onestream

Presentation Guide

9.4.1 Release

Copyright © 2026 OneStream Software LLC. All rights reserved.

All trademarks, logos, and brand names used on this website are the property of their respective owners. This document and its contents are the exclusive property of OneStream Software LLC and are protected under international intellectual property laws. Any reproduction, modification, distribution or public display of this documentation, in whole or part, without written prior consent from OneStream Software LLC is strictly prohibited.

Table of Contents

Overview 1

Presenting Data With Extensible Documents 2

Extensible Document Framework 2

Creating Extensible Documents 3

Using Document Variables in Extensible Documents 4

Using Parameters and Substitution Variables in Extensible Documents 5

Using Images in Extensible Documents 6

Using Retrieve Functions in Extensible Documents 6

Using Rich Text Content Controls 8

Document Variable Setup 8

Document Variable Field 9

Field Code 11

Rich Text Functionality 12

Refresh Document 12

Preserve Formatting During Updates 13

Parameters in Extensible Documents 15

Parameter and Substitution Variable Setup in Text Editor 16

Create an Extensible Document in Microsoft Word 17

Insert a Cube View in Microsoft Word 19

Insert Excel Sheets and Excel Named Ranges in Microsoft Word 23

Insert a Microsoft Word Document in Microsoft Word 28

Insert a Rich Text Document in Microsoft Word 31

Insert a Text Document in Microsoft Word 34

Insert a Report in Microsoft Word 37

Creating an Extensible Document in Text Editor 40

Insert a Cube View in Text Editor 41

Excel Sheets and Excel Named Ranges in Text Editor 45

Insert a Microsoft Word Document in Text Editor 50

Insert a Rich Text Document in Text Editor 53
Insert a Text Document in Text Editor 57
Insert a Report in Text Editor 61
Creating a Document in Microsoft PowerPoint 65
Creating a Document in Microsoft Excel 67
Using Extensible Documents in OneStream 70
Extensible Documents in Report Books 70
Extensible Documents in Dashboards 71
Extensible Documents in Data Management 71
Extensible Document Limitations 72
Presenting Data With Cube Views 81
Build a Simple Cube View 81
Cube View Toolbar 82
Create Cube View Groups and Profiles 83
Assign Cube View Groups to Cube View Profiles 87
Create a New Cube View 88

Create a Cube View Using Shared Rows and Columns 94
Copy a Cube View 96
Cube View Shortcuts 96
Member Filter Builder 97
Navigation 98
Member Expansion Tab 100
Time Functions Tab 102
Variables Tab 103
Samples Tab 105
Where Clauses 105
Cube View Performance 106
Database Sparsity 107
Sparse Row Suppression 113
Cube View Paging 114
Advanced Cube Views 116
Workflow 116

Dashboard and Form Parameters 116
Changing Member Display Name 118
Format a Cube View 118
Rows and Columns Member Filters Tab 119
Rename Rows and Columns 120
Rows and Columns Formatting Tab 121
Rows and Columns Data Tab 162
General Settings 165
Configuring Navigation Links for Linked Dashboards 172
Linked Cube Views 181
Linking Cube Views Through the Cube POV 188
Format a Report Page 190
Calculations 194
Calculation Types 194
GetDataCell Expressions 195
Column and Row Expressions 206

Dynamic Member Formulas and Cube Views 211
Presenting Data with Dashboards 226
Dashboard Groups 227
General (Dashboard Group) 228
Dashboard Profile Properties 228
General (Dashboard Profile) 228
Security 229
Dashboard Design Mode 230
Design Mode 231
View Dashboards 233
Dashboard Properties 235
General (Dashboard) 235
Behavior 236
Formatting 238
Action (Primary Dashboard Only) 241
Grid Layout Type 241

Display Format Settings 242
Dashboard Components 245
Create a Dashboard Component 245
Manage Components on a Dashboard 246
Position On Dashboard Properties 247
BI Viewer 248
Book Viewer 251
Button 251
Chart (Basic) 259
Chart (Advanced) 266
Check Box 282
Combo Box 282
Cube View 283
Data Explorer 283
Data Explorer Report 284
Date Selector 284

Dynamic Grid 286

Embedded Dashboard 307

File Viewer 308

Filter Editor 310

Gantt View 316

Grid View 317

Image 325

List Box 327

Label 327

Large Data Pivot Grid 329

Logo 331

Map 331

Member Tree 332

Menu 333

Password Box 354

Pivot Grid 354

Radio Button Group 356
Report 357
Text Editor 365
Text Viewer 367
Spreadsheet 367
Sankey Diagram 369
State Indicator 373
Text Box 377
Tree View 389
SQL Table Editor 390
Web Content 409
Component Formatting and Action Properties 411
Component Display Format Properties 420
Data Adapters 426
Create a Data Adapter 427
Command Types 427

Parameters 451
Files 452
Strings 453
Linked Dashboards 453
Reporting and Linking Objectives 454
Linked Dashboards Example 454
Presenting Data Using Report Books 463
Create a Report Book 464
Book Designer Toolbar 466
Book Properties 469
Determine Parameters from Content 469
Required Input Parameters 469
File Type 470
Excel Export Item 471
Report 471
Loop 472

Change Parameters 475
Workflow 476
POV 477
Variables 477
Parameters 478
If Statement 480
Using Variables in If Statements 480
Using Parameters in If Statements 480
Combining Statements 481
Else/Else If Statement 481
Book Preview Toolbar 482
Right-Click Options 484
Find and Assign Values Using Object Lookup 485
Parameters Example 487
Cube Views Example 487
Extensible Document Settings 489

Insert Content Using Document Variables 489

Overview

OneStream offers multiple methods for presenting data, each designed to support different reporting and analysis requirements. Report Books enable the consolidation of diverse reports and files into a single, comprehensive output. Cube Views provide a flexible way to query cube data and display it in various ways. Dashboards integrate components, data adapters, and other assets to deliver visually driven presentations. This guide provides an overview of these approaches, as well as additional data presentation options available within the platform, such as Extensible Documents.

This guide includes the following sections:

- Presenting Data With Extensible Documents
- Presenting Data With Cube Views
- Presenting Data with Dashboards
- Presenting Data Using Report Books

Presenting Data With Extensible Documents

The extensible document framework integrates OneStream content with Microsoft Office and text files. In turn, these documents become known as Extensible Documents. You can create Extensible Documents by selecting the desired document type: Microsoft Word document, Text Editor document, Excel spreadsheet, OneStream Spreadsheet, PowerPoint presentation, or a text file.

The value of Extensible Documents is inherently linked to OneStream and eliminates the need for third-party integration tools. When a document is launched, the content displays based on the parameter values, images, or retrieve function values. Extensible Documents enable you to display current data and objects from OneStream, which enables dynamic use and confidence in the information displayed.

> **NOTE:** Extensible Documents are also referenced as XF Docs.

Extensible Document Framework

After selecting the type of document desired, use OneStream's custom parameters, substitution variables, document variables, and retrieve functions to get specific information such as time periods, dimension member names and properties, data cell values, and other information. You can insert images into Extensible Documents that query and display Excel sheets, Excel Named Ranges, PDF files, OneStream reports, OneStream charts, and OneStream Cube Views.

After you configure an Extensible Document, save it as <NameofDocument>.xfdoc.ext, for example CostSpreadsheet.xfdox.xlsx. Xfdoc tells OneStream that this is an Extensible document, and .ext tells OneStream the file type. Load the document into OneStream's file share to launch it from OneStream or include it in dashboards or Report Books.

Creating Extensible Documents

To create an Extensible Document, start with the Microsoft document or text file needed to create the framework. Next, decide the type of data required and insert the parameters, images, or retrieve functions needed to pull data from OneStream and dynamically update the Extensible Document. Once the document is complete, save it with the appropriate file name. The following examples use the file name GolfStream-Report.

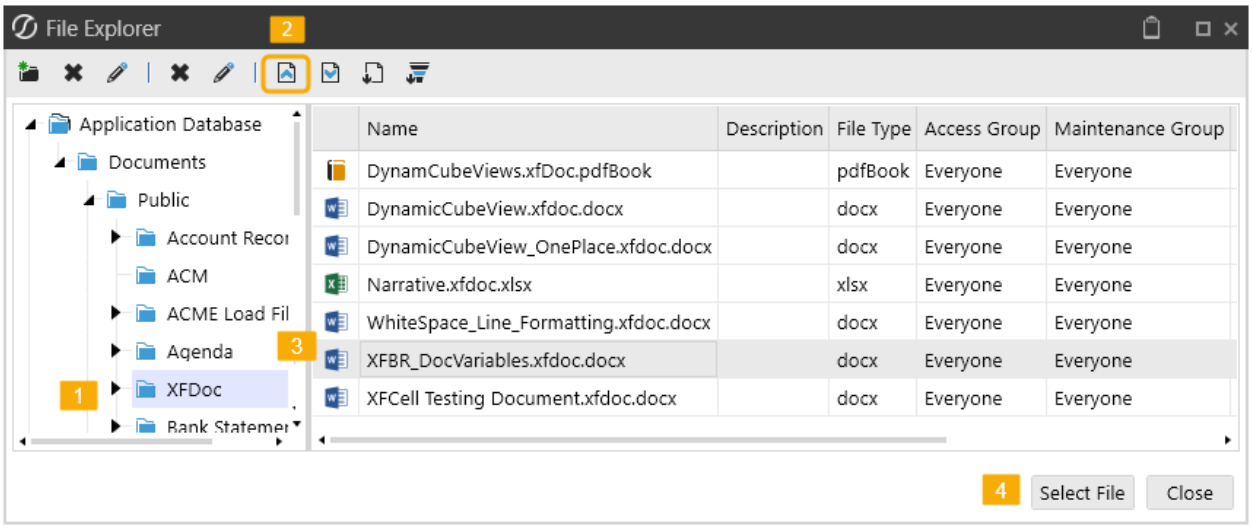
- **Word document**: GolfStream-Report.xfdoc.docx
- **Excel spreadsheet**: GolfStream Report.xfdoc.xlsx
- **PowerPoint presentation**: GolfStream Report.xfdoc.pptx
- **Text file**: GolfStream Report.xfdoc.txt

> **NOTE:** Extensible Documents are compatible with Microsoft Office version 2007 and later.

> **IMPORTANT:** When processing document variables within the .xfdoc file extension, OneStream enables you to reload the file without stripping the .xfdoc from the filename.

Once the document is saved in the format mentioned above, upload the document into OneStream's File Explorer.

Select the **File Explorer**  icon in OneStream to open the **File Explorer** window.



1. Select a folder to save the extensible documents in.
2. Select the **Upload File** icon to upload the document to OneStream.
3. Select the desired file name.
4. Select the **Select File** button to launch the document and see the updated values. It can also be run directly in OneStream Text Editor.

Using Document Variables in Extensible Documents

Document variables enable you to insert content into a Microsoft Word document or Text Editor as Extensible Documents. When using this method, you can view the processed OneStream content and edit the document simultaneously for Cube Views, Excel sheets, Excel named ranges, dashboard reports, and more.

See Document Variable Setup and Creating an Extensible Document in Text Editor.

Using Parameters and Substitution Variables in Extensible Documents

When creating Extensible Documents, use parameters as placeholders for OneStream member names, data, or user comments. Insert the parameter wherever specific information is needed in the document. For example, if the |!MyEntity!| parameter is used in the document's framework, once it is launched from OneStream, the parameter is recognized and replaced with the desired entity name. If double exclamation points are used, for example, |!!MyEntity!!|, it will pull the member description.

Some parameters prompt users to select an entity, account, time period, or other criteria required for the document to display correctly. Parameters can be used in any Word document, PowerPoint presentation, Excel spreadsheet, or text file. Any parameter that exists in OneStream can be used in an Extensible Document. See Parameters.

You can preserve formatting if you are inserting a parameter into an Extensible Document. This is optional and enables you to include formatting and ensure it is preserved during updates. See Parameters in Extensible Documents.

Substitution variables can also be used in Extensible Documents. These variables call out details such as an Application Name |AppName| or User Name |UserName|. They can also refer to a specific POV, creating versatility when reusing the document. See "Substitution Variables" in the *Application Tools Guide*.

See Creating a Document in Microsoft PowerPoint for an example on how to incorporate substitution variables into an Extensible Document.

Using Images in Extensible Documents

You can display a variety of reports, charts, Cube Views, Excel spreadsheets, PDFs, and rich text in an Extensible Document. Any of these items can be used in a Word document, PowerPoint presentation, or Excel spreadsheet.

Insert and right-click an image, then select the **View Alt Text** option. In Microsoft Word, this opens the View Alt Text box. In the **Alt Text** box, enter the required document variable arguments, referencing the path to the content that should replace the current image.

After saving the document variable argument alt text in the image, process and open the document through the OneStream File Explorer, or refresh the OneStream Text Editor or Spreadsheet page to process it. After processing the Extensible Document, the document variable content will replace the image.

For more details on an image's configuration, refer to the Extensible Document Settings in the Object Lookup dialog box in OneStream. See Extensible Document Settings.

See Create an Extensible Document in Microsoft Word and Insert Rich Text.

Using Retrieve Functions in Extensible Documents

You can use retrieve functions such as XFGetCell in Excel. These functions run on the server when launched from OneStream.In the Excel Add-In, load a Cube View, convert to XFGetCell, then select **Refresh Data** to display the updated Excel spreadsheet.

You can also create Excel charts based on the retrieve values. These charts display the correct data when you launch the spreadsheet from OneStream. Use the XFGetCellVolatile retrieve function to refresh the Excel chart and display updated data. Excel requires a volatile function for proper refreshing when using charts that reference calculated cells. See Creating a Document in Microsoft Excel.

The XFCell retrieve formula retrieves data from a single cell in OneStream. This is intended for text documents such as Word or PowerPoint. To prevent the function from defaulting off the cube POV, fill out as much of the POV as you can.

Parameters and substitution variables can also be used in an XFCell formula to ensure it is dynamic when using Extensible Documents.

> **Example:** XFCell (A#20500:E#|!MyEntityParameter!|:T#|Global|) returns a value for the specified account, the entity selected at run-time, and your application's global time period.

You can include additional settings in an XFCell function to format resulting data within an Extensible Document.

> **Example:** XFCell(A#20500:E#Clubs, Culture=User, NumberFormat=N3, DisplayNoDataAsZero=True, Scale=3, FlipSign=True, ShowPercentSign=False.

> **NOTE:** Any dimensions not specified in the formula come from the current POV.

For more details on XFCell syntax, refer to the Extensible Document Settings in the Object Lookup dialog box in OneStream, or see Extensible Document Settings.

See Creating a Document in Microsoft PowerPoint for an example on how to use XFCell with Extensible Documents.

Using Rich Text Content Controls

Rich text content controls are used in Microsoft Word only and enable users to add a Cube View, a dashboard report, a Word document (.docx or .xfDoc.docx), a rich text file (.rtf), or a text file (.txt) to any extensible Word document. When the Extensible Document is launched, the content control is replaced with formatted text that can be edited and reformatted. See Create an Extensible Document in Microsoft Word.

Document Variable Setup

To embed a document variable with narrative, start in Microsoft Word or OneStream Text Editor. When processing the document in OneStream File Explorer, select **Process and Open in Text Editor Page** or **Process and Open** to see the output in Text Editor or Microsoft Word. You can also edit the narrative, non-document variable content in Text Editor, including inserting new text or images.

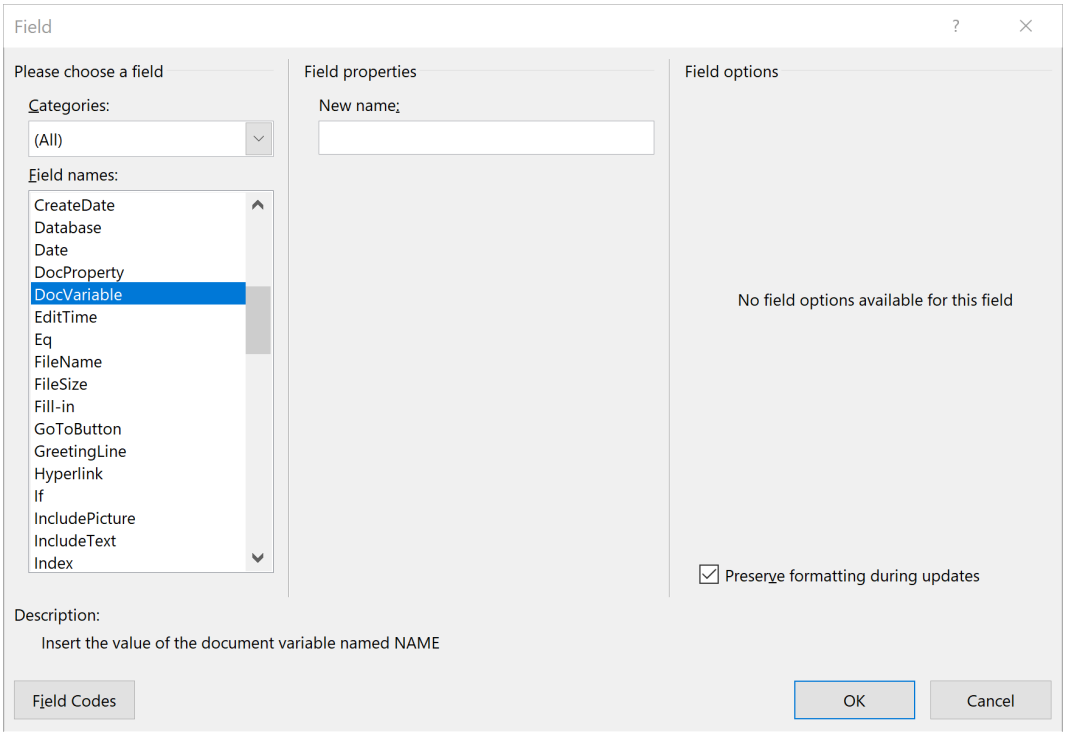
The following functions can be used to embed refreshable content:

- **Document Variables**: Enables you to insert content in a Microsoft Word document or OneStream Text Editor for XF Docs.
- **Rich Text Functionality**: Enables document variables to support rich text functionality and view rich text content in Microsoft Word or OneStream Text Editor.
- **Show/Hide Field Codes**: Enables you to view embedded content while in edit mode by toggling the content between processed and unprocessed views. Instead of navigating to OneStream File Explorer to process and open again, you can update document variables quickly, then refresh the Text Editor page to view updates.

- **Refresh Content**: Enables you to save and refresh your edits directly from the document to easily view your updates
- **MergeFormat**: Enables you to preserve document formatting when using parameters or substitution variables.

Document Variable Field

The DocVariable field enables you to insert extensible content from OneStream objects in Microsoft Word or Text Editor for XF Docs.

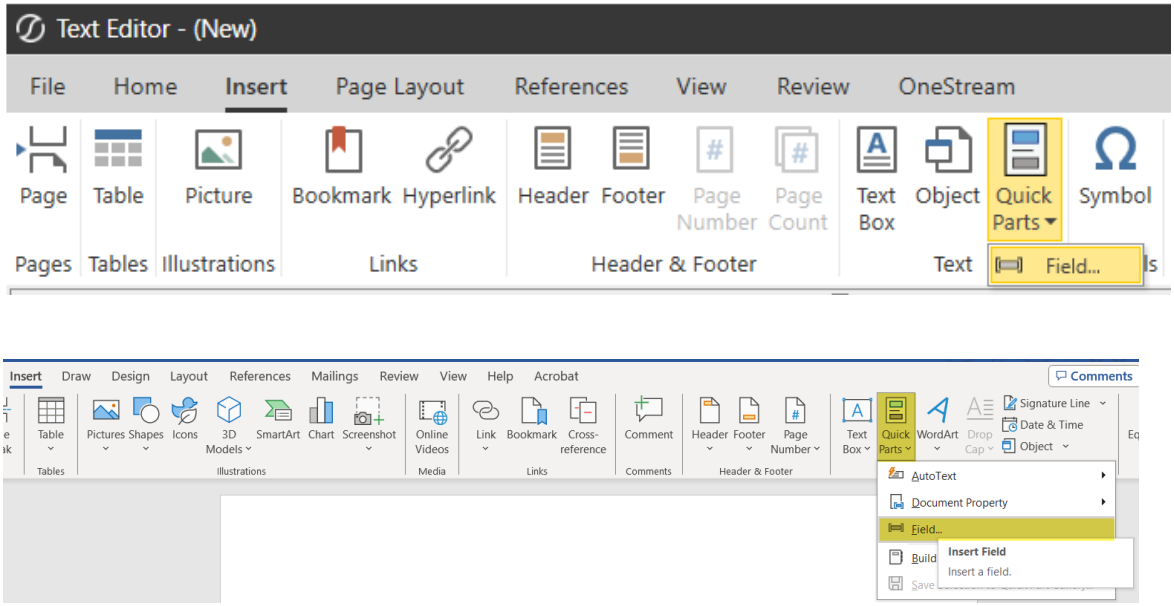




Quick Parts

Quick Parts is a Microsoft Word document feature available in Text Editor. The concept is similar to substitution variables in OneStream.

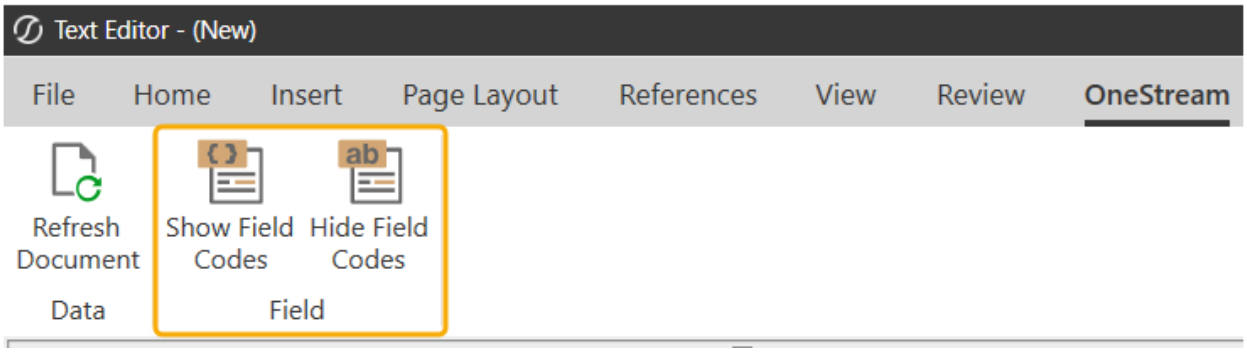
1. Go to **Insert** > **Quick Parts** > **Field**. Insert fields into a document that serve as placeholders for items such as Cube Views reports, Excel sheets, Excel Named Ranges, and more.



2. When the document is saved, refreshed, and opened, the fields dynamically update and display content
3. Field attributes and settings control what the field displays, how it operates, and where it is placed within the document.

Field Code

This function enables you to define the content item and formatting. Within Text Editor, go to the **OneStream** ribbon and select the **Show Field Codes** or **Hide Field Codes** button.



For the same result in Microsoft Word, right-click and select the Toggle option. You can also press **Alt** + **F9**.

> **TIP:** If pressing **Alt** + **F9** does not work, press **Alt** + **Fn** + **F9** simultaneously

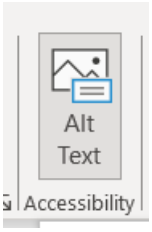
Alt Text

You can edit using the Alt Text functionality within OneStream Text Editor. Go to **Format** > **Alt Text**. Alt Text is disabled for text boxes and objects within Text Editor.

> **NOTE:** In OneStream Text Editor, the Format ribbon will not display unless an image is selected.



In Microsoft Word, go to **Picture Format** > **Alt Text**.



Rich Text Functionality

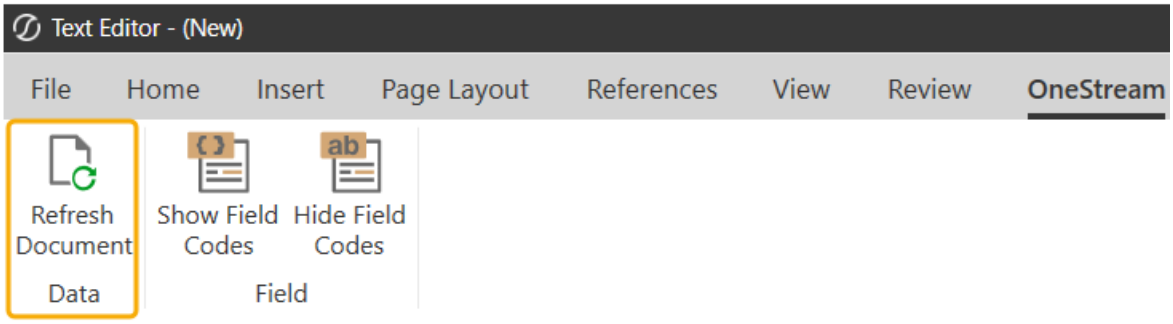
Document variables support the use of parameters in Microsoft Word or OneStream Text Editor. You can view parameters containing Literal Values equal to rich text format, which can be referenced in the document variable. You can also paste rich text format or plain text to display in the specific document variable. Then, you can process and open the XF Docs in Microsoft Word and OneStream Text Editor.

> **TIP:** If the rich text parameter content does not immediately display in Microsoft Word, press the **Alt** + **F9** or **Alt** + **Fn** + **F9** keys simultaneously to display the processed content.

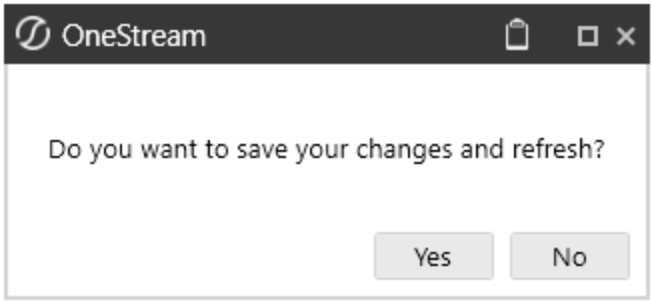
Refresh Document

You can automatically refresh a document within Text Editor and OneStream updates all the new content and narrative immediately. This enables you to quickly edit, save, and re-process all content.

1. In Text Editor, go to **OneStream** > **Refresh Document**.



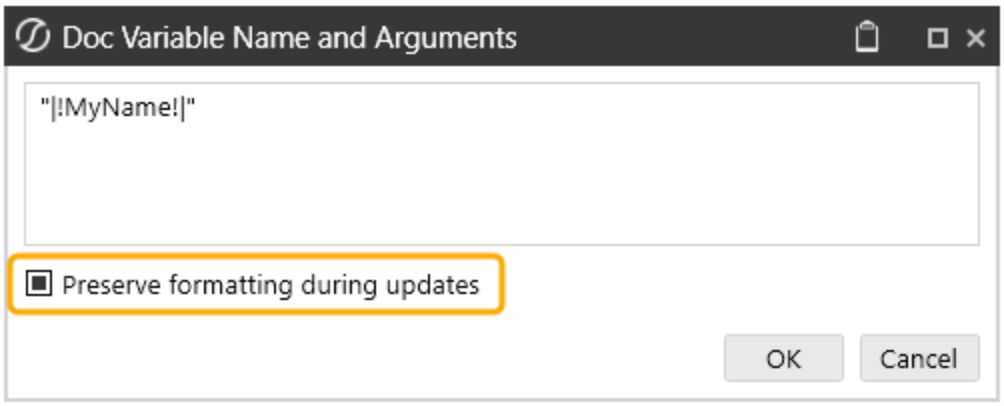
2. Select the **Refresh Document** button to refresh embedded content.
3. When selecting the Refresh Document button for an unprocessed XF Doc, you are prompted with a dialog box asking if you want to save your changes and refresh. Select the **Yes** button to save and re-process the document.



Preserve Formatting During Updates

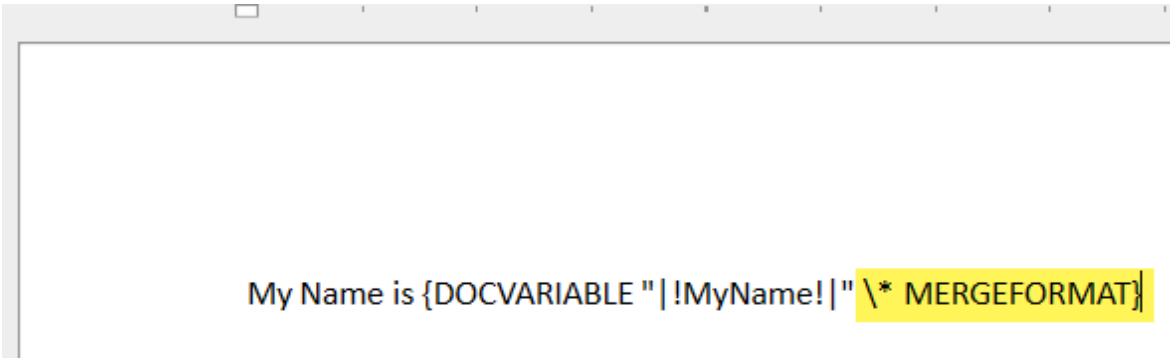
You can preserve formatting when inserting a parameter in an XF Doc. This is optional and enables you to include formatting that is preserved during updates through MergeFormat.

1. When inserting a document variable within Microsoft Word or Text Editor, the document variable field dialog box displays. Select the **Preserve formatting during updates** checkbox.

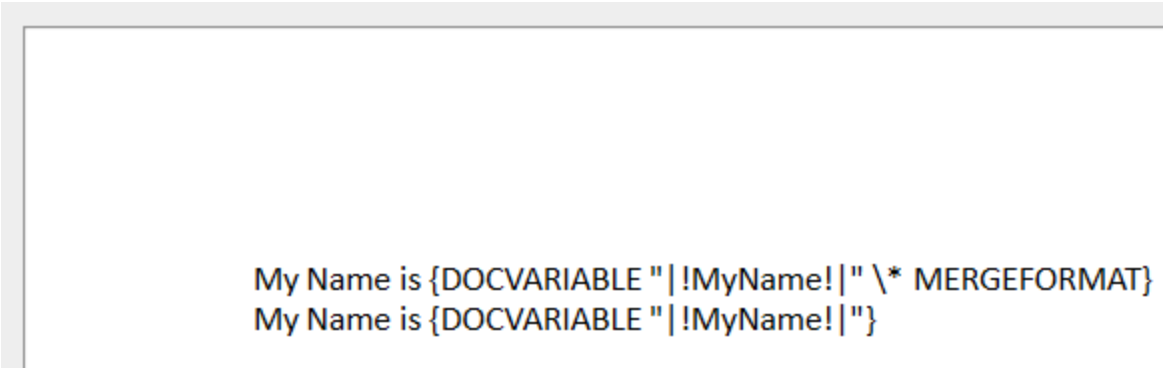


> **NOTE:** Within Microsoft Word, this is selected by default.

2. When this checkbox is selected, it adds the *MERGEFORMAT syntax to the end of the variable.



When this checkbox is deselected, the *MERGEFORMAT syntax is not added to the end of the variable.



Parameters in Extensible Documents

You can use parameters and substitution variables as placeholders to enhance extensible documents with selections for time period, entity, data cell values, and more.

The parameter name was created to display the entity input parameter for this document. When the XF Doc is processed at run-time, you are prompted to select a value.



After selecting parameters, the document displays the desired data. Additionally, you can update content on the parameter and add more narrative to an XF Doc. A parameter can inherit its format from the paragraph where it is inserted.

You make edits to your XF Docs by making the text red, bold, and font size 18. Within OneStream Text Editor or Microsoft Word, the values display with the associated Cube View requested in the Document Variable field, and the text entered appears with the correct font, color, and boldness.

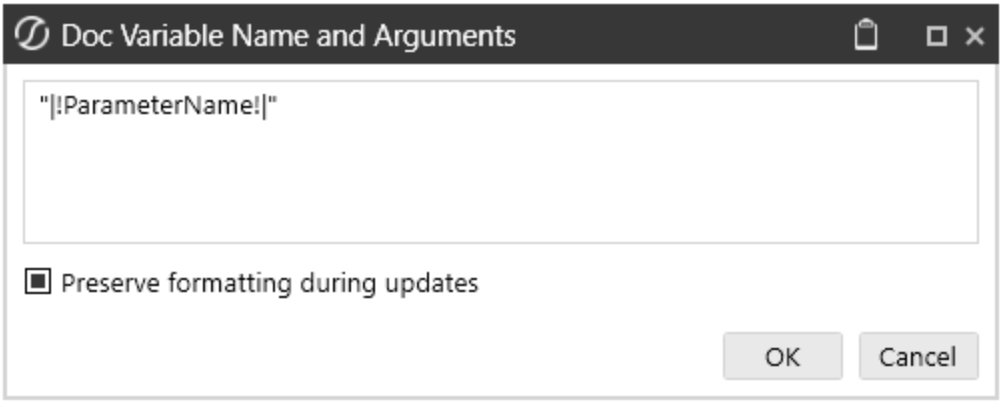
Parameter and Substitution Variable Setup in Text Editor

Complete the following steps in a new or existing XF Doc:

1. Within Text Editor, navigate to **Insert** > **Quick Parts** > **Field**.
2. Remove all existing syntax between the first set of quotes and everything after.
3. Copy and paste the parameter or substitution variable name surrounded by quotes. Ensure that you include standard object syntax.
4. Select the **Preserve formatting during updates** checkbox.

   > **NOTE:** This is an optional setting that will preserve formatting.

5. Select the **OK** button.



(Parameter Example) Key Data for {DOCVARIABLE "|!OpRev_Entity!|" * MERGEFORMAT}
(Sub Var Example) Accounting Period {DOCVARIABLE "|WFTimeDesc" * MERGEFORMAT}

Create an Extensible Document in Microsoft Word

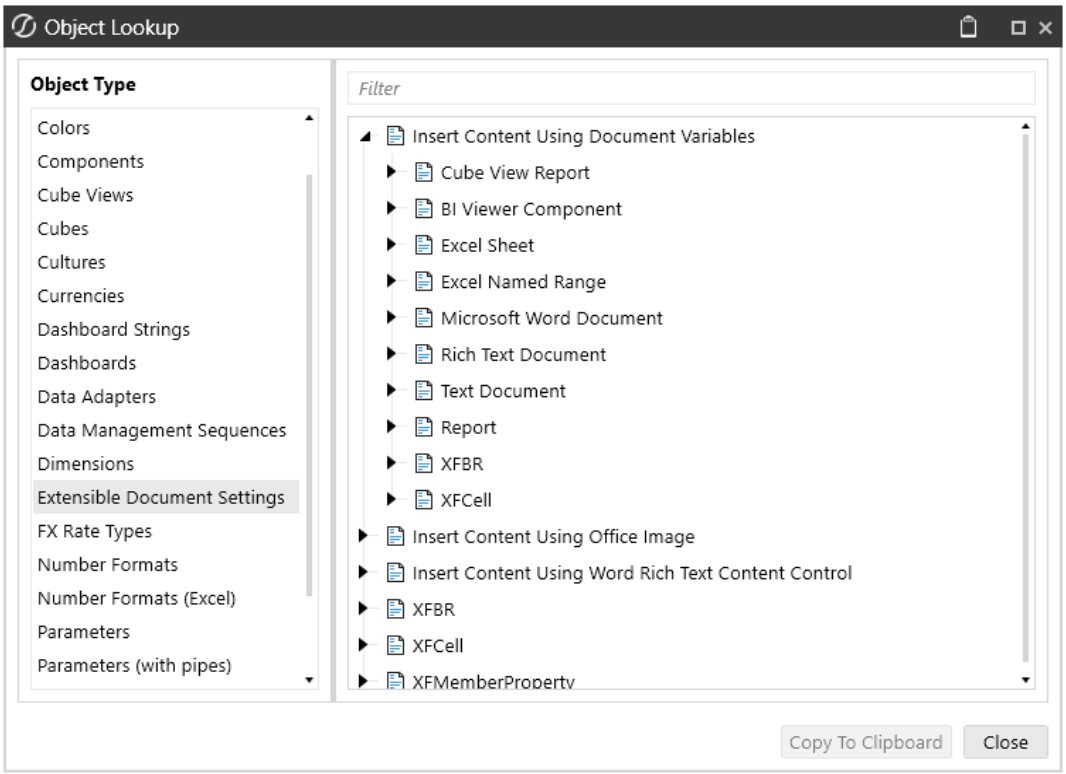
Refreshable content can be embedded in a Microsoft Word document using document variables. This enables you to edit and narrate a document to be published into a report.

- Insert a Cube View in Microsoft Word
- Insert Excel Sheets and Excel Named Ranges in Microsoft Word
- Insert a Microsoft Word Document in Microsoft Word
- Insert a Rich Text Document in Microsoft Word
- Insert a Cube View in Microsoft Word
- Insert a Report in Microsoft Word

> **NOTE:** When inserting content into a Word document, the embedded content takes on the page settings of the main document. To change the page settings for the inserted content, add a section break before the inserted content. After the section break, specify the desired page settings for the embedded content.

To insert content using document variables, copy all syntax directly from the Object Lookup. To insert content using document variables, copy all syntax directly from the Object Lookup.

Navigate to the **Object Lookup** icon and open the Object Lookup dialog box. This dialog box provides the syntax needed to insert content into an Extensible Document. This icon can be found under the Application tab on the following pages: Form Templates, Books, Cube Views, Workspaces, and Data Management.

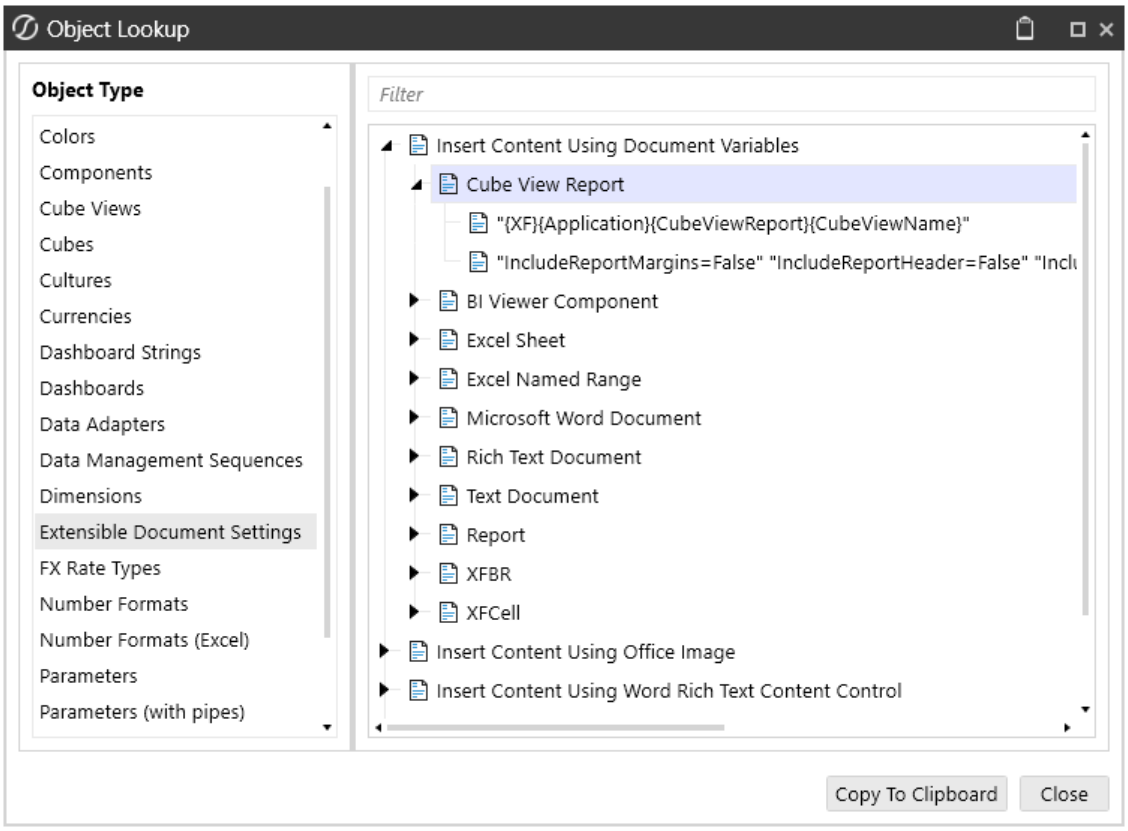


See Extensible Document Settings.

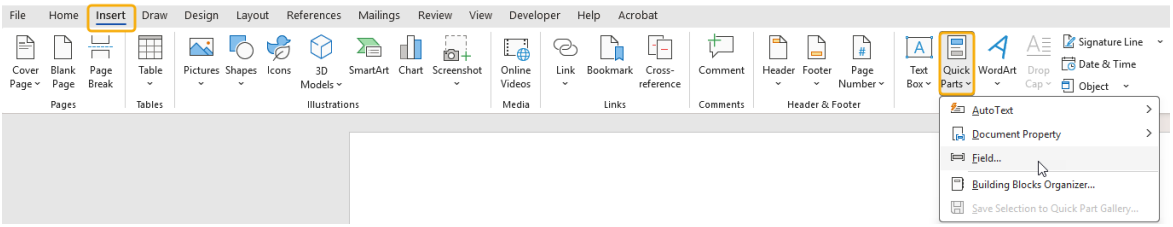
Insert a Cube View in Microsoft Word

> **NOTE:** The same format for inserting an image is used in Microsoft PowerPoint and Excel. For Microsoft upgrade Office version 1804 or higher, Microsoft changed how the Alt Text Title property is stored for embedded images. This impacts the creation of Extensible Documents in OneStream. Existing Extensible Documents continue to run as expected. This only impacts newly created Extensible Documents for version 1804 or higher of Microsoft Office.

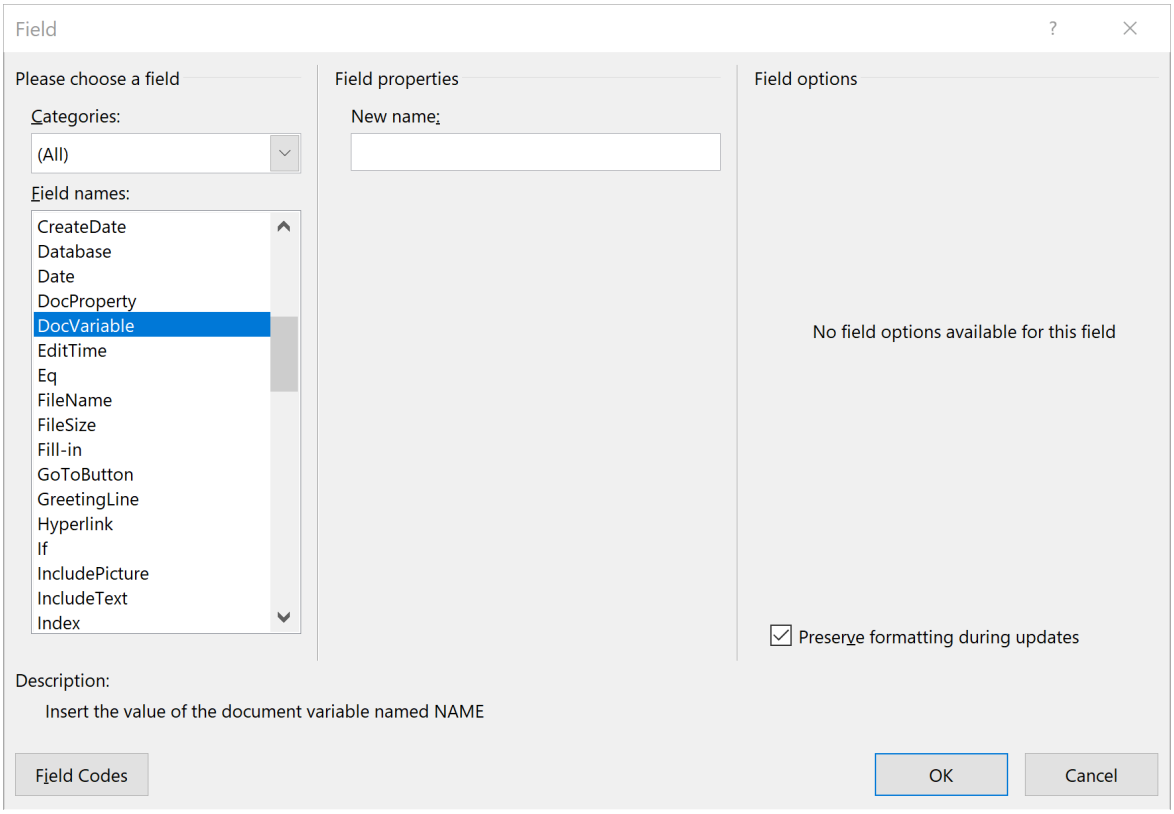
1. In the **Object Lookup** dialog box, select **Extensible Document Settings**. Expand **Insert Content Using Document Variables** and expand **Cube View Report**. Select the syntax line and then select the **Copy To Clipboard** button to copy the string.



2. Begin with a new or existing document in Microsoft Word.
3. To insert your string, go to **Insert** > **Quick Parts** > **Field**.



4. The **Field** dialog box displays. Under **Field Names**, select **DocVariable**. In the **New name** field, paste the string copied from Object Lookup.



After the first string in the DocVariable field, you can optionally paste the IncludeReportMargins=False arguments. The arguments must be separated by a space. See Extensible Document Settings.

> **TIP:** Under **Field Options**, select the **Preserve formatting during updates** checkbox. See Document Variable Setup.

5. Update the values to reflect the folder structure and the Cube View you are adding to the report. Select the **OK** button. The final string will display in the Microsoft Word document.

   > **Example:** {DOCVARIABLE}{XF}{Application}{CubeViewReport}{IncomeStatement}

> **IMPORTANT:** You must manually update and remove extra quotation marks and backslash symbols otherwise the document variable will not process correctly.

6. Place your cursor on the page, and press **Alt** + **F9**. This enables the selected value to be shown or hidden on the page.

   > **NOTE:** If expected content does not display after selecting the OK button in the Field dialog box, press **Alt** + **F9** or press **Alt** + **Fn** + **F9** simultaneously.

7. Select **File** > **Save As**. Select a file name and enter the **.xfdoc** extension to the name. The file name is appended with xfdoc.docx.
8. Select the **Save** button and ensure the file is saved in a familiar folder.
9. Navigate back to OneStream and open **File Explorer**. In the folder of your choice, select and upload the file.
10. Right-click on the file and select **Process and Open**. You may need to enter or select a parameter.

    The Cube View Report is now embedded in the extensible Word document.

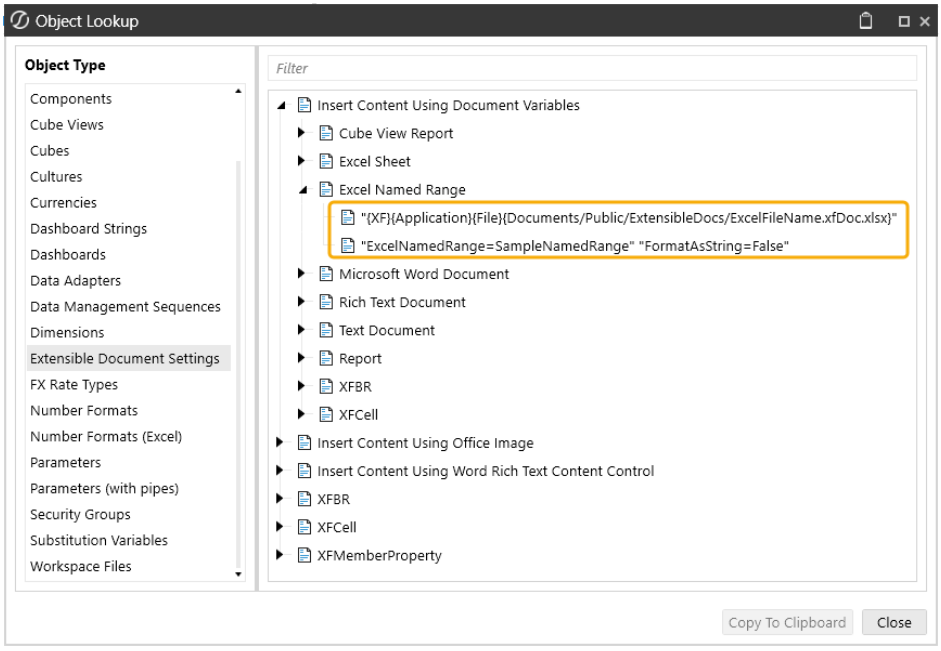
> **NOTE:** You can insert a Cube View using Microsoft Office Update version 1804 and higher.

> **TIP:** If the document variable content is shaded after processing and opening in Microsoft Word, update your settings in **File** > **Options** > **Advanced** > **Field shading: Never**.

Insert Excel Sheets and Excel Named Ranges in Microsoft Word

> **IMPORTANT:** Excel sheets and Excel Named Ranges must have 500 or fewer rows and 50 or fewer columns. If you include an argument for an Excel sheet or Excel Named Range that exceeds the maximum limit, you will receive an error message when attempting to process.

1. In the **Object Lookup** dialog box, select **Extensible Document Settings**. Expand **Insert Content Using Document Variables** and then expand **Excel Named Range**. Select the syntax line and then select the **Copy To Clipboard** button to copy the string.

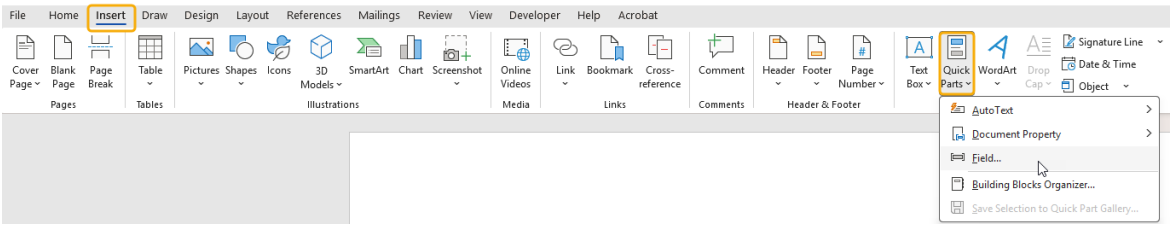


The default value for the FormatAsString parameter is False. When FormatAsString = True, the named range displays as a string instead of a table. If the named range has multiple cells or rows, the cell values are appended together and each row displays on a new line. If FormatAsString is removed from the document variable, the named range displays as a table as if FormatAsString=False.

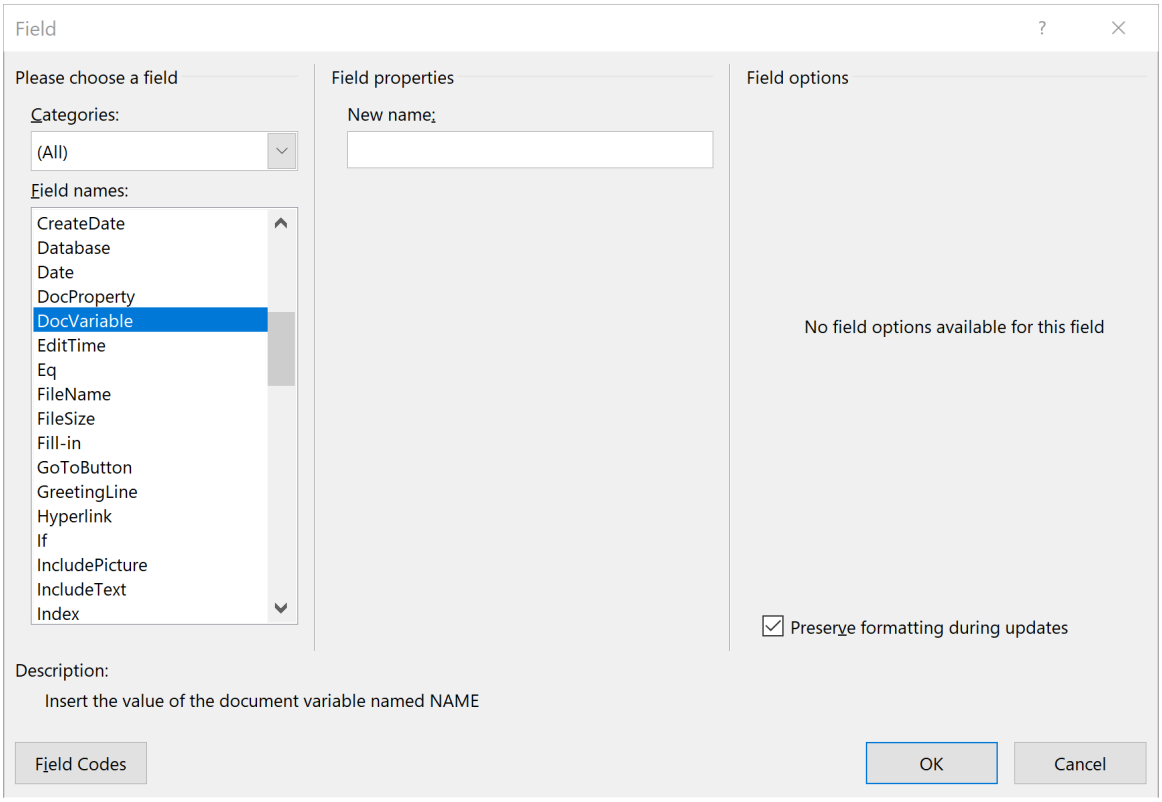
The default value for FitToPage parameter is False. When FitToPage=True, the named range is resized in the document to fit to page. The default value for Scaling parameter is False. When FitToPage=True, the scaling parameter can be set to True to further resize the named range. If FitToPage or Scaling are removed from the document, the value will display as False.

"{XF}{Application}{File}{Documents/Public/ExtensibleDocs/ExcelFileName.xfDoc.xlsx}"
"ExcelNamedRange=SampleNamedRange" "FormatAsString=False" "FitToPage=False" "Scaling=False"

2. Begin with a new or existing document in Microsoft Word.
3. To insert your string, go to **Insert** > **Quick Parts** > **Field**.



4. The Field dialog box displays. Under **Field Names**, select **DocVariable**. In the **New name** field, paste the string copied from Object Lookup.



After the first string in the DocVariable field, you can optionally paste the ExcelSheet=Sheet1 or ExcelNamedRange=SampleNamedRange arguments. The arguments must be separated by a space. See Extensible Document Settings.

> **TIP:** Under **Field Options**, select the **Preserve formatting during updates** checkbox. See Document Variable Setup.

5. Update the values to reflect the folder structure and Excel path. Select the **OK** button. The final string will display in the Microsoft Word document.

> **Example:** {DOCVARIABLE "{XF}{Application}{File}{Documents/Public/Extensible Documents/Excel4.xfDoc.xlsx}"ExcelSheet=Sheet2"}

Excel Sheet Example

{DOCVARIABLE "{XF}{Application}{File}{Documents/Public/Extensible Documents/ExcelSpreadsheet.xfDoc.xlsx}" "ExcelSheet=SheetExample" }

Excel Named Range Example

{DOCVARIABLE "{XF}{Application}{File}{Documents/Public/Extensible Documents/ExcelSpreadsheet.xfDoc.xlsx}" "ExcelNamedRange=NamedRangeExample" }

> **IMPORTANT:** You must manually update and remove extra quotation marks and backslash symbols otherwise the document variable will not process correctly.

6. Place your cursor on the page, and press **Alt** + **F9**. This enables the selected value to be shown or hidden on the page.

   > **NOTE:** If expected content does not display after selecting the OK button in the Field dialog box, press **Alt** + **F9** or press **Alt** + **Fn** + **F9** simultaneously.

7. Select **File** > **Save As**. Select a file name and add the .xfdoc extension to the name. The file name is appended with .xfdoc.docx.
8. Select the **Save** button and ensure it is saved in a familiar folder.
9. Navigate back to OneStream and open **File Explorer**. Under the folder of your choice, select and upload the file.

10. Right-click on the file and select **Process and Open**. You may need to enter or select a parameter.

    The Microsoft Word document displays the data being requested by the Excel Sheet or Excel Named Range.

Excel Sheet Example

1	2	3
4	5	6
7	8	9
10	11	12

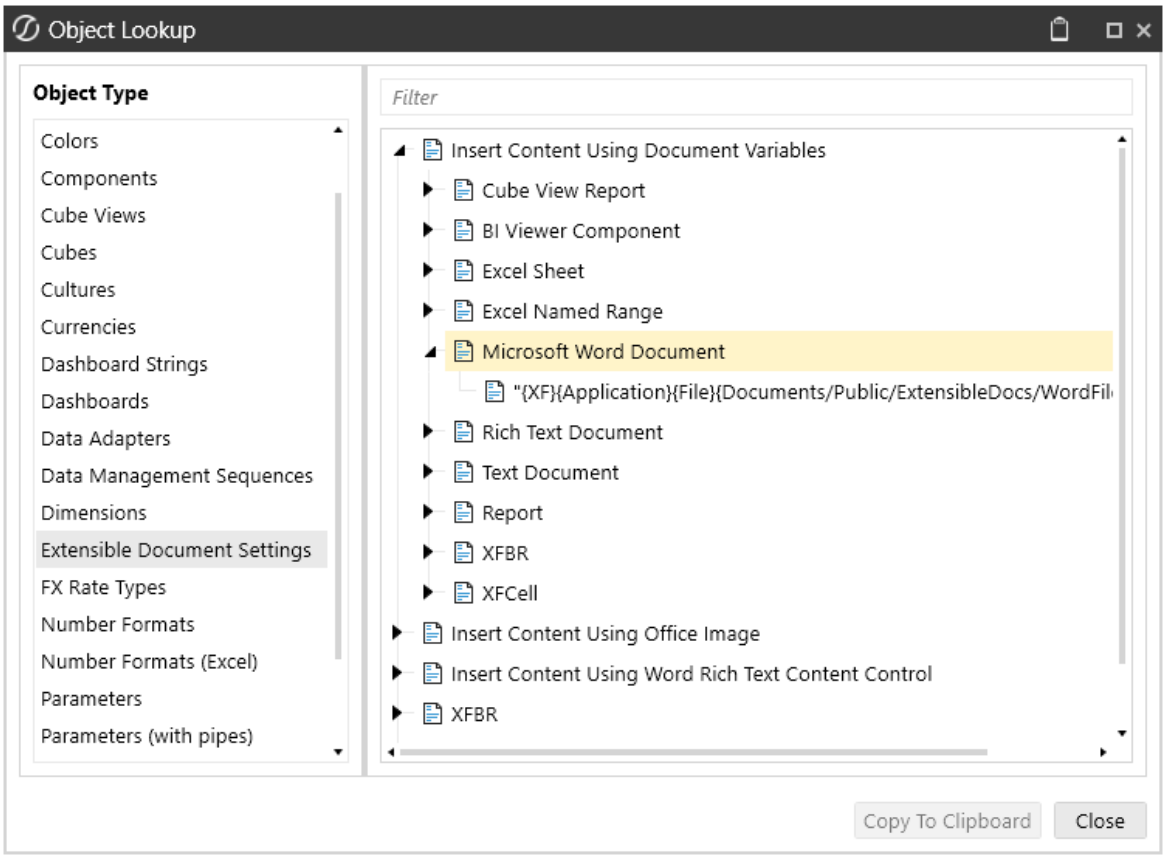
Excel Named Range Example

5	10	15	20

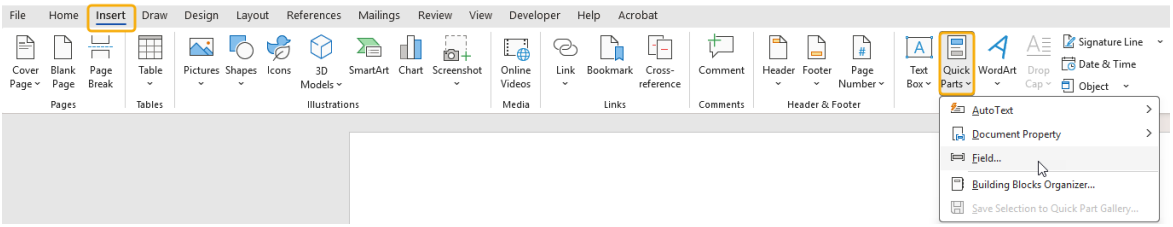
> **TIP:** If the document variable content is shaded after processing and opening in Microsoft Word, update your settings in **File** > **Options** > **Advanced** > **Field shading: Never**.

Insert a Microsoft Word Document in Microsoft Word

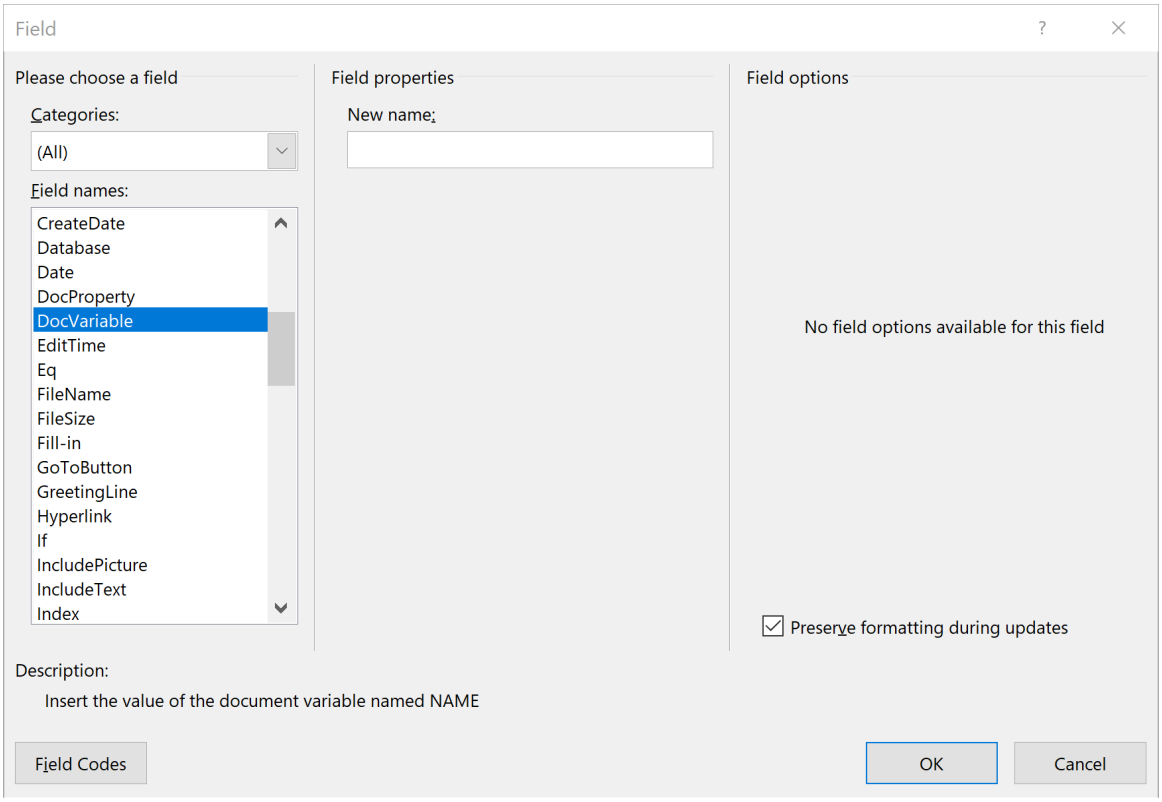
1. In the **Object Lookup** dialog box, select **Extensible Document Settings**. Expand **Insert Content Using Document Variables** and then expand **Microsoft Word Document**. Select the syntax line and then select the **Copy To Clipboard** button to copy the string.



2. Begin with a new or existing document in Microsoft Word.
3. To insert your string, go to **Insert** > **Quick Parts** > **Field**.



4. The Field dialog box displays. Under **Field Names**, select **DocVariable**. In the **New name** field, paste the string.



See Extensible Document Settings.

> **TIP:** Under **Field Options**, select the **Preserve formatting during updates** checkbox. See Document Variable Setup.

5. Update the values to reflect the folder structure and the Microsoft Word Document you want to include in the report. Select the **OK** button. The final string displays in the Microsoft Word document.

   > **Example:** { DOCVARIABLE \"{XF}{Application{File}{Documents/Public/ExtensibleDocs/WordFileName.docx}\" * MERGEFORMAT }

   > **IMPORTANT:** You must manually update and remove extra quotation marks and backslash symbols otherwise the document variable will not process correctly.

6. Place your cursor on the page and press **Alt** + **F9**. This enables the selected value to be shown or hidden on the page.

   > **NOTE:** If expected content does not display after selecting the OK button in the Field dialog box, press **Alt** + **F9** or press **Alt** + **Fn** + **F9** simultaneously.

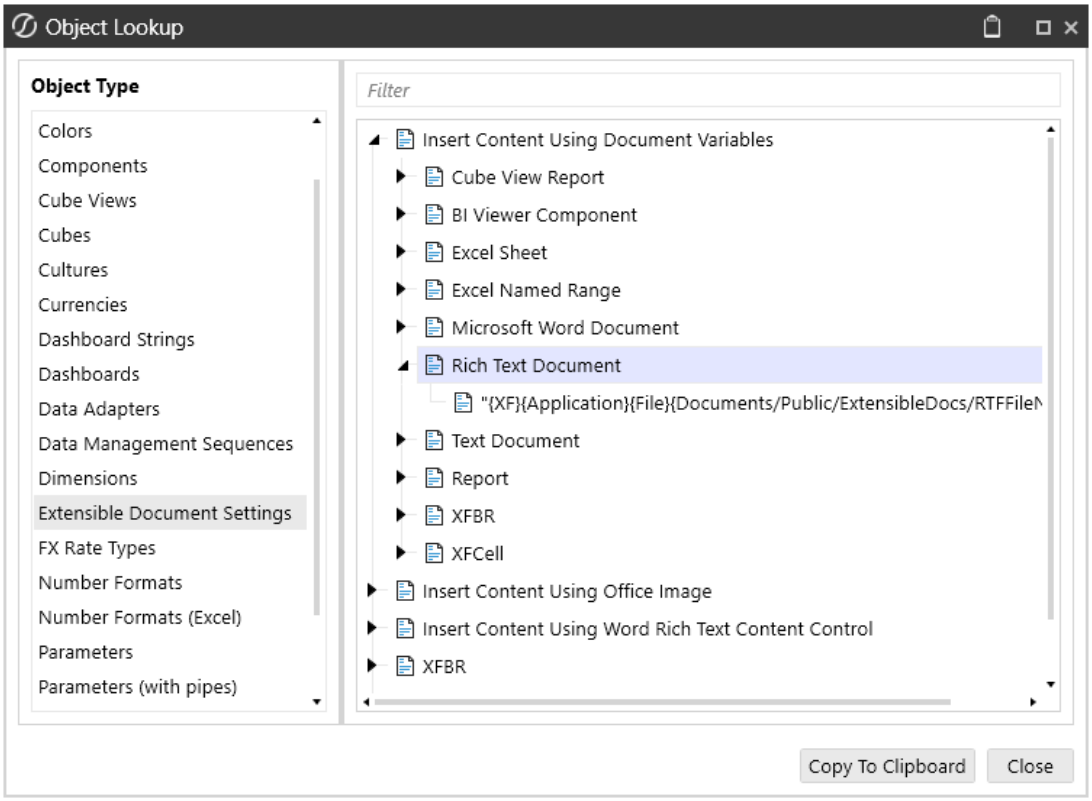
7. Select **File** > **Save As**. Select a file name and add the .xfdoc extension to the name. The file name is appended with .xfdoc.docx.
8. Select the **Save** button and ensure the file is saved in a familiar folder.
9. Navigate back to OneStream and open **File Explorer**. Under the folder of your choice, select and upload the filter.
10. Right-click on the file and select **Process and Open**. You may need to enter or select a parameter.

    The Microsoft Word document displays the data being requested from the referenced Microsoft Word Document.

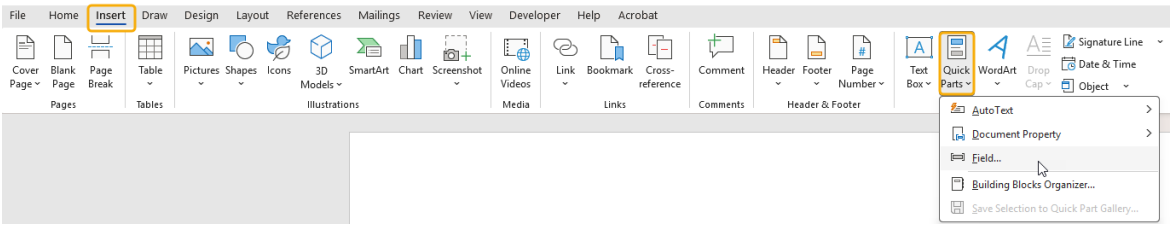
> **TIP:** If the document variable content is shaded after processing and opening in Microsoft Word, update your settings in **File** > **Options** > **Advanced** > **Field shading: Never**.

Insert a Rich Text Document in Microsoft Word

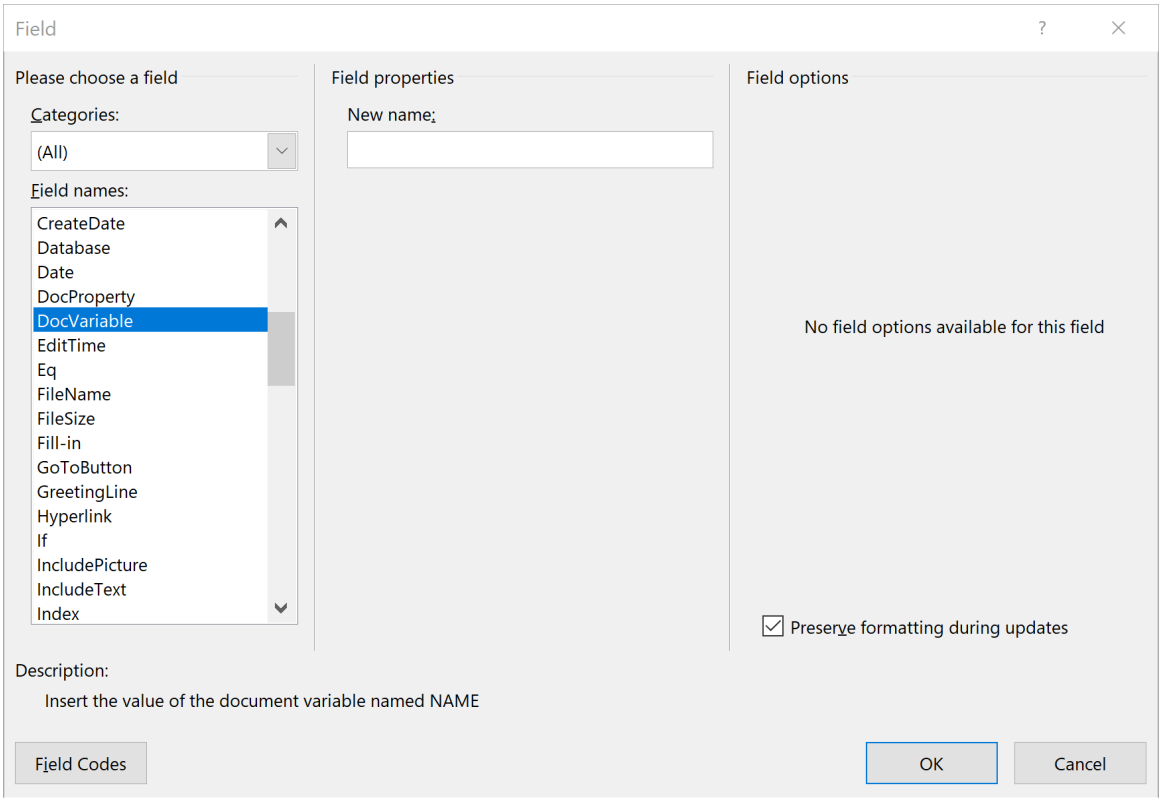
1. In the **Object Lookup** dialog box, select **Extensible Document Settings**. Expand **Insert Content Using Document Variables** and then expand **Rich Text Document**. Select the syntax line and then select the **Copy to Clipboard** button to copy the string.



2. Begin with a new or existing document in Microsoft Word.
3. To insert your string, go to **Insert** > **Quick Parts** > **Field**.



4. The Field dialog box displays. Under **Field Names**, select **DocVariable**. In the **New name** field, paste the string.



See Extensible Document Settings.

> **TIP:** Under **Field Options**, select the **Preserve formatting during updates** checkbox. See Document Variable Setup.

5. Update the values to reflect the folder structure and the rich text document path. Select the **OK** button. The final string displays in the Microsoft Word document.

   > **Example:** {DOCVARIABLE"{XF}{Application}{File}{Documents/Public/Extensible Documents/RichTextFormatDoc.rtf}"}

   > **IMPORTANT:** You must manually update and remove extra quotation marks and backslash symbols otherwise the document variable will not process correctly.

6. Place your cursor on the page, and press **Alt** + **F9**. This enables the selected value to be shown or hidden on the page.

   > **NOTE:** If expected content does not display after selecting the OK button in the Field dialog box, press **Alt** + **F9** or press **Alt** + **Fn** + **F9** simultaneously.

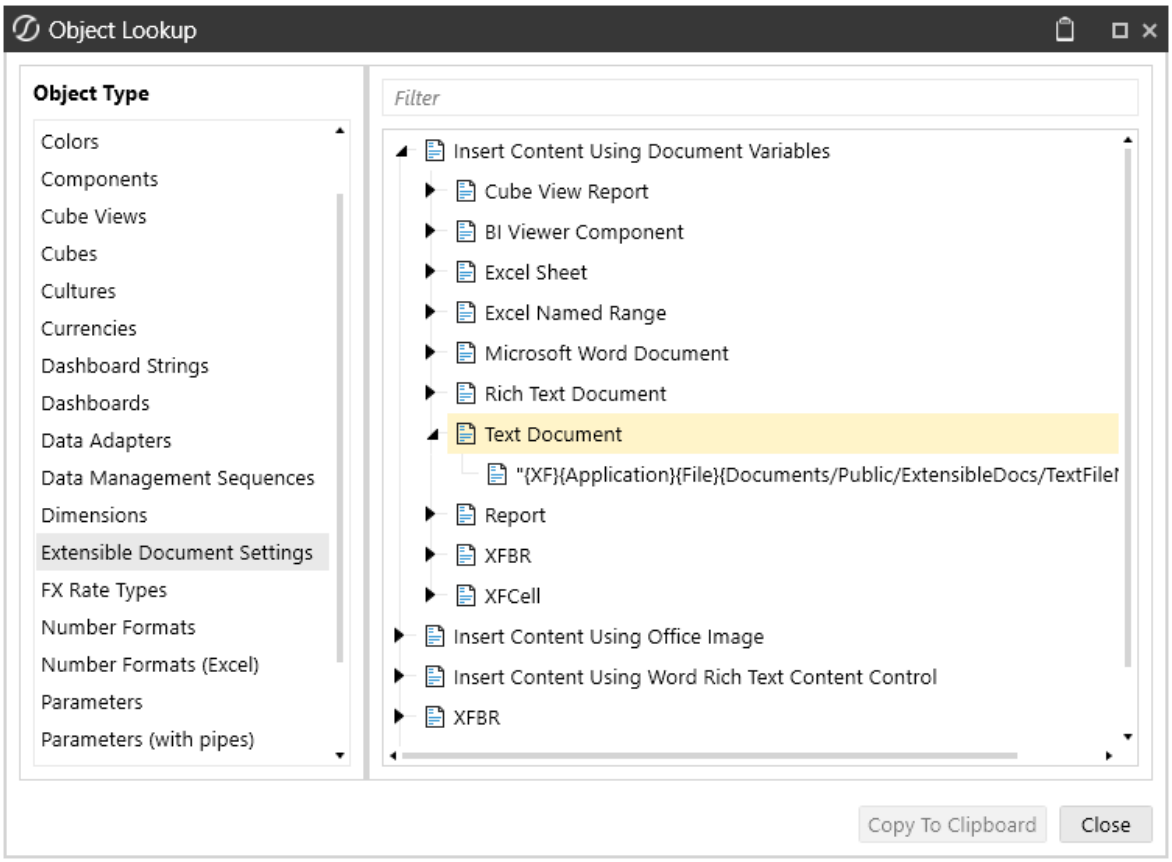
7. Select **File** > **Save As**. Select a file name and add the .xfdoc extension to the name. The file name is appended with .xfdoc.docx.
8. Select the **Save** button and ensure it is saved in a familiar folder.
9. Navigate back to OneStream and open **File Explorer**. Under the folder of your choice, select and upload the file.
10. Right-click on the file and select **Process and Open**. You may need to enter or select a parameter.

    The Microsoft Word document displays the data being requested for the Rich Text Document.

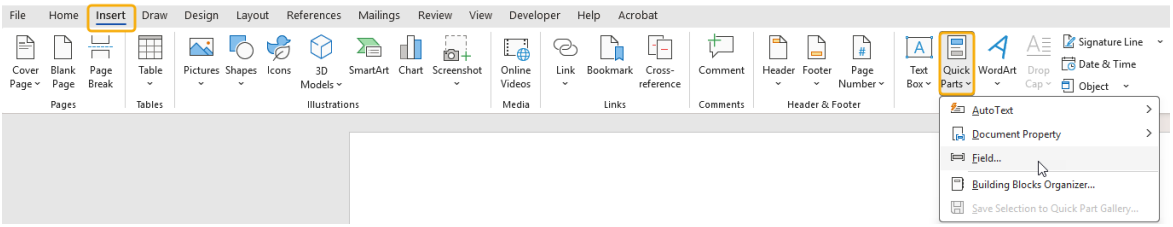
> **TIP:** If the document variable content is shaded after processing and opening in Microsoft Word, update your settings in **File** > **Options** > **Advanced** > **Field shading: Never**.

Insert a Text Document in Microsoft Word

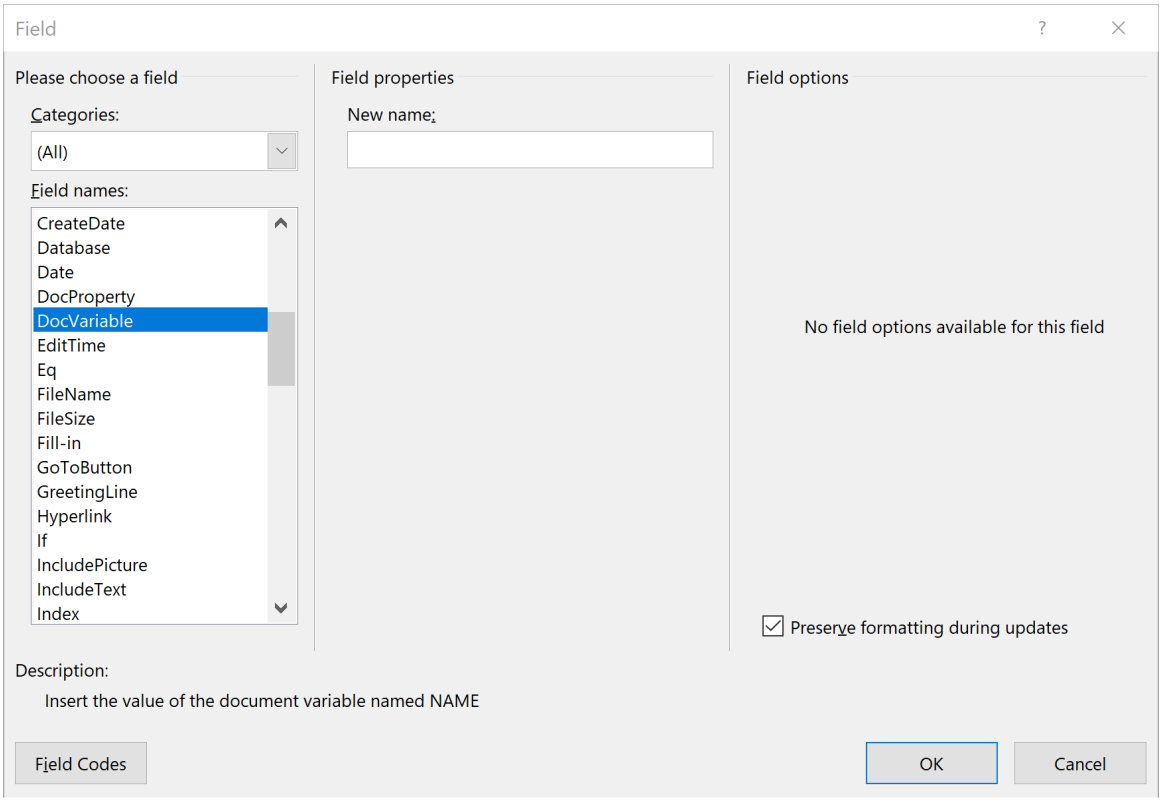
1. In the **Object Lookup** dialog box, select **Extensible Document Settings**. Expand **Insert Content Using Document Variables** and then expand **Text Document**. Select the syntax line and then select the **Copy to Clipboard** button to copy the string.



2. Begin with a new or existing document in Microsoft Word.
3. To insert your string, go to **Insert** > **Quick Parts** > **Field**.



4. The Field dialog box displays. Under **Field Names**, select **DocVariable**. In the **New name** field, paste the string.



See Extensible Document Settings.

> **TIP:** Under **Field Options**, select the **Preserve formatting during updates** checkbox. See Document Variable Setup.

5. Update the values to reflect the folder structure and text file path. Select the **OK** button. The final string displays in the Microsoft Word document.

   > **Example:** {DOCVARIABLE "{XF}{Application}{File}{Documents/Public/Extensible Documents/Testtoupdate.txt}"}

   > **IMPORTANT:** You must manually update and remove extra quotation marks and backslash symbols otherwise the document variable will not process correctly.

6. Place your cursor on the page and press **Alt** + **F9**. This enables the selected value to be shown or hidden on the page.

   > **NOTE:** If expected content does not display after selecting the OK button in the Field dialog box, press **Alt** + **F9** or press **Alt** + **Fn** + **F9** simultaneously.

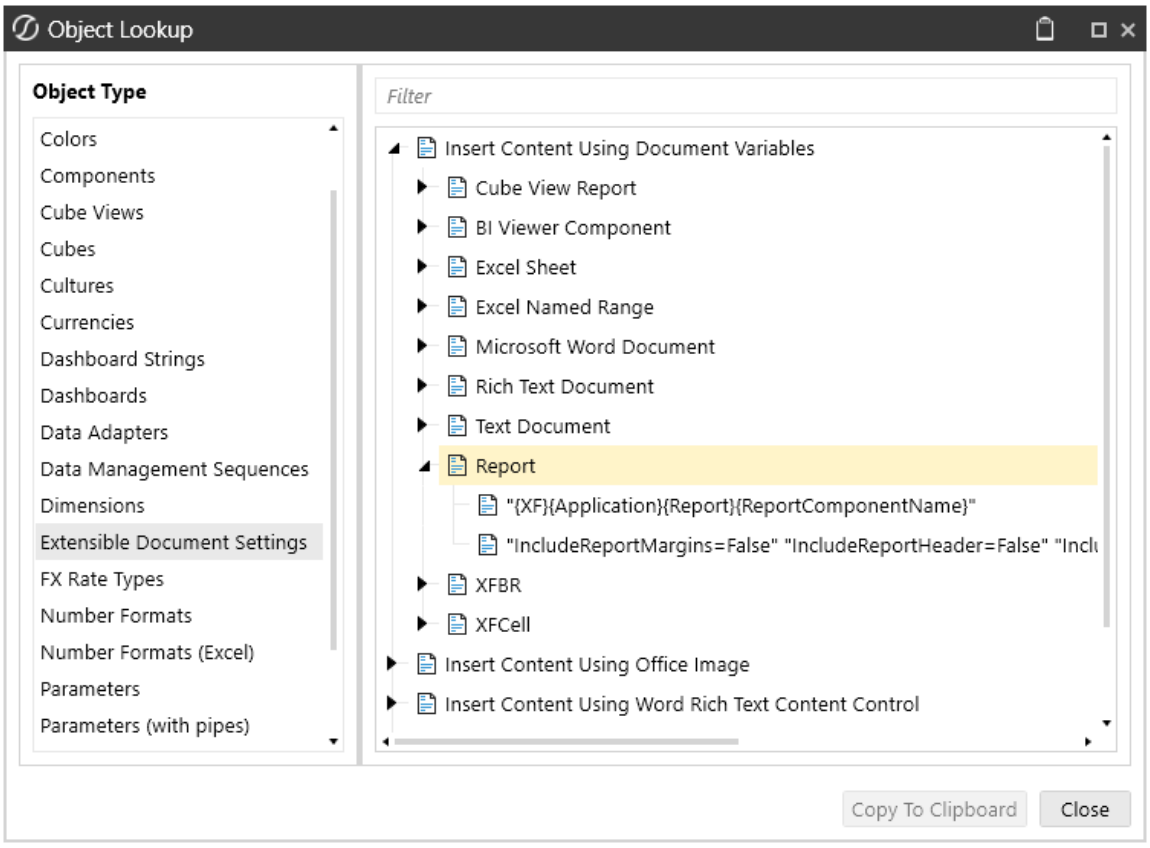
7. Select **File** > **Save As**. Select a file name and add .xfdoc extension to the name. The file name is appended with .xfdoc.docx.
8. Select the **Save** button and ensure the file is saved in a familiar folder.
9. Navigate back to OneStream and open **File Explorer**. Under the folder of your choice, select and upload the file.
10. Right-click on the file and select **Process and Open**. You may need to enter or select a parameter.

    The Microsoft Word document displays the data being requested from the Text File content.

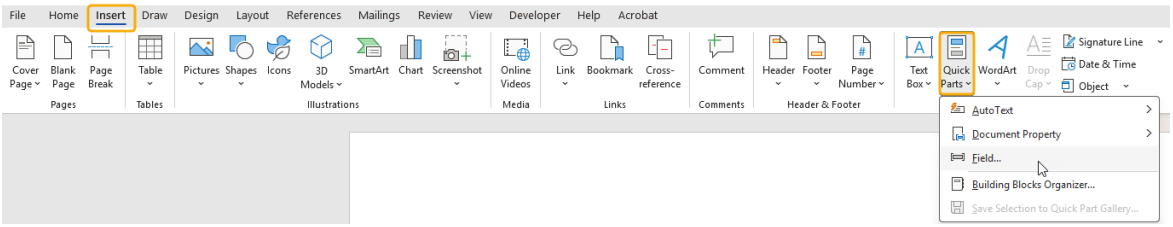
> **TIP:** If the document variable content is shaded after processing and opening in Microsoft Word, update your settings in **File** > **Options** > **Advanced** > **Field shading: Never**.

Insert a Report in Microsoft Word

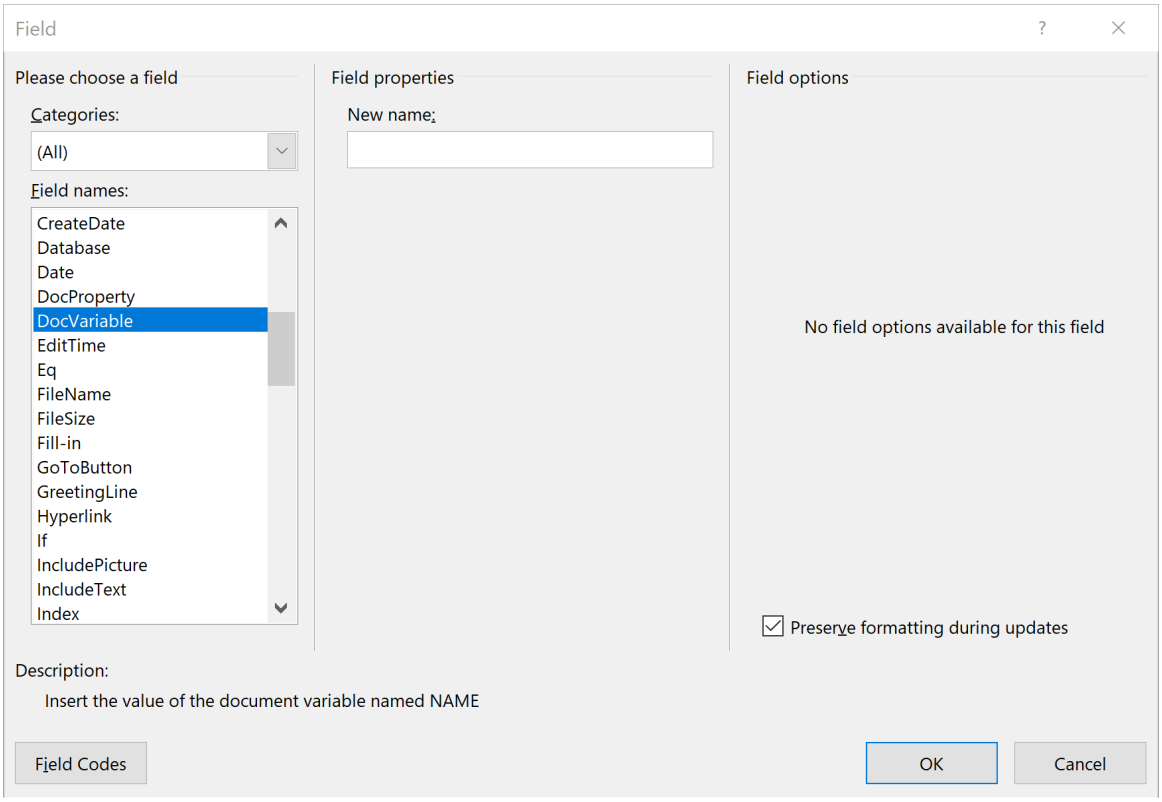
1. In the **Object Lookup** dialog box, select **Extensible Document Settings**. Expand **Insert Content Using Document Variables** and then expand **Report**. Select the syntax line and then select the **Copy to Clipboard** button to copy the string.



2. Begin with a new or existing document in Microsoft Word.
3. To insert your string, go to **Insert** > **Quick Parts** > **Field**.



4. The Field dialog box displays. Under **Field Names**, select **DocVariable**. In the **New name** field, paste the string.



After the first string in the DocVariable field, you can optionally paste the IncludeReportMargins=False arguments. All arguments must be separated by a space. See Extensible Document Settings.

> **TIP:** Under **Field Options**, select the **Preserve formatting during updates** checkbox. See Document Variable Setup.

5. Update the values to reflect the report component name. Select the **OK** button. The final string displays in the Microsoft Word document:

   > **Example:** {DOCVARIABLE "{XF}{Application}{Report}{Report3}"}

   > **IMPORTANT:** You must manually update and remove extra quotation marks and backslash symbols otherwise the document variable will not process correctly.

6. Place your cursor on the page, and press **Alt** + **F9**. This enables the selected value to be shown or hidden on the page.

   > **NOTE:** If expected content does not display after selecting the OK button in the Field dialog box, press **Alt** + **F9** or press **Alt** + **Fn** + **F9** simultaneously.

7. Select **File** > **Save As**. Select a file name and add the.xfdoc extension to the name. The file name is appended with .xfdoc.docx.
8. Select the **Save** button and ensure it is saved in a familiar folder.
9. Navigate back to OneStream and open **File Explorer**. Under the folder of your choice, select and upload the file.
10. Right-click on the file and select **Process and Open**. You may need to enter or select a parameter.

    The Microsoft Word document displays the embedded report.

> **TIP:** If the document variable content is shaded after processing and opening in Microsoft Word, update your settings in **File** > **Options** > **Advanced** > **Field shading: Never**.

Creating an Extensible Document in Text Editor

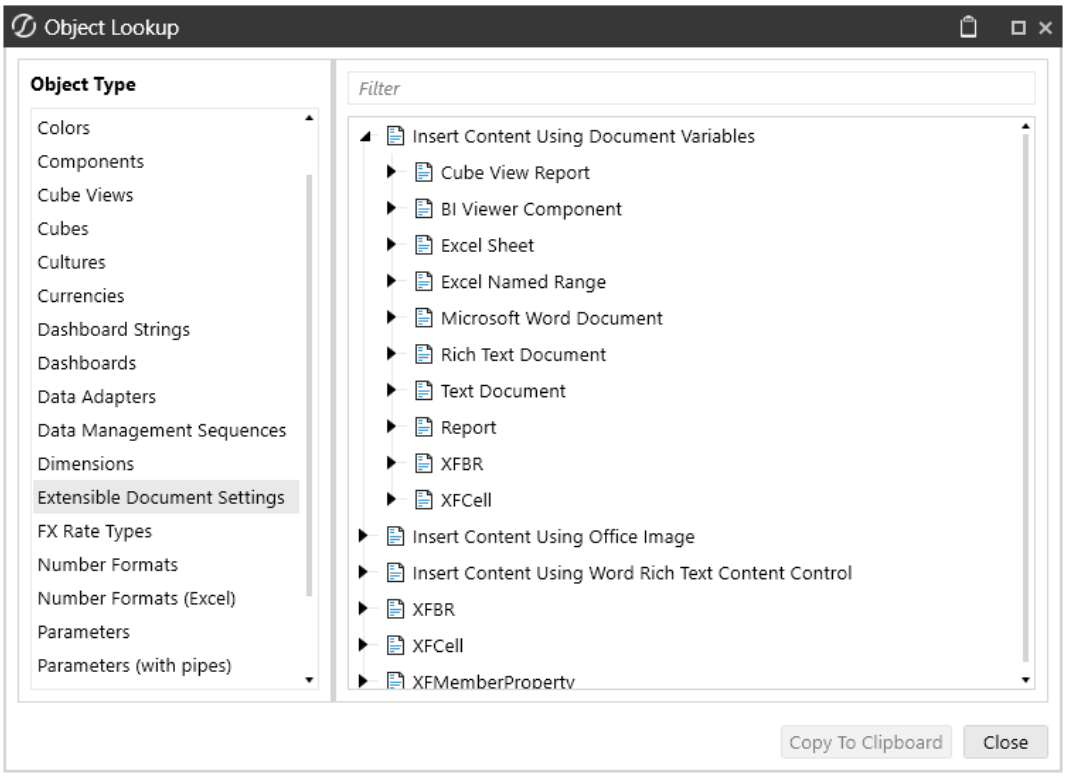
Refreshable content can be embedded in Text Editor using document variables. This enables you to edit and narrate a document to be published into a report.

- Insert a Cube View in Text Editor
- Excel Sheets and Excel Named Ranges in Text Editor
- Insert a Microsoft Word Document in Text Editor
- Insert a Rich Text Document in Text Editor
- Insert a Text Document in Text Editor
- Insert a Report in Text Editor

> **NOTE:** Text Editor contains the OneStream ribbon. See Text Editor Ribbon.

To insert content using document variables, copy all syntax directly from the Object Lookup. To insert content using document variables, copy all syntax directly from the Object Lookup.

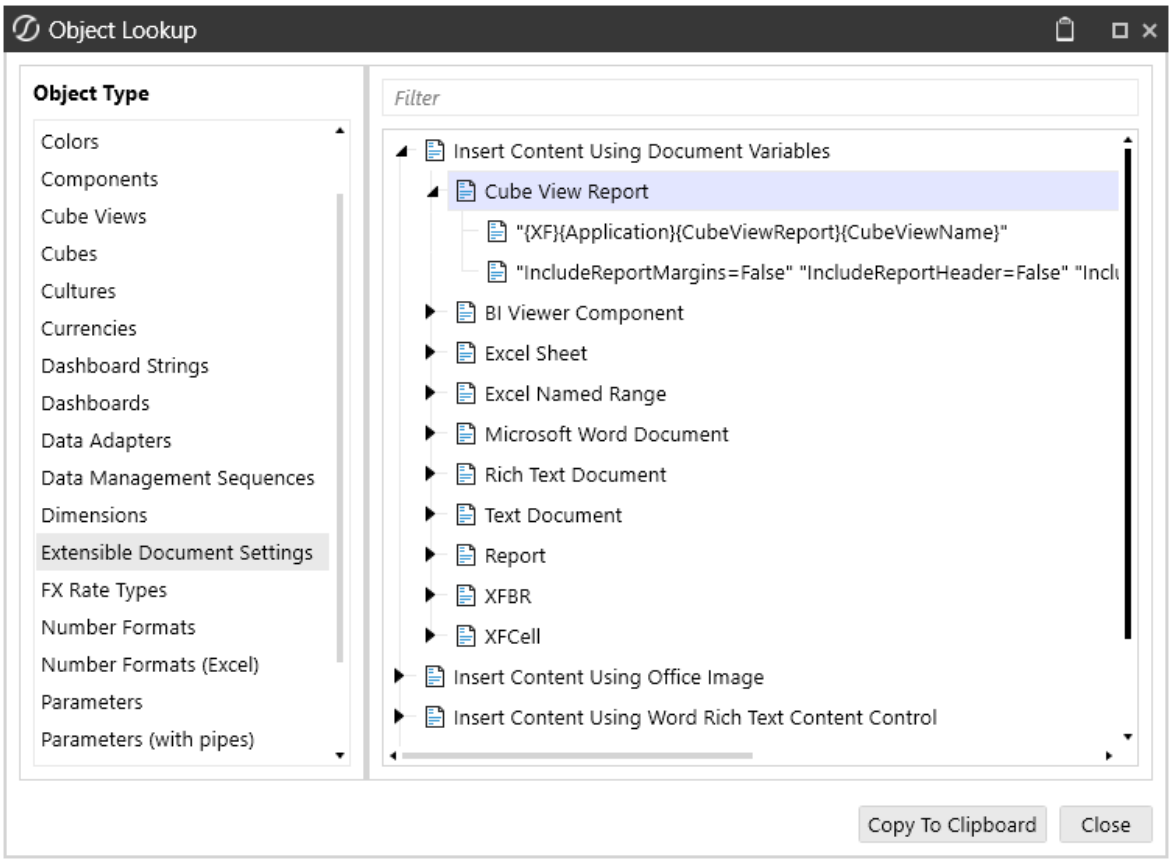
Navigate to the **Object Lookup**  icon and open the Object Lookup dialog box. This dialog box provides the syntax needed to insert content into an extensible document. This icon can be found under the Application tab on the following pages: Form Templates, Books, Cube Views, Workspaces, and Data Management.



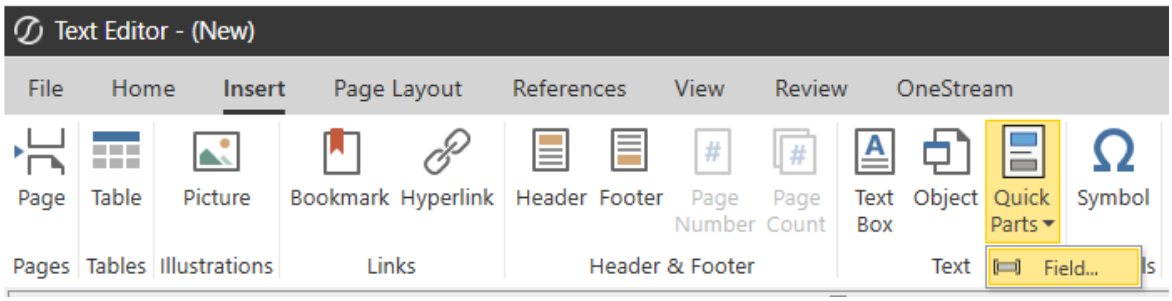
See Extensible Document Settings.

Insert a Cube View in Text Editor

1. Open the **Object Lookup** dialog box and select **Extensible Document Settings**. Expand **Insert Content Using Document Variables** and then expand **Cube View Report**. Select syntax line and then select the **Copy to Clipboard** button to copy the string.



2. Go to **Application** > **Tools** > **Text Editor**. Begin with a new or existing file.
3. To inset your string, go to **Insert** > **Quick Parts** > **Field**.



4. The **Doc Variable Name and Arguments** dialog box displays. Highlight the placeholder text and paste the copied syntax.



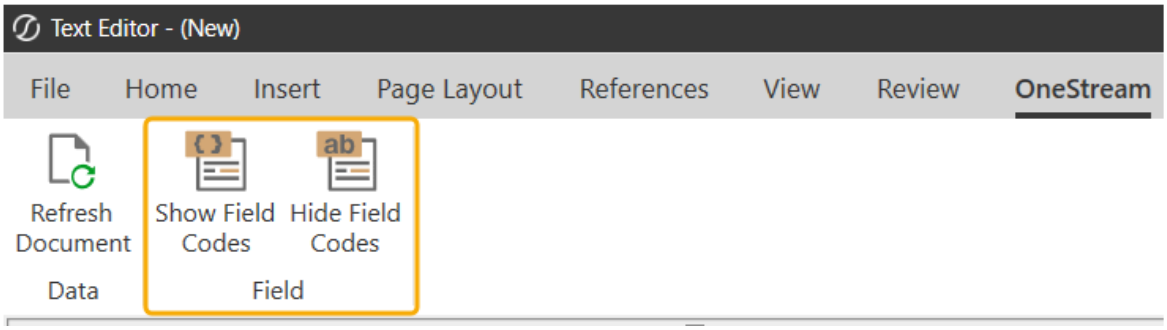
   Optionally, Cube View report arguments, such as IncludeReportMargins=False, can be pasted in the "Argument1" "Argument2" sections. See Extensible Document Settings.

   > **TIP:** Under **Field Options**, select the **Preserve formatting during updates** checkbox. See Document Variable Setup.

5. Update the values to reflect the Cube View you are adding to the report. Select the **OK** button. The final string displays in Text Editor.

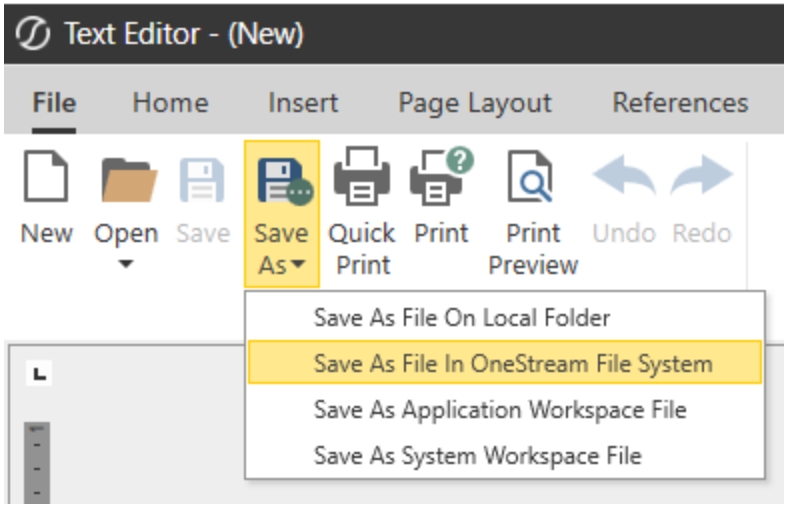
   > **Example:** {DOCVARIABLE "{XF}{Application}{CubeViewReport}{IncomeStatement}

6. Place your cursor on the page and press **Alt** + **F9**. This enables the selected value to be shown or hidden on the page. You can also use the **Show Field Codes** and **Hide Field Codes** buttons in the OneStream ribbon to show or hide field codes such as document variables.



> **TIP:** If pressing **Alt** + **F9** does not work, press **Alt** + **Fn** + **F9** simultaneously.

7. Select **File** > **Save As** > **Save As File in OneStream File System**.



8. Select a specific folder in File Explorer, then enter a file name and add the .xfdoc extension. The file name is appended with .xfdoc.docx.
9. In the OneStream ribbon, select the **Refresh Document** button.

