CHAPTER 5

Creating Dashboards

The following topics tell you how to create and manage Prime Analytics dashboards:

Overview to Prime Analytics Dashboards

Dashboards allow you to display your data in many different ways so that you can interpret data quickly and effectively. Prime Analytics dashboards are built using open source components and applications, including the Pentaho CTools framework. CTools applications and tool sets include

- Community Dashboards Framework (CDF)
- Community Chart Components (CCC)
- Community Data Access (CDA)
- Community Dashboard Editor (CDE)

Pentaho CTools information and documentation is available from the Pentaho Community website: *http://community.pentaho.com/.*

Creating a dashboard in Prime Analytics is accomplished using the following processes:

1. Lay out a grid for your dashboard.
2. Identify the data sources to be visualized in the dashboard.
3. Create the dashboard visual components, connect them to the data sources, and assign them to the layout grid.
4. Improve and modify the dashboard; add flexibility by parameterizing components and letting components interact with one another.

The steps can be completed in any order. In fact, beginning with a basic dashboard and then adding additional components and interactivity works best as you gain experience.

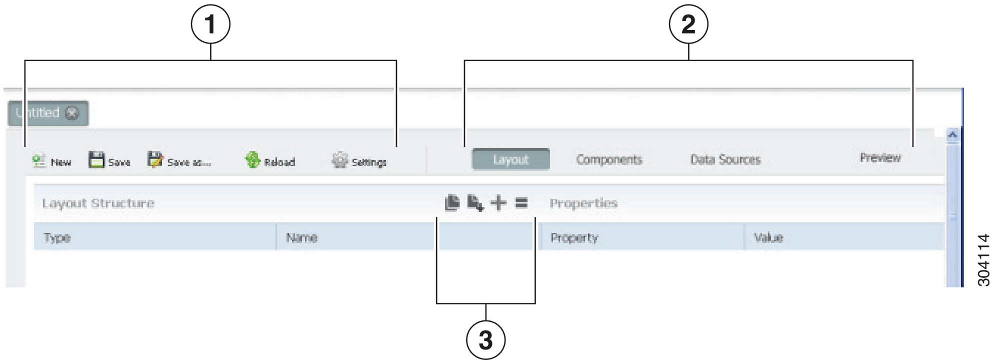
Navigating the Dashboard Workspace

To launch the dashboard workspace, from the File menu, choose **File > New > Dashboard**.

The dashboard workspace appears (Figure 5-1). The workspace has the following general components:

- Top level menu—Provides actions that affect the overall dashboard:
  - New—Creates a new dashboard.
  - Save—Saves the dashboard.
  - Save as—Allows you to save a dashboard under a new name or in a different directory.
  - Reload—Reloads the dashboard.
  - Settings—Displays dashboard settings. Settings include the dashboard title, author, and description. It also shows the dashboard style and type. Currently, only one style, Clean, and one dashboard type, blueprint, are available.
- Dashboard work Areas—All elements you work with to create your dashboard are contained in three work areas:
  - Layout—The work area where you define the dashboard layout.
  - Components—Contain all elements that you can add to the dashboard: charts, parameters, scripts, selectors, components, and other elements.
  - Data Sources—Define the sources of data used in the dashboard.
  - Preview—Allows you to preview your dashboard during its development.
- Toolbar—Provide common actions used during dashboard creation. The actions vary, depending on what work area you are working in. For example, in Layout, toolbar options are Save, Save as template, Add Resource, and Add Row. In Components and Data Sources, the toolbar options are Duplicate Component and Delete.

Figure 5-1 Dashboard Workspace



1	Top level menu	**3**	Work area toolbar
2	Work areas		

Creating a New Dashboard

In the following procedure, you will create a new dashboard and set up the dashboard layout.

To create a new dashboard:

Step 1 Log into the user console. For procedures, see Navigating the Dashboard Workspace, page 5-2.

Step 2 Review the folders under Browse. If you will store your dashboard in one of them, continue with the next step. If you need to create a new directory:

a. Right-click the Browse area and choose **New Folder**.

b. In the New Folder dialog, enter the new folder name, then click **OK**.

Step 3 From the File menu, choose **File > New > Dashboard.**

A new dashboard is created.

Step 4 From the File menu, choose **Save as**.

Step 5 In the Save as dialog box:

a. Choose the directory where you want to save your file. All dashboard files must be stored in a Sample subdirectory.

Note All dashboards are associated with the Prime Analytics installation. You cannot move or edit dashboard files manually because they require bipuser permission.

b. In the File Name field, enter the dashboard file name.

c. (Optional) In the Title and Description fields, enter the dashboard title and description.

d. Click **OK**.

