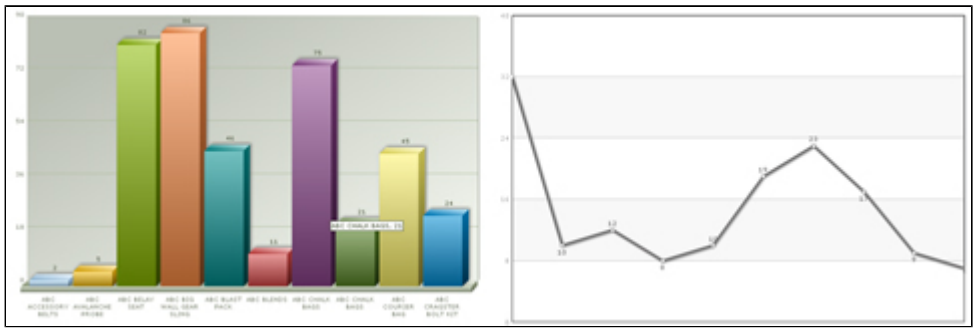
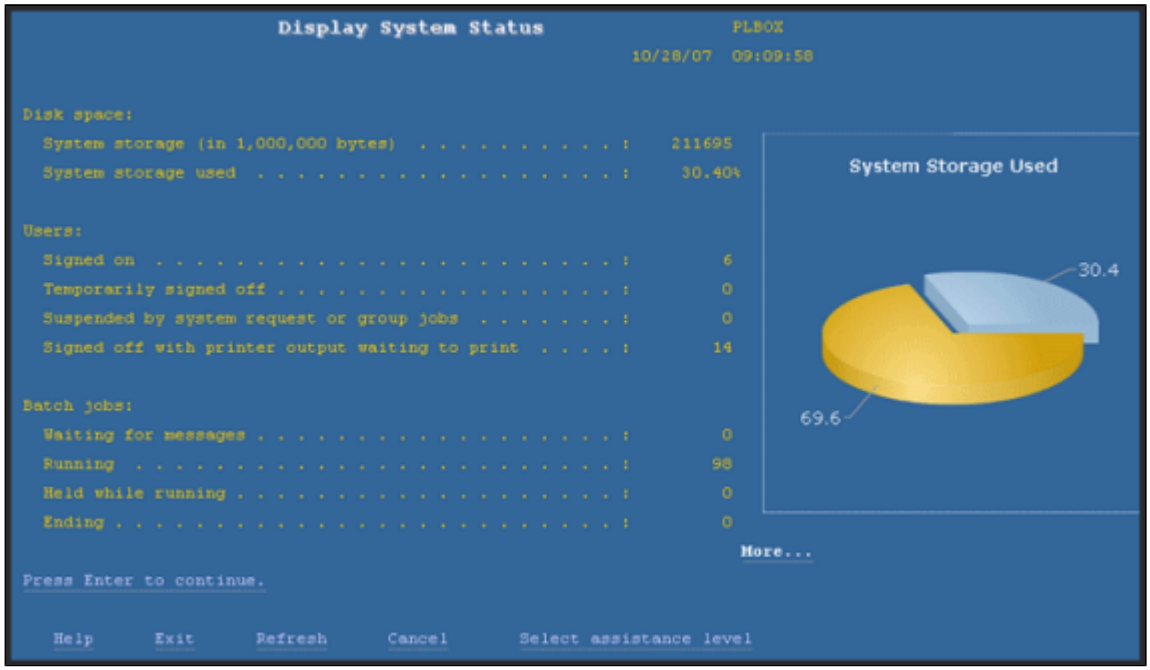
Genie Chart

Genie is packaged with many different types of HTML5 charts, and they are extremely easy to integrate with your applications. Simply pick the chart type and point it at any DB2 table on the IBM i for chart data. Alternatively, if you want to generate chart data on the fly, this can be done in any Web capable language that can produce XML or JSON. Languages such as RPG, PHP, and Java may be used. If you own RPGsp, it provides chart wizards for creating RPG programs that generate the appropriate XML on-the-fly.



Additionally, you can generate charts from data displayed on the screen rather than using data from a database table.

Shown below is a Display System Status screen in which the data displayed on the screen is used to generate a pie chart for System Storage.



Genie Charts are based on FusionCharts. For more information, click here.

Creating a New Chart

A new Chart can be created by selecting Chart on the Widgets menu on the left side of the screen while in Design mode.

A new resizable chart placeholder will be created with a template preview. You will now need to edit the Chart Settings.

Chart Settings

Select a "chart type" from the dropdown in the Properties Window. Or select Other... in the "chart type" property and type in a valid FusionChart name manually.