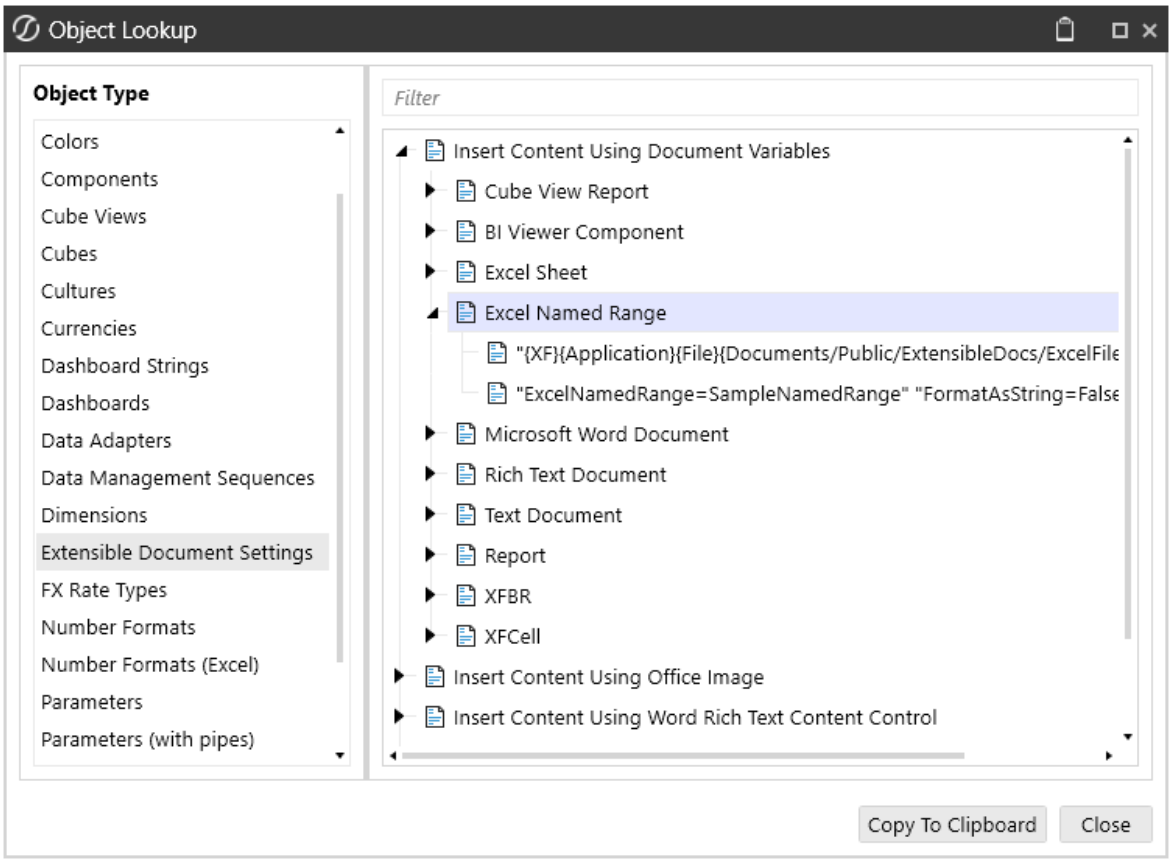


The Text Editor page refreshes and the embedded Cube View report is displayed.

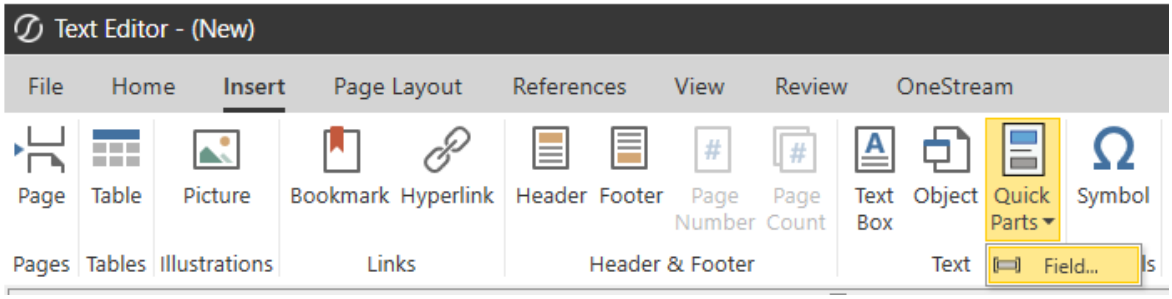
Excel Sheets and Excel Named Ranges in Text Editor

> **IMPORTANT:** Excel sheets and Excel Named Ranges can only have 500 or less rows and 50 or less columns. If you include an Argument for an Excel sheet or Excel Named Range that exceeds the maximum limit, you will receive an error message when attempting to process.

1. Open the **Object Lookup** dialog box and select **Extensible Document Settings**. Expand **Insert Content Using Document Variables** and then expand **Excel Named Range**. Select the syntax line and then select the **Copy to Clipboard** button to copy the string.



2. Go to **Application** > **Tools** > **Text Editor**. Begin with a new or existing file.
3. To inset your string, go to **Insert** > **Quick Parts** > **Field**.



4. The **Doc Variable Name and Arguments** dialog box displays. Highlight the placeholder text and paste the copied syntax.



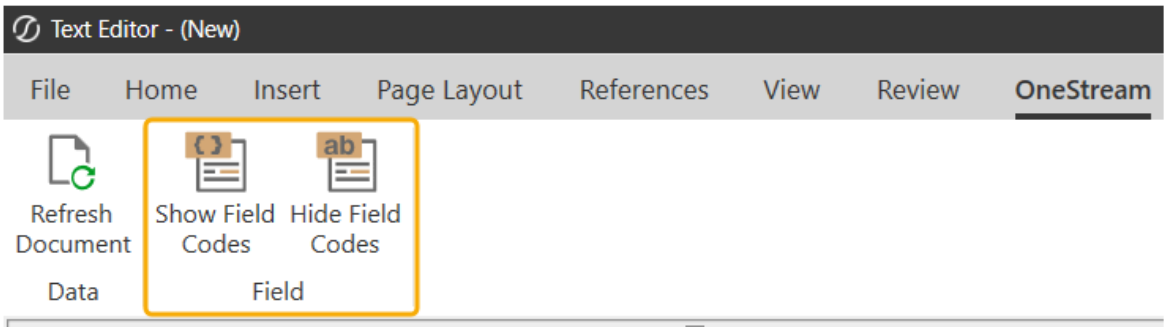
Optionally, the ExcelSheet=Sheet1 or ExcelNamedRange=SampleNamedRange arguments can be pasted in the "Argument1" "Argument2" sections. See Extensible Document Settings.

> **TIP:** Under **Field Options**, select the **Preserve formatting during updates** checkbox. See Document Variable Setup.

5. Update the values to reflect the folder structure and excel file path. Select the **OK** button. The string displays in Text Editor.

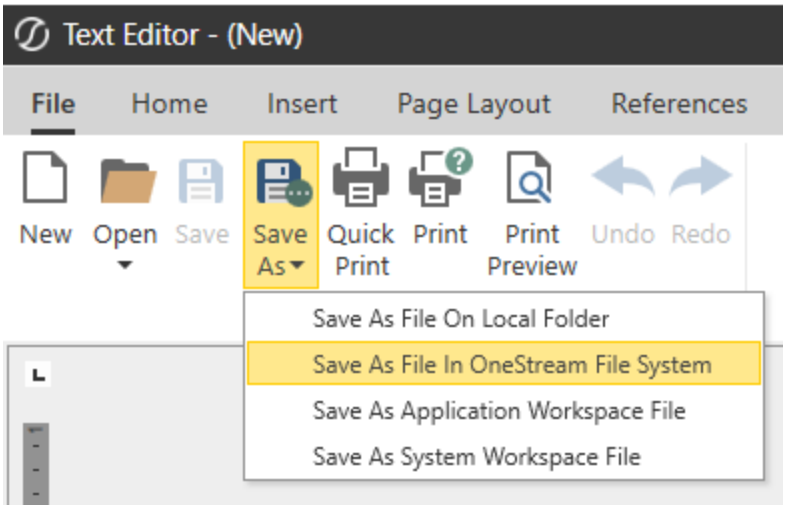
> **Example:** {DOCVARIABLE "{XF}{Application}{File}{Documents/Public/Extensible Documents/ExcelTextEditor.xfDoc.xlsx}" "ExcelSheet=Sheet2"

6. Place your cursor on the page and press **Alt** + **F9**. This enables the selected value to be shown or hidden on the page. You can also use the **Show Field Codes** and **Hide Field Codes** buttons in the OneStream ribbon to show or hide field codes such as document variables.



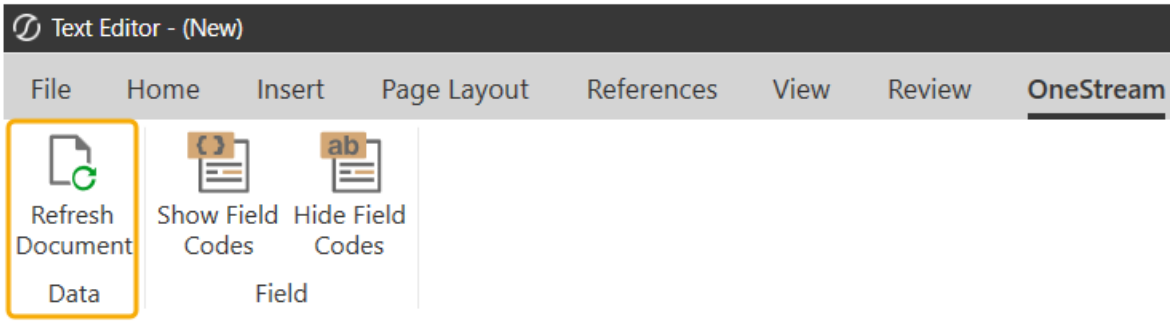
> **TIP:** If pressing **Alt** + **F9** does not work, press **Alt** + **Fn** + **F9** simultaneously.

7. Select **File** > **Save As** > **Save As File in OneStream File System**.

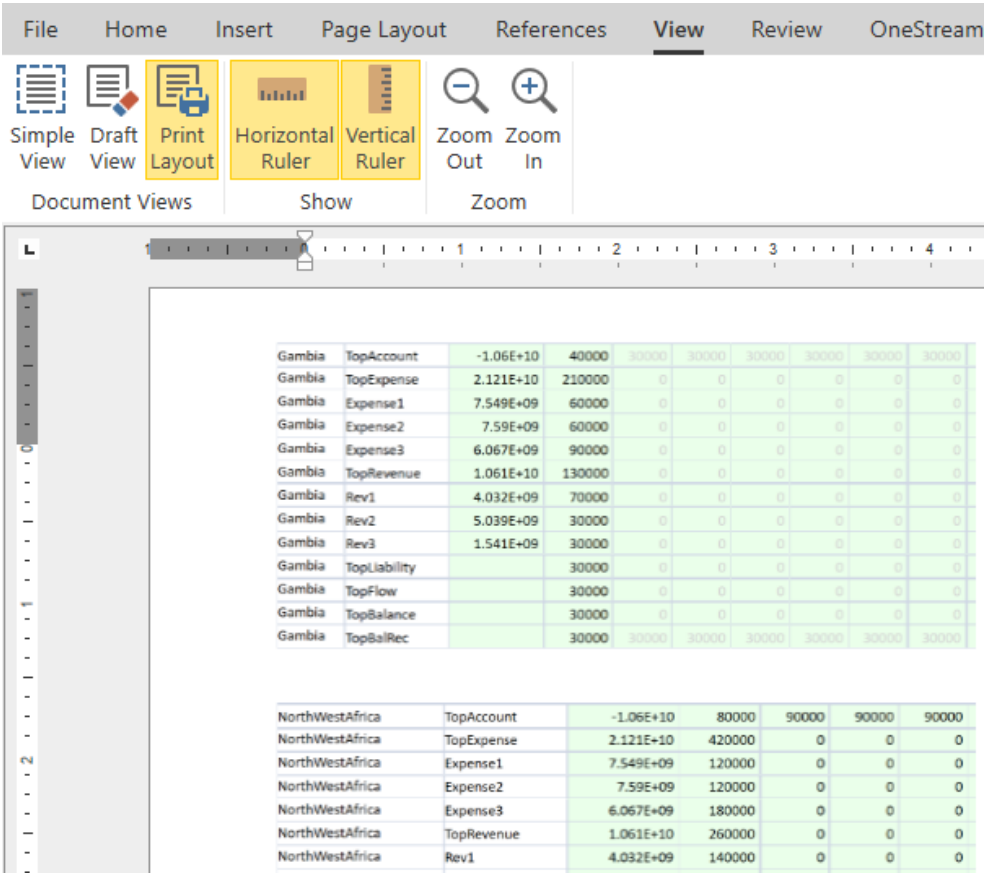


8. Select a specific folder in File Explorer, then enter a file name and add the .xfdoc extension. The file name is appended with .xfdoc.docx.

9. In the OneStream ribbon, select the **Refresh Document** button.



The Text Editor page refreshes and the Excel Sheet or Excel Named Range content displays in place of the document variables.

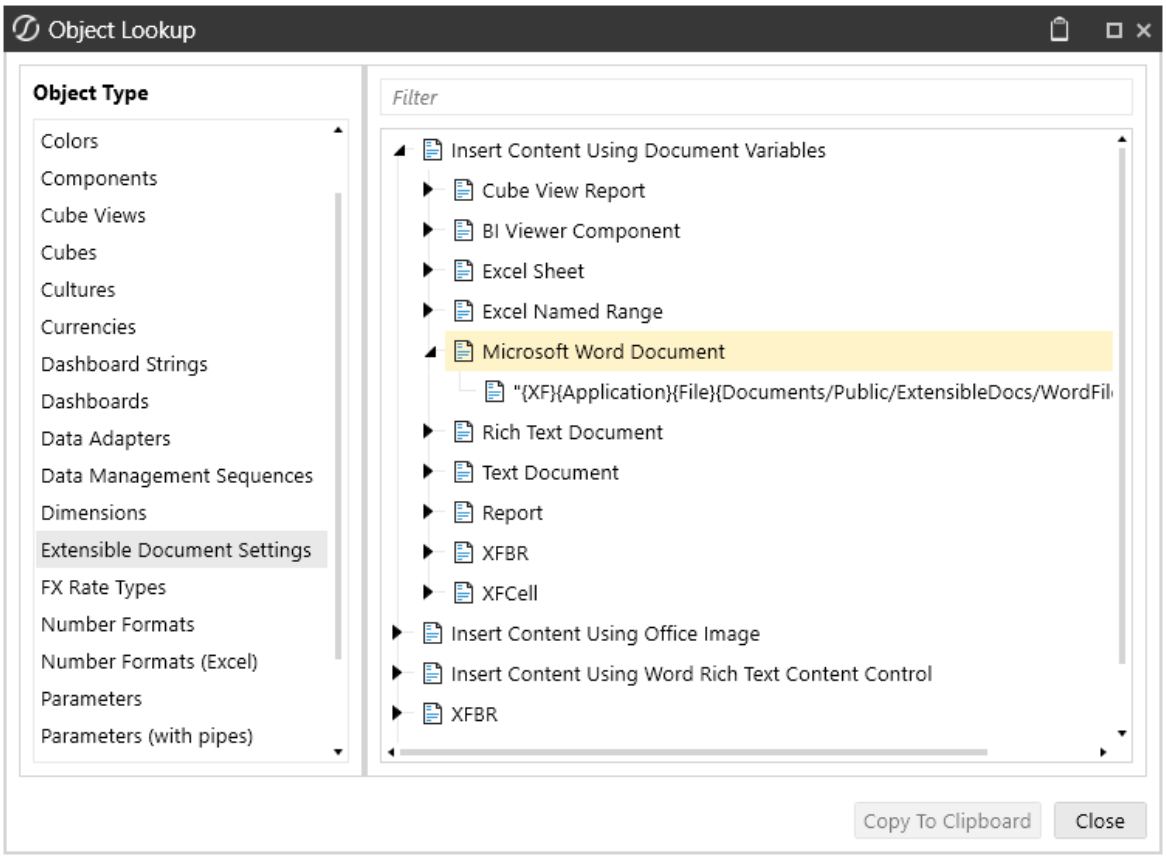


Gambia	TopAccount	-1.06E+10	40000	30000	30000	30000	30000	30000	30000
Gambia	TopExpense	2.121E+10	210000	0	0	0	0	0	0
Gambia	Expense1	7.549E+09	60000	0	0	0	0	0	0
Gambia	Expense2	7.59E+09	60000	0	0	0	0	0	0
Gambia	Expense3	6.067E+09	90000	0	0	0	0	0	0
Gambia	TopRevenue	1.061E+10	130000	0	0	0	0	0	0
Gambia	Rev1	4.032E+09	70000	0	0	0	0	0	0
Gambia	Rev2	5.039E+09	30000	0	0	0	0	0	0
Gambia	Rev3	1.541E+09	30000	0	0	0	0	0	0
Gambia	TopLiability		30000	0	0	0	0	0	0
Gambia	TopFlow		30000	0	0	0	0	0	0
Gambia	TopBalance		30000	0	0	0	0	0	0
Gambia	TopBalRec		30000	30000	30000	30000	30000	30000	30000

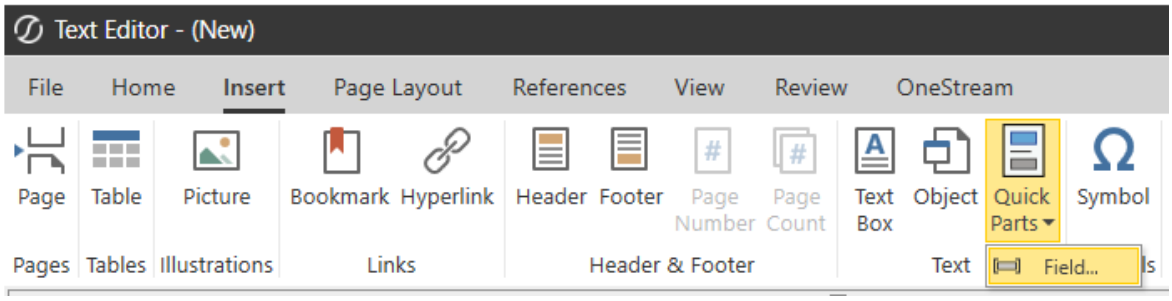
NorthWestAfrica	TopAccount	-1.06E+10	80000	90000	90000	90000
NorthWestAfrica	TopExpense	2.121E+10	420000	0	0	0
NorthWestAfrica	Expense1	7.549E+09	120000	0	0	0
NorthWestAfrica	Expense2	7.59E+09	120000	0	0	0
NorthWestAfrica	Expense3	6.067E+09	180000	0	0	0
NorthWestAfrica	TopRevenue	1.061E+10	260000	0	0	0
NorthWestAfrica	Rev1	4.032E+09	140000	0	0	0

Insert a Microsoft Word Document in Text Editor

1. Open the **Object Lookup** dialog box and select **Extensible Document Settings**. Expand **Insert Content Using Document Variables** and expand **Microsoft Word Document**. Select the syntax line and then select the **Copy to Clipboard** button to copy the string.



2. Go to **Application** > **Tools** > **Text Editor**. Begin with a new or existing file.
3. To inset your string, go to **Insert** > **Quick Parts** > **Field**.



4. The **Doc Variable Name and Arguments** dialog box displays. Highlight the placeholder text and paste the copied syntax.



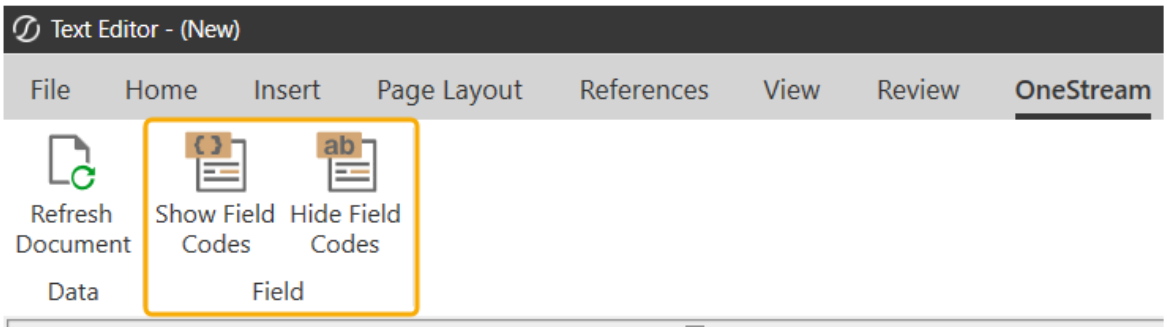
See Extensible Document Settings.

> **TIP:** Under **Field Options**, select the **Preserve formatting during updates** checkbox. See Document Variable Setup.

5. Update the values to reflect the folder structure and the word document path. Select the **OK** button. The final string displays in Text Editor.

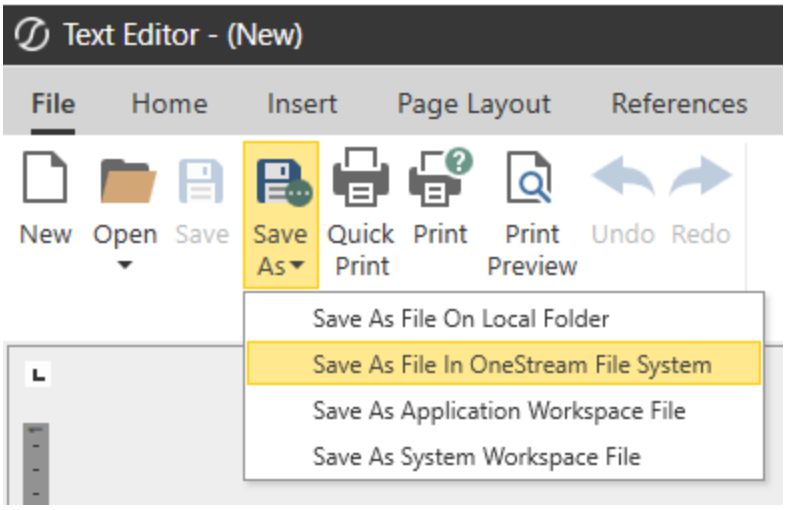
> **Example:** {DOCVARIABLE "{XF}{Application}{File}{Documents/Public/Extensible Documents/OneStreamWordDocument.docx}"}

6. Place your cursor on the page and press **Alt** + **F9**. This enables the selected value to be shown or hidden on the page. You can also use the **Show Field Codes** and **Hide Field Codes** buttons in the OneStream ribbon to show or hide field codes such as document variables.



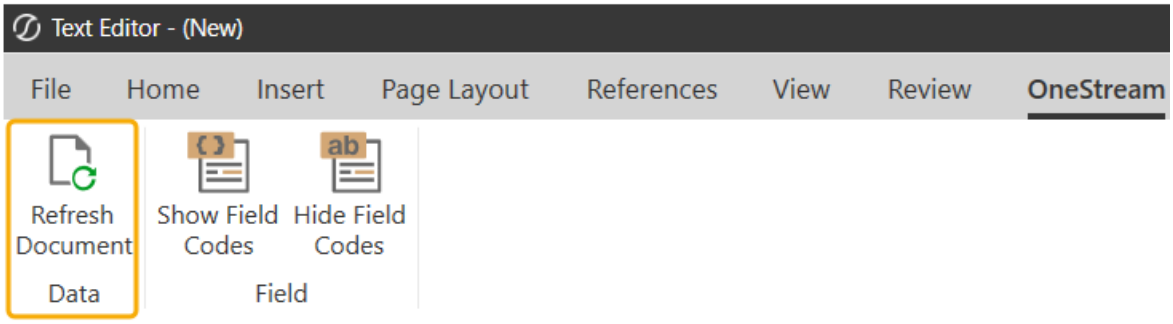
> **TIP:** If pressing **Alt** + **F9** does not work, press **Alt** + **Fn** + **F9** simultaneously.

7. Select **File** > **Save As** > **Save As File in OneStream File System**.



8. Select a specific folder in File Explorer, then enter a name and add the .xfdoc extension. The file name is appended with .xfdoc.docx.

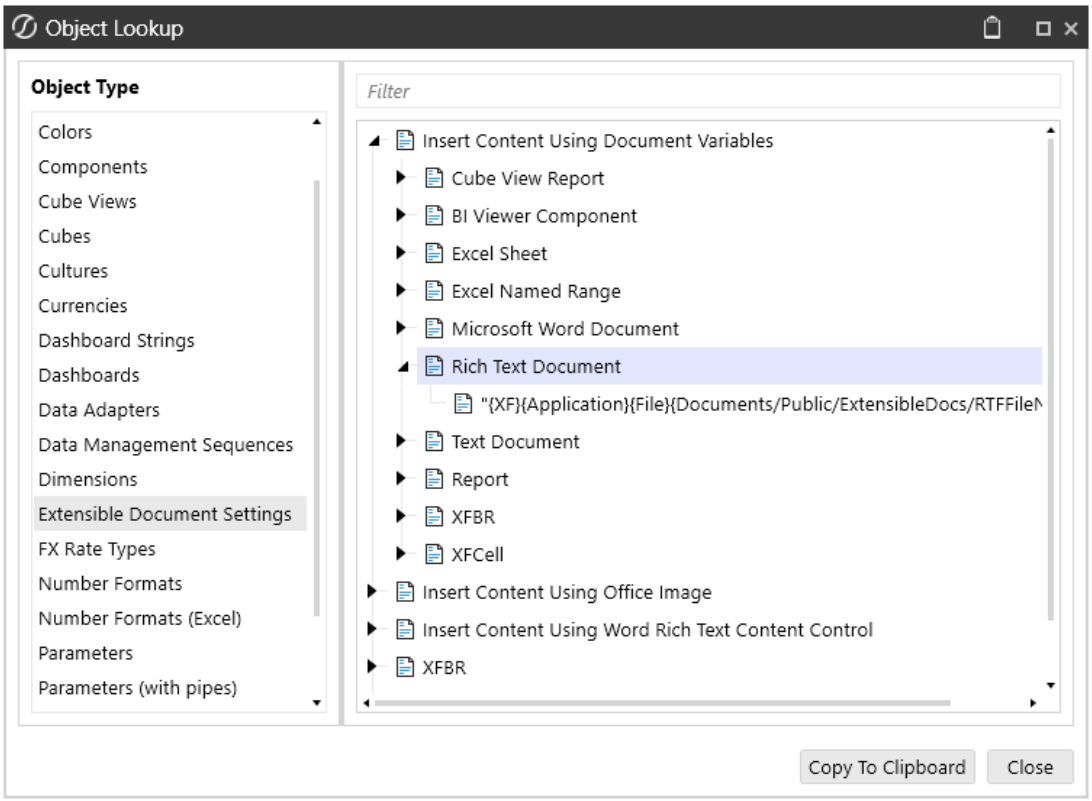
9. In the OneStream ribbon, select the **Refresh Document** button.



The Text Editor page refreshes and the embedded Microsoft Word document is displayed.

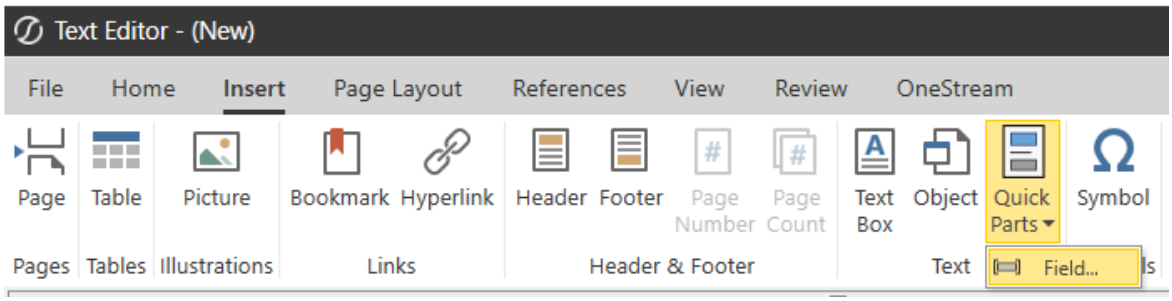
Insert a Rich Text Document in Text Editor

1. Open the **Object Lookup** dialog box and select **Extensible Document Settings**. Expand **Insert Content Using Document Variables** and then expand **Rich Text Document**. Select the syntax and then select the **Copy to Clipboard** button to copy the string.



2. Go to **Application** > **Tools** > **Text Editor**. Begin with a new or existing file.

3. To inset your string, go to **Insert** > **Quick Parts** > **Field**.



4. The **Doc Variable Name and Arguments** dialog box displays. Highlight the placeholder text and paste the copied syntax.



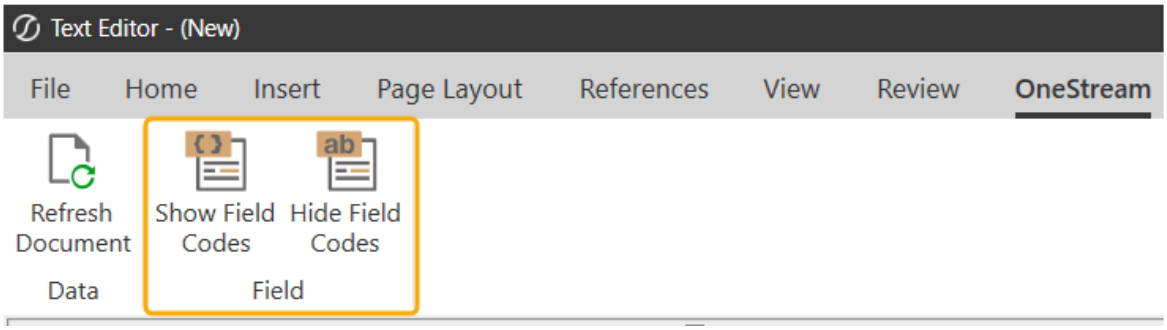
See Extensible Document Settings.

> **TIP:** Under **Field Options**, select the **Preserve formatting during updates** checkbox. See Document Variable Setup.

5. Update the values to reflect the rich text format file path. Select the **OK** button. The final string displays in Text Editor.

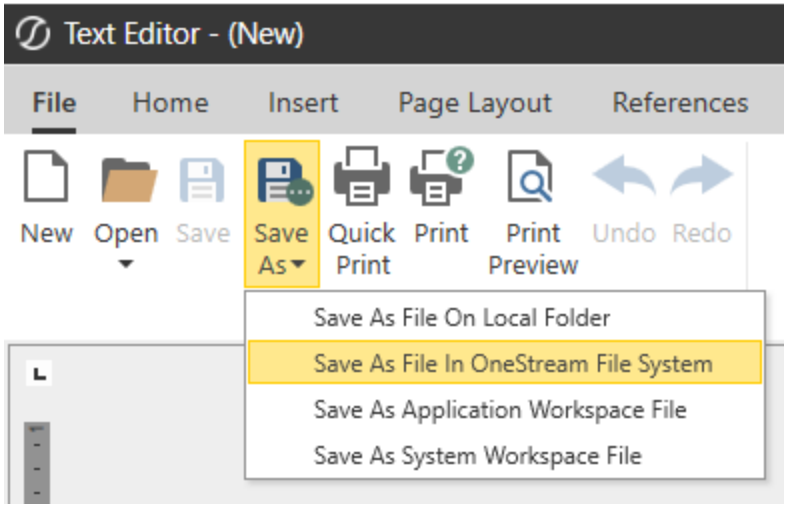
   > **Example:** {DOCVARIABLE "{XF}{Application}{File}{Documents/Public/Extensible Documents/RichTextFormatDoc.rtf}"}

6. Place your cursor on the page and press **Alt** + **F9**. This enables the selected value to be shown or hidden on the page. You can also use the **Show Field Codes** and **Hide Field Codes** buttons in the OneStream ribbon to show or hide field codes such as document variables.



> **TIP:** If pressing **Alt** + **F9** does not work, press **Alt** + **Fn** + **F9** simultaneously.

7. Select **File** > **Save As** > **Save As File in OneStream File System**.



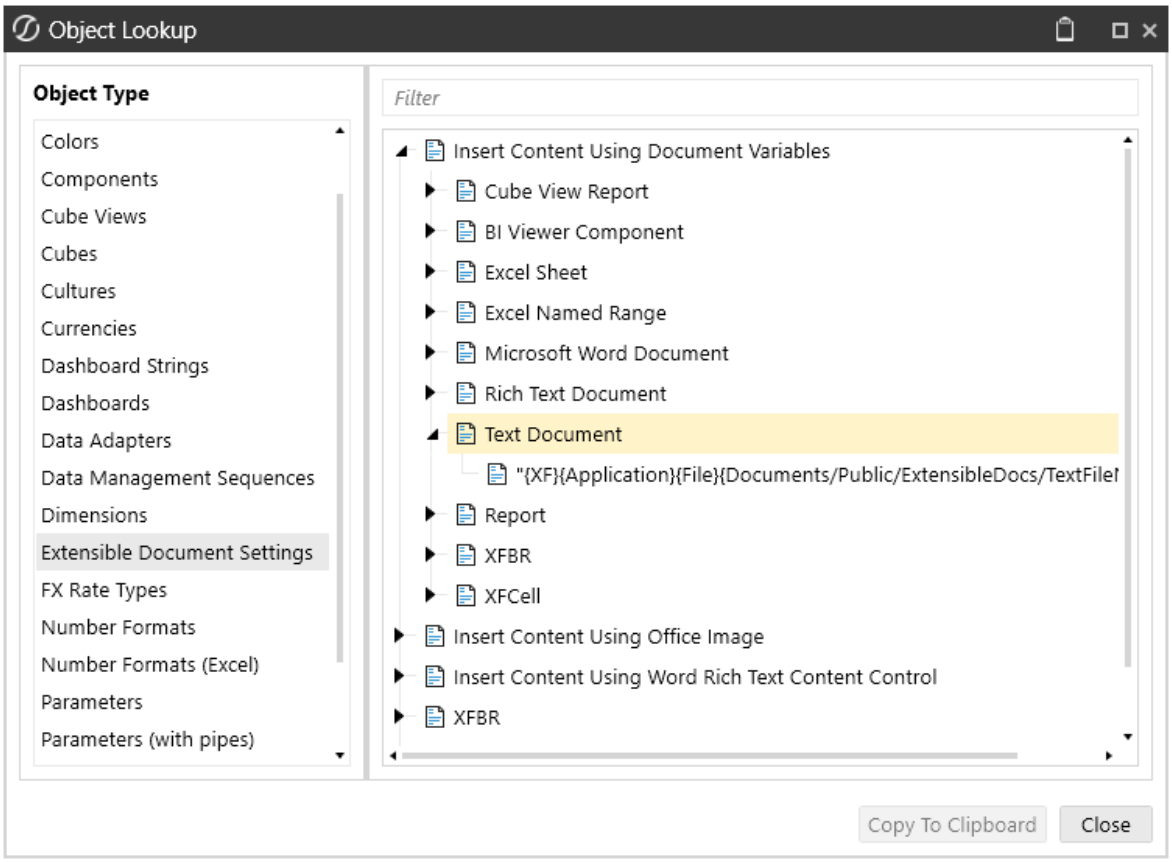
8. Select a specific folder in File Explorer, then enter a file name and add the .xfdoc extension. The file name is appended with .xfdoc.docx.
9. In the OneStream ribbon, select the **Refresh Document** button.



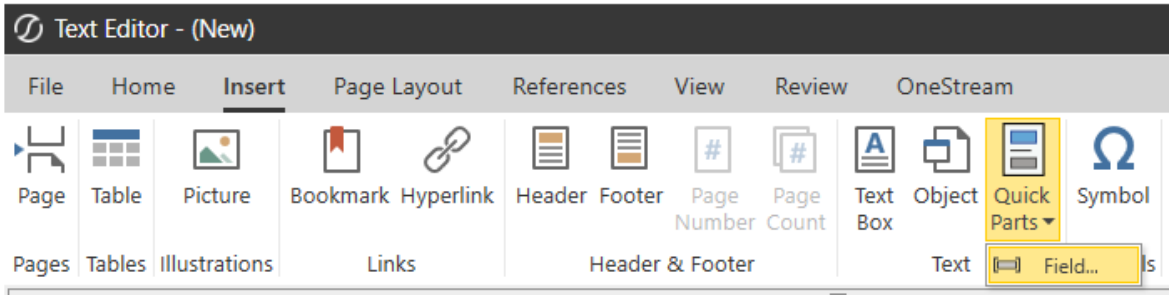
The Text Editor page refreshes and the embedded Rich Text document is displayed.

Insert a Text Document in Text Editor

1. Open the **Object Lookup** the dialog box and select **Extensible Document Settings**. Expand **Insert Content using Document Variables** and then expand **Text Document**. Select the syntax line and then select the **Copy to Clipboard** button to copy the string.



2. Go to **Application** > **Tools** > **Text Editor**. Begin with a new or existing file.
3. To inset your string, go to **Insert** > **Quick Parts** > **Field**.



4. The **Doc Variable Name and Arguments** dialog box displays. Highlight the placeholder text and paste the copied syntax.



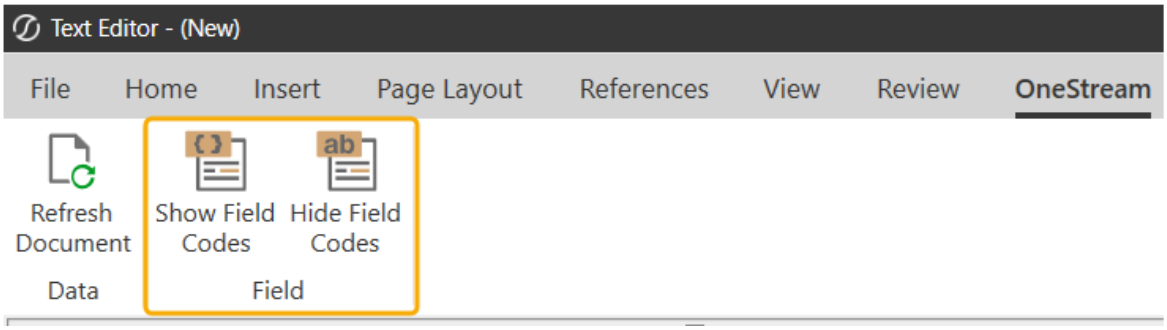
See Extensible Document Settings.

> **TIP:** Under **Field Options**, select the **Preserve formatting during updates** checkbox. See Document Variable Setup.

5. Update the values to reflect the folder structure and the text document path. Select the **OK** button. The final string displays in Text Editor.

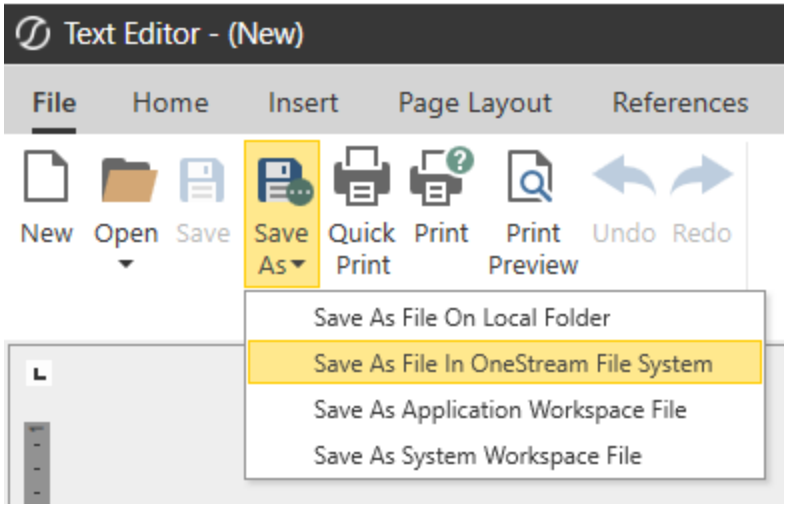
> **Example:** {DOCVARIABLE "{XF}{Application}{File}{Documents/Public/Extensible Documents/Testtoupdate.txt}"}

6. Place your cursor on the page and press **Alt** + **F9**. This enables the selected value to be shown or hidden on the page. You can also use the **Show Field Codes** and **Hide Field Codes** buttons in the OneStream ribbon to show or hide field codes such as document variables.



> **TIP:** If pressing **Alt** + **F9** does not work, press **Alt** + **Fn** + **F9** simultaneously.

7. Select **File** > **Save As** > **Save As File in OneStream File System**.



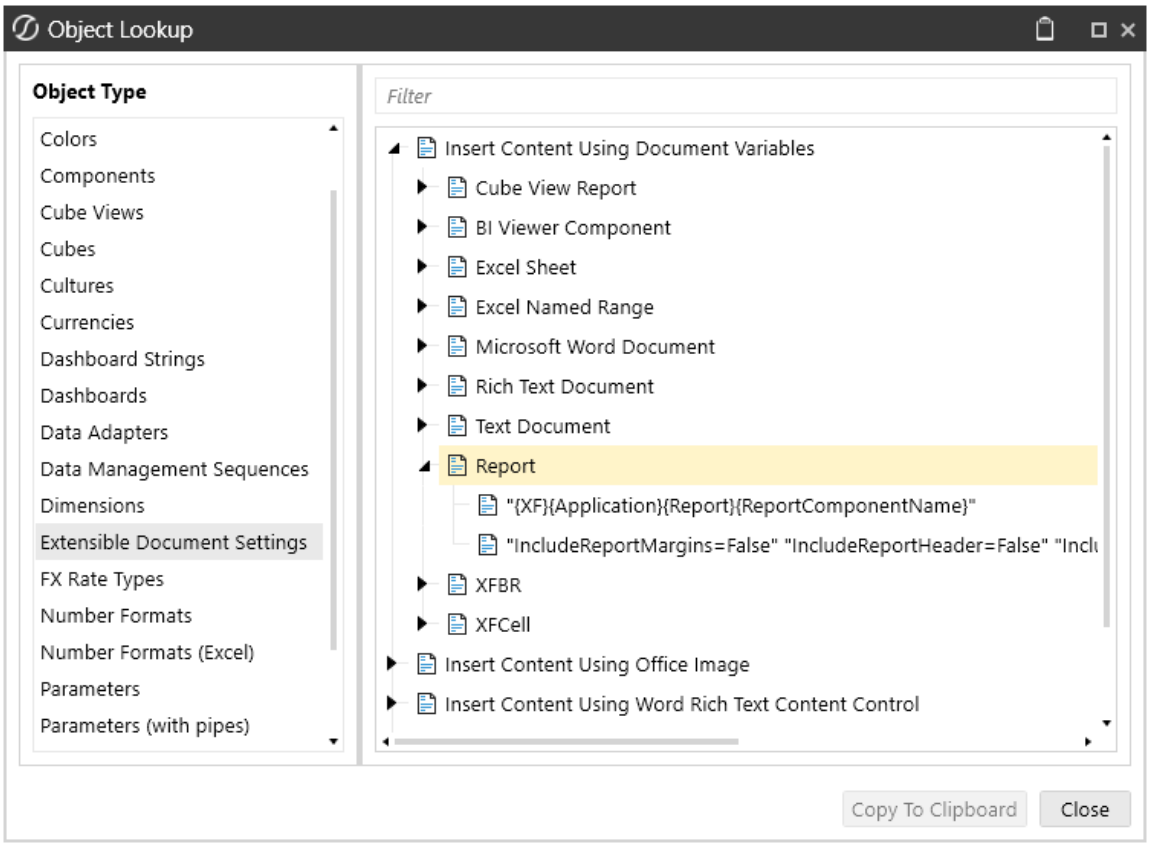
8. Select a specific folder in the File Explorer, then enter a name and add the .xfdoc extension. The file name is appended with .xfdoc.docx.
9. In the OneStream ribbon, select the **Refresh Document** button.



The Text Editor page refreshes and the embedded text document is displayed.

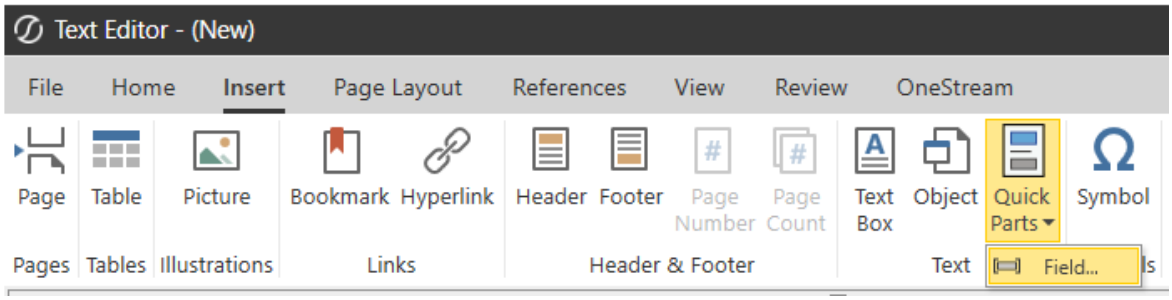
Insert a Report in Text Editor

1. Open the **Object Lookup** dialog box, select **Extensible Document Settings**. Expand **Insert Content using Document Variables** and then expand **Report**. Select a syntax line and then select the **Copy to Clipboard** button to copy the string.



2. Go to **Application** > **Tools** > **Text Editor**. Begin with a new or existing file.

3. To inset your string, go to **Insert** > **Quick Parts** > **Field**.



4. The **Doc Variable Name and Arguments** dialog box displays. Highlight the placeholder text and paste the copied syntax.



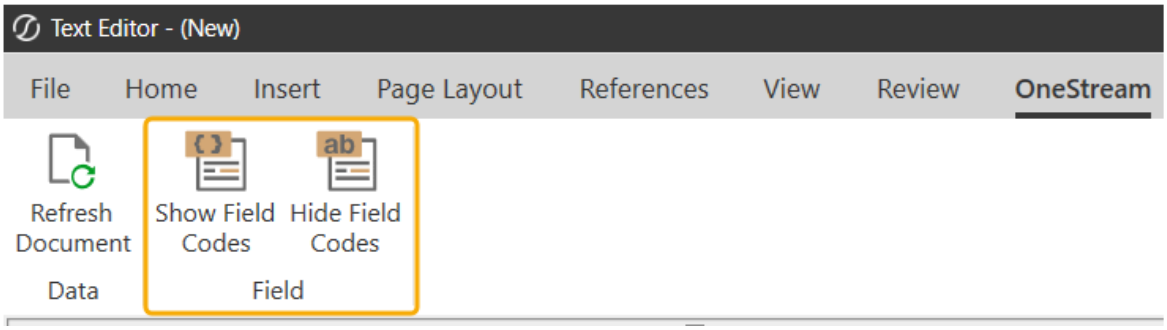
   Optionally, report arguments, such as IncludeReportMargins=False, can be pasted in the"Argument1" "Argument2" sections. See Extensible Document Settings.

   > **TIP:** Under **Field Options**, select the **Preserve formatting during updates** checkbox. See Document Variable Setup.

5. Update the values to reflect the report you are adding. Select the **OK** button. The final string displays in Text Editor.

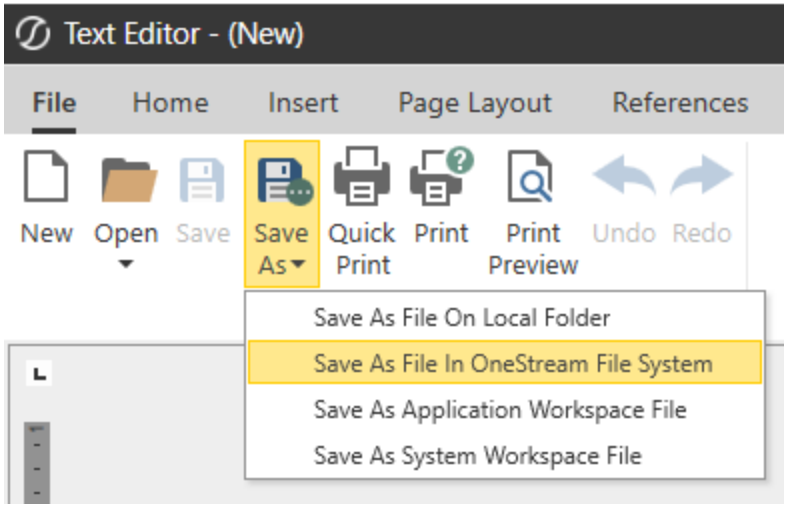
   > **Example:** {DOCVARIABLE "{XF}{Application}{Report}{Report3}"}

6. Place your cursor on the page and press **Alt** + **F9**. This enables the selected value to be shown or hidden on the page. You can also use the **Show Field Codes** and **Hide Field Codes** buttons in the OneStream ribbon to show or hide field codes such as document variables.



> **TIP:** If pressing **Alt** + **F9** does not work, press **Alt** + **Fn** + **F9** simultaneously.

7. Select **File** > **Save As** > **Save As File in OneStream File System**.



8. Select a specific folder in File Explorer, then enter a file name and add the .xfdoc extension. The file name is appended with .xfdoc.docx.
9. In the OneStream ribbon, select the **Refresh Document** button.

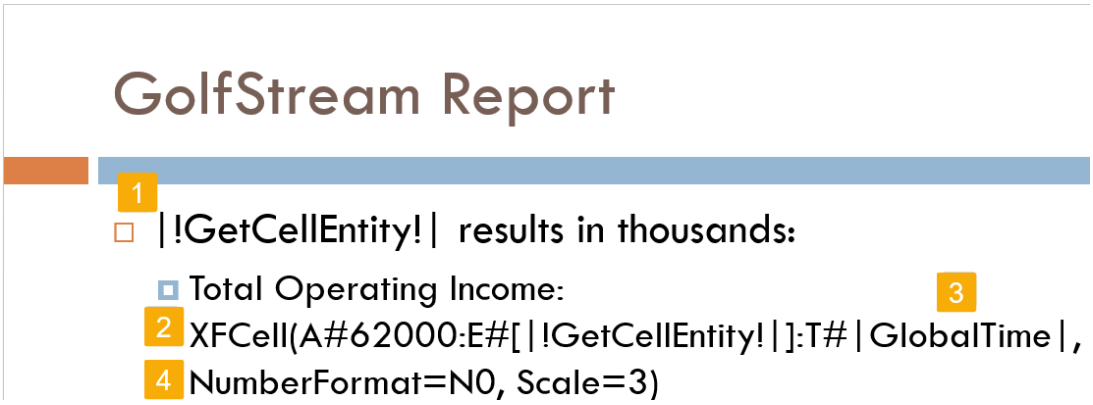


The Text Editor page refreshes and the embedded report is displayed.

Creating a Document in Microsoft PowerPoint

This example shows a slide from a management presentation and demonstrates how to use parameters, substitution variables, and the XFCell retrieve function.

> **NOTE:** The same format for XFCell is used in Microsoft Word and text files.

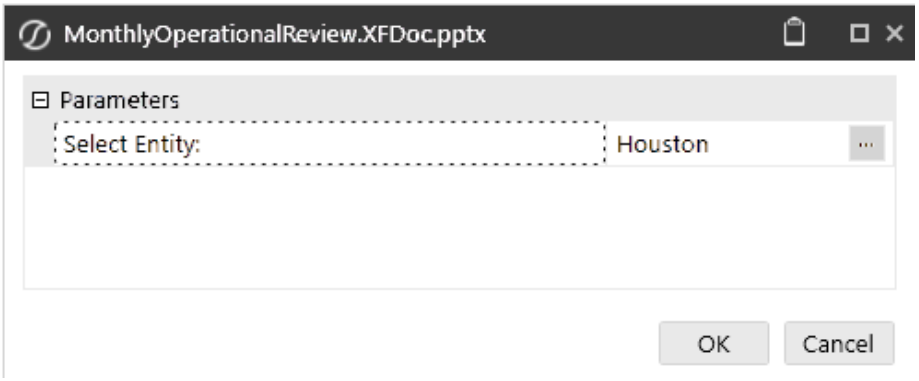


1. A parameter named |!GetCellEntity!| enables the user to select a specific entity to base data on. When the extensible document is processed at run-time, the user is prompted to select the entity.
2. XFCell retrieves data from a single cell in OneStream. In this example, data is retrieved from Account 62000 for the entity selected and the application's global time period.

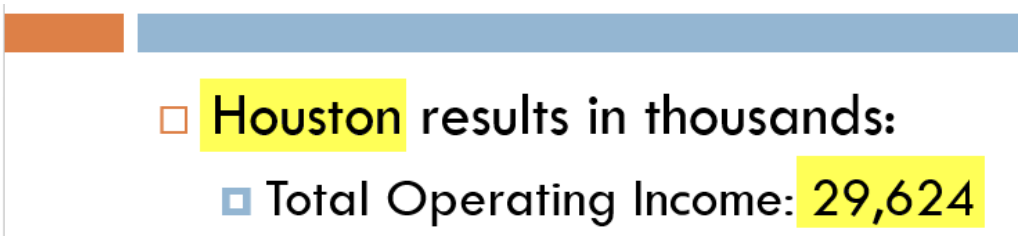
   > **NOTE:** See Extensible Document Settings for more details on XFCell syntax. You can also copy and paste XFCell syntax from a Cube View's Cell POV Information dialog box into a text file. See "OnePlace Cube Views" in the *Getting Started with OneStream Guide*.

3. A substitution variable calls the application's global time, which updates with the Global Time period currently set in the application.
4. Additional format settings can be included to control options such as number formatting or scaling.

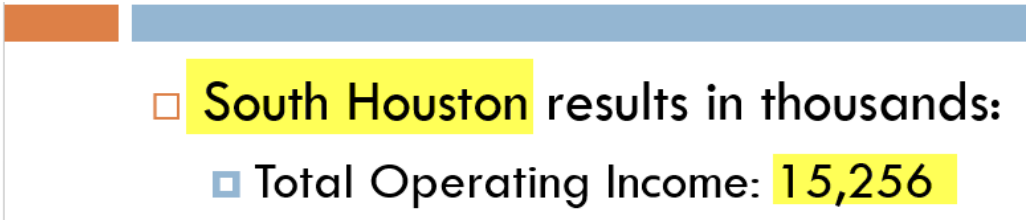
When the document is processed, the user is prompted to select an entity.



The document displays updated data for the selected entity. In this example, Houston has replaced the parameter, and 29,624 is the XFCell function result.

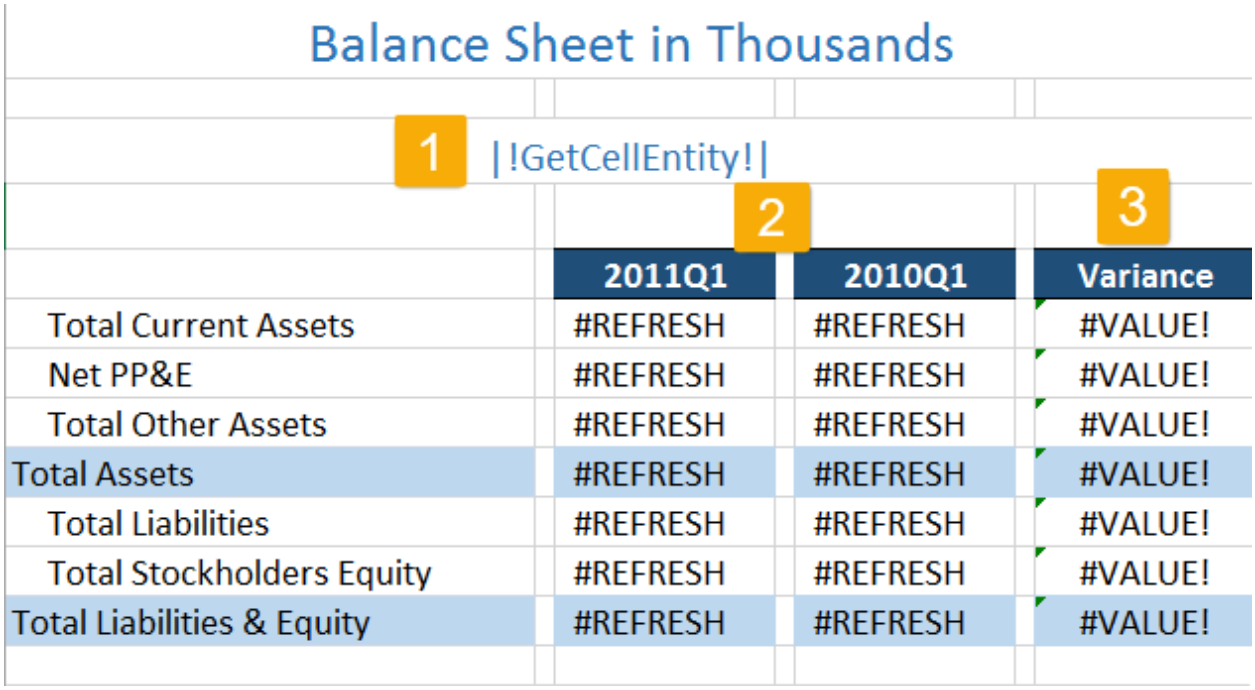


Data refreshes and updates when you select a different entity.



Creating a Document in Microsoft Excel

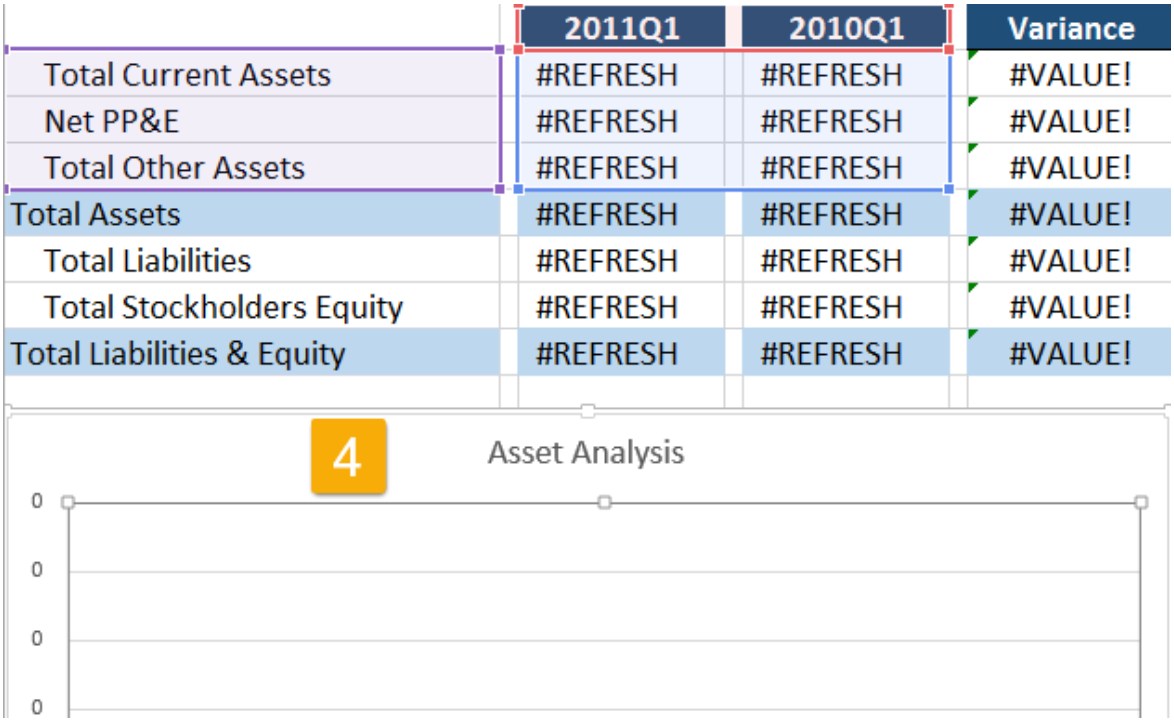
This example is a summary balance sheet built in Excel that demonstrates how to use parameters, the XFGetCellVolatile retrieve function, and the Excel If function. It also incorporates an Excel chart derived from the functions in an Excel spreadsheet.



	2011Q1	2010Q1	Variance
Total Current Assets	#REFRESH	#REFRESH	#VALUE!
Net PP&E	#REFRESH	#REFRESH	#VALUE!
Total Other Assets	#REFRESH	#REFRESH	#VALUE!
Total Assets	#REFRESH	#REFRESH	#VALUE!
Total Liabilities	#REFRESH	#REFRESH	#VALUE!
Total Stockholders Equity	#REFRESH	#REFRESH	#VALUE!
Total Liabilities & Equity	#REFRESH	#REFRESH	#VALUE!

1. A parameter named |!GetCellEntity!| enables the user to select a specific entity to base data on. When the extensible document is processed at run-time, the user is prompted to select the entity.
2. The XFGetCellVolatile function retrieves specific data from OneStream and updates the Excel chart when the data is refreshed. A volatile function is required in Excel to properly refresh when using charts that reference calculated cells. This XFGetCellVolatile formula derives from the |!GetCellEntity!| parameter and displays updated data when an entity is selected. See the *Excel Add-In Guide* for details on retrieve functions.
3. The IF Excel function derives from the XFGetCellVolatile function. This data updates when the spreadsheet launches from OneStream and the data refreshes. Review the following IF formula example:

```
=IF((F6=0),"", (D6-F6)/F6)
```



	2011Q1	2010Q1	Variance
Total Current Assets	#REFRESH	#REFRESH	#VALUE!
Net PP&E	#REFRESH	#REFRESH	#VALUE!
Total Other Assets	#REFRESH	#REFRESH	#VALUE!
Total Assets	#REFRESH	#REFRESH	#VALUE!
Total Liabilities	#REFRESH	#REFRESH	#VALUE!
Total Stockholders Equity	#REFRESH	#REFRESH	#VALUE!
Total Liabilities & Equity	#REFRESH	#REFRESH	#VALUE!

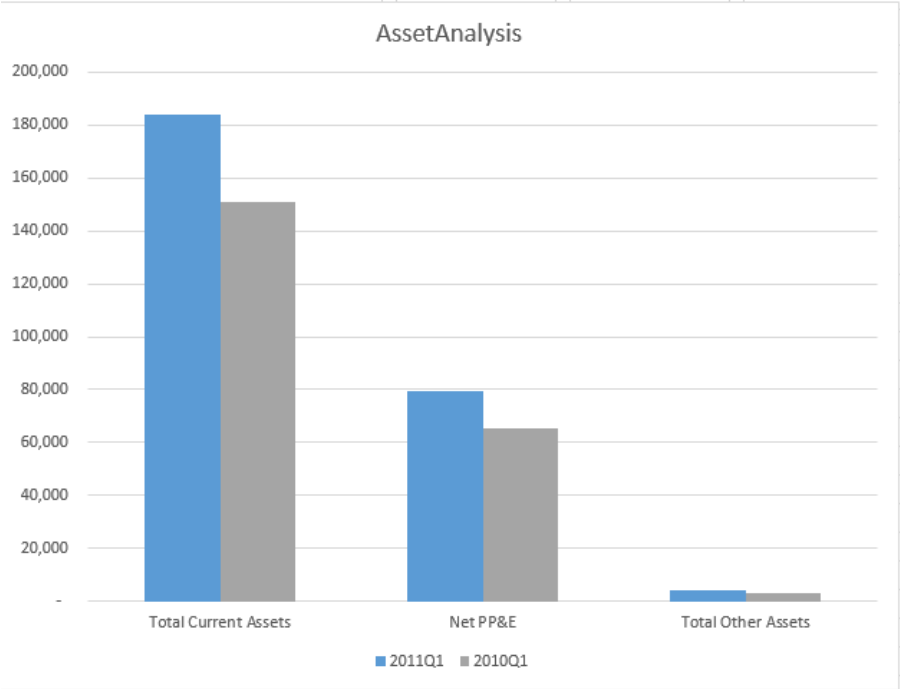
4. The data drives an Excel chart inserted into the spreadsheet. The chart display the correct values whenever it is refreshed.

When the document processes, the user is prompted to select an entity and the Excel spreadsheet is run. Log into the Excel Add-In and select **Refresh Data** to see updated values.

Balance Sheet in Thousands

Houston

	2011Q1	2010Q1	Variance
Total Current Assets	184,172	150,887	22.1%
Net PP&E	79,573	65,250	22.0%
Total Other Assets	3,867	3,171	22.0%
Total Assets	267,613	219,308	22.0%
Total Liabilities	196,919	207,433	(5.1%)
Total Stockholders Equity	70,694	11,876	495.3%
Total Liabilities & Equity	267,613	219,308	22.0%

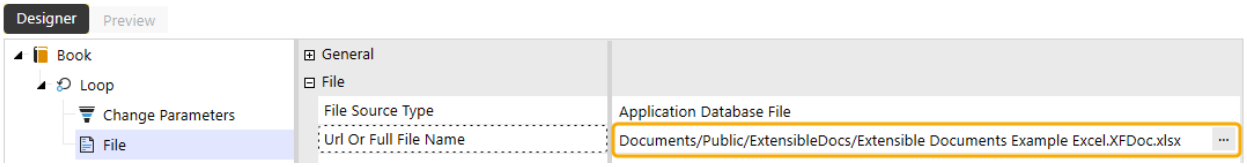


Using Extensible Documents in OneStream

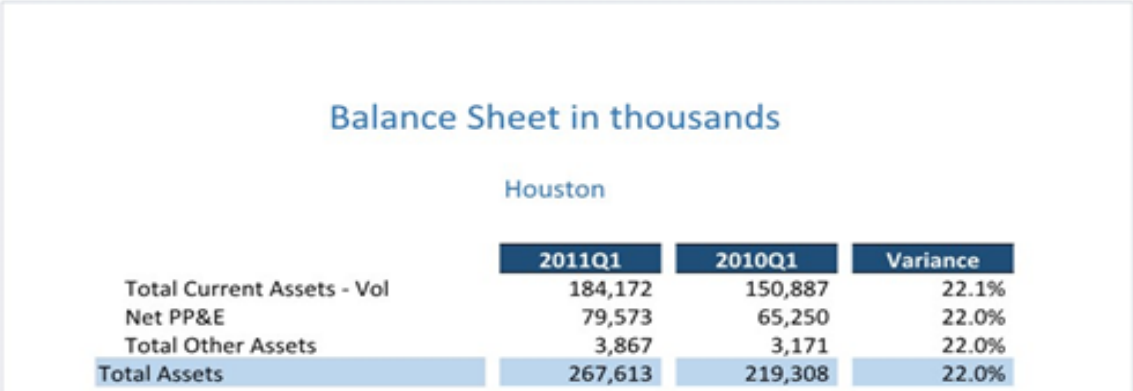
You can use extensible documents throughout OneStream. Each extensible document file is stored in the file share and can be kept private for specific users or placed into public folders for other users.

Extensible Documents in Report Books

Add extensible documents to report books by selecting a File book item and assigning the Extensible Document file.



The following image shows the assignment result:



Balance Sheet in thousands

Houston

	2011Q1	2010Q1	Variance
Total Current Assets - Vol	184,172	150,887	22.1%
Net PP&E	79,573	65,250	22.0%
Total Other Assets	3,867	3,171	22.0%
Total Assets	267,613	219,308	22.0%

You must have access to the Extensible Document to run the book or the file displays blank pages.

Extensible Documents in Dashboards

You can view Extensible Documents in dashboards by assigning them to a File Viewer component that opens and processes the Extensible Document at run-time. You can also view Extensible Documents assigned to a book that is assigned to a Book Viewer component. See Dashboard Components .

> **NOTE:** If a book contains an extensible Excel document using the XFGetCell function, you do not need to log in to the Excel Add-In to see updated values.

Extensible Documents in Data Management

To run Extensible Documents using a data management sequence, select the **Export File** step type and assign the specific Extensible Document file to it. The Extensible Document exports to the OneStream file share when this step processes. See "Data Management Steps" in the *Application Tools Guide*.

Extensible Document Limitations

There are some limitations to Extensible Documents when using different file types.

> **NOTE:** Extensible Documents also cannot reference themselves.

Text File Limitations

This file type does not support images.

Microsoft Word Limitations

In this file type, Word charts cannot be converted to PDF.

Microsoft PowerPoint Limitations

Review the following limitations:

- PowerPoint cannot be converted to PDF.
- When using the Rich Text Box component with any Extensible Document or dashboard, the following display format properties are ignored in the OneStream Windows application and Modern Browser Experience:
    - FontFamily
    - FontSize
    - Bold
    - Italic
    - TextColor

- BackgroundColor
- BorderColor

- You can process and open an extensible PowerPoint document that contains an Alt Text - document variable with MeasurementUnit, PageWidth, and PageHeight settings that are equal to < 1 inch.
- The out-of-range settings are automatically converted to the minimum values equivalent to MeasurementUnit=Inch, PageWidth=1, and PageHeight=1. This includes:
  - When MeasurementUnit=Document and PageWidth < 300 or PageHeight < 300.
  - When MeasurementUnit=Centimeter and PageWidth < 2.54 or PageHeight < 2.54.
  - When MeasurementUnit=Inch and PageWidth < 1 or PageHeight < 1.
  - When MeasurementUnit=Millimeter and PageWidth < 25.4 or PageHeight < 25.4.
  - When MeasurementUnit=Point and PageWidth < 72 or PageHeight < 72.

> **NOTE:** You can process and open an extensible PowerPoint document that contains a measurement equal to an integer, for example, MeasurementUnit=123. The integer is treated as an invalid value, such as MeasurementUnit=invalid, and is automatically converted to the default Inch setting.

Microsoft Excel Limitations

- You can only have 500 or less rows and 50 or less columns. If you include an Argument for an Excel sheet or Excel Named Range that exceeds the maximum limit, you will receive an error message when attempting to process.

- The center across columns function is not supported.
- Excel Spark lines is not supported.
- Word wrap is not supported.
- 3D charts are not supported.
- Modifications to colors of individual chart items in a series do not carry through to a PDF.
- Fitting columns to a page is not supported.
- Repeating columns and rows is not supported.
- Cell comments do not display in a PDF after processing.
- Background images do not display after processing in Excel or PDF.
- Shapes do not display after processing in Excel or PDF.
- Smart Art does not display after processing in Excel or PDF.
- Macros are not supported.

The following Excel functions are not supported in extensible documents:

- ACCRINT
- ACCRINTM
- AGGREGATE
- AMORDEGRC
- AMORLINC
- BAHTTEXT
- BETA.DIST
- BETA.INV

- BINOM.DIST
- BINOM.DIST.RANGE
- BINOM.INV
- BINOMDIST
- CHISQ.DIST
- CHISQ.DIST.RT
- CHISQ.INV
- CHISQ.INV.RT
- CHISQ.TEST
- CONFIDENCE.T
- COUPDAYBS
- COUPDAYS
- COUPDAYSNC
- COUPNCD
- COUPNUM
- COUPPCD
- COVARIANCE.P
- COVARIANCE.S
- CRITBINOM
- CUBEKIPIMEMBER
- CUBEMEMBER

- CUBEMEMBERPROPERTY
- CUBERANKEDMEMBER
- CUBESET
- CUBESETCOUNT
- CUBEVALUE
- DDB
- DEVSQ
- DISC
- DURATION
- EXPON.DIST
- EXPONDIST
- F.DIST
- F.DIST.RT
- F.INV
- F.INV.RT
- F.TEST
- FACTDOUBLE
- FILTERXML
- FISHER
- FISHERINV
- FORECAST

- FVSCHEDULE
- GAMMA
- GAMMA.DIST
- GAMMA.INV
- GAUSS
- GCD
- GETPIVOTDATA
- GROWTH
- HARMEAN
- HYPGEOM.DIST
- HYPGEOMDIST
- INTRATE
- ISPMT
- KURT
- LCM
- LIMEST
- LOGEST
- MDURATION
- MULTINOMINAL
- NEGBINOM.DIST
- NEGBINOMDIST

- ODDFPRICE
- OFFFYIELD
- ODDLPRICE
- ODDLYIELD
- PERCENTILE.EXC
- PERCENTILE.ING
- PERCENTILERANK.EXC
- PERCENTILERANK.INC
- PERMUT
- PERMUTATIONA
- PHI
- POISSON
- POISSON.DIST
- PRICE
- PRICEDISC
- PRICEMAT
- PROB
- QUARTILE.EXC
- QUARTILE.INC
- RANK.AVG
- RANK.EQ

- RECEIVED
- ROMAN
- RSQ
- RTD
- SERIESSUM
- SKEW.P
- SLN
- STANDARDIZE
- STDEVPA
- STEYK
- SYN
- T.DIST
- T.DIST.2T
- T.DIST.RT
- T.INV
- T.INV.2T
- T.TEST
- TBILLEQ
- TBILLPRICE
- TBILLYIELD
- TREND

- TRIMMEAN
- UNICHAR
- VARA
- VARPA
- VDB
- WEBSERVICE
- WEIBULL
- WEIBULL.DIST
- XIRR
- YIELD
- YIELDDISC
- YIELDMAT
- Z.TEST
- ZTEST

Presenting Data With Cube Views

Cube Views submit and present cube data. You can format Cube Views to display final reports or forms for data entry, both of which can be exported to a PDF or Excel spreadsheet. You create and maintain Cube Views in Application > Presentation > Cube Views . You can also access and run Cube Views from:

- Workflow Task (forms)
- Workflow Analysis section of a workflow
- Cube Views section in the OnePlace tab
- Spreadsheet and Excel Add-In

Cube Views are the building blocks of reporting and can be incorporated in Report Books, dashboards, form templates, spreadsheets, and Extensible Documents. You can also link Cube Views to support a more focused, granular analysis. See the following sections:

- Build a Simple Cube View
- Member Filter Builder
- Cube View Performance
- Linked Cube Views
- Linking Cube Views Through the Cube POV

Build a Simple Cube View

A Cube View can query and submit data into a cube. This section focuses on how to build a basic Cube View.

Cube View Toolbar

Use the toolbar items to manage Cube Views.



- To create a group and organize Cube Views, select the **Create Group** icon.
- To create a profile and organize Cube View groups, select the **Create Profile** icon.
- To assign Cube View groups to Cube View profiles, select the **Manage Profile Members** icon.
- To create a new Cube View in the Cube View group, select the **Create Cube View** icon.
- To delete an item, such as a Cube View or Cube View group, select the **Delete Selected Item** icon.
- To rename an item, select the **Rename Selected Item** icon.
- To cancel unsaved changes, select the **Cancel All Changes Since Last Save** icon.
- To save your changes, select the **Save** icon.
- To copy the selected Cube View and use it as a template for another Cube View, select the **Copy Selected Cube View** icon.

- To search for Cube Views, select the **Search** icon.
- To run a Cube View and view it in a data explorer report, select the **Open Data Explorer** icon.
- To view other areas where the selected item is used, select the **Show Objects That Reference The Selected Item** icon. For example, view where a Cube View is being used in an application to understand the impact of a change.
- To find and select an object to copy, select the **Object Lookup** icon.

Create Cube View Groups and Profiles

A Cube View group must be created before a Cube View. Cube View groups organize the artifacts a user is building, and Cube Views created are added to existing Cube View groups. Cube View groups are organized into Cube View profiles. Profiles further structure the artifacts and enable them to be accessed by users in other areas of the application, such as a Workflow Profile. Cube View profile can include one or more Cube View groups.

Complete the following steps to create a Cube View group:

1. On the **Cube Views** page, select the **Create Group** icon.
2. Complete the following General and Security properties:

Property	Description
General (Cube View Group)	
Name	The name of the Cube View group.
Workspace	The currently selected Workspace.
Maintenance Unit	The currently selected maintenance unit.
Description	A short description of how the group is used or what it contains.
Security	
Access Group	Members of this group can access the Cube Views in the Cube View group.
Maintenance Group	Members of this group can maintain the Cube Views in the Cube View group.

> **NOTE:** Select the ellipses to open the **Object Lookup** dialog box. Enter the name of the security group in the search bar. Select the desired group from the filtered results and select the **OK** button or press **CTRL** and double-click to enter the selected name in the field.

3. Select the **Save**  icon.

Complete the following steps to create a Cube View profile:

1. Select the **Create Profile**  icon.
2. Complete the following General and Security properties:

Property	Description
General (Cube View Profile)	
Name	The name of the Cube View profile.
Workspace	The currently selected Workspace.
Maintenance Unit	The currently selected maintenance unit.
Description	A short description of how the profile is used or what it contains.
Visibility	Use this drop-down menu to specify where Cube Views can be viewed in the application. The default is Always, which displays the Cube Views in all listed locations. If you select a specific location, the Cube View will display in that location. If you select Never, the Cube View only displays on the Cube Views page in the Application tab.
Security	
Access Group	Members of this group can access the Cube Views in the Cube View profile.

Property	Description
Maintenance Group	Members of this group can maintain the Cube Views in the Cube View profile.

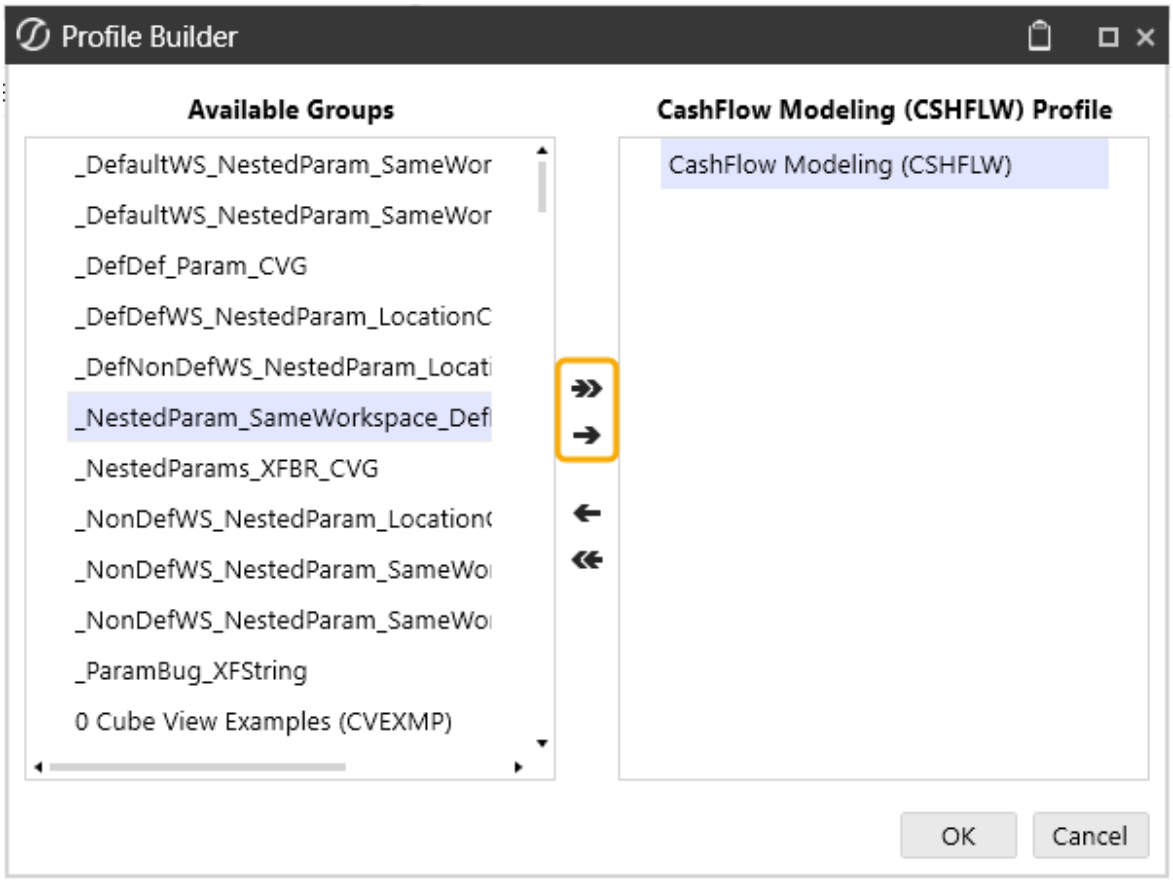
> **NOTE:** Select the ellipses ... to open the **Object Lookup** dialog box. Enter the name of the security group in the search bar. Select the desired group from the filtered results and select the **OK** button or press **CTRL** and double-click to enter the selected name in the field.

3. Select the **Save** icon.

Assign Cube View Groups to Cube View Profiles

Assign Cube View groups to Cube View profiles to control how the Cube Views in the group are accessed. Complete the following steps:

1. On the **Cube Views** page, under **Cube View Profiles**, select a profile.
2. Select the **Manage Profile Members** icon.
3. In the **Profile Builder** dialog box, under **Available Groups**, select all desired groups and add them to the profile.



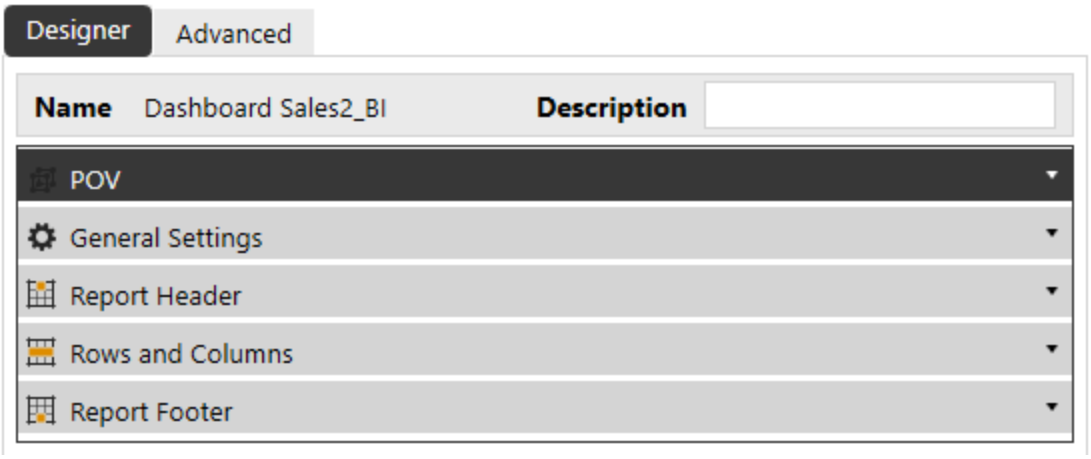
4. Select the **OK** button.

> **NOTE:** You can assign a Cube View group to multiple Cube View profiles.

Create a New Cube View

Complete the following steps to create a Cube View:

1. On the **Cube Views** page, under **Cube View Groups**, select a group.
2. Select the **Create Cube View** icon.
3. Enter Cube View properties.



> **NOTE:** The Designer and Advanced tabs have the same properties, but they are organized differently. You can use either tab.

Property	Description
POV	Make selections for the cube and all dimensions used in the Cube View. You do not need to make selections for any dimensions used in rows or columns. Dimensions can also be selected in the Cube POV. See Set the Cube View POV.
General Settings	Apply row and column templates, control common settings, configure headings and formatting, and set navigation links using bound parameters. See General Settings.

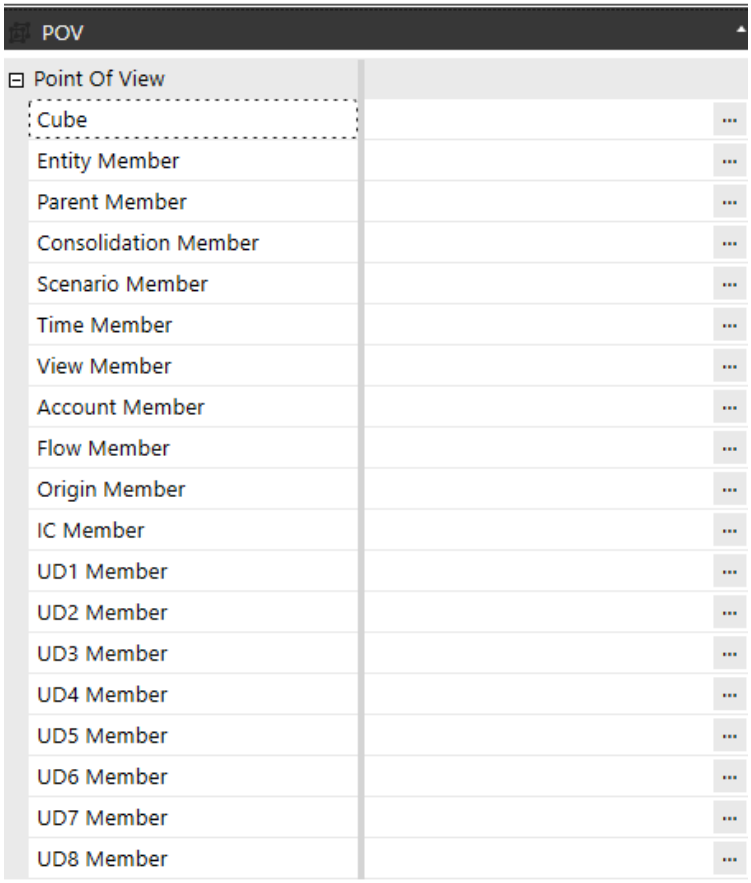
Property	Description
Report Header	Control what displays in the header when a Cube View is run as a data explorer report in PDF style. See Format a Report Page.
Rows and Columns	Determine the content for the rows and columns. See Create Cube View Rows and Columns and Format a Cube View.
Report Footer	Control what displays in the footer when a Cube View is run as a data explorer report in PDF style. See Format a Report Page.

4. Select the **Save** icon.

Set the Cube View POV

A Cube View needs POV settings to retrieve and query data. The Cube View POV sets the cube and all dimensions used for querying the various cells in the Cube View.

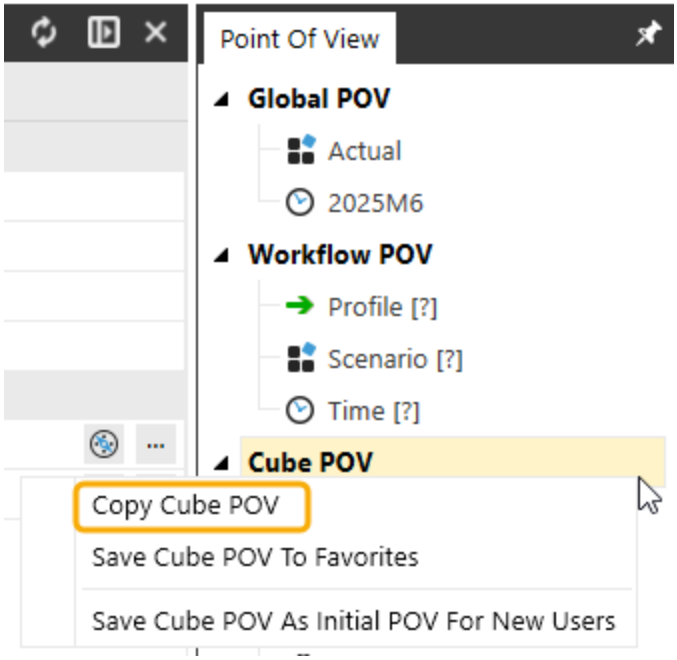
1. On the **Cube Views** page, under **Cube View Groups**, select a Cube View.
2. Select the **POV** slider.



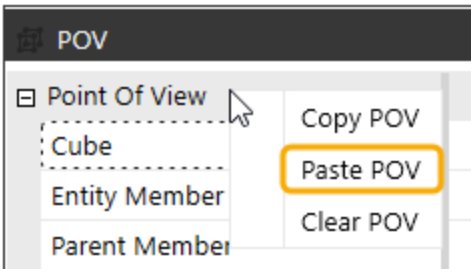
3. Enter information individually in the fields, select the ellipsis for each member, or move an entire Cube POV.

Complete the following steps to move a Cube POV:

1. To move a Cube POV, navigate to the POV pane by selecting the **POV Pane**  icon.
2. Right-click the **Cube POV** header and select **Copy Cube POV**.



3. In the Cube View POV properties, right-click the **Point Of View** header and select **Paste POV**.



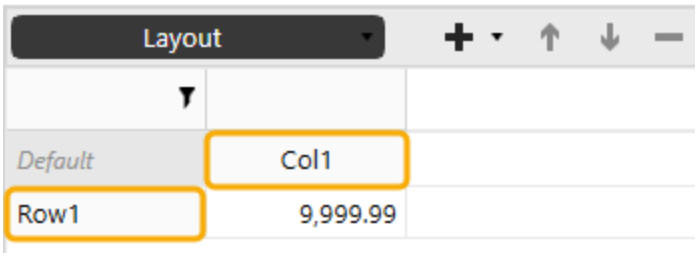
> **TIP:** To move an entire cube POV, you can also drag and drop the Cube POV from the POV pane into the Cube View POV.

4. Select the **Save** icon.

> **NOTE:** The information in the POV slider takes priority over the information in the POV pane.

Create Cube View Rows and Columns

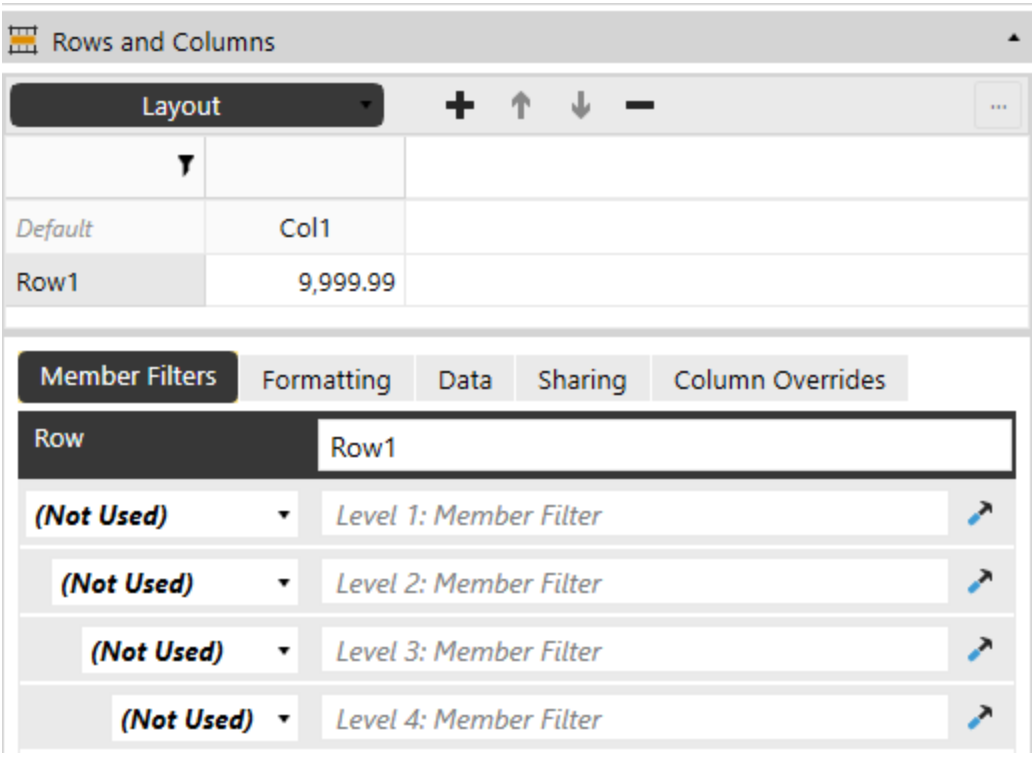
1. To set Cube View rows and columns, select the **Rows and Columns** slider. The preview grid for the new Cube View displays default row and column names: Row1 and Col1. You can edit these names as needed.



2. Use the following toolbar icons to manage rows and columns:

   - To add rows or columns, select the **Add Row or Column** icon.
   - To move rows or columns up, select the **Move Up** icon.
   - To move rows or columns down, select the **Move Down** icon.
   - To remove a row or column, select the **Remove Selected Row Or Column** icon.

3. Select a row, column, or intersection in the preview grid and set display options in the settings tabs. The member filters, formatting, data suppression options, and overrides can be applied at the row and column level. See Format a Cube View and Member Filter Builder.



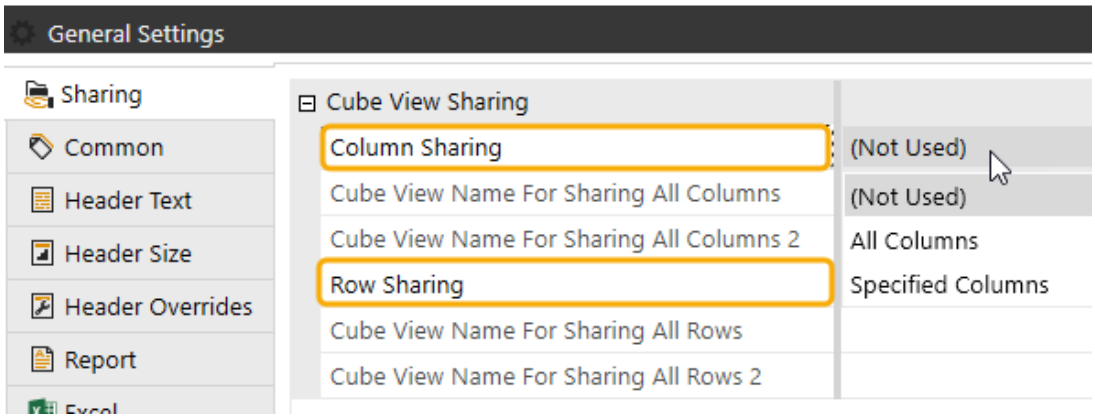
4. Select the **Save** icon.

Create a Cube View Using Shared Rows and Columns

An administrator can share all rows and columns or selected rows and columns from another Cube View. Sharing enables users to efficiently manage and build Cube Views.

Complete the following steps:

1. On the **Cube Views** page, under **Cube View Groups**, select a Cube View.
2. Go to **General Settings** > **Sharing**.
3. In the **Row Sharing** and **Column Sharing** fields, select an option from the drop-down menus.
    - **All Rows/Columns**: Enter the source Cube View name of the rows and columns to be shared.
    - **Specified Rows/Columns**: Reference a single row or column from the source Cube View when designing rows and columns.



> **TIP:** You can share rows and columns from two Cube Views. If you share rows or columns with another Cube View, updates are automatically linked. This can make building Cube Views more efficient.

4. If you selected the option to share all rows or columns, select the ellipsis ... and select the Cube View. Select the **OK** button.
5. Select the **Save** icon.

Copy a Cube View

To improve efficiency, copy a Cube View and use its formatting settings rather than creating a new Cube View. A Cube View copy is not linked to another Cube View.

1. On the **Cube Views** page, under **Cube View Groups**, select a Cube View.
2. Select the **Copy Selected Cube View** icon.
3. In the dialog box, select a Cube View group from the drop-down menu.
4. Enter a unique Cube View name and select the **OK** button.

Use a Cube View Template

Creating and using Cube Views as templates increases efficiency and consistency for Cube Views that have similar formatting and properties for rows and columns.

Select a Cube View to use as a template and select the **Copy Selected Cube View** icon. Select a Cube View group for the template and enter a new Cube View name.

> **TIP:** If you plan to use templates, create a Cube View group specifically for this purpose. For example, create a Cube View group with the name template, Templates_Columns or Templates_Rows, and save Cube Views to use as templates in this Cube View group. Then, other users can find and use those templates.

Cube View Shortcuts

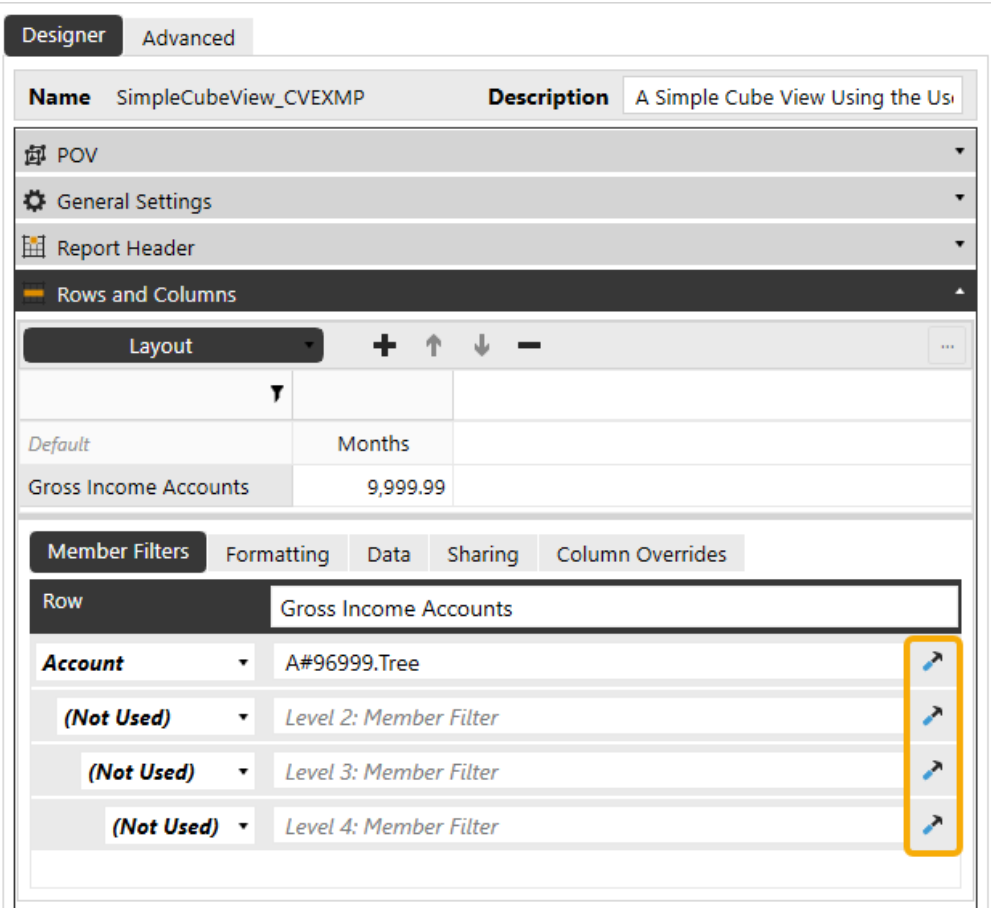
Cube View shortcuts are used to launch other Cube Views. This is useful when an administrator wants to use the same Cube View but not the same parameters.

A user launches an Income Statement and is prompted with a ParamView parameter. The ParamView has two values: YTD or Periodic. This means an Income Statement can be launched to show the data with a Year to Date view or a periodic view, such as Month to Date.

In this case, a shortcut can be used to save two versions of the same report without prompting the user or having to maintain two Cube Views. In each case, the shortcut Cube View name would be the same, but the literal parameter value would be different. The YTD version would be ParamView=[YTD] and the Periodic version would be ParamView=[Periodic]. Each would open the Income Statement Cube View to the proper view settings without prompting the user.

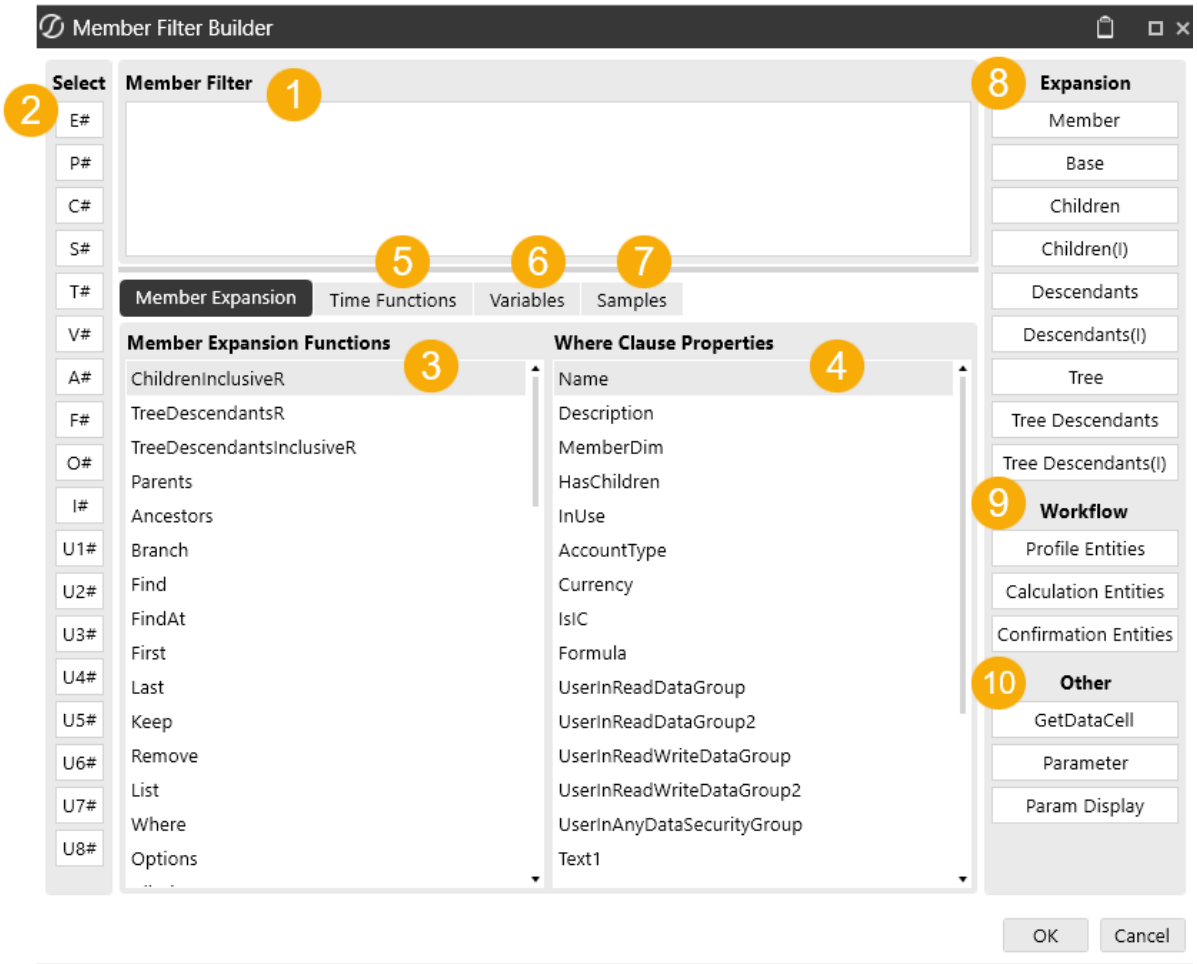
Member Filter Builder

The Member Filter Builder dialog box enables you to build complex member filters without having to remember or look up the proper syntax. On the **Cube Views** page, select a Cube View. Select the **Rows and Columns** slider and then select a row or column header. In the Member Filters tab, click the **Member Filter Builder** icon.



Navigation

Hover over the items in the Member Filter Builder dialog box for additional information.



1. **Member Filter**: Location where the member filter is built. Type in this section or use the other items in the dialog box to enter information into the field.
2. **Dimension Tokens**: There is a button for each dimension that launches the appropriate selection dialog box.

> **NOTE:** When searching for a member in the Select Member window, you can search by its name, alias, or description.

3. **Member Expansion Functions**: Double-click a member expansion to add it to the member filter.
4. **Member Expansion Where Clause Properties**: If you select Member Expansion Where, use the where clause properties to complete the expression.

   > **Example:** UD2#AllProducts.Children.Where(Name Contains Clubs)

5. **Time Functions**: Only applies to the Time dimension, such as T#POVPrior1. Double-click a time function to add it.
6. **Variables**: Double-click a system-wide substitution variable to add it.
7. **Samples**: This tab provides example syntax for complex queries and calculations, including member expansions, where clause and time expressions, GetDataCell and row/column calculations, custom member list expressions, and XFBR, XFCell, and XFGetMemberProperty.
8. **Expansion**: Commonly used member expansions added to the end of filters. Double-click the expansion to add it.
9. **Workflow**: Commonly used workflow member expansions used in Cube Views that point to a report, form, or dashboard and are affiliated with a specific Workflow Profile.
10. **Other**: Commonly used member filter functions that enable you to create calculated rows and columns or use a custom parameter to store member lists.

Member Expansion Tab

To reduce Cube View maintenance, use simple member expansions and connect your Cube Views to the metadata when complex expansions are needed. Avoid referencing individual members. Select a member expansion and double-click it to add the sample syntax to the Member Filter.

- **Reverse order expansions**: These selections include ChildrenInclusiveR, TreeDescendantsR, and TreeDescendantsInclusiveR.
- **Where clauses**: Enable further qualification of the results and can pull members based on properties such as text or name contains.
- **Remove functions**: Removes some of the members from the results.
- **Parents**: Returns the direct parent of the member. There may be multiple parents if there are alternate hierarchies.
- **Ancestors**: Returns all members up the chain from the original member.
- **Options**: Advanced option if you are a report builder familiar with the extensibility design in your application. This expansion is commonly used to focus on one portion of your Extensible Dimensionality.

This is not an exhaustive list of all the member expansions. Each expansion can also be found in the Samples tab. Hover over each to see the purpose of the function and a sample of the expected syntax.

Use the member filter to filter data by creating a list of restricted members. Member Filters can contain multiple member scripts. A member script queries a defined set of dimensional members. Members can be specified for any or all dimensions, and the primary dimension can also specify a member expansion formula, for example, Descendants.

> **Example:** This member script returns the year 2022: T#2022.

> **Example:** This member script has a member expansion that returns all income statement accounts: A#[Income Statement].Descendants.

If one or more dimensions in the member script have a dimension token, the remaining dimensions are pulled from the Cube View POV, the Global POV, or Workflow, Time and Scenario. In the member script, separate each dimension that has a token with a colon.

> **Example:**
> Cb#GolfStream:E#Houston:P#Texas:C#Local:S#Budget:T#2022M3:V#YTD:A#60000:F#None:O#Forms:IC#None: U1#Sales: U2#HybridXL: U3#Northeast: U4#SportsCo: U5#None: U6#None: U7#None: U8#None

> **Example:** This Member Filter contains three different member scripts that returns the Actual, Budget, and Forecast scenarios: S#Actual, S#Budget, S#Forecast.

Time Functions Tab

Time is a fixed dimension and is based on the time dimension type associated with the application. OneStream has custom time dimensions, like weekly and monthly, that affect time functions and expansions. Time functions pivot the member, and time expansions extend the member.

> **Example:** This time function displays the prior period of the time member in a Cube POV: T#POVPrior1

> **Example:** This time expansion displays all months in the year 2022: T#2022.Months

Complete the following steps to add a time function from the Time Functions tab:

1. In the **Member Filter Builder** dialog box, select the **Time Functions** tab.
2. Select the type of time function desired. You can choose from the following options: All, POV, WF, Global, and General.
3. Double-click a function to populate it in the **Member Filter** field.

> **NOTE:** Select the **Samples** tab to view more time functions.