Step 6 From the View menu, choose **Refresh**.

Your new directory is displayed in the directory where you placed it.

Step 7 Close the Untitled dashboard.

Step 8 Under Browse, select the folder containing your new dashboard.

Step 9 Under Files, select the new dashboard, then click the **Edit** tool.

Note If you choose Open, the blank dashboard will appear, but you will not be able to edit it.

Your dashboard appears in the work area with dashboard edit tools available. The dashboard tab name is Edit: *dashboardtitle*.

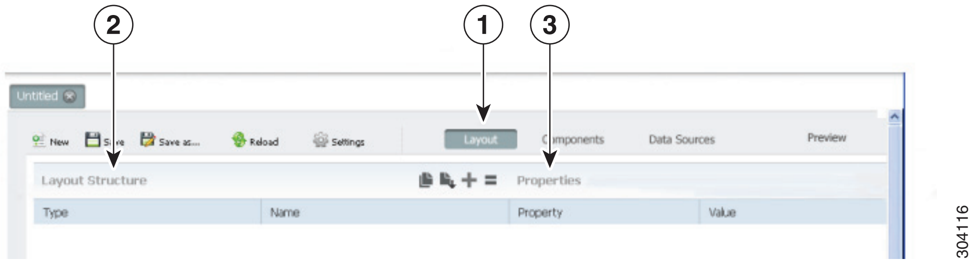
Step 10 Continue with the following topics:

- Setting Up the Dashboard Layout, page 5-4
- Adding Components to Dashboards, page 5-7
- Adding Data Source Queries to Dashboards, page 5-12

Setting Up the Dashboard Layout

When you first open a dashboard, the layout area is displayed with an empty layout (as shown in Figure 5-2). The layout workspace consists of two areas, Layout Structure and Properties. Layout Structure displays the current layout design. Properties displays the properties of the element currently is selected under Layout Structure.

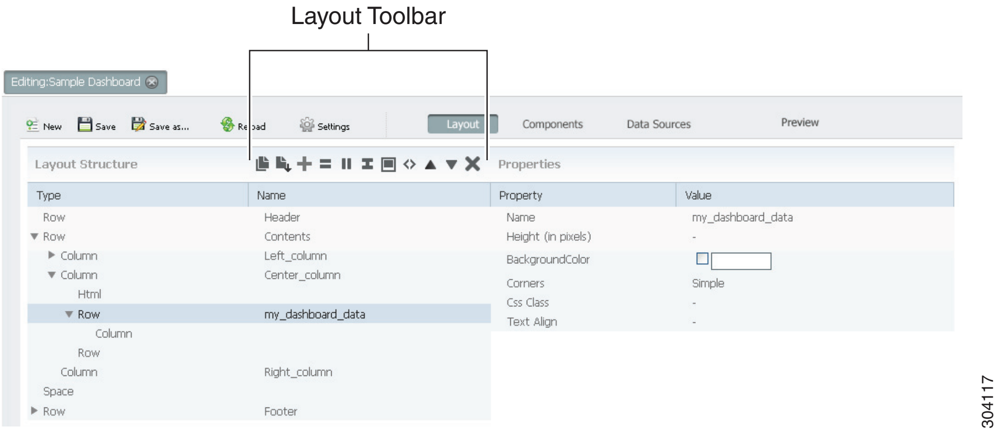
Figure 5-2 Layout Workspace



1	Layout workspace	**3**	Properties
2	Layout Structure		

Figure 5-3 shows the Layout workspace with a simple page design. The left pane shows that the page is divided in three rows labeled Header, Contents, and Footer. The Header row is a simple row containing no elements. The Contents row has several elements. These elements are expanded. You can expand or collapse elements by clicking the triangle located to left of the row element. In the example, the Footer row is collapsed. The triangle in front points to the right, indicating that this row contains collapsed elements.

Figure 5-3 Dashboard Workspace With Sample Layout



Type	Name
Row	Header
▼ Row	Contents
▶ Column	Left_column
▼ Column	Center_column
Html	
▼ Row	my_dashboard_data
Column	
Row	
Column	Right_column
Space	
▶ Row	Footer

Property	Value
Name	my_dashboard_data
Height (in pixels)	-
BackgroundColor	□
Corners	Simple
Css Class	-
Text Align	-

Layout Toolbar

Use the layout toolbar to customize your layout. The toolbar is dynamic, and displays between two to eleven tools, depending on the layout objects that you select. Actions that you can perform include:

- Save as template—Saves the dashboard as a template. The template includes the complete layout; you can choose whether the template also includes the components and data sources that are defined in the Components and Data Sources workspaces. After you save the dashboard as a template, a new empty dashboard is displayed.
- Apply template—Allows you to apply a template to the dashboard. Eight standard layouts are provided, including two, three, and four column templates, and five others.

