Q1	Low sales		
	Examples/Dashboards/Alerts Components/Sales Overview Report				Low revenue		

*Task with one triggered alert*

Alert that wasn't triggered at the last check:

Name	Created by	Last checked	Next check	Last triggered	Email del.	FTP del.	Success
<b>Quick Start4</b>	admin	11/25/10 17:19:00	Finished!	At last check!	<input checked="" type="checkbox"/>	<input type="checkbox"/>	NO
Template name			Parameter set name	Parameter values	Alert name		
	Examples/Dashboards/Alerts Components/Sales By Employee Chart		Parameter Set 1	2001,Q1	Low sales		
	Examples/Dashboards/Alerts Components/Sales Overview Report				Low revenue		

*Untriggered alert*

Alert that was triggered at the last check:

Name	Created by	Last checked	Next check	Last triggered	Email del.	FTP del.	Success
<b>Quick Start5</b>	admin	11/29/10 10:17:00	11/29/10 10:18:00	At last check!	<input type="checkbox"/>	<input checked="" type="checkbox"/>	OK - no notif.
Template name			Parameter set name	Parameter values	Alert name		
	Examples/Dashboards/Alerts Components/Sales By Employee Chart		Parameter Set 1	2001,Q1	Low sales		
					No sales		
			Parameter Set 2	2001,Q3	Low sales		

*Triggered alert*

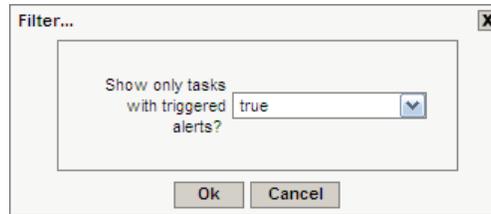
Each task has a *Success* indicator which is not to be confused with triggered/not triggered alerts. The indicator tells whether the last check was performed successfully. There are a few things that can go wrong during checks - monitored templates or alerts could have been renamed or removed, one of the delivery options (FTP or Email delivery) may have failed because of network problem... If a check has failed, the *Success* indicator will look like this: **NO**. You can get more details about failed checks in a drilldown, which will be described later.

A template can have multiple parameter sets and/or multiple monitored alerts. In such situations, the template will take several rows in the template list, but its name will be displayed only once (see the following picture).

Name	Created by	Last checked	Next check	Last triggered	Email del.	FTP del.	Success
<b>Quick Start5</b>	admin	11/29/10 17:25:34	11/29/10 17:26:00	At last check!	<input type="checkbox"/>	<input checked="" type="checkbox"/>	OK - no notif.
Template name			Parameter set name	Parameter values	Alert name		
	Examples/Dashboards/Alerts Components/Sales By Employee Chart		Parameter Set 1	2001,Q1	Low sales		
					No sales		
			Parameter Set 2	2001,Q3	Low sales		
					No sales		

*Parameterized chart with two monitored alerts*

It is possible to exclude tasks without any triggered alerts from the log. To do that, click on the *Filter* icon and set the *Show only tasks with triggered alerts?* parameter to *true*.

*Log filter*

To display detailed report about a task, click on its name. A second level of the monitoring log will open.

Created by:	admin
Next check:	This task has finished
Last triggered:	11/25/10 16:44:00
Delivery settings:	Email delivery: No
	FTP delivery: No
	Send notifications: Always

Check time: 11/25/10 16:44:00				Total success:	OK
Name	Parameter set name	Parameter values	Alert name	Success	
New Project/Stock			Low stock	OK	
Check time: 11/25/10 16:43:00				Total success:	OK
Name	Parameter set name	Parameter values	Alert name	Success	
New Project/Stock			Low stock	OK	
Check time: 11/25/10 16:42:00				Total success:	OK
Name	Parameter set name	Parameter values	Alert name	Success	
New Project/Stock			Low stock	OK	
Check time: 11/25/10 16:41:00				Total success:	OK
Name	Parameter set name	Parameter values	Alert name	Success	
New Project/Stock			Low stock	OK	

*Task details*

The second level log contains results of all alert checks (unlike the root level log which only contains result of the last check). Triggered alerts are also highlighted by red color. The *Success* indicator is displayed for all templates.

If you are wondering why notifications were not sent, always check the *Send notifications* indicator. The indicator has two possible states:

**Always -**

notifications will be sent every time an alert occurs

**First alert occurrence -**

notifications will be sent only when an alert wasn't triggered during the previous check, but was triggered at the current check. So if an alert was triggered in two consecutive checks, notifications would be sent only once. If notifications were not sent because of this feature, the task's *Success* indicator will look like this **OK - no notif.**

To go back to the root level log, click on the  *back* icon.