You can also control the chart overlay. When the chart overlay is set to true, the Chart panel will overlay any other content on the screen, regardless of z-index settings. When set to false, the Chart panel will behave according to normal layering rules, based on z-index. The default value is false.

The next step is to identify the data source of the chart. The data could be provided through three different methods:

(1) Chart Data From Screen

(2) Database-Driven Chart Data.

(3) Dynamic Chart Data.

In the next sections, we will demonstrate each one of the data sources for the Chart.

Chart Data from Screen

In this section, we will chart the data from the screen through the help of the designer.

This could actually be accomplished in three different ways:

(1) Manual data entry through the Chart Properties Window.

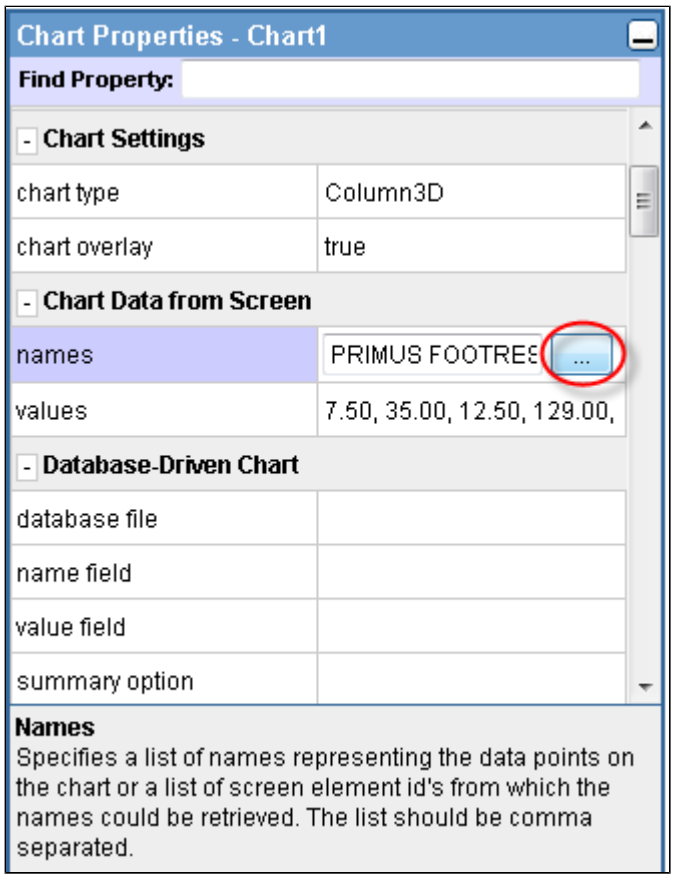
(2) Manual data entry of element names/IDs on the screen to chart.

(3) JavaScript expression that generates comma separated data.

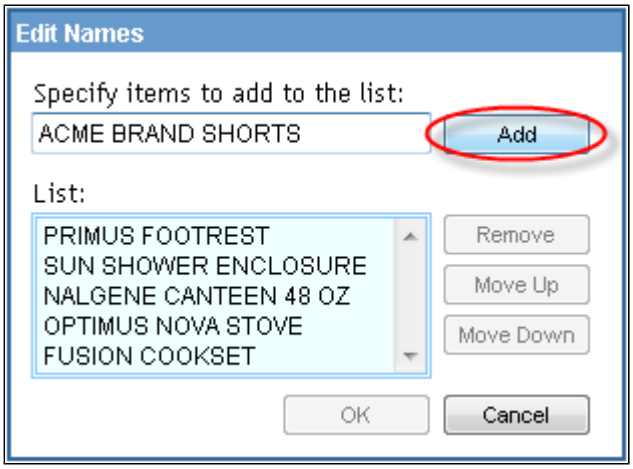
Chart Properties Window Data Entry

Step 1:

You can manually enter the names by which the chart values will be represented using the names property separated by commas.



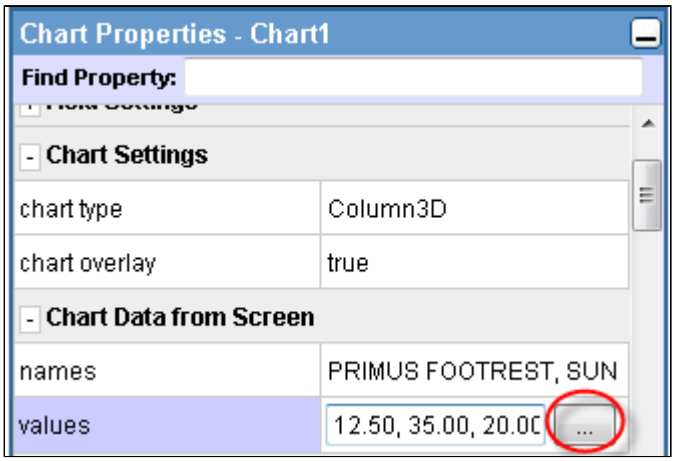
Alternatively, you can add the names through the Edit Names dialog. Type in the name in the textbox and press the Add button (highlighted in red).