Note The Apply Template tool is only available when you display a new template with no added elements. If you save a it as a template, you must open it as a normal dashboard file.

- Add resource—Adds a cascading style sheet (.css) or a JavaScript (.js) resource to the dashboard.
- Add row—Adds a row to the layout. If the currently selected element is a column, the row is added as a column element. Otherwise, the row is added at the same level as the selected element.
- Add column—Adds a column to the layout. If the current element is a row, the column is added as a column element. Otherwise, the column is added at the same level the selected element.
- Spacer—Add space between subsequent elements. If a row is selected, a vertical spacer is added. If a column is selected, a horizontal spacer is added.
- Add image—Adds an image to the current select row or column element. When you select this action, a dialog appears
- Add HTML—Adds an HTML block to the selected row or column element.
- Move up—Moves the current element one step up in the list (while remaining at the same level).
- Move down—Moves the current element one step down.
- Delete—Deletes the selected element.

Layout Properties

Layout properties also vary, depending on the element that you select. The first property in Figure 5-3 is Name. This property sets the element name. This name is shown in the Name column of the layout structure as well, and can also be used to attach components to the layout cells. You can use any alphanumeric character in a name. Underscores (_) are the only permissible special character in a name.

The dashboard layouts are based on the blueprint.css stylesheet. To minimize the need to manually adjust widths, blueprint.css divides a screen into 24 columns. If you create a column in the layout workspace it has three blueprint.css parameters:

- Span size—The column width.
- Prepend size—The amount of empty space to insert before the column.
- Append size—The amount of empty space to insert after the column.

The layout workspace width is set in blueprint.css columns. Although you can force the width of a component in pixels in the component view, it is not recommended. The recommended practice is to leave the component width empty so its width can be determined from its contents and the cell layout (<div>) where the component is placed.

If the combined width of the columns on a line exceed the page width, the columns that do not fit are wrapped to the next line. The blueprint.css does not provide a height requirement. Column height is based on the height of its contents. Therefore, an empty column is invisible as it has a height of 0 pixels (unless you explicitly set the column height). For more information on the blueprint.css framework see http://www.blueprintcss.org.

Color Editor

Layouts include an option to set the background color property of rows and columns.

To set the background color:

Step 1 In the Background Color property, click the small box.

The color editor is displayed.

Step 2 Select the Hue (color range) by drawing a tiny rectangle on the rainbow selector in the middle to the appropriate color range,

Step 3 Select the saturation and brightness by drawing the white circle in the square box to the exact color you want (initially this circle is located in the left bottom of the box), and

Step 4 Click the color circle located in the lower right corner of the color editor to confirm the selection.

Note You can also enter the RGB, HSB or hexadecimal color codes in the R, G, B, or H, S, B, or # fields.

Working in the Layout Workspace

As you work with the dashboard editor Layout workspace, keep the following in mind:

- When entering a property, always press Enter to save your property. If you do not press Enter and select another layout element, the editor discards the value you entered without notice. Because you changed to a new layout element, you might not notice your data entry was discarded. If you use your mouse to select another property, the data entry for the preceding property is accepted.
- Some fields are completed automatically. These fields are identified with a drop-down list below the field.
- Use care when adding a multiple columns or rows simultaneously. As soon as you add an element the toolbar might change. For example if you start with an empty layout, the Add Row button is in the far right toolbar position. However, as soon as a row is added, the Delete tool added at the far right position of the toolbar. Double clicking the far right tool in an empty layout adds a row and immediately deletes it with the second click, resulting in an empty layout.
- The Properties area shows the properties of the layout currently-selected element. These properties vary by element, so the list changes dynamically if you move to another element in the layout structure.

