



PUBLIC

Document Version: 2024.8 – 2024-04-11

Acquiring, Preparing, and Modeling Data with SAP Datasphere

Content

- 1 Acquiring, Preparing, and Modeling Data with SAP Datasphere. 8**
- 2 Creating, Finding, and Sharing Objects. 11**
 - 2.1 Accessing SAP Datasphere Spaces. 12
 - 2.2 Repository Explorer. 12
 - 2.3 Folders. 19
 - 2.4 Packages. 20
 - 2.5 Finding and Accessing Data in the Catalog. 23
 - Evaluating Catalog Assets. 25
 - 2.6 Impact and Lineage Analysis. 34
 - 2.7 Saving and Deploying Objects. 39
 - 2.8 Modifying Objects That Have Dependent Objects. 41
 - Review the Objects That Depend on Your Table or View. 43
 - 2.9 Sharing Tables and Views To Other Spaces. 44
 - 2.10 Importing and Exporting Objects in CSN/JSON Files. 48
 - Importing Objects from a CSN/JSON File. 49
 - Exporting Objects to a CSN/JSON File. 50
 - 2.11 Importing and Exporting Objects via the Command Line. 51
 - 2.12 Enabling Currency Conversion with TCUR* Tables and Views. 52
 - 2.13 Deleting Objects. 54
- 3 Semantic Onboarding. 55**
- 4 Purchasing Data from Data Marketplace. 57**
 - 4.1 Searching and Browsing. 58
 - Overview of Filters. 59
 - Using Bookmarks. 62
 - 4.2 Evaluating your Data Product. 62
 - Testing with Sample Data. 64
 - Data Shipment. 65
 - Contract Type and Pricing Model. 65
 - 4.3 Acquiring a Data Product. 67
 - Acquiring a Free Data Product. 67
 - Acquiring a Free Data Product on Request. 68
 - Acquiring a Data Product on Request. 69
 - Acquiring a Data Product that needs a License Key. 70
 - 4.4 Managing your Data Products. 71
 - Updating your Data Products. 72

	Tracking your Deliveries.	73
4.5	Managing your Licenses.	74
4.6	Managing your Contexts.	75
4.7	Frequently Asked Questions by Data Product Consumers.	76
4.8	Glossary.	78
5	Acquiring Data in the Data Builder.	79
5.1	Importing Objects with Semantics from SAP S/4HANA, SAP BW/4HANA and SAP BW Bridge	83
	Importing Entities with Semantics from SAP S/4HANA.	83
	Importing Entities with Semantics from SAP BW/4HANA or SAP BW Bridge.	86
5.2	Importing Tables and Views from Sources.	90
	Import Remote Tables.	91
	Review and Edit Imported Table Properties.	91
	Restrict Remote Table Data Loads.	95
	Replicate Remote Table Data.	98
	Accelerate Table Data Access with In-Memory Storage.	100
	Process Source Changes in the Table Editor.	100
	Process Source Changes for Several Remote Tables.	102
	Modify or Duplicate Remote Tables.	104
5.3	Creating a Local Table.	106
	Columns.	108
	Column Data Types.	110
	Load or Delete Local Table Data.	113
	Maintain Local Table Data.	115
	Partitioning Local Tables.	117
	Capturing Delta Changes in Your Local Table.	118
5.4	Creating a Local Table from a CSV File.	122
	Column Properties and Profiling.	123
	Apply Data Transforms.	124
	Concatenate Columns.	126
	Split a Column.	126
	Extract Text to a New Column.	126
	Change the Case of a Text Column.	127
	Find and Replace Data.	127
	Filter and Delete Rows.	128
5.5	Creating a Data Flow.	128
	Add a Source.	132
	Create a Join Operator.	135
	Create a Union Operator.	137
	Create a Projection Operator.	138
	Create a Calculated Column in a Projection Operator.	139

	Create an Aggregation.	140
	Create a Script Operator.	141
	Create an Input Parameter.	142
	Add or Create a Target Table.	143
	Script Operator Python Reference.	146
	Process Source/Target Changes in the Data Flow Editor.	154
5.6	Creating a Replication Flow.	155
	Add a Source.	157
	Add a Target.	159
	Configure Your Replication Flow.	160
	Define Filters.	161
	Define Mapping.	162
	Using a Cloud Storage Provider As the Target.	163
	Using Google BigQuery As the Target.	165
	Using Apache Kafka As the Target.	168
	Using Confluent Kafka As the Target.	170
	Editing an Existing Replication Flow.	172
	Deleting a Replication Flow.	172
5.7	Creating a Transformation Flow.	173
	Create a Graphical View Transform.	175
	Create an SQL View Transform.	188
	Add or Create a Target Table.	191
	Processing Changes to Source and Target Tables.	192
5.8	Running a Flow.	196
5.9	Creating a Task Chain.	197
6	Preparing Data in the Data Builder.	210
6.1	Creating a Graphical View.	213
	Add a Source.	217
	Create a Join.	218
	Create a Union.	221
	Reorder, Rename, and Exclude Columns.	223
	Create a Column.	224
	Create a Geo-Coordinates Column.	226
	Create a Currency Conversion Column.	227
	Create an Input Parameter.	231
	Create an Association.	234
	Filter Data.	236
	Aggregate Data.	238
	Visualize the Lineage of Columns and Input Parameters in a Graphical View.	240
	Persist View Data.	242
	Apply a Data Access Control.	245

	Process Source Changes in the Graphical View Editor.	246
	Replace a Source.	248
	Edit a Custom CSN Annotation.	249
6.2	Creating an SQL View.	250
	Process Source Input Parameters in an SQL View.	254
	SQL Reference.	255
	SQL Functions Reference.	257
	SQLScript Reference.	264
	Process Source Changes in the SQL View Editor.	264
6.3	Creating an Intelligent Lookup.	265
	Prepare Input and Lookup Entities.	270
	Create an Exact Match Rule.	271
	Create a Fuzzy Match Rule.	274
	Configure the View Output by an Intelligent Lookup.	277
	Process Matched Results.	278
	Process Review Results.	279
	Process Multiple Match Results.	280
	Process Unmatched Results.	282
	Example: Harmonizing County Data for UK Charging Sites.	283
	Example: Adding Latitude and Longitude Data with a Multi-Rule Intelligent Lookup.	284
6.4	Creating an Entity-Relationship Model.	286
	Create a Table in an E/R Model.	288
	Create a View in an E/R Model.	289
	Create an Association in an E/R Model Diagram.	289
	Add Related Entities to an E/R Model Diagram.	291
6.5	Using the Source Browser.	292
	Add Objects from the Repository.	292
	Import an Object from a Connection or Other Source.	294
	Import Multiple Objects from a Connection.	296
6.6	Viewing or Previewing Data in Data Builder Objects.	297
7	Modeling Data in the Data Builder.	301
7.1	Creating a Fact.	305
	Specify Measures.	307
	Specify Semantic Types for Measures and Attributes.	309
	Exposing a View For Consumption.	314
7.2	Creating a Dimension.	314
	Specify Attributes.	315
	Set Key Columns to Uniquely Identify Records.	318
	Add a Hierarchy to a Dimension.	321
	Enable Time-Dependency for a Dimension or Text Entity.	324
7.3	Create a Text Entity for Attribute Translation.	327

	Create a Text Association.	329
7.4	Creating an External Hierarchy.	330
7.5	Creating a Hierarchy with Directory.	331
7.6	Creating an Analytic Model.	337
	Create an Analytic Model Directly From a View or Table.	340
	Add a Source.	340
	Add a Dimension.	341
	Add Measures.	343
	Add a Variable.	348
	Using the Data Preview.	350
7.7	Analytical Datasets (Deprecated).	351
	Create a Story Filter (Deprecated).	353
8	Modeling Data in the Business Builder.	355
8.1	Example for Using the Business Builder.	356
8.2	Business Builder Start Page.	357
8.3	Creating a Business Entity.	358
	Define Measures.	360
	Define Attributes.	364
	Define a Key.	365
	Define Associations.	366
	Define Input Parameters.	367
	Add an External Hierarchy.	368
8.4	Authorization Scenario.	368
	Creating an Authorization Scenario.	369
	Assigning an Authorization Scenario.	370
	Using an Authorization Scenario in a Consumption Model.	371
8.5	Creating a Fact Model.	372
	Define Measures.	373
	Define Attributes.	374
	Expose Dimension Sources.	374
	Define Filters.	375
8.6	Creating a Consumption Model.	376
	Define Attributes.	377
	Define Measures.	379
	Add an External Hierarchy.	379
	Define Filters.	380
	Define Perspectives.	380
8.7	Change the Data Source of a Business Entity.	383
8.8	Create Versions of an Object.	384
8.9	Previewing Data in Business Builder Objects.	385
8.10	Saving and Deploying Data Objects.	385

- 8.11 Importing SAP BW/4HANA Models. 386
 - Imported Objects. 388





- 9 SQL and SQLScript Reference. 390**
- 9.1 SQL Reference. 390
- 9.2 SQL Functions Reference. 392
- 9.3 SQLScript Reference. 399

1 Acquiring, Preparing, and Modeling Data with SAP Datasphere

Users with the *DW Modeler* role can bring data into the *Data Builder*, combine it (including with external data from the *Data Marketplace*) and prepare it for consumption in SAP Analytics Cloud and other BI clients either directly or after further modeling in the *Business Builder*.



→ Tip

The English version of this guide is open for contributions and feedback using GitHub. This allows you to get in contact with responsible authors of SAP Help Portal pages and the development team to discuss documentation-related issues. To contribute to this guide, or to provide feedback, choose the corresponding option on SAP Help Portal:

- [Feedback](#)  [Edit page](#) : Contribute to a documentation page. This option opens a pull request on GitHub.
- [Feedback](#)  [Create issue](#) : Provide feedback about a documentation page. This option opens an issue on GitHub.

You need a GitHub account to use these options.

More information:

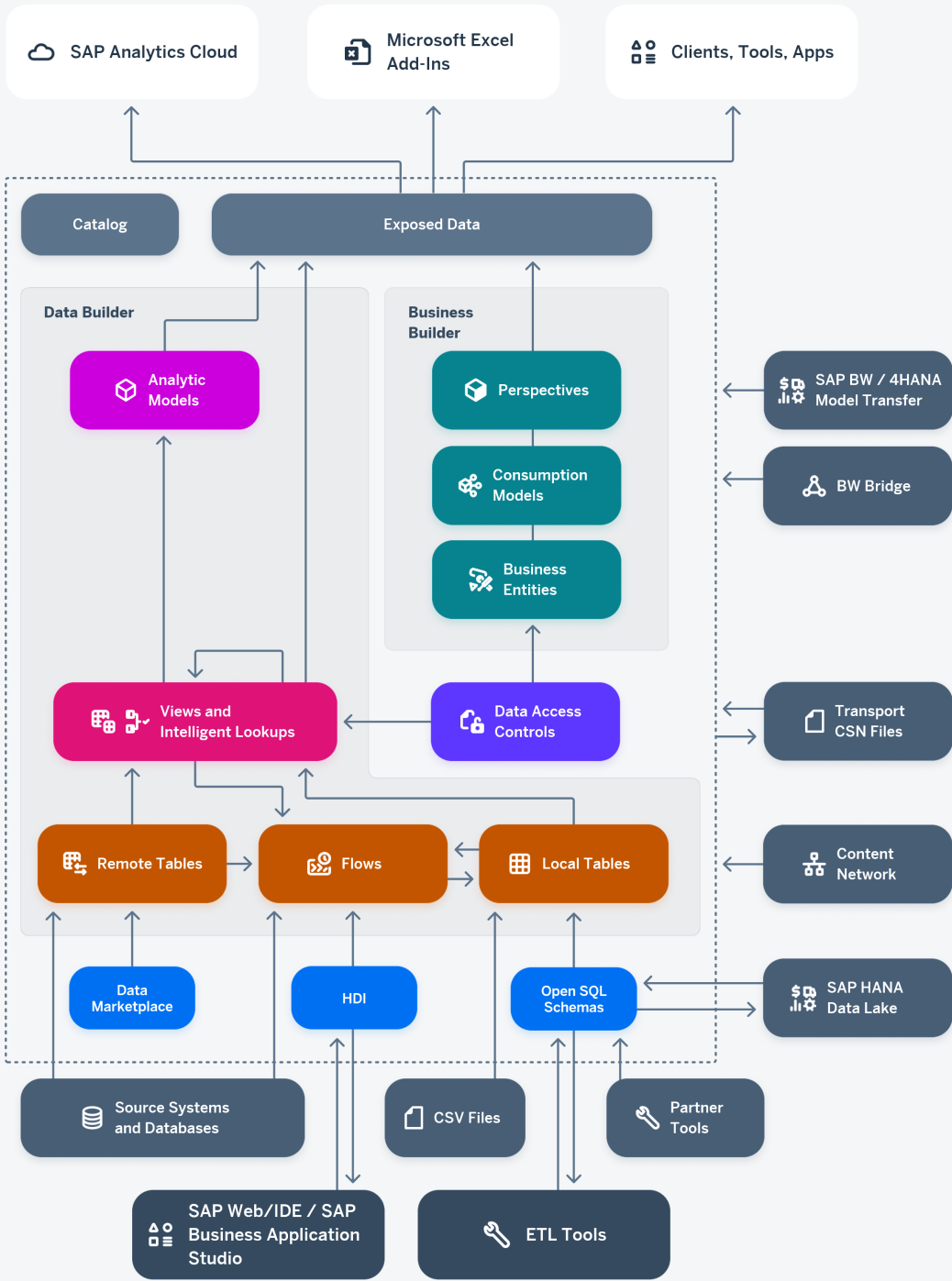
- [Contribution Guidelines](#)
- [Introduction Video: Open Documentation Initiative](#) 
- [Blog Post: Introducing the Open Documentation Initiative](#) 

This diagram shows how you acquire data from sources, prepare and model it in SAP Datasphere, and expose it for consumption in SAP Analytics Cloud, Microsoft Excel, and other clients, tools, and apps.

SAP Datasphere Overview

Acquiring, Preparing, and Modeling Data for Consumption

Select the **colored boxes** to jump to related help articles



- **Creating, Finding, and Sharing Objects** - All SAP Datasphere users use the *Repository Explorer* to browse objects. Modelers can create objects and otherwise act on them here or in the dedicated builder apps. Users with the *Catalog User* role can, additionally, browse the *Catalog* for trusted data assets that have been published there. For more information, see [Creating, Finding, and Sharing Objects \[page 11\]](#).
- **Purchasing External Data** - Users with the *DW Modeler* role can purchase external data from the *Data Marketplace* to combine with and enrich their internal data. For more information, see [Purchasing Data from Data Marketplace \[page 57\]](#).
- **Acquiring and Preparing Data** - Users with the *DW Modeler* role can import data directly into the *Data Builder* from connections and other sources, and use flows to replicate, extract, transform and load data. For more information, see [Acquiring Data in the Data Builder \[page 79\]](#).
- **Modeling Data in the Data Builder** - Users with the *DW Modeler* role can add semantic information to their entities and expose them directly to clients, tools, and apps, or combine, refine, and enrich them in tightly-focused analytic models for consumption in SAP Analytics Cloud, MS Excel, and other clients, apps, and tools. For more information, see [Modeling Data in the Data Builder \[page 301\]](#).
- **Modeling Data in the Business Builder** - Users with the *DW Modeler* role can use the *Business Builder* editors to combine, refine, and enrich *Data Builder* objects and expose lightweight, tightly-focused perspectives for consumption by SAP Analytics Cloud and MS Excel. For more information, see [Modeling Data in the Business Builder \[page 355\]](#).

2 Creating, Finding, and Sharing Objects

All SAP Datasphere users use the *Repository Explorer* to browse objects. Modelers can create objects and otherwise act on them here or in the dedicated builder apps. Users with the *Catalog User* role can, additionally, browse the *Catalog* for trusted data assets that have been published there.

This topic contains the following sections:

- [Create or Import Objects \[page 11\]](#)
- [Find Objects \[page 11\]](#)
- [Share Objects \[page 11\]](#)

Create or Import Objects

You can create objects in the *Repository Explorer* (see [Repository Explorer \[page 12\]](#)) or in the dedicated builders.

You can also import objects from CSN/JSON files (see [Importing and Exporting Objects in CSN/JSON Files \[page 48\]](#)), including via the command line (see [Importing and Exporting Objects via the Command Line \[page 51\]](#)).

Find Objects

You can search across all the objects in all the spaces you're assigned to in the *Repository Explorer* (see [Repository Explorer \[page 12\]](#)).

Alternatively, you can browse for trusted data assets that have been published to the *Catalog* (see [Finding and Accessing Data in the Catalog \[page 23\]](#)).

Share Objects

Only users assigned to a space can view objects in the space. To make a table or view in your space available to users assigned to another space, you must share it there (see [Sharing Tables and Views To Other Spaces \[page 44\]](#)).

2.1 Accessing SAP Datasphere Spaces

All data acquisition, preparation, and modeling in SAP Datasphere happens inside spaces. A space is a secure area - space data cannot be accessed outside the space unless it is shared to another space or exposed for consumption.

You may be assigned to one or more spaces. For each space you are assigned to, you can see all the objects in that space.

Many apps, such as the *Data Builder*, require you to select a space before you can open the app. When you select a space, the app will then show you only the relevant objects that belong to that space. The *Repository Explorer*, by contrast shows you objects from all the spaces you are assigned to, in order to give you a global view of all your objects.

2.2 Repository Explorer

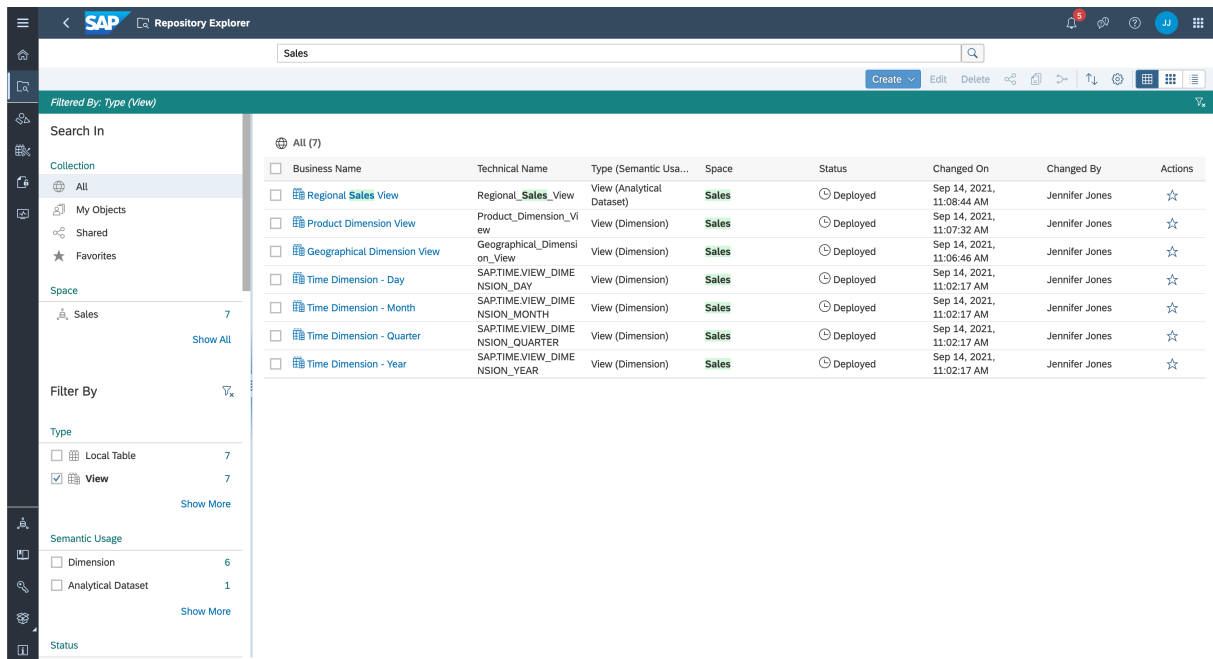
The *Repository Explorer* gives you access to all your SAP Datasphere objects. You can search and filter the list, open or act on existing objects, and create new objects.

This topic contains the following sections:

- [Open the Repository Explorer \[page 12\]](#)
- [Enter a String to Search On \[page 13\]](#)
- [Filter by Collection, Space, or Folder \[page 13\]](#)
- [Search Mode vs List Mode \[page 14\]](#)
- [Filter by Criteria \[page 15\]](#)
- [Define Advanced Filter Conditions \[page 16\]](#)
- [Favorite Objects \[page 17\]](#)
- [Create Objects and Act on Existing Objects \[page 17\]](#)

Open the Repository Explorer

Click *Repository Explorer* in the left navigation area. There is no need to select a space. The *Explorer* shows you all the objects in all the spaces you are assigned to.



Enter a String to Search On

Enter one or more characters in the *Search* field and press *Enter* (or click *Search*).

As you type, the field will begin proposing objects and search strings. Select an object to open it directly. Click on a string to trigger a search on it.

The search is case-insensitive and automatically applies wildcards so that, for example, the string "lender" will find objects containing both "lender" and "calendar".

Filter by Collection, Space, or Folder

Restrict the scope of the list by selecting a collection, space, or folder in the *Search In* area of the left panel.

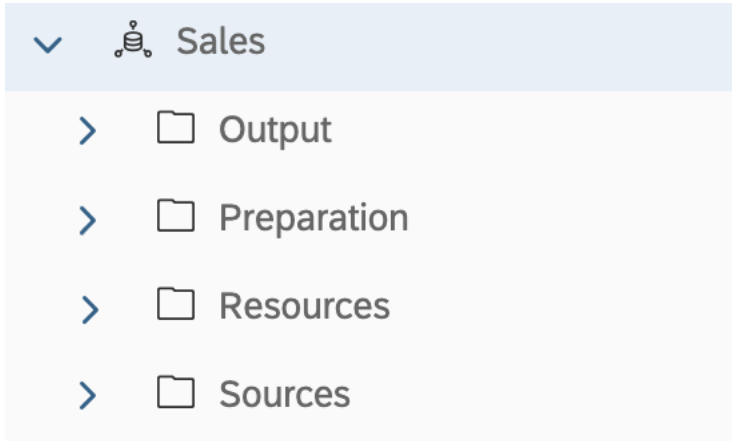
Choose one collection, space, or folder:

- *Collection*:
 - *All* (default)
 - *Recent* - Objects that you recently opened
 - *My Objects* - Objects that you created
 - *Shared* - Objects that are shared from/to their space to/from other spaces

Note

If you are not assigned to a space an object is shared from, it appears greyed out and you cannot open or otherwise act on it.

- *Favorites* - Objects that you have favorited
- *Space*: Spaces you are assigned to are listed here in order of your last visit. Click a space to show only objects contained in that space. If your space contains folders, you can drill down into the folder structure. Click a folder to show only objects contained in that folder.



Search Mode vs List Mode

By default, when you enter the *Repository Explorer*, you are searching the *All* collection, which shows all the objects you are permitted to see across all your spaces (and folders) in a flat list. In this example, we see that we are searching in the *All* collection, and 462 objects are shown:

Search In: All (462)					
<input type="checkbox"/> Business Name	Technical Name	Type (Semantic Usage)	Space	Folder	
<input type="checkbox"/> SALES EU	SALES_EU	Remote Table (Relational Dataset)	Sales	-	
<input type="checkbox"/> SALES US	SALES_US	Remote Table (Relational Dataset)	Sales	-	
<input type="checkbox"/> Asian Sales View	Asian Sales View	View (Relational Dataset)	Sales	-	

If you click a space or folder, then the display changes to list only the objects contained directly in that space or folder. To see the contents of a sub-folder, click it to go down into it.

Here, we see that there are only the two objects contained directly in the *Sales > Resources* folder, both of which are sub-folders:

Search In: "Resources"

+ ▾ ↗ ▾ ✎ 🗑

List Objects In: [All](#) > [Sales](#) > [Resources](#) (2)

<input type="checkbox"/> Business Name	Technical Name	Type (Semantic Usage)	Space	Folder
<input type="checkbox"/> Currency	-	Folder	Sales	Resources
<input type="checkbox"/> Time	-	Folder	Sales	Resources

If you then search for a string (or apply a filter), the display changes back to searching in the current space or folder (and any sub-folders) and presents the results in a flat list. Here, we see that we are now searching for the string `time` in the [Sales > Resources](#) folder. The folder structure is no longer displayed, and nine objects are shown in a flat list, including the [Time](#) folder, and a number of objects inside it:

time

+ ▾ ↗ ▾ ✎ 🗑

Search In: [All](#) > [Sales](#) > [Resources](#) (9)

<input type="checkbox"/> Business Name	Technical Name	Type (Semantic Usage)	Space	Folder
<input type="checkbox"/> Time Table	SAP.TIME.M_TIME_DIMENSION	Local Table (Relational Dataset)	Sales	Time
<input type="checkbox"/> Time Dimension - Month	SAP.TIME.VIEW_DIMENSION_MONTH	View (Dimension)	Sales	Time
<input type="checkbox"/> Time	-	Folder	Sales	Resources
<input type="checkbox"/> Time Dimension - Day	SAP.TIME.VIEW_DIMENSION_DAY	View (Dimension)	Sales	Time

Filter by Criteria

Filter by any of the categories listed in the [Filter By](#) area of the left panel.

You can select one or more values in each filter category in the [Filter By](#) section:

- Each value selected in a category acts as an OR condition.
- Values selected in separate categories act together as AND conditions.

For example, to:

To Display	Select Filter Criteria
All objects with a semantic usage of <i>Fact</i>	Semantic Usage: Fact
All objects with a semantic usage of <i>Fact</i> or <i>Dimension</i>	Semantic Usage: Fact, Dimension

To Display	Select Filter Criteria
Only views with a semantic usage of <i>Fact</i> or <i>Dimension</i>	▶ <i>Type</i> ▶ <i>View</i> ▶ and <i>Semantic Usage: Fact, Dimension</i>

Filtering for shared objects changes when you select a space:

To Display	Select Filter Criteria
All objects that are shared with any other spaces	▶ <i>Sharing</i> ▶ <i>Shared</i> ▶
All objects that are shared from a particular space	▶ <i>Space</i> ▶ <i><Space></i> ▶ and ▶ <i>Sharing</i> ▶ <i>Shared from My Space</i> ▶
All objects that are shared with a particular space	▶ <i>Space</i> ▶ <i><Space></i> ▶ and ▶ <i>Sharing</i> ▶ <i>Shared to My Space</i> ▶

Note

If you are not assigned to a space an object is shared from, it appears greyed out and you cannot open or otherwise act on it.

Define Advanced Filter Conditions

For those filter options that are not related to date or time, you can create a custom filter with specific conditions.

1. Click *Show More* at the bottom of a filter category to open the Filter Settings dialog. Some filters have a *Select Items* and a *Define Conditions* tab.
2. On the *Define Conditions* tab, choose an operator and enter a value in the *Filter Condition* box.
3. Click the plus icon to create an alternative condition. Each condition in one category acts as an **OR** operator, so that an object must meet one of the conditions to be included in the search results. If you define one condition in two categories, then each category acts as an **AND** operator, so that both conditions must be true for the object to be returned in the search results.

You can create advanced filter conditions for multiple filter categories. An object must meet one of the conditions in each of the categories to be included in the search results.

Favorite Objects

To add an object to your favorites, click the ☆ (*Add to Favorites*) tool in the *Actions* column.

To view your favorites, select *Favorites* in the *Collection* list

Create Objects and Act on Existing Objects

You can act on objects in the list in the following ways:

- Click the *Business Name* of an object to open it in its editor.
- Click the *Favorite* tool in the *Actions* column to add an object to your *Favorites* collection.
- Select one or more objects and use any of the following tools:

Note

If you select an object that is shared from a space you're not assigned to, you cannot access any of these tools.

Tool	Description
Create	Create a new object (independent of any selection) and open it in the appropriate editor.
Import	Import CSV files or objects from files or connections. For more information, see: <ul style="list-style-type: none">• Creating a Local Table from a CSV File [page 122]• Importing Objects from a CSN/JSON File [page 49]• Importing Objects with Semantics from SAP S/4HANA, SAP BW/4HANA and SAP BW Bridge [page 83]• Import Remote Tables [page 91]• Process Source Changes for Several Remote Tables [page 102]
Edit	Open the selected object in the appropriate editor.
Delete	Delete the selected objects. Allows multi-selection. If the object is used by one or more other objects then a dialog listing these dependencies opens, and the deletion is canceled.

Note

If you want to delete a remote table with a data access of *Replicated (Real-Time)*, you must ensure that:

- The data provisioning agent is connected.
- The real-time replication is not paused and is working normally.

If either of these requirements is not met, you must remove the replicated data before you can delete the remote table.

Tool	Description
Deploy	<p>Deploy the selected objects. Allows multi-selection of supported object types from a single space.</p> <p>Only the following object types can be deployed here:</p> <ul style="list-style-type: none"> • Local Table (see Creating a Local Table [page 106]) • Graphical View (see Creating a Graphical View [page 213]) • SQL View (see Creating an SQL View [page 250]) • Data Access Control (see Create a "Single Values" Data Access Control) • Analytic Model (see Creating an Analytic Model [page 337]) • Task Chain (see Creating a Task Chain [page 197]) <p>Other types of objects can only be deployed from their editors.</p> <p>If one or more objects that you have selected cannot be deployed, the <i>Deploy</i> dialog opens, allowing you to review your selection. Click <i>Deploy</i> to deploy those objects listed on the Deployable tab, or <i>Cancel</i> to go back and alter your selection.</p>
Share	<p>Share the selected objects to other spaces. Allows multi-selection from a single space.</p> <p>For more information, see Sharing Tables and Views To Other Spaces [page 44].</p>
Move To	<p>Move the selected objects to another folder. Allows multi-selection from a single space.</p> <p>For more information, see Folders [page 19].</p>
Copy	<p>Create a copy of the selected object in the same space. You must specify a new business and technical name.</p>
Impact and Lineage Analysis	<p>Open the <i>Impact and Lineage Analysis</i> dialog for the selected object to see the objects on which it depends and the objects that depend on it.</p> <p>For more information, see Impact and Lineage Analysis [page 34].</p>
Sort	<p>Open the <i>Sort</i> dialog to control the ordering of the results table.</p> <p>By default, the table is sorted by <i>Best Match on Top</i>, which calculates relevance on a range of criteria, including objects that you have recently changed, those that you have created, and those that have validation errors. To sort on a specific column, select a <i>Sort Order</i> and a <i>Sort By</i> column, and then click <i>OK</i> to apply them.</p>
Select Columns	<p>Open the <i>Columns</i> dialog to control the display of columns in the results table.</p> <p>Modify the column list in any of the following ways, and then click <i>OK</i> to apply your changes:</p> <ul style="list-style-type: none"> • To select a column for display, select its checkbox. To hide a column deselect its checkbox. • Click on a column token to highlight it and use the arrow buttons to move it in the list. • Click <i>Reset</i> to go back to the default column display.
Display as...	<p>Set the presentation of the object list to <i>Table</i> (default), <i>Grid</i>, or <i>List</i>.</p>

2.3 Folders

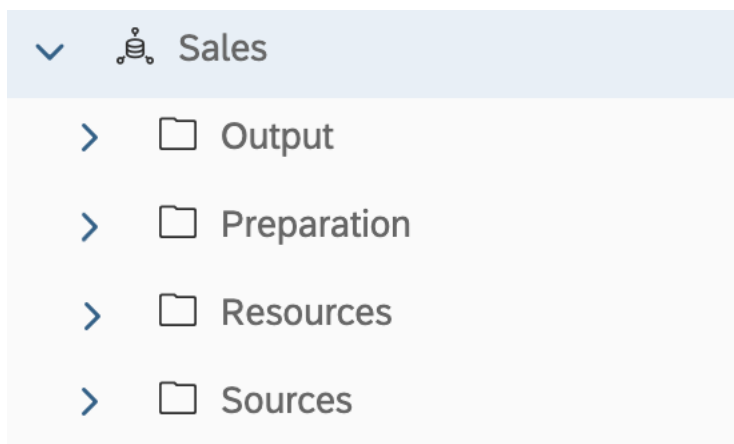
Users with the *DW Modeler* role (or equivalent privileges) can create folders to organize the objects in their spaces.

This topic contains the following sections:

- [Introduction to Folders \[page 19\]](#)
- [Create a Folder \[page 19\]](#)
- [Move Objects into a Folder \[page 20\]](#)
- [Delete a Folder \[page 20\]](#)

Introduction to Folders

Folders help to organize objects in your space. You can create a folder in the *Repository Explorer*, *Data Builder*, or *Business Builder* and it is immediately available across each of these apps.



Once a folder is created, you can move objects to it or create objects in it. You can create as many levels of folder as necessary.

Note

Each object in a space must have a unique technical name. You may not create two views with the name *Sales_US* in the *Sales* space, even if they are saved in different folders.

Create a Folder

Users with the *DW Modeler* role (or equivalent privileges) can create folders:

- In the *Repository Explorer*, click **Create > Folder**, select a space if necessary, enter a name, and click *Create*.

- In the *Data Builder*, click **► Create ► Folder**, enter a name, and click *Create*.
- In the *Business Builder*, click **► Folder ► New Folder**, enter a name, and click *Create*.

Move Objects into a Folder

Users with the *DW Modeler* role (or equivalent privileges) can move objects into folders:

- In the *Repository Explorer*, select the objects you want to move, click *Move To*, select your target folder, and click *Move*.
- In the *Data Builder*, select the objects you want to move, click *Move To*, select your target folder, and click *Move*.
- In the *Business Builder*, select the objects you want to move, click **► Folder ► Move**, select your target folder, and click *Apply*.

Note

You cannot move objects that are shared to your space into folders.

Delete a Folder

Users with the *DW Modeler* role (or equivalent privileges) can delete folders:

- In the *Repository Explorer*, select the folders you want to delete, click *Delete*, and then click *Delete* again to confirm the deletion.
- In the *Data Builder*, select the folders you want to delete, click *Delete*, and then click *Delete* again to confirm the deletion..
- In the *Business Builder*, select the folders you want to delete, click *Delete*, and then click *Delete* again to confirm the deletion.

Note

Deleting a folder will delete all its contents. This action cannot be undone.

2.4 Packages

Users with the *DW Space Administrator* role can create packages to model groups of related objects for transport between tenants. Modelers can add objects to packages via the *Package* field, which appears in editors when a package is created in their space. Once a package is complete and validated, the space administrator can export it to the Content Network. The structure of your package is preserved and, as the objects it contains evolve, you can easily export updated versions of it.

This topic contains the following sections:

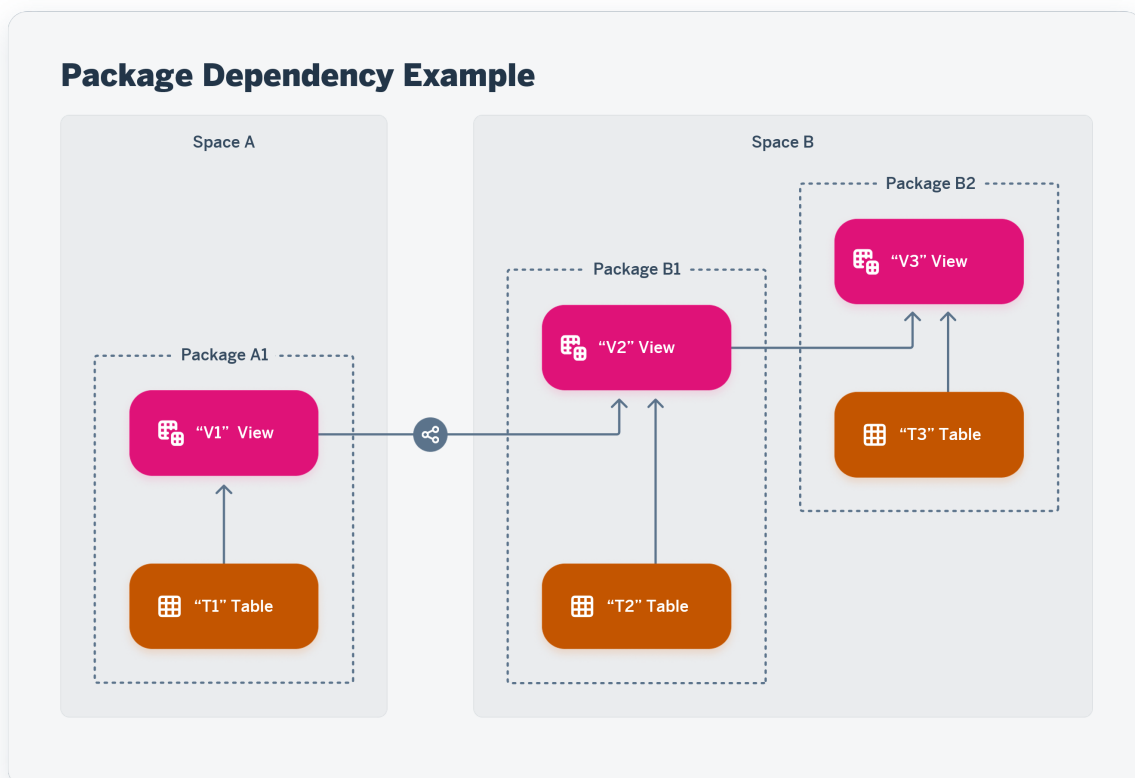
- [Introduction to Packages \[page 21\]](#)
- [Create a Package \[page 22\]](#)
- [Add an Object to a Package \[page 22\]](#)
- [Update an Object in a Package \[page 23\]](#)

Introduction to Packages

Each package must contain a complete and coherent set of objects:

- A package can contain one or more objects from one space.
- Each object can belong to only one package.
- A package must:
 - Contain the complete lineage of all the objects that belong to it, or
 - Include other packages containing all or part of this lineage as required packages.
- Required packages can be selected from other spaces.

If an object in a package has, as part of its lineage, an object shared from another space, then the shared object must be included in a package defined in its origin space, which is then added as a required package.



In our example:

- Package A1 - Contains T1 and v1, where v1 has a single source, T1.

- Package B1 - Contains T2 and V2, where V2 has two sources, T2 and V1.
 - Package B1 adds Package A1 as a required package.
- Package B2 - Contains T3, and V3, where V3 has two sources, T3 and V2.
 - Package B2 adds Package B1 as a required package.

Create a Package

Only a user with the *DW Space Administrator* role (or equivalent privileges) can create a package (see [Creating Packages to Export](#)).

Once one or more packages is created in your space, the *Package* property becomes available in the table, view, ER model and analytic model editors, and modelers can add their objects to the package (see [Add an Object to a Package \[page 22\]](#)).

Add an Object to a Package

Users with the *DW Space Administrator* role (or equivalent privileges) can add any object to a package using the package editor (see [Creating Packages to Export](#)).

Modelers can add tables, view, ER models, and analytic models to a package from the relevant editor:

1. Open your object in its editor.
2. In the property sheet, select the package from the *Package* field.

Note

The *Package* field is only available if one or more packages has been created in the space.

A warning is displayed, explaining that you have to save your object to confirm and validate the package assignment.

3. Click *Save* and then, when the *Validation Messages* dialog opens, click *Save Anyway*.
The object is saved and the updated package is validated to determine if the full lineage of your object is assigned to the package (or to one of its required packages):
 - If the validation is successful, the warning is removed.
 - If the validation is not successful, a new warning is displayed explaining that the dependencies of your object cannot be resolved in the package. You should then add missing dependencies to the package or contact a space administrator to ask them to review and resolve the situation.

Note

Whether your object's dependencies can be resolved or not, the package assignment becomes read-only and can no longer be modified in the object editor. Only a space administrator can remove an object from a package in the package editor (see [Creating Packages to Export](#)).

Update an Object in a Package

Modelers can freely update objects that are assigned to a package (though they may not change the package assignment). When changes to an object are made, the status of the package is changed to *Changes to Export*. A space administrator can choose to export the new changes at any time, either overwriting the current version or creating a new version in the Content Network.

2.5 Finding and Accessing Data in the Catalog

Discover data by searching and filtering results. Mark your favorite assets, listed data products, terms, and key performance indicators (KPIs).

The catalog provides an effective data governance strategy by bringing together an organized inventory of business metadata and data assets to enable business and technical users to unlock the full potential of their enterprise data. The catalog is a central place to discover, classify, understand, and prepare all the data in your enterprise.

Use the catalog to find the single-source of truth for your data domain to build reusable business models. With the catalog, you can find which stories are impacted by your changes. It all begins by finding the data you need.

You can search for assets by clicking  (*Catalog*) in the side navigation menu.

Note

You must be assigned one of the following:

- The **Catalog User** role.
- A custom role with the *Read* permission for *Catalog Asset*, *Catalog Tag Hierarchy*, *Catalog Glossary Object*, and *Catalog KPI Object*.

Search by Entering a String

You can find objects globally by using the search bar and entering all or part of the characters in a term, asset, listed data product, or KPI. Enter one or more characters in the *Search* field and press *Enter* (or click *Search*).


As you type, the field will begin proposing objects and search strings. Select an object to open it directly. Click on a string to trigger a search on it.

Filter the search results based on selected criteria to limit the number of results.

There are five tabs next to the *Filter* icon that you can use to narrow your results:

Tab	Description
All	All the objects in the catalog are shown.

Tab	Description
Assets	Shows only assets. An asset is any data or analytic object made available in the catalog. Assets are the objects that you provide governance around and publish to the catalog for users to discover, evaluate, and ultimately open and start using for their work.
Data Products	Shows only listed data products. Data products must be listed in Data Marketplace before they appear in the catalog. A data product is either free or purchased data from a third-party provider that you can use in this product.
Terms	Shows only terms. Terms are contained in a business glossary and provide meaning to your data. The business glossary provides a central and shared repository for defining terms and describing how and where they are used in the business.
KPIs	Shows only key performance indicators (KPIs). KPIs measure progress towards a result such as a goal or objective. Use KPIs to track performance and provide an analytical basis for decision-making.
Favorites	Shows the assets, terms, and KPIs that have been marked as a favorite.
Recent	Shows the assets, terms, and KPIs that have been recently added to the catalog.

You can also open a side panel when you click  (*Show filters*) to filter by these categories and more. Click [Show More](#) to open a dialog with additional filter options.

Filter by Criteria

Filter by any of the categories listed in the *Filter By* area of the left panel.

You can select one or more values in each filter category in the *Filter By* section:

- Each value selected in a category acts as an **OR** condition.
- Each value selected in separate categories acts together as **AND** conditions.

To Display	Select Filter Criteria
Assets that are a <i>Local Table</i>	<i>Type: Local Table</i>
Assets that are a <i>Local Table</i> or <i>View</i>	<i>Type: Local Table, View</i>
Assets that are a <i>Local Table</i> or <i>View</i> and on the system <i>SAP Datasphere</i>	<i>Type: Story, View</i> <i>System Type: SAP Datasphere</i>

Filter Options

Define Advanced Filter Conditions

For those filter options that are not related to date or time, you can create a custom filter with specific conditions.

1. Click [Show More](#) at the bottom of a filter category to open the Filter Settings dialog. Some filters have a [Select Items](#) and a [Define Conditions](#) tab.



2. On the *Define Conditions* tab, choose an operator and enter a value in the *Filter Condition* box.
3. Click the plus icon to create an alternative condition. Each condition in one category acts as an **OR** operator, so that an object must meet one of the conditions to be included in the search results. If you define one condition in two categories, then each category acts as an **AND** operator, so that both conditions must be true for the object to be returned in the search results.

You can create advanced filter conditions for multiple filter categories. An object must meet one of the conditions in each of the categories to be included in the search results.

For example, if you are viewing terms and want to create a condition to filter on the keywords “investment” and “stock”, you would click **+** (*Add*). Then choose *Contains all of these words*. In the Filter Condition box, enter **investment**. Click **+** (*Add*) again. Choose *Contains all of these words*. In the Filter Condition box, enter **stock**, and then click *OK*. The new condition is listed in the Keyword group in the filter panel, and the filter results are automatically shown.

Change the View for Search Results

You can change the view to see different information.

-  (*Display as Grid*)
-  (*Display as List*)

Related Information

[Evaluating Catalog Assets \[page 25\]](#)

[Evaluating your Data Product \[page 62\]](#)

2.5.1 Evaluating Catalog Assets

When you find the asset you want, you can select it to view its overview information, a preview of its detailed metadata, and a diagram of its impact and lineage. This information includes metadata that is extracted from the source system and data enrichments added in the catalog.

Prerequisites

You must be assigned one of the following:


- The *Catalog User* role.
- A custom role with the *Read* permission for *Catalog Asset*.

Note

To see the details of any terms, tags, or KPIs, the role must also have the [Read](#) permission for each of the following privileges: [Catalog Glossary Object](#), [Catalog Tag Hierarchy](#), and [Catalog KPI Object](#).

Tip

If you have the [Catalog Administrator](#) role, you can enrich the information for an asset. See [Enriching, Classifying, and Publishing](#).

From the  ([Catalog](#)) home page, you can select the [Assets](#) filter to find the asset you want.

When you open an individual asset from the catalog home page, you're taken to a page that provides many different types of information about the asset. This information can include extracted metadata, like the asset name, properties, description, and impact and lineage diagram. It also includes other data enrichments, such as glossary term, tag, and key performance indicator (KPI) relationships applied to the asset, and much more. You can use the information provided to evaluate and make an assessment on whether the asset is the right one you need for your business task at hand.

For example, as a data modeler, you can review the details of a catalog asset to determine whether it is the one you need to add to your model. You can also view the impact and lineage diagram of that asset to see other assets that are consumed or impacted by it.

Viewing the Catalog Asset Header

The asset header provides high-level information about the asset.



Section	Description
Asset Name	Displays the asset name and type with an identifying icon. This name might not match the name of the underlying source object if it was enriched in the catalog.
Asset Functional Status and Source System	Displays the functional status of the asset with the source system name and type. For information on the functional statuses, see Publishing to the Catalog .
Tabs	Select a tab to view more information about the asset, such as an overview of the asset's properties and descriptions, a preview of the asset's details, and a diagram of the asset's lineage and its impact on other objects.

Section	Description
Actions	<p>Select the action you want to perform on the asset.</p> <ul style="list-style-type: none"> • Open: Opens the asset in the source system where you can view or edit it. This button appears if you have permission to access the asset in the source system. • ☆ (Add to Favorites): Adds frequently used assets to your favorites.

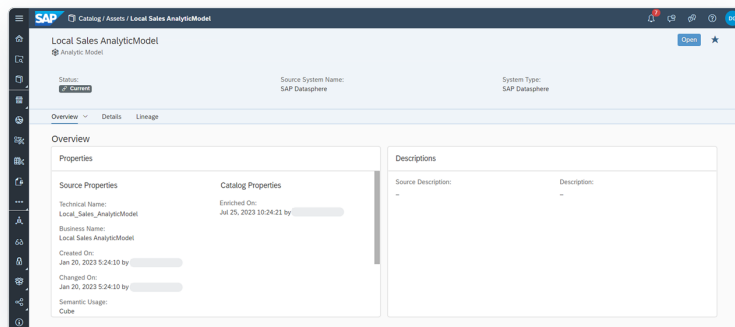
Viewing the Catalog Asset Overview

Use the [Overview](#) tab to view general information about the asset and relationship links to the asset. The [Overview](#) tab is divided into the following sections:

- Overview
- Relationships

Asset Overview

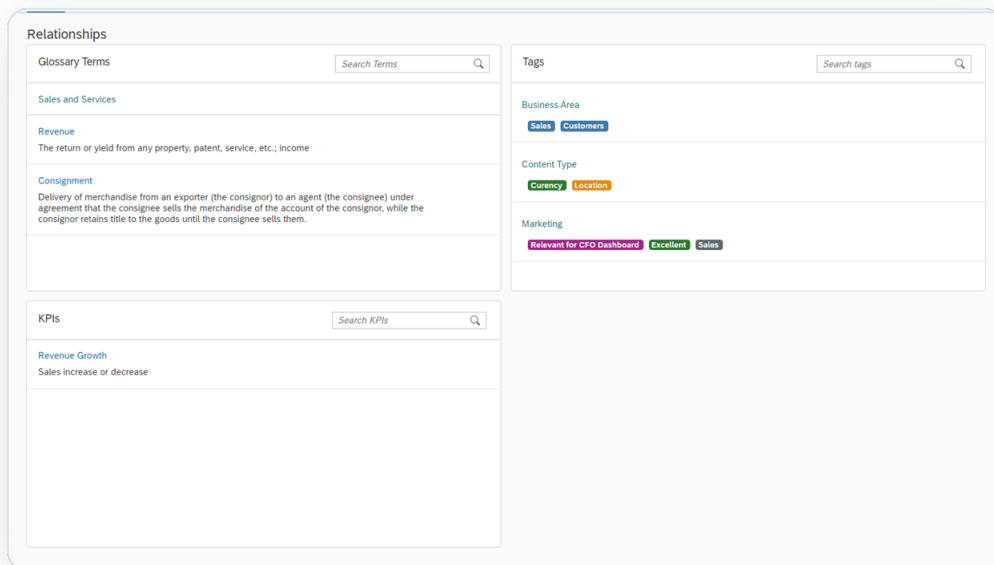
Displays the properties and description of the asset. This information is extracted from a data source or the catalog.



Section Name	Description
Properties	<p>Displays asset properties extracted from the source system and data enrichments added in the catalog. Properties are divided into source properties and catalog properties.</p> <p>Source Properties</p> <p>Source properties common among all assets include:</p> <ul style="list-style-type: none"> • Name: The file name of the asset on the source system. The asset name can also include a <i>Business Name</i> or a <i>Technical Name</i>. For example, assets in SAP Analytics Cloud have one name that appears. Assets in SAP Datasphere have a business name, which is the descriptive name of the asset that helps you identify the asset, and a technical name, which is the name that is used in scripts and code and is synchronized with the business name. • Created On: The date and time on which the asset was created on the source system. The name of the user who created the asset is also shown. • Changed On: The date and time on which the asset was changed on the source system. The name of the user who changed the asset is also shown. <p>Source properties that show information about the source system include:</p> <ul style="list-style-type: none"> • Container Name: Name of the location of the asset. For example, the container name for assets in SAP Datasphere is the space name. The container for assets in SAP Analytics Cloud is the parent folder name. • Container Business Name: Business name of the location of the asset. For example, the container business name for assets in SAP Datasphere is a descriptive name for the space. • Path: Folder location of the asset, if available. For example, this path appears for assets in the SAP Analytics Cloud. • Type: Type of location where the asset is saved. The location type appears for assets in the SAP Datasphere. <p>Source properties common among SAP Datasphere assets include:</p> <ul style="list-style-type: none"> • Semantic Usage: The way the entity should be used. For example, Fact, Dimension, Hierarchy, or Text. • Exposed for Consumption: An indicator that shows whether the asset is made available for consumption in SAP Analytics Cloud and other BI clients. For more information on these properties, see Creating a Graphical View [page 213]. <p>Catalog Properties</p> <p>If the asset data was enriched, the date of the change appears as a catalog property:</p> <ul style="list-style-type: none"> • Enriched On: The date and time on which the data enrichment was added to the asset. The name of the user who enriched the asset is also shown.
Descriptions	<p>Displays the asset description.</p> <ul style="list-style-type: none"> • Source Description: This description is extracted from the source system and can't be edited. • Description: This catalog description is edited by users who have the <i>Catalog Administrator</i> role.

Asset Relationships

Displays the relationships for the asset. These relationships can include glossary terms, tags, and KPIs that are linked to the asset.



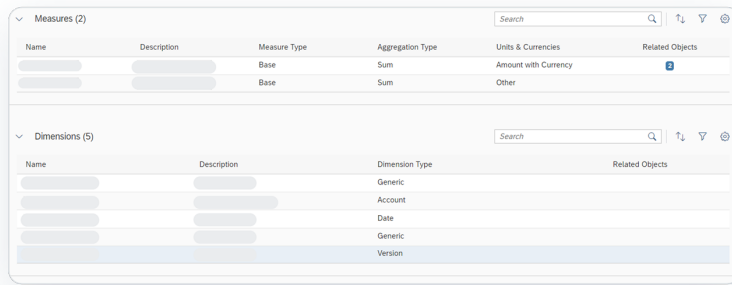
Section Name	Description
Glossary Terms	Displays a list of business glossary terms that are linked to the asset. You can use the free text search to see if a particular term is linked to the asset.
Tags	Displays a hierarchical list of all tags that are linked to the asset. Tags help classify the types of assets that are in the catalog. You can use the free text search to see if a particular tag is linked to the asset.
KPIs	Displays a list of all key performance indicators (KPIs) that are linked to the asset. KPIs are used to track business requirements or goals. You can use the free text search to see if a particular KPI is linked to the asset.

Viewing Detailed Metadata for an Asset

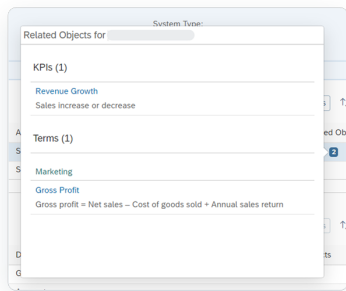
Use the [Details](#) tab to see a preview of the detailed metadata about the asset, which can include columns, attributes, measures, dimensions, and properties specific to each object. This tab appears only if the asset has detailed metadata that can be shown.

Note

If you are viewing an SAP Analytics Cloud asset, this tab is available only for the following model types: planning and analytical models. For a full list of the supported object types, see [Understanding How Automatic Extraction Works](#).



In addition to the preview of the detailed metadata, each metadata table has a column for *Related Objects*. If a row has one or more object (term or KPI) relationships, a button with the total number of object relationships is displayed. You can select the button to see the relationships for the row. To view the term or KPI details, select the link in the dialog.



For each detailed metadata table available, you can use the toolbar to search for and organize the information:

Tool	Description
🔍 (Search)	Use the free-text search to search for a row by its name or description.
↕ (Sort)	Sort the rows in the table by ascending or descending order based on the column you select.
🔽 (Filter)	Select column values for filtering the table.
⚙️ (Select Columns)	Select which columns you want to show in the table.

For information on the metadata that appears in this tab, see the help documentation for the source system and search for the data object or analytic object you want to know more about:

- For SAP Datasphere, see [Acquiring, Preparing, and Modeling Data with SAP Datasphere \[page 8\]](#).
- For SAP Analytics Cloud, see [Welcome to the SAP Analytics Cloud Help](#).

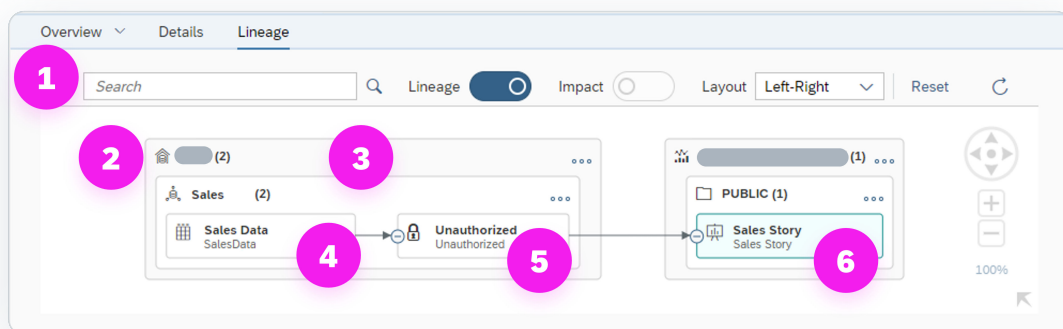
Analyzing Data Impact and Lineage

Use the *Lineage* tab to view the **Impact and Lineage Analysis** diagram. This diagram shows the data analysis of the asset and provides an end-to-end visualization of the asset dependencies across multiple systems and layers. It can help you better understand the lineage (also known as data provenance) and impacts of

a selected asset in the catalog. Impact and lineage contain information about the source of the asset, the transformations it goes through, its final state, and objects affected by changes made to it. Impact and lineage serve distinct purposes.





- **Lineage** is displayed to the left of the object (or below it). It shows objects that the analyzed asset uses as sources. It allows you to trace errors back to the root cause.
- **Impact** is displayed to the right of the object (or above it). It shows objects that use the analyzed asset as a source. It allows you to understand the impact of changes on dependent objects.

In this example, a user views the diagram to analyze the **Sales** story asset. The **Sales** story asset is in the **Public** folder in SAP Analytics Cloud and has two sources that are objects in an SAP Datasphere space.



The diagram provides the data analysis of the asset and contains the following features.

Feature	Description
(1) Toolbar and Diagram Tools	Use the toolbar and diagram tools to control the layout of the diagram. Click Refresh to update the diagram with the latest changes.
(2) Source System	The source system is the outermost object and has an icon that represents the type of system (for example, SAP Datasphere or SAP Analytics Cloud tenant). The number in brackets indicates the total number of objects in the source system that are part of the impact or lineage of the analyzed object. You can expand or collapse a source system, using the Show/Hide All Objects menu on the top-right corner of the symbol.
(3) Container	The container is directly inside the source system and has an icon that represents its type (for example, SAP Datasphere space or SAP Analytics Cloud folder). It contains assets that appear in the lineage of or that impact the analyzed object. The number in brackets indicates the total number of objects in the container that are part of the impact or lineage of the analyzed object. You can expand or collapse a container, using the Show/Hide All Objects menu on the top-right corner of the container.

Feature	Description
(4) Authorized Object	Authorized and unauthorized objects appear in the lineage or impact of the analyzed object.
(5) Unauthorized Object	<ul style="list-style-type: none"> Authorized objects are published and can be discovered in the catalog. You can view the information page for the asset by clicking  (<i>Open Asset Details</i>) icon.
(6) Analyzed Object	<ul style="list-style-type: none"> Unauthorized objects are unpublished and are not available in the catalog. They are shown with the  (<i>Locked</i>). The analyzed object appears as a light blue object. <p>You can show or hide the objects on either side of any object by clicking the  (<i>Show Next Level</i>) or  (<i>Hide All</i>) on the object.</p>

For information on how to control the diagram layout and use tools to further analyze the objects, see [Impact and Lineage Analysis \[page 34\]](#).

Viewing or Editing an Asset

Context


After you find the asset you want, you can open the asset in the source system to view or edit it. When the source file for an asset has been shared with you, the *Open* button appears in the top-right corner. For information about sharing files with other users within a source system, see the documentation for the specific source system:

- For SAP Datasphere, see [Acquiring, Preparing, and Modeling Data with SAP Datasphere \[page 8\]](#).
- For SAP Analytics Cloud, see [Welcome to the SAP Analytics Cloud Help](#).

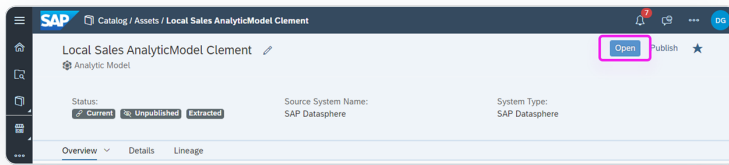
→ Tip

If the source file for the asset has not been shared with you, contact the person who created the asset or the person who most recently changed it. You can find this information in the asset properties.

Procedure

- In the side navigation area, click  (*Catalog*).
- On the *Catalog* home page, use the filters or the search to find the asset you want. For more information, see [Finding and Accessing Data in the Catalog \[page 23\]](#).
- When viewing the page for the asset, in the top-right corner, click the *Open* button to open the source file in the source system in a new browser tab.

The *Open* button appears only if the asset has been shared with you in the source system and you have permission to view or edit it.



4. Depending on how the source file is shared with you, you can view the asset in full and explore it, or you can edit it as needed.

Results

If you edited a file, the catalog automatically detects the change. The metadata for the asset is automatically updated in real time, and the functional status label *Current* is applied.

Using the Asset in a Data Project



Context

After you evaluate and determine that the asset is the right one for your needs, you can use it as part of a data project to build something new. For example:


- As a data modeler in SAP Datasphere, you can use the asset as a source for a Data Builder or Business Builder object. For more information, see [Acquiring Data in the Data Builder \[page 79\]](#) or [Modeling Data in the Data Builder \[page 301\]](#).
- As a content creator in SAP Analytics Cloud, you can use the asset in data object (for example, a story or digital boardroom). For more information, see [Welcome to the SAP Analytics Cloud Help](#).

→ Tip

Before you search for an asset in the catalog, determine which application you want to use the asset in and open it in a new browser tab. By keeping the catalog open in a separate tab, you can find the asset you want and have its property information readily available as you switch between tabs.

Depending on the SAP application you are using, you can open it from the  (*Product Switch*) or from the side navigation. For example, to use SAP Datasphere, in the side navigation area, right-click the application you want to use and click *Open App in New Tab*. Or to use SAP Analytics Cloud, in the side navigation area, right-click any application and click *Open App in New Tab*. In the shell bar click  (*Product Switch*) and click *Analytics*.

Procedure

1. In the side navigation area, click  (*Catalog*).
2. On the *Catalog* home page, use the filters or the search to find the asset you want. For more information, see [Finding and Accessing Data in the Catalog \[page 23\]](#).
3. Go to the tab where the application you want is open.
4. Find and open an existing object or create a new object and add the asset. If you haven't opened the application in a new tab, do that now.
5. Save the object in the location you want.

Note

If the object you are saving is in SAP Datasphere, you must save and then deploy it before it can be added to the catalog. For more information, see [Saving and Deploying Objects \[page 39\]](#).

Results

The catalog automatically detects the change in real time:

- If you created a new file, a new unpublished catalog asset is created and the functional status is set to *Current*. This asset won't be available in the catalog until a user with the *Catalog Administrator* role enriches the metadata and publishes it.
- If you edited an existing file, the metadata for the asset is automatically updated.

Related Information

[Evaluating your Data Product \[page 62\]](#)

2.6 Impact and Lineage Analysis

The *Impact and Lineage Analysis* diagram helps you to understand the lineage (or data provenance) of a selected object, along with its impacts - the objects that depend on it and that will be impacted by any changes that are made to it.




This topic contains the following sections:

- [Open the Diagram \[page 35\]](#)
- [Control the Diagram Layout \[page 35\]](#)
- [Display Object Properties and Analyze or Open Objects \[page 36\]](#)
- [Dependency Analysis Mode \[page 37\]](#)

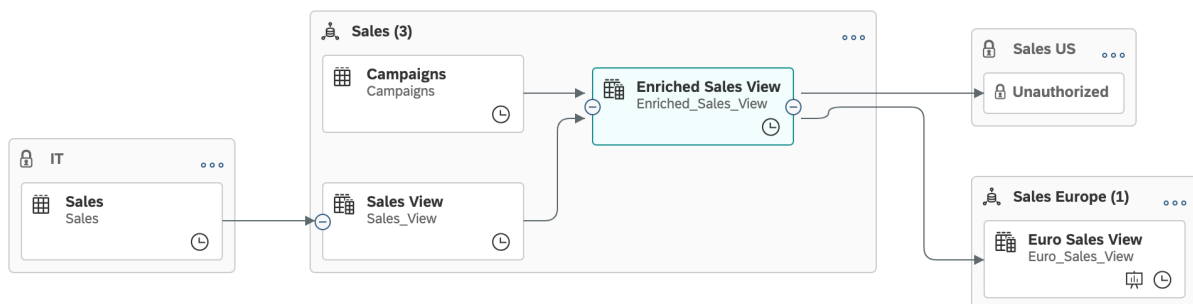
-
- [Shared Objects and Unauthorized Objects \[page 38\]](#)

Open the Diagram

You can open the *Impact and Lineage Analysis* dialog from various screens:

- In the *Repository Explorer* or the *Data Builder* start page, select an object to analyze and click  (*Impact and Lineage Analysis*) in the toolbar.
- In the *Catalog*, click the *Lineage* tab of an asset page.
- In *Data Builder* object editors, click  (*Impact and Lineage Analysis*) in the toolbar or, if the editor contains a diagram, select a table, view, or data flow symbol and click the tool in the symbol toolbar.
- In *Data Access Controls*, click  (*Impact and Lineage Analysis*) in the toolbar.




In this example, a user opens the diagram to analyze the *Enriched Sales View* in the *Sales* space, which has two sources and which is a source for two objects in other spaces:











Control the Diagram Layout

Use the diagram tools to control the layout of the diagram.

Tool	Description
Data Analysis / Dependency Analysis	Toggle between the two modes: <ul style="list-style-type: none"> • <i>Data Analysis</i> - Focus exclusively on data movements and transformations. • <i>Dependency Analysis</i> - Additionally display objects connected through associations and data access controls (see Dependency Analysis Mode [page 37]).
Search	Find and select objects in the diagram. Results are proposed once three characters are entered. Click a result in the list to select the object symbol and highlight other objects on its path to the analyzed object.
Lineage	Enable/disable the display of the lineage of the analyzed object.
Impact	Enable/disable the display of the impacts of the analyzed object.




Tool	Description
Layout	Change the orientation of the diagram: <ul style="list-style-type: none"> • <i>Left-Right</i> - [default] Display lineage objects on the left and impacts on the right of the analyzed object. • <i>Bottom-Top</i> - Display lineage objects below and impacts above the analyzed object.
Reset	Restore the default layout. Changing the mode also resets the layout.
	Scroll, zoom, or recenter the diagram: <ul style="list-style-type: none"> • Click  (or press F6) to zoom in. • Click  (or press F7) to zoom out. • Click the center button (or press F8) to fit to screen, CTRL-click the center button (or press CTRL + F5) to zoom to 100% scale, or enter a percentage. • Click the arrow buttons (or press the arrow keys) to scroll horizontally or vertically.

The diagram can contain various types of symbols.




Symbol	Description
	The object being analyzed. You can show or hide the objects on either side of this or any object by clicking the  (<i>Show Next Level</i>) or  (<i>Hide All</i>) buttons on the symbol.
	Objects appearing in the lineage or impacts of the analyzed object. If you do not have permission to view an object, it is shown with the  (<i>Locked</i>) icon and the name <i>Unauthorized</i> (see Shared Objects and Unauthorized Objects [page 38]).
	Spaces that contain objects appearing in the lineage or impacts of the analyzed objects. The number in brackets indicates the total number of objects in the space that are part of the impact or lineage of the analyzed object. You can expand or collapse a space, using the  (<i>Show/Hide All Objects</i>) menu on the top-right corner of the symbol. If you do not have permission to view a space, it is shown with the  (<i>Locked</i>) icon (see Shared Objects and Unauthorized Objects [page 38]).

Display Object Properties and Analyze or Open Objects

In addition to their business and technical names and their object type (indicated by the icon in the top-left corner, object symbols can display the following properties:

Icon	Description
 (Persisted)/ (Replicated)	Persisted and replicated objects have their data copied locally to enhance performance (see Persist View Data [page 242] and Replicate Remote Table Data [page 98]).
 (Exposed for Consumption)	Exposed objects can be consumed by SAP Analytics Cloud and other BI clients, tools and apps (see Exposing a View For Consumption [page 314]).
 (Deployed)	Deployment Status (see Saving and Deploying Objects [page 39]).

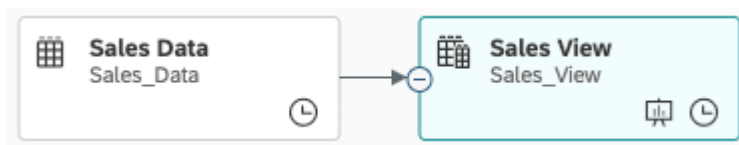
Select an object to display its context menu, which contains tools for furthering the analysis.

Tool	Description
 (Show Details)	Preview the object's properties.
 (Impact and Lineage Analysis)	Change the focus of the diagram to analyze this object. You can go back to the previous diagram by clicking < (Navigate Back) in the dialog header.
 (Open in New Tab)	Open the object in its editor in a new browser tab.

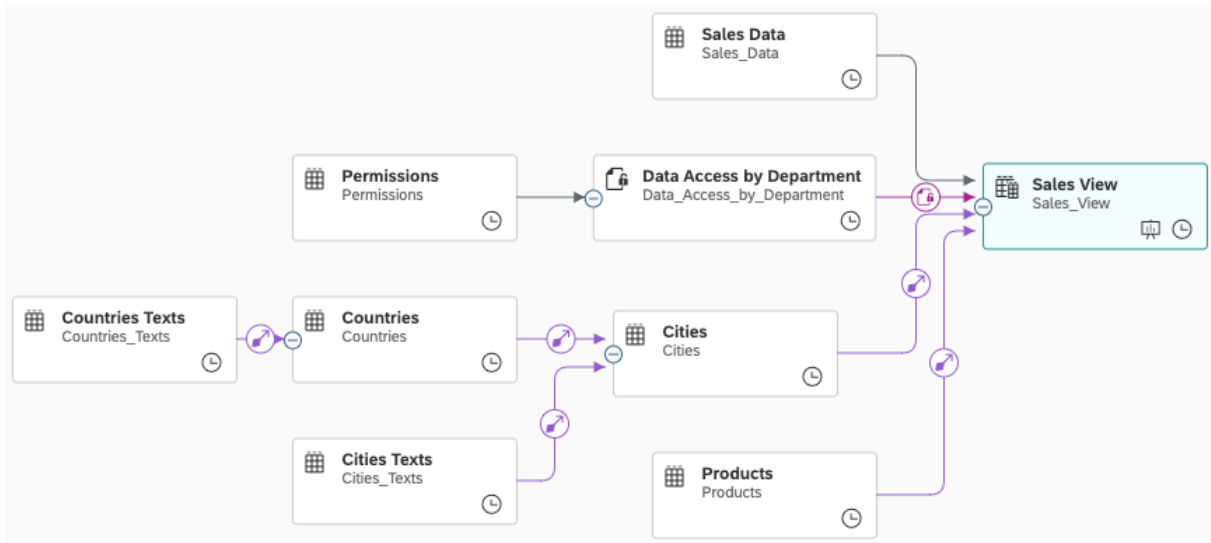
Dependency Analysis Mode

While the default *Data Analysis* mode focuses exclusively on data movements and transformations, the *Dependency Analysis* mode adds objects that are connected to the analyzed object by associations and data access controls.



In this example, the *Sales View* entity shows a single source table in *Data Analysis* mode:




In *Dependency Analysis* mode, it shows several other entities, to which it is linked via associations, along with a data access control:




These additional link types display as follows::

-  (*Association*) - Association pointing to another entity to indicate a potential join between them (see [Create an Association \[page 234\]](#)).
-  (*Data Access Control*) - Data access control providing row-level security (see [Securing Data with Data Access Controls](#)).

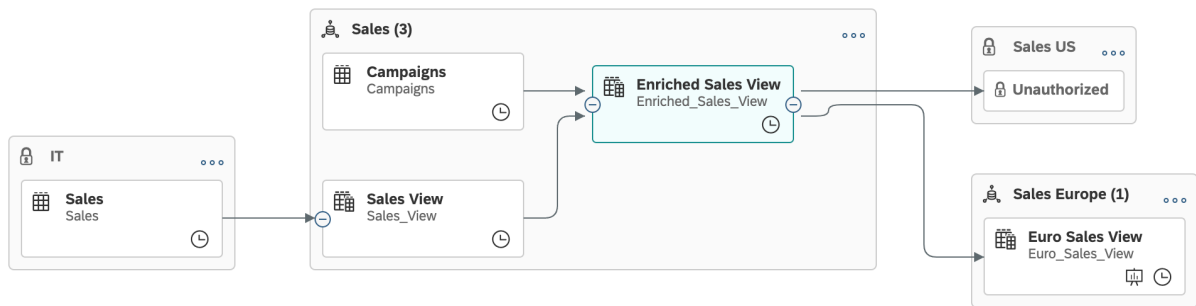
Shared Objects and Unauthorized Objects

You only have permission to see an object if it is created in (or shared to) a space you're assigned to. When exploring the diagram, you may encounter objects that you do not have permission to see. Such objects are shown as *Unauthorized*, and both the object and space show a  (*Locked*) icon:

- **Unauthorized Lineage Objects** - If the analyzed object (or an object appearing in its lineage) is shared from a space you are not assigned to, to a space you are assigned to, this shared object is shown inside its space, which has a  (*Locked*) icon.
You can show the immediate sources of this object but, if you do not have permission to see one or more of them, then they will be represented by *Unauthorized* placeholders. You cannot view the properties of these placeholder objects, perform any actions on them, or continue the analysis beyond them in any way.
- **Unauthorized Impact Objects** - The analyzed object (or an object appearing in its impacts) may be shared to a space you're assigned to and used as a source by objects in that space.
You can show the objects that are immediately impacted by this object in their space, which has a locked icon but, if you do not have permission to see one or more of them, then they will be represented by *Unauthorized* placeholders. You cannot view the properties of these placeholder objects, perform any actions on them, or continue the analysis beyond them in any way.

In our example:

- The user can see the *Sales* table that is shared from the *IT* space to the *Sales* space, but does not have permission to see the sources of that table (if any) or any other objects in the *IT* space.
- The user does not have permission to see any objects in the *Sales US* space, to which the *Enriched Sales View* is shared.



2.7 Saving and Deploying Objects

When you save an object, it is stored in the SAP Datasphere repository, which contains the design-time definitions of all your objects. When you deploy an object, you are creating a run-time version for use in the SAP Datasphere database.

This topic contains the following sections:

- [Create a Copy of an Object \[page 39\]](#)
- [Preview Data \[page 39\]](#)
- [Deploy Objects \[page 40\]](#)
- [Save Anyway and Deploy Anyway \[page 40\]](#)
- [Run-Time Database Unavailable \[page 41\]](#)

Create a Copy of an Object

You can create a copy of many types of objects by:

- Selecting the object in the Repository Explorer and clicking *Duplicate*.
- Opening the object in its editor and clicking **Save > Save As**

Preview Data




Design-time objects do not contain any data, but you can preview the data they will contain when they are deployed to the run-time environment (see [Viewing or Previewing Data in Data Builder Objects \[page 297\]](#)). You can save an object even if it contains validation errors.

Deploy Objects

Note

All Data Builder objects and data access controls can be saved and deployed. In the Business Builder, only perspectives are saved and deployed. Other Business Builder objects do not need to be deployed.

The current status of an object is shown in its *Status* property, which can have the following values:

-  (*Not Deployed*) - The object has never been deployed and exists only as a design-time artifact.
-  (*Deployed*) - The object is deployed to the run-time database and its design-time and run-time versions are identical.
-  (*Changes to Deploy*) - The design-time version of the object contains changes that need to be deployed to make them available in the run-time.
- (Design-Time Error) - The design-time version of the object is invalid. You can save an object with design-time errors, but you cannot deploy it. To correct design-time errors, review each of the validation messages in the editor, make the necessary changes, and then save the object.
- (Run-Time Error) - The run-time version of the object is invalid and can no longer be used. Run-time errors may be caused by changes in one or more of the object's sources (see [Modifying Objects That Have Dependent Objects \[page 41\]](#)). To correct the errors, review each of the validation messages in the editor, make the necessary changes, and then save and deploy the object.

When you deploy a table or view, you create a run-time version, which can be used by other run-time objects.

Note

If your space is locked because it has reached its storage capacity, then you will not be allowed to deploy objects. Contact your DW Space Administrator.

The following types of object are automatically deployed upon creation and immediately available in the run-time:

- Remote tables imported into the repository from a source system, database user schema or HDI container (see [Importing Tables and Views from Sources \[page 90\]](#)).
- Tables created by importing data from a * .csv file (see [Creating a Local Table from a CSV File \[page 122\]](#)).

Save Anyway and Deploy Anyway

If other objects use your object as a source or otherwise depend on your object, then any changes you make to your object may impact (change or break) these other objects. If changes you have made to an object will impact its dependent objects, then validation messages are generated to warn you of these impacts.

When you try to save or deploy your object, the *Validation Messages* dialog opens to allow you to review these messages. You can still save or deploy your object (by clicking *Save Anyway* or *Deploy Anyway*), but you should aim to review and resolve these impacts as soon as possible. For more information, see [Modifying Objects That Have Dependent Objects \[page 41\]](#).

Run-Time Database Unavailable

The run-time database server might be unavailable under exceptional circumstances. In this case, a message strip is displayed informing you of the database status and the following features are disabled:

- [Deploy](#)
- [Share](#)
- [Delete](#)
- [Source Browser](#)
- [Preview Data](#)
- [Import CSV File](#)
- [Validate SQL and Preview Data](#)
- [Run a flow](#)

You can try to solve this issue by waiting a few minutes and refreshing your web browser. If the problem persists, [open a support ticket](#) 🙋.

2.8 Modifying Objects That Have Dependent Objects

Once a table or view is created, it can be used as a source, or pointed to via an association by other objects. When a data access control is created it can be attached to one or more views to provide row level security. When you modify objects that have dependent objects, you may receive validation messages to warn you that your changes can impact these dependent objects.

Procedure

1. Open your table, view, or data access control to edit it in the usual way.

You can see the objects that depend on your object by reviewing its *Dependent Objects* section (see [Review the Objects That Depend on Your Table or View \[page 43\]](#)).

2. Make any appropriate changes. Certain changes may produce warning messages:

Change	Validation Message
Add Column	None. New columns are excluded by default from the output of dependent views and data flows and information messages are displayed.

Change	Validation Message
Change Column Data Type	<p>Warning listing all dependent objects that use the column.</p> <div style="border: 1px solid #ccc; padding: 10px; margin: 10px 0;"> <p>Note</p> <p>You cannot change the data type of any columns that are used by a data access control (see Process Source Changes in the Data Access Control Editor).</p> </div> <p>In the case of tables:</p> <ul style="list-style-type: none"> You can always increase the length of a data type, but you can reduce it only if the change will not truncate any data currently held in the column. For example, if your column's current data type is <code>string(256)</code> and the longest string contained in it is 200 characters, then you cannot reduce the length to less than <code>string(200)</code>. You can change the data type if the new type is compatible with the family of the old type and the data can be converted. For more information about data type conversions, see Column Data Types [page 110]. <div style="border: 1px solid #ccc; padding: 10px; margin: 10px 0;"> <p>Note</p> <p>You cannot change the data types of columns containing <code>ST_POINT</code> or <code>ST_GEOMETRY</code> data.</p> </div>
Delete Column	<p>Warning listing all dependent objects that use the column.</p> <p>Column deletions will break dependent objects that use these columns. The last user to edit a dependent object will receive a notification inviting them to review the changes.</p>
Add, Delete, or Change Data Type of Input Parameter	<p>Warning listing all dependent objects.</p> <p>All changes to input parameters must be processed in all dependent objects.</p> <div style="border: 1px solid #ccc; padding: 10px; margin: 10px 0;"> <p>Note</p> <p>You cannot add input parameters to views that are used as sources in data flows.</p> </div>
Change Business Properties	<p>None.</p> <p>Changes to the business name and other business properties of the object or its columns are propagated to dependent objects.</p>

- Each time that you make a change, SAP Datasphere checks all dependent objects, generates a warning message if necessary, and updates the status of each object in the [Dependent Objects](#) list.