If the task uses some additional delivery method(s) (FTP or Email), you can click on the *Success* indicator. A third level of the monitoring log will open showing the delivery method's settings and results. If there is no additional delivery method, the *Success* field will not be clickable.

The third log level's appearance may vary. If a notification was performed, its detailed log will open. If a notification wasn't performed (for example: the alert wasn't triggered), only delivery settings will be shown.



## Note

There is a difference between not performed delivery and failed delivery. A delivery isn't performed when the alert wasn't triggered or, in some cases, when the *Send notifications on first alert occurrence only* feature is enabled (as described above). In that case, the ERES server doesn't even try to perform the delivery. If a delivery failed, it means that the ERES server tried to perform the delivery (because the alert was triggered), but it wasn't able to finish (for example: because of a network problem, a human error...).

### Delivery settings

## FTP delivery settings

<b>Name:</b>	Examples/Dashboards/Alerts Components/Sales By Employee Chart
<b>FTP Host:</b>	gamma.quadbase.com
<b>FTP Path:</b>	/reports
<b>FTP User name:</b>	user
<b>Parameter set name:</b>	Parameter Set 1
<b>Parameter values:</b>	2001,Q1
<b>Alert name:</b>	Low sales

*FTP delivery settings*

## Email delivery settings

<b>Name:</b>	Examples/Dashboards/Alerts Components/Sales By Employee Chart
<b>Parameter set name:</b>	Parameter Set 1
<b>Parameter values:</b>	2001,Q1
<b>Alert name:</b>	Low sales
<b>Subject:</b>	Monthly report
<b>From:</b>	clint@quadbase.com
<b>Recipients</b>	peter@quadbase.com, john@quadbase.com

*Email delivery settings*

### Delivery logs

FTP delivery log shows FTP delivery settings + results. If the delivery has failed, an error message is also displayed.

## FTP Delivery log

<b>Run time:</b>	11/25/10 17:19:00
<b>Name:</b>	Examples/Dashboards/Alerts Components/Sales Overview Report
<b>FTP Host:</b>	gamma.quadbase.com
<b>FTP Path:</b>	/reports
<b>FTP User name:</b>	user
<b>Parameter set name:</b>	
<b>Parameter values:</b>	
<b>Alert name:</b>	Low revenue
<b>Success:</b>	<b>NO</b>
<b>Error message</b>	
gamma.quadbase.com	

*FTP delivery log*

Email delivery log shows basic email settings and results for all recipients. In the following picture, you can see that emails from `clint@quadbase.com` to `john@quadbase.com` and `peter@quadbase.com` (not actual email addresses) couldn't be sent because the ERES server wasn't able to connect to the SMTP server at `localhost:25`. It could be caused by a network problem or by incorrect SMTP settings (as described in Section 1.4.1.2 - Setting Info). This is probably the most common problem with email delivery.

## Email delivery log

<b>Name:</b>	QuickStart4
<b>Check time:</b>	11/25/10 17:19:00
<b>From:</b>	clint@quadbase.com
<b>Subject:</b>	Monthly report

<b>To:</b> john@quadbase.com				
<b>Name</b>	<b>Parameter set name</b>	<b>Parameter values</b>	<b>Alert name</b>	<b>Success</b>
Examples/Dashboards/Alerts Components/Sales Overview Report			Low revenue	NO
<b>Error message</b>				
Sending failed; nested exception is: class javax.mail.MessagingException: Could not connect to SMTP host: localhost, port: 25; nested exception is: java.net.ConnectException: Connection refused: connect				

<b>To:</b> peter@quadbase.com				
<b>Name</b>	<b>Parameter set name</b>	<b>Parameter values</b>	<b>Alert name</b>	<b>Success</b>
Examples/Dashboards/Alerts Components/Sales Overview Report			Low revenue	NO
<b>Error message</b>				
Sending failed; nested exception is: class javax.mail.MessagingException: Could not connect to SMTP host: localhost, port: 25; nested exception is: java.net.ConnectException: Connection refused: connect				

*Email delivery log*

## 11.5. Handling special situations

### 11.5.1. Change in script/control area/range

If for some reason an alert is removed or renamed, the monitoring job or dashboard alert cannot find the missing alert. The following approaches will be taken to resolve the situation.

- For monitoring – An error email is sent. An explanation is included, e.g. alert <ALERT\_NAME> not found in the template <TEMPLATE\_NAME>.
- For dashboard alert – An error icon is displayed in the template header. If you mouse over the error icon, you will see error message explaining the problem. If you are using the Dashboard Builder at that time, you can fix the problem immediately.

## 11.5.2. Missed monitoring jobs

If a monitor check is missed because server was not running, it will run immediately after the server is started. This is analogy to missed scheduled jobs. You can turn this feature on/off using *Re-run Missed Schedule Jobs* option in the Admin Console. Visit Section 1.4.1.3 - Server Options for more information.