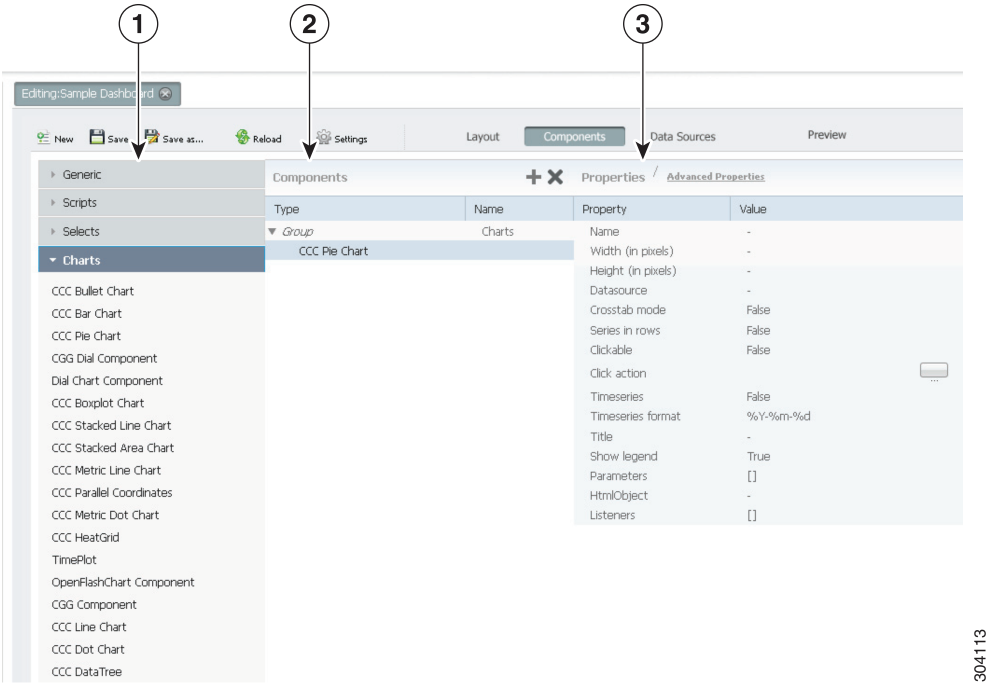
Adding Components to Dashboards

After you set up your dashboard layout (see Setting Up the Dashboard Layout, page 5-4), you can begin adding components to it. A wide range of components are available, from scripts and charts to buttons and maps. Familiarizing yourself components will take time. These topics will provide a general overview.

To add components, within a dashboard, choose **Components**.

The component workspace, shown in Figure 5-4, is displayed.

Figure 5-4 Components Workspace



Type	Name
Group	Charts
CCC Pie Chart	

Property	Value
Name	-
Width (in pixels)	-
Height (in pixels)	-
Datasource	-
Crosstab mode	False
Series in rows	False
Clickable	False
Click action	
Timeseries	False
Timeseries format	%Y-%m-%d
Title	-
Show legend	True
Parameters	[]
HtmlObject	-
Listeners	[]

1	Components list	**3**	Properties
2	Selected components		

This workspace has three areas. The left area displays components classes containing components that you can add to your dashboard. The middle area shows the components that have been added to the dashboard, and the right area shows the properties of the components selected in the Components list.

Component Categories Area

Available components are divided into eleven categories: Generic, Scripts, Selects (#1), Charts, Others, Widgets, Real Time Charts, Community Contributions, CDF Core Functionality, Custom, and Selects (#2). Many components are provided with the Pentaho Community Tools (CTools) application. Pentaho CTools acronyms referenced in the components include:

- CCC—Community Chart Components)
- CDE—Community Dashboard Editor

- CGG—Community Graphics Generator
- CDF—Community Dashboard Framework

For more information about Pentaho Community Tools, see http://community.pentaho.com.

Dashboard components provided with Prime Analytics include:

- Generic—Contains the following parameters:
  - Simple parameter—Adds a JavaScript parameter that sets a primitive default value, such as integer, double, string, or others. You can use JavaScript parameters with other parameters to connect components.
  - Custom parameter—Adds a JavaScript parameter that sets a compound-value default, for example, array, object, function. It can also be used when the value is initialized by another Java function.
  - Date parameter—Sets a date. This can be a specific date, for example today's date, or it can be linked to a date picker.
- Scripts—Adds a script to the dashboard. The provided JavaScript Function script defines a JavaScript function.

  Note If you have a long function or multiple functions to add, you can also place the JavaScript file on the server and include file as a layout workspace resource.