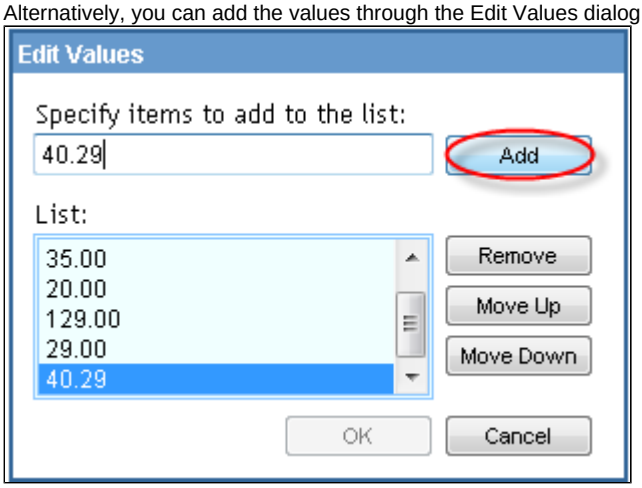
Click the OK button and proceed to add the corresponding values.

Step 2:

You can also manually enter the values corresponding to the names entered in the previous step through the values field separated by commas.

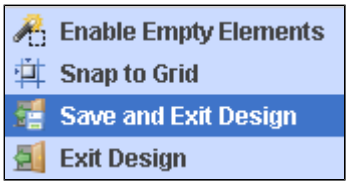


Alternatively, you can add the values through the Edit Values dialog, as shown below:



Step 3:

Right-click on the screen to get the main context menu and then select the ***Save and Exit Design*** option.



You should be able to see the chart generated from the data created in the previous steps.

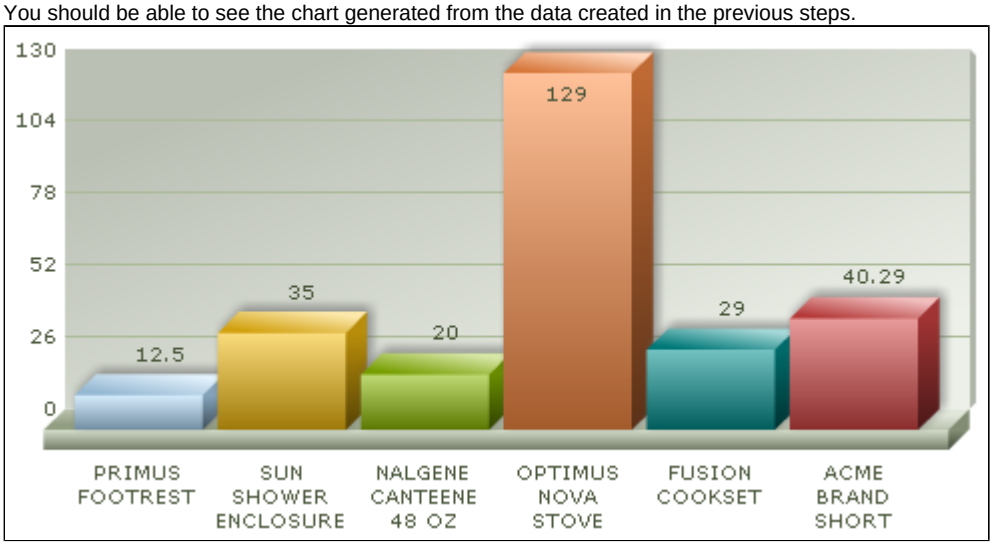
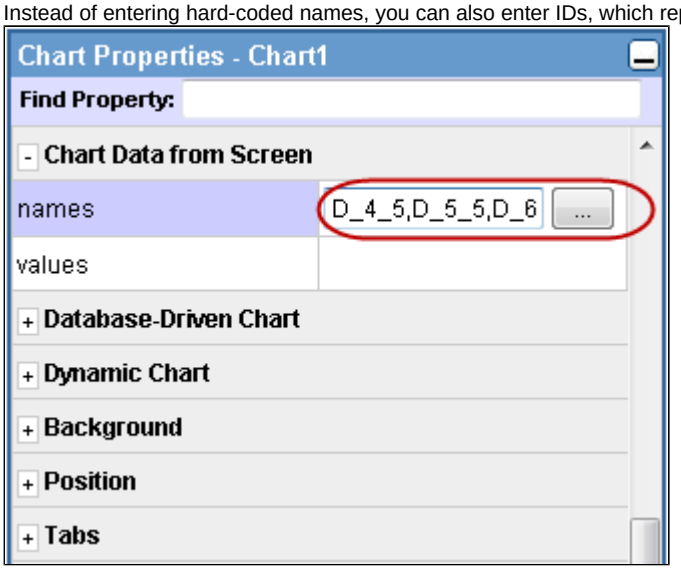