For example there are ten views that use your view as a source, and which are listed in your view's [Dependent Objects](#) list. When you delete a column in your view, SAP Datasphere, checks to see which of the ten views use that column and, if one or more do, it generates a warning message and updates the status of each impacted object to [Design-Time Error](#).

- When you have finished making changes, click [Save](#) to save your object.

If any warnings have been generated, the [Validation Messages](#) dialog opens to allow you to review them. Click [Save Anyway](#) to save your changes and dismiss the warning messages.

Notifications are sent to the last user who edited each of the dependent objects that now has a status of *Design-Time Error*, inviting them to review the changes. For more information, see:

- [Process Source Changes in the Table Editor \[page 100\]](#)
- [Process Source Changes in the Graphical View Editor \[page 246\]](#)
- [Process Source Changes in the SQL View Editor \[page 264\]](#)
- [Process Source/Target Changes in the Data Flow Editor \[page 154\]](#)
- [Process Source Changes in the Data Access Control Editor](#)

5. Click  (*Deploy*) to deploy your object.

If any warnings have been generated, the *Validation Messages* dialog opens to allow you to review them. Click *Deploy Anyway* to deploy your changes and dismiss the warning messages.

Note

You should consider the impact of your object on its dependent objects before deploying it. If many objects or particularly important objects will be impacted, you should coordinate with the appropriate other users to deploy the source and its dependent objects together. You could create an ER model (see [Creating an Entity-Relationship Model \[page 286\]](#)) to contain the source and its dependent objects and use the ER model *Deploy* button to deploy them simultaneously once all the changes are processed.

2.8.1 Review the Objects That Depend on Your Table or View

Review the objects that use your entity as a source or via an association. You can click any of the listed objects to navigate to it.

The *Dependent Objects* list is available in your entity property sheet, and includes the following information:

Property	Description
Business / Technical Name	Identify the object that depends on the present entity.
Status	Displays the status of the dependent object.
Dependency Type	Displays the type of the dependency, which can be: <ul style="list-style-type: none"> • <i>Source</i> - The dependent object uses the present object as a source. • <i>Association</i> - The dependent object points to the present object with an association.
Sub-Dependencies	Displays the number of objects that are, in turn, dependent on the dependent object, and how many of these objects have errors.

Note

Including the *Sub-Dependencies* column, you can see two levels of dependencies in the *Dependent Objects* list. To see the full graph of dependencies, click the *Impact and Lineage Analysis* button in the toolbar and review the *Impacts* side of the graph (see [Impact and Lineage Analysis \[page 34\]](#)).

Property	Description
Deployed On / Changed By / Changed On	[table list only] Display information about the last user to change the object and when it was last changed and deployed.

2.9 Sharing Tables and Views To Other Spaces

Share a *Data Builder* table or view to another space to allow that users assigned to the space use it as a source for their objects.

This topic contains the following sections:

- [Preparing to Share \[page 44\]](#)
- [Share Entities \[page 46\]](#)
- [Share Entities Containing Associations \[page 47\]](#)
- [Review Entities Shared With Your Space \[page 48\]](#)
- [Unshare an Entity \[page 48\]](#)

Preparing to Share

Your space is a secure area and its entities and other objects cannot be seen in other spaces unless you choose to share them. When you share an entity to another space, users in that space can use it as a source for their views and other objects.

Note

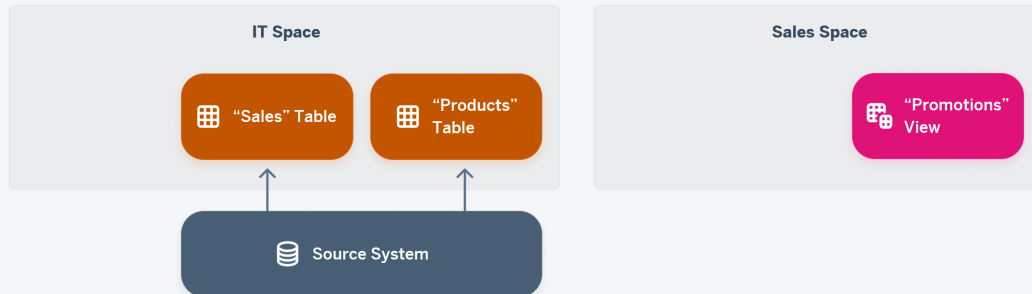
Only *Data Builder* tables and views can be shared. Other *Data Builder* objects and *Business Builder* objects cannot be shared to other spaces.

You can share tables and views, but only views can have row-level security applied to protect data (see [Securing Data with Data Access Controls](#)).

In this example, the *IT* space has imported the *Sales* and *Products* tables from a source system and wants to share them to the *Sales* space, but the *Sales* table contains sensitive data:

Step 1: Identify Objects to Share

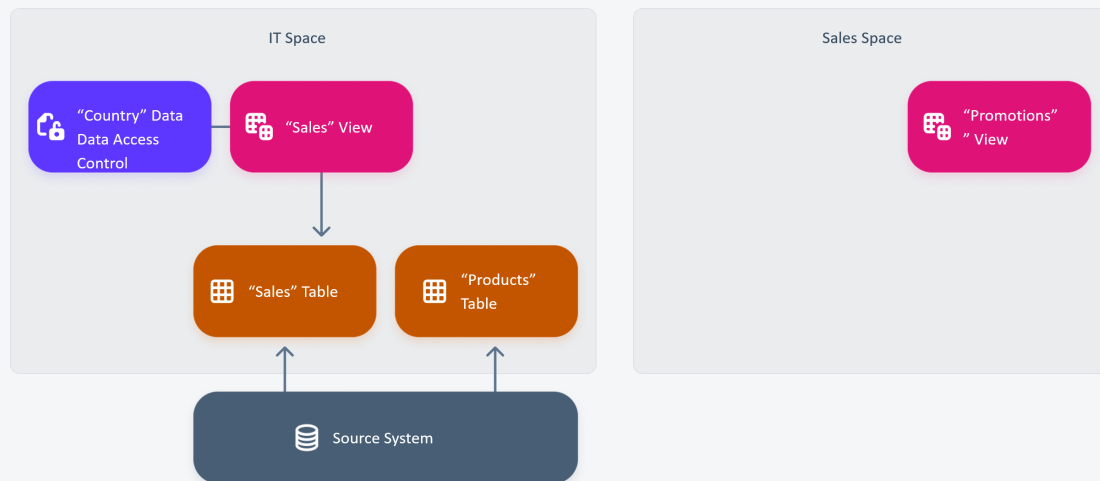
You can share tables and views to other spaces.



In order to protect that data and ensure that users can view only data for which they have permission, an *IT* space user consumes the *Sales* table in the *Sales* view and applies a data access control to the view (see [Apply a Data Access Control \[page 245\]](#)):

Step Two: Secure Data Before Sharing

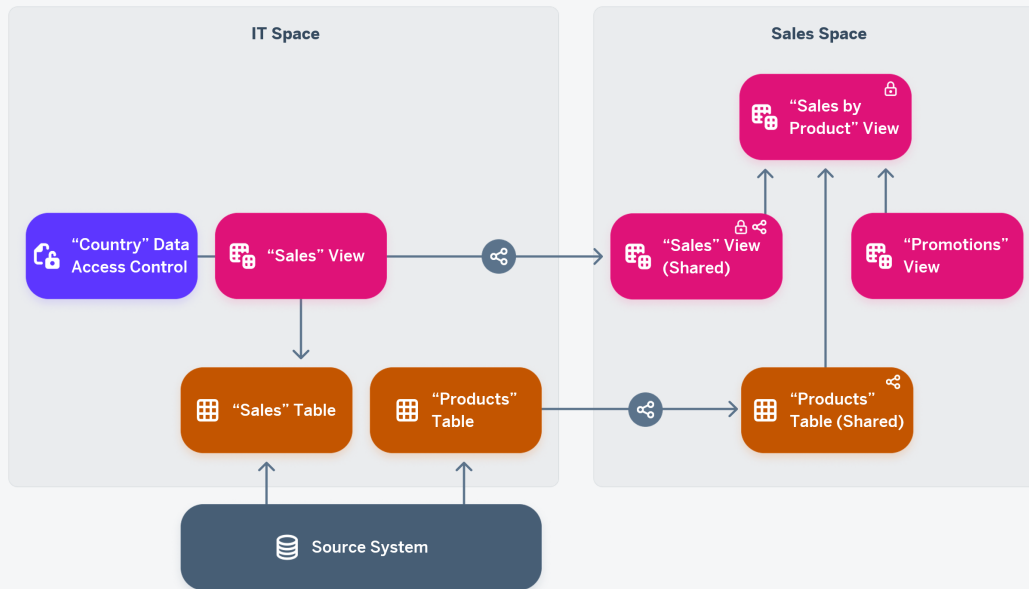
Apply a data access control to any object containing sensitive data before you share it.



Now that the data is protected, the *Sales* view and the *Products* table can both be shared to the *Sales* space, where they are joined with the *Promotions* view to produce the *Sales by Product* view, which inherits the row-level security from the shared *Sales* view:

Step 3: Share Data

Shared data can be used as a source for objects in the spaces it is shared to.





Share Entities

1. Select the entity or entities you want to share in the *Data Builder* or *Repository Explorer*. Alternatively, open an entity in its editor.

Note

An entity must be deployed before it can be shared.

2. Click  (*Share*) to open the *Share* dialog.
3. In the *Add Spaces* field, enter the name of the space or spaces that you want to share the entities to (or click  to select the spaces in a list and then click *Select*).


Note

You can share entities to any spaces in your SAP Datasphere tenant, including those you are not assigned to.

4. Click *Share* to grant the other spaces *Read* access to the objects. If you have selected a single space, the *Shared with* list is updated to include these spaces.

Note

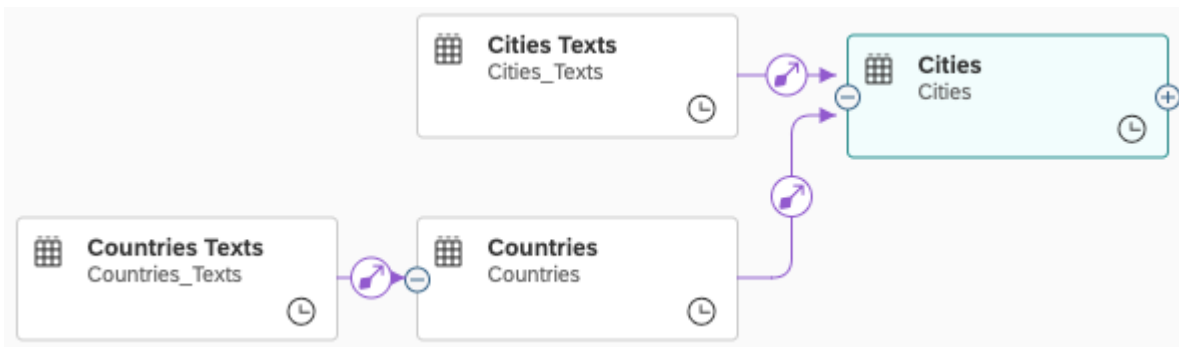
The *Shared with* list is not shown if two or more entities are selected for sharing.

- Click *Close* to close the *Share* dialog.
Entities that are shared to other spaces display a  (*Share*) icon after their business name in the *Data Builder* start page and *Repository Explorer*. You can click this icon to open the *Share* dialog, review the spaces the object is shared to, and, if appropriate, stop sharing it.

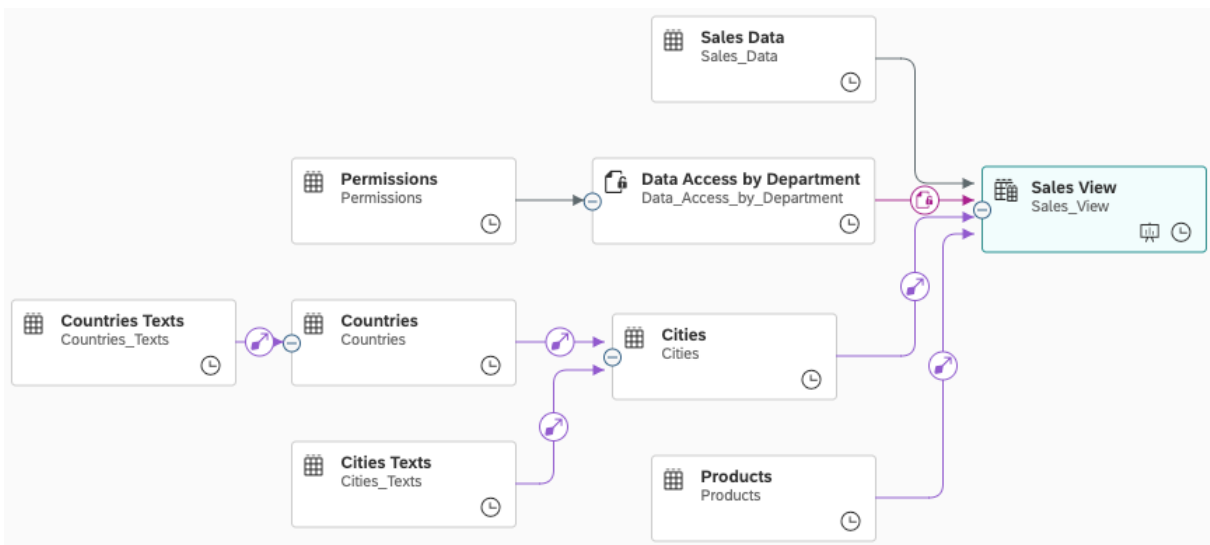
Share Entities Containing Associations

Entities with a semantic usage of *Fact* or *Dimension* commonly have associations to other entities and when sharing them, you will often want to share the associated entities too.

In this example, the *Cities* entity has a semantic usage of *Dimension* and has associations to the *Cities Texts* text entity and to the *Countries* dimension (which, itself, has an association to the *Countries Texts* text entity). When sharing *Cities*, you would generally share the other three entities as well:



In this example, the *Sales View* entity has a semantic usage of *Fact* and has associations to the *Cities* and *Products* dimensions. These dimensions then have further associations to other dimensions and text entities. When sharing *Sales View*, you would generally share the other five entities as well:



Note

You should only share additional entities (dimensions, text entities, and hierarchies) that are connected by associations. In this example, you would not share the source table, *Sales Data*, nor the data access control, *Data Access by Department*.

Review Entities Shared With Your Space

You can see the entities shared with your space:

- In the *Repository Explorer* (see [Repository Explorer \[page 12\]](#)).
- In the *Shared Objects* folder on the *Repository* tab of the *Source Browser* in any *Data Builder* editor (see [Using the Source Browser \[page 292\]](#)).

Note

Entities shared to your space are not listed in the *Data Builder* start page.

Unshare an Entity

To stop sharing an entity, expand the *Shared with* list, select the spaces you want to stop sharing it with, and click *Unshare*.

2.10 Importing and Exporting Objects in CSN/JSON Files

You can use the tools in certain *Data Builder* editors to import objects to and export objects from your space.

The following object types can be exported and imported via CSN files:

Object Type	Details
Local Tables	The definition of a local table contains the structure of the table only, and does not have dependencies on any other objects.
Remote Tables	The definition of a remote table contains information about its connection.


Note

Remote tables exported from one space can be imported into another only if they were originally imported from a connection created in v2021.19 or later.

Object Type	Details
Views	The definition of a view contains the definitions of all its sources and any used data access controls. When you export a view, these objects are exported too.
Flows	The definition of a data flow, replication flow, or transformation flow contains the definitions of all its sources and its target table. When you export a flow, these objects are exported too.
Intelligent Lookups	The definition of an intelligent lookup contains the definitions of its input and lookup entities. When you export an intelligent lookup, these entities are exported too.
Analytic Models	The definition of an analytic model contains the definitions of its fact and dimension sources. When you export an analytic model, these entities are exported too.
E/R Models	The definitions of all the tables and views in the model are exported. The model itself is not exported.

Note

You can also export content from and import content to your space via:

- The  (*Transport*) app (see [Transporting Content Between Tenants](#)).
- The `datasphere` command line interface `objects` commands (see [Manage Modeling Objects via the Command Line](#)).




2.10.1 Importing Objects from a CSN/JSON File

Import objects from a CSN/JSON file into your space from the *Data Builder* start page or an entity-relationship model.

Context

A CSN (Core Data Services Schema Notation) file contains only the definition of the structure of your tables, views, and other objects, and does not contain any data. A CSN file can be imported into another space or another instance of SAP Datasphere. For detailed information about the CSN file format, see [Core Data Services Schema Notation \(CSN\)](#).

Procedure

1. Navigate to the *Data Builder* start page or to an entity-relationship model (see [Creating an Entity-Relationship Model \[page 286\]](#)) and click  (*Import*)  *Import Objects from CSN/JSON File* .
2. Select the CSN/JSON file you want to import, and click *Next* to open the *Select Objects to Import* dialog, which lists the objects contained in the CSN/JSON file, their type and semantic usage, as well as status:
 - *Ready to Import* - No object with this technical name is present in the space.

- *Already in the Repository* - The object is already present in the repository. If you select it for import, it will overwrite the existing object
3. Select the objects you want to import and click *Import*.

Note

Remote tables exported from one space can be imported into another only if they were originally imported from a connection created in v2021.19 or later.

In addition to the objects you select, other objects may be imported:

- Any objects on which your objects depend. If any of these objects are already present in the repository, you will see a message asking if you wish to re-import them and overwrite the repository version. Click:
 - *Yes* - To re-import these objects and overwrite the repository versions. Some data may be lost if the structure of an object has changed.
 - *No* - To keep the repository versions.
 - *Cancel* - To cancel the import.
- Any simple types contained in the file. These types will be available for selection as a *Data Type* for columns. You cannot delete or modify simple types within SAP Datasphere.

The objects are imported to the space. If you are in an entity-relationship model, the imported objects are added to the diagram.

Note

Objects imported from a CSN/JSON file are not automatically deployed.

2.10.2 Exporting Objects to a CSN/JSON File

Export the definitions of your tables, views, and other objects to a CSN/JSON file, which can be imported into another space.

Context

A CSN (Core Data Services Schema Notation) file contains only the definition of the structure of your tables, views, and other objects, and does not contain any data. A CSN file can be imported into another space or another instance of SAP Datasphere. For detailed information about the CSN file format, see [Core Data Services Schema Notation \(CSN\)](#).

The *Export to CSN/JSON File* button is available in the following editors:

- Table editor
- Graphical view editor
- SQL view editor
- Data flow editor
- Intelligent lookup editor

- Analytic model editor
- Entity-relationship model editor

Procedure

1. In the editor toolbar, click  (*Export*)  *Export to CSN/JSON File* .
2. If the editor contains unsaved changes, you are prompted to save. Click *OK*.

The contents of the editor are exported to `<Object_Name>.json` and downloaded to your browser:

Object Type	Details
Local Tables	The definition of a local table contains the structure of the table only, and does not have dependencies on any other objects.
Remote Tables	The definition of a remote table contains information about its connection.
<div style="background-color: #f0f0f0; padding: 10px; border: 1px solid #ccc;"> <p>Note</p> <p>Remote tables exported from one space can be imported into another only if they were originally imported from a connection created in v2021.19 or later.</p> </div>	
Views	The definition of a view contains the definitions of all its sources and any used data access controls. When you export a view, these objects are exported too.
Flows	The definition of a data flow, replication flow, or transformation flow contains the definitions of all its sources and its target table. When you export a flow, these objects are exported too.
Intelligent Lookups	The definition of an intelligent lookup contains the definitions of its input and lookup entities. When you export an intelligent lookup, these entities are exported too.
Analytic Models	The definition of an analytic model contains the definitions of its fact and dimension sources. When you export an analytic model, these entities are exported too.
E/R Models	The definitions of all the tables and views in the model are exported. The model itself is not exported.

2.11 Importing and Exporting Objects via the Command Line

You can use the SAP Datasphere command line interface, `datasphere`, to import objects to and export objects from your space.

To use `datasphere` to export and import objects you must have an SAP Datasphere user with the *DW Modeler* role or equivalent permissions (see [Roles and Privileges by App and Feature](#)).

For more information, see [Manage Spaces via the Command Line](#).

2.12 Enabling Currency Conversion with TCUR* Tables and Views


Create currency conversion views and the necessary supporting objects based on an SAP connection, tables shared from another space, or manually.

To enable currency conversion, you create the necessary objects in your space.

- You can create currency conversion local tables and views and then manually add and maintain currency conversion data.
- You can import currency conversion objects and data from these SAP connections: SAP S/4 HANA Cloud, SAP S/4 HANA On-Premise and SAP ABAP.
- You can use the currency conversion tables shared from another space.

Once the objects are created, you will be able to convert currency values into another currency using a currency conversion column in a graphical view or an SQL view. See [Create a Currency Conversion Column \[page 227\]](#).

Create Currency Conversion Objects

1. In the side navigation area, click  (*Data Builder*), select a space if necessary, and click +, then *Currency Conversion Views*. The *Currency Conversion Views* dialog opens.

Note

You can also access the *Currency Conversion Views* dialog from the *Repository Explorer*.

2. In the *Source* dropdown list, the local connections (of type SAP S/4 HANA Cloud, SAP S/4 HANA On-Premise, SAP ABAP) that are in your space are displayed. Also, if the TCUR* tables of another space have been shared to your space, the name of the space is displayed in the list. Select one of the following:
 - *Manual* - Local tables and views will be created and you will then need to enter or import the data.
 - An SAP S/4 HANA Cloud or SAP S/4 HANA On-Premise connection - Local tables, views and data flows will be imported.
 - An SAP ABAP connection - Local tables, views, data flows and remote tables will be imported.
 - A space name - Views will be created. Data is read from the shared local currency conversion tables that views are based on.

You can see that the objects listed below will be generated for each object type: local tables (unless the tables are shared from another space), views, and, depending on the connection you've selected, data flows and remote tables:

- TCURV - Exchange rate types
- TCURW - Exchange rate type text
- TCURX - Decimal places in currencies
- TCURN - Quotations
- TCURR - Exchange rates
- TCURF - Conversion factors

- TCURC - Currency codes
 - TCURT - Currency text
3. If needed, rename the business names of each object.
 4. Click [Create](#).
The creation process runs in the background and you can keep working in SAP Datasphere. You're notified when the creation of the objects has been completed.
 5. Once the creation of the objects is finished, do one of the following:
 - If you've chosen [Manual](#), local tables and views have been created and you now need to enter currency conversion data manually or for example import the data from a CSV file in a local table.
 - If you've chosen a connection as a source, local tables, views, data flows and - for an SAP ABAP connection - remote tables have been created. You now need to run the data flows to fill the tables with data. See [Creating a Data Flow \[page 128\]](#).
 - If you've chosen a space as a source, views have been created and data is read from the tables.


Next steps: Now that the necessary objects are created, you can convert currency values into another currency using a currency conversion column in a graphical view. See [Create a Currency Conversion Column \[page 227\]](#).

Share Currency Conversion Tables to Other Spaces

Once you've created the currency conversion objects in your space, you can share the TCUR* tables to other spaces. The spaces to which you share these tables will then have your space as an alternative source.

Note

As a general rule, you can share objects (including TCUR* tables) to another space by following another procedure. For more information, see [Sharing Tables and Views To Other Spaces \[page 44\]](#)

1. In the side navigation area, click  ([Data Builder](#)), select a space if necessary, and click +, then [Currency Conversion Views](#). The [Currency Conversion Views](#) dialog opens.
2. Click [Share](#).

Note

If the currency conversion objects have not been created for your space, the [Share](#) button is disabled.

3. In the dialog box that opens, enter or select the spaces to which you want to share the TCUR* currency conversion tables and click [Share](#).

Next steps: Now that the TCUR* tables are shared to other spaces, these spaces will see your space as an alternative source in the [Source](#) field.

2.13 Deleting Objects

You can delete objects from SAP Datasphere only if other objects do not depend on them.

For example, if **View 1** consumes **Table 1** and **Table 2** as sources, then you cannot delete either table unless or until you delete **View 1** or modify it so that it consumes other sources instead.

If the object is used by one or more other objects then a dialog listing these dependencies opens, and the deletion is canceled.

Note

If you want to delete a remote table with a data access of *Replicated (Real-Time)*, you must ensure that:

- The data provisioning agent is connected.
- The real-time replication is not paused and is working normally.

If either of these requirements is not met, you must remove the replicated data before you can delete the remote table.

3 Semantic Onboarding

The *Semantic Onboarding* app provides a central entry point to import semantically-rich objects from your SAP systems and the Content Network, as well as the Public Data Marketplace and other marketplace contexts.

This topic contains the following sections:

- [Import Entities from SAP Systems \[page 55\]](#)
- [Import Business Content from the Content Network \[page 55\]](#)
- [Acquire Data Products from the Public Data Marketplace and Other Contexts \[page 56\]](#)

Import Entities from SAP Systems

Users with the *DW Modeler* role (or equivalent privileges) can use the *Import Entities* wizard to import semantically-rich objects from your SAP S/4HANA Cloud, SAP S/4HANA on-premise, SAP BW/4HANA, and SAP BW bridge connections. The wizard creates *Business Builder* and *Data Builder* entities (along with all the objects on which they depend) in SAP Datasphere.

Click a tile to open the *Import Entities* wizard for that connection type (see [Importing Objects with Semantics from SAP S/4HANA, SAP BW/4HANA and SAP BW Bridge \[page 83\]](#)).

Each tile shows the total number of connections of that type available to you across all the spaces you have access to. If you do not have access to a particular type of connection, you must create it or ask a colleague space administrator or integrator to do so:

- *SAP S/4HANA Cloud* - See [SAP S/4HANA Cloud Connections](#).
- *SAP S/4HANA On-Premise* - See [SAP S/4HANA On-Premise Connections](#).
- *SAP BW/4HANA* - See [SAP BW/4HANA Model Transfer Connections](#).
- *SAP BW Bridge* - See [SAP BW Bridge Connection](#).

Click *Open Import Logs* and select a space to view the results of imports in your space.

The left pane contains a list of imports with the start time, duration, and result. Select an import in the left pane to show its log messages.

Import Business Content from the Content Network

Users with the *DW Space Administrator* role (or equivalent privileges) can import business content and sample content from SAP and partners from the *Content Network*.

- *Business Content*: End-to-end business scenarios created by SAP for various industries and lines of business.
- *3rd Party Content*: End-to-end business scenarios created by SAP partners. Some third-party content has to be purchased in the [SAP Store](#).

- [Samples](#): Sample content to get started with SAP Datasphere.

For more information, see [Importing SAP and Partner Business Content from the Content Network](#).

Acquire Data Products from the Public Data Marketplace and Other Contexts

Users with the [DW Modeler](#) role (or equivalent privileges) can acquire data products from data providers and import them into their space to combine with internal data.

Click a tile to open the context in the data marketplace (see [Searching and Browsing \[page 58\]](#)).

4 Purchasing Data from Data Marketplace

Enrich your own data with third-party data products.

Note

Data Marketplace and the Data Sharing Cockpit are only available in SAP Datasphere tenants provisioned since January 25, 2021. See [3148585](#) for more information.

Overview

Data Marketplace is fully integrated into SAP Datasphere. It's tailored for businesses to easily integrate third-party data. You can search and purchase analytical data from data providers. The data comes in form of objects packaged as data products that can be used in one or several spaces of your SAP Datasphere tenant.

Data products are either provided for free or require the purchase of a license at a certain cost. Some data products are available as one-time shipments, other data products are regularly updated by data providers.

To get data products into your SAP Datasphere tenant and consume them, you can follow a simple workflow:

3 steps to enrich your data with third-party data



- [Purchasing Data from Data Marketplace \[page 57\]](#)
- [Evaluating your Data Product \[page 62\]](#)
- [Testing with Sample Data \[page 64\]](#)
- [Acquiring a Data Product \[page 67\]](#)
- [Managing your Data Products \[page 71\]](#)

- <https://help.sap.com/viewer/c8a54ee704e94e15926551293243fd1d/cloud/en-US/b4a5d02cefdf45478e7376860c985202.html> [<https://help.sap.com/viewer/c8a54ee704e94e15926551293243fd1d/cloud/en-US/b4a5d02cefdf45478e7376860c985202.html>]

Independent from time, you can also view and manage your licenses. For more information, see [Managing your Licenses \[page 74\]](#).

ⓘ Note


Please note that cross-landscape visibility and installation of data products is not supported. That means products created in landscape A can only be seen and acquired in landscape A. They are not visible in other landscapes, e.g. in landscape B or landscape C.

4.1 Searching and Browsing

Search or browse for products using the search function or landing page.

You find the [Search](#) function either on the [Landing Page](#) or within the Data Marketplace menu. Both options allow you to browse through the search result list by different filter criteria.

You can search and browse for data products and data providers. Either select [Data Products](#) or [Providers](#) from the dropdown list next to the search input field. After you entered the search term press on your

keyboard or click on the icon .

The search results will be displayed in a list and are sorted by best match by default. You can change the sorting criteria if you like.

You can use filter criteria to limit the search results. On the left-hand side next to the search result list you find the filter criteria. Click on [Show All Filters](#) at the end of the filter list shown to get the complete list of all filters and options. On the popup you can also define filter conditions for each filter.

To see only free data products, select the option [Free](#) from the filter [Contract Type](#). All data products which have a different contract type, for example License Key, are set to hidden in the search result list. The list will get smaller.

For more information on the available filters, see [Overview of Filters \[page 59\]](#).

By clicking an entry in the search result list, you will be directly forwarded to the corresponding data product page or data provider profile. There, you can evaluate the data product or data provider and eventually bookmark them. For more information, see [Using Bookmarks \[page 62\]](#).

4.1.1 Overview of Filters

To limit the search results, you can combine different filters with different options.

Filter by Contract Type

Filter by Contract Type

Free

Data products that are free of charge are provided without any payment. This could be public domain data or data provided by a commercial data provider as give-away data, for example. The data can be activated after accepting the data specific terms of use.

With License Key

An activation key provides authorization to access one or multiple data products. The data provider sends the key via an email after a price has been determined between the consumer and provider. The price is negotiated outside of data marketplace.

License keys can be used:

- for commercial products and are provided by the data provider after closure of the commercial agreement.
- for any other type of product that needs access authorization (for example for scenarios where non-public data is exchanged without payment).

On Request

To activate a data product, you need to send a request to the data provider. You can do this directly through the user interface where an email template is created. You can edit the predefined email and then send it to the provider who then gets in touch with you.

Filter by Data Shipment

Filter by Data Shipment

Direct

The data is copied directly into the space you've selected. After activating the product, a protocol is created in *My Data Products* → *Delivery Tracking*. Once the data is available, you can use the data as any other data in SAP Datasphere, for example, by using the data as a source in the *Data Builder*. The data also becomes visible in the *Explorer*.

Open SQL

You need to create an OpenSQL Schema in the Space Management and share the details with the Data Provider. The Data Provider is then able to push the data directly into your tenant. Once data is delivered into the OpenSQL Schema by the data provider, it is then accessible through the [Data Builder](#) and the provided schema appears as a source. Inside this schema, you'll find the data as views or tables.

External Delivery

The data is delivered by sharing files outside SAP Datasphere.

Filter by Data Category

Data providers tag their products with related categories and keywords to make your search easier and more efficient.

Filter by Delivery Mode

Filter by Delivery Mode

Full

You can receive update of data. Updated data are available under [My Data Products](#).

One Time

Only one data delivery is provided without any additional corrections or updates.

Filter by Delivery Pattern

You can filter by delivery pattern if you want to get a data product that gets regular updates for example.

Filter by Distribution Type

You can only filter by [Public](#). Public data products are available for everyone.

Filter by Industry

Data providers tag their products with related categories and keywords to make your search easier and more efficient.

Filter by Provider

Filter by Provider

List of all available data providers

Choose which data providers and their products you like to include in the search result.

Filter by Aggregator

Content Aggregator is a company managing one or multiple data provider profiles. You can search by Aggregator, and see all products belonging to data providers managed by an aggregator.

Filter by Regional Coverage

You can choose the regions/countries you like to include in the search result.

Filter by SAP Application

Data Providers can tag one or multiple SAP applications to which their data product fits. This is done especially in cases when the external data shall be harmonized with internal data from the tagged SAP application. An example would be a financial KPI data product that works with SAP S/4HANA which would be the tagged SAP Application then.

Filter by Size Category

Filter by Size Category

List of all available data product sizes: S, M, L, XL, XXL, XXXL

Filter by the size of the data product, compared to t-shirt size method:

- S: less than 1000 number of records
 - M: between 1000 and 100000 number of records
 - L: between 100000 and 1 million number of records
 - XL: between 1 million and 10 million number of records
 - XXL: between 10 million and 100 million number of records
 - XXXL: more than 100 million number of records
-

4.1.2 Using Bookmarks

You can bookmark your favorite data providers and data products to quickly retrieve them.

If you want to save for future subscriptions you can use the bookmark feature contained in Data Marketplace.

To set a data provider or data product as a bookmark, just click on the bookmark icon on the corresponding data provider or data product page.

You can retrieve your personal bookmark list in the Data Marketplace main menu. It's then easy from this place to open your entries and navigate to the corresponding data provider or data product.

To remove a bookmark, again click the bookmark icon on the corresponding data provider or data product page.

4.2 Evaluating your Data Product

Each data product has a dedicated page that describes the data product in detail to allow a transparent elaboration.

Default Information

For each data product the following information is displayed by default:

Entry	Description
Views	How many times the data product was viewed in Data Marketplace.
Price	The data product's pricing model. See section Pricing below.
Lifecycle Status	The data products that you can see have the status Listed.
<i>Listing managed by</i>	It indicates if the data provider is managed by a content aggregator.

Overview

A description of the data product.

Sample Data

Some data products offer free sample data that you can activate and then use for testing purposes. Sample data can only be downloaded as Microsoft Excel files.

Product Details

Each product page provides details about your product and helps you to decide which product to download.

Product Details	Description	Comments
Data Category	Indicates one or more categories where the data product fits in.	
Industry	Indicates one or more industries where the data product fits in.	
SAP Application	Indicates one or more SAP Applications where the data product fits in.	
Regional Coverage	Indicates the countries and regions for which the data product is assigned for.	
Data Shipment	Indicates the technical shipment of the data product. It varies depending on how the data provider offers their products. For more information, Data Shipment [page 65] .	Managed by Data Marketplace: <ul style="list-style-type: none">• Direct Download Managed by the data provider: <ul style="list-style-type: none">• Open SQL• External Delivery
Delivery Mode	Indicates if new deliveries can be offered by the data provider (updates in data, for example) and how these new deliveries can affect existing ones.	<ul style="list-style-type: none">• One-Time Only one data delivery is provided without any additional corrections or updates.• Full You can receive update of data. Updated data are available under My Data Products.
Delivery Pattern	Indicates when and how often data products are updated.	For example: <ul style="list-style-type: none">• Biweekly• Daily• Monthly• Other• Quarterly• Weekly• Yearly

Product Details	Description	Comments
Size Category	Indicates the size of your product, using the t-shirt size method.	<ul style="list-style-type: none"> • S: less than 1000 number of records • M: between 1000 and 100000 number of records • L: between 100000 and 1 million number of records • XL: between 1 million and 10 million number of records • XXL: between 10 million and 100 million number of records • XXXL: more than 100 million number of records
Visibility	Indicates who can view the product details.	Currently only public data products can be offered and can be seen by all Data Marketplace users.
Product ID	The unique identification of a data product.	
Additional Documents	Additional documents can be attached to the data product.	
Data Provider Page	The data provider profile page provides additional information about the data provider and other products or product groups.	

Pricing

In this section you find pricing information like price and the pricing model. If the data provider offers the possibility to buy a license in an external store, you can click on the given URL to purchase your own license.

4.2.1 Testing with Sample Data

Use sample data to see whether the data product is something for you.

Some data products offer free sample data that you can download and then use for testing purposes. The data is provided with an Excel file that you can download to your local computer.

4.2.2 Data Shipment

See how the data product will be delivered to your space.

The delivery of data products is either managed by Data Marketplace (direct) or by the data provider (indirect). Depending on the physical shipment that your data provider offers, you might need to provide some additional information.

The delivery is managed by Data Marketplace:

Data Shipment	Description
Integrated Delivery	The data is copied directly into the space you've selected. After activating the product, a protocol is created in My Data Products → Delivery Tracking . Once the data is available, you can use the data as any other data in SAP Datasphere, for example, by using the data as a source in the Data Builder . The data also becomes visible in the Explorer .

The delivery is managed by the data provider:

Open SQL	You need to create an OpenSQL Schema in the Space Management and share the details with the Data Provider. The Data Provider is then able to push the data directly into your tenant. Once data is delivered into the OpenSQL Schema by the data provider, it is then accessible through the Data Builder and the provided schema appears as a source. Inside this schema, you'll find the data as views or tables.
External Delivery	The data is delivered by sharing files outside SAP Datasphere.

4.2.3 Contract Type and Pricing Model

The way the data product is delivered depends on the contract type and pricing model applied by the data provider.

The data provider can propose different types of contract:

Contract Type	Descriptions
License Key	<p>An activation key provides authorization to access one or multiple data products. The data provider sends the key via an email after a price has been determined between the consumer and provider. The price is negotiated outside of data marketplace.</p> <p>Enter the activation key you received, accept the terms of use, and select a space where the data should be stored.</p> <p>License keys can be used:</p> <ul style="list-style-type: none"> • for commercial products and are provided by the data provider after closure of the commercial agreement, • for any other type of product that needs access authorization (for example for scenarios where non-public data is exchanged without payment). <p>For more information, see Acquiring a Data Product that needs a License Key [page 70].</p>
Free	<p>Data products that are free of charge are provided without any payment. This could be public domain data or data provided by a commercial data provider as give-away data, for example. The data can be activated after accepting the data specific terms of use.</p> <p>After accepting the terms of use, and selecting your space, the data product is downloaded.</p> <p>For more information, see Acquiring a Free Data Product [page 67] and Acquiring a Free Data Product on Request [page 68].</p>
On Request	<p>To activate a data product, you need to send a request to the data provider. You can do this directly through the user interface where an email template is created. You can edit the predefined email and then send it to the provider who then gets in touch with you.</p> <p>For more information, see Acquiring a Data Product on Request [page 69].</p>

Each of these contract types can be used with one of the following pricing models:

Pricing Model	Field	Description
One Time	Pricing Description	Some words on the pricing model
	Price	Price of the one time subscription
	Currency Code	Currency Code to be used, for example: EUR, USD

Pricing Model	Field	Description
Monthly	Pricing Description	Some words on the pricing model
	Price per Month	Price of the monthly subscription
	Currency Code	Currency Code to be used, for example: EUR, USD

4.3 Acquiring a Data Product

Acquire a data product and make it available to your space for further use.

The way you receive data products, depends on the contract type and pricing model that the data provider has defined for the data product. For more information see [Data Shipment \[page 65\]](#) and [Contract Type and Pricing Model \[page 65\]](#).

You can acquire a product using different ways. For more information, refer to following chapters:

- [Acquiring a Free Data Product \[page 67\]](#)
- [Acquiring a Free Data Product on Request \[page 68\]](#)
- [Acquiring a Data Product on Request \[page 69\]](#)
- [Acquiring a Data Product that needs a License Key \[page 70\]](#)

Once the data product with direct download delivery mode is available to your space, the status of the delivery becomes available on the *Delivery Tracking* tab in the app *My Data Products*. You can select the data product as a source. For example, when in the *Data Builder*, you can create a new graphical view. Under *Sources* → *Connections*, you'll find the data products with their corresponding schema.

For more information, see [Managing your Data Products \[page 71\]](#).

4.3.1 Acquiring a Free Data Product

A free data product does not need a license and can be downloaded directly from the data marketplace.

Prerequisites

You need a space and you need to be assigned to this space as a user.

You need to have either the roles DW Integrator and DW Modeler, as the data marketplace creates artifacts and loads data. If you do not have sufficient privileges, the load button is disabled.

Procedure

1. On the relevant data product page, click [Load for Free](#).
2. Select a space to download the data product to.

When you click on the icon on the right-hand side of the input field you get a list of all spaces that are assigned to you. Select one of them. You can also type the name of the space in the input field. You will receive auto-suggests as you type.

Note

If the product was already installed in one space, you can't install it a second time in this space.

3. Select an [Update Mode](#) if necessary.

Note

Data products that are delivered in full mode can receive updates. Updates are available under [My Data Products](#) > [Update \(tab\)](#).

4. Read the terms of use and check the additional information if available.
5. Click [Load Product](#) to download the data in your space.

Results

Once the download is finished, you can work with the data within your space.

4.3.2 Acquiring a Free Data Product on Request

A data provider offers a free product, but it can't be downloaded directly from the data marketplace.

Prerequisites

You need a space and you need to be assigned to this space as a user.

You need to have either the roles DW Integrator and DW Modeler, or the DW Admin role as the data marketplace creates artifacts and loads data.

Context

You want to acquire a data product that is available on request only.

Procedure

1. On the relevant data product page, click [Request for free](#).
2. Enter relevant information in the template mail to request the data provider to deliver the data product. The data provider will contact you and deliver the data either using an "Open SQL" delivery or via an "External Delivery. For more information, see [Acquiring a Data Product \[page 67\]](#).

4.3.3 Acquiring a Data Product on Request

A data provider offers a data product but it can't be downloaded or activated directly from the data marketplace.

Prerequisites

You need a space and you need to be assigned to this space as a user.

If the product is delivered using the "Open SQL" delivery mode, you need access to a space and an Open SQL schema with credentials that you can share with the data provider.

Context

You want to acquire a data product that is available on request only.

Procedure

1. On the relevant data product page, click [Request Product](#).
2. Enter relevant information in the template mail to start negotiation with the data provider. You'll negotiate outside the Data Marketplace the payment and how the data will be delivered. Once the data provider has received the payment, the data will be delivered either using an "Open SQL" delivery or via an "External Delivery. For more information, see [Acquiring a Data Product \[page 67\]](#).

4.3.4 Acquiring a Data Product that needs a License Key

A data provider offers a data product that needs a license key.

Prerequisites

You need a space and you need to be assigned to this space as a user.

Context

You want to acquire a product that needs to be activated with a license key.

Procedure

1. If the data provider uses an external shop for the purchasing of licenses, click on [Get License](#) on top of the data product page. After you read through the disclaimer text, click on [Agree](#). You will be forwarded to the shop of the data provider. Here you can purchase the license for the data product. Proceed with step 4. In case there is no [Get License](#) button, contact the data provider with the e-mail icon and follow steps 2-3.
2. Negotiate with the data provider outside the Data Marketplace the payment method and the price.
3. Once payment is done, the data provider will send the license key.
4. Go back to the relevant data product page and click [Load with License Key](#).
5. Enter the activation key you have received from your data provider.
6. Select a space to download the data product to.

When you click on the icon on the right-hand side of the input field you get a list of all spaces that are assigned to you. Select one of them. You can also type the name of the space in the input field. You will receive auto-suggests as you type.

Note

If the product was already installed in one space, you can't install it a second time in this space.

7. Select the [Update Mode](#) that fits you needs.

Data products that are delivered in full mode can receive updates. Updates are available under [My Data Products](#) > [Update \(tab\)](#).

8. Read the terms of use and check the additional information if available.
9. Click [Load Product](#).

Results

The data is available in your space and can be selected as a source.

Note

Once the license is expired, the product can neither be installed into other spaces nor updates can be triggered.

4.4 Managing your Data Products

Keep track of your deliveries and updates after acquiring your data products.

The [My Data Products](#) page gives an overview of all activated data products with direct download delivery, their updates, and delivery status.

Note

Note that Open SQL products or products with external delivery will not be shown on the [My Data Products](#) page.

Tab: My Products

On this tab you get an overview over:

- total activations
- top providers
- the validity of your licenses

You also get a complete overview of all your activated data products with direct download delivery, with details.

This information is displayed in a table containing the following details:

Attribute	Description
Activation Date	The date when the data product was activated
Provider Name	The provider's name. Following the link to go to their page.
Product Name	The name of the data product that was activated. Click the link to go to the product page.
Delivery Mode	Delivery mode. For more information see Data Shipment [page 65] .
Payment Method	Indicates whether the data product was activated with a License or if it is free of charge. For more information, see Contract Type and Pricing Model [page 65]

Attribute	Description
License Validity	The date until when the license is valid.
	<div style="border: 1px solid #ccc; background-color: #f9f9f9; padding: 10px;"> <p>Note</p> <p>For free data products this field will be empty as they have no expiration date.</p> </div>
Space	The space that was selected to store the data product.

To delete a downloaded data product from the space, click on the bin icon.

Tab: Updates

Depending on the update mode you can track the status of the automatic data updates or start an update manually.

For more information, see [Updating your Data Products \[page 72\]](#).

Tab: Delivery Tracking

On this tab you can check the status of your data deliveries.

For more information, see [Tracking your Deliveries \[page 73\]](#).

4.4.1 Updating your Data Products

Keep track on recurring data product updates.

The tab [Updates](#) on the [My Data Products](#) app contains an overview of all your activated data products with direct download delivery, showing update-specific details. If there is a new release for your data product available, the status *Outdated* is displayed. Then update the data product manually.

The following update-specific details are displayed in the table:

Attribute	Description
Activation Date	The date when the data product was activated
Provider Name	The provider's name. Following the link to go to their page.
Product Name	The name of the data product that was activated. Click the link to go to the product page.

Attribute	Description
Delivery Mode	Indicates the delivery mode. For more information see Data Shipment [page 65] .
Last Update	The date of the last update.
Update Mode	The way the updates are delivered. Data products that are delivered in full mode can receive updates. Updates are available under ► My Data Products ► Update (tab) ▾.
<div style="background-color: #f0f0f0; padding: 10px; border: 1px solid #ccc;"> <p>Note</p> <p>One time products don't offer updates. That's why the update mode can not be changed.</p> </div>	
License Validity	The date until when the license is valid.
Space	The space that was selected to store the data product.
Status	The download status of the data product: <ul style="list-style-type: none"> • Up-to-date: Your data is up-to-date. This means that you've already downloaded the latest version. • Outdated: A new version is available for your data. You can download it to get a data update. • Failed: The download of data has failed. You need to restart it or contact the data provider if you can't download the data after a while. • Downloading: The download of data is running. • Deleting: The data deletion is running. • Deleted: The data is deleted and can't be accessed anymore.
Product Lifecycle	Indicates whether the data product is listed or not in Data Marketplace.

To update a data product, select it in the list and click the >> icon.

4.4.2 Tracking your Deliveries

You can check whether your data products are up-to-date or if updates are running or if they failed.

Attribute	Description
Activation Date	The date when the data product was activated
Provider Name	The provider's name. Click on the link to go to their profile page.

Attribute	Description
Product Name	The name of the data product that was activated. Click the link to go to the product page.
Space	The space that was selected to store the data product.
Status	<p>The download status of the data product can be:</p> <ul style="list-style-type: none"> • <i>Up-to-date</i>: Your data is up-to-date. This means that you've already downloaded the latest version. • <i>Outdated</i>: A new version is available for your data. You can download it to get a data update. • <i>Failed</i>: The download of data has failed. You need to restart it or contact the data provider if you can't download the data after a while. • <i>Downloading</i>: The download of data is running. • <i>Deleting</i>: The data deletion is running. • <i>Deleted</i>: The data is deleted and can't be accessed anymore.
>	Opens the details area where further information on the latest deliveries is displayed. By clicking on > you get to the complete list with all information (start time, duration, number of records, etc) on the update.

To update the displayed information in the table, click on the Refresh icon above the table.

4.5 Managing your Licenses

Keep track of your licenses.

In this app you can view and manage all licenses you have activated or in use.

Attribute	Description
Data Provider	The name of the data provider who licensed the data product to you.
Reference	Describes a reference.
Valid until	The date until when the license is valid.
Status	The status of the license.

To just license a data product to you, without installing the data product, click on [Enter Activation Key](#).

4.6 Managing your Contexts

Manage the contexts you are member of.

The *My Contexts* app gives you an overview of all contexts you are a member of. You can join new contexts or leave the ones you don't want to be member of any longer.

Depending on the data providers' choice, they can publish their data products in the *Public Data Marketplace*, which is managed by SAP for all users of SAP Datasphere or in their own private data marketplace.

Contexts help data providers to differentiate between public and private data marketplaces. In a private data marketplace only you and others who have joined the corresponding context as members have access to the data products offered by the private data provider. Private data products cannot be found in the Public Data Marketplace in SAP Datasphere.

To become a member of a context, you must activate a member activation key. You will receive this key from the data provider. When you are member of a context, you have access to all data products which are published in this context.

You find the *My Contexts* app within the Data Marketplace menu.

The app shows a table including details of the contexts you are a member of:

Attribute	Description
Context Owner ID	Shows the icon and name of the data provider who owns the context. By clicking on the name you can open the data provider profile page.
Context Name	Shows the icon and name of the context.
Hidden	Specifies whether the data products of this context are hidden or visible on the landing page and via search. You can use this flag as a filter, to limit the visibility of data products which belong to a certain context only.
Validity	Shows the validity date of the context. If no date is specified, the context will not expire.
Status	The status of the context can be: <ul style="list-style-type: none">• Active• Expired
>	Opens the details area where further information on the context (like description, domains, etc) is displayed. Click on <i>X</i> to close the details area.

Joining a Context

To join a context, become a member by clicking on the button *Join*.

Enter the context member activation key which you have received from the data provider who is also the context owner and click on the button *Activate*.

You can now find and acquire the data products which are available in this context. Remember that you still would need a licence key for some data products.

Leaving a Context

To leave a context, choose the respective context from the table and click on the button [Leave](#). Confirm your action.

If you leave a context, you will no longer be able to find or acquire the data products which are published in this context. Data products which you have purchased will still be available in your [My Data Products](#) app and will receive also future updates.

Note

It's not possible to leave the context [Public Data Marketplace](#) by SAP which is available by default for all users of Data Marketplace. You can set this context to hidden if you don't want to see its data products on the landing page and while searching.

4.7 Frequently Asked Questions by Data Product Consumers

This topic answers frequently asked questions from the data product consumers point of view.

Frequently Asked Questions and Answers

What is an activation key and how can I get it?

An activation key provides authorization to access one or multiple data products. The data provider sends the key via an email after a price has been determined between the consumer and provider. The price is negotiated outside of data marketplace.

How can I contact a data provider?

You can contact a data provider either through the data provider profile or the product page. In both cases an option is provided which allows you to send an email to the data provider.

What different data products are there?

To find out which specific data products are available in your own Data Marketplace, you go to the landing page and start a search or browse through the displayed categories.

Where do I find the data that I've downloaded?

Once the data product with direct download delivery mode is available to your space, the status of the delivery becomes available on the [Delivery Tracking](#) tab in the app [My Data Products](#). You can select the data product

as a source. For example, when in the *Data Builder*, you can create a new graphical view. Under *Sources* → *Connections*, you'll find the data products with their corresponding schema.

How can I remove data products?

It's possible to delete data products from your space. Select the data product on the *My Products* tab in the app *My Data Products* and click on the bin icon.

For more information, see [Managing your Data Products \[page 71\]](#).

What to do when data products or updates don't work anymore?

If you find that updates or data products don't work anymore, you should contact your administrator and/or your data provider.

Can I forward activation keys?

Yes any Data Marketplace user can activate a data product with the appropriate activation key.

Can I activate a data product to multiple spaces?

Yes, you can select other spaces as well. After you've activated a data product, you can select to activate it again and then choose a different space.

What do the multiple versions of a data product mean?

A version is a new release which contains a different data set. The data can be offered for a certain period of time, for example there can be monthly releases which contain data for the last 30 days.

What is the prerequisite for me to download and use data products?

To download data products, you need to have a space where your data products can be saved in.

In addition, depending on the data product that you're interested in, it might be necessary to obtain a license key from your data provider. Some data products might also require you to provide additional information, like an SQL schema.

How can I pay for data products?

Data marketplace provides a platform for data providers and consumers to easily transfer data products, but the price for one or multiple data products is negotiated between the provider and consumer outside of data marketplace. Once a price has been negotiated, the data provider sends an activation key to the consumer. With this key the buyer is authorized to activate the data product.

How do I receive new data from the Data Marketplace?

This depends on the type of data product that you downloaded. If you downloaded a data product with recurring updates, you can go to the app *My Data Products* → *Updates* and do a manual update. When exactly an update becomes available depends on your data provider and can vary. You can find this information in the details area of your product under "Periodic Data Updates".

For more information, see [Updating your Data Products \[page 72\]](#).

How long does it take to download data from the Data Marketplace?

This depends on the data product's type and size.

Are there restrictions on how to use the data I got from the Data Marketplace?

To be on the safe side, read the terms of use before you download and use a certain data product.

How can I report the misuse of data in Data Marketplace?

In this case contact the data provider. You find their contact information on the data provider profile page.

4.8 Glossary

Find all important terms and definitions here.

Terms and Definitions

Term	Definition
License Key	An activation key provides authorization to access one or multiple data products. The data provider sends the key via an email after a price has been determined between the consumer and provider. The price is negotiated outside of data marketplace.
Data Product	A data product is a set of data that can either be acquired by consumers or that can be offered by data providers through the data marketplace.
Data Provider	A data provider is a person or company that offers one or multiple data products through data marketplace.
Pricing Model	A pricing model mode defines how often the data is delivered: either "One-time" (no other delivery will happen) or "Full" (new data deliveries replace existing ones). For more information, see Contract Type and Pricing Model [page 65]
Data Shipment	A data shipment defines how the the data is shipped to the consumer. It's either managed by Data Marketplace or by the Data provider. For more information, see Data Shipment [page 65]

5 Acquiring Data in the Data Builder

Users with the *DW Modeler* role can import data directly into the *Data Builder* from connections and other sources, and use flows to replicate, extract, transform and load data.

This topic contains the following sections:


- [Federate and Replicate Data in Remote Tables \[page 79\]](#)
- [Extract, Transform, and Load Data with Data Flows \[page 80\]](#)
- [Load Data from Multiple Objects with Replication Flows \[page 80\]](#)
- [Load and Transform Data \(Including Delta Change Support\) with Transformation Flows \[page 80\]](#)
- [Import Entities from SAP S/4HANA \[page 80\]](#)
- [Import Entities from SAP BW Bridge \[page 81\]](#)
- [Import Data from CSV Files \[page 81\]](#)
- [Purchase Data from Data Marketplace \[page 81\]](#)
- [Create and Import Objects to Receive and Prepare Data \[page 81\]](#)
- [Create Objects and Act On Existing Objects \[page 81\]](#)

Space administrators and integrators prepare connections and other sources to allow modelers to acquire data (see [Integrating Data and Managing Spaces in SAP Datasphere](#)).

Federate and Replicate Data in Remote Tables

Many connections (including most connections to SAP systems) support importing remote tables to federate or replicate data (see [Integrating Data via Connections](#)).


You can import remote tables to make the data available in your space from the *Data Builder* start page, in an entity-relationship model, or directly as a source in a view.

- To get started: In the side navigation area, click  (*Data Builder*), select a space if necessary, and click [▶ Import ▶ Import Remote Tables ▶](#). See [Import Remote Tables \[page 91\]](#).
- By default, remote tables federate data, and each time the data is used a call is made to the remote system to load it. You can improve performance by enabling replication to store the data in SAP Datasphere. Some connections support real-time replication and for others, you can keep your data fresh by scheduling regular updates (see [Replicate Remote Table Data \[page 98\]](#)).
- To optimize replication performance and reduce your data footprint, you can remove unnecessary columns and set filters (see [Restrict Remote Table Data Loads \[page 95\]](#)).
- To maximize access performance, you can store the replicated data in-memory (see [Accelerate Table Data Access with In-Memory Storage \[page 100\]](#)).
- Once a remote table is imported, it is available for use by all users of the space and can be used as a source for views.
- You can automate sequences of data replication and loading tasks with task chains (see [Creating a Task Chain \[page 197\]](#)).

Extract, Transform, and Load Data with Data Flows

Many connections (including most connections to SAP systems) support loading data to SAP Datasphere via data flows (see [Integrating Data via Connections](#)).

Data flows support a wide range of extract, transform, and load (ETL) operations.

- To get started: In the side navigation area, click  (*Data Builder*), select a space if necessary, and click *New Data Flow* to open the editor. See [Creating a Data Flow \[page 128\]](#).
- To add a source to your data flow, drag it from the *Source Browser* (see [Using the Source Browser \[page 292\]](#)).
- In addition to connections, data flows can load and transform data from the following kinds of sources:
 - Open SQL schemas (see [Integrating Data via Database Users/Open SQL Schemas](#))
 - HDI containers (see [Exchanging Data with SAP SQL Data Warehousing HDI Containers](#)).
 - Objects that are already in the SAP Datasphere repository (see [Add Objects from the Repository \[page 292\]](#)).
- Data flows load data into local tables.
- You can automate sequences of data replication and loading tasks with task chains (see [Creating a Task Chain \[page 197\]](#)).

Load Data from Multiple Objects with Replication Flows

Certain connections support loading data from multiple source objects to SAP Datasphere via a replication flow. You can enable a single initial load or request initial and delta loads and perform simple projection operations (see [Creating a Replication Flow \[page 155\]](#)).

Load and Transform Data (Including Delta Change Support) with Transformation Flows

Create a transformation flow to load data from one or more source repository tables, apply transformations, and output the result to a target table. You can load a full set of data or only delta changes from each source table (see [Creating a Transformation Flow \[page 173\]](#)).

Import Entities from SAP S/4HANA

The *Import Entities* wizard allows you to import entities from and SAP S/4HANA on-premise systems with rich metadata (see [Importing Entities with Semantics from SAP S/4HANA \[page 83\]](#)).

Import Entities from SAP BW Bridge

SAP BW bridge enables you to use SAP BW functionality in the public cloud and to import bridge entities into SAP Datasphere (see [Importing Entities with Semantics from SAP BW/4HANA or SAP BW Bridge \[page 86\]](#)).

Import Data from CSV Files

You can import data from a CSV file to create a new local table (see [Creating a Local Table from a CSV File \[page 122\]](#)).

Purchase Data from Data Marketplace

Purchase data products from providers and download them directly into your space (see [Purchasing Data from Data Marketplace \[page 57\]](#)).

You can become a data provider and offer your own data products for sale in Data Marketplace via the Data Sharing Cockpit (see [Data Marketplace - Data Provider's Guide](#)).


Create and Import Objects to Receive and Prepare Data

You can create and import empty tables and views to receive and prepare data:

- You can create an empty local table ready to receive data from a CSV file or from a data flow (see [Creating a Local Table \[page 106\]](#)).
- You can import business content prepared by SAP and partners to support end-to-end business scenarios (see [Importing SAP and Partner Business Content from the Content Network](#)).
- You can import object definitions from a CSN/JSON file (see [Importing Objects from a CSN/JSON File \[page 49\]](#)).

Create Objects and Act On Existing Objects

All the objects you import or create in the *Data Builder* are listed on the *Data Builder* start page. You can act on objects in the list in the following ways:

- Click one of the tabs to filter the list by object type.
- Click a tile to create a new object
- Click  (*Show filters*) to filter the list on collections and search by criteria. Click *Show More* to open a dialog with additional filter options.
- Enter a string in the *Search* field to filter the list on business and technical names and users.

- Click a column header to sort or filter the list by values in the column.
- Select one or more objects and use any of the following tools:

Tool	Description
New	Create <i>Data Builder</i> objects (independent of any selection).
Import	Import objects from files and connections: <ul style="list-style-type: none"> • <i>Import CSV File</i> - Import data from a CSV file to create a local table (see Creating a Local Table from a CSV File [page 122]) • <i>Import Objects from CSN/JSON File</i> - Import table and view definitions from a CSN file to create tables and views or import data flow definitions from a JSON file to create data flows. (see Importing Objects from a CSN/JSON File [page 49]). • <i>Import Remote Tables</i> - Import remote tables from a connection (see Import Remote Tables [page 91]).
Edit	Open the selected object in the appropriate editor. Alternatively, click the object directly in the list.
Deploy	Select one or more objects and deploy them at once. Choose from the following entity types: <ul style="list-style-type: none"> • Local Table (see Creating a Local Table [page 106]) • Graphical View (see Creating a Graphical View [page 213]) • SQL View (see Creating an SQL View [page 250]) • Data Access Control (see Create a "Single Values" Data Access Control) • Analytic Model (see Creating an Analytic Model [page 337]) • Task Chain (see Creating a Task Chain [page 197]) Other types of objects cannot be deployed. If one or more objects that you have selected cannot be deployed, the <i>Deploy</i> dialog opens, allowing you to review your selection. Click <i>Deploy</i> to deploy those objects listed on the <i>Deployable</i> tab, or <i>Cancel</i> to go back and alter your selection.
Share	Share the selected tables and views to other spaces (see Sharing Tables and Views To Other Spaces [page 44]).
Impact and Lineage Analysis	View the objects that depend on an analyzed object (its impacts) and the objects on which the analyzed object depends (its lineage)(see Impact and Lineage Analysis [page 34]).
Delete	Delete the selected objects.

Note

If the object is used by one or more other objects then a dialog listing these dependencies opens, and the deletion is canceled.

Note

In various places in editors where there is not enough room to show both business and technical names for entities and columns, you can choose which of these names to display. Click ► *Profile* ► *Settings* ► *UI Settings* ► and select either *Show Business Name* or *Show Technical Name*.

5.1 Importing Objects with Semantics from SAP S/4HANA, SAP BW/4HANA and SAP BW Bridge

Users with the *DW Modeler* role (or equivalent privileges) can use the *Import Entities* wizard to import semantically-rich objects from your SAP S/4HANA Cloud, SAP S/4HANA on-premise, SAP BW/4HANA, and SAP BW bridge connections. The wizard creates *Business Builder* and *Data Builder* entities (along with all the objects on which they depend) in SAP Datasphere.

Objects with the following modeling patterns can be imported:

Modeling Pattern	Supported Connections
ANALYTICAL_QUERY	SAP BW/4HANA, SAP BW bridge
ANALYTICAL_CUBE	SAP BW/4HANA, SAP BW bridge
ANALYTICAL_FACT	SAP S/4HANA on-premise only
ANALYTICAL_DIMENSION	SAP BW bridge, SAP S/4HANA Cloud, SAP S/4HANA on-premise
LANGUAGE_DEPENDENT_TEXT	SAP BW bridge, SAP S/4HANA Cloud, SAP S/4HANA on-premise
ANALYTICAL_PARENT_CHILD_HIERARCHY_NODE	SAP S/4HANA Cloud, SAP S/4HANA on-premise

5.1.1 Importing Entities with Semantics from SAP S/4HANA

You can use the *Import Entities* wizard to load metadata from your SAP S/4HANA Cloud and SAP S/4HANA on-premise connections via semantically-rich objects. The wizard creates *Business Builder* and *Data Builder* entities (along with all the objects on which they depend) in SAP Datasphere.

Context

You can import entities from the following types of sources:

- SAP S/4HANA Cloud (see [SAP S/4HANA Cloud Connections](#)).

Note

All the connectivity preparations for model import must be completed, including the activation of the necessary communication scenarios in the SAP S/4HANA Cloud system (see [Prepare Connectivity to SAP S/4HANA Cloud](#)).

- SAP S/4HANA On-Premise (see [SAP S/4HANA On-Premise Connections](#)).

Note





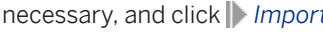
The on-premise system must be v1610 or higher, and all the connectivity preparations must be completed (see [Prepare Connectivity to SAP S/4HANA On-Premise](#)).

We recommend that, where possible, you use the *Import Entities* wizard for importing CDS views from these connection types, as it is able to leverage their rich semantics to import higher-level objects and to follow associations to dimensions, hierarchies, and text entities and include them in the import.

Note

Only SAP S/4HANA Cloud connections can include associated entities in the import. SAP S/4HANA on-premise connections cannot follow associations, but the information is included in each imported object's CSN definition, and associations will be automatically recreated in SAP Datasphere if their target entities are already present or are subsequently imported.

Procedure


1. Open the wizard from *Semantic Onboarding*, the *Repository Explorer*, or the *Data Builder*:
 - In the side navigation area, click  (*Repository Explorer*), and click .
 - In the side navigation area, click  (*Semantic Onboarding*), and then click the appropriate tile.
 - In the side navigation area, click  (*Data Builder*), select a space if necessary, and click .
2. Select your connection and target space:

Semantic Onboarding

Select the connection to the source you want to import from from the list, and then click *Next Step*.

You can use the *Filter By* pane on the left to filter the connections by space.

Repository Explorer / Data Builder

1. Select the *SAP S/4HANA Cloud* or *SAP S/4HANA* connection type, and then click *Next Step*.
2. Select the target space that you want to import the entities into, and click *Next Step*.
If you opened the wizard from the  (*Data Builder*), only your current space can be selected.
3. Click the specific connection to the source you want to import from, and click *Next Step*.

3. On the *Select Entities* page, select the entities that you want to import from the *Results* list, and click *Next Step*.

Use the *Search* field to search by label. Depending on your connection type, you can filter by:

- Application Component
- Modeling Pattern (see [Supported Capabilities for CDS Views in the SAP S/4HANA Cloud](#) documentation)
- Release Contract (see [Stability Contracts for CDS Views in the SAP S/4HANA Cloud](#) documentation)

Note

Entities that have a release contract of *CI* (and that are extraction-enabled) can be imported as remote tables, which support data federation and replication natively. Other entities can only be imported as local tables and you must provide data replication manually by creating a data flow.

- Software Component
4. On the *Review Entities* page, review the entities that you will import.

The left pane lists the entities you have selected for import and the right pane has two tabs listing all the entities that will be created in the SAP Datasphere *Business Builder* and *Data Builder* during the import of the selected entity.

The following entities are created when you import entities with the most common modeling patterns:

Source Modeling Pattern	Business Builder Objects Created	Data Builder Objects Created
ANALYTICAL_FACT (SAP S/4HANA on-premise only)	<ul style="list-style-type: none"> • Selected entity: <ul style="list-style-type: none"> • One business entity (<i>Fact</i>) • Supporting entities: <ul style="list-style-type: none"> • One entity (<i>Dimension</i>) for each associated dimension. 	<ul style="list-style-type: none"> • Selected entity: <ul style="list-style-type: none"> • One remote table (<i>Relational Dataset</i>) • Supporting entities: <ul style="list-style-type: none"> • One remote table for each associated dimension, hierarchy, and text entity.
ANALYTICAL_DIMENSION	<ul style="list-style-type: none"> • Selected entity: <ul style="list-style-type: none"> • One business entity (<i>Dimension</i>) • Supporting entities: <ul style="list-style-type: none"> • One entity (<i>Dimension</i>) for each associated dimension. 	<ul style="list-style-type: none"> • Selected entity: <ul style="list-style-type: none"> • One view (<i>Dimension</i>) • One remote table (<i>Relational Dataset</i>) • Supporting entities: <ul style="list-style-type: none"> • One view and one remote table for each associated dimension, hierarchy, and text entity.
LANGUAGE_DEPENDENT_TEXT	(none)	<ul style="list-style-type: none"> • Selected entity: <ul style="list-style-type: none"> • One remote table (<i>Text</i>)
ANALYTICAL_PARENT_CHILD_HIERARCHY_NODE	(none)	<ul style="list-style-type: none"> • Selected entity: <ul style="list-style-type: none"> • One remote table (<i>Hierarchy</i>)

Note

When importing from an SAP S/4HANA on-premise connection, only a business entity and a remote table are created for the selected entity. Associated objects are not created.




Each object has the following read-only properties:

Property	Description
Business Name	Generated from the <i>Label</i> value.
Technical Name	Generated from the <i>ID</i> value as: <code>remote.<connection_name>.<ID></code> .

Property	Description
Import Status	<p>Specifies whether the object needs to be created in SAP Datasphere during the import:</p> <ul style="list-style-type: none"> • <i>Ready for Import</i> - The object will be created. • <i>Already Imported</i> - The object has already been created during a previous import. If the structure or semantics of the object have subsequently changed in the source system, the object will be automatically updated and redeployed.

5. Click *Import and Deploy*.

A notification is sent immediately, and this notification will update as the import process continues. When the import is complete:

- You have two notifications:
 - Click the first notification to see the imported entities listed in the  (*Repository Explorer*).
 - Click the second notification to view the import log messages. If any entity could not be created, an error is given.
- The imported entities are available for use in the  (*Business Builder*) and the  (*Data Builder*).

6. Navigate to the appropriate objects to review data access:

- *Remote Tables* - By default, data is only federated. To replicate the data, open the *Data Integration Monitor* (see [Replicating Data and Monitoring Remote Tables](#)).
- *Local Tables* - In cases where only local tables are imported, you must create your own flow to load data to them (see [Creating a Replication Flow \[page 155\]](#)).

5.1.2 Importing Entities with Semantics from SAP BW/4HANA or SAP BW Bridge

You can use the *Import Entities* wizard to load metadata from your SAP BW/4HANA and SAP BW bridge connection via semantically-rich objects. The wizard creates *Business Builder* and *Data Builder* entities (along with all the objects on which they depend) in SAP Datasphere.

Prerequisites

You must have a user in the bridge tools with the business role template `SAP_DW4_BC_MODELING_DWC_PC`. For more information, see [Business Role Templates](#) in the *SAP Datasphere, SAP BW Bridge* documentation.

Context

You can import entities from the following types of sources:

- SAP BW/4HANA (see [SAP BW/4HANA Model Transfer Connections](#)).

Note

When you re-import a SAP BW/4HANA entity, the imported table in SAP Datasphere is overwritten. So in case you have made changes to the imported table, these changes will be lost.

If you would like to keep the changes you have made to the table in SAP Datasphere, you can refresh the table in the table editor in order to get updates from the original entity in the SAP BW/4HANA system.






- SAP BW bridge
If the SAP Datasphere, SAP BW bridge option is enabled:
 - A SAP BW bridge space is created, containing an SAP BW bridge connection to your bridge tools.
 - You can use the *Import Entities* wizard to import your SAP BW bridge Queries, CompositeProviders, DataStore objects (inbound table and/or active table depending on the type of the DataStore object), and Master data tables:
 - Remote tables to load the data are created in the SAP BW bridge space and then automatically shared to the target space you specify in the wizard.
 - Views and business entities that draw their data from and are built on these remote tables are created in your target space.

For importing queries, there are certain restrictions:

- When an object is remodeled in SAP BW bridge, the query definition is broken.
- Input and exit variables are removed.
- Constant selection is not supported.
- For dimensions, the following features are not supported: external hierarchies, compound characteristics, and time dependency.

You can use the app *Query* to get an overview on the supported analytical queries. For more information, see [Overview of Apps](#).


Procedure

1. Open the wizard from *Semantic Onboarding*, the *Repository Explorer*, or the *Data Builder*:
 - In the side navigation area, click  (*Repository Explorer*), and click .
 - In the side navigation area, click , and then click the appropriate tile.
 - In the side navigation area, click , select a space if necessary, and click .
2. Select your connection and target space:

Select the connection to the source you want to import from from the list, and then click *Next Step*.

You can use the *Filter By* pane on the left to filter the connections by space.

1. Select the *SAP BW/4HANA* or *SAP BW Bridge* connection type, and then click *Next Step*.
2. Select the target space that you want to import the entities into, and click *Next Step*.

If you opened the wizard from the  (*Data Builder*), only your current space can be selected.

Note

The *SAP BW Bridge* space cannot be selected as the target space. When importing entities from bridge, any remote tables are imported to the *SAP BW Bridge* space and automatically shared to your selected target space, where the remaining objects are created.

3. Click the specific connection to the source you want to import from, and click *Next Step*.

3. On the *Select Entities* page, select the entities that you want to import from the *Results* list, and click *Next Step*.

Use the *Search* field to search by label. Depending on your connection type, you can filter by:

- Application Component
- Modeling Pattern
- Software Component

4. On the *Review Entities* page, review the entities that you will import.

The left pane lists the entities you have selected for import and the right pane has two tabs listing all the entities that will be created in the SAP Datasphere *Business Builder* and *Data Builder* during the import of the selected entity.

The following entities are created when you import entities with the most common modeling patterns:

Note

If an associated entity cannot be imported, it is left out. The generated object however will still be a complete object.

Source Modeling Pattern	Business Builder Objects Created	Data Builder Objects Created
ANALYTICAL_QUERY	<ul style="list-style-type: none"> • Selected entity: <ul style="list-style-type: none"> • One consumption model containing a perspective. • Supporting entities: <ul style="list-style-type: none"> • One entity (<i>Fact</i>) • One entity (<i>Dimension</i>) for each associated dimension. 	<ul style="list-style-type: none"> • Supporting entities: <ul style="list-style-type: none"> • One view (<i>Fact</i>) • One remote table (<i>Relational Dataset</i>) • One view and one remote table for each associated dimension.




Source Modeling Pattern	Business Builder Objects Created	Data Builder Objects Created
ANALYTICAL_CUBE	<ul style="list-style-type: none"> Selected entity: <ul style="list-style-type: none"> One business entity (<i>Fact</i>) Supporting entities: <ul style="list-style-type: none"> One entity (<i>Dimension</i>) for each associated dimension. 	<ul style="list-style-type: none"> Selected entity: <ul style="list-style-type: none"> One view (<i>Fact</i>) One remote table (<i>Relational Dataset</i>) Supporting entities: <ul style="list-style-type: none"> One view and one remote table for each associated dimension.
ANALYTICAL_DIMENSION (SAP BW Bridge only)	<ul style="list-style-type: none"> Selected entity: <ul style="list-style-type: none"> One business entity (<i>Dimension</i>) Supporting entities: <ul style="list-style-type: none"> One entity (<i>Dimension</i>) for each associated dimension. 	<ul style="list-style-type: none"> Selected entity: <ul style="list-style-type: none"> One view (<i>Dimension</i>) One remote table (<i>Relational Dataset</i>) Supporting entities: <ul style="list-style-type: none"> One view and one remote table for each associated dimension, hierarchy, and text entity.
LANGUAGE_DEPENDENT_TEXT (SAP BW Bridge only)	(none)	<ul style="list-style-type: none"> Selected entity: <ul style="list-style-type: none"> One remote table (<i>Text</i>)

Each object has the following read-only properties:

Property	Description
Business Name	Generated from the <i>Label</i> value.
Technical Name	Generated from the <i>ID</i> value as: <code>remote.<connection_name>.<ID></code> .
Import Status	Specifies whether the object needs to be created in SAP Datasphere during the import: <ul style="list-style-type: none"> <i>Ready for Import</i> - The object will be created. <i>Already Imported</i> - The object has already been created during a previous import. If the structure or semantics of the object have subsequently changed in the source system, the object will be automatically updated and redeployed.

5. Click *Import and Deploy*.

A notification is sent immediately, and this notification will update as the import process continues. When the import is complete:

- You have two notifications:
 - Click the first notification to see the imported entities listed in the  (*Repository Explorer*).
 - Click the second notification to view the import log messages. If any entity could not be created, an error is given.
- The imported entities are available for use in the  (*Business Builder*) and the  (*Data Builder*).

6. [optional] By default, data is only federated to your remote tables. To replicate the data, open the *Data Integration Monitor* (see [Replicating Data and Monitoring Remote Tables](#)).

5.2 Importing Tables and Views from Sources

Import tables and views from a connection, Open SQL schema, HDI container or other source available in your space.

SAP Datasphere provides various methods for importing data into your space:

- *Repository Explorer / Data Builder* start page - Use the import wizards to:
 - Import semantically-rich objects (along with the objects they depend on) from SAP S/4HANA, SAP BW/4HANA, and SAP Datasphere, SAP BW Bridge connections (see [Importing Entities with Semantics from SAP S/4HANA \[page 83\]](#)).
 - Import remote tables from any connection (see [Import Remote Tables \[page 91\]](#)).
- E/R Model - Import tables into the space and add them to the E/R model diagram (see [Import an Object from a Connection or Other Source \[page 294\]](#) or [Import Multiple Objects from a Connection \[page 296\]](#)).
- Graphical or SQL view - Import tables into the space and use them directly as sources in the view (see [Add a Source \[page 217\]](#)).

Imported tables are created as remote tables or local tables depending on the type of source:

Source Type	Table Type
Connection (see Integrating Data via Connections)	Remote table - Data is federated by default Use the tools in the <i>Remote</i> section in the table editor to control replication of data (see Replicate Remote Table Data [page 98]).
Open SQL schema (see Integrating Data via Database Users/Open SQL Schemas)	Local table
HDI container (Exchanging Data with SAP SQL Data Warehousing HDI Containers)	As these sources are already integrated into SAP Datasphere in staging tables next to your space, there is no need to replicate data.
Partner tool (see Connections to Partner Tools)	

Note

For connection types that allow to enable remote tables, two SAP HANA Cloud features are used today:

- SAP HANA smart data integration (SDI), which is used in connections with *Data Provisioning* option *Data Provisioning Agent*
- SAP HANA smart data access (SDA), which is used in connections with no *Data Provisioning* option or *Data Provisioning* option = *Cloud Connection*.

For more information on these adapters, see [Connecting SAP HANA Cloud, SAP HANA Database to Remote Data Sources](#)

When you import an object through any of these methods, it is added to the repository and is available for use by any user assigned to the space in:

- The *Data Builder* start page.
- The *Repository* tab of the *Source Browser* from which it can be added as a source for views and other objects (see [Add Objects from the Repository \[page 292\]](#)).