The following time function example demonstrates a flexible syntax style. You can choose from a variety of time functions, but this example enables you to pivot easily on the year and period separately:

T#YearPrior1(|PovTime|)PeriodNext1 (|PovTime|)

- Prior year from the POV time: T#YearPrior1(|PovTime|)
- Next period from the POV time: PeriodNext1 (|PovTime|)

Variables Tab

Substitution variables are short scripts that use pipe characters to include a predefined substitution variable. They come with every installation of OneStream and cannot be edited. You do not need to create or maintain them. In the Variables tab, double-click a substitution variable to add it to a Member Filter.

> **Example:** |WFProfile| would refresh to display the current Workflow Profile name.

Substitution variables can be used throughout the application. For Cube Views, you can use them for headers and footers, rows and columns, and Cube View page captions.

> **NOTE:** Substitution variables are always referenced with pipes, for example, |POVTime|.

The Member Filter Builder dialog box includes substitution variables in the following categories:

- **POV**: Cube POV
- **WF**: Workflow POV
- **Global**: Global POV
- **CV**: Cube View POV
- **MF**: Member Filter
- **General**: Items not related to a POV

When using a substitution variable to return a member name, the prefix indicates where the value of the variable is pulled from. You can choose if it should refresh based on the Cube View POV (CV), Workflow POV (WF), Cube POV (POV), Member Filter (MF), or Global POV (Global).

You can choose if you want to display the member description, usually in headers and footers, by adding Desc as the suffix. Time has an added short description option that is set through the time profiles.

> **Example:** |WFTimeDesc| returns the description Feb 2022, and |WFTimeShortDesc| returns the short description Feb.

Review the following common substitution variables:

- Username that ran the report: |UserName|
- Cube View name: |CVName|
- Members text properties: |Text1|
- Today's date: |DateDDMMYYYY|

> **TIP:** In addition to the Variables tab, you can also copy a substitution variable from the Object Lookup dialog box. Select the **Object Lookup** icon and go to **Object Type** > **Substitution Variables**.

Samples Tab

The Samples tab provides example syntax for more complex queries and calculations in the following categories:

- Member Expansions
- Where Clause Expressions
- Time Expressions
- GetDataCell Expressions
- Column/Row Expressions
- Custom Member List Expressions
- XFBR, XFCell, and XFGetMemberProperty

Select a sample and double-click it to add the syntax to the Member Filter.

Where Clauses

Where clauses are commonly used in reporting to create a more flexible query. Select a where clause and double-click it to add the syntax to the Member Filter.

> **NOTE:** You can view common Where clauses in the Samples tab.

The where clause can pull members based on properties:

- Text properties
- Portions of the description or name, for example, starts with or contains.
- Security
- Account types. This is Account dimension only.
- Intercompany. This is Entity and Account dimension only.
- Specific currency. This is Entity dimension only.
- In use property
- Has children

> **NOTE:** Where(HasChildren = True) can only be used for the dimensions assigned to the cube on the default Scenario Type.

- Has a member formula

Cube View Performance

This section includes the following information:

- Database Sparsity
- Row and Column Suppression
- Suppressed Members
- Sparse Row Suppression
- Cube View Paging

Database Sparsity

Sparsity is the ratio of data record volumes in the cube compared to the dimensions modeled in the cube. Sparsity can be seen when the Data Unit, defined by the cube, entity, parent, consolidation, scenario, and time, has sparse intersections across the account-type dimensions, which include account, intercompany, flow, and user-defined.The absence of data records can affect reporting performance because sparse data makes reports harder to render.

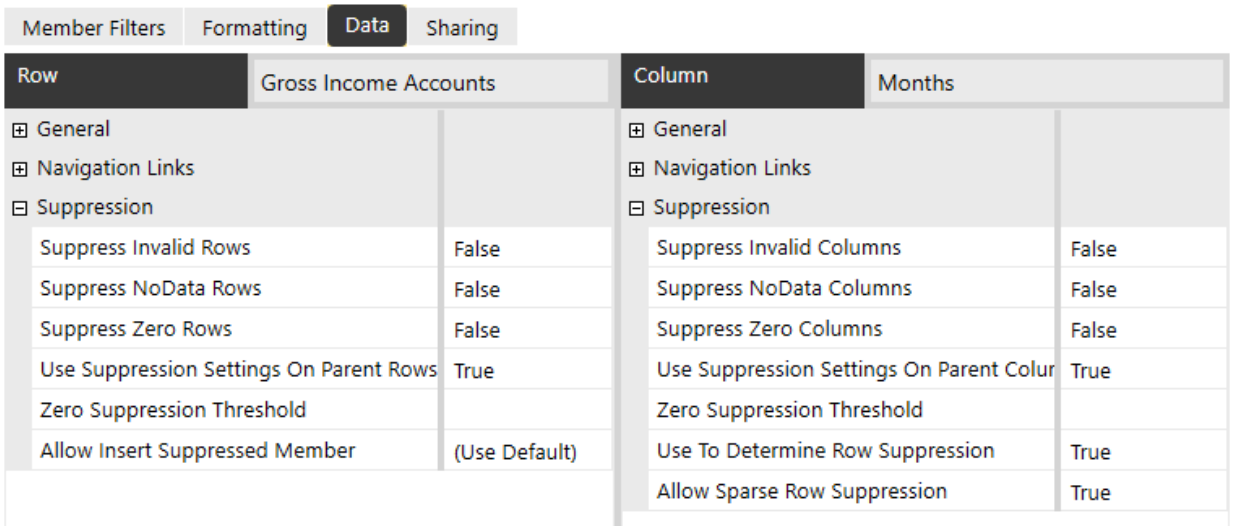
You can minimize sparsity in your application design by utilizing analytic blend, extensibility, or other design frameworks to ensure the Dimension Library is designed for your reporting structure. However, even with an optimal design, sparsity can still occur if a large report pulls in many dimensions. If this is the case, suppression settings help improve reporting performance.

Row and Column Suppression

You can use row and column suppression in larger reports to make them easily readable. This also enables the report builder to create a low maintenance report based on the metadata design.

Complete the following steps to adjust suppression settings:

1. On the **Cube Views** page, under **Cube View Groups**, select a Cube View.
2. Select the **Rows and Columns** slider.
3. Select the row or column and select the **Data** tab.



Property	Description
Suppress Invalid Rows/Columns	This property suppresses any invalid cells when set to True.
Suppress NoData Rows/Columns	This property suppress any cells without data when set to True. When data entry is required, False is the recommended setting.
Suppress Zero Rows/Columns	This property suppresses any cells containing zeros when set to True.
Use Suppression Settings On Parent Rows/Columns	This property relates to the prior properties and controls whether the parent members in this Member Filter use the same settings.

Property	Description
Zero Suppression Threshold	When the Suppress Zero Rows/Columns property is set to True, enter a threshold value. All numbers below the specified number are recognized as zeros. For example, entering 499.99 results in every number lower than that value being recognized as zero and therefore suppressed.
Allow Insert Suppressed Member	Use this property to access a member currently suppressed for data entry purposes. This can only be used with Cube Views and form templates. You can choose from the following options: • **False**: All row expansions remain suppressed. • **All**: Enables visibility to all Cube View row expansions. • **Nested**: Enables visibility of row expansions two through four. • **Innermost**: Enables visibility of the bottom level row expansion.
Use To Determine Row Suppression	When set to True, this property improves performance on large Cube Views by enabling the designer to better define how to apply row suppression.
Allow Sparse Row Suppression	When set to True, sparse row suppression is applied to the entire Cube View. It can be turned on and off for specific columns. This provides performance improvements for Cube Views that use multiple nested row dimensions. This works in conjunction with a General Settings property. See Sparse Row Suppression.

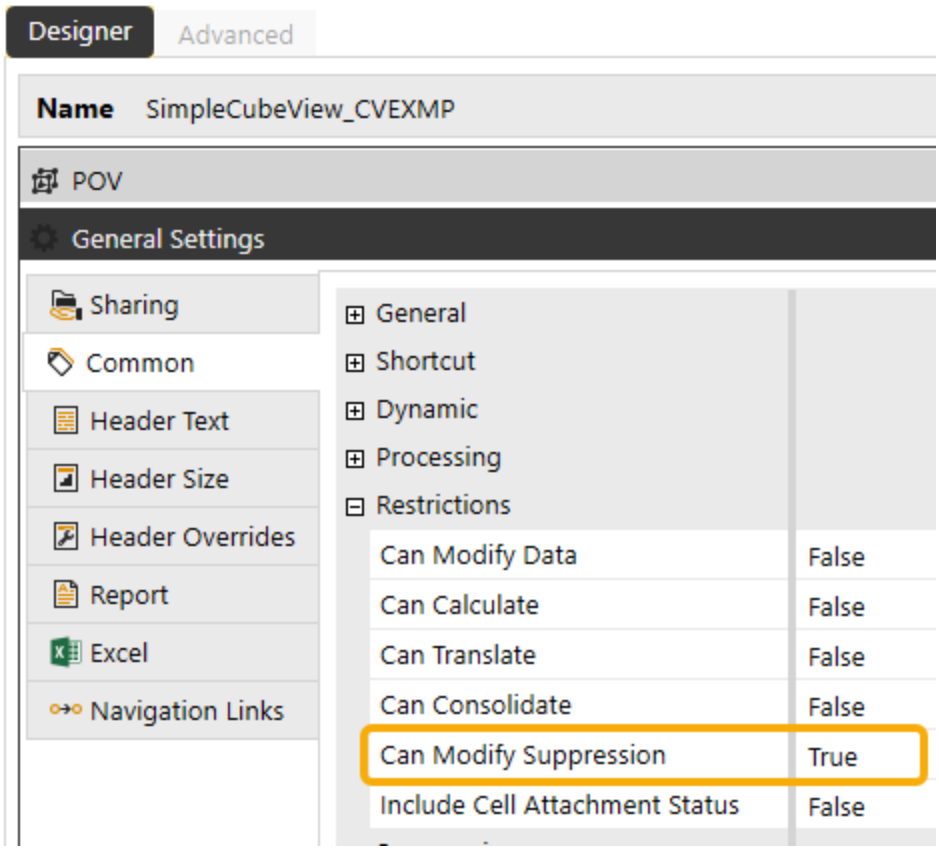
Suppressed Members

This section includes instructions for how to manage suppressed members.

Apply Modify Suppression Property

With the Can Modify Suppression property, you can choose whether rows are suppressed from the data explorer grid. This is useful for a small Cube View that will not generate a large number of additional rows if suppression is turned off.

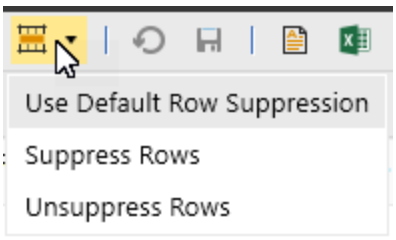
Select a Cube View and go to **General Settings** > **Common** > **Restrictions** and set the **Can Modify Suppression** property to **True**.



Modify Suppression

To modify suppression, you must first apply the Can Modify Suppression property. See Apply Modify Suppression Property.

1. Select a Cube View and select the **Open Data Explorer** icon.
2. Select the **Row Suppression** drop-down menu.

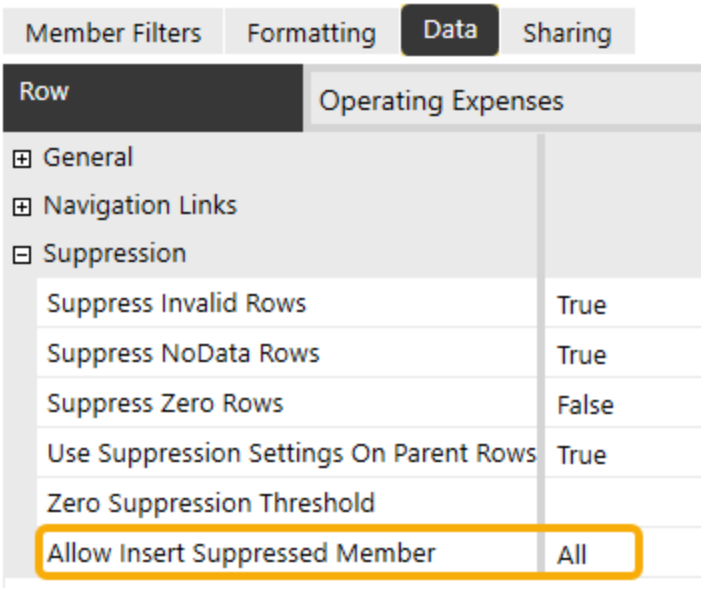


3. Choose from the following options:
    - **Use Default Row Suppression**: Apply the Cube View suppression settings.
    - **Suppress Rows**: Suppress any rows with zeros or no data regardless of the Cube View settings.
    - **Unsuppress Rows**: Unsuppress all rows that were suppressed with zeros, no data, or invalid data regardless of the Cube View settings.

Apply Property to Allow Insert Suppressed Members

You can insert suppressed members so that their data is entered in the Cube View. Select the Cube View and go to **Rows and Columns** > **Data** > **Suppression** and set the **Allow Insert Suppressed Member** property to **All**.

> **NOTE:** The Allow Insert Suppressed Member option is only available for rows.



Insert Suppressed Members

To insert suppressed members, you must first apply the Can Modify Suppression property. See Apply Property to Allow Insert Suppressed Members.

1. Select a Cube View and select the **Open Data Explorer** icon.
2. Right-click in the account or entity.
3. Select **Insert Suppressed Member**.
4. To the right of the account or entity field, select **Select Member**.
5. Select an item and select the arrow to add it to the **Result List**.
6. Select the **OK** button and then select the **OK** button to confirm.
7. Enter data as needed in the white cells.
8. Select the **Save** button and then select the **OK** button.

Sparse Row Suppression

Designs for analytic reports typically have multiple dimensions nested in rows. The combination of members generated from the nested expansions can easily result in billions of potential expanded rows, many of which may not have data. In these designs, standard row and column suppression in the Cube View would still require each of those billion rows to be inspected individually for data. So, it is recommended that sparse row suppression be enabled to enhance the performance of a large Cube View when, due to widespread database sparsity, the report is likely to return many records without data and take a long time to run. Sparse row suppression evaluates the data records of the Cube View intersections and filters records with no data (not zeros) before rendering the Cube View.

> **IMPORTANT:** Sparse row suppression cannot be applied to dynamically calculated data through dynamically calculated members and Cube View math. Avoid errors by correctly applying the settings on the dynamically calculated columns.

> **NOTE:** When sparse row suppression is applied, OneStream assesses all row data before displaying the Cube View. Cells containing dynamic calculations are populated on-the-fly. When OneStream assesses the rows of data, the rows containing dynamically calculated data are omitted from the Cube View because the data is not saved in the database.

To enable sparse row suppression in Cube Views, complete the following steps.

1. Select a Cube View and go to **General Settings** > **Common** > **Suppression**.
2. Set **Allow Sparse Row Suppression** property to **True**.
3. In the **Rows and Columns** slide, select the row and then select the **Data** tab.
4. Ensure that the additional row suppression properties you will use are also set to **True**.

> **NOTE:** Any row assigned a suppression setting is enabled for sparse row suppression. If no suppression is applied, sparse row suppression are not applied.

5. In the **Rows and Columns** slider, select the column and then select the **Data** tab.
6. Under **Suppression**, set **Allow Sparse Row Suppression** to **True**.
7. Ensure that the additional column suppression properties you will use are also set to **True**.

> **NOTE:** If columns contain dynamically calculated data, set **Use to Determine Row Suppression** and **Allow Sparse Row Suppression** to **True** to avoid an error.

Cube View Paging

Cube View paging is only applied to the data explorer view of a Cube View and is used to enhance the performance of Cube Views containing more than 10,000 unsuppressed rows. The Cube View will attempt to return up to 2000 unsuppressed rows within a maximum processing time of 20 seconds. The purpose of paging is to protect the server from large Cube Views that could affect application performance.

Evaluations are performed on the potential size of the rows and the processing requirements to determine if paging is enabled and the number of rows returned.

Evaluations are performed on the potential size of the rows and the processing requirements to determine if paging is enabled and the number of rows returned.

- **Evaluation 1**: Enable Paging. This evaluation is performed on the entire Cube View to determine the total number of possible unsuppressed rows that will be generated. If the total number of potential unsuppressed rows is less than 10,000, no paging is enabled.

- **Evaluation 2**: Paging Enabled. If the total number of unsuppressed rows is greater than or equal to 10,000, paging is enabled.
- **Evaluation 3**: Paging. Once paging begins, the Cube View evaluates the rows attempting to return a minimum of 20 to a maximum of 2000 unsuppressed rows. In the case of nested dimensions on rows, the evaluation starts on the left-most dimension expansion, as defined in the Cube View. After a maximum processing time of 20 seconds, the first page of the Cube View is returned for display containing only the rows that completed processing during the time constraint.

  For this reason, Cube View pages are not a fixed number of rows. The rows are ultimately determined by their time to process. This also relates directly to the percentage display because each page is generated by processing time requirements, and the last page is not known while the Cube View is running. Therefore, this percentage is not intended to be a precise measurement.

When a row is defined with nested dimensions, the paging evaluation is performed on the left-most dimension. For each expansion of the left-most dimension, the paging will not progress to the next sibling until all records are returned by all other dimension expansions to be completed.

To set the properties, select a Cube View and go to **General Settings** > **Common** > **Paging**. The following properties are available:

- **Max Unsuppressed Rows Per Page**: This property determines how many rows are written before the Cube View starts paging. The default value is -1. The maximum value is 100,000.
- **Max Seconds To Process**: This determines how many seconds the Cube View processes before it starts paging. The default value is -1. The maximum value is 600 seconds.

See "Cube View Paging" in the *Logging Guide* for more information on how to cancel a long-running Cube View through Task Activity.

Advanced Cube Views

Cube Views offer several advanced configuration options that reduce the number of Cube Views you need to create and maintain. For example, you can configure a Cube View to change based on Point of View or workflow.

Workflow

Set the POV, rows, and columns in the Cube View so it is driven by the Workflow POV and the entities assigned to the Workflow Profile in use. This enables you to make forms, dashboards, Cube Views, and reports that are dynamically driven by the Workflow Profile.

- **Workflow Entities**: In the Rows and Columns tab, using an expression like E#Root.WFProfileEntities shows the entity or entities assigned to that particular Workflow Profile at run-time. WFProfileEntities or similar expressions cannot be assigned to the POV because they can return more than on entity and the POV only requires a single member.
- **Workflow Scenario**: In the Cube View POV slider, select the WF Member for the Scenario dimension. Alternatively, use |WFScenario| or a similar substitution variable in the Rows and Columns slider.
- **Workflow Time**: In the Cube View POV slider, select the WF Member for the Time dimension. Alternatively, use |WFTime| or a similar substitution variable in the Rows and Columns slider.

Dashboard and Form Parameters

When using dashboards and forms, use parameters to focus the data to what is needed. For example, when a dashboard is launched, it can prompt for the specific entity or region needed. See "Filter Runtime Data and Reduce Maintenance with Parameters" in the *Understanding the Financial Model Guide*.

In the Cube View, apply parameters in the POV and Rows and Columns sliders to ensure the query returns only the intended data. Surround the parameter name with pipes and exclamation points, for example, |!ParameterName!|.

The following image displays an example where entity and region parameters are used in the Cube View's Point Of View.

POV

Point Of View	
Cube	GolfStream
Entity Member	|!WorkflowEntityPrompt!|
Parent Member	
Consolidation Member	Local
Scenario Member	WF
Time Member	WF
View Member	YTD
Account Member	
Flow Member	None
Origin Member	Top
IC Member	Top
UD1 Member	Top
UD2 Member	Top
UD3 Member	|!Region Parameter (Workflow)!|
UD4 Member	Top

> **TIP:** Use dashboard parameters as a single repository for parameters that are used in dashboards or forms. If a parameter is referred to within a Cube View and there is not a parameter by that name associated with the form, it will search through the application's dashboard parameters for one with that name and use it.

Changing Member Display Name

You can change the name returned with a member script in a Cube View by adding :Name(“enter name here”) at the end of the member filter. The double quotes around the Name() value are optional.

> **Example:** A#60999:Name(“Net Sales”) or A#60999:Name (Net Sales)

Format a Cube View

Formatting settings and overrides combine to create the cell format. You can apply formatting and isolate it to nested expansion levels on rows or columns by using property filters found in the Conditional Formatting dialog box.

The primary output for a Cube View is the data explorer grid. You can also export to Excel or use a report viewer.

Apply property filters in this order:

1. Application properties standard report settings
2. Cube View default settings
3. Column settings
4. Row settings
5. Column row overrides
6. Row column overrides

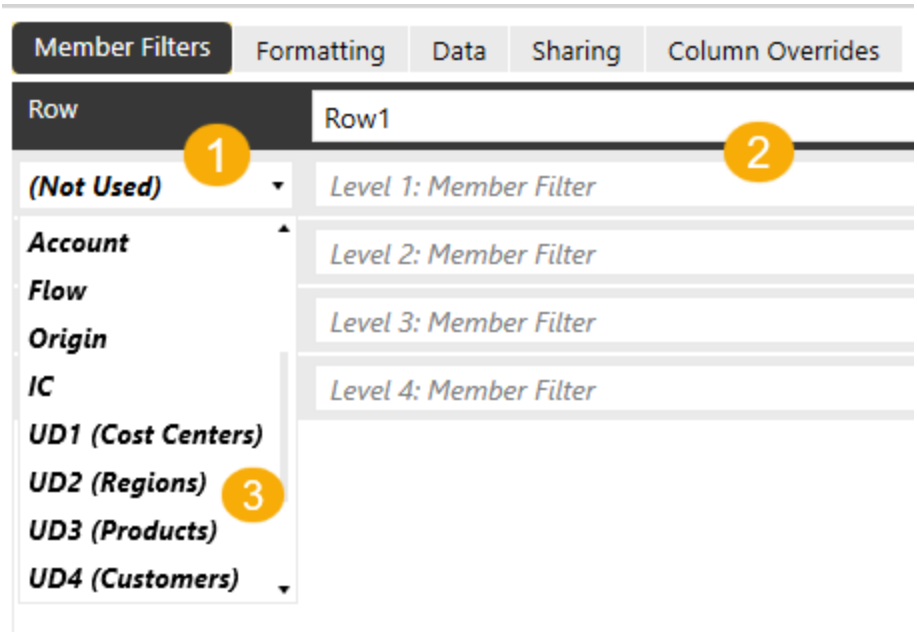
Review the following sections on formatting a Cube View:

- Rows and Columns Member Filters Tab
- Rename Rows and Columns
- Rows and Columns Formatting Tab
- General Settings
- Configuring Navigation Links for Linked Dashboards
- Linked Cube Views
- Linking Cube Views Through the Cube POV
- Format a Report Page

Rows and Columns Member Filters Tab

In a Cube View, you can nest up to four different dimensions as rows and two different dimensions as columns.

> **NOTE:** When nesting Cube View columns, if the top column header is longer, it will span the nested column headers when displayed on a Cube View report.



1. You can select a dimension type from the drop-down menu. This indicates which dimension type will display on the selected row or column.
2. Enter a Member Filter to express the specific members needed from the dimension.
3. The custom user-defined descriptions display in the Designer and Advanced tabs in the Member Filters dimension type.

Rename Rows and Columns

By default, a Cube View has standard headers to represent rows and columns. If the default is not used, the header settings control the row and column headers presented when the Cube View is run. See General Settings.

If the header name for a row or column must be different than the dimension name or description, use the rename function. The rename function is an expansion at the end of a Member Filter that enables you to define the display name.

> **Example:** A#CashBalance:Name(Cash Balance) will display Cash Balance as its header.

You can also include the XFMemberProperty, which returns member properties.

> **Example:** A#6009.base:Name(XFMemberProperty (DimType=Account, Member=|MFAccount|, Property=AccountType))

1. Use the rename function to pull in the member property: **Name(XFMemberProperty**
2. Define the dimension type: **(DimType=Account,**
3. Determine what will be in the member filter: **Member=|MFAccount|,**
4. Pull in the property: **Property=AccountType))**

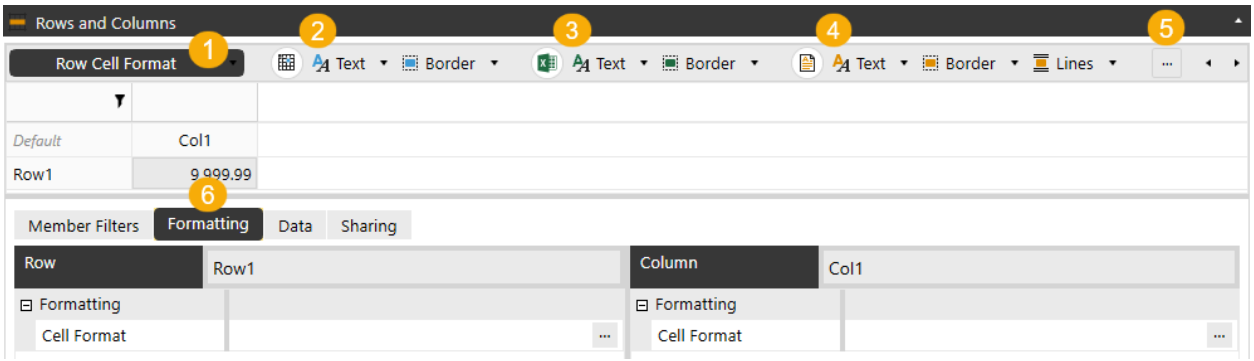
Rows and Columns Formatting Tab

Cube View formatting controls how Cube Views display. Most of the format settings transfer to Excel and can be overridden with standard Excel styles. The formatting also transfers to reports where the Cube View is used in a data adapter.

Cube View formatting can be selected for the entire Cube View or for specific rows, columns, headers, and individual cells. The formatting options enable number formats, percentage signs, scaling, currency symbols, colors, fonts, and font sizes to be unique to business needs. You can label row and column sets to support Cube View maintenance when editing, formatting, and applying calculations.

In the Cube View, select the **Rows and Columns** slider. Select an individual cell or header and then select the **Formatting** tab to view the toolbar options for data explorer, Excel, and report formats.

The icons in the toolbar include the most common formatting properties. Each icon coincides with the cell selected in the Rows and Columns slider. For example, if a column header is selected, the formatting changes made only affect the selected column header.



1. Selector
2. Data explorer formatting options
3. Excel export formatting options
4. Report formatting options
5. Advanced formatting options
6. Formatting tab

Set Up Basic Formatting for a Cube View

1. Select a Cube View and select the **Rows and Columns** slider.
2. Select the row, column, or cell to edit.

   > **NOTE:** Select the **Default** cell to update the default settings for all headers and cells.

3. Select the **Formatting** tab.

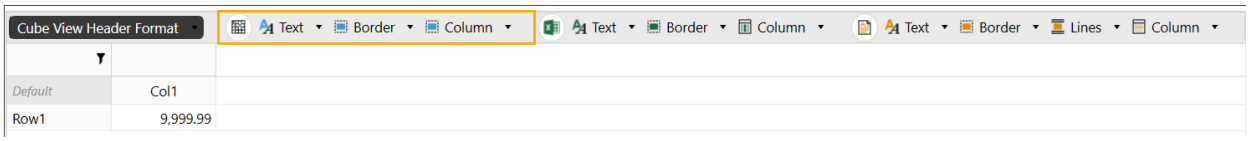
4. Select options to change the header and cell format settings for the data explorer, Excel, and report output types.

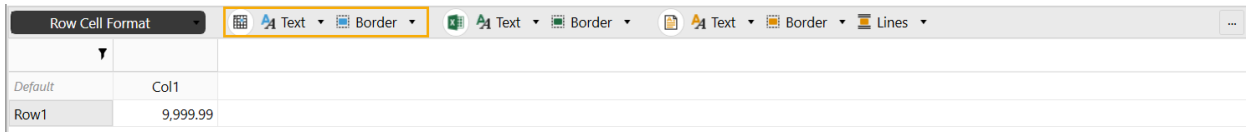
Data Explorer

The formatting settings available for data explorer are organized in the following groups:

- **Text**: Font, color, size, bold, and italic settings. When you select a cell, the following formatting settings become available in the Text drop-down menu: Number Format, Zero Offset for Formatting, Scale, Flip Sign, Show Percent Sign, and Show Currency.
- **Border**: Background color and gridline color
- **Column**: Column width and visibility.

There are different settings available for the headers and cells.





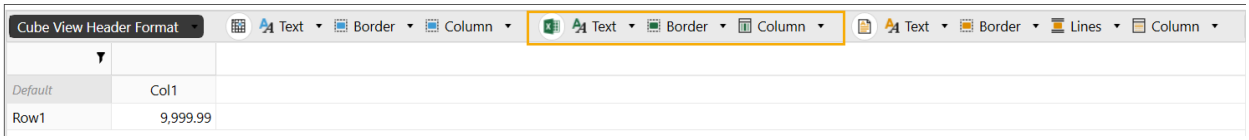
Excel

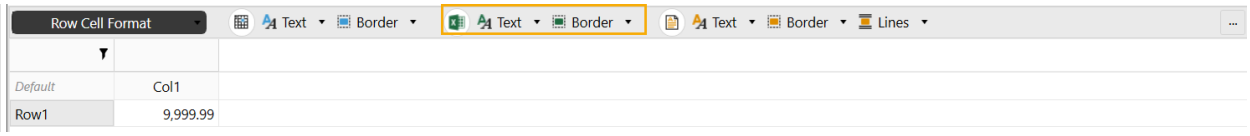
The formatting settings available are organized in the following groups:

- **Text**: Text Color, Horizontal Alignment, Vertical Alignment, Indent Level, and Wrap settings. When you select a cell, the following formatting settings become available in the Text drop-down menu: Number Format and Use Scale.

- **Border**: The color and line styles of the cell borders and background color.
- **Column**: Column Width setting.

There are different settings available for the headers and cells.



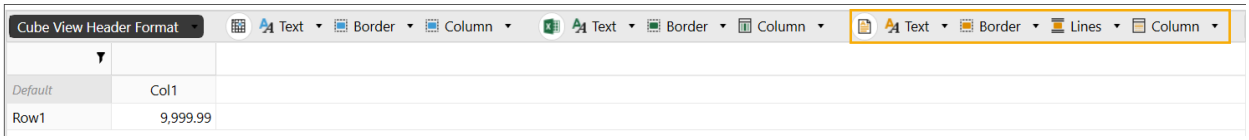


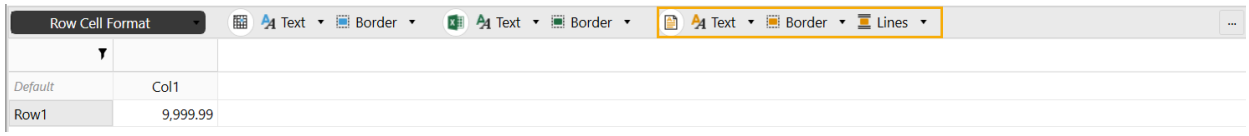
Report

The formatting settings available are organized in the following groups.

- **Text**: Text Color, Text Alignment, Text Size, and Underline settings. When you select a cell, the following formatting settings become available in the Text drop-down menu: Negative Text Color, No Data Number Format, Use Numeric Binding, and Numeric Binding Format.
- **Border**: Background color and cell borders.
- **Lines**: Settings for padding, color, and thickness for the top and bottom lines of the first and last rows.
- **Column**: Column Width setting.

There are different settings available for the headers and cells.

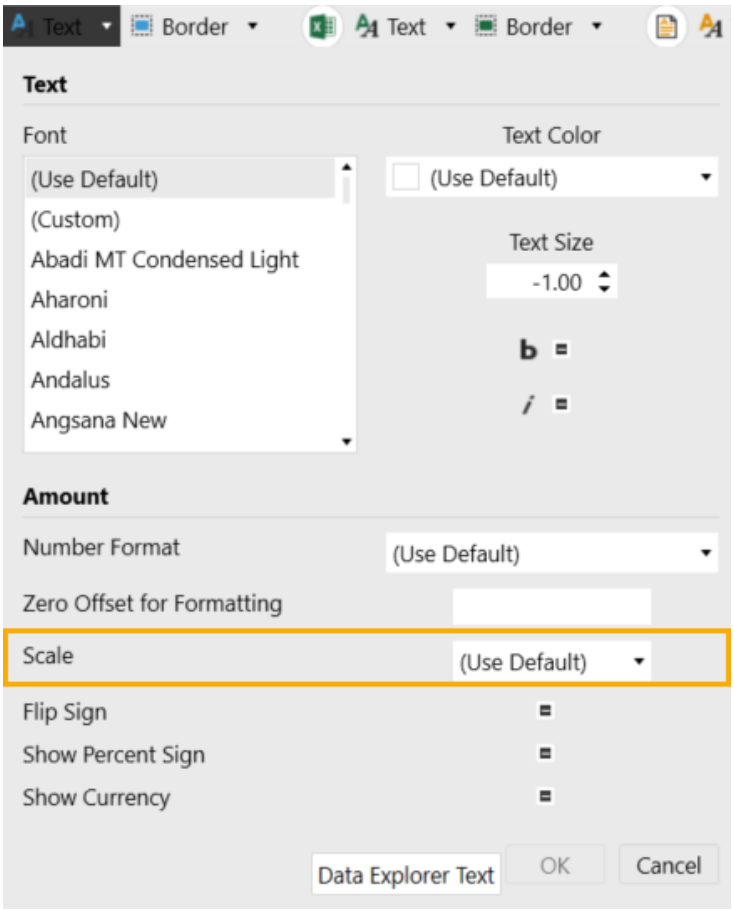




Apply Scaling, Indents, and Currencies

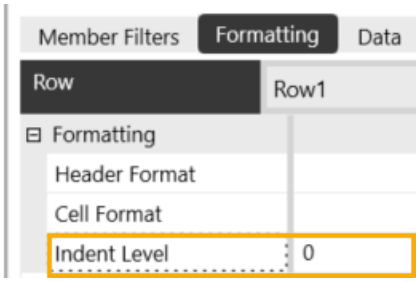
Scale

To use the Scale setting, open the data explorer **Text** drop-down menu. You can also apply it to Excel and report formatting.



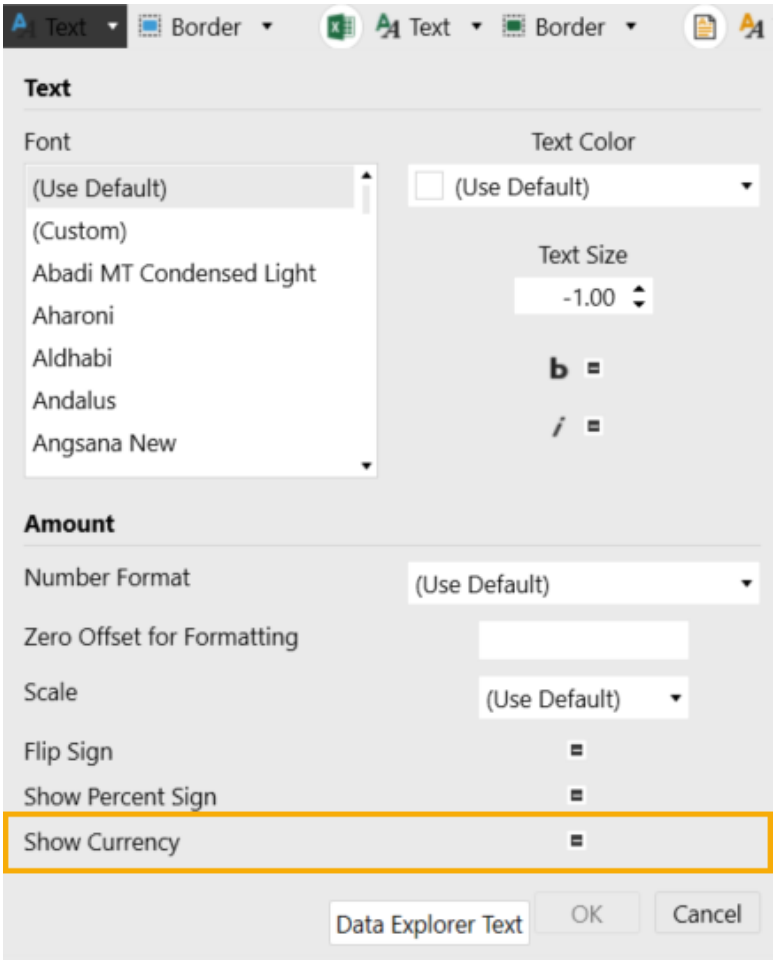
Indent Level

To view the Indent Level setting, select the **Formatting** tab for a row or column header.



Show currency

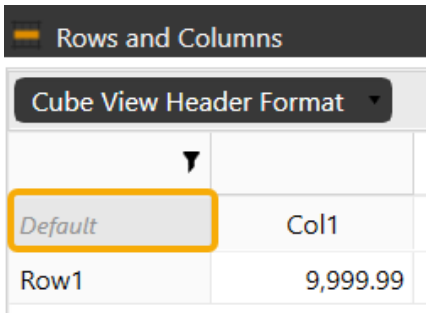
To view the Show Currency setting, open the data explorer **Text** drop-down menu.



Header Format Properties

The formatting properties for a Cube View header include default, row, and column headers. These properties are in the Rows and Columns slider. Not all properties are available for each header.

1. In the **Rows and Columns** slider, select the **Default** cell to apply the formatting settings to all Cube View headers.



> **NOTE:** You can select an individual row or column header instead if you want to edit the settings for a specific header.

2. In the **Formatting** tab, select the ellipsis to the right of **Header Format**.
3. In the **Header Format** dialog box, select the **Format** button to modify options. The Excel section only affects how the Cube View data display in Excel.

Review the following Header Format properties:

Property	Description
General	

Property	Description
Custom Parameters	Select the ellipsis to select and assign a custom parameter, such as a Cube View style.
RowExpansionMode	This controls the expansion of nested rows. This determines how rows are displayed in the data explorer grid. The default setting is False.
ShowDimensionImages	This determines if dimension icons display in the data explorer grid row and column headers. The default setting displays the dimension icons. Set to False to hide dimension icons.
HeaderWrapText	When set to False, column and row header text does not wrap. When set to True, column and row header text displays as wrapped. This applies to Default Cube Views and works with Header Overrides. **NOTE:** ColumnWidth and Row Header Width must be set from the Header Size settings menu for row header text to wrap

Property	Description
IsColumnVisible	This determines the visibility of specific columns. Set to False to hide columns. Set this property at run-time with a parameter to show and hide detail.
ColumnWidth	Enter a numerical value for column width. If the header label exceeds the column width, the header will automatically wrap the text.
ColumnHeaderWrapText	When set to False, column header text does not wrap. When set to True, column header text displays as wrapped. This applies to individual column headers and works with Header Overrides. ColumnWidth must be set for Column Headers and header text to wrap.
MergeAndCenterColumnHeaders	When set to False, column headers are left justified. When set to True, column headers are centered. This applies to default Cube Views or individual column formatting in data explorer and reports. **NOTE:** MergeAndCenterColumnHeaders does not apply to Header Overrides.

Property	Description
IsRowVisible	Set to False to hide specific rows. Set this property at run-time with a parameter to show and hide detail.
TreeExpansionLevel1-4	You can automatically expand up to four nested rows in data explorer. Enter the number of rows to be expanded from 0 – 4 in the corresponding TreeExpansionLevel. See TreeExpansionLevel Property.
RowHeaderWrapText	When set to False, row header text does not wrap. When set to True, row header text displays as wrapped. This applies to individual row headers and works with Header Overrides. **NOTE:** Row Header Width must be set from the Header Size settings menu for row header text to wrap.
Font and Colors	Font and color options are available. For the FontFamily, any .Net font installed on the Excel client operating system is included in the font family. The FontSize default value is 11.

Property	Description
Excel	
ExcelMaxOutlineLevelOnRows and ExcelMaxOutlineLevelOnCols	Enter the number of outline levels needed. Up to six outline levels can be used for the creation of collapsible and expandable Excel groups of rows or columns when exporting. This relates to the Excel Indent Level setting that is applied in the header format on each row or column. The number entered here should equal the highest setting applied on the rows or columns.
ExcelExpandedOutlineLevelOnRows and ExcelExpandedOutlineLevelOnCols	When a Cube View is exported to Excel, these settings control which outline level the file is initially opened to. The default value is 1, which fully collapses each grouping.
ExcelColumnWidth	The default Excel column width in pixels unless overridden in columns.
ExcelOutlineLevelCol and ExcelOutlineLevel	The default Excel outline level for rows and columns is 1 unless specified here or overridden in rows or columns.

Property	Description
ExcelHorizontalAlignment and ExcelVerticalAlignment	Control how the data is aligned horizontally and vertically using the Excel standard alignment options.
ExcelIndentLevel	The number of characters to indent the text or value.
ExcelWrapText	This determines if text like headers will be wrapped in the output.
Reporting	

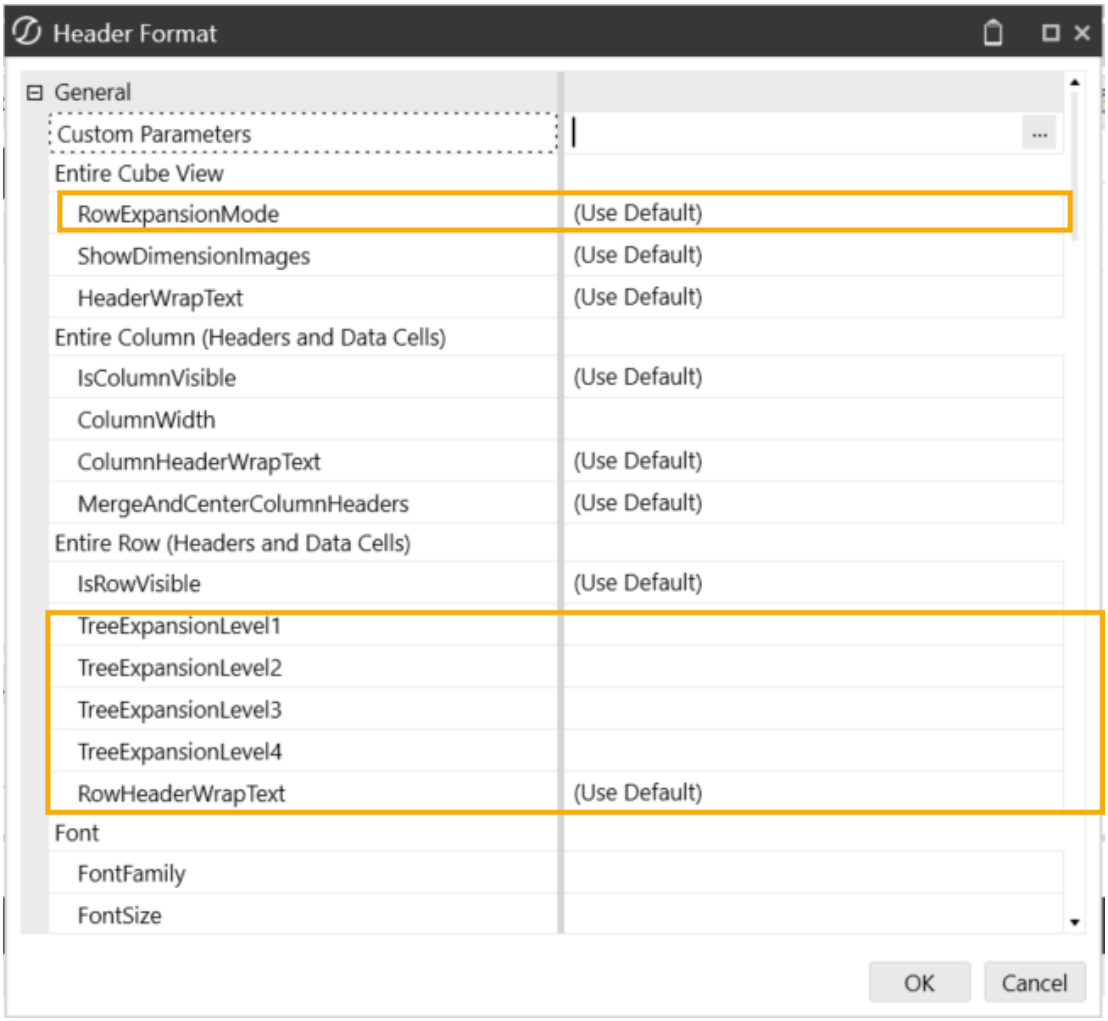
Property	Description
ReportSeparateBandPerCubeViewRow	This setting optimizes performance for large Cube Views that will be dynamically generating a data explorer report with many row definitions with unique formatting. The Default setting is (Use Default), which is a conditional setting. This applies when a data explorer report is run from a Cube View and creates background temporary database tables in memory, one for each row name in the Cube View. If there are more than 100 row definitions, fewer report bands are created and rows with the same formatting are combined. Report bands are separate, temporary in-memory tables. Set this property to True to create a separate table per Cube View row definition. This may affect performance with hundreds or thousands of row definitions. Set this property to False to combine row definitions of the same format into fewer temporary in-memory tables.
ReportColumnWidth and ReportRowHeight	This sets the width of a column or the height of a row by number of pixels.

Property	Description
ReportTopLinesOnFirstRowOnly and ReportBottomLinesOnLastRowOnly	If the row definition results in multiple rows, a top line is placed on the first line only or a bottom line will be placed on the last line only.
ReportBandHeightBeforeFirstRow and ReportBandHeightAfterLastRow	This enables a number to be entered in pixels to add before the first row or after the last row of where a row definition starts. **NOTE:** The top and bottom report bands used for lines and spacing are removed for suppressed rows or if there are no data rows.
ReportRowPageBreak	This applies a page break where appropriate for this row.
ReportRowContentTop	This specifies the vertical position of the row relative to the row above it. If this is set to zero, the top of the row immediately follows the row above it with no vertical spacing.
ReportRowContentHeight	The height of the row in pixels.
ReportRowPaddingTop and ReportRowPaddingBottom	Control the extra space above and below a row.

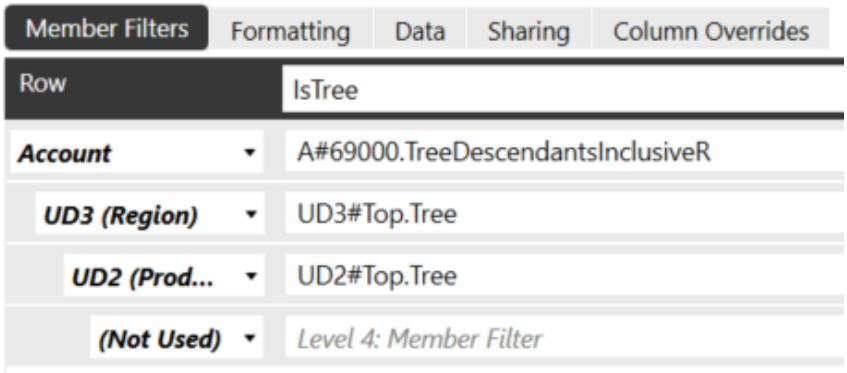
Property	Description
ReportFontSize	This is the point size of the font on the report if it differs from the font size displayed in data explorer.
ReportTextAllignment	This controls how data is aligned in a report.
ReportUnderline	This creates a simple underline of report values.
Top Lines and Bottom Lines	These are used for underlines and overlines. The Line1 and Line2 options can be used together to create a double underline or double overline. For the lines to have a small gap between each column, use the ReportBottomLine1PaddingLeft property or one similar.
Borders	This enables a line to be drawn around the border of a cell.

TreeExpansionLevel Property

For TreeExpansionLevel to work, RowExpansionMode from the Default Header must be set to Use Default. Using Collapse All or Expand All overrides this setting.

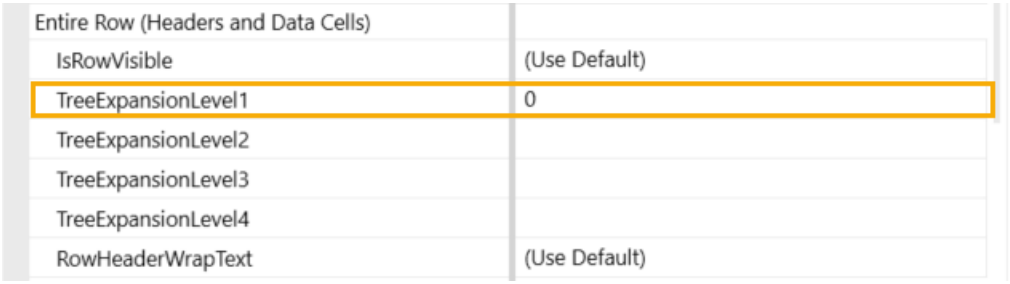


TreeExpansionLevels are tied to the Member Filter dimension types in the Designer tab and require a Tree Member Expansion.



NOTE: Default settings are overwritten by Row Overrides when set.

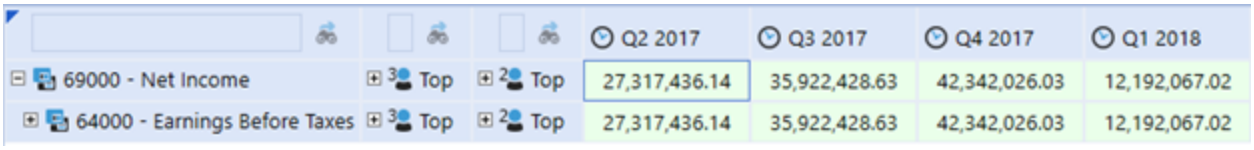
Set the TreeExpansionLevel to zero to collapse all nested rows.





Set the TreeExpansionLevel to one to expand the first level of nested rows.

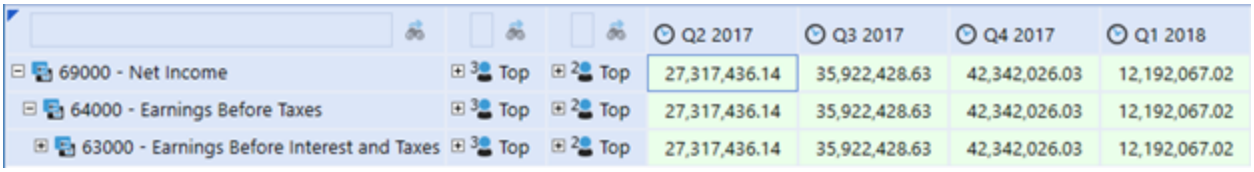
Entire Row (Headers and Data Cells)	
IsRowVisible	(Use Default)
TreeExpansionLevel1	1
TreeExpansionLevel2	
TreeExpansionLevel3	
TreeExpansionLevel4	
RowHeaderWrapText	(Use Default)



			Q2 2017	Q3 2017	Q4 2017	Q1 2018
69000 - Net Income	Top	Top	27,317,436.14	35,922,428.63	42,342,026.03	12,192,067.02
64000 - Earnings Before Taxes	Top	Top	27,317,436.14	35,922,428.63	42,342,026.03	12,192,067.02

Set the TreeExpansionLevel to two to expand the first two levels of nested rows.

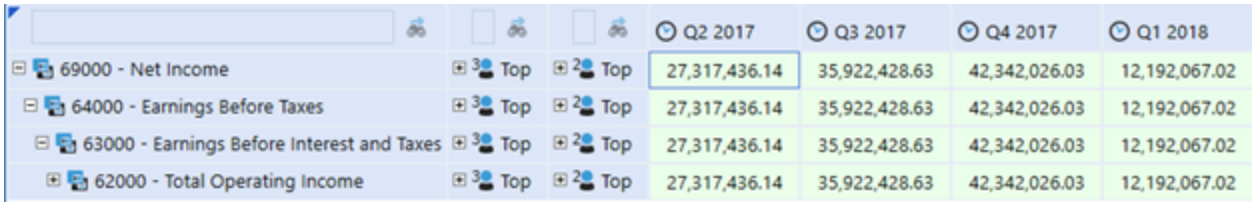
Entire Row (Headers and Data Cells)	
IsRowVisible	(Use Default)
TreeExpansionLevel1	2
TreeExpansionLevel2	
TreeExpansionLevel3	
TreeExpansionLevel4	
RowHeaderWrapText	(Use Default)



			Q2 2017	Q3 2017	Q4 2017	Q1 2018
69000 - Net Income	Top	Top	27,317,436.14	35,922,428.63	42,342,026.03	12,192,067.02
64000 - Earnings Before Taxes	Top	Top	27,317,436.14	35,922,428.63	42,342,026.03	12,192,067.02
63000 - Earnings Before Interest and Taxes	Top	Top	27,317,436.14	35,922,428.63	42,342,026.03	12,192,067.02

Set the TreeExpansionLevel to three to expand the first three levels of nested rows.

Entire Row (Headers and Data Cells)	
IsRowVisible	(Use Default)
TreeExpansionLevel1	3
TreeExpansionLevel2	
TreeExpansionLevel3	
TreeExpansionLevel4	
RowHeaderWrapText	(Use Default)



			Q2 2017	Q3 2017	Q4 2017	Q1 2018
69000 - Net Income	Top	Top	27,317,436.14	35,922,428.63	42,342,026.03	12,192,067.02
64000 - Earnings Before Taxes	Top	Top	27,317,436.14	35,922,428.63	42,342,026.03	12,192,067.02
63000 - Earnings Before Interest and Taxes	Top	Top	27,317,436.14	35,922,428.63	42,342,026.03	12,192,067.02
62000 - Total Operating Income	Top	Top	27,317,436.14	35,922,428.63	42,342,026.03	12,192,067.02

This example shows the TreeExpansionLevel 1, 2, and 3 expanded.

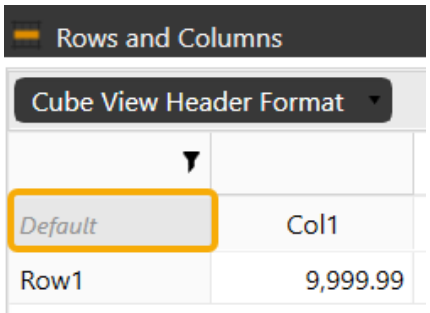
Entire Row (Headers and Data Cells)	
IsRowVisible	(Use Default)
TreeExpansionLevel1	1
TreeExpansionLevel2	1
TreeExpansionLevel3	1
TreeExpansionLevel4	
RowHeaderWrapText	(Use Default)

			Q2 2017	Q3 2017	Q4 2017	Q1 2018
69000 - Net Income	Top	Top	27,317,436.14	35,922,428.63	42,342,026.03	12,192,067.02
		Clubs	49,143,137.47	64,623,225.86	76,171,863.15	22,312,036.97
		Parts	-619,440.76	-814,564.61	-960,133.18	-295,648.79
		Subassemblies	126,520.26	166,374.14	196,106.40	56,124.75
		None	-21,332,780.83	-28,052,606.77	-33,065,810.33	-9,880,445.92
	Americas	Top	64,313,347.73	84,572,052.35	99,685,689.05	29,476,951.01
		Clubs	64,313,347.73	84,572,052.35	99,685,689.05	29,476,951.01
	Europe, Middle East, Africa	Top	9,458,612.29	12,438,075.17	14,660,849.07	4,335,197.30
		Clubs	9,458,612.29	12,438,075.17	14,660,849.07	4,335,197.30
	Asia Pacific	Top	12,768,362.19	16,790,396.30	19,790,961.40	5,852,166.00
		Clubs	12,768,362.19	16,790,396.30	19,790,961.40	5,852,166.00
	None	Top	-59,222,886.07	-77,878,095.18	-91,795,473.49	-27,472,247.29
		Clubs	-37,397,184.74	-49,177,297.95	-57,965,636.37	-17,352,277.34
		Parts	-619,440.76	-814,564.61	-960,133.18	-295,648.79
		Subassemblies	126,520.26	166,374.14	196,106.40	56,124.75
		None	-21,332,780.83	-28,052,606.77	-33,065,810.33	-9,880,445.92
64000 - Earnings Before Taxes	Top	Top	27,317,436.14	35,922,428.63	42,342,026.03	12,192,067.02

Cell Format Properties

The format properties for a cell include font, colors, background, and grid lines.

1. In the **Rows and Columns** slider, select the **Default** cell to apply the formatting settings to all Cube View cells.



> **NOTE:** You can select an individual cell instead if you want to edit the settings for a specific cell.

1. In the **Formatting** tab, select the ellipsis to the right of **Cell Format**.
2. In the **Cell Format** dialog box, select the **Format** button to modify options. The Excel section only affects how the Cube View data display in Excel.

Review the following Cell Format properties:

Property	Description
General	
Custom Parameters	Select the ellipsis to select and assign a custom parameter to the cell, such as a Cube View style.

Property	Description
NumberFormat	Uses the Microsoft .NET standard number format syntax, which allows different formats to be specified for positive and negative numbers separated by a semi-colon. This is not available in Excel. Review the following examples: #,### ;(#,###);0 will show the number using a comma as the thousands separator, no degrees of precision, parenthesis around negative numbers, and a zero for null values. The pattern of this format choice is Positive;Negative;Null values with each separated by a semicolon. There is a space after the first #,###. This enables the numbers to line up with negative numbers due to the parenthesis. #,###.#% ;(#,###.#%);" – " will show a percentage with a comma as the thousands separator, one degree of precision, negative percentages in parenthesis, and a dash for null values. N2 will show the data as a numeric value with two degrees of precision. Negative numbers display with a minus sign. P1 will show a percentage with one degree of precision.

Property	Description
ZeroOffsetForFormatting	This is related to NegativeTextColor, which is determined by whether a number is less than zero. For example, if sales were less than 100, rather than 0, they could be displayed in red. A valid setting is any number other than zero.
Scale	The valid values for the scale are -12 to +12. For example, to show a number in thousands, the scale should equal three, or for millions, the scale should equal six.
FlipSign	This flips the display value between positive and negative. This is useful for reports where certain expense numbers are stored as positive or negative and need to be shown on the report.
ShowPercentageSign	This determines whether a percentage sign is displayed.
ShowCurrency	This shows the currency code, for example, EUR. This is not available in Excel.
FontFamily	Any .Net font installed on the Excel client operating system is included in the font family.
FontSize	The default font size is 11.

Property	Description
NegativeTextColor, WritableBackgroundColor, and SelectedGridLinesColor	These can optionally override the format for negative numbers, writeable data cells, and selected data cells.
Excel	
ExcelNumberFormat	Apply this to control how numbers display when exporting to Excel. A number format can have up to four sections of code separated by semicolons. These code sections define the format for positive numbers, negative numbers, zero values, and text:<POSITIVE>;<NEGATIVE>;<ZERO>;<TEXT> You can include an underscore to create a space after the trailing positive number, and you can control the color of the negative number: #,##0.00_);[Red](#,##0.00);0.00 **NOTE:** ExcelNumberFormat settings [#,##0,,.0 ;[Red](#,##0,,.0)] on a Cube View scale numbers when the Cube View is exported to Excel.
ExcelUseScale	This determines if Excel uses the scale property.
Reporting	

Property	Description
ReportNoDataNumberFormat	By default, NoData (null) cells display as empty in the data explorer and as zeros in a report. However, any .NET number format text can be specified to format those zero values differently in the report. For example, you can type NODATA in the setting. To display empty text, you can type # in the setting. If the number format setting of the cell already does something with null values, this property does not need a value.

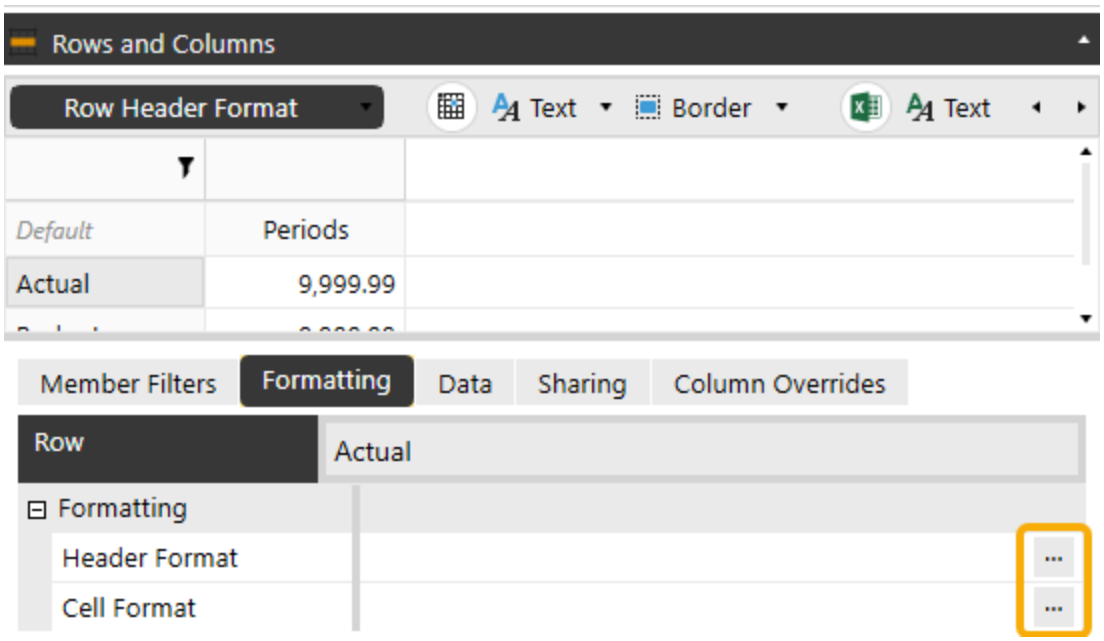
Property	Description
ReportUseNumericBinding and ReportNumericBindingFormat	These are related to the export of dashboard reports to Excel, so numbers can be rendered in the proper number format rather than display in Excel as text. Set ReportUseNumericBinding to True to use numeric amounts instead of OneStream text-based formatting when generating a Cube View report. The ReportNumericBindingFormat setting must follow a specific syntax. This number format is related to the report engine,. Excel standard number formats are not used. For example, this represents a number format for positive, negative, and no values:{0:#,# ;(#,#);" - "} This generates numbers instead of text when exporting a dashboard report to Excel. However, it cannot be used for calculation status and annotation data cells because those features cannot be represented as numbers. In those cases, use column overrides to display the values as text.

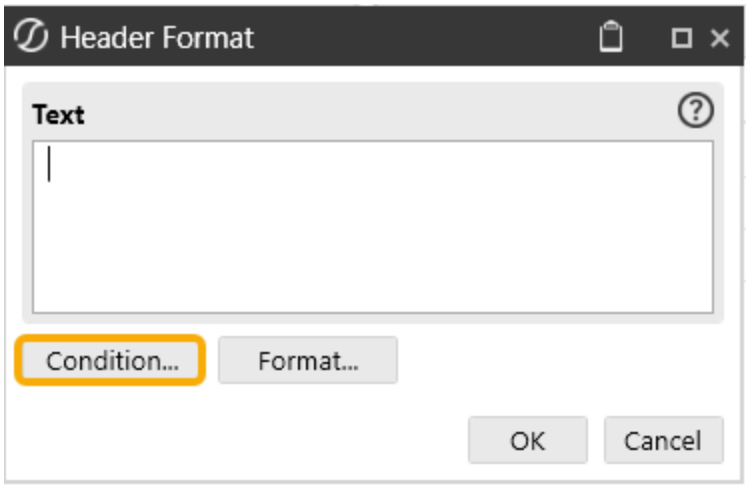
NOTE: For position, font, color, top line, bottom line, and border properties, see Header Format Properties.

Create Conditional Formatting

Conditional formatting enables you to format headers or cells based on defined criteria. Conditional formatting in Cube Views supports data analysis by highlighting cells or ranges of cells, identifying key values, and using data bars, color scales, and icon sets that correspond to specific variations in the data. Conditional formatting is applied and visible in data explorer, Excel, and report views.

To apply conditional formatting to rows or columns, go to **Rows and Columns** > **Formatting** > **Header/Cell Format** and select the ellipses. In the **Header/Cell Format** dialog box, select the **Condition** button.





Conditional formatting is available on following elements of a Cube View:

- Cube View default formatting
- Row and column headers
- Row and column data cells
- Row and column overrides

Conditional formatting follows the Cube View processing in the following order of operations:

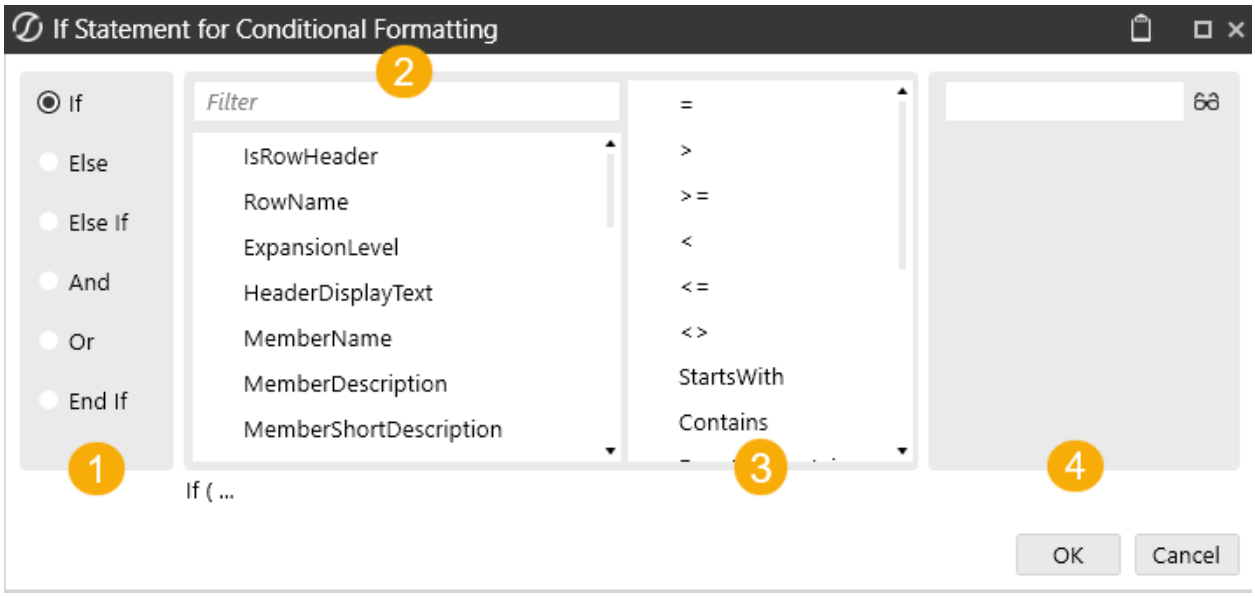
1. Number format as defined in application properties
2. Cube View default
3. Column formatting
4. Row formatting
5. Column overrides
6. Row overrides

The combination of formats and overrides equals the format for the cell when rendered. Formatting can be applied and isolated to the nested expansion levels on rows and columns by using property filters in the dialog box.

> **NOTE:** If there are any existing formats before applying conditional formatting, they will be retained if the range of cells containing the conditional formats does not meet the conditions of the rule. All styles from the Cube View and the selection styles that are applied to that range are overridden by conditional formatting.

Conditional Formatting Properties

General and conditional formatting can be applied in the Header Format and Cell Format dialog boxes. The Condition and Format buttons launch additional dialog boxes with available options. The dialog box for conditional formatting is divided into sections.



1. **Conditional statement**: Defines the options for conditional statements to apply formatting constraints.
2. **Filter**: Applies the criteria relevant to your statement.
3. **Operator**: Criteria filters for the conditional statement.
4. **Text/object**: Find the text or object to be tested.

Header Property Filters

Header Property Filter	Description
IsRowHeader	Boolean. This determines if an object is a row header field. This is appropriate for Cube View default.
IsColHeader	Boolean. This determines if an object is a column header field. This is appropriate for Cube View default.
RowName	Cube View row name
ColName	Cube View column name
ExpansionLevel	This determines the expansion level for rows 1-4 and columns 1-2. This is related to isolating the row expansion headers.
HeaderDisplayText	Custom descriptions used with the :Name() function
MemberName	Metadata member labels
MemberDescription	Metadata member descriptions

Header Property Filter	Description
MemberShortDescription	Metadata member short descriptions. This is only the Time dimension.
IndentLevel	The indentation level derived as a formatting setting or system-generated based on tree expansions.

Expansion Specific Property Filters

Expansion Specific Property Filter	Option	Description
RowE expansion level - criteria	Row expansions 1–4	Identify nested or expansion rows with specified criteria
ColE expansion level - criteria	Column expansion 1 and 2	Identify nested or expansion columns with specified criteria

Cell Format Property Filters

Cell Format Property Filter	Description
IsNoData	Test for no data
IsRealData	Test for stored data, ignoring derived Zero-View data

Cell Format Property Filter	Description
IsDerivedData	Test for derived data, commonly resulting from Scenario Zero-View settings
IsRowNumberEven	Test for the row number as an expansion or on a fixed row
ExpandedRowNum	Test the count of expanded rows, Zero-based. This property filter is based on the total Cube View count of rows generated from each row and its expansions.
CellAmount	Test cell data amount
CellStorageType	Test the method used to store data

Define Properties for Conditional Formatting

Conditional formatting can be applied based on a data point. Cell formatting conditions can occur based on member names or descriptions, indent levels, row and column names, and expanded and even numbered rows.

Conditional formatting can also be applied based on members. The ability to format based on the name or description is enhanced by using a standardized metadata naming convention. Summary level members with keywords such as Total, or prefixes or suffixes such as Tot, can be used in conditional formatting. Members can be formatted using dynamic criteria, such as StartsWith or EndsWith. HeaderDisplayText differs from the MemberName and MemberDescription property filter because it references the custom name parameter in a member filter. Dynamic criteria can be applied to the name and description property filters to apply the required formatting.

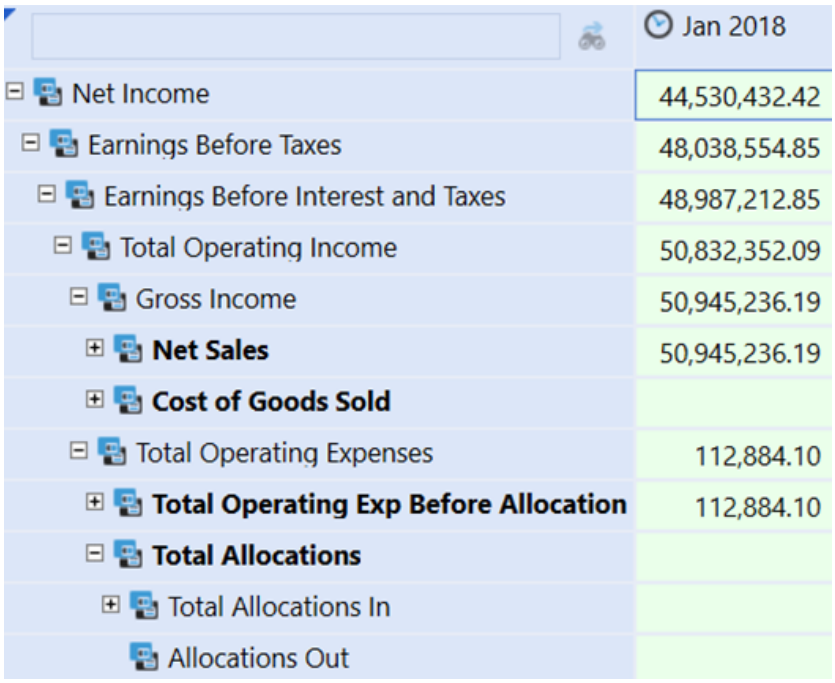
1. Select the **Formatting** tab.
2. Select the ellipsis to launch the dialog box.
3. Select the **Condition** button, set the requirements, and then select the **OK** button.
4. Select the **Format** button, set the requirements, and then select the **OK** button.
5. Select **Condition** and then **End If** to close the condition.
6. Select the **OK** button.

Examples Applying Conditional Formatting

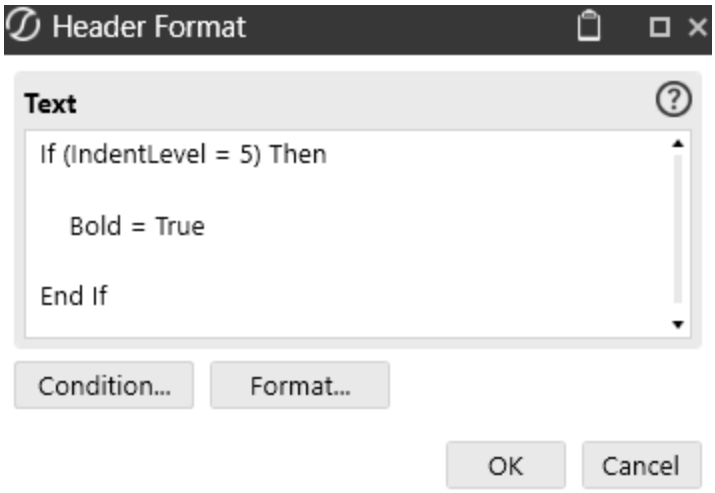
The following examples show common usages of conditional formatting.

Indent Level

The IndentLevel Property filter dynamically formats from defined rows or expansions. Indentation is zero-based. The formatting can be applied to the default or to rows. This can speed up formatting for summary level dimension members.



Apply the formatting to the Cube View's Header section or to specific rows.



Using Conditional Property Filters For Conditional Traffic-Lighting

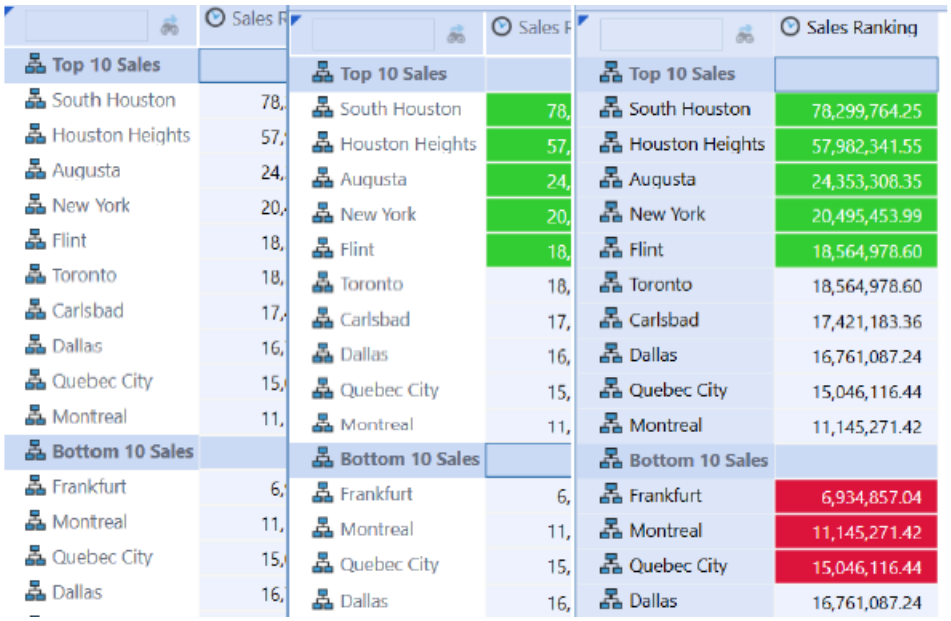
Traffic-Lighting is data related and applied as a cell format. The designer can apply the conditional formatting to either a row or a column. The order of operations for formatting can impact the decision. Row overrides are the final layer of formatting applied to a Cube View and are not impacted by general formatting.

	Actual	Plan	BW Diff	Var %
	Jan 2018	Jan 2018	Jan 2018	Jan 2018
60000 - Operating Sales	57,982,341.55	60,275,344.46	-2,293,002.91	-3.80
2000_100 - Third Party Sales	50,945,236.19	54,511,402.72	-3,566,166.53	-6.54
2000_200 - OEM Sales	4,606,156.98	4,145,541.28	460,615.70	11.11
2000_300 - Subassembly Sales	382,840.04	287,130.03	95,710.01	33.33
2000_400 - Parts Sales	2,048,108.34	1,331,270.42	716,837.92	53.85

A definition applied in the row Formatting tab would apply to all columns. A row override would isolate the formatting to specified columns. The CellAmount filter is used within multiple If/ElseIf statements to define the various tests required for the report.

ExpandedRowNum Expansion Range

Conditional formatting enables you to format an expanded range of data cells using the ExpandedRowNum property filter. This supports Top 10 type reports. This filter best supports Cube View designs that use known expansions, such as ranking business rules or member lists. If applied to specific rows, the formatting applies to the defined row, but the Expansion Number Reference relates to the entire Cube View. Formatting defined on subsequent rows is impacted if the expansion members change on previous rows. The ExpandedRowNum property filter can also be applied as a default Cube View format, in which the definition applys to all rows. This would require If/ElseIf type statements to support all rows.



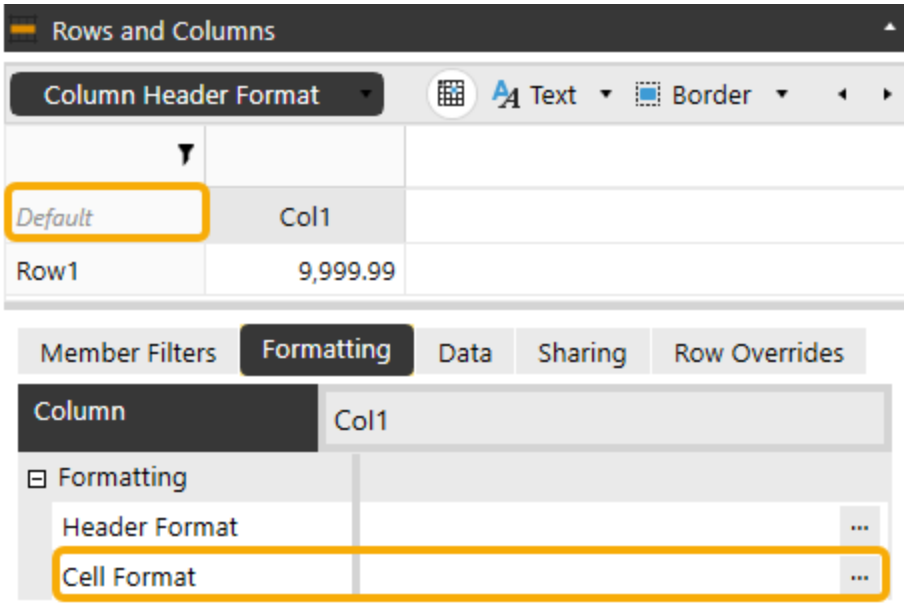
	Sales R
Top 10 Sales	
South Houston	78,
Houston Heights	57,
Augusta	24,
New York	20,
Flint	18,
Toronto	18,
Carlsbad	17,
Dallas	16,
Quebec City	15,
Montreal	11,
Bottom 10 Sales	
Frankfurt	6,
Montreal	11,
Quebec City	15,
Dallas	16,

	Sales F
Top 10 Sales	
South Houston	78,
Houston Heights	57,
Augusta	24,
New York	20,
Flint	18,
Toronto	18,
Carlsbad	17,
Dallas	16,
Quebec City	15,
Montreal	11,
Bottom 10 Sales	
Frankfurt	6,
Montreal	11,
Quebec City	15,
Dallas	16,

	Sales Ranking
Top 10 Sales	
South Houston	78,299,764.25
Houston Heights	57,982,341.55
Augusta	24,353,308.35
New York	20,495,453.99
Flint	18,564,978.60
Toronto	18,564,978.60
Carlsbad	17,421,183.36
Dallas	16,761,087.24
Quebec City	15,046,116.44
Montreal	11,145,271.42
Bottom 10 Sales	
Frankfurt	6,934,857.04
Montreal	11,145,271.42
Quebec City	15,046,116.44
Dallas	16,761,087.24

Complete the following steps:

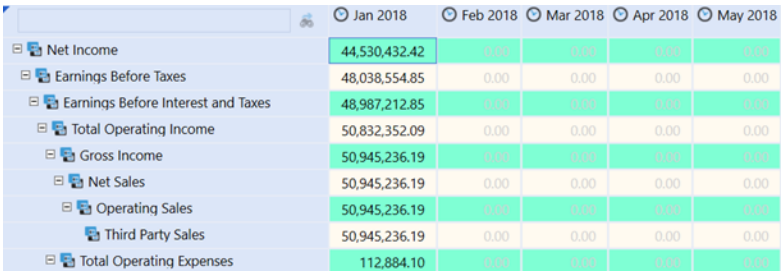
1. Design or open a report that supports formatting for ExpandedRowNum, such as a ranking report.
2. Determine if the formatting should be applied as default, row, or column. The following example uses the Cube View default formatting.



3. Cell format is applied for conditional formatting using the ExpandedRowNum filter. Because this is zero-based and must account for each row and its expansions:
    a. Condition1 – Rows < 6 is defined because the text header row Top10Title initiates the count at zero.
    b. Condition2 – Rows > 11 and < 15 is defined because all rows up to Row4, SalesBottom10, reflect rows 0-11 in the Cube View. The conditional row references must reflect all Cube View expansions.

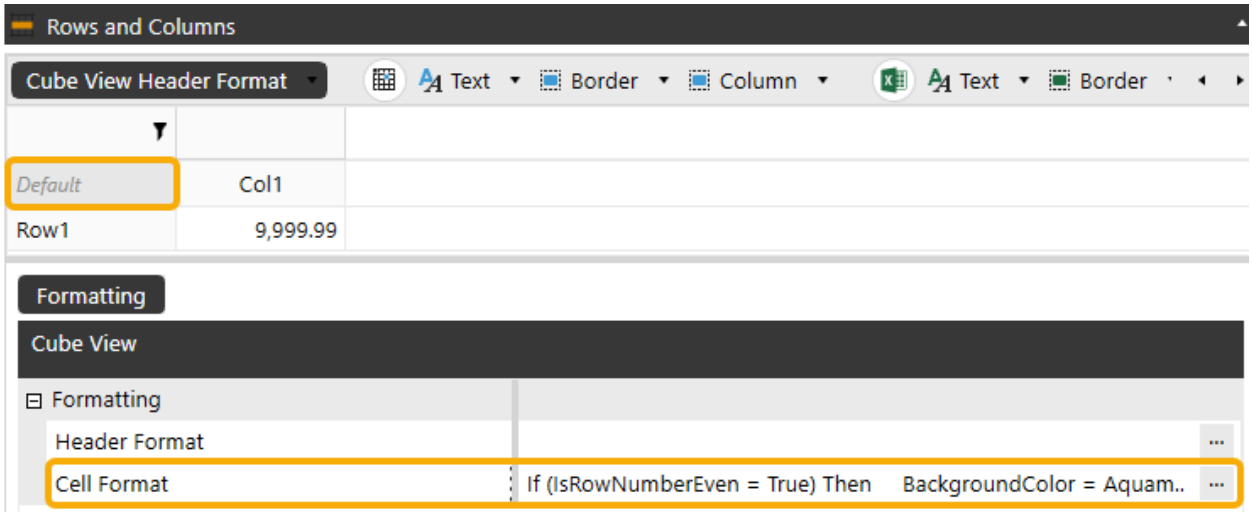
IsRowNumberEven

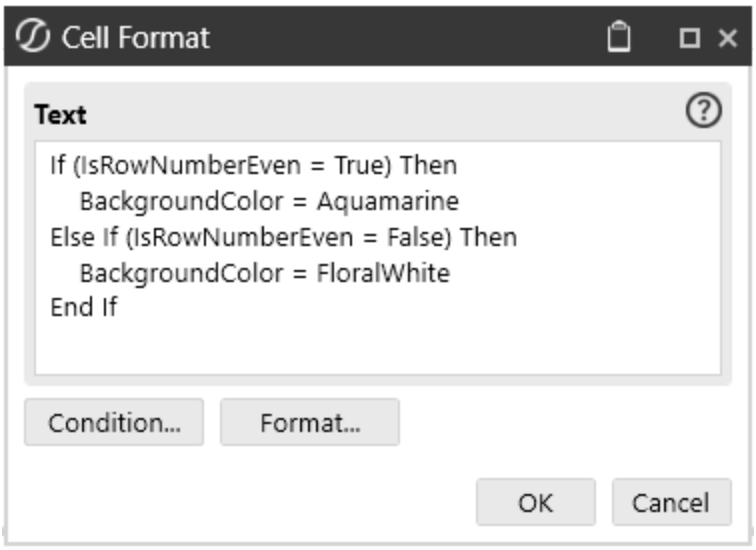
The IsRowNumberEven property filter can enable you to vary formatting by the even/odd numeric expansion of rows in a Cube View. For example, the IsRowNumber Even is a suitable filter for a green-bar style report.



	Jan 2018	Feb 2018	Mar 2018	Apr 2018	May 2018
Net Income	44,530,432.42	0.00	0.00	0.00	0.00
Earnings Before Taxes	48,038,554.85	0.00	0.00	0.00	0.00
Earnings Before Interest and Taxes	48,987,212.85	0.00	0.00	0.00	0.00
Total Operating Income	50,832,352.09	0.00	0.00	0.00	0.00
Gross Income	50,945,236.19	0.00	0.00	0.00	0.00
Net Sales	50,945,236.19	0.00	0.00	0.00	0.00
Operating Sales	50,945,236.19	0.00	0.00	0.00	0.00
Third Party Sales	50,945,236.19	0.00	0.00	0.00	0.00
Total Operating Expenses	112,884.10	0.00	0.00	0.00	0.00

In this example, the formatting is applied to the Cube View default cell format since it globally applies to all rows.

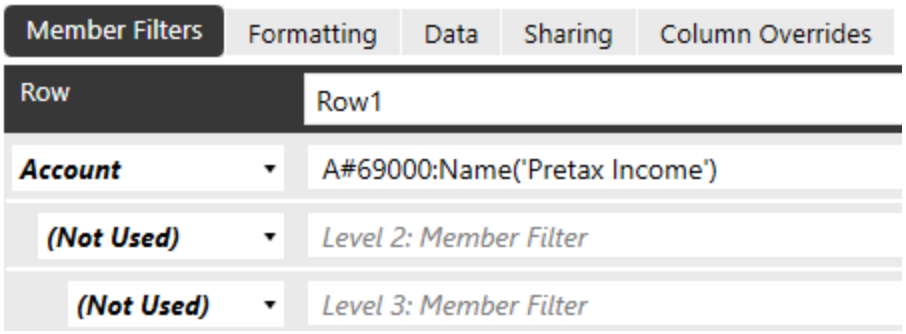




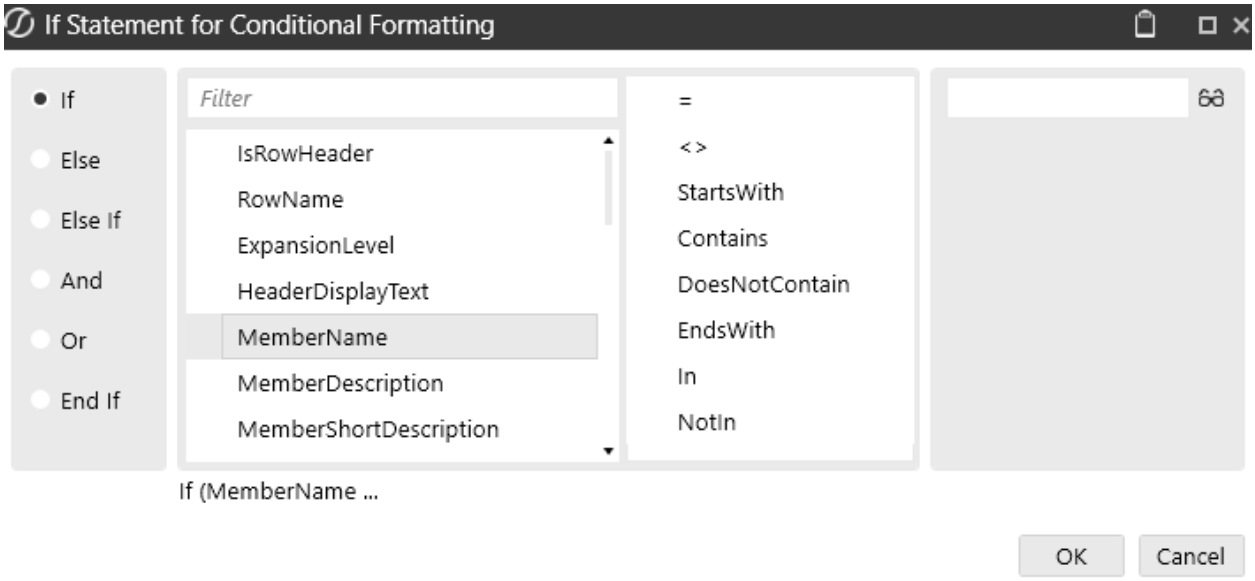
MemberName, MemberDescription, and HeaderDisplayText

Following a standardized metadata naming convention enhances the ability to format based on name or description. Summary level members with keywords such as Total, or prefixes or suffixes such as Tot, can be used in conditional formatting. Members can be formatted using dynamic criteria, such as StartsWith or EndsWith.

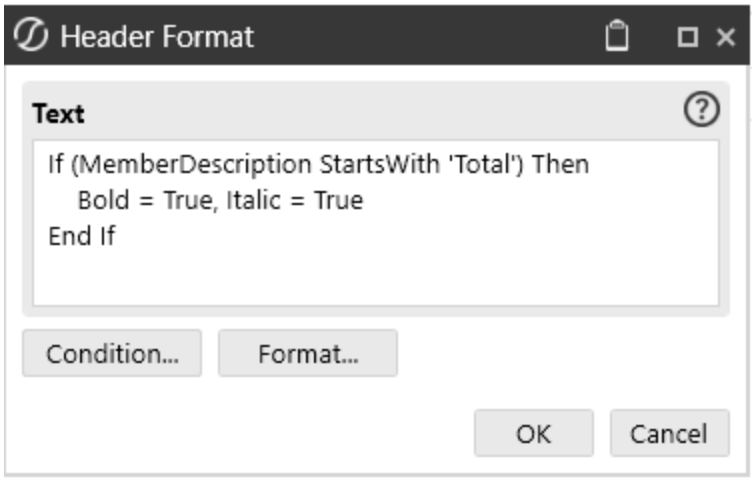
HeaderDisplayText differs from the MemberName and MemberDescription property filters in that it references the custom Name parameter in a Member Filter.



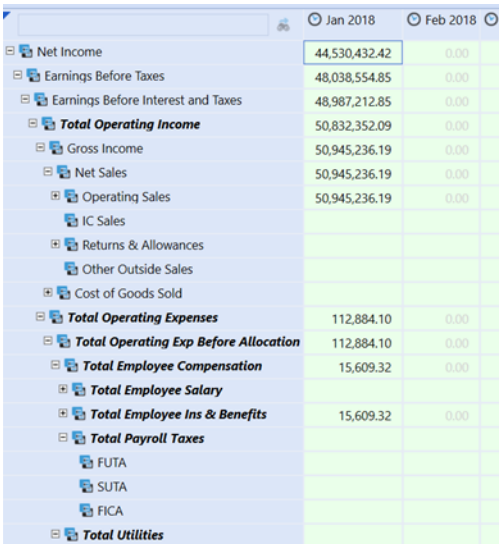
Dynamic criteria can be applied to the MemberName and MemberDescription property filters to apply the required formatting.



Apply conditional formatting to the row header.



The result is a dynamically formatted report.

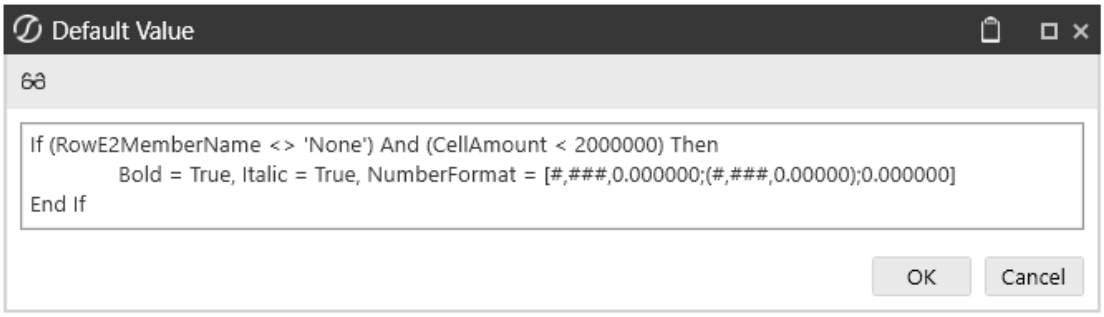


	Jan 2018	Feb 2018
Net Income	44,530,432.42	0.00
Earnings Before Taxes	48,038,554.85	0.00
Earnings Before Interest and Taxes	48,987,212.85	0.00
Total Operating Income	50,832,352.09	0.00
Gross Income	50,945,236.19	0.00
Net Sales	50,945,236.19	0.00
Operating Sales	50,945,236.19	0.00
IC Sales		
Returns & Allowances		
Other Outside Sales		
Cost of Goods Sold		
Total Operating Expenses	112,884.10	0.00
Total Operating Exp Before Allocation	112,884.10	0.00
Total Employee Compensation	15,609.32	0.00
Total Employee Salary		
Total Employee Ins & Benefits	15,609.32	0.00
Total Payroll Taxes		
FUTA		
SUTA		
FICA		
Total Utilities		

Parameter Formatting

Conditional formatting definitions can be applied to a Cube View as a Literal parameter. To apply this, select the format string from the Cube View and press **Ctrl-C**. In a parameter with a Literal Value parameter type, select the **Default Value** field and press **Ctrl-V**. Once saved, this parameter can be referred to in a Cube View format. In this example, the reference of |!Param_ Conditional_PLTree!| in a Cube View format applies the associated format string.

General (Parameter)	
Name	Param_Conditional_PLTree
Workspace	Test (TST)
Maintenance Unit	TestMU (TST)
Description	|!Param_Conditional_PLTree!|
User Prompt	
Sort Order	0
Data Source	
Parameter Type	Literal Value
Default Value	If (RowE2MemberName <> 'None') And (CellAmount < 2000000) Then



Formatting

Cube View

Formatting	
Header Format	
Cell Format	|!Param_Conditional_PLTree!|

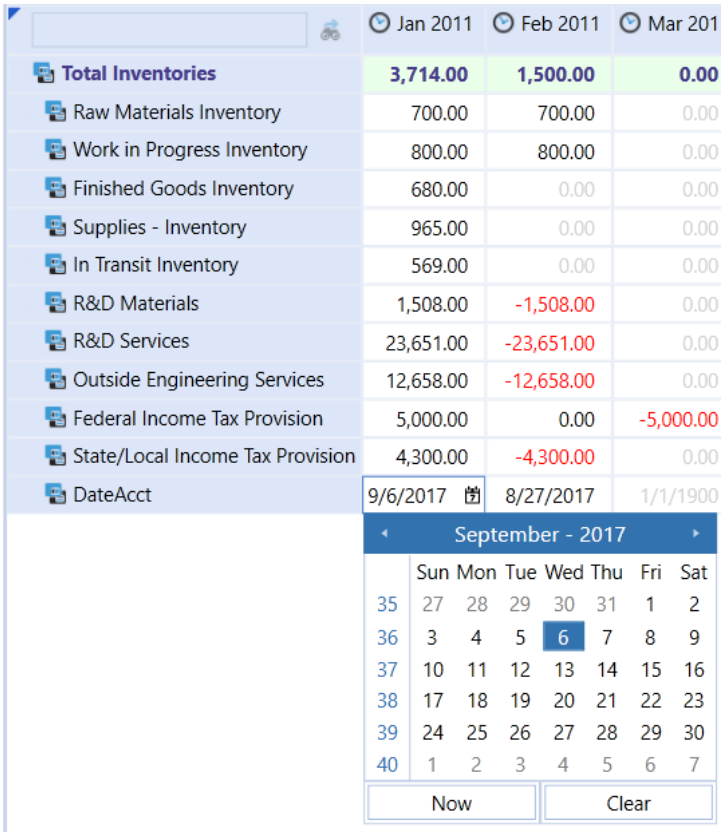
Rows and Columns Data Tab

General

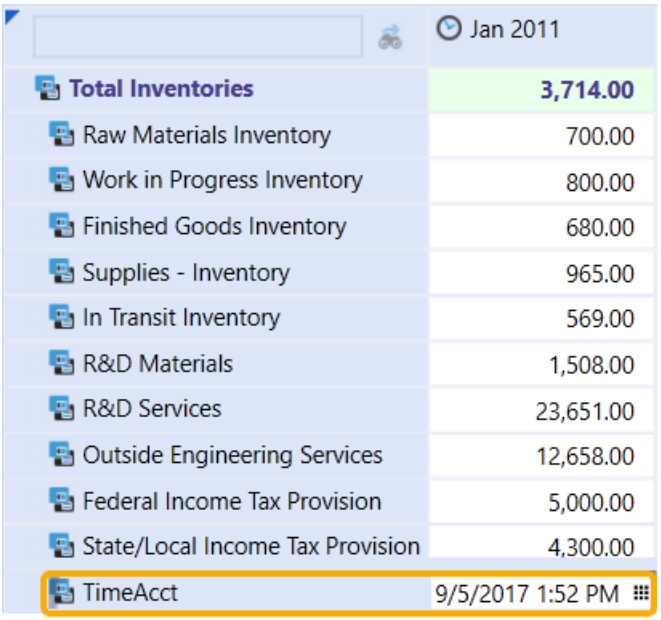
Can Modify Data
When set to False, specific rows or columns are read-only. When set to True, the False setting will override it. However, if Can Modify Data is False, the entire Cube View is read-only and this setting will not override if True.

Cell Type
Select one of the following cell types:

- **Text Box**: This makes the Cube View cell numerical. Only numerical values can be entered in the data cell at run-time. This is the default setting.
- **Combo Box**: Select this when a List Parameter is used in a Cube View data cell. Enter the name of the List Parameter in the List Parameter property.
- **Date**: Enables a calendar in the Cube View cell. You can select a specific date in the data cell.



- **Date Time**: Enables a calendar and time display. You can select a specific date and time in a data cell.

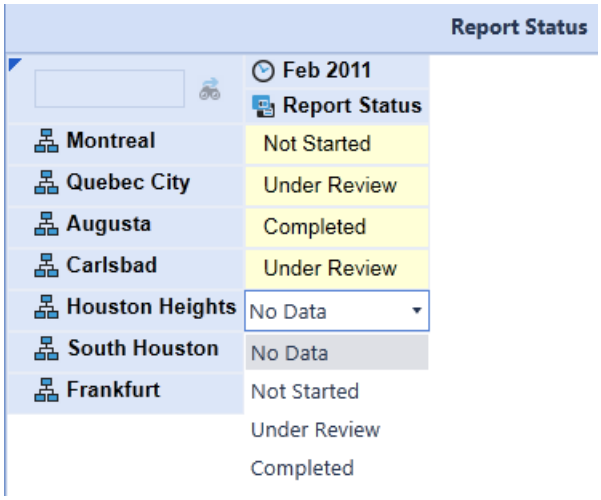


	Jan 2011
Total Inventories	3,714.00
Raw Materials Inventory	700.00
Work in Progress Inventory	800.00
Finished Goods Inventory	680.00
Supplies - Inventory	965.00
In Transit Inventory	569.00
R&D Materials	1,508.00
R&D Services	23,651.00
Outside Engineering Services	12,658.00
Federal Income Tax Provision	5,000.00
State/Local Income Tax Provision	4,300.00
TimeAcct	9/5/2017 1:52 PM

> **NOTE:** Selecting Now in the calendar enters the current date or current date and time.

List Parameter

Enter a parameter to create a drop-down list of choices in a Cube View data cell. The Cell Type property must be set to Combo Box for this field to be enabled.



	Report Status
	Feb 2011
	Report Status
Montreal	Not Started
Quebec City	Under Review
Augusta	Completed
Carlsbad	Under Review
Houston Heights	No Data
South Houston	No Data
Frankfurt	Not Started
	Under Review
	Completed

Navigation Links

Enable Report Navigation Link (Rows Only)
When set to True, this enables Navigation Link Drill Downs in dashboards. Enter the name of the dashboard to display in the Dashboard to Open in Dialog field. Set to False if this function is not used for the Cube View. See Configuring Navigation Links for Linked Dashboards.

Dashboard to Open in Dialog (Rows Only)
The name of the dashboard or parameter used for the report navigation link. Use a parameter for a Cube View acting as a row template across multiple Cube Views.

Linked Cube Views
Enter a comma-separated list of Cube Views. These will be available for you to open when viewing a Cube View in the Data Explorer Grid. The Cube Views specified here apply to the selected row or column and display when you right-click on any data cell within that row or column.

> **NOTE:** Row settings override column settings which override Cube View settings.

General Settings

This section explains how to view, define, and apply standard report settings. These settings determine what information appears in the Cube View and which actions you can perform. To manage these settings, select a Cube View and select the **General Settings** slider.

Sharing

Share all or specific rows or columns from another Cube View. Sharing can increase efficiency and consistency when building and managing Cube Views.

Common

These options control general, shortcut, dynamic, processing, restrictions, suppression, and paging properties.

Is Visible In Profiles
When set to True, the Cube View is visible in the Cube View Profile. When set to False, the Cube View is not visible within the Cube View Profile. In these cases, the Cube View would not be visible in workflow, Excel, dashboards, or OnePlace.

Page Caption
This appears at the top of the Data Explorer grid when viewing the output of a Cube View. If left blank, it gets its value from the Cube View description. If the description is blank, it displays the Cube View name.

Is Shortcut
This determines whether the Cube View is a shortcut.

Shortcut Cube View Name
Enter the name of another Cube View that will open when this Cube View is used as a shortcut.

Is Dynamic
Set this property to True to make the Cube View dynamic. When set to True, the developer can programmatically manipulate Cube View formatting and data at run-time through the Dynamic Cube View service type.

Literal Parameter Values
Enter a comma-separated list of name–value pairs that provide parameter values when this Cube View is used as a shortcut to open another Cube View. Piped variable names cannot be used, for example, |!ParamView!| in this setting.

> **Example:** Param1=Value1, Param2=[Value 2]

Can Modify Data

If set to True, the Cube View can be modified in OnePlace. If set to False, the Cube View is read only.

> **NOTE:** You can still make annotations to accounts in a Cube View if this property is set to False.

Can Calculate, Can Translate, and Can Consolidate

These can be set to True or True (Exclude Force). If set to True, you can right-click on a cell and calculate, translate, or consolidate data in OnePlace. If set to True (Exclude Force), force actions, such as Force Consolidate, and their logging variants display in the user interface but are disabled. You can run standard consolidations, translations, or calculations without forcing them. This setting applies to the OneStream Windows application, the Modern Browser Experience, the Excel Add-In, and Cube View connections.

> **NOTE:** These properties coincide with the Can Calculate from Grids Scenario security group. If set to True, but the user is not in the group for that given scenario member, they cannot calculate from a Cube View grid.

Force Consolidate, Force Calculate, and Force Translate

These control who can perform force operations independently of access to standard consolidate/calculate/translate operations through security groups. Users in the security group can successfully perform force operations. Users not in the security group receive an error when attempting to perform force operations. This setting applies to the OneStream Windows application, the Modern Browser Experience, the Excel Add-In, and data management. This enables administrators to control force operations at the scenario level, ensuring that only authorized users can trigger them.

Can Modify Suppression

When set to True, this enables a row suppression icon in the data explorer grid toolbar. Users can turn the suppression settings on and off when viewing a Cube View.

Include Cell Attachment Status

When set to True, cells containing a data attachment display a red tick mark. When set to False, there is no indication of a data attachment in the cell. In large data sets, this can affect performance when set to True.

> **NOTE:** This setting only applies to Cube Views in Spreadsheet and the OneStream Excel Add-In. It does not currently apply to Quick Views.

Allow Sparse Row Suppression

See Cube View Performance

Max Unsuppressed Rows Per Page and Max Seconds To Process

See Cube View Performance.

Header Text

The default is a Cube View with standard row and column headers. All header settings control the row and column headers presented when the Cube View is run if the default is not used.

Header Size

Column Header Heights

These properties align column headers with the bottom of a report. The default setting is -1, which means they are auto-sized. For bottom alignment to work in column headers, the height of the column header needs to be set. This is useful for wrapped headers, which may cause the columns to expand.

Row Header Widths

These properties are viewed in the Cube View. Excel Row Header Widths are viewed when exporting to or rendering a Cube View in Excel. Report Header Widths are viewed when a report is generated on-the-fly from a Cube View. These can be changed from the default value of -1. This default indicates that the Row Header Widths are auto-sized. When changing the value, set a positive number of at least 100 to determine the number of pixels wide (96 pixels = 2.5 cm). These settings can change for a maximum of six nested rows. The same can be done when reporting to Excel by using the ExcelRowHeader1Width property. The default value for this property is -1.

> **NOTE:** If a Cube View Row Header Width is not specified, text and font information from all expanded headers is used to automatically determine the width. The automatic row width maximum is half a page. Any text longer than half a page is wrapped.

Header Overrides

By default, the dimensions to show for row and column headers are based on the member expansions specified in the Rows and Columns slider. These dimensions can also be selected manually.