Chart Properties – Using data from screen elements

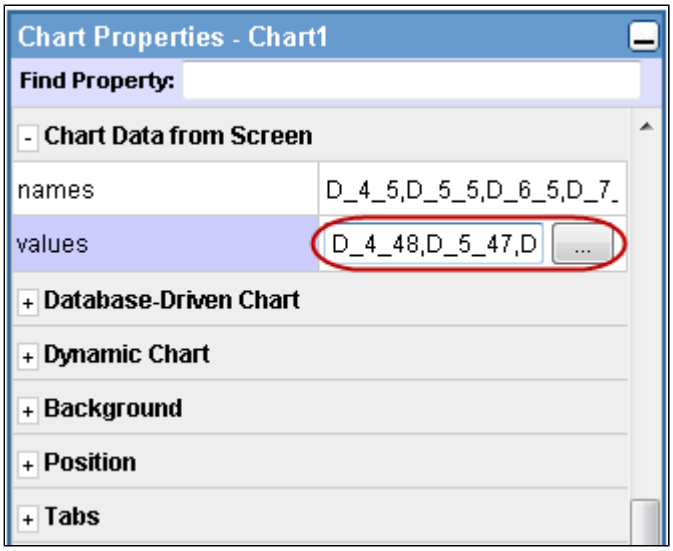
Step 1:

Instead of entering hard-coded names, you can also enter IDs, which represent elements that contain the appropriate names.



Step 2:

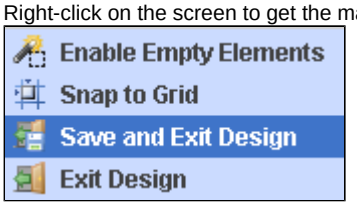
Similarly, you can enter the IDs of the elements that contain the chart values field separated by commas. The **values** property specifies a list of numerical values used to build the chart or a list of screen element IDs from which the values could be retrieved.



Note: You can combine manually entered names and values along with element IDs in the names or values fields in the Chart's Properties Dialog.

Step 3:

Right-click on the screen to get the main context menu and then select the ***Save and Exit Design*** option.



In this example, we are charting the Billing History data displayed on the Billing History screen. The ***Date*** is displayed on the X-Axis, while the ***Balance*** is displayed as the value criteria for the chart, or the Y-Axis.

You should see the chart generated from the data on the screen as shown below:

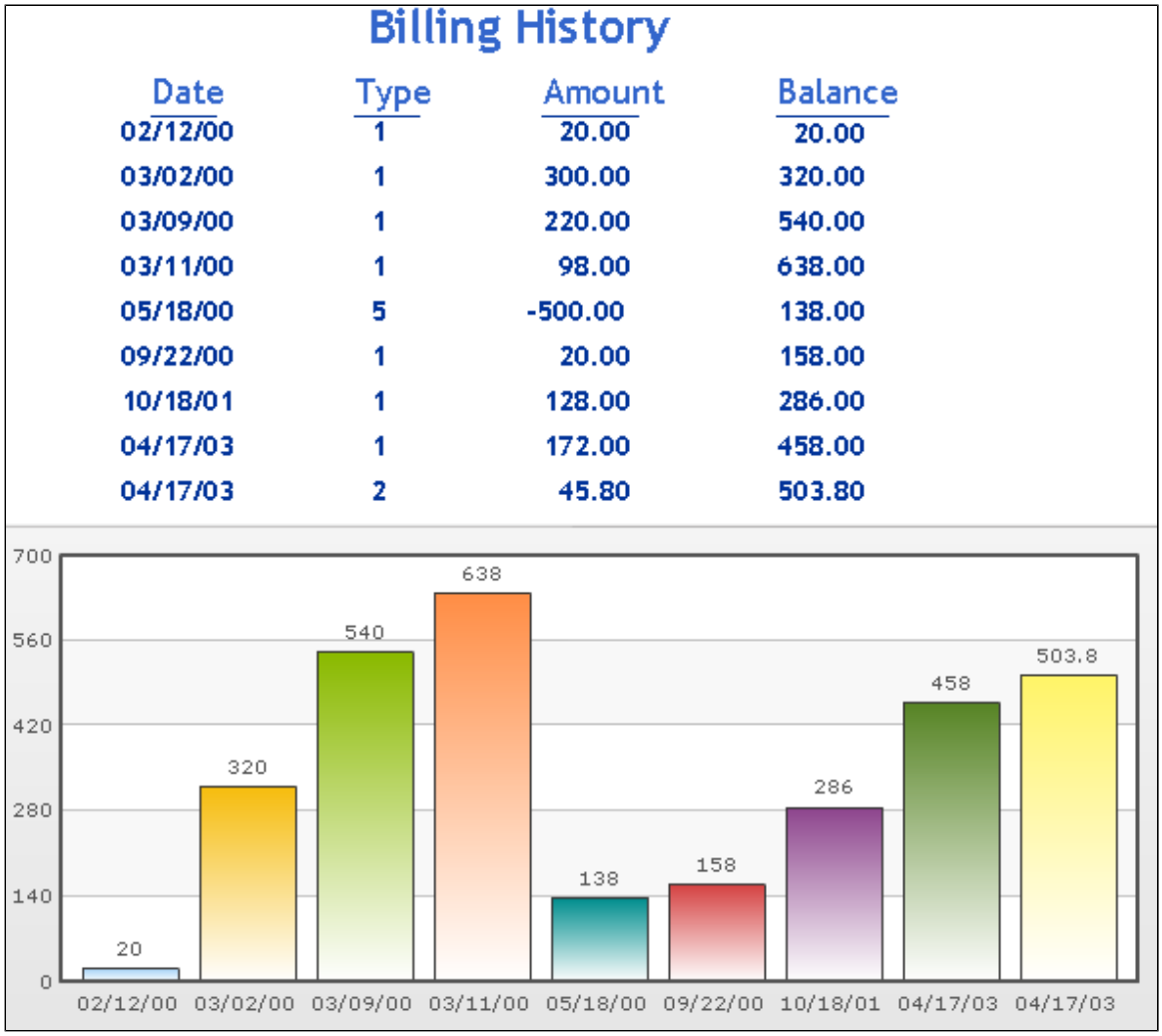


Chart Properties – Using JavaScript expressions to generate chart data

You can use JavaScript to populate your chart. This allows you to retrieve screen data as well perform various calculations on that data. In the example below, we will build a 3D Pie Chart on the Display System Status (DSPSYSSTS) screen. This chart will display the total system storage available versus the storage in use.

Step 1:

Select Pie3D in the charts list inside Widgets.

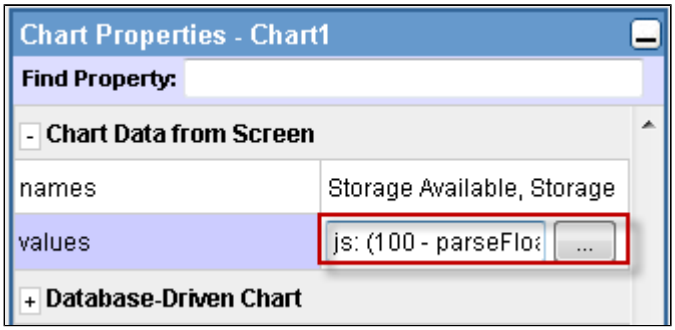
Step 2:

Type in the following expression in the values property inside of the Chart Properties:

js: (100 - parseFloat(get("**D_6_64**"))) + "," + parseFloat(get("**D_6_64**"))

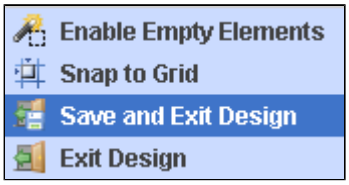
The expression must return a comma separated list of values. In this pie chart, there are only two pie pieces. Once evaluated, the above expression will produce the following result: **36.89, 63.11**.

Note: D_6_64 is the ID of the output field that contains the system storage used (63.11%). parseFloat() is a native JavaScript function that parses a string and returns a floating point number. In our case, it will parse 63.11% and return 63.11 as a value rather than string with the percent sign.