5.2.1 Import Remote Tables

Import remote tables from a connection into your space directly from the Data Builder start page or the Repository Explorer.

Context

You can import tables from any valid connection that supports the remote tables feature (except partner connections). You can also import tables via the connection type SAP BW bridge (only available within a SAP BW bridge space).

Procedure

1. From the side navigation, choose *Data Builder* and select a space if necessary.
2. Choose  *Import Remote Tables*. All the available connections are listed.
3. Select the connection. Choose *Next Step*.
4. Select the objects you would like to import.
5. Choose *Next Step*. You get an overview of the objects that you will import. If an object has already been imported, it is listed on the tab *Already in the Repository* and will not be imported again.
6. For the objects to be imported, you can change the technical name and the business name. Choose *Import and Deploy*.
7. The import status is displayed. Choose *Close* to close the wizard.

Results

In the list of files in the Data Builder, the imported remote tables will be displayed.

5.2.2 Review and Edit Imported Table Properties

Provide business-friendly names for your table and its columns, identify its semantic usage, enable replication and memory storage, and set other properties.

Procedure

1. Open your imported table in the table editor and review the properties in the *General* section:

Property	Description
Business Name	Enter a descriptive name to help users identify the object. This name can be changed at any time.
Technical Name	[read-only] The technical names of remote tables cannot be modified.
Semantic Usage	<p>Select the way your entity should be used for data modeling purposes.</p> <p>Choose from the following:</p> <ul style="list-style-type: none"> • <i>Fact</i> - Contains one or more measures and attributes. A fact typically has associations pointing to one or more dimensions and is consumed by analytic models (see Creating a Fact [page 305]). • <i>Dimension</i> - Contains attributes containing master data like a product list or store directory, and supporting hierarchies (see Creating a Dimension [page 314]). • <i>Hierarchy</i> - Contains attributes defining a parent-child hierarchy (see Creating an External Hierarchy [page 330]). • <i>Hierarchy with Directory</i> - Contains one or more parent-child hierarchies (see Creating a Hierarchy with Directory [page 331]). • <i>Text</i> - Contains attributes used to provide textual content in one or more languages (see Create a Text Entity for Attribute Translation [page 327]). • <i>Relational Dataset</i> - [default] Contains columns with no specific analytical purpose. • <i>Analytical Dataset (Deprecated)</i> - Use <i>Fact</i> instead (see Analytical Datasets (Deprecated) [page 351]).

2. [remote tables] Review the properties in the [Remote](#) section:

Note

To enable replication, use the tools at the top of this section (see [Replicate Remote Table Data \[page 98\]](#)).

Replication is not available if the connection of your remote table is configured as data access: [Remote Only](#).

Property	Description
<i>Connection (Business Name) or Connection (Technical Name)</i>	<p>[read-only]</p> <p>Displays the name of the connection the remote table belongs to. Technical or Business Name is displayed, depending on how you have configured your UI settings in Your Profile Settings.</p>
Remote Table	[read-only] Displays the full path to the remote table in the source system.

Property	Description
Data Access	<p>[read-only] Displays how the remote table data is stored:</p> <ul style="list-style-type: none"> • <i>Remote</i>: Data is accessed directly from the source (federation) and read from the virtual table. • <i>Replicated</i>: Data is copied in SAP Datasphere and is read from the replica table. <ul style="list-style-type: none"> • <i>Replicated (Snapshot)</i>: The data is read from the replica table but is not expected to be updated in real-time. You can create a schedule to refresh the data regularly. • <i>Replicated (Real-Time)</i>: The data is read from the replica table and expected to be updated in real-time.
Last Updated	<p>[read-only]</p> <p>Displays when the data was last successfully updated in SAP Datasphere.</p> <div style="border: 1px solid #ccc; background-color: #f0f0f0; padding: 10px; margin-top: 10px;"> <p>Note</p> <p>This section is not available if the connection of your remote table is configured as data access: <i>Remote Only</i>.</p> </div>
Frequency	<p>[read-only]</p> <p>Displays how often data is replicated. The value here shows the intended frequency that might have not been reached yet depending on the status of the remote table.</p> <ul style="list-style-type: none"> • <i>--</i>: Refresh frequency doesn't apply. • <i>None</i>: There is no schedule defined for this table. • <i>Scheduled</i>: A schedule is defined for this table. • <i>Real-Time</i>: Real-time replication has been set up for the remote table. • <i>Paused</i>: Data refresh is paused. <div style="border: 1px solid #ccc; background-color: #f0f0f0; padding: 10px; margin-top: 10px;"> <p>Note</p> <p>This section is not available if the connection of your remote table is configured as data access: <i>Remote Only</i>.</p> </div>

Property	Description
Status	<p>[read-only]</p> <p>Displays the current status of the remote table.</p> <ul style="list-style-type: none"> • <code>---</code>: Initial state • <i>Loading</i>: Started to load a new snapshot • <i>Available</i>: Snapshot has been loaded and is available in the replica table • <i>Initializing</i>: Started to enable real-time data replication • <i>Active</i>: Data is replicated and updated in real-time • <i>Error</i>: Failed to load a new snapshot, or failed to enable real-time data replication, or data is replicated in real-time but remote exceptions occurred • <i>Paused</i>: Real-time replication is paused • <i>Disconnected</i>: SAP HANA smart data integration Data Provisioning Agent got disconnected <p>For information about how to connect the agent, see Connect and Configure the Data Provisioning Agent.</p> <div style="border: 1px solid #ccc; padding: 5px; margin-top: 10px;"> <p>Note</p> <p>This section is not available if the connection of your remote table is configured as data access: <i>Remote Only</i>.</p> </div>
Statistics Type	<p>Displays the statistics type that has been created for the remote table :</p> <ul style="list-style-type: none"> • <i>HISTOGRAM</i>: Creates a data statistics object that helps the query optimizer estimate the data distribution. • <i>RECORD COUNT</i>: Creates a data statistics object that helps the query optimizer calculate the number of records (rows) in a table data source. The <i>RECORD COUNT</i> type is a table-wide statistic. • <i>SIMPLE</i>: Creates a data statistics object that helps the query optimizer calculate basic statistics, such as min, max, null count, count, and distinct count. <div style="border: 1px solid #ccc; padding: 5px; margin-top: 10px;"> <p>Note</p> <p>Statistics can be created after you have deployed your remote table, for more information, see Creating Statistics for Your Remote Tables</p> </div>
Statistics Last Updated	<p>Displays the date when the last statistics update has been refreshed for the remote table.</p>

You can, at any time, click [Open Monitor](#) and navigate to the *Remote Tables* monitor to review details of recent replication runs (see [Replicating Data and Monitoring Remote Tables](#)), or to *Remote Table Statistics* to create or review existing statistics for the remote table (see [Creating Statistics for Your Remote Tables](#)).

Note

If the connection of your remote table source is configured as data access: *Remote Only*, you can navigate only to the *Remote Table Statistics* monitor.

3. Review the columns that are contained in your table. You can change the *Business Name* of columns and set keys here (see [Set Key Columns to Uniquely Identify Records \[page 318\]](#)).

- [remote tables] Create filter conditions to load only the data that is needed (see [Restrict Remote Table Data Loads \[page 95\]](#)).
- [imported remote and local tables] *Input Parameters*: If your remote table consumes an SAP HANA SQL view with input parameters, the parameter properties are displayed. You can define a default value for each parameter:

Input Parameters (1)

Name	Data Type	Default Value
STR42	String(10)	dd

⚠ Restriction


Parameters with data type cds.LargeBinary and parameters that start with '\$\$' are not supported.

Tables that contain input parameters require special treatment in the following situations:

- Previewing data - Accept the default value, if one is provided, or enter a value for each input parameter (see [Viewing or Previewing Data in Data Builder Objects \[page 297\]](#)).
 - Adding a table as a source in the graphical view editor - Map each input parameter in the source table to a value or an input parameter in the view (see [Add a Source \[page 217\]](#)).
 - Adding a table as a source in the SQL view editor - Complete the syntax to map each input parameter in the source table to a value or an input parameter in the new view (see [Process Source Input Parameters in an SQL View \[page 254\]](#)).
 - Adding a table as a source in the analytic model editor - Map each input parameter in the source table to a variable in the model (see [Add a Variable \[page 348\]](#)).
6. Save and Deploy

📌 Note

[remote tables connected via SAP HANA smart data integration (SDI) only]:

If a remote table definition created by an import doesn't match the remote table source definition, you won't be allowed to deploy it. You'll need to click  (*Refresh*) to repair the table structure and continue with deployment.

5.2.3 Restrict Remote Table Data Loads

Remove unnecessary columns, create filters or add columns to reduce the volume of data that is loaded in your remote table.

Prerequisites

- Your remote table must be already deployed in your space.

- Remote Tables that use an SAP HANA smart data integration ABAP or SAP HANA smart data integration CDI based connections must have been imported using a DP Agent with version equal or higher than 2.5.4.1.
- To ensure that the filters won't impact existing partitioning or statistics, the `DWC_DATAINTEGRATION.read` privilege must be assigned to your profile.

⚠ Caution

You might check the impact on the existing dependent objects when you create filters, add columns or remove columns.

Context



Loading larger data volumes can take a long time and cause high memory peaks. To optimize performance and reduce data footprint, you can load only the data you need in your remote table. SAP Datasphere offers several ways to control the volume of data that is loaded in your remote table. From the table editor of the *Data Builder*, you can remove the unnecessary columns, define a central filter to load only the data that is needed, and you can add new columns available in the source, or reinsert previously excluded columns.

Procedure

1. Remove columns from your remote table definitions:

⚠ Restriction

You can't delete key columns, and columns used in filters, partitioning, associations and hierarchies.

1. Go to *Columns* section.
2. Select the columns that you want to remove.
3. Click  (*Remove*).
4.  (*Deploy*) your remote table.

⚠ Restriction

In case you remove columns from the remote table definition compared to the source object (remote table having less columns than the source entity), real-time replications don't work for remote tables connected via SAP HANA smart data access or Cloud Connector for SAP HANA on-premise versions lower than 2.0 SPS06.


2. Creating a filter.
 1. Go to the *Filter* tab Select *Create filter* and define your filter conditions:

Property	Description
Column	<p>Select the column that will be used to filter your data.</p> <div data-bbox="703 383 1394 1249" style="border: 1px solid #ccc; padding: 10px; background-color: #f9f9f9;"> <p>Note</p> <p>Filtering on columns with the following data types is not supported:</p> <ul style="list-style-type: none"> • For SAP HANA data sources: <ul style="list-style-type: none"> • Datetime types: DATE, TIME, TIMESTAMP, SECONDDATE • Numeric types: TINYINT, SMALLINT, INT, BIGINT, DECIMAL(p,s) • Character string: NVARCHAR • For ODP data sources, only the following columns can be selected: <ul style="list-style-type: none"> • Columns which support where clause of the SELECT statement. • The length of string columns must not exceed 45 characters. Otherwise, filter values would be cut off after 45 characters. • For SAP HANA smart data integration CDI based connections, you can filter only on columns with the following data type: <ul style="list-style-type: none"> • NVARCHAR • DATE • BOOLEAN • INTEGER • BIGINT </div> <p>We recommend that you define your filters on columns that are not changing after initial creation. In case of using the advanced property "Triggers Record Primary Keys Only" in SAP HANA smart data integration based connections, filters on non-primary key columns won't work as expected.</p>
Operator	<p>The filter only supports including conditions. You can choose between <i>Equal to</i> and <i>Between</i>.</p> <div data-bbox="703 1509 1394 1664" style="border: 1px solid #ccc; padding: 10px; background-color: #f9f9f9;"> <p>Note</p> <p>Filter conditions on one column will be defined as OR condition, while filters on distinct columns are applied as AND condition</p> </div>

Property	Description
Values	You must select existing values only.

Note

- No dynamic filtering such as "last 3 years" is supported.
- If a filter on a specific column is defined, records with null values in this column will not be replicated.



2.  *(Deploy)*
3. Replicate your data

Note

- When previewing data, filters are applied.
- The task log displayed in the *Remote Tables* monitor details view will summarize the filter conditions used to load the new snapshot of data.
- If you want to change your filter conditions, you need to remove the replicated data first, as filters cannot get changed when the remote table is in data access "replicated".

Caution

You might check the impact of your defined filters on data access control filters: after the filter conditions are applied, they may have different outputs or return no results.

3. Add columns:
 1. Go to the *Columns* section.
 2. Click  *(Add)*
 3. Select the columns to add and click *OK*. You can add new columns available from the data source, or re-add previously excluded columns.
 4.  *(Deploy)*
 5. Replicate your data.

5.2.4 Replicate Remote Table Data

By default, when you import a remote table, its data is not replicated and must be accessed via federation each time from the remote system. You can improve performance by replicating the data to SAP Datasphere and you can schedule regular updates (or, for many connection types, enable real-time replication) to keep the data fresh and up-to-date.

Context

Procedure

1. Open your remote table in the table editor and scroll to the *Remote* section.
2. Use the following tools to enable replication:

- Click *Table Replication* to:
 - *Start Data Replication*
Directly start a copy of the full set of data from the source in the background.

Note

If you start data replication on a remote table whose data access is *Replicated (Real-time)*, you'll remove the real-time replication for this table. Your data will no longer be updated in real-time and you'll have to start a new data replication or create a schedule to get your data updated.

For more information, see [Replicate Full Set of Data](#).

- *Remove Replicated Data*
Stop replication and delete data from replica table.

Note

If you need to remove the replicated data for a remote table whose data access is *Replicated (Real-time)*, you must ensure that the following requirements are met to allow a proper deletion:

- The data provisioning agent is not disconnected.
- The real-time replication is not paused at connection level and is working properly.

If one of the requirement is not met, some objects won't be deleted and you'll have to delete them manually. See [3307334](#) for more information.

- *Enable Real-Time Data Replication*
Start replication of data changes in the source in real-time.

Note

When enabling the replication in real-time, the data is updated in real-time. You don't need to start a new data replication or to create a schedule to update your data


For more information, see [Replicate Data Changes in Real-Time](#).

- Click *Schedule Replication* to:
 - *Create Schedule*
Create a schedule to start a data replication in the background according to the settings defined in the schedule.

Note

If you create a schedule for a remote table whose data access is *Replicated (Real-time)*, the replication type will change from real-time replication to batch replication at the next run of the schedule. The data will no longer be updated in real-time

For more information, see [Scheduling Data Integration Tasks](#).

- [Edit Schedule](#)
Change how the schedule is specified, or change the owner of the schedule.
For more information, see [Take Over the Ownership of a Schedule](#).
 - [Delete Schedule](#)
Delete the schedule if required.
3. [optional] You may need to click the [Refresh](#) tool to refresh the properties in the [Remote](#) section see [Process Source Changes in the Table Editor \[page 100\]](#). You can, at any time, click  ([Open Monitor](#)) and, either navigate to the [Remote Tables](#) monitor to review details of recent replication runs (see [Replicating Data and Monitoring Remote Tables](#)), or navigate to [Remote Table Statistics](#) to create or review existing statistics for the remote table (see [Creating Statistics for Your Remote Tables](#)).

5.2.5 Accelerate Table Data Access with In-Memory Storage

By default, table data is stored on disk. You can improve performance by enabling in-memory storage.

Procedure

1. Open your table in the table editor and scroll to the [Table Services](#) section.
2. Enable the following option:

Property	Description
Store Table Data in Memory	Enable this option to store the table data directly in memory (hot store). By default, table data is stored on disk (warm store).

3. Click  ([Deploy](#)) to deploy the table and move its data into in-memory storage.

5.2.6 Process Source Changes in the Table Editor

Identify available table structure updates in your data sources and resolve conflicting errors resulting from these updates. This operation applies to any remote tables as well as local tables from an Open SQL Schema.

Prerequisites:

- Your table data is imported via an Open SQL Schema or from an HDI container, and is already saved and deployed in your space.
- Or it's a remote table connected via an SAP HANA smart data integration or SAP HANA smart data access.

Note


For SAP HANA smart data access, the remote table must be connected from one of the supported connection types:

- SAP HANA Cloud via ODBC. For more information, see [Connect to the SAP HANA Database in SAP HANA Cloud via ODBC](#).

- SAP S/4HANA Cloud and Cloud Data Integration. For more information, see [SAP S/4HANA Cloud and Cloud Data Integration](#).
- SAP ABAP. For more information, see [SAP ABAP](#).

⚠ Restriction

In case you remove columns from the remote table definition compared to the source object (remote table having less columns than the source entity), real-time replications don't work for remote tables connected via SAP HANA smart data access or Cloud Connector for SAP HANA on-premise versions lower than 2.0 SPS06.

Keeping your data up-to-date can sometimes be a challenge for modelers. When an update is available in your data source, you can do a refresh of your table structure in the *Data Builder* or click  [Validate Remote Tables](#) from the *Data Builder* landing page. For more information, see [Process Source Changes for Several Remote Tables \[page 102\]](#).


ℹ Note

This refresh is a manual action from the *Table Editor*. When you click [Refresh](#), you will receive a notification of any structural changes in the remote source and can then decide whether to proceed and import the changes or cancel.

⚠ Caution

Refreshing table structures will affect dependent objects. But you'll be notified about the changes and inconsistencies that can result because of these changes.

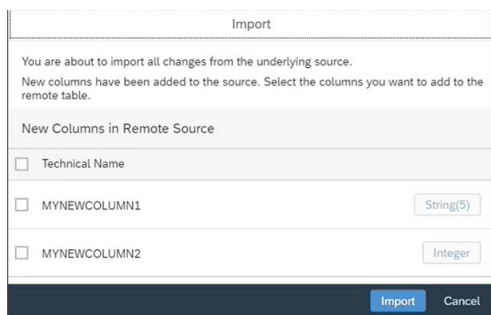
To import changes from the underlying sources,

1. Click  ([Refresh](#)) to import available changes.

ℹ Note

Changes in data type, length, or precision are also automatically updated or aligned with the source table structure for either SAP HANA smart data access or SAP HANA smart data integration remote connections.



If new columns exist in the data source, a windows allows you to add those you need:



ℹ Note

- You must add new key columns.



- Double check columns that contain spatial data type as remote tables don't support this data type: The spatial reference system might be wrong. For more information on spatial reference system, see [Spatial Reference Systems \(SRS\) and Spatial Reference Identifiers \(SRID\)](#)
- Columns can be added or removed at any time from the column section. For more information, see [Restrict Remote Table Data Loads \[page 95\]](#).

2. Once import is completed, you can see the changes in the validation area  ([Validation Messages](#)).
3. Click  ([Preview Data](#)) to preview the data contained in your remote table.

Note

Previewing data on a remote system is similar to previewing data of a local table. However, it might take longer, it can impact the remote system's performance, and some preview controls aren't available. You can replicate the table locally to address performance issues and have a richer preview experience. For more information on previewing data, see [Viewing or Previewing Data in Data Builder Objects \[page 297\]](#). Moreover, the remote table data preview is limited for the following connection types:

- *ABAP* connection: the filters = and IN only are available.
- *Cloud Data Integration (CDI)* connection: rows won't be counted in headers.

4.  ([Save](#)) Save your changes.
5.  ([Deploy](#)). A pop up informs you about the source updates before you can go forward with deployment and access data from the updated remote source.

Caution

If your remote table is replicated, you can refresh, import, preview, and save the changes as stated above. However, once import of changes is completed, you can't directly redeploy your table without first going back to the [Remote Tables](#) monitor and removing the replicated data. Once replicated data is removed, you'll be able to redeploy your table with its updated structure. For more information on removing replicated data, see [Replicating Data and Monitoring Remote Tables](#).

5.2.7 Process Source Changes for Several Remote Tables

Identify available table structure updates for all tables sharing the same source connection, and avoid errors and impact on dependent objects and runtimes in SAP Datasphere resulting from these updates, such as view runs, remote table replications or deployment.

Context


Before you can process source changes for several remote table, you must fulfill the following prerequisites:

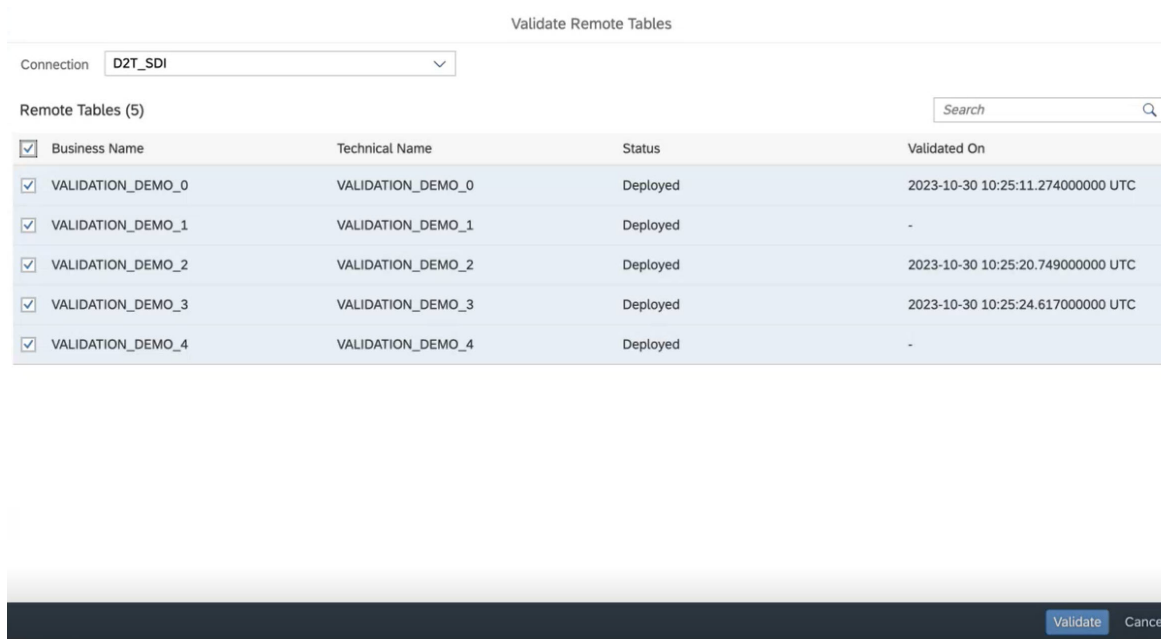
- The remote tables are already saved in your space.
- They are connected via an SAP HANA smart data integration or SAP HANA smart data access.

When changes are made in source models, they might not be reflected immediately in SAP Datasphere. This can result in errors and impact on dependent objects, and runtimes, and you need to do a refresh to get these updates in your remote table definition.

To identify source changes for several remote tables sharing the same connection, you can either proceed from the [Repository Explorer](#) or from the [Data Builder](#) landing page:

Procedure

1. Click  [Validate Remote Tables](#).
2. From the windows [Validate Remote Tables](#), select the relevant source connection and then the remote tables you want to check, and click [Validate](#):



Validate Remote Tables

Connection:

Remote Tables (5)

<input checked="" type="checkbox"/>	Business Name	Technical Name	Status	Validated On
<input checked="" type="checkbox"/>	VALIDATION_DEMO_0	VALIDATION_DEMO_0	Deployed	2023-10-30 10:25:11.274000000 UTC
<input checked="" type="checkbox"/>	VALIDATION_DEMO_1	VALIDATION_DEMO_1	Deployed	-
<input checked="" type="checkbox"/>	VALIDATION_DEMO_2	VALIDATION_DEMO_2	Deployed	2023-10-30 10:25:20.749000000 UTC
<input checked="" type="checkbox"/>	VALIDATION_DEMO_3	VALIDATION_DEMO_3	Deployed	2023-10-30 10:25:24.617000000 UTC
<input checked="" type="checkbox"/>	VALIDATION_DEMO_4	VALIDATION_DEMO_4	Deployed	-

Note

From the [Repository Explorer](#), if you have remote tables in several spaces, you will be prompted to select a space.

SAP Datasphere will compare the source and the target remote tables, and will refresh the status: Remote tables with incompatible changes will get the status [Runtime Error](#) (for example a column has been removed), whereas remote tables with compatible changes will get the status [Changes to Deploy](#) (for example a non-key column has been added). Remote tables that are not deployed at this time and have any changes in the source will get the status [Design Time Error](#).

3. To apply the changes, open the remote table in the table editor.
The editor automatically opens a window to allow you to proceed with the changes:

Refresh

The source has been modified in the remote system. Column deletions and modifications must be applied.

You can choose to add any or all of the new columns to the remote table definition. By default, only new key columns will be added.

New Columns in Source Table

Technical Name

E Decimal(9, 4)

Apply
Cancel

The object status set by *Validate Remote Tables* is only an indicator that a certain remote table requires action in the *Table Editor*. As there may be a delay between the *Validate* action and the time you enter into *Table Editor* screen, it is required to proceed with the *Refresh* action in the *Table Editor* to integrate the actual source changes in your remote table.

4. Select the changes you want to apply to your remote table.
5. Redeploy the remote table.

5.2.8 Modify or Duplicate Remote Tables

From the remote table editor, you can change the remote table connection and/or the remote table source object. You can also create a copy of an existing remote table.

In some cases, you might need to change the connection of your remote table, or to change the remote table source object. For example, if for performance reasons you have decided to split one connection into several ones, or you have decided to move on remote table from one connection to another. It might be also helpful to create a copy of an existing remote table, if you made changes in your remote table but want to keep the 2 versions of your remote table, or you want to have several identical remote tables with different filter conditions. Note that for remote tables connected via SAP HANA smart data access, only one remote table per source entity of a connection can be in status *Replicated*, though.

i Note

The remote table must be in *Data Access = Remote*

From the *Data Builder*, open the remote table in the table editor.

Changing the Connection and/or the Remote Table Source Object.

In the *Remote* section of your remote table, click *Change Remote Table Source*. A wizard opens and allows you to change the connection and/or the remote table source object in 3 steps:

1. **Connections:** Change the connection. All the connections available in your space are displayed. Select the new connection and click [Next Step](#) to move to the [Tables](#) section.

Note

If you want to change only the remote table source object, you can ignore this step by reselecting the current connection and click [Next Step](#).

2. **Tables:** Change the remote table source. If you want to change the remote table source, you can do so by selecting [Change Remote Table](#). You can then browse through schemas available for the selected connection (on the right side of the screen), and their associated tables (left side of the screen).

Note

Schemas only exist for SAP HANA smart data access-based connections and SAP HANA smart data integration-based database connections.

Select the relevant schema and table you need and click [Next Step](#).

Note

If you want to change only the connection, you can ignore this step and click [Next Step](#).

3. **Review:** Review your changes and confirm. When confirming a validation is performed to check any inconsistencies in the remote table structure. You'll get information in the validation section.

Note

If the changes affect the remote source type, additional validation checks are performed, which could block the deployment. For example if you switch from an SAP HANA smart data integration connection into an SAP HANA smart data access connection:

- You must remove the remote table filter conditions as they are not supported by an SAP HANA smart data access connection.
- You must remove partitioning on non-key columns as they are not supported by an SAP HANA smart data access connection.

Once you have confirmed your changes, a refresh is automatically done and you can see the changes in the remote table editor. You can then save your changes and redeploy the remote table.

Creating a Copy of an Existing Remote Table

Copying your remote table can be helpful in some cases. For example if you made changes in your remote table but want to keep the 2 versions of your remote table. Or you want to have several identical remote tables with different filter condition. Then you can use the Save as option. This option allows you to create a copy of an existing table with a new technical name.

From the [Data Builder](#), open the relevant remote table in the table editor. Select [Save As](#).

Note

For remote tables connected via SAP HANA smart data access, only one remote table per source entity of one connection can be in status [Replicated](#) (Snapshot or Real-Time). For more information, see [Replicating Data and Monitoring Remote Tables](#).

5.3 Creating a Local Table


Create a table and define columns to receive data from a flow or a CSV file. You can also import tables from a connection or a CSN file.

Context

This procedure explains how to create an empty table by defining its columns. You can, alternatively:

- Create a table by importing a CSV file (see [Creating a Local Table from a CSV File \[page 122\]](#)).
- Create a table as the output of a flow (see [Add or Create a Target Table \[page 143\]](#)).
- Import a table from:
 - A connection or other source in the *Sources* tab of the *Source Browser* in any of the data builder editors (see [Import an Object from a Connection or Other Source \[page 294\]](#)).
 - A CSN file via the *Import* menu in the data builder start page or an ER model (see [Importing Objects from a CSN/JSON File \[page 49\]](#)).

Procedure

1. In the side navigation area, click  (*Data Builder*), select a space if necessary, and click *New Table* to open the editor.
2. Enter the following properties as appropriate:

Property	Description
Business Name	Enter a descriptive name to help users identify the object. This name can be changed at any time.
Technical Name	Displays the name used in scripts and code, synchronized by default with the <i>Business Name</i> . To override the default technical name, enter a new one in the field. Technical names can contain only alphanumeric characters and underscores.

Note

Once the object is saved, the technical name can no longer be modified.

Property	Description
Package	<p>Select the package to which the object belongs.</p> <p>Packages are used to group related objects in order to facilitate their transport between tenants.</p> <div style="border: 1px solid #ccc; background-color: #f9f9f9; padding: 10px; margin-top: 10px;"> <p>Note</p> <p>Once a package is selected, it cannot be changed here. Only a user with the DW Space Administrator role (or equivalent privileges) can modify a package assignment in the <i>Packages</i> editor.</p> </div> <p>For more information, see Creating Packages to Export.</p>
Semantic Usage	<p>Select the way your entity should be used for data modeling purposes.</p> <p>Choose from the following:</p> <ul style="list-style-type: none"> • <i>Fact</i> - Contains one or more measures and attributes. A fact typically has associations pointing to one or more dimensions and is consumed by analytic models (see Creating a Fact [page 305]). • <i>Dimension</i> - Contains attributes containing master data like a product list or store directory, and supporting hierarchies (see Creating a Dimension [page 314]). • <i>Hierarchy</i> - Contains attributes defining a parent-child hierarchy (see Creating an External Hierarchy [page 330]). • <i>Hierarchy with Directory</i> - Contains one or more parent-child hierarchies (see Creating a Hierarchy with Directory [page 331]). • <i>Text</i> - Contains attributes used to provide textual content in one or more languages (see Create a Text Entity for Attribute Translation [page 327]). • <i>Relational Dataset</i> - [default] Contains columns with no specific analytical purpose. • <i>Analytical Dataset (Deprecated)</i> - Use <i>Fact</i> instead (see Analytical Datasets (Deprecated) [page 351]).
Delta Capture	<p>Track the delta changes that are made in the table adding 2 delta capture columns: <code>Change Type</code> and <code>Change Date</code>. See Capturing Delta Changes in Your Local Table [page 118]</p>
Source Object (Open SQL Schema/HDI Container)	<p>[read-only] Displays the technical name of the Open SQL Schema or HDI Container and the object name.</p>
Status	<p>[read-only] Displays the deployment and error status of the object.</p> <p>For more information, see Saving and Deploying Objects [page 39].</p>






3. Based on the *Semantic Usage* of your entity, review and modify its *Columns*, *Attributes*, and/or *Measures*:

- *Fact* - Review the lists of measures and attributes (see [Creating a Fact \[page 305\]](#)).
- *Dimension* - Review the list of attributes (see [Creating a Dimension \[page 314\]](#)).
- *Hierarchy* - Define the parent and child columns (see [Creating an External Hierarchy \[page 330\]](#)).
- *Hierarchy with Directory* - Define all the necessary attributes and settings (see [Creating a Hierarchy with Directory \[page 331\]](#)).
- *Text* - Review the list of attributes (see [Create a Text Entity for Attribute Translation \[page 327\]](#)).
- *Relational Dataset* - Review the list of columns (see [Columns \[page 108\]](#)).

4. Complete or consult other sections as appropriate:
 - *Associations* - Create associations to other entities (see [Create an Association \[page 234\]](#)).
 - *Business Purpose* - Provide a description, purpose, contacts, and tags to help other users understand your entity.
 - *Table Services* - Enable the *Memory Storage* option to store the table data directly in memory (see [Accelerate Table Data Access with In-Memory Storage \[page 100\]](#)).

Note

If the connection of your remote table source is configured as data access: *Remote Only*, you can navigate only to the *Remote Table Statistics* monitor.

- *Partitions*- Define partitions for your local table. For more information, see [Partitioning Local Tables \[page 117\]](#).
 - *Dependent Objects*- If your entity is used as a source or as association target for other entities, then they are listed here. For more information, see [Review the Objects That Depend on Your Table or View \[page 43\]](#).
5. [optional] Click  (*Edit Custom CSN Annotations*) to open the *Edit Custom CSN Annotations* dialog. For more information, see [Edit a Custom CSN Annotation \[page 249\]](#)
 6. Click  (*Save*)   to save your entity or click  (*Deploy*) to save and deploy it immediately. For more information, see [Saving and Deploying Objects \[page 39\]](#).
 7. Once your table is deployed, you can:
 - Import data from a CSV file or delete all table data (see [Load or Delete Local Table Data \[page 113\]](#)).
 - Manually add, edit, duplicate, or delete individual records (see [Maintain Local Table Data \[page 115\]](#)).
 - Understand the lineage and impacts of the table (see [Impact and Lineage Analysis \[page 34\]](#)).

5.3.1 Columns

Columns appear in tables and views with a *Semantic Usage* of *Relational Dataset*.

Columns are displayed in the *Columns* section of tables and views.

Note

In the graphical view and sql view editors, you can click the *Edit Columns* button in the *Columns* list to open it in a dialog.

Columns have the following properties:

Property	Description
Key	<p>Select the checkbox to specify that the column is a primary key column. Key columns cannot be null.</p> <div style="border: 1px solid #ccc; background-color: #f9f9f9; padding: 10px; margin-top: 10px;"> <p>Note</p> <p>To set or remove an attribute as a primary key column in a view editor side panel, hover over it and click ⋮ (Menu) >> Set as Key or Remove as Key.</p> </div>
Business Name	<p>Enter a descriptive name to help users identify the object. This name can be changed at any time.</p>
Technical Name	<p>Displays the name used in scripts and code, synchronized by default with the <i>Business Name</i>.</p> <p>To override the default technical name, enter a new one in the field. Technical names can contain only alphanumeric characters and underscores.</p> <div style="border: 1px solid #ccc; background-color: #f9f9f9; padding: 10px; margin-top: 10px;"> <p>Note</p> <p>Once the object is saved, the technical name can no longer be modified.</p> </div>
Data Type	<p>Select the type of data that the column will contain.</p> <p>You can change the data type of a column. For more information, see Modifying Objects That Have Dependent Objects [page 41]</p> <p>For a list of available data types and their supported data type conversions, see Column Data Types [page 110].</p>
Default Value	<p>Enter an appropriate default value for the column.</p> <div style="border: 1px solid #ccc; background-color: #f9f9f9; padding: 10px; margin-top: 10px;"> <p>Note</p> <p>This property is only displayed in the table editor.</p> </div>
Not Null	<p>Select this option to indicate that the column must contain a value.</p> <div style="border: 1px solid #ccc; background-color: #f9f9f9; padding: 10px; margin-top: 10px;"> <p>Note</p> <p>This property is only displayed in the table editor.</p> </div>
Change Type	<p>This column will track the type of last change made to a record. When a record is inserted or updated corresponding change types are used (for example "I" or "U"). When an existing record is deleted other specific change types are used (for example "D"). Note that deleting a record will not physically delete it, so that the changes can be propagated to the different objects that consume it in delta mode. It is however filtered out when accessing the Local Table (using the Active Records Table). Also, note that the change types provided by the different SAP Datasphere apps vary and may depend on the actual source that is connected. The handling of the different change types is implemented internally by SAP Datasphere apps that consume the Delta Capture Table with no need for consideration in modeling. For more information on records deletion, see Load or Delete Local Table Data [page 113].</p>

Property	Description
Change Date	The column will track the last date and time of the last change to an individual record. Current UTC timestamp.

5.3.2 Column Data Types

SAP Datasphere supports a wide range of data types for your columns. Once a table is deployed and contains data, column data types can only be changed if the change is compatible with the type of data contained in the column and will not truncate any value.

Note

The SAP Cloud Application Programming Model (CAP) recommends to use the Core Data Services built-in types (see [Built-in Types](#)), but it is also possible to use the SAP HANA-specific data types, which all have a "hana." prefix (see [SAP HANA-Specific Data Types](#)).

Datetime Data Types

Data Type	Default Format	Supported Value Range	Can Be Converted To...
Date	YYYY-MM-DD	'0001-01-01' to '9999-12-31'	Datetime, String
Datetime	YYYY-MM-DD HH24:MI:SS	'0001-01-01 00:00:00' to '9999-12-31 23:59:59'	Date, String, Time
Time	HH24:MI:SS	'00:00:00' to '23:59:59'	Datetime, String
Timestamp	YYYY-MM-DD HH24:MI:SS	'0001-01-01 00:00:00' to '9999-12-31 23:59:59'	String

Character String Data Types

Data Type	Can contain...	Can Be Converted To...
LargeString	Variable length string of up to 2 GB.	No data type conversion possible
String(n)	Variable-length Unicode string of up 5000 characters.	Binary, Boolean, Date, Datetime, Decimal, Double, hana.SMALLINT, hana.TINYINT, hana.SMALLDECIMAL, hana.REAL, hana.BINARY, Integer, Integer64, LargeBinary, LargeString, Time, Timestamp.

Data Type	Can contain...	Can Be Converted To...
<div style="border: 1px solid #ccc; padding: 5px; background-color: #f9f9f9;"> <p>Note</p> <p><code>String(36)</code> only can also be converted to UUID.</p> </div>		

Numeric Data Types

Data Type	Description	Example Value	Can Be Converted To...
Decimal (p, s)	<p>Precision (p) defines the number of total digits and can be between 1 and 38.</p> <p>Scale (s) defines the number of digits after the decimal point and can be between 0 and p.</p>	15.2	Double, String
DecimalFloat	Decimal floating-point number with 34 mantissa digits.	15.2	
Double	Double-precision, 64-bit floating-point number	15.2	String
Integer	Respective container's standard signed integer.	1337	Boolean, Decimal, Double, hana.SMALLINT, hana.TINYINT, Integer64, String
Integer64	Signed 64-bit integer	1337	Boolean, Decimal, Double, hana.SMALLINT, hana.TINYINT, Integer, String
hana.REAL	32-bit binary floating-point number	15.2	Double, hana.SMALLDECIMAL, String
hana.SMALLDECIMAL	64-bit decimal floating-point number, where (p) can be between 1 and 16 and s can be between -369 and 368.	15.2	Double, hana.REAL, String
hana.SMALLINT	Signed 16-bit integer supporting the values -32,768 to 32,767	15	Decimal, Double, hana.SMALLDECIMAL, hana.TINYINT, Integer,

Data Type	Description	Example Value	Can Be Converted To...
			Integer64, hana.REAL, String
hana.TINYINT	Unsigned 8-bit integer supporting the values 0 to 255	15	Decimal, Double, hana.SMALLDECIMAL, hana.SMALLINT, hana.REAL, Integer, Integer64, String

For more information, see [Numeric Data Types](#) in the *SAP HANA Cloud, SAP HANA Database* documentation.

Boolean Data Types

Data Type	Can contain...	Can Be Converted To...
Boolean	TRUE, FALSE and UNKNOWN, where UNKNOWN is a synonym of NULL.	Integer, String

Binary Data Types

Data Type	Can contain...	Can Be Converted To...
Binary(n)	Variable length byte string with user-defined length limit of up to 4000 bytes.	String, LargeBinary, hana.BINARY
LargeBinary	Variable length byte string of up to 2 GB.	No data type conversion possible
hana.BINARY (n)	Byte string of fixed length (n).	LargeBinary, String, Binary

Universally Unique Identifier Data Type

Data Type	Description	Example Value	Can Be Converted To...
UUID	Universally unique identifier encoded as a 128-bit integer.	be071623-8699-4106-b6fa-8e3cb04c261e	String(n) where n >=36

Spatial Data Types

Data Type	Can contain...	Can Be Converted To...
hana.ST_GEOMETRY[(<i><srid></i>)]	Spatial data in any form, including 0-dimensional points, lines, multi-lines, and polygons.	No data type conversion possible
hana.ST_POINT[(<i><srid></i>)]	Spatial data in the form of 0-dimensional points that represents a single location in coordinate space.	No data type conversion possible

Note

Columns with data types `hana.ST_POINT[(<srid>)]` and `hana.ST_GEOMETRY[(<srid>)]` cannot be consumed directly in SAP Analytics Cloud in a view with a semantic usage of *Fact*. To use these geo-coordinates columns in SAP Analytics Cloud, add them to a view with a semantic usage of *Dimension*, and then create an association from your *Fact* to the *Dimension*.

5.3.3 Load or Delete Local Table Data

You can upload data from a CSV file to a local table. You can also delete all data from the table or in case of table with delta capture enabled, delete records that have already fully processed..

Loading Local Table Data

From the toolbar, select  *Edit* >  (*Upload Data From CSV File*).

Note

You must have the standard *DW Modeler* role to use this tool. For more information, see [Roles and Privileges by App and Feature](#)

Select your CSV file and follow the instructions.

Note

The file extension must be *.csv. The file size must not exceed 25MB.

The following options are available:

- Select the *Delete Existing Data Before Upload* option if appropriate.
- Select *Use first row as column header* if your file contains column headers in its first line.
- Next to *Insert missing string value as*, select *Empty value* or *NULL* before adding a record. All new empty values are stored as the selected value.

- Select the character used to signify the boundary between columns in *CSV Delimiter*. In general, the default value **Auto-detect** is sufficient.

Review the matching of columns in your table with those in your CSV file.

Note

This data upload does not support the data transformations available when creating a table from a CSV file (see [Creating a Local Table from a CSV File \[page 122\]](#)).

The data in your CSV file must match the table structure, including respecting all data types and structures.

Date and time format support in this data upload is restricted as follows:

- Date columns: YYYY-MM-DD, YYYY/MM/DD, YYYY/MM-DD, YYYY-MM/DD, or YYYYMMDD
- Time columns: HH:MI:SS or HH24:MI[:SS]
- Date time columns: YYYY-MM-DD HH:MI:SS

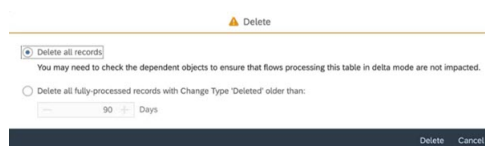
Deleting Local Table Data

From the toolbar, select  (*Delete Data From Table*).

Caution

To delete data, you must have the role *DW Integrator* or *DW Administrator*.

Depending if your local table is delta enabled or not, you will have different options. In the case of a table that is not delta enabled, you can delete all data. In case your table is delta capture enabled, you can choose between 2 options: delete all records or delete all fully-processed records with change type "Deleted".



Delete All Records

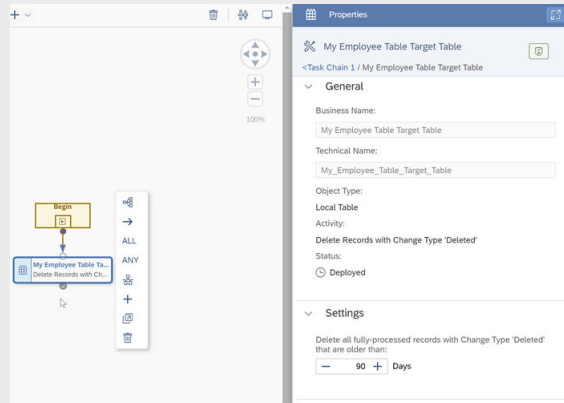
This option allows you to delete all records from the table, no matter if they are used by other apps or not. It's up to you to check if the data you are about to delete have dependencies.

Delete All Fully-Processed Records with Change Type "Deleted"

If your table is delta capture enabled, when a record is deleted, it's marked as deleted so that it won't be visible in consuming (view) models, but it's not physically deleted from the database. Indeed such deletion records from local tables with delta capture are considered by flows that are using the load type *Initial and Delta*. The records marked as deleted can't be physically deleted until they have been processed by these flows. For more information, see [Capturing Delta Changes in Your Local Table \[page 118\]](#). Once this is given, selecting this option allows you to safely delete records already fully-processed. You also define a retention period.

Note

You can automate the deletion of those records using a scheduling option within a task chain.



For more information, see [Creating a Task Chain \[page 197\]](#).

5.3.4 Maintain Local Table Data

Maintain data stored in a local table in the table editor by clicking the *Data Editor* button. You can add, edit, duplicate, and delete records.

Context

Note

Users with the standard *DW Modeler* role can edit data in this screen. For more information, see [Roles and Privileges by App and Feature](#).

You can maintain data in local tables that:

- Have a status *Deployed*.
- Have at least one key column. Tables with binary key columns can't be updated in the *Data Editor*.

Note

You cannot edit the data in the generated time table (see [Create Time Data and Dimensions](#)).







Procedure

1. In the *Data Builder Entry Page*, click a local table to open your table's data for editing.
2. Click *Data Editor*.

→ Tip

Directly open an object in its data editor by pasting its URL in your browser's address bar.

You can sort, reorder, filter, or replace your data:

- **Reorder** - Drag and drop your columns to reorder them.
- **Sort** - Click on the column header and click  (*Sort Ascending*) or  (*Sort Descending*).
- **Filter** - Click on the column header and click  (*Filter*) to open the *Define Filter* dialog. Click  (*Add Filter*), select a column to filter on, a filtering option, and a value. You can apply several filters.
- **Find and Replace** - Click on the column header and click  (*Find and Replace*) to open the *Find and Replace* dialog. You can find and replace string values that aren't key columns or delta table columns. Enter a string value or select *Empty* or *Null* values in the *Find* and *Replace* fields. Check *Match Case* or *Match Entire Field* in the *Settings* sections. (optional) Toggle on the occurrence count.
- **Control the previewed columns** - Click  (*Columns Settings*) and select the columns to show in the preview.

ⓘ Note

- The values of a hana.REAL column are automatically overwritten when their format is invalid.

Example

The value **1.1** is incorrectly formatted and will be overwritten as **1.100000023841858**.

This will cause issues if this column is used as a calculated column. In that case, avoid conversion by setting the column to another data type.

- Opening the data editor isn't be possible if:
 - A key column has its data type set to *cds.Binary* or *cds.hana.BINARY*.
 - A binary column has records appearing as *empty* and *empty* records set as *NULL*.
 - A *Not Null* column has its data type is set to one of the following types:
 - hana.BINARY
 - Binary
 - LargeBinary
 - abap.geom_ewkb
 - hana.ST_POINT
 - hana.ST_GEOMETRY

3. Update the table data as appropriate. You can:

- **Add a record** - Click *Add* to create a blank record and fill the cells as appropriate.
- **Edit a record** - Click in the cell you want to edit and enter a new value.

ⓘ Note

Columns with the data types Binary, hana.BINARY, LargeBinary, hana.ST_GEOMETRY, and hana.ST_POINT, are read-only and can't be updated.

- **Duplicate a record** - Select one or more records to duplicate and click *Duplicate*. Update cells as appropriate. You must enter a new, unique value in key columns.
- **Delete a record** - Select one or more records and click *Delete*. For local table with delta capture enabled, the record is not physically deleted but only marked as "Deleted".


⚠ Caution

To physically delete a record, you also need the role *DW Integrator* or *DW Administrator*. For more information, see [Load or Delete Local Table Data \[page 113\]](#) and [Capturing Delta Changes in Your Local Table \[page 118\]](#)

- **Restore last saved values** - Click *Restore*. All unsaved changes will be lost.
- **Define a default value for empty strings** - Next to *Insert missing string value as*, select *Empty value* or *NULL* before adding a record. All new empty string cells are automatically filled in with the selected value.

📌 Note

- *Add* and *Duplicate* tools are restricted when a Key column or a Not Null column without a set default value are hidden. Show these columns to enable all tools.
- *Filter*, *Sort*, *Delete*, and *Column Settings* tools are disabled by unsaved changes. Save your changes to enable them.
- When invalid values are added to or duplicated in the table, a red or yellow frame appears to let you know where changes are required. Frames appear when:
 - An invalid value is added.
 - A key value is duplicated and must be edited to be unique.
 - The value's format is invalid.
 - A field cannot be left empty.

Click  (*Validation Messages*) on the top right corner of your screen to get guidance on how to solve your issue.

⚠ Caution

If you have enabled *Delta Capture* for your local table, changes are tracked with 2 columns, *Change Type* and *Change Date*. Deleting a record will not physically delete it, but will set the change type to "D" (Delete). Therefore, you can't add a new record with a key that already exists, even if the record has the change type set to "D". For more information, see [Capturing Delta Changes in Your Local Table \[page 118\]](#)

4. Click  (*Save*) to save your changes and click *Table Editor* to close the *Data Editor* and return to the *Table Editor*.

5.3.5 Partitioning Local Tables

Create partitions based on one column of your local table to break your data down into smaller tables, and better manage tables with large volume of data.

Working with large volume of data can cause memory shortage and take many system resources. One of the solution is to use partitions: You create partitions based on one column of your table to break your data down into smaller and more manageable parts. When a table is partitioned, the split is done in such a way that each partition contains a different set of rows of the table, depending on the range definition you have set.

⚠ Restriction

- If you have defined key columns for your table, only one of these key columns can be used as partitioning column.
- Supported data types include:
 - String
 - Integer, Integer 64, Decimal, hana.SMALLINT, hana.TINYINT
 - Date, DateTime, Time, Timestamp
 - Binary

For more information, see [Partitioning Limits](#) in the *SAP HANA Cloud, SAP HANA Database* documentation.

- Your table must not contain data yet.

To create partitions for local table, go to [Data Builder](#), create your local table (for more information, see [Creating a Local Table \[page 106\]](#)), and then go to the [Partitions](#) tab.

Click [Define Partitions](#) and define the partitions ranges:

Property	Description
Attribute/Column	Select the column from which the partition will be defined. Note If you have defined key columns, you can select only a key column. If you have no key columns defined, all columns with compatible data types can be selected. Note that you won't be able to define a key column later, until the partitions are in place.
Partitions	Create the number of desired partitions by entering a range for each of them. Note that ranges can't overlap. Note If you have data outside the defined partition ranges, an Others partition will be created to store this data

Deploy the table.

ⓘ Note

Once deployed, you won't be able to change the data type of the primary key that has been used for partitioning. But you can still change or delete the partitions until you have added data in your table.

5.3.6 Capturing Delta Changes in Your Local Table

Track the changes that will be made later on your local table after you have deployed it.

Local tables can be used as source data or target data by SAP Datasphere apps. For some business scenarios, you might need to keep an eye on changes that will be made after you have deployed your local table. For

example, when you import a new csv file, or while running a replication flow, you might want to know which data is updated or deleted. When creating a local table, you can switch on a toggle that will capture the future updates made in your table.

Switching Delta Capture On

⚠ Caution

You need to meet the following requirements to add the setting *Delta Capture*:

- Your table must not be deployed yet. After deployment, it is not possible to turn an existing local table into a table that allows *Delta Capture*.
- You must define at least one key column.

You enable *Delta Capture* while creating a local table. See [Creating a Local Table \[page 106\]](#).

When *Delta Capture* is switched on:

- *Delta Capture Table* field is added and a default name for the delta capture table is defined (Technical name + Delta).

🔗 Example

I define the business name "My Employee Data" for my local table with delta capture enabled, the technical name is "My_Employee_Data". The default delta capture table name will be

"My_Employee_Data_Delta":

Business Name:

Technical Name: *

Semantic Usage:

Delta Capture:
 ON

Delta Capture Table:

- 2 additional columns are automatically created in my table:

Column	Description	Possible Values
Change Date	The column will track the last date and time of the last change to an individual record.	Current UTC timestamp.
Change Type	This column will track the type of last change made to a record.	This column will track the type of last change made to a record. When a record is inserted or updated corresponding change types are used

Column	Description	Possible Values
		(for example "I" or "U"). When an existing record is deleted other specific change types are used (for example "D"). Note that deleting a record will not physically delete it, so that the changes can be propagated to the different objects that consume it in delta mode. It is however filtered out when accessing the Local Table (using the Active Records Table). Also, note that the change types provided by the different SAP Datasphere apps vary and may depend on the actual source that is connected. The handling of the different change types is implemented internally by SAP Datasphere apps that consume the Delta Capture Table with no need for consideration in modeling. For more information on records deletion, see Load or Delete Local Table Data [page 113]

<input type="checkbox"/>	<input type="checkbox"/>	Business Name	Technical Name	Data Type	Default Value	Not Null
<input type="checkbox"/>	<input type="checkbox"/>	Change Type	Change_Type	String(1)	I	<input checked="" type="checkbox"/>
<input type="checkbox"/>	<input type="checkbox"/>	Change Date	Change_Date	Timestamp	CURRENT_UTCTIMESTA...	<input checked="" type="checkbox"/>
<input type="checkbox"/>	<input checked="" type="checkbox"/>	Employee ID	Employee_ID	String(100)		<input checked="" type="checkbox"/>

You can change both business name and technical name but you can't change the data type (read-only).

Note

- The delta capture columns can't be set as key column.
- They can't be deleted if the toggle is switched on.
- Once the table is deployed, the toggle can't be switched off.

Importing a File to Your Local Table With Delta Capture Enabled

You can import another csv file containing data in your table. In this case, the newly imported data will get change type as "I".

Note

You can't import data that will erase existing key records data. You first need to delete the existing data by selecting the option [Delete Existing Data Before Upload](#) when importing the new csv file.

For example, if I import a csv file with new data, the change type is set to "I":

Employee ID	Change Type	Change Date	Employee Name
100002	I	Aug 29, 2023, 11:45:50	SILVE
100039	I	Aug 29, 2023, 11:45:50	DUFOUD
100700	I	Aug 29, 2023, 11:45:50	SANCHEZ

For more information, see [Load or Delete Local Table Data \[page 113\]](#)

Deployment and Consumption of Local Table With Delta Capture

When a local table with delta capture is deployed, the following objects are created and stored in the repository:

- The table that contains the delta capture columns. The technical name ends with "Delta".
- The table that contains only the active records. It excludes both the delta capture columns and deleted records ("M" and "D"), and keeps only the active records.

Note

This table is saved in the repository, but is deployed as a view in the database.

The 2 objects are consumed differently by SAP Datasphere apps:

- Most SAP Datasphere apps consume a local table with delta capture through the *Active Records* table only. In these cases, local tables behave the same way independent of whether *Delta Capture* is set to "On" or "Off". For examples in Graphical Views, SQL Views, E/R Modeler or Business Builder.
- The following SAP Datasphere apps also interact with the Delta Capture Table that contains the delta columns:
 - *Transformation Flow*:
 - As Source, you can choose between source with "Delta Capture" or "All Active Records". See [Add a Source Table \[page 176\]](#)
 - As target, it depends of the combination of the load type used and the table type (local table with or without delta capture). See [Processing Changes to Source and Target Tables \[page 192\]](#) and [Add or Create a Target Table \[page 191\]](#)
 - *Replication Flow*: The Delta Capture Table can be used as target, see [Creating a Replication Flow \[page 155\]](#)
 - *Table Editor*:
 - Data Preview: Once deployment is completed, it show only the delta capture tables. See [Viewing or Previewing Data in Data Builder Objects \[page 297\]](#)
 - Data Maintenance: You can perform table maintenance on the delta capture table (only) once deployment is completed. See [Maintain Local Table Data \[page 115\]](#)
 - File upload: You can update the delta capture table by uploading a new csv file, after deployment is completed. See [Load or Delete Local Table Data \[page 113\]](#)

Restriction


The Delta Capture Table is an internal table whose structure can incompatibly change at any time. It is not permitted for external data access and is only consumed by the above SAP Datasphere internal apps. Using the internal delta capture columns (*Change Date* or *Change Type*) or their content directly or indirectly for external delta replication outside the Premium Outbound Integration is also not permitted. For more information, see [Premium Outbound Integration](#).

5.4 Creating a Local Table from a CSV File





Import a .csv file to create a table and fill it with the data from the file.

Context

Note

This procedure explains how to upload a CSV file to create a new local table in SAP Datasphere with the possibility to perform some data transformations during the upload. To load data into an existing local table (without transformations), use the  ([Upload Data From CSV File](#)) tool in the table editor (see [Creating a Local Table \[page 106\]](#)).

Procedure

1. In the side navigation area, click  ([Data Builder](#)), select a space if necessary, and click  [Import](#)  [Import CSV File](#)  to open the editor.
2. Click [Select Source File](#), navigate to, and select the CSV file you want to upload.

Note

The file must have the extension * .csv and contain Unicode text only. The file size must not exceed 25 MB.

3. Review the following options, and then click [Upload](#) to open your file in SAP Datasphere:

Option	Description
Use first row as column header.	Select if your file contains column headers in its first line.
CSV Delimiter	Select the character used to signify the boundary between columns. In general, the default value, Auto-detect is sufficient.

The preview page shows a data preview on the left and a side panel showing the properties of the file or of a selected column, on the right.

4. [optional] In the side panel, review the list of columns to be created. Hover over a column in the list and click [Details](#) to view and modify the properties of the column (see [Column Properties and Profiling \[page 123\]](#)).

Note

The values of a hana.REAL column are automatically converted when their format is invalid.

❖ Example

The value **1.1** will be read and written as **1.10000023841858**.

This will cause issues if this column is used as a calculated column. In that case, avoid conversion by setting the column to another data type.

5. [optional] Perform data preparation transformations to modify the list of columns to be created, to delete rows, and to modify data in cells (see [Apply Data Transforms \[page 124\]](#)).
6. When you have performed all necessary transformations and validations, click *Deploy* to open the *Deploy Table* dialog.
7. Enter a business and technical name for your table and click *Deploy* to create the table and deploy it with the imported data.

You are returned to the *Data Builder* start page and will receive a notification when the deployment is complete and the table can be opened in the table editor.

5.4.1 Column Properties and Profiling

To review the properties of a column and obtain profiling information for the data it contains, hover over a column in the list in the *Details* side panel, and click the *Details* icon.

Property	Description
Column Details	Displays the name of the column that will be created. You can modify this name.
Data Type	<p>Displays the data type of the column that will be created. The following data types are available:</p> <ul style="list-style-type: none">• Boolean• Integer• Number - Converted into <code>Decimal(38,19)</code> after import.• String - Converted into <code>String(5000)</code> after import.• Date• Time• Date and Time

Note

More advanced SAP HANA data types such as `ST_POINT` (see [Column Data Types \[page 110\]](#)) are not supported here, but you can derive columns such as these in a view builder.

You can select a different data type from the list and validate the choice by clicking the *Validation* button. Invalid values are marked in red and will be imported as `NULL`.

Property	Description
Conversion Format	<p>Displays the syntax for <code>Date</code>, <code>Time</code>, and <code>Date and Time</code> data:</p> <ul style="list-style-type: none"> <code>Date</code> data must include, as a minimum, a year in <code>YY</code> or <code>YYYY</code> format. <code>Time</code> data must include, as a minimum, hours in <code>HH12</code>, <code>HH24</code> format. Click the <i>i</i> button for detailed format information. <code>Date and Time</code> data must include, as a minimum, a date and hours in <code>YYYY-MM-DDTHH12</code> or <code>YYYY-MM-DDTHH24</code> format. Click the <i>i</i> button for detailed format information.
Set as Key	Select the checkbox to specify that the column acts as a primary key. Each row in a key column should contain a unique value.
Data Distribution	<p>Displays profiling information for the data in the column textually and as a bar chart:</p> <ul style="list-style-type: none"> <i>Rows / Sampled Rows</i> - Displays the number of rows found in the file. <i>Unique Values</i> - [string columns] - Displays the number of unique values found in the column, with each value having a bar in the chart sorted with the most common value at the top. Click a bar to select each row with that value in the data preview on the left. <i>Number of Bars</i> - [numeric columns] Displays the number of ranges into which the values are divided, with each range having a bar in the chart. You can use the slider to control the number of bars displayed. Click a bar to display further statistical information below the chart and to select each row with that value in the data preview on the left.
Validation	Displays any issues detected in the data in the column.

5.4.2 Apply Data Transforms

You can perform data preparation transformations to modify the list of columns to be created, to delete rows, and to modify data in cells before creating your table.

Note

For many transformations, you can hover over the menu item to see a preview of the effect of the transformation. To confirm the transformation, press `Enter` or click the checkmark in the *Create Transform* bar.

Modify Columns

Modify the columns to be created in any of the following ways:

- Delete columns - Select one or more columns and click `⋮ (Menu) >> Delete Column >`.

- Duplicate a column - Select a column and click **⋮ (Menu) >> Duplicate Column >**.
- Concatenate two or more text columns together - Select a column, click in the *Create Transform* bar, select *Concatenate*, and complete the formula (see [Concatenate Columns \[page 126\]](#)).
- Split a text column on a delimiter character - Select a column, click in the *Create Transform* bar, select *Split*, and complete the formula (see [Split a Column \[page 126\]](#)).
- Extract text to a new column - Select a column, click in the *Create Transform* bar, select *Extract*, and complete the formula (see [Extract Text to a New Column \[page 126\]](#)).
- Change the case of a text column - Select a column, click in the *Create Transform* bar, select *Change*, and complete the formula (see [Change the Case of a Text Column \[page 127\]](#)).

Delete Rows

Reduce the amount of data to be imported by deleting rows in either of the following ways:

- Select one or more values in a column and:
 - Click **⋮ (Menu) >> Delete Selected Rows >** to delete all rows where the column contains values in your selection.
 - Click **⋮ (Menu) >> Keep Selected Rows >** to delete all rows where the column contains values not in your selection.
- Click in the *Create Transform* bar, select *Filter*, and complete the formula ([Filter and Delete Rows \[page 128\]](#)).

Modify Cell Data

- Replace data in cells - Select one or more cells in a column containing the values you want to modify, and:
 - Click **🔍 (Transform)** and select one of the proposed replacements.
 - Click in the *Create Transform* bar, select *Replace*, and complete the formula (see [Find and Replace Data \[page 127\]](#)).

Undo and Redo Transformations

You can undo and redo transformations with the **↶ (Undo)** and **↷ (Redo)** buttons on the toolbar or in the *Transform Log*.

You can review the history of your transformations in the *Transform Log*. Hover over a specific transformation to highlight the impacted column. To roll back a change, click **⊗ (Undo)** to delete the entry.

Note

You can undo a transformation even out of sequential order if it displays an **⊗ (Undo)** button. If the button is not available, then other transformations depend on this transformation, and these must be deleted before you can undo it.

5.4.3 Concatenate Columns

You can concatenate two or more text columns together during data preparation.

Select a column, click in the *Create Transform* bar, select *Concatenate*, and complete the formula as follows:

```
Concatenate [<Column1>], [<Column2>]... using "<string>"
```

Where:

- `<string>` - [optional] Enter a string to insert between concatenated values.

5.4.4 Split a Column

You can split a text column on a delimiter character during data preparation.

Select a column, click in the *Create Transform* bar, select *Split*, and complete the formula as follows:

```
Split [<column>] on "<string>" repeat "<number>"
```

Where:

- `<string>` - [required] Enter a string to trigger the split.
- `<number>` - [default="1"] Enter the number of times the split can be applied (and hence the number of additional columns to create).

5.4.5 Extract Text to a New Column

You can extract data from a text column to a new column during data preparation.

Select a column, click in the *Create Transform* bar, select *Extract*, and complete the formula as follows:

```
Extract [<unit>]from [<column name>][<location>] [<occurrence>] ["<value>"]  
[<include/exclude>]
```

Where:

- `<unit>` - Choose what to extract:
 - `word` [default] - Extract text and stop at the nearest space.
 - `everything` - Extract all text without regard to spaces.
- `<location>` - Choose where to find the unit to extract in relation to a string:
 - `before` - Find the unit before a string.
 - `after` - [default] Find the unit after a string.
 - `between` - Find the unit between two strings.
 - `containing` - Find the unit inside a string. For example, if your target is `ship`, both `ship` and `shipping` will be extracted.
 - `equal to` - Extracts the specific target value if it exists in the cell.

- `<occurrence>` - Choose which occurrence of the value to search around:
 - `first` - The first occurrence of the string.
 - `last` - The last occurrence.
 - `occurrence` - Specify a particular occurrence by entering a number (1-10).

For example:

- To extract the first number from all cells in `MyColumn`, specify:

```
Extract before first " " from [ MyColumn ]
```

To extract all text between parenthesis from all cells in `MyColumn`, specify:

```
Extract everything between "(" and ")" from [ MyColumn ]
```

5.4.6 Change the Case of a Text Column

You can change the case of a text column during data preparation.

Select a column, click in the *Create Transform* bar, select *Change*, and complete the formula as follows:

```
Change [<Column>] to (<UPPERCASE/lowercase>)
```

5.4.7 Find and Replace Data

You can find and replace data in cells during data preparation.

Select one or more cells in a column, click in the *Create Transform* bar, select *Replace*, and complete the formula as follows:

```
Replace (<scope>) in [<Column>] matching "<value>" with "<value>" <...>
```

Where:

- `<scope>`: Choose the scope of the find/replace action::
 - `cell` - Cells in the column matching `<value>`
 - `all values` - All cells in the column.
 - `content` - Strings in any cell in the column matching `<value>`
 - `null` - Cells containing null values.
- `matching "<value>"` - Specify the string to find.
- `with "<value>"` - Specify the string to replace with.
- `...` - [optional] Choose where to specify a where clause:
 - `[<Column>]` - Specify the column to search for the where condition.
 - `matches "<value>"` - Enter a value to match.

5.4.8 Filter and Delete Rows

You can filter data and delete any rows that match (or do not match) the criteria during data preparation.

Select one or more cells in a column, click in the *Create Transform* bar, select *Filter*, and complete the formula as follows:

```
Filter values in [<Column> ][<criteria>] "<values>"
```

Where:

- **<criteria>** - Choose the criterion type on which to filter and delete rows:
 - `matching` - Matching any of the specified **<values>**
 - `not matching` - [default] Not matching any of the specified **<values>**
 - `between` - [numeric columns] Between the **<from>** and **<to>** values
 - `matching null` - Containing null values
 - `not matching null` - Containing any non-null **<values>**.
- **<values>** - Enter the strings to match against, using commas to separate multiple values.

5.5 Creating a Data Flow


Create a data flow to move and transform data in an intuitive graphical interface. You can drag and drop sources from the *Source Browser*, join them as appropriate, add other operators to remove or create columns, aggregate data, and do Python scripting, before writing the data to the target table.

Context

📌 Note

For optimal performance, it is recommended that you consider staggering the scheduled run time of tasks such as data flows and task chains that may contain these tasks. There is a limit on how many tasks can be started at the same time. If you come close to this limit, scheduled task runs may be delayed and, if you go beyond the limit, some scheduled task runs might even be skipped.

Procedure

1. In the side navigation area, click  (*Data Builder*), select a space if necessary, and click *New Data Flow* to open the editor.
2. Drag one or more source objects from the *Source Browser* and drop it into the diagram (see [Add a Source \[page 132\]](#)).





⚠ Restriction


- Data flows support loading data exclusively to local tables in the SAP Datasphere repository.
- Data flow currently doesn't support double quotes in column names, table names, owners, or other identifiers. If the source or target operators in a data flow contains double quotes, we recommend you to create a view in the source or in SAP Datasphere that renames the columns containing double quotes.
- Data flows don't support spatial data type columns.

3. Transform your data using one or more operators:


- *Join* - Insert a join operator to merge two data sets together using a join definition to match the records. See [Create a Join Operator \[page 135\]](#).
- *Union* - Insert a union operator to combine two data sets that share the same schema definition. See [Create a Union Operator \[page 137\]](#).
- *Projection* - Insert a projection operator to add, remove, reorder, or rename columns. See [Create a Projection Operator \[page 138\]](#).
- *Aggregation* - Insert an aggregation operator to perform SUM, AVG, MIN, MAX, or COUNT calculations. See [Create an Aggregation \[page 140\]](#).
- *Script* - Insert a script operator to transform incoming data with a Python script and output structured data to the next operator. See [Create a Script Operator \[page 141\]](#).

4. Select an object in the canvas to display its properties in the side panel and to reveal its contextual toolbar, which contains some or all of the following tools:

Tool	Description
→ (Create Flow)	Click to create a flow to another operator in the data flow. Drag and drop the yellow dotted line to the required operator to which you want to connect.
 (Preview Data)	Click to preview data of the selected operator. The Data Preview section is displayed. <div data-bbox="478 1310 1396 1657" style="border: 1px solid #ccc; padding: 10px; margin-top: 10px;"><h4>📌 Note</h4><ul style="list-style-type: none">• If the table is wide or contains a number of large column types, the result in data preview may be truncated in order to avoid out of memory issues.• Data preview is not available for ABAP sources. Except for:<ul style="list-style-type: none">• CDS views if the source connection is SAP S/4HANA Cloud 2302, SAP S/4HANA on-premise 1909 or higher.• SAP Landscape Transformation Replication Server objects if ABAP Add-on DMIS 2018 SP08 / DMIS 2020 SP04 is installed.• You can't perform data preview on the transformation operators.</div>
 (Add Table)	Create a new target table.
 (Impact and Lineage Analysis)	Open the Impact and Lineage Analysis diagram. This diagram enables you to understand the lineage and impacts of the selected object. (see Impact and Lineage Analysis [page 34] .)
 (Open in New Tab)	Open the selected entity in its own editor in a new tab.

Tool	Description
 (<i>Remove</i>)	Click to delete the selected operator.

Note

You can also delete the selected operator by clicking  in the toolbar.

Tip

In the toolbar, you can use  (*Auto Layout*) and  (*Zoom to Fit*) to organize the objects in your canvas.

5. Add or create a target table that the data flow will write its data to.

Note

You can only have one target table in a data flow.

For more information, see [Add or Create a Target Table \[page 143\]](#).

6. Click on the canva and review your data flow properties in the right panel:
 - Under *General*:

Property	Description
Business Name	Enter a descriptive name to help users identify the object. This name can be changed at any time.
Technical Name	<p>Displays the name used in scripts and code, synchronized by default with the <i>Business Name</i>.</p> <p>To override the default technical name, enter a new one in the field. Technical names can contain only alphanumeric characters and underscores.</p>
	<p>Note</p> <p>Once the object is saved, the technical name can no longer be modified.</p>
Package	<p>Select the package to which the object belongs.</p> <p>Packages are used to group related objects in order to facilitate their transport between tenants.</p>
	<p>Note</p> <p>Once a package is selected, it cannot be changed here. Only a user with the DW Space Administrator</p>



Property	Description
	<p>role (or equivalent privileges) can modify a package assignment in the <i>Packages</i> editor.</p> <p>For more information, see Creating Packages to Export.</p>
Status	<p>[read-only] Displays the deployment and error status of the object.</p> <p>For more information, see Saving and Deploying Objects [page 39].</p>

- Under Run Status:
Displays the status of the flow run:
 - *Running*: The flow is currently running.
 - *Completed*: The flow is completed successfully.
 - *Failed*: Something goes wrong during the flow run and it could not be completed. Go to the details screen of your flow run and check the logs to identify to issue.
- Under Input Parameters: Create a new input parameter or modify an existing one. For more information, see [Create an Input Parameter \[page 142\]](#)
- Under Advanced Properties
 - *Dynamic Memory Allocation* You can allocate memory usage manually. Set the *Expected Data Volume* to *Small*, *Medium*, or *Large*.

Note

- Dynamic memory allocation should be done only if your data flow run is facing out-of-memory failures.
- If multiple data flows are scheduled with *Expected Data Volume* as large, it doesn't alter/ increase the actual memory needed for the data flow. However, the execution engine will allocate enough memory to handle such large volume of data and only after successful memory allocation, the data flow run is started.
Execution engine allocates memory based on the data volume configured and the complexity of the operations performed in the data flow.

- *Automatic restart on run failure*: Set this option to restart the data flow automatically if there are any failures or system upgrades, for example.

7. Click  (*Save*) to save your data flow:
 - *Save* to save the data flow.
 - *Save As* to create a local a copy of the data flow. The data flow must have been previously saved at least once. The *Save* dialog opens. Enter new business and technical names and click *Save*.
8. Click  (*Deploy*) to deploy the data flow:
 - Newly created data flows are deployed for the first time.
 - Data flows that have changes to deploy are redeployed.

With deployment, you will be able to save draft version of your data flow without affecting the execution.

5.5.1 Add a Source

Add a source to read data from. You can add multiple sources and combine them together using join or union operators.

Procedure

1. If the *Source Browser* panel is not visible on the left of the screen, click *Source Browser* in the toolbar to show it.
2. Browse or search for the object you want to add on either of the tabs.
 - The *Repository* tab lists all the tables, views, and intelligent lookups that are available in the space (including objects shared to the space).. For more information, see [Add Objects from the Repository \[page 292\]](#).
 - The *Sources* tab lists all the connections and other data sources that have been integrated to the space from which you can import tables. However it shows only limited records. If you can't see the sources you are looking for, use *Import from Connection* to perform search. You can:
 - Expand the data sources to browse through their objects (see [Import an Object from a Connection or Other Source \[page 294\]](#)).
 - Open the *Import Objects from Connection* dialog on a particular connection to select multiple objects for import (see [Import Multiple Objects from a Connection \[page 296\]](#)).
3. Select the object of your choice, and then drag and drop it onto the diagram.

Note

- You cannot use views with input parameters as sources in a data flow.
- When browsing a remote file storage such as Amazon Simple Storage Service, Google Cloud Storage, or Microsoft Azure Blob Storage, you can only select files of type JSON/JSONL, CSV, XLS/XLSX, ORC, or PARQUET. Note that each cloud provider has its own naming convention for defining bucket and object names. These conventions should be followed accordingly to avoid any source-related issue. Even though Flowagent-based operators accept most special characters, we recommend that only alphanumeric characters are used (no multibyte characters). This will avoid any undesired issue because all sources work well with such characters. Furthermore, some special characters such as ", +, and , are not allowed by our Flowagent File Producer operator and should not be used.
- Local tables with delta capture can be added as source tables. However, only the active records will be used. See [Capturing Delta Changes in Your Local Table \[page 118\]](#)

Restriction

If you add an excel file, you must ensure that the file size does not exceed 50 MB.

4. Click the source node to display its properties in the side panel, and complete the properties in the *General* section:

Property	Description
Label	Enter a label to display in the source symbol.
Business Name / Technical Name / Type / Connection	[read-only] Provide information to identify the source table.
Package	<p>Select the package to which the object belongs.</p> <p>Packages are used to group related objects in order to facilitate their transport between tenants.</p> <div style="border: 1px solid #ccc; padding: 5px; margin-top: 10px;"> <p>Note</p> <p>Once a package is selected, it cannot be changed here. Only a user with the DW Space Administrator role (or equivalent privileges) can modify a package assignment in the Packages editor.</p> </div> <p>For more information, see Creating Packages to Export.</p>
Type	Type of object. For example: a local table.
Status	<p>[read-only] Displays the deployment and error status of the object.</p> <p>For more information, see Saving and Deploying Objects [page 39].</p>
Use As	<p>[read-only] Specifies whether the table is used as a Source or Target in the data flow.</p> <div style="border: 1px solid #ccc; padding: 5px; margin-top: 10px;"> <p>Note</p> <p>Changing the use of a table will reset all its properties.</p> </div>

5. In the [Columns](#) section, review the columns of the source.

- To view the data type of a column, hover over it and click ⓘ
- If the source is a CSV, JSON, or Excel file, you can change the data type of a column and its corresponding properties by hovering over the column and clicking ⋮ (Menu) ▶ [Edit Column](#) ▶.

ⓘ Note

For CSV, JSON, or Excel files, the schema is calculated based on a subset of data and hence you can adjust/verify the suggested data as per your requirements to cover all the data.

- For a remote source, you can control the data being read.
 - To add/delete columns from source, click + / 🗑️ ([Delete](#)) respectively.
 - To add new/removed columns from the source based on latest metadata, click 🔄 ([Refresh to fetch column changes from source](#))

6. [optional] For CDI, OData, and ABAP CDS sources complete the [Source Filters](#) section to define filter conditions for the consumption of their data:

When defining filter conditions for columns, the following rules are applied:

- An "OR" is applied between the conditions defined on the same column.
- An "AND" is applied between the conditions defined on different columns.


Note


To apply the defined filters upon data preview, toggle the switch button, *ON* in the *Properties* pane.

For SAP SuccessFactors, OData, and Cloud Data Integration (CDI) sources, you can enable delta loading. That is, move over the changed data records instead of doing a full load every time.

To do this, in the *Source Filters* dialog, you must manually enter the condition `$MAX_VALUE_<column_name>` in the *Value* field.


`<column_name>` is a column in the target table where data is being replicated, normally of type `date` or `timestamp`.

- [optional] For source tables from OData remote connections, you can add or edit the parameters of a selected source table in the *OData Parameters* section. OData parameters allow you to define additional information or constraints to control how the data is returned from the OData connection. Click  (*Define Parameters*) to open the *OData Parameters* dialog. You can:

- Manually enter a custom parameter according to URI conventions (see [URI Conventions](#) ).
- Click the *Parameters* dropdown button to see a list of suggested parameters.

Add a value and click *OK* to validate your changes or *Reset* to erase all parameters.

In the *OData Parameters* section, enable the *Apply Parameters on Preview* toggle if you want to see how the changes in parameter impact the table preview.

- [optional] For source tables from OData remote connections, you can edit the depth properties of a selected source table in the *OData Properties* section. Click  (*Edit OData Properties*) to open the *OData Properties* dialog. You can set the depth of the source table to either 1 or 2.

The depth of an OData object refers to the level of related entities that are included in the response when querying the OData service. The depth is by default set to 1 so that only the properties of the requested entity are returned. You can change the depth to 2 to include a second level. Depth is useful when you want to optimize performance by controlling the amount of data returned in a single request.

Example

If an OData service has two objects for **Products** and **Orders**, and a **Product** object has a navigation property to related **Order** objects, a request for a specific **Product** with a depth of 1 will return the properties of that **Product**, but not the related **Order** objects. But if you specify a depth of 2, it will also return the related **Order** objects and their properties.

