TOC

Logi Symphony 24

Use Managed Dashboards, Reports, and Visualizations in Symphony 24

Symphony Managed Reports

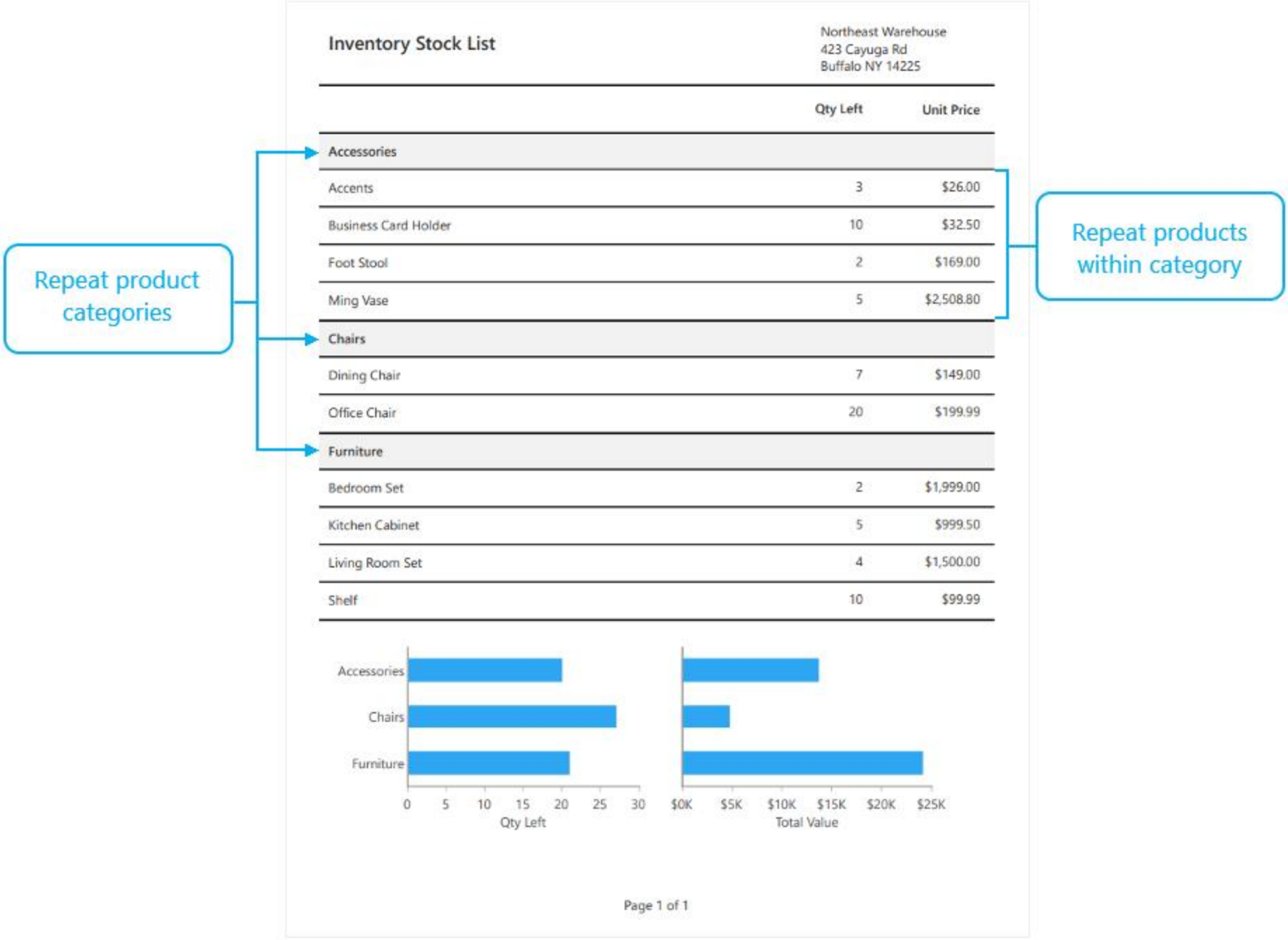
This applies to: *Managed Reports*

This article introduces the concepts behind managed reports in Symphony, which are one type of view you can create. This type of report is sometimes called multi-page or paginated reports.

There are two main differences between Symphony dashboards and reports:

- Reports are designed to be laid out into pages and you typically scroll them to see their content, while dashboards are usually designed for displaying on computer and mobile screens. Reports include page headers and footers and are ideal for saving as a PDF file and printing, although you can also view them directly within Symphony where they are interactive like dashboards.

- Reports enable you to create detailed views made up of visualizations and other content repeated like a template for each of a series of values in your data. Using template regions, you can produce a wide variety of page-formatted reports.


Repeat product categories
Repeat products within category
Inventory Stock List
Northeast Warehouse
423 Cayuga Rd
Buffalo NY 14225
Qty Left
Unit Price
Accessories
Accents 3 $26.00
Business Card Holder 10 $32.50
Foot Stool 2 $169.00
Ming Vase 5 $2,508.80
Chairs
Dining Chair 7 $149.00
Office Chair 20 $199.99
Furniture
Bedroom Set 2 $1,999.00
Kitchen Cabinet 5 $999.50
Living Room Set 4 $1,500.00
Shelf 10 $99.99
Accessories
Chairs
Furniture
0 5 10 15 20 25 30
Qty Left
$0K $5K $10K $15K $20K $25K
Total Value
Page 1 of 1

Note: If the type of report you want to create is not meant to be paginated, and is laid out by hand rather than repeating vertically or can be created with a table visualization, you can use a dashboard instead.

Related video: Scorecards and Reports

For more information about managed reports, see the following articles:

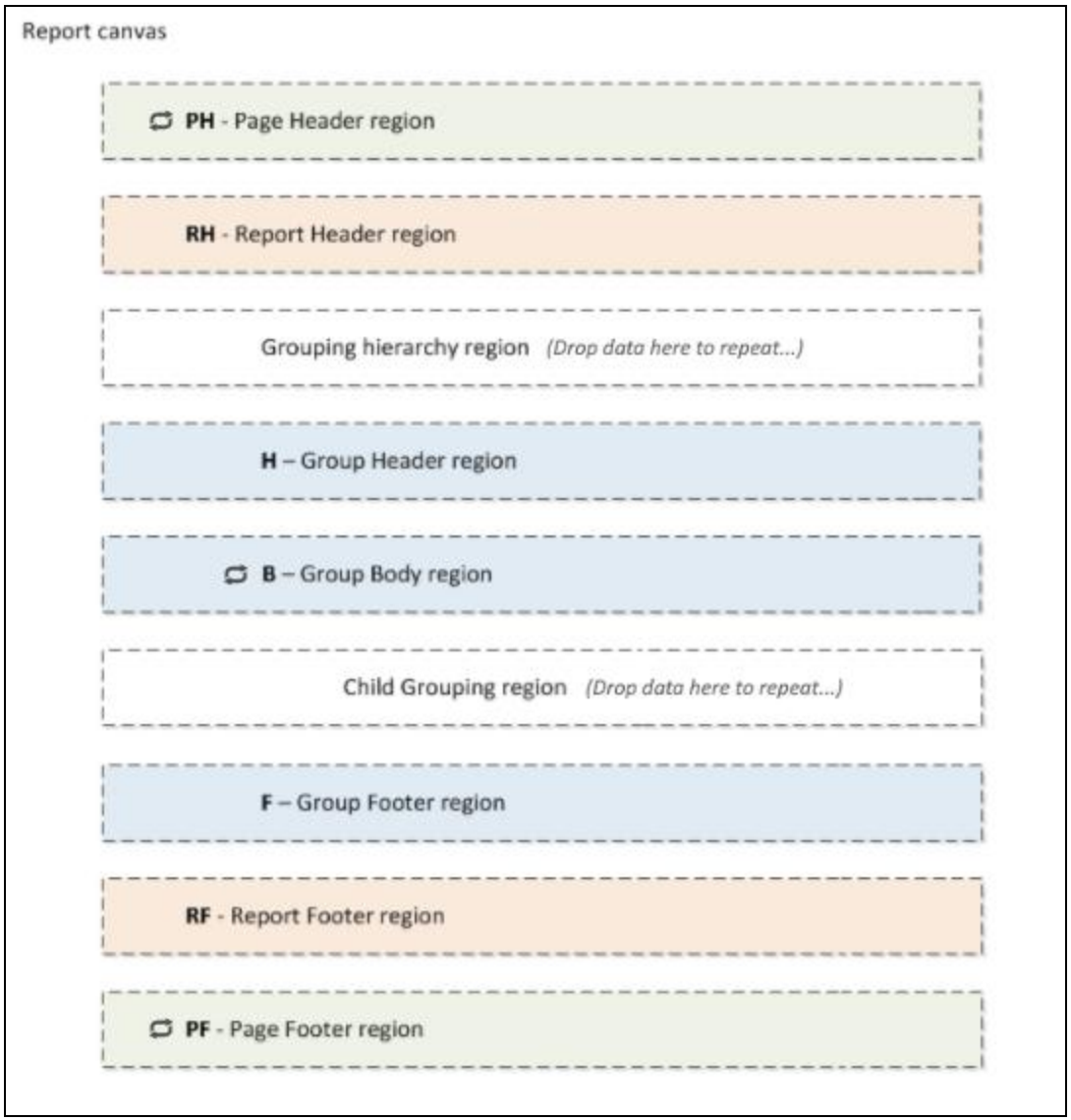
- Managed Report Layout Regions
  - Report Headers and Footers
  - Page Headers and Footers
  - Grouping Hierarchy Region
  - Group Headers and Footers
  - Group Body
  - Child Groups
  - Sibling Groups
  - Add a Table of Contents
- Report Layout Example

Managed Report Layout Regions

This applies to: *Managed Reports*

You can choose any Symphony visualization or any available component and arrange it anywhere on the canvas. In a report, the canvas contains several specialized regions, many of which repeat the content that you lay out within it like a template. For example, the inventory report shown above includes a row for each product category (e.g., *Accessories*), and you are able to design the contents of each row to include whatever you like.

The diagram below illustrates the initial regions available in a report that repeats by one set of data. You can drop items onto these regions when editing, then switch to **View** in the toolbar to see the actual report consisting of repeated elements.



You can add more of any template region to better organize your reports. Simply select one and choose **Add Template Area Before** or **Add Template Area After** in the toolbar. Unused template areas will not be visible in the report unless you uncheck the **Hidden If Empty** property.

For more information about the regions of managed reports, see the following articles:

Report Headers and Footers

This applies to: *Managed Reports*

The **Report Header** region lets you place elements that appear just once per report. For example, you can show a title for the overall report here. When you create a new report from the main menu, Symphony automatically adds a Label component with the text Title and places it in the Report Header region, which you can edit by double-clicking or by using the toolbar or Properties window. You can resize this region and add any other content you like.

Similarly, in the **Report Footer** region, you can add content to close off your report.

For more on Scorecard and Report design, see:

- Symphony Scorecards
- Symphony Managed Reports
- Design a Scorecard
- Report Options
- Design Tips for Reports and Scorecards
- Design Tips

Page Headers and Footers

This applies to: *Managed Reports*

Since reports are laid out onto pages, the **Page Header** and **Page Footer** regions let you add content to be repeated once per page. Symphony automatically adds a Label component with the text *[PageNumber]* in the Page Header region and another Label component with the text Page *[PageNumber]* of *[TotalPages]* in the Page Footer region. These are examples of placeholder labels that you can add yourself from the Labels window.

You can add other content to this header and footer such as a smaller title, a logo, or a separator line.

For more on Scorecard and Report design, see:

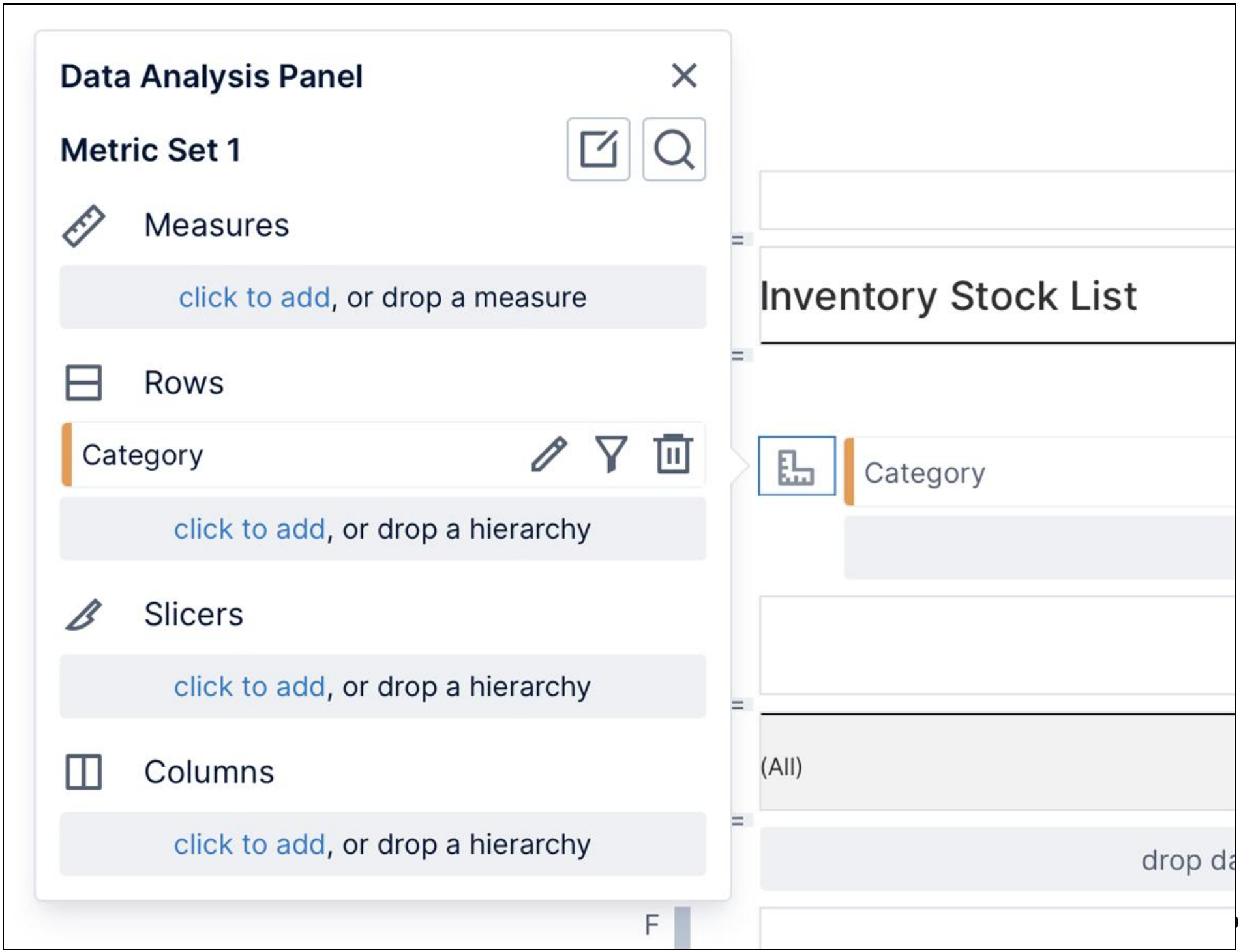
- Symphony Scorecards
- Symphony Managed Reports
- Design a Scorecard
- Report Options
- Design Tips for Reports and Scorecards
- Design Tips

Grouping Hierarchy Region

This applies to: *Managed Reports*

The grouping hierarchy region *(drop data here to repeat...)* or drop zone, is the key to repeating the **Group Body** region template. This region is equivalent to the **Rows** of the Data Analysis Panel for a regular metric set, where you also drag data from the **Explore** window to display a row for each value. For example, *Product Category* values were dragged here to achieve the repetition of the gray rows for each category in the inventory report shown earlier. You can drag a hierarchy, hierarchy level, column of data, or a complete metric set here to determine which rows to display.

This region makes up the Rows of the group metric set, which has its own Data Analysis Panel accessible from the cube icon displayed to its left. This Data Analysis Panel allows you to optionally further customize how the Group Body rows are repeated. For example, you can filter the hierarchy values to remove *Accessories*, or add a measure such as *Qty Left* and sort the categories by its values.


Data Analysis Panel
Metric Set 1
Measures
click to add, or drop a measure
Rows
Category
click to add, or drop a hierarchy
Slicers
click to add, or drop a hierarchy
Columns
click to add, or drop a hierarchy
Inventory Stock List
Category
(All)
drop da
F

Note: Reports are made up only of rows that repeat content vertically. Adding data under Columns in the Data Analysis Panel will produce an error when viewing.

Like Rows in a regular metric set, adding multiple grouping hierarchies to the same region will repeat the group body for each combination of hierarchy values occurring in your data.

For more on Scorecard and Report design, see:

- Symphony Scorecards
- Symphony Managed Reports
- Design a Scorecard
- Report Options
- Design Tips for Reports and Scorecards
- Design Tips

Group Headers and Footers

This applies to: *Managed Reports*

The **Group Header** and **Group Footer** regions let you place content that appears once per group, either above or below its rows. For example, for the inventory report example shown in Symphony Managed Reports, *Header* or *Label* components are added to the Group Header region in order to display the *Qty Left* and *Unit Price* column headings. You can also drag data or add visualizations here to display aggregated data in the header or footer summarizing the group items above or below.

Group headers and footers are not repeated unless this group is a 'child' or inner group placed inside another repeating group.

For more on Scorecard and Report design, see:

- Symphony Scorecards
- Symphony Managed Reports
- Design a Scorecard
- Report Options
- Design Tips for Reports and Scorecards
- Design Tips

Group Body

This applies to: *Managed Reports*

When viewing a report, the **Group Body** region is repeated once for each grouping hierarchy value. For example, if *Product Category* values are added to the grouping hierarchy region, and there are 20 such categories in your data, the Group Body region will be repeated 20 times and will display data specific to each category.

Within this region, *data labels* are added automatically to display values as text like a column in a table. You can re-visualize to or add any visualization and display any additional data you like detailing or describing each grouping hierarchy value, such as a line chart displaying the historical stock quantities for each product category over time, or a map visualizing its geographic distribution.

For more on Scorecard and Report design, see:

- Symphony Scorecards
- Symphony Managed Reports
- Design a Scorecard
- Report Options
- Design Tips for Reports and Scorecards
- Design Tips

Child Groups

This applies to: *Managed Reports*

Child grouping regions let you add an inner or nested level of repetition in order to create a report that breaks down details further. For example, the inventory report example shown in Symphony Managed Reports displayed rows for individual products after the row for each product category, which was done by adding *Product* as a child group within the *Product Category* group.

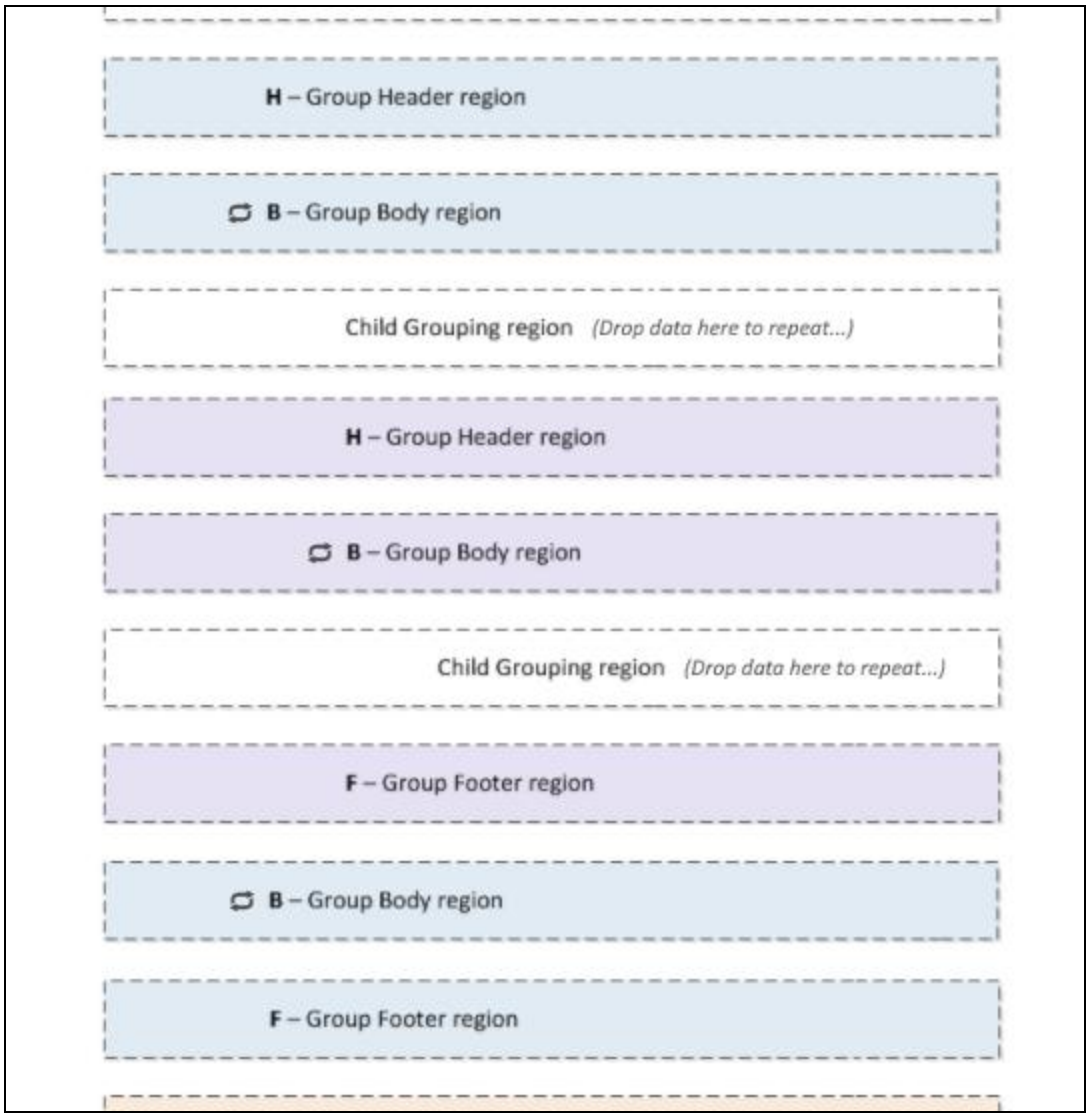
When you drag a hierarchy or level to this region, four new regions are automatically inserted into the report, forming a child group consisting of:

- a child group header region - (optional)
- a child group body region - this is the part that repeats
- a child grouping region - lets you insert another nested/child group
- a child group footer region - (optional)

In fact, you can keep on inserting child groups to create deeper and deeper levels of repetition. This allows you to design complex reports that break down details on multiple levels.

The child group has its own **Group Body** that you can design separately from the outer or parent grouping. In our inventory report example, the product category rows have a light gray background and bold fonts, while the product rows have normal fonts and display measure values toward the right.

Because they are displayed inside another group that is already repeating, child groups can appear multiple times - for example, there is a separate list of products inside each product category in our inventory report. You can choose to add content to the child **Group Header** or child **Group Footer**, and these will appear above and below each set of child group body rows.



When there is a child group, the parent group has a second group body available for displaying content after the child group. The parent group's body regions and the child group's header and footer are similar, but child group regions do not appear if there are no child grouping hierarchy values. For example, if the parent group is displaying product categories and there is an empty category that contains no products, the parent group body will be displayed for this category but the child group header, body, and footer will not display.

Note: Indented rectangles to the left of each region are used to distinguish the child regions from the parent regions. You can also click one and the other regions from the same group will be highlighted to make them easier to identify.

For more on Scorecard and Report design, see:

- Symphony Scorecards
- Symphony Managed Reports
- Design a Scorecard
- Report Options
- Design Tips for Reports and Scorecards
- Design Tips

Sibling Groups

This applies to: *Managed Reports*

In addition to displaying groups inside other groups, you also have the option of adding **sibling groups**. These are independent from the neighboring groups and may contain data unrelated to the rest. Sibling groups are positioned at the top level of the report like the initial group, and their repetitions are generated independently, either before or after all of the data from neighboring sibling groups.

Sibling groups can be added using the **+** buttons to the right of the report canvas.

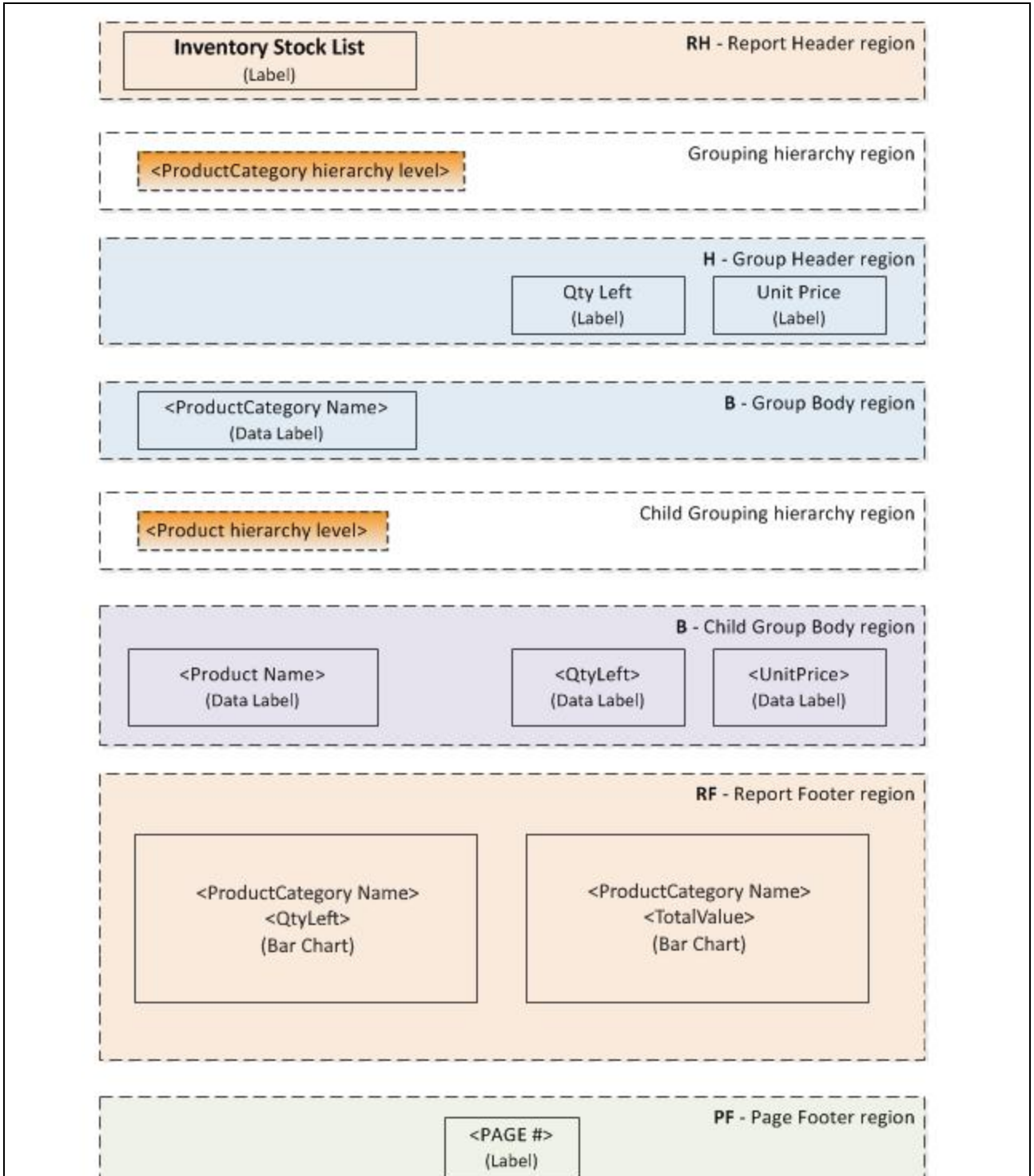
For more on Scorecard and Report design, see:

- Symphony Scorecards
- Symphony Managed Reports
- Design a Scorecard
- Report Options
- Design Tips for Reports and Scorecards
- Design Tips

Report Layout Example

This applies to: *Managed Reports*

The following diagram illustrates a simplified design for the inventory report example shown in Symphony Managed Reports (unused regions are omitted and are hidden when viewing). This report repeats visualizations on two levels: an initial group that displays each *Product Category* and a child group displaying each category's *Product*.


Inventory Stock List
(Label)
RH - Report Header region
<ProductCategory hierarchy level>
Grouping hierarchy region
H - Group Header region
Qty Left
(Label)
Unit Price
(Label)
<ProductCategory Name>
(Data Label)
B - Group Body region
<Product hierarchy level>
Child Grouping hierarchy region
B - Child Group Body region
<Product Name>
(Data Label)
<QtyLeft>
(Data Label)
<UnitPrice>
(Data Label)
RF - Report Footer region
<ProductCategory Name>
<QtyLeft>
(Bar Chart)
<ProductCategory Name>
<TotalValue>
(Bar Chart)
PF - Page Footer region
<PAGE #>
(Label)

For more on Scorecard and Report design, see:

- Symphony Scorecards
- Symphony Managed Reports
- Design a Scorecard
- Report Options
- Design Tips for Reports and Scorecards
- Design Tips

Using Interactions

This applies to: *Managed Dashboards, Managed Reports*

In Symphony, you can set up actions or interactions on any visualization, filter, or component, or view such as a dashboard.

You can set up common interactions from the toolbar or context menu, and any number of other action combinations using the Properties window.

Related video: Interactions

Actions

You can set up actions to run when some event occurs for a dashboard or other view, or one of its elements, such as when a viewer clicks (or taps) an element. These are the actions available:

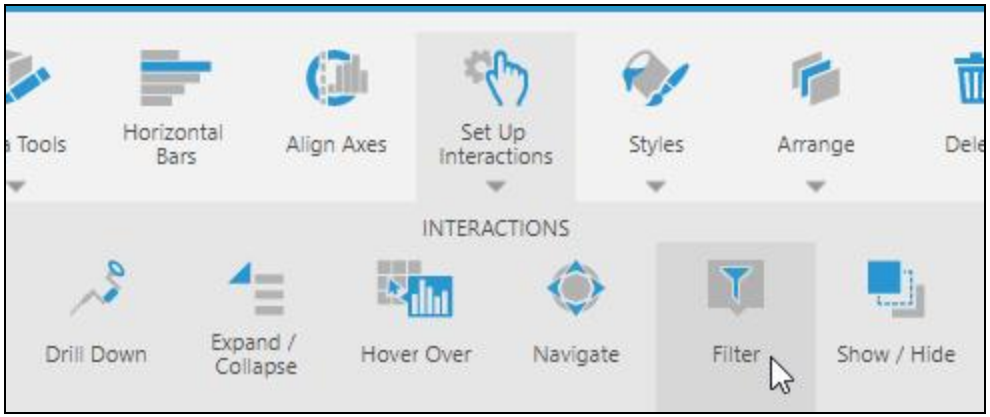
- Navigate – navigates to another dashboard or other view, or to a webpage
- Hover Over or Pop-up – displays a pop-up of another view such as a dashboard
- Drill Down – goes one level down in the hierarchy and filters to the selected data
- Filter – filters to the selected data
- Expand/Collapse – expands a hierarchy member to view its values from one level down, or collapses them up
- Show / Hide or Change Layer – shows or hides content
- Toggle Template Cell – expands or collapses a cell in a dashboard's template grid
- Data Input – appends data to a Data Input transform in a data cube (not available on data visualizations)
- Script – executes the provided JavaScript code (only available from the Properties window)

Any element can run actions for common events such as *Click*, while some elements have more events available such as *Selection Changed* on a Drop Down List or *No Data* on a data visualization. Views have events such as *Ready*, which occurs when initial loading has finished. Certain elements or events may not support all types of actions.

Creating an Interaction

From the Toolbar

To quickly create an interaction for the element that's selected, choose **Set Up Interactions** in the toolbar and choose an option. This will automatically set up an action for the most common event and open its dialog for you to configure or confirm its settings.



Choose the same option from the toolbar again to edit or delete the action, or you can find this action in the corresponding actions list in the Properties window as described in the next section.

For **Hover Over**, a *Pop-up* action is always added to the **Hover** actions of the element for when viewers hover over it with their mouse. Other interactions are set up depending on the element:

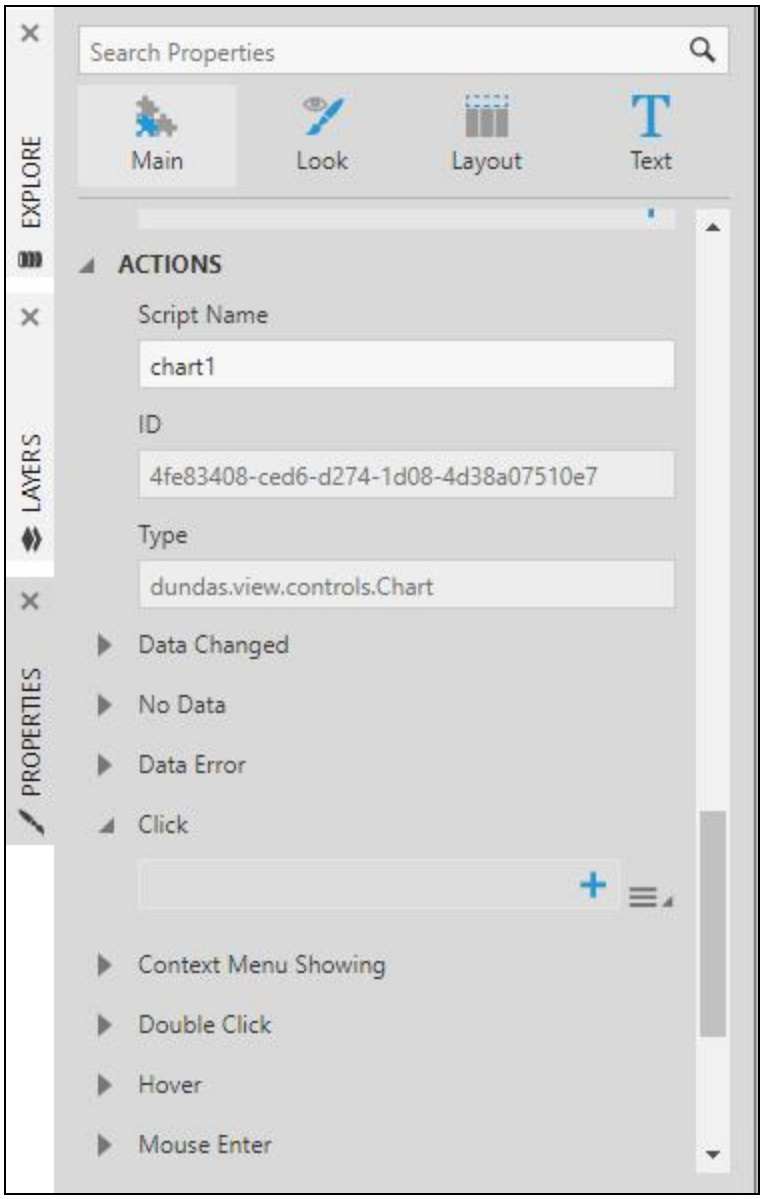
- Data visualizations – **Click**
- Filters – **Value Changed**
- Radio Button List – **Selection Changed**
- Drop Down List – **Selection Changed**
- Timer – **Timer Interval Tick**

- Slider component – **Changed**
- Other components – **Click**

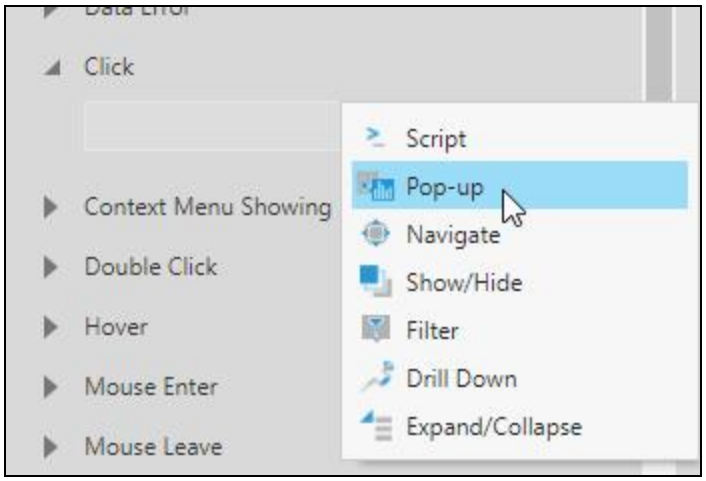
In the Properties Window

You can use the **Properties** window to add an action for whichever event you prefer, or to add or reorder multiple actions. As an example, you can make a pop-up appear when a data point is clicked, rather than when the mouse hovers over it.

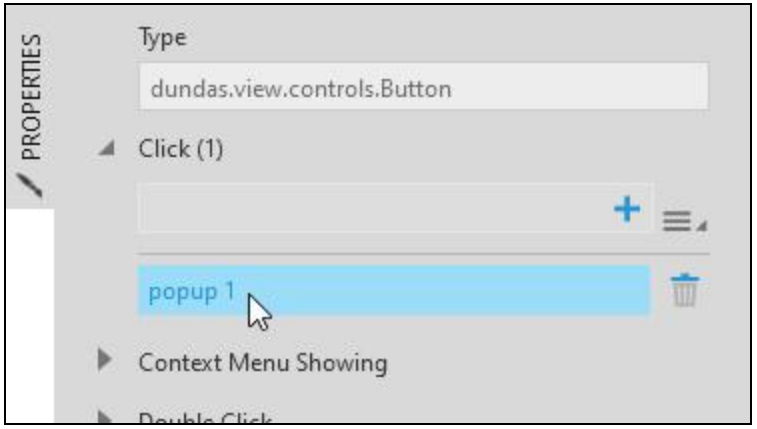
In the **Main** tab of the Properties window, find and expand the actions you're interested in under the *Actions* category. For example, expand the **Click** actions (which also run when a viewer taps on a touchscreen).



Click the menu icon on the right, and choose a type of action to add. We will select **Pop-up**.



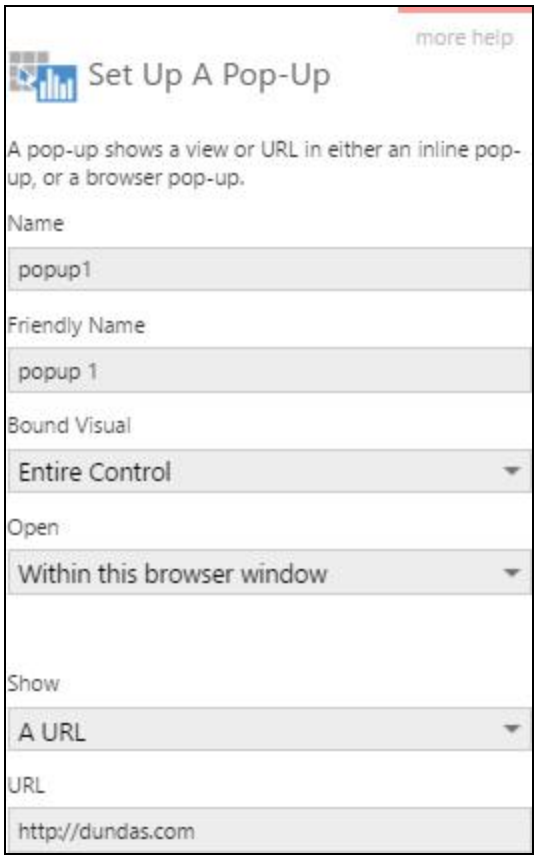
Click the action in the list to configure it.



If there are multiple actions added to the same list, arrow buttons will appear next to each action that you can click to reorder them. Actions will run in the order that is listed.

Configuring an Interaction

When the configuration dialog appears, in many cases the interaction will set itself up automatically and you can immediately click the submit button at the bottom to close it.

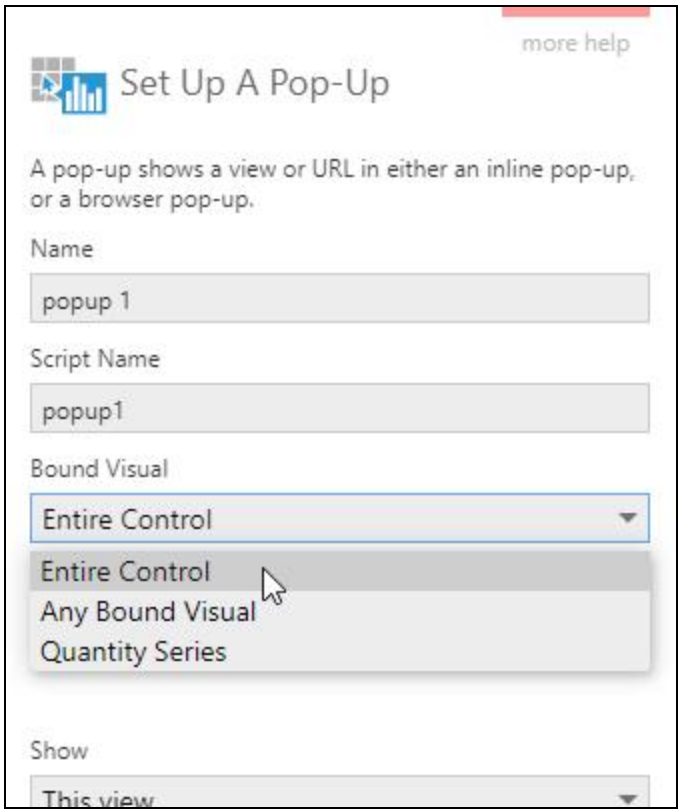


Some actions have additional options to configure how the interaction is triggered. Common configuration options are described here, while others are included in walkthroughs linked to for each type of action in the list above.

All actions have a **Name** (sometimes called 'friendly name') you can optionally customize to help identify actions in the Properties or Layers windows.

Bound Visual

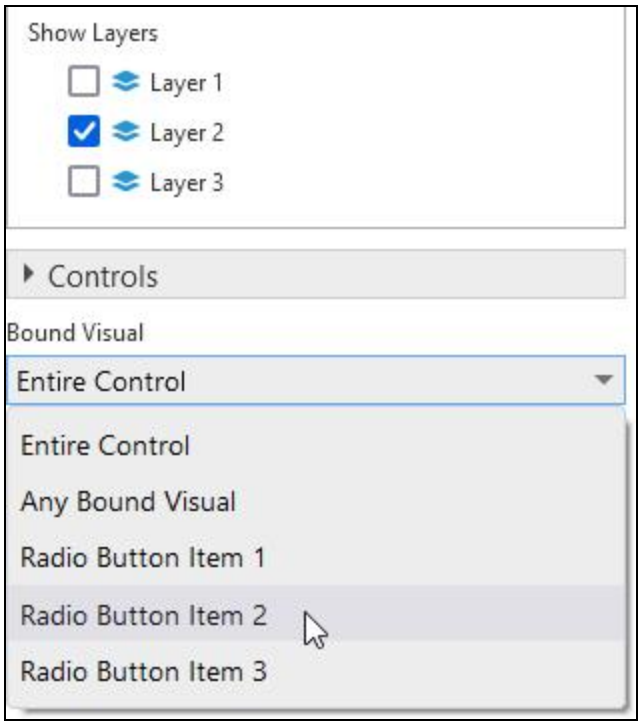
The **Bound Visual** option (if available) causes the action to occur only on the indicated part of that control or visualization. You can add multiple actions to the same event via the Properties window if you want different actions for different Bound Visuals.



Many visualizations allow you to limit an interaction to a specific area, such as a column in a table or one series of data points in a chart.

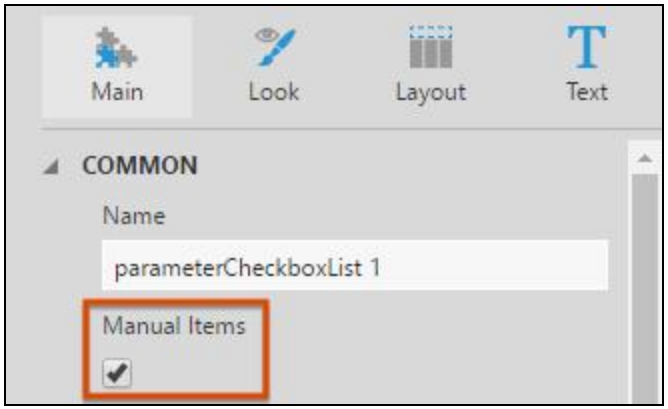
Checkboxes can be in one of two states: *Checked*, and *Unchecked*, and you can bind an interaction to either state. For example, you can show a layer of content when a checkbox is *Checked* and hide it when Unchecked.

Drop Down Lists and Radio Buttons allow you to set up their items or buttons yourself in the Properties window, and you can then use the Bound Visual option in interactions to bind separate interactions to each one. For example, a Show/Hide interaction can show a particular layer of content on a dashboard with other layers hidden when the second of three radio buttons is selected.



To use a filter in this way instead of as a regular filter, check its **Manual Items** property. For example, you can add items to a Checkbox List filter and set up an action for each item.

Important: This means you will be using manually-added items instead of any connected parameter.

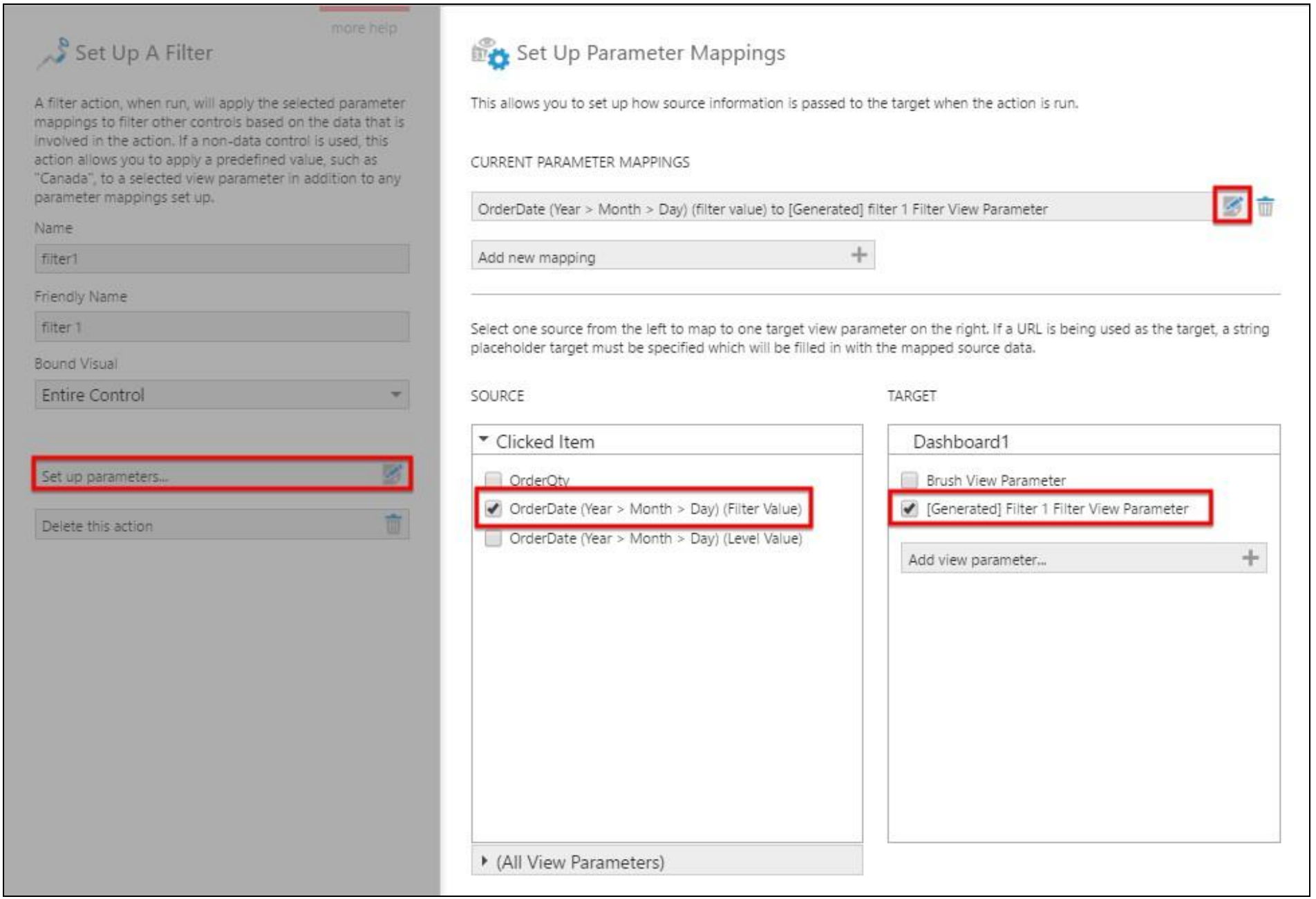


Set up Parameters

Many actions involve choosing what data should be passed to a parameter (e.g., what data to filter by).

When setting up a filter or drill down interaction, Symphony will automatically make a choice for you based on compatible data on the current dashboard or view, but you can confirm or change this setting. Navigate and hover over/pop-up interactions require setting up the parameters if you want the other dashboard or view to be filtered.

To set these up, click **Set up parameters** in the interaction dialog.


more help
Set Up A Filter
A filter action, when run, will apply the selected parameter mappings to filter other controls based on the data that is involved in the action. If a non-data control is used, this action allows you to apply a predefined value, such as "Canada", to a selected view parameter in addition to any parameter mappings set up.
Name
filter1
Friendly Name
filter 1
Bound Visual
Entire Control
Set up parameters...
Delete this action
Set Up Parameter Mappings
This allows you to set up how source information is passed to the target when the action is run.
CURRENT PARAMETER MAPPINGS
OrderDate (Year > Month > Day) (filter value) to [Generated] filter 1 Filter View Parameter
Add new mapping
Select one source from the left to map to one target view parameter on the right. If a URL is being used as the target, a string placeholder target must be specified which will be filled in with the mapped source data.
SOURCE
TARGET
Clicked Item
OrderQty
OrderDate (Year > Month > Day) (Filter Value)
OrderDate (Year > Month > Day) (Level Value)
Dashboard1
Brush View Parameter
[Generated] Filter 1 Filter View Parameter
Add view parameter...
(All View Parameters)

Under **Source**, choose what value should be passed to a parameter. To pass more than one clicked value at the same time, click the **Add new mapping** button to add multiple mappings: one for each value.

Under **Target**, choose a parameter to pass that value to. Symphony will automatically generate a parameter for a filter or drill down interaction connected to compatible hierarchies on your dashboard or view, or you can choose your own parameter. You can create parameters or change what the parameter is connected to in the Parameters window.

For a walkthrough on setting up parameters, see the linked articles for each action.

For more information, see:

- Video: Interactions
- Design Overview
- Add a Change Layer Action on Ready Event
- Handling No Data and Data Errors

Create a Help Overlay Using Layers

This applies to: *Managed Dashboards, Managed Reports*

Use interactions to allow viewers to show & hide content in dashboards and other views, which can include components, visualizations, and filters.

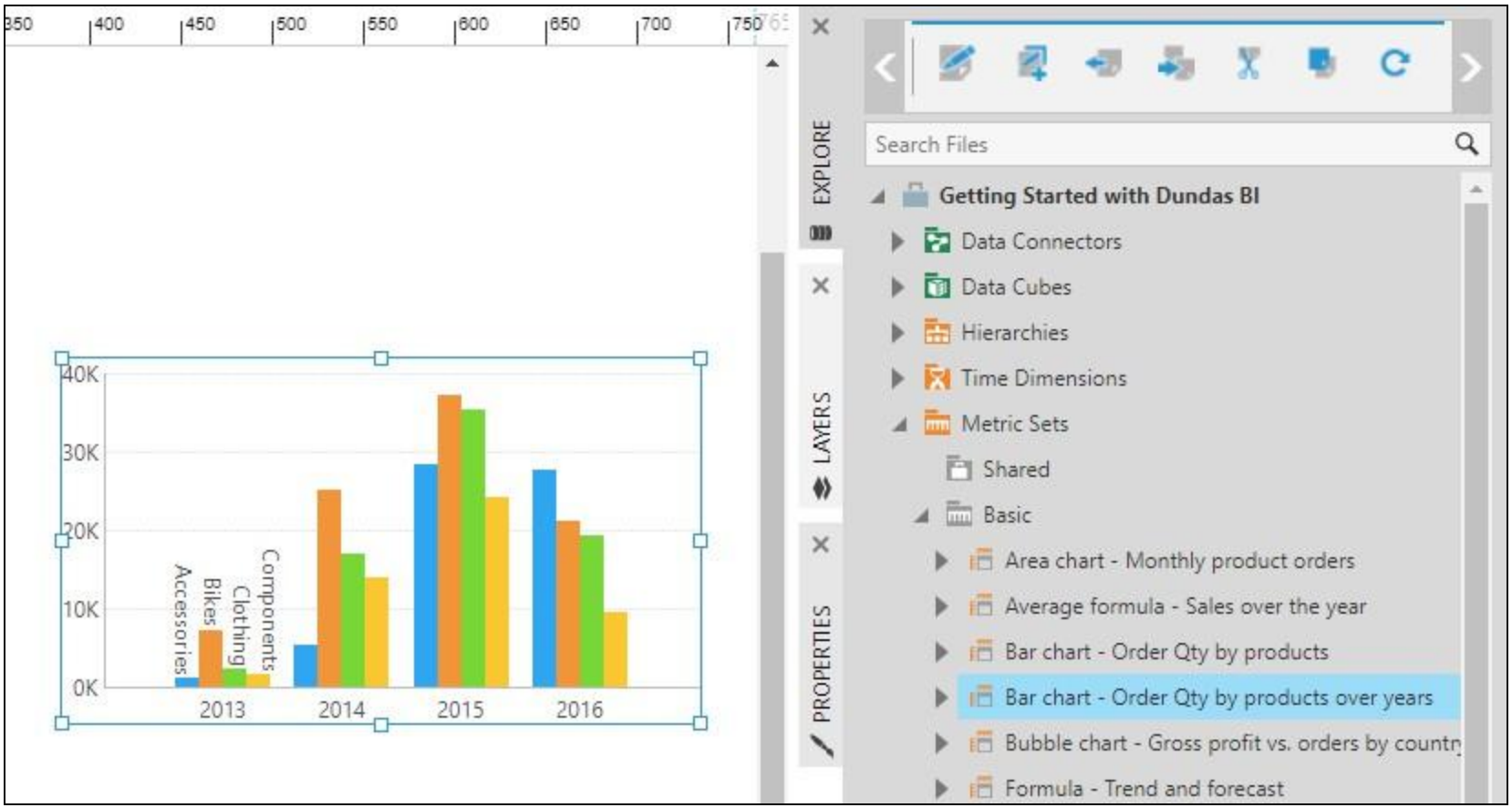
You could use this interaction to make elements appear like a pop-up or window. To show and hide multiple elements at once, you can group them into layers, then hide and show the entire layer. A basic interactive help overlay is added to a dashboard as an example below, which displays information about how to use a dashboard when a user hovers over or clicks on a help button.

Note: Organizing elements into layers is optional. You can also use a single interaction to toggle back and forth between showing and hiding content.

Related video: Interactions

Create a New Dashboard

Create a basic dashboard by dragging an existing metric set from the **Explore** window to the canvas.



We have a bar chart in our example.

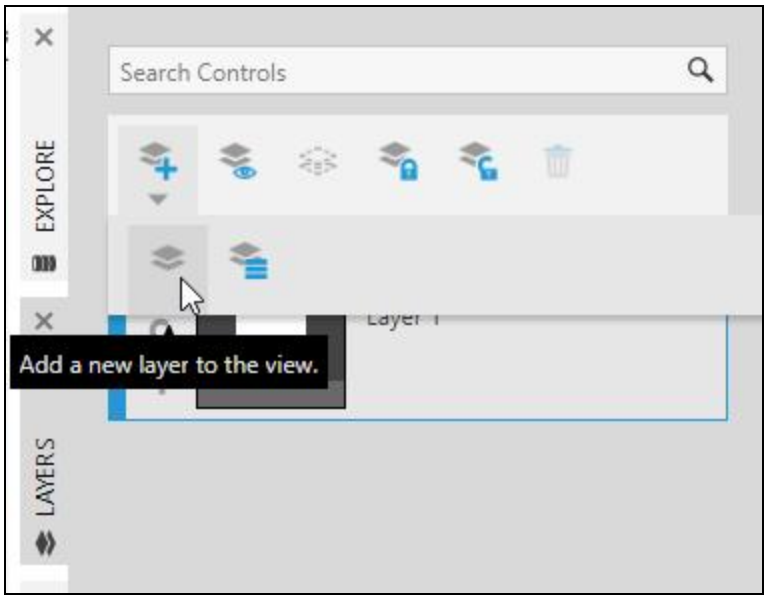
Create the Help Overlay Layer

This example shows and hides multiple elements at once, so we will add them to a layer so we can easily show and hide all of them when viewing as well as when editing.

Add a New Layer

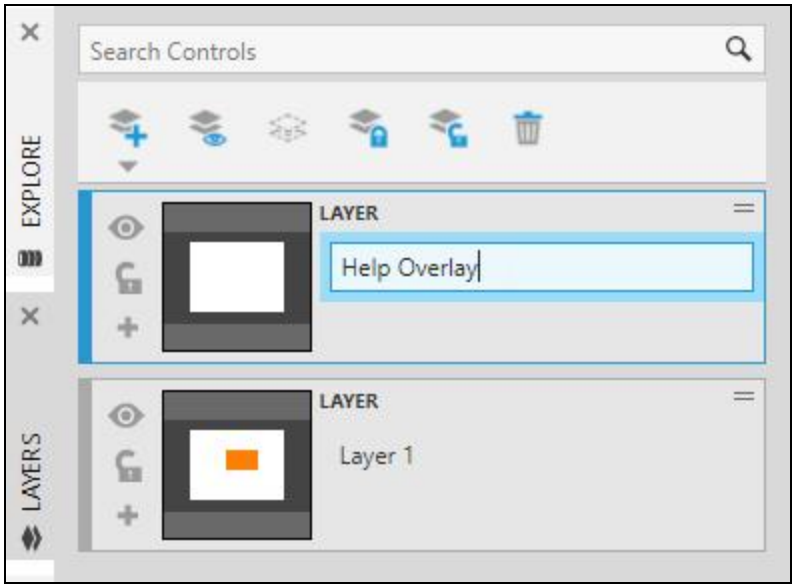
Open the **Layers** window and then click the toolbar button with the *plus* symbol to add a new layer.

If a submenu appears, choose the first icon to add a plain layer.



Layer 2 is added to the Layers window and becomes the current layer. This will contain help-related components such as a frame and a label.

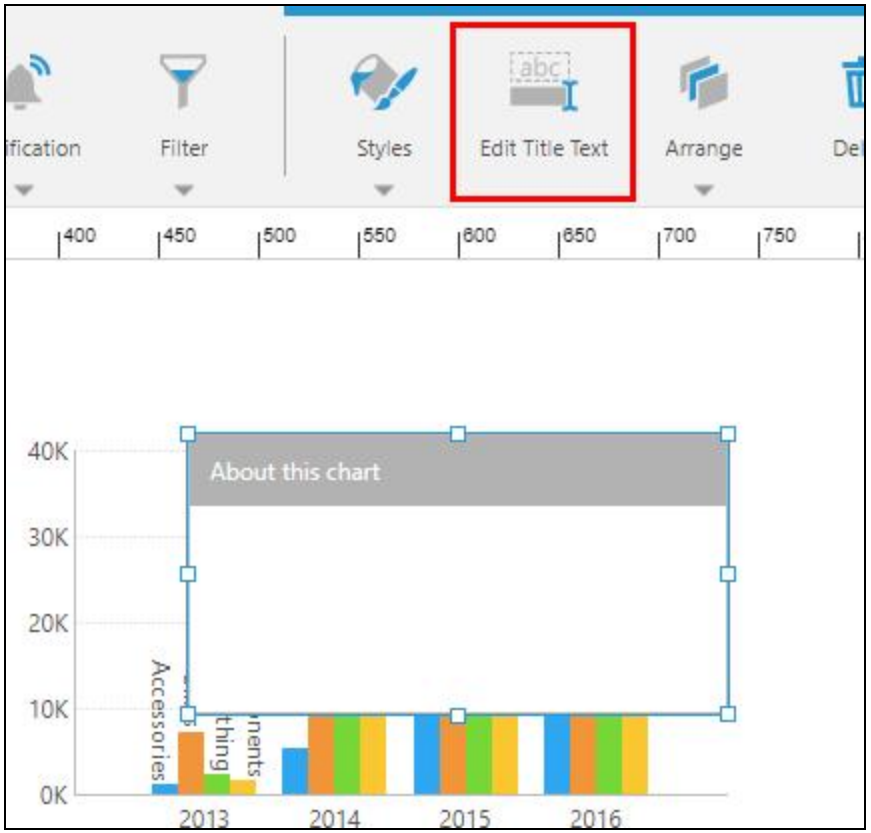
Next, rename the layer by double-clicking its name or using the right-click option.



Add a Frame Component to the New Layer

Click **Components** in the toolbar, scroll the submenu to the right, and choose **Frame**. Position the frame component over the top-right corner of the bar chart.

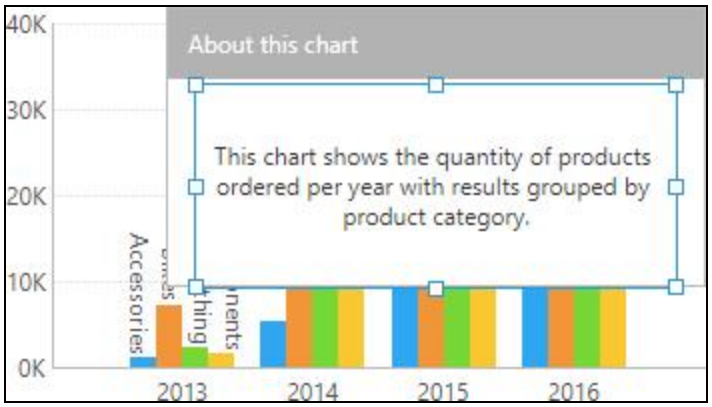
In the toolbar, choose **Edit Title Text** and type in a new title for the frame such as *About this chart*. Press Enter or click away when finished typing.



Add a Label on Top of the Frame

Go to the toolbar, click **Components**, and then click **Label**.

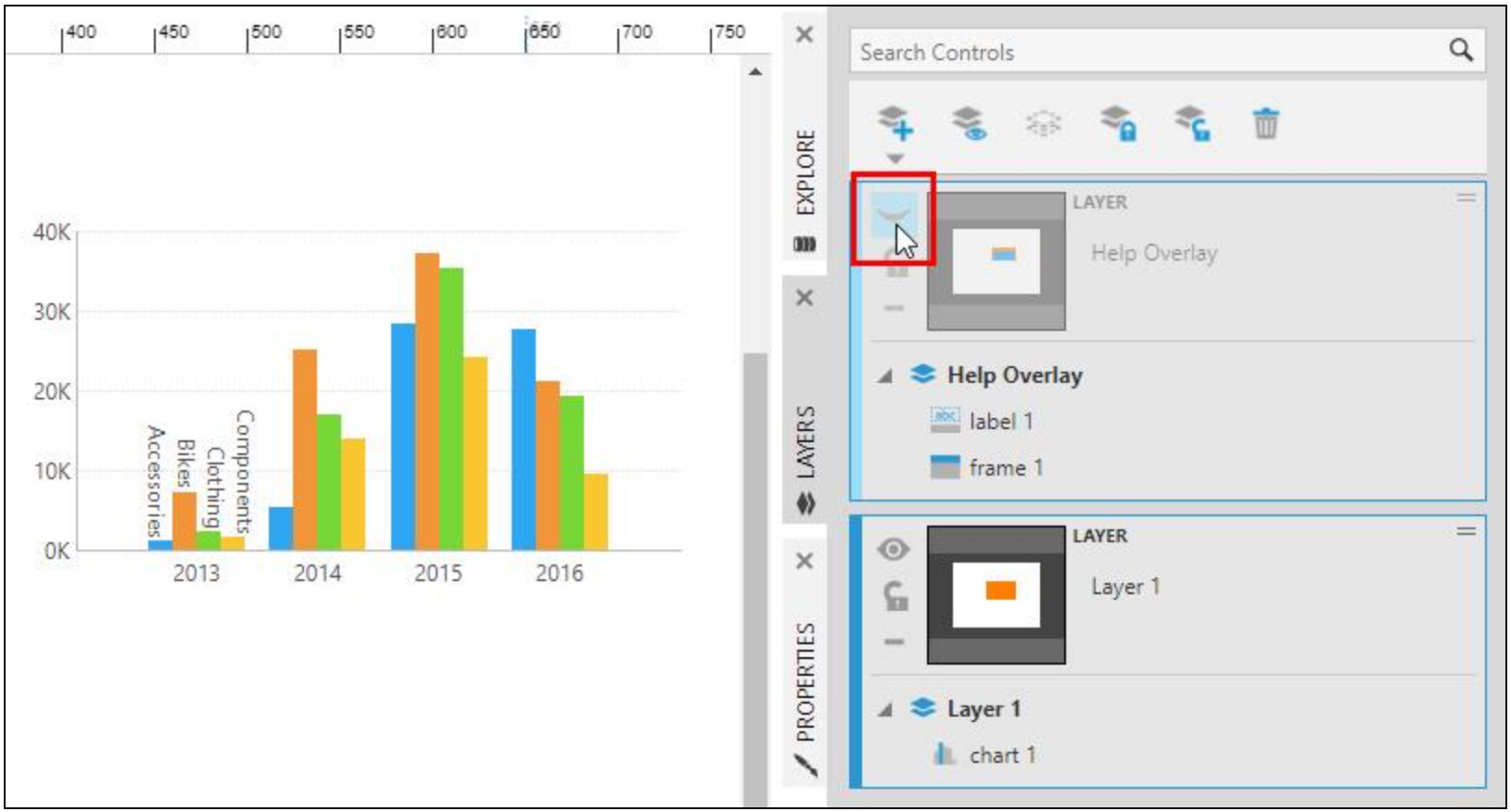
Position the label over the frame component, then double-click it or choose the toolbar option to edit its text, and type in a description.



Hide the Help Overlay Layer

Initially, the help overlay layer should be hidden when viewing the dashboard.

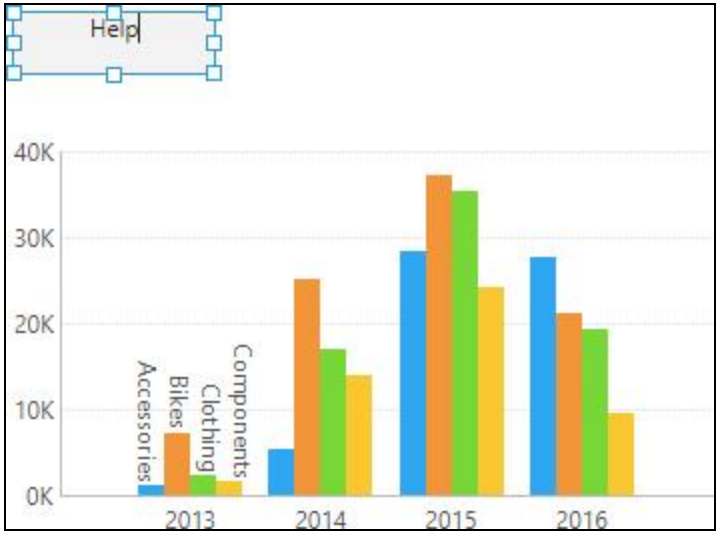
In the **Layers** window, click the eye icon for the current layer (e.g., *Help Overlay*) in order to hide the layer.



Add a Help Button

Now that the help overlay layer is completed, make sure the first layer is the current one in the **Layers** window by clicking to select **Layer 1**. In the toolbar, click **Components** and choose **Button**.

Double-click the button or choose the toolbar option to edit its text, and type the new button text as *Help*.

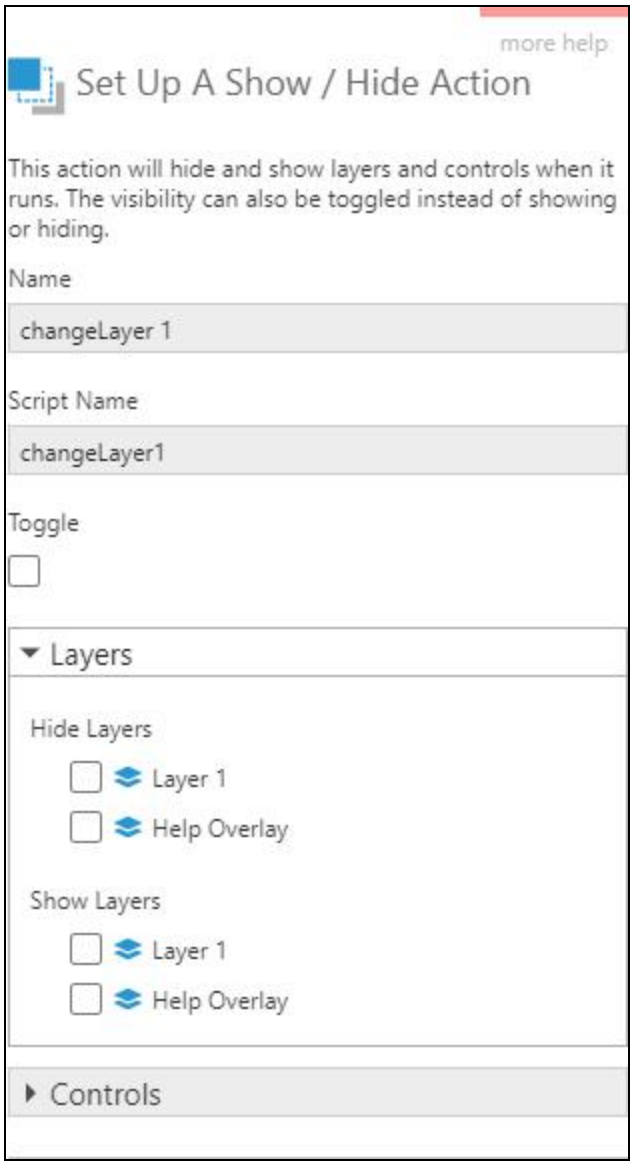


Set Up the Show / Hide Interaction

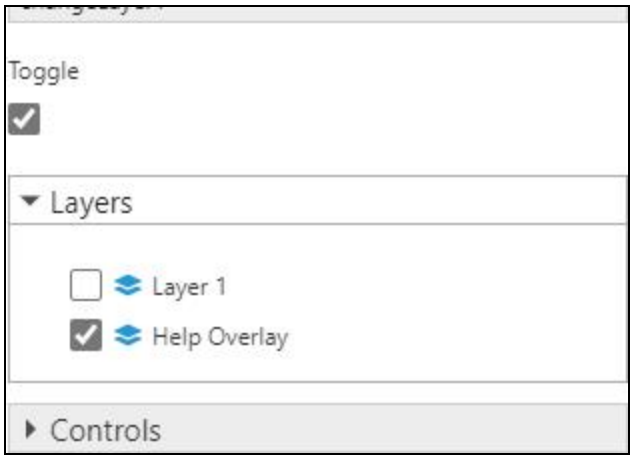
Click **Set Up Interactions** in the toolbar, then choose **Show / Hide**. (This is also known as *Change Layer* and it is labeled this way in earlier versions.)

In the *Set Up A Show / Hide Action* dialog, you can select a layer under **Show Layers** to be shown. To show one or more elements individually instead of an entire layer, click to expand **Controls** and select elements under **Show Controls**.

When showing a layer or control, you may want to set up a separate interaction to hide the same content later. For example, viewers could click on the layer's content to hide it again, or you could add a separate close button.



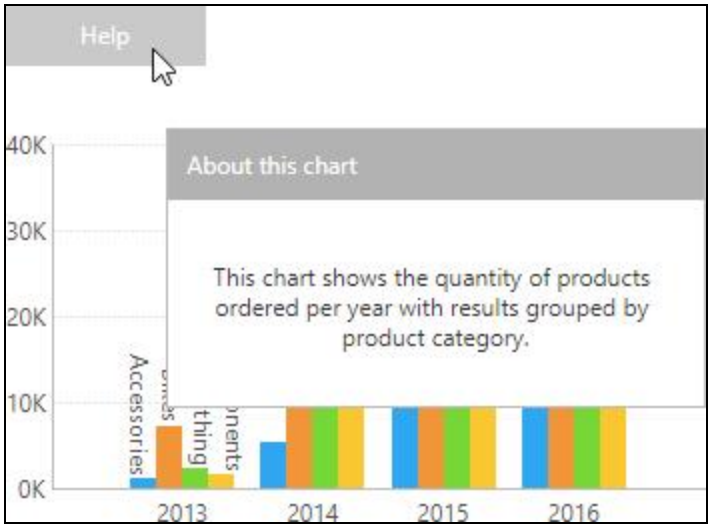
To use a single button both to show and then hide content, click to enable the **Toggle** checkbox option, then click to select the items to alternate between shown & hidden (e.g., the *Help Overlay* layer).



Click the submit button at the bottom of the dialog to finish setting up your interaction for button clicks.

Test the Help Overlay

To test your interaction, click **Sandbox View** in the toolbar to view the dashboard in a new browser tab, and then click your **Help** button.



Note: You can also switch to View mode in the toolbar, but changes that you make through interactions when viewing such as hiding and showing layers will be saved if you later switch back to edit mode and allow them to be auto-saved.

Scripting Layers

Change layer actions can be used to show, hide, or toggle layers from any built-in event, but if you need to trigger the same action from your own script instead, you can use JavaScript like the following:

```
var canvasService = this.getService("CanvasService");
var helpLayer = canvasService.getLayersByFriendlyName("Help Overlay Layer")[0];
// var helpLayer = this.baseViewService.currentView.control.layers[1];
if (helpLayer.hidden)
	canvasService.showLayers([helpLayer]);
else
	canvasService.hideLayers([helpLayer]);
```

This script gets the layer by its display name such the one assigned to the layer in this example, or you can get it by index as indicated in the commented-out code.

For more information, see:

- Using Interactions
- Layers and Groups
- Add a Change Layer Action on Ready Event
- Design Overview

Change Label Color By Script

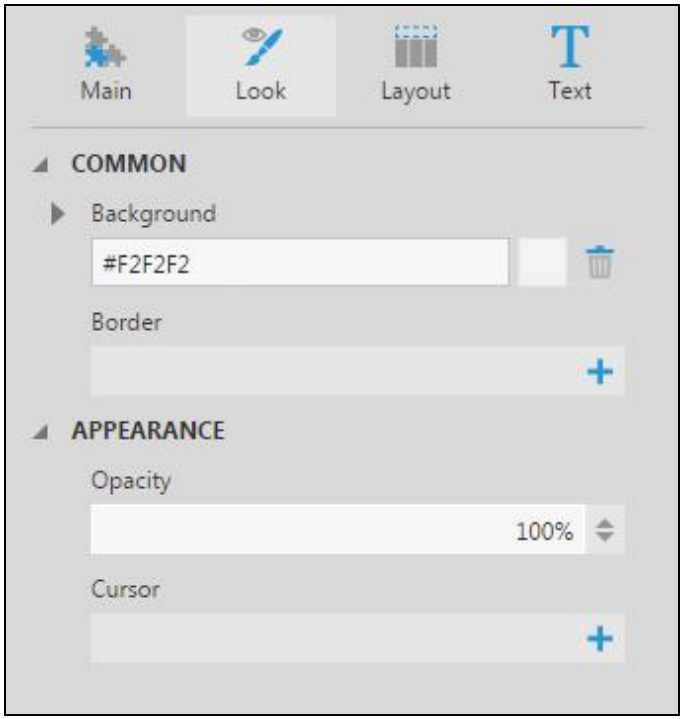
This applies to: *Managed Dashboards, Managed Reports*

This article shows you how to change a label or another component's background color using script. For example, you may want to change the color of the background of a component when it is clicked or hovered over.

Setting Up

In this example, we have a label component.

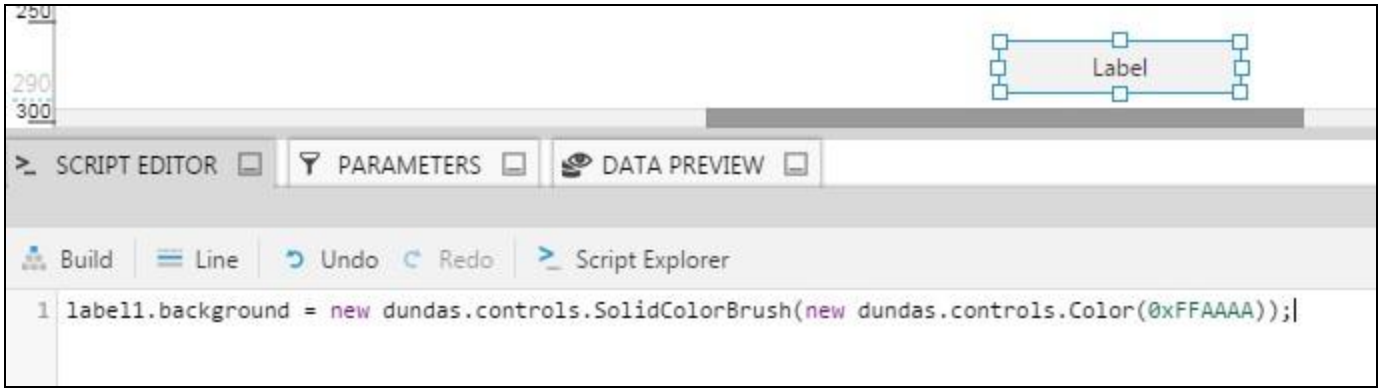
It has the following **Look** properties, with a background color of #F2F2F2.



Adding the Script

If you want to change the label when an action is performed, add the following script to a list in the **Properties** window's Actions section, such as **Click**:

```
label1.background = new dundas.controls.SolidColorBrush(new dundas.controls.Color(0xFFAAAA));
```



This uses the following API properties and classes:

- Label.background Property
- dundas.controls.SolidColorBrush Class
- dundas.controls.Color Class

Once this is added, you can choose Sandbox View in the toolbar to test the script.

For more information, see:

- Script Library: Using different brushes
- JavaScript API
- Using the Script Editor
- Edit Versus View Mode and Sandbox View

Connect to OLAP Data and Apply a Formula

This applies to: *Managed Dashboards, Managed Reports*

This walkthrough shows you how to connect to OLAP data such as an Analysis Services database, view the data on a dashboard, and apply formulas to data. We edit a dashboard here, but you can follow similar steps for editing metric sets.

You can work with data from an OLAP database the same way as with data from other data sources, except that cubes have already been completely prepared in the database. These existing cubes are often used directly instead of data cubes, and they contain measures and dimensions or hierarchies rather than tables.

Related video: Formulas

Connect to OLAP Data

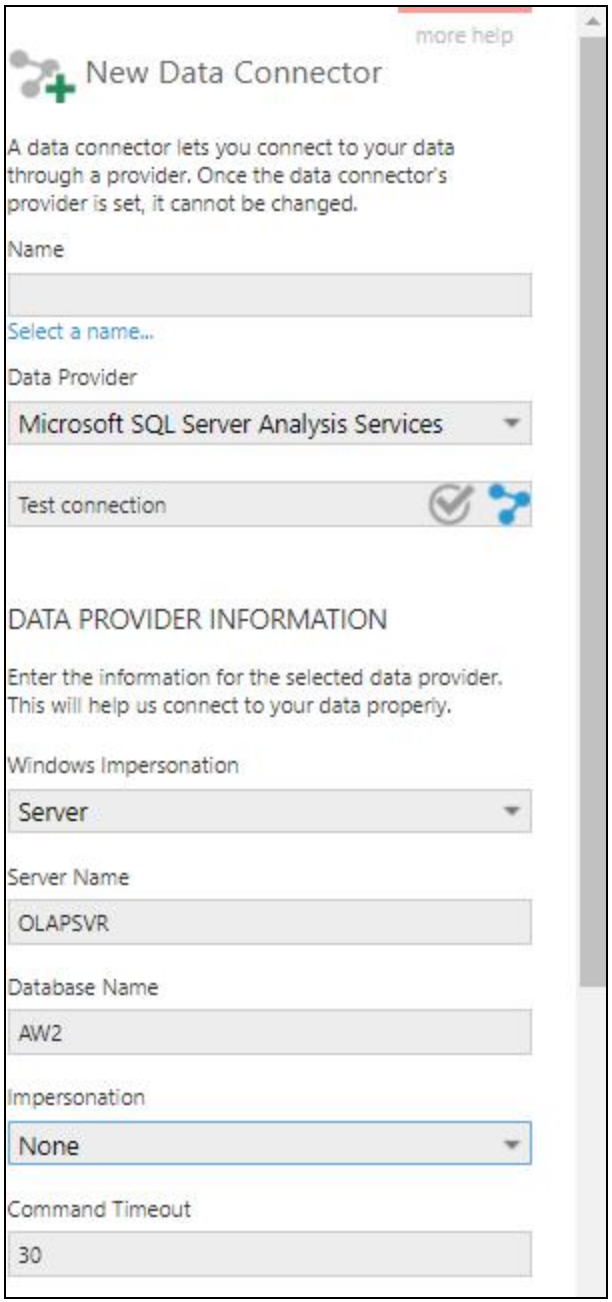
To connect to OLAP data, you'll need to create a new data connector, create a new dashboard and create a measure, then add hierarchies to interact with the data.

Create a New Data Connector

Create a data connector from the main menu or one of the other ways mentioned here: Connect to Data and View it on a Dashboard.

Click in the **Name** box to type in a file name for your data connector, and set the **Data Provider** to the type of data source. We chose Microsoft SQL Server Analysis Services in the image below as an example, which is also used for connecting to Azure Analysis Services.

Fill in the fields that appear below, such as **Server Name** and **Database Name**, and change any of the other options as needed to successfully access your data source. The example used in this article is the Adventure Works sample database for Analysis Services.


more help
New Data Connector
A data connector lets you connect to your data through a provider. Once the data connector's provider is set, it cannot be changed.
Name
Select a name...
Data Provider
Microsoft SQL Server Analysis Services
Test connection
DATA PROVIDER INFORMATION
Enter the information for the selected data provider. This will help us connect to your data properly.
Windows Impersonation
Server
Server Name
OLAPSVR
Database Name
AW2
Impersonation
None
Command Timeout
30

Optionally, click **Test connection** to verify the settings you entered.

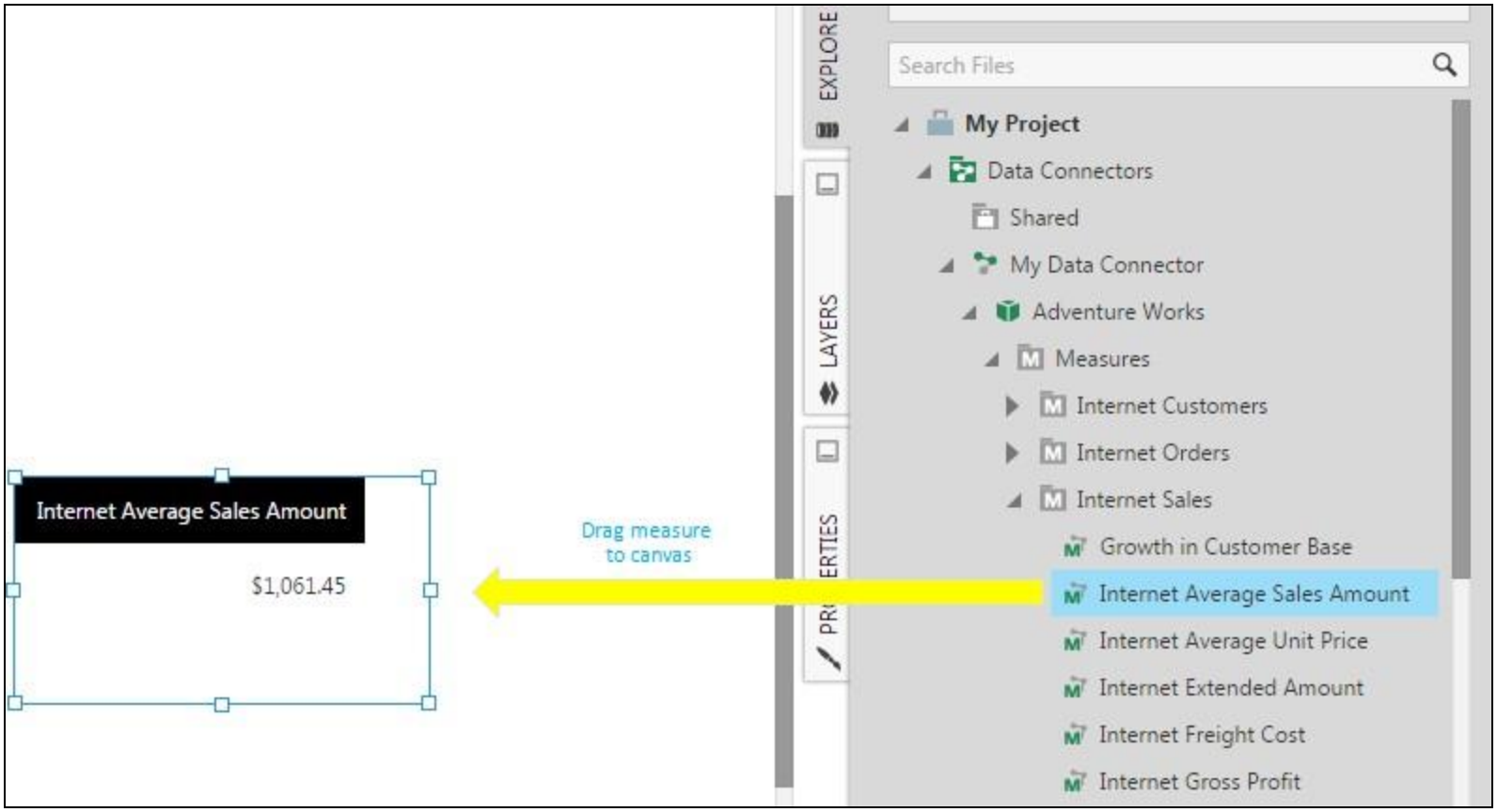
Click the submit (checkmark) button at the bottom to check in the data connector and discover the available cubes in the background.

Create a New Dashboard and Add a Measure

You can now use this data connector in metric sets or in views such as dashboards.

For example, create a new a dashboard from the menu, selecting the *Blank template* option.

Go to the **Explore** window, locate the data connector you just created, and expand it to see a list of its available cubes. Expand a cube to see a list of available measures and dimensions.



Find the *Internet Average Sales Amount* measure and drag it onto the dashboard canvas. A table visualization is automatically created, displaying a single measure value because no hierarchies have been added yet.

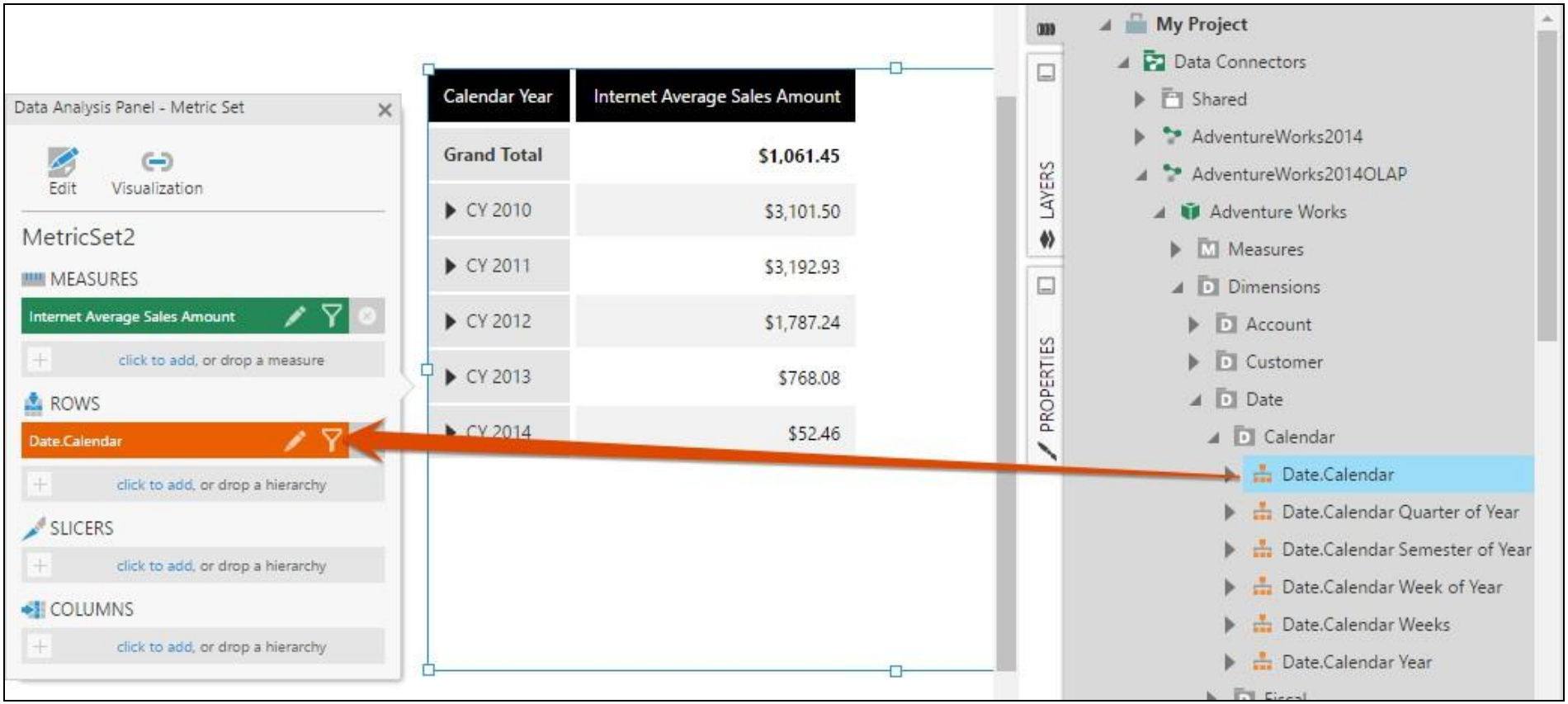
Add Hierarchies

Next, go to the **Explore** window, expand the *Dimensions* folder under your cube, and find the Date.Calendar hierarchy.

Drag the Date.Calendar hierarchy and drop it onto the label drop a hierarchy under **Rows** in the *Data Analysis Panel* that appeared to the left of the table visualization.

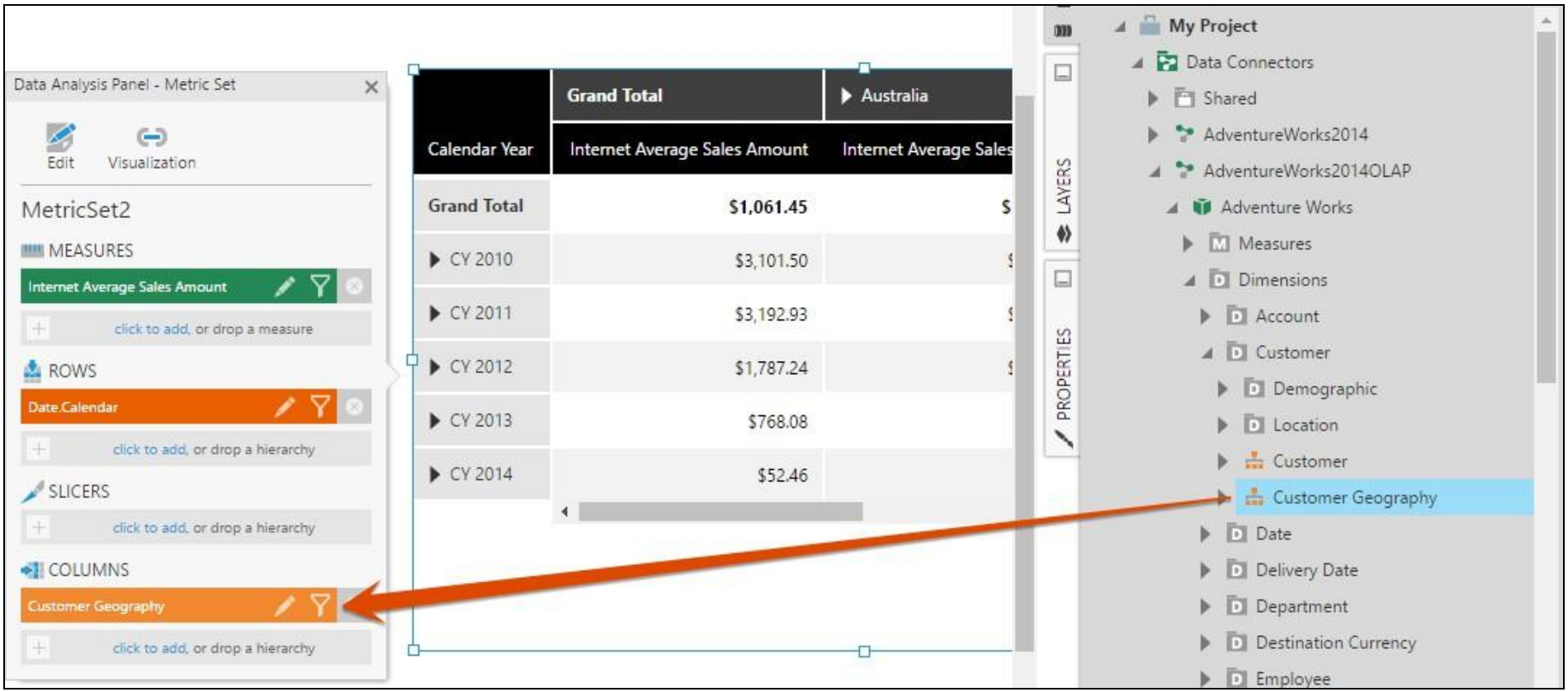
Note: Set up date mapping to take full advantage of dates and time dimensions from OLAP cubes.

The table visualization automatically updates to show expandable date members along its rows axis.



Similarly, drag the Customer Geography hierarchy to **Columns** in the Data Analysis Panel.

The table visualization updates to additionally show expandable country members along its columns axis.

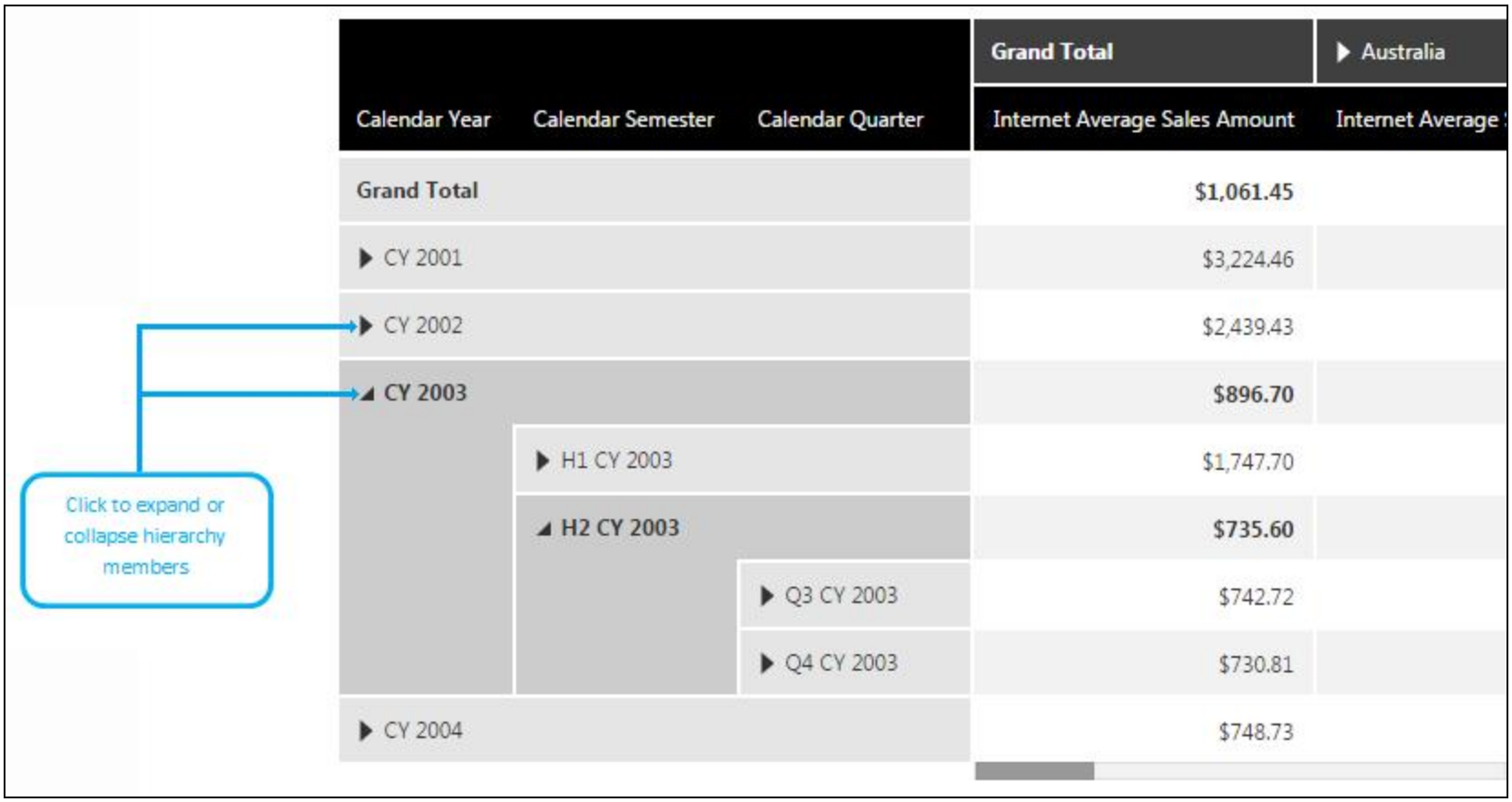


Note: By default, calculated members in the OLAP dimension will be excluded. To include them, right-click on the dimension in the Explore window, click **Utility Hierarchy**, and select **Enable**.

Switch to View Mode

Resize the table visualization a bit larger and then click **View** in the toolbar so you can start interacting with the dashboard.

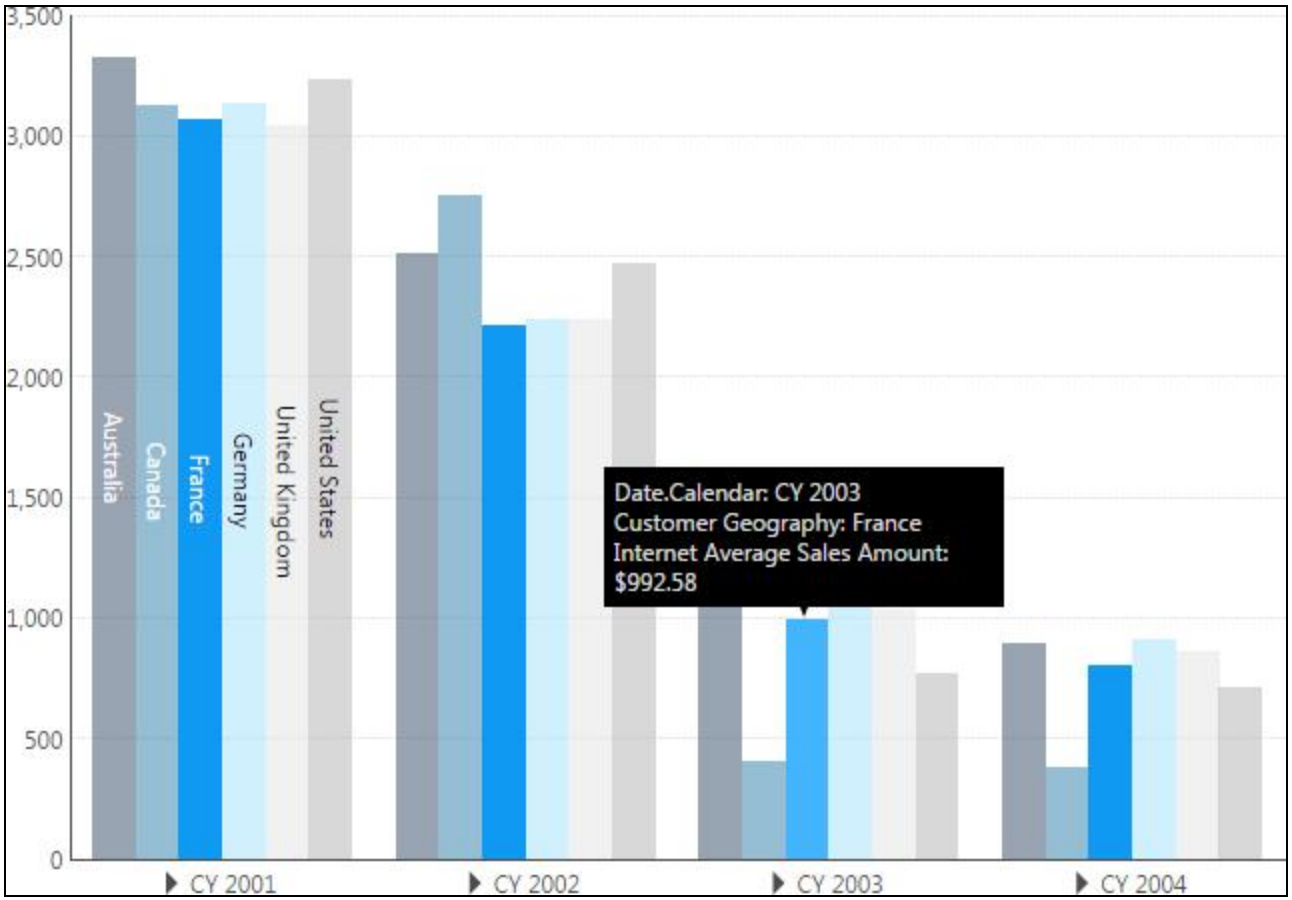
For example, click the triangle in the CY 2003 cell to see calendar half year members. Click the triangle in the H2 CY 2003 cell to see calendar quarter members.



Re-Visualize As a Bar Chart

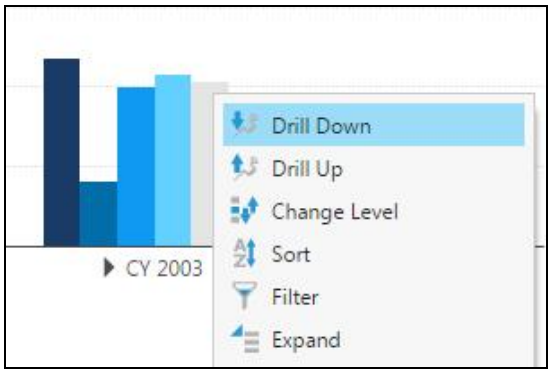
While in View mode, right-click (or long-tap) over the table visualization, click **Re-Visualize**, and then click **Bar**.

The table changes to a bar chart that displays calendar year (date) members along the X axis, and Internet Average Sales Amount measure values on the Y axis. The chart also shows multiple series, where each series (bars of the same color) corresponds to a specific Customer Geography member (e.g., country). You can see this if you hover over a bar to see its corresponding information in a tooltip.

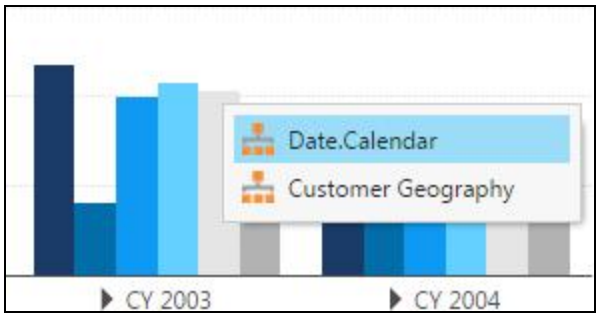


Drill Down On the Date Hierarchy

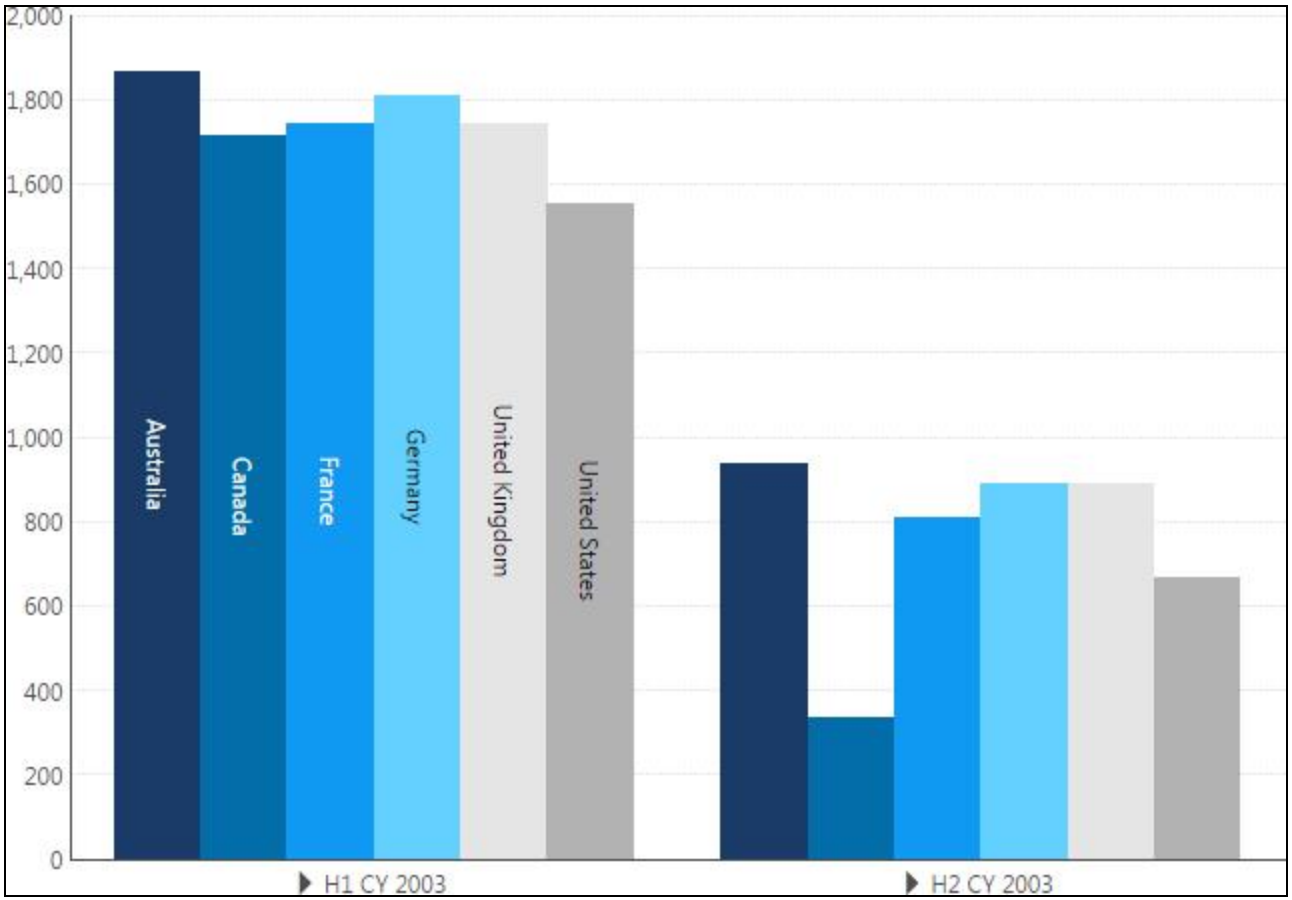
Right-click (or long-tap) a CY 2003 bar on the chart.



From the context menu that appears, choose **Drill Down**, and then select **Date.Calendar** as the hierarchy to drill down on.



The chart now displays date members along the X axis that are one hierarchy level down and which correspond to the CY 2003 member. You can **Drill Up** the same way, or switch back to Edit mode. If you switch back to Edit mode, you'll see that the chart remains in the same state it was in before you exited View mode (e.g., in the drilled down state).



Using Formulas

In Managed Dashboard and Reports, you apply formulas by writing small expressions much like you would in a spreadsheet application. A formula expression can be as simple as a constant number value, or it can use math and formula functions.

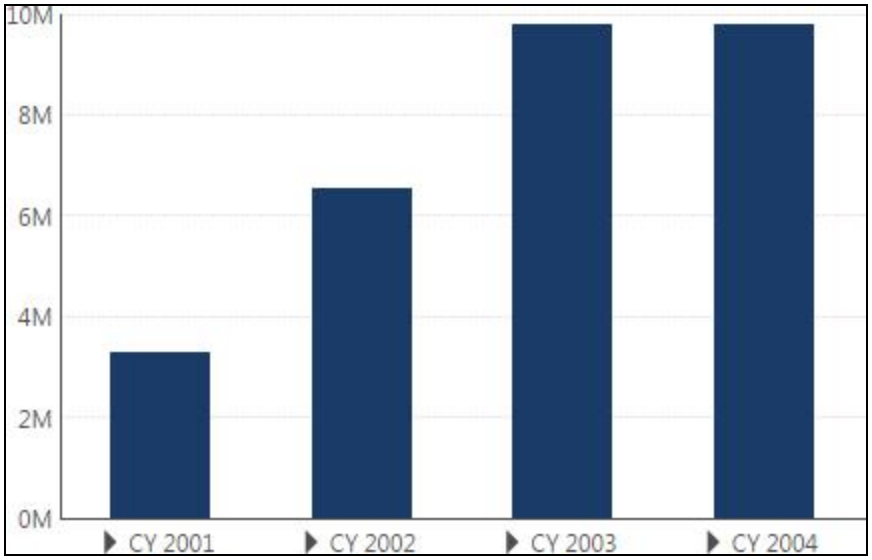
Create Another Dashboard

For this part of the walkthrough, create a second dashboard.

Go to the **Explore** window and drag the *Internet Sales Amount measure* to the dashboard canvas as before. A table visualization is added to the dashboard, showing a single measure value. The *Data Analysis Panel* appears to the side.

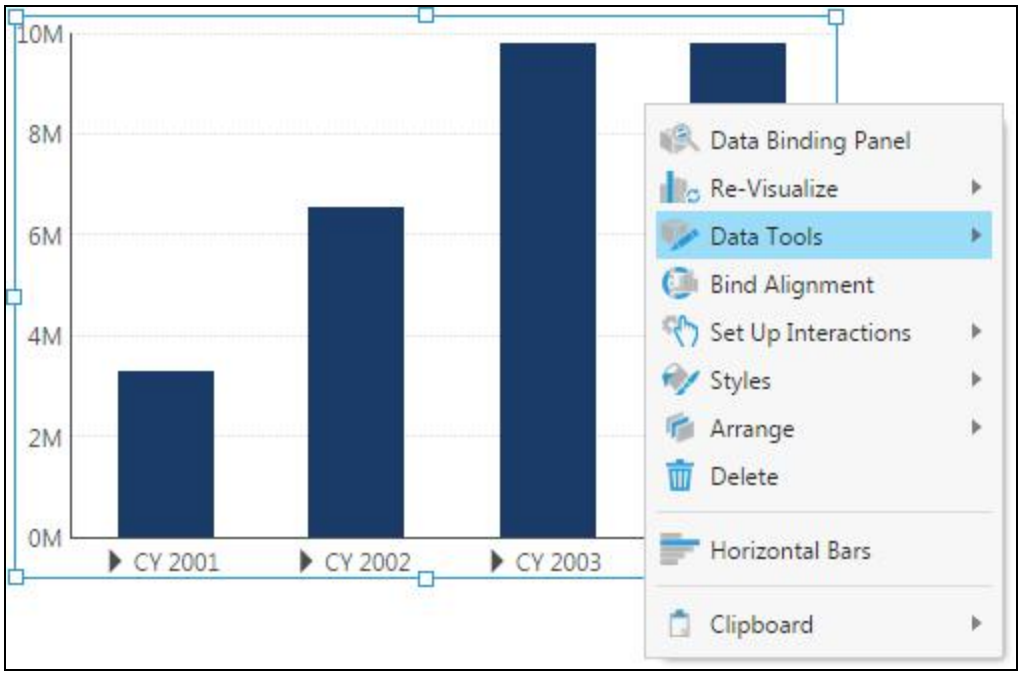
Drag the *Date.Calendar* hierarchy to **Rows** in the Data Analysis panel. The table visualization is updated with calendar year members along its rows axis.

Select the table visualization, go to the toolbar, click **Re-Visualize**, and then click **Bar** from the drop-down menu. The table visualization changes to a bar chart.

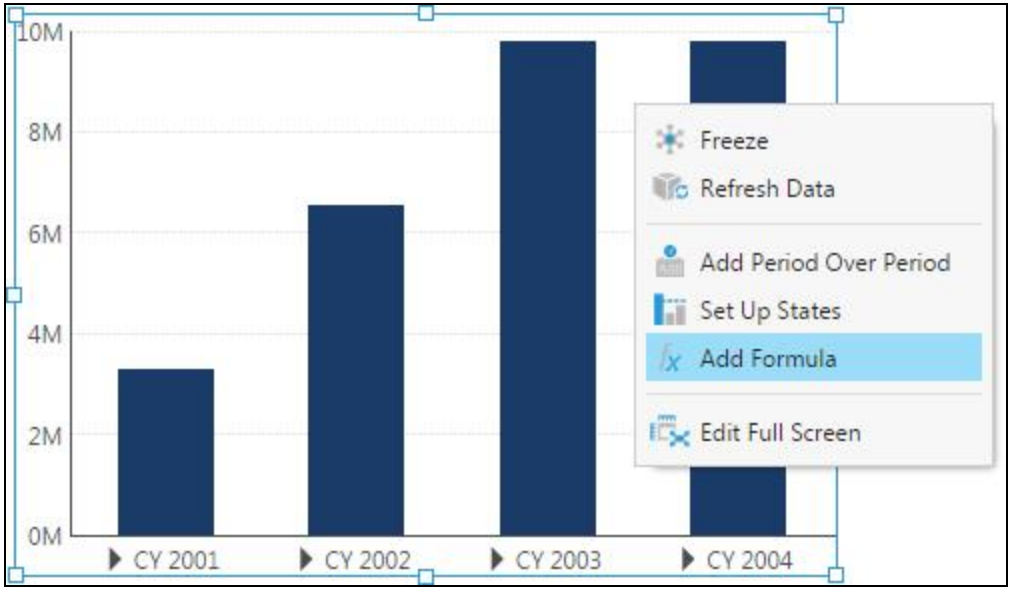


Add a Formula To the Bar Chart

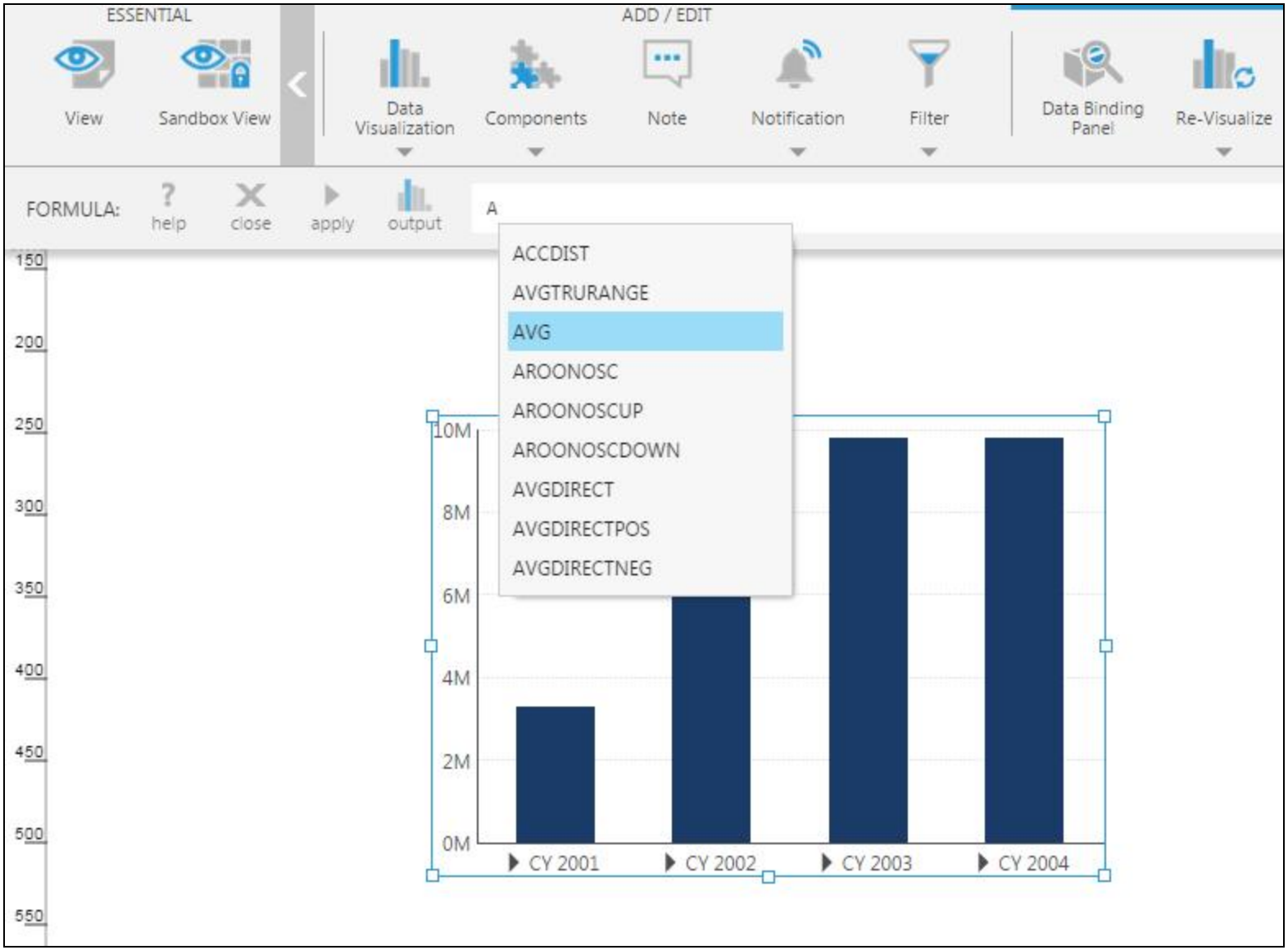
To begin applying a formula to the bar chart, choose **Data Tools** in the toolbar, or right-click (long-tap) the chart and choose it from the context menu.



Then choose **Add Formula**.



Below the toolbar, you'll see a formula bar appear. This lets you type in a formula expression such as a call to a formula function. For example, to apply a formula that performs an average operation on the bar chart, go to the formula bar and type the letter **A**. From the auto-complete menu, select the **AVG** function.

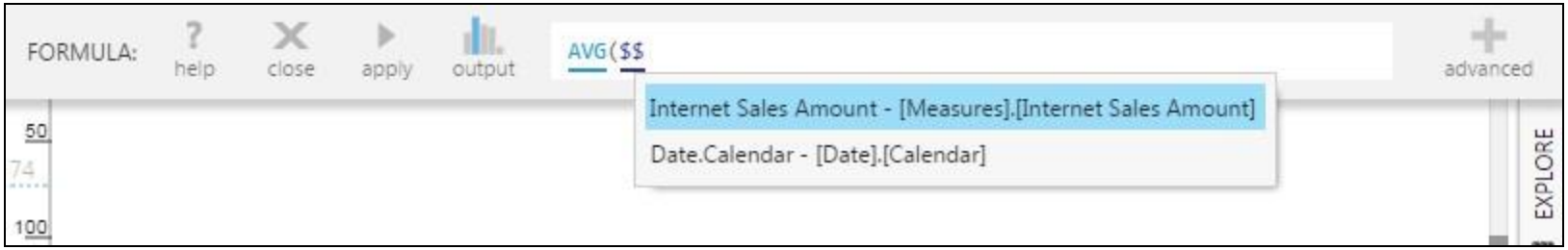


Type an opening parenthesis after *AVG* like the following:

```
AVG(
```

Once the above has been entered into the formula bar, a tooltip popup appears describing the function and its required arguments.

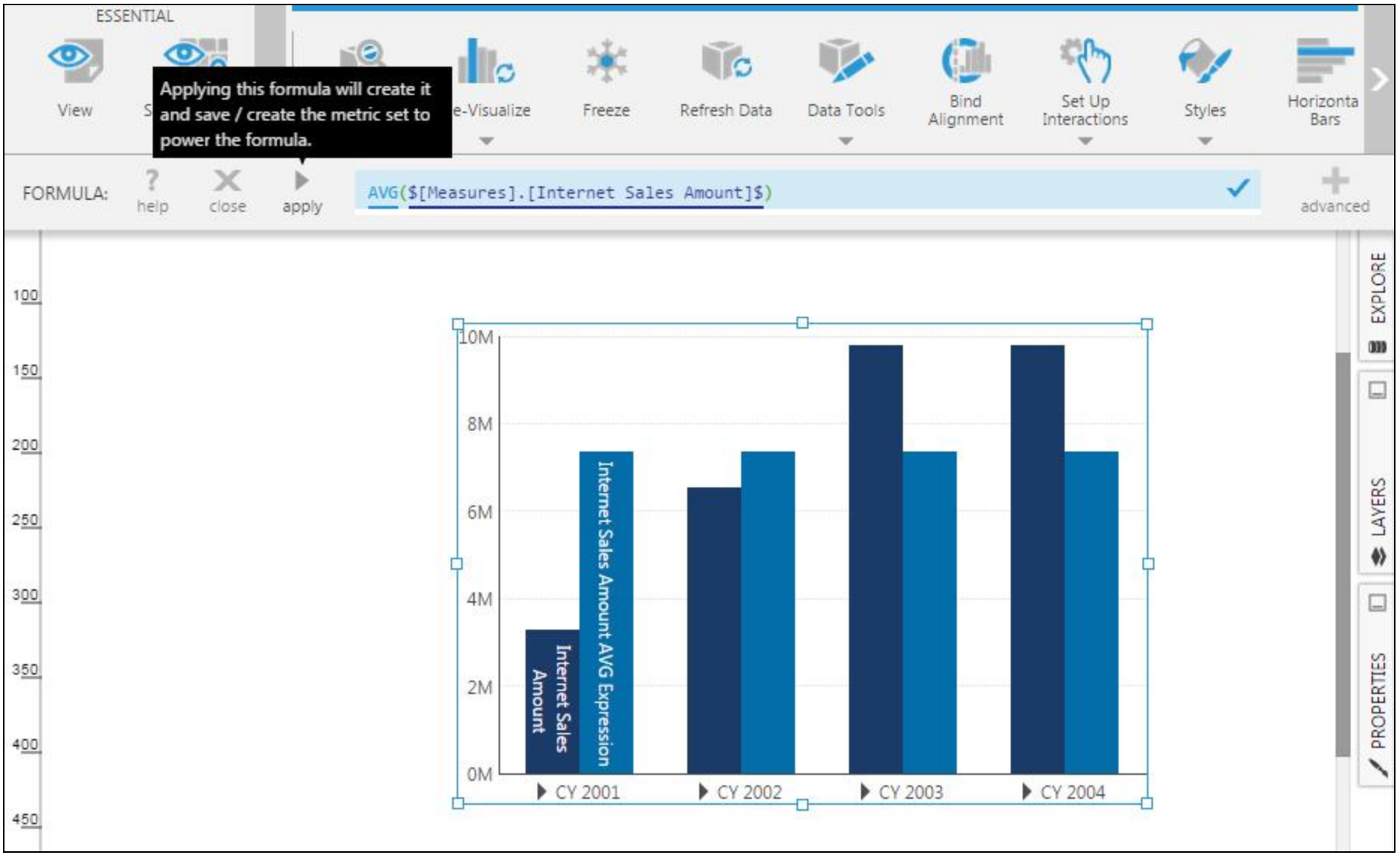
Specifically, the AVG function requires one argument, which is a set of data values. To indicate that you want to compute the average of the measure values displayed on the existing bar chart, **click any data point on the chart**, or type the **dollar sign** character (**$**) in the formula bar. This opens a menu that lets you choose an applicable argument for the function. For example, select [Measures].[Internet Sales Amount].



Make sure there is a closing dollar sign ($) followed by a closing bracket. The entire formula entry now reads:

```
AVG($[Measures].[Internet Sales Amount]$)
```

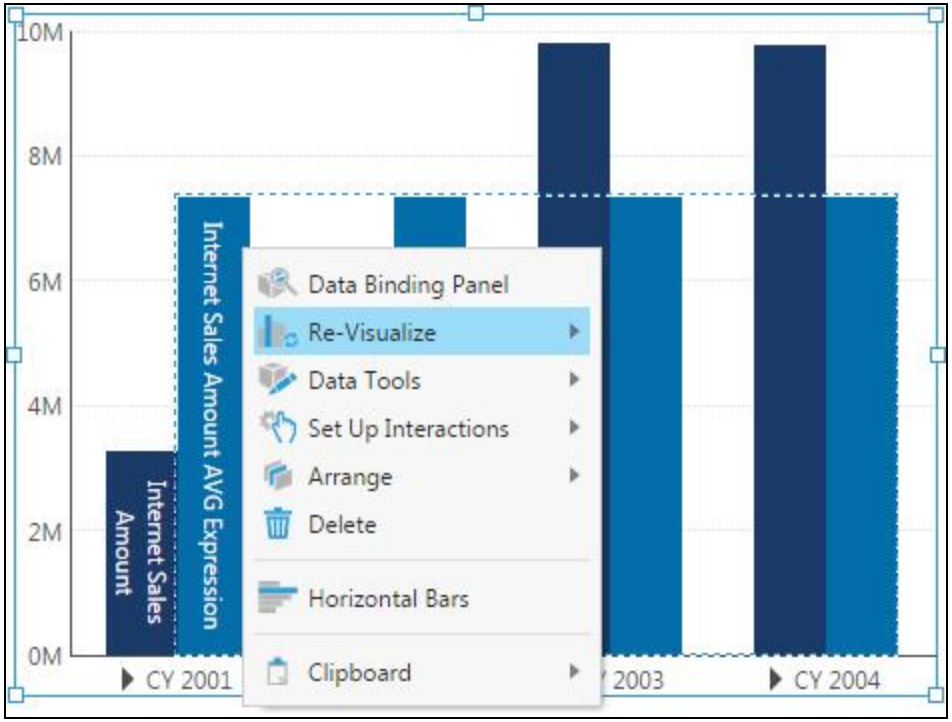
Click the **apply** button. If the formula syntax is correct, a checkmark is displayed in the formula bar, and the bar chart is updated to show the corresponding formula series.



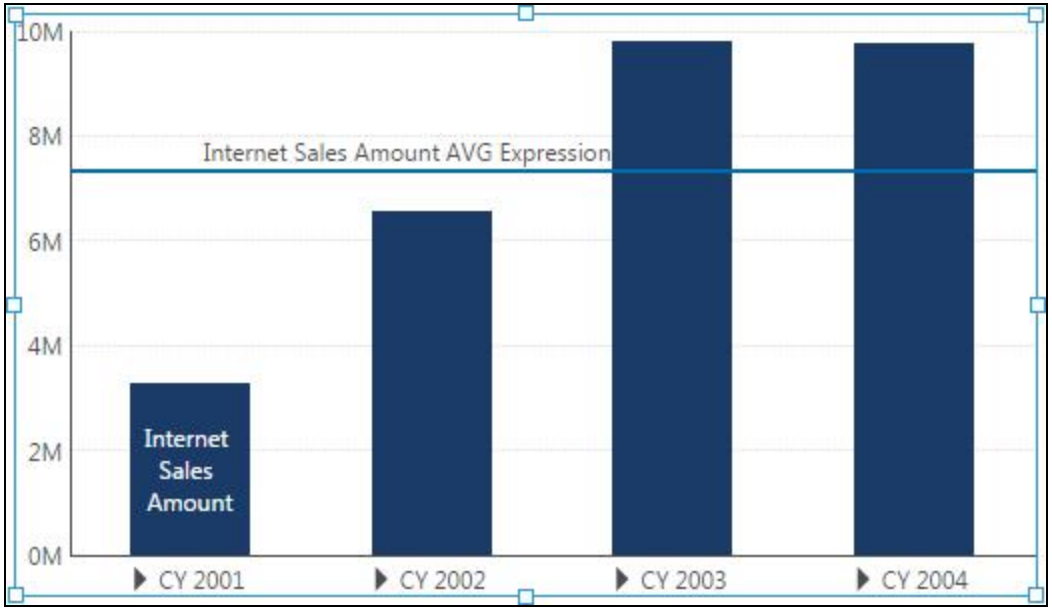
For more information on the kinds of formulas you can apply, see Adding Formulas.

Re-Visualize the Formula Series

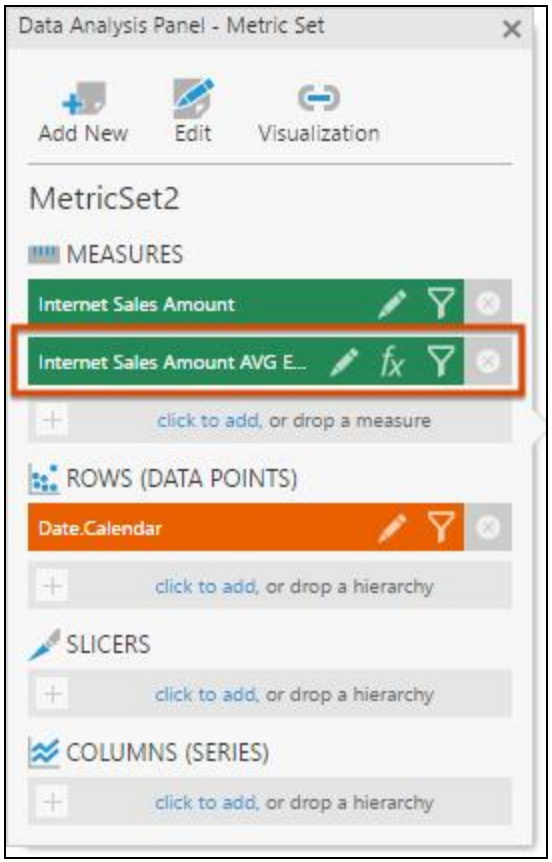
Next, right-click over one of the formula bars on the chart.



From the context menu that appears, click **Re-Visualize**, and then click **Line**. The formula series is now displayed as a horizontal line on the bar chart.



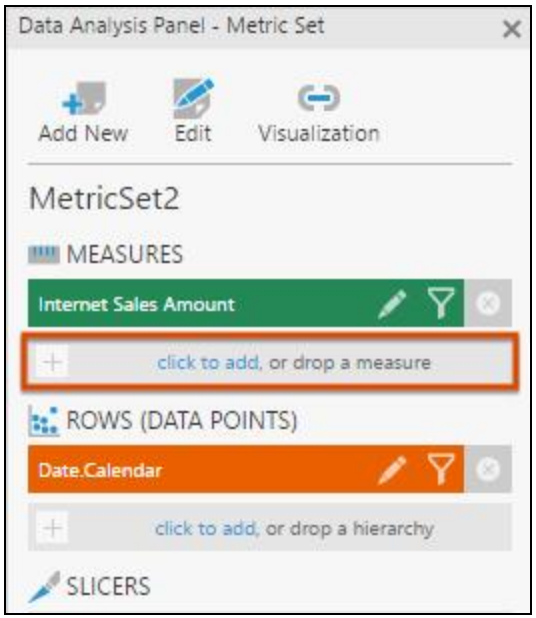
If you right-click over the chart and click **Data Analysis Panel**, you'll see that the formula result has been added as a second measure which is now part of this metric set. You can remove the formula result if desired by clicking its x icon on the right.



Note: When first adding a formula, you can click the **output** button in the formula bar to choose the chart type ahead of time.

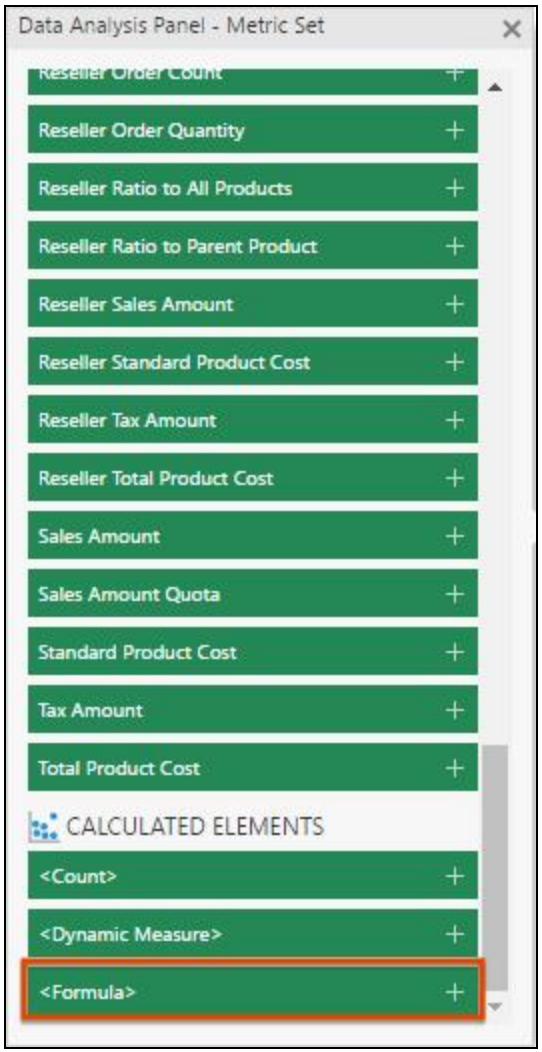
Adding a Formula From the Data Analysis Panel

You saw earlier how to add a formula from the toolbar or context menu. If you have the Data Analysis Panel open, you can also add a formula by clicking the **click to add** link in the *Measures* section.



You'll see a list of available measures to add to this metric set.

Scroll all the way down to the *Calculated Elements* section and click **<Formula>**. The formula bar will appear below the main toolbar as before.

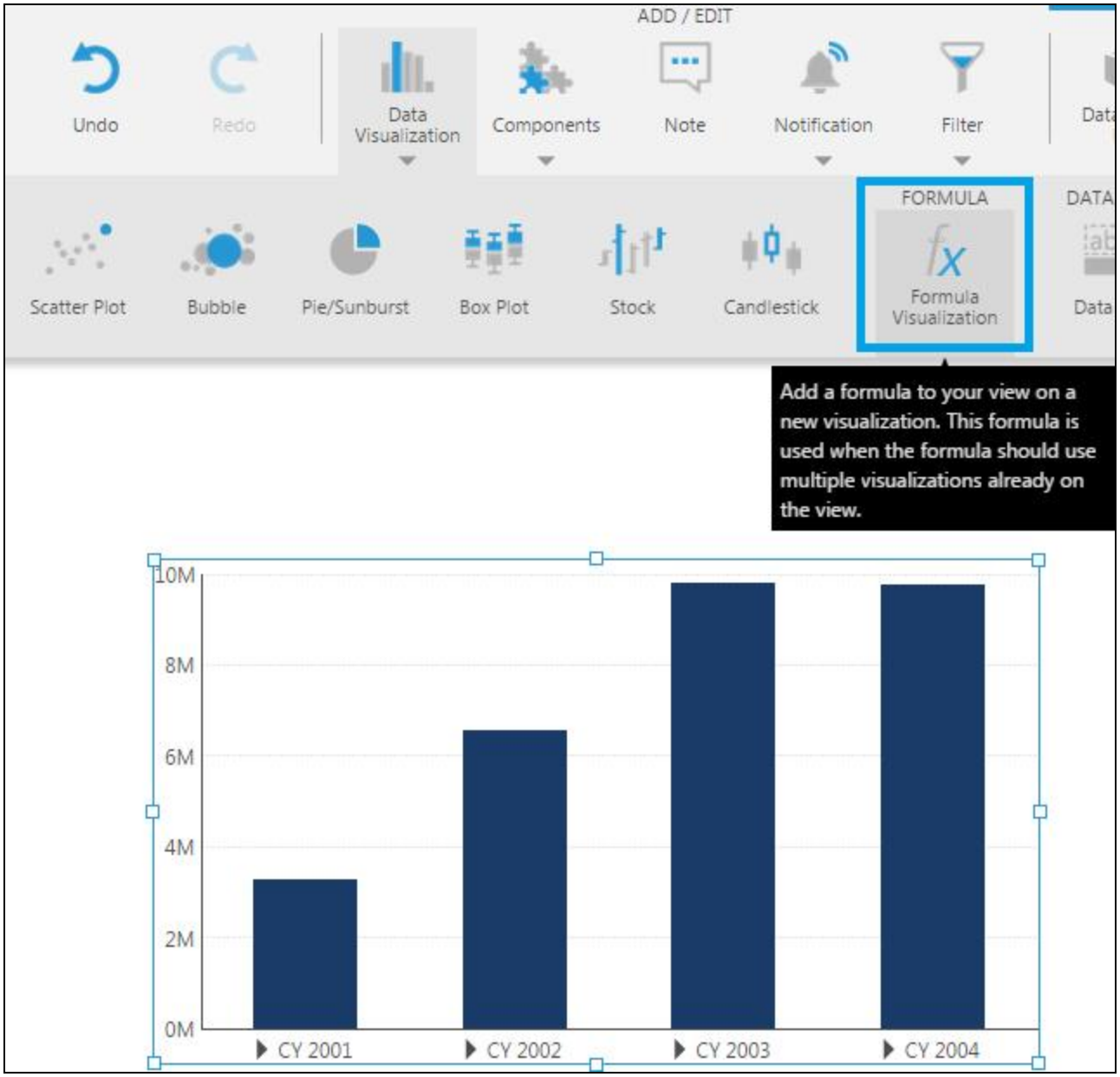


Adding a Formula As a New Visualization

The previous example showed how to add a formula as a second measure/series on an existing chart (metric set).

But you can also apply a formula on an existing visualization and have the result displayed in a new metric set and visualization added to the dashboard. This option will leave your original visualization and metric set unchanged.

As an example, first create a new dashboard as before. It should have a bar chart (e.g., chart1) that displays an *Internet Sales Amount* measure against a Date.Calendar hierarchy. Go to the toolbar, click **Data Visualization**, and then scroll until you find the **Formula Visualization** item.



Click **Formula Visualization** to open the formula bar.

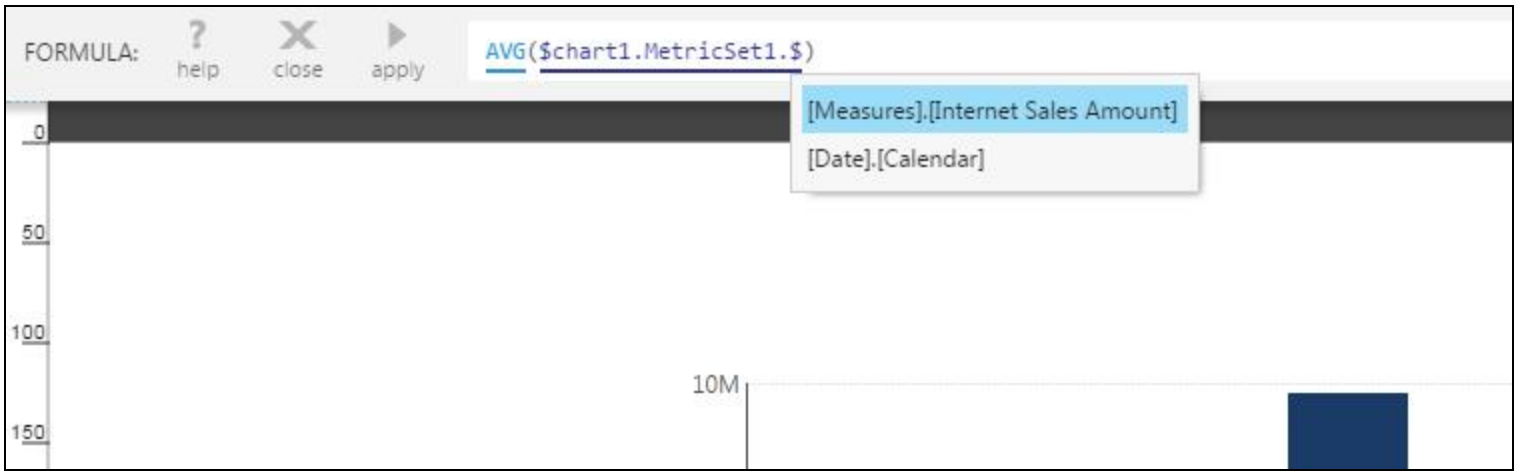
Type the letter **A**. From the auto-complete menu, select the **AVG** math function.

Now can refer to any or multiple visualizations on the dashboard as inputs to the formula, so the placeholders referring to measures include the visualization and metric set names. The easiest way to enter this is to **click on a data point on the chart**, or you can type it yourself as follows:

Type a dollar sign character ($). From the menu, select **chart1**.

Type a period character (.) and then from the menu, select **MetricSet1**.

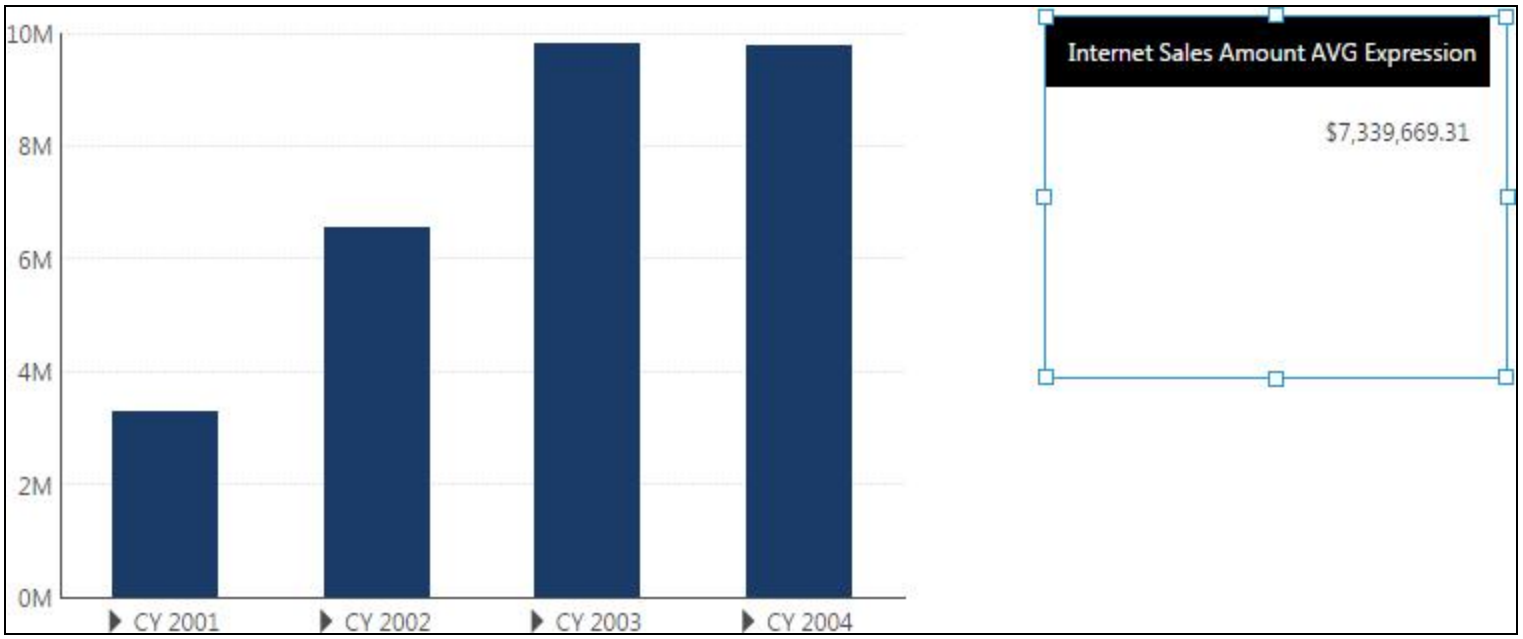
Type a period character (.) and then from the menu, select **[Measures].[Internet Sales Amount]**.



The formula entry looks like this so far:

```
AVG($chart1.MetricSet1.[Measures].[Internet Sales Amount]$)
```

Click apply. The formula result is displayed as a new table visualization (and metric set) which is added to your dashboard. This new metric set is called a formula metric set.



For more information, see:

- Video: Formulas
- Set Up Date Mapping on a Native OLAP Cube
- Using a Formula Visualization

Set Up Date Mapping on a Native OLAP Cube

This applies to: *Managed Dashboards, Managed Reports*

You can connect to an OLAP database, and right away begin dragging measures and hierarchies from its cubes to create metric sets, making use of its abilities to change levels, drill down, and more. There are additional

functions that you won't be able to perform on your OLAP data until you set up a *date mapping* on your OLAP cube's time hierarchies. A date mapping allows a time dimension hierarchy from your native OLAP cube to be interpreted as dates, allowing you to:

- Connect it to calendar and date filters, and optionally filter OLAP and non-OLAP data together.
- Display it along a date/time axis on a chart instead of a categorical axis.
- Customize the date formatting in visualizations.
- Add period over period comparisons.
- Apply forecasting formulas.

Related video: Introduction to OLAP Date Mapping

Setting Up Date Mapping

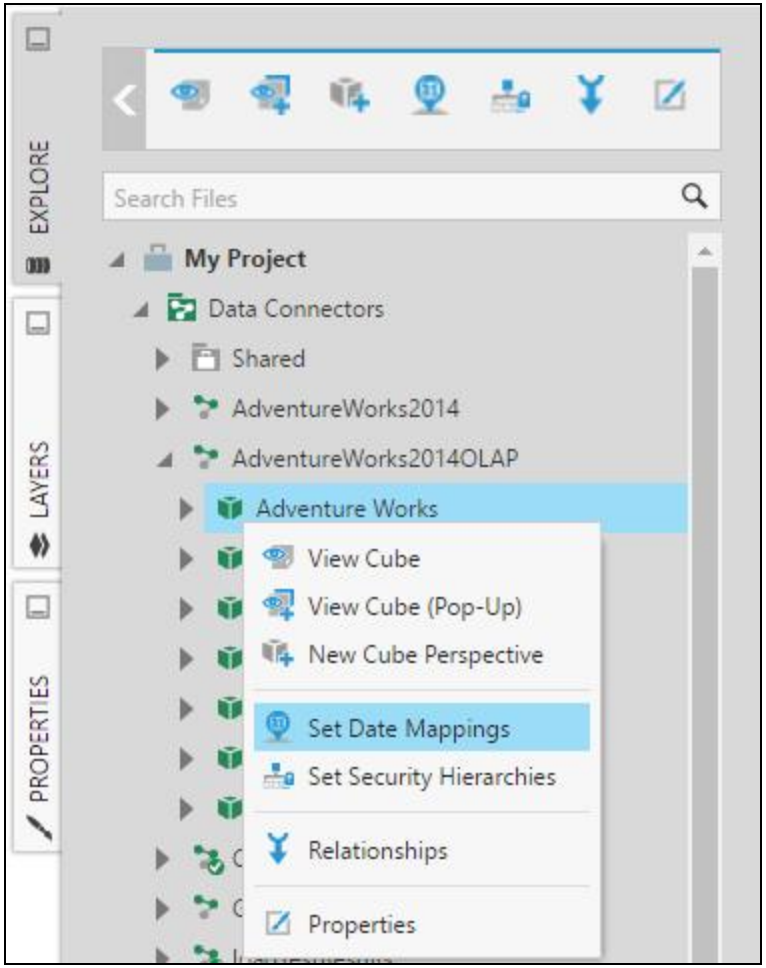
Connect To an OLAP Data Source

In order to set up date mapping, you need to connect to an OLAP data source which gives you access to a native OLAP cube. SeeConnect to OLAP Data and Apply a Formula.

Set Up Date Mapping on Your Native Cube

Find your OLAP data connector in the *Data Connectors* folder in the Explore window when you have another file open such as a dashboard or metric set.

Expand the data connector, right-click (or long-tap) the cube and select **Set Date Mappings** from the menu.

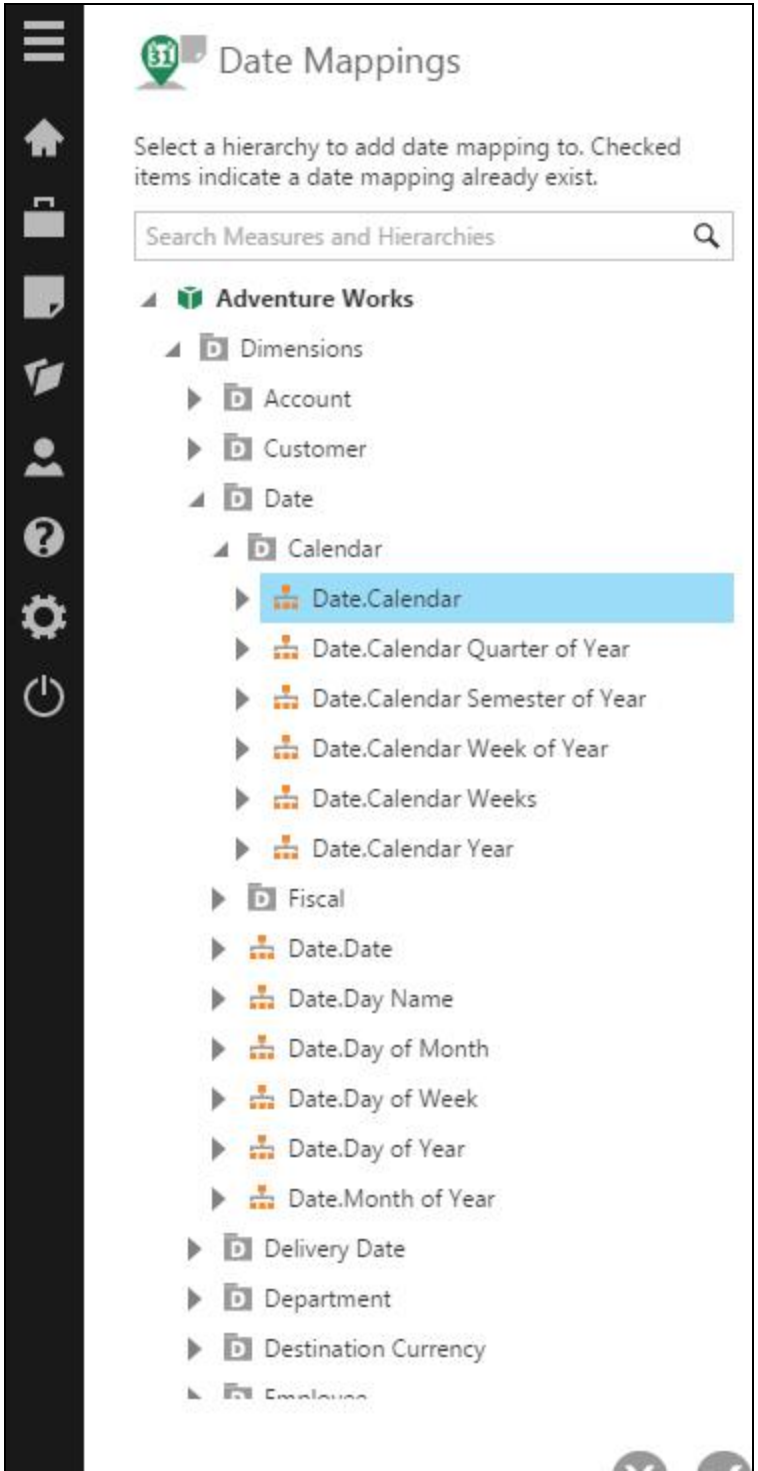


The *Date Mappings* dialog is displayed.

Select a Native Hierarchy

The *Date Mappings* dialog shows you the available dimensions from your native cube.

Expand the folders and select a hierarchy for date mapping.

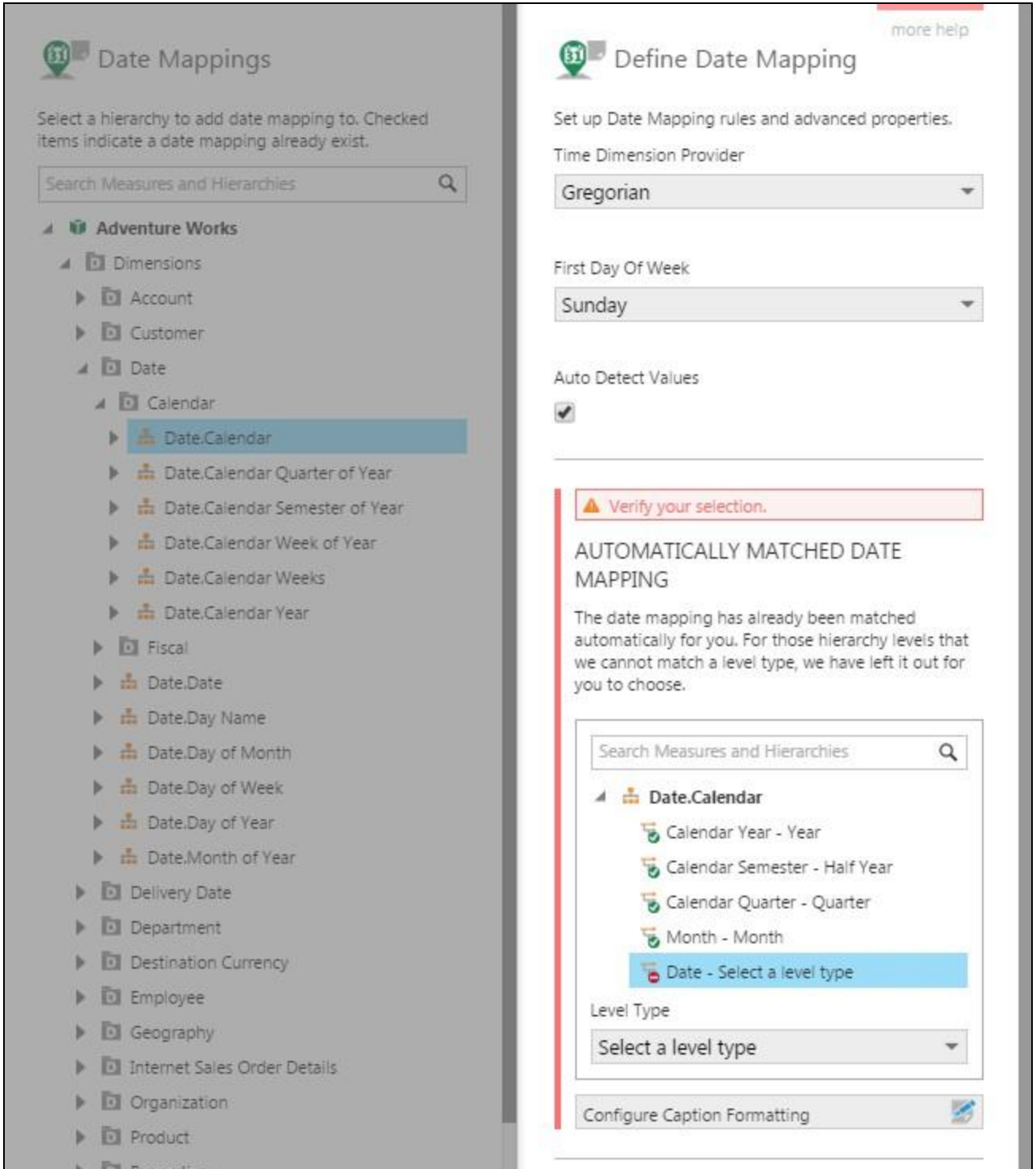

Date Mappings
Select a hierarchy to add date mapping to. Checked items indicate a date mapping already exist.
Search Measures and Hierarchies
Adventure Works
Dimensions
Account
Customer
Date
Calendar
Date.Calendar
Date.Calendar Quarter of Year
Date.Calendar Semester of Year
Date.Calendar Week of Year
Date.Calendar Weeks
Date.Calendar Year
Fiscal
Date.Date
Date.Day Name
Date.Day of Month
Date.Day of Week
Date.Day of Year
Date.Month of Year
Delivery Date
Department
Destination Currency

(If a hierarchy already has a date mapping, it will appear with a small checkmark icon beside it.)

Once you've selected a hierarchy, click **Submit** (check mark). This opens the *Define Date Mapping* dialog.

Define the Date Mapping

The *Define Date Mapping* dialog is where you configure the details of the mapping.

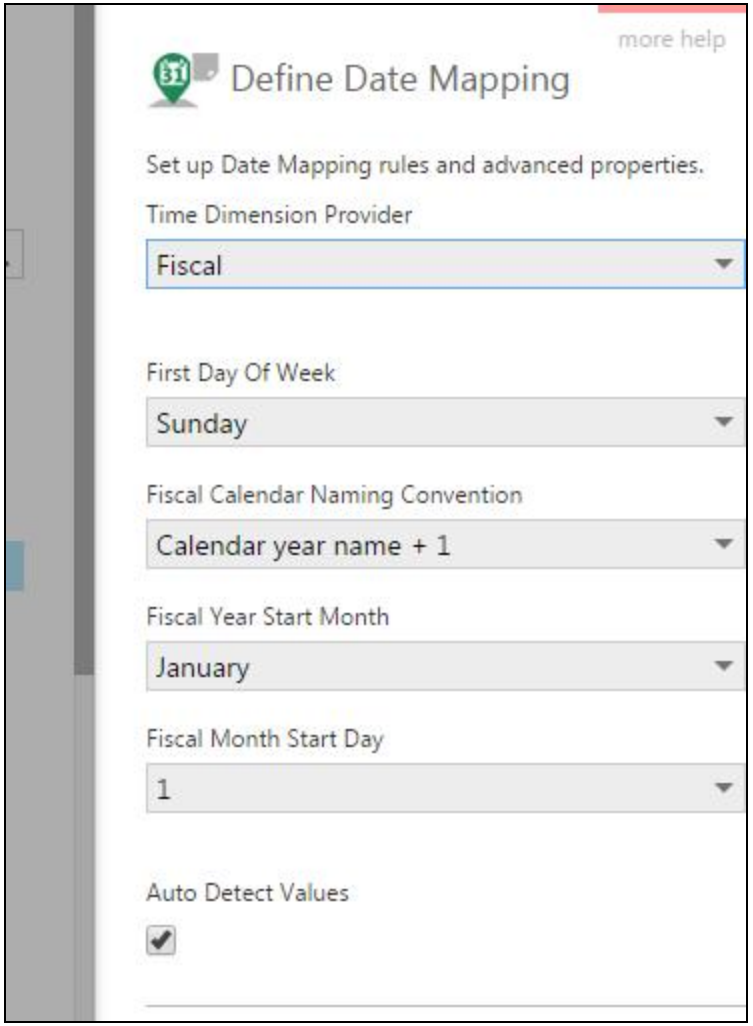

Date Mappings
Select a hierarchy to add date mapping to. Checked items indicate a date mapping already exist.
Search Measures and Hierarchies
Adventure Works
Dimensions
Account
Customer
Date
Calendar
Date.Calendar
Date.Calendar Quarter of Year
Date.Calendar Semester of Year
Date.Calendar Week of Year
Date.Calendar Weeks
Date.Calendar Year
Fiscal
Date.Date
Date.Day Name
Date.Day of Month
Date.Day of Week
Date.Day of Year
Date.Month of Year
Delivery Date
Department
Destination Currency
Employee
Geography
Internet Sales Order Details
Organization
Product
more help
Define Date Mapping
Set up Date Mapping rules and advanced properties.
Time Dimension Provider
Gregorian
First Day Of Week
Sunday
Auto Detect Values
Verify your selection.
AUTOMATICALLY MATCHED DATE MAPPING
The date mapping has already been matched automatically for you. For those hierarchy levels that we cannot match a level type, we have left it out for you to choose.
Search Measures and Hierarchies
Date.Calendar
Calendar Year - Year
Calendar Semester - Half Year
Calendar Quarter - Quarter
Month - Month
Date - Select a level type
Level Type
Select a level type
Configure Caption Formatting

Select a Time Dimension Provider

The first step is to select a **Time Dimension Provider**. This is the target calendar system you want to map to.

Symphony supports different types of calendar systems including: *Gregorian, Fiscal, ISO 8601* (week numbers), and *Reporting (Marketing)*. The default is *Gregorian*.

Depending on the provider chosen, you can set additional properties which are similar to the options available when creating a new time dimension. For example, for Fiscal or Reporting providers, you'll likely want to set the *Fiscal Year Start Month*.



Auto Detect Values

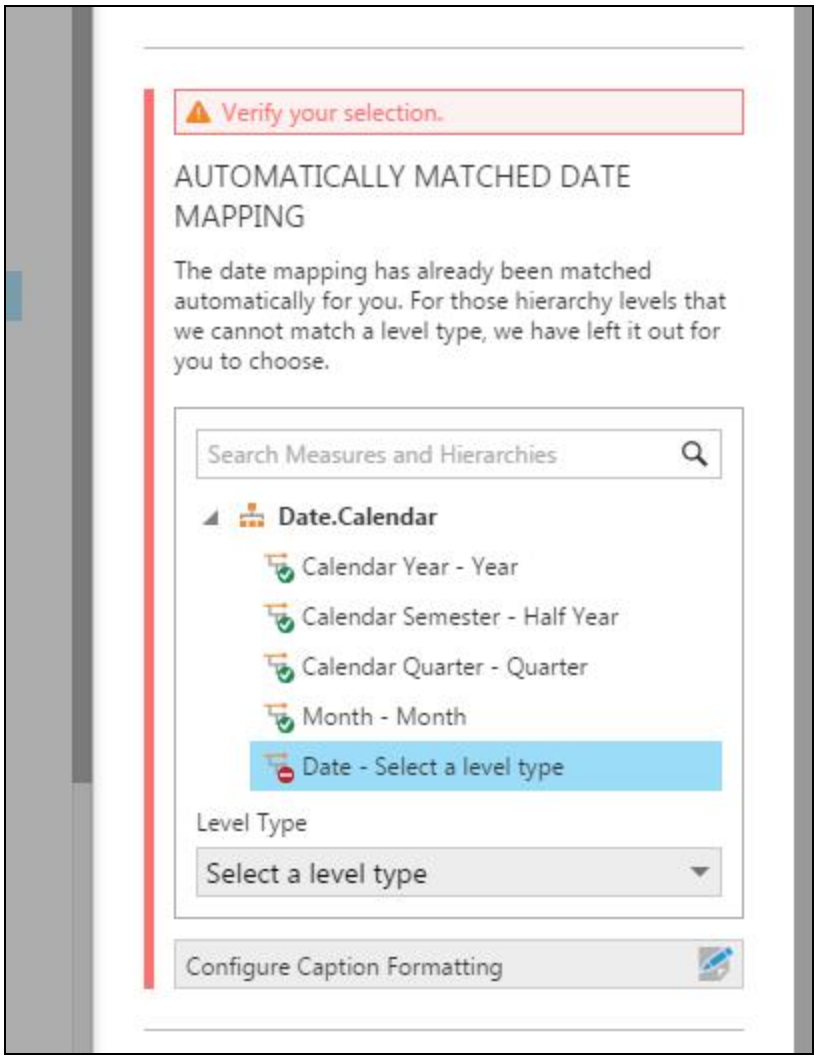
The **Auto Detect Values** option tells Symphony to automatically detect a start date for the mapping. Uncheck this option if you want to set the *start date* manually yourself.

> **Note:** In some cases, Symphony will be unable to automatically detect the start date and will show an error. If this happens, uncheck the Auto Detect Values option and manually specify a start date.

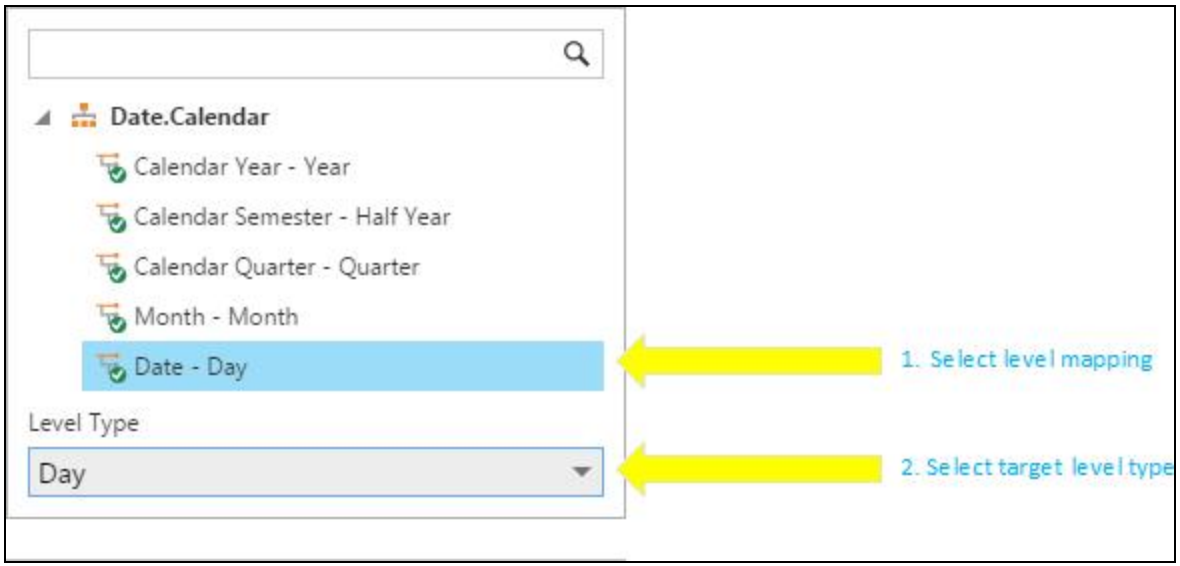
Automatically Matched Date Mapping

Symphony will automatically attempt to map each OLAP hierarchy level with a corresponding time dimension hierarchy level. For example, in the figure below, the OLAP *Calendar Semester* level is matched to the Gregorian *Half Year* level.

But for some hierarchy levels, you will need to configure the mapping yourself. These levels have a red 'minus' icon beside them. For example, in the figure below, the OLAP *Date* level could not be matched automatically.



To configure the mapping manually, select the mapping with the red minus icon, and then select the target level to map to from the drop-down list.



Configure Caption Formatting

Click **Configure Caption Formatting** if you want to customize how member captions should be formatted from what was set up in the cube. The options here are similar to the formatting options available when you edit a time dimension.

Note: This button may not appear when setting up a new date mapping. First submit the dialog to create the date mapping, then re-open the dialog to customize the formats.

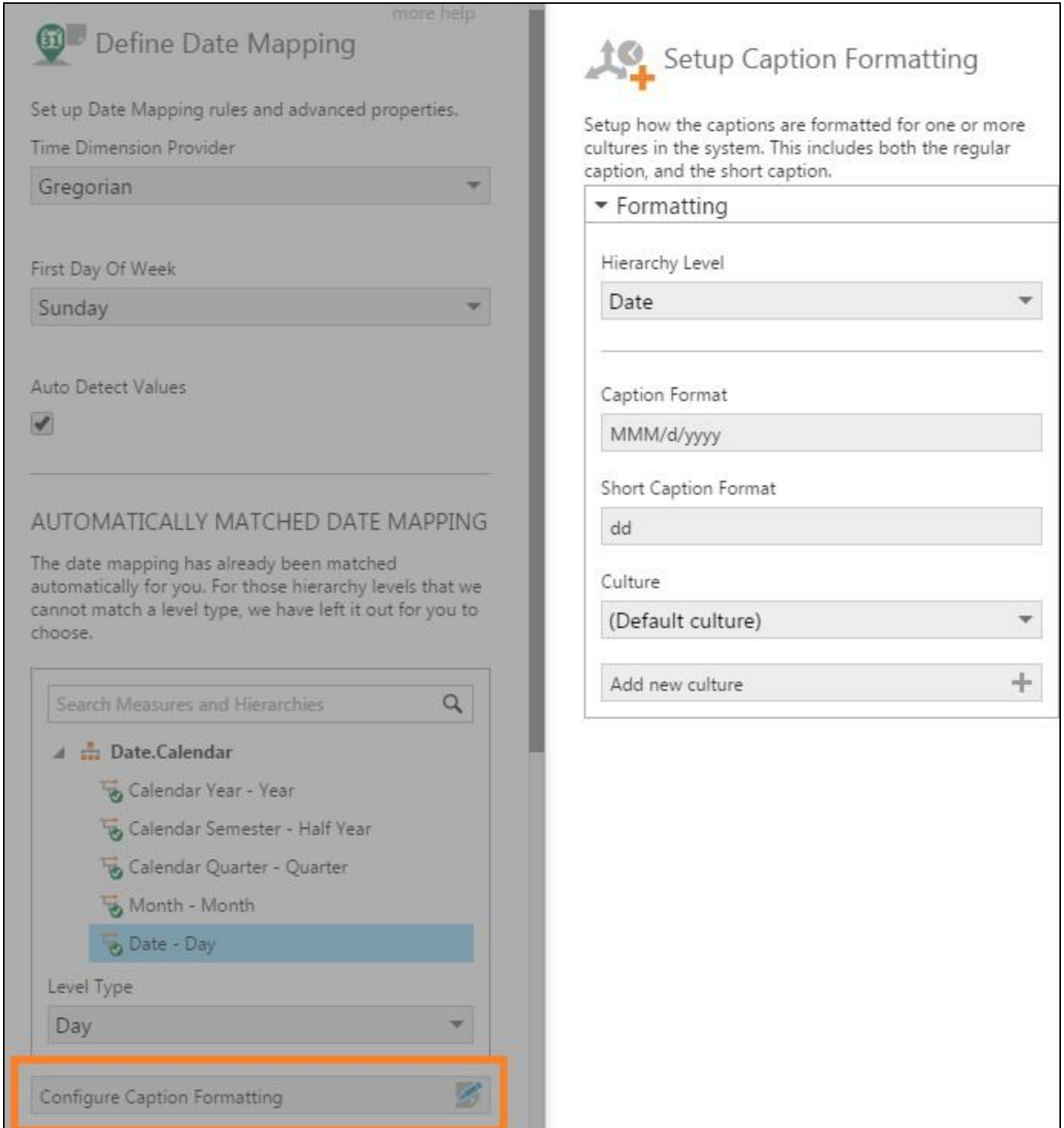
In the *Setup Caption Formatting* dialog, select an OLAP **Hierarchy Level** (e.g., *Date*).

Enter a regular **Caption Format**, such as *MMM/d/yyyy* for the Date level.

Enter a **Short Caption Format**, such as *dd* for the Date level.

If you want to apply different format settings for certain cultures (languages), select the **Culture** from the drop-down if already listed, or click **Add new culture** and enter a language tag (e.g., *en-US*).

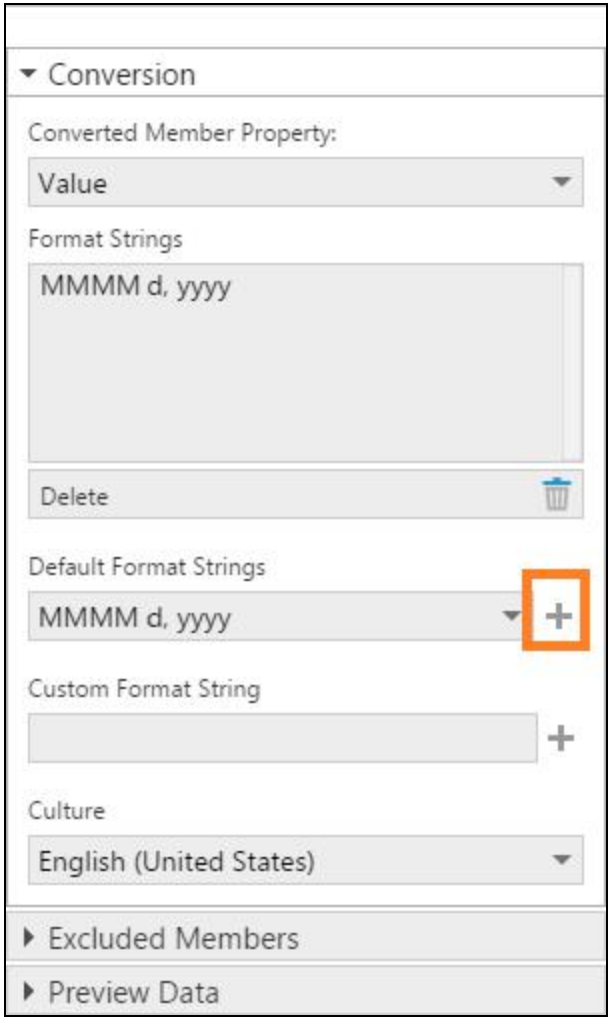
You can then select each culture available from the drop-down and set the Caption Format and Short Caption Format separately for each culture.


more help
Define Date Mapping
Set up Date Mapping rules and advanced properties.
Time Dimension Provider
Gregorian
First Day Of Week
Sunday
Auto Detect Values
AUTOMATICALLY MATCHED DATE MAPPING
The date mapping has already been matched automatically for you. For those hierarchy levels that we cannot match a level type, we have left it out for you to choose.
Search Measures and Hierarchies
Date.Calendar
Calendar Year - Year
Calendar Semester - Half Year
Calendar Quarter - Quarter
Month - Month
Date - Day
Level Type
Day
Configure Caption Formatting
Setup Caption Formatting
Setup how the captions are formatted for one or more cultures in the system. This includes both the regular caption, and the short caption.
Formatting
Hierarchy Level
Date
Caption Format
MMM/d/yyyy
Short Caption Format
dd
Culture
(Default culture)
Add new culture

You can repeat the above for other OLAP hierarchy levels.

Conversion

Expand the **Conversion** section to customize the OLAP hierarchy member property that needs to be converted and add date format strings.



The first step is to choose the **Converted Member Property** from these options:

- Value
- Name
- Unique Name
- Caption
- Custom Property (described under the sub-section below)

Next, add one or more **Format Strings** for the conversion. These formats should match the values of the member property selected above.

You can use the **Default Format Strings** drop-down to select a pre-defined date format, and then click the **Add** (plus sign) button to add it to the list of format strings.

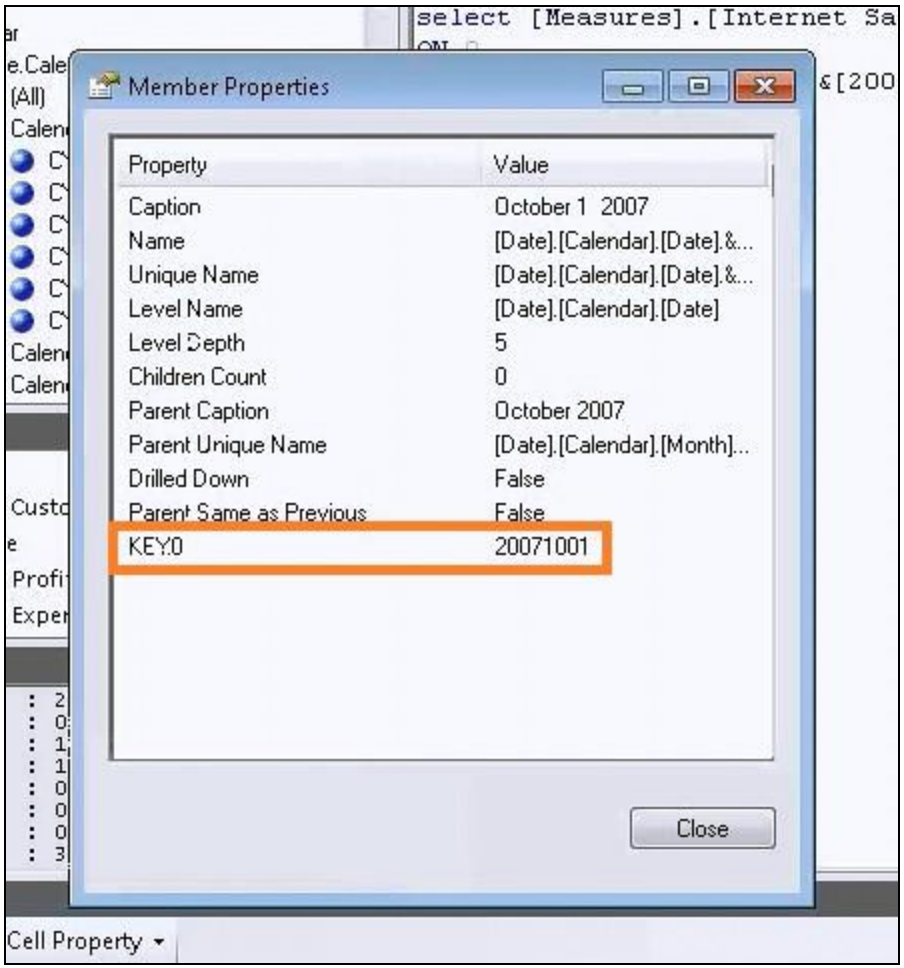
Alternatively, enter a **Custom Format String** and then click the **Add** (plus sign) button to add it to the list.

If the format strings should be used with a particular culture when parsing the member values into dates, select that **Culture**.

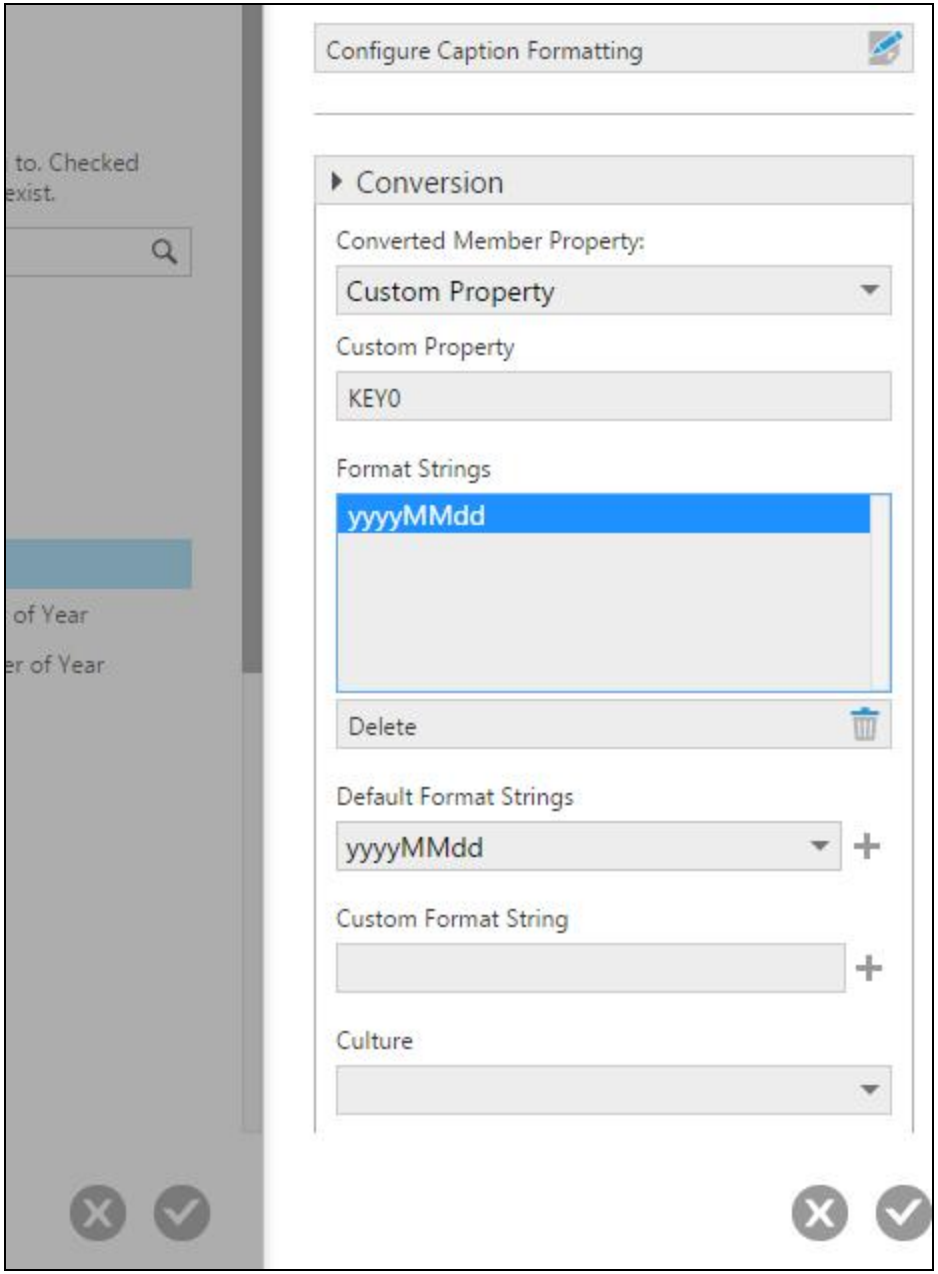
Using a Custom Property For Conversion

The *Custom Property* option for the Converted Member Property setting lets you specify a different property such as the 'key' of the member.

For example, the following dialog from SQL Server Analysis Services shows there is a member property named *KEY0* with a value *20071001* which indicates the corresponding date format string should be *yyyyMMdd*.



Returning to the *Conversion* section in Symphony, set the **Custom Property** to *KEY0* and then add *yyyyMMdd* as a format string.

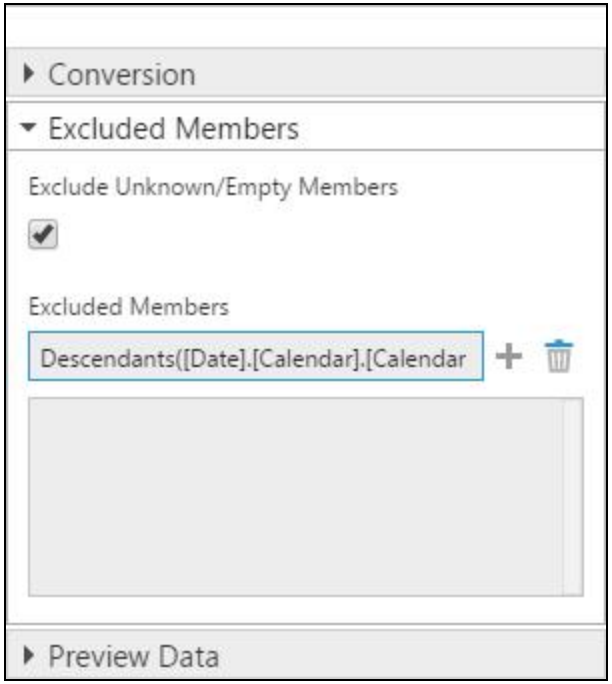

Configure Caption Formatting
to. Checked
exist.
of Year
er of Year
Conversion
Converted Member Property:
Custom Property
Custom Property
KEY0
Format Strings
yyyyMMdd
Delete
Default Format Strings
yyyyMMdd
Custom Format String
Culture

Excluded Members

If your OLAP time hierarchy has empty or unknown members, auto-detection of the start date may not work properly. In this case, use the **Excluded Members** section to exclude empty, unknown, or other members from the date mapping.

Expand the Excluded Members section, and then select the **Exclude Unknown/Empty members** checkbox to hide members with empty captions, for example.

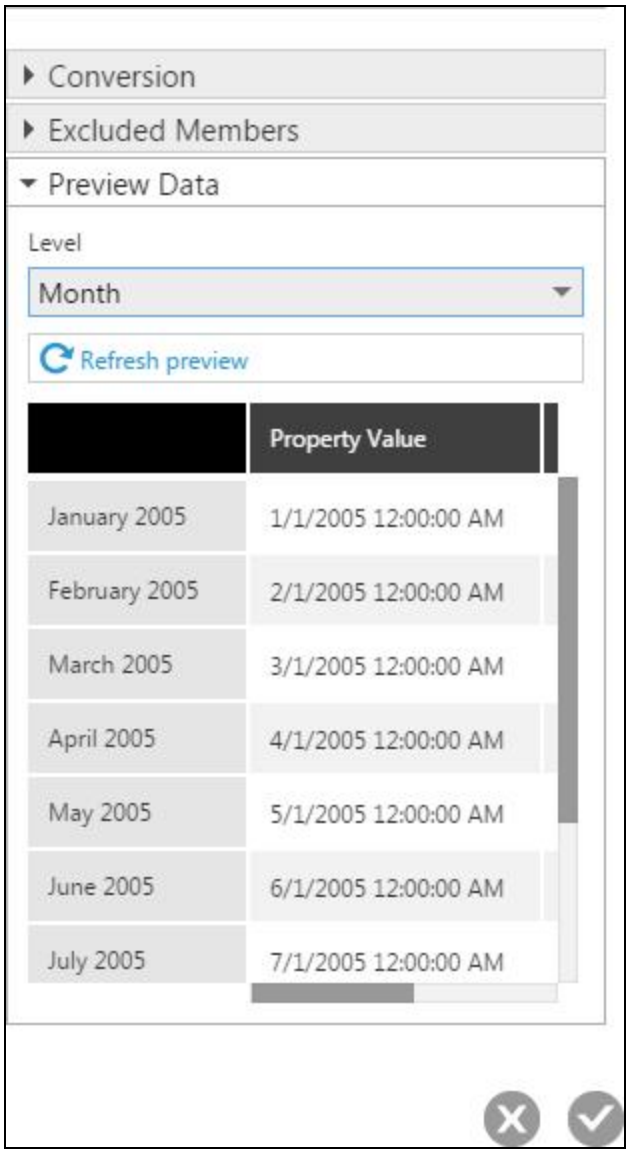
To exclude specific members by name, type a unique name in the text box, and then click the **Add** (plus sign) button to add it to the exclusion list. Instead of a unique name, you can also add an MDX expression.



Preview Data

Use the **Preview Data** section to check the mapping/conversion for each OLAP hierarchy level.

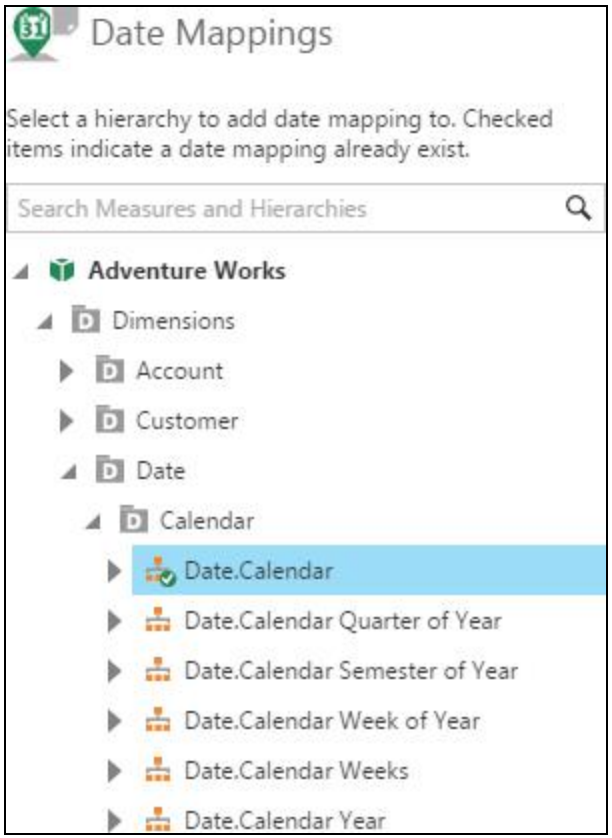
Select the OLAP hierarchy level using the **Level** drop-down list. The table will be updated with the mappings of OLAP property values to target time values.



Submit

Finally, click **Submit** (check mark) to create the date mapping.

Now if you locate your OLAP hierarchy in the *Date Mappings* dialog, you'll see that it has a checkmark icon, which indicates it has a date mapping.



Using a Date Mapping

Once you have a date mapping in place on an OLAP time hierarchy, you can create a new metric set and make use of the new abilities. For example, you can drag the hierarchy from the cube to your dashboard and then add a *Calendar* filter to operate on that hierarchy.

For more information, see:

- Video: Introduction to OLAP Date Mapping
- Microsoft Docs: Standard Date and Time Format Strings
- Microsoft Docs: Custom Date and Time Format Strings

Database Write Back

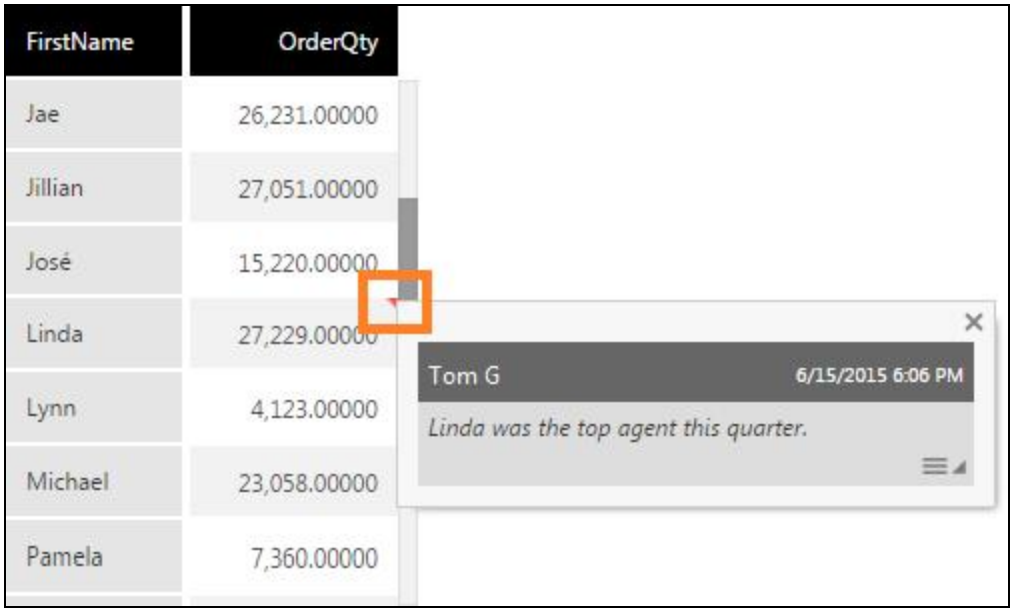
This applies to: *Managed Dashboards, Managed Reports*

Symphony includes built-in data input features for storing data entered by end users in your warehouse data storage area, as well as notes functionality to allow users to add and reply to notes tagged to data points.

In some scenarios, there may be a specific requirement to store input data outside of Symphony instead. The following article outlines an example of how you can use Symphony to write data to your own database.

Write Back Sample

Symphony allows users to provide more context or communicate with other users viewing the same data cube data by adding notes or annotations to a data point.



Instead of viewing notes by hovering over the red triangle like in the image above, we will allow users to view or add live comments in another column of the table visualization and store them in our own database.

Budget	Overall	Project	LatestUpdate
3	2	Project Chatcube	System Administrator: Tue Apr 26 2016 15:48:37 GMT-0400 (Eastern D... This is a test comment
1	2	Project Dabpulse	System Administrator: Tue May 17 2016 17:54:25 GMT-0400 (Eastern D...
2	1	Project Edgelist	..
1	2	Project Twitterbird	..
1	1	Project Yakimbo	..
	1	Project Yondo	..

Here is an overview of how to write back data from a dashboard:

- Define a procedure or write a manual query that will capture and write back to the data to a database (and also return some data).
- Create a data cube using that procedure or manual query that will capture and write back the data to a database using public parameters passed from the dashboard.
- Design a dashboard that will provide the user option to input the information to write back using different components or filters, such as a drop down list or textbox.
- Connect view parameters and/or filters to the data cube parameters to pass them the captured information for executing the procedure or query and writing back to the database.

Set Up the Data

The sample setup below uses SQL Server as the data source.

In this case, a new column in the SQL database table is used to store the comments entered by users. A placeholder value for the comments such as '..' allows it to be intentionally shown as empty for this example.

Results | Messages

	Project	Sponsor	PM	Budget	Schedule	Scope	Overall	Progress	LatestUpdate
1	Project Chatcube	Strief	Erinn Schroepfer	3	1	1	2	3	..
2	Project Dabpulse	Luchsinger	Marlon Allam	1	1	1	2	1	..
3	Project Edgelist	Strief	Roman Budin	2	1	1	1	2	..
4	Project Twitterbird	Rohan Phillips	Roman Budin	1	2	1	2	1	..
5	Project Yakimbo	Nyholm	Erinn Schroepfer	1	1	1	1	1	..
6	Project Yondo	Nyholm	Roman Budin	NULL	NULL	NULL	1	NULL	..

Create the Stored Procedure

Create a stored procedure that receives two parameters: the comment column (LatestUpdate) and a unique ID such as the Project name in the sample data, which identifies one specific row to be updated in the database.

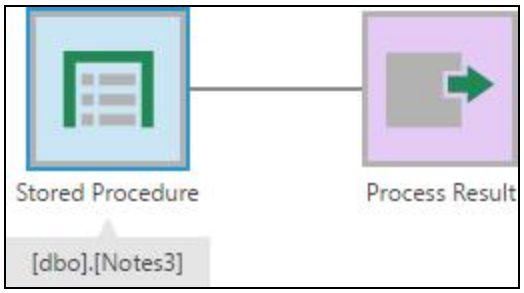
This example also returns the updated data from the table to be displayed using the same stored procedure.

```
CREATE PROCEDURE [dbo].[Notes]
	-- Add the parameters for the stored procedure here
	@project nvarchar(50),
	@comment nvarchar(MAX)
AS
BEGIN
	-- SET NOCOUNT ON added to prevent extra result sets from
	-- interfering with SELECT statements.
	SET NOCOUNT ON;

	UPDATE [dbo].[Table_1]
	SET LatestUpdate = @comment
	WHERE Project = @project
	SELECT * from [dbo].[Table_1]

END
```

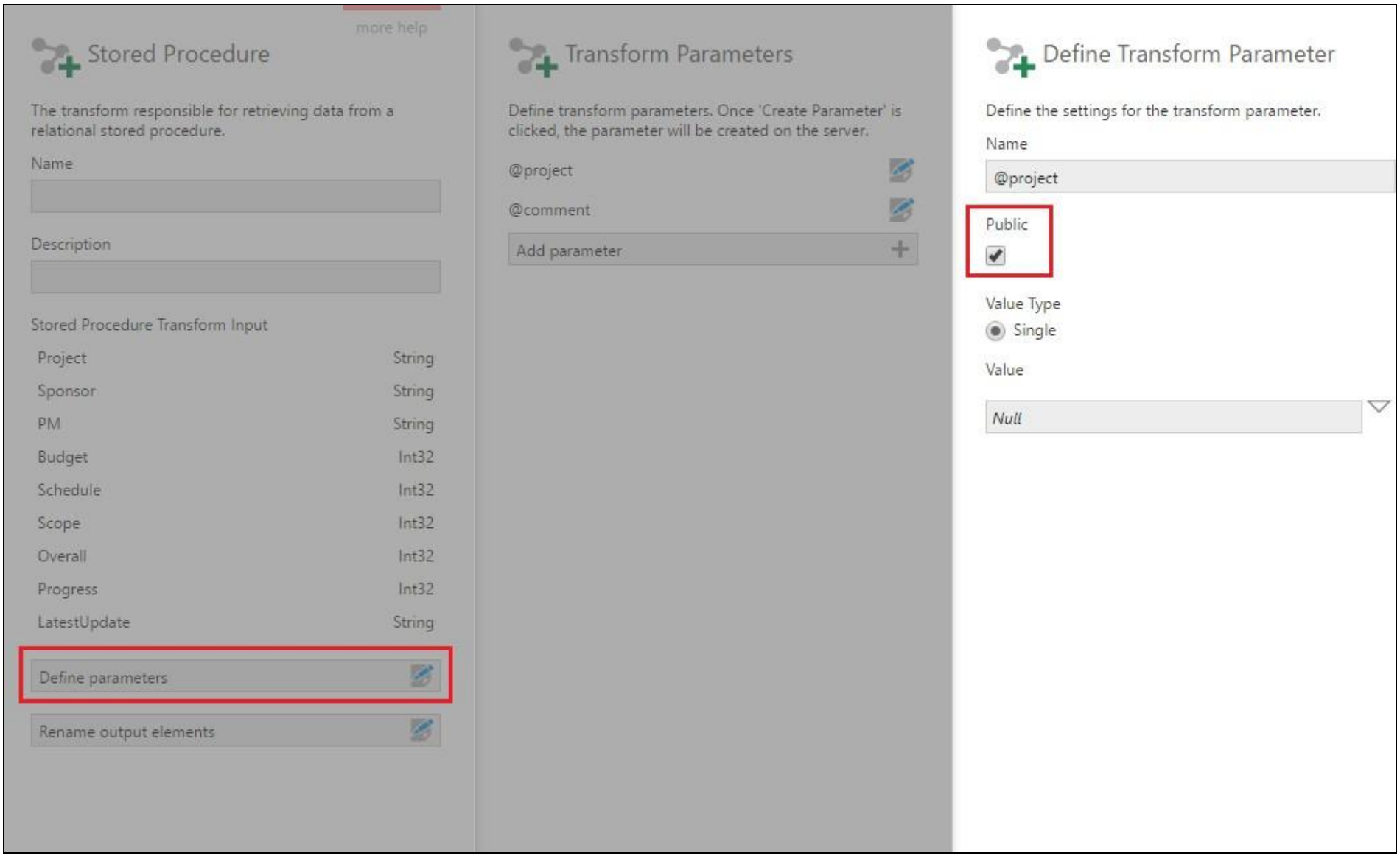
Important: There are various ways your stored procedure could get called again with the same parameter values when used on a dashboard. An ID that can uniquely identify each update can prevent unwanted duplicates.