Report Column Index for Row Headers

This can be used for a report to display row headers after the column instead of on the left side. The default value is -1. If you set this to a positive number, the Cube View column index changes. Specifying several columns greater than the existing number of columns reverts the row headers back to the left side of the report. Hidden columns are not included in the column count.

> **NOTE:** If a Cube View report exceeds the width of the page, a page break is added automatically on the appropriate column and repeats the row headers on the following page. If the Report Column Index for Row Headers or Auto Fit to Page Width properties are enabled, the report does not repeat row headers.

Report

These properties apply advanced formatting to Cube Views.

Options are available to set paper size, page orientation, and margins. You can also select to automatically fit information to a specific page width or number of pages.

Custom Report Task
Custom report formatting can be applied to Cube Views using a Cube View extender business rule or inline formula. You can choose from the following options:

- **No Task**: Select this option if there are no Cube View business rules or inline formulas running on this report.
- **Execute Cube View Extender Business Rule**: Select this option to call a Cube View extender.
- **Execute Cube View Extender Inline Formula**: Select this option to include an inline formula.

Business Rule
This property is only enabled when Execute Cube View Extender business rule is selected as the custom report task. Select the ellipsis and select the Cube View Extender business rule.

Formula
This property is only enabled when Execute Cube View Extender Inline Formula is selected as the custom report task. Select the ellipsis and select an inline formula.

Excel

Display Gridlines
This controls whether gridlines are displayed in the Excel file.

Navigation Links

Navigation links are used to link dashboards or Cube Views to other dashboards or Cube Views. They can be used to launch additional content, enabling you to display related data or drill into more detail. Navigation links can be applied to the entire Cube View, or you can create individual links at specific rows or columns.

Linked Cube Views
Enter a comma-separated list of Cube Views that will be available to open when viewing a Cube View in the data explorer grid. The Cube Views specified in this field apply to the entire Cube View and are available when you right-click any data cell. See Linked Cube Views.

Linked Dashboards
Enter a comma-separated list of dashboards. Right-click to open these dashboards when viewing a Cube View in the data explorer grid.

Include Default NavLink Parameters
When set to True in source Cube Views, you can use NavLink parameters, with intuitive standard syntax, as Bound parameters for any dimension. This simplifies Cube View design and reduces error due to the following benefits:

- You do not have to define parameters for navigation. You can use the following standard syntax to ensure parameter names match across Cube Views and other objects: |!<dimension>NavLink!|

  > **Example:** |!UD8NavLink!| and |!AccountNavLink!|

- You can navigate Cube Views from any dimensional intersection to analyze data that is refined by passed point of view selections in linked Cube Views. See Linking Cube Views Through the Cube POV.

Bound Parameter Names for Navigation Links

Lists all the dimensions that may be assigned to a row when the Report Navigation Links property is set to True. This represents the member name being passed from this report to another using navigation links.

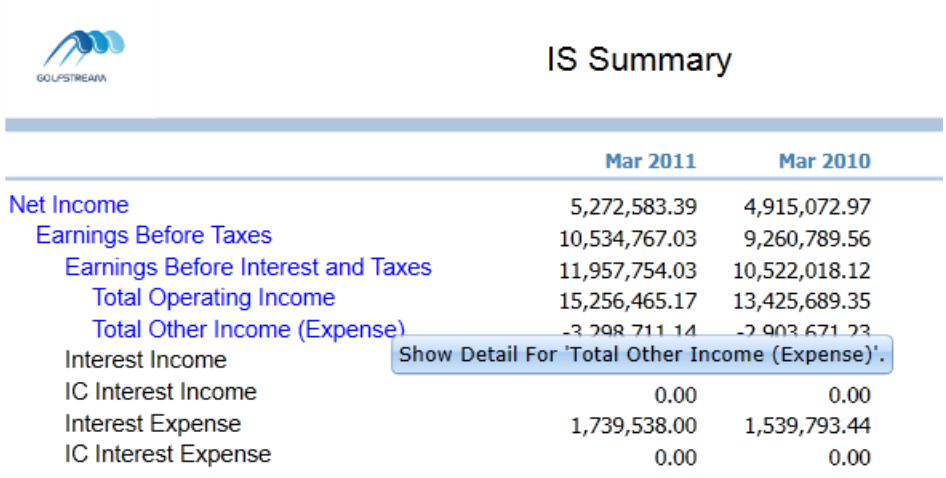
Bound Parameter Names

For a linked Cube View to open and display the correct member data for a selected cell, parameters with the member must be specified using a bound parameter. This lists all dimensions that may have a parameter included in a linked Cube View.

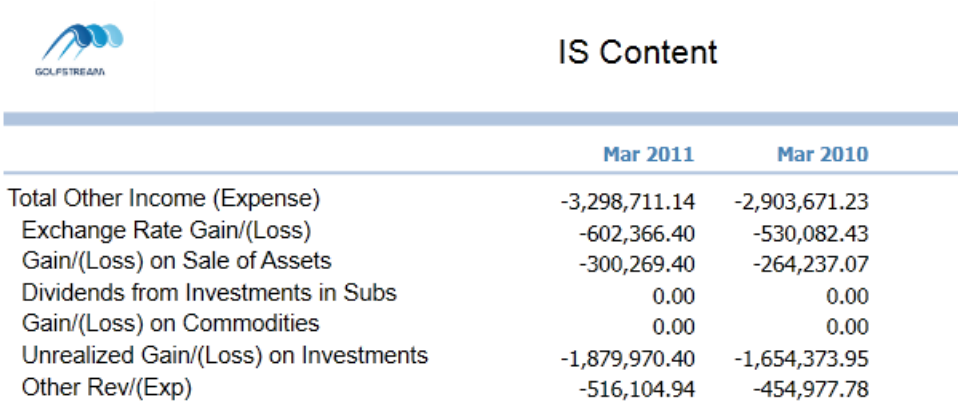
Configuring Navigation Links for Linked Dashboards

Navigation links are used to link dashboards or Cube Views to other dashboards or Cube Views. They can be used to launch additional content, enabling you to display related data or drill into more detail. Navigation links can be applied to the entire Cube View, or you can create individual links at specific rows or columns. They can also create a chart of the reviewed data.

The following section provides a detailed example and outlines the steps required to create navigation links. Although implementations may vary, this example shows an Income Statement Summary dashboard that launches another dashboard with additional detail.



When you select Total Other Income (Expense), the following dashboard displays:



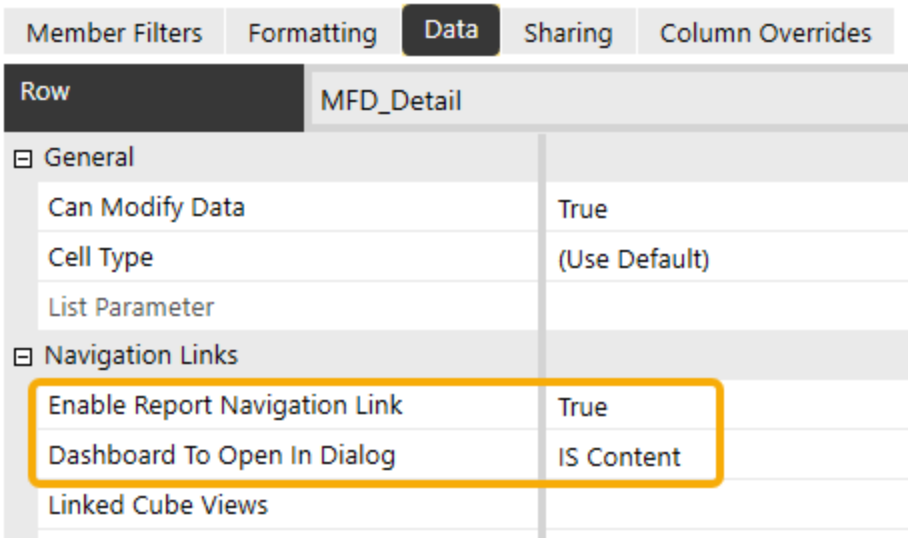
IS Content

	Mar 2011	Mar 2010
Total Other Income (Expense)	-3,298,711.14	-2,903,671.23
Exchange Rate Gain/(Loss)	-602,366.40	-530,082.43
Gain/(Loss) on Sale of Assets	-300,269.40	-264,237.07
Dividends from Investments in Subs	0.00	0.00
Gain/(Loss) on Commodities	0.00	0.00
Unrealized Gain/(Loss) on Investments	-1,879,970.40	-1,654,373.95
Other Rev/(Exp)	-516,104.94	-454,977.78

The following example is a sample Cube View called IS Summary, which contains income statement summary accounts. This example does not use or enable the default NavLink parameters. In this example, two Cube Views are set up, then both Cube Views are added to dashboard components.

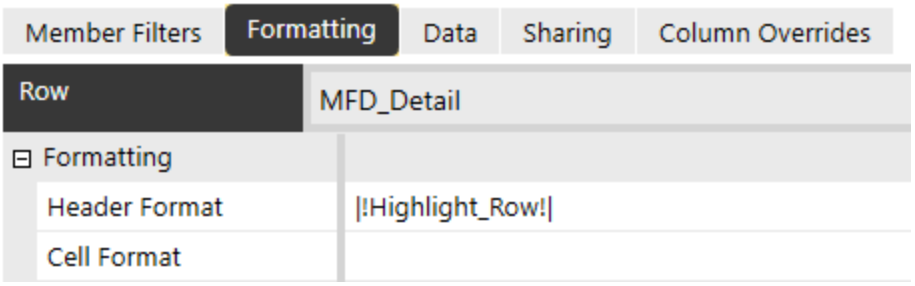
To set up navigation links for linked dashboards, complete the following steps:

1. Navigate to the **Rows and Columns** slider and select the row to be highlighted for navigation.
2. In the **Data** tab, set the **Enable Report Navigation Link** property to **True**.
3. In the **Dashboard To Open In Dialog** field, enter the name of the dashboard to open. The specified dashboard, not a Cube View, opens when this highlighted row is selected from this Cube View's related dashboard.



4. (Optional) You can add a header format using a Cube View style. This displays the row in a color different from the other rows to indicate that the row can be selected. In this example, the |!Highlight_Row!| parameter was created and used. See "Create a Parameter" in the *Understanding the Financial Model Guide*.

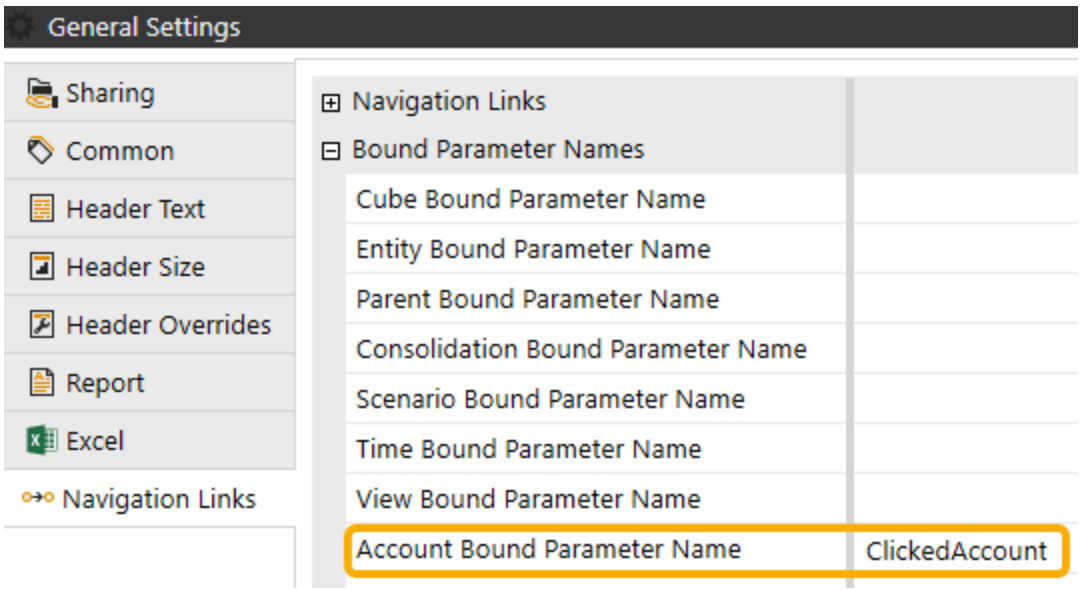
General (Parameter)	
Name	Highlight_Row
Workspace	Test
Maintenance Unit	GolfStream Navigation Link Example
Description	
User Prompt	
Sort Order	0
Data Source	
Parameter Type	Literal Value
Default Value	TextColor=Blue



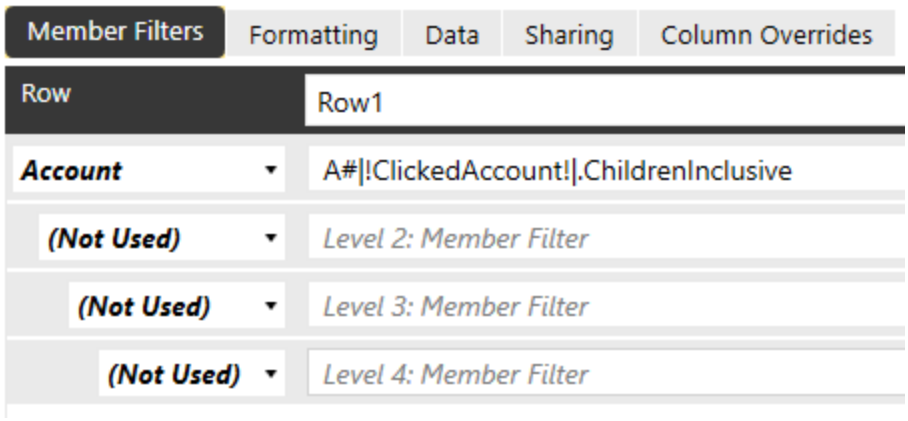
When this Cube View is run, the format of TextColor=Blue activates for the row.

5. When you select an account in the Cube View, that account value is sent to the other dashboard where the account and its children can be viewed. To determine how the account is passed, go to **General Settings** > **Navigation Links**. In this example, the value ClickedAccount is entered in the Account Bound Parameter Name, which will be passed from one dashboard to another.

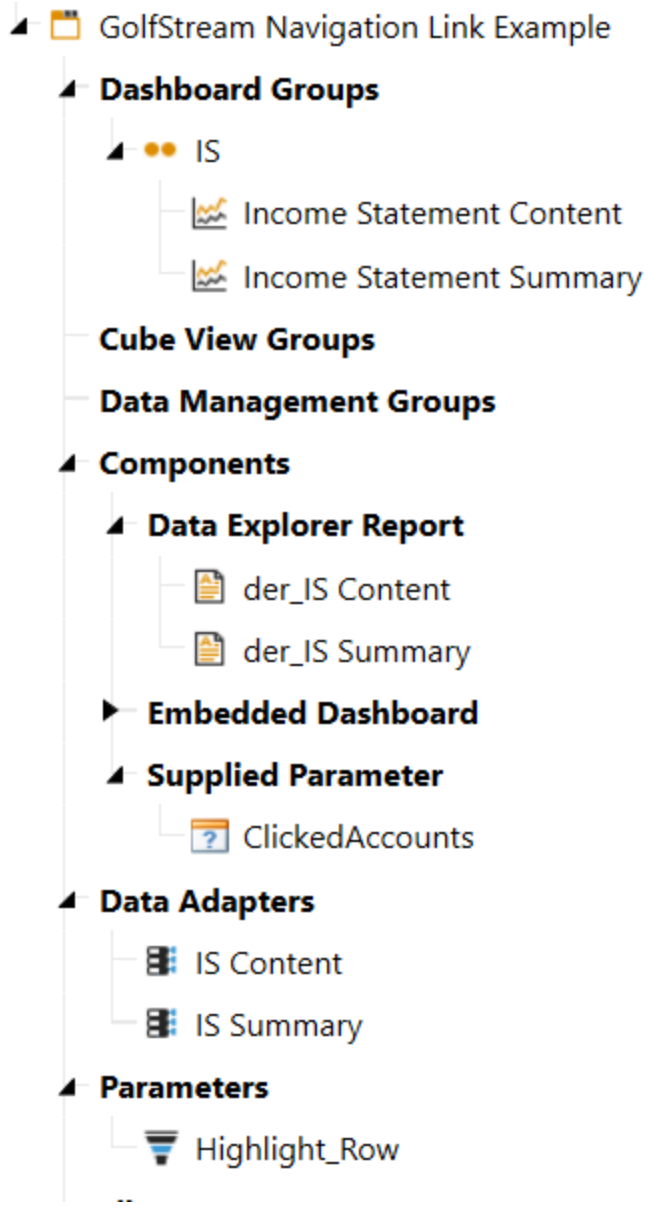
> **NOTE:** More than one dimension can be passed if it is defined in rows.



6. Create the linked Cube View. In this example, the Cube View is called IS Content because it contains the drilled data.
7. In the copied Cube View, the rows should be populated with the chosen account in the prior Cube View. Ensure that the same link is passed into the row POV for the account member. In this example, the parameter ClickedAccount is referred to and the ChildrenInclusive extension is added.



8. In **Application** > **Workspaces**, set up the following objects: Data Adapters, Data Explorer Report components, Parameter component, and a dashboard group.



a. Data adapters are required to point to the Cube Views. Create a data adapter and set the **Command Type** property to **Cube View**. Set the **Cube View** property to the correct Cube View. See Data Adapters.

In this example, a data adapter is set up for IS Content. The Cube View property is set to the IS Content Cube View.

⊟ General (Data Adapter)	
Name	IS Content
Workspace	Test
Maintenance Unit	GolfStream Navigation Link Example
Description	
⊟ Processing	
Template Parameter Values (e.g., Param1=Value1, ...)	
Text 1	
Text 2	
⊟ Data Source	
Command Type	Cube View
Cube View	IS Content

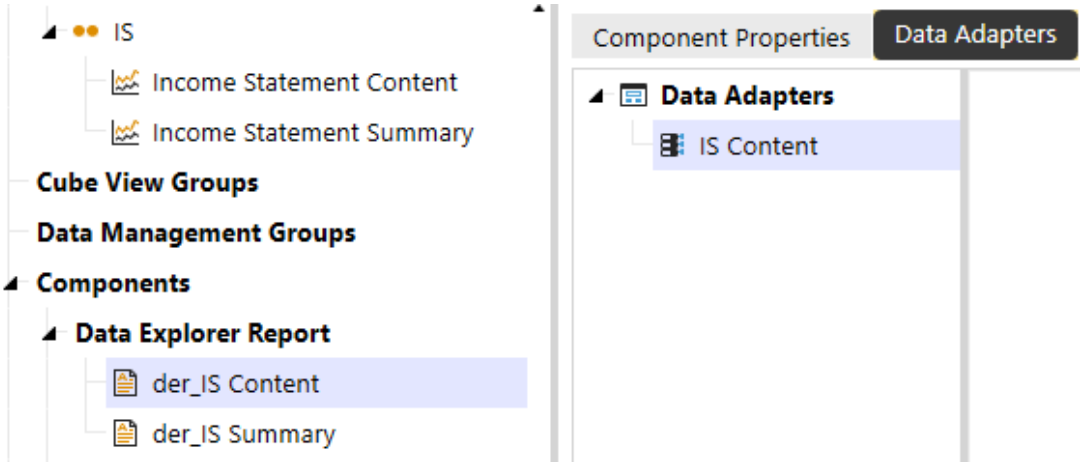
Create an additional data adapter and set the **Command Type** property to **Cube View**. Set the **Cube View** property to the correct Cube View and set the **Include Row Navigation Link** property to **True** to drill from Cube Views in a dashboard.

Include Member Details	False
Include Row Navigation Link	True
Include HasData Status	True

In this example, a data adapter is set up for IS Summary. The Cube View property is set to the IS Summary Cube View.

⊟ General (Data Adapter)	
Name	IS Summary
Workspace	Test
Maintenance Unit	GolfStream Navigation Link Example
Description	
⊟ Processing	
Template Parameter Values (e.g., Param1=Value1, ...)	
Text 1	
Text 2	
⊟ Data Source	
Command Type	Cube View
Cube View	IS Summary

b. Create Data Explorer Report components for a dashboard. See Dashboard Components . In the **Data Adapters** tab, add the corresponding data adapters to each Data Explorer Report component.

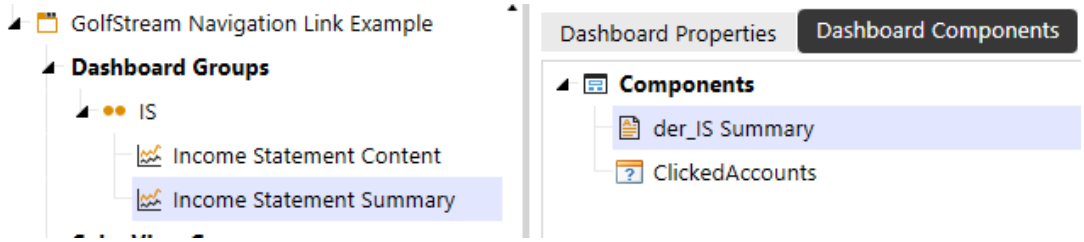


c. Create a Supplied Parameter component. See Dashboard Components and Supplied Parameter. In the **Bound Parameter** field, enter the name of the parameter being passed. In this example, it is ClickedAccount. Do this for each parameter being passed from dashboard to dashboard.

⊟ General (Component)	
Name	ClickedAccount
Workspace	Test
Maintenance Unit	GolfStream Navigation Link Example
Description	
Component Type	Supplied Parameter
⊟ Processing	
Template Parameter Values (e.g., Param1=Value1, ...)	
Text 1	
Text 2	
⊟ Action	
Bound Parameter	ClickedAccount

d. Create a dashboard group with two dashboards. Create one for the launching dashboard, which is IS Summary in this example. Create one for the launched dashboard, which is IS Content in this example. In this example, the Layout Type property is set to Uniform, but the use of this feature can vary. See Dashboard Properties.

e. In the **Dashboard Components** tab, attach the Data Explorer Report components to the appropriate dashboards. For the launching dashboard, attach the parameter component being passed. In this example, the ClickedAccounts Supplied Parameter component is added to IS Summary. This enables that account to be passed from the initial dashboard to the other dashboard.



Linked Cube Views

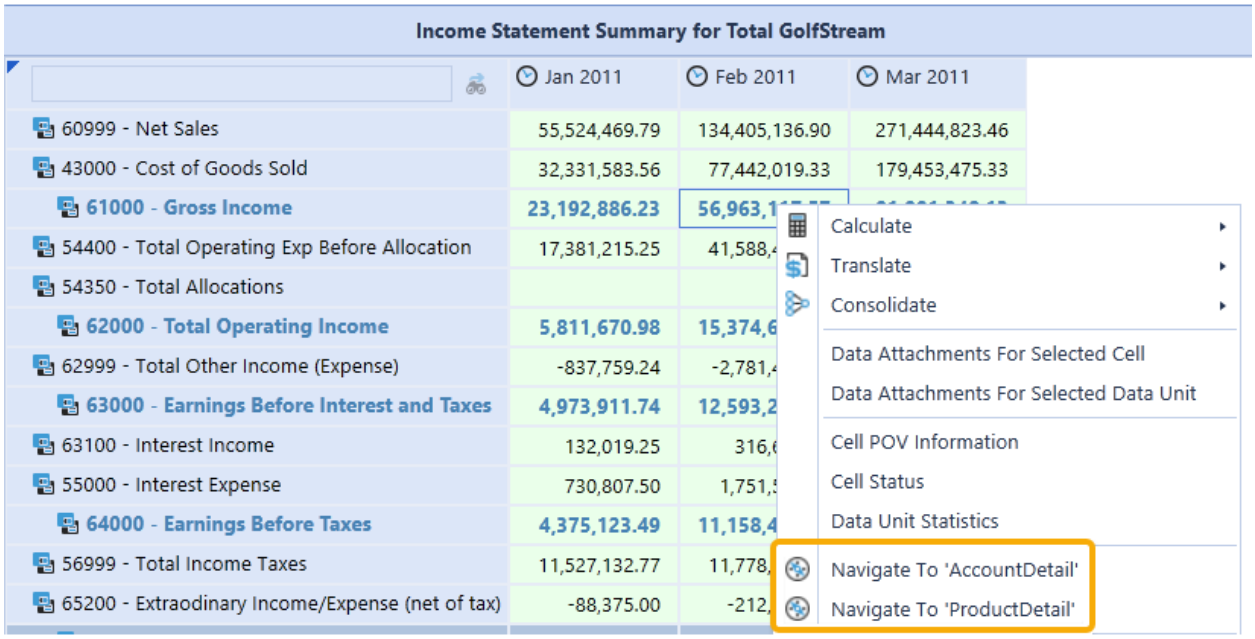
Linked Cube Views enable you to launch a Cube View from another when viewing it in data explorer, Spreadsheet, or Excel. You can right-click a cell to launch a separate Cube View and access additional detail and visibility. This applies to any Cube View, such as those for data entry forms.

> **TIP:** For information about evaluating source Cube View data in a linked Cube View using passed point of view selections, see Linking Cube Views Through the Cube POV.

> **NOTE:** For linked Cube Views, the Workspace name is not automatically prefixed. The following search order of operations is used to find the correct Cube View: Requested Workspace, current Workspace, default Workspace, and then shared Workspaces.

Create a Linked Cube View

The following example describes how to create a linked Cube View. In this example, an Income Statement Summary Cube View has an Account Detail Cube View assigned to all data cells, and a Sales by Product Cube View linked to specific rows. You can also add a second level of linked Cube Views for more granular analysis.



Link a Cube View to an Entire Cube View

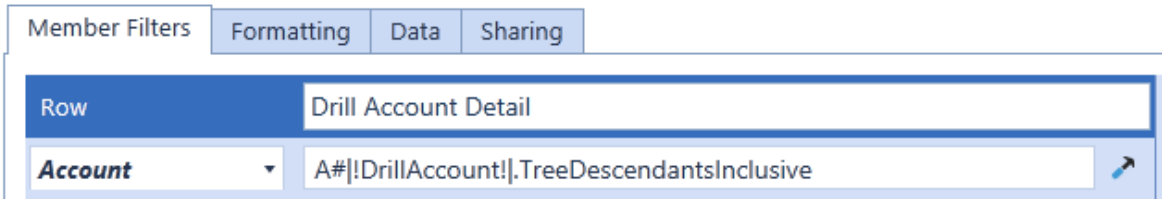
If the linked Cube View applies to the entire Cube View, you can right-click any data cell in data explorer, spreadsheet, or Excel to access the Cube View.

Complete the following steps to link a Cube View to all data cells:

1. Start with two created Cube Views: one main Cube View and one linked Cube View. In this example, the main Cube View is IncomeStatementSummary, and the linked Cube View is AccountDetail. The IncomeStatementSummary Cube View displays Parent accounts for a specific time. The AccountDetail Cube View displays the selected account's children for that same time period.
2. Configure the navigation links on the main Cube View. To assign the linked Cube View to the entire main Cube View, select the main Cube View and go to **General** > **Navigation Links**. In the **Linked Cube Views** field, enter the name of the Cube View or select the ellipsis to find the Cube View using the Object Lookup dialog box.

3. Enter the Bound parameters, which will pass from one Cube View to the next. In this example, a Bound parameter was entered for Account and Time so that both members are based on the selected IncomeStatementSummary data cell.
4. In the linked Cube View, configure the columns and rows to query the correct data.

   In this example, |!DrillAccount!| is a Bound parameter that displays the correct account data based on the selected account in the IncomeStatementSummary Cube View.

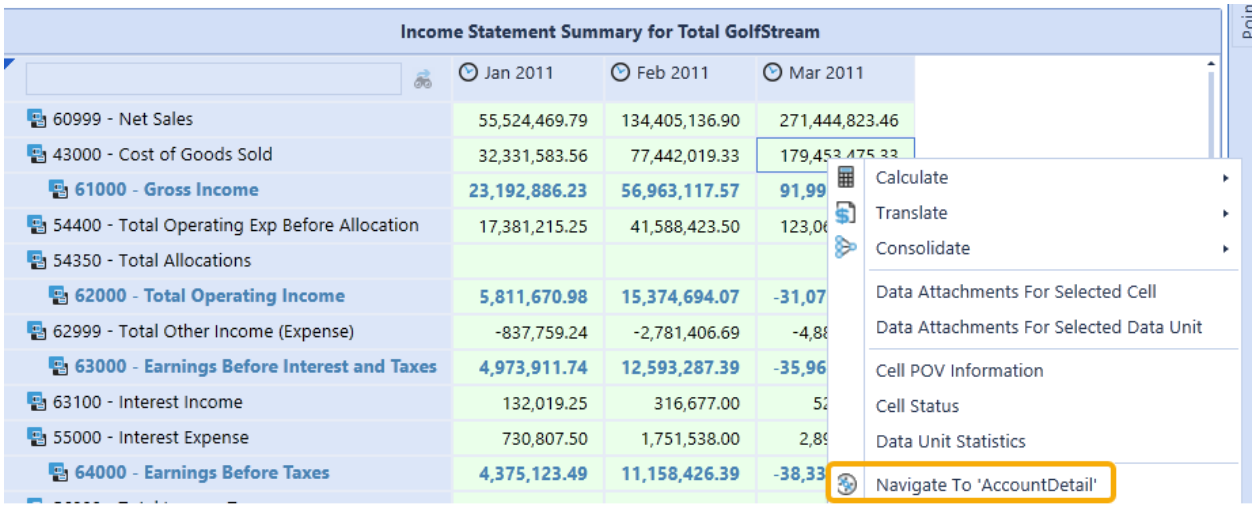


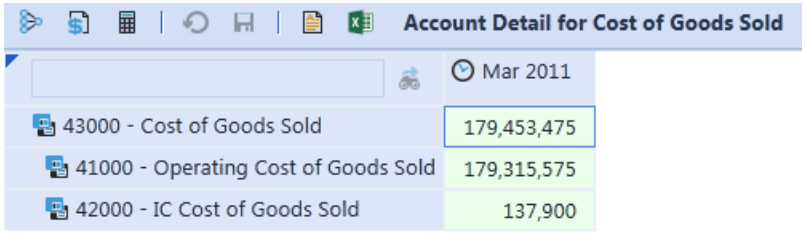
   |!DrillTime!| is a Bound parameter that displays the correct time data based on the selected time in the IncomeStatementSummary Cube View.



5. In this example, the AccountDetail POV matches the IncomeStatementSummary POV with the exception of Account and Time. The |!DrillAccount!| and |!DrillTime!| parameters are entered into these member fields to ensure the AccountDetail Cube View displays data based on the selected account and time in the IncomeStatementSummary Cube View.

When the IncomeStatementSummary Cube View is run, you can right-click any cell to access the AccountDetail Cube View.



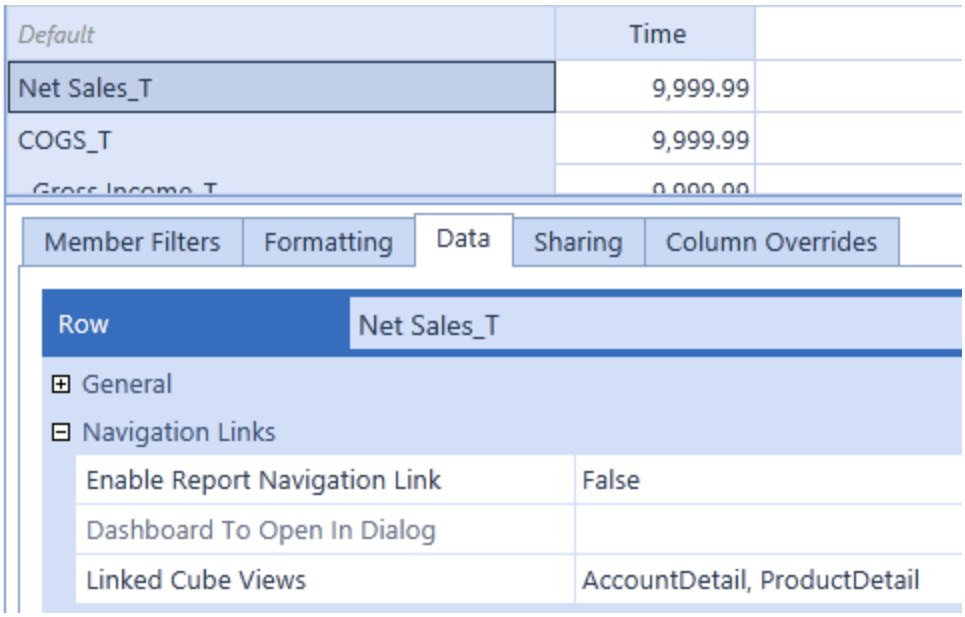


Link a Cube View to Specific Rows or Columns

If the linked Cube View only applies to columns or rows, it is only available when you right-click on a data cell in that row or column. In this example, the ProductDetail Cube View displays product details based on the IncomeStatementSummary Cube View's selected data cell. The IncomeStatementSymmary Cube View's data displays parent accounts for a specific time, so the ProductDetail Cube View's data displays the selected account's product sales.

Complete the following steps to link a Cube View to specific rows or columns:

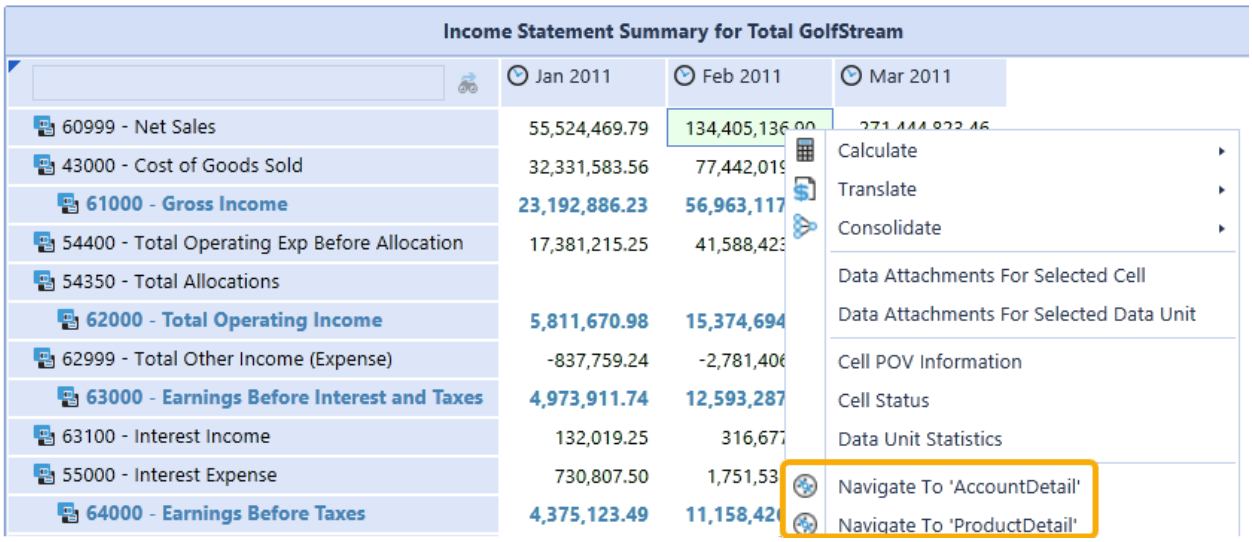
1. In the linked Cube View, configure the rows/columns, POV, and formatting to ensure the user sees the correct data.
2. To assign the linked Cube View to the base Cube View's rows, select the base Cube View and go to **Rows and Columns** > **Data** > **Linked Cube Views**. Select the desired row or column and enter the name of the linked Cube View.



> **NOTE:** In this example, the AccountDetail Cube VIew is assigned to this property as well due to overrides. In the previous example, AccountDetail was assigned to the entire Cube View. However, because a Cube View is being assigned to a row, the Cube View settings are overridden and only display the Cube View specified for this property. Both must be specified to make them available on the row.

3. If any Bound parameters need to be entered, go to **General Settings** > **Navigation Links** > **Bound Parameter Names**. In this example, the same Bound parameters are used for both Cube Views.

In this example, the ProductDetail Cube View is assigned to the Net Sales row. When a cell in that row is right-clicked, the Cube View is available.

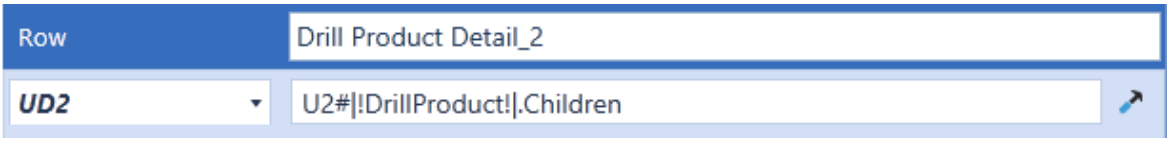


Nest Linked Cube Views

You can link a Cube Views to Cube Views that are already linked. In this example, the ProductDetail_2 Cube View is linked to ProductDetail and displays specific product data tied to a selected ProductDetail cell.

1. In ProductDetail_2, the rows and columns are configured as needed.

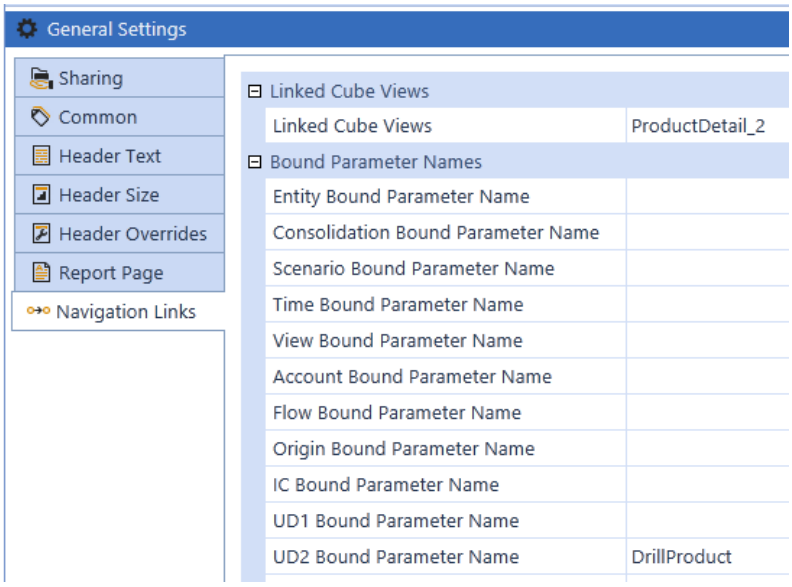
   A |!DrillProduct!| Bound parameter is entered to display specific product details. For example, if you right-click a Clubs cell in the ProductDetail Cube View and select the ProductDetail_2, sales details for Clubs displays.

   A |!DrillTime!| parameter must also be used.



2. The POV for ProductDetail_2 is set so it matches the ProductDetail POV, except for the User Defined 2 member. In this example, products are viewed using the UD2 member. |!DrillProduct!| is entered in User Defined 2.
3. In the ProductDetail Cube View, in General Settings > Navigation Links > Linked Cube Views, ProdictDetail_2 is entered. This links the Cube View to the entire ProductDetail Cube View.
4. The Bound parameter from ProductDetail_2 is entered in the UD2 Bound Parameter Name field.



When you launch the ProductDetail Cube View from the IncomeStatementSummary Cube View, you can right-click any cell in the ProductDetail Cube View and open the ProductDetail_2 Cube View.

Linking Cube Views Through the Cube POV

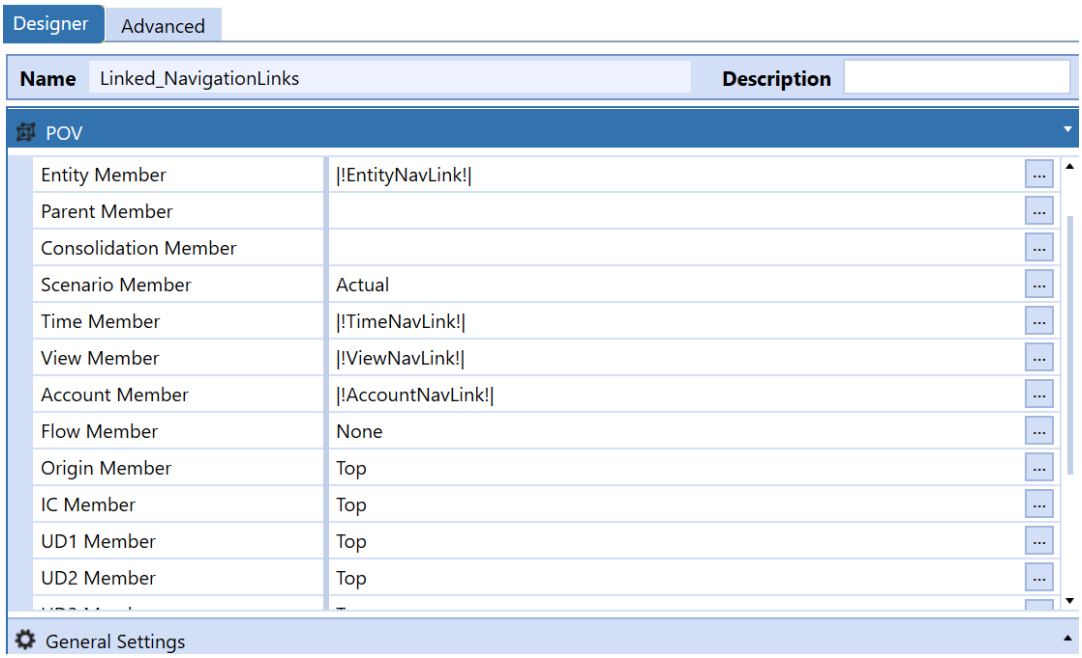
You can create NavLink parameters to be used in the rows, columns, or Cube View POV of the linked Cube View. You can use the Cube View POV to pass in multiple dimensions from the source Cube View. This ensures users are looking at the same dimensions between Cube Views. Point of view selections specified in the source Cube View can be passed to the linked Cube View, enabling a refined, focused analysis.

You can use the standard syntax for the parameters or you can create your own. To use the standard syntax, the Include Default NavLink Parameters property must be set to True. Use the following syntax: |!<dimension>NavLink!|

> **Example:** To pass the Time dimension from a source Cube View to a linked Cube View, specify |!TimeNavLink!| in the linked Cube View POV. The data in the cell point of view in the source Cube View passes to the linked Cube View point of view at runtime.

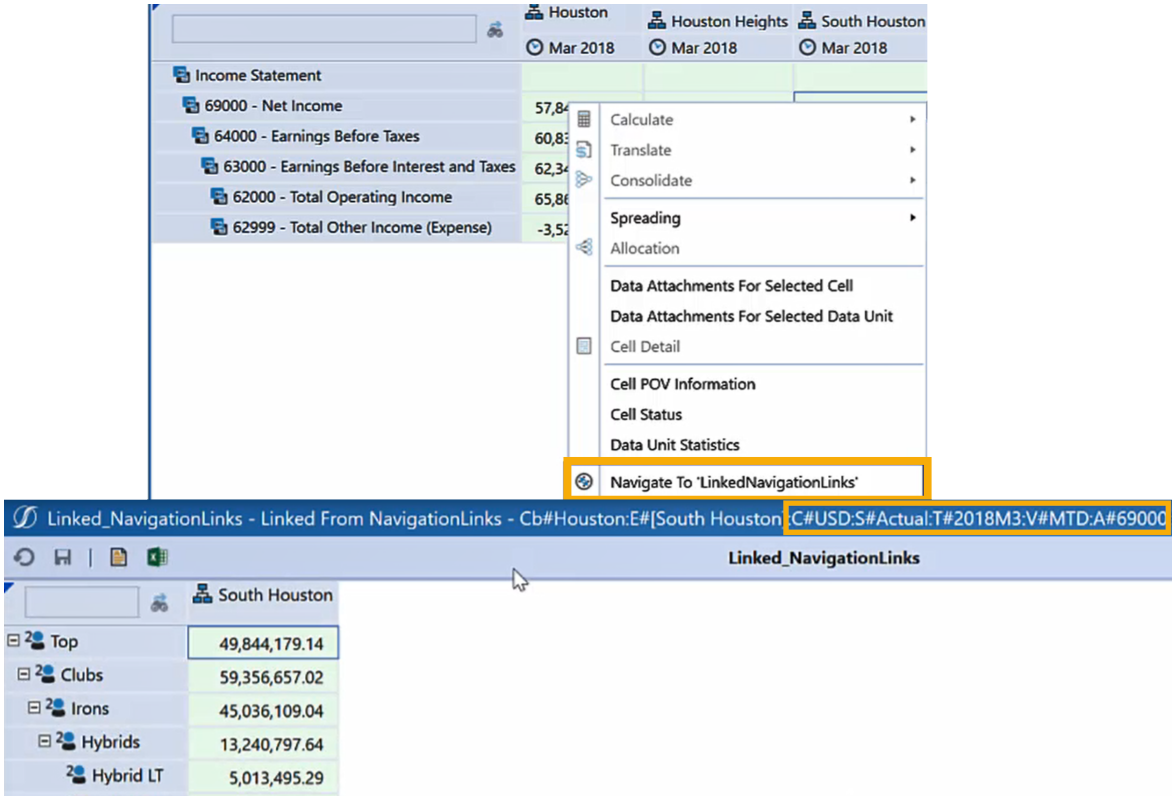
To use passed point of view selections in a linked Cube View, complete the following steps:

1. Go to **Application** > **Cube Views** and select the base Cube View containing the source data you want to access and analyze further in a linked Cube View.
2. Go to **General Settings** > **Navigation Links** and set the **Include Default NavLink Parameters** property to **True**. Select the **Save** icon.
3. Select the **POV** slider and specify |!dimensionNavLink!| for each dimension to analyze. Select the **Save** icon.



4. Return to the source Cube View and select the **Open Data Explorer** icon.

5. Right-click cells to navigate to the linked Cube View, retrieving data for the specified point of view parameters. These are also identified in the header.



Format a Report Page

The following settings in the Cube View address page formatting:

- Header (text, size, overrides)
- Report
- Headers and footers

Ensure the general settings are correct before formatting your page. See General Settings.

To edit the header and report settings, on the **Cube Views** page, under **Cube View Groups**, select a Cube View and expand the **General Settings** slider.

Header Settings

Row and column headers can be adjusted to refine the Cube View.

- **Header Text**: Decide if you want to see the name, description, or both for each of the dimension types. Set the Culture property for the Cube View to determine the language of the report. Ensure the Member Descriptions Culture properties are set up correctly before setting the Culture property.
- **Header Size**: Provide a height for columns and width for rows. The rows can be edited separately by Data Grid (Row Header), Excel (Excel Row Header), and PDF (Report Row Header). Each of these properties default to -1. You can change them by entering a pixel number.
- **Header Overrides**: Change which dimensions display in the rows and columns. The default for a Cube View is to display what is available in the rows and columns. However, you can apply Header Overrides to change the order or add dimensions.

  The Report Column Index for Row Headers property is also set here. This changes the location of the row headers from their usual place on the left. Type the number of columns that should display before the row headers are shown. This only applies to the PDF version of the report.

Report Settings

These properties apply advanced formatting to Cube Views.

- **Custom Report Formatting**: Write or assign Cube View extender rules to provide intricate Cube View formatting. This only applies to the PDF version.

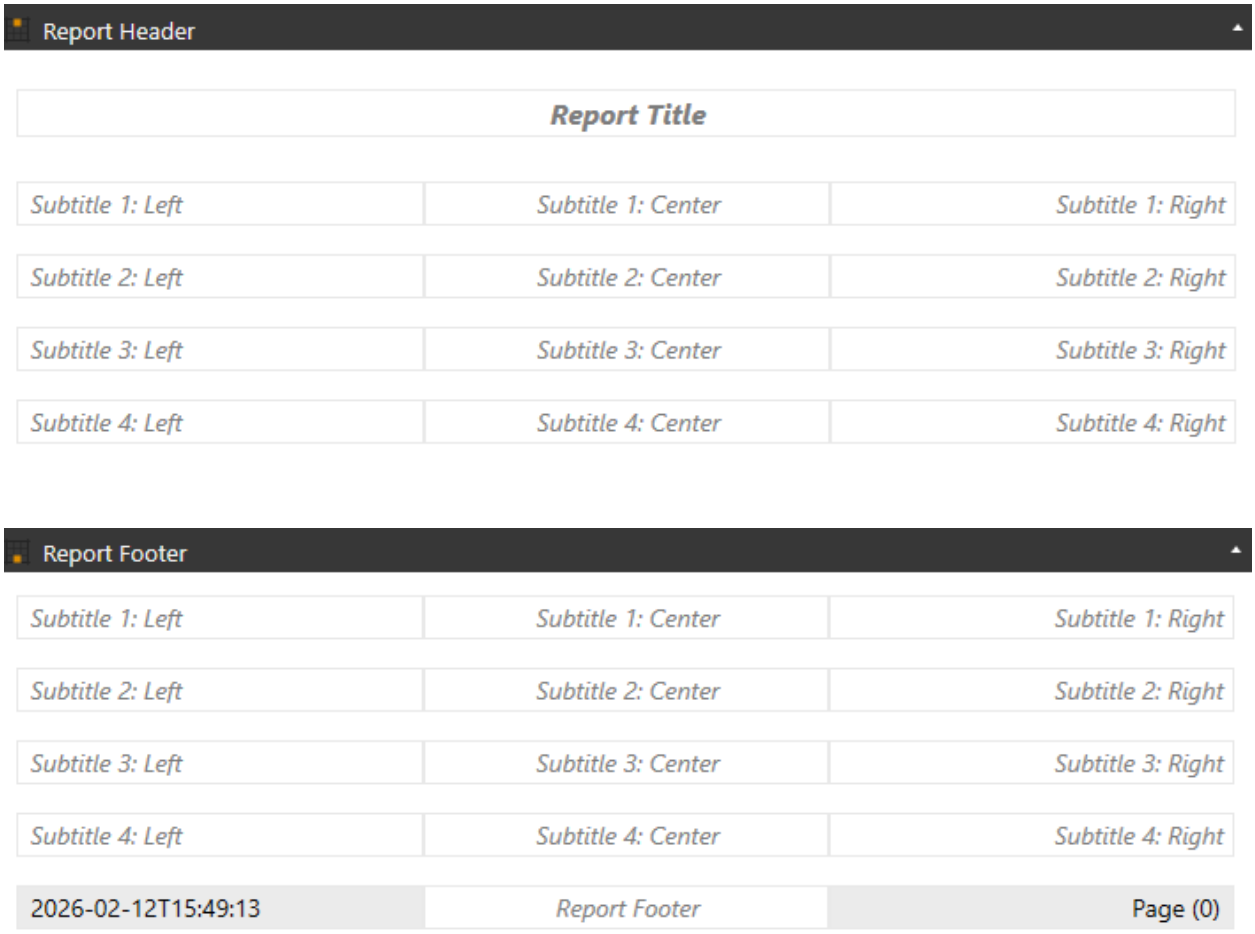
- **Paper**: Select the paper size.
- **Landscape**: Select the page orientation.
- **Margin**: Set margins on individual reports. Enter a pixel number to override the default settings of the page in the General Settings.
- **Auto Fit Settings**:
    - Set the Auto Fit to Page Width to True if the report should re-size to fit all columns on one page.
    - Use the Auto Fit Number of Pages Wide property to enter the number of pages to display the columns.
- **Scale Factor**:This property sets and adjusts the scale of the report by this value and automatically centers report content horizontally. The default value is 1, and the maximum value is 2. If you enter a value greater than 2, it will default to a scale factor of 2.

> **NOTE:** Scale Factor is not compatible with Auto Fit to Page Width.

Header and Footer Settings

Report headers and footers control what displays at the top and bottom of the page when a Cube View is run as a data explorer report. Standardize headers and footers across reports and make them as dynamic as possible to reduce reporting maintenance.

In the Designer tab, Report Header and Report Footer are separate sliders. Enter text in the fields or use substitution variables.



Use substitution variables to reduce header and footer maintenance. Substitution variables are short scripts that use pipe characters to include a predefined substitution variable. They come with every installation of OneStream and cannot be edited, so you do not need to create or maintain them. Substitution variables can be used throughout the application. For Cube Views, you can use them for headers and footers, rows and columns, and Cube View page captions.

To copy and paste a substitution variable into your application, complete the following steps:

1. On the **Cube Views** page, select the **Object Lookup** icon to open the Object Lookup dialog box.
2. Under **Object Type**, select **Substitution Variables** to see the list of options.
3. Select the name of the substitution variable.
4. Select the **Copy to Clipboard** button.
5. Paste the substitution variable where needed.

Calculations

When you create calculations in a Cube View, data is generated on-the-fly and is not stored in the database. The calculation returns a value when the Cube View is run.

You can use calculations in a Cube View to calculate:

- Variances
- Sums
- Differences
- Ratios

Calculations are useful for variance reporting, What If scenarios in Cube Views, and KPI calculations.

Calculation Types

There are three ways to create calculations in Cube Views. Each option uses the GetDataCell function, which retrieves specific cells and performs math or business rule operations.

The most common ways to create calculations in Cube View use:

- GetDataCell Expressions
- Column and Row Expressions
- Dynamic Member Formulas and Cube Views

Choose the option with the least maintenance and highest usability.

> **NOTE:** You can also create more complex calculations with business rules.

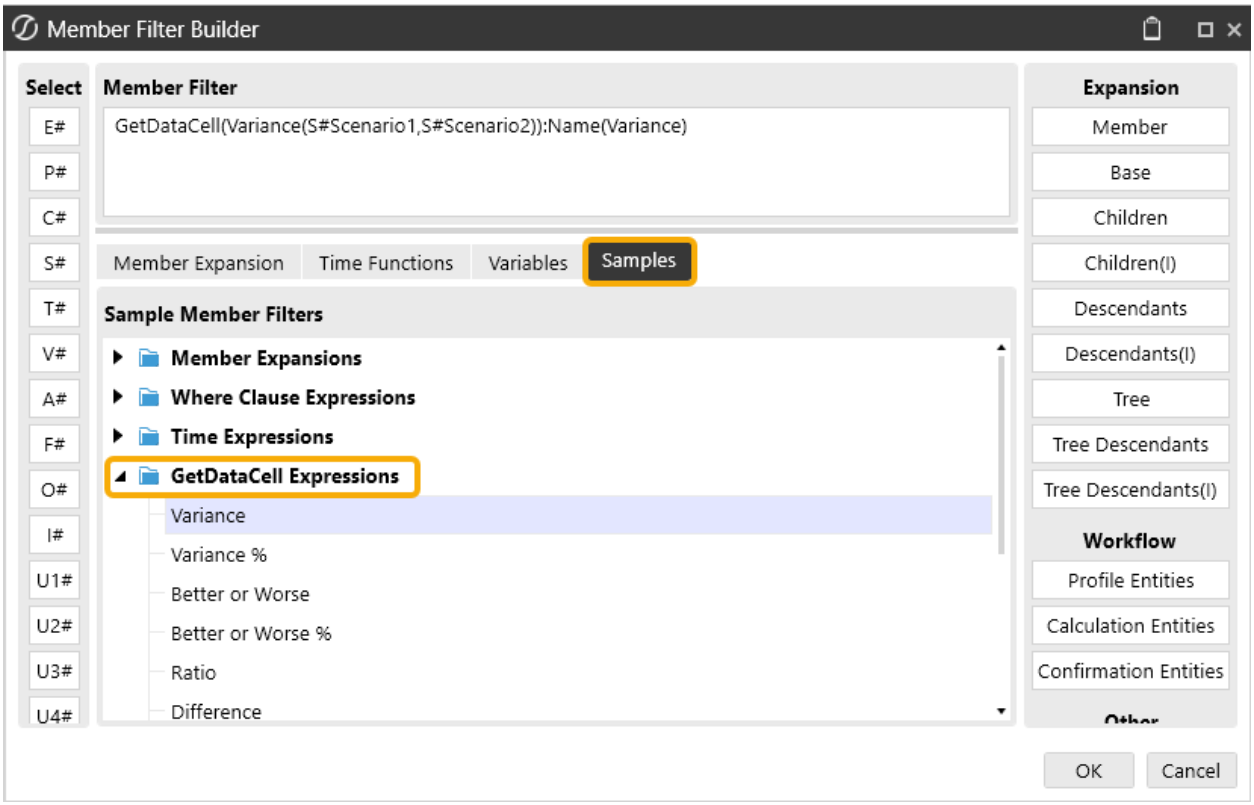
GetDataCell Expressions

GetDataCell expressions are written in the Cube View row or column and are calculated at run-time. The expressions require a reference to specific dimension members to return a data point. Use GetDataCell expressions if you are running calculations on members that are not currently in the Cube View or if the expression is easy to create and maintain. GetDataCell expressions are simpler versions of dynamic calculations.

> **NOTE:** Use a different calculation type if you are querying multiple member combinations.

GetDataCell Types

In the Member Filter Builder, you can use the Samples tab to create simple GetDataCell expressions in your Cube View.



Review the following GetDataCell expressions:

Function Name	Description	Example
Variance	Calculates the difference as a ratio between two scenarios using the Variance function.	GetDataCell(Variance(S#Scenario1, S#Scenario2)):Name(Variance)

Function Name	Description	Example
Variance%	Calculates the difference as a percentage between two scenarios using the VariancePercent function.	GetDataCell(VariancePercent (S#Scenario1, S#Scenario2)):Name(Var%)
Better or Worse	Calculates the difference between two scenarios while considering account types using the BWDiff function.	GetDataCell(BWDiff(S#Scenario1, S#Scenario2)):Name(BW Diff)
Better or Worse%	Calculates the difference as a percentage between two scenarios while considering account types using the BWPercent function.	GetDataCell(BWPercent(S#Scenario1, S#Scenario2)):Name(BW%)

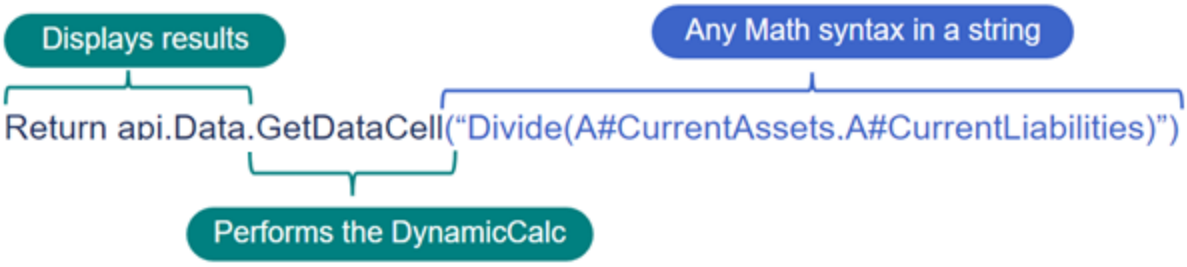
Function Name	Description	Example
Ratio	Calculates the ratio between two scenarios using the Divide function.	GetDataCell(Divide(S#Scenario1, S#Scenario2)):Name(Ratio)
Difference	Calculates the difference between scenarios using the Subtraction function.	GetDataCell(S#Scenario1-S#Scenario2): Name(Difference)
Sum	Calculates the sum of two scenarios using the Addition function.	GetDataCell(S#Scenario1+ S#Scenario2: Name(Total)
Custom Function (Condensed Syntax)	Calls a Finance business rule using the BR# function.	GetDataCell(BR#[MyBusinessRuleName, MyFunctionName]): Name(Custom Function)
Custom Function	Calls a Finance business rule using the BR# function.	GetDataCell(BR# [BRName=MyBusinessRuleName, FunctionName=MyFunctionName]):Name (Custom Function)

Function Name	Description	Example
Custom Function with Parameters	Calls a Finance business rule using the BR# function.	GetDataCell(BR# [BRName=MyBusinessRuleName, FunctionName=MyFunctionName, Name1=Value1, AnotherName=[Another Value]]):Name(Custom Function)

> **NOTE:** You can also find a complete list of GetDataCell expressions under **OnePlace** > **Dashboards** > **Application Reports** > **Application Reports Dashboards** > **Report Groups** > **Application Analysis**.

Components of GetDataCell

A GetDataCell expression is the performance section of a dynamic calculation. It tells the dynamic calculation what it will be doing. The following example shows a common GetDataCell expression. GetDataCell expressions are simpler versions of dynamic calculations.



Return api.Data.GetDataCell("Divide(A#CurrentAssets,A#CurrentLiabilities)")

- **Return api.Data**: This tells OneStream which results to display.

  > **NOTE:** This is not required for GetDataCell expressions, but it is required for dynamic calculations.

- **GetDataCell**: This performs the calculation.
- **("Divide(A#CurrentAssets,A#CurrentLiabilities)**: Math syntax in a string format to calculate.

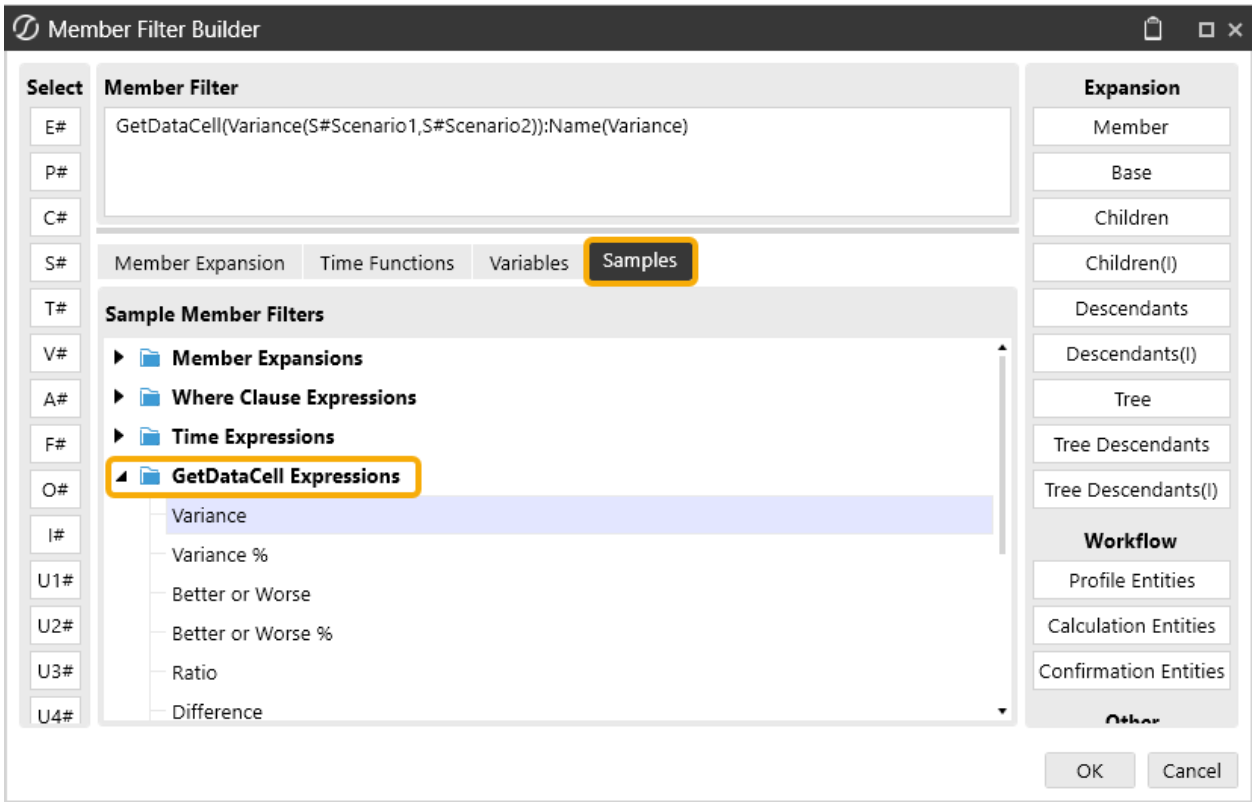
Create a GetDataCell Calculation

GetDataCell expressions run a calculation directly in a Cube View by referencing specific dimension members.

Some common errors when developing GetDataCell expressions are:

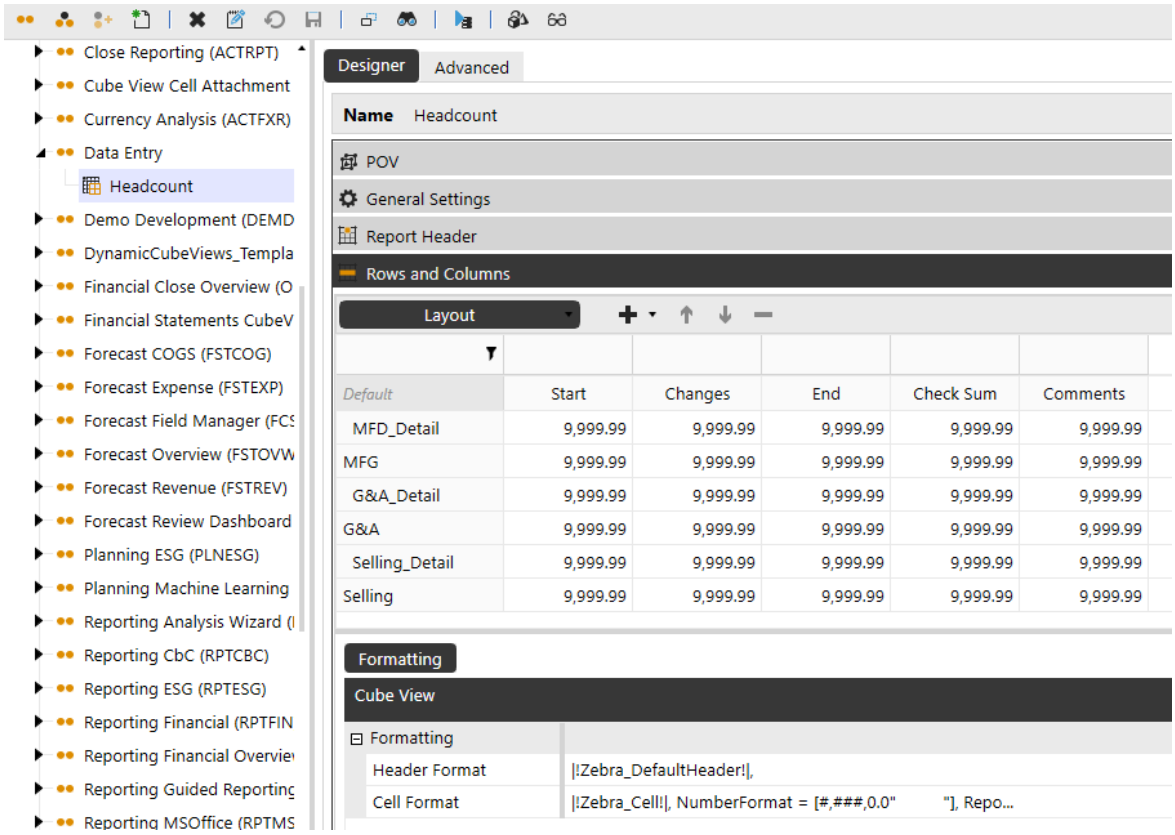
- Not having the correct dimension listed. Use the Member Filter Builder if there is any uncertainty about what to use.
- Not having the correct number of parentheses. Make sure to use the correct number of opening and closing parentheses.

In the Member Filter Builder, use the Samples tab to create simple GetDataCell expressions.



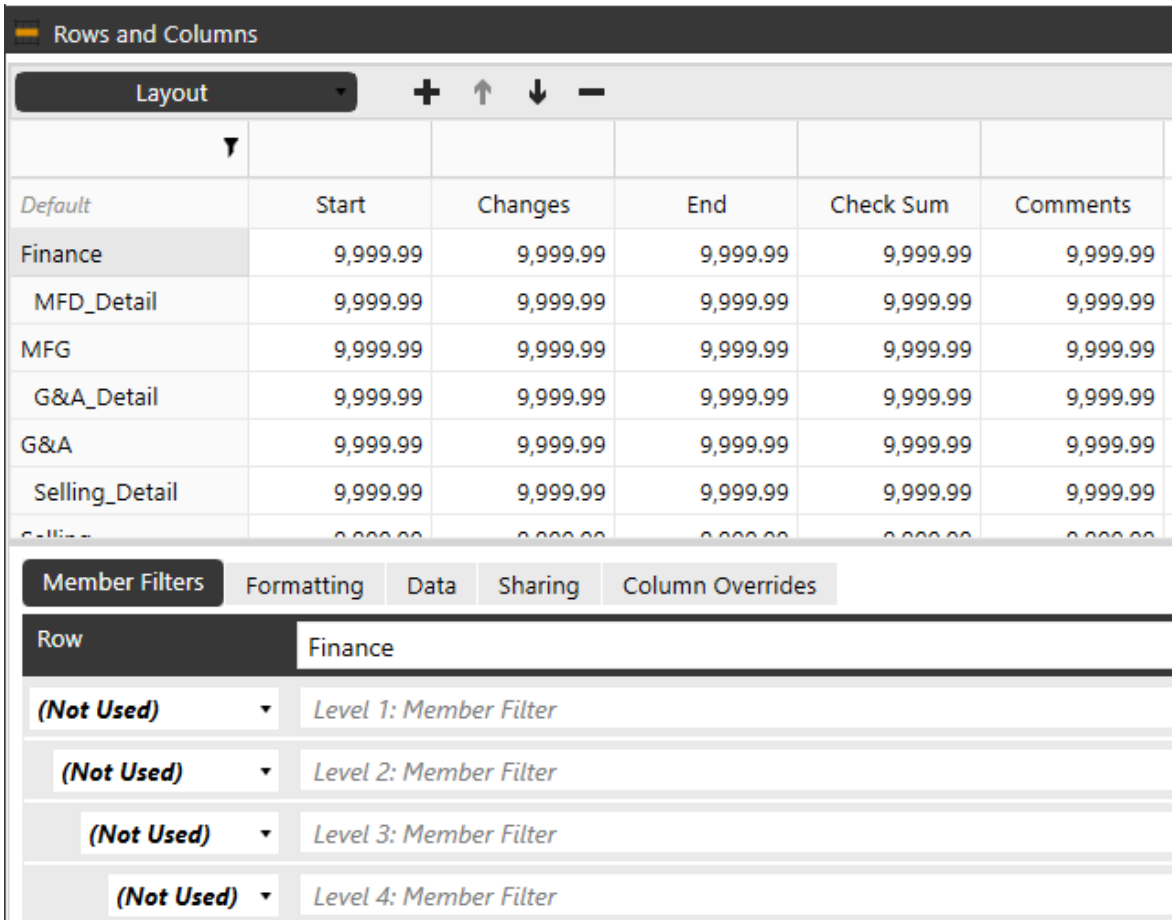
To add a row expression to a column in a Cube View, complete the following steps:

1. Go to **Application** > **Presentation** > **Cube Views**.
2. Select a Cube View Group and then select a Cube View. Expand the **Rows and Columns** slider.



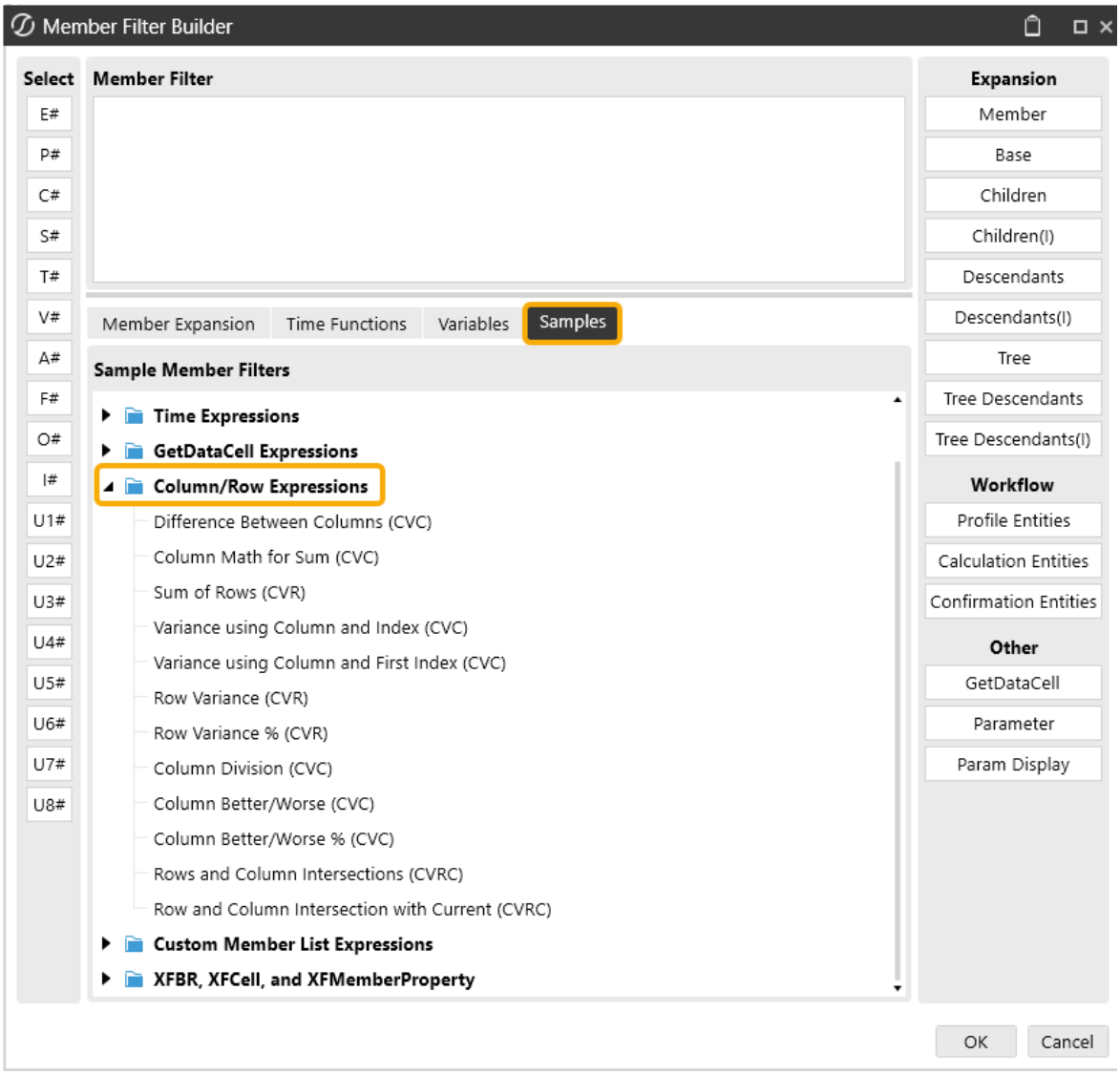
3. To add a row and insert the CVC/CVR calculation, select the **Add Row or Column** icon.

4. In **Member Filters** tab, rename the row.

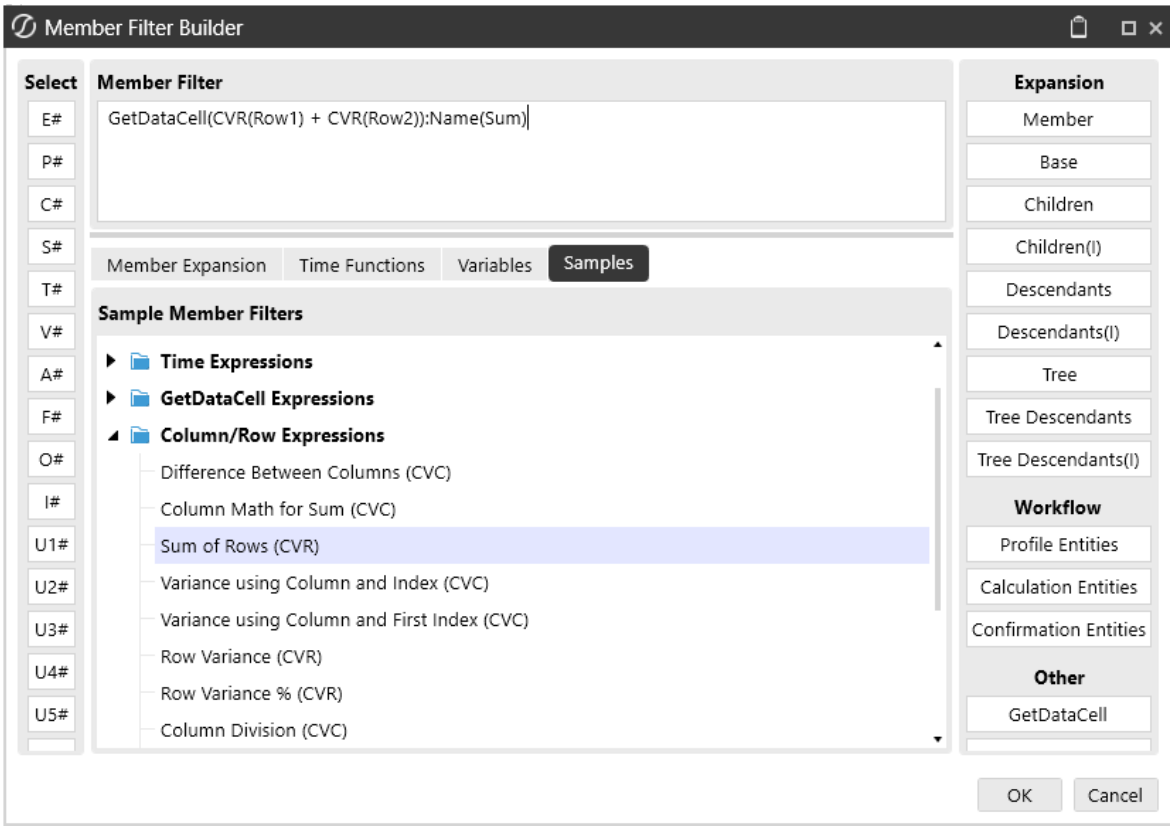


5. To insert a calculation. select the **Member Filter Builder** icon for the row.

6. Select the **Samples** tab and then expand the **Column/Row Expressions**.



7. Double-click **Sum of Rows (CVR)**. The GetDataCell syntax is generated and displays in the Member Filter field.



8. Update the row names to add the rows together. For example, rename Row1 to Cash and Row2 to AR.
9. Replace the name at the end of the expression to reflect the row name. The expression is complete.
10. Select the **OK** button and then select the **Save** icon.
11. Select the **Open Data Explorer** icon to run the Cube View and verify result.

Column and Row Expressions

Cube Views accept calculated columns and rows using the GetDataCell function and references to Cube View column and row names. This is known as column/row math. Column and row expressions are written in the Cube View row or column and are calculated at run-time. These expressions require a reference to the specific Cube View column or row name to return a data point.

Use column and row expressions to perform math across columns and rows. A common use case is with variance columns. Column and row expressions are useful if the query is more complex or if the expression is easy to understand and maintain.

> **NOTE:** Use one of the other expression types if you are using multiple member expansions, especially when performing math on rows.

Create Cube View Column and Row Calculations

Cube Views accept calculated columns or rows using the GetDataCell() function and references to Cube View column or row names. These are based on the names given to a Cube View column, such as Col1 or TimePeriods, or a Cube View row, such as Row2 or Accounts.

> **NOTE:** Using the column or row names may be more useful than using Member Filters.

Cube View columns use the GetDataCell function to perform math on columns, and Cube View rows use the GetDataCell function to perform math on rows. There are several variations on this method, depending on whether the expression refers to columns, rows, or a combination.

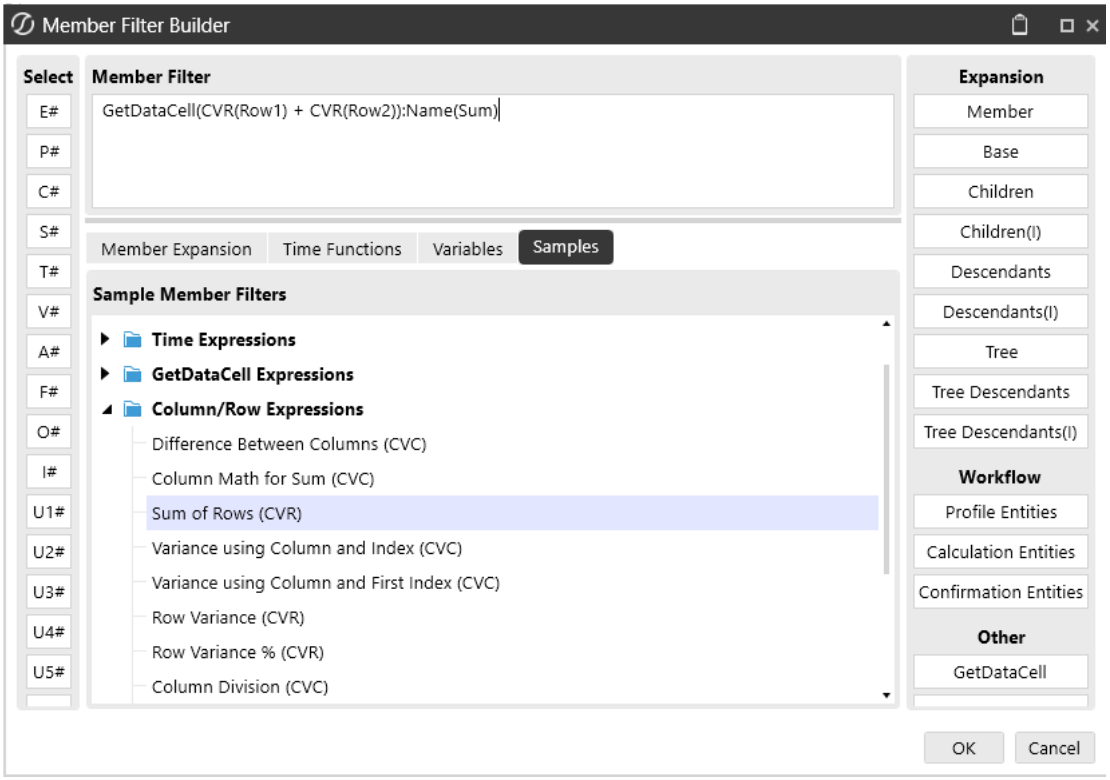
When naming a new calculation, do not include spaces or special characters that could make the column/row calculations read incorrectly.

> **NOTE:** To create advanced column row expressions, use an index or a CVRC expression to create more intricate formulas. The Member Filter Builder has complex options in the Samples tab.

Create a CVC/CVR Calculation

Complete the following steps to create a CVC/CVR calculation:

1. Navigate to the row or column and select the **Member Filter builder**  icon.
2. Select the **Samples** tab and then expand the **Column/Row Expressions**.
3. Double-click **Sum of Rows (CVR)**. The GetDataCell syntax is generated and displays in the Member Filter field.



4. Update the row names to add the rows together. For example, rename Row1 to Cash and Row2 to AR where Cash and AR are row header names.



5. Replace the name at the end of the expression, for example, Current Assets.



6. Select the **OK** button and then select the **Save** icon.

7. Select the **Open Data Explorer** icon to run the Cube View and verify result.

Cube View Column Calculations

The following syntax is a column math example of the difference between columns (CVC):
GetDataCell(CVC(SomeColumnName) - CVC(SomeOtherColumnName)):Name(Header Name)

The example of simple member math includes the :Name() function typically applied to a column in a Cube View.

> **Example:** GetDataCell(CVC(Col1) - CVC(Col2)):Name (Variance)

> **Example:** GetDataCell(CVC(Col1) + 1):Name(Column Plus One)

> **Example:** GetDataCell(CVC(Col1) * (-1)): Name(Column with Sign Flipped)

Cube View Row Calculations

The following syntax is a row math example of Sum of Rows (CVR): GetDataCell(CVR (SomeRowName) + CVR(SomeOtherRowName)):Name(HeaderName). Instead of CVC, a calculated row uses CVR for retrieving the value of a row in a formula.

> **NOTE:** If a column name is numeric, single quotes are required when specifying the row name. Square brackets are allowed, but not required. For example, GetDataCell(CVR ('123') - CVR(['4,567'])):Name(Difference)

When you need a column and row index: Col1 has a member filter of S#Actual, S#Budget, meaning it will return two columns. In this case, a variance between Actual and Budget scenarios can be shown with the following syntax: GetDataCell(CVC(Col1, 1) - CVC(Col1, 2)):Name (Variance).

Alternative syntaxes are GetDataCell(CVC(Col1, First) - CVC(Col1, 2)):Name(Variance) and GetDataCell(CVC(Col1) - CVC(Col1, 2)):Name(Variance)

Column Math Example with Division and Other Advanced Functions

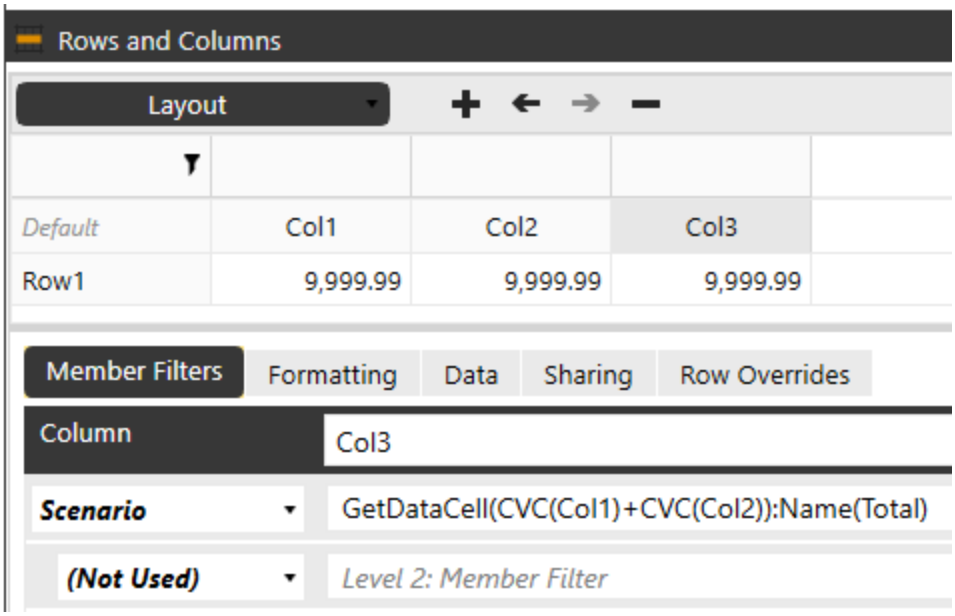
Member Expansions may require some more intricate formulas. This may require using indexes or a CVRC expression. Use functions like Divide to avoid divide by zero situations.

> **Example:** GetDataCell(Variance(CVR(Col1,2), CVR (Col1,1)):Name(Variance)GetDataCell(VariancePercent(CVR (Col1,2), CVR(Col1,1)):Name(Variance %) GetDataCell (Divide(CVC(Col3), CVC(Col2))):Name(Ratio) GetDataCell (BWDiff(CVC(Col1), CVC(Col2))):Name(BetterWorse Difference)GetDataCell(BWPercent(CVC(Col1), CVC (Col2))):Name(BetterWorse %)

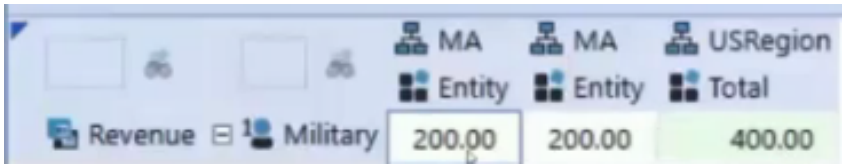
Reference Hidden Columns with Cube View Column Math

The CVMathOnly enables you to perform math on columns when the columns are not visible.

1. Navigate to the Cube View and expand the **Rows and Columns** slider. In this example, there are three columns, where Column 3 has the GetDataCell() function applied for the total of Column 1 and Column 2.



2. Select the **Open Data Explorer** icon to generate the Cube View and see the data. In this example, it is adding the value 200.00 from Column 1 to the value 200.00 from Column 2 to generate the total of 400.00 in Column 3.



3. Select the **Edit Cube View** icon.
4. Select the row or column and select the **Formatting** tab.
5. **In Header Format**, select the ellipsis. The Header Format window displays.
6. Select the **Format** button.
7. In the General section, go to the IsColumnVisible property and select **CVMathOnly**.

Dynamic Member Formulas and Cube Views

Dynamic calculations enhance the consolidation performance because the amount is calculated when requested for display and is not written to the database. Use dynamic calculations when you need a repository of commonly used calculations to ensure consistency and reduce maintenance.

Using dynamic calculations in Cube Views and business rules has the following benefits:

- There is no performance impact on consolidation times.
- You can make dynamic calculations drillable so users can see the source for the calculated result when drilling down.
- You can use these calculations for ratios or variances.

In the dimension library, you can write dynamic calculations on specific members. They will calculate at run-time.

Dynamic calculations have a syntax similar to GetDataCell expressions because they also require reference to specific dimension members to return a data point.

> **NOTE:** A difference between dynamic calculations and GetDataCell expressions is that dynamic calculations require proper business rule logic.

Create common calculations in the UD8 dimension. This enables you to have the most flexibility with your rule syntax.

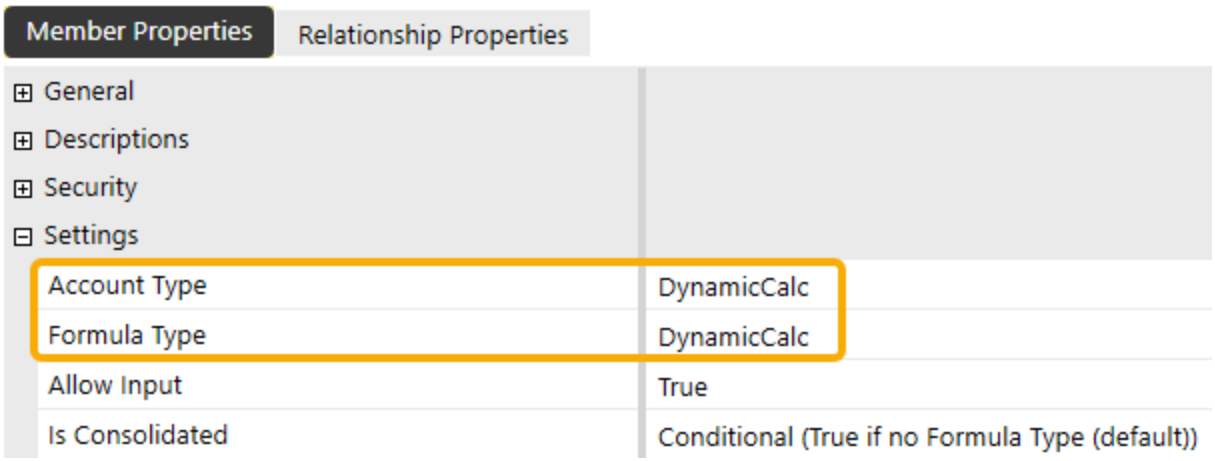
Use another calculation type when the calculation is specific to a single Cube View, because you may not want to create a member in the dimension library.

Consider granting a report builder access to the dimension library to view, create, and edit these calculations.

Create Dynamic Calculations

Account, Flow, and all UDs can contain dynamically calculated members.

The Account Type and Formula Type properties must be set to make the calculation dynamic. Ensure that the Formula Type property is set to DynamicCalc. If using on an Account dimension type, both the Account Type and Formula Type properties must be set to DynamicCalc.



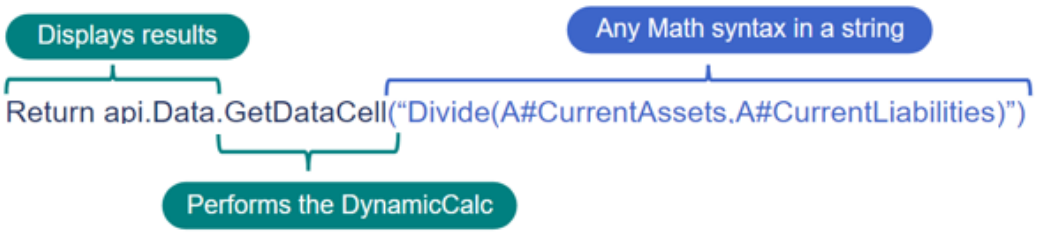
Here are some common examples of how dynamic calculations are used:

- Api.Data.Calculate("Target = A + B")
- Api.data.calculate("A#InsuranceExp = A#Payroll * A#InsuranceCostPcent")
- Api.Data.GetDataCell("A + B")
- Return Api.data.GetDataCell("A#GrossSales – A#ReturnsAndAllowances")

Calculated Members and Example

In the Dimension Library, there are likely a few members that are calculated using a member formula. Dynamically calculated members are members that use the GetDataCell function to return values based on math when they are queried.

> **NOTE:** For these members, the Formula Type property must be set to DynamicCalc before writing a member formula. If this property is not defined, the member will return an error and the rule will not compile.



Return api.Data.GetDataCell("Divide(A#CurrentAssets,A#CurrentLiabilities)")

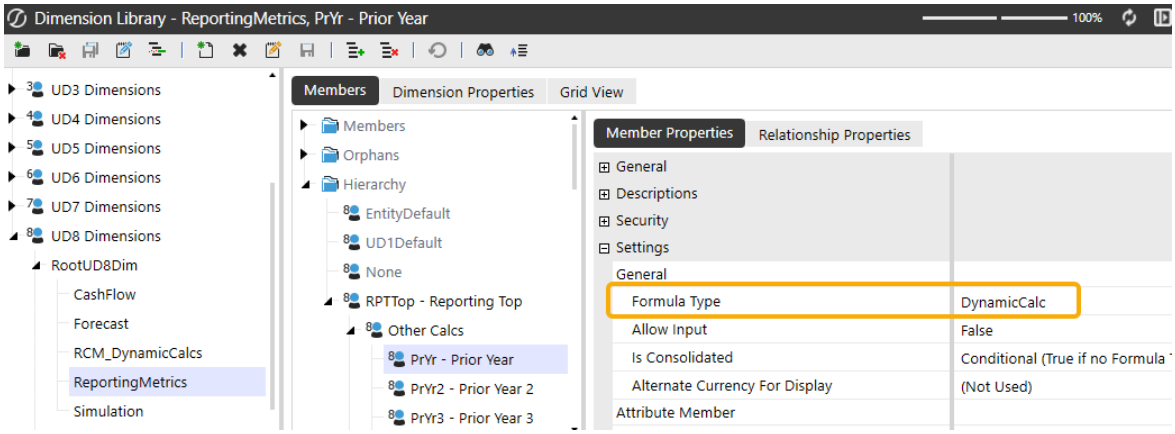
- **Display Results**: This section tells the system how to display the results of the function.
- **Performs the DynamicCalc**: This section tells the system to perform the DynamicCalc using the GetDataCell function.
- **Any Math syntax in a string**: This tells the system what math to perform.

Define a Dynamic Calculation

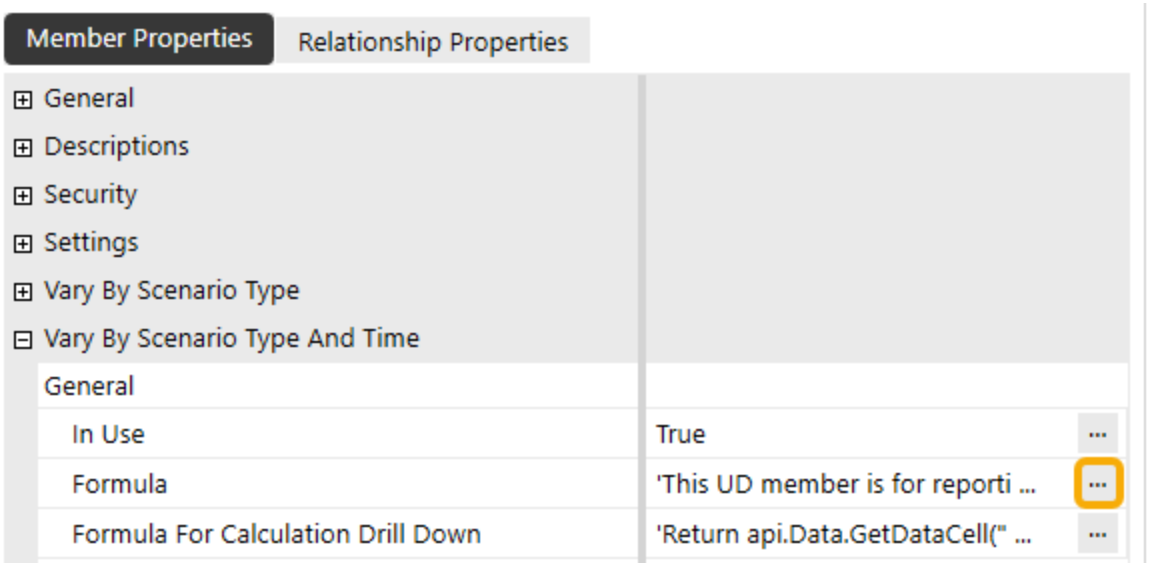
Before you add a dynamic calculation to a Cube View, you must define it or write the simple dynamic calculation.

Complete the following steps to define a dynamic Calculation. In this example, a UD8 member is updated and defined.

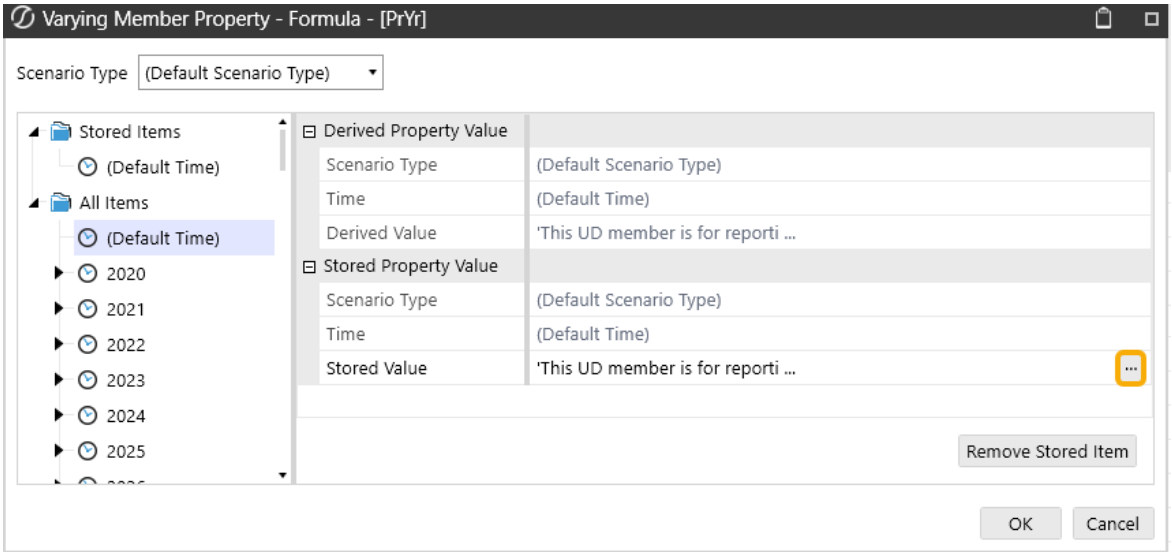
1. Navigate to the **Dimension Library** and open the member. In Settings, set the **Formula Type** property to **DynamicCalc**.



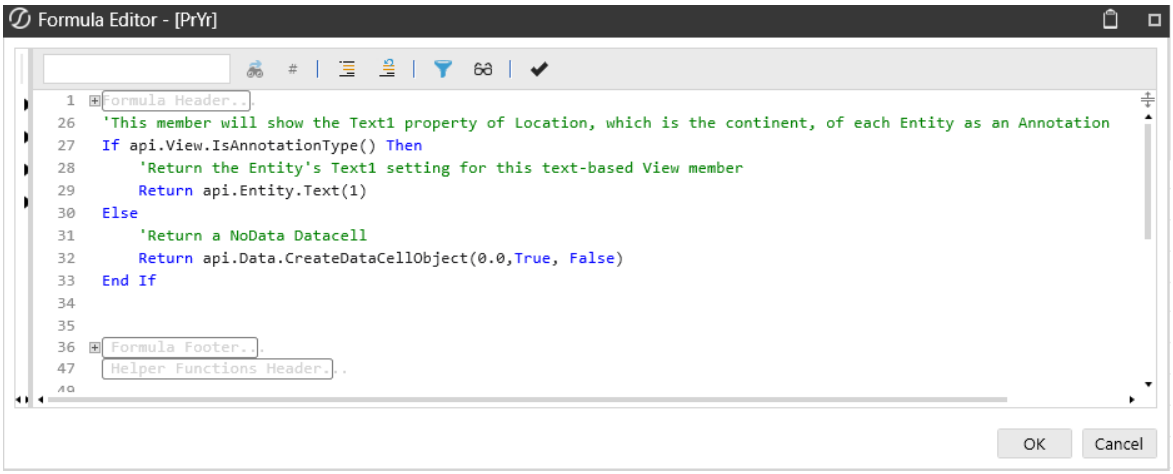
2. In Vary By Scenario Type And Time, go to the **Formula** field and select the ellipses.



3. In the Varying Member Property - Formula window, in Stored Property Value, go to the **Stored Value** field and select the ellipses.



4. In the Formula Editor, write your dynamic calculation. In the following example, the formula displays an entity's Text1 value if an Annotation type View member is referenced.



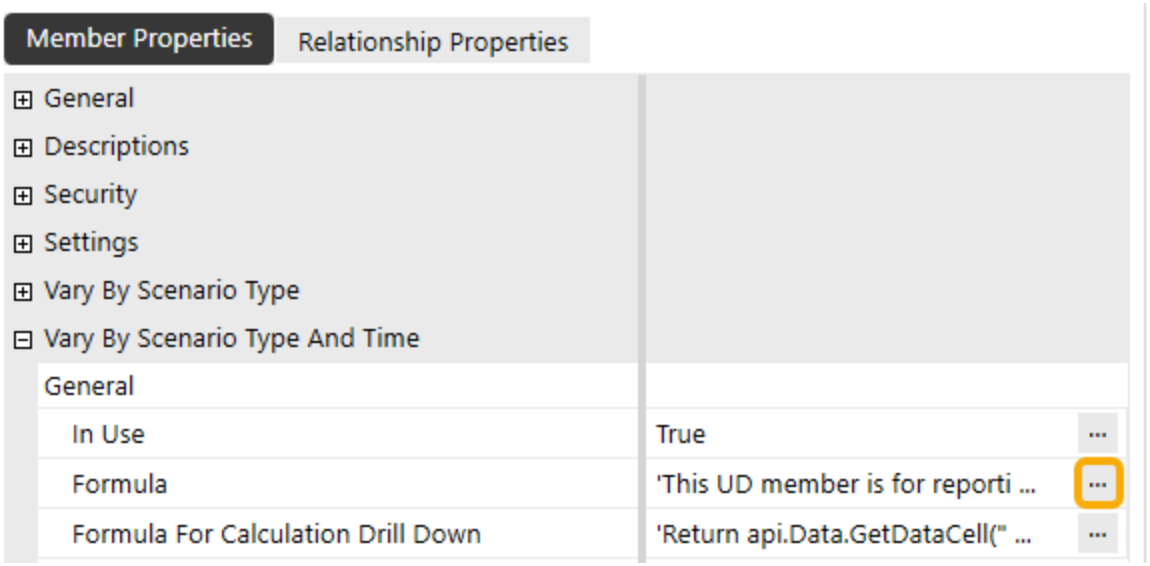

```
'This member displays the Text1 property of Location, which is continent, of each
Entity as an Annotation.
If api.View.IsAnnotationType() Then
    'Return the Entity's Text1 setting for this text-based View member.
    Return api.Entity.Text(1)
Else
    'Return a NoData DataCell.
```

```
    Return api.Data.CreateDataCellObject(0.0, True, False)
End If
```

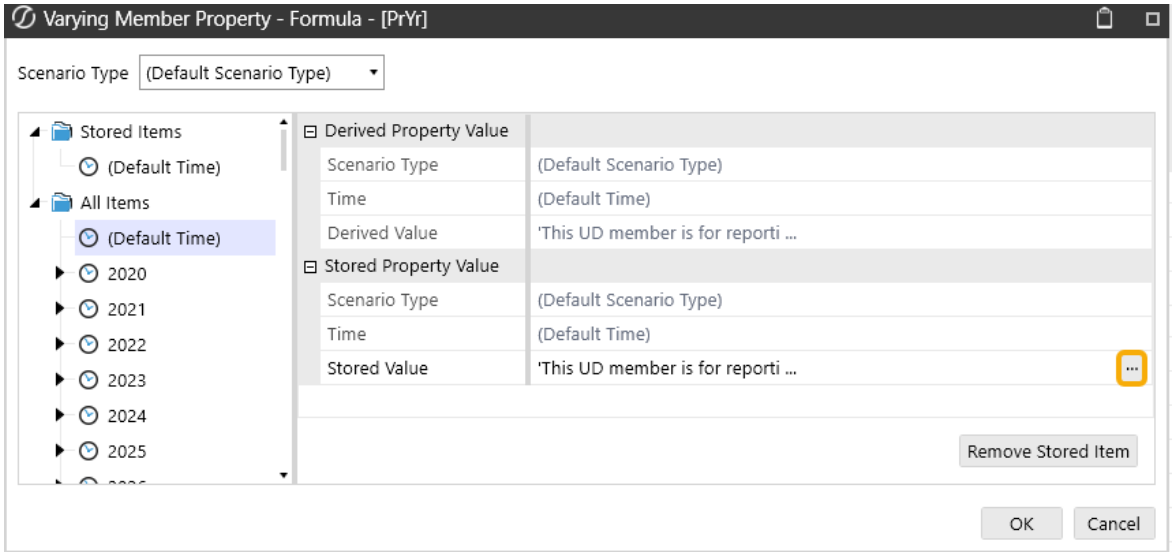
Write a Dynamic Calculation

Write the dynamic calculation before adding it to a Cube View. Complete the following steps:

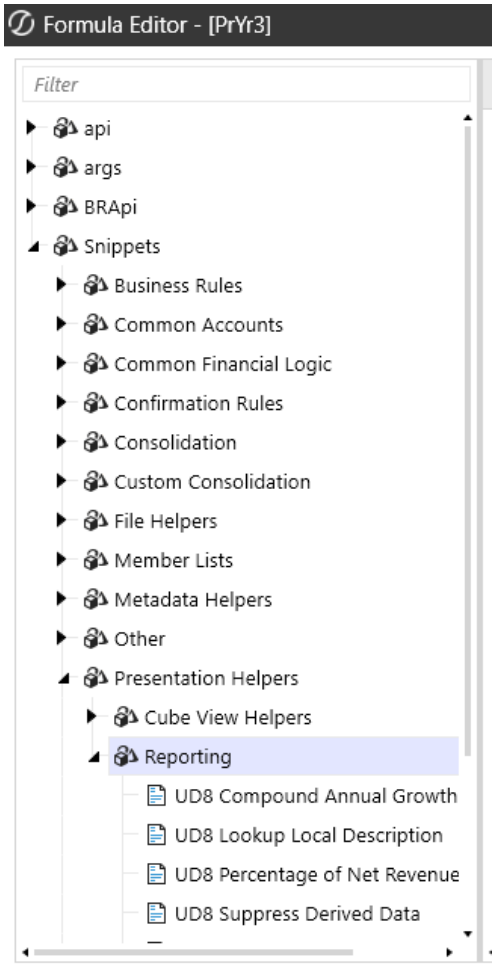
1. In the member you want to update, got to **Member Properties** > **Vary By Scenario Type And Time** > **Formula** and select the ellipses.