Restriction

When the depth is set to 2, the *Columns* section shows the columns from the two collections (or levels) and the following problems occur:

- The *Data Preview* cannot show more than one collection. The columns from the second collection cannot be previewed.
- The Data Flow run will fail.

Retaining columns from just one collection and deleting the rest fixes both issues.

- [optional] For sources that are files, complete the appropriate additional section:
 - For `.csv` files, click *Modify* in the *CSV Properties* section to configure various csv properties for your `.csv` file (column separator, text separator, character set, escape character, etc.).

- For `.json` and `.json1` files, click [Modify](#) in the *JSON Properties* section to configure the flattening levels of the JSON file.
- For `.xls` and `.xlsx` files, click [Modify](#) in the *Excel Properties* to choose the Excel worksheet to be used and to set a header row.

Note

You can reset all your changes in the section by clicking [Modify](#) [Reset](#).

10. [optional] In the *Advanced* section, configure the following properties:

Property	Description
Control Fetch Size	To enforce a specific value for Batch Size, toggle this switch.
Fetch Size (Number of Rows)	Specifies the amount of data being read. It indicates the number of rows that will be committed to target on each batch.
Batch Query	Fetches the data batch by batch according to the fetch size defined above. Use this option for very large odata source files.
Fail Run on String Truncation	Fails the data flow run if string truncation is detected while fetching the source columns. This property is available only for CSV, JSON, and Excel files.
Enable ODBC Tracing	: [SAP HANA connection only] Enable this option to create a new log file in the vflow graph pod with HANA ODBC debug logs.

Caution

Enabling this option must be used for troubleshooting purposes only, as it uses a lot of system resources.

Note

You can reset all your changes in the section by clicking [Restore Default](#).

11. To finalize the source, create a flow from it to an operator or to the target table, as appropriate.

5.5.2 Create a Join Operator

Insert a join operator to merge two data sets together using a join definition to match the records.

Context

The join operator requires two inputs, and it generates a single output.

Note

The size limit for files being processed by the join operator is 10 GB.











Procedure

1. Click the *Join* tool, drag it onto the diagram canvas, and release it where you want to create the join.
2. Click and drag the port on the right of the first source node that you want to join and drop it onto the join operator.

A flow is created between the source and the join operator.

3. Repeat the action for the second source node to create its flow to the join operator.
4. Click the join operator to display its properties in the side panel, and complete the properties in the *General* section:

Property	Description
<i>Label</i>	Provide a suitable name for your join as per your requirement.
<i>Join Type</i>	Choose the type of join. The joins that are currently supported are INNER, LEFT_OUTER, and RIGHT_OUTER.

5. In the *Join Definition* section, click on  (*Edit*) to modify the predefined join mappings.
 - Click on  (*Auto Map*) to automatically create a join that matches column names if possible.
 - To manually create a new join mapping, drag and drop any left column onto any right column.
 - To delete a join mapping, select it and click .
 - To clear all mappings, click  (*Remove all Mappings*).
6. In the *Columns* section, specify the mapping of your columns:
 - All unique columns from the input operators are displayed. Hover on a column and click  to view its data type.
 - You can reorder the columns by dragging and dropping.
 - You can add any missing columns from input operators using  (*Add Column*) and select the columns.
 - Click  (*Menu*) to rename, remove, or duplicate selected column.
 - To remove multiple columns at once, press the `Ctrl` key, click on each columns you want to select, and click  (*Delete Columns*).
 - To remove all columns, click *Select All* and choose  (*Delete Columns*).
7. In the *Advanced Properties* section, click on  (*Edit*) to define the join optimization. By default, it is set to *Automatic*. Manual optimization should be done only when absolutely necessary.
 - Select *Join Optimization* as *Manual*.
 - For left and right joins, respectively;
 - Choose *Cache* as *Yes* or *No*.
 - Specify the *Rank* value within the range 0–100.
8. To complete the definition of the join, create a flow from it to the next operator or the target table, as appropriate.

5.5.3 Create a Union Operator

Insert a union operator to combine two data sets that share the same schema definition.


Procedure

1. Click the *Union* tool, drag it onto the diagram canvas, and release it where you want to create the union.
2. Click and drag the port on the right of the first source node that you want to join and drop it onto the union operator.

A flow is created between the source and the union operator.


3. Repeat the action for the second source node to create its flow to the union operator.
4. Click the union operator to display its properties in the side panel, and complete the properties in the *General* section:

Property	Description
<i>Label</i>	Provide a suitable name for your union operator as per your requirement.
<i>Union All</i>	Toggle this button to include/exclude the duplicate records from the input operators. For example, if <i>Union All</i> is disabled, all the duplicate records are removed from the final result.

5. In the *Mappings* section, review the predefined union mappings.
 - Click on *All*, *Mapped*, or *Unmapped* to show the union columns of your choice.
 - Drag a column from source and drop onto the required union column to create or modify the mapping.
 - To delete mappings, select one or more existing mappings and click .

You can click  (*Settings*) to manually add or delete source columns as output columns of the union, and select the required action.

Property	Description
<i>Add All Source Columns as Output Columns</i>	Add all columns in the left list to the right list (if they are not already present).
<i>Add Selected Source Columns as Output Columns</i>	Add columns selected in the left list to the right list (if they are not already present).
<i>Delete Selected Output Columns</i>	Delete any columns selected in the right list so that they are no longer union output columns.
<i>Delete All Output Columns</i>	Delete all columns in the right list so that they are no longer union output columns.

6. In the *Columns* section, specify the columns.
 - All columns from the first input operator are displayed. Hover on a column and click  to see its data type.

- You can reorder the columns by dragging and dropping.
 - Click **⋮** (*Menu*) to rename or remove selected column.
 - To remove multiple columns at once, press the **Ctrl** key, click on each columns you want to select, and click **✖** (*Delete Columns*).
7. To complete the union, create a flow from it to the next operator or the target table, as appropriate.

5.5.4 Create a Projection Operator

Insert a projection operator to add, remove, reorder, or rename columns.

Procedure

1. Click the *Projection* tool, drag it onto the diagram canvas, and release it where you want to create the projection.
2. Click and drag the port on the right of the source node that you want to join and drop it onto the projection operator.

A flow is created between the source and the projection operator.

3. Click the projection operator to display its properties in the side panel, and complete the properties in the *General* section:

Property	Description
<i>Label</i>	Provide a suitable name for your projection as per your requirement.

4. In the *Columns* section, specify the mapping of your columns.
 - Hover on a column and click **ⓘ** to view its data type.
 - You can reorder the columns by dragging and dropping.
 - You can add any missing columns from input operator using **+** (*Add Column*) and select the columns. You can also add calculated column. See [Create a Calculated Column in a Projection Operator \[page 139\]](#) for more information.
 - Click **⋮** (*Menu*) to rename, remove, or duplicate a selected column.
 - You can edit the name, data type, and expression of a column by choosing **>** option.
 - To remove multiple columns at once, press the **Ctrl** key, click on each columns you want to select, and click **✖** (*Delete Columns*).
 - To remove all columns, click *Select All* and choose **✖** (*Delete Columns*).
5. (Optional) In the *Filter* section, define filters.

In the *Expression* field, enter the required expression. You can build your expression by choosing from the *Functions* (see [Function Reference for Data Transformation Language](#)), *Columns*, *Input Parameters* (see [Create an Input Parameter \[page 142\]](#)), and *Operators*.

Note

To validate your expression so that you can find any issues with it and fix them, click the [Validate](#) button.

- In the expression, you must enclose a column name within double quotations and a constant string value within single quotations.

Caution

If your expression contains boolean columns, the validation capabilities are limited.

6. To complete the projection, create a flow from it to the next operator or the target table, as appropriate.

5.5.5 Create a Calculated Column in a Projection Operator

Add a new column in a projection operator and define the expression that will fill it with data.

Procedure

1. In the properties panel of the projection operator, choose [+](#) ([Add New Calculated Column](#)) in the [Columns](#) section.
2. Provide a name and data type for the column. Additional fields are displayed for the following data types:

Data Type	Additional Field
string	(Optional) Length : Define the maximum number of characters that is allowed for the column.
binary	(Optional) Length : Define the maximum number of characters that is allowed for the column.
decimal	Precision : Define the maximum number of digits (including Scale) that is allowed for a value in the column. Scale : Define the maximum number of digits that is allowed after the decimal point for a value in the column.

Note

Precision must be greater than scale. For example, if Precision is 4 and Scale is 2, the value allowed for the column is XX.xx

For more information, see [Data Types](#) from the [SAP HANA Cloud SQL Reference Guide](#).

3. In the [Expression](#) field, define the expression for the calculated column. You can build your expression by choosing from the [Functions](#) (see [Function Reference for Data Transformation Language](#)), [Columns](#), [Input Parameters](#) (see [Create an Input Parameter \[page 142\]](#)), and [Operators](#).

Note

To validate your expression so that you can find any issues with it and fix them, click the [Validate](#) button.

- In the expression, you must enclose a column name within double quotations and a constant string value within single quotations.

5.5.6 Create an Aggregation

Insert an aggregation operator to perform SUM, AVG, MIN, MAX, or COUNT calculations.

Procedure

1. Create a projection operator (see [Create a Projection Operator \[page 138\]](#)) and remove all the columns from your dataset except those that you want to aggregate and those that you want to group the aggregations by.
2. Click the [Aggregation](#) tool, drag it onto the diagram canvas, and release it where you want to create the aggregation.
3. Click and drag the port on the right of the projection operator and drop it onto the aggregation operator.
A flow is created between the projection and the aggregation operator.
4. Click the aggregation operator to display its properties in the side panel, and complete the properties in the [General](#) section:

Property	Description
Label	Provide a suitable name for your aggregation as per your requirement.

5. In the [Columns](#) section, review the output columns and set aggregations.

Hover over a column and click:

- [⋮ \(Menu\) >> Change Aggregation](#) to change its aggregation. You can choose:
 - [no value] - Default. Use as grouping column.
 - SUM - [numeric columns only] Calculate the total value of all the rows.
 - AVG - [numeric columns only] Calculate the average value of all the rows.
 - MIN - [numeric columns only] Calculate the minimum value of all the rows.
 - MAX - [numeric columns only] Calculate the maximum value of all the rows.
 - COUNT - Calculate the number of distinct values.

Note

Any column that has the default, empty aggregation is used to group the aggregations by its unique values. If multiple columns are used to group, then their unique value combinations are used for grouping.

- ... (Menu) ► ► *Change Name* ► to change its name.
 - ⓘ to view its data type.
6. To complete the aggregation, create a flow from it to the next operator or the target table, as appropriate.

5.5.7 Create a Script Operator

Insert a script operator to transform incoming data with a Python script and output structured data to the next operator.

Context

The script operator receives the data from a previous operator. You can provide transformation logic as a body of the `transform` function in the Script property of the operator. Incoming data is fed into the `data` parameter of the `transform` function and the result from this function is returned to the output.

The script operator allows data manipulation and vector operations by providing support for **NumPy** and **Pandas** modules. Non-I/O objects and functions of **NumPy** and **Pandas** can be used with aliases `np` and `pd`, respectively, without any requirements to explicitly import them.

The incoming `data` parameter in the `transform` function is of type Pandas DataFrame. The input table is converted into a DataFrame and fed into transform function as `data` parameter. You are expected to provide scripts for the intended transformation of the incoming DataFrame and also return a valid DataFrame from transform function. It is important that the returning DataFrame from the transform function has the same column names, types and order as the specified table for the output. Otherwise, execution of the data flow results in failure.

⚠ Restriction

In a data flow, the script operator may receive the incoming table in multiple batches of rows, depending on the size of the table. This means that the `transform` function is called multiple times, for each batch of rows, and that its `data` parameter contains only the rows for data given batch.

Hence, the operations that require the complete table within the `data` parameter are not possible. For example, removing duplicates.

Procedure

1. Click the *Script* tool, drag it onto the diagram canvas, and release it where you want to create the script operator.
2. Click and drag the port on the right of the source node that you want to join and drop it onto the script operator.

A flow is created between the source and the script operator.

- Click the script operator to display its properties in the side panel, and complete the properties in the *General* section:







Property	Description
Label	Provide a suitable name for the operator.
Code Language	[read-only] The supported language is Python.

- In the *Script* section, enter your Python script to transform the incoming data and produce an output schema.

For information about Python support in the script operator, see [Script Operator Python Reference \[page 146\]](#).

- In the *Columns* section, manually specify the columns that will be output by your script.

By default, all the input columns are displayed. You can:



- Hover on a column and click  to see its data type.
- Reorder the columns by dragging and dropping.
- Add any missing columns from input operators using  (*Add Column*) and select the columns.
- Create new columns using  (*Create New Column*) and provide a name and data type for the column. The new columns are easily identified in the the *Script* node *Columns* section with the  (*New Column*) icon.
- Remove multiple columns at once, `ctrl` + `click` to select the columns and choose  (*Delete Columns*) .
- To remove all columns, click *Select All* and choose  (*Delete Columns*) .

- To complete the script operator, create a flow from it to the next operator or the target table, as appropriate.

5.5.8 Create an Input Parameter

Create input parameters in your data flows for use in projection operator filter conditions or calculated columns. When you want to starting a data flow run, you are prompted to enter a value and this value is used to filter the data to be loaded

Procedure

- Open the data flow properties in the side panel, scroll down to the *Input Parameter* section and click  (*Input Parameters*).
- In the *Input Parameter* dialog, click  (*Add*) to create a new parameter.
- Complete the properties of the input parameter:

Property	Description
Name	Enter a descriptive name to help users identify the object. This name can be changed at any time.
Data Type	Displays <i>string</i> as the currently available type.
Default Value	[optional] Enter a default value for the input parameter. Each time the user is required to enter a value for the parameter, they can accept the default value or override it. The values must be entered inside of single quotes, for example, 'Germany'. The default value is used whenever the data flow is run as part of a schedule or task chain. You can enter <code>CURRENT_DATE ()</code> or <code>CURRENT_TIME ()</code> to obtain the current UTC date or timestamp at runtime.

- Click *OK* to close the dialog and return to the side panel where the input parameter is available for use.
- Use the input parameter as appropriate in the projection operator.

Example	Description
<code>Country = \$ { IP_Country }</code>	Projection Filter expression (see Create a Projection Operator [page 138]). This filter expression takes the value entered by the user for <code>IP_Country</code> and uses it to restrict the data returned by the data flow to only those rows where the <code>Country</code> column contains this value.
<code>ADD_DAYS (Date , \$ { IP_Days })</code>	Calculated Column expression (see Create a Calculated Column in a Projection Operator [page 139]). This expression will fill the column with a value calculated by taking the value in the <code>Date</code> column and adding the value entered by the user (days) to the date.

Note

At runtime, the string entered by the user is converted into the correct data type.

Note


You must use each of the input parameters that you create at least once in your data flow or you will receive an error instructing you to use or delete them.

5.5.9 Add or Create a Target Table

Add a target table to write data to. You can only have one target table in a data flow.

Procedure

- Add a target table to the data flow in one of the following ways:

- Click  (*Add Table*) in the toolbar, and drag and drop it onto the canvas to create a new target table. The target table is added to your data flow. The text **New** at the bottom-right of the operator shows that the table is newly added.
- Drag an existing table from the *Source Browser*, drop it onto the canvas, and click the floating *Target* button to use it as the target table. You can choose:
 - On the *Repository* tab - A local table.
 - On the *Sources* tab:
 - A table in an Open SQL schema - if the space is granted write access to the table or schema (see [Allow the Space to Write to the Open SQL Schema](#)).
 - A table in a database user group schema - if the space is granted write access to the table or schema (see [Allow a Space to Write to the Database User Group Schema](#)).
 - A table in an SAP HDI container - if the space is granted write access to the table or schema (see [Allow Your Space to Write to Your HDI Container](#)).


Note

- Local table with delta capture on can't be used as target table.
- If your target table is a virtual table, the following modes are not supported:
 - *APPEND* mode if *Update Records By Primary Key (UPSERT)* is selected.
 - *DELETE* mode.

2. Click and drag the port on the right of the last operator in your data flow and drop it onto the target table. A flow is created between the operator and the target table.
3. Click the target node to display its properties in the side panel, and complete the properties in the *General* section:

Property	Description
<i>Label</i>	Enter a label to display in the target table symbol.
<i>Business Name / Technical Name / Type / Connection</i>	[read-only] Provide information to identify the target table.
Package	Select the package to which the object belongs. Packages are used to group related objects in order to facilitate their transport between tenants.
	<div data-bbox="496 1574 601 1610" data-label="Section-Header"> <h3>Note</h3> </div> <div data-bbox="496 1628 1340 1729" data-label="Text"> <p>Once a package is selected, it cannot be changed here. Only a user with the DW Space Administrator role (or equivalent privileges) can modify a package assignment in the <i>Packages</i> editor.</p> </div> <div data-bbox="470 1767 1016 1798" data-label="Text"> <p>For more information, see Creating Packages to Export.</p> </div>
<i>Type</i>	Type of object. For example: a local table.

Property	Description
<i>Mode</i>	<p>Specifies the mode with which to write to the target table or modify the data in the target table..</p> <p>You can choose between:</p> <ul style="list-style-type: none"> • <i>Append</i>: Write the data obtained from the data flow as new records appended to the end of the target table. • <i>Truncate</i>: Erase the existing data in the target table and replace it with the data obtained from the data flow. • <i>Delete</i>: Delete records in the target table based on the columns mapped in the target. All the column mappings are considered for the match condition and if the value of these columns match the record in the target table, the record will be deleted. <div style="border: 1px solid #ccc; padding: 5px; margin-top: 10px;"> <p>Note</p> <p>The Delete mode fails for a target table when a mapped column contains NULL values.</p> </div>
<i>Update Records By Primary Key (UPSERT)</i>	<p>[Append mode] Instructs the flow to update, where appropriate, existing target table records that can be identified by their primary keys. If this option is not selected then all source records (including those that are already present in the target table) are appended, which may cause duplicate key errors.</p> <div style="border: 1px solid #ccc; padding: 5px; margin-top: 10px;"> <p>Note</p> <ul style="list-style-type: none"> • When working with data lake, only APPEND (UPSERT option not selected) and TRUNCATE modes are supported. APPEND (with UPSERT option selected) and DELETE modes are not supported. • If no primary key is defined in the target table, then all records will be appended. </div> <p>This feature is based on the UPSERT statement (see Upsert in the SQL Reference for SAP Vora in SAP Data Intelligence guide).</p>
<i>Status</i>	<p>[read-only] Displays the deployment and error status of the object.</p> <p>For more information, see Saving and Deploying Objects [page 39].</p>
<i>Use As</i>	<p>[read-only] Specifies whether the table is used as a Source or Target in the data flow.</p> <div style="border: 1px solid #ccc; padding: 5px; margin-top: 10px;"> <p>Note</p> <p>Changing the use of a table will reset all its properties.</p> </div>

- In the *Mappings* section, review the mappings of incoming columns to the target table columns.
 - When you connect any operator to the target table, each incoming columns that has the same name and a compatible data type with a column in the target table is automatically mapped to it.
 - You can manually map an incoming columns with a target table column that has a compatible datatype by dragging the column from the left list and dropping it onto the appropriate column in the right list.
 - To delete a mapping, select it, and click *Delete*. To delete all mappings, *Remove all Mappings*.
 - You can, at any time, click  (*Auto Map*) to relaunch automapping based on names and datatypes.

- In the *Columns* section, view the columns of the target table. To check the data type of a column, hover over it and click ⓘ.
- [optional] In the *Advanced* section, configure the following properties:

Property	Description
<i>Control Batch Size</i>	To enforce a specific value for Batch Size, toggle this switch.
<i>Batch Size (Number of Rows)</i>	Specifies the amount of data being read. That is, it indicates the number of rows that will be fetched from the source in each request.
<i>Enable ODBC Tracing</i>	: [SAP HANA connection only] Enable this option to create a new log file in the vflow graph pod with HANA ODBC debug logs.

⚠ Caution

Enabling this option must be used for troubleshooting purposes only, as it uses a lot of system resources.

ⓘ Note

You can click the icon *Restore Default* to restore the default settings.

- Click *Create and Deploy Table* in the properties panel to create and deploy the newly added target table to the repository in your space.

This step is not necessary if you added an existing table.

5.5.10 Script Operator Python Reference

Learn about the Python support offered by the data flow script operator.

This topic contains the following sections:

- [Support for builtins, Numpy, and Pandas Modules \[page 146\]](#)
- [Support for builtins Exception Types \[page 149\]](#)
- [Data Mapping \[page 150\]](#)
- [Basic Examples \[page 151\]](#)
- [Handling Decimal Types \[page 153\]](#)

Support for builtins, Numpy, and Pandas Modules

The script operator offers support for data manipulation and vector operations via support for the `builtins` (version 3.9), `NumPy` (version 1.21.5), and `Pandas` (version 1.2.5) modules. In order to achieve a safe execution environment, some restrictions are applied.

ⓘ Note

Use of the unsupported functions or objects listed here triggers an internal exception, resulting in a runtime data flow execution failure.

Built-in Module (version 3.9)**NumPy Module (version 1.21.5)****Pandas Module (version 1.2.5)**

All constants and functions in the Python `builtins` module are supported except the following:

- `eval`
- `exec`
- `compile`
- `classmethod`
- `staticmethod`
- `breakpoint`
- `memoryview`
- `__import__()` i.e. import statement
- `input`
- `open`
- `dir`
- `object`
- `print`
- `id`
- `vars`
- `hasattr`
- `getattr` and `__getattr__`
- `setattr` and `__setattr__`
- `delattr` and `__delattr__`

All objects and functions in the NumPy module are supported (using the standard `np` alias) except the following:

- `load`
- `save`
- `savez`
- `savez_compressed`
- `loadtxt`
- `savetxt`
- `genfromtxt`
- `fromregex`
- `fromfile`
- `memmap`
- `DataSource`
- `ndarray.tofile`
- `ndarray.dump`

All objects and functions in the Pandas module are supported (using the standard `pd` alias) except the following:

- `read_pickle`
- `read_table`
- `read_csv`
- `read_fwf`
- `read_clipboard`
- `read_excel`
- `read_json`
- `read_html`
- `read_hdf`
- `read_feather`
- `read_parquet`
- `read_sas`
- `read_spss`
- `read_gbq`
- `read_stata`
- `read_sql_table`
- `read_sql_query`
- `io.excel.ExcelFile`
- `io.excel.ExcelWriter`
- `io.stata.StataReader`
- `io.sql.SQLDatabase`
- `io.sql.SQLiteDatabase`
- `io.pytables.HDFStore`
- `DataFrame.to_pickle`
- `DataFrame.to_csv`
- `DataFrame.to_hdf`
- `DataFrame.to_sql`
- `DataFrame.to_excel`
- `DataFrame.to_html`
- `DataFrame.to_feather`
- `DataFrame.to_stata`
- `DataFrame.to_gbq`
- `DataFrame.to_clipboard`
- `DataFrame.to_json`
- `DataFrame.to_latex`

-
- DataFrame.to_markdown
 - Series.to_pickle
 - Series.to_csv
 - Series.to_excel
 - Series.to_hdf
 - Series.to_sql
 - Series.to_clipboard
 - Series.to_json
 - Series.to_latex
 - Series.to_markdown
-

Note

These lists are subject to change.

Support for builtins Exception Types

The following builtin exception types are supported for use in error handling.

- ArithmeticError
- AssertionError
- AttributeError
- BufferError
- FloatingPointError
- IndexError
- KeyError
- LookupError
- NameError
- NotImplementedError
- OverflowError
- RuntimeError
- StopIteration
- TypeError
- UnboundLocalError
- UnicodeDecodeError
- UnicodeEncodeError
- UnicodeError
- UnicodeTranslateError
- ValueError

- ZeroDivisionError

Note

This list is subject to change.

Data Mapping

The incoming table from the input port is converted to a **Pandas** DataFrame before being passed as the data parameter to the transform function. Similarly, the table returned from the user code is converted from a **Pandas** DataFrame to the data types common to all the operators.

You can find the correspondence between the supported column data types and Python objects in the table below.

Column Data Type	Python Class	dtype	Restrictions	Observation
Binary	byte	object		
Boolean	numpy.bool	bool		
Date	pandas.Timestamp	datetime64[ns]	Can only represent range: 1677-09-21 to 2262-04-11	
DateTime	pandas.Timestamp	datetime64[ns]	Can only represent range: 1677-09-21 00:12:43.145225 to 2262-04-11 23:47:16.854775807	
Decimal	decimal.Decimal	object		
DecimalFloat	decimal.Decimal	object		
Integer	int	int64*		*The smallest possible between uint8, uint16, uint32, and uint64 that can hold all values.
Integer64	int	int64		
LargeBinary	bytes	object		
LargeString	string	object		
String	string	object		
Time	pandas.Timestamp	datetime64[ns]		
TimeStamp	pandas.Timestamp	datetime64[ns]	Can only represent range: 1677-09-21 00:12:43.145225 to 2262-04-11 23:47:16.854775807	

Note

In the presence of null values, the corresponding value in a DataFrame is of type pd.NA (not available). When null values are present, the columns have mixed types and the column dtype is object.

Basic Examples

Type of incoming data is restricted to Pandas DataFrame.

Example 1

Sample Code

```
def transform(data):  
    return data[['first', 'customer_name', 'office_location']]
```

This script creates a projection from an incoming DataFrame. It assumes that there would be at least these three columns in the incoming DataFrame: `first`, `customer_name`, and `office_location`. That is an alternative way to say that the input table has at least these three attributes. The projection script then extracts only these three columns and returns them as a DataFrame.

Input table (Example DataFrame stored locally in the script as `data`):

first	last	birthday	customer_name	office_location
John	Doe	2002-04-09 12:33:58	RedBull	Walldorf
Jane	Doe	2005-03-03 12:33:58	GoPro	São Leopoldo

Output table (Example returning DataFrame projection):

first	customer_name	office_location
John	RedBull	Walldorf
Jane	GoPro	São Leopoldo

It expects that the output table has only these three attributes, `first`, `customer_name`, and `office_location`. If the output table schema is different in any ways from this, the data flow execution fails.

Example 2

Sample Code

```
def transform(data):  
    df = data[['first', 'last']]  
    df[full] = df['first'] + '.' + df['last']  
    return df
```

This script extracts the `'first'` and `'last'` columns of the incoming DataFrame, assuming that they exist in the input table, and stores these two columns as a separate DataFrame in the local variable `df`. It then computes a third column email from the extracted columns, updates `df`, and returns it.

Input table (Example DataFrame stored locally in the script as df):

first	last
John	Doe
Jane	Doe

Output table (Example returning DataFrame df):

first	last	full
John	Doe	John.Doe
Jane	Doe	Jane.Doe

The data flow would fail if the input table has neither the first nor the last attribute, among others. Similarly, it would also fail if the output table schema does not consist of just these three attributes, `first`, `last`, and `full`.

Example 3

Sample Code

```
def transform(data):
    timestamps = data['birthday'].apply(lambda x: pd.Timestamp(x))
    data['birthday'] = timestamps.apply(lambda x: f'{x.month_name()} {x.day},
    {x.year}, {x.day_name()}')
    return data
```

This script assumes that there is a `birthday` column among others, with the string values in the incoming DataFrame. It first attempts to convert the string values of this column to pandas **Timestamp** and store as a Series. Finally, it makes a specific string representation of the birth days in the Series, using the methods and attributes from the **Timestamp** class, and updates the same in the original DataFrame, before returning the same.

Input table (Example of incoming DataFrame as data):

first	last	birthday
John	Doe	2002-04-09 12:33:58
Jane	Doe	2005-03-03 12:33:58

Output table (Example returning DataFrame as data):

first	last	birthday
John	Doe	April 9, 2002, Tuesday
Jane	Doe	March 3, 2005, Thursday

In this case, it assumes that the input and output tables have the same schema and data types.

Handling Decimal Types

To work with decimal types in the script operator, the built-in Python `Decimal` class should be used. There is no way or need to import it, as it is available for the user inside the body of the transform function. Few examples of using the `Decimal` type in the script operator are provided below.

Example 4

Sample Code

```
def transform(data):
    data['start_date_utc_long'] = Decimal('123456789012345678901')
    return data
```

This script assumes that the input table has a column name `dept_no` and `dept_name` and it wants to add decimal type arbitrary column `start_date_utc_long` to it. The actual data type precision of this column is assumed to be `decimal(21,0)` in the output table definition.

Input table (Example of incoming DataFrame as data):

dept_no	dept_name
dept_101	Audit
dept_102	Advisory

Output table (Example returning DataFrame as data):

dept_no	dept_name	start_date_utc_long
dept_101	Audit	Decimal("123456789012345678901")
dept_102	Advisory	Decimal("123456789012345678901")

In this case, it assumes that the input and output tables have the same schema and data types.

Example 5

Sample Code

```
def transform(data):
    data = data[data.salary > Decimal("1299.99")]
    return data
```

This script assumes that the input table has a column name `salary` and this column is of data type `Decimal`. The script applies a filter on the column `salary` comparing the value of its contents to a python `Decimal` object and returns the filtered table.

Input table (Example of incoming DataFrame as data):

first	last	salary
John	Doe	Decimal("1099.99")
Jane	Doe	Decimal("1599.50")

Output table (Example returning DataFrame as data):

first	last	salary
Jane	Doe	Decimal("1599.50")

The example assumes that the input and output tables have the same schema and data types.

5.5.11 Process Source/Target Changes in the Data Flow Editor

If one or more of the sources (or the target table) of your data flow is modified, then the next time you open the data flow, you will be asked to process the changes. If a change has generated errors in your data flow, you will receive a notification inviting you to review them.

Procedure

1. If a source or target change has generated errors in your data flow, you will receive a notification inviting you to review them, and you can click the notification to open it directly.

The *Source/Target Updates* dialog opens, listing the objects that have been modified.

Note

If the changes do not generate errors (for example, new columns are available), you will not receive a notification, but the dialog will still be displayed the next time that you open the data flow.

2. Click *OK* to dismiss the dialog and open the diagram. The following validation messages may be displayed:

Change	Impact in Dependent Data Flows
New Column	<p>New source or target columns are left unmapped by default in the target table node of dependent data flows:</p> <ul style="list-style-type: none"> • An information message listing all new columns is displayed on the source and/or target.
Change Column Data Type	<p>Source or target column data type changes may generate information or error messages in dependent data flows:</p> <ul style="list-style-type: none"> • An information message listing all columns with changed data types is displayed on the source and/or target. • An error message is displayed on any node where an incompatible data type change is present.


Change	Impact in Dependent Data Flows
Delete Column	<p>Deleted source columns generate errors in dependent data flows if the columns are used in these data flows:</p> <ul style="list-style-type: none"> • An information message listing all deleted columns is displayed on the source and/or target. • An error message is displayed on any intermediate node (join, union, projection, aggregation, script) in which a deleted column was used.

3. Review all information messages to ensure that they do not adversely impact the output of your data flow.
4. Review and resolve any error messages on intermediate data flow nodes.
5. If appropriate, map any new source or target columns in the target table node.

You cannot modify the structure of the target table directly in the data flow editor. If you need to make such changes (for example to add new source columns or change data types), you must open and edit it in the table editor. Once you have saved the updated target table, you can return to the data flow and adjust mappings as appropriate in the target table node.

Note

If you leave any target table column unmapped, new runs of the data flow will insert null values into it.

6. Click  (*Save*) to save the changes to your data flow and dismiss the information messages.

5.6 Creating a Replication Flow

Create a replication flow to copy multiple data assets from a source to a target.

Context

You can use replication flows to copy data from the following source objects:

- CDS views (in ABAP-based SAP systems) that are enabled for extraction
- Tables that have a primary key
- Objects from ODP providers, such as extractors or SAP BW artifacts

For more information about available connection types, sources, and targets, see [Integrating Data via Connections](#).


Note

Replication flows may not be available in SAP Datasphere tenants provisioned prior to version 2021.03. To request the migration of your tenant, see SAP note [3268282](#).

Note


To make sure that you have the most up-to-date information and important considerations regarding replication flows, please read SAP Note [3297105](#) before you start creating a replication flow.

Procedure

1. In the side navigation area, click  (*Data Builder*), select a space if necessary, and click *New Replication Flow* to open the editor.
2. Select a source connection and a source container, then add source objects (see [Add a Source \[page 157\]](#)).

The side panel shows the properties of your replication flow. Complete the missing information as appropriate:

Property	Description
Business Name	Enter a descriptive name to help users identify the object. This name can be changed at any time.
Technical Name	Displays the name used in scripts and code, synchronized by default with the <i>Business Name</i> .
Package	Select the package to which the object belongs. Packages are used to group related objects in order to facilitate their transport between tenants.
<div data-bbox="555 1227 1396 1406"><h3>Note</h3><p>Once a package is selected, it cannot be changed here. Only a user with the DW Space Administrator role (or equivalent privileges) can modify a package assignment in the <i>Packages</i> editor.</p></div> <p>For more information, see Creating Packages to Export.</p>	
Status	[read-only] Displays the deployment and error status of the object. For more information, see Saving and Deploying Objects [page 39] .
Delta Load Interval	[only relevant for load type <code>Initial</code> and <code>Delta</code>] Define the time interval for replicating changes from the source to the target. For more information, see Configure Your Replication Flow [page 160] .
Run Status	[read-only] Displays the overall status of the replication flow run, for example <code>Not Run Yet</code> . For more detailed information, go to the flow monitor.



3. Select a target connection and target container (see [Add a Target \[page 159\]](#)).
4. Click  (*Browse target settings*) to review the default target settings for your replication flow and change or complete them as appropriate (see [Configure Your Replication Flow \[page 160\]](#)).
5. Select a replication object in the canvas to review its properties in the side panel and change or complete them as appropriate:

Property	Description
Projections	Add a projection to define a filter or mappings.
Delta Capture	[only relevant for local tables] Select this option if you want the system to keep track of changes in your data source. For more information, see Capturing Delta Changes in Your Local Table [page 118] .
Load Type	Select how you want to load the data (initial only or initial and delta). For more information, see Configure Your Replication Flow [page 160] .
Truncate	Enable this option to delete any existing content in the target. For more information, see Configure Your Replication Flow [page 160] .
Target Columns	Lists the target column names. A key symbol next to a column name indicates that this column is a key column.

Note

Some further properties are only relevant for specific types of targets. You can find a list of these properties in the detailed information for the respective targets:

- [Using a Cloud Storage Provider As the Target \[page 163\]](#)
- [Using Google BigQuery As the Target \[page 165\]](#)
- [Using Apache Kafka As the Target \[page 168\]](#).

6. Click  (*Save*).
7. Click  (*Deploy*) to make your replication flow ready to run.

When you deploy your replication flow, it gets saved in the shared repository and is available for other users to view, import, or modify.

When deploying your replication flow, the system does a series of validation checks and outputs an error message if it finds an issue in the flow configuration.

8. Click  (*Run*) to start your replication flow.

For more information about how to monitor your replication flow run, see [Monitoring Flows](#).

5.6.1 Add a Source

Define the source for your replication flow (connection, container, and objects).

Context

- Connections are created by your system administration. You can only use a data source if a connection has been created for it in your space and if you have the necessary authorizations.
- Containers are the parent objects that hold the data:

- For CDS view entities, the container is called SQL_SERVICE.
- For standard CDS views, the container is the CDS root folder (CDS_EXTRACTION).
If a standard CDS view for which replication is enabled is not shown in the CDS_EXTRACTION folder, make sure that the user in the source connection has the required authorizations. For connections to an SAP S/4HANA Cloud source system, this might mean that the user must be assigned to an authorization group that contains the CDS view (as described in [Integrating CDS Views Using ABAP CDS Pipeline](#)).
- For database tables, the container is the schema that includes the table.
- For SAP Landscape Transformation Replication Server (SLT), the container is the relevant mass transfer ID. Make sure that it is available in SLT before you start creating your replication flow.
- You can use the SQL service exposure from SAP BTP, ABAP environment, or SAP S/4HANA Cloud, respectively, to replicate custom and standard CDS view entities if your system administration has created the relevant communication arrangements. For more information, see [Data Consumption using SAP Datasphere](#). For information about the relevant integration scenario, see [Integrating SQL Services using SAP Datasphere](#).
- Replication objects are the datasets that you choose for replication, for example individual CDS view entities or database tables.

Procedure

1. Choose [Select Source Connection](#) at the bottom left of the screen. A list of available source connections appears. Select the relevant one for your use case.

If you are not sure which one to choose, or if none of the connections in the list is suitable for your purposes, contact your administration.

Alternatively, you can get started by clicking  ([Browse source connections](#))

2. Choose [Select Source Container](#). A list of available source containers appears. Select the relevant one for your use case.

To narrow down the selection, start typing a part of the folder name in the [Search](#) field.

- If you choose SAP Datasphere as the source connection, the source container is automatically defined as the space you are in. In addition, the load type is automatically set to [Initial](#) because [Initial and Delta](#) is not supported for SAP Datasphere as the source.
 - If CDS view entities have been made available using the SQL service exposure in SAP BTP, ABAP environment, you find these entities in a folder called SQL_SERVICE.
3. Choose [Add Source Objects](#). A list of available objects appears. Select the relevant ones for your use case and choose [Next](#). A list of the objects you selected appears.

Note

- The list only shows objects for which replication is supported. For example, if you select SAP S/4HANA Cloud as the source and the folder SQL_SERVICE as the container, you will only be shown CDS view entities that have the required annotations for data extraction.
- If you use SAP Datasphere as the source connection, your source objects must be local tables that have been deployed, are **not** enabled for delta capturing, and have a primary key.

4. If you decide that you do not want to include an object after all, select it and choose [Remove from Selection](#). If you want to include more objects, go back to the [Available](#) tab and select the relevant objects. When you are done, choose [Add Selection](#). The system then imports the object definitions so that they are available for the subsequent process steps.

Note

An object can only be included in one replication flow at any given point in time (not multiple ones).

Results

You have all necessary information about your data source in place and can move on to add your target.


5.6.2 Add a Target

Select a target (connection and container) to define the target environment for your replication flow.

Context

- Connections are created by your system administration. You can only use a target if a connection has been created for it in your space and if you have the necessary authorizations. For more information about connections and connection types, see [Integrating Data via Connections](#).
- Containers are the parent objects that hold the data.

Procedure

1. Choose  ([Browse target connections](#)). A list of available target connections appears.
2. Select the relevant one for your use case.
3. If you replicate to the local repository (SAP Datasphere), the target container is automatically defined as the space you are in, and the data is replicated to a local table. If you want to use load type *Initial and Delta*, the local table must support delta capturing. For more information, see [Capturing Delta Changes in Your Local Table \[page 118\]](#).

For any other target, click [Browse target container](#). A list of available containers appears. Select the relevant one for your use case.

To narrow down the selection, start typing a part of the container name in the [Search](#) field.

4. Review the target settings and properties and change or complete them as appropriate.

For more information, see [Configure Your Replication Flow \[page 160\]](#).

5.6.3 Configure Your Replication Flow

Define settings and properties for your replication flow and individual replication objects.

Procedure

1. Go to the [Settings](#) tab of the canvas to review the general settings (load type and truncate) and change them as appropriate.

Alternatively, you can select the relevant replication object and review its settings and properties in the side panel.

2. Select the relevant load type:
 - **Initial Only**: Load all selected data once.
 - **Initial and Delta**: After the initial load, the system checks for source data changes (delta) at regular intervals and copies the changes to the target. The default value for the delta load interval is 60 minutes. You can change it in the side panel by entering an integer between 0 and 24 for hours and 0 and 59 for minutes, respectively. The maximum allowed value is 24 hours 0 minutes. If you enter 0 hours and 0 minutes, the system replicates any source changes immediately.

Note


- A replication flow that contains objects with load type **Initial and Delta** does not have an end date. Once started, it remains in status **Running** until it is stopped or paused or an issue occurs.
- The system load caused by the delta load operations can vary substantially depending on the frequency of changes in your data source in combination with the interval length you define. Make sure that your tenant configuration supports your settings. For more information, see [Configure the Size of your SAP Datasphere Tenant](#).
- The next interval starts after all changes from the previous interval have been replicated. For example, if replicating a set of changes starts at 10:30 a. m. and takes until 10:45 a. m., and you have defined one-hour intervals, the next delta replication starts at 11:45 a. m.

For more information about using this load type in connection with local tables, see [Capturing Delta Changes in Your Local Table](#).


3. The **Truncate** setting is relevant if the target structure already exists and contains data. Review the default setting and change it if required:
 - If **Truncate** is marked for a **database table**, when you start the replication run, the system deletes the table content, but leaves the table structure intact and fills it with the relevant data from the source. If not, the system inserts new data records after the existing data in the target. For data records that already exist in the target and have been changed in the source, the system updates the target records with the changed data from the source using the UPSERT mode.
 - For cloud storage provider targets, **Truncate** must always be set. (If you still try to run a replication flow for an existing target without the **Truncate** option, you get an error message.) When you start the replication run, the system deletes the object completely (data and structure) and re-creates it based on the source data.

- For Apache Kafka and Confluent Kafka, when *Truncate* is enabled, the target topic is re-created. This means that all existing records in that topic are deleted as well. Truncation has no effect on the schema registry.

If the target structure does not yet exist or is empty, you can ignore the *Truncate* setting.

4. Click  ([Browse target settings](#)) to review the target settings **at replication flow level** and change them as appropriate.

- Replication Thread Limit: To modify the throughput, you can increase or decrease the default value for the number of replication threads as appropriate for your use case.

You can change the thread limit in the source accordingly by clicking  ([Browse source settings](#)) and entering the appropriate value. The recommendation is to use the same value for the source and the target.

See also [Replication Flow Blog Series Part 4 - Sizing](#).

- Overwrite Target Settings at Object Level: [only relevant if the target is a cloud storage provider] By default, any settings that you have made at replication object level are kept intact if you make a different setting at replication flow level. To change this, enable this option.
- Additional settings that are only relevant for a specific target type and can be made for the replication flow itself as well as for individual replication objects. For more information, see
 - [Using a Cloud Storage Provider As the Target \[page 163\]](#)
 - [Using Google BigQuery As the Target \[page 165\]](#)
 - [Using Apache Kafka As the Target \[page 168\]](#).

5. Review the settings for each replication object and change or complete them as appropriate.

To do so, select the relevant replication object. Its properties are then displayed in the side panel.

For a list of properties and further information, see [Creating a Replication Flow \[page 155\]](#).

5.6.4 Define Filters

Define filters to delimit the scope of your replication flow.

Procedure

1. Select the source object for which you want to define a filter.
2. Choose [Add Projection](#).
3. On the *Filter* tab of the [Projections](#) dialog, select the column by which you want to filter from the list on the left.

The key symbol next to a column name indicates a key field.

4. Select the relevant condition and enter the relevant values.

What options you have here depends on the replication objects you choose and their respective data types.

For a string field, for example, you can only choose *equal to* and enter a value. For an integer field, you can use mathematical operators, such as *greater than*, and enter a numeric value.

5. When you're done, choose [Add Expression](#).
6. Repeat steps 3 to 5 until you have defined all filters you need.
 - More than one filter on different columns applies the AND operator.
 - More than one filter on the same column applies the OR operator.
 - The line at the bottom of the screen summarizes how all the filter expressions you defined work together.

For example, filtering on the product ID 123 for the countries United States and Germany results in the following:

(PRODUCT_ID = 123) AND (COUNTRY = 'US' OR COUNTRY = 'DE')

7. Enter a name for your filter at the top of the screen, then click [OK](#).

5.6.5 Define Mapping

Create mappings to specify how the source data is to be changed on its way into the target.

Procedure

1. Select the source object for which you want to define mapping.
2. Choose [Add Projection](#).
3. Go to the [Mapping](#) tab of the [Projections](#) dialog and enter a name for your mapping.
4. Enter your mapping. You have the following options:
 - Change the name of a target column: By default, the system uses the name of the corresponding source column as the target column name. To change this, overwrite the default name with the name you want to use.
 - Change the data type: To change the data type, select the relevant one from the dropdown list of data types. Where applicable, you can make further changes. For data type 'string', for example, you can change the length.
 - Change the source mapping: Choose the relevant entry from the dropdown list of source columns.
 - Edit the target content: Mark [Edit](#) for the relevant row and enter the content you want to see. All fields of the target column are then automatically filled with the content you enter here.

ⓘ Note

- The [Primary Key](#) column is for information purposes. You cannot change whether a column is part of the primary key or not.
- For datetime columns, selecting [Edit Content](#) will auto-fill the fields with the following function:
 - Date - CURRENT_UTCDATE
 - Time - CURRENT_UTCTIME
 - Timestamp - CURRENT_UTCTIMESTAMP
- Add a new column: Choose [Add](#) and enter the necessary values for the new column.

- Remove a column: Select the relevant column and choose [Remove](#).

📌 Note

It is not possible to remove primary key columns or to map a key column in the source to multiple non-key columns in the target.

- Change the column sequence: Select the column that you want to move to a different place and use the [Up](#) and [Down](#) buttons.
- Restore default settings: To discard any changes and restore the default mapping settings, choose [Auto-Map](#).

⚠️ Caution

If you choose [Auto-Map](#), the system overwrites **all** mapping changes you made for this object with the default values.

5.6.6 Using a Cloud Storage Provider As the Target

If you use a cloud storage provider as the target for your replication flow, you need to consider additional specifics and conditions.

📌 Note

You can only use a non-SAP target for a replication flow if your admin has assigned capacity units to Premium Outbound Integration. For more information, see [Configure the Size of Your SAP Datasphere Tenant](#).

This topic contains the following sections:

- [Available Targets \[page 163\]](#)
- [Additional Properties \[page 164\]](#)
- [Files \[page 164\]](#)


Available Targets

The following cloud storage providers can be used as the target of a replication flow.

- Data lake Files from SAP HANA Cloud, data lake
- Amazon Simple Storage Service
- Google Cloud Storage
- Microsoft Azure Data Lake Gen2

For more information about the corresponding connection types, see [Integrating Data via Connections](#).

Additional Properties

In addition to the properties that apply to all targets (see the list in [Creating a Replication Flow \[page 155\]](#)), there are the following properties that are only relevant for cloud storage providers. You can review and change these properties for the replication flow itself (by choosing  ([Browse target settings](#))) or for individual replication objects (by selecting an object so that its properties get displayed in the side panel).

Property	Description
Group Delta by	[only relevant for load type Initial and Delta] Select <i>None</i> [default], <i>Date</i> , or <i>Hour</i> to specify whether you want to create additional folders for sorting updates based on the date or hour.
File Type	Select the file type. You can choose between CSV, JSON, JSONLines, and Parquet [default]. For JSON and JSON Lines, generated files are encoded in UTF-8 format.
Enable Apache Spark Compatibility	[only relevant for file type Parquet] Enable this option to convert and store time data type columns to int64 in the Parquet files. The int64 data type represents microseconds after midnight. This conversion allows the columns to be consumed by Apache Spark.
File Compression	[only relevant for file type Parquet] Select the file compression type. You can choose between None [default], Gzip, and Snappy.
Suppress Duplicates	Enable this option to avoid duplicate records in your target file. During initial load, if a data record already exists in the target, the default system behavior with cloud storage provider targets would be to write this record to the target once again, which results in duplicate records. However, this is not the desired behavior for many use cases of replication flows, and consequently duplicate suppression is active by default. You can deactivate it if required for your specific use case (for example to continue using existing replication flows that do not support duplicate suppression).
Delimiter	[only relevant for file type CSV] Select the character you want to use to separate the columns from each other. You can choose between Comma [default], Colon, Pipe, Semicolon, and Tab.
Header Line	[only relevant for file type CSV] Select True [default] or False to specify whether you want the file to have a header line or not.
Orient	[only relevant for file type JSON] Select the internal structure for the generated JSON files. Currently the only available option is Records: [{column -> value}, ... ,{column -> value}]

Files

Running a replication flow with a cloud storage target creates various files and structures:

Upon completion of the initial load, the system writes a `_SUCCESS` file. Downstream applications that can access the object store directly can use this file to verify that the replication completed successfully without having to check the replication flow status.

The `.sap.partfile.metadata` objects include metadata information for the replication object. It exists in the root of each data file directory and is created and referenced as part of replication flow processing. Additionally, you can leverage these objects for interpreting and processing the data files.

The replication flow creates multiple files (`part-*.<extension>`) during initial and delta loading. The number and size of these files depends on the source table size and structure as well as change frequency (during delta loading).

Each file contains the source columns as defined in the mapping for the replication object in the replication flow. The system appends the following columns:

- `__operation_type`: Identifies the type of target row:
 - `L`: Written as part of the initial load.
 - `I`: After the initial load completed, new source row added.
 - `U`: After the initial load completed, after image of an update to a source row.

Note

For some sources, the system switches the value `U` to `A` after you apply SAP Note [3044005](#). The `APE_KEEP_UPDATE_OPERATION` parameter is described in the SAP Note.

- `B`: After the initial load completed, before image of an update to a source row. These records are only sent by some sources (like SAP HANA) and only when the after image of the update is not passing the filters specified in the replication task.
- `X`: After the initial load completed, source row deleted. The only target columns to contain data for this operation code are codes that reflect the source key columns. All other target columns are empty.
- `M`: After the initial load completed, archiving operations.
- `__sequence_number`: An integer value that reflects the sequential order of the delta row in relation to other deltas. This column is empty for initial load rows and is not populated for all source systems (for example, ABAP).
- `__timestamp`: The UTC date and time the system wrote the row.

5.6.7 Using Google BigQuery As the Target

If you use Google BigQuery as the target for your replication flow, you need to consider the following additional specifics and conditions.

Note

You can only use a non-SAP target for a replication flow if your admin has assigned capacity units to Premium Outbound Integration. For more information, see [Configure the Size of Your SAP Datasphere Tenant](#).

This topic contains the following sections:

- [General Properties \[page 166\]](#)
- [Target Tables \[page 166\]](#)
- [Target Columns \[page 166\]](#)
- [Data Types \[page 167\]](#)
- [Primary Key \[page 167\]](#)
- [SQL Statement \[page 167\]](#)

General Properties

The system always uses **write mode** *Append*.

The *Truncate* setting is not available.

Target Tables

Target table names may only contain the following special characters:

- Hyphen (-)
- Underscore (_)
- Space ()

The maximum length for target column names is 300 characters.

If the **target structure already exists** in Google BigQuery, you cannot make changes such as renaming a column or changing a data type in SAP Datasphere. You need to do this directly in Google BigQuery.

Target Columns

The system automatically adds the following columns to the **schema of the target table**:

- OPERATION_FLAG
- IS_DELETED
- RECORDSTAMP

These columns are needed to capture changes in the source so that they can be replicated to the target, and you cannot change their names or settings.

In the following cases, target column names have to be changed so that they can be used for replication into Google BigQuery. You can change the names for the relevant target columns manually or choose *Auto-Projection*.

- If a target column has the same name as one of the columns for change data capturing, the target column has to be renamed. Auto-Projection does this by adding the prefix AUTOPREFIX_ to the name of the target column, for example AUTOPREFIX_RECORDSTAMP.
- Some characters, such as slash (/), cannot be used in column names in Google BigQuery. If a target column name contains any of these characters, Auto-Projection replaces the unallowed characters with an underscore (_).
- Some column name prefixes are reserved within Google BigQuery, for example _TABLE_, _FILE_, or _PARTITION_. If a target column name contains any of these prefixes, Auto-Projection adds AUTOPREFIX_ in front of the existing name.

Data Types

Google BigQuery does not support the data types DECFLOAT16, DECFLOAT34 and UINT64. The system automatically converts DECFLOAT16 and DECFLOAT34 into DECIMAL(38,9) and UINT64 into DECIMAL(20,0). The first value in brackets is the precision (total number of digits), the second one is the scale (number of digits after the decimal point).

For the DECFLOAT data types, the following conditions apply:

- Precision can be between 38 and 77.
- Scale can be between 9 and 38.
- Precision minus scale must be 29 or larger.
- By default, values that are too large are clamped in accordance with the maximum values for the respective target data type.

If you don't want this, you can deactivate clamping for individual replication objects (by deselecting *Clamp Decimal Floating-Point Data Type* in the *Details* side panel) or for all objects in the replication flow (by deselecting *Clamp Decimal Floating-Point Data Type* in the target settings). If you do so, and if there are values that are too large, the replication fails and you get an error message informing you about the issue.

For data type UINT64, the number of digits before the decimal point (precision minus scale) must be 20 or greater.

Primary Key

A **primary key** can be created in Google BigQuery if the following prerequisites are met:

- There are no more than **16 key columns**.
- All key columns have one of the following **Google BigQuery data types**:
 - BIGNUMERIC
 - BOOLEAN
 - DATE
 - DATETIME
 - INT64
 - NUMERIC
 - STRING
 - TIMESTAMP

If either of these prerequisites is not met and you still run a replication flow, **none** of the columns in the target gets defined as a primary key column.

SQL Statement

You can view and copy the **SQL Create Table statement** of a replication object so that you can modify and run it directly in Google BigQuery. This can be useful, for example, if you want to change partitions and clusters for

a new target object. To do so, choose View SQL Create Table Statement or Copy SQL Create Table Statement from the context menu for the relevant target object.

5.6.8 Using Apache Kafka As the Target

If you use Apache Kafka as the target for your replication flow, you need to consider the following additional specifics and conditions.

Note

You can only use a non-SAP target for a replication flow if your admin has assigned capacity units to Premium Outbound Integration. For more information, see [Configure the Size of Your SAP Datasphere Tenant](#).

Each record from the source system is transferred into a single **message** in the target topic. The key of the messages is the combination of all primary key values of the record concatenated by "_".

Schema registries are **not** supported. If you choose AVRO as the serialization format, the schema is contained in every message. For the JSON serialization format, no schema information is provided.

The **target container** is automatically set to "/" because Kafka does not have a superordinate container layer.

Note

If the Kafka cluster is behind an SAP Cloud Connector (SCC), the Kafka cluster and the SCC must be configured such that the broker addresses advertised by the cluster match the virtual hosts maintained for the brokers in the SCC. The simplest solution is to use the same value for virtual and internal hosts in the SCC and to maintain no dedicated advertised listeners for the Kafka brokers. If advertised listeners are maintained, these must be used as virtual hosts in SCC and as broker addresses in the connection definition.

The following decimal floating point and fixed point decimal values may be **clamped** according to the data type used in the target serialization:

- decfloat16 and decfloat34 are clamped to DECIMAL(28,6) and DECIMAL(38,6), respectively, for AVRO and to double range (set to +-inf) for JSON.
- DECIMAL(p,s) is clamped to double range (set to +-inf) for JSON.

You can **rename** target objects (Kafka topics). The following conditions apply:


- The new name may consist of the following characters: small latin letters (a-z), capital latin letters (A-Z), numbers (0-9), period (.), underscore (_), hyphen (-).
- The maximum length for the new name is 249 characters.

For more information and examples, see also [SAP Datasphere Replication Flows Blog Series Part 3 – Integration with Kafka](#).

This topic contains the following sections:

- [Additional Properties \[page 169\]](#)
- [Additional Message Headers \[page 169\]](#)

Additional Properties

In addition to the properties that apply to all targets (see the list in [Creating a Replication Flow \[page 155\]](#)), there are the following properties that are only relevant for Apache Kafka. You can review and change these properties for the replication flow itself (by choosing  ([Browse target settings](#))) or for individual replication objects (by selecting an object so that its properties get displayed in the side panel).

Property	Description
Number of Partitions	Enter the number of Kafka partitions to be used for the Kafka topic.
Replication Factor	Enter the Kafka replication factor to be used for the Kafka topic.
Message Encoder	Select the message format for the Kafka topic You can choose between AVRO and JSON [default]. If you select AVRO, the column name must consist of only alphanumeric and underscore characters, and it must start with a letter or an underscore.
Message Compression	Select the message compression type. You can choose between No Compression [default], Gzip, Snappy, LZ4, and ZStandard.

Additional Message Headers

Each topic contains the source columns as defined in the mapping for the replication object in the replication flow. The system appends the following columns:

- *kafkaSerializationType*: AVRO or JSON
- *OpType*: Identifies the type of target row:
 - *L*: Written as part of the initial load.
 - *I*: After the initial load completed, new source row added.
 - *U*: After the initial load completed, after image of an update to a source row.

Note

For some sources, the system switches the value *U* to *A* after you apply SAP Note [3044005](#). The APE_KEEP_UPDATE_OPERATION parameter is described in the SAP Note.

- *B*: After the initial load completed, before image of an update to a source row. These records are only sent by some sources (like SAP HANA) and only when the after image of the update is not passing the filters specified in the replication task.
- *X*: After the initial load completed, source row deleted. The only target columns to contain data for this operation code are codes that reflect the source key columns. All other target columns are empty.
- *M*: After the initial load completed, archiving operations.
- *Seq*: Sequence number, an integer value that reflects the sequential order of the delta row in relation to other deltas. This column is empty for initial load rows and is not populated for all source systems (for example, ABAP).

5.6.9 Using Confluent Kafka As the Target

If you use Confluent Kafka as the target for your replication flow, you need to consider the following additional specifics and conditions.


Note

You can only use a non-SAP target for a replication flow if your admin has assigned capacity units to Premium Outbound Integration. For more information, see [Configure the Size of Your SAP Datasphere Tenant](#).

This topic contains the following sections:

- [Additional Properties \[page 170\]](#)
- [Further Information \[page 171\]](#)

Additional Properties

In addition to the properties that apply to all targets (see the list in [Creating a Replication Flow \[page 155\]](#)), there are the following properties that are only relevant for Confluent Kafka. You can review and change these properties for the replication flow itself (by choosing  ([Browse target settings](#))) or for individual replication objects (by selecting an object so that its properties get displayed in the side panel).

Property	Description
Number of Partitions	Enter the number of Confluent partitions to be used for the Confluent topic. The default value is 1.
Replication Factor	Enter the Confluent replication factor to be used for the Confluent topic. The default value is 3.
Serialization Format	Select the serialization format for the Confluent topic. You can choose between AVRO [default] and JSON. If you select AVRO, the column name must consist of only alphanumeric and underscore characters, and it must start with a letter or an underscore.
Compression Type	Select the message compression type. You can choose between No Compression [default], Gzip, Snappy, LZ4, and ZStandard.
Use Schema Registry	If this option is enabled, the schema of the sent messages is registered in the provided schema registry. If it is disabled, the messages do not contain any schema information or schema-id. Using no schema registry is only allowed for serialization format JSON.

Property	Description
Record Name	[only relevant if Use Schema Registry is enabled] Enter the record name to be used in the schema that is registered in the schema registry. If you choose serialization type AVRO, the record name must consist of only alphanumeric and underscore characters, and it must start with a letter or an underscore.
Subject Name Strategy	[only relevant if Use Schema Registry is enabled] Select the strategy for building the subject name under which the schema will be registered in the schema registry. You can choose between Topic [default], Record, and Topic-Record.
Compatibility Mode	[only relevant if Use Schema Registry is enabled] Select the compatibility mode for the created subject in the schema registry. If the subject already exists and a value other than DEFAULT is specified, the compatibility mode of the subject is changed to the provided value.
Clamp Decimal Floating Point Data Types	Clamping is enabled by default and cannot be switched off. For more information, see Using Confluent Kafka As the Target [page 170] .

Further Information

The **target container** is automatically set to "/" because Confluent Kafka does not have a superordinate container layer.

Note

If the Kafka cluster is behind an SAP Cloud Connector (SCC), the Kafka cluster and the SCC must be configured such that the broker addresses advertised by the cluster match the virtual hosts maintained for the brokers in the SCC. The simplest solution is to use the same value for virtual and internal hosts in the SCC and to maintain no dedicated advertised listeners for the Kafka brokers. If advertised listeners are maintained, these must be used as virtual hosts in SCC and as broker addresses in the connection definition.

Each record from the source system is transferred into a single **message** in the target topic. The key of the messages is the combination of all primary key values of the record concatenated by "_".

Clamping is enabled on by default and cannot be switched off. The following decimal floating point and fixed point decimal values may be **clamped** according to the data type used in the target serialization:

- decfloat16 and decfloat34 are clamped to DECIMAL(28,6) and DECIMAL(38,6), respectively, for AVRO and to double range (set to +-inf) for JSON.
- DECIMAL(p,s) is clamped to double range (set to +-inf) for JSON.

Each topic contains the source columns as defined in the mapping for the replication object in the replication flow. The system **appends** the following **columns**:

- `__operation_type`: Identifies the type of target row:
 - `L`: Written as part of the initial load.
 - `I`: After the initial load completed, new source row added.
 - `U`: After the initial load completed, after image of an update to a source row.

Note

For some sources, the system switches the value *U* to *A* after you apply SAP Note [3044005](#). The `APE_KEEP_UPDATE_OPERATION` parameter is described in the SAP Note.

- *B*: After the initial load completed, before image of an update to a source row. These records are only sent by some sources (like SAP HANA) and only when the after image of the update is not passing the filters specified in the replication task.
- *X*: After the initial load completed, source row deleted. The only target columns to contain data for this operation code are codes that reflect the source key columns. All other target columns are empty.
- *M*: After the initial load completed, archiving operations.
- `__sequence_number`: An integer value that reflects the sequential order of the delta row in relation to other deltas. This column is empty for initial load rows and is not populated for all source systems (for example, ABAP).
- `__timestamp`: The UTC date and time the system wrote the row.

You can **rename** target objects (topics). The following conditions apply:

- The new name may consist of the following characters: small latin letters (a-z), capital latin letters (A-Z), numbers (0-9), period (.), underscore (_), hyphen (-).
- The maximum length for the new name is 249 characters.

5.6.10 Editing an Existing Replication Flow

You can change all settings for a replication flow as long as you haven't deployed it.

Once you have deployed it, you can no longer use the mapping function to add columns to your target structure. However you can still add columns manually using the table editor.

You can remove individual replication objects from a running replication flow and then deploy the flow again without stopping it first.


When a replication flow has run or is currently in status *Running*, you cannot change the load type anymore because data inconsistencies might occur if the source data gets changed while the load type is being changed. You need to stop and delete the existing replication flow and create a new one with the required load type.

However it is still possible to change the delta load interval for replication flows that already have load type *Initial and Delta*. To do so, stop the replication flow, change the interval as appropriate (in the side panel), deploy the flow, and run it. The replication flow run will then start all over again with an initial load.

For more information, see [Working With Existing Replication Flow Runs](#).

5.6.11 Deleting a Replication Flow

You can delete a replication flow if you do not need it anymore and thus free up capacity.

In the Data Builder, select the relevant replication flow and choose  (*Delete*).

You can only delete replication flows that have the status *Completed* or *Stopped*.

5.7 Creating a Transformation Flow

Create a transformation flow to load data from one or more source tables, apply transformations (such as a join), and output the result in a target table. You can load a full set of data from one or more source tables to a target table. You can add local tables and also remote tables located in BW Bridge spaces. Note that remote tables located in SAP BW bridge spaces must be shared with the SAP Datasphere space you are using to create your transformation flow. You can also load delta changes (including deleted records) from one source table to a target table.

Context

A transformation flow run is a one-time event that completes when the relevant data is loaded to the target table successfully. You can run a transformation flow multiple times, for example if you are loading delta changes to the target table.

Creating a transformation flow involves two important steps:

- Creating a graphical view transform or an SQL view transform to load data from source tables and to transform data.
- Adding a target table to the transformation flow and mapping the columns from the graphical view transform or SQL view transform created in the first step to the target table.


Note

A transformation flow only supports the loading of data to a local table in the SAP Datasphere repository.

Note

A transformation flow generates internal SQL objects that must only be consumed by SAP Datasphere internal apps, and are not allowed for external data access.


Procedure

1. In the side navigation area, click  (*Data Builder*), select a space if necessary, and click *New Transformation Flow* to open the editor. The system displays the *New Transformation Flow* screen.
2. Create a graphical view transform or an SQL view transform to load data from source tables and to transform data. To create a graphical view transform, click the *Graphical View Transform* button (see [Create a Graphical View Transform \[page 175\]](#)). To create an SQL view transform, click the *SQL View Transform* button (see [Create an SQL View Transform \[page 188\]](#)).
3. Click the *Back* button to return to the *Transformation Flow Editor*. Add or create a target table that the transformation flow will write its data to (see [Add or Create a Target Table \[page 191\]](#)).
4. Click the *Details* button to display the *Transformation Flow Properties* panel.


In the *General* section, you can view the following properties:

Property	Description
Business Name / Technical Name	Information to identify the transformation flow.
Package	<p>Select the package to which the object belongs.</p> <p>Packages are used to group related objects in order to facilitate their transport between tenants.</p> <div style="border: 1px solid #ccc; background-color: #f9f9f9; padding: 10px; margin: 10px 0;"> <p>Note</p> <p>Once a package is selected, it cannot be changed here. Only a user with the DW Space Administrator role (or equivalent privileges) can modify a package assignment in the <i>Packages</i> editor.</p> </div> <p>For more information, see Creating Packages to Export.</p>
Status	The deployment and error status of the object. For more information, see Saving and Deploying Objects [page 39] .
Load Type	<p>Select the relevant load type.</p> <p>The following options are possible:</p> <ul style="list-style-type: none"> • Initial Only Loads the full set of data to the target table. • Initial and Delta The first time you run the transformation flow, the system will load all the full set of data to the target table. For subsequent runs, the system will only load delta changes to the target table. If you want to load delta changes from a source table, you need to ensure that the value of the option <i>Delta Capture</i> is <i>On</i> for both the source table and for the target table.
Run Status	<p>View the details of the last run.</p> <p>You can view the details of the last run. You can view the date and time of the run as well as the status, for example <i>Completed</i>. In addition, you can choose the <i>Schedule</i> button to run your transformation flow at later time, or on regular basis. For more information, see Running a Flow [page 196]. You can also open the transformation flow run in the <i>Data Integration Monitor</i>.</p>

In the *Run Status* section, you can view the status of the transformation flow run.

- Click  (*Save*). A dialog box appears. Enter a business name and a technical name for your transformation flow.

When you enter a business name, the system automatically suggests a corresponding technical name, but you can change the technical name if required. Both names must be unique within the space you work in.

- Click  (*Deploy*) to make your transformation flow ready to run.
 - Newly created transformation flows are deployed for the first time.
 - Transformation flows that have changes to deploy are redeployed.

5.7.1 Create a Graphical View Transform









Create a graphical view transform that uses SQL syntax to combine and transform data as part of your transformation flow. You can drag and drop source tables from the *Repository*, join them as appropriate, add other operators to remove or create columns and filter or aggregate data.


Procedure

1. On the *New Transformation Flow* screen, click the *View Transform* node.
2. Create a new *Graphical View Transform* for your transformation flow by clicking the *Graphical View Transform* button.
3. The system displays the *Graphical View Editor*. Add a source table. For more information, see [Add a Source Table \[page 176\]](#).
4. Optional: Drag additional source tables from the repository and drop them onto a source table to create a join or union. For more information, see [Create a Join \[page 178\]](#) and [Create a Union \[page 179\]](#).
5. Click a node in the diagram to display tools for creating operators and performing other actions:

Note

You can only create one each of the *Filter*, *Projection*, *Calculated Columns*, and *Aggregation* operators per source or join in your diagram. Operators can be created in any order.

Tools	Description
 (<i>Filter</i>)	Add a <i>Filter</i> node to filter your data with an SQL expression. For more information, see Filter Data [page 236] .
 (<i>Rename/Exclude Columns</i>)	Add a <i>Projection</i> node to rename, reorder, or exclude columns. For more information, see Reorder, Rename, and Exclude Columns [page 223] .
 (<i>Calculated Columns</i>)	Add a <i>Calculated Columns</i> node to create new columns and define calculations in them. For more information, see Create a Column [page 224] .
 (<i>Aggregation</i>)	Add an <i>Aggregation</i> node to perform SUM, COUNT, MIN, and MAX calculations. For more information, see Aggregate Data [page 185] .
 (<i>Join Suggestion</i>)	Create a join from a list of <i>Related Entities</i> that is populated based on the presence of associations between the current source and other artifacts.
 (<i>Preview Data</i>)	Preview the data output by the selected diagram node in the <i>Data Preview</i> panel (see Viewing or Previewing Data in Data Builder Objects [page 297]). You can preview the SQL generated for the node by clicking the <i>Preview SQL</i> button in the panel or by clicking  <i>Export</i> and selecting <i>Preview SQL</i> . Click <i>Copy</i> to copy the SQL code for pasting into the SQL View editor or elsewhere.
 (<i>Impact and Lineage Analysis</i>)	Open the Impact and Lineage Analysis diagram. This diagram enables you to understand the lineage and impacts of the selected object. (see Impact and Lineage Analysis [page 34] .)

Tools	Description
 (Open in New Tab)	Open the object in its own editor in a new tab.

- Click the [Back](#) button to save your work and return to the [Transformation Flow Editor](#). You can then add or create a target table for the transformation flow. For more information, see [Add or Create a Target Table \[page 191\]](#).

5.7.1.1 Add a Source Table

Add a source table to read data from. You can add multiple source tables and combine them together using join or union operators.

Procedure

- Navigate to the [Graphical View Editor](#).
- Drag a table from the [Repository](#) and drop it onto the diagram.
- Click the source table node to display its properties in the side panel, and review the properties in the [General](#) section:

Property	Description
Business Name / Technical Name	Information to identify the source table.
Alias	You can specify an SQL alias for the table if required.
Distinct Values	Indicates whether the source table uses SELECT DISTINCT statements to return only distinct values.
Delta Capture	Indicates whether the delta capture setting is enabled for a table. For more information, see Capturing Delta Changes in Your Local Table [page 118] .
Status	The deployment and error status of the object. For more information, see Saving and Deploying Objects [page 39] .

If the delta capture setting is enabled for a table, you can view the [Delta Settings](#) section. By default, the option [Delta Capture](#) is selected. This option loads delta changes to the target table. If required, you can select the option [All Active Records](#) to load the full set of data to the target table.

Additional Information About Loading Delta Changes

To load delta changes from a source table (including deleted records), the delta capture setting must be enabled for the table. It is only possible to load delta changes from one source table. If you want to load delta changes from a source table, you need to ensure that the value of the option [Delta Capture](#) is [On](#) for

both the source table and for the target table. In the *Transformation Flow Properties* panel, ensure that the load type *Initial and Delta* is selected.

4. In the *Columns* section, you can view the columns of the source table. To check the data type of a column, hover over it and click ⓘ.

If the delta capture setting is enabled for a table, two additional columns are present in the table to track changes. For more information, see [Capturing Delta Changes in Your Local Table \[page 118\]](#).

ⓘ Note

If the delta capture setting is enabled for a source table, the columns *Change Date* and *Change Type* are automatically mapped to these columns in the target table. Mapping these columns (or a calculated column that contains the content of these columns) to any other target column is not permitted. For more information, see [Capturing Delta Changes in Your Local Table \[page 118\]](#).

ⓘ Note

The *Change Type* column does not support null values. Ensure that no null values are written to the *Change Type* column of the target table.

Example

You add two source tables. The delta capture setting is enabled for one table and disabled for the other. You create a union for these two tables. In the target table, the *Change Type* column will contain null values. In this case, running the transformation flow will result in errors.

5.7.1.1.1 Additional Information About Remote Tables

You can add remote tables located in SAP BW bridge spaces to a transformation flow. Note that remote tables located in SAP BW bridge spaces must be shared with the SAP Datasphere space you are using to create your transformation flow.

ⓘ Note

If you add a remote table located in SAP BW bridge spaces, it is not possible to preview the data being output by the view.

If you add a remote table located in an SAP BW bridge space that is configured to load delta changes, the system adds the following input parameters to the *Properties* panel of the view transform:

- REQTSN_LOW
When loading delta changes, the value of this input parameter is the watermark. When loading all active records, the value of this input parameter is the minimum timestamp. For more information, see [Watermarks](#).
- REQTSN_HIGH
The value of this input parameter is the maximum timestamp.
- EXTRACTION_MODE
The value of this input parameter is DELTA_AI to indicate the loading of delta changes from the remote table.

5.7.1.2 Create a Join


Use a join operator to merge two data sets together using a join definition to match the records.

Procedure

1. Select a source table in the *Repository*, and then drag it over a source table in the diagram. Wait for the context menu to appear, slide your cursor over the *Join* option and then release the mouse button.
2. Optional. Rename the node in its side panel to clearly identify it. This name can be changed at any time and can contain only alphanumeric characters and underscores.
3. Set the following properties in the *General* section:

Property	Description
Join Type	<p>Specifies the type of SQL join to create. You can choose from:</p> <ul style="list-style-type: none">• Cross - Return all of the rows from the left table joined to all of the rows from the right table.• Full - Return all of the rows from both tables, joining records from the left table where possible.• Inner - [default] - Return all of the rows in the left table that have a matching row in the right table.• Left - Return all of the rows from the left table (whether or not they have a match in the right table), and any matching rows from the right table.• Right - Return all of the rows from the right table (whether or not they have a match in the left table), and any matching rows from the left table.
Distinct Values	Return only unique values.
Cardinality	<p>Specifies the number of rows matching another table in a join and may improve the view's performance. You can choose from:</p> <ul style="list-style-type: none">• MANY TO ONE• MANY TO EXACT ONE• MANY TO MANY• ONE TO ONE• ONE TO EXACT ONE• ONE TO MANY• EXACT ONE TO ONE• EXACT ONE TO EXACT ONE• EXACT ONE TO MANY <p>The cardinality is integrated in the code and is visible in the SQL preview.</p>

4. Specify the mapping of join columns in the *Mappings* section:
 - A mapping is automatically created by matching column names if possible.
 - To manually map columns, drag a column from the left list and drop it onto a column in the right list.

- To delete a mapping, click the link and then click the *Delete* tool.
 - You can filter the *Mappings* section to show only mapped or unmapped pairs of columns.
 - You can filter or sort the left or right column lists independently
5. Click  (*Preview Data*) to open the *Data Preview* panel and review the data output by this node. For more information, see [Viewing or Previewing Data in Data Builder Objects \[page 297\]](#).

5.7.1.3 Create a Union

A union combines the results from two select statements on separate sources.

Procedure


1. Select a source table in the *Repository*, and then drag it over a source table in the diagram. Wait for the context menu to appear, slide your cursor over the *Union* option and then release the mouse button.

The source table is added to the diagram and it is unioned with the target node. The union symbol is selected and its properties are displayed in the side panel. You can add multiple tables to the union by dragging and dropping them on the union symbol.

2. Optional. Rename the node in its side panel to clearly identify it. This name can be changed at any time and can contain only alphanumeric characters and underscores.
3. Set the following properties in the *General* section:

Property	Description
Union All	Default option and fastest to create. It combines two or more <code>SELECT</code> statements or queries and includes all rows, including duplicates. Disabling this option applies <i>Union</i> .
Union	It combines the result set of two or more <code>SELECT</code> statements or queries (only distinct values) and returns fewer rows. It takes longer to create it because it removes duplicate rows.


4. The *Mappings* section shows, by default, the recently dropped source object columns on the left and the union output columns, initialized from the target source columns on the right.
 - Mappings are automatically created by matching column names where possible.
 - To manually map input columns to union output columns, drag a column from the left list and drop it onto a column in the right list.
 - To delete a mapping, click the link and then click the *Delete* tool.
 - You can filter the *Mappings* section to show only mapped or unmapped pairs of columns.
 - You can filter or sort the left or right column lists independently.
 - You can modify the list of union output columns using the Menu above the right list:
 - *Add All Source Columns as Union Columns* - Add all columns in the left list to the right list (if they are not already present).
 - *Add Selected Source Columns as Union Columns* - Add columns selected in the left list to the right list (if they are not already present).

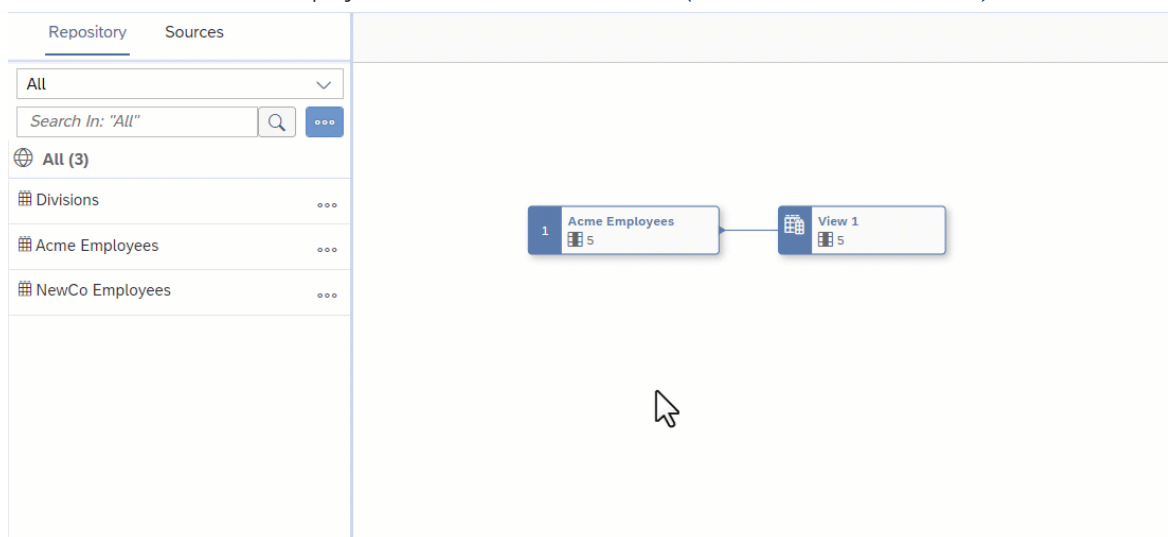
- [Delete Selected Union Columns](#) - Delete any columns selected in the right list so that they are no longer union output columns.
 - [Delete All Union Columns](#) - Delete all columns in the right list so that they are no longer union output columns.
 - To switch to viewing another input, click the drop-down arrow above the left column list and select it in the list.
5. Optional. Drop another source object onto the union node to union it with the existing unioned sources. The source table is added to the diagram and added to the union node. The union symbol is selected, its properties are displayed in the side panel, the new source table is displayed in the [Mappings](#) section and its columns are mapped automatically by matching column names where possible.
 6. Click  ([Preview Data](#)) to open the [Data Preview](#) panel and review the data output by this node. For more information, see [Viewing or Previewing Data in Data Builder Objects \[page 297\]](#).