Create a Data Cube

Create a new data connector with the database table modified above. Create a new data cube using the stored procedure or query.



Configure the Stored Procedure transform, or the manual query's placeholder parameters, and expose the parameters as Public so they can be used outside of the data cube.



Design the Dashboard

Drag and drop the data cube on the dashboard canvas. The data appears as a table visualization.

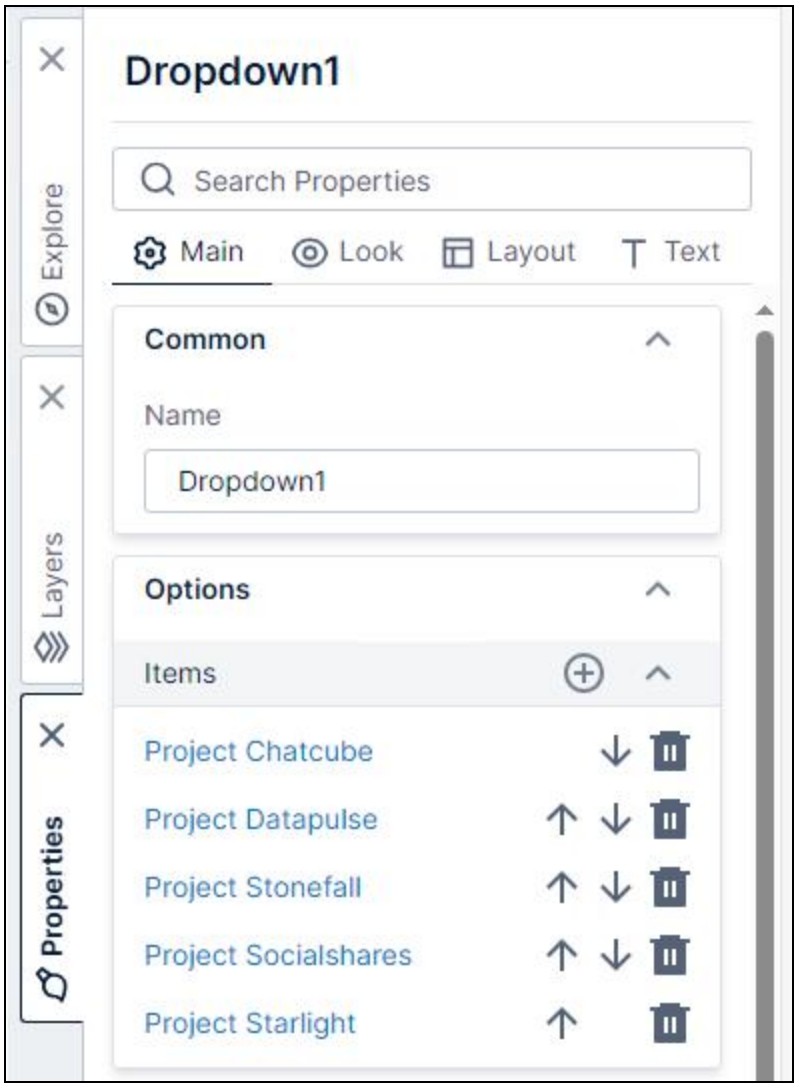
In our example, which returns all of the updated table data from the same stored procedure, continue adding the rest of the data to be displayed as columns in the table.

To prevent old data from being returned if the same parameter values are passed later but are meant to execute the procedure/query each time, you can bypass the data cache on this metric set.

Add a Drop Down List

This example uses a drop down list to pass parameter values, although you can use other controls instead.

1. From **Components** in the toolbar, select **Drop Down List**, as one example of how users can choose a value to pass as a parameter.
2. With the drop down list selected, open the **Properties** window.
3. In the **Main** tab under **Items**, you'll see that the drop down list is configured with a single item by default. Click the item to modify it.
4. In the **Main** tab, set the `Value` property to the name of the **Project** as it is in the database. In the **Text** tab, set the `Caption` property to the same.
5. Add new items and repeat these steps for other values to select.



Add a Textbox

With nothing selected on the canvas, go the toolbar, and from **Filters** select **Textbox**.

If the filter was automatically connected, disconnect it and open the **Properties** window to change its settings:

- Check the `Manual Items` property.
- In the **Look** tab, check `Hide Token Menu`.
- In the **Layout** tab, check `Multi-Line`.
- In the **Text** tab, clear the label and tooltip properties.

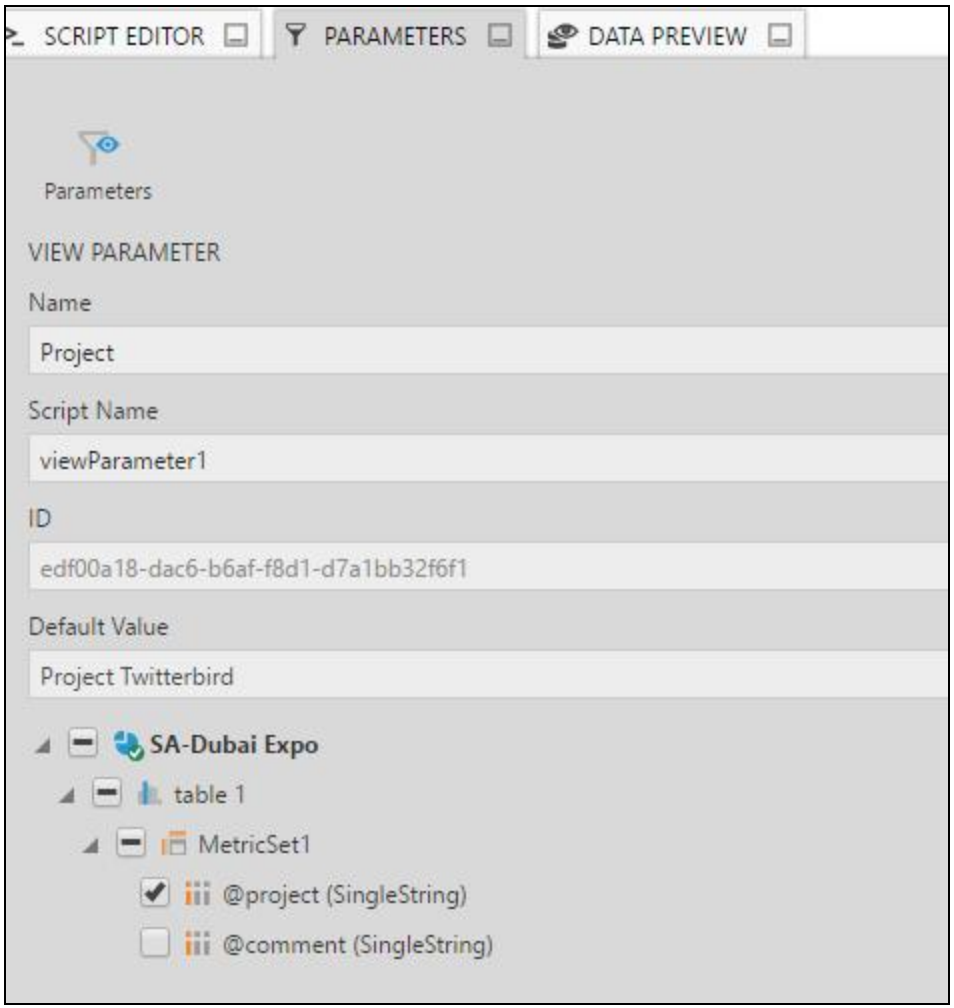
(If the textbox displays a value from when it was connected to data, it will instead display the **value** from its Value property the next time the dashboard is opened.)

Add View Parameters

Open the **Parameters** window, and select **Add New** to add a new view parameter.

Rename this view parameter to something meaningful for ease of use, such as **Project**.

Assign this to the `@project` parameter defined in the stored procedure.



Click the **Parameters** button to return to the list of All View Parameters.

Add another view parameter and assign it to the `@comment` parameter defined in the stored procedure.

Add the Script

This example will use a button with script to pass the values to the parameters, but it's also possible to connect filters to data cube parameters and use an update button.

In the toolbar, select **Components** and choose **Button**.

In the **Click** event of the button, add the following script in the script editor, and update it to refer to the **Script Name** you chose for each of your view parameters, etc.

```
var view = this.parentView.control;
var vpProject = view.getViewParameterByName("project");

vpProject.parameterValue.clearTokens();
vpProject.parameterValue.clearValues();

// Populate the @project parameter from the drop down list.
vpProject.parameterValue.value = dropDownList1.value;
vpProject.invalidateParameterValueLastModifiedTime();

var vpComment = view.getViewParameterByName("comment");

// Access the current logged in user name - optional.
var currentLogin = dundas.context.currentSession.accountDisplayName;

vpComment.parameterValue.clearTokens();
vpComment.parameterValue.clearValues();

// Populate the @comment parameter from the textbox.
vpComment.parameterValue.value = currentLogin + ": " + parameterTextBox1.control.value;
vpComment.invalidateParameterValueLastModifiedTime();

//refresh the table control to view the comments in the table.
table1.loadData();
```

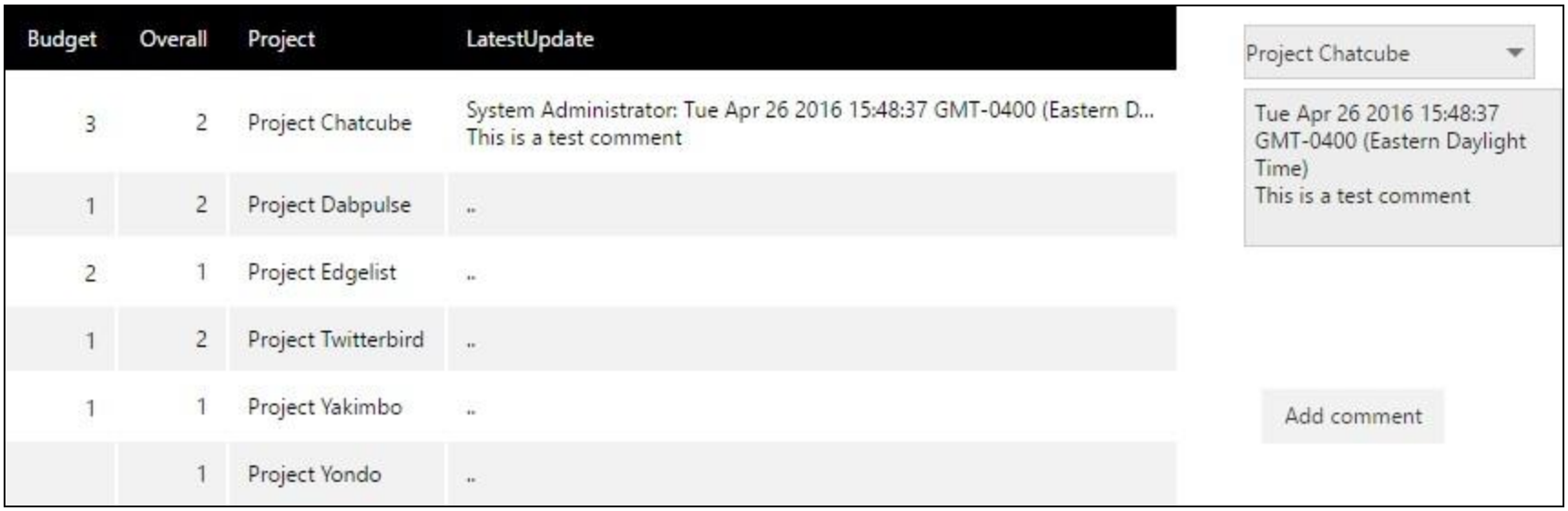
Test the Dashboard

Click **Sandbox View** in the toolbar.

Choose a project from the drop down list.

Add a comment in the textbox and click on the button.

The comment is added to the table:



For more information, see:

- JavaScript API
- Using the Script Editor
- Edit Versus View Mode and Sandbox View
- Modify a Filter / View Parameter Using Scripting
- Passing parameter value from dashboard to stored procedure
- Setting Up Data Input

Expand Or Collapse a Template Grid Cell

This applies to: *Managed Dashboards, Managed Reports*

When using a template grid for resizing, you can choose to add a component to a cell such as button, and a *Toggle Template Cell* interaction to expand & collapse the cell. When collapsed, the cell is reduced to the size of the component, and all other contents will be hidden.

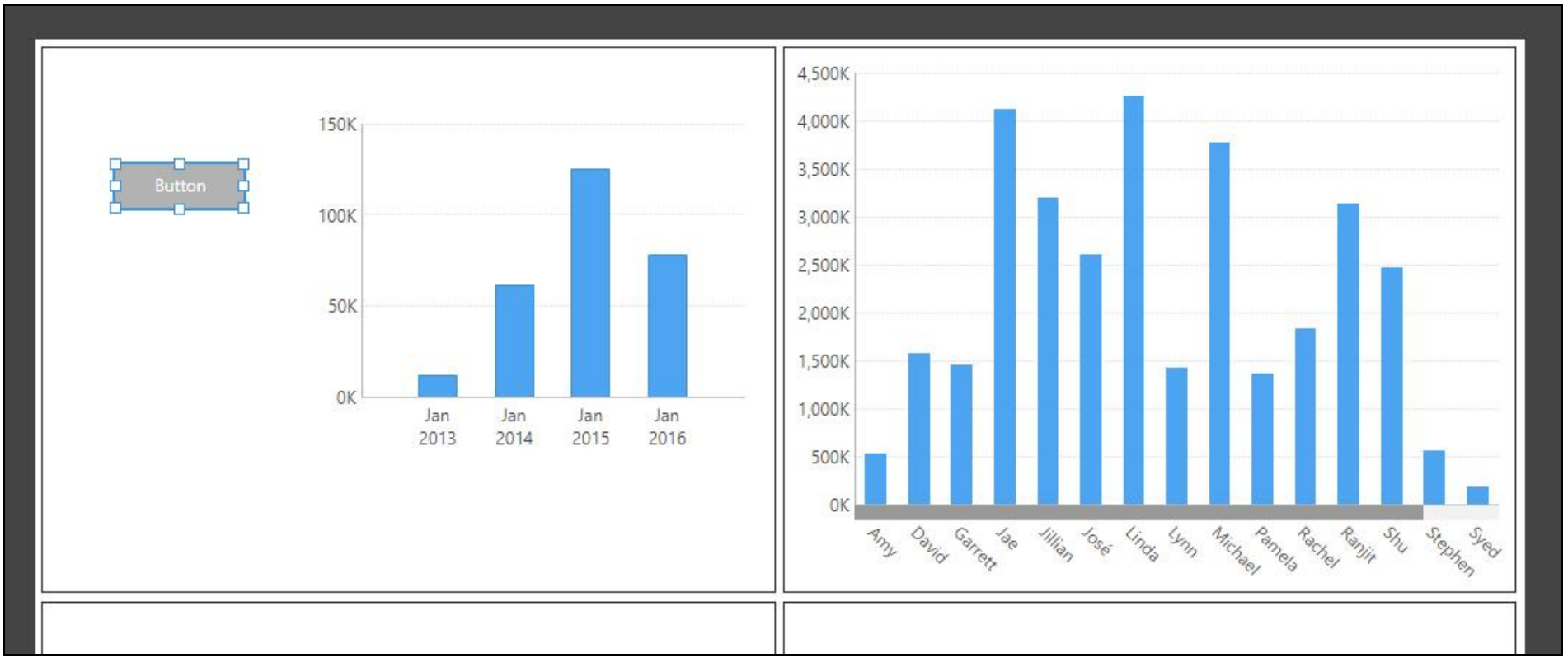
How to set up a button to expand or collapse a template grid cell is shown as an example below.

Set Up the Template Grid

This example starts from a *new blank* dashboard: choose **Define Template Grid**, and add a 2x2 grid.

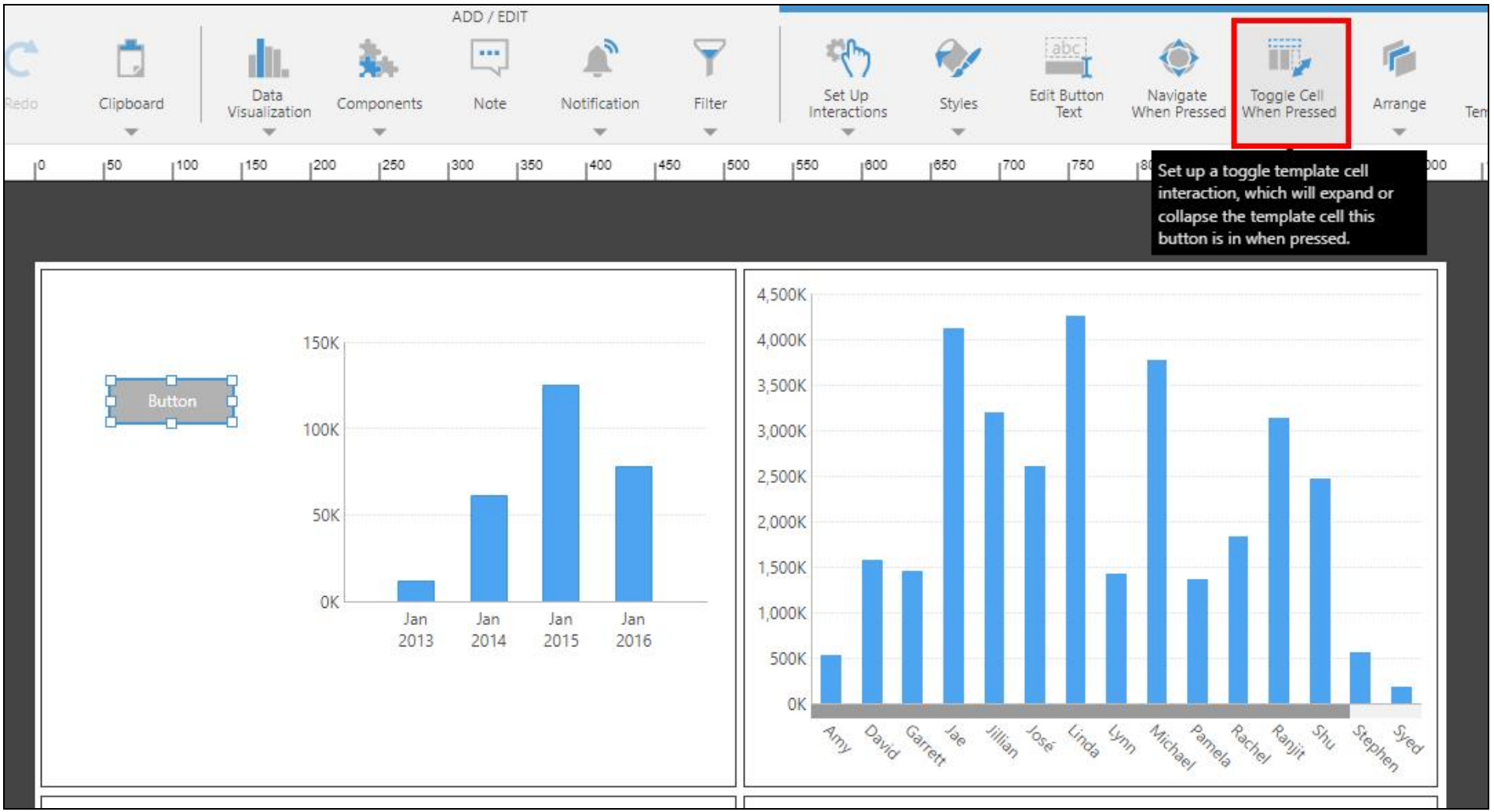
Also set the **Re-Size Mode** property of the dashboard to Resize.

Add a button and a small chart to the first cell. Then add a second chart that fills up the second cell.

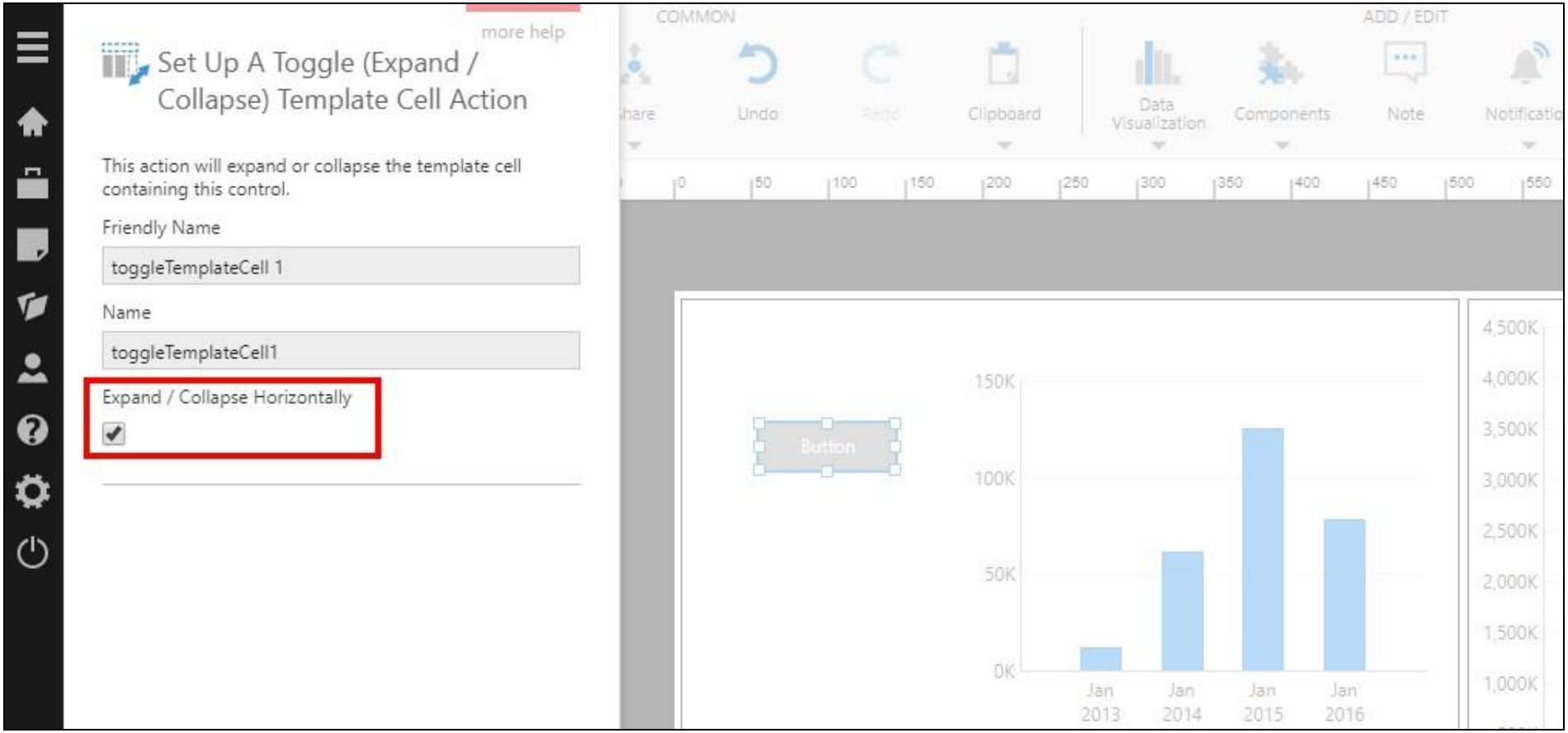


Set Up the Interaction

Select the button and choose **Toggle Cell When Pressed** in the toolbar.

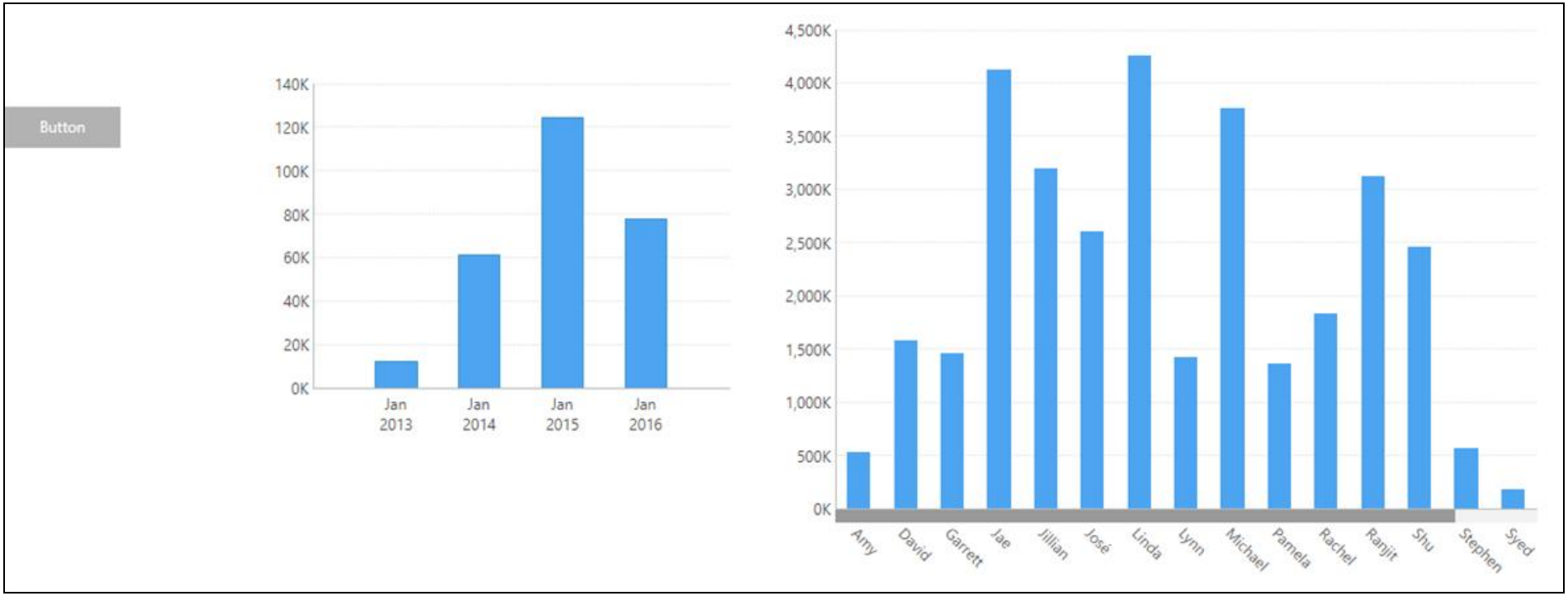


In the dialog, select the option to **Expand / Collapse Horizontally**. This option forces the expand/collapse to work horizontally only instead of in both directions (vertical and horizontal).

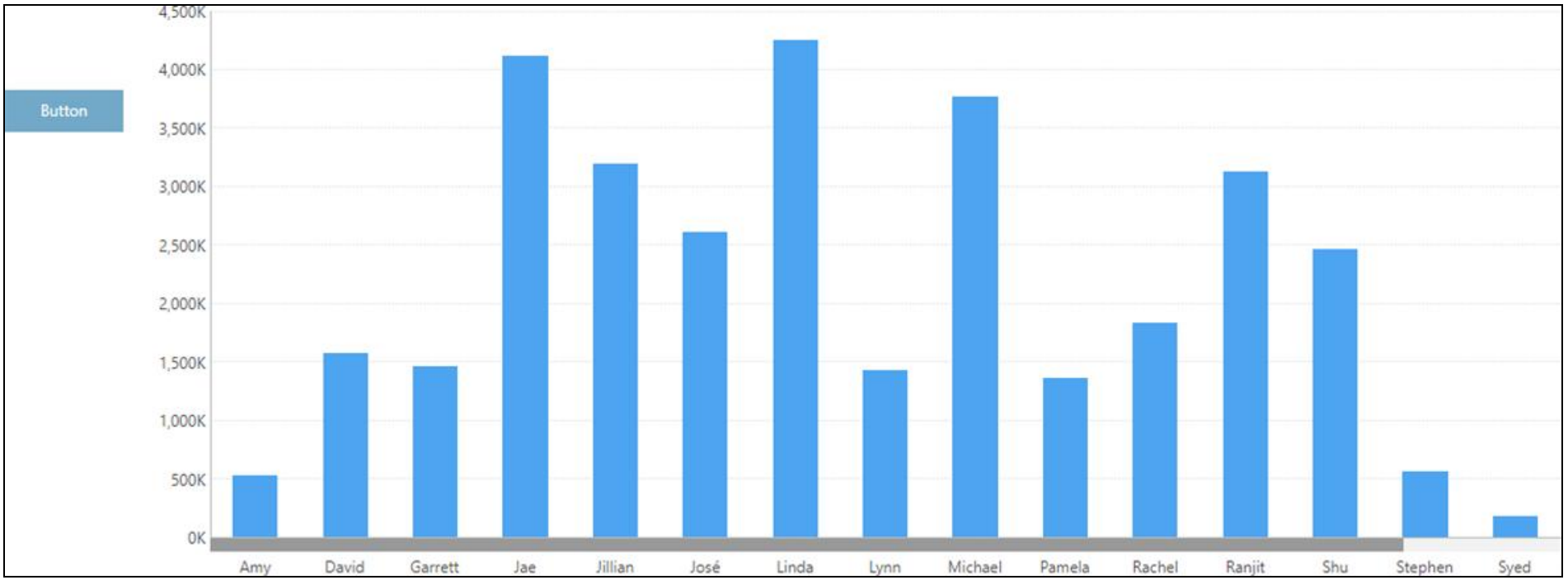


View the Dashboard

Click **Sandbox View** to test the dashboard.



Click the button to collapse the first cell horizontally. Observe that the first chart is now hidden and the second chart expands to fill up the extra space.



Click the button again to expand the first cell.

If you view the dashboard normally to test, be aware that if you switch to **Edit** mode while the cell is collapsed, it will remain and be saved as collapsed.

For more information, see:

- Using a Template Grid for Resizing
- Edit Versus View Mode and Sandbox View
- Design Overview

Navigate to URL

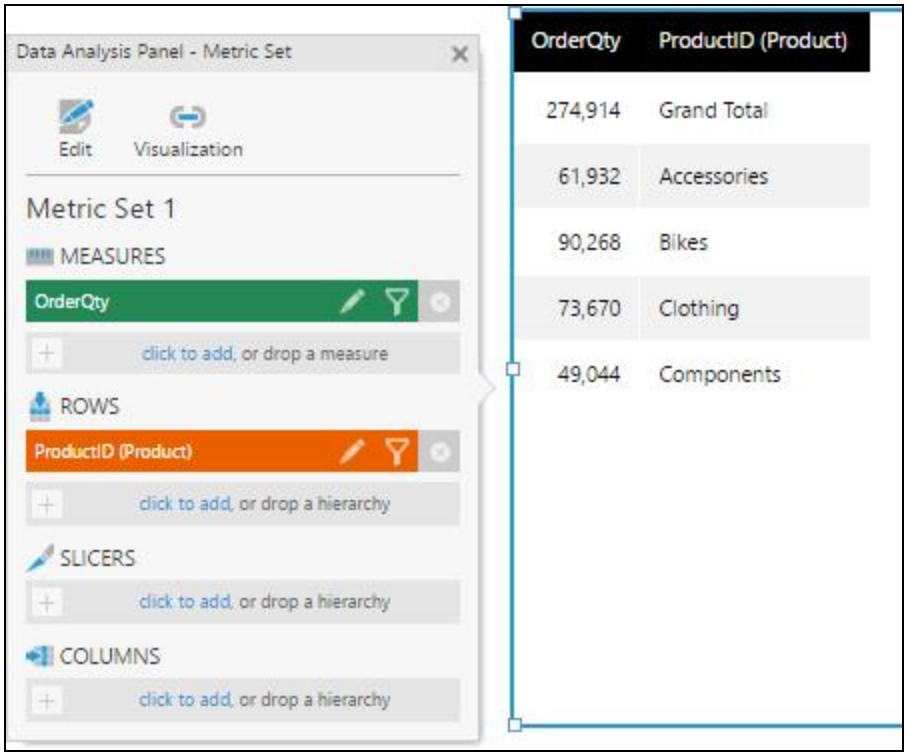
This applies to: *Managed Dashboards, Managed Reports*

This article shows you how to set up a navigate interaction so that when a user clicks on a table cell, a URL is opened and the caption of the cell is passed in the URL as part of a query string (or as the URL itself).

You can also follow similar steps when using drill down or hover over interactions, which can be set to show *A URL*.

Create a Dashboard

For this example, create a dashboard with a table visualization that displays two columns.



Set Up the Navigate Interaction

With the visualization selected, click **Set Up Interactions** then **Navigate** in the toolbar.

In the setup dialog, set the **Bound Visual** to *Column 2*. This is so that the interaction fires only when a *Product* cell is clicked.

Set the **Navigate** drop-down to *To a URL*. Then change **Open** to *In a new window* if preferred.

In the **URL** field, enter text that includes a placeholder of your choice representing whichever value was clicked. For example, the following URL passes a product value represented by the placeholder text *varProduct* into a Google search: http://www.google.com/search?q=varProduct

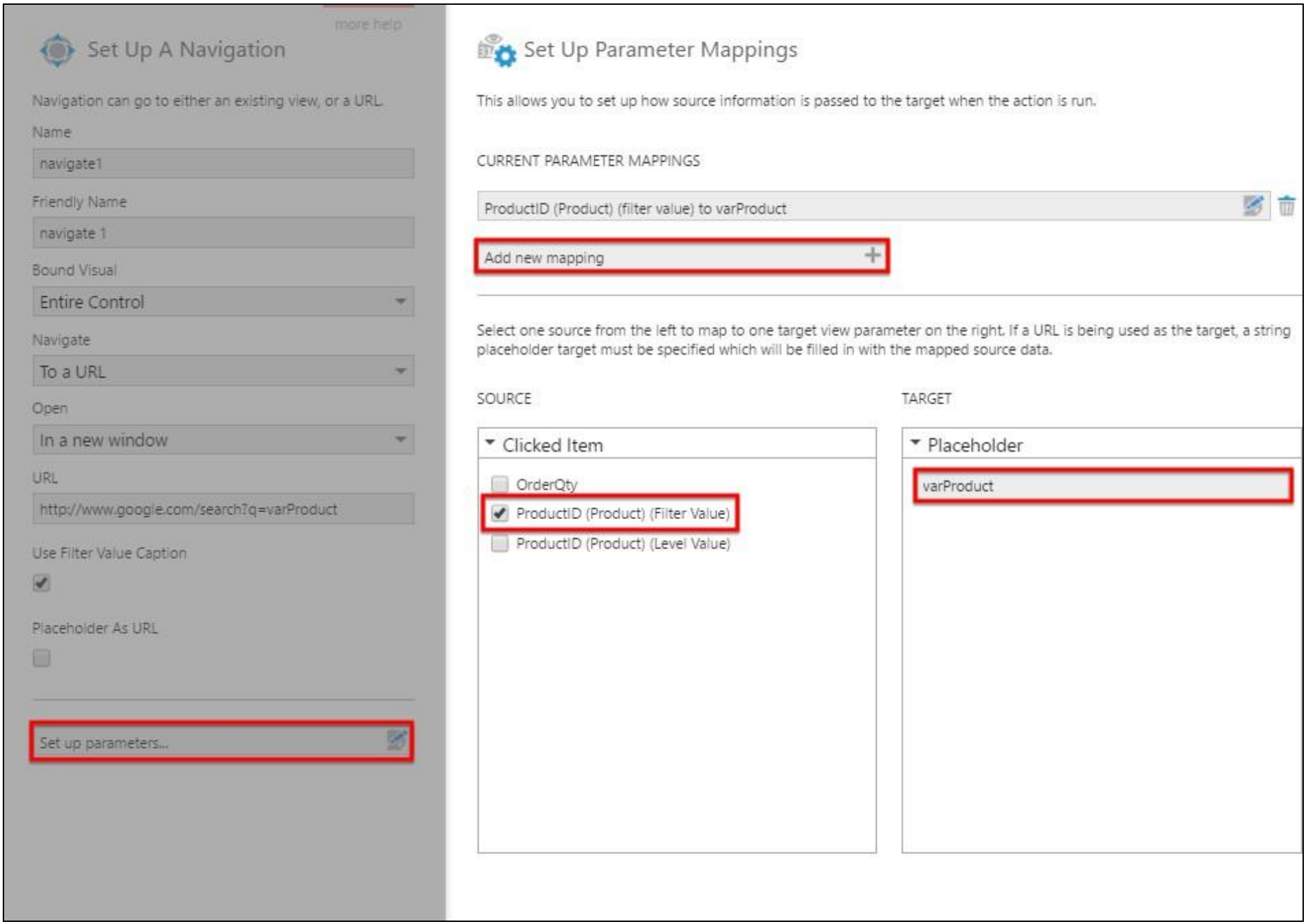
Filter Value Caption

Select the **Use Filter Value Caption** checkbox to pass the caption of the filter value into the URL instead of the filter value (or unique name). A caption in this case refers to the text you see in a table cell, such as *Bikes*.

Next, click **Set up parameters**. In the parameter mappings setup dialog, click **Add new mapping**.

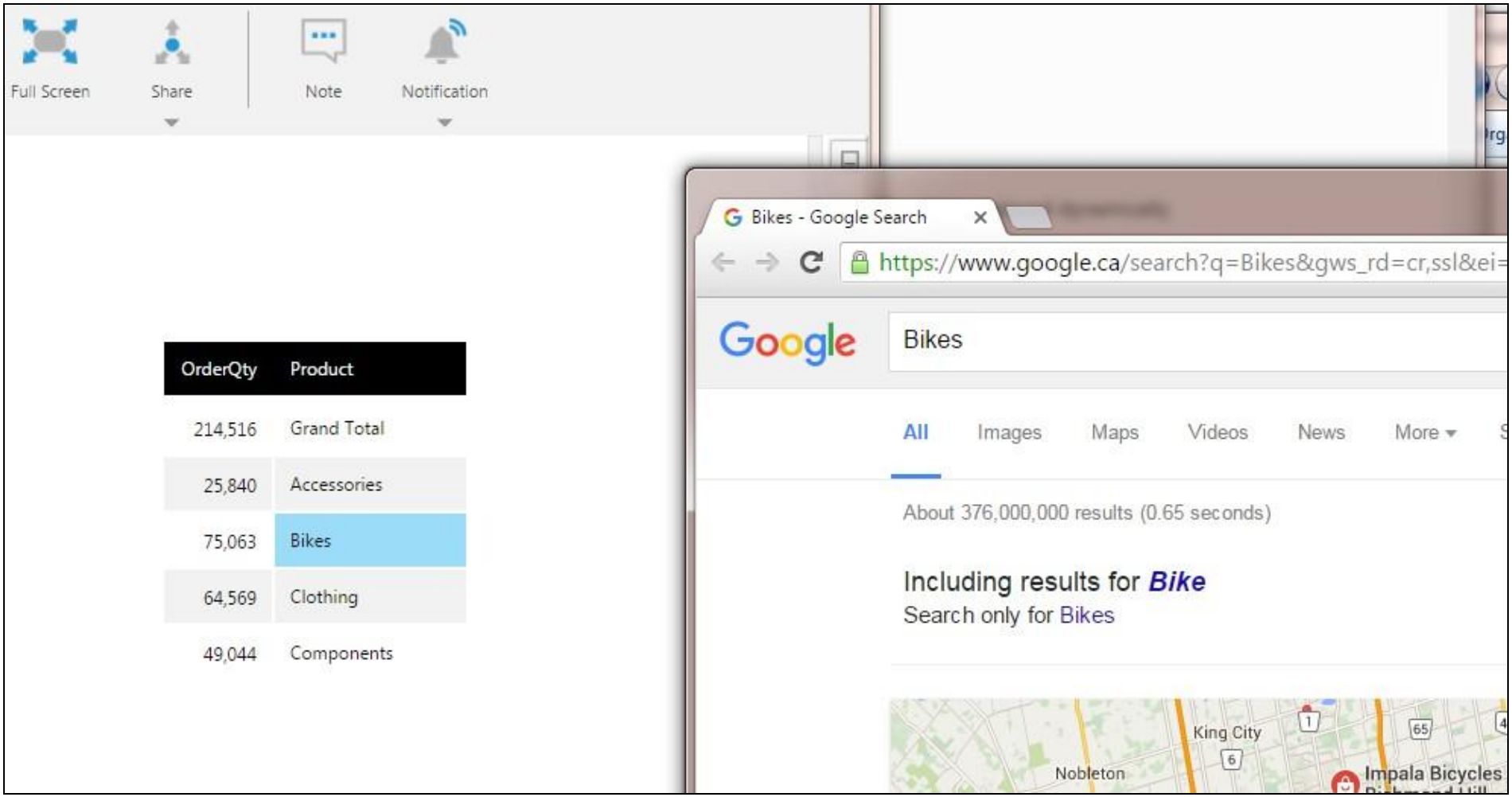
In the **Source** list, select *Product (Filter Value)*.

In the **Target** list, type your placeholder text. In our example above, this was *varProduct*. At viewing time, this placeholder text in the URL will be replaced dynamically with the caption of the clicked table cell.


more help
Set Up A Navigation
Navigation can go to either an existing view, or a URL
Name
navigate1
Friendly Name
navigate 1
Bound Visual
Entire Control
Navigate
To a URL
Open
In a new window
URL
http://www.google.com/search?q=varProduct
Use Filter Value Caption
Placeholder As URL
Set up parameters...
Set Up Parameter Mappings
This allows you to set up how source information is passed to the target when the action is run.
CURRENT PARAMETER MAPPINGS
ProductID (Product) (filter value) to varProduct
Add new mapping
Select one source from the left to map to one target view parameter on the right. If a URL is being used as the target, a string placeholder target must be specified which will be filled in with the mapped source data.
SOURCE
TARGET
Clicked Item
OrderQty
ProductID (Product) (Filter Value)
ProductID (Product) (Level Value)
Placeholder
varProduct

Switch to **View** mode and test the interaction by clicking on a cell in the Product column.

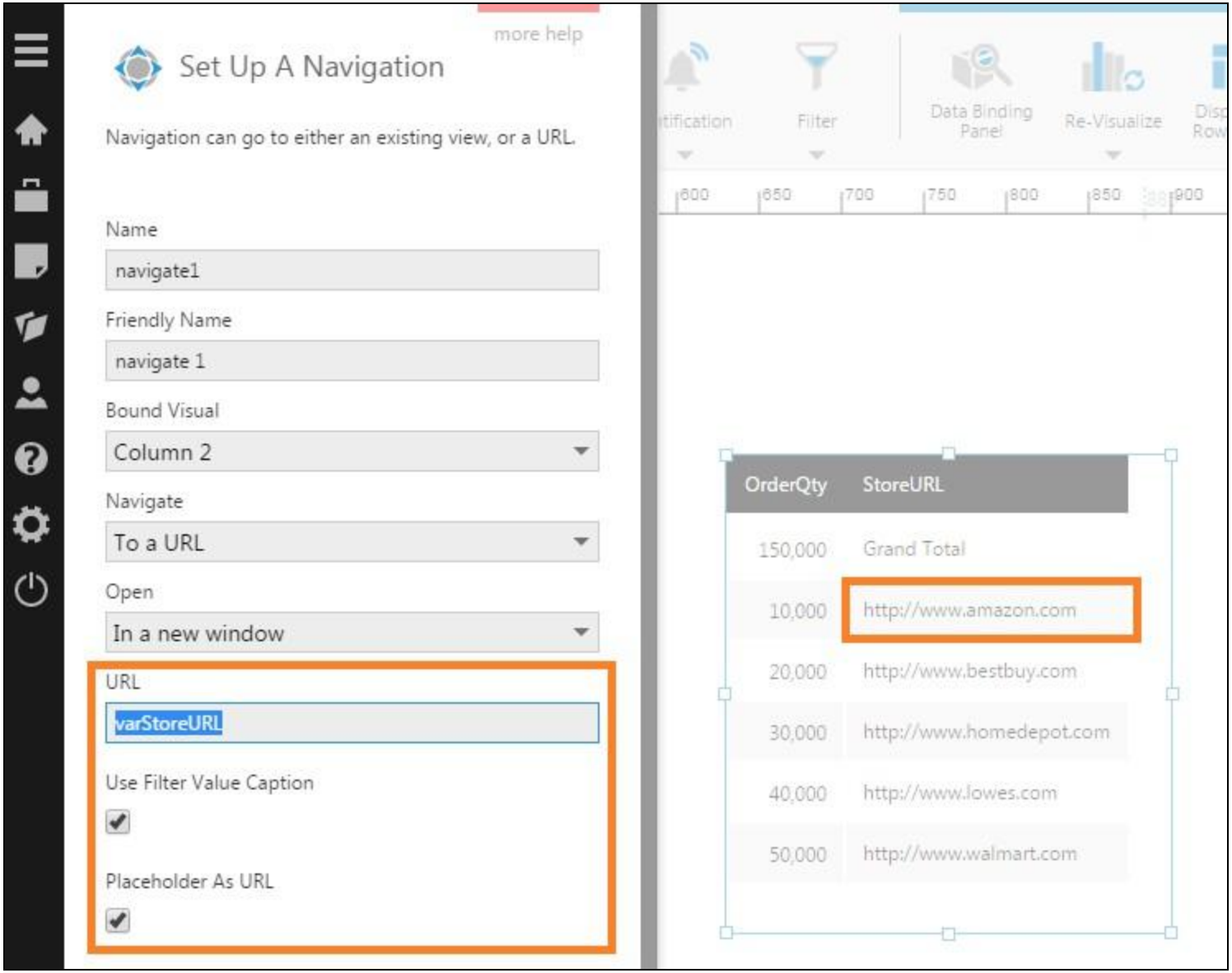
In our example, a new browser window or tab should be opened with the results of a Google search corresponding to the clicked cell.



Placeholder as URL

What if your filter value caption or unique name (i.e., your data) contains actual or partial URL values? In this case, the navigation will fail because the placeholder value will be encoded as part of the eventual URL.

To prevent this encoding, select the **Placeholder As URL** checkbox (usually in addition to Filter Value Caption).



If your data contains full URLs, just enter the name of your placeholder in the **URL** field.

If your data contains partial URLs, for example without the HTTP/HTTPS prefix, enter the missing portions into the **URL** field. For example: `https://myPlaceholder`.

For more information, see:

- Set Up a Navigate Interaction
- Set Up a Hover Over Interaction
- Set Up a Drill Down Interaction

Set Up a Data Input Interaction

This applies to: *Managed Dashboards, Managed Reports*

A data input interaction lets viewers of dashboards or other views append data to the Data Input transform in a data cube. This article walks through an example of how you can set up this type of interaction.

Data can be entered by a viewer through a variety of input controls available, which can be combined together into a form for users to fill out. It can also come from an interaction with existing data or from script by using a view parameter.

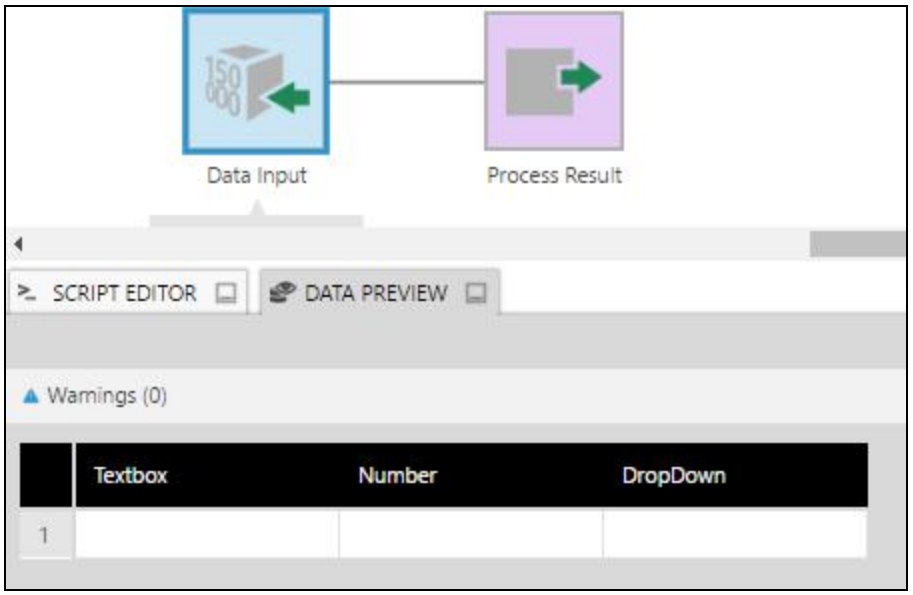
Note: This interaction can't be added directly to a data visualization.

Create a Data Cube

First, create a simple data cube using the Data Input transform.

Add three data input columns:

- *Textbox* (containing the *String* data type)
- *Number* (containing the *Integer* data type)
- *DropDown* (containing the *String* data type)

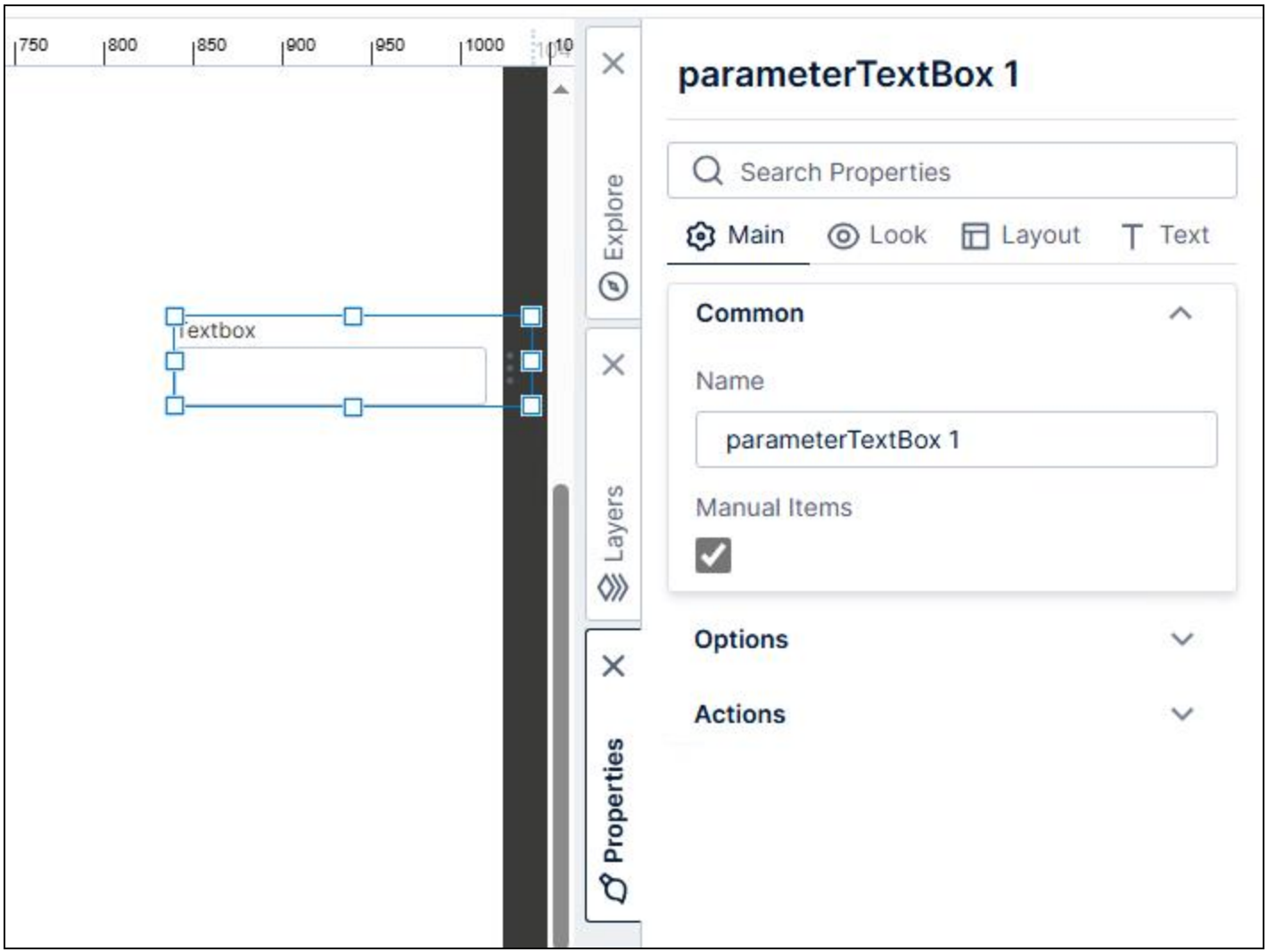


Create a Dashboard

We can create a simple user input form on a dashboard to append data to these columns.

From the toolbar, add a **Textbox** filter:

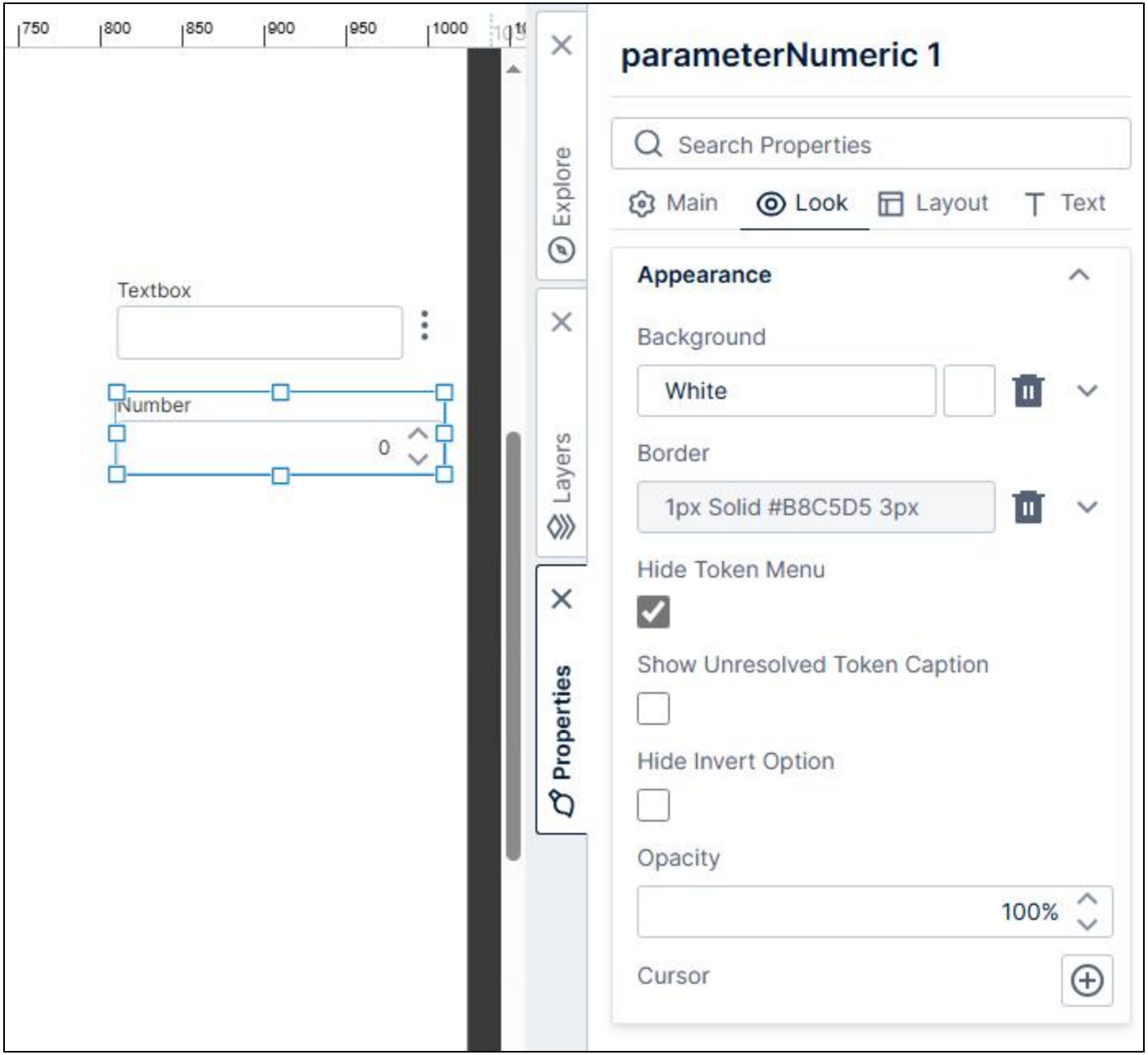
- Change the label text to *Textbox*.
- Enable the **Manual Items** property.
- Select **Hide Token** Menu.



Add a Single *Number* filter:

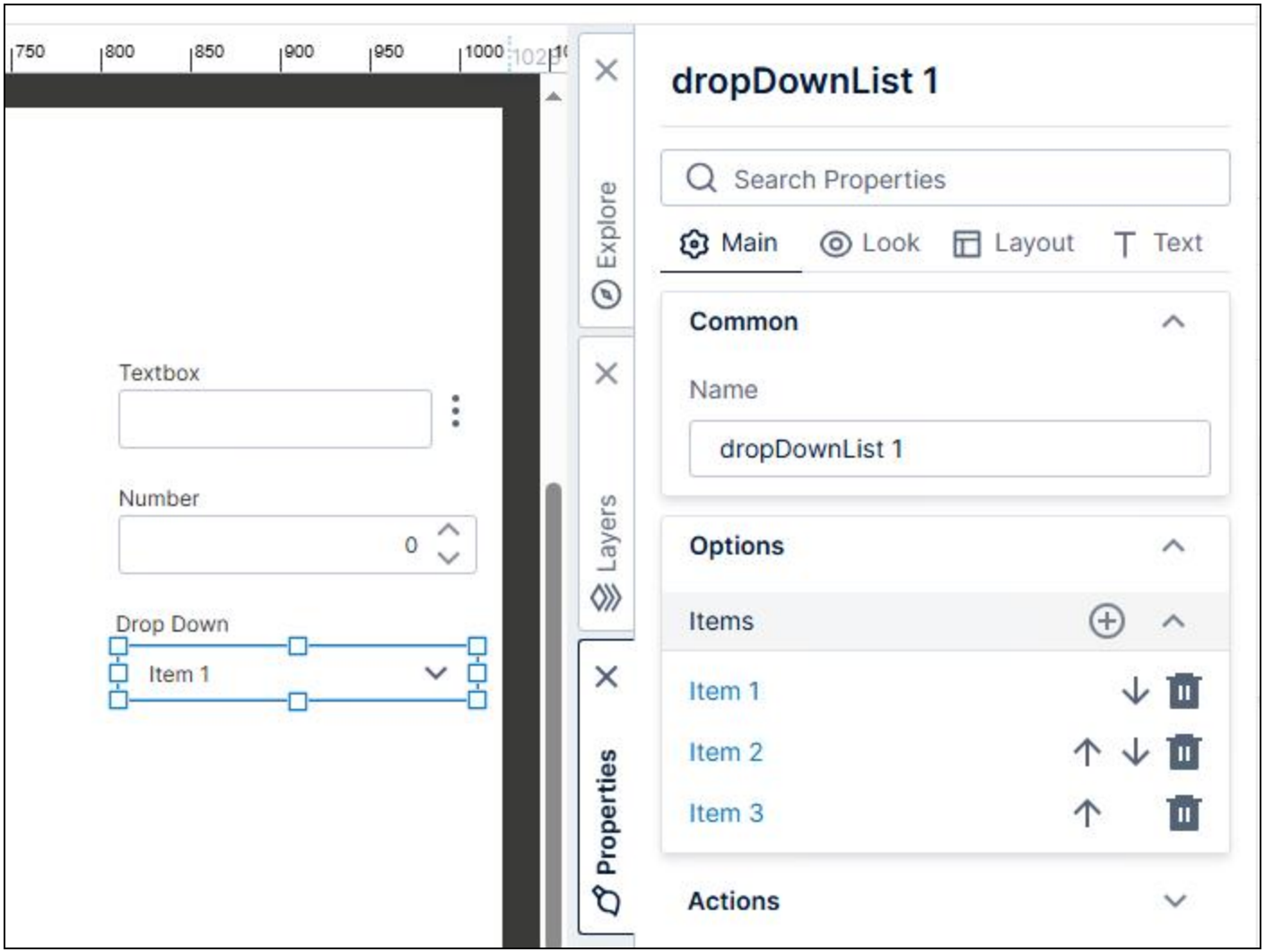
- Change the label text to *Number*.
- Enable the **Manual Items** property.

- Select **Hide Token** Menu.

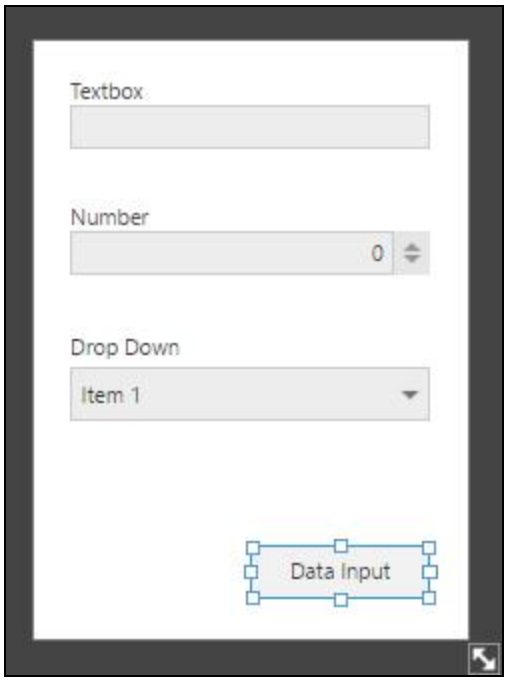


Add a **Drop Down List** component:

- Adjust the **Items** list to have *Item 1, Item 2,* and *Item 3*.
- Set the **Value** of each item to be the same as the caption.
- Add a label component with the text *Drop Down* above.

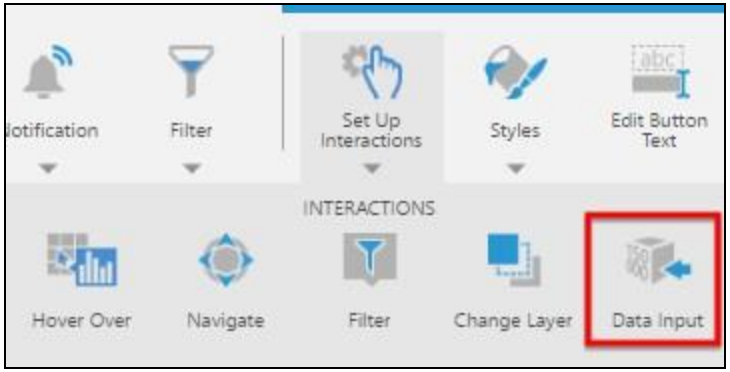


Add a **Button** component and change the button text to *Data Input*.



Set Up an Interaction

With the button selected, click **Set Up Interactions** from the toolbar and select **Data Input**.

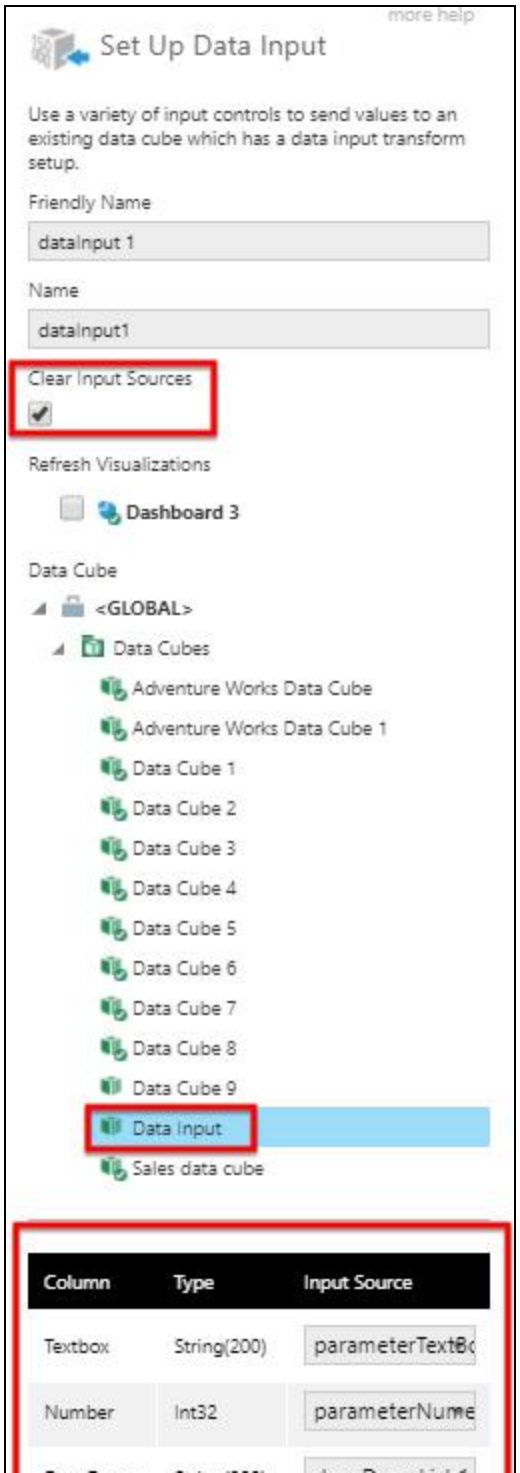


In the *Set Up Data Input* dialog, tick the *Clear Input Sources* checkbox and select the prepared data cube.

For each column in the Data Input transform in the cube, choose an input source. These can be either an input control such as a textbox, or a view parameter.

For our example, set up the following mappings given the default names identifying our input controls:

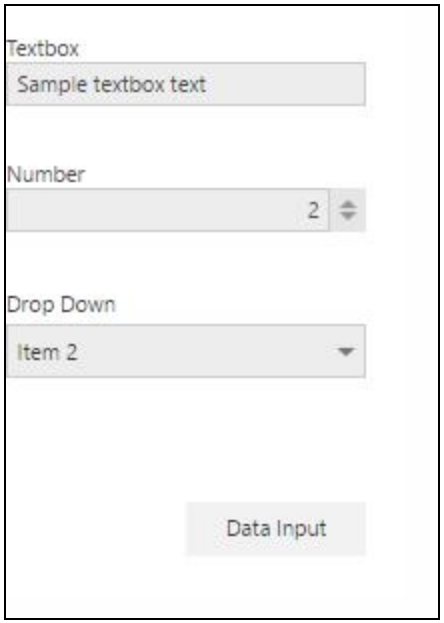
- Textbox – *parameterTextBox 1*
- Number – *parameterNumeric 1*
- DropDown – *dropDownList 1*


more help
Set Up Data Input
Use a variety of input controls to send values to an existing data cube which has a data input transform setup.
Friendly Name
dataInput 1
Name
dataInput1
Clear Input Sources
Refresh Visualizations
Dashboard 3
Data Cube
<GLOBAL>
Data Cubes
Adventure Works Data Cube
Adventure Works Data Cube 1
Data Cube 1
Data Cube 2
Data Cube 3
Data Cube 4
Data Cube 5
Data Cube 6
Data Cube 7
Data Cube 8
Data Cube 9
Data Input
Sales data cube
Column
Type
Input Source
Textbox
String(200)
parameterTextBo
Number
Int32
parameterNume

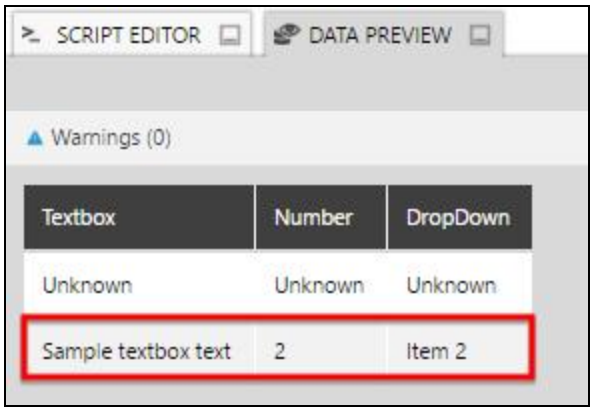
Click **Submit**.

Test the Interaction

Switch to **View** mode, fill out the form, and click the button.



Then, go to the data cube and open the **Data Preview** panel.



- The appended data is always a single value. In the case of a range or collection, the input is the beginning or first value respectively.
- Users require the Data Input privilege to be able to input data.

For more information, see:

- Data Input
- Set Up a Data Input Interaction
- Using Interactions
- Application Privileges

Set Up a Drill Down Interaction

This applies to: *Managed Dashboards, Managed Reports*

A drill down interaction lets you navigate one level down in a hierarchy when you click or tap a data point, and filters to the data points that make up the one you selected. For example, drill down on the year 2012 to view data broken down by each of its months.

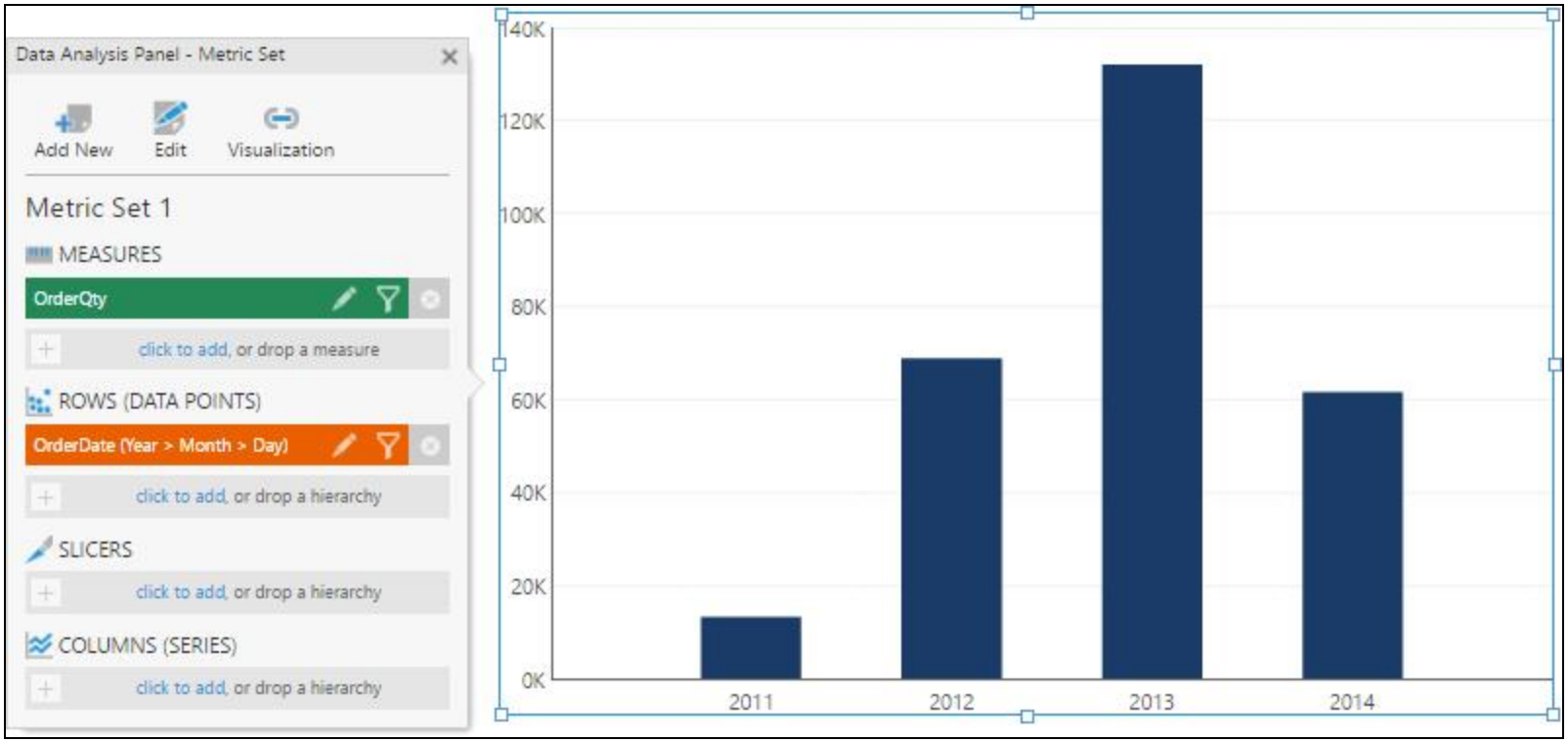
This article shows a basic example of how you can set this type of interaction up to happen on a single click of a data point.

Note: Viewers can also drill down using the context menu without setting up an interaction ahead of time.

Related video: Interactions

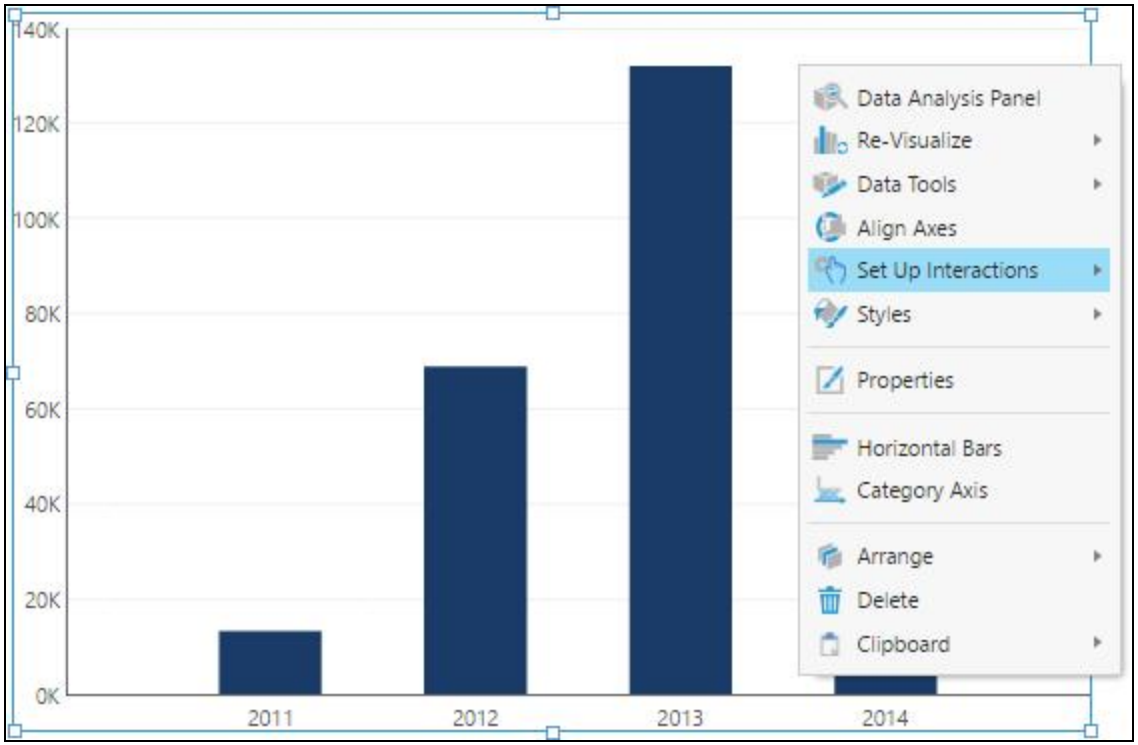
Create a Dashboard

First, create a metric set and add it to a dashboard in order to show *OrderQty by Date* as an example.

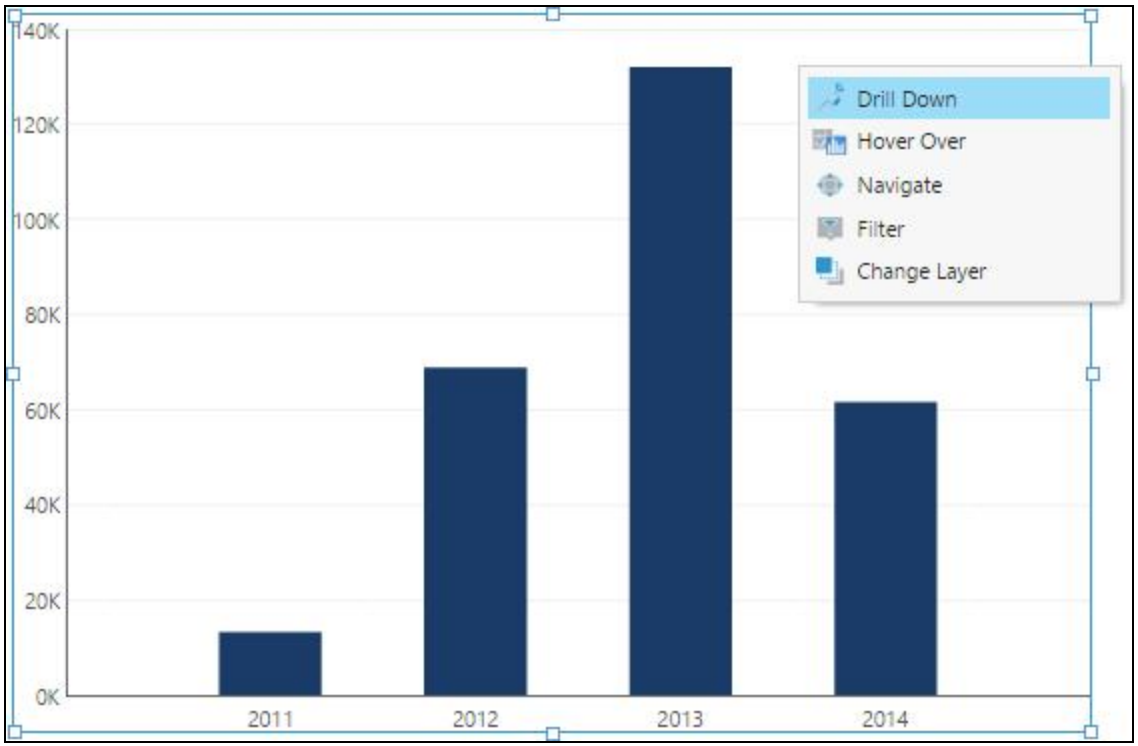


Set Up the Drill Down Interaction

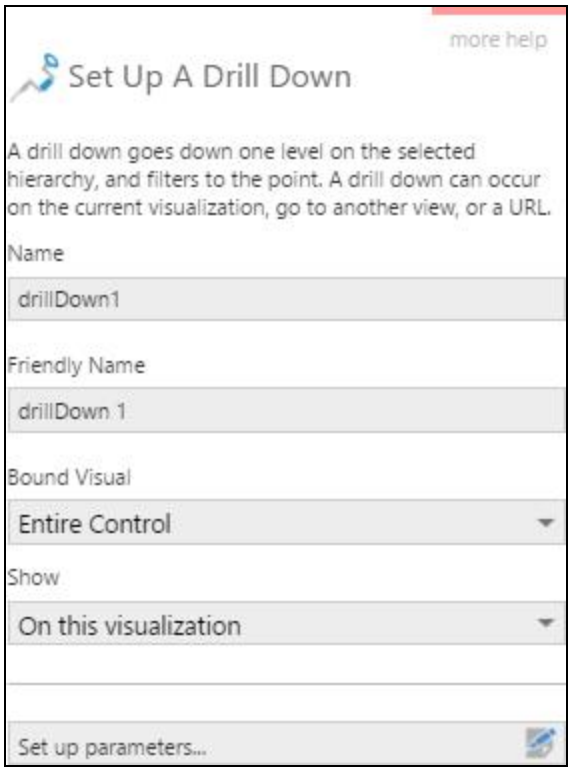
Right-click (or long-tap) over the bar chart on the canvas and select **Set Up Interactions**.



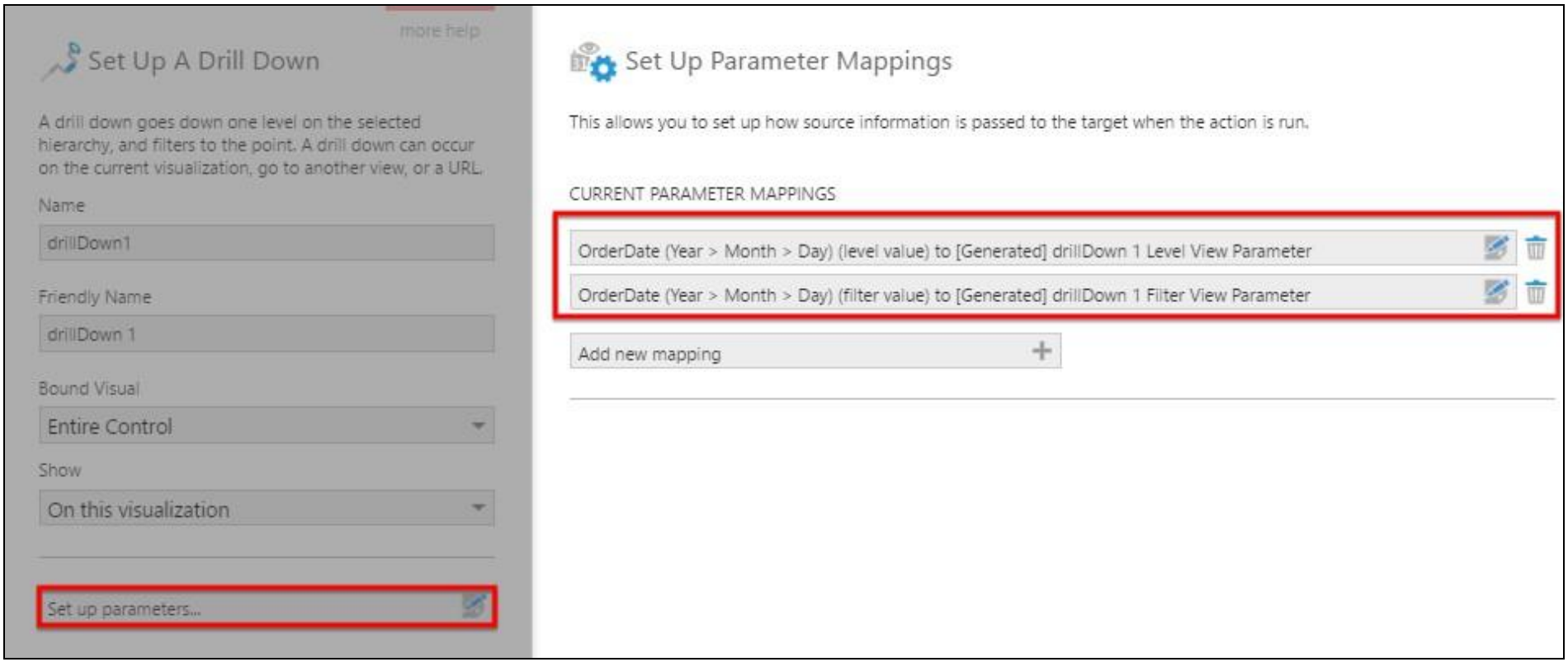
From the submenu, select **Drill Down**.



In the *Set up a Drill Down* dialog, you can leave the default settings and submit the dialog, but for this example we are first going to look at the settings Symphony applied automatically. Click **Set up parameters** before continuing.

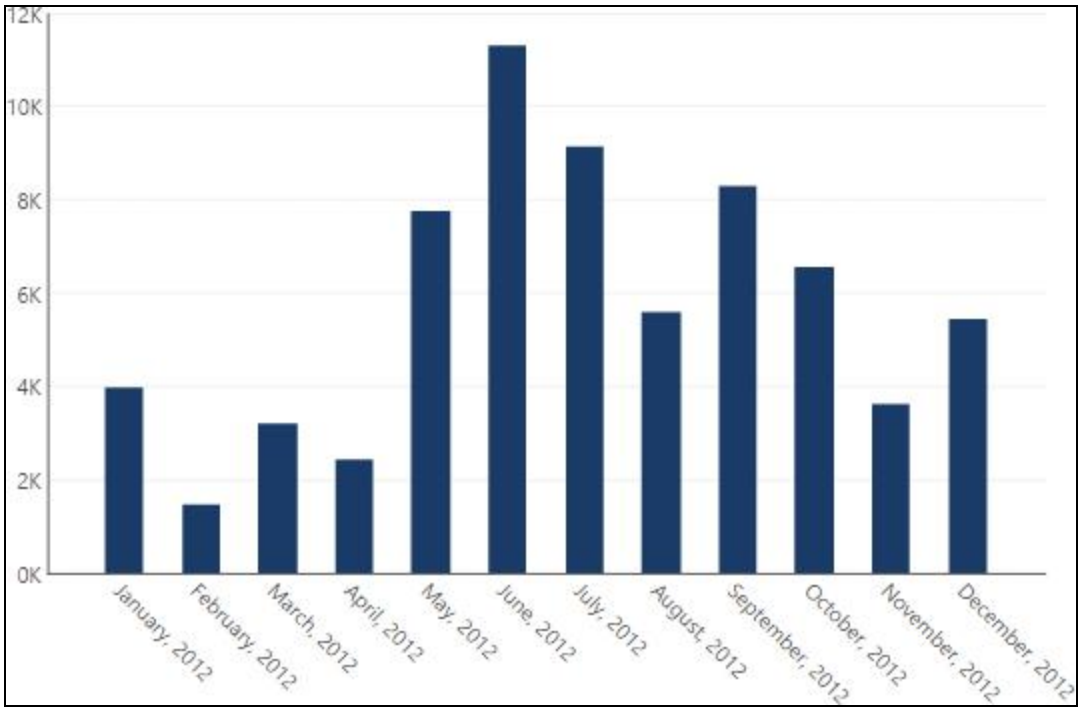


In the *Set Up Parameter Mappings* dialog, you'll see that two mappings have been added automatically for changing the level and then filtering to the data selection. This is where you can choose which hierarchy is filtered and changed down a level, but no further steps are needed right now. Just close this dialog to return to the Setup dialog and click **Submit**.



Test the Interaction

Click **View** in the toolbar, and click a data point (Year 2012) on the bar chart. You should see the level change to Month and the data filtered to 2012.



To drill back up, you can use the browser Back button while the page remains open, right-click (long-tap) options such as **Drill Up**, or corresponding filters if they were added to the view.

Note: You can also add a button with filter actions to reset the level to the top level and the filter to **All**.

For more information, see:

- Using Interactions
- Set Up a Hover Over Interaction
- View Data With a Chart and Drill Down
- Design Overview

Get the Selected Rows in a Table

This applies to: *Managed Dashboards, Managed Reports*

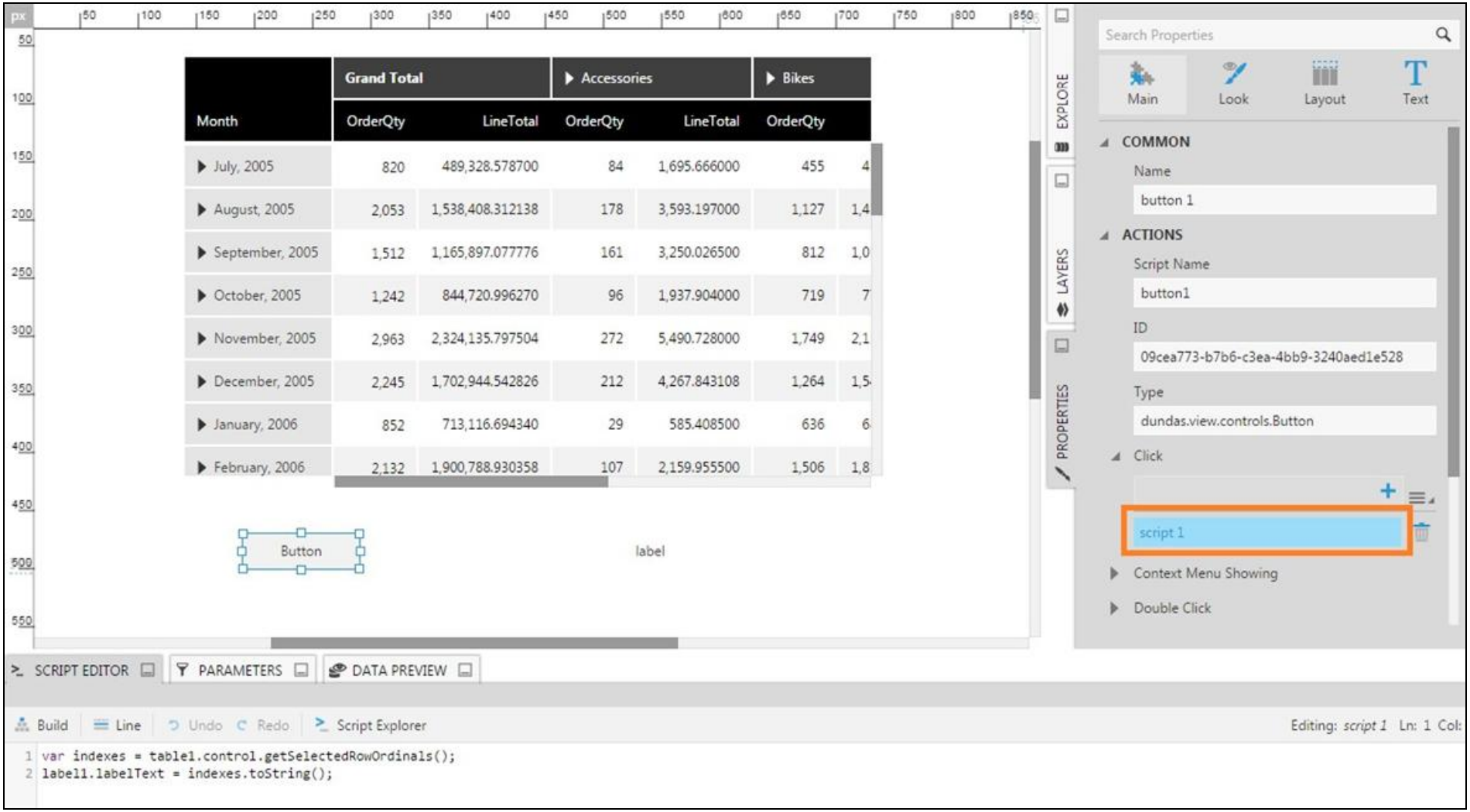
Learn how to get the selected row ordinals and the selected row hierarchy members in a table using script.

Script Library sample: Get selected table data

Get the Selected Row Ordinals

Create a new dashboard and add a table visualization to it with data.

For this example, we will enable the **Allow Multiple Row Selection property** of the table. Also add a button component and a label component.



Add the following script on the Click event of the button.

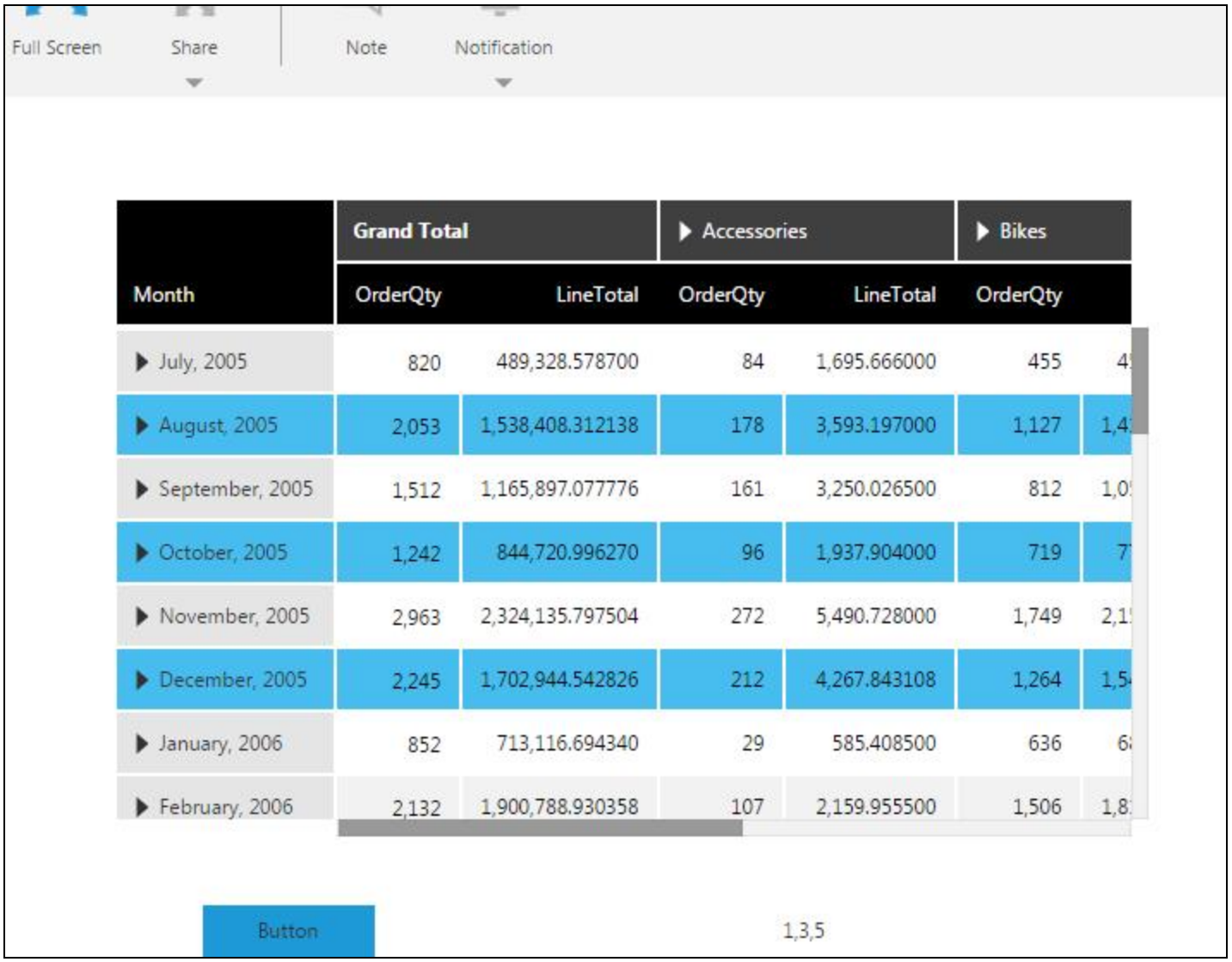
```
var indexes = table1.control.getSelectedRowOrdinals();
```

```
label1.labelText = indexes.toString();
```

This script refers to the following API methods and properties:

- DataGrid.getSelectedRowOrdinals Method
- Label.labelText Property

Switch to **View** mode and select one or more rows in the table. Click the button to see the selected row indexes/ordinals appear in the label.

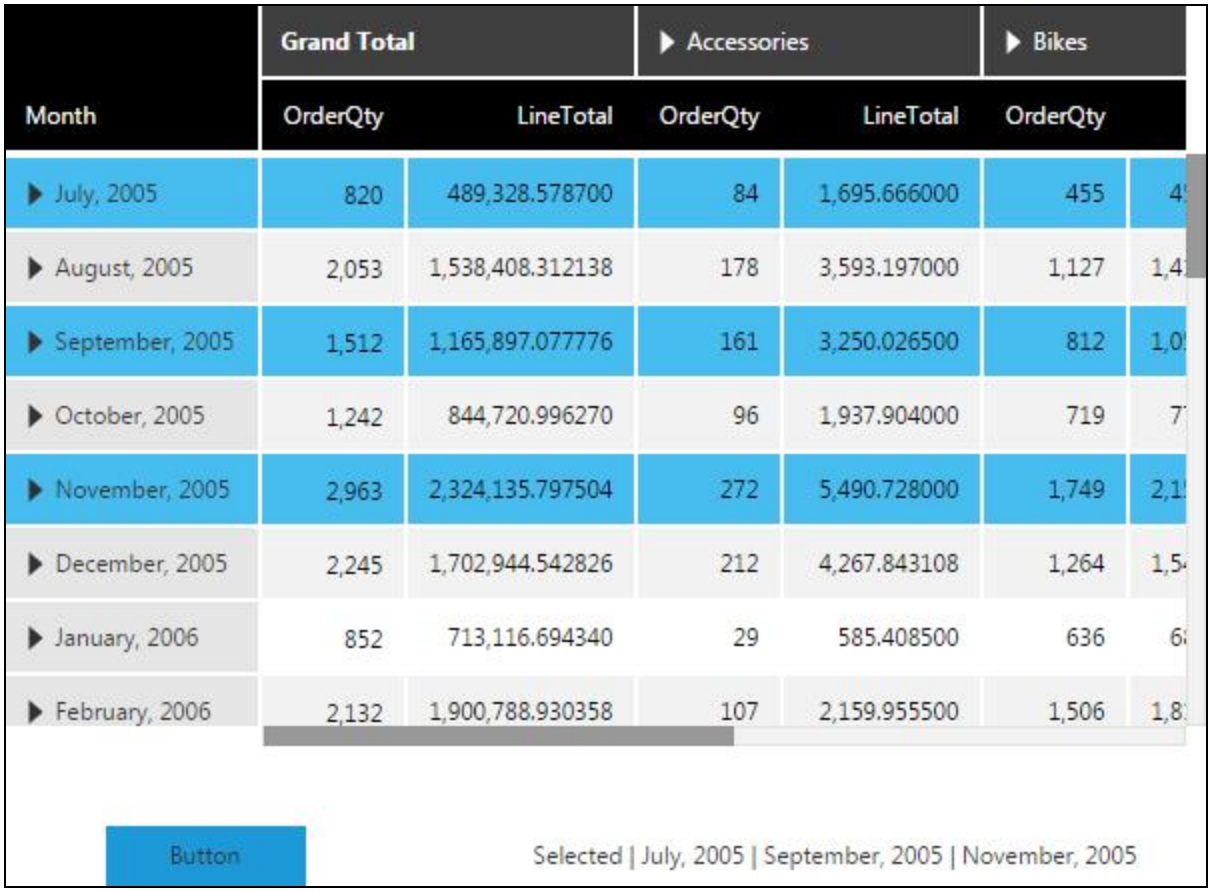


Get the Selected Hierarchy Members

To get the list of hierarchy members of the selected rows in a row header, change the button click script to use the getSelectedRowHierarchyMembers method:

```
var members = table1.getSelectedRowHierarchyMembers();
var text = "Selected";
for(i = 0; i < members.length; i++) {
       text += " | " + members[i].caption;
}
label1.labelText = text;
```

Switch to **View** mode and select one or more rows in the table. Click the button to see the selected hierarchy members appear in the label.



For more information, see:

- JavaScript API
- Script Library: Get selected table data
- Using the Script Editor
- Using a Table Visualization

Handling No Data and Data Errors

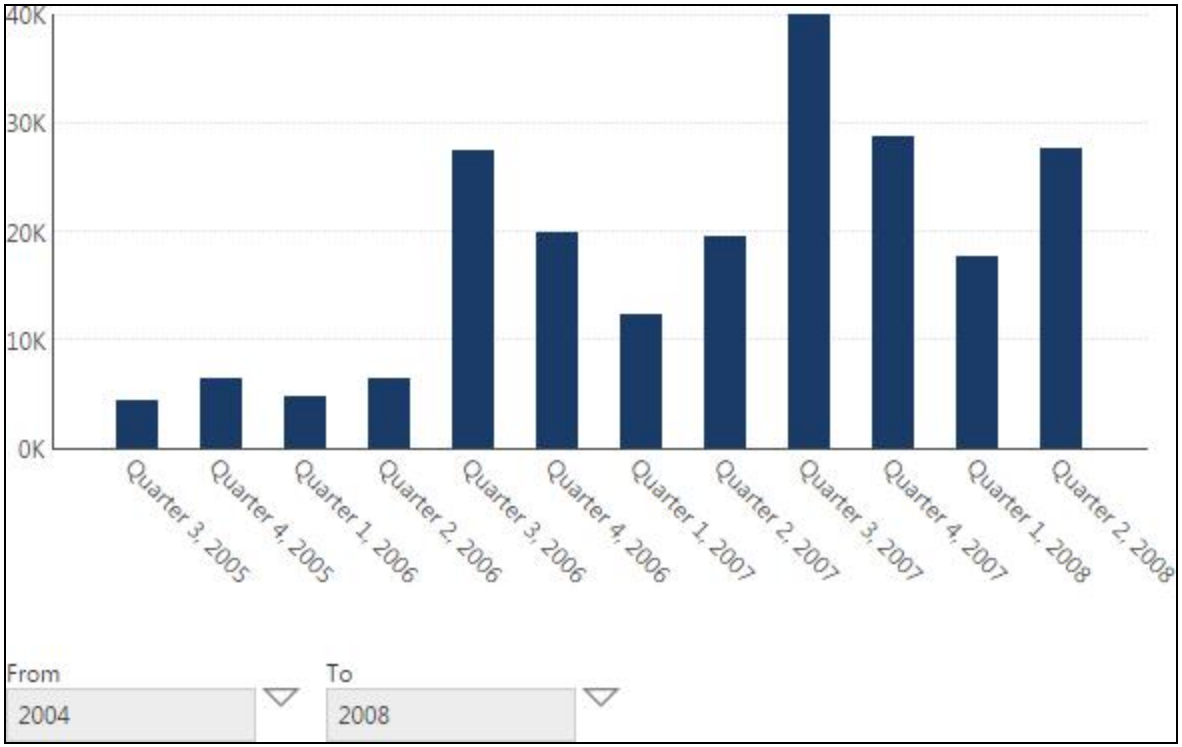
This applies to: *Managed Dashboards, Managed Reports*

At viewing time, if no data is returned for a visualization or if a data error occurs, you'll see a corresponding message displayed by the visualization instead of a blank area. For each data visualization, you can add an interaction when a *No Data* event or a *Data Error* event occurs. This allows you to show/hide layers of content or execute some custom script when these events occur.

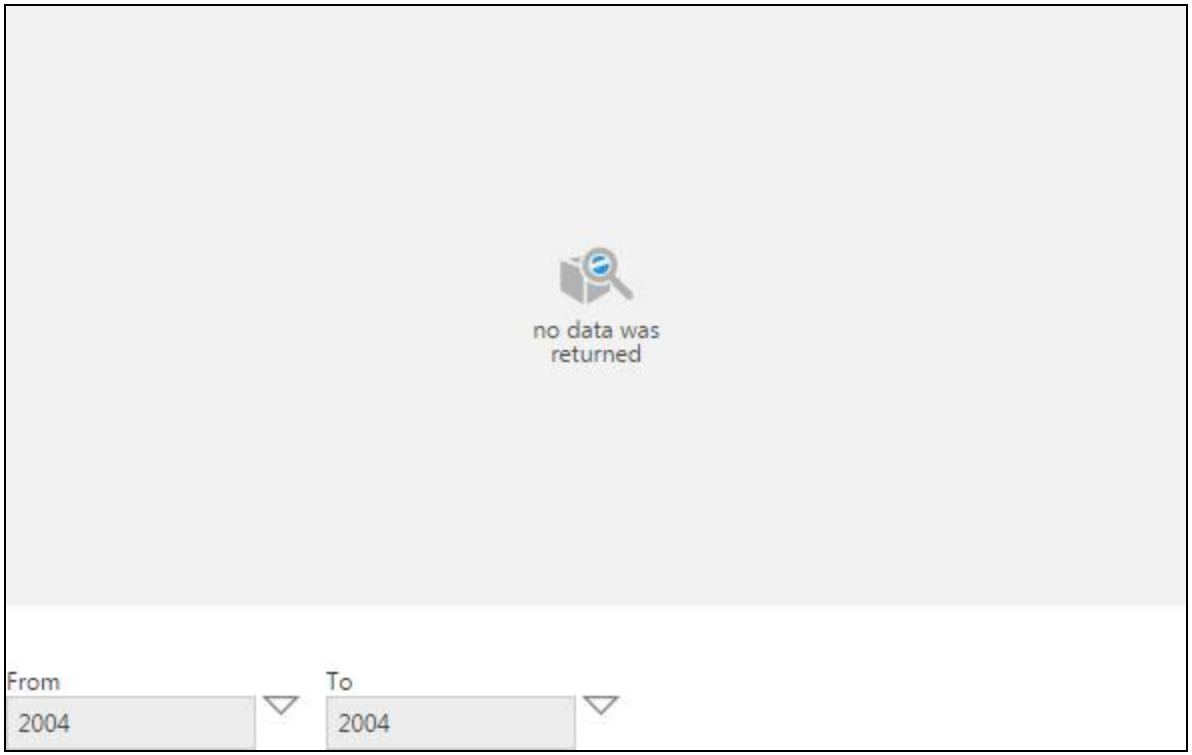
Note: Data error details can be hidden from end users by enabling the advanced configuration setting Hide Error Stack Traces, with the full details available in the logs.

No Data Indicator

As an example, consider the following dashboard consisting of a chart and a calendar range filter.



In View mode, set the calendar range of the filter to a value for which there is no data (e.g., 2004 to 2004). Instead of showing a blank chart, you see a no data was returned indicator.



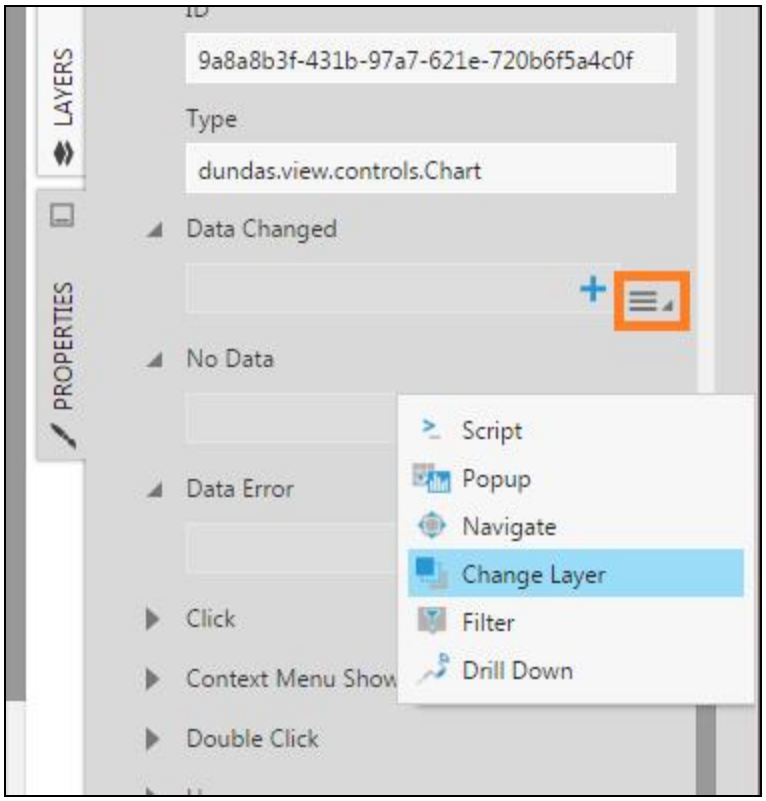
Data-Related Events

There are three data-related events that can occur for data visualizations. For each of these events you can add an action to run such as Change Layer or your own script. For example, if No Data occurs for a chart, you may want to hide the chart and associated labels completely.

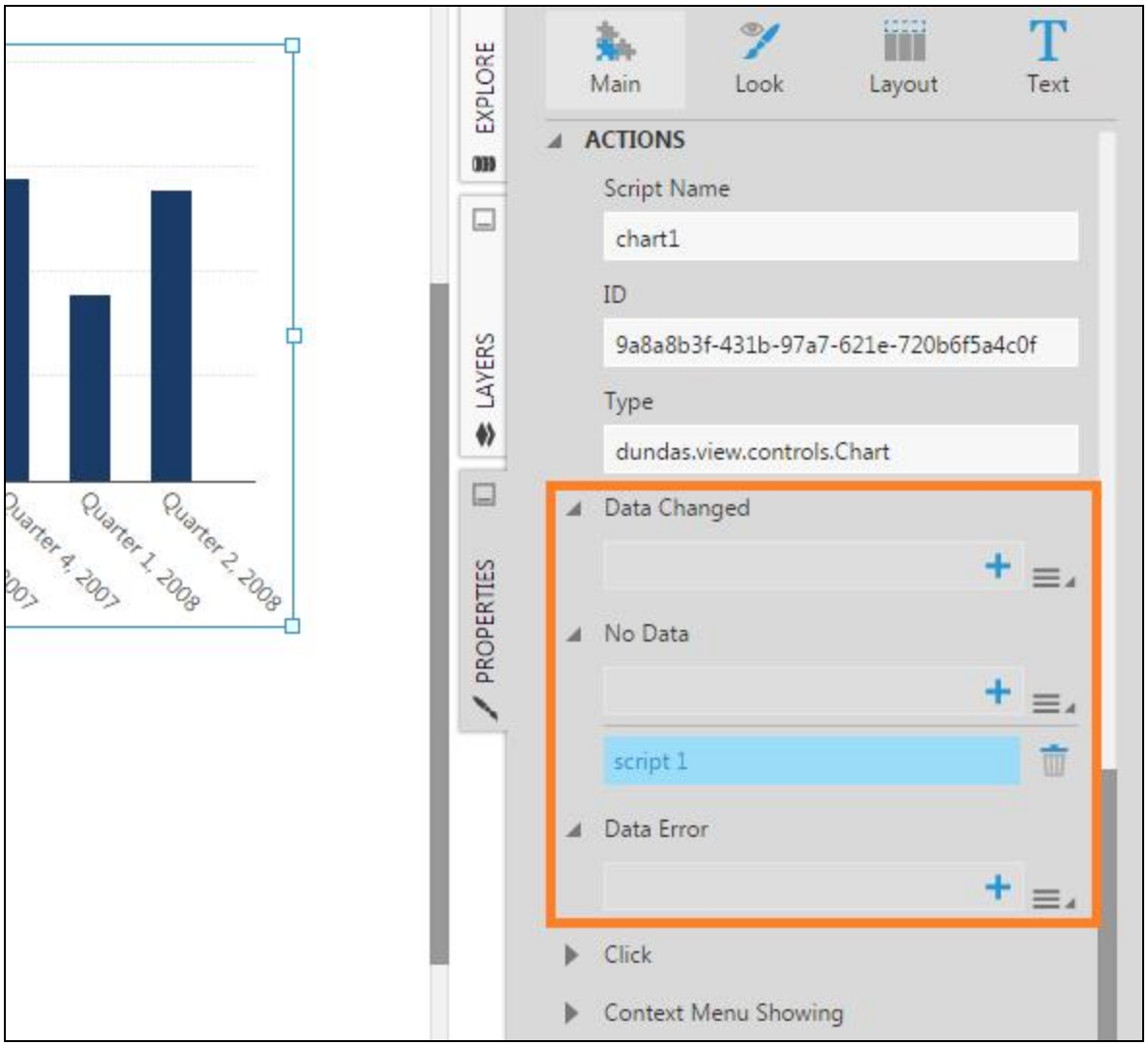
You can find these in the **Properties** window of any data visualization in the Actions section:

- **Data Change** - Occurs when the data visualization's data changes in any way after initially loading. This will always fire before No Data and Data Error.
- **No Data** - Occurs when the control doesn't receive any data back.
- **Data Error** - Occurs when the control's data changes, and the data has an error and cannot be shown.

To choose an action to add, click the menu icon to the right.



To quickly add a default action (e.g., script if you are a developer user), click the **+** button.



For more information about setting up interactions, see Using Interactions.

For more information, see:

- Layers and Groups
- Add a Change Layer Action on Ready Event

How To Get The Unique Name of a Hierarchy Member

This applies to: *Managed Dashboards, Managed Reports*

This article demonstrates how to get the unique name of a hierarchy member. This can be useful in cases such as setting row-level security using a security hierarchy or passing parameter values via query string.

You can display the unique name of the hierarchy member from different screens within . Wherever available, hover over the hierarchy member to display a tooltip with the hierarchy member's unique name. However, it is important to distinguish between two stages of the hierarchy that have different unique names:

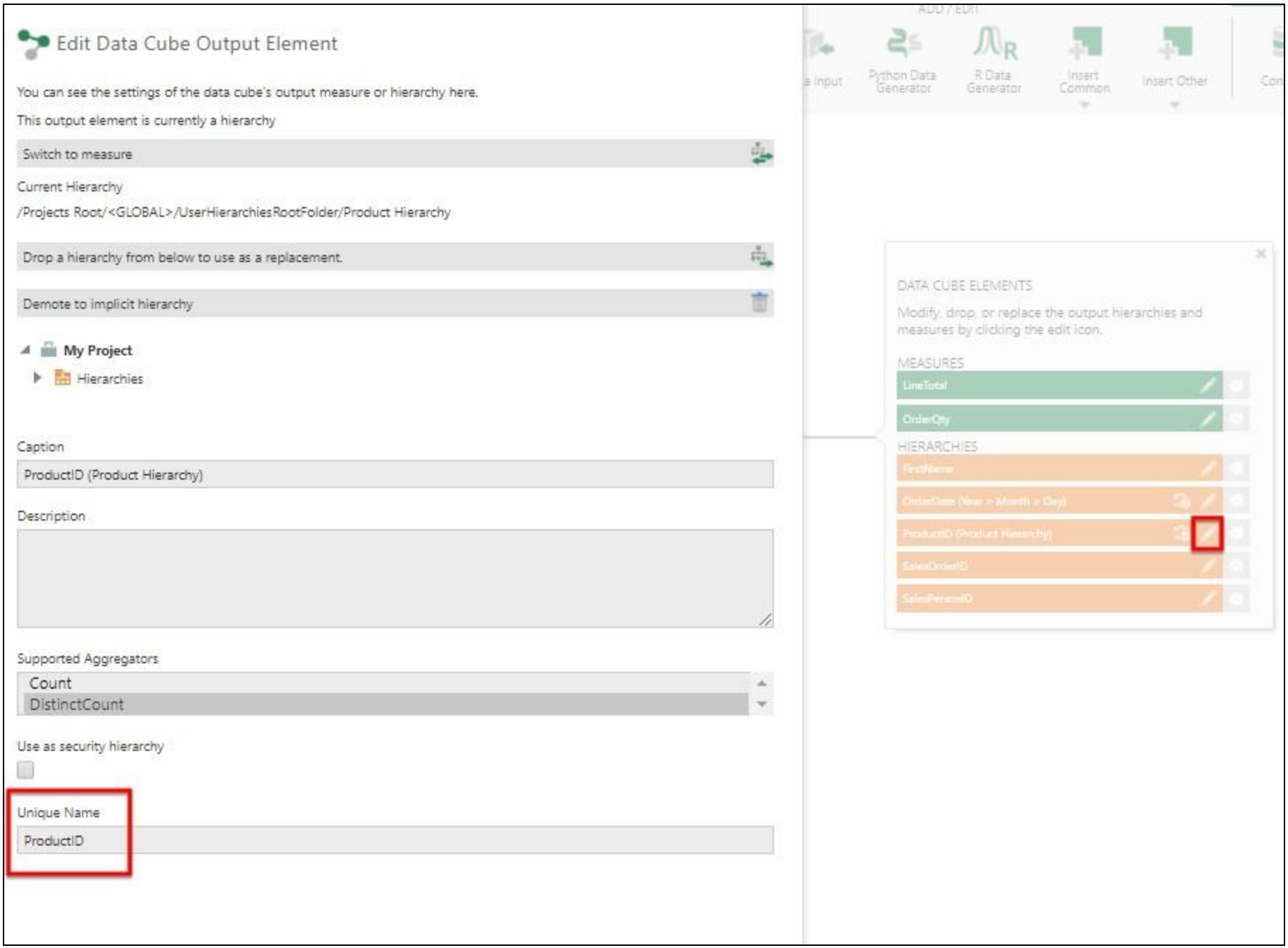
- When creating a hierarchy from the main menu or through a time dimension, the members have a shorter unique name we can call the 'design' unique name, because it is not yet used in a data cube or metric set. For example, the fourth item of the first level of the Product hierarchy may get the unique name **4.A**.

- When using one of these hierarchies in a data cube or metric set, the unique names may change and incorporate the name of the element that's using the hierarchy. For example, after promoting the ProductID column, the member above may get the unique name 4.A.ProductID.

In most cases, you will be working with the second case.

Customize the Hierarchy Unique Name

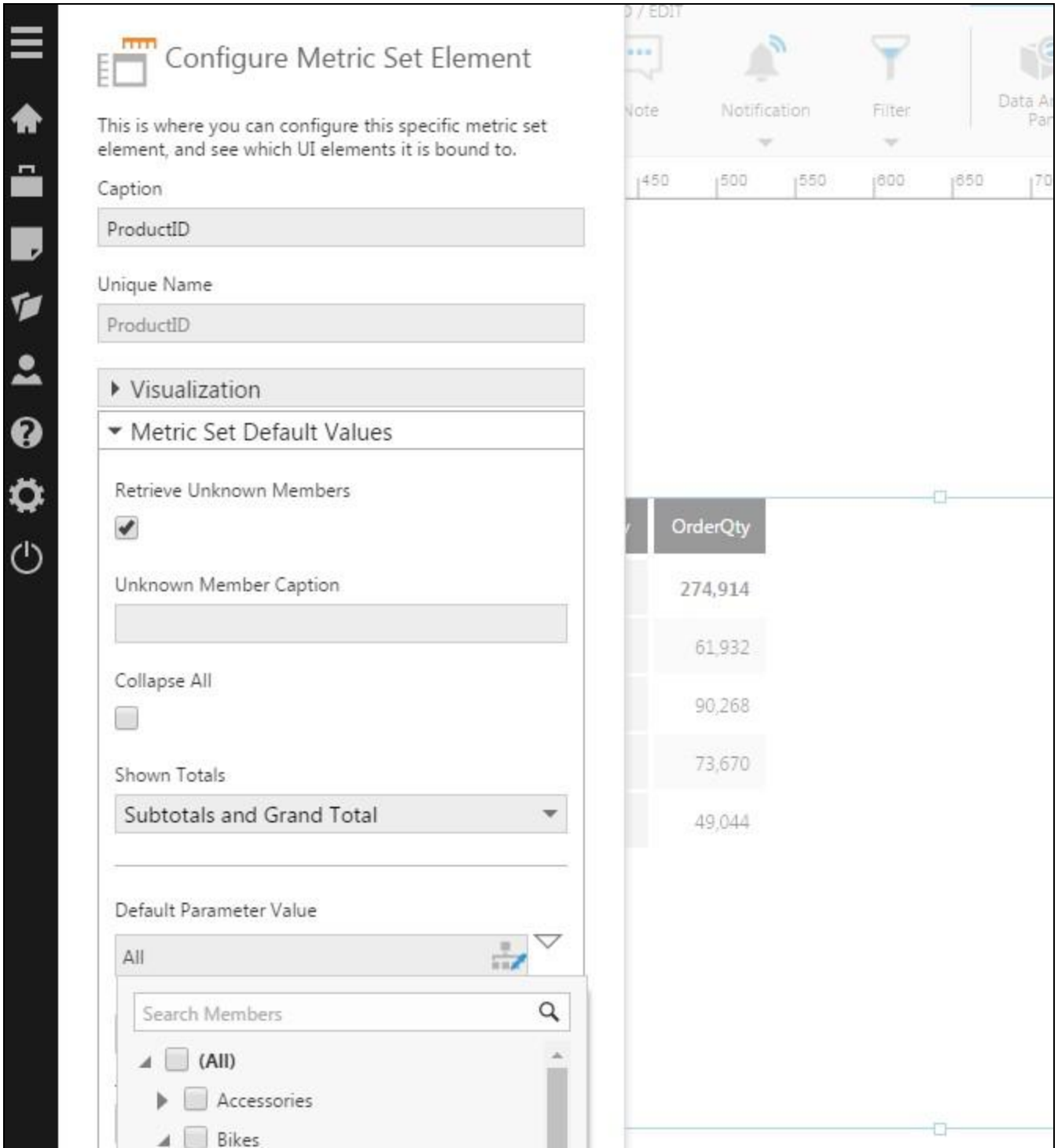
When you are using a data cube to replace or promote a column to a hierarchy, you can customize the unique name for that hierarchy. This will be used together with the shorter unique name to generate the unique names for the members of the hierarchy.


Edit Data Cube Output Element
You can see the settings of the data cube's output measure or hierarchy here.
This output element is currently a hierarchy
Switch to measure
Current Hierarchy
/Projects Root/<GLOBAL>/UserHierarchiesRootFolder/Product Hierarchy
Drop a hierarchy from below to use as a replacement.
Demote to implicit hierarchy
My Project
Hierarchies
Caption
ProductID (Product Hierarchy)
Description
Supported Aggregators
Count
DistinctCount
Use as security hierarchy
Unique Name
ProductID
ADD / EDIT
Python Data Generator
R Data Generator
Insert Common
Insert Other
DATA CUBE ELEMENTS
Modify, drop, or replace the output hierarchies and measures by clicking the edit icon.
MEASURES
LineTotal
OrderQty
HIERARCHIES

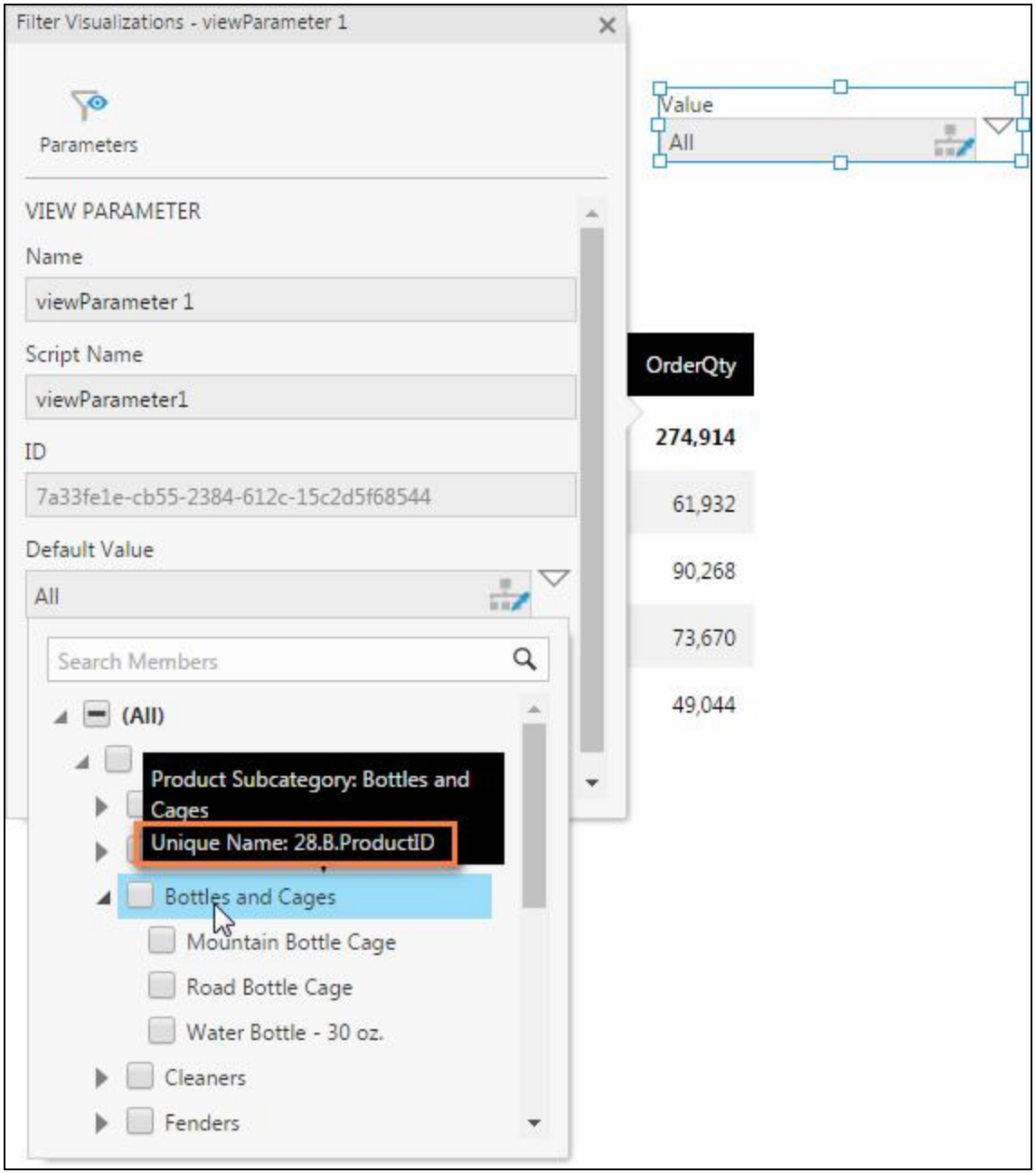
Show the Member Unique Name

When a filter is connected to the hierarchy and lists its members, you can access the unique names in a tooltip by hovering over them when logged in as a power user or higher.

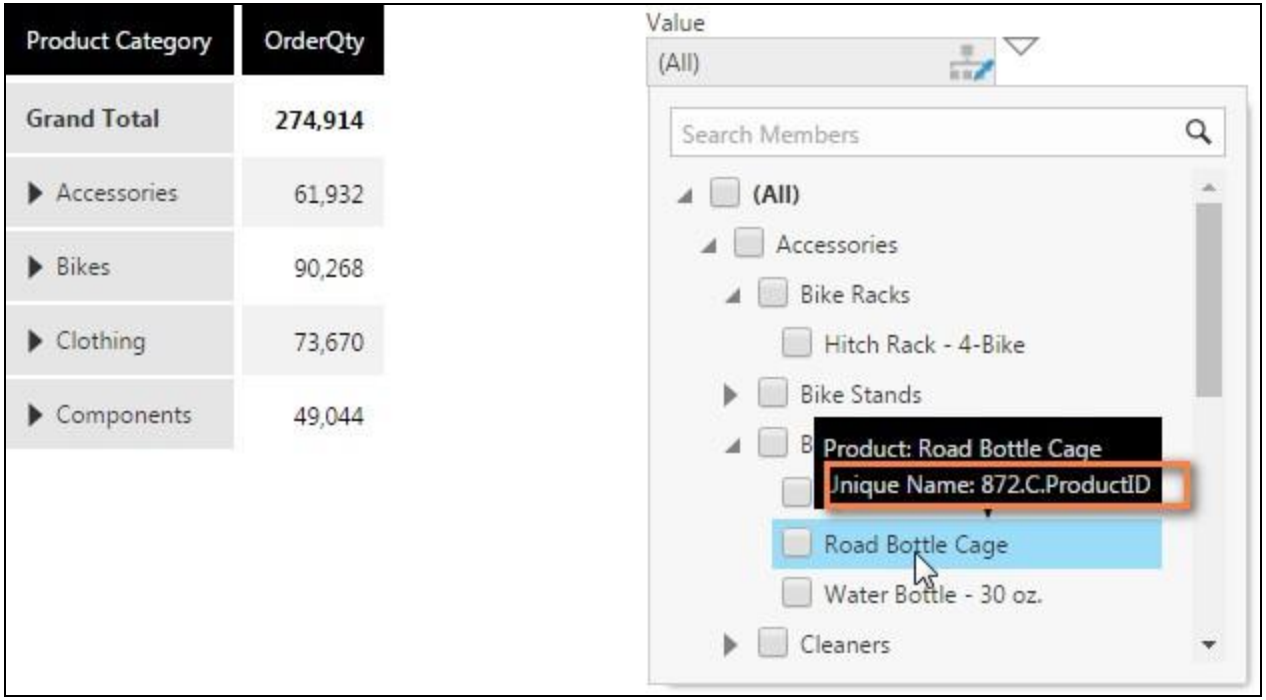
- From the Configure Metric Set Element dialog opened from the Data Analysis Panel:

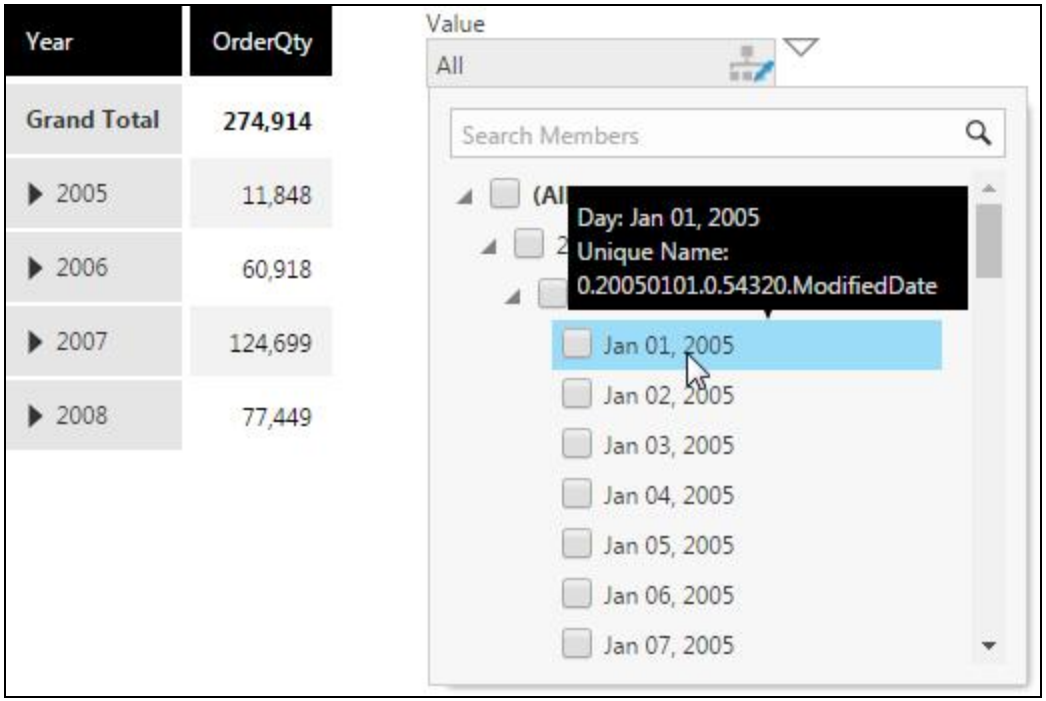

Configure Metric Set Element
This is where you can configure this specific metric set element, and see which UI elements it is bound to.
Caption
ProductID
Unique Name
ProductID
Visualization
Metric Set Default Values
Retrieve Unknown Members
Unknown Member Caption
Collapse All
Shown Totals
Subtotals and Grand Total
Default Parameter Value
All
Search Members
(All)
Accessories
Bikes
Note
Notification
Filter
450
500
550
600
650
OrderQty
274,914
61,932
90,268
73,670
49,044

- From the *Filter Visualizations* popup:


Filter Visualizations - viewParameter 1
Parameters
VIEW PARAMETER
Name
viewParameter 1
Script Name
viewParameter1
ID
7a33fe1e-cb55-2384-612c-15c2d5f68544
Default Value
All
Search Members
(All)
Product Subcategory: Bottles and Cages
Unique Name: 28.B.ProductID
Bottles and Cages
Mountain Bottle Cage
Road Bottle Cage
Water Bottle - 30 oz.
Cleaners
Fenders
Value
All
OrderQty
274,914
61,932
90,268
73,670
49,044

- From a filter when viewing a dashboard or other view:

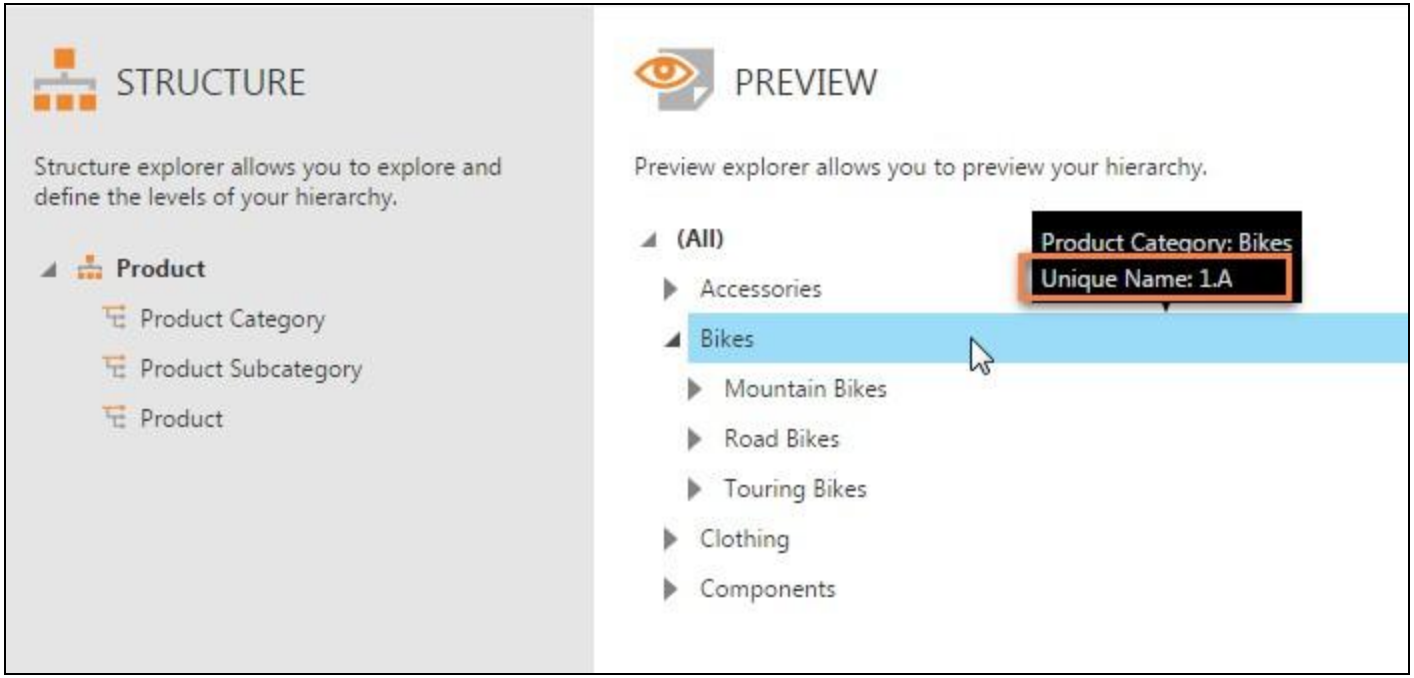


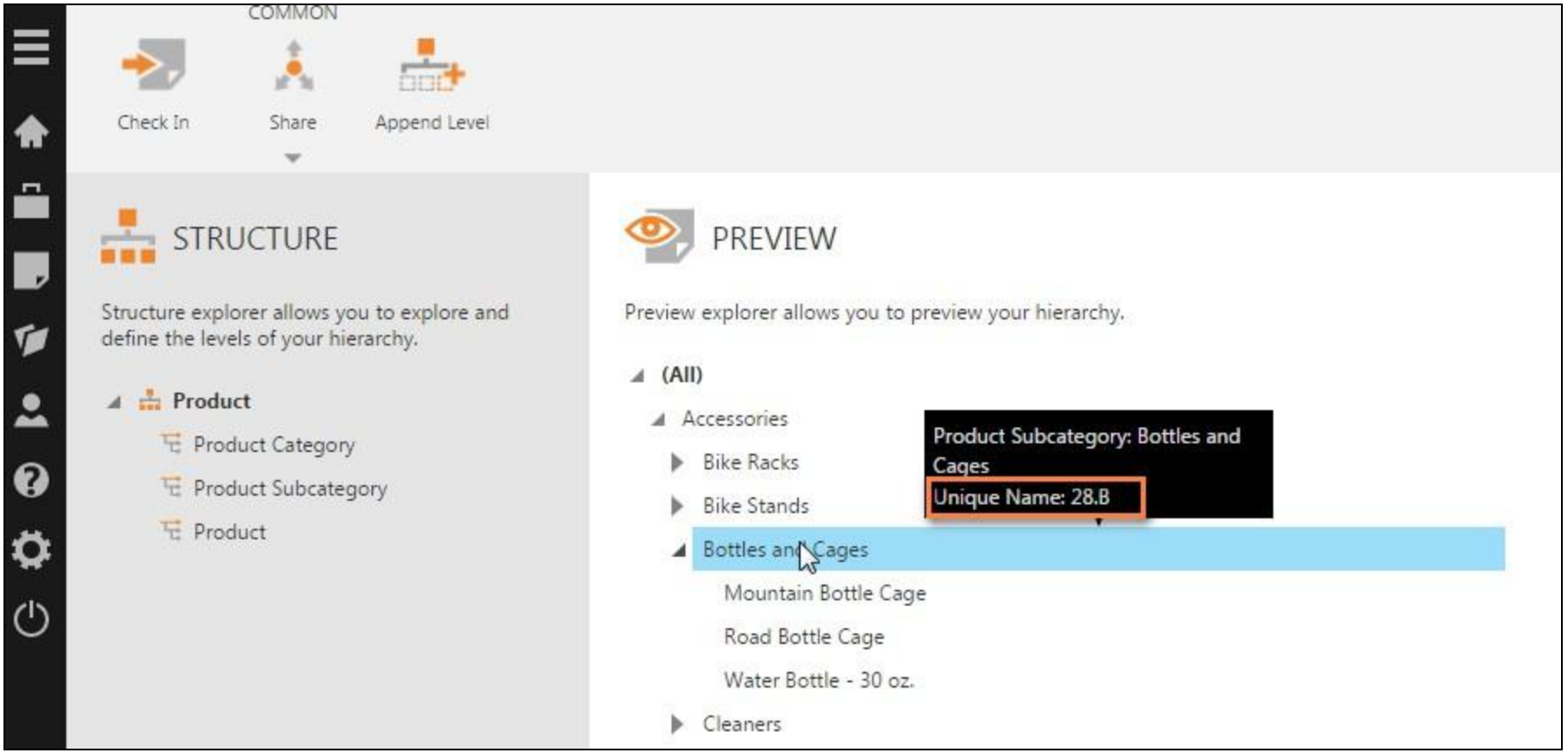


Show the Design Unique Name

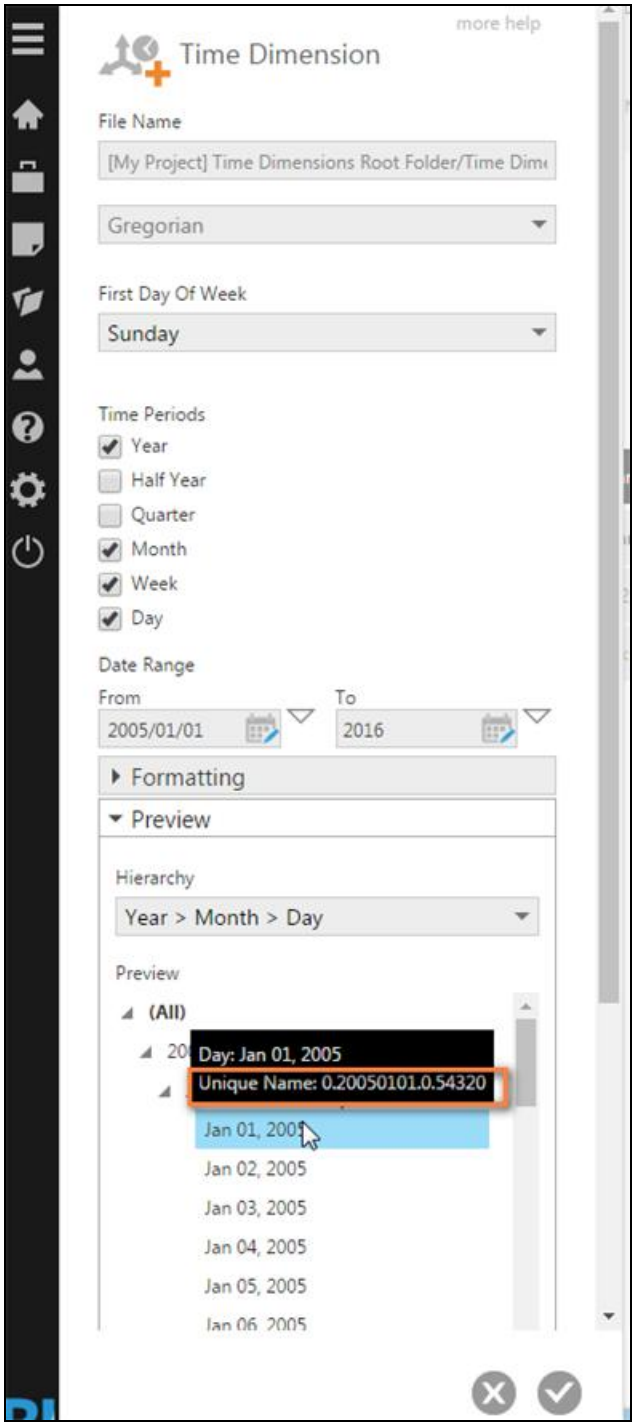
You can hover over members while editing a hierarchy or time dimension in a similar way, which displays the unique name before it's used in a data cube or metric set:

- From the hierarchy editor preview:





- From the Time Dimension preview:


more help
Time Dimension
File Name
[My Project] Time Dimensions Root Folder/Time Dim
Gregorian
First Day Of Week
Sunday
Time Periods
Year
Half Year
Quarter
Month
Week
Day
Date Range
From
2005/01/01
To
2016
Formatting
Preview
Hierarchy
Year > Month > Day
Preview
(All)
Day: Jan 01, 2005
Unique Name: 0.20050101.0.54320
Jan 01, 2005
Jan 02, 2005
Jan 03, 2005
Jan 04, 2005
Jan 05, 2005

For more information, see:

- Automatic Joins and Hierarchies
- Using a Security Hierarchy to Filter Data by User
- Create a Time Dimension

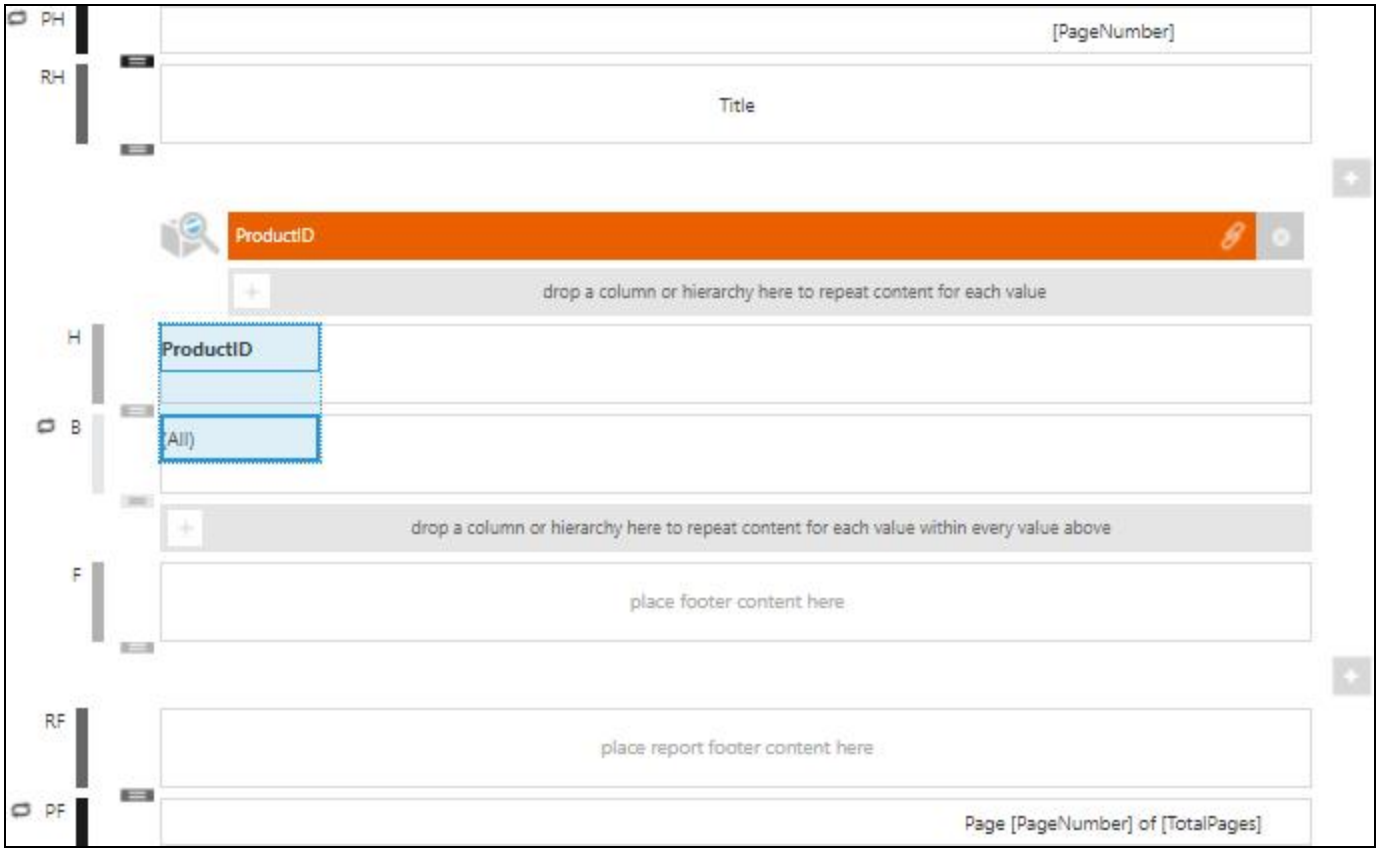
Promote a Report Column to a Hierarchy

This applies to: *Managed Dashboards, Managed Reports*

This article shows you how to promote or replace a column with a predefined hierarchy in a report, scorecard, or small multiple.

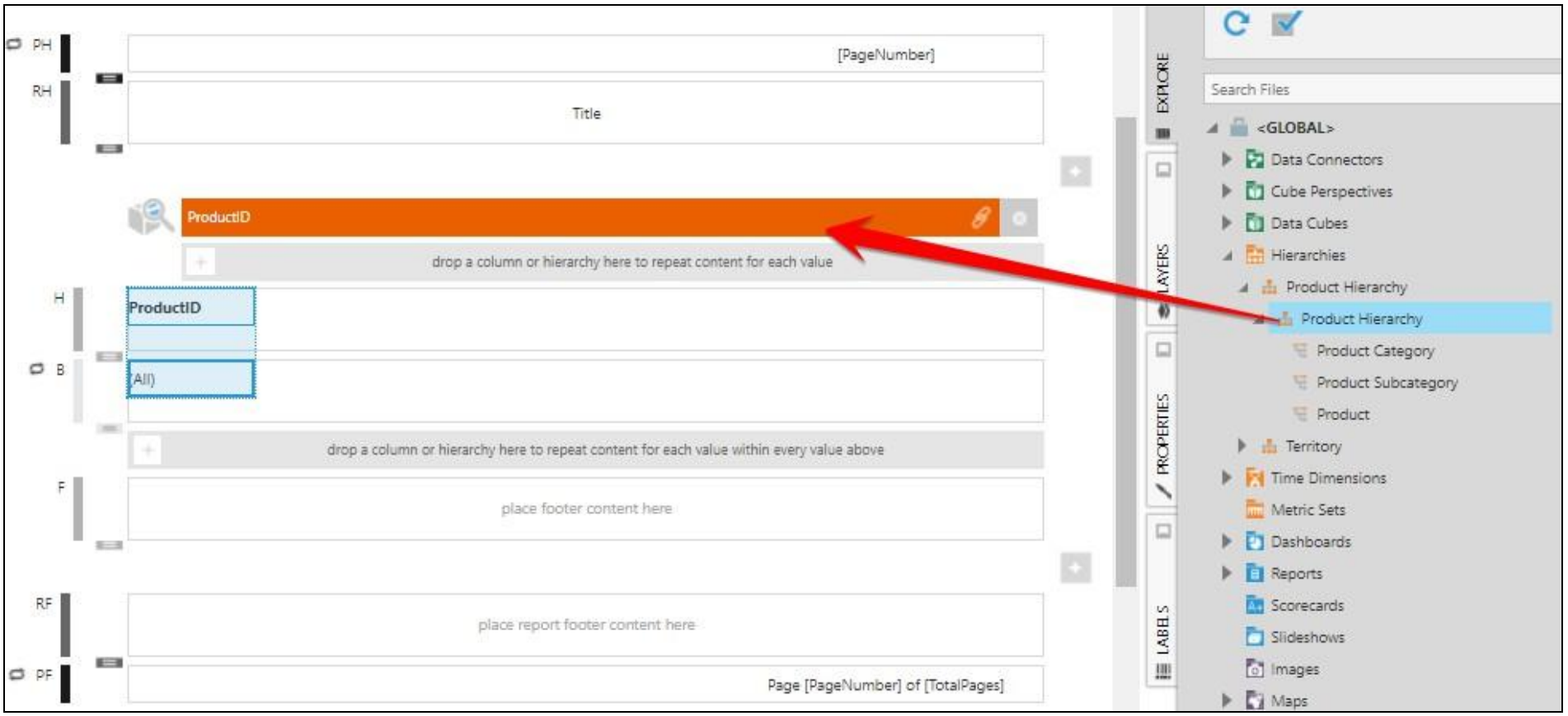
As an example, create a new report from the main menu.

Drag the ProductID column from the [Sales].[SalesOrderDetail] table to the report grouping area (drop data here...). It will appear as a data label and create a corresponding header element in the report header region.



If you switch to **View** mode now, the report will display a list of product IDs, which isn't very useful.

To promote this column to a hierarchy, drag the *Product* hierarchy from the **Explore** window onto the ProductID repeater column in the grouping area.



Note: You can also open the group metric set's Data Analysis Panel by clicking the cube icon to the left, where you can promote the hierarchy and access other options the same as for any other metric set.

Switch to **View** mode to see the report. Observe that the data label displays product names instead of IDs.

For more information, see:

- Design Tips
- Automatic Joins and Hierarchies

Read Data From a Visualization By Script

This applies to: *Managed Dashboards, Managed Reports*

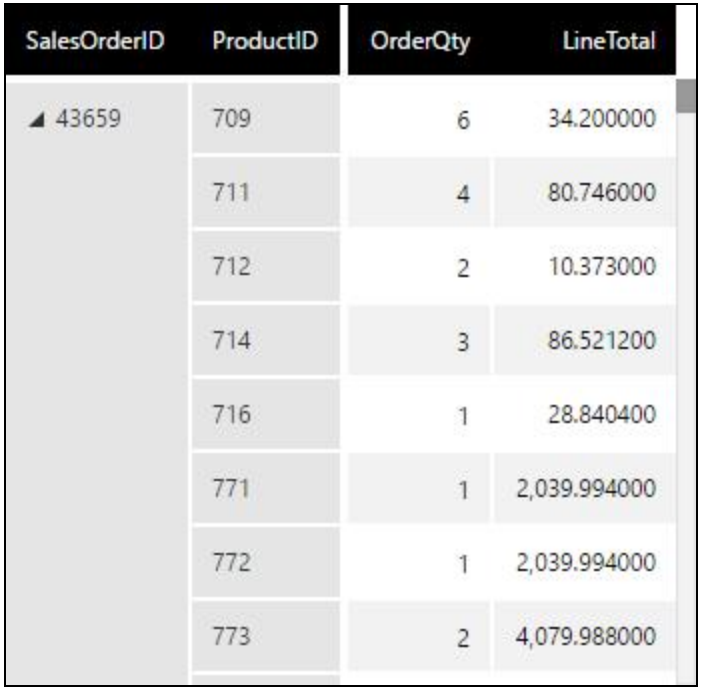
Visualizations request a metric set's data from the server for display purposes only, and may not immediately include the entire result, for example if that data is currently scrolled out of view. The JavaScript API can be used to issue a request for a metric set's data for your own purposes.

Script Library sample: Insert scripted third party web content sample

Set Up the Visualization

The data for any visualization is always retrieved using metric sets in the same way, so the same script is applicable for all visualization types, or for any metric set if you specify its ID as the *objectId*.

For this example, we are dragging data from a data cube or data connector to create a table on the dashboard canvas with the desired data. We can then request the same data displayed by this table using script.



SalesOrderID	ProductID	OrderQty	LineTotal
◢ 43659	709	6	34.200000
	711	4	80.746000
	712	2	10.373000
	714	3	86.521200
	716	1	28.840400
	771	1	2,039.994000
	772	1	2,039.994000
	773	2	4,079.988000

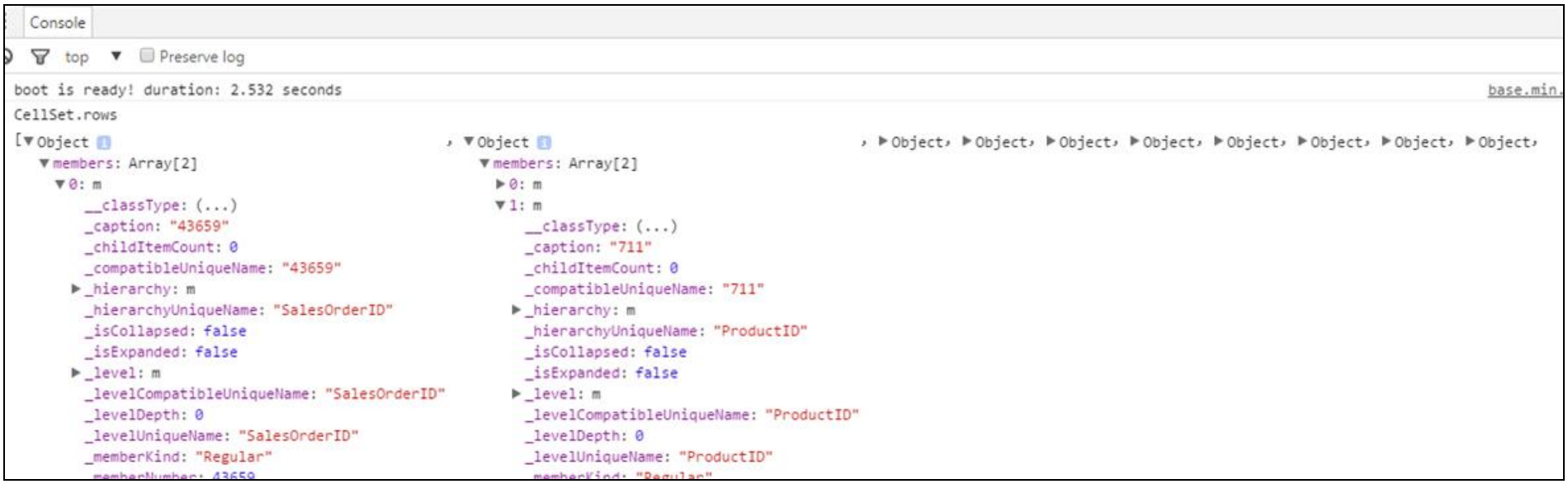
Retrieve Data

We will trigger script using a button for this example, choosing **Components** and then **Button** from the toolbar to add one.

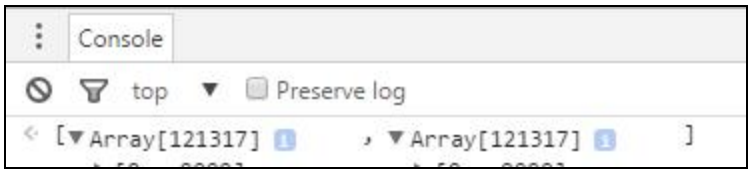
In the **Properties** window under Actions, look for the button's **Click** actions. Click the **+** button to add a script action for retrieving data, then click the new item below to open the script editor and enter JavaScript like the following:

```
var dataRetrievalService = this.getService("DataRetrievalService");

// Create a Request object with our own paging settings:
var request = new dundas.data.Request({ objectId: table1.metricSetBindings[0].metricSetId });
request.pagingOptions.pagingKind = dundas.data.PagingKind.SEQUENCE;
request.pagingOptions.rowSequenceSize = 100;

// Request the data:
var def = dataRetrievalService.getData(request);

def.done(function(results) {
       // The request was successful:
       var cellset = results[0].cellset;
       debugger; // (check the result in the browser's debugging tools, then replace this line)
});
```

This example accesses data from the same metric set displayed by a table visualization with script name table1 on the dashboard. Once the cellset has been retrieved, the data contained in its rows, columns, and cells can be accessed: see the dundas.data.DataCellSet class in the JavaScript API reference for details. The cellset for the example above contains 100 rows:

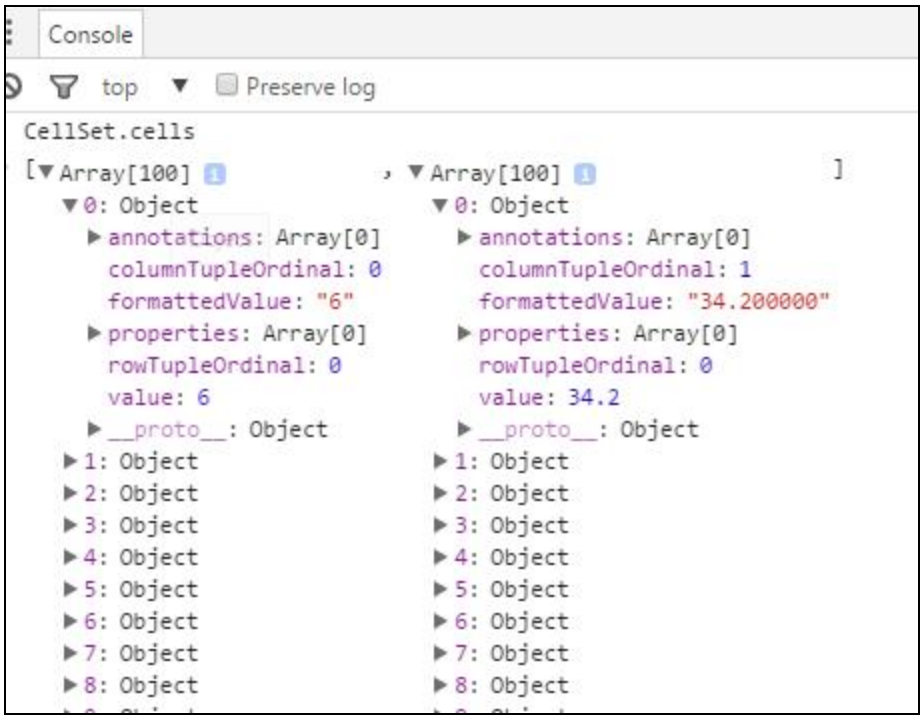


The dundas.data.Request class has options that can be customized. When pagingKind is set to dundas.data.PagingKind.NONE, all the rows in the table are retrieved:



Note: If the pagingKind is set to NONE and the data volume is too much, the browser may not be able to handle the amount of data requested. In this case, consider if a server-side plugin should be used instead.

The measure values can be accessed by looping through the nested cells array, e.g., cellset.cells[columnIndex][rowIndex]. As there are two measures in the table above, cells contains two arrays of size 100 (as specified in the paging options).



Retrieve Filtered Data

The request created in the code above will fetch all the data without filtering. To retrieve only data filtered like some visualization on the dashboard, parameter values should be included in the request.

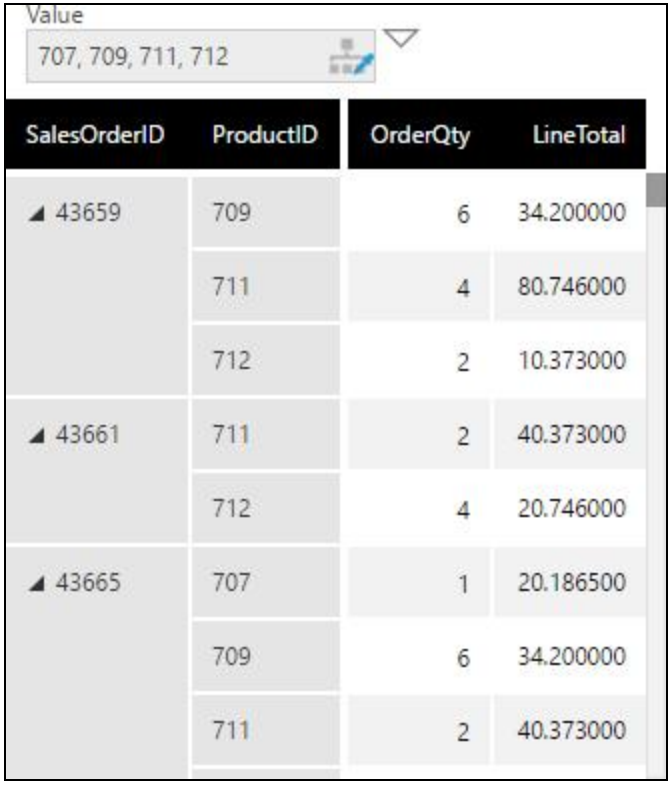
Parameter values can be created in script as done in other articles, or parameter values could be copied from a visualization displaying the same metric set.

Assuming the same metric set is also dragged onto the dashboard, the code above could be modified as follows to refer to it (i.e., table1):

```
var dataRetrievalService = this.getService("DataRetrievalService");

// Create a Request object with our own paging settings:
var request = new dundas.data.Request({ objectId: table1.metricSetBindings[0].metricSetId });
request.pagingOptions.pagingKind = dundas.data.PagingKind.SEQUENCE;
request.pagingOptions.rowSequenceSize = 100;

// Get the last request from the visualization:
```

```
var vizRequest = table1.metricSetBindings[0].dataResult.request;

// Copy the parameter values, which will have matching parameterIds if for the same metric set:
vizRequest.parameterValues.forEach(function (parameterValue) {
request.parameterValues.push(Class.clone(parameterValue));
});

var def = dataRetrievalService.getData(request);
def.done(function(results) {
       var cellset = results[0].cellset;
       debugger; // (check the result in the browser's debugging tools, then replace this line)
});
```

This cellset will contain only the rows and columns that match the filtering criteria. For example, if a filter is applied on the **ProductID** in the sample table:



SalesOrderID	ProductID	OrderQty	LineTotal
◢ 43659	709	6	34.200000
	711	4	80.746000
	712	2	10.373000
◢ 43661	711	2	40.373000
	712	4	20.746000
◢ 43665	707	1	20.186500
	709	6	34.200000
	711	2	40.373000

cellset.cells contains only 9743 rows instead of the total available 12173 rows.

For more information, see:

- Javascript API
- Script Library: Insert scripted third party web content sample
- Script Library: Get click or interactive data
- DataRetrievalService API
- DataRetrievalController class
- Get the Selected Rows in a Table
- Writing Scripts Using Browser Developer Tools

Rename a Script

This applies to: *Managed Dashboards, Managed Reports*

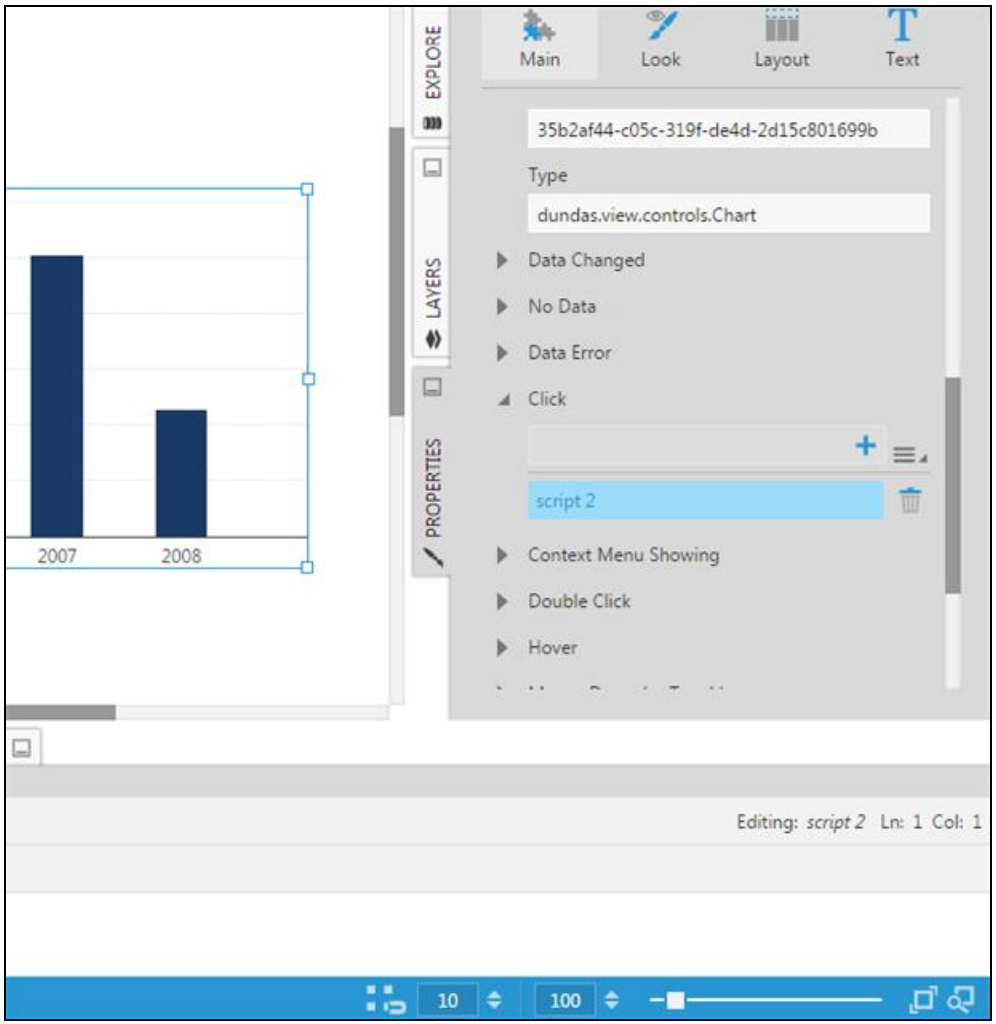
When you add a script action to a data visualization, the script is automatically named *script 1, script 2,* and so on. This article shows you how to rename these script actions to something more descriptive using the Layers window.

Note: You can also rename other actions such as change layer in a similar way.

Add a Script

For this example, first add a chart visualization to your dashboard.

Go to **Properties** and add a script for the chart's **Click** event.

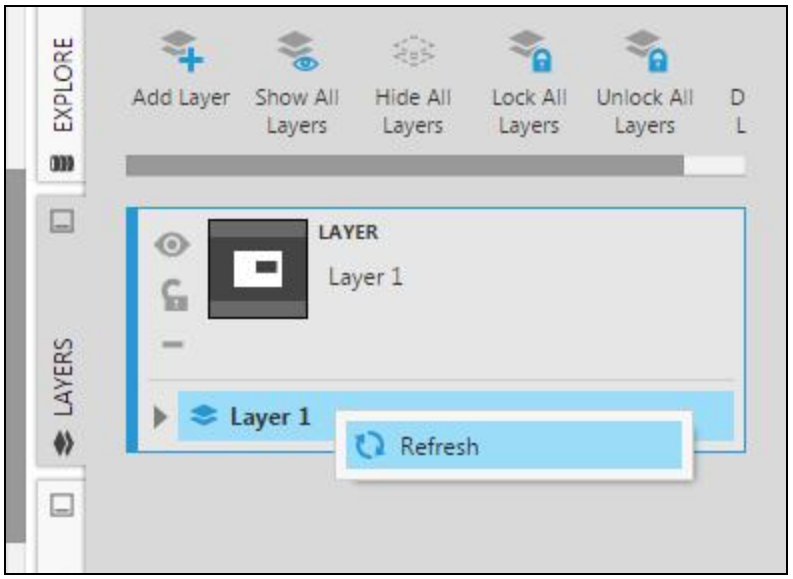


Rename the Script Action

Go to the **Layers** window and click the layer containing the chart.

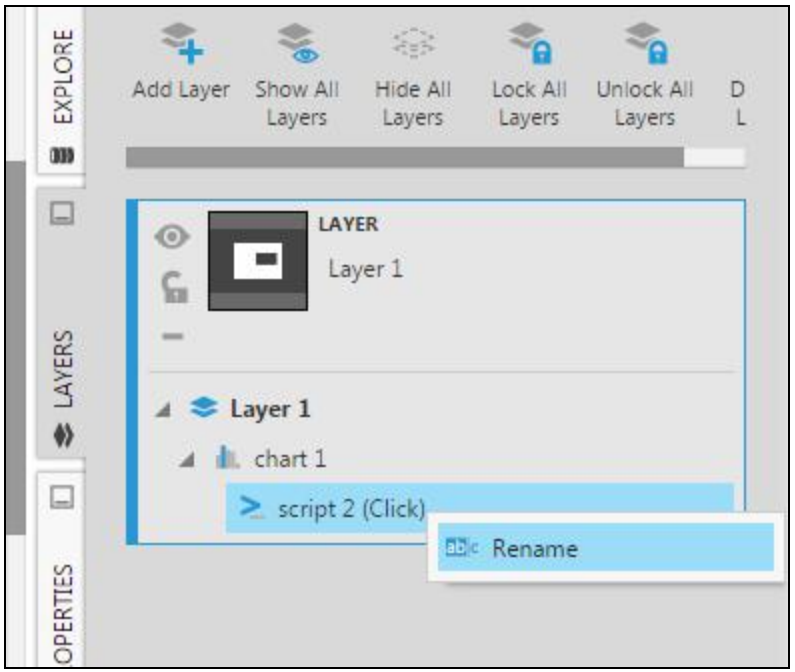
Click the **plus sign** to see the list of elements within the layer.

Right-click over the layer node and select **Refresh**. This ensures the list of elements is up-to-date and includes the script you just added.

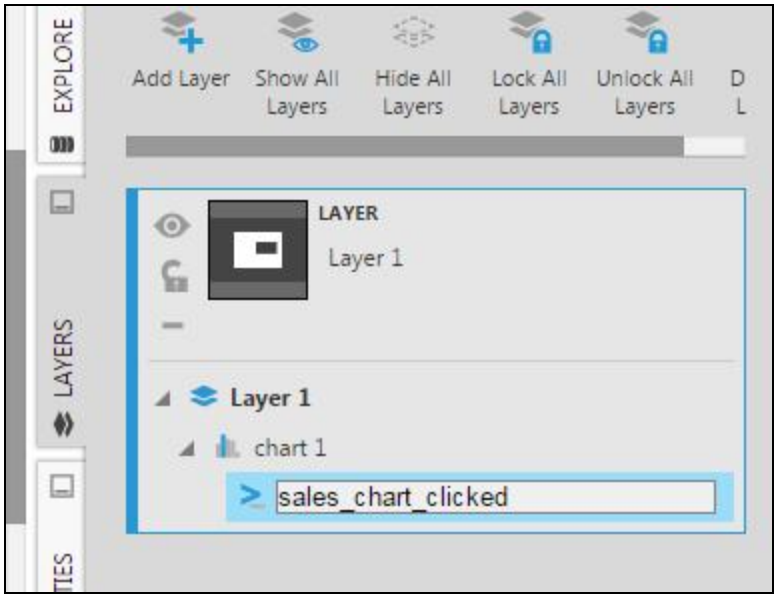


Expand the layer node and the chart node. You should see a script node listed as well.

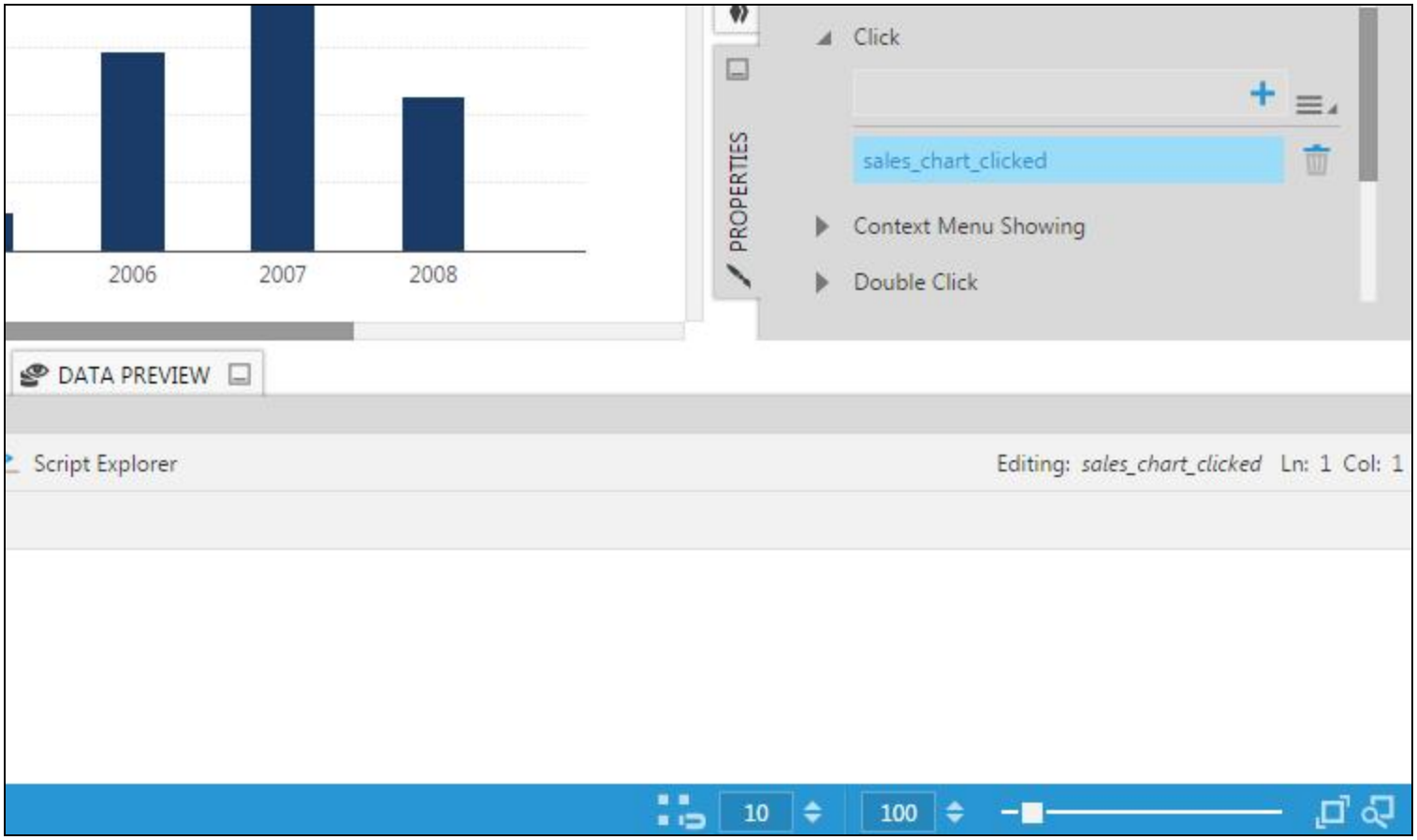
Right-click over the script node and select **Rename**.



Enter the new name for the script in the text box.



Go back to the Script Editor to see that the script has been renamed.



For more information, see:

- Layers and Groups
- Using the Script Editor

How To Set Up a Slideshow

This applies to: *Managed Dashboards, Managed Reports*

Similar to a PowerPoint presentation, slideshows allow users to display their own selection of dashboards or other views in sequence. You can use them in a presentation, or to rotate through different views for monitoring purposes (e.g., in an office, operation center, or kiosk).

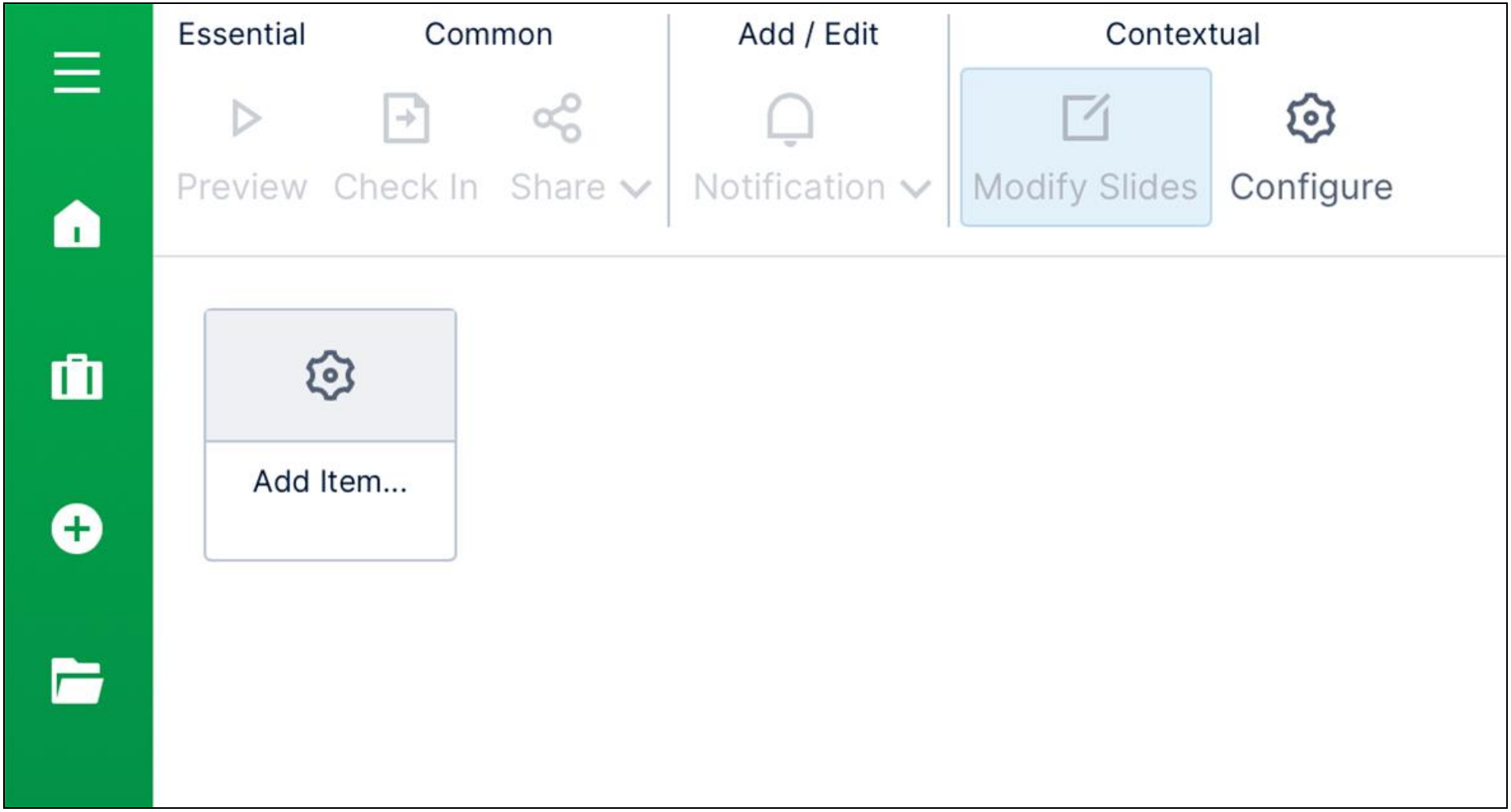
You can also export a slideshow as a PDF file, a PowerPoint presentation, or a set of images in version 23.3 and higher, so that you can create and share a storyboard or larger report made up of individual dashboards, reports, and other views. Notifications can also export and send this content on a schedule.

Create a New Slideshow

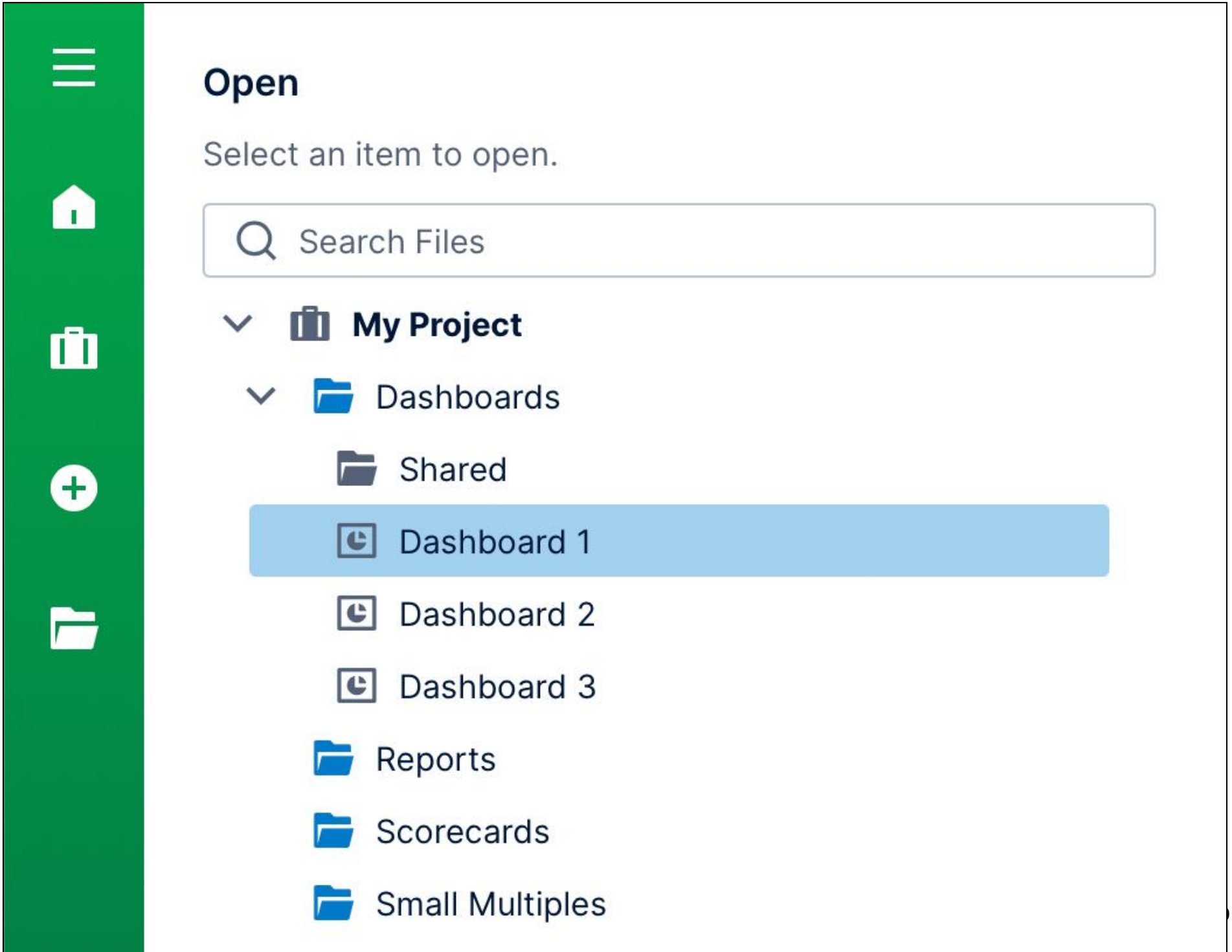
To create a slideshow, **Views** in the main menu, switch to **Slideshow** along the top, then click **Create**.

Add Views to the Slideshow

In the slideshow editor, click **Add Item**.

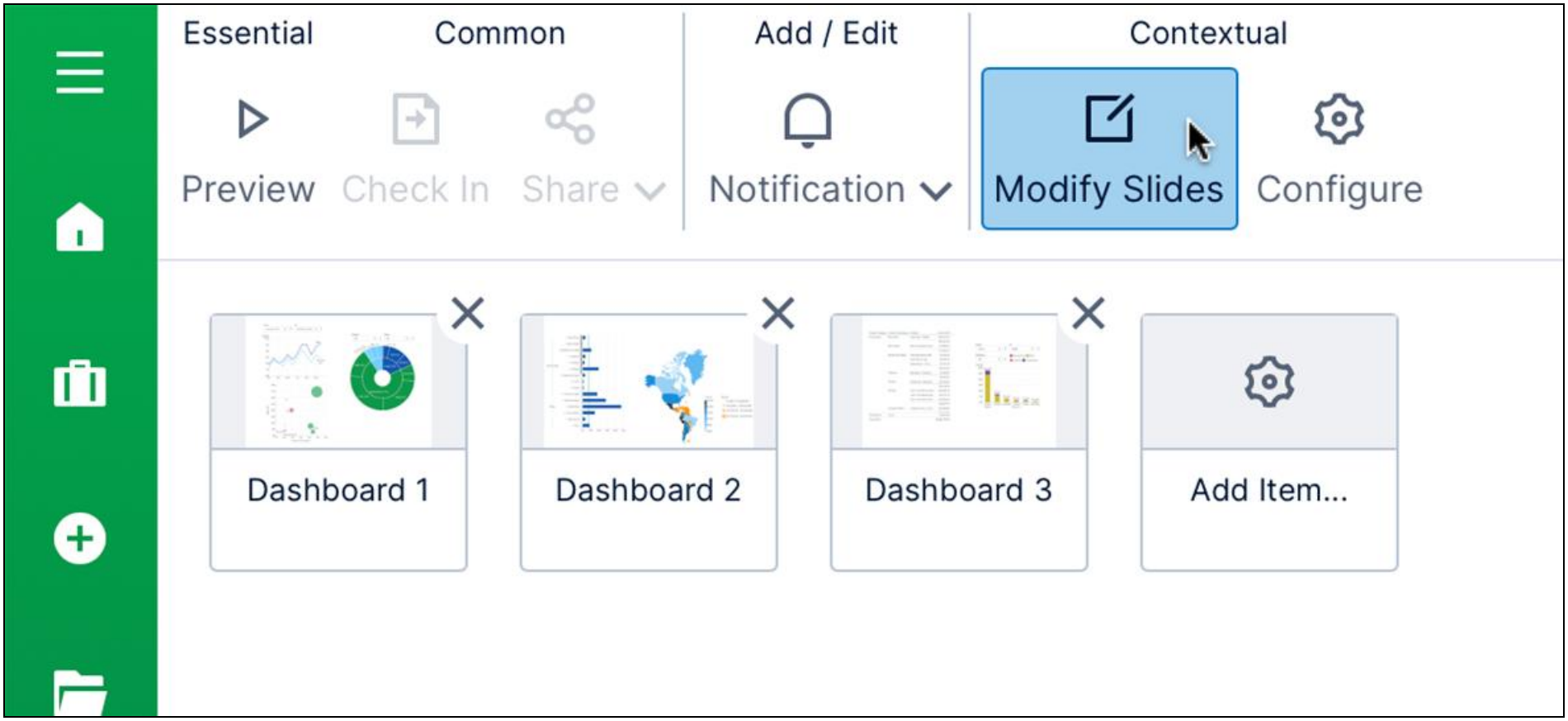


Select a dashboard or other view to be added to the slideshow.

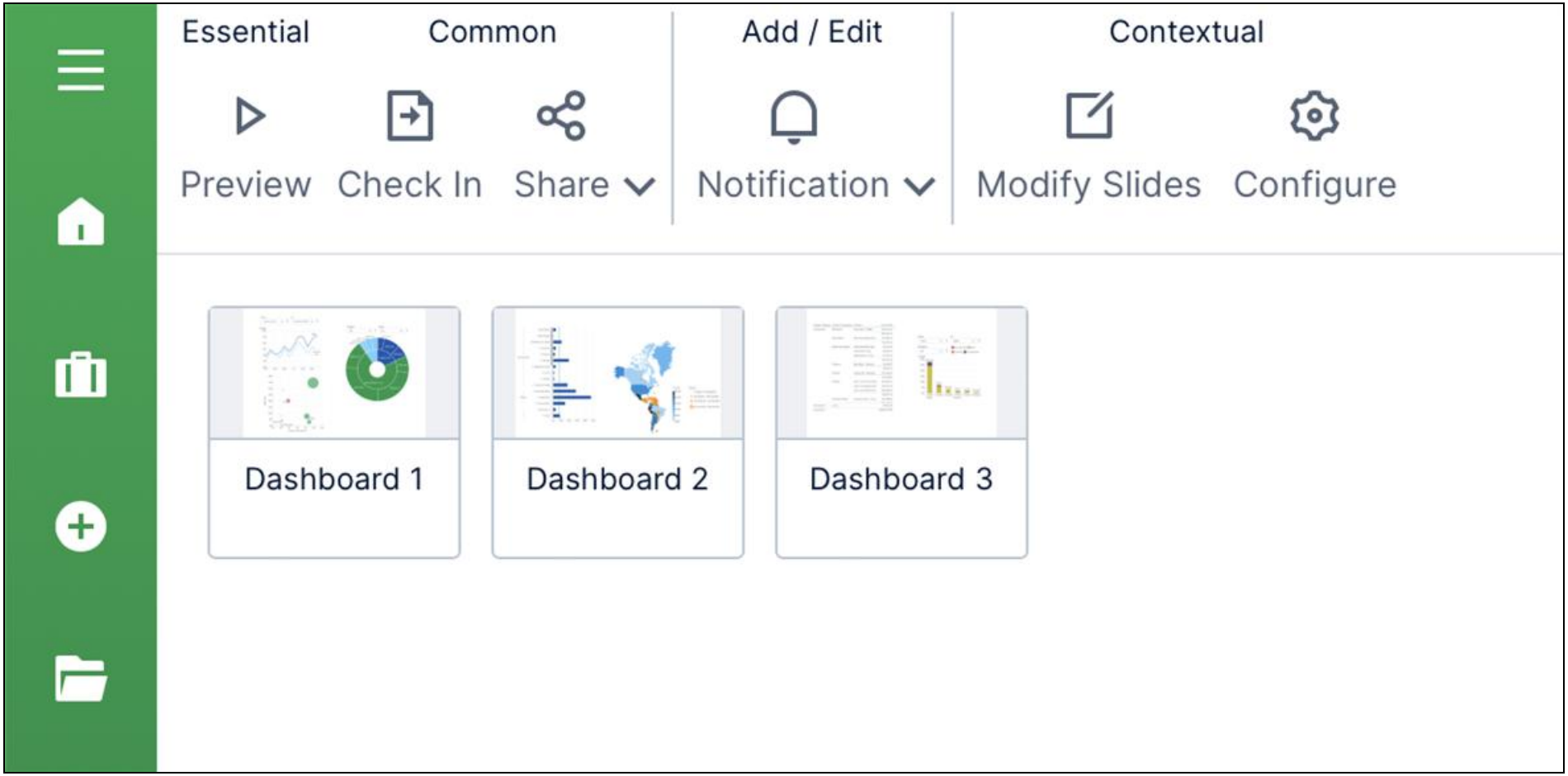

Open
Select an item to open.
Search Files
My Project
Dashboards
Shared
Dashboard 1
Dashboard 2
Dashboard 3
Reports
Scorecards
Small Multiples

Repeat to add additional views as slides to the slideshow.

You can reorder the slides by dragging them, or remove a slide by clicking the x icon. Click **Modify Slides** in the toolbar when you're finished to save your changes.



This is a toggle button, and you can click it again to go back to edit the slides.

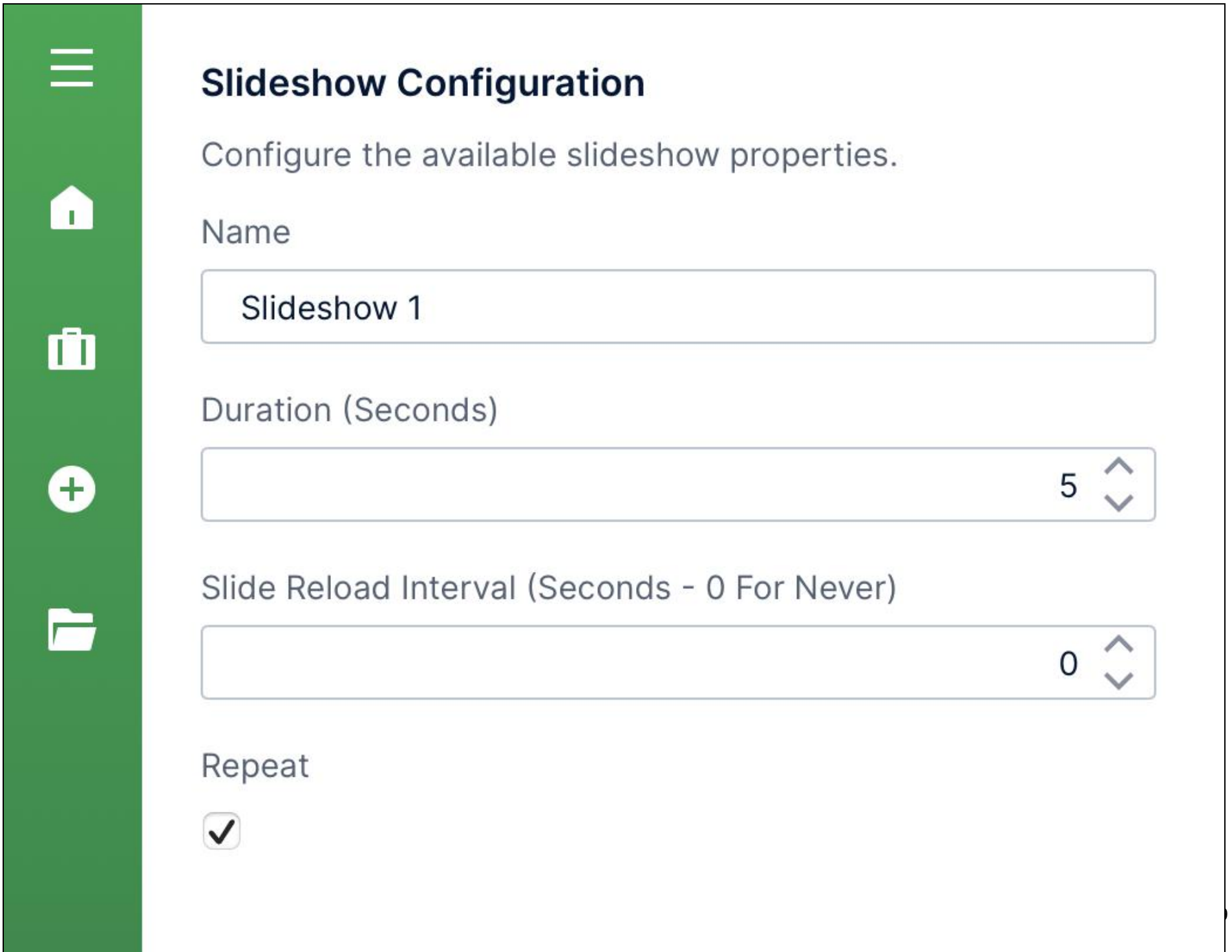


Configure Slideshow Options

When editing a slideshow, click **Configure** in the toolbar to set slideshow options such as:

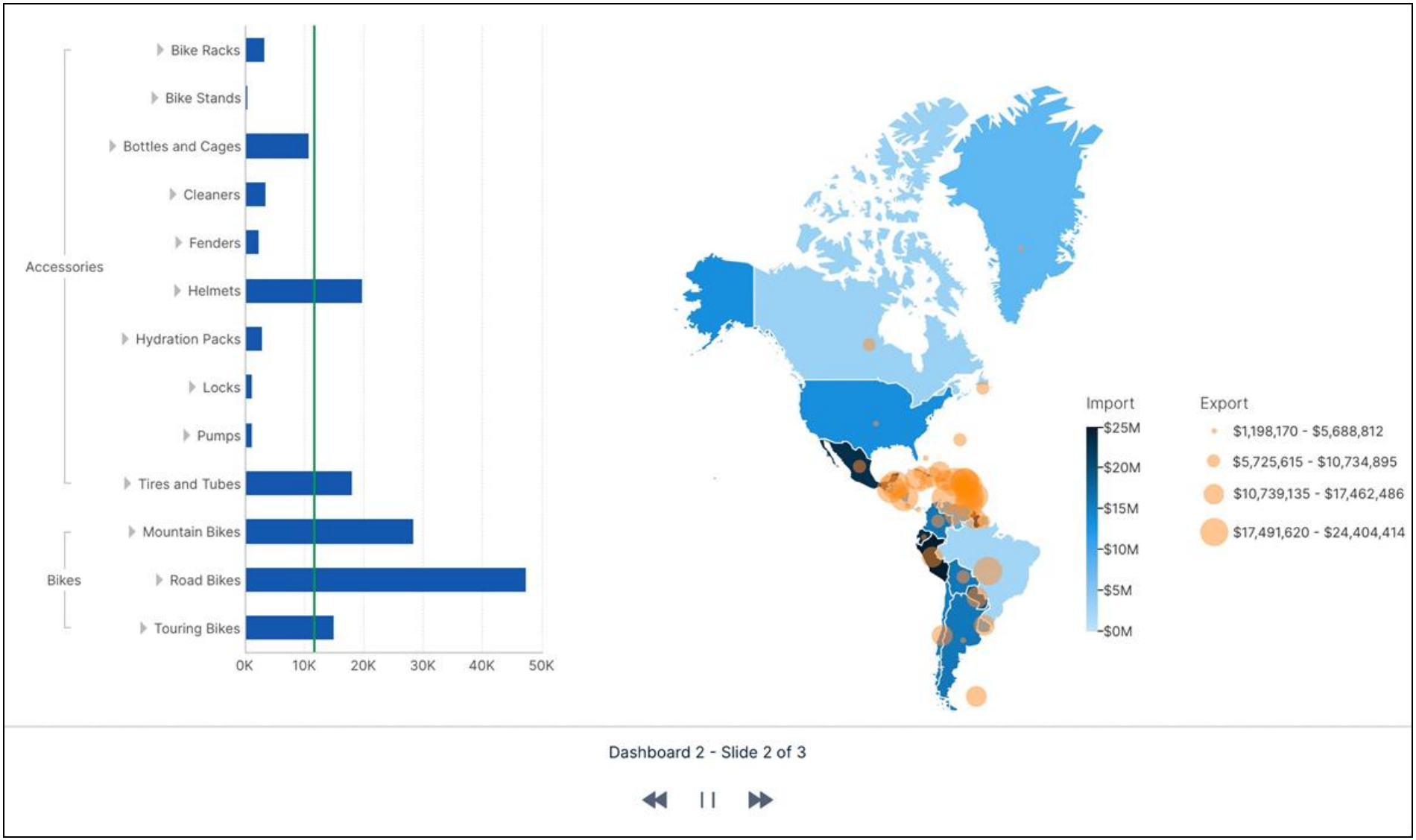
- **Name** of the slideshow
- **Duration** for each slide in seconds

- **Slide Reload Interval** in seconds, if you want each slide to also reload itself at some interval before moving to the next
- **Repeat** – loops through the slides continuously


Slideshow Configuration
Configure the available slideshow properties.
Name
Slideshow 1
Duration (Seconds)
5
Slide Reload Interval (Seconds - 0 For Never)
0
Repeat

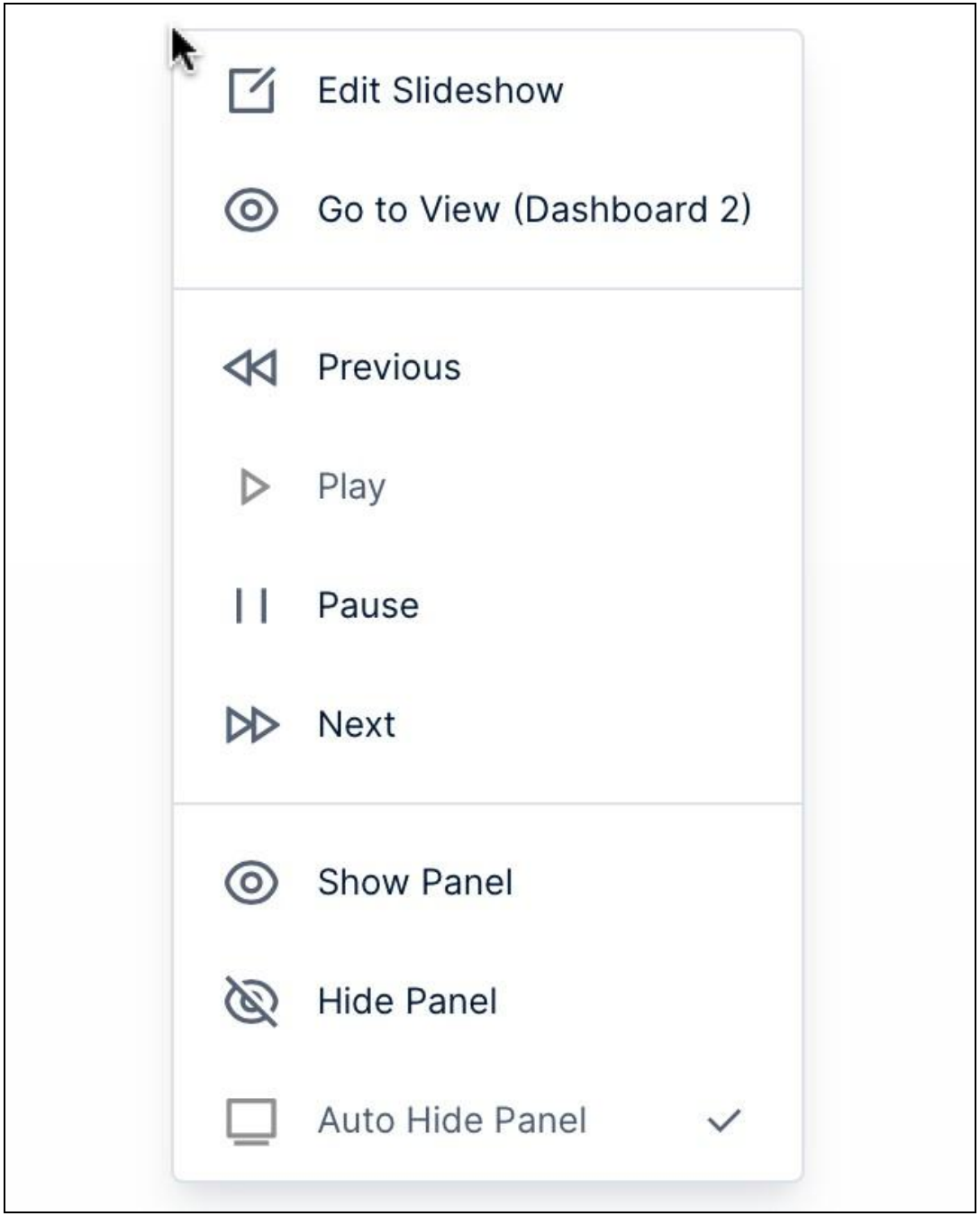
Preview and Play the Slideshow

When editing, click **Preview** in the toolbar to play the slideshow. This also exits slide editing mode and saves any changes you made.