- Selects—Adds single-function elements:
  - MultiButtonComponent—Adds a multibutton component.
  - SelectMulti Component—Adds a multiple selection component. Users can choose one or more elements from a list and place them in a JavaScript variable.
  - TextInput Component—Text input without auto-completion.
  - Date Range Input Component—Adds a date range input. However, many servers do not permit this SQL query, as the client-side SQL introduces a security risk.
  - Date Input Component—Allows users to add a date.
  - Check Component—Lists one or more check boxes and connects them to a JavaScript variable.
  - Radiobutton Component—Adds a radio button that selects one item from a list of items.
  - TextareaInput Component—Adds an area where text can be entered.
  - Auto complete Component—Adds a text entry component with auto completion capability.
  - Month Picker Component—Adds a month selection element.
  - Simple Auto Complete Component—Adds a simple text entry component with auto completion capability.
  - Select Component—Presents a list box to select a parameter. It returns the selected parameter or its display value.
- Charts
  - CCC Bullet Chart
  - CCC Bar Chart—See CCC dot chart.
  - CCC Pie Chart—See CCC dot chart.
  - CGG Dial Component

  - Dial Chart component
  - CCC Boxplot chart—See CCC dot chart.
  - CCC Stacked Line Chart—See CCC dot chart.
  - CCC Stacked Area Chart—See CCC dot chart.
  - CCC Metric Line Chart—See CCC dot chart.
  - CCC Parallel Coordinates Chart—See CCC dot chart.
  - CCC Metric Doc Chart—See CCC dot chart.
  - CCC HeatGrid—See CCC dot chart.
  - TimePlot
  - OpenFlashChart component—Component using OpenFlashChart for visualization.
  - CGG Component
  - CCC Line chart—See CCC dot chart.
  - CCC Dot chart—the CCC charts are the preferred charts in the CDE because these charts look pleasant and have many features for interaction, like tooltips, animation and drill-down options.
  - CCC Data Tree
  - CCC Waterfall Chart
  - Chart component—Component using the Jfree chart graphics.
  - CCC 100% Stacked Bar Chart
  - Protovis component—A graphical toolkit for visualization. The CCC-chart are also based on Protovis. However, this components gives you access to the full power of the protovis library, which is a powerful javaScript library for a wide range of visualizations.
- Others—Provides a variety of additional components:
  - Freeform Component
  - Related Content Component
  - Pivot Link Component
  - Mobile Navigation Component
  - Xaction Component
  - Execute XAction Component
  - Export Button
  - Text Component—Shows the result of an expression, where the expression is provided by a JavaScript function.
  - Query Component— Executes a SQL or MDX query and returns the result.
  - PRPT Component—Executes the Pentaho Reporting unified format.
  - Table Component—Shows the contents of a data source in table format. By default, tables are sorted, searchable, and paginated. An extensive options list to customize the table component is provided.
  - Button Component
  - Comments Component—Adds a comments section to a page.
  - Popup

  - Traffic Component—Generates a traffic light image without needing an xaction file.
  - Duplicate Component
  - Execute Prpt Component—Executes a PRPT and displays the result in a new popup window.
  - Pivot Component—Executes a JPivot Action Sequence and creates an iframe where the pivot table is embedded.
  - ExportPopupComponent
  - Navigation Menu Component—Introduces a menu that allows you to browse the complete Pentaho solution repository (non-hidden folders).
- Widgets—Provides a single, general-use widget.
- Real Time Charts—Charts designed for Prime Analytics continuous query data streams.
  - Real Time Line Dual Y
  - Real Time Column
  - Real Time StackedArea
  - Real Time Line
  - Real Time Angular Gauge
- Community Contributions—Provides components provided by the Pentaho community. In this release, a Google Analytics component is provided.
- CDF Core Functionality—Provides the Pentaho CDF View Manager Component component:
- Custom—Provides a set of custom components:
  - Raphael component—Raphael is a library for Scalable Vector Graphics (svg).
  - NewMapComponent
  - AjaxRequestComponent
  - SiteMap
- Selects—Provides an additional OLAP Selector component.

Components Area

The selected components area shows the components that you added to the dashboard. These component are grouped by category. In the example in Figure 5-4, one component is added, a CCC Pie chart that belongs to the group charts. The middle pane has a dynamic toolbar. Two tools are shown: Duplicate Component and Delete. If more charts were added, the Move Up and Move Down tools are added.

Properties Area

The component properties area shows the properties of the selected component. Next graph shows component workspace. The available list of properties is already fairly long. For the CCC Pie chart in figure 6 this list contains 15 properties that can be set. If you click Advanced Properties, the advanced properties are added to the list. Properties that appear for many components include:

- Name—The component name. This name is also in the components list.
- Width—The preferred component width in pixels. By default the width is based on the layout you selected in the layout workspace. Only modify this option if the layout does not provide your component with sufficient space.
- Height—The component height in pixels. Only modify this parameter if the defaults do not provide sufficient room.

- Datasource—The data source defined in the Data Source workspace to which the component is connected.
- Parameter—The JavaScript parameters used by the component.
- HtmlObject—The name of the layout container defined in the Layout workspace that displays the component.
- Listeners—A list of JavaScript variables that this component watches. If the value of one of these variables changes, a redraw of the component is triggered.