5.7.1.4 Reorder, Rename, and Exclude Columns

Add a [Projection](#) node to rename, reorder, or exclude columns.

Procedure

1. Select a node in order to display its context tools, and click  ([Rename/Exclude Columns](#)).



- A projection node is created, its symbol is selected, and its properties are displayed in the side panel.
2. Optional. Rename the node in its side panel to clearly identify it. This name can be changed at any time and can contain only alphanumeric characters and underscores.
 3. The [Columns](#) list displays the columns that are available. You can filter the list with the [Search](#) box and reorder it.
 4. Modify the [Columns](#) list as appropriate. You can:
 - Reorder a column by dragging and dropping it in the list below [Columns](#).

- Rename a column by clicking **⋮** (*Menu*), and selecting *Change Name* to open the *Change Name* dialog. Edit the business and/or technical names as appropriate and click *Save*.

Note

By default, SAP Datasphere displays the business names of your objects. To show technical names instead, click **► Profile ► Settings ►**, select the *UI Settings* tab, and then select *Show Technical Name*.

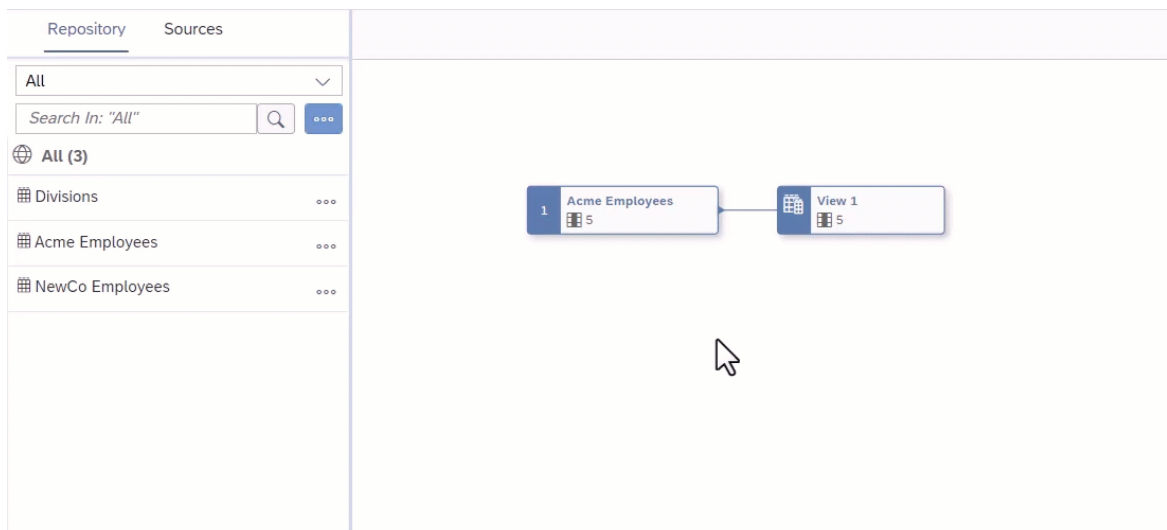
- Exclude a column by clicking its **⋮** (*Menu*) button, and selecting *Exclude Column*. To select multiple columns for exclusion, press **Ctrl** while selecting them. Alternatively, click *Select All* and press **Ctrl** while deselecting columns.
5. Click **🔍** (*Preview Data*) to open the *Data Preview* panel and review the data output by this node. For more information, see [Viewing or Previewing Data in Data Builder Objects \[page 297\]](#).

5.7.1.5 Create a Column

Add a *Calculated Columns* node to create new columns and define calculations in them.

Procedure

1. Select an object in order to display its context tools, and click **fx** *Calculated Columns*.



A calculated column node is created, its symbol is selected, and its properties are displayed in the side panel.

2. Click **+** (*Add New Calculated Column*) to create a new column. The new column properties are displayed in the side panel.
3. Enter a *Business Name* and a *Technical Name* for the column, and specify its *Data Type*.
4. Enter a SQL expression into the *Expression* field. You can use the following items in your SQL expression:

- *Insert Values* - Click to open the *Insert Value* dialog, select the values you need to insert in the list, and click *Insert Value*.

Note

- The button is enabled when, in the *Expression* field, a column name is followed by the operator =, >, <, >=, <=, !=, IN, BETWEEN, or LIKE.
- The values listed in the dialog are retrieved from the sources and intermediate nodes of your object. You can insert values of the data types string, integer, boolean, date, and time. The data types binary and UUID aren't supported.
- Once your expression is valid, additional values cannot be inserted. You must edit your expression to make it incomplete; this enables the *Insert Values* button and allows you to insert other values.
- The *Insert Values* dialog doesn't format the expression. Manual formatting may be required after inserting values.

Example

- The expression `Column1 =` enables the *Insert Values* dialog. You can select a single value from the list.
- The expression `Column2 BETWEEN` enables the *Insert Values* dialog. You can select two values from the list. If both values are available in a single search results, both of them can be selected together. However, if the values needs to be selected from two different search results, you have to select and insert one value at once.
- The expression `Column3 IN` enables the *Insert Values* dialog. You can select multiple values from the list. If all the required values are available in a single search results, those values can be selected together. However, if the values have to be selected from different search results, you have to select and insert one value at once. You select and insert the values 2, 5, 6, and 9. The expression `Column3 IN (2, 5, 6, 9)` is valid. You want to add another value to it, so you click *Insert Values*. In the *Insert Values* dialog, you can only see the values 2, 5, 6, and 9 because the dialog is already filtered by the selected values. To see all available values, edit your expression to make it invalid: `Column3 IN (2, 5, 6, 9, .` The *Insert Values* button is available again and you can add new values.

Select available value(s) and click *Insert* to add them to your expression.

- *Functions* - Browse, select a category, or filter available functions (see [SQL Functions Reference \[page 257\]](#)). Click a function name to see its syntax or click elsewhere in its token to add it to your expression.
- *Columns* - Browse or filter available columns. Click a column name to see its properties or click elsewhere in its token to add it to your expression.
- *Parameters* - Browse or filter available input parameters (see [Create an Input Parameter \[page 231\]](#)). Click a parameter name to see its properties or click elsewhere in its token to add it to your expression.
- *Other* - Browse available operators, predicates, and case expressions, and click one to add it to your expression (see [SQL Reference \[page 255\]](#)).

For example, if you want to calculate the price of a product minus a 15% discount, enter `Price*0.85`.

When working on a large expression, click  (*Enter Full Screen*) to expand the expression editor.

5. Click *Validate* to check if your expression is semantically correct, and fix any errors signaled. You can reference columns by name as well as HANA functions, operators, predicates, and case expressions. When

you are satisfied, click the breadcrumbs at the top of the side panel to drill back up to the calculated column node properties.

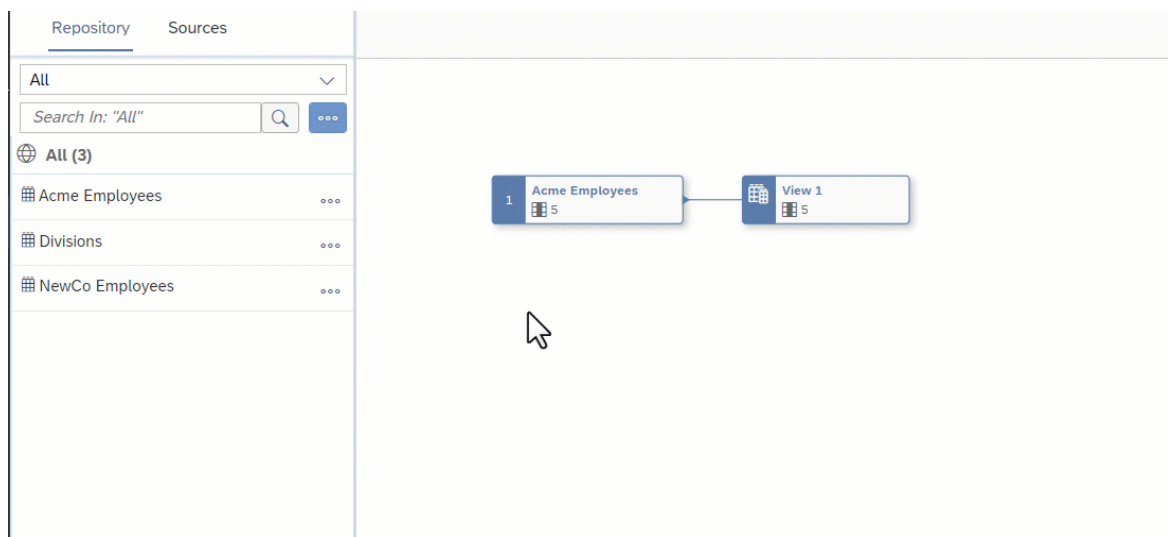
- The list displays all the columns output by the node. You can:
 - Filter the list with the *Search* box (or by clicking *Hide Unmodified Columns*) and reorder it.
 - Modify the expression output by any column by clicking on the chevron on the right of its token.
 - Delete a column that has been created in the node by clicking *Delete Selected Calculated Column*.
- Click (*Preview Data*) to open the *Data Preview* panel and review the data output by this node. For more information, see [Viewing or Previewing Data in Data Builder Objects \[page 297\]](#).

5.7.1.6 Filter Data

Add a *Filter* node to filter your data with an SQL expression.

Procedure

- Select an object in order to display its context tools, and click *Filter*.



A filter node is created, its symbol is selected, and its properties are displayed in the side panel.

- Optional. Rename the node in its side panel to clearly identify it. This name can be changed at any time and can contain only alphanumeric characters and underscores.
- Enter a SQL expression into the *Expression* field. You can use the following items in your SQL expression:
 - Insert Values* - Click to open the *Insert Value* dialog, select the values you need to insert in the list, and click *Insert Value*.

Note

- The button is enabled when, in the *Expression* field, a column name is followed by the operator `=`, `>`, `<`, `>=`, `<=`, `!=`, `IN`, `BETWEEN`, or `LIKE`.

- The values listed in the dialog are retrieved from the sources and intermediate nodes of your object. You can insert values of the data types string, integer, boolean, date, and time. The data types binary and UUID aren't supported.
- Once your expression is valid, additional values cannot be inserted. You must edit your expression to make it incomplete; this enables the *Insert Values* button and allows you to insert other values.
- The *Insert Values* dialog doesn't format the expression. Manual formatting may be required after inserting values.

❖ Example


- The expression `Column1 =` enables the *Insert Values* dialog. You can select a single value from the list.
- The expression `Column2 BETWEEN` enables the *Insert Values* dialog. You can select two values from the list. If both values are available in a single search results, both of them can be selected together. However, if the values needs to be selected from two different search results, you have to select and insert one value at once.
- The expression `Column3 IN` enables the *Insert Values* dialog. You can select multiple values from the list. If all the required values are available in a single search results, those values can be selected together. However, if the values have to be selected from different search results, you have to select and insert one value at once. You select and insert the values 2, 5, 6, and 9. The expression `Column3 IN (2, 5, 6, 9)` is valid. You want to add another value to it, so you click *Insert Values*. In the *Insert Values* dialog, you can only see the values 2, 5, 6, and 9 because the dialog is already filtered by the selected values. To see all available values, edit your expression to make it invalid: `Column3 IN (2, 5, 6, 9, .`. The *Insert Values* button is available again and you can add new values.

Select available value(s) and click *Insert* to add them to your expression.

- *Functions* - Browse, select a category, or filter available functions (see [SQL Functions Reference \[page 257\]](#)). Click a function name to see its syntax or click elsewhere in its token to add it to your expression.
- *Columns* - Browse or filter available columns. Click a column name to see its properties or click elsewhere in its token to add it to your expression.
- *Parameters* - Browse or filter available input parameters (see [Create an Input Parameter \[page 231\]](#)). Click a parameter name to see its properties or click elsewhere in its token to add it to your expression.
- *Other* - Browse available operators, predicates, and case expressions, and click one to add it to your expression (see [SQL Reference \[page 255\]](#)).

For example, if you want to list only those products with 10 units or less in stock, enter `Units_on_hand <= 10`

When working on a large expression, click  (*Enter Full Screen*) to expand the expression editor.

4. Click *Validate* to check if your expression is semantically correct, and fix any errors signaled. You can reference columns by name as well as HANA functions, operators, predicates, and case expressions.
5. Click  (*Preview Data*) to open the *Data Preview* panel and review the data output by this node. For more information, see [Viewing or Previewing Data in Data Builder Objects \[page 297\]](#).

5.7.1.7 Aggregate Data

Add an *Aggregation* node to perform SUM, COUNT, MIN, and MAX calculations. You group by any non-aggregated columns and can optionally filter by specifying a HAVING clause.

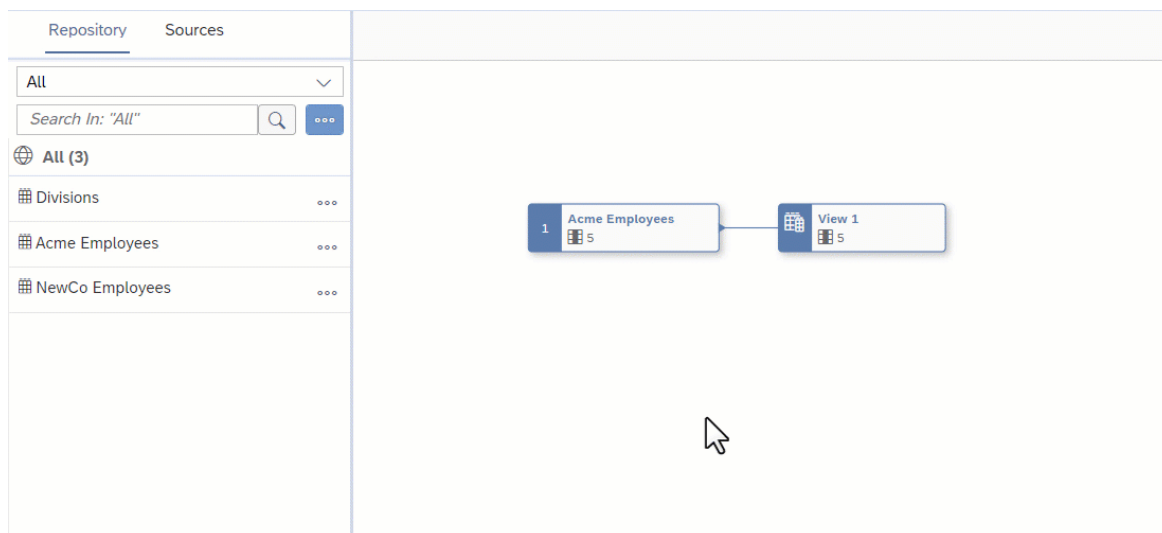
Context

Note

It is not possible to use an *Aggregation* node if the delta capture setting is enabled for the source table.

Procedure

1. Identify one or more columns containing data that you want to aggregate. The columns must have a numerical data type and can be calculated columns.
2. Create a projection node to exclude all columns except this column and the column(s) by which you want to group your values.
For example, if you want to aggregate **Revenue** per **Country**, you should exclude all columns except **Revenue** and **Country**.
3. Select the projection node in order to display its context tools, and click Σ (*Aggregation*).



An *Aggregation* node is created, its symbol is selected, and its properties are displayed in the side panel.

4. Optional. Rename the node in its side panel to clearly identify it. This name can be changed at any time and can contain only alphanumeric characters and underscores.
5. In the *Columns* section, click the **NONE** on a numeric column and select the appropriate aggregation function:
 - **SUM**: calculate the total value

- **COUNT**: calculate the number of distinct values
 - **MIN**: calculate the minimum value
 - **MAX**: calculate the maximum value
6. [optional]. Enter an expression in the *Having* section to filter the aggregated data. You can use the following items in your SQL expression:
- *Insert Values* - Click to open the *Insert Value* dialog, select the values you need to insert in the list, and click *Insert Value*.

Note

- The button is enabled when, in the *Expression* field, a column name is followed by the operator `=`, `>`, `<`, `>=`, `<=`, `!=`, `IN`, `BETWEEN`, or `LIKE`.
- The values listed in the dialog are retrieved from the sources and intermediate nodes of your object. You can insert values of the data types string, integer, boolean, date, and time. The data types binary and UUID aren't supported.
- Once your expression is valid, additional values cannot be inserted. You must edit your expression to make it incomplete; this enables the *Insert Values* button and allows you to insert other values.
- The *Insert Values* dialog doesn't format the expression. Manual formatting may be required after inserting values.

Example

- The expression `Column1 =` enables the *Insert Values* dialog. You can select a single value from the list.
- The expression `Column2 BETWEEN` enables the *Insert Values* dialog. You can select two values from the list. If both values are available in a single search results, both of them can be selected together. However, if the values needs to be selected from two different search results, you have to select and insert one value at once.
- The expression `Column3 IN` enables the *Insert Values* dialog. You can select multiple values from the list. If all the required values are available in a single search results, those values can be selected together. However, if the values have to be selected from different search results, you have to select and insert one value at once. You select and insert the values 2, 5, 6, and 9. The expression `Column3 IN (2, 5, 6, 9)` is valid. You want to add another value to it, so you click *Insert Values*. In the *Insert Values* dialog, you can only see the values 2, 5, 6, and 9 because the dialog is already filtered by the selected values. To see all available values, edit your expression to make it invalid: `Column3 IN (2, 5, 6, 9, .`. The *Insert Values* button is available again and you can add new values.


Select available value(s) and click *Insert* to add them to your expression.

- *Functions* - Browse, select a category, or filter available functions (see [SQL Functions Reference \[page 257\]](#)). Click a function name to see its syntax or click elsewhere in its token to add it to your expression.
- *Columns* - Browse or filter available columns. Click a column name to see its properties or click elsewhere in its token to add it to your expression.
- *Parameters* - Browse or filter available input parameters (see [Create an Input Parameter \[page 231\]](#)). Click a parameter name to see its properties or click elsewhere in its token to add it to your expression.
- *Other* - Browse available operators, predicates, and case expressions, and click one to add it to your expression (see [SQL Reference \[page 255\]](#)).

For example, if you have aggregated your **Revenue** column using `SUM`, and want to show only:

- Total revenues of more than 1m, enter `SUM(Revenue) > 1000000`
- Total revenues for the US only, enter `Country= 'US'`

When working on a large expression, click  (*Enter Full Screen*) to expand the expression editor.

7. Click  (*Preview Data*) to open the *Data Preview* panel and review the data output by this node. For more information, see [Viewing or Previewing Data in Data Builder Objects \[page 297\]](#).

5.7.1.8 Replace a Source

Drag a source from the *Source Browser*, hover over an existing source, and click *Replace*. You are guided through mapping the columns from the old source to the new source.

Context

For example, you may want to replace a local table containing sample data from a CSV file, with a different table containing current data from a source system.

The output structure of the view is preserved as far as possible, and any impacts caused by the replacement of the source are indicated by validation messages.

Procedure

1. If the *Source Browser* panel is not visible on the left of the screen, click *Source Browser* in the toolbar to show it.
2. Select the new source and drag it over the source you want to replace in the diagram. Wait for the context menu to appear, slide your cursor over the *Replace* option and then release the mouse button.

The *Replace Source* dialog opens, showing the old source and its columns on the left side and the new source and its columns on the right side, with mappings proposed, where possible between them.

Note

Only the columns used in the current view are listed on the left side. Any columns that are excluded in your view are not used, and therefore do not need to be remapped.

3. Review the proposed mappings and try to map any unmapped old source columns if possible.

Note

You are not required to map all the columns from the old source, but any you leave unmapped will generate validation warnings or errors. Any new source columns you leave unmapped will be excluded by default.

4. Click [Replace](#) to close the dialog and confirm the replacement. SAP Datasphere performs the following actions:
 - Replace the old source by the new source in your view.
 - If any old source columns were left unmapped, generate warning and error messages for each node that is impacted.
 - If any new source columns were left unmapped, insert a projection node after the source (if one is not already present) and exclude these new columns so that they do not modify the output columns.
 - Update the *Status* of your view.
5. [if new source columns were left unmapped] Review the projection node and restore any new columns that you want to make available (see [Reorder, Rename, and Exclude Columns \[page 223\]](#))
6. [if old source columns were left unmapped] Review each of the validation warnings and errors and correct them as appropriate.
7. Click the [Back](#) button to return to the [Transformation Flow Editor](#), and then save the transformation flow.

Note

If you have not resolved all the validation errors, then you can still save your view, but you will not be allowed to deploy it.

8. If your view is itself used as a source by other views, then review the [Dependent Objects](#) section in the side panel to see if your change has impacted any of them. You can click on an object in the list to navigate to it and review any errors and warnings.
9. When appropriate, click [↶](#) ([Deploy](#)) to deploy your transformation flow.

5.7.2 Create an SQL View Transform

Create an SQL view transform to combine and transform data in a powerful SQL editor. You can choose between writing a standard SQL query using `SELECT` statements and operators such as `JOIN` and `UNION`, or use SQLScript to produce a table function. You can drag sources from the [Repository](#), and easily view the columns defined for your output structure in the side panel.

Context

If you are not comfortable with SQL, you can still build a view by using the [Graphical View Editor](#), which lets you compose SQL code using an intuitive graphical interface. For more information, see [Create a Graphical View Transform \[page 175\]](#).

Procedure

1. On the [New Transformation Flow](#) screen, click the [View Transform](#) node.
2. Create a new [SQL View Transform](#) for your transformation flow by clicking the [SQL View Transform](#) button. The system displays the [SQL View Editor](#).

3. In the side panel, select the language that you want to use. You can choose between:

- [SQL \(Standard Query\)](#) - [default] Create a standard SQL query, based around `SELECT` statements (see [SQL Reference \[page 255\]](#)).
- [SQLScript \(Table Function\)](#) - Use SQLScript with support for `IF` statements, loops, and other more complex structures (see [SQLScript Reference \[page 264\]](#)).

Note

SAP Datasphere supports a subset of the SQL functions supported by SAP HANA Cloud. For more information, see [SQL Functions Reference \[page 257\]](#).

Note

If you use an SQL view transform to load delta changes from a remote table located in an SAP BW bridge space, the language must be [SQL \(Standard Query\)](#). [SQLScript \(Table Function\)](#) is not supported.

4. Enter your code in the [SQL View Editor](#). You can:

- Access auto-complete suggestions for keywords and object names including source tables by typing.
- Drag source tables from the [Repository](#) to the [SQL View Editor](#).

Note

If the delta capture setting is enabled for a source table, the columns [Change Date](#) and [Change Type](#) are automatically mapped to these columns in the target table. Mapping these columns (or a calculated column that contains the content of these columns) to any other target column is not permitted. For more information, see [Capturing Delta Changes in Your Local Table \[page 118\]](#).

Note

The [Change Type](#) column does not support null values. Ensure that no null values are written to the [Change Type](#) column of the target table.

- Add comments to document your code:
 - Comment out a single line or the rest of a line with a double dash: `-- Your comment here.`
 - Comment out multiple lines or part of a line with: `/* Your comment here */.`


This example contains various forms of comments:

```
/*
    This is a multi-line comment to
    introduce this view
*/
SELECT "ID",
       "Date",
       -- "Sales Person", comment out a line
       "City", -- comment at end of line
       "Net Sales"
FROM /* mid-line comment */ "Sales"
```

- Format your SQL code by clicking [Format](#).

Note

SQLScript cannot be formatted.

- Validate your code and update the display of your output structure in the side panel at any time by clicking  ([Validate SQL and Preview Data](#)).


Note

If you add a new column to a table, but do not deploy the table, the system will display an error message if you validate SQL code that references that column.

SQL errors and warnings are shown in the [Errors](#) pane at the bottom of the editor. Problems with the output structure are shown on the [Validation Messages](#) button in the side panel header.

Note

If your SQLScript is complicated, SAP Datasphere may not be able to determine the output structure. In this case, you are requested to review the list of columns. Click the [Edit](#) button to add or delete buttons or change column names and data types.

- Preview the data being output by the view by clicking  ([Show or hide data preview](#)) (see [Viewing or Previewing Data in Data Builder Objects \[page 297\]](#)).

Note

It is not possible to preview data if you are using SQLScript.

5. Review the list of columns (see [Columns \[page 108\]](#)).
6. Click the [Back](#) button to save your work and return to the [Transformation Flow Editor](#). You can then add or create a target table for the transformation flow. For more information, see [Add or Create a Target Table \[page 191\]](#).

5.7.2.1 Additional Information About Remote Tables

You can add remote tables located in SAP BW bridge spaces to a transformation flow. Note that remote tables located in SAP BW bridge spaces must be shared with the SAP Datasphere space you are using to create your transformation flow.

Note

If you add a remote table located in SAP BW bridge spaces, it is not possible to preview the data being output by the view.

If you add a remote table located in an SAP BW bridge space that is configured to load delta changes, the system adds the following input parameters to the [Properties](#) panel of the view transform:

- **REQTSN_LOW**
When loading delta changes, the value of this input parameter is the watermark. When loading all active records, the value of this input parameter is the minimum timestamp. For more information, see [Watermarks](#).
- **REQTSN_HIGH**
The value of this input parameter is the maximum timestamp.

- EXTRACTION_MODE
The value of this input parameter is DELTA_AI to indicate the loading of delta changes from the remote table.

5.7.3 Add or Create a Target Table

A transformation flow writes data to a target table. You can create a new target table or use an existing one.

Procedure

1. Add a target table to the transformation flow in one of the following ways:
 - Click the [Create New Table](#) button.
The target table is added to your transformation flow, and the [Properties](#) panel for the table appears. Enter a business name and a technical name for your target table. When you enter a business name, the system automatically suggests a corresponding technical name, but you can change the technical name if required.
 - Drag an existing table from the [Repository](#), and drop it onto the [Target](#) node.

Note

If the load type of the transformation flow is [Initial and Delta](#) and you change the target table, only delta data will be transferred to the new target table. If you want to transfer all data to the target table, you can reset the watermark in the [Data Integration Monitor](#). For more information, see [Watermarks](#).

2. Click the target node to display its properties in the side panel, and review the properties in the [General](#) section:


Property	Description
Business Name / Technical Name	Information to identify the target table.
Status	The deployment and error status of the object. For more information, see Saving and Deploying Objects [page 39] .
Truncate	<p>If the value of the Truncate option is Yes, when you start the transformation flow, the system deletes the table content, but leaves the table structure intact and fills it with the relevant data from the source table.</p> <p>If not, the system inserts new data records after the existing data in the target table. For data records that already exist in the target table and have been changed in the source, the system updates the target records with the changed data from the source using the UPSERT mode. Note that the system will only update such target records if the target table has a primary key column.</p>

Property	Description
Delta Capture	Indicates whether the delta capture setting is enabled for the table. For more information, see Capturing Delta Changes in Your Local Table [page 118] .
Delta Capture Table Name	[read-only] The delta capture table name. The system displays this field if the delta capture setting is enabled for the table. For more information, see Capturing Delta Changes in Your Local Table [page 118] .

- In the *Mappings* section, review the mappings of incoming columns to the target table columns.
 - When you connect any operator to the target table, each incoming column that has the same name and a compatible data type with a column in the target table is automatically mapped to it.
 - You can manually map an incoming columns with a target table column that has a compatible datatype by dragging the column from the left list and dropping it onto the appropriate column in the right list.

Note

For the data type "string", incoming columns and target table columns do not need to be the same length. It is possible to map incoming columns to target table columns that have a greater length.

- To delete a mapping, select it, and click *Delete*. To delete all mappings, *Remove all Mappings*.
- You can, at any time, click  (*Auto Map*) to relaunch automapping based on names and datatypes.

Note

If the delta capture setting is enabled for the target table of a transformation flow, the *Change Date* column of the target table will always contain the time that the transformation flow was run, even if an incoming column is mapped to the *Change Date* column.

- In the *Columns* section, view the columns of the target table. To check the data type of a column, hover over it and click .

If the delta capture setting is enabled for a table, two additional columns are present in the table to track changes. For more information, see [Capturing Delta Changes in Your Local Table \[page 118\]](#).

5.7.4 Processing Changes to Source and Target Tables

If source or target tables of your transformation flow are modified, you can review the changes and decide how your transformation flow should handle the changes.

It is only possible to review target table changes in the *Transformation Flow Editor*. For more information, see [Process Target Changes in the Transformation Flow Editor \[page 193\]](#).

It is only possible to review source table changes in the *Graphical View Editor* or in the *SQL View Editor*. If a source table of your transformation flow is modified, then the next time you open the transformation flow a message will inform you that the table was modified. However, depending on the type of view used by the transformation flow, you will need to either edit the graphical view transform or the SQL view transform to view the details of the change. For more information, see [Process Source Changes in the Graphical View Editor \[page 194\]](#) and [Process Source Changes in the SQL View Editor \[page 195\]](#).

5.7.4.1 Process Target Changes in the Transformation Flow Editor

If the target table of your transformation flow is modified, then the next time you open the transformation flow, you will be asked to process the changes. If a change has generated errors in your transformation flow, you will receive a notification inviting you to review them.

Procedure

1. If a target change has generated errors in your transformation flow, you will receive a notification inviting you to review them, and you can click the notification to open it directly.

The *Target Updates* dialog opens, listing the objects that have been modified.

Note

If the changes do not generate errors (for example, new columns are available), you will not receive a notification, but the dialog will still be displayed the next time that you open the transformation flow.

2. Click *OK* to dismiss the dialog and open the diagram. The following validation messages may be displayed:

Change	Impact
New Column	New target columns are left unmapped by default in the target table node of the transformation flow: <ul style="list-style-type: none">• An information message listing all new columns is displayed on the target.
Change Column Data Type	Target column data type changes may generate information or error messages for the transformation flow: <ul style="list-style-type: none">• An information message listing all columns with changed data types is displayed on the target.• An error message is displayed on any node where an incompatible data type change is present.
Delete Column	Deleted target columns generate errors for the transformation flow if the columns are used in the transformation flow: <ul style="list-style-type: none">• An information message listing all deleted columns is displayed on the target.• An error message is displayed on any intermediate node (join, union, projection, aggregation, script) in which a deleted column was used.
Change Key Column	Changes to key columns will generate information messages: <ul style="list-style-type: none">• An information message listing all changed key columns is displayed on the target.

3. Review all information messages to ensure that they do not adversely impact the output of your transformation flow.
4. Review and resolve any error messages.
5. If appropriate, map any new target columns in the target table node.

You cannot modify the structure of the target table directly in the transformation flow editor. If you need to make such changes (for example to add new source columns or change data types), you must open

and edit it in the table editor. Once you have saved the updated target table, you can return to the transformation flow and adjust mappings as appropriate in the target table node.

6. Click  (*Save*) to save the changes to your transformation flow and dismiss the information messages.

5.7.4.2 Process Source Changes in the Graphical View Editor

If one or more of the source tables of your view transform is modified, then the next time you open the *Graphical View Editor*, you can view any changes by viewing the relevant validation messages.

Procedure

1. If a source change has generated warnings or errors in your graphical view transform, you can view any changes by viewing the relevant validation messages.
2. The following validation messages may be displayed:

Change in Source	Impact
Add Column	Columns added to source tables or views are excluded by default for the view transform: <ul style="list-style-type: none">• An information message listing all new columns is displayed on the source.• The new columns are excluded in a projection node directly after the source. If no projection node was previously present, then one is added.
Change Business Name	Changes to the business name and other business properties of a source table or its columns are automatically propagated to dependent objects: <ul style="list-style-type: none">• An information message specifying the source object changes or listing all changed columns is displayed on the source.
Change Column Data Type	Changes column data types in source tables generate warnings: <ul style="list-style-type: none">• An information message listing all columns with changed data types is displayed on the source.• An information message listing all columns with changed data types present in the output node <i>Columns</i> list is displayed on the output node.• If output columns with changed data types are used by any dependent objects of this view, then a warning message listing those views is displayed on the output node.

Change in Source	Impact
Delete Column	<p>Deletions of columns in source tables generate errors if the columns are used by the view transforms:</p> <ul style="list-style-type: none"> • An information message listing all deleted columns is displayed on the source. • An error message is displayed on any intermediate node (join, union, filter, calculated columns, aggregation) in which a deleted column was used. • A warning message listing all deleted columns that were previously present in the output node <i>Columns</i> list is displayed on the output node. • If deleted output columns were used by any dependent objects of this view, then a warning message listing those views is displayed on the output node.

3. Review all information and warning messages to ensure that they do not adversely impact the output of your graphical view transform.
4. Review and resolve any error messages on intermediate nodes.
5. Click the *Back* button to return to the *Transformation Flow Editor*, and then save and deploy the transformation flow.

5.7.4.3 Process Source Changes in the SQL View Editor

Context

If one or more of the source tables of your SQL view transform is modified, then the next time you edit the SQL view transform you can view any changes by viewing the relevant validation messages.

Procedure

1. If a source change has generated warnings or errors in your SQL view transform, you can view any changes by viewing the relevant validation messages.
2. Click the *Validate SQL* button and review and correct any errors that appear in the *Errors* panel.
3. Review all information and warning messages to ensure that they do not adversely impact the output of your SQL view transform.
4. Click the *Back* button to return to the *Transformation Flow Editor*, and then save and deploy the transformation flow.


5.8 Running a Flow

Once your flow is configured and deployed, you can run it.

To run a flow, you have 3 main options depending on your flow type:

- Click [Run](#) to start a direct run.
- Click [Schedule](#) to run your data flow or your transformation flow at later time, or on regular basis (this is not available for [Replication Flow](#)).
- Add the flow into a task chain. This is only valid for a data flow. For more information, see [Creating a Task Chain \[page 197\]](#).

Run a Direct Flow

Once you have completed the flow configuration and saved it, you can run it. Click  ([Run](#)) to start the process to acquire and transform data as per your defined settings. Once completed, the [Run Status](#) section in the property panel is updated. You can navigate to the [Flows](#) monitor to get more detail on the run. See [Monitoring Flows](#).

Note

Regarding data flows:

- If your data flow contains input parameters, a dialog box appears prompting the user to enter a value for each input parameter. You can either keep the default value or override it (see [Create an Input Parameter \[page 142\]](#)).
- The initialization time for running a data flow takes an average of 20 seconds even with smaller data loads causing longer runtime for the data flow.
- Metrics are displayed only for source and target tables and can be used for further analysis. They are available in the [Flows](#) monitor.
- If a source view or an underlying view of the source view has data access controls applied to it, then no data is read from the view during the execution of the data flow. This results in incorrect data or no data in the output.


For more information, see [Securing Data with Data Access Controls](#).

Note

Regarding transformation flows:

You can cancel a running transformation flow in the [Data Integration Monitor](#). For more information, see [Cancel a Transformation Flow Run](#).

Create a Schedule to Run Your Flow

You can also create a schedule to run your data flow or your transformation flow on regular basis. Click  (*Schedule*) and define your schedule. For more information, see [Scheduling Data Integration Tasks](#).

Note

For optimal performance, it is recommended that you consider staggering the scheduled run time of tasks such as data flows and task chains that may contain these tasks. There is a limit on how many tasks can be started at the same time. If you come close to this limit, scheduled task runs may be delayed and, if you go beyond the limit, some scheduled task runs might even be skipped.

Note

In very rare situations, an error may occur the first time that a scheduled data flow is run in a space. In this case, you should run the data flow manually once. Subsequent scheduled runs will not require further intervention.

Run Your Data Flow Using a Task Chain

You can run a data flow using a task chain. For more information, see [Creating a Task Chain \[page 197\]](#).

Alternatively, you can also run your flows from the  (*Data Integration Monitor*).

From the *Flows* monitor, you can check details of your runs, and perform other actions on your flows. For more information, see [Monitoring Flows](#).

5.9 Creating a Task Chain

Group multiple tasks into a task chain and run them manually once, or periodically, through a schedule. You can create linear task chains in which one task is run after another. (You can also nest other task chains within a task chain.) Or, you can create task chains in which individual tasks are run in parallel and successful continuation of the entire task chain run depends on whether ANY or ALL parallel tasks are completed successfully. In addition, when creating or editing a task chain, you can also set up email notification for deployed task chains to notify selected users of task chain completion.

Prerequisites

- The DW Modeler role is required to create task chains, and the additional DW Integrator role is required to set up email notification for completion of executed task chains, and to run the activity *Delete Records with Change Type "Deleted"* in the case of local table with delta capture enabled. For more information, see [Standard Roles Delivered with SAP Datasphere](#). In addition to these two role privileges, when setting

up email notifications, either the Team.Read or User.Read privilege is also required to display and add notification recipients from a list of current tenant members. See [Privileges and Permissions](#).

- Objects must have been already deployed, so that they can be added to the task chain. Task chains must also be deployed to allow selection of tenant users or specify email addresses for notification of task chain completion.
- Remote tables must not be set to real-time replication.
- Views must not have parameters or data access controls assigned to them.
- If a data flow that has input parameters is included in a task chain, task chain runs will use default parameter values defined for the data flow.
- A replication flow can be included in a task chain if all objects in the flow have load type *Initial Only*.

Context

Linear task chains allow you to define a group or series of tasks and execute those tasks in a serial process, one after another. A succeeding task is only executed once the previous task in the series has finished successfully with a *completed* status. The execution of tasks in the series will not resume if the previous task has a *failed* status.

Parallel task chain branches allow you to specify that some individual tasks are run in parallel and successful continuation of the entire task chain run depends on whether ANY or ALL parallel tasks are completed successfully.

Tasks chain scheduling may include execution of Remote Table Replication, View Persistency, Intelligent Lookup, Data Flow, Replication Flow (load type *Initial Only*), and Transformation Flow runs. You can also nest other task chains within a task chain.


Note

For optimal performance, it is recommended that you consider staggering the scheduled run time of tasks such as data flows and task chains that may contain these tasks. There is a limit on how many tasks can be started at the same time. If you come close to this limit, scheduled task runs may be delayed and, if you go beyond the limit, some scheduled task runs might even be skipped.

After deploying a task chain, you can add tenant users or email addresses to notify individuals when task chain runs are completed.

You can monitor the status of task chain runs from the Data Integration Monitor. For more information, see [Monitoring Task Chains](#).

Note

Exporting and importing task chains via the  (*Transport*) app may not be supported for SAP Datasphere tenants provisioned prior to version 2021.03. To request the migration of your tenant, see SAP note [3268282](#).

This topic contains information on performing the following tasks:

- Basic linear task chain creation
- Creation of parallel task branches in chains

- Task chain email notification

Procedure

Basic Creation of a Linear Task Chain

A basic or linear task chain allows you to define a group or series of tasks and execute those tasks in a serial process, one after another. A succeeding task is only executed once the previous task in the series has finished successfully with a completed status. The execution of tasks in the series will not resume if the previous task has a failed status.

1. From the *Data Builder*, click *New Task Chain*.
2. From the left-side panel, drag and drop a first object on to the task chain canvas from those available in the repository.

Note

In the repository, you can see the remote tables, views, intelligent lookups, data flow, replication flow, and transformation flow objects that meet prerequisites and are available to be added to the task chain.

Note

If you add a remote table whose data access is *Replicated (Real-time)* in a task chain, the replication type will change from real-time replication to batch replication at the next run of the task chain. The data will no longer be updated in real-time.

3. Drag a second object on to the first object in the task chain. As you drag the object over the top of the first object, a context menu displays options *Add as New Task* (the default), *Replace Existing*, or *Add as Parallel* (described in the next section) to place the new object.

Choosing the *Add as New Task* option automatically connects the new object task to the previous object task. The properties panel for the task chain is also updated with the added objects.

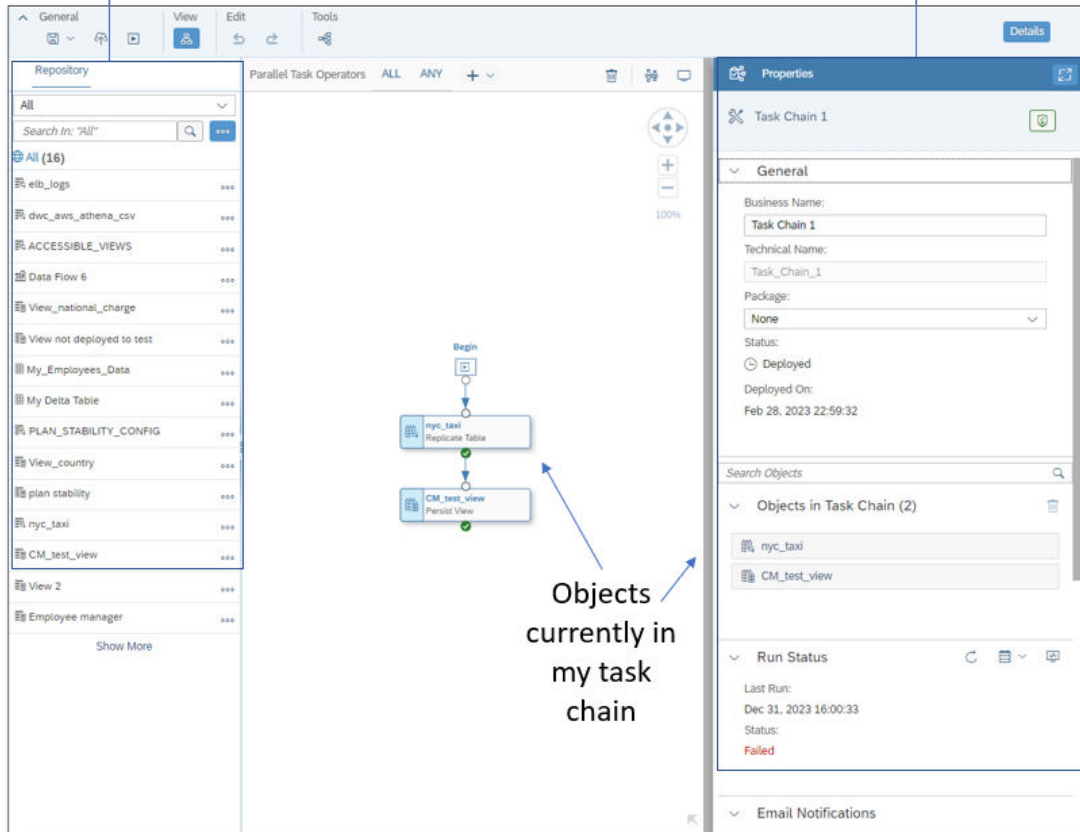
4. Continue adding remaining object tasks you want to include in the task chain.

In addition to adding or replacing object tasks in a task chain, you can drag objects already on the task chain canvas to change the order in which tasks are executed.

5. In the properties panel, specify a name for the task chain.

Objects in the Datasphere repository
that can be added to the task chain

Properties of the task chain



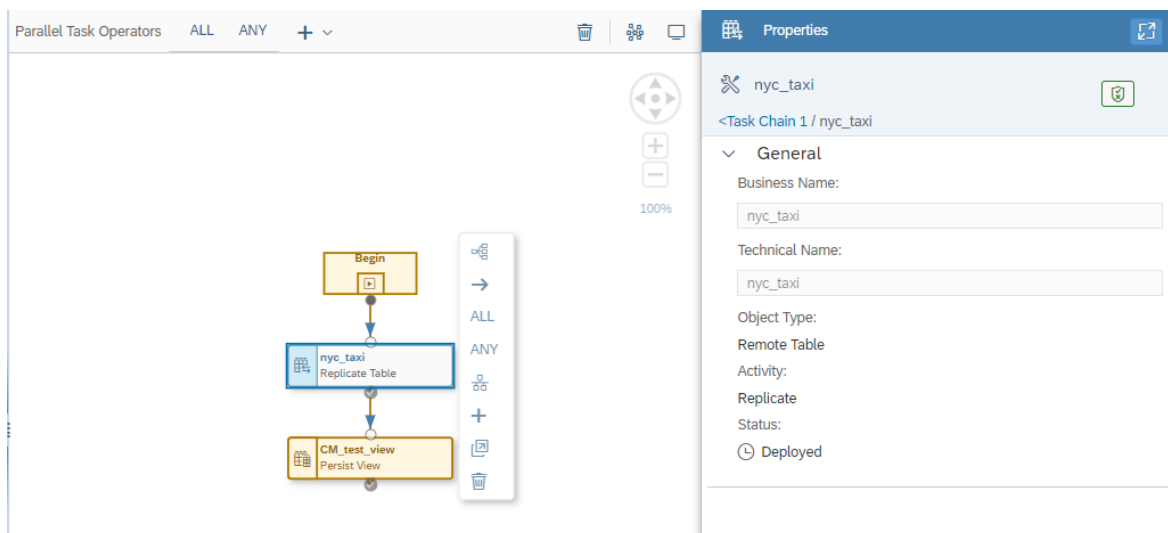
Objects
currently in
my task
chain

Task chain properties:

Properties	Comments
Business Name	Enter a descriptive name to help users identify the object. This name can be changed at any time.
Technical Name	Displays the name used in scripts and code, synchronized by default with the <i>Business Name</i> .

Properties	Comments
Package	<p>Select the package to which the object belongs.</p> <p>Packages are used to group related objects in order to facilitate their transport between tenants.</p> <div style="border: 1px solid #ccc; padding: 5px; margin-top: 10px;"> <p>Note</p> <p>Once a package is selected, it cannot be changed here. Only a user with the DW Space Administrator role (or equivalent privileges) can modify a package assignment in the <i>Packages</i> editor.</p> </div> <p>For more information, see Creating Packages to Export.</p>
Status	Displays the deployment status of the task chain: it can be deployed, not deployed, or changes to deploy.
Objects in Task Chain	Displays all objects that have been added to the task chain.
Run Status	Displays the current run status of the task chain: Not run yet, running, failed, or completed.
Email Notifications	Set up email notification for task chain run completion.

When you click on one of the task chain objects, the properties for this selected task object is displayed in the properties panel:



Note that you can also access the details of each task chain object in the task chain properties panel. Select the relevant object in the object list and click > :



Task chain object properties:

Properties	Comments
Business Name	Name of the object
Technical Name	Technical name of the object
Object Type	A remote table, view, intelligent lookup, data flow, replication flow (load type <i>Initial Only</i>), or transformation flow
Activity	Activity that will be triggered by the task chain: <ul style="list-style-type: none">• Remote table - Replicate• View - Persist• Intelligent lookup - Run• Data flow - Run• Replication Flow - Run• Transformation flow - Run• Local table - Delete Records with Change Type "Deleted"
Status	Deployment status of the task chain: it can be deployed, not deployed, or changes to deploy

Note

When you select an object, you can delete it from the task chain or navigate to the corresponding editor for that object.

6. Save and deploy your task chain.

The properties of your task chain are updated.

Run Status is updated:
 - Run date and time are added
 - Status of the run is updated.

Task chain status is updated and date and time of the deployment are added

New actions are available:
 - Refresh to refresh the run status
 - Schedule: to create a schedule for next run
 - Open in Task Chain Monitor to navigate to the Data integration monitor and monitor your task chain

Once the task chain is deployed, you can then run the task chain or create a schedule to run your task chain periodically, and navigate to the *Task Chains* monitor to check your task chain runs. For more information, see [Scheduling Data Integration Tasks](#) and [Monitoring Task Chains](#).

Once a task chain run has started, it will continue running as long as possible. Until all tasks in the chain have been executed and are in a non-running state, the task chain itself is considered to be "running". When finished, the overall state or status of the task chain will be reported as "failed" if any task in the chain has "failed". The final status of COMPLETED for a task chain is reported only if all tasks are COMPLETED.

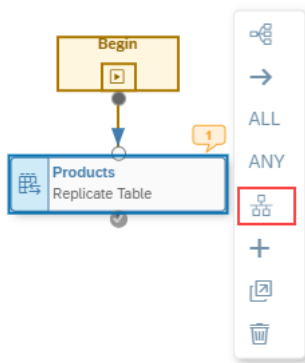
Note

If a nested task chain within a parent task chain fails, you need to retry the parent task chain, not the specific nested chain that failed. In that case, when you retry the parent task chain, only the nested task chain that failed will be run again, not any of the other tasks in the chain that had already run successfully.

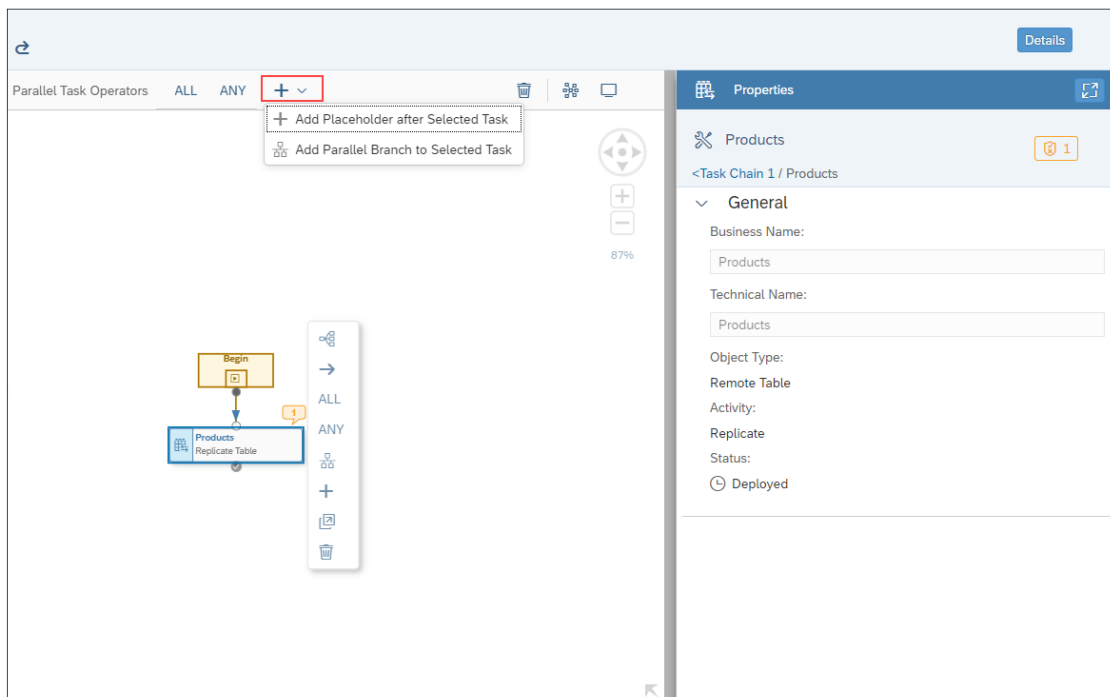
Executing Parallel Tasks in a Task Chain

In addition to linear task chains in which one task is executed after another, you can also create task chains in which individual tasks are run in parallel and continuation of the entire task chain run depends on whether ANY or ALL parallel tasks are completed successfully.

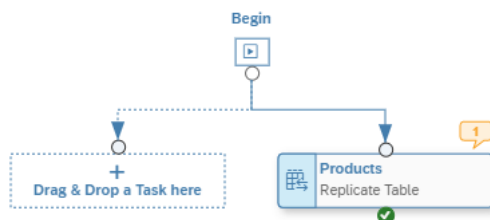
- After adding an object to the task chain canvas, there are a few different ways in which you can specify that the object or task you added is part of a parallel branch in the task chain:
 - Select the task object you added to the canvas and then click the *Add as Parallel Branch* option from the list of context menu options available.



- Alternatively, you can select the task object, click **+** *Add* from the shell bar above the canvas, and select the *Add Parallel Branch to Selected Task* menu option.



If you choose either of these options, a new task placeholder is added to the canvas, next to the currently selected task object.



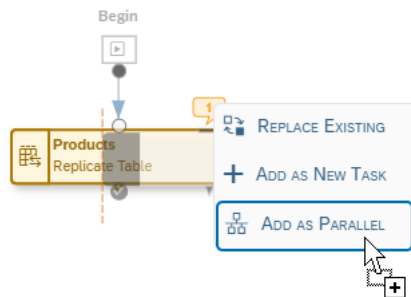
You can then drag another task object on to the canvas to take the place of the new task placeholder.

Note

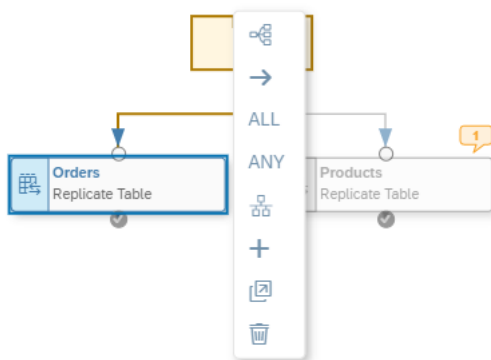
The task chain editor displays indications of warnings and errors with the bubble icon appearing next to specific task objects. You can click on an individual bubble to view the specific warning or

error that was detected; for example, if a specific task was already included in another task chain, or if you need to make additional changes or updates to the task chain for it to be deployed and run.

- Another way to create a parallel branch in a task chain is simply to drag a new object to the canvas on top of the currently selected task object and then choose the *Add as Parallel* context menu option.



Following this selection, the new task object will be placed next to the currently selected object. For example:



You may continue to add additional parallel task objects in the same way. There is no predefined limit on the number of tasks you can include in a parallel task chain branch, however you cannot nest another task chain within an existing one.

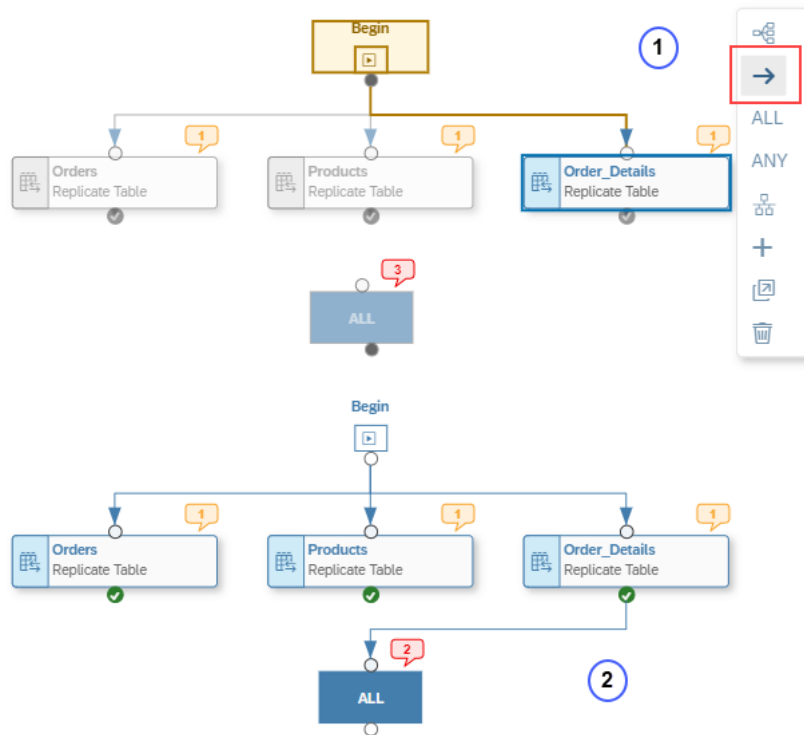
8. When you have finished adding task objects to the parallel task chain branch, select the ANY or ALL operator to apply to the execution of the parallel task objects in the chain.

To do that, select one of the parallel branch task objects and select either the *ANY* or *ALL* operator from the task object's context menu. Or, you can click and drag the ANY or ALL operator from the shell bar on to the task chain canvas below the parallel task chain branch you've created.

Note

To switch the ANY or ALL operator, after adding it to the task chain canvas, simply select the current operator and then toggle the ANY/ALL option in the task chain's Properties panel.

9. Next, connect each of the parallel branch task objects to the ANY or ALL operator you placed on the canvas. To do that, select a task object in the branch, then click and drag the → *Connect* arrow to the ANY or ALL operator to connect the selected task object.



10. Connect the remaining task objects in the branch to the ANY or ALL operator, in the same way, to complete creation of the parallel task chain branch.

Following the completion of the task chain branch, you can then continue to add additional linear tasks, or create additional task object branches in the task chain.

11. When you've finished adding tasks objects to the task chain, save and deploy your new task chain.

Note

SAP Datasphere allows you to save task chains that may have unconnected task objects on the canvas. However, you will not be able to deploy and run them until all task objects are connected to define their order of execution when the task chain is run.

When a task chain is run that includes a parallel task chain branch, execution of all the branch tasks are triggered to be run in parallel. The ANY or ALL condition applied to the branch specifies whether ANY or ALL branch tasks must be completed successfully to continue with execution of remaining tasks in the chain.

After finishing a task chain run that includes one or more parallel task branches, it may be possible that one or more tasks may be reported in an error state (in each branch). For example, in branches where completion of tasks is evaluated with the ANY operator. In that case, if you restart or retry the task chain, SAP Datasphere will then restart previously-failed tasks and run all subsequent tasks that had not yet run. In particular, this means that if a failed task is in a parallel branch which was evaluated with the ANY operator, those tasks in the same branch which had run successfully will not be run again. Only those tasks that have failed will be retried or run again.

Configuring Email Notifications

After creating and deploying a task chain, you can set up email notification for completion of task chain runs.

Note

The DW Integrator role is required to set up email notification for completion of executed task chains. The *Email Notifications* section of the task chain *Properties* panel will not appear if you do not have this privilege assigned. For more information, see [Standard Roles Delivered with SAP Datasphere](#). In addition to the DW Integrator role, when setting up email notifications, either the Team.Read or User.Read privilege is also required to display and add notification recipients from a list of current tenant members.

To set up email notification:

12. In the *Email Notifications* section of the *Properties* panel, select when you want notifications to be sent for the current task chain. You can choose from the following options:
 - Send email notification only when the run has completed with an error.
 - Send email notification only when the run has completed successfully.
 - Send email notification when the run has completed.


The *Email Notifications* section expands to show details of the email notification message to be sent to users on completion of task chain runs.

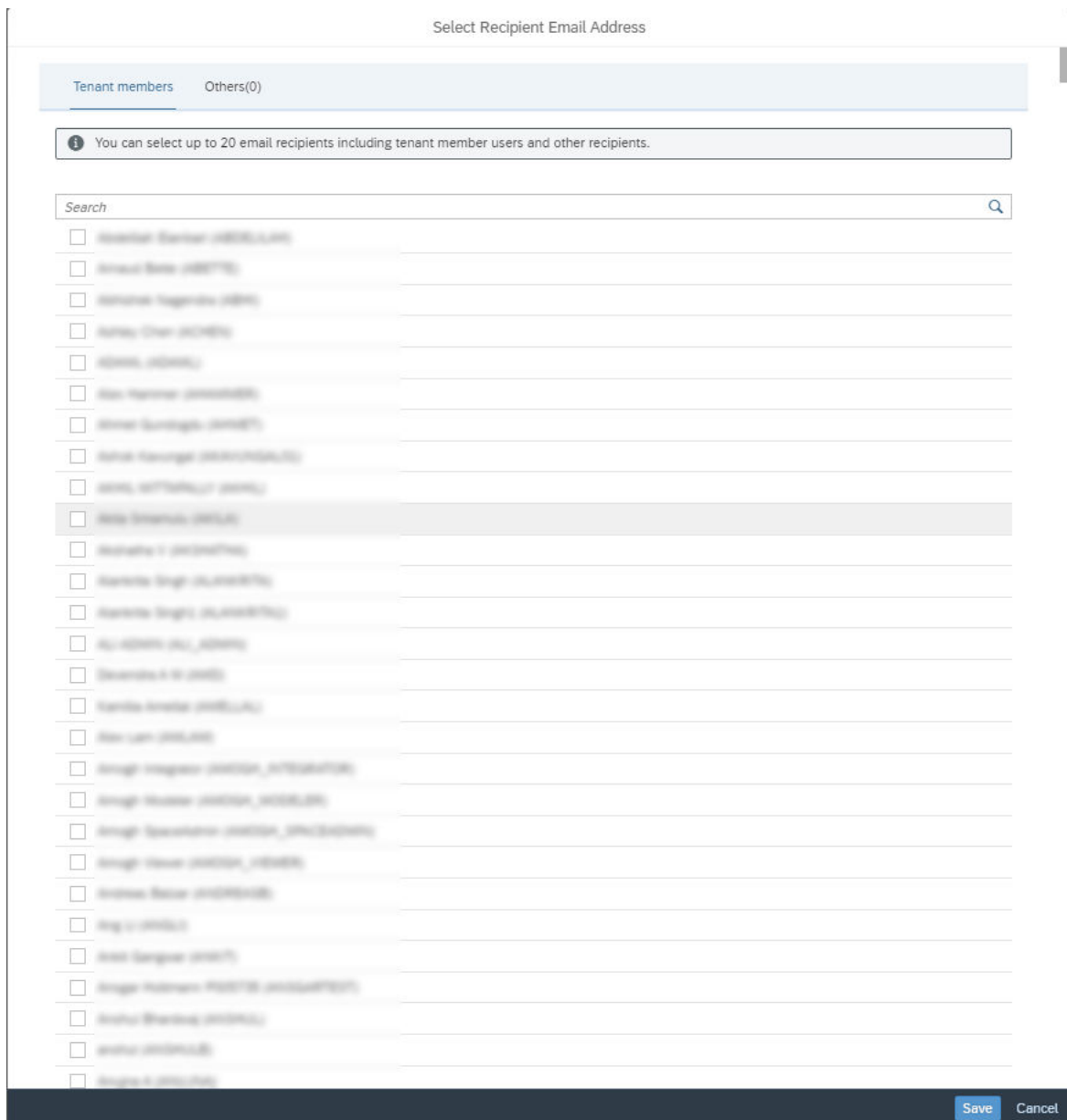
The screenshot shows the 'Email Notifications' configuration panel. It includes a dropdown menu for 'Notification Settings' with the option 'Send email when the run has completed' selected. Below this is a text field for 'Recipient Email Address' with a copy icon on the right. The 'Email Subject' field contains the text 'Notification for Task Chain: \$\$taskChainName\$\$ - Space: \$\$...'. The 'Email Message' field contains a template with the following text: 'Hello, Your task chain labeled \$\$taskChainName\$\$ has finished with status \$\$status\$\$.' followed by a list of details: 'Here are some other details about the task chain: - Task chain technical name: \$\$taskChainName\$\$ - Log ID of the task chain run: \$\$logId\$\$ - User that ran the task chain: \$\$user\$\$ - Link to the log display of the task chain: \$\$uiLink\$\$ - Start time of the task chain run: \$\$startTime\$\$ - End time of the task chain run: \$\$endTime\$\$ - Name of the space: \$\$spaceId\$\$'.

The notification setup includes a default template you can customize for both the email message subject and body text. You can choose to send notifications to other tenant users or specify email addresses of other users to receive notifications (up to 20 recipients total). Email addresses must be in the same domain as the tenant owner.

Note

Task chains must also first be deployed before you can select or specify users to receive notifications.

13. Click the  link on the right side of the *Recipient Email Address* field to open a popup dialog in which you can add recipients of task chain notification email messages.



From this dialog, you can select member users of the same tenant or click the *Others* tab to specify email addresses of other users you want to receive notifications (up to 20 total recipients). Email addresses must match the domain of the tenant owner, for example, `jdoe@sap.com`. After saving your selections, the display returns to the *Properties* panel, showing the selected users in the *Recipient Email Address* field.

Note

When setting up email notification, either the `Team.Read` or `User.Read` privilege is required to be able to display and add notification recipients from the list of current tenant members. If you do not have this privilege assigned, you can still add recipients manually from the *Others* tab.

- Review the default email subject and message body text and make any updates to either the text or placeholder variables used in the notification email message sent for the current task chain.

Placeholder variables within the subject and message fields are enclosed by \$\$ characters, for example, \$\$status\$\$\$. You can click the ⓘ icon to display a list of available placeholder variable names you may include in either the email subject or message body text fields.

ⓘ Note

Changes you make to the email notification template are saved when you redeploy the updated task chain that has email notification subject and body text template changes. When the task chain is run and notification emails are sent out, placeholder variables in the notification template will be replaced with actual values available at runtime. You should note, however, that no email notifications will be sent out if any error occurs during initialization or preparation to run a task chain, before task execution actually starts.

6 Preparing Data in the Data Builder

Users with the *DW Modeler* role can use views and intelligent lookups in the *Data Builder* to combine, clean, and otherwise prepare data.



This topic contains the following sections:

- [Combine, Filter, and Enrich Data with Views \[page 210\]](#)
- [Combine Data via Match Rules in an Intelligent Lookup \[page 211\]](#)
- [Browse the Catalog for Trusted Data Assets \[page 211\]](#)
- [Visualize and Understand the Dependencies Between Objects \[page 211\]](#)
- [Create Objects and Act On Existing Objects \[page 211\]](#)

For information about identifying the semantic usage of your entities and modeling them for consumption, see [Modeling Data in the Data Builder \[page 301\]](#).

Combine, Filter, and Enrich Data with Views

You can combine, filter, enrich and otherwise prepare data in views.

- You can write SQL or SQLScript (table function) code in a powerful SQL editor (see [Creating an SQL View \[page 250\]](#)).
 - To get started: In the side navigation area, click  (*Data Builder*), select a space if necessary, and click [New SQL View](#) to open the editor.
 - SAP Datasphere supports:
 - A subset of the SQL syntax supported by SAP HANA Cloud (see [SQL Reference \[page 255\]](#)).
 - The SQLScript syntax for table user-defined functions in SAP HANA Cloud (see [SQLScript Reference \[page 264\]](#)).
- You can prepare your data in a graphical no code/low code environment (see [Creating a Graphical View \[page 213\]](#)).
 - To get started: In the side navigation area, click  (*Data Builder*), select a space if necessary, and click [New Graphical View](#) to open the editor.
 - You can add and combine your sources by drag and drop (see [Add a Source \[page 217\]](#), [Create a Join \[page 218\]](#), and [Create a Union \[page 221\]](#)).
 - You can refine, filter, and enrich your data in the diagram (see [Reorder, Rename, and Exclude Columns \[page 223\]](#), [Create a Column \[page 224\]](#), [Filter Data \[page 236\]](#), and [Aggregate Data \[page 238\]](#)).
- By default, views are virtual and must be run each time they are accessed. You can improve performance by persisting the view (see [Persist View Data \[page 242\]](#)).

Combine Data via Match Rules in an Intelligent Lookup

You can join two entities even where there is no appropriate foreign key column or where its data is incomplete or unreliable, with an intelligent lookup. You can iteratively join two entities by defining rules to match records and then reviewing and processing the results (see [Creating an Intelligent Lookup \[page 265\]](#)).

Browse the Catalog for Trusted Data Assets

You can browse the catalog to discover high-quality trusted data assets to use as sources in your views and other objects (see [Finding and Accessing Data in the Catalog \[page 23\]](#)).


Visualize and Understand the Dependencies Between Objects

SAP Datasphere provides various ways to visualize and understand the dependencies between your entities and other objects:

- You can visualize the objects that your object depends on (its lineage) and those that depend on it (its impacts) by opening its impact and lineage analysis (see [Impact and Lineage Analysis \[page 34\]](#)).
- You can visualize a set of entities and the associations between them by adding them to an entity-relationship model (see [Creating an Entity-Relationship Model \[page 286\]](#)).
- You can trace the source of a column in your graphical view and the transformations it has passed through (see [Visualize the Lineage of Columns and Input Parameters in a Graphical View \[page 240\]](#)).

Create Objects and Act On Existing Objects

All the objects you import or create in the *Data Builder* are listed on the *Data Builder* start page. You can act on objects in the list in the following ways:

- Click one of the tabs to filter the list by object type.
- Click a tile to create a new object
- Click  (*Show filters*) to filter the list on collections and search by criteria. Click *Show More* to open a dialog with additional filter options.
- Enter a string in the *Search* field to filter the list on business and technical names and users.
- Click a column header to sort or filter the list by values in the column.
- Select one or more objects and use any of the following tools:

Tool	Description
New	Create <i>Data Builder</i> objects (independent of any selection).

Tool	Description
Import	<p>Import objects from files and connections:</p> <ul style="list-style-type: none"> • Import CSV File - Import data from a CSV file to create a local table (see Creating a Local Table from a CSV File [page 122]) • Import Objects from CSN/JSON File - Import table and view definitions from a CSN file to create tables and views or import data flow definitions from a JSON file to create data flows. (see Importing Objects from a CSN/JSON File [page 49]). • Import Remote Tables - Import remote tables from a connection (see Import Remote Tables [page 91]).
Edit	Open the selected object in the appropriate editor. Alternatively, click the object directly in the list.
Deploy	<p>Select one or more objects and deploy them at once.</p> <p>Choose from the following entity types:</p> <ul style="list-style-type: none"> • Local Table (see Creating a Local Table [page 106]) • Graphical View (see Creating a Graphical View [page 213]) • SQL View (see Creating an SQL View [page 250]) • Data Access Control (see Create a "Single Values" Data Access Control) • Analytic Model (see Creating an Analytic Model [page 337]) • Task Chain (see Creating a Task Chain [page 197]) <p>Other types of objects cannot be deployed.</p> <p>If one or more objects that you have selected cannot be deployed, the <i>Deploy</i> dialog opens, allowing you to review your selection. Click <i>Deploy</i> to deploy those objects listed on the <i>Deployable</i> tab, or <i>Cancel</i> to go back and alter your selection.</p>
Share	Share the selected tables and views to other spaces (see Sharing Tables and Views To Other Spaces [page 44]).
Impact and Lineage Analysis	View the objects that depend on an analyzed object (its impacts) and the objects on which the analyzed object depends (its lineage)(see Impact and Lineage Analysis [page 34]).
Delete	Delete the selected objects.

Note

If the object is used by one or more other objects then a dialog listing these dependencies opens, and the deletion is canceled.

Note

In various places in editors where there is not enough room to show both business and technical names for entities and columns, you can choose which of these names to display. Click ► *Profile* ► *Settings* ► *UI Settings* ► and select either *Show Business Name* or *Show Technical Name*.

6.1 Creating a Graphical View

Create a view to query sources in an intuitive graphical interface. You can drag and drop sources from the [Source Browser](#), join them as appropriate, add other operators to remove or create columns and filter or aggregate data, and specify measures and other aspects of your output structure in the output node.

Context

If you are comfortable writing SQL code or want to use SQL Script to create your view, you can use the SQL View editor (see [Creating an SQL View \[page 250\]](#)).


Note

There are two methods for exposing view data for consumption outside SAP Datasphere:

- SAP Analytics Cloud (and Microsoft Excel via an SAP add-in) do not consume view data directly. Set the *Semantic Usage* of your view to *Fact* and then add it to an analytic model to expose it (see [Creating an Analytic Model \[page 337\]](#)). There is no need to enable the *Expose for Consumption* switch.
- Other third-party BI clients, tools, and apps can consume data from views with any *Semantic Usage* via OData or ODBC if the *Expose for Consumption* switch is enabled.

For more information, see [Consuming Data Exposed by SAP Datasphere](#).

Procedure

1. In the side navigation area, click  (*Data Builder*), select a space if necessary, and click *New Graphical View* to open the editor.
2. Drag a first source from the *Source Browser* and drop it into the diagram.

The source is added to the diagram along with an output node, initially entitled **view 1**, which represents the final output structure of the view.

For more information, see [Add a Source \[page 217\]](#).

If your source has an input parameter, the *Map Input Parameter* dialog appears. You can either set a value for each parameter or map to an input parameter in the view. If you choose to set a value, you can either enter a value manually or, if your input parameter has a predefined default value, get access to a list of predefined values with the *Value Help Dialog*. Only the first 10 records are shown by default. You can see more by using the *Search* bar or clicking *More*.

For more information, see [Create an Input Parameter \[page 231\]](#).










3. Optional. Drag additional sources from the *Source Browser* and drop them on your source to create a join or union.

For more information, see [Create a Join \[page 218\]](#) and [Create a Union \[page 221\]](#).

4. Click a node in the diagram to display tools for creating operators and performing other actions:

Note

You can only create one each of the *Filter*, *Projection*, *Calculated Columns*, and *Aggregation* operators per source or join in your diagram. Operators can be created in any order.

Tools	Description
 (Filter)	Add a <i>Filter</i> node to filter your data with an SQL expression. For more information, see Filter Data [page 236] .
 (Rename/Exclude Columns)	Add a <i>Projection</i> node to rename, reorder, or exclude columns. For more information, see Reorder, Rename, and Exclude Columns [page 223] .
 (Calculated Columns)	Add a <i>Calculated Columns</i> node to create new columns and define calculations in them. For more information, see Create a Column [page 224] or SAP HANA SQL and System Views Reference .
 (Aggregation)	Add an <i>Aggregation</i> node to perform SUM, COUNT, MIN, and MAX calculations. For more information, see Aggregate Data [page 238] .
 (Join Suggestion)	Create a join from a list of <i>Related Entities</i> that is populated based on the presence of associations between the current source and other artifacts.
 (Preview Data)	Preview the data output by the selected diagram node in the <i>Data Preview</i> panel (see Viewing or Previewing Data in Data Builder Objects [page 297]).
<div data-bbox="651 1050 756 1084" data-label="Section-Header"><h3>Note</h3></div> <div data-bbox="651 1102 1339 1238" data-label="Text"><p>Users with the <i>DW Viewer</i> role cannot preview data if the <i>Expose for Consumption</i> switch is disabled and, if the switch is enabled, can only preview data in the output node. For more information, see Roles and Privileges by App and Feature.</p></div>	
<p>You can preview the SQL generated for the node by clicking the Preview SQL button in the panel or by clicking  <i>Export</i> and selecting <i>Preview SQL</i>. Click <i>Copy</i> to copy the SQL code for pasting into the SQL View editor or elsewhere.</p>	
 (Impact and Lineage Analysis)	Open the Impact and Lineage Analysis diagram. This diagram enables you to understand the lineage and impacts of the selected object. (see Impact and Lineage Analysis [page 34] .)
 (Open in New Tab)	Open the object in its own editor in a new tab.







5. Click the output node to show the view's properties in the side panel. Enter the following properties as appropriate:

Property	Description
Business Name	Enter a descriptive name to help users identify the object. This name can be changed at any time.

Property	Description
Technical Name	<p>Displays the name used in scripts and code, synchronized by default with the <i>Business Name</i>.</p> <p>To override the default technical name, enter a new one in the field. Technical names can contain only alphanumeric characters and underscores.</p> <div style="background-color: #f0f0f0; padding: 5px;"> <p>Note</p> <p>Once the object is saved, the technical name can no longer be modified.</p> </div>
Package	<p>Select the package to which the object belongs.</p> <p>Packages are used to group related objects in order to facilitate their transport between tenants.</p> <div style="background-color: #f0f0f0; padding: 5px;"> <p>Note</p> <p>Once a package is selected, it cannot be changed here. Only a user with the DW Space Administrator role (or equivalent privileges) can modify a package assignment in the <i>Packages</i> editor.</p> </div> <p>For more information, see Creating Packages to Export.</p>
Semantic Usage	<p>Select the way your entity should be used for data modeling purposes.</p> <p>Choose from the following:</p> <ul style="list-style-type: none"> • <i>Fact</i> - Contains one or more measures and attributes. A fact typically has associations pointing to one or more dimensions and is consumed by analytic models (see Creating a Fact [page 305]). • <i>Dimension</i> - Contains attributes containing master data like a product list or store directory, and supporting hierarchies (see Creating a Dimension [page 314]). • <i>Hierarchy</i> - Contains attributes defining a parent-child hierarchy (see Creating an External Hierarchy [page 330]). • <i>Hierarchy with Directory</i> - Contains one or more parent-child hierarchies (see Creating a Hierarchy with Directory [page 331]). • <i>Text</i> - Contains attributes used to provide textual content in one or more languages (see Create a Text Entity for Attribute Translation [page 327]). • <i>Relational Dataset</i> - [default] Contains columns with no specific analytical purpose. • <i>Analytical Dataset (Deprecated)</i> - Use <i>Fact</i> instead (see Analytical Datasets (Deprecated) [page 351]).

Property	Description
Expose for Consumption	<p>Enable this option to make the view available for consumption outside SAP Datasphere via OData or ODBC.</p> <div style="border: 1px solid #ccc; padding: 10px; margin: 10px 0;"> <p>Note</p> <p>There are two methods for exposing view data for consumption outside SAP Datasphere:</p> <ul style="list-style-type: none"> SAP Analytics Cloud (and Microsoft Excel via an SAP add-in) do not consume view data directly. Set the <i>Semantic Usage</i> of your view to <i>Fact</i> and then add it to an analytic model to expose it (see Creating an Analytic Model [page 337]). There is no need to enable the <i>Expose for Consumption</i> switch. Other third-party BI clients, tools, and apps can consume data from views with any <i>Semantic Usage</i> via OData or ODBC if the <i>Expose for Consumption</i> switch is enabled. <p>For more information, see Consuming Data Exposed by SAP Datasphere.</p> </div>
Run in Analytical Mode	<p>Enable this option to send the <code>USE_OLAP_PLAN</code> hint to the SQL optimizer.</p> <p>This may improve view performance, particularly if a union is performed. It is only available if <i>Expose for Consumption</i> is enabled.</p> <p>For more information, see HINT Details in the <i>SAP HANA Cloud. SAP HANA Database SQL Reference Guide</i>.</p>
Status	<p>[read-only] Displays the deployment and error status of the object.</p> <p>For more information, see Saving and Deploying Objects [page 39].</p>

6. Based on the *Semantic Usage* of your entity, review and modify its *Columns*, *Attributes*, and/or *Measures*:
 - *Fact* - Review the lists of measures and attributes (see [Creating a Fact \[page 305\]](#)).
 - *Dimension* - Review the list of attributes (see [Creating a Dimension \[page 314\]](#)).
 - *Hierarchy* - Define the parent and child columns (see [Creating an External Hierarchy \[page 330\]](#)).
 - *Hierarchy with Directory* - Define all the necessary attributes and settings (see [Creating a Hierarchy with Directory \[page 331\]](#)).
 - *Text* - Review the list of attributes (see [Create a Text Entity for Attribute Translation \[page 327\]](#)).
 - *Relational Dataset* - Review the list of columns (see [Columns \[page 108\]](#)).
7. Complete or consult other sections as appropriate:
 - *Input Parameters* - Create input parameters to require the user to enter a value for use in calculated column, filter, and aggregation nodes (see [Create an Input Parameter \[page 231\]](#)).
 - *Data Persistence* - Persist the view data to improve performance (see [Persist View Data \[page 242\]](#)).
 - *Associations* - Create associations to other entities (see [Create an Association \[page 234\]](#)).
 - *Data Access Controls* - Add data access controls to apply row-based security and control access to individual rows based on various criteria (see [Securing Data with Data Access Controls](#)).
 - *Business Purpose* - Provide a description, purpose, contacts, and tags to help other users understand your entity.
 - *Dependent Objects* - If your entity is used as a source or association target for other entities, then they are listed here (see [Review the Objects That Depend on Your Table or View \[page 43\]](#)).