A panel appears automatically by default when moving the mouse toward the bottom of the screen, containing convenient options for pausing or playing the slideshow, and previous next buttons.

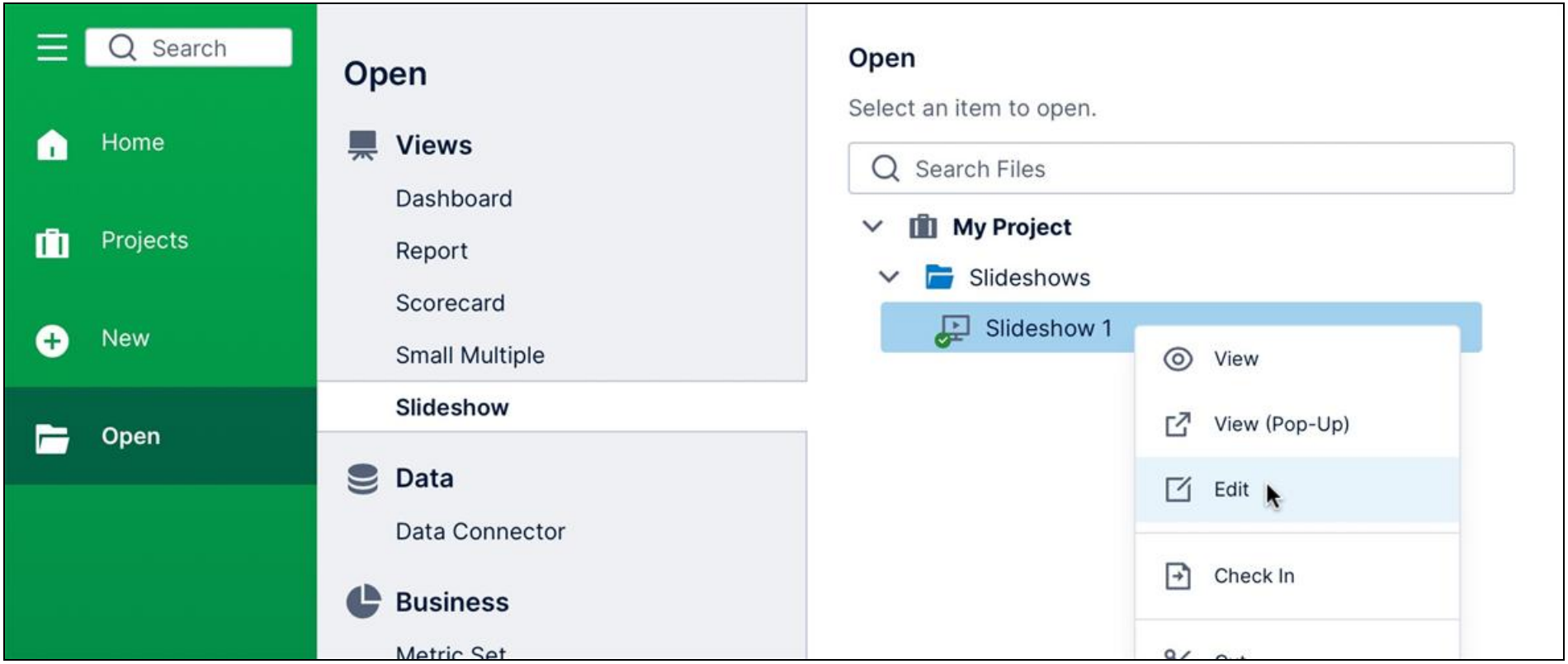
To access additional options, including for returning to the slideshow editor, right-click (or long-tap) the slideshow to access its context menu:


Edit Slideshow
Go to View (Dashboard 2)
Previous
Play
Pause
Next
Show Panel
Hide Panel
Auto Hide Panel

Edit a Slideshow

To edit an existing slideshow, open it first from the main menu. It will open in view (or presentation) mode by default, in which case you can right-click or long-tap to access the context menu and choose **Edit Slideshow** as shown in the previous section.

You can also open a slideshow from the main menu directly into edit mode like you can for other items, by right-clicking it (or long-tapping) from the *Open* dialog and choosing **Edit**.



If you want to add, remove, or reorder the slides, click **Modify Slides** in the toolbar if it's not already highlighted, and then you can follow the same steps shown earlier for a new slideshow.

Exporting and Notifications

You can access the **Share** and **Notification** toolbar buttons in the toolbar when the slideshow is open for editing. If the **Modify Slides** button is highlighted, click it first to save your changes and exit slide editing mode.

You can choose from the following export formats either to download or send as a notification:

- PDF – contains a page for each view.
- PowerPoint – a PPTX presentation file containing a slide for each view.
- Image – a ZIP file containing a PNG image for each view.

Reports are themselves often made up of multiple pages, in which case each of them will also make up multiple PDF pages, PowerPoint slides, or images within the exported file.

In Symphony v24.3 and later, PDF exports will include a table of contents accessible in most PDF viewers via a sidebar, for example, allowing you to click to navigate between the different items, similar to tables of contents in reports. A report's table of contents will be included below the top level of items.

For more information, see:

- Check In, Check Out and Auto Saving
- Share and Export Symphony Dashboards and Reports
- Symphony Notifications (Alerts)

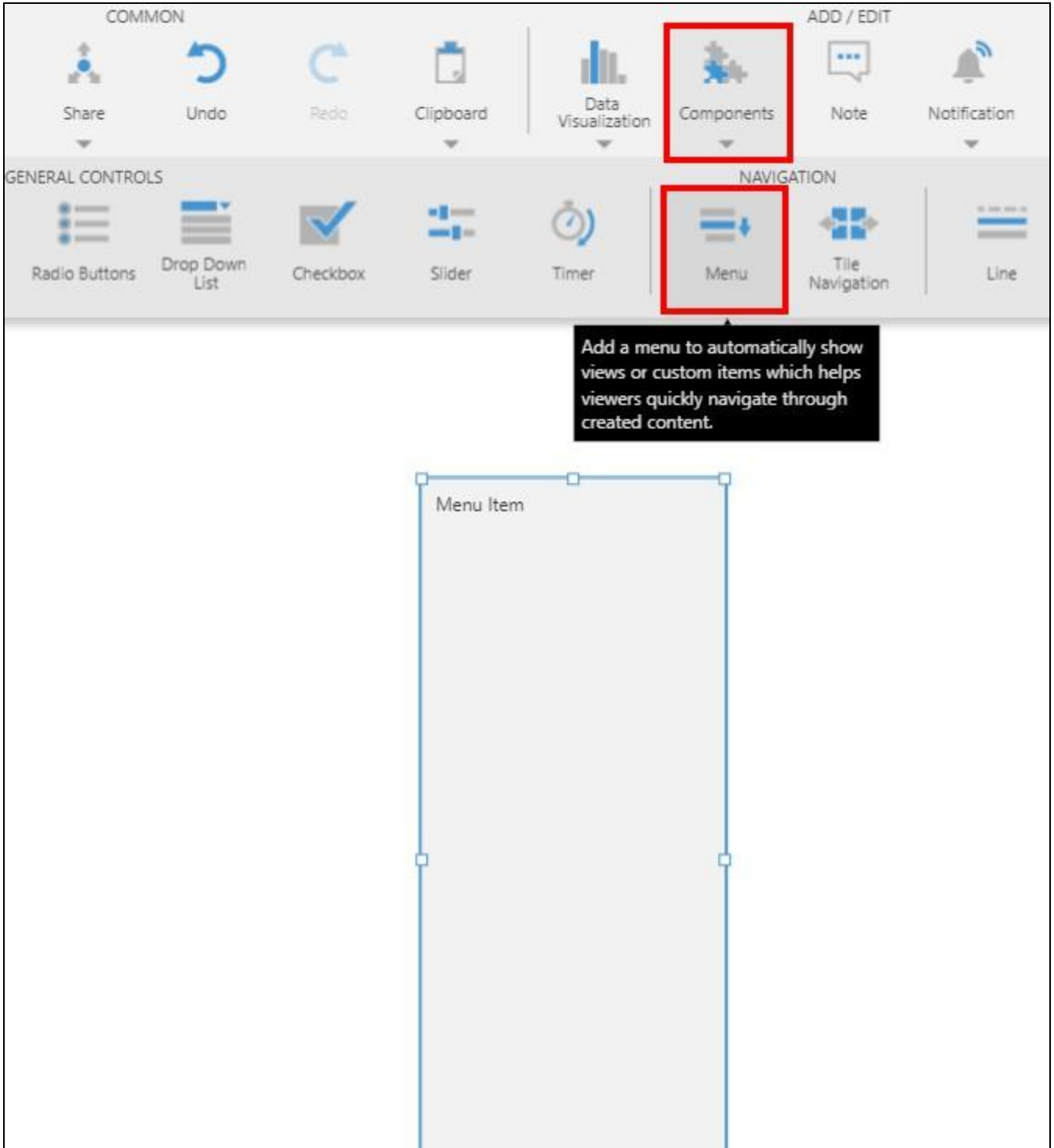
How To Set Up Menu Navigation

This applies to: *Managed Dashboards, Managed Reports*

The Menu component lays out a menu of items that users can select to navigate to content, expand submenus, and trigger other interactions. You can customize it a variety of ways to suit your design and layout needs.

Add a Menu Component

Go to the toolbar, click **Components**, and then click **Menu**.


COMMON
ADD / EDIT
Share
Undo
Redo
Clipboard
Data Visualization
Components
Note
Notification
GENERAL CONTROLS
NAVIGATION
Radio Buttons
Drop Down List
Checkbox
Slider
Timer
Menu
Tile Navigation
Line
Add a menu to automatically show views or custom items which helps viewers quickly navigate through created content.
Menu Item

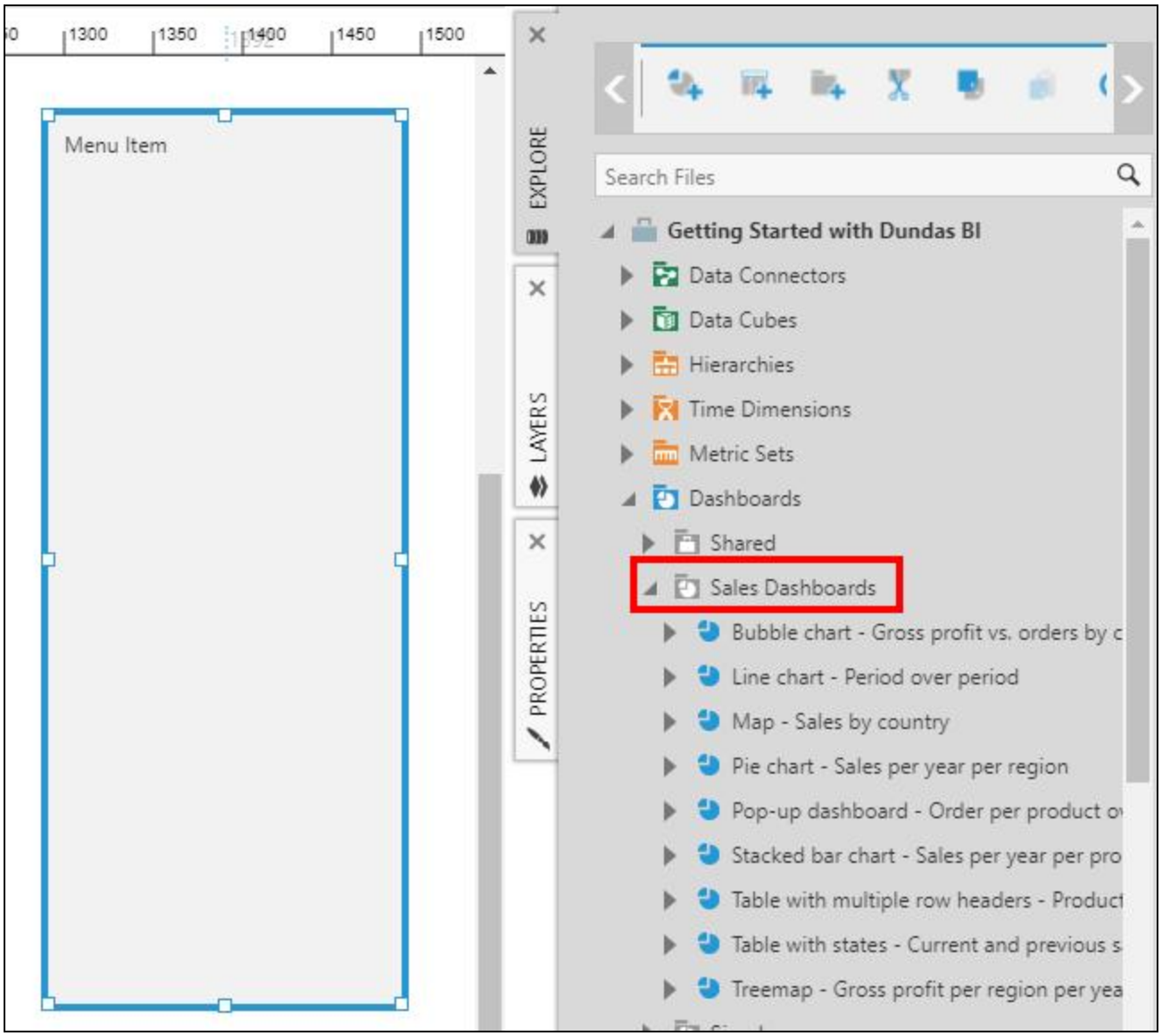
Add Menu Items

You can add items to the menu by dragging them directly from the **Explore** window or you can add them manually by specifying which items to include from the **Properties** window.

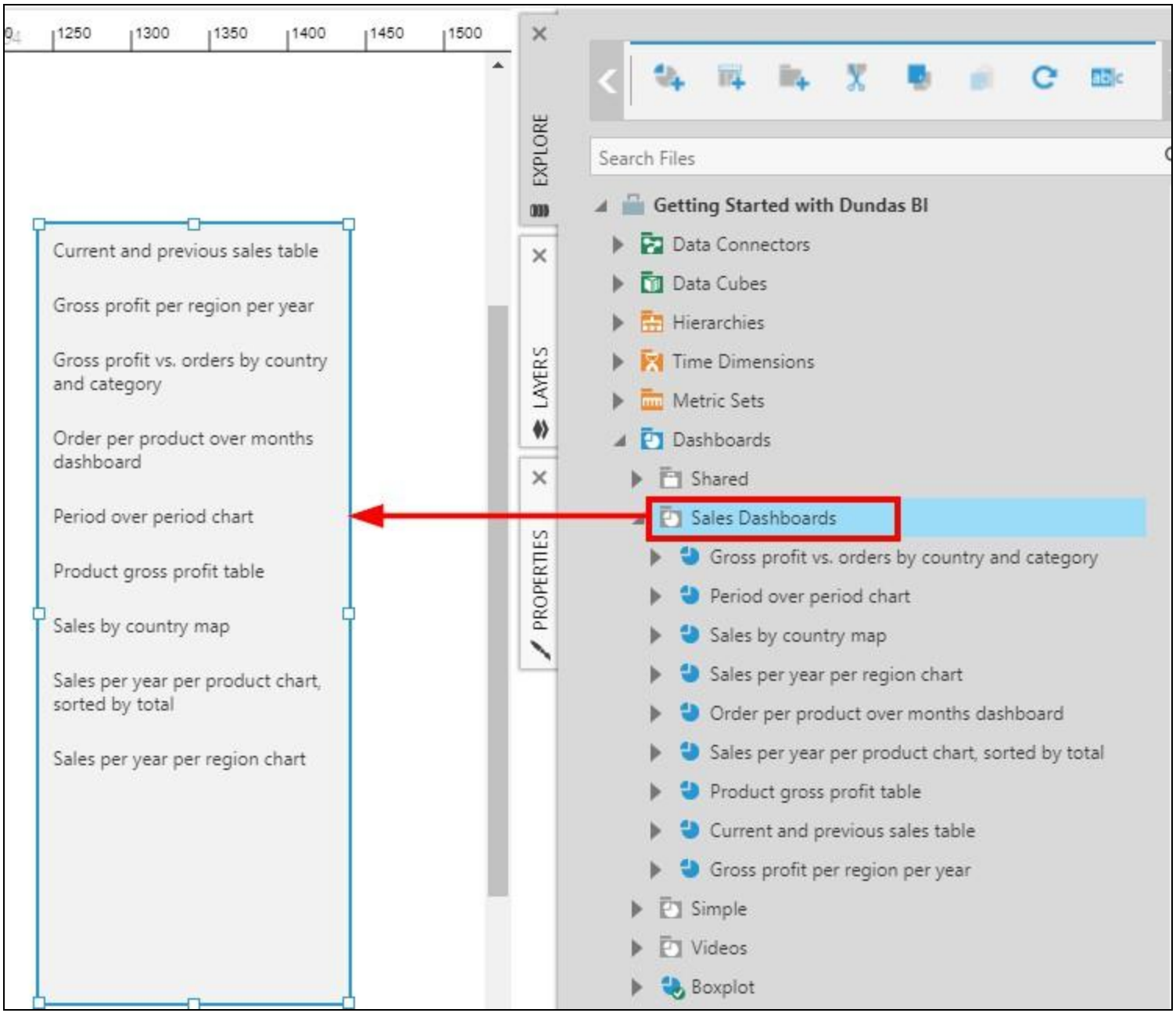
Drag Views or Folders

From the **Explore** window, drag a view or a folder of views onto the empty menu to populate the menu items. For this example we chose to drag a folder containing various views.

Important: Only one dragged folder is displayed at a time.



The views contained in the folder are listed as menu items in the menu component, and you can select to navigate to each one.


1250
1300
1350
1400
1450
1500
Current and previous sales table
Gross profit per region per year
Gross profit vs. orders by country and category
Order per product over months dashboard
Period over period chart
Product gross profit table
Sales by country map
Sales per year per product chart, sorted by total
Sales per year per region chart
EXPLORE
LAYERS
PROPERTIES
Search Files
Getting Started with Dundas BI
Data Connectors
Data Cubes
Hierarchies
Time Dimensions
Metric Sets
Dashboards
Shared
Sales Dashboards
Gross profit vs. orders by country and category
Period over period chart
Sales by country map
Sales per year per region chart
Order per product over months dashboard
Sales per year per product chart, sorted by total
Product gross profit table
Current and previous sales table
Gross profit per region per year
Simple
Videos
Boxplot

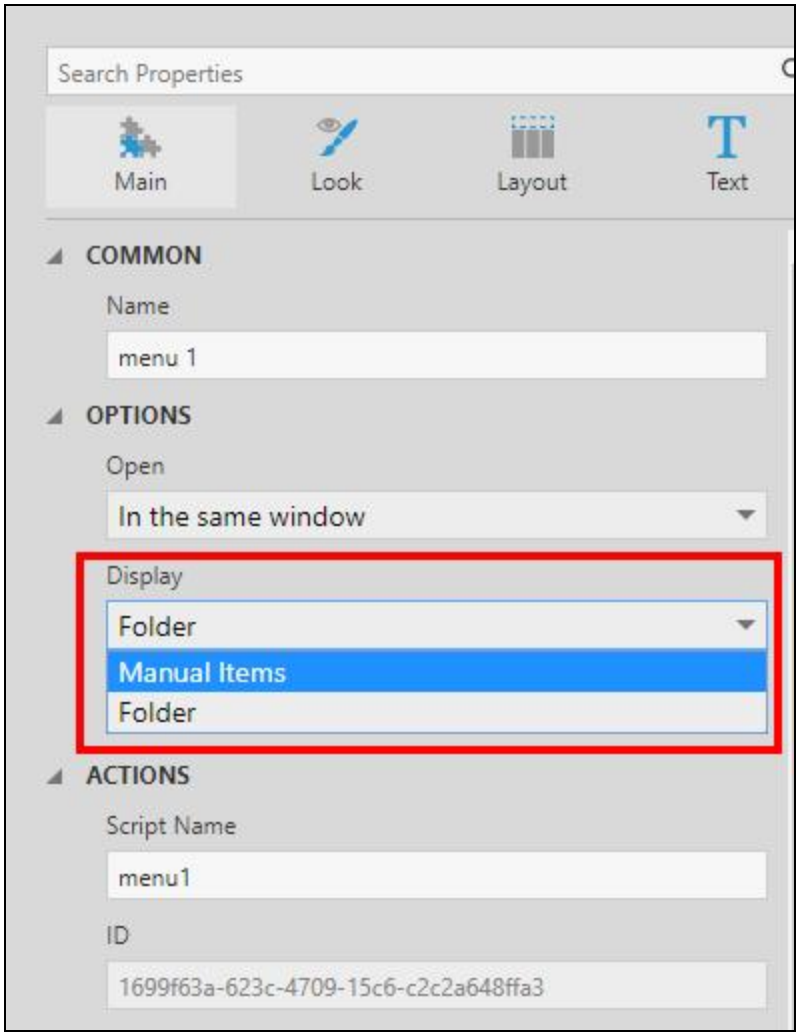
Add Menu Items Manually

Alternatively, you can use the **Properties** window to add individual items manually or select a folder to populate the menu.

Enter Manual Items

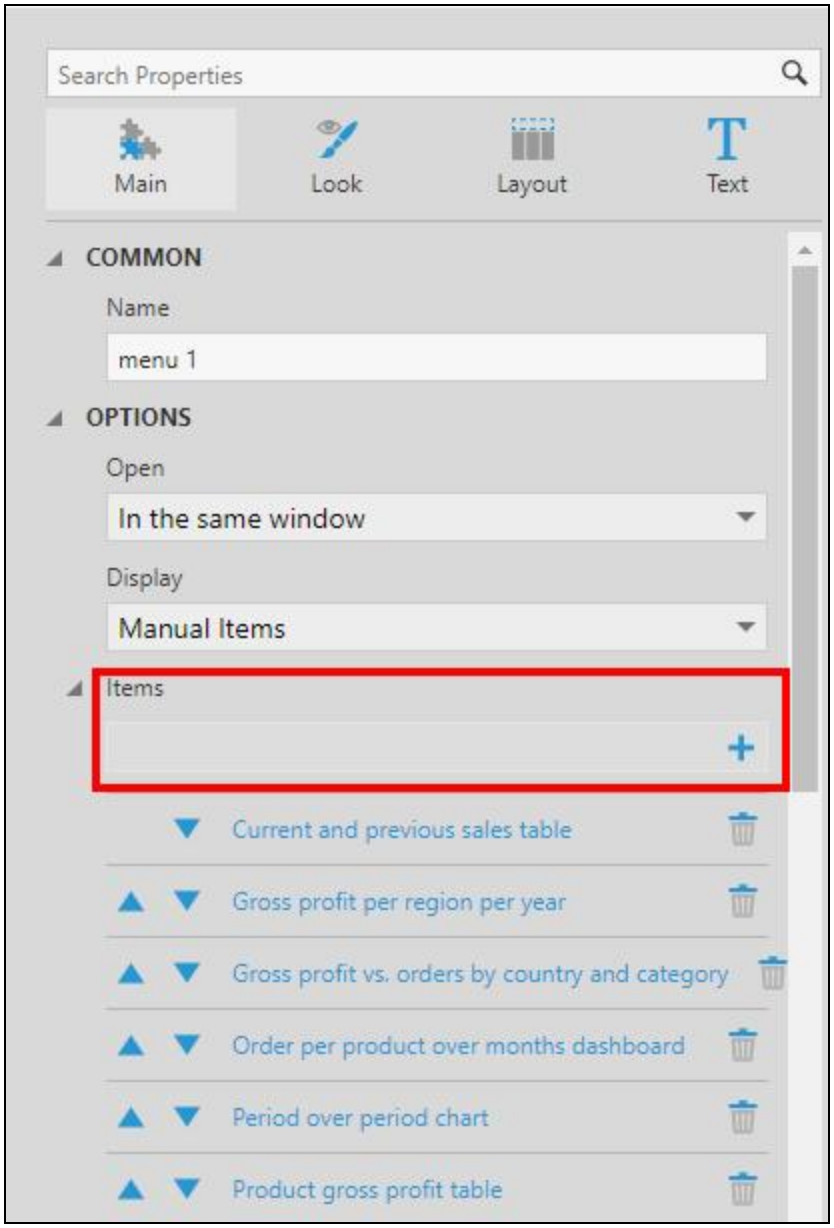
Go to the **Properties** window and select the menu component on the canvas.

Set the `Display` property to **Manual Items**.



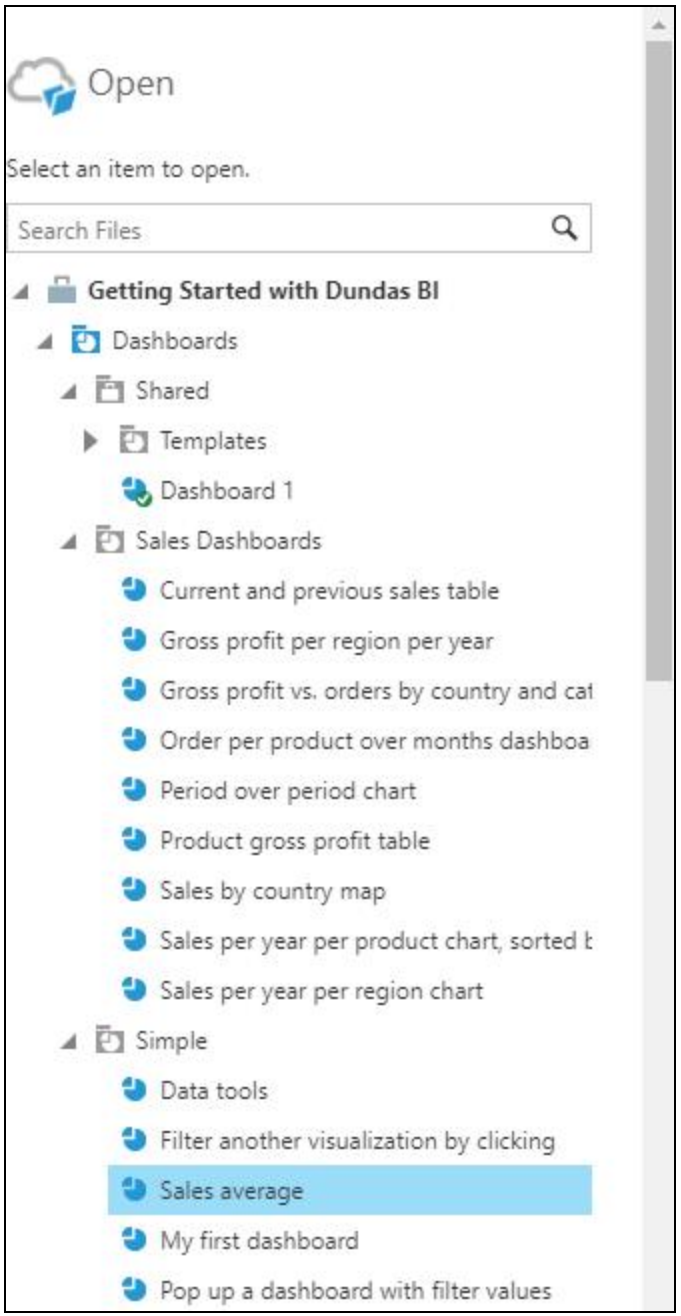
In our example, we have some existing `Items` already listed below what we added by dragging.

Select the **+** button to add a new item, then select the item to edit it.



Typically, menu items have a navigation target set up on them directly. Select the **+** button under `Navigation Target` to set one, then select the **Navigation Target** link to choose the target.

Select a view to navigate to. For this example, we selected the **Sales average** view.

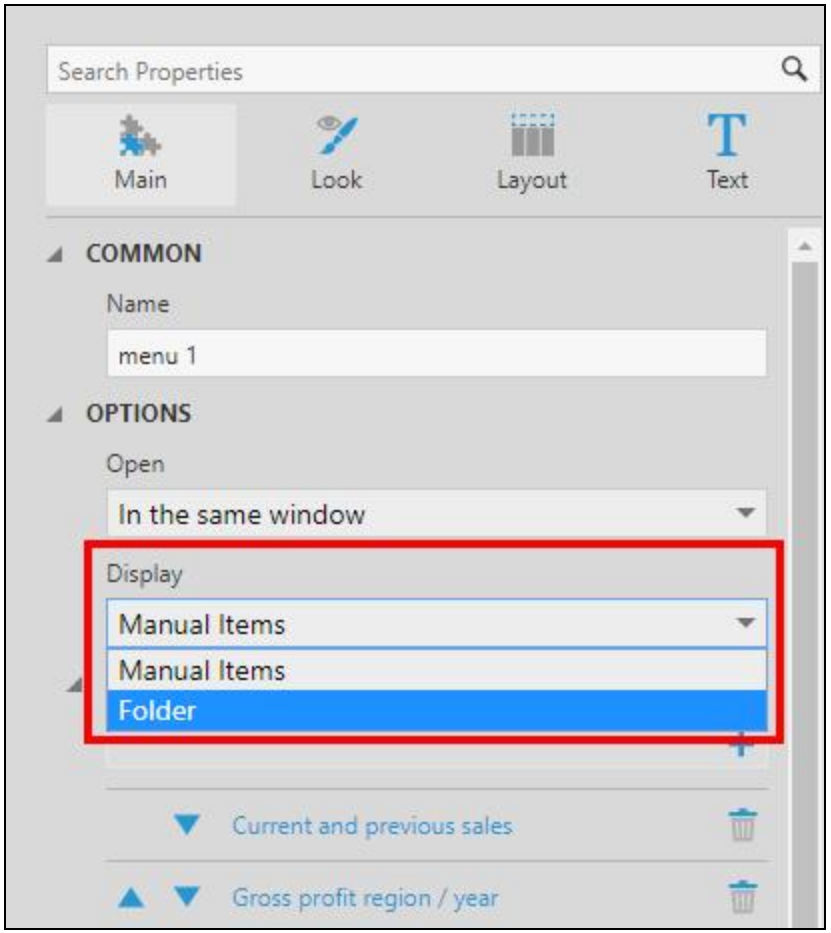

Open
Select an item to open.
Search Files
Getting Started with Dundas BI
Dashboards
Shared
Templates
Dashboard 1
Sales Dashboards
Current and previous sales table
Gross profit per region per year
Gross profit vs. orders by country and cat
Order per product over months dashboa
Period over period chart
Product gross profit table
Sales by country map
Sales per year per product chart, sorted b
Sales per year per region chart
Simple
Data tools
Filter another visualization by clicking
Sales average
My first dashboard
Pop up a dashboard with filter values

Select **Open** at the bottom of the dialog. You can use the properties in the **Text** tab to edit the display text of this item.

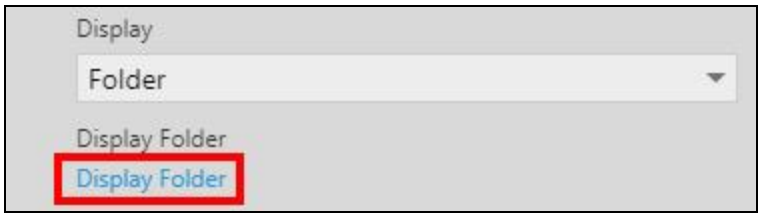
Select a Folder

Add items dynamically to the menu component by selecting an entire folder to display in the menu using the **Folder** option. When you use this option, any items added or removed from the folder will reflect in the list of menu items.

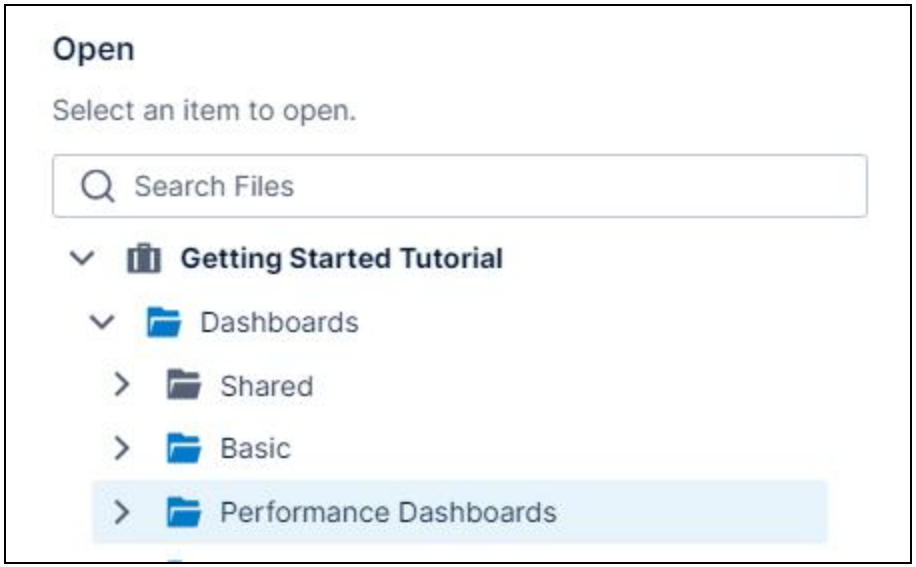
With the menu component selected, go to the **Properties** window and set the `Display` property to **Folder**.



Click to select the **Display Folder** link.



Select a folder from the **Open dialog** to add to the menu. For this example, we selected the **Performance Dashboards** folder.

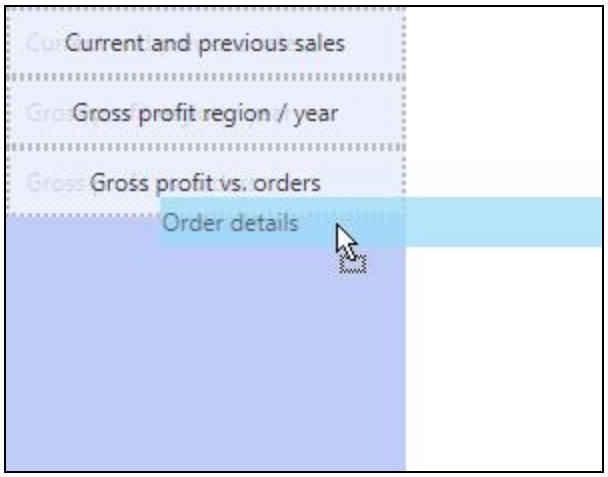


Select **Open** at the bottom of the dialog. All the views from the folder are displayed as items in the menu.

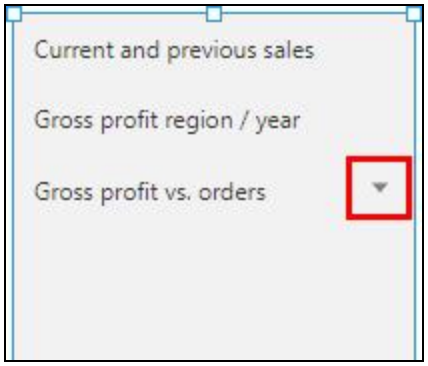
Add Submenu Items

You can add additional levels of menu items or **submenus** for access under another menu item.

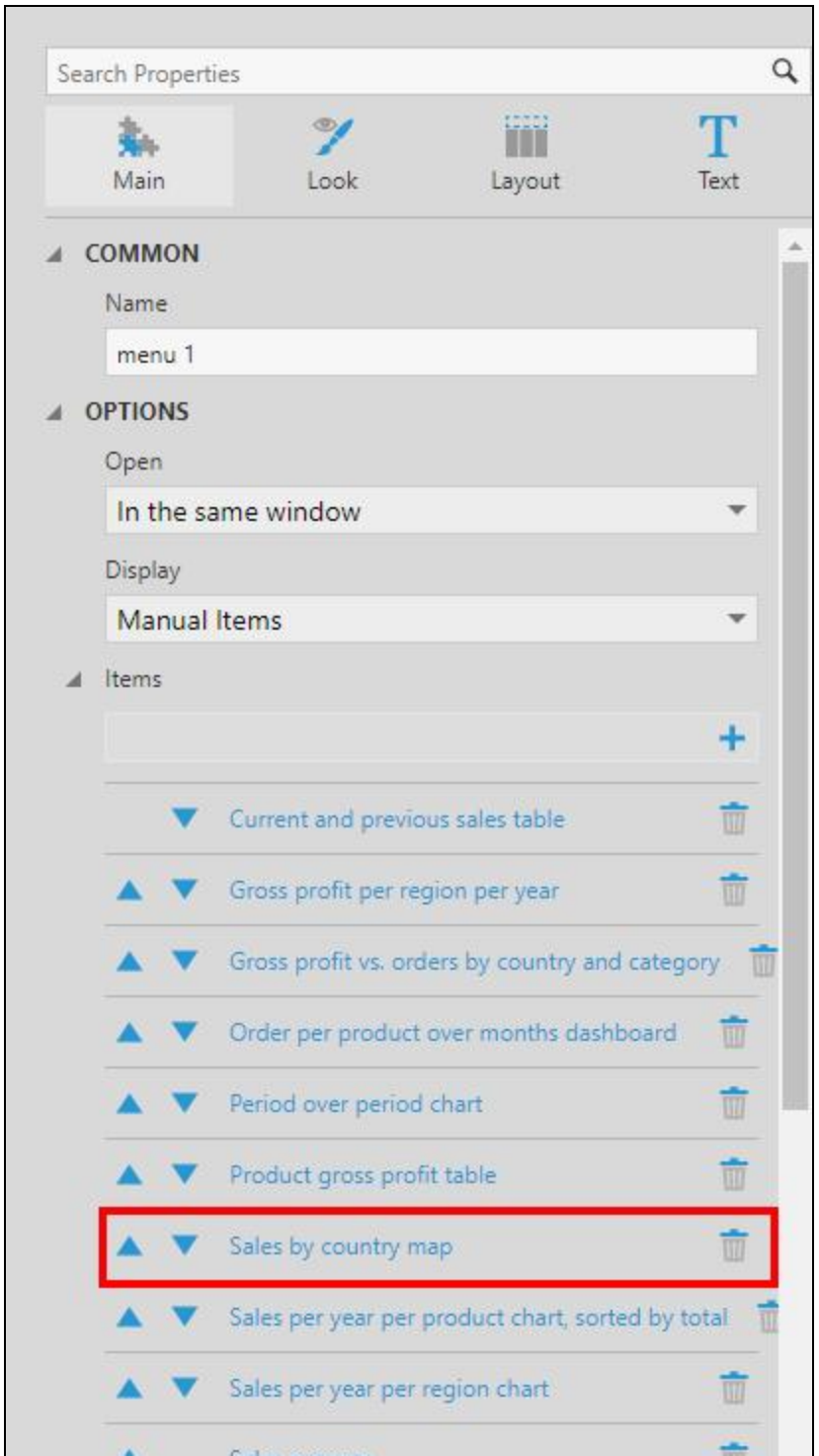
To add an item under an existing menu item, drag a file from the **Explore** window and drop it onto the existing item.



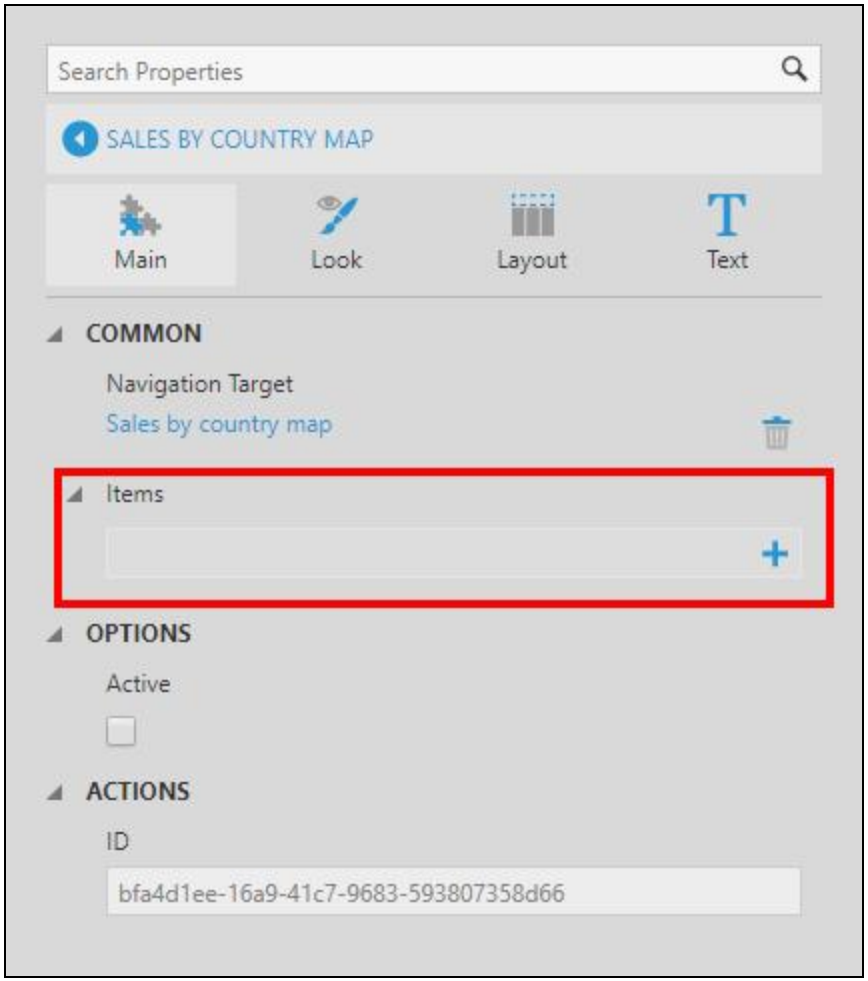
An arrow is displayed to the right of the menu item, called an indicator icon, indicating that it can be expanded when viewing. This icon can be customized in the menu properties.



You can also add submenu items manually in the **Properties** window. Select one of the items in the menu's `Items` property to edit it, or click the item in the menu directly.


Search Properties
Main
Look
Layout
Text
COMMON
Name
menu 1
OPTIONS
Open
In the same window
Display
Manual Items
Items
Current and previous sales table
Gross profit per region per year
Gross profit vs. orders by country and category
Order per product over months dashboard
Period over period chart
Product gross profit table
Sales by country map
Sales per year per product chart, sorted by total
Sales per year per region chart

Now click to add an item to this `Items` property.



Click on the new item to edit its settings.

This submenu item can be set up manually just like the top level menu item shown earlier. You can also add another level of submenu items below this item.

Customization Properties

You can customize the appearance of the menu and its items in the **Properties** window.

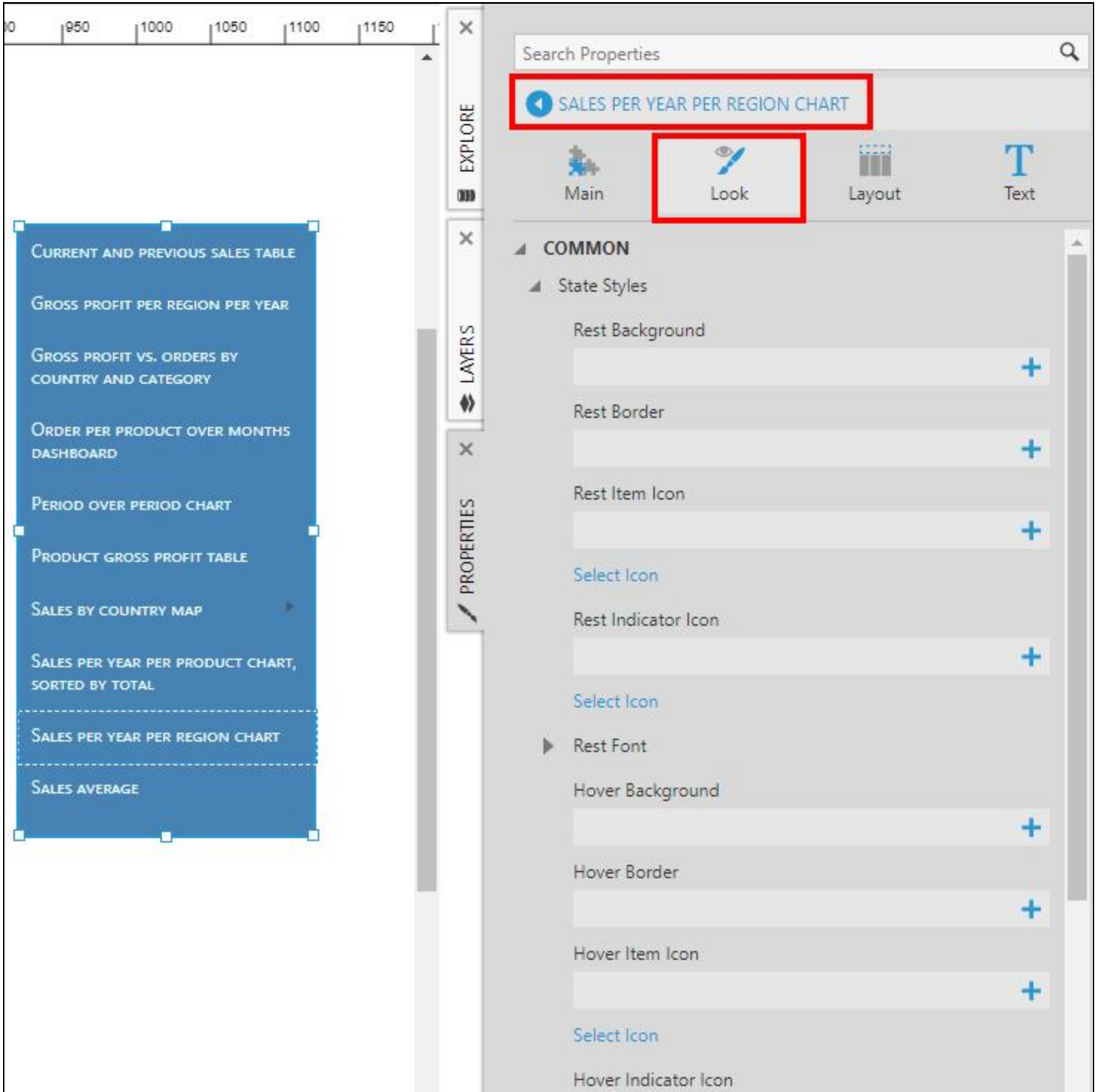
Style Menu Items

Style Individual Menu Items

You can choose to style each menu item individually. In this case, the menu's `Display` property should be set to **Manual Items**.

You can click to select a menu item directly on the menu component itself, or click to edit an item under the `Items` property.

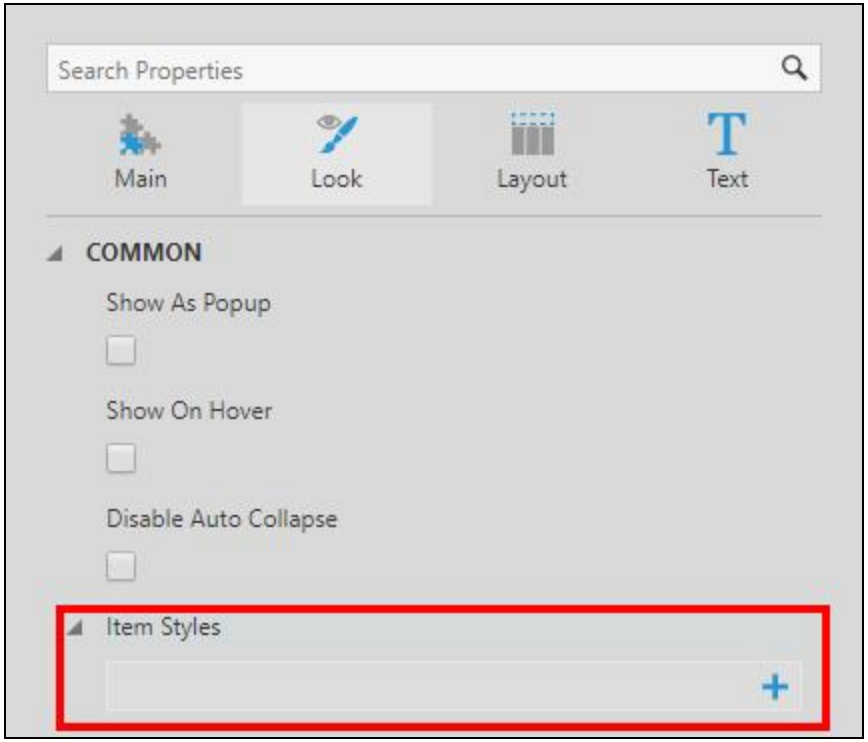
In the **Look** tab, you can change the background, border or icon when the menu item is in various states during viewing such as at rest, when hovered over or when active.


950
1000
1050
1100
1150
CURRENT AND PREVIOUS SALES TABLE
GROSS PROFIT PER REGION PER YEAR
GROSS PROFIT VS. ORDERS BY COUNTRY AND CATEGORY
ORDER PER PRODUCT OVER MONTHS DASHBOARD
PERIOD OVER PERIOD CHART
PRODUCT GROSS PROFIT TABLE
SALES BY COUNTRY MAP
SALES PER YEAR PER PRODUCT CHART, SORTED BY TOTAL
SALES PER YEAR PER REGION CHART
SALES AVERAGE
EXPLORE
LAYERS
PROPERTIES
Search Properties
SALES PER YEAR PER REGION CHART
Main
Look
Layout
Text
COMMON
State Styles
Rest Background
Rest Border
Rest Item Icon
Select Icon
Rest Indicator Icon
Select Icon
Rest Font
Hover Background
Hover Border
Hover Item Icon
Select Icon
Hover Indicator Icon

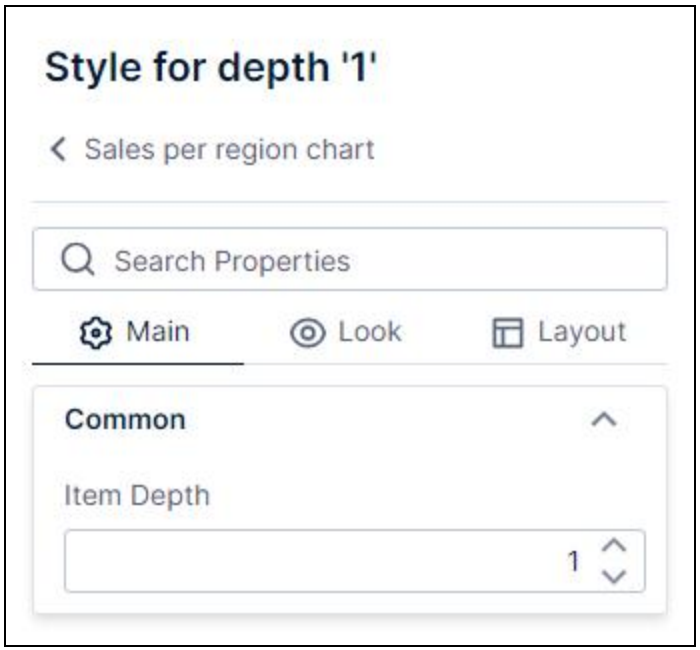
Level Styles

You can also specify styles for all the items at once in each level of the menu. This option will work if you are dynamically populating items from the file system. If you have submenus, you can set up a distinct style for each level.

To add a new style, go to the menu's properties, click on the **Look** tab, and click **+** under `Item Styles`.



Select the new item to edit it. If you have multiple levels of items (or submenus), increase the `Item Depth` property to the submenu level you want to customize. For example, if set to **1**, items displayed directly under a top-level menu item will be styled.

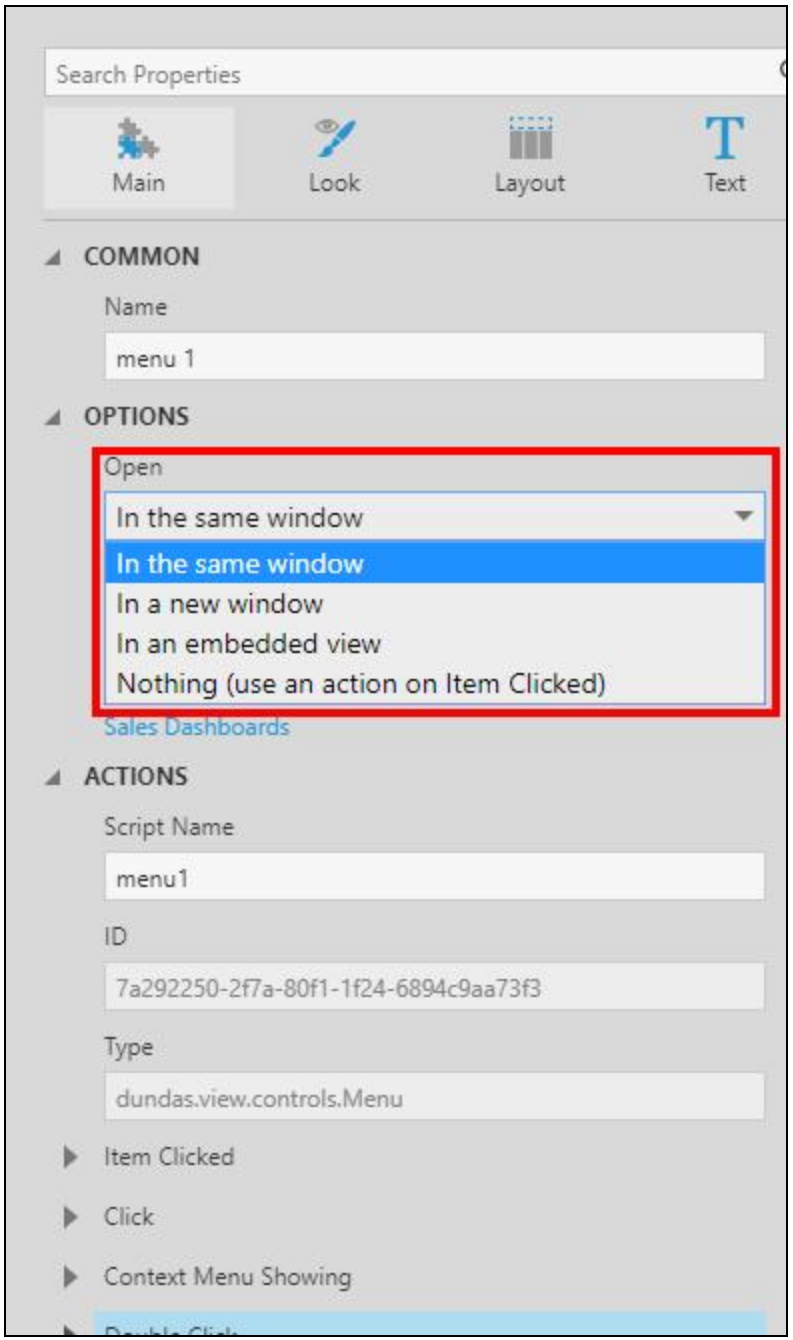


You can add more than one item style for the same item depth. In this case, the styles act as a palette, for example if you wanted to alternate the items between two background colors.

Open

Use the menu's `Open` property to specify what happens or how the navigation works when a menu item is clicked. The available options are:

- Open the clicked item in the same window
- Open the clicked item in a new window
- Open the clicked item in an embedded view (a view container elsewhere on the canvas)
- Invoke a custom interaction (Nothing option)


Search Properties
Main
Look
Layout
Text
COMMON
Name
menu 1
OPTIONS
Open
In the same window
In the same window
In a new window
In an embedded view
Nothing (use an action on Item Clicked)
Sales Dashboards
ACTIONS
Script Name
menu1
ID
7a292250-2f7a-80f1-1f24-6894c9aa73f3
Type
dundas.view.controls.Menu
Item Clicked
Click
Context Menu Showing

If you choose the **Nothing** option, you can add custom actions for each item instead as described in a later section.

Submenu Properties

You can use the following properties of the menu in the **Look** tab to customize how submenus are displayed when viewed:

- **Show As Popup** - Enable to display submenus as a popup, otherwise they are expanded below or to the right of the parent item.
- **Show On Hover** - Displays submenus by hovering over the parent item rather than clicking. When enabled, the Hover Delay property can be set in milliseconds.
- **Disable Auto Collapse** - Selecting this option allows multiple menu items to be expanded at the same time.

Horizontal Layout

In the **Layout** tab, under Common, select the menu's `Horizontal Layout` property to display the items side-by-side horizontally rather than in a vertical column.

Set Menu Items to Active

If needed, you can use the `Active` property on an individual menu item to ensure that a menu item is initially in the **Active** state when the dashboard is loaded during viewing. If this item contains submenu items, this expands the submenu.

Icons

You can add your own icons to identify each menu item and/or to indicate a submenu, for each of the various states of the menu items:

- Rest
- Active
- Hover

Add Item Icons

To add an icon to the left of the item's text, select the menu item, and go to the **Look** tab in the **Properties** window. Select **+** under the `Rest Item Icon` property, for example, to set it.

To use an image uploaded to Symphony, click the **Select Icon** link, select an image in the **Open** dialog, then click **Submit**.

The selected icon is added to the left of the text.

The selected icon is added to the left of the text.

You can add different icons for the other states of the menu item, or allow the rest icon to display for the other states.

Add Indicator Icons

You can add your own icon to the right of the text to indicate that a menu item can be expanded to show submenu items rather than use the default indicator.

Select the menu item, go to the **Look** tab in the Properties window, and click the **+** under the `Rest Indicator Icon` property, for example.

To use an image uploaded to Symphony, select the **Select Icon** link, browse and select an icon in the Open dialog, then select **Submit**.

The selected icon is added to the right of the text.

When you have a custom indicator icon set, you can enable the `Always Show Indicator Icon` property in the **Layout** tab if you want it to appear even if there are no submenu items.

Change Icon Size

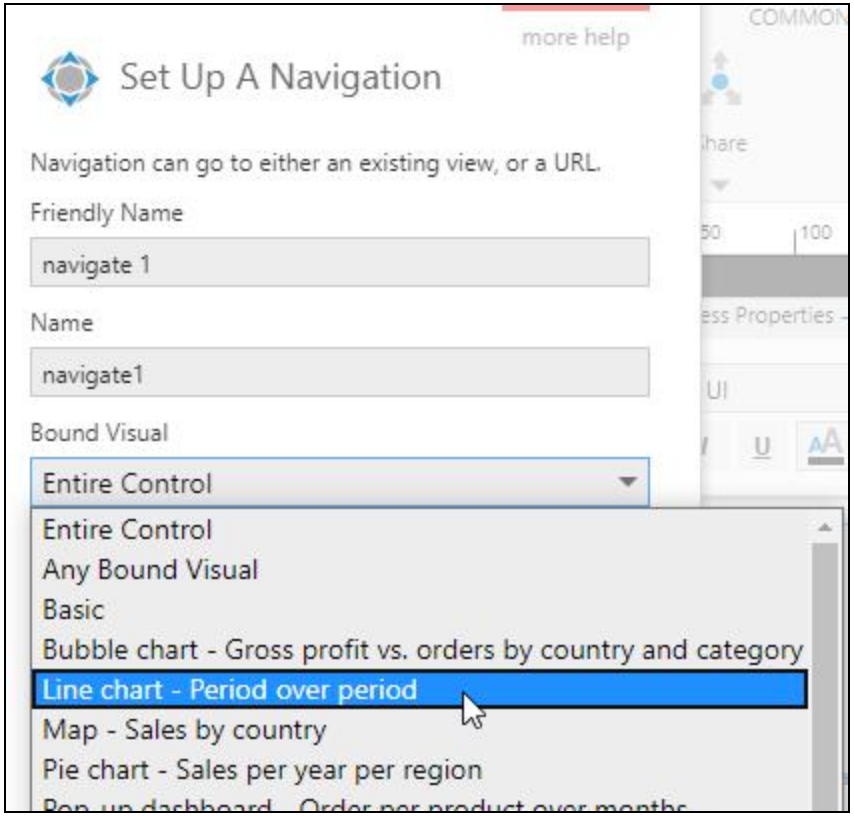
The `Icon Size` property in the **Layout** tab allows you to set the icon size for both the item and the indicator icons. The size specified applies to the height and width of the icons.

Custom Interactions

You can set up your own interactions for menu items rather than set a navigation target on them directly. The menu's Display property should be set to Manual Items for this to work.

For example, go to the **Properties** window for the menu and select to expand the **Item Clicked** actions. Select the menu icon on the right to choose which type of action to add, such as **Navigate** or **Change Layer**.

You would typically need to add a separate action for each item. Select an action to edit it, then set the **Bound Visual** setting to correspond to a specific menu item so that each one does something different.



How To Set Up Tile Navigation

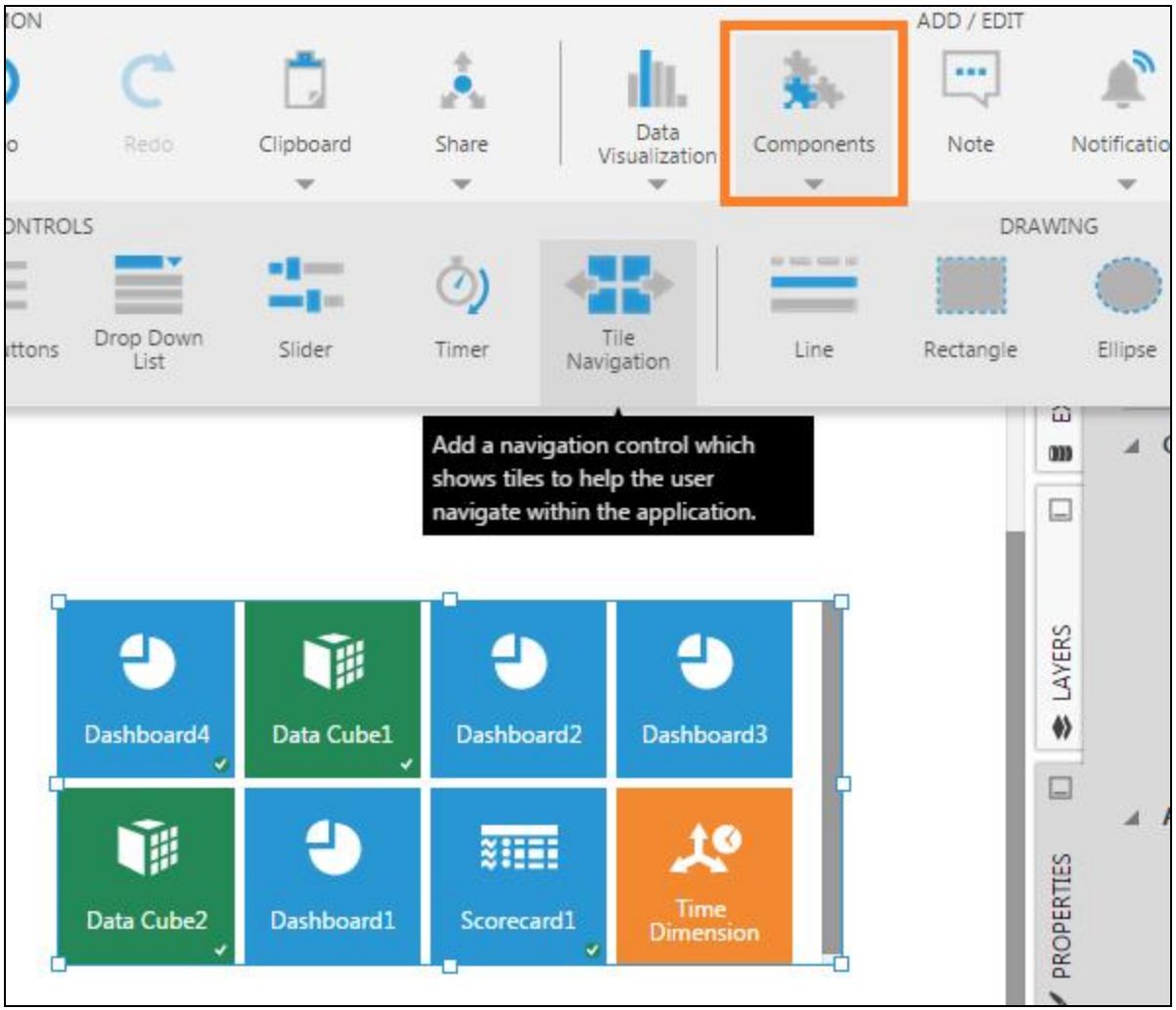
This applies to: *Managed Dashboards, Managed Reports*

Tile Navigation is a component which you can add to your dashboard canvas at design-time. You can use it to create a dashboard solely for navigation, or simply add a navigation bar to an existing dashboard that has charts or other content.

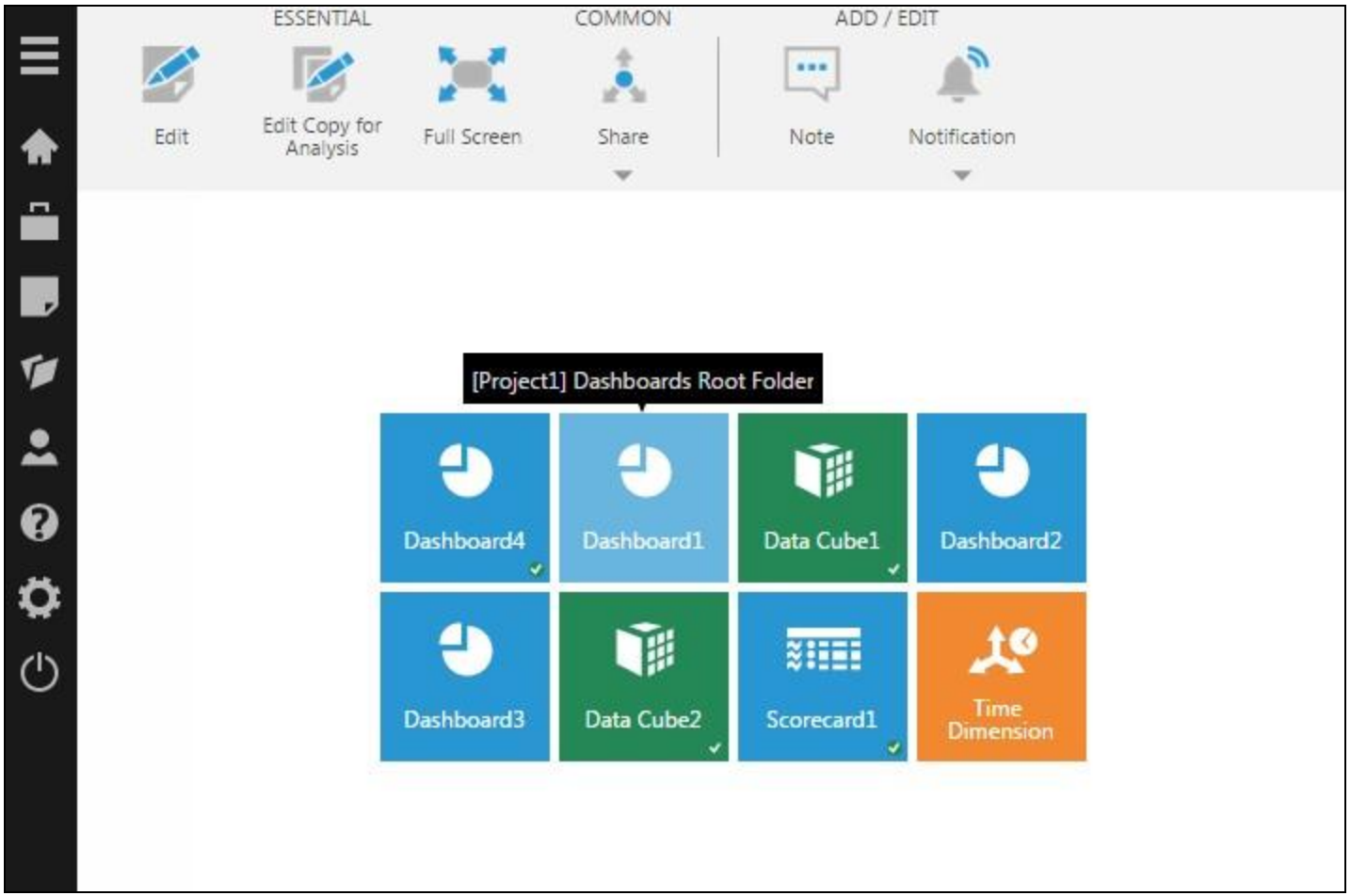
Users navigate with this component by selecting tiles like the ones on the home screen, which can similarly display thumbnail previews of views such as dashboards that have been checked in.

Add a Tile Navigation Component

Go to the toolbar, click **Components**, and then click **Tile Navigation**. You'll see a block of tiles like the ones on the home screen. By default, the tiles correspond to your recently viewed items.



Switch to **View** mode to use the component. Select a tile to navigate to another dashboard, for example.



Properties

By default, the Tile Navigation component is configured to show recently viewed items. You can change this as well as customize the appearance of the tiles from the **Properties** window for the component.

Open

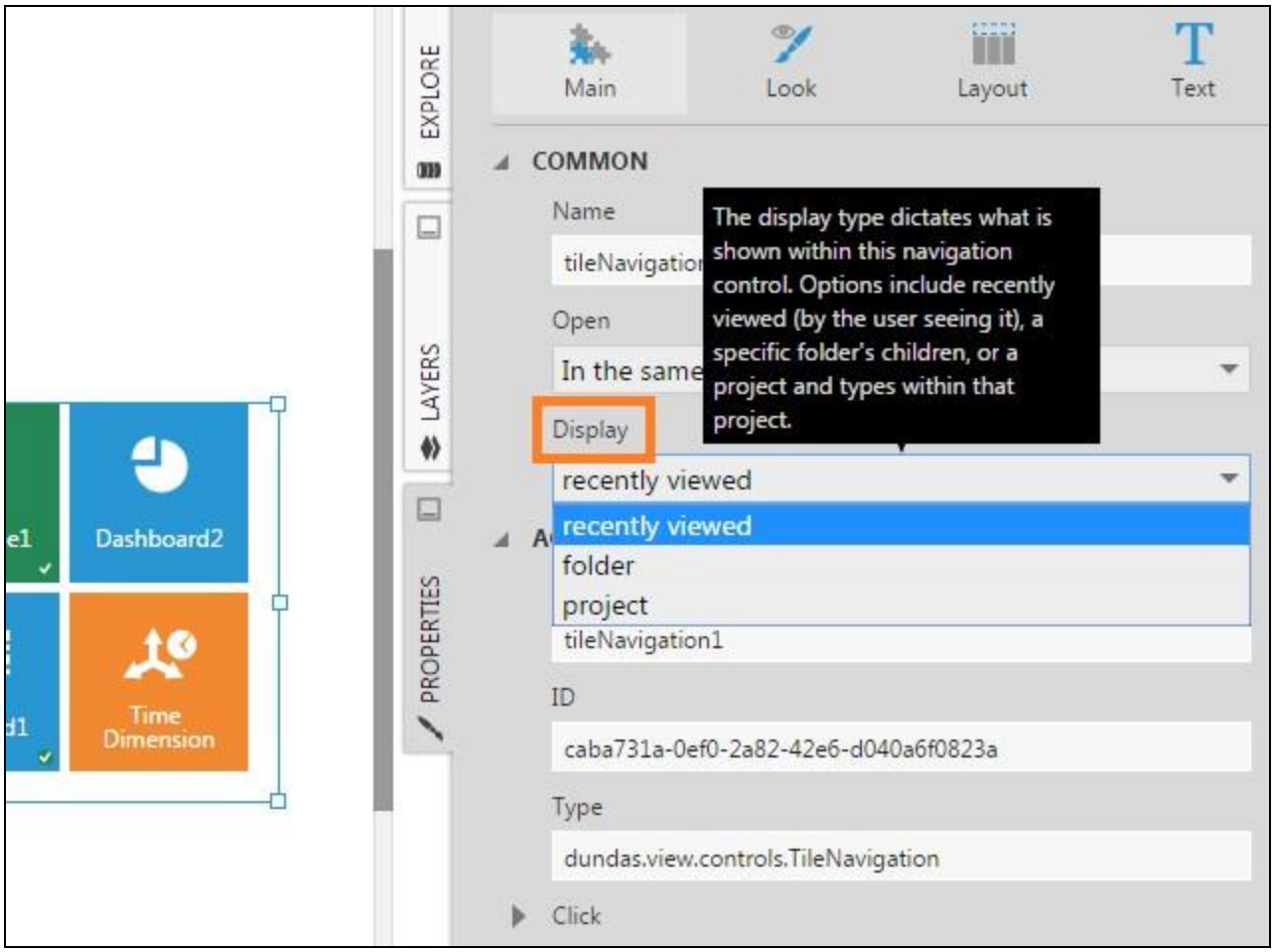
Use the `Open` property to specify what happens (or where to navigate to) when a tile is clicked. The available options are:

- Open the clicked item **in the same window** (selected by default)
- Open the clicked item **in a new window**
- Open the clicked item **in an embedded view** (i.e., a view container elsewhere on the canvas)
- Invoke a **Tile Clicked** script action (**Nowhere** option)

Display

Use the **Display** property to specify what tiles to display. The available options are:

- Display tiles corresponding to the list of recently viewed items
- Display tiles corresponding to the child items of a selected folder
- Display tiles corresponding to items in a selected project with a certain file type



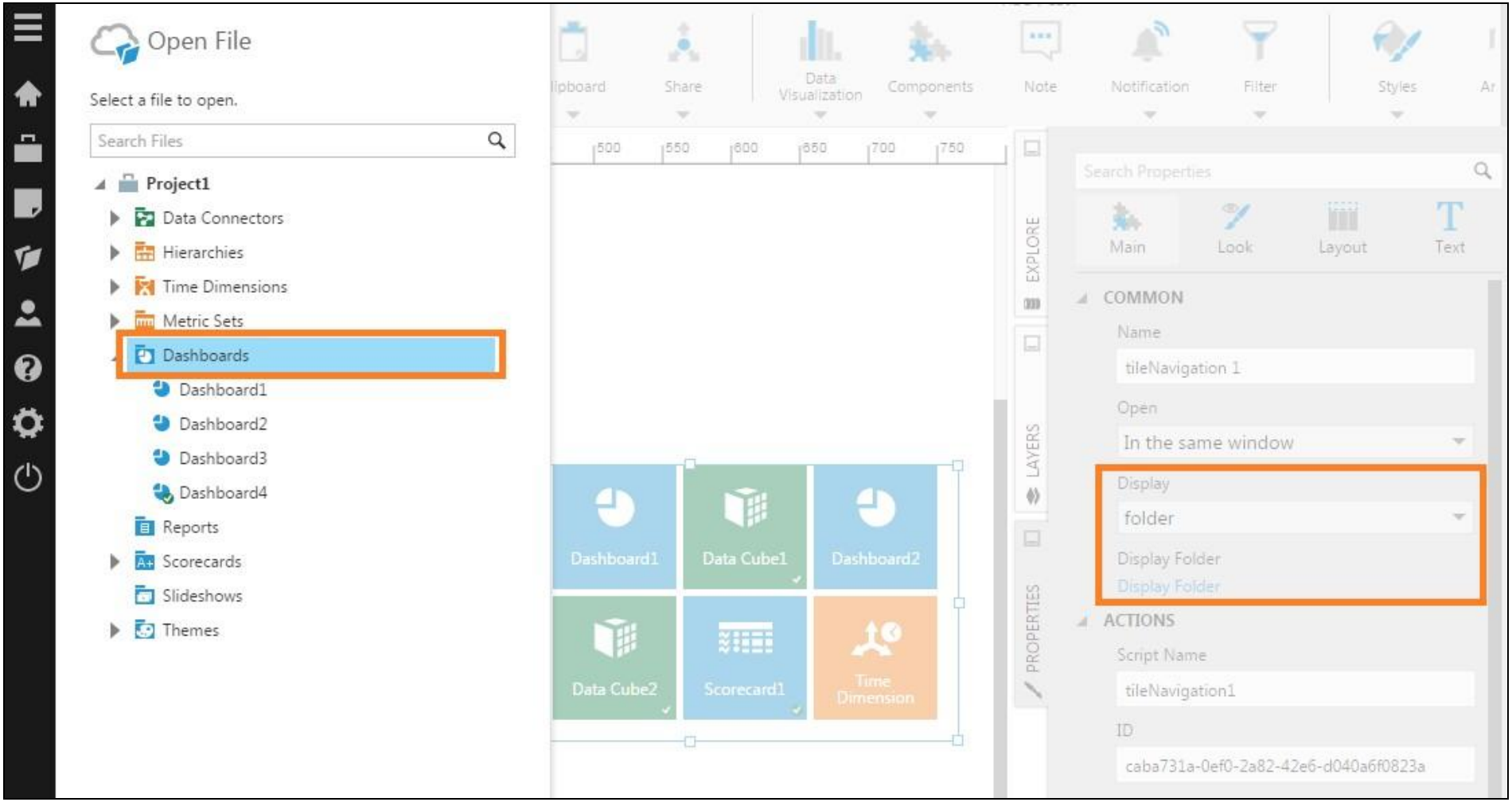
Displaying Recently Viewed Items

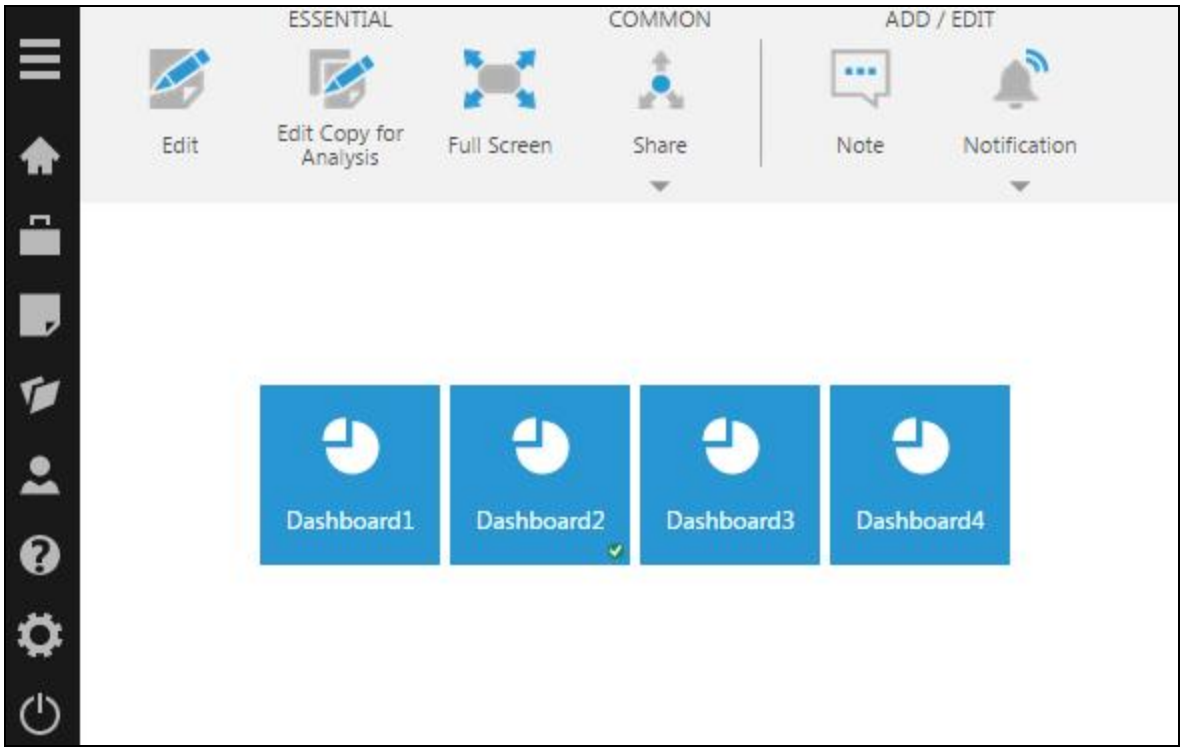
When the **Recently Viewed** option is selected, the Tile Navigation component displays items that the current user has recently viewed.

> **Note:** The number of recently viewed items is configurable by the setting Most Recently Used Entries List Maximum Size in administration. (The number of items may initially remain higher after lowering this setting until the user views new items.)

Displaying a Folder

When the **Folder** option is selected, click **Display Folder** to choose a folder in the active project. Tiles will be displayed for each immediate child item in the folder.

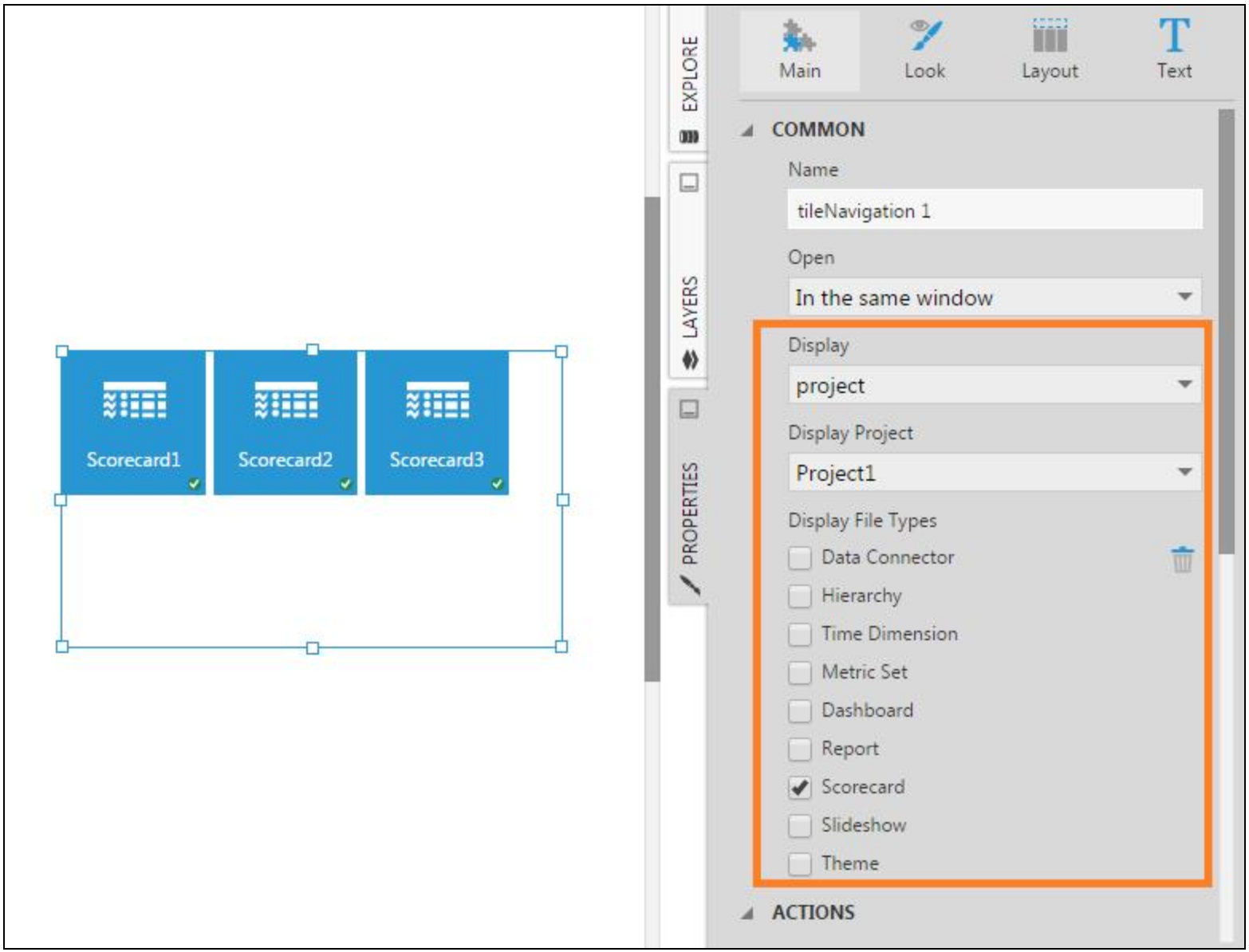

Open File
Select a file to open.
Search Files
Project1
Data Connectors
Hierarchies
Time Dimensions
Metric Sets
Dashboards
Dashboard1
Dashboard2
Dashboard3
Dashboard4
Reports
Scorecards
Slideshows
Themes
Share
Data Visualization
Components
Note
Notification
Filter
Styles
Dashboard1
Data Cube1
Dashboard2
Data Cube2
Scorecard1
Time Dimension
EXPLORE
LAYERS
PROPERTIES
Search Properties
Main
Look
Layout
Text
COMMON
Name
tileNavigation 1
Open
In the same window
Display
folder
Display Folder
ACTIONS
Script Name
tileNavigation1
ID
caba731a-0ef0-2a82-42e6-d040a6f0823a



Displaying a Project

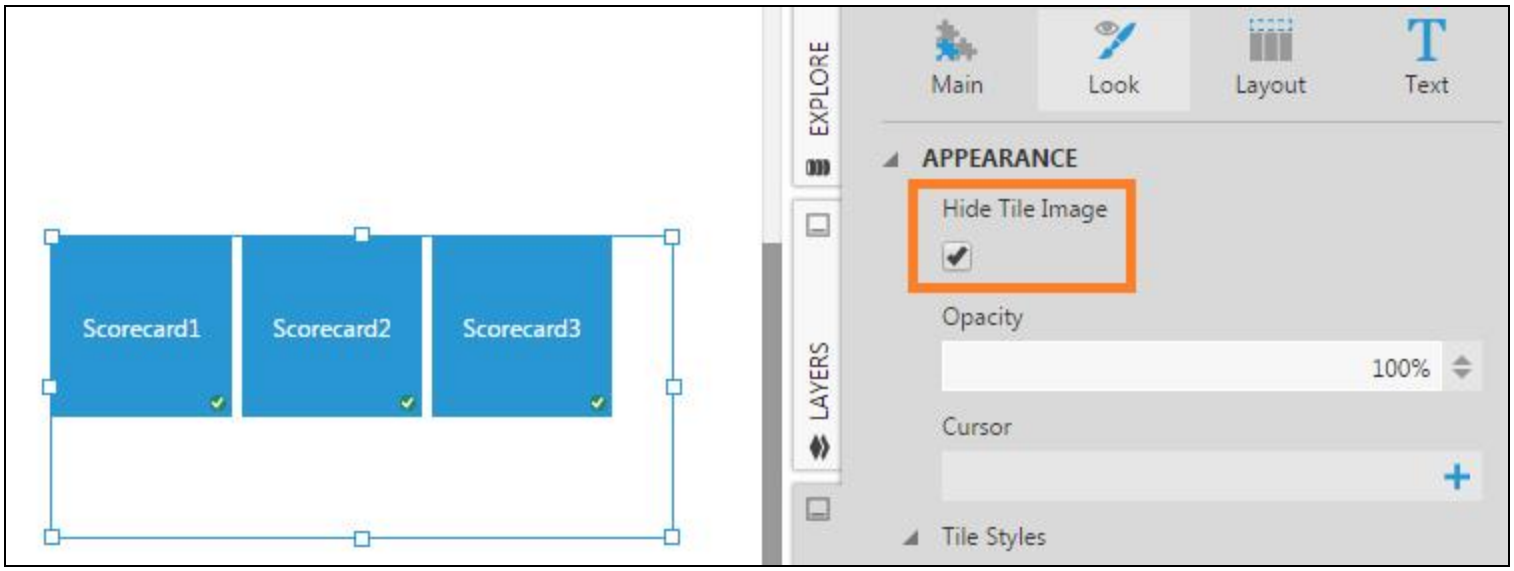
Choose the **Project** option to display tiles corresponding to items in the project selected as the `Display Project`.

You can set `Display File Types` property to limit the types of files that are displayed.



Hide Tile Image

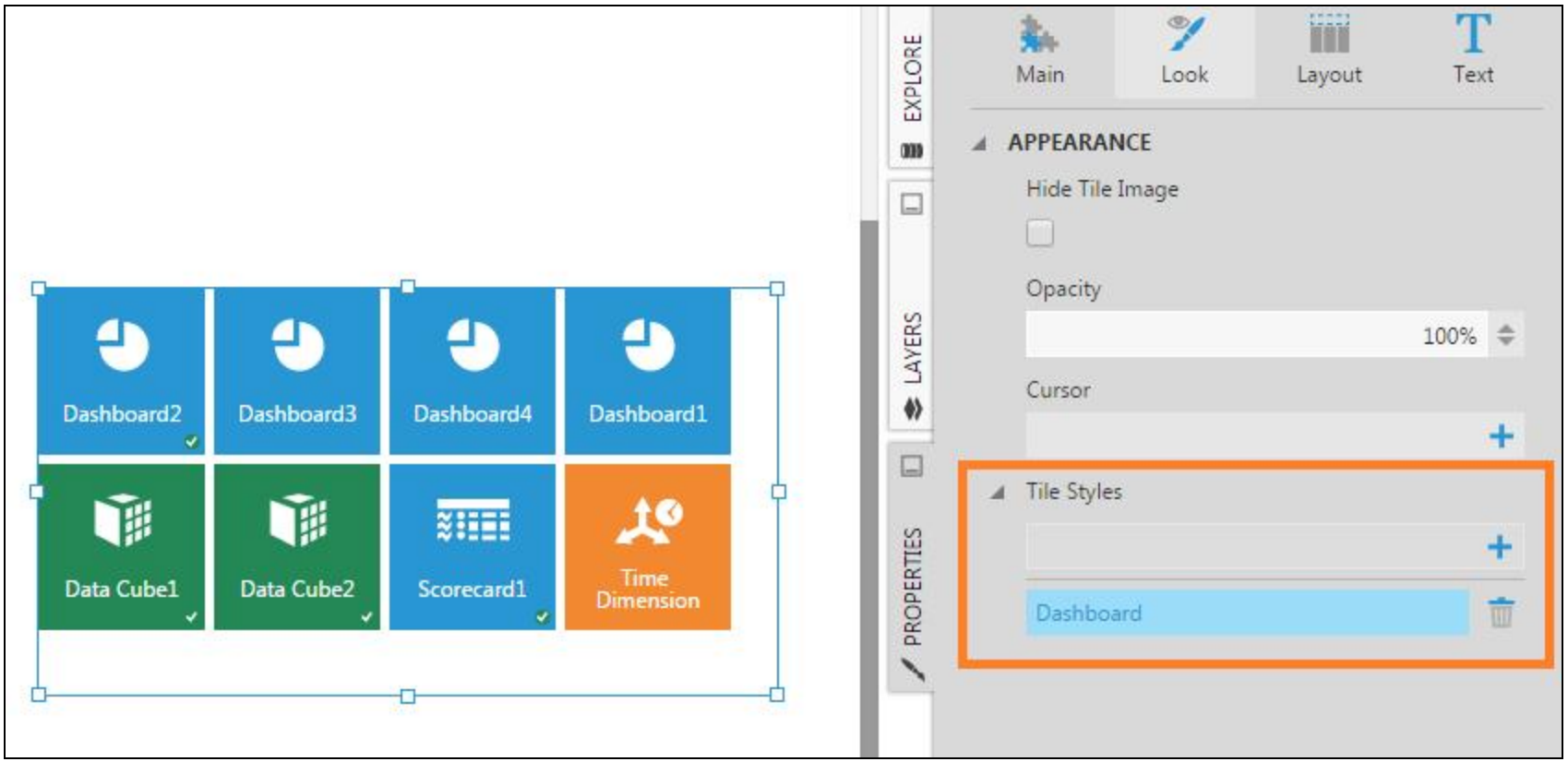
In the **Look** tab under **Appearance**, select the **Hide Tile Image** option if you want to display only text in the tiles.



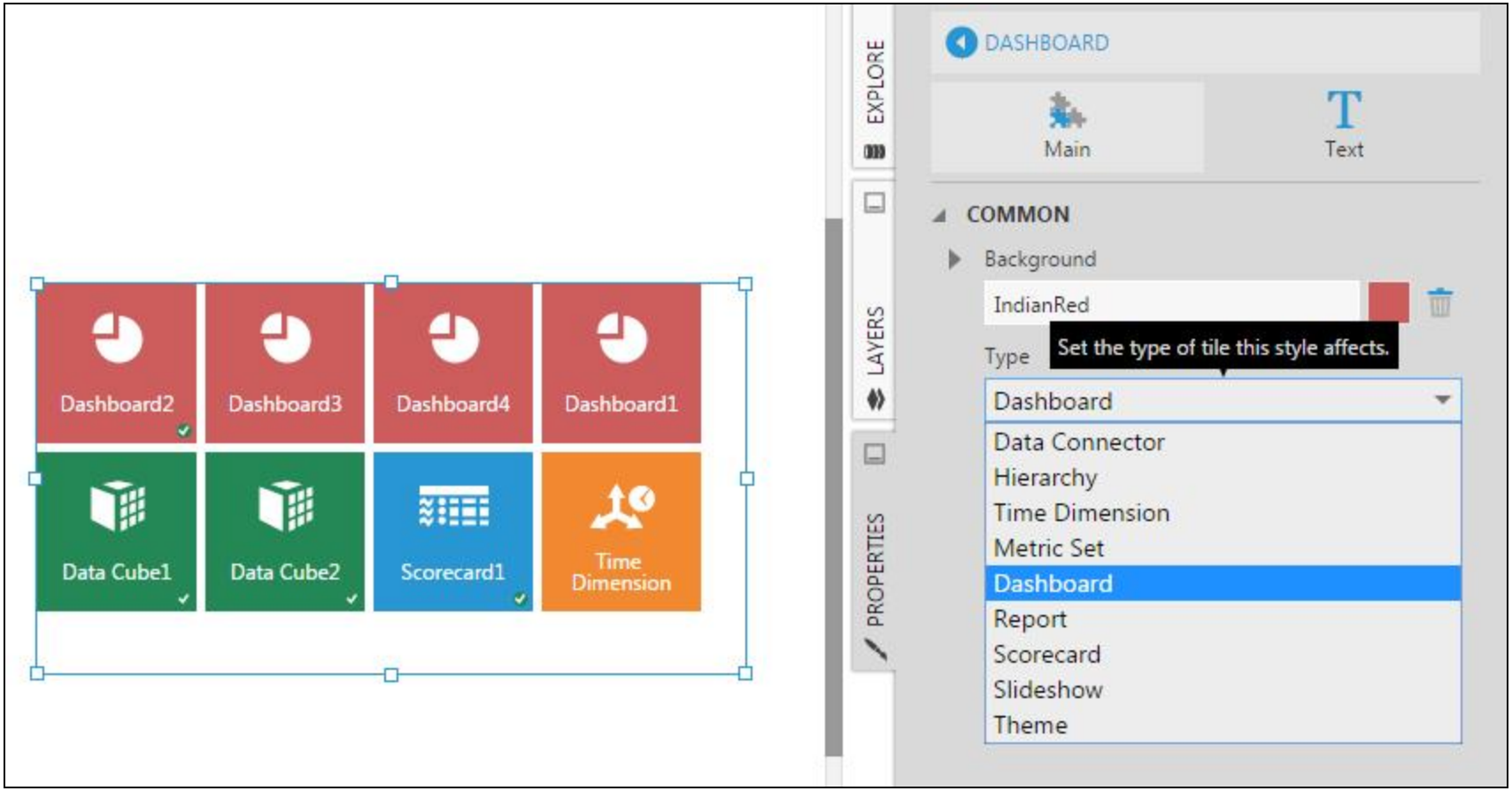
Tile Styles

Tiles for dashboards have a different background color than tiles for data cubes. You can customize the background color of tiles on a per-file type basis by adding one or more tile styles.

In the **Look** tab under **Appearance**, select the plus sign (**+**) to add a tile style. A tile style for dashboards is added by default (named **Dashboard**).



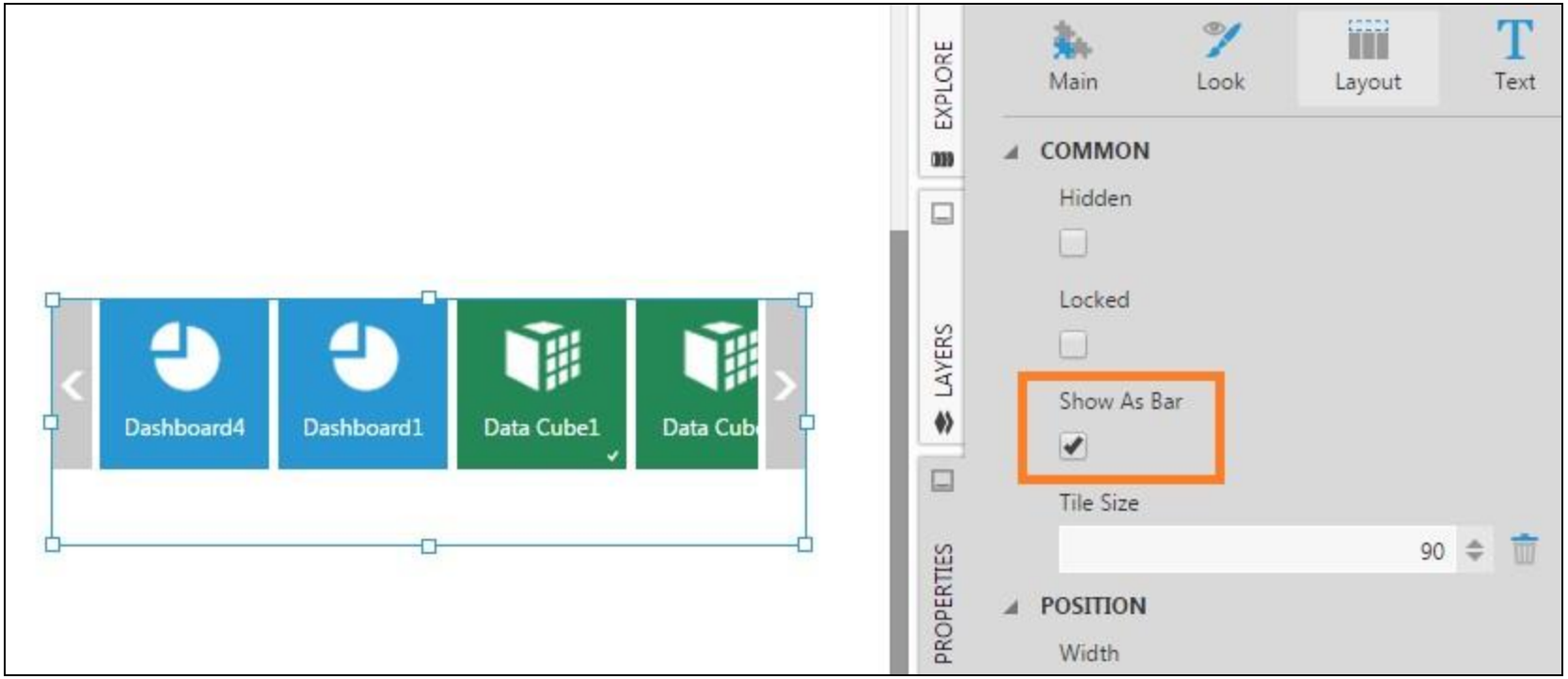
Select the dashboard tile style to edit it. You can now set the background color or change the file type.



Show As Bar

By default, tiles are displayed in a rectangular grid which wraps depending on the size of the component. You can change the component to display the tiles as a horizontal bar with left and right arrow buttons as needed (similar to the main toolbar).

In the **Layout** tab under **Common**, select the `Show As Bar` option.



Tile Size

Use the `Tile Size` property in the **Layout** tab under Common to change the size of the tiles in pixels.

For more information, see: How To Set Up Menu Navigation

Using a View Container

This applies to: *Managed Dashboards, Managed Reports*

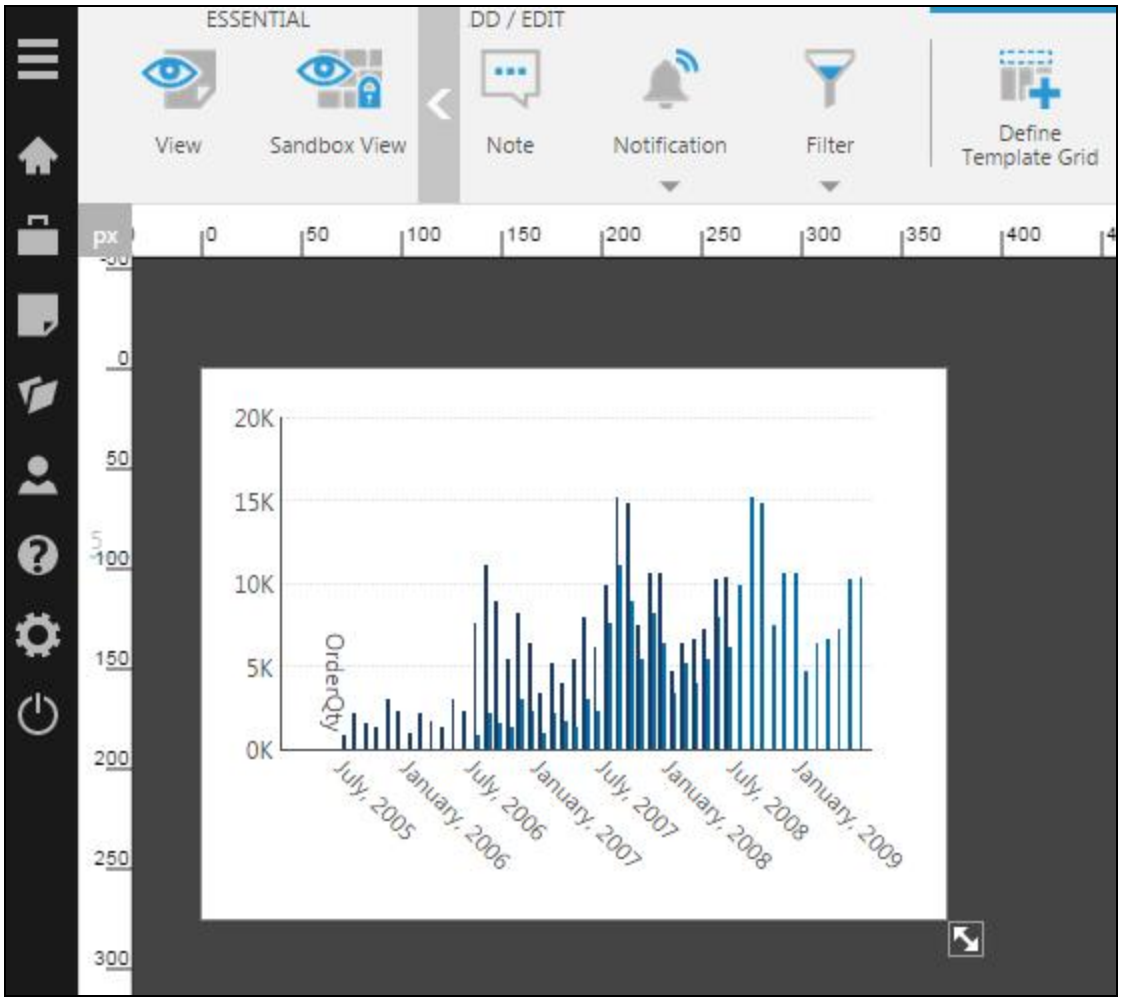
Use a view container component to embed another dashboard, report, or other view within your current view.

Drag the other view from the Explore window, or you can add and set up a Container component.

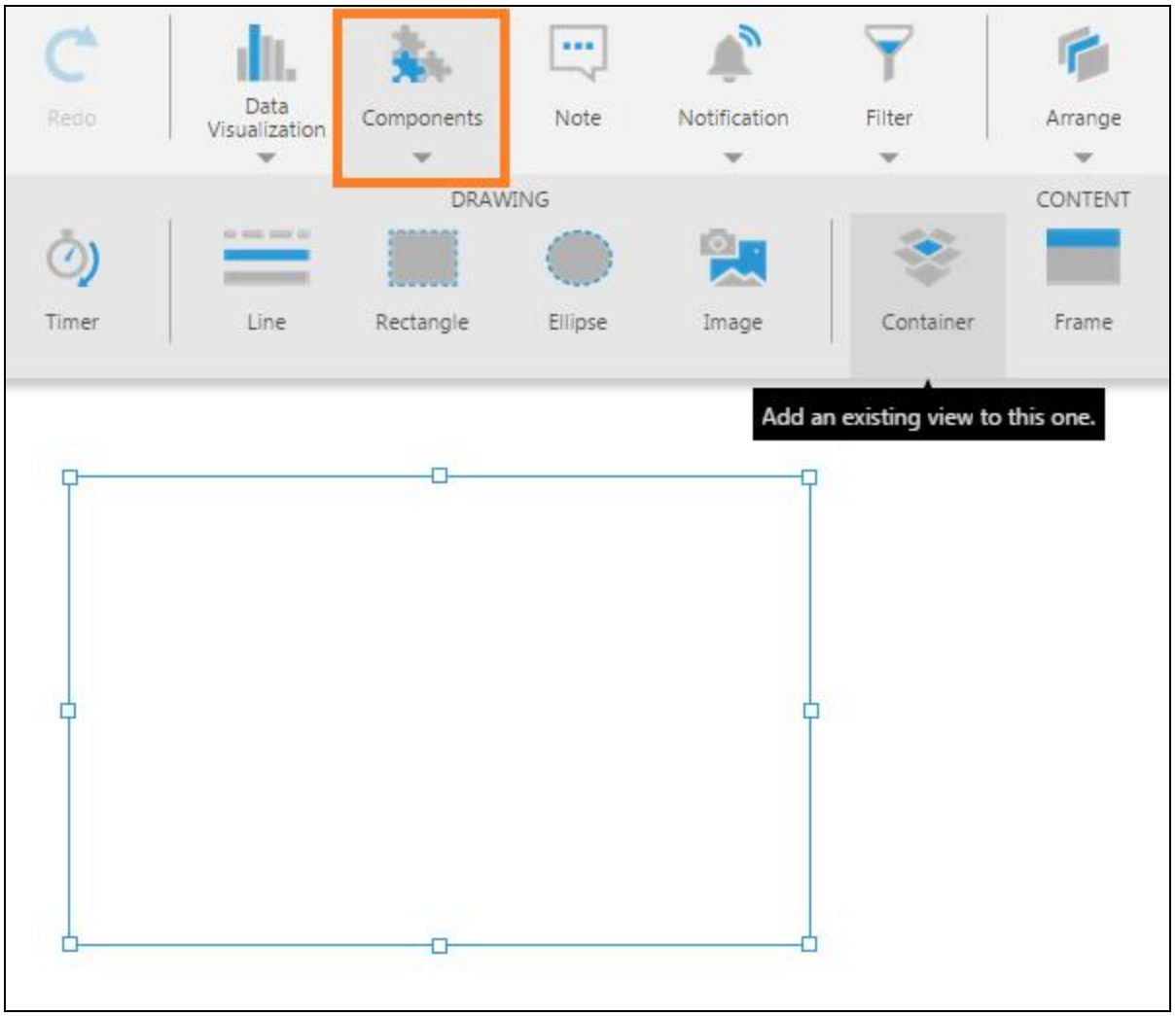
Related video: Dashboard Components

Add a View Container Component

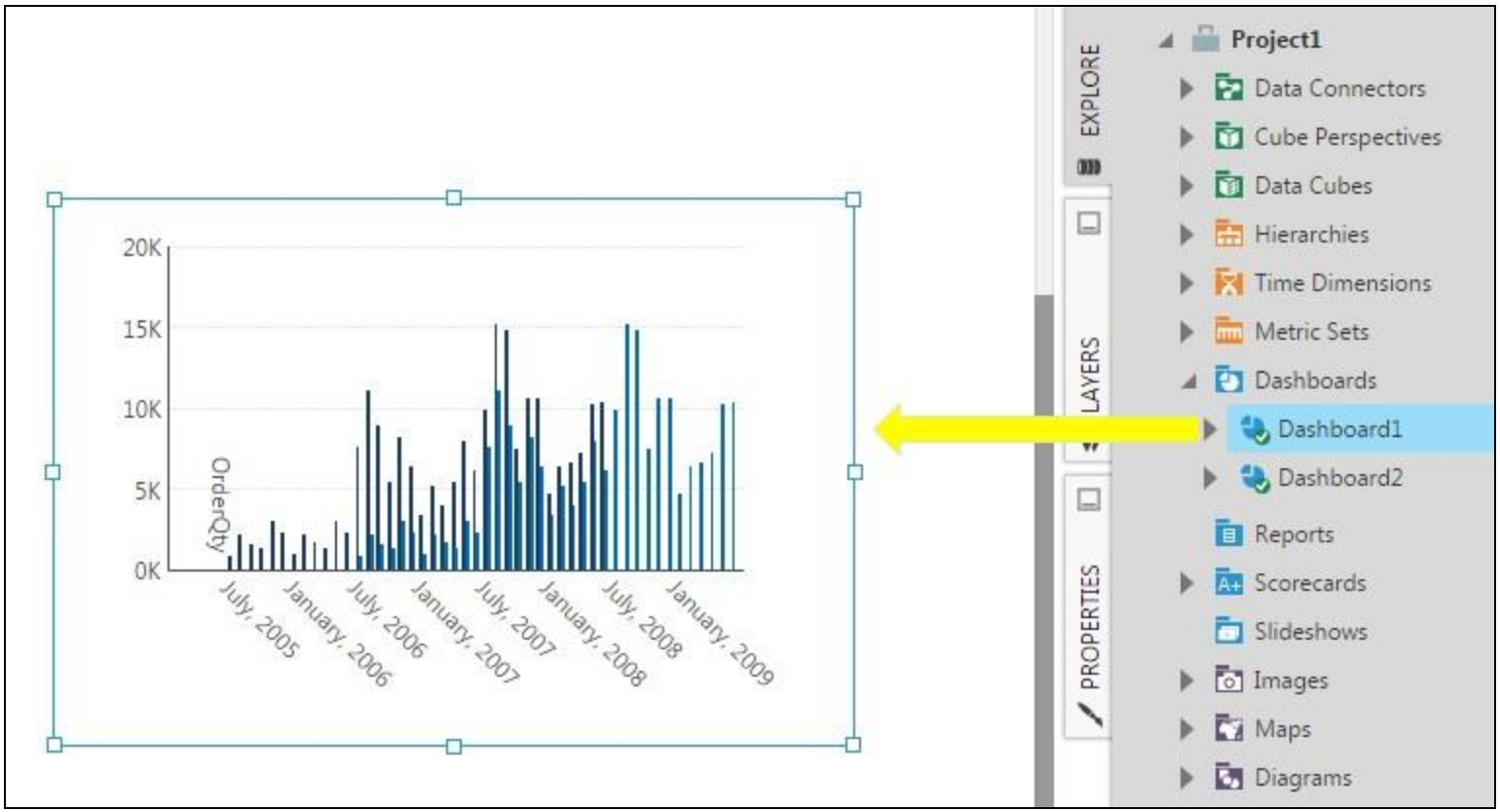
For this example, we have a small dashboard named *Dashboard1* containing one chart, and it was resized just to fit the chart.



While editing another dashboard or view, click **Components** in the toolbar and choose **Container** to add an empty view container to the canvas.



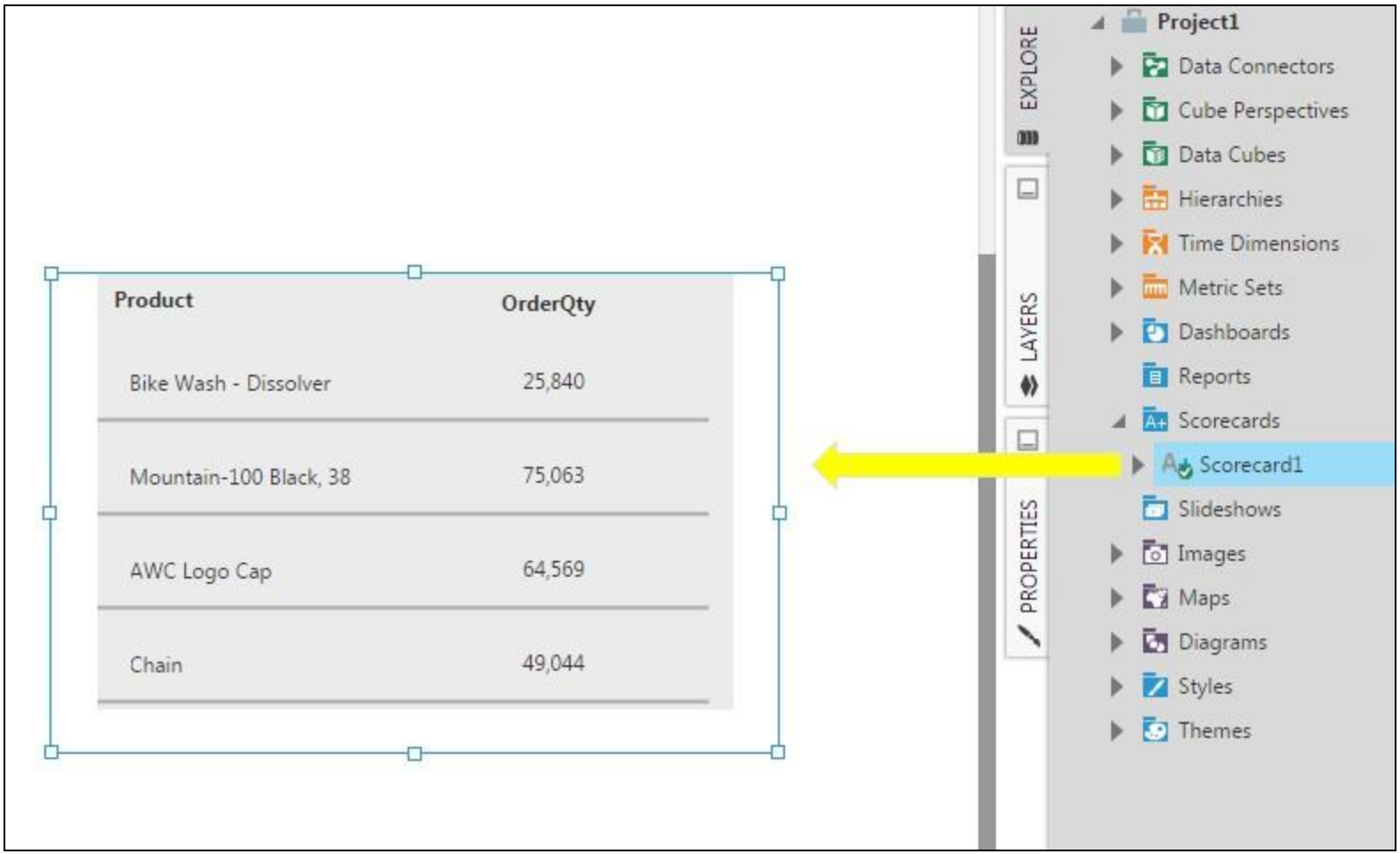
Drag the other view (e.g., Dashboard1) from the **Explore** window onto the empty view container to embed it.



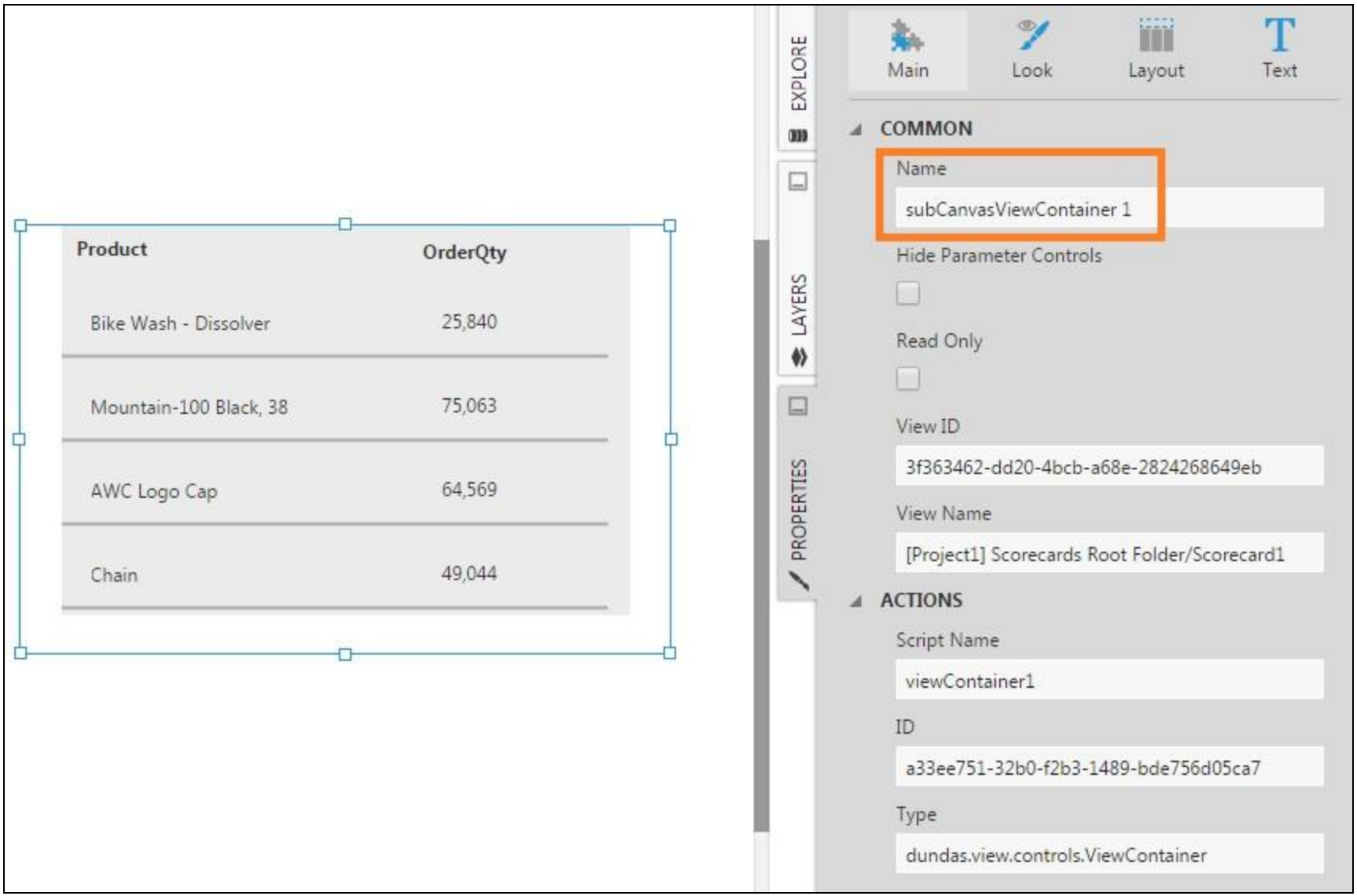
Drag a View Onto the Canvas

Instead of adding an empty view container to the canvas first, you can simply drag any other view from the Explore window directly to your canvas. A view container will be created automatically and used to contain the dragged view.

For example, drag a scorecard from **Explore** to the canvas.



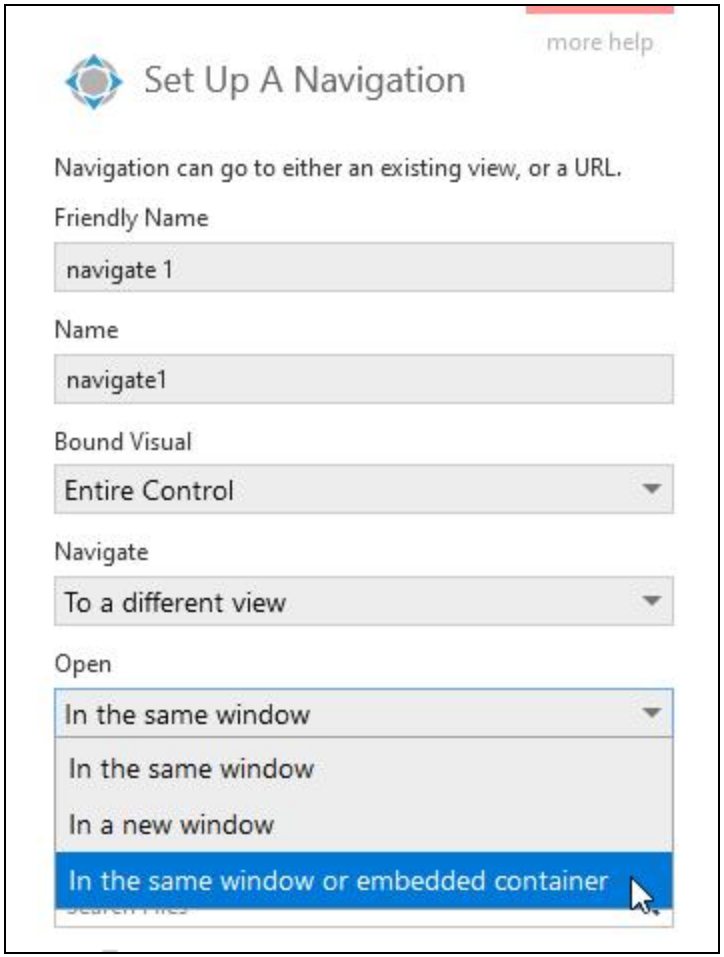
Go to **Properties** and you'll see that this is a view container.



Navigation

Navigation interactions and other interactions such as drilling down can open a different view. If you are setting up this interaction on a view that will be embedded inside another view, you can choose for the navigation to occur only within the embedded container rather than for the entire current window.

To do this, set the **Open** option to In the *same window* or *embedded container* in the interaction dialog.



If you want your viewers to be able to click to load different views within the view container from the outer dashboard, you can use a menu component for this. Its **Open** setting can be set to **In an embedded view** and you can then set **View Container** to your container. See How To Set Up Menu Navigation.

Scripting with View Containers

Views embedded in other views are displayed in a 'subcanvas' container rather than an iFrame element, which improves loading performance. This can impact the use of script in embedded views.

Most uses of `dundas.context.*` refer to the context of the top- level view and its canvas instead of the embedded one. If you access this within an embedded view, the top-level view or canvas will be returned, which is likely not what you want.

Certain features not directly related to a specific dashboard or view are safe to access using `dundas.context`, such as:

- `dundas.context.currentDialogShown`
- `dundas.context.originalQueryString`
- `dundas.context.currentProjectId`
- `dundas.context.currentSession`
- `dundas.context.currentSessionId`

Otherwise, references to `dundas.context` should be replaced with properties and methods of `this`. For example:

```
this.getService("BaseViewService").
```

For convenience, in most cases you can call `this.baseViewService` to get the BaseViewService, or `this.parentView` to get the Dashboard or other view (or simply `this` in one of the events of the dashboard or view, such as **Ready**).

You can check which kind of view container you have by going to its **Properties** in the Actions section and finding the `Type` property. A `dundas.view.controls.ViewContainer` is the older iframe-based view container, whereas the newer type is `dundas.view.controls.SubCanvasViewContainer`.

For more information, see:

- Video: Dashboard Components
- Filter Child Dashboards From a Parent Dashboard

- Use a Dashboard in a Report
- Set Up a Navigate Interaction
- How To Set Up Menu Navigation
- Symphony Managed Reports
- Symphony Scorecards

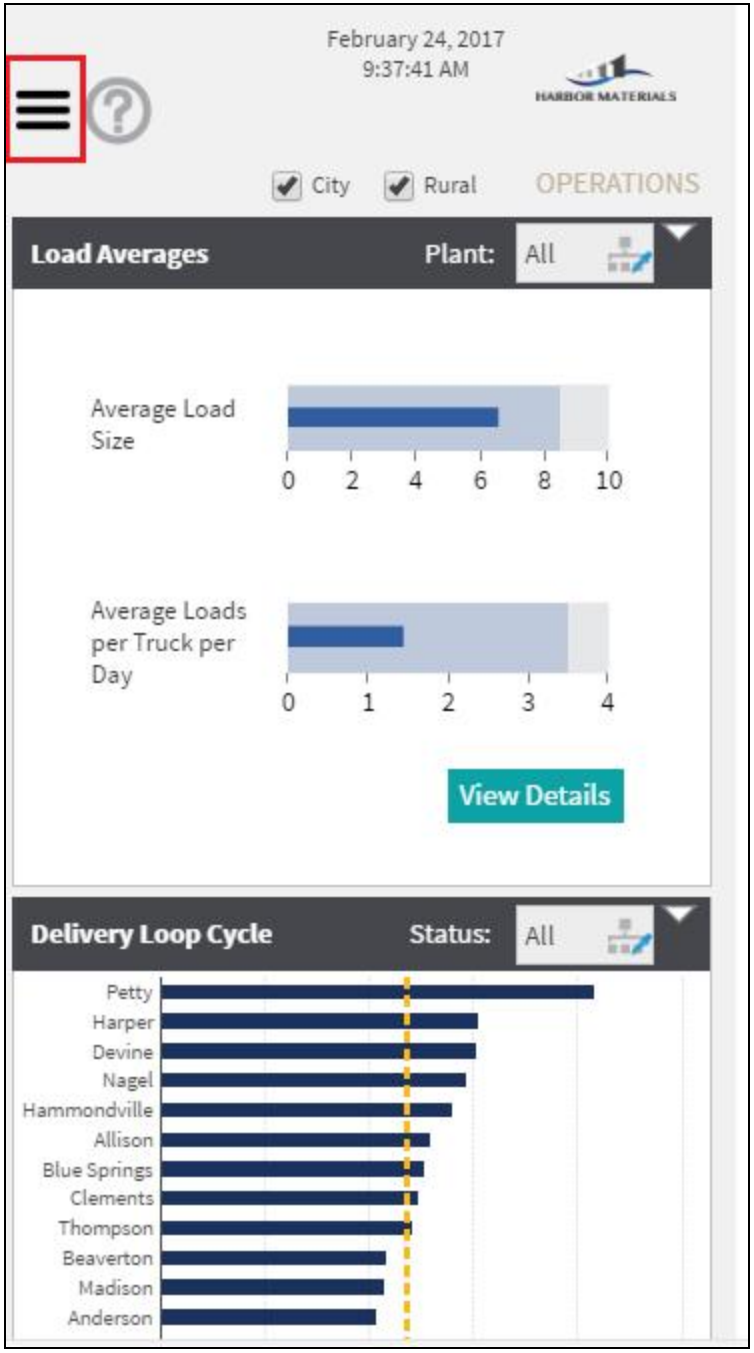
Implementing a Mobile Optimized Navigation Menu

This applies to: *Managed Dashboards, Managed Reports*

Many users may want to be able to view a dashboard on their phone or on a tablet in portrait mode. This is possible for any dashboard, or you might create a mobile-optimized version, but you can also design a single

version of a dashboard that specially adapts to fit its content onto the current screen size using Responsive re-size mode. This mode automatically re-organizes the content as needed so that it fills the given screen real estate but scrolls to additional content that would not fit at the same time.

For the smallest screen sizes, you may need to further minimize the amount of space taken up for navigation or other options on your dashboard. This article explains how to add a hamburger menu (also known as the menu button or navigation drawer) for those devices to navigate between dashboards.


February 24, 2017
9:37:41 AM
HARBOR MATERIALS
City
Rural
OPERATIONS
Load Averages
Plant:
All
Average Load Size
0
2
4
6
8
10
Average Loads per Truck per Day
0
1
2
3
4
View Details
Delivery Loop Cycle
Status:
All
Petty
Harper
Devine
Nagel
Hammondville
Allison
Blue Springs
Clements
Thompson
Beaverton
Madison
Anderson

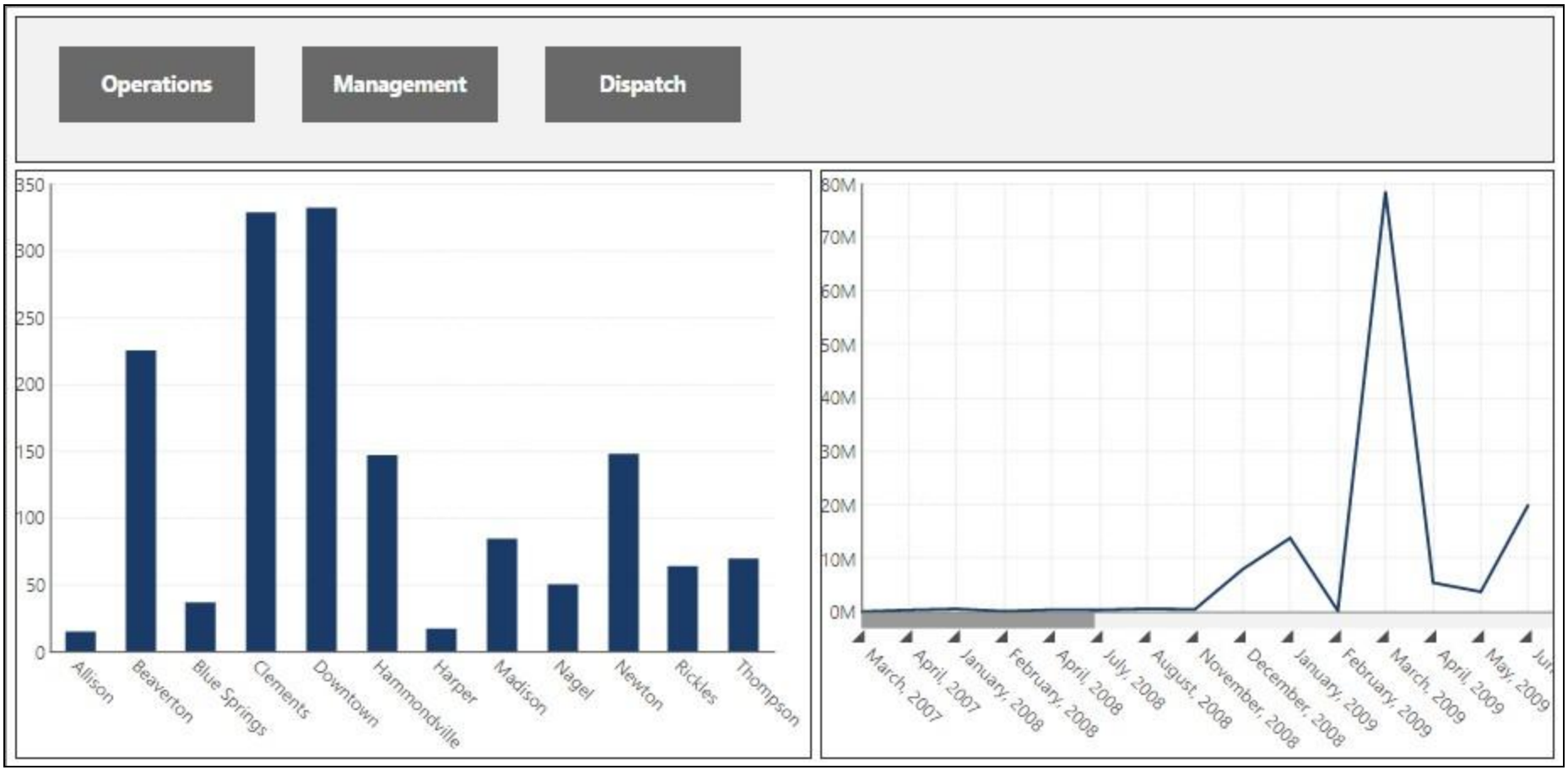
Understanding the Browser Window Resizing Event

To determine when to show the full-size navigation buttons and when to show the hamburger menu button, it is important to understand when the dashboard window resizing event is fired. When the dashboard loads for the first time, Symphony detects the browser's window size and adjusts the dashboard canvas to fit. For this example, it is required to detect the browser window size when loading the dashboard and hide or show the navigation buttons accordingly.

When using smartphones or tablets, users may rotate the device to change the display orientation between portrait and landscape mode. This is another case to consider for hiding or showing different navigation buttons, and if you decide to handle it, the dashboard's Re-Size actions can be used in a similar way as done for the dashboard's Ready actions below.

Example Dashboard

Consider a simple dashboard based on the template grid example.



For the above dashboard, the default settings are given below:

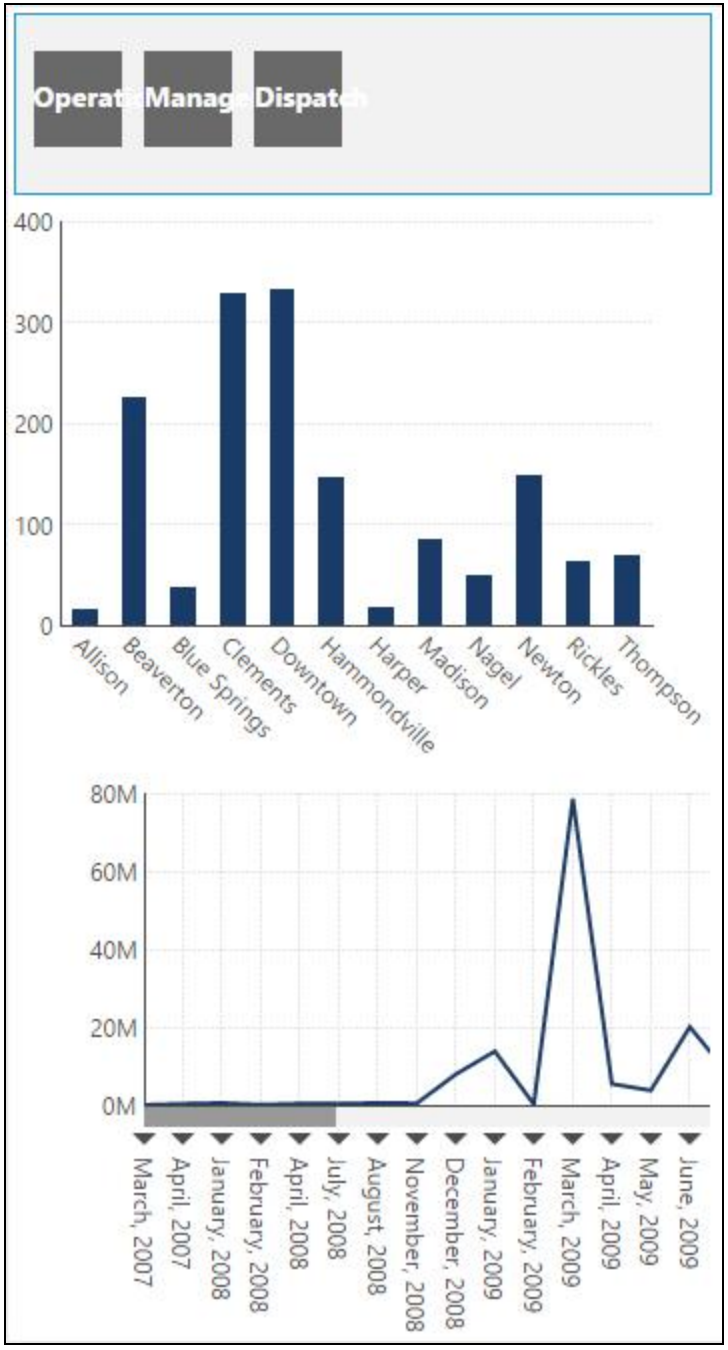
- Re-Size Mode: Responsive
- Width: 1024
- Height: 500

The dashboard also has two layers:

- MainLayer, which contains two charts.
- NavigationLayer, which contains three navigation buttons (Operations, Management, and Dispatch).

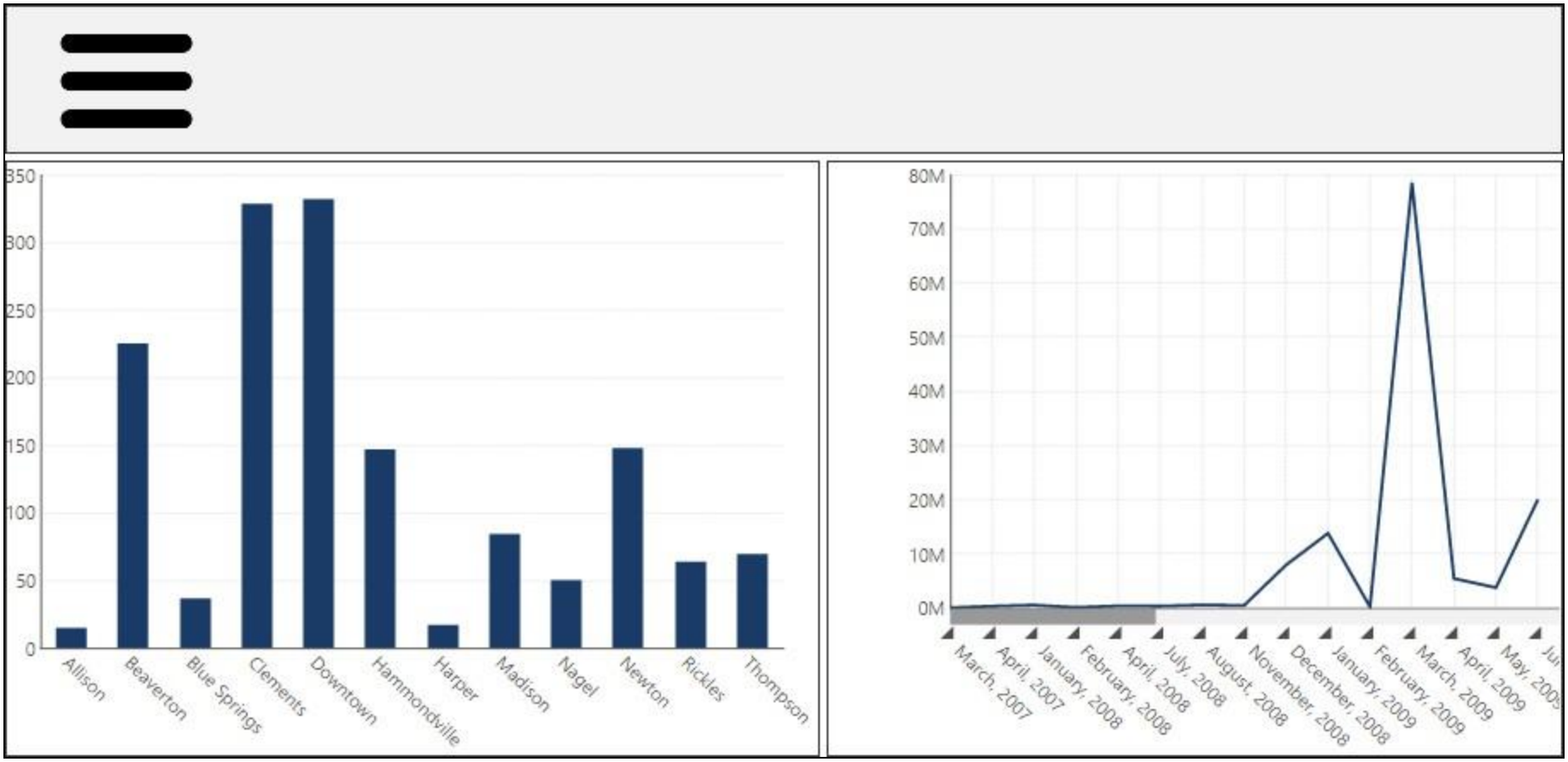
Note: The NavigationLayer buttons are inside a single cell and are not attached to any cell within the template grid, while the MainLayer charts are attached to different cells.

When the dashboard is viewed on a large monitor, it looks fine. However, when the dashboard is viewed on a smaller screen, the space is too small to fit them and their labels in one row.

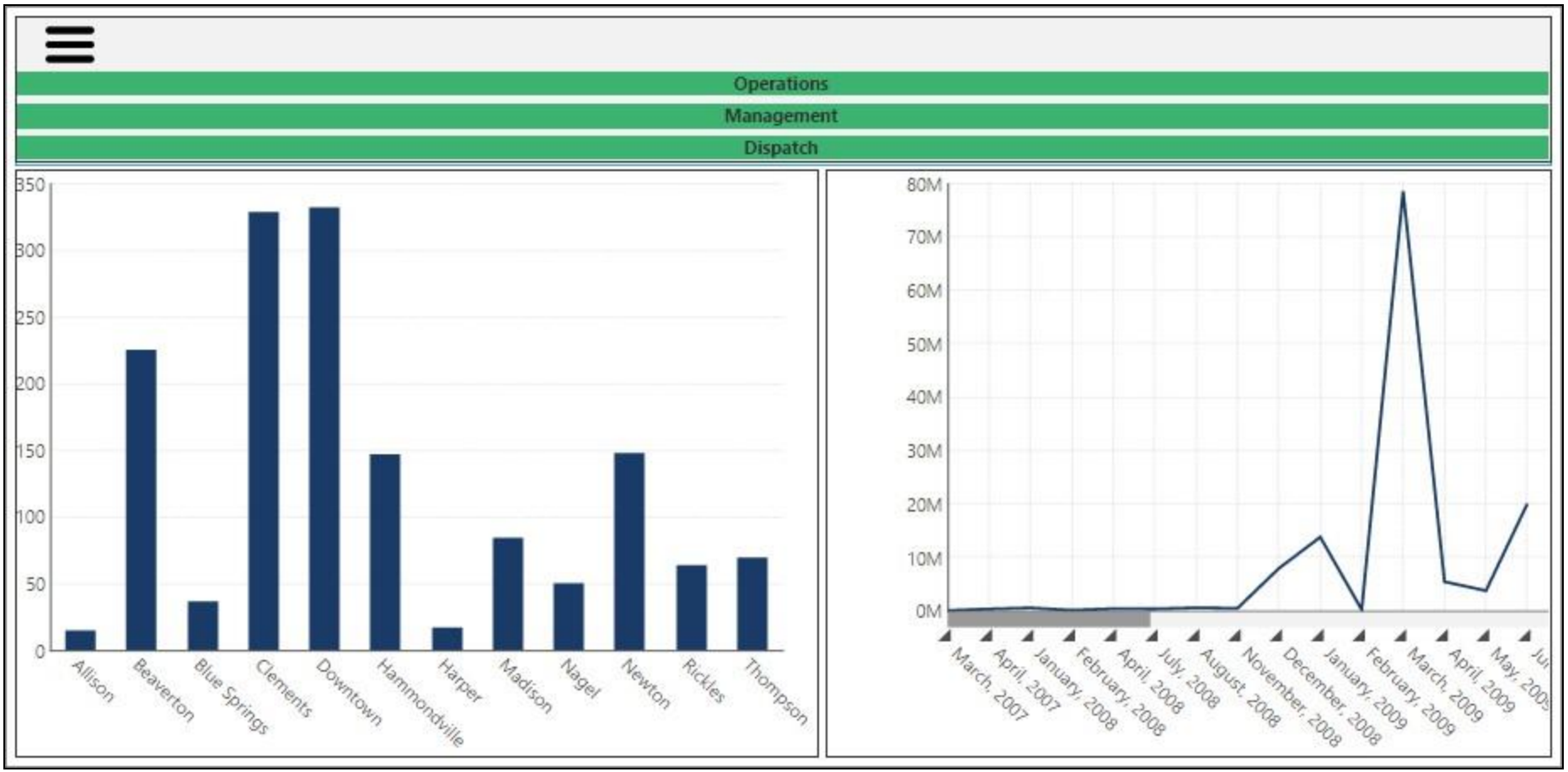

Operati
Manag
Dispatch
400
300
200
100
0
Allison
Beaverton
Blue Springs
Clements
Downtown
Hammondville
Harper
Madison
Nagel
Newton
Rickles
Thompson
80M
60M
40M
20M
0M
March, 2007
April, 2007
January, 2008
February, 2008
April, 2008
July, 2008
August, 2008
November, 2008
December, 2008
January, 2009
February, 2009
March, 2009
April, 2009
May, 2009
June, 2009

Adding the Hamburger Menu

To replace the buttons above with a collapsible menu, add a new layer (HamburgerLayer) in the Layers window and add an Image component to that layer that displays the icon image.



Next, add another layer (DropdownLayer) and add three buttons to that layer, which will make up the menu.



Adding the Scripts

For this example, we will show the hamburger menu if the browser window is less than 1020 pixels in width. In this case, we want to show the HamburgerLayer and hide the NavigationLayer; otherwise, we want to show the NavigationLayer and hide the HamburgerLayer.

To achieve this, use the following script under the **Ready** action of the dashboard:

```
// Get the current window width
var windowWidth = $(window).width();
// Get the canvas service
```

```
var canvasService = this.getService("CanvasService");
// Get the navigation layer
var navigationLayer = canvasService.getLayersByFriendlyName("NavigationLayer")[0];
// Get the hamburger layer
var hamburgerLayer = canvasService.getLayersByFriendlyName("HamburgerLayer")[0];

// Check the window width and show/hide appropriate layers
if (windowWidth < 1020){
	if (hamburgerLayer.hidden) {
		canvasService.showLayers([hamburgerLayer]);
		canvasService.hideLayers([navigationLayer]);
	}
} else {
	if (navigationLayer.hidden) {
		canvasService.showLayers([navigationLayer]);
		canvasService.hideLayers([hamburgerLayer]);
	}
}
```

While the dashboard is viewed on a smaller screen, you want to show a drop-down menu if the user clicks/taps on the hamburger menu button. The idea is to show the DropdownLayer if it is hidden, and vice versa. To achieve this, add the following script under the **Click** action of the Hamburger menu button image.

```
// Get the canvas service
var canvasService = this.getService("CanvasService");
var dropdownLayer = canvasService.getLayersByFriendlyName("DropdownLayer")[0];

if (dropdownLayer.hidden) {
	canvasService.showLayers([dropdownLayer]);
}
else {
	canvasService.hideLayers([dropdownLayer]);
}
```

Testing the Hamburger Menu Button

If you open the dashboard on a small screen device, it will detect the browser window size when the dashboard is at the Ready state and will show the Hamburger menu button on the top left corner, as per the design.

If you tap the Hamburger menu button, a drop-down menu with three buttons will appear underneath the Hamburger button.

For more information, see:

- JavaScript API
- Script Library: Show or hide a layer
- Script Library: Animated sliding content layer
- Samples: Material Operations Dashboard
- Writing Scripts Using Browser Developer Tools

Improving Responsive Mode With CSS

This applies to: *Managed Dashboards, Managed Reports*

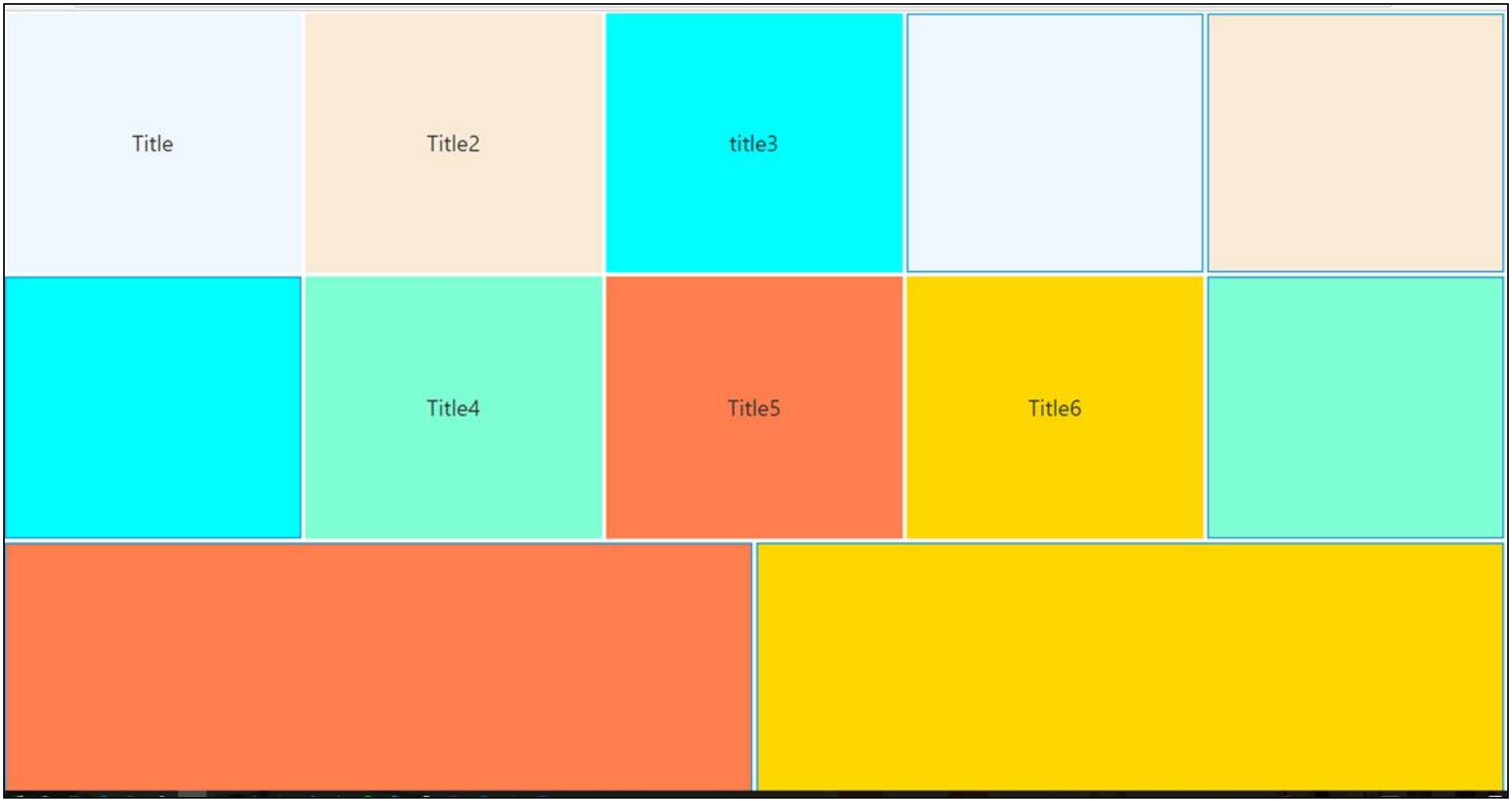
When using a single dashboard for both very small and very large screen sizes, the responsive layout may not behave as you prefer. This article provides an example of how to customize the responsive layout for some dashboards using media queries in CSS.

Preparing the Dashboard

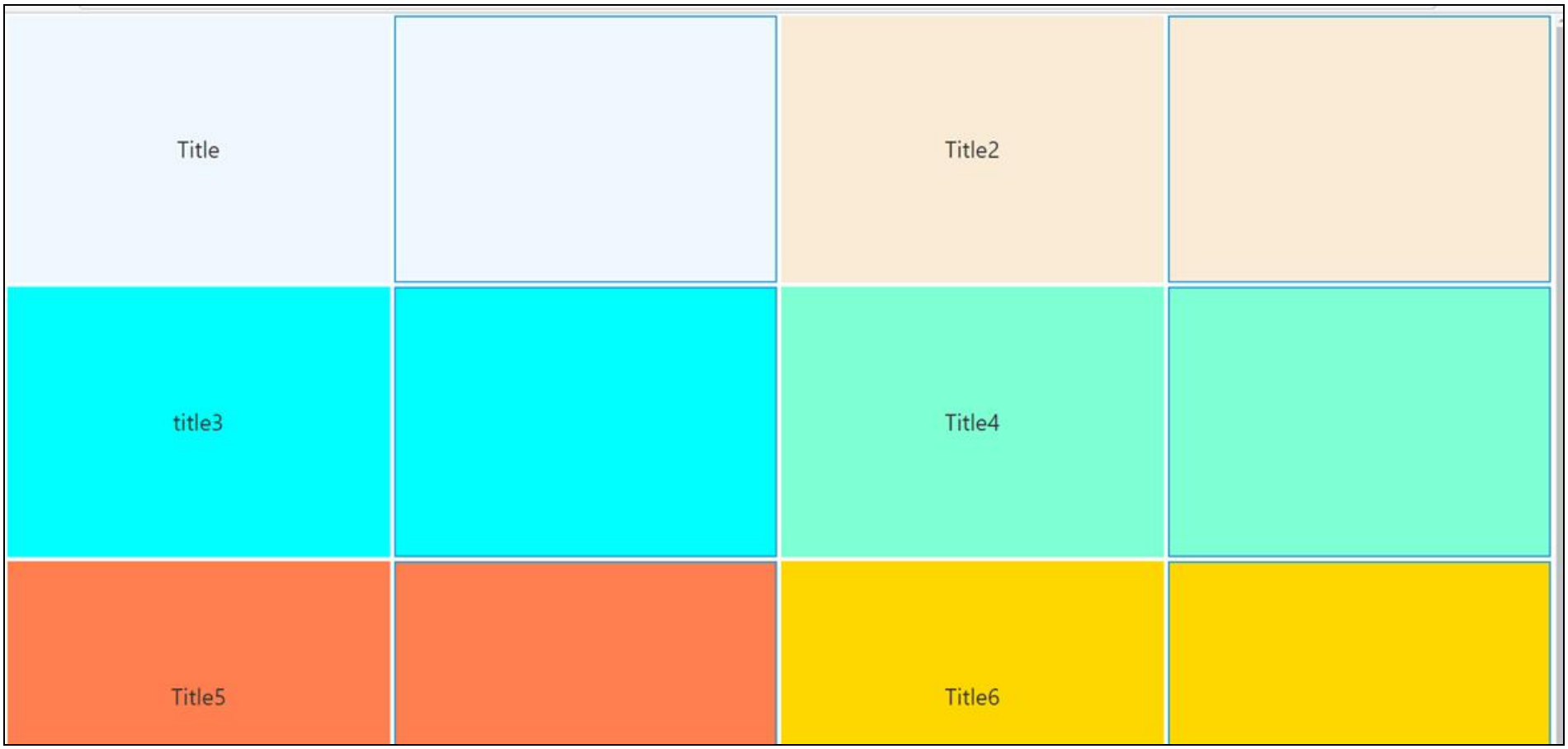
Consider the following dashboard in Edit mode:


Title
Title2
title3
Title4
Title5
Title6

When viewing this dashboard in Responsive mode and on a large screen, you will see the following result:



Back in Edit mode, group each title cell with the appropriate content cell. When viewing on a large screen, you will now see the following:

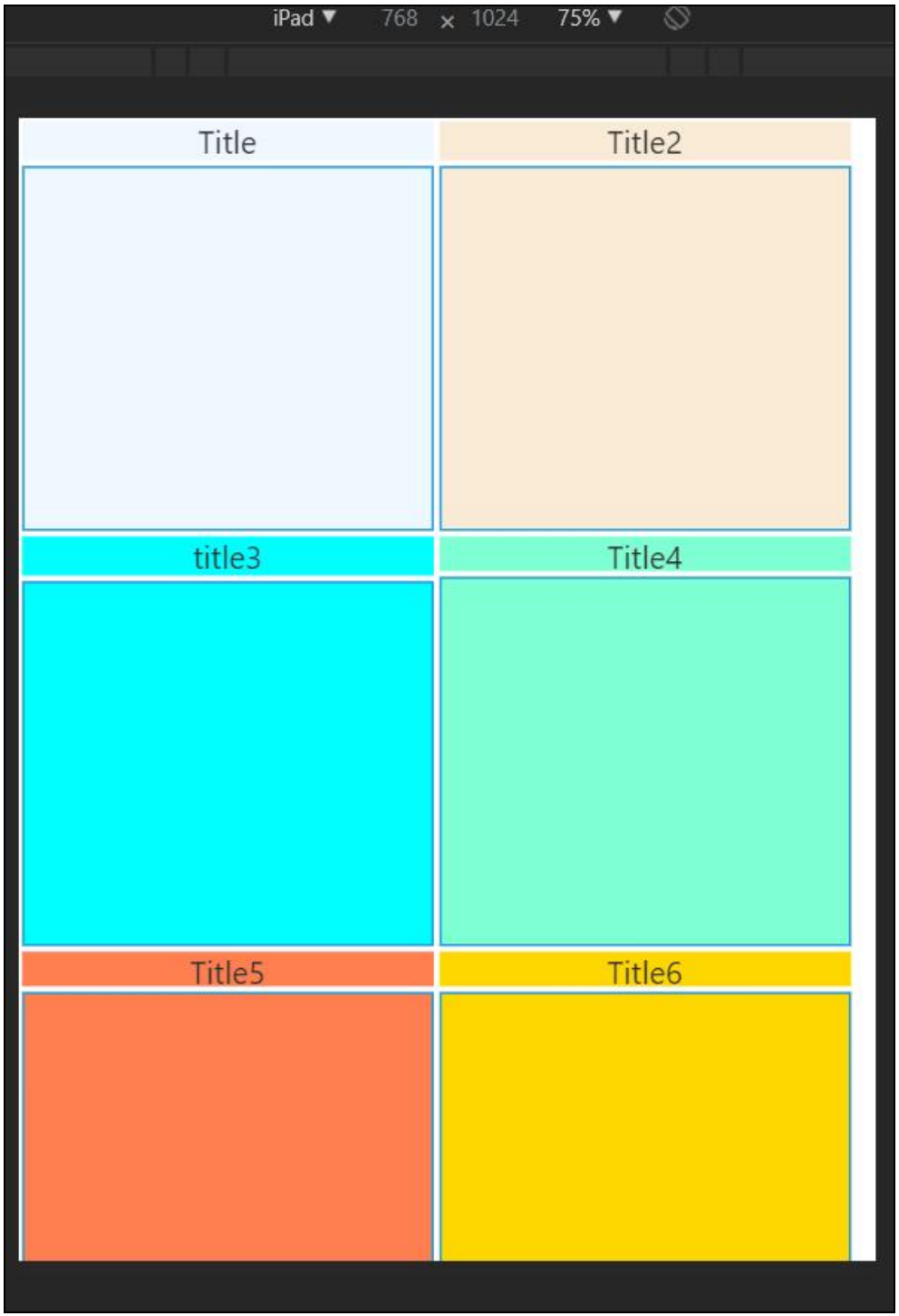


Indicate each title cell as a header.

When viewing on a large screen, you will now see the following:

In comparison, on smaller screens, the resizing behavior of the dashboard works as intended.


Nexus 6P
412 x 732
100%
Title
Title2


iPad
768 x 1024
75%
Title
Title2
title3
Title4
Title5
Title6

Adding CSS and Media Queries

The template grid uses native CSS flex-box, which allows you to use CSS media queries to change the behavior of the template cells depending on the screen size.

In our example, given the pixel widths of the cells in edit mode and from testing in view mode, additional cells are currently added to the top row with a page width above 1740px. The goal will be to adjust the template cells at certain screen widths to each take up about a third of the available width and have equal widths, producing three cells in each row rather than up to five.

For this example, we can define a CSS media query for any screen size larger than 1740px that adjusts the min-width of each template cell group to just under a third of the space (20px are subtracted to allow for the 5px spacing between the cell groups). Since this example uses template cell groups, we target the templatecell-group class, found when inspecting the page and defined in dundas.controls.GridTemplateAdornerConstants. (You could target templategrid-responsivecell instead, for example.)

By finding the layer ID(s) on your dashboard in the Properties window and referring to it in your CSS, as done below with the ID `0a1bd025-4a3f-0bec-ac72-33d81ca721f5`, you could add the CSS to your environment as described in White labeling the application without affecting other dashboards:

```
@media (min-width: 1740px) {
	#dashboardSurface.viewmode.viewmode-responsive .layer[id="0a1bd025-4a3f-0bec-ac72-33d81ca721f5"] .templatecell-group {
		min-width: calc((100% - 20px) / 3) !important
	}
}
```

You can also add CSS directly within a dashboard using an HTML Label to add a style element like the following:

```
<style type="text/css">@media (min-width: 1740px) {
	#dashboardSurface.viewmode.viewmode-responsive .templatecell-group {
		min-width: calc((100% - 20px) / 3) !important
	}
}</style>
```

Important: The CSS rule #dashboardSurface.viewmode.viewmode- responsive limits the scope to view mode. We do not want custom CSS to affect the template cells while editing.

For more information, see:

- Implementing a Mobile Optimized Navigation Menu
- Using a Template Grid for Resizing

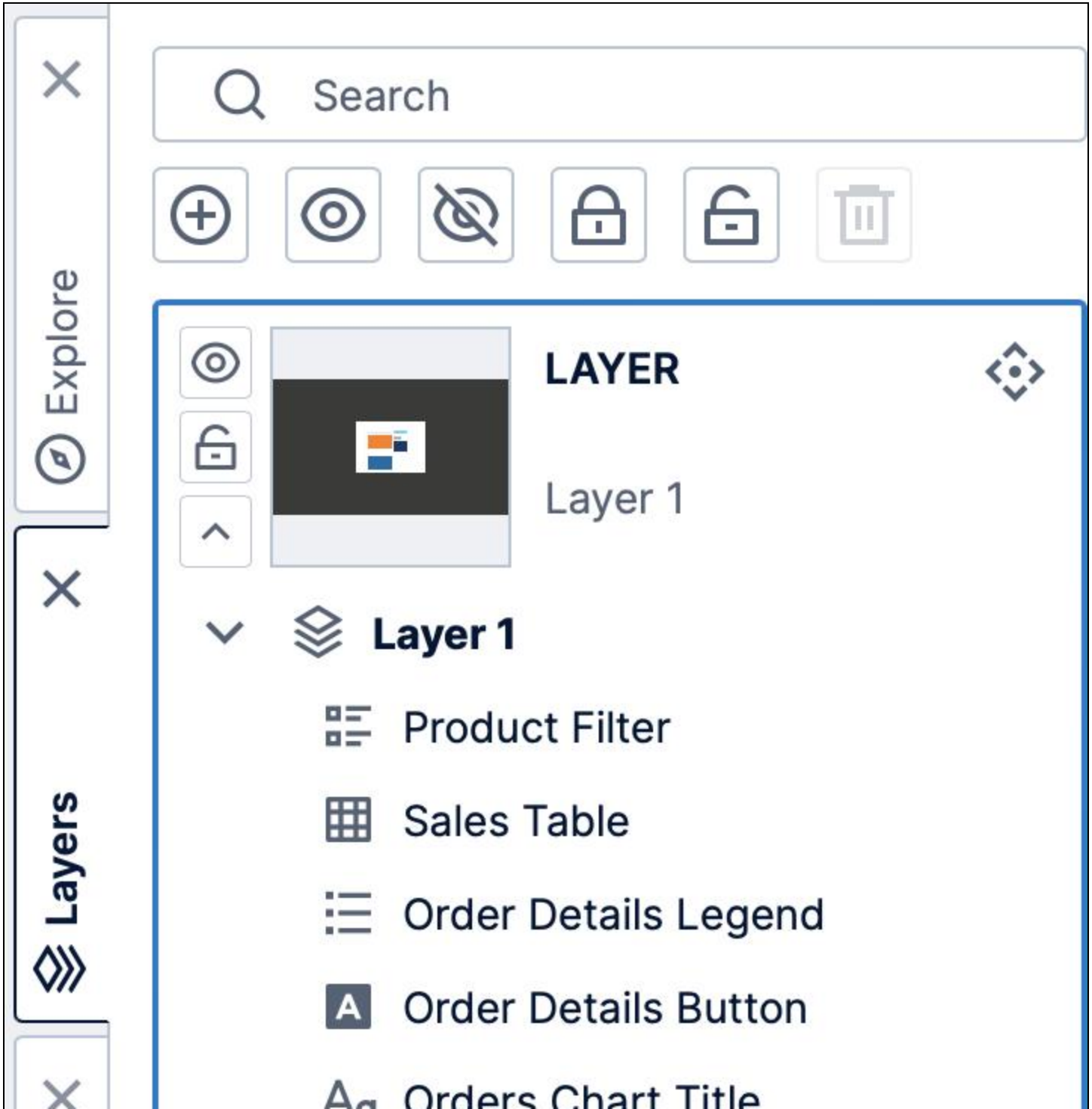
- Video: Template Grids

Layers and Groups

This applies to: *Managed Dashboards, Managed Reports*

Layers and groups can be used in dashboards, reports, and other views. They provide ways to manage and organize multiple elements such as visualizations, filters, and components, and to make changes to multiple elements at once. Both layers and groups are useful when editing, while layers can also be shown or hidden interactively when viewing.

Even when not using multiple layers, the Layers window can help you find and work with elements on your canvas.


Explore
Search
LAYER
Layer 1
Layers
Layer 1
Product Filter
Sales Table
Order Details Legend
Order Details Button
Orders Chart Title

Related video: Group and Layers

Groups

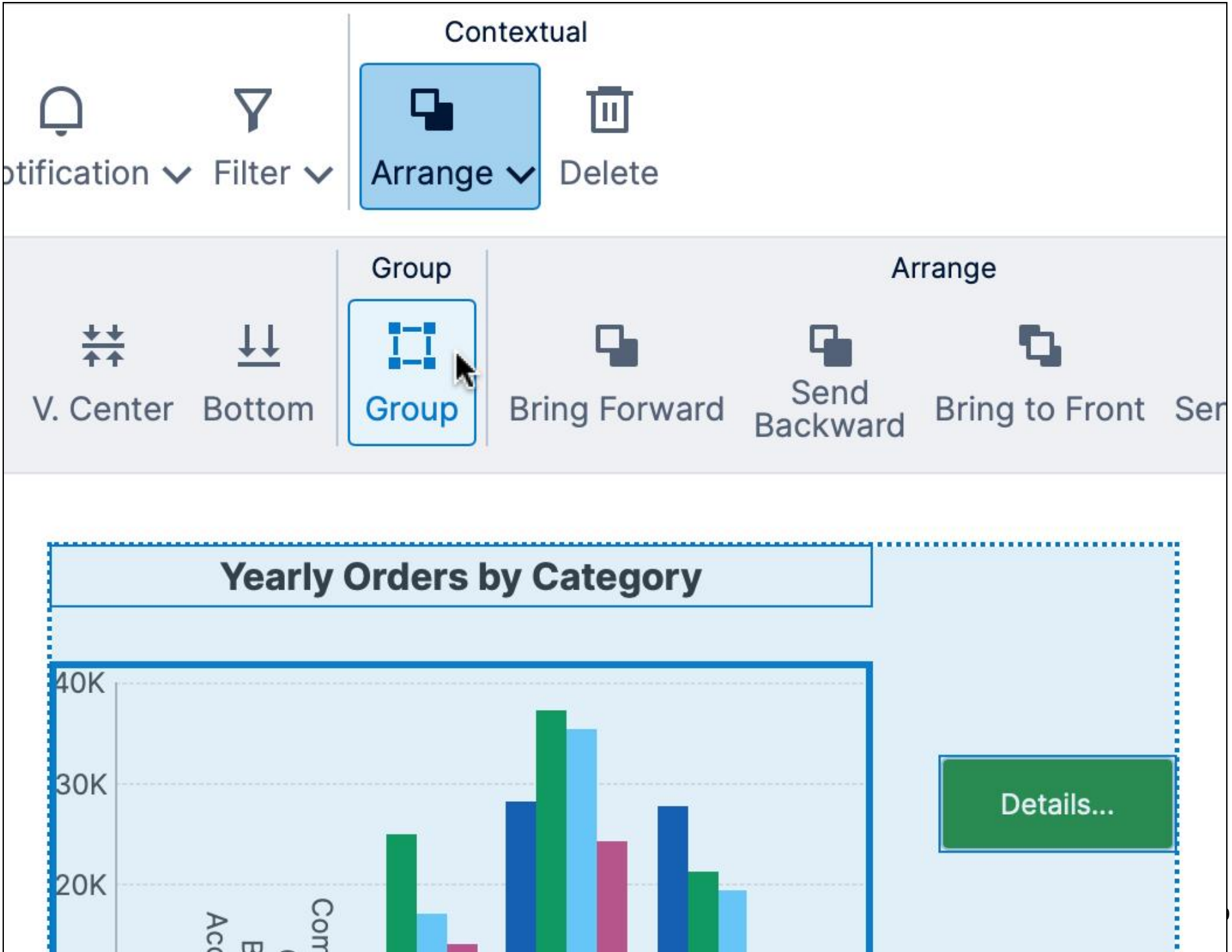
You can group multiple elements on the canvas together when editing. Once grouped, you can easily apply operations to the set of elements as a whole, such as dragging the group to a new location, without having to re-select all of them each time.

If you have related items such as a chart, its title label, and a legend, you may want to group the elements together so they always appear side-by-side no matter where you move the group on the canvas.

Group Multiple Elements Together

To group two or more elements, first select them on the canvas by dragging a selection rectangle around all of the elements. You can also press the CTRL (⌘ on Macs) or SHIFT key while clicking each element that you want to select.

Once you've made the selection, go to the toolbar, click **Arrange**, and then click **Group**.

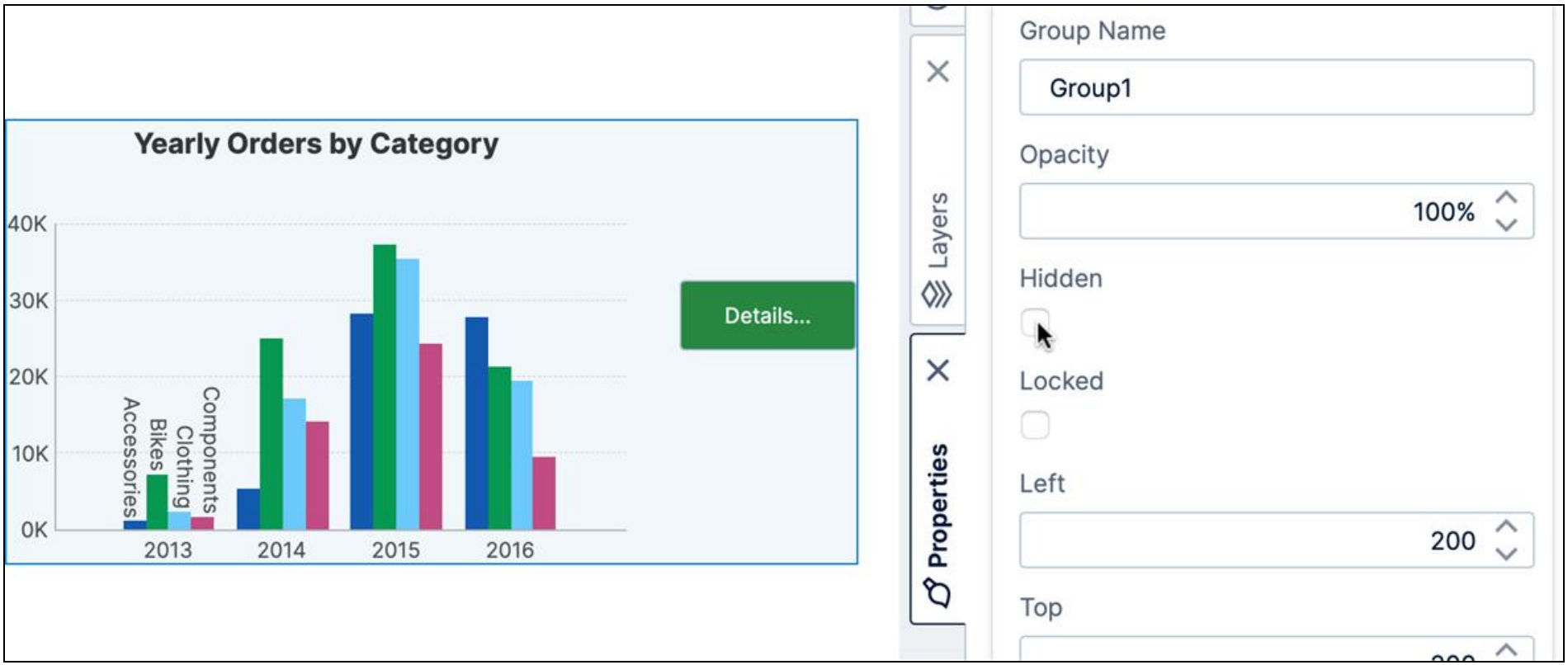

Contextual
otification
Filter
Arrange
Delete
Group
Arrange
V. Center
Bottom
Group
Bring Forward
Send Backward
Bring to Front
Ser
Yearly Orders by Category
40K
30K
20K
Details...

The grouped set of elements appears with a light blue background. You can move the group around the canvas by dragging it. You can even delete the group (and all of its elements).

Show or Hide a Group

A group has its own set of properties, just like any individual element on the canvas.

Select a group on the canvas, and then open the **Properties** window to see the properties of the group. Check or uncheck the`Hidden` property to hide or show the group.

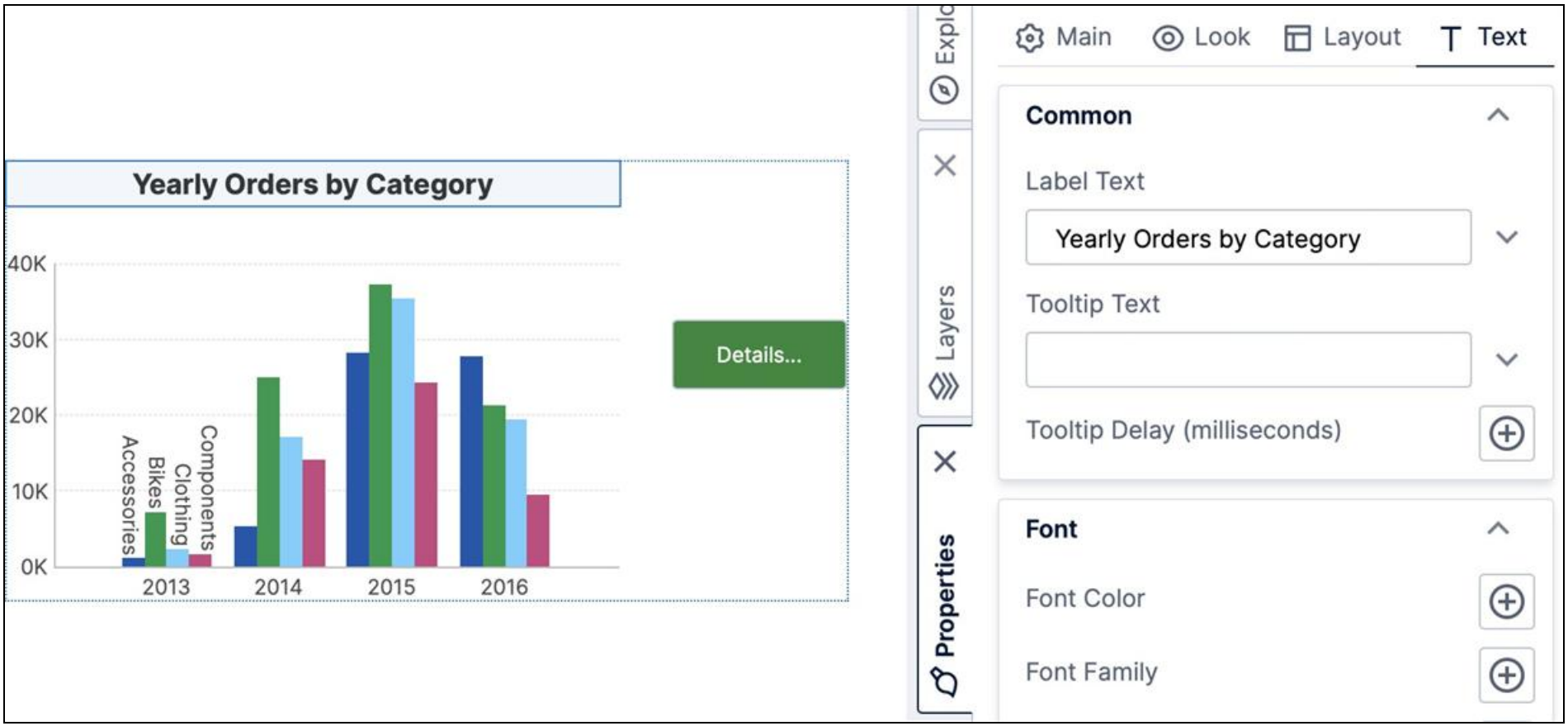


Note: If you've hidden a group and then de-selected it, you can find your group in the Layers window and unhide it or re-select it. See below for details.

Access Group Members

You can still access each element individually when it's part of a group.

After first clicking one of the elements of the group to select the group, click a second time on one of the elements to select it.

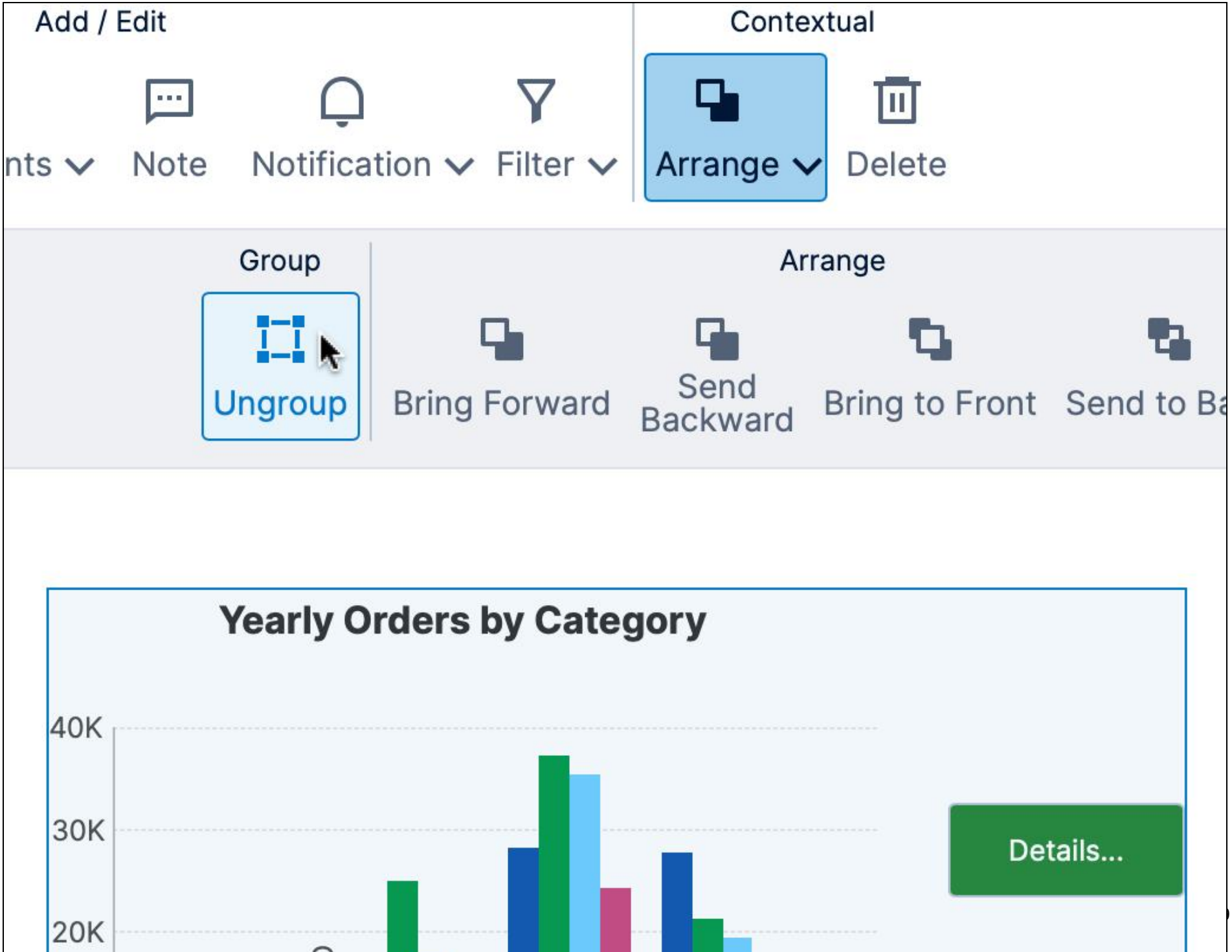


The element will be highlighted separately from the rest of the group, and you can access its options in the toolbar, context menu, and Properties window.

Ungroup

To ungroup a set of elements, first select the group on the canvas.

Then go to the toolbar, click **Arrange**, and then click **Ungroup**.

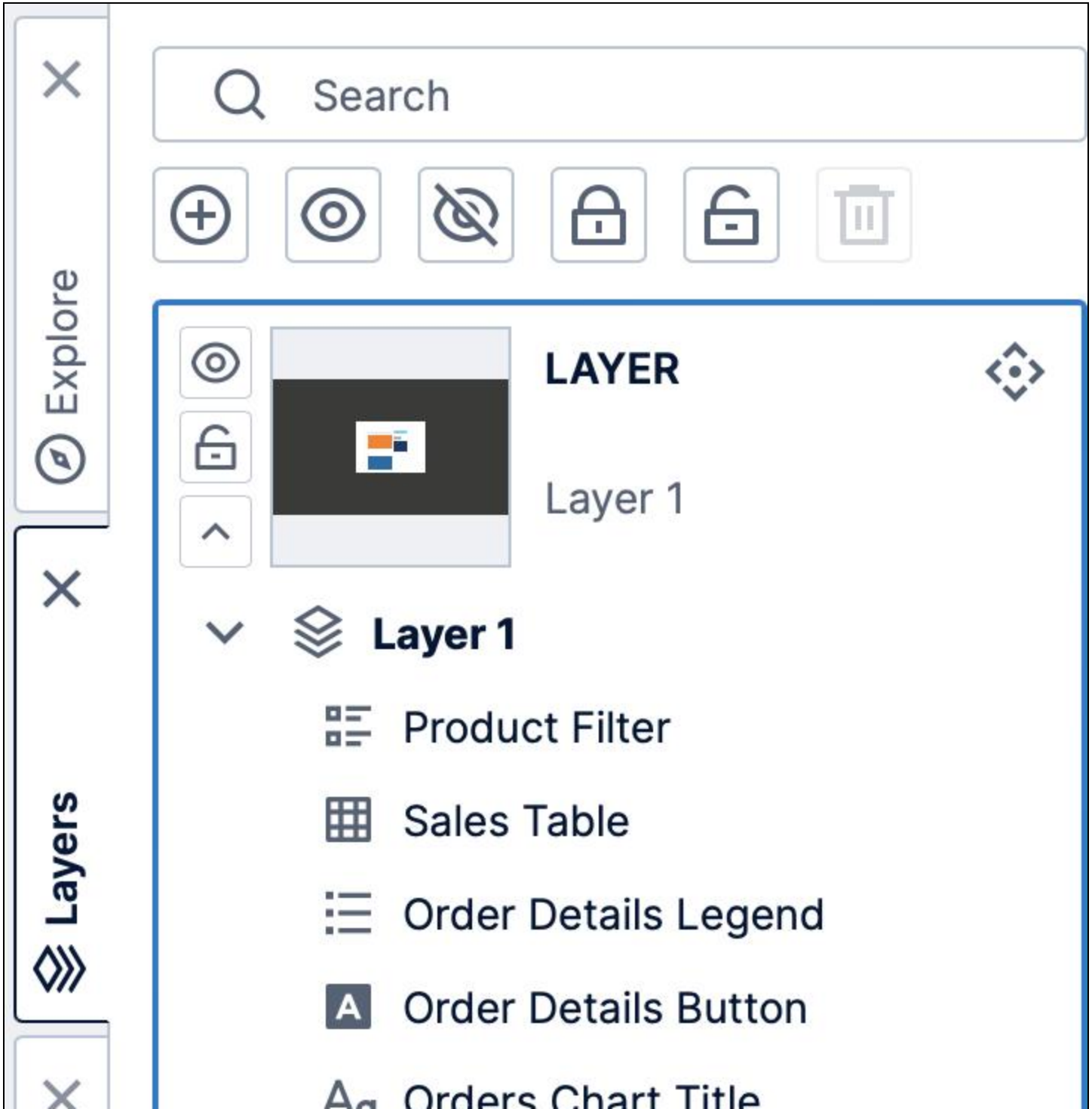

Add / Edit
Contextual
nts
Note
Notification
Filter
Arrange
Delete
Group
Arrange
Ungroup
Bring Forward
Send Backward
Bring to Front
Send to Ba
Yearly Orders by Category
40K
30K
20K
Details...

The group is removed and you can access the elements individually again.

Layers

Whenever you add an element to the canvas, the element is assigned into the current layer. There is always at least one layer in every dashboard, report, or other view, and you can also add new layers and insert elements into them as you like.

Even with only one layer, the **Layers** window is useful for seeing and managing the elements on your canvas.


Explore
Search
LAYER
Layer 1
Layers
Layer 1
Product Filter
Sales Table
Order Details Legend
Order Details Button
Orders Chart Title

Separating elements into multiple layers makes it easy to set up an interactionwith a few clicks to allow viewers to show and hide a layer's content. For example, a common case is to create a help overlay using layers for a dashboard, which is displayed when the viewer hovers over or clicks on a help button. You can also create a layer that will appear over other content like a popup, containing filter options or additional visualizations, for example.

Multiple layers can also be useful for complex views with overlapping elements. As you're designing, you can hide layers that are completed and show just the layers you are actively working on. Layers can also be locked to prevent accidental modification (such as moving a chart out of position by mistake).

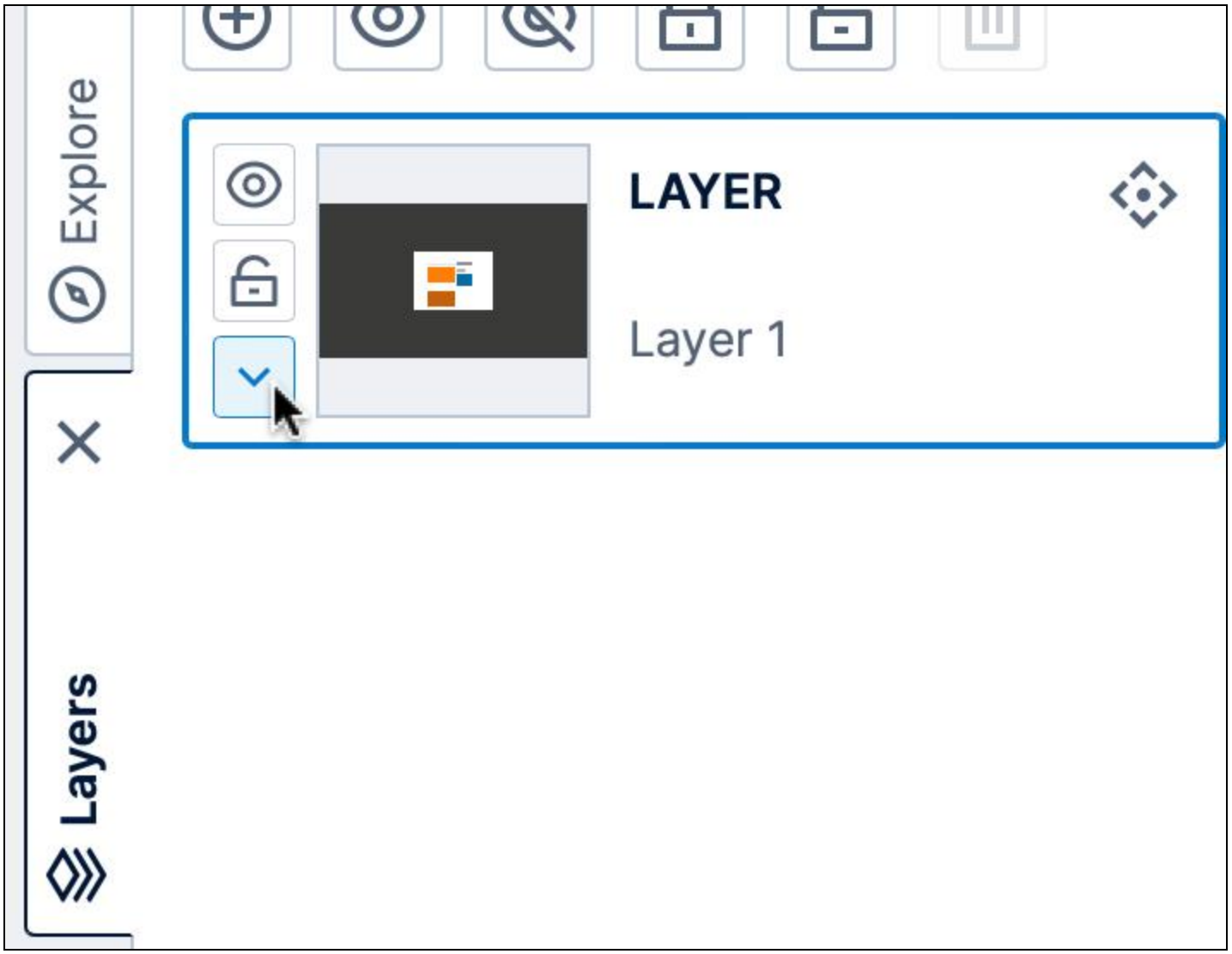
View Elements Within a Layer

A very useful feature in the Layers window is the ability to see a list of all elements in the view grouped by layer. Visualizations, components, groups, scripts, hidden items, and other elements are each listed according to their *z-order*, which means items listed first are displayed on top of other items if they overlap.

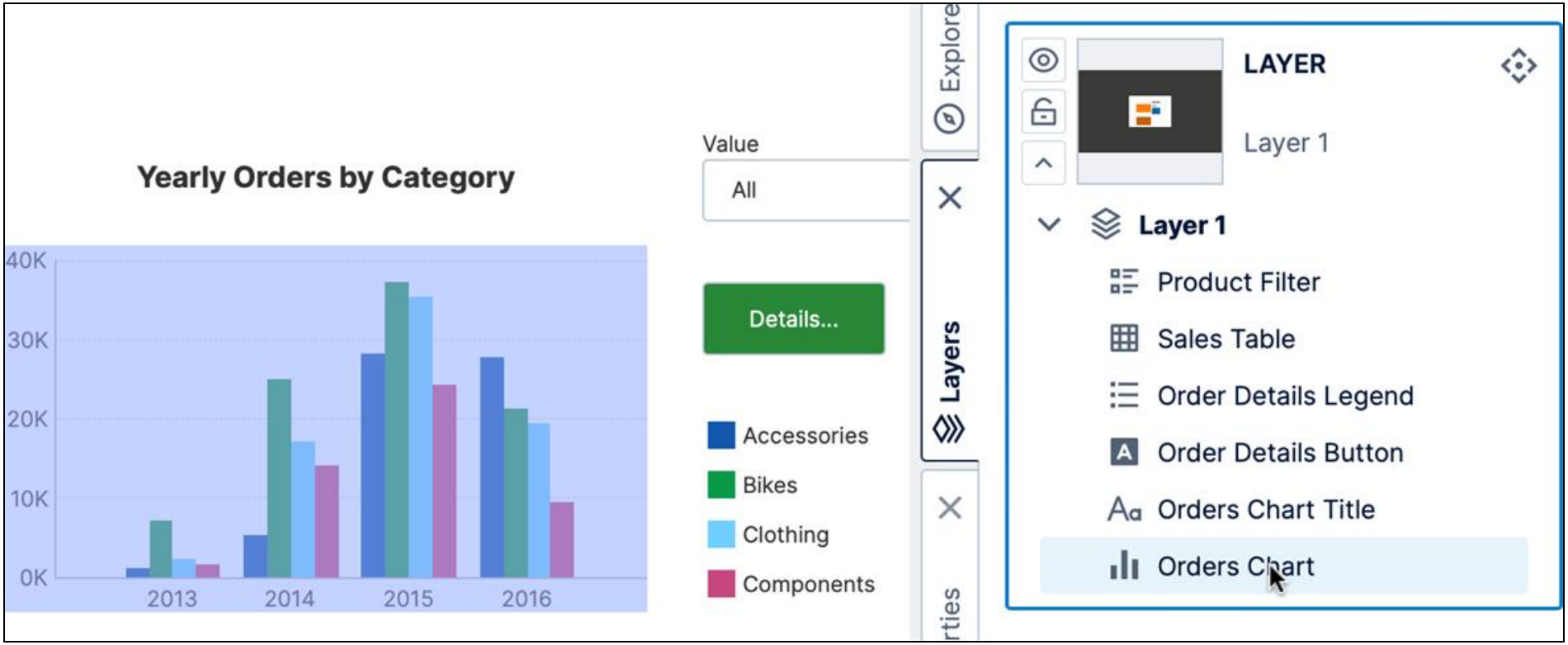
Note: To change the order of elements within a layer, select an element and choose options from **Arrange** in the toolbar such as **Bring To Front**.

In addition, the icon for a layer contains a small preview of elements within the layer.

To expand the list of elements in a layer, click its down arrow button on its left. You can also search for an element by its name at the top of the window.