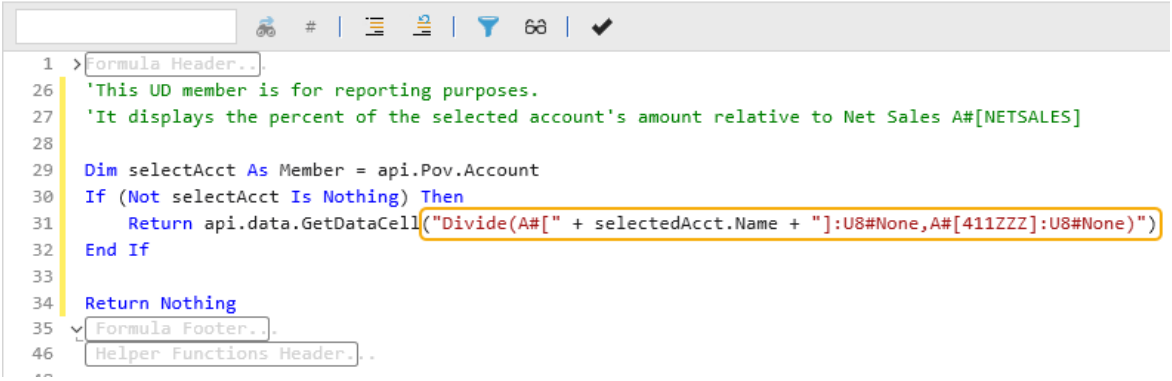
2. In the Varying Member Property - Formula window, in Stored Property Value, go to the **Stored Value** field and select the ellipses.



3. In the Formula Editor window, go to **Snippets** > **Presentation Helpers** > **Reporting**.



4. Select the dynamic calculation you want to use.
5. In the Samples tab, select the **Copy to Clipboard** icon to copy the syntax.
6. Edit the dimensions in the pasted syntax to ensure they are correct.

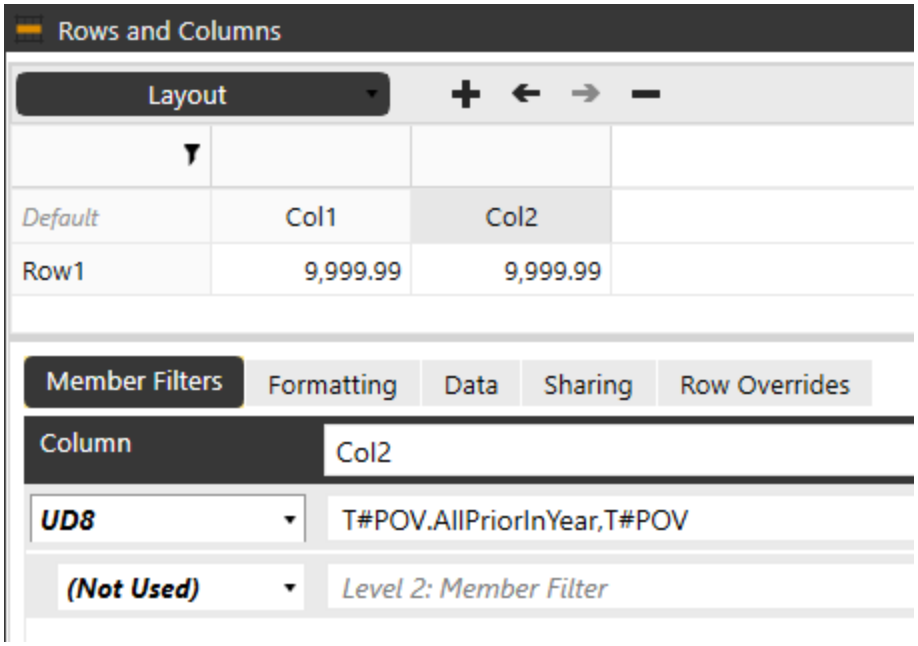


7. Select the **OK** button and then select the **Save** icon.

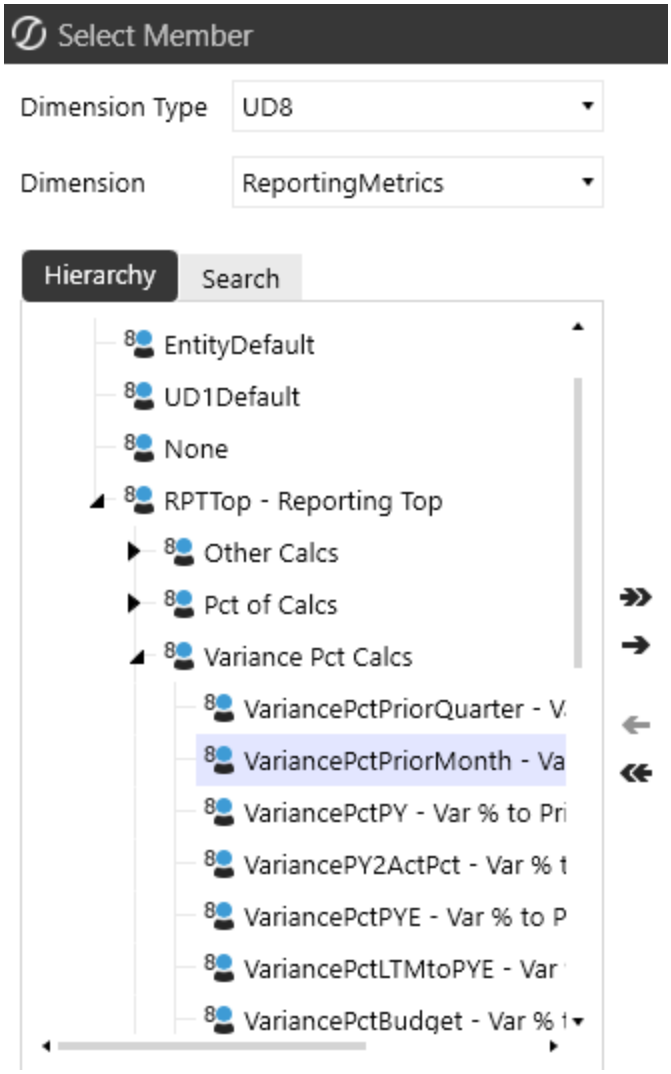
Add a Dynamic Calculation to a Cube View

After you define and write a dynamic calculation, you can add the calculation to a Cube View. Complete the following steps:

1. Go to **Application** > **Presentation** > **Cube Views** > **Cube View Groups** and select a Cube View.
2. Create a new column. All the selections and properties of Col1 are carried over to the new Col2. In this example, the reporting dynamic calculations are located in UD8, so the dimension is updated to UD8.



3. Select the **Member Filter Builder** icon and make your member selections. In this example, VariancePctPriorMonth and Variance PriorMonth are selected.



4. Select the **Save** icon and select the **Open Data Explorer** icon to run the Cube View. In this example, the Cube View displays the dynamic calculations in the two far-right columns.

February 2023 Top	March 2023 Top	April 2023 Top	May 2023 Top	June 2023 Top	June 2023 Variance Pct to Prior Month	June 2023 Variance to Prior Month
335,697.13	486,871.11	633,992.29	780,653.83	929,445.93	0.19	148,792.10
201,680.40	281,831.08	355,957.90	427,900.89	500,390.22	0.17	72,489.33
199,017.42	274,508.15	342,498.23	406,948.51	470,464.01	0.16	63,515.50
337,868.40	488,569.98	634,572.77	779,481.60	925,741.91	0.19	146,260.32
335,697.13	486,871.11	633,992.29	780,653.83	929,445.93	0.19	148,792.10

Override Complex Calculations

When you apply dynamic calculations to both rows and columns, the row formula takes precedence, whether it is a dynamic calculation member or Cube View math.

Rows and columns contain several override properties by row:

- When you use the row override in the column, it will use the column formulas for the specified rows.
- When you use the column override in the rows, it will use the formulas for the specified columns.

Apply Overrides

Complete the following steps to apply overrides:

1. In the Cube View, select a column.
2. Select the **Row Overrides** tab.
3. In the **Row Range** field, define the row range. Enter a row name or apply a range of row names to override the results for the specific rows in the column.
4. In the **Member Filter** field, enter the Member Filter to override the value for the row range. This defines what dimension member data displays or specifies complex calculations for the row range.

5. In the **Cell Format** field, set the format for the row range, which includes selecting fonts, text colors, cell background color, grid lines, cell borders, and so on.
6. Select a cell type and list parameter if needed.
7. Select the **Save**  icon.

Row/Column Overrides

You can have up to four overrides for your column and four overrides for your row. These can be used to change the display for a given column or row.

> **Example:** There are two columns: Flash Scenario (Col1) and Budget (Col2). If Row1 is being used by the Net Sales Account (A#60999) and it must be compared to the Flash Net Sales, which is a different account (A#FlashNetSales), then you would use a column override.

Row/Column Range

Type a column name to display a different value in the Cube View. You must use the name of your rows or columns. The row range does not need a numerical order. Enter the actual name of the rows and columns in your Cube View.

> **Example:** Col1 would be typed to show a different value for Flash Scenario Column.

> **TIP:** Your row range can be a single row or column. To apply to multiple rows or columns, use a comma-separated list or use a hyphen to define a range.

Member Filter

Type a Member Filter to override the value for the columns or rows.

> **Example:** A#FlashNetSales to overrides the results from A#60999. The override results will display the value from A#FlashNetSales in Col1 for the Flash Scenario.

Cell Format

See Rows and Columns Formatting Tab.

List Parameter

See Rows and Columns Data Tab

Presenting Data with Dashboards

Dashboards enable users to view formatted Cube Views, pixel perfect financial reports, and expressive charts. Dashboards are contained in a dashboard group and include components, data adapters, and potentially parameters. All objects are managed and shared across maintenance units. Maintenance units are stored, created, and maintained in Workspaces. See "Workspaces" in the *Developer Guide*.

Workspaces vary by dashboard project needs and applications. They store maintenance units and facilitate community development by providing an isolated environment for developers to segregate and organize dashboard objects.

The primary use for maintenance units is to share key dashboard artifacts like parameters, data adapters, and components across multiple dashboard groups. These objects do not have security access settings, so they assume the settings of the maintenance unit. After a maintenance unit is created, dashboard groups, components, data adapters, parameters, and files can be created within it.

The following process describes how dashboards are created using objects, dashboard groups, maintenance units, and Workspaces:

1. Parameters supply data adapters.
2. Data adapters add data to components.
3. Components make up dashboards.
4. Dashboards fill maintenance units.
5. Maintenance units are stored in dashboard groups.
6. Dashboard groups are contained in Workspaces.



Dashboard groups are created to organize dashboards and function as placeholders where the dashboards reside. The dashboard groups are then available to assign to dashboard profiles for viewing throughout the workflow process.

Dashboards are composed of components. Components can have one to many data adapters and can be used across multiple dashboards and dashboard groups in the same maintenance unit.

Data adapters are the minimum building block for components. These specify where data is coming from for the dashboard component. A list can be made of each data adapter that names the resulting table created when the dashboard runs.

Parameters can be used to filter data in the resulting dashboard. See "Filter Run-time Data and Reduce Maintenance with Parameters" in the *Understanding the Financial Model Guide*.

The File section enables an administrator to create company specific dashboards by uploading documents and images.

> **NOTE:** Optimal display size is 1920 x 1080 resolution. In the Windows application, the recommendation is to set the scale to 100% and use the zoom functionality to zoom in. This is helpful in dashboards such as People Planning and Reporting Compliance.

Dashboard Groups

A Workspace and maintenance unit must be set up before a dashboard group can be created. See "Workspaces Properties" and "Maintenance Unit Properties" in the *Developer Guide*.

To create a dashboard group, select the **Create Group** icon.

General (Dashboard Group)

Name
The name of the dashboard group.

Workspace
The Workspace the dashboard group is in.

Maintenance Unit
The maintenance unit the dashboard group is in.

Description
A short description of how the dashboard group is used or what it contains.

Dashboard Profile Properties

After a dashboard group is created, it can be organized into a dashboard profile. Profiles are then assigned to various functions. To create a dashboard profile, select **Dashboard Profiles** and then the **Create Profile**  icon. To assign a group to a profile, select a profile and then the **Manage Profile Members**  icon. The Workspace name displays with its associated dashboard group.

General (Dashboard Profile)

Name
The name of the dashboard profile.

Description

A short description of how the dashboard profile is used or what it contains.

Client/Device Visibility

This determines which clients or devices will display the dashboard. The supported clients and devices include WinApp, Web, Tablet, and Phone.

> **NOTE:** For a dashboard to display in the Modern Browser Experience and on specific devices, the dashboard profile the dashboard group is assigned to must have Web, Tablet, or Phone selected for this property.

Visibility

Dashboard Visibility controls which dashboards are viewed throughout the application. This property is essential for all dashboard profiles. You can choose from the following options:

- **Never**: This is used for dashboards that are expired or no longer in use.
- **Always**: This allows the dashboard to be available in OnePlace and Workflow.
- **OnePlace**: This allows the dashboard to be viewed in OnePlace > Dashboards.
- **Workflow**: This allows the dashboard to be attached to a workflow profile. This can be completed in the Workflow Profile's Integration and Data Quality settings.

> **NOTE:** If a dashboard is not displaying in the Modern Browser Experience, verify that this property is set to Always or OnePlace.

Security

Access Group

This manages the users that have access to the dashboard profile.

Maintenance Group

This manages the users that have access to maintain and administer the dashboard profile.

To open the Object Lookup window and search for security groups, select the ellipsis button.

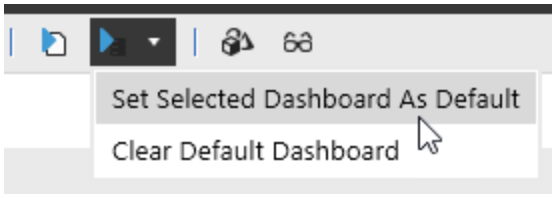
To go to the Security page, select the **Navigate To Security** icon. This enables you to easily make changes to security users or groups before assigning them.

Dashboard Design Mode

Dashboard design mode enables dashboards to be easily designed and edited. It pinpoints exactly where the desired change needs to be made and displays a dashboard draft with all maintenance unit items associated with it. You can select different sections of the maintenance unit hierarchy to highlight the corresponding portion of the dashboard. You can also select a portion of the dashboard to be directed to that exact item in the maintenance unit.

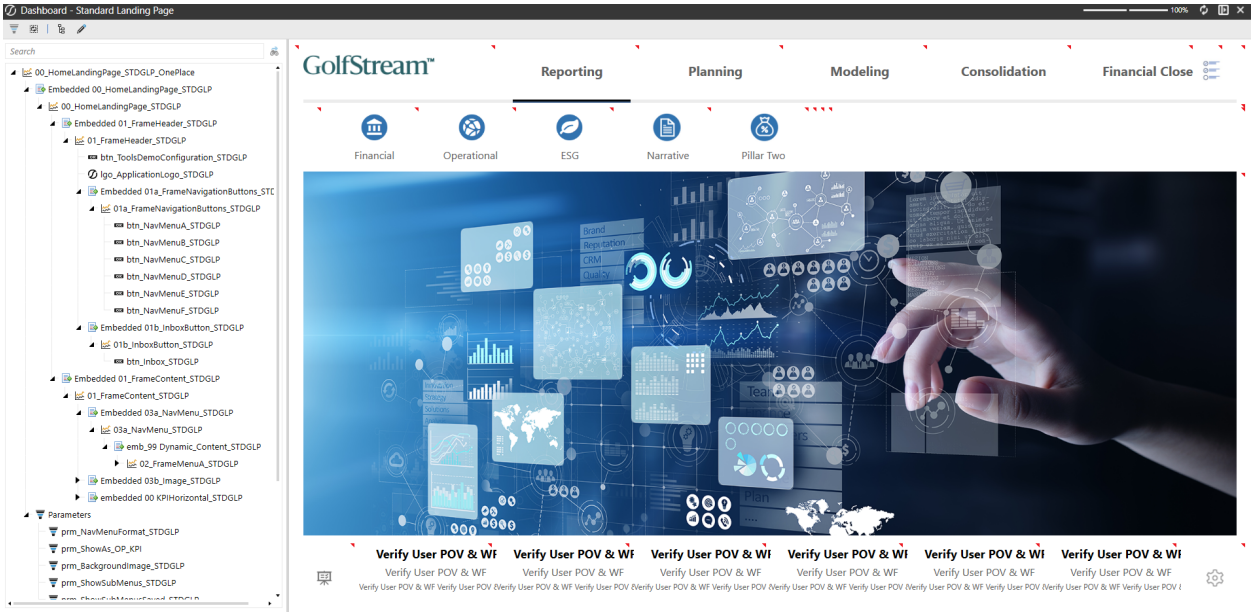
To run a dashboard in design mode, select the dashboard. Select the drop-down arrow and select **Set Selected Dashboard As Default**. Then, select the **View Dashboard In Design Mode** icon.

> **NOTE:** This retains the selected dashboard as the default until it is cleared and a new one set as the default.

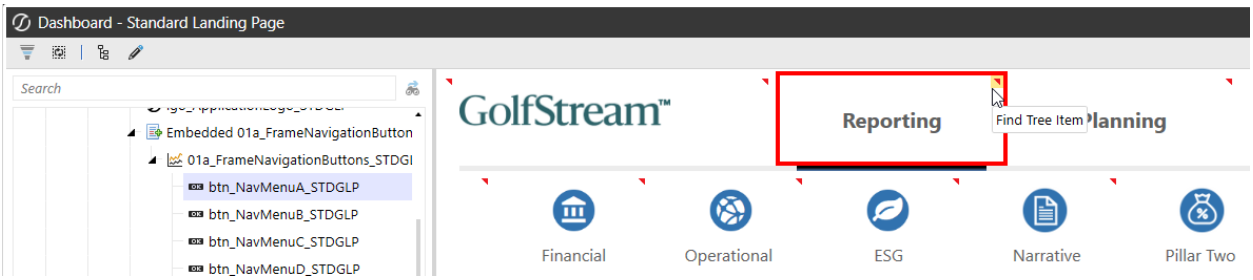


Design Mode

The dashboard displays with a navigation tree showing all maintenance unit items. This provides the entire maintenance hierarchy for the dashboard. The following sample dashboard displays numerous components and parameters. You can hover the mouse over each parameter item to view the value being used.

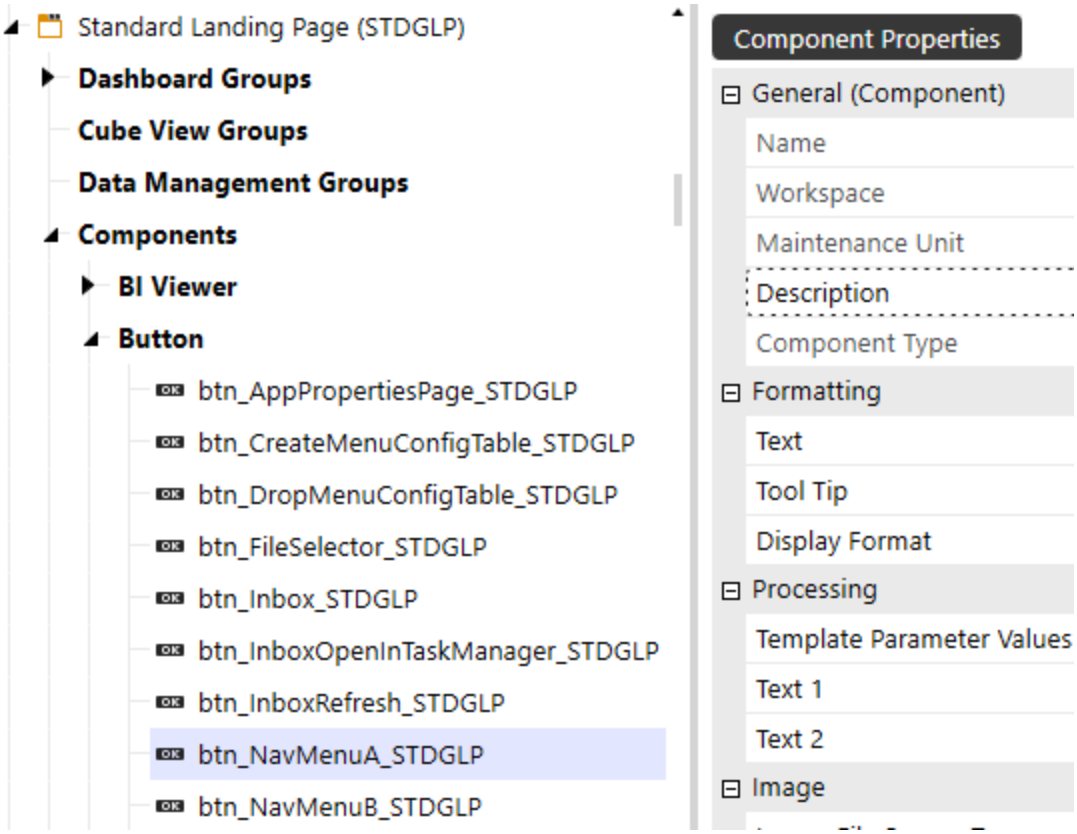


Select the **Find Tree Item** icon to highlight the corresponding maintenance unit item in the navigation tree. You can also select an item from the navigation tree to highlight its corresponding dashboard element.



To hide or display the maintenance unit items, select the **Toggle Design Mode** icon.

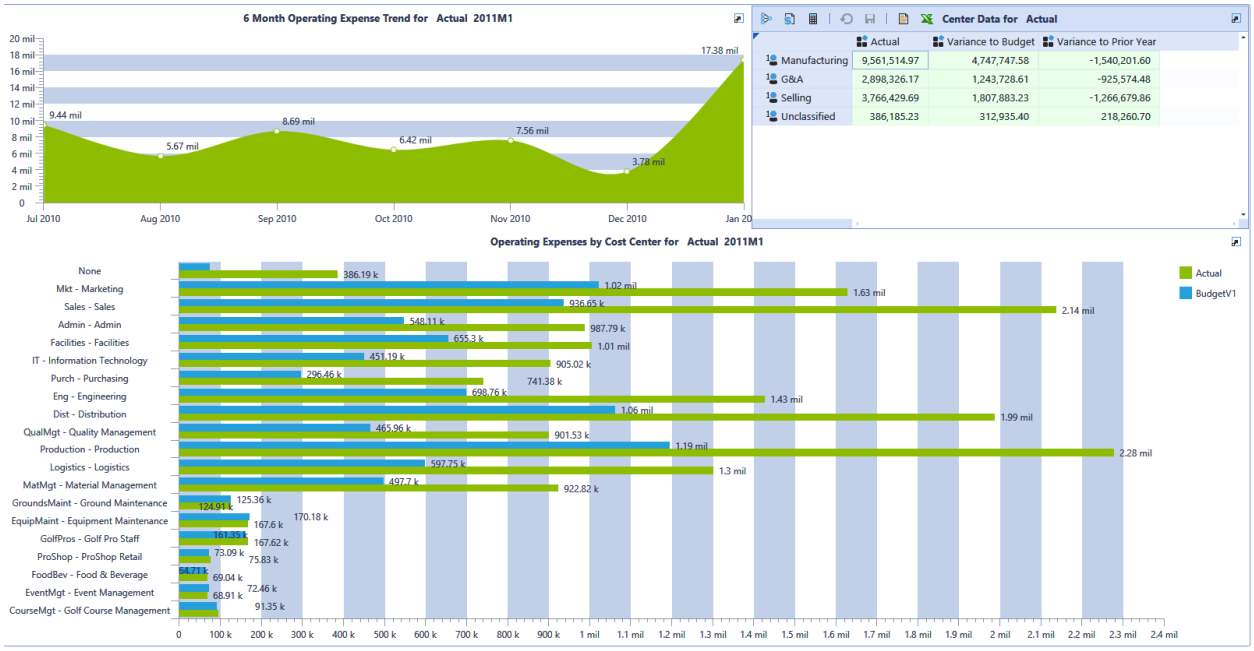
To navigate back to the Workspaces page and modify the item in its Maintenance Unit, select the **Edit Component** icon.



NOTE: If the dashboard includes a dialog box, the preview tree displays but the Edit Component icon cannot be used.

View Dashboards

Select the **View Dashboard** icon to display a preview of the dashboard. This view enables you to review your current configurations, apply various parameters, and return to the Workspaces page to make updates as needed.

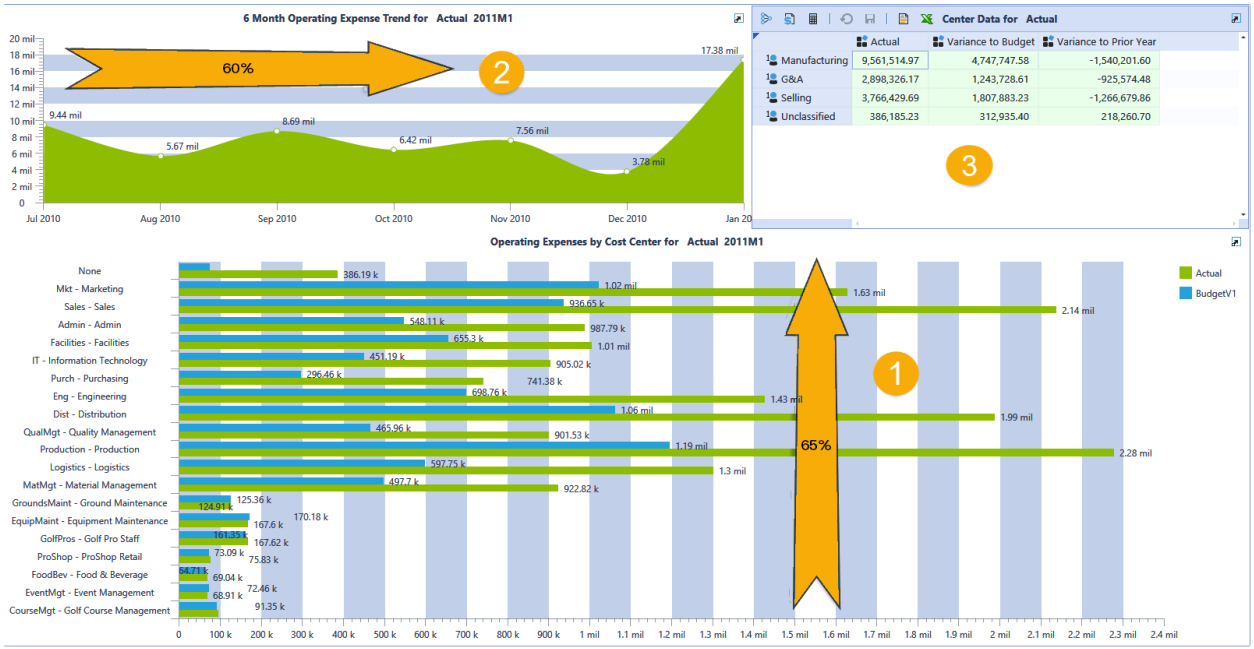


To run the dashboard again under a different input value, select the **Select Parameters** icon.

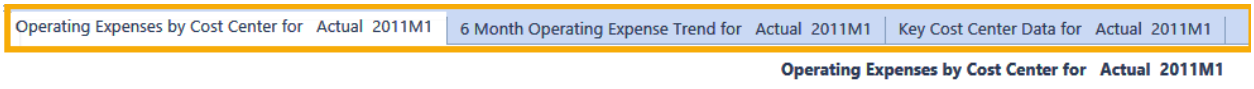
To return to the dashboard on the Workspaces page, select the **Edit Dashboard** icon.

The following example shows a dashboard with two Chart components and one Data Explorer component.

- Component 1 has the Dock Position set to Bottom and Height to 65%.
- Component 2 has the Dock Position set to Left and Width to 60%.
- Component 3 has the Dock Position set to any position such as Right and all other setting are left blank. The dashboard will automatically dock this component to fill the rest of the available space.



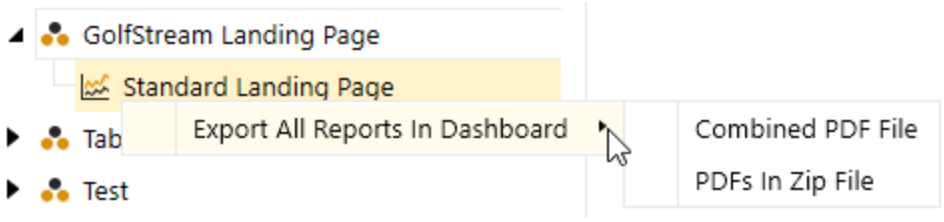
You can set the dashboard layout type to Tabs to view each component in its own tab.



In OnePlace > Dashboards, you can select a dashboard profile and export all dashboards in the profile out to a combined PDF file.

1. Right-click the dashboard profile and hover over **Export All Reports In Profile**.
2. Select **Combined PDF File** or **PDFs In Zip File**.

This is a quick way to produce report books. The only exception is that currently graphs and charts do not export to PDF in these mass forms.



Dashboard Properties

To create a dashboard, select the **Create Dashboard** icon.

General (Dashboard)

Name
The name of the dashboard.

Workspace
The Workspace the dashboard is in.

Maintenance Unit
The maintenance unit the dashboard is in.

Dashboard Group
The dashboard group the dashboard is in.

Description
A brief description of the dashboard.

Page Caption
This is used to give a name to the dashboard when it is displayed. The name will be displayed at the top left of the dashboard. If the Page Caption is left blank, it will default to the name given to the dashboard.

Notes
Enter notes or usage instructions for dashboard implementation. For Custom Control dashboards, it is recommended that you provide information about its Required Input Parameters, Event Listener Names, and Event Listener Output Parameter Names that need to be handled in the parent Embedded Dashboard component.

Behavior

Dashboard Type
This indicates how each dashboard within a group is used. It helps reduce the number of groups needed and simplifies dashboard maintenance.

- **(Use Default)**: Default is Top Level.
- **Top Level**: This dashboard type indicates which dashboards in a group should be visible if assigned to OnePlace or a workflow.
- **Top Level Without Parameter Prompts**: This dashboard type has the same behavior as Top Level, but if the dashboard has runtime parameters, it will have a slight performance improvement. It does not prompt for parameters.
- **Embedded**: This dashboard type indicates which dashboards in a group should not be visible in OnePlace because they are dependent on other dashboard objects.

- **Embedded Dynamic**: This dashboard type enables you to dynamically modify dashboard properties, component properties, parameters, adapters, and Cube View properties. See "Dynamic Dashboards" in the *Developer Guide*.
- **Embedded Dynamic Repeater**: This dashboard type enables you to dynamically modify dashboard properties, component properties, parameters, adapters, and Cube View properties. This dashboard type can be used to add multiple instances of the same component without having to recreate each component individually. See "Embedded Dynamic Repeater Dashboard" in the *Developer Guide*.
- **Embedded Top Level**: This dashboard type acts as a new top level with its own parameter resolution and rendering stages. As a result, it allows for the Load Dashboard Server Task to be run even when embedded.

  > **NOTE:** Currently, Embedded Top Level dashboards should not be used with Smart Links.

- **Embedded Top Level Without Parameter Prompts**: This dashboard type acts as a new top level with its own parameter resolution and rendering stages. As a result, it allows for the Load Dashboard Server Task to be run even when embedded. It does not prompt for parameters, resulting in a slight performance improvement.

  > **NOTE:** Currently, Embedded Top Level Without Parameter Prompts dashboards should not be used with Smart Links.

- **Custom Control**: This dashboard type has the same behavior as Embedded, but each instance maintains its own set of input parameters, variables, and Event Listeners (actions).

Custom Control

A new dashboard type with required input parameters, which is a comma-separated list of parameter names that should be provided by the containing dashboard when processing a custom control dashboard. For example, Param 1, Param 2, Param 3. A parameter component inside the custom control can forward events to the parent dashboard.

> **NOTE:** No validation of this field is performed by either the control or the contained dashboard. No error is generated by default if parameter values are not correctly provided on either side. Developers must ensure the content of this field matches actual parameters being used or reacts to incorrect states.

Literal Parameter Values

Name Value Pairs (for example, Param1=Value1)

Enter a comma-separated list of name-value pairs for this dashboard's static parameter values. Piped variable names cannot be used in this setting.

Formatting

Layout Type

Select a layout type for the dashboard. You can choose from the following options:

- **Canvas (Windows App Only)**: A component docks to the top, bottom, right, or left of the dashboard. The component only shows in the space defined by the dock position and does not display fully on a screen. Slide bars are used to view the components. When you add a component to a dashboard with this layout type, select a dock position and specify the component's Left, Top, Width, and Height settings.

- **Dock (Windows App Only)**: A component docks to the top, bottom, right, or left of the dashboard. When adding a component to a dashboard with this layout, use the dock position selection to determine how much of the screen is used and specify either Width or Height settings.

  Dock positions Left and Right are used in combination with Width. Dock positions Top and Bottom are used in combination with Height.

- **Grid**: This enables you to make a dashboard in a grid format by using rows and columns to add components, lines, or movable splitters. This selection displays the Grid Layout Type properties. The Position On Dashboard properties are ignored for components added to a dashboard with this layout type.

- **Horizontal Stack Panel**: Each component will display horizontally with one row and at least one column. Horizontal Stack Panel dashboards are generally used as part of other dashboards. Position On Dashboard properties are ignored for components added to a dashboard with this layout type.

- **Tabs**: Each component will display in its own tab. Position On Dashboard properties are ignored for components added to a dashboard with this layout type.

- **Uniform**: Each component is docked, but all are equal in size. Position On Dashboard properties are ignored for components added to a dashboard with this layout type.

  > **NOTE:** If the Toggle Component Size button is used, the dashboard is rendered using Uniform Layout Type.

- **Vertical Stack Panel**: Each component will display vertically with one column and at least one row. Vertical Stack Panel dashboards are generally used as part of other dashboards. Position On Dashboard properties are ignored for components added to a dashboard with this layout type.

- **Wrap (Windows App Only)**: The layout container wraps each component according to its specified size. When adding a component to a dashboard with this layout, select a dock position and specify either Width or Height settings.

Is Initially Visible If Embedded

When set to True, the selected dashboard is hidden if it is retrieved from another dashboard. To set the hidden dashboard as visible from a component, go to **Component Properties** > **Action** > **User Interface Action** > **Selection Changed User Interface Action** and select **Refresh**. In the **Dashboards To Show** field, list the dashboard containing the embedded dashboard.

Display Format

This enables unique display formatting for dashboard background color, dialog box width and height, and border formatting. See Display Format Settings.

Show Title

The default value for this property is True. When this property is True, the dashboard title displays in the Modern Browser Experience. This property does not apply to the Windows application.

Scroll Position

This property enables you to configure the position of the scroll bar when you refresh for Horizontal Stack Panel and Vertical Stack Panel dashboard layout types. This property is turned off for other dashboard layout types. You can choose from the following options:

- **Default (Beginning)**: When the dashboard is refreshed, the scroll bar is positioned at the beginning of the dashboard. The property defaults to this value.
- **End**: When the dashboard is refreshed, the scroll bar is positioned at the end of the dashboard.
- **Save Current Position**: When the dashboard is refreshed, the scroll bar maintains its current position on the dashboard.

Action (Primary Dashboard Only)

Load Dashboard Server Task

You can run dashboard extender business rules when launching and refreshing dashboards. This only applies to the main dashboard, not embedded dashboards. You can choose from the following options:

- **No Task**: A dashboard extender business rule is not being used on this dashboard.
- **Execute Dashboard Extender Business Rule (Once)**: This runs the assigned business rule once at refresh or initial launch.
- **Execute Dashboard Extender Business Rule (All Actions)**: This does an iterative action and runs the business rule as many times as needed.

Load Dashboard Server Task Arguments

Enter the arguments required by the server task. Use the following syntax for the arguments: {BusinessRuleName}{FunctionName}{Optional Named-Value Pairs}. Select the ellipsis and select the desired argument format and then enter each object.

> **Example:** {DashboardLoadBRName}{TestFunction}
> {Param1=[Value1], Param2=[Value2]}

Grid Layout Type

These properties are only available if Grid is chosen as the dashboard Layout Type.

Number of Rows

The number of rows the dashboard will use for the grid layout.

Number of Columns

The number of columns the dashboard will use for the grid layout.

Row/Column Type
Select the row and column type. You can choose from the following options:

- **Component**: Use this option if a component is needed for the row or column of the grid layout for the dashboard.
- **Line**: Use this option if a line is needed for the row or column on the dashboard.
- **Movable Splitter**: Use this option if a movable splitter is needed for the row or column on the dashboard. This enables you to move the grid up and down or left and right.

Row/Column Height/Width
Use Row/Column Height/Width to customize the height and width of the rows and columns to be displayed on the dashboard. Auto expands the size based on the contents of the row. You can use an asterisk to create a fractional value of the remaining space. For example, if there are three rows, then each row will take a third of the space.

Display Format Settings

Dialog

DialogDisplayStyle
This controls how a dashboard is sized when viewed as a dialog box. If the specified value exceeds the current screen, the value is changed to the maximum display setting for height and width to fit to screen. You can choose from the following options:

- **Maximize**: This opens the Dialog Maximize mode to fill screen.
- **Normal**: This is the default setting. It opens to the defined Dialog Height and Width settings.

DialogWidth
Customize the width of the dialog box for the dashboard. The dialog box width supports up to 10,000. If set to -1, the size sets at runtime to the maximum width for the active screen.

DialogHeight

Customize the height of the dialog box for the dashboard. The dialog box height supports up to 10,000. If set to -1, the size sets at runtime to the maximum height for the active screen.

EnableSystemClose

This is used to control the display and use of the dialog box Close button. Setting this to True removes the Close button.

Background

BackgroundColor

This is used to display the background color of one dashboard.

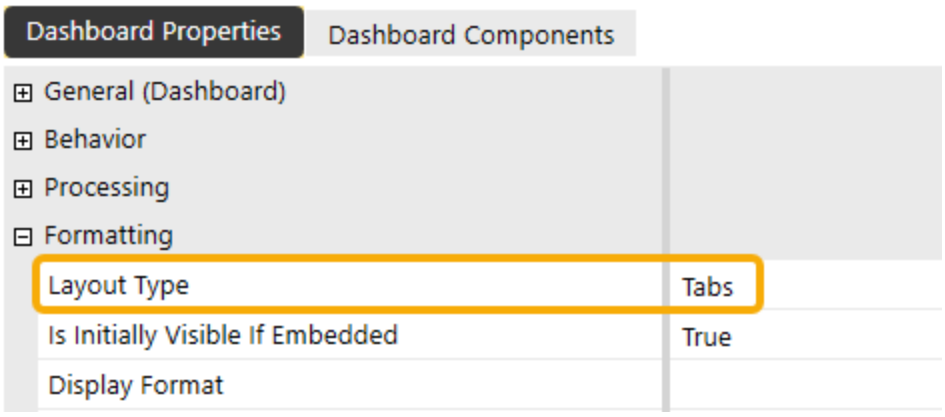
InnerBackgroundColor

This is used to define the background color of a dashboard containing embedded dashboards.

Styles

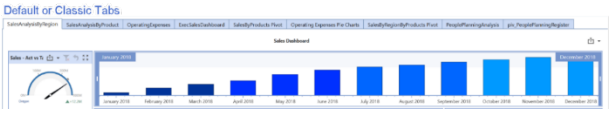
TabControlStyle

Select a tab style. The tab control functionality remains the same. This property is enabled when the Layout Type property is set to Tabs. See Dashboard Properties.



You can choose from the following options:

- **(Use Default) or Classic**: The original style for dashboard tabs.



- **No Border**: Rounded corners and no borders.



- **Rounded Corners**: Rounded on the edges with borders.



> **NOTE:** These tabs are only available for use with the dashboards and do not apply to the rest of the product.

Border

BorderThickness

This is used to customize the border thickness for the dashboard.

BorderCornerRadius

This is used to customize the border corner shape for the dashboard. If left blank, the border will be square.

BorderColor

This is used to customize the outer border coloring for the dashboard.

BorderBackgroundColor
This is used to specify the border background color of a dashboard containing embedded dashboards.

Margin

BorderMarginLeft/Top/Right/Bottom
This is used to customize the margin of the border on the dashboard.

Padding

BorderPaddingLeft/Top/Right/Bottom
This is used to customize the padding or spacing of the border on the dashboard.

Dashboard Components

Components enable you to build the dashboard's user interface and display data. You can use components across multiple dashboards and dashboard groups in the same maintenance unit.

Create a Dashboard Component

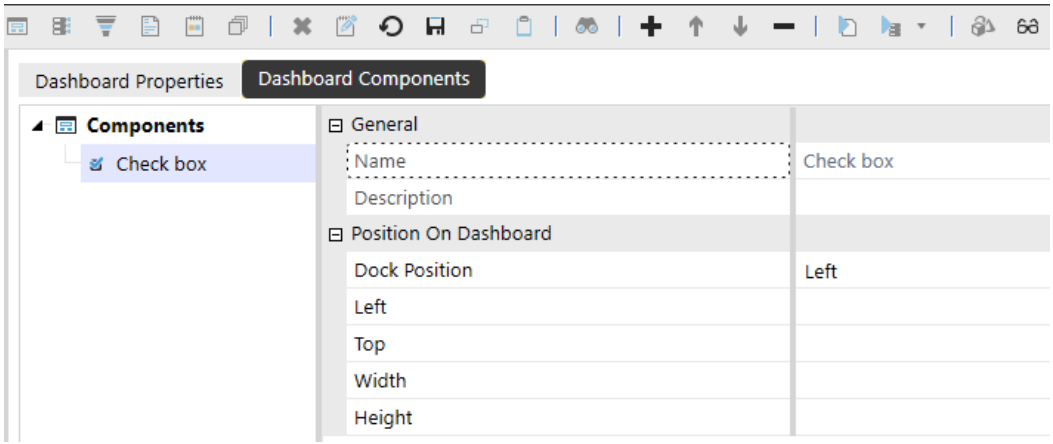
 Select the **Components** node and then the **Create Dashboard Component** icon. From the **Create Dashboard Component** window, select a component, then select the **OK** button.

In the **Component Properties** tab, name the component. You can add parameters and other substitution variables into the description so that they display in the resulting component when run in a dashboard.

> **Example:** Account: |!TrendAccountParam!| Year:|WFYear|. If the existing and accessible Parameter TrendAccountParam returns Net Income Region and the user's Workflow POV Year is set to 2012, the description text Account: Net Income Region Year: 2012 will display when processed.

Manage Components on a Dashboard

Navigate to the dashboard in your maintenance unit and select the **Dashboard Components** tab.





To add and assign a dashboard component, select the **Add Dashboard Component** icon.



To move a dashboard component up in the Components list, select the component and then the **Move Up** icon.



To move a dashboard component down in the Components list, select the component and then the **Move Down** icon.



To remove a dashboard component from the Components list, select the component and then the **Remove Selected Dashboard Component** icon.

> **NOTE:** Press **CTRL** to multiselect components from the Add Dashboard Component window.

Position On Dashboard Properties

Dock Position
This enables you to dock a component in a specific position on the dashboard. You can choose from the following options: Left, Right, Top, and Bottom.

Left
This sets the amount of dashboard space to allocate to the left dock position.

Top
This sets the amount of dashboard space to allocate to the top dock position.

Width
This property only applies to Dock and Canvas dashboard layout types. Enter a percentage or pixel value to set the width of the dashboard.

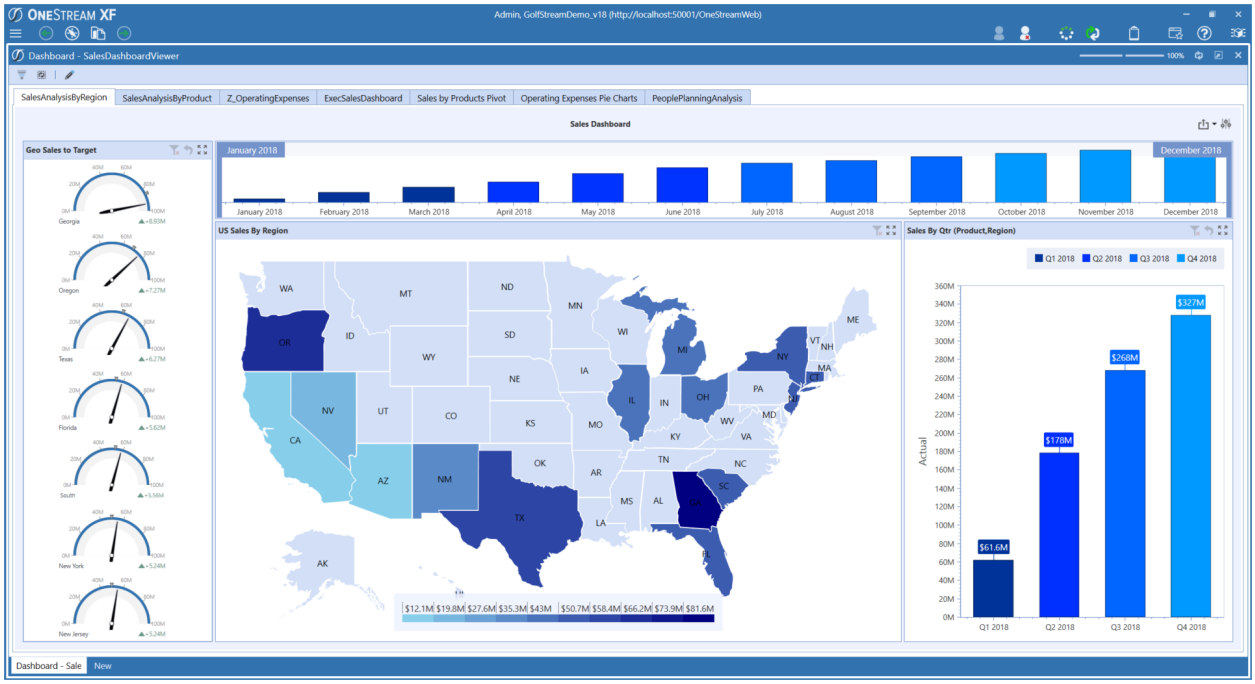
Height
This property only applies to Dock and Canvas dashboard layout types. Enter a percentage or pixel value to set the height of the dashboard.

> **NOTE:** The Position On Dashboard settings do not apply to some dashboard layout types. See Dashboard Properties.

BI Viewer

This is an Interactive Dashboard Engine that is used to quickly create a business intelligence (BI) visualization of data from existing or new data adapters. You can quickly create BI dashboards that enable you to visualize and analyze data.

The BI dashboards are a powerful tool that allow you to dynamically add data and build charts, gauges, maps, grids, cards. This BI component integrates well with the Cube View MD data adapter. When designing, it enables you to add calculated fields to the data source, define Drill Down paths, apply multi-select filtering, use conditional formatting for dynamic viewing at runtime, and customize palette colors. See the *BI Viewer Guide*.



Key Elements of the BI Viewer

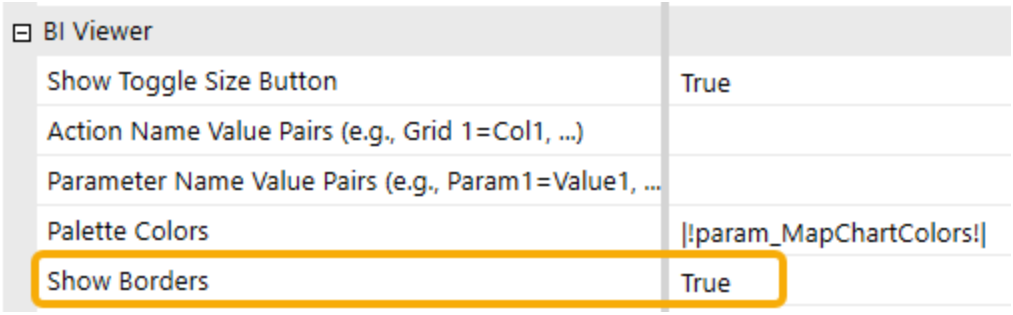
- Compatible with existing dashboards
- Simple to build, share, and reuse components
- Fast drag and drop ad hoc data exploration
- Easily combine OneStream and non-OneStream data
- Bring trusted, validated OneStream data users

Key Items for a Quick Start

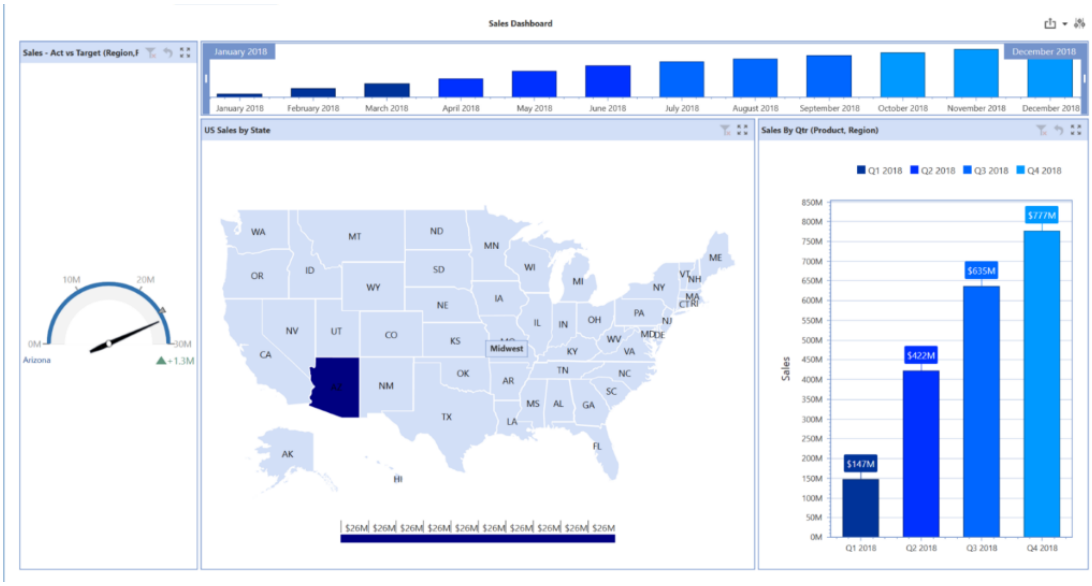
- **Data Adapter**: To source the data and render the data set
- **BI Viewer component**: To design the dashboard
- **Dashboard component**: To display the BI dashboard
- **Literal Parameter (optional)**: To store the color palette

Show or Hide Borders

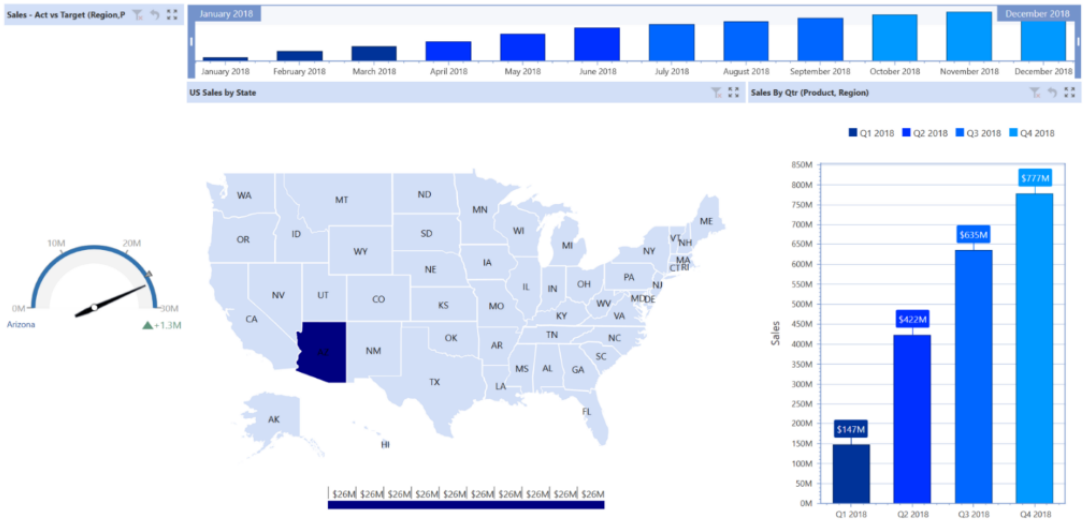
To use borders to separate areas on dashboards, go to **Component Properties** > **BI Viewer** and set the **Show Borders** property to **True**.



The following image displays a BI Viewer component with the Show Borders property set to True.



The following image displays a BI Viewer component with the Show Borders property set to False.



For additional information on the use and design elements of BI Viewer, see the *BI Viewer Guide.*

Book Viewer

This component is used to display a Report Book in a dashboard.

File Source Type
Select a source type for the file. You can choose from the following options:

- **Workspace File**: Display a Book stored in a Workspace maintenance unit File section.
- **Application Database File**: Display a Book stored in the Application Database Share.
- **System Database File**: Display a Book stored in the System Database Share.
- **File Share File**: Display a Book from the File Share.

Full File Name
The name of the Book. Select the ellipsis and browse in the **File Explorer** to the Book's file name.

Show Header
When set to True, this displays a header derived from the component's name or description.

Show Toggle Size Button
When set to True, this enables the toggle button and lets users select the size of the component at runtime. When set to False, the toggle button is hidden.

Initial Content Item
Enter a number to navigate to a specific page in the Report Book when the dashboard runs.

Button

This displays a button on a dashboard. Buttons can be used to initiate a variety of functions including business rules, dashboards, and file uploads.

Image

Image File Source Type
The image displayed on the button. You can choose from the following options:

- **Url**: The image displayed on the button is based on a URL.
- **Workspace File**: The image displayed on the button is in a file stored in the Workspace maintenance unit File section.
- **Application Database File**: The image displayed on the button is in a file in the Application Database Share.
- **System Database File**: The image displayed on the button is in a file in the System Database Share.
- **File Share File**: The image displayed on the button is in the File Share.

Image Url Or Full Name File
The URL, file path, or file name for the displayed image. Review the following syntax examples:

- **Built-in OneStream Image**: /OneS.ClientImages;component/Misc/ToolbarImage.png
- **Web Server**: http://www.onestreamsoftware.com/img/onestream-logo.png
- **Database File System**: Enter the path and file name, Documents/Public/TestImage.png
- **Workspace File**: Enter the name of the file resource, TestImage.png

The following Excel properties are used to specify an Excel file from either an Extensible Document or Report Book to display it as an image or button.

Excel Page Number
Enter the page number of the Excel document to display.

Excel Sheet
Enter the name of the Excel worksheet to display.

Excel Named Range

Enter the named range from Excel to display. The named range must be in the specified Excel sheet in the Excel Sheet property, or the Excel Sheet property must be blank to use a named range. See Component Display Format Properties to position and align the Excel image as needed.

Button

Button Type

Specify the functionality assigned to the button. You can choose from the following options:

- **Standard**: Displays a generic button that can have custom functionality assigned, such as display a dashboard or run a business rule.
- **File Explorer**: This sets the button to display the File Explorer dialog box.
- **File Upload**: This sets the button to display the File Upload dialog box.
- **Select Member**: This sets the button to display the Select Member dialog box. This selection displays these Select Member properties:
    - **Dimension Type**: Specify the dimension type. This is required.
    - **Use All Dimension for Dimension Type**: When set to True, it overrides the Cube Restriction and displays all dimension types. When set to False, the dimensions are limited to defined dimension types.

      > **NOTE:** If no dimension type is defined, the selections revert to the default dimension type.

    - **Dimension**: Specify a specific dimension. This is optional.

      > **Example:** HoustonSalesRegionalBudget

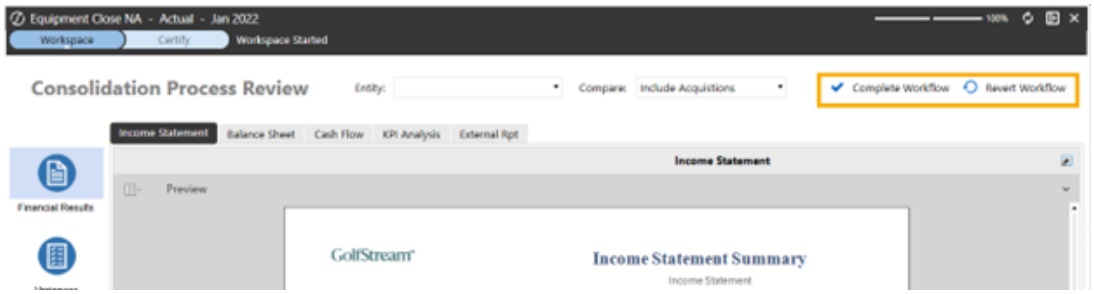
- **Cube**: Specify a cube. This is optional.
- **Member Filter**: Specify a Member Filter. This is optional.

> **Example:** U3#Root.TreeDescendantInclusive

- **Member Filter Builder**: This sets the button to display the Member Filter Builder dialog box. This selection displays the Default Member Filter property.
  - **Default Member Filter**: Specify a default Member Filter to display upon opening. This is optional.

> **Example:** E#[Total Golfstream]

- **Workflow**: This sets the button for Complete and Revert Workflow, replacing the need for a custom business rule. This selection displays these workflow properties:
  - **Workflow Button Action Type**: Complete Workflow enables the complete action button. Revert Workflow enables the revert action button.
  - **Workflow Button Image Type**: Choose one of the standard button images. The Use Button Image option enables you to use a custom image set up in the Image properties.



Action

This section contains Action properties specific to the button component. For a full description of all Action properties, see Component Formatting and Action Properties.

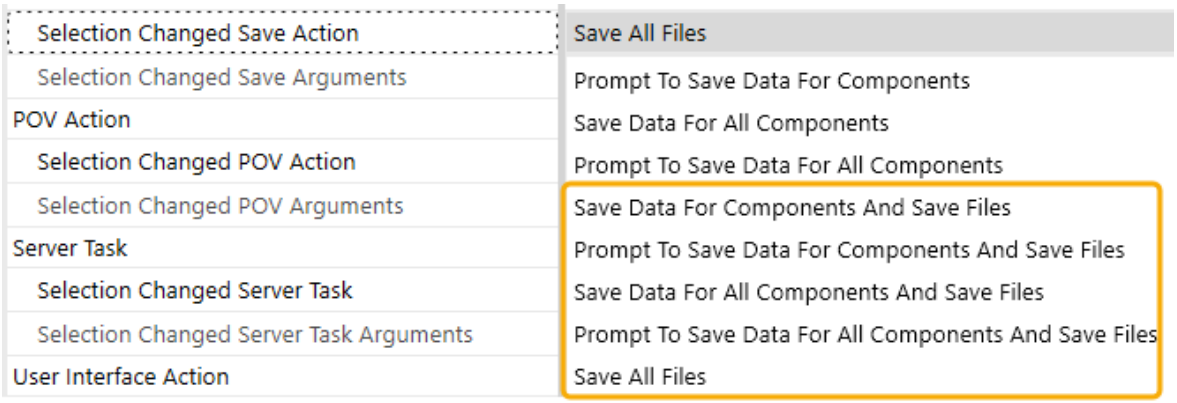
Parameter Value for Button Click
The value or string to pass into the Bound parameter.

Save Action

Selection Changed Save Action
What occurs when the button is pressed. The following options are specific to the Button component:

- **Save Data for Components and Save Files**: This saves the components modified by the dashboard components specified in the Selection Changed Save Arguments property and their files.
- **Prompt to Save Data for Components and Save Files**: This asks the user if the modified data should be saved or not for the defined list of dashboard components and their files.
- **Save Data for All Components and Save Files**: This saves all components modified by the dashboard components included in the dashboard and their files.
- **Prompt to Save Data for All Components and Save Files**: This asks the user of the modified data should be saved or not for all dashboard components and their files.
- **Save All Files**: This saves all modified file data within the dashboard excluding any modified component data.

> **NOTE:** The Selection Changed Save Argument field is only active when you choose a Selection Changed Save Action that requires you to specify components.



You can use these Save Files properties to save edits made in Text Editor and Spreadsheet component files through a dashboard button.

> **Example:** The user has a dashboard containing a spreadsheet and Button component, and the developer sets the Button component's Selection Changed Save Action property to one of the Save Files options. When the user edits the file data in the Spreadsheet component's file and then selects the dashboard button, all file changes are saved. The dashboard component changes will also be saved. Component changes are only not saved if the Save All Files property is selected, which only saves modified file data.

You can use these Save Files properties through Button components instead of having to navigate to File > Save in the component's toolbar to save file changes.

> **NOTE:** Since Text Editor components cannot have nested components, the Save Data For Components And Save Files, Save Data For All Components And Save Files, and Save All Files options will only save the Text Editor component's file.

> **NOTE:** When working in a Spreadsheet component embedded in a dashboard, the Save Files properties enable users to save nested components while also saving the spreadsheet file. When the dashboard button is set to Save All Files, only spreadsheet file modifications are saved.

Use Case

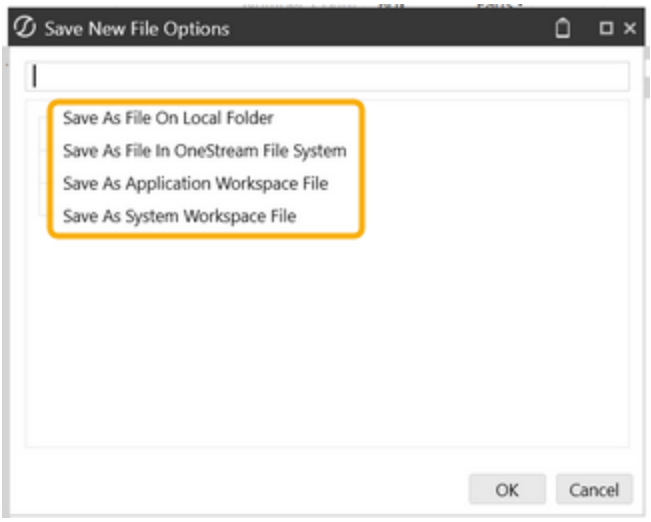
This Use Case demonstrates the following situation:

- **Scenario**: A user is in the Narrative Reporting solution, which uses a dashboard Text Editor component that displays the content of a specific file. The developer uses a Save File action to enable users to save their file edits by using the Check In button.
- **Role**: Developer
- **Benefits**: The developer can streamline the save process for users in Narrative Reporting by reducing the possibility of users checking in files without saving. When the user selects the Check In button, it performs the same save action as the Text Editor's File > Save toolbar button in the ribbon by saving all file modifications.

Save As

If the Text Editor or Spreadsheet components are not linked to or displaying an existing file, selecting a Button component with the Save File options opens the Save New File Options window. This enables users to save a new file in different locations and performs the same function as the component's File > Save As toolbar button action.

> **NOTE:** If you have already used the Button component to Save As and have created a new file, when you select the same Button component again, the updates made in the dashboard component will save to that initially created file. It will not prompt the Save New File Options window to create a second new file until the you refresh or reload the dashboard.



Local Folder

Select a location on local computer or network to save a file to.

> **NOTE:** The Save As File on Local Folder option is not available in the Modern Browser Experience.

OneStream File System

Select a location within OneStream File Explorer to save a file to.

Application Workspace File

Select an Application Workspace location to save a file to.

System Dashboard File

Select a System Workspace location to save a file to.

Chart (Basic)

This is used to display a chart in a dashboard.

> **IMPORTANT:** The Basic Chart dashboard component will be deprecated in a future release. Before upgrading to a 2026 Platform Release, we recommend all customers confirm Basic Chart compatibility in the Release Notes and determine if any of their dashboards or applications are using the Basic Chart. After it is deprecated, the Basic Chart component will no longer function. Therefore, dashboards should use the more versatile, and browser-compatible, Advanced Chart or BI Viewer components. In cases where a OneStream Solution is impacted by the Basic Chart component, new Solution versions will be made available on Solution Exchange to support 2026 upgrades.
>
> Support for the Basic Chart on earlier releases will follow the policies defined in the "Product Support and Maintenance Policy" found on the Community "Commercial Terms and Conditions" page. See OneStream Knowledge Base article KB0015315 to assist with application analysis for the presence of the Basic Chart dashboard component.

Chart Type

Area

An area chart or area graph displays graphically quantitative data and it is based on the line chart. The area between axis and line are commonly emphasized with colors, textures, and hatchings. Commonly, two or more quantities are compared with an area chart. Area charts are used to represent cumulated totals using numbers or percentages over time. Use the area chart to show trends over time among related attributes.

Bar

A bar graph is a chart that uses vertical bars to show comparisons among categories. One axis of the chart shows the specific categories being compared, and the other axis represents a discrete value.

Bubble

A bubble chart is a variation of a scatter chart in which the data points are replaced with bubbles, and an additional dimension of the data is represented in the size of the bubbles.

CandleStick

A candlestick chart is a combination of a line and bar chart, in that each bar represents the range of price movement over a given time interval.

Doughnut

A doughnut chart is functionally identical to a pie chart, except for a blank center and the ability to support multiple statistics as one.

Horizontal Bar

A horizontal bar graph is a chart that uses horizontal bars to show comparisons among categories. One axis of the chart shows the specific categories being compared, and the other axis represents a discrete value.

Horizontal Stacked Bar

A horizontal stacked bar chart stacks multiple data points in each bar on the chart instead of a single data point.

Line

A line chart displays information as a series of data points connected by straight line segments.

Pie

A pie chart is a circular chart divided into sectors, illustrating numerical proportion

Range

The range chart displays a range of data by plotting two Y values per data point, with each Y value being drawn as a line chart. The range between the Y values can then be filled with color.

Scatter

A scatter chart displays numerical values along the horizontal axis and the vertical axis, combining these values into single data points that are displayed in uneven intervals.

Spline

A spline chart is a specialized form of a line chart. Unlike conventional charts that connect data points with straight lines, a spline chart draws a fitted curve through the data points.

Spline Area

A spline area chart is a specialized form of an area chart. Unlike conventional charts that connect data points with straight lines, a spline chart draws a fitted curve through the data points.

Spline Range

The spline range chart displays a range of data by plotting two Y values per data point, with each Y value drawn as a line chart. The range between the Y values can be filled with color.

Stacked Area

The stacked area chart stacks two or more data series on top of one another.

Stacked Area 100

The stacked area 100 chart displays multiple series of data as stacked areas. The cumulative proportion of each stacked element is always 100% of the Y axis.

Stacked Bar

A stacked bar chart stacks multiple data points in each bar on the chart instead of a single data point.

Stacked Bar 100

A stacked bar 100 chart displays multiple data series as stacked bars where the cumulative proportion of each stacked element always totals 100%.

Stacked Line
A stacked line chart has lines that do not intersect because they are cumulative at each point.

Stacked Spline
A stacked spline chart has lines that do not intersect because they are cumulative at each point.

Stacked Spline Area
A stacked spline area chart is very similar to a spline chart. Data is displayed using different colors in the area below the line. Each series of points is represented with a different color.

Stacked Spline Area 100
The stacked spline area 100 chart is a variation of the spline area chart. The areas are stacked so that each series adjoins, but does not overlap the preceding series. This chart displays contributions for each data point to the category as a percentage that totals to 100%.

Stick
A stick chart is a combination of a line and bar chart.

Waterfall
A waterfall chart is a visualization of the sequence of positive and negative values that arrive at a final value.

Show Toggle Size Button
Select True to enable the toggle button and allow users to select the size of the component at runtime. Select False to hide the toggle button.

Show Legend
If set to True, a legend containing a list of the variables will display in the chart with an example of each appearance. This information allows the data from each variable to be identified in the chart.

Legend Title
The text to be displayed under the legend.

Legend Position
The location of the legend in relation to the chart. The options are Bottom, Left, Top, Right.

Chart X/Y Axis

Title
The title for the horizontal axis of the chart.

Label Rotation Angle
The angle to display the x-axis and y-axis labels.

Use Automatic Range
If set to True, it automatically determines the range of values to display on the x-axis and y-axis based on the values being shown in the graph. If set to False, it needs to be set manually in the following fields.

Minimum Value
Manually set the x-axis and y-axis starting value. This can only be set if Use Automatic Range is False.

Maximum Value
Manually set the x-axis and y-axis maximum value. This can only be set if Use Automatic Range is False.

Step
Manually set the change in values displayed on the x-axis and y-axis. This can only be set if Use Automatic Range is False.

Chart Data

Data Series Source Type
The data source for the chart.

Cube View
A Cube View from the chart's Data Adapter is being used as the data source.

One Column Multiple Rows

One column and multiple rows from the chart's Data Adapter are being used for the data source.

One Row Multiple Columns

One row and multiple columns from the chart's Data Adapter are being used for the data source.

Suppress Zeroes

If set to True, this removes all results that are zero.

Row List Type

This specifies the rows to include from the chart's Data Adapter.

All Rows

The chart will include all rows.

First Row

The chart will only include the first row.

Row Index List

The chart will include the list of rows specified in the Row Index List field.

Row Index List

Enter the list of rows to include in the chart.

Column List Type for X Axis

This specifies the columns from the Data Adapter to include in the chart's x-axis. When the Data Series Type is set to Cube View, all columns are included.

Column List

A specified list of columns used as x-axis types for the chart. These columns are entered in the Column List for X Axis field.

All Columns

All columns are used as x-axis members.

All Numeric Columns
All columns with numeric values are used as x-axis members.

Column List for X Axis
The column names used as x-axis members. This field is only used when Column List is selected as the Column List Type for X Axis.

Use Column Headers for X Axis
If set to True, the column names are used as x-axis members. If set to False, the values and text in the named column are used as x-axis members.

Column List Type for Data
This specifies the columns from the Data Adapter to include in the chart. When the Data Series Type is set to Cube View, all columns are included.

Column List
A specified list of columns used as Data Sources for the chart. These columns are entered in the Column List for Data field.

All Columns
All columns are used as Data Sources.

All Numeric Columns
All columns with numeric values are used as Data Sources.

Column List for Data
The column name containing the data to display in the chart. This field is only used when the Column List Type for Data is set to Column List.

Cube View Data Point Legend Type

Default
The legend text comes from the row headers.

Annotation
The legend text comes from the Annotation View Member in the Cube View instead of the default. The Data Adapter must include this information for this option.

Assumptions
The legend text comes from the Assumptions View Member in the Cube View instead of the default. The Data Adapter must include this information for this option.

Audit Comment
The legend text comes from the AuditComment View Member in the Cube View instead of the default. The Data Adapter must include this information for this option.

Footnote
The legend text comes from the Footnote View Member in the Cube View instead of the default. The Data Adapter must include this information for this option.

Variance Explanation
The legend text comes from the VarianceExplanation View Member in the Cube View instead of the default. The Data Adapter must include this information for this option.

Column for Legend Label
The legend text comes from the column name specified here.

Legend Labels List
Comma-separated strings used in the legend to identify the data series.

Chart (Advanced)

This component can be used with dashboards to display charts.

Formatting and Action

See Component Formatting and Action Properties.

Chart

Show Toggle Size Button

This controls whether the toggle  icon displays on the dashboard at runtime.

Diagram Type

Select a diagram type to determine how data displays in the chart. You can choose from the following options:

- **Polar 2D**: This is used to display data as a circular graph and displays values on the basis of angles.
- **Radar 2D**: This is used to display data as a circular graph and has multiple axes along which data can be plotted.
- **Simple 2D**: This is commonly used to compare percentage values of different point arguments in the same series.
- **XY2D**: This is commonly used to show filled areas on a diagram. Values can be used separately or aggregated.

> **NOTE:** The Chart properties do not apply to all diagram types.

Swap Axes

When set to True, this exchanges the X-axis with the Y-axis.

Domain Color

This is the chart's background color.

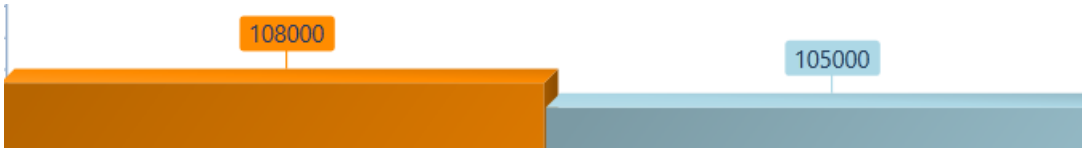
Show Point Labels

When set to True, this displays data values above each data series on the chart.

Point Label Text Format

Enter a Value Identifier within braces and, if needed, a colon followed by a Format String. The Value Identifier can be S for series name, V for value, or A for argument. Argument is typically used for the x-axis text format. Format Strings specify how to display the series name, value, or argument.

To display a value to two decimal places, enter **{V:0.00}** where 0.00 is the Format String. Enter **{V:0,,M}** to display the value in millions. Enter **{V:0,T}** to display the value in thousands. To apply a standard number format to a value, enter **{V:NumberFormat}**.



Apply a number format after the colon to display the value in a desired format. For example, enter **{V:#,###,0}**.



Any text written within the braces after the Format String will be displayed as text. Additionally, any text written before or after the braces will also be displayed as text.

Enable Animations

This controls how the chart renders at runtime. Set this to True to include chart animations when launching the dashboard.

Show Border

When set True, a border displays around the chart.

Use Clockwise Rotation

When set to True, this displays the legend in clockwise order.

Legend

Show Legend
When set to True, this creates and displays a customized legend for the dashboard chart.

Title
Enter the legend title.

Vertical/Horizontal Position
This controls where on the screen the legend displays.

Orientation
This determines whether the legend displays vertically or horizontally.

Show Check Boxes
When set to True, this creates a checkbox next to each legend item. Users can then select and deselect specific data points to display or hide them while viewing the dashboard chart.

Show Border
When set to True, a border displays around the legend.

Crosshair

Crosshair Enabled
When set to True, the crosshair function is enabled. A crosshair is similar to a point label, but it only displays when the user hovers over the data series.

Show Crosshair Lines
This identifies where the axis hits per data point.

Show Crosshair Labels
This displays the crosshair data in a dialog box.

Crosshair Label Mode
This determines where crosshair labels display. You can choose from the following options:

- **Show Common for All Series**: This displays the crosshair data for the entire chart no matter where the user is hovering.
- **Show for Each Series**: This displays crosshair data for each series simultaneously.
- **Show for Nearest Series**: This only displays crosshair data when the user is hovering on a specific series.

Crosshair Label Text Format
Enter a Value Identifier within braces and, if needed, a colon followed by a Format String. The Value Identifier can be S for series name, V for value, or A for argument. Argument is typically used for the x-axis text format. Format Strings specify how to display the series name, value, or argument.

Chart X/Y-Axis

Show X-Axis/Show Y-Axis
Set this to True to display the x-axis and y-axis. To turn off either axis, set the appropriate property to False.

Title
The title for each axis.

Argument Type
This specifies the type of information feeding into the chart's arguments. You can choose from the following options:

- **Date Time**: Use this if dates are included in the arguments. This causes the chart's x-axis to be continuous, which means the X-axis ticks are independent of the point's argument.
- **Double**: This causes the chart's x-axis to be continuous, which means the x-axis ticks are independent of the point's argument.

- **String**: This causes the chart to be discrete, which means the axis ticks are dependent on the point's arguments.

Text Format
Enter a Value Identifier within braces and, if needed, a colon followed by a Format String. The Value Identifier can be S for series name, V for value, or A for argument. Argument is typically used for the x-axis text format. Format Strings specify how to display the series name, value, or argument.

Label Rotation Angle
Enter a value to rotate the x-axis or y-axis labels. The default is no label rotation.

Logarithmic
When set to True, the axis values are forced to follow a logarithmic pattern specified in the Logarithmic Base property.

Logarithmic Base
Set the Logarithmic Base value. For example, logarithmic values with base 10 yields the axis values 0, 1, 10, 100, 1000, 10000, and so on.

Use Automatic Range
When set to True, this sets the minimum and maximum values automatically depending on the points contained in the chart. To customize the minimum/maximum value, set this to False and specify the values in the properties below.

Minimum/Maximum Value
Specify the minimum and maximum range values.

Use Automatic Step
Set this to True to create the tick marks by certain interval values automatically. Set this to False to specify the specific intervals for the tick marks.

Step
Specify the intervals for the tick marks.

Reverse Order
By default, the order of values on a chart begins from the bottom to the top, so the first color in the legend is the bottom value's color. Setting this property to True reverses that order. When the order is reversed, the values are displayed from top to bottom, following the order of the legend.

Interlaced
When set to True, the chart’s domain color alternates between the interlaced color. If this is set on the x-axis, the color displays horizontally. If this is set on the y-axis, the color displays vertically.

Interlaced Color
Enter the interlaced color.

Show Grid Lines
When set to True, this displays grid lines at each data point on the axis.

Show Minor Grid Lines
When set to True, this displays grid lines at each tick mark on the chart’s axis.

Series Properties

Type
Select a type. If your Diagram Type selection is Polar2D, the following Series Types display:

- **PolarArea**: The data series displays as a filled area on a circular diagram.
- **PolarLine**: The data series displays in a continuous line on a circular diagram.
- **PolarPoint**: The data series displays in a series of small circles on a circular diagram.

If your Diagram Type selection is Radar2D, the following Series Types display:

- **RadarArea**: The data series displays as a filled area on a circular grid with multiple axes along which data can be plotted.
- **RadarLine**: The data series displays as a line on a circular grid with multiple axes along which data can be plotted.
- **RadarPoint**: The data series displays as a series of small circles on a circular grid with multiple axes along which data can be plotted.

If your Diagram Type selection is Simple2D, the following Series Types display:

- **Funnel**: A funnel chart displays a wide area at the top, indicating the total points' value, while other areas are proportionally smaller.
- **NestedDonut**: This is similar to a donut chart, but it compares the data with one donut nested in another.
- **PieAndDonut**: This is used to compare the percentage values of different data points in the same series.

If your Diagram Type selection is XY2D, the following Series Types display:

- **Area**: An area chart or area graph displays graphically quantitative data, and it is based on a line chart. Area charts are used to represent cumulative totals using numbers or percentages over time.
- **AreaRange**: This displays the data series as filled areas on a diagram, with two data points that define minimum and maximum limits. This chart is used to illustrate the difference between start and end values.
- **BarRangeOverlappedWaterfall**: This displays either vertical or horizontal bars along the Y-axis (the axis of values). Each bar represents a range of data with two values. This chart type is used to show activity from different data series one above another to compare.

- **BarRangeSideBySideWaterfall**: This displays either vertical or horizontal bars along the Y-axis (the axis of values). Each bar represents a range of data with two values. This chart type is used to show activity from different data series grouped by their settings.
- **BarSideBySide**: This displays the data series as individual bars where the height of each bar is determined by the data value.
- **BarSideBySideFullStacked**: This can stack different bars and combine them into groups displayed side-by-side across the same axis value.
- **BarSideBySideStacked**: This can stack different bars and combine them into groups displayed side-by-side across the same axis value.
- **Bubble**: A bubble chart is a variation of a scatter chart in which the data points are replaced with bubbles, and an additional dimension of the data is represented in the size of the bubbles.
- **CandleStick**: A candlestick chart is a combination of a line and bar chart, in that each bar represents the range of price movement over a given time interval.
- **Line**: A line chart displays information as a series of data points connected by straight line segments.
- **LineScatter**: This is a type of line chart where the data points are connected by a continuous line.
- **LineStep**: This is used to display to what extent values have changed for different points in the same series.
- **Point**: This is used to show points from two or more different series on the same chart plot.
- **Spline**: A spline chart is a specialized form of a line chart. Unlike conventional charts that connect data points with straight lines, a spline chart draws a fitted curve through the data points.
- **Stock**: This is used to show variation in stock prices over the course of time.

- **Waterfall**: A waterfall chart is a visualization of the sequence of positive and negative values that arrive at a final value.

Model Display
Change the look of the series and determine which preset model the chart uses by choosing between these tiers: Basic, Moderate, and Advanced. Selecting Moderate or Advanced provides a more enhanced view of the chart.

Show Markers
This is used in conjunction with line charts to display a circular marker at each end of the line. When set to True, this displays line markers.

Marker Size
Enter a value for the marker size. Values are entered in pixels.

Bar Width
This is used in conjunction with bar charts. Enter a value to control the width of each bar on the chart.

Line Thickness
Enter a value measured in pixels to control the line thickness displayed in a line chart.

Line Style Type
This determines whether a line chart displays solid or dashed lines.

Pie Hole Radius Percent
This is used in conjunction with pie and donut charts and controls how large the middle hole is. A value of zero results in no hole.

Waterfall Series Properties

See Waterfall Series Type.

Series Colors

Series 1-10 Color
Select a color for each Data Series.

> **Example:** If a chart is displaying data for an Actual and Budget Scenario, data displayed for Actual is one data series and data displayed for Budget is a second data series. In this example, a color can be entered into Series 1 Color and Series 2 Color. By default, the order of values on a chart begins from the bottom to the top, so the first color in the legend is the bottom value's color. Use the Reverse Order property to change this order.

Series 1-10 Point Colors
Enter a color for the data points within a specific data series. If there are several data points and each one needs to be a different color, enter a comma-separated list of colors. If there are more data points in the series than specified colors, the data points cycle through the colors and repeat as needed.

Chart Data

Data Series Source Type
Select a source for the data series. You can choose from the following options:

- **Cube View**: A Cube View from the chart's data adapter is being used as the data source.
- **Business Rule**: A Dashboard Data Set business rule from the chart's data adapter is being used as the data source. Using a business rule gives more control to the chart's data series and provides additional customized settings.

Suppress Zeros
When set to True, this removes all results that are zero.

Row List Type
This specifies the rows to include from the chart's data adapter. You can choose from the following options:

- **All Rows**: The chart includes all rows.
- **First Row**: The chart only includes the first row.
- **Row Index List**: The chart includes the list of rows specified in the Row Index List field.

Row Index List
Enter the list of rows to include in the chart.

Cube View Data Point Legend Type
Select where the legend text comes from. You can choose from the following options:

- **Default**: The legend text comes from the row headers.
- **Annotation**: The legend text comes from the Annotation View Member in the Cube View instead of the default. The data adapter must include this information for this option.
- **Assumptions**: The legend text comes from the Assumptions View Member in the Cube View instead of the default. The data adapter must include this information for this option.
- **Audit Comment**: The legend text comes from the AuditComment View Member in the Cube View instead of the default. The data adapter must include this information for this option.
- **Footnote**: The legend text comes from the Footnote View Member in the Cube View instead of the default. The data adapter must include this information for this option.
- **Variance Explanation**: The legend text comes from the VarianceExplanation View Member in the Cube View instead of the default. The data adapter must include this information for this option.

Waterfall Series Type

A waterfall chart, also known as a walk or bridge chart, provides a visual story of the net changes of values between two identified points, such as starting and ending values in income statements, balance sheets, and operational expenses. You can build a waterfall chart with data supplied in a particular format from a data adapter. You can also convert existing BarRangeSideBySideWaterfall charts to the waterfall chart type.

Waterfall charts display net changes of a single value per bar. They provide positive variance values above the bar and negative variance values below the bar.



When you select the Waterfall type, the Waterfall Series Properties section provides configuration options for the chart:

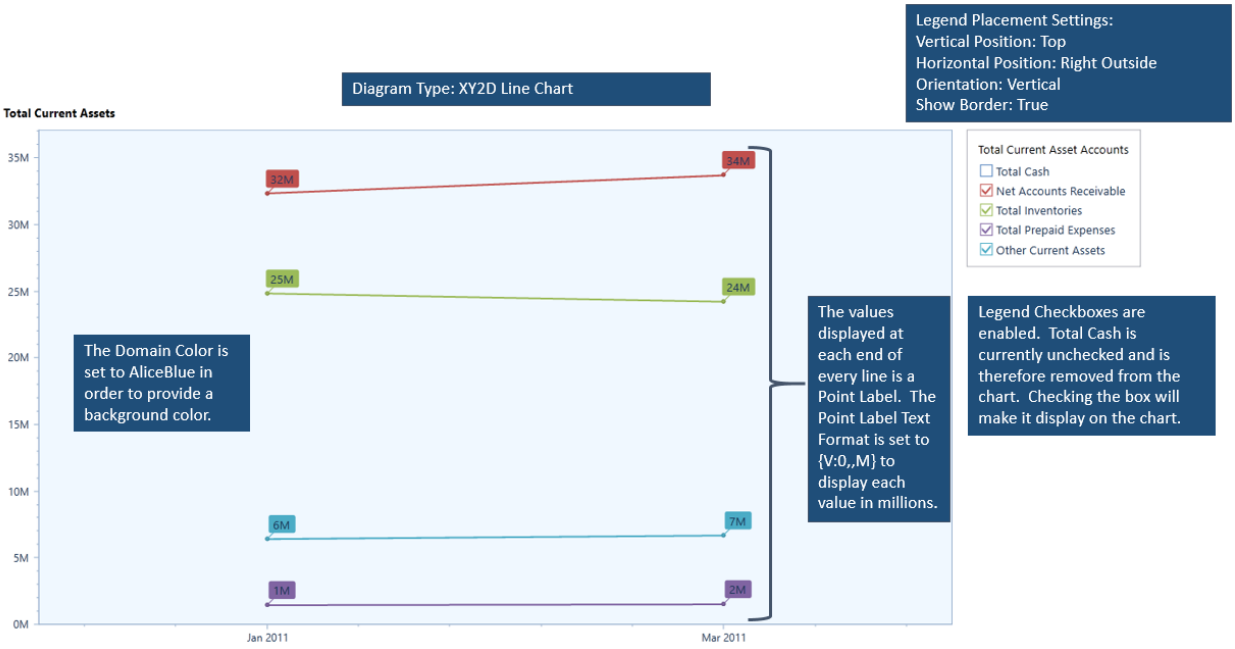
- **Start Bar Color**: This sets the color of the first bar in the series.
- **Total Bar Color**: This sets the color of the bar that represents a Total.
- **Rising Bar Color**: This sets the color of all bars whose values are positive, but are not Start or Total.

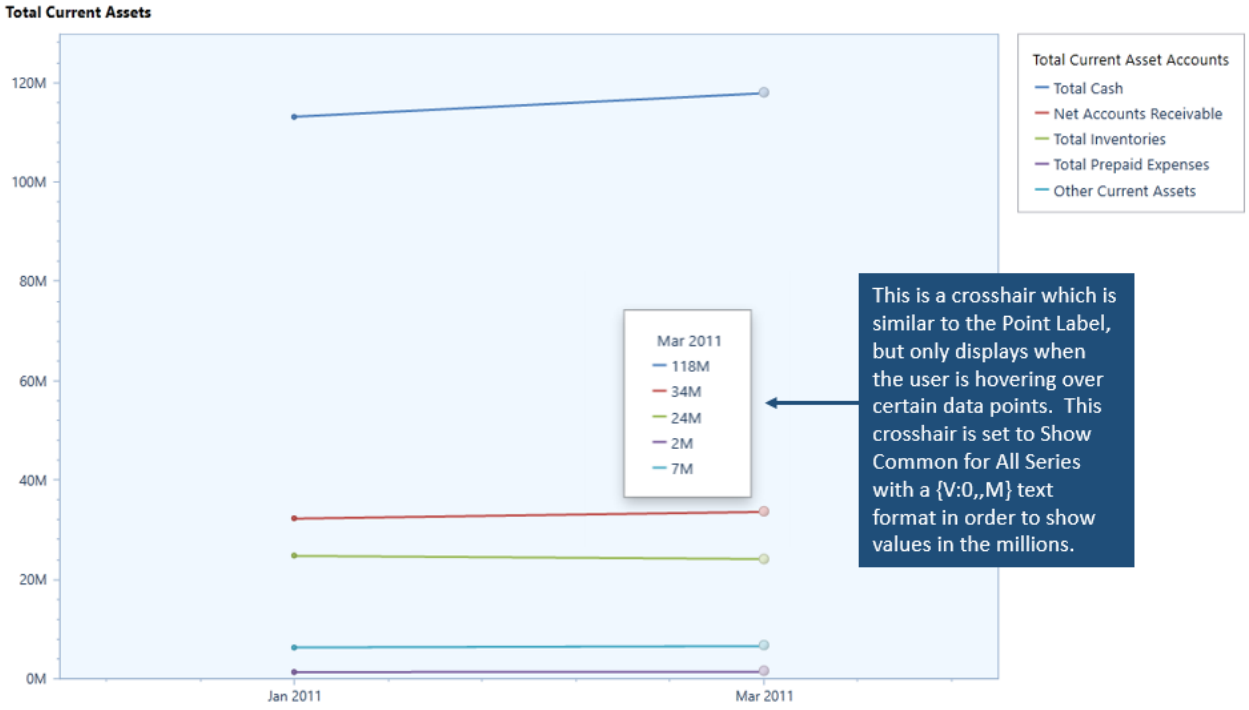
- **Failing Bar Color**: This sets the color of all bars whose values are negative, but are not Start or Total.
- **Total Included in Series**: This indicates if the data source includes a total value. When set to True, the chart identifies the last value in the data source as a Total bar on the chart. When set to False, the chart automatically calculates a sum and adds the result as a Total bar on the chart.
- **Total Value Label**: Sets the label of the Total bar.
- **Include Subtotals**: Indicates automatically calculated subtotals added to the waterfall chart. When set to True, you can indicate where you want to place the subtotal bars. When set to False, no subtotals are defined. The default value is False.
- **Subtotal Indexes**: Enter a comma-separated list of numbers to indicate where your subtotal bars will be placed. For example, to display a subtotal after your first delta value, enter 1. You can also enter 2,4, which would display a subtotal after the second delta value and the fourth delta value. These numbers count the delta values.
- **Subtotal Labels**: Enter a comma-separated list of labels for each subtotal. Each label needs to have unique text. The number of labels must equal the number of subtotal indexes specified. If nothing is listed, this property defaults to Subtotal 1, Subtotal 2, Subtotal 3, and so on.
- **Subtotal Bar Color**: This sets the color of bars that represent a subtotal. This property defaults to XFDarkBlueBackground.
- **Label Position**: This sets the position of the bar labels. The default is Auto. You can choose from the following options:
    - **Auto**: Labels are placed above the bar for rising values and below the bar for falling values.
    - **Center**: Labels are placed in the center of the bars.

- **InsideEnd**: Labels are placed on the inside bottom of the bar for falling values and on the inside top of the bar for rising values.
- **InsideStart**: Labels are placed on the inside bottom of the bar for rising values and on the inside top of the bar for falling values.

Advanced Chart Examples

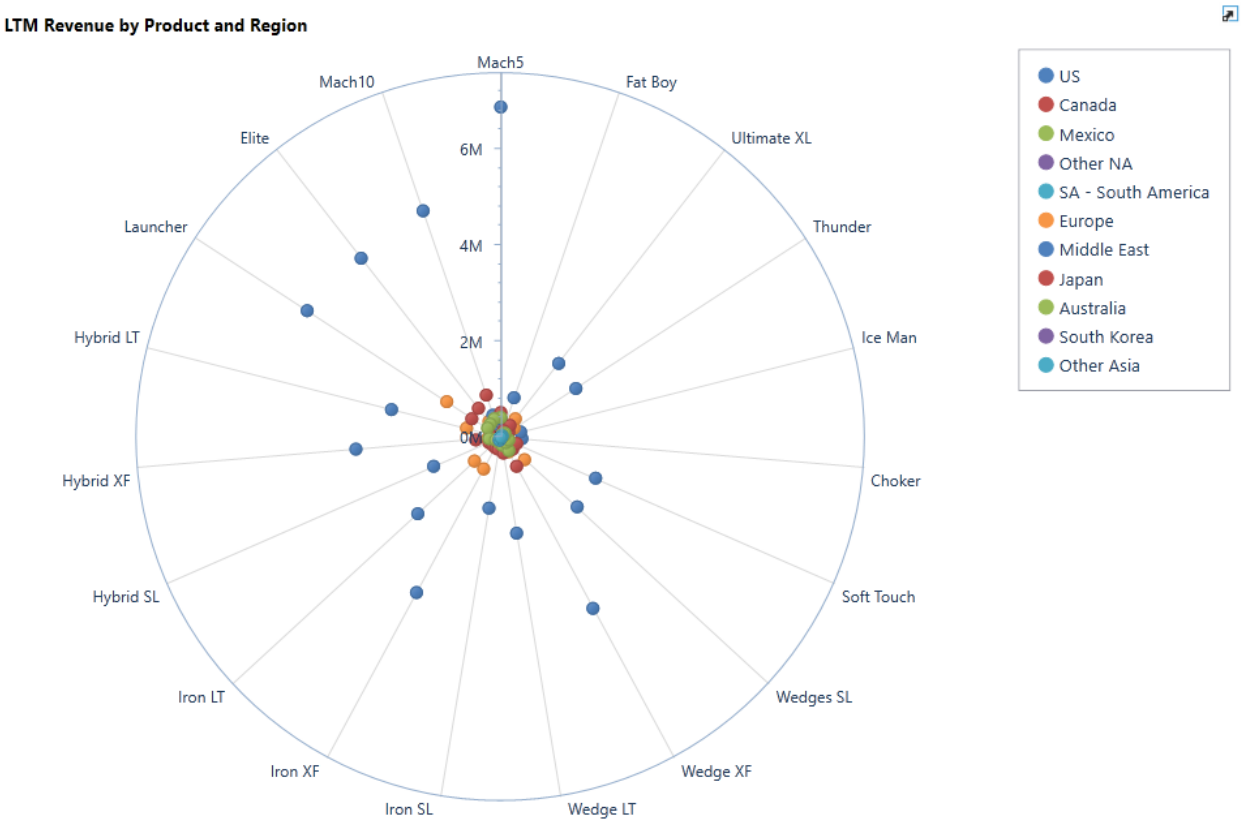
XY2D Line Chart Examples





Radar2D Radar Point Example

The following chart displays product revenue by region. This data is driven by a Cube View where the Cube View rows display the region and the Cube View columns display the products. Only the y-axis can be customized with this chart type, which is driven by the Cube View data. The legend is driven by the members on the Cube View rows, and the column members display multiple axes organized in a circle.



Check Box

This provides a small box that is filled with a check mark when selected.

Formatting and Action

See Component Formatting and Action Properties.

Combo Box

This component displays a drop-down list of strings.

Formatting and Action

See Component Formatting and Action Properties.

Cube View

Use this component to attach a Cube View to a dashboard.

Cube View

Select the ellipsis button and select the Cube View name.

Show Header
When set to True, the Cube View's Page Caption displays. When set to False, it is hidden.

Show Toggle Size Button
When set to True, the toggle button is enabled and users can select the component size at runtime. When set to False, the toggle button is hidden.

Actions
See Component Formatting and Action Properties.

> **NOTE:** For component actions to function, the Cube View must have Navigation Link parameters configured. Navigate to the Cube View and go to **General Settings** > **Navigation Links**. See Configuring Navigation Links for Linked Dashboards.

Data Explorer

This component is linked to a Cube View and displays in the standard data explorer view, which is seen when a Cube View is launched from the OnePlace tab. To use this component, specify a name and add a data adapter linked to a Cube View. See Data Adapters.

Data Explorer Report

This is a component linked to a Cube View and displayed in Report Viewer as if generated on the fly from a Data Explorer window. To use this, name the component and add a data adapter linked to a Cube View. See Data Adapters.

Date Selector

A Date Selector component enables you to set a date in a dashboard.

Date Selector

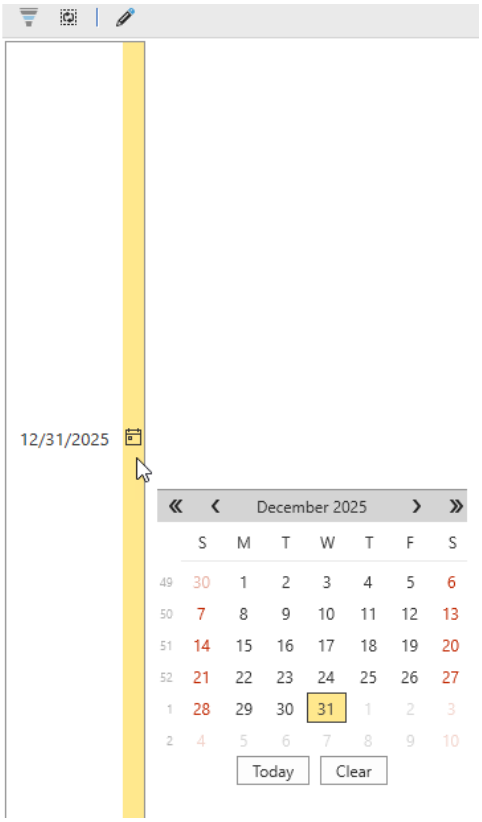
Min/Max Date
Set a minimum and maximum date range to be reviewed at runtime. Minimum and maximum date values should be entered in the following format: yyyyMMdd. For example, if you set the minimum date to 20202101 and the maximum date to 20201231, users can only select dates within that range. The value is converted at runtime based on your application culture settings.

> **NOTE:** If no minimum date is specified, the minimum date is set to 19000101. If no maximum date is specified, the maximum date is 99991231.

Select the ellipsis ... to choose an existing parameter. You can also enter and use a dashboard XFBR string.

Date Selector	
Min Date	|!DateSelectorParameter!|
Max Date	XFBR(XFBRStringExamples, GetEndDate)

Select the component to display the calendar window. Use the double arrows to show the minimum and maximum date range.



Formatting and Action

See Component Formatting and Action Properties.

Dynamic Grid

The Dynamic Grid component displays data in a tabular format from a Workspace assembly-driven data source. Each column represents a field, and each row represents a record. This grid enables you to read data from various sources, transform that data, and write it back to the grid (or leverage the Workspace assembly to direct data to another target if needed). This offers a flexible and high-performance feature for managing large datasets. It supports paging, dynamic columns, custom formatting, and other customizations. This component maintains the core functionality of the SQL Table Editor component while enhancing the flexibility of data presentation and end-user interaction.

The key features of the Dynamic Grid component include the following benefits:

- **Flexible data transformation**: Read, write, and transform data from various sources using Workspace assemblies.
- **High performance**: Manage large datasets efficiently with paging.
- **Dynamic columns**: Add, remove, and customize columns.
- **Custom formatting**: Apply custom formatting to data for better visualization.

This component is created in the Windows application and can be displayed in the Modern Browser Experience and the Windows application.

To use the Dynamic Grid component, follow these steps:

1. Review the Prerequisites.
2. Create a Dynamic Grid Component.
3. Update Configuration Settings.
4. Add the Dynamic Grid Component to a Dashboard.
5. Create a Workspace Assembly Dynamic Grid Service.

Review the Prerequisites

Before you create a Dynamic Grid, review these developer and setting prerequisites.

Developer

- Primarily, the Dynamic Grid will be created by a developer. Users will interact with the grid.
- The developer should have in-depth knowledge of OneStream Workspace assemblies, source data, and target data.

Settings

- Create a Workspace and maintenance unit. See "Create a Workspace" in the *Developer Guide*.
- Create a dashboard group and dashboard. See Dashboard Groups.

Create a Dynamic Grid Component

1. Go to **Application** > **Presentation** > **Workspaces**.
2. Select the node to the left of the Workspace to expand the navigation tree.
3. Select the to the left of the maintenance unit to expand the navigation tree.

   > **NOTE:** Do not select the Default maintenance unit if you have also selected the Default Workspace because you will not be able to add a dashboard component.

4. Select **Components**.
5. In the toolbar, select the **Create Dashboard Component**  icon.
6. In the **Create Dashboard Component** dialog box, select **Dynamic Grid** and then select the **OK** button.

7. Name the Dynamic Grid.

   > **NOTE:** The Dynamic Grid name must be unique within its Workspace.

8. Update the configuration settings. See Update Configuration Settings.

9. Select the **Save** icon.

Update Configuration Settings

Configure the available settings for the Dynamic Grid component.

Property	Description
General (Component)	
Name	Enter a name for the Dynamic Grid component. The name must be unique within its Workspace.
Description	Enter a description for the Dynamic Grid component. The description will display as the title text in the title header row.
Formatting	

Property	Description
Display Format	Select the ellipsis ... to select an option for how the colors display in the background. • **BackgroundColor**: Select this option to apply one background color to all rows in the grid. • **AlternateRowBackgroundColor**: Select this option to apply a specified color to every other row in the grid, creating a striped effect. This option can improve readability by visually distinguishing between adjacent rows. Display Format settings in the assembly code will override the option selected in this property.
Processing	
Template Parameter Values	Enter a comma-separated list of name-value pairs to specify Template Parameter Values. These values are used as the default settings for Template Parameters referenced using the ~!tParam!~ syntax related to Dynamic Workspace processing. For example, “tParam1=Value1, tParam2=Value2”.

Property	Description
Text 1	Enter arbitrary text that can be accessed by business rules or dynamic Workspace processing. For example, this property can be used to specify a parameter name or a functional name as a hint when implementing Selection Changed actions. See User Interface Action and Navigation Action in Component Formatting and Action Properties. It also supports template variables, for example, ~!tParam!~, for dynamic Workspace processing.
Text 2	Enter arbitrary text that can be accessed by business rules or dynamic Workspace processing. For example, this property can be used to specify a parameter name or a functional name as a hint when implementing Selection Changed actions. See User Interface Action and Navigation Action in Component Formatting and Action Properties. It also supports template variables, for example, ~!tParam1!~ for dynamic Workspace processing.
Action	
See Component Formatting and Action Properties.	
Dynamic Grid	

Property	Description
Filter Mode	Configure to filter data from column headings using options Popup and Filter Row. • **Popup**: Select to filter by selecting the funnel button in the column heading, which opens a popup window. This is the default. • **Filter Row**: Select to filter directly in the column heading. This setting is the same as the Filter Mode setting for the SQL Table Editor component. See SQL Table Editor Filtering.
Default For Columns Are Visible	Select True to display all columns in the grid (unless overridden at the individual column level). Select False to hide all columns in the grid (unless overridden at the individual column level).
Show Title Header	Configure to show (True) or hide (False) the title header row at the top of the grid. **NOTE:** If set to False, the Default For Allow Column Updates, Process Selection Changed For Inserts, Allow Inserts, and Allow Deletes options remain enabled.
Default For Allow Column Updates	Select True to allow the data in the existing table to be updated. Select False to turn off this option.

Property	Description
Show Title	Configure to show (True) or hide (False) the title text in the title header row.
Show Column Settings Button	Configure to show (True) or hide (False) the Column Settings button, which enables users to reorder the columns and set the column visibility.
Show Deselect All Button	Configure to show (True) or hide (False) the Deselect All button, which enables users to deselect the checkboxes for all selected rows.
Process Selection Changed For Inserts	Select True, and set the Allow Inserts option to True to allow the Insert button to process the selection changed event, which is often configured to refresh a portion of the dashboard. Select False to turn off this option.
Allow Inserts	Select True to allow rows of data to be added to the table. Select False to turn off this option.
Allow Deletes	Select True to allow rows of data to be deleted from the table. Select False to turn off this option.
Retain Table Column Order	Select True to allow the column order to be changed using the assembly service. Select False to turn off this option.