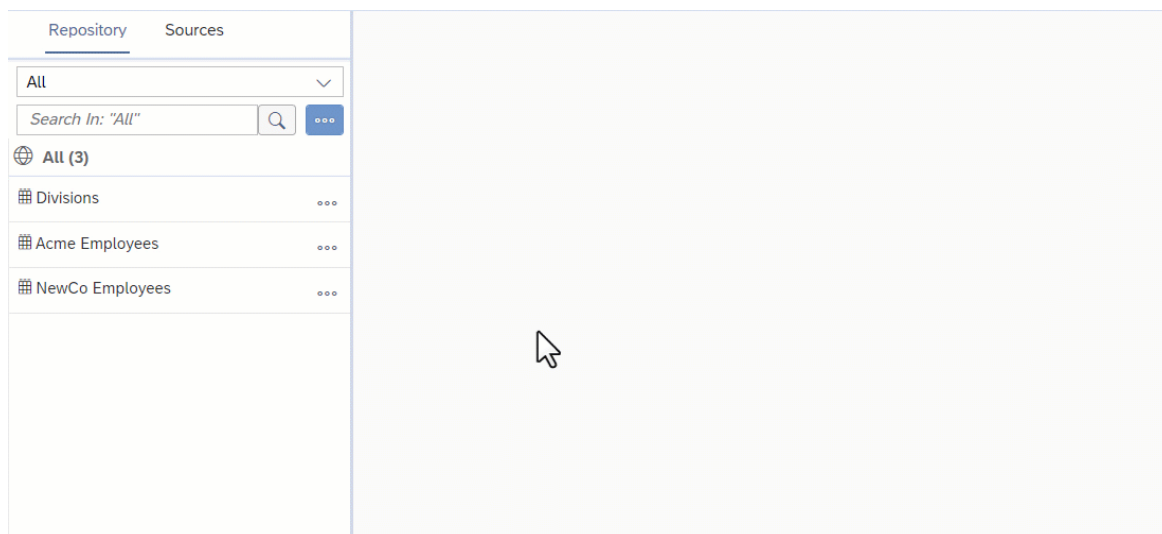
8. Click  (*Save*)  *Save*  to save your entity or click  (*Deploy*) to save and deploy it immediately. For more information, see [Saving and Deploying Objects \[page 39\]](#).
9. [optional] Click  (*Edit Custom CSN Annotations*) to open the *Edit Custom CSN Annotations* dialog. For more information, see [Edit a Custom CSN Annotation \[page 249\]](#)
10. Click  (*Impact and Lineage Analysis*) in the toolbar to understand the lineage and impacts of the output (see [Impact and Lineage Analysis \[page 34\]](#)).

6.1.1 Add a Source

Drag a table or view from the *Source Browser* panel and drop it on the diagram.

Procedure

1. If the *Source Browser* panel is not visible on the left of the screen, click *Source Browser* in the toolbar to show it.
2. Browse or search for the object you want to add on either of the tabs.
 - The *Repository* tab lists all the tables, views, and intelligent lookups that are available in the space (including objects shared to the space).. For more information, see [Add Objects from the Repository \[page 292\]](#).
 - The *Sources* tab lists all the connections and other data sources that have been integrated to the space from which you can import tables. However it shows only limited records. If you can't see the sources you are looking for, use *Import from Connection* to perform search. You can:
 - Expand the data sources to browse through their objects (see [Import an Object from a Connection or Other Source \[page 294\]](#)).
 - Open the *Import Objects from Connection* dialog on a particular connection to select multiple objects for import (see [Import Multiple Objects from a Connection \[page 296\]](#)).
3. Select your object and drag and drop it onto the diagram.



The source is added to the diagram, its symbol is selected, and its properties are displayed in the side

panel. In addition, an output node, initially entitled **view 1**, which represents the final output structure of the view, is created in the diagram and linked to the source.

Note

If you choose a table or view from the *Sources* tab, it is automatically imported into the repository and deployed, and will be available on the *Repository* tab for future use by you or others.

- If the source is a view containing one or more input parameters (see [Create an Input Parameter \[page 231\]](#)), the *Map Input Parameters* dialog is displayed, and you must decide how each input parameter will be processed:


- Map To** - Map the source input parameter to an input parameter in the view. You can select an existing input parameter or create a new one.
Users of this view will need to provide a value for the input parameter.
- Set Value** - Enter a value to resolve the input parameter.
The input parameter is resolved and users of this view will no longer need to provide a value for it.

Note

You can click *Cancel* in the dialog and not map the input parameters immediately, but then an error is displayed on the source, and you must subsequently map them in the source side panel *Input Parameters* section.

The source symbol displays the number of input parameters present in the source. In this example, the source view and output view both contain two input parameters::



- Click  (*Preview Data*) to open the *Data Preview* panel and review the data output by this node. For more information, see [Viewing or Previewing Data in Data Builder Objects \[page 297\]](#).

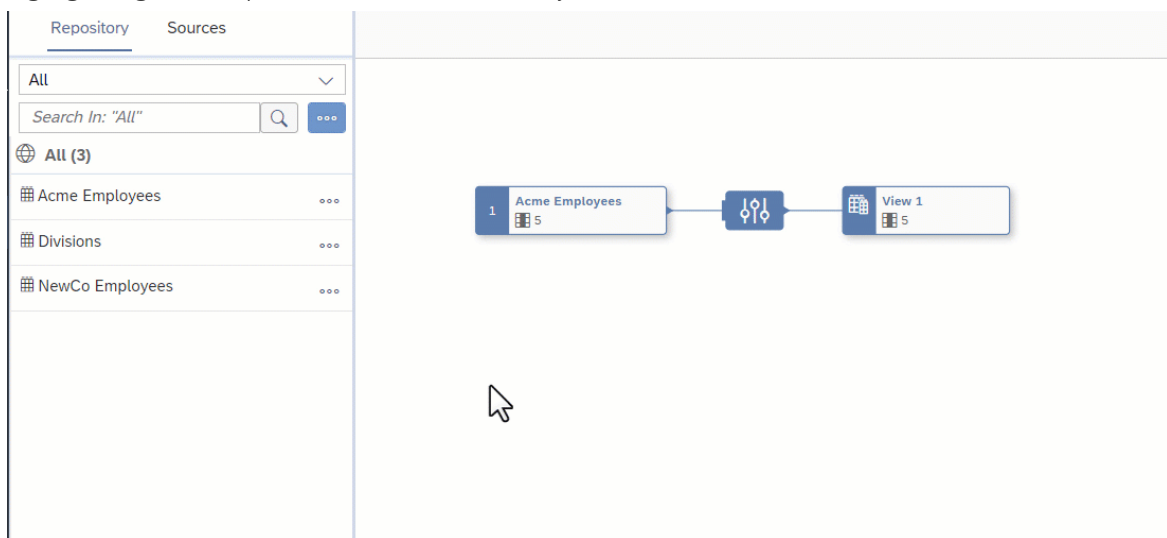
6.1.2 Create a Join

Drag a source from the *Source Browser* and drop it onto a source or other node in the diagram.

Procedure

- If the *Source Browser* panel is not visible on the left of the screen, click *Source Browser* in the toolbar to show it.
- Browse or search for the object you want to add on either of the tabs.

- The *Repository* tab lists all the tables, views, and intelligent lookups that are available in the space (including objects shared to the space).. For more information, see [Add Objects from the Repository \[page 292\]](#).
 - The *Sources* tab lists all the connections and other data sources that have been integrated to the space from which you can import tables. However it shows only limited records. If you can't see the sources you are looking for, use *Import from Connection* to perform search. You can:
 - Expand the data sources to browse through their objects (see [Import an Object from a Connection or Other Source \[page 294\]](#)).
 - Open the *Import Objects from Connection* dialog on a particular connection to select multiple objects for import (see [Import Multiple Objects from a Connection \[page 296\]](#)).
3. Select your object and drag it over a source or other node in the diagram. When the target node is highlighted green, drop the new source to create a join.




The source is added to the diagram and it is joined to the target node. The join symbol is selected and its properties are displayed in the side panel. A projection node is added after the join node to remove any duplicate columns.

Note

If you choose a table or view from the *Sources* tab, it is automatically imported into the repository and deployed, and will be available on the *Repository* tab for future use by you or others.

4. Optional. Rename the node in its side panel to clearly identify it. This name can be changed at any time and can contain only alphanumeric characters and underscores.
5. Set the following properties in the *General* section:

Property	Description
Join Type	<p>Specifies the type of SQL join to create. You can choose from:</p> <ul style="list-style-type: none"> • Cross - Return all of the rows from the left table joined to all of the rows from the right table. • Full - Return all of the rows from both tables, joining records from the left table where possible. • Inner - [default] - Return all of the rows in the left table that have a matching row in the right table. • Left - Return all of the rows from the left table (whether or not they have a match in the right table), and any matching rows from the right table. • Right - Return all of the rows from the right table (whether or not they have a match in the left table), and any matching rows from the left table.
Distinct Values	Return only unique values.
Cardinality	<p>Specifies the number of rows matching another table in a join and may improve the view's performance. You can choose from:</p> <ul style="list-style-type: none"> • MANY TO ONE • MANY TO EXACT ONE • MANY TO MANY • ONE TO ONE • ONE TO EXACT ONE • ONE TO MANY • EXACT ONE TO ONE • EXACT ONE TO EXACT ONE • EXACT ONE TO MANY <p>The cardinality is integrated in the code and is visible in the SQL preview.</p>

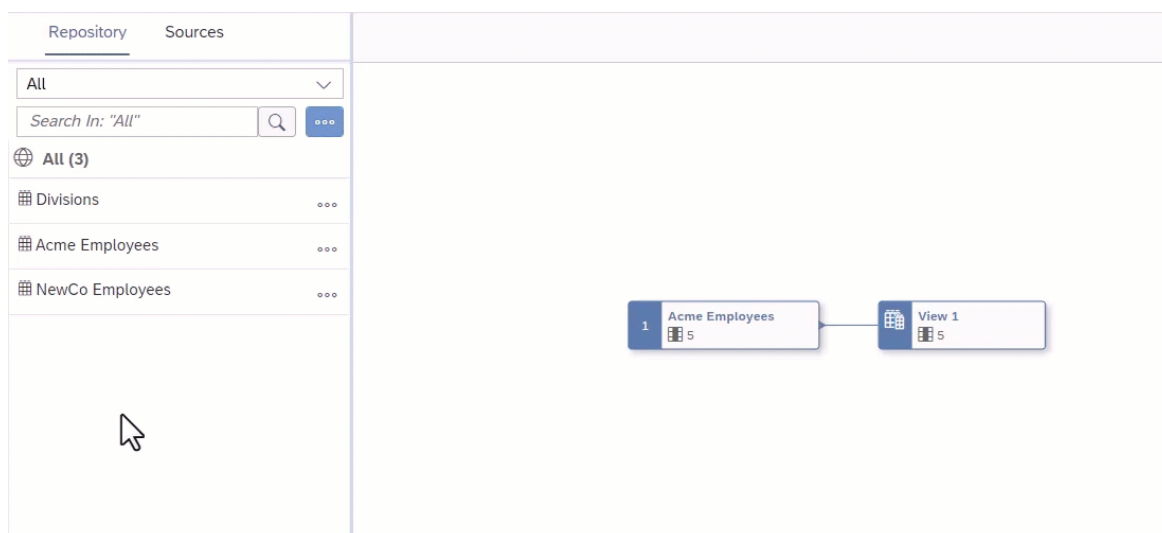
- Specify the mapping of join columns in the *Mappings* section:
 - A mapping is automatically created by matching column names if possible.
 - To manually map columns, drag a column from the left list and drop it onto a column in the right list.
 - To delete a mapping, click the link and then click the *Delete* tool.
 - You can filter the *Mappings* section to show only mapped or unmapped pairs of columns.
 - You can filter or sort the left or right column lists independently
- Click  (*Preview Data*) to open the *Data Preview* panel and review the data output by this node. For more information, see [Viewing or Previewing Data in Data Builder Objects \[page 297\]](#).

6.1.3 Create a Union

Drag a source from the *Source Browser*, hover over a source or other node, and click *Union*. A union combines the results from two select statements on separate sources.

Procedure

1. If the *Source Browser* panel is not visible on the left of the screen, click *Source Browser* in the toolbar to show it.
2. Browse or search for the object you want to add on either of the tabs.
 - The *Repository* tab lists all the tables, views, and intelligent lookups that are available in the space (including objects shared to the space).. For more information, see [Add Objects from the Repository \[page 292\]](#).
 - The *Sources* tab lists all the connections and other data sources that have been integrated to the space from which you can import tables. However it shows only limited records. If you can't see the sources you are looking for, use *Import from Connection* to perform search. You can:
 - Expand the data sources to browse through their objects (see [Import an Object from a Connection or Other Source \[page 294\]](#)).
 - Open the *Import Objects from Connection* dialog on a particular connection to select multiple objects for import (see [Import Multiple Objects from a Connection \[page 296\]](#)).
3. Select your object and drag it over a source or other node in the diagram. Wait for the context menu to appear, slide your cursor over the *Union* option and then release the mouse button.




The source is added to the diagram and it is unioned with the target node. The union symbol is selected and its properties are displayed in the side panel. You can add multiple tables to the union by dragging and dropping them on the union symbol.

Note

- The union node should come last as you aren't able to add another object to an output after a union. If you do, you will experience technical issues. You can add sources to the nodes preceding the union node.
- If you choose a table or view from the *Sources* tab, it is automatically imported into the repository and deployed, and will be available on the *Repository* tab for future use by you or others.

4. Optional. Rename the node in its side panel to clearly identify it. This name can be changed at any time and can contain only alphanumeric characters and underscores.
5. Set the following properties in the *General* section:

Property	Description
Union All	Default option and fastest to create. It combines two or more <code>SELECT</code> statements or queries and includes all rows, including duplicates. Disabling this option applies <i>Union</i> .
Union	It combines the result set of two or more <code>SELECT</code> statements or queries (only distinct values) and returns fewer rows. It takes longer to create it because it removes duplicate rows.

6. The *Mappings* section shows, by default, the recently dropped source object columns on the left and the union output columns, initialized from the target source columns on the right.
 - Mappings are automatically created by matching column names where possible.
 - To manually map input columns to union output columns, drag a column from the left list and drop it onto a column in the right list.
 - To delete a mapping, click the link and then click the *Delete* tool.
 - You can filter the *Mappings* section to show only mapped or unmapped pairs of columns.
 - You can filter or sort the left or right column lists independently.
 - You can modify the list of union output columns using the Menu above the right list:
 - *Add All Source Columns as Union Columns* - Add all columns in the left list to the right list (if they are not already present).
 - *Add Selected Source Columns as Union Columns* - Add columns selected in the left list to the right list (if they are not already present).
 - *Delete Selected Union Columns* - Delete any columns selected in the right list so that they are no longer union output columns.
 - *Delete All Union Columns* - Delete all columns in the right list so that they are no longer union output columns.
 - To switch to viewing another input, click the drop-down arrow above the left column list and select it in the list.
7. Optional. Drop another source object onto the union node to union it with the existing unioned sources. The source is added to the diagram and added to the union node. The union symbol is selected, its properties are displayed in the side panel, the new source is displayed in the *Mappings* section and its columns are mapped automatically by matching column names where possible.
8. Click  (*Preview Data*) to open the *Data Preview* panel and review the data output by this node. For more information, see [Viewing or Previewing Data in Data Builder Objects \[page 297\]](#).

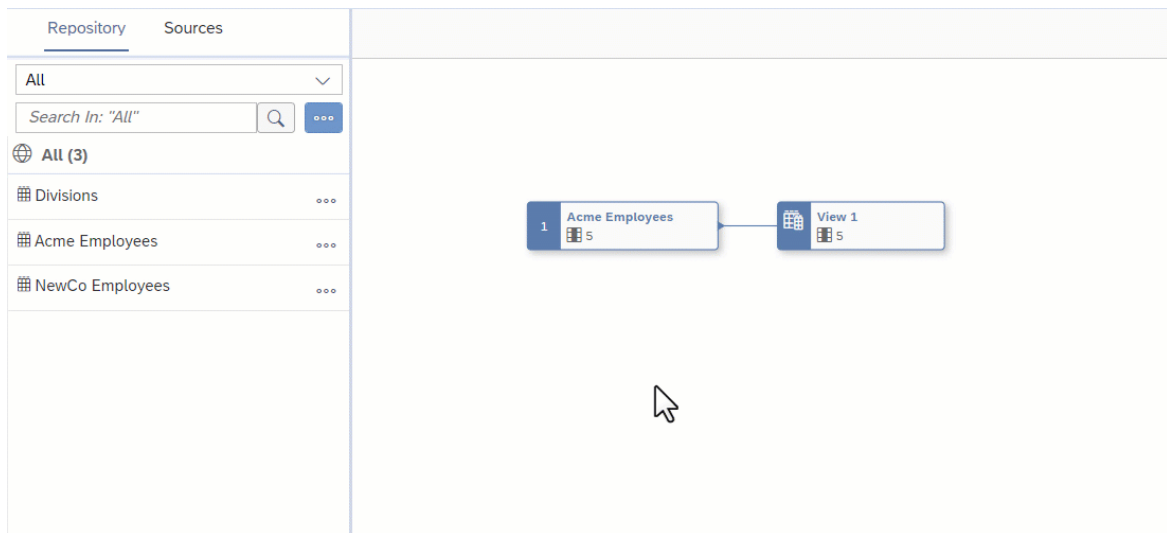
Next Steps

6.1.4 Reorder, Rename, and Exclude Columns

Add a *Projection* node to rename, reorder, or exclude columns.

Procedure




1. Select a node in order to display its context tools, and click  (*Rename/Exclude Columns*).




A projection node is created, its symbol is selected, and its properties are displayed in the side panel.

2. Optional. Rename the node in its side panel to clearly identify it. This name can be changed at any time and can contain only alphanumeric characters and underscores.
3. The *Columns* list displays the columns that are available. You can filter the list with the *Search* box and reorder it.
4. Modify the *Columns* list as appropriate. You can:
 - Reorder a column by dragging and dropping it in the list below *Columns*.
 - Rename a column by clicking *...* (*Menu*), and selecting *Change Name* to open the *Change Name* dialog. Edit the business and/or technical names as appropriate and click *Save*.

Note

By default, SAP Datasphere displays the business names of your objects. To show technical names instead, click  *Profile*  *Settings* , select the *UI Settings* tab, and then select *Show Technical Name*.

- Exclude a column by clicking its *...* (*Menu*) button, and selecting *Exclude Column*. To select multiple columns for exclusion, press `Ctrl` while selecting them. Alternatively, click *Select All* and press `Ctrl` while deselecting columns.

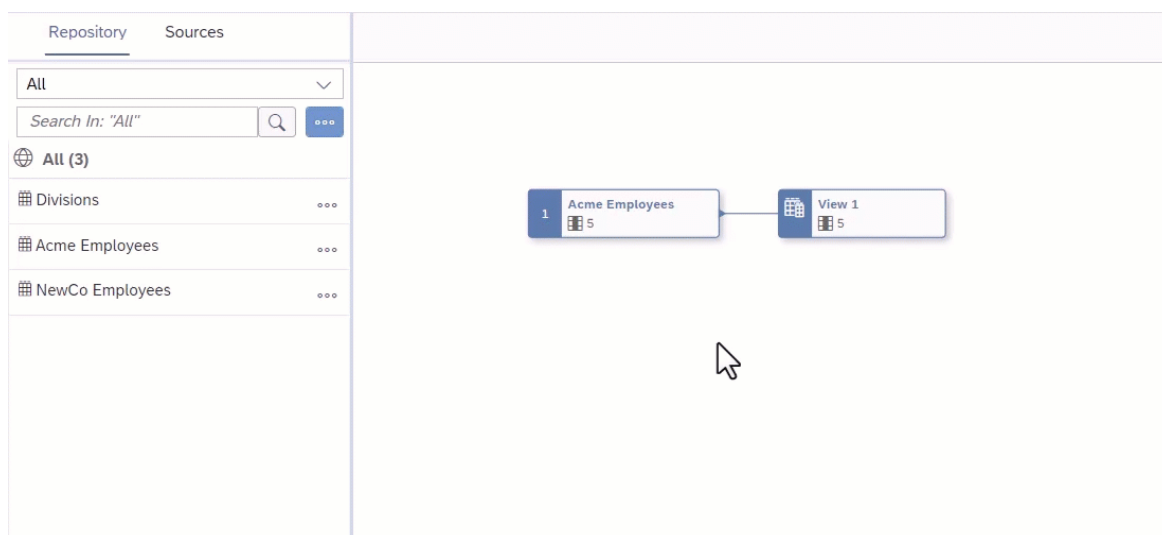
- Click  (*Preview Data*) to open the *Data Preview* panel and review the data output by this node. For more information, see [Viewing or Previewing Data in Data Builder Objects \[page 297\]](#).

6.1.5 Create a Column


Add a *Calculated Columns* node to create new columns and define calculations in them.

Procedure

- Select an object in order to display its context tools, and click  *Calculated Columns*.



A calculated column node is created, its symbol is selected, and its properties are displayed in the side panel.

- Click  (*Add New Calculated Column*) to create a new column. The new column properties are displayed in the side panel.
- Enter a *Business Name* and a *Technical Name* for the column, and specify its *Data Type*.
- Enter a SQL expression into the *Expression* field. You can use the following items in your SQL expression:
 - Insert Values* - Click to open the *Insert Value* dialog, select the values you need to insert in the list, and click *Insert Value*.

Note

- The button is enabled when, in the *Expression* field, a column name is followed by the operator =, >, <, >=, <=, !=, IN, BETWEEN, or LIKE.
- The values listed in the dialog are retrieved from the sources and intermediate nodes of your object. You can insert values of the data types string, integer, boolean, date, and time. The data types binary and UUID aren't supported.

- Once your expression is valid, additional values cannot be inserted. You must edit your expression to make it incomplete; this enables the *Insert Values* button and allows you to insert other values.
- The *Insert Values* dialog doesn't format the expression. Manual formatting may be required after inserting values.

❖ Example




- The expression `Column1 =` enables the *Insert Values* dialog. You can select a single value from the list.
- The expression `Column2 BETWEEN` enables the *Insert Values* dialog. You can select two values from the list. If both values are available in a single search results, both of them can be selected together. However, if the values need to be selected from two different search results, you have to select and insert one value at once.
- The expression `Column3 IN` enables the *Insert Values* dialog. You can select multiple values from the list. If all the required values are available in a single search results, those values can be selected together. However, if the values have to be selected from different search results, you have to select and insert one value at once. You select and insert the values 2, 5, 6, and 9. The expression `Column3 IN (2, 5, 6, 9)` is valid. You want to add another value to it, so you click *Insert Values*. In the *Insert Values* dialog, you can only see the values 2, 5, 6, and 9 because the dialog is already filtered by the selected values. To see all available values, edit your expression to make it invalid: `Column3 IN (2, 5, 6, 9, .`. The *Insert Values* button is available again and you can add new values.

Select available value(s) and click *Insert* to add them to your expression.

- *Functions* - Browse, select a category, or filter available functions (see [SQL Functions Reference \[page 257\]](#)). Click a function name to see its syntax or click elsewhere in its token to add it to your expression.
- *Columns* - Browse or filter available columns. Click a column name to see its properties or click elsewhere in its token to add it to your expression.
- *Parameters* - Browse or filter available input parameters (see [Create an Input Parameter \[page 231\]](#)). Click a parameter name to see its properties or click elsewhere in its token to add it to your expression.
- *Other* - Browse available operators, predicates, and case expressions, and click one to add it to your expression (see [SQL Reference \[page 255\]](#)).

For example, if you want to calculate the price of a product minus a 15% discount, enter `Price*0.85`.

When working on a large expression, click  (*Enter Full Screen*) to expand the expression editor.

5. Click *Validate* to check if your expression is semantically correct, and fix any errors signaled. You can reference columns by name as well as HANA functions, operators, predicates, and case expressions. When you are satisfied, click the breadcrumbs at the top of the side panel to drill back up to the calculated column node properties.
6. The list displays all the columns output by the node. You can:
 - Filter the list with the *Search* box (or by clicking  *Hide Unmodified Columns*) and reorder it.
 - Modify the expression output by any column by clicking on the chevron on the right of its token.
 - Delete a column that has been created in the node by clicking  *Delete Selected Calculated Column*.
7. Click  (*Preview Data*) to open the *Data Preview* panel and review the data output by this node. For more information, see [Viewing or Previewing Data in Data Builder Objects \[page 297\]](#).

6.1.6 Create a Geo-Coordinates Column

Create a geo-coordinates column to combine latitude and longitude values and output coordinate data that can be used in an SAP Analytics Cloud geo map.

Context

You can create a geo coordinates only in a graphical view with a semantic usage of *Dimension*. To use the geo-coordinates column in SAP Analytics Cloud you must:


- Create a view with a semantic usage of *Fact* and enable *Expose for Consumption*.
- Create an association in your *Fact* that points to your *Dimension*.

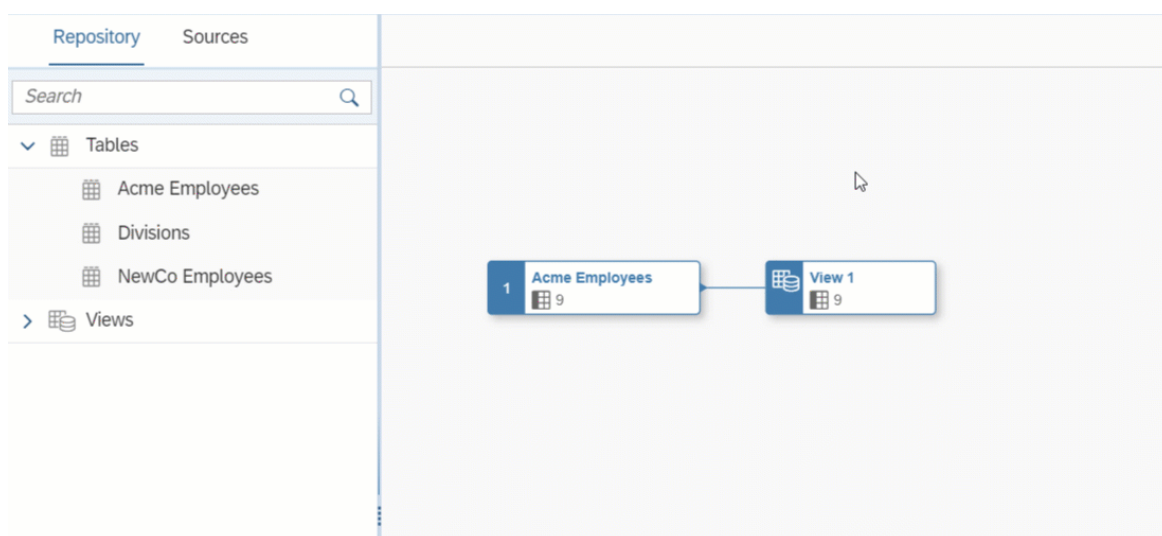
When using the geo-coordinates column in an SAP Analytics Cloud geo map, the following types of layers are supported:

- Bubble
- Heat Map
- Choropleth / Drill
- Feature
- Flow

For more information see [Creating a Geo Map in SAP Analytics Cloud](#).

Procedure

1. Ensure that the following properties are set correctly for your view output node:
 - *Semantic Usage - Dimension*
 - *Expose for Consumption - Enabled*
2. Select any object in your graphical view to display its context tools and click  *Calculated Columns*.



A calculated column node is created, its symbol is selected, and its properties are displayed in the side panel.

- Click **+** (*Add New Calculated Column*) > **📍** (*Geo-Coordinates Column*) to create a new geo-coordinates column.

The new column properties are displayed in the side panel. The column is named *Location* by default.

- Complete the column's properties:

Property	Description
Business Name	Enter a descriptive name to help users identify the object. This name can be changed at any time.
Technical Name	<p>Displays the name used in scripts and code, synchronized by default with the <i>Business Name</i>.</p> <p>To override the default technical name, enter a new one in the field. Technical names can contain only alphanumeric characters and underscores.</p> <div style="border: 1px solid #ccc; background-color: #f0f0f0; padding: 5px; margin-top: 10px;"> <p>Note</p> <p>Once the object is saved, the technical name can no longer be modified.</p> </div>
Spatial Reference Identifier	Every spatial reference system has an identifier called a <i>Spatial Reference Identifier</i> (or <i>SRID</i>). By default, it is set to <i>4326</i> but can be changed to <i>0</i> or <i>3857</i> . For more information, see Spatial Reference Identifier on SAP HANA .
Latitude and Longitude	These columns must have string or double as data types.

- Click **📁** (*Save*) >> **Save** to save your entity or click **🚀** (*Deploy*) to save and deploy it immediately. For more information, see [Saving and Deploying Objects \[page 39\]](#).
- Open a view that will point to your *Dimension* with an association and set its properties as follows:
 - Semantic Usage - Fact*
 - Expose for Consumption* - Enabled
 - Create an association and point to your dimension.

6.1.7 Create a Currency Conversion Column

You can convert currency values into another currency using a *Currency Conversion Column*.

Context

To perform currency conversion, the following tables must be available in your space:

- TCURV - Exchange rate types
- TCURW - Exchange rate type text
- TCURX - Decimal places in currencies

- TCURN - Quotations
- TCURR - Exchange rates
- TCURF - Conversion factors
- TCURC - Currency codes
- TCURT - Currency text

For more information about the creation of the tables, see [Enabling Currency Conversion with TCUR* Tables and Views \[page 52\]](#).

Note



SAP Analytics Cloud can consume views performing currency conversion only if the following restrictions are respected:


- If the TCUR* tables are shared to your space from another space, then you must encapsulate each table in its own view (which can have the same name as the shared table) and expose each of these encapsulating views for consumption.
- Views containing columns that perform currency conversion using the TCUR* tables cannot be shared to other spaces.

Procedure

1. In a *Calculated Columns* node, click **+** (*Add New Calculated Column*) > **💰** (*Currency Conversion Column*) to create a new *Currency Conversion* column.
The new column properties are displayed in the side panel.
2. Enter a *Business Name* and a *Technical Name* for the column, and specify an appropriate numerical *Data Type*.
3. Use the *Currency Properties* section to set the properties of your source and target currencies.

Property	Description
Source Amount Column	Select the column identifier containing the values to be converted.

Property	Description
Steps	<p>Define the steps that should be included in the conversion process:</p> <ul style="list-style-type: none"> • <i>Shift</i> - enables a decimal shift according to the source currency selected. • <i>Convert</i> - triggers the actual conversion from the source to the target currency. • <i>Round</i> - rounds the converted value to the number of digits of the target currency. Use this step carefully if subsequent aggregations take place on the number, as rounding errors could accumulate. • <i>Shift Back</i> - while shift changes the decimals from two to the configured precision of the source currency, <i>Shift Back</i> changes them back to two, but from the target currency. If error handling is set to keep unconverted the output currency might be the source instead of the target currency. In case of an error, the rounding and the <i>Shift Back</i> are done with respect to the source currency and the conversion is dropped. This renders all steps redundant, and yields the input value again.
Source Currency	<p>Select the column describing the input unit. Click  and choose a type:</p> <ul style="list-style-type: none"> • <i>Column</i> - Select a column and click <i>Select</i>. • <i>Fixed Value</i> - Select a fixed value and click <i>Select</i>. A fixed value is a constant of the type string.
Target Currency	<p>Select the column describing the target unit. Click  and choose a type:</p> <ul style="list-style-type: none"> • <i>Column</i> - Select a column and click <i>Select</i>. • <i>Fixed Value</i> - Select a fixed value and click <i>Select</i>. A fixed value is a constant of the type string. • <i>Input Parameter</i> - Select an Input Parameter and click <i>Select</i>. You can also directly enter the technical name in the currency conversion column's properties panel.

Property	Description
Reference Date	<p>Enter the column describing the currency reference date. Click  and choose a type:</p> <ul style="list-style-type: none"> • <i>Column</i> - Select a column and click <i>Select</i>. • <i>Fixed Date</i> - Select a date on the calendar or enter it in the format YEAR-MONTH-DAY. For example, for the 1st of December 2021, enter 2021-12-01. • <i>Input Parameter</i> - Select an Input Parameter and click <i>Select</i>. You can also directly enter the technical name in the currency conversion column's properties panel. • <i>Expression</i> - Enter a SQL expression into the <i>Expression</i> field. Browse, select a category, or filter available <i>Date & Time Functions</i> (see SQL Functions Reference [page 257]). Click a function name to see its syntax or click elsewhere in its token to add it to your expression. Click <i>Validate</i> to check if your expression is semantically correct, and fix any errors signaled. For example,

4. Use the [Conversion Properties](#) section to refer to table columns for the conversion.

Property	Description
Client	<p>Enter a three character string that is used to separate tenants within ERP system tables. This is used in the conversion tables to select the correct rows for each user.</p> <p>This parameter is mandatory, as the CLIENT session context variable is not used by this command.</p>
Conversion Type	<p>Define the conversion type as stored in the conversion tables. The conversion types available in your system vary according to the setup of your ERP system. In general, these are either M or EURX. Contact your system administrator for the details of your specific table configuration.</p>
Error Handling	<p>Select a value to define how the system handles a situation where a row could not be converted. All mandatory values must be provided beforehand:</p> <ul style="list-style-type: none"> • Fail on Error - the conversion fails with an error. • Set to Null - the output from the row that caused the error is set to NULL. • Keep Unconverted - the input value is returned.

5. Use the [Advanced Properties](#) section to define constant parameter values used in the conversion.

Property	Description
Lookup	<p>Define the type of lookup to be performed:</p> <ul style="list-style-type: none"> • Regular - a regular conversion is performed. • Reverse - performs a conversion with the input units swapped.

Property	Description
Accuracy	Defines the rounding behavior of the system: <ul style="list-style-type: none"> • Compatibility - mimics ERP behavior by rounding the result. • Highest - keeps as many digits as possible in the result.
Date Format	Defines the format in which the reference date is presented: <ul style="list-style-type: none"> • Auto Detect - attempt automatic detection of the date format. • Normal - date is provided in a regular format. • Inverted - date is provided in inverted SAP legacy format.
Precision Entity	Select the entity identifier of the precision entity.
Configuration Entity	Select the entity identifier of the conversion type configuration.
Prefactors Entity	Select the entity identifier of the pre-factors entity.
Rates Entity	Select the entity identifier of the conversion rates entity.
Notations Entity	Select the entity identifier of the entity that stores notations.

Note

You can also create a Currency Conversion Column using the `CONVERT_CURRENCY` function in a Calculated Column's *Expression* field. For full documentation of this function, see [CONVERT_FUNCTION](#) in the *SAP HANA SQL Reference Guide for SAP HANA Platform*.

6. Click *Validate* once all properties are set to check if they are correct, and fix any error signaled. When you are satisfied, open or refresh the *Data Preview* panel to review the results of your currency conversion in your new column.

6.1.8 Create an Input Parameter

Create an input parameter to prompt the user to enter a value for use in filtering or other calculations when the view is run. If the view is consumed in an SAP Analytics Cloud story the user will be prompted to enter a value for the input parameter in the *Set Variables* dialog.

Context

You can use input parameters when creating graphical and SQL views:



- In the graphical view editor, you can use input parameters to require user input in any sql expression, including those used in filter, calculated columns, and aggregation nodes.


- In the SQL view editor, you can use input parameters in your `WHERE` clause or anywhere else in your SQL code.

Views that contain input parameters require special treatment in the following situations:

- Previewing data - Accept the default value, if one is provided, or enter a value for each input parameter (see [Viewing or Previewing Data in Data Builder Objects \[page 297\]](#)).
Previewing a SQL view isn't possible if one of the view's objects is shared from another space and has an input parameter.
- Adding a view as a source for a graphical view - Map each input parameter in the source view to a value or an input parameter in the new view (see [Add a Source \[page 217\]](#)).
- Adding a view as a source for a SQL view - Complete the syntax to map each input parameter in the source view to a value or an input parameter in the new view (see [Process Source Input Parameters in an SQL View \[page 254\]](#)).
- Adding a table as a source for an analytic model - Map each input parameter in the source table to a variable in the model (see [Add a Variable \[page 348\]](#)).
- Consuming view data via the OData API - Set a value for each input parameter using the `(<param>='<val>' [, ...]) /Set` syntax (see [Consume Data via the OData API](#)).
- Using a view as a data source in an SAP Analytics Cloud story - Enter a value for each input parameter in the *Set Variables* dialog (see [Setting Story Variables](#) in the *SAP Analytics Cloud Help Library*).

Procedure

1. Select the output node of your view to display its properties in the side panel, scroll down to the *Input Parameters* section, and click  (*Input Parameters*).
2. In the *Input Parameters* dialog, click  (*Add*) to create a new parameter.
3. Complete the properties of the input parameter:

Property	Description
Business Name	Enter a descriptive name to help users identify the object. This name can be changed at any time.
Technical Name	Displays the name used in scripts and code, synchronized by default with the <i>Business Name</i> . To override the default technical name, enter a new one in the field. Technical names can contain only alphanumeric characters and underscores.
<div style="background-color: #f0f0f0; padding: 10px; border: 1px solid #ccc;"> <p> Note</p> <p>Once the object is saved, the technical name can no longer be modified.</p> </div>	
Data Type	Specify the type of data that the input parameter will contain (see Column Data Types [page 110]), and which should be appropriate for the contexts in which it will be used. Some data types require you to specify a length and or precision.

Property	Description
Default Value	[optional] Enter a default value for the input parameter. Each time that a user is required to enter a value for the parameter, they can accept the default value or override it with their own chosen value.
Default Value Help	<p>Define if your input parameter needs a default value help or not:</p> <ul style="list-style-type: none"> • If not, select <i>No Value Help</i>. • If yes, select <i>Predefined Value from an Object</i>. The predefined value comes from an attribute from any table or view of the type dimension. Once this option is selected, you must define: <ol style="list-style-type: none"> 1. <i>Object</i>: Click to open the <i>Add Objects</i> dialog. Select an object among the list of available items and click <i>Add Object</i>. 2. <i>Column</i>: Click to open the <i>Value Help Dialog</i> and see the list of available columns from which you want to propagate the default value. <p>All records are loaded by default. You can see more by using the <i>Search</i> bar or scrolling down. You may experience performance issues if too many records are to be loaded.</p>

4. Click *OK* to close the dialog and return to the side panel, where the input parameter is available for use. The output symbol displays the number of input parameters present in the view. In this example, the source view and output view both contain two input parameters::



5. Use the input parameter as appropriate in your view:
- In graphical views, input parameters can be used in the following nodes:

Example	Description
<code>Country = :IP_Country</code>	<p>Filter node (see Filter Data [page 236])</p> <p>This filter expression takes the value entered by the user for <code>IP_Country</code> and uses it to restrict the data returned by the view to only those rows where the <code>Country</code> column contains this value.</p>
<code>Total * ((100 - IP_Discount) / 100)</code>	<p>Calculated Columns node (see Create a Column [page 224])</p> <p>This expression will fill the column with a value calculated by taking the value in the <code>Total</code> column and deducting the percentage value entered by the user.</p>
<code>SUM(Revenue) > IP_Min_Rev</code>	<p>Aggregation node (see Aggregate Data [page 238])</p> <p>This <code>HAVING</code> expression will limit the values aggregated to only those with a total <code>Revenue</code> over the value entered by the user.</p>

- In SQL views, input parameters can be used in your `WHERE` statement and any other appropriate context.

In this example, two input parameters are defined and used:

- `IP_DISCOUNT` - is used to calculate the new column, `Discounted Sale`.
- `IP_CITY` - is used in the `WHERE` clause to prompt the user to specify a city to filter on.

```
SELECT "ID" ,
       "Date",
       "City",
       "Product",
       "Customer",
       "Net Sale",
       "Gross Sale",
       "Quantity",
       "Net Sale" * ((100 - :IP_DISCOUNT) / 100) AS "Discounted Sale"
FROM "Sales"
WHERE "City" = :IP_CITY
```

Note

You must use each of the input parameters that you create at least once in your view or you will receive an error instructing you to use or delete them.

6.1.9 Create an Association

Click the [Create](#) button in the [Associations](#) section of the side panel of your table or view to create an association to another data entity. You can create an association from any table or view to any other table or view at any level of the data layer, including to define the relationships between facts and dimensions among your consumable views.

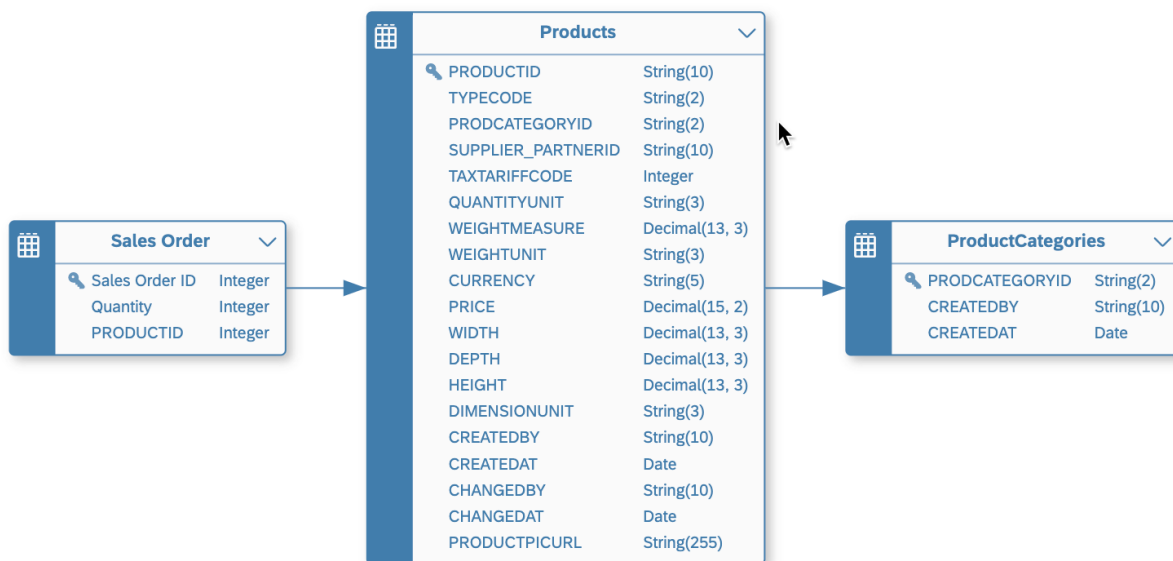
Context

You can create associations in the:

- E/R model editor (for any table or view), in the side panel [Associations](#) section or by drawing associations in the diagram (see [Create an Association in an E/R Model Diagram \[page 289\]](#)).
- Table editor [Associations](#) section.
- Graphical view editor/SQL view editor, in the output node [Associations](#) section.

In this example:

- The `Sales Order` table has a semantic usage of *Fact* and has an association from its `Product ID` column to the `Product ID` key of the `Products` table, which is a *Dimension*.
- The `Products Dimension` has an association from its `PRODCATEGORYID` column to the `PRODCATEGORYID` key of the `Product Categories` table, which is also a *Dimension*.




Procedure


1. Open the table or view that will be the source of your association (or select it in an E/R model to open its *Properties* panel).
2. In the *Associations* section, click **+** (*Create Association*) to open the *Select Object* dialog. Find available objects by entering the object's name in the search bar or click **∨** (*Show filters*) and filter by *Semantic Usage* or other criteria.
3. Select the appropriate target data entity from the list and click *OK* to create the association and open it in the side panel.


The rules for creating associations depend on the *Semantic Usage* of the entity:

- A *Fact* can point to a:
 - *Dimension* - One attribute in the (source) *Fact* must be mapped to each (target) *Dimension* key column so that all target key columns are mapped.
 - *Text Entity* - An attribute in the (source) *Fact* must be mapped to the (target) *Text Entity* identifier key column.
- A *Dimension* can point to a:
 - *Dimension* - One attribute in the (source) *Dimension* must be mapped to each (target) *Dimension* key column so that all target key columns are mapped.
 - *Text Entity* - An attribute in the (source) *Dimension* must be mapped to the (target) *Text Entity* identifier key column.
 - *Hierarchy* - The key attribute in the (source) *Dimension* must be mapped to the (target) *Hierarchy* child attribute key column.
- A *Text Entity* must not point to other entities.
- A *Hierarchy* will generally not point to other entities.
- A *Hierarchy with Directory* must point to:

- A *Dimension* acting as its directory - The hierarchy name attribute in the (source) hierarchy entity must be mapped to the primary key column in the (target) dimension.
 - Any non-leaf *Dimension* providing nodes to the hierarchy - The appropriate node type values columns in the (source) hierarchy must be mapped to the key columns in the (target) *Dimension*.
 - A *Relational Dataset* can point to any other entity and should generally follow the rules for dimensions.
4. In the *General* section, review the default *Business Name* and *Technical Name* and modify them if appropriate.
 5. Specify the mapping of join columns in the *Join* section:
 - A default mapping is automatically created by matching column names if possible. For example if the originating entity contains a column, **Product ID**, and the target entity has a column with the same name, then a default mapping is created between these two columns.
 - To delete a mapping, select the link and then click  (*Delete*).
 - To manually map columns, drag a column from the left list and drop it onto a column in the right list.
 - You can filter the *Join* section to show only mapped or unmapped pairs of columns.
 - You can filter or sort the left or right column lists independently

Note

To delete an association, select it in the list and click  (*Delete Association*).

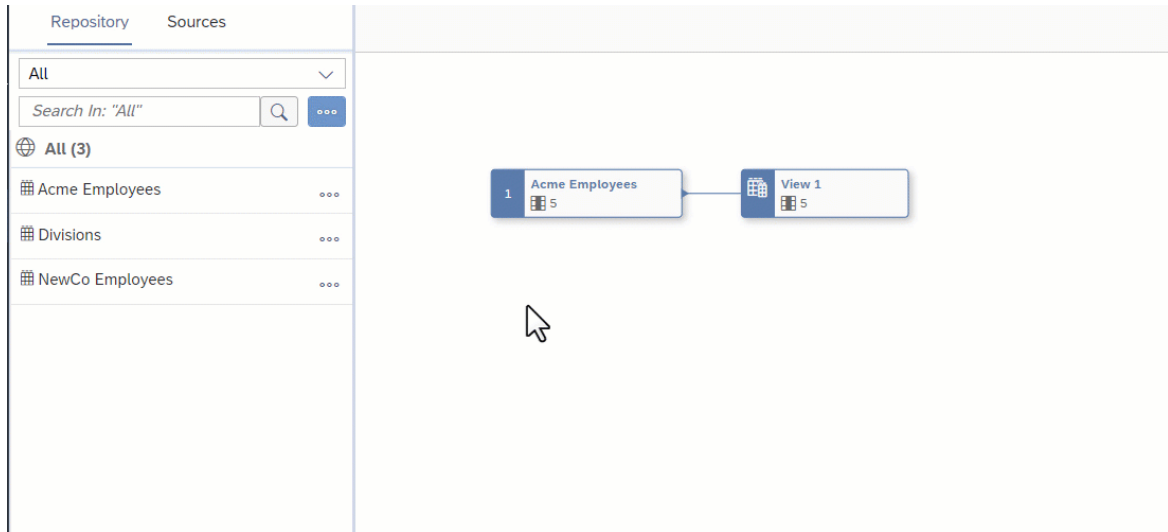
6. If the view builds on another source view that itself has an association, you can copy that association:
 1. Click  (*Create Association*) > *Copy from Source*. The *Copy Associations from Source* dialog opens and shows a table containing all source associations available to copy.
 2. Select the association(s) to copy and click *Create*. The new association(s) is added in the *Association* section in the output node's side panel.

6.1.10 Filter Data

Add a *Filter* node to filter your data with an SQL expression.

Procedure

1. Select an object in order to display its context tools, and click  *Filter*.



A filter node is created, its symbol is selected, and its properties are displayed in the side panel.

2. Optional. Rename the node in its side panel to clearly identify it. This name can be changed at any time and can contain only alphanumeric characters and underscores.
3. Enter a SQL expression into the *Expression* field. You can use the following items in your SQL expression:
 - *Insert Values* - Click to open the *Insert Value* dialog, select the values you need to insert in the list, and click *Insert Value*.

Note

- The button is enabled when, in the *Expression* field, a column name is followed by the operator =, >, <, >=, <=, !=, IN, BETWEEN, or LIKE.
- The values listed in the dialog are retrieved from the sources and intermediate nodes of your object. You can insert values of the data types string, integer, boolean, date, and time. The data types binary and UUID aren't supported.
- Once your expression is valid, additional values cannot be inserted. You must edit your expression to make it incomplete; this enables the *Insert Values* button and allows you to insert other values.
- The *Insert Values* dialog doesn't format the expression. Manual formatting may be required after inserting values.

Example

- The expression `Column1 =` enables the *Insert Values* dialog. You can select a single value from the list.
- The expression `Column2 BETWEEN` enables the *Insert Values* dialog. You can select two values from the list. If both values are available in a single search results, both of them can be selected together. However, if the values needs to be selected from two different search results, you have to select and insert one value at once.
- The expression `Column3 IN` enables the *Insert Values* dialog. You can select multiple values from the list. If all the required values are available in a single search results, those values can be selected together. However, if the values have to be selected from different search results, you have to select and insert one value at once. You select and insert the values 2, 5, 6, and 9. The expression `Column3 IN (2, 5, 6, 9)` is valid. You want to add another value to it, so you click *Insert Values*. In the *Insert Values* dialog, you can only


see the values 2, 5, 6, and 9 because the dialog is already filtered by the selected values. To see all available values, edit your expression to make it invalid: `column3 IN (2, 5, 6, 9, .` . The *Insert Values* button is available again and you can add new values.

Select available value(s) and click *Insert* to add them to your expression.

- *Functions* - Browse, select a category, or filter available functions (see [SQL Functions Reference \[page 257\]](#)). Click a function name to see its syntax or click elsewhere in its token to add it to your expression.
- *Columns* - Browse or filter available columns. Click a column name to see its properties or click elsewhere in its token to add it to your expression.
- *Parameters* - Browse or filter available input parameters (see [Create an Input Parameter \[page 231\]](#)). Click a parameter name to see its properties or click elsewhere in its token to add it to your expression.
- *Other* - Browse available operators, predicates, and case expressions, and click one to add it to your expression (see [SQL Reference \[page 255\]](#)).

For example, if you want to list only those products with 10 units or less in stock, enter `units_on_hand <= 10`

When working on a large expression, click  (*Enter Full Screen*) to expand the expression editor.

4. Click *Validate* to check if your expression is semantically correct, and fix any errors signaled. You can reference columns by name as well as HANA functions, operators, predicates, and case expressions.
5. Click  (*Preview Data*) to open the *Data Preview* panel and review the data output by this node. For more information, see [Viewing or Previewing Data in Data Builder Objects \[page 297\]](#).

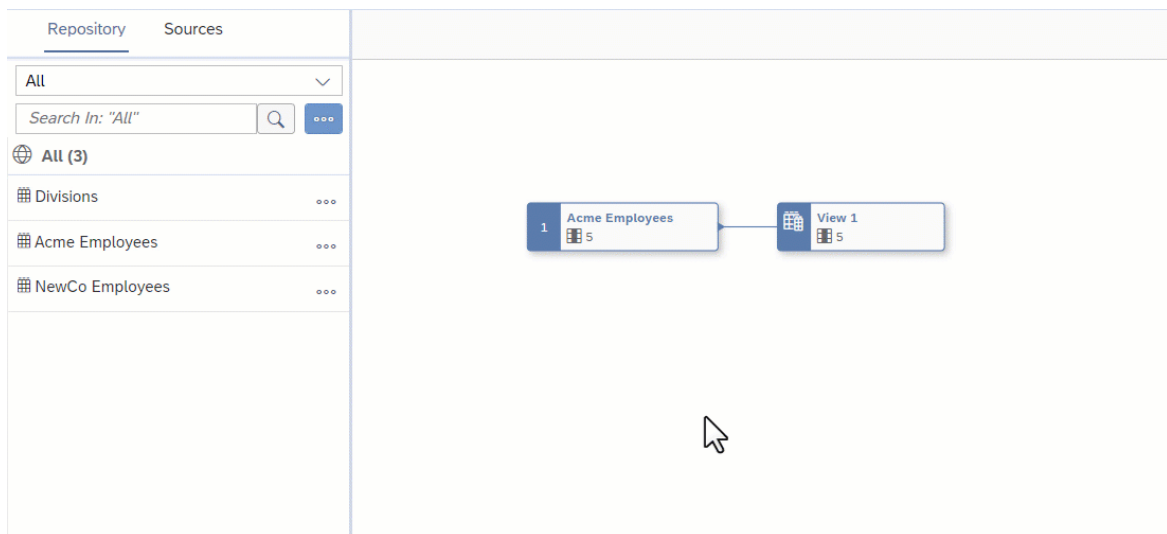
6.1.11 Aggregate Data

Add an *Aggregation* node to perform SUM, COUNT, MIN, and MAX calculations. You group by any non-aggregated columns and can optionally filter by specifying a HAVING clause.

Context

Procedure

1. Identify one or more columns containing data that you want to aggregate. The columns must have a numerical data type and can be calculated columns.
2. Create a projection node to exclude all columns except this column and the column(s) by which you want to group your values.
For example, if you want to aggregate **Revenue** per **Country**, you should exclude all columns except **Revenue** and **Country**.
3. Select the projection node in order to display its context tools, and click Σ (*Aggregation*).



An *Aggregation* node is created, its symbol is selected, and its properties are displayed in the side panel.

4. Optional. Rename the node in its side panel to clearly identify it. This name can be changed at any time and can contain only alphanumeric characters and underscores.
5. In the *Columns* section, click the **NONE** on a numeric column and select the appropriate aggregation function:
 - **SUM**: calculate the total value
 - **COUNT**: calculate the number of distinct values
 - **MIN**: calculate the minimum value
 - **MAX**: calculate the maximum value
6. [optional]. Enter an expression in the *Having* section to filter the aggregated data. You can use the following items in your SQL expression:
 - *Insert Values* - Click to open the *Insert Value* dialog, select the values you need to insert in the list, and click *Insert Value*.

Note

- The button is enabled when, in the *Expression* field, a column name is followed by the operator **=**, **>**, **<**, **>=**, **<=**, **!=**, **IN**, **BETWEEN**, or **LIKE**.
- The values listed in the dialog are retrieved from the sources and intermediate nodes of your object. You can insert values of the data types string, integer, boolean, date, and time. The data types binary and UUID aren't supported.
- Once your expression is valid, additional values cannot be inserted. You must edit your expression to make it incomplete; this enables the *Insert Values* button and allows you to insert other values.
- The *Insert Values* dialog doesn't format the expression. Manual formatting may be required after inserting values.

Example

- The expression `Column1 =` enables the *Insert Values* dialog. You can select a single value from the list.
- The expression `Column2 BETWEEN` enables the *Insert Values* dialog. You can select two values from the list. If both values are available in a single search results, both of them can

be selected together. However, if the values need to be selected from two different search results, you have to select and insert one value at once.

- The expression `Column3 IN` enables the *Insert Values* dialog. You can select multiple values from the list. If all the required values are available in a single search results, those values can be selected together. However, if the values have to be selected from different search results, you have to select and insert one value at once. You select and insert the values 2, 5, 6, and 9. The expression `Column3 IN (2, 5, 6, 9)` is valid. You want to add another value to it, so you click *Insert Values*. In the *Insert Values* dialog, you can only see the values 2, 5, 6, and 9 because the dialog is already filtered by the selected values. To see all available values, edit your expression to make it invalid: `Column3 IN (2, 5, 6, 9, .`. The *Insert Values* button is available again and you can add new values.


Select available value(s) and click *Insert* to add them to your expression.

- *Functions* - Browse, select a category, or filter available functions (see [SQL Functions Reference \[page 257\]](#)). Click a function name to see its syntax or click elsewhere in its token to add it to your expression.
- *Columns* - Browse or filter available columns. Click a column name to see its properties or click elsewhere in its token to add it to your expression.
- *Parameters* - Browse or filter available input parameters (see [Create an Input Parameter \[page 231\]](#)). Click a parameter name to see its properties or click elsewhere in its token to add it to your expression.
- *Other* - Browse available operators, predicates, and case expressions, and click one to add it to your expression (see [SQL Reference \[page 255\]](#)).

For example, if you have aggregated your **Revenue** column using `SUM`, and want to show only:

- Total revenues of more than 1m, enter `SUM(Revenue) > 1000000`
- Total revenues for the US only, enter `Country='US'`

When working on a large expression, click  (*Enter Full Screen*) to expand the expression editor.

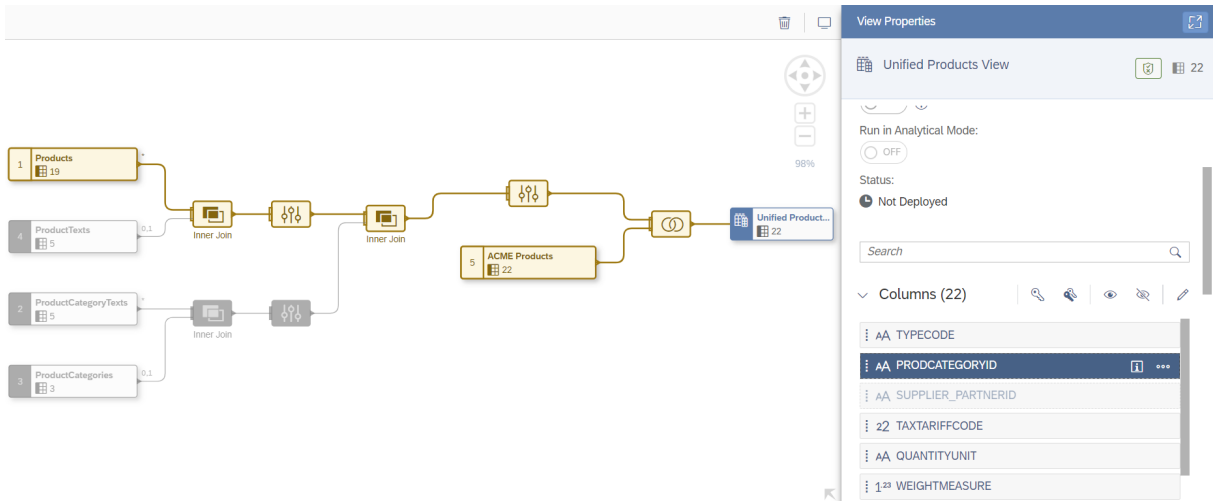
7. Click  (*Preview Data*) to open the *Data Preview* panel and review the data output by this node. For more information, see [Viewing or Previewing Data in Data Builder Objects \[page 297\]](#).

6.1.12 Visualize the Lineage of Columns and Input Parameters in a Graphical View

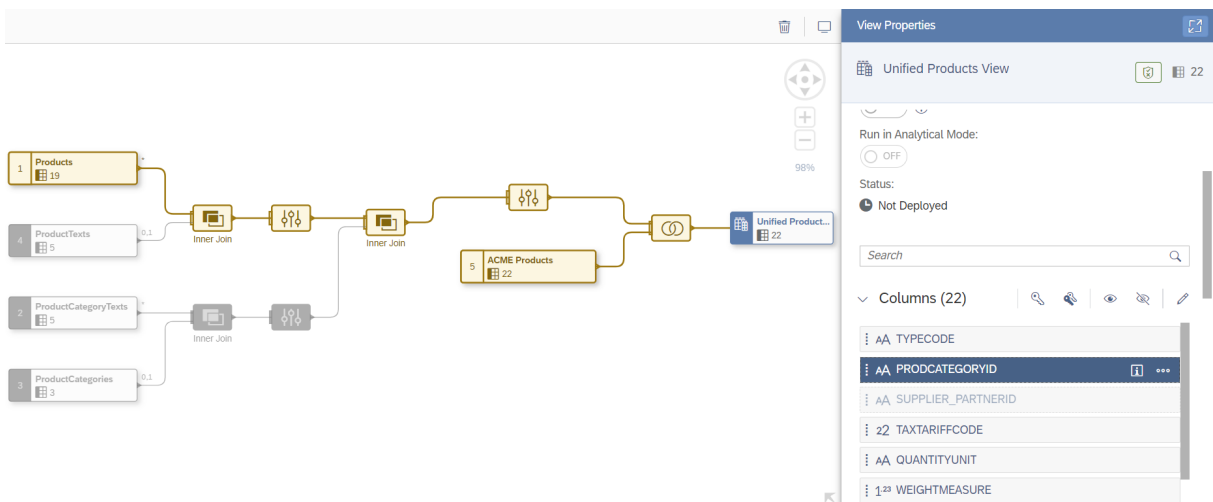
Select one or more columns in any diagram node or an input parameter in the output node to trace their lineage back to their sources.

To remove the highlight, deselect the columns or input parameters or click anywhere in the diagram.

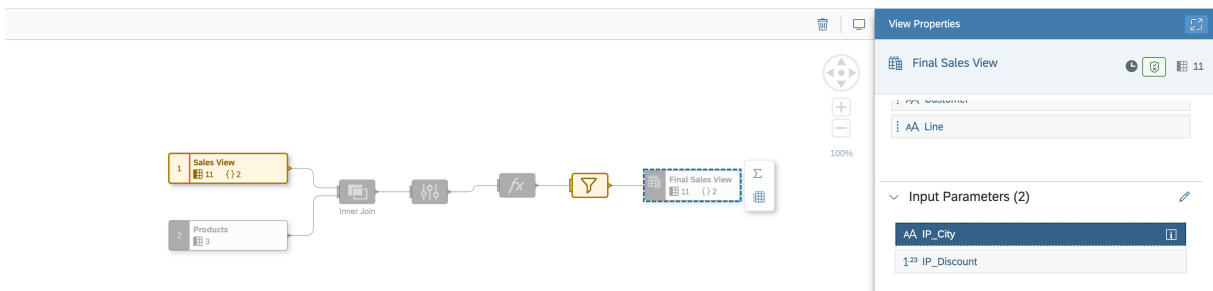
In this example, one column is selected in the output node's side panel and the highlight shows the paths to its two sources:



In this example, two columns are selected in a projection node's side panel and the highlight shows the path to their sources:



In this example, an input parameter is selected in the output node's side panel, and the highlighting shows from which source it originates and that it is used in the filter node:



6.1.13 Persist View Data

Improve the performance while working with views by persisting the view data, and scheduling regular updates to keep your data up-to-date.

By default a view is run every time it is accessed and, if the view is complex or a large amount of data is processed, this may impact the performance of other views or dashboards built on top of it. You can improve performance by persisting the view data and you can schedule regular updates to keep the data fresh.


Note

If your view consumes remote tables, check for additional and potential limitations. See [Integrating Data via Connections](#).

While persisting complex views, see [Persisted Views and Memory Consumption](#).

When opening your view, in the *Properties* panel, under *Data Persistence*, you can see if your view has been made persisted or not.

Note

You can monitor persisted views in the *Data Integration Monitor*, under *Views*. You can access it directly by clicking  (*Views Monitor*). For more information, see [Persisting and Monitoring Views](#).

Information	Description
Data Access	<p>Shows how you currently access your view.</p> <ul style="list-style-type: none">• <i>Persisted</i>: The view is persisted can be used immediately.• <i>Virtual</i>: The view is accessed directly, no intermediate persistency is used. Or the view was persisted and has now been turned into virtual to free up memory space, for example.

Information	Description
<i>Status</i>	<p>Shows the status of the persisted view.</p> <ul style="list-style-type: none"> • <i>Available</i>: The persisted view is available and can be used. • <i>Loading</i>: The persisted view is currently creating or updating. You might need to refresh your screen until the loading is completed to get the final status. Until then the virtual access or the old persisted data is used if the view is accessed. • <i>Error</i>: Something goes wrong during the load of the data to the persisted table. The old persisted data is used or if the view was not successfully loaded before, the data is still accessed via virtual access (status is virtual). You need to fix the error to be able to complete the persisted view creation or update.
<i>Last updated</i>	Shows when the persisted view was last updated.

You can perform actions on data by clicking *Data Persistence*:

- *Start Data Persistence*: Start a new data persistence to update or create the persisted view.

Note

You can set up a view as persisted even if it has been created on top of remote data.

Restriction

A view, which contains input parameters can't be persisted.

- *Remove Persisted Data*: Remove the data that have been persisted in the view and switch the access back to virtual.

Schedule Data Persistence Tasks

Schedule Data Persistence

From this menu, you can :

- *Create Schedule*: Select the relevant persisted view and create a simple or recurring schedule for your view. You define your scheduling options and thus ensure that you always have an up-to-date persisted view.
- *Edit Schedule*: Your scheduling options need to be updated? You can adapt them to your needs at any time from this menu.
- *Delete Schedule*: You don't need to schedule a Data Persistence task anymore? Then you can simply delete it from this menu.
- *Assign Schedule To Me*: Become the owner of the schedule.
- *Pause Schedule*: Pause the scheduled task

- [Resume Schedule](#): Resume the pause scheduled task

Note

You need to authorize SAP to run the recurring scheduled tasks on your behalf. You can do so via the message that is displayed at the top of the monitor, or in your profile settings under [Schedule Consent Settings](#).

For more information, see [Schedule a Data Integration Task \(Simple Schedule\)](#).

Persisted Views and Deployment

When you deploy a persisted view, you need to consider the following cases:

- If you update or redeploy a persisted view, you need the right permission level (Data Integration - Update). For more information, see [Permissions](#). Note that after a redeployment, the data persistence might be deleted and the data access could be changed back to virtual for views with structural changes. This will happen as soon as the underlying database object is recreated as part of the deployment. This can be seen in the [Data Access](#) status in the [View Builder](#) or in the [Data Integration Monitor > Views](#). The update of the data persistence has to be triggered again.
- If the view uses other views that were changed but not yet deployed, they are deployed as well. If these views are persisted, the persistence is deleted as well.
- If the view uses other views that were not changed, these views are not touched by the deployment and therefore the persistence is still available
- If you update or redeploy a view while you are persisting data, the persistence will fail. In this case, try again to persist the view or wait until the next scheduled run.
- If the persisted view is consuming a view for which a data access control has changed (a data access control is added or removed, or its assignment has changed), then the persistence of your parent view is removed when the underlying view is redeployed.

Persisted Views and Data Access Control

While defining [Data Access Control](#) in your view, you need to consider the impact on the persistency. For more information, see [Persisted Views and Data Access Control](#).

6.1.14 Apply a Data Access Control

You can apply one or more data access controls to a view to control the data that users will see based on the specified criteria.

Context

For detailed information about defining and using data access controls, see [Securing Data with Data Access Controls](#).

Procedure

1. In the left navigation area, click [Data Builder](#) and select the space you want to work in.
2. Open the view that you want to apply the data access control to.
3. Select the output node of the view and open the [Data Access Control](#) section in its side panel and click [+](#) (*Add*).
Find available objects by entering the object's name in the search bar or click [▽](#) (*Show filters*) and filter by [Semantic Usage](#) or other criteria.
4. In the [Join](#) section, map columns from your view to the criteria columns defined in the data access control.
You must map a column from your view to each of the criteria columns defined in the data access control in order to correctly filter your view's data by the criteria.
5. Click [↻](#) (*Deploy*) to save and deploy the view.

Results

Users opening your view in SAP Datasphere or in SAP Analytics Cloud or other analytics tools will only see the data permitted by the assignment of their username to the criteria defined in the data access control.

6.1.15 Process Source Changes in the Graphical View Editor

If one or more of the sources of your graphical view is modified, then the next time you open the view, you will be asked to process the changes. If a source change has generated warnings or errors in your view, its status will be updated and you will receive a notification inviting you to review the changes.

Context

Your object's status may be updated when changes to one or more of its sources are saved or deployed:

- If no warnings or errors were generated by the source changes, then the status will not be changed.
- If warnings or errors were generated by the source changes and:
 - Source changes were saved but not deployed, your object's status will be set to *Design-Time Error*.
 - Source changes were saved and deployed, your object's status will be set to *Run-Time Error*.

Procedure

1. If a source change has generated warnings or errors in your view, you will receive a notification inviting you to review them, and you can click the notification to open it directly.

The *Source Updates* dialog opens, listing the sources that have been modified.

Note

If the changes do not generate warnings or errors (for example, new columns are available), you will not receive a notification, but the dialog will still be displayed the next time that you open the view.

2. Click *OK* to dismiss the dialog and open the diagram. The following validation messages may be displayed:

Change in Source	Impact in Dependent Views
Add Column	Columns added to source tables or views are excluded by default from the output of dependent views: <ul style="list-style-type: none">• An information message listing all new columns is displayed on the source.• The new columns are excluded in a projection node directly after the source. If no projection node was previously present, then one is added.
Change Business Name	Changes to the business name and other business properties of a source table or view or its columns are automatically propagated to dependent objects: <ul style="list-style-type: none">• An information message specifying the source object changes or listing all changed columns is displayed on the source.

Change in Source	Impact in Dependent Views
Change Column Data Type	<p>Changes column data types in source tables or views generate warnings in dependent views:</p> <ul style="list-style-type: none"> • An information message listing all columns with changed data types is displayed on the source. • An information message listing all columns with changed data types present in the output node <i>Columns</i> list is displayed on the output node. • If output columns with changed data types are used by any dependent objects of this view, then a warning message listing those views is displayed on the output node.
Delete Column	<p>Deletions of columns in source tables or views generate errors in dependent views if the columns are used in these views:</p> <ul style="list-style-type: none"> • An information message listing all deleted columns is displayed on the source. • An error message is displayed on any intermediate node (join, union, filter, calculated columns, aggregation) in which a deleted column was used. • A warning message listing all deleted columns that were previously present in the output node <i>Columns</i> list is displayed on the output node. • If deleted output columns were used by any dependent objects of this view, then a warning message listing those views is displayed on the output node.
Change to Input Parameter	<p>Changes to input parameters in source views generate messages in dependent views:</p> <ul style="list-style-type: none"> • An information message listing all input parameter changes is displayed on the source. • An error or warning is displayed on the source for each input parameter that is added, deleted, or changed. • An error message is displayed on any intermediate node (join, union, filter, calculated columns, aggregation) in which a deleted input parameter was used.
<div style="background-color: #f0f0f0; padding: 10px;"> <p>Note</p> <p>If an input parameter that has a default value is added to a source view, then this source input parameter is, by default, mapped to the default value in dependent views. You can override this default mapping and map the source input parameter to an input parameter in the dependent view.</p> </div>	
Change to Data Access Control	<p>Changes to a data access control generate warnings in views to which they are attached:</p> <ul style="list-style-type: none"> • A warning message is displayed on the output node to encourage you to review the changes before redeploying the view.

3. Review all information and warning messages to ensure that they do not adversely impact the output of your view.

Note

If you want to include a new source column in your view output, select it in the projection node and click **⋮ (Menu) >> Restore Column**.

4. Review and resolve any error messages on intermediate nodes.
5. Click **Save** to save the changes to your object and dismiss any info and warning messages.

If the source changes were saved but not deployed, your object's status should change from *Design-Time Error* to *Changes to Deploy*.

Note

As the new version of your object depends on the source updates, you should coordinate with the person who has modified the source about deployment. The new version of your object will work correctly in the run-time only when both it and its source are deployed.

6. Click  (*Deploy*) to deploy your object.

Your object's status should change from *Changes to Deploy* (or *Run-Time Error*) to *Deployed*.

6.1.16 Replace a Source

Drag a source from the *Source Browser*, hover over an existing source, and click *Replace*. You are guided through mapping the columns from the old source to the new source.

Context

For example, you may want to replace a local table containing sample data from a CSV file, which is a source in your view, with a remote table containing current data from a source system.

The output structure of the view is preserved as far as possible, and any impacts caused by the replacement of the source are indicated by validation messages.

Procedure

1. If the *Source Browser* panel is not visible on the left of the screen, click *Source Browser* in the toolbar to show it.
2. Select the new source and drag it over the source you want to replace in the diagram. Wait for the context menu to appear, slide your cursor over the *Replace* option and then release the mouse button.

The *Replace Source* dialog opens, showing the old source and its columns on the left side and the new source and its columns on the right side, with mappings proposed, where possible between them.


Note

Only the columns used in the current view are listed on the left side. Any columns that are excluded in your view are not used, and therefore do not need to be remapped.

3. Review the proposed mappings and try to map any unmapped old source columns if possible.


Note

You are not required to map all the columns from the old source, but any you leave unmapped will generate validation warnings or errors. Any new source columns you leave unmapped will be excluded by default.

4. Click [Replace](#) to close the dialog and confirm the replacement. SAP Datasphere performs the following actions:
 - Replace the old source by the new source in your view.
 - If any old source columns were left unmapped, generate warning and error messages for each node that is impacted.
 - If any new source columns were left unmapped, insert a projection node after the source (if one is not already present) and exclude these new columns so that they do not modify the output columns.
 - Update the *Status* of your view.
5. [if new source columns were left unmapped] Review the projection node and restore any new columns that you want to make available (see [Reorder, Rename, and Exclude Columns \[page 223\]](#))
6. [if old source columns were left unmapped] Review each of the validation warnings and errors and correct them as appropriate.
7. Click  ([Save](#)) to save your view and update its *Status*.

Note

If you have not resolved all the validation errors, then you can still save your view, but you will not be allowed to deploy it.

8. If your view is itself used as a source by other views, then review the [Dependent Objects](#) section in the side panel to see if your change has impacted any of them. You can click on an object in the list to navigate to it and review any errors and warnings.
9. When appropriate, click  ([Deploy](#)) to deploy your view.

6.1.17 Edit a Custom CSN Annotation


Edit custom CSN annotations in the [Data Layer](#) editors.

Context

Custom CSN annotations allow you to add semantic metadata that cannot at present be set in the entity properties panel. They can be present in imported .csn files or be added manually as SAP Datasphere doesn't provide an interface for certain modeling possibilities. For detailed information on CSN annotations, see [Core Data Services Schema Notation \(CSN\)](#).

The [Table Editor](#), [Graphical View Editor](#), [SQL Editor](#), and [E/R Modeler](#) allow the editing and validation of custom CSN annotations.

Procedure

1. Click  ([Edit Custom CSN Annotations](#)) to open the *Edit Custom CSN Annotations* dialog:

- Click [◀ \(Previous Annotation\)](#) and [▶ \(Next Annotation\)](#) to navigate from one custom CSN annotation to another.
 - Enter changes for the custom CSN annotations in the code.
 - Click [🛡️ \(Validate CSN expression\)](#) to check if changes invalidate or not the CSN expression.
2. Click [OK](#) to save the changes or [Cancel](#) to go back to the editor without saving.

6.2 Creating an SQL View

Create a view to query sources in a powerful SQL editor. You can choose between writing a standard SQL query using `SELECT` statements and operators such as `JOIN` and `UNION`, or use SQLScript to produce a table function. You can drag sources from the [Source Browser](#), and specify measures and other aspects of your output structure in the side panel.

Context

If you are not comfortable with SQL, you can still build a view in SAP Datasphere by using the Graphical View editor, which lets you compose SQL code using an intuitive graphical interface (see [Creating a Graphical View \[page 213\]](#)).

📘 Note

There are two methods for exposing view data for consumption outside SAP Datasphere:

- SAP Analytics Cloud (and Microsoft Excel via an SAP add-in) do not consume view data directly. Set the *Semantic Usage* of your view to *Fact* and then add it to an analytic model to expose it (see [Creating an Analytic Model \[page 337\]](#)). There is no need to enable the *Expose for Consumption* switch.
- Other third-party BI clients, tools, and apps can consume data from views with any *Semantic Usage* via OData or ODBC if the *Expose for Consumption* switch is enabled.

For more information, see [Consuming Data Exposed by SAP Datasphere](#).

Procedure

1. In the side navigation area, click [🔗 \(Data Builder\)](#), select a space if necessary, and click [New SQL View](#) to open the editor.
2. In the side panel, select the language that you want to use. You can choose between:
 - [SQL \(Standard Query\)](#) - [default] Create a standard SQL query, based around `SELECT` statements (see [SQL Reference \[page 255\]](#)).
 - [SQLScript \(Table Function\)](#) - Use SQLScript with support for `IF` statements, loops, and other more complex structures (see [SQLScript Reference \[page 264\]](#)).

3. Enter your code in the editor panel. You can:

- Access auto-complete suggestions for keywords and object names including source entities by typing.
- Drag sources from the [Source Browser](#) into the editor panel.

Note

If you add a source view containing input parameters, the source and its parameters will be added to your code and you will need to process each parameter (see [Process Source Input Parameters in an SQL View \[page 254\]](#))

- Add comments to document your code:
 - Comment out a single line or the rest of a line with a double dash: `-- Your comment here.`
 - Comment out multiple lines or part of a line with: `/* Your comment here */.`


This example contains various forms of comments:

```
/*
   This is a multi-line comment to
   introduce this view
*/
SELECT "ID",
       "Date",
--    "Sales Person", comment out a line
       "City", -- comment at end of line
       "Net Sales"
FROM /* mid-line comment */ "Sales"
```

- Format your SQL code by clicking [Format](#).






Note


SQLScript cannot be formatted.

- Validate your code and update the display of your output structure in the side panel at any time by clicking  ([Validate SQL and Preview Data](#)). SQL errors and warnings are shown in the [Errors](#) pane at the bottom of the editor. Problems with the output structure are shown on the [Validation Messages](#) button in the side panel header.