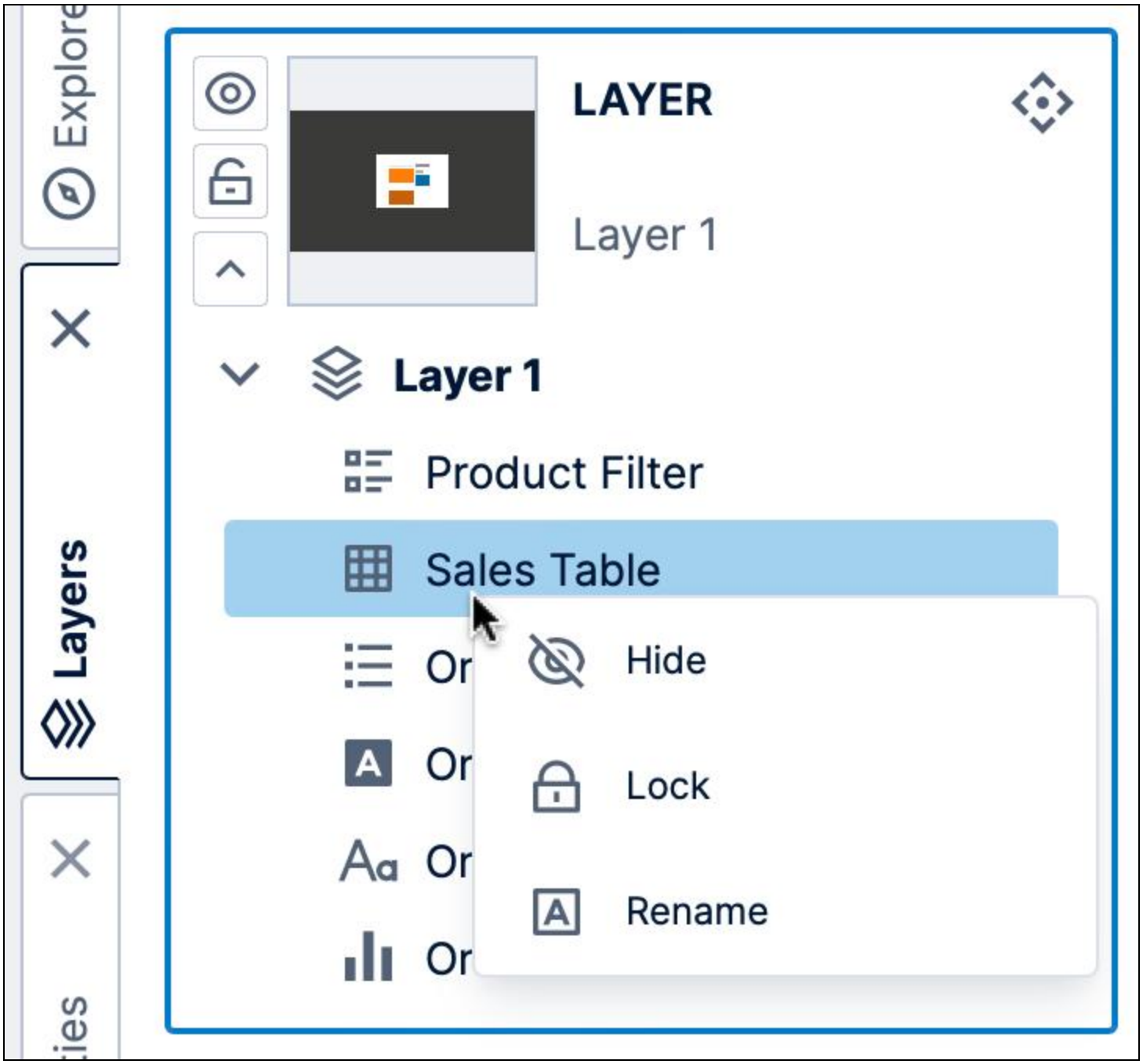


Hover over an item to highlight the corresponding element on the canvas. Similarly, clicking to select one or more items on the canvas will highlight them in the Layers window.



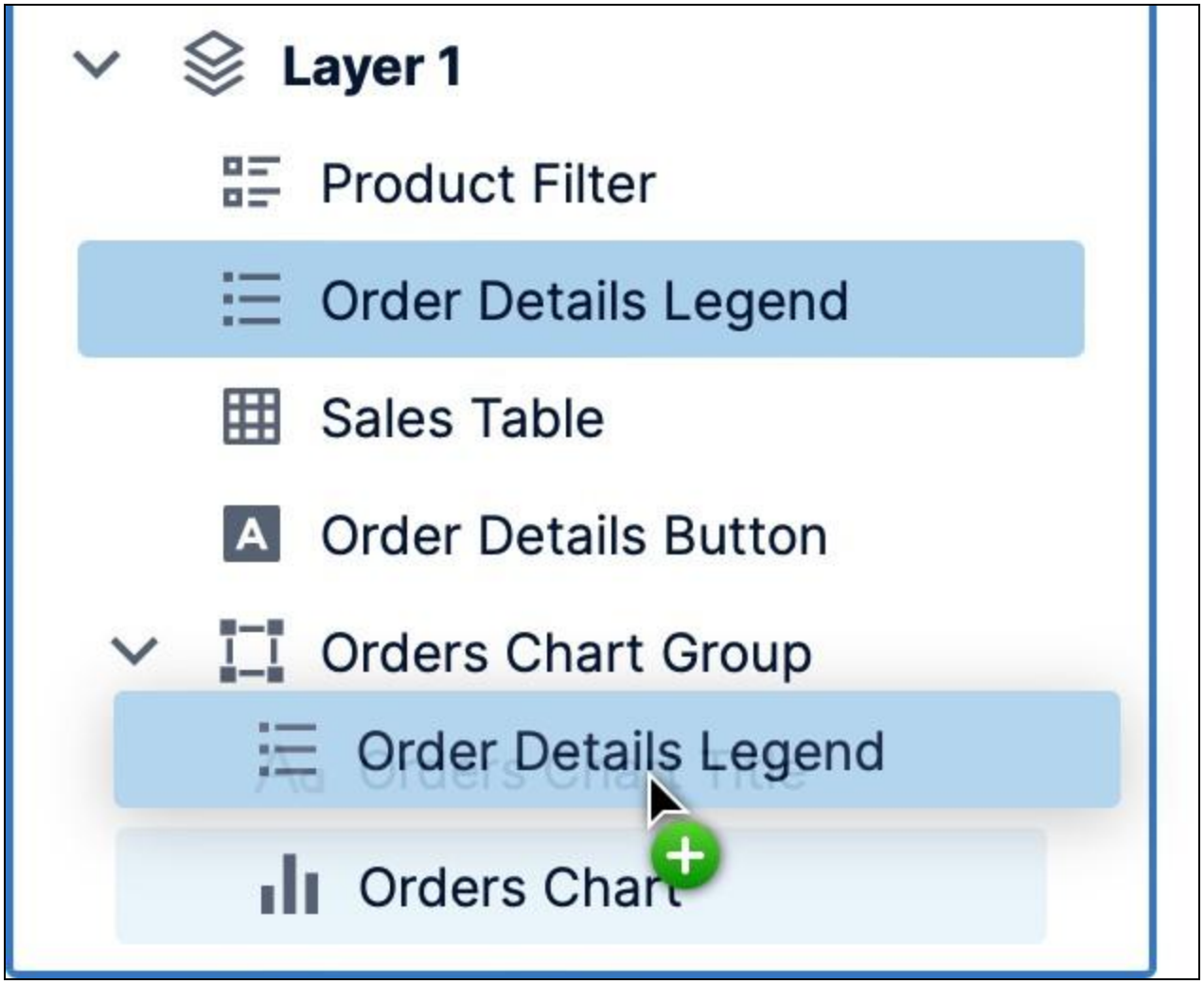
If you click to select an item in the Layers window, it will be selected the same as if you clicked on it directly on the canvas, which means you can switch to the **Properties** window for the element for example. Double-click an item on the list to center your screen on it.

Right-click (or long-tap) an item to access common options such as renaming, showing & hiding, or locking/unlocking.


Explore
Layers
LAYER
Layer 1
Layer 1
Product Filter
Sales Table
Hide
Lock
Rename

Groups are also listed with each of its contained elements listed separately below and indented.

You can click and drag an element into or out of a group.

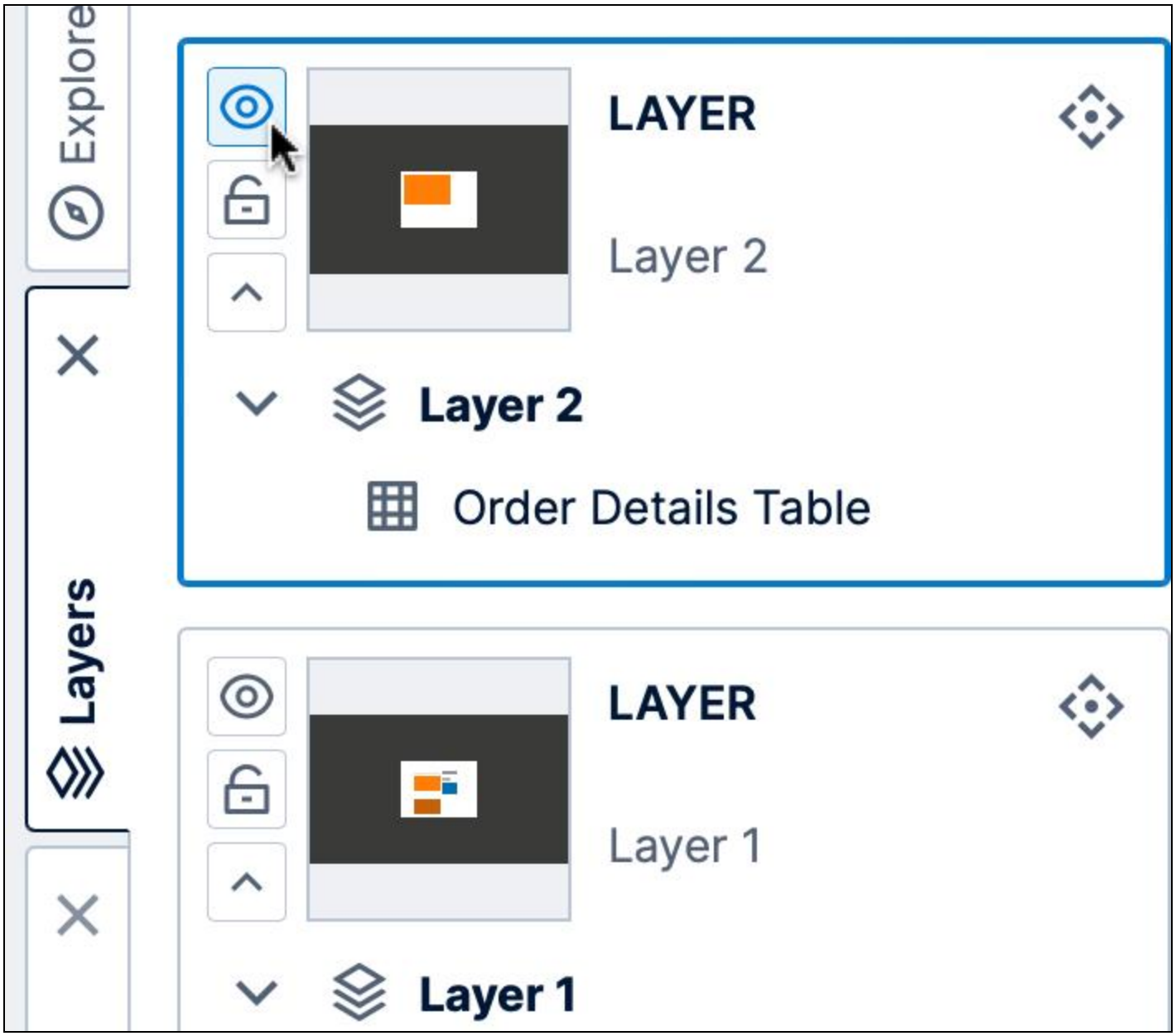


If an item in the list besides a group has an expander arrow, it has an interaction or script assigned to it.

Click to expand the item to view its actions, each with its associated event in parentheses (e.g., click). Click an interaction to open its dialog, or on script to open it in the Script Editor.

Show or Hide a Layer

To show or hide a layer, toggle the eye icon on its left.



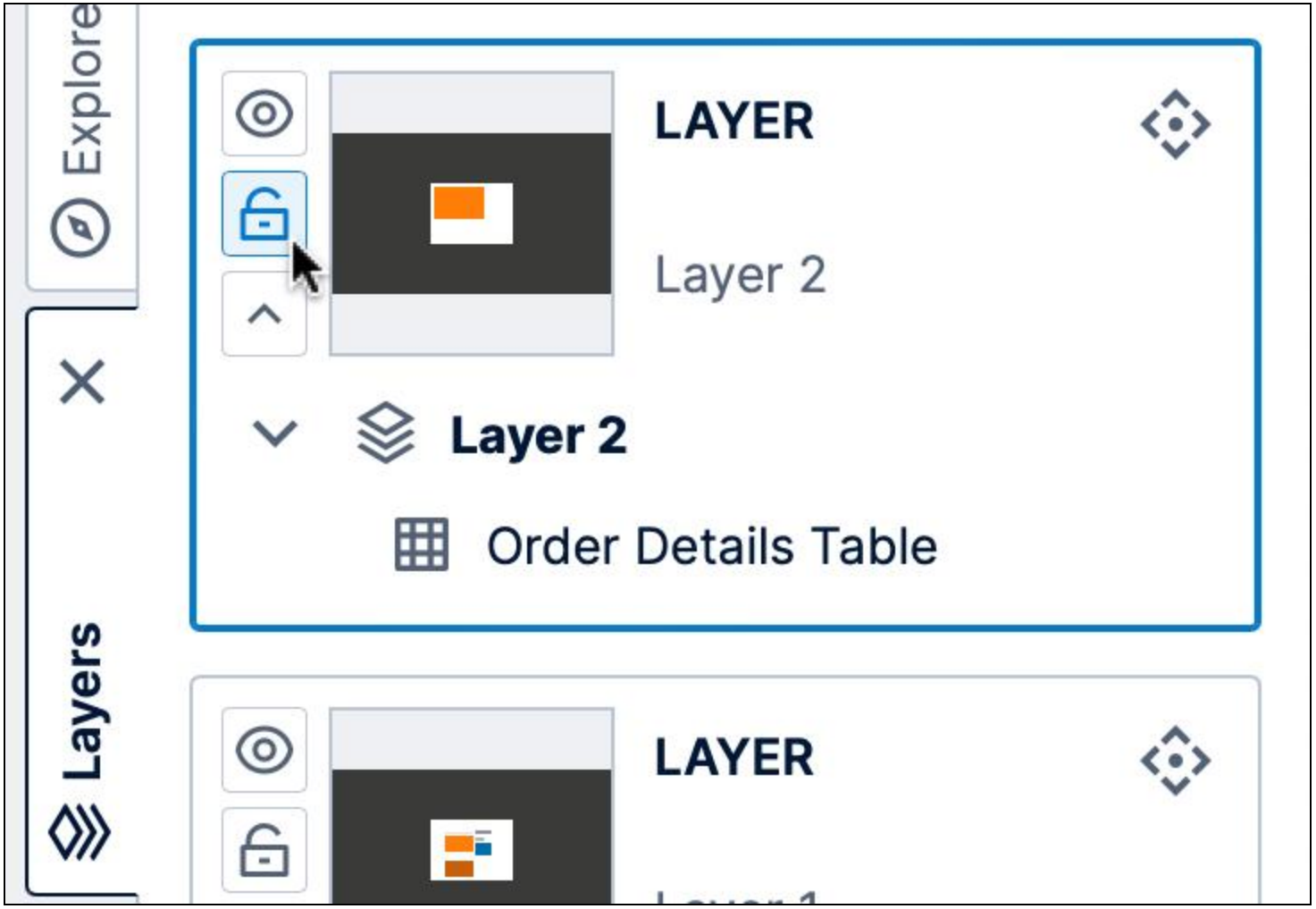
To show or hide all layers at once, click the corresponding button at the top of the Layers window.

You can also set up interactions to show and hide layers, which will allow each viewer to interactively show and hide content.

Lock or Unlock a Layer

Layers can be locked to prevent accidental modification (e.g., moving a chart out of position by mistake).

To lock or unlock a layer, toggle the lock icon on its left.

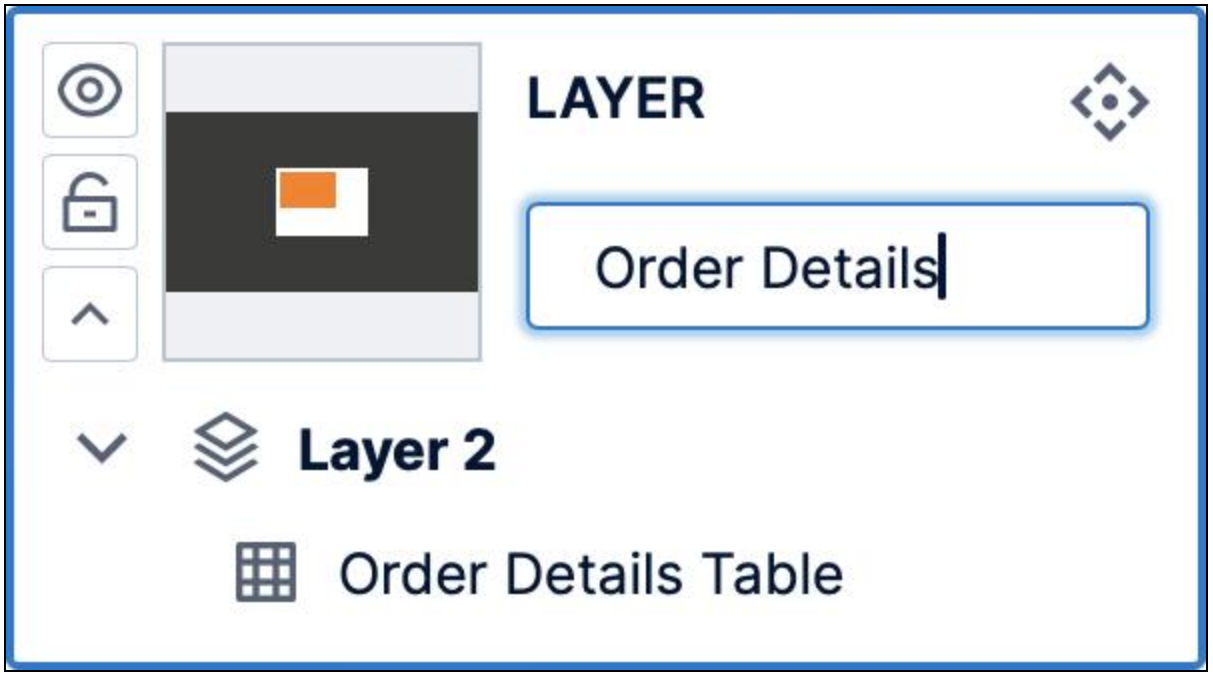


To lock or unlock all layers in one operation, click the corresponding button at the top of the Layers window.

Rename a Layer

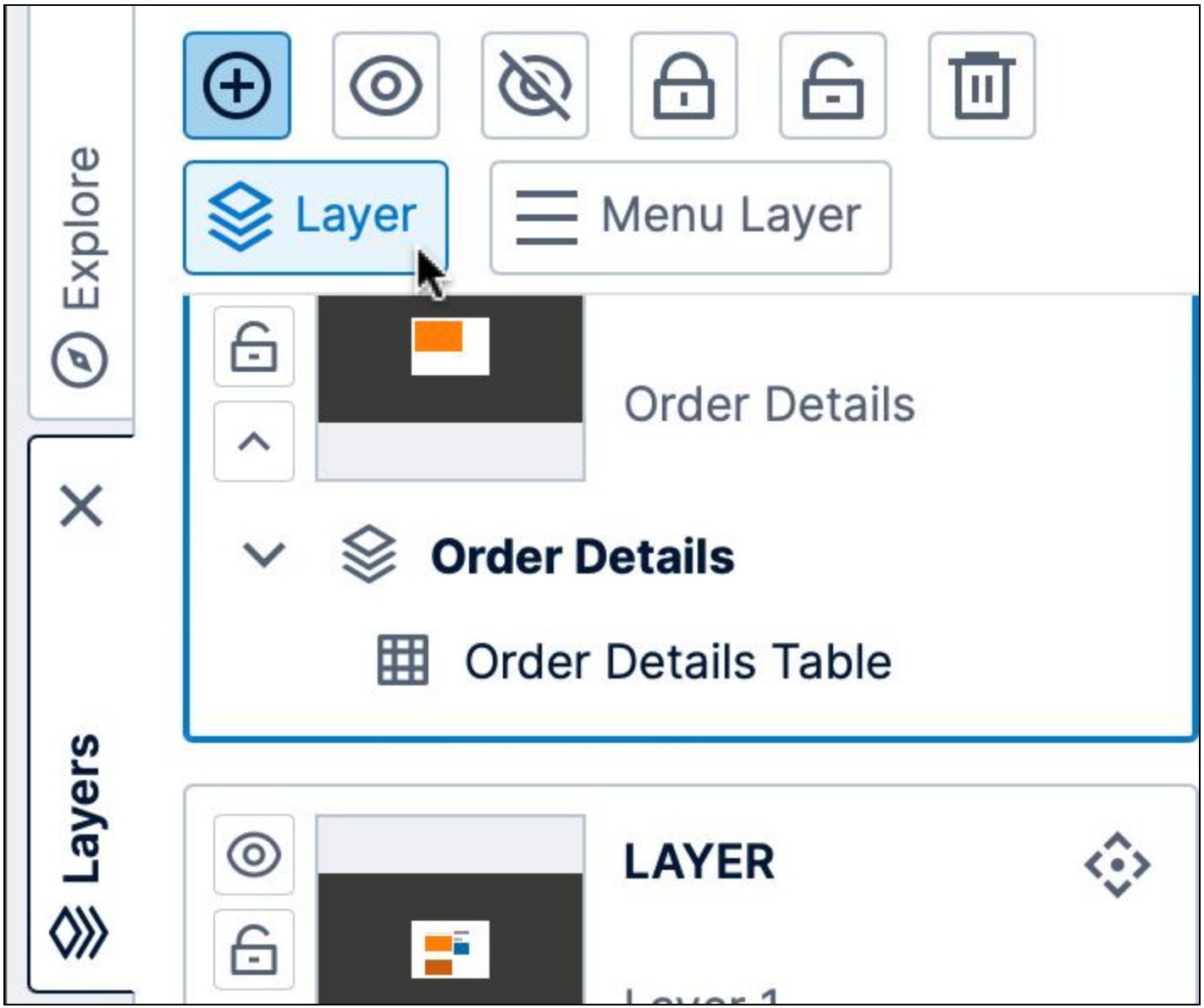
Double-click the name of a layer to change it, then press the enter/return key or click elsewhere.

You can also right-click and choose **Rename** from the context menu, or you can use the Properties window to rename it.



Add a New Layer

In the Layers window, click the **Add** button in the toolbar.



For dashboards, you then choose between a regular layer and a menu layer, described below. In most cases, you can choose the first option to create a regular layer.

When added, the new layer will be positioned above existing layers in terms of *z-order*. It will also be set as the new current layer as indicated by its blue selection border. When you add new content, it will be assigned to this layer.

The menu layer option can be useful for dashboards that use a template grid, because it contains only a single template cell that is separate from the dashboard's main template grid. You could use this for dashboards with menu navigation that should span the full width or height of your dashboard, for example. For more information, see the article on menu layers.

Reorder the List of Layers

In the Layers window, you can drag a layer and move it above another layer in the list. This will change the z-order of the layers. The elements of a layer higher in the list will always appear on top of elements in a layer below.

Set the Current Layer

The current layer is shown as selected in the Layers window (appearing in blue). New elements that you add to the canvas will go into this current layer by default.

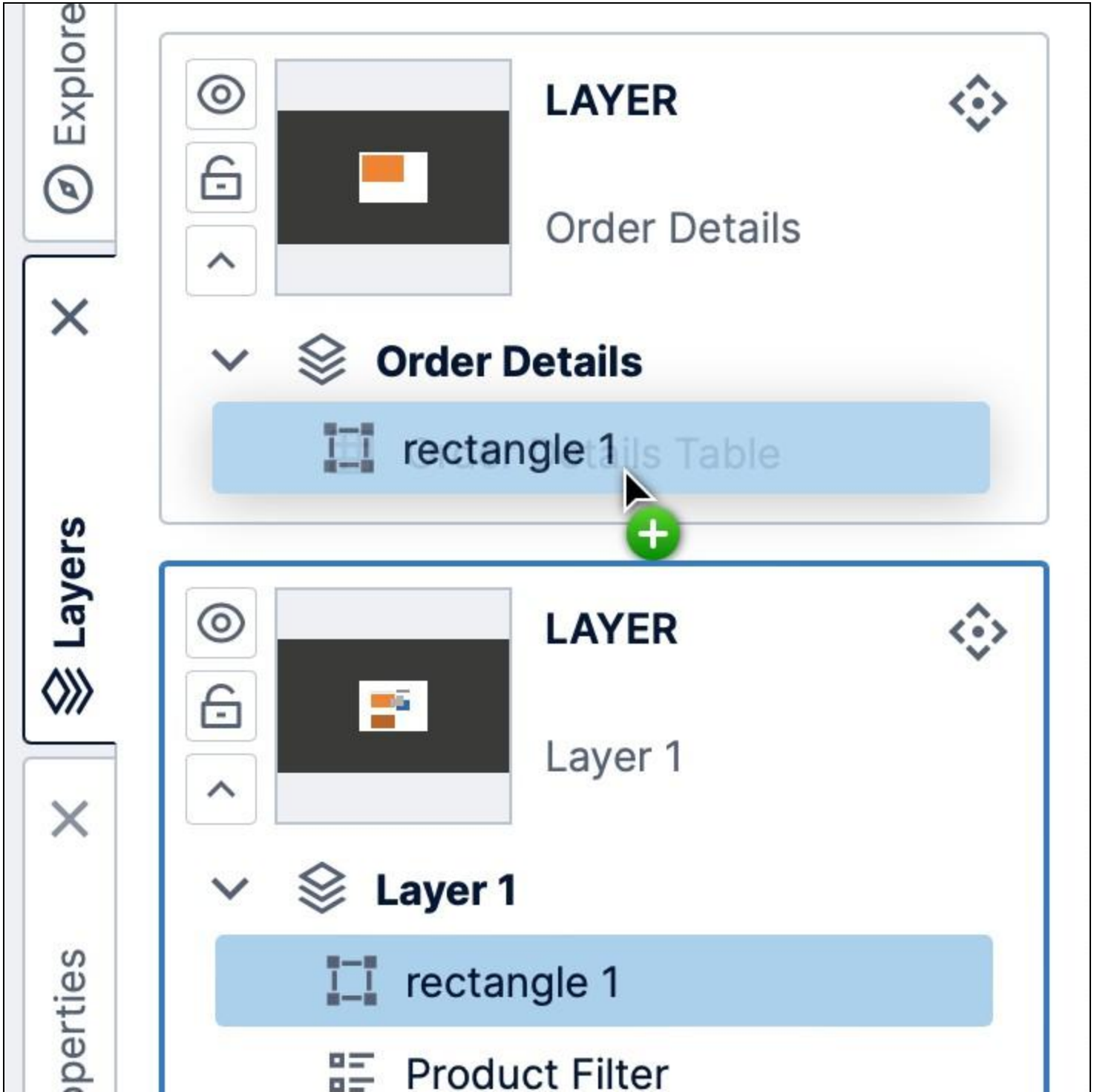
To change the current layer, click any part of the layer you want to select in the Layers window.

Delete a Layer

To delete a layer (and all of its contained elements), select the layer in the Layers window and then click **Delete** at the top of the Layers window.

Move an Element From One Layer to Another

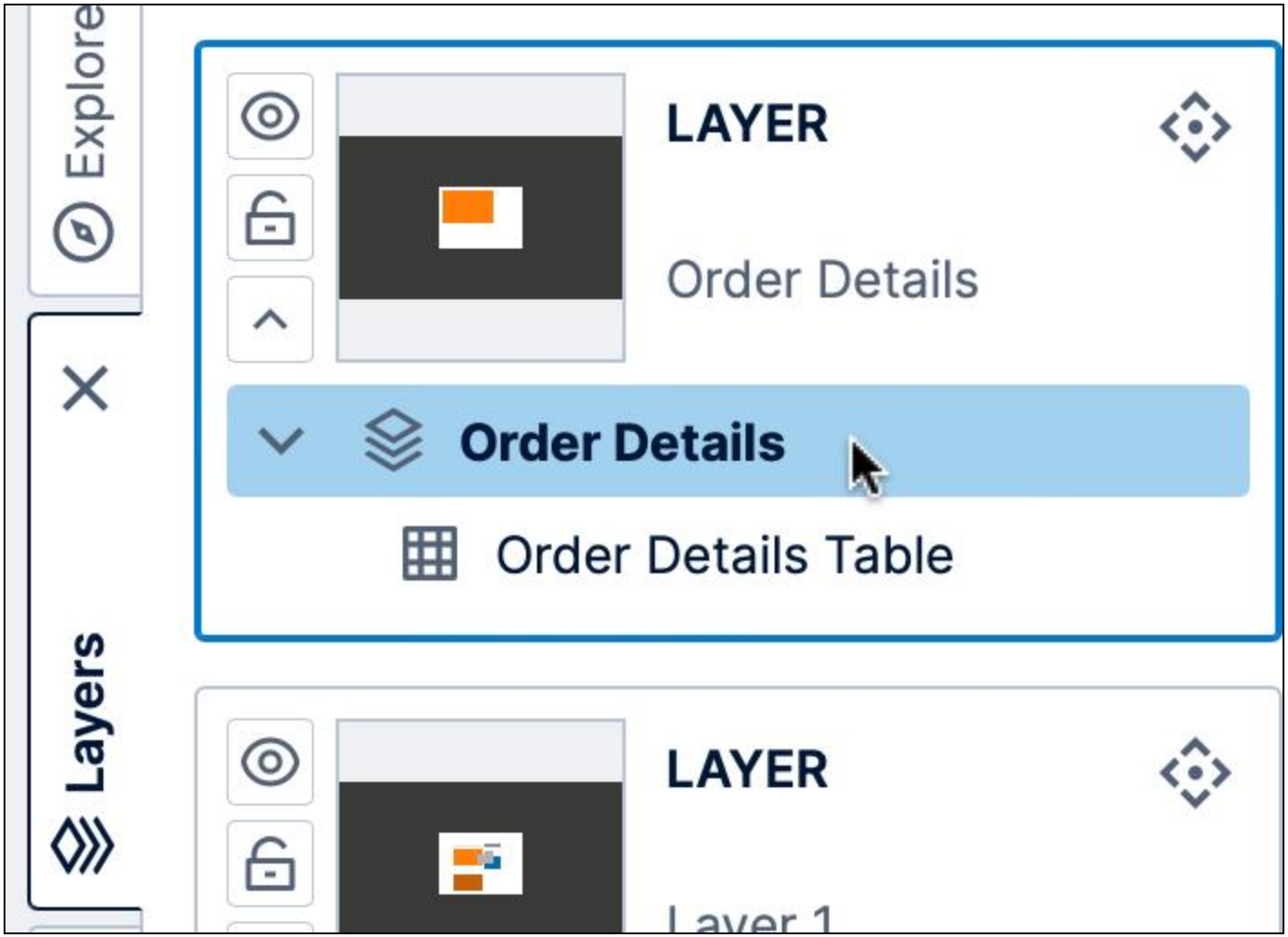
In the Layers window, after clicking the down arrow buttons to list the elements of two layers, you can drag an element from one and drop it on the other to move it.


Explore
Layers
Properties
LAYER
Order Details
Order Details
rectangle 1
LAYER
Layer 1
Layer 1
rectangle 1
Product Filter

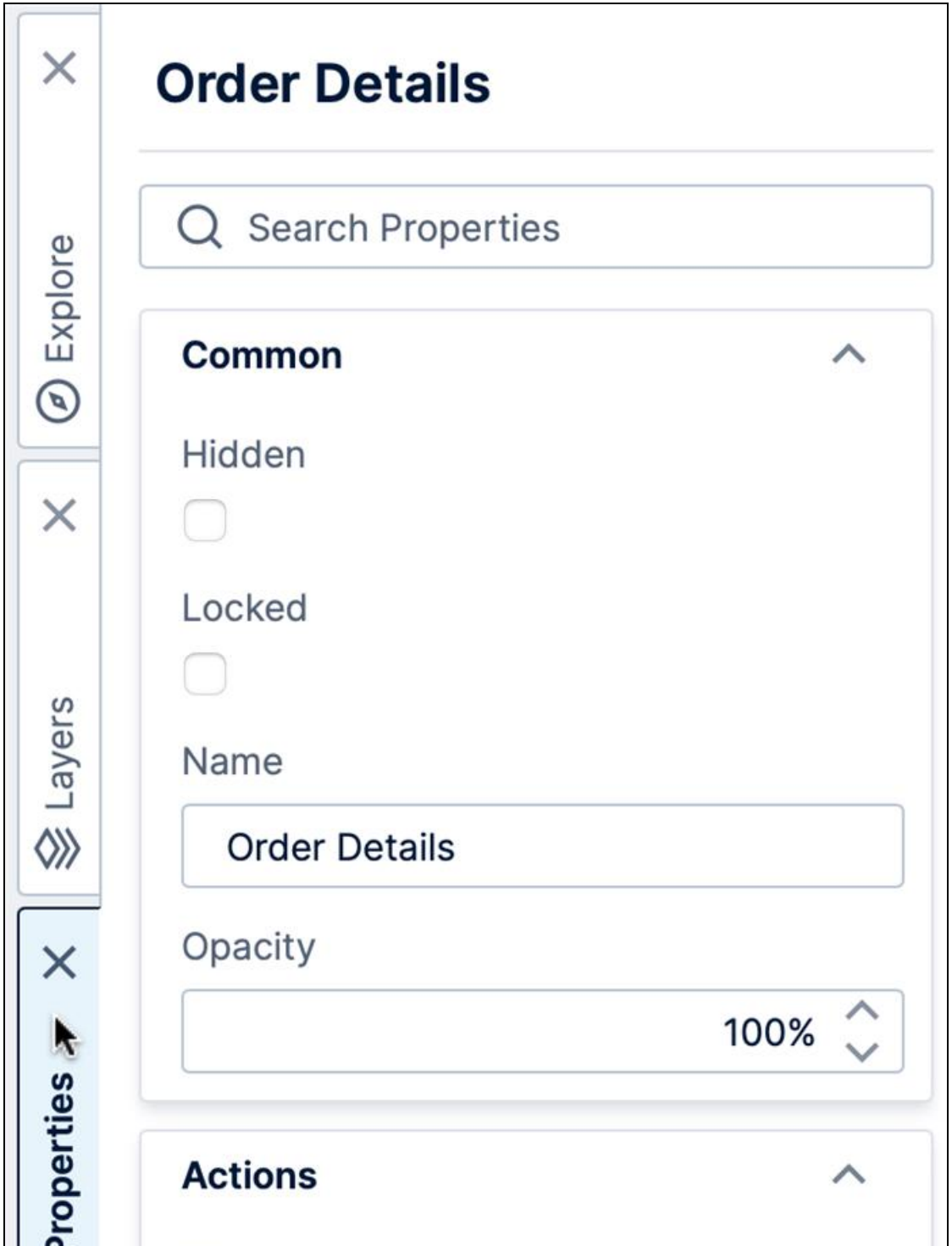
Alternatively, right-click or long-tap an element and choose **Send To Layer**, then choose from the list of layers.

Properties of a Layer

To modify the properties of a layer, go to the **Layers** window. Expand the layer's list of elements by clicking its down arrow button, and then click to select the layer itself at the top of the list.



Now you can go to **Properties** to view or modify the properties of the layer, including its **Script Name**.


Order Details
Search Properties
Common
Hidden
Locked
Name
Order Details
Opacity
100%
Actions
Explore
Layers
Properties

For more information, see:

- Video: Group and Layers
- Design Overview
- Using Dockable Windows
- Using Interactions
- Create a Dashboard Template
- Using a Template Grid for Resizing

Set Up a Filter Interaction

This applies to: *Managed Dashboards, Managed Reports*

This article shows you how to set up a data visualization with a filter interaction so that when data is clicked or tapped, other visualizations with compatible data can be automatically filtered to the same selection.

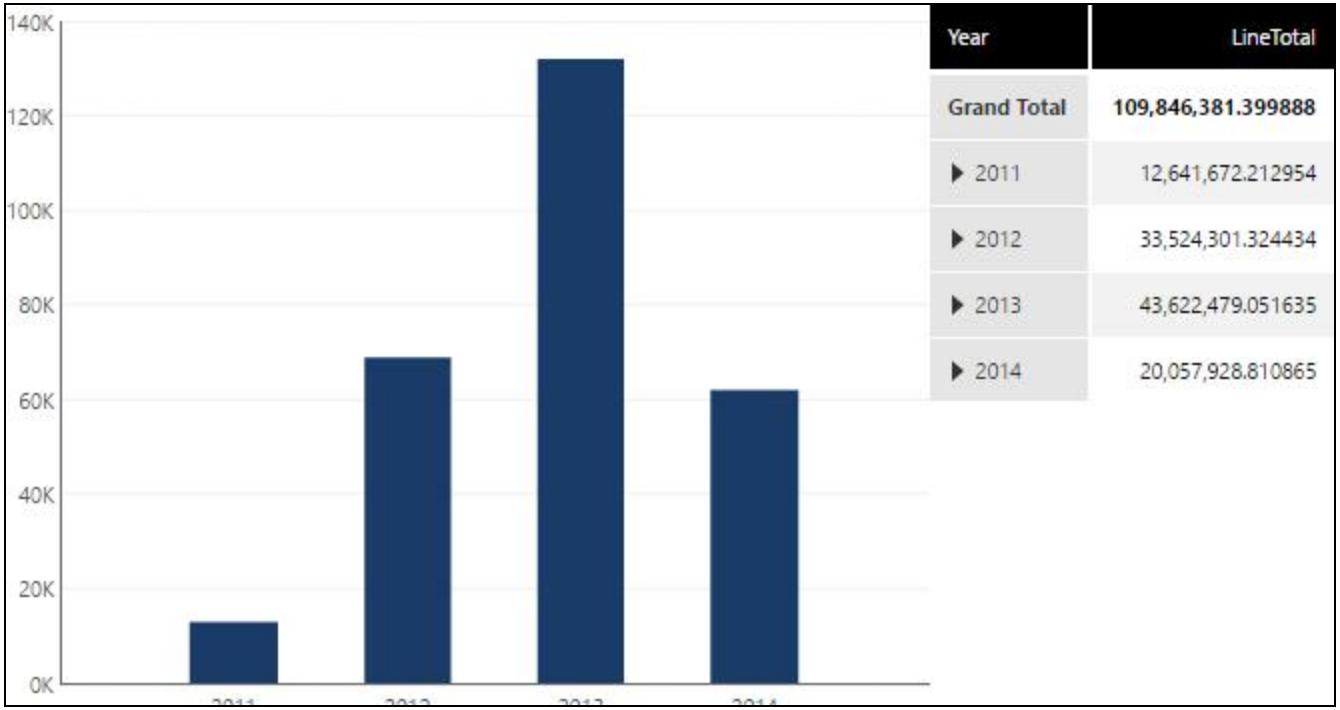
You can also edit the parameter mappings to customize what data is passed and where during the interaction. For non-visualization controls such as buttons, you can apply a predefined filter value instead.

Related video: Interactions

Create a Dashboard

For this example, create a metric set and add it to a new dashboard in order to show *OrderQty by Date* in the form of a bar chart.

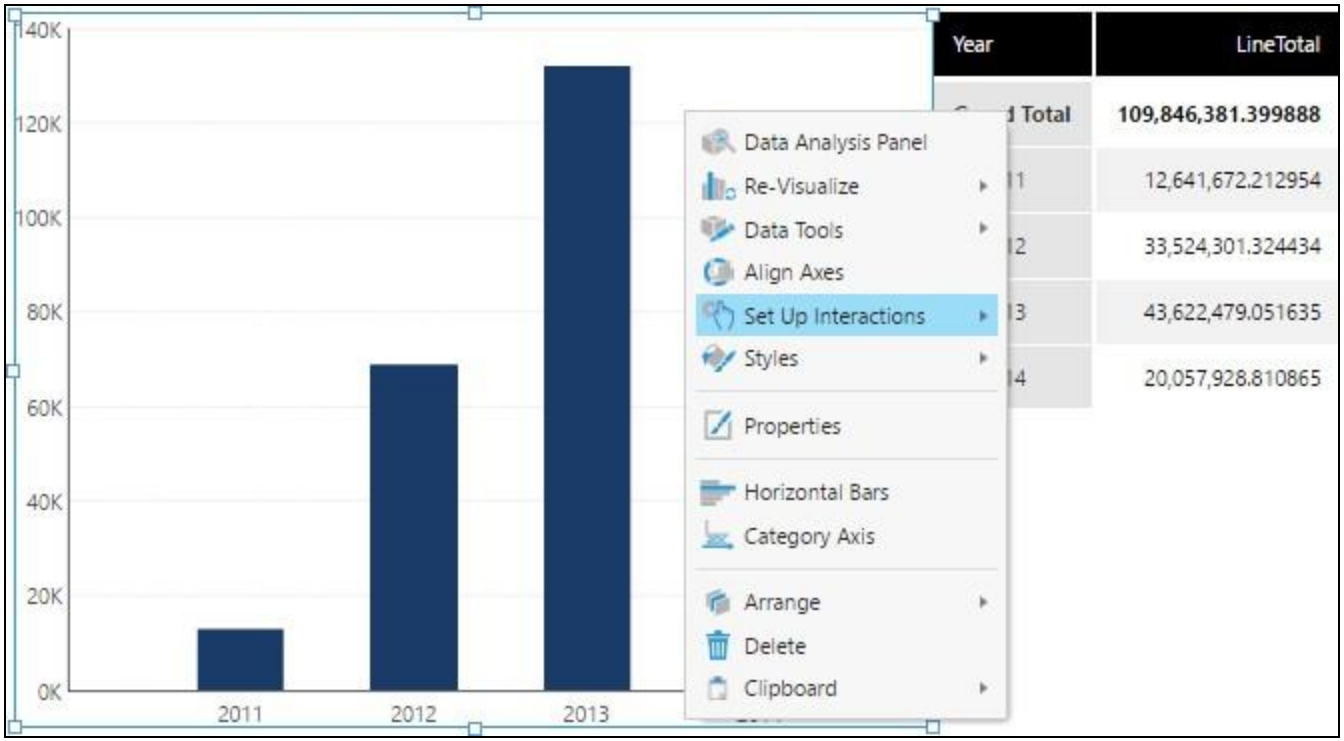
Next, create a second metric set and add it to the dashboard in order to show *LineTotal by Date* in the form of a table visualization.



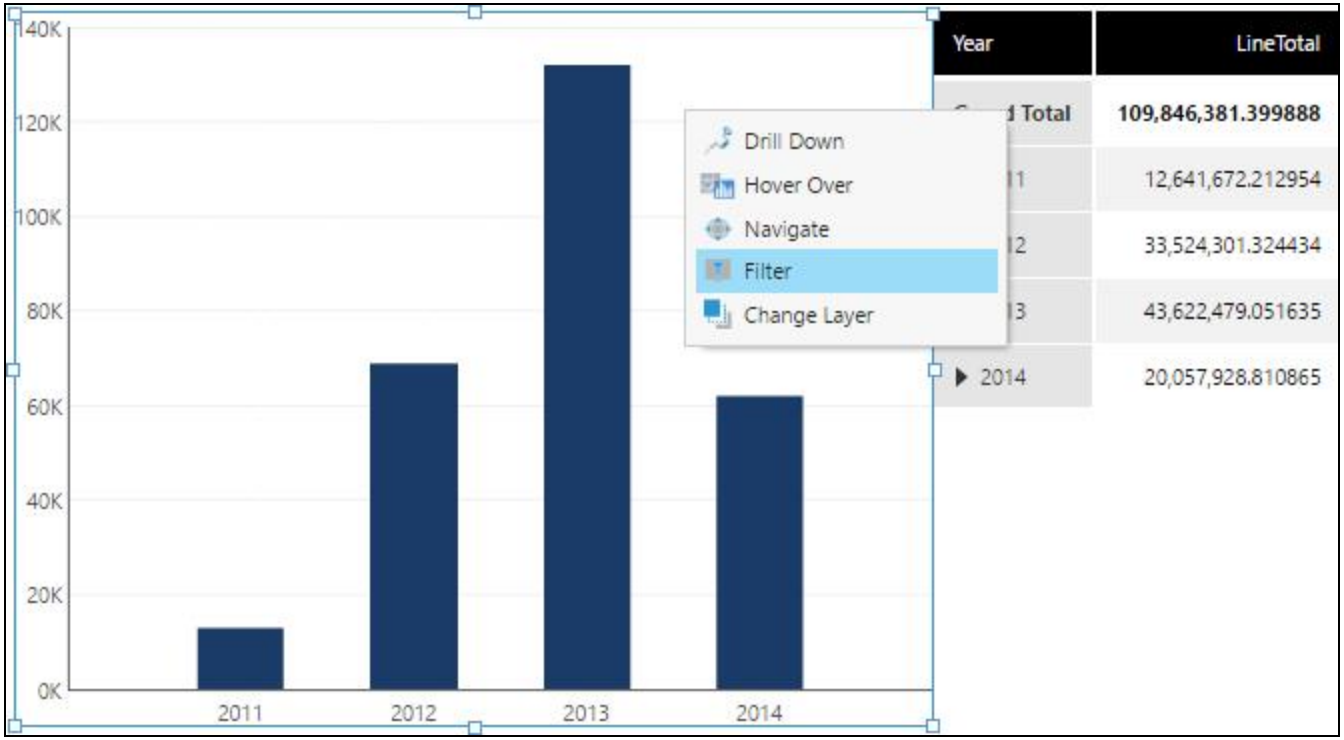
Note that these two metric sets have compatible time hierarchies.

Set Up the Filter Interaction

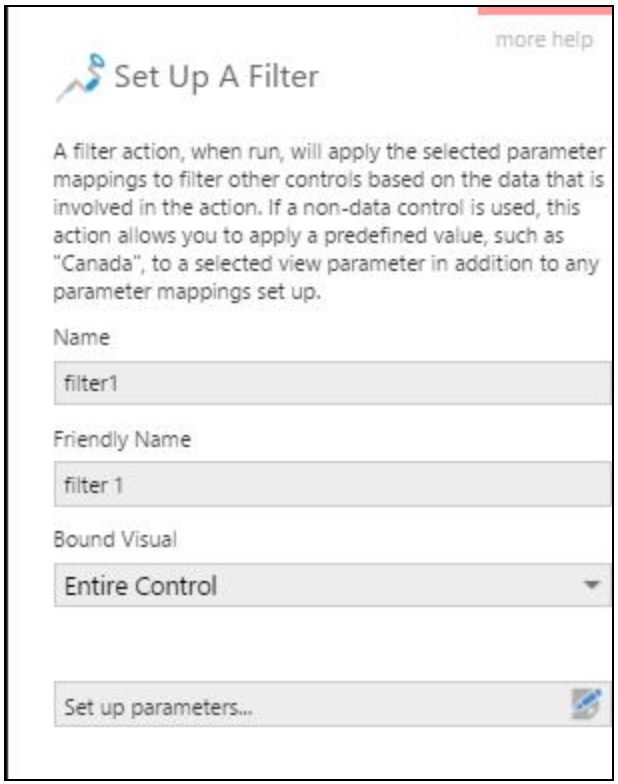
Right-click the bar chart and select **Set Up Interactions** (or click to select the chart and choose the same option in the toolbar).



From the submenu, select **Filter**.



The filter setup dialog is displayed.



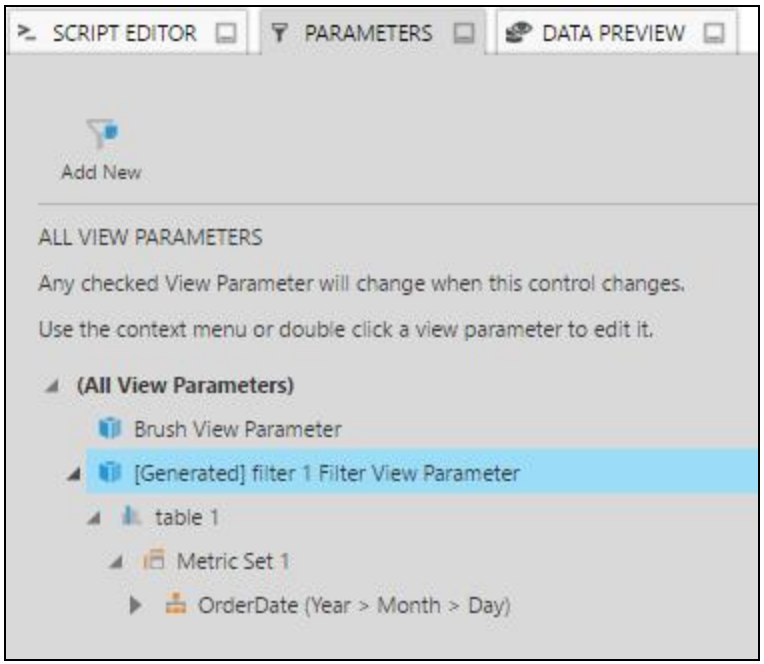
In most cases, you can just leave the default settings and click to **Save** at the bottom of the dialog. The interaction will be set up for you automatically based on compatible data added to the Data Analysis Panel for visualizations on the dashboard. You can test the filter interaction by switching to **View** mode and clicking on the bar chart.

Important: Interactions and their parameters work ith data added to the Data Analysis Panel for visualizations. For example if you want to filter other visualizations, first add the data you want to filter by to **Slicers** or another option in the Data Analysis Panel.

The next section is normally optional and shows you how this filtering was automatically configured by Symphony. You can follow along to get a better understanding of how it works.

Filter View Parameter Mapping

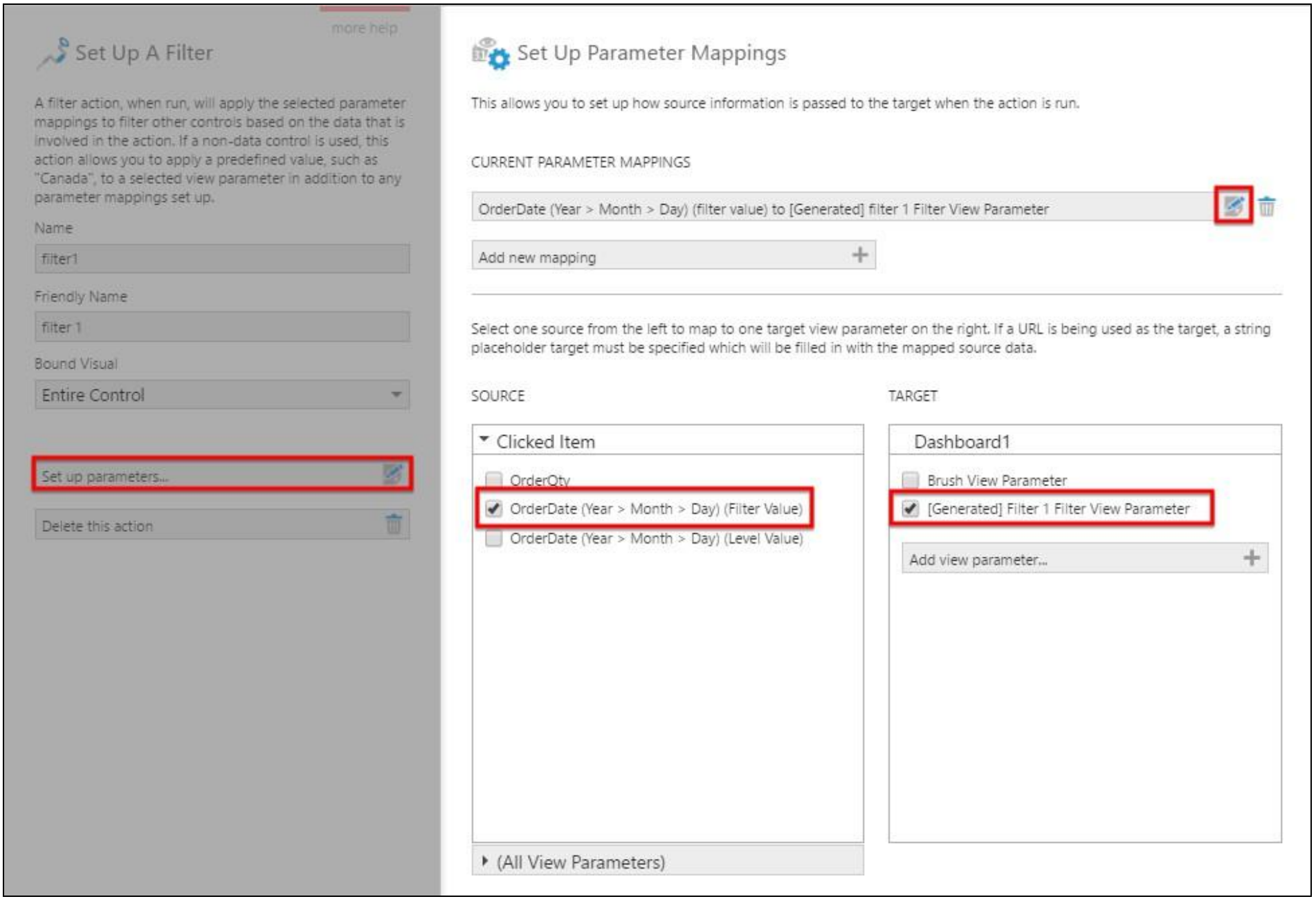
In the dashboard editor, click to open the **Parameters** window, which is docked on the bottom by default. Look for a filter view parameter item and expand it. This view parameter was automatically generated by the filter setup dialog and connected to the *Date* hierarchy on the second metric set (table visualization).



Open the filter setup dialog again and click **Set up parameters**.

In the parameter mappings setup dialog that opens, you'll see that a filter view parameter mapping has already been set up.

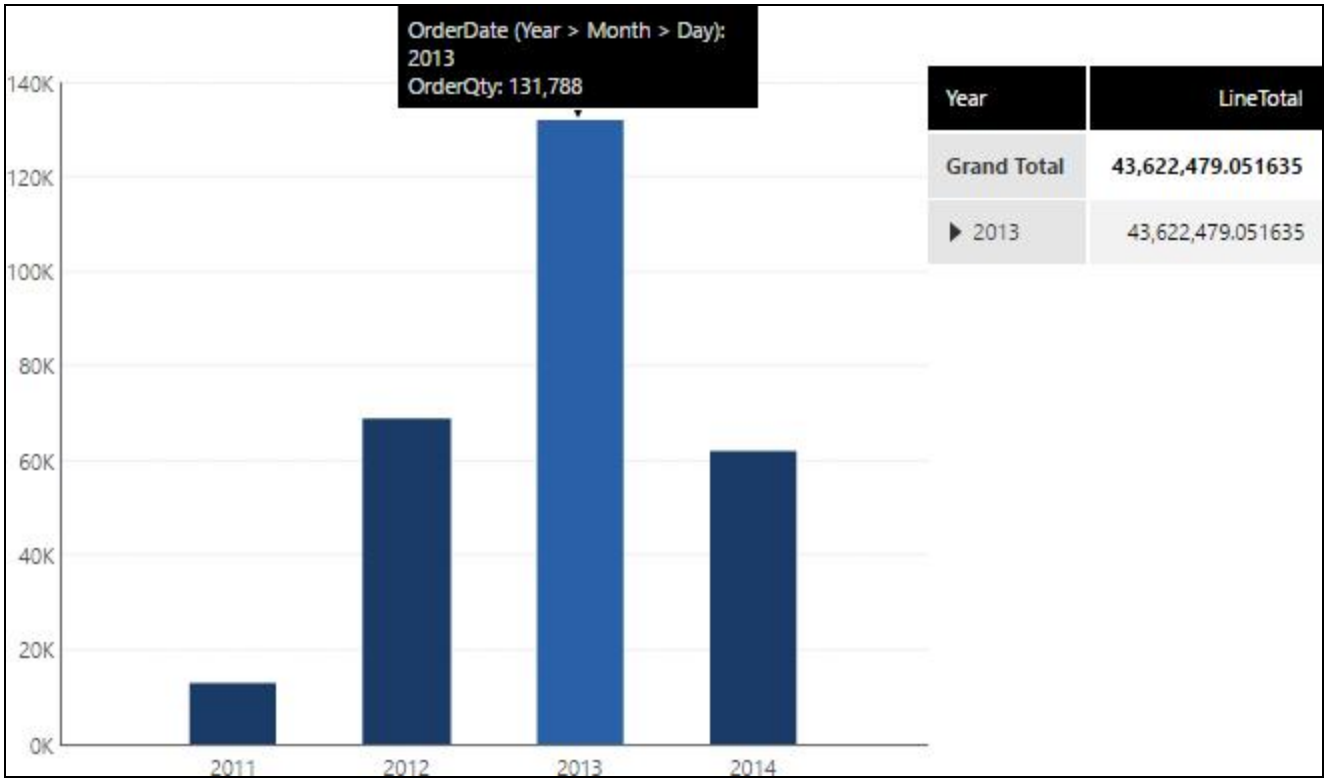
Click the **Edit** icon for this mapping to see its details.


more help
Set Up A Filter
A filter action, when run, will apply the selected parameter mappings to filter other controls based on the data that is involved in the action. If a non-data control is used, this action allows you to apply a predefined value, such as "Canada", to a selected view parameter in addition to any parameter mappings set up.
Name
filter1
Friendly Name
filter 1
Bound Visual
Entire Control
Set up parameters...
Delete this action
Set Up Parameter Mappings
This allows you to set up how source information is passed to the target when the action is run.
CURRENT PARAMETER MAPPINGS
OrderDate (Year > Month > Day) (filter value) to [Generated] filter 1 Filter View Parameter
Add new mapping
Select one source from the left to map to one target view parameter on the right. If a URL is being used as the target, a string placeholder target must be specified which will be filled in with the mapped source data.
SOURCE
TARGET
Clicked Item
OrderQty
OrderDate (Year > Month > Day) (Filter Value)
OrderDate (Year > Month > Day) (Level Value)
Dashboard1
Brush View Parameter
[Generated] Filter 1 Filter View Parameter
Add view parameter...
(All View Parameters)

In the **Source** list, the *OrderDate* filter checkbox is selected so that when a date is clicked on the bar chart, it will be passed to the filter view parameter in the **Target** list and filter the table visualization. This filter view parameter is the generated one that appears in the Parameters window.

Test the Interaction

Switch to **View** mode and click a data point on the bar chart. You should see the table visualization filtered to the same time value.



For more information, see:

- Using Interactions
- Set Up a Hover Over Interaction

- Use a Button to Apply a Filter Value
- Design Overview

Modify a Filter / View Parameter Using Scripting

This applies to: *Managed Dashboards, Managed Reports*

The goal of this article is to demonstrate passing various types of values to view parameters through script when you don't want to use a built-in interaction.

Live samples can also be viewed by visiting our script library.

What is a View Parameter?

A view parameter passes parameter values selected through filter controls or interactions into the parameters of metric sets. Filter controls and interactions can be set up without script, but script can be used to set parameter values as part of custom functionality.

Types of View Parameters

There are many types of parameter values available, which can be found in our JavaScript API Reference. We will be using the following most common types in this article:

- RangeNumber (Range Number filter)
- HierarchyLevel (Level filter)
- TimeHierarchyOffset (Time Lag filter)
- CollectionMember (Hierarchy Value, Cascading, Checkbox List filter)
- SingleMember (Calendar filter)
- RangeMember (Hierarchy Value Range, Calendar Range filter)
- RangeDateTime (Range Date filter)

What are Tokens?

Tokens are predefined values that can be accessed using the menu to the right of most filter controls. The tokens available for a given filter will depend on the data type, and whether you're setting the From or To value in the case of a range. If the data type is of a time hierarchy, various convenient choices are available such as Today, Current Month, Previous Year, End of Year, etc.

Note: Using a range token in a single date/time filter will default to the beginning of that range (e.g., *Current Year* will be January 1st of that year).

Parameters Values and Token Values

RangeNumber

```
// In a view's (e.g., dashboard's) ready or load actions, use this rather than this.parentView:
var viewParameter = this.parentView.control.getViewParameterByName("viewParameter1");

// Remove all values and tokens from the parameter value
viewParameter.parameterValue.clearTokens();
viewParameter.parameterValue.clearValues();

// Set the lower and upper values
viewParameter.parameterValue.lowerBoundaryValue = 100;
viewParameter.parameterValue.upperBoundaryValue = 200;

// Set the last modified time so this parameter takes precedence over any others
viewParameter.invalidateParameterValueLastModifiedTime();

// Update all the connected adapters with the newly modified values
// Includes data visualizations and filter controls
// Pass the current view in case it's embedded in another
```

HierarchyLevel

```
// In a view's (e.g., dashboard's) ready or load actions, use this rather than this.parentView:
var viewParameter = this.parentView.control.getViewParameterByName("viewParameter1");

// Set the depth of the hierarchy to be displayed
viewParameter.parameterValue.detailsLevelDepth = 1;

// Set the top level if you wish to display more than 1 level
// Default is -1 which will use the details level depth instead
viewParameter.parameterValue.topLevelDepth = 0;

// Set the last modified time so this parameter takes precedence over any others
viewParameter.invalidateParameterValueLastModifiedTime();
```

```
// Update all the connected adapters with the newly modified values
// Includes data visualizations and filter controls
// Pass the current view in case it's embedded in another
viewParameter.refreshAllAdapters(null, this.parentView);
```

TimeHierarchyOffset

```
// In a view's (e.g., dashboard's) ready or load actions, use this rather than this.parentView:
var viewParameter = this.parentView.control.getViewParameterByName("viewParameter1");

// Set the offset of the Period over Period to be displayed
// Negative values will set the offset 'Backward'
viewParameter.parameterValue.offset = -1;

// Level value format = "Level.Code.TimeDimensionName"
// Day=0 Week=1 Month=2 Quarter=3 Half=4 Year=5
// Example: "2.54320.DueDate"

// Only modify the level index by taking a substring
var levelUniqueName = viewParameter.parameterValue.level;
var endOfLevel = levelUniqueName.substring(levelUniqueName.indexOf('.'));

// Set the level to the level number and concatenate endOfLevel
viewParameter.parameterValue.level = "5" + endOfLevel;
// This will result in Backwards 1 Year

// Set the last modified time so this parameter takes precedence over any others
viewParameter.invalidateParameterValueLastModifiedTime();

// Update all the connected adapters with the newly modified values
// Includes data visualizations and filter controls
// Pass the current view in case it's embedded in another
viewParameter.refreshAllAdapters(null, this.parentView);
```

CollectionMember

```
// In a view's (e.g., dashboard's) ready or load actions, use this rather than this.parentView:
var view = this.parentView;
var viewParameter = view.control.getViewParameterByName("viewParameter1");
```

```
// Remove all values and tokens from the parameter value
viewParameter.parameterValue.clearTokens();
viewParameter.parameterValue.clearValues();

// One option is to set all the necessary properties of the new member.
// One way to get this information is to use the developer console
// to read a view parameter's parameter value after it's set in the UI.

// Below is an example of a member value from a flat hierarchy.
// Create a new MemberValue to push onto the values array.
var memberValue = new dundas.data.MemberValue({
       "hierarchyUniqueName": "Name",
       "levelUniqueName": "Name",
       "memberKind": dundas.data.MemberKind.REGULAR,
       "uniqueName": "Cable Lock.Name"
});

// When building a member value from a multi-level hierarchy, the uniqueName
// property of the member value will heavily depend on how the hierarchy was built
// Example: 5 level Address hierarchy = "Ontario.Toronto.D.Address"
// Example: 3 level Product hierarchy = "857.C.Products"
// 857 is the product ID but what's displayed is "Men's Bib-Shorts, L"

// Another option is to use the hierarchy service to get the desired member.
// Get the cellset from a visualization's metric set binding
var cellset = table1.metricSetBindings[0].dataResult.cellset;

// Get the hierarchy from the cellset based on its unique name
var hierarchy = cellset.getHierarchyByUniqueName("Product");

// Use the service since the cellset in the table only includes what's needed to render
// so it could be an incomplete set of data
var hierarchyService = this.getService("HierarchyService");

// This example gets all the members with "Bike" in the caption. The caption
// is the text displayed to users, unlike the unique name.
// getMembers can return all matching members from the specified hierarchy.
// analysisStructureId in this case is the data cube ID
var def = hierarchyService.getMembers(hierarchy.analysisStructureId, {
       // specify the unique name since there's likely more than 1 hierarchy in
       // the cube
       "hierarchyUniqueName": hierarchy.uniqueName,
       // specify the deepest hierarchy level unique name
       "levelUniqueName": hierarchy.levels[hierarchy.levels.length - 1].uniqueName,
```

```
        // filter by caption
        "memberFilter": new dundas.data.MemberQueryFilterRule({
                "field": dundas.data.MemberQueryField.CAPTION,
                "operator": dundas.QueryFilterOperator.CONTAINS,
                "value": "Bike"
        })
});

def.done(function (members) {
        // Loop through all the members and load their member values for the parameter
        members.forEach(function (member) {
                viewParameter.parameterValue.values.push(member.loadMemberValue());
        });

        // Set the last modified time so this parameter takes precedence over any others
        viewParameter.invalidateParameterValueLastModifiedTime();

        // Update all the connected adapters with the newly modified values
        // Includes data visualizations and filter controls
        // Pass the current view in case it's embedded in another
        viewParameter.refreshAllAdapters(null, view);
});
```

Note: You can also find the unique name of a hierarchy member through the UI if you want to refer to it directly rather than use the hierarchy service.

SingleMember

```
// In a view's (e.g., dashboard's) ready or load actions, use this rather than this.parentView:
var view = this.parentView;
var viewParameter = view.control.getViewParameterByName("viewParameter1");

// Remove all values and tokens from the parameter value
viewParameter.parameterValue.clearTokens();
viewParameter.parameterValue.clearValues();

// Time dimensions have a little bit more going on, so in this example
// we will be pulling the member value from an existing metric set,
// and just to make it interesting, we will use a data retrieval service
// to get the metric set data

// The metric set ID, which can be found in the edit dialog from the Data Analysis Panel
```

```
var metricSetId = "4f0ed0b9-3a4e-4517-a000-a21e593ca0c3";

// Get the DataRetrievalService
var dataRetrievalService = this.getService("DataRetrievalService");
var viewService = this.getService("ViewService");

// Create a request object using the metric set ID
var request = new dundas.data.Request({
       "objectId": metricSetId,
       "requestData": dundas.Utility.createGuid(),
});

// Call getData from the service and wait for it to finish using its deferred object
var def = dataRetrievalService.getData(request);

// Show a loading message if the response takes some time
viewService.showLoadingRestCall(def);

// When we have the data, run the following
def.done(function (dataResults) {
       // Here we load the first member from the first row of the data result
       var memberValue = dataResults[0].cellset.rows[0].members[0].loadMemberValue();

       viewParameter.parameterValue.value = memberValue;

       // Set the last modified time so this parameter takes precedence over any others
       viewParameter.invalidateParameterValueLastModifiedTime();

       // Update all the connected adapters with the newly modified values
       // Includes data visualizations and filter controls
       // Pass the current view in case it's embedded in another
       viewParameter.refreshAllAdapters(null, view);
});
```

RangeMember

```
// In a view's (e.g., dashboard's) ready or load actions, use this rather than this.parentView:
var view = this.parentView;
var viewParameter = view.control.getViewParameterByName("viewParameter1");

// Remove all values and tokens from the parameter value
viewParameter.parameterValue.clearTokens();
viewParameter.parameterValue.clearValues();
```

```
// Similar to the SingleMember example, we will pulling members from an existing
// metric set since the RangeMember uses an upper member and a lower member

// The metric set ID, which can be found in the edit dialog from the Data Analysis Panel
var metricSetId = "4f0ed0b9-3a4e-4517-a000-a21e593ca0c3";

// Get the DataRetrievalService
var dataRetrievalService = this.getService("DataRetrievalService");
var viewService = this.getService("ViewService");

// Create a request object using the metric set ID
var request = new dundas.data.Request({
       "objectId": metricSetId,
       "requestData": dundas.Utility.createGuid()
});

// Call getData from the service and wait for it to finish using its deferred object
var def = dataRetrievalService.getData(request);

// Show a loading message if the response takes some time
viewService.showLoadingRestCall(def);

// When we have the data, run the following
def.done(function (dataResult) {
       // Load two members from the data result
       var lowerMemberValue = dataResult[0].cellset.rows[0].members[0].loadMemberValue();
       var upperMemberValue = dataResult[0].cellset.rows[9].members[0].loadMemberValue();

       // Since this is a member range, we have an upper member and lower member
       viewParameter.parameterValue.lowerBoundaryValue = lowerMemberValue;
       viewParameter.parameterValue.upperBoundaryValue = upperMemberValue;

       // Set the last modified time so this parameter takes precedence over any others
       viewParameter.invalidateParameterValueLastModifiedTime();

       // Update all the connected adapters with the newly modified values
       // Includes data visualizations and filter controls
       // Pass the current view in case it's embedded in another
       viewParameter.refreshAllAdapters(null, view);
});
```

RangeDateTime

```
// In a view's (e.g., dashboard's) ready or load actions, use this rather than this.parentView:
var viewParameter = this.parentView.control.getViewParameterByName("viewParameter1");

// Remove all values and tokens from the parameter value
viewParameter.parameterValue.clearTokens();
viewParameter.parameterValue.clearValues();

// A RangeDateTime parameter value will have an upper and lower value set
// similar to that of the RangeMember, but instead of member values inside
// the upper and lower, there are just dates

// Create a date for the current time
var currentDateTime = new Date(Date.now());

// Create a date time object for this time one year ago
var oneYearAgo = new Date(Date.now());
oneYearAgo.addYears(-1);

// Offset the dates from the client's local time zone to UTC to be sent to the server
currentDateTime = dundas.Utility.getUTCOffsetDateTime(currentDateTime);
oneYearAgo = dundas.Utility.getUTCOffsetDateTime(oneYearAgo);

// Assign the dates to their respective upper and lower values
viewParameter.parameterValue.lowerBoundaryValue = oneYearAgo;
viewParameter.parameterValue.upperBoundaryValue = currentDateTime;

// Set the last modified time so this parameter takes precedence over any others
viewParameter.invalidateParameterValueLastModifiedTime();

// Update all the connected adapters with the newly modified values
// Includes data visualizations and filter controls
// Pass the current view in case it's embedded in another
viewParameter.refreshAllAdapters(null, this.parentView);
```

Tokens

```
// In a view's (e.g., dashboard's) ready or load actions, use this rather than this.parentView:
var view = this.parentView;
var viewParameter = view.control.getViewParameterByName("viewParameter1");
```

```
// Remove all values and tokens from the parameter value.
viewParameter.parameterValue.clearTokens();
viewParameter.parameterValue.clearValues();

// We are going to set the lower token to Open Range Boundary
// and the upper range to End of Current Year (-1).
// If you know the ID of the token you want to use, you don't need to
// use the service. You can simply pass a new object into the token
// constructor with the desired properties.
// The dundas.constants contains some common token IDs:
viewParameter.parameterValue.lowerBoundaryToken = new dundas.data.ParameterToken({
       "id": dundas.constants.OPEN_RANGE_BOUNDARY_TOKEN_DEFINITION_ID,
       "caption": "Open Range Boundary"
});

// If we don't know the token ID, we can use the token service
var tokenService = view.getService("TokenService");

// This example uses time hierarchy tokens
var timeHierarchyDataType = dundas.data.CompatibleDataTypes.TIME_HIERARCHY;

// This example will use tokens that define the end of a range
var rangeEndTokenTypes = dundas.data.TokenDisplayTypes.DISPLAY_RANGE_END;

// Use the deferred object to wait for the token service to get the tokens from the server
var def = tokenService.getParameterTokens(timeHierarchyDataType, rangeEndTokenTypes);

// When request is done, use the result to get the desired token(s)
def.done(function (tokens) {
       // Store the token to modify later.
       var endCurrentYearToken = tokens.filter(function (token) {
              return token.caption == "End of current year";
       })[0];

       // Change the offset of the token
       endCurrentYearToken.offset = -1;

       // Set the upper token to the end of year token
       viewParameter.parameterValue.upperBoundaryToken = endCurrentYearToken;

       // Set the last modified time so this parameter takes precedence over any others
       viewParameter.invalidateParameterValueLastModifiedTime();

       // Update all the connected adapters with the newly modified values
```

```
        // Includes data visualizations and filter controls
        // Pass the current view in case it's embedded in another
        viewParameter.refreshAllAdapters(null, view);
});
```

Note: The ID of any token can found from the Tokens screen in Administration by selecting one and clicking Details in the toolbar, and you can use this directly in your script instead of calling the token service.

Reset Parameter Values to "All" or "Default"

All the filters on the dashboard can be reset to "All" or "Default" values via a script. The script works for all types of filters.

```
var view = this.parentView; // In a view's actions, use 'this' rather than this.parentView
view.control.viewParameters.forEach(function(viewParameter, index) {
        if (index != 0) { // skip the brush view parameter
                viewParameter.parameterValue.clearTokens();
                viewParameter.parameterValue.clearValues();
                viewParameter.parameterValue.token = new dundas.data.ParameterToken({
                        // Use dundas.constants.ALL_TOKEN_DEFINITION_ID for "All" token
                        "id": dundas.constants.DEFAULT_TOKEN_DEFINITION_ID,
                        // Use "All" caption for All token
                        "caption": "Default"
                });
                viewParameter.invalidateParameterValueLastModifiedTime();
                // Pass the current view in case it's embedded in another
                viewParameter.refreshAllAdapters(null, view);
        }
});
```

For more information, see:

- Script Library: Filter and Parameter
- Set Filter Value By Script
- Set Token Value By Script
- JavaScript API

- Writing Scripts Using Browser Developer Tools
- Adding Filters

Set Filter Value By Script

This applies to: *Managed Dashboards, Managed Reports*

This article walks through how to set a parameter value using script. In the example shown below, a filter is set to a hierarchy value on a click of a button.

While script is not required to accomplish an interaction like this, the purpose of this article is to demonstrate how you can perform this action using the API as part of your own custom functionality.

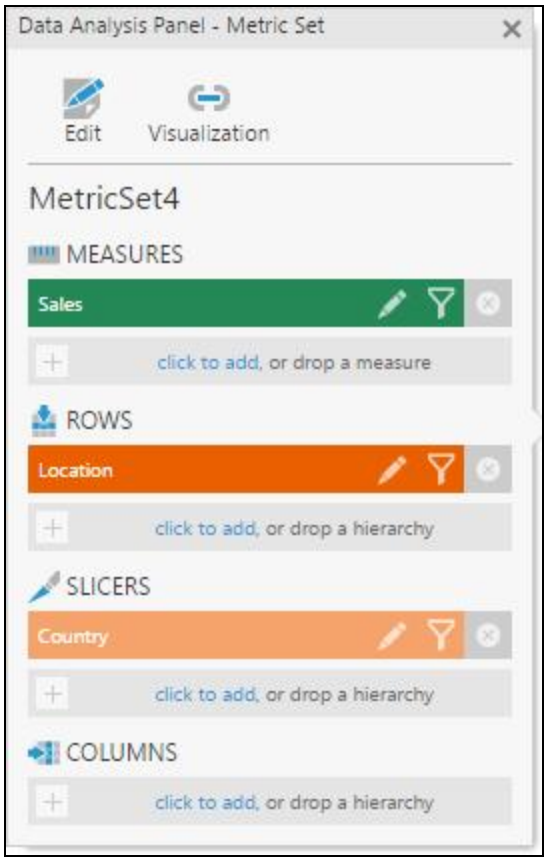
Script Library samples: Filter and Parameter

Set up the dashboard

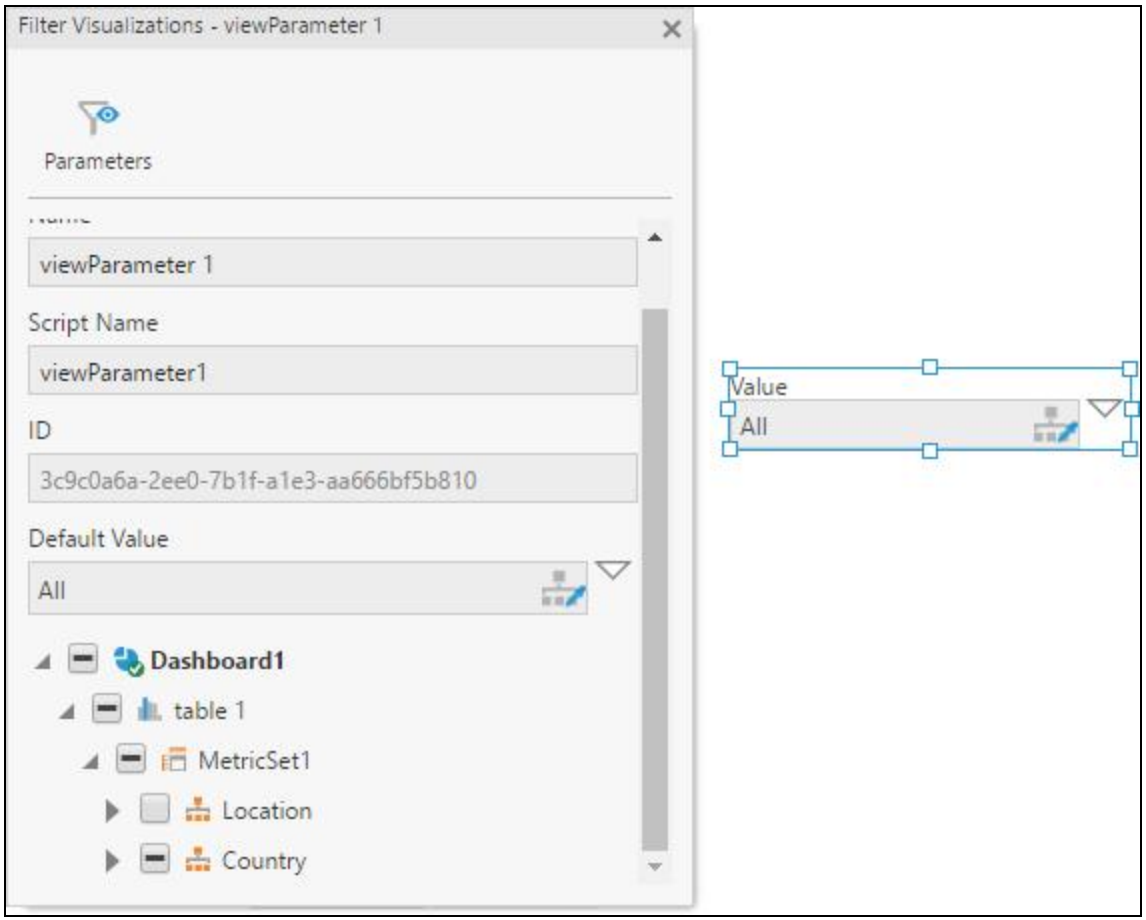
For this example, a relational table (non-OLAP) is used.

	Continent	Country	Location	Sales	Sales Target
1	North America	USA	California	2500	3000
2	North America	USA	Texas	2000	3000
3	North America	Canada	Ontario	3000	4000
4	North America	Canada	Quebec	4000	4000
5	Asia	China	Beijing	5000	5000
6	Asia	China	Shanghai	2000	5000
7	Asia	Japan	Tokyo	8100	5000
8	Asia	Japan	Narita	15000	8000

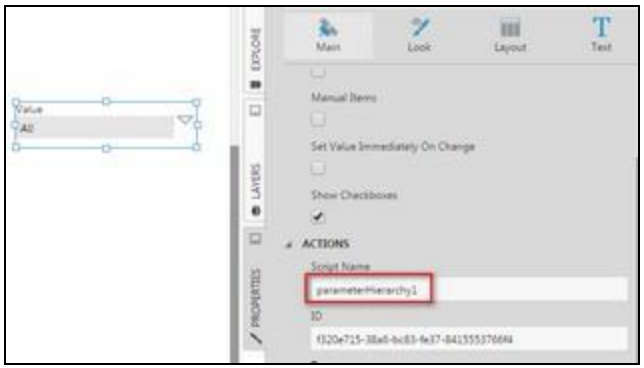
Add a data visualization to the dashboard such as a table or chart. Drop a measure under **Measures** and a hierarchy to **Rows** in the *Data Analysis Panel*. In this example, the *Country* column is also added to **Slicers**.



Add a hierarchy filter and connect it to your column. Take note of the View Parameter **Script Name**, which in this case is *viewParameter1*. For your first filter created on the dashboard, this is the default name, but you can rename it if you choose.



Take note also of the filter control name, which is *parameterHierarchy1*. Like the View Parameter, this is the default name for the first filter added, but you can rename this if you choose.

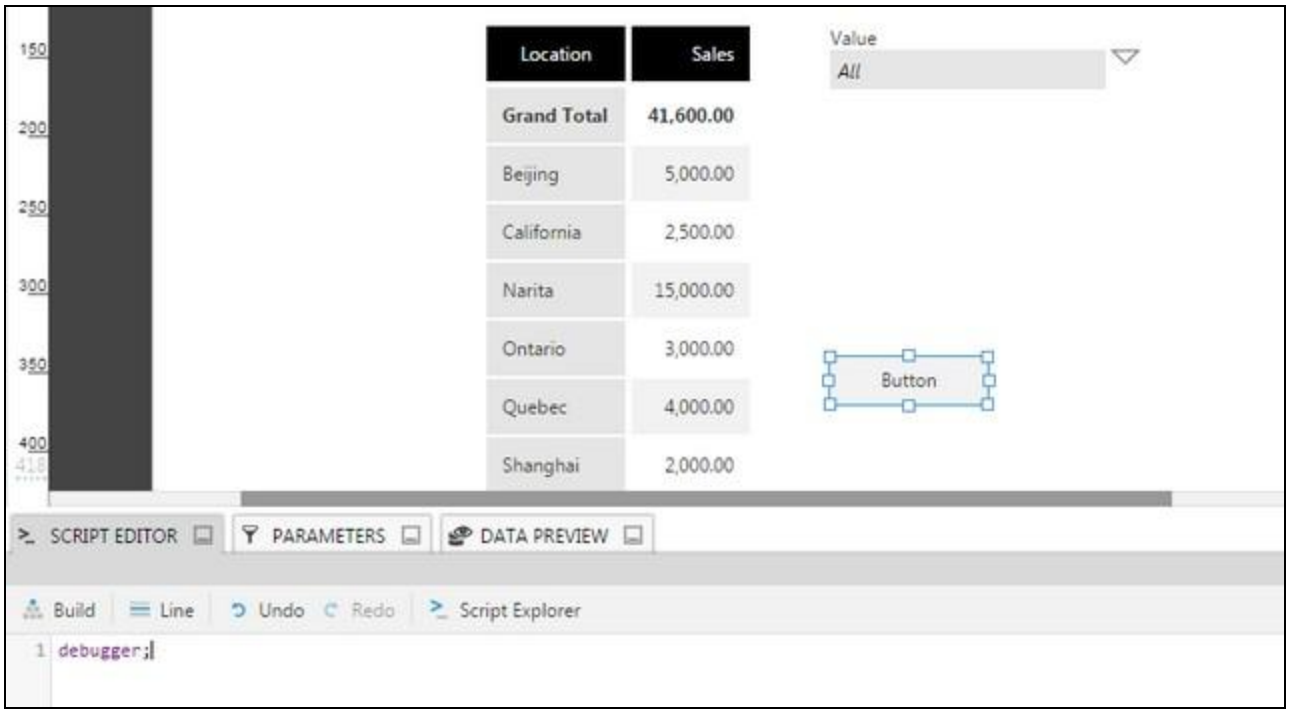


Determine the Required Script

The script to use depends on the type of value you want to set and how you determine its value. You can find details and examples of various parameter types here: Modify a Filter / View Parameter Using Scripting.

Below, we will walk through one way to find a hierarchy member's information interactively and use it to set a CollectionMemberValue type parameter.

Add a button component and add the script debugger; in the button's **Click** script action. See Writing scripts using browser developer tools for more details on this JavaScript statement.

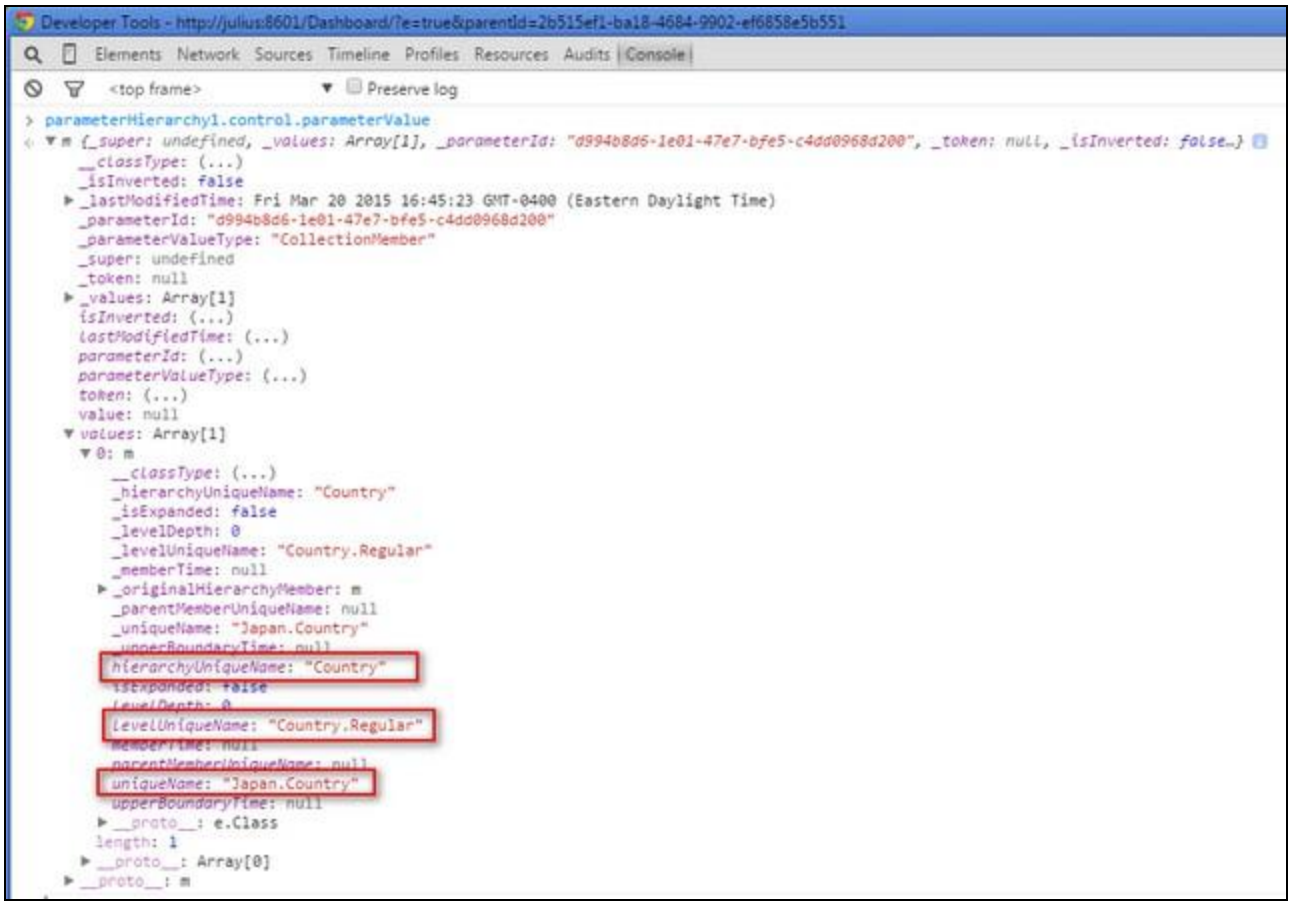


View the dashboard and select the filter value (Japan in this example).

Click the button to fire the **Click** script action.

In the browser's JavaScript console, type the script below while the debugger is paused within your click script action (adjust the filter control name if necessary):

```
parameterHierarchy1.control.parameterValue;
```



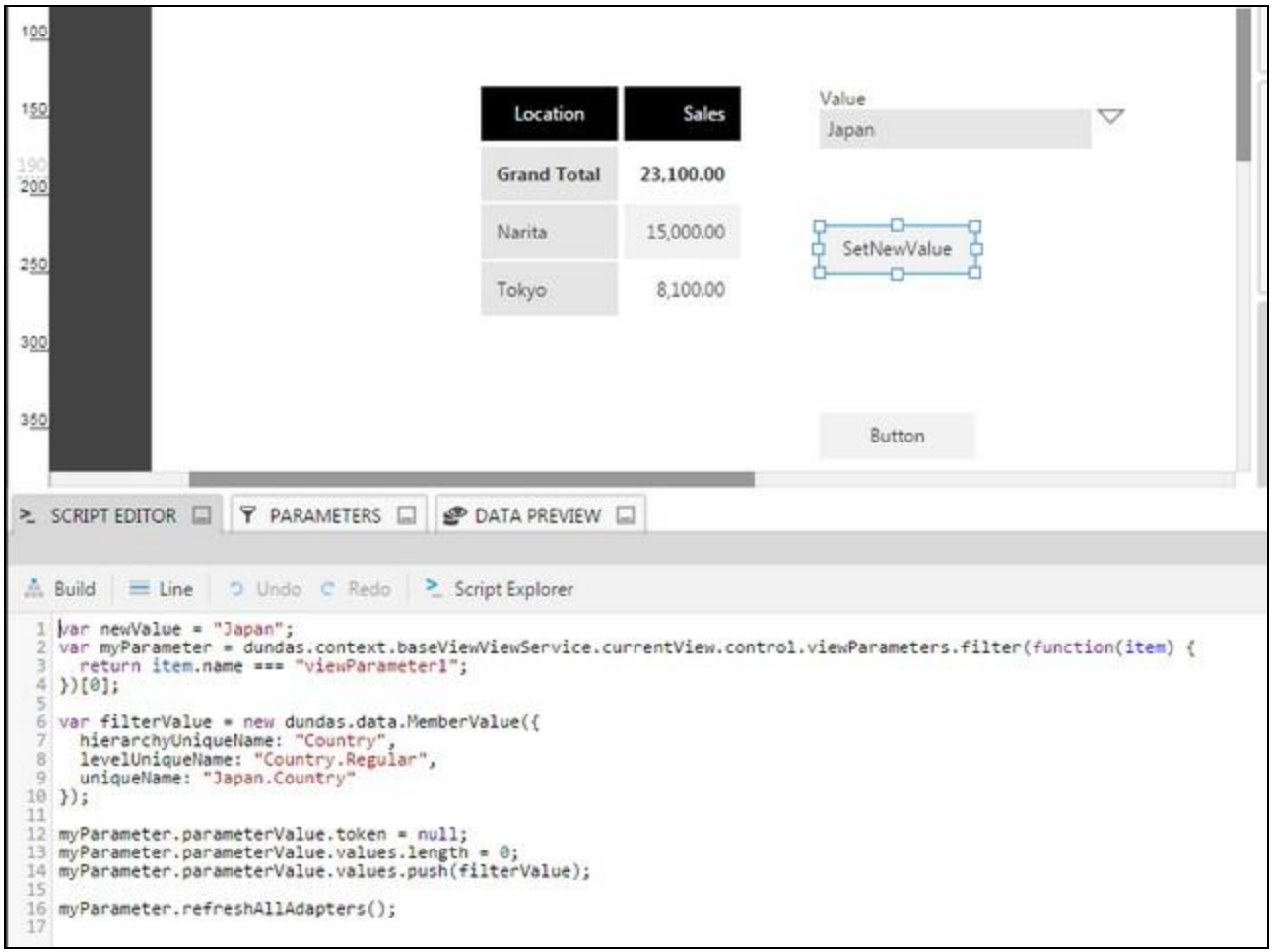
Inspect the parameter's structure and the values, particularly the following:

- hierarchyUniqueName = "Country"
- LevelUniqueName = "Country.Regular"
- uniqueName = "Japan.Country"

Add the Script

Note: You can add this script to any event, such as a button's **Click**, a dashboard's **Load** or **Ready**, a parameter's value changed actions, etc. If you use the script in the actions of a dashboard or other view, you should replace *this.parentView* with simply this.

```
var viewParameter = this.parentView.control.getViewParameterByName("viewParameter1");
var filterValue = new dundas.data.MemberValue({
       hierarchyUniqueName: "Country",
       levelUniqueName: "Country.Regular",
       uniqueName: "Japan.Country"
});

viewParameter.parameterValue.token = null;
viewParameter.parameterValue.values.length = 0;
viewParameter.parameterValue.values.push(filterValue);

viewParameter.invalidateParameterValueLastModifiedTime();
viewParameter.refreshAllAdapters(null, this.parentView);
// (Pass the current view in case it's embedded inside another)
```



Test the Script

1. View the dashboard.
2. Select any filter value other than the one you're setting through script.
3. Click the Button control to fire the script.

Result: The new filter value should be set:

For more information, see:

- JavaScript API
- Script Library: Filter and Parameter
- Use a Button to Apply a Filter Value
- Set Token Value By Script
- Modify a Filter / View Parameter Using Scripting
- Using the Script Editor
- Writing Scripts Using Browser Developer Tools

Set Token Value By Script

This applies to: *Managed Dashboards, Managed Reports*

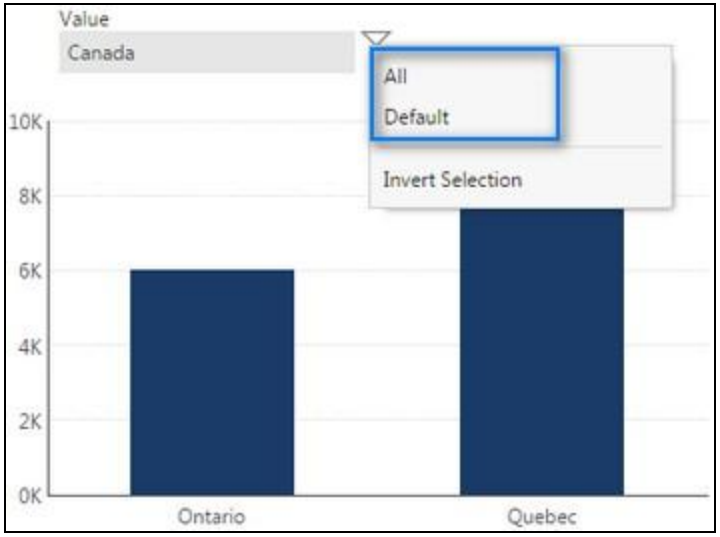
This article walks through how to set a token value using script. In the example below, the filter value is set to All when a button is clicked. See Set Filter Value By Script for the steps for setting up the dashboard and filter used in this example.

While script is not required to accomplish an interaction like this, the purpose of this article is to demonstrate how you can perform this action using the API as part of your own custom functionality.

Script Library sample: Set range number parameter token

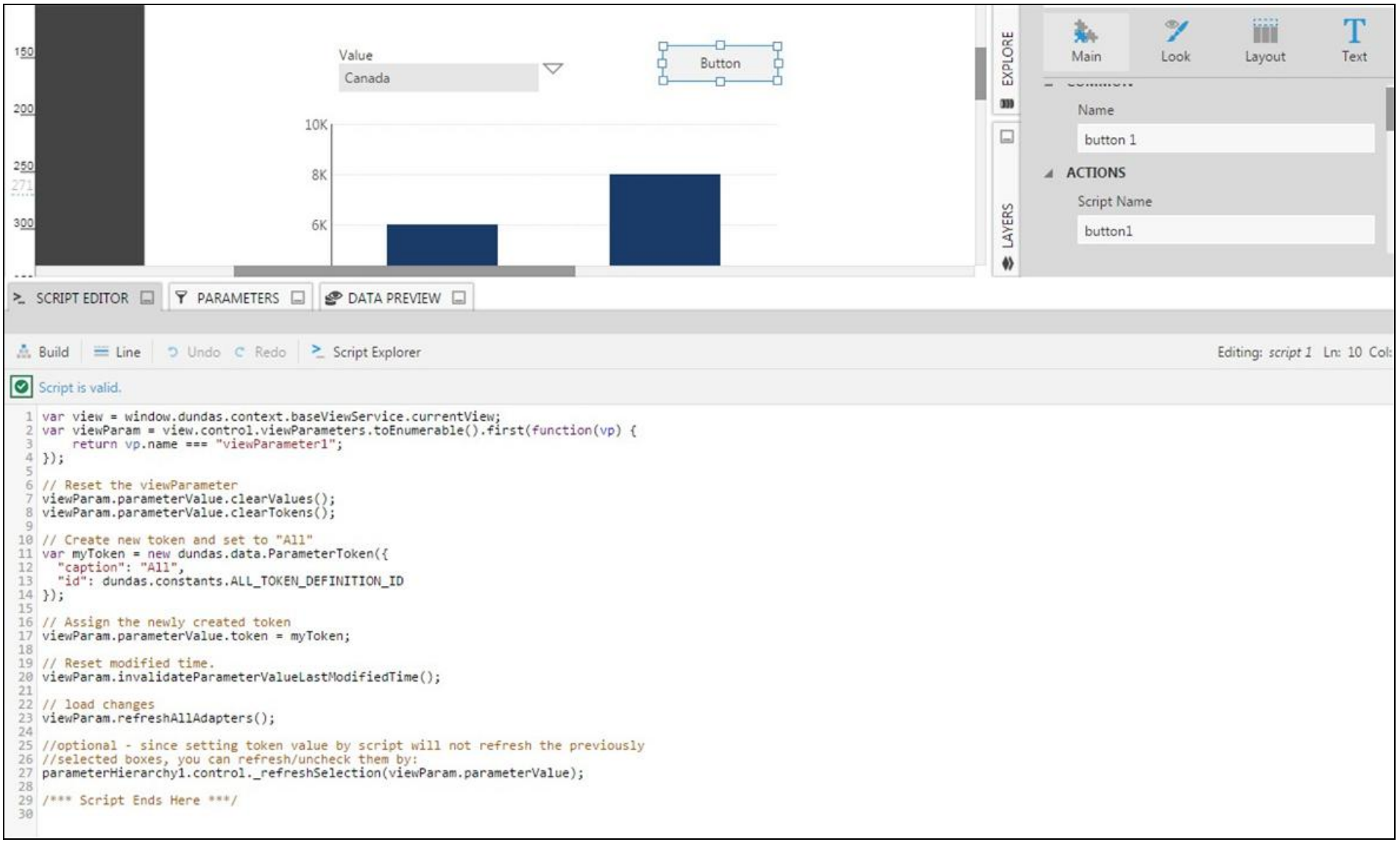
Check applicable token values

You can view the dashboard and check the applicable token values. In this case, **All** and **Default**.



Add the Script

Add the script to the button's **Click** actions.




```
var viewParameter = this.parentView.control.getViewParameterByName("viewParameter1");
```

```
// Reset the view parameter
viewParameter.parameterValue.clearValues();
viewParameter.parameterValue.clearTokens();

// Create new token and set to "All"
var myToken = new dundas.data.ParameterToken({
       "caption": "All",
       "id": dundas.constants.ALL_TOKEN_DEFINITION_ID
});

// Assign the newly created token
viewParameter.parameterValue.token = myToken;

// Reset modified time to take precedence
viewParameter.invalidateParameterValueLastModifiedTime();

// Load changes
viewParameter.refreshAllAdapters(null, this.parentView);
// (Pass the current view in case it's embedded inside another)
```

Here is a list of tokens with predefined IDs you can use as done above. Note that not all are applicable for the member filter used in this sample. Any token can be used once you find its ID - see Modify a Filter / View Parameter Using Scripting for more details and examples.

- `ALL_TOKEN_DEFINITION_ID`: "All"
- `DEFAULT_TOKEN_DEFINITION_ID`: "Default"
- `NULL_TOKEN_DEFINITION_ID`: "Null" (only if your filter allows null)
- `NO_SELECTION_TOKEN_DEFINITION_ID`: "No Selection"
- `OPEN_RANGE_BOUNDARY_TOKEN_DEFINITION_ID`: "Open Range" (normally used for date filters)

Tip

The ID of any token can found from the Tokens screen in Administration by selecting one and clicking Details in the toolbar.

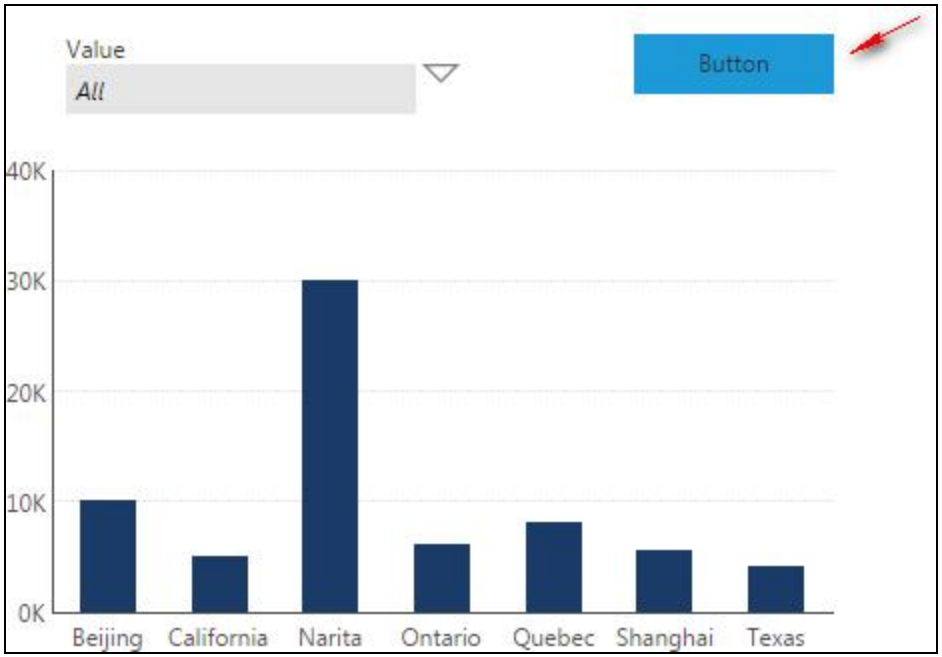
Test the Script

Choose **Sandbox View** in the toolbar to test the dashboard without saving changes made while viewing.

Select any filter specific value.

Click the button.

Result: The new filter value should be set to "All".



For more information, see:

- JavaScript API
- Script Library: Set range number parameter token
- Use a Button to Apply a Filter Value
- Set Filter Value By Script
- Modify a Filter / View Parameter Using Scripting

- Using the Script Editor
- Writing Scripts Using Browser Developer Tools

Set Up a Navigational Image Button

This applies to: *Managed Dashboards, Managed Reports*

This article shows you how to configure a button component to display an image and perform a navigate action when clicked.

Create Your Dashboards and Button

Create a target dashboard

First, create a new dashboard from the main menu and have it display a simple chart.

This will be the dashboard (**Dashboard1**) to navigate to.

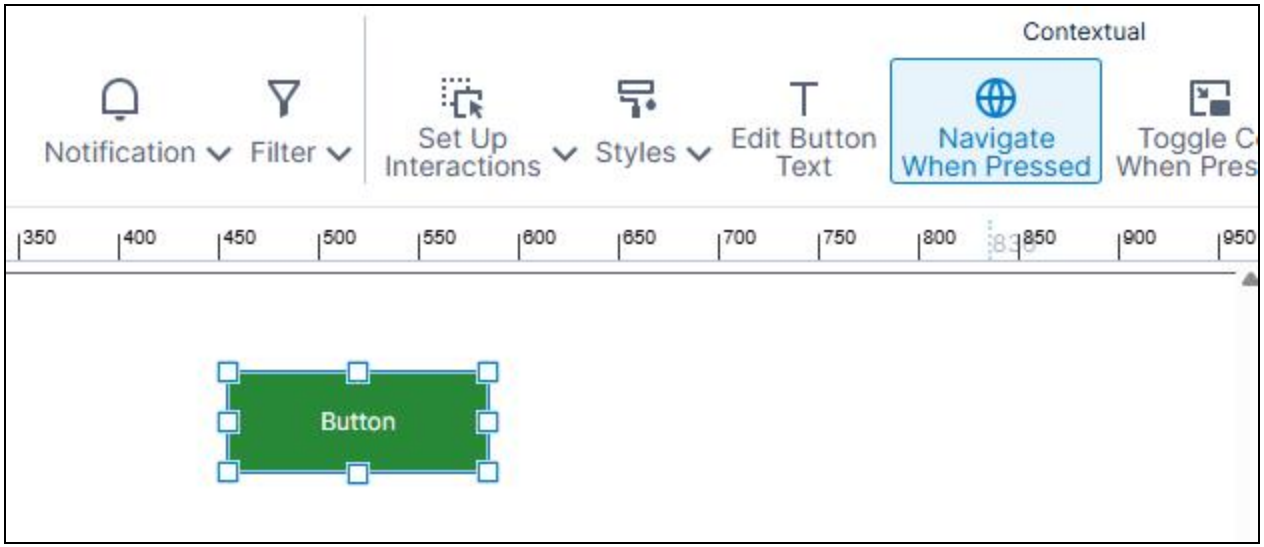
Create a source dashboard

Create a second dashboard (**Dashboard2**) which will contain the navigational image button.

Go to the **toolbar**, click **Components**, and then select **Button** to add a button component to the canvas.

Set up the navigation

With the button selected on the canvas, go to the toolbar and click **Navigate When Pressed**.



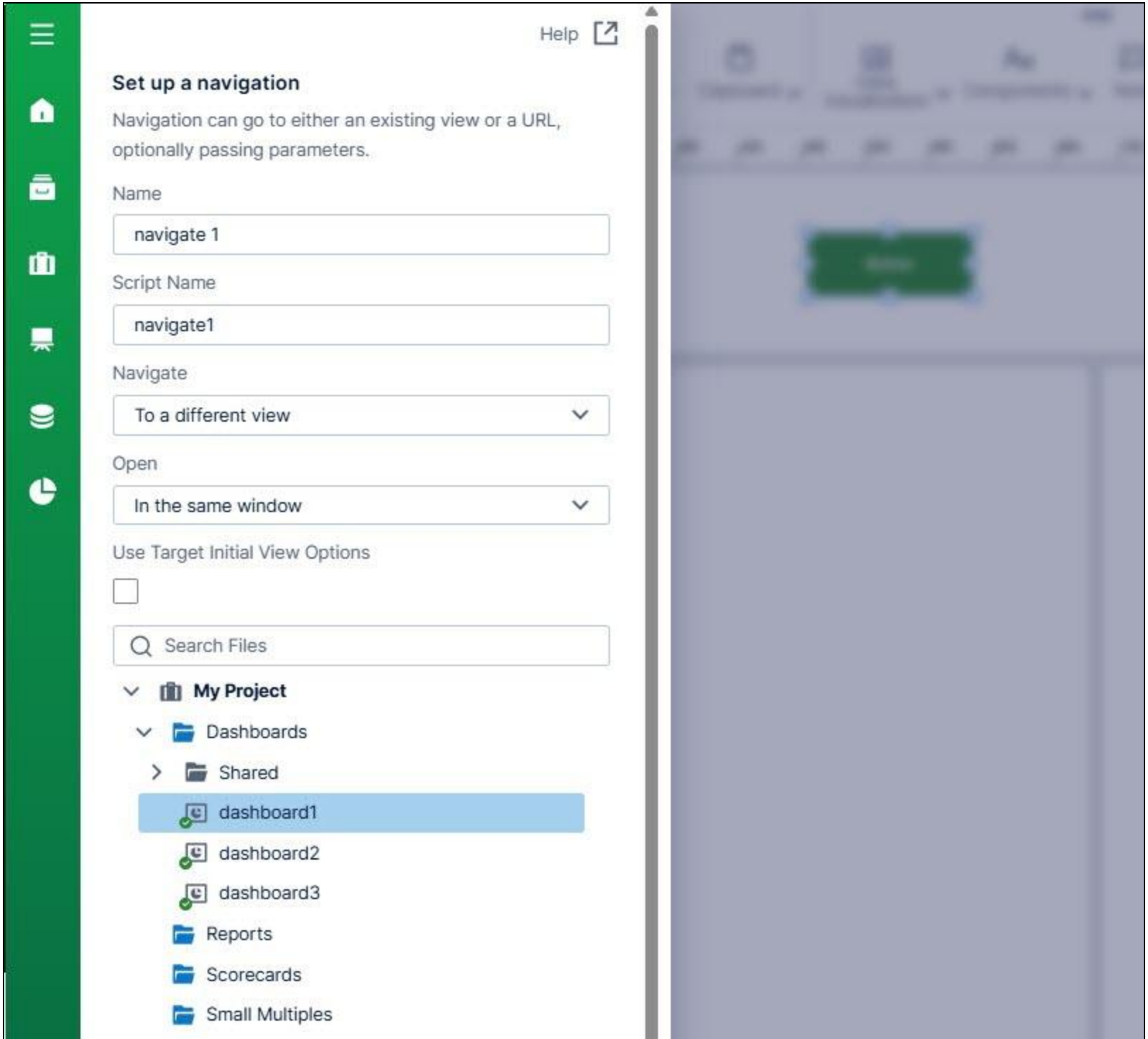
In the **Set up a Navigation** dialog, choose a **Navigate** option from the following:

- **To a different view** - This is the default. Use this option for this example.
- **To a URL**
- **Back** - This is for navigating back. You use this on a target dashboard to go back to the source dashboard.
- **To this same view**

Next, choose the **Open** option:

- **In the same window** - This is the default. Use this option for this example.
- **In a new window**

Finally, select the target dashboard to navigate to.


Help
Set up a navigation
Navigation can go to either an existing view or a URL, optionally passing parameters.
Name
navigate 1
Script Name
navigate1
Navigate
To a different view
Open
In the same window
Use Target Initial View Options
Search Files
My Project
Dashboards
Shared
dashboard1
dashboard2
dashboard3
Reports
Scorecards
Small Multiples

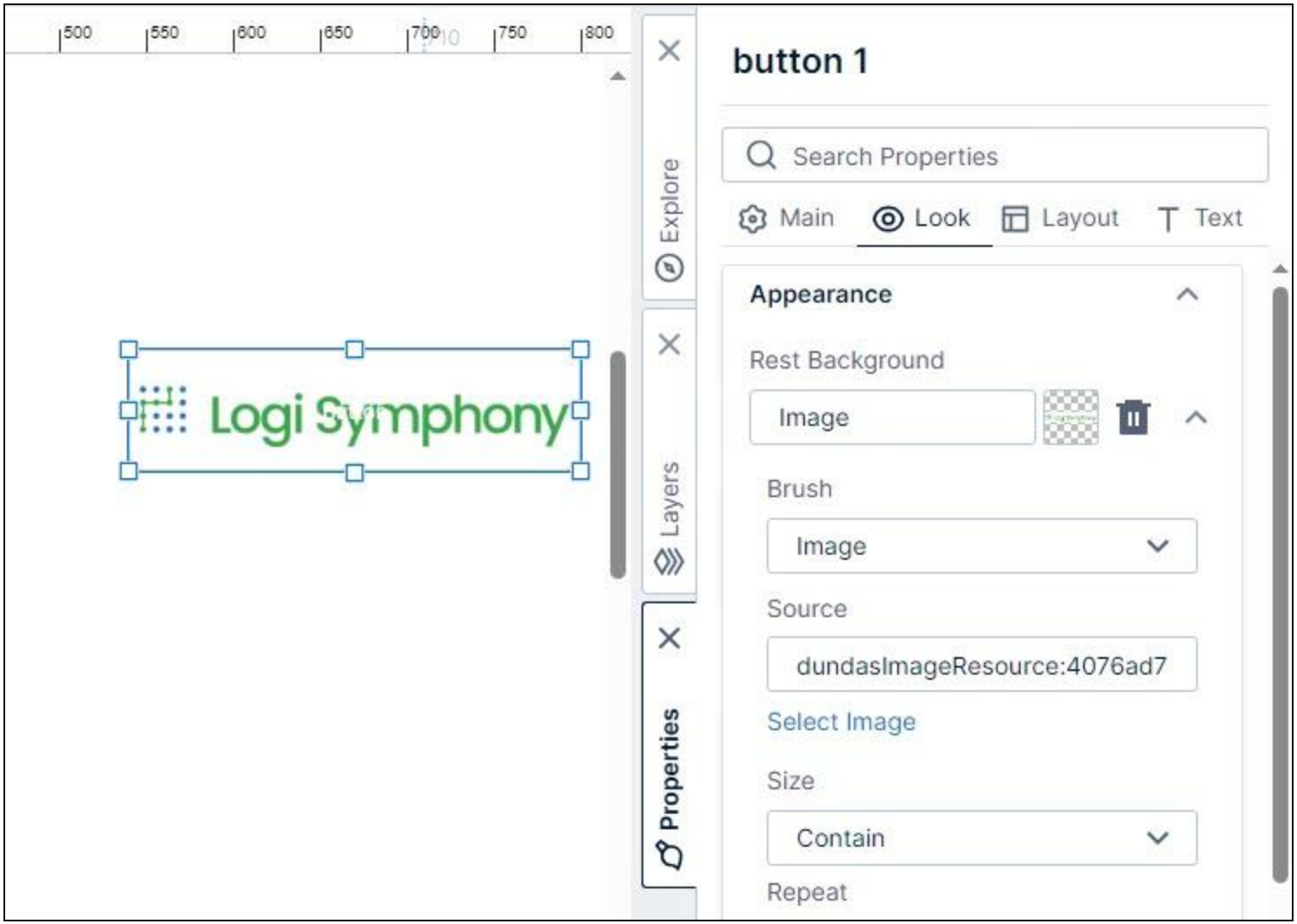
Add an Image to the Button

Add an image resource to your project by dragging a PNG, GIF or JPEG image file from Windows Explorer or Finder to the **Explore** window.

The image will appear under the **Images** folder in **Explore**.

Select the button on the canvas and open the **Properties** window. Select the **Look** tab.

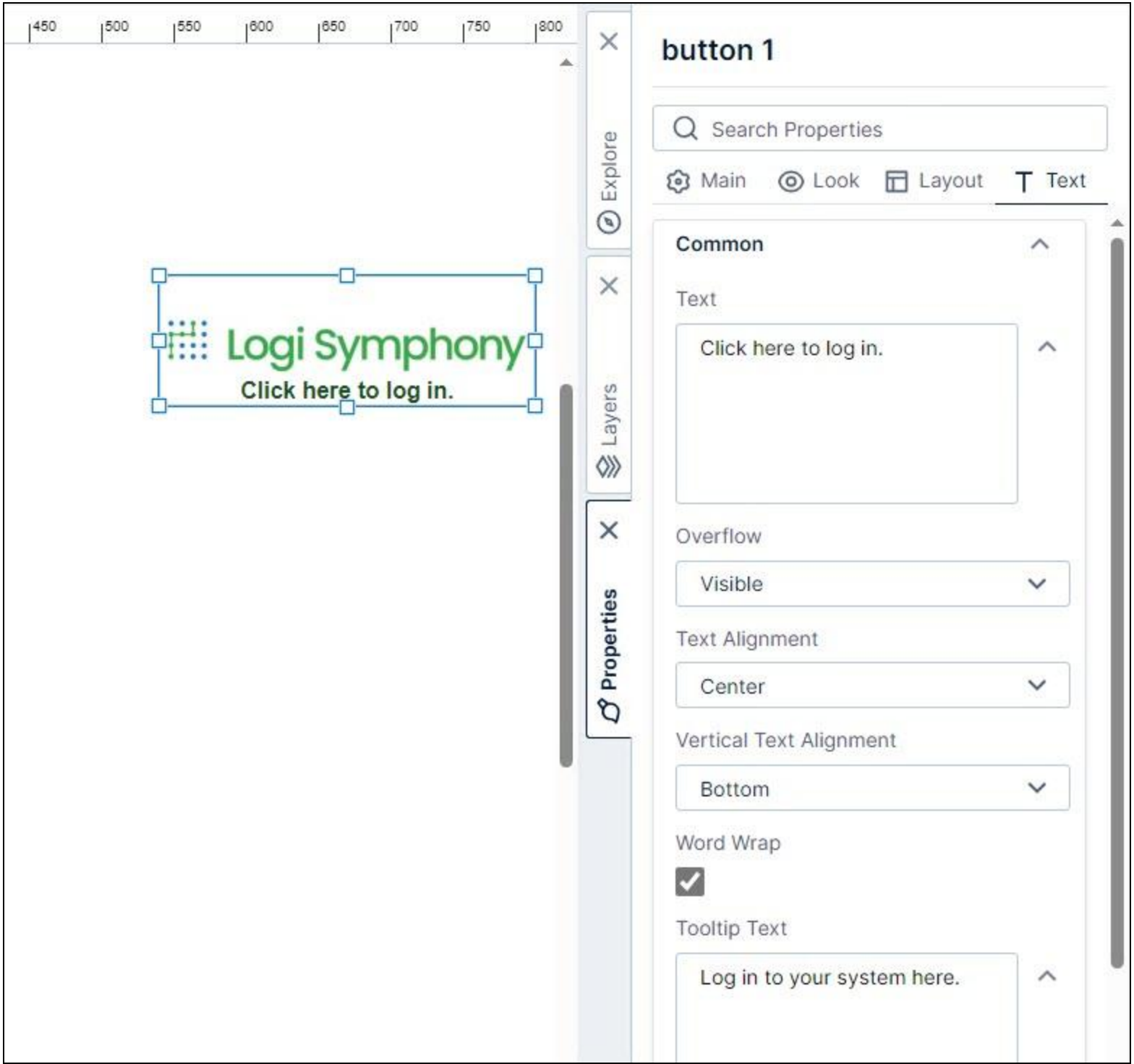
Expand the `Rest Background` property and set the **Brush** type to Image. Click the Select **Image** link and choose the PNG image from the Images folder.



Repeat the above steps for the **Hover** and **Down** states of the button using the same image or different images.

Adjust the Button Text

If you would still like to display text in the button go to its **Text** properties. There are a number of properties for setting the text, overflow, text alignment, and vertical text alignment which you can use to position the text below the image, for example.


450
500
550
600
650
700
750
800
Logi Symphony
Click here to log in.
Explore
Layers
Properties
button 1
Search Properties
Main
Look
Layout
Text
Common
Text
Click here to log in.
Overflow
Visible
Text Alignment
Center
Vertical Text Alignment
Bottom
Word Wrap
Tooltip Text
Log in to your system here.

Test the Navigation

Switch to **View** mode and click the button to test the navigation.

For more information, see:

- Using Image Brushes
- Set Up a Navigate Interaction

Set Up a Hover Over Interaction

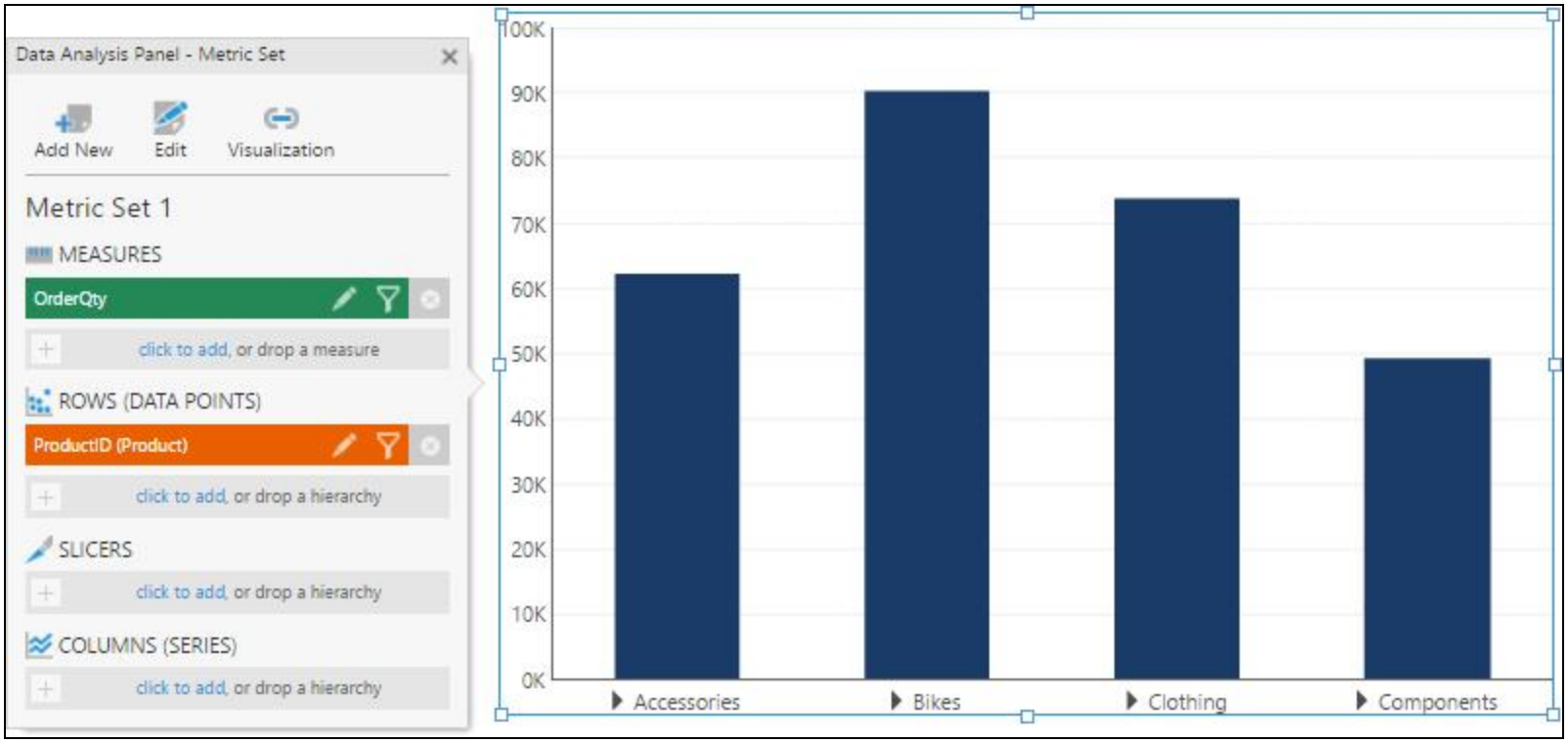
This applies to: *Managed Dashboards, Managed Reports*

This article shows you how to set up a hover over interaction so that a second dashboard is displayed in a popup and filtered accordingly when you hover over a data point on the first dashboard.

Related video: Interactions

Create the First Dashboard

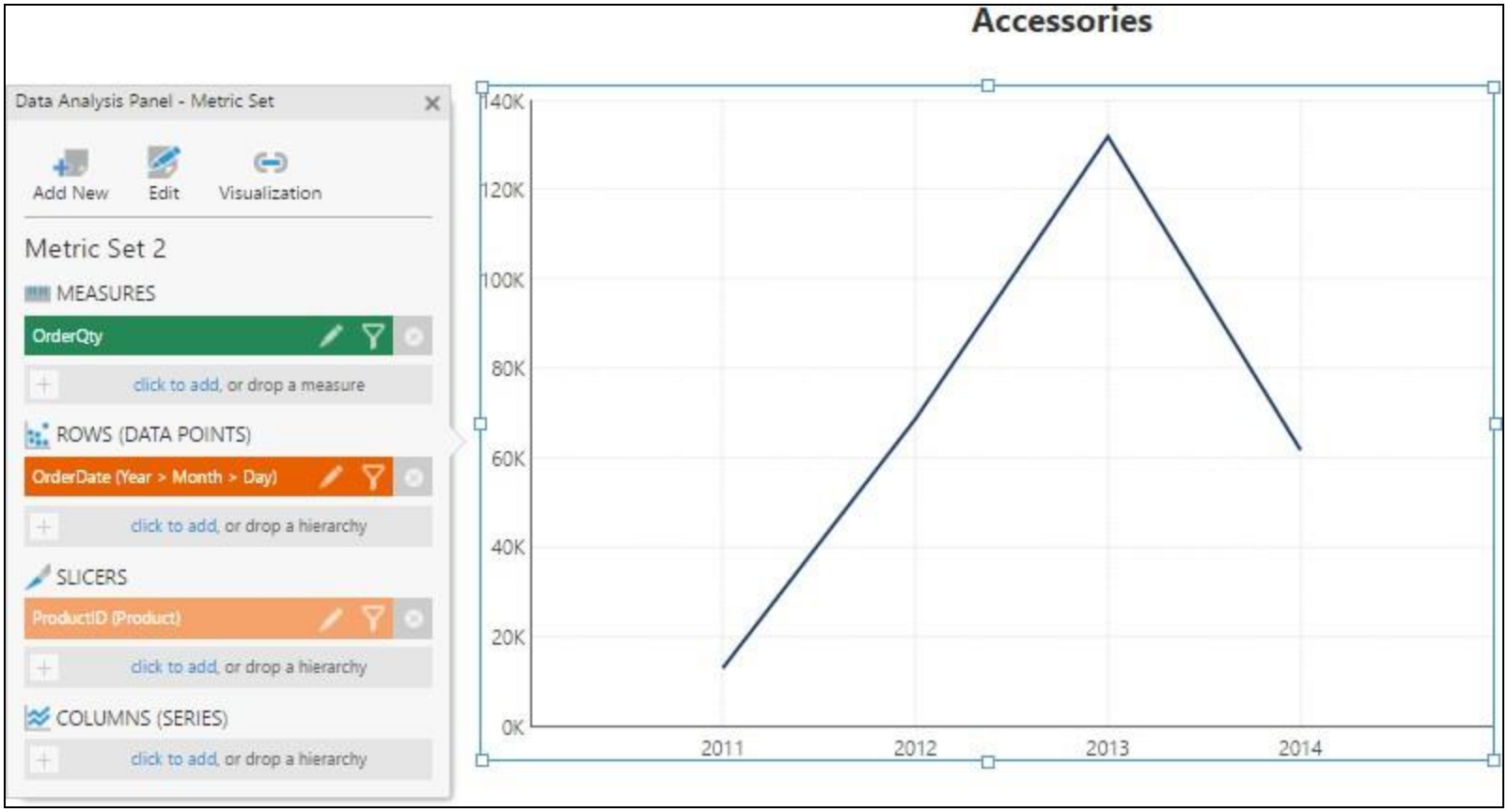
Create a metric set and add it to a dashboard in order to show *OrderQty by Product*.



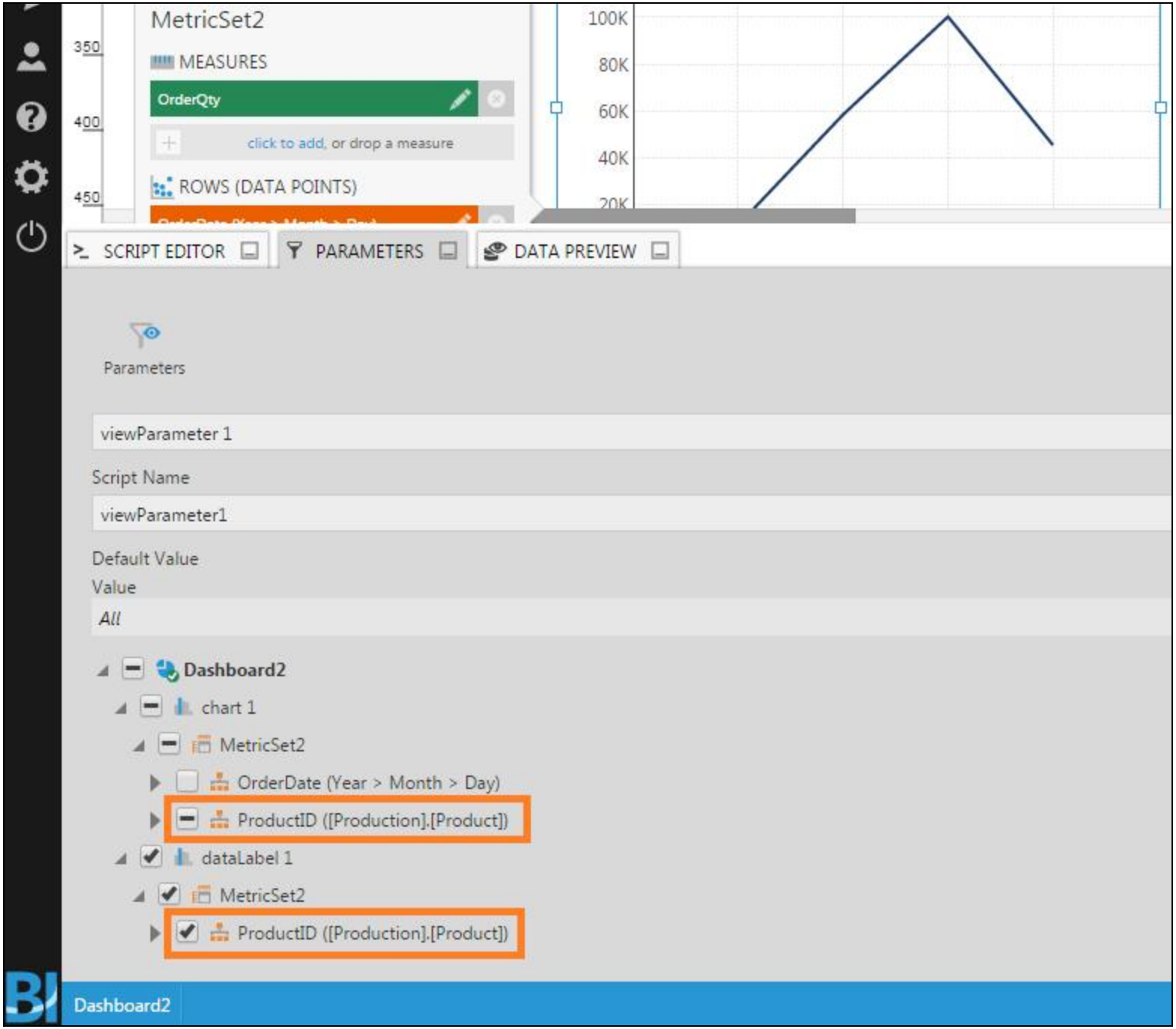
Create a Second Dashboard

Create another metric set and add it to a second dashboard in order to show *OrderQty by Date*. This dashboard will be displayed in a hover over / popup. The chart in this dashboard is further filtered by *Product* by adding the corresponding hierarchy under **Slicers**.

Besides the chart, also add a data label control and drag the second metric set's Product hierarchy onto the data label. This data label will be used to display the slicers (filter) value.

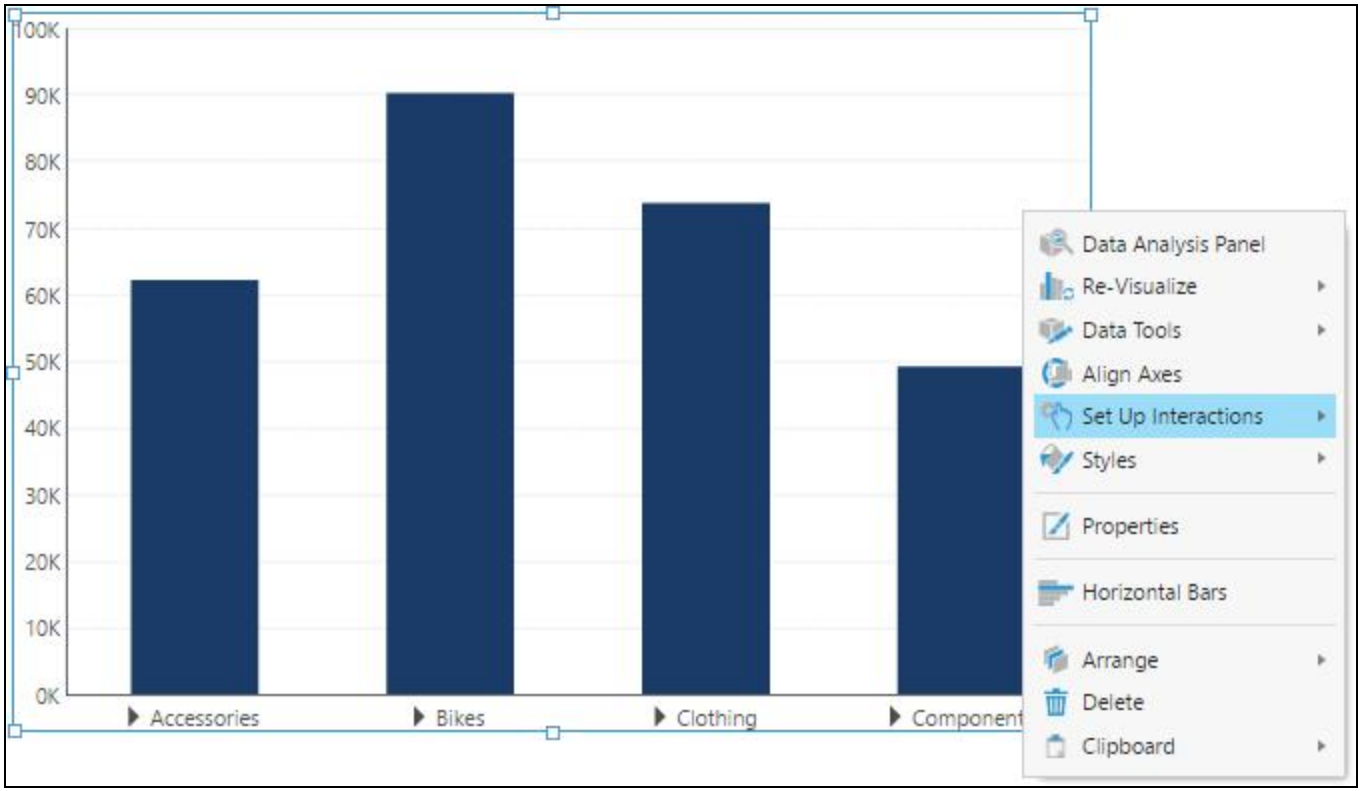


Next, expand the **Parameters** window which is docked along the bottom of the screen. Click **Add New** to add a new view parameter and then connect it to the Product hierarchy for both the chart and the data label.

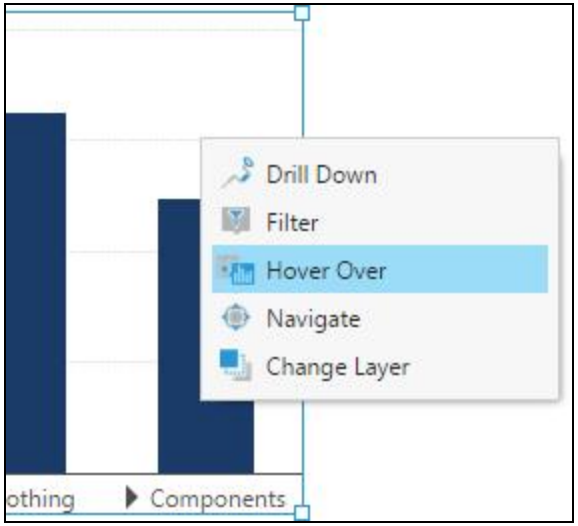

MetricSet2
MEASURES
OrderQty
click to add, or drop a measure
ROWS (DATA POINTS)
350
400
450
100K
80K
60K
40K
20K
SCRIPT EDITOR
PARAMETERS
DATA PREVIEW
Parameters
viewParameter 1
Script Name
viewParameter1
Default Value
Value
All
Dashboard2
chart 1
MetricSet2
OrderDate (Year > Month > Day)
ProductID ([Production].[Product])
dataLabel 1
MetricSet2
ProductID ([Production].[Product])
Dashboard2

Set Up the Hover Over Interaction

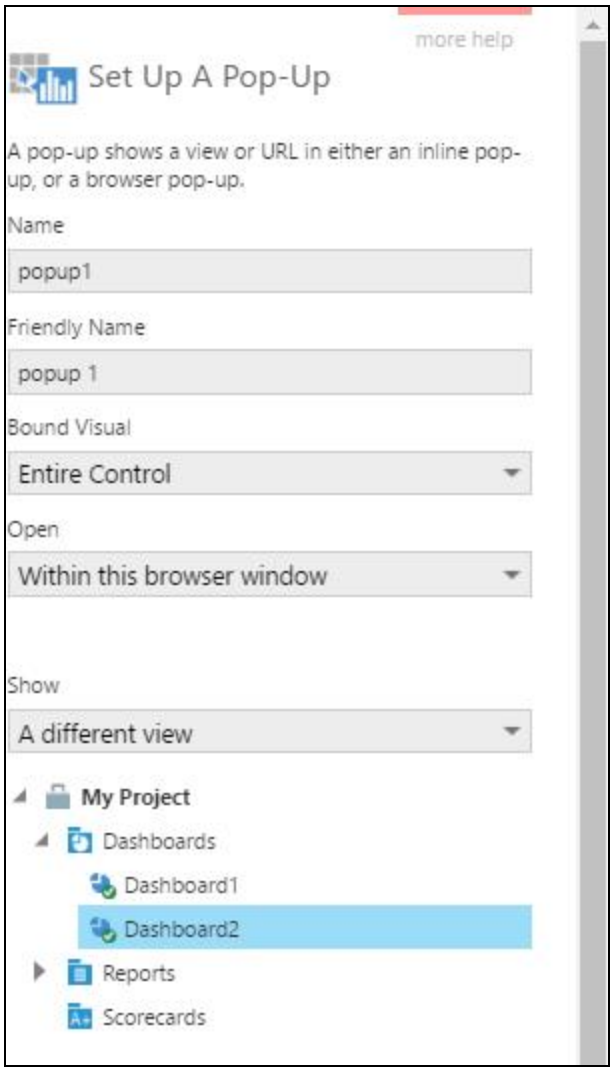
Edit the first dashboard again. Right click over the bar chart and click **Set Up Interactions** from the menu.



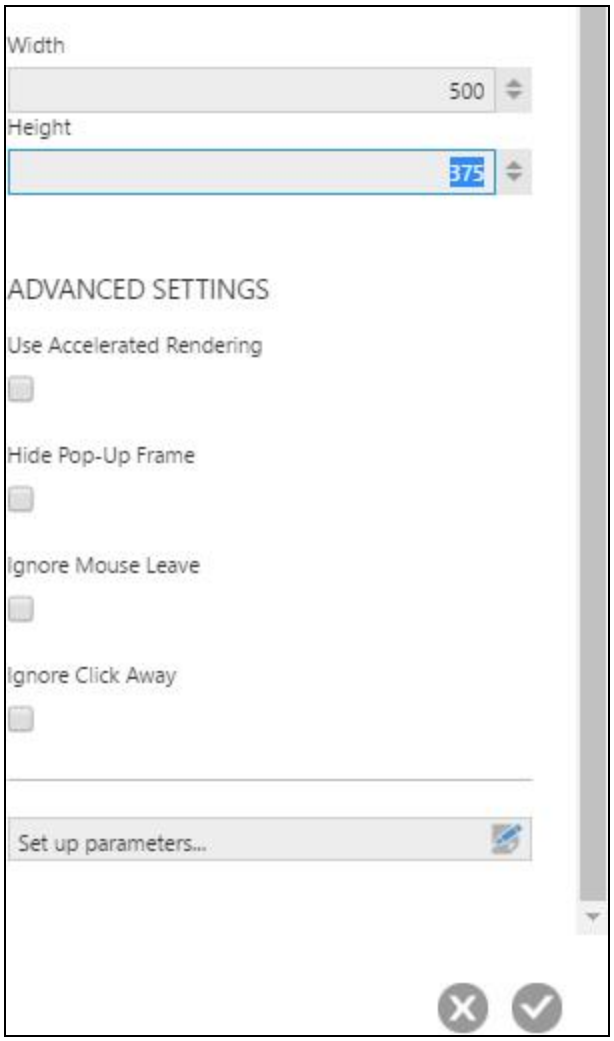
Click **Hover Over**.



In the *Set up* a pop-up dialog, select *Dashboard 2* as the different view to be displayed on hover-over.



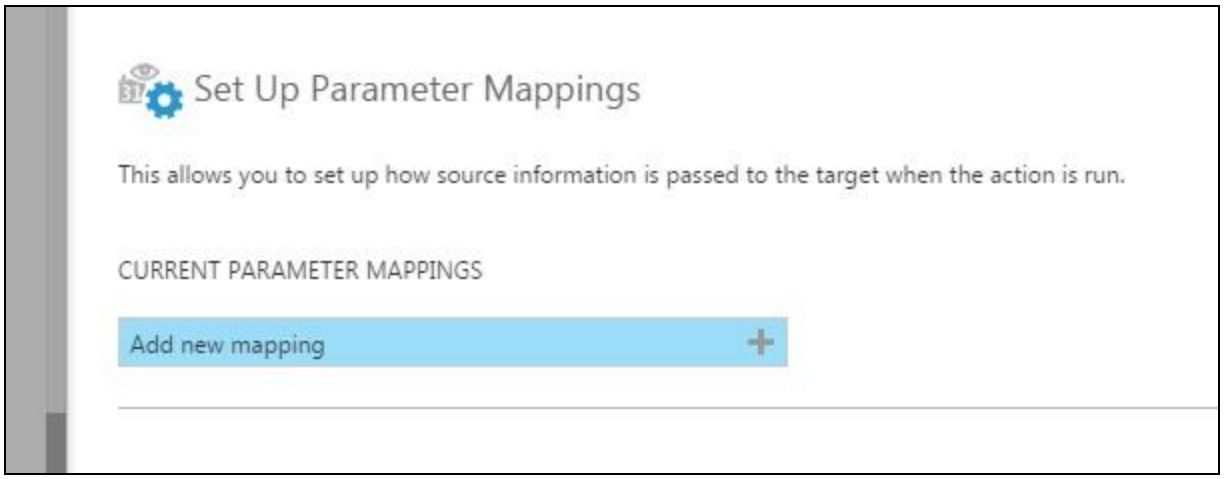
Scroll further down and set the **Width** (of the popup) to *500* and the **Height** to *375*.



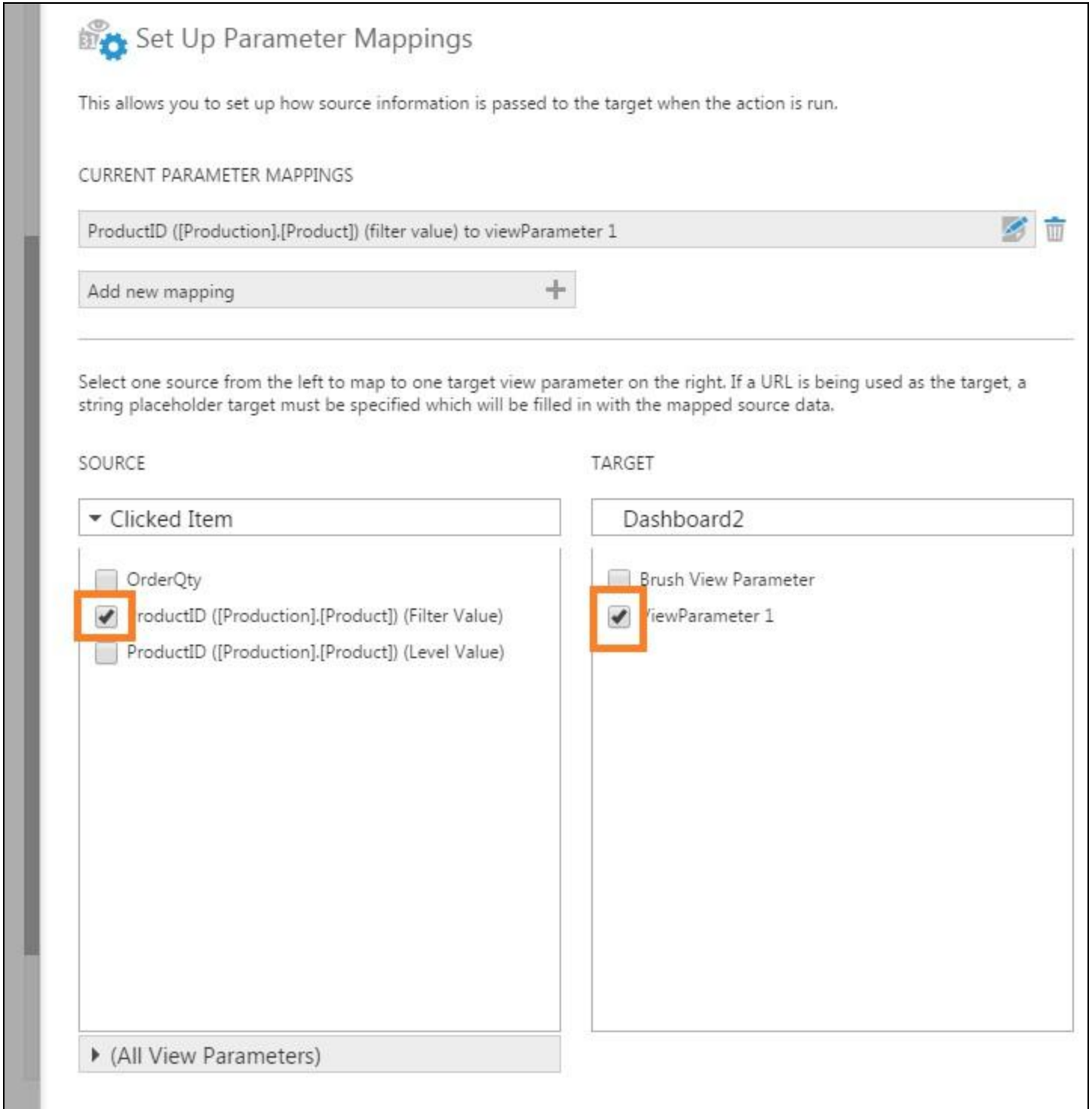
Note: The option Use Accelerated Rendering significantly speeds up the appearance of the popup, but may require a change in related scripts if applicable. For example, with this option enabled, the method *action.getPopupContext()* will have to be replaced with *action.getPopupCanvasAdapter()*.

Click **Set up parameters**.

In the *Set up Parameter Mappings* dialog, click **Add new mapping**.

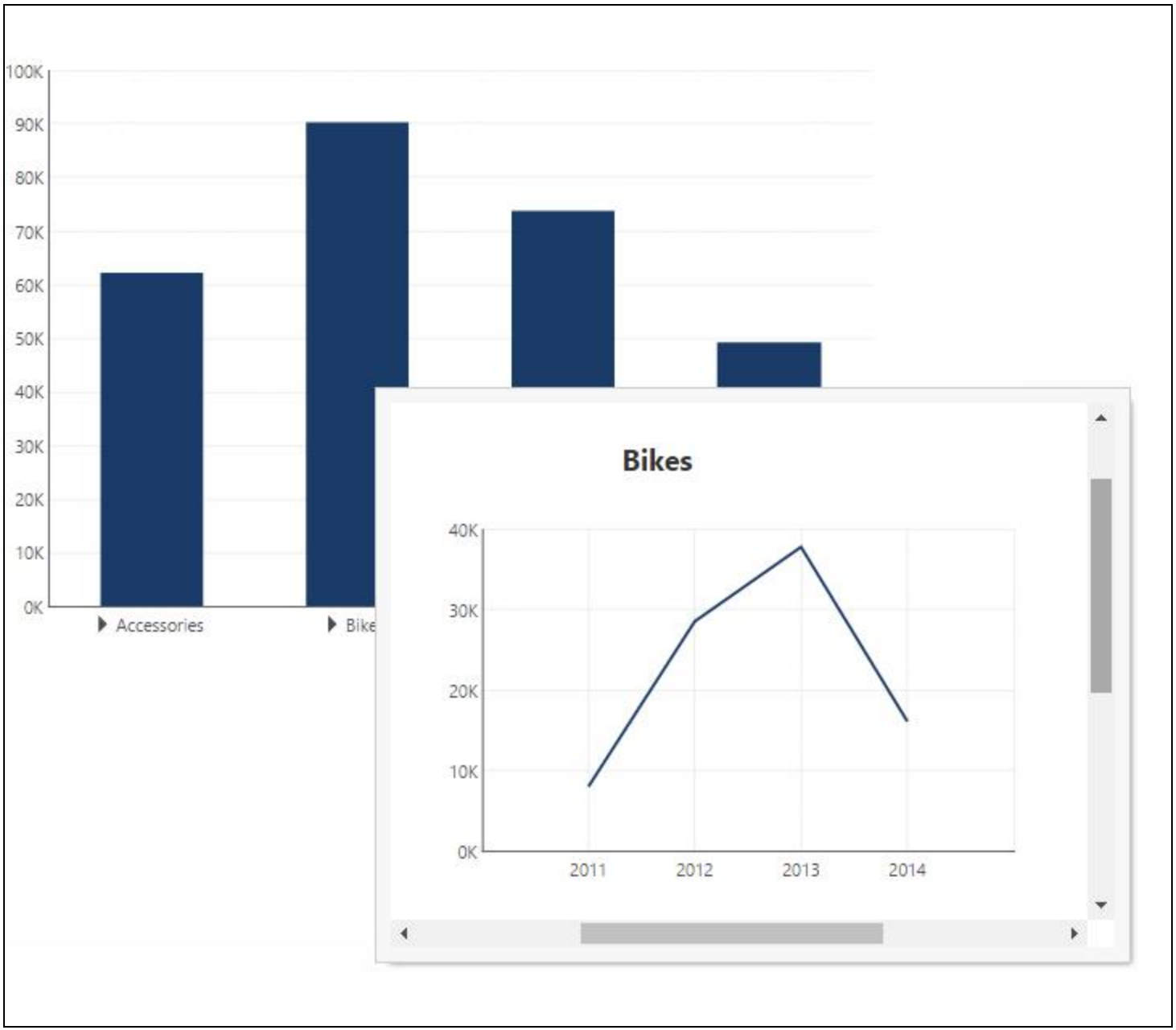


In the **Source** list, select the **ProductID** filter checkbox. In the **Target** list, select **viewParameter1**. This forms a mapping so that when the user hovers over a data point (**Product**) on **Dashboard 1**, the corresponding product value will be passed to the view parameter on **Dashboard 2**. Since the view parameter is hooked up to the Product slicers hierarchy, the line chart on **Dashboard 2** will be filtered accordingly when it is displayed in a popup.


Set Up Parameter Mappings
This allows you to set up how source information is passed to the target when the action is run.
CURRENT PARAMETER MAPPINGS
ProductID ([Production].[Product]) (filter value) to viewParameter 1
Add new mapping
Select one source from the left to map to one target view parameter on the right. If a URL is being used as the target, a string placeholder target must be specified which will be filled in with the mapped source data.
SOURCE
TARGET
Clicked Item
Dashboard2
OrderQty
roductID ([Production].[Product]) (Filter Value)
ProductID ([Production].[Product]) (Level Value)
Brush View Parameter
iewParameter 1
(All View Parameters)

Test the Interaction

Switch to **View** mode and hover over a data point (**Bikes** product) on **Dashboard 1**. You should see **Dashboard 2** displayed in a popup that shows *OrderQty by Date* filtered by the product in question.



For more information, see:

- Using Interactions
- Set Up a Navigate Interaction
- Design Overview

Set Up a Navigate Interaction

This applies to: *Managed Dashboards, Managed Reports*

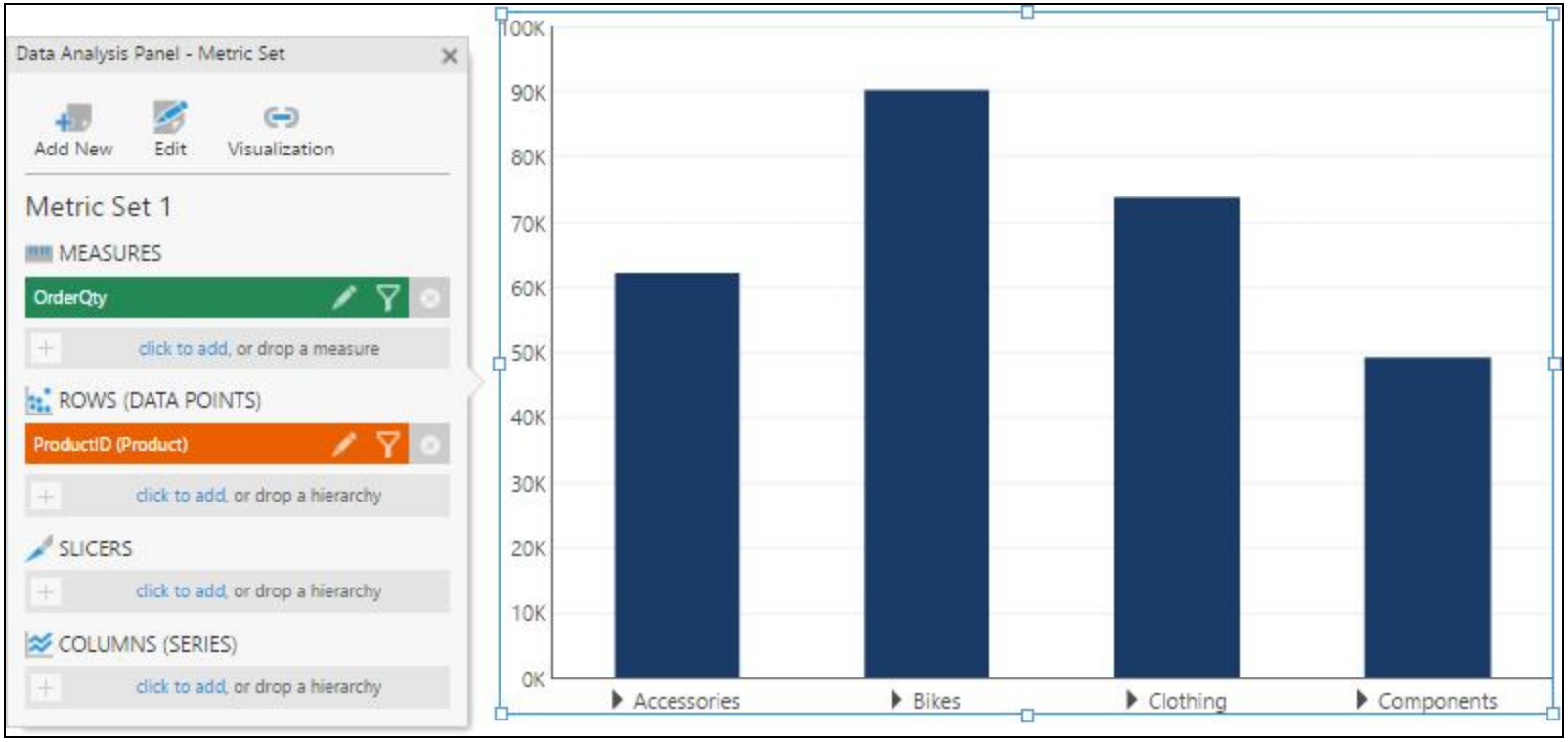
This article shows you how to set up a navigate interaction so that when you click on a data point, the current view is replaced with a second view such as a dashboard showing filtered data corresponding to the clicked data point. You can also set up navigation on any component such as buttons or images.

You can navigate to any type of view, including dashboards, reports, scorecards, and small multiples, or to any webpage via its URL, and optionally pass information about what data was clicked as a parameter. Passing parameters to a URL is shown in Navigate to URL.

Related video: Interactions

Create the First Dashboard

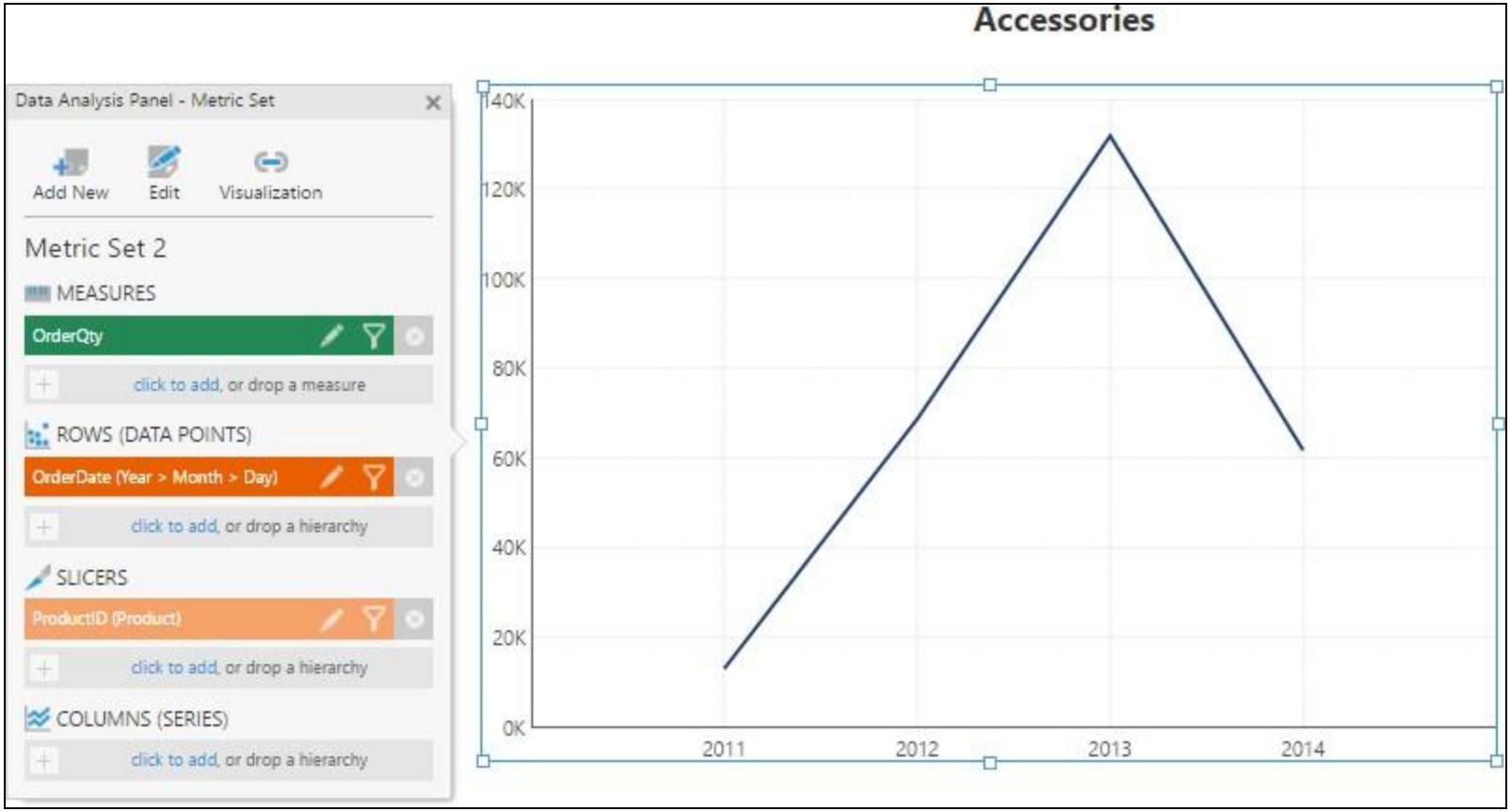
Create a metric set and add it to a dashboard in order to show **OrderQty by Product** for example.



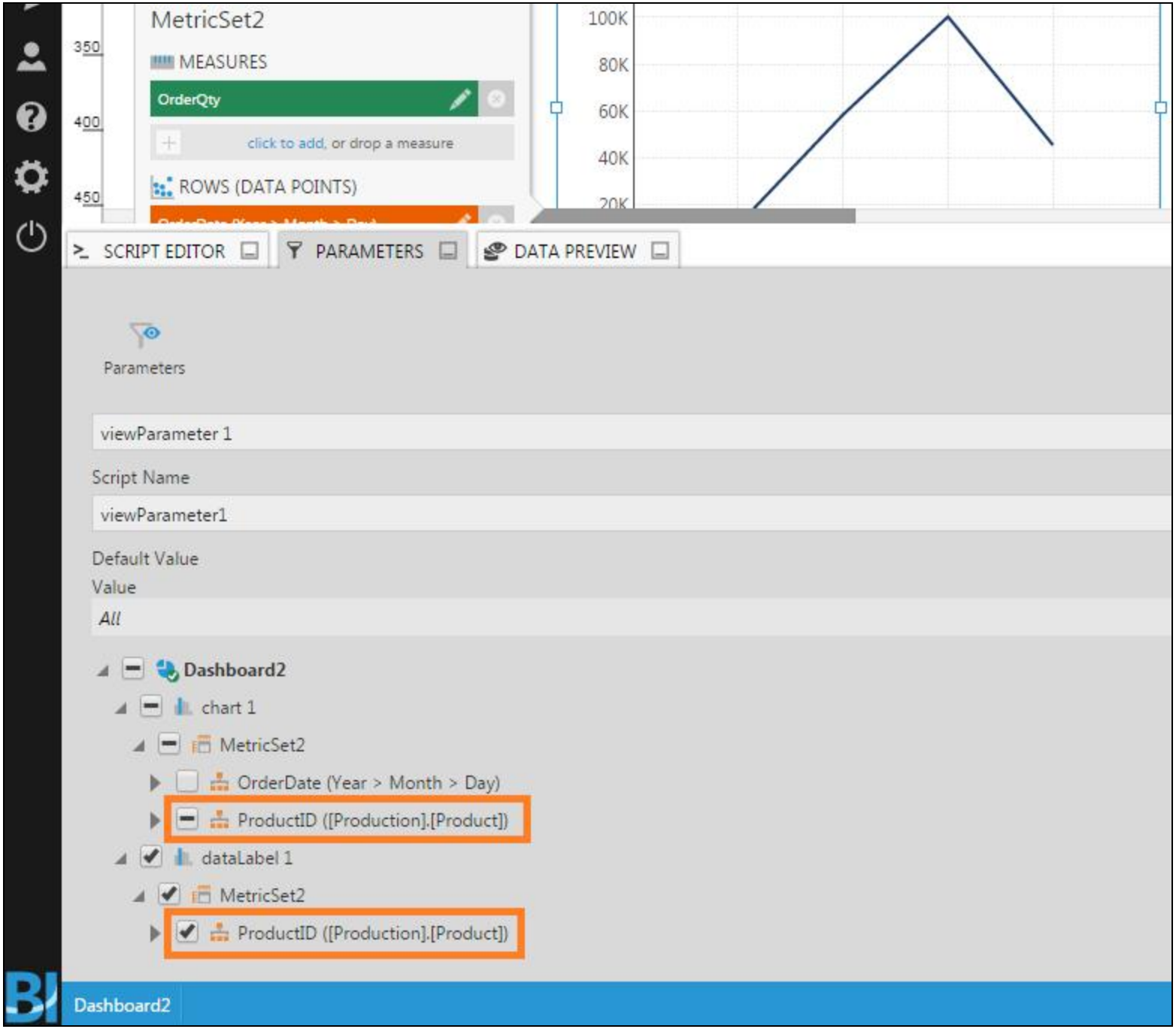
Create a Second Dashboard

Create another metric set and add it to a second dashboard to navigate to from the first one, which will show *OrderQty by Date* in our example. In the Data Analysis Panel, add *Product* under **Slicers** to be able to filter by the product clicked on in the first dashboard.

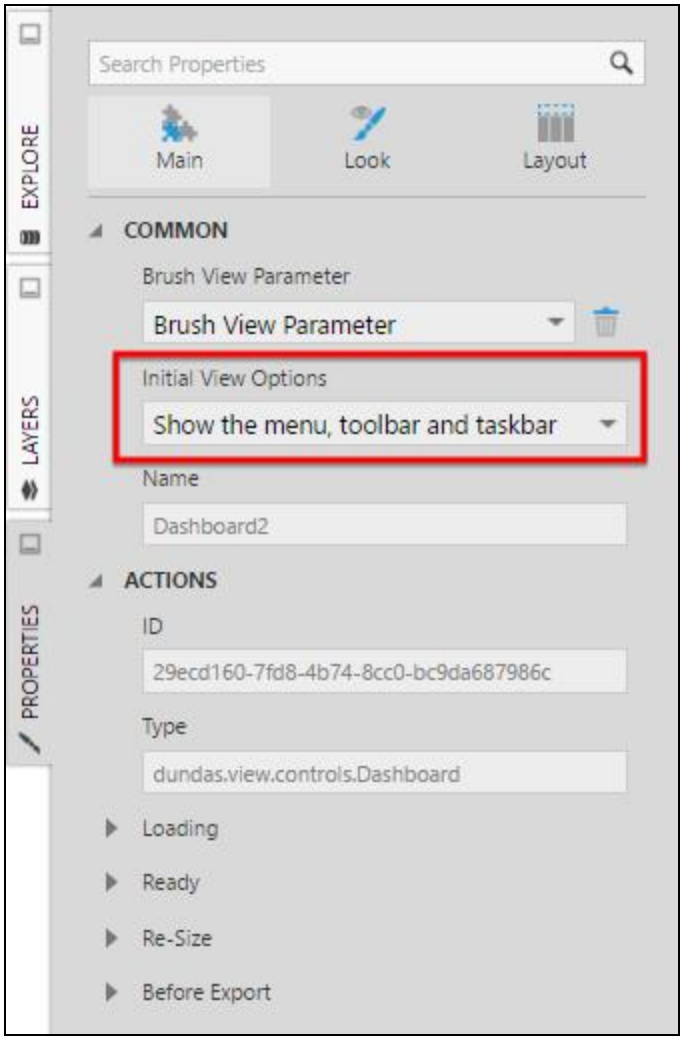
Besides the chart, also add a data label control and drag *Product* onto it, so that we can display the filter value passed from the first dashboard.



Next, expand the **Parameters** window which is docked along the bottom of the screen. Click **Add New** to add a new view parameter and then connect it to the Product hierarchy for both the chart and the data label.

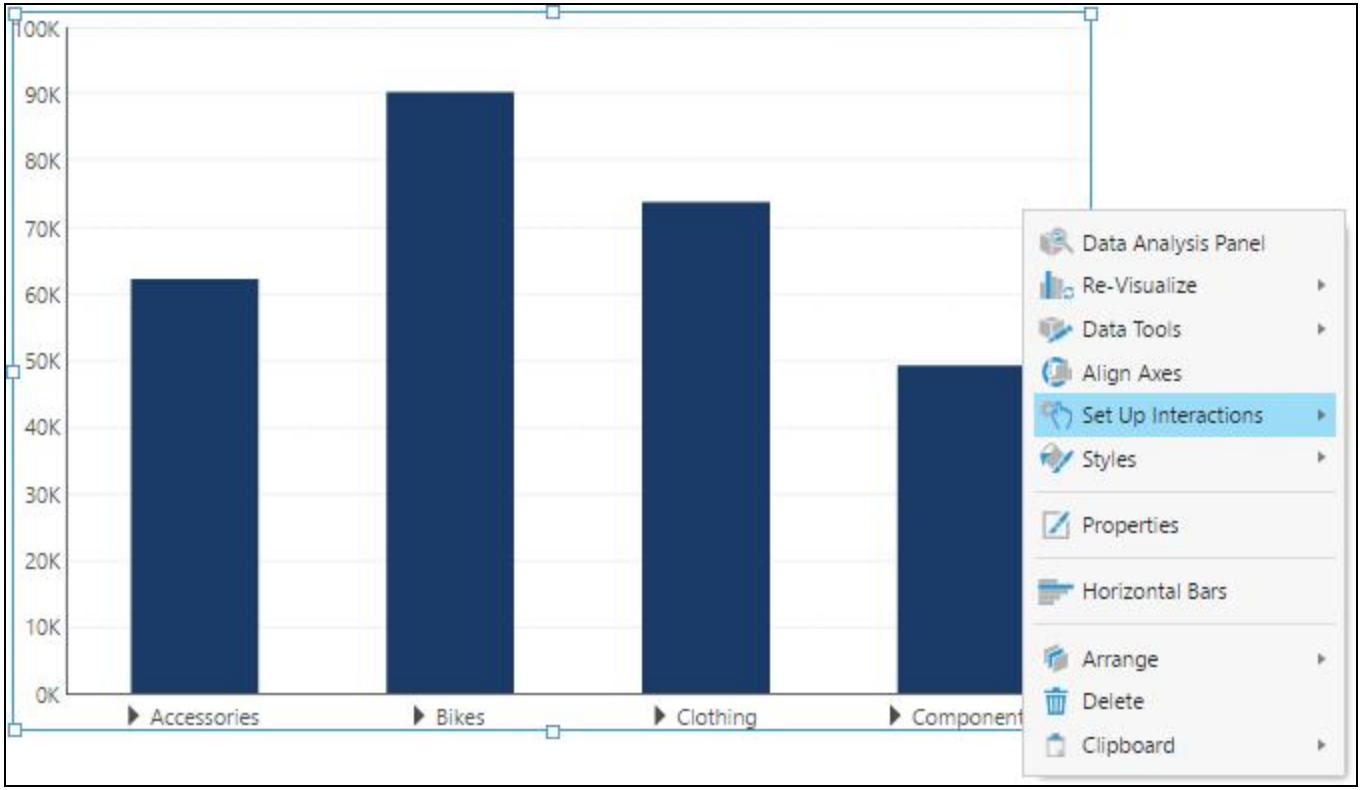

MetricSet2
MEASURES
OrderQty
click to add, or drop a measure
ROWS (DATA POINTS)
350
400
450
100K
80K
60K
40K
20K
SCRIPT EDITOR
PARAMETERS
DATA PREVIEW
Parameters
viewParameter 1
Script Name
viewParameter1
Default Value
Value
All
Dashboard2
chart 1
MetricSet2
OrderDate (Year > Month > Day)
ProductID ([Production].[Product])
dataLabel 1
MetricSet2
ProductID ([Production].[Product])
Dashboard2

Finally, click away from visualization to de-select it and select the dashboard itself. Open the **Properties** window, and as an example change the **Initial View Options** to *Show the menu, toolbar and taskbar*.

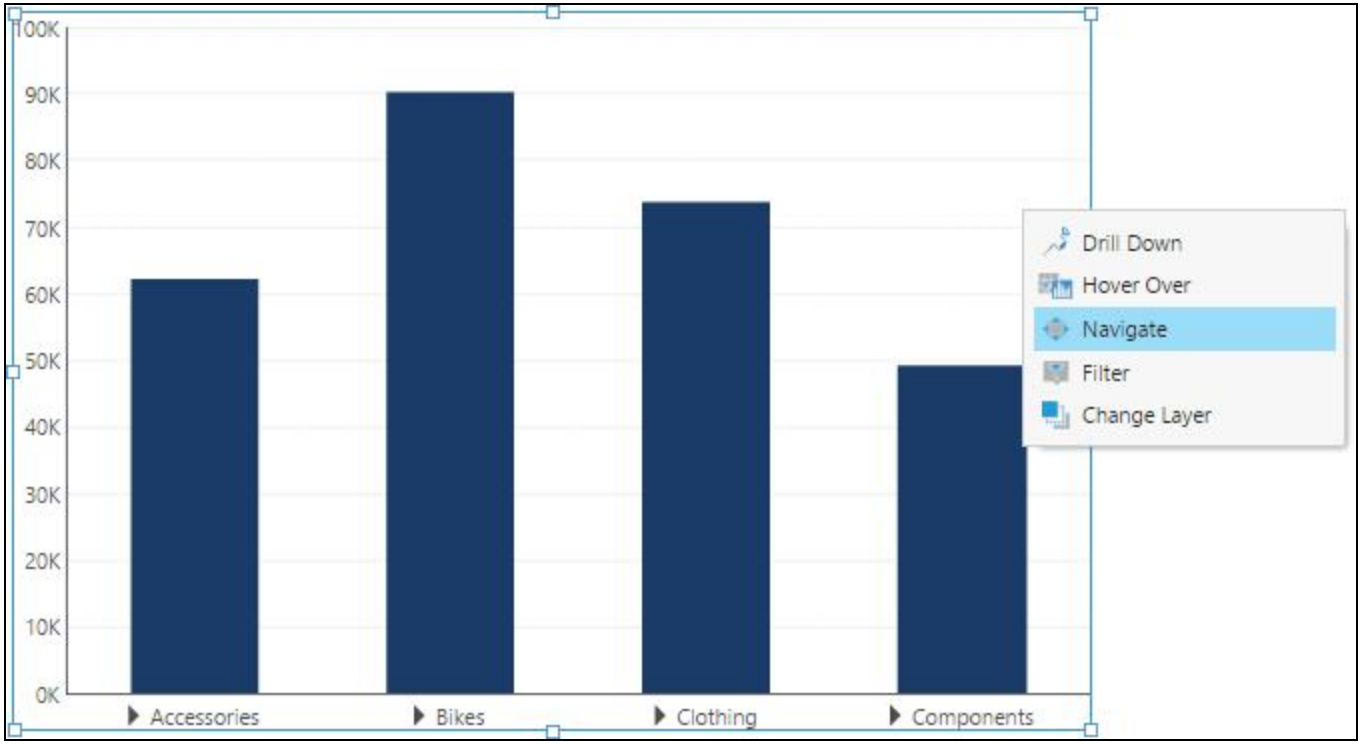


Set Up the Navigate Interaction

Edit the first dashboard again. Right click (or long-tap) over the bar chart and click **Set Up Interactions** from the menu.



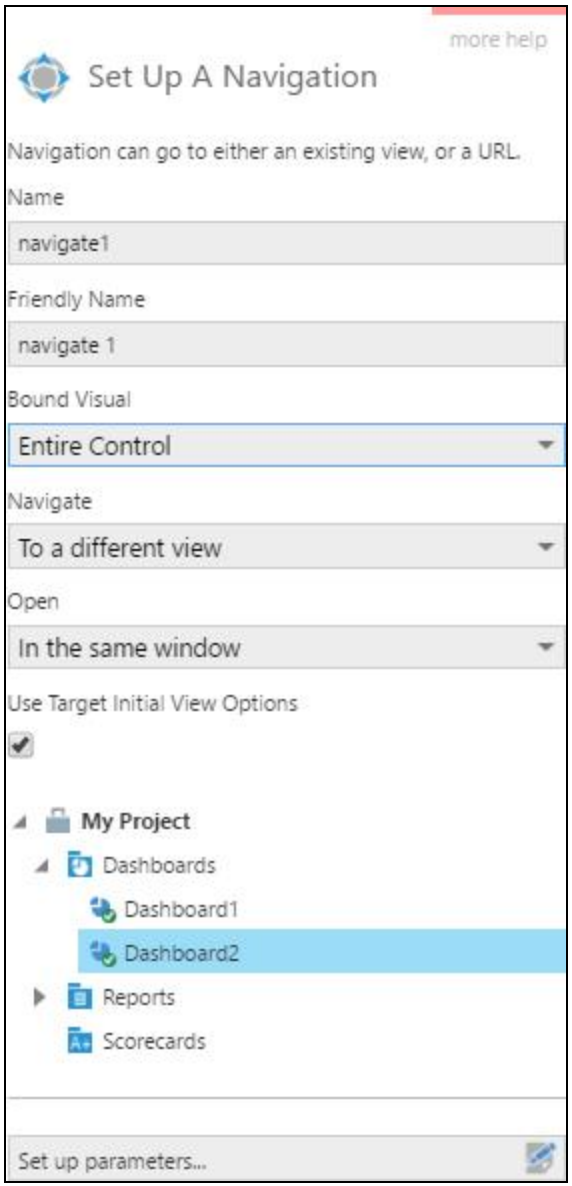
Click **Navigate**.



In the *Set up a Navigation* dialog, set the navigation to go *To a different view* and open *In the same window*.

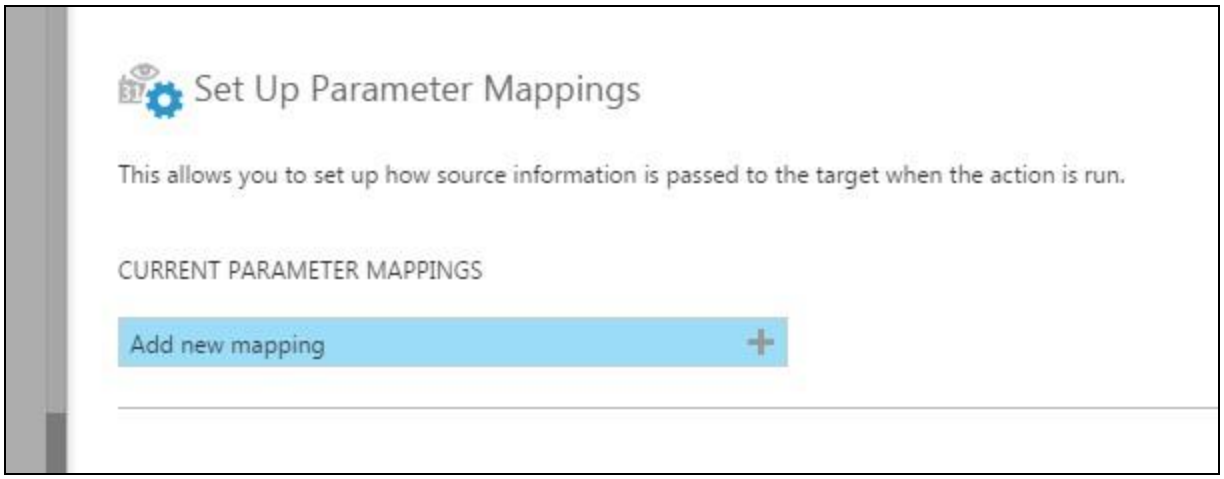
Select **Use Target Initial View Options** to use the settings in Dashboard 2 rather than the default full-screen view.

Select *Dashboard2* as the different view to navigate to.



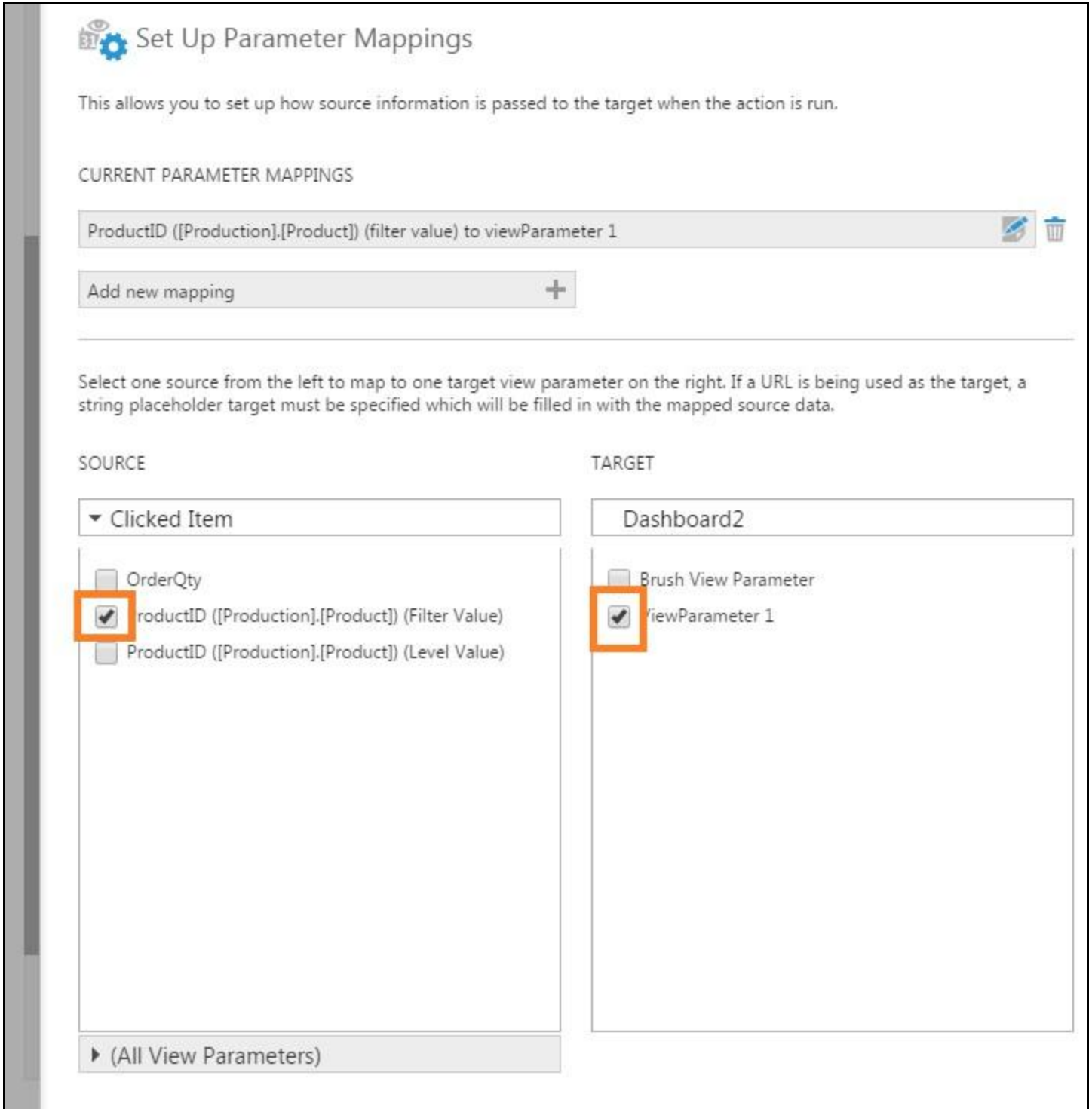
Scroll further down and click **Set up parameters**.

In the *Set up Parameter Mappings* dialog, click **Add new mapping**.



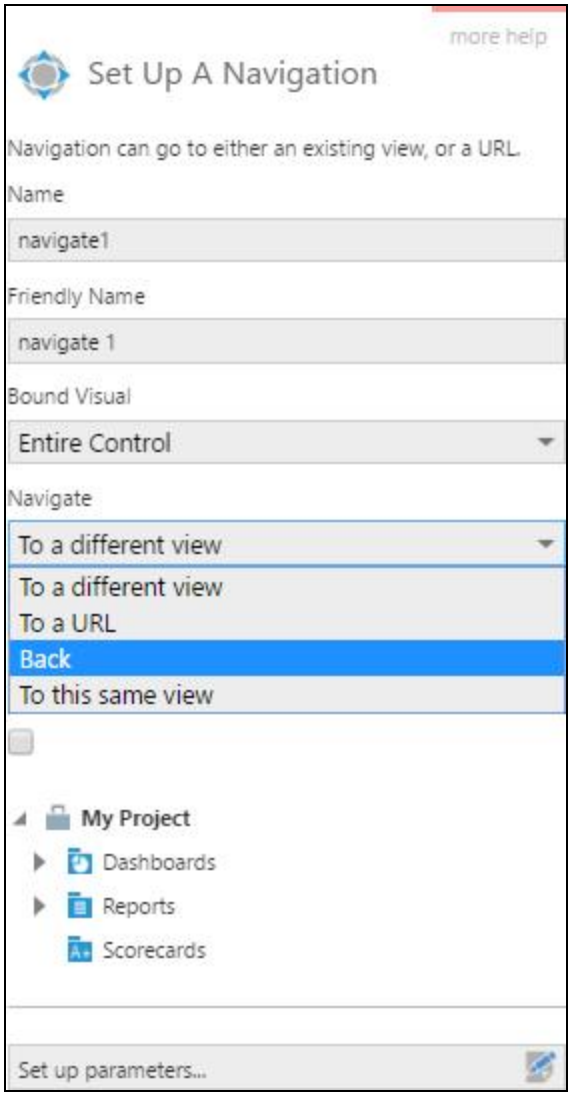
In the **Source** list, select the *ProductID* filter checkbox. In the **Target** list, select *viewParameter1* (or whatever you named the parameter on your second dashboard, for example).

This forms a mapping so that when the user clicks on a data point on Dashboard1, the clicked Product value will be passed to the parameter on Dashboard2, which will then filter the line chart and data label that we connected to that view parameter earlier.


Set Up Parameter Mappings
This allows you to set up how source information is passed to the target when the action is run.
CURRENT PARAMETER MAPPINGS
ProductID ([Production].[Product]) (filter value) to viewParameter 1
Add new mapping
Select one source from the left to map to one target view parameter on the right. If a URL is being used as the target, a string placeholder target must be specified which will be filled in with the mapped source data.
SOURCE
TARGET
Clicked Item
Dashboard2
OrderQty
roductID ([Production].[Product]) (Filter Value)
ProductID ([Production].[Product]) (Level Value)
Brush View Parameter
iewParameter 1
(All View Parameters)

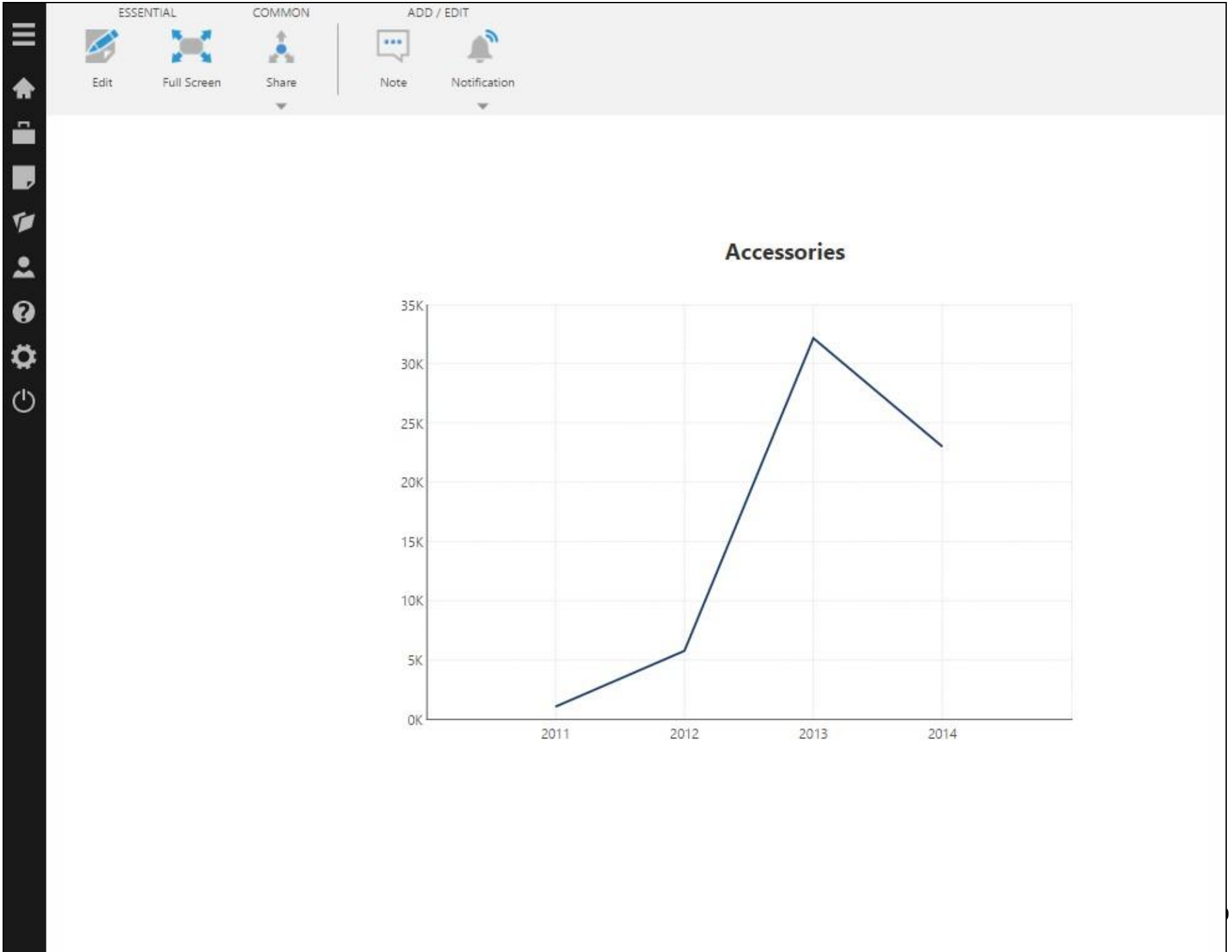
Add Another Navigate Interaction To Go Back

Edit the second dashboard. Follow similar steps as for the first dashboard in order to add a navigate interaction to the chart. Except this time, *Navigate* should be set to *Back*.



Test the Interaction

View the first dashboard, and then click a data point (e.g., *Accessories*) on Dashboard1. You should see the view replaced by Dashboard2, which shows *OrderQty by Date* filtered by the value that you clicked on.


ESSENTIAL
COMMON
ADD / EDIT
Edit
Full Screen
Share
Note
Notification
Accessories
35K
30K
25K
20K
15K
10K
5K
0K
2011
2012
2013
2014

Observe that the menu, toolbar, and taskbar are visible.

Click the line chart to go back to Dashboard1.

For more information, see:

- Using Interactions
- Set Up a Hover Over Interaction
- Navigate to URL
- Design Overview

Use a Button to Apply a Filter Value

This applies to: *Managed Dashboards, Managed Reports*

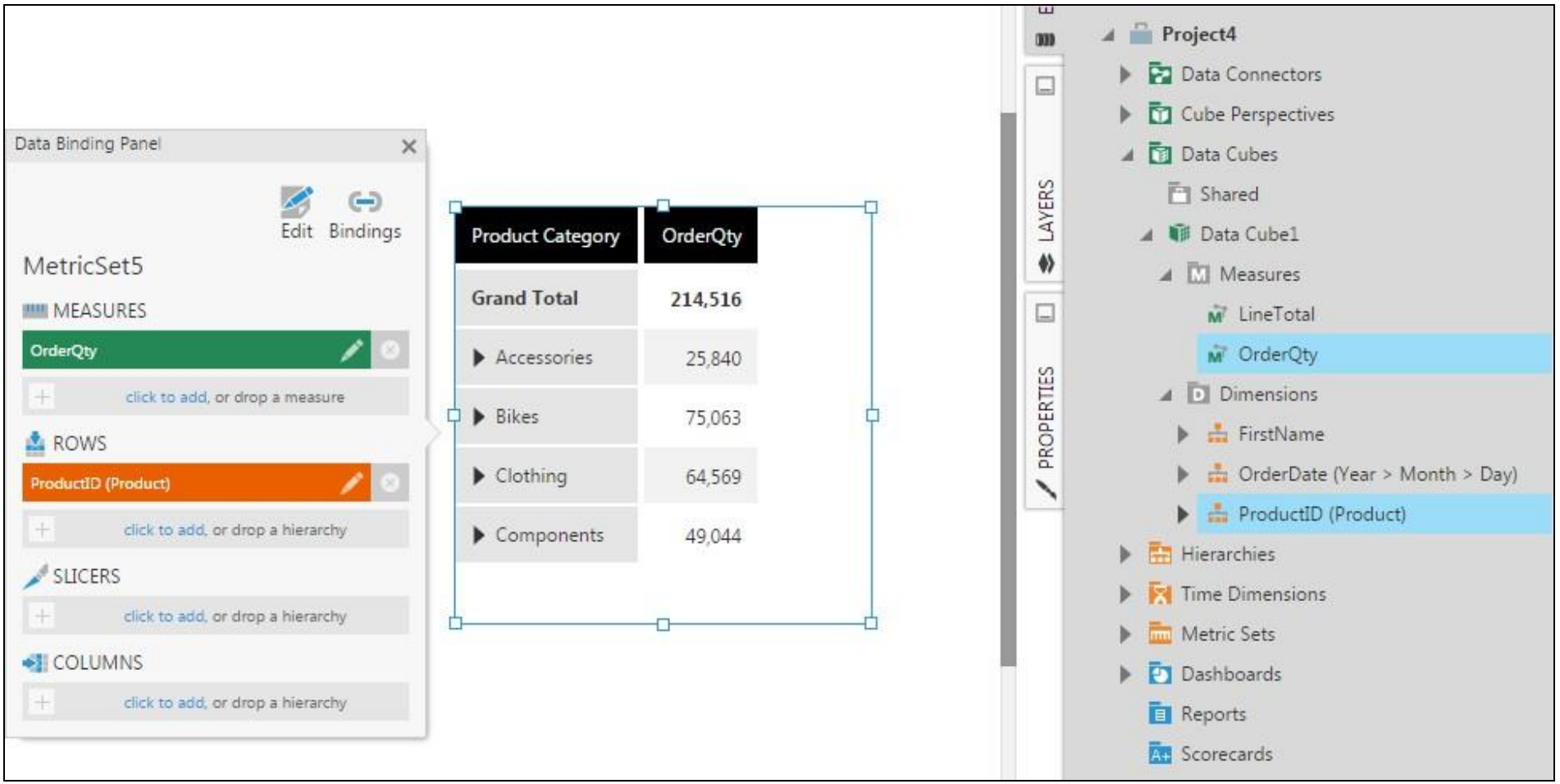
This article shows you how to set up a filter interaction on a component such as a button, which will filter a visualization when clicked.

The following example uses a button, but other non-data components such as labels can be set up the same way.

Create a New Dashboard

As an example, create a new dashboard using the *Blank* template.

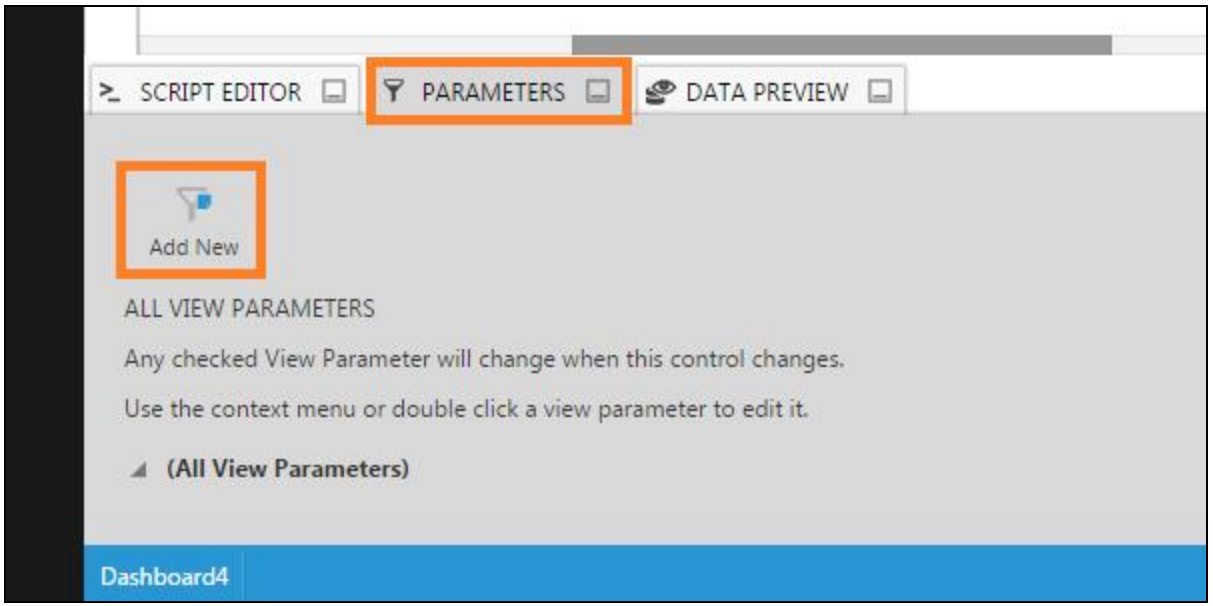
Drag a measure (OrderQty) and row hierarchy (Product) from a data cube to the dashboard canvas. You'll see a table visualization like the one below.



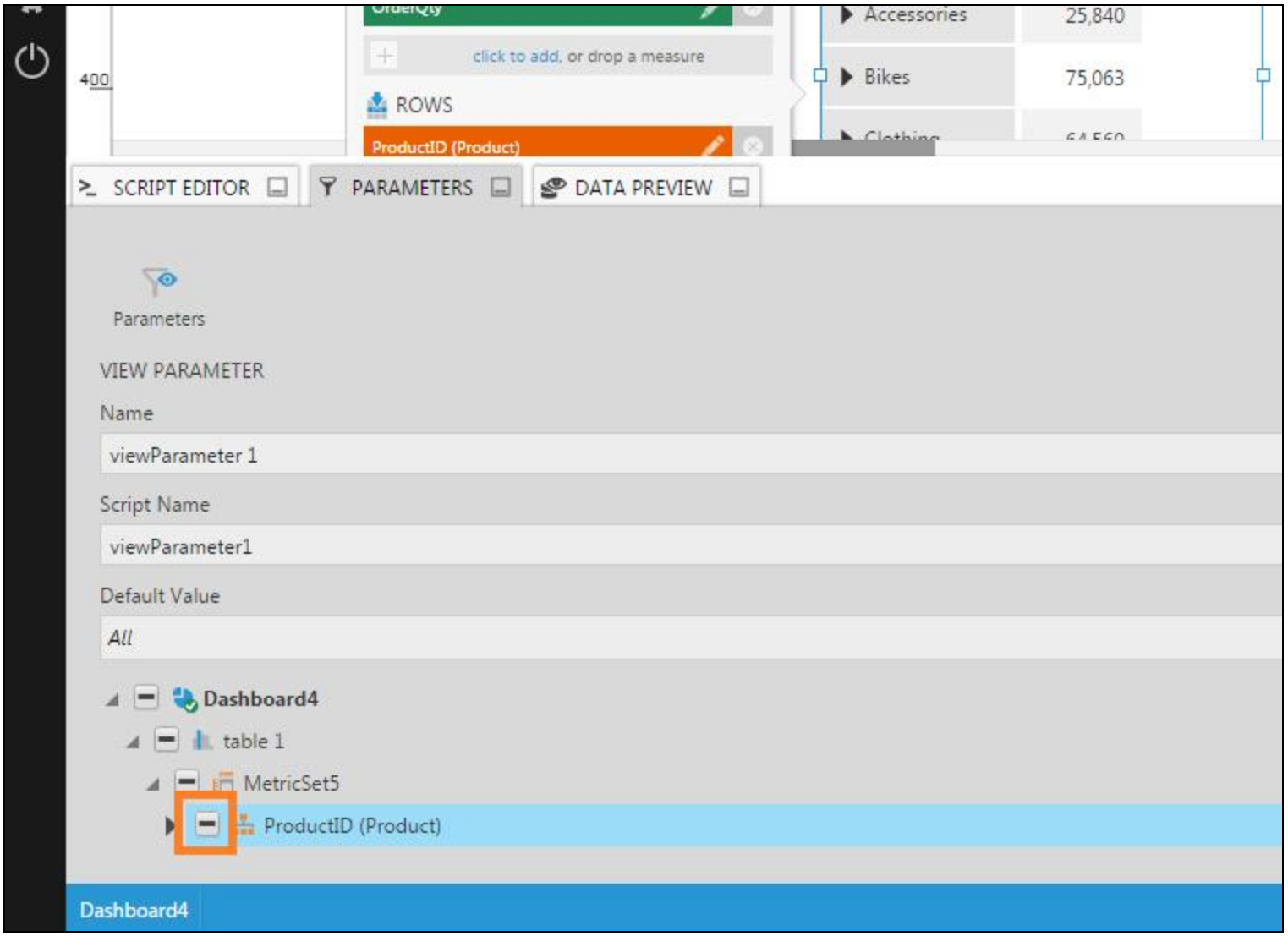
Add a View Parameter

Go to the bottom of the dashboard editor and click to open the **Parameters** window.