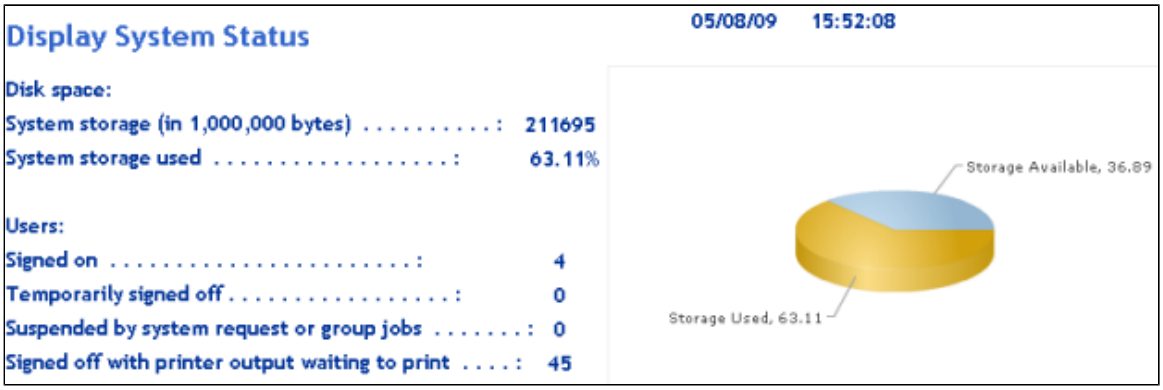


Step 3:

Right-click on the screen to get the main context menu and then select the ***Save and Exit Design.***



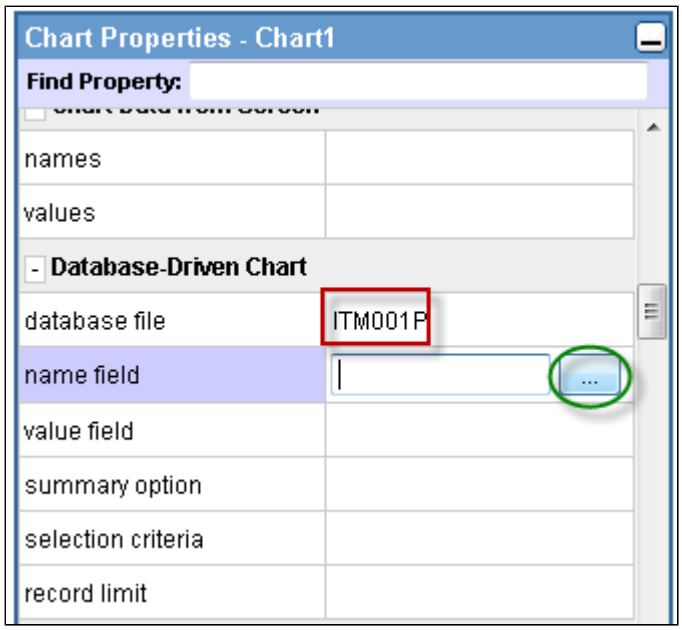
You should see something like this screen when you're done.



Database-driven Charts

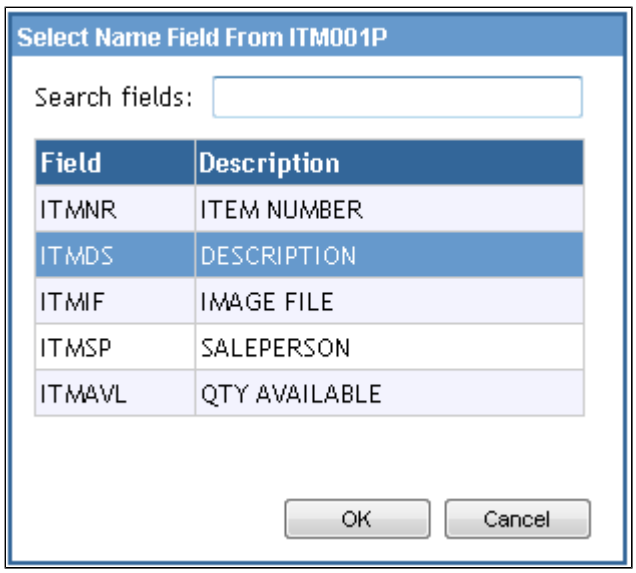
In this section, we will chart the data using a database file. In the illustration below, we are using an item master file named ITM001P for the database file property. The file name can be qualified with a library; however, if the library is omitted, the session's library list will be used to find the database file.

Next, you must specify the name field property, which will be used to determine the names by which records would be represented in the chart. You can type the field name or choose it from a dialog. Click on the button circled in green to display the database fields dialog for the database file you specified.