Property	Description
Read Only Text Color	Select (Use Default) to set the default text color to standard black text. Update to use any text color, including a system color, for example, XFReadOnlyText.
Column Name for Bound Parameter	Enter the name of the column to use to change the value of the bound parameter when a database row is selected. This is used when a Dynamic Grid is being used to affect the display of other dashboard components, for example, when showing detailed information for the selected row.
Default For Show Search	Select True to enable search functionality within the drop-down menu for all columns that are bound to a List parameter, unless specified otherwise for individual columns. This setting is the same as the Default For Show Search setting for the SQL Table Editor Component. See SQL Table Editor Searchable Drop-Down Menu. Select False to turn off this option.

Property	Description
Allow Multiselect	Select True to allow checkboxes to be selected for multiple rows. The active selected items will be passed to the defined Bound parameter field as a comma-delimited list. The Bound parameter format will be: Item1, Item2, Item3. If three values, A, B, C,D, are selected (where C,D is one value), the resulting Bound parameter string is: A, B, "C,D". This setting is the same as the Allow Multiselect setting for the SQL Table Editor component. See SQL Table Editor Multiselect. Select False to turn off this option and only pass a single Bound parameter at a time. The checkboxes will not display in the user interface. **IMPORTANT:** To use multiselect in the Dynamic Grid, ensure the XFDataTable returned from the Dynamic Grid service has at least one primary key column. Especially when retrieving data from multiple sources, ensure each primary key column is set. See "Dynamic Grid Service Type" in the *Developer Guide*.
Rows Per Page	Enter a numeric value between 1 and 3000 to specify the number of rows to display before a page break. The default is -1, which turns off the property and uses your defined user preference property Grid Rows Per Page, which is located in the Security settings on the user profile. If you enter a value greater than 3000, the maximum number of results per page is still 3000.
Save State Settings	

Property	Description
Save State	Select True to retain the user settings on the component. User preferences for columns saved will be for Order, Visibility, Filtering, Freeze Bar, Sorting, and Widths. This setting is the same as the Save State setting for the SQL Table Editor component. See SQL Table Editor Save State User Preferences. Select False to turn off this option. **NOTE:** To reset the Save State to default, right-click on the dashboard to enable the Reset State to return to its Component Properties settings.
Vary Save State By	Select True to save the Save State settings by Workflow Profile and Workflow Scenario. The related dashboard component will have the option Reset All States, which can be used to clear the Save States for a user across all the Vary Save State By parameters. Select False or (Use Default) to turn off this option.
Data Manipulation Buttons	
Show Data Manipulation Buttons	Configure to show (True) or hide (False) the Add Row, Remove Row, Cancel, and Save buttons.
Show Add Row Button	Configure to show (True) or hide (False) the Add Row button.

Property	Description
Show Remove Row Button	Configure to show (True) or hide (False) the Remove Row button.
Show Cancel Button	Configure to show (True) or hide (False) the Cancel button.
Show Save Button	Configure to show (True) or hide (False) the Save button.

Add the Dynamic Grid Component to a Dashboard

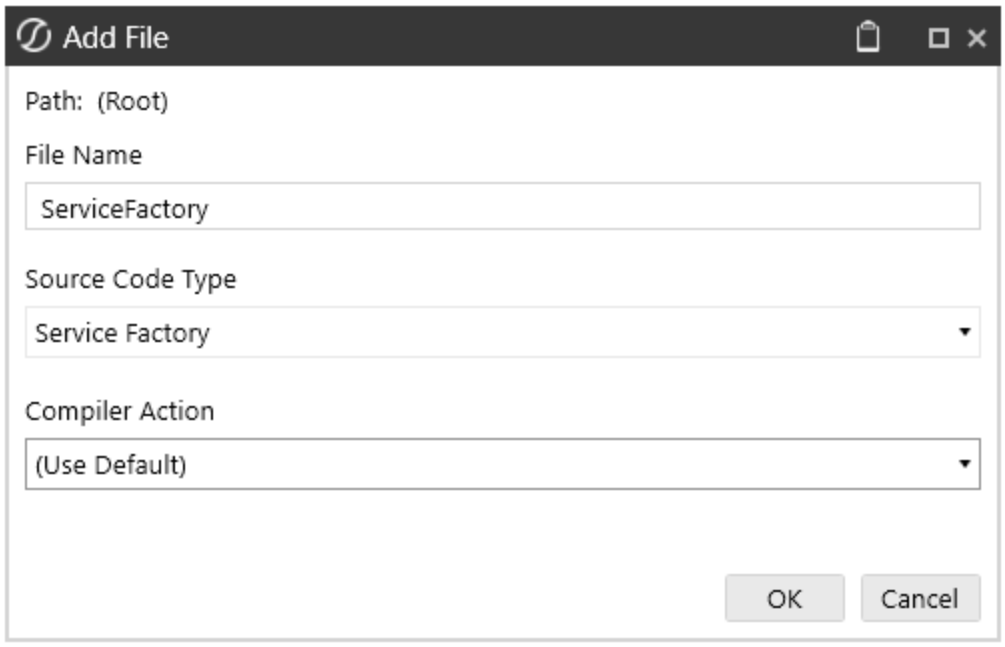
1. In the same maintenance unit as the Dynamic Grid component, select the node to the left of **Dashboard Groups** to expand the navigation tree.
2. Select a dashboard.
3. Select the **Dashboard Components** tab.
4. In the toolbar, select the **Add Dashboard Component** icon.
5. In the **Add Dashboard Component** dialog box, select the Dynamic Grid component and then select the **OK** button.
6. Select the **Save** icon.

Create a Workspace Assembly Dynamic Grid Service

1. In the same maintenance unit as the Dynamic Grid component, select **Assemblies**.
2. In the toolbar, select the **Create Assembly**  icon.
3. On the **Assembly Properties** tab, in the **Name** field, name the Assembly.

   > **NOTE:** The Assembly name must be unique within its Workspace.

4. In the **Compiler Language** drop-down menu, select a language.
5. In the toolbar, select the **Save**  icon.
6. Create files for the service factory and the Dynamic Grid service.
   a. On the **Assemblies Files** tab, right-click on **Files** and select **Add File**. In the **Add File** dialog box:
      i. In the **File Name** field, enter a name for the file, for example, ServiceFactory.
      ii. In the **Source Code Type** drop-down menu, select **Service Factory**.
      iii. In the **Compiler Action** drop-down menu, select **(Use Default)**.
      iv. Select the **OK** button.



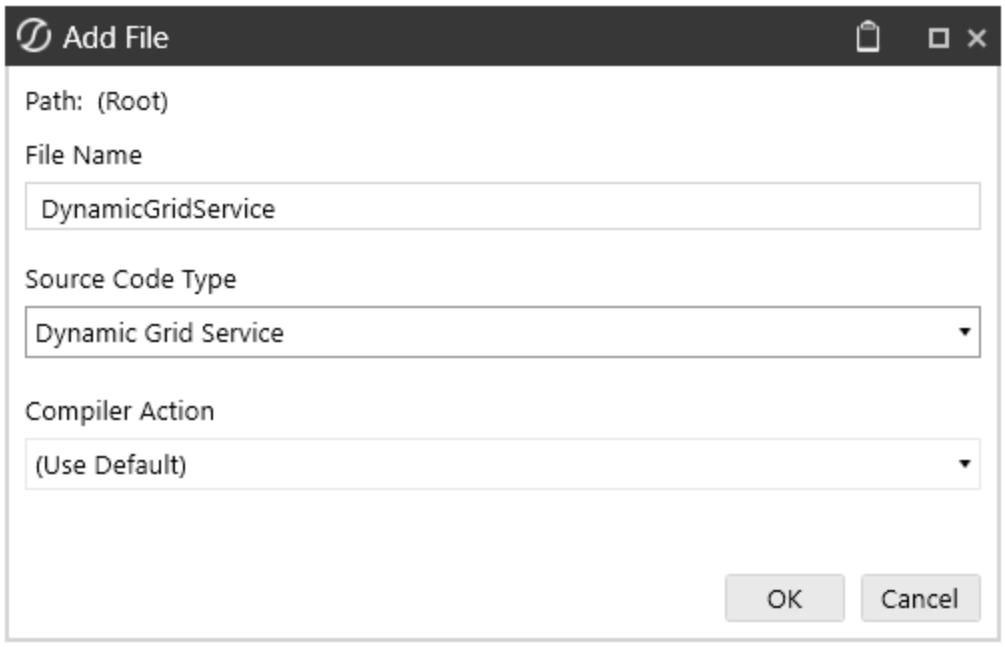
b. On the **Assemblies Files** tab, right-click on **Files** and select **Add File**. In the **Add File** dialog box:

   i. In the **File Name** field, enter a name for the file, for example, DynamicGridService.

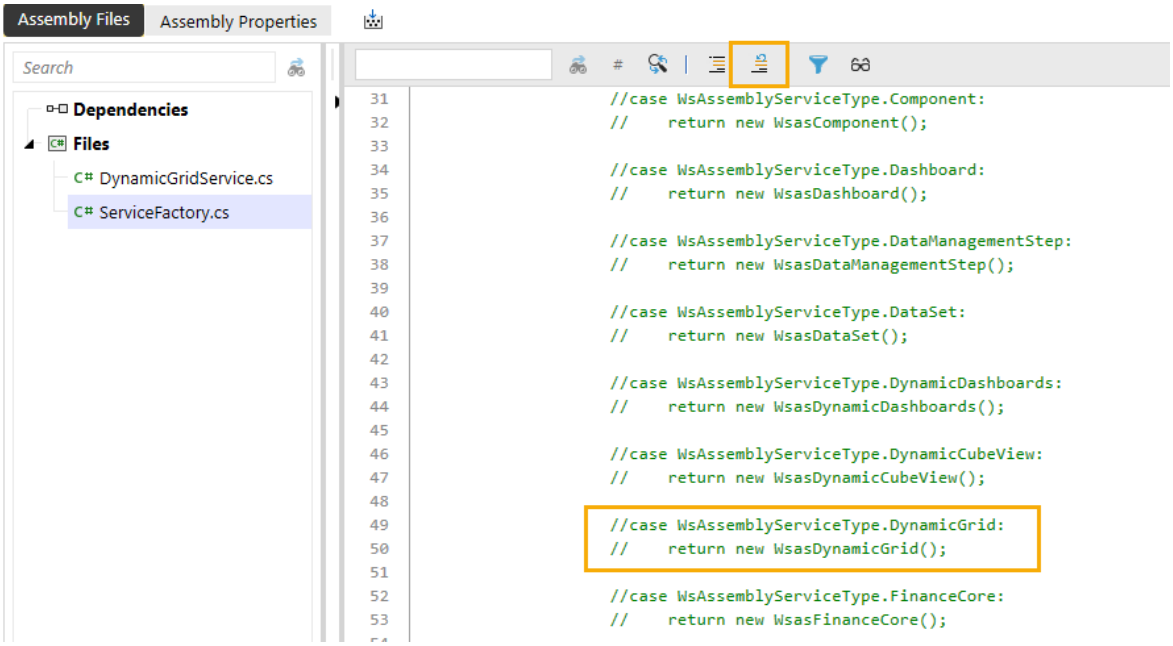
   ii. In the **Source Code Type** drop-down menu, select **Dynamic Grid Service**.

   iii. In the **Compiler Action** drop-down menu, select **(Use Default)**.

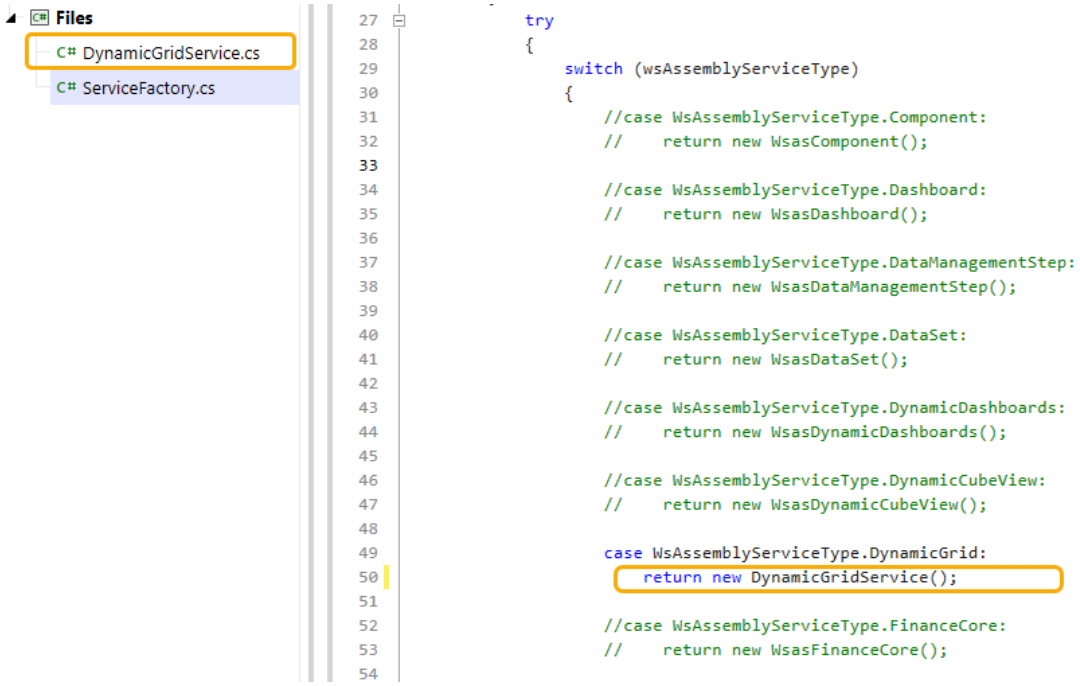
   iv. Select the **OK** button.



7. By default, Dynamic Grid methods for Get and Save are included. See "Dynamic Grid Service Type" in the *Developer Guide*. Modify the code as needed.
8. To enable the Dynamic Grid, in the **Service Factory** assembly file, highlight the lines for the Dynamic Grid, and select the **Uncomment the selected lines** icon.



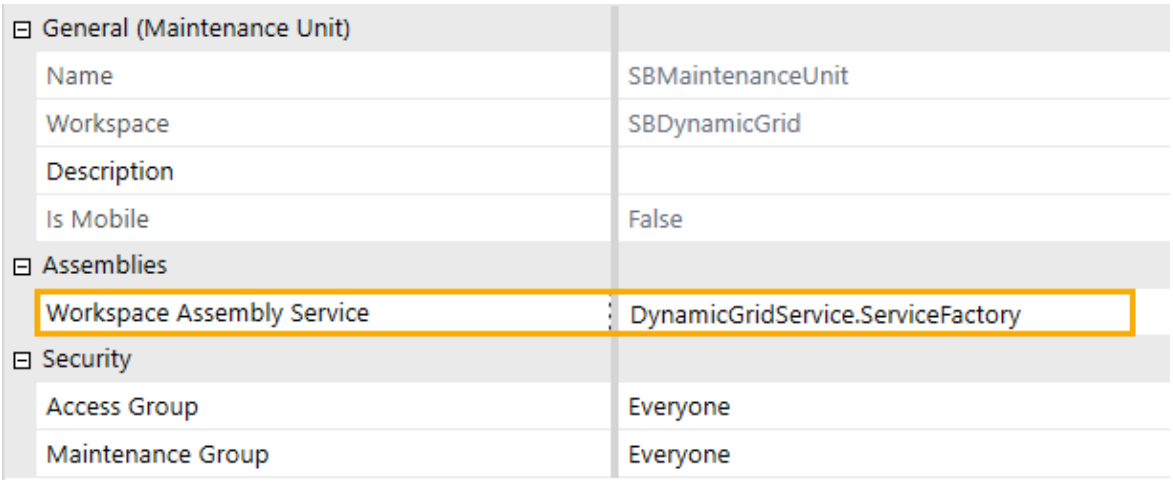
NOTE: Ensure the object name matches the name of the Dynamic Grid service file.



9. In the toolbar, select the **Save** icon.

10. Select the maintenance unit. In the **Workspace Assembly Service** field, enter the name of the assembly and the service factory assembly file in this format: <Assemblyname>.<ServiceFactoryAssemblyFilename>

> **Example:** If the assembly name is DynamicGridService and the service factory assembly file name is ServiceFactory, the Workspace assembly service would be: DynamicGridService.ServiceFactory



11. In the toolbar, select the **Save** icon.

12. Select the assembly, and select the **Compile Assembly to check syntax** icon to ensure the Workspace assembly files compile.

Row and Cell Formatting

Designers can set the column, row, and cell formatting of a Dynamic Grid through the Workspace assembly to enhance the visual presentation of data. Cell formatting takes precedence over row formatting, and row formatting takes precedence over column formatting. Also, grid settings and properties defined in the Workspace assembly take precedence over settings defined in the component property grid.

The following grid formatting properties are available for use in the Workspace assembly. The following example shows how to configure a column definition object using these properties.

```
'       .CustomParameters = String.Empty, -----> This is looking for a literal parameter BRApi.Dashboards.Parameters.GetLiteralParameterValue(si, isSystemLevel, workspaceIDIfNotInName, combinedWsAndParamName)
'       .ColumnName = String.Empty, -----------> This is looking for a string name of a column: Example: "Name here"
'       .Description = String.Empty, ----------> This is looking for a string description of a column: Example: "Description here"
'       .ColumnDisplayType = columnDisplayType > This will look for one of three items XFGridColumnDisplayType.Standard, XFGridColumnDisplayType.OneStreamClientImage or XFGridColumnDisplayType.WorkflowStatusImage
'       .IsVisible = True, ---------------------> This is a true or false setting
'       .AllowUpdates = True,------------------> This is a true or false setting
'       .IsFromTable = True, -------------------> This is a true or false setting
'       .FilterByColumnName = String.Empty, ---> You can base a column's header filter on values from another column. This includes any column that is in the data set, even if the column is hidden. This is useful for name-v
'       .ParameterName = String.Empty, --------> This is looking for a literal parameter BRApi.Dashboards.Parameters.GetLiteralParameterValue(si, isSystemLevel, workspaceIDIfNotInName, combinedWsAndParamName)
'       .ShowSearch = True, --------------------> This is a true or false setting
'       .BackgroundColor = XFColors.AliceBlue, > This requires the XFColors.Color and you can replace the color with a color found inside of OneStream
'       .DefaultValue = String.Empty, ---------> The default value to be entered when allowing new records to be added. This helps ensure that invalid blank cells are not created.
'       .IsMultilineText = String.Empty, ------> This is a true or false setting
'       .DataFormatString = String.Empty, -----> This is a string setting for the number format, if you need a list of formats available you can click the object look up (glasses) and then look at the number formats
'       .Width = 300 ---------------------------> This setting determines the width of the column via pixel count
```

VB.Net

```
'.CustomParameters = String.Empty, -----> This is looking for a literal parameter
BRApi.Dashboards.Parameters.GetLiteralParameterValue(si, isSystemLevel,
workspaceIDIfNotInName, combinedWsAndParamName)
'.ColumnName = String.Empty, -----------> This is looking for a string name of a column:
Example: "Name here"
'.Description = String.Empty, ----------> This is looking for a string description of a
column: Example: "Description here"
'.ColumnDisplayType = columnDisplayType > This will look for one of three items
XFGridColumnDisplayType.Standard, XFGridColumnDisplayType.OneStreamClientImage or
XFGridColumnDisplayType.WorkflowStatusImage
'.IsVisible = True, ---------------------> This is a true or false setting
'.AllowUpdates = True,------------------> This is a true or false setting
'.IsFromTable = True, -------------------> This is a true or false setting
'.FilterByColumnName = String.Empty, ---> You can base a column's header filter on values
from another column. This includes any column that is in the data set, even if the column is
hidden. This is useful for name-value pair parameter columns as long as the values and names
are pulled from individual columns.
'.ParameterName = String.Empty, --------> This is looking for a literal parameter
BRApi.Dashboards.Parameters.GetLiteralParameterValue(si, isSystemLevel,
workspaceIDIfNotInName, combinedWsAndParamName)
'.ShowSearch = True, --------------------> This is a true or false setting
'.BackgroundColor = XFColors.AliceBlue, > This requires the XFColors.Color and you can
replace the color with a color found inside of OneStream
'.DefaultValue = String.Empty, ---------> The default value to be entered when allowing new
records to be added. This helps ensure that invalid blank cells are not created.
'.IsMultilineText = String.Empty, ------> This is a true or false setting
'.DataFormatString = String.Empty, -----> This is a string setting for the number format, if
you need a list of formats available you can click the object look up (glasses) and then
look at the number formats
'.Width = 300 ---------------------------> This setting determines the width of the column
via pixel count ------------------> This setting determines the width of the column via
pixel count
```

```
Dim columnDefinition1 As New XFDynamicGridColumnDefinition() With {
    .CustomParameters = String.Empty,
    .ColumnName = "id",
    .Description = "NameChangeID",
    .ColumnDisplayType = XFGridColumnDisplayType.Standard,
    .IsVisible = True,
    .AllowUpdates = True,
    .IsFromTable = True,
    .FilterByColumnName = String.Empty,
    .ParameterName = String.Empty,
    .ShowSearch = True,
    .BackgroundColor = XFColors.Red,
    .DefaultValue = String.Empty,
    .IsMultilineText = True,
    .DataFormatString = String.Empty,
    .Width = 100
}
```

VB.Net

```
Dim columnDefinition1 As New XFDynamicGridColumnDefinition() With {
        .CustomParameters = String.Empty,
        .ColumnName = "id",
        .Description = "NameChangeID",
        .ColumnDisplayType = XFGridColumnDisplayType.Standard,
        .IsVisible = True,
        .AllowUpdates = True,
        .IsFromTable = True,
        .FilterByColumnName = String.Empty,
        .ParameterName = String.Empty,
        .ShowSearch = True,
        .BackgroundColor = XFColors.Red,
        .DefaultValue = String.Empty,
        .IsMultilineText = True,
        .DataFormatString = String.Empty,
        .Width = 100
    }
```

The following example displays row and cell formatting in the Workspace assembly. Details are also provided.

C#

```
private List<XFDynamicGridRowFormat> GetRowFormats(XFDataRowCollection rows,
XFDataColumnCollection columns, bool tableHasPrimaryKeyColumns
{
    List<XFDynamicGridRowFormat> rowFormats = new List<XFDynamicGridRowFormat>();

    XFDataRow dataRowToFormat = rows[2];

    // Create the row format.
    XFDynamicGridRowFormat rowFormat = new XFDynamicGridRowFormat();
    //rowFormat. RowHash = XFDynamicGridRowFormatFlelper.CalculateRowHash(dataRowToFormat,
columns, tableHasPrimaryKeyColumns);
    rowFormat.IsEditable = TriStateBool.TrueValue;
    rowFormat.BackgroundColor = XFColors .Green;
    rowFormat.Italic = TriStateBool .TrueValue;
    rowFormat.Bold = TriStateBool.TrueValue;


    // Create the link between the row to format and the XFDynamicGridRowFormat object.
    Guid rowID = Guid.NewGuid();
    rowFormat.RowID = rowID;
    dataRowToFormat.UniquelDForRowFormat = rowID;

    // Create cell formats.
    XFDynamicGridRowCellFormat cellFormatl = new XFDynamicGridRowCellFormat();
    cellFormatl.ColumnName = "amount";
    cellFormatl.BackgroundColor = XFColors.Red;
    cellFormatl.IsEditable = TriStateBool.FalseValue;
    cellFormatl.Bold = TriStateBool.FalseValue;

    XFDynamicGridRowCellFormat cellFormat2 = new XFDynamicGridRowCellFormat();
    cellFormat2.ColumnName = "location";
    cellFormat2.BackgroundColor = XFColors.Red;
    cellFormat2.IsEditable = TriStateBool.FalseValue;
    cellFormat2.Bold = TriStateBool.FalseValue;

    rowFormat.CellFormats.Add(cellFormatl);
    rowFormat.CellFormats.Add(cellFormat2);
}
```

```
private List<XFDynamicGridRowFormat> GetRowFormats(XFDataRowCollection rows, XFDataColumnCollection columns, bool tableHasPrimaryKeyColumns
{
    List<XFDynamicGridRowFormat> rowFormats = new List<XFDynamicGridRowFormat>();

1   XFDataRow dataRowToFormat = rows[2];

    // Create the row format.
2   XFDynamicGridRowFormat rowFormat = new XFDynamicGridRowFormat();
    //rowFormat.RowHash = XFDynamicGridRowFormatHelper.CalculateRowHash(dataRowToFormat, columns, tableHasPrimaryKeyColumns);
    rowFormat.IsEditable = TriStateBool.TrueValue;
3   rowFormat.BackgroundColor = XFColors.Green;
    rowFormat.Italic = TriStateBool.TrueValue;
    rowFormat.Bold = TriStateBool.TrueValue;

    // Create the link between the row to format and the XFDynamicGridRowFormat object.
4   Guid rowID = Guid.NewGuid();
    rowFormat.RowID = rowID;
    dataRowToFormat.UniqueIDForRowFormat = rowID;

    // Create cell formats.
    XFDynamicGridRowCellFormat cellFormat1 = new XFDynamicGridRowCellFormat();
5   cellFormat1.ColumnName = "amount";
    cellFormat1.BackgroundColor = XFColors.Red;
    cellFormat1.IsEditable = TriStateBool.FalseValue;
    cellFormat1.Bold = TriStateBool.FalseValue;

    XFDynamicGridRowCellFormat cellFormat2 = new XFDynamicGridRowCellFormat();
    cellFormat2.ColumnName = "location";
    cellFormat2.BackgroundColor = XFColors.Red;
    cellFormat2.IsEditable = TriStateBool.FalseValue;
    cellFormat2.Bold = TriStateBool.FalseValue;

    rowFormat.CellFormats.Add(cellFormat1);
    rowFormat.CellFormats.Add(cellFormat2);
```

1. Create an XFDataRow object with a row, using its specific row number.
2. Create an XFDynamicGridRowFormat object.
3. Enter display formatting options through code by specifying a value for a format property of the object.
4. Create a Guid and apply as a value to the following properties:
    a. .RowID (XFDynamicGridRowFormat object)
    b. UniqueIDForRowFormat (XFDataRow object)
5. Create an XFDynamicGridRowCellFormat object to format a specific cell or cells in the row. Cell formats can be added to the RowFormat object to be applied to specific cells based on column name.

Export the Dynamic Grid

Export data from the Dynamic Grid to create Excel files that can be used outside of OneStream. The display formatting configured for the Dynamic Grid also applies to any exported data.

To export, complete the following steps:

1. In the Dynamic Grid toolbar, select the context menu button.
2. In the context menu, select **Export**.
3. In the **Export to Excel** dialog box, select a page range. You can select: **CurrentPage**, **AllPages**, or **Custom**. If you select **Custom**, enter a page range of two numbers separated by a hyphen in the following text field.
4. (Optional) Select the **Apply Current Filters** checkbox to include matching rows based on the filters that are currently applied to data in the Dynamic Grid.
5. (Optional) Select the **Apply Current Sorting** checkbox to sort the exported data based on sorting currently applied to data in the Dynamic Grid.
6. Select the checkbox next to eachcolumn to export.
7. Select the **OK** button.

Embedded Dashboard

When a dashboard is created in the maintenance unit, an Embedded Dashboard component is also created. Embedded Dashboards are used for sharing across dashboard groups in the same maintenance unit.

Embedded Custom Control Dashboard

A parent dashboard can contain multiple instances of the same custom control. Independent input parameters are mapped to each instance, while each Embedded Dashboard component monitors for events from the custom control.

> **NOTE:** The parent dashboard's Dashboard Type property must be set to Custom Control.

Instance Name
An optional name that uniquely differentiates this instance of an Embedded Custom Control dashboard from other instances. This instance name is used when saving state information related to interactions with the dashboard when using multiple instances of the same Custom Control dashboard.

Input Parameter Values
A comma-separated list of name-value pairs for the Embedded Custom Control dashboard initial parameter values.

> **Example:** "Paraml=Value1, Param2=(Value2], Param3=|!AnotherParamName!|."

Event Listeners
Actions to take when a component inside the referenced Embedded Custom Control dashboard uses its navigation action settings to forward a selection changed event to the parent dashboard. This requires setting the Navigation Action appropriately in the component itself. See Custom Control Dashboard Type.

File Viewer

This component is used to present files, such as PDFs stored in the OneStream File Share.

There are the only situations where users can see these files:

- They must have the ManageFileShare system security role.
- The files must be stored in a folder under the Incoming folder in File Share.

  > **Example:**
  > \\MyServerName\OneStreamShare\Applications\MyApplicationName\Incoming\MyDashboardFiles.

- The file is placed in one of the folders under \\MyServerName\OneStreamShare\Applications\MyApplicationName\Groups and then in a named folder with the same name as a user group. The file must be in the folder with same name as the user group and the user must have permissions to that group.

File Source Type
Select a source type for the file:

- **URL**: Display a file from an internal or external Web page.
- **Workspace File**: Display a file stored in a Workspace maintenance unit File section.
- **Application Database File**: Display a file stored in the Application Database Share.
- **System Database File**: Display a file stored in the System Database Share.
- **File Share File**: Display a file from the File Share.

URL Or Full File Name

The URL or name of the file being used. Enter the full URL name or select the ellipsis ... and browse to the desired file.

Show Header
When set to True, a header derived from the component's name or description is displayed.

Show Toggle Size Button
When set to True, the toggle button is enabled and users can toggle the size of the component at runtime. When set to False, the toggle button is hidden.

Show PDF Toolbar
When set to True, the PDF toolbar displays while viewing the component on a dashboard. When set to False, the toolbar is hidden.

Auto Play (audio and video files)
When set to True, the video and audio files shown in the component will automatically start.

Process Extensible Document
When set to True, the File Viewer component runs and processes the attached Extensible Document. When set to False, the unprocessed file displays, which is mainly used for testing purposes.

> **NOTE:** An Extensible Document is a Text, Word, PowerPoint, or Excel file that uses parameters in its content. The file name must contain .xfDoc before the extension, for example, StatusReport.xfDoc.docx. See Presenting Data With Extensible Documents.

Filter Editor

The Filter Editor component provides an efficient method for applying filters to external data sources defined by a business rule through a data adapter.

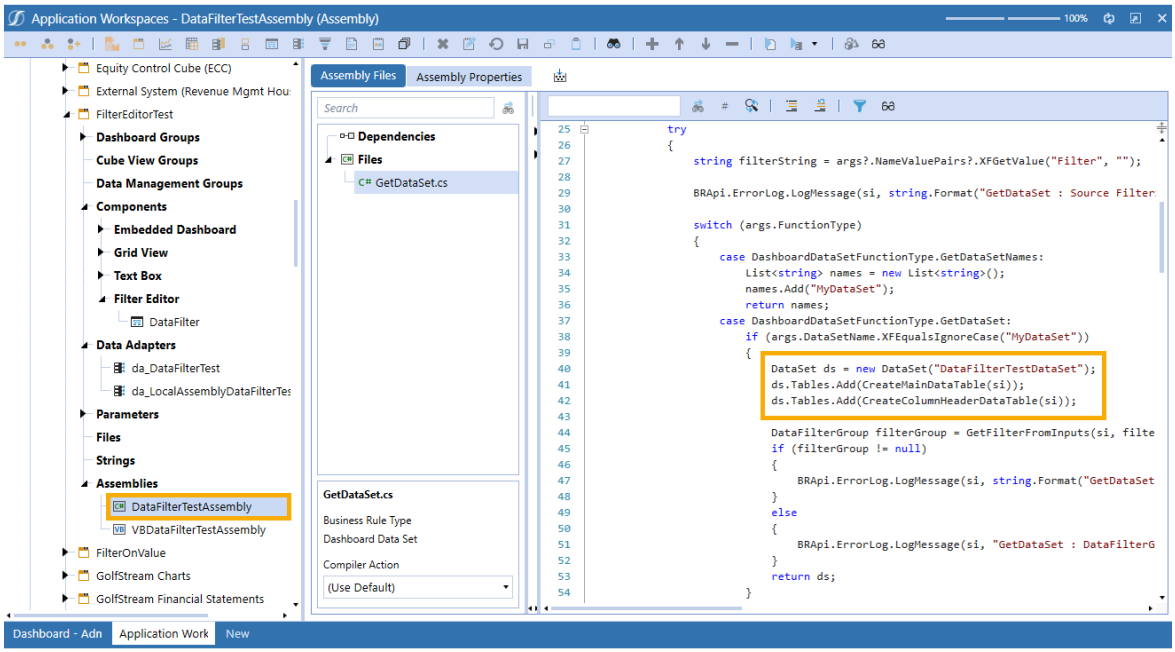
Data Adapter

A data adapter is required to gather information for the Filter Editor. Column names and data types are pulled from the first data table. A second data table can be created if the column display names must be altered.

> **NOTE:** The second data table must contain the same number of columns in the same order as the first table. Only column definitions, not data, need to be in the second table.

Follow these steps to create a data adapter for Filter Editor:

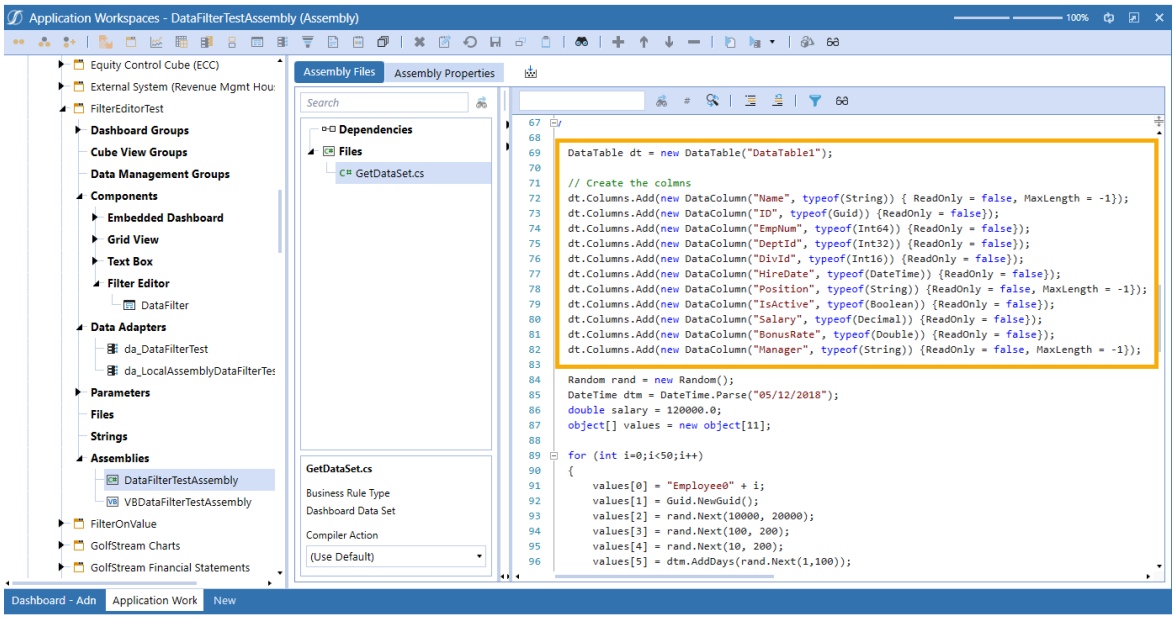
1. Define the Data Table using a business rule.



C#

```
case DashboardDataSetFunctionType.GetDataSetNames:
    List<string> names = new List<string>();
    names.Add("MyDataSet") ;
    return names;
case DashboardDataSetFunctionType.GetDataSet:
    if (args.DataSetName.XFEqualsIgnoreCase("MyDataSet"))
    {
        DataSet ds = new DataSet("DataFilterTestDataSet");
        ds.Tables.Add(CreateMainDataTable(si));
        ds.Tables.Add(CreateColumnHeaderDataTable(si)) ;
```

2. (Optional) To alter the Filter Editor column names for display, create a second Data Table and define new column names.

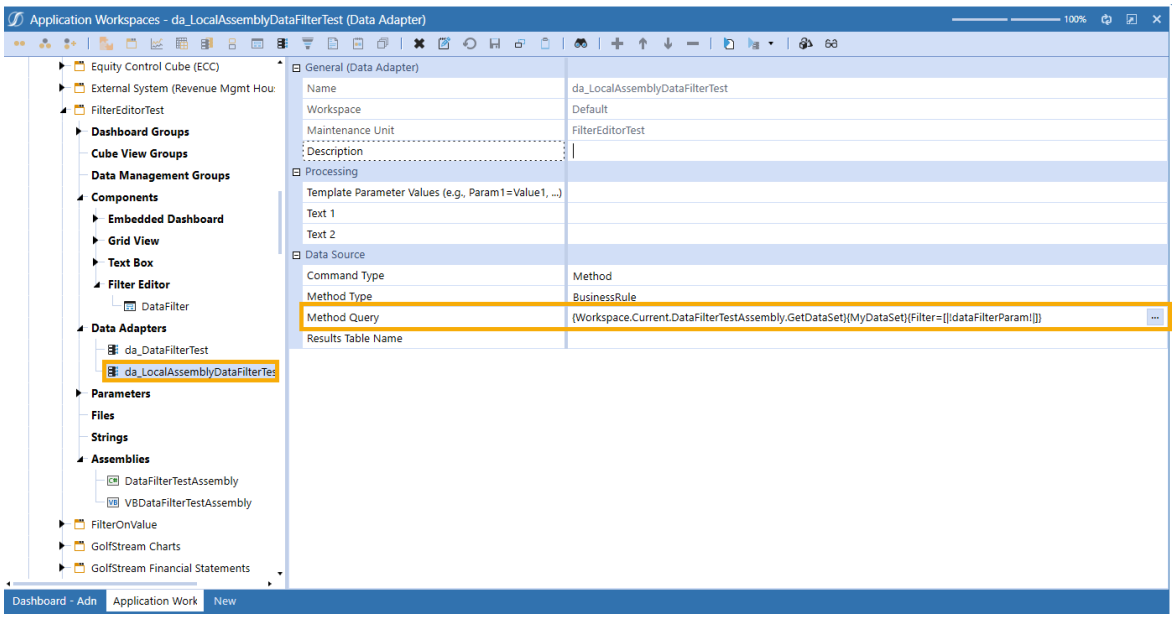


C#

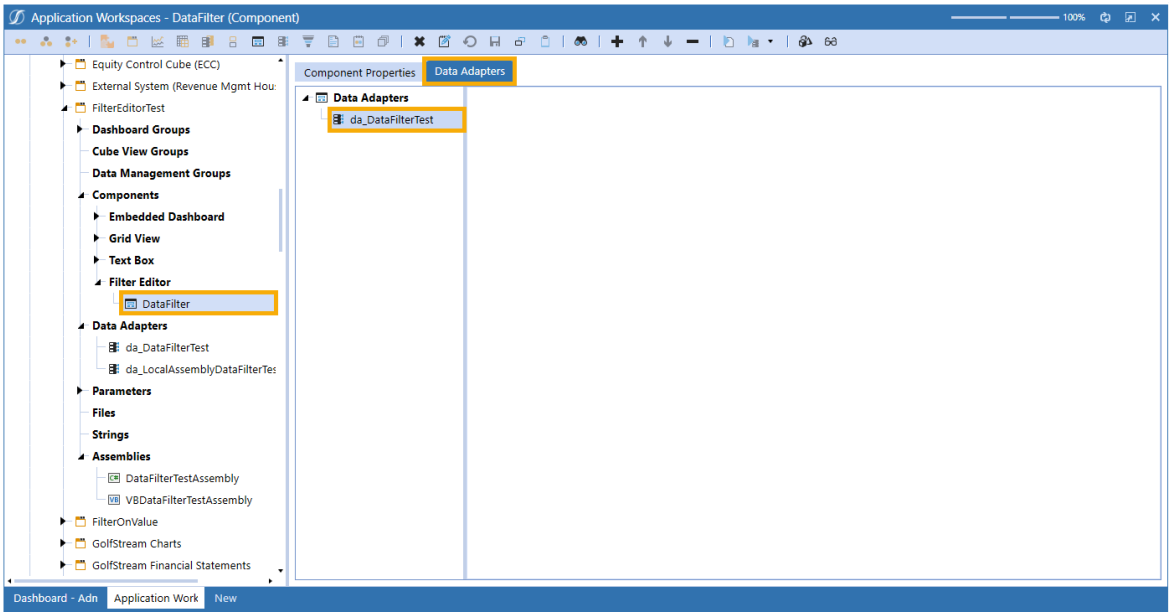
```
DataTable dt = new DataTable("DataTable1");

// Create the columns
dt.Columns.Add(new DataColumn("Name", typeof(String)) { ReadOnly = false, MaxLength =
-1});
dt.Columns.Add(new DataColumn("ID", typeof(Guid)) { ReadOnly = false});
dt.Columns.Add(new DataColumn("EmpNum", typeof(Int64)) { ReadOnly = false});
dt.Columns.Add(new DataColumn("DeptId", typeof(Int32)) { ReadOnly = false});
dt.Columns.Add(new DataColumn("DivId", typeof(Int16)) { ReadOnly = false});
dt.Columns.Add(new DataColumn("HireDate", typeof(DateTime)) { ReadOnly = false});
dt.Columns.Add(new DataColumn("Position", typeof(String)) { ReadOnly = false,
MaxLength = -1});
dt.Columns.Add(new DataColumn("IsActive", typeof(Boolean)) { ReadOnly = false});
dt.Columns.Add(new DataColumn("Salary", typeof(Decimal)) { ReadOnly = false});
dt.Columns.Add(new DataColumn("BonusRate",typeof(Double)) { ReadOnly = false});
dt.Columns.Add(new DataColumn("Manager", typeof(String)) { ReadOnly = false, MaxLength
= -1});
```

3. Create the data adapter. From the **Command Type** property, select **Method** and attach the business rule. See Data Adapters.



4. Create the Filter Editor dashboard component, then attach the data adapter and define the properties. See Dashboard Components



Settings

Date Filter Format

Select a format for the appearance of data in the filter editor.

Output Format

Change the output format to JSON or XML.

Enable Apply Button

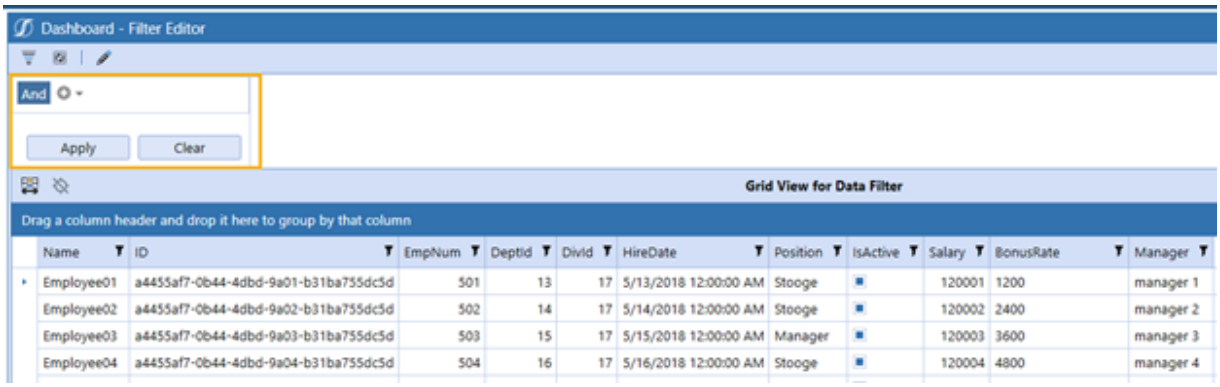
When set to True, this property enables users to build the formula and manually select when to apply it. When set to False, the filter runs automatically after each session.

Enable Clear Button

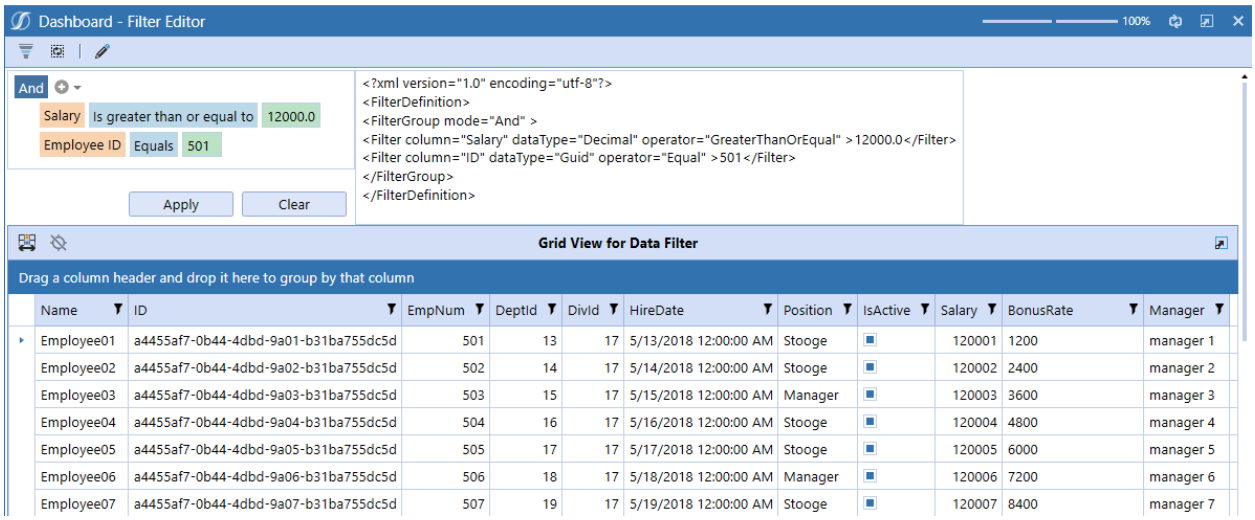
When set to True, this property enables users to use the Clear icon to clear the entire editor at once. When set to False, this property removes the Clear icon and users can only delete a single formula at a time.

Build the Filter

Navigate to the dashboard and select the **View Dashboard** icon. Filter your data using the Filter Editor component.



Build your filter using operators And, Or, Not And, and Not Or as part of a Condition, Group, or Custom Expression.



Select the **+** icon to add a Condition. Use the drop-down menu to select a Condition, Group, or Custom Expression. Select the **x** icon when hovering over a condition to delete that line item.

Select the **Apply** button to filter your data based on the filter. Select the **Clear** button to remove the entire filter.

Select each checkbox for a valid drop-down menu of choices or for the ability to enter text.

> **NOTE:** Select **Add Custom Expression** to display the **Expression Editor** dialog box. Double-click each selection to build your expression and then select the **OK** button. Select the **Apply** button to run the filter.

Gantt View

The Gantt View component is a graphical illustration of a schedule over time and is used for relational or hierarchical data (non-Cube data). This component can use a SQL query to query the data, but to fill these components with data, a Dashboard Data Set business rule must be used within a data adapter. The business rule must use an XFGanttTaskCollection object that helps the user convert a normal data source into the object-based data source required to feed these new hierarchical components.

Action

See Component Formatting and Action Properties.

Gantt View

Show Header

When set to True, the grid name and description will display as the component's header on the dashboard. When set to False, the header will be hidden.

Show Toggle Size Button
When set to True, the toggle button is enabled, and users can toggle the size of the component at runtime. When set to False, the toggle button is hidden.

Grid View

This component displays data from a data adapter in a grid. This is similar to the SQL Table Editor, except data is not editable.

> **NOTE:** The Grid View component should not be used for large datasets because it does not use paging. In this instance, a SQL Table Editor component should be used to display the data to the end user, and it will use paging to return the full dataset.

Formatting and Action

See Component Formatting and Action Properties.

Grid View

Table Name
The name of the table created in the data parameter.

Show Header
When set to True, the grid name and description display as the component's header on the dashboard. When set to False, the header is hidden.

Show Column Settings Button

When set to True, the Column Settings  icon displays on the dashboard. When set to False, the button is hidden. The button enables users to reorder columns and set column visibility.

Show Deselect All Button
When set to True, the Deselect All button displays. When set to False, it is hidden.

Allow Column Reorder
When set to True, you can drag and reorder columns. When set to False, you cannot reorder columns.

Show Toggle Size Button
When set to True, the toggle button is enabled, and users can toggle the size of the component at runtime. When set to False, the toggle button is hidden.

Show Group Panel
When set to True, the column grouping option is shown on the dashboard. Active users can drag and drop column headers to a group area to apply grouping on the data in the grid. When set to False, it is hidden.

Show Row Headers
When set to True, the grid row headers are shown on the dashboard. When set to False, they are hidden.

Show Column Headers
When set to True, the grid column headers are shown on the dashboard. When set to False, they are hidden.

Show Horizontal Grid Lines
When set to True, the horizontal grid lines are shown on the dashboard. When set to False, they are hidden.

Show Vertical Grid Lines
When set to True, the vertical grid lines are shown on the dashboard. When set to False, they are hidden.

Default for Columns are Visible
When set to True, all columns are hidden unless overridden on an individual column.

Retain Table Column Order
When set to True, the column order can be changed through the Grid View Column Format section.

Column Name for Bound Parameter
Specify the name of the database column to be used to change the value of the Bound parameter when a database row is selected. This is used when a SQL Table Editor is being used to affect the display of other components, for example, when showing detailed information for the selected row.

Allow Multiselect
Multiselect generates the Bound parameter as a comma-delimited list. Use one of the following methods to multiselect:

- Select all rows with the **Select All** checkbox.
- Select multiple rows using the **Shift** key.
- Select individual rows with **Ctrl-** select or select each row individually.

Deselect each row individually or select the **Deselect All** button if enabled.

Save State
This property enables user preferences for column order, visibility, filtering, sorting, and width. When set to True, the column activity can be changed and saved through the Grid View Column Format section. When set to False, the Vary Save State By property does not apply.

Vary Save State By
Apply Vary Save State By to Workflow Profile settings and scenario. The current Save State elements will have the control state tied to the current Workflow Profile and workflow scenario settings. If you select Use Default, Vary By Save State does not apply.

When Vary Save State By is enabled, the related component will have the additional reset option of Reset All States, which can be used to clear the user's Save States across all the Vary Save State By parameters.

Grid View Column Format

Select the ellipsis to display the **Column Format** window for the selected column.

ColumnName
The name of the column from the table where the selected formatting will be applied. The specified name must match the column name from the table exactly.

Description
The column description to display. By default, the column name from the table is displayed.

Column Display Type
You can choose from the following options:

- **Standard**: This displays the default data in the column.
- **OneStreamClientImage**: This replaces the data in the column with the associated image.
- **WorkflowStatusImage**: This replaces the data in the column with the associated workflow status indicator image.

IsVisible
This property overrides the default Columns Visible setting. The settings are True, False, or Use Default.

ParameterName
The assigned parameter name used to store the parameter value from the specified column.

Background Color

The background color to displayed on the selected column. You can choose a color from the drop-down menu.

IsGroupable

This determines whether the column can be used in the Group panel on the grid. The settings are True or False.

IsFilterable

This setting is located at the top of the column and turns the Filter button on and off. The settings are True or False.

ShowDistinctFilters

This setting turns the filter option on and off, which enables users to use checkboxes to select filtered members. The settings are True or False.

ShowFieldFilters

This enables and turns off advanced filtering on the specified column. The settings are True or False.

IsSortable

This enables and turns off sorting on the specified column. The settings are True or False.

IsMultilineText

This setting allows columns to display data on multiple rows if the column is not wide enough to display the full value. The settings are True or False.

> **NOTE:** The data will wrap on string spacing. If there are no blank spaces, the column data will not wrap. Additionally, keyed fields cannot be wrapped.

DataFormatString

Specify a number and date format for the data in the column.

> **Example:** mm/dd/yyyy will return the current Month/Day/Year using a slash

> **Example:** MM-dd-yyyy will return the Month-Day-Year using a dash.

> **Example:** N0 will return a number without a decimal point

> **Example:** #,###,0 will return a number without a decimal and a comma depicting the thousandth place.

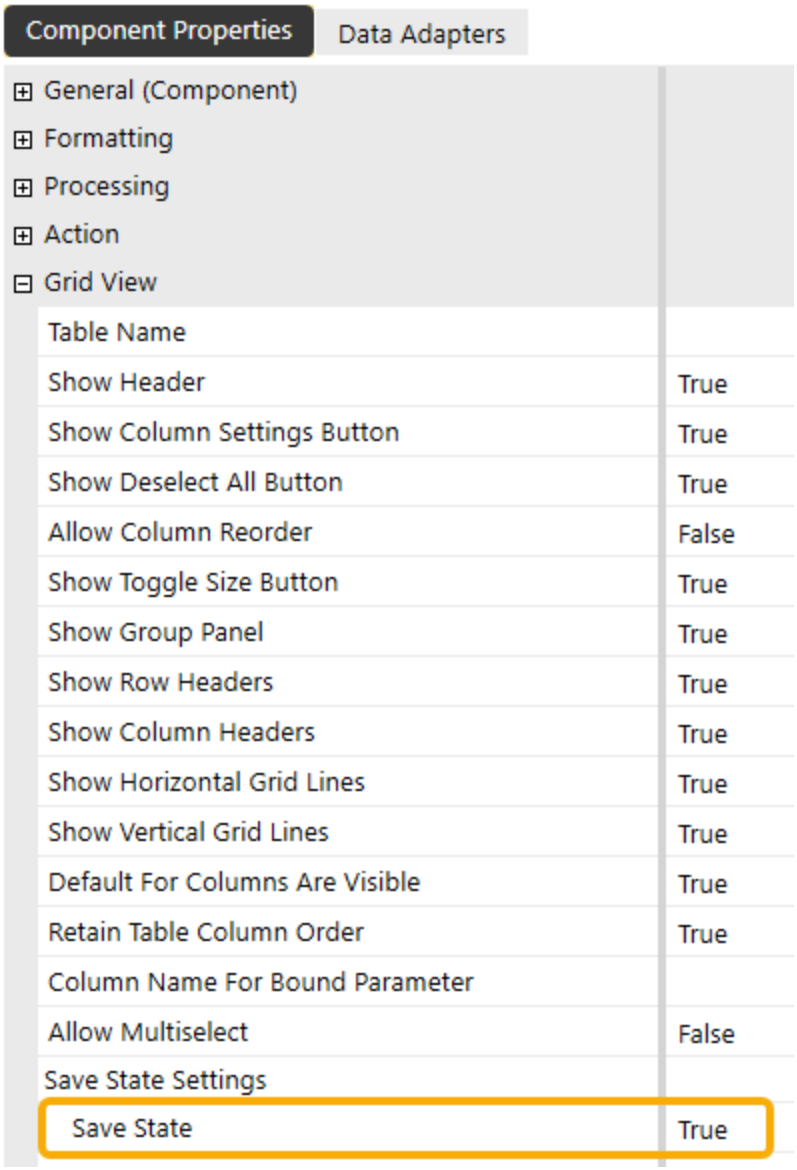
See "General Properties" in the *Application Tools Guide* for more examples of number formats.

Width

This property specifies the default column width to be displayed.

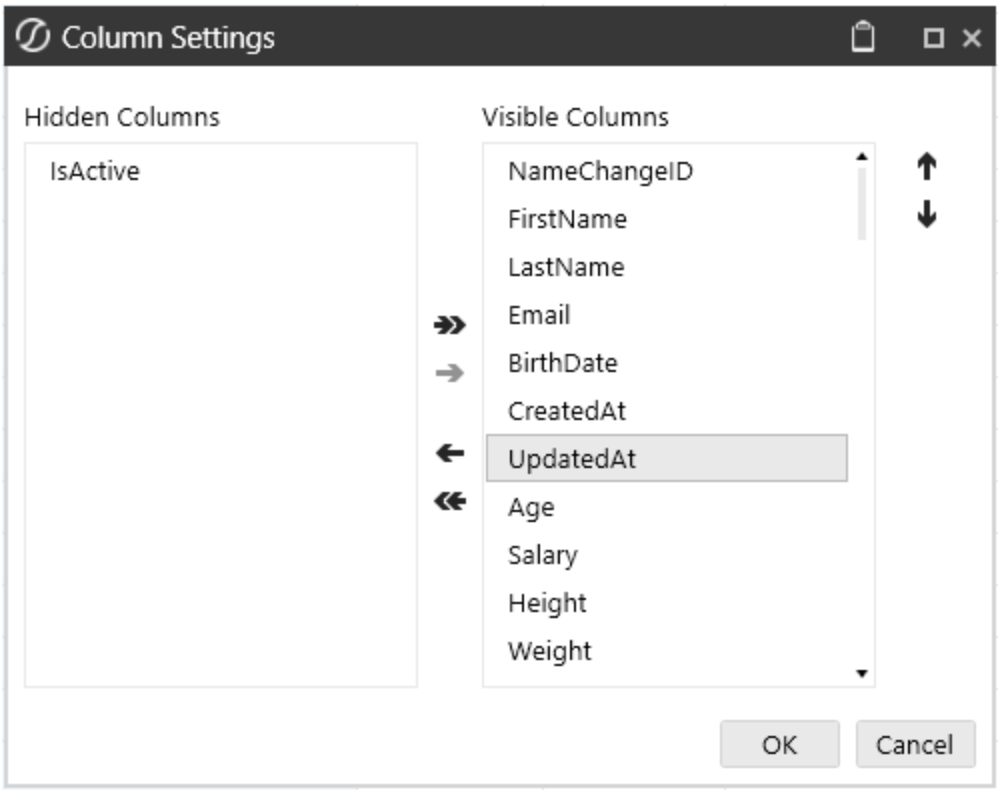
Drag and Drop

You can rearrange the columns, drag and drop the headers, and sort the columns. When the Save State property is set to True, changes to headers and columns are saved.

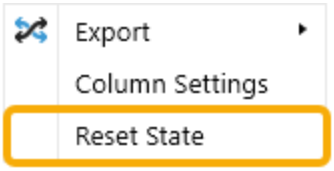


When the Save State is False, you cannot drag and drop headers when you view the dashboard.

Select the **Column Settings** icon to open the **Column Settings** window. You can hide and reorder columns on the dashboard.



Reset State

When you view a dashboard, right-click and select **Reset State** 

to go back to the default settings of the Grid View component.

Sort Column

When you select a column header, it will sort the column:

- **First click**: Ascending order
- **Second click**: Descending order
- **Third click**: No sorting order

When Save State is set to True, the changes to the sort order are saved.

Filter

When a new filter is applied to the grid, the filter icon is orange to show that it has been changed.



Image

This component is used to display an image on a dashboard.

Formatting

Tool Tip
The text to display when a mouse hovers over the component.

Display Format
The formatting assigned to the component. See Component Display Format Properties.

Image

File Source Type
The image to display.

- **URL**: The image displayed is located on a URL.
- **Workspace File**: The image displayed is located in a file stored in the Workspace maintenance unit File Section.
- **Application Database File**: The image displayed is located in a file in the Application Database Share.
- **System Database File**: The image displayed is located in a file in the System Database Share.
- **File Share File**: The image displayed is located in the File Share.

URL Or Full Name File
The URL, file path, or file name for the displayed image. Review the following syntax examples:

- **Built-in OneStream Image**: /OneStream.ClientImages;component/Misc/ToolbarImage.png
- **Web Server**: http://www.onestreamsoftware.com/img/onestream-logo.png
- **Database File System**: Enter the path and file name, Documents/Public/TestImage.png
- **Workspace File**: Enter the name of the file resource, TestImage.png

The following Excel properties are used to specify an Excel file from either an Extensible Document or Report Book to display it as an Image or Button on a dashboard.

Excel Page Number
Enter the page number of the Excel document to display.

Excel Sheet
Enter the name of the Excel worksheet to display.

Excel Named Range
Enter the named range from Excel to display. The named range must be the specified Excel sheet in the Excel Sheet property, or the Excel Sheet property must be blank to use a named range. See Component Display Format Properties to position and align the Excel image as needed.

List Box

This component displays a vertical list of predefined strings that are contained in a box and can be configured to allow single or multiple selections.

Formatting and Action

See Component Formatting and Action Properties.

Label

This is used to display text strings on a dashboard.

Formatting

Text (use {1}, {2} for Data Table Cells)
The text to display in the Label. Use {1}, {2} to reference cells from the associated data adapter's data table.

> **Example:** Sales and Profit for |!EntityParam!| are {1} and {2}.

Tool Tip
The text to display when a mouse hovers over the component.

Display Format
The formatting assigned to the control. See Component Display Format Properties

Data Table Cells From Adapter

Number of Data Table Cells
Specify the Data Table Cells available for display.

The following properties become available when a number is specified in the Number Of Data Table Cells field.

Data Row Selection Type
This specifies how the data cell row from the attached data adapter is identified and passed into the Label placeholder. You can choose from the following options:

- **First Row**: This is the first row of data from the data adapter.
- **Row Index (0-based)**: The row is specified by entering the row number in the Key Column Value Or Row Index field. This selection uses 0-base row numbering: the first row is identified as row 0. For example, if there are six rows returned in the data adapter, enter 5 to get the last row.
- **Key Column Name And Value**: The row is specified by querying the attached data adapter. Enter the column name to search in the Key Column Name field and enter the string/value to search in the Key Column Value Or Row Index field. The row where the value is found will be the specified row.

Key Column Name
This property is only enabled when Key Column Name And Value is selected for the Data Row Selection Type. Enter the column name to search.

Key Column Value Or Row Index
This property is only available when Row Index (0-based) or Key Column Name And Value are selected for the Data Row Selection Type. Enter the Key Column string/value to search or enter the Row Index number.

Result Column Name
Enter the column name to return to the Label placeholder.

Number Format
The formatting applied to a value returned to the label placeholder if a string is not returned.

Large Data Pivot Grid

The Large Data Pivot Grid is a component that allows you connect to external tables or large database tables for pivot-style analytic reporting. The grid helps designers seamlessly integrate data from external tables into a OneStream environment through a dashboard for analytic reporting.

The grid's paging and server-based processing features support large datasets and tables. In some cases, this component requires server configuration settings so users can access specific database tables.

You can specify the following properties:

Property	Setting
Show Toggle Size Button	Select True to enable the Toggle Size button.
Database Location	Identify the type of table location: Application, Framework, or External.
External Database Connection	This property is enabled when you select an External Database Location. Select a Database Connection defined in the Application Server Configuration.

Property	Setting
Table Name	Table name to be retrieved.
Row Fields	Comma-separated database column names to be placed in Row area by default.
Column Fields	Comma-separated database column names to be placed in Column area by default.
Data Fields	Comma-separated database column names to be placed in Data area by default.
Filter Fields	Assign dimensions as a default filter and assign filtering.
Where Clause	Used to assign global filters to focus the returned results from the source table.
Data Field Aggregation Types	List of comma-separated key-value pairs that specify one aggregate function type per data field. Sum, Average, Min, and Max are supported.
Excluded Fields	Allows Database Column Dimensions to be excluded from the pivot.
Page Size	Number of records returned in a page. This property defaults to 500 with a maximum value of 3000.

Property	Setting
Save State	This enables the Save button on the Large Data Pivot Grid when set to True and turns off the Save button when set to False.

Logo

This component is used to display a logo on a dashboard.

Formatting

Tool Tip
The text to display when a mouse hovers over the component.

Display Format
The formatting assigned to the control. See Component Display Format Properties.

Map

The Map component is used to display specific locations on a geographical map through a Dashboard Data Set business rule. Within each location, users can pass in parameters to perform specific actions and display additional data. Map components can only display data through a Dashboard Data Set business rule using the XFMapItemCollection objects. Define the dataset in the business rule, assign the rule to a data adapter, and assign the data adapter to the Map component. See Data Adapters.

Action

See Component Formatting and Action Properties.

Map

Show Toggle Size Button

This controls whether the toggle  icon displays on the dashboard at runtime.

Display Type

You can choose from the following options:

- **Standard**: This displays all countries, states, and cities in the respective local language. The map displays geographical location labels and road lines.
- **Humanitarian**: This displays all countries, states, and cities in English. The geographical location labels display smaller than Standard and Transport.
- **Transport**: This displays all countries, states, and cities in the respective local language. The map displays major cities with large labels.

Zoom Level

Enter the zoom value to which the map opens when the dashboard is run.

Center Latitude/Longitude

Enter the GPS coordinates to use when the dashboard is launched.

> **NOTE:** To define the exact GPS coordinates, refer to a GPS coordinate website.

Member Tree

This component displays dimensional hierarchies represented in a navigation tree. The dimensional hierarchies are passed to the component by adding a Bound parameter and are filterable based on the component's configuration using a Member Filter.

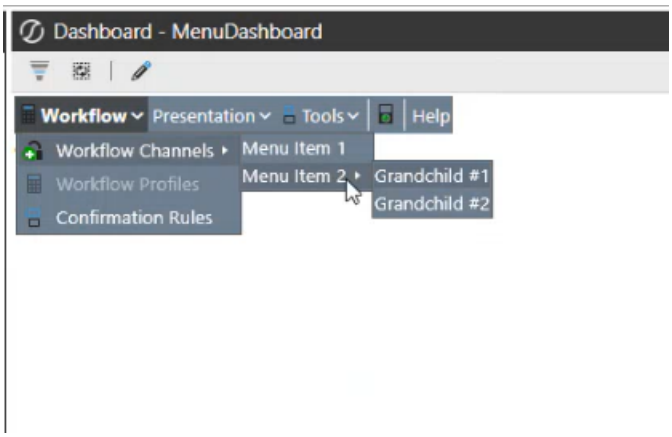
Formatting and Action

See Component Formatting and Action Properties.

> **NOTE:** When adding a Bound parameter, the only valid parameter Type for the Member Tree is Member Dialog.

Menu

This component enables developers to add navigational menus to their solutions. With the addition of navigational menus, users can easily navigate throughout the solutions. You can hover over or select a menu item to view a drop-down menu with nested options. Then, explore additional menu items by selecting those options to navigate seamlessly to your desired page.



Menu Component and Items

You can integrate the following Menu component customizations. The menu items can be set through the Workspace assembly or business rule code. The Workspace assembly or business rule code containing menu items can then be used to populate the Data Table within a data adapter. You can configure specific Display Format settings for the main Menu components through the Display Format properties. A custom parameter can also be included in the Display Format properties.

Nested Menus

You can create nested menus with the Menu component. A nested menu is a menu item with a submenu.

> **NOTE:** Parent menu items cannot have more than four nested child levels.



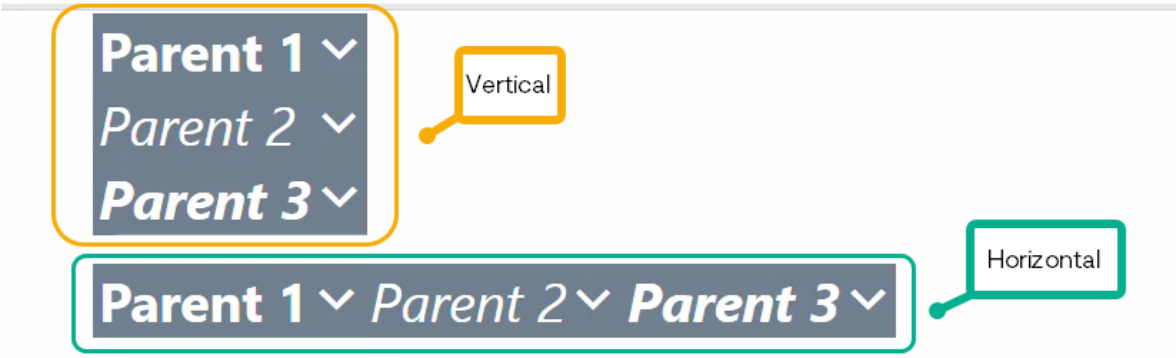
Enable or Turn Off Menu Items

You can turn specific menu items on or off. When turned off (bool isEnabled = false), you cannot select that menu item. Both parent and child menu items can be set to true or false.



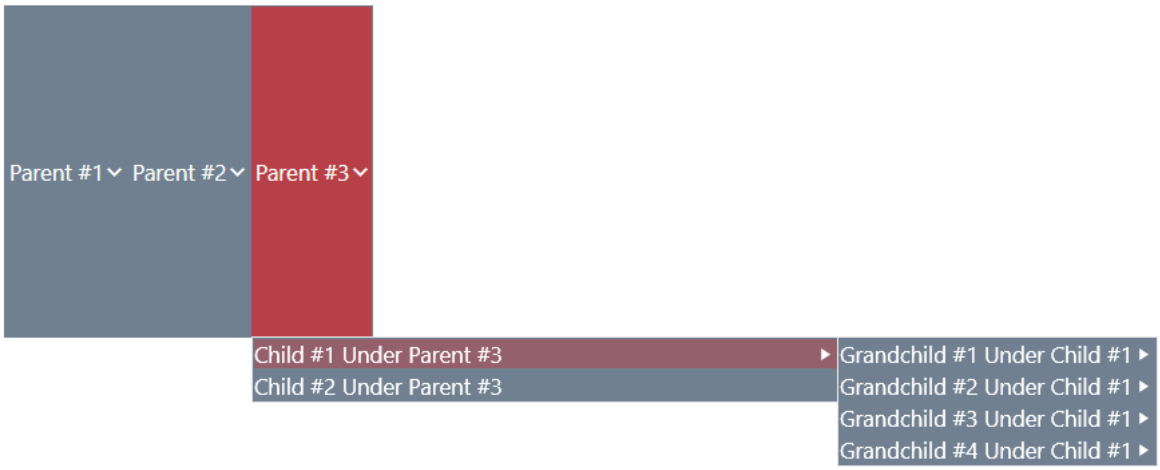
Vertical or Horizontal Menus

You can display Menu components in a vertical or horizontal format. This property is configured in the Display Format properties. See Component Display Format Properties.



Width and Height

You can configure the overall Menu component width and height along with setting each individual menu item width and height. The width or height of the overall Menu component is set in Display Format properties. See Component Display Format Properties.The width or height of individual menu items within the Menu component are set through the Workspace assembly or business rule code. Menu item width and height can be specified at any nested level depth.

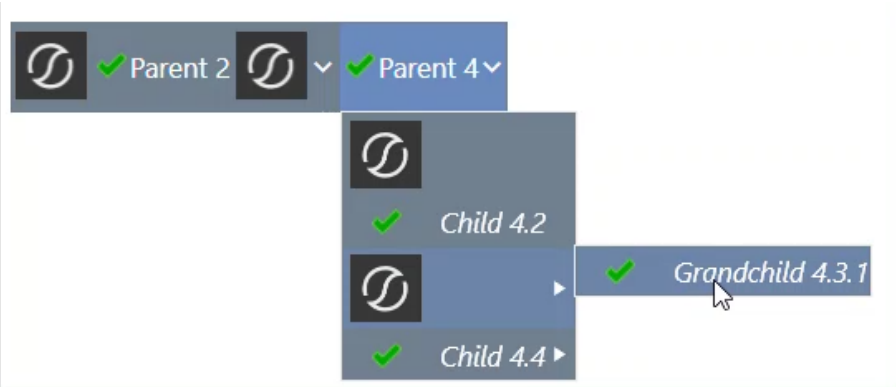


Images

Images can be set to display within a Menu component's parent and nested menu items. You can choose to display the image on its own within the menu item or along with header text or an arrow icon. If the menu item uses an image and the drop-down arrow, the image will always display to the left of the arrow.

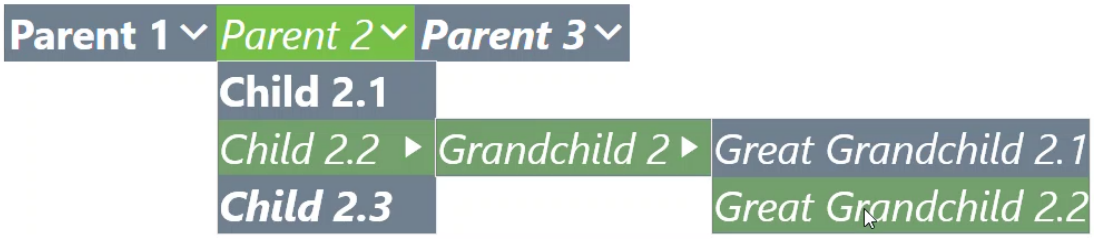
> **NOTE:** The arrow icon will only display for parent menu items if the Show Down Arrow property is set to True for the Menu component and they contain nested items. Nested items under the parent menu items, or children, are not affected by this setting and always display an arrow if they contain their own nested items.

> **NOTE:** Menu component images expand to fit all available space in the Modern Browser Experience. For consistent display between the Windows application and Modern Browser Experience, ensure that the image source being used is set to 96 DPI or that the ImageHeight and ImageWidth parameters within XFMenuItem are being used.



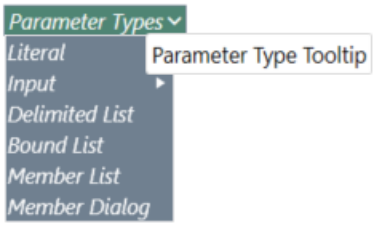
Bold and Italic Menu Options

You can set the menu item text to be bold and italic in the Workspace assembly or business rule code.



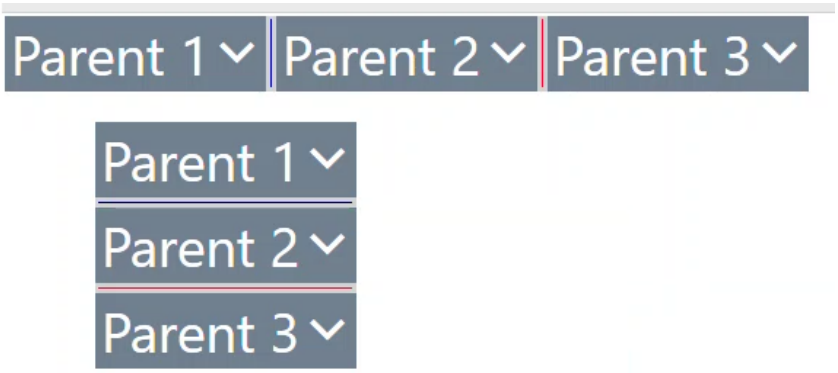
Tooltip

You can customize tooltips for the Menu component's menu items. When the menu item has a set string for the tooltip, the text displays when you hover the cursor over that menu item.



Separators

You can set and create horizontal and vertical separators within the Menu component. Separators are their own individual menu items set through Workspace assembly or business rule code and can be listed anywhere in the Menu component. When the isSeparator arguments is set to True on an XFMenuItem class, then it ignores most settings except for the color formatting options. If the Menu component is horizontal, all Separator menu items created at the parent/root level are vertical. If it is vertical, all Separator menu items created at the parent/root level are horizontal. Nested separators always display horizontally between child items.



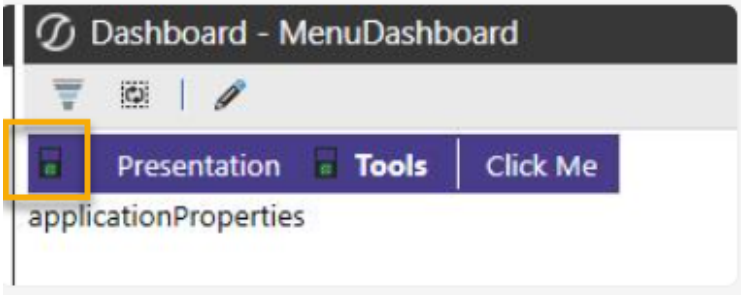
Text Color for Background and Text Color Styles

You can customize the text, background, and border colors. This can be displayed at any nested level depth.



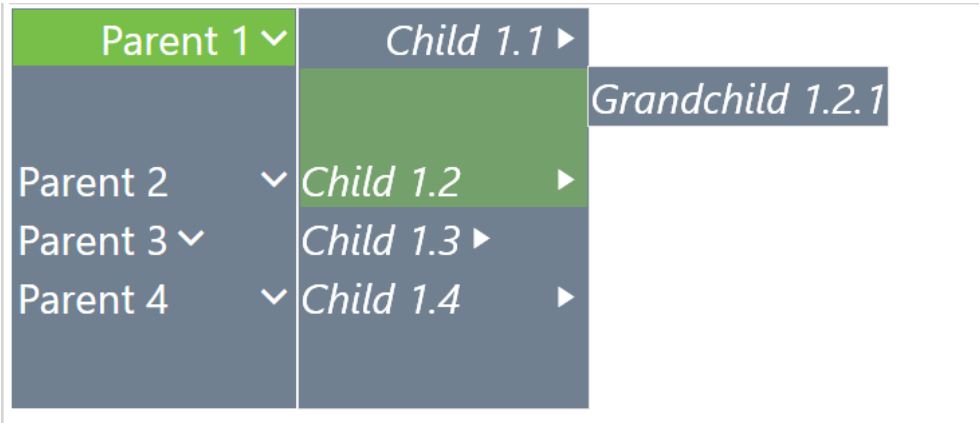
Toolbar Image Menu

You can convert navigational menus into toolbars with image icons. You can omit header text and configure the width, height, and padding properties in the Workspace assembly or business rule code. You can also pull images from dashboard Files, Client Images, and URLs.



Padding Options

You can add and set padding for each menu item. The padding settings are set in the Workspace assembly or business rule code (PaddingLeft, PaddingTop, PaddingRight, and PaddingBottom) and can be set for each individual menu item.



Set Up Menu Component

To use the full capacity of the Menu component, you must create the following items:

- Menu component
- Data adapter. See Data Adapters.
- Parameter. See "Create a Parameter" in the *Understanding the Financial Model Guide*.

  > **NOTE:** Parameters are not a necessity but can be used in certain Command Type queries.

- Data Set Service or Dashboard Data Set created through a Workspace assembly or business rule. See "Create an Assembly," "Configure the Service Factory," "Data Set Service Type," and "Build the Menu Component through Dashboard Data Sets" in the *Developer Guide*.

> **IMPORTANT:** You can complete these steps in any order.

Create a Menu Component

1. Navigate to Workspace and maintenance unit and select **Components**. Select the **Create Dashboard Component**  icon.
2. In the **Create Dashboard Component** dialog box, select **Menu**, then select the **OK** button.
3. Name the Menu component. See Menu Component Properties.
4. Select the **Save**  icon.

Select the **WsasDataSet.cs (Data Set)** Assembly File, and customize your Menu component. See Customization and Menu .

Menu Component Properties

The Menu component has the following options you can integrate and use within your navigational menu. You can configure specific Display Format settings for the main Menu components through the Display Format properties. A custom parameter can also be included in the Display Format properties.

See Component Formatting and Action Properties and Component Display Format Properties.

Menu

Auto Open
This property defaults to False. When it is set to False, you must select each Menu component item to expand its nested children. When it is set to True, the Menu component items automatically expand nested children when you hover the cursor over the item.

> **NOTE:** When Auto Open is set to True and the Menu component is displayed on a tablet, you must tap to open the menu items.

Show Down Arrow
This property defaults to False. When it is set to False, the arrow icon is not displayed on the top-level menu items even if they contain nested children. When it is set to True, the arrow icon displays on the top-level menu items containing nested children.

Display Format

> **NOTE:** When using tablets, the Menu Hover Color will display after tapping to select.

IsHorizontalOrientation
This defaults to (Use Default). Select (Use Default), True, or False.

Customization

Developers can customize the Menu component content through the Data Set assembly file. Menu components can contain nested menu items that will expand to display new menu items when hovering over or selecting them.

> **IMPORTANT:** Parent menu items can have a maximum of four nested child levels.

See Menu for a list of all options you can use and set up in the Workspace assembly or business rule code.

The Data Set assembly file returns a method call that contains the whole collection of menu items that build your Menu component. When building this out, you will notice that the code contains repetitive elements. When creating new menu items, it auto-completes and displays the default arguments, including the menu item's unique name, header text, foreground color, and background color.

C#

```
XFMenuItem XFMenuItem(string uniqueName, string headerText, string foreground, string
background, bool isBold, bool isItalic, bool isEnabled, bool isSeparator, double width,
double height, double paddingLeft, double paddingTop, double paddingRight, double
paddingBottom, XFImageFileSourceType imageFileSourceType, string imageNameOrPath, double
imageHeight, double imageWidth, double imageColumnWidth, string parameterValue, string
toolTip, List<XFMenuItem> children)
```

VB.Net

```
Sub New(uniqueName As String, headerText As String, foreground As String, background
As String, isBold As Boolean, isItalic As Boolean, isEnabled As Boolean, isSeparator
As Boolean, width As Double, height As Double, paddingLeft As Double, paddingTop As Double,
paddingRight As Double, paddingBottom As Double, imageFileSourceType As
XFImageFileSourceType, imageNameOrPath As String, imageHeight As Double, imageWidth
As Double, imageColumnWidth As Double, parameterValue As String, toolTip As String, children
As List(Of XFMenuItem))
```

After customizing and creating your Menu component, you must update, link, and connect these items. See Data Adapter and Menu Component

Sample Customization

The following sample is a fully customized Data Set service file of a custom Menu component.

VB.Net

```
Imports System
Imports System.Collections.Generic
Imports System.Data
Imports System.Data.Common
Imports System.Globalization
Imports System.IO
Imports System.Linq
Imports Microsoft.VisualBasic
Imports OneStream.Finance.Database
Imports OneStream.Finance.Engine
Imports OneStream.Shared.Common
Imports OneStream.Shared.Database
Imports OneStream.Shared.Engine
Imports OneStream.Shared.Wcf
Imports OneStream.Stage.Database
Imports OneStream.Stage.Engine
Imports OneStream.Data.DataFrame
Imports OneStream.Data.DataFrame.Abstractions
Imports OneStreamWorkspacesApi
Imports OneStreamWorkspacesApi.V800

Namespace Workspace.__WsNamespacePrefix.__WsAssemblyName
    Public Class rastr
        Implements IWsasDataSetV800

        Public Function GetDataSet(ByVal si As SessionInfo, ByVal globals As BRGlobals,
ByVal workspace As DashboardWorkspace, _
            ByVal args As DashboardDataSetArgs) As Object Implements
IWsasDataSetV800.GetDataSet
            Try
                If (globals IsNot Nothing) AndAlso (workspace IsNot Nothing) AndAlso (args
IsNot Nothing) Then
                    If args.DataSetName.XFEqualsIgnoreCase("MyDataSet") Then
                        Dim menu = New XFMenuItemCollection()
                      Dim topLevelMenuItemOne = New XFMenuItem("1", "Menu Item #1",
"White", "SlateGray", _
                                                False, False, True, False, Double.NaN, Double.NaN,
Double.NaN, Double.NaN, Double.NaN, Double.NaN,  _
                                                XFImageFileSourceType.Unknown, Nothing, Double.NaN,
Double.NaN, Double.NaN, _
                                                "menuItem1value", "Tooltip Top 1", Nothing)
                        topLevelMenuItemOne.Children = New List(Of XFMenuItem)()
```

```
                        topLevelMenuItemOne.Children.Add( _
                            New XFMenuItem("2", "Child #1", "White", "SlateGray", _
                                           False, False, True, False, Double.NaN, Double.NaN,
Double.NaN, Double.NaN, Double.NaN, Double.NaN, _
                                           XFImageFileSourceType.Unknown, Nothing, Double.NaN,
Double.NaN, Double.NaN, _
                                           "child1value", "Tooltip Child #1", Nothing))
                        topLevelMenuItemOne.Children.Add( _
                            New XFMenuItem("3", "Child #2", "White", "SlateGray", _
                                       False, False, True, False, Double.NaN, Double.NaN,
Double.NaN, Double.NaN, Double.NaN, Double.NaN, _
                                       XFImageFileSourceType.Unknown, Nothing, Double.NaN,
Double.NaN, Double.NaN, _
                                       "child2value", "Tooltip Child #2", New List(Of
XFMenuItem)() From {
                                               New XFMenuItem("4", "Grandchild #1", "White",
"SlateGray", _
                                                   False, False, True, False, Double.NaN,
Double.NaN, Double.NaN, Double.NaN, Double.NaN, Double.NaN, _
                                                   XFImageFileSourceType.Unknown, Nothing,
Double.NaN, Double.NaN, Double.NaN, _
                                                   "grandChild1value", "Tooltip GrandChild #1",
Nothing), _
                                               New XFMenuItem("5", "Grandchild #2", "White",
"SlateGray", _
                                                   False, False, True, False, Double.NaN,
Double.NaN, Double.NaN, Double.NaN, Double.NaN, Double.NaN, _
                                                   XFImageFileSourceType.Unknown, Nothing,
Double.NaN, Double.NaN, Double.NaN, _
                                                   "grandChild2value", "Tooltip GrandChild #2",
Nothing)
                        }))
                        Dim topLevelMenuItemTwo = New XFMenuItem("6", "Menu Item #1",
"White", "SlateGray", _
                                                     False, False, True, False, Double.NaN,
Double.NaN, Double.NaN, Double.NaN, Double.NaN, Double.NaN, _
                                                     XFImageFileSourceType.Unknown, Nothing,
Double.NaN, Double.NaN, Double.NaN, _
                                                     "menuItem2value", "Tooltip Top 2",
Nothing)
                        menu.MenuItems.Add(topLevelMenuItemOne)
                        menu.MenuItems.Add(topLevelMenuItemTwo)
                        Return menu.CreateDataSet(si)
                    End If
                End If

                Return Nothing
            Catch ex As Exception
                Throw New XFException(si, ex)
            End Try
        End Function

    End Class
End Namespace
```

This is how this sample Menu component displays in a dashboard:



The following sample is a fully customized Data Set Service File in C# of a custom Menu component.

```
using System;
using System.Collections.Generic;
using System.Data;
using System.Data.Common;
using System.Globalization;
using System.IO;
using System.Linq;
using Microsoft.CSharp;
using OneStream.Finance.Database;
using OneStream.Finance.Engine;
using OneStream.Shared.Common;
using OneStream.Shared.Database;
using OneStream.Shared.Engine;
using OneStream.Shared.Wcf;
using OneStream.Stage.Database;
using OneStream.Stage.Engine;
using OneStreamWorkspacesApi;
using OneStreamWorkspacesApi.V800;

namespace Workspace.__WsNamespacePrefix.__WsAssemblyName
{
    public class MenuSvc : IWsasDataSetV800
    {
        public object GetDataSet(SessionInfo si, BRGlobals brGlobals, DashboardWorkspace
workspace, DashboardDataSetArgs args)
        {
         try {
                if ((brGlobals != null) && (workspace != null) && (args != null)) {

                    // sample menu
                    if (args.DataSetName.XFEqualsIgnoreCase("MainMenu"))
                    {
                        // create the menu object
                        XFMenuItemCollection menu = new XFMenuItemCollection();

                        // create top-level items
                        XFMenuItem parentMenuItemOne = new XFMenuItem("1", "Parent 1",
                            "White", "SlateGray", false, false, true, false, null);
                        // create items for the second level
                        XFMenuItem childMenuItemOne = new XFMenuItem("1.1", "Child 1",
                            "Black", "White", true, true, true, false, null);
                        XFMenuItem childMenuItemTwo = new XFMenuItem("1.2", "Child 2",
```

```
                        "Black", "White", true, true, true, false, null);
                    // create items for the third level
                    XFMenuItem grandChildMenuItemOne = new XFMenuItem("1.1.1",
"Grandchild 1",
                        "White", "SlateGray", true, true, true, false, null);

                    // create the hierarchy by adding children to direct parents as a
List
                    childMenuItemOne.Children = new List<XFMenuItem>(){
                        grandChildMenuItemOne
                    };
                    parentMenuItemOne.Children = new List<XFMenuItem>(){
                        childMenuItemOne
                    };

                    // you can also manipulate the list once created
                    parentMenuItemOne.Children.Add(childMenuItemTwo);

                    // add top-level items to the menu object
                    menu.MenuItems.Add(parentMenuItemOne);

                    // generate the dataset and return it
                    return menu.CreateDataSet(si);
                }
            }

            return null;

        } catch (Exception ex) {
            throw new XFException(si, ex);
        }
    }
}
}
```

This is how this sample Menu component displays in a dashboard:



Data Adapter and Menu Component

After customizing and creating your Menu component, you must update, link, and connect these items. The Workspace retrieves information through its Workspace Assembly Service property, which contains the path to the assembly and service factory - assembly file. For example, MenuAssembly.WsasFactory. The service factory file references the Data Set assembly file, which contains the data and content needed for the data adapter. The data adapter can then query this data and be connected to the Menu component to generate the menu items' content on a dashboard.

> **IMPORTANT:** You can connect any of the following items in any order. You do not need to follow this order.

Update the Data Adapter

1. In the maintenance unit, navigate to the data adapter.
2. Update the **Data Source** properties:
    a. For the **Command Type** property, select **Method**.
    b. For the **Method Type** property, select **Business Rule**.
    c. For the **Method Query** property, select the ellipsis and update the field to reflect the Workspace Menu component item as it is being returned in the assembly file or business rule. You can also do this through other Command Types, Method Types, or Queries.

        > **Example:** {WS}{MainMenu}{SelectedMenuItem=[|!SelectedMenuItem!|]}

3. Select the **Save**  icon.

Update the Bound Parameter

> **NOTE:** (Optional) Follow these steps if you are using Bound parameters as part of your implementation.

1. In the maintenance unit, navigate to the parameter.
2. Rename the parameter to work with the path of the data adapter. Select the **Rename Selected Item** icon.

   > **Example:** Rename Test Parameter to SelectedMenuItem.

3. Select the **Save** icon.
4. Add the specified Bound parameter, for example, SelectedMenuItem, to the Menu component's Bound Parameter property.

Link the Data Adapter

1. In the maintenance unit, navigate to the Menu component.
2. Select the **Data Adapters** tab.
3. Select the **Add Dashboard Component** icon, and select the data adapter.
4. Select the **OK** button and then select the **Save** icon.

Link the Menu Component

1. Navigate to the dashboard and select the **Dashboard Components** tab.
2. Select the **Add Dashboard Component**  icon, and select the Menu component.
3. Select the **OK** button and then select the **Save**  icon.

Update the Workspace Assembly Service

1. Navigate to the Workspace properties.
2. Under **Assemblies**, in the **Workspace Assembly Service** field, enter the path to the service factory - assembly file.

   > **Example:** MenuAssembly.WsasFactory

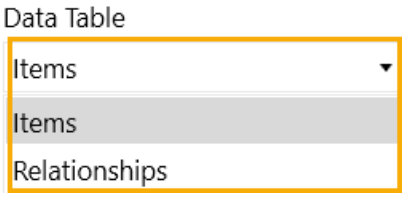
3. Select the **Save**  icon.

Run the Dashboard

1. Navigate to the dashboard linked with the Menu component.
2. Select the **View Dashboard**  icon.
3. If the Parameter dialog box displays, enter a valid argument and then select the **OK** button.

Data Table

The Data Table within a data adapter displays the Items and Relationships in your Menu component.



To view the data of an existing data adapter, follow these steps:

1. Expand the **Workspace** > **Maintenance Unit** > **Data Adapter** node, then select an existing data adapter.
2. Select the **Test Data Adapter** icon.
3. If a Parameter dialog box displays, enter valid arguments for the Parameters, then select the **OK** button.

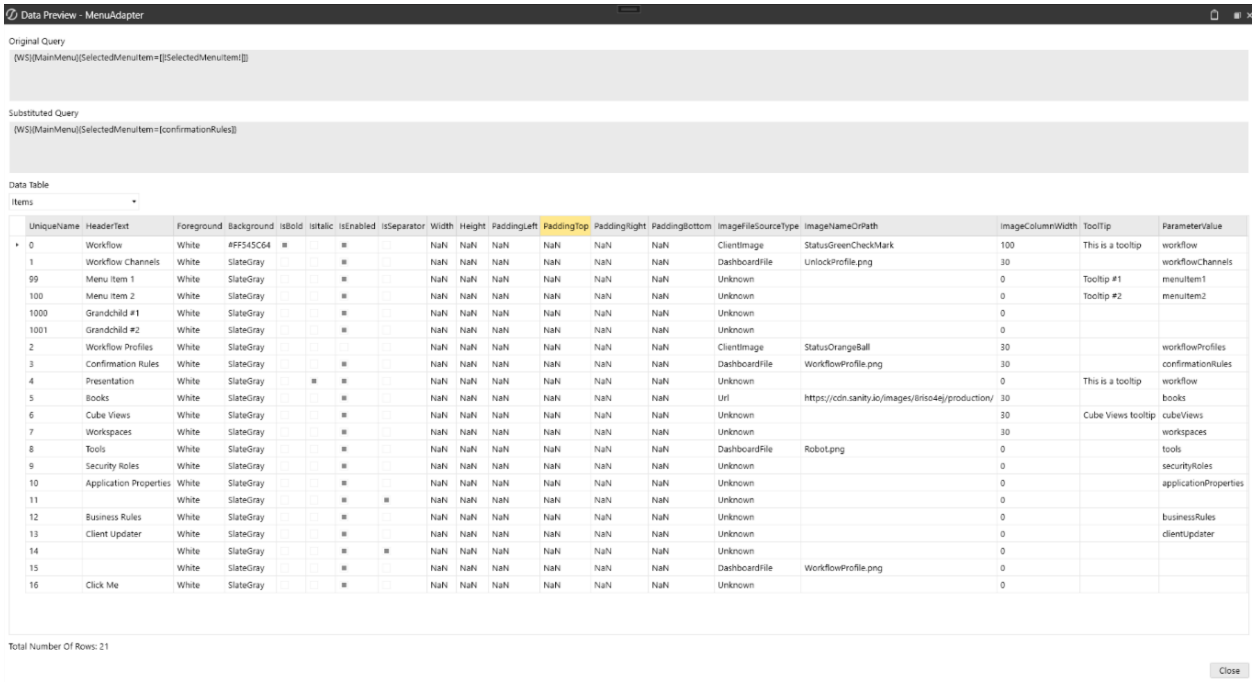
Items

The Menu component data adapter contains the following fields and displays them in a table format:

Field	Description
UniqueName	Unique Id
HeaderText	Menu item name (text that displays)
Foreground	Foreground color of menu item (this displays the text color)
Background	Background color of menu item (this displays the background color behind the text)
IsBold	Determines if HeaderText is bolded (True or False)

Field	Description
IsItalic	Determines if HeaderText is italicized (True or False)
IsEnabled	Determines if the menu item is selectable (True or False) **NOTE:** IsEnabled is different from IsVisible. When IsEnabled is False, the menu item will still be visible but cannot be selected. IsVisible determines if the Menu component is visible.
IsSeparator	Determines if the menu item is a separator (True or False)
Width	Width of menu item label
Height	Height of menu item label
PaddingLeft	Amount of padding added between the HeaderText and left border of the menu item
PaddingTop	Amount of padding added between the HeaderText and top border of the menu item
PaddingRight	Amount of padding added between the HeaderText and right border of the menu item
PaddingBottom	Amount of padding added between the HeaderText and bottom border of the menu item

Field	Description
ImageFileSourceType	Source of the image, for example, dashboard or File Explorer
ImageNameOrPath	Name of image
ImageHeight	The image height
ImageWidth	The image width
ImageColumnWidth	Amount of width added to the column of the image
Tooltip	Displays text when hovering over the menu item
ParameterValue	Parameter value of the menu item



UniqueName	HeaderText	Foreground	Background	IsBold	IsItalic	IsEnabled	IsSeparator	Width	Height	PaddingLeft	PaddingTop	PaddingRight	PaddingBottom	ImageFileSourceType	ImageNameOrPath	ImageColumnWidth	ToolTip	ParameterValue
0	Workflow	White	#FF545C64	■		■		NaN	NaN	NaN	NaN	NaN	NaN	ClientImage	StatusGreenCheckMark	100	This is a tooltip	workflow
1	Workflow Channels	White	SlateGray			■		NaN	NaN	NaN	NaN	NaN	NaN	DashboardFile	UnlockProfile.png	30		workflowChannels
99	Menu Item 1	White	SlateGray			■		NaN	NaN	NaN	NaN	NaN	NaN	Unknown		0	Tooltip #1	menuItem1
100	Menu Item 2	White	SlateGray			■		NaN	NaN	NaN	NaN	NaN	NaN	Unknown		0	Tooltip #2	menuItem2
1000	Grandchild #1	White	SlateGray			■		NaN	NaN	NaN	NaN	NaN	NaN	Unknown		0		
1001	Grandchild #2	White	SlateGray			■		NaN	NaN	NaN	NaN	NaN	NaN	Unknown		0		
2	Workflow Profiles	White	SlateGray					NaN	NaN	NaN	NaN	NaN	NaN	ClientImage	StatusOrangeBall	30		workflowProfiles
3	Confirmation Rules	White	SlateGray			■		NaN	NaN	NaN	NaN	NaN	NaN	DashboardFile	WorkflowProfile.png	30		confirmationRules
4	Presentation	White	SlateGray		■	■		NaN	NaN	NaN	NaN	NaN	NaN	Unknown		0	This is a tooltip	workflow
5	Books	White	SlateGray			■		NaN	NaN	NaN	NaN	NaN	NaN	Url	https://cdn.sanity.io/images/8riso4ej/production/	30		books
6	Cube Views	White	SlateGray			■		NaN	NaN	NaN	NaN	NaN	NaN	Unknown		30	Cube Views tooltip	cubeViews
7	Workspaces	White	SlateGray			■		NaN	NaN	NaN	NaN	NaN	NaN	Unknown		30		workspaces
8	Tools	White	SlateGray			■		NaN	NaN	NaN	NaN	NaN	NaN	DashboardFile	Robot.png	0		tools
9	Security Roles	White	SlateGray			■		NaN	NaN	NaN	NaN	NaN	NaN	Unknown		0		securityRoles
10	Application Properties	White	SlateGray			■		NaN	NaN	NaN	NaN	NaN	NaN	Unknown		0		applicationProperties
11		White	SlateGray			■	■	NaN	NaN	NaN	NaN	NaN	NaN	Unknown		0		
12	Business Rules	White	SlateGray			■		NaN	NaN	NaN	NaN	NaN	NaN	Unknown		0		businessRules
13	Client Updater	White	SlateGray			■		NaN	NaN	NaN	NaN	NaN	NaN	Unknown		0		clientUpdater
14		White	SlateGray			■	■	NaN	NaN	NaN	NaN	NaN	NaN	Unknown		0		
15		White	SlateGray			■		NaN	NaN	NaN	NaN	NaN	NaN	DashboardFile	WorkflowProfile.png	0		
16	Click Me	White	SlateGray			■		NaN	NaN	NaN	NaN	NaN	NaN	Unknown		0		

Relationships

- **ParentName**: This is the name of the parent menu item of a menu item.

> **NOTE:** Top-Level Menu Items will be blank or display Empty for the ParentName field.

- **ChildName**: This is the name of the child menu item.
- **SortOrder**: This number specifies the numerical order of the menu item in the parent-child relationship.

> **NOTE:** The first menu item in the parent-child relationship will have a SortOrder value of 0.

Data Table

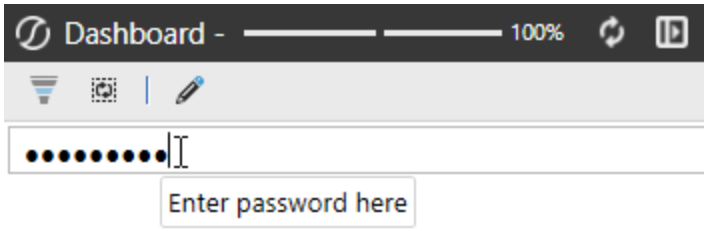
Relationships

ParentName	ChildName	SortOrder
1	1.2	1
1.2	1.2.1	0
1.2	1.2.2	1
1.2	1.2.3	2
1.2	1.2.4	3
	2	1
2	2.1	0
2	2.2	1
2	2.3	2
2	2.4	3
2	2.5	4
	3	2
3	3.1	0
3.1	3.1.1	0
3.1.1	3.1.1.1	0

Total Number Of Rows: 37

Password Box

The Password Box displays a text box where entered characters are hidden, so you can securely enter a password. When you create a Password Box, you can add tooltip text.



Formatting and Action

See Component Formatting and Action Properties.

Pivot Grid

The Pivot Grid component enables you to create a pivot table, using data from new and existing data adapters to perform multidimensional analysis. At runtime, the user can customize the layout of the report using simple drag-and-drop operations and conditional formatting based on their analysis requirements. In the Pivot Grid, data can be summarized, and calculated fields, both text and decimal, can be added and displayed in a cross-tabular format that can be sorted, grouped, and filtered. The Pivot Grid also supports drill-down, which enables you to view the underlying data adapter Results Table. The resulting output can be printed or exported to various file formats, for example, PDF, XLS, and XLSX.

All data adapters are supported when using the Pivot Grid. Using Cube View MD (Multi-Dimensional) as the source of a data adapter will return the selected Cube View as a Multi-Dimensional Fact Table and provide a source table for the Pivot Grid. The results of the Cube View MD are the following dimensions as columns: Entity, Consolidation, Scenario, Time, View, Account, Flow, Origin, IC, and UD1-UD8.

See the *Pivot Grids Guide*.

General

Name
The name of the Pivot Grid.

Description
A description of the Pivot Grid.

Maintenance Unit
The maintenance unit the component belongs in.

Component Type
The type of component.

Pivot Grid

Show Toggle Size Button
When set to True, the toggle button is enabled and users can toggle the size of the component at runtime. When set to False, the toggle button is hidden.

Row Fields
Enter a comma-separated list of column names from the Results Table to be placed in the Row Area of the Pivot Grid by default.

Column Fields
Enter a comma-separated list of column names from the Results Table to be placed in the Column Area of the Pivot Grid by default.

Data Fields
Add a Measure from the Results Table to be assigned as the default Measure. Multiple measures can be added using a comma to separate the column names.

Field Groups
Enter a comma-separated list of column names from the Results Table to be grouped together as a default.

Save State
When set to True, the Save button is enabled on the Pivot Grid. When set to False, the Save button on the Pivot Grid is turned off.

> **NOTE:** Layout button on Pivot Grid when copying dashboards: When copying a dashboard that contains a Pivot Grid or Large Data Pivot Grid, the Pivot Grid and Large Data Pivot Grid component layouts are saved per dashboard location, not by component. The next time each dashboard is run, the Pivot Grids should have different saved layouts. For information on the use and design elements of the Pivot Grid, see "Using the Pivot Grid" in the *Pivot Grids Guide*.

Radio Button Group

This displays a choice of mutually exclusive related options where users can only choose one.

Formatting and Action

See Component Formatting and Action Properties.

Report

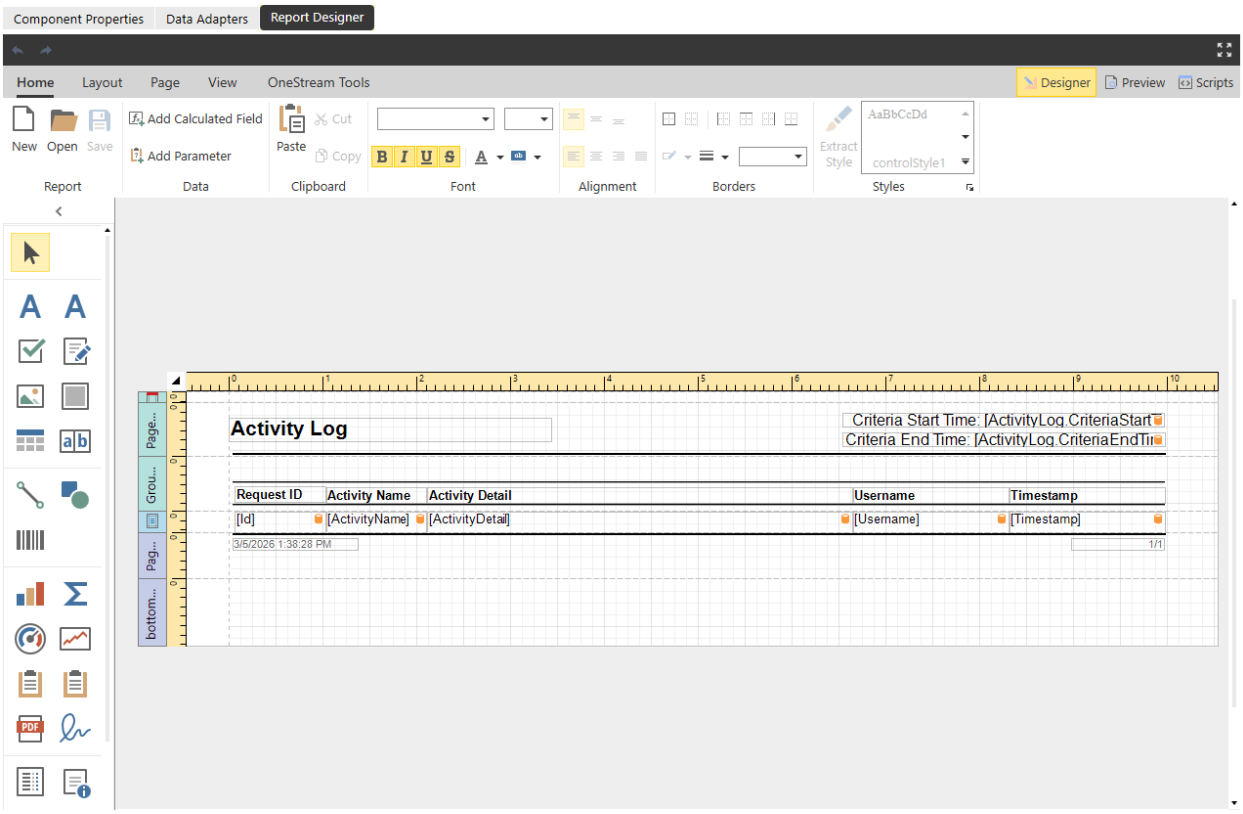
This component enables you to display a report.