In our case, we will click **Add New** to add a view parameter for this visualization. You can skip this step if you already have one from another interaction or a filter that you want to use.

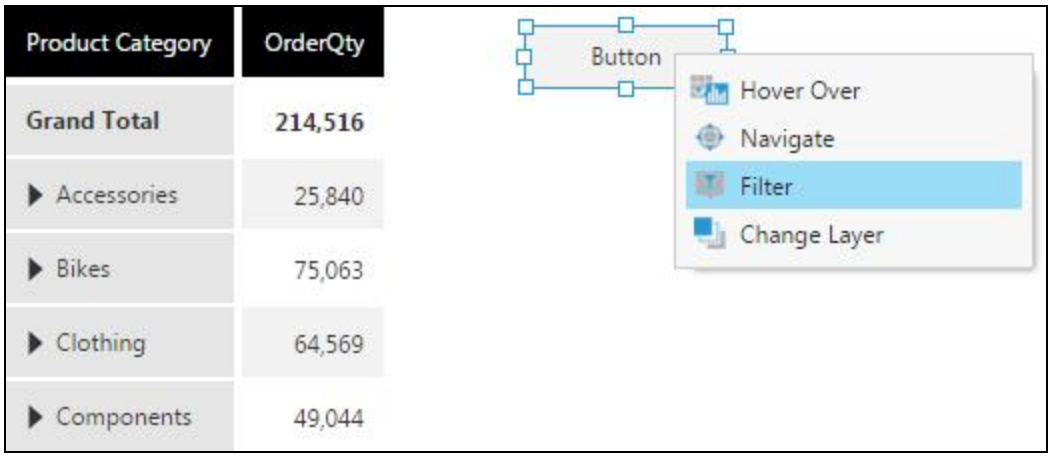


We will connect the new view parameter to the Product hierarchy in our table visualization. You can give it a descriptive **Name**.



Add a Button Component and Set Up Filter Interaction

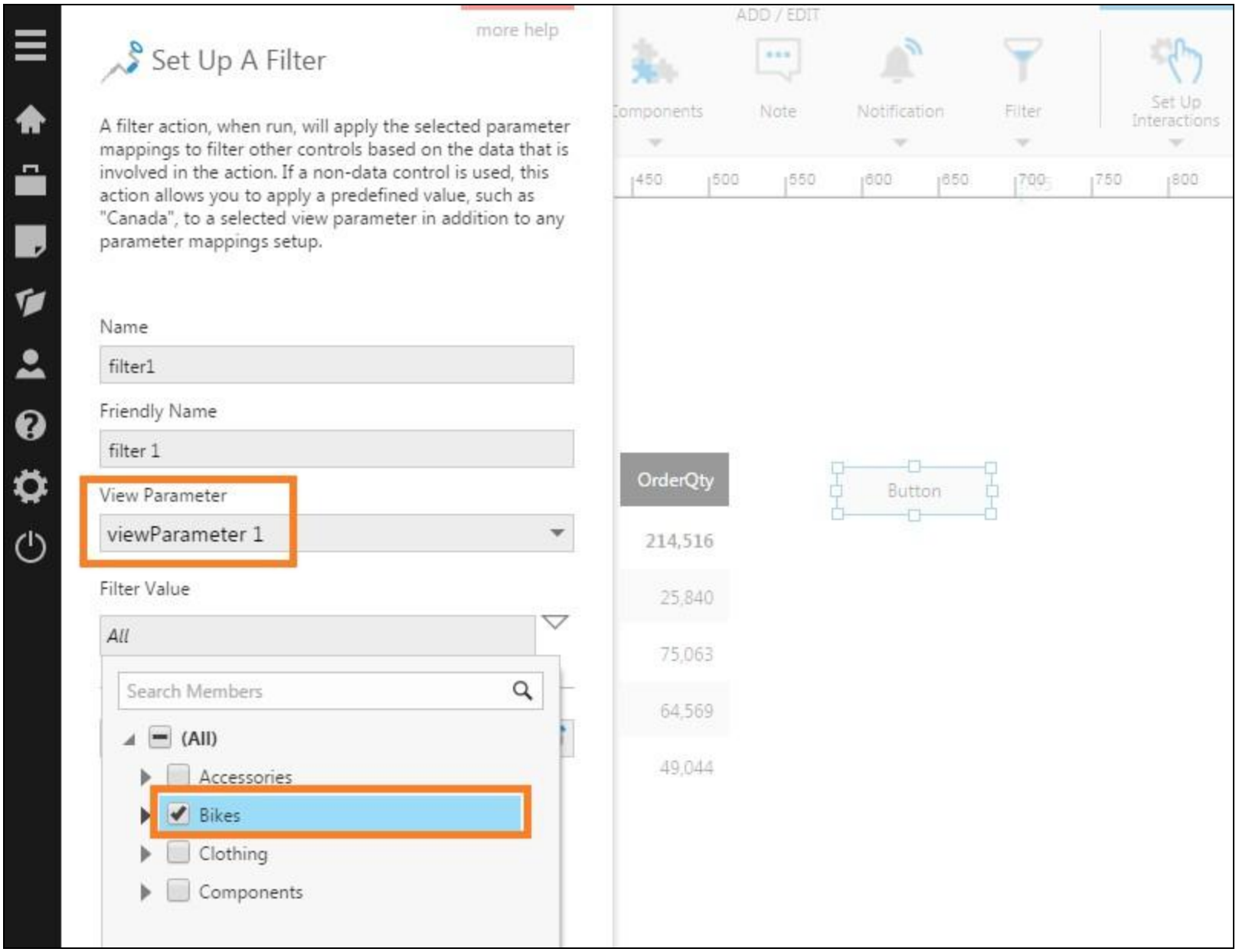
Click **Components** in the toolbar, and in this example, we will select to add a **Button** to filter our *Product* data to a particular value. In the toolbar or right-click menu for the button, click **Set Up Interactions**, then click **Filter**.



Note: To set multiple view parameters at once on button click, you can add additional Filter actions on Click in the Properties window.

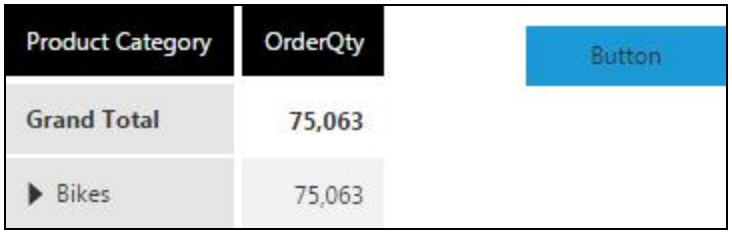
In the filter setup dialog, set the **View Parameter** to the view parameter you want to change on button click.

Under **Filter Value**, select the specific filter value you want the button to apply.


more help
Set Up A Filter
A filter action, when run, will apply the selected parameter mappings to filter other controls based on the data that is involved in the action. If a non-data control is used, this action allows you to apply a predefined value, such as "Canada", to a selected view parameter in addition to any parameter mappings setup.
Name
filter1
Friendly Name
filter 1
View Parameter
viewParameter 1
Filter Value
All
Search Members
(All)
Accessories
Bikes
Clothing
Components
ADD / EDIT
Components
Note
Notification
Filter
Set Up Interactions
450
500
550
600
650
700
750
800
OrderQty
214,516
25,840
75,063
64,569
49,044
Button

View the Dashboard

Switch to **View** mode to test the dashboard. Click the button to see the table visualization filtered to the value you selected during setup.



For more information, see:

- Set Up a Filter Interaction
- Adding Filters

Using a Template Grid for Resizing

This applies to: *Managed Dashboards, Managed Reports*

Dashboards can adapt to different screen sizes, such as monitors, widescreen TVs, and tablets.

A *template grid* can be added to any dashboard for an easy drag-and-drop grid layout, but you can also use it in a resizable or responsive dashboard to resize its contents.

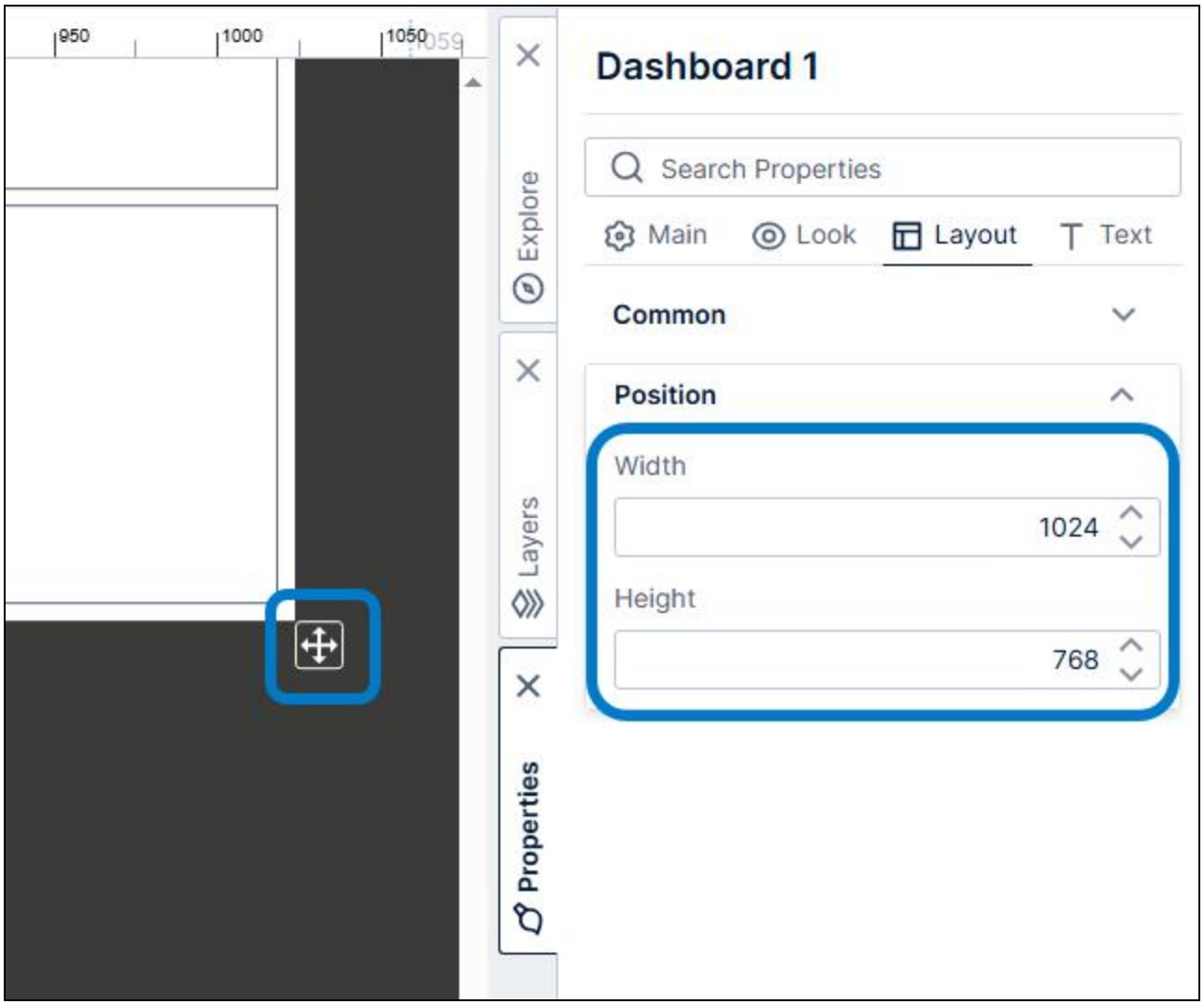
Related videos:

- Template Grid
- Resize Mode

Set the View Size

The size set up for a dashboard when editing is often used to determine its size and layout when viewing.

When editing, you can drag the resize icon in the bottom right corner.



This updates the dashboard's **Width** and **Height** properties accordingly, found in the **Properties** window in the **Layout** tab, or you can also set these properties directly.

The Re-Size Mode Property

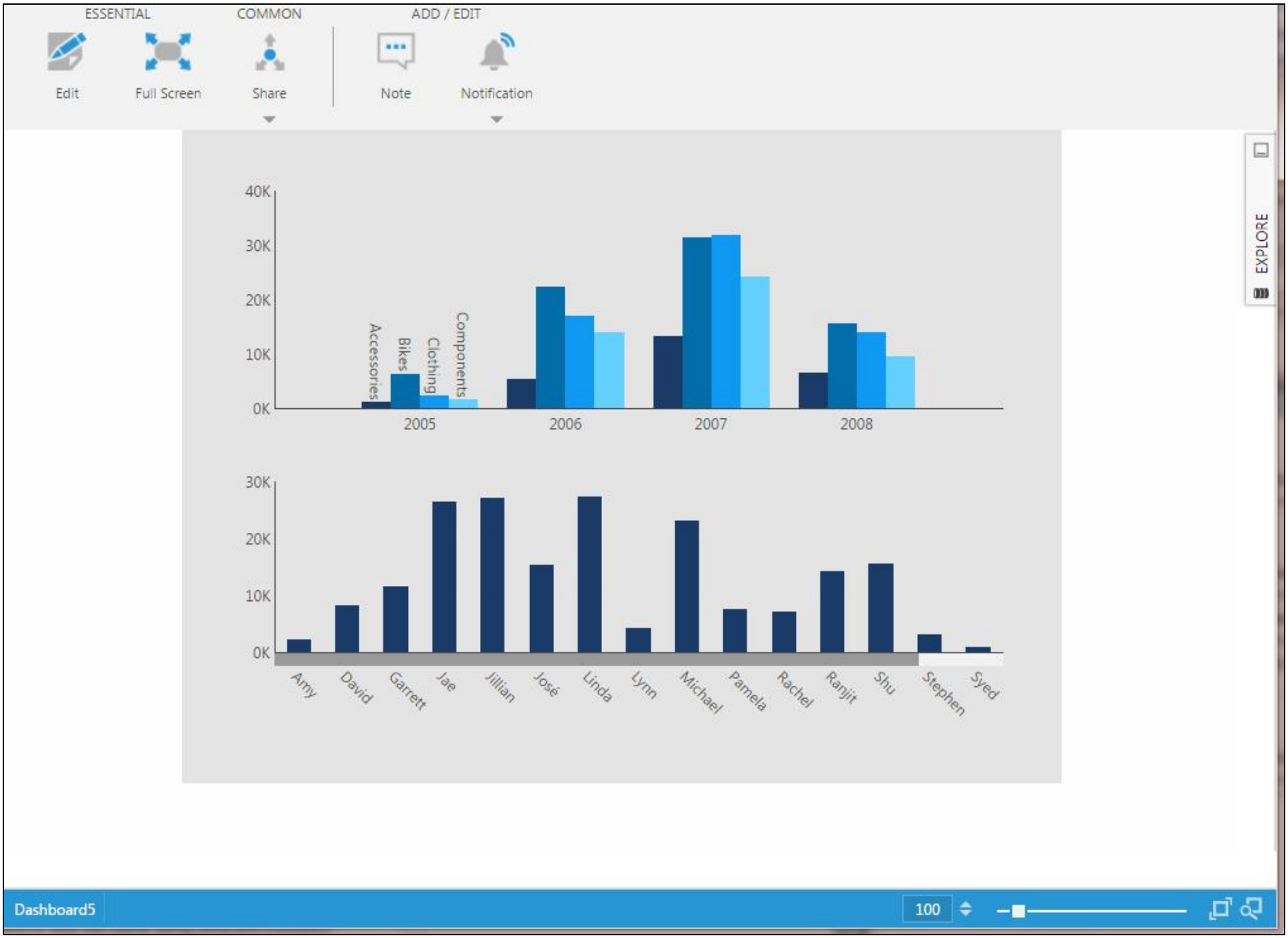
When viewing the dashboard, its size can change according to the dashboard's **Re-Size Mode** property, which has four options: *Scroll*, *Scale*, *Resize*, and *Responsive*.

This property is found in the **Properties** window for the dashboard in the **Layout** tab, above the **Width** and **Height** properties. (If an element on the dashboard is selected, click an empty part of the dashboard to access its properties.)

Scroll

When you create a new dashboard, its **Re-Size Mode** property is initially set to scroll. In this mode:

- The dashboard does not change size (width or height) when you resize your browser window.
- The elements on the dashboard (e.g., charts, filters, and components) do not change size or position relative to the dashboard when the browser window is resized.
- If the browser window is resized large enough, the dashboard will be centered horizontally and positioned at the top of the viewing area.


ESSENTIAL
COMMON
ADD / EDIT
Edit
Full Screen
Share
Note
Notification
EXPLORE
40K
30K
20K
10K
0K
Accessories
Bikes
Clothing
Components
2005
2006
2007
2008
30K
20K
10K
0K
Amy
David
Garrett
Jae
Jillian
José
Linda
Lynn
Michael
Pamela
Rachel
Ranjit
Shu
Stephen
Syed
Dashboard5
100

As the browser window is resized smaller, scrollbars will appear, allowing you to scroll the parts of the dashboard you want to see into view.

- You can use the zoom slider and the view panel in the status bar to zoom in, zoom out, or pan the dashboard.
- 'Pinch-to-zoom' will also work in scroll mode on touch devices.
- No further design work is necessary other than deciding on the width and height of the dashboard. The dashboard contents will look more or less the same

using different monitors, devices, and resolutions.

- If your dashboard is very large, important information may be scrolled out of view initially and potentially missed by a viewer.

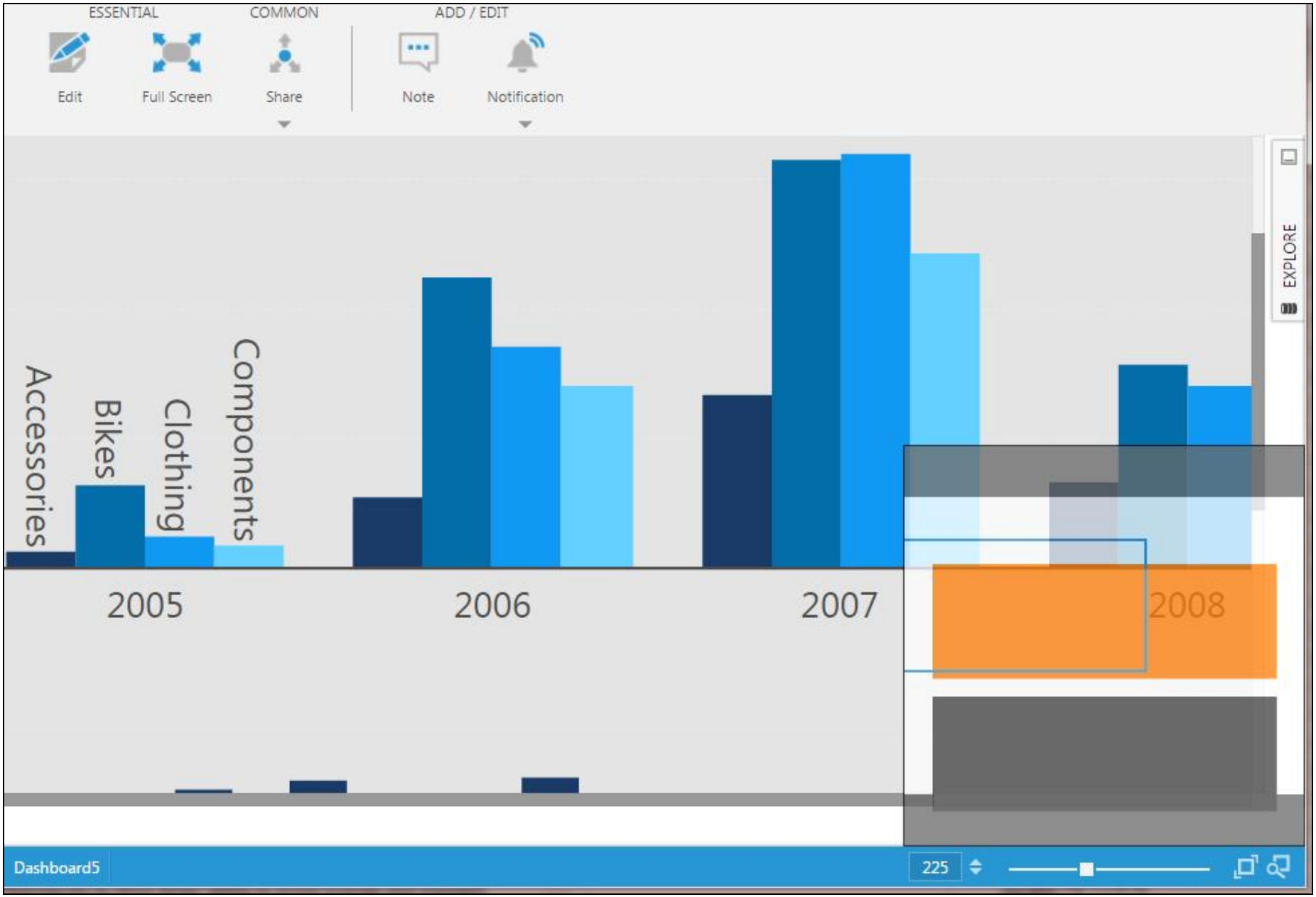
Scale

When the **Re-Size Mode** property of a dashboard is set to scale, the dashboard and its contents will be scaled to fit the available viewing area. In this mode:

- The entire dashboard will be initially visible without any scrollbars present. This is also the behavior whenever the browser window is resized.

- By default, the aspect ratio of the dashboard is maintained for different screen and window sizes. If you don't want the aspect ratio to be maintained, enable the dashboard's property option **Ignore Aspect Ratio On Re-Size**.

- You can use the zoom slider and the view panel in the status bar to zoom in, zoom out, or pan the dashboard. Scrollbars will appear as needed when zooming in.


ESSENTIAL
COMMON
ADD / EDIT
Edit
Full Screen
Share
Note
Notification
EXPLORE
Accessories
Bikes
Clothing
Components
2005
2006
2007
2008
Dashboard5
225

- 'Pinch-to-zoom' will also work in scale mode on touch devices.
- The downside of using the scale mode is that content may initially be too small to see when the screen or window is too small.

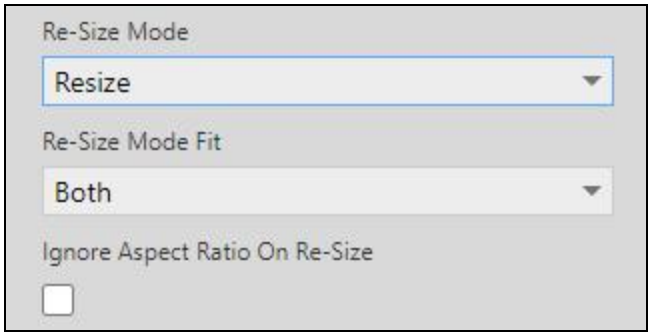
Resize

The resize mode is a smart resizing option that is designed to be used in conjunction with a *template grid*, which allows you to set up the elements of your dashboard for resizing within a grid-like layout. In this mode:

The dashboard canvas is resized to fit the available viewing area without scrollbars.

The zoom slider and view panel are not available in this mode.

The **Re-Size Mode Fit** property determines if the dashboard should be resized to fit the width of the screen, the height of the screen, or both.



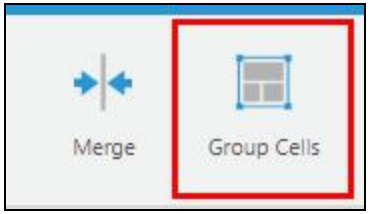
By default, the aspect ratio of the dashboard is maintained when resizing. If you always want content to be resized to fill all available screen space, enable the **Ignore Aspect Ratio On Re-Size** property option.

Elements on the dashboard will resize along with the dashboard and the screen or window provided they have been attached to the template grid's cells (see below). This lets you take advantage of any smart layout or labeling logic that may be built into visualizations. For example, when a chart is resized smaller, the chart axes adjust the label text size and layout to avoid overlap and maintain readability.

Responsive

The responsive mode is also designed to be used in conjunction with a *template grid*, but takes smart resizing a step further. This mode reflows and wraps the individual cells from left to right and then from top to bottom according to the available space on the screen, allowing the same dashboard to fit both the largest screens and the smallest screens automatically. In this mode:

- The width of the dashboard canvas is resized to fit the available width of the viewing area.
- The cells in the template grid may be rearranged into additional rows if there is less width, or to fit additional cells in each row if there is more width.
- Vertical scrolling is used to see additional content if it does not all fit on the screen or in the window.
- The layout of the cells on the canvas is computed based on the page size and is recomputed when resizing. For example, the layout will change when a mobile device changes from portrait to landscape mode.
- You can use *cell group*s to keep cells together when the layout is computed.



- You can turn a cell or cell group into a header row if it spans the entire row to preserve it and ensure cells are not added or removed when viewing. You can also designate a header row inside a cell group.

For additional information on Responsive mode, see Responsive Layout Dashboard Mode.

Define a Template Grid

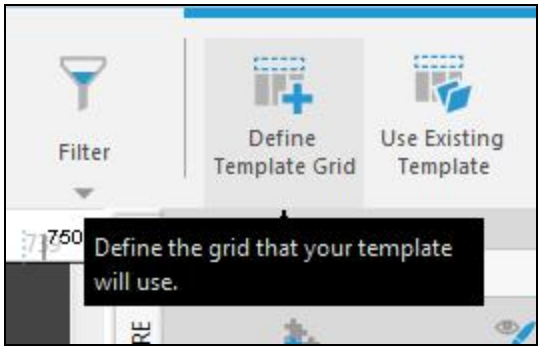
This section shows you how to define a template grid and use it to resize components on a dashboard with **Re-Size Mode** set to Resize or Responsive.

Set Up the Dashboard

Set the dashboard's **Re-Size Mode** property to *Resize or Responsive* to take advantage of the template grid for resizing the content. You can resize the dashboard to a more suitable size to use while editing, or you can change it later.

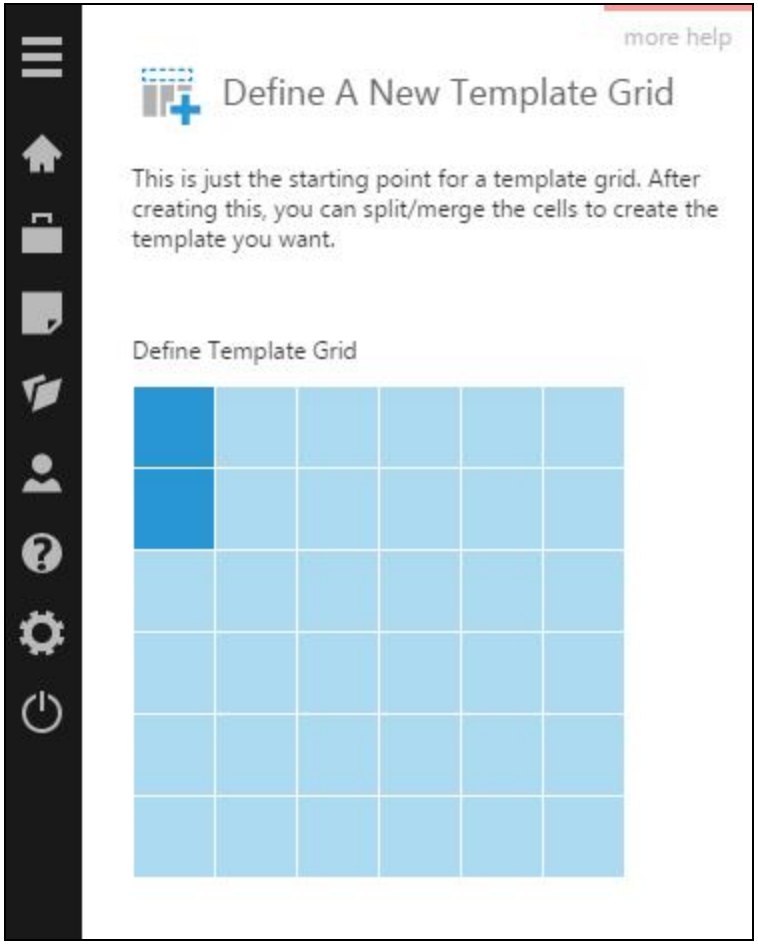
Add a Template Grid

In the toolbar, click **Define Template Grid**. (If an element on the dashboard is selected, first click an empty part of the dashboard to see this option.)

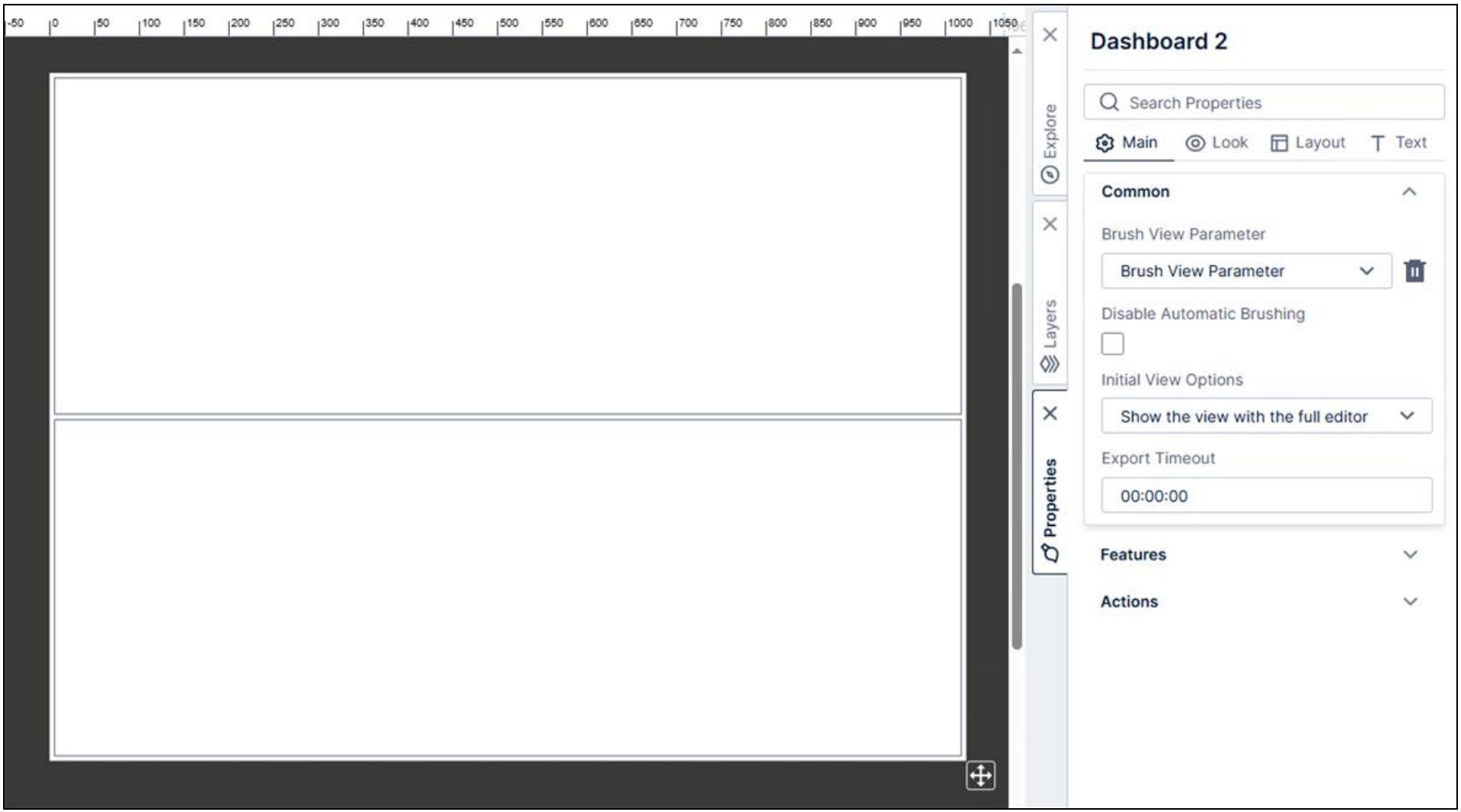


In the *Define a New Template Grid dialog*, grid cells are highlighted as you move the mouse.

Move your mouse around until the desired layout of grid cells is highlighted (e.g., 2x3 or 3x3) and then click to choose it.



Click Submit to go back to your dashboard. You'll see that a grid of cells has been added.

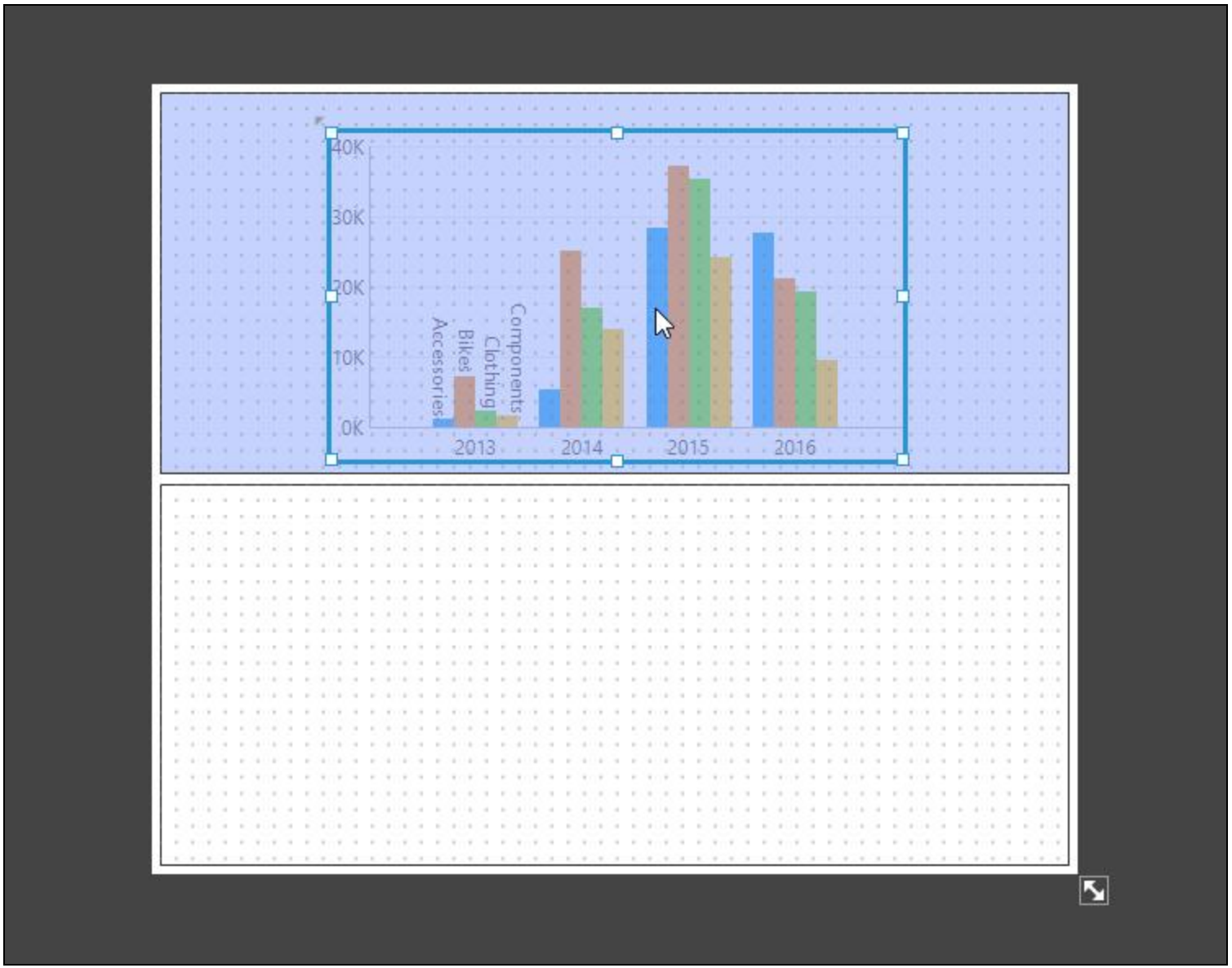


Add Elements to Grid Cells

Each element on the dashboard needs to be attached or added to one of the template grid cells in order for it to resize with the rest of the dashboard or screen.

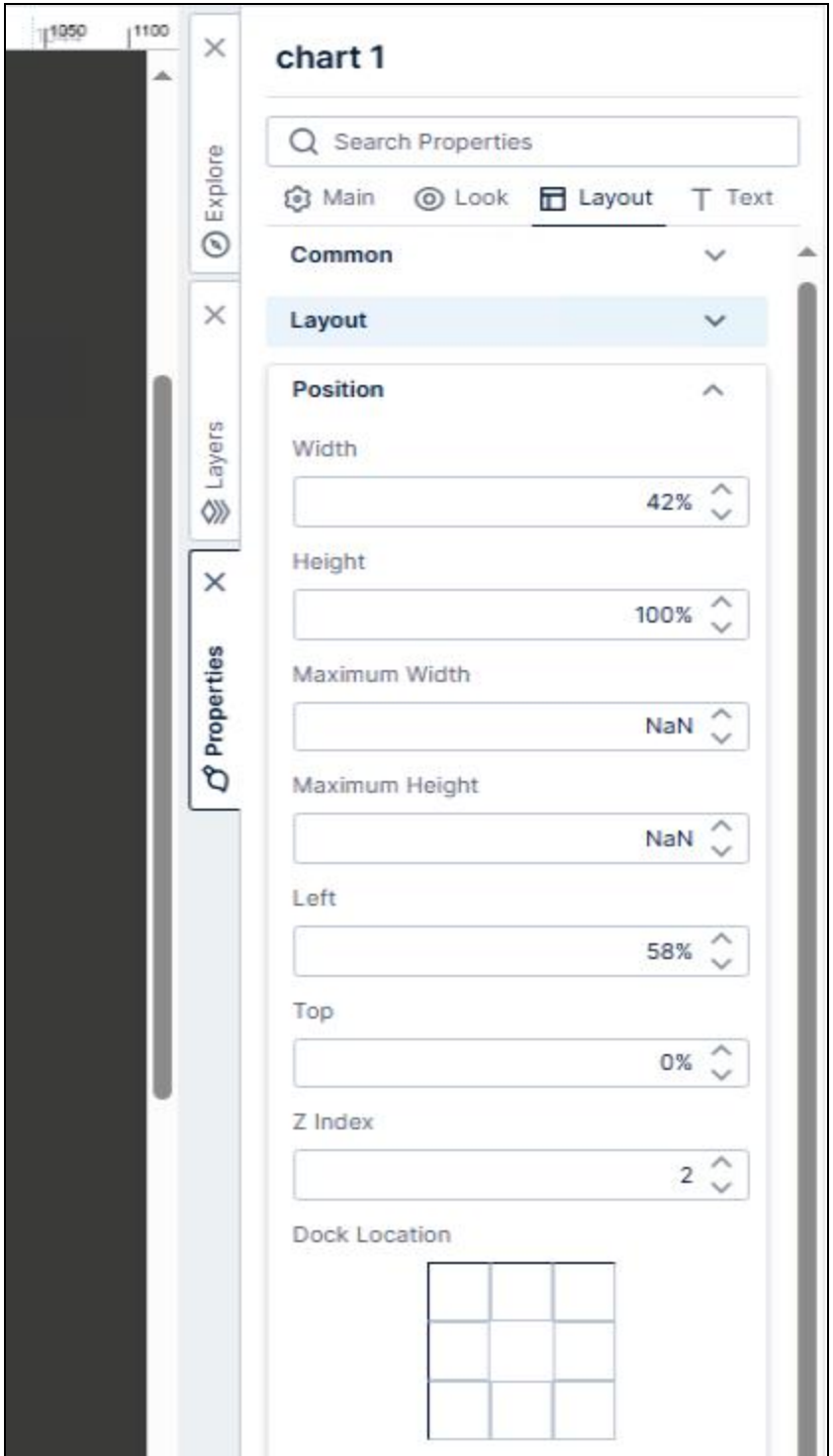
When dragging data, metric sets, or other content from the **Explore** window to add it to your dashboard, drag it onto the cell where you want it displayed.

To attach some existing content on the dashboard to a grid cell, drag it and drop it over a highlighted grid cell.



If it's the first element dragged into that cell, the element will automatically resize to fill the cell. You can then resize it if you want, for example to leave room above a chart to add a title or legend.

In the **Properties** window, you can see that the positioning of the element is now relative to its cell.

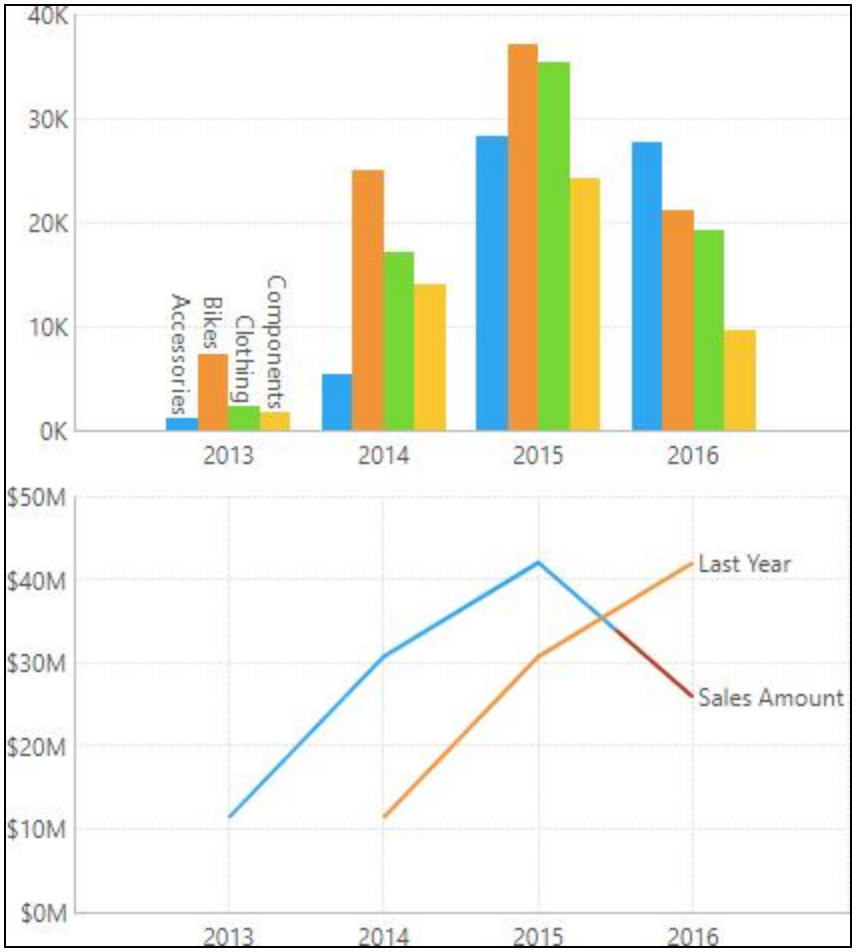

1050
1100
chart 1
Search Properties
Main
Look
Layout
Text
Common
Layout
Position
Width
42%
Height
100%
Maximum Width
NaN
Maximum Height
NaN
Left
58%
Top
0%
Z Index
2
Dock Location
Explore
Layers
Properties

The position coordinates are also percentages relative to the size of the cell by default, but you can change this using the **Position Kind** property to absolute pixels relative to the top-left corner of the cell, or dock the element to another side or corner instead.

Test Resize Behavior

Test the resize behavior by switching to View or **Sandbox View** in the toolbar and resizing your browser window.

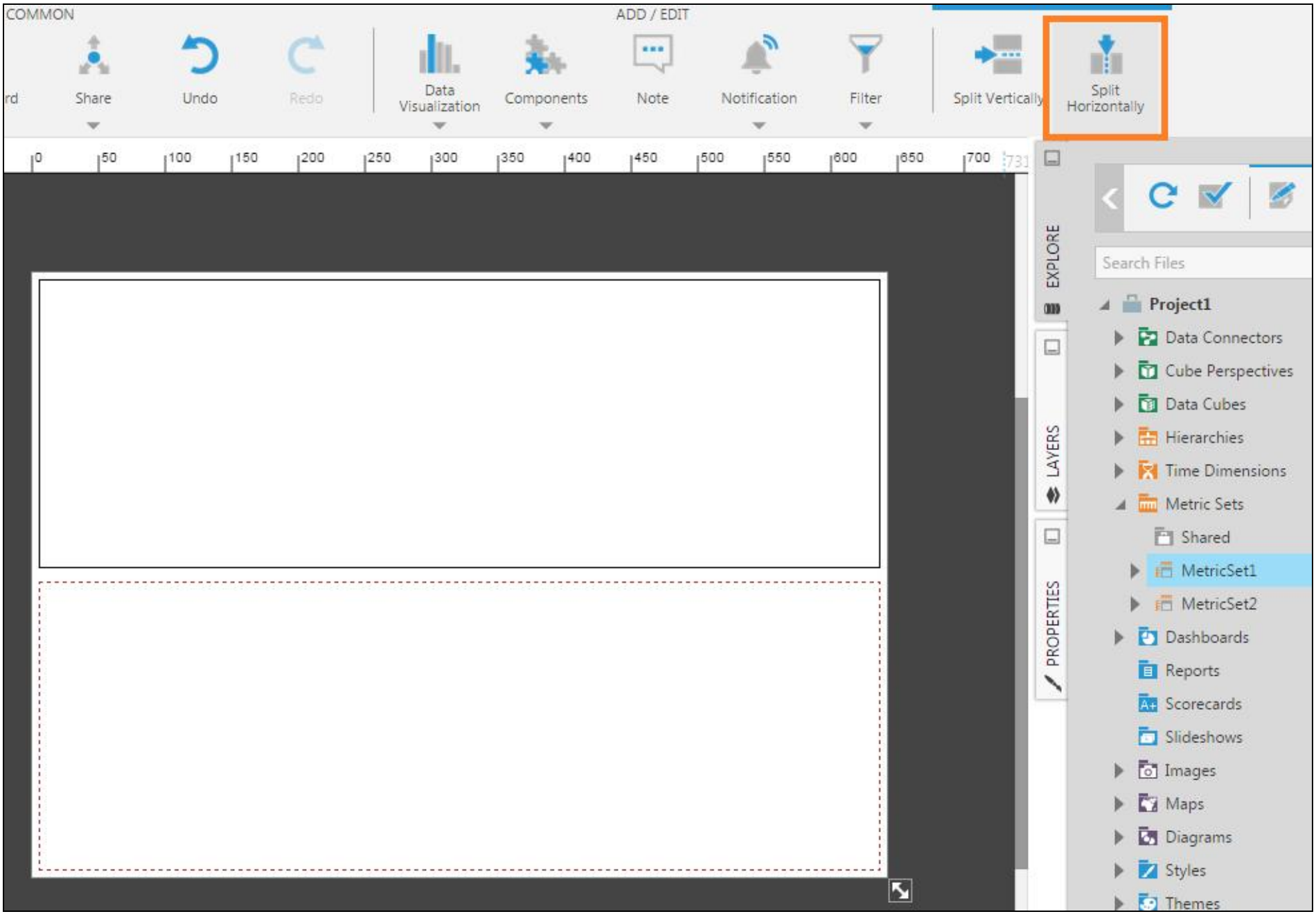
With the dashboard set to *Resize* in this example, observe that the dashboard can be manipulated into different aspect ratios, or made smaller, but visualizations such as charts are able to adjust their layouts intelligently to match.



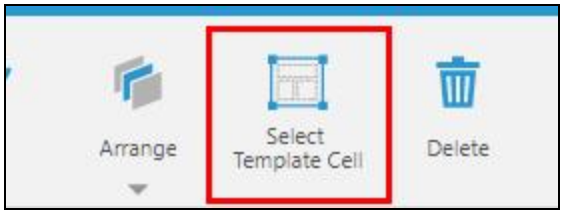
Merge and Split Cells

The template grid initially fills your dashboard with the same number of cells in each row (or column). For a more customized layout, you can split any cell horizontally or vertically into two, or merge two or more selected cells into a single cell.

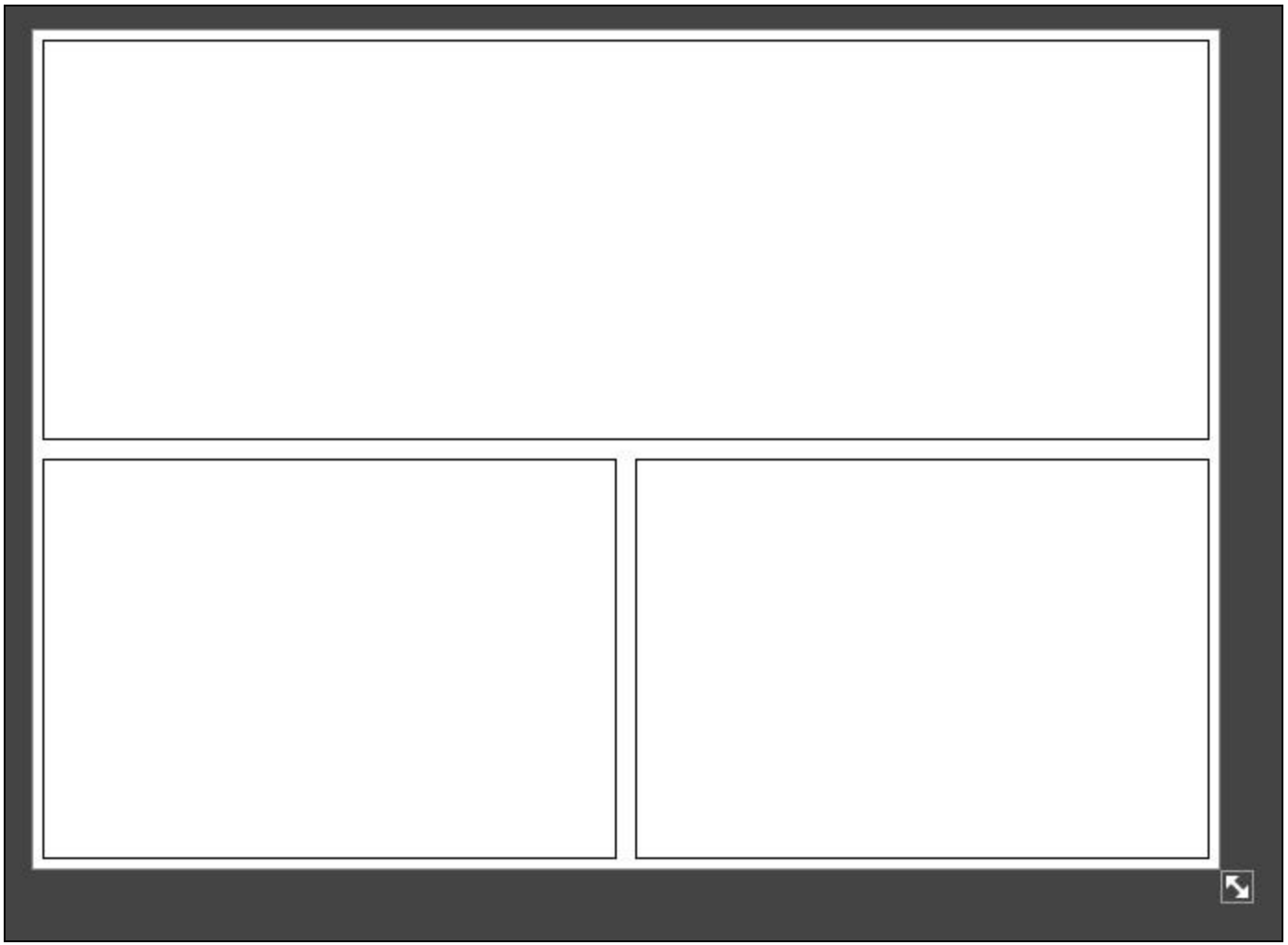
For example, in a dashboard with a 1x2 template grid as shown below, you can split the bottom cell into left and right parts by clicking to select the cell and then clicking **Split Horizontally** in the toolbar.


COMMON
ADD / EDIT
Share
Undo
Redo
Data Visualization
Components
Note
Notification
Filter
Split Vertically
Split Horizontally
EXPLORE
LAYERS
PROPERTIES
Search Files
Project1
Data Connectors
Cube Perspectives
Data Cubes
Hierarchies
Time Dimensions
Metric Sets
Shared
MetricSet1
MetricSet2
Dashboards
Reports
Scorecards
Slideshows
Images
Maps
Diagrams
Styles
Themes

If you've already completely filled a template grid cell with content, you may not be able to click directly on the cell but instead one of the elements you placed there. With one of the elements in the cell selected, you can still select the template grid cell by clicking **Select Template Cell** in the toolbar.

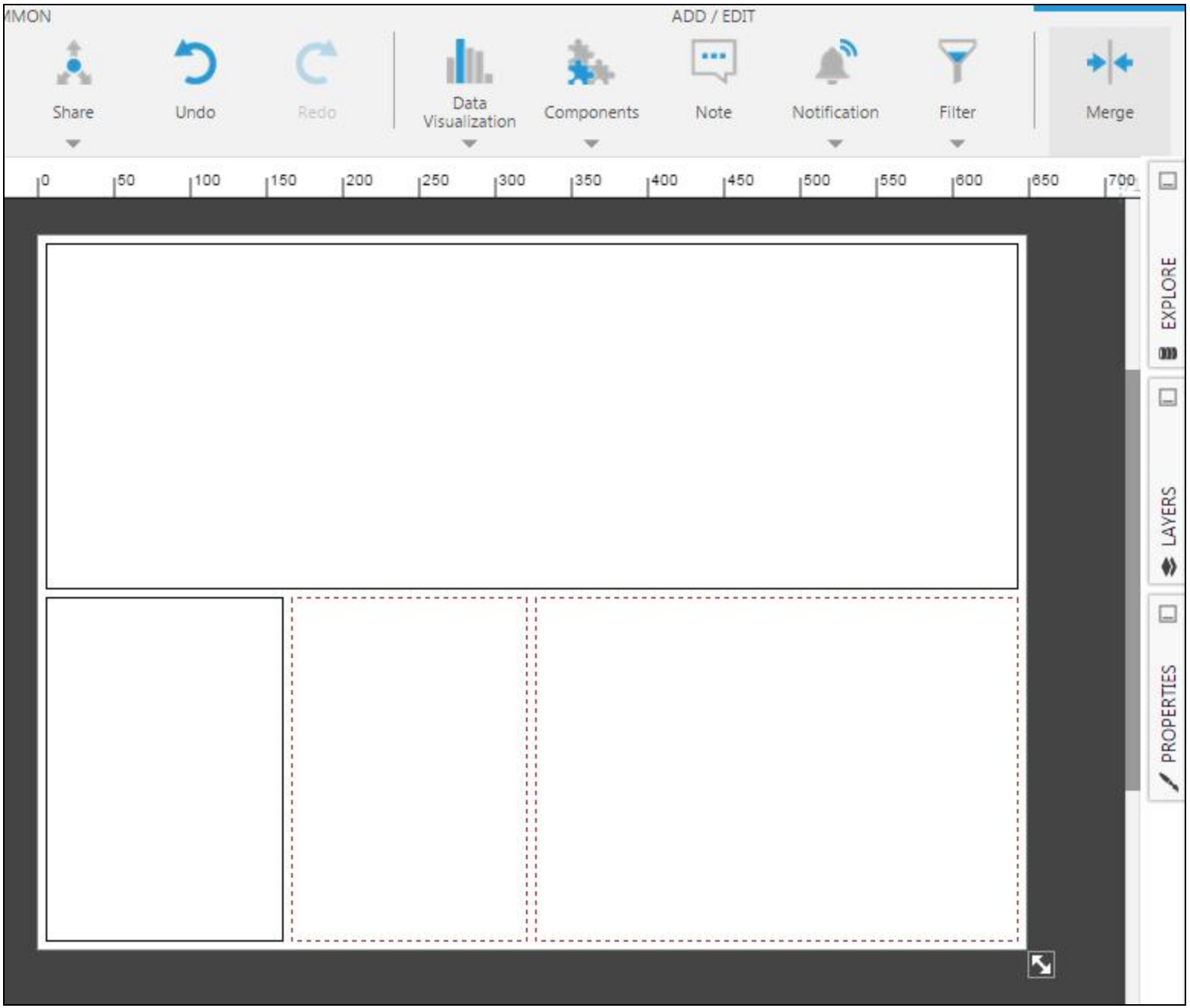


After splitting horizontally, the bottom cell is now divided into a left cell and a right cell. (Click **Split Vertically** to divide a cell into top and bottom cells.)

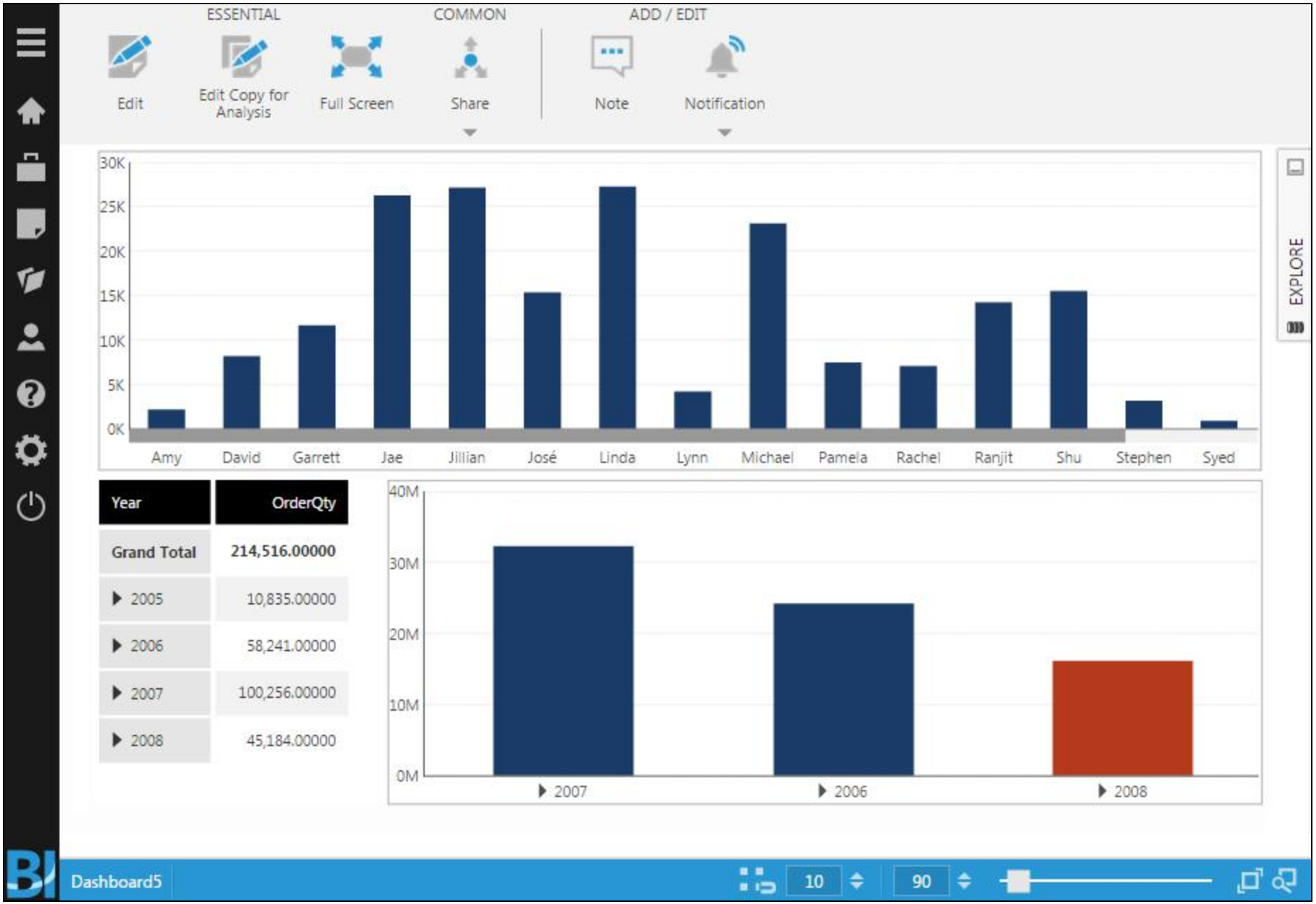


You can split multiple times: for example, select the bottom left cell and divide again by clicking **Split Horizontally**. There are now 3 cells in the bottom row of the template grid.

To merge two or more cells, including cells that may have previously been split, select one cell and then hold the Ctrl or Shift key while clicking on adjacent cells in the same row or column. Go to the toolbar and click **Merge**.


ADD / EDIT
Share
Undo
Redo
Data Visualization
Components
Note
Notification
Filter
Merge
0
50
100
150
200
250
300
350
400
450
500
550
600
650
700
EXPLORE
LAYERS
PROPERTIES

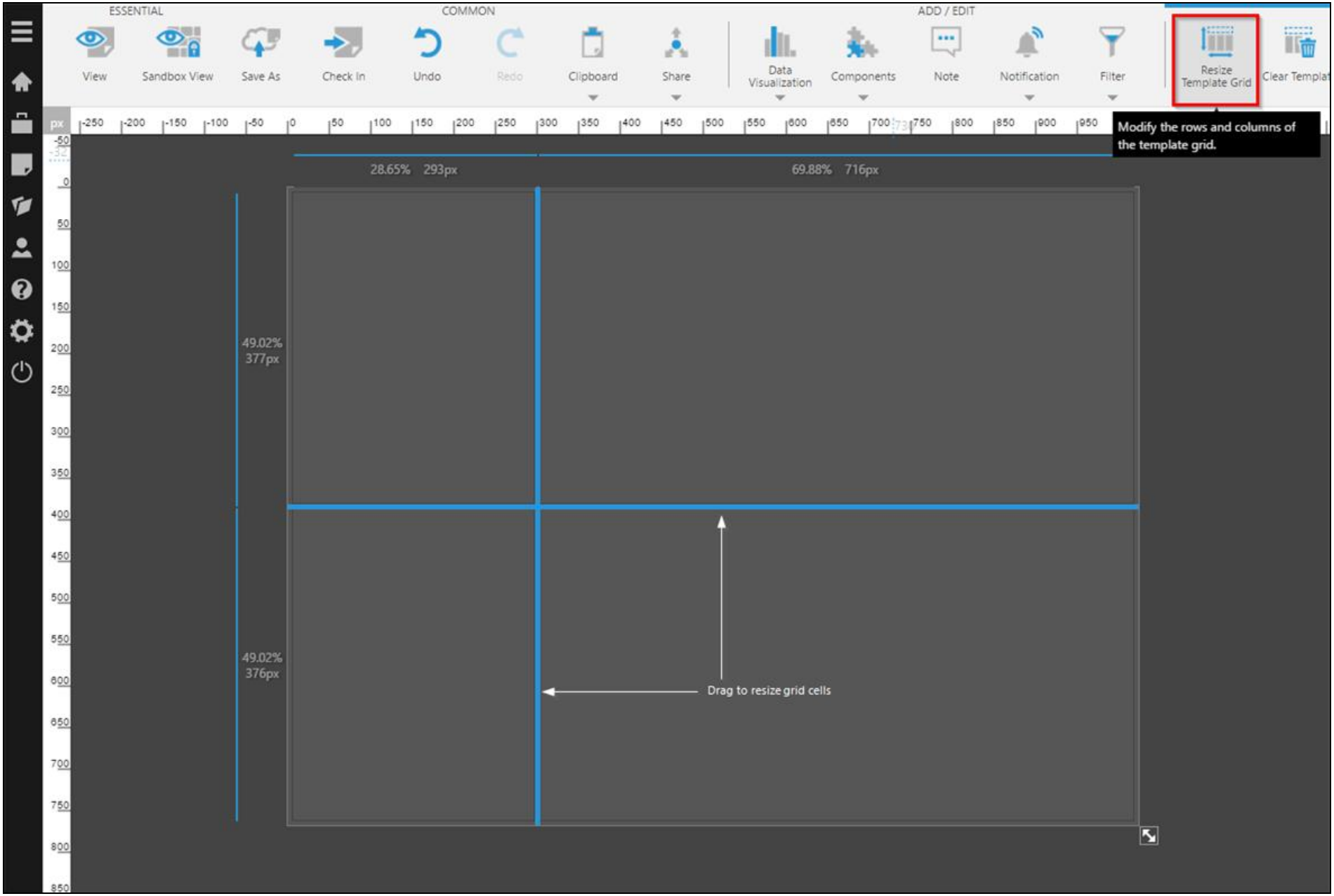
In our example shown above, the two bottom-right cells were merged and visualizations added to each of the three final cells, as shown below in view mode.



Resize Template Grid

The initial template grid that you define is uniform where each cell has the same size. You can adjust their sizes by clicking **Resize Template Grid** in the toolbar.

While resizing, the divisions between grid cells are outlined by thick blue bars that you can drag. You can also resize the grid by clicking on a percentage or pixel size, typing in a new value, then pressing the Enter key.


ESSENTIAL
COMMON
ADD / EDIT
View
Sandbox View
Save As
Check In
Undo
Redo
Clipboard
Share
Data Visualization
Components
Note
Notification
Filter
Resize Template Grid
Clear Templat
Modify the rows and columns of the template grid.
28.65% 293px
69.88% 716px
49.02% 377px
49.02% 376px
Drag to resize grid cells

Once you're done resizing the grid cells, click Resize Template Grid again in the toolbar.

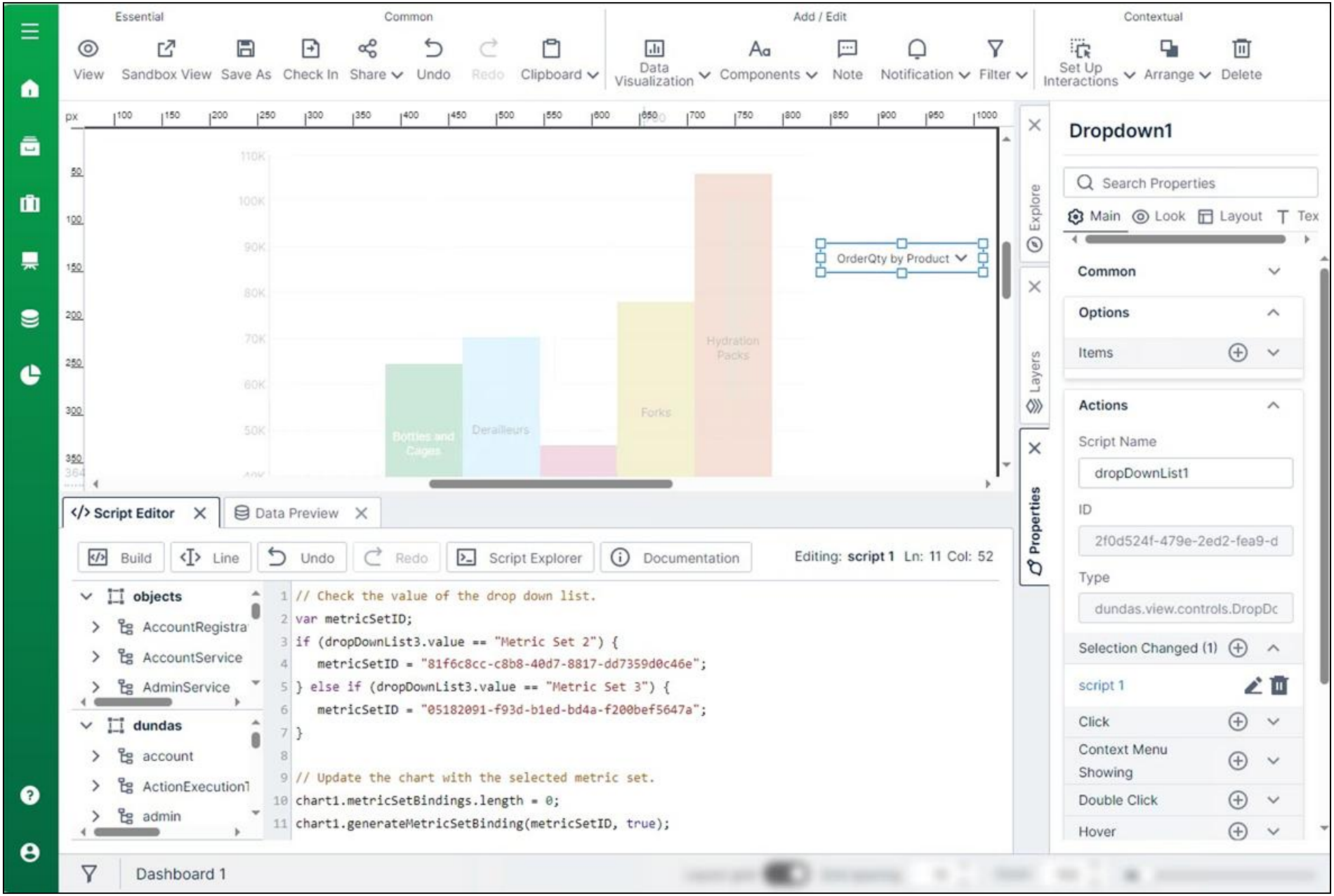
For more information, see:

- Video: Template Grid
- Video: Resize Mode
- Responsive Layout Dashboard Mode
- Layers and Groups
- List of Keyboard Shortcuts
- Using the Status Bar

Using the Script Editor

This applies to: *Managed Dashboards, Managed Reports*

When editing dashboards or other views, the script editor lets you add custom functionality. The scripting language that this built-in editor supports is **JavaScript**.


Essential
Common
Add / Edit
Contextual
View
Sandbox View
Save As
Check In
Share
Undo
Redo
Clipboard
Data Visualization
Components
Note
Notification
Filter
Set Up Interactions
Arrange
Delete
OrderQty by Product
Bottles and Cages
Derailleurs
Forks
Hydration Packs
Dropdown1
Search Properties
Main
Look
Layout
Common
Options
Items
Actions
Script Name
dropDownList1
ID
2f0d524f-479e-2ed2-fea9-d
Type
dundas.view.controls.DropDc
Selection Changed (1)
script 1
Click
Context Menu Showing
Double Click
Hover
Explore
Layers
Properties
Script Editor
Data Preview
Build
Line
Undo
Redo
Script Explorer
Documentation
Editing: script 1 Ln: 11 Col: 52
objects
AccountRegistra
AccountService
AdminService
dundas
account
ActionExecution1
admin
// Check the value of the drop down list.
var metricSetID;
if (dropDownList3.value == "Metric Set 2") {
metricSetID = "81f6c8cc-c8b8-40d7-8817-dd7359d0c46e";
} else if (dropDownList3.value == "Metric Set 3") {
metricSetID = "05182091-f93d-b1ed-bd4a-f200bef5647a";
}
// Update the chart with the selected metric set.
chart1.metricSetBindings.length = 0;
chart1.generateMetricSetBinding(metricSetID, true);
Dashboard 1

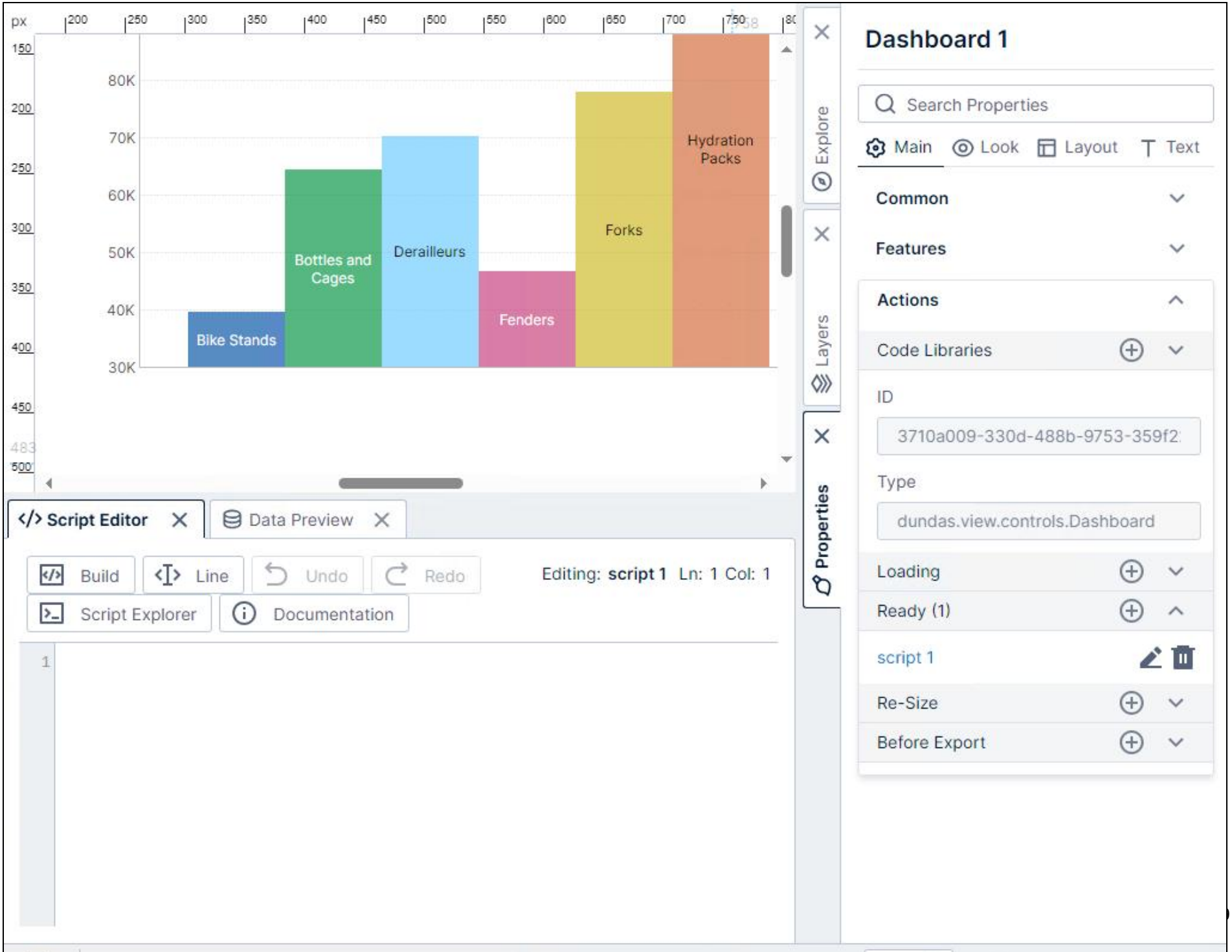
The Script Editor window is also available when editing data cubes and offers a subset of the following features for writing SQL or MDX, Python or R, or DundasScript.

> **Note:** You can also use code libraries to reuse scripts between multiple dashboards and other views. Administrators also have the ability to set up custom JavaScript to run at the site level.

Opening the Script Editor

Your account needs to have a developer seat to edit scripts. Open a script for editing by going to the **Properties** window for a dashboard or other view, or when some element on its canvas is selected.

Scroll down to the **Actions** section of the **Main** tab and expand the event you want to handle. Select the **+** button to add a new script or select an existing script.


px
80K
70K
60K
50K
40K
30K
Bike Stands
Bottles and Cages
Derailleurs
Fenders
Forks
Hydration Packs
Explore
Layers
Properties
Dashboard 1
Search Properties
Main
Look
Layout
Text
Common
Features
Actions
Code Libraries
ID
3710a009-330d-488b-9753-359f2
Type
dundas.view.controls.Dashboard
Loading
Ready (1)
script 1
Re-Size
Before Export
Script Editor
Data Preview
Build
Line
Undo
Redo
Editing: script 1 Ln: 1 Col: 1
Script Explorer
Documentation

The Script Editor window opens and is docked along the bottom of the canvas by default.

You can also find existing script listed in the Layers window and select to open them from there.

Elements of The Script Editor

The script editor consists of these main elements:

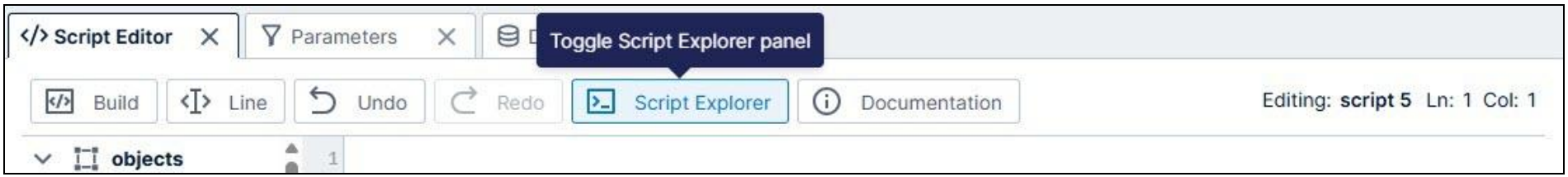
- Toolbar
- Script Explorer panel
- Editor area

Note: You can drag the splitters between the different areas and panels to resize them.

Toolbar

The toolbar in the script editor lets you:

- **Build** the script to check if it is valid
- Go to a specific **line** number in the script
- **Undo** or **redo** your changes in the editor
- Show or hide the **Script Explorer** panel
- See the **name** of your script
- See what line number and **column** you are at in the editor (i.e., cursor position)

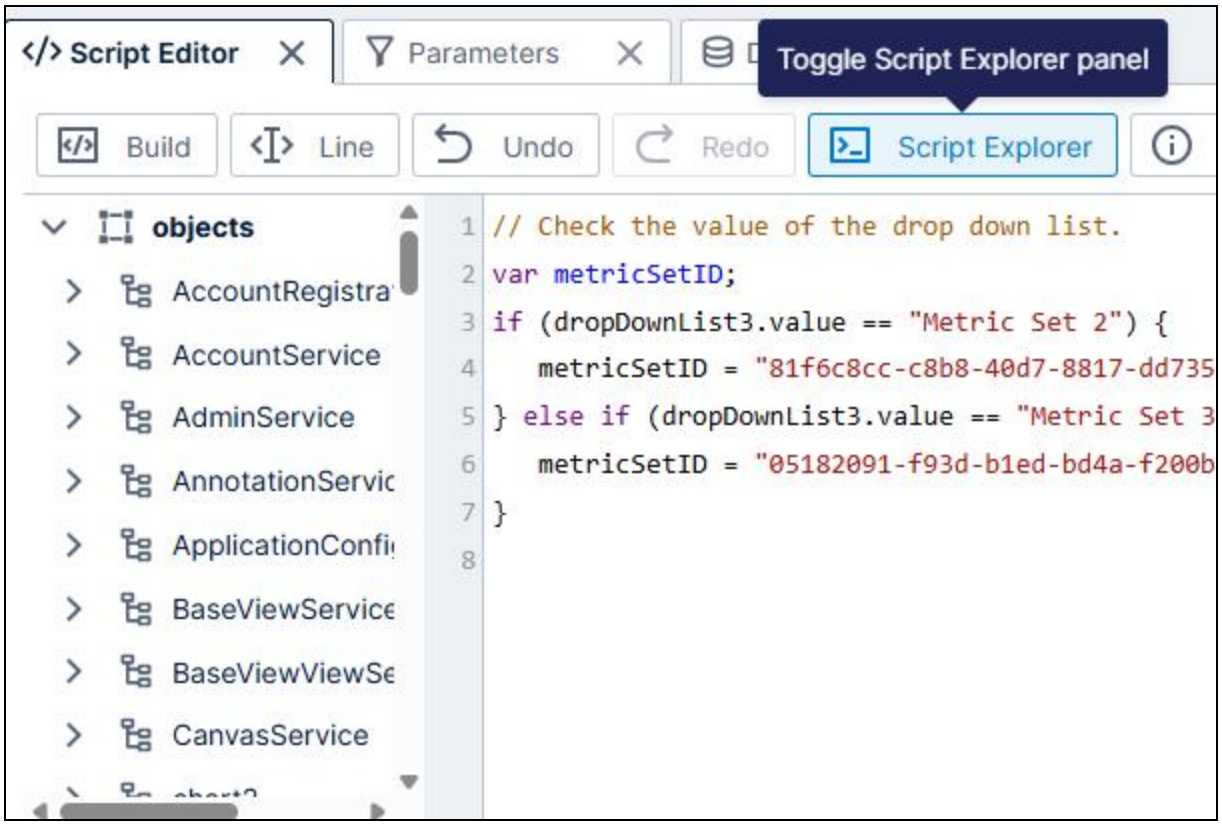


Script Explorer

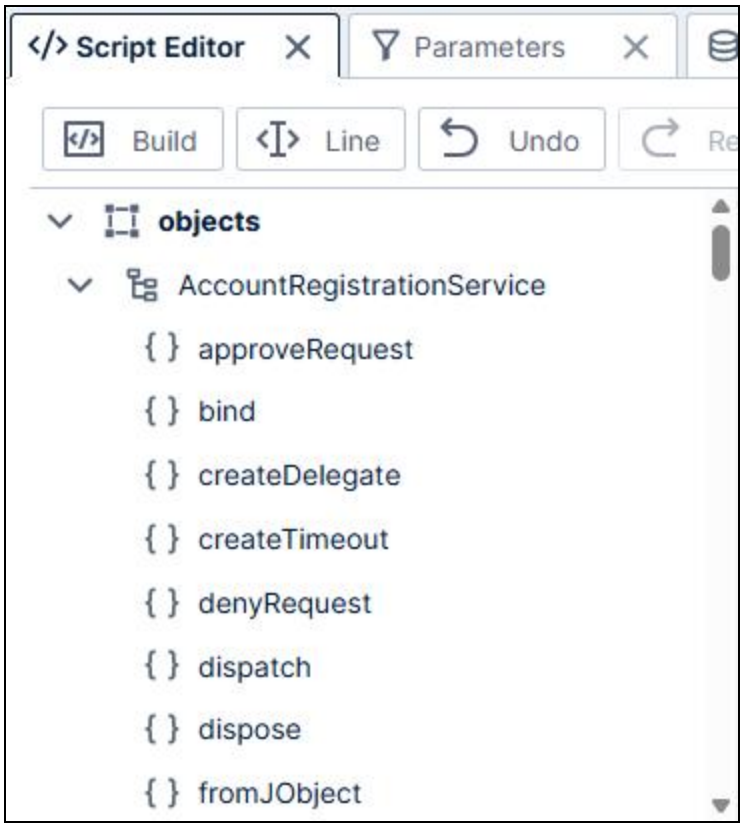
The script explorer lets you navigate the properties and methods of objects on the current view as well as the data types available in the JavaScript API.

To show or hide the Script Explorer panel, select the **Script Explorer** button in the editor toolbar. The Script Explorer panel has two parts: an Object Explorer and a Type Explorer.

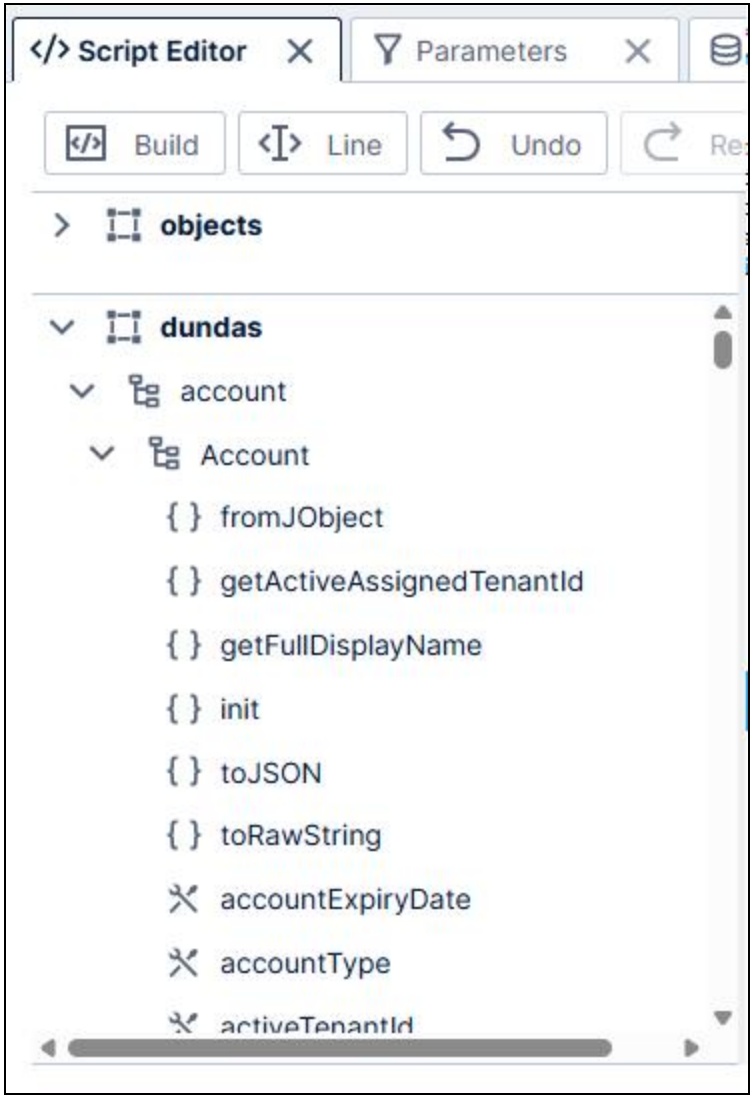
The **Object Explorer** lists all of the currently-available objects on your view (in alphabetical order), which can be manipulated by the statements you write in the editor.



Expand the triangles on the left to see what properties and methods are available on each object.

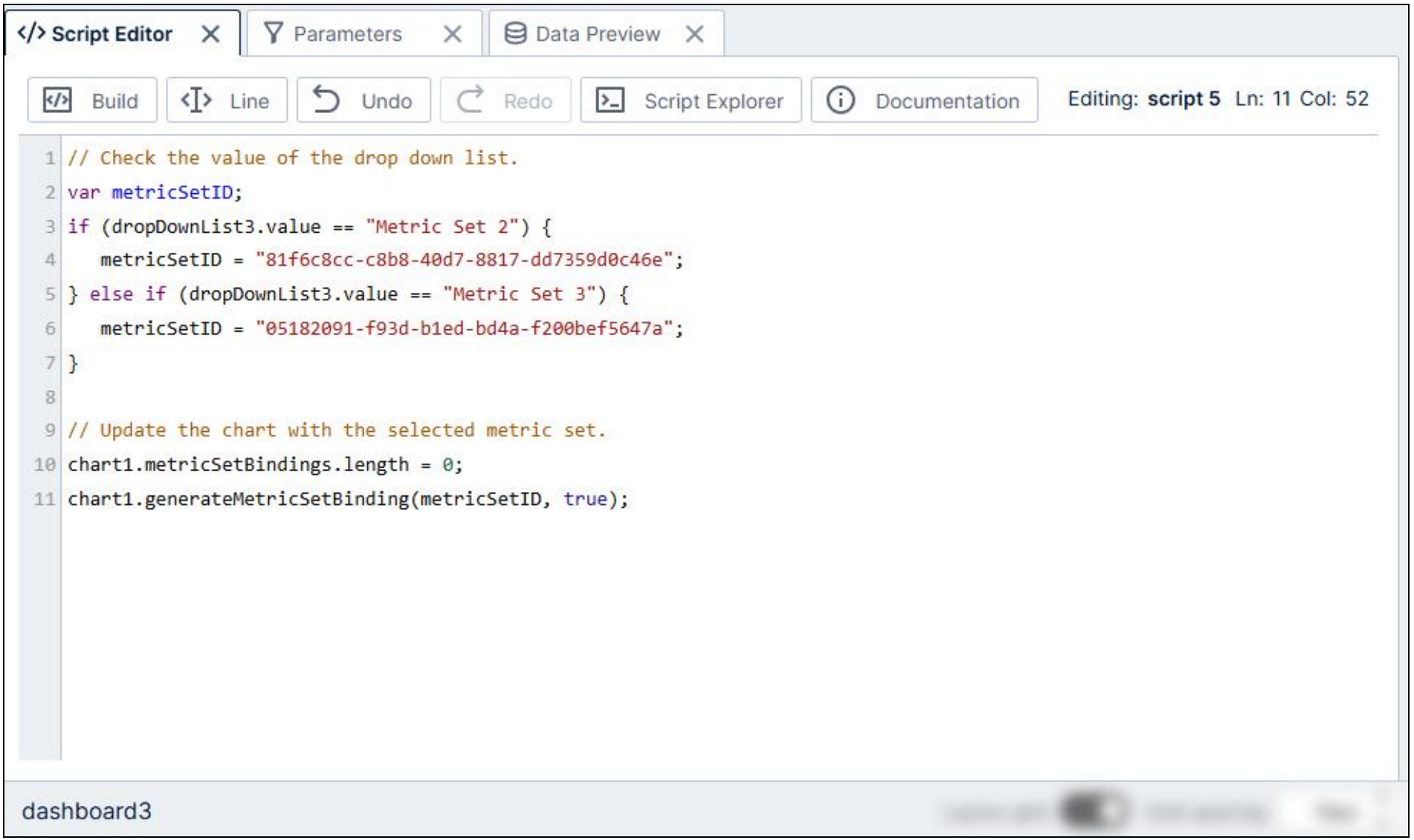


The **Type Explorer** lists all of the data types that are supported for scripting (similar to an API reference).



Editor

The editor is the large text area where you compose and edit your script.

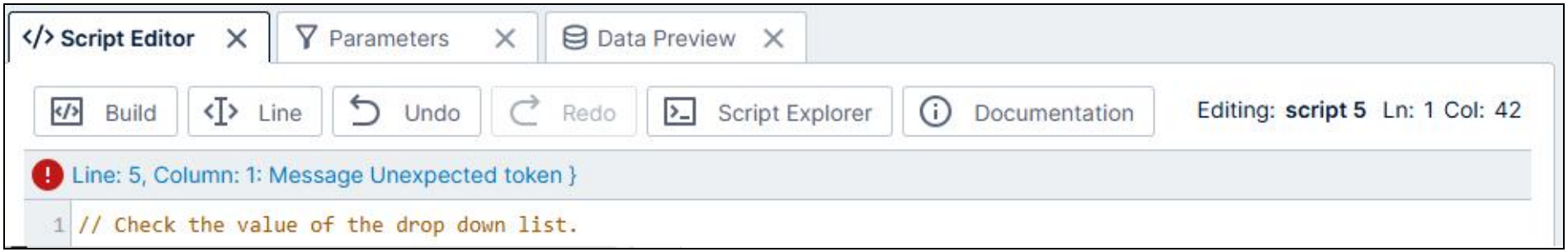


Functionality

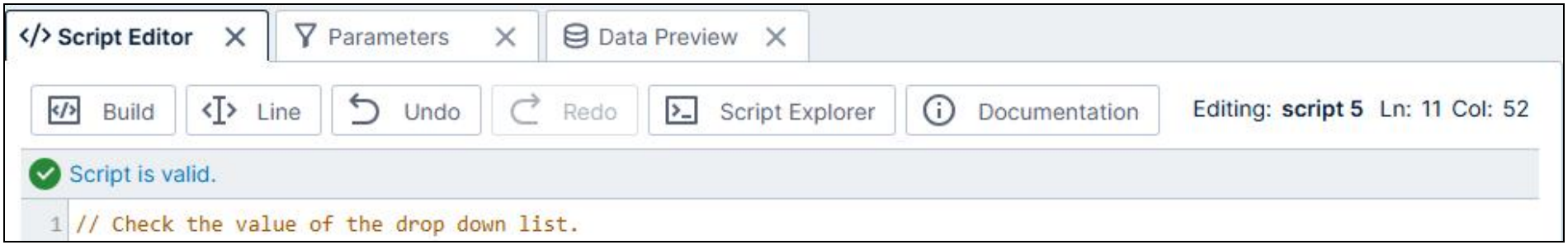
Build Script

Select **Build** in the editor toolbar to build your script and check for syntax errors.

If an error is detected, select the error link to move the editor cursor to the location of the error.



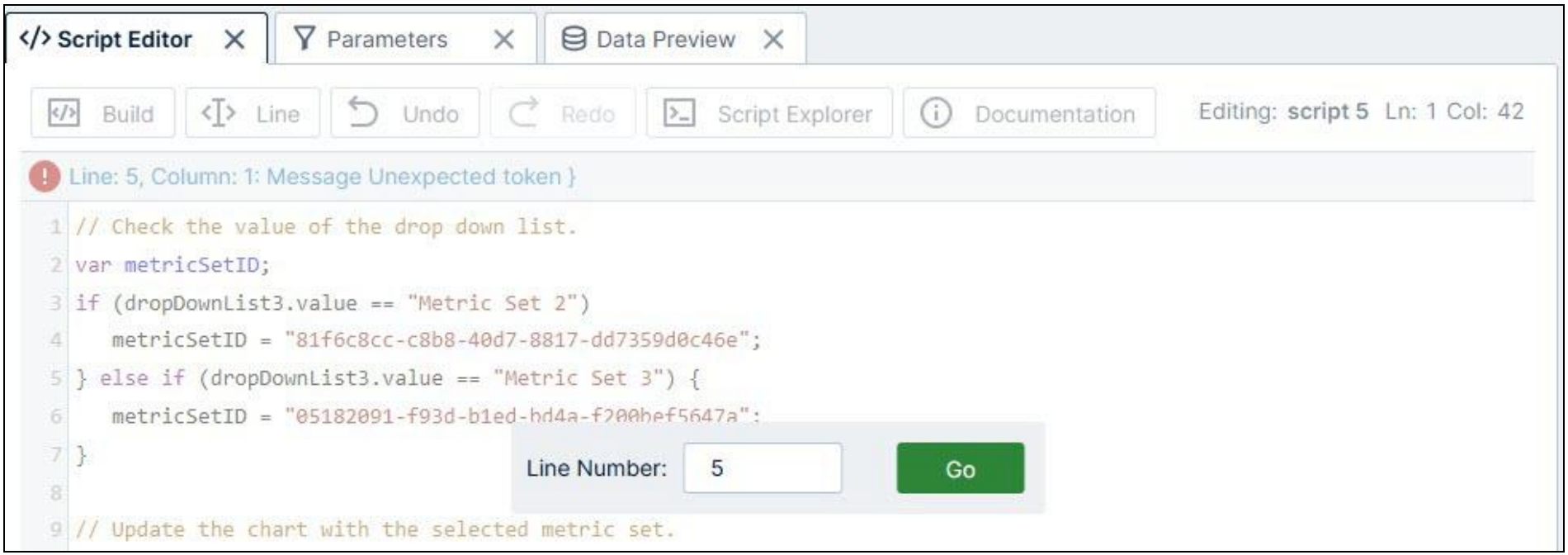
If there are no errors, you'll see a message indicating the script is valid.



Go to Line Number

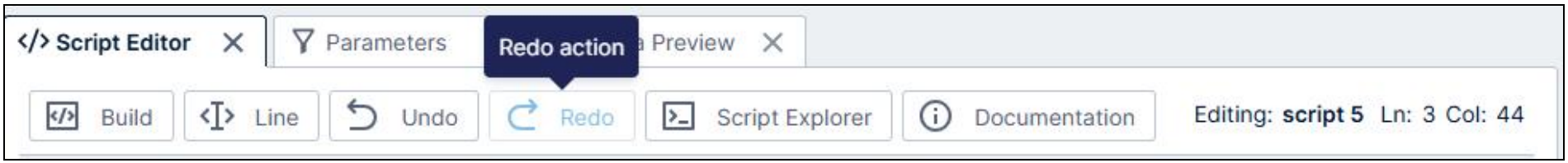
To move the cursor to a specific line number in your script, select **Line** in the editor toolbar.

Enter the line number and then select **Go**.



Undo and Redo

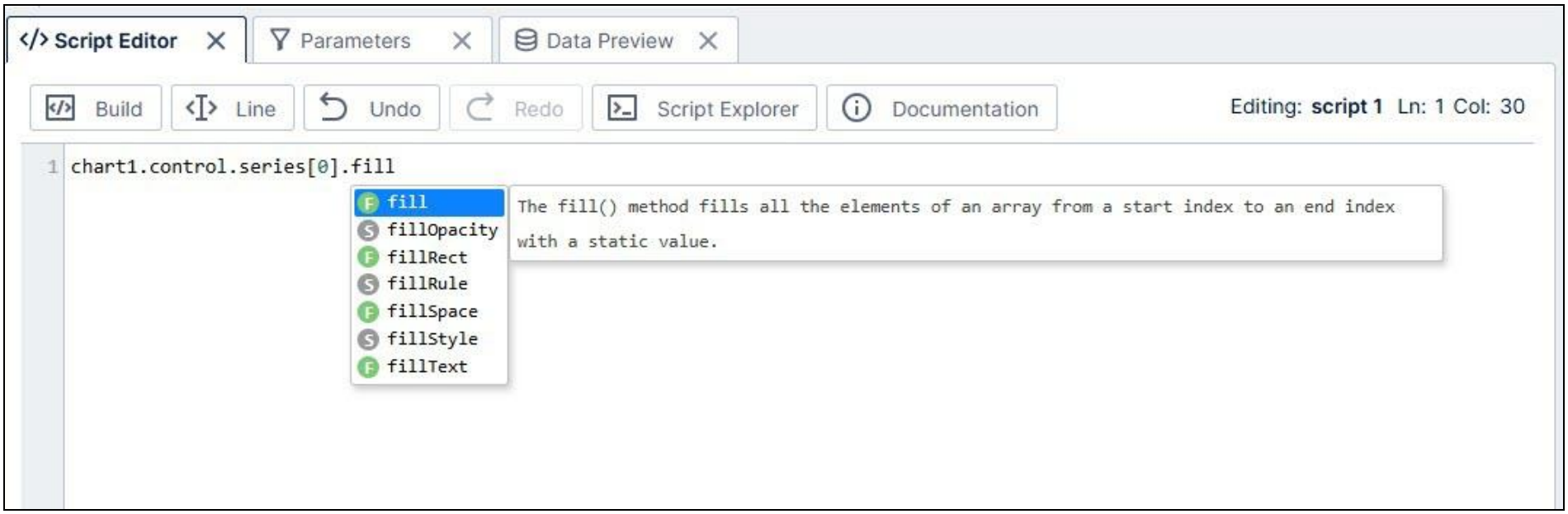
Use the **Undo** and **Redo** buttons to revert or re-apply your actions in the editor area.

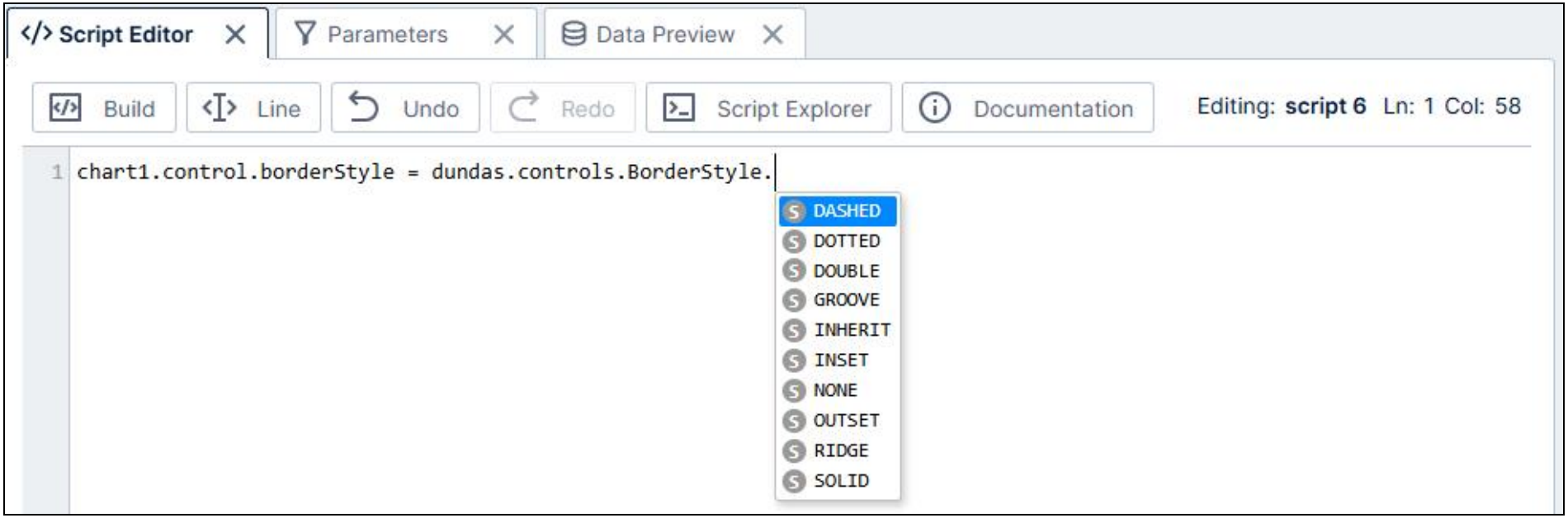


Auto-Complete

The script editor offers auto-complete (or code completion) support for: ECMA5, ECMA6, Browser, jQuery, and the JavaScript API. As you type in the editor area, auto-completion popups appear which help you to choose the desired property, method, function, or other construct to include. For example:

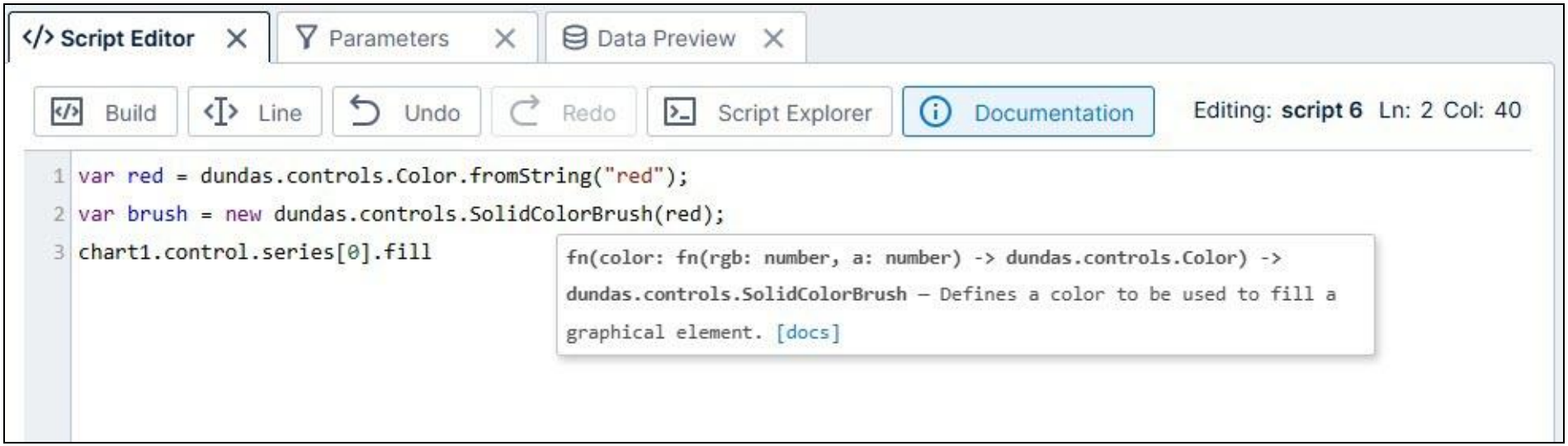
List properties and methods - Type the period character after any variable reference, property name, or array element in the editor. You will immediately see an auto-complete popup listing the properties and methods on that object, along with the corresponding API description. Use the arrow keys to cycle through the available choices, or select an item with your mouse. When you've found the item you want to insert into your script, press **ENTER** or double-click the item.


Script Editor
Parameters
Data Preview
Build
Line
Undo
Redo
Script Explorer
Documentation
Editing: script 1 Ln: 1 Col: 30
1 chart1.control.series[0].fill
fill
fillOpacity
fillRect
fillRule
fillSpace
fillStyle
fillText
The fill() method fills all the elements of an array from a start index to an end index with a static value.

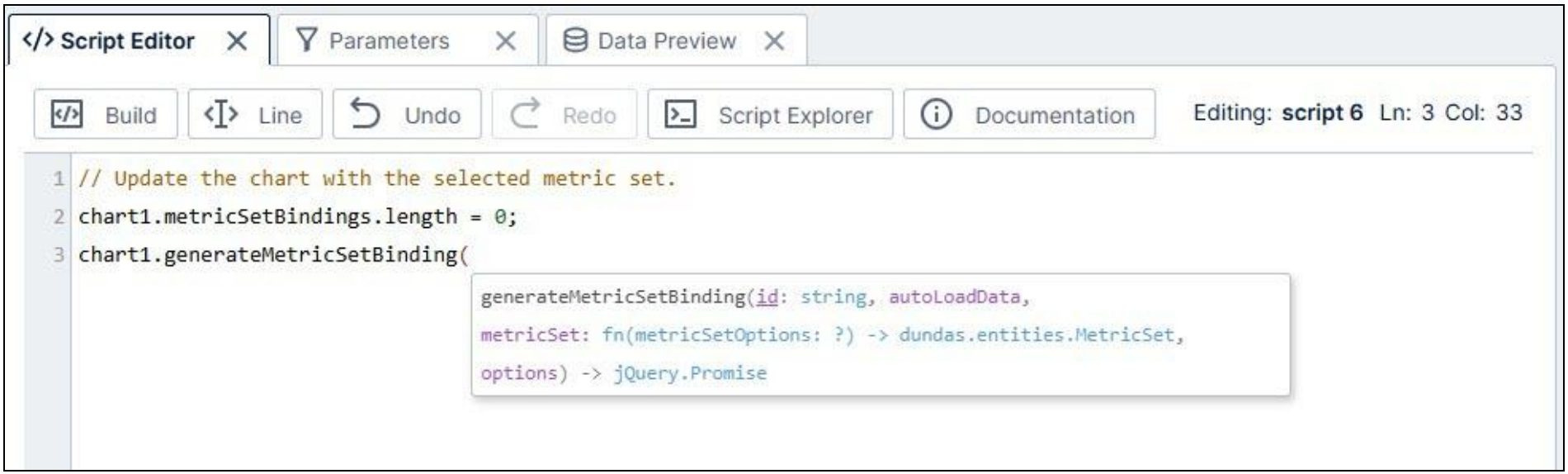

Script Editor
Parameters
Data Preview
Build
Line
Undo
Redo
Script Explorer
Documentation
Editing: script 6 Ln: 1 Col: 58
1 chart1.control.borderStyle = dundas.controls.BorderStyle.
DASHED
DOTTED
DOUBLE
GROOVE
INHERIT
INSET
NONE
OUTSET
RIDGE
SOLID

Show documentation - Wherever you are typing, select **Documentation** in the editor toolbar or press **CTRL+O** to display the documentation for the current property, object, or argument in a popup, including its type.

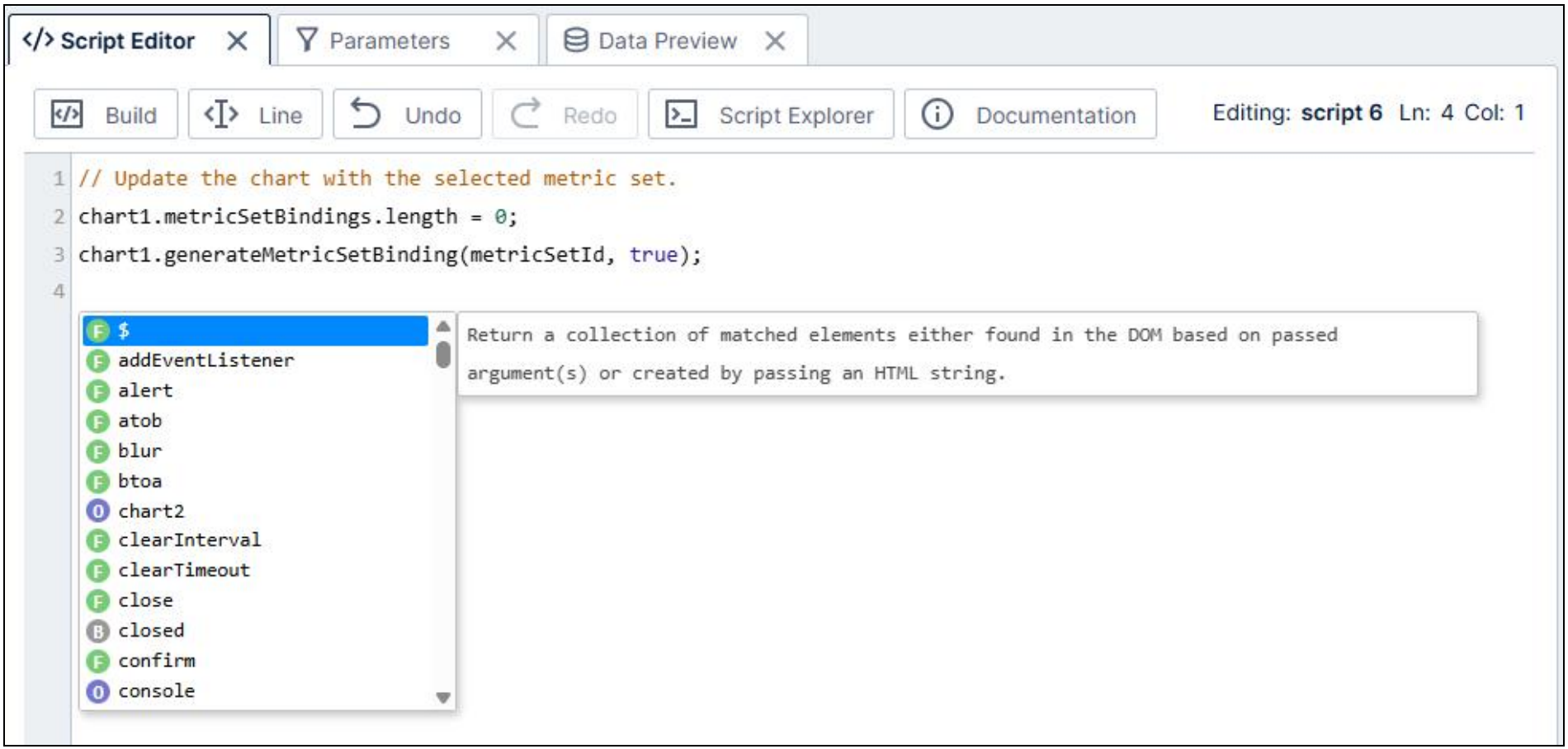
Select the docs link to bring up the related page in the JavaScript API reference.



Function/method parameters - In the case of a method or function, once you've added it to your script by pressing ENTER, type the open bracket '(' to see the function/method signature. The first required parameter will be underlined. As you type each parameter, the popup will underline the next parameter that is required.



All available types - While on a new line, press **CTRL+SPACE** to see the auto-complete popup showing all available types.



The following table describes what the icons in the auto-complete popup means:

Icon	JS Type
A	Array
F	Function
B	Boolean
1	Number
S	String
O	Object
?	Unknown

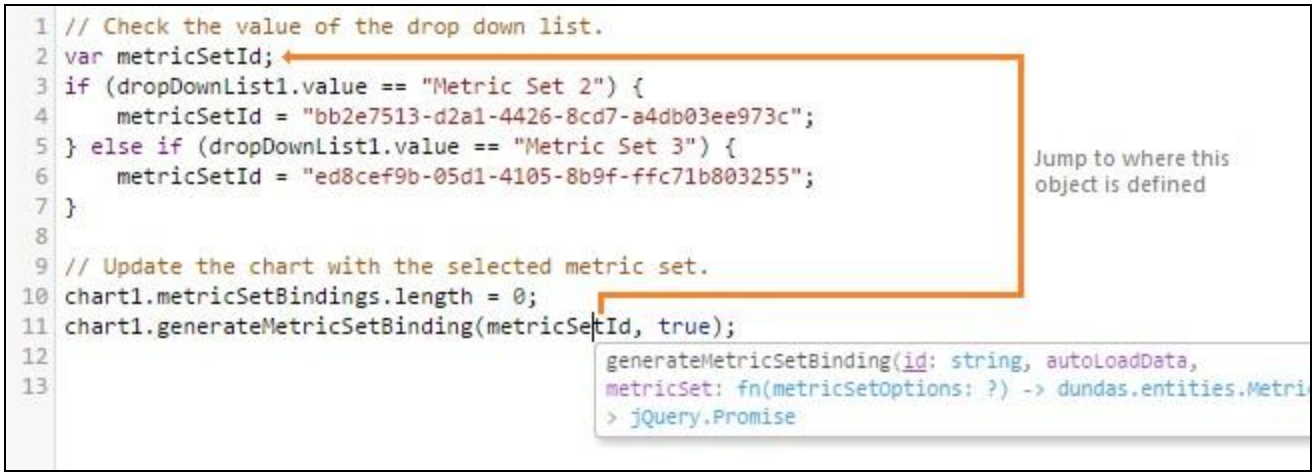
Highlight Instances of Current Object for Renaming

To rename all instances of an object, move your cursor over one of the instances or select it.

Press **CTRL+**. to select all instances of the object, and then type the new name.

Jump to Object Declaration

While your cursor is over an object, press **ALT+**. to highlight the object where it is defined (i.e., its **var** statement).

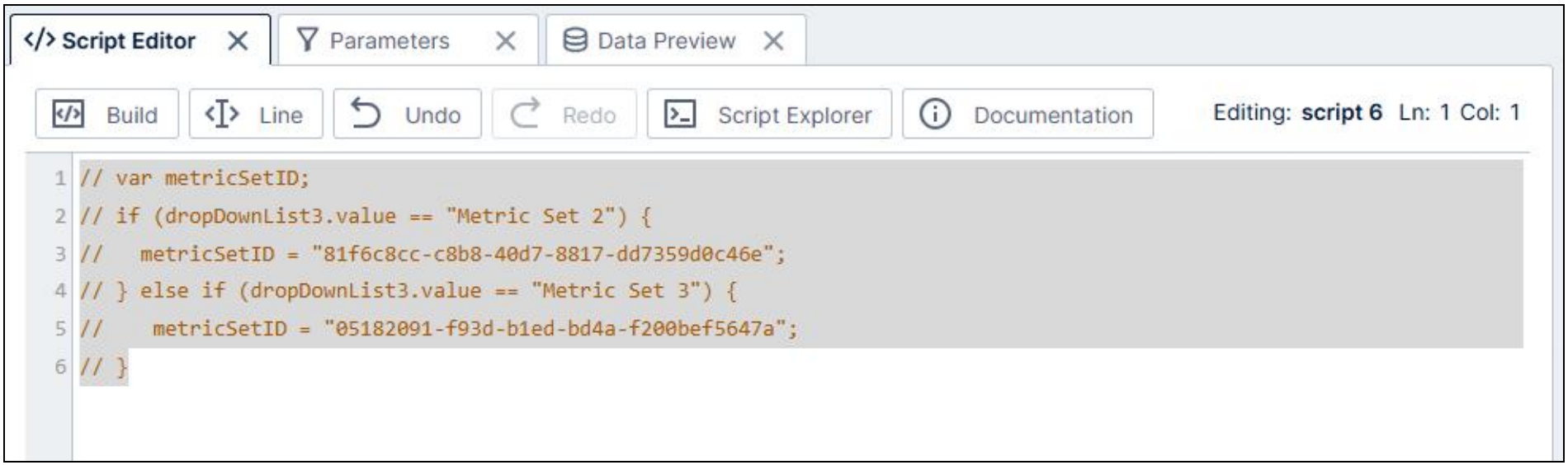


Comment the Current Line or Selected Lines

Press **CTRL+/** to comment the current line (where the cursor is located) or multiple selected lines in the editor.

If the current line or selected lines are already commented, this key sequence will uncomment the line or selected lines.

On a Mac computer, using **CTRL+/** or **CMD+/** will work the same to comment or uncomment code.



Running Scripts

Script actions that you add to events will only run in view mode. It's good practice to select **Sandbox View** in the toolbar to test your scripts in a separate browser tab with the toolbar hidden, because you usually don't want the script's results to be saved and then loaded next time for everyone.

Note that events such as **Loading** or **Ready** on a dashboard will only fire when that dashboard is opened directly into view mode. You can re-open that dashboard, reload the page, or use the Sandbox View option.

For more information, see:

- JavaScript API
- Using Code Libraries
- Script Library
- Using Dockable Windows
- Writing Scripts Using Browser Developer Tools

- White Labeling the Application
- w3schools.com - JavaScript Tutorial

Context Menu Not Working

This applies to: *Managed Dashboards, Managed Reports*

The context menu may not show on right-click for certain devices and browsers with touch screen capabilities. Symphony will automatically switch to **Use Touch Events** when such capabilities are detected.

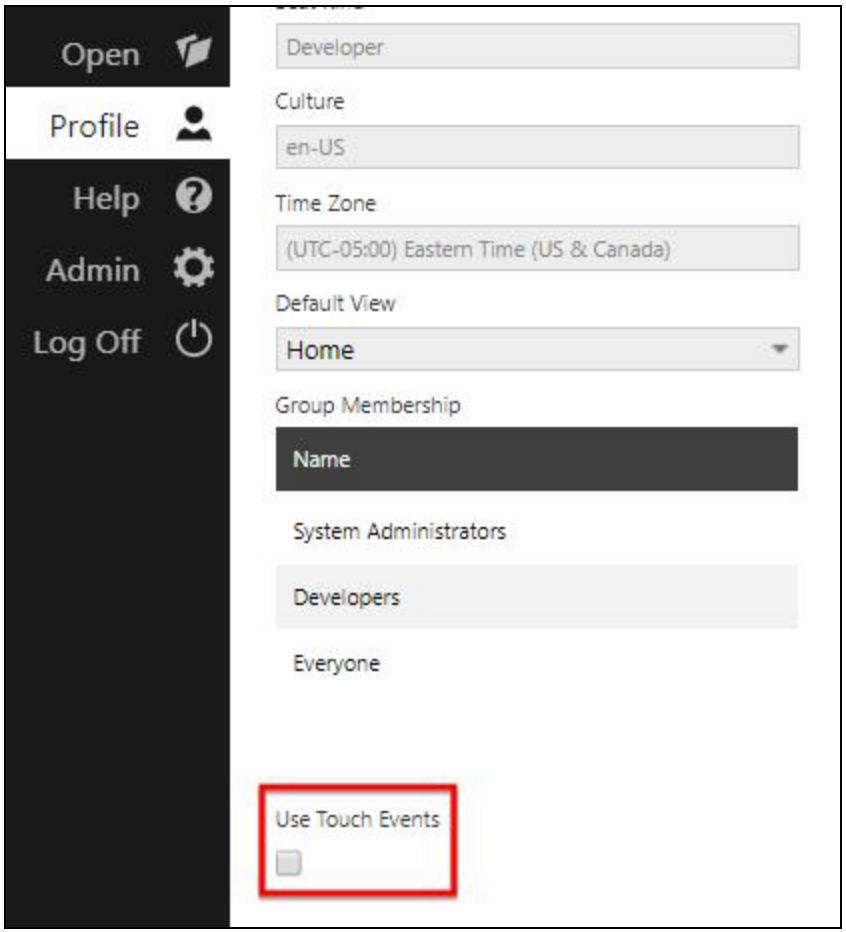
To open the context menu when **Use Touch Events** is enabled, long-tap on your touch screen or long- press the left mouse button.

Enable or Disable Touch Events

You can switch between touch events such as long-tap and mouse events such as right-click by changing the **Use Touch Events** setting.

Select **Profile** from the main menu.

Select or clear the **Use Touch Events** checkbox.



Click the **OK** button in the dialog that appears to reload the browser.

For more information, see:

- Edit Your Profile and Change Your Password

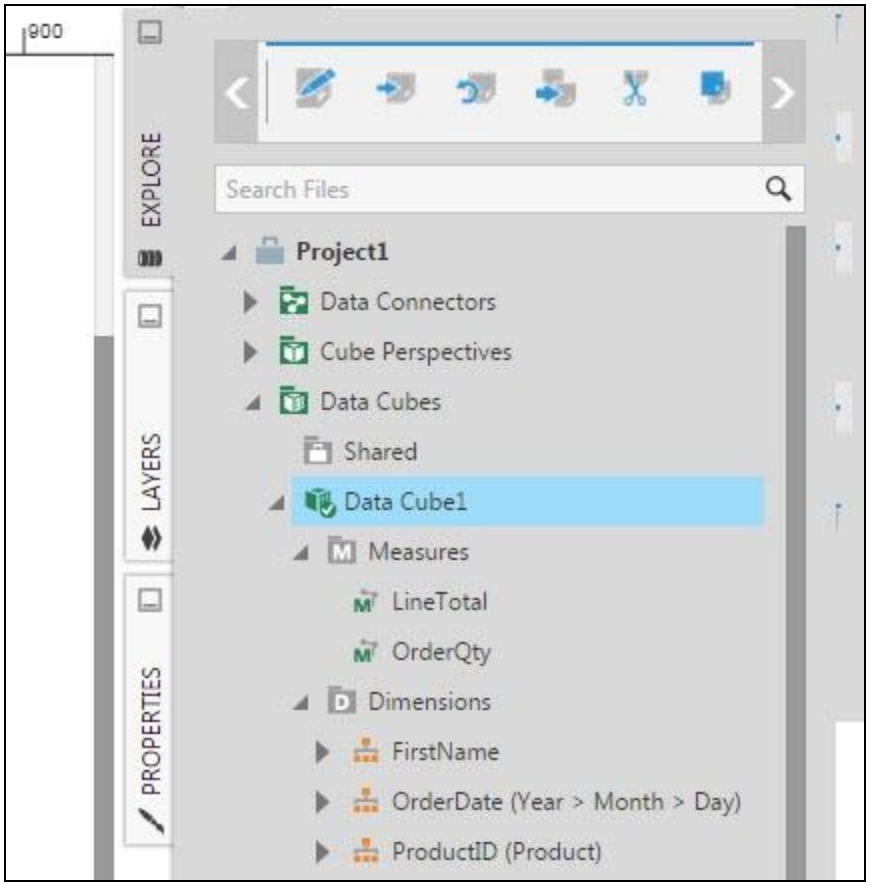
Adding a Dynamic Element Filter

This applies to: *Managed Dashboards, Managed Reports*

A dynamic element filter allows viewers to choose the measure or hierarchy they want to see in a visualization via a drop down list, without any scripting.

Setup

For this example, a data cube will be used which exposes two measures (LineTotal, OrderQty) and three hierarchies (FirstName, OrderDate, Product). You can build a similar data cube yourself by following the steps in Create a New Data Cube.

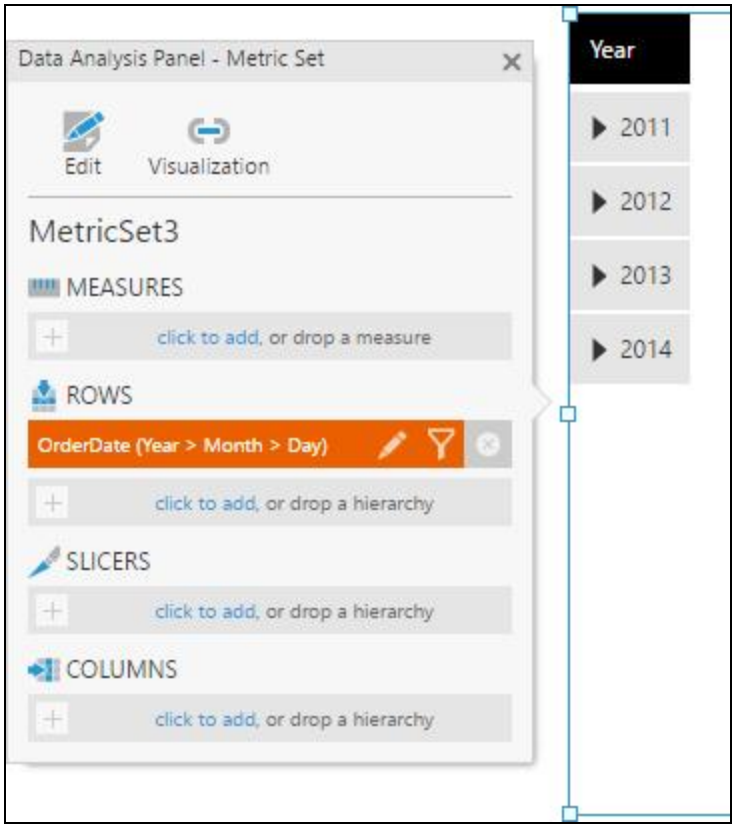


Dynamic Measure

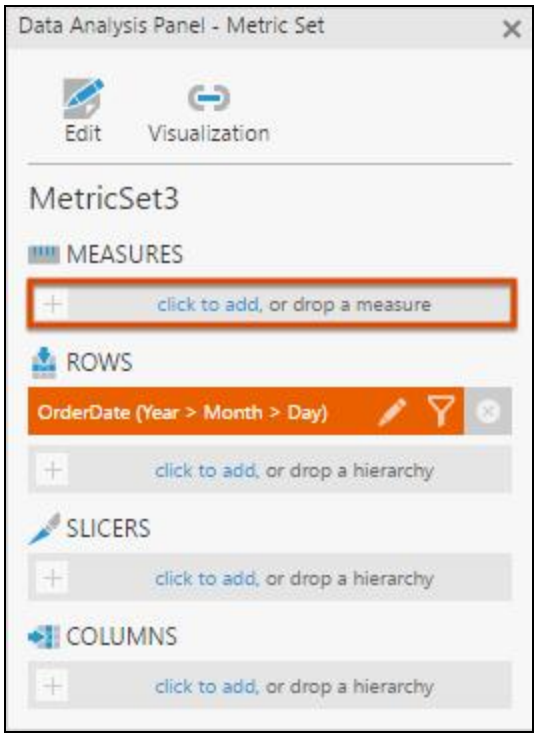
Add the Dynamic Element

In this example, we first create a new dashboard for this example from the main menu using the **Blank** template.

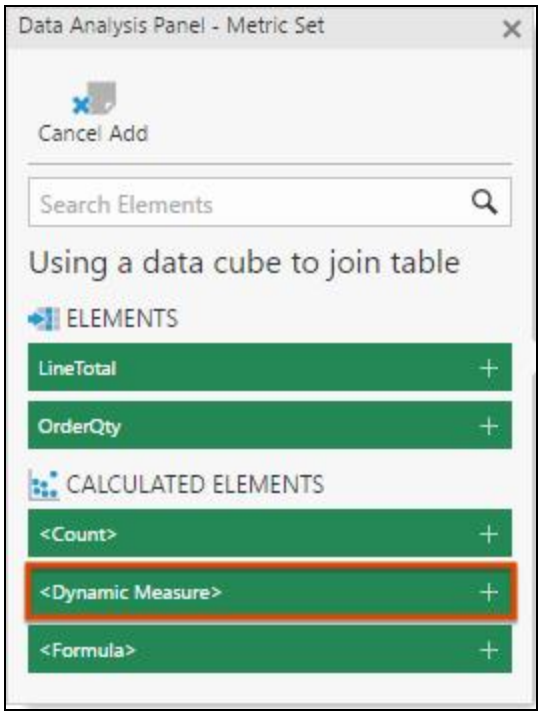
We drag the **OrderDate** hierarchy from **Explore** to the dashboard canvas. It will appear as a table visualization with the Data Analysis Panel beside it.



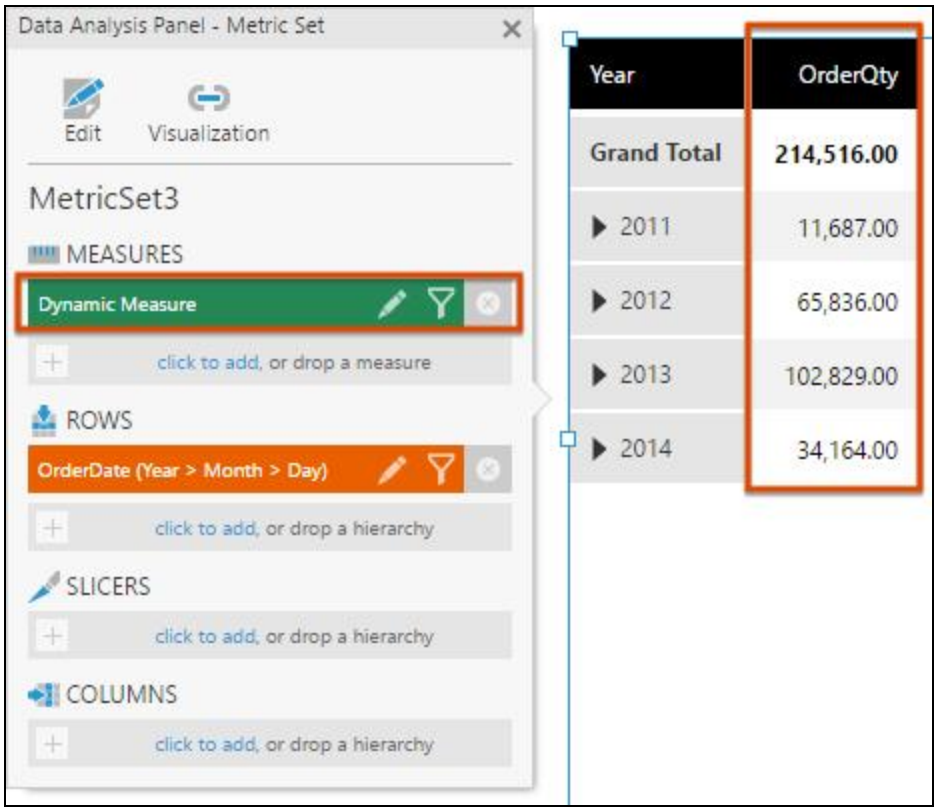
In the Data Analysis Panel, click the **click to add** link under **Measures**.



In the list of available measures to add, go to the **Calculated Elements** section and click **<Dynamic Measure>**.

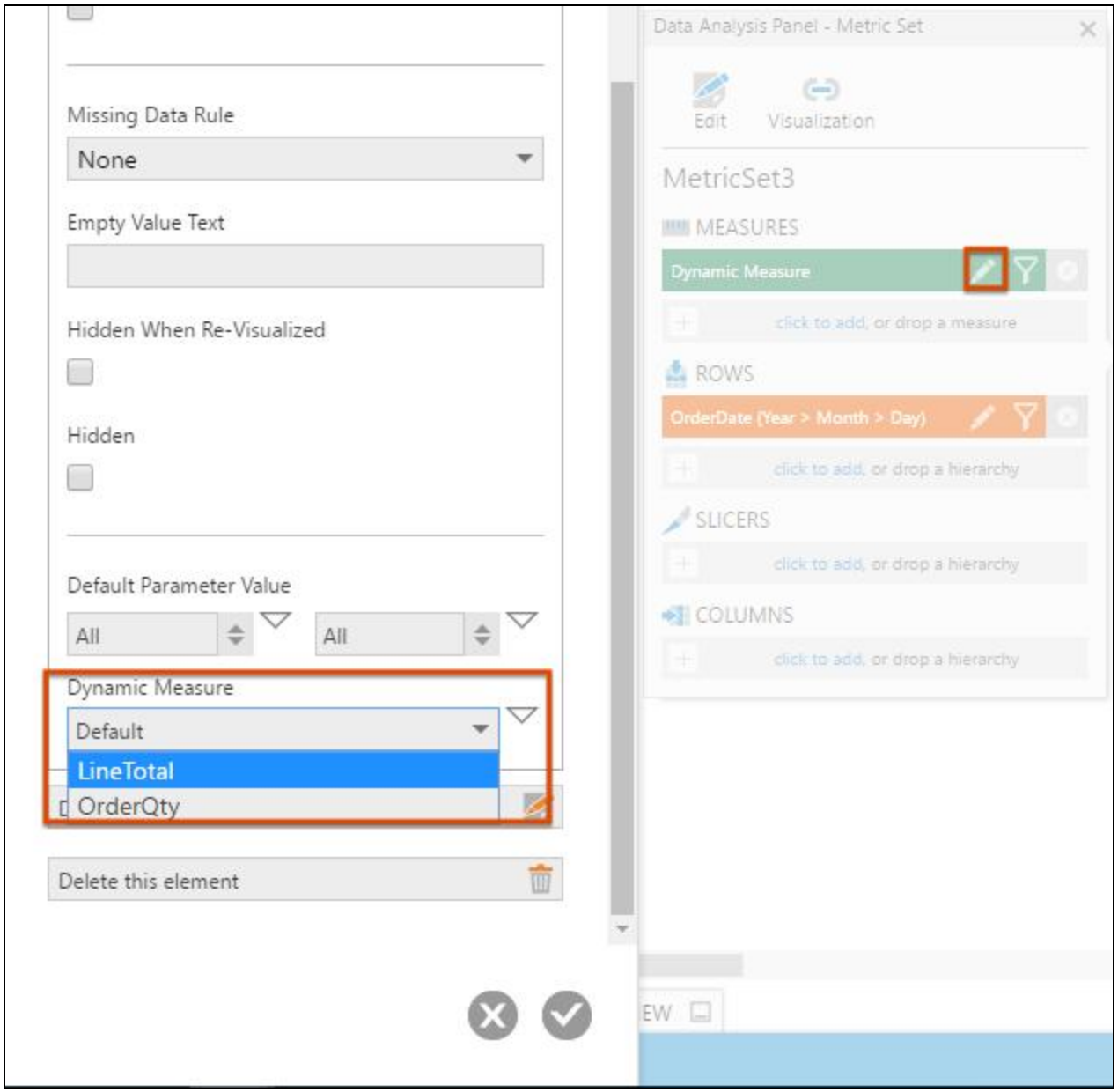


The dynamic measure is added to the Data Analysis Panel and you can see that it is set to a default measure (OrderQty) in the table visualization.

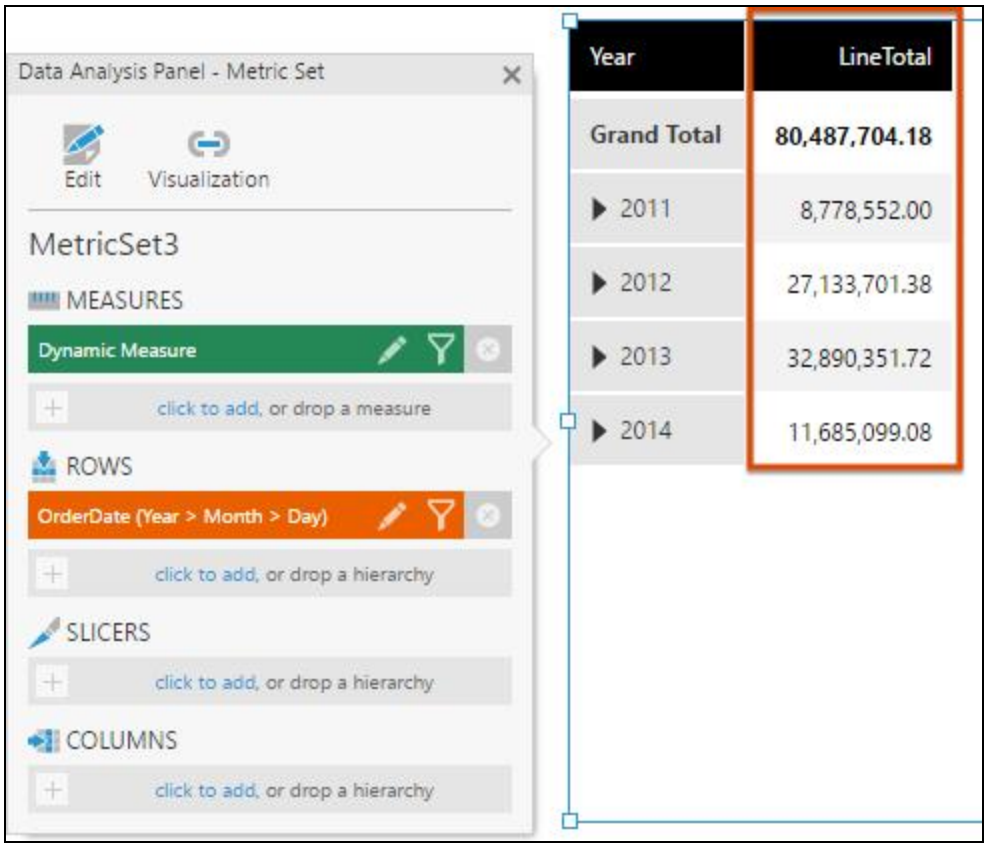


To change the default dynamic measure, click the Dynamic Measure in the Data Analysis Panel to edit it.

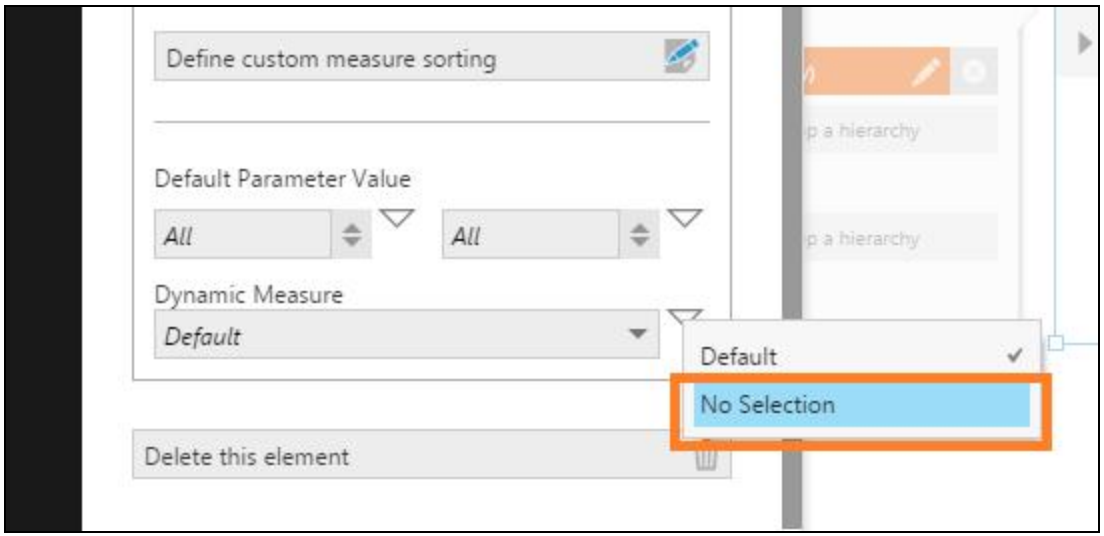
In the configuration dialog, expand the **Parameter Values** section if needed and set the **Dynamic Measure** drop down list to the default that you want. For example, change it to **LineTotal**.



The table visualization now shows the **LineTotal** measure by default.



Alternatively, you can open the token menu to the right for **Dynamic Measure** and set the default to **No Selection**. This will cause the table visualization to not display any dynamic measure initially.

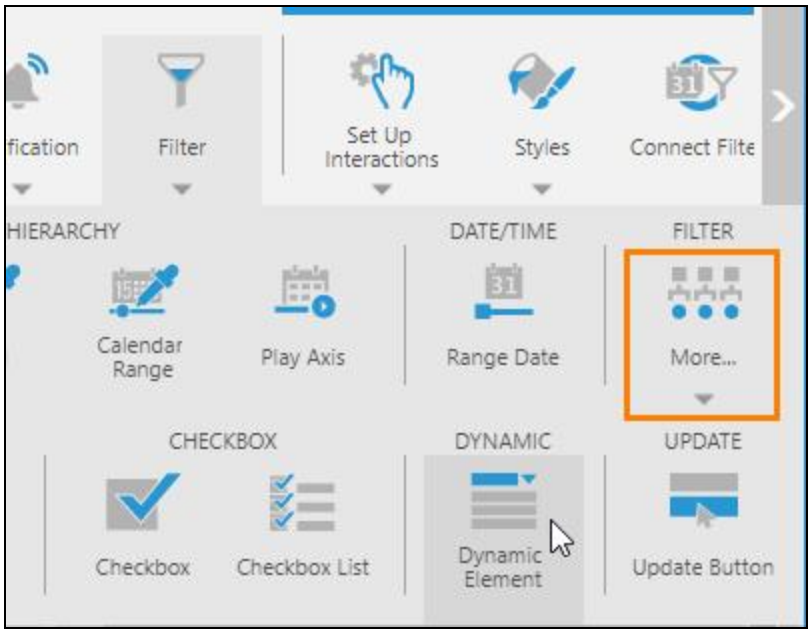


To determine the aggregator and formatting of each measure when it's selected dynamically, use a data cube as your data source (or promote the auto-generated one behind your metric set). The measure's supported aggregators and formatting settings in the data cube Process Result take effect, since the Data Analysis Panel will not be available in view mode after checking in. You could choose only one supported aggregator so that it always takes effect, and can set up distinct formatting for each measure.

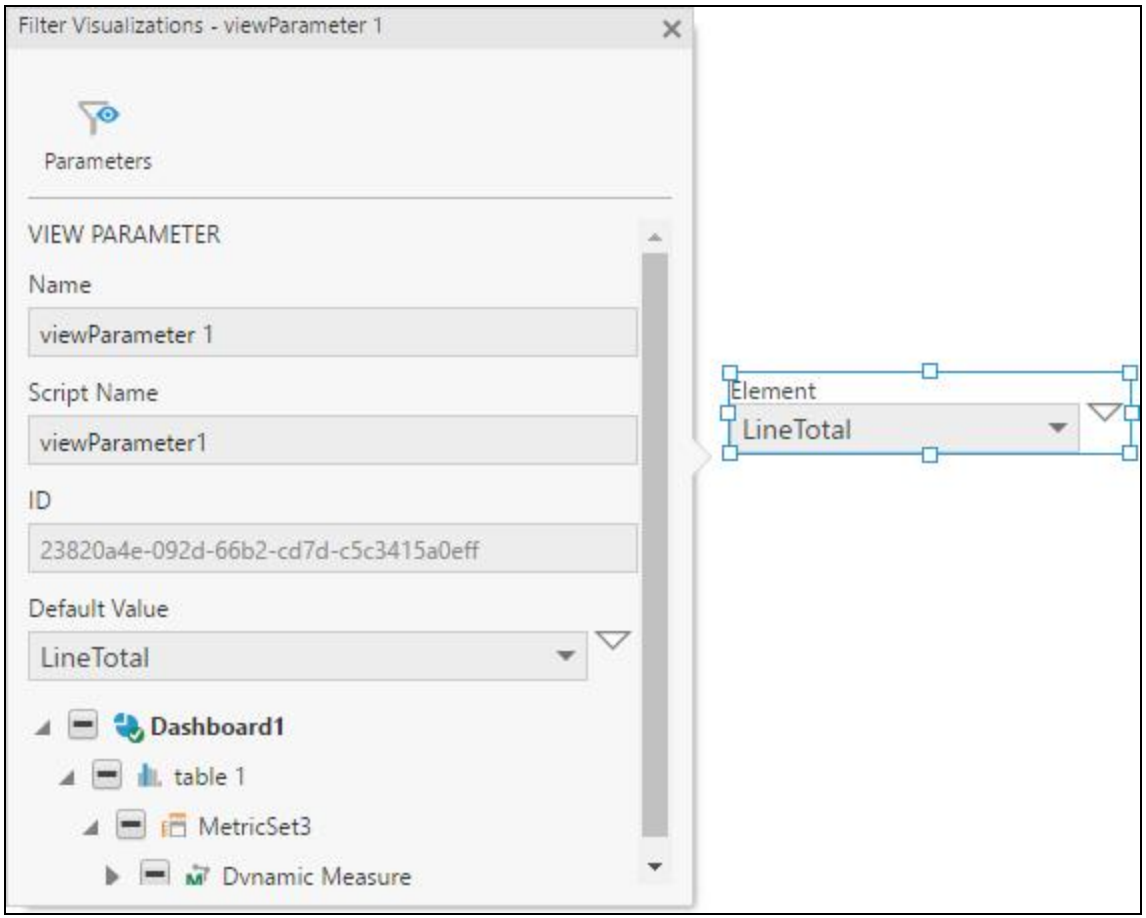
Add the Dynamic Filter

The next step is to add a Dynamic Element filter to the dashboard to allow users to switch measures via a drop down list.

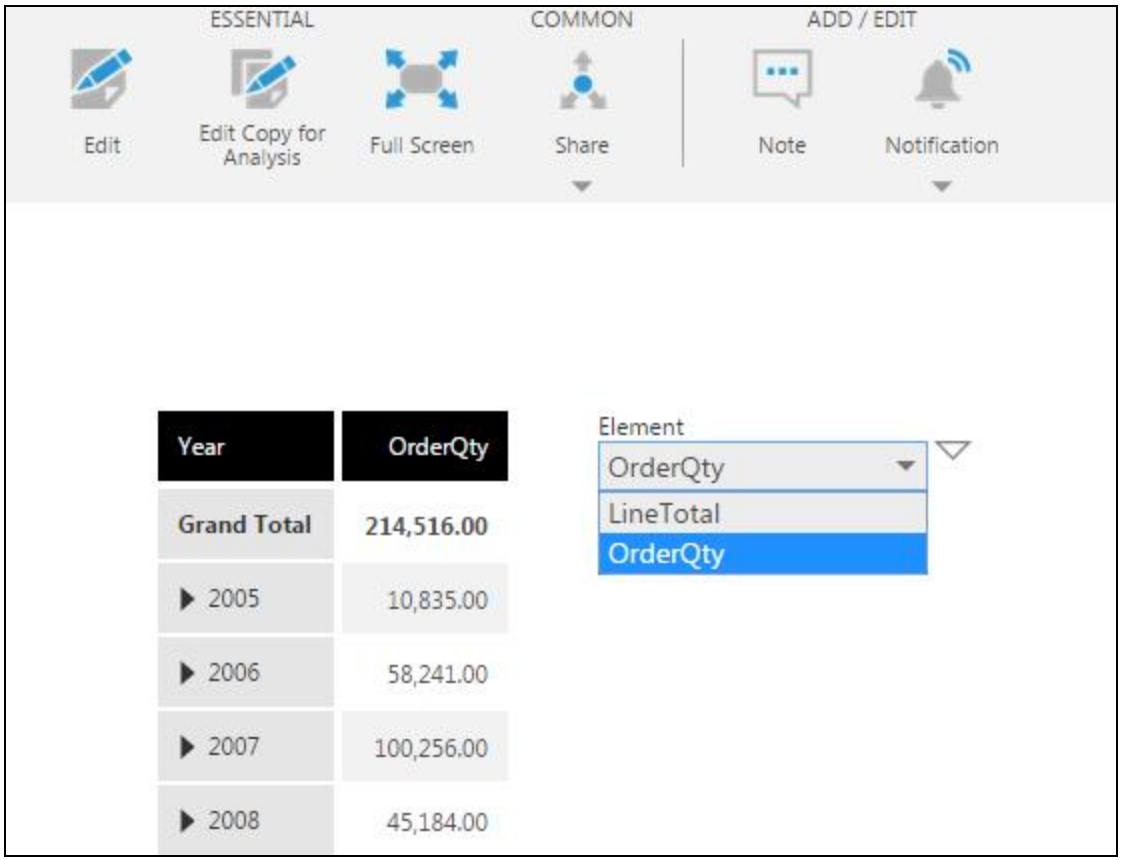
Select the table visualization on the canvas, click **Filter** in the toolbar, and then click **Dynamic Element** (found under **More**).



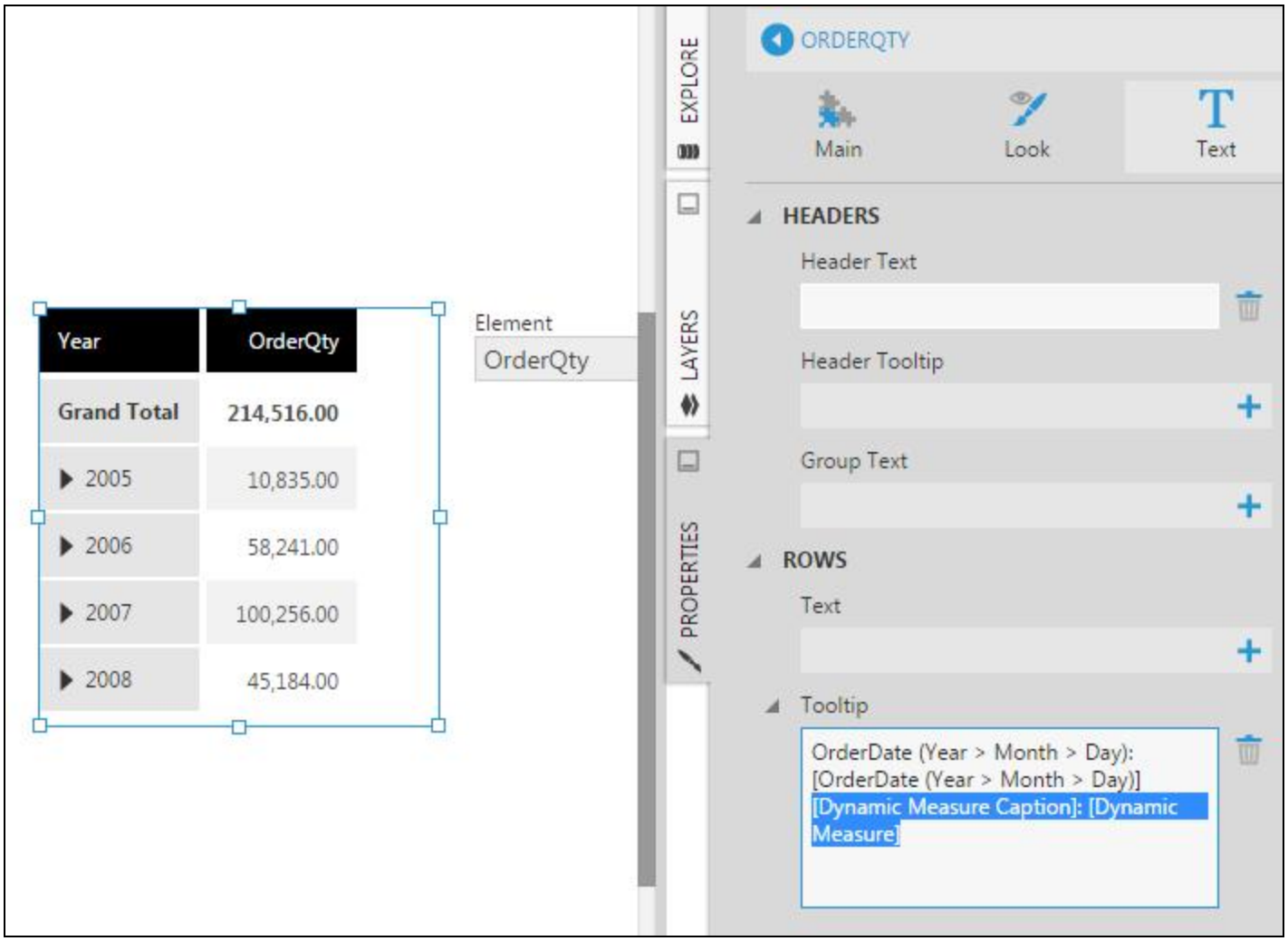
A dynamic element filter control is added to the canvas and the **Filter Visualizations** panel is opened automatically. The filter is already connected to the dynamic measure by default, so you can just close the panel.



Switch to **View** mode and use the filter drop down list to choose the measure you want to see in the table visualization. The filter's token menu can be used to switch the dynamic measure back to its default, or to **No Selection**.



If you hover over a table cell, the tooltip will show the caption and value of the current measure. This is possible because of special keywords **[Dynamic Measure Caption]** and **[Dynamic Measure]** which you can see in the **Properties** for the table column.



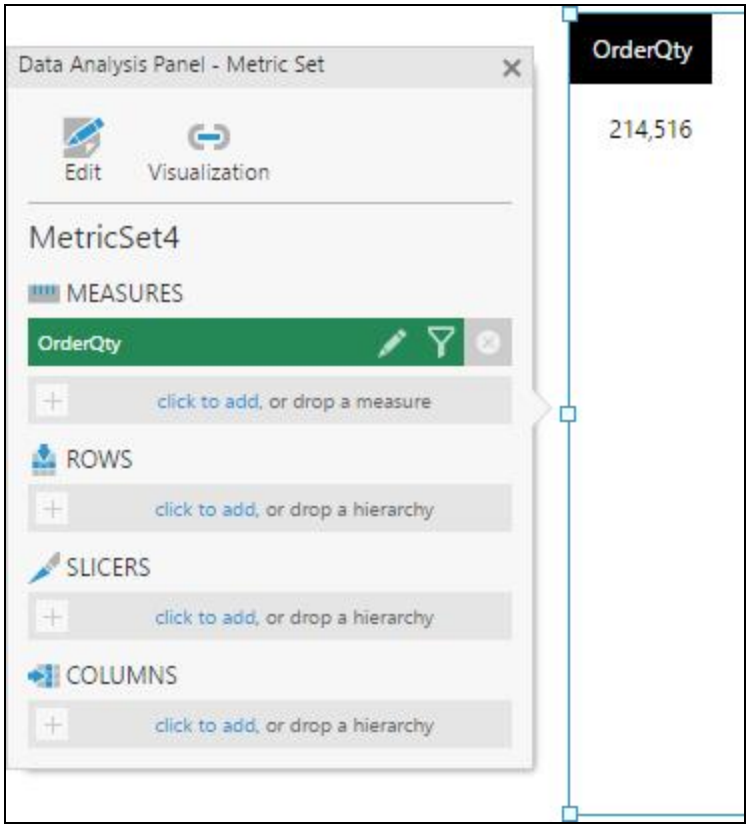
Dynamic Hierarchy

Add the Dynamic Element

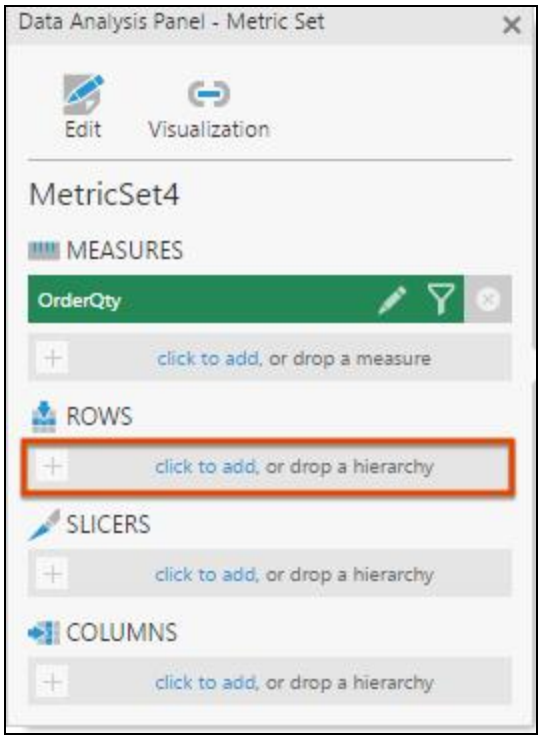
Adding a dynamic hierarchy is similar to adding a dynamic measure.

For this example, first create a new dashboard from the main menu using the **Blank** template.

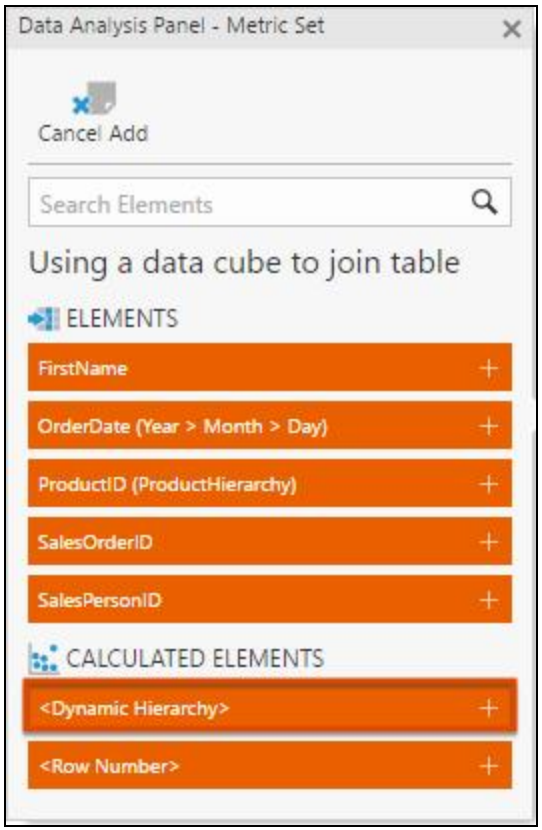
Drag the **OrderQty** measure from **Explore** to the dashboard canvas. It will appear as a table visualization with the Data Analysis Panel beside it.



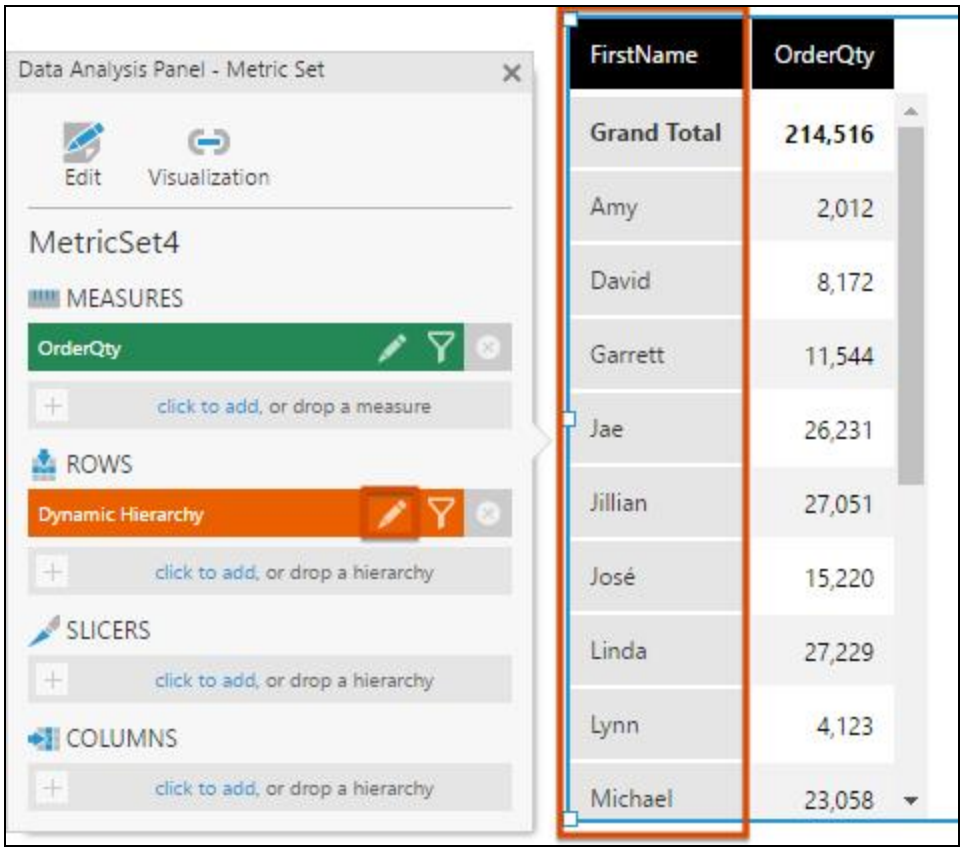
In the Data Analysis Panel, click the **click to add** link under **Rows**.



In the list of available hierarchies to add, go to the **Calculated Elements** section and click **<Dynamic Hierarchy>**.

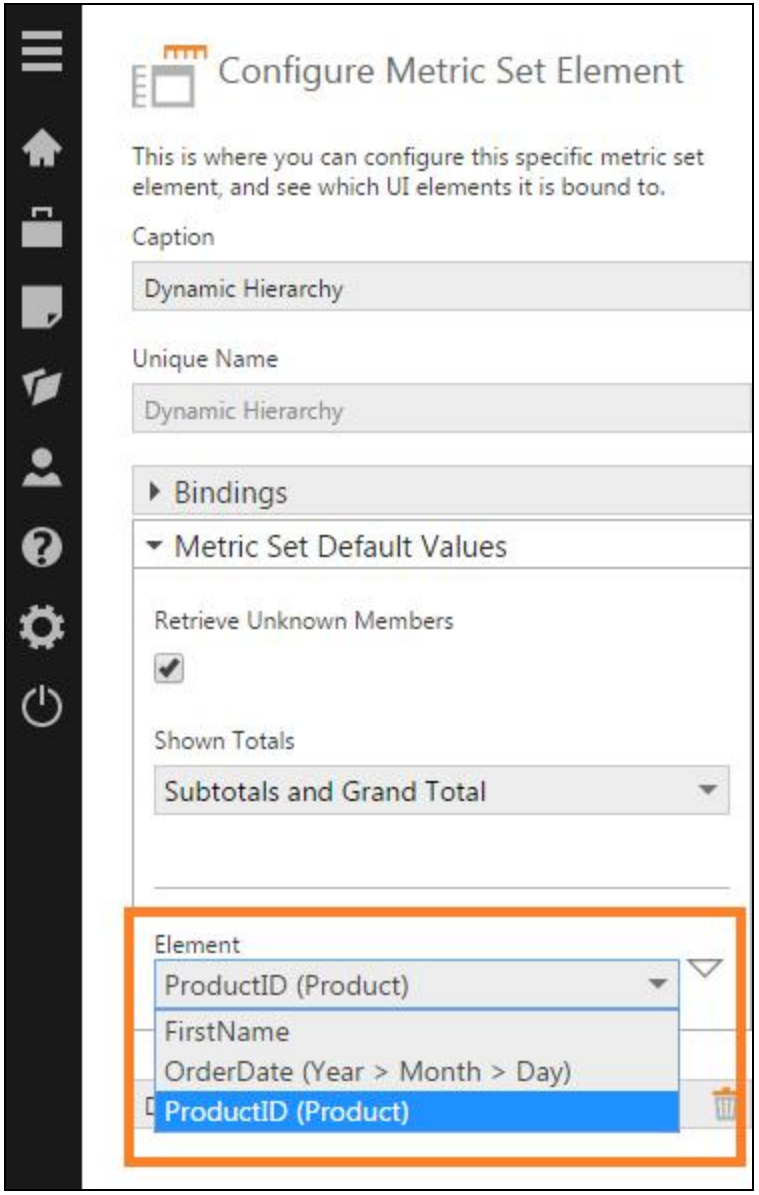


The dynamic hierarchy is added to the Data Analysis Panel and you can see that it is set to a default hierarchy **(FirstName)** in the table visualization.

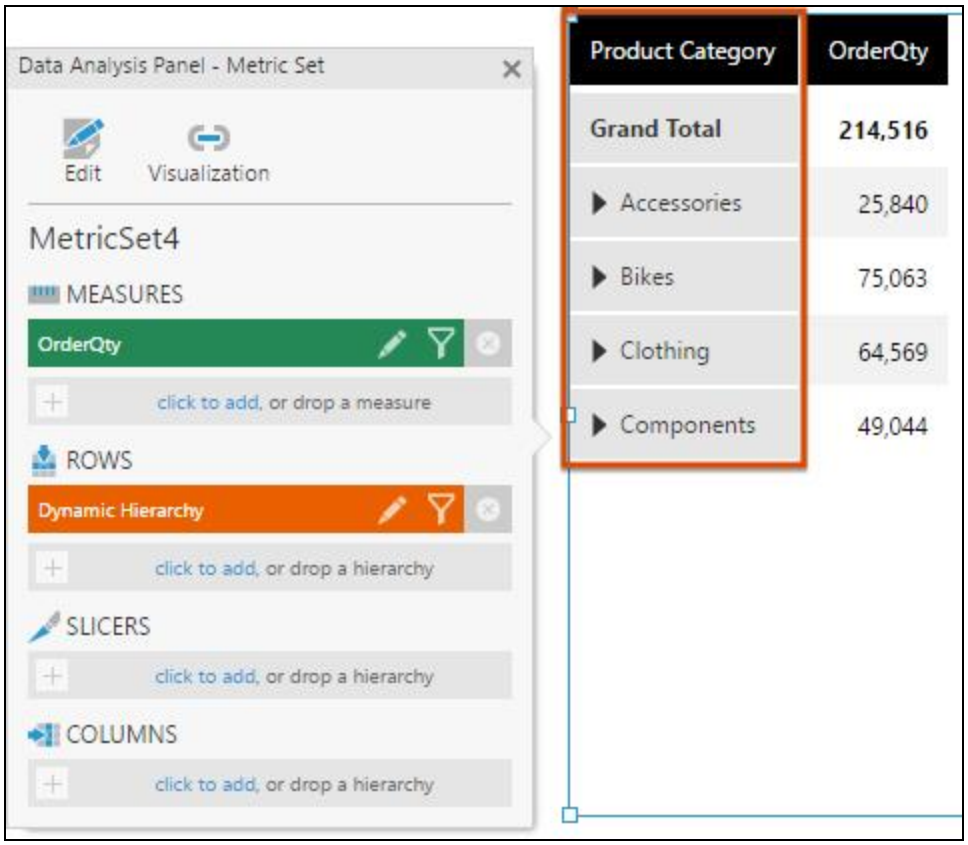


To change the default dynamic hierarchy, click the Dynamic Hierarchy in the Data Analysis Panel to edit it.

In the configuration dialog, click to expand **Parameter Values** if necessary and set the **Element** drop down list to the default that you want. For example, change it to **Product** instead.



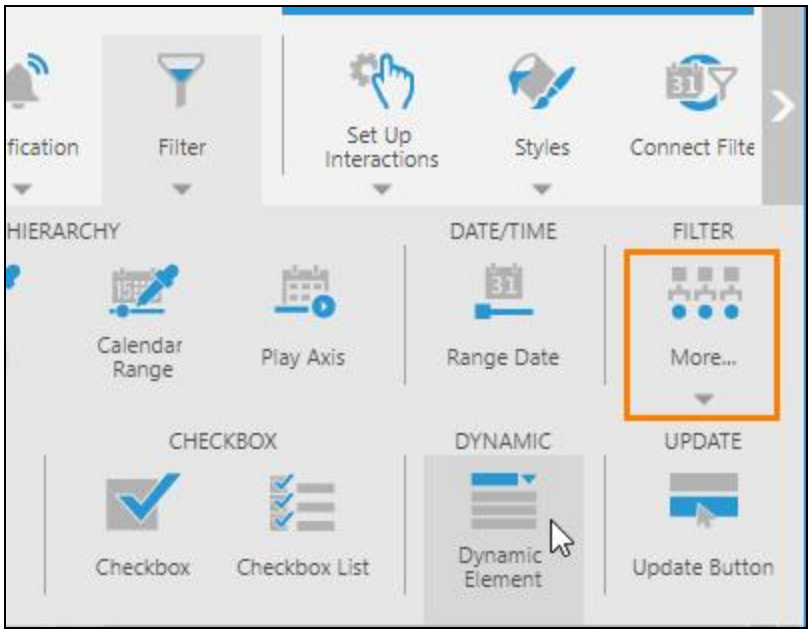
The table visualization now shows a row for each **Product** by default.



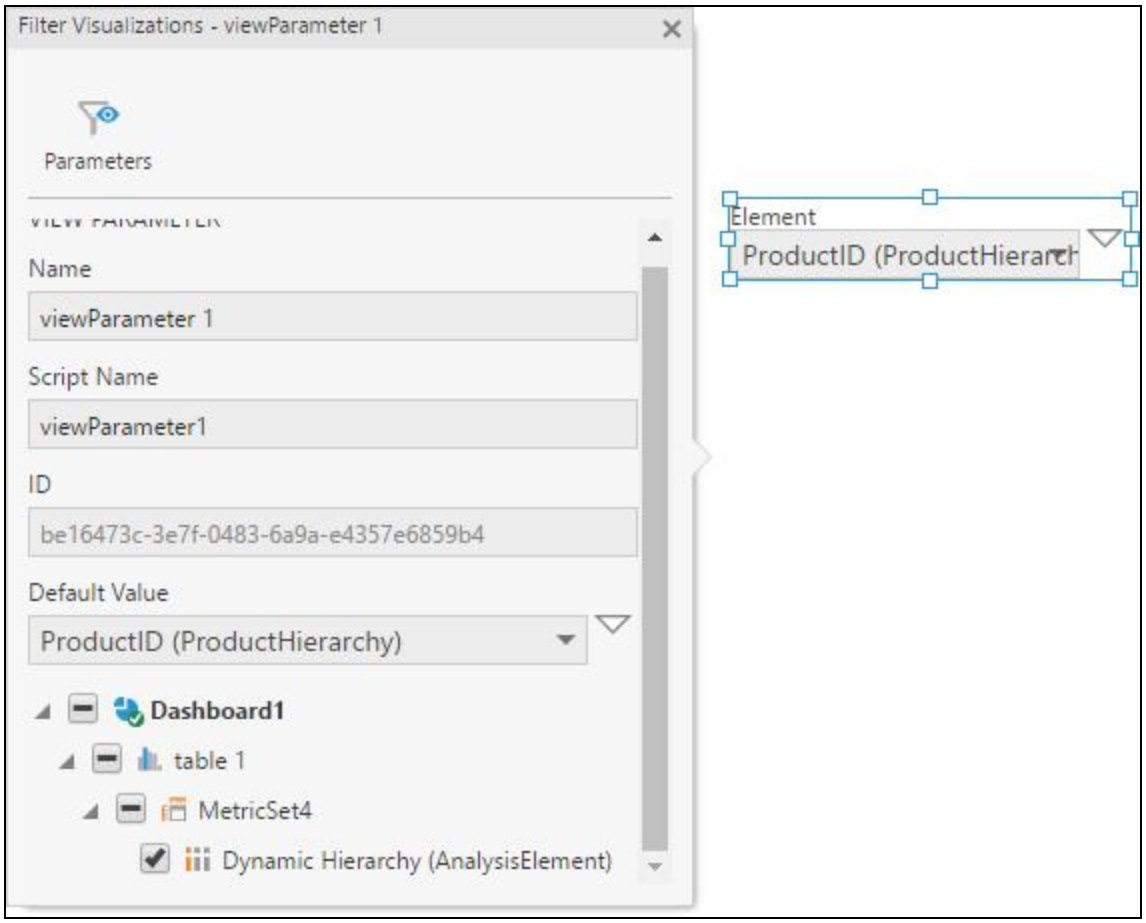
Add the Dynamic Filter

The next step is to add a Dynamic Element filter to the dashboard to allow users to switch hierarchies via a drop down list.

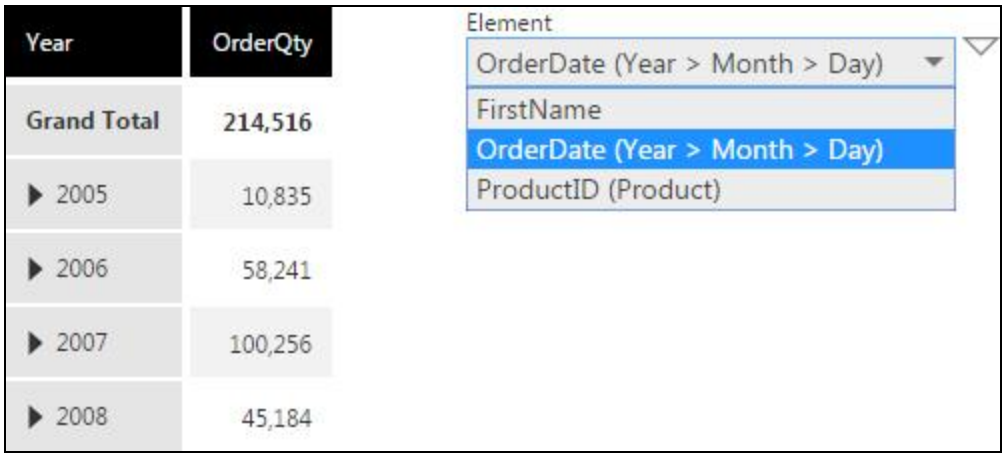
Select the table visualization on the canvas, click **Filter** in the toolbar and then click **Dynamic Element** (found under **More**).



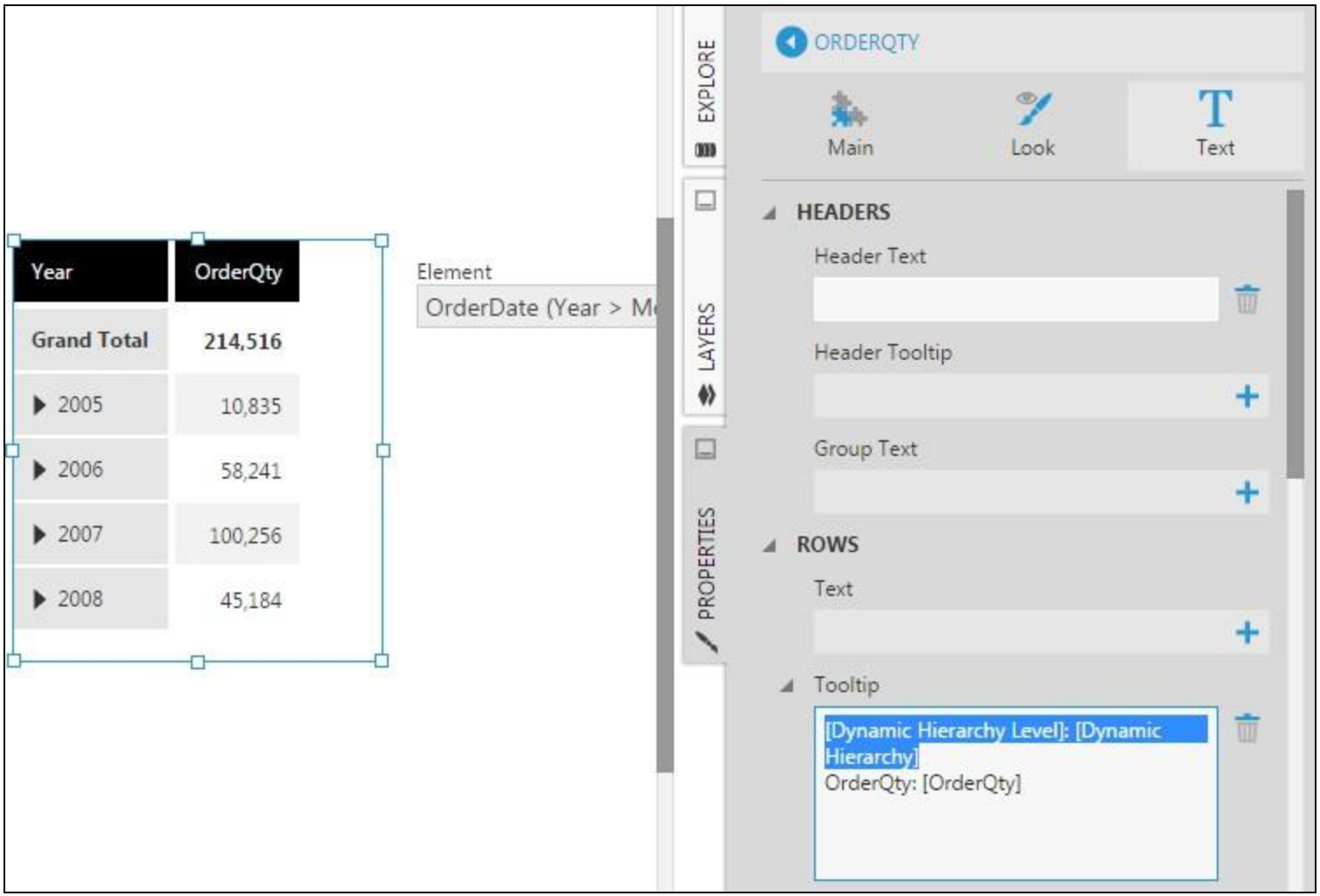
A dynamic element filter control is added to the canvas and the **Filter Visualizations** panel is opened automatically. The filter is already connected to the dynamic hierarchy by default so you can just close the panel.



Switch to **View** mode from the toolbar and use the filter drop down list to choose the hierarchy you want to see in the table visualization. The filter's token menu can be used to switch the dynamic hierarchy back to its default.



If you hover over a table cell, the tooltip will show the current hierarchy level and corresponding member. This is possible because of special keywords **[Dynamic Hierarchy Level]** and **[Dynamic Hierarchy]** which you can see in the **Properties** for the table column.



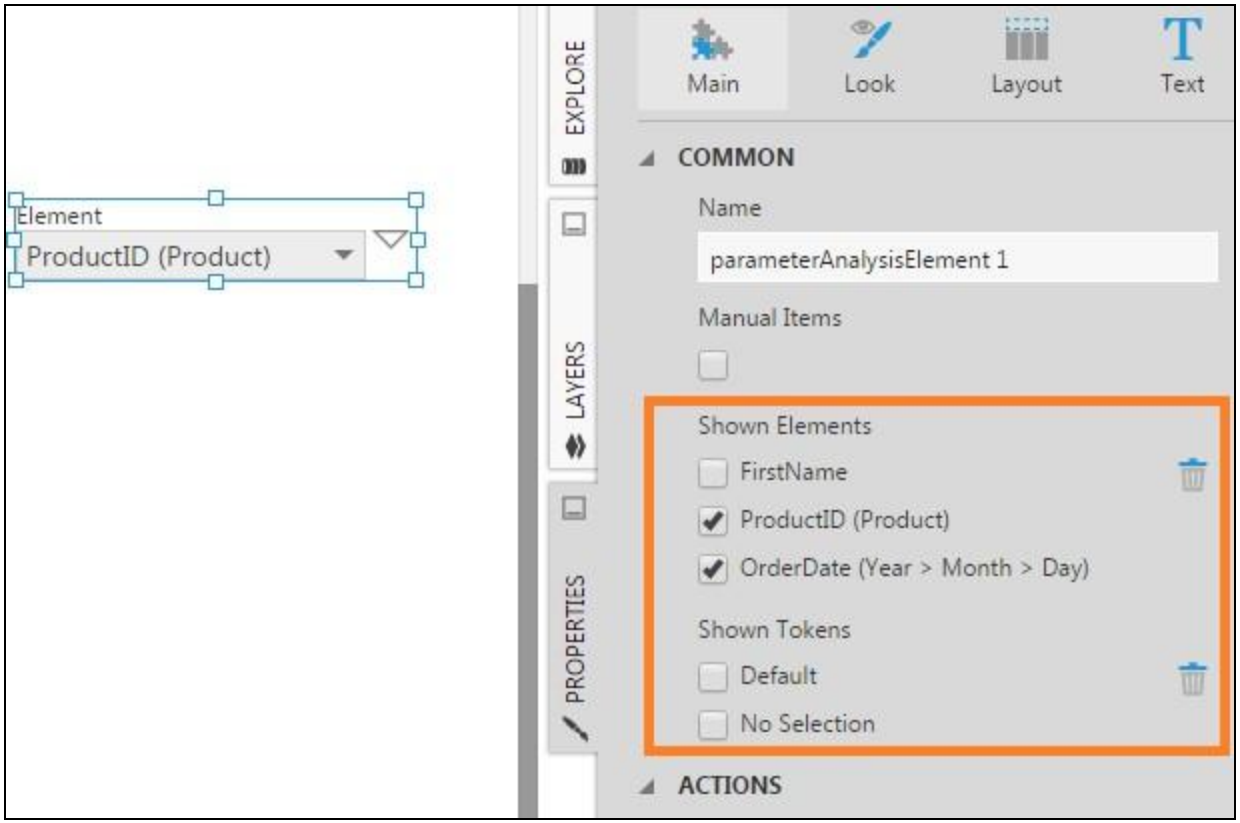
Properties

This section describes some of the key properties of a dynamic element filter that you can customize in the Properties window.

Shown Elements

The list of available measures (or hierarchies) displayed by default in a dynamic filter may include measures or hierarchies that you don't want. Instead of modifying the underlying data cube you can control exactly which elements (measures or hierarchies) will appear in the filter from its Properties.

Under the **Main** properties for the filter, click the **+** button under **Shown Elements**. You'll see a checkbox list of available measures or hierarchies. Simply select the ones you want to appear in the dynamic filter.



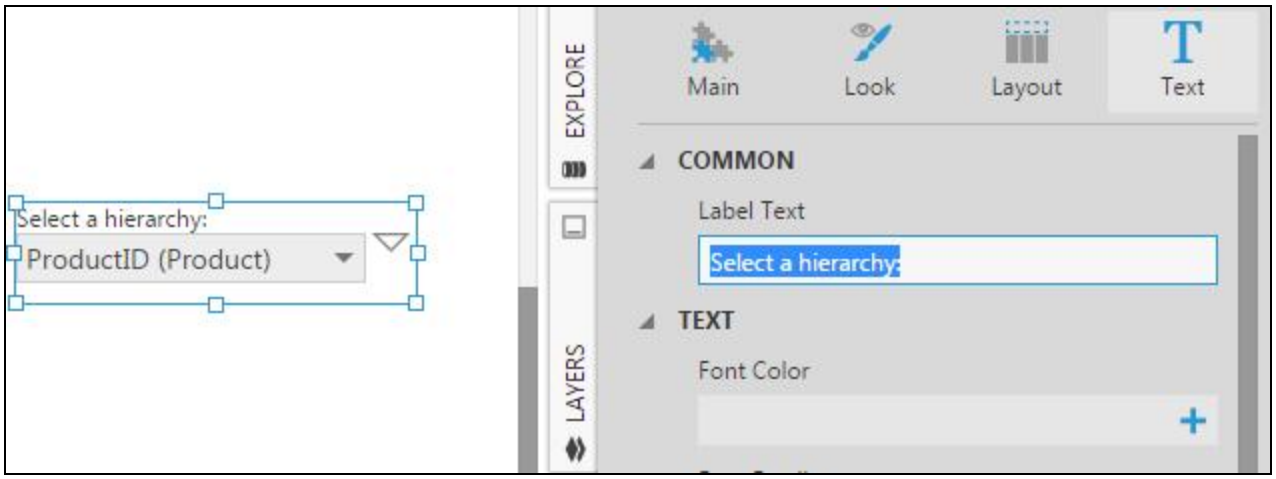
Shown Tokens

In a similar way, you can control which tokens you want to display in the filter's token menu via the **Shown Tokens** property. If you want to hide the token menu altogether, you can do so in the **Look** tab under **Appearance**.

Note: The **No selection token** option is not available for dynamic hierarchies.

Label Text

Under **Text** properties, use the **Label Text** property to change the text that appears above the filter's drop-down.



Note: Dynamic hierarchies and measures are not meant to replace editing a metric set for performing analysis, and have more limited filtering and sorting options.

For more information, see:

- Create a New Data Cube
- Adding Filters
- Process Result
- Formatting Text
- Use a Drop Down List to Switch Metric Sets With Scripting

Adding Filters

This applies to: *Managed Dashboards, Managed Reports*

Filters can be added to your dashboard or view to allow you to easily see and change the current filtering while viewing. All user types can add filters when editing, and can connect them to multiple visualizations to filter them at the same time.

There are a variety of filter types available for different types of data, offering different ways to interact with the filters. When using a dashboard, you can also arrange the filters wherever you like on the canvas.

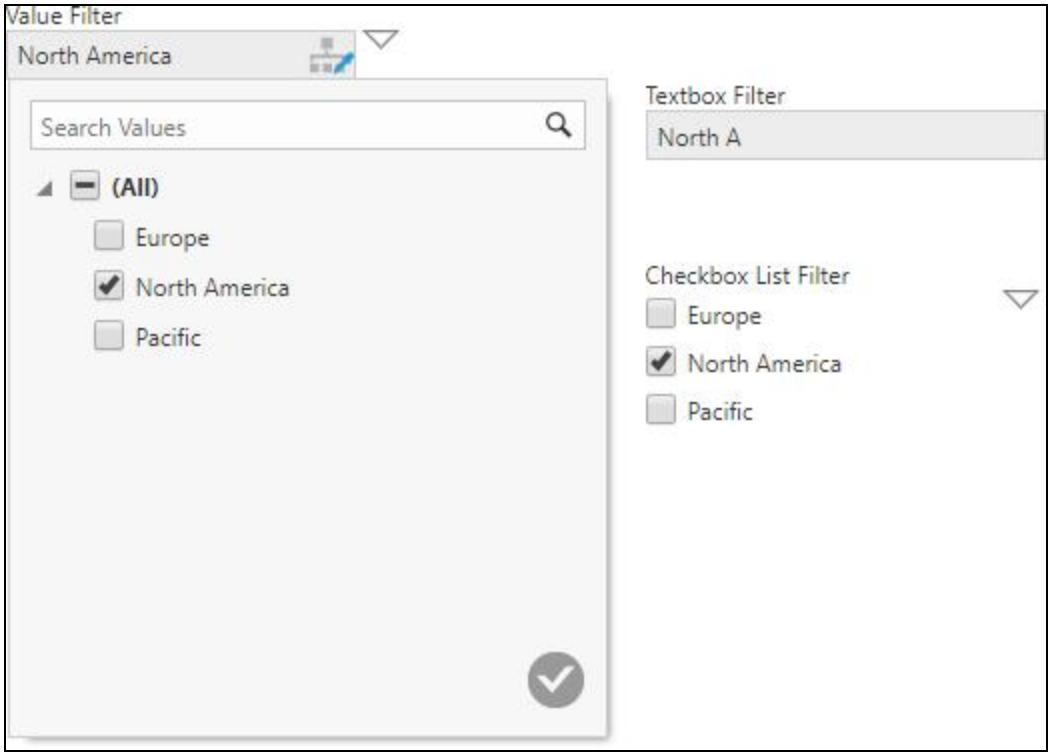
> **Note:** When editing a metric set, you can instead filter the data directly via the Data Analysis Panel. You can add the metric set to a dashboard and add your own filter as shown below for more filtering options.

Related video: Filters

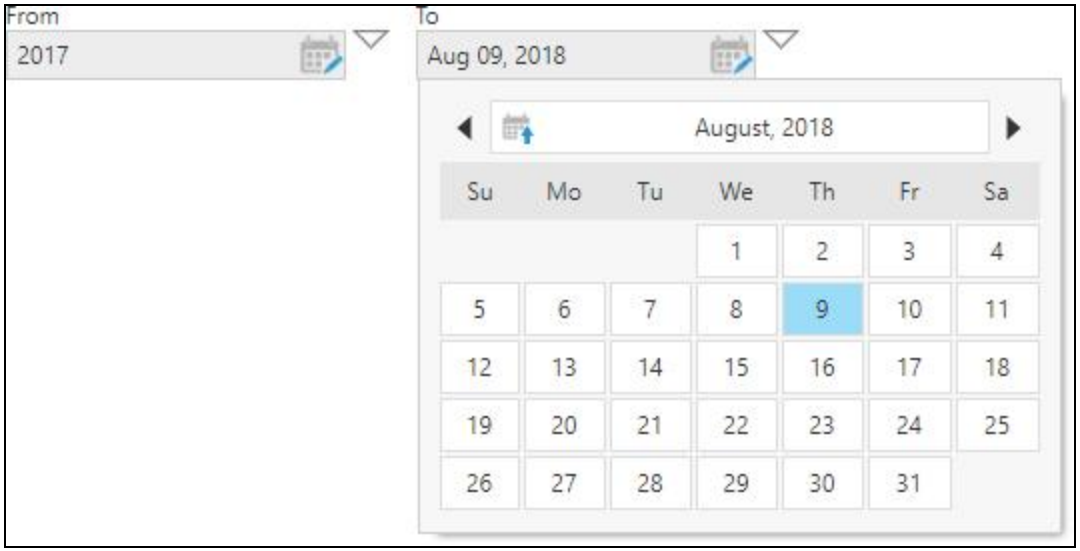
Types of Filters

Filters can be added to filter the data directly by some values, or to filter in other ways:

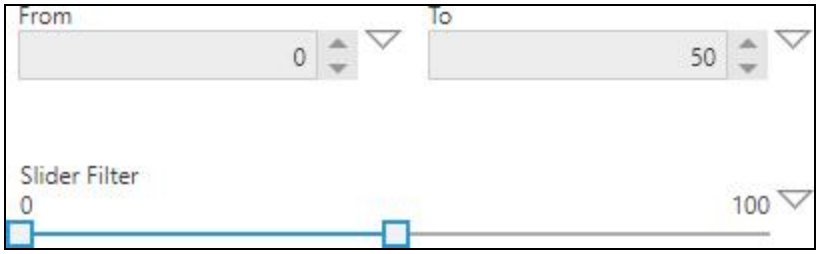
- **Hierarchy & string values:** All non-numeric data in Symphony becomes a hierarchy when selected in a metric set. Data cubes and formulas may also have string (text) parameters that you can connect to.



- **Date values**: Date/time, calendar, and the play axis filter are available for dates and time hierarchies.

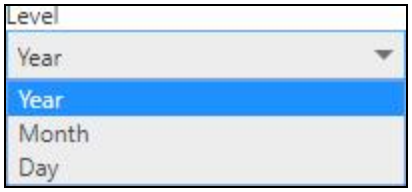


- **Numeric values**: You can filter numeric data by inputting a number or range, or using a slider.

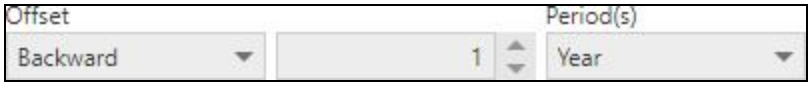


- **Boolean (True/False) values**: A checkbox filter can filter by Boolean-type parameters from a data cube or formula.

- **Hierarchy level**: Add a hierarchy level filter to easily switch between levels in a multi-level hierarchy.



- **Period over period**: The Time Lag filter can change which period is displayed alongside your measure values for comparison, e.g., last year vs. two years ago.

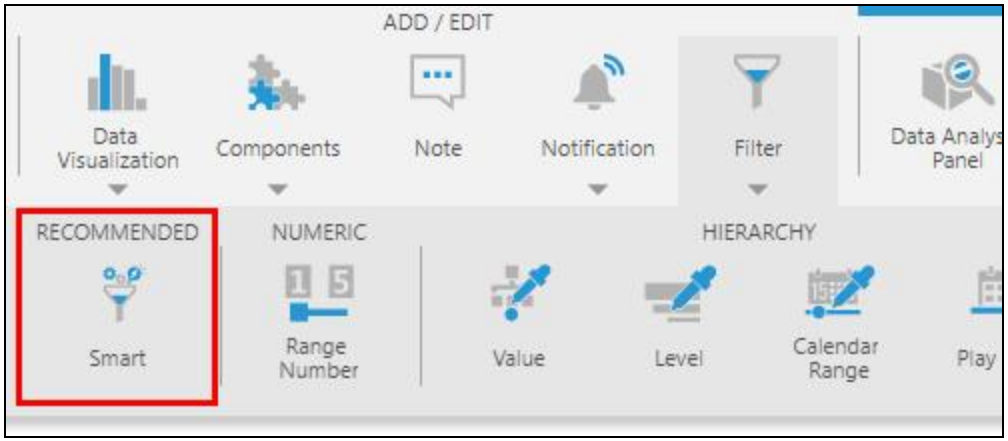


- Dynamic elements: The Dynamic Element filter can be used if you are using dynamic measures or dynamic hierarchies.
- Update button: Although not a type of filter, you can add an update button connected to other filters to apply them all at once only when the user clicks it.

Adding a Filter

Filters can be added when editing a dashboard or other views such as reports.

To automatically add the most common type of filter for your data, simply select one or more visualizations displaying the data that you want to filter, choose **Filter** from the toolbar, and then **Smart**.



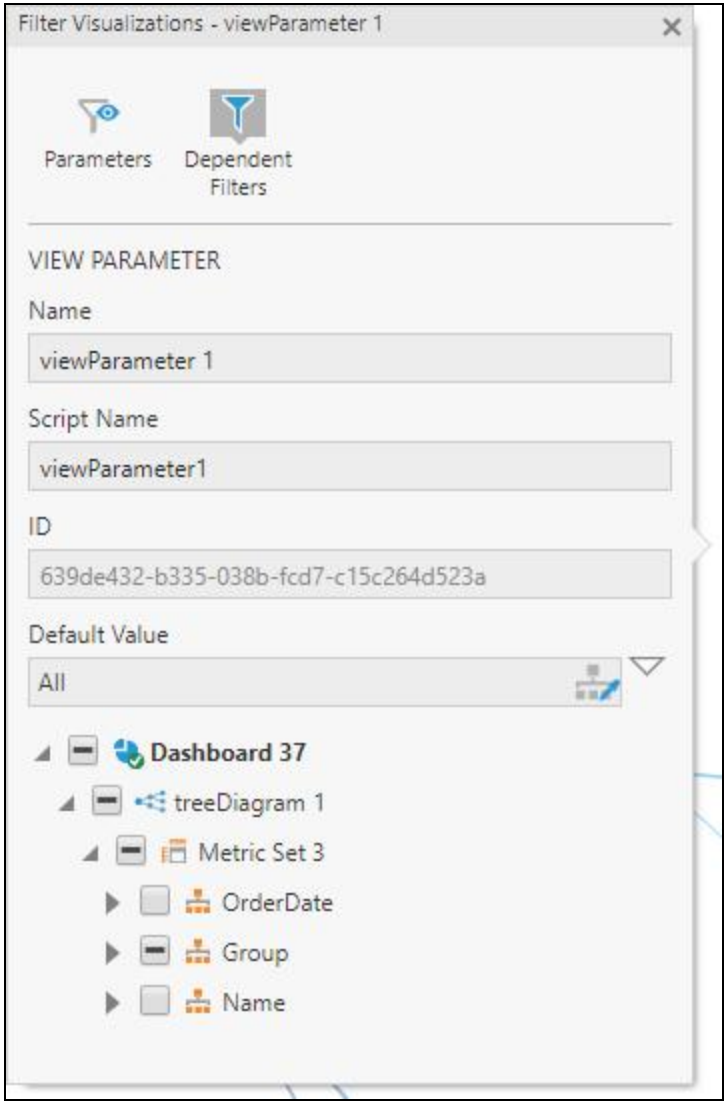
If editing a dashboard, the filter will then be added to the canvas, where you can position it like other content wherever you like. With other types of views such as reports, filters will be added to the parameter bar instead.

If you know what type of filter you want to add, you can choose that instead from the toolbar. Or you can keep choosing the Smart option to add filters for other data, and then delete any you don't want.

To interact with the filter directly, click **View** in the toolbar. (In a report or scorecard, open the filter bar using the **Change Filters** button in the toolbar. For details, seeSorting and Filtering a Report or Scorecard).

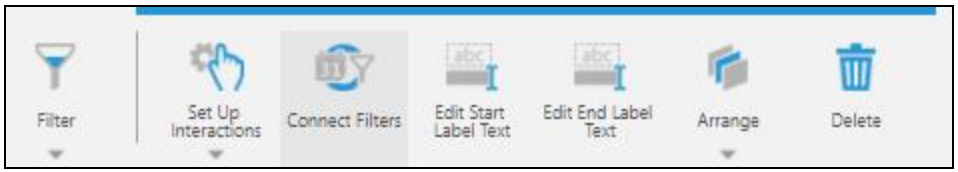
Connecting Filters

When a new filter is added, the Filter Visualizations panel appears beside it, allowing you to choose which data under which visualizations should be affected by this filter. You can connect to more than one visualization if they are displaying values that are compatible, even if they come from different data sources.



If your data comes from a data cube with parameters that someone made public, or contains formula parameters, these may also be listed for you to connect to if the filter is compatible.

To re-open this panel later, select the filter and choose **Connect Filters** in the toolbar.



If a visualization was selected when you added the filter, one of its hierarchies or measures will be checked automatically.

To change it, un-check it first and then check the hierarchy or measure you want. (When an item is checked, other items may be hidden if they have an incompatible data type.)

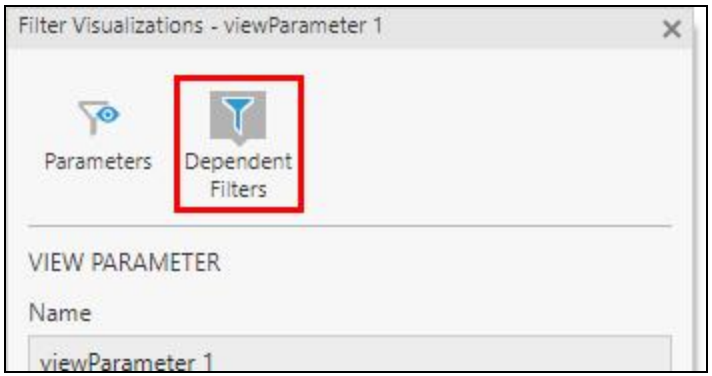
You can set the default value for the filter in this panel, which is the filter setting that will take effect for each viewer until they change it.

Dependent Filters

You can create a dependency between filters if you want one filter to determine what values can be chosen in another filter.

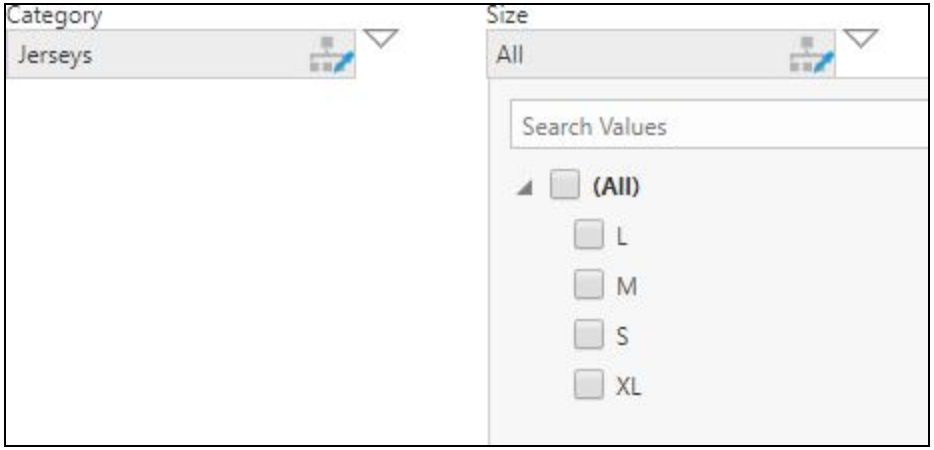
Note: This functionality is available only when the hierarchies belong to the same data cube, and does not work with bridge parameters or OLAP data.