Adding Charts to Dashboards

Below are the general steps associated with creating a chart:

1. Select a data source.
2. Build a query.
3. Set the data definitions: values, series, category.
4. Select a chart type and theme.
5. Enter labels for the chart title, and X and Y axes.
6. If applicable, adjust scaling and label rotation.
7. Place your chart in the dashboard.
8. Save your dashboard.

If you are new to charting, here are guidelines that may help you determine what type of chart is best suited for the data you want to present in your dashboard:

Bar Charts

Bar charts are useful when comparing items during a specific time period. Key words to think about when creating a bar chart are compare or rank. For example, if you want to compare items sold to show which one made the most profit, you can create a bar chart that ranks the products from the lowest to highest profit. The bar's length determines its ranking; the label identifies the item. Bar chart data can be presented horizontally or vertically depending on your requirements.

Pie Charts

Pie charts are useful when comparing parts of a whole. Key words associated with charts include, portion, share, and percentage. For example, if you want to demonstrate the proportion of the company's budget spent on health insurance, use a pie chart. To make the chart easier to read, limit the number of slices to five. Pie charts can also be exploded, which means certain slices are pulled away from the remainder of the chart for emphasis.

Line Charts

Line charts are useful for showing changes over time. Key words associated with data that is best suited for a line chart are trend, growth, and decline. If, for example, you want to show how product sales have changed over five years, use a line chart. The slope of the line helps users quickly identify the direction of the trend.

Dial Charts

Dial charts are often associated with Key Performance Indicators (KPIs). Dial charts are circular and contain a scale, a needle, and one or more a dial sectors. The dial sector is used to identify a specified area on a dial chart using a particular color. For example, you could have a dial plotting inventory with a minimum dial value of 10000 and a maximum dial value of 50000. There could be a red dial sector for the region between 2000 and 4000 indicating that if the needle is in this area, there is a danger of a supply inventory shortage.

Area Charts

Area charts can be used to show a comparison of the same item during different points in time. Area charts are not designed to provide exact data; they provide users visual clues of the relative sizes of the items they represent.

Real Time Charts

Prime Analytics includes real time charts designed for continuous query data. These include:

- Real Time Line Dual Y
- Real Time Column
- Real Time Stacked Area
- Real Time Line
- Real Time Angular Gauge

Working with the Components Workspace

Most properties have an auto-completion function. When you edit such a property, a list pops up containing the available options. The list is context-sensitive. For example, if you click the HtmlObject, a list of all layout containers appears. However, if you click the parameter property, it will show a list of defined JavaScript variables. (This list will be limited to the JavaScript variables that you defined using the parameter components. It will not include parameters defined in your own JavaScript resources that you included in the dashboard editor.) When you enter a property value, the dashboard editor only does a basic validation. If you include a value that does not exist, the editor might not raise an exception. Also when you preview a dashboard, the editor might not call out errors. The only notification you might get is when your component is not drawn. Therefore, use the auto-completion function whenever possible. Most components have many properties. If a component fails, you might need considerable time to track down the property value that you misspelled.

The dashboard editor does not consistently warn you when you make an error in a component. The main error message is the failure of component render on the screen. Therefore:

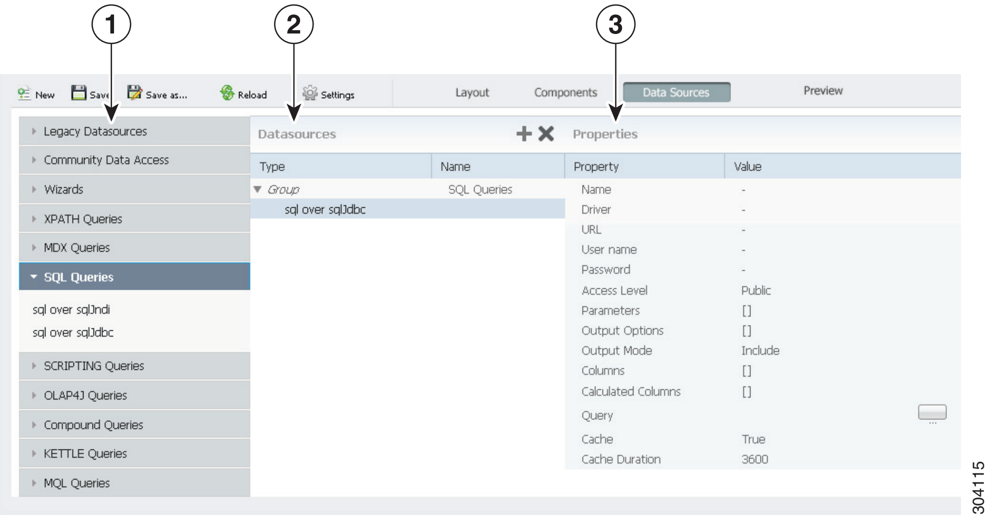
- Be very careful to avoid making errors.
- Preview your dashboard any time you make a significant change. Doing frequent previews takes little time, but can save you much time later because identifying the error is easier if you only made one or two small changes. If you save your file after each successful preview, you can easily backtrack to the previous version using the reload when you observe an error.