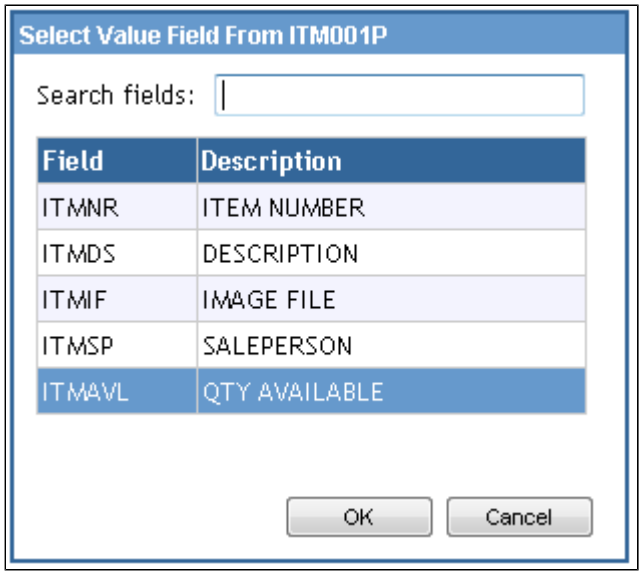


Note: Make sure that the library in which the database file exists is included within your session's library list. If the library does not exist, you will get a message that looks like this:

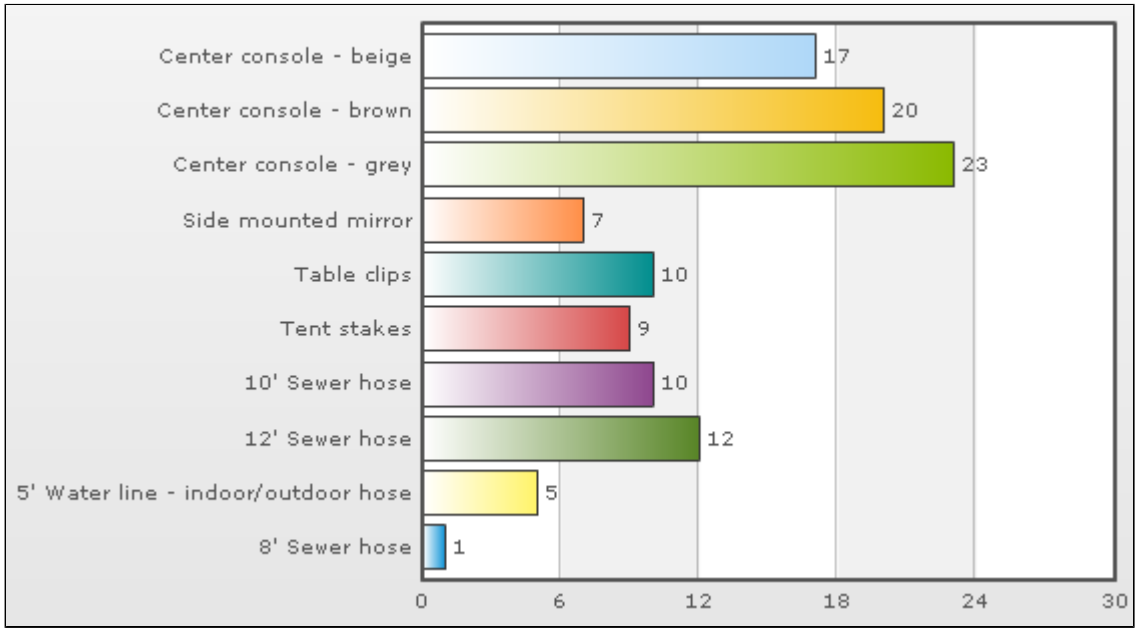
Otherwise, you will get a dialog with the available database field names to pick from. We will select the Description from the field list:



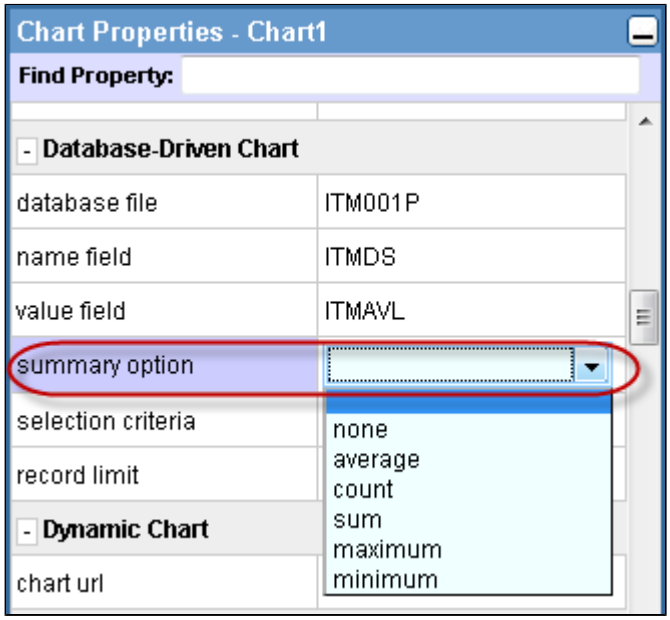
Perform the same steps to select the appropriate value field. This time, we will select the Quantity Available to be displayed as the value.