In the **Filter Visualizations** panel, click **Dependent Filters**.



Check the checkbox next to whatever filter should be "filtered" by this one.

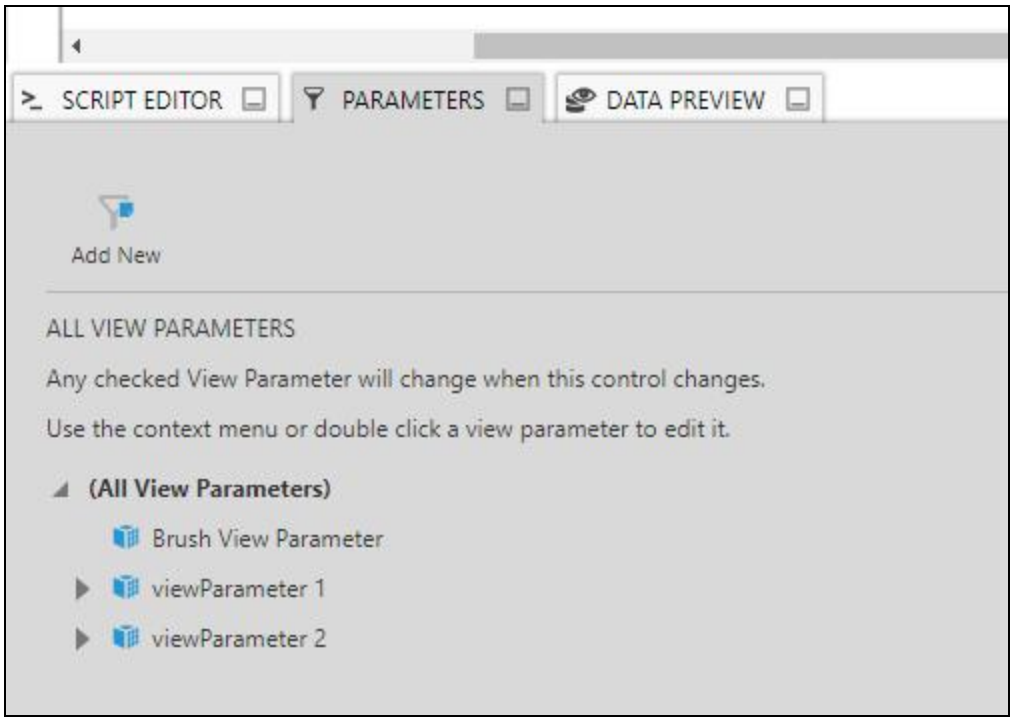
Click **View** in the toolbar, and change this filter to a subset of the available data. Observe that the dependent filter now offers only a subset of options.



Parameters

In the Filter Visualizations panel described above, notice that it mentions a **view parameter**. View parameters can filter visualizations on your dashboard or other view by passing filter values to the metric sets and data cubes powering those visualizations.

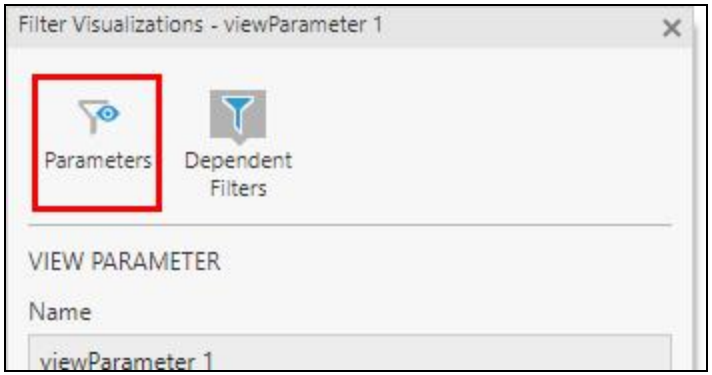
A view parameter is created automatically for the filter when you add it, and is opened for editing automatically in the Filter Visualizations panel so that you can connect it. You can also find this parameter in the Parameters window, normally located at the bottom of the screen when editing.



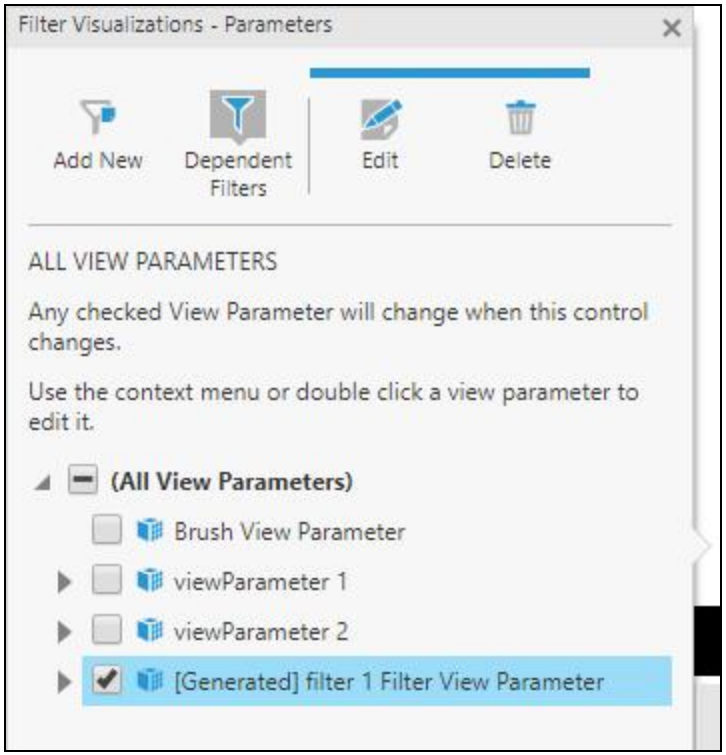
View parameters are also used for interactions, such as when a chart is set up to filter when a data point is clicked. See Using Interactions.

If the filter placed on the dashboard and an interaction are connected to the same view parameter, the filter can automatically update when the user interacts with a value in the visualization, triggering the filter interaction.

If you want to change what view parameter a filter is connected to, click the **Parameters** button in the Filter Visualizations panel.



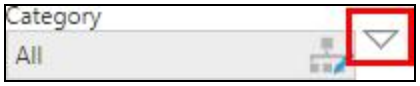
Here you can select which view parameter should be changed by this filter. To ensure the filter also updates itself automatically when the view parameter changes a different way, such as through a filter interaction, choose just one view parameter.



To return to the view parameter's details normally shown in the Filter Visualizations panel, choose **Edit** from the panel's toolbar or the view parameter's context menu (right-click menu).

Tokens

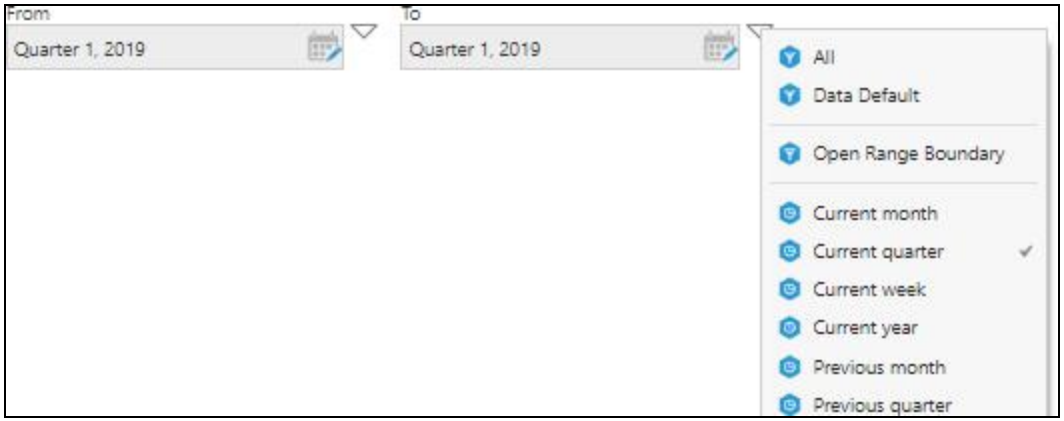
Most filters and their parameters allow you to choose from a list of options besides the data source values, called **tokens**. The token menu can be accessed from the triangle-shaped button to the right of the filter.



The most common is the **All** token, which is what is often selected by default. This represents all possible data source values, or no filtering.

The **Data Default** token represents the default value that was set at the metric set level or at the data cube level.

Date and calendar filters offer a wide variety of common relative time tokens that can automatically change over time while being set only once, for example **Year to date**. When using a range filter with **From** and **To** inputs, each token menu may contain different options.



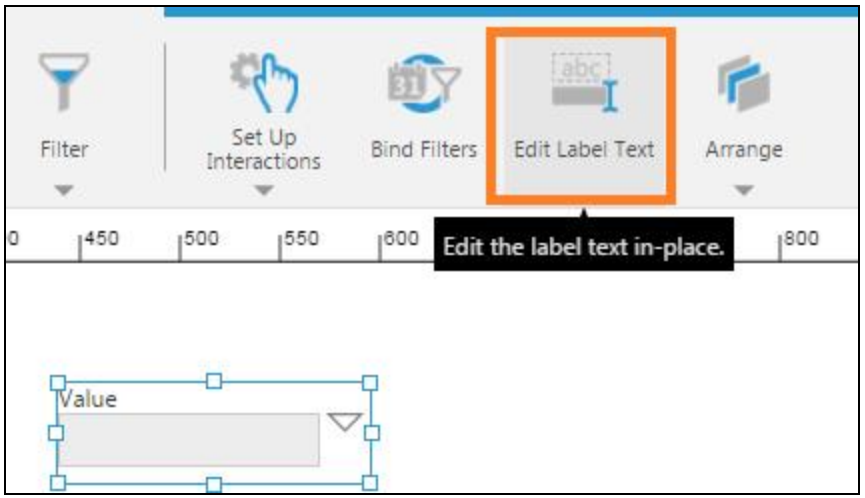
When the relative time tokens don't provide what you need, choose **Advanced...** at the bottom of the tokens menu, choose one of the existing tokens, and then an offset. For example, **Today** offset by **1** represents tomorrow, while **Current month** offset by **-3** represents three months ago.

You can also create your own tokens:

- Add a **Data** token to represent whatever the first, last, minimum, or maximum value of a column currently is.
- For time dimensions or date/time data, create custom **Relative Time** tokens by choosing a time dimension level such as **Month** and an offset, and assigning a name.
- Assign a name to a set of values selected in a filter to create a **Named Set** token to reuse later.
- You can create a **Script** token to use in the token menu for any type of filter or data.
- See Creating Custom Tokens.

Customizing Filters

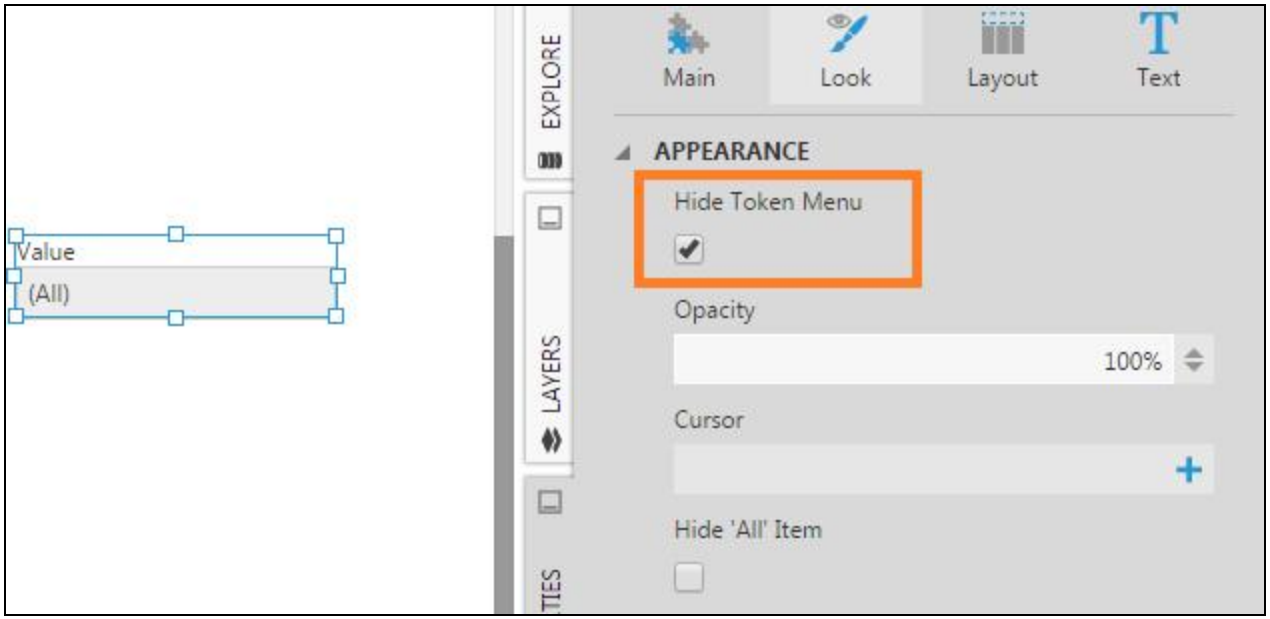
You may want to change the label displayed above the filter. Choose **Edit Label Text** (or **Edit Start Label Text**, etc.) in the toolbar to type in a new label.



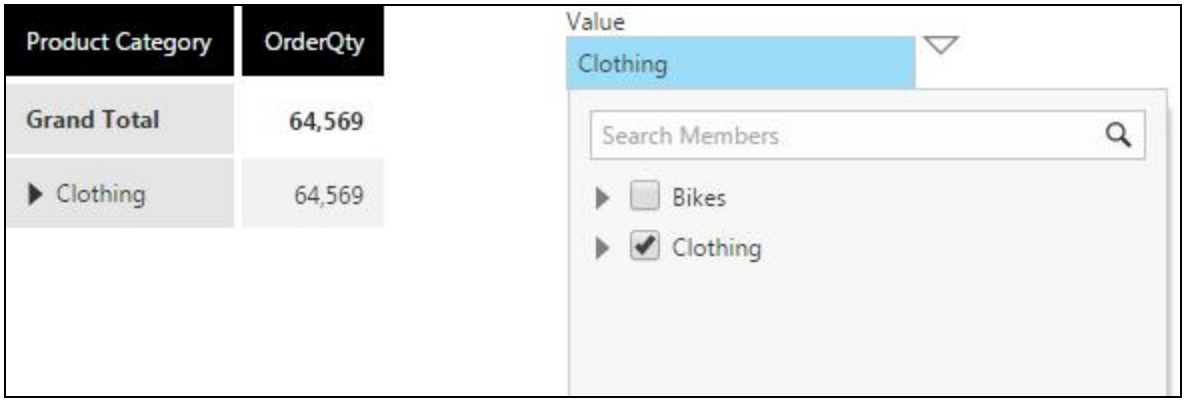
Users with access to the **Properties** window can customize filters further. Some common properties:

Use the **Shown Tokens** property on most filter controls to restrict the tokens available from the token menu.

You can choose whether to show the token menu using the **Hide Token Menu** property in the Look tab. For range filters, set the **Token Menu Visibility** property instead and choose whether to show/hide the menu for the start value, end value, or both.



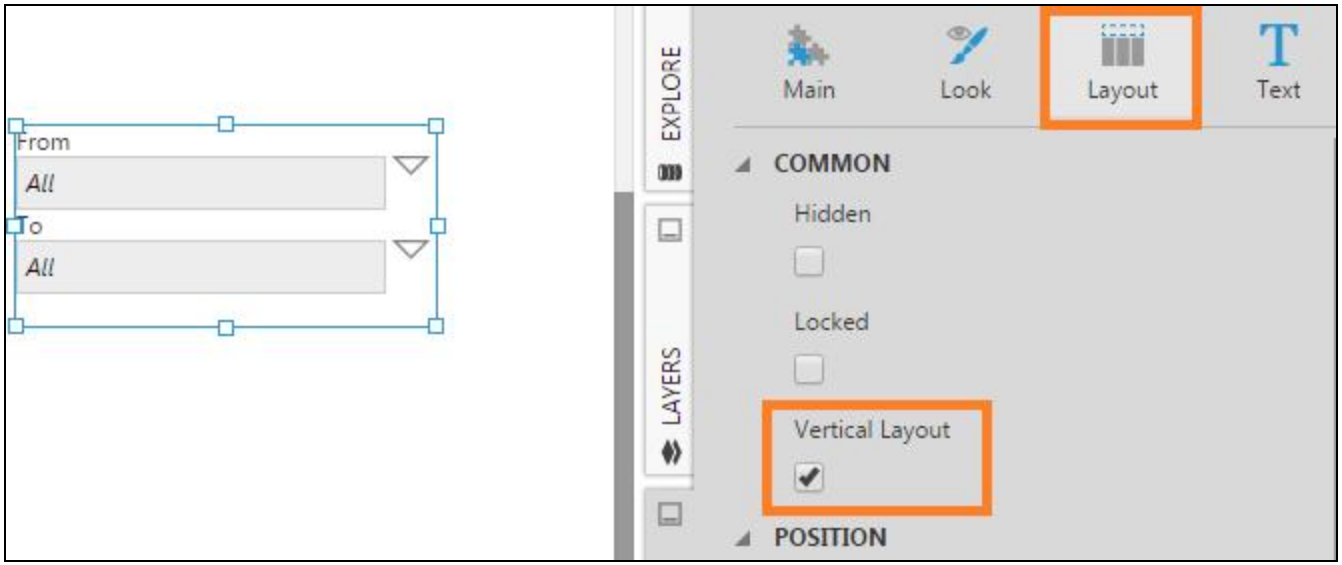
In filters that display hierarchy members, check the **Hide 'All' Item** property in the **Look** tab to hide the hierarchy's own 'All' member, which is separate from the All token.



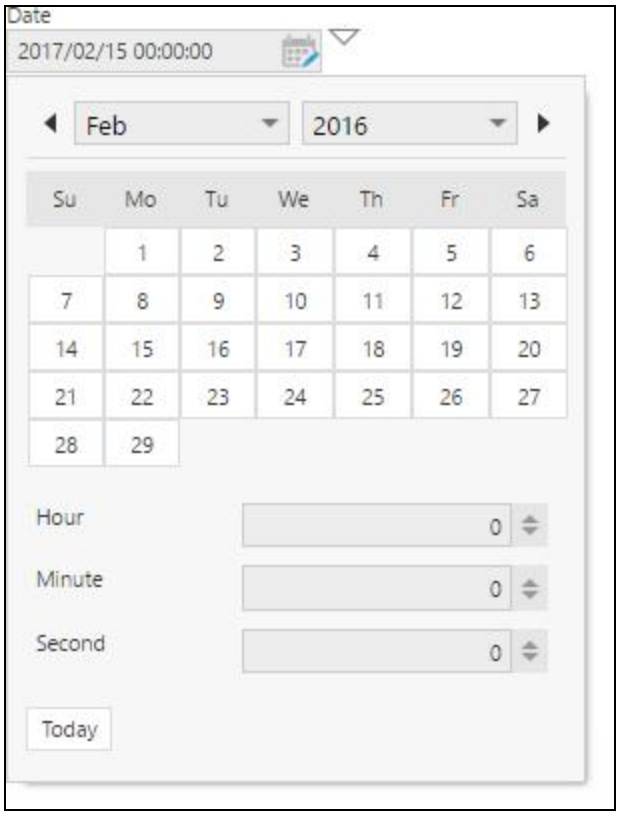
Use the **Shown Members** property of a hierarchy value filter to restrict the values that can be chosen.

Un-check the **Show Checkboxes** property of a hierarchy value filter to allow only a single value to be selected.

Filters with multiple fields such as Calendar Range and Cascading can be laid out vertically rather than horizontally by checking **Vertical Layout** in the **Layout** tab.



Date/time filters can allow time of day to be entered if the **Time Granularity** property is set to a unit of time. Choose the smallest unit of time that you want to be able to enter.



Note: The granularity may be limited by some touch devices. For example, seconds may be unavailable.

For more information, see:

- Design Overview
- Design Tips
- Create a Metric Set and Add a Filter
- Adding Filters

- Filter Child Dashboards From a Parent Dashboard
- Sorting a Table Visualization
- Symphony Metric Sets
- Displaying Top/Bottom Records
- Define Custom Hierarchy Sorting
- Define Custom Measure Sorting

Check Similar Parameters in a View

This applies to: *Managed Dashboards, Managed Reports*

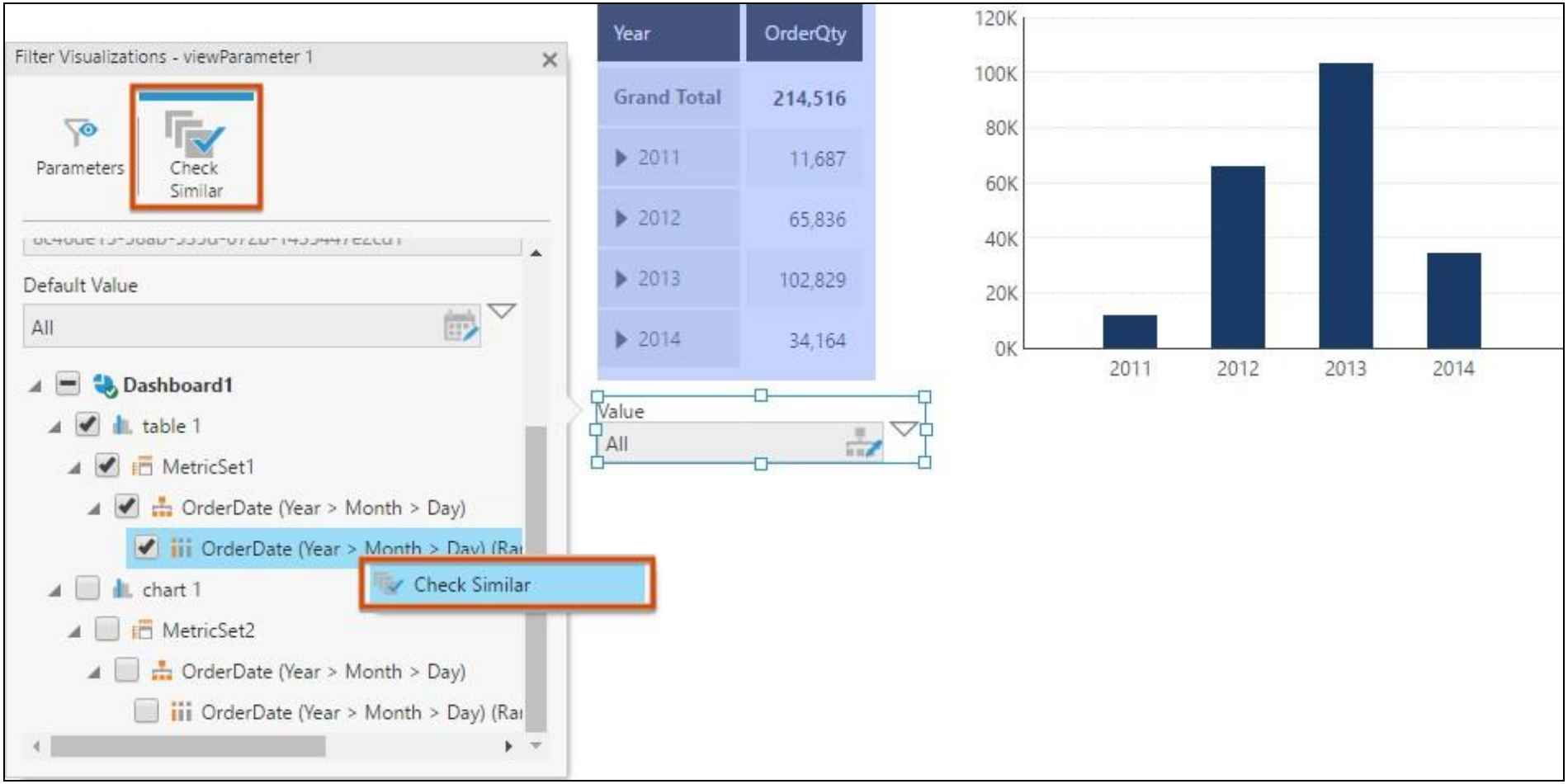
The **Check Similar** option makes it easy to connect a filter or view parameter to multiple metric sets.

If you have multiple metric sets displayed on a dashboard, and each makes use of the same or a compatible hierarchy such as **OrderDate**. When you first add a filter, compatible parameters are automatically connected in all of the visualizations you currently happen to have selected. Afterwards, you can still choose to connect to all compatible parameters in one step as shown below. Compatible parameters generally have the same type and unique name, except for date/time type parameters that are considered compatible regardless.

Using the Filter Visualizations Panel

Open the **Filter Visualizations** panel by choosing **Connect Filters** in the toolbar.

Expand the hierarchy you want to connect to (e.g., **OrderDate**) to find its associated parameter (e.g., **OrderDate** (**RangeDateTime**, **AllHierarchyValues**)). Right-click on this parameter and select **Check Similar** from the context menu, or select the parameter and click **Check Similar** from the toolbar.

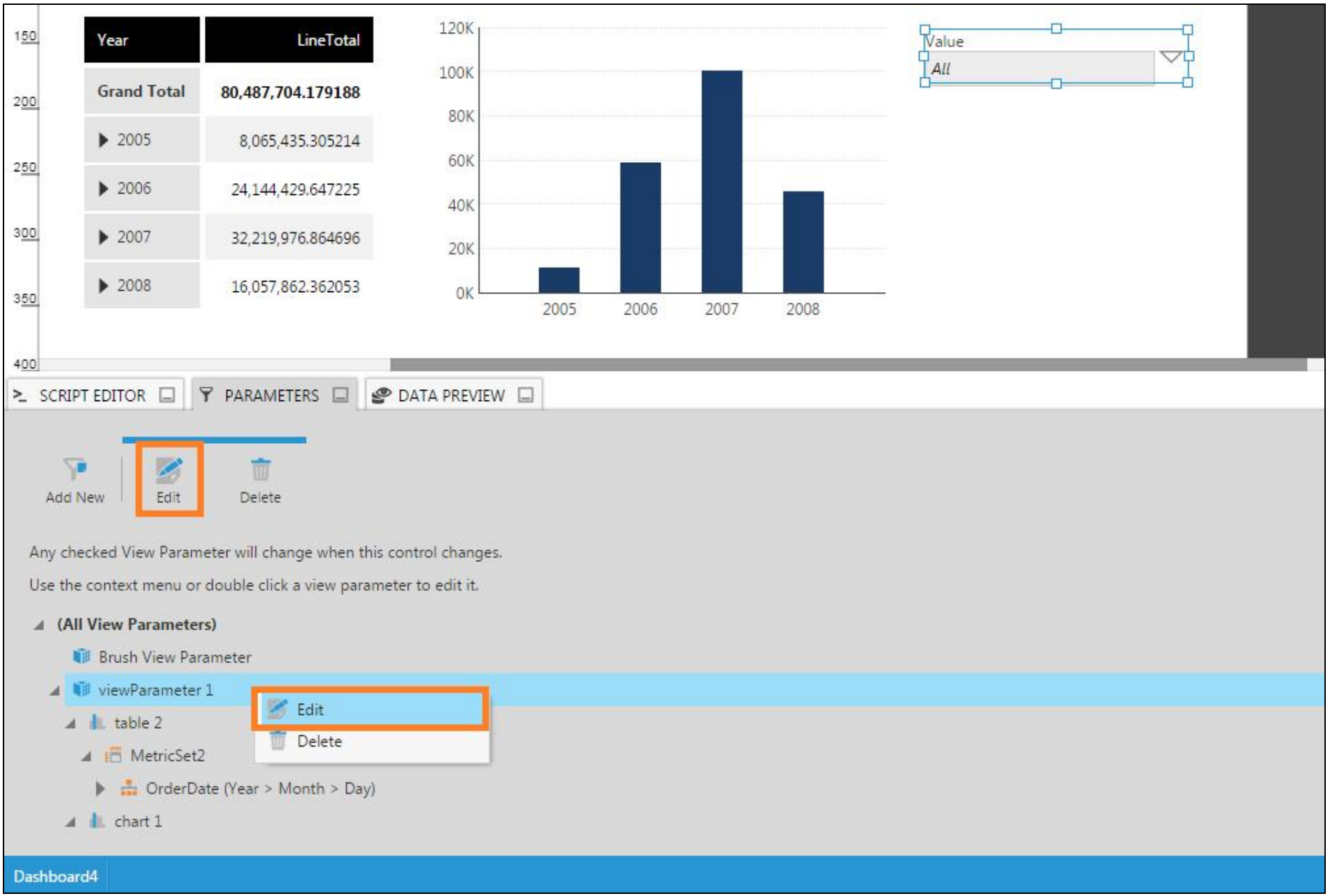


Compatible parameters from other metric sets will be automatically checked for you.

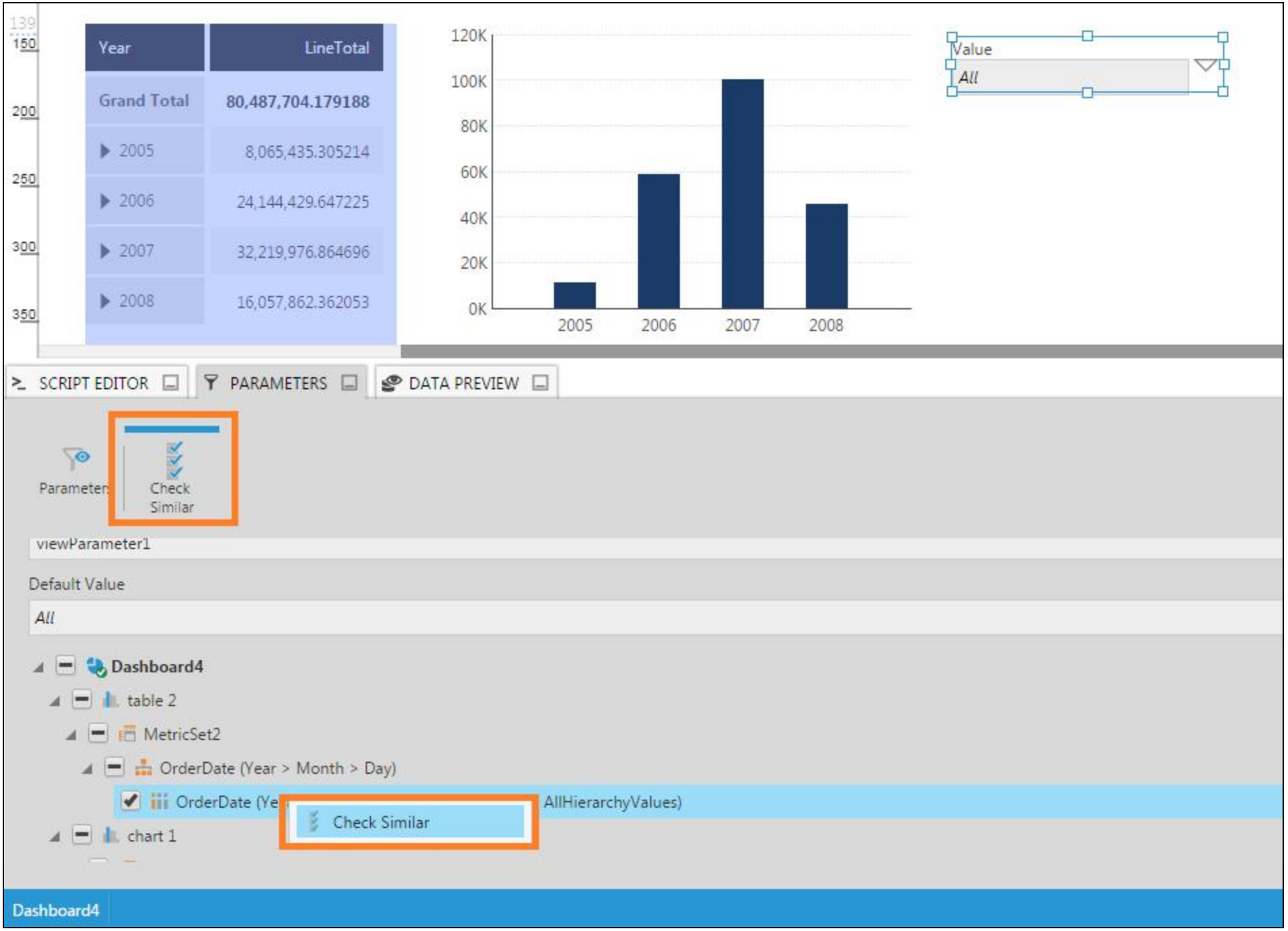
Using the Parameters Window

You can also perform the automatic connection from the **Parameters** window.

Locate your view parameter in the window. Right-click on the view parameter and select **Edit** from the menu, or select the view parameter and click **Edit** from the toolbar.


Year
LineTotal
Grand Total
80,487,704.179188
2005
8,065,435.305214
2006
24,144,429.647225
2007
32,219,976.864696
2008
16,057,862.362053
120K
100K
80K
60K
40K
20K
0K
2005
2006
2007
2008
Value
All
SCRIPT EDITOR
PARAMETERS
DATA PREVIEW
Add New
Edit
Delete
Any checked View Parameter will change when this control changes.
Use the context menu or double click a view parameter to edit it.
(All View Parameters)
Brush View Parameter
viewParameter 1
Edit
Delete
table 2
MetricSet2
OrderDate (Year > Month > Day)
chart 1
Dashboard4

Expand one of the hierarchies or measures you want to connect to until you see its associated parameter. Right-click on this parameter and select **Check Similar** from the menu, or select the parameter and click **Check Similar** from the toolbar.


Year
LineTotal
Grand Total
80,487,704.179188
2005
8,065,435.305214
2006
24,144,429.647225
2007
32,219,976.864696
2008
16,057,862.362053
120K
100K
80K
60K
40K
20K
0K
2005
2006
2007
2008
Value
All
SCRIPT EDITOR
PARAMETERS
DATA PREVIEW
Parameters
Check Similar
viewParameter1
Default Value
All
Dashboard4
table 2
MetricSet2
OrderDate (Year > Month > Day)
OrderDate (Ye
AllHierarchyValues)
Check Similar
chart 1
Dashboard4

Parameters from other metric sets that are compatible will be automatically checked for you.

For more information, see:

- Adding Filters
- Create a Metric Set and Add a Filter
- Design Overview

Using Code Libraries

This applies to: *Managed Dashboards, Managed Reports*

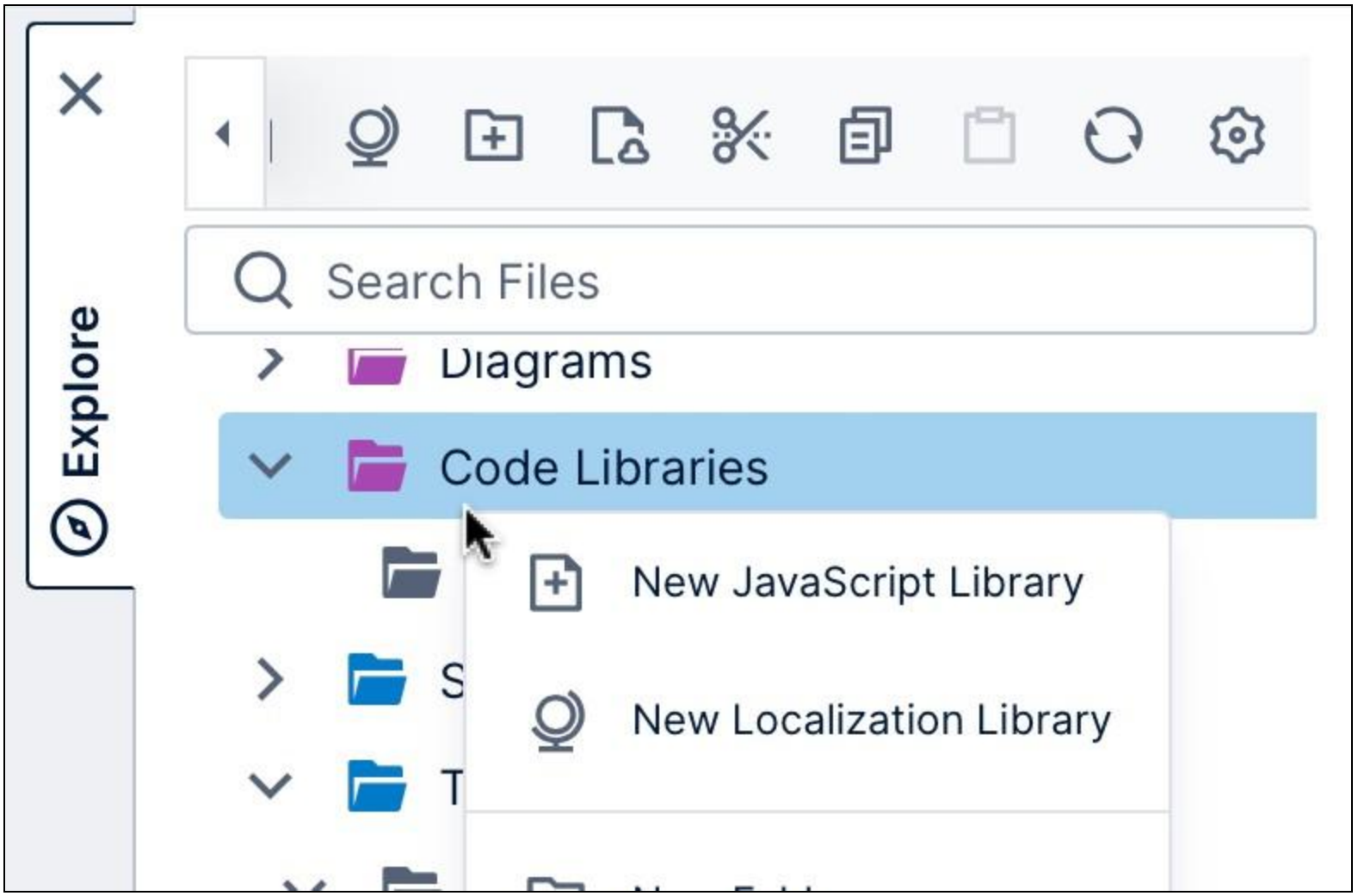
Code libraries allow developer users to create and manage reusable scripts and localizations to add to dashboards and other views.

For example, use JavaScript code libraries for helper functions that you want to call from multiple dashboards, or even to reference from other code libraries. Localization code libraries store localized strings in different languages that can be loaded and displayed automatically in your views depending on each user's culture.

Create a Code Library

You can create a code library while you have a file open for editing such as a dashboard or another type of view.

In the **Explore** window, right-click the **Code Libraries** folder, then choose **New JavaScript Library** or **New Localization Library**



In the dialog that opens, click into the **Name** field to name the library and confirm the save location in the Save dialog that appears next. Optionally create a new subfolder by right-clicking the **Code Libraries** folder and choosing **New Folder**, then typing a name and the enter/return key. Click to save at the bottom to return the code library dialog.

Adding content is different for the JavaScript libraries detailed next versus localization libraries described later.

JavaScript Libraries

Help

New Code Library

Create a new code library containing script or localized strings to use in views such as dashboards. Localization libraries are a container that you can create separate localizations inside of for each targeted culture.

Name

Select a name...

Content Type

JavaScript

References

Add reference

Upload a file to use for the content, or enter or paste it as text below.

Choose File no file selected

Build Undo Redo Editing: **JavaScript** Ln: 1 Col: 1

For a JavaScript library, click **Add reference** if you have other code libraries containing script that this code library should be able to refer to or should otherwise be included whenever this code library is used.

You can write or paste your script directly into the editor shown in the dialog, which has many of the same features as the script editor shown when editing a view, including auto-complete popups. You can also click **Choose File** or **Browse**... (depending on the browser) to upload a file with a .js file extension.

Code library JavaScript will run immediately when a view that references it is loaded, as a <*script*> element on its page. You can include any JavaScript that should be included on these pages, such as functions that should be called from script actions added to events in a dashboard. Below are some examples:

```
function mySimpleFunction() {
	return "Hello!";
}

window.corporateLibrary = {

	palette: ["#418cf0", "#fcb441", "#e0400a", "#056492", "#bfbfbf", "#1a3c69", "#e1e480", "#119cdd", "#cb6a49", "#005cdb",
"#f4d288", "#506381", "#f1b9a8", "#e0830a", "#7893be"],

	applyPaletteColors: function (colorRules) {
		var discreteRules = colorRules.filter(function (rule) {
			return rule instanceof dundas.controls.DiscreteColorRule;
		});
		discreteRules.forEach(function (rule, index) {
			var colorString = corporateLibrary.palette[index % corporateLibrary.palette.length];
			rule.value = dundas.controls.Color.fromString(colorString);
		});
	}
};
```

Click **Save** at the bottom of the dialog to save and check in the code library in the location you chose within the Code Libraries folder.

To edit the code library later, find it in the **Explore** window and double-click it, or right-click it and choose **Edit**.

Localization Libraries

A localization library acts as a container for the separate localizations you add for different languages/cultures, so you usually just enter a name in its initial dialog. Click **Save** at the bottom to save and check in the localization library in the location you chose within the *Code Libraries folder*.

Next, right-click your localization library in the **Explore** window, and for each language or culture you are entering text for, choose **New Localization**. Enter a **Name** for your own reference, then customize the default XML that's populated initially.

Help

New Code Library

Create a new code library containing script or localized strings to use in views such as dashboards. Localization libraries are a container that you can create separate localizations inside of for each targeted culture.

Name

Content Type

Localization

Upload a file to use for the content, or enter or paste it as text below.

Choose File no file selected

```
1 <?xml version="1.0" encoding="utf-8"?>
2 <localization xmlns="http://dundas.com/schemas/BI/customLocalization.xsd" cu
3   <group name="General">
4     <string key="LabelName1">This is a localized label</string>
5   </group>
```

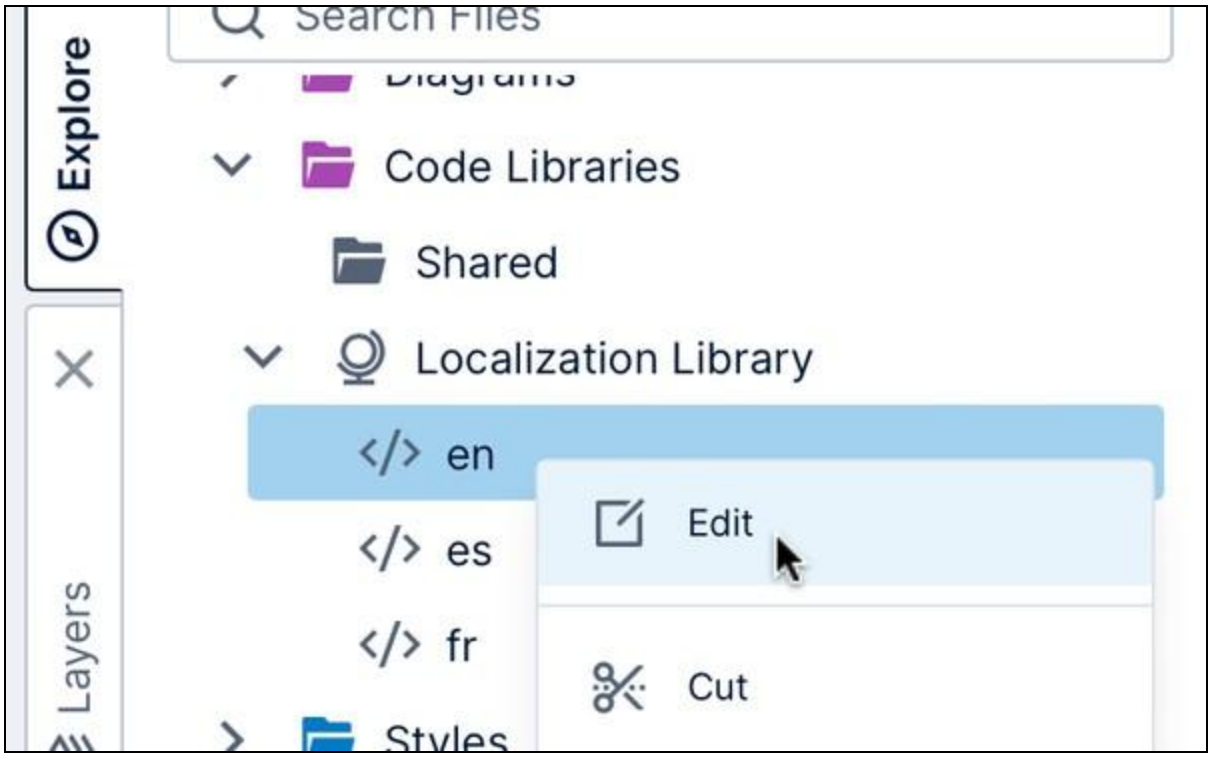
The default XML consists of a *localization* element like the following where the *culture* attribute is initially set to en, representing English for no particular culture. Change this to another culture name or language tag depending on which localization you are adding. An extensive list of the ones you can enter is available from Microsoft.

Use a simple two-letter language code when you want one localization to apply for multiple cultures, for example en applies to users to cultures *en-US* (United States), *en-GB* (United Kingdom), *en-AU* (Australia), and others. You can add a mix of a localizations with and without specific culture/region codes, and the most specific match to each user's culture will be loaded if available.

```
<?xml version="1.0" encoding="utf-8"?>
<localization xmlns="http://dundas.com/schemas/BI/customLocalization.xsd" culture="en">
       <group name="General">
              <string key="LabelName1">This is a localized label</string>
       </group>
</localization>
```

Add *string* elements with a key attribute setting whatever name you want to use to refer to this string, then enter the localized text inside the element that you want displayed. You can add them inside group elements like in this example – these are optional and used for your own convenience to maintain some organization for the strings.

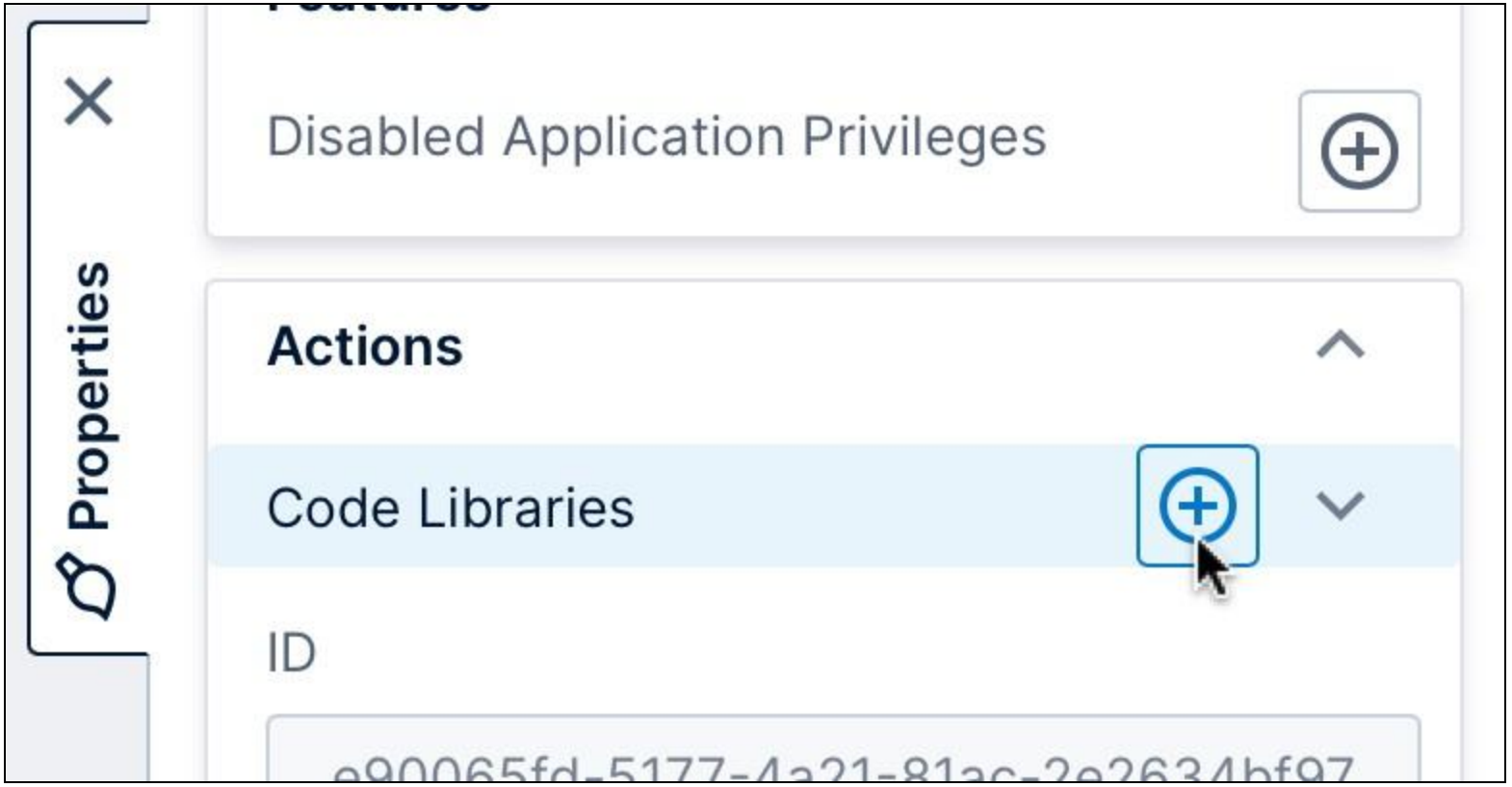
After clicking **Save**, you will find the localization under the localization library after expanding it. Right-click the localization and choose **Edit** when you want to re-open the dialog.



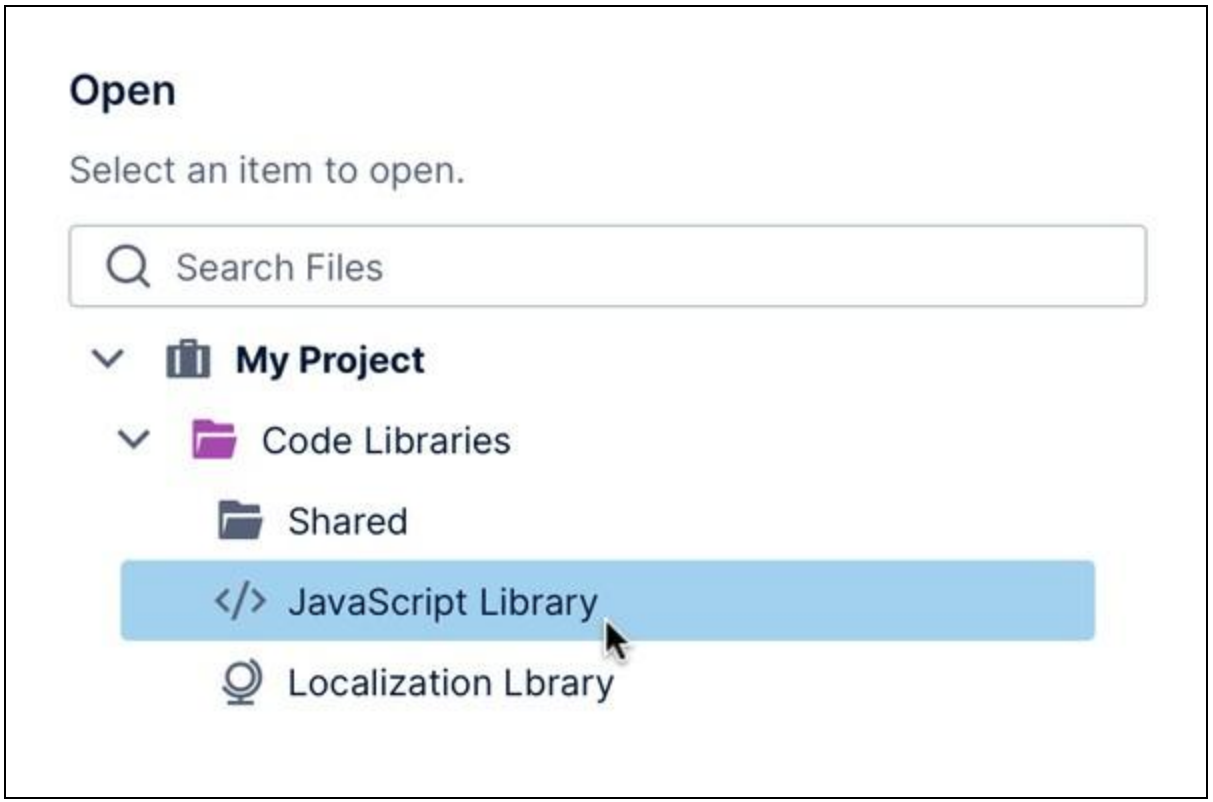
Using a Code Library

When editing a dashboard, report, or other view, you can add a code library in the **Properties** window for that view. If any element or control is selected, click an empty part of the canvas to de-deselect it and show the view's own properties instead.

Under the *Actions* category, expand **Code Libraries** then click the **+** button.

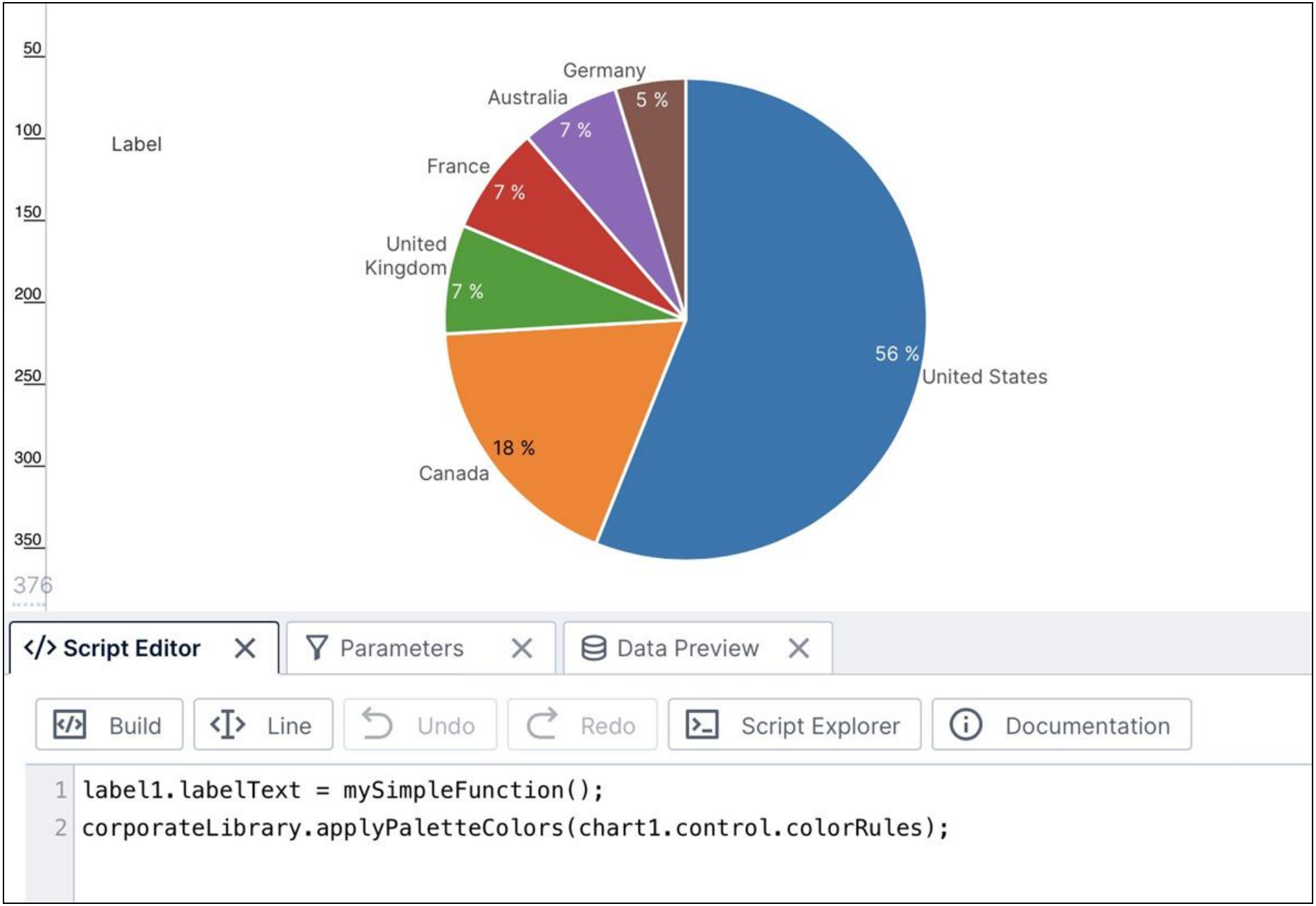


In the *Open dialog* that appears, select the code library that you want to use and click the submit button at the bottom.



JavaScript Libraries

Code library script often needs to be accessed or called from script actions. Using the script editor accessed for an event such as *Loading or Ready* on the dashboard or view itself, or *Click* on a visualization or other control, you can now refer to the code library's script.

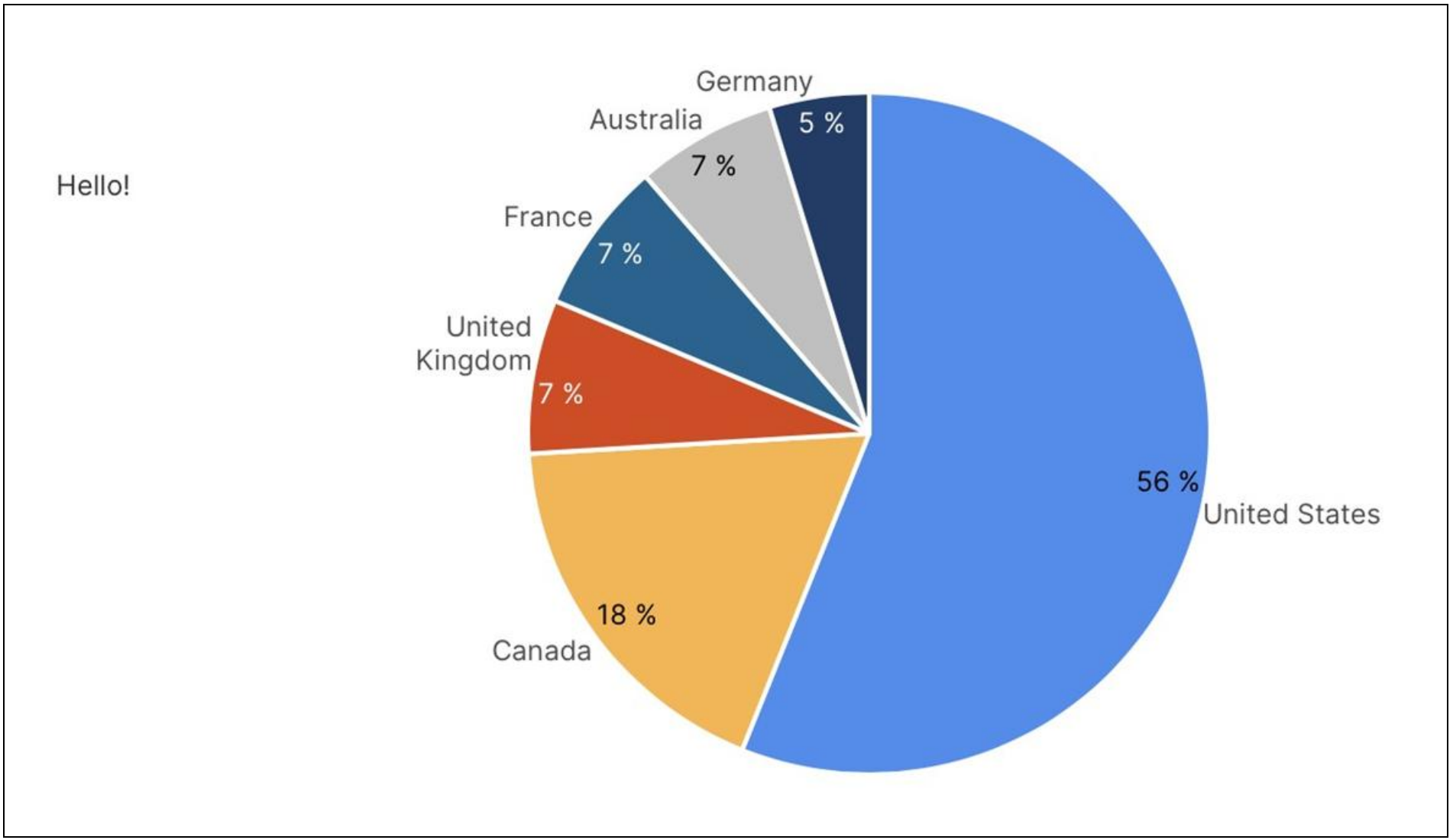

50
100
150
200
250
300
350
376
Label
Germany
Australia
5 %
7 %
France
7 %
United Kingdom
7 %
56 %
United States
18 %
Canada
Script Editor
Parameters
Data Preview
Build
Line
Undo
Redo
Script Explorer
Documentation
1 label1.labelText = mySimpleFunction();
2 corporateLibrary.applyPaletteColors(chart1.control.colorRules);

JavaScript libraries added to a view are included on the page when that view is first opened. Click Sandbox View in the toolbar to immediately open the dashboard in view mode in a new tab or window with all JavaScript libraries loaded and scripts running.

Note: If your code library script is causing a problem with your view even in edit mode, you can add ? I*gnoreBuiltInResources=true* to the URL in your browser's address bar (or *&IgnoreBuiltInResources=true* following other parameters).

Note: When using a view container to embed a view, JavaScript libraries from both views will be added to the page.

You can also re-open the current file or refresh the page for any newly added JavaScript libraries to be loaded, but note that script actions only run in view mode, and dashboards must be opened directly into view mode for script actions to run.



Localization Libraries

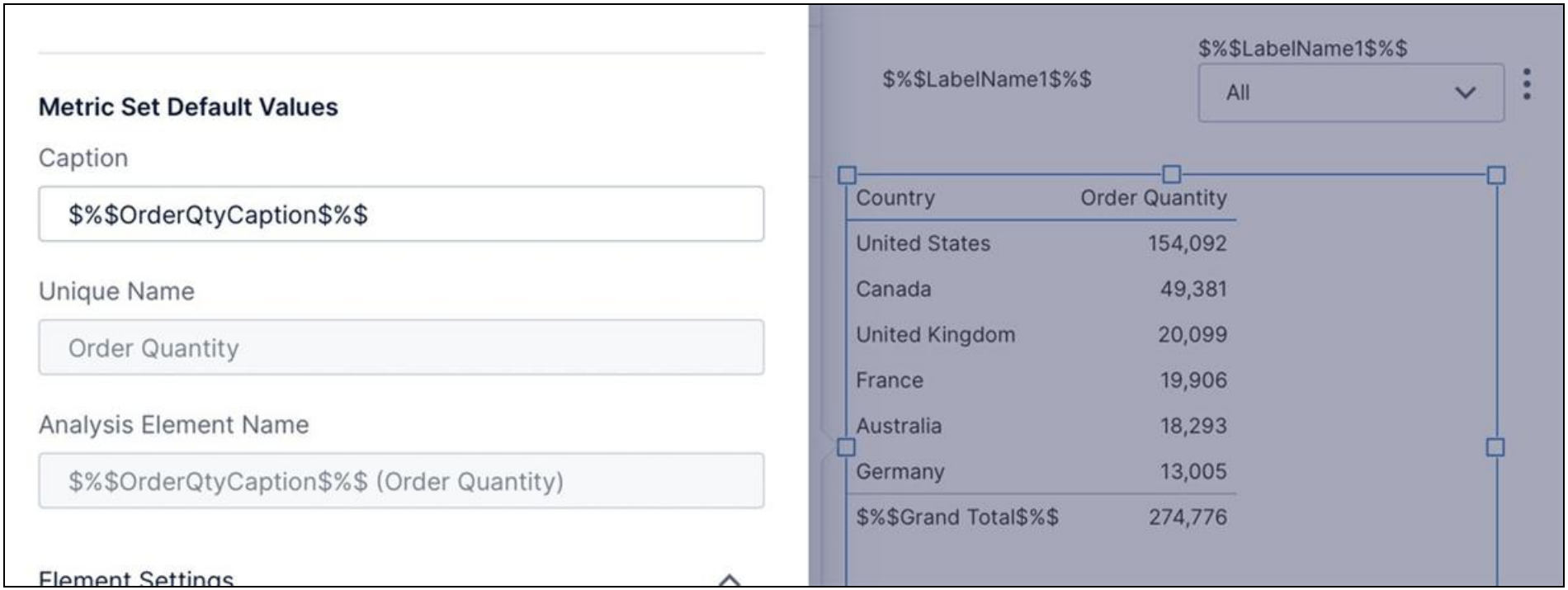
With your dashboard or other view open in edit mode, you can 'globalize' any displayable text setting by entering one of the keys you specified in your localization library surrounded on either side by the characters $%$.

For example, to use the default localization string from the example shown earlier, enter the following as a label's text:

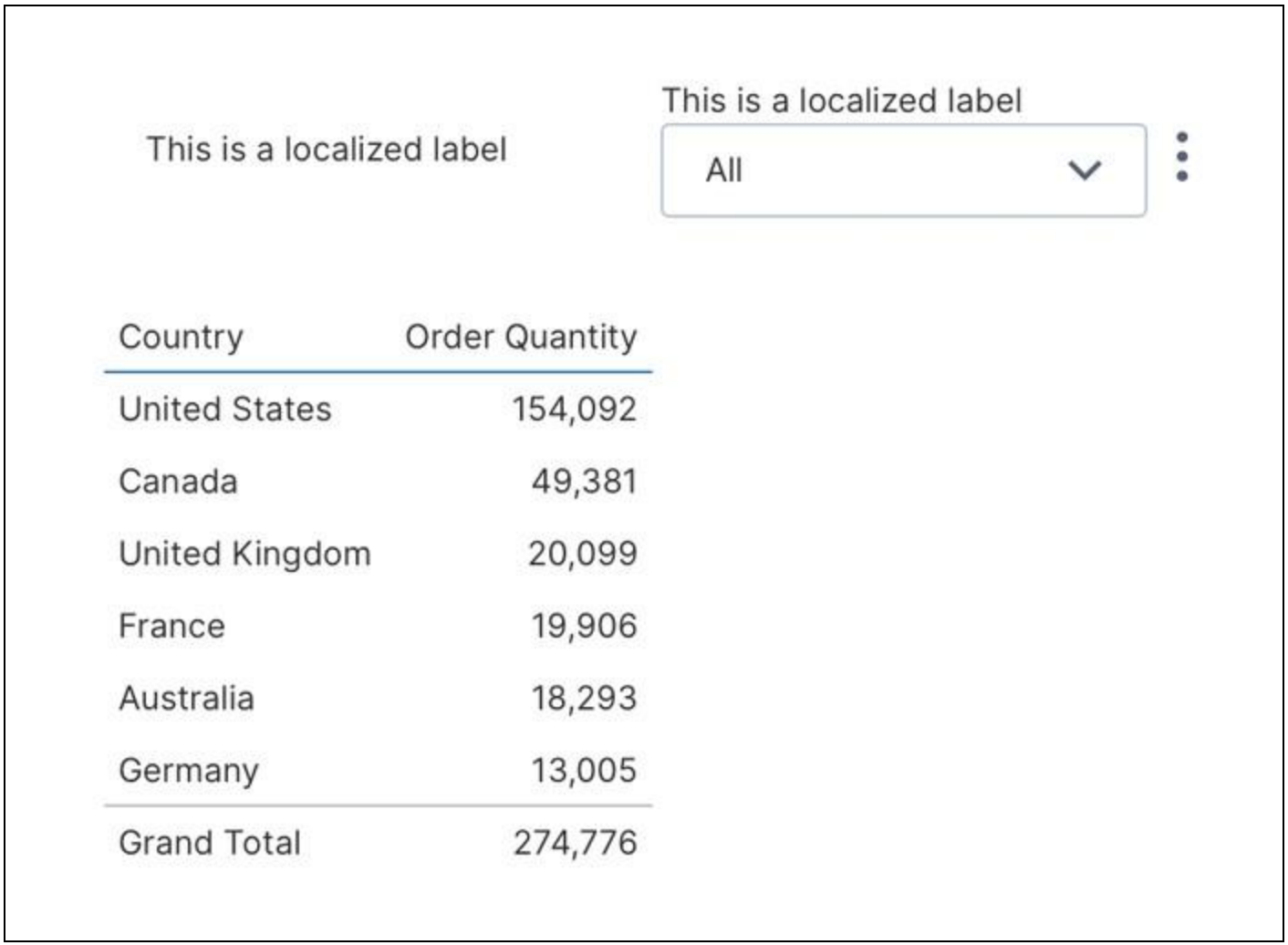
```
$%$LabelName1$%$
```

Any other text set up for a visualization, component, or filter in the **Properties** window can be set the same way, such as **Tooltip Text**, a column's **Header Text** in a table, a filter's built-in label text, and so on. You can combine your localization key with any other text characters or placeholder keywords you want displayed.

You can also localize settings in the metric sets you've added to your view. For example, when you want to localize text that comes from the names (captions) of the measures and hierarchies in your metric set, the best way is to edit their settings from the Data Analysis Panel and set their Caption to use your localization key. This way, your localized text will be displayed by default everywhere the data is assigned in the Data Analysis Panel's Visualization tab. There is also a Name when editing the overall metric set settings that can be displayed to viewers when setting up some notifications, for example.

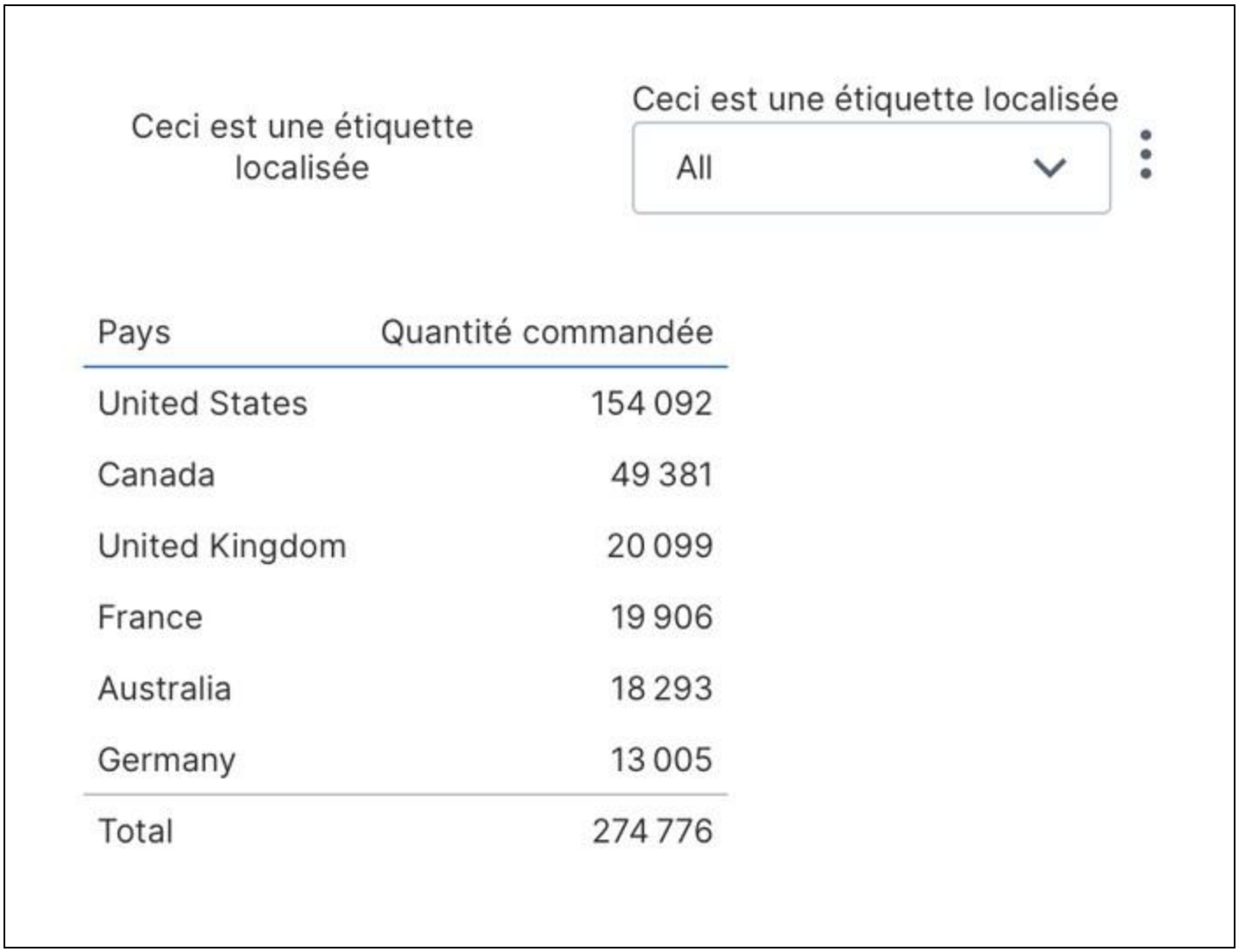


In general, your localization keys will be displayed in edit mode as you entered them. Switch to **View** mode from the toolbar to see your localized text take effect for your own culture (you can check your current culture from your profile).



To view the dashboard with a different culture, you can either log on as a user with a different culture set up, or you can add *?cultureName=<culture>* to the logon URL with a culture name that corresponds to one of your localizations. For more details on culture names and different ways you can set them, see Localization and Multi-Language Support.

The following shows the result when viewing the same dashboard as in the previous two images above, with a localization added for culture fr (French) and after logging on with the culture name fr-FR (French - France):



Code libraries are used only for settings that are set directly while editing views such as dashboards and reports (and their metric sets). Other features can be localized separately:

- The entire application's user interface or any required portion of it can be localized by administrators as shown in Localization and multi-language support.
- Tokens shown in filters such as *All or Current year* can be localized for various languages/cultures when editing the token's name or caption.

- Symphony does not perform localization of data from your data source (i.e., rows of data or data points) on its own, but a data source may localize the data for each user if impersonation is supported and set up in its data connector. In a data cube, a calculated element's script can access the current session's culture name to determine data to return, or Current Culture Name is one of the built-in attribute tokens that can be passed as a filter parameter, query placeholder, or to a stored procedure.

For more information, see:

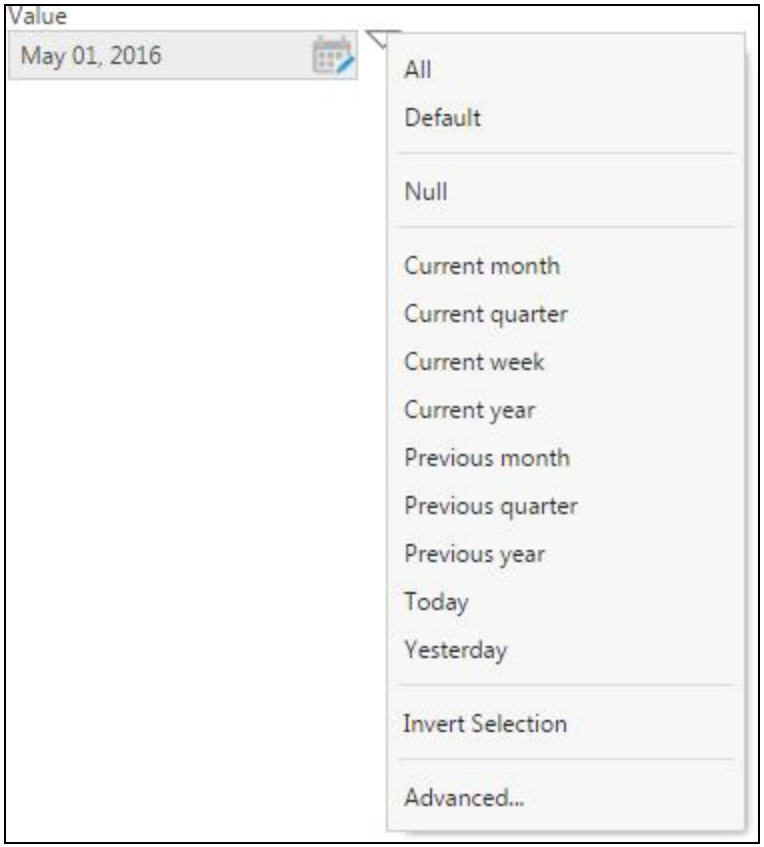
- Using the Script Editor
- Localization and Multi-Language Support
- Edit Versus View Mode and Sandbox View
- Design Overview
- Seat Types

Creating Custom Tokens

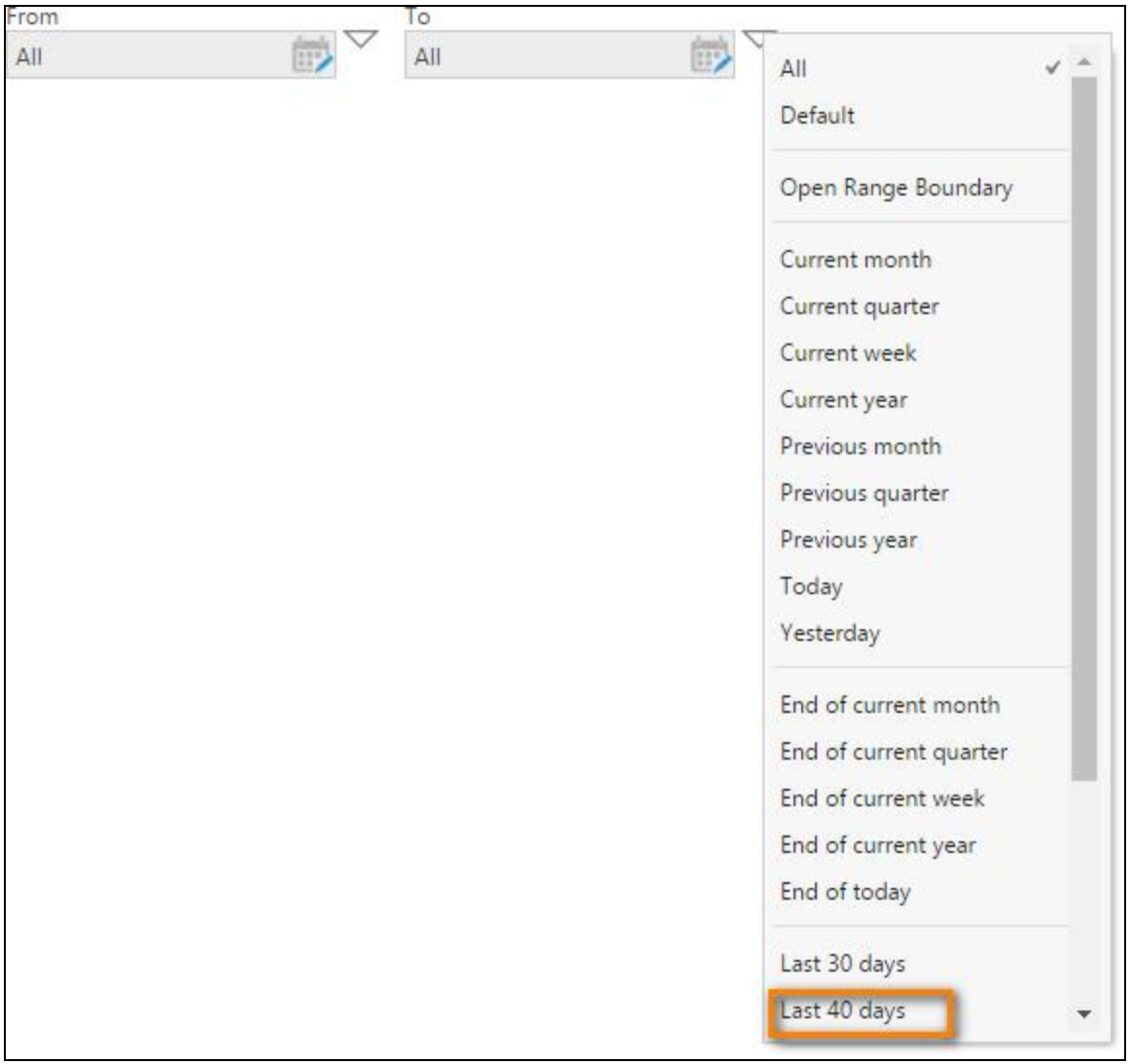
This applies to: *Managed Dashboards, Managed Reports*

Custom tokens can be added to the tokens menu of filters to provide your own filtering options based on a data source, a predefined set, a time offset, or script.

Built-in token options include **All** and **Null**, and various relative time tokens shown for date/time and calendar filters such as **Today**, **Current week**, or **Year to date**.



All Symphony users with the Create Token privilege can create new tokens.



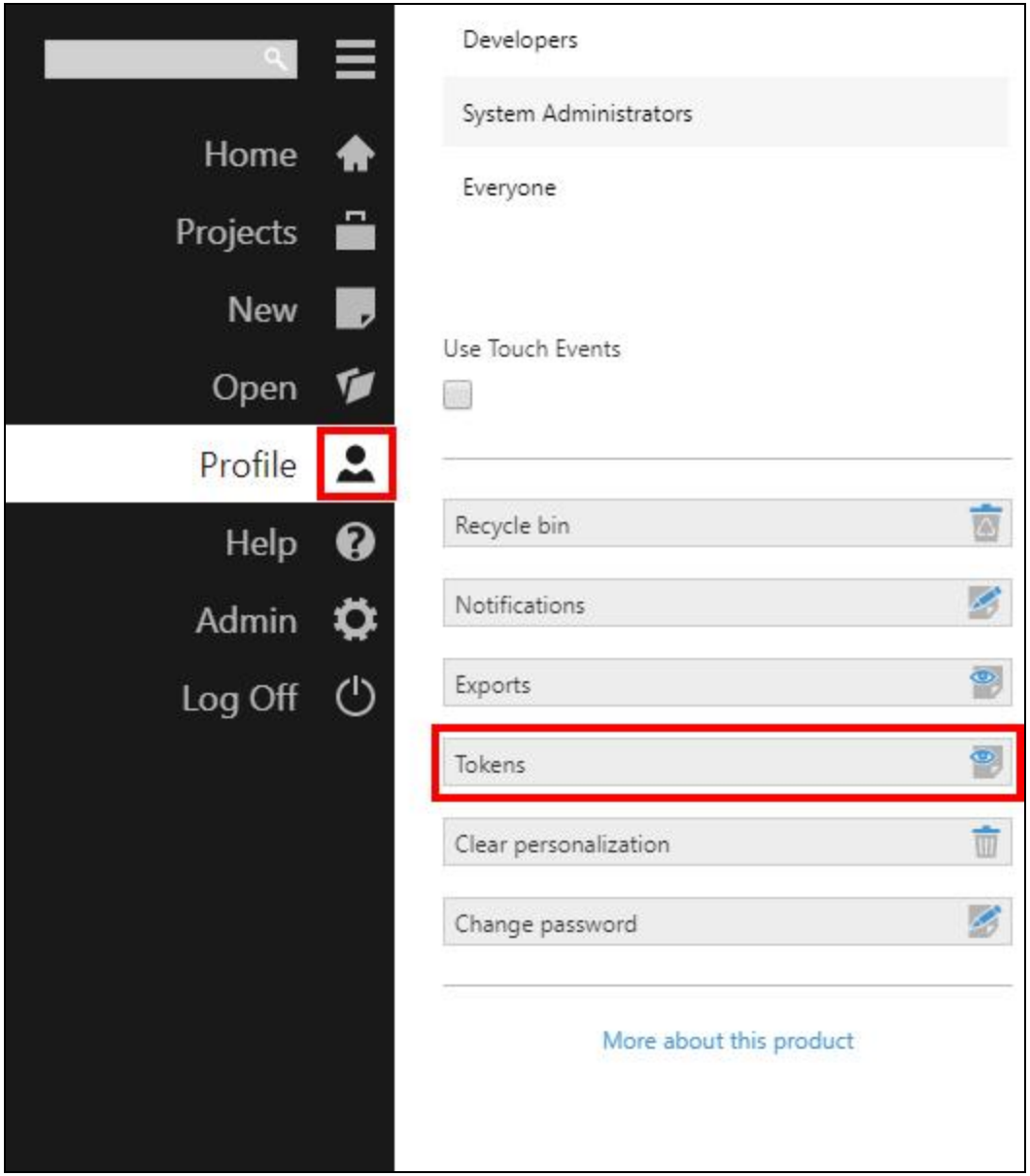
The various types of custom tokens you can create are:

Token Category	Description	Example
Data	Defined by querying the raw data behind a data source	First day, Largest order
Named Set	Defined on a specific dimension hierarchy	USA, Canada, Paris

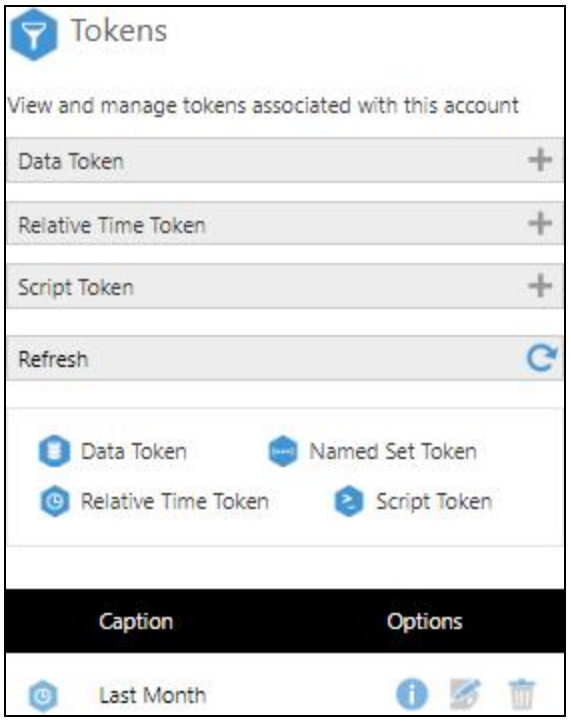
Token Category	Description	Example
Relative Time	Defined on time based parameter	Current week, Last 30 days, Beginning of today
Script	Defined using DundasScript , allowing unlimited flexibility	return TimeZoneInfo.ConvertTimeFromUtc(DateTime.UtcNow, currentSession.TimeZone);

Tokens List

To find a list of your existing tokens and to create new data, relative time, or script tokens, click the **Profile** button in Symphony's left menu, and scroll down to find **Tokens**.



Click the action icons next to existing tokens to view details, edit, or delete.

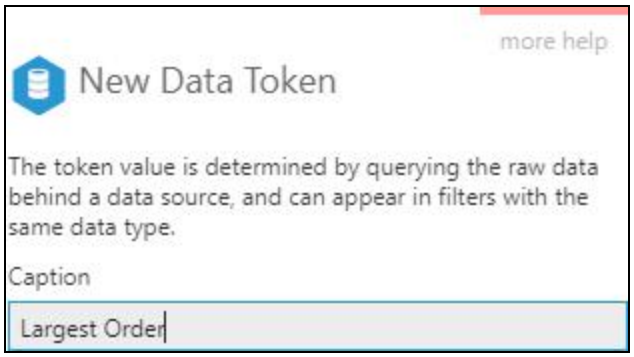


Creating New Tokens

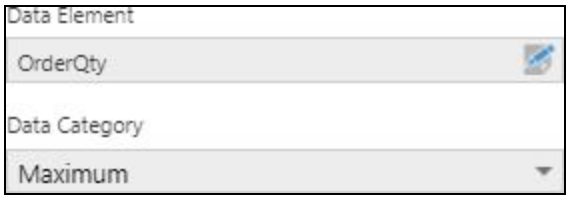
Data Tokens

In the tokens list found in your Symphony profile, click the **Data Token** button to create a new data token.

In the **New Data Token** dialog, enter a **Caption** for the token.

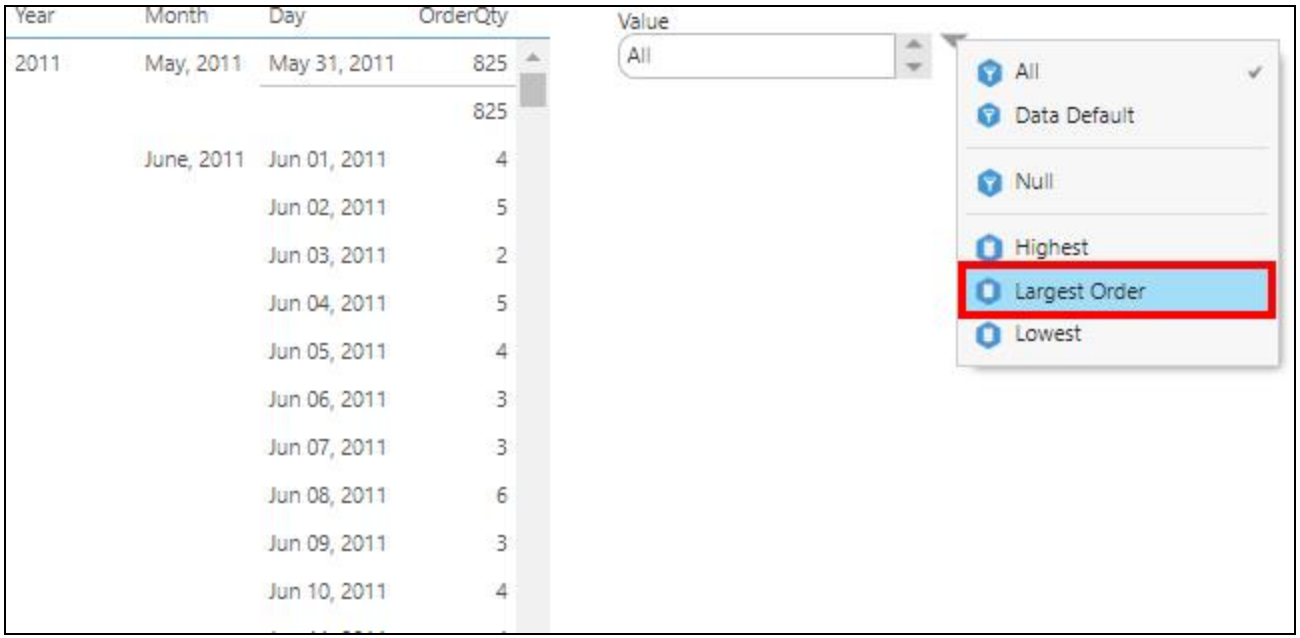


Select a column or element from a data source or cube as the **Data Element** for this token, then choose a **Data Category** for determining which value to use.



Click **Submit**.

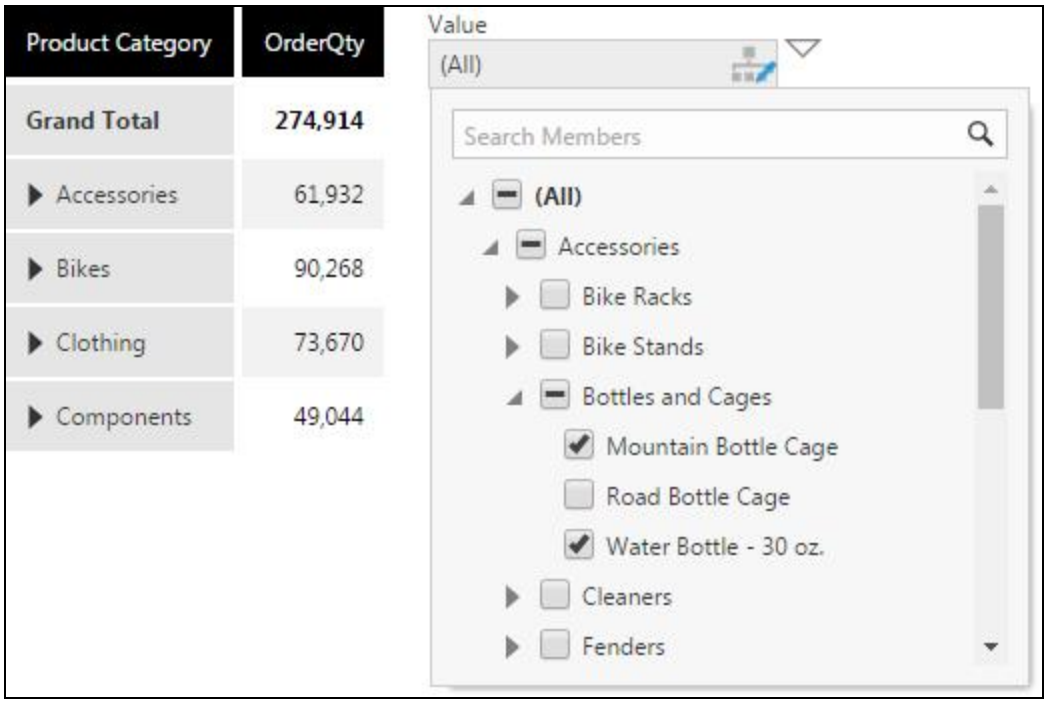
The token will appear on filters that use the same data type as the data behind the element you chose.



Note: If you choose a hierarchy from a data cube as your element, the token's data type is **String** and can only appear for string parameter types rather than hierarchy parameters.

Named Set Token

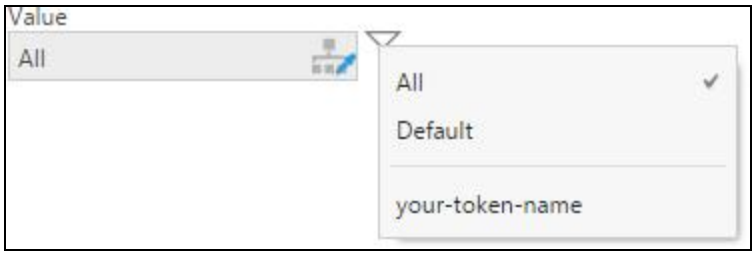
To create a new named set token, start by selecting hierarchy member values from a filter in a metric set, or in a dashboard or other view.



From the hierarchy filter's token menu, select **Save Selection as Token....**

Enter a **Caption** and click the submit button at the bottom of the dialog.

You can now find this token in the filter's token menu so you can easily re-select these values.

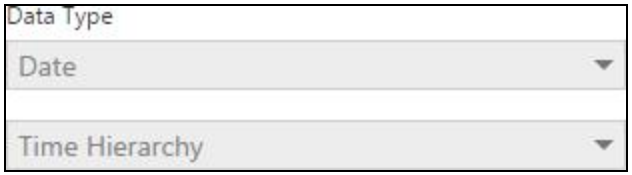


Relative Time Token

In the tokens list found in your Symphony profile, click the **Relative Time Token** button to create a new relative time token.

In the **New Relative Time Token** dialog, enter a **Caption** for the token.

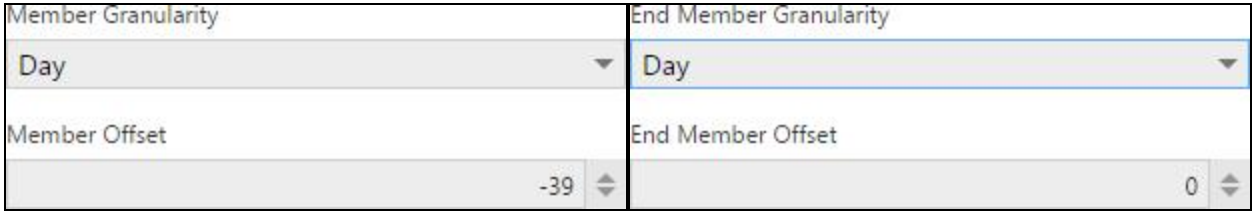
For a relative time token, the data type is locked to **Date** and **Time Hierarchy**.



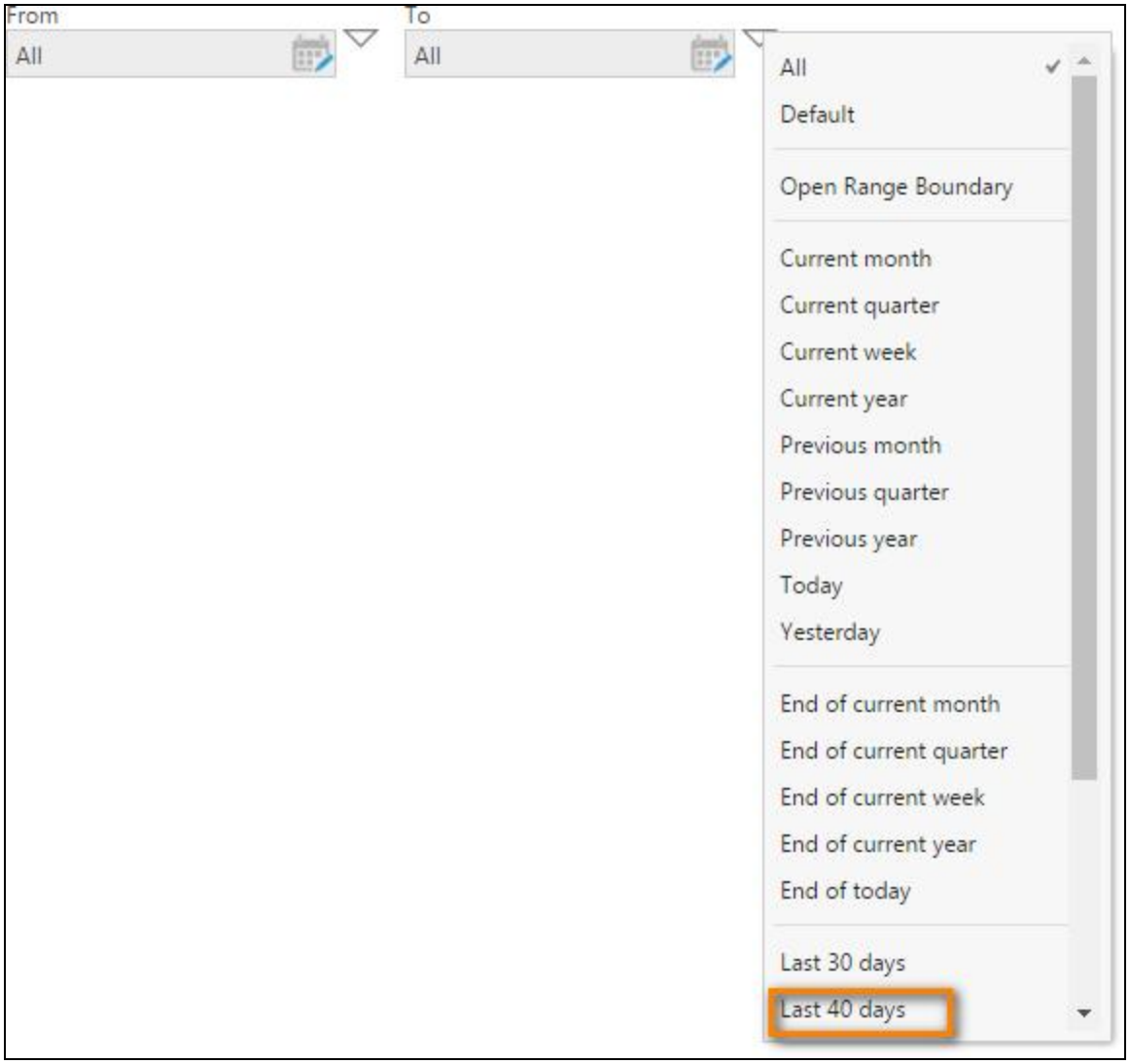
Select the **Display Type**. For our "Last 40 days" example, we select **Range Full** since it defines both a start and and end to a filter range.

If desired, set **Format Override** to display date values in a particular format in filters with this token selected.

Depending on what Display Type was chosen, select granularity and offset for the start and/or end of the range.



Click the submit button to create the token and return to the updated tokens setup screen. The new token can now be found in the tokens menus for calendar and date/time filters.



Script Token

In the tokens list found in yourSymphony profile, click the **Script Token** button.

Enter a **Caption**.

The script language used for tokens is DundasScript. Set **Data Type** to the type of value that will be returned by your script:

- Date (DateTime)
- Time Hierarchy (MemberValue or DateTime)
- Numeric (double, float, decimal, int, etc.)
- String
- Hierarchy (MemberValue)

Most data types support more than one **Display Type**, which determine which filter types can display this token:

- Single (e.g., Single Date, Single Number filters)
- Range Full, Range Start, Range End (e.g., Calendar Range, Range Number filters)
- Collection (e.g., Hierarchy Value, Cascading filters)

Some additional options may appear depending on what you chose above:

- If you chose a **Hierarchy** data type, click the **Select Hierarchy** link that appears to choose which hierarchy's members you will be returning.
- To allow your token to appear in the **Token Offset** dialog available as the **Advanced** option in filter token menus, check **Allow offset support as script parameter** if applicable. You can access the offset in your script as the variable **offset**.
- For tokens with a date or time hierarchy data type, optionally set the **Format Override** to determine how the dates are displayed in filters.

Your script should include one or more return statements, returning a value of the data type selected above. When returning a full range or a collection of values, return an array or another type of list of values, instead of just a single value of your data type.

For example, the following script returns a range from April 1, 2017 to today, where the end date is **exclusive**:

```
DateTime[] dates = new DateTime[2];
dates[0] = new DateTime(2017, 4, 1);
dates[1] = DateTime.Today;
return dates;
```

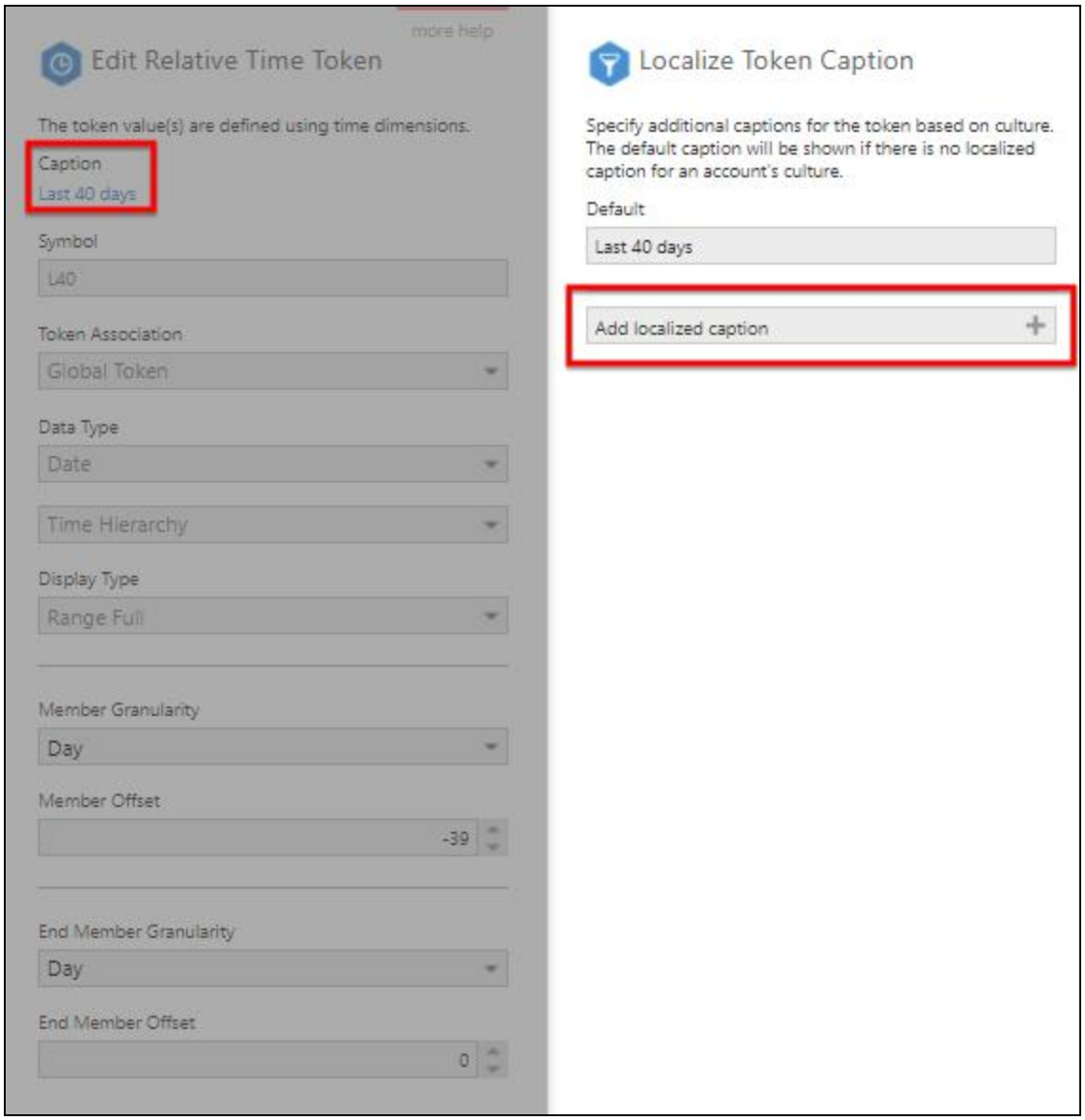
Click the **submit** button at the bottom of the dialog to verify the script and save the token. You will not be able to change the Data Type, Display Type, or hierarchy after this step.

Rename or Localize a Token

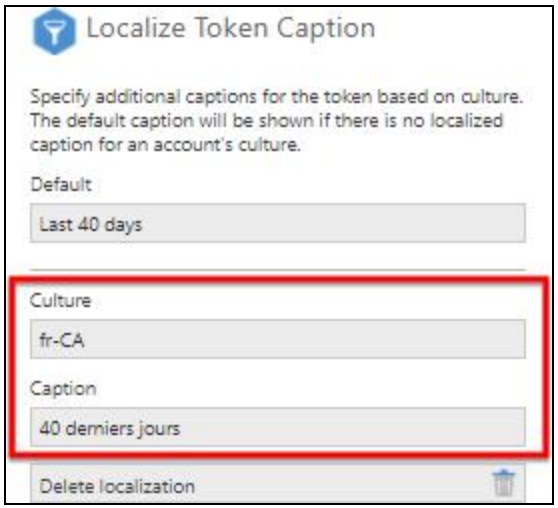
You can rename a token, or specify different captions for a token to be displayed depending on the current language/culture of your user account. First locate the token in the tokens list, select it and click the **Edit** icon.

Click the caption link at the top of the edit dialog.

To rename the token, change the text under **Default**; otherwise, click **Add localized caption**.



For a localized caption, enter the IETF language tag in the **Culture** field, and fill in the **Caption** field.



Click the **submit** button at the bottom of the dialog to apply the changes.

For more information, see:

- Adding Filters
- Application Privileges
- Managing Tokens
- Off the Charts: How to Create Custom Tokens

Filter Child Dashboards From a Parent Dashboard

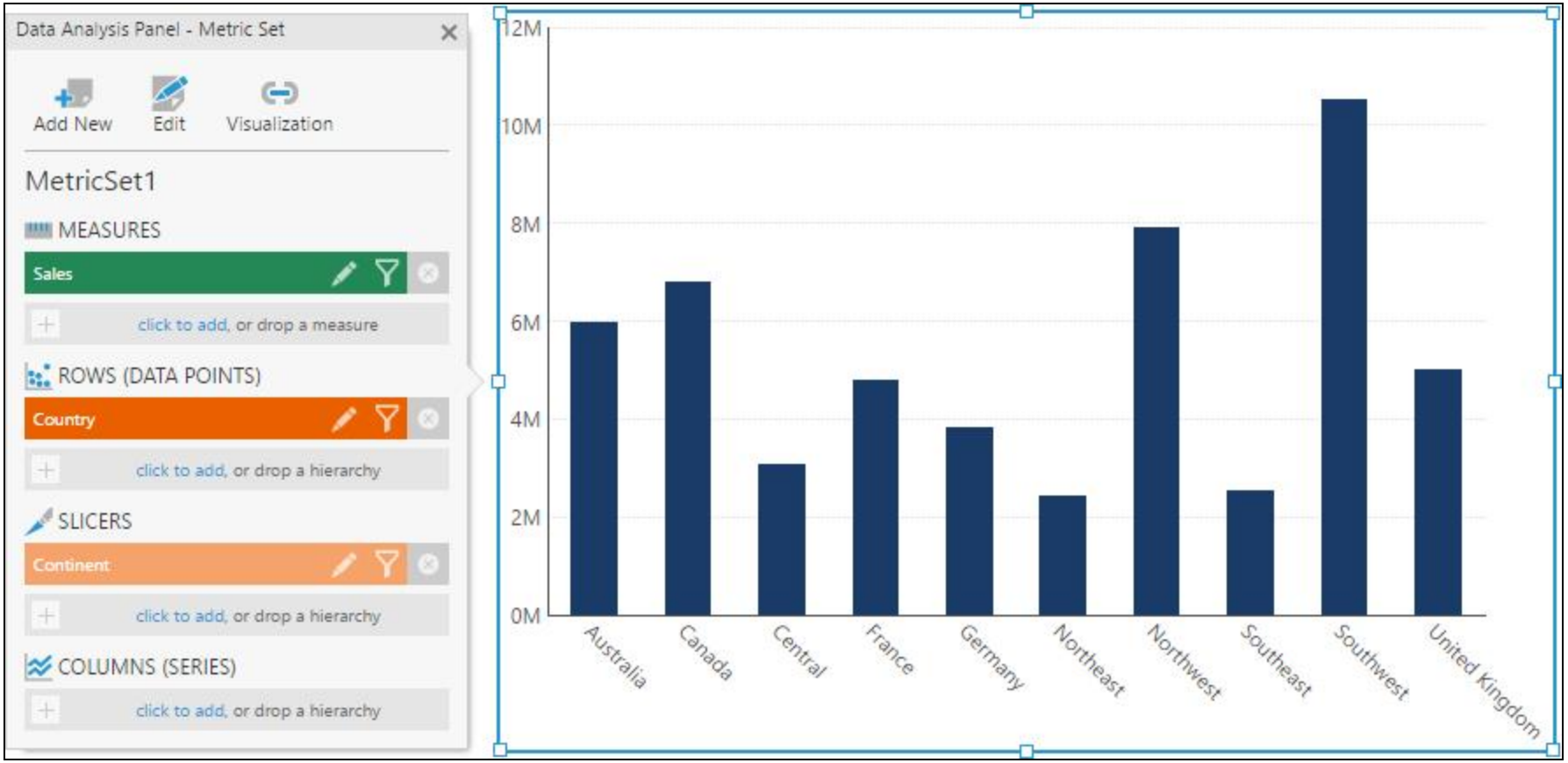
This applies to: *Managed Dashboards, Managed Reports*

You can embed views such as dashboards inside other views, and also filter them from the top level. For example, you can display one dashboard inside another dashboard, while using filters on the parent dashboard that affect the embedded dashboard.

Note: There may be some overhead when embedding dashboards as compared to using metric sets directly on the parent dashboard.

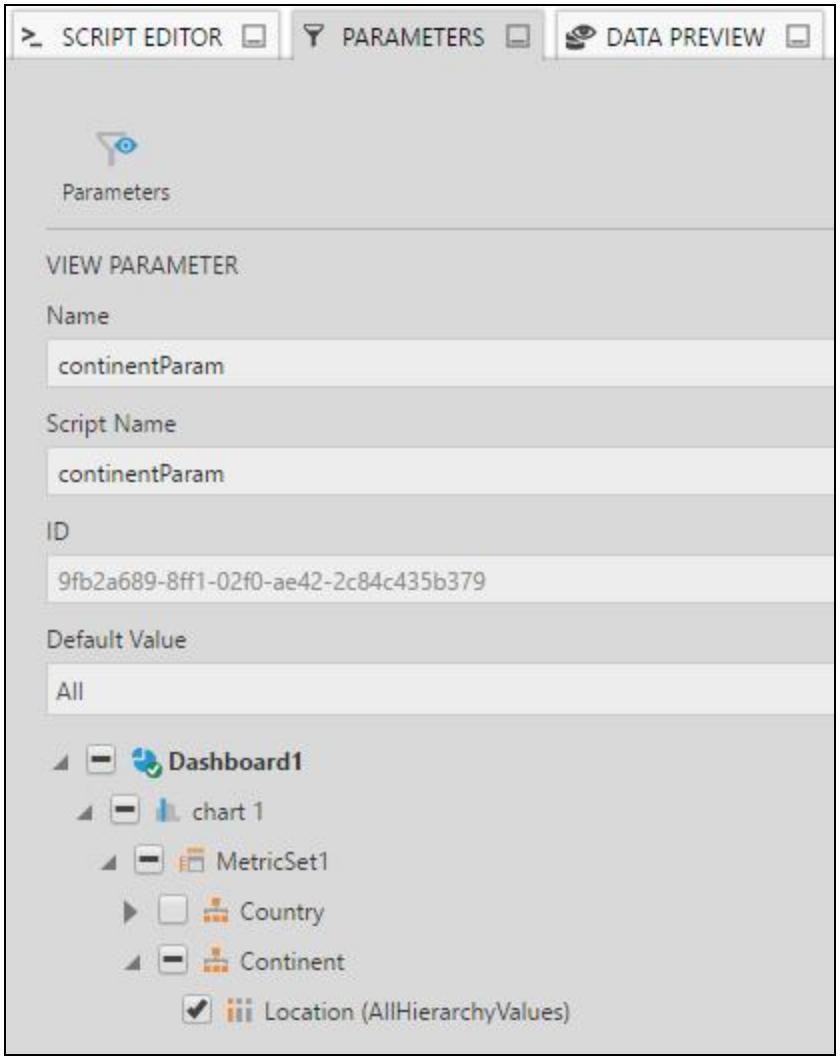
Set Up the Embedded View

Create or edit a dashboard or another type of view.



Add a view parameter in the **Parameters** window, or add a filter, which automatically adds a view parameter, then select checkboxes for data on this view that should be filtered.

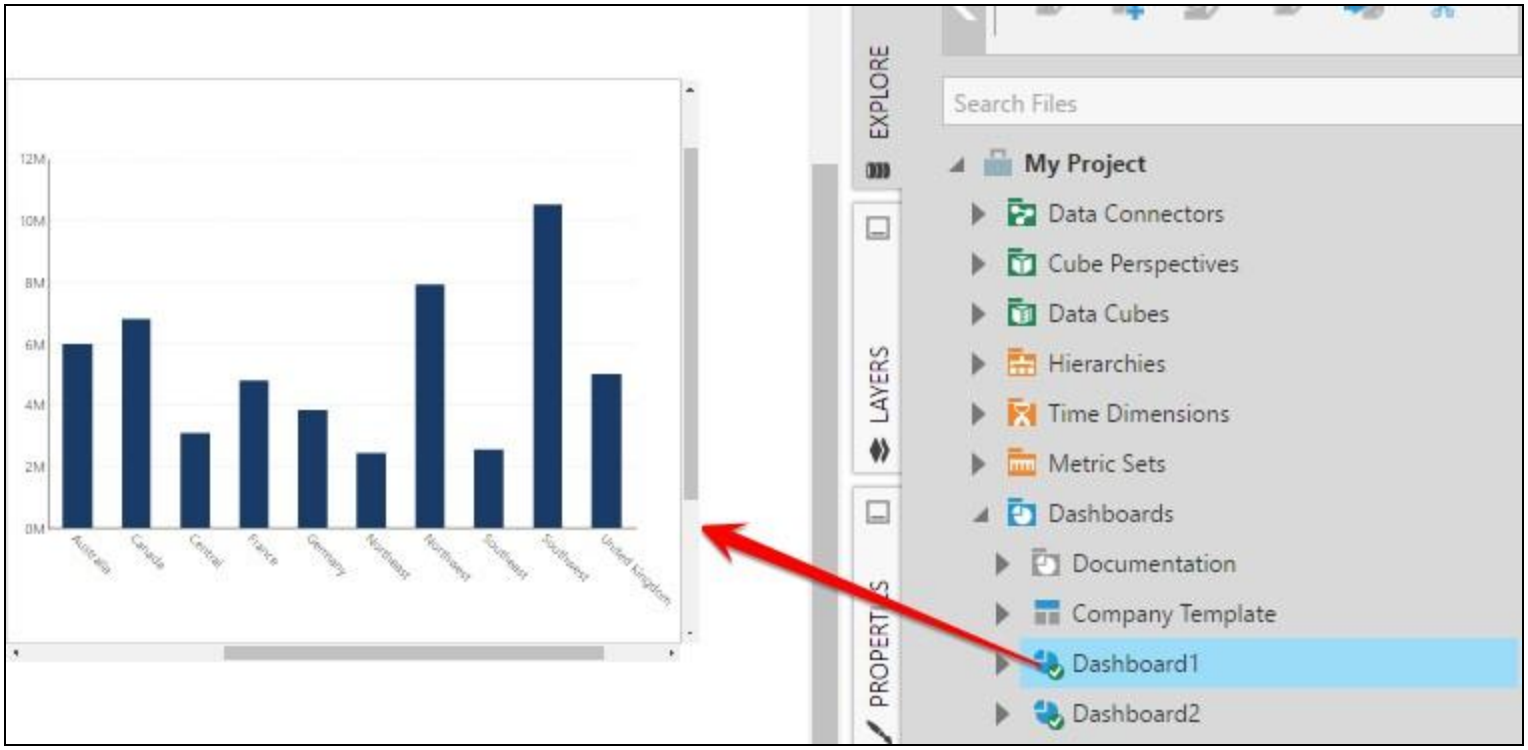
Customize the **Name** and **Script Name** depending on which data is being filtered.



If you will be navigating from this embedded view to another one, for example using a menu component on a parent view, or by setting up a navigation interaction in this view, your view parameter connections will continue to work as long as the view parameters in the other embedded views have a matching **Script Name** set.

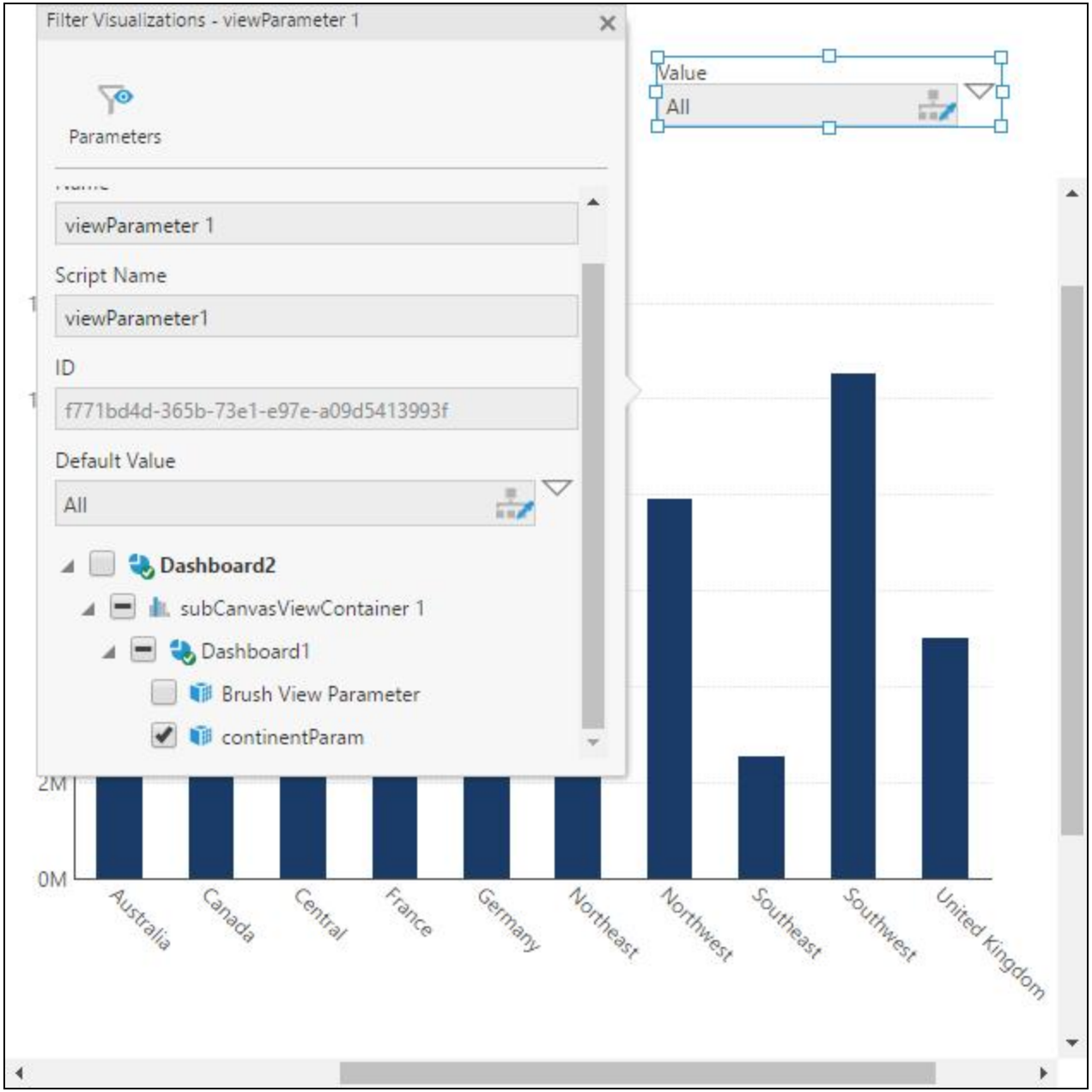
Embed the View

In another view such as a dashboard, drag and drop the view that you want to embed from the **Explore** window onto the canvas. This creates a view container to display that view.



Add a filter to use for filtering the child dashboard, or set up existing filters by choosing **Connect Filters** in the toolbar.

In the **Filter Visualizations** popup that appears, select the checkbox to connect to the child dashboard's view parameter that you created earlier.


Filter Visualizations - viewParameter 1
Parameters
viewParameter 1
Script Name
viewParameter1
ID
f771bd4d-365b-73e1-e97e-a09d5413993f
Default Value
All
Dashboard2
subCanvasViewContainer 1
Dashboard1
Brush View Parameter
continentParam
Value
All
2M
0M
Australia
Canada
Central
France
Germany
Northeast
Northwest
Southeast
Southwest
United Kingdom

Note: If you added filters to the embedded view, you can prevent them from displaying while embedded by enabling the **Hide Parameter Controls** checkbox option in the **Properties** window for the view container.

For more information, see:

- Using a View Container
- Use a Dashboard in a Report
- How To Set Up Menu Navigation
- Set Up a Navigate Interaction
- Design Overview

Using a Cascading Member Filter

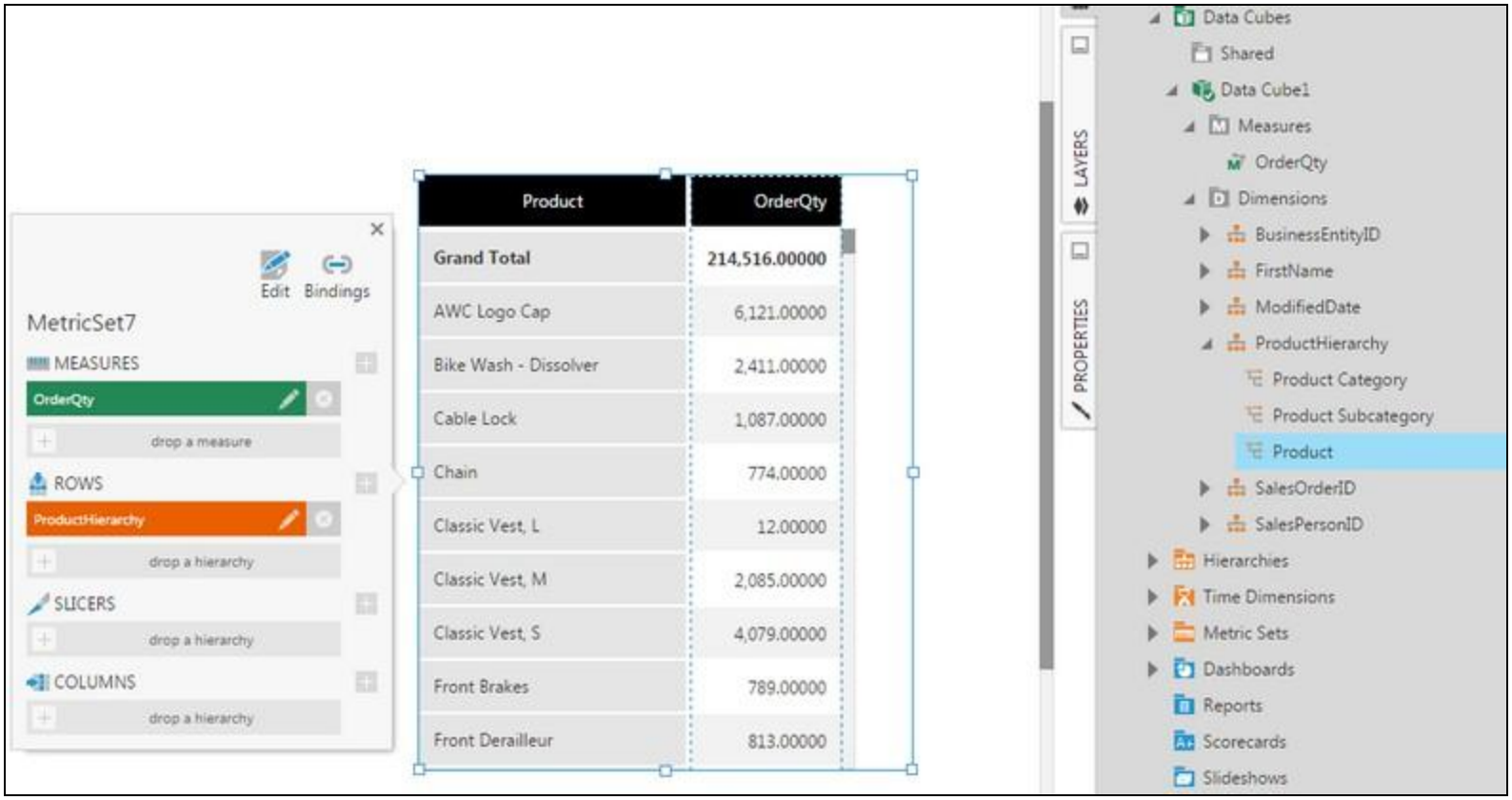
This applies to: *Managed Dashboards, Managed Reports*

A cascading filter allows you to select values from each level of a hierarchy from separate drop-down lists. The options in each drop-down are filtered based on what was selected in the previous drop-down.

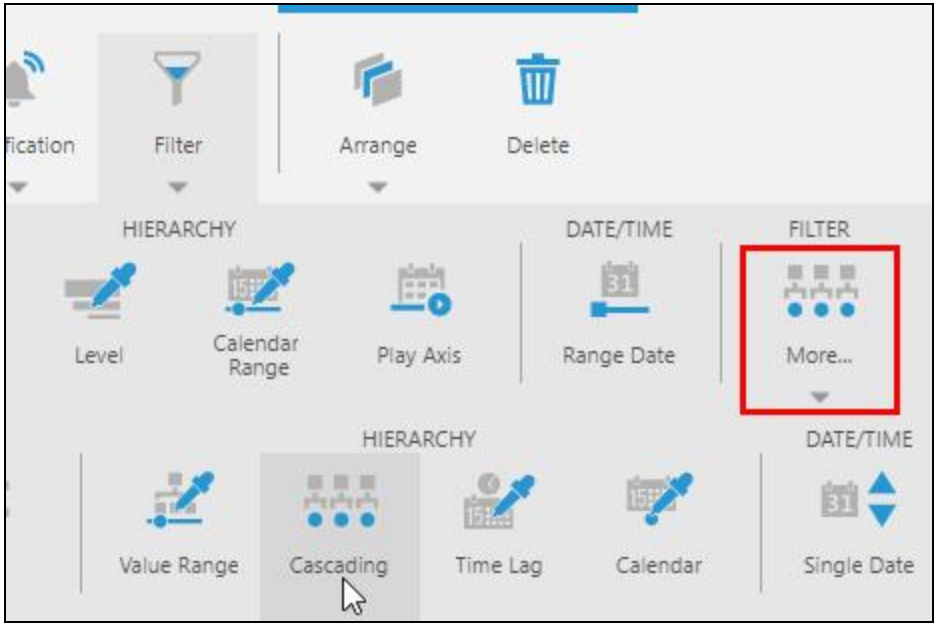
In order to take advantage of separate drop-downs for each hierarchy level, you need to be using a hierarchy with multiple levels. These can be created from the main menu to use if needed, or may have been added already to a data cube or OLAP cube.

Setting Up the Cascading Filter

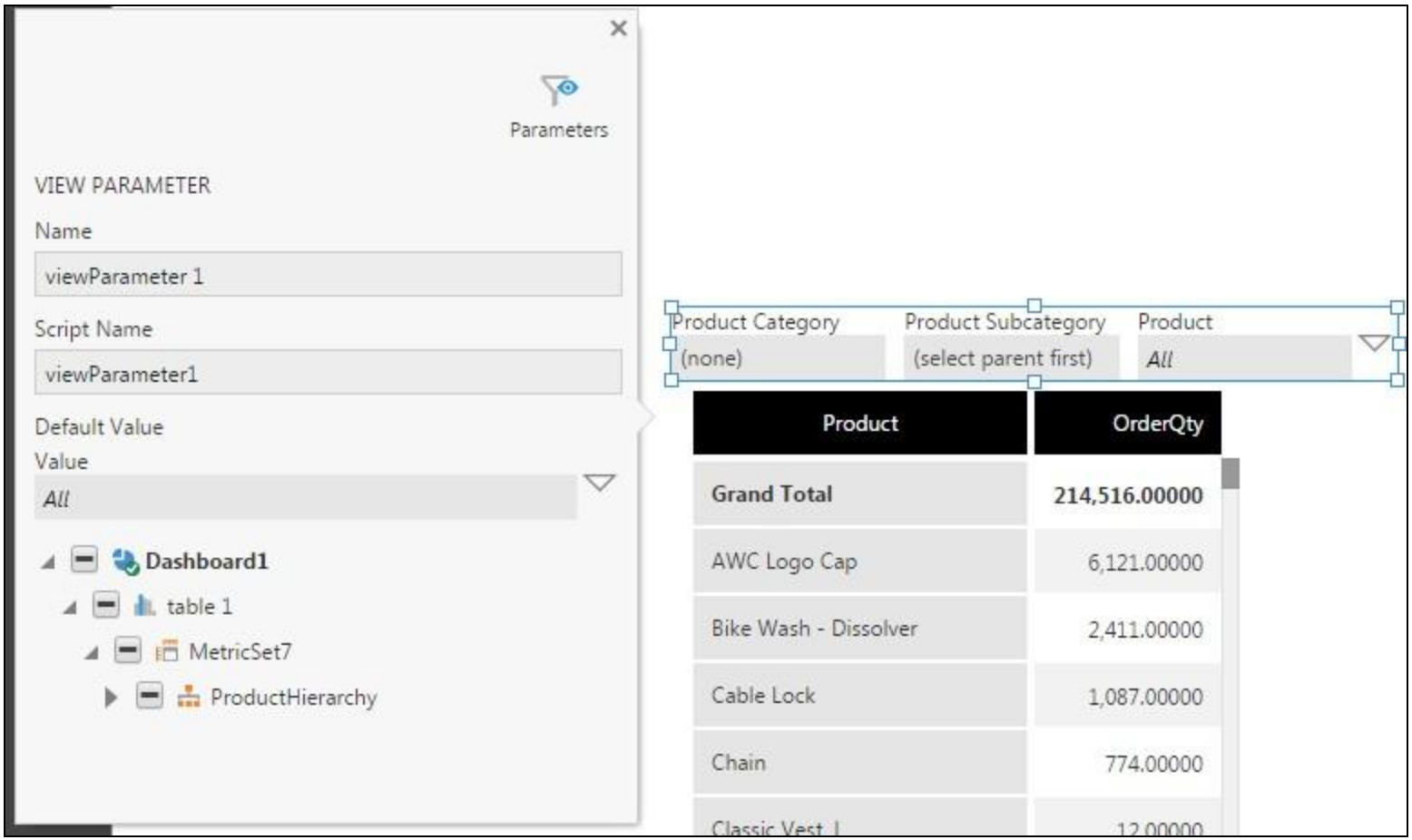
Add the multi-level hierarchy and other data you want to see in a metric set on a dashboard or other view.



In the toolbar, click **Filter**, click **More** to see all the options, and choose a **Cascading** filter.



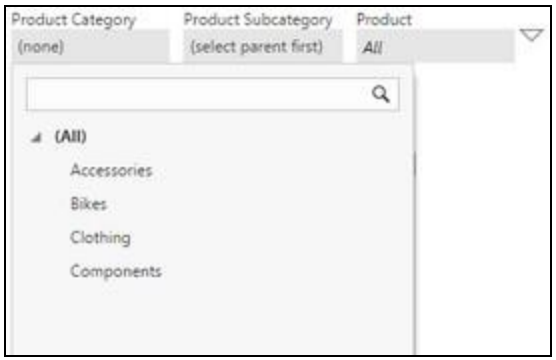
Connect the filter to your hierarchy.



Testing

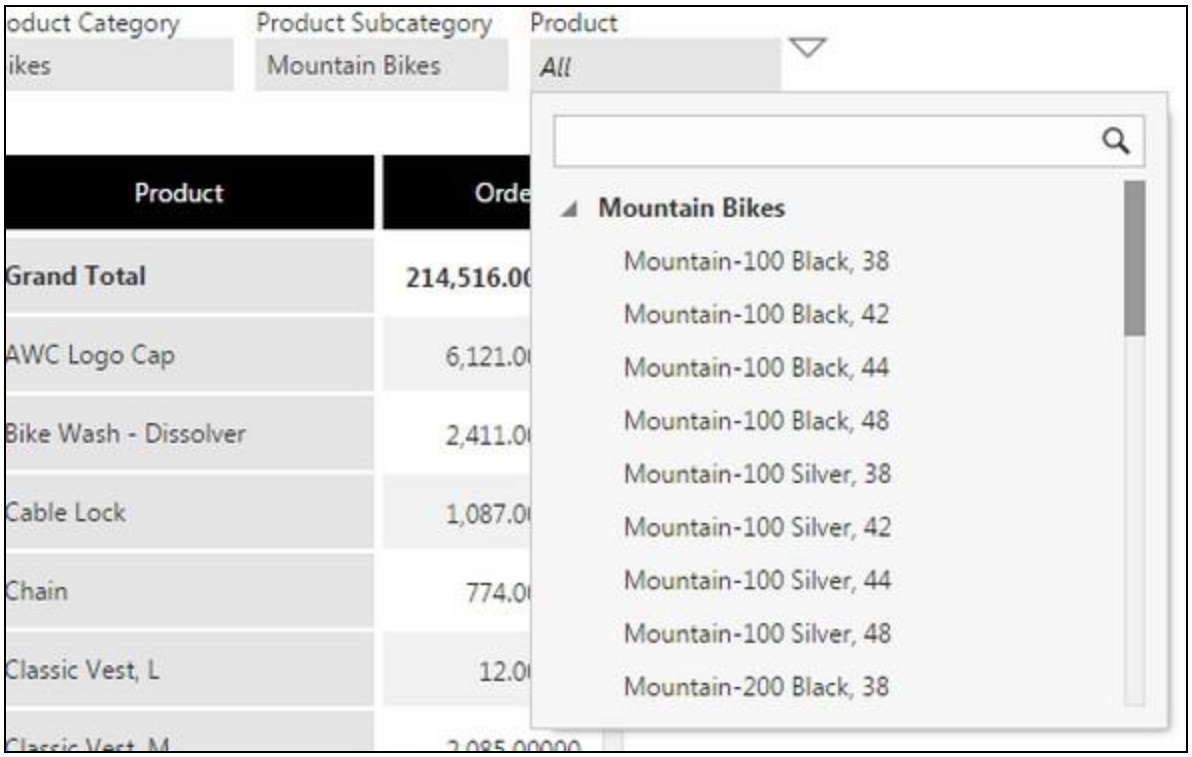
The filter in this example connects to a product hierarchy like the one created in Automatic Joins and Hierarchies.

Select **Bikes** on the first level drop-down (**Product Category level**).

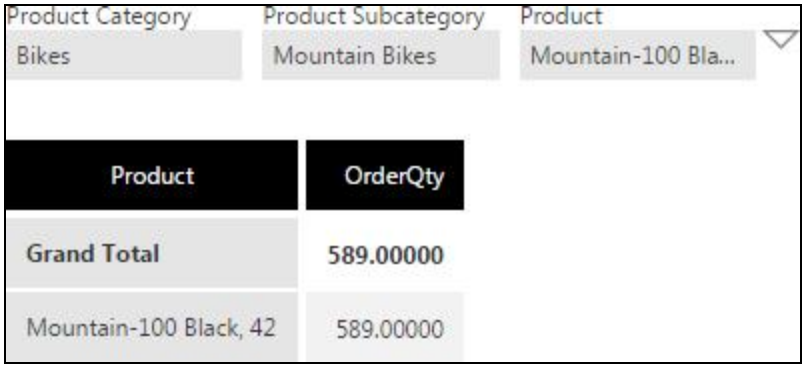


Click on the second level drop-down. The values listed should belong to the value that was previously selected (**Bikes**). Select **Mountain Bikes**.

Click on the third level drop-down. The values listed should again belong to previous value (**Mountain Bikes**). Select **Mountain-100 Black, 42**.


oduct Category
ikes
Product Subcategory
Mountain Bikes
Product
All
Product
Orde
Grand Total
214,516.0
AWC Logo Cap
6,121.0
Bike Wash - Dissolver
2,411.0
Cable Lock
1,087.0
Chain
774.0
Classic Vest, L
12.0
Mountain Bikes
Mountain-100 Black, 38
Mountain-100 Black, 42
Mountain-100 Black, 44
Mountain-100 Black, 48
Mountain-100 Silver, 38
Mountain-100 Silver, 42
Mountain-100 Silver, 44
Mountain-100 Silver, 48
Mountain-200 Black, 38

Result


Product Category
Bikes
Product Subcategory
Mountain Bikes
Product
Mountain-100 Bla...
Product
OrderQty
Grand Total
589.00000
Mountain-100 Black, 42
589.00000

Properties

You can customize properties such as the following in the Properties window:

- **Hide Token Menu:** If this option is selected, the triangular token menu button will not be displayed.
- **Manual Items:** If enabled, the item(s) are manually controlled instead of auto-populated from your data. This is most useful when using this control with script instead of with a parameter, and does not apply when connected to a view parameter and hierarchy.
- **Show Checkboxes:** Enable the user to select multiple items. This only applies to the last hierarchy picker.
- **Set Value On Any Filter Change:** By default, any visualizations affected by this filter will only update when a value is chosen in the last drop-down. Enabling this option will cause the visualizations to update whenever any of the drop-downs change.
- **Show Search:** Show the search bar to enable users to search for specific hierarchy members. This applies to all drop-downs.

For more information, see:

- Video: Filters
- Data Cubes
- Automatic Joins and Hierarchies
- Design Overview

Using a Play Axis Filter

This applies to: *Managed Dashboards, Managed Reports*

This article shows you how to use the Play Axis and Hierarchy Slider filters to select a value or range along a slider, and progress automatically when you press 'play'.

These filters can be used with any data added to Rows, Columns, or Slicers in your metric set, but normally make sense with a time dimension hierarchy, or a column of dates or date & time values. They can make connected visualizations change and animate through time as the play axis progresses, or allow you to select a range of dates to filter by.

Example Preparation

The example shown here uses the **Product Sales** data cube from the Getting Started Tutorial sample project.

We've created a new dashboard and dragged the following data to the canvas and the data analysis panel that appears:

- **Measures:** Sales Amount, Order Quantity, Gross Profit
- **Rows:** Country, Product Category
- **Slicers:** Date

The data should automatically re-visualize into a bubble chart, or you can choose one from **Re-Visualize** in the toolbar.

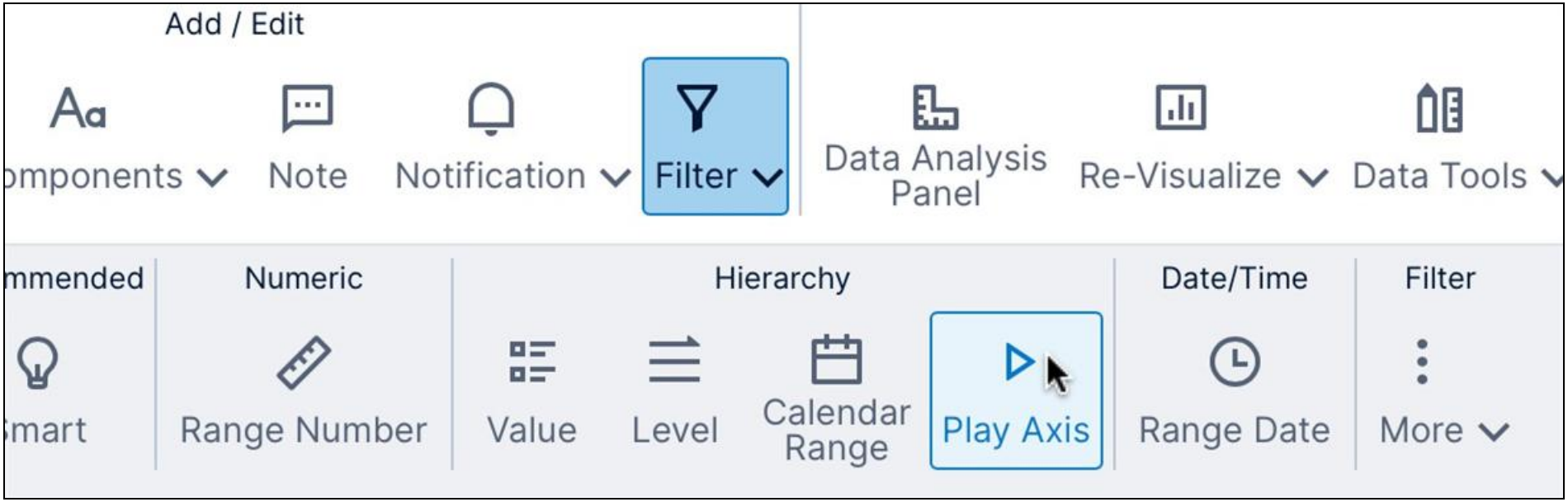
For the effect of data changing over time, you will usually want to place date/time data under **Slicers** and connect it to the play axis.


Data Analysis Panel: Metric Set
Edit Full Screen
Visualization
Metric Set 1
Measures
Sales Amount
SUM
Order Quantity
SUM
Gross Profit
SUM
click to add, or drop a measure
Rows (Data Points)
Country
Product
click to add, or drop a hierarchy
Slicers
Date
click to add, or drop a hierarchy
Columns (Series)
click to add, or drop a hierarchy
Sales Amount
70M
60M
50M
40M
30M
20M
10M
0M
-10M
-500K
0K
500K
1,000K
1,500K
2,000K
2,500K
3,000K
3,500K
4,000K
Gross Profit

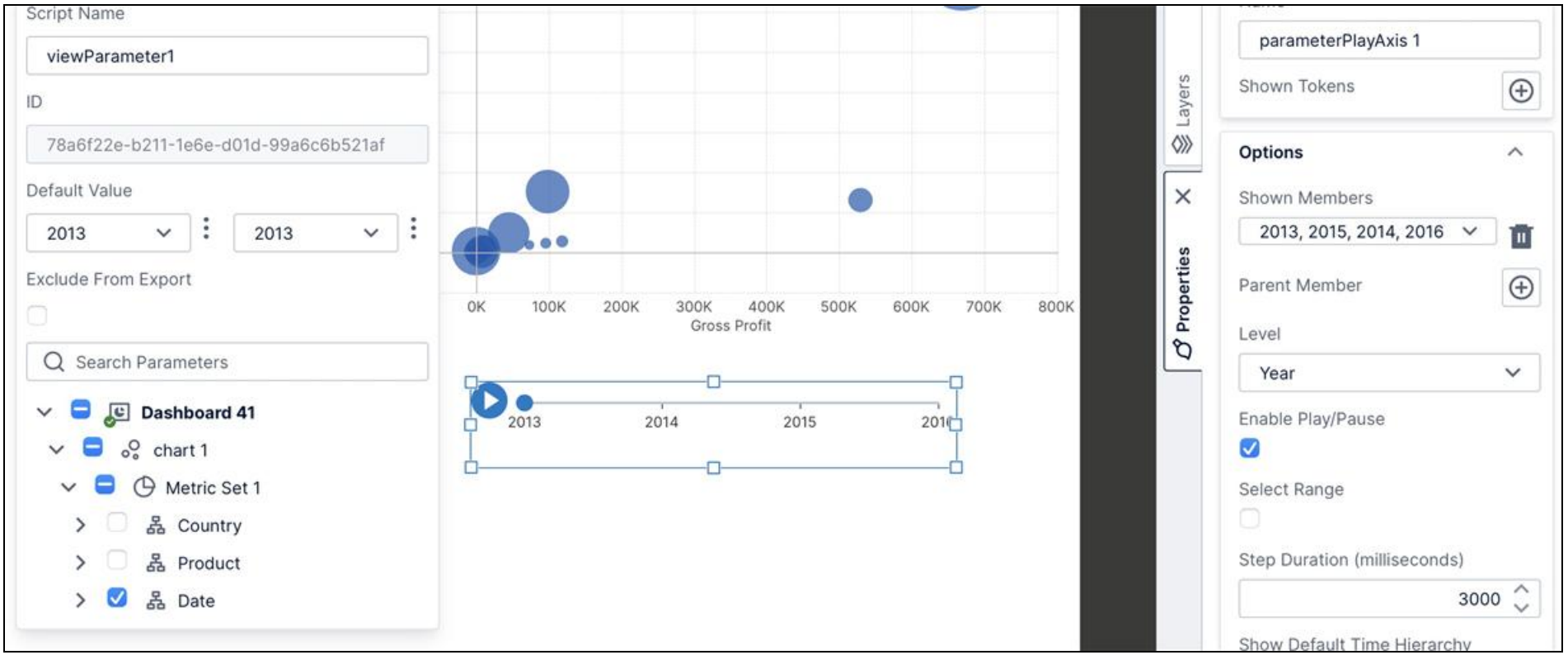
Note: The Play Axis filter can also be used with any other type of visualization on your dashboard, and in other types of views such as reports, just like other filters.

Creating a Play Axis

From the toolbar, click **Filter** and select **Play Axis**.



In the filter connections panel that appears, some data is normally selected automatically (**Date** in this example). You can change this if you want to filter something else.



When connected to date/time data, the play axis automatically displays a selection of values relative to the current date. In our particular example shown here, our data set is older so that the default values don't have any corresponding data.

To adjust what values are included along the play axis, open the Properties window for the play axis and change the values selected for **Shown Members**.

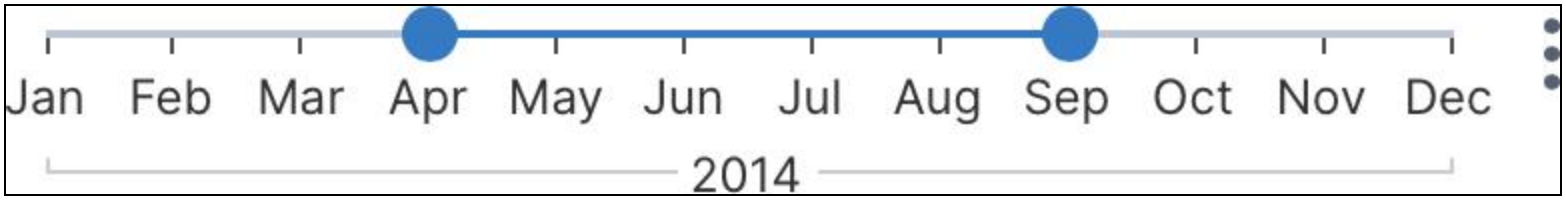
Creating a Hierarchy Slider

When you click **Filter** in the toolbar and then **More**, you can find a new **Slider** option under **Hierarchy**.

This is actually the same as a play axis filter but with different default settings selected in the Properties window. You can also choose these yourself after choosing a Play Axis from the toolbar, or customize them after choosing a Hierarchy Slider:

- The play/pause button is disabled (**Enable Play/Pause** is de-selected).
- Range selection is enabled (**Select Range** is selected).

- The token menu is enabled (**Hide Token Menu** is de-selected).



Viewing

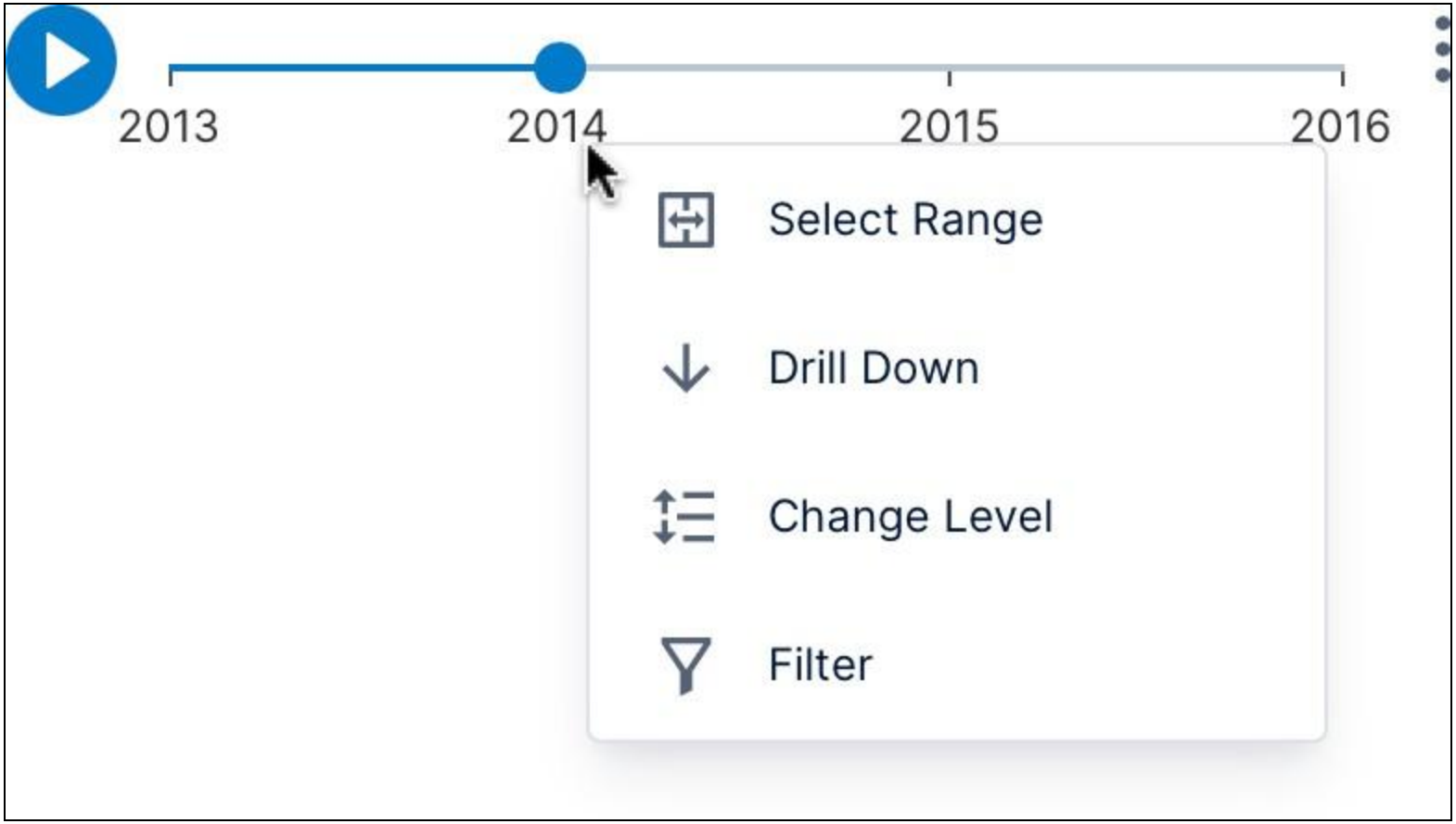
Switch to **View** mode in the toolbar. Press the play/pause button if enabled, or click on values along the slider. When range selection is enabled, click and drag a 'handle' (e.g., blue circle) along the slider to change the start or end of the selected range, or click along the slider to move the nearest handle to that location.

You and other viewers of this dashboard (or other view) can right-click or long-tap:

- A specific value to drill down, filter out, or filter to that value along the slider.
- Anywhere on the filter to clear filtering, change the level or drill up.
- Anywhere on the filter to access range selection options if the connected view parameter value supports a range.

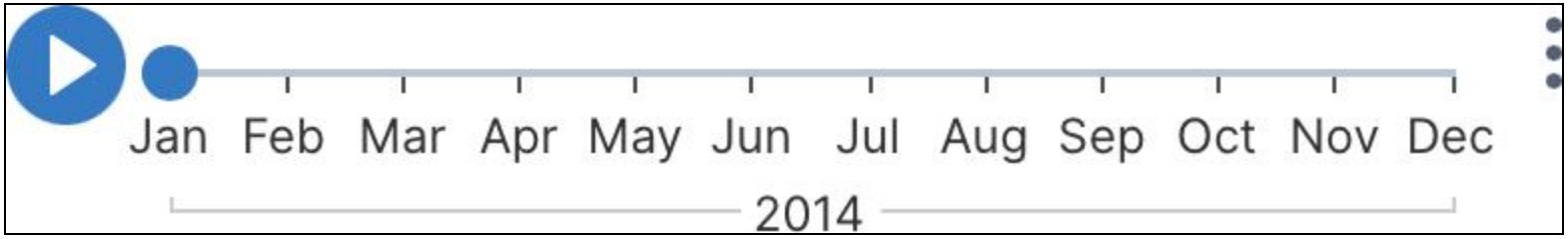
  Note: Range selection is only available if the connected view parameter value supports a range of values. For example, if you first created a Calendar filter (which supports choosing only a single value), then clicked the Parameters button in the filter connections panel for a Play Axis filter, you could connect the filter to the same view parameter as the Calendar filter and it would not support range selection.

- When range selection and the play/pause button are enabled, you can choose the **Pin Range Start** option if you want pressing 'play' to automatically progress the end value of the range while keeping the same start value.



Note: The Pin Range Start option can have the effect of displaying cumulative or running totals over time if your measure's aggregator is **Sum**.

For example, right-clicking 2014 and choosing **Drill Down** will list along the slider the values for 2014 from one level below, such as months:



If you want to prevent viewers from accessing some or all context menu options, you can disable application privileges the same way as for visualizations. Privileges can be disabled in the Properties window for the play axis or for the dashboard (or other view), or for certain users.

Properties

When editing, you can optionally change settings in the **Properties** window for this filter that will take effect when the dashboard or other view is initially opened for viewing. For example:

- **Shown Members** – Select certain values if only those values should be initially displayed along the axis, or leave un-set to display them all. This is equivalent to the Filter context menu option in view mode.
- **Parent Member** – Set this property to display all the values from the level below that make up the selected value (this will override the Shown Members property). This is equivalent to the Drill Down context menu option for that value in view mode.
- **Level** – Select the current level of the hierarchy to display along the play axis, if it offers multiple levels. Viewers can also change the level from the context menu. (The displayed members will be filtered by the Shown Members, even if set to a higher level, e.g., setting Shown Members to 2017, 2018 and Level to Month will display only the months between 2017-2018).
- **Enable Play/Pause** – Shows the play/pause button for viewers to click to automatically progress the slider and stop the progression. If this option is de-selected, the filter will function mostly as a slider that works for time dimensions and other hierarchies.
- **Select Range** – Allows setting both a start value and end value along the slider to select the range of values between them, available only if the connected view parameter value supports a range. Viewers can also access this option from the context menu.
- **Pin Range Start** – When the play/pause button and range selection are both enabled, pressing 'play' automatically progresses only the selected range end value while keeping the same start value. Viewers can also access this option from the context menu.
- **Step Duration** – determines how fast the play axis progresses when playing. (You may also need to adjust the animation properties for some visualizations to match as described below).
- **Hide Token Menu** – this option in the Look tab can be unchecked to allow viewers to choose tokens such as relative time options Today or Previous month. The corresponding values will be indicated automatically along the slider if available and enabled in the connected time dimension hierarchy.
- **Show Value Tooltips** – this option in the Text tab is enabled by default to automatically pop up a tooltip with the current value when it changes. You can disable it, or use the property Value Tooltip Duration to indicate how long this tooltip will be visible.

Visualization Properties

Connected visualizations, and charts in particular, have their own animation settings that you may want to adjust when using a play axis filter:

- In the Properties window for a chart, find animation settings in the Look tab: consider de-selecting the **Animate In Stages** setting, and setting **Data Change Staggering** to **0%**.
- Be sure the visualization's **Transition Duration** or other animation duration property settings are far enough below the set Step Duration for the play axis for your particular data source, or consider improving its performance if needed.

> **Note:** Range selection is only available if the connected view parameter value supports a range of values. For example, if you first created a Calendar filter (which supports choosing only a single value), then clicked the Parameters button in the filter connections panel for a Play Axis filter, you could connect the filter to the same view parameter as the Calendar filter and it would not support range selection.

For more information, see:

- Slicers Versus Columns and Rows
- Adding Filters
- Using a Slider Filter
- Disable View Functionality
- Application Privileges
- Set or Reset the Default Time Dimension

Using a Slider Filter

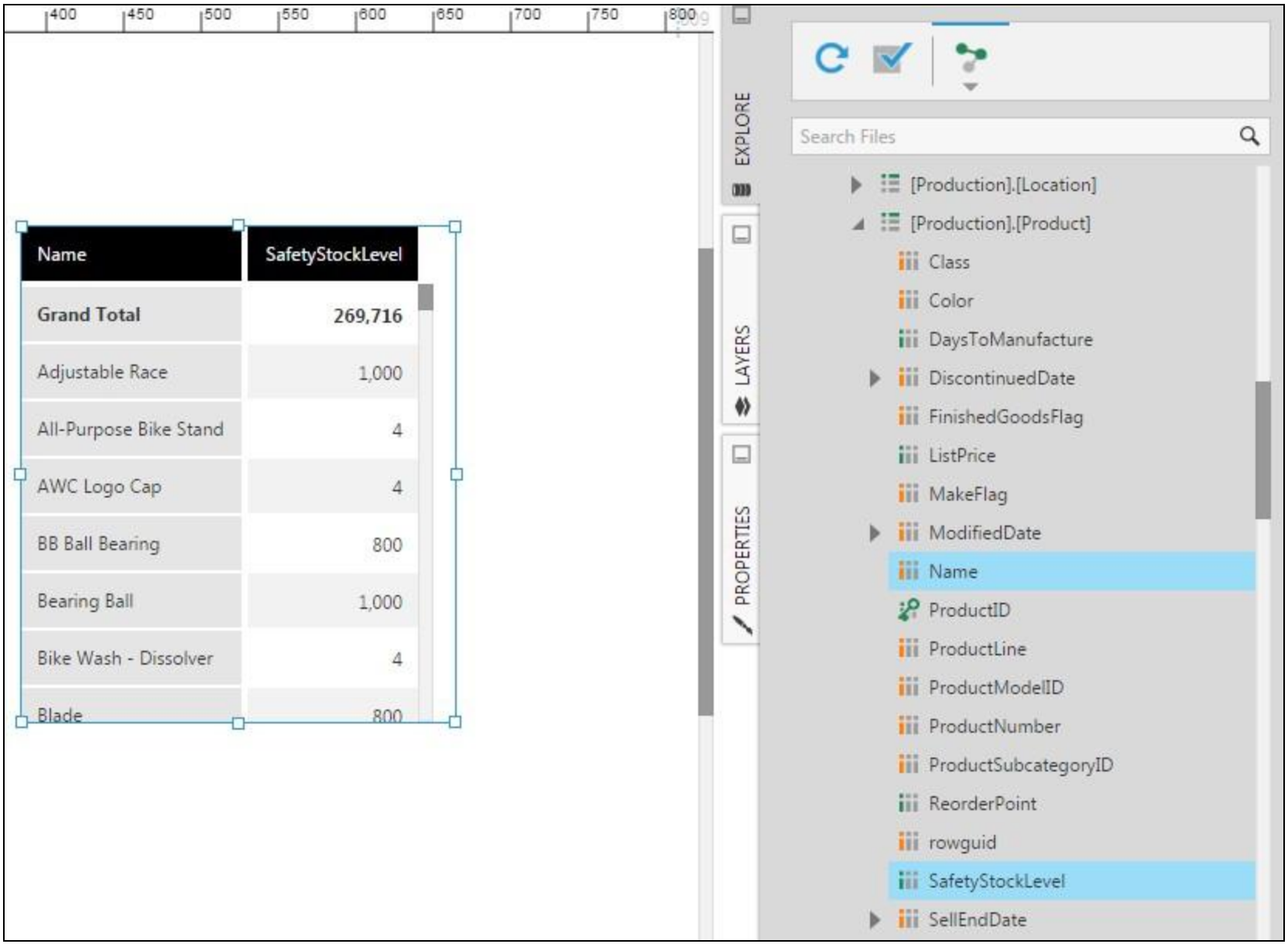
This applies to: *Managed Dashboards, Managed Reports*

This article shows you how to use a slider filter to select a single or a range of numeric values.

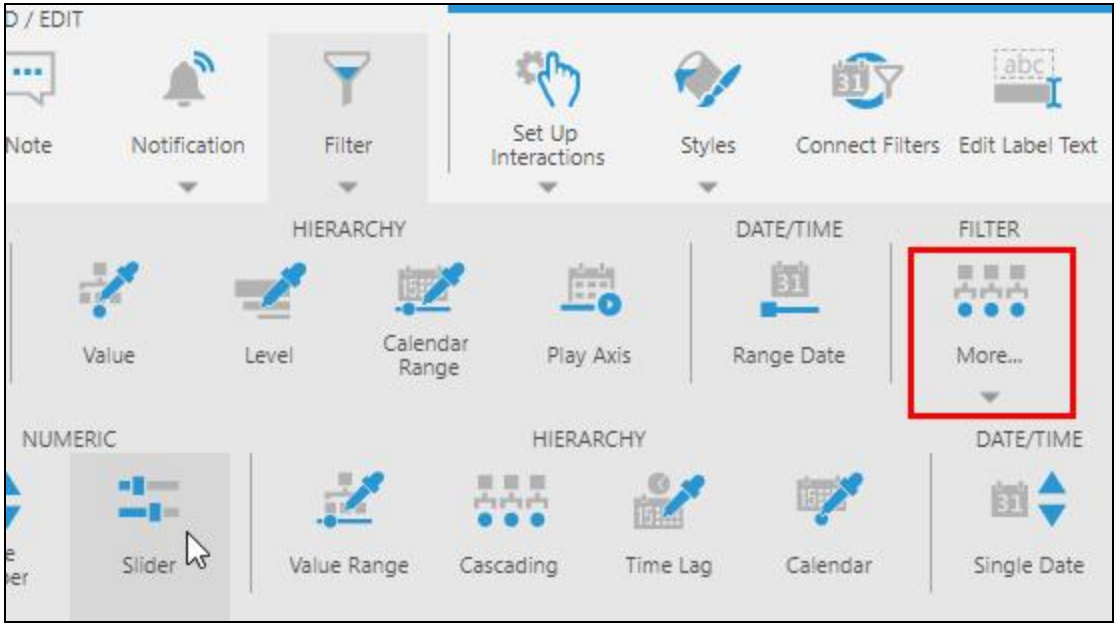
Filtering a Numeric Range

For this example, create a new dashboard using the **Blank** template.

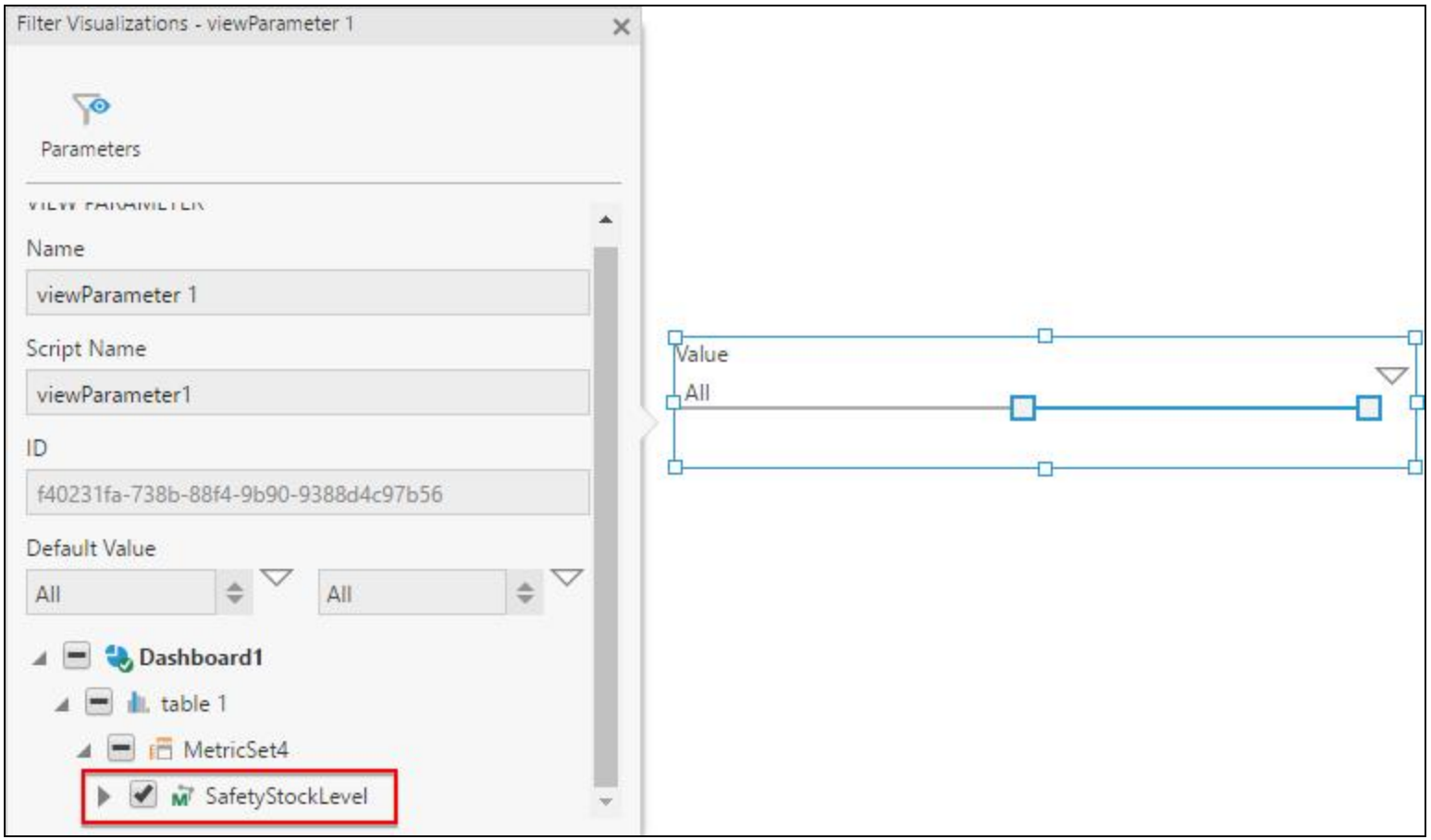
Drag a measure (**SafetyStockLevel**) and a row hierarchy (**Name**) from the **[Production].[Product]** table in **Explore**.



Click **Filter** in the toolbar, and then select **Slider** (found under **More**).



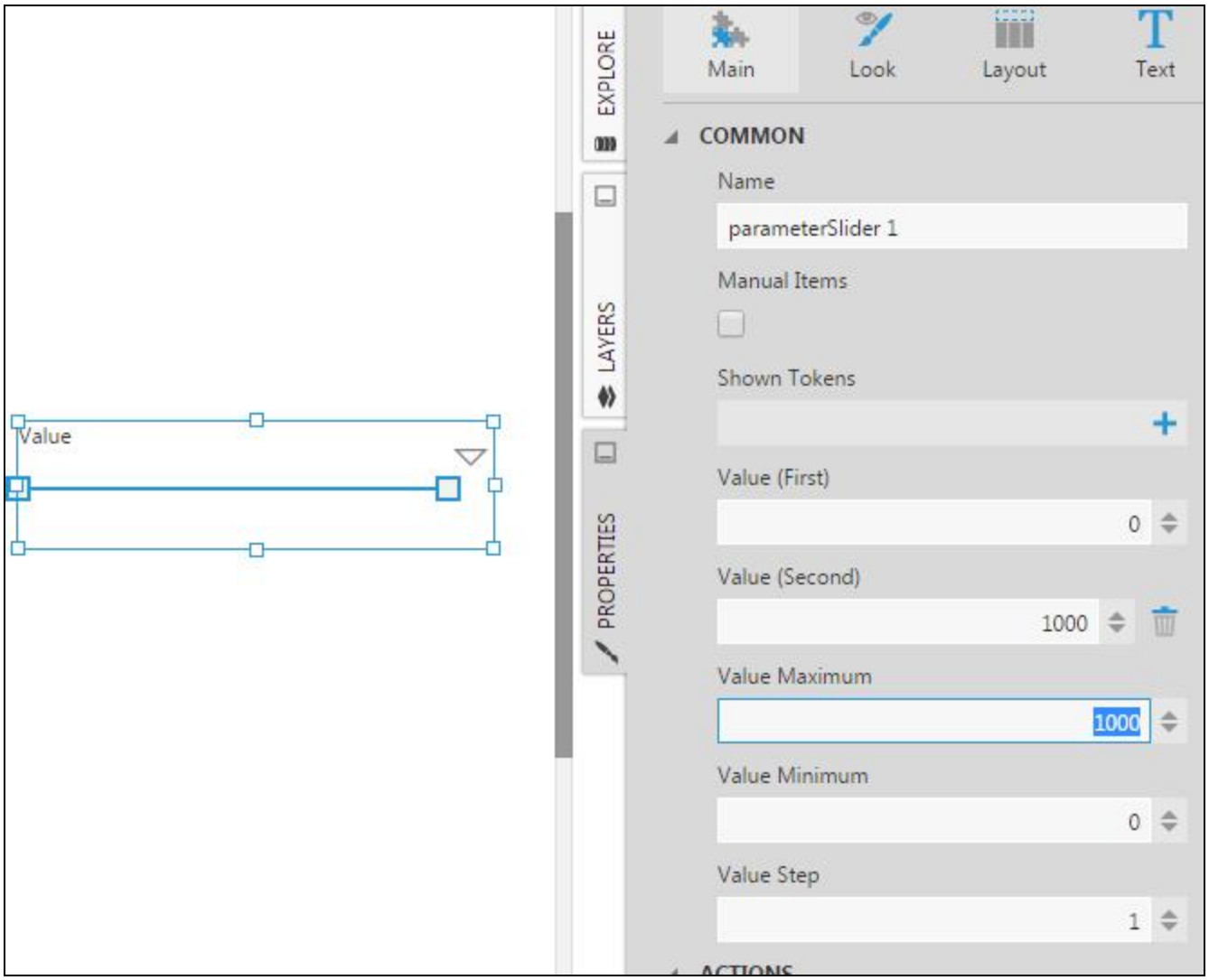
The **Filter Visualizations** panel for the slider filter shows that it is already connected to the **SafetyStockLevel** measure.



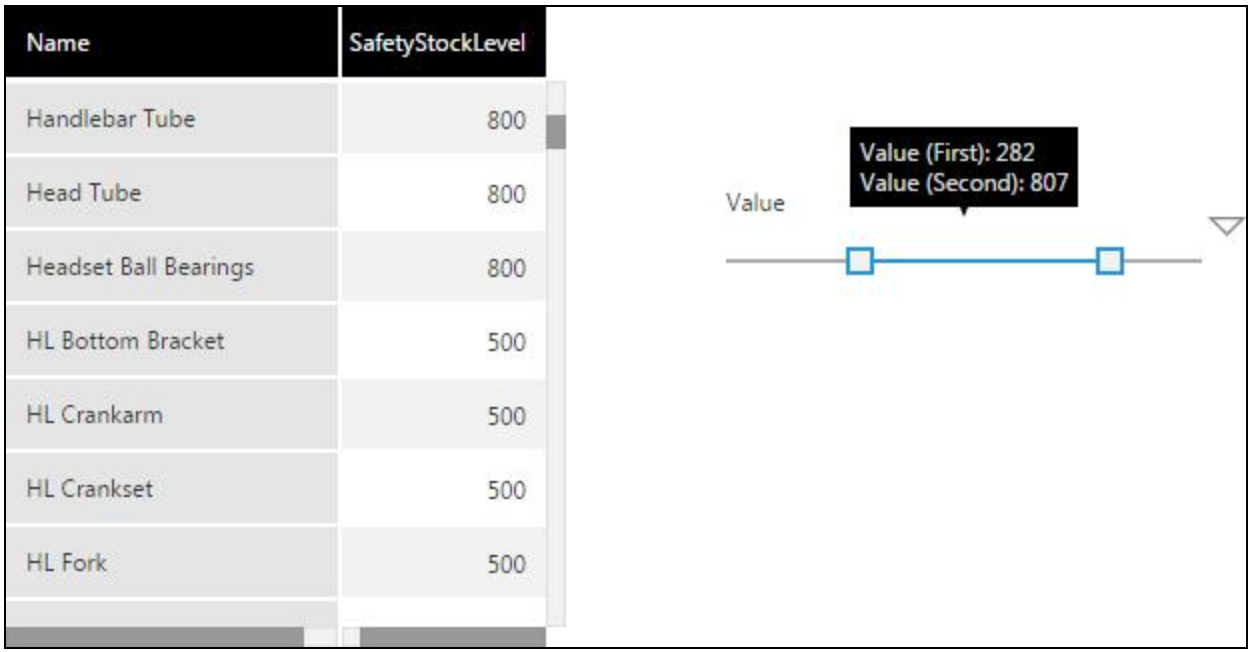
The Slider filter supports selecting a range of numeric values, which is why it shows two slider handles in this scenario. The minimum and maximum values of this numeric range need to be set next (unless the defaults of 0 and 100 are appropriate for your data).

Go to **Properties** for the slider filter, and for example:

- Set **Value (First)** to **0** - This is the first slider (handle) value.
- Set **Value (Second)** to **1000** - This is the second slider value (if slider is in range mode).
- Set **Value Maximum** to **1000** - This is the maximum value of the slider.

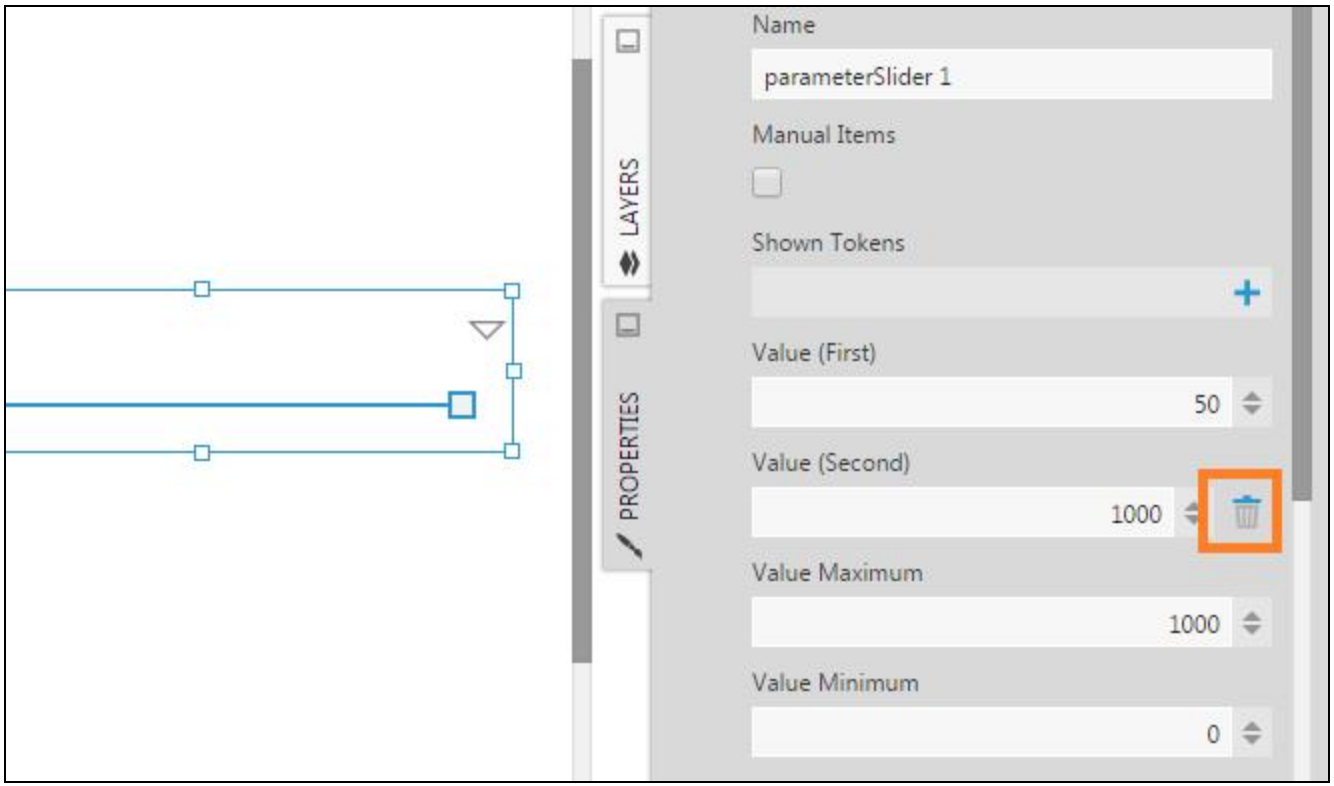


Switch to **View** mode from the toolbar and start using the slider. Hover over the range portion of the slider to see the current first and second slider values displayed in a tooltip.



Filtering a Single Numeric Value

To change the previous example to a single-valued slider, the easiest way is to go to the **Properties** window for the slider and delete the **Value (Second)** item.



Another approach is to start from the beginning and add a Single Number filter first. This will automatically give you an associated view parameter (**viewParameter 1**) that supports a single value.

Next, delete the **Single Number** filter but say no when it asks if you want to delete associated view parameters.

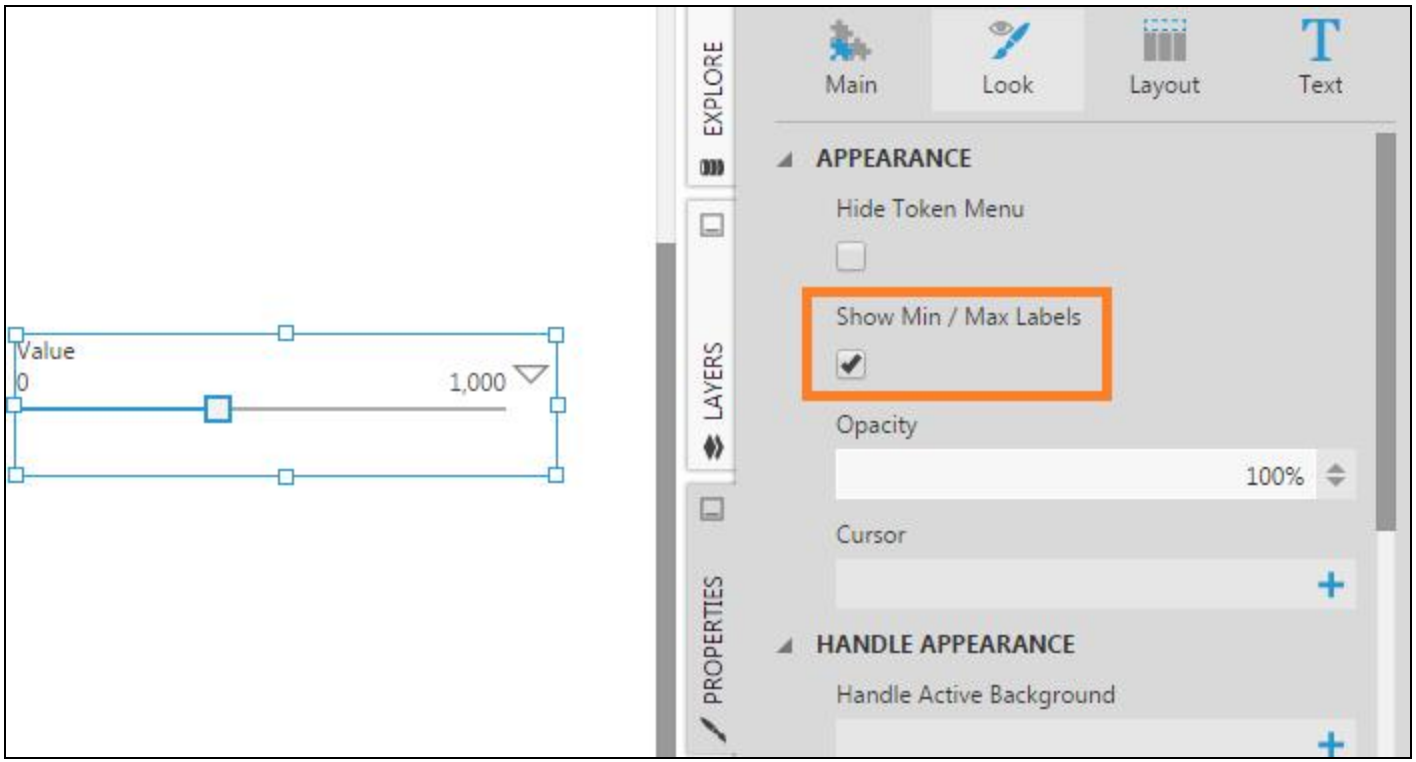
Add a **Slider** filter. In the **Filter Visualizations** panel, **click** Parameters in the top-right and connect to **viewParameter 1** instead.

Properties

The following are some additional properties available to customize in the Properties window:

Show Min / Max Labels

Under Look Properties, use the **Show Min / Max Labels** property to display the minimum and maximum slider values as labels.



Vertical Layout

Under Layout Properties, use the **Vertical Layout** property to orient the slider filter vertically instead of horizontally.

For more information, see:

- Adding Filters
- Create a Metric Set and Add a Filter

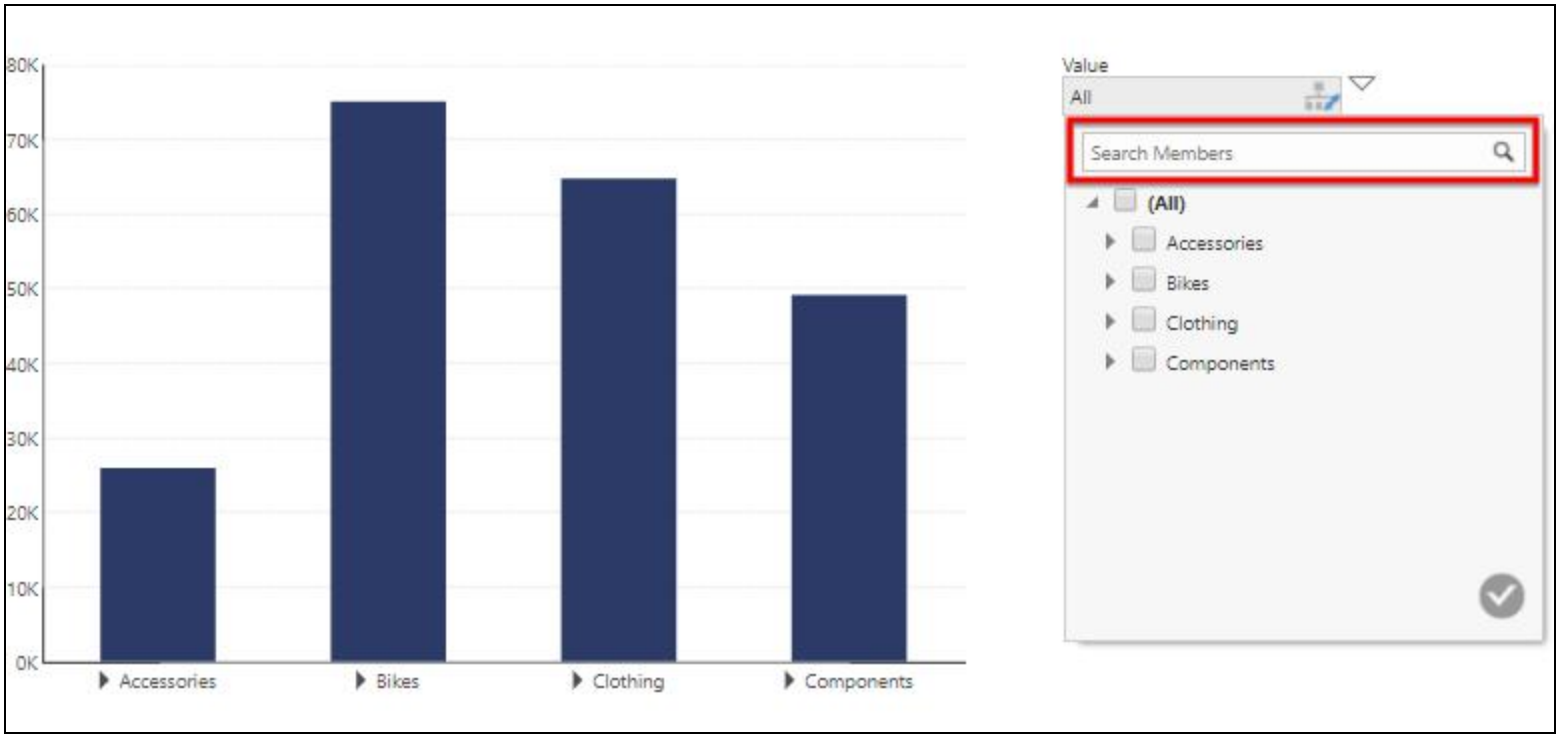
Using Search in Filter Controls

This applies to: *Managed Dashboards, Managed Reports*

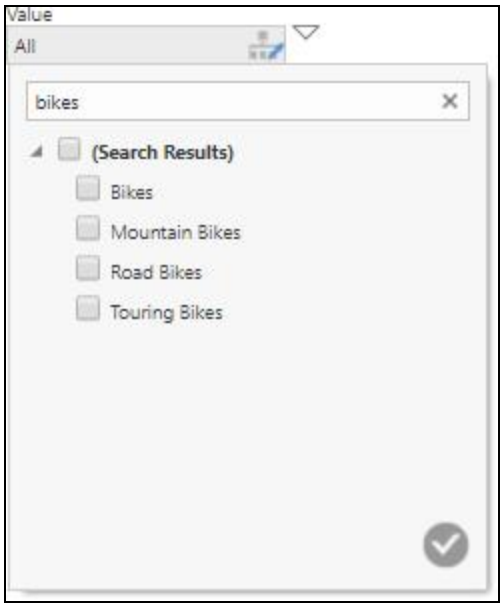
This article shows you how to use the search functionality that is built into various filter controls.

Hierarchy Filters

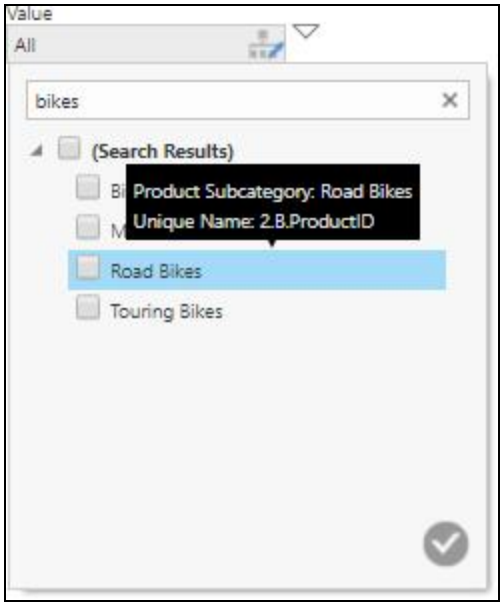
Hierarchy filters such as hierarchy Value, Value Range, and Cascading filters have a search box accessible when you use them while viewing a dashboard or other view. This lets you search for specific hierarchy values.



As you type your search text in the box, a flat list of search results is displayed.

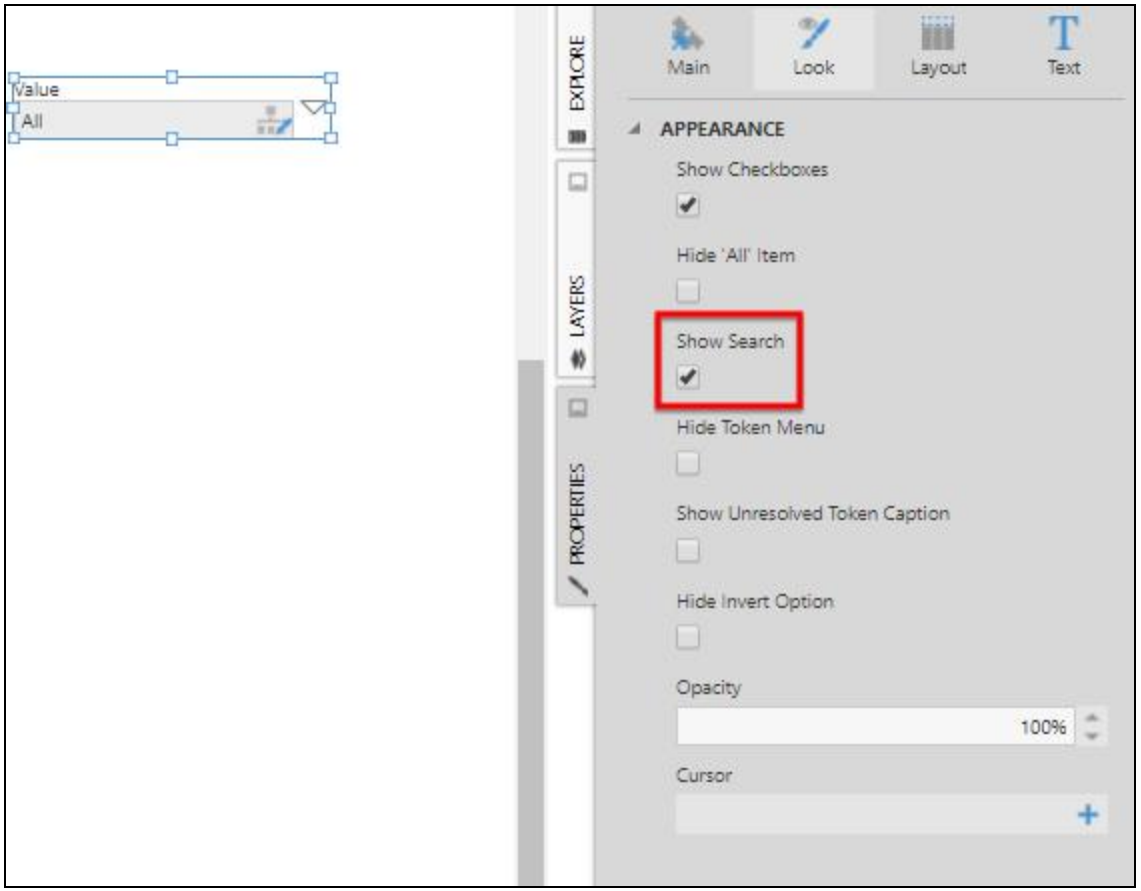


If you want to know what hierarchy level a search result item corresponds to, hover over the item to see its level and caption displayed in a tooltip.



After selecting an item, you can click **x** in the search field to clear it and then go back to the expandable tree of values (if applicable), where you can see where the selected items are located.

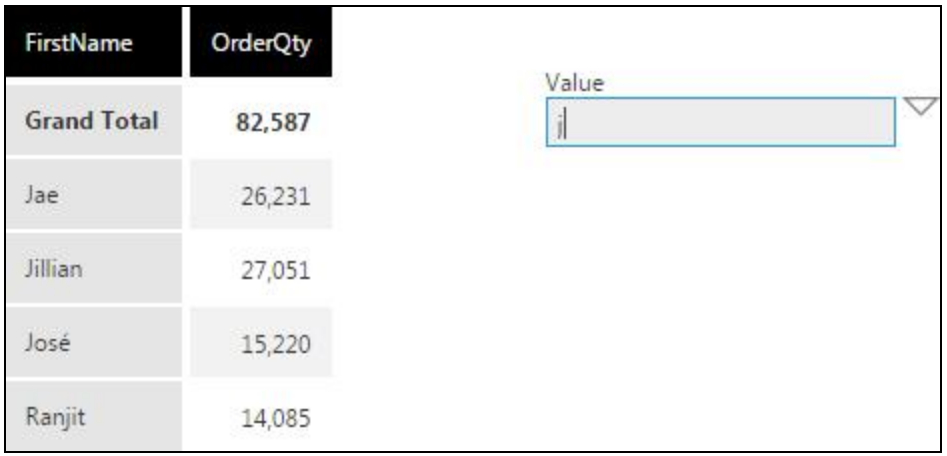
As an editor of a dashboard or other view, you can optionally hide the search box for a hierarchy filter. Go the **Properties** window for the filter when editing, and in the **Look** tab, uncheck the **Show Search** property.

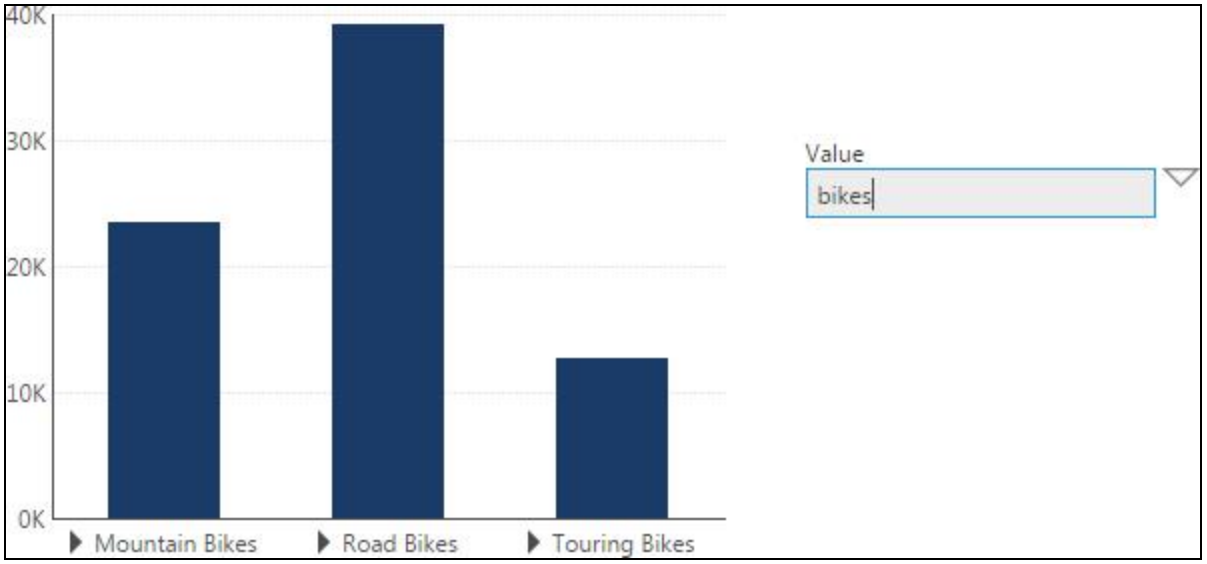


Textbox Filter

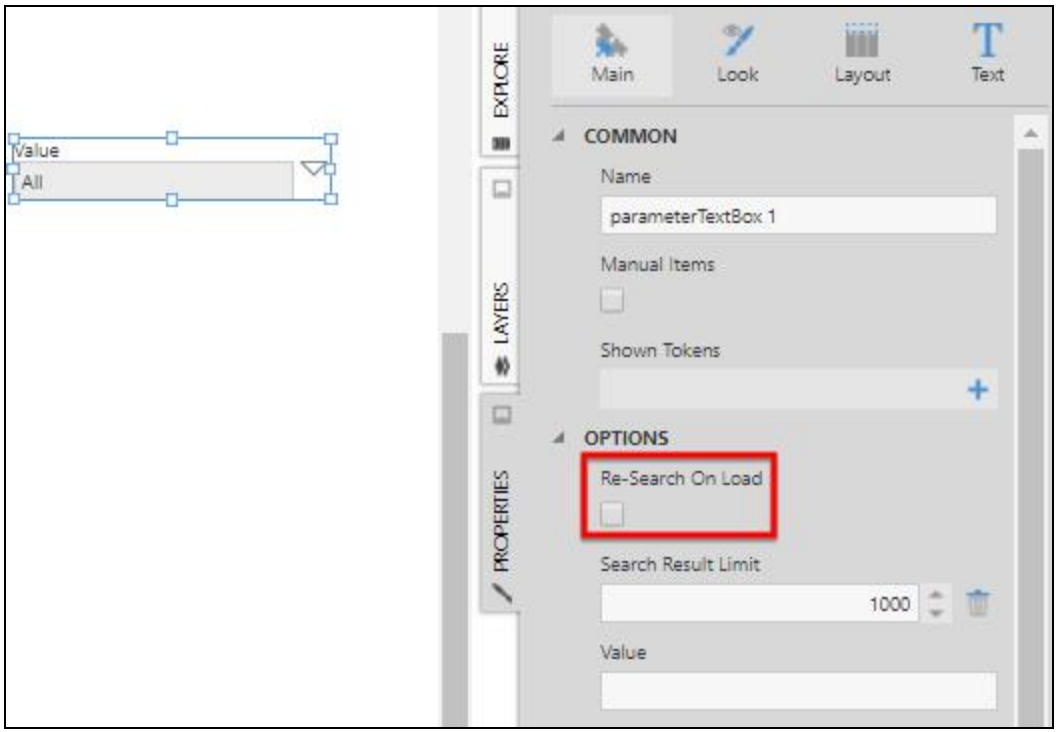
The textbox filter control can be connected to text-based columns (e.g., FirstName) as well as single/collection parameters. This means the textbox filter can be used to filter hierarchies as well.