Adding Data Source Queries to Dashboards

The Data Sources workspace, shown in Figure 5-5, is where you add the data source queries to your dashboard components. The Data Sources workspace has the same layout as the Components workspace. Available data source queries are displayed in the left area, the middle area shows the data source query that you have created for the dashboard and the right area shows the properties of the current selected data source query.

> **Note** The dashboard Data Source workspace is only used for queries on static databases. For continuous queries databases, see Adding Charts for Real-Time Data Sources, page 5-16.

Figure 5-5 Data Sources Workspace



Type	Name
Group	SQL Queries
sql over sqlJdbc	

Property	Value
Name	-
Driver	-
URL	-
User name	-
Password	-
Access Level	Public
Parameters	[]
Output Options	[]
Output Mode	Include
Columns	[]
Calculated Columns	[]
Query	
Cache	True
Cache Duration	3600

1	Data source list	**3**	Properties
2	Selected data source		

Available Data Source Queries

The left area of the Data Sources workspace displays the data source queries and other processing that you can add to the dashboard. These include:

- Legacy Data Sources
  - OLAP MDX query
  - SQL query
  - Xaction result set
  - Kettle transformation
- Community Data Access
  - CDA data source—Includes an existing .cda file in your dashboard
- Wizards
  - OLAP members wizard
  - OLAP Chart wizard
- XPATH Queries—XML Path (XPath) queries. XPath is a query language used to select nodes from an XML document.
  - Xpath over Xpath
- MDX Queries—MultiDimensional eXpressions queries. MDX queries provide a specialized syntax for querying data in Online Analytical Processing cubes.

  - mdx over mondrianJndi—MDX over Mondrian JNDI.
  - denormalizedMdx over mondrianJdbc—Denormalized MDX over Mondrian JDBC.
  - denormalizedMdx over mondrianJndi——Denormalized MDX over Mondrian JNDI.
  - mdx over mondrianJdbc——MDX over Mondrian JDBC.
- SQL Queries
  - Sql over sqlJdbc—Access a database using the Java DataBase Connectivity API (JDBC).
  - Sql over sqlJndi—Access a database using the Java Naming and Directory Interface (JDNI).

Note Many data source queries are available in JDBC or JNDI. The JDBC API provides universal data access from the Java programming language. Using the JDBC API you can access virtually any data source, from relational databases to spreadsheets and flat files. JDBC also provides a common base on which tools and alternate interfaces can be built. JNDI is part of the Java platform. It provides applications based on Java technology with a unified interface to multiple naming and directory services.When using the JNDI data sources, define the data source in the Prime Analytics Administration Console. If you must access several tables from a database, use JDNI because it is referenced using a single parameter, whereas a JDBC data source requires four parameters (database name, connection-URL, username, and Password) to connect to the database server.

- Scripting Queries
  - Scriptable over scripting—Currently the only supporting scripting language is BeanShell, which is a small and embeddable Java source interpreter.
- OLAP4J Queries—OLAP4J queries. OLAP4J is a common API for any OLAP server.
  - Olap4J over olapJdbc—OLAP4J over OLAP JDBC.
  - Olap4J over olapJndi—OLAP4J over OLAP JNDI.
  - denormalizedOlap4j over olapJdbc—Denormalized OLAP4J over OLAP JDBC.
  - denormalizedOlap4j over olapJndi——Denormalized OLAP4J over OLAP JNDI
- Compound Queries
  - Join—Joins two data sources. This is often used to join tables residing in different databases.
  - Union—Concatenate two data sources that have the same column format.
- KETTLE Queries—Pentaho Data Integration Community Edition queries.
  - Kettle over kettleTransFromFile
- MQL Queries—Metaweb Query Language queries.
  - Mql over metadata

Data Source Area

The Data Source area in the middle of the workspace shows the current data sources defined for the dashboard. Above the data source list you have the same dynamic toolbar as described for the components workspace, allowing you to delete or duplicate data sources, or move them up or down.