Note

If your SQLScript is complicated, SAP Datasphere may not be able to determine the output structure. In this case, you are requested to review the list of columns. Click the [Edit](#) button to add or delete buttons or change column names and data types.





- Preview the data being output by the view by clicking:  ([Show or hide data preview](#)) (see [Viewing or Previewing Data in Data Builder Objects \[page 297\]](#)). Previewing a SQL view isn't possible if one of the view's objects is shared from another space and has an input parameter.
- Click  ([Edit Custom CSN Annotations](#)) to open the [Edit Custom CSN Annotations](#) dialog:
 - Click  ([Previous Annotation](#)) and  ([Next Annotation](#)) to navigate from one custom CSN annotation to another.
 - Enter changes for the custom CSN annotations in the code.
 - Click  ([Validate CSN expression](#)) to check if changes invalidate or not the CSN expression.

- View the objects that depend on an analyzed object (its impacts) and the objects on which the analyzed object depends (its lineage) by clicking  (*Impact and Lineage Analysis*) (see [Impact and Lineage Analysis \[page 34\]](#)).
4. Enter the following properties as appropriate in the side panel:

Property	Description
Business Name	Enter a descriptive name to help users identify the object. This name can be changed at any time.
Technical Name	<p>Displays the name used in scripts and code, synchronized by default with the <i>Business Name</i>.</p> <p>To override the default technical name, enter a new one in the field. Technical names can contain only alphanumeric characters and underscores.</p> <div style="border: 1px solid #ccc; background-color: #f9f9f9; padding: 10px; margin-top: 10px;"> <p>Note</p> <p>Once the object is saved, the technical name can no longer be modified.</p> </div>
Package	<p>Select the package to which the object belongs.</p> <p>Packages are used to group related objects in order to facilitate their transport between tenants.</p> <div style="border: 1px solid #ccc; background-color: #f9f9f9; padding: 10px; margin-top: 10px;"> <p>Note</p> <p>Once a package is selected, it cannot be changed here. Only a user with the DW Space Administrator role (or equivalent privileges) can modify a package assignment in the <i>Packages</i> editor.</p> </div> <p>For more information, see Creating Packages to Export.</p>
Language	<p>Select the SQL language to use.</p> <p>Choose from the following:</p> <ul style="list-style-type: none"> • <i>SQL (Standard Query)</i> - [default] Create a standard SQL query, based around SELECT statements (see SQL Reference [page 255]). • <i>SQLScript (Table Function)</i> - Use SQLScript with support for IF statements, loops, and other more complex structures (see SQLScript Reference [page 264]).
Source Object (Open SQL Schema/HDI Container)	[read-only] Displays the technical name of the Open SQL Schema or HDI Container and the object name.

Property	Description
Semantic Usage	<p>Select the way your entity should be used for data modeling purposes.</p> <p>Choose from the following:</p> <ul style="list-style-type: none"> • <i>Fact</i> - Contains one or more measures and attributes. A fact typically has associations pointing to one or more dimensions and is consumed by analytic models (see Creating a Fact [page 305]). • <i>Dimension</i> - Contains attributes containing master data like a product list or store directory, and supporting hierarchies (see Creating a Dimension [page 314]). • <i>Hierarchy</i> - Contains attributes defining a parent-child hierarchy (see Creating an External Hierarchy [page 330]). • <i>Hierarchy with Directory</i> - Contains one or more parent-child hierarchies (see Creating a Hierarchy with Directory [page 331]). • <i>Text</i> - Contains attributes used to provide textual content in one or more languages (see Create a Text Entity for Attribute Translation [page 327]). • <i>Relational Dataset</i> - [default] Contains columns with no specific analytical purpose. • <i>Analytical Dataset (Deprecated)</i> - Use <i>Fact</i> instead (see Analytical Datasets (Deprecated) [page 351]).
Expose for Consumption	<p>Enable this option to make the view available for consumption outside SAP Datasphere via OData or ODBC.</p> <div style="border: 1px solid #ccc; padding: 10px; margin-top: 10px;"> <p>Note</p> <p>There are two methods for exposing view data for consumption outside SAP Datasphere:</p> <ul style="list-style-type: none"> • SAP Analytics Cloud (and Microsoft Excel via an SAP add-in) do not consume view data directly. Set the <i>Semantic Usage</i> of your view to <i>Fact</i> and then add it to an analytic model to expose it (see Creating an Analytic Model [page 337]). There is no need to enable the <i>Expose for Consumption</i> switch. • Other third-party BI clients, tools, and apps can consume data from views with any <i>Semantic Usage</i> via OData or ODBC if the <i>Expose for Consumption</i> switch is enabled. <p>For more information, see Consuming Data Exposed by SAP Datasphere.</p> </div>
Run in Analytical Mode	<p>Enable this option to send the USE_OLAP_PLAN hint to the SQL optimizer.</p> <p>This may improve view performance, particularly if a union is performed. It is only available if <i>Expose for Consumption</i> is enabled.</p> <p>For more information, see HINT Details in the <i>SAP HANA Cloud, SAP HANA Database SQL Reference Guide</i>.</p>
Status	<p>[read-only] Displays the deployment and error status of the object.</p> <p>For more information, see Saving and Deploying Objects [page 39].</p>

- Based on the *Semantic Usage* of your entity, review and modify its *Columns*, *Attributes*, and/or *Measures*:
 - *Fact* - Review the lists of measures and attributes (see [Creating a Fact \[page 305\]](#)).
 - *Dimension* - Review the list of attributes (see [Creating a Dimension \[page 314\]](#)).

- *Hierarchy* - Define the parent and child columns (see [Creating an External Hierarchy \[page 330\]](#)).
 - *Hierarchy with Directory* - Define all the necessary attributes and settings (see [Creating a Hierarchy with Directory \[page 331\]](#)).
 - *Text* - Review the list of attributes (see [Create a Text Entity for Attribute Translation \[page 327\]](#)).
 - *Relational Dataset* - Review the list of columns (see [Columns \[page 108\]](#)).
6. Complete or consult other sections as appropriate:
- *Input Parameters* - Create input parameters to require the user to enter a value for use in calculated column, filter, and aggregation nodes (see [Create an Input Parameter \[page 231\]](#)).
 - *Data Persistence* - Persist the view data to improve performance (see [Persist View Data \[page 242\]](#)).
 - *Associations* - Create associations to other entities (see [Create an Association \[page 234\]](#)).
 - *Data Access Controls* - Add data access controls to apply row-based security and control access to individual rows based on various criteria (see [Securing Data with Data Access Controls](#)).
 - *Business Purpose* - Provide a description, purpose, contacts, and tags to help other users understand your entity.
 - *Dependent Objects* - If your entity is used as a source or association target for other entities, then they are listed here (see [Review the Objects That Depend on Your Table or View \[page 43\]](#)).
7. Click  (*Save*)  *Save*  to save your entity or click  (*Deploy*) to save and deploy it immediately. For more information, see [Saving and Deploying Objects \[page 39\]](#).

6.2.1 Process Source Input Parameters in an SQL View

Process input parameters contained in source views by mapping them to new input parameters or entering a value to resolve them.

Context

Graphical and SQL views can contain input parameters, which prompt the user to enter a value for use in filtering or other calculations when the view is run (see [Create an Input Parameter \[page 231\]](#)).

When adding such a view as a source in your SQL view, you must process each of the input parameters.

Procedure

1. Add a view to your SQL view by typing its name or by dragging it from the *Source Browser* into the editor panel.

If the view contains input parameters it will be added to your code in the following syntax, which differs slightly between SQL and SQLScript:

SQL View	SQLScript View
<code>"<SourceName>" (<IPName>: :PARAM_1)</code>	<code>"<SourceName>" (<IPName>=>:PARAM_1)</code>

2. Choose how you want to process each input parameter. You can:

- Map the source input parameter to an input parameter in the view. You must first create the input parameter in the side panel (see [Create an Input Parameter \[page 231\]](#)) and then enter its name in the code, prefixed by a colon. The new input parameters can have the same name as the source input parameters, but this is not required. In this example the `sales` view is added as a source and its input parameters, `IP_CITY` and `IP_DISCOUNT`, are mapped to new input parameters with the same names:

SQL View	SQLScript View
<code>"Sales" (IP_CITY: :IP_CITY, IP_DISCOUNT: :IP_DISCOUNT)</code>	<code>"Sales" (IP_CITY=>:IP_CITY, IP_DISCOUNT=>:IP_DISCOUNT)</code>

- Enter a value to resolve the input parameter. In this example the `IP_CITY` and `IP_DISCOUNT` input parameters are set to the values `Tokyo` and `10` respectively:

SQL View	SQLScript View
<code>"Sales" (IP_CITY: 'Tokyo', IP_DISCOUNT: 10)</code>	<code>"Sales" (IP_CITY=>'Tokyo', IP_DISCOUNT=>10)</code>

3. Click [Validate SQL](#) to validate the completed syntax.

6.2.2 SQL Reference

SAP Datasphere views support a subset of the SQL syntax supported by SAP HANA Cloud.

SAP Datasphere primarily supports the SQL syntax listed in the [SELECT Statement \(Data Manipulation\)](#) and related sections of the *SAP HANA Cloud, SAP HANA Database SQL Reference Guide*.

SAP Datasphere supports the following:

- Operators (see [SQL Operators \[page 256\]](#))
- Predicates (see [SQL Predicates \[page 256\]](#))
- Expressions (see [Expressions \[page 257\]](#))
- Functions (see [SQL Functions Reference \[page 257\]](#))

→ Tip

An alternative to using `TOP`, which is not supported in SAP Datasphere is to use `limit`.

So, instead of using:

```
SELECT TOP 3 "SMALL_INT" FROM "ALLDATASV0" ORDER BY "SMALL_INT" DESC
```

You can use:

```
SELECT "SMALL_INT" FROM "ALLDATASV0" ORDER BY "SMALL_INT" DESC limit 3
```

SQL Operators

SAP Datasphere supports the following operators:

Operator	Description
+	Addition
-	Subtraction
*	Multiplication
/	Division
AND	Logical conjunction
OR	Logical disjunction
NOT	Logical negation
	Concatenation
<	Less than
>	Greater than
=	Equal
!=	Not equal
<=	Less than or equal
>=	Greater than or equal
LIKE	Value similar to
IS NULL	Value does not exist
BETWEEN	Value between two other values
()	Parentheses

For more information, see [Operators](#) in the *SAP HANA Cloud, SAP HANA Database SQL Reference Guide*.

SQL Predicates

SAP Datasphere supports the following predicates:

Predicate	Description
ANY, SOME, ALL	Compares values using the specified comparison operator and returns true, false, or unknown.
BETWEEN	Compares a value with a list of values within the specified range and returns true or false.
CONTAINS	Matches a search string with the results of a subquery.
EXISTS	Tests for the presence of a value in a set and returns either true or false.
IN	Searches for a value in a set of values and returns true or false.
LIKE	Performs a comparison to see if a character string matches a specified pattern.
MEMBER OF	Determines whether a value is a member of an array.
NULL	Performs a comparison of the value of an expression with NULL.

For more information, see [Predicates](#) in the *SAP HANA Cloud, SAP HANA Database SQL Reference Guide*.

Expressions

SAP Datasphere supports the `CASE` expression:

Expression	Description
<code>CASE WHEN THEN ELSE END</code>	Provides if, then, else logic.

For more information, see [Expressions](#) in the *SAP HANA Cloud, SAP HANA Database SQL Reference Guide*.

6.2.3 SQL Functions Reference

SAP Datasphere views support a subset of the SQL functions supported by SAP HANA Cloud.

This topic contains the following sections:

- [Array Functions \[page 258\]](#)
- [Data Type Conversion Functions \[page 258\]](#)
- [Datetime Functions \[page 259\]](#)
- [Fulltext Functions \[page 260\]](#)
- [Miscellaneous Functions \[page 260\]](#)
- [Numeric Functions \[page 260\]](#)
- [Security Functions \[page 261\]](#)
- [String Functions \[page 262\]](#)

- [Window Functions \[page 263\]](#)
- [Example: Converting Currency Values with CONVERT_CURRENCY \[page 264\]](#)

Array Functions

SAP Datasphere supports the following array functions, which take arrays as input:

- SUBARRAY
- TRIM_ARRAY

For detailed documentation of these functions, see [Array Functions](#) in the *SAP HANA Cloud, SAP HANA Database SQL Reference Guide*.

Data Type Conversion Functions

SAP Datasphere supports the following data type conversion functions, which convert data from one data type to another:

- CAST
- TO_BIGINT
- TO_BINARY
- TO_BLOB
- TO_BOOLEAN
- TO_CLOB
- TO_DATE
- TO_DATS
- TO_DECIMAL (DECFLOAT)
- TO_DOUBLE (DOUBLE)
- TO_FIXEDCHAR
- TO_INT (INT)
- TO_INTEGER (INT)
- TO_NCLOB
- TO_NVARCHAR
- TO_REAL (FLOAT)
- TO_SECONDSDATE
- TO_SMALLDECIMAL
- TO_SMALLINT
- TO_TIME
- TO_TIMESTAMP
- TO_TINYINT
- TO_VARBINARY
- TO_VARCHAR

For detailed documentation of these functions, see [Data Type Conversion Functions](#) in the *SAP HANA Cloud, SAP HANA Database SQL Reference Guide*.

Datetime Functions

SAP Datasphere supports the following datetime functions, which perform operations on date and time data types or return date or time information:

- `ADD_DAYS`
- `ADD_MONTHS`
- `ADD_MONTHS_LAST`
- `ADD_NANO100`
- `ADD_SECONDS`
- `ADD_WORKDAYS`
- `ADD_YEARS`
- `CURRENT_DATE`
- `CURRENT_DATE ()`
- `CURRENT_TIME`
- `CURRENT_TIME ()`
- `CURRENT_TIMESTAMP`
- `CURRENT_TIMESTAMP ()`
- `DAYNAME`
- `DAYOFMONTH`
- `DAYOFYEAR`
- `DAYS_BETWEEN`
- `HOUR`
- `ISOWEEK`
- `LAST_DAY`
- `LOCALTOUTC`
- `MINUTE`
- `MONTH`
- `MONTHNAME`
- `MONTHS_BETWEEN`
- `NEXT_DAY`
- `NOW`
- `QUARTER`
- `SECOND`
- `SECONDS_BETWEEN`
- `UTCTOLOCAL`
- `WEEK`
- `WEEKDAY`
- `WORKDAYS_BETWEEN`

- YEAR
- YEARS_BETWEEN

For detailed documentation of these functions, see [Datetime Functions](#) in the *SAP HANA Cloud, SAP HANA Database SQL Reference Guide*.

Fulltext Functions

SAP Datasphere supports the following fulltext functions, which perform operations on data that has a fulltext index:

- SCORE

For detailed documentation of this function, see [SCORE Function](#) in the *SAP HANA Cloud, SAP HANA Database SQL Reference Guide*.

Miscellaneous Functions

SAP Datasphere supports the following miscellaneous functions, which return system values and perform various operations on values, expressions, and return values of other functions:

- COALESCE
- CONVERT_CURRENCY - See [Example: Converting Currency Values with CONVERT_CURRENCY \[page 264\]](#)
- CONVERT_UNIT
- GREATEST (MAX)
- HASH_SHA256
- IFNULL
- LEAST (MIN)
- SESSION_CONTEXT
- SESSION_USER
- SYSUID
- WIDTH_BUCKET
- XMLEXTRACT
- XMLEXTRACTVALUE

For detailed documentation of these functions, see [Miscellaneous Functions](#) in the *SAP HANA Cloud, SAP HANA Database SQL Reference Guide*.

Numeric Functions

SAP Datasphere supports the following numeric functions, which perform mathematical operations on numerical data types or return numeric information:

- ABS
- ACOS
- ASIN
- ATAN
- ATAN2
- BITAND
- BITCOUNT
- BITNOT
- BITOR
- BITXOR
- CEIL
- COS
- COSH
- COT
- EXP
- FLOOR
- LN (LOG)
- LOG (LOG10)
- MOD (%)
- NDIV0
- POWER (**)
- RAND
- ROUND
- SIGN
- SIN
- SINH
- SQRT
- TAN
- TANH
- UMINUS

For detailed documentation of these functions, see [Numeric Functions](#) in the *SAP HANA Cloud, SAP HANA Database SQL Reference Guide*.

Security Functions

SAP Datasphere supports the following security functions, which provide special functionality for security purposes:

- ESCAPE_DOUBLE_QUOTES
- ESCAPE_SINGLE_QUOTES
- IS_SQL_INJECTION_SAFE

For detailed documentation of these functions, see [Security Functions](#) in the *SAP HANA Cloud, SAP HANA Database SQL Reference Guide*.

String Functions

SAP Datasphere supports the following string functions, which perform extraction and manipulation on strings, or return information about strings:

- ABAP_ALPHANUM
- ABAP_LOWER
- ABAP_NUMC
- ABAP_UPPER
- ASCII
- BINTOHEX
- BINTOSTR
- CHAR
- CONCAT
- HEXTOBIN
- LCASE
- LEFT
- LENGTH
- LOCATE
- LOWER
- LPAD
- LTRIM
- NCHAR
- REPLACE
- REPLACE_REGEXPR
- RIGHT
- RPAD
- RTRIM
- SOUNDEX
- STRTOBIN
- SUBSTR_AFTER
- SUBSTR_BEFORE
- SUBSTRING
- UCASE
- UNICODE
- UPPER

For detailed documentation, see [String Functions](#) in the *SAP HANA Cloud, SAP HANA Database SQL Reference Guide*.

Window Functions

SAP Datasphere supports the following window and window aggregation functions, which allow you to perform analytic operations over a set of input rows:

- AVG
- BINNING
- CORR
- CORR_SPEARMAN
- COUNT
- CUBIC_SPLINE_APPROX
- CUME_DIST
- DENSE_RANK
- FIRST_VALUE
- LAG
- LAST_VALUE
- LEAD
- MAX
- MEDIAN
- MIN
- NTH_VALUE
- NTILE
- PERCENT_RANK
- RANDOM_PARTITION
- RANK
- ROW_NUMBER
- SERIES_FILTER
- STDDEV
- SUM
- VAR
- WEIGHTED_AVG

Note

In the graphical view editor, window functions can only be used in a *Calculated Columns* node (and not in *Filter* or *Aggregation* nodes).

The following types of functions and syntaxes are not supported in SAP Datasphere views:

- Inverse distribution functions
- COLLATE
- SERIES
- WITHIN GROUP

For detailed documentation of these functions, see [Window Functions and the Window Specification](#) in the *SAP HANA Cloud, SAP HANA Database SQL Reference Guide*.

Example: Converting Currency Values with `CONVERT_CURRENCY`

When using the `CONVERT_CURRENCY` function you must, as a minimum, complete the following parameters:

- `AMOUNT` - Specify the column containing the currency value to be converted
- `CLIENT` - Specify the three character string identifying the tenant in your SAP system
- `SOURCE_UNIT` - Specify the column containing the source currency or enter the source currency code
- `TARGET_UNIT` - Specify the column containing the target currency or enter the target currency code
- `REFERENCE_DATE` - Specify the date to use when obtaining the conversion rate
- `SCHEMA` - Specify the space name.

In our example, which is in a space named `SALES`, we are converting the value in the `Sale_Amount` column from Euros to US Dollars using the conversion rate from 30 September 2021:

```
CONVERT_CURRENCY(  
    "AMOUNT" => "Sale_Amount",  
    "SOURCE_UNIT" => 'EUR',  
    "TARGET_UNIT" => 'USD',  
    "SCHEMA" => 'SALES',  
    "REFERENCE_DATE" => '2021-09-30',  
    "CLIENT" => '000'  
)
```

For full documentation of this function, see [CONVERT_CURRENCY](#) in the *SAP HANA Cloud, SAP HANA Database SQL Reference Guide*.

6.2.4 SQLScript Reference

When you set the SAP Datasphere SQL view editor *Language* property to *SQLScript (Table Function)*, it creates an SAP HANA Cloud table user-defined function.

SAP Datasphere supports the SQLScript syntax documented for table user-defined functions in [User-Defined Functions](#) and associated sections of the [SAP HANA Cloud, SAP HANA SQLScript Reference](#).

6.2.5 Process Source Changes in the SQL View Editor

If one or more of the sources of your SQL view is modified, and the changes have generated warnings or errors in your view, its status will be updated and you will receive a notification inviting you to review them.

Context

Your object's status may be updated when changes to one or more of its sources are saved or deployed:

- If no warnings or errors were generated by the source changes, then the status will not be changed.
- If warnings or errors were generated by the source changes and:


- Source changes were saved but not deployed, your object's status will be set to *Design-Time Error*.
- Source changes were saved and deployed, your object's status will be set to *Run-Time Error*.

Procedure

1. If a source change has generated warnings or errors in your view, you will receive a notification inviting you to review them, and you can click the notification to open it directly.
2. Click the *Validate and Preview SQL Data* tool and review and correct any errors that appear in the *Errors* panel.

Note

Changes to source view input parameters are not shown in validation messages and must be processed manually.

3. If a data access control that is attached to your view has changed, a warning message is displayed on the side panel to encourage you to review the changes before redeploying the view.
4. Click  (*Save*) to save the changes to your object and dismiss any info and warning messages.

If the source changes were saved but not deployed, your object's status should change from *Design-Time Error* to *Changes to Deploy*.

Note

As the new version of your object depends on the source updates, you should coordinate with the person who has modified the source about deployment. The new version of your object will work correctly in the run-time only when both it and its source are deployed.

5. Click  (*Deploy*) to deploy your object.

Your object's status should change from *Changes to Deploy* (or *Run-Time Error*) to *Deployed*.

6.3 Creating an Intelligent Lookup

Create an intelligent lookup to merge data from two entities even if there are problems joining them. Intelligent lookup offers a business-centric, interactive data harmonization environment for subject matter experts.

Context

Note

The following privileges are required to work with intelligent lookups:

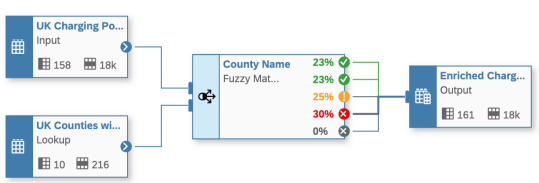
Action	Requires Privileges	Contained in Standard Roles
Create, edit, deploy, delete intelligent lookups	<i>Data Warehouse Data Builder</i> (CRUD----)	<i>DW Modeler</i>
Run intelligent lookups and process results	<i>Data Warehouse Data Integration</i> (--U-----)	<i>DW Integrator</i>

For more information about the roles and privileges needed to work with editors, see [Roles and Privileges by App and Feature](#).

When combining data, there may be no column in your primary entity that contains data to uniquely identify a record in the other entity, and which would thus allow the creation of a standard join. Or, if such a column (a *foreign key*) exists, its data may be incomplete or unreliable. This can be particularly common when one of the entities comes from outside your organization. It may require a lot of manual work in spreadsheets to join the entities, and the results of this work may not be easy to reuse when new data arrives.

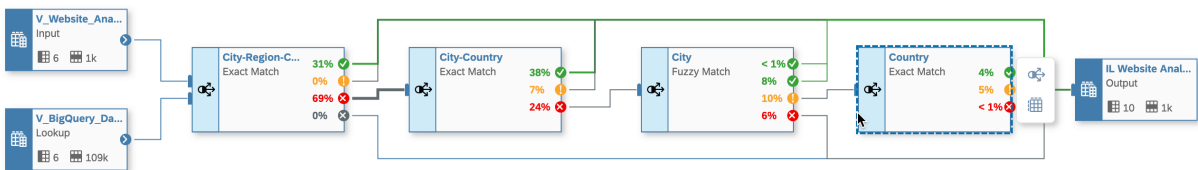
An intelligent lookup lets you iteratively join entities by defining rules to match records and then reviewing and processing the results.

In this example, we use fuzzy matching to enrich a view containing 19,000 UK electric vehicle charging points with accurate UK county names:



For more information, see [Example: Harmonizing County Data for UK Charging Sites \[page 283\]](#).

In this example, we use a series of rules to iteratively join our website analytics view with a view providing latitude and longitude data for more than 100,000 cities around the world:



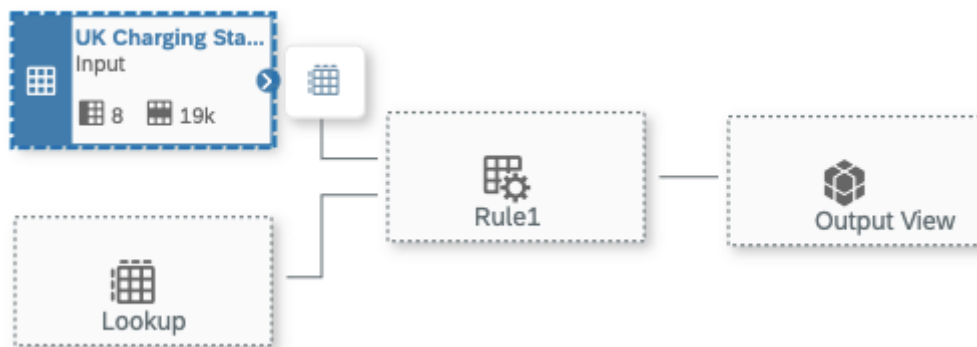
For more information, see [Example: Adding Latitude and Longitude Data with a Multi-Rule Intelligent Lookup \[page 284\]](#).

See the blog [One-Stop-Shop to Intelligent Lookup in SAP Datasphere](#) (published in December 2021) for links to blogs and videos providing further information about intelligent lookup.

Procedure

1. In the side navigation area, click (*Data Builder*), select a space if necessary, and click *New Intelligent Lookup* to open the editor.

2. Select your input entity:
 - a. Drag the table or view that you want to enrich from the *Source Browser* panel and drop it on the *Input* placeholder (see [Prepare Input and Lookup Entities \[page 270\]](#)).



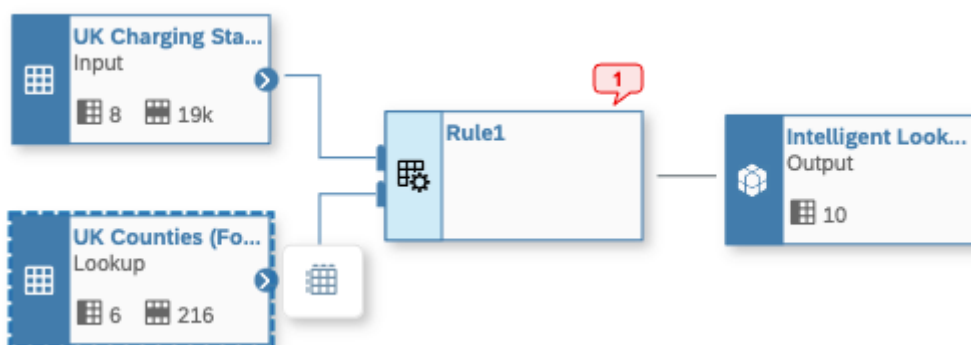
- b. In the *Input* node side panel, identify your pairing column by dragging it from the *Columns* section, and dropping it in the *Pairing Column* section.

You should choose:

- A column intended to identify individual objects of the type stored in the lookup table. These columns often end in *ID* (such as a *Product ID*, *Customer ID*, or *Building ID*) or are some other kind of unique identifier, like *Email Address*.
- If such a column does not exist, you may choose to create a calculated column and concatenate values from other columns into it to provide unique identifier values.
- If the column does not exist and cannot be easily created, or if the objects in the input and lookup entities are of the same type, select the key column.

3. Select your lookup entity (see [Prepare Input and Lookup Entities \[page 270\]](#)):

- a. Drag the table or view containing the columns you want to use to enrich your input entity from the *Source Browser* panel and drop it on the *Lookup* placeholder.



- b. In the *Lookup* node side panel, identify the columns you want to add to your input entity by dragging them from the *Columns* section and dropping them in the *Return Columns* section.
4. Click the *Rule* node and choose a strategy to compare values in one or more columns of your input and lookup entities:
 - *Exact Match* - Require exact matches between values in input entity columns and lookup entity columns (see [Create an Exact Match Rule \[page 271\]](#)).
 - *Fuzzy Match* - Search across one or more string columns and set threshold scores for determining matches (see [Create a Fuzzy Match Rule \[page 274\]](#)).

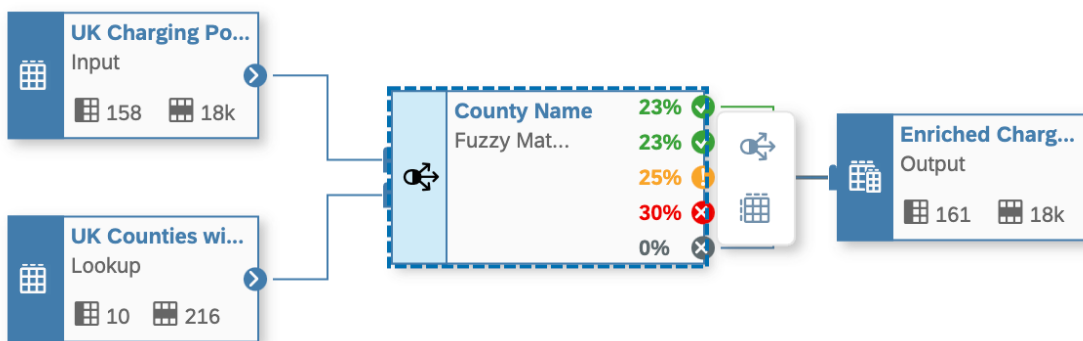
- Click the *Output* node and configure the view that will be output by your intelligent lookup.

You can set the name of the view, choose to include unmatched records in it, review its columns, and set default values to insert into the return columns of unmatched records (see [Configure the View Output by an Intelligent Lookup \[page 277\]](#)).

- Click the diagram background and enter the following properties of your intelligent lookup as necessary:

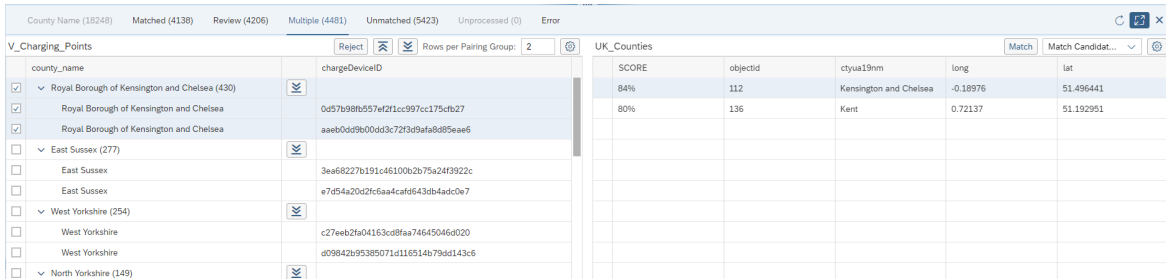
Property	Description
Business Name	Enter a descriptive name to help users identify the object. This name can be changed at any time.
Technical Name	<p>Displays the name used in scripts and code, synchronized by default with the <i>Business Name</i>.</p> <p>To override the default technical name, enter a new one in the field. Technical names can contain only alphanumeric characters and underscores.</p> <div style="border: 1px solid #ccc; padding: 5px; margin-top: 10px;"> <p>Note</p> <p>Once the object is saved, the technical name can no longer be modified.</p> </div>
Package	<p>Select the package to which the object belongs.</p> <p>Packages are used to group related objects in order to facilitate their transport between tenants.</p> <div style="border: 1px solid #ccc; padding: 5px; margin-top: 10px;"> <p>Note</p> <p>Once a package is selected, it cannot be changed here. Only a user with the DW Space Administrator role (or equivalent privileges) can modify a package assignment in the <i>Packages</i> editor.</p> </div> <p>For more information, see Creating Packages to Export.</p>
Status	<p>[read-only] Displays the deployment and error status of the object.</p> <p>For more information, see Saving and Deploying Objects [page 39].</p>

- Deploy and run the intelligent lookup:
 - Click (*Deploy*), enter business and technical names in the *Save Intelligent Lookup* dialog, then click *Save*.
 - Once deployment is complete, click (*Run*) to run the intelligent lookup and review the results.



Once the run is complete, the color-coded results of the rule are displayed on its symbol as percentages of the input records.

8. Select the rule node and then click  (*Preview Data*) to open the *Preview Data* panel and process the results in the various tabs:



county_name	chargeDeviceID
Royal Borough of Kensington and Chelsea (430)	
Royal Borough of Kensington and Chelsea	0d57b98b557ef2f21cc99cc175cfb27
Royal Borough of Kensington and Chelsea	aae0cd5b00dd3c72f3d9afa8d85eae6
East Sussex (277)	
East Sussex	3ea68227b191c46100b2b75a24f3922c
East Sussex	e7d54a20d2fc6aa4caf643db4adc0e7
West Yorkshire (254)	
West Yorkshire	c27eeb2fa04163cd8faa74645046d020
West Yorkshire	d09842b95385071d116514b79dd143c6
North Yorkshire (149)	

SCORE	objectId	ctyua19nm	long	lat
84%	112	Kensington and Chelsea	-0.18976	51.496441
80%	136	Kent	0.72137	51.192951

- **Matched** (Green) - Matched results are input records that are matched with a lookup record. You can reject any matches that you do not agree with. See [Process Matched Results \[page 278\]](#).
- **Review** (Green) - [fuzzy rules only] Review results are input records that are matched with a lookup record with a score between the **Review** and **Matched** thresholds. You can approve or reject proposed matches. See [Process Review Results \[page 279\]](#).
- **Multiple** (Yellow) - Multiple results are input records that are matched with two or more lookup records. You can choose among the match candidates or apply a new rule to them. See [Process Multiple Match Results \[page 280\]](#).
- **Unmatched** (Red) - Unmatched results are input records that are not matched with any lookup record. You can try to manually match them or apply a new rule to them. See [Process Unmatched Results \[page 282\]](#).
- **Unprocessed** (Grey) - Unprocessed records were not present the last time that the rule was run. You can re-run your intelligent lookup to process these records.

Note

Only records that have new pairing column values are included here. New records that have the same pairing column value as an existing pairing group are automatically placed in the appropriate results category for that pairing group.

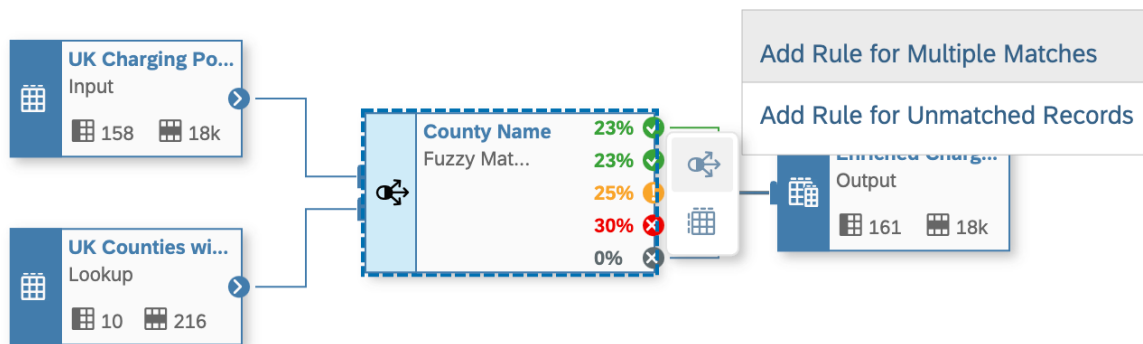
9. If necessary, modify your rule or add further rules to refine your output.

Note

You are not required to match all records. You can stop at any time when the view produced by the intelligent lookup is satisfactory for your needs.

To add a further rule, select a rule node, click the [Add Rule](#) button and then click:

- **Add Rule for Multiple Matches** - To apply a new rule to the records in the **Multiple** category of the present rule. The two rules are combined using an **AND** logic with the new rule applied to the match candidates produced by the present rule.
- **Add Rule for Unmatched Records** - To apply a new rule to the records in the **Unmatched** category of the present rule.



Note

If you modify a rule after having run the intelligent lookup, you are prompted to delete all the results from it and any subsequent rules. You can choose to keep or delete user-confirmed matches (that you have confirmed on the *Review* tab or manually matched on the *Multiple* or *Unmatched* tabs).

You can choose to delete all matches at any time by clicking the *Delete All Matches* button on the main toolbar.

If you modify a rule or add a new rule, you must redeploy the intelligent lookup before you can rerun it.

6.3.1 Prepare Input and Lookup Entities

Prepare your input entity, which you want to enrich, and your lookup entity, from which you want to retrieve columns to add to your intelligent lookup.

An intelligent lookup can join any two entities that have some kind of semantic relationship, even if the input entity is not able consistently to identify records in the lookup entity. It can be particularly helpful for harmonizing heterogeneous data (including data from external sources) in the following common situations:

- Combine internal product data with retailer sales transactions
- Combine OEM warranty claims with supplier data
- Combine marketing data with official government data
- Combine internal customer revenue data with externally-available data such stock price, turnover, market capitalization, number of employees

Identify and prepare your input and lookup entities as follows:

Your Input Entity

- Is a table or view that you want to enrich by joining it to a lookup entity.
- Must contain a key column to uniquely identify each record.
- Passes all of its columns to the output view produced by the intelligent lookup.
- May contain transactional, master, or general data.
- Must contain a column that can be used as the pairing column. You should choose:
 - A column intended to identify individual objects of the type stored in the lookup table. These columns often end in *ID* (such as a `Product ID`, `Customer ID`, or `Building ID`) or are some other kind of unique identifier, like `Email Address`.
 - If such a column does not exist, you may choose to create a calculated column and concatenate values from other columns into it to provide unique identifier values.
 - If the column does not exist and cannot be easily created, or if the objects in the input and lookup entities are of the same type, select the key column.

Your Lookup Entity

- Is a table or view that you want to join to your input entity, in order to enrich it.
- Must contain a key column to uniquely identify each record.
- Passes only columns that you identify as return columns to the output view produced by the intelligent lookup.
- Will generally contain master data, such as:
 - Product data
 - Employee data
 - Supplier data
 - Customer data

Note

Views containing input parameters (see [Create an Input Parameter \[page 231\]](#)) cannot be used as input or lookup entities in an intelligent lookup.

6.3.2 Create an Exact Match Rule

Create an exact match rule to compare values in input entity columns and lookup entity columns.

Procedure



1. Click the rule node to open its properties panel and set the following properties in the *General* section:

Property	Description
Business Name	Enter a name for your rule to identify it in the diagram.
Match Strategy	Choose <i>Exact Match</i> .
Input Scope	[read-only] Specifies which input records are processed by the rule.

Property	Description
Lookup Scope	[read-only] Specifies which lookup records are available for matching by the rule.

- Use the *Match Columns* section to specify columns to compare.

Drag a column from the input entity and drop it onto a column in the lookup entity to specify that the values in the two columns must match exactly. If you map additional columns, the values in each column pair must all match exactly.

Example	Matching Criteria
	<p>An input record is matched with a lookup record if:</p> <ul style="list-style-type: none"> The value in the input <code>County</code> column exactly matches the value in the lookup <code>County Name</code> column.
	<p>An input record is matched with a lookup record if:</p> <ul style="list-style-type: none"> The value in the input <code>Town</code> column exactly matches the value in the lookup <code>Town Name</code> column, and The value in the input <code>County</code> column exactly matches the value in the lookup <code>County Name</code> column.

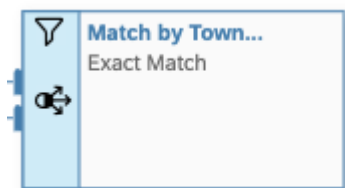
Note

You can only map a column to a column of the same data type.

- [optional] Specify filters to restrict the records from your input and/or lookup entities that will be considered for matching in the rule using the following sections:
 - Input Records Filter* - Specify a filter expression to restrict the input records that are considered for matching in the rule. You can reference any of the columns in the input entity and use any standard operators.
 - Lookup Records Filter* - Specify a filter expression to restrict the lookup records that are considered for matching in the rule. You can reference any of the columns in the lookup entity and use any standard operators.

For example entering `Revenue > 100000 AND Status = 'Active'` would restrict the records to be processed to only those meeting both these criteria.


If an *Input Records Filter* is defined, a filter symbol is displayed in the top-left corner of the rule symbol.



If a *Lookup Records Filter* is defined, a filter symbol is displayed in the bottom-left corner of the rule symbol.

4. [optional] Control case-sensitivity and whitespace trimming with the options in the *Advanced Settings* section.


Property	Description
Case-Sensitive	<p>Disable this option to allow case-insensitive matching.</p> <p>By default, exact match rules are case-sensitive so that, for example, an input column containing the string "paris" will not be matched with a lookup column containing "Paris".</p>
Trim Leading and Trailing Whitespace	<p>Disable this option to take into account any space characters preceding or following string values.</p> <p>By default, exact match rules trim whitespace so that, for example, an input column containing the string " London " or London will be matched with a lookup column containing London.</p>

5. Save, deploy, and run your intelligent lookup to see the results of your rule.
 - a. Click  (*Deploy*) to save and deploy your intelligent lookup.

If you have not previously saved the intelligent lookup, you will be prompted to provide a business and technical name for it. Enter appropriate names, and then click Save.

Note

You need to re-deploy your intelligent lookup each time you make a change to it. If you have pending changes, they will be saved automatically before deployment.

- b. Once deployment is complete, click  (*Run*) to run the intelligent lookup and review the results.

6.3.3 Create a Fuzzy Match Rule

Create a fuzzy match rule to compare strings in input entity columns with strings in lookup entity columns using a fault-tolerant search. Each match is allocated a percentage score and you can set thresholds for marking records as matched or requiring review.

Context

A fuzzy match rule lets you broaden the search for matches to include data that is spread across multiple columns. It matches records based on the percentage of matching characters in selected columns using the SAP HANA fuzzy search.

Procedure

1. Click the rule node to open its properties panel and set the following properties in the *General* section:

Property	Description
Business Name	Enter a name for your rule to identify it in the diagram.
Match Strategy	Choose <i>Fuzzy Match</i> to display the <i>Match Thresholds</i> section.
Input Scope	[read-only] Specifies which input records are processed by the rule.
Lookup Scope	[read-only] Specifies which lookup records are available for matching by the rule.

2. Set fuzzy match scores to decide when to place input records in the *Matched* and *Review* results categories.

Property	Description
Matched Records Score	<p>Set the minimum score needed to match an input record with a lookup record and place it in the <i>Matched</i> results category.</p> <p>If an input record is matched to more than one lookup records with a score reaching this threshold, it is placed in the <i>Multiple</i> results category.</p> <p>Default: 100%</p>
Review Records Score	<p>Set the minimum score needed to match an input record with a lookup record and place it in the <i>Review</i> results category.</p> <p>If an input record is matched to more than one lookup records with a score between this threshold and the <i>Matched Records Score</i>, it is placed in the <i>Multiple</i> results category.</p> <p>Default: 80%</p>

Property	Description
Unmatched Records Score	[read-only] Displays the score beneath which an input record cannot be matched. If an input record is not matched to any lookup record with a score above this threshold, then it is placed in the <i>Unmatched</i> results category.

3. Use the *Match Columns* section to specify columns to compare.

Click **+** (*Add*), and select one or more string columns from each entity to compare. When you select more than one column on either side, the columns are concatenated together in the order in which they are displayed in the list.

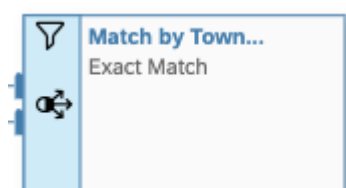
Example	Match Criteria
<div style="display: flex; justify-content: space-between; align-items: center;"> <div style="border: 1px solid #ccc; padding: 2px; display: flex; align-items: center; gap: 5px;"> AA Last Name ⊗ </div> <div style="border: 1px solid #ccc; padding: 2px; display: flex; align-items: center; gap: 5px;"> AA Last Name ⊗ </div> </div>	<p>An input record is matched with a lookup record if:</p> <ul style="list-style-type: none"> The value in the input <code>Last Name</code> column, Matches the value in the lookup <code>Last Name</code> column, With a score above the <i>Matched Records Score</i>.
<div style="display: flex; justify-content: space-between; align-items: center;"> <div style="border: 1px solid #ccc; padding: 2px; display: flex; align-items: center; gap: 5px;"> AA First Name ⊗ </div> <div style="border: 1px solid #ccc; padding: 2px; display: flex; align-items: center; gap: 5px;"> AA First Name ⊗ </div> </div> <div style="display: flex; justify-content: space-between; align-items: center; margin-top: 5px;"> <div style="border: 1px solid #ccc; padding: 2px; display: flex; align-items: center; gap: 5px;"> AA Last Name ⊗ </div> <div style="border: 1px solid #ccc; padding: 2px; display: flex; align-items: center; gap: 5px;"> AA Last Name ⊗ </div> </div>	<p>An input record is matched with a lookup record if the:</p> <ul style="list-style-type: none"> The value obtained by concatenating the input <code>First Name</code> and <code>Last Name</code> columns, Matches the value obtained by concatenating the lookup <code>First Name</code> and <code>Last Name</code> columns, With a score above the <i>Matched Records Score</i>.
<div style="display: flex; justify-content: space-between; align-items: center;"> <div style="border: 1px solid #ccc; padding: 2px; display: flex; align-items: center; gap: 5px;"> AA Full Name ⊗ </div> <div style="border: 1px solid #ccc; padding: 2px; display: flex; align-items: center; gap: 5px;"> AA First Name ⊗ </div> </div> <div style="display: flex; justify-content: space-between; align-items: center; margin-top: 5px;"> <div style="border: 1px solid #ccc; padding: 2px; display: flex; align-items: center; gap: 5px;"> AA Last Name ⊗ </div> </div>	<p>An input record is matched with a lookup record if:</p> <ul style="list-style-type: none"> The value in the input <code>Full Name</code> column, Matches the value obtained by concatenating the lookup <code>First Name</code> and <code>Last Name</code> columns, With a score above the <i>Matched Records Score</i>.

4. [optional] Specify filters to restrict the records from your input and/or lookup entities that will be considered for matching in the rule using the following sections:

- Input Records Filter* - Specify a filter expression to restrict the input records that are considered for matching in the rule. You can reference any of the columns in the input entity and use any standard operators.
- Lookup Records Filter* - Specify a filter expression to restrict the lookup records that are considered for matching in the rule. You can reference any of the columns in the lookup entity and use any standard operators.

For example entering `Revenue > 100000 AND Status = 'Active'` would restrict the records to be processed to only those meeting both these criteria.

If an *Input Records Filter* is defined, a filter symbol is displayed in the top-left corner of the rule symbol.



If a *Lookup Records Filter* is defined, a filter symbol is displayed in the bottom-left corner of the rule symbol.

5. [optional] Control auto-selection among multiple matches in the *Advanced Settings* section.

Property	Description
Multiple Match Processing	<p>Enables auto-selection among multiple matches when one match candidate scores conclusively higher than any other.</p> <p>The following settings are available:</p> <ul style="list-style-type: none"> • <i>Auto-Select</i> [default] - Automatically places pairing groups in the <i>Matched</i> results category if either of the following conditions is met : <ul style="list-style-type: none"> • Only one match candidate has a 100% match score. • Only one match candidate equals or exceeds the <i>Matched Record Score</i>. • <i>Always Review Match Candidates</i> - Keeps pairing groups in the <i>Matched</i> results category to allow you to manually review them even if one of the above conditions is met.

6. Save, deploy, and run your intelligent lookup to see the results of your rule.

- a. Click (*Deploy*) to save and deploy your intelligent lookup.

If you have not previously saved the intelligent lookup, you will be prompted to provide a business and technical name for it. Enter appropriate names, and then click Save.

Note

You need to re-deploy your intelligent lookup each time you make a change to it. If you have pending changes, they will be saved automatically before deployment.

- b. Once deployment is complete, click (*Run*) to run the intelligent lookup and review the results.

6.3.4 Configure the View Output by an Intelligent Lookup

Your intelligent lookup outputs a view that can be used as a source for other views. The view is available for use as soon as the intelligent lookup is run.

Context

You can use the output of your intelligent lookup as a source for a view or data flow, where it will be listed in the *Source Browser* on the *Repository* tab in the *Intelligent Lookups* category.

ⓘ Note

The following restrictions apply to the view produced by an intelligent lookup. You cannot:

- Expose an intelligent lookup for consumption directly (though it can serve as the source for a view that is exposed).
- Show an intelligent lookup in an E/R model.
- Share an intelligent lookup to another space.

Procedure

1. Click the *Output* node and configure the view that will be output by your intelligent lookup.
2. In the *General* section, enter the following properties:

Property	Description
Business Name	Enter a descriptive name to help users identify the object. This name can be changed at any time.
Technical Name	Displays the name used in scripts and code, synchronized by default with the <i>Business Name</i> . To override the default technical name, enter a new one in the field. Technical names can contain only alphanumeric characters and underscores.
	ⓘ Note Once the object is saved, the technical name can no longer be modified.
Semantic Usage	[read-only] The semantic usage of the view produced by an intelligent lookup is always set to <i>Relational Dataset</i> .
Include Review Records	Include all records that remain in the <i>Review</i> category in the output. Default: <i>On</i>

Property	Description
Include Unmatched Records	Include all records that remain in the <i>Multiple</i> and <i>Unmatched</i> categories after all rules are applied in the output. Input records in these categories have default values assigned in their return columns. Default: <i>Off</i>
Include "Unmatched Record" Column	Adds an <i>Unmatched Record</i> column to the intelligent lookup output. This column contains the value <i>Yes</i> if the record remains in either the <i>Multiple</i> or <i>Unmatched</i> category after all rules are applied.

- In the *Columns* section, review the columns that will be output in the view .
- [optional] To set a default value for a return column (to be used when the *Include Unmatched Records* option is enabled), hover over the return column, click **⋮ (Menu)** **▶▶ Set Default Value**, enter a default value in the dialog and click *OK*.

The default value is shown on the token under the column name.

6.3.5 Process Matched Results

Matched results are input records that are matched with a lookup record. You can reject any matches that you do not agree with.

Procedure

- Select the rule node and then click **🔍 (Preview Data)** to open the *Preview* panel.
- Click the *Matched* tab.

The panel contains a single table listing matched records grouped by pairing column value.

In our example, 4,138 out of 18,248 records have been matched with a score of 100% and are displayed grouped by our pairing column, *County*.

The first pairing group, *Buckinghamshire*, contains 288 records, but only two records are shown, as this is the number given in the *Rows per Pairing Group* field:

county_name	Match_Status	chargeDeviceID	SCORE	objectid	ctysua19nm	long	lat
✓ Buckinghamshire (288)	Matched (Rule)	0fe392f865944e20304b8e...	100%	126	Buckinghamshire	-0.80569	51.769661
✓ Buckinghamshire	Matched (Rule)	ed1f71d5ed1a92b021579...	100%	126	Buckinghamshire	-0.80569	51.769661
□ Hampshire (279)	Matched (Rule)	1cc2352eaf22f31007e079...	100%	134	Hampshire	-1.24735	51.044739
□ Hampshire	Matched (Rule)	e4c0c03020ab57a7193a...	100%	134	Hampshire	-1.24735	51.044739
□ Hertfordshire (191)	Matched (Rule)	446e520ce36e073a153fe...	100%	135	Hertfordshire	-0.27699	51.808788
□ Hertfordshire	Matched (Rule)	bdae89883f9206bc02b7...	100%	135	Hertfordshire	-0.27699	51.808788
□ Lancashire (189)							

The following tools are available above the table:

Tools	Description
↑ (Collapse All) / ↓ (Expand All)	Collapse or expand all pairing groups.
Reject	Reject the selected match and move the matched records to the <i>Unmatched</i> tab.
Rows per Pairing Group	Specify the maximum number of rows to display for each pairing group.

Note

Click in any table column header to reorder or filter by that column's values.

- Review each of the pairing groups that the rule has matched with lookup records.


If you don't agree with the matching of a pairing group, then select the checkbox to the left of the group, and click *Reject* to move all its records to the *Unmatched* tab.

Records in the *Matched* category are always included in the output of your intelligent lookup.

6.3.6 Process Review Results

Review results are input records that are matched with a lookup record with a score between the *Review* and *Matched* thresholds. You can approve or reject proposed matches.

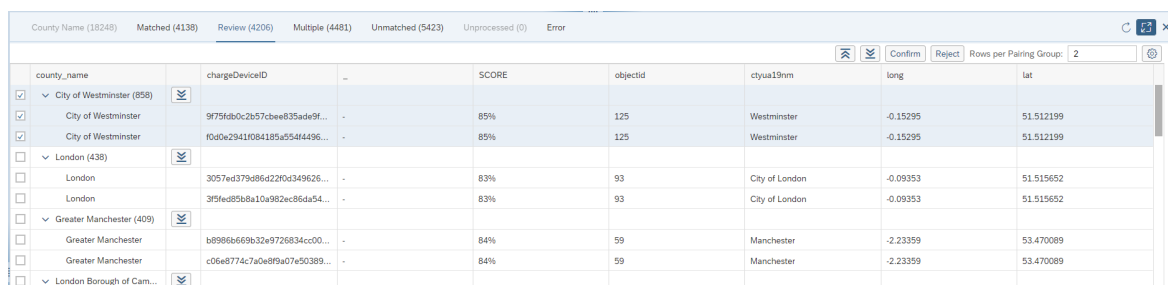
Procedure

- Select the rule node and then click  (*Preview Data*) to open the *Preview Data* panel.
- Click the *Review* tab.

The panel contains a single table listing matched records requiring review grouped by pairing column value.

In our example, 4,206 records out of 18,248 have been matched with a score between 80% and 100% and are displayed grouped by our pairing column, *County*.

The first pairing group, *City of Westminster*, contains 858 records, but only two records are shown, as this is the number given in the *Rows per Pairing Group* field:



county_name	chargeDeviceID	SCORE	objectid	ctyua19nm	long	lat
City of Westminster (858)						
<input checked="" type="checkbox"/> City of Westminster	9f75f60c2b57cbee835ade9f...	85%	125	Westminster	-0.15295	51.512199
<input checked="" type="checkbox"/> City of Westminster	f0d0e2941f084185a55414496...	85%	125	Westminster	-0.15295	51.512199
London (438)						
<input type="checkbox"/> London	3057ed379d86422f0d349626...	83%	93	City of London	-0.09353	51.515652
<input type="checkbox"/> London	3f5fe05b8a10a982ec86da54...	83%	93	City of London	-0.09353	51.515652
Greater Manchester (409)						
<input type="checkbox"/> Greater Manchester	b89866669b32e9726834cc00...	84%	59	Manchester	-2.23359	53.470089
<input type="checkbox"/> Greater Manchester	c06e8774c7a0e8f9a07e50389...	84%	59	Manchester	-2.23359	53.470089
London Borough of Cam...						

The following tools are available above the table:

Tools	Description
↑ (Collapse All) / ↓ (Expand All)	Collapse or expand all pairing groups.
Confirm	Confirm the selected match and move the records to the <i>Matched</i> tab.
Reject	Reject the selected match and move the matched records to the <i>Unmatched</i> tab.
Rows per Pairing Group	Specify the maximum number of rows to display for each pairing group.
⚙ (Settings)	Select and reorder table columns.

Note

Click in any table column header to reorder or filter by that column's values.

3. Review each of the pairing groups that the rule has tentatively matched:

Select the checkbox to the left of one or more pairing groups:


- If you agree with the proposed match, click *Confirm* to move all the records in the selected pairing groups to the *Matched* tab.
- If you don't agree with the proposed match, click *Reject* to move all the records in the selected pairing groups to the *Unmatched* tab.

Records in the *Review* category are included in the output of your intelligent lookup unless you deselect the *Include Review Records* option.

6.3.7 Process Multiple Match Results

Multiple results are input records that are matched with two or more lookup records. You can choose among the match candidates or apply a new rule to them.

Procedure

1. Select the rule node and then click  (*Preview Data*) to open the *Preview Data* panel.
2. Click the *Multiple* tab.

Note

If you have added a further rule to process the records in the *Multiple* category of a rule, then you can review the records in the first rule but you cannot process them.

The panel contains two tables:

- The left table lists input records for which the rule has found two or more match candidates with a score above the *Review Records Score*, grouped by pairing column value.
- The right table lists all of the lookup entity records. When you select a pairing group on the left, this table is filtered to show match candidates for that pairing column value in descending order of match score.

In our example, 4,481 records out of 18,248 have two or more matches with a score higher than 80% and are displayed grouped by our pairing column, *County*.

The first pairing group, *Royal Borough of Kensington and Chelsea* contains 430 records, but only two records are shown, as this is the number given in the *Rows per Pairing Group* field. As this group is selected, the lookup table is filtered to show the two match candidates for this pairing group:

county_name	chargeDeviceID	SCORE	objectId	ctyua19nm	long	lat
Royal Borough of Kensington and Chelsea	0d57b98b557ef21cc99cc175dfb27	84%	112	Kensington and Chelsea	-0.18976	51.496441
Royal Borough of Kensington and Chelsea	aaeb0dd9b00dd3c72f3d9af8d85ee6	80%	136	Kent	0.72137	51.192951

The following tools are available above the tables:

Tools	Description
Reject	Reject all the match candidates for the selected input records and move the input records to the <i>Unmatched</i> tab.
↑ (Collapse All) / ↓ (Expand All)	Collapse or expand all pairing groups.
Rows per Pairing Group	Specify the maximum number of rows to display for each pairing group.
Match	Match all the records in the selected pairing group to the selected lookup record.
Match Records	Select whether to display only <i>Match Candidates</i> for the selected pairing group or <i>All Records</i> in the lookup entity.
⚙️ (Settings)	Select and reorder table columns. In the right-hand lookup records table <i>View Settings</i> dialog, you can use the <i>Sort</i> and <i>Filter</i> tabs to sort on and filter by values in multiple columns.

Note

Click in any table column header to reorder or filter by that column's values.

3. Review each of the pairing groups that the rule has found multiple matches for.

Select a pairing group in the left table to filter the right table to show only the match candidates for it:

- If one of the match candidates in the right table is correct, select it and click *Match* to move all the records in the pairing group to the *Matched* tab.
- If none of the match candidates is correct, you can:


- Select to show *All Records* in the lookup table and manually search for the appropriate match.
 - Leave the record unmatched.
4. If appropriate, add a new rule to further process the remaining pairing groups and their match candidates.

Records that remain in the *Multiple* category at the end of the flow are not included in the output of your intelligent lookup unless you select the *Include Unmatched Records* option.

6.3.8 Process Unmatched Results

Unmatched results are input records that are not matched with any lookup record. You can try to manually match them or apply a new rule to them.

Procedure

1. Select the rule node and then click  (*Preview Data*) to open the *Preview Data* panel.
2. Click the *Unmatched* tab.

Note

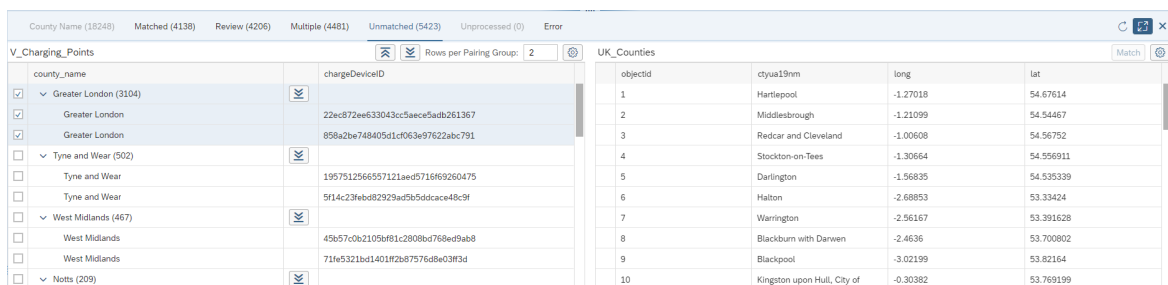
If you have added a further rule to process the records in the *Unmatched* category of a rule, then you can review the records but you cannot process them.

Two tables are displayed:

- The left table lists input records for which the rule has found no matches, grouped by pairing column value.
- The right table lists all of the lookup entity records.

In our example, 5,423 records out of 18,248 have no matches with a score higher than our 80% threshold and so appear on the *Unmatched* tab.

The first pairing group, *Greater London* contains 3,104 records:



county_name	chargeDeviceID	objectid	ctyua19nm	long	lat
Greater London	22ec872ee633043cc5a6ce5ad6261367	1	Hartlepool	-1.27018	54.67614
Greater London	858a2be748405d1cf063e97622abc791	2	Middlesbrough	-1.21099	54.54467
Greater London		3	Redcar and Cleveland	-1.00608	54.56752
Greater London		4	Stockton-on-Tees	-1.30664	54.556911
Greater London		5	Darlington	-1.56835	54.535339
Greater London		6	Halton	-2.68853	53.33424
Greater London		7	Warrington	-2.56167	53.391628
Greater London		8	Blackburn with Darwen	-2.4636	53.700802
Greater London		9	Blackpool	-3.02199	53.82164
Greater London		10	Kingston upon Hull, City of	-0.30382	53.769199

The following tools are available above the left table:

Tools	Description
↑ (Collapse All) / ↓ (Expand All)	Collapse or expand all pairing groups.
Rows per Pairing Group	Specify the maximum number of rows to display for each pairing group.
Match	Match all the records in the selected pairing group to the selected lookup record.
⚙️ (Settings)	Select and reorder table columns. In the right-hand lookup records table View Settings dialog, you can use the Sort and Filter tabs to sort on and filter by values in multiple columns.

Note

Click in any table column header to reorder or filter by that column's values.

- Review each of the pairing groups that the rule has found no matches for.

Select a pairing group in the left table and consult the lookup records in the right table:

- If you find a lookup record that is a correct match, select it and click [Match](#) to move all the records in the pairing group to the [Matched](#) tab.
- If you do not find a suitable lookup record, you can leave the record unmatched.

- If appropriate, add a new rule to further process the remaining pairing groups and their match candidates.

Records in the [Unmatched](#) category at the end of the flow are not included in the output of your intelligent lookup unless you select the [Include Unmatched Records](#) option.


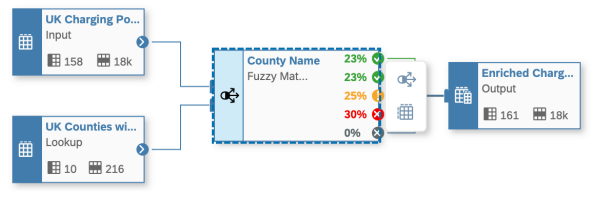
6.3.9 Example: Harmonizing County Data for UK Charging Sites

We have downloaded UK government data on electric vehicle charging sites, which we want to join to our sales data, aggregated by UK county.

Unfortunately, the downloaded data has some missing and inconsistent values for county names, so that many values do not match our master data for counties, and the `COUNTY` column cannot be reliably used to aggregate the data by county or join the table to location data. For example, the county `Argyll` and `Bute` is entered in several different ways in our electric vehicle charging data:

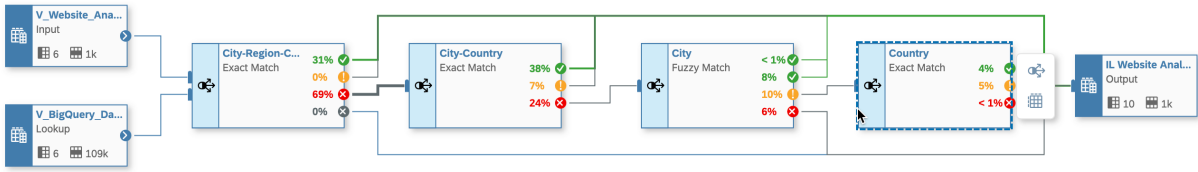
Electric Vehicle Charging Entity County Column Value	Geo Master Data County Name Column
Argyll	-
Argyll & Bute	-
Argyll and Bute	Argyll and Bute

Once we've loaded the raw data into SAP Datasphere:

Step	Result
<p>We create a view to:</p> <ul style="list-style-type: none"> Remove unnecessary columns Clean up some column names Create a new column to calculate the total wattage available in each station. 	
<p>We create an intelligent lookup, add our view as the input entity, and set the pairing column to <code>County</code>, as this is the column that contains the data that should be joined with our lookup entity.</p>	<div data-bbox="804 577 1402 689"> <p>Pairing Columns (1)</p> <p>AA County</p> </div>
<p>We add our Geo view as the lookup entity, and select the <code>County ID</code> and <code>County Name</code> columns as return columns.</p>	<div data-bbox="804 734 1402 913"> <p>Return Columns (2)</p> <p>22 County_ID</p> <p>AA County_Name</p> </div>
<p>We create a fuzzy rule to compare county names in each entity.</p>	<div data-bbox="804 936 1402 1294"> <p>Match Thresholds</p> <p>Matched Records Score ≥ 100 %</p> <p>Review Records Score ≥ 80 %</p> <p>Unmatched Records Score < 80 %</p> <p>Match Columns</p> <p>UK Charging Station... UK Counties (Focus...)</p> <p>AA County</p> <p>AA County_Name</p> </div>
<p>We deploy and run the intelligent lookup and our rule is able to match just under half the records.</p> <p>We review and process the remaining records in the Preview Data panel.</p>	

6.3.10 Example: Adding Latitude and Longitude Data with a Multi-Rule Intelligent Lookup

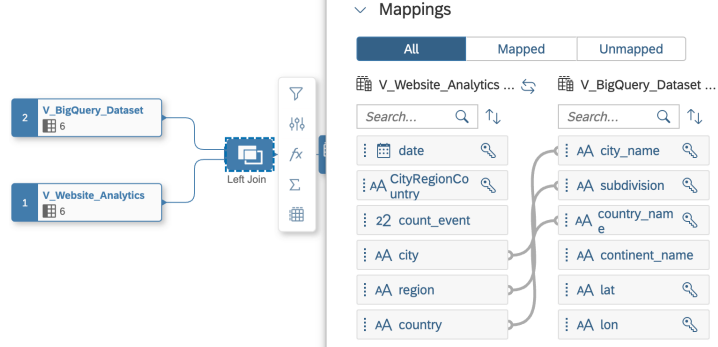
In this example, we want to enrich our website analytics data with latitude and longitude values from another view containing geo data for over 100,000 cities across the world. But the data identifying the city, country, and region from which each of our website clicks originates is not always complete and even where it is, it does not always match with our geo data values.



Step

Result

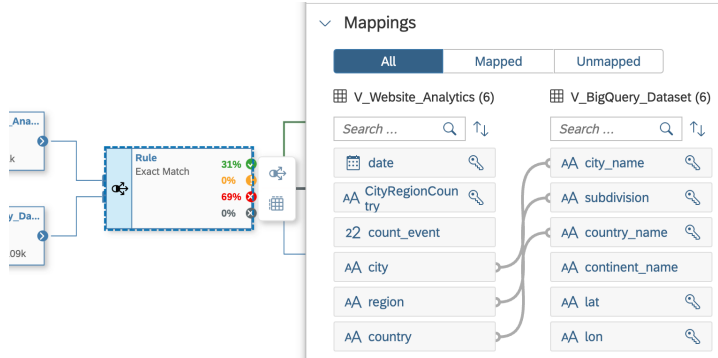
First, we try to join the two views by city, country, and region in a standard view but, since one or more of these columns may be empty, we find that only around 31% of our records can be matched.



We decide to join our entities via an intelligent lookup. We have previously created a calculated column to concatenate the City, Region, and Country columns to generate a unique identifier for each city and we select that as the pairing column.

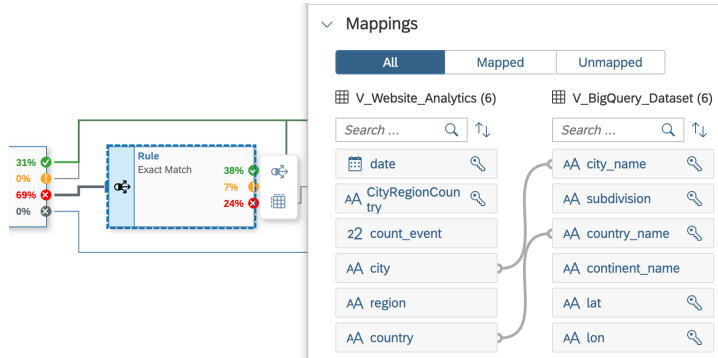


We recreate the same join mapping in an exact match rule and get the same result as for our view join.



We add a second exact match rule to treat the unmatched records, this time mapping on just city and country, and get:

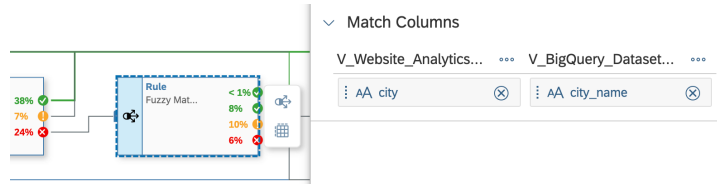
- 38% of the total records matched
- 7% of the records with multiple potential matches, most of which can be reviewed and manually resolved
- 24% still unmatched



Step

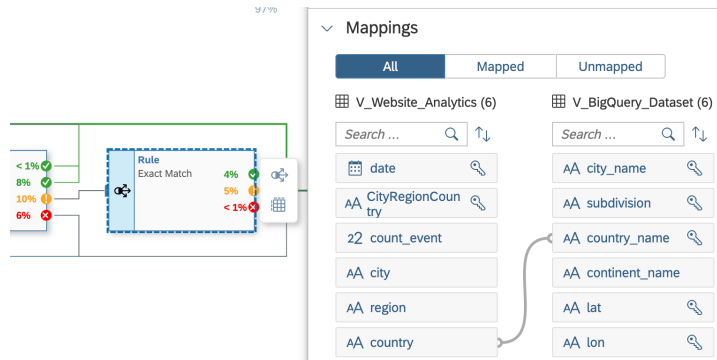
Result

Next, we take the remaining 24% of unmatched records, create a fuzzy rule on just `city` with the default 100% (match) and 80% (review) thresholds, and get:



- < 1% matched
- 8% matched but requiring review
- 10% with multiple matches
- 6% still unmatched

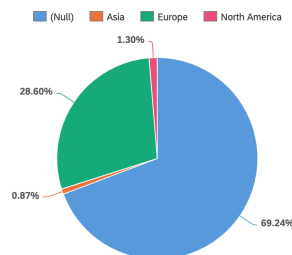
We create a final exact match rule to add a further condition for the multiple matches. The logic here is fuzzy match on `city` AND exact match on `country` and get:



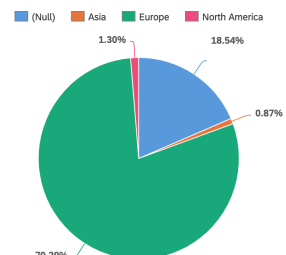
- 4% matched
- 5% with multiple matches
- <1% unmatched

Taken together, our four rules take us from just 31% of records matched with the standard view join, all the way to 79% matched with the intelligent lookup before any manual review.

Clicks Per Region (Standard View Join)



Clicks Per Region (Intelligent Lookup)



6.4 Creating an Entity-Relationship Model

Create an E/R model to import, visualize, edit, and deploy multiple tables and views together. You can use an E/R model to better understand a subset of the entities in your space, and to communicate this information to other stakeholders.


Context










An E/R model provides a diagram for arranging your data entities (tables and views) in relation to one another. You can:


- Add entities from the repository or import them from a connection or a CSN file, as well as creating new entities directly.
- Modify the properties of your entities including adding human-readable business names and creating associations directly in the diagram.
- Save and deploy all the contents of your model with a single action.



The work that you do in an E/R model benefits all the users in your space as they can use the entities that you import or enhance as sources in their views.

Procedure

1. In the side navigation area, click  (*Data Builder*), select a space if necessary, and click *New ER Model* to open the editor.
2. Add, import, or create entities in the diagram:
 - **Add objects from the repository** - Drag entities from the *Repository* tab of the *Source Browser* and drop them onto the diagram (see [Add Objects from the Repository \[page 292\]](#)).
 - **Import objects from a connection** - Drag entities from the *Sources* tab of the *Source Browser* and drop them onto the diagram (see [Import an Object from a Connection or Other Source \[page 294\]](#)).
 - **Import objects from a CSN/JSON file** (see [Importing Objects from a CSN/JSON File \[page 49\]](#)).
 - **Create a table** (see [Create a Table in an E/R Model \[page 288\]](#)).
 - **Create a view** (see [Create a View in an E/R Model \[page 289\]](#)).
3. Select an entity to access its toolbar:

Button	Action
 (<i>Add Column</i>)	Add a column to the selected table.
 (<i>Create View from Selection</i>)	Create a view from the selected entity (see Create a View in an E/R Model [page 289]).
 (<i>Create Table</i>)	Create a table along with an association pointing from the selected entity to the new table (see Create a Table in an E/R Model [page 288]).
 (<i>Create Association</i>)	Create an association from the selected entity to another entity (see Create an Association in an E/R Model Diagram [page 289]).
 (<i>Add Related Entities</i>)	Add entities that are related (by association) to the selected entity (see Add Related Entities to an E/R Model Diagram [page 291]).
 (<i>Preview Data</i>)	Preview the data for the selected entity (see Viewing or Previewing Data in Data Builder Objects [page 297]).
 (<i>Impact and Lineage Analysis</i>)	Open the Impact and Lineage Analysis diagram. This diagram enables you to understand the lineage and impacts of the selected object. (see Impact and Lineage Analysis [page 34] .)
 (<i>Edit Custom CSN Annotations</i>)	[optional] Click  (<i>Edit Custom CSN Annotations</i>) to open the <i>Edit Custom CSN Annotations</i> dialog.

Button	Action
 <i>(Open in New Tab)</i>	Open the selected entity in its own editor in a new tab.

4. Edit the selected entity's properties in the *Properties* panel.
5. Click  *(Save)* to save your model:
 - *Save* to save the entities displayed in your E/R model.
 - *Save As* to create a local a copy of the entities displayed in your E/R model. The model must have been previously saved at least once. The *Save* dialog opens. Enter new business and technical names and click *Save*.
6. Click  *(Deploy)* to deploy the entities displayed in your E/R model:
 - Newly-created entities are deployed for the first time.
 - Entities that have changes to deploy are re-deployed.


Note

The E/R model itself is not deployed and does not have a deployment status.


6.4.1 Create a Table in an E/R Model

You can create a local table directly in an E/R model.

Procedure

1. Click  *(New Table)*, move your pointer to empty space in the diagram and then click again to create the table.

Alternatively, select a table or view in the diagram and click its *Create Table* tool to create a table along with an association pointing to it from the selected entity.

2. [optional] Quickly add placeholder columns in the table symbol by clicking .
3. Click  *(Enter Full Screen)* in the side panel header and complete the definition of your table (see [Creating a Local Table \[page 106\]](#)).


Note

You can enable *Delta Capture* for your table. For more information, see [Capturing Delta Changes in Your Local Table \[page 118\]](#)

6.4.2 Create a View in an E/R Model

You can create a view directly in an E/R model, and complete the definition of the view in the graphical view editor.

Procedure

1. Select a table or view in the diagram as a source for the new view, and click  (*Create View from Selection*) to open the *Create View* dialog.
2. Enter a business name and a technical name for the new view.
Optionally, select *Open in View Builder* to open the view in the graphical view editor upon creation.
3. Click *OK* to create the view in the diagram.

If you selected *Open in View Builder*, you will be prompted to save your E/R model and the new view will open in the graphical view editor (see [Creating a Graphical View \[page 213\]](#)).


6.4.3 Create an Association in an E/R Model Diagram

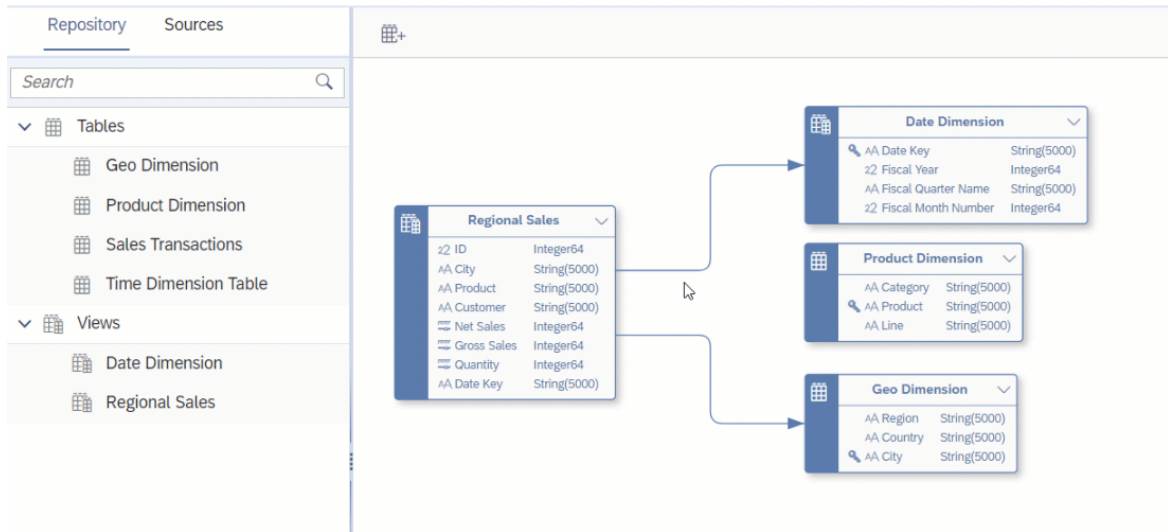
Create an association between two entities graphically in an E/R model diagram.

Context


In addition to this method, you can also create associations in the side panel in an E/R model, or in the table editor, graphical view editor, or SQL view editor (see [Create an Association \[page 234\]](#)).

Procedure

1. Select an object to open its context menu.
2. Click and hold  (*Create Association*), then release once you've placed your cursor over the target object.