To use the textbox filter, type your search text and then press **Enter**.





In the case of a hierarchy, you can enable the textbox filter's **Re-Search On Load** property to have it re-search the hierarchy whenever the dashboard is loaded.



For more information, see:

- Create a Metric Set and Add a Filter
- Adding Filters

Using the Checkbox List Filter

This applies to: *Managed Dashboards, Managed Reports*

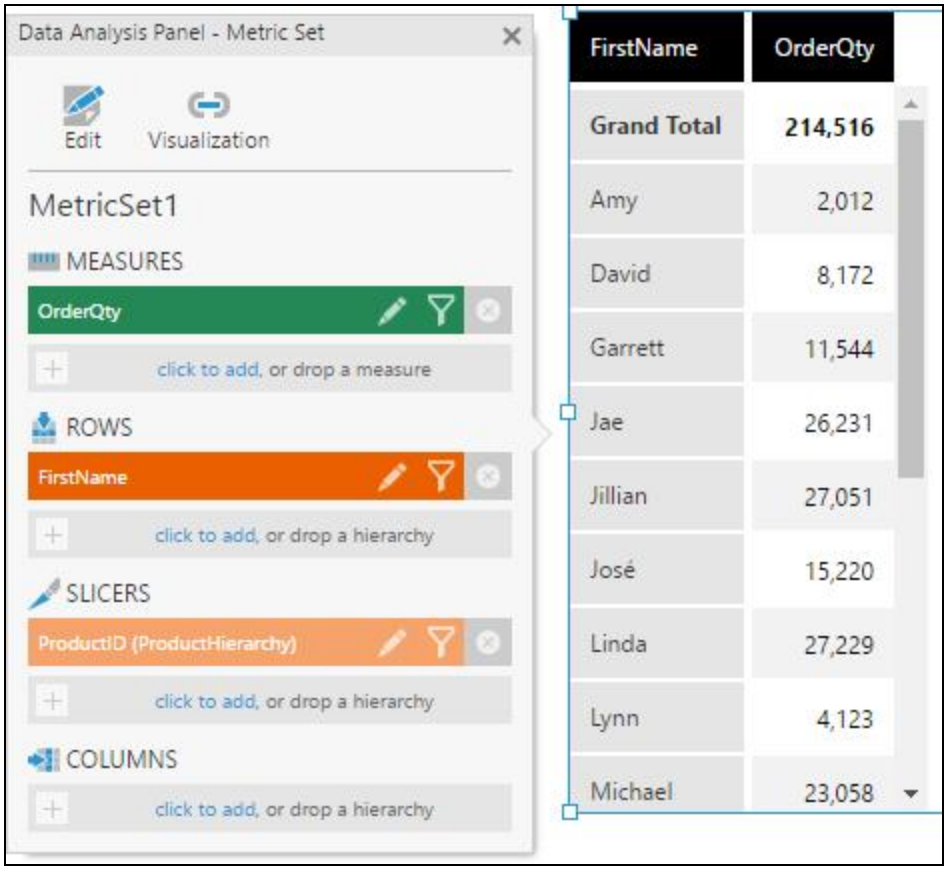
A checkbox list filter shows a list of values with checkboxes directly on your dashboard or filter bar, allowing you to select one or multiple. When set up for single value selection, the checkboxes appear as 'radio buttons'.

Only the first level of the hierarchy is visible in the checkbox list.

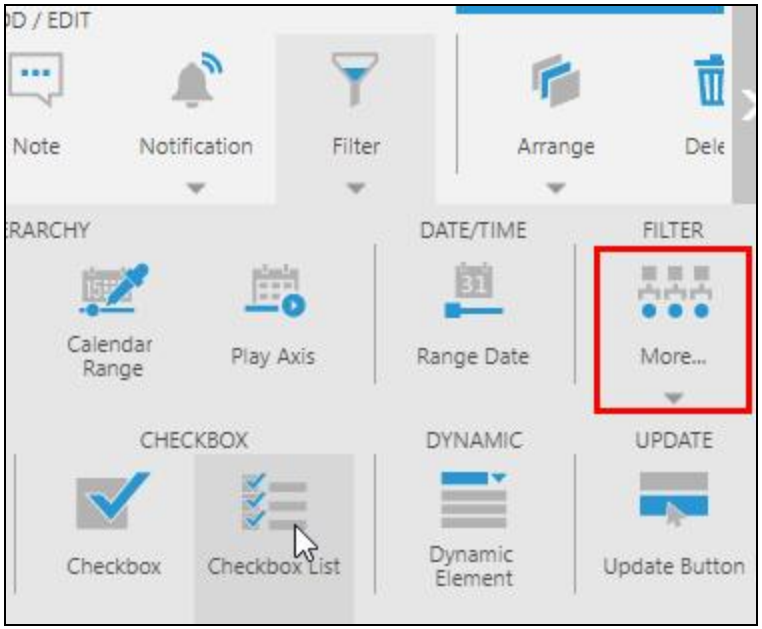
Adding the Checkbox List

The checkbox list filter can be connected to any hierarchy selected in your metric set.

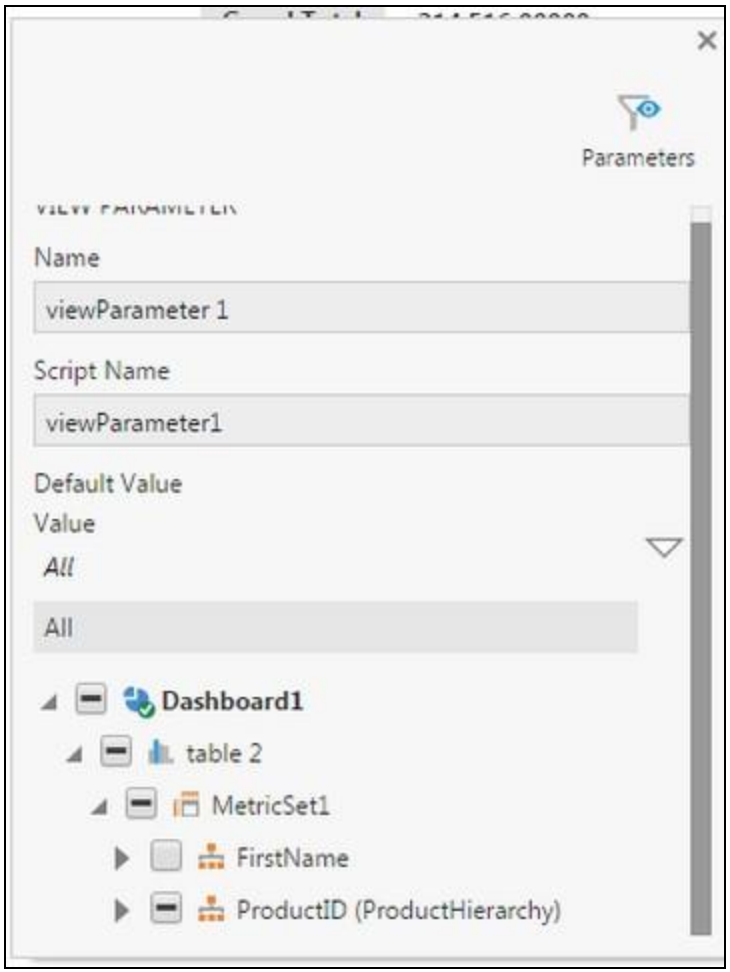
In the following example, the checkbox list could be connected to either FirstName or ProductID.



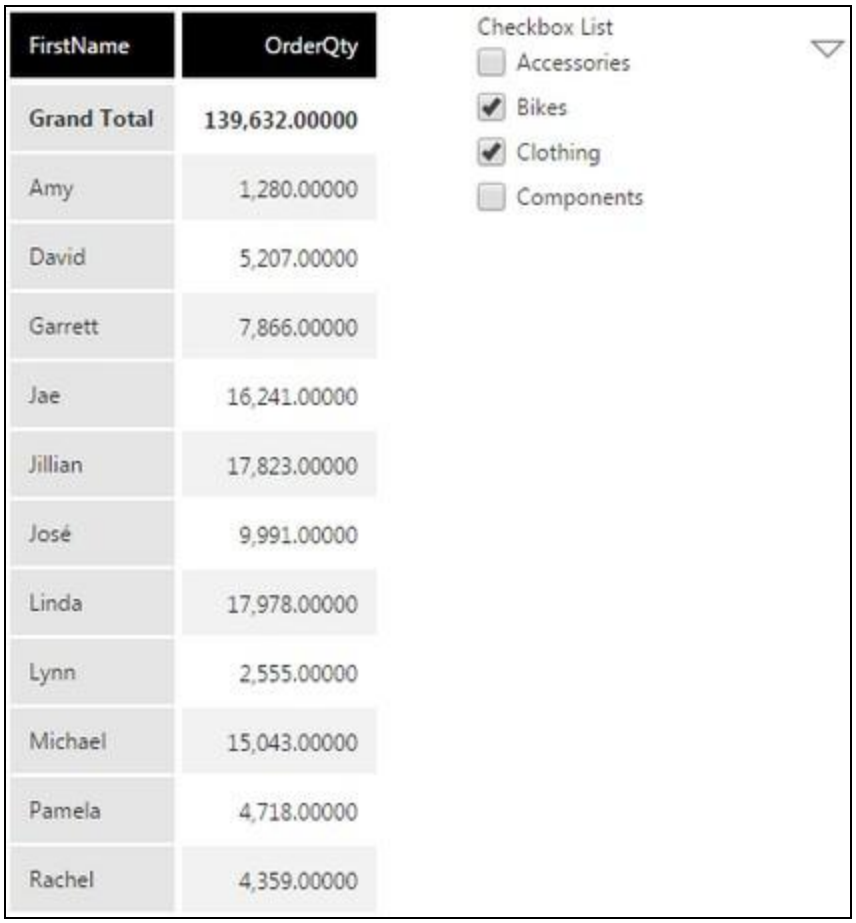
With the visualization selected that you want to filter, add a **Checkbox List** filter from the toolbar (found under **More**).



If not already connected, select your hierarchy to filter it in the popup or window that appears. You might need to uncheck a hierarchy before the other can be listed.



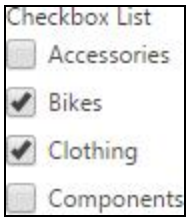
You can now switch to **View** mode in the toolbar and select one or more values from the hierarchy from the checkbox list displayed directly on the dashboard, or in the parameter bar for a report, scorecard, or small multiple.



Properties

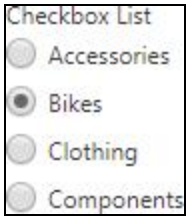
Properties including the following can be customized in the Properties window:

Hide Token Menu: If this is checked, the triangular token menu button will not be displayed.



Manual Items: If enabled, the item(s) displayed in the control are manually controlled instead of auto-populated from any associated view parameters. This can allow you to set up an interaction for each item. This is not applicable if the filter is connected to a hierarchy, like in the example above.

Single Selection Mode: If enabled, the checkbox list will allow only single selection and display as 'radio buttons' instead of checkboxes.



Vertical Layout: If this option in the Layout tab is un-checked, the checkbox list will be laid out horizontally instead of vertically.

For more information, see:

- Video: Filters
- Adding Filters
- Using Interactions
- Design Overview

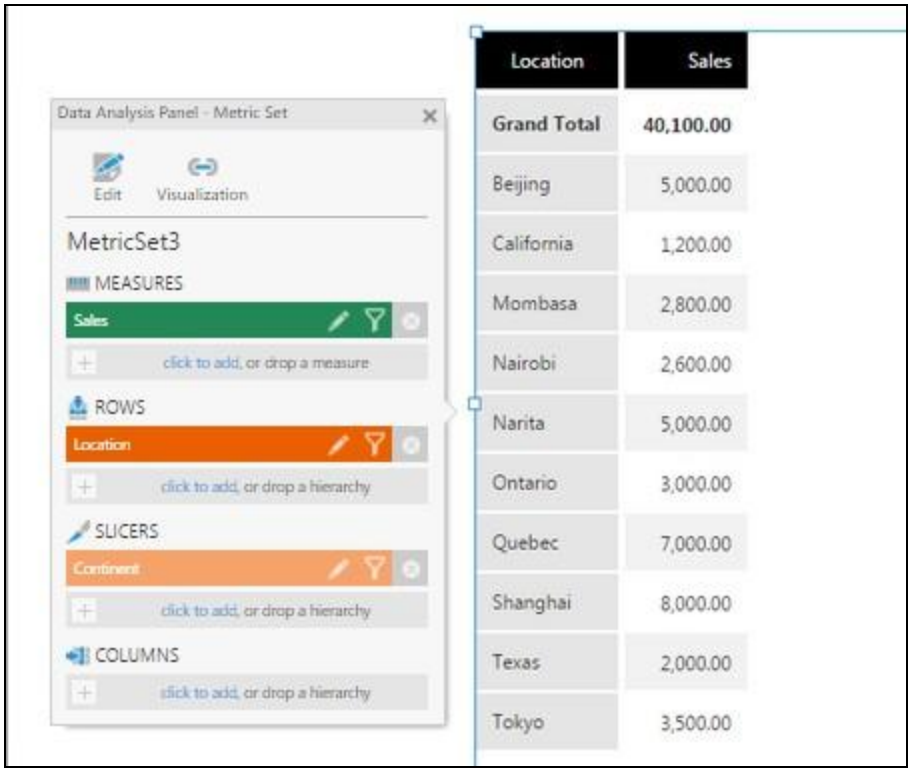
Using the Parameter Update Button

This applies to: *Managed Dashboards, Managed Reports*

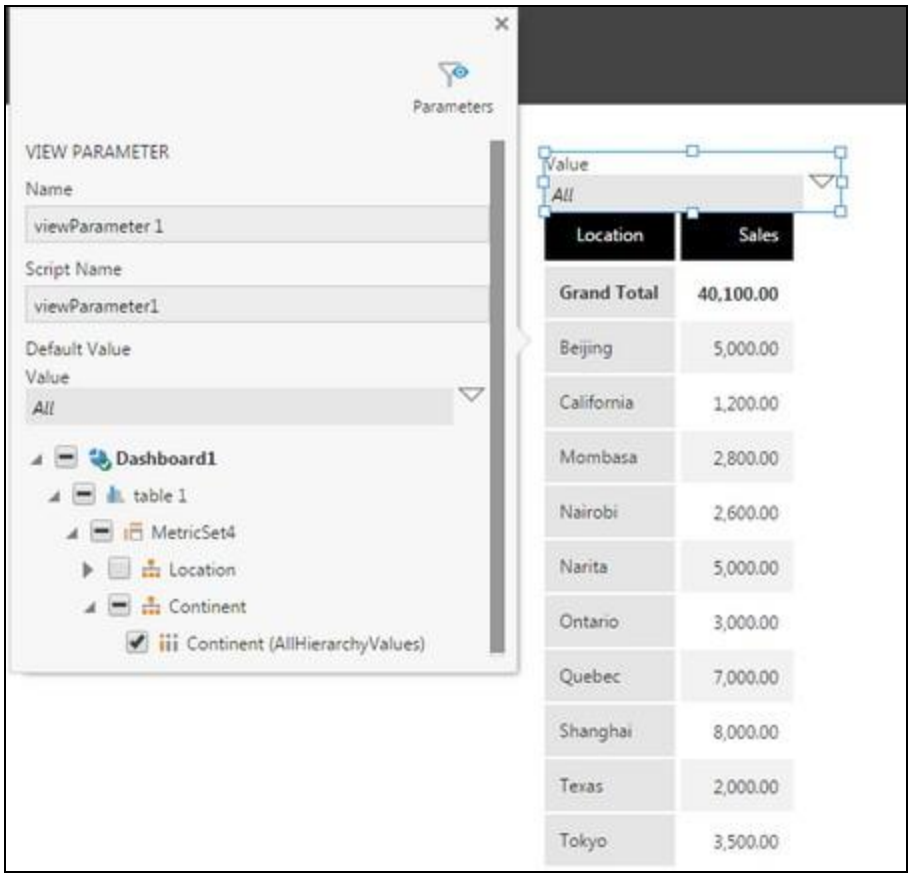
The parameter update button lets users decide when changes in filters should be applied to the connected visualizations. This is useful when you want to first set up multiple filters all at once before the corresponding data is loaded.

Set Up the View

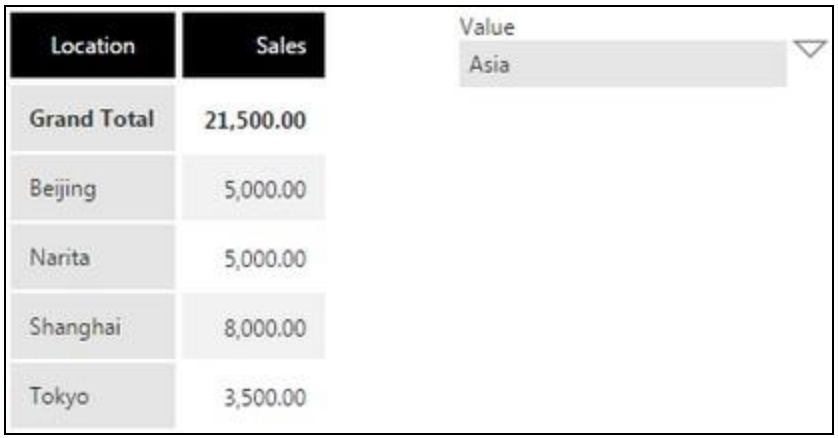
For this example, consider the following visualization placed on a dashboard:



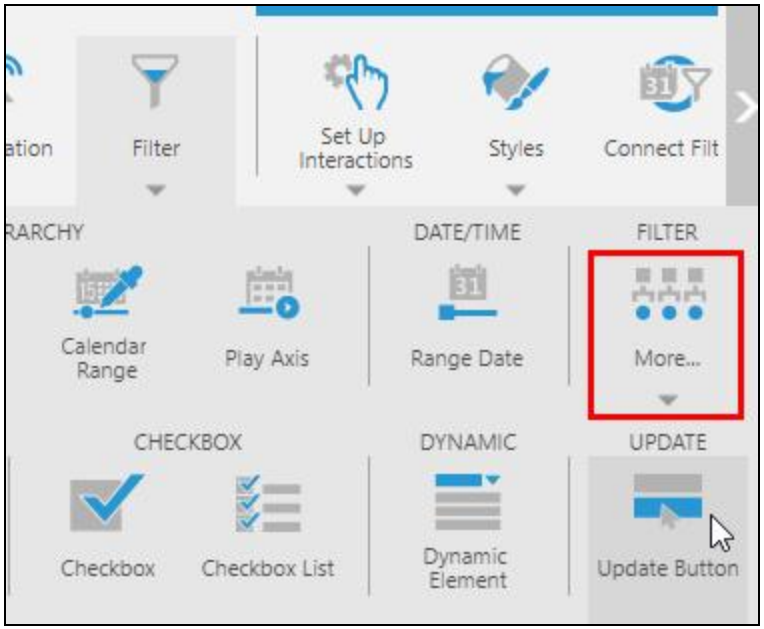
Add a filter and connect it to one of the hierarchies. In this example, it is connected to the slicer.



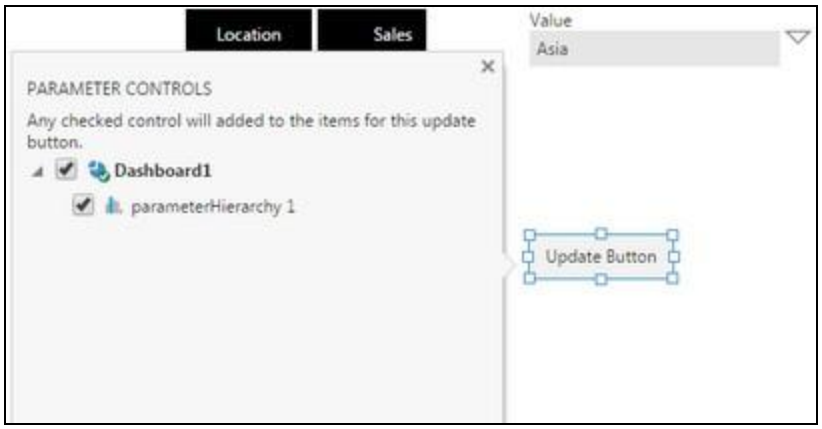
Note that without the update button, data is retrieved right away after changing the filter:



Click Filters in the toolbar and choose **Update Button** (found under **More**).



If the view you're editing is a dashboard, use the checkboxes in the **Parameter Controls** popup to assign the filters that should wait for this button to be clicked before being applied.



Note: When used in a report, scorecard, or small multiple, all filters present in the view are automatically linked to the update button and you cannot choose individual filters.

To change the button text, choose **Edit Button Text** in the toolbar or double-click the button.

Testing

Change the filter value. Notice that data is not retrieved right away.

Click the update button to see the changes to the filtering applied.

For more information, see:

- Video: Filters
- Adding Filters
- Sorting and Filtering a Report or Scorecard
- Design Overview
- Create a Metric Set and Add a Filter

Use a Drop Down List to Switch Metric Sets With Scripting

This applies to: *Managed Dashboards, Managed Reports*

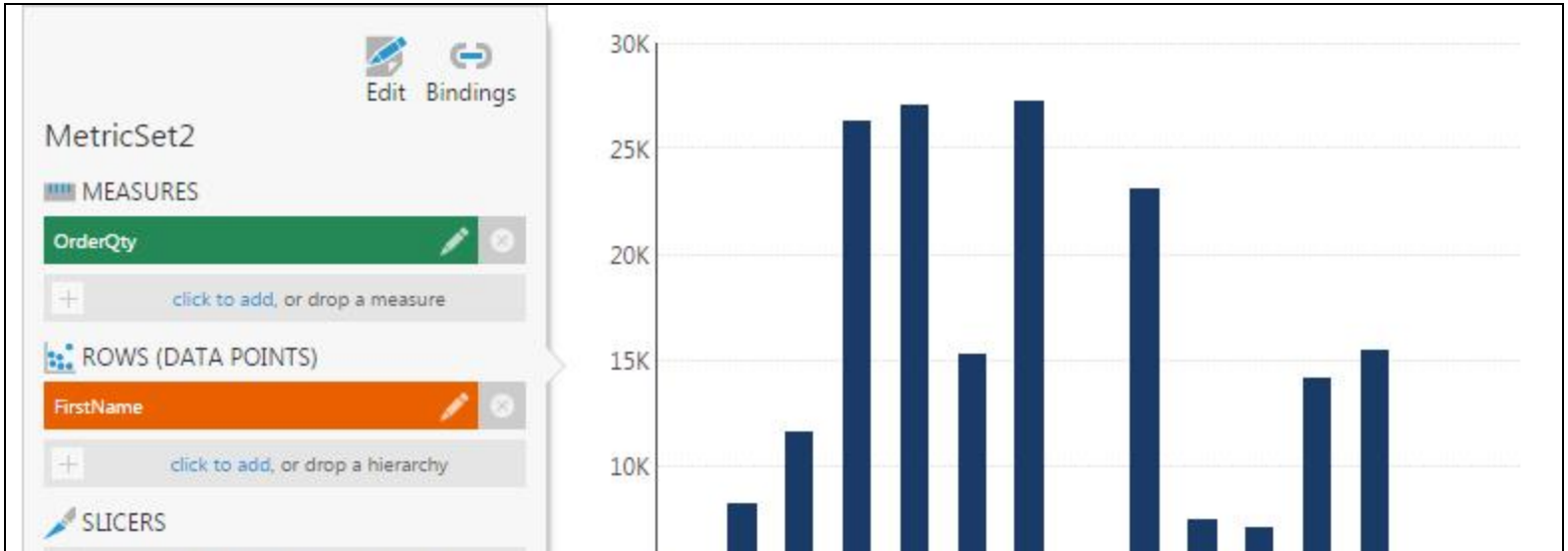
Use Symphony to program a drop down list component to switch the metric sets displayed by a chart. We will use JavaScript that runs in the drop down list's **Selection Changed** action.

Keep in mind that there are options to accomplish a similar result without script by setting up charts in two dashboard layers and using Change Layer actions. The purpose of this article is to demonstrate how additional custom functionality is possible using Symphony APIs.

Create Metric Sets

For this example, create two metric sets that will be switched using the drop down list.


Edit
Bindings
MetricSet1
MEASURES
OrderQty
click to add, or drop a measure
ROWS (DATA POINTS)
ProductID (Product)
click to add, or drop a hierarchy
SLICERS
click to add, or drop a hierarchy
COLUMNS (SERIES)
click to add, or drop a hierarchy
80K
70K
60K
50K
40K
30K
20K
10K
0K
Accessories
Bikes
Clothing
Components
MetricSet1

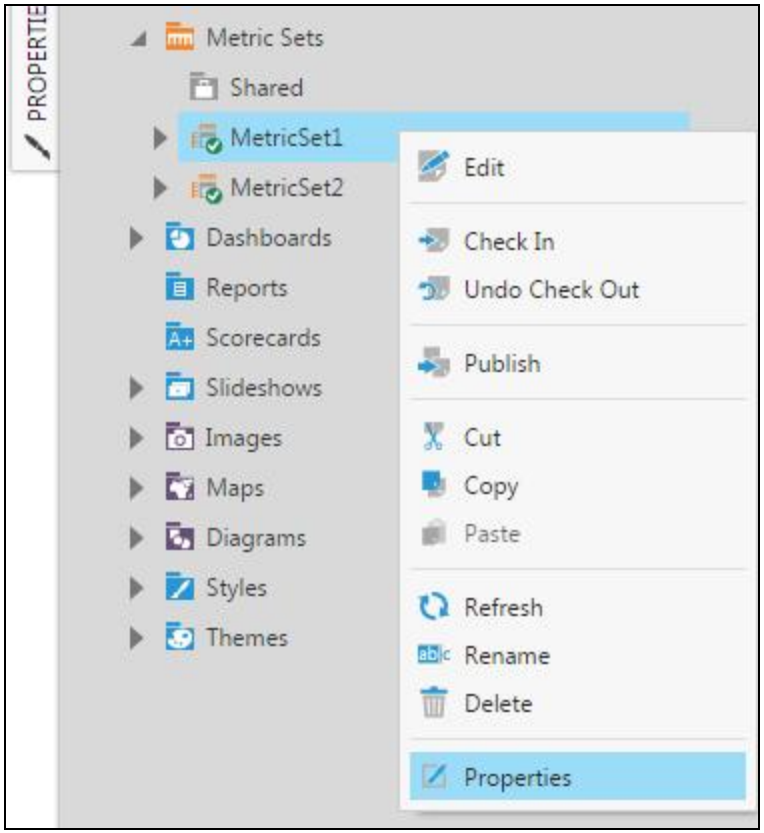

Edit
Bindings
MetricSet2
MEASURES
OrderQty
click to add, or drop a measure
ROWS (DATA POINTS)
FirstName
click to add, or drop a hierarchy
SLICERS
30K
25K
20K
15K
10K

Record the IDs of the Metric Sets

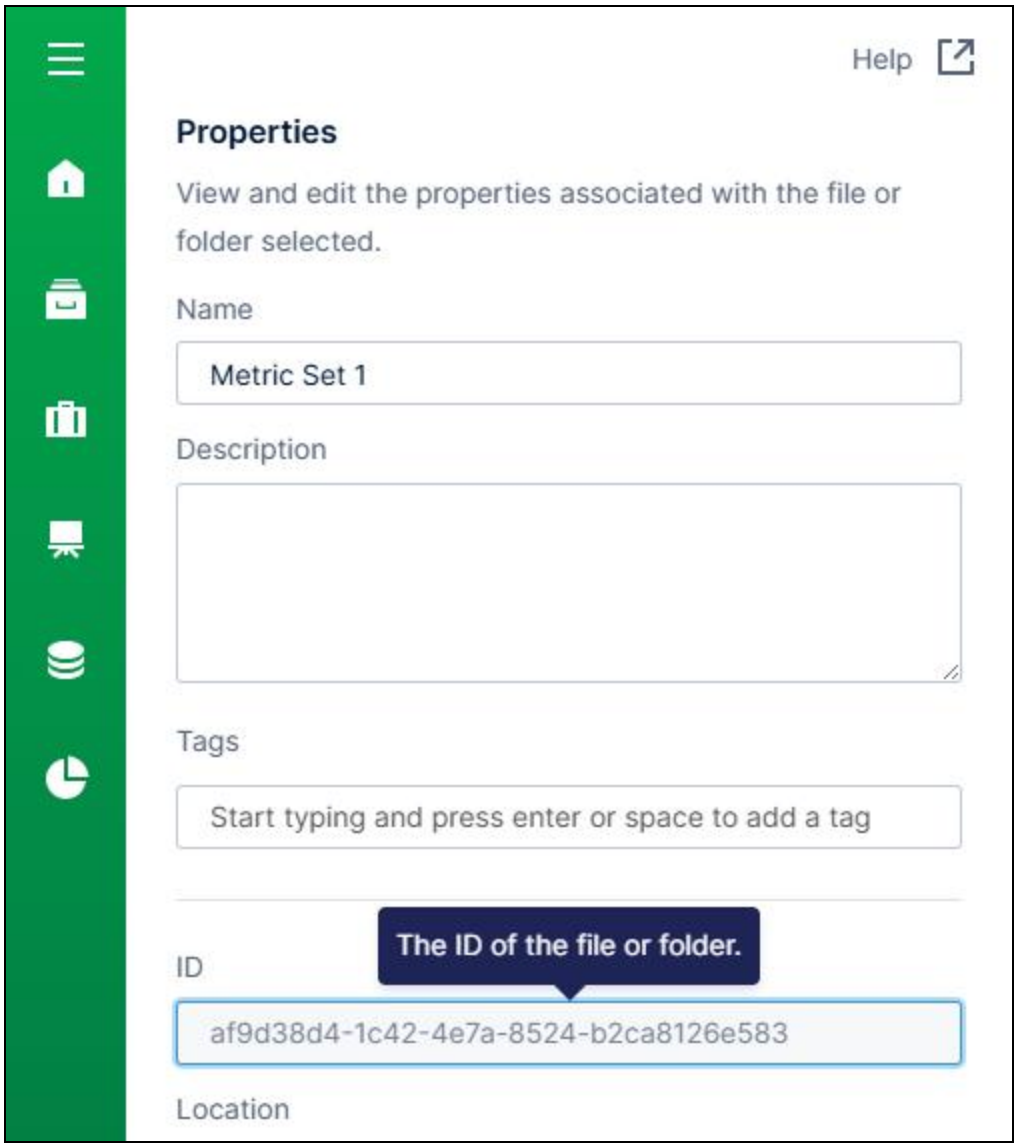
In order to reference the metric sets from script, you'll need to get their IDs.

For each metric set, go to the **Explore** window and locate it in the *Metric Sets* folder.

Right-click on the metric set, and then select **Properties** from the menu.



In the *Properties* dialog, locate the **ID** field and copy the text to the clipboard. You can paste the two IDs into another application now, or copy them the same way later while editing the dashboard.

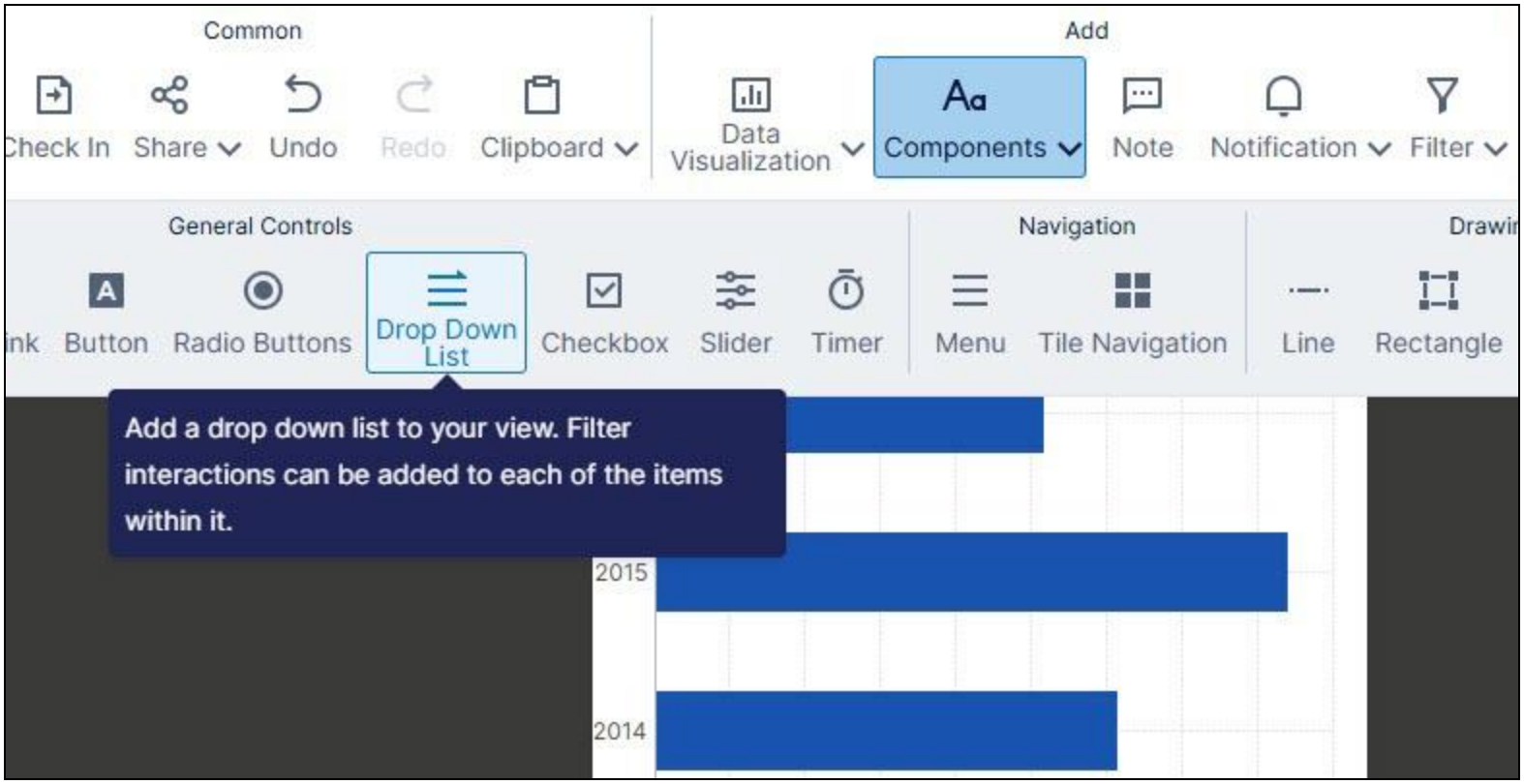


Design the Dashboard

Create a new dashboard, then drag the first metric set from the **Explore** window to the canvas. The metric set appears as a chart visualization named *chart1*.

Add a Drop Down List

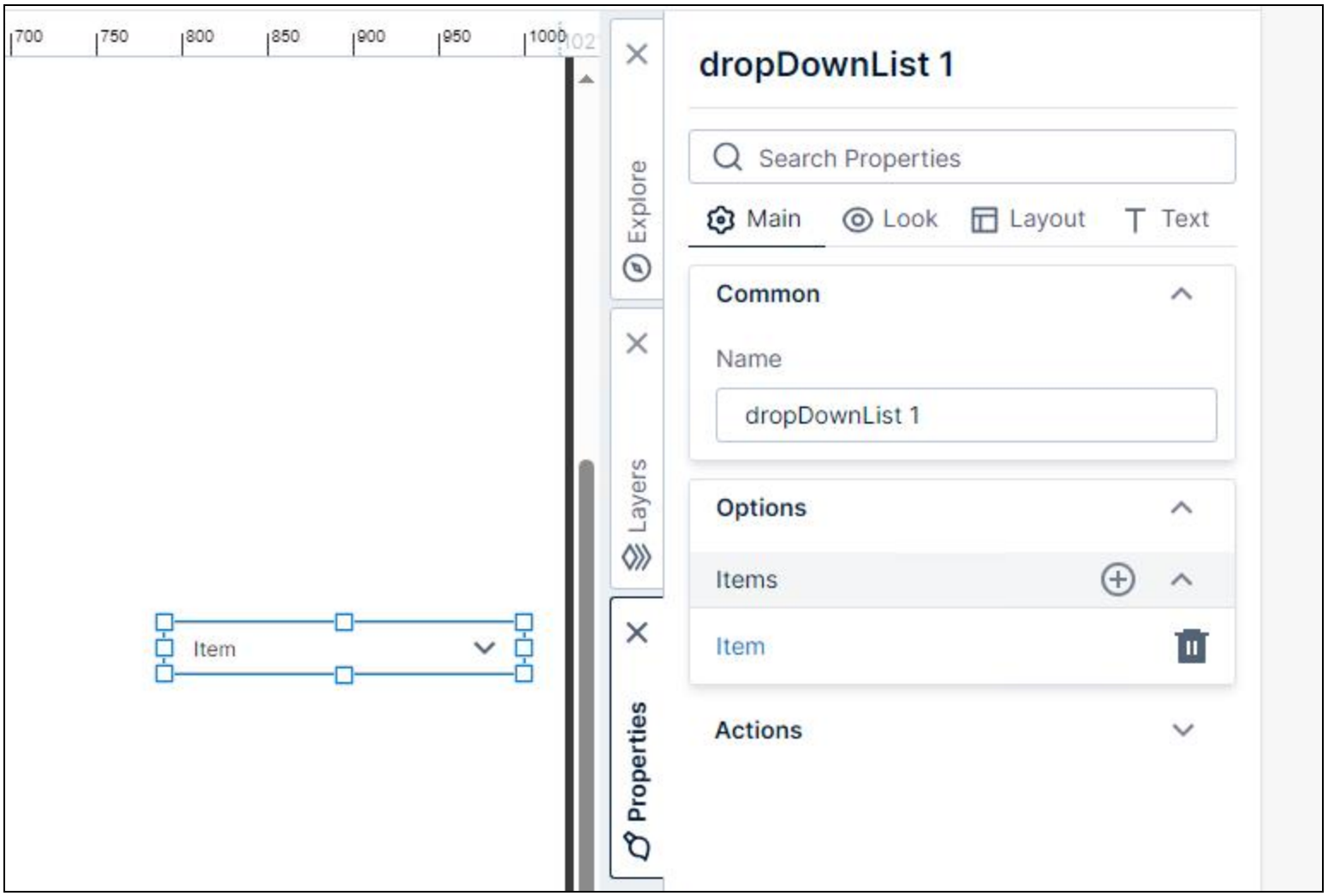
Next, click **Components** in the toolbar, then select the **Drop Down List** component. A drop down list component is added to the dashboard canvas.



Configure Drop Down List Items

Select the drop down list on the canvas, and then open the **Properties** window.

You'll see that the drop down list is configured with a single item by default. Select the item to modify it.

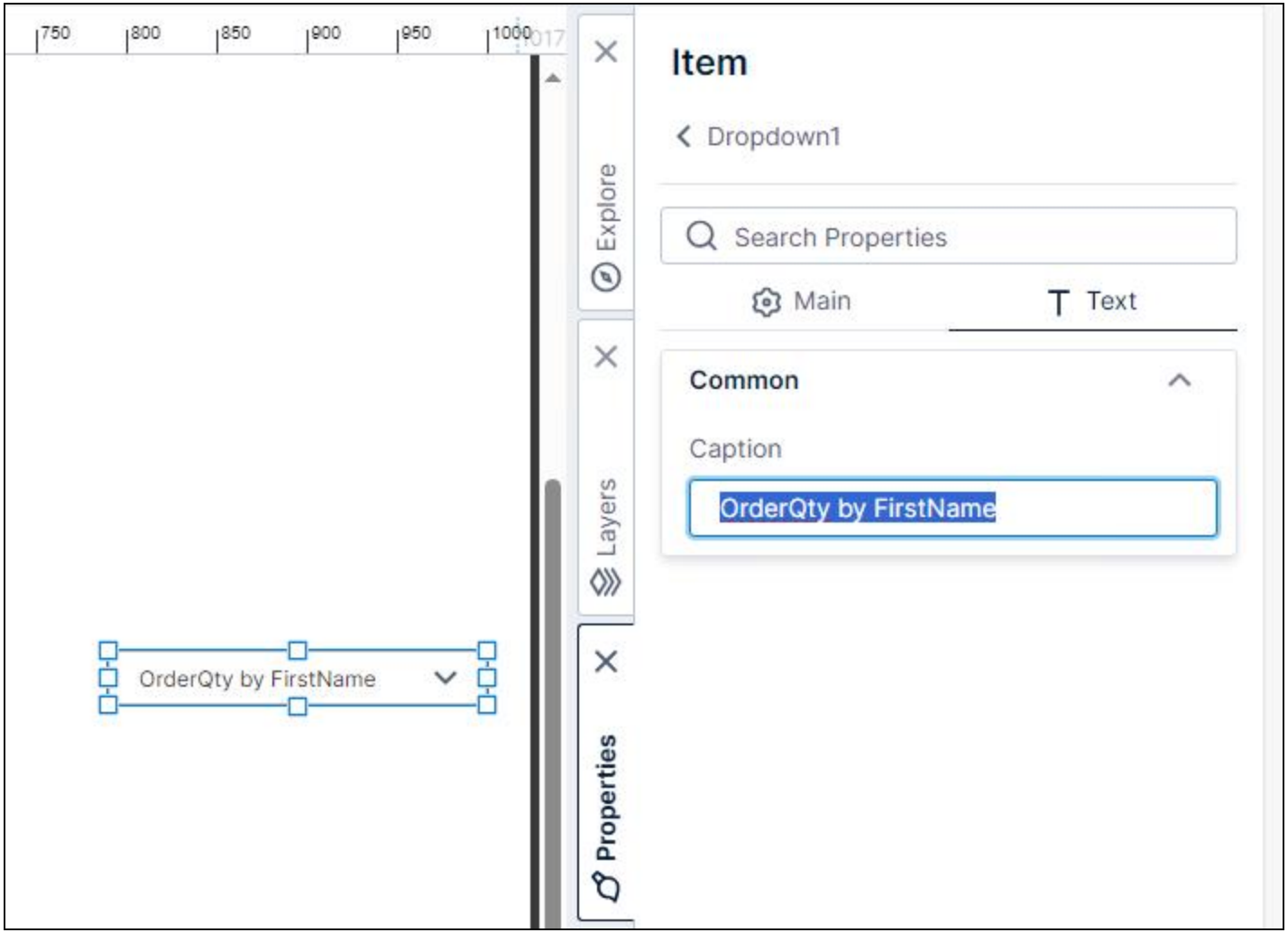


In the **Main** tab, paste the ID from the first metric set into the `Value` property. Then, select the **Text** tab and set the `Caption` property to **OrderQty by Product**.

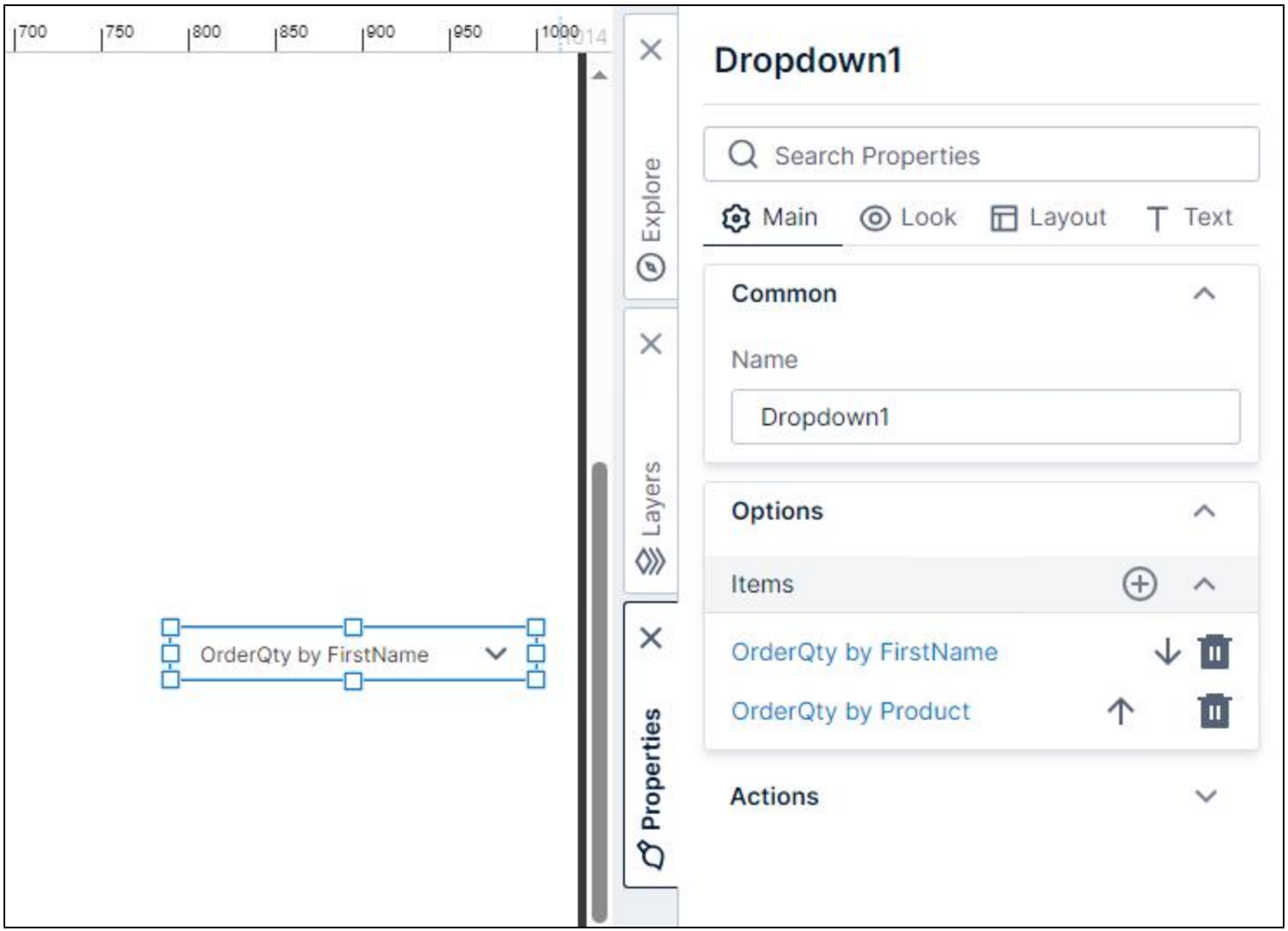
Click the back button at the top of the properties to go back to the list of items.

Click the **+** button to add a second item, then click the new item to edit it.

In the **Main** tab, paste the ID from the second metric set into the `Value` property. Then, click the **Text** tab and set the `Caption` property to **OrderQty by FirstName**.



Click the back button at the top of the properties to go back to the list of items. The drop down list has two items configured now.



Add a Script

In the **Main** properties for the drop down list, scroll down to see the Actions section. Locate the **Selection Changed** action and expand it.

Select the **+** button to add a script to handle this action. Select the newly added **script 1** to open the **Script Editor**.

Copy and paste the following JavaScript code into the Script Editor.

```
// Get the metric set ID value from the drop down list.
var metricSetId = dropDownList1.value;

// Update the chart with the selected metric set.
chart1.metricSetBindings.length = 0;
chart1.generateMetricSetBinding(metricSetId, true);
```

Select **Build** to check for any script errors. Script Editor auto-saves your changes as you type.

Re-Connect Filters

If the metric sets have a filter connected to it, the connection is lost when switching the metric sets. The filter can be re-connected by adding a **done()** callback to the generateMetricSetBinding() method.

An update to the JavaScript code above will look like this:

```
// Get the metric set ID value from the drop down list.
var metricSetId = dropDownList1.value;
var parameterName = "viewParameter1"; // the Script Name of the view parameter
var parentView = this.parentView;

// Update the chart with the selected metric set.
chart1.metricSetBindings.length = 0;
chart1.generateMetricSetBinding(metricSetId, true).done(function (result) {

	// Get the view parameter
	var viewParameter = parentView.control.getViewParameterByName(parameterName);

	// Find the metric set's hierarchy usage
	var element = result.metricSet.rowHierarchies[0];

	// Re-link the filters
	var elementParameterLink = new dundas.view.ElementParameterLink({
		"adapterId": chart1.id,
		"metricSetBindingId": result.metricSetBinding.id,
		"elementUsageUniqueName": uniqueName,
		"parameterId": element.parameter.id
	});
	viewParameter.clearElementParameterLinks();
	viewParameter.addElementParameterLink(elementParameterLink);
	viewParameter.refreshAllAdapters(null, parentView);
});
```

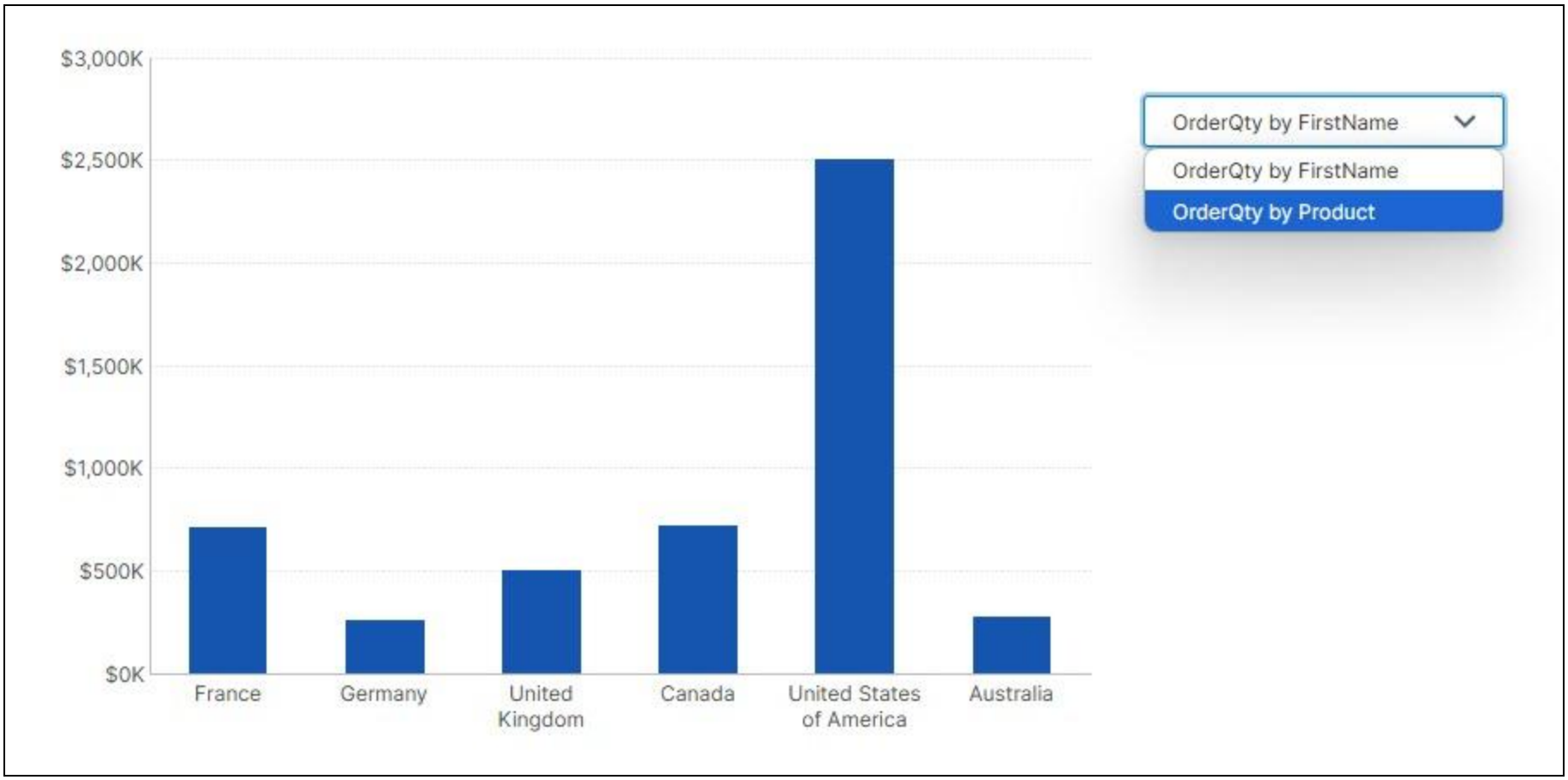
Using Table Visualizations

If your two metric sets are visualized as tables (instead of charts), the script portion that updates the table with the selected metric set is slightly different.

```
// Update the table with the selected metric set.
table1.metricSetBindings.length = 0;
table1.control.columns = [];
table1.control.rowHeaderColumns = [];
table1.generateMetricSetBinding(metricSetId, true);
```

Using the Drop Down List

Choose **Sandbox View** in the toolbar to test the script in a new tab without saving results of running the script, and then use the drop down list to choose which metric set to display in the bar chart.



For more information, see:

- JavaScript API
- Using Interactions
- Adding a Dynamic Element Filter
- Using the Script Editor

Symphony Managed Dashboards and Reports Config File

This applies to: *Managed Dashboards, Managed Reports*

This article describes the `dbi.config` file used to define pieces of information needed to start the application. This is initially configured as part of the installation process.

Application Database Connection String

The connection string that Symphony uses to connect to its application database is defined in the **connectionStrings** element:

```
<connectionStrings>
  <add
  name="AppConnectionString"
  connectionString="Data Source=localhost;Initial Catalog=&quot;Dundas BI Instance1&quot;;User ID=DundasUser;Password=1234"
 />
</connectionStrings>
```

If this section is encrypted, it can be decrypted using the dt command line tool. See Using the dt Command Line Tool.

Symphony can use either a Microsoft SQL Server database or a PostgreSQL database to store its application data, and connection strings are defined differently for each. For syntax details including available parameters/keywords, see Microsoft Docs for SQL Server or Npgsql for PostgreSQL.

Note: The warehouse database connection string can be changed from the configuration settings while logged in as an administrator. Its connection string uses the same syntax.

Application Storage Type

The **ApplicationStorage** key should be set to the type of database you are using. Below is an example of how this would look for SQL Server:

```
<!-- The type of database server hosting the Application database.
Can be "SqlServer" or "Postgres". -->
<add key="ApplicationStorage" value="SqlServer" />
```

Setting the Temp Data Path

The temp data path is the path where any temporary application data is saved. If it is not set or set to empty string, the App_Data path is used. The following folders would exist under the folder defined by this setting:

- BLOBFiles
- Logs
- TempLib
- ViewThumbnails
- FileDelivery
- DownloadStaging

```
<!-- The path where any temporary application data is saved.
If not set or set to empty string, the App_Data path is used.". -->
<add key="TempDataPath" value="" />
```

Recycle the Application Pool

For changes to this file to take effect, you need to recycle the application pool in IIS or the website service on Linux.

Complete Example

The following is a complete example of a `dbi.config` file's contents, including some example keys provided as comments for reference:

```
<?xml version="1.0"?>
<configuration>
  <connectionStrings>
    <!-- SQL SERVER -->
    <add name="AppConnectionString" connectionString="Data Source=localhost; Initial Catalog=&quot;Dundas BI
Instance1&quot;;User ID=DundasUser;Password=1234" />

    <!-- PostgreSQL -->
    <!--<add name="AppConnectionString" connectionString="Host=localhost;Port=5432;Database=&quot;Dundas BI
Instance1&quot;;User ID=DundasBIUser;Password=1234;" />-->
  </connectionStrings>
  <appSettings>
    <!-- The type of database server hosting the Application database. Can be "SqlServer" or "Postgres". -->
```

```
    <add key="ApplicationStorage" value="SqlServer" />

    <!-- The path where any temporary application data is saved. If not set or set to empty string, the App_Data path is
used.". -->
    <!--<add key="TempDataPath" value="" />-->
  </appSettings>
</configuration>
```

For more information, see:

- Configuration Settings
- Using the dt Command Line Tool
- How to Recycle the Application Pool

Install and Configure R

This applies to: *Managed Dashboards, Managed Reports*

R is a popular programming language and environment for statistical computing and predictive analysis.

Symphony integrates with an existing R system at the data cube level via two transforms which you can use to supply data to the cube or perform data processing/analysis on other data.

In order to use the R-related functionality in Symphony, you must have access to an existing R server. There are configuration settings in Symphony that allow administrators to specify the details for connecting to an external R server.

Install the R Server

The open source Rserve (Binary R Server) software is available for Unix, Linux, and Windows platforms, however the Windows version is not recommended other than for evaluation purposes due to limitations.

The Windows version of Rserve works in cooperative mode only, which means that only one connection at a time is allowed and all subsequent connections share the same namespace. For example, using both an R Data Generator transform and an R Language Analysis transform in the same data cube will result in an error because of this limitation.

Unix / Linux

For details on installing Rserve on Unix or Linux systems, see https://rforge.net/Rserve/doc.html#intro.

Windows

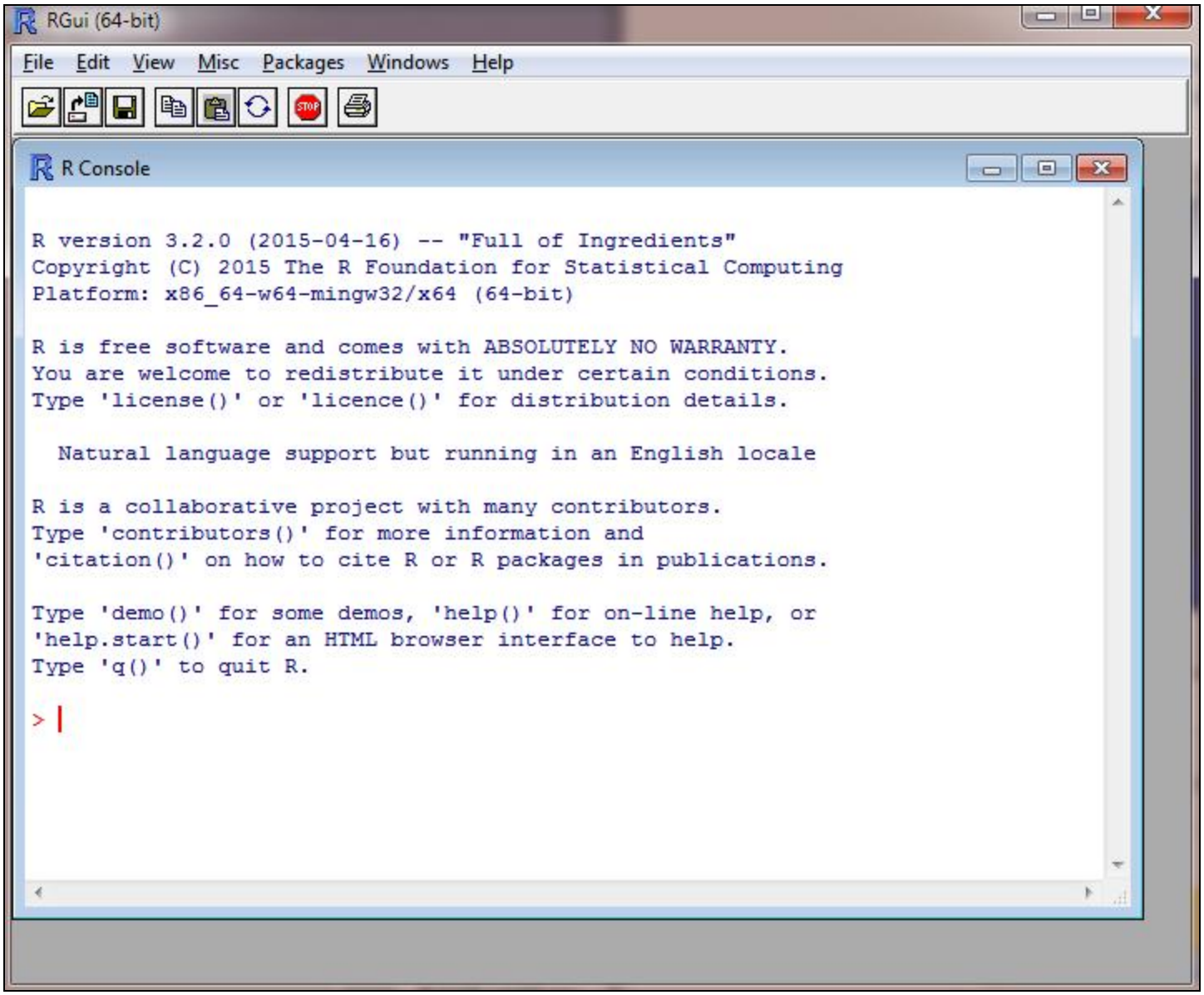
If you want to run Rserve on Windows with Symphony (for testing or non-production purposes), first download and install R for Windows in your environment.

This installer package will install the R binaries to a folder similar to: **C:\Program Files\R\R-3.2.0\bin\x64**

Once R for Windows is installed, add the above folder path to your Windows Path environment variable. This will allow the Rserve program to locate `R.dll`.

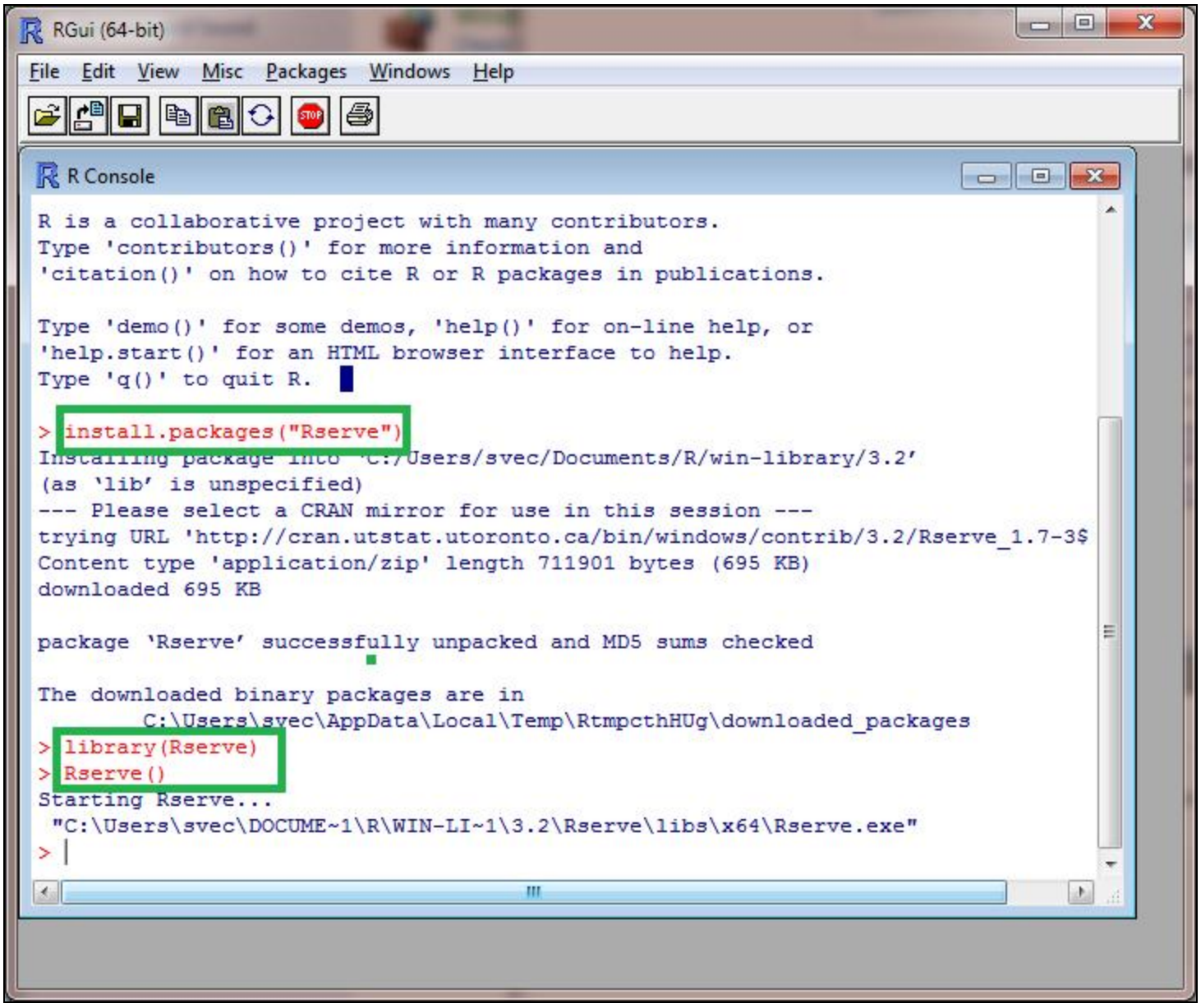
Next, go to your Start Programs menu and expand the newly installed R folder.

Click **R x64 3.2.0** (or similar) to launch the **R Console**.



In the R Console, type the following commands in order to download and install Rserve (this is the R server) and run the program.

```
install.packages("Rserve")
library(Rserve)
Rserve()
```



The R server is now running and you can use the R transforms in Symphony as described in the articles for the R Data Generator and R Language Analysis transforms.

Note: To run Rserve without the R Console, open a Command Prompt window and start the Rserve.exe program. You can also copy the Rserve.exe program to your Windows Startup folder to have it run automatically when Windows starts.

Configure Rserve Connection Settings

Once your R server is running, you'll need to update the Rserve Connection settings in Symphony in order to properly connect to the R server.

For more information, see:

- R Data Generator
- R Language Analysis
- Configuration Settings

Install Python

This applies to: *Managed Dashboards, Managed Reports*

Python is a popular programming language that can be used for advanced analytics and data modeling. Symphonyintegrates with Python at the data cube level via two transforms, which you can use to supply data to the data cube or perform data processing and analysis on other data.

Important: The minimum supported version of Python for Managed Dashboards and Reports is Python 3.8.

Install Python

To install Python on Windows, download and run the installer for the 64-bit version of Python.

On Linux, the standard default version of *python3* provided by supported Linux distributions and versions is normally used except where noted otherwise in the system requirements for your version of the software.

The process on Windows is generally:

1. Run the Python setup.
2. Select **Customize installation** and **click Next** on the **Optional Features** page.
3. Select **Install for all users** on the **Advanced Options** page and click **Install**.

Install Packages

Packages other than those installed automatically by Symphony must be installed by an administrator on the server that runs Symphony before they can be used in Python transforms.

For example, to install the common **pandas** package, which includes other packages such as NumPy:

1. Open the command prompt or terminal on the Symphony server as an administrator (e.g., right-click on Command Prompt in the start menu and choose **Run as administrator**).
2. You may need to navigate to the particular **Scripts** folder corresponding with the version of Python used by Symphony. For example:

```
cd C:\Program Files\Python38\Scripts
```

3. Use **pip** to install the package. For example:

```
pip install pandas
```

- The Python-related transforms in Symphony require a result to be returned by the Python script, which can be represented as a table. This includes results generated using the NumPy and Pandas packages for data analysis.
- You may be required to restart the server after installing Python.

For more information, see:

- Python Data Generator
- Python Analysis

Using A Managed Dashboard and Managed Reports Gateway

This applies to: *Managed Dashboards, Managed Reports*

You can use a gateway to connect to your on-premise data from an instance of Symphony.

Once the gateway is installed on your local network and configured in Symphony, users can create data connectors to access the following on-premise data sources via the gateway:

- dBase (DBF) files
- Flat files
- IBM Db2 (v24.3 and later)
- JDBC
- MemSQL
- Microsoft Excel
- Microsoft SQL Server
- MySQL
- OData
- ODBC (install ODBC drivers to use for additional data sources)
- Oracle database
- PostgreSQL
- Teradata
- XML files

Note: For the full list of data sources supported without a gateway, see the system requirements. You don't need a gateway for cloud or publicly accessible data sources, or for uploaded files.

System Requirements

Some data sources may require drivers to be installed on the same computer as the gateway. The gateway also has minimum requirements for the operating system and for .NET that are similar to the server requirements for a Symphony instance, but the minimums listed for evaluation use can also be suitable for a gateway.

Note: The gateway should be installed on a computer that remains online in order to prevent errors when using data connectors in Symphony that use this gateway.

Linux Installation

The Symphony Deployment wizard is used to create and manage gateways on Linux.

Follow the instructions in Installing Symphony on Linux for downloading and installing the Deployment wizard for your distribution of Linux, but do so on a computer that can be used to provide access to your data sources. Rather than launching the wizard to create an instance, you will create a gateway as shown below.

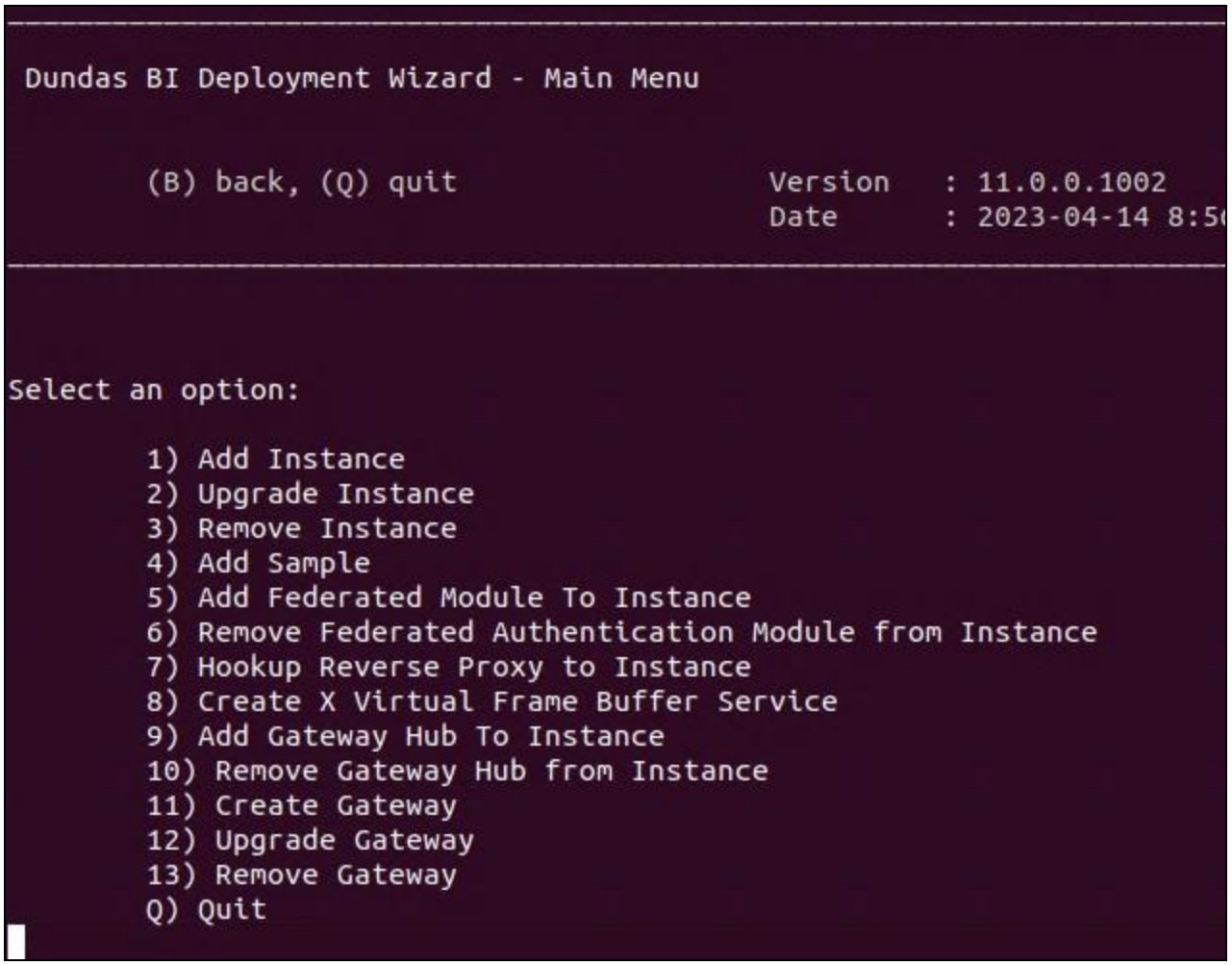
When a new version of Symphony becomes available, you can download and install the new version of the Deployment wizard, then run the step to upgrade your existing gateway.

Managing Gateways

Launch the Symphony Deployment wizard in a terminal using the --gateway option. For example:

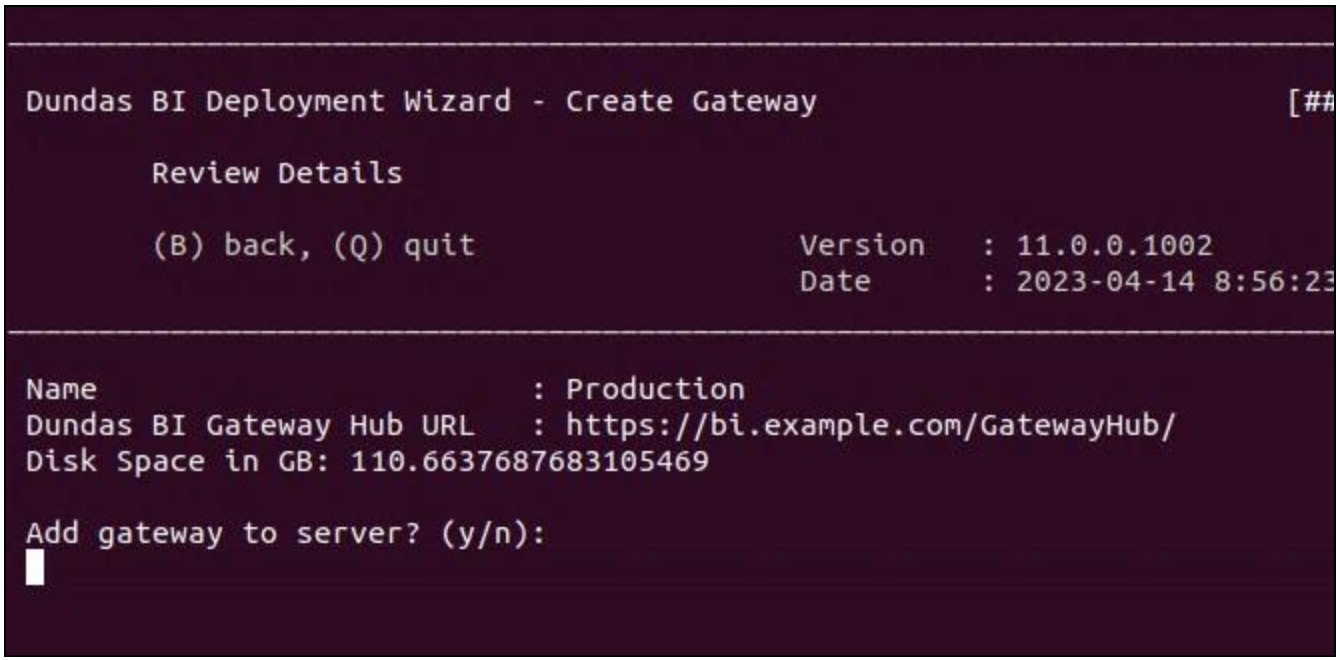
```
sudo dundasbi --wizard --gateway
```

To create a new gateway, type the number indicated for the step **Create Gateway** followed by the enter key. There are also steps available to **Upgrade Gateway** and **Remove Gateway**.

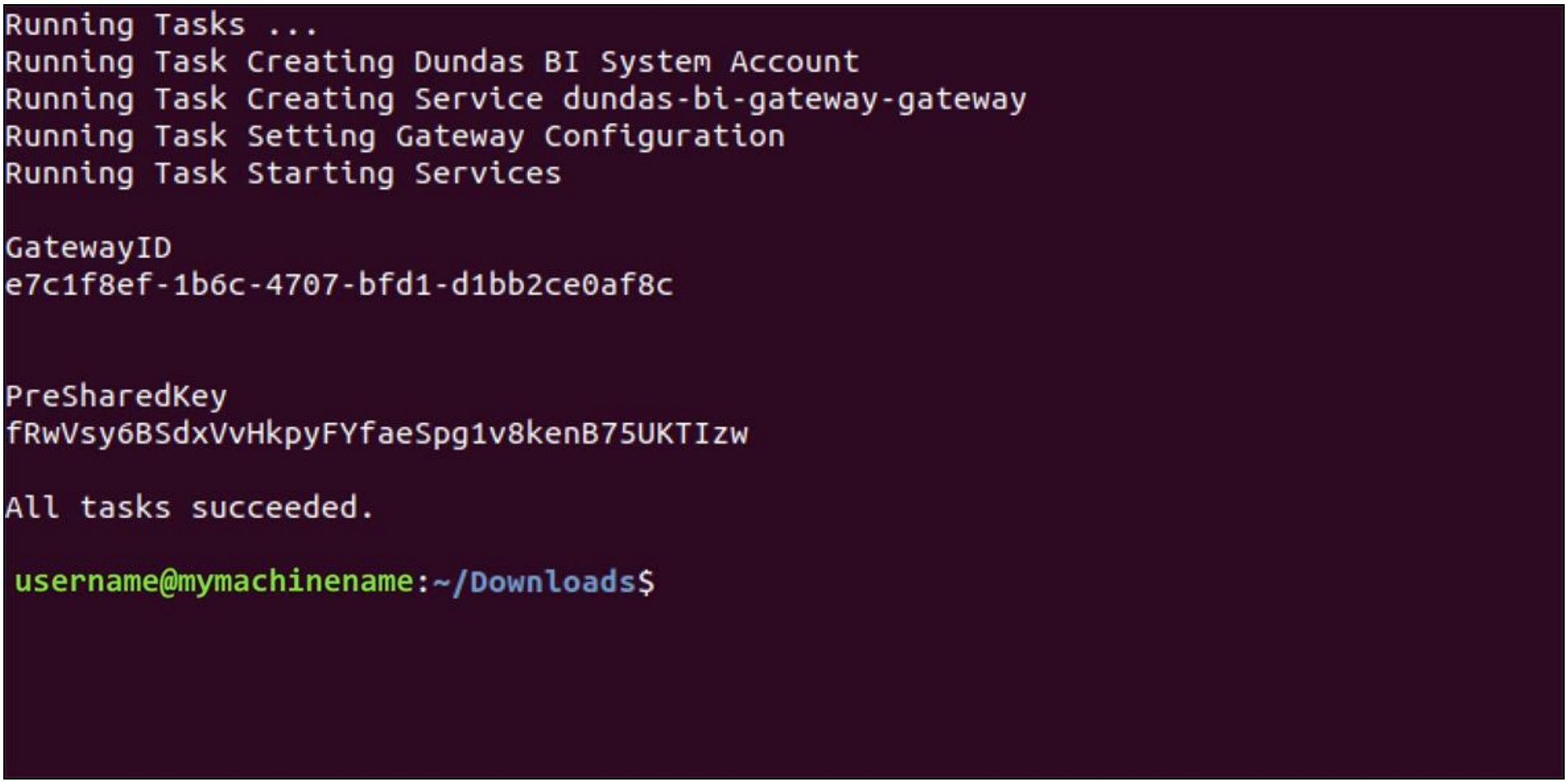


A prerequisites check may run. Follow the prompts, or type **y** and the enter key to continue if the check passed.

You will be prompted to enter a **Name** for the Symphony instance you want to use this gateway (for your reference), followed by the **SymphonyGateway Hub URL** based on the address you use to access Symphony with `/GatewayHub/` added. For example, if you can navigate to Symphony at `https://symphony.example.com/`, enter `https://symphony.example.com/managed/GatewayHub/`.



These details will be displayed back to you next and you can type **y** followed by the enter key to confirm and create the gateway.



Once the tasks are complete, note the **GatewayID** and **PreSharedKey** values indicated in your terminal, as you will need these when configuring the gateway in Symphony. If you need to find them again later, they are also defined within the **ConfigOverride.xml** file in the **App_Data** folder.

Once created, your gateway's **App_Data** directory is located at:

```
/usr/share/dundas/bi/Gateways/Files/{InstanceName}/App_Data
```

In some cases, drivers may need to be installed in subdirectories created under this directory if indicated in links from the data sources section of Symphony's system requirements. There is also a **Logs** subdirectory with files that may record errors.

A service is also installed that needs to continue running in order to provide access to your data sources for Symphony. You can control or check the status of this service using the following commands (you may need to use the sudo command in front):

Command	Result
systemctl stop dundas-bi-gateway-{InstanceName}.service	Stops the Symphony gateway service.
systemctl start dundas-bi-gateway-{InstanceName}.service	Starts the Symphony gateway service.
systemctl restart dundas-bi-gateway-{InstanceName}.service	Restarts the Symphony gateway service.
systemctl status dundas-bi-gateway-{InstanceName}.service	Get the Symphony gateway service status.

On Docker

Like other Docker images, the gateway can run as a Docker container using the image dundas/dundas-bi-gateway. This allows you to run on other machines that can access your data sources besides the supported Windows and Linux environments if compatible .

The gateway is configured using environment variables set when starting the container:

Environment Variable	Description
DUNDAS_BI_EXTERNAL_APPLICATION_URL	The URL to your instance, e.g.: https://symphony.example.com/
DUNDAS_BI_GATEWAY_ID	A unique ID to identify this gateway. You could generate a UUID for this value using uuidgen in a terminal or online.
DUNDAS_BI_GATEWAY_PRESHARED_KEY	A strong pre-shared key value matching the one you configure in your Symphony instance as described below to ensure only your gateway is granted access. Generate one or find tips here: https://cloud.google.com/network-connectivity/docs/vpn/how-to/generating-pre-shared- key
DUNDAS_BI_AGREE_TO_TERMS_OF_GATEWAY_EULA	You must accept the terms of the Gateway Application License Agreement to run the gateway. Set to **true** to confirm your acceptance.
DUNDAS_BI_PRINT_CONFIG_OVERRIDE	(Optional) Prints configuration XML text containing the gateway ID and pre-shared key you specified, as confirmation.
DUNDAS_BI_GATEWAY_JDBC_DRIVER_PATH	(Optional) The path to JDBC driver files you intend to use with the JDBC data provider, mounted as a volume or added by extending the image.

Set up the Gateway Manifest in Symphony as described in the following section first using the gateway ID and pre-shared key you plan to set as environment variables, and then start up your gateway container.

Here is an example for starting the container using docker run:

```
docker run -it \
-e DUNDAS_BI_EXTERNAL_APPLICATION_URL="https://dundas.example.com/" \
-e DUNDAS_BI_GATEWAY_ID="b3f999b0-5dc5-4790-bc9c-3a2e97d8701f" \
-e DUNDAS_BI_GATEWAY_PRESHARED_KEY="h9LR6yUxFpdXAserzdXjVxB4DQdNl3rTdKDj2Mxr" \
-e DUNDAS_BI_AGREE_TO_TERMS_OF_GATEWAY_EULA="true" \
dundas/dundas-bi-gateway
```

Set Up Symphony

Symphony must be set up for gateways before you can use them, by installing a gateway hub and configuring it for your on-premise gateway.

Install the Gateway Hub

Add the gateway hub to your Symphony instance on the server where Symphony was installed, which will add support for connecting with your gateway running on-premises.

Use the same installation tool you used to install your Symphony instance:

- When Symphony is installed directly on Linux, run the Deployment Wizard and choose the option to add the gateway hub. If you are using your own reverse proxy configuration rather than the Deployment wizard's provided one, it will need to be updated for the gateway hub.

- For Kubernetes deployments, use appropriate Helm chart options. For example:

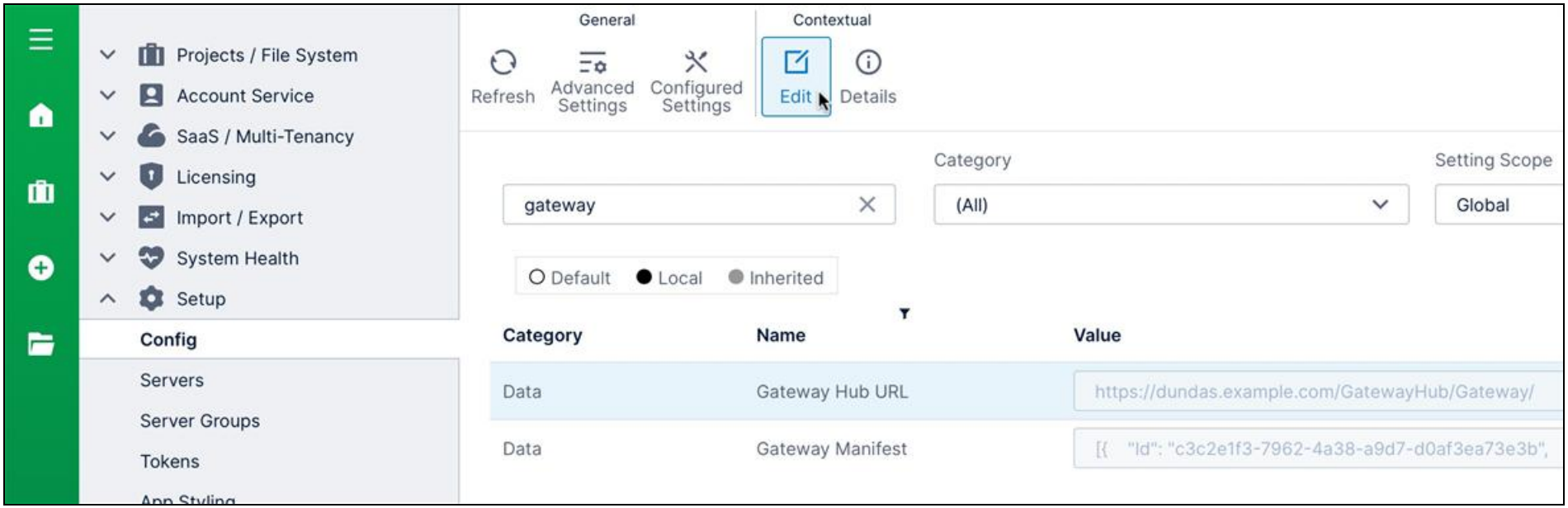
```
mananged:
  dundas:
    bi:
      gatewayhub:
        # enables the gatewayhub deployment.
        enabled: true

        # The gatewayhub service.
        service:
          enabled: true
```

- If you are running Symphony containers yourself with Docker directly, configure your reverse proxy for the dundas-bi-gateway-hub container to be accessible as the subpath `/GatewayHub` under your instance.

Configure Your Gateway

Log into Symphony as an administrator user and access the Administration area from the main menu.

Click to expand **Setup** and then to navigate to **Config** to access Symphony's configuration settings.



Find or search for the Gateway Manifest setting, located in the Data category, then double-click it or select it and click Edit.

In the configuration dialog, click Edit value, and then enter your configuration into the text area below as a JSON array like the following:

```
[{
    "Id": "c3c2e1f3-7962-4a38-a9d7-d0af3ea73e3b",
    "DisplayName": "Office",
    "PreSharedKey": "Ut5x8VGGPolxpylVmF5h5iVgmgDlphkJXiRUiTnf",
    "ipAllowList": "192.0.2.0/24"
}]
```

If you have multiple gateways, you can provide multiple comma-separated objects as defined within curly braces { }. Each object should provide the following details for a gateway:

- **Id** - The gateway ID provided to you earlier, from your gateway's Config Override File or from your terminal after adding the gateway on Linux.
- **DisplayName** - The name used to identify this gateway when using it in a new data connector.
- **PreSharedKey** - The pre-shared key value provided to you earlier along with the gateway ID.
- **ipAllowList** - (optional) If specified, incoming gateway connections to Symphony are only accepted from the specified comma-separated list of IP addresses and/or IP address ranges. An IP address range may be specified using CIDR notation (e.g., 10.10.10.0/24) or as two addresses separated by a hyphen (e.g., 10.10.10.0-10.10.10.255).

> **Note:** `ipAllowList` is optional and may be left out, but can improve security if specified.

Click to **Save** in the dialog for the configuration to take effect.

Depending on your scenario, you may also wish to review the other Gateway configuration settings available, such as the min/max pool size and timeout settings. These can be adjusted as necessary in case of issues connecting to data through the gateway.

Configure the Gateway Hub

In configuration settings, ensure that the Gateway Hub URL is set to the `/GatewayHub/Gateway/` subpath under your Symphony instance. For example, if the address for Symphony's administration pages including configuration settings begin with `https://symphony.example.com/managed/Admin/`, the gateway hub URL should be `https://symphony.example.com/managed/GatewayHub/Gateway/`.

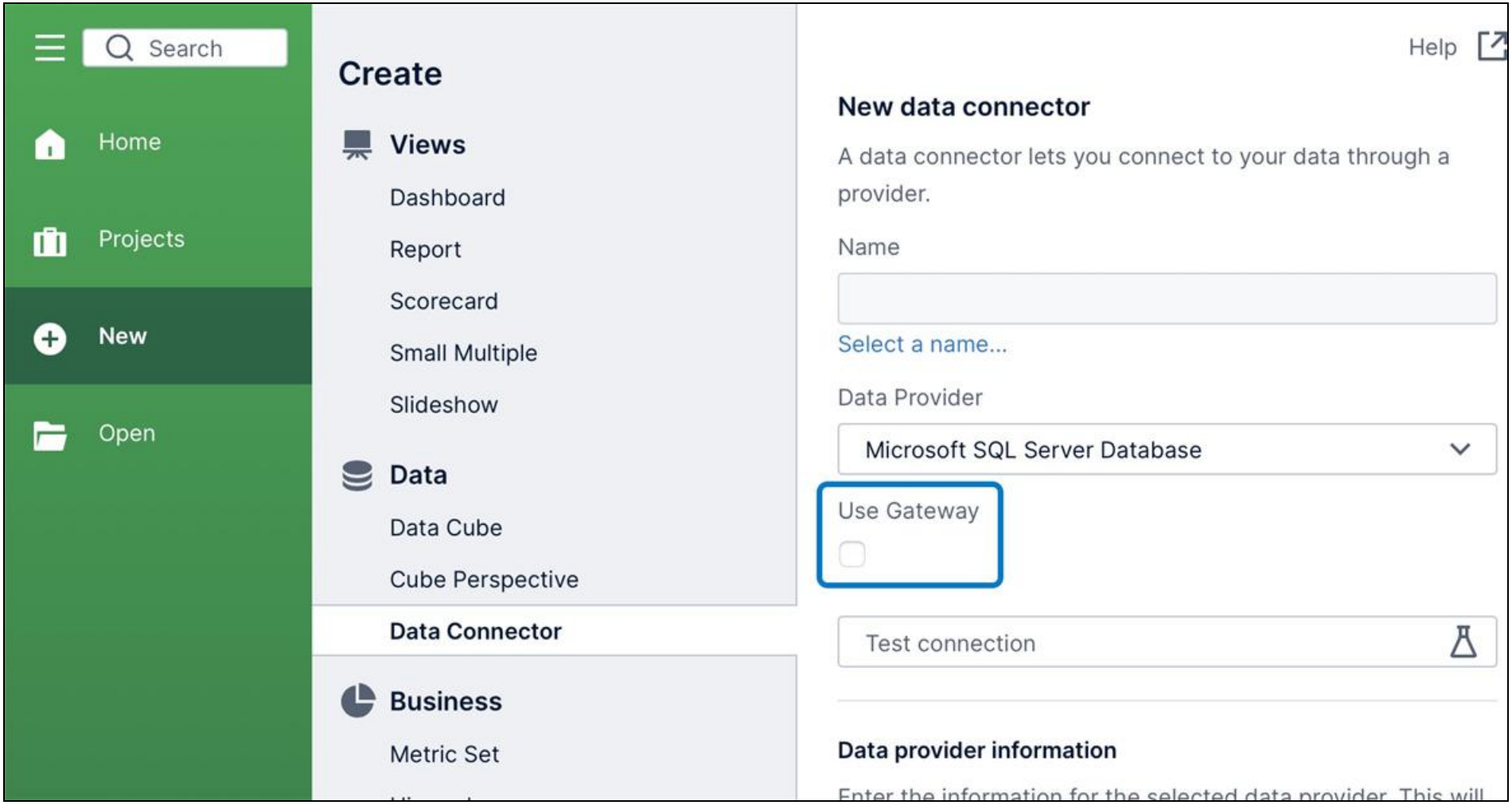
Restart Services

You may need to restart the website's application pool, or container for the changes to take effect once the gateway and gateway hub are configured.

Connect to Your Gateway

Once your gateway and gateway hub are set up and any necessary drivers are installed, users can use that gateway to connect to on- premise data sources in data connectors.

When creating or editing a data connector with Data Provider set to a supported data source, select the **Use Gateway** checkbox and then select your **Gateway** from the list that appears.



Now you can enter the connection details your gateway should use to access your data source.

Security

- The gateway uses only outgoing connections, so your environment where the gateway is installed does not need to support incoming or inbound connections for the gateway.

- Your gateway connects to the Symphony URL you specified when adding it. Your Symphony instance should be configured to use HTTPS encryption to secure this traffic with a URL that starts with https://.
- Trust is established between your Symphony instance and gateway through the pre-shared key used during configuration, so that your instance will only accept incoming connections from your gateway.
- When data source connection details need to be communicated to your gateway, a pre-shared key is used with AES encryption and a SHA-512 hash function.
- You can further improve the security for incoming gateway connections to Symphonyby specifying ipAllowList in your gateway manifest.

For more information, see:

- Connect to Data and View it on a Dashboard
- Configuration Settings
- Configuration Best Practices

How to Enable SQL Server Authentication

This applies to: *Managed Dashboards, Managed Reports*

SQL Server authentication is recommended for connecting Symphony to its application and warehouse databases for security reasons. This article explains how to enable SQL Server authentication, and how to use it in your Symphony environment.

This applies when installing or upgrading Symphony when it uses Microsoft SQL Server to store its own data. Within Symphony, users can then connect, analyze, and visualize data from a variety of other data sources or use other authentication methods.

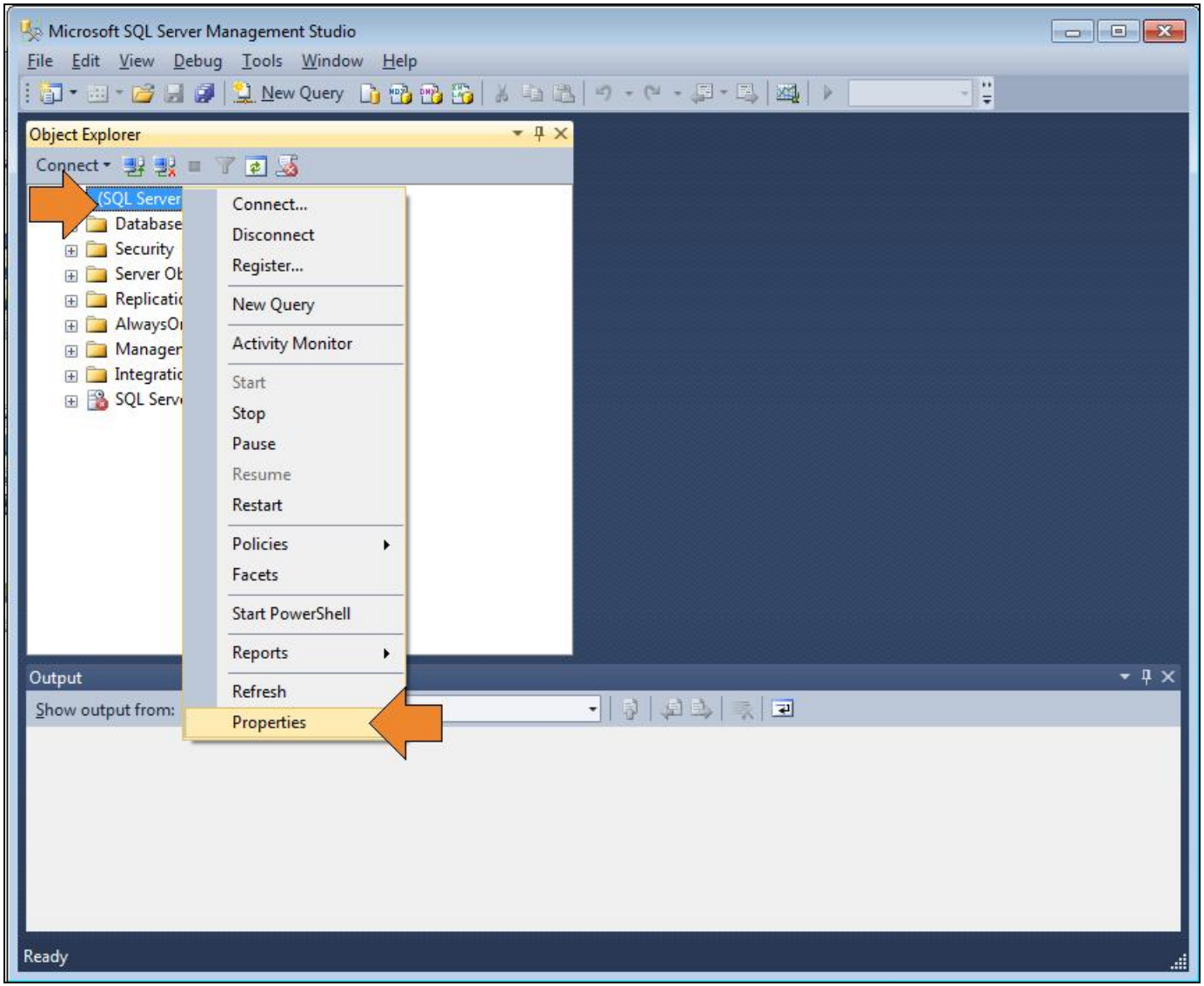
Note: You should have a basic understanding of using SQL Server Management Studio to deploy in your environment.

Enabling SQL Server Authentication through SQL Management Studio

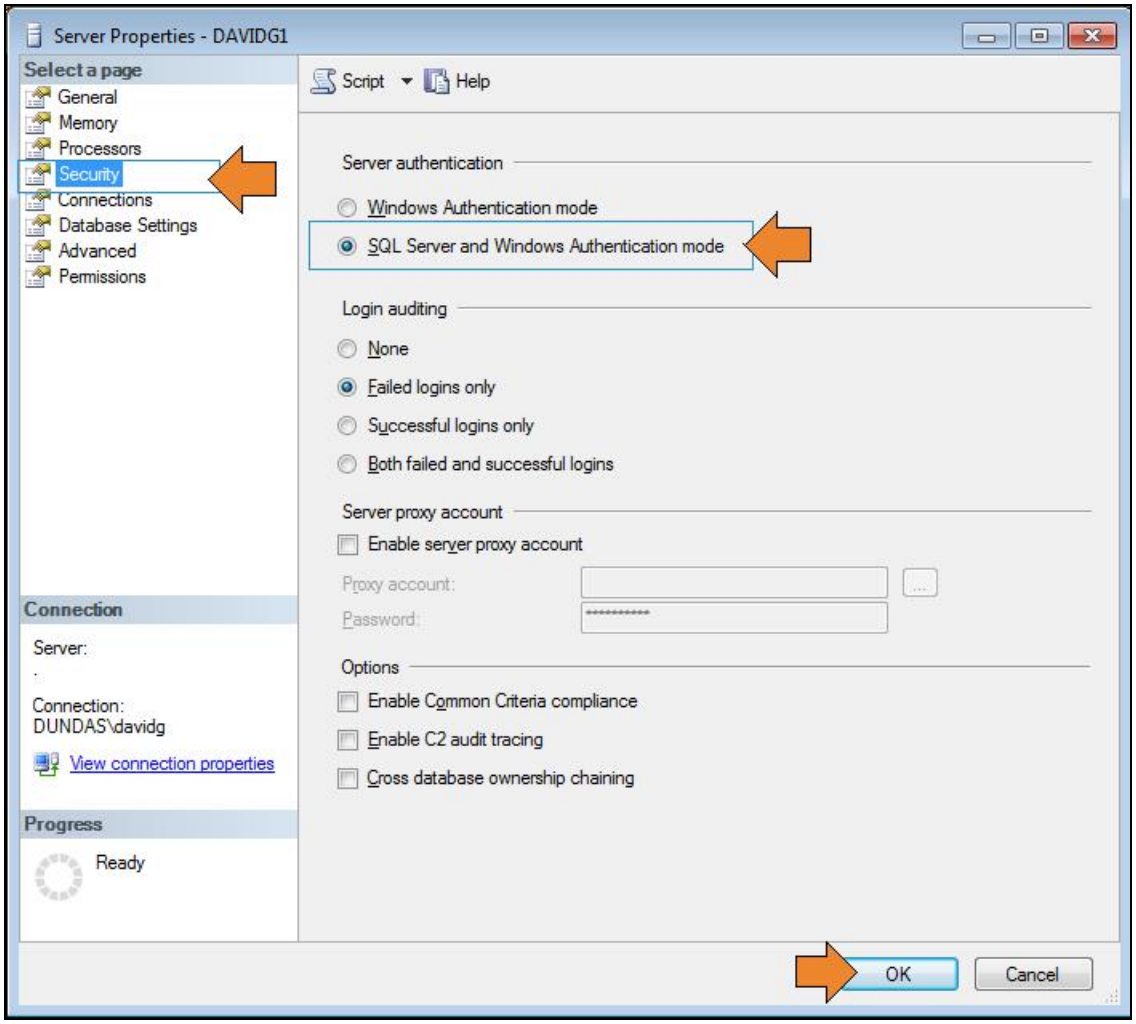
Enable SQL Server Authentication for your instance

1. Open SQL Server Management Studio.
2. Connect to the SQL Server instance you would like to use for Symphony.

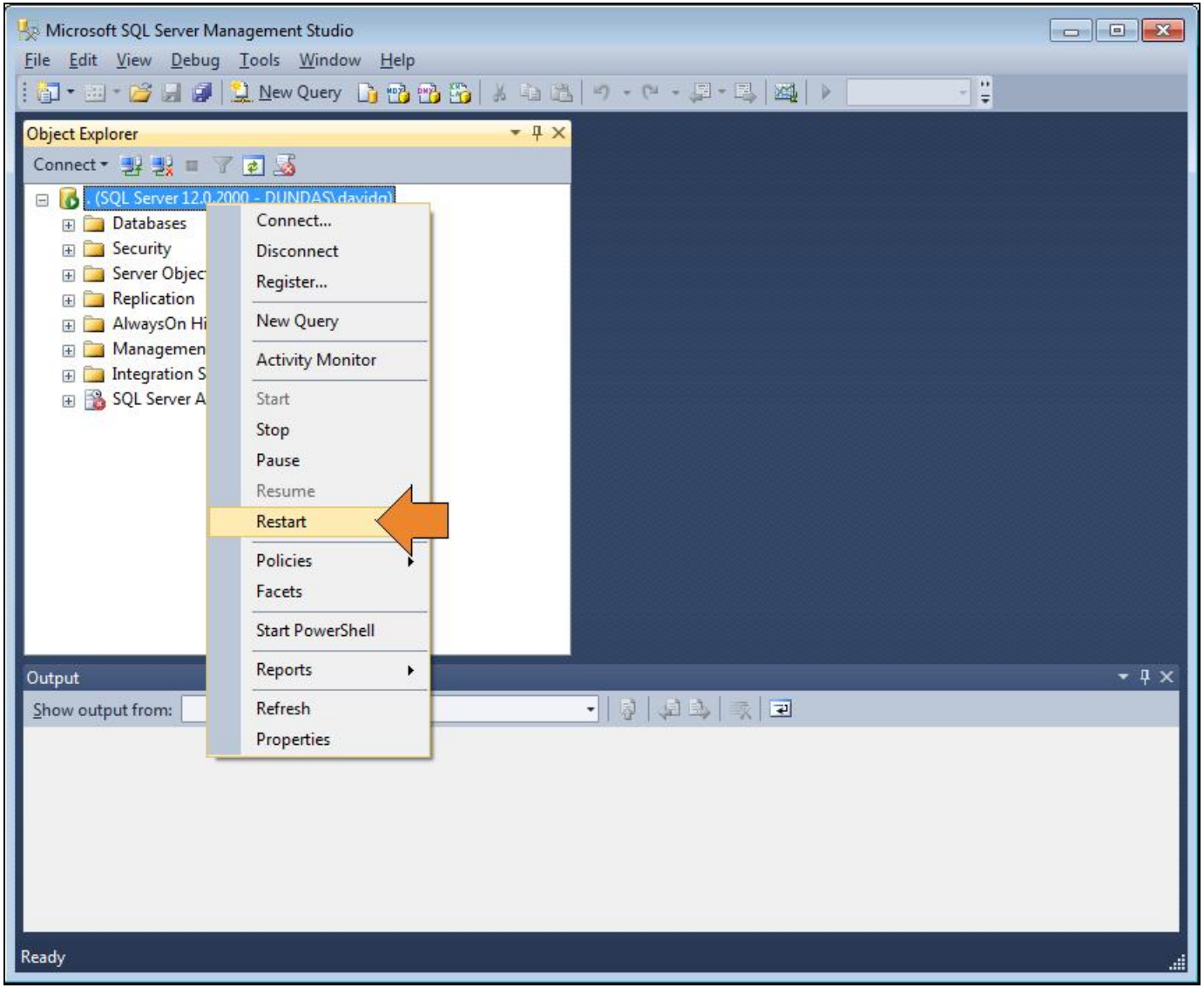
3. In the Object Explorer, right-click the server and click **Properties**.



4. On the **Security** page under **Server authentication**, select **SQL Server and Windows Authentication mode** and then click **OK**.



5. In the Object Explorer, right-click your server and click **Restart**. If the SQL Server Agent is running, it must also be restarted.



Using SQL Server Authentication in Symphony

Certain database permissions are needed for the user Symphony connects as while deploying a Symphony instance, which can be reduced when deploying is finished.

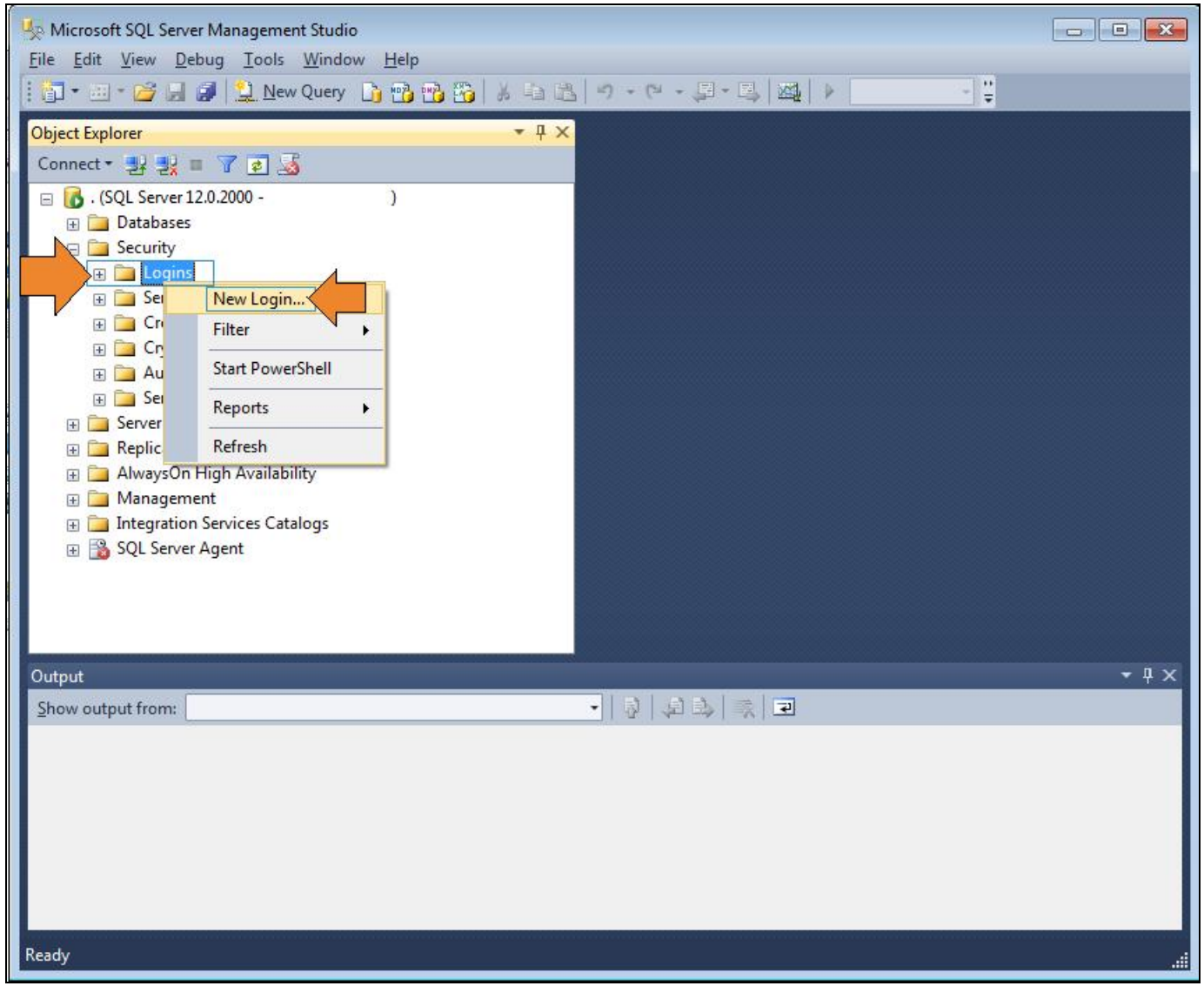
Before Deploying an Instance

Create a SQL Server authentication user

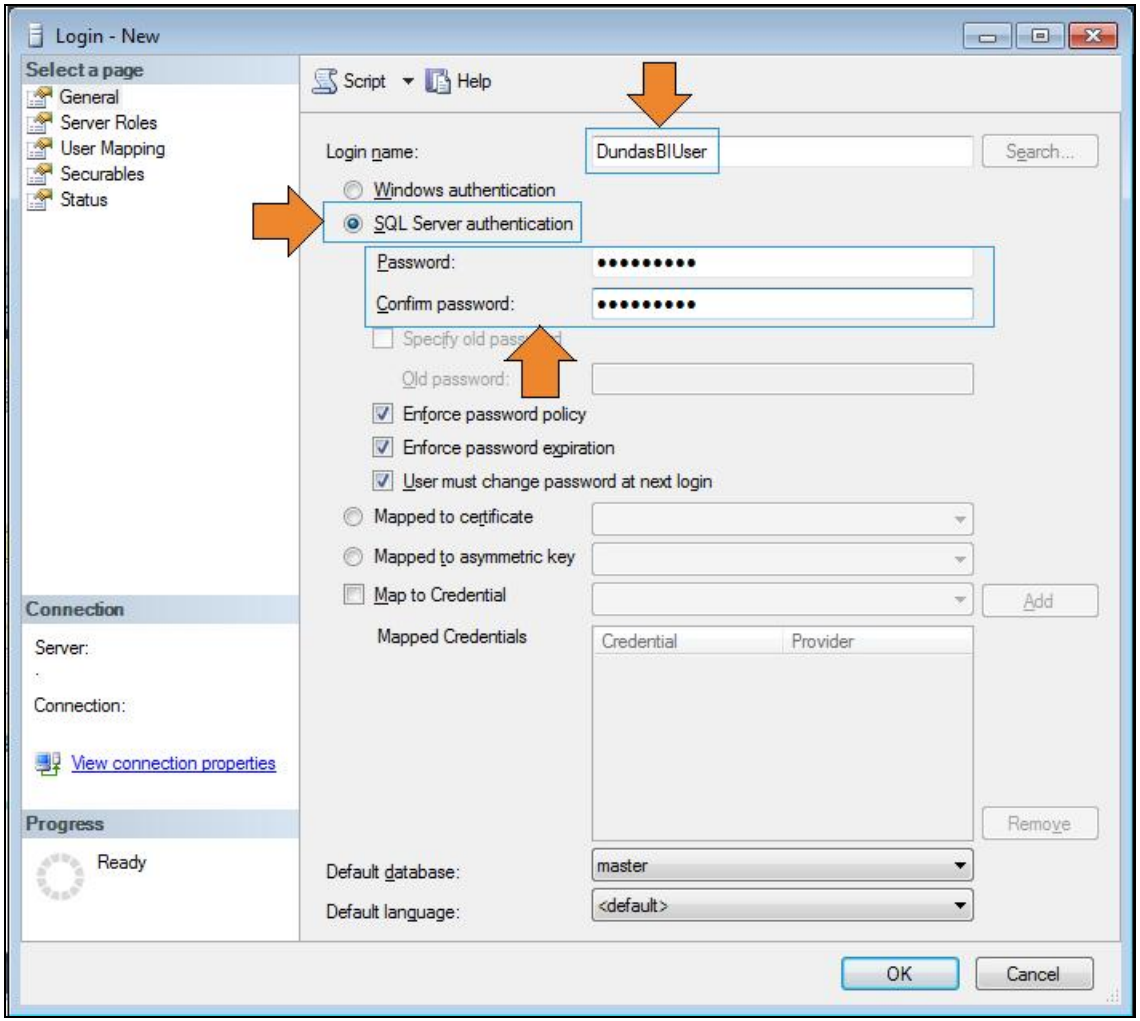
During the deployment of an instance with new databases, the user must have the **SysAdmin** role, or else all of the following: **DbCreator**, **DiskAdmin**, **ProcessAdmin**, and **SecurityAdmin**.

Create a user like this in SQL Management Studio:

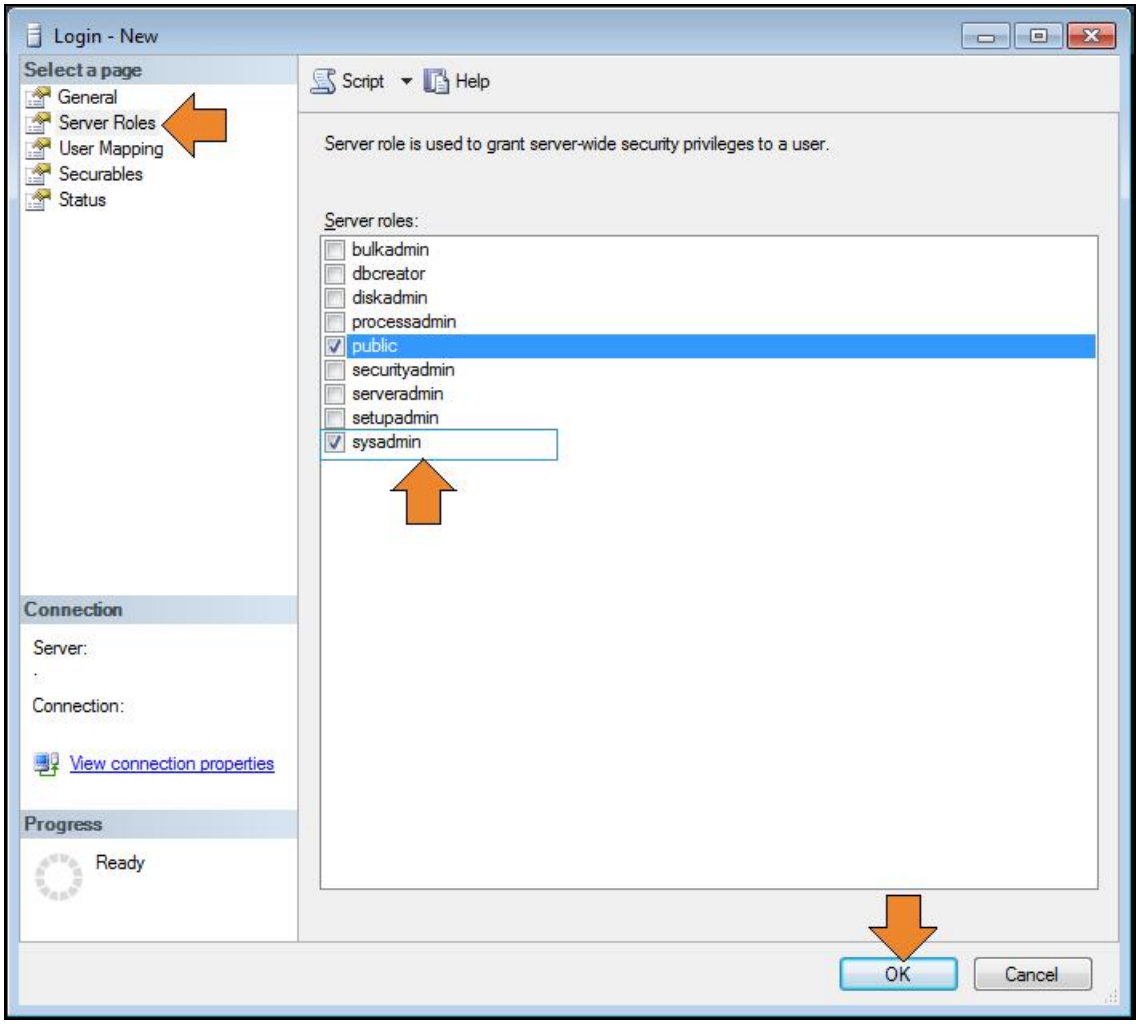
1. Open SQL Server Management Studio.
2. Connect to the SQL Server instance you would like to use for Symphony.


Microsoft SQL Server Management Studio
File Edit View Debug Tools Window Help
New Query
Object Explorer
Connect
. (SQL Server 12.0.2000 -)
Databases
Security
Logins
New Login...
Filter
Start PowerShell
Reports
Refresh
AlwaysOn High Availability
Management
Integration Services Catalogs
SQL Server Agent
Output
Show output from:
Ready

4. Enter the login name as **DundasBIUser**, select **SQL Server authentication**, and enter the Password.

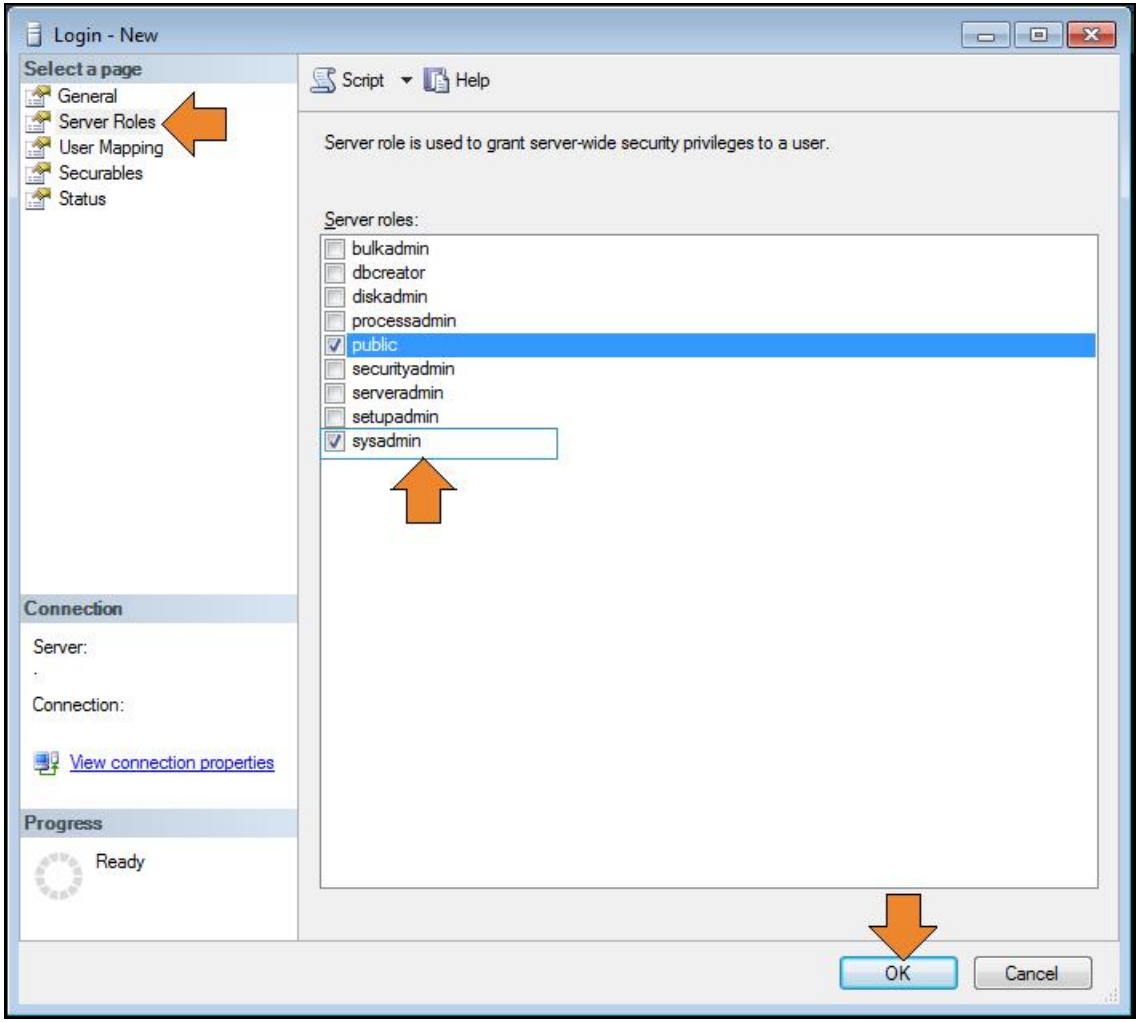


5. Click the **Server Roles** page and enable the **SysAdmin** role, or a combination of **DbCreator**, **DiskAdmin**, **ProcessAdmin**, and **SecurityAdmin** roles.



Specifying the SQL Server Authentication User

Now that the SQL Server authentication user has been created, use these credentials when you deploy Symphony and set up the application database connection.



After Deploying an Instance

After deploying, it is possible to remove the **SysAdmin** role from this user. For regular operation, the user will only require the **dbo** default schema and the **db_owner** role membership.

The **SysAdmin** role will be required again when attempting to upgrade an instance.

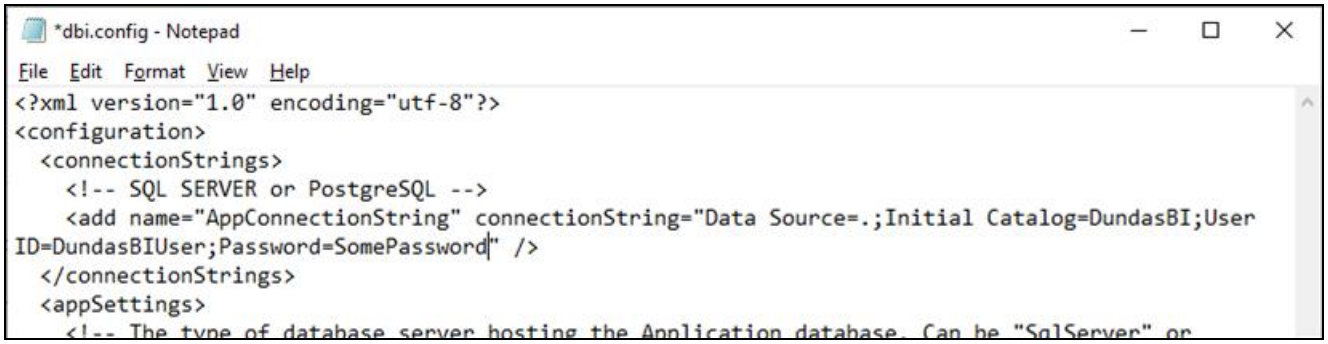
Change an existing Instance to Use SQL Server Authentication

Follow the steps below to change an existing instance to use SQL Server authentication for its application and warehouse databases.

Application Database

Open a text editor such as Notepad as an administrator, by right-clicking its shortcut and choosing **Run as administrator**. Open the configuration file.

Edit the connection string to use SQL Server authentication by specifying the User ID and Password. After saving, recycle the application pool in IIS or restart the Linux service for Symphony's website.



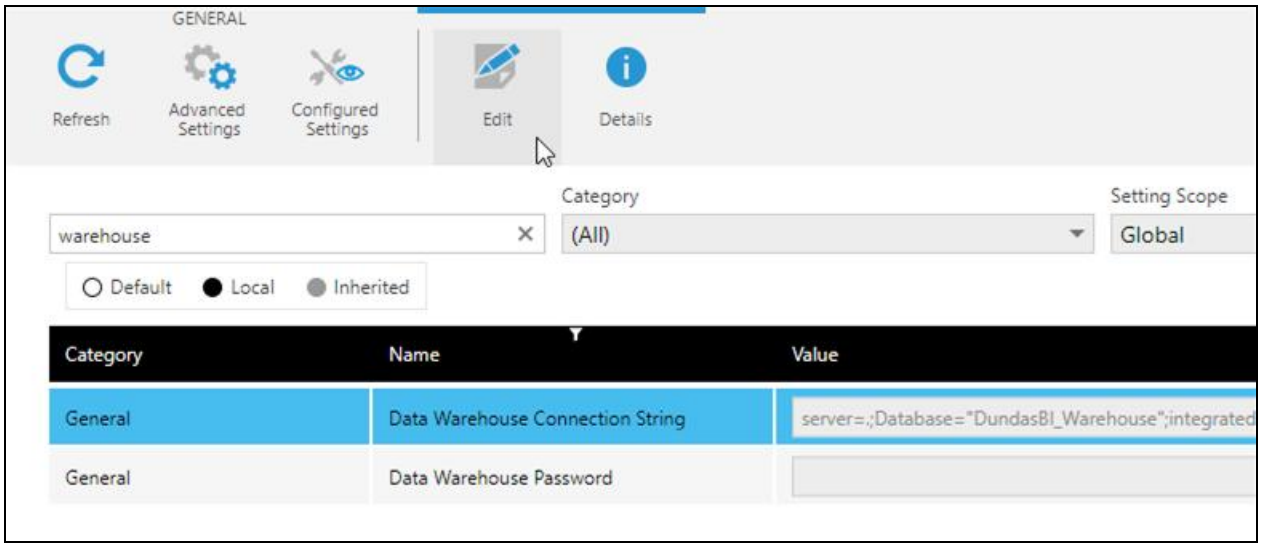
Note: To change the credentials when the connection string is encrypted, first decrypt the connection string using the dt command line tool, then update the credentials. To resume the secure state, encrypt the connection string when you have updated the credentials.

Warehouse Database

The warehouse database connection is defined in Symphony's configuration settings.

Log into Symphony as an administrator, and access Administration from the main menu.

Click to expand **Setup** and then click **Config**.



For more information, see:

- Microsoft Docs: SQL Server Management Studio
- Microsoft Docs: Choose an Authentication Mode
- Symphony Managed Dashboards and Reports Config File
- Configuration Settings

Getting Started With Logi AI

> **Important:** This feature is considered to be released in beta for your testing purposes. Workflows and features may change before a production-ready version is released.

Add Logi AI to your Symphony environment to create chatbots and chatflows to quickly add value for your internal and external users and customers by linking together the data and content you have quickly and easily. Offer up a mix of internal, external, and proprietary information using Logi AI within your Symphony environment, in your own embedded application, or anywhere you can embed a chatbot.

Logi AI can be used in a number of ways to present data you have outside of your Symphony environment, and combine it with information you're already bringing together with the help of Symphony's powerful analytic capabilities.

To support your AI needs, we've integrated Flowise natively to allow you to use this powerful tool easily in conjunction with Symphony. We've added a document loader, the Logi Symphony Component, that helps you bring together dashboards, visuals, and metric sets as inputs to output as a document to the Flowise architecture. Use the documents you generate in chatflows you can feed into a vector storage node for flexible access.

Embed your content anywhere. Build a chatflow to support the custom chatbot you design and embed on any web page or in any of your applications that support embedding. Include Symphony components where you need them, and leave them out where you don't.

Embed your content in dashboards. Give your users chatbots that you customize to give them specific and accurate data for their use cases. Add chatflows to help them generate fusion data, generate custom SQL to build complicated sources, or design a custom chatbot for your visuals to allow your customers to discuss the data the visual represents.

Bring in your existing Flowise templates to integrate Symphony into your existing chatflows, or jump in with one of our starter workflows from the marketplace, linked in the main menu of this module.

- **Logi Symphony QnA**: Run a live QnA using in-memory vector storage to query data per session or per question.
- **Logi Symphony Query Engine**: Use the LlamaIndex Query Engine to query your data directly after you've upserted it. Use on one data set at a time.
- **Logi Symphony Local**: Run a live QnA locally, similar to Logi Symphony QnA. This runs locally, allowing for air-gapped query and response.
- **Logi Symphony Prompt Chaining**: Build a flow of custom prompts to pull the appropriate data from Symphony, create context, and return more accurate answers.

Access Logi AI

Launch Logi AI by selecting the appropriate **Open** button on the Symphony home page. You can access any tasks you need to accomplish in this module here.

Use the main menu to access chatflows, market places, tools, assistants, or items you need to build your chatflows. Manage credentials, variables, API keys, and settings as needed.

You can access this menu at any time when you use Symphony. Select the menu icon () to expand and collapse the main menu. Select the Home button at any time to return to the home page and navigate to another section or module of Symphony.

As you select from the available options, each section may open new work area or series of nested menus you can use to navigate existing items or create new items.

The main pane of this work area, by default, allows you access your existing chatflows. When you select a menu option, more menu options may expand, and the pane changes to reflect the tasks you're performing.

Once you create a chatflow or chatbot and give users access to it, they can use the chatflow as a Symphony user. To prevent this, you can optionally guard the chatflow by using an API key.

Create and Use a Chatflow

Important: This feature is considered to be released in beta for your testing purposes. Workflows and features may change before a production-ready version is released.

Build chatflows to easily give your customers complex information brought together from a number of disparate sources, both inside and outside of your Symphony environment.

Bring nodes together from an number of available sources using the LangChain or LlamaIndex frameworks. We've also included our own Document Loader. Use this document loader to bring your customers data from all of Symphony: managed dashboards, managed reports, and visual data discovery.

Create a New Chatflow

1. Select **Chatflows** from the main menu of the Logi AI module. The chatflow work area opens, displaying a list of existing chatflows in column or tile format.
2. Select **Add New**. A work area opens you can use to build your new chatflow, named **Untitled chatflow**.
3. Select the plus symbol (**+**) to add nodes. An **Add Nodes** menu opens.
4. Navigate among available nodes from the offered frameworks to find a node to add to your canvas.
5. Grab a node, drag it to the canvas, and enter the required information to enable the node to works. Connect in your chatflow to complete your process.
6. Select the save icon in the upper right header of the work area to save a full or partial chatflow with a name you define.

   Once you've named the chatflow, you can rename it by selecting the edit icon next to the chatflow name you have assigned.

After you've saved your chatflow, you can:

- Test your chatflow by selecting the chatbot icon in the upper right corner of your chatflow canvas.

  Learn more about working with Flowise here: https://docs.flowiseai.com/getting-started

- Select the API Endpoint icon (**</>**) to open an embed dialog box. You can select one of several options for embedding a chatflow in a website or use it as an API. Learn more about working with embedded chatbots here: https://docs.flowiseai.com/using-flowise/embed

  Note: When you embed a chatbot, it will always override any dashboard ID defined in the chatflow with the dashboard ID of the dashboard in the chatbot is embedded in.

Important: If you upsert via API using the Logi Symphony document loader, you must supply overrides to ensure a session ID is known.

- Select the settings option to view messages related to your chatflow, to configure the chatflow, to duplicate, load, and export the chatflow, or to delete the chatflow.
- Add the chatbot to a dashboard in your Symphony environment.

Logi Symphony Document Loader

Important: This feature is considered to be released in beta for your testing purposes. Workflows and features may change before a production-ready version is released.

Use the Logi Symphony document loader to load data from Symphony into your chatflow. Depending on where your chatflow is used or how it is referenced by your Retrieval-Augmented Generation (RAG) model.

To learn more about document loaders, see: https://docs.flowiseai.com/integrations/langchain/document-loaders.

Inputs and Outputs

All inputs for the document loader are optional. These inputs include:

Input	Description
Connect Credential	Use to create and update credentials to connect to yourSymphony environment. If used, takes precedent over any session ID sent. Generally, credentials are for an API account. ■ Select **- Create New -** to add new connection credentials. ■ Select an existing saved credential to use in this chatflow. ■ Select the edit icon next to a selected credential to make changes to the stored information. ■ Select the x next to a selected credential to clear this field. If no credential is used, send a session ID instead. If you're using this document loader inside of a chatflow you've added to a dashboard in your Symphony environment, Symphony provides the user's session ID automatically.
Metric Set or Visual ID	Optionally, include the ID of a metric set from managed dashboards, or a visual from Visual data discovery for your chatflow to work with.
Dashboard ID	Optionally, include the ID of a dashboard to work with.
Row Limit	Optionally, define the maximum number of rows returned from Symphony

Input	Description
	when called.

Connect the document loader output to your in-memory vector store, and continue building your chatflow as needed.

Connect Credential

Chatflows using this document loader may not need defined credentials to work with the data. Send a session ID from a user when making a call instead of defining credentials.

If you're using this document loader inside of a chatflow you've added to a dashboard in your Symphony environment, Symphony provides the user's session ID automatically.

If you choose to use credentials for an API account, you can ensure the same data is accessed every time, and results are not changed or restricted by the limitations of the user sending a session ID.

When you create a new connect credential, the following required and optional fields include:

Field	Description
Credential Name	A name for these credentials. This name does not have to be unique. Required.
Logi Symphony User	The user name of an account to connect to Symphony data. This generally should be credentials associated with an API account. Required.
Logi Symphony Password	The password for the user name connecting to Symphony. Required.
Delete Other Sessions	Disabled by default. Enable to forcibly log off any other sessions that may be using this document loader. Generally, this is only a concern when using a reserved seat.

Add Chatbots to Dashboards

Important: This feature is considered to be released in beta for your testing purposes. Workflows and features may change before a production-ready version is released.

Logi AI is a comprehensive artificial intelligence (AI) and natural language (NLG) module designed to transform the way you analyze your data. Embedded chatbots help you elevate the user experience Symphony, making complex data analytics intuitive and effective for informed data-driven decisions. To learn more about embedding chatbots into any of your own applications outside of Symphony, see Use and Embed Chatbots In Your Application

After you've created your chatflow, you can add it to any dashboard in available modules, Visual data discovery or Managed Dashboards and Reports as a chatbot.

Important: If you filter data in a dashboard or metric set after it's been loaded by the Logi Symphony document loader, it may not be reflected in the results returned by the AI. See Logi AI Limitations.

Add Chatbots to Your Dashboards

Data Discovery Dashboards: Add or Remove a Chatbot

1. Open a data discovery dashboard you can edit to add a chatbot.
2. If one or more chatflows are available to include in this dashboard, a **Chat Settings** icon appears in the upper left corner of the dashboard. This icon is green if a chatflow is used in this dashboard, and grey if a chatflow is available but not in use.
3. Select the **Chat Settings** icon. The Chat Settings sidebar menu opens.
4. Select the name of a **Chat Flow** from the list of available options in the dropdown list.

   Select **None** to remove the chatbot.
5. A chat bubble is immediately added to the bottom of your dashboard if you've added a chatbot, or removed if you selected **None**.
6. Save the dashboard to include your chatbot change to the dashboard.

Managed Dashboards: Add or Remove a Chatbot

1. Edit and check out a managed dashboard to add or remove a chatbot.
2. Navigate to the **Properties** panel and select a **Chatflow** in the **Features** section of the panel.

   If a chatbot is already included in this dashboard, select the delete icon to remove the chatbot.
3. A chat bubble, visible in view mode, is immediately added to the bottom of your dashboard if you've added a chatbot, or removed if you deleted the selected chatbot.

   It is not visible for dashboards embedded inside of dashboards.
4. Check in the dashboard to include your chatbot change to the dashboard.

Embedding Dashboards That Include Chatbots

If you embed a Symphony dashboard into your own application or another dashboard, any chatbot you've included in the dashboard is not shown. To support a chatbot for an embedded dashboard, embed the chatbot using embed code instead. See Use and Embed Chatbots In Your Application.

Generate SQL Statements

Important: This feature is considered to be released in beta for your testing purposes. Workflows and features may change before a production-ready version is released.

This applies to: *Visual Data Discovery*

Use chatflows and Logi AI to generate your SQL statements in Visual Data Discovery when you create or update an entity in your data source.

You have several options when creating your statements:

- Create your own chatflows to generate statements
- Get started quickly by downloading and implement the Logi AI SQL chatflow template

Use the chatflows and the results generated as-is, or customize further to suit your needs.

Connect Your Chatflow

After you have downloaded or set up your chatflow in Logi AI, register the chatflow in Visual Data Discovery.

1. As an administrative user, log in and navigate to the Visual Data Discovery module, then select **Tools** > **Advanced** from the main menu. A work area opens that you can use to manage instance level variables.

   Caution: Changing toggles or editing content other than as instructed in this work area may prevent your users from using various components of Symphony.

2. Locate the **Key** `symphony-ai-sql-flow`, and enter the ID of your chatflow as the corresponding **Value**. This ID is shown as part of the URL when you have the chatflow open for editing in the Logi AI module.
3. Select Save to save your changes to your environment.
4. The next time you open a work area that allows you to add an SQL statement, Symphony displays a new option: **Generate SQL with AI**.

Use Your Chatflow to Generate SQL Statements and Queries

Important: Custom SQL queries are a powerful tool for performing complex data queries. However, be careful when creating custom SQL queries because it is easy to define a heavy query or a query that may overwhelm your database. Use this feature carefully.

1. Navigate to and select an Entity in the source creation work area, then select the **Custom SQL** option in the Entity Details work area.
2. Select the option below the Custom SQL editing pane, **Generate SQL with AI**. The **AI SQL Generator** dialog opens.
3. Enter a prompt in the text field, then select the send icon to generate a prompt from your registered chatflow based on the data available and the schema information for your entity. Symphony returns your generated statement.
4. Review your statement result. Select **Use this SQL** to copy this statement to your Custom SQL editor.

   Note: You can edit your prompt and regenerate a statement, or select the send icon to generate a different response from the same prompt.
5. Run your statement to check it, then make further edits or save your changes.

Use and Embed Chatbots In Your Application

Important: This feature is considered to be released in beta for your testing purposes. Workflows and features may change before a production-ready version is released.

Logi AI is a comprehensive artificial intelligence (AI) and natural language (NLG) module designed to transform the way you analyze your data. Embedded chatbots help you elevate the user experience in any application, making complex data analytics intuitive and effective for informed data-driven decisions. To learn more about adding chatbots to dashboards, see Add Chatbots to Dashboards.

After you've created your chatflow, you can add it to any web page using HTML. Logi AI can generate code for you, or you can build your own from the example provided.

Alternatively, select the API Endpoint icon (**</>**) to open an embed dialog box while you have the chatflow open. You can select one of several options for embedding a chatflow in a website or use it as an API.

Important: If you upsert via API using the Logi Symphony document loader, you must supply overrides to ensure a session ID is known.

Embed Chatbots Using Generated Code

Logi AI provides an option to generate simple and customizable code to add your chatbot to any applicable html file. If you prefer to build your own code, more examples are provided here.

Generate code to embed a chatbot in a web page

1. Open the chatflow you want to embed in Logi AI. An **Embed in website or use as API** work area opens.
2. Select the embed options you'd like to use in the work area.
3. Copy the generated code using the copy icon, then paste inside the `<body>` tag of your html file.
4. If you prefer to configure the look at feel of your chatbot, select **Show Embed Chat Config** instead. Use the copy icon to copy the generated code. Make any layout changes you prefer, then paste inside the `<body>` tag of your html file.

Embed Using HTML

You can build your own html embed code following guidelines provided by Flowise or provided in our fork.

- Flowise documentation: https://docs.flowiseai.com/using-flowise/embed
- Flowise repository: https://github.com/FlowiseAI/FlowiseChatEmbed#configuration
- Our fork: https://github.com/LogiComposer/FlowiseChatEmbed#configuration

A simple HTML example with no values is provided below.

```
<script type="module">
	import Chatbot from "https://{logi-symphony-url}/aichatbot/web.js"
	Chatbot.init({
		chatflowid: "{chatflow-id}",
		apiHost: "https://{logi-symphony-url}/aichatbot",
	})
</script>
```

If API keys are enabled for authorization, your chatbot can't be embedded, but used as an API endpoint only.

If you're using the Logi Symphony document loader with no values, provide them at embed time. The example below includes both a Session ID and Dashboard ID.

```
<script type="module">
	import Chatbot from "https://{logi-symphony-url}/aichatbot/web.js"
	Chatbot.init({
		chatflowid: "{chatflow-id}",
		apiHost: "https://{logi-symphony-url}/aichatbot",
		chatflowConfig: {
			logiSymphonyDashboardId: "{dashboard-id}",
			logiSymphonySessionId: "{session-id}"
		}
	})
</script>
```

Configuration Options for Your Chatflow

All available properties you can set for `chatflowConfig` are optional.

Setting	Description
`logiSymphonyMetricSetVisualId`	The ID of the metric set from managed dashboards or visual from Data Discovery to work with.

Setting	Description
`logiSymphonyDashboardId`	The ID of the dashboard to work with from managed dashboards or data discovery.
`logiSymphonyRowLimit`	The maximum number of rows you want returned by the call for data from Symphony.
`logiSymphonySessionId`	The ID of the session to use to access Symphony. If a session ID is not set, credentials must be set in the chatflow to use. **Note:** If credentials are sent, they take precedent over any session ID sent.

To obtain the `logiSymphonySessionId`, see Embed Options.

```
<script src="https://{logi-symphony-url}/managed/Scripts/Embed/dundas.embedded.min.js" type="text/javascript"></script>
<script>
	// Relative or absolute URL to Dundas BI:
	const promise = window.setupDundasEmbedded('https://{logi-symphony-url}/managed');
	promise.then(() => {

		// FOR TESTING ONLY!
		// This should be something like:
		// let logonToken = myApp.getLogonTokenFromServer();
		const getTokenPromise = dundas.context.getService("LogOnService").getLogOnToken("user", "password");
		getTokenPromise.done((logOnTokenResult) => {
			if (logOnTokenResult.logOnToken) {

				// From here the code is fine to use as-is after the logontoken is retrieved from the parent application.
				dundas.context.getService("LogOnService").logOnByToken(logOnTokenResult.logOnToken).done((result) => {
					if (result.sessionId) {
						import("https://{logi-symphony-url}/aichatbot/web.js").then(() => {
							window.Chatbot.init({
								chatflowid: "{chatflow-id}",
								apiHost: "https://{logi-symphony-url}/aichatbot",
								chatflowConfig: {
									logiSymphonyMetricSetVisualId: "{metric-set-or-visual-id}",
									logiSymphonySessionId: result.sessionId
								}
							});
						});
					}
				});
			}
		});
```

```
        });
</script>
```

Logi AI Limitations

Important: This feature is considered to be released in beta for your testing purposes. Workflows and features may change before a production-ready version is released.

Logi AI is a comprehensive and agnostic generative artificial intelligence (AI) module designed to transform the way you analyze your data. As a flexible AI platform, you can create your own AI-powered chatbots using any LLM or SLM you have, and embed them directly into dashboards and visualizations or even any webpage you might have. Chatflows can securely access data from Logi Symphony and use it for Retrieval-Augmented Generation (RAG).

Some important information and limitations to this release to keep in mind as we develop and expand Logi AI to meet your data analysis needs.

- Back up your chatflows manually before you upgrade or apply incremental updates in your environment of any related or linked components, including Symphony and Flowise.
- If you change and further filter data in a dashboard or metric set after it's been loaded by the Logi Symphony document loader and in-memory vector stores, it may not be reflected in the results returned by the AI. Changes may be included if values are changed in metric set or a visual and your chat reloaded to pick up the updated data. Alternatively, use upsert ahead of time, using a larger data set in a persistent vector storage. One work around to this would be to use a dynamic `sessionId` that changes at every `init` or by recreating the chatbot (erasing the chat history).
- There is only one user in Logi AI: all system administrators use the same Logi AI user to create, edit, and delete chatflows.
- When you deploy Logi AI with our helm chart, an AI database is required, as well as a persistent volume claim to store API keys and logs. A persistent vector storage may also be used for larger data sets.
- When you use the upsert API with the Logi Symphony document loader you are required to specify a session ID. Optionally, specify a dashboard ID.
- If used, connection credentials take precedent over any session ID sent in the document loader. Optionally, embed a chatflow without specifying a dashboard ID.
- Access to the user interface of this module is guarded. Only system administrators (not tenant administrators) can access the UI. Many of the APIs are whitelisted and do not require a session ID to call to data for embedding. You can still guard these calls via CORS and other mechanisms.
- Logi AI checks for configuration changes every 60 minutes. You can forcibly restart the pod to update it immediately, reading in any changes to CORS/CSP headers.
- Some visuals may not reflect the fully expressed data correctly at this time.

- If you select a node that uses LlamaIndex, run upsert with SimpleStore before using the chatflow.
- Subpath deployment is not supported. You must deploy your application at the index of your site, where Logi AI is in `/aichatbot`.

Embed Options

This applies to: *Managed Dashboards, Managed Reports*

You can use a number of available methods to embed views such as dashboards and reports, individual visualizations, or a simplified, self-service dashboard creation experience, directly in your own web application without using iFrames. These are powered by the JavaScript API, which you can also access with this embedding method.

Alternatively, embed the entire application or any page from it using other embedding methods. Take advantage of white labeling, API, and extension support, depending on your scenario and preferences.

Set Up IFrameless Embedding

The first step to embedding Managed Dashboard content directly on your page is to add `dundas.embedded.min.js`. This file is included in your instance installation at the following location on disk:

```
[Instance Root]\www\BIWebsite\wwwroot\Scripts\Embed\dundas.embedded.min.js
```

This can be accessed under a Symphony instance URL as `/Scripts/Embed/dundas.embedded.min.js`. For example, if Symphony is available as `https://dbi.example.com/` , your page might add:

```
<script src="https://dbi.example.com/Scripts/Embed/dundas.embedded.min.js" type="text/javascript"></script>
```

Note: This is different from the use of `dundas.embed.min.js`, the embed library used to embed Symphony via an iFrame on your page.

There is a single method to call, which sets up Symphony on your page by adding and loading all CSS and JavaScript dependencies:

```
// Relative or absolute URL to your instance:
let promise = window.setupDundasEmbedded('https://dbi.example.com/');
```

The following table describes available options for these properties and objects.

Property/Object	Default	Description
`dundasBIUrl`	none	The relative or absolute URL to access the root of the Symphony application from the current page.

Property/Object	Default	Description
		Type: string
`doNotAutoCreateBootstrap`	none	Optional. By default, a new `dundas.views.embedded.Bootstrap` is created immediately. If the document is not yet ready, pass `true` for this parameter and call this bootstrap's constructor at a later point. Type: boolean

This returns a Promise object that resolves once the Dundas JavaScript API bootstrap is ready for use. This is not a jQuery.Promise: third party libraries are not yet loaded when this method is called.

Important: If the Symphony URL has a different domain or origin from your page, cross-origin resource sharing (CORS) must be set up. Browsers such as Chrome also require both sites use the HTTPS protocol for cross-origin sharing.

Use the JavaScript API

When the `setupDundasEmbedded` method's Promise resolves, you can access the JavaScript API via `dundas.context` the same way as on a page within the Symphony application, and access its services.

Most actions require authentication, so first use the `LogOnService` to log on using a one-time token obtained from the Symphony server like in the following example, or another available `LogOnService` method:

```
let logonToken = myApp.getLogonTokenFromServer();
let promise = window.setupDundasEmbedded('https://dbi.example.com/');
promise.then(() => {
	dundas.context.getService("LogOnService").logOnByToken(logonToken).done((result) => {
		if (result.sessionId) {
			// Logon successful!
		}
	});
});
```

The result of a log on method has the properties defined by `LogOnResultData` (beginning in lowercase in JavaScript). If a session ID is not populated in the result, the log on was likely not successful and the other properties will contain details about the reason.

After a successful log on, set the session ID so that you can embed content or access other services. You can call `setSessionId` on `dundas.context`, or pass the session ID into one of the embedding methods as a shortcut.

Embed Content

When Symphony is embedded directly onto your page, `dundas.context` is a `dundas.views.embedded.Bootstrap`.

Call its `embedView` method to embed an existing dashboard, report, or other view type by its ID, which you can find from its URL or properties dialog. It is rendered inside the DOM element you specify, which should have some set size independent of its contents (i.e., not auto). The following embeds a dashboard:

```
let element = document.getElementById("viewContainer");
dundas.context.embedView(element, "bdba4200-8d2d-0f90-0049-cfbe81e00004", {
       sessionId: sessionId
});
```

The third argument is an object specifying additional optional values. Important options include passing `sessionId` as an alternative to first calling and waiting for `setSessionId`, and `viewType`when embedding a type of view other than a dashboard like in the next example. A `jQuery.Promise` is returned that you can use to wait for the embedding to complete.

Content is always embedded on its own without other interface elements like the application's toolbar or main menu, but with most functionality accessible by default as usual from the context menu via right-click or long-tap.

Alternative Embed Methods

The entire application or any desired page from it can also be embedded via an iFrame, or a WebView or similar component in mobile or desktop apps, or by simply linking to a Symphony URL.

- In a web page or web application, the Symphony-provided embed library can make this process easier, including helper methods facilitating any needed communication between your application's JavaScript and the Symphony embedded application.

- Alternatively, create an iFrame element or WebView component yourself if desired, and load a URL including optional parameter values. You can still communicate with an embedded Symphony application using script as shown in the article Using cross-origin resource sharing.

You may also need to communicate with a Symphony server from your application instead of or in addition to embedding it on the page. You can do this within a web application by embedding the JavaScript API directly onto your page as shown earlier, or you can also use the REST API directly from all types of applications. The .NET API is available to use from .NET-based applications.

You can make the Symphony interface match your own branding using white labeling support including custom CSS, JavaScript, HTML, and interface text, with support also for separate branding per tenant. You can plug in any custom functionality you might need via extensions anywhere from data providers, transforms, formulas, and visualizations through to export and notification delivery providers.

Besides logging your users in with one-time tokens, other single sign-on options are available including federated authentication with another identity provider you may already be using such as Azure Active Directory or Okta, or another provider using standard protocols such as SAML 2.0 and OpenID Connect (OIDC).

Technical Notes

The following applies when embedding Symphony on your page without an iFrame as described above:

- Only one ad hoc dashboard editor can be embedded on a page, but the embedView and embedMetricSet methods can be called multiple times per page.
- Only one user session is supported at a time within a single open page in a browser.
- As this is an experimental API, there may be breaking changes in a future version if needed based on feedback and for added enhancements.

For more information, see the following topics:

- **Embed Viewer**: Managed Dashboards And Reports Embed Viewer
- **Embed Metric Set**: Embed Metric Sets
- **Embed Ad Hoc Editor**: Embed Ad-Hoc

Embed Metric Sets

This applies to: *Managed Dashboards, Managed Reports*

When Symphony is embedded directly onto your page, `dundas.context` is a `dundas.views.embedded.Bootstrap`.

Call its `embedView` method to embed an existing dashboard, report, or other view type by its ID, which you can find from its URL or properties dialog. It is rendered inside the DOM element you specify, which should have some set size independent of its contents (i.e., not auto). The following embeds a dashboard:

```
let element = document.getElementById("viewContainer");
dundas.context.embedView(element, "bdba4200-8d2d-0f90-0049-cfbe81e00004", {
	sessionId: sessionId
});
```

The third argument is an object specifying additional optional values. Important options include passing `sessionId` as an alternative to first calling and waiting for `setSessionId`, and `viewType`when embedding a type of view other than a dashboard like in the next example. A `jQuery.Promise` is returned that you can use to wait for the embedding to complete.

Content is always embedded on its own without other interface elements like the application's toolbar or main menu, but with most functionality accessible by default as usual from the context menu via right-click or long-tap.

If you are embedding multiple items on your page, a separate `EmbeddedViewService` is used for displaying each one. Specify an `embeddedViewServiceName`of your choice starting at least with the second embedding call (omitting this will cause an error such as "Service EmbeddedViewService is already registered"). Either to wait for the first embedding method to finish, or to call and wait for `setSessionId` first before embedding any content as follows:

```
let element1 = document.getElementById("container1");
let element2 = document.getElementById("container2");
dundas.context.setSessionId(result.sessionId).done(() => {
	dundas.context.embedView(element1, "8166d221-af1a-1881-40c6-f09ed09fe735");
	dundas.context.embedView(element2, "06294698-2d5a-9b4a-7b46-195db1d30397", {
		embeddedViewServiceName: "EmbeddedReportService",
		viewType: dundas.entities.ViewType.REPORT
	});
});
```

Note: You can use the name you specified for an embedded view service or else its class name `EmbeddedViewService` when calling `getService` to get its instance. For example, you can call its dispose method if you want to remove the content from your page.

Parameter Values

When calling `embedView`, you can use the `viewOverrides` option to pass in as well as create view parameter values. The same parameter value types are used as shown in Modify a Filter / View Parameter Using Scripting.

The following example sets an existing numeric parameter with the script name `viewParameter1` when embedding the dashboard:

```
var numberValue = new dundas.data.RangeNumberValue({
	lowerBoundaryValue: 1200,
	upperBoundaryValue: 1800
});
var overrides = new dundas.view.controls.ViewOverrides();
overrides.defaultViewParameterValues.push({
	viewParameterName: "viewParameter1",
	parameterValue: numberValue
});

dundas.context.embedView(document.getElementById("viewContainer"), "<insert ID>", {
	sessionId: sessionId,
	viewOverrides: overrides
});
```

Similarly, you can create a view parameter that did not already exist, like the following that sets some specific hierarchy members by their unique names:

```
var newViewParameter = new dundas.view.ViewParameter();
newViewParameter.addElementParameterLink(new dundas.view.ElementParameterLink({
	adapterId: "9ae85f75-0213-f2b6-98e8-e2000042b2cf",
	elementUsageUniqueName: "Customer Count",
	metricSetBindingId: "ba37ecdc-a049-9fca-00b1-1ef9f911147f",
	parameterId: "58e027c3-003d-4cc8-a5b9-1cf83a36deaf"
}));
newViewParameter.parameterValue = new dundas.data.CollectionMemberValue({
	values: [
		new dundas.data.MemberValue({
			"hierarchyUniqueName": "Product",
			"uniqueName": "Accessories.C.Product",
			"levelUniqueName": "C.Product",
		}),
		new dundas.data.MemberValue({
			"hierarchyUniqueName": "Product",
			"uniqueName": "Clothing.C.Product",
			"levelUniqueName": "C.Product",
		})
```

```
        ]
});

var overrides = new dundas.view.controls.ViewOverrides();
overrides.viewParameters.push(newViewParameter);

dundas.context.embedView(document.getElementById("viewContainer"), "<insert ID>", {
        embeddedViewServiceName: "DashboardEmbeddedViewService",
        viewOverrides: overrides
});
```

Note: Find a visualization's `adapterId` in the Properties window, and the `metricSetBindingId` in the metric set settings dialog accessible from the Data Analysis Panel. The parameter ID is defined on each `AnalysisElementUsage` in a metric set's measures or hierarchies as `analysisElementUsage.parameter.id`.

Embed Metric Sets

Calling embedMetricSet is similar to calling embedView but for an existing metric set and its default visualization. The same additional arguments are needed when embedding multiple Symphony items on the same page.

The following embeds a single metric set:

```
let container = document.getElementById("visualizationContainer");
dundas.context.embedMetricSet(container, "7b8000df-36fd-4994-b6f6-2f6d96999fab", {
        sessionId: sessionId
});
```

Note: To access embedded metric sets directly rather than through an embedded view such as an existing dashboard, a user must have a power user seat license or higher.

Managed Dashboards and Reports Embed Viewer

This applies to: *Managed Dashboards, Managed Reports*

When Symphony is embedded directly onto your page, `dundas.context` is a `dundas.views.embedded.Bootstrap`.

Call its `embedView` method to embed an existing `viewType` by its ID, which you can find from its URL or properties dialog. It is rendered inside the DOM element you specify, which should have some set size independent of its contents (i.e., not auto). The following embeds a dashboard:

```
let element = document.getElementById("viewContainer");
dundas.context.embedView(element, "bdba4200-8d2d-0f90-0049-cfbe81e00004", {
	sessionId: sessionId
});
```

The third argument is an object specifying additional optional values. Important options include passing `sessionId` as an alternative to first calling and waiting for `setSessionId`, and `viewType`when embedding a type of view other than a dashboard like in the next example. A `jQuery.Promise` is returned that you can use to wait for the embedding to complete.

Content is always embedded on its own without other interface elements like the application's toolbar or main menu, but with most functionality accessible by default as usual from the context menu via right-click or long-tap.

If you are embedding multiple items on your page, a separate `EmbeddedViewService` is used for displaying each one. Specify an `embeddedViewServiceName`of your choice starting at least with the second embedding call (omitting this will cause an error such as "Service EmbeddedViewService is already registered"). Either to wait for the first embedding method to finish, or to call and wait for `setSessionId` first before embedding any content as follows:

```
let element1 = document.getElementById("container1");
let element2 = document.getElementById("container2");
dundas.context.setSessionId(result.sessionId).done(() => {
	dundas.context.embedView(element1, "8166d221-af1a-1881-40c6-f09ed09fe735");
	dundas.context.embedView(element2, "06294698-2d5a-9b4a-7b46-195db1d30397", {
		embeddedViewServiceName: "EmbeddedReportService",
		viewType: dundas.entities.ViewType.REPORT
	});
});
```

Note: You can use the name you specified for an embedded view service or else its class name `EmbeddedViewService` when calling `getService` to get its instance. For example, you can call its dispose method if you want to remove the content from your page.

Parameter Values

When calling `embedView`, you can use the `viewOverrides` option to pass in as well as create view parameter values. The same parameter value types are used as shown in Modify a Filter / View Parameter Using Scripting.

The following example sets an existing numeric parameter with the script name `viewParameter1` when embedding the dashboard:

```
var numberValue = new dundas.data.RangeNumberValue({
	lowerBoundaryValue: 1200,
	upperBoundaryValue: 1800
});
var overrides = new dundas.view.controls.ViewOverrides();
overrides.defaultViewParameterValues.push({
	viewParameterName: "viewParameter1",
	parameterValue: numberValue
});

dundas.context.embedView(document.getElementById("viewContainer"), "<insert ID>", {
	sessionId: sessionId,
	viewOverrides: overrides
});
```

Similarly, you can create a view parameter that did not already exist, like the following that sets some specific hierarchy members by their unique names:

```
var newViewParameter = new dundas.view.ViewParameter();
newViewParameter.addElementParameterLink(new dundas.view.ElementParameterLink({
	adapterId: "9ae85f75-0213-f2b6-98e8-e2000042b2cf",
	elementUsageUniqueName: "Customer Count",
	metricSetBindingId: "ba37ecdc-a049-9fca-00b1-1ef9f911147f",
	parameterId: "58e027c3-003d-4cc8-a5b9-1cf83a36deaf"
}));
newViewParameter.parameterValue = new dundas.data.CollectionMemberValue({
	values: [
		new dundas.data.MemberValue({
			"hierarchyUniqueName": "Product",
			"uniqueName": "Accessories.C.Product",
			"levelUniqueName": "C.Product",
		}),
		new dundas.data.MemberValue({
			"hierarchyUniqueName": "Product",
			"uniqueName": "Clothing.C.Product",
			"levelUniqueName": "C.Product",
		})
```

```
        ]
});

var overrides = new dundas.view.controls.ViewOverrides();
overrides.viewParameters.push(newViewParameter);

dundas.context.embedView(document.getElementById("viewContainer"), "<insert ID>", {
        embeddedViewServiceName: "DashboardEmbeddedViewService",
        viewOverrides: overrides
});
```

Note: Find a visualization's `adapterId` in the Properties window, and the `metricSetBindingId` in the metric set settings dialog accessible from the Data Analysis Panel. The parameter ID is defined on each `AnalysisElementUsage` in a metric set's measures or hierarchies as `analysisElementUsage.parameter.id`.

Apply Themes

You can use other services from the JavaScript API to use the ThemeService to get a theme by its file ID, then call the method on the current view (e.g., dashboard or report) to apply it. See the following script example.

```
dundas.context.getService("ThemeService").getThemeById("89f89a14-000f-435b-ae7b-78265f2cdb7e").done((theme) => {
        dundas.context.baseViewService.currentView.applyTheme(theme);
});
```

View Type

This applies to: *Managed Dashboards, Managed Reports*

Create a `ViewType` object from a given `Object Type` using the `fromObject` Type method.

Properties

Name	Description
DASHBOARD	
REPORT	
SCORECARD	
SMALL_MULTIPLE	

Constants

Name	Description
DASHBOARD	
REPORT	
SCORECARD	
SMALL_MULTIPLE	

Embed Ad-Hoc

This applies to: *Managed Dashboards, Managed Reports*

Embed a simplified, self-serve dashboard creation experience directly onto your own application's page by calling the `createAdhocDashboard` method. This enables your application's users to create and customize their own ad hoc dashboards and visualizations using a subset of features of the full design experience to ensure all types of users can instantly take advantage of its capabilities.

Note: Ad hoc dashboards include the ability to customize metric sets. Users must have a power user seat license or higher to perform this customization.

See the method's arguments for options you can pass for various customizations, plus the same `sessionId` and `embeddedViewServiceName` options as described inEmbed Ad-Hoc. The following example passes the session ID as a shortcut to first calling `setSessionId`, and otherwise uses default options:

```
let container = document.getElementById("visualizationContainer");
dundas.context.embedMetricSet(container, "7b8000df-36fd-4994-b6f6-2f6d96999fab", {
       sessionId: sessionId
});
```

By default, the blank ad hoc dashboard editor opens and allows users to choose an existing ad hoc dashboard to open, or to click in the toolbar to add existing data or other content to create a new one. These are saved in and opened from the project specified by the `defaultProjectId` option, or the user's "my project" if not specified.

Create and Edit an Ad Hoc Dashboard

From the blank editor, clicking on a toolbar button adds content to a new ad hoc dashboard. Users can click the first button to add data via a metric set, choosing from a list of all metric sets made available to the logged-in user according to their set file security privileges.

Added content is always positioned automatically to fill all available space and dynamically resize itself, with a built-in title that users can click to change.

The menu icon in the top-right corner of each content item provides options such as **Delete** to remove it from the dashboard, or **Re-Visualize** for switching between Symphony's visualization types while displaying the same data. Users can also right-click or long-tap directly on data to access the metric set's contextual options such as drill down, filtering, and more.

Metric sets are added to ad hoc dashboards as a copy so that users can make any desired changes to the data displayed, via the **Data Analysis** panel on the right. This is based on the full Data Analysis Panel provided for metric sets, allowing data to be added and removed, grouped, sorted, filtered, and more. Clicking on **Visualization** in this panel allows for changing the way data is visualized, such as by changing color for different values or displaying data in labels and tooltips.

The panels on the right can be switched between like tabs, or clicked to expand or collapse them. The **Properties** panel provides a simplified subset of the property options provided when logging into Symphony directly, for styling and other customizations. For quicker access to specific parts of a visualization, users can right-click or long-tap them directly and choose a properties option such as **Series Properties** for a chart's data point series or **Column Properties** in a table.

After adding multiple metric sets or other content items, the ad hoc dashboard's layout adjusts automatically to accommodate them but the user can adjust the sizing of rows and columns by clicking on the **Grid** icon in the toolbar. Users can click on a visualization to switch to it from another one when using the Properties or Data Analysis panels.

Existing full-featured dashboards and reports that were created directly in Symphony can be added from the toolbar the same way as metric sets, and their data can be interacted with, but settings and styling can't be changed.

Save and Publish an Ad Hoc Dashboard

Ad hoc dashboards are not saved automatically, instead users can click the **Save** button in the toolbar to choose a name and location within the project specified by the `defaultProjectId` option. If no project ID was specified, ad hoc dashboards will be saved and opened from each user's personal project or "My Project".

Once saved, users can click to **Publish** the dashboard using the next toolbar button to allow others with access to the same project to view the changes. Users can publish the same dashboard again to save and share any changes made after that. This is like the check in option in the full Symphony application including saving the revision history. By default, only the user that created it is assigned file security privileges for modifying it, but other users can view it or save a copy, or the file privileges can be changed.

> **Note:** Users can't access other user's personal projects by default, so set a project ID if you intend for different users to be able to access each other's published dashboards.

Using the **Share** button in the toolbar, users can share the ad hoc dashboard as an exported file such as an image, PDF, or Excel. The **Notification** button sets up a notification or alert to export and deliver content via email to selected users/addresses or via other enabled delivery providers.

Ad hoc dashboards are saved in the applicable project like a regular dashboard file, and can be accessed and managed by users logged into the full Symphony application with sufficient access rights or as an administrator. While they cannot be created or edited directly in this mode, they can be opened for viewing, and you can use the **Save As** option in the toolbar to convert it to a full dashboard to edit it.

Create a Dashboard Template

This applies to: *Managed Dashboards, Managed Reports*

Use a dashboard template to define a common base or starting point from which to design dashboards. For example, if you want to create a set of dashboards with the same banner and background graphics, you can put these common elements in a dashboard template, and then use the template to create new dashboards. If you ever need to change the common elements, you just need to do this in one place by modifying the template instead of changing multiple dashboards.

Related video: Dashboard Template

Creating a Template

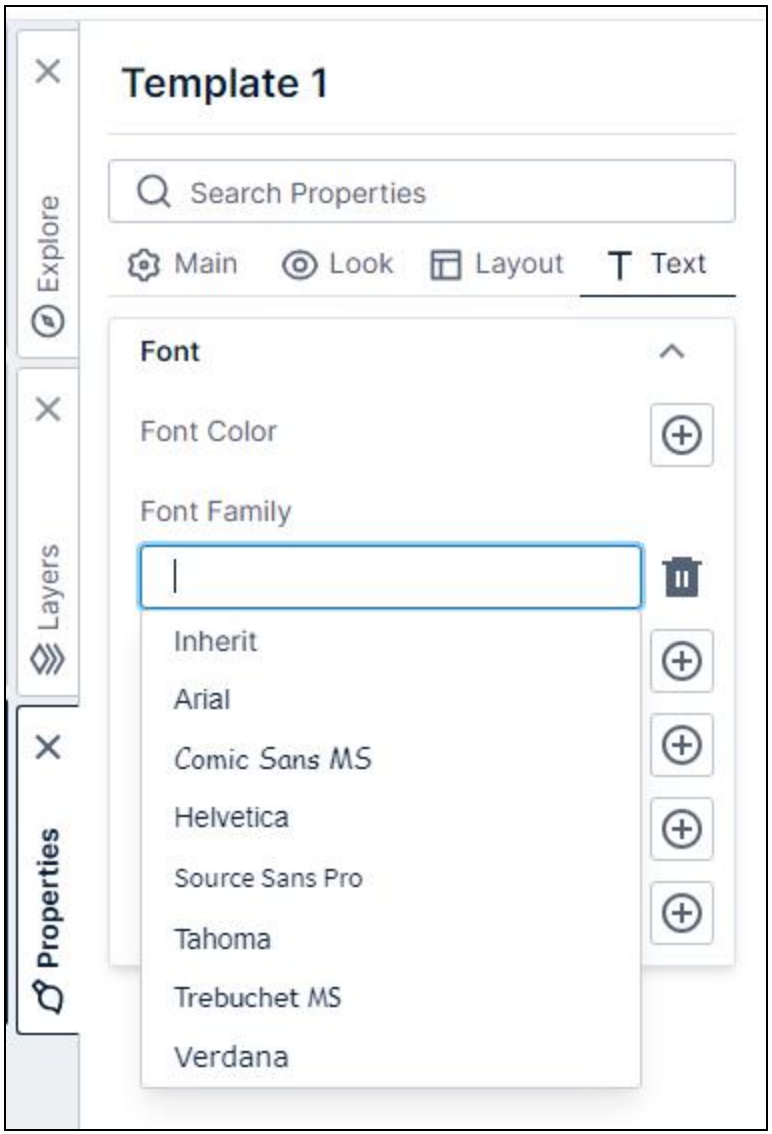
To create a new template, click on **Views** in the menu, click **Create**, and then click **New Template**.

Your new template is edited just like a new dashboard, and is saved under the **Dashboards** folder.

Adjust the Properties

You can adjust some canvas properties that will affect all of the dashboards based on this template. This includes changing the canvas background, the re-size mode, and the font settings.

For example, open **Properties** and indicate the default font settings in the **Text** tab.

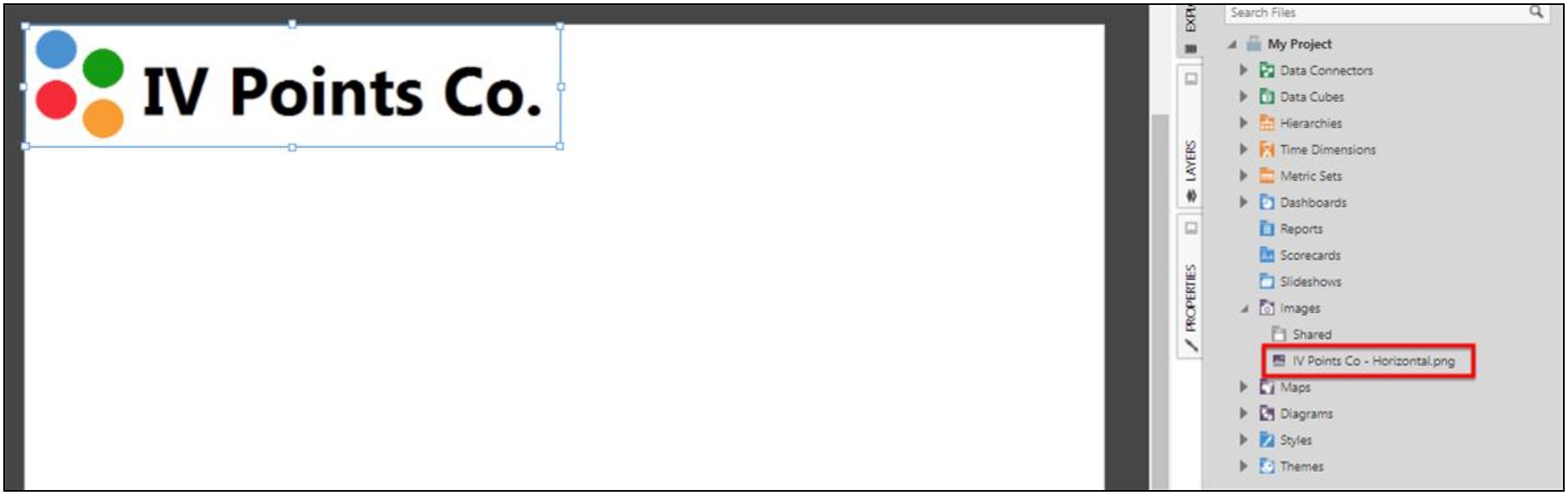


Note: This will define the default dashboard-wide font settings, which can be overridden on individual elements.

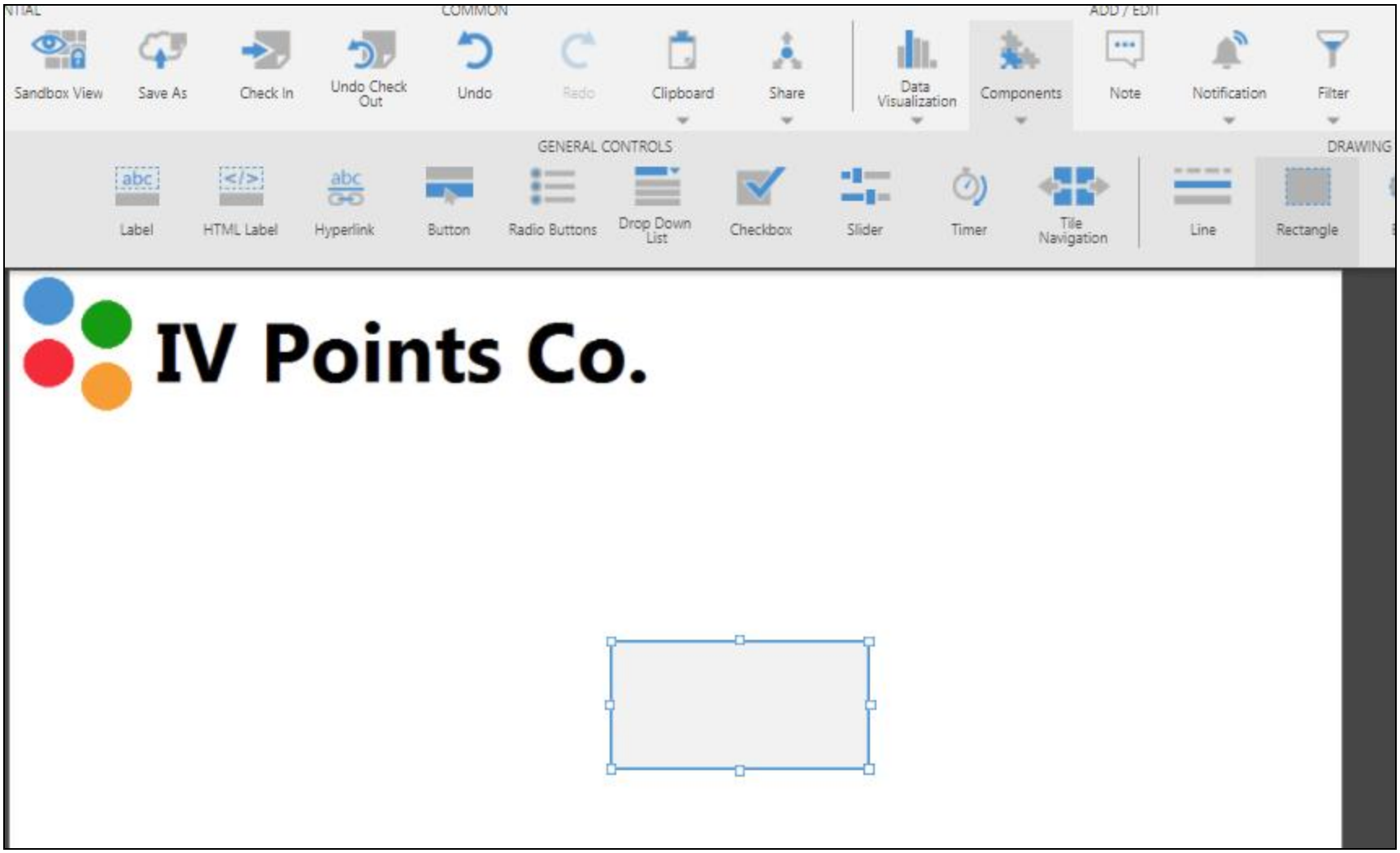
Add Elements

You can add elements to the canvas for your template just like you would for a regular dashboard.

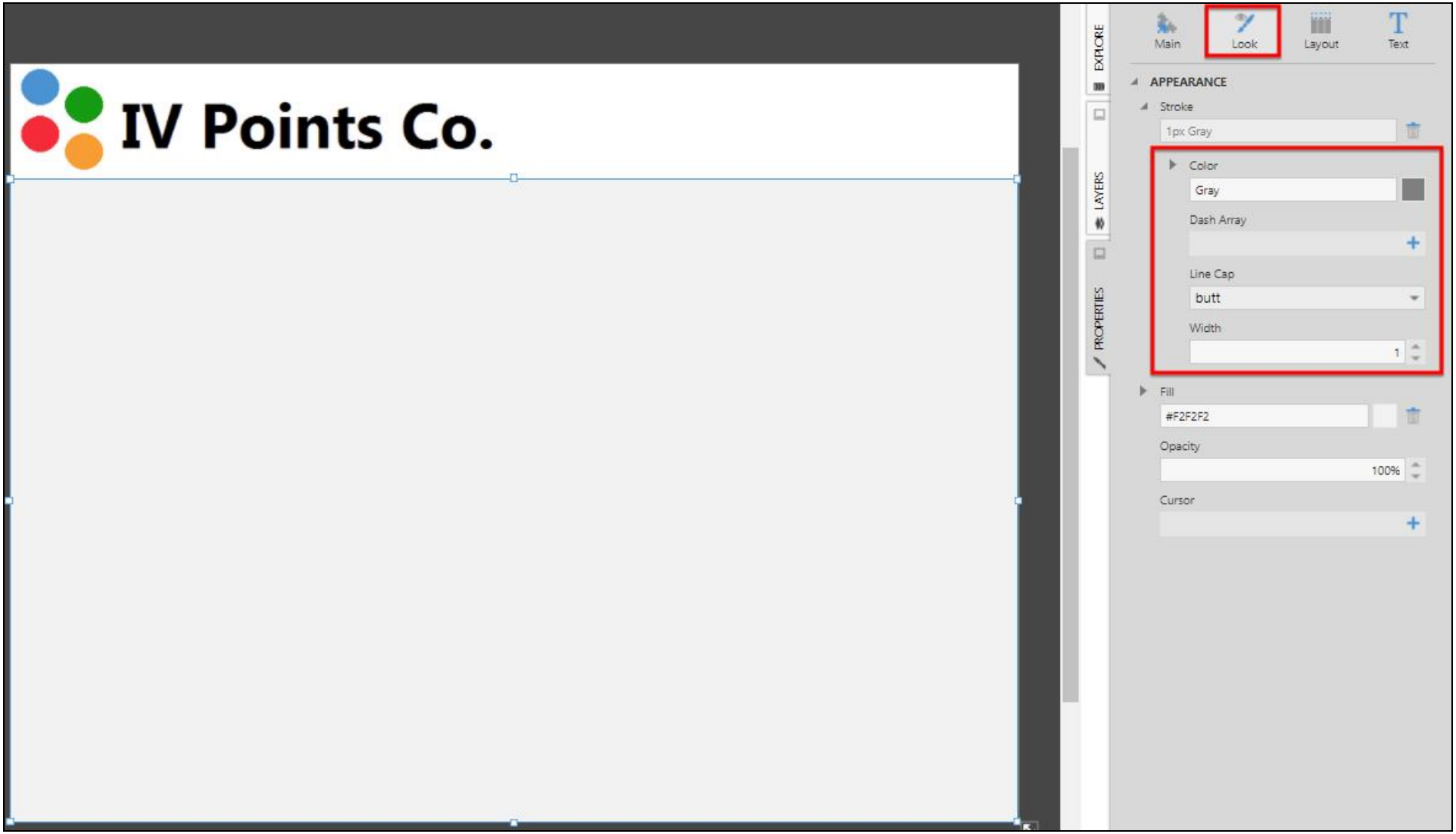
For example, drag an image file from Windows Explorer or Finder and drop it directly onto the template canvas. The image can be used as a banner for a series of related dashboards.



Next, click **Components** from the toolbar and select **Rectangle**. This rectangle will serve as a background frame for data visualizations.



Reposition and resize the rectangle, then set its appearance properties from the **Properties** window.



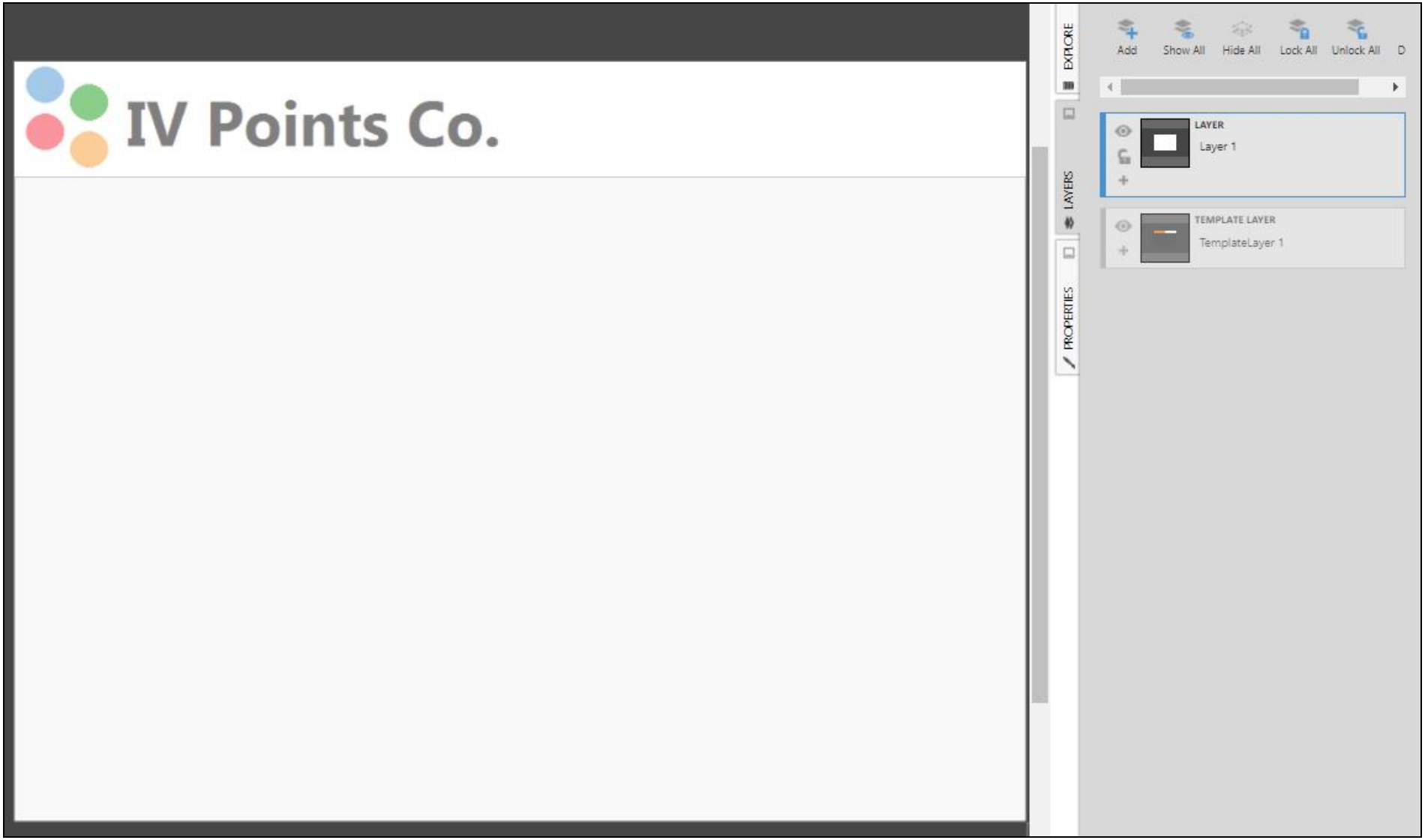
Using a Template

- You can use a template when you create a new dashboard: click **Views** in the main menu, click **Create**, and then choose a template from the list that includes the templates you created.

- When editing an existing dashboard, click an empty area of the canvas to de-select any elements, click **Template** in the toolbar, and then choose **Use Existing Template**.

The dashboard will show elements from the template grayed out. You won't be able to move any of them because the template is locked.

If you open the **Layers** window, you'll see that there is a **template layer** corresponding to the template this dashboard is based upon. It's possible to have more than one template layer if your template has multiple layers. Template layers appear grayed out in the **Layers** window, but you can still show or hide them, or click the plus sign to see the elements within the layer. The lock icon is not available for template layers because they are always locked.

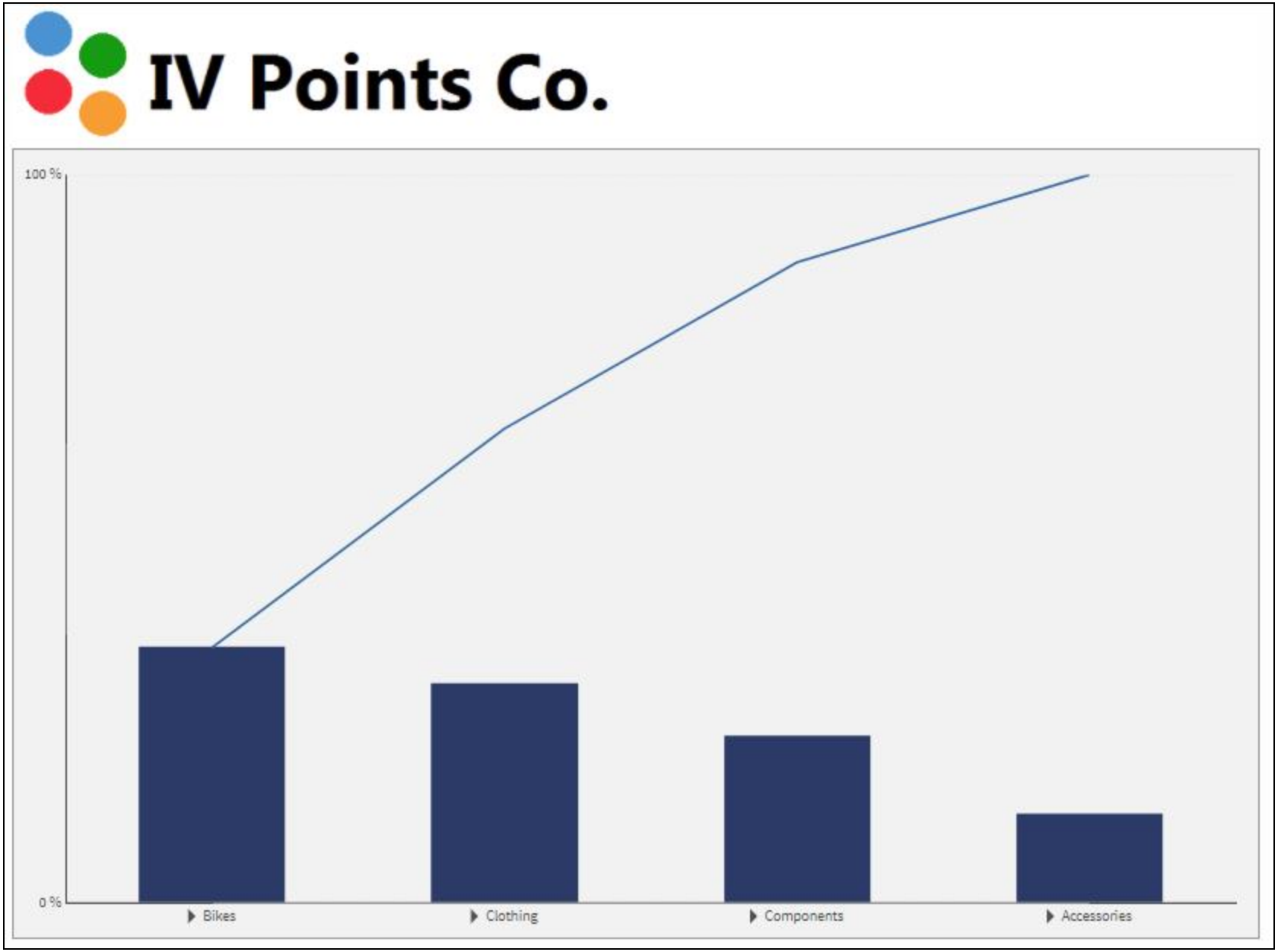


Note: You cannot reorder template layers in the Layers window. You can, however, reorder regular layers relative to the template layers.

You can add your own content to the dashboard in addition to the template.

For example, drag a metric set from the **Explore** window and drop it onto the canvas. Then reposition the resulting visualization on top of the background rectangle.

Switch to **View** mode to see the completed dashboard.


IV Points Co.
100 %
0 %
Bikes
Clothing
Components
Accessories

Detaching a Template

In some situations, you may need to change the template on one dashboard without it having an effect on the other dashboards.

With nothing selected on the canvas, click **Template** in the toolbar, then **Detach Template**.

> **Note:** The option to Detach Template will not be available if there is a conflict of IDs or script names between the template and dashboard. This may happen if you manually changed script names in a conflicting manner.

Unlike when using the Clear Template option, all of the elements and settings are carried over from the template to the dashboard. They still appear grayed out and are located in the template layer, but you can now move and edit them.

For more information, see:

- Video: Dashboard Template
- Layers and Groups
- Using a Template Grid for Resizing
- Design Overview

Styles and Themes

This applies to: *Managed Dashboards, Managed Reports*

Styles and themes give you an easy way to apply predefined settings to your visualizations, components, or filters to give them a consistent style or to reuse previous settings.

A typical scenario for creating styles and themes is when you have several related dashboards or reports and you want all their contents to have a consistent appearance from one dashboard to the next. You can also apply one of the existing themes or styles provided.

Related video: Styles and Themes

Styles

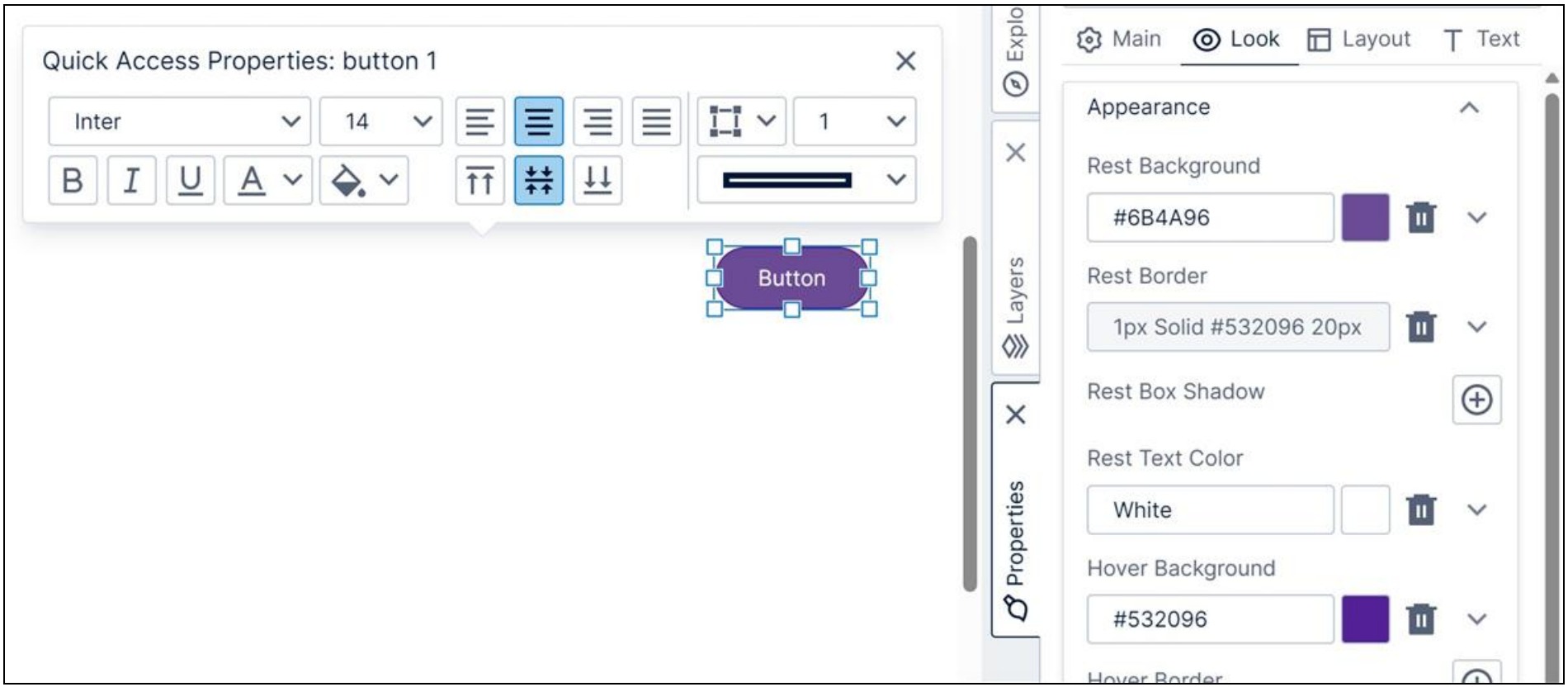
A style contains the settings for most of the properties of a component, filter, or data visualization. The only exclusions are properties that are typically never shared, such as references to a specific metric set, or the specific text displayed in a label or button.

You can save your own style and then easily apply it to another of the same type of element, or you can apply one of the existing provided styles. The following steps work the same way for any component, filter or visualization, including the visualization in a full screen metric set.

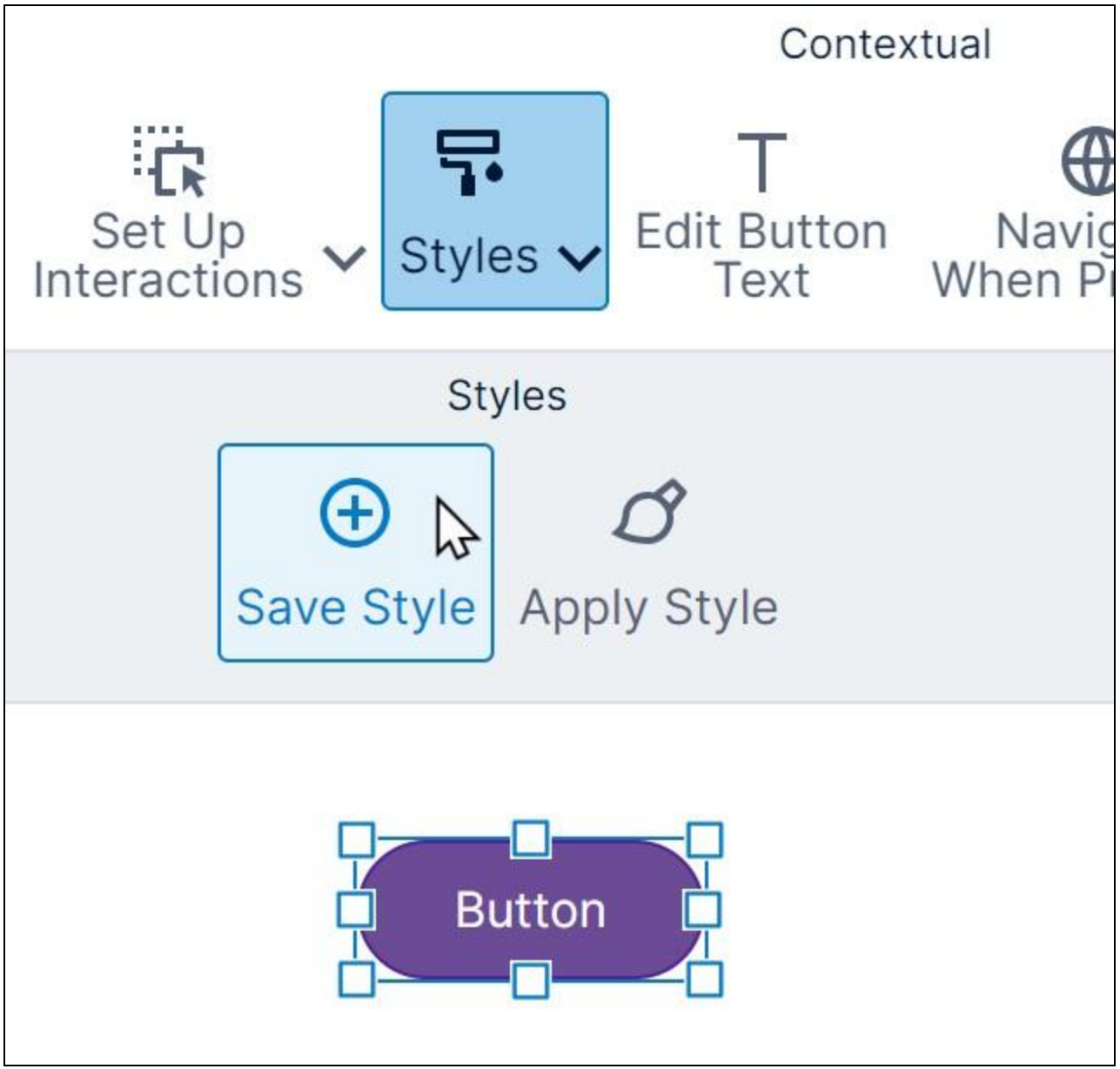
Save a Style

As an example, create a new blank dashboard, and click **Components** and then **Button** in the toolbar to add a button component.