Data Source Properties

The Data Source Properties shows the selected data source properties. Properties that appear for most data sources include:

- Name—The data source name. This name is also be shown in the middle area. It is the name used to make a connection to the data source from components in the Component workspace.
- Access Level—The data source access level.
- Parameters—JavaScript parameters that you can use to create a parameterized data sources

Note The parameterization is limited by the underlying data engine. In an SQL query you can use the parameter to limit the result set (WHERE statement) or to change grouping or ordering. However, in SQL you cannot parameterize the columns that you want to retrieve.

- Output Options—The data source output options.
- Output Mode—The data source output mode.
- Columns—The data source columns.
- Calculated Columns—The data source calculated columns.
- Query—The query used to extract data from data source.
- Cache——The data source cache.
- Cache Duration——The data source cache duration.

Working With the Data Source Workspace

The dashboard editor does not check your queries for errors. If your query has an error, the query usually fails. Errors returned by the database engine are not shown to you. The only feedback you receive is the failure of the component using the data to render. Therefore, the following practices are recommended:

- You copy and paste the query to your database/mdx client to test whether this query is valid and returns the correct result set.
- Use the CDA preview to verify the data source to see whether you get the proper results. An error in the data source settings might prevent the data source from returning the right result even when the query is correct. The CDA definitions are stored in a separate file (the .cda file). This file can be opened by clicking it in the user console URL field. However, there are two caveats:
  - Always save the CDE; otherwise the new definition is not be stored in the .cda file. It is stored in a temporary file that is not visible in the file browser.
  - If the data-source is executed, either in the CDA preview, or because you used the CDE preview, the results are stored in the CDA cache. The results are also cached if the query returns zero records due to an error in your query. Always save the dashboard to clear the CDA cache after you repair or modify a data source because some versions of the CDA do not use the temporary CDA file22. Clearing the CDA cache is needed in some CDE versions because the CDA does not detect that the .cda file is more recent than the cached results.

Adding Charts for Real-Time Data Sources

Complete the following steps to add data sources for real-time components:

Step 1 Log into the Prime Analytics user console as an administrator or designer user. See Logging Into the User Console, page 3-1.

Step 2 From the File menu, choose **New** > **Dashboard**.

Step 3 In the Layout workspace toolbar, click **Apply Template**.

Step 4 In the template dialog box, select a template and click **OK**.

Step 5 Click **Components**.

Step 6 Click **Real Time Charts** and choose one of the following charts:

- Real Time Line DY
- Real Time Column
- Real Time Stacked Area
- Real Time Line
- Real Time Angular Gauge

Step 7 For the chart you selected, enter the following minimum properties:

- Name
- Width
- Height
- HtmlObject.

Step 8 Next to Path, click **View SQL** to display a list of available continuous SQL connections.

Step 9 Click on **Path** and select the continuous SQL (CDA file).

Step 10 Enter **1** in DataAccess ID.

Step 11 On the dashboard toolbar, click **Save as**…" to save the dashboard in a location of your choosing.

Step 12 Click **Preview** to view the results.

Note The workflow for real-time charts differs from other components. The dashboard editor Data Sources workspace is not used for real-time components.

Improving Dashboard Performance

As you work with dashboards, you might find the need to improve performance in certain situations. The following provides some performance improve dashboard performance, For example, to retrieve records you use: select cq_close(*), count(*) from event <slices '5 seconds'>. Each time you get 2000 records. To make this more manageable, use an advanced property on the user console to set the maximum number of data points to plot. If you set it 20, it will show 20 records each time.

Because 2000 is much more than 20, a performance issue might arise. To solve this, add a limit to the SQL, for example, select cq_close(*), count(*) from event <slices '5 seconds'> limit 20. However, setting the limit to 20 means you cannot see 90% of the records. While no ideal solution exists, to see more data you can increase the maximum number of data points to plot, for example, 100. However, if this setting is too large, individual data points might be difficult to see on the dashboards. Ultimately, you will need to find a balance, so experiment to find the best balance.

Another performance improvement suggestion is to remove charts that aren’t used. For example, you add a static chart, such as a pie chart, to a component. You might also add a JDBC data source component to the chart. You might later decide you don’t want to see the chart show and remove the HtmlObject relation, such as Panel_1, from property while leaving the chart still in component.

The next time you run the dashboard, the chart will not be visible. However, it will still have a performance cost because of the JDBC data connection. Therefore, if you decide you do not want to display a chart, always delete it from the component to avoid drains on performance.