The rules for creating associations depend on the *Semantic Usage* of the entity:

- A *Fact* can point to a:
 - *Dimension* - One attribute in the (source) *Fact* must be mapped to each (target) *Dimension* key column so that all target key columns are mapped.
 - *Text Entity* - An attribute in the (source) *Fact* must be mapped to the (target) *Text Entity* identifier key column.
 - A *Dimension* can point to a:
 - *Dimension* - One attribute in the (source) *Dimension* must be mapped to each (target) *Dimension* key column so that all target key columns are mapped.
 - *Text Entity* - An attribute in the (source) *Dimension* must be mapped to the (target) *Text Entity* identifier key column.
 - *Hierarchy* - The key attribute in the (source) *Dimension* must be mapped to the (target) *Hierarchy* child attribute key column.
 - A *Text Entity* must not point to other entities.
 - A *Hierarchy* will generally not point to other entities.
 - A *Hierarchy with Directory* must point to:
 - A *Dimension* acting as its directory - The hierarchy name attribute in the (source) hierarchy entity must be mapped to the primary key column in the (target) dimension.
 - Any non-leaf *Dimension* providing nodes to the hierarchy - The appropriate node type values columns in the (source) hierarchy must be mapped to the key columns in the (target) *Dimension*.
 - A *Relational Dataset* can point to any other entity and should generally follow the rules for dimensions.
3. In the *General* section, review the default *Business Name* and *Technical Name* and modify them if appropriate.
 4. Specify the mapping of join columns in the *Join* section:
 - A default mapping is automatically created by matching column names if possible. For example if the originating entity contains a column, **Product ID**, and the target entity has a column with the same name, then a default mapping is created between these two columns.
 - To delete a mapping, select the link and then click  (*Delete*).
 - To manually map columns, drag a column from the left list and drop it onto a column in the right list.
 - You can filter the *Join* section to show only mapped or unmapped pairs of columns.

- You can filter or sort the left or right column lists independently

Note


To delete an association, select it in the list and click  (*Delete Association*).

- To delete an association, select the arrow and click  (*Delete Association*).


6.4.4 Add Related Entities to an E/R Model Diagram

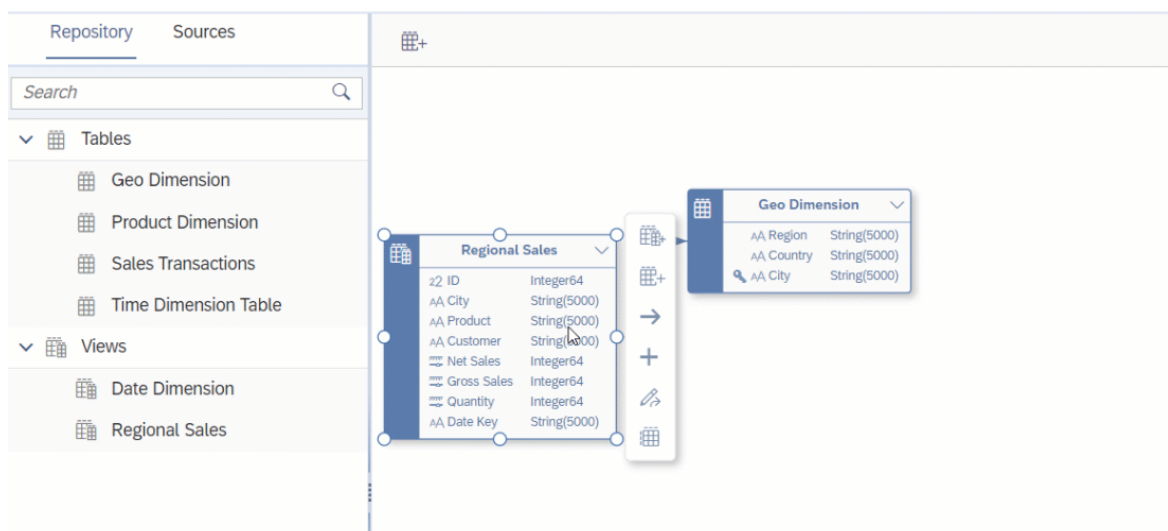
If a table or view in your E/R model diagram has previously been connected to other entities by associations, you can find these entities and add them to the diagram using the *Add Related Entities* dialog.

Procedure

- Selecting the entity to open its context menu, and then click  (*Add Related Entities*) to open the *Select Object* dialog.

Any entities that are connected to the current entity by an association and that aren't already present in the diagram are listed.

Find available objects by entering the object's name in the search bar or click  (*Show filters*) and filter by *Semantic Usage* or other criteria.



- Select the entities that you want to add to your diagram and click *Select*.

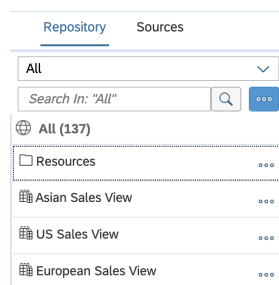
The entities and the associations connecting them are added to the diagram.


6.5 Using the Source Browser

You use the *Source Browser* to add objects as sources for your data flow, graphical view, SQL view, or intelligent lookup. In an E/R model you add objects to visualize them together in a diagram, including importing objects from connections and other sources, and prepare them for use in other editors.

The *Source Browser* is available in the left panel in each of these editors, and gives you access to all the objects that have been imported into or created in the space, and also to all the connections and other sources that are available in the space:

- The *Repository* tab lists all the tables, views, and intelligent lookups that are available in the space (including objects shared to the space). You can filter and sort the list as follows:



- Select a *Collection*:
 - *All* (default)
 - *Recent* - Objects that you recently opened
 - *My Objects* - Objects that you created
 - *Shared* - Objects that are shared to your space
 - *Favorites* - Objects that you have favorited
 - Enter one or more characters in the *Search* field and press *Enter* (or click *Search*). As you type, the field will begin proposing objects and search strings. Select an object to open it directly. Click on a string to trigger a search on it.
 - Click *...* (*More*) to access the *Sort* and *Filter* options.
 - Click a folder in the list to drill down into and restrict your search to the folder.
-
- The *Sources* tab lists all the connections and other data sources that have been integrated to the space from which you can import tables. These can include:
 - *Connections* - Lists all the connections that have been created in the space (see [Integrating Data via Connections](#)).
You can browse and search on up to 1,000 objects per connection. To browse all available objects, hover over any schema and click  (*Import from Connection*) to open the *Import Objects from Connection* dialog (see [Import Multiple Objects from a Connection \[page 296\]](#)).
 - Open SQL Schemas - Each Open SQL schema appears as a root node in this tab (see [Integrating Data via Database Users/Open SQL Schemas](#)).
 - HDI Containers - Each SAP HDI container added to the space appears as a root node in this tab (see [Exchanging Data with SAP SQL Data Warehousing HDI Containers](#)).

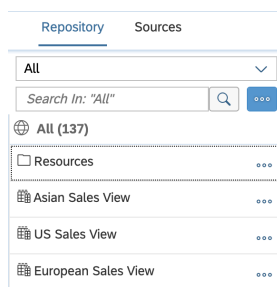
6.5.1 Add Objects from the Repository

Drag objects from the *Repository* tab of the *Source Browser* to add them as sources in your data flow, graphical view, SQL view, or intelligent lookup. In an E/R model, you add objects to visualize them together in a diagram and prepare them for use in other editors.

Procedure

1. If the *Source Browser* panel is not visible on the left of the screen, click *Source Browser* in the toolbar to show it.
2. Click the *Repository* tab.

The *Repository* tab lists all the tables, views, and intelligent lookups that are available in the space (including objects shared to the space). You can filter and sort the list as follows:



- Select a *Collection*:
 - *All* (default)
 - *Recent* - Objects that you recently opened
 - *My Objects* - Objects that you created
 - *Shared* - Objects that are shared to your space
 - *Favorites* - Objects that you have favorited
- Enter one or more characters in the *Search* field and press *Enter* (or click *Search*). As you type, the field will begin proposing objects and search strings. Select an object to open it directly. Click on a string to trigger a search on it.
- Click *...* (*More*) to access the *Sort* and *Filter* options.
- Click a folder in the list to drill down into and restrict your search to the folder.

3. When you find an object, you can preview it before adding it. Click *...* (*More*) and select:
 - *Show Info* - See additional properties and optionally open it in a new tab.
 - *Show Data Preview* - Preview the object's data (see [Viewing or Previewing Data in Data Builder Objects \[page 297\]](#)).
4. Drag the object from the *Repository Browser* and drop it on the diagram.

Note

In an E/R model, you can additionally::

- Add multiple objects simultaneously, click **+** (*Add from Repository*) to open the *Add Repository Object* dialog. Select the objects you want to import and click *OK*.
- Add objects that are related to those already in the diagram (see [Add Related Entities to an E/R Model Diagram \[page 291\]](#)).
- Add objects from a CSN file (see [Importing Objects from a CSN/JSON File \[page 49\]](#)).

6.5.2 Import an Object from a Connection or Other Source

Drag objects from the *Sources* tab of the *Source Browser* to add them as sources in your data flow, graphical view, or SQL view. In an E/R model, you can add objects from any connections and other sources, and prepare them for use in other editors.

Context


Note

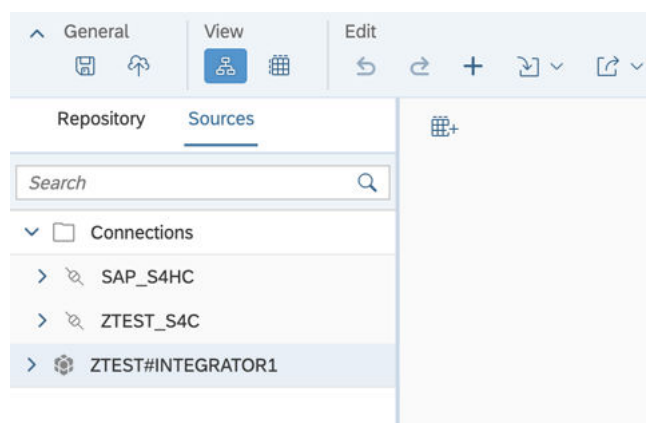
In the data flow editor, tables taken from connections are simply connected to (and not imported).

Procedure

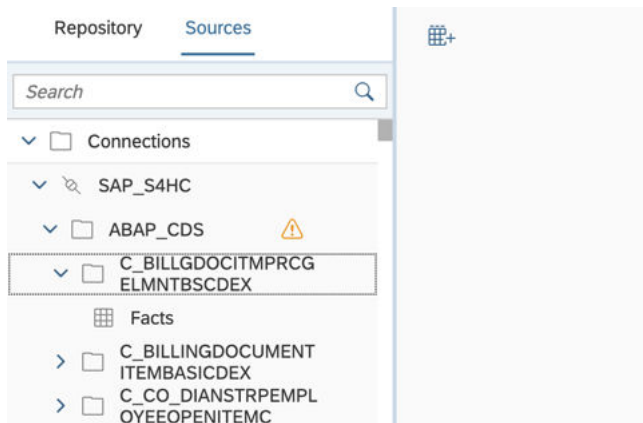
1. If the *Source Browser* panel is not visible on the left of the screen, click *Source Browser* in the toolbar to show it.
2. Click the *Sources* tab.

The *Sources* tab lists all the connections and other data sources that have been integrated to the space from which you can import tables. These can include:

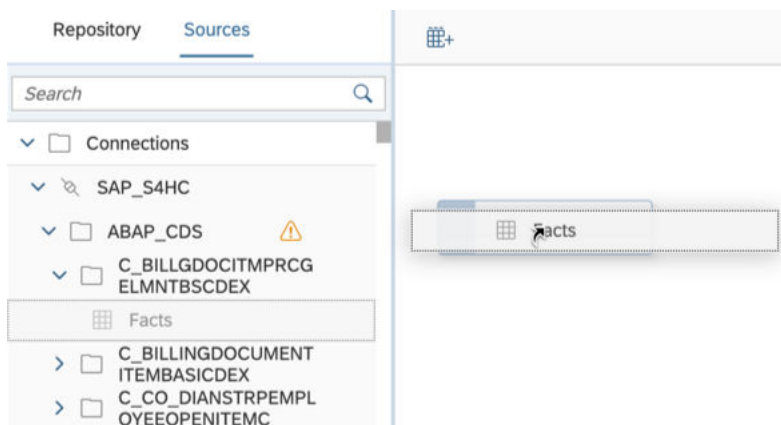
- *Connections* - Lists all the connections that have been created in the space (see [Integrating Data via Connections](#)).
You can browse and search on up to 1,000 objects per connection. To browse all available objects, hover over any schema and click  (*Import from Connection*) to open the *Import Objects from Connection* dialog (see [Import Multiple Objects from a Connection \[page 296\]](#)).
- Open SQL Schemas - Each Open SQL schema appears as a root node in this tab (see [Integrating Data via Database Users/Open SQL Schemas](#)).
- HDI Containers - Each SAP HDI container added to the space appears as a root node in this tab (see [Exchanging Data with SAP SQL Data Warehousing HDI Containers](#)).



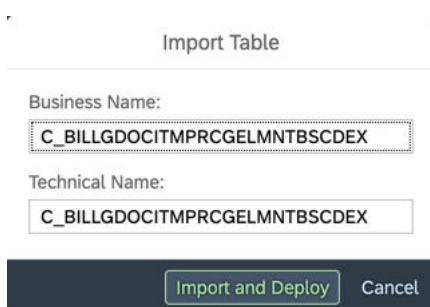
3. Expand the source from which you want to import a table and then expand any sub-folders until you arrive at the table you want to import:



4. Drag the table you want to import and drop it on the diagram canvas:



5. In the *Import Table* dialog, accept the default business and technical names or overwrite them and then click *Import and Deploy*:



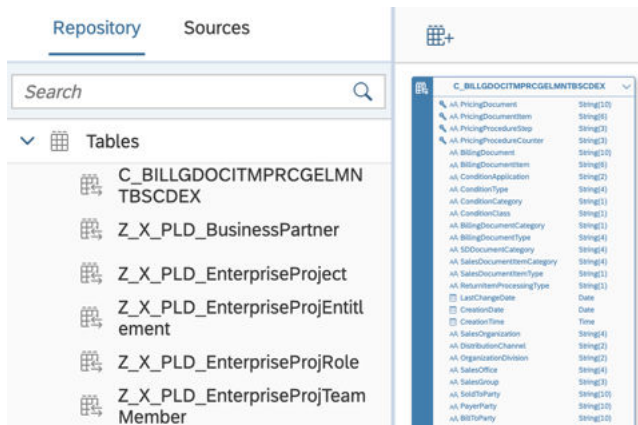
The source is added to your diagram and is imported to the repository. From now on it is available to everyone in your space on the *Repository* tab of the *Source Browser*:

- Connections - Tables are imported as remote tables

Note

In the data flow editor, tables taken from connections are simply connected to (and not imported).

- Open SQL queries - Tables are imported as local tables
- HDI containers - Tables are imported as local tables



6.5.3 Import Multiple Objects from a Connection


Use the *Import Objects from Connection* dialog to select multiple objects from a connection to add as sources in your data flow, graphical view, or SQL view. In an E/R model, you can add objects from a connections, and prepare them for use in other editors.

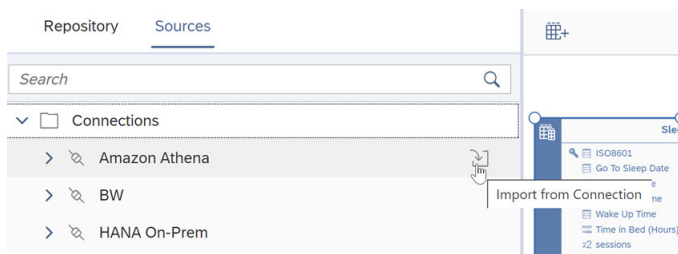
Context


Note

In the data flow editor, tables taken from connections are simply connected to (and not imported).

Procedure

1. Click the *Sources* tab of the *Source Browser*, expand the *Connections* node, and then hover over a sub-node and click the  (*Import from Connection*) button:



Alternatively, in an E/R model, click  (*Import*) in the toolbar, select a connection, and click *Next*.

The *Import Objects from Connection* dialog opens on the selected connection.

Note

ABAP SLT connections do not support selecting objects from this dialog (see [SAP ABAP Connections](#)).

2. On the *Available* tab:
 - a. Navigate in the connection in the left pane, and click a folder in the tree to show its contents in the right pane.
 - b. Select objects to be imported in the right pane.
 - c. [optional] Select further folders and select additional objects to be imported from them.
 - d. When all appropriate objects are selected, click *Next* (or click the *Selection* tab).
3. On the *Selection* tab:
 - a. Review the objects selected.
 - b. [optional] Modify the *Business Name* and/or *Technical Name* of one or more of the selected objects. These names are used to identify the objects in SAP Datasphere. While you can edit a *Business Name* at any time, the *Technical Name* cannot be edited after an object is imported.

Note

In the data flow editor, tables are simply connected to (and not imported) and there is therefore no need or option to set these names.

- c. [optional] To remove an object from the selection, select it and click *Remove from Selection*.
4. When you are satisfied with your selection, click *Import and Deploy* (or, in the data flow editor, *Add Selection*).

The dialog will show the progress for each object in the *Status* column and then close when the process is complete:

- Data flow - The objects are connected to and added to the diagram as sources.
- Graphical view - The objects are imported to the repository as remote tables and added to the diagram as sources.
- SQL view - The objects are imported to the repository as remote tables and are available on the *Repository* tab of the *Source Browser* to be added as sources.
- E/R model - The objects are imported to the repository as remote tables and added to the diagram.

6.6 Viewing or Previewing Data in Data Builder Objects

View the data contained in your tables and views and, when working in the graphical view editor, the data output by each of the nodes in the diagram. Preview the data contained in your tables and views when working in the E/R modeler or data flow.

This topic contains the following sections:

- [Viewing Data in Data Builder Objects \[page 298\]](#) in the graphical view editor, table editor, and SQL editor
- [Previewing Data in Data Flows and E/R Models \[page 300\]](#) in the E/R modeler and data flow

Note

Users with the standard *DW Modeler* role can preview data in any object in their space. Users with the *DW Viewer* role can only preview data output by views with the *Expose for Consumption* switch enabled. For more information, see [Roles and Privileges by App and Feature](#).

Viewing Data in Data Builder Objects

You can view the data contained in tables, views, and graphical view output nodes when working in the graphical view editor, table editor, and SQL editor.

Open the Data Viewer

You can view data in the following contexts in the data layer:

- Table editor - View the data stored in a local table, a remote table, or an Open SQL/HDI Schema.
- SQL view editor - View the SQL view output.
- Graphical view editor - Click on any diagram symbol (sources, intermediate nodes, or the output node).
- *Repository* tab of the *Source Browser* - Click on any entity to view its data before adding it to your diagram.

To open the *Data Viewer* panel, click  (*Show or hide data viewer*) in a data object context menu, in the main toolbar, or in a diagram symbol toolbar.

Note

- In the table editor or the view editor, the *Input Parameters* dialog appears if the remote table or the view contains one or more input parameters (see [Create an Input Parameter \[page 231\]](#)). A value might already appear if the input parameter has a predefined default value. You can edit values at any moment by clicking the input parameter strip and entering a new value or get access to a list of predefined values with the *Value Help Dialog* if your input parameter has a predefined default value.

Note

Previewing a SQL view isn't possible if one of the view's objects is shared from another space and has an input parameter.

- If a column has *Binary*, *Large Binary*, *ST_Point*, and *ST_Geometry* set as data type, the data viewer is unavailable and records are shown as *Cannot be shown*. If no data can be found, the data viewer shows it as *NULL*.



The viewer headers indicate the total number of rows that can be displayed by scrolling down.

Caution


If you view data of a view that is persisted, the data is read from the corresponding persistency table of this view. In case the view is changed in the editor and not yet deployed, the data preview bypasses the persistency table and reads data from the view definition.


Sort Data

You can control the data viewer in the following ways:

- Reorder your columns in the panel by drag and drop.
- Sort on a column by clicking the column header and then clicking  (*Sort Ascending*) or  (*Sort Descending*).
- Filter on a column. For more information, see the **Filter Data** section below.

Filter Data

Filter on a column by clicking the column header, clicking  (*Filter*) to open the *Define Filter* dialog. The advanced filtering options are available:

1. Choose the appropriate section for your filter. If your filter is meant to include data in the Data Viewer (you could say "I want my Data Viewer to show"), add your filter in the *Include* section. If your filter is meant to exclude data from the Data Viewer (you could say "I want my Data Viewer to hide"), add your filter in the *Exclude* section. When in the appropriate section, click  (*Add Filter*) to add a filter.
2. Select a column to filter on, a filtering option, and a value. You can add several filters. Click *OK* to apply the filter(s). The currently applied filters are displayed above the table.

Example


To only see senior employees born between 1960 and 1969, select the column containing birth dates, the filtering value *between*, and the earliest and latest dates needed. Once applied, the filter will be displayed in a strip above the table as following: **Birth Date (Jan 1, 1960...Dec 31, 1969)**.

Note

- The filtering options available depend on the data type of the column you filter on.
- The filtering option *empty* takes into account both empty and *NULL* values.
- Filters applied to text columns are case-sensitive.
- You can enter filter or sort values in multiple columns.




3. Click *Clear Filter* in the filter strip or  (*Remove Filter*) in the *Define Filter* dialog to remove the filter.

Choose Columns to Display

Show, hide, sort, and filter on multiple columns by clicking  (*Columns Settings*) to open the *View Settings* dialog. Three tabs allow you to change your data viewer:

- **Columns** - select columns as appropriate and click *OK*.
- **Sort** - select the column(s) to sort and if you want them sorted in and ascending or descending order.
- **Filter** - filter on columns.

When you're done, click *OK* to update the viewer or *Cancel* to erase your changes. To return to the default data viewer columns, click *Reset*.

- Refresh the data viewer at any time by clicking  (*Refresh*).
- Close the data viewer by clicking  (*Close*).
- In the *Graphical View* editor, preview the SQL generated for the node by clicking  (*Preview SQL*). Click *Copy* to copy the SQL code for pasting into the *SQL View editor* or elsewhere.

Note

The data view may not be fully displayed:

- If your data object is or relies on a remote table, viewing data may take a long time to load or cannot succeed. The data viewer also has a timeout limit of three minutes and will display an error on the *Errors* tab if it is unable to return data in that time. The restrictions inherent to remote tables are passed along to the output view. Replicate the remote table data to avoid long loading times and view instantly. For more information, see [Replicate Remote Table Data \[page 98\]](#).
- Your data object data type is a *HANA Numeric* data type and cannot be viewed. In such cases, change the data type of your object to the corresponding one supported by SAP Datasphere. For more information, see [Column Data Types \[page 110\]](#).

Previewing Data in Data Flows and E/R Models

You can preview data contained in tables or views when working in the E/R modeler or data flow. Previewing and viewing data work in similar ways, except for the limitations listed below.


Open the Data Preview

You can preview data in the following contexts in the data layer:

- Data flow editor - Click on source or output symbols (intermediate nodes do not support data preview).
- E/R model editor - Click on any diagram symbol (tables and views).

For more information, see the **Open the Data Viewer** section above.

Working with the data preview

- Sort data like in the data viewer. For more information, see the **Sort Data** section above.
- Filter on a column by clicking the column header, clicking  (*Filter*), and entering a value to filter on.
- Select columns to display as appropriate in the *Columns* dialog. For more information, see the **Choose Columns to Display** section above.

Restriction

Contrarily to the data viewer, the data preview has the following restrictions:

- The preview is limited to the first 1000 rows.
- The *Define Filter* dialog is unavailable.
- The *View Settings* dialog is unavailable. Only the viewer's *Columns* tab is available as the preview's *Columns* dialog.

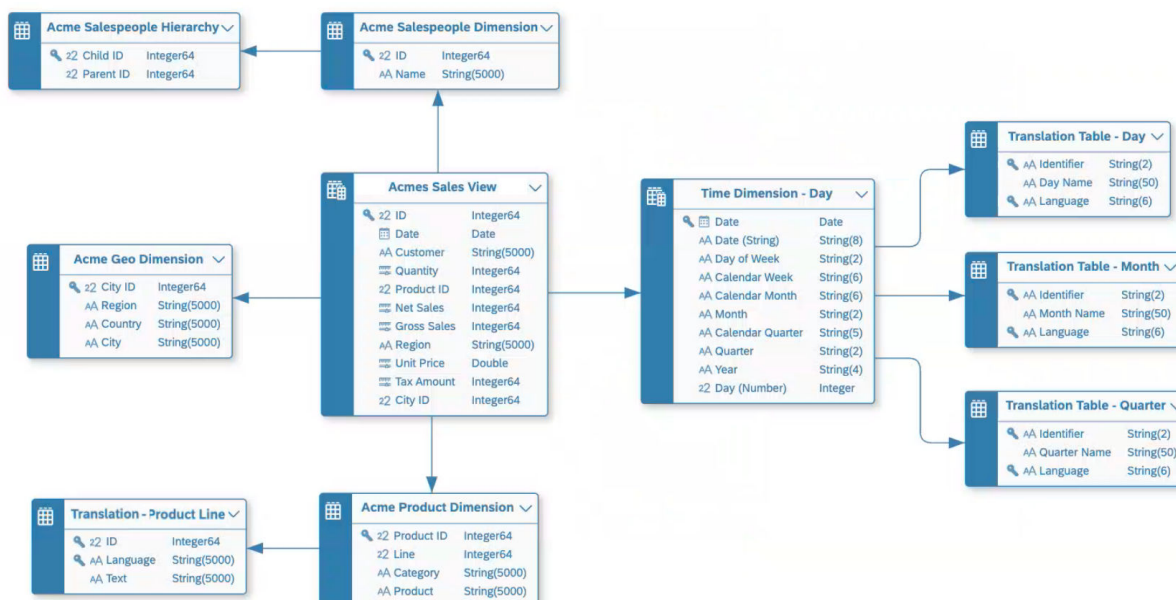
7 Modeling Data in the Data Builder

Users with the *DW Modeler* role can add semantic information to their entities and expose them directly to clients, tools, and apps, or combine, refine, and enrich them in tightly-focused analytic models for consumption in SAP Analytics Cloud, MS Excel, and other clients, apps, and tools.

This topic contains the following sections:

- [Model Facts, Dimensions, Texts, and Hierarchies \[page 301\]](#)
- [Identify Measures to Analyze in a Fact \[page 302\]](#)
- [Prepare Master Data for Grouping in a Dimension \[page 302\]](#)
- [Support Translations of Attributes with a Text Entity \[page 303\]](#)
- [Enable Drill-Down with a Hierarchy \[page 303\]](#)
- [Create Heterogeneous Hierarchies with a Hierarchy with Directory \[page 303\]](#)
- [Expose View Data for Consumption Outside SAP Datasphere \[page 303\]](#)
- [Combine Entities for Consumption in an Analytic Model \[page 304\]](#)
- [Create Objects and Act On Existing Objects \[page 304\]](#)

Model Facts, Dimensions, Texts, and Hierarchies



Use the *Semantic Usage* property to indicate the type of data contained in your entity:

- Select a *Semantic Usage* of *Fact* to indicate that your entity contains numerical measures that can be analyzed.

In our example, **Acme Sales View** is a fact containing sales data.

- Select a *Semantic Usage* of *Dimension* to indicate that your entity contains attributes that can be used to analyze and categorize measures defined in other entities.
In our example, four dimensions surround the fact, allowing us to analyze it by **Salespeople**, **Time**, **Product**, and **Geo** attributes.
- Select a *Semantic Usage* of *Text* to indicate that your entity contains strings with language identifiers to translate text attributes in other entities.
In our example, there are four translation entities to translate time and product dimension attributes.
- Select a *Semantic Usage* of *Hierarchy* to indicate that your entity contains parent-child relationships for members in a dimension.
In our example, the **Acme Salespeople Hierarchy** provides a hierarchy for the **Salespeople** dimension.

Identify Measures to Analyze in a Fact

Facts are entities that contain numerical measures that can be analyzed and are the principal type of object that is consumed by BI clients (see [Creating a Fact \[page 305\]](#)).

- To get started: Select a *Semantic Usage* of *Fact* to indicate that your entity contains numerical measures that can be analyzed.
- You must identify at least one measure (see [Specify Measures \[page 307\]](#)).
- You can create associations to dimensions and text entities (see [Create an Association \[page 234\]](#)).
- To expose your data for consumption in SAP Analytics Cloud, add it to an analytic model (see [Creating an Analytic Model \[page 337\]](#)).

Prepare Master Data for Grouping in a Dimension

Dimensions are entities that contain master data that categorize and group the numerical data contained in your measures (see [Creating a Dimension \[page 314\]](#)).

- To get started: Select a *Semantic Usage* of *Dimension* to indicate that your entity contains attributes that can be used to analyze and categorize measures defined in other entities.
- You must set at least one key column (see [Set Key Columns to Uniquely Identify Records \[page 318\]](#)).
- You can create associations to other dimensions, text entities, and hierarchies (see [Create an Association \[page 234\]](#)).
- You can add parent-child or level-based hierarchies to support drill-down (see [Add a Hierarchy to a Dimension \[page 321\]](#)).
- You can make your dimension time-dependent, so that its members can change over time (see [Enable Time-Dependency for a Dimension or Text Entity \[page 324\]](#)).

Support Translations of Attributes with a Text Entity

Text entities are entities that contain data to store strings in multiple languages for translating attributes in other entities (see [Create a Text Entity for Attribute Translation \[page 327\]](#)).

- To get started: Select a *Semantic Usage* of *Text* to indicate that your entity contains strings with language identifiers to translate text attributes in other entities.
- You must specify attributes and keys to uniquely identify a master data member and a language.
- You can make your text entity time-dependent, so that the texts it contains can change over time (see [Enable Time-Dependency for a Dimension or Text Entity \[page 324\]](#)).

Enable Drill-Down with a Hierarchy

External hierarchies are entities that contain data to define parent-child relationships for a dimension (see [Creating an External Hierarchy \[page 330\]](#)).

- To get started: Select a *Semantic Usage* of *Hierarchy* to indicate that your entity contains parent-child relationships for members in a dimension.
- You must specify the parent and child attributes and set the child attribute as a key.

Note

Parent-child and level-based hierarchies can also be defined directly in a dimension. See [Add a Hierarchy to a Dimension \[page 321\]](#).

Create Heterogeneous Hierarchies with a Hierarchy with Directory

A hierarchy with directory is an entity that contains one or more parent-child hierarchies and has an association to a directory dimension containing a list of the hierarchies. These types of hierarchy entities can include nodes from multiple dimensions (for example, country, cost center group, and cost center) and are commonly imported from SAP S/4HANA Cloud and SAP BW systems (see [Creating a Hierarchy with Directory \[page 331\]](#)).

Expose View Data for Consumption Outside SAP Datasphere


There are two methods for exposing view data for consumption outside SAP Datasphere:

- SAP Analytics Cloud (and Microsoft Excel via an SAP add-in) do not consume view data directly. Set the *Semantic Usage* of your view to *Fact* and then add it to an analytic model to expose it (see [Creating an Analytic Model \[page 337\]](#)). There is no need to enable the *Expose for Consumption* switch.
- Other third-party BI clients, tools, and apps can consume data from views with any *Semantic Usage* via OData or ODBC if the *Expose for Consumption* switch is enabled.

For more information, see [Consuming Data Exposed by SAP Datasphere](#).


Combine Entities for Consumption in an Analytic Model

Once your fact is ready for use, create an analytic model from it to consume its data in SAP Analytics Cloud (see [Creating an Analytic Model \[page 337\]](#)).

- To get started: In the side navigation area, click  (*Data Builder*), select a space if necessary, and click [New Analytic Model](#) to open the editor.
- You must add a fact as a source and can choose to copy all its measures, attributes and associated dimensions to the analytic model (see [Add a Source \[page 340\]](#)).
- You can deselect measures and attributes to leave only those that are relevant to answer your particular analytic question.
- You can create additional calculated and restricted measures (see [Add Measures \[page 343\]](#)).
- You can create multiple tightly-focused analytic models from a single fact, each providing only the data needed for a particular BI context, and enriched with appropriate variables, filters, and additional measures as necessary.

Create Objects and Act On Existing Objects

All the objects you import or create in the *Data Builder* are listed on the *Data Builder* start page. You can act on objects in the list in the following ways:

- Click one of the tabs to filter the list by object type.
- Click a tile to create a new object
- Click  (*Show filters*) to filter the list on collections and search by criteria. Click [Show More](#) to open a dialog with additional filter options.
- Enter a string in the [Search](#) field to filter the list on business and technical names and users.
- Click a column header to sort or filter the list by values in the column.
- Select one or more objects and use any of the following tools:

Tool	Description
New	Create <i>Data Builder</i> objects (independent of any selection).
Import	Import objects from files and connections: <ul style="list-style-type: none">• Import CSV File - Import data from a CSV file to create a local table (see Creating a Local Table from a CSV File [page 122])• Import Objects from CSN/JSON File - Import table and view definitions from a CSN file to create tables and views or import data flow definitions from a JSON file to create data flows. (see Importing Objects from a CSN/JSON File [page 49]).• Import Remote Tables - Import remote tables from a connection (see Import Remote Tables [page 91]).
Edit	Open the selected object in the appropriate editor. Alternatively, click the object directly in the list.

Tool	Description
Deploy	<p>Select one or more objects and deploy them at once.</p> <p>Choose from the following entity types:</p> <ul style="list-style-type: none"> • Local Table (see Creating a Local Table [page 106]) • Graphical View (see Creating a Graphical View [page 213]) • SQL View (see Creating an SQL View [page 250]) • Data Access Control (see Create a "Single Values" Data Access Control) • Analytic Model (see Creating an Analytic Model [page 337]) • Task Chain (see Creating a Task Chain [page 197]) <p>Other types of objects cannot be deployed.</p> <p>If one or more objects that you have selected cannot be deployed, the <i>Deploy</i> dialog opens, allowing you to review your selection. Click <i>Deploy</i> to deploy those objects listed on the <i>Deployable</i> tab, or <i>Cancel</i> to go back and alter your selection.</p>
Share	Share the selected tables and views to other spaces (see Sharing Tables and Views To Other Spaces [page 44]).
Impact and Lineage Analysis	View the objects that depend on an analyzed object (its impacts) and the objects on which the analyzed object depends (its lineage)(see Impact and Lineage Analysis [page 34]).
Delete	Delete the selected objects.

Note

If the object is used by one or more other objects then a dialog listing these dependencies opens, and the deletion is canceled.

Note

In various places in editors where there is not enough room to show both business and technical names for entities and columns, you can choose which of these names to display. Click ► *Profile* ► *Settings* ► *UI Settings* ► and select either *Show Business Name* or *Show Technical Name*.

7.1 Creating a Fact

Select a *Semantic Usage* of *Fact* to indicate that your entity contains numerical measures that can be analyzed.

Context

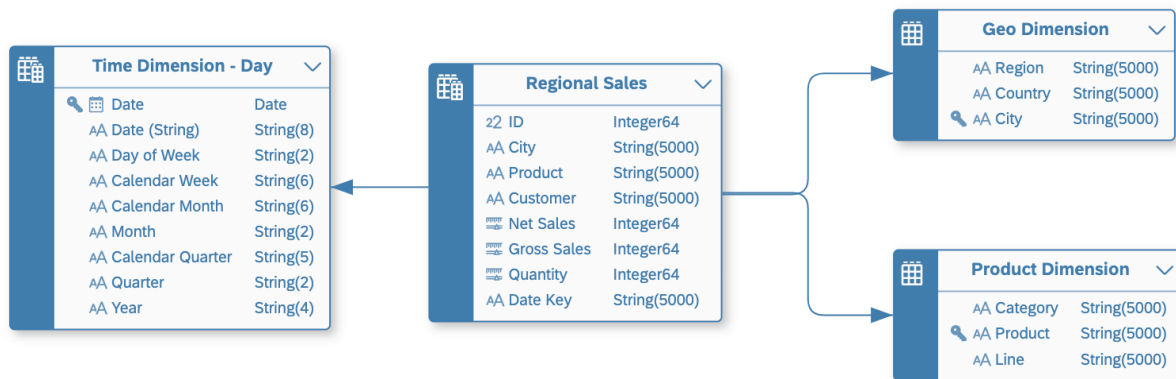
Facts typically contain transactional data, such as:

- Financial - Sales, Purchases, Orders

- Logistical - Deliveries, Orders
- Organizational - Attendance, Leave, Salaries, Expenses

In this example, *Regional Sales* is a *Fact* with:

- Three measures: *Net Sales*, *Gross Sales*, and *Quantity*.
- Associations to three dimensions: *Time Dimension - Day*, *Geo Dimension*, and *Product Dimension*.



Procedure

1. In the table editor or view editor output node side panel, set the *Semantic Usage* property to *Fact*.
2. [views] To make your view available for consumption outside SAP Datasphere enable the *Expose for Consumption* property (see [Consuming Data Exposed by SAP Datasphere](#)).

Note

To consume your data in SAP Analytics Cloud, add it to an analytic model (see [Creating an Analytic Model \[page 337\]](#)).

3. Specify one or more measures (see [Specify Measures \[page 307\]](#)).
4. Specify your attributes (see [Specify Attributes \[page 315\]](#)).
5. [optional] Set attributes as keys to indicate that the data they contain can uniquely identify records.

Note





You may set one or more key attributes for a *Fact*, but none are required.

To set an attribute as a key column, select the checkbox in the *Key* column or hover over the attribute in the side panel and click **⋮ (Menu) >> Set as Key**.

6. [optional] Create associations to point to other entities (see [Create an Association \[page 234\]](#)).

A *Fact* can point to a:

- *Dimension* - One attribute in the (source) *Fact* must be mapped to each (target) *Dimension* key column so that all target key columns are mapped.

- *Text Entity* - An attribute in the (source) *Fact* must be mapped to the (target) *Text Entity* identifier key column.
7. Complete or consult other sections as appropriate:
 - *Input Parameters* - Create input parameters to require the user to enter a value for use in calculated column, filter, and aggregation nodes (see [Create an Input Parameter \[page 231\]](#)).
 - *Data Persistence* - Persist the view data to improve performance (see [Persist View Data \[page 242\]](#)).
 - *Associations* - Create associations to other entities (see [Create an Association \[page 234\]](#)).
 - *Data Access Controls* - Add data access controls to apply row-based security and control access to individual rows based on various criteria (see [Securing Data with Data Access Controls](#)).
 - *Business Purpose* - Provide a description, purpose, contacts, and tags to help other users understand your entity.
 - *Dependent Objects* - If your entity is used as a source or association target for other entities, then they are listed here (see [Review the Objects That Depend on Your Table or View \[page 43\]](#)).
 8. Click  (*Save*)  *Save*  to save your entity or click  (*Deploy*) to save and deploy it immediately. For more information, see [Saving and Deploying Objects \[page 39\]](#).

7.1.1 Specify Measures

Measures appear in tables and views with a *Semantic Usage* of *Fact* and are columns containing numerical values that you want to analyze. Each *Fact* must contain at least one measure.

Typical measures include:







- Gross Sales
- Quantity
- Price
- Hours
- Distance

Measures are displayed in the *Measures* section of tables and views.

Note

In the graphical view and sql view editors, you can click the *Edit Columns* button in the *Measures* list to open it in a dialog.

By default, all columns in a fact are identified as attributes, and you must convert one or more into measures:

- In the graphical or SQL view editor:
 - To convert a numerical attribute into a measure, hover over it and click  (*Menu*)  *Change to Measure* , or drag the attribute token and drop it into the *Measures* list.
 - To convert a measure into an attribute, hover over it and click  (*Menu*)  *Change to Attribute* , or drag the measure token and drop it into the *Attributes* list.
- In the local or remote table editor:
 - To convert an attribute into a measure, select it in the *Attributes* list and click *Change to Measure*.

- To convert a measure into an attribute, select it in the [Measures](#) list and click [Change to Attribute](#).

Measures have the following properties:

Property	Description
Business Name	Enter a descriptive name to help users identify the object. This name can be changed at any time.
Technical Name	<p>Displays the name used in scripts and code, synchronized by default with the Business Name.</p> <p>To override the default technical name, enter a new one in the field. Technical names can contain only alphanumeric characters and underscores.</p> <div style="background-color: #f0f0f0; padding: 5px; border: 1px solid #ccc;"> <p>Note</p> <p>Once the object is saved, the technical name can no longer be modified.</p> </div>
Data Type	<p>Select the type of data that the column will contain.</p> <p>For a list of available data types and their supported data type conversions, see Column Data Types [page 110].</p>
Aggregation	<p>Select the default aggregation type for the column.</p> <p>Choose from the following:</p> <ul style="list-style-type: none"> SUM - [default] Calculate the total value for all rows. COUNT - Calculate the number of distinct values. MIN - Calculate the minimum value. MAX - Calculate the maximum value. NONE - Perform no aggregation. <div style="background-color: #f0f0f0; padding: 5px; border: 1px solid #ccc;"> <p>Note</p> <p>To modify the default aggregation type of a column in a view editor side panel, hover over it and click ⋮ (Menu) >> Change Aggregation > <Aggregation> >.</p> </div>
Semantic Type	<p>Select the appropriate semantic type to understand the value in the column.</p> <p>Choose from the following:</p> <ul style="list-style-type: none"> None - [default] No semantic meaning. Amount with Currency - The column contains monetary values. To complete the definition, select an attribute with the semantic type Currency Code in the Unit Column. Quantity with Unit - The column contains a physical values. To complete the definition, select an attribute with the semantic type Unit of Measure in the Unit Column.
Unit Column	Select an attribute with a semantic type of Currency Code or Unit of Measure to complete the semantic definition of the column.

Property	Description
Default Value	<p>Enter an appropriate default value for the column.</p> <p>Note This property is only displayed in the table editor.</p>
Not Null	<p>Select this option to indicate that the column must contain a value.</p> <p>Note This property is only displayed in the table editor.</p>
Show	<p>[deprecated] Deselect the checkbox to hide the column from users in SAP Analytics Cloud.</p> <p>You can now control the visibility of columns in SAP Analytics Cloud by choosing to include or exclude them in your analytic model (see Creating an Analytic Model [page 337]).</p> <p>Note To control the visibility of a column in a view editor side panel, hover over it and click ⋮ (Menu) ▶▶ Hide in Story or Show in Story.</p>

7.1.2 Specify Semantic Types for Measures and Attributes

Specify semantic types to identify the type of data in your columns (measures and attributes). Semantic types include values, quantities, dates, and geo and textual information.

This topic contains the following sections:

- [Specify Semantic Types \[page 309\]](#)
- [Amount and Quantity Measures \[page 310\]](#)
- [Currency Code and Unit of Measure Attributes \[page 310\]](#)
- [Language and Text Attributes \[page 310\]](#)
- [Image URL Attributes \[page 310\]](#)
- [Geolocation Attributes \[page 311\]](#)
- [Business Date Attributes \[page 311\]](#)
- [Calendar, Fiscal, and System Date Attributes \[page 312\]](#)

Specify Semantic Types

Semantic types are only available for measures and attributes in entities with a *Semantic Usage* of *Fact*, *Dimension* or *Text*.

To specify a semantic type, hover over the attribute or measure, and click **⋮ (Menu)** **▶▶ Semantic Type** **> <Type>**.

Amount and Quantity Measures

These types identify the type of value that the measure contains:

Name	Description
Amount with Currency	Specifies that the measure contains a monetary amount. You should additionally select an attribute with the semantic type <i>Currency Code</i> in the <i>Unit Column</i> column (see Currency Code and Unit of Measure Attributes [page 310]).
Quantity with Unit	Specifies that the measure contains a quantity of some kind. You should additionally select an attribute with the semantic type <i>Unit of Measure</i> in the <i>Unit Column</i> column (see Currency Code and Unit of Measure Attributes [page 310]).

Currency Code and Unit of Measure Attributes

These types are used with measure types (see [Currency Code and Unit of Measure Attributes \[page 310\]](#)) to identify the currency or unit contained in the measure:

Name	Description
Currency Code	A currency code. This can be either an ISO code or an SAP currency code (data type CUKY).
Unit of Measure	A unit of measure.

Language and Text Attributes

These types are typically used in text entities and identify columns containing language codes and strings in the selected language

Name	Description
Language	A language code.
Text	A human-readable text that is not necessarily language-dependent.

Image URL Attributes

This type identifies attributes containing URLs to image files:

Name	Description
Image URL	A URL pointing to an image file. The image will be displayed when the attribute is shown in an SAP Analytics Cloud table.

Note

This semantic type is only available for attributes contained in an entity with semantic type *Dimension*.

Geolocation Attributes

These types identify attributes containing latitudes, longitudes, and other geolocation information:

Name	Description
Geolocation - Cartoid	A geographical point.

Note

Not supported in SAP Analytics Cloud.

Geolocation - Latitude	A latitude value for a city or other geographical point.
Geolocation - Longitude	A longitude value for a city or other geographical point.
Geolocation - Normalized Name	A readable name for a city or other geographical point.

Note

Not supported in SAP Analytics Cloud.

Business Date Attributes

These types identify attributes containing dates for which information is being requested or is valid:

Note

For information about SAP Datasphere support for time-dependent dimensions using *Business Date - At*, *Business Date - From*, and *Business Date - To* semantic types, see [Enable Time-Dependency for a Dimension or Text Entity \[page 324\]](#).

Name	Description
Business Date - At	A key date.

Name	Description
Business Date - From	A date, timestamp, or interval that defines the validity of the data of the database table record from a business point of view. Use <i>System Date - Created at</i> to represent the date when the record was created.
Business Date - To	A date, timestamp, or interval that defines the validity of the data of the database table record from a business point of view.

Calendar, Fiscal, and System Date Attributes

The types identify attributes containing various kinds of dates:

Name	Description
Calendar - Day of Month	A day number relative to a calendar month. The number must be in the range 1 - 31.
Calendar - Day of Year	A day number relative to a calendar year. The number must be in the range 1 - 366.
Calendar - Halfyear	A calendar halfyear number as a string following the logical pattern H consisting of a single digit. The string must match the regex pattern [1-2].
Calendar - Month	A calendar month number as a string following the logical pattern MM consisting of two digits. The string must match the regex pattern 0 [1-9] 1 [0-2].
Calendar - Quarter	A calendar quarter number as a string following the logical pattern Q consisting of a single digit. The string must match the regex pattern [1-4].
Calendar - Week	A calendar week number as a string following the logical pattern WW consisting of two digits. The string must match the regex pattern 0 [1-9] [1-4] [0-9] 5 [2-3].
Calendar - Year	A year number as a string following the logical pattern (- ?) YYYY (Y *) consisting of an optional minus sign for years B.C., followed by at least four digits. The string must match the regex pattern - ? ([1-9] [0-9] { 3 , } 0 [0-9] { 3 }) .
Calendar - Year Half-year	A calendar halfyear number as a string following the logical pattern (- ?) YYYY (Y *) H consisting of an optional minus sign for years B.C., followed by at least five digits, where the last digit represents the halfyear. The string must match the regex pattern - ? ([1-9] [0-9] { 3 , } 0 [0-9] { 3 }) [1-2] .

Name	Description
Calendar - Year Month	<p>A calendar year and month as a string following the logical pattern (- ?) YYY Y (Y *) MM consisting of an optional minus sign for years B.C., followed by at least six digits, where the last two digits represent the months January to December.</p> <p>The string must match the regex pattern - ? ([1 - 9] [0 - 9] { 3 , } 0 [0 - 9] { 3 }) (0 [1 - 9] 1 [0 - 2]) .</p>
Calendar - Year Quarter	<p>A calendar year and quarter as a string following the logical pattern (- ?) YYY Y (Y *) Q consisting of an optional minus sign for years B.C., followed by at least five digits, where the last digit represents the quarter.</p> <p>The string must match the regex pattern - ? ([1 - 9] [0 - 9] { 3 , } 0 [0 - 9] { 3 }) [1 - 4] .</p>
Calendar - Year Week	<p>A calendar year and week as a string following the logical pattern (- ?) YYY Y (Y *) WW consisting of an optional minus sign for years B.C., followed by at least six digits, where the last two digits represent week number in the year.</p> <p>The string must match the regex pattern - ? ([1 - 9] [0 - 9] { 3 , } 0 [0 - 9] { 3 }) (0 [1 - 9] [1 - 4] [0 - 9] 5 [2 - 3]) .</p>
Fiscal - Period	<p>A fiscal period as a string following the logical pattern PPP consisting of three digits. This fiscal period usually is a quarter of a year.</p> <p>A fiscal period is covered by financial reports, for example, an annual report covers a fiscal period of one year, but a quarterly report includes accounting data for three months.</p> <p>The string must match the regex pattern [0 - 9] { 3 } .</p>
Fiscal - Year	<p>A fiscal year number as a string following the logical pattern YYYY consisting of four digits.</p> <p>The string must match the regex pattern [1 - 9] [0 - 9] { 3 } .</p>
Fiscal - Year Period	<p>A fiscal year and period as a string following the logical pattern YYYYPPP consisting of seven digits. The last three digits represent the fiscal period in the year.</p> <p>The string must match the regex pattern ([1 - 9] [0 - 9] { 3 }) ([0 - 9] { 3 }) .</p>
Fiscal - Year Variant	<p>A fiscal year variant, which describes the number of periods in a fiscal year and how they match the calendar year.</p>
System Date - Created at	<p>A timestamp for when the database record was created.</p>
System Date - Last changed at	<p>A timestamp for when the database record was last changed.</p>

7.1.3 Exposing a View For Consumption

When your view is ready, you can make its data available for consumption in SAP Analytics Cloud and other clients, tools, and apps.

There are two methods for exposing view data for consumption outside SAP Datasphere:

- SAP Analytics Cloud (and Microsoft Excel via an SAP add-in) do not consume view data directly. Set the *Semantic Usage* of your view to *Fact* and then add it to an analytic model to expose it (see [Creating an Analytic Model \[page 337\]](#)). There is no need to enable the *Expose for Consumption* switch.
- Other third-party BI clients, tools, and apps can consume data from views with any *Semantic Usage* via OData or ODBC if the *Expose for Consumption* switch is enabled.

For more information, see [Consuming Data Exposed by SAP Datasphere](#).

7.2 Creating a Dimension

Select a *Semantic Usage* of *Dimension* to indicate that your entity contains attributes that can be used to analyze and categorize measures defined in other entities.

Context

Typical types of dimensions include:

- Geography - Region, Country, State, City
- Product - Range, Category, Product
- Customer
- Organization - Company, Department, Organization Unit
- Time - Year, Quarter, Month, Day

Note

In order to use a time dimension in SAP Analytics Cloud, you must create your dimension by following the procedure at [Create Time Data and Dimensions](#). Manually-created or other time dimensions may not function correctly.




Procedure

1. In the table editor or view editor output node side panel, set the *Semantic Usage* property to *Dimension*.
2. Specify your attributes (see [Specify Attributes \[page 315\]](#)).
3. Set attributes as keys to indicate that the data they contain can uniquely identify records.

Note

You must set one or more key attributes for a *Dimension*:

- One or more identifier columns
- Validity Start Date - if time-dependency is required (see [Enable Time-Dependency for a Dimension or Text Entity \[page 324\]](#))

To set an attribute as a key column, select the checkbox in the *Key* column or hover over the attribute in the side panel and click  (Menu)  **Set as Key** .





4. [optional] Create associations to point to other entities (see [Create an Association \[page 234\]](#)).

A *Dimension* can point to a:

- *Dimension* - One attribute in the (source) *Dimension* must be mapped to each (target) *Dimension* key column so that all target key columns are mapped.
- *Text Entity* - An attribute in the (source) *Dimension* must be mapped to the (target) *Text Entity* identifier key column.
- *Hierarchy* - The key attribute in the (source) *Dimension* must be mapped to the (target) *Hierarchy* child attribute key column.

5. Complete or consult other sections as appropriate:

- *Input Parameters* - Create input parameters to require the user to enter a value for use in calculated column, filter, and aggregation nodes (see [Create an Input Parameter \[page 231\]](#)).
- *Data Persistence* - Persist the view data to improve performance (see [Persist View Data \[page 242\]](#)).
- *Associations* - Create associations to other entities (see [Create an Association \[page 234\]](#)).
- *Data Access Controls* - Add data access controls to apply row-based security and control access to individual rows based on various criteria (see [Securing Data with Data Access Controls](#)).
- *Business Purpose* - Provide a description, purpose, contacts, and tags to help other users understand your entity.
- *Dependent Objects* - If your entity is used as a source or association target for other entities, then they are listed here (see [Review the Objects That Depend on Your Table or View \[page 43\]](#)).

6. Click  (Save)  **Save**  to save your entity or click  (Deploy) to save and deploy it immediately.

For more information, see [Saving and Deploying Objects \[page 39\]](#).

7.2.1 Specify Attributes

Attributes are all columns that are not identified as measures, and can contain identifiers, master data, and other data that is used in support of analytics. Attributes appear in tables and views with any *Semantic Usage* except *Relational Dataset*.

Attributes are used in:

- Facts - to contain the keys of associated dimensions. For example, *Product ID* or *Store ID*.
- Dimensions - to contain categories for analyzing data and the keys for identifying them. For example, *Country*, *Sales Organization*, *Customer*.
- Text entities - to contain texts in multiple languages and the keys for identifying them.
- Hierarchies - to define parent-child relationships between dimension members.

Attributes are displayed in the *Attributes* section of tables and views.

ⓘ Note

In the graphical view and sql view editors, you can click the *Edit Columns* button in the *Attributes* list to open it in a dialog.

Attributes have the following properties:

Property	Description
Key	<p>Select the checkbox to specify that the column is a primary key column. Key columns cannot be null.</p> <p>ⓘ Note</p> <p>To set or remove an attribute as a primary key column in a view editor side panel, hover over it and click ⋮ (Menu) > Set as Key or Remove as Key.</p>
Business Name	<p>Enter a descriptive name to help users identify the object. This name can be changed at any time.</p>
Technical Name	<p>Displays the name used in scripts and code, synchronized by default with the <i>Business Name</i>.</p> <p>To override the default technical name, enter a new one in the field. Technical names can contain only alphanumeric characters and underscores.</p> <p>ⓘ Note</p> <p>Once the object is saved, the technical name can no longer be modified.</p>
Data Type	<p>Select the type of data that the column will contain.</p> <p>For a list of available data types and their supported data type conversions, see Column Data Types [page 110].</p>

Property	Description
Semantic Type	<p>Select the appropriate semantic type to understand the value in the column.</p> <p>Choose from the following:</p> <ul style="list-style-type: none"> • <i>None</i> - [default] No semantic meaning. • <i>Currency Code</i> - The column specifies the currency for one or more measures with the semantic type <i>Amount with Currency</i>. Attributes with this semantic type can be selected in the <i>Measures</i> list <i>Unit Column</i>. • <i>Unit of Measure</i> - The column specifies the unit for one or more measures with the semantic type <i>Quantity with Unit</i>. Attributes with this semantic type can be selected in the <i>Measures</i> list <i>Unit Column</i>. • <i>Text</i> - The column contains text. Attributes with this semantic type can be selected in the <i>Attributes</i> list <i>Label Column</i>. • <i>Business Date...</i>, <i>Fiscal...</i>, <i>Calendar...</i>, <i>System Date...</i> - The column contains a date or fiscal or calendar period. • <i>Language</i> - The column contains a language code. • <i>Geolocation...</i> - The column contains geo data. <p>For more information, see Specify Semantic Types for Measures and Attributes [page 309].</p>
Label Column	Select an attribute with a semantic type of <i>Text</i> to act as the label for the column.
Default Value	<p>Enter an appropriate default value for the column.</p> <div style="background-color: #f0f0f0; padding: 5px; border: 1px solid #ccc;"> <p>Note</p> <p>This property is only displayed in the table editor.</p> </div>
Not Null	<p>Select this option to indicate that the column must contain a value.</p> <div style="background-color: #f0f0f0; padding: 5px; border: 1px solid #ccc;"> <p>Note</p> <p>This property is only displayed in the table editor.</p> </div>
Show	<p>[deprecated] Deselect the checkbox to hide the column from users in SAP Analytics Cloud.</p> <p>You can now control the visibility of columns in SAP Analytics Cloud by choosing to include or exclude them in your analytic model (see Creating an Analytic Model [page 337]).</p> <div style="background-color: #f0f0f0; padding: 5px; border: 1px solid #ccc;"> <p>Note</p> <p>To control the visibility of a column in a view editor side panel, hover over it and click ⋮ (Menu) ▶▶ Hide in Story ▶ or Show in Story.</p> </div>

In certain contexts, further actions are available in the *Attributes* list:

- [dimensions] Create hierarchies as appropriate (see [Add a Hierarchy to a Dimension \[page 321\]](#)).
- [graphical or SQL views] Set any filters that you want to prompt users in SAP Analytics Cloud to specify a value for the attribute (see [Create a Story Filter \(Deprecated\) \[page 353\]](#)).

7.2.2 Set Key Columns to Uniquely Identify Records

Set one or more columns as keys to uniquely identify records in an entity. Each value in a key column (or each combination of values, when two or more key columns are set) must be unique.

It is important to set the necessary keys for your entity, based on its *Semantic Usage*, and to be aware of the requirements for mapping to these keys when creating associations:

Semantic Usage	Keys Required	Mapping Keys When Creating Associations
<p><i>Fact</i></p> <p>See Creating a Fact [page 305].</p>	<p>You may set one or more key attributes for a <i>Fact</i>, but none are required.</p>	<p>A <i>Fact</i> can point to a:</p> <ul style="list-style-type: none"> • <i>Dimension</i> - One attribute in the (source) <i>Fact</i> must be mapped to each (target) <i>Dimension</i> key column so that all target key columns are mapped. • <i>Text Entity</i> - An attribute in the (source) <i>Fact</i> must be mapped to the (target) <i>Text Entity</i> identifier key column.
<p><i>Dimension</i></p> <p>See Creating a Dimension [page 314].</p>	<p>You must set one or more key attributes for a <i>Dimension</i>:</p> <ul style="list-style-type: none"> • One or more identifier columns • Validity Start Date - if time-dependency is required (see Enable Time-Dependency for a Dimension or Text Entity [page 324]) 	<p>A <i>Dimension</i> can point to a:</p> <ul style="list-style-type: none"> • <i>Dimension</i> - One attribute in the (source) <i>Dimension</i> must be mapped to each (target) <i>Dimension</i> key column so that all target key columns are mapped. • <i>Text Entity</i> - An attribute in the (source) <i>Dimension</i> must be mapped to the (target) <i>Text Entity</i> identifier key column. • <i>Hierarchy</i> - The key attribute in the (source) <i>Dimension</i> must be mapped to the (target) <i>Hierarchy</i> child attribute key column.
<p><i>Text Entity</i></p> <p>See Create a Text Entity for Attribute Translation [page 327].</p>	<p>You must set two or three key attributes for a <i>Text Entity</i>:</p> <ul style="list-style-type: none"> • Identifier • Language Identifier • Validity Start Date - if time-dependency is required (see Enable Time-Dependency for a Dimension or Text Entity [page 324]) 	<p>A <i>Text Entity</i> must not point to other entities.</p>

Semantic Usage	Keys Required	Mapping Keys When Creating Associations
<p><i>Hierarchy</i></p> <p>See Creating an External Hierarchy [page 330].</p>	<p>You must set exactly one key attribute for a <i>Hierarchy</i>:</p> <ul style="list-style-type: none"> Child attribute of the parent-child hierarchy structure 	<p>A <i>Hierarchy</i> will generally not point to other entities.</p>
<p><i>Hierarchy with Directory</i></p> <p>See Creating a Hierarchy with Directory [page 331].</p>	<p>You must set exactly two key attributes for a <i>Hierarchy with Directory</i>:</p> <ul style="list-style-type: none"> Child attribute of the parent-child hierarchy structure Hierarchy name attribute 	<p>A <i>Hierarchy with Directory</i> must point to:</p> <ul style="list-style-type: none"> A <i>Dimension</i> acting as its directory - The hierarchy name attribute in the (source) hierarchy entity must be mapped to the primary key column in the (target) dimension. Any non-leaf <i>Dimension</i> providing nodes to the hierarchy - The appropriate node type values columns in the (source) hierarchy must be mapped to the key columns in the (target) <i>Dimension</i>.
<p><i>Relational Dataset</i></p>	<p>You may set one or more key attributes for a <i>Relational Dataset</i>, but none are required.</p>	<p>A <i>Relational Dataset</i> can point to any other entity and should generally follow the rules for dimensions.</p>

7.2.2.1 Set a Compound Key

In some cases, key columns aren't enough to ensure the uniqueness of records. Defining a compound key ensures that records are uniquely identified.

Prerequisites

You must meet the following prerequisites to be able to set a compound key:

- set the *Semantic Usage* of your entity as *Dimension*, *Text*, or *External Hierarchy*.
- define at least two columns as key columns.

Context

Setting a single key column might not always be enough to ensure the uniqueness of records, and the combination of two or more key columns may be necessary. In such cases, you may need to set a compound key (also known as composite key).

A compound key is a key that consists of two or more key columns, namely a key column and a representative key. The representative key is the key holding the most specific/granular level of data. Together, these keys uniquely identify rows.

❖ Example

You have two tables, **CostArea** and **CostCenter**:

CostArea	Description
A	Cost area A
B	Cost area B
C	Cost area C





CostCenter	Description
1	Cost center for 1
2	Cost center for 2
3	Cost center for 3

You need to compound these two keys to ensure that records are uniquely identified. Setting a compound key column allows analytical tools to combine the columns **CostArea** and **CostCenter** in order to uniquely represent records in analytical tool via the **CostCenter(Representative Key)**. The compound key doesn't exist as an entity, but more like a semantic instruction for analytical tools. You may notice that the cost centers B2 and C3 don't exist:

CostArea	CostCenter	CostCenter (Representative Key)	Description
A	1	A\1	Cost center 1 for the area A
A	2	A\2	Cost center 2 for the area A
A	3	A\3	Cost center 3 for the area A
B	1	B\1	Cost center 1 for the area B
B	3	B\3	Cost center 3 for the area B
C	1	C\1	Cost center 1 for the area C
C	2	C\2	Cost center 2 for the area C

In an analytical tool, when selecting **CostCenter** as an attribute, the results will automatically be compounded with **CostArea** because **CostCenter** was set as a representative key..

Procedure

1. You can access the *Compound Key* dialog via two entry points:
 - In the *Table Editor*, in the *Attributes* table toolbar, select at least two columns and click  (*Edit Compound Key*).
 - In the *Graphical View* editor and *E/R Model* editor, in the *Properties* panel, in the *Attributes* section, click  (*Edit Compound Key*).
2. [optional] Reorder keys in the *Key Column(s)* section by clicking  (*Move Up*) and  (*Move Down*).
3. Define a representative key by selecting a key in the *Key Column(s)* section and clicking *Add*. The key has been transferred to the *Representative Key* section.
In the *Compound Key* section, you can see the name of the compound key made out of the names of the key columns and representative key composing it.
4. Click *Close* to finish setting the compound key and close the dialog.

Note

Closing the dialog without setting or after removing a representative key deletes the compound key.

7.2.3 Add a Hierarchy to a Dimension

Add a hierarchy to your dimension to support drill-down and drill-up in BI clients.

Context

You can specify the following types of hierarchy:

- Parent-Child - the hierarchy is recursive, may have any number of levels, and is defined by specifying a parent column and a child column within the dimension. For example, a departmental hierarchy could be modeled with the **Parent Department ID** and **Department ID** columns.
- Level-Based - the hierarchy is non-recursive, has a fixed number of levels, and is defined by specifying two or more level columns within the dimension. For example, a time hierarchy could be modeled with the: **Year**, **Quarter**, **Month**, **Week**, and **Day** columns.
- External Hierarchy - the parent-child hierarchy information is contained in a separate entity, which needs to be associated with the dimension (see [Creating an External Hierarchy \[page 330\]](#)).

Note

You can create more than one hierarchy for a dimension, but they must all be of the same type (internal and external Parent-Child hierarchies can be mixed).

The following hierarchy features are not supported:

- Multiple parents
- Unassigned members


- Cycle handling

Procedure

1. Open your dimension and click  (*Hierarchies*) to open the *Hierarchies* dialog.

Note

For tables, the editor for hierarchies is in the menu bar on top of the canvas. For views and entity-relationship models, it is on the properties panel.

2. Click  (*Add*) and then select:

- *Parent-Child Hierarchy* or *Level-Based Hierarchy* - To create a new hierarchy inside the dimension. Complete the following properties as appropriate:

Property	Description
Business Name	Enter a descriptive name to help users identify the object. This name can be changed at any time.
Technical Name	<p>Displays the name used in scripts and code, synchronized by default with the <i>Business Name</i>.</p> <p>To override the default technical name, enter a new one in the field. Technical names can contain only alphanumeric characters and underscores.</p>

Note

Once the object is saved, the technical name can no longer be modified.

Property	Description
Parent Column / Child Column	[parent-child hierarchies] Click + (Add) to add a new parent-child hierarchy. Select the parent and child columns that represent the hierarchy. You can set as many as necessary. The results of these hierarchies can be compounded.

Example

You have a **CostArea** and **CostCenter** attributes with the following pairs of parent-child hierarchies:

ParentCostArea	CostArea
B	C
A	B
NULL	A

In the **CostArea** parent-child hierarchy:

- A is parent to B.
- B is child to A and parent to C.
- C is child to B.

ParentCostCenter	CostCenter
2	3
1	2
NULL	1

In the **CostCenter** parent-child hierarchy:

- 1 is parent to 2.
- 2 is child to 1 and parent to 3.
- 3 is child to 2.

Compounding these two hierarchies will create the following compounded parent-child hierarchy, with the hierarchical structure of the two initial dimensions taken into consideration:

NULL
 --> A/1
 --> B/2
 --> C/3

 --> A/2
 --> B/3

 --> B/1

Property	Description
	--> C/2 --> C/1 -->A/3
Levels	[level-based hierarchies] Click + (<i>Add</i>) to add a level and select the column containing values for the highest level of your hierarchy. Click + (<i>Add</i>) again to add the next level down. Keep adding levels until you reach the lowest level of your hierarchy. For example, in a Products dimension, you may add three levels containing the columns Product Line , Product Category , and Product .

- **External Hierarchy** - To open the External Hierarchy dialog:
 1. Select the external hierarchy and click *OK* to show the mapping editor.
 2. Drag the key column of your dimension and drop it on the child column of the hierarchy to map them.

Note

The child attribute must be a key column.

3. Click *Add* to add the external hierarchy and return to the *Hierarchies* dialog.

Note

Alternatively, you can add an external hierarchy in the *Associations* section of the dimension property sheet. Click **+** (*Create Association*) and select *Hierarchy Association*. Drag and drop the key attribute of your dimension on the child column of the hierarchy to map them.

3. Click *Close* to close the dialog and return to your dimension.

7.2.4 Enable Time-Dependency for a Dimension or Text Entity

Enable time-dependency in your dimension or text entity to define periods of validity for each row and ensure that only the appropriate dimension members with the correct names are displayed for any given time period.

This topic contains the following sections:

- [Specify Time-Dependent Semantic Types in your Dimension or Text Entity \[page 325\]](#)
- [Create a Reference Date Variable in Your Analytic Model \[page 326\]](#)
- [Create a Time-Dependent Parameter in Your Analytical Dataset \(Deprecated\) \[page 326\]](#)

Specify Time-Dependent Semantic Types in your Dimension or Text Entity

To enable time-dependency in a dimension or text entity, you must identify one column containing the date from which the row is valid, and a second column containing the date after which the row is no longer valid:

- Dimensions - The columns identify the start and end dates for the period during which each dimension member is valid.
 - Text entities - The columns identify the start and end dates for the period during which each dimension member name in a given language is valid.
1. Open your dimension or text entity, and ensure that its *Semantic Usage* is correctly set.
 2. Locate the column containing the start date for the validity of dimension members (or texts) and set its *Semantic Type* to *Business Date - From*.
 3. Locate the column containing the end date for the validity of dimension members (or texts) and set its *Semantic Type* to *Business Date - To*.

Note

Both columns must have the same data type from the datetime family. You should avoid overlapping periods of validity, as this may lead to duplicate data in your stories.

4. Ensure that one or other of these dates is selected as a key.

In this example of a time-dependent text entity:

- The column `Validity Start Date` has a semantic type of *Business Date - From* and is set as a key (along with `Product ID` and `Language`).
- The column `Validity End Date` has a semantic type of *Business Date - To*.

Business Name	Technical Name	Data Type	Semantic Type	Label Column
<input checked="" type="checkbox"/> Product ID	Product_ID	Integer64	None	Product Name
<input checked="" type="checkbox"/> Language	Language	String(5000)	Language	
<input checked="" type="checkbox"/> Validity Start Date	Validity_Start_Date	Date	Business Date - From	
<input type="checkbox"/> Validity End Date	Validity_End_Date	Date	Business Date - To	
<input type="checkbox"/> Product Name	Product_Name	String(5000)	Text	

Note

For information about setting keys and mapping them in associations, see [Set Key Columns to Uniquely Identify Records \[page 318\]](#).

5. [optional] By default, the *Business Date - From* and *Business Date - To* are treated inclusively. If you want to exclude one or both, you must edit the CSN code for your entity:

1. Click *Edit Custom CSN Annotations* (see [Edit a Custom CSN Annotation \[page 249\]](#)).
2. Directly after the opening brace following the entity name, add the following code:

```
"@Semantics.interval" : [{
  "lowerBoundaryElement" : "<Start Date Column>",
  "lowerBoundaryIncluded" : true,
  "upperBoundaryElement" : "<End Date Column>",
  "upperBoundaryIncluded" : true
}]
```

3. Change one or both of the `true` values to `false` to exclude the relevant date.

In this example, the `Validity Start Date` is treated inclusively but the `Validity End Date` is treated exclusively:

```
1 {
2   "TD_Product_Texts_View": {
3     "// not editable @EndUserText.label": "TD Product Texts View",
4     "@Semantics.interval": [
5       {
6         "lowerBoundaryElement": "Validity_Start_Date",
7         "lowerBoundaryIncluded": true,
8         "upperBoundaryElement": "Validity_End_Date",
9         "upperBoundaryIncluded": false
10      }
11    ],

```

4. Click *OK* to save your changes.
6. Click *Deploy* to save and deploy your changes.

When an analytic model containing a fact that has an association pointing to the dimension is consumed in SAP Analytics Cloud, the names of dimension members are determined displayed based on the current date.

Create a Reference Date Variable in Your Analytic Model

Any analytic model containing a fact pointing to your dimension via an association will, by default, benefit from its time-dependent data so that the dimension members and their names are displayed based on the current date.

You can, optionally, create a reference date variable in your analytic model to allow SAP Analytics Cloud users to enter a date of their choice and show dimension members based on that date (see [Add a Variable \[page 348\]](#)).

Create a Time-Dependent Parameter in Your Analytical Dataset (Deprecated)

Note

The preferred way to expose data to SAP Analytics Cloud is now to identify your measures in a table or view with a semantic usage of *Fact* and then to use this fact in one or more analytic models, each of which can be consumed by one or more stories (see [Creating an Analytic Model \[page 337\]](#)).

Any analytical dataset pointing to your dimension via an association will, by default, benefit from its time-dependent data so that the dimension members and their names are displayed based on the current date.

You can, optionally, add an input parameter to your analytical dataset to allow SAP Analytics Cloud users to enter a date of their choice and show dimension members based on that date.

1. Create an input parameter (see [Create an Input Parameter \[page 231\]](#)) with data type *Date*.
2. Edit the CSN code to add a semantic type to the input parameter:

1. Click *Edit Custom CSN Annotations*
2. In the `params` section, find your parameter and add the following line after its `type`:

```
"@Semantics.businessDate.at" : true
```

In this example, the input parameter `IP_DATE` has the semantic type *Business Date - At*:

```
21 ▾ "params": {
22 ▾   "IP_DATE": {
23     "@Semantics.businessDate.at": true,
24     "// not editable @EndUserText.label": "IP Date"
25   }
26 },
```

3. Click *OK* to save your changes.
3. Click *Deploy* to save and deploy your changes.

When the analytical dataset is consumed in SAP Analytics Cloud the names of dimension members are displayed based on the specified date.

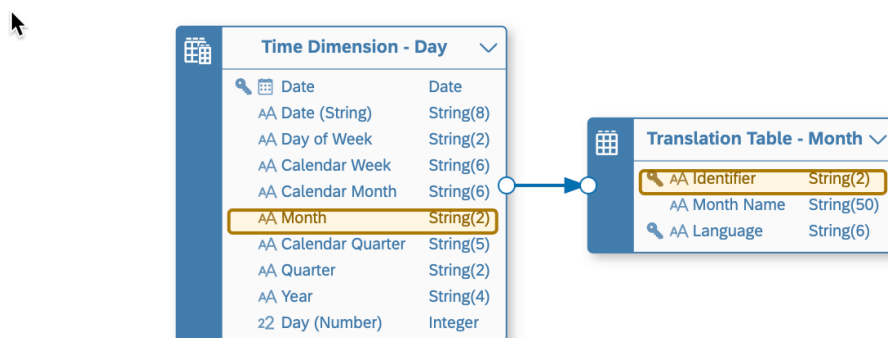
7.3 Create a Text Entity for Attribute Translation

Select a *Semantic Usage* of *Text* to indicate that your entity contains strings with language identifiers to translate text attributes in other entities.

Context

For example:

- The *Time Dimension - Day* dimension generated by SAP Datasphere contains a *Month* column that stores as numbers (with 01 representing January, 02 February, 03 for March, and so on).
- The text entity *Translation Table - Month* contains the names of these months in English, French, German, and Spanish and can provide the name in the appropriate language.
- A text association is drawn from the *Month* column to the *Identifier* column in the *Translation Table - Month* text entity.



Procedure

1. In the table editor or view editor output node side panel, set the *Semantic Usage* property to *Text*.
2. Specify your attributes (see [Specify Attributes \[page 315\]](#)).

You must specify at least the following attributes:

Content	Key	Semantic Type
Identifier Specifies the dimension member for which the text is provided. Example: 01	Yes	None
Language Identifier Specifies the language in which the text is provided and must be used with the appropriate <code>String(<n>)</code> data type. Example: fr	Yes	<i>Language</i>
<div style="background-color: #f0f0f0; padding: 10px; border: 1px solid #ccc;"> <p>Note</p> <p>The following identifier formats are supported:</p> <ul style="list-style-type: none"> • <code>F - String(1)</code> • <code>fr - String(2)</code> • <code>FR - String(2)</code> • <code>fr-FR - String(5)</code> <p>For the list of data access languages supported by SAP Analytics Cloud, see Supported Data Access Languages.</p> </div>		
Validity Start Date [optional] For time-dependent text entities, specifies the date from which the text is valid (see Enable Time-Dependency for a Dimension or Text Entity [page 324]). Example: 01/01/2022	Yes	<i>Business Date - From</i>
Validity End Date [optional] For time-dependent text entities, specifies the date until which the text is valid. Example: 31/12/2022	No	<i>Business Date - To</i>




Content	Key	Semantic Type
Text	No	<i>Text</i>
Provides the text in the appropriate language.		
Example: January		





3. Set attributes as keys to indicate that the data they contain can uniquely identify records.

Note

You must set two or three key attributes for a *Text Entity*:

- Identifier
- Language Identifier
- Validity Start Date - if time-dependency is required (see [Enable Time-Dependency for a Dimension or Text Entity \[page 324\]](#))

To set an attribute as a key column, select the checkbox in the *Key* column or hover over the attribute in the side panel and click  (*Menu*)  *Set as Key* .



4. Click  (*Save*)  *Save*  to save your entity or click  (*Deploy*) to save and deploy it immediately. For more information, see [Saving and Deploying Objects \[page 39\]](#).


7.3.1 Create a Text Association


Context

You need to link a text dependent dimension to a text entity thanks to a text association.

Procedure





1. Select your text dependent dimension to show its side panel. In the *Attributes* section, hover over an attribute and click  (*More*) > *Add Text Association*. The *Select Object* dialog opens and text entities available for association are listed.
Find available objects by entering the object's name in the search bar or click  (*Show filters*) and filter by *Semantic Usage* or other criteria.
2. Select a target entity and click *OK*.
3. Specify the text association mapping of attributes in the *Mappings* section:
 - A default mapping is automatically created by matching a selected element to the first key element whose *Semantic Type* is *None*.

- To delete a mapping, select the link and then click  (*Delete*).
- To manually map columns, drag a column from the left list and drop it onto a column in the right list.
- You can filter the *Mappings* section to show only all, mapped, or unmapped pairs of columns.
- You can filter or sort the left or right column lists independently

The text dependent dimension is associated to the *Text* entity's attribute. The icon  signals which attributes and associations are related to the dimension.

Note

You can also create a *Text Association* by going to the dimension's *Associations* section and clicking [→ \(Create Association\) > Text Association](#).

4. Click  (*Save*)   *Save* to save your entity or click  (*Deploy*) to save and deploy it immediately. For more information, see [Saving and Deploying Objects \[page 39\]](#).

7.4 Creating an External Hierarchy

Select a *Semantic Usage* of *Hierarchy* to indicate that your entity contains parent-child relationships for members in a dimension.

Procedure

1. Open the table or view containing the hierarchy information and set the following properties:

Property	Description
Business Name	Enter a descriptive name to help users identify the object. This name can be changed at any time.
Technical Name	Displays the name used in scripts and code, synchronized by default with the <i>Business Name</i> .
Semantic Usage	Choose <i>Hierarchy</i> .
Parent Column / Child Column	Select the columns containing the parent and child information for your hierarchy.

Note




The parent column and the child column must be of the same data type.





2. Set attributes as keys to indicate that the data they contain can uniquely identify records.

Note

You must set exactly one key attribute for a *Hierarchy*:

- Child attribute of the parent-child hierarchy structure



To set an attribute as a key column, select the checkbox in the *Key* column or hover over the attribute in the side panel and click  (Menu)  *Set as Key* .

3. Click  (Save)  *Save*  to save your entity or click  (Deploy) to save and deploy it immediately.
For more information, see [Saving and Deploying Objects \[page 39\]](#).

7.5 Creating a Hierarchy with Directory

Select