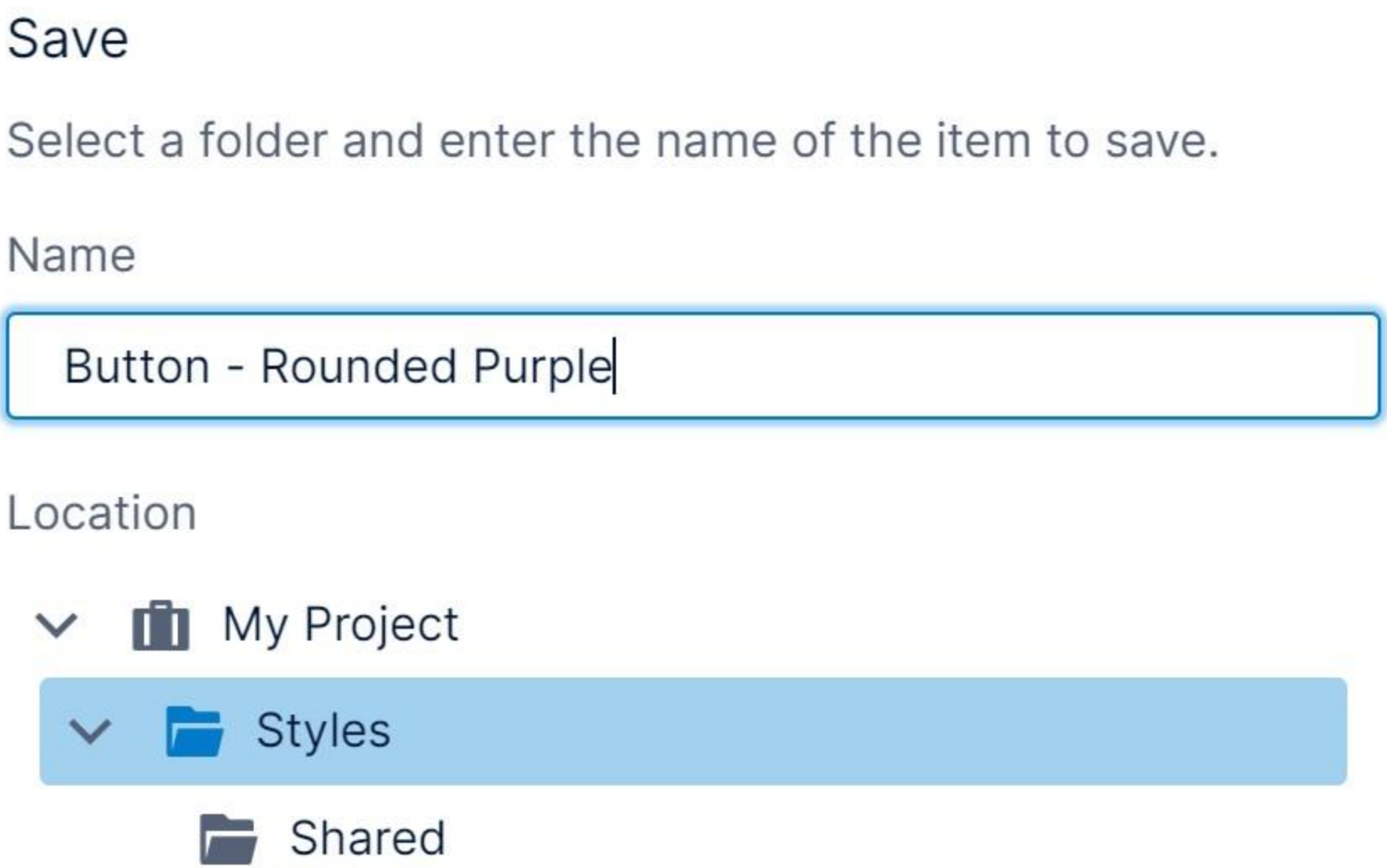
Use Quick Access Properties or the **Properties** window to change some settings for the button. For example, switch to the **Look** tab in the Properties window and customize some of the 'rest' properties.



In the toolbar, click **Styles**, and then click **Save Style**.



In the **Save** dialog, select a folder location and enter a file name for the style. You may want to include the type of the element in the name for easy reference.


Save
Select a folder and enter the name of the item to save.
Name
Button - Rounded Purple
Location
My Project
Styles
Shared

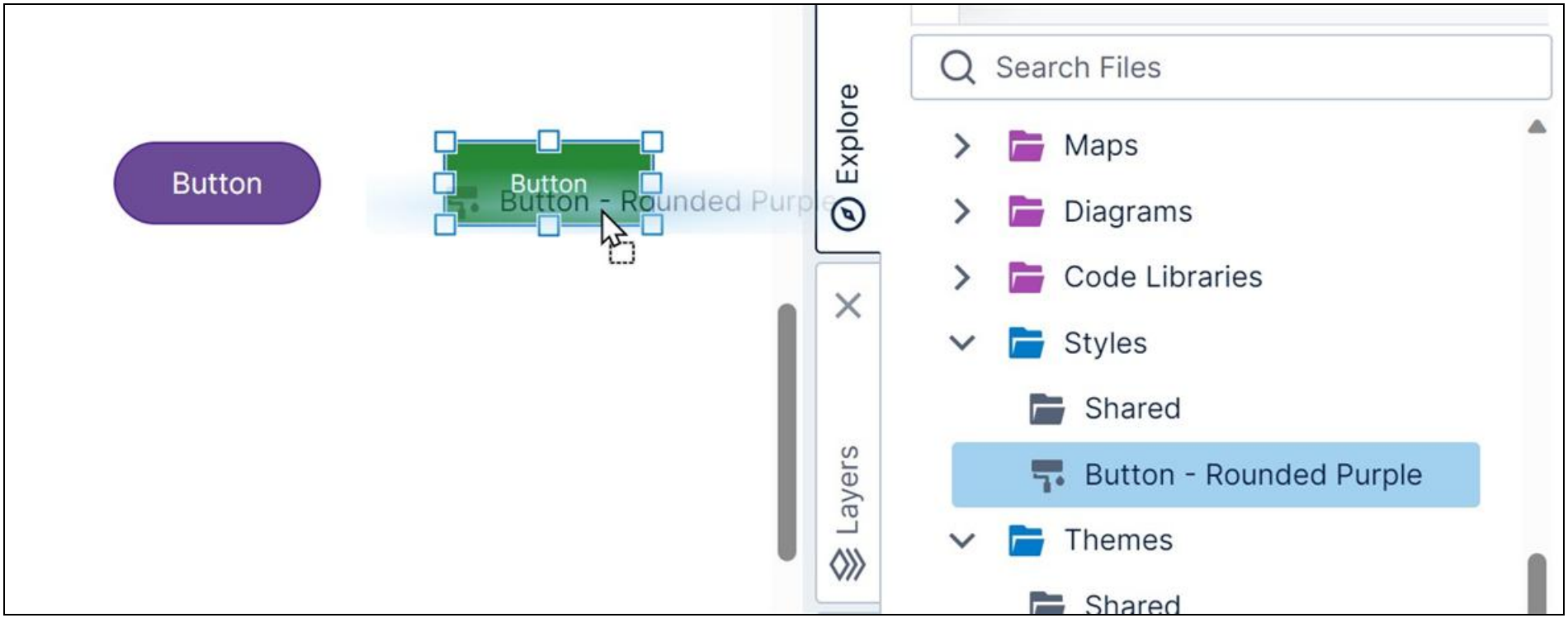
Click **Save** at the bottom of the dialog.

If you expanded the **Styles** folder earlier in the Explore window as shown below, your new style may not appear until you right-click the folder and choose **Refresh**.

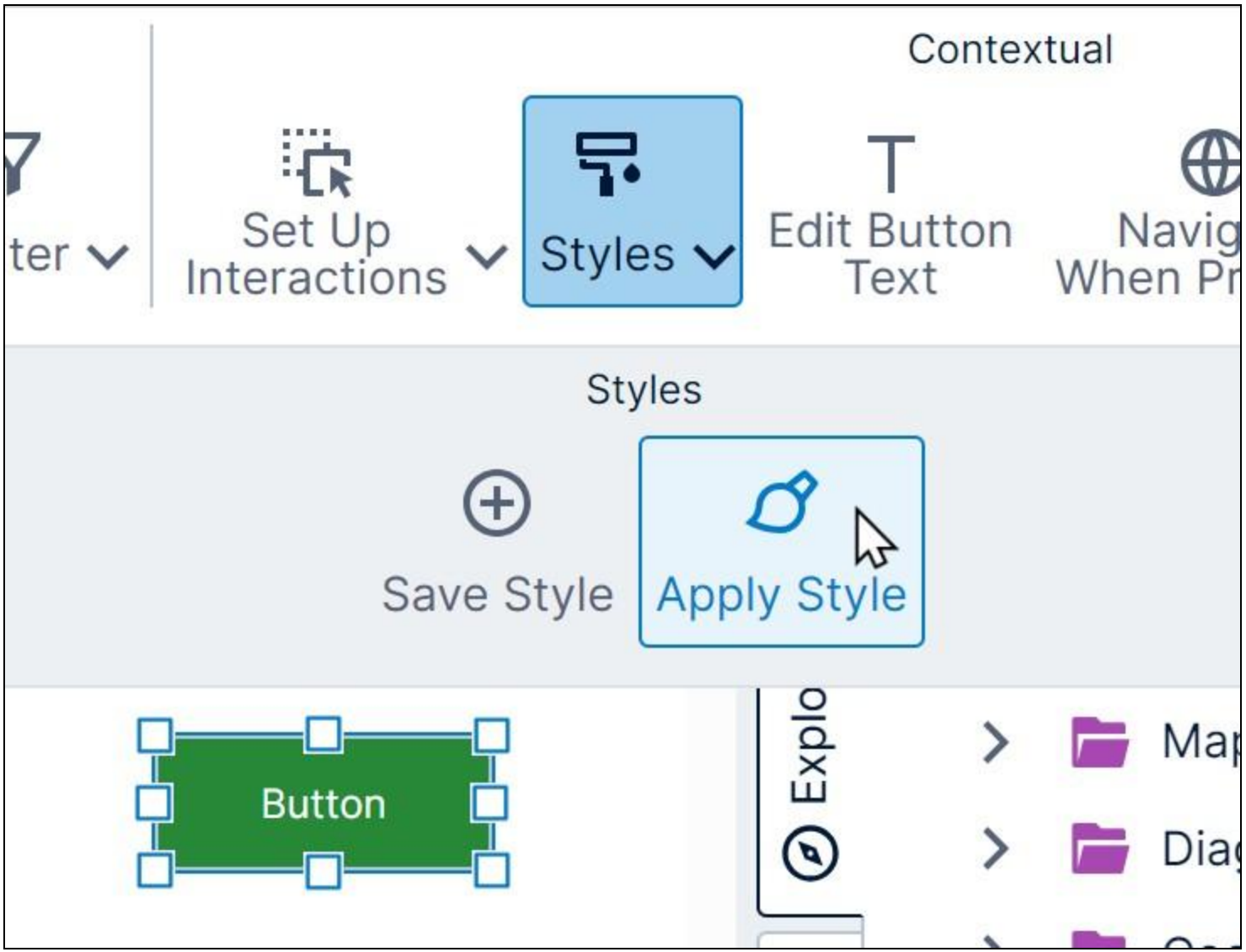
Apply a Style

Continuing the example, add a new button to the dashboard. There are a couple of ways to apply a saved style:

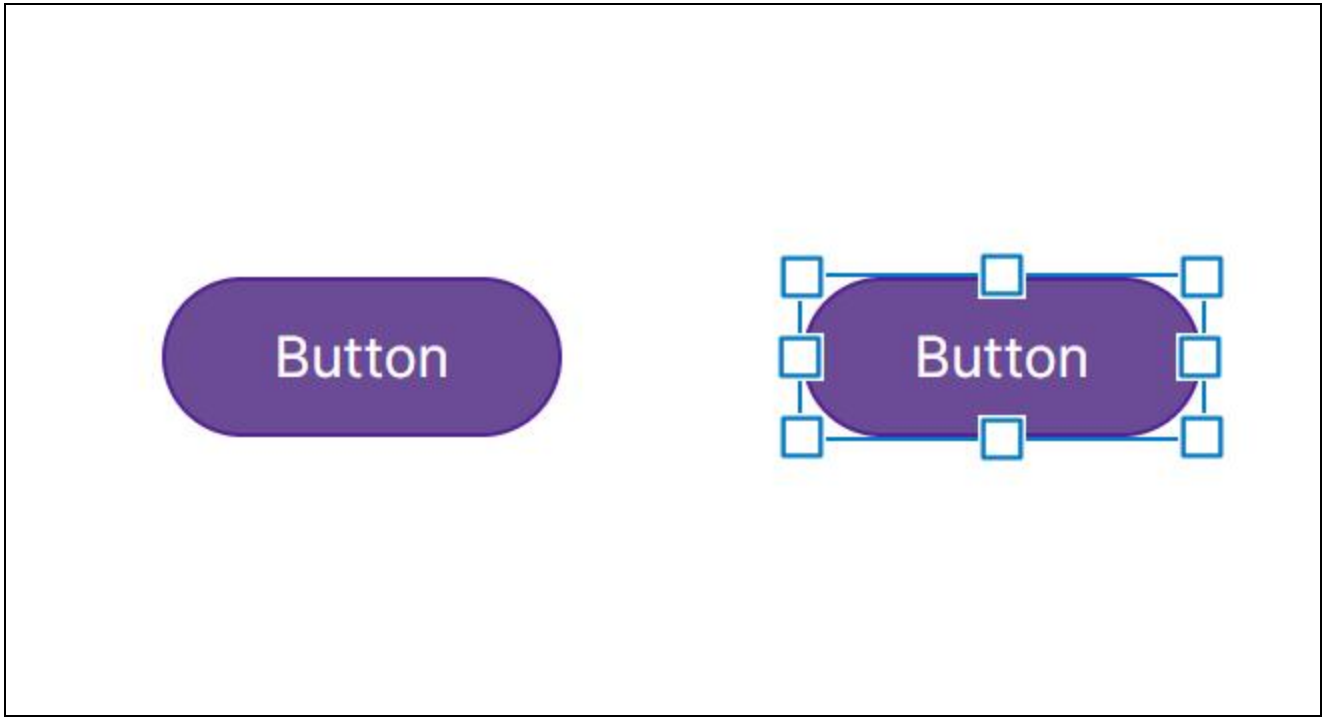
You can go to the **Explore** window and expand the *Styles* folder, then drag a style onto an element (the button in our example).



Another way is to click to select an element on the canvas if applicable, then click **Styles** in the toolbar and **Apply Style**. Select a style in the list that appears, then click **Open**.



In our example, we now have two similarly-styled buttons.



When applying a style, you can expand the **Shared** subfolder under **Styles** to find various styles provided for you. Choose one named with an element type that matches your element, for example, **Button**.

Compatibility

You can usually only apply a style to the same type of element that was used to save the style. For example:

- Each type of gauge uses separate styles, for example, **radial** vs. **linear** gauges. A single gauge can have multiple pointers with different pointer types, so pointer properties are only applied to matching pointers.
- **Sparkline**, **Data Bar**, and **Horizontal Plot** are specialized versions of charts in reports and scorecards. These are not compatible with regular charts so that a theme can contain separate styles for line charts (with visible axes) and for sparklines (without visible axes), for example.

For visualizations, there are certain exceptions where you can apply the same style to different visualization types:

- A single chart can contain different chart types on each data point series, so any chart style can be applied to any other chart. Data Point Series properties from the style are only applied to series with a similar chart type setting: for example, a style created from a line chart can be applied to a bar chart, but the

styling of a line series will not be applied to the bar series because these are very different.

- Styles for all of the relationship diagram types are compatible with each other because they are each made up of nodes and links that have similar settings.

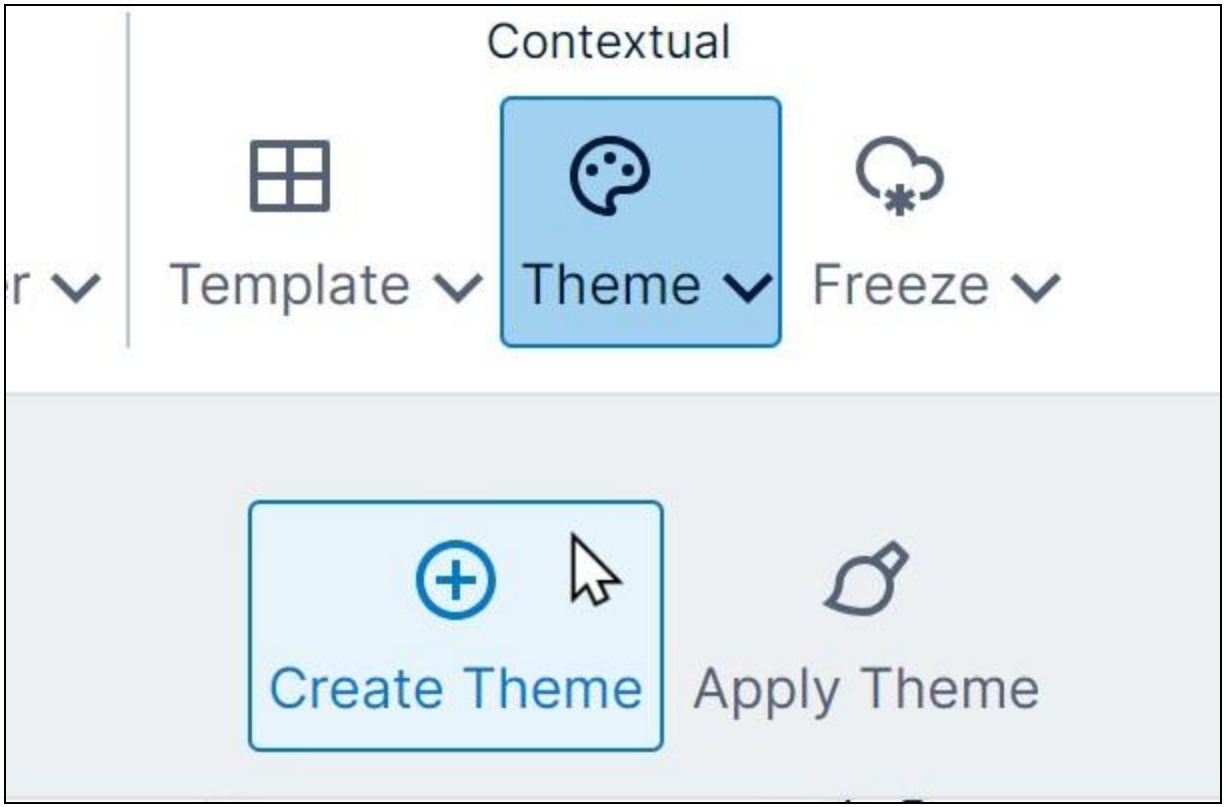
Note: Try to apply styles before making your own customizations. Applying a style may change existing settings you might not have thought of.

Themes

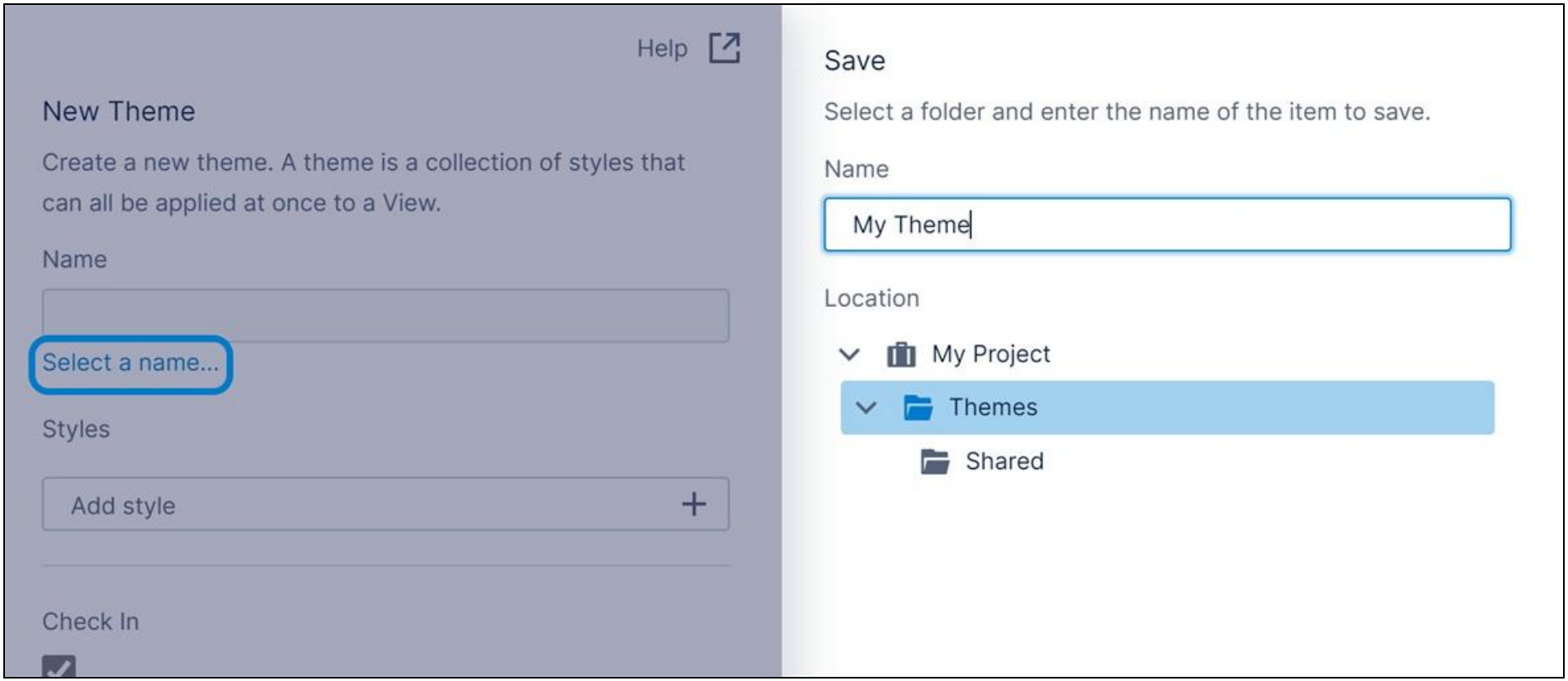
A theme is a collection of existing styles, which you can apply all at once to a view such as a dashboard or report.

Create a Theme

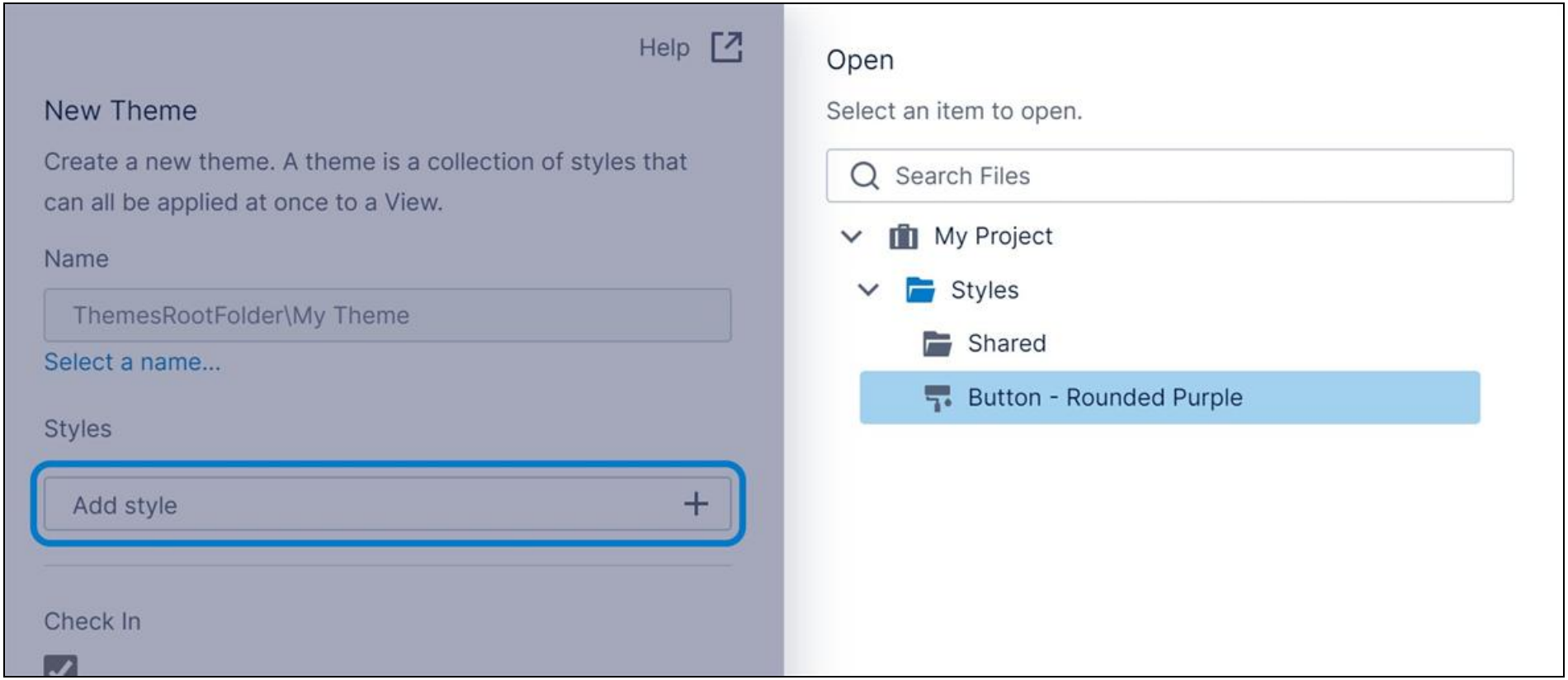
To create a theme, click **Theme** in the toolbar, then **Create Theme**. (If an element on the canvas is selected, click an empty area first to de-select it.)



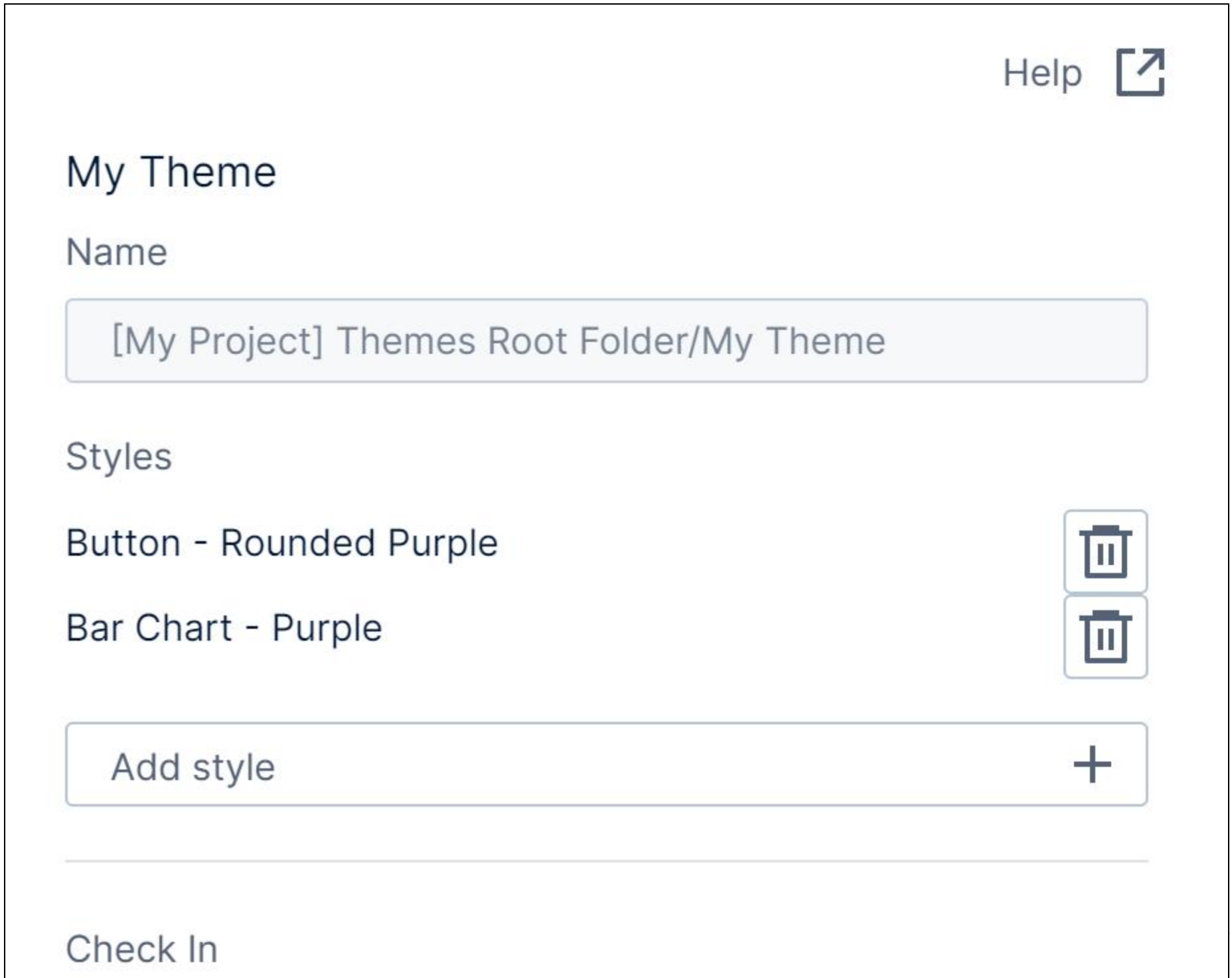
In the **New Theme** dialog, set a **Name**. In the **Save** dialog that appears, optionally also change the selected folder **Location** for the theme, then select **Save** at the bottom.



Returning to the **New Theme** dialog, select **Add style**. In the **Open** dialog, select an existing style to be added to the theme and select **Open** at the bottom.



Add other styles to the theme as desired.


Help
My Theme
Name
[My Project] Themes Root Folder/My Theme
Styles
Button - Rounded Purple
Bar Chart - Purple
Add style
Check In

Note: Adding multiple styles compatible with the same element will apply them all in order. For example, chart styles are compatible with all other charts.

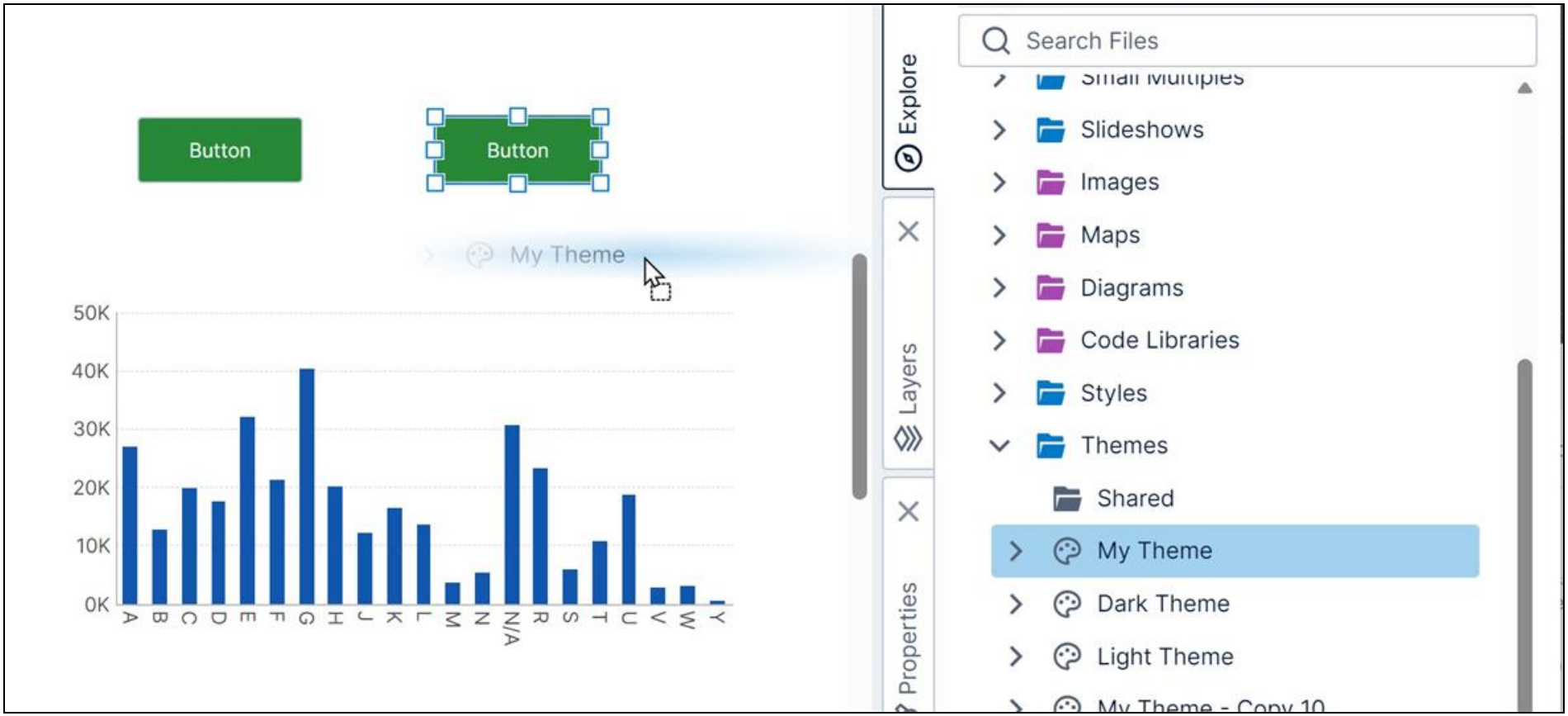
Click the **Save** button at the bottom.

If you expanded the **Themes** folder earlier in the **Explore** window as shown below, your new theme might not appear until you right-click the folder and choose **Refresh**.

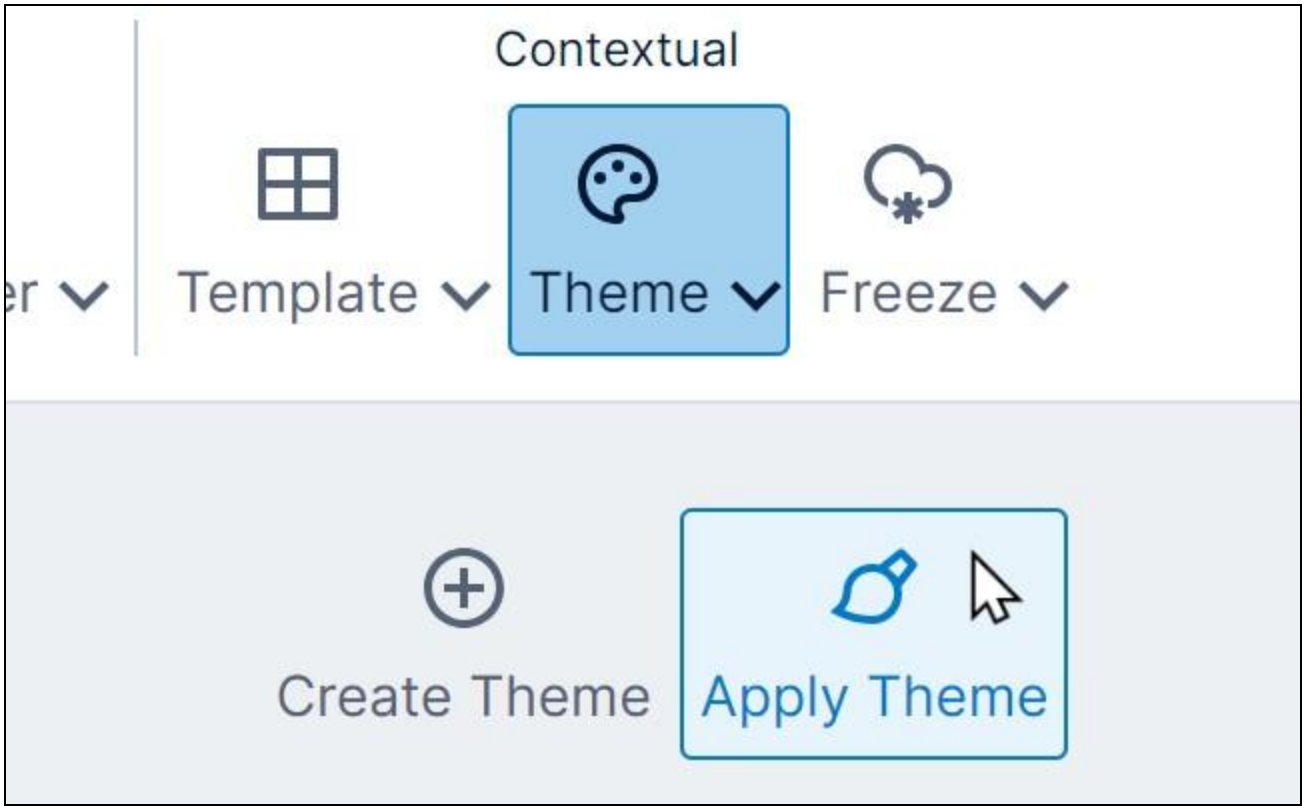
Apply a Theme

In this example, we added a couple of button components and a bar chart with some data to a dashboard. There are a couple of ways to apply a theme to your dashboard, report, or other view:

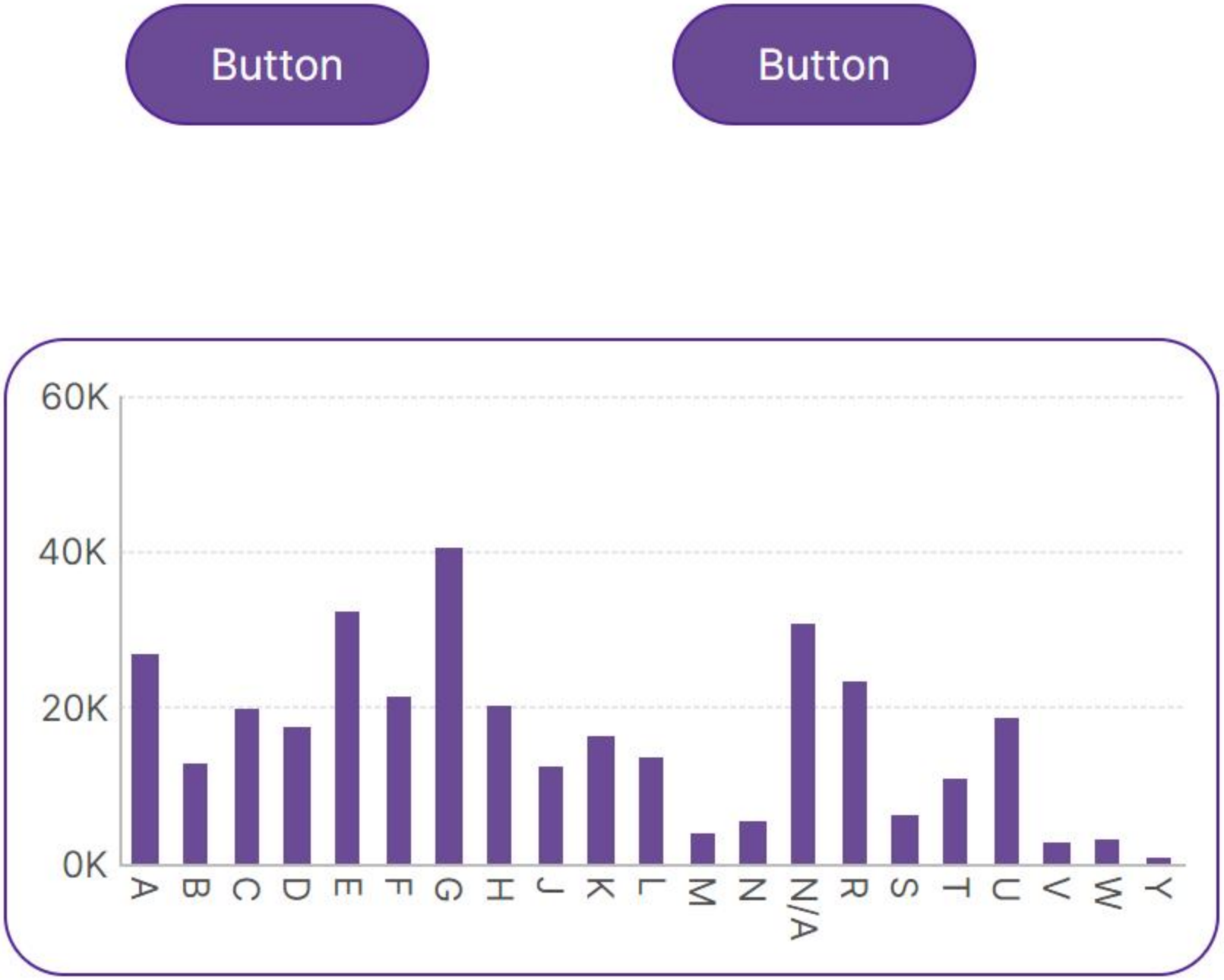
- You can drag a theme from the **Explore** window and drop it onto an empty area of the canvas.



With no elements selected on the canvas, click **Theme** in the toolbar, and then **Apply Theme**. Select a theme and then click **Open**.



The theme is applied to the current view, applying its styles to every element that's compatible.


Button
Button
60K
40K
20K
0K
A
B
C
D
E
F
G
H
J
K
L
M
N
N/A
R
S
T
U
V
W
Y

Setting a Default Theme

Administrators can configure the application so that a default theme is automatically applied when first creating visualizations, components, and

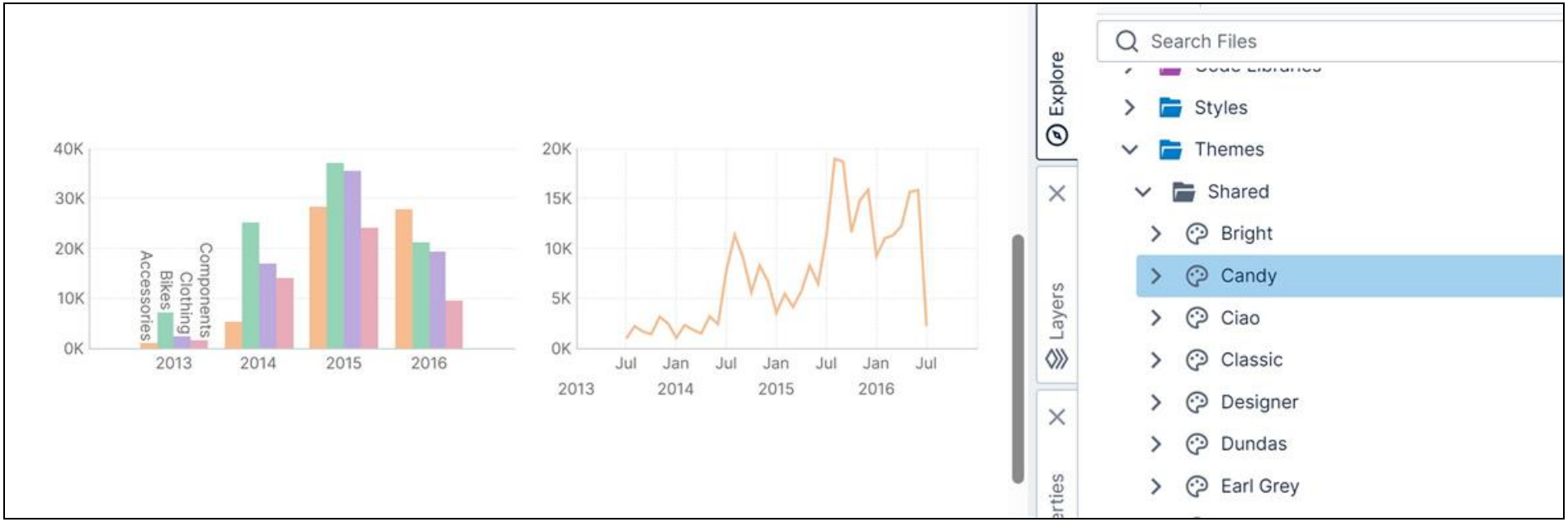
filters, to customize their default colors and other appearance. This includes when data is first added to a visualization, or when re-visualizing the entire visualization when editing or viewing.

See Setting a Default Theme.

Built-in Styles and Themes

The Global project includes a set of built-in styles and themes you can use.

You don't need to switch to the **<Global>** project to access these styles and themes: just expand the **Shared** subfolder under **Styles** or **Themes** in your current project.



For more information, see:

- Video: Styles and Themes
- Working with Gauges
- Setting Up the Visualization
- Using Chart Properties
- Using a Table Visualization
- Working With Projects

Disable View Functionality

This applies to: *Managed Dashboards, Managed Reports*

Application privileges assigned on user groups and accounts by an administrator determine which application functionality and UI options are accessible to each non-administrator user. Some of these privileges, such as the ability to use the context menu, can also be disabled in a particular view or on each of its visualizations. Disabling application privileges this way will apply to all non- administrator users viewing that dashboard, report, or other view, even if they are otherwise granted those privileges.

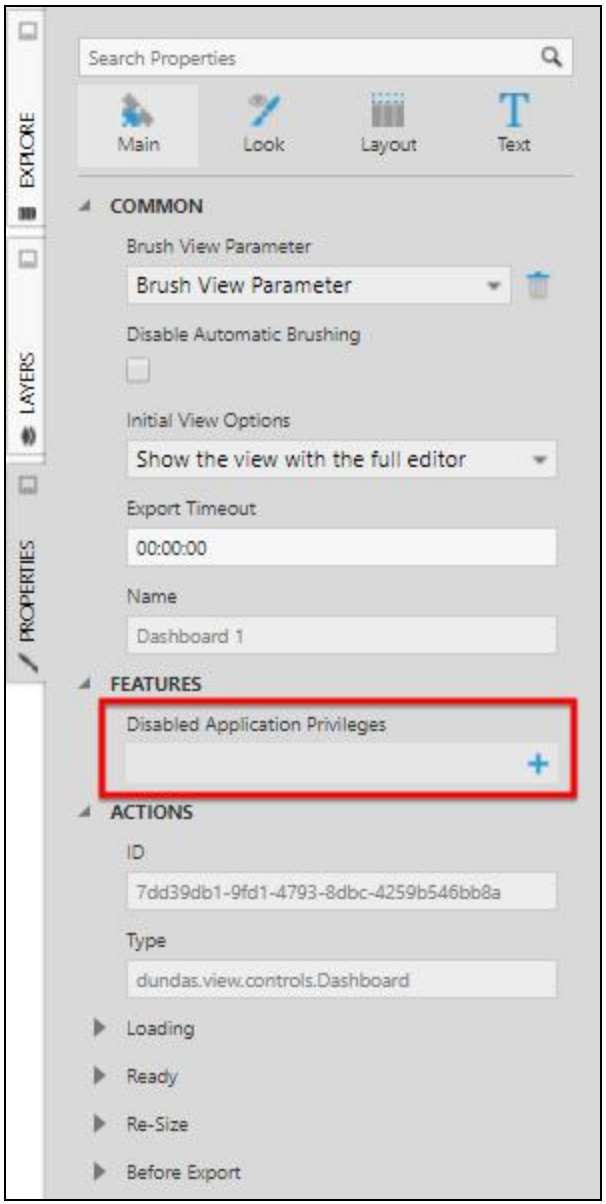
You can also customize view functionality, such as which options should be available under Re-Visualize in the context menu, rather than disabling the corresponding privilege completely.

Note: The properties on a view or visualization can only disable application privileges. To grant additional privileges to users, an administrator must set their application privileges.

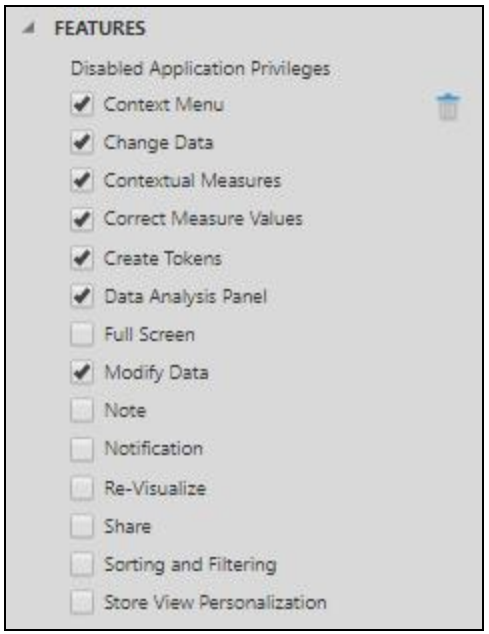
Disable Application Privileges on the View

With nothing selected on the canvas, open the **Properties** window for the dashboard or other view.

In the *Features* category, click the plus under **Disabled Application Privilege**s.



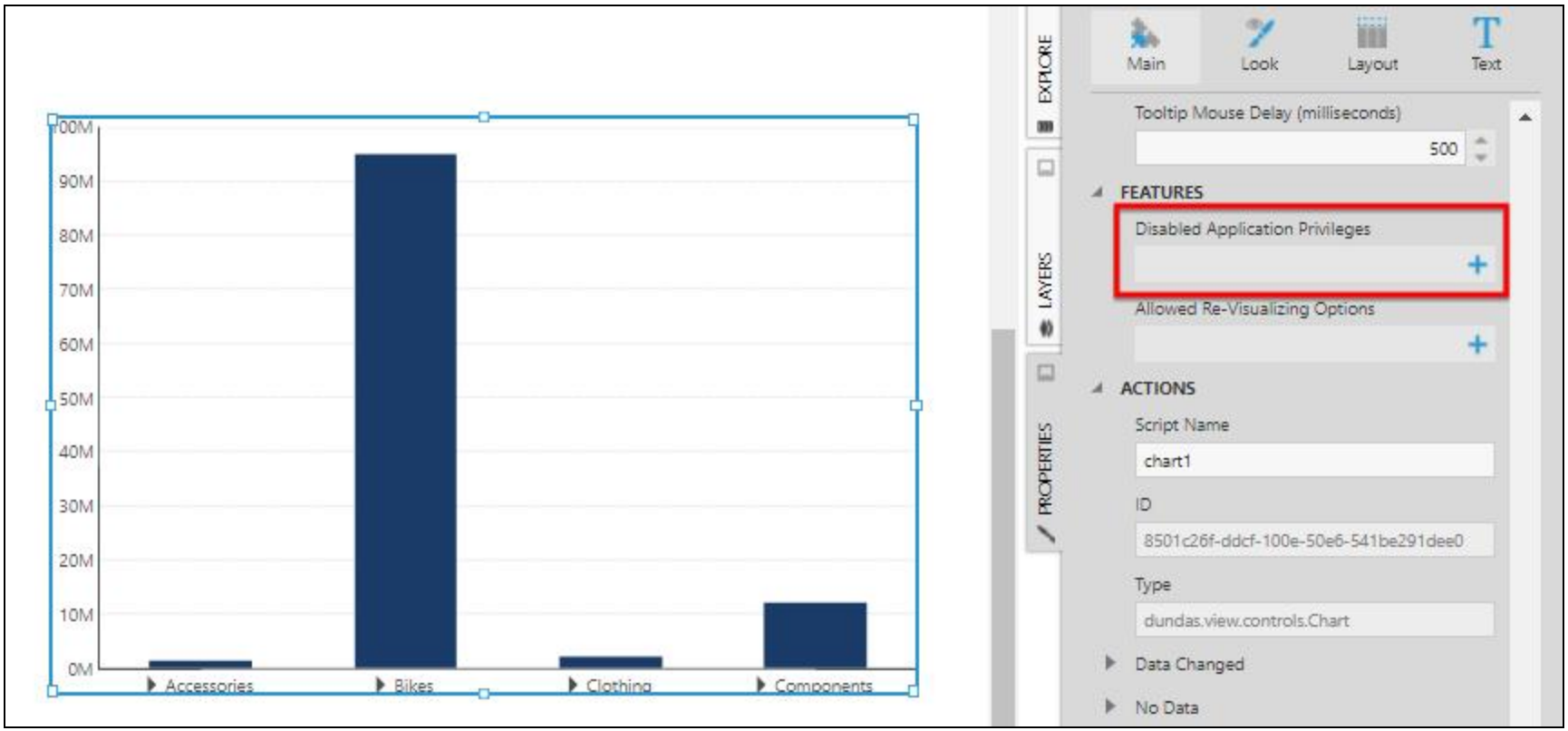
Select all the privileges you want to disable on this view.



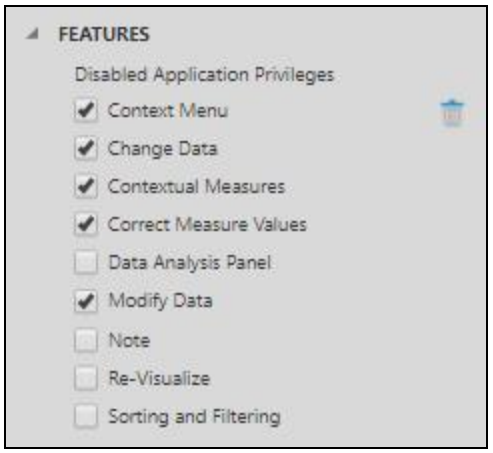
Disable Application Privileges on an Element

Select an individual element of the view such as a data visualization and open the **Properties** window.

In the *Features* category, click the plus under **Disabled Application Privileges**.



Select all the privileges you want to disable.

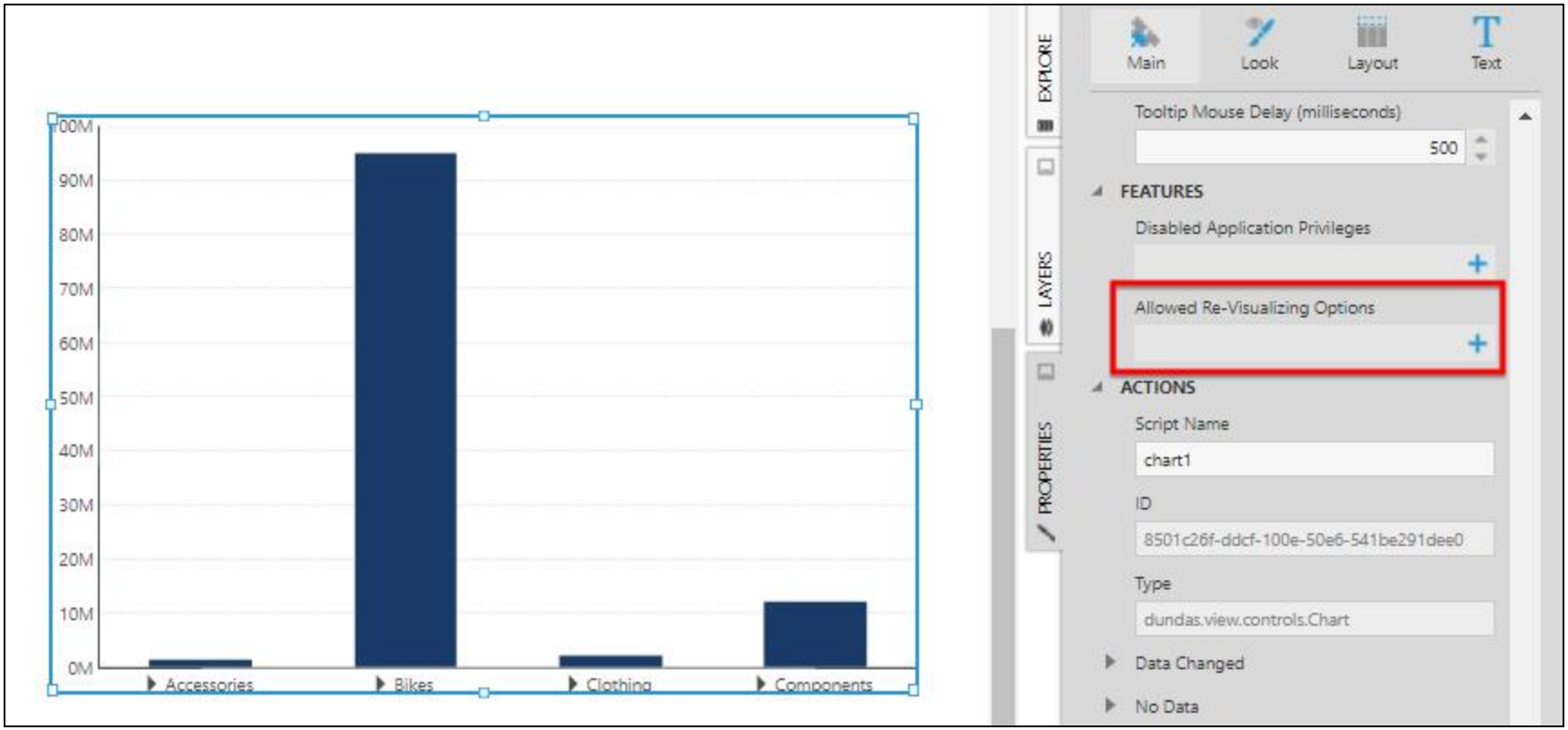


Note: You are indicating which privileges to disable on this specific data visualization. This will not affect other visualizations, even if they display the same metric set.

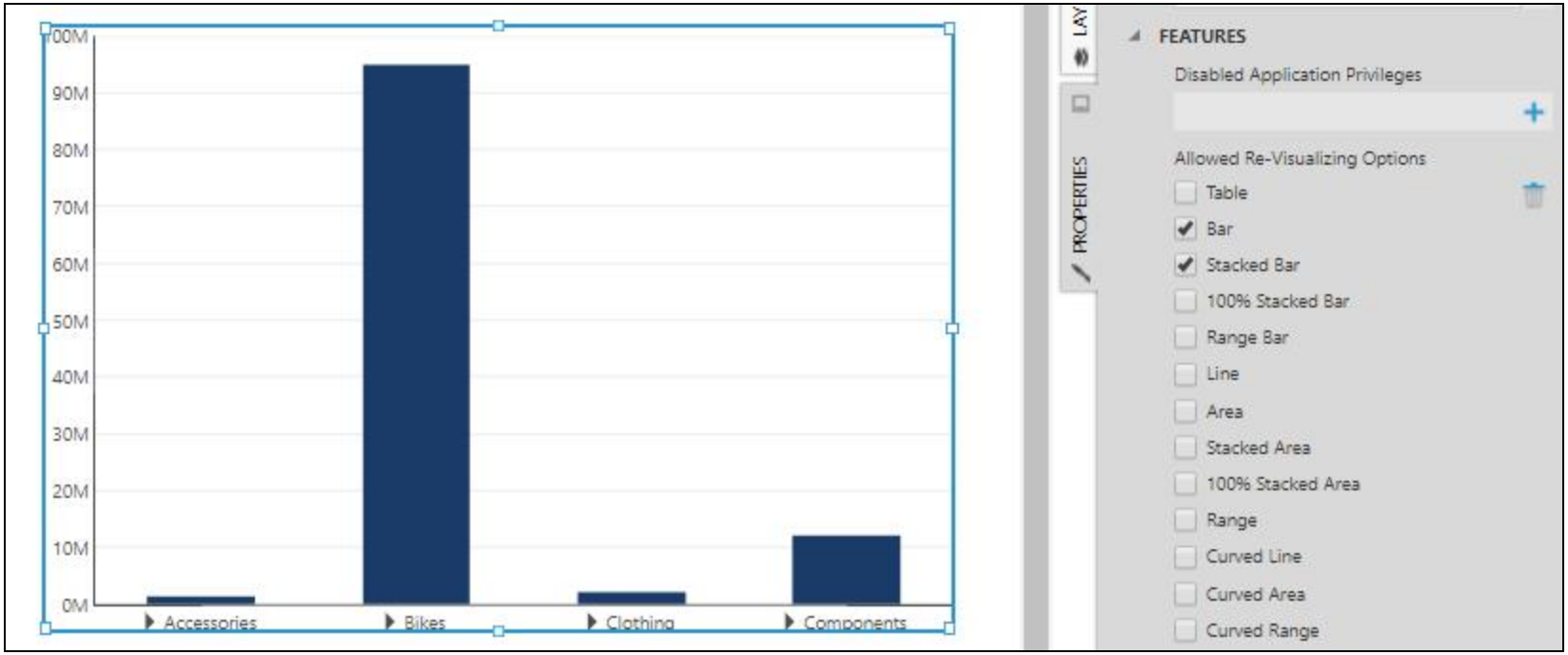
Limit Re-Visualize Options

Select a data visualization and open the **Properties** window.

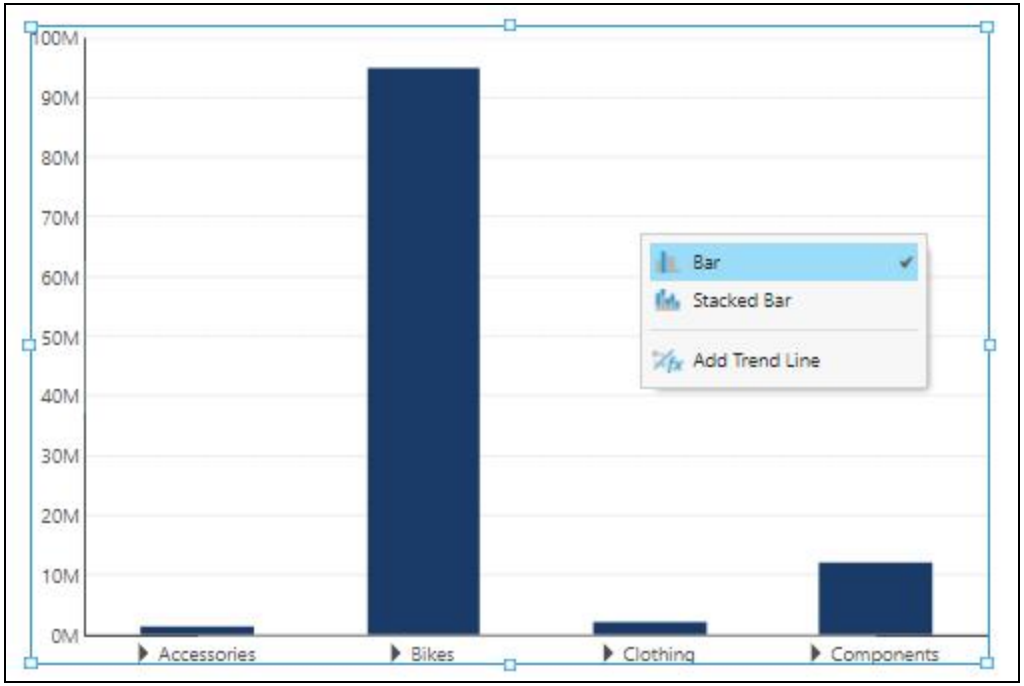
In the *Features* category, click the plus under **Allowed Re-Visualizing Options**.



Select all the visualization options you want to be available for viewers.



To test, right-click the data visualization and select **Re-Visualize.**



For more information, see:

- Application Logs
- Advanced Group Management

Use a Dashboard in a Report

This applies to: *Managed Dashboards, Managed Reports*

You can use an existing view such as a dashboard in your report, scorecard, or small multiple.

Reports, scorecards, and small multiples are able to repeat the same view multiple times, each for a different parameter value. You might also use a report to format multiple dashboards into pages that can be exported to PDF.

Example Dashboard

In this example, we will use *My First Dashboard* from the *Getting Started project*.

Open your instance and switch to the *Getting Started* project.

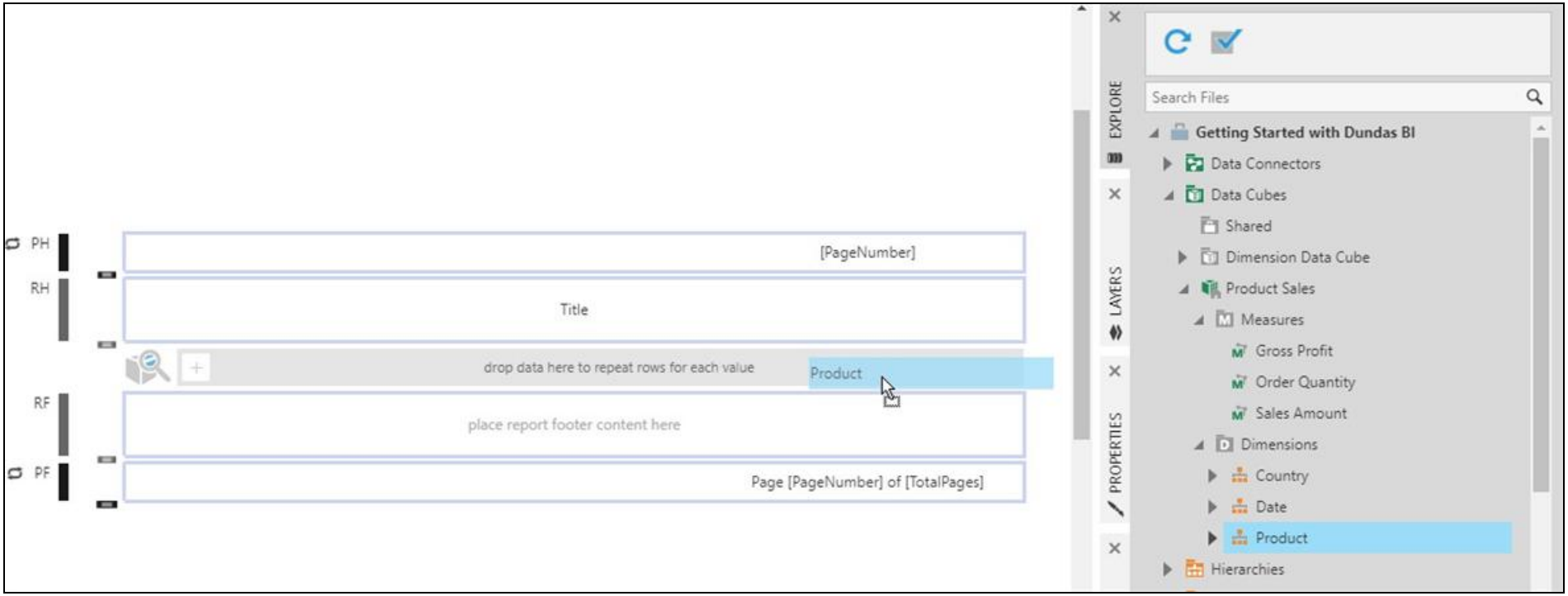
My first dashboard is found in this project.

Repeating a Dashboard

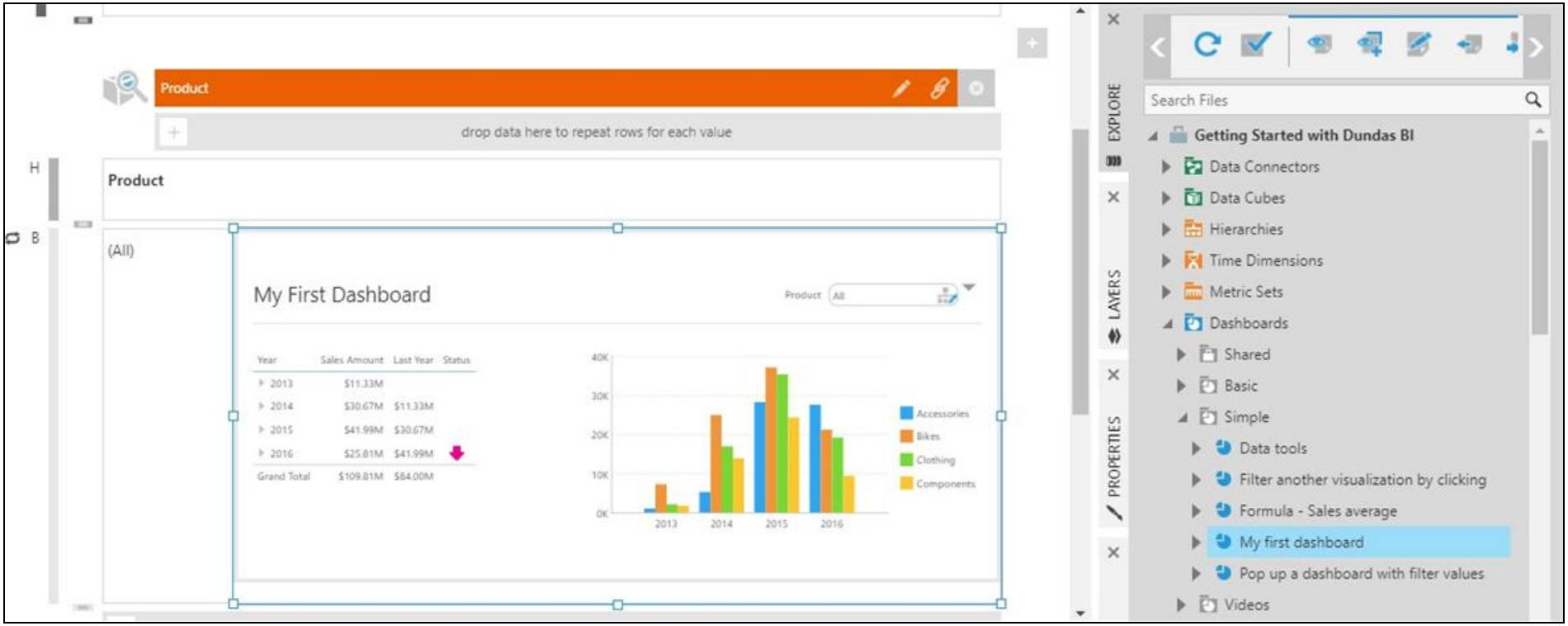
From the main menu, click **New** and select **Report**, for example, to create a new report. The steps are similar for scorecards and small multiples.

In the **Explore** window, locate the *Product Sales data* cube and drag *Product* onto the grouping drop area (drop data here...).

Symphony automatically creates a data label and header for the grouping hierarchy, which are optional and you can delete them if preferred.

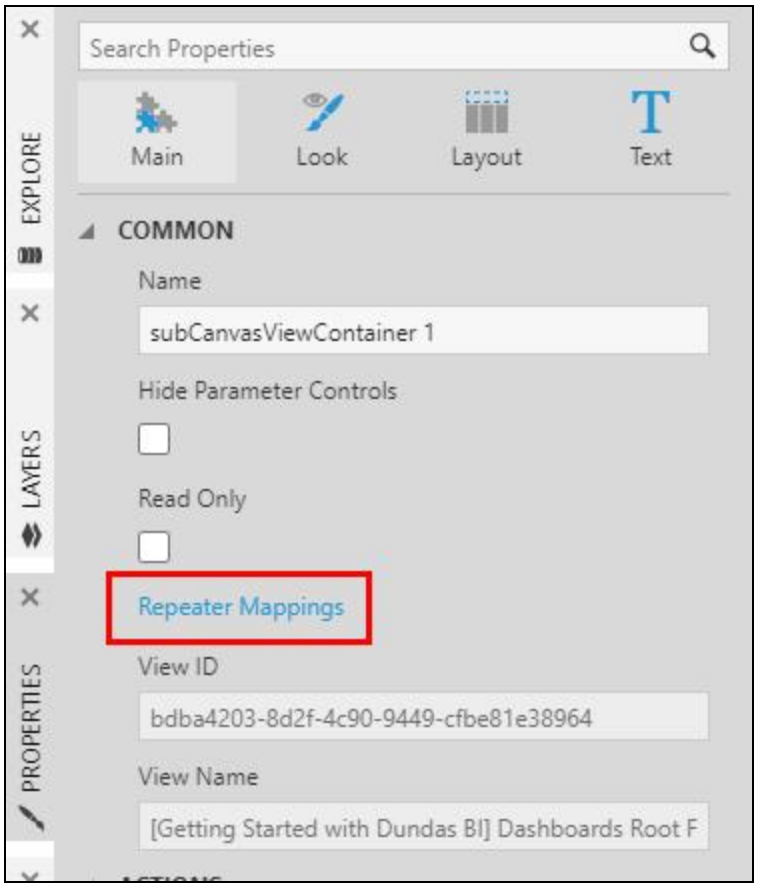


Locate *My first dashboard* and drag it onto the Group Body region. Resize it as needed to fit the dashboard.



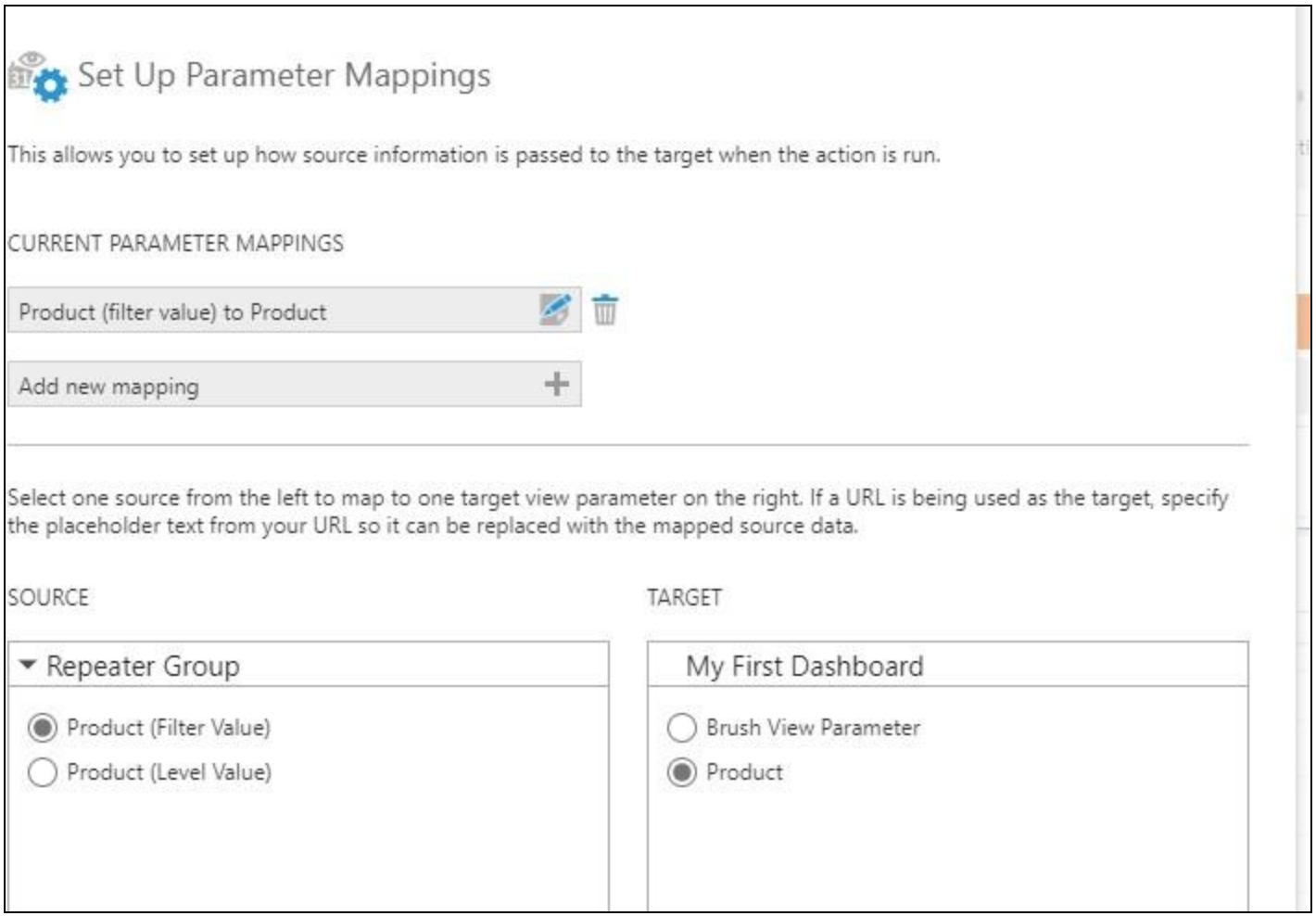
With the dashboard selected, open the **Properties** window.

Click **Repeater Mappings**.



In the *Set Up Parameter* Mappings dialog, click **Add new mapping**.

Select **Product (Filter Value)** under Repeater Group and **Product** under *My First Dashboard*.

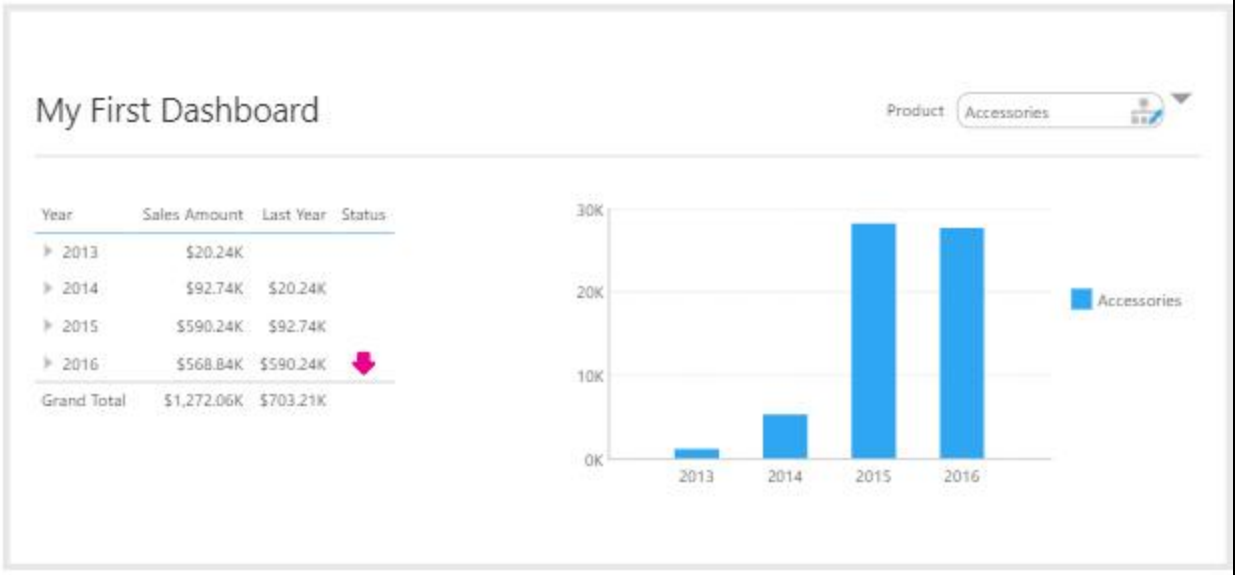


Click the **Submit** button at the bottom of the dialog, and switch to View mode.

Sales by Product

Product

Accessories



Bikes

With the repeater mappings set up, each copy of the dashboard is filtered to a single product category in our example: each bar chart has only a single series, and the filter above it shows the corresponding product category.

When the view contains filters that you don't want displayed while embedded inside of another view, you can go to the **Properties** window with the view selected in edit mode, and enable the **Hide Parameter Controls** option.

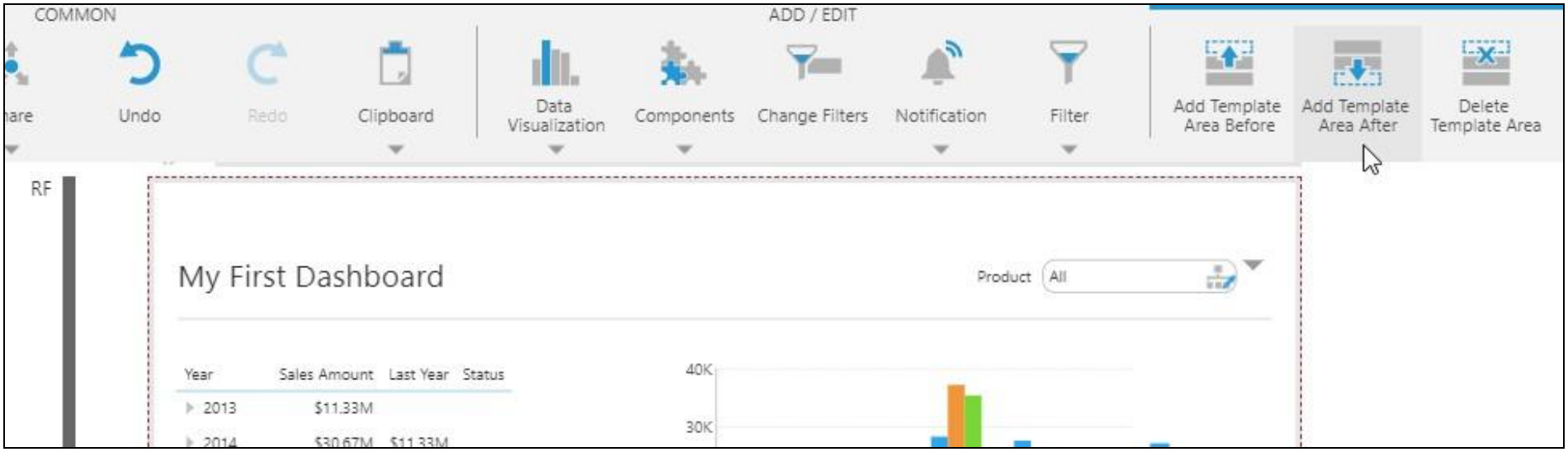
Exporting Dashboards

Rather than repeat a dashboard or view multiple times, you can also use a report to format dashboards or other views into pages to be exported into a single PDF, which can then be shared or saved, delivered by e-mail, or printed.

You don't necessarily need to drag data to repeat any content in your report, and can simply drag views such as dashboards to the Report Footer region, for example.

To add additional template regions, select a region such as the Report Footer. If the region already contains content, you can click the area labeled RF

or similar on the left, or select content inside the region and click **Select Template Cell**.



Click **Add Template Area After** or **Add Template Area Before** in the toolbar, then drag another view into the new region.

For more information, see:

- Symphony Managed Reports
- Symphony Scorecards

- Create Small Multiples
- Using a View Container
- Filter Child Dashboards From a Parent Dashboard
- Design Tips for Reports and Scorecards

Symphony Metric Sets

This applies to: *Managed Dashboards, Managed Reports*

A metric set allows you to select and analyze a set of data in Symphony. It's displayed in your choice of data visualization while you work with one, but can be reused in different visualizations and by other people you share it with.

Metric sets power all visualizations inSymphony. You can work with metric sets one at a time in the full-screen editor, or combine them in a dashboard or another view to edit and analyze them together. This article introduces metric set functions as seen when editing one full-screen, but they are also available when working with metric sets on a dashboard or another view.

The full set of analysis capabilities are available while editing a metric set that is checked out to you, while anyone viewing the metric set is able to access powerful metric set features such as drill down, sorting, and grouping.

Note: When viewing a checked-in dashboard as a power user or above, you can right-click a visualization and choose **Data Tools**, then **Analyze Data** to open a copy of that metric set for editing.

Related videos:

- Introduction to Metric Set Designer
- Using The Data Analysis Panel
- Configuring Metric Sets

Walkthrough: Create a Metric Set and Add a Filter

For more information, see:

- Select And Group Data For A Metric Set
- Move, Reorder, Remove Data For A Metric Sets
- Replace Data In A Metric Set With A Hierarchy
- Visualizations For Symphony Metric Sets
- Sort And Filter Metric Data

Select and Group Data for a Metric Set

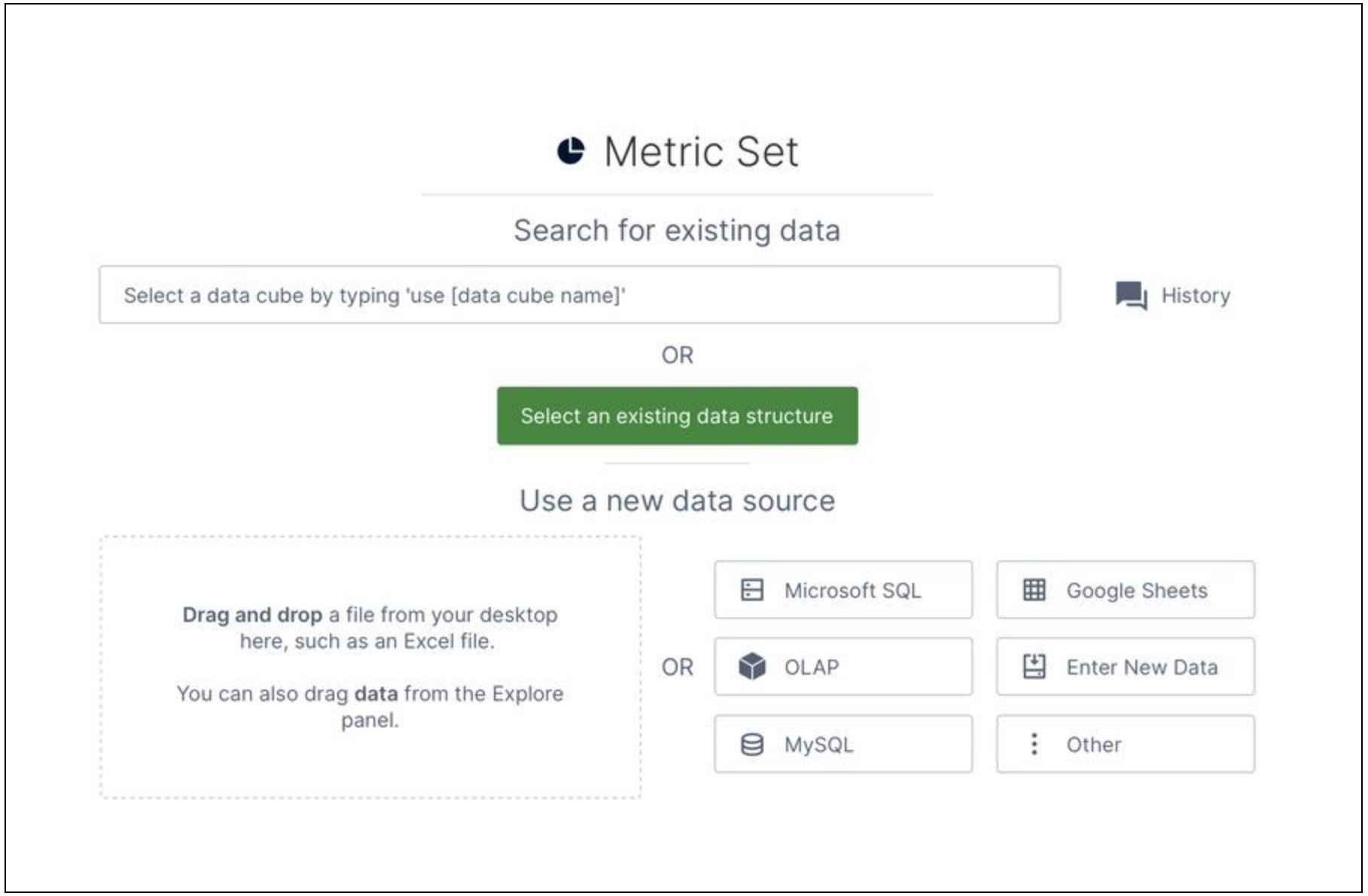
This applies to: *Managed Dashboards, Managed Reports*

This topic provides links to topics related to selecting and grouping data.

New Metric Set

1. To create a metric set in the full-screen editor, select **Business** in the main menu, then **Metric Set**.

2. Start by choosing your first data source. a wide variety of security and permissions



This can be any data found under a data connector or data cube someone has already added to Symphony, or a file from your computer like a spreadsheet.

3. Drag it onto the canvas, or choose **Use an existing data structure** to locate it in Symphony.

4. A table visualization now displays whatever data you chose with the *Data Analysis Panel* to the left.

Note: Create and edit metric sets directly on a dashboard or other view by dragging data onto its canvas, then using the same Data Analysis Panel.

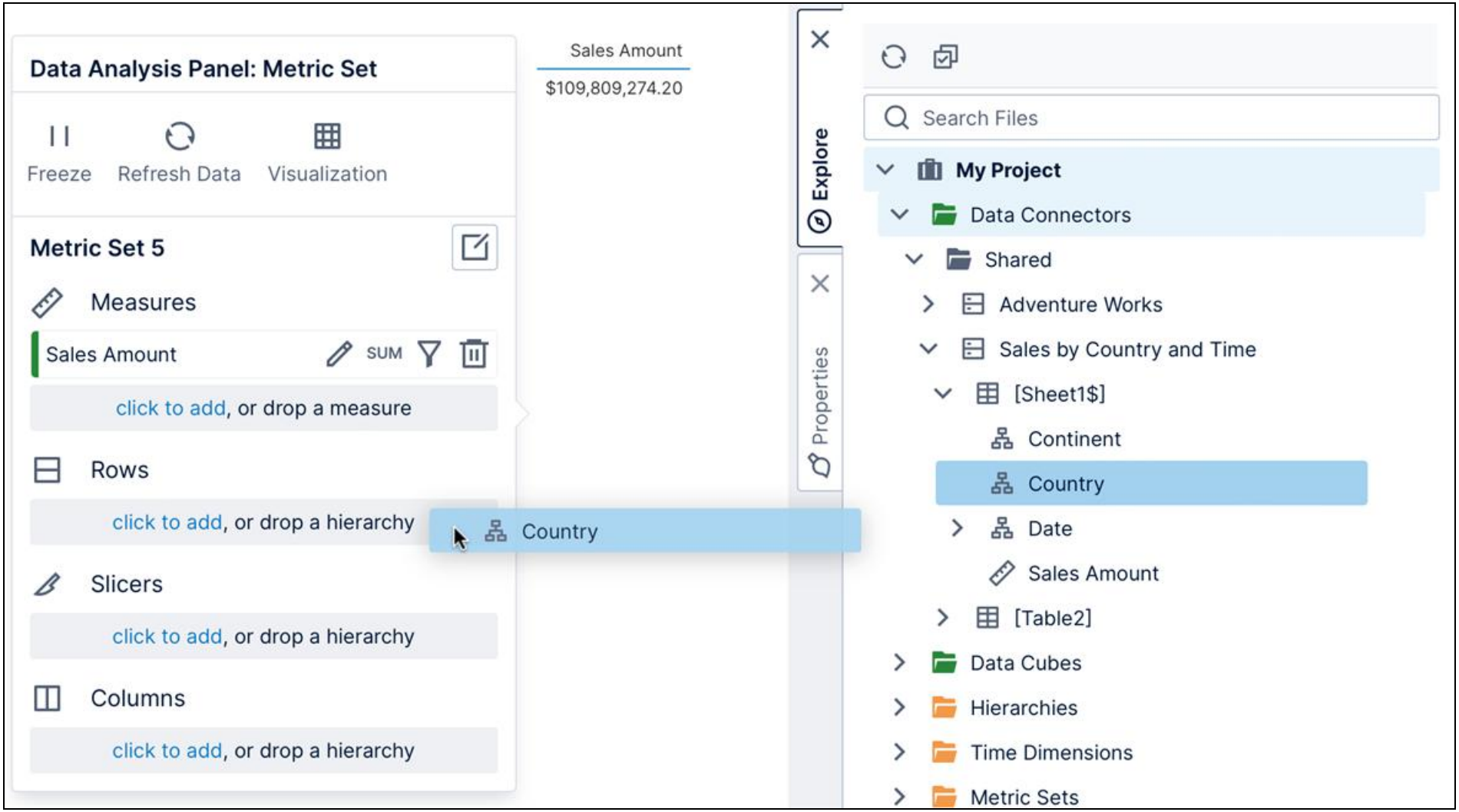
Adding More Data

The Data Analysis Panel is central to choosing and analyzing data. Regardless of which visualization you use, you can group and filter your data by placing it under one of these same four options:

- **Measures**: numeric values that can be grouped and filtered
- **Rows**: group by this data into a row for each value
- **Slicers**: filter the data by the selected values
- **Columns**: group by this data into a column for each value

You can find more details and examples in Slicers versus columns and rows.

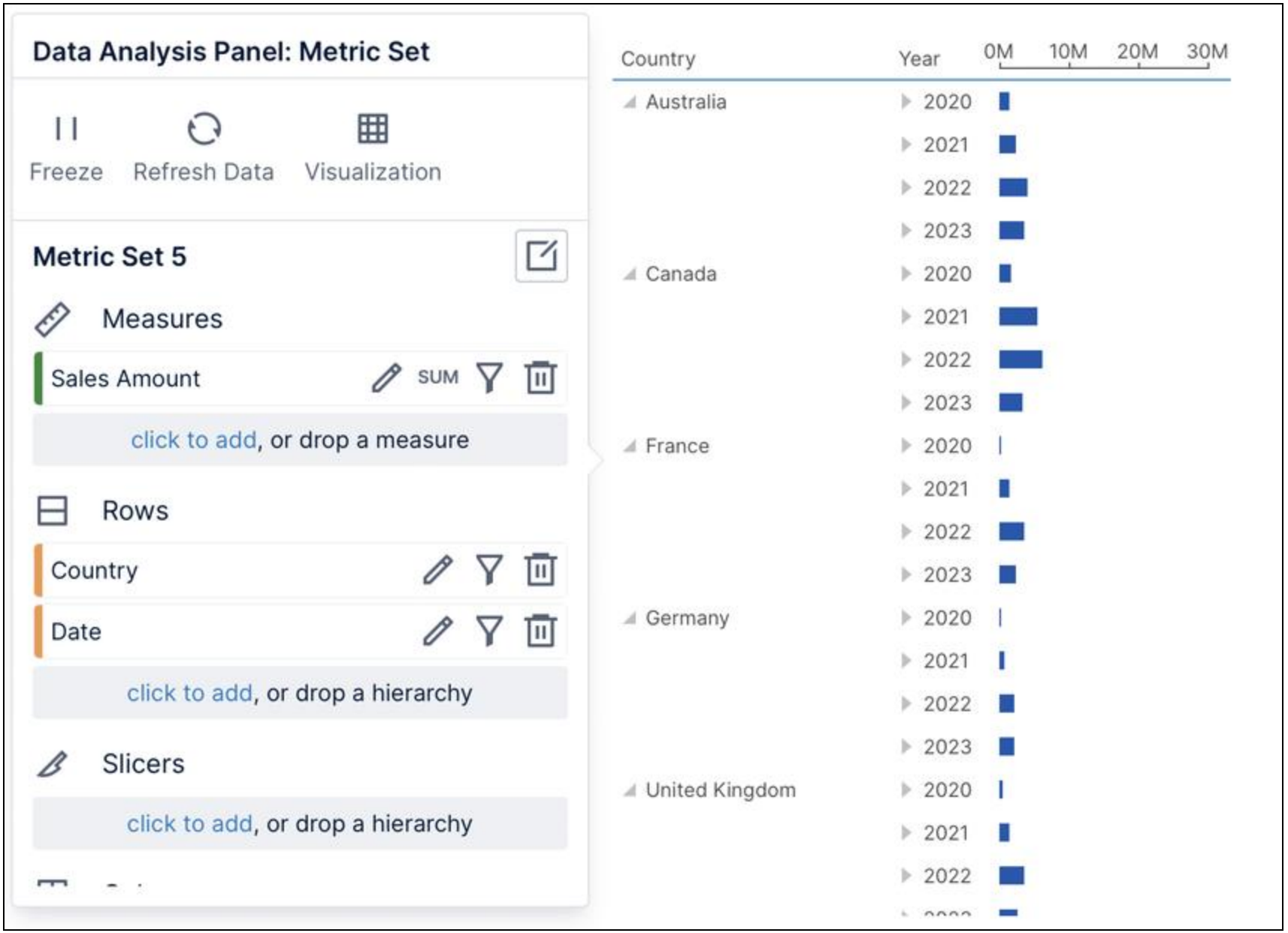
To see more data, drag it from the **Explore** window, or you can **click to add** more data from your current data source.



Note: In the Explore window, different icons indicate whether data is has been categorized as a measure (numeric data) or a hierarchy (usually non-numeric data). If you are working with data from a data connector directly, you can use numeric data as either a measure or hierarchy regardless, or you can right-click and choose **Change Category** to switch between them. (Database primary keys with key icons are typically numeric but used as hierarchies.)

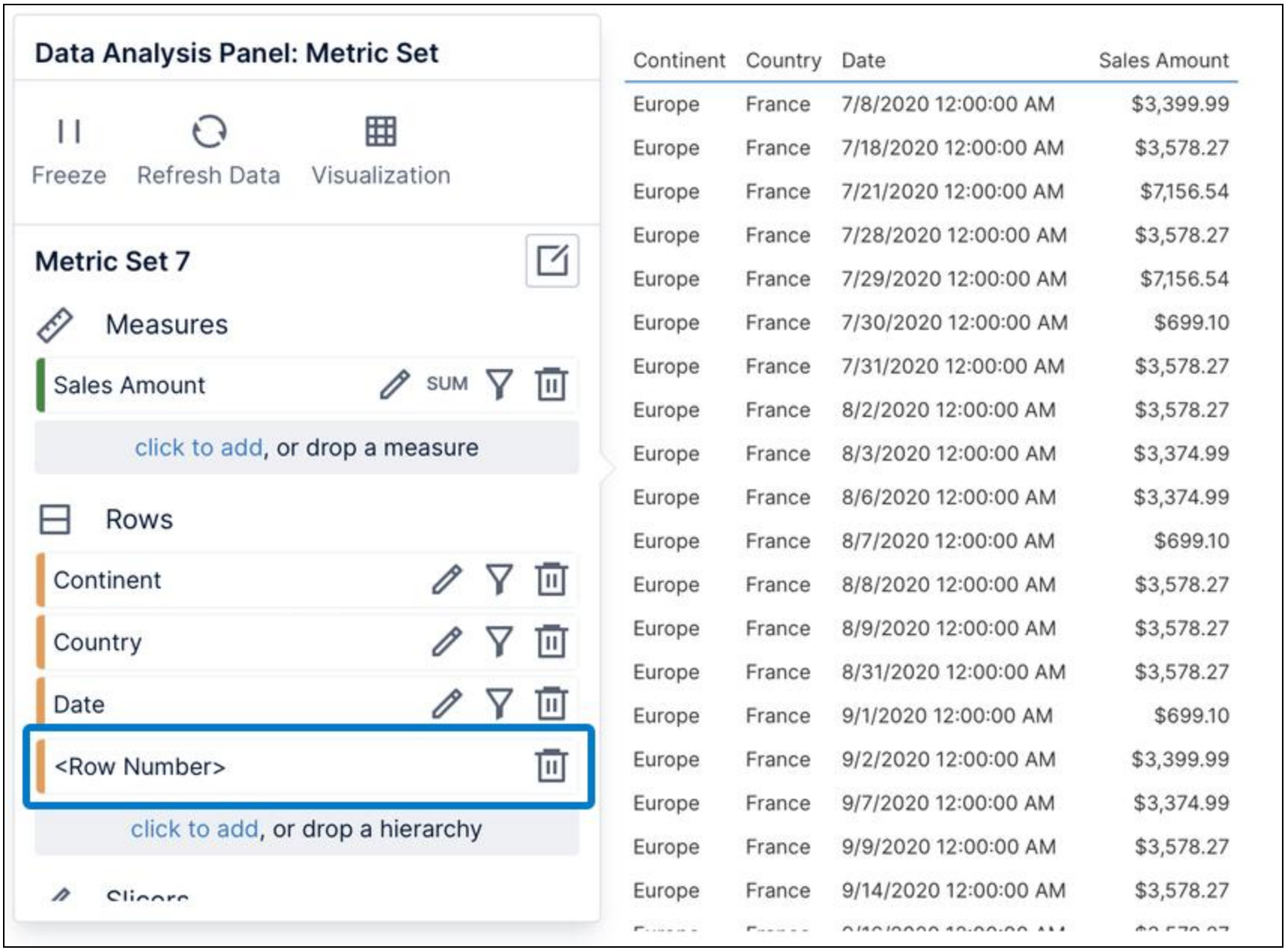
When dragging, you can drop data directly where you want it in the Data Analysis Panel, or onto the blue highlighted area of the visualization if you're not sure where to drop it. In a full screen metric set, you can also access the **Assistant** from the toolbar and type what you want to see.

By placing data on **Rows** or **Columns**, you are grouping by those values in the order they are placed. For example, place a time dimension hierarchy on Rows to see sales numbers grouped by year, even if the data source contains many rows for individual dates.



Note: You can drag data from another table or data source in the Explore window and it will be automatically joined. This uses database relationships for tables from the same database, or relationships for different data sources.

If you **click to add** the special option **<Row Number>**, the data will not be grouped and all original rows will be displayed unaggregated and unsorted as raw data. There are various limitations with raw data mode, but in return this allows you to view the underlying data unmodified. This option is also initially selected when you select or drag an entire table or spreadsheet as the data for your metric set.

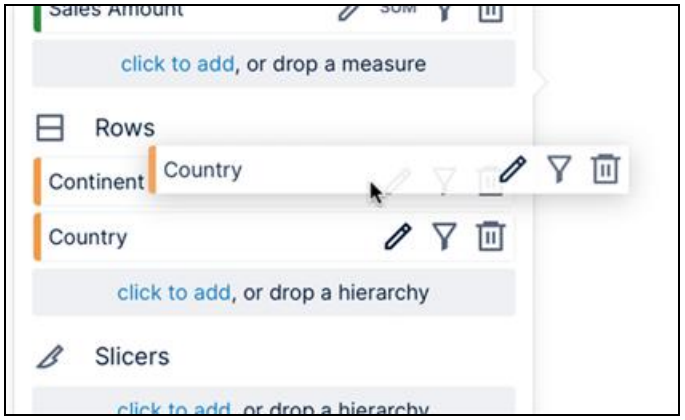

Data Analysis Panel: Metric Set
Freeze
Refresh Data
Visualization
Metric Set 7
Measures
Sales Amount
SUM
click to add, or drop a measure
Rows
Continent
Country
Date
<Row Number>
click to add, or drop a hierarchy
Slicers
Continent Country Date Sales Amount
Europe France 7/8/2020 12:00:00 AM $3,399.99
Europe France 7/18/2020 12:00:00 AM $3,578.27
Europe France 7/21/2020 12:00:00 AM $7,156.54
Europe France 7/28/2020 12:00:00 AM $3,578.27
Europe France 7/29/2020 12:00:00 AM $7,156.54
Europe France 7/30/2020 12:00:00 AM $699.10
Europe France 7/31/2020 12:00:00 AM $3,578.27
Europe France 8/2/2020 12:00:00 AM $3,578.27
Europe France 8/3/2020 12:00:00 AM $3,374.99
Europe France 8/6/2020 12:00:00 AM $3,374.99
Europe France 8/7/2020 12:00:00 AM $699.10
Europe France 8/8/2020 12:00:00 AM $3,578.27
Europe France 8/9/2020 12:00:00 AM $3,578.27
Europe France 8/31/2020 12:00:00 AM $3,578.27
Europe France 9/1/2020 12:00:00 AM $699.10
Europe France 9/2/2020 12:00:00 AM $3,399.99
Europe France 9/7/2020 12:00:00 AM $3,374.99
Europe France 9/9/2020 12:00:00 AM $3,578.27
Europe France 9/14/2020 12:00:00 AM $3,578.27

Move, Reorder, Remove Data for a Metric Sets

This applies to: *Managed Dashboards, Managed Reports*

To move data between Rows, Slicers, and Columns, drag it from where it is placed already in the Data Analysis Panel to the new area where the label *drop a hierarchy* appears.

To change the order of multiple items placed under the same heading, drag a lower item and drop it where you want it located.



The order of hierarchies determines their grouping order. The order of measures determines their sorting priority when sorting by more than one measure.

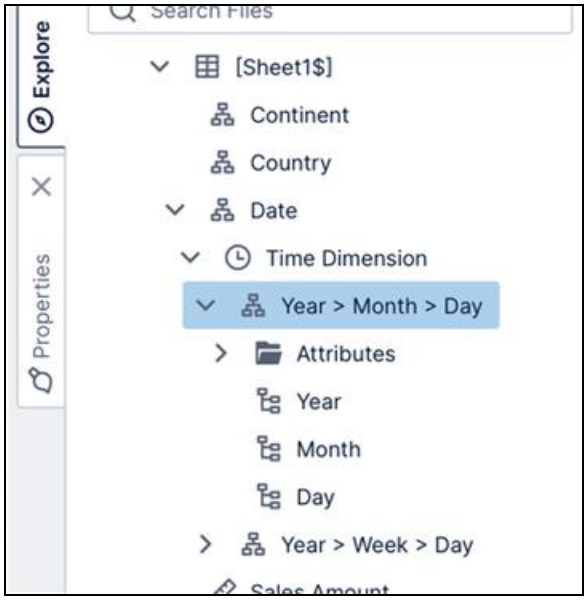
Click the delete button to the right of the hierarchy or measure to remove it from the metric set.

Replace Data in a Metric Set with a Hierarchy

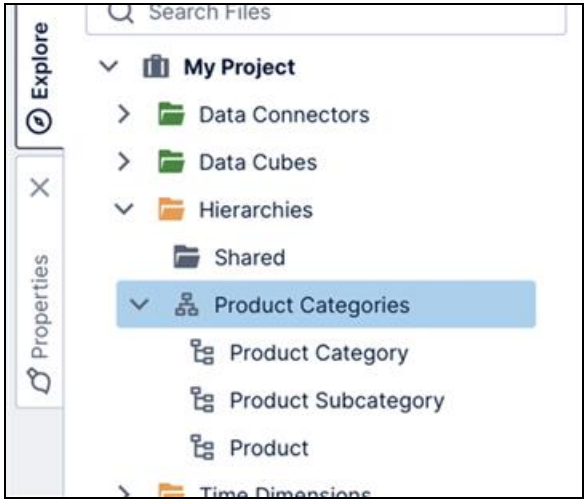
This applies to: *Managed Dashboards, Managed Reports*

If you are working with data directly from a database, spreadsheet, or another regular table of data that was not prepared as a cube, you can replace a regular column with a hierarchy from the Explore window.

- Date/time columns can be expanded in the Explore window to find time dimension hierarchies and attributes, which you can drag onto the Data Analysis Panel or your visualization to group by time periods such as Year, drill up & down, and more. You can expand a hierarchy and drag a specific level you want to group by, or change the level later.



- If you already added data related to a hierarchy that was created or if you dragged date/time data without a time dimension, drag the hierarchy from the Hierarchies or Time Dimensions folder on top of it in the Data Analysis Panel. You can also drop a simple column displaying friendlier names on top of a column of IDs or keys. See Automatic Joins and Hierarchies.


Search Files
Explore
Properties
My Project
Data Connectors
Data Cubes
Hierarchies
Shared
Product Categories
Product Category
Product Subcategory
Product

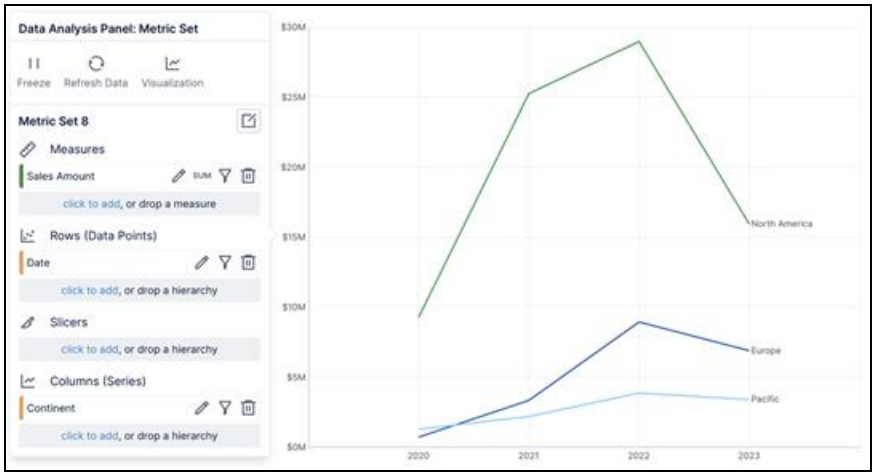
Visualizations for Symphony Metric Sets

This applies to: *Managed Dashboards, Managed Reports*

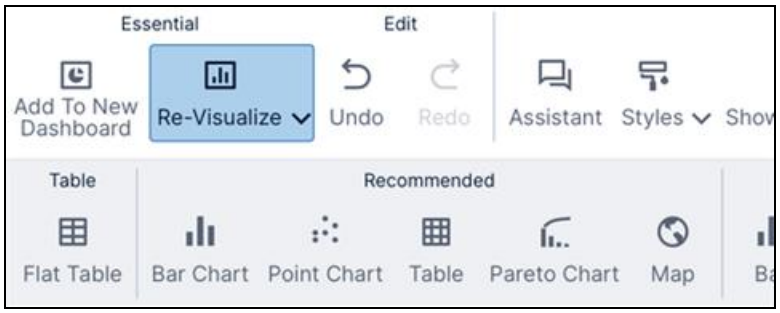
A metric set's data is always shown in a visualization that you can change and customize. When you edit a metric set full screen, this visualization is the default visualization used when the metric set is first added to a dashboard or other view, but you can also use it just for your own analysis.

Note: The metric set defines all of the data that's displayed and its settings, such as initial filtering and sorting. Each visualization is separate so it can be changed and customized when reusing a metric set.

If you didn't choose a particular visualization yet or customize it, Symphony will automatically change the visualization when you first add data based on its recommendations. For example, a measure and a time dimension hierarchy will be automatically visualized as a line chart, which is best for seeing the trending of a measure over time.

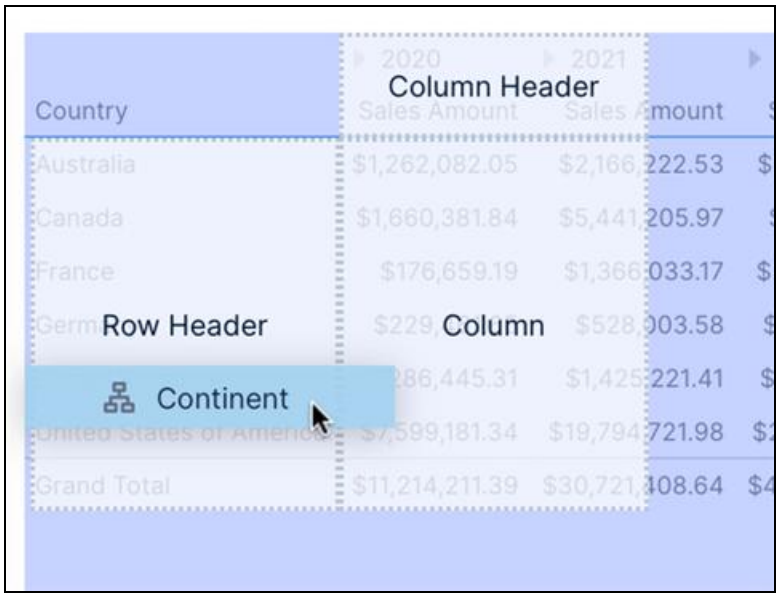


At any time, you can change to the visualization of your choice by clicking **Re-Visualize** in the toolbar and choosing an option. Once you have chosen a visualization yourself, customized it, or saved and re-opened this metric set, it will not be changed automatically.

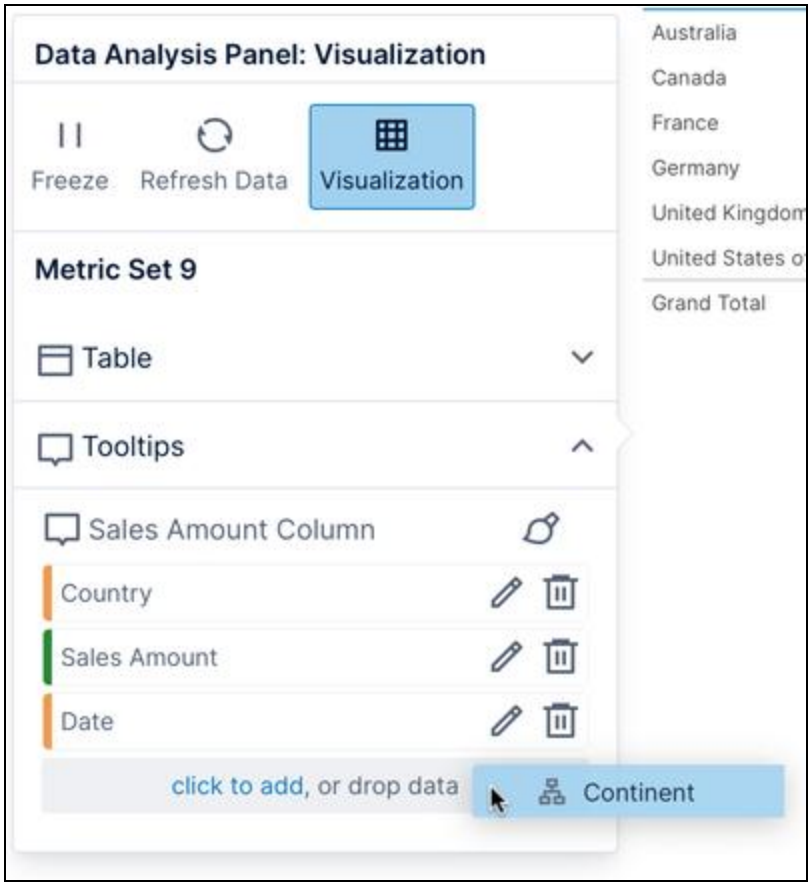


You can customize a particular visualization in a number of ways:

- Drop zones may appear over your visualization when dragging data from the Explore window, allowing you to choose how it will be visualized. For example, you can decide whether to display a hierarchy as a regular column, a row header column, or as a row of column headers in a table.

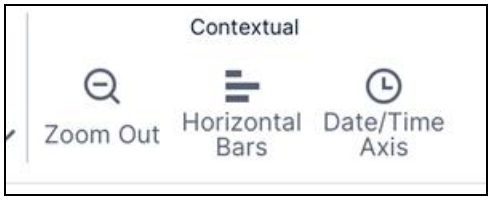


- Similarly, you can click the **Visualization** button in the Data Analysis Panel of any visualization and drag data from the **Explore** window or **click to add** data directly under a visualization option, such as **Tooltip** for displaying additional details in a popup when hovered or long-tapped.



See Setting Up the Visualization and Visualization Tab Examples.

- Common visualization options can be found in the toolbar under *Contextual* section, and in the right-click context menu, such as changing a vertical bar (column) chart to a horizontal bar chart.



- The full set of visualization settings are found in the Properties window.

When a metric set is added to a dashboard or another view, the default visualization from the full screen editor is used as the template for a new visualization. The metric set is reusable and each visualization is separate, so you and other people you share the metric set with can customize it and visualize it differently.

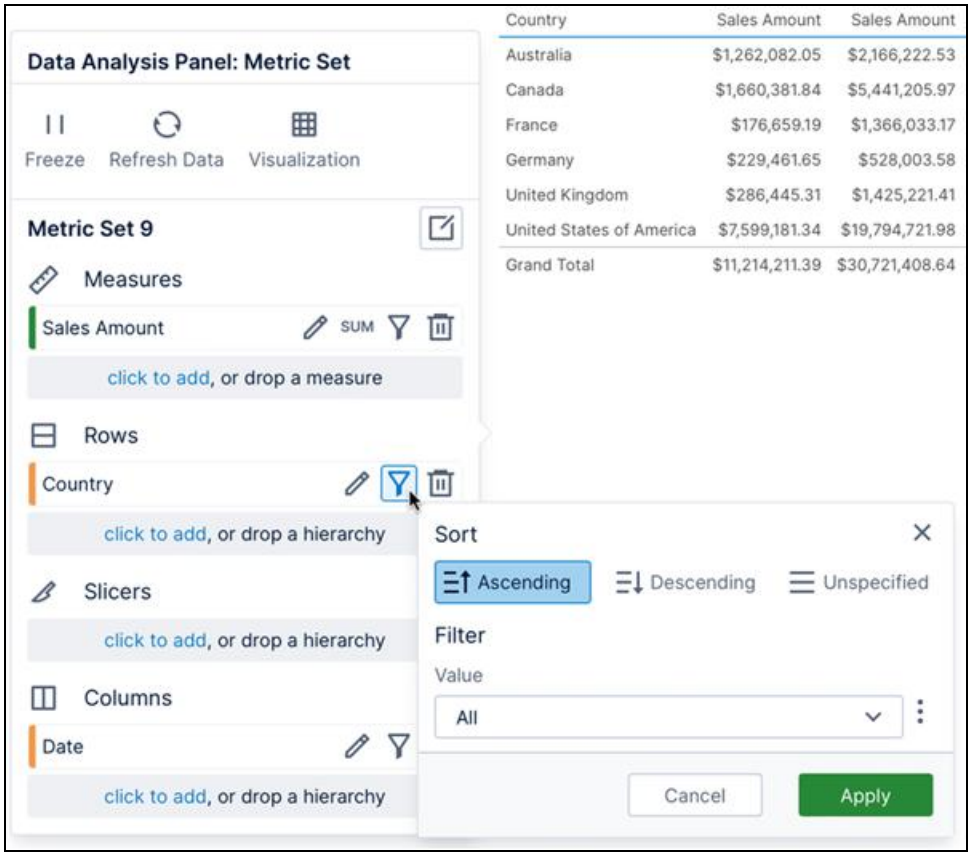
Note: If you want to reuse a visualization in multiple places and be able to make changes to all of them at once, you can create a dashboard for that visualization and drag it onto other dashboards and views.

Sort and Filter Metric Data

This applies to: *Managed Dashboards, Managed Reports*

To quickly filter or sort data while editing, use the sort & filter icon shown for the values you want to sort by or filter.

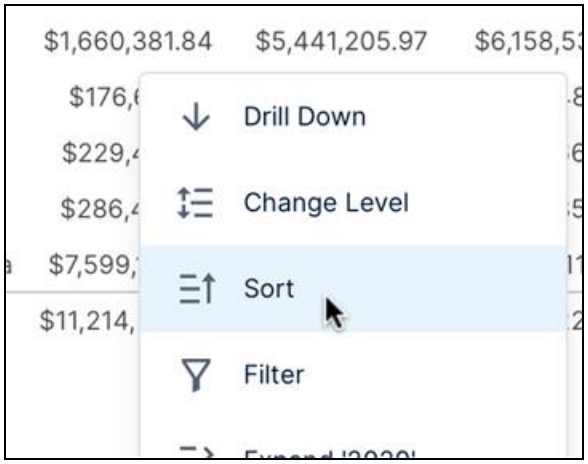
For example, you can filter the range of dates shown on a chart, or sort a table's rows by a measure.



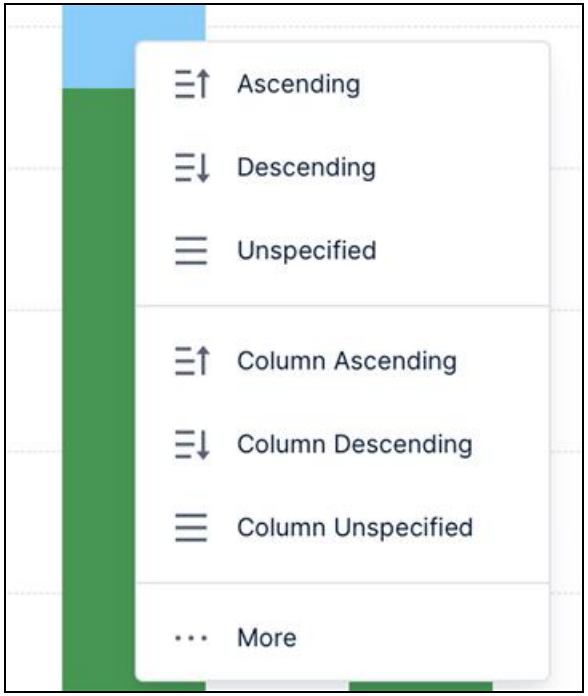
Note: For measures, this quick sorting option reorders the rows of data by its values. If you want to reorder the columns by this measure instead, use the right-click sorting options or set up custom measure sorting.

This is the same pop-up shown when sorting a table visualization from one of its column headers, and it can contain options such as Sort On a Measure and a Hierarchy.

You can also right-click (or long-tap) data directly in the visualization and access sorting and filtering options from the context menu.

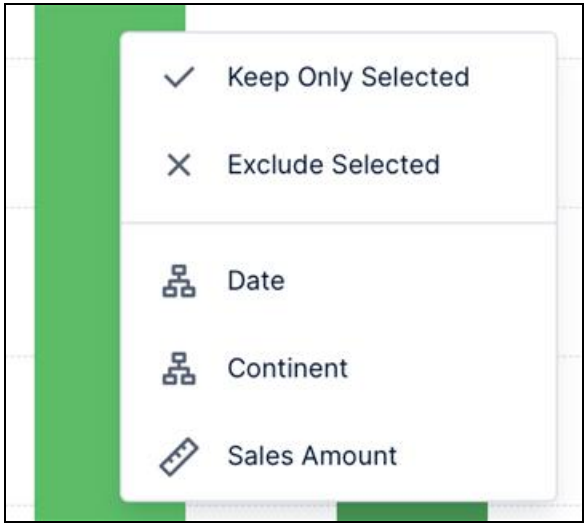


If you choose Sort, a list of measures or hierarchies may appear if there is more than one related to what you clicked. Choose what values should be used to sort the rows (or columns), then choose the sort order. For an example of sorting using the context menu, see Sorting a Table Visualization.



If there are multiple columns (or series) of data that can be sorted, additional options appear allowing you to sort the **Columns Ascending** or **Columns Descending** instead of sorting the rows. The **More...** option brings up the same sort & filter popup shown above with options to change how the data is grouped.

If you choose **Filter**, you can choose to keep only or exclude the data you right-clicked on. This can apply to multiple rows, columns, or data points at once if you clicked or dragged to select multiple. You can also choose the name of a hierarchy listed to choose some of its values to filter by, or choose **Clear All** to clear any previous filtering (if applicable).



For an example of excluding selected data, see Exclude Data Points from the Visualization.

These context menu options are available even when you or others are viewing this metric set after it's been added to a dashboard and shared. By sorting and filtering with the metric set checked out to you, you can determine the initial settings when it's viewed the first time.

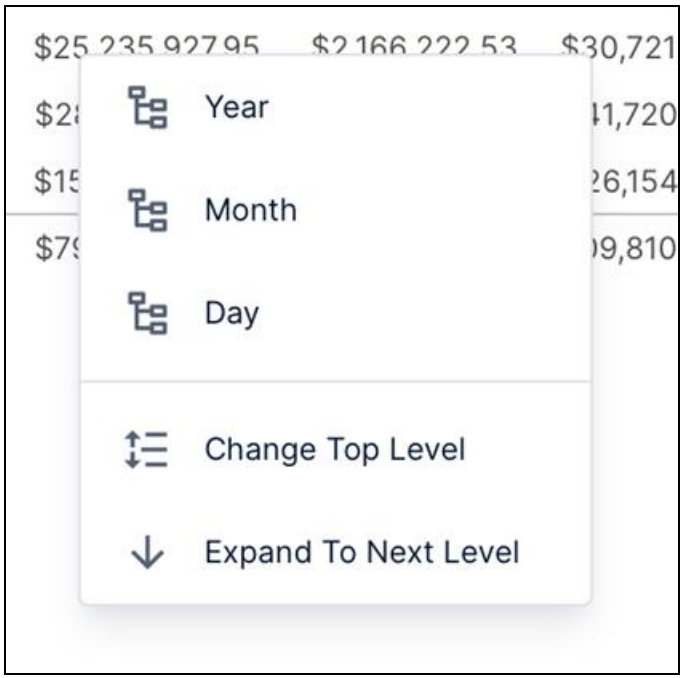
More advanced options are available from the Data Analysis Panel as described in Define Custom Hierarchy Sorting, Define Custom Measure Sorting, and Displaying Top/Bottom Records.

Change Levels in a Hierarchy for a Metric Set

This applies to: *Managed Dashboards, Managed Reports*

When displaying data from a multi-level hierarchy, you can change which levels are displayed, or drill down to see the values in the level below that make up a value.

To quickly change the level, right-click (or long-tap) anywhere on the visualization and choose **Change Level**. (If there are multiple hierarchies to choose from, choose one from the list that appears next.)



Choose a level to change the level for the entire current set of data.

Many visualizations can display multiple levels at once, including tables and relationship diagrams. To select a range of levels instead of just one, choose **Change Top Level** and then a level, or **Expand To Next Level** to display both the current level and the next level down. When the main level is different than the top level, both levels and the levels in between will be displayed at once.

Year	Month	Sales Amount	Order Count
2020	February, 2020	$43,188.76	6
	July, 2020	$838,818.77	67
	August, 2020	$2,056,504.73	86
	September, 2020	$1,647,447.20	87
	October, 2020	$1,354,040.40	87
	November, 2020	$2,856,790.80	86
	December, 2020	$2,417,420.73	106
		$11,214,211.39	525
2021	January, 2021	$1,356,959.36	97
	February, 2021	$2,405,144.59	93
	March, 2021	$2,126,739.08	110
	April, 2021	$1,504,003.07	96
	May, 2021	$3,007,741.85	111
	June, 2021	$1,631,307.05	99
	July, 2021	$2,946,753.18	122
	August, 2021	$4,124,133.13	128
2021		$30,721,408.64	1,322
Grand Total		$109,810,538.20	4,977

Since these are metric set settings, you can also set the **Level** and **Top Level** from the Data Analysis Panel by clicking to edit the hierarchy and expanding the **Parameter Values** section. You can also drag hierarchy levels onto your visualization from the Explore window as a shortcut.

Right-click (or long-tap) data in a visualization and choose **Drill Down** to filter to that value and change the level of the hierarchy down. (If there are multiple hierarchies to choose from, choose one from the list that appears next.)

For example, drill down on 2022 to see the individual monthly values that make up that year.

Month	Sales Amount	Order Count
January, 2022	$1,787,049.85	133
February, 2022	$2,855,152.20	118
March, 2022	$2,041,999.94	130
April, 2022	$2,354,527.85	122
May, 2022	$3,432,276.50	137
June, 2022	$2,551,292.16	128
July, 2022	$3,478,195.17	149
August, 2022	$5,060,295.96	179
September, 2022	$5,062,564.44	180
October, 2022	$3,322,180.66	186
November, 2022	$4,597,208.34	179
December, 2022	$5,177,485.89	186
Grand Total	$41,720,228.96	1,827

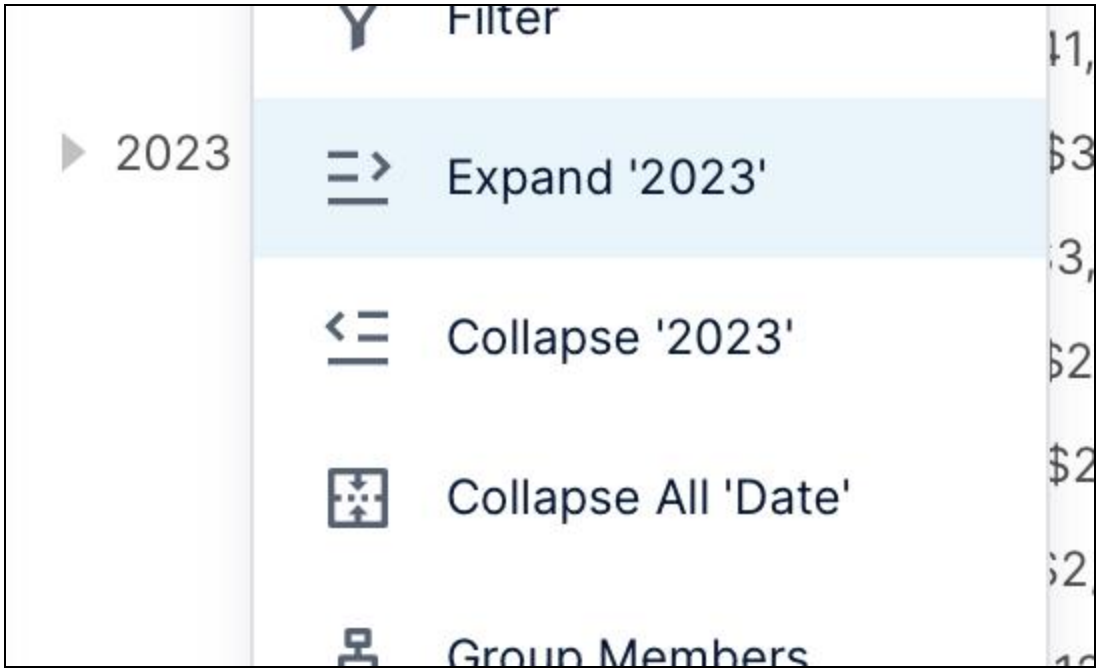
To change the level back up and clear the filtering, choose **Drill Up** in the context menu.

Like sorting and filtering, these context menu options are available even when you or others are viewing this metric set after it's been added to a dashboard and shared. Changing the level and drilling down while the metric set is checked out to you determines the initial settings when it's viewed the first time.

Expand and Collapse Metric Set Data

This applies to: *Managed Dashboards, Managed Reports*

When multiple hierarchies are added under Rows or Columns in the Data Analysis Panel, or if a multi-level hierarchy is added, you can right-click (or long-tap) values and choose **Expand** or **Collapse**. Some visualizations may also offer triangular expander buttons.



Expanding is similar to drilling down, except the other data remains displayed as before, with the values added from the level below or from the next hierarchy just for the expanded value.

Year	Month	Sales Amount	Order Count
▶ 2020		$11,214,211.39	525
▶ 2021		$30,721,408.64	1,322
▶ 2022		$41,720,228.96	1,827
◢ 2023	▶ January, 2023	$3,085,082.68	186
	▶ February, 2023	$4,101,619.01	168
	▶ March, 2023	$4,219,750.75	186
	▶ April, 2023	$3,716,892.31	178
	▶ May, 2023	$5,204,229.12	185
	▶ June, 2023	$5,317,581.78	180
	▶ July, 2023	$497,145.10	183
	▶ August, 2023	$12,388.45	37
		$26,154,689.22	1,303
Grand Total		$109,810,538.20	4,977

Collapsing removes the values below just one value, leaving the rest of the data in place.

Continent	Year	Sales Amount	Order Count
Europe	2020	$692,566.16	185
	2021	$3,319,258.15	520
	2022	$8,919,008.16	896
	2023	$6,870,865.60	646
	(All)	$19,801,698.06	2,247
North America	(All)	$79,353,404.18	1,672
Pacific	2020	$1,262,082.05	157
	2021	$2,166,222.53	321
	2022	$3,849,580.02	361
	2023	$3,377,551.36	219
	(All)	$10,655,435.96	1,058
Grand Total		$109,810,538.20	4,977

Use the **Collapse All** option to collapse all of the values of any hierarchy into a single total.

For example, you can collapse all individual years into a summary grouped by the remaining values.

Continent	Year	Sales Amount	Order Count
Europe	(All)	$19,801,698.06	2,247
North America	(All)	$79,353,404.18	1,672
Pacific	(All)	$10,655,435.96	1,058
Grand Total		$109,810,538.20	4,977

Notice above how this is similar to collapsing all the individual values in the leftmost column.

Use **Expand All** to reverse this action.

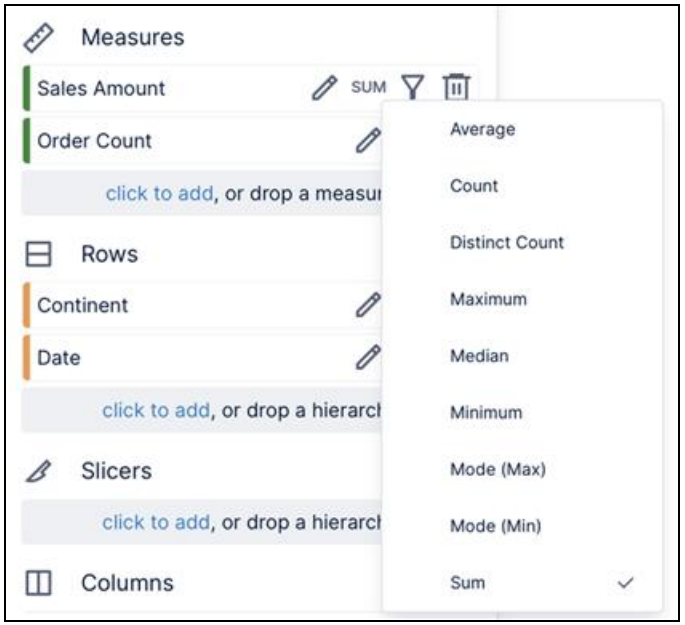
Note: If you want to expand all of a hierarchy level's values at once to the next level of that hierarchy, e.g., expand all the years down to months, you can use the options under Change Level for this as shown in Change Levels in a Hierarchy for a Metric Set.

Unlike expanding and collapsing individual values, **Collapse All** is a metric set setting that is saved, and can also be found in the dialog for each hierarchy in the Data Analysis Panel by clicking its orange tile to edit it. For more details and examples of expanding and collapsing, see Expand and Collapse Hierarchy Members.

Measure Options for a Metric Set

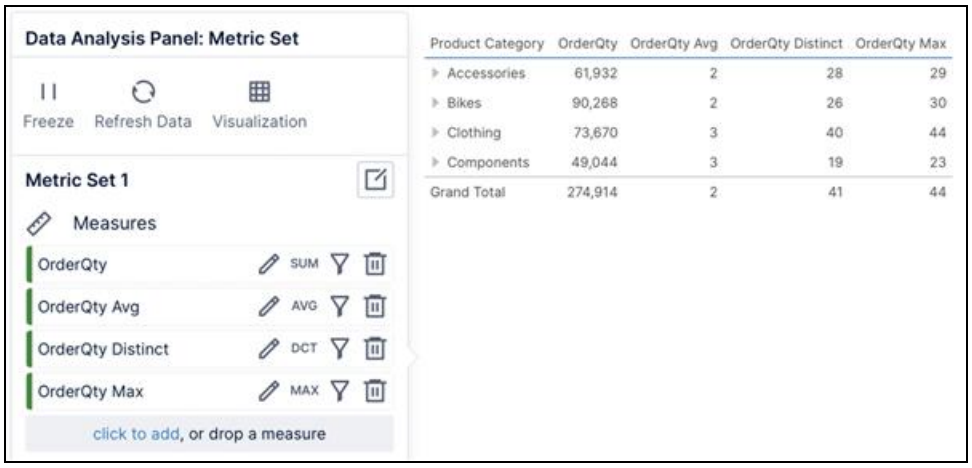
This applies to: *Managed Dashboards, Managed Reports*

When numeric data is first added under Measures in the Data Analysis Panel, its values are grouped and normally summed, as indicated by the displayed text *SUM*.

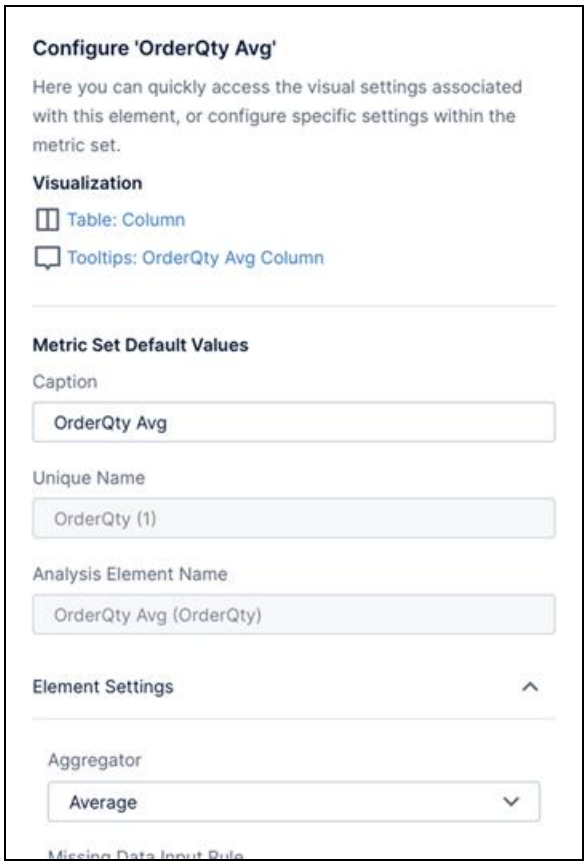


To quickly change this aggregator to another one, click this indicator and choose another from the list. For example, you can see the average order quantity for each year instead of the sum.

You can add a measure multiple times: drag the same data from **Explore** or **click to add** it again under **Measures**. Each copy of the measure can be set to a different aggregator (you should edit their captions to distinguish between them).



To access the full set of measure options, click its green tile or pencil icon in the Data Analysis Panel to edit it.



The dialog includes settings such as:

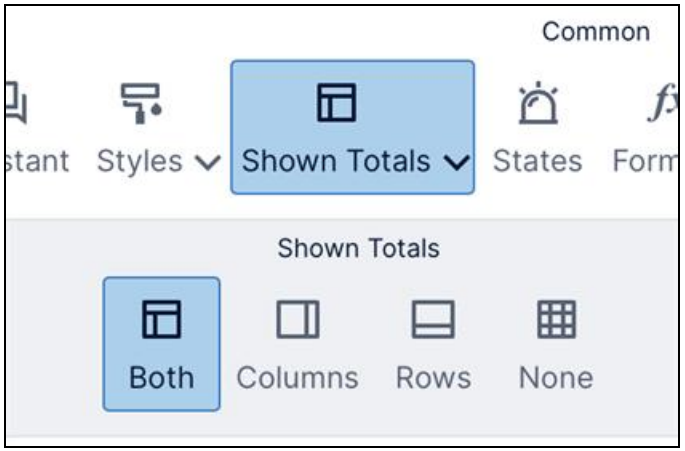
- **Caption**: Determines how this measure is identified in visualizations.
- **Format**: Determines how numeric data is formatted and displayed as text. See Formatting Text.
- **Missing Data Rules**: These settings determine if rows or columns of data should still be displayed if there is no corresponding measure value available, and whether to fill them in with replacement values. See Handling Missing Data for details.
- **Hidden**: If you don't want to display the measure but only use it for a formula, sorting, or other metric set features, check this box to prevent the measure's values from being seen directly or exported.

Changing settings such as Caption and Format here take effect everywhere at once as long as you haven't customized a visualization setting on top of them. For example, changing the caption affects column headers and tooltips, and changing the format affects the cells of a table, chart labels, and tooltips. They will also take effect if you re-visualize the metric set.

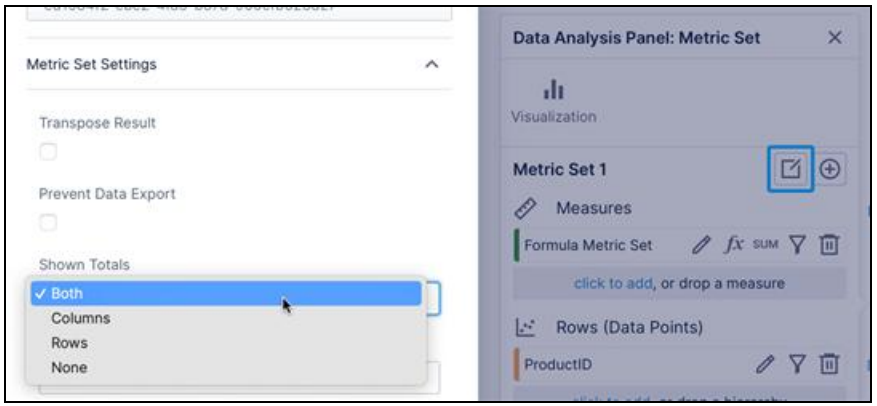
Totals for Metric Sets

This applies to: *Managed Dashboards, Managed Reports*

Tables and some other visualizations have support for displaying totals. To quickly turn all totals on or off when they are available, choose **Shown Totals** in the toolbar when editing a full screen metric set and choose which totals you want to show.

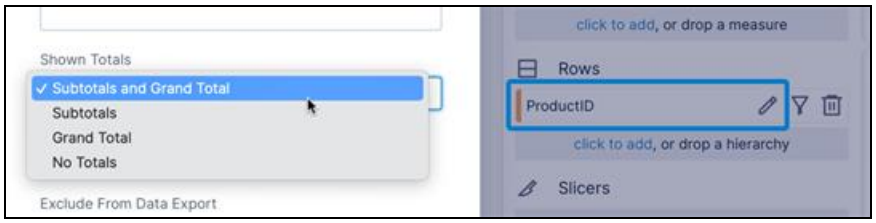


You can also access this setting from the Data Analysis Panel by clicking the metric set's **Edit** icon.



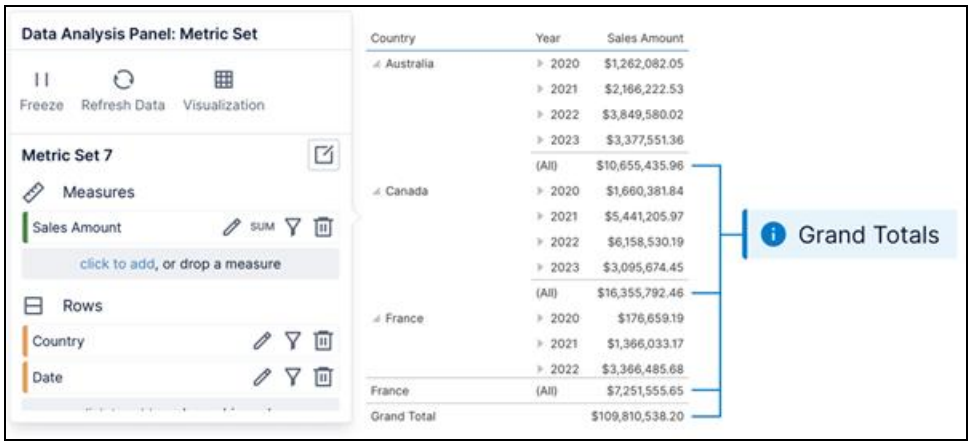
Note: Charts are typically disrupted by totals, so they are hidden by default by the property **Hide Total Values** in the Properties window for most chart types. This can be un-checked to Show or Hide Total Values On a Chart.

There are also separate settings available on each hierarchy, which you can access by clicking its orange tile in the Data Analysis Panel.



When changing the setting for a hierarchy, you can choose whether to include subtotals, the grand total, or both.

The main total for a hierarchy is its *Grand Total*. The figure below indicates the grand total for all countries plus each grand total for all years.



A hierarchy's *Subtotals* are displayed for its upper levels when displaying more than one or when it has been expanded to a lower level. The figure below indicates the subtotal for 2011 above months, and the subtotal for June, 2011 above its days because it has been expanded.


2023
January, 2023 $442,478.40
February, 2023 $469,983.90
March, 2023 $610,301.02
April, 2023 $460,055.75
May, 2023 $567,839.45
June, 2023 $707,105.01
July, 2023 $117,713.83
August, 2023
Aug 01, 2023 $227.39
Aug 02, 2023 $305.89
Aug 03, 2023 $300.90
Aug 04, 2023 $274.98
Aug 05, 2023 $95.47
Aug 06, 2023 $324.98
Aug 07, 2023 $544.39
$2,074.00
$3,377,551.36
(All) $10,655,435.96
Subtotals

Data Summaries for Metric Sets

This applies to: *Managed Dashboards, Managed Reports*

When selecting rows or columns in a table, or data points in a chart, some basic statistics of that data will appear in the status bar: sum (SUM), average (AVG), minimum (MIN), maximum (MAX), the number of points (COUNT), and number of distinct values (DISTINCT).

If there are multiple measures, it will cycle through the statistics for each measure plus the statistics for all measure values mixed together. Hover over the data summary for a tooltip showing all of the statistics at once.

Note: Data summary is not available for waterfall and stacked chart types.

You can click and drag your mouse to select multiple points in a chart. Hold the Ctrl or Shift keys while doing this to select additional areas of data points. To select multiple rows or columns in a table, hold Ctrl while clicking them, or hold Shift to select a group of rows or columns in between the ones you click.

Other Metric Set Tools

This applies to: *Managed Dashboards, Managed Reports*

Metric sets include many more tools to help with your analysis, which are detailed in separate articles.

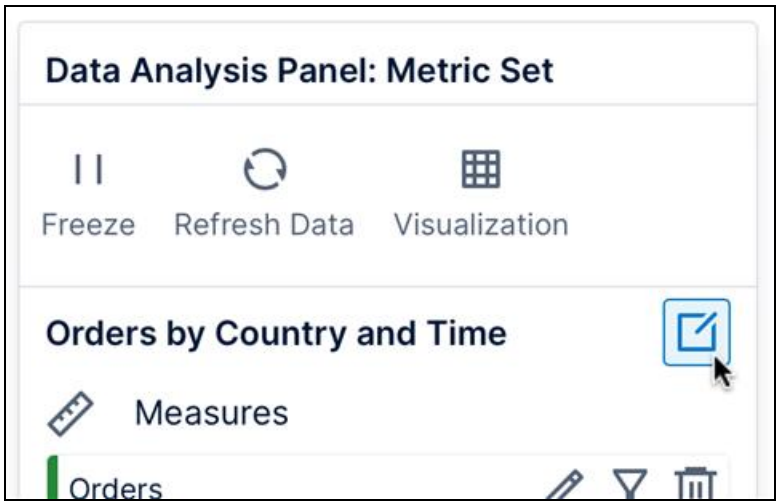
In the toolbar, you can find:

- States for calling attention to some data, or displaying it differently based on conditions.
- Period over period or slicer comparisons for viewing data from different time periods or filter values, or using it for state conditions or formulas.
- Formulas for adding a new measure or an entirely new metric set based on a formula or script.
- Contextual measures for inputting values to compare against your data, such as goals or targets.

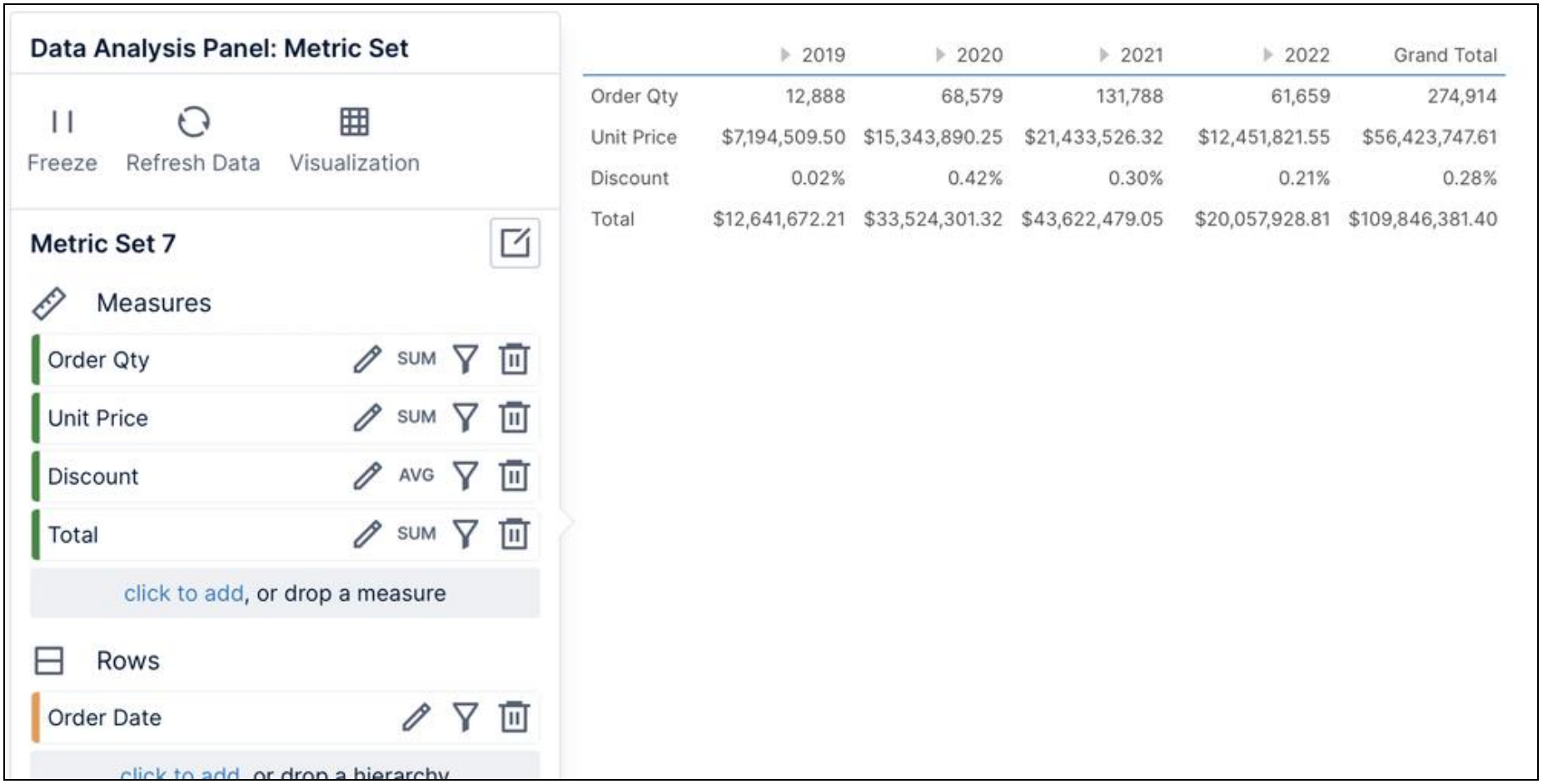
The context menu provides you with additional options when you right-click or long-tap data:

- Group members: for grouping together the items of your choice into a single item.
- Measure corrections/data annotations: for correcting or updating data for public viewing, or for what-if scenarios for your own analysis.

The metric set has overall settings you can access by clicking the **Edit** icon in the Data Analysis Panel.



- **Transpose Result**: A transposed metric set swaps the rows with the columns in the result, including displaying a different measure in each row instead of a separate column for each measure. This can be used to display a single column or chart series comparing different measure values, whereas each measure is normally a separate column or series.



- Top/Bottom Items and Other Group: you can filter the metric to display only the top or bottom 'N' items of some number, or group together the smallest items into a single item (or use other criteria).

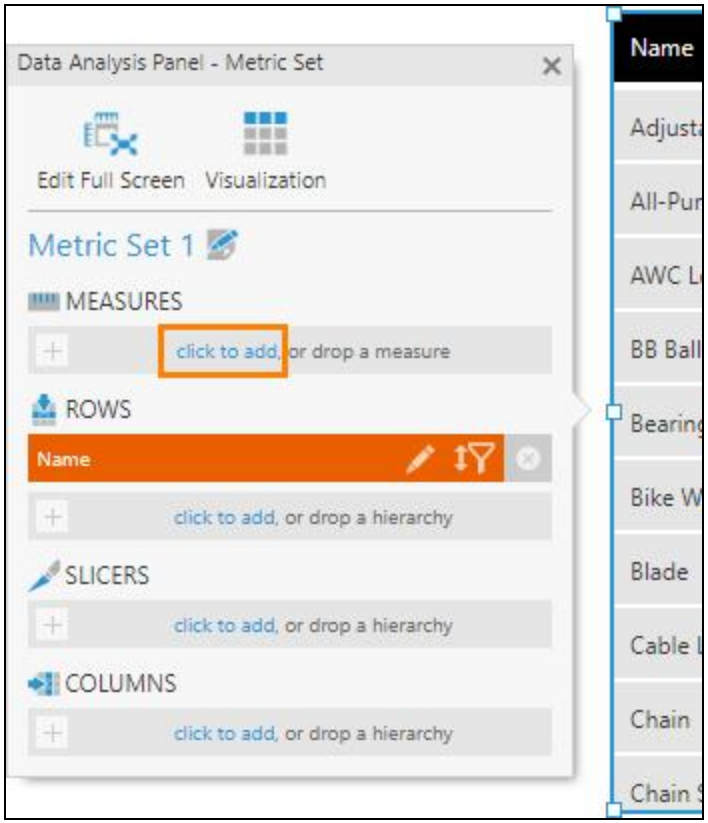
Add a Count Measure to a Metric Set

Add a count measure to display the number of records or values behind each row or column.

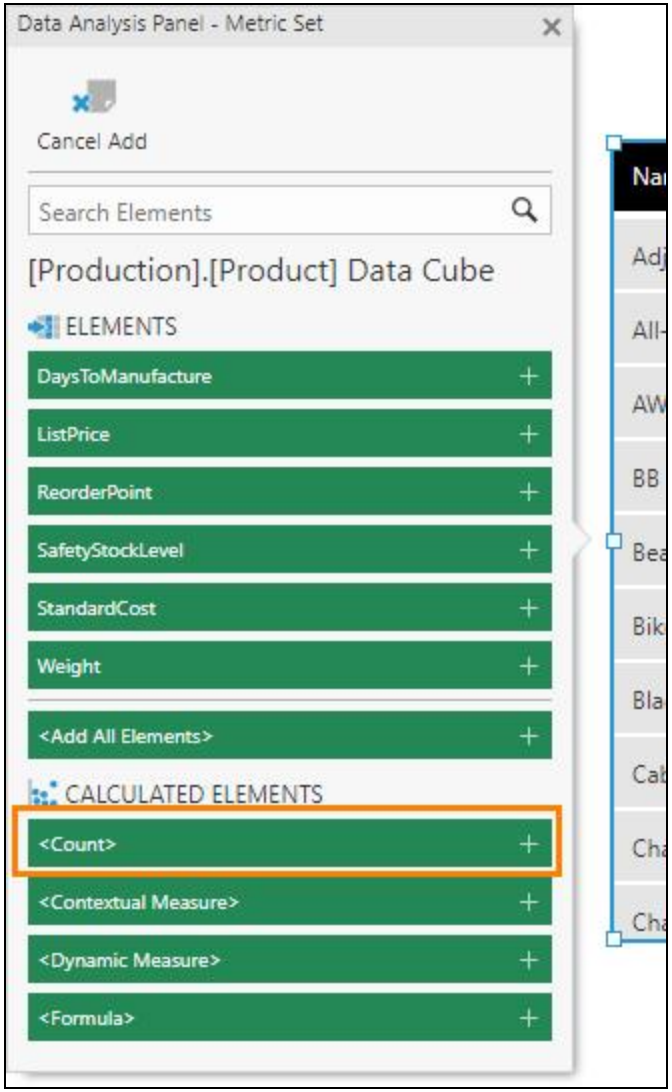
Adding a Count Measure

A count measure displays the number of records making up each row or column. This can be added to any metric set except those displaying data from an OLAP cube.

Under Measures in the Data Analysis Panel, click to add a measure.

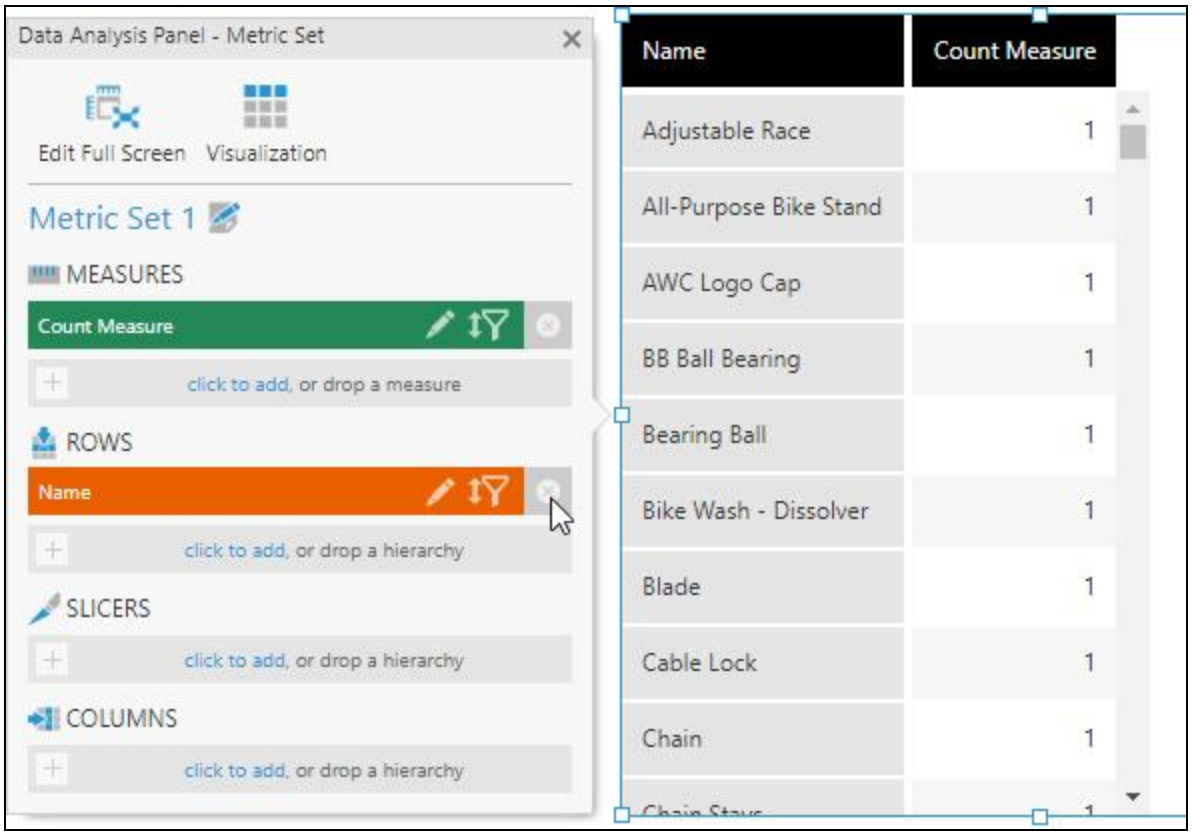


In the list of available measures you can add, scroll down to the Calculated Elements section and click <Count>.

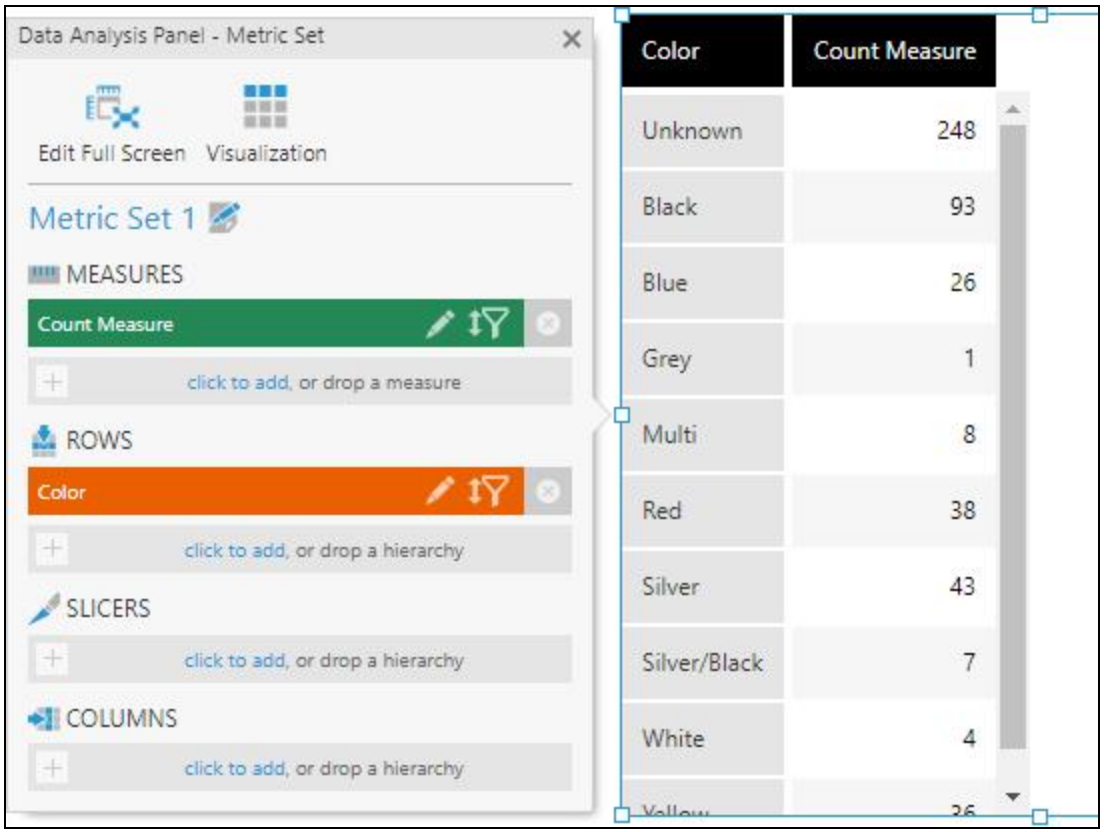


A Count measure is added to the Data Analysis Panel and your visualization.

In our example below, all of the original product records are displayed so the count is 1 in every row, but this would change if the data is grouped differently by removing Name or replacing it with a different hierarchy.



The following example shows the number of product table records for each Color:

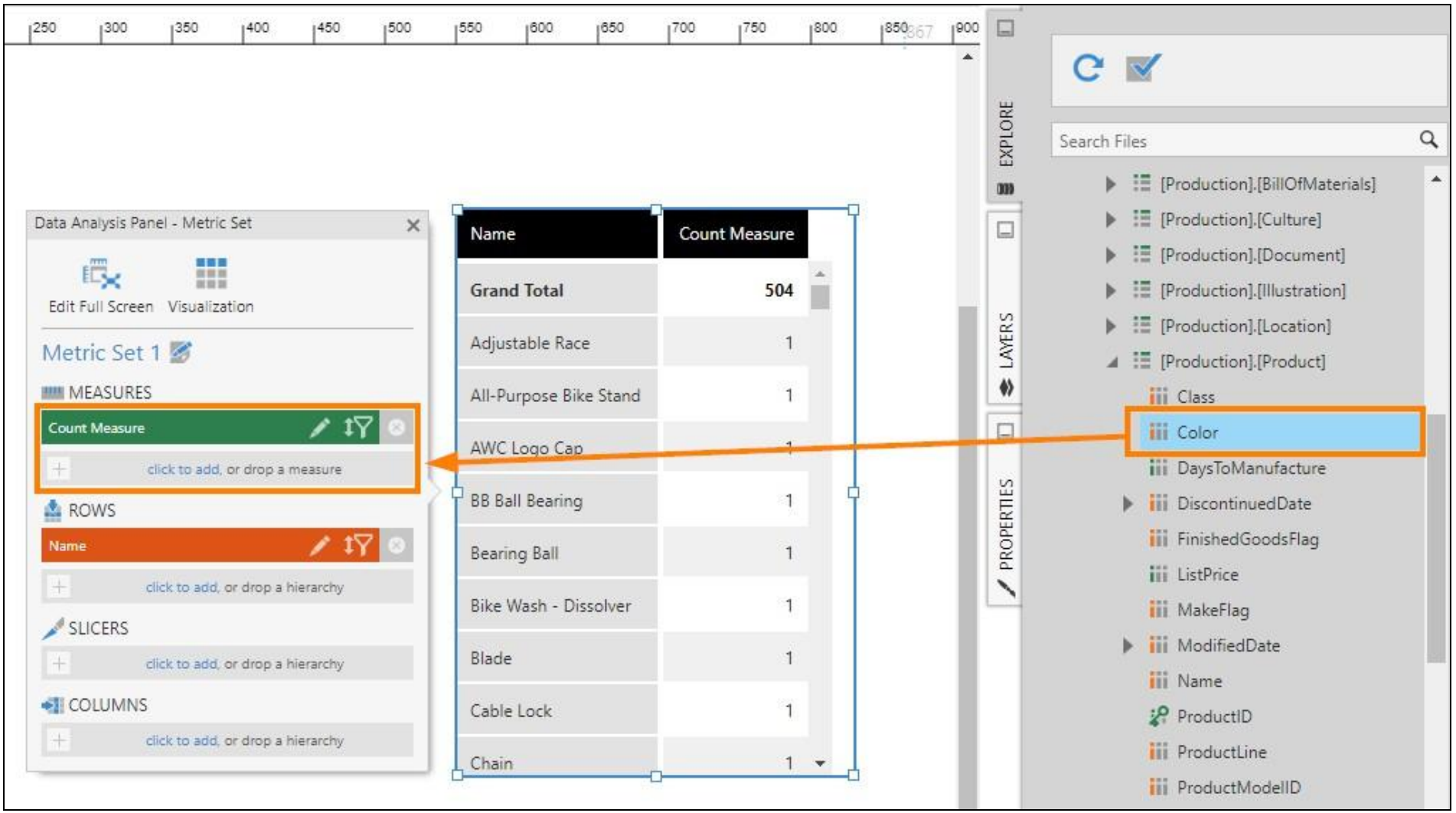


Note: You can also edit or re-add a measure (the same measure can be added multiple times) and change its aggregator to Count, although this excludes records with a value missing or null for that measure.

Add a Hierarchy Count Measure

A hierarchy count measure displays either the total number of values (Count) or the number of distinct values (Distinct Count) present for a particular hierarchy or non-numeric column.

To add a hierarchy count measure, drag a hierarchy or a column of non-numeric data from the Explore window to Measures in the Data Analysis Panel.

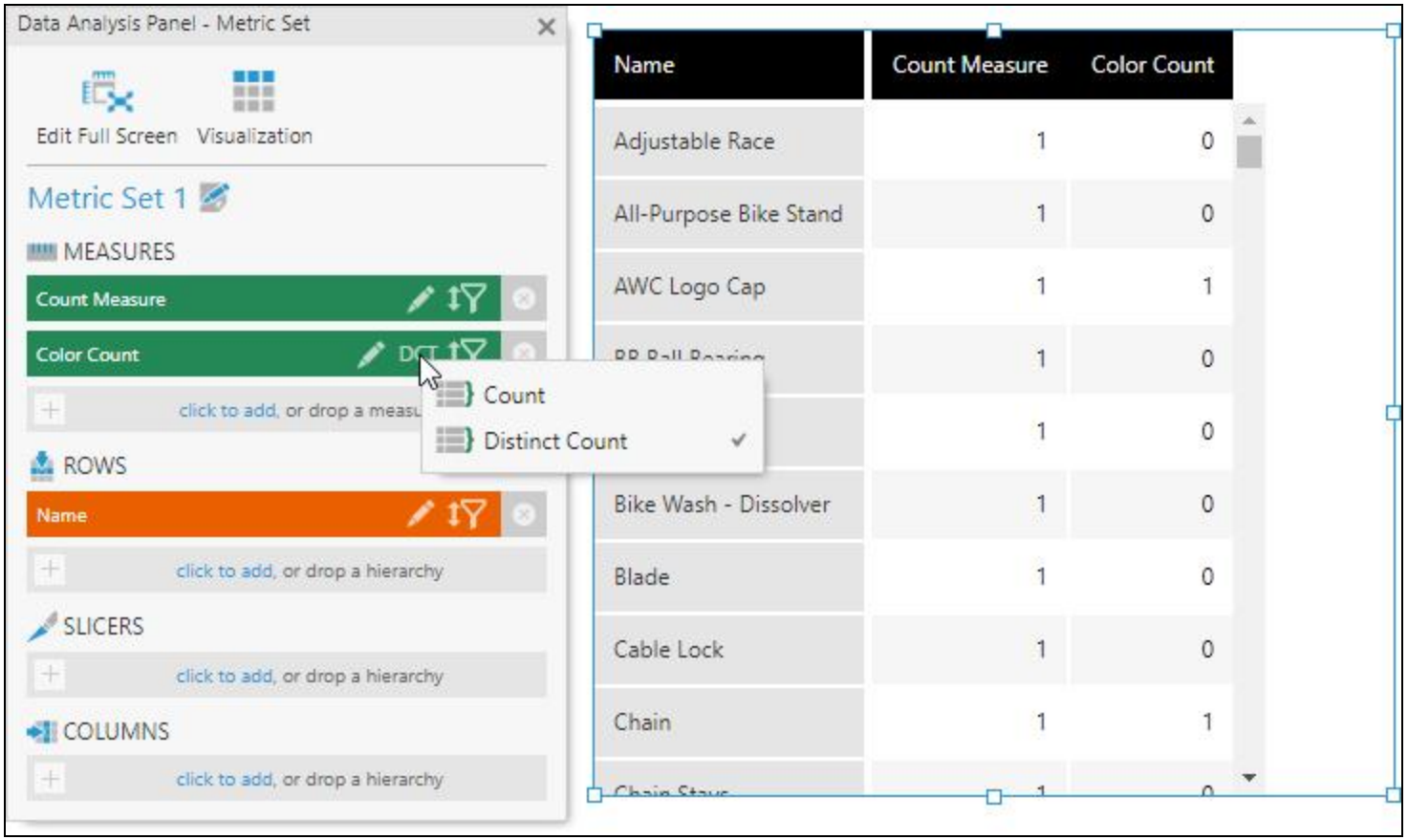


Note: You can create a similar count measure for this data in a data cube's Process Result by switching it to a measure.

A count measure will be automatically created and named based on the name of the hierarchy, e.g., Color Count.

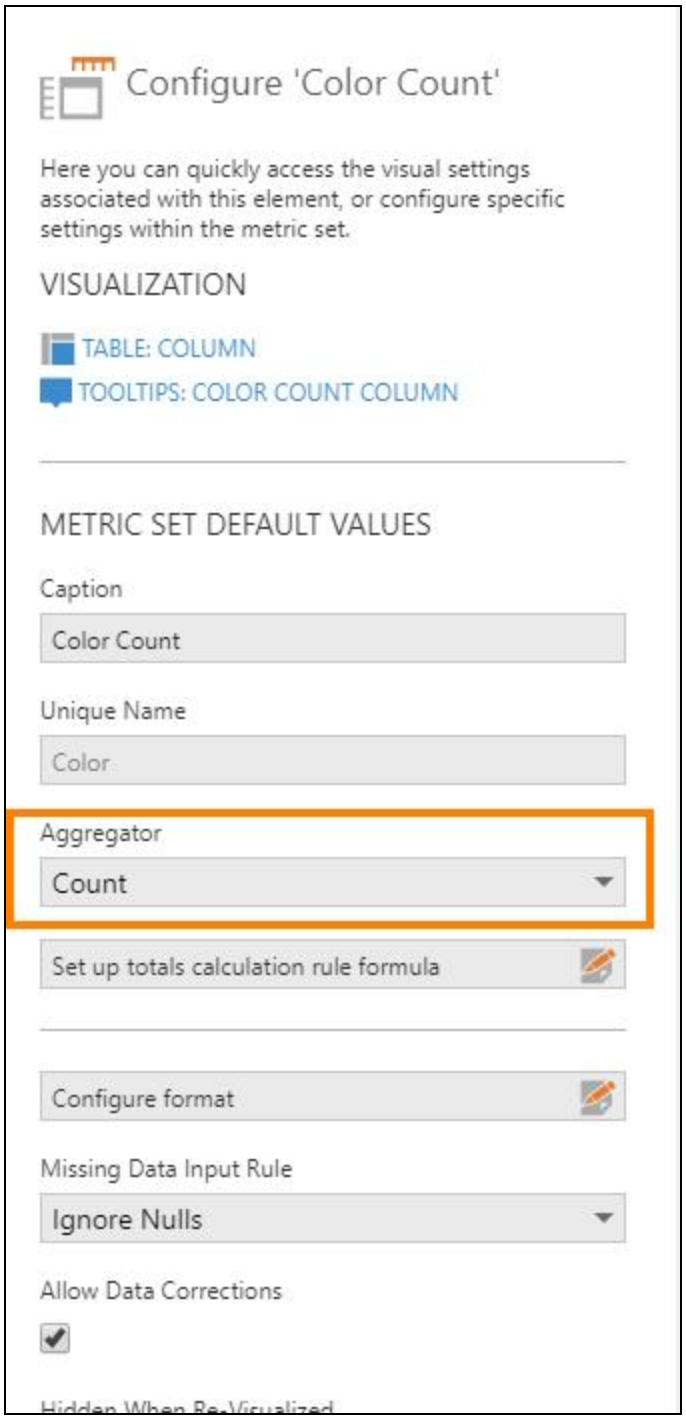
By default, the aggregator for a hierarchy count is set to Distinct Count, in this example displaying the number of distinct colors available for each product (sometimes none or null).

To quickly change the aggregator, click the abbreviated aggregator name shown for the measure in the Data Analysis Panel (e.g., DCT) and choose a different one from the popup.



Note: OLAP hierarchies support only Distinct Count and not the Count aggregator. A data cube's Process Result settings may also limit the available aggregators.

Alternatively, click the Edit icon for the measure to open its full configuration dialog, and change the Aggregator.


Configure 'Color Count'
Here you can quickly access the visual settings associated with this element, or configure specific settings within the metric set.
VISUALIZATION
TABLE: COLUMN
TOOLTIPS: COLOR COUNT COLUMN
METRIC SET DEFAULT VALUES
Caption
Color Count
Unique Name
Color
Aggregator
Count
Set up totals calculation rule formula
Configure format
Missing Data Input Rule
Ignore Nulls
Allow Data Corrections

You can add multiple hierarchy count measures for the same hierarchy in version 10 and higher if you want to use multiple aggregators at once.

Automatic count measures are not supported when displaying data from an OLAP cube. The hierarchy count measures described above are supported, as well as count measures created in the OLAP cube.

The available aggregators for both measures and hierarchy count measures can be changed in the data cube.

Totals are not retrieved from OLAP when using hierarchy count measures together with top/bottom settings.

For more information, see:

- Other Metric Set Tools
- Create a Metric Set and Add a Filter
- The Managed Dashboards Data Model
- Formatting Text

Add a Period Over Period Comparison

This walkthrough shows you how to add a period over period (or date offset) comparison.

Period over period lets you compare measure values for one period of time against the results from a previous or subsequent period of time, for example year-over-year.

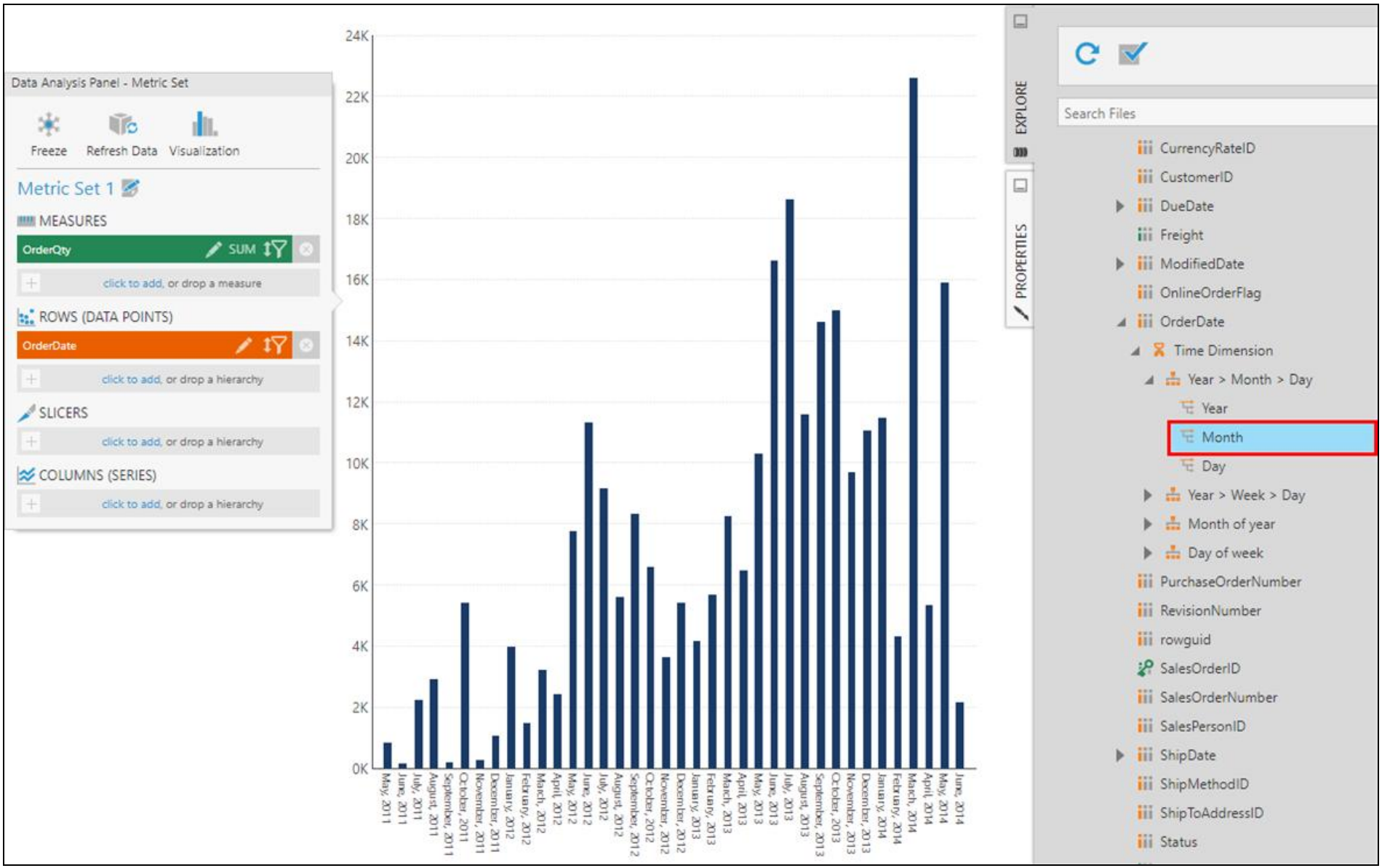
Related video: Period over Period Analysis

A period over period measure can be added to any metric set with a measure and a time dimension hierarchy. This example starts with a new metric set.

Create a Metric Set

You can create a full screen metric set from the main menu by choosing Business and then Metric Set, then clicking Create. You can also follow the steps below when editing a metric set on a dashboard or other view.

Add a measure (numeric data) to Measures in the Data Analysis Panel, and also add a time dimension hierarchy. In the figure below visualizing as a bar chart, we expanded OrderDate under a data connector to find a time dimension hierarchy and dragged its Month level to Rows (Data Points). If you are using a data cube instead, it must have a time dimension hierarchy set up already in its Process Result for you to use.



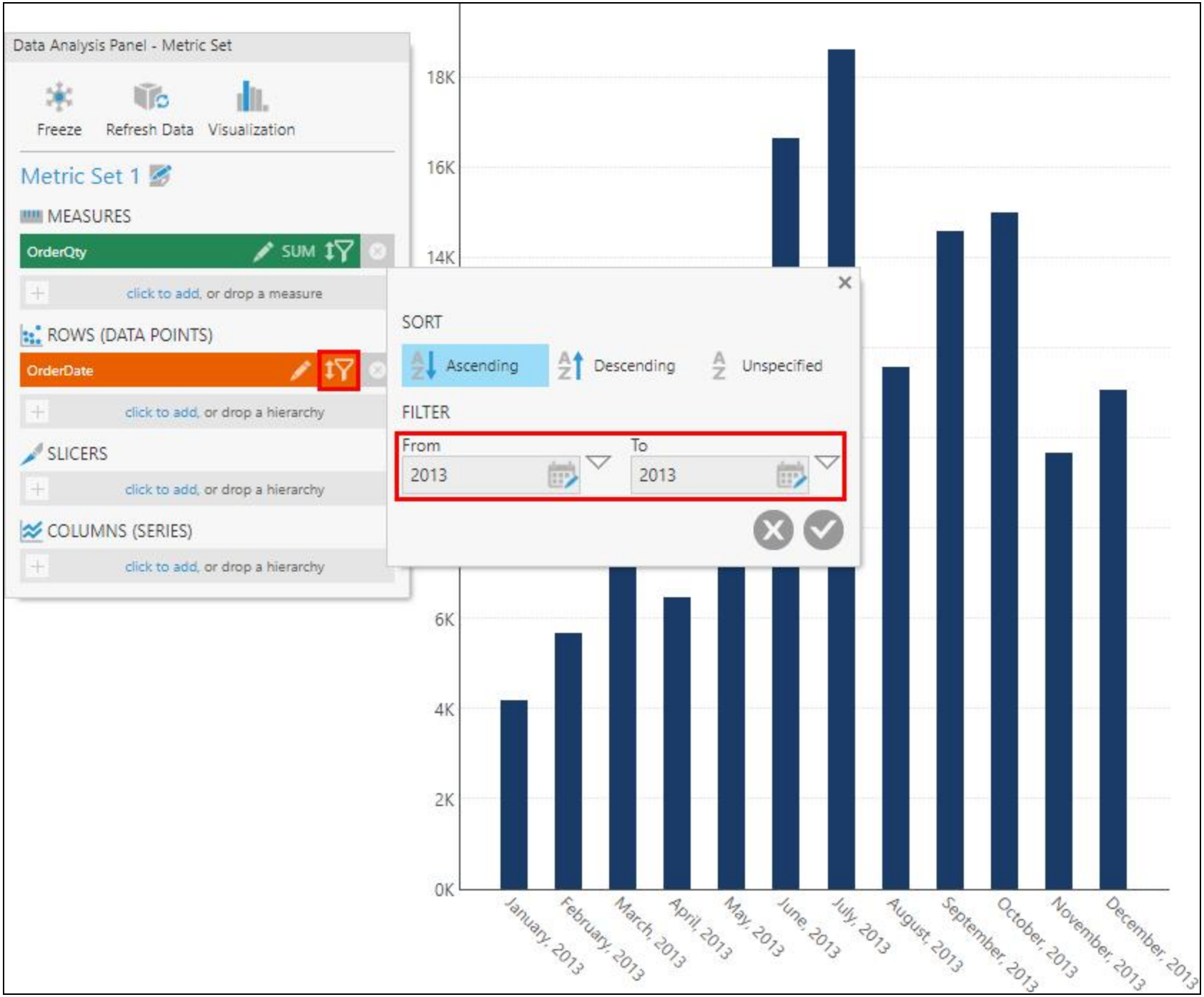
The time dimension hierarchy can also be selected on Slicers or Columns if you prefer.

There are built-in time dimensions ready for you to use, or you can create and/or customize one. For more information, see Create a Time Dimension.

Filter the Dates

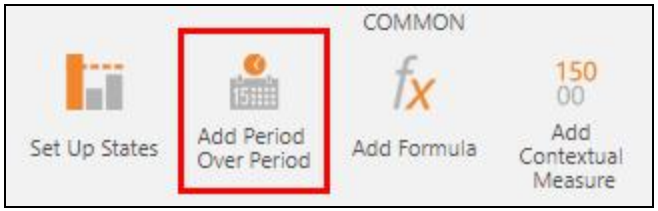
A period over period measure is most useful when comparing against a filtered range of dates.

In our example, we will filter our time dimension to show a single year, which you can do from the filter icon in the Data Analysis Panel.


Data Analysis Panel - Metric Set
Freeze
Refresh Data
Visualization
Metric Set 1
MEASURES
OrderQty
SUM
click to add, or drop a measure
ROWS (DATA POINTS)
OrderDate
click to add, or drop a hierarchy
SLICERS
click to add, or drop a hierarchy
COLUMNS (SERIES)
click to add, or drop a hierarchy
SORT
Ascending
Descending
Unspecified
FILTER
From
2013
To
2013
18K
16K
14K
6K
4K
2K
0K
January, 2013
February, 2013
March, 2013
April, 2013
May, 2013
June, 2013
July, 2013
August, 2013
September, 2013
October, 2013
November, 2013
December, 2013

Add a Period Over Period Measure

In the toolbar, click Add Period Over Period. When editing a dashboard or other view, you can find this option under Data Tools in the toolbar.

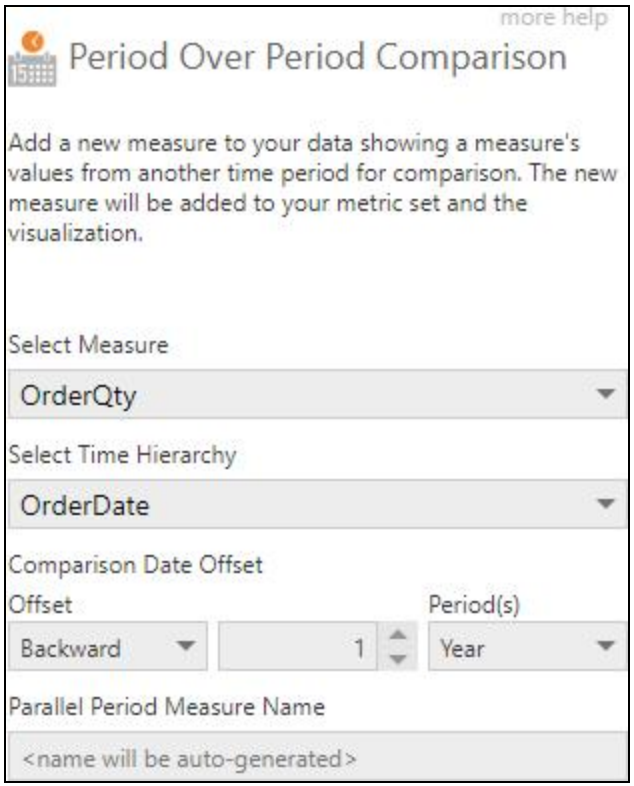


Note: This option won't be available until you add a time dimension as shown earlier.

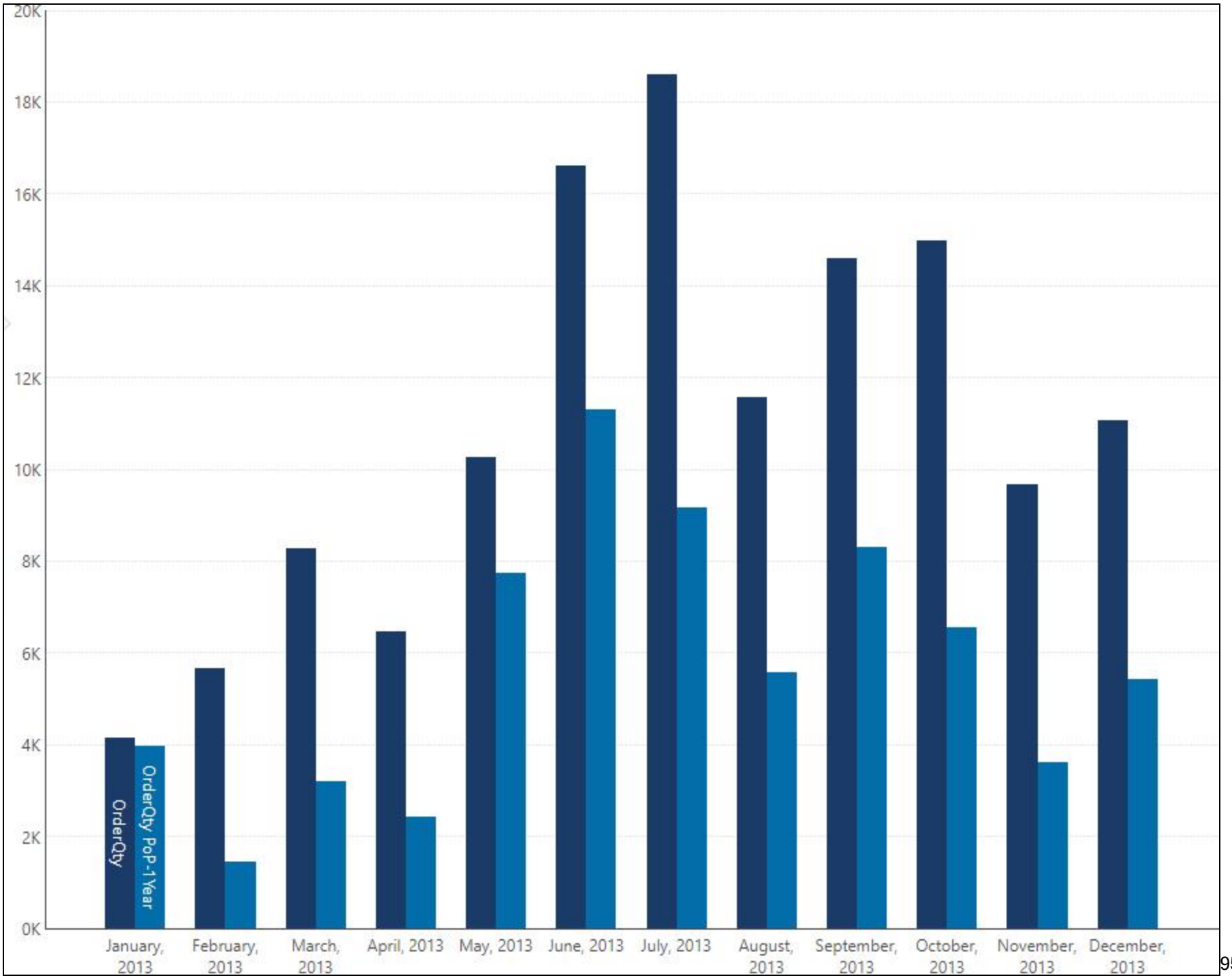
The Period Over Period Comparison dialog is displayed.

By default, the previous year's data is selected for comparison. Adjust these options if you prefer.

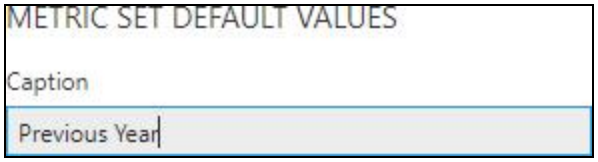
When editing a dashboard, there is a checkbox labeled Automatically Add Parameter Control. If selected, a Time Lag filter is added allowing you to change these period over period settings while viewing the dashboard. You can also add this filter later to your dashboard or to other types of views like reports.



Click the submit button at the bottom of the dialog to update the metric set with a new measure displaying data from a different time period directly alongside the original.


20K
18K
16K
14K
12K
10K
8K
6K
4K
2K
0K
OrderQty
OrderQty PoP-1Year
January, 2013
February, 2013
March, 2013
April, 2013
May, 2013
June, 2013
July, 2013
August, 2013
September, 2013
October, 2013
November, 2013
December, 2013

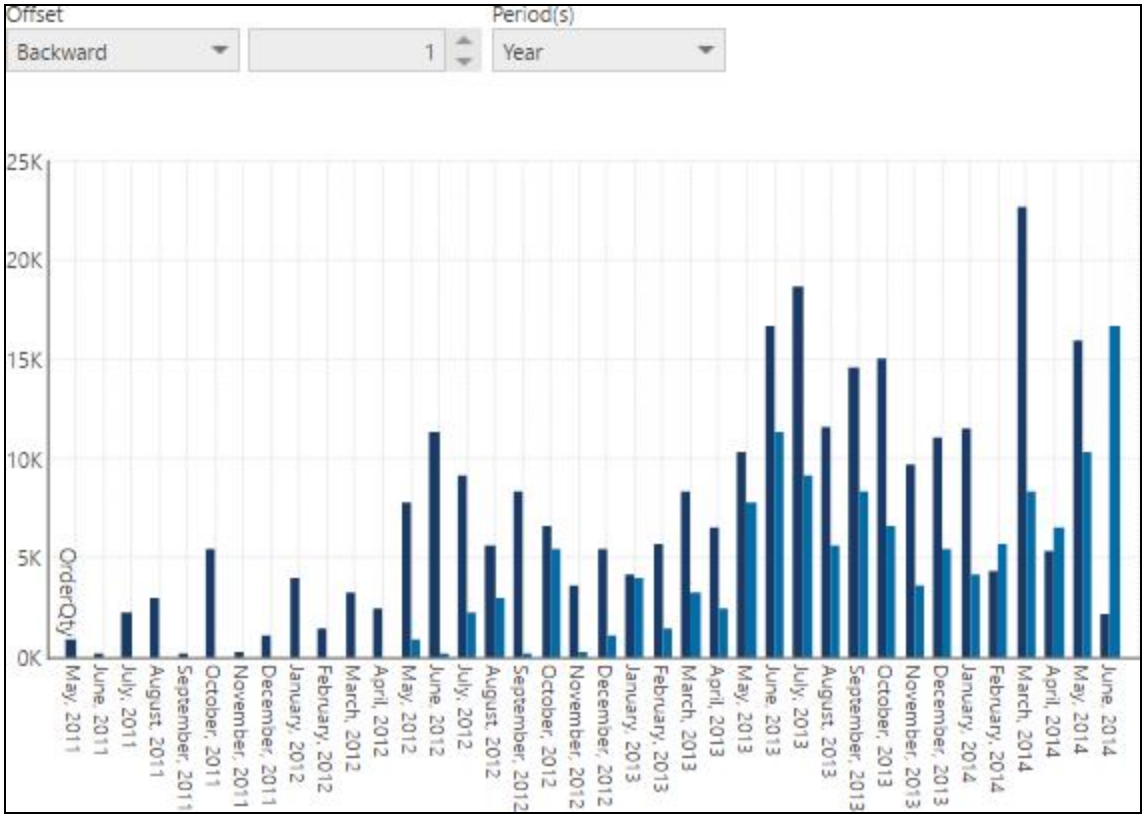
To rename the new measure, click to edit it in the Data Analysis Panel and change the Caption.



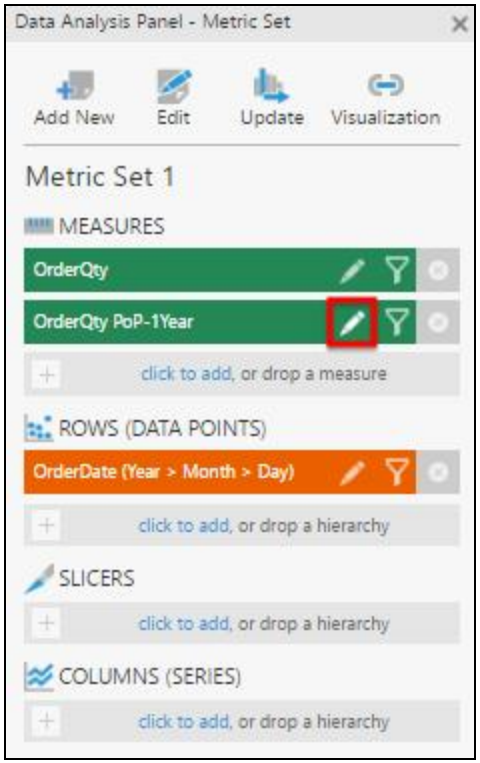
Returning All Comparison Data

A period over period measure normally only returns values for dates where there are corresponding values for the input measure (for example, OrderQty). However, there may be situations where you want to see all of the period over period measure values, regardless of whether there is anything to compare against. For example, you may want to see the period over period values extended into the future in order to determine sales goals.

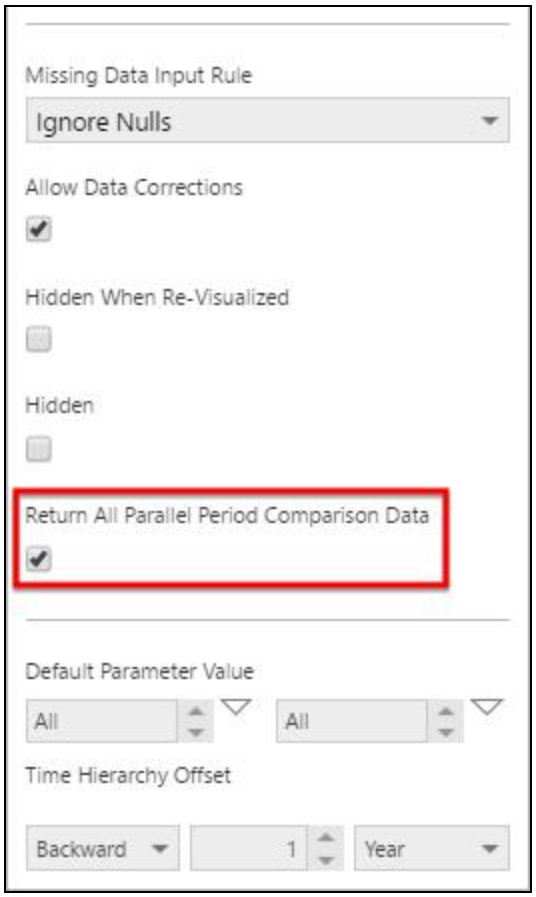
As an example, consider the same bar chart visualization of OrderQty and OrderQty PoP-1Year measures, but without the member filter applied.



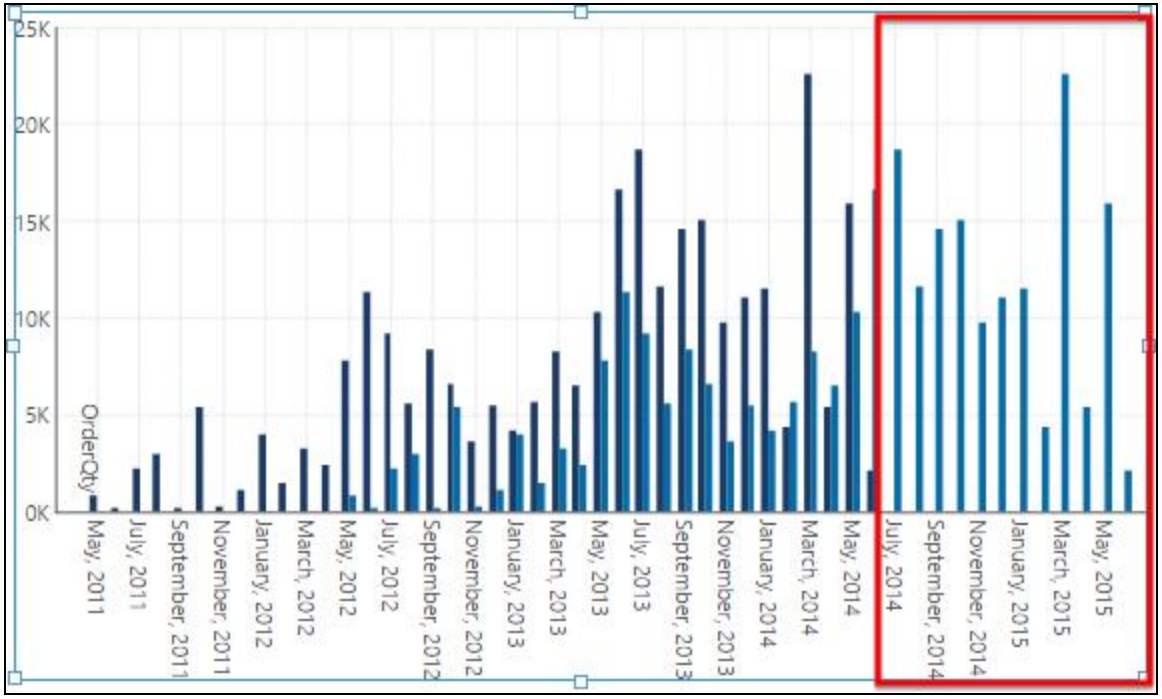
To have this chart show all of the period over period values, open the Data Analysis Panel and edit the OrderQty PoP-1Year measure.



In the Configure Metric Set Element dialog, select the Return All Parallel Period Comparison Data checkbox and click Submit.



The chart now displays all of the period over period values, which includes dates where there are no OrderQty values.



For more information, see:

- Video: Period over Period Analysis
- Other Metric Set Tools
- Adding Filters
- Symphony Metric Sets