Auto Fit To Page Width

When set to True, this will auto fit the Report to the area allocated in the dashboard.

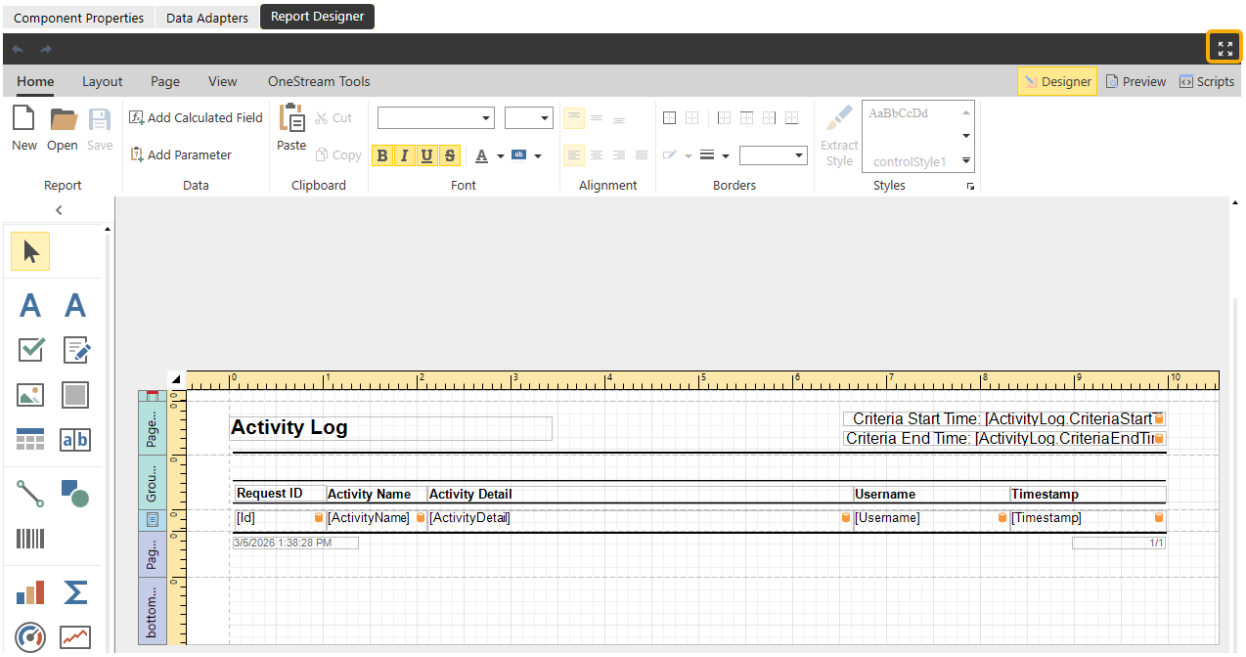
Report Designer

The Report Designer is built into the dashboard Report component and enables you to edit reports on the report itself. Select the component and then select the **Report Designer** tab.

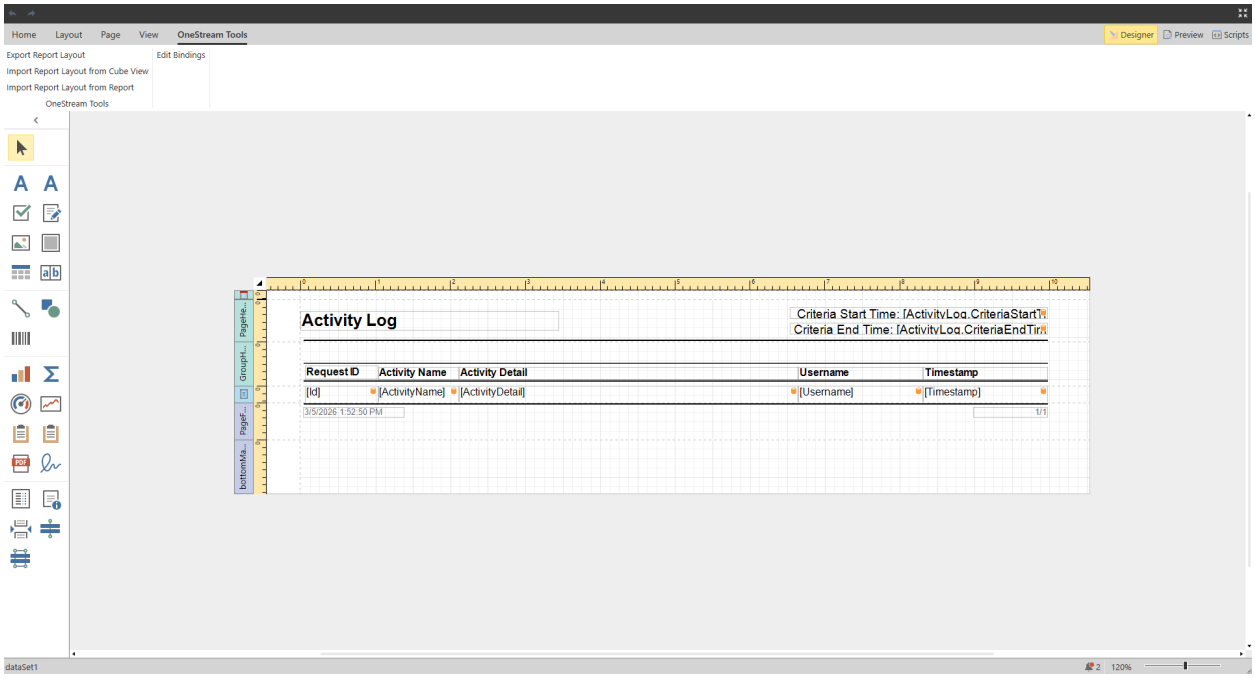


Toggle Page Size

Select the **Maximize** icon in the upper right corner of the Report Designer to make the Report Designer full screen.



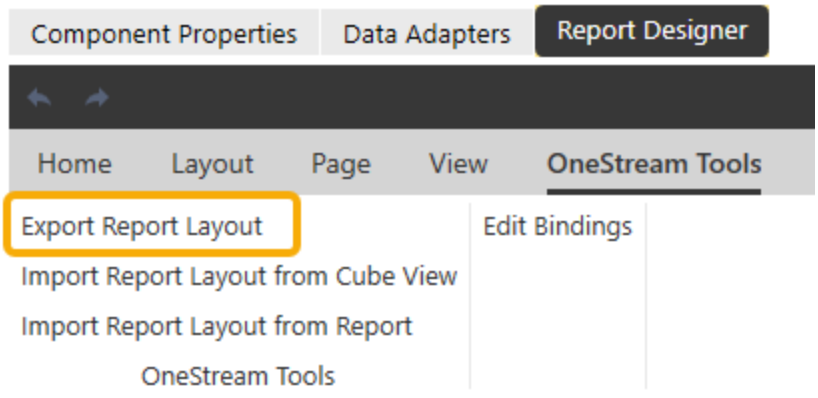
The report maximizes to full screen.



Export Report Layout

You can export the report `*.repx` file.

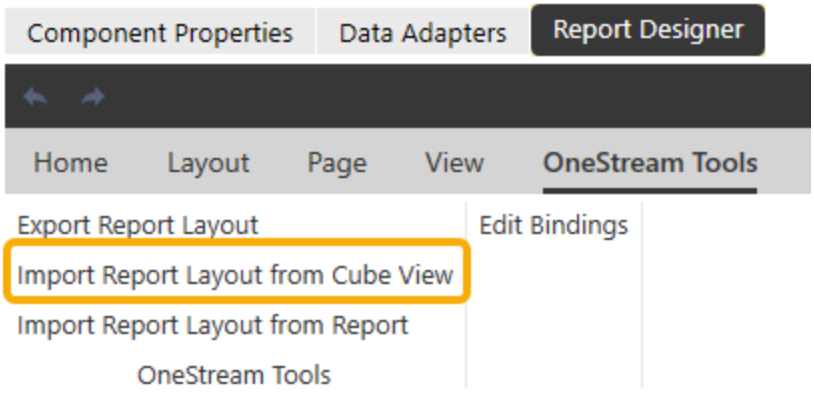
Select the **OneStream Tools** tab and then select **Export Report Layout**. The Save As window displays.



Browse to a location to save the report and select the **Save** button.

Import Report Layout from Cube View

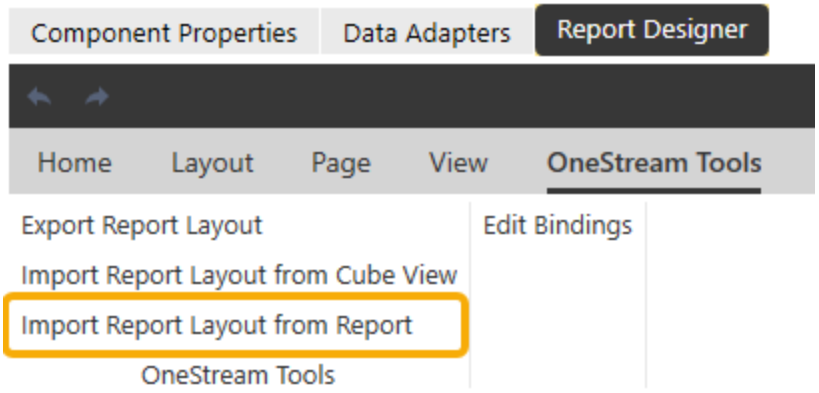
1. Select the **OneStream Tools** tab and then **Import Report Layout from Cube View**.



2. In the **Object Lookup** window, browse to a Cube View and select the **OK** button.
3. The report layout changes to the formatting of the selected Cube View.
4. Select the **Save** button to save the new report layout.

Import Report Layout from Report

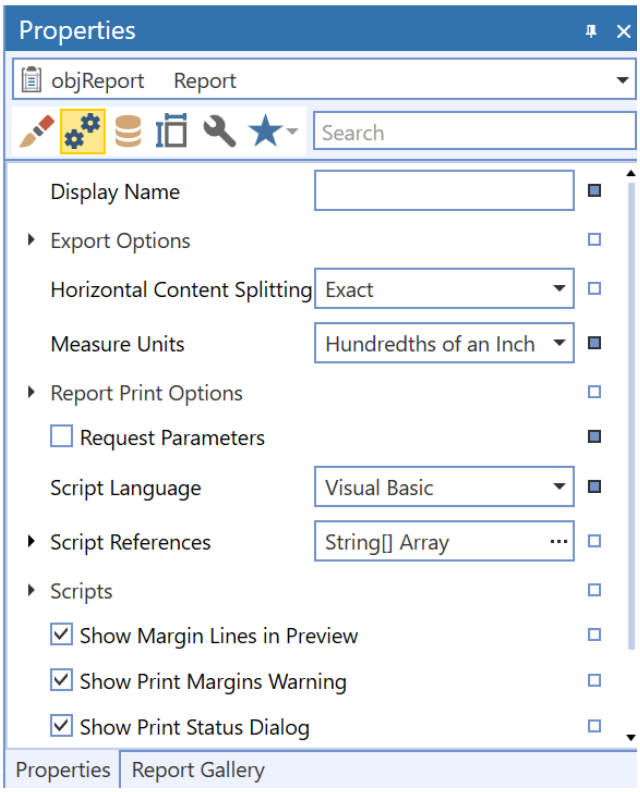
1. Select the **OneStream Tools** tab and then **Import Report Layout from Report**.



2. In the **Object Lookup** window, browse to a dashboard report component and then select the **OK** button.
3. The report layout changes to the formatting of the Report component you selected.
4. Select the **Save** button to save the new report layout.

Functionality

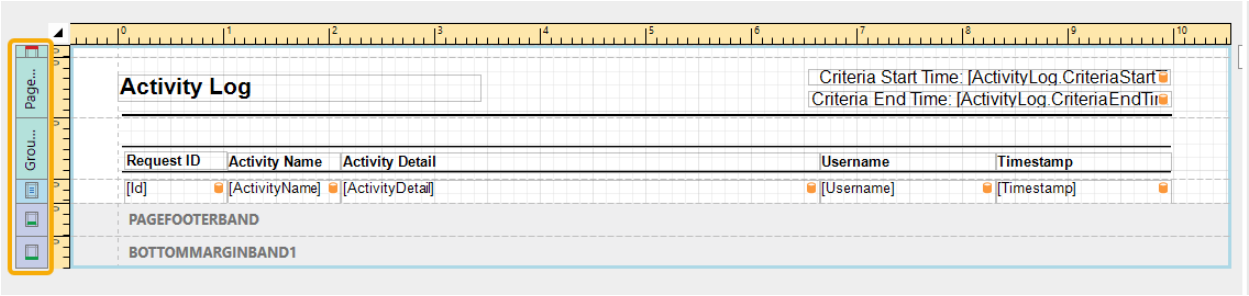
For most right-click functionality in Report Designer, use the **Properties** dialog box.



> **NOTE:** When editing sub reports, you can access the Generate Own Pages option from the Properties menu.

Display Band Details

To expand any of the bands, double-click the icon in the left column.

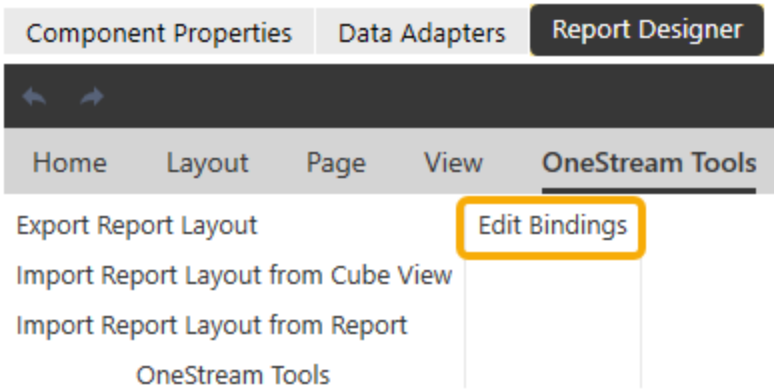


Edit Bindings by Report

Edit bindings directly and evaluate all the controls with the built-in component in the Report Designer. The Edit Bindings window lists all controls, property types, and data bindings. You can view bindings with an invalid status and apply corrections. The Property Type column contains type definitions.

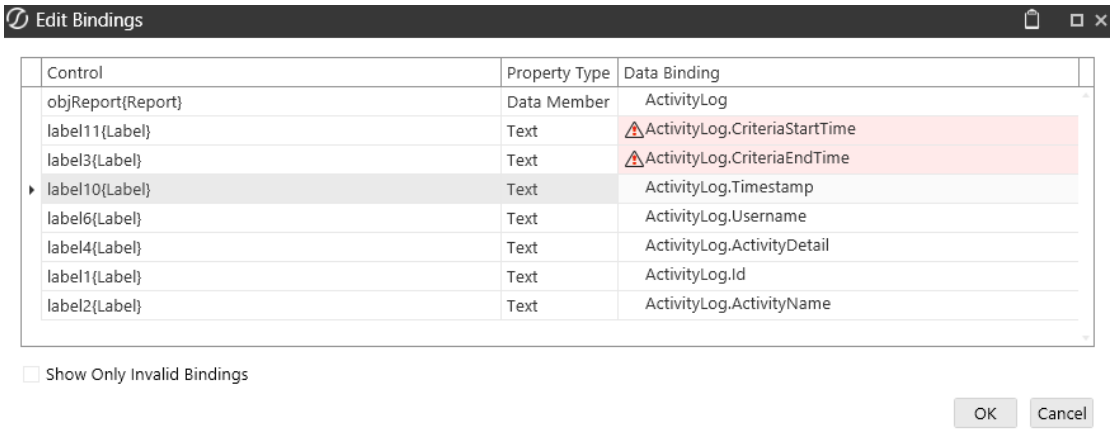
Complete the following steps to edit bindings:

1. Select the **OneStream Tools** tab, then select **Edit Bindings**.



2. In the **Edit Bindings** window, select any of the column headings to sort the data in the

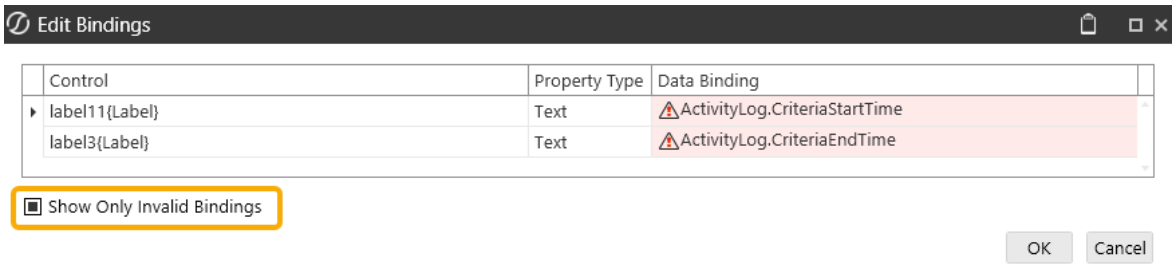
column.



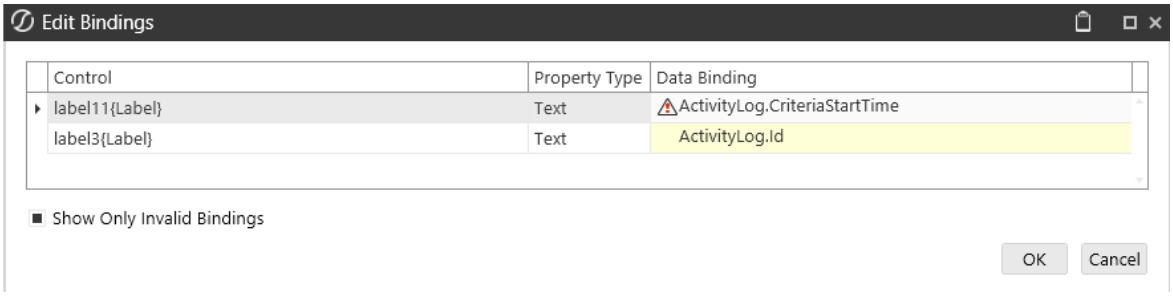
Show Invalid Bindings

The Edit Bindings window lists invalid bindings. The source binding is set to a path that does not exist in your dataset.

1. In the **Edit Bindings** window, select the **Show Only Invalid Bindings** checkbox. A list of invalid bindings displays. The invalid bindings have a red background.



2. From the **Data Binding** column, select one of the invalid bindings. From the drop-down menu, select the correct data to a path that exists in your dataset.
3. After you select an item outside of the row, the color of the row changes to yellow to identify that it was modified.



> **NOTE:** Even if you modify an invalid binding to make it valid, it still displays in the invalid list.

4. Select the **OK** button.
5. Select the **Save** button to save your changes to the report and database.

Text Editor

This is used to view the Text Editor component in a dashboard. This is available in the Windows application and can be displayed in the Modern Browser Experience.

Text Editor

Show Toggle Size Button
When set to True, the toggle button is enabled and users can select the size of the component at runtime. When set to False, the toggle button is hidden.

File Source Type
The file to display. You can choose from the following options:

- **Workspace File**: Displays a file stored in a Workspace maintenance unit's file section.
- **Application Database File**: Displays a file stored in the Application Database.
- **System Database File**: Displays a file stored in the System Database.

- **File Share File**: Displays a file from the File Share.

> **NOTE:** Application Database, System Database, and File Share exist in the File Explorer.

Url Or Full File Name

Specifies the URL or full name of the file being used.

Process Extensible Document

This is set to True, which will process Extensible Documents when embedded in a dashboard.

Allow Create File

When set to True, this enables you to create files.

Allow Open File

When set to True, this enables you to open text files.

Allow Save File

When set to True, this enables you to save files.

Allow Refresh

When set to True, the Refresh Document toolbar button displays under the OneStream ribbon in the Text Editor component. When set to False, the Refresh Document button does not display.

Show Ribbon

When set to True, this enables you to view the ribbon below the menu bar. If set to False, this ribbon is hidden.

Minimize Ribbon

This defaults to False. When set to True, this minimizes the ribbon below the menu bar. When set to False, the ribbon displays as normal.

Text Viewer

This is used to view text and rich text documents similar to those created in Microsoft Word. Documents cannot be created or edited using this tool.

Show Toggle Size Button
When set to True, the toggle button is enabled and users can update the size of the component at runtime. When set to False, the toggle button is hidden.

File Source Type
The file to display. You can choose from the following options:

- **Workspace File**: Displays a file stored in a Workspace maintenance unit's File section.
- **Application Database File**: Displays a file stored in the Application Database.
- **System Database File**: Displays a file stored in the System Database.
- **File Share File**: Displays a file from the File Share.

> **NOTE:** Application Database, System Database, and File Share exist in the File Explorer.

Url Or Full File Name
Specifies the URL or full name of the file being used.

Process Extensible Document
This is set to True, which will process Extensible Documents when embedded in a dashboard.

Spreadsheet

This is used to display spreadsheet files created with the Spreadsheet feature in a dashboard. See "Spreadsheet" in the *Application Tools Guide*.

> **NOTE:** If a Spreadsheet referenced by a dashboard contains functions such as XFGetCell or XFSetCell that use a custom dashboard parameter, additional configuration is required. In this scenario, use XFGetDashboardParameterValue to retrieve the parameter value. For example, if the dashboard provides a custom parameter named ParamEntity, use XFGetDashboardParameterValue to retrieve the parameter value and store it in a spreadsheet cell, such as B1. Then, configure XFGetCell, XFSetCell, or similar functions to reference cell B1 instead of the dashboard parameter directly.

Spreadsheet

Show Toggle Size Button
When set to True, the toggle button is enabled and users can select the size of the component at runtime. When set to False, the toggle button is hidden.

File Source Type
The file to display. You can choose from the following options:

- **Workspace File**: Displays a file stored in a Workspace maintenance unit's File section.
- **Application Database File**: Displays a file stored in the Application Database.
- **System Database File**: Displays a file stored in the System Database.
- **File Share File**: Displays a file from the File Share.

Url Or Full File Name
Displays the URL or full name of the file being used.

Process Extensible Document
This is set to True. It will process extensible documents when embedded in a dashboard.

Refresh Spreadsheet When Opened
When set to True, this automatically refreshes the spreadsheet file when it is opened.

Allow Create File
When set to True, this enables users to create spreadsheet files.

Allow Open File
When set to True, this enables users to open spreadsheet text files.

Allow Save File
When set to True, this enables users to save spreadsheet files.

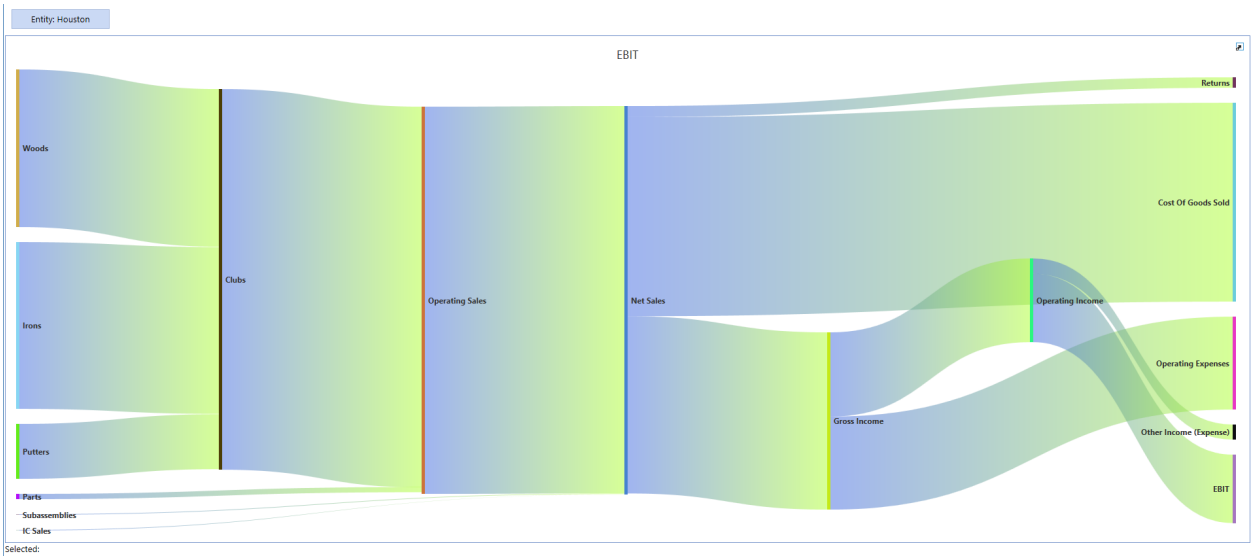
Allow Submit Data
When set to True, the Submit button in the ribbon is enabled, and users can load data to the forms channel.

Allow Calculate Data
When set to True, the Calculate button in the ribbon is enabled, and users can call calculate options using it.

Show Ribbon
When set to True, users can see the Ribbon below the Menu Bar. When set to False, the Ribbon is hidden

Sankey Diagram

This component can be used with Windows application dashboards to display Sankey diagrams. Sankey diagrams enhance the visualization of connected data sources and entities using method command type business rules or single-row Cube Views with multiple columns.



> **NOTE:** Diagrams can be exported as .png files by right-clicking outside of the links and then selecting PNG.

Action

See Component Formatting and Action Properties.

Sankey Diagram

Show Toggle Size Button

When set to True, the toggle button is enabled, and users can select the size of the component at runtime. When set to False, the toggle button is hidden.

Show Title

When set to True, the title is centered above the diagram. When set to False, no title is shown in the diagram.

Show Node Labels
When set to True, node labels display on the diagram. When set to False, node labels do not display on the diagram.

Source Link Color
When set to (Use Default), the source link colors automatically change with each refresh. Set a custom or preloaded color to maintain the same color.

Target Link Color
When set to (Use Default), target link colors automatically change with each refresh. Set a custom or preloaded color to maintain the same color.

Link Transparency
Specify the level of transparency displayed on the diagram by selecting a value between zero and one. Zero is opaque and one is transparent. The default is 0.5.

Node Width
Specify the width of the bar for the node. Increase this number for a wider bar.

Vertical Node Indent
Specify the vertical space between nodes. Increase this number for more white space.

Node Alignment
Specify the vertical alignment to the top, bottom, or center of the diagram.

Node Link Weight Tooltip Prefix
Set the tooltip prefix as any keyboard character (!@#$%^&*()<>?) or alphanumeric text. This lets you input a culture currency like $ or €.

The following example sets the prefix as $ if the user is set to English:

```
if (args.FunctionName.XFEqualsIgnoreCase("GetUserCultureCurrency"))
{
UserInfo userInfo = EngineUsers.GetUser(si);
CultureInfo culture = new CultureInfo(userInfo.UserPreferences.Culture);
```

```
return culture.NumberFormat.CurrencySymbol;
}
```

XFBR can be changed for culture. To get another user culture, replace `GetUser(si)` with `GetUser(si, "nameOfUser")`.

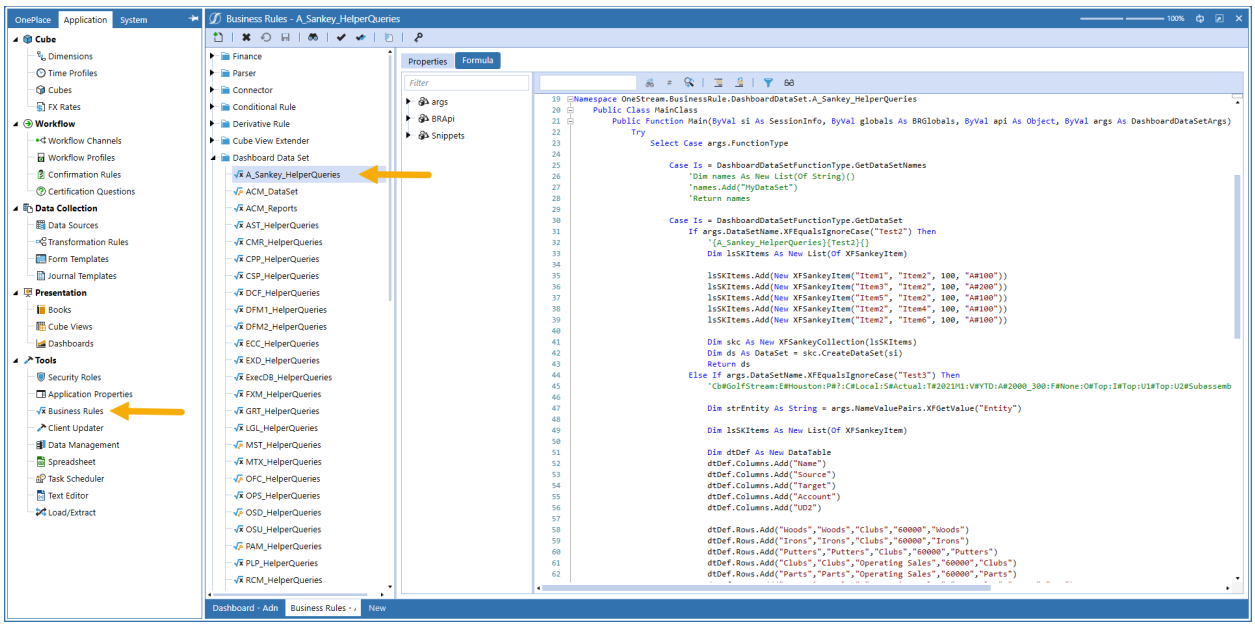
Node Link Weight Tooltip Format

Specify the tooltip formating.

> **Example:** 0.00; #.##; #,###,###.00; 000.#'string'

Business Rules and Sankey Diagrams

When using a method command type business rule, in addition to defining the Sankey items, these columns must be defined from the data table: Name, Source, Target, and Value.



> **NOTE:** Negative values are not recognized by Sankey diagrams. Links between items do not flow when a negative value is present.

State Indicator

This is used to indicate a specific status on a dashboard.

Formatting

Text (use {1}, {2} for Data Table Cells)
The text to display in the component. Use {1}, {2} to reference cells from the associated data adapter's data table.

> **Example:** Sales and Profit for |!EntityParam!| are {1} and {2}.

Tool Tip
The text to display when the cursor hovers over the component.

Display Format
The formatting assigned to the control. See Component Formatting and Action Properties.

Data Table Cells from Adapter

Number of Data Table Cells
Specify the Data Table Cells to determine which state to display.

The following properties become available when a number is enters in the Number of Data Table Cells field.

Data Row Selection Type
This specifies how the data cell row from the attached data adapter is identified and passed into the Label placeholder. You can choose from the following options:

- **First Row**: This is the first row of data from the data adapter.
- **Row Index (0-based)**: The row is specified by entering the row number in the Key Column Value Or Row Index field. This selection uses 0-base row numbering where the first row is identified as row 0, for example, if there are six rows returned in the data adapter, enter 5 to get the last row.
- **Key Column Name And Value**: The row is specified by querying the attached data adapter. Enter the column name to search in the Key Column Name field and enter the string/value to search in the Key Column Value Or Row Index field. The row where the value is found will be the specified row.

Key Column Name
This property is only enabled when Key Column Name And Value is selected for the Data Row Selection Type. Enter the column name to search.

Key Column Value Or Row Index
This property is only available when Row Index (0-based) or Key Column Name And Value is selected for the Data Row Selection Type. Enter the Key Column string/value to search, or enter the Row Index number.

Result Column Name
Enter the column name to return.

Number Format
The formatting applied to a value returned if a string is not returned.

States

State Indicator Type
The type of image used for indicating the status. Valid states start at index zero and include the options:

- **Arrow**: Up, Down, Left, Right Arrow
- **Lamp**: Off, Green, Yellow, Red
- **Smile**: Very Happy, Happy, Neutral, Sad
- **Traffic Lights**: Off, Green, Yellow, Red

Default State Index
The default state is 0-4.

State Selection Type (Image = Up)
Specify the state selection type and then define the minimum amount, maximum amount, or a list of comma-separated text items that the value from the first Data Table Cell must satisfy for the State to be selected. You can choose from the following options:

- **(Not Used)**: The state is not used.
- **Minimum Amount**: The minimum amount for the state to be selected.
- **Minimum Amount Inclusive**: The minimum amount including the specified value for the state to be selected.
- **Maximum Amount**: The maximum amount for the state to be selected.
- **Maximum Amount Inclusive**: The maximum amount including the specified value for the state to be selected.
- **Range**: The amount between two values for the state to be selected.
- **Range Minimum Amount Inclusive**: The amount between two values including the minimum amount for the state to be selected.
- **Range Maximum Amount Inclusive**: The amount between two values including the maximum amount for the state to be selected.

- **Range Minimum Maximum Amount Inclusive**: The amount between two values including both the minimum and the maximum amount for the state to be selected.
- **Equals Text Item**: If the string equals the string specified in the Text Items field, it should be selected for the specified state.
- **Starts With Text Item**: If the value starts with the string specified in the Text Items field, it should be selected for the specified state.
- **Ends With Text Item**: If the value ends with the string specified in the Text Items field, it should be selected for the specified state.
- **Contains Text Item**: If the value is listed in the Text Items field, it should be selected for the specified state.

Minimum Amount
The value used in state tests for minimum amount.

Maximum Amount
The value used in state tests for maximum amount.

Text Items
The text values used in the following State Selection Types: Equals Text Item, Starts With Text Item, Ends With Text Item, and Contains Text Item.

Supplied Parameter

This is a control that holds a parameter value for use on a dashboard, but it will not be displayed.

Bound Parameter

This is the name of the parameter that the component will represent. The component will use the parameter to get its list of options. The selection will specify a value for the parameter when it is referenced elsewhere through piped variable naming. Select the ellipsis ... to open the **Object Lookup** dialog box. Either search the parameter name, or use the navigation tree to expand Workspace maintenance units to find the parameter.

> **NOTE:** Piped variable names are not available in this setting.

Text Box

This is used to display a text box. You can also add formatted rich text to embedded refreshable content. This content can be used for narrative and can be edited and published into a report.

- Special Formatting Changes
- Insert Rich Text
- Modern Browser Experience with Rich Text

Text Box Properties

Allow Rich Text
When this is set to True, you can format text directly in the Text Box component using the built-in formatting toolbar. You can format elements like font, color, and alignment. This property defaults to False. When set to False, this defaults to plain text and the component does not contain formatting options.

> **NOTE:** When set to True, the Multiline property is not enabled.

Enable Spell Check
This property is enabled when the Allow Rich Text property is set to True. When set to True, this property displays the spell check options in the toolbar.

> **NOTE:** This property cannot be updated through parameters or business rules.

When Enable Spell Check is set to True, these spell check icons are available in the Text Box ribbon:

- **Proofing**: Spelling and language
- **Spell Check**: English (United States)

The default for Spell Check is English. Spell Check only functions when English is selected in Windows for international users. When set to Not Used, spelling is not checked.

> **NOTE:** Existing Text Box components have Spell Check turned off, and the default is set to False.

Formatting and Action

See Component Formatting and Action Properties and Component Display Format Properties.

IsReadOnly
This property enables you to set the rich text functionality in the Text Box component to be read-only or editable for select users. When set to True, the text box content is read-only. When set to False or (Use Default), you can edit the text box content.

You can use the Display Format window to select True or False, or you can type IsReadOnly= to equal either True or False in the Display Format field. This property affects existing text boxes and rich text boxes.

Set IsReadOnly Property

You can use a business rule or Custom parameters to set the IsReadOnly parameter to True or False. You can set this property through the following methods:

Method 1

You can set a Dashboard String business rule in a Workspace assembly with the IsReadOnly property. You can set the IsReadOnly property using Workspace assemblies by creating a Dashboard String Function business rule, which returns "IsReadOnly = True" or "IsReadOnly = False". This will not be automatically added to the assembly file code.

For example, in **Component Properties** > **Display Format**, enter the following string in the field:

```
XFBR(Workspace.Current.MyAssembly.MyStrings, IsReadOnly)
```

In the Workspace assembly file, update the XFBR string to return either "IsReadOnly = True" or "IsReadOnly = False." Refresh your dashboard after updating the XFBR string to display the updated state. If IsReadOnly is set to False, users can edit.

Method 2

You can reference a Custom Parameter in the Component Properties > Display Format field.

1. Go to **Component Properties**.
2. In the **Display Format** field, enter the Custom parameter, for example, **|!TextBoxReadOnly!**.
3. In the **Custom Parameter** field, enter **IsReadOnly = True**.

If the Display Format property does not reference IsReadOnly, but the Display Format field references the Custom parameter, which contains IsReadOnly = True, the text box will be IsReadOnly and users cannot edit.

Method 3

You can set a Dashboard XFBR String business rule with the IsReadOnly property. You can set the IsReadOnly property using business rules by creating a Dashboard XFBR String Function business rule which returns "IsReadOnly = True" or "IsReadOnly = False". This will not be automatically added to the args line of code.

In a Dashboard XFBR String business rule, update the XFBR string to return either `"IsReadOnly = True"` or `"IsReadOnly = False"` inside the args lines of code. Refresh your dashboard after updating the XFBR string to display the updated state. If IsReadOnly is set to False, users can edit.

Special Formatting Changes

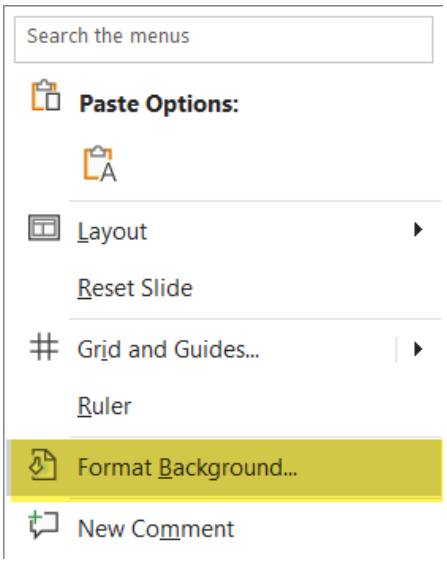
Background Color

You can include the optional BackgroundColor setting to specify the background color of Rich Text images. If the BackgroundColor setting is blank or not valid, it will default to #FFFFFF (white). The value for this setting must be in Hexadecimal format (#RRGGBB or #AARRGGBB) and can be prefixed with or without #.

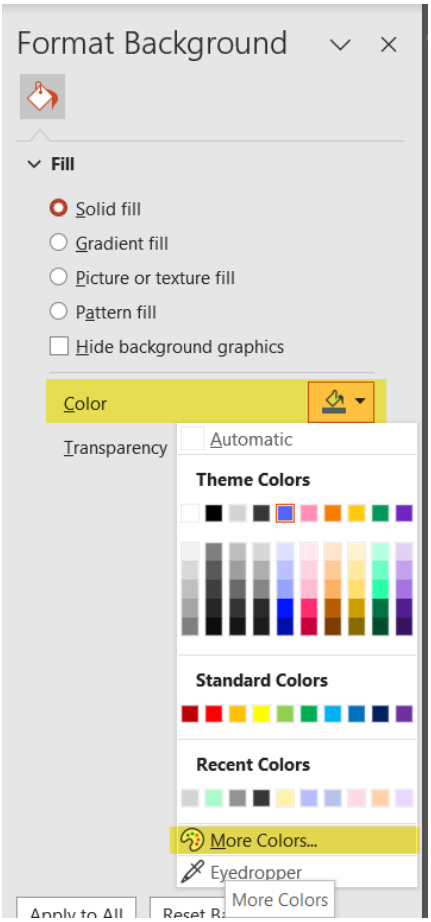
> **NOTE:** The background cannot be transparent. If transparency values are included in the BackgroundColor value, for example, AA IN #AARRGGBB, it is ignored and treated as #RRGGBB (no transparency) and remains opaque.

When using Microsoft PowerPoint to insert rich text, you can find a Hex code on the slide by following these steps:

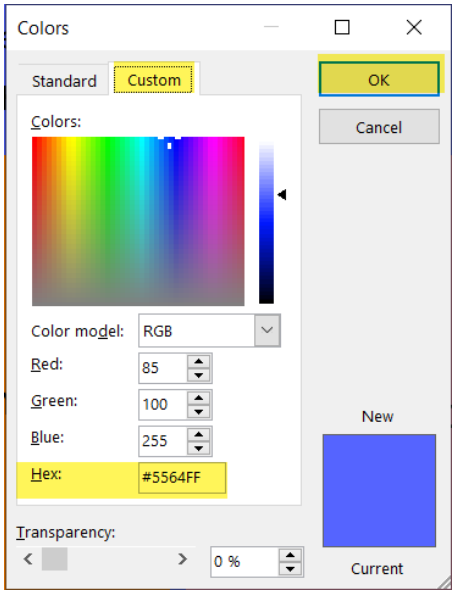
1. Right-click on the slide and select **Format Background**.



2. Next to **Color**, select the **Fill Color (paint icon)** and then **More Colors**.



3. Select the **Custom** tab. The Hex code is located in **Hex**. Enter the code and then select the **OK** button. The hexadecimal code is listed in the Hex field.



Measurement Unit

You can include the optional MeasurementUnit setting to specify the measurement unit used by the PageWidth and PageHeight settings. If the MeasurementUnit setting is blank or not valid, it defaults to Inch.

The following values are permitted for MeasurementUnit:

- **Document**: A document unit corresponds to 1/300 of an inch.
- **Centimeter**
- **Inch**
- **Millimeter**
- **Point**: A point unit corresponds to 1/72 of an inch.

Page Settings

You can include the optional PageWidth, PageHeight, and PageNumber settings.

- **PageWidth**: This affects the width of the image that is generated and wraps the text to fit within the page width. If the PageWidth setting is blank or not valid, it defaults to 8.5 inches.
- **PageHeight**: This affects the height of the image that is generated. If the PageHeight setting is blank or not valid, it defaults to 11 inches.
- **PageNumber**: If the PageNumber setting is blank or not valid, it defaults to the first page.

Borders

You can include the optional IncludeBorders setting to display borders in the resulting image. If the IncludeBorders setting is blank or not valid, it defaults to False.

Insert Rich Text

You can use formatted rich text to embed refreshable content in a document and publish it in various areas.

> **IMPORTANT:** You can copy all the syntax needed to insert rich text by navigating to Object Lookup > Extensible Document Settings. See Extensible Document Settings.

Microsoft PowerPoint

1. Select the **Object Lookup**  icon in OneStream. This window provides all syntax needed to insert content into an extensible document. This icon can be found on the following pages under the **Application** tab:
    - Form Templates
    - Books
    - Cube Views

- Workspaces
- Data Management

2. Select **Extensible Document Settings**. Expand **Insert Content Using Office Image** and then expand **Rich Text**. Select the line of syntax, and select the **Copy to Clipboard** button to copy the first string.
3. In Microsoft PowerPoint, open a new or existing presentation, and insert an image.
4. Right-click on the image, and select the **View Alt Text** option.
5. In the **Alt Text** field, paste the string. The string `{XF}{Application}{RichText}{RichTextContent}` displays in the field. You must update the RichTextContent placeholder text to a specific rich text format or a parameter containing rich text to display.
6. (Optional) Return to **Object Lookup** > **Insert Content Using Office Image** > **Rich Text** and then copy the second string. The second string includes additional formatting options. See Extensible Document Settings.
7. Enter your special formatting changes to use with the rich text. See Special Formatting Changes and Extensible Document Settings.
8. Select **File and Save As**. Select a file name, and add the **.xfdoc** extension to the name. The file name is appended with .xfdoc.pptx.
9. Select the **Save** button.
10. Navigate to OneStream and open **File Explorer**. Select a folder and then upload the extensible file.
11. Right-click on the file and select **Process and Open**. You may need to enter or select a parameter. Microsoft PowerPoint will display the rich text content in place of the original image.

Text Box Component

1. Go to **Application** > **Presentation** > **Workspaces**.
2. Create a Workspace and maintenance unit or select an existing one. See "Create a Workspace" in the *Developer Guide*.
3. Navigate to Components, and select the **Create Dashboard Component**  icon.
4. In the **Create Dashboard Component** dialog box, select **Text Box**, then select the **OK** button.

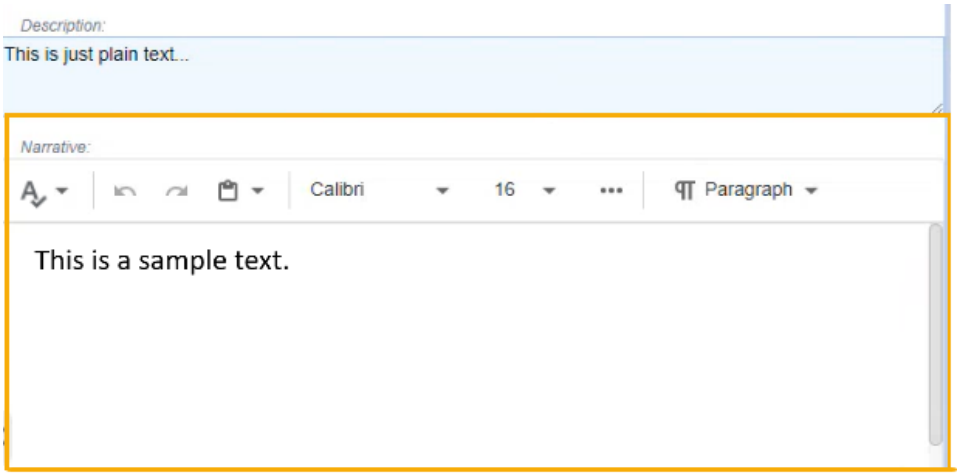
   > **NOTE:** Rich text properties are set in Component Properties > Text Box. See Text Box.

5. Set rich text properties and name the text box. Select the **Save**  icon.
6. Add the Text Box component to a new or existing dashboard within the same Workspace.
7. Select the **View Dashboard**  icon to run your dashboard with the Text Box component.
8. If IsReadOnly is set to (Use Default) or False, you can begin to edit the rich text box.

Modern Browser Experience with Rich Text

You can add formatted rich text to embedded refreshable content. This content can be used for narrative and can be edited and published into a report in the Modern Browser Experience. You can use formatted rich text in the component

You can add formatted text using the formatting toolbar in the Text Box dashboard component in the Modern Browser Experience.



> **NOTE:** The Find and Replace formatting options are not available in Modern Browser Experience.

Supported Rich Text Functionality

Allow Rich Text

When this is set to True, you can format text directly in the Text Box component using the built-in formatting toolbar. You can format elements like font, color, and alignment.

> **IMPORTANT:** You may need to alter the design of your dashboard properties or individual components to display the text box component properly in your dashboard layout.

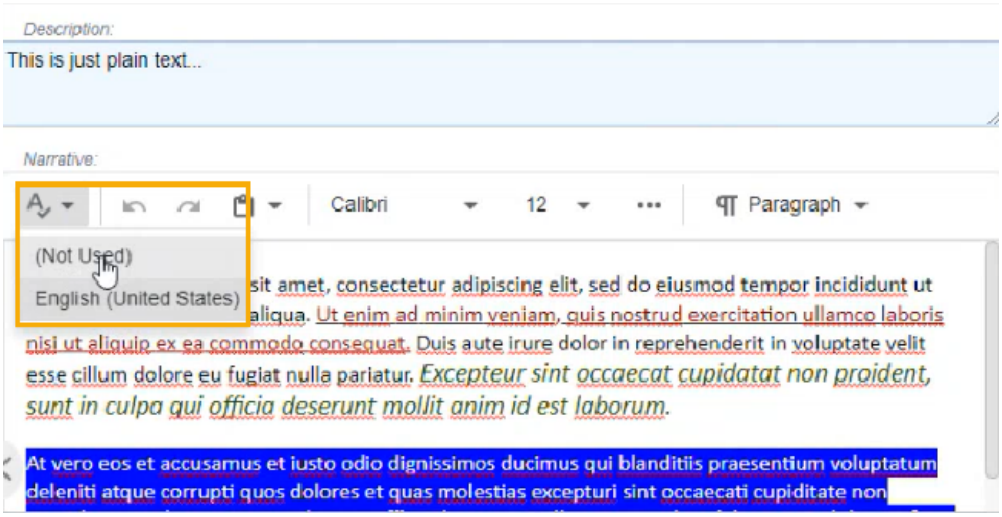
Spell Check

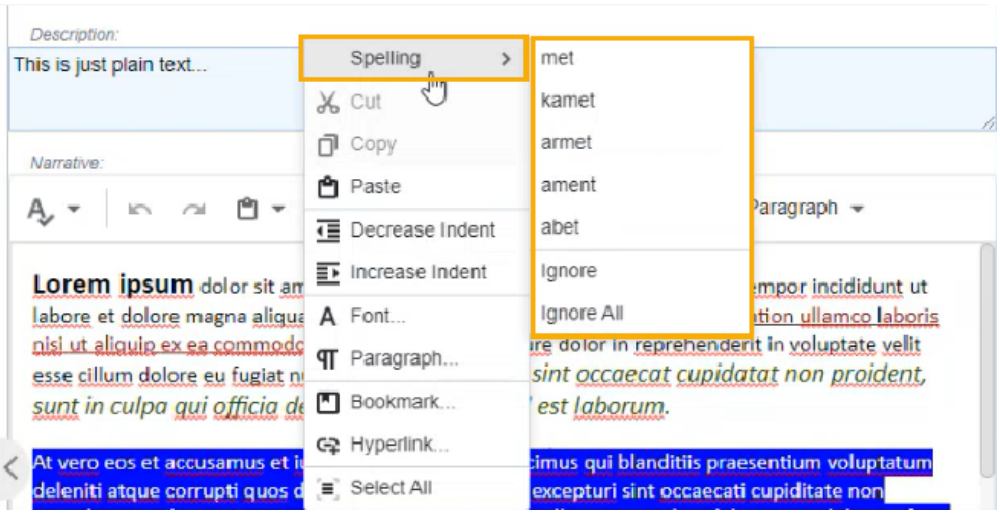
You can perform spell checks with rich text in the Text Box ribbon in Modern Browser Experience. When spell check is enabled, the Spell Check icon is available with the following options:

- (Not Used)
- English (United States)

The default for this property is Use English. When spell check is enabled, you can right-click on a word in Modern Browser Experience and select different options.

> **NOTE:** The limit of word suggestions is seven.

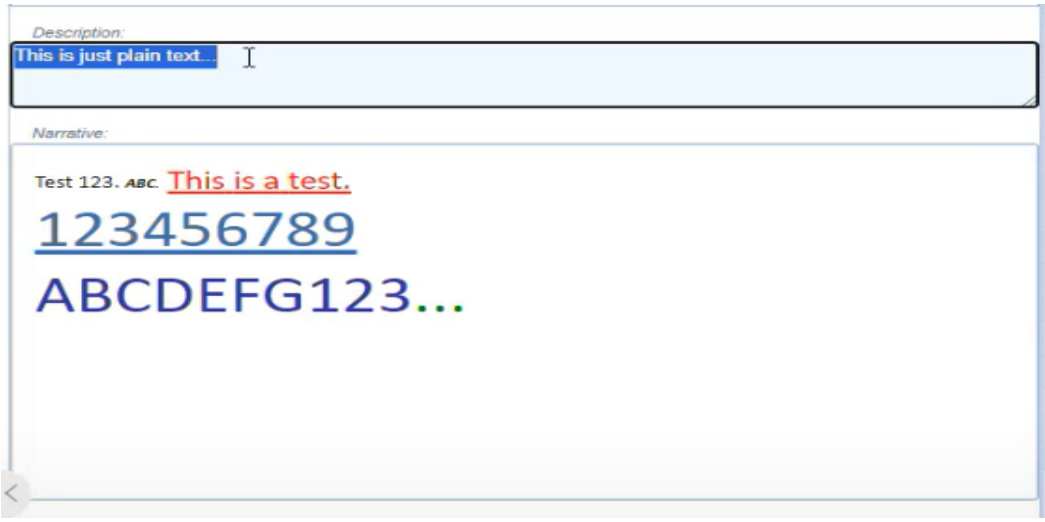




IsReadOnly

This property enables you to set the rich text functionality in the Text Box component to be read-only or editable for select users. When set to True, the text box content is read-only not editable. When set to False or (Use Default), you can edit the text box content.

> **NOTE:** In the following example, IsReadOnly is set to True and no formatting options are available in the rich text box.



Tree View

This provides a graphical illustration of hierarchical data. This is used for relational or hierarchical data (non-Cube data).

Formatting and Action

See Component Formatting and Action Properties and Component Display Format Properties.

SQL Table Editor

This component displays a grid linked to a SQL table, and the contents of the table are shown in the grid. The settings in the object determine whether the grid allows changes to the table data.

Formatting and Action

See Component Formatting and Action Properties and Component Display Format Properties.

SQL Table Editor

Database Location
Select a database location. You can choose from the following options:

- **Application**: Select this if the table to be displayed is located in the application database.
- **Framework**: Select this if the table to be displayed is located in the framework database.
- **External**: Select this if the table to be displayed is located in an external database.

External Database Connection
Select to enable connection to an external database. This property is only enabled if the Database Location property is set to External.

Table Name
The name of the table being displayed in the component.

Table Column Names
A comma-separated list of database table column names you can use in the SQL Select statement. Only the entered columns are displayed in the component. This property is optional.

> **NOTE:** If editing existing data or adding new rows is enabled, any columns that exist in the underlying database table are excluded from the list. This property must have default constraints set in the database table.

Where Clause

The SQL string where clause used to pull data from the table.

Order By Clause

The SQL string to order the results of the table.

> **NOTE:** The Order By Clause property does not allow SQL Functions like YEAR (effectiveDate). When run inside a dashboard, the Table Editor columns allow you to select a heading to change the order to ascending, descending, or no sort. SQL functions cannot be allowed in the ordering.

Filter Mode

The SQL Table Editor enables you to filter results using Filter Mode. This filtering mode is built into the header cell of each filterable column. It is simpler to use than pop-up mode, which is an alternative filtering mode, and uses only one field editor to filter. All filters start as empty until a value is entered into the editor.

Default for Columns Are Visible

When set to True, all columns are hidden unless overridden on an individual column.

Show Title Header

This property enables you to show or hide the entire bar including the items within the bar. When this property is set to False, the Show Title and Show Deselect All Button properties are turned off.

Default For Allow Column Updates

When set to True, the data in the existing table can be modified. When set to False, it cannot be modified.

Show Title

Show or Hide the text in the Title Header.

Show Column Settings Button

When set to True, this shows the Column Settings button, which enables users to reorder and set visibility of columns. When set to False, the Column Settings button is hidden.

Show Deselect All Button

Shows or Hides the Deselect All button, which enables the user to deselect all rows.

Process Selection Changed For Inserts

When both this property and the Allow Inserts property are set to True, they enable the Insert button to process the selection changed event, which is often configured to refresh a portion of the dashboard. When set to False, it will not allow the Insert button to process the selection changed event.

Allow Inserts

When set to True, new rows of data can be added to existing data in the table. When set to False, new rows cannot be added.

Allow Deletes

When set to True, rows of data can be deleted from the existing data in the table. When set to False, rows cannot be deleted.

Retain Table Column Order

When set to True, the column order can be changed through the SQL Table Editor Column Format properties. When set to False, the order cannot be changed.

Read Only Text Color

This field sets the default color to standard black text. The setting can be changed to any color including a system color, for example, XFReadOnlyText.

Column Name for Bound Parameter

Specify the name of the database column to use to change the value of the Bound parameter when a database row is selected. This is used when a SQL Table Editor is being used to affect the display of other components, for example, when showing detailed information for the selected row.

Allow Multiselect

When set to True, the multiple selection of records is enabled by item selection or by using a selection checkbox. The active selected items are passed to the defined Bound parameter field as a comma-delimited list. The Bound parameter format will be: Item1, Item2, Item3. If the 3 values A, B, C,D are selected, where C,D is one value, the resulting bound parameter string is A, B, "C,D". When set to False, only a single Bound parameter can be passed at a time, and the selection boxes are deactivated from the user interface. See SQL Table Editor Multiselect.

Rows Per Page

Specify how many rows to display before a page break. This is a numeric value between 1 and 3000. By default, this property is set to -1, which turns off the property and uses the defined user preference property Grid Rows Per Page. Any value greater than 3000 results in a maximum value of 3000.

Save State

When set to True, user settings on the component are retained. User preferences for columns saved are for Order, Visibility, Filtering, Freeze Bar, Sorting, and Widths.

> **NOTE:** To reset Save State back to default, right-click on the dashboard to enable the Reset State to return back to its Component Properties settings.

Vary Save State By

Use this property to apply Vary Save State By to Workflow Profile settings and scenario. The current Save State elements have the control state tied to the current Workflow Profile and Workflow Scenario settings. If you select False or Use Default, Vary Save State By will not apply. When Vary Save State By is enabled, the related component has the additional reset option of Reset All States, which can be used to clear the user's Save States across all the Vary Save State By parameters.

Show Data Manipulation Buttons

Use this property to show or hide the following buttons: Insert Rows, Delete Rows, Cancel All Changes Since Last Save, and Save.

Show Add Row Button

Hide or show the Add Row button.

Show Remove Row Button

Hide or show the Remove Row button.

Show Cancel Button

Hide or show the Cancel button.

Show Save Button

Hide or show the Save button.

Create Table If Necessary

The option to create the table if it does not already exist.

Table Creation Script

The SQL script to create the new table.

Save Data Server Task

You can choose from the following Server Task options:

- **No Task**: There is not a performed task for a SQL table save data event.
- **Execute Dashboard Extender Business Rule**: This will use a Dashboard Extender business rule to perform a save data event.

Save Data Server Task Arguments
Enter the arguments required by the server task, for example, {Business Rule Name}{Function Name}{Optional Name-Value Pairs}.

> **TIP:** The data will wrap on string spacing. If there are no blank spaces, the column data will not wrap. Additionally, keyed fields cannot be wrapped.

DataFormatString
Specify a number/date format to the data in the column. For example, mm/dd/yyyy will return the current month/day/year using a slash. mm-dd-yyyy will return the month-day-year using a dash. N0 will return a number without a decimal point, and #,###,0 will return a number without a decimal point and a comma depicting the thousandth place. See "General Properties" in the *Application Tools Guide* for more examples of number formats.

Width
This specifies the default column width to be displayed.

SQL Table Editor Column Format

ColumnName
The column name from the table where the selected formatting is to be applied. The specified name must match the column name from the table exactly.

Description
The column description to be displayed. By default, the column name from the table is displayed.

Column Display Type
You can choose from the following options:

- **Standard**: This displays the default data in the column.
- **OneStreamClientImage**: This replaces the data in the column with the associated image.
- **WorkflowStatusImage**: This replaces the data in the column with the associated workflow status indicator image.

IsVisible
This setting overrides the default Columns Visible setting. The options are True, False, or Use Default

AllowUpdates
This a setting overrides the Default For Allow Column Updates setting for each individual column. When set to True, users can modify data displayed in the grid. The Use Default setting uses the Default For Allow Column Updates setting.

ParameterName
The assigned parameter name to be used to store the parameter value from the specified column.

DefaultValue
The default value to be entered when allowing new records to be added. This helps ensure that blank cells that are not valid are not created.

IsMultilineText
When set to True, this enables columns to display data on multiple rows if the column is not wide enough to display the full value.

Primary Key Behavior

If the database table has a primary key, the SQL Table Editor component handles that column differently than the other columns, regardless of the component's other property settings.

- The Unique Row ID column is always included in the grid results, even if it is not listed in Table Column Names.
- A primary key column cannot be edited on existing rows. However, the primary key cell is editable on the insert row.
- If the primary key value is calculated in the database, for example, a uniqueidentifier column with a default of newid(), the calculation runs when the row is saved. The value entered in the grid is replaced with the value generated by the database. If a save fails, the inserted row remains and a notification displays explaining the adjustment needed.

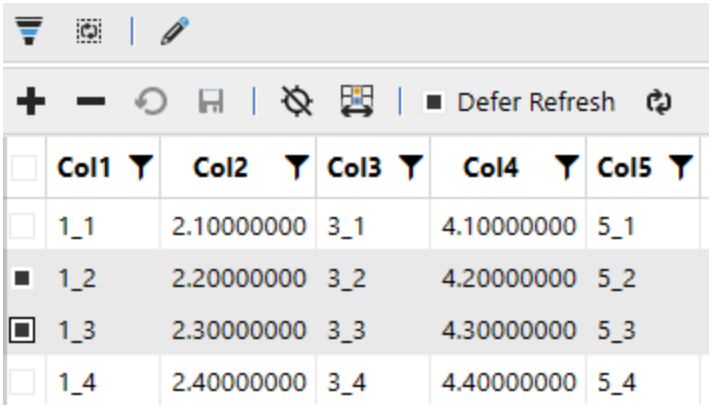
SQL Table Editor Multiselect

Enabling the Allow Multiselect property for the SQL Table Editor component expands the functionality of the component for use in interactive dashboards.

- The Allow Multiselect property generates the Bound parameter as a comma-delimited list.
- Selection Methods can be performed with the Select All checkbox. You can select a range using the Shift key and select by item using Ctrl-Click.
- Deselecting can be done by item or using the Deselect All icon.
- You cannot apply a Column Filter after selections are activated. Applying a column filter will clear any existing selected records. The column must be first filtered. Then, selections can be performed.
- The Defer Refresh button can be used to manually control whether tasks are run as defined in the component's Actions properties. This may provide better performance within complex dashboards.

> **NOTE:** Defer Refresh remains selected after the Refresh button is selected in

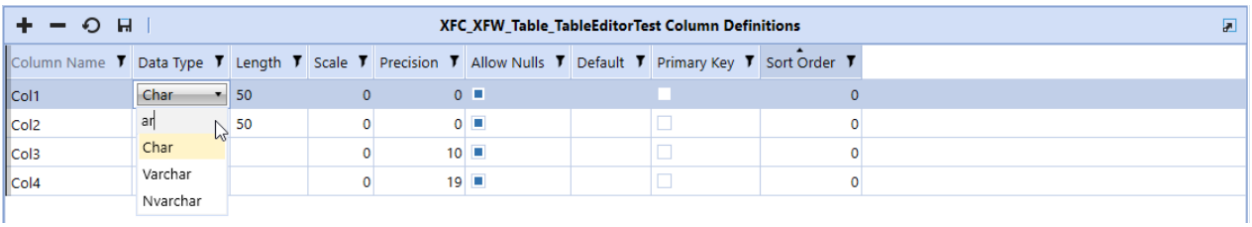
SQL Table Editor.



NOTE: Multiselect retains selected items after column filtering and sorting by using a table's primary key columns. If the referenced table does not contain a primary key column, the selection is not retained across pages or after filtering or sorting columns.

SQL Table Editor Searchable Drop-Down Menu

Searchable drop-down menus enable you to find the correct item in the list quickly and easily. As you type, search options are narrowed down to those that contain the entered text, so you no longer have to scroll through a long list.



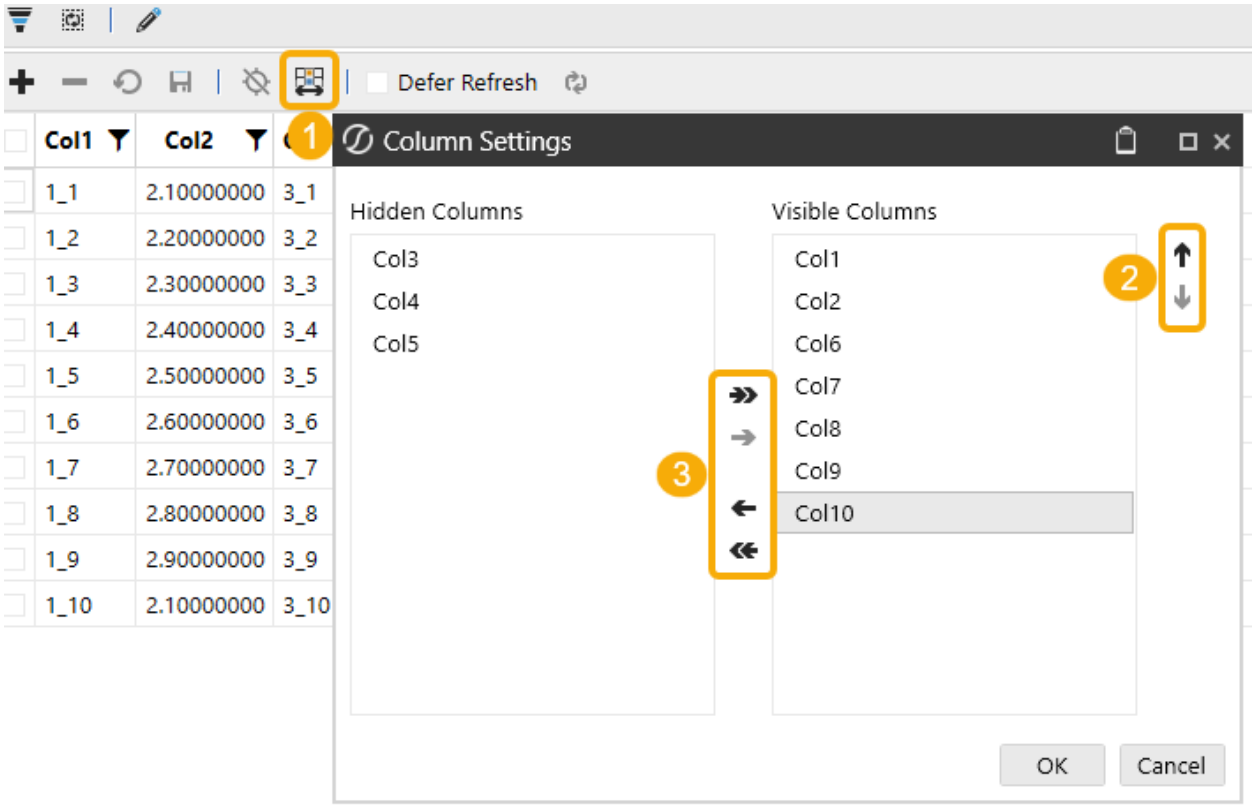
> **NOTE:** To use this search feature, the component must be configured so that you can edit existing data or add new rows. Columns must be configured with a parameter type of Delimited List, Member List, or Bound List, which is what configures a column as a standard drop-down menu.

Enable the Search

1. Navigate to the SQL Table Editor **Component Properties** tab.
2. Enable the search feature.
    - For all columns that are configured with a list parameter:
        a. In the **SQL Table Editor** section, for the **Default For Show Search** property, select **True**.
        b. Select the **Save** icon.
    - For specific columns only:
        a. In the **SQL Table Editor Column Format** section, locate the column to enable the search feature for and select the ellipsis ... .
        b. In the **Column Format** dialog box, set the **ShowSearch** property to **True**.
        c. Select the **OK** button.
        d. Select the **Save** icon.

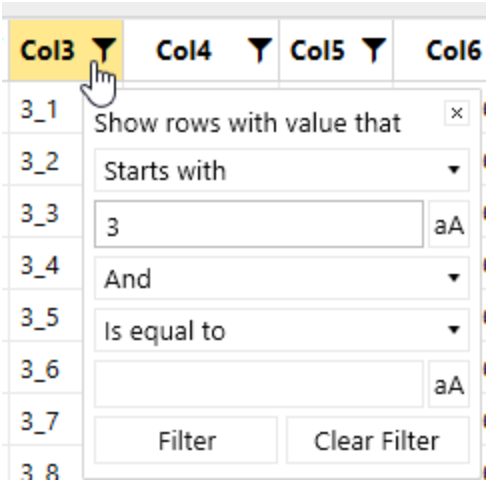
SQL Table Editor Save State User Preferences

Use the Save State property to retain your user preferences for the SQL Table Editor display. Your preferences for column order and visible or hidden columns are saved if Saved State is set to True.

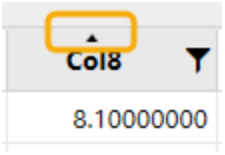


1. The Column Settings icon enables you to reorder columns.
2. Use the up and down arrow icons to set column order.
3. Use the left and right arrow icons to assign visibility.

Column filters can be set by the user and are preserved if the Saved State is set to True.



However, the column sort order will not be preserved once the SQL Pivot Grid is deselected or the dashboard is closed.



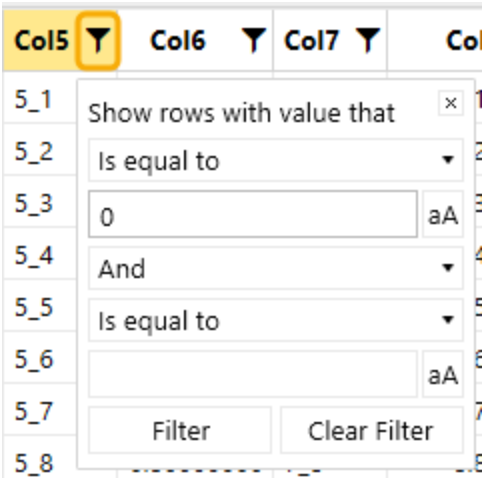
Save State for Freeze Bar

When the Save State property is set to True, the freeze bar retains its position if you drag it to a certain column.

SQL Table Editor Filtering

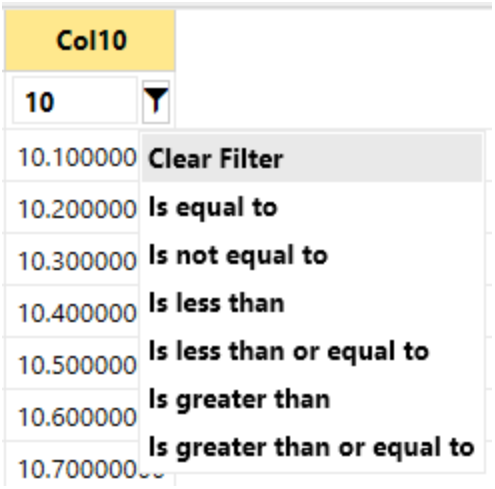
The SQL Table Editor component enables you to filter results using the following methods:

- **Popup Mode**: This is the default filtering mode. With this mode, you can activate filtering by selecting the filter icon in the column header, which opens a dialog box.



  The dialog box contains fields that enable you to create filtering criteria, for example, value is greater than 18 and is less than 60. The filter has multiple parts: the top field is a combo box and the lower field is the field filter editor. In the top field, you can use filter operators like Is equal to or Is less than that change based on the data type of the column. The field filter editor provides filtering criteria based on the data type of the column. For example, if you have a date column, you can use a date/time picker control as the filtering criteria.

- **Filter Row Mode**: This filtering mode is built into the header cell of each filterable column. It is simpler to use than the dialog box and uses only one field editor to filter. All filters start as empty until a value is entered into the editor. Select the filter icon to display a drop-down list of all operators available to the data type of the column. If you select Clear Filter from the list, it will clear and reset the column filter.



You can configure the Filter Mode property of the SQL Table Editor to use either of the modes.

1. Go to the SQL Table Editor **Component Properties** tab.
2. In the **SQL Table Editor** section, set the **Filter Mode** property to either **Filter Row** or **Popup**. The default setting is Popup.

> **NOTE:** Consolidated column filtering is not supported for certain data types. Applying filters on these data types will not yield consolidated filtered results. The following data types are unsupported: Binary, char, date, datetime2, datetimeoffset, float, geography, geometry, hierarchyid, image, filestream, money, nchar, ntext, numeric, nvarchar, real, rowversion, smalldatetime, smallmoney, sql_variant, text, time, timestamp, tinyint, varbinary, and xml.

Filtering Columns Based on Values from Another Column

You can base a column's header filter on values from another column. This includes any column that is in the dataset, even if the column is hidden. This is useful for name-value pair parameter columns as long as the values and names are pulled from individual columns.

Set up and configure the SQL Table Editor component:

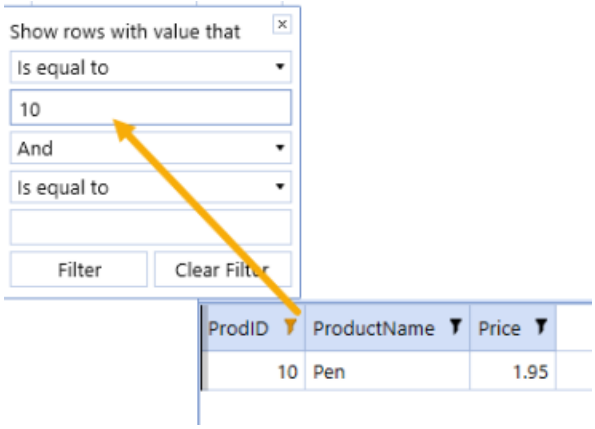
1. In the Workspace, create and configure an SQL Table Editor component according to your needs. See SQL Table Editor.
2. Identify the column to filter along with the column the filter is based on.
3. In the SQL Table Editor component, configure the column format for the column to filter by entering the name of the column in the **FilterByColumnName** property.

Example

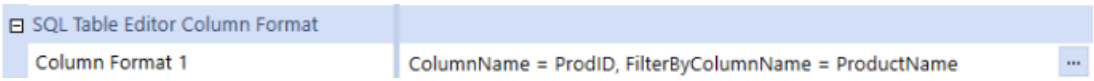
Begin with an SQL Table Editor grid. The dataset can come from a SQL view or a SQL table.

ProdID	ProductName	Price
10	Pen	1.95
20	Mug	5.99
30	Tshirt	12.00
40	Speaker	35.99

When filtering data using a column header filter, the default behavior is to filter by the value stored in that column. For example, filtering the ProdID column on a value of 10 returns one row where ProdID equals 10.



To change the ProdID column filter to base it on ProductName, configure the column format for ProdID by setting FilterByColumnName to ProductName.



Column filtering for column ProdID is now based on values from the ProductName column.

SQL Table Editor Modify Table Data Using a View

You can modify the data of an underlying base table through a SQL View. A SQL View is a virtual table where column and row contents are defined by a SQL query. The SQL Table Editor displays data from the virtual table, but you can change the data of the underlying base table. This provides more flexibility and control over how you present and manipulate data in the SQL Table Editor.

Use the following rules to modify data through a View:

- Use Views to create a view of the data tables in the database. You can add calculated columns to your View.
- You can only update data in a single underlying base table. For example, if you create a View that combines data from multiple tables, you can only update data from one of those tables.
- You need the correct permissions to the underlying base table.

Configure a SQL Table Editor Component

1. Inside the Workspace, create a SQL Table Editor component.
2. In the **Table Name** property, specify the underlying base table that contains the editable data.
3. In the **View Name** property, enter a name for the view.

4. Identify columns in the view that do not exist in the underlying table. These columns are excluded from the Insert and Update statements that the SQL Table Editor generates when updating the base table.
5. Configure a column format for each column in the view that does not exist in the base table.
6. Set the **IsFromTable** property to **False**.

Example

This example has two tables. The first is a Products table.

ProdID	ProductName	Price
10	Pen	1.95
20	Mug	5.99
30	Tshirt	12.00
40	Speaker	35.99

The second is an Orders table.

OrderID	prodID	Quantity
7800	10	5
7800	30	1
7800	40	2
9991	20	4

The tables are linked by the ProdID value. Next, a simple view called OrderDetails is created to show products specific to each order.

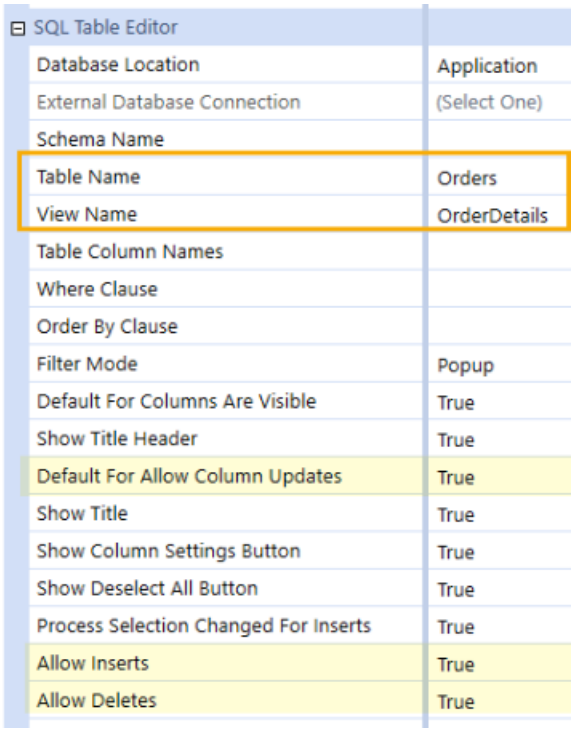
```
SELECT dbo.Orders.OrderID, dbo.Orders.prodID, dbo.Orders.Quantity, dbo.Products.productName,
    dbo.Products.Price
FROM dbo.Orders
INNER JOIN dbo.Products
ON dbo.Orders.prodID = dbo.Products.prodID
```

This example produces the following view:

OrderID	prodID	Quantity	productName	Price
7800	10	5	Pen	1.95
7800	30	15	Tshirt	12.00
7800	40	2	Speaker	35.99
9991	20	4	Mug	5.99

At this point, you can expose the view to users so they can update order quantities or add additional orders. This data is stored in the underlying base table, Orders.

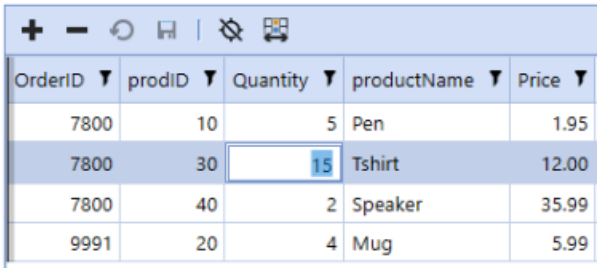
In the SQL Table Editor, Orders can be set as the Table Name and OrderDetails as the View Name



Identify columns in the view that do not exist in the underlying base table. In this example, that would be productName and Price. For both columns, configure a column format where IsFromTable is set to False. You can also make the columns read-only by setting AllowUpdates to False.

SQL Table Editor Column Format	
Column Format 1	ColumnName = productName, IsFromTable = False, AllowUpdates = False
Column Format 2	ColumnName = Price, IsFromTable = False, AllowUpdates = False

You can now see all columns of the view while modifying orders.

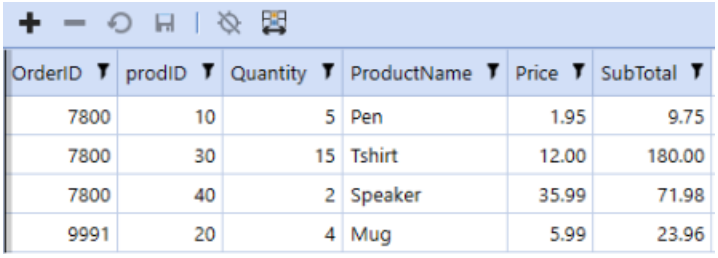


OrderID	prodID	Quantity	productName	Price
7800	10	5	Pen	1.95
7800	30	15	Tshirt	12.00
7800	40	2	Speaker	35.99
9991	20	4	Mug	5.99

You can also add calculated columns to views. These columns are virtual and do not exist in the database They derive their values dynamically when the view is run.

In this example, a column is added that multiplies Quantity times Price. The View can also be updated with an expression. The expression can be given any name, for example, SubTotal.

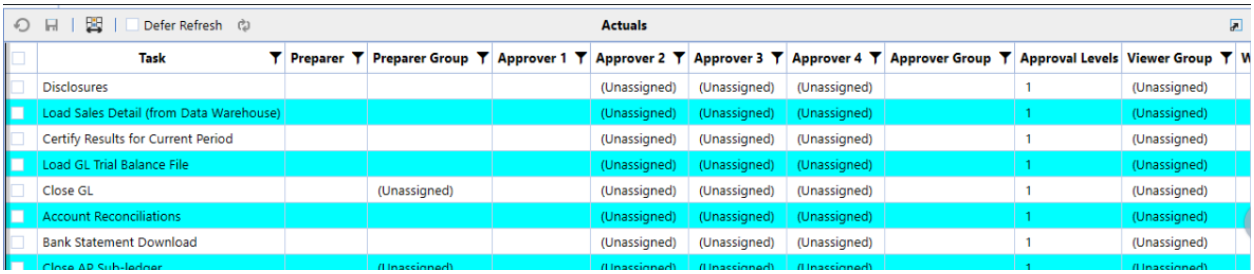
```
SELECT dbo.Orders.OrderID, dbo.Orders.prodID, dbo.Orders.Quantity, dbo.Products.productName,
    dbo.Products.Price,
(dbo.Orders.Quantity*dbo.Products.Price) AS SubTotal
FROM dbo.Orders
INNER JOIN dbo.Products
ON dbo.Orders.prodID = dbo.Products.prodID
```



OrderID	prodID	Quantity	ProductName	Price	SubTotal
7800	10	5	Pen	1.95	9.75
7800	30	15	Tshirt	12.00	180.00
7800	40	2	Speaker	35.99	71.98
9991	20	4	Mug	5.99	23.96

SQL Table Editor Row Background Colors

You can apply a background color to all rows or alternate rows using the BackgroundColor property and the AlternateRowBackgroundColor property. This makes it easier to distinguish between rows of data at a glance and can significantly reduce eye strain.



To configure these properties, complete the following steps:

1. Navigate to the SQL Table Editor **Component Properties** tab.
2. Go to **Formatting** > **Display Format** and select the ellipsis ... .
3. In the **Display Format** dialog box, in the **BackgroundColor** field, select a color.
4. In the **AlternateRowBackgroundColor** field, select a color.
5. Select the **OK** button.
6. Select the **Save** icon.

Web Content

This is used to display a URL or file embedded in a dashboard.

> **NOTE:** This component uses Microsoft Edge WebView2 Runtime to embed the web content. It must be installed on the client device. If the client device does not have it installed, you will receive an error when accessing a dashboard that is configured with a Web Content component.

Web Content

Web Content Type
Select a Web Content Type. You can choose from the following options:

- **URL**: The URL to a Web page. This can be internal or external.
- **Workspace File**: Display a file stored in the Workspace maintenance unit File section.
- **Application Database File**: Display a file from the Application Database share.
- **System Database File**: Display a file from the System Database share.
- **File Share File**: Display a file from the File Share.

URL Or Item Name
URL or file to display in the control.

Open Content in New Tab (Web Only)
When set to True, Web content displays in a new tab. When set to False, Web content displays embedded in the dashboard.

> **NOTE:** Some Web content may not function in the Modern Browser Experience.

Show Borders
When set to True, the border displays around the component. When set to False, the border around the component is removed. This property defaults to True.

Grid View

Show Header
When set to True, a header derived from the component's name or description displays.

Show Toggle Size Button
When set to True, the toggle button displays and users can select the size of the component at runtime. When set to False, the toggle button is hidden.

Component Formatting and Action Properties

Formatting

Text
The text displayed with the component. This is an optional field. If left blank, the Text field defaults to display the component name. To clear the default name from displaying, enter **|SPACE|**.

Tool Tip
The text displayed when hovering over the component.

Display Format
The formatting assigned to the component. See Component Display Format Properties.

Action

Bound Parameter
This is the name of the parameter the component will represent. The component uses the parameter to get its list of options. The selection specifies a value for the parameter when it is referenced elsewhere through piped variable naming.

Select the ellipsis to open the **Object Lookup** dialog box. Either search the parameter name, or use the navigation tree to expand Workspace Maintenance Units to find the parameter.

> **NOTE:** Piped variable names are not available in this setting.

Apply Selected Value to Current Dashboard
When set to True, the Bound parameter on the current dashboard uses the current updated value or the value before the update. When set to False, the current dashboard is not updated.

Save Action

Selection Changed Save Action
The save action that occurs when the component is used. You can choose from the following options:

- **No Action**: When the components are changed, changes are not saved.
- **Save Data for Components**: This saves the data modified by the components specified in the Selection Changed Save Arguments property.
- **Prompt to Save Data for Components**: This asks the user if the modified data should be saved or not for a defined list of components.
- **Save Data for all Components**: This saves all data modified by each component assigned to the dashboard. These components do not have to be specified in the Selection Changed Save Arguments property.
- **Prompt to Save Data for All Components**: This asks the user if the modified data on any included component should be saved.

For Save Files properties specific to the Button component, see Button.

POV Action

Selection Changed POV Action

The changes that occur to a user's POV. You can choose from the following options:

- **No Action**: When components are changed, no change is made to the user's POV.
- **Change POV**: This setting changes the user's POV to a specified POV when components are changed.
- **Change Workflow**: This changes the user's workflow to a specified workflow when components are changed.
- **Change POV and Workflow**: This changes the workflow and POV to a specified value when components are changed.

Selection Changed POV Arguments

Enter the arguments for the selected POV action. See the following examples:

- **Change POV**: Cube=Corporate, Entity=|!MyEntityParam!|, Scenario=Actual.
- **Change Workflow**: Cube=Corporate, Entity=|!MyEntityParam!|, Scenario=Actual
- **Change Workflow**: Enter a value for WFProfile, WFScenario, and WFTime. WFProfile=|!MyWFProfileParam!|, WFScenario=Actual, WFTime=2012M1

> **NOTE:** If the Selection Changed POV Action is set to No Action, no change occurs.

Server Task

Selection Changed Server Task

The server task that is completed. You can choose from the following options:

- **No Task**: When components are changed, no server task is run.
- **Execute Dashboard Extender Business Rule**: Runs the assigned business rule when components are changed. For a long-running business rule, it can run on a specific server type, such as a Consolidation server type. You can choose from the following types:
    - Execute Dashboard Extender Business Rule (General Server)
    - Execute Dashboard Extender Business Rule (Stage Server)
    - Execute Dashboard Extender Business Rule (Consolidation Server)
    - Execute Dashboard Extender Business Rule (Data Management Server)
- **Execute Finance Custom Calculate Business Rule**: Runs a single year custom calculation from a Finance business rule when components are changed. The calculation is performed on the Cube View assigned to the dashboard data adapter, and it calculates time based on the Cube View Time member set in the Point of View. This is considered a partial calculation and does not store the calculated data or run the calculation during a consolidation. Running a custom calculation from a dashboard impacts calculation status for the affected Data Unit even if the data does not change. A calculated dimension member can call this business rule to do a full stored calculation. See the *Understanding the Financial Model Guide* for an example of this calculation logic.
- **Execute Data Management Sequence**: Runs the assigned data management sequence when components are changed.
- **Calculate**: Runs a Calculation when components are changed.
- **Force Calculate**: Runs a Force Calculate when components are changed.
- **Calculate with Logging**: Runs a Calculate with Logging when components are changed.
- **Force Calculate with Logging**: Runs a Force Calculate with Logging when components are changed.

- **Translate**: Runs a Translate when components are changed.
- **Force Translate**: Runs Force Translate when components are changed.
- **Translate with Logging**: Runs Translate with Logging when components are changed.
- **Force Translate with Logging**: Runs Force Translate with Logging when components are changed.
- **Consolidate**: Runs Consolidate when components are changed.
- **Force Consolidate**: Runs Force Consolidate when components are changed.
- **Consolidate with Logging**: Runs Consolidate with Logging when components are changed.
- **Force Consolidate with Logging**: Runs Force Consolidate with Logging when components are changed.

For more details on these calculation types, see "Steps" in the *Application Tools Guide*.

Selection Changed Server Task Arguments

Enter the components the Server Task action applies to. See the following examples:

- **Dashboard Extender Business Rule**: {Business Rule Name}{Function Name}{Optional Name-Value Pairs}.
- **Finance Custom Calculate Business Rule**: {Business Rule Name}{Function Name}{Optional Name-Value Pairs}.

> **NOTE:** Name-value pairs can have settings for cube, parent, and any dimension type. They can also have a setting for any time member filter, including CurrentPeriod, AllPriorInYearInclusive, or AllInYear. They can have any other Name-Value Pair for use by the business rule since they are passed into the rule using the CustomCalculateArgs object.

- **Data Management Sequence Example**: {Sequence Name}{Optional Name-Value Pairs}.
- **Calculation, Translation, or Consolidation**: {Data Unit Member Script} {E#|!EntityParam!|:C#Local:S#Actual:T#|!TimeParam!|}.

User Interface Action

Selection Changed User Interface Action

The user interface action that is completed. You can choose from the following options:

- **No Action**: When components are changed, no user interface action is run.
- **Redraw**: This repaints the screen without making a connection to the database acquiring updated data.
- **Refresh**: This refreshes the dashboard specified in the Dashboards to Redraw property, which makes a connection back to the database acquiring all new data.
- **Close Dialog**: This closes the dialog box after performing a specified action in a dashboard.
- **Close Dialog As OK**: This closes the dialog box after the user selects the OK button.

  > **NOTE:** The refresh of the parent dashboard is based on performing a specified action set up in a dashboard. For example, if a dashboard is set to Open Dialog With No Buttons and Refresh, the Close Dialog As OK will refresh the parent dashboard and run the user interface (UI) action.

- **Close Dialog As Cancel**: This closes the dialog box after the user selects the Cancel button.
- **Close All Dialogs**: This closes all open dialog boxes after performing a specified action in a dashboard.
- **Open Dialog**: This opens a dashboard in a dialog box.

- **Open Dialog with No Buttons**: Performs an open dialog action and does not display the Close button.
- **Open Dialog with No Buttons and Redraw**: Performs an open dialog and enables the user to apply changes and redraw without displaying the Close button.
- **Open Dialog with No Buttons and Refresh**: Performs both an open dialog and refresh action without displaying the Close button.
- **Open Dialog and Redraw**: Performs both an open dialog and redraw action.
- **Open Dialog and Refresh**: Performs both an open dialog and refresh action.
- **Open Dialog, Apply Changes, and Redraw**: Performs an open dialog and enables the user to apply changes and redraw.
- **Open Dialog, Apply Changes, and Refresh**: Performs an open dialog and enables the user to apply changes and refresh.
- **Open Dialog, Apply Changes, and Redraw if OK**: Performs an open dialog and enables the user to apply changes and redraws after the user selects the OK button.
- **Open Dialog, Apply Changes, and Refresh if OK**: Performs an open dialog and enables the user to apply changes and refreshes after the user selects the OK button.

Dashboards to Redraw

A comma-separated list of embedded dashboards to redraw when the user changes the selection in the component.

Dashboards to Show

A comma-separated list of embedded dashboards to make visible when the user changes the selection in the component. This is typically used when multiple components are needed to collect settings to display an Embedded dashboard. After the last selection is made, that component shows the previously hidden Embedded dashboard.

Dashboard to Hide

A comma-separated list of embedded dashboards to hide when components are changed.

Dashboards to Open in Dialog

A single dashboard to open in a dialog box when the button is selected.

Dialog Initial Parameter Values

Enter a comma-separated list of name-value pairs to specify initial parameter values for the opened dialog box.

> **Example:** DlgParam1=Value1, DlgParam2=[Value 2]

Dialog Input Parameter Map

Enter a comma-separated list of name-value pairs to map parameter names used by this component's dashboard to parameter names used by the opened dialog box.

> **Example:** DlgParamName1 = This ParamName1, DlgParamName2=ThisParamName2. When the dialog box opens, the names of the parameters adjust to accommodate the dialog box.

Dialog Output Parameter Map

Enter a comma-separated list of name-value pairs to map parameter names modified by the closed dialog to parameter names used by this component's dashboard.

> **Example:** ThisParamName1=DlgParamName1, ThisParamName2=DlgParamName2. When the dialog box closes, the names of the parameters adjust to accommodate this component's dashboard.

Navigation Action

Selection Changed Navigation Action.
The navigation action that is completed. You can choose from the following options:

- **No Action**: No navigation action is performed when components are changed.
- **Open File**: When components are changed by a user, a new file opens. This is specified in the Selection Changed Navigation Arguments field.
- **Open Page**: When components are changed by a user, a new page opens. This is specified in the Selection Changed Navigation Arguments field.
- **Open Web Site**: When components are changed by the user, a website opens in a new browser. This is specified in the Selection Changed Navigation Arguments.

Selection Changed Navigation Arguments
Enter the arguments for the selected navigation. See the following examples:

- **Open Page**: Specify the page name followed by a colon and the page type's arguments separated by two ampersands: XFPage=Dashboard:NameofDashboard or XFPage=Workflow:NameofWorkflow.
- **Open Page in New Tab**: An option in the Open Page Navigation type that allows dashboards launched from within a dashboard to launch into a new tab. Valid values are True and False: XFPage=Dashboard:MyDashboardName, OpenInNewXFPage=True.
- **Open File or Web Site**: Specify the file source type, for example, Dashboard, Application, System, FileShare, or URL, and specify the file name. Also, use PinNavName=True/False or PinPOVName=True/False to open or close the Navigation or POV panes: FileSourceType=FileShare, FullFileName=Document/Public/NameofFile, OpeninXFPageIfPossible=False.

> **NOTE:** Select the ellipsis ... button to open the **Object Lookup** dialog box. Either search the argument string, or use the navigation tree to expand Workspace maintenance units. After you select the argument string, select the **OK** button to enter the correct string in the field.

Component Display Format Properties

The following properties are available in the Display Format window for components. To open Display Format properties, in the Display Format field, select the ellipsis ... . Not all properties in this section are available for all components.

For information about parameters, see "Filter Runtime Data and Reduce Maintenance with Parameters" in the *Understanding the Financial Model Guide*.

General

Custom Parameters

Select the ellipsis ... to select and assign a custom parameter, such as a Cube View style to the Cube View header. You can use parameters from the following locations: Default Workspace and Non-Default maintenance unit or Non-Default Workspace and maintenance unit. You can enter the name of a parameter that contains a string specifying Display Format settings.

Position

IsEnabled

When set to True or (Use Default), the component is enabled. When set to False, the component remains visible but cannot be selected.

> **NOTE:** The IsEnabled property is only available for these components: Button, Combo Box, Check Box, List Box, and Radio Button Group.

IsVisible

This determines if the component is visible on the dashboard. The settings are True, False, or Use Default.

ShowSearch

This adjusts the behavior of the Combo Box component. When set to True, the Combo Box retains typed text and auto-complete is enabled. When set to False, typed text is not retained and auto-complete is turned off.

HorizontalAlignment

This determines how the component is aligned horizontally on the area allocated in the dashboard. You can choose from the following options:

- **(Use Default)**: Select this to use the default setting.
- **Left**: The component is aligned on the left of the allocated area.
- **Center**: The component is aligned in the center of the allocated area.
- **Right**: The component is aligned on the right of the allocated area.
- **Stretch**: The component is stretched to fill the allocated area.

VerticalAlignment

This determines how the component is aligned vertically on the area allocated in the dashboard. You can choose from the following options:

- **(Use Default)**: Select this to use the default setting.
- **Top**: The component is aligned on the top of the allocated area.
- **Center**: The component is aligned in the center of the allocated area.

- **Bottom**: The component is aligned on the bottom of the allocated area.
- **Stretch**: The component is stretched to fill the allocated area.

Width
The width of the component.

Height
The height of the component.

MarginLeft, Top, Right, and Bottom
The spacing used on the sides of the component.

Borders

BorderThickness
The thickness of the border used when displaying the component.

RoundedBorders
When set to True, the component's borders are rounded. When set to False or (Use Default), the component's borders are squared.

> **NOTE:** This property is only available for the Button component.

Font

FontFamily
The font type used on the component, for example, Ariel or Times Roman.

FontSize
The font size used for the component text.

> **NOTE:** When configuring the Grid View component for the Modern Browser Experience, the maximum FontSize to be displayed is 18.

Bold
When set to True, the font is bold.

Italic
When set to True, the font is italicized.

Color

These properties determine the color of the text displayed in the component.

TextColor
Choose a color for the component text from the drop-down menu.

> **NOTE:** When configuring the SQL Table Editor component, Read Only Text Color takes precedence over TextColor for Read Only columns.

BackgroundColor
Choose a color for the text background from the drop-down menu.

AlternateRowBackgroundColor
Choose an alternate row background color from the drop-down menu.

BorderColor
Choose a color for the border of the component from the drop-down menu.

HoverColor
Choose a hover color for a component from the drop-down menu.

Image

HorizontalContentAlignment
This determines the horizontal alignment of the image's content on the area allocated in the baseboard.

VerticalContentAlignment
This determines the vertical alignment of the image's content on the area allocated in the baseboard.

ImageStretch
This setting allows the image to be resized. You can choose from the following options:

- **None**: This does not resize the image.
- **Fill**: The image is stretched to fill the component.
- **Uniform**: This preserves the aspect ratio of the image.
- **UniformToFill**: This scales the source image to fit within the bounds of the image object and keeps the source image centered within the image object.

ImageCropLeft, Top, Width, and Height
Use these settings to crop and position the image to align the content appropriately.

Label

Position

LabelPosition
The location where the control label is placed. You can choose from the following options:

- **(Use Default)**: Use the default label display location.
- **None**: This does not display a label for the component.
- **Left**: This displays the label to the left of the component.
- **Top**: This displays the label above the component.
- **Right**: This displays the label to the right of the component.
- **Button**: This displays the label below the component.

LabelTextHorizontalAlignment

If your LabelPosition is set to Top or Bottom, this property enables you to configure horizontal text alignment for the label text. You can choose from the following options:

- **(Use Default)**: The label text is not impacted. The default label text alignment is left.
- **Left**: The label text is left-justified.
- **Center**: The label text is centered.
- **Right**: The label text is right-justified.

> **NOTE:** This property only applies to the Button component.

Font

LabelFontFamily

The font type to use on the label, for example, Ariel or Times Roman.

LabelFontSize

The font size to use on the label.

LabelBold

When set to True, the font is bold.

LabelItalic

When set to True, the font is italicized.

LabelUnderline

If set to True, the label text is underlined. The (Use Default) setting behaves the same as False. The underline color follows the label's text color settings. If UseTextWrapping property is set to True, the underline extends across all wrapped lines of the label text.

Colors

LabelZeroOffsetForFormatting
This holds a number with the default value of 0.0. When a label is used to display a number, and the number is greater than the number in this field, it is displayed using the color associated with LabelTextColor. Otherwise, the number is displayed using the color associated with LabelNegativeTextColor.

LabelTextColor
Choose a label text color from the drop-down menu.

LabelNegativeTextColor
Choose a color for the label's negative text from the drop-down menu.

Data Adapters

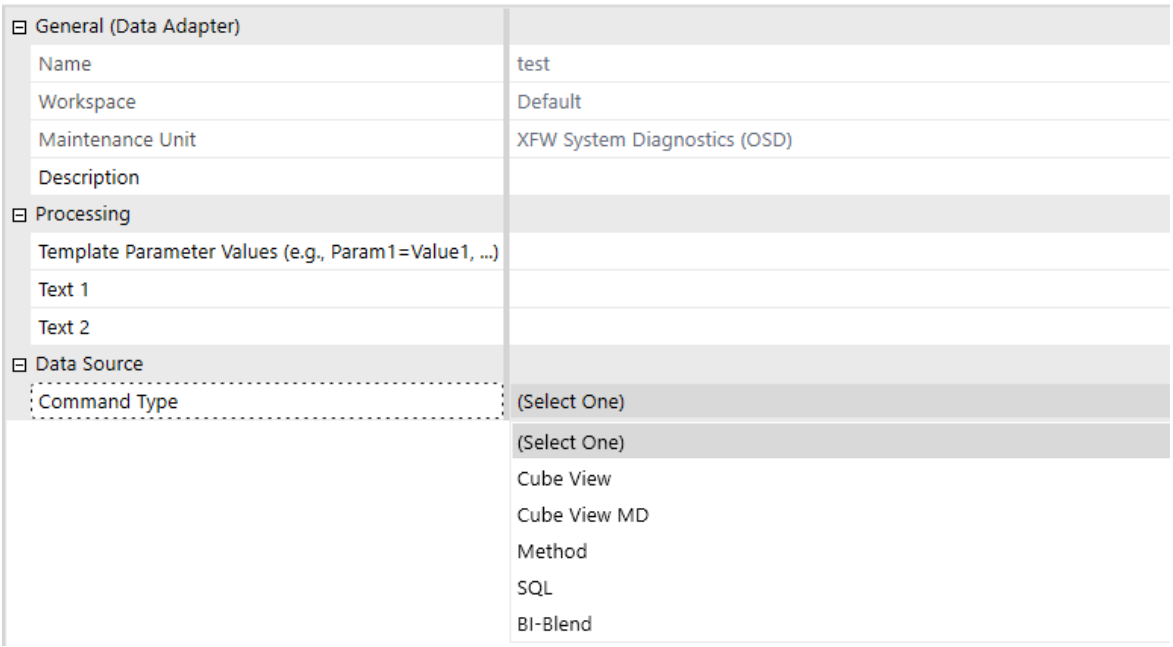
Data adapters specify the type of data used within a dashboard. Once the data adapter is configured and pointing to the appropriate data, attach it to a component to display it on a dashboard.

To manage data adapters, go to **Application** > **Presentation** > **Workspaces**. Select a Workspace and maintenance unit and then select **Data Adapters**.

The data adapter provides General (Data Adapter), Processing, and Data Source sections for configuration. Depending on the selected Data Source Command Type, additional options may be displayed.

Create a Data Adapter

1. Select the **Create Data Adapter** icon.
2. In the **General** and **Processing** sections, enter the required data.
3. In the **Data Source** section, from the **Command Type** drop-down menu, select **Cube View**, **Cube View MD**, **Method**, **SQL**, or **BI-Blend**.



4. To configure the Data Adapter, see Command Types

Command Types

These Command Types are available:

- Cube View Command Type
- Cube View MD (Multi-Dimensional) Command Type
- Method Command Type
- SQL Command Type
- BI Blend Command Type

Cube View Command Type

To use a Cube View as the data adapter source to return a Reporting Table, from the **Command Type** drop-down menu, select **Cube View**.

> **NOTE:** To return a multi-dimensional fact table, use the Cube View MD command type.

Additional options can be selected here to include supplemental information for the resulting table.

> **NOTE:** Adding information that is beyond the default options may have a slight impact on performance:

Cube View
This command type enables a preconfigured Cube View to be the data source for a dashboard.

To select a Cube View, select the ellipsis and use the **Object Lookup** dialog box.

Data Table Per Cube View Row
This property defaults to True. When set to True, a Data Table is created for each row in the dashboard. This enables conditional formatting per Cube View row. When set to False, the rows are merged into one table. This omits any undefined Cube View row.

Include Title
This property defaults to False. When set to True, the title will be displayed from the Report section of the Cube View as the title for the dashboard.

Include Header Left Label 1-4
This property defaults to False. When set to True, the left header labels will be displayed from the report section of the Cube View for the dashboard.

Include Header Center Label 1-4
At the creation of the data adapter, the default is set to False. When set to True, the center header labels will be displayed from the report section of the Cube View for the dashboard.

Include Header Right Label 1-4
This property defaults to False. When set to True, the right header labels will be displayed from the report section of the Cube View for the dashboard.

Include...POV
When set to True, the POV information for the cube, entity, and all other dimensions are included. Use these if the report or dashboard needs this information.

Include Member Details
When set to True, additional Member property details are included in the results.

Include Row Navigation Link
When set to True, this data adapter includes a row navigation link from a Cube View.

Include HasData Status
Includes additional true/false data on whether the row of results contains data for filtering purposes.

Include...View Member Text
This determines whether different Data Attachment text is going to be part of the results. This is necessary for showing text in a Data Explorer object or for using a waterfall chart and wanting to optionally show comments.

Results Table Name
This specifies the name of the resulting table generated when the data adapter is run. Otherwise, it defaults to the name Table.

Cube View MD (Multi-Dimensional) Command Type

To use a Cube View as the data adapter source to return a Multi-Dimensional Fact Table, from the **Command Type** drop-down menu, select **Cube View MD**. The results of the Cube View MD are dimensions (Entity, Consolidation, Scenario, Time, View, Account, Flow, Origin, Intercompany (IC), UD1-UD8) as columns. This simplifies the report building process in the BI Designer, Pivot Grid, and dashboard development.

Loop parameter options are available and can be selected to include incremental information from the modified Cube View definition in the resulting table.

> **NOTE:** To return a reporting table, use the Cube View command type.

Additional options can be selected here to include supplemental information for the resulting tables.

> **NOTE:** Adding information that is beyond the default options may have a slight impact on performance.

Data Source

Cube View
This command type enables a pre-configured Cube View to be the data source for a dashboard.

To select a Cube View, select the ellipsis and use the **Object Lookup** dialog box.

Results Table Name

The name of the table that is generated when the data adapter runs. The default name is Table. Enter a name that can be used to identify the content of the table and the associated data adapter.