Once we save design and exit, the following chart appears:



You can also provide different summary options for your chart. The summary option determines how values are used when creating the chart.



Here is a list of the different summary options that could be used:

none: No summarization will be done. When no summary option is selected, none is assumed.

average: Finds the average dependent value corresponding to each ***name field*** value.

count: Counts the number of duplicate **name field** values within your data (ignores the **value field**).

sum: Shows the sum of the dependent values for each ***name field*** value.

maximum: Compares maximum dependent values among ***name field*** values.

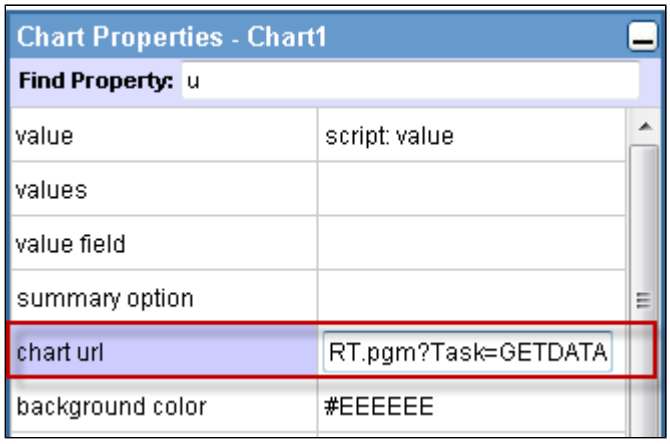
minimum: Compares minimum dependent values among ***name field*** values.

Dynamic Chart via URL

In Genie, charts can also be populated using an external program. This program or web service can be written in any web capable language and hosted on either the IBM i or on any other platform.

Since the chart expects XML as its data source, the program or web service needs to send the data to the chart in that format. This can be accomplished by calling the program that generates the XML data from within Genie through a URL.

From the properties dialog window, find the chart's URL property and type in the URL of the program that you wish to call to send the XML data as shown in the screenshot below:



The chart URL property sets the URL to a program or web service that returns the chart definition and data in XML format. The definition of the chart includes the chart type, width, height, caption, sub-caption, x-axis caption, y-axis caption, etc.

Note: If a chart's URL is used, the database file, name field, value field, summary option, and record limit properties are all ignored.

Here is a sample of the XML data that is sent to the chart through the program/web service to build and populate the chart.

```
<chart yAxisName="Quantity Available" xAxisName="Sales Person" subCaption="Sales Person/Quantity" caption="Item Master">
  <set hoverText="5' Water line - indoor/outdoor hose" showName="1" color="FF0000" value="5" name="Bob" />
  <set hoverText="Side mounted mirror" showName="1" color="1941A5" value="7" name="Jay" />
  <set hoverText="Center console - brown" showName="1" color="9966FF" value="20" name="Bob" />
  <set hoverText="Center console - grey" showName="1" color="996633" value="23" name="Bob" />
  <set hoverText="Center console - beige" showName="1" color="660066" value="17" name="Bob" />
  <set hoverText="8' Sewer hose" showName="1" color="FF66CC" value="1" name="Jay" />
  <set hoverText="10' Sewer hose" showName="1" color="990000" value="10" name="Jay" />
  <set hoverText="12' Sewer hose" showName="1" color="996633" value="12" name="Jay" />
  <set hoverText="Tent stakes" showName="1" color="669966" value="9" name="Nancy" />
  <set hoverText="Table clips" showName="1" color="FF6699" value="10" name="Nancy" />
</chart>
```

The format of the XML data can be found in the FusionCharts Data Format section of the FusionCharts documentation site.

Using Chart XML or Chart JSON directly

Chart data can also be provided using the "chart xml" or the "chart json" property. The property value can be static or generated dynamically through the use of property scripting. To use property scripting, prepend **js:** or **script:** in front of the property.

When property scripting is used with "chart xml", the JavaScript expression must return a string value containing the XML.

When property scripting is used with "chart json", the JavaScript expression can either return a string value containing JSON or a JavaScript object.

The format of the XML and/or JSON data can be found in the FusionCharts Data Format section of the FusionCharts documentation site.