> **Example:** The data adapter name is OperatingExpenses_CVMD. The Results Table Name is tbl_OperatingExpenses to identify the table and the data adapter associated with the table.

Miscellaneous

Add Start End Calendar Time

When set to True, the Data Table incorporates the Start and End Dates used in the POV/Time Profile for the Cube View and creates additional columns in a Date/Time Field Type Format: StartDate, EndDate for each row in the dashboard. This enables you to use the Date Grouping functions in the BI Designer. When set to False, these Date Time fields are not added or displayed.

Header Text

Entity

Controls the display of the entity name and description in the results table from the Cube View. The default value is Name and Description.

Consolidation

Controls the display of the consolidation name and description in the results table from the Cube View. The default value is Name and Description.

Scenario

Controls the display of scenario name and description in the results table from the Cube View. The default value is Name and Description.

Time
Controls the display of the time name and description in the results table from the Cube View. The default value is Name and Description.

View
Controls the display of the view name and description in the results table from the Cube View. The default value is Name and Description.

Account
Controls the display of the account name and description in the results table from the Cube View. The default value is Name and Description.

Flow
Controls the display of the flow name and description in the results table from the Cube View. The default value is Name and Description.

Origin
Controls the display of the origin name and description in the results table from the Cube View. The default value is Name and Description.

IC
Controls the display of the IC name and description in the results table from the Cube View. The default value is Name and Description.

UD1-UD8
Controls the display of the UD1-UD8 name and description in the results table from the Cube View. The default value is Name and Description.

Loop Parameters

These properties enable you to make changes to the output of a Cube View definition being used in a table for reporting. The Loop parameter filters the results and considers the additional parameters to be passed to the Cube View definition and adds those results to the table accordingly.

The parameters override the POV. So, if the entity POV is set to CT, and the loop parameter filters are set to NY, MA, and NJ, then data for those is returned not CT. A Loop must be used to change parameters.

> **Example:** A Loop parameter may be used to loop through a list of entities in the Cube View definition and return multiple entities for that specific Cube View. The dimension type and Member Filters should be added here to pass along the appropriate Loop, for example, Dimension Type=Entity, Member Filter= E#US.Base, which applies to each entity included in the Loop.

> **NOTE:** It is recommended to not loop any dimensions that already exist in the Cube View rows or columns.

The default value for both dimension type properties is (Not Used), and the Member Filter properties are not available by default. This configuration displays the results without any consideration of additional parameters to pass to the query. When the Dimension Types are set along with the Member Filters, the results will consider the additional parameters to be passed to the Cube View definition and will add those results to the table accordingly.

Dimension Type 1 and 2

The dimension type containing the list of members, for example, entity or account.

Member Filter 1 and 2

Enter a Member Filter to determine what is seen in the parameter. Select the ellipsis to display the Member Script Builder dialog box. Enter a member script to change the Cube View definition.

> **Example:** This example changes the POV for the Products:
> UD2; U2#Top.Base

Dimension Leveling

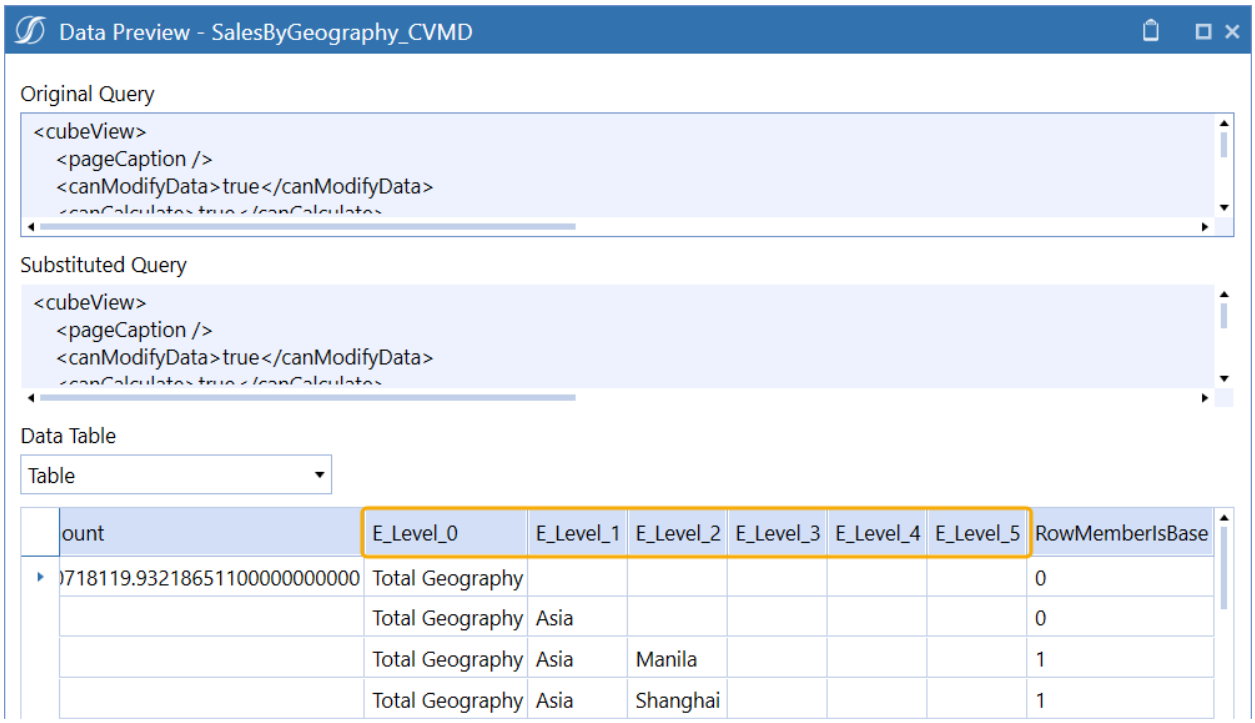
The Dimension To Level property setting displays dimensional data as a hierarchical tree in the BI Viewer. Dimension leveling enables you to display data in a hierarchical structure so that you can drill down to view child data. After the data is leveled, you can use the BI Viewer to view data as a tree, a pivot table, a chart, or a grid. You can choose from the following options:

- **Outermost Row**: The hierarchy uses the first row and first level (of row) definition.
- **Outermost Column**: The hierarchy uses the first column and first level (of column) definition.
- **Both**: The hierarchy uses the first row and first level definition and uses the first column and first level (of column) definition.

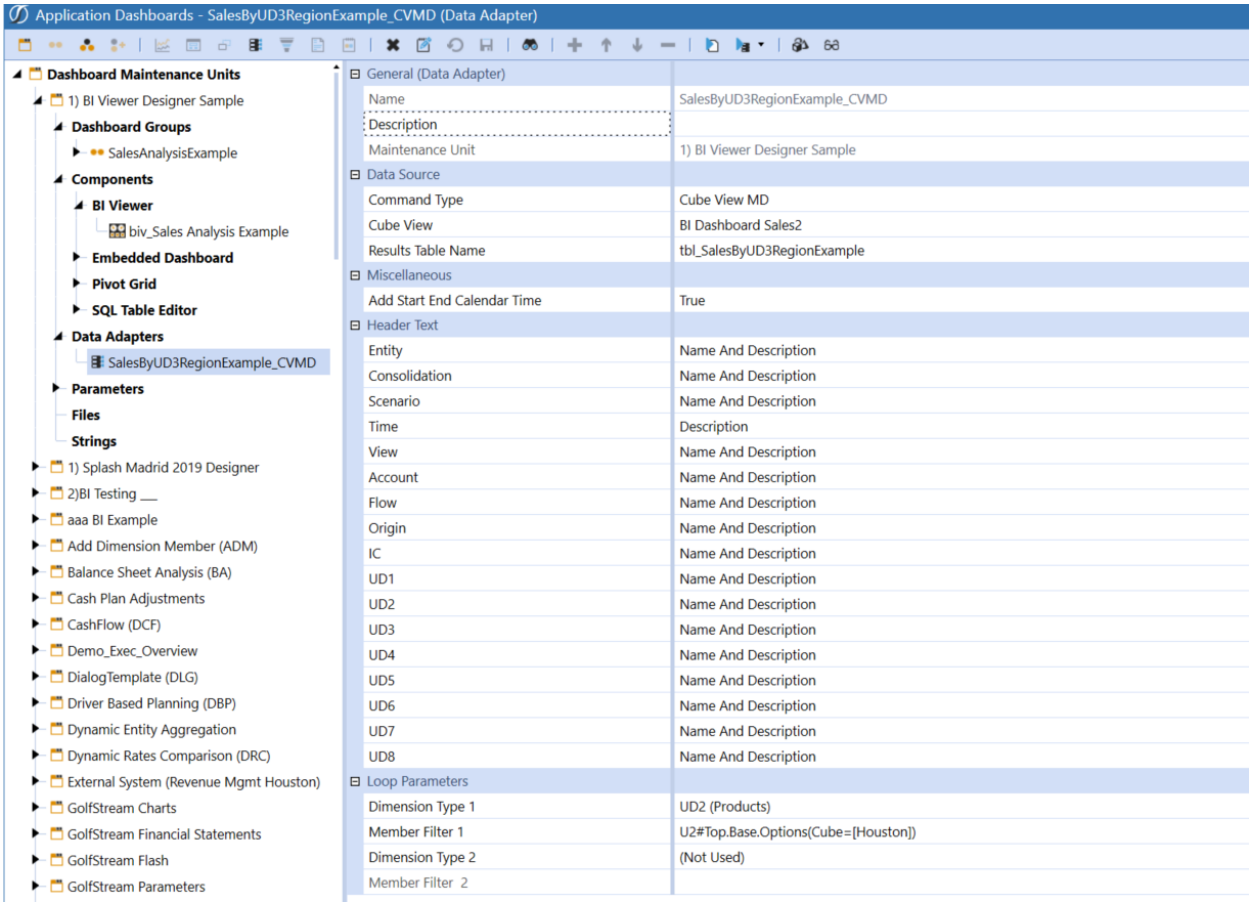
When the data adapter is run, the Data Table generates the additional columns for the levels including columns to determine the status of the level.

> **Example:** RowMemberIsBase = 1: Row Member contains the Base level of data RowMemberIsBase = 0: Row Member is not the Base level of data ColumnMemberIsBase= 1: Column Member contains the Base level of data ColumnMemberIsBase= 0: Column Member is not the Base level of data.

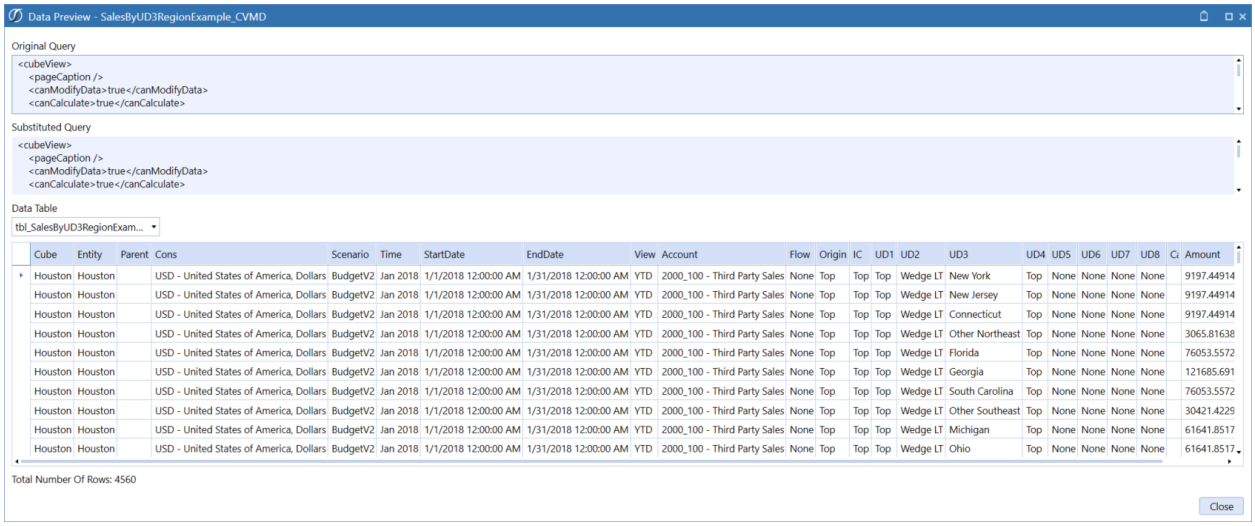
To test the data adapter, select the Test Data Adapter icon. The Data Preview window displays the table data. Each dimension leveled column is prefixed with an E to indicate it is an Entity, followed by a level number. A RowMemberIsBase value of 1 indicates that there is no remaining child data.



Cube View MD Data Adapter Example



The Results Table from the Cube View MD data adapter example presented.



View Results in the BI Viewer

With dimension leveling, you can view data in an easy-to-understand, hierarchical format. With the BI Viewer, you can design various dashboard items in which to view the dimension leveled data. For example, you can configure the BI Viewer to display data in pivot tables and charts at the same time.

To configure the BI Viewer, complete the following steps:

1. Create a BI Viewer component:
    a. In the navigation tree, select **Components** and select the **Create Dashboard Component**  icon.
    b. In the **Create Dashboard Component** dialog box, select **BI Viewer** and then select the **OK** button.
    c. On the **Component Properties** tab, in the **Name** field, enter the name of the component.

d. Select the **Data Adapters** tab and select the **Add Dashboard Component** icon.

e. In the **Add Data Adapter** dialog box, select the appropriate data adapter and then select the **OK** button.

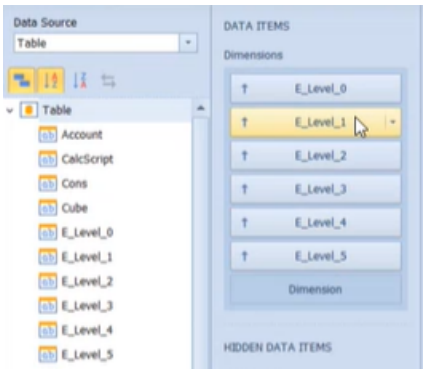
f. Select the **Save** icon.

2. Design the BI Viewer dashboard:

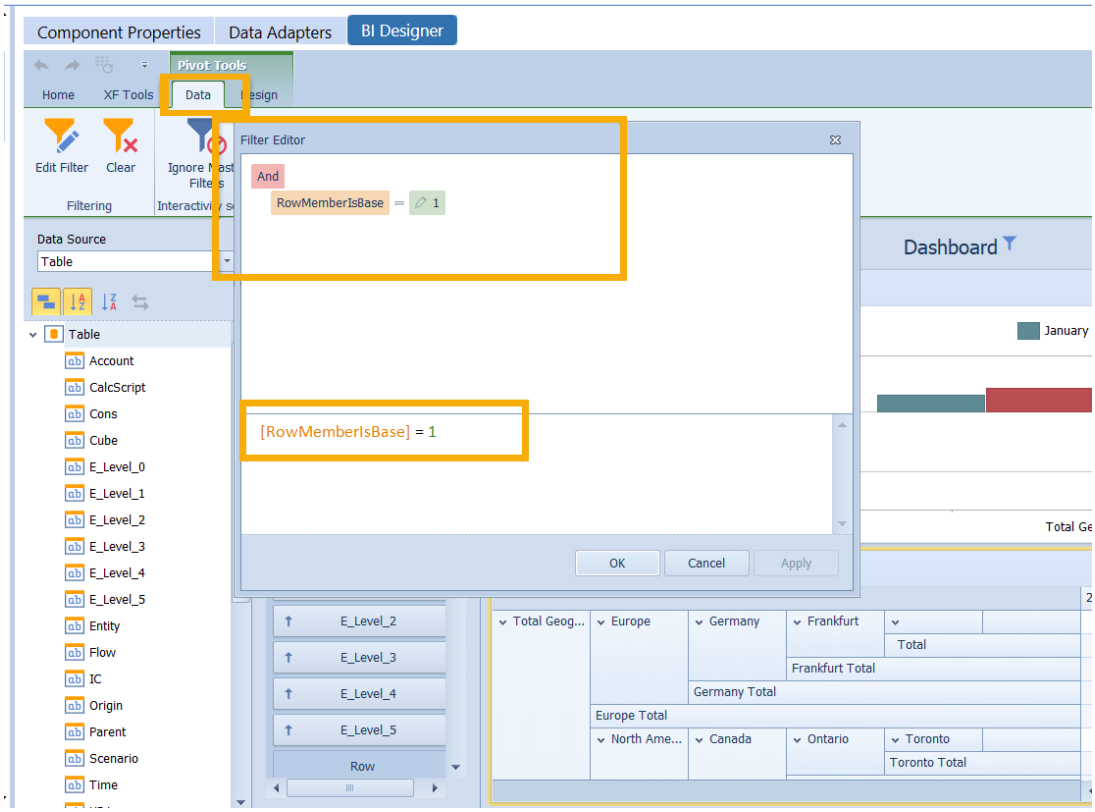
a. Select the **BI Designer** tab at the top of the window.

b. From the **Data Source** drop-down menu, select the data source from which to pull data. This is usually a table.

c. Drag each dimension level to view from the Table view into the DATA ITEMS column under Dimensions.

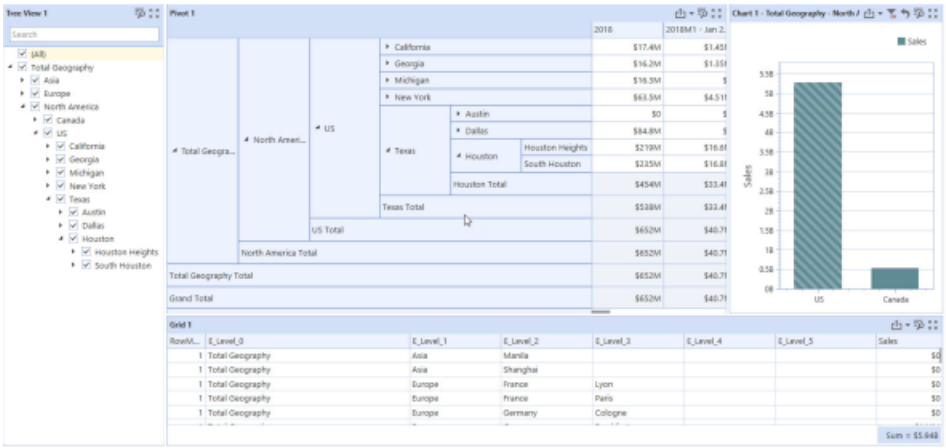


d. To filter items based on whether they are base items, drag the **RowNumberIsBase** item into the DATA ITEMS column under HIDDEN DATA ITEMS. This item will not display in the resulting dashboard but will be available for you to filter on if necessary.



e. From the BI Designer ribbon, select the type of dashboard item to view in the dashboard. For example, select **Filter Element** > **Tree View**. In the resulting tree view, notice that you can expand parent members down to their base child members.

   If you add a Grid, all levels of the parent entity display in their own columns. To see a pivot table, select **Pivot** from the ribbon and notice that each level is expandable similar to tree view.



Method Command Type

To use a Method as the Data Adapter source to return a table, from the **Command Type** drop-down menu, select **Method**. From the **Method Type** drop-down menu, select a method type.

> **TIP:** To view an example Method Query, leave the Method Query field blank, select the **Save** icon, and then select the **Test Data Adapter** icon.

Method Types

BusinessRule

Use the business rule method when creating a custom rule to incorporate within a Method Query. The business rule is used as the first set of braces within the Method Query.

```
{XFR_DataUnitCompare}{DataUnitComparisonDataSet}{Cube1=|!Members_Cubes!|, Entity1=|!Members_
Entities_AllDims_Base!|,
Parent1=[], Cons1=|!Members_Cons_Statutory!|,Scenario1=|!Members_Scenarios_AllDims_Base!|,
Time1=|!Members_Time_WFYear_
Base!|, Cube2=|!Members_Cubes!|, Entity2=|!Members_Entities_AllDims_Base!|,
Parent2=[], Cons2=|!Members_Cons_Statutory!|, Scenario2=|!Members_Scenarios_
AllDims_Base_Var!|, Time2=|!Members_Time_WFYear_Base!|, View=
|!Members_View_Numeric!|, SuppressMatches=|!DataUnit_SuppressMatches!|}
```

```
{MyBusinessRuleName}{MyDataSetName}{Name1=Value1, Name2=[Value2]}
```

CertificationforWorkflowUnit

The CertificationforWorkflowUnit method lists all Certification Questions for the particular Workflow Unit. Parameters should include each value enclosed in braces.

```
{MyWorkflowProfileName}{Actual}{2015M1}{true}{Empty String or Filter Expression}
```

```
{Dallas}{Actual}{2011M2}{true}{}
```

ConfirmationforWorkflowUnit

The ConfirmationforWorkflowUnit method lists the Confirmation Rules results for a particular Workflow Unit. Parameters should include each value enclosed in braces.

```
{MyWorkflowProfileName}{Actual}{2015M1}{true}{Empty String or Filter Expression}
```

```
{Montreal}{Actual}{2011M6}{true}{}
```

DataUnit

The DataUnit method returns all rows of data related to the specified Data Unit, for example, Cube, Entity, Parent, Consolidation Member, Scenario, Time, and View. Parameters should include each value enclosed in braces.

```
{Cube}{Entity}{Parent}{Cons}{Scenario}{Time}{View}{True}{Empty String or Filter Expression}
```

DataUnitComparison

This returns all rows from two different Data Units specified for comparison purposes. Parameters should include each value enclosed in braces.

```
{Cube1}{Entity1}{Parent1}{Cons1}{Scenario1}{Time1}{Cube2}{Entity2}{Parent2}{Cons2}
{Scenario2}{Time2}{View}{True}{True}{Empty String or Filter Expression}
```

ExcelFile

The ExcelFile method returns data sourced from an Excel file. Parameters should include each value enclosed in braces.

```
{Application}{Documents/Public/MyExcelfile.xfDoc.xlsx}{Sheet1 or Empty}
{MyDataRange or Empty}{False}{False}{}
```

FormsStatusForWorkflowUnit

The FormsStatusForWorkflowUnit method lists detailed information about the Forms for a particular Workflow Unit. Parameters should include each value enclosed in braces.

```
{Houston}{Actual}{2011M1}{All}{}
```

```
{MyWorkflowProfileName}{Actual}{2015M1}{All}{Empty String or Filter Expression}
```

Groups

The Groups method returns the Group ID, Name, Description, and whether this is an Exclusion Group. Parameters should include each value enclosed in braces.

```
{GroupName = 'FinanceGroup'}
```

GroupsforUsers

The GroupsforUsers method returns the Select User properties and Groups. This returns the same group properties as the Group Method Query. Parameters should include each value enclosed in braces.

```
{UserName = 'Administrator'}{}
```

ICMatchingforWorkflowUnit

The ICMatchingforWorkflowUnit method returns a detailed Intercompany Matching Report table for the given Workflow Unit and several other Parameters. The Parameters override what is set up in the Workflow Profile. Parameters should include each value enclosed in braces.

```
{Workflow Profile Name}{Scenario Name}{Time Name}{Plug Account Override}{
Suppress Matches Override}{Tolerance Override}{Filter}
```

```
{Flint}{Actual}{2011M1}{Empty String or A#MyPlugAccount}{Empty String or
true/false}{Empty String or 0.0}{Empty String or Filter Expression}.
```

```
{MyWorkflowProfileName}{Actual}{2015M1}{PlugAccount for Workflow Parameter Set (Exclude A#)}
{Empty String or C#MyCurrencyOverride}{Empty String or V#MyViewOverride}{Empty String or
A#MyPlugAccountOverride}{Empty String or true/false}{Empty String or 0.0}{Empty String or
E#MyEntityOverride}{Empty String or E#MyPartnerOverride}{Empty String or
MyDetailDimsOverride (F#All:O#Top:U1#All:U2#All:U3#All:U4#All:U5#All:U6#All:U7#All:U8#All)}
{Empty String or Filter Expression}
```

ICMatchingPlugAccountsforWorkflowUnit

The ICMatchingPlugAccountsforWorkflowUnit method returns the list of Intercompany Plug Accounts for a given Workflow Profile and Scenario Type configured for the Workflow Profile. Parameters should include each value enclosed in braces.

```
{MyWorkflowProfileName}{Actual}{2015M1}
```

JournalforWorkflowUnit

The JournalforWorkflowUnit method lists the Journals entered for a given Workflow Unit. Parameters should include each value enclosed in braces.

```
{Frankfurt}{Actual}{2011M3}{All}{}
```

```
{MyWorkflowProfileName}{Actual}{2015M1}{All}{Empty String or Filter Expression}
```

Members

The Members method returns Dimension ID, Member information, such as ID, Name, and Description, and a few other properties for the chosen dimension and Member Filter. Parameters should include each value enclosed in braces.

```
{Account}{MyAccountDim}{A#Root.TreeDescendants}{Empty String or Filter Expression}
```

UserCubeSliceRights

The UserCubeSliceRights method lists each user's Data Access settings on a given cube. Parameters should include each value enclosed in braces.

```
{AllUsers}{AllCubes}{Empty String or Filter Expression}
```

UserEntityRights

The UserEntityRights method returns the Cubes and Entities the user has access to according to the security settings under Entities. Parameters should include each value enclosed in braces.

```
{AllUsers}{AllCubes}{AllEntities}{Empty String or Filter Expression}
```

Users

The Users method returns all properties associated for the chosen User Name. Parameters should include each value enclosed in braces.

```
{UserName = 'Administrator'}
```

UserScenarioRights

The UserScenarioRights method returns all accessible Scenarios and many related Scenario properties for the chosen User Name filter and cube. Parameters should include each value enclosed in braces.

```
{AllUsers}{AllCubes}{Empty String or Filter Expression}
```

UserinGroups

The UserinGroups method returns a list of Users and selected User properties for the chosen User Group. Parameters should include each value enclosed in braces.

```
{GroupName = 'FinanceGroup'}{}
```

UserWorkflowProfileRights

The UserWorkflowProfileRights method lists the rights assigned to users for Workflow Profiles. Parameters should include each value enclosed in braces.

```
{User Name}{Workflow Cube Name}{Workflow Profile Type}{Filter}
```

```
{Administrator}{GolfStream}{AllProfiles}{}
```

```
{AllUsers}{AllProfiles}{AllProfiles}{Empty String or Filter Expression}
```

WorkflowandEntityStatus

The WorkflowandEntityStatus method returns properties for Workflow status, status code/description, last executed step, date/time information, completed steps, and data status for the chosen Workflow Unit, including both the Workflow Profile level and individual entity level.

Workflow Profile Description is a clear, human-readable description of the Workflow Profile that better communicates its purpose and scope. Last Step User is the person who most recently completed a workflow step, and it provides visibility into who last acted on that workflow.

Parameters should include each value enclosed in braces.

```
{MyWorkflowProfileName}{Actual}{2015M1}{AllProfiles}{Descendants}{Empty String or Filter Expression}
```

WorkflowCalculationEntities

This lists the entities that display under Calculation Definitions for this Workflow Profile.

WorkflowConfirmationEntites

This lists the entities that display under Calculation Definitions with a Confirmed checkbox for this Workflow Profile.

WorkflowLockHistory

The WorkflowLockHistory method displays all Lock history details in a report for a given Workflow Profile with the ability to filter by Scenario, Time, Workflow Profile, Origin, Channel, Time, User, and Lock Status. Parameters should include each value enclosed in braces.

```
{MyWorkflowProfileName}{Actual}{2015M1}{AllProfiles}{Descendants}{Empty String or Filter Expression}
```

WorkflowProfileandDescendantEntities

The WorkflowProfileandDescendantEntities method creates a list of Entities and all descendants located under Entity Assignment for this Workflow Profile.

WorkflowProfileEntities

The WorkflowProfileEntities method creates a list of entities located under Entity Assignment for this Workflow Profile.

WorkflowProfileRelatives

The WorkflowProfileRelatives method lists related Workflow Profiles based on certain criteria. Parameters should include each value enclosed in braces.

```
{GolfStream}{Actual}{2011M1}{AllProfiles}{Descendants}{true}{}
```

```
{MyWorkflowProfileName}{Actual}{2015M1}{AllProfiles}{Descendants}{true}{Empty String or
Filter Expression}
```

WorkflowProfiles

The WorkflowProfiles method lists the Workflow Profiles. Parameters should include each value enclosed in braces.

```
{AllProfiles}{Type = 'InputAdjChild'}.
```

WorkflowStatus

The WorkflowStatus method lists the status, lock status, and last step completed of a given Workflow Unit. Parameters should include each value enclosed in braces.

```
{Houston}{Actual}{2011M1}{AllProfiles}{Descendants}{}
```

```
{MyWorkflowProfileName}{Actual}{2015M1}{AllProfiles}{Descendants}{Empty String or Filter Expression}
```

WorkflowStatusTwelvePeriod

This returns status value and text summary for 12 months for a given Workflow Profile, scenario, and year. Parameters should include each value enclosed in braces.

```
{MyWorkflowProfileName}{Actual}{2011}{AllProfiles}{Descendants}{Empty String or Filter Expression}
```

SQL Command Type

A SQL query against either the application or framework database can be written as a data source. Reference substitution variables such as |WFProfile| from within the SQL statement.

Database Location
Select Application, Framework, or External data source.

- **Application**: The current OneStream Application database where Stage and financial cube data reside.
- **Framework**: The connected OneStream Framework database where security and log data reside.
- **External**: Any other database outside of OneStream. If External is the chosen Database Location, select the External Database Connection name. This list is defined in the OneStream Server Configuration Utility.

SQL Query
The SQL statement ran for this data adapter.

Results Table Name
This specifies the name of the resulting table generated when the data adapter is run. Otherwise, it defaults to the name Table.

BI Blend Command Type

The purpose of the BI Blend adapter is to provide dashboard designers with a predefined interface to query BI Blend tables. The data sourced by the BI Blend adapter can be used to design standard OneStream dashboards.

The BI Blend Workflow process is designed to generate large external database tables formatted in a column store index optimized for analytic reporting. The overall number of records that may be generated by the BI Blend process may be too large for the dashboard components to process. Therefore, the designer should manage the returned dataset by defining an appropriate Where Clause to retrieve a slice of the BI Blend table that is suitable for the BI Blend adapter.

Results Table Name
Define a table name to store the content of the BI Blend adapter query.

Table Info
This defines the name of the BI Blend table name to be queried. This label supports the use of parameters and substitution variables.

Group By
This defines the source database columns and their order, which are to be returned to the data adapter. These labels must match the names specified on the database. Any field to be pivoted must be in the Group By list.

Data Field Aggregation Types

The Data Field Aggregation Type is used to identify which database column should be used as the Measure field. The adapter can support multiple database columns as measures. Each defined measure can be defined to derive the results as a different Aggregation Types. Parameters and substitution variables are supported.

- Sum
- Min
- Max
- Avg
- Count

The syntax required defines each Aggregation Type result as a unique key. AggregationType = [databaseColumnName, Type]

> **Example:** AggType1 = [2018M1, Count], AggType2 = [2018M1, Sum]

> **Example:** AggType1 = [|WFYear|M1, Count]

Where Clause

The is used to pre-filter the table. The BI Blend data adapter queries the source table and returns the results to the client for processing, such as for use in a Pivot Grid. The size of the query must be managed to ensure the overall performance of the dashboard report is optimized. The Where Clause uses standard SQL syntax to define the filters that are applied to the query.

> **Example:** DatabaseColumn = 'Text'

> **Example:** DatabaseColumn Like '2019%' - As all columns beginning with 2019

> **Example:** DatabaseColumn1 = ‘CostCenter’ And DatabaseColumn2 = ‘Midwest’

Direct Load Reporting Data Adapter Method Types

Dashboard reporting against Direct Load Workflow activity is simplified with specialty data adapter methods.

- **Direct Load Info**: Retrieves results from the StageDirectLoadInformation table.

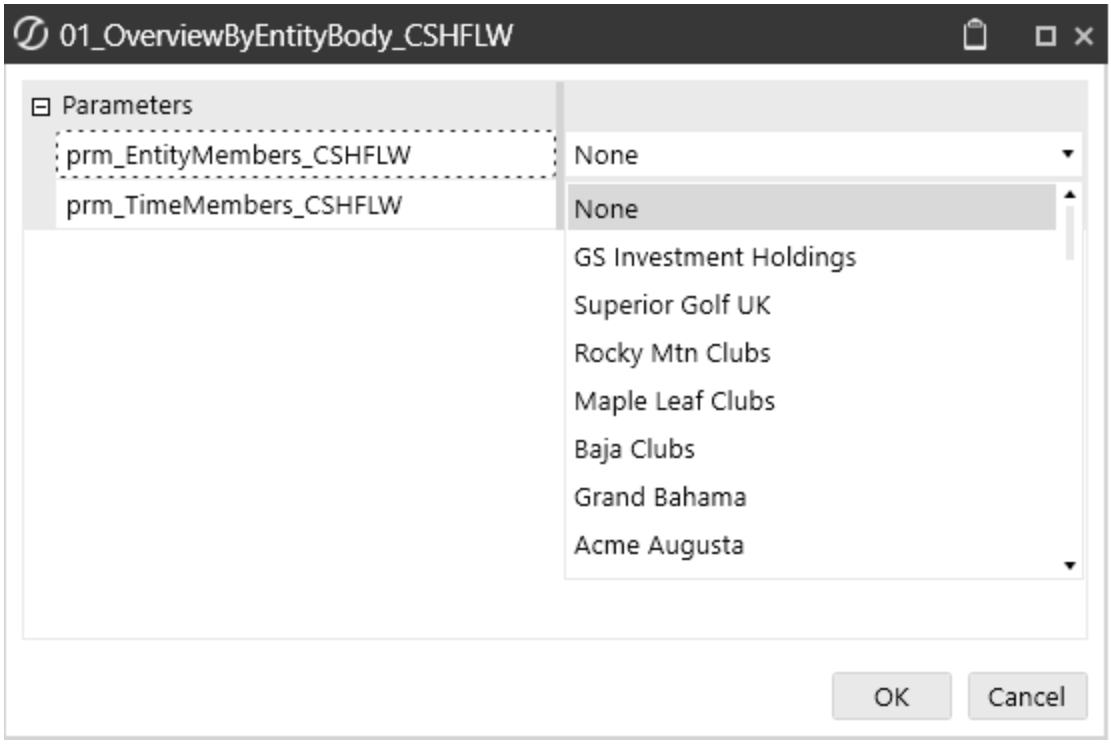
  > **Example:** {Workflow Profile Name}{Scenario Name} {Time Name}{Filter}

- **Stage Summary Target Data**: Query designed to automatically determine the method used to manage Summary Target Data, as Row or Blob, and appropriately return the results.

  > **Example:** {Workflow Profile Name}{Scenario Name} {Time Name}{Filter}

Parameters

Parameters prompt you to enter or modify values so you can filter the data to display on a dashboard.



See "Create a Parameter" in the *Understanding the Financial Model Guide*:

Files

Administrators can use the File option to upload images and documents to create personalized, company-specific dashboards. Items such as PDF report books and Word or Excel Extensible Documents can also be stored and used in dashboards. With the File option, you can complete these actions:

- Specify a file name and a brief description.
- Identify the maintenance unit to which the file belongs.
- Upload or browse to the content file to use.

To add a file, follow these instructions:

1. Go to the maintenance unit.
2. Select **Files**.
3. Select the **Create File** icon.
4. In the **Content File** property, select the **Upload File** icon.
5. Go to the file and select the **Open** button.

Strings

Strings are set up as an object type under a Workspace maintenance unit. For example, Strings can be used with XFString functionality and displayed in a Cube View as a Caption or Heading. See Reference Alias with XFString" in the *System Administration Guide*.

Linked Dashboards

Similar to linked Cube Views, linked dashboards offer flexibility and easy data analysis. They provide the option to launch a dashboard from a Cube View when the Cube View is viewed in either the data explorer grid, the Windows application, the Spreadsheet tool, or the Excel Add-In.

Linked dashboards enhance data analysis, providing an all-encompassing, guided-reporting experience as the data presented in the linked reports is directly related to the main Cube View through Bound parameters. It also eliminates the need to incorporate too much detail in a single Cube View or dashboard because you can drill down into the details as needed. Linked dashboards enable you to drill down to Workspaces, resources, and interactive dashboards for data analysis, for example, BI Viewer or Advanced Charts.

When you right-click on a Cube View data cell, the context menu displays a list of linked dashboards with linked Cube Views. Selecting a link launches the linked dashboard in a dialog box providing more detail and visibility.

To launch a Cube View from a source Cube View, see Linked Cube Views.

Reporting and Linking Objectives

Review the following reporting objectives to understand how to link reports and determine what must be built:

- Determine the overall purpose of the linked dashboards in relation to the source Cube View.
- Determine the data you want to display with the source Cube View as well as the detail data you want to display in the linked dashboards. This may require additional detail Cube Views.
- Determine the POV information to pass from the source Cube View to the dashboard. This determines the Bound parameters needed on the source Cube View.
- Determine how the POV information will be used in the dashboard. This determines where the Bound parameters need to be defined.

Linked Dashboards Example

The following example uses a linked dashboard and Cube View for budget data entry. The objective is to launch an Operating Expense dashboard form from a Budget Review source Cube View. Linked forms must be specific to the selected entity, time, and account from the source Cube View.

Source Cube View Setup

1. Create a source Cube View to link to the dashboard.

   In this example, the Budget Review Cube View displays parent Operating Expense Accounts by scenario. Two Member List parameters are used to select an entity and time period at runtime.

Budget Review Entity: Houston Heights Time: Jan 2022					
	Baseline Budget V1	Revisions	BudgetV2	Target	Comments
50300 - Total Employee Compensation	1,928,768	214,409	2,143,177	2,281,469	
51099 - Total Utilities	68,094		68,094	68,592	
51199 - Total Professional Services	43,516		43,516	43,835	
52099 - Marketing & Advertising	177,289		177,289	178,586	
52199 - Travel & Entertainment	57,147		57,147	57,565	
52299 - Total Facility Expense	76,494	4,390	80,884	77,053	
52399 - Total HR Expenses	6,666		6,666	6,715	
52499 - Total Equip Expense	46,568		46,568	46,909	
53099 - Total Telecom	13,371		13,371	13,469	
53199 - Total R&D Expenses	62,603		62,603	63,061	
54099 - Depreciation & Amortization Expense	5,269		5,269	9,434	
54199 - Total Other Operating Expenses	87,427		87,427	88,067	
54400 - Total Operating Exp Before Allocation	2,573,214	218,799	2,792,013	2,934,754	
54299 - Total Allocations In	503,328	-503,328			
54300 - Allocations Out	-503,328	503,328			
54350 - Total Allocations	0				
54500 - Total Operating Expenses	2,573,214	218,799	2,792,013	2,934,754	
62000 - Total Operating Income	4,145,096	-218,799	3,926,297	4,315,560	

2. Navigate to **General Settings** > **Navigation Links** > **Bound Parameter Names** and define a Bound parameter for each POV member you will pass to the linked reports. This step varies and may not always be applicable.

   In this example, the linked form must be specific to the selected entity, time, and account

from the source Cube View. Each of those dimensions require a Bound parameter.

Bound Parameter Names	
Cube Bound Parameter Name	
Entity Bound Parameter Name	drillentity
Parent Bound Parameter Name	
Consolidation Bound Parameter Name	
Scenario Bound Parameter Name	
Time Bound Parameter Name	drilltime
View Bound Parameter Name	
Account Bound Parameter Name	drillaccount

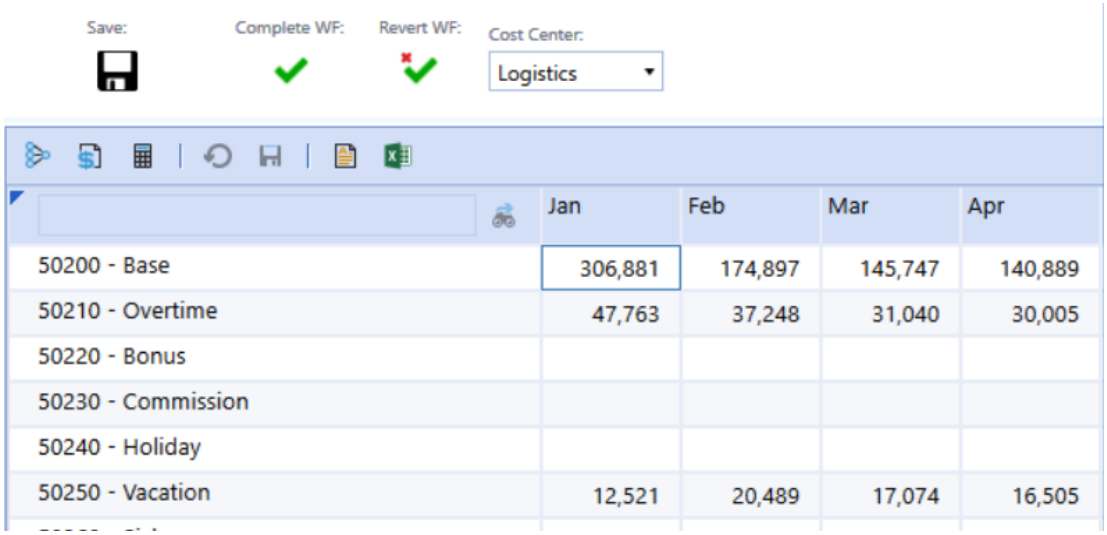
Linked Dashboard Setup

Dashboard setup varies based on the report purpose and how the dashboard and Cube Views are linked.

In this example, the Operating Expense dashboard contains the following objects:

- **Button components**: Run specific Save and Workflow Task Actions.
- **Combo Box component**: Select a Cost Center.

- **Cube View component**: Display an Operating Expense data entry Cube View.



Cube View Bound Parameters in the Linked Dashboard

When using Bound parameters on the source Cube View, it is important to understand how the source Cube View's Bound parameters relate to the dashboard's components and how they impact dashboard results.

The intent of Bound parameters is to pass the parameter value in the background rather than through a prompt to create seamless navigation. Bound parameters impact the dashboard when there is a member dependency between one of its components and the source Cube View.

Passing Bound parameters from a Cube View to a linked dashboard has the following requirements:

- You must properly apply them to the component data source.
- You must create a Supplied parameter component for each Cube View Bound parameter and assign the components to the dashboard. See Supplied Parameter.

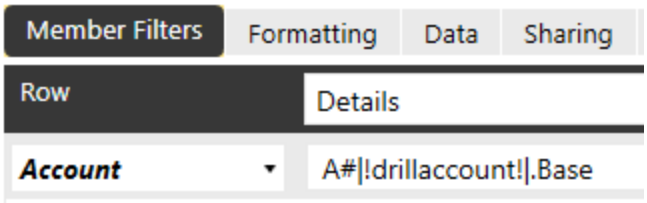
Requirement 1

In this example, the data source for the Cube View component is an Operating Expense Cube View. The results of this Cube View are dependent on the source Cube View account, entity, and time members.

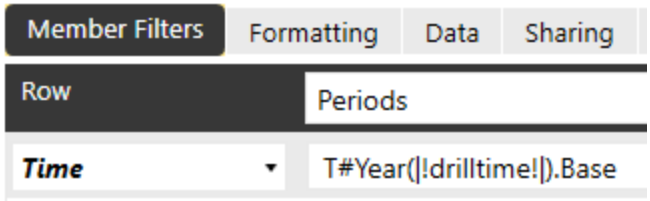
To pass these values, these Bound parameters from the source Cube View must be assigned to the Operating Expense Cube View: drilltime, drillaccount, and drillentity. This is the Cube View assigned to the Cube View component, and it must use the same entity member.



Rows must display the base accounts of the source Cube View selected parent account.



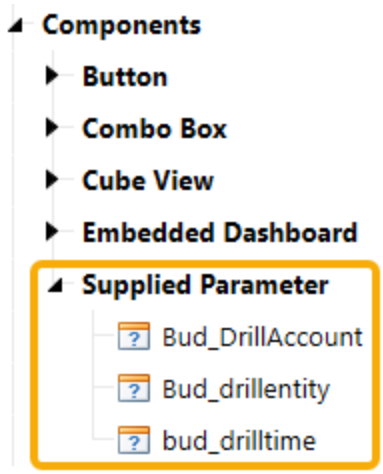
Columns must display time periods based on the selected year in the source Cube View.



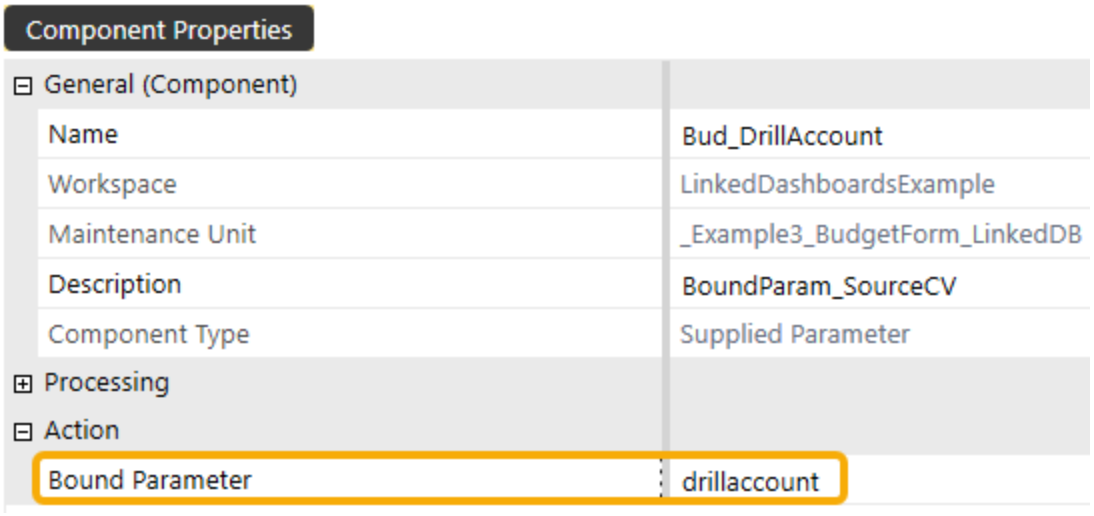
> **NOTE:** In this example, the selected Cost Center in the dashboard combo box is also used by the OpEx Cube View. This does not require a cost center (UD1) value/Bound parameter from the source and is specific to the dashboard functionality.

Requirement 2

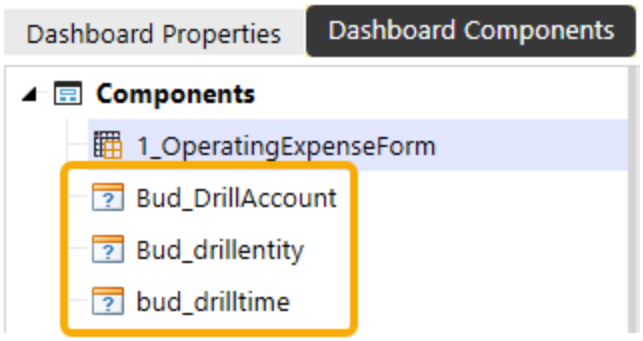
The source Cube View uses these Bound parameters: drillaccount, drillentity, and drilltime. Each requires three supplied parameter components.



Go to **Component Properties** > **Action** > **Bound Parameter** and type the name of the respective Cube View Bound parameter.

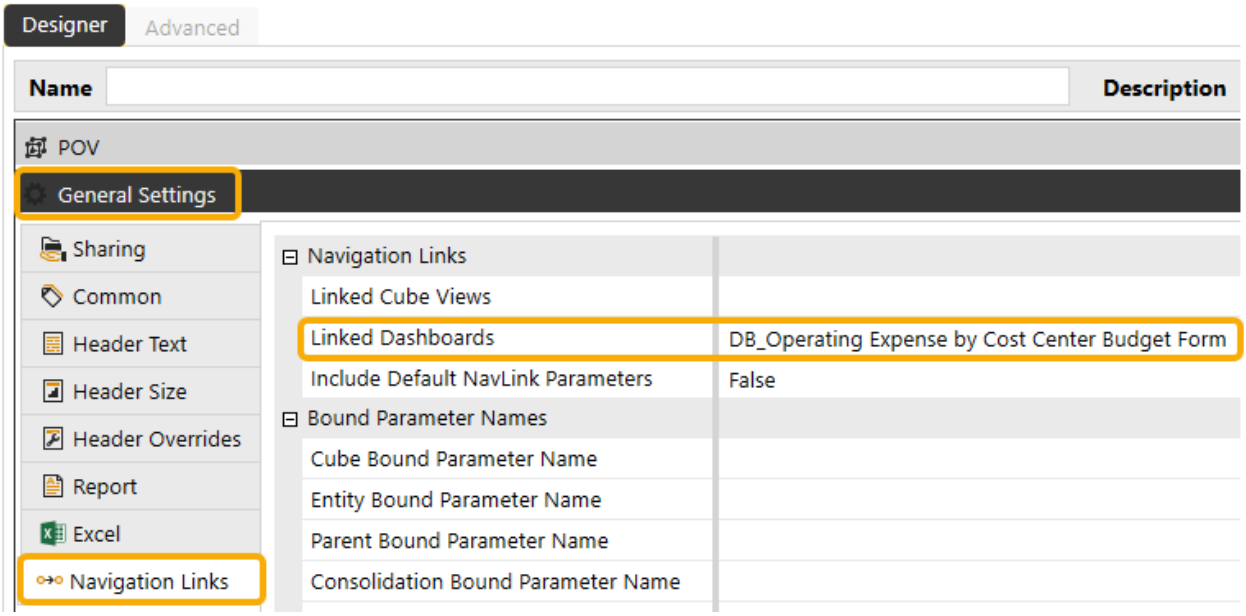


Construct and assign components, and ensure you include the Supplied parameter components. To assign components to a dashboard, see Dashboard Components .

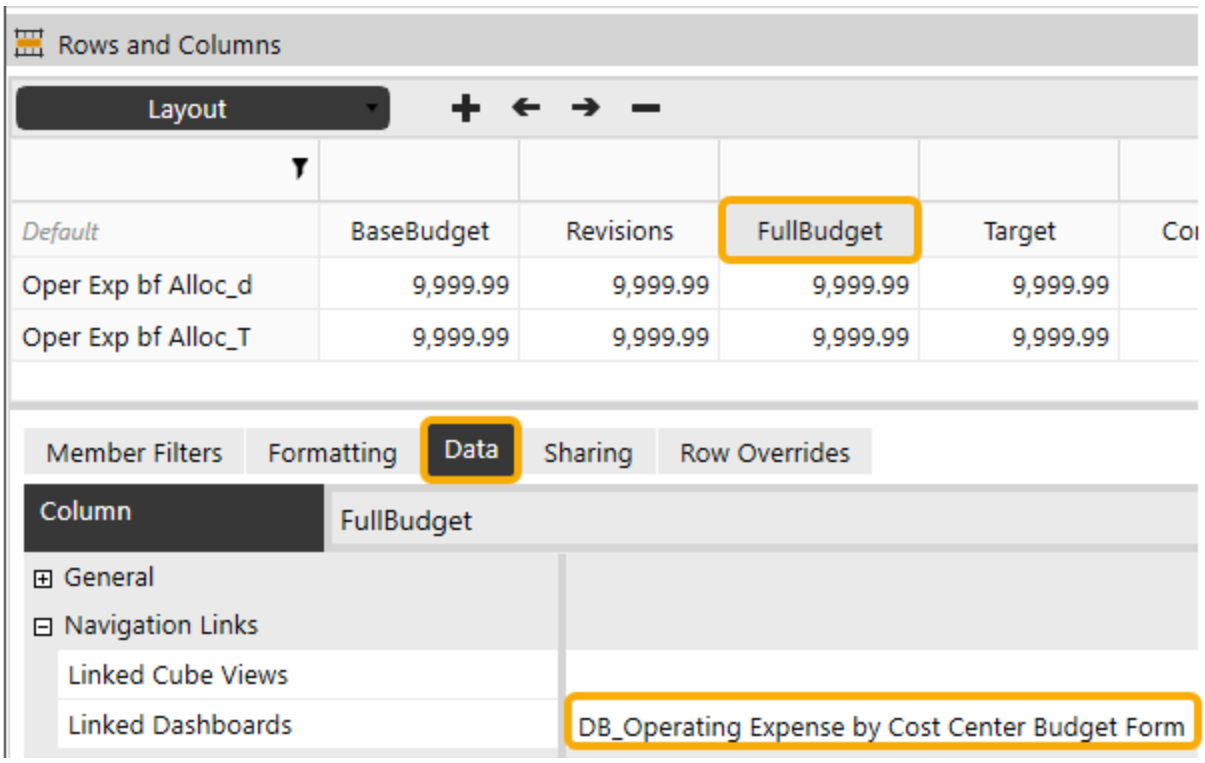


Link the Dashboard to the Cube View

Navigate back to the source Cube View and assign the dashboard accordingly. This has the same behavior as linked Cube Views. When the Cube View is run, the dashboard will be available to launch when you right-click on any Cube View data cell.

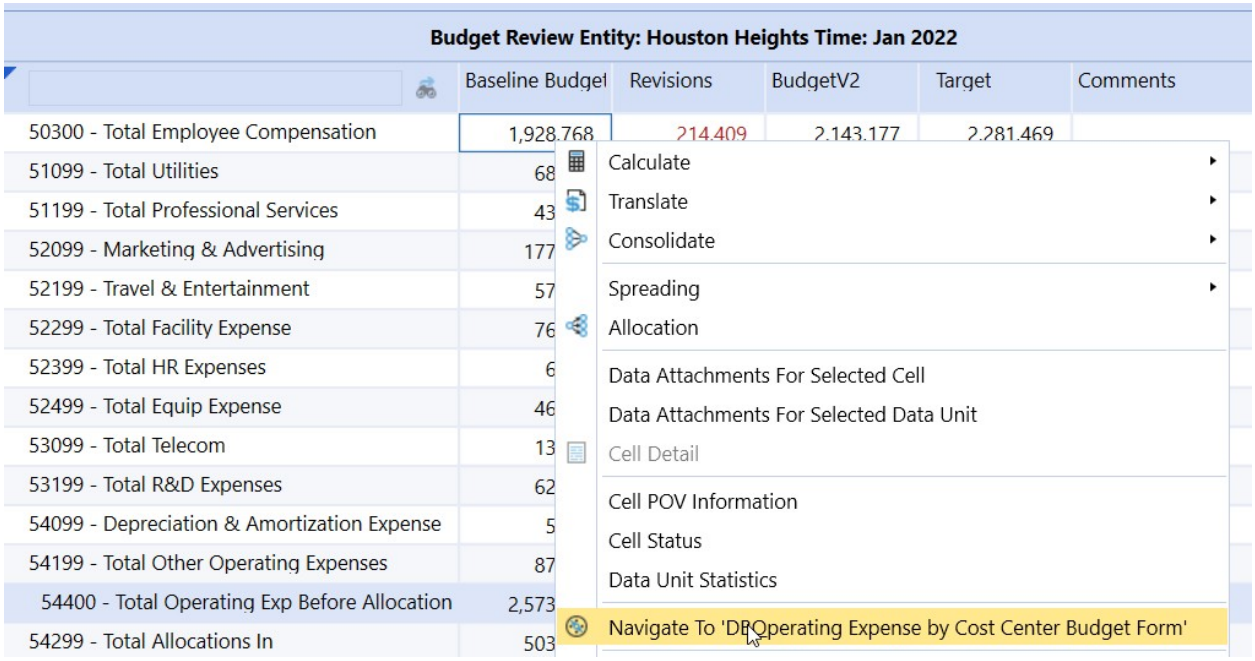


The dashboard is only available when you right-click on data cells in this specific column or row.



Launch the Dashboard

To run the source Cube View, select the Cube View and then select the **Open Data Explorer**  icon. Right-click on the data cell that corresponds to the linked dashboard assignment and navigate to the dashboard.



Presenting Data Using Report Books

Report Books enable you to combine a variety of reports and files into a single document. They are commonly used to create financial statements and management report packages. Anyone who needs a snapshot view of their company financials can run report books. This is done most often by managers and individuals who report to management.

The following pre-built reports can be customized to fit your reporting needs:

- Cube views. These can be exported to an Excel file.
- Dashboard reports
- Charts
- Extensible Documents

After creating a Report Book, it can be viewed and used in different contexts. For example, it can be added to other books, added to dashboards using either the Book Viewer or File Viewer components, or emailed through the OneStream Parcel Service. You can also generate and view Report Books by running a data management sequence.

Books can be stored in locations such as a File Share, your computer desktop, or a dashboard file. When saving a report book, it is important to follow pre-defined naming conventions depending on the format of the book.

> **Example:** Bookname.xfDoc.pdfBook for PDF books.

> **Example:** Bookname.xfDoc.zipBook for zipped books

> **Example:** Bookname.xfDoc.xlBook for Excel books

> **NOTE:** Embedding fonts in a PDF Report Book could increase the size of the PDF file. Use the PDF Embedded Fonts to Remove property in the Application Server Configuration file to specify which fonts should not be embedded. This will reduce the size of PDF files and control the resolution during report book PDF generation. The default font-stack setting is: Arial; Calibri; Segoe UI; Tahoma; Times New Roman; Verdana.

Create a Report Book

You can create three types of books: Excel books, PDF books, and Zip File books. Excel books produce an Excel workbook with each piece of content on a new tab. PDF books compile all content in a single PDF file. Zip books produce a zip file where each piece of content exists individually. You can create each book type by saving it with the following file suffixes:

- **Excel books**: ReportBookName.xfDoc.xlBook
- **PDF books**: ReportBookName.xfDoc.pdfBook
- **Zip File books**: ReportBookName.xfDoc.zipBook

To create a report book, complete the following steps:

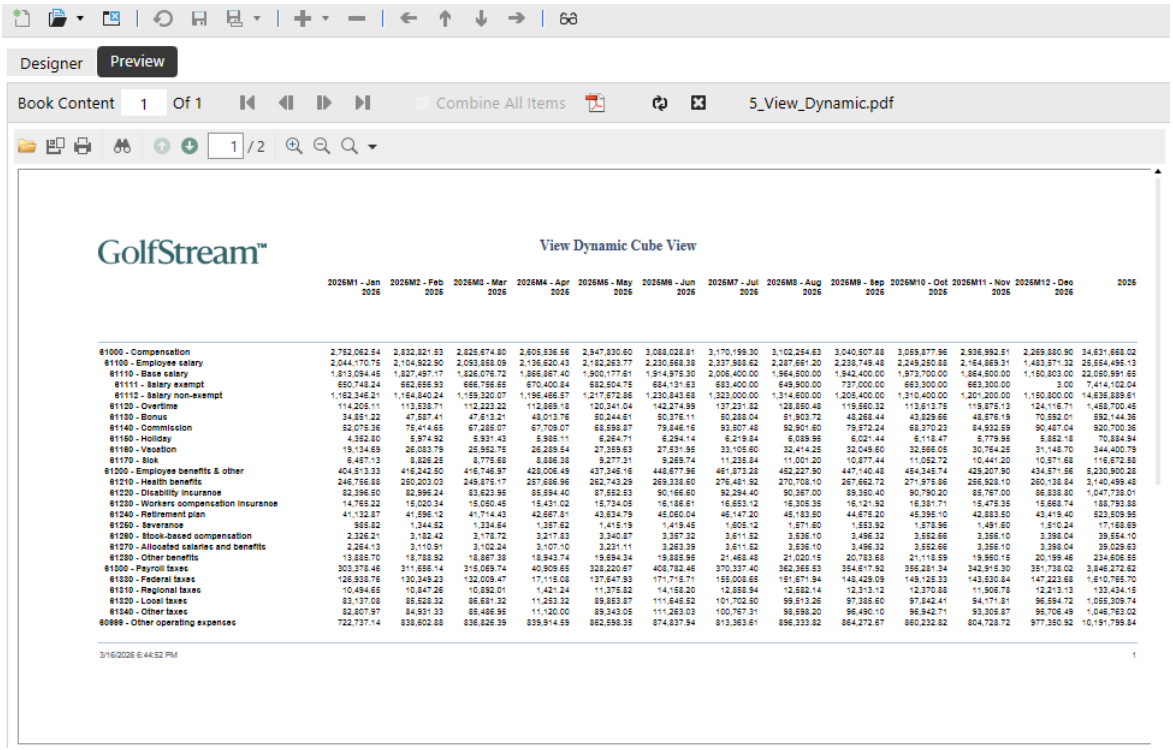
1. Go to **Application** > **Presentation** > **Books**.
2. Select the **Create New Book**  icon.
3. Set the **Determine Parameters From Content** property to either **True** or **False**. See Book Properties.

- If you set the property to False, in the **Required Input Parameters** property, type the appropriate parameters.
- If you set the property to True, leave the Required Input Parameters property blank.

4. Select the **Add Item**  icon and select items to add to the book. You can choose from the following options:
   - **Add File**: See File Type
   - **Excel Export Item**: See Excel Export Item
   - **Report**: See Report
   - **Loop**: See Loop
   - **Conditional Statement**: See If Statement and Else/Else If Statement
   - **Add Item to Change Parameters**: See Parameters

   > **NOTE:** When creating books, you can drag and drop book items. This enables you to easily move items up and down the hierarchy.

5. If you need to remove items from the Designer tab, select the **Remove Item** icon.
6. Select the **Save As**  icon and select a location. See Book Designer Toolbar.
7. Select the **Preview** tab to see the new book.



8. Select the **Close Book** icon on the toolbar when finished. You can open another book by selecting the **Open Book** icon and selecting the book type. See Book Designer Toolbar.

Book Designer Toolbar

Use the toolbar items to manage report books.



- To create a new report book, select the **Create New Book**  icon.
- To open a report book, select the **Open Book**  icon and choose from the following options:
  - **Open File From OneStream File System**: This enables you to select a file from File Explorer.
  - **Open File From Local Folder**: This enables you to select a local file on your system.
  - **Open Application Workspace File**: This enables you to select an Application Workspace file from the Object Lookup dialog box.
  - **Open System Workspace File**: This enables you to select an System Workspace file from the Object Lookup dialog box.
- To close the book you are working on, select the **Close Book**  icon.
- To cancel unsaved changes, select the **Cancel All Changes Since Last Save**  icon.
- To save your changes, select the **Save**  icon.
- To specify the kind of book you are creating and to specify the location, select the **Save As**  icon. You can choose from the following options:
  - **Save as File In OneStream File System**: This saves the book in File Explorer. It can be saved in a public folder or a local folder on your system.
  - **Save As File on Local Folder**: This saves the book in a local folder on your system.
  - **Save As Application Workspace File**: This saves the book in the file section of an Application Workspace maintenance unit.

- **Save As System Workspace File**: This saves the book in the file section of a System Workspace maintenance unit.

- To add items to the report book, select the **Add Item**  icon. You can choose from the following options:
  - **Add File**: See File Type
  - **Add Excel Export Item**: See Excel Export Item
  - **Add Report**: See Report
  - **Add Loop**: See Loop
  - **Add If Statement**: See If Statement
  - **Add Else/Else If Statement**: See Else/Else If Statement
  - **Add Item to Change Parameters**: See Parameters
- To remove an item from the report book, select the **Remove Item** icon.
- To move an item out of its parent item, select the **Move Out Of Parent**  icon.
- To move an item up in the hierarchy, select the **Move Up**  icon.
- To move an item down in the hierarchy, select the **Move Down**  icon.
- To make an item the child of the item above it, select the **Move Into Item Above**  icon.
- To find and select an object to copy, select the **Object Lookup**  icon.

Book Properties

When you are creating a new book, regardless of type, there are two book properties to consider:

- Determine Parameters from Content
- Required Input Parameters

Parameters let you determine what data should be displayed in the resulting report book. Parameters function as filters because only certain data is selected and displayed in the Report Book.

Determine Parameters from Content

When set to True, this property automatically determines the required input parameters from the book's content items. For large report books, setting this to False and manually specifying the required parameters results in a performance gain.

Required Input Parameters

This field requires a comma-separated list of parameter names. You can leave this field blank if the Determine Parameters from Content property is set to True.

Using an Item to Change Parameters

Change parameters are used to turn off the prompt that the parameter would normally generate to define the desired result of the book. The loop definition populates the items instead.

For example, imagine a Cube View that would normally prompt you for the time member to be added to the Report Book. In this case, you might not want the user to make a selection, and instead, you want the report book to run for the workflow time. Rather than copying or altering the Cube View, you would add an item to Change Parameters.

File Type

When adding a File type to the report book, complete the File properties.

File Source Type
You can select the following source types for your files:

- **URL**: Specify a file from an internal or external web page using the full URL.
- **Application Workspace File**: Select a file stored in the Application Workspace maintenance unit file section.
- **System Workspace File**: Select a file stored in the System Workspace maintenance unit file section.
- **Application Database File**: Select a file stored in the Application Database Share.
- **System Database File**: Select a file stored in the System Database Share.
- **File Share File**: Select a file from the File Share. You can add many file types, such as Word, PDF, CSV, and XLSX.

URL or Full File Name

Select the URL or name of the file you want to use. Type the full URL or select the ellipsis to browse to a file.

Output Name
Type a name for the file when creating a Zip book. If the .pdf extension is included for a Word or Excel file, the file is converted into a PDF. This property is optional and does not apply to PDF books.

Excel Export Item

Use Excel export items when creating an XLbook. You can specify a Cube View to export to an Excel file. Each Cube View displays on a separate worksheet in the book. These are useful when added to XLBooks rather than PDFBooks.

> **NOTE:** Report and File items are not supported by Excel books and are ignored if added.

To create an Excel export item, set the following properties:

Cube View Name

Select the ellipsis  to open the Object Lookup window and select a Cube View to add to the Excel Book.

Output Name

This field is optional. Type a name for the Cube View to display on the Excel worksheet. The character limit is 31.

Report

When adding a Report type to the Report Book, complete the report properties.

Report Type

Your report type selection may depend on how you saved the Report Book. You can select from the following options:

- **Cube View**: The report is based on a Cube View.
- **Dashboard Chart**: The report is based on an application dashboard chart.
- **Dashboard Report**: The report is based on an application dashboard report.
- **System Dashboard Chart**: The report is based on a system dashboard chart.
- **System Dashboard Report**: The report is based on a system dashboard report.

> **NOTE:** The chart and report items are application dashboard Chart components, rather than dashboard components.

Cube View Or Component Name
Type the name of the Cube View, dashboard report, or dashboard chart being used in the Report Book. Select the ellipsis ... and browse to the source for a report.

Output Name
Type a name for the report when it is a Zip book. This field is optional and does not apply to PDF books.

Include Report Margins/Report Header/Page Header/Report Footer/Page Footer
When set to True, the original margins, report or page headers, and report or page footers are kept. When set to False, the original formatting is removed.

Loop

A loop is a sequence of instructions that continually runs a process as many times as is defined in the loop definition. For example, a book can be set up to loop through all base entities under a particular hierarchy and generate an instance of the same Cube View report for each entity.

Loop Type
There are three types of Loops you can use when building your Report Book:

- **Comma Separated List**: Select this option to enter values separated by a comma to be referenced later in the book hierarchy.
- **Dashboard Parameter**: Select this option to use a pre-configured parameter found on the Application Workspaces page.

> **Example:** A ParamSalesRegions parameter returns a list of all Sales Regions within the application resulting in the report book's loop variables using the same list.

- **Member Filter**: Select this option to utilize the member filter builder to build a member filter based on dimension members. This loop runs each of the report book items for the specified members.

Loop Definition

When using the Comma Separated List loop type, enter a comma-separated list of text values to loop over. Use brackets when names have spaces.

> **Example:** (Houston, Clubs, [Houston Heights])

When using the Dashboard Parameter loop type, enter the name of a dashboard parameter to create a list based on an existing parameter.

> **Example:** A ParamSalesRegions parameter returns a list of all Sales Regions within the application resulting in the report book's loop variables using the same list.

When using the Member Filter loop type, enter a member filter to supply a list of members to use in the loop.

> **Example:** E#Frankfurt, E#Houston, E#Montreal. This loops over each entity and performs the process three times.

> **Example:** E#[NA Clubs].Base. This loops over each base entity under NA Clubs and performs the process for however many base entities there are.

Dimension

This property is only enabled when the Member Filter Loop type is selected. Include the name of the specific dimension used by the Member Filter. Select the ellipsis ... and choose the correct dimension.

Loop Variables

Create loop variables by selecting a variable from the drop down menu. You can choose from the following options:

- **|Loop1-4Variable|**: This enables all report book items located in the loop's hierarchy to reference the loop definition's values by name. Up to four loop variables can be referenced and all must be enclosed in pipes. Use |Loop2Variable| through |Loop4Variable| to create nested loops within a loop.

  > **Example:** |Loop1Variable|

- **|LoopDisplay1-4Variable|**: This enables all report book items in the loop hierarchy to reference the loop definition's values by description. |Use |Loop2Display| through |Loop4Display| to create nested loops within a loop.

  > **Example:** |Loop1Display|

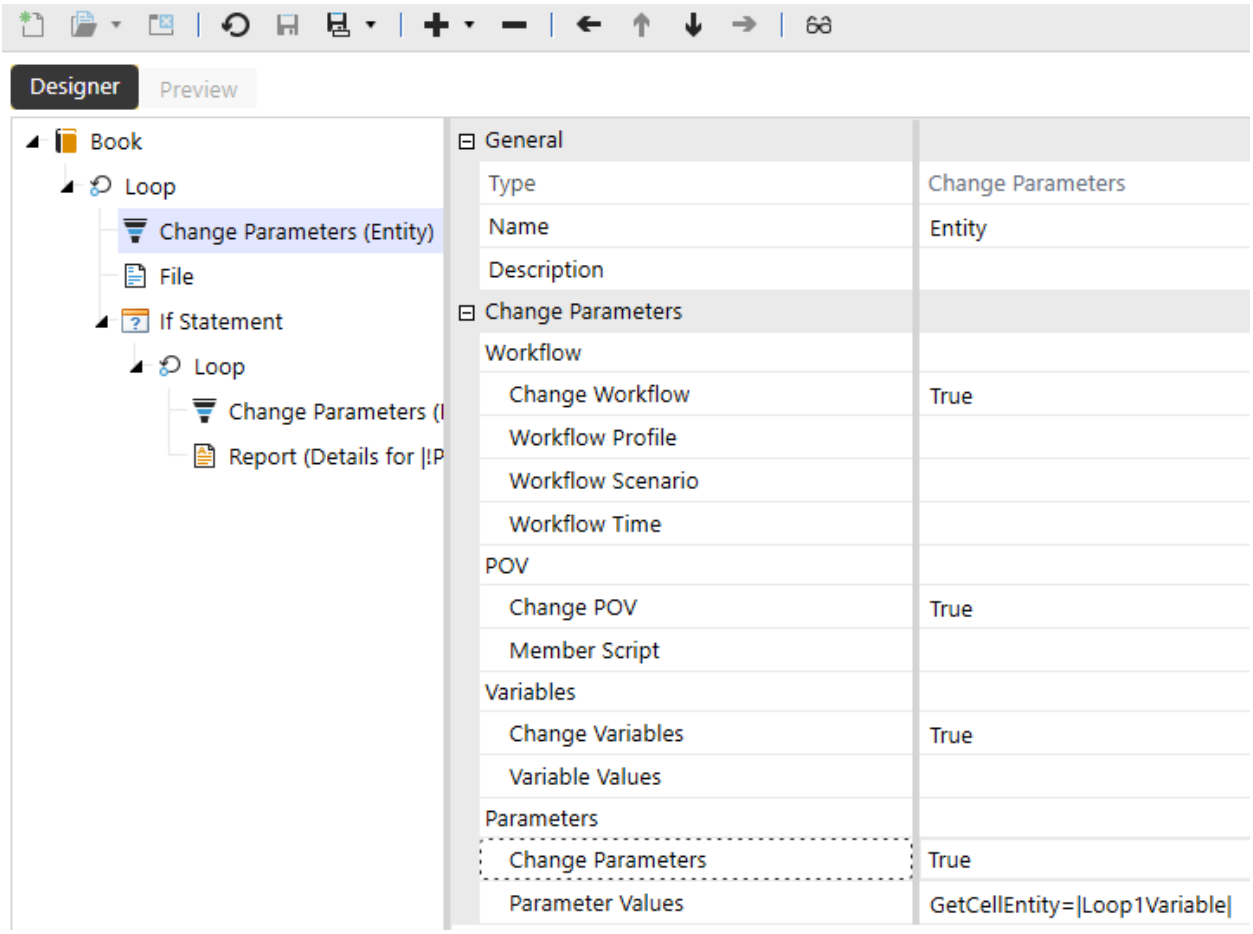
- **|Loop1-4Index|**: This assigns a number to the values in the loop definition beginning with number one, which can be referenced in the report book items in the hierarchy. Use |Loop2Index| through |Loop4Index| to create nested loops in a loop.

Change Parameters

Change parameters are used to enhance the data in report books. Setting these parameters customizes the report book output without altering the input from the source.

Loops require Change parameters to run, but Change parameters are not exclusive to loops. When a Change parameter type is encountered in a loop, the loop variable is updated to use the next ;oop variable.

For example, a loop may be used to loop through a list of entities and run a report for each. The Change parameters book Item should be added here to pass along the appropriate loop variable, such as |Loop2Variable|, which applies to each entity included in the loop.



Workflow

Change Workflow

When set to True, you can change settings for Cube Views or reports driven by Workflow POV information.

Workflow Profile

Specify the Workflow Profile name to replace the original Workflow Profile referenced in the Cube View or report.

Workflow Scenario

Specify the Workflow Profile scenario type to replace the original Workflow Profile scenario type referenced in the Cube View or report.

Workflow Time

Specify the Workflow Profile time to replace the original Workflow Profile time referenced in the Cube View or report.

POV

Change POV

When set to True, you can change the book's POV without having to change the actual Cube View or report.

> **NOTE:** To use this feature, the POV tab in the Cube View should not have any members selected.

Member Script

Select the ellipsis button ... to launch the Member Filter Builder and enter a member script to change the POV.

> **Example:** E#[|Loop1Variable|]:A#Sales. This example changes the POV for both the entity and account.

Variables

Change Variables

This serves as a placeholder that can store up to ten variables. This is valuable when If statements are used.

Variable Values

Enter a comma-separated list of name-value pairs to change the values of the predefined variables named Variable 1-10.

> **Example:** Variable1=Red, Variable2=Large, Variable3=[(Loop1Variable)]

Parameters

Change Parameters

Use this property to specify a value for a dashboard parameter found in a Cube View.

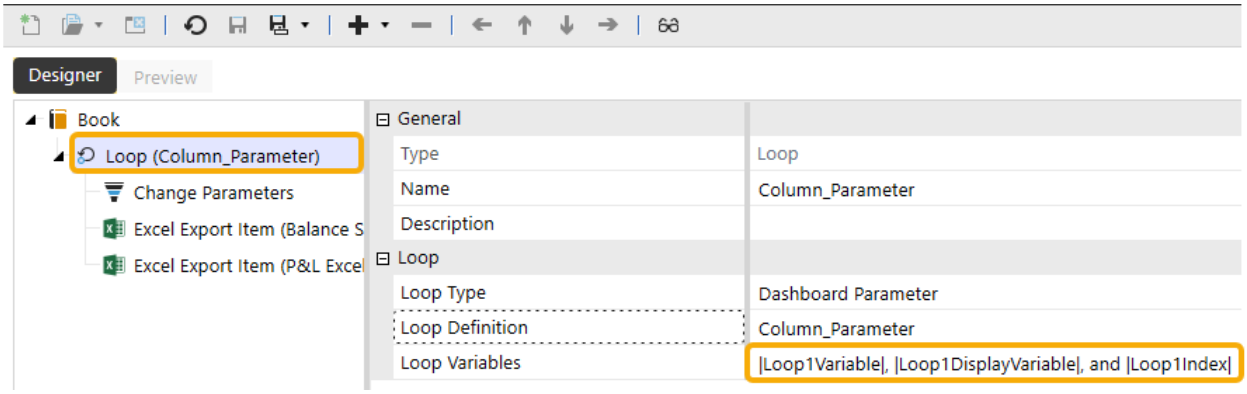
Parameter Values

Select the ellipses ... to select a parameter. Enter a comma-separated list of name-value pairs to override the custom parameter's values.

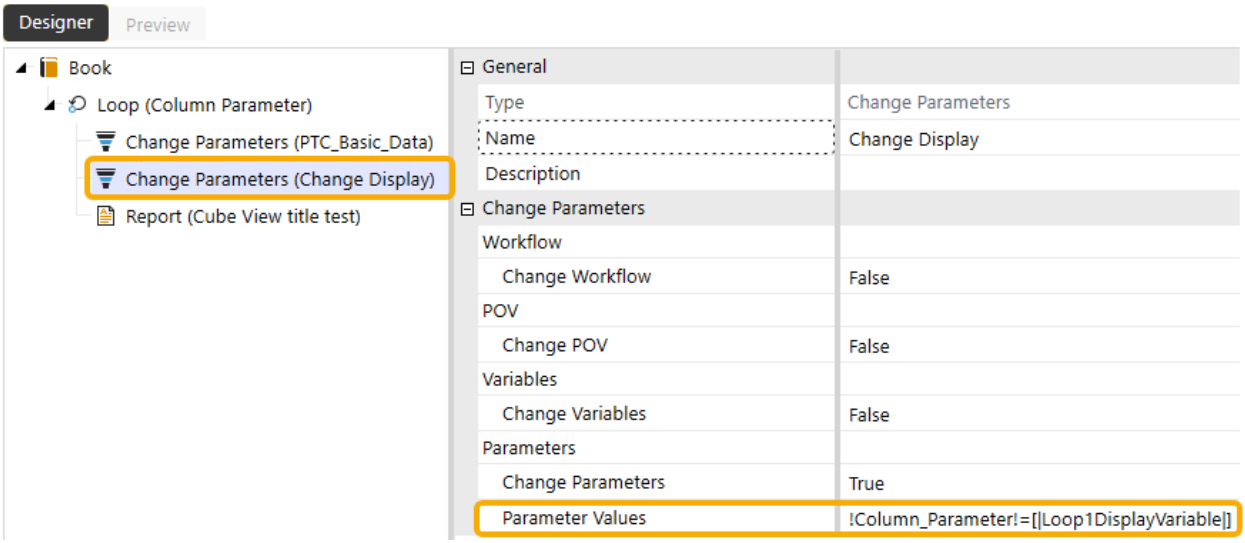
> **Example:** MyParam=Red, MyOtherParam=[|Loop1Variable|]

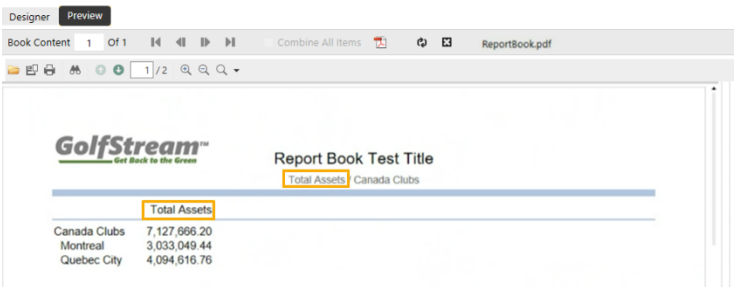
> **NOTE:** Use !! to display the Display items in a report header. Use ! to display the Value items in a report header.

The loop definition has a single Change parameters definition. Within that definition, it changes to a single variable, Loop1Variable. The Loop1DisplayVariable and Loop1Index variables never change.



To change the Report Book title based on the Cube View column results, add a second Change parameters definition, changing the Loop1DisplayVariable variable.





If Statement

If statements can be used to provide conditional logic to your Report Books. They are commonly used in tandem with loops to create a more fine-tuned output, but they can also stand on their own. Additional logic can also be added through the use of Else If and Else statements. If statements determine how book items within the hierarchy are processed.

Statement
Enter a conditional statement using parameters to determine whether the book items will be processed.

Using Variables in If Statements

The following example is an If statement that would have existing with a loop.

> **Example:** (|Loop1Variable| = [Frankfurt])

If Frankfurt is found within the Loop Definition for the Loop1Variables, the book items in this hierarchy processes for Frankfurt.

Using Parameters in If Statements

> **Example:** (|!UserName!| = Administrator)

If the user running the book has the user name Administrator, the book items in this hierarchy are included in the book.

Combining Statements

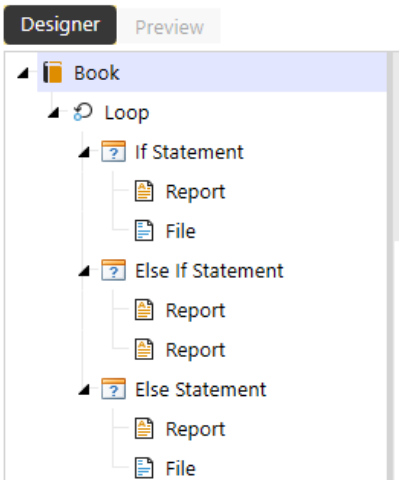
> **Example:** (|!UserName!| = Administrator) Or (|!UserName!| = JSmith)

If the user running the book has the user name Administrator or JSmith, the book items in this hierarchy are included in the book.

Else/Else If Statement

Else and Else If statements, like If statements, may or may not be used within loops to provide additional logic to the report book output. Else and Else If statements determine how book items within the hierarchy are processed. An If statement is needed to use an Else or Else If statement.

Statement
Enter a conditional statement using parameters to determine whether the child items are processed. See examples in If Statement.



> **Example:** Loop1Variable: E#[Frankfurt], E#[Europe Clubs], E#[Clubs]

If Statement

(|Loop1Variable| = [Frankfurt]). If Loop1Variable is Frankfurt, run the report and file under the If statement.

Else If Statement

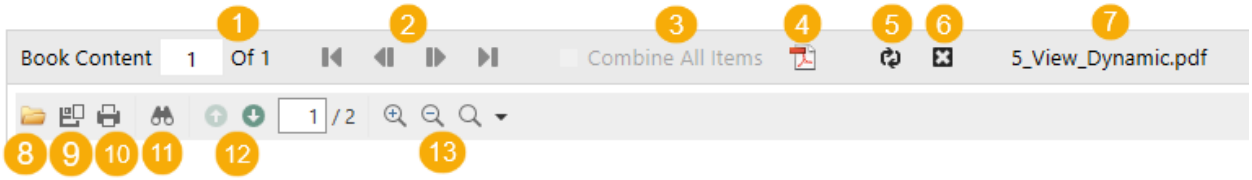
(|Loop1Variable| = [Europe Clubs]). Else If, or otherwise if, the Loop1Variable is Europe Clubs, run the report under the Else If statement.

Else

Else, in all other cases, run the report under the Else statement.

Book Preview Toolbar

The Book preview toolbar allows you to interact and navigate in the book being previewed.

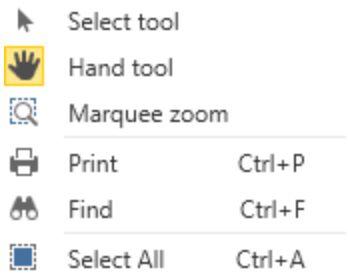


1. **Page Navigation**: This displays what page is currently being previewed in the report book. To navigate to a specific page, enter the page number and press **Enter**.
2. **First/Last and Next/Previous**: Use these icons to navigate to the first, last, next, and previous pages of the Report Book.

3. **Combine All Items**: This combines the Report Book's pages and treats them as one content item. Use this in conjunction with saving or printing an entire Report Book. If selected, you can save the entire Report Book as one PDF file, print the entire book, and navigate page by page using the blue navigation icons. If this box is deselected, only the current page will save or print, and black navigation icons are enabled.
4. **Download Combined PDF File**: This combines the entire Report Book into a PDF file using the standard Adobe rendering process and it uses less memory when attempting to display large combined PDF files. This does not require you to select the Combine All Items checkbox. This is the suggested method when downloading any report book to PDF.
5. **Refresh**: Use this icon to refresh the Report Book and select new parameter values.
6. **Close**: Use this icon to close the Report Book in the preview screen.
7. **File name**: This shows the name of the file currently being previewed.
8. **Open**: Use this icon to open a Report Book from a computer desktop or folder.
9. **Save**: Use this icon to save the current Report Book page. To save the entire Report Book as one PDF file, ensure that the Combine All Items checkbox is selected.
10. **Print**: Use this icon to open the Print dialog box. To print the entire Report Book, ensure that the Combine All Items checkbox is selected.
11. **Find**: Use this icon to find specific keywords in a Report Book.
12. **Previous Page/Next Page**: Use these icons to navigate pages of a single Report Book content item or to navigate a report book's pages when the Combine All Items checkbox is selected. If a particular content item, such as a Cube View report, is more than one page, use these icons to navigate that report.
13. **Zoom**: Use these icons to zoom in and out of a Report Book.

Right-Click Options

Right-click anywhere within the report to select these options.



Select tool: Use this to select portions of the Report Book. This feature enables you to copy and paste. Highlight a portion of the report book, press **Ctrl** + **C** and then press **Ctrl** + **V**.

Hand tool: Use this to scroll through a Report Book by selecting anywhere on the screen and moving the mouse.

Marquee Zoom: Use this tool to select an area of interest to zoom in on.

> **TIP:** You can press **Alt** + **Left** to return to the previous view or press **Ctrl** + **0** to return to a page level view.

Print: Use this to open the Print dialog box.

Find: Use this to find specific keywords in a Report Book.

Select All: Use this to select all the content in a Report Book.

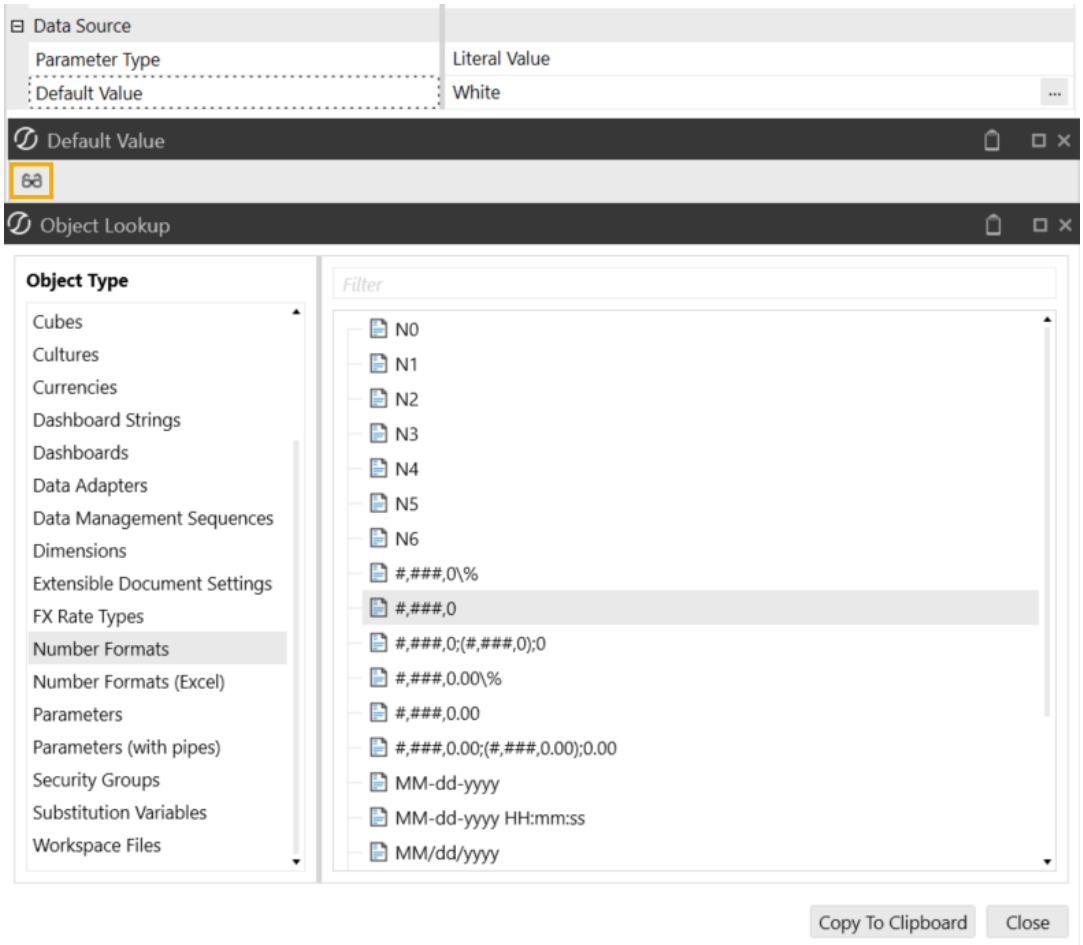
Find and Assign Values Using Object Lookup

Use the Object Lookup dialog box throughout OneStream to search for objects to assign. This eliminates the need to remember specific object names or formats. It enables you to browse application values and objects, such as business rules, colors, Cube Views, dimensions, and number formats, and then copy and paste them where needed. Instead of entering a default value, you can use Object Lookup to find and assign a value or object. The Object Lookup dialog box is also useful when you do not know the value or object to apply with a parameter or to display as the default value the first time a parameter runs.

In an Input Value parameter, the default value is the object or value the parameter applies when assigned to an object. For the other parameter types, the default value is the object or value to initially display when a parameter first runs. For example, to display a specific member at the top of a list of six members in a member list parameter, specify that member name as the default value.

For example, to use Object Lookup to add a default value to a parameter, first create the parameter, then complete the following steps:

1. Navigate to the Parameter's properties. In the **Data Source** section, in the **Default Value** property, select the ellipses ... and then select the **Object Lookup** icon.



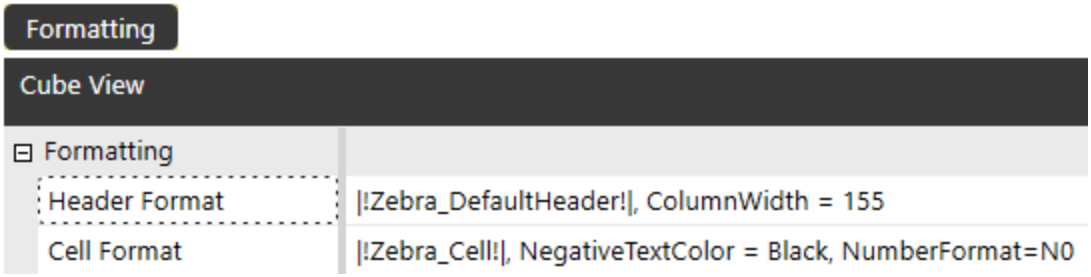
2. In the **Object Lookup** dialog box, select an object type to browse available objects or values. To refine the objects displayed, filter by name.
3. Select the object and select the **Copy To Clipboard** button.
4. In the **Default Value** dialog box, right-click, select **Paste**, and then select the **OK** button.

To remove or use another value or object, delete the contents in the **Default Value** dialog box and select **Object Lookup**.

Parameters Example

When designing a Cube View header, complete the following steps:

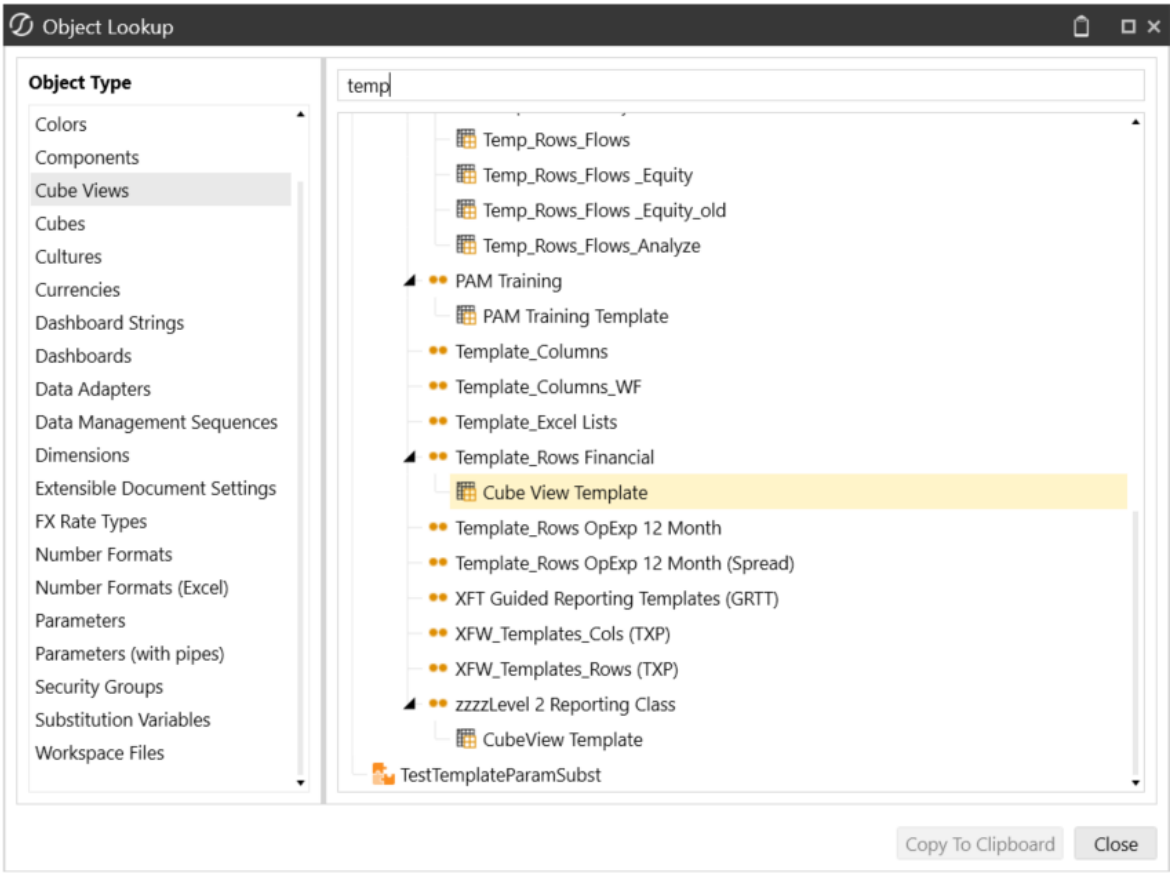
1. Select the **Object Lookup** icon and select **Parameters (with Pipes)** from the Object Type menu.
2. Select the desired Cube View styles parameter, such as DefaultHeader.
3. Select the **Copy to Clipboard** button. Alternatively, you can press **CTRL** and double-click to copy.
4. Select into the empty field where the copied parameter should go, and paste the parameter.



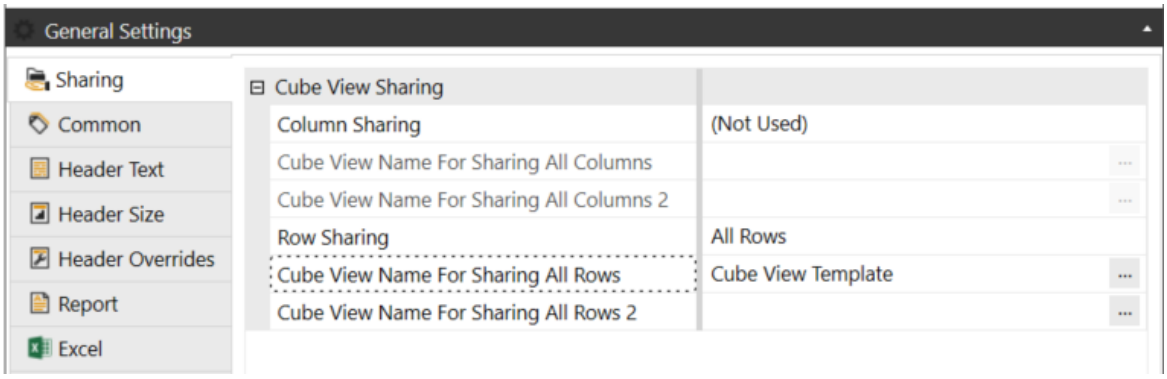
Cube Views Example

The Object Lookup dialog box can be used when using Cube View templates to create Cube View rows or columns. Complete the following steps:

1. In the Object Lookup dialog box, navigate to the desired Cube View Template.
2. Select the template and select the **Copy to Clipboard** button.



3. In the Cube View, go to **General Settings** > **Sharing** > **Cube View Sharing** > **Cube View Name For Sharing All Rows** and paste the template.



Extensible Document Settings

Each of the strings for Extensible Document Settings can be found in the Object Lookup dialog box. There is a variety of item types that can be used in Extensible Documents, however, each one needs to be configured a specific way to process correctly at run-time. An example of each item type's configuration is provided in this section. For more details on how to utilize and configure these item types in Extensible Documents, see Presenting Data With Extensible Documents.

Insert Content Using Document Variables

Cube View Report

To insert a Cube View report into an Extensible Document, the following string must be updated and pasted when configuring the document variables:"{XF}{Application}{CubeViewReport}{CubeViewName}"

The following string is optional formatting: “IncludeReportMargins=False” “IncludeReportHeader=False” “IncludeReportFooter=False” “IncludePageHeader=True” “IncludePageFooter=True”

> **IMPORTANT:** Arguments such as the "CubeViewName" are placeholders for Cube Views, reports, and files that must be updated.

> **Example:** "{XF}{Application}{CubeViewReport}{CubeViewStatement}"

Excel Sheet

To insert an Excel sheet into an Extensible Document, the following string must be updated and pasted when configuring the document variables: "{XF}{Application}{File}{Documents/Public/ExtensibleDocs/ExcelFileName.xfDoc.xlsx}"

The following string is optional formatting: “ExcelSheet=Sheet1"

> **IMPORTANT:** The specific path to the file in the File Explorer must be included and updated.

> **Example:** "{XF}{Application}{File}{Documents/Public/ExtensibleDocs/Excel4.xfDoc.xlsx} "

> **Example:** "{XF}{Application}{File}{Documents/Public/ExtensibleDocs/Excel4.xfDoc.xlsx}" "ExcelSheet=Sheet1"

Excel Named Range

To insert an Excel Named Range into an Extensible Document, the following string must be updated and pasted when configuring the document variables: "{XF}{Application}{File}{Documents/Public/ExtensibleDocs/ExcelFileName.xfDoc.xlsx}"

The following string is optional formatting: “ExcelNamedRange=SampleNamedRange"

> **Example:** "{XF}{Application}{File}
> {Documents/Public/ExtensibleDocs/Excel5.xfDoc.xlsx} "

> **Example:** "{XF}{Application}{File}
> {Documents/Public/ExtensibleDocs/Excel4.xfDoc.xlsx}"
> "ExcelNamedRange=NamedRange1"

Microsoft Word Document

To insert a Microsoft Word Document into an Extensible Document, the following string must be updated and pasted when configuring the document variables: "{XF}{Application}{File} {Documents/Public/ExtensibleDocs/WordFileName.docx}"

> **Example:** "{XF}{Application}{File}
> {Documents/Public/ExtensibleDocs/OneStreamWordDocume
> nt.docx}"

Rich Text Document

To insert a Rich Text Document into an Extensible Document, the following string must be updated and pasted when configuring the document variables: "{XF}{Application}{File} {Documents/Public/ExtensibleDocs/RTFFileName.rtf}"

> **Example:** "{XF}{Application}{File}
> {Documents/Public/ExtensibleDocs/RichTextFormatDoc.rtf}"

Text Document

To insert a Text Document into an Extensible Document, the following string must be updated and pasted when configuring the document variables: "{XF}{Application}{File} {Documents/Public/ExtensibleDocs/TextFileName.txt}"

> **Example:** "{XF}{Application}{File}
> {Documents/Public/ExtensibleDocs/TextFileData.txt}"

Report

To insert a Report into an Extensible Document, the following string must be updated and pasted when configuring the document variables: "{XF}{Application}{Report}{ReportComponentName}"

The following string is optional formatting: “IncludeReportMargins=False” “IncludeReportHeader=False” “IncludeReportFooter=False” “IncludePageHeader=True” “IncludePageFooter=True”

> **Example:** "{XF}{Application}{Report}{Report3}"

> **NOTE:** Parameters and substitution variables are not listed under Extensible Document Settings.

Insert Content Using Office Image

Chart/Chart Report

To insert a Chart or Chart Report component into an Extensible Document, the following string must be updated and pasted in the Content Control Properties dialog box: {XF}{ItemLocation}{ItemType}{ChartComponentName}

> **Example:** {XF}{Application}{Chart}{Waterfall}

> **Example:** {XF}{Application}{ChartReport}{Waterfall}

> **NOTE:** Use this for all Chart (Advanced) components.

Cube View Report

To insert a Cube View report into an Extensible Document, the following string must be updated and pasted in the Content Control Properties dialog box when configuring the image: {XF}{Application}{ItemType}{CubeViewName}

> **Example:** {XF}{Application}{CubeViewReport}{BalanceSheetSummary}

Excel Sheet/Excel Named Range

To insert an Excel Sheet or Excel Named Range into an Extensible Document, the following string must be updated and pasted in the Content Control Properties dialog box when configuring the image: {XF}{Application}{ItemType}{FilePath}

> **Example:** {XF}{Application}{ExcelFile}{Documents/Users/jsmith/Favorites/VarianceReport.xfDoc.xlsx}

> **NOTE:** For Excel Named Range Item Types, the Excel Named Range name is configured in the formatting string. For example, ExcelNamedRange=TotalAssets.

PDF

To insert a PDF into an Extensible Document, the following string must be updated and pasted in the Content Control Properties dialog box when configuring the image: {XF}{Application}{ItemType}{FilePath}

> **Example:** {XF}{Application}{FileViaPDF}{Documents/Users/jsmith/Favorites/IS.pdf}

Report

To insert a Report component into an Extensible Document, the following string must be updated and pasted in the Content Control Properties dialog box when configuring the image: {XF}{ItemLocation}{ItemType}{ReportComponentName}

> **Example:** {XF}{Application}{Report}{UserTaskActivity}

Rich Text

To insert Rich Text into an Extensible Document, the following string must be updated and pasted into the Alt Text Description field for the specific image: {XF}{Application}{RichText}{RichTextContent}

The following string is optional formatting:: PageNumber=1, CropLeft=0, CropTop=0, CropWidth=0, CropHeight=0, Zoom=100, MaintainAspectRatio=True, FillMode=Width, Anchor=TopLeft, IncludeBorders=False, BackgroundColor=#FFFFFF, MeasurementUnit=Inch, PageWidth=8.5, PageHeight=11

> **Example:** {XF}{Application}{RichText}{|!Narrative2!|} BackgroundColor=#C1E5F5, PageNumber=1, IncludeBorders=False

> **NOTE:** "Narrative2" is a parameter containing a specific rich text. You can also paste the specific rich text format in the string.

Options

The following options are available when formatting Extensible Document image content. The user can copy and paste the desired formatting string into the Description field when configuring the image. If changes must be made to the string, delete the current option and replace it with the correct one. The list below covers all formatting options for each image type.

Item Location

Application, System

Item Type

Chart, ChartReport, CubeViewReport, ExcelFile, FileViaPDF, Report

FillMode Options

Width, Height, LargestSide, SmallestSide

Anchor Options

BottomCenter/Left/Right, MiddleCenter/Left/Right, TopCenter/Left/Right

Cropping Options

This enables you to narrow in on a portion of the image before other settings are applied. The default setting is zero, which means cropping is not being used. The options are CropLeft, CropTop, CropWidth, and CropHeight.

XFCell

XFCell is a retrieve function used mainly in text documents such as Microsoft Word or PowerPoint. The specific dimension details provided in the function obtain a single cell of data from OneStream and display the updated value in an Extensible Document at run-time. Examples of common XFCell formulas are provided below. See Presenting Data With Extensible Documents for more details on how to configure XFCells in an Extensible Document.

> **Example:** XFCell(E#US:A#Sales). This pulls data for a specific entity and account.

> **Example:** XFCell(E#US:A#Sales:T#2014). Additional dimension members can be added for more specific detail.

> **NOTE:** Any dimensions not specified in the formula will come from the user's POV.

Number formatting and scaling can also be added to an XFCell.

> **Example:** XFCell(E#US:A#Sales, NumberFormat=N3, Scale=6). This example uses the number format N3 with a scaling of 6.

Culture Invariant is a default culture not associated with a specific country. It is typically used when users must convert a number to a string, but do not want the result to be different if one user is running the string on a French PC and another user is using an English PC.

> **Example:** XFCell(E#US:A#Sales, Culture=Invariant)

Include member scripts in XFCells to retrieve data for specific dimension members. The Culture User is based on the computer's Windows settings and the User settings.

> **Example:** XFCell(Memberscript, Culture=User, NumberFormat=N3, DisplayNoDataAsZero=True, Scale=3, FlipSign=True, ShowPercentageSign=False)

Use Business Rules in Extensible Documents

Business rules can also be referenced in an XFCell function. The following string must be updated to function correctly: BR#[BRName, FunctionName=yourFunctionName, Name1=Value1, AnotherName=[AnotherValue]]

> **Example:** BR#[BRName=FMK_Helper, FunctionName=GetUserSetting, SolutionID=MBL, FieldName=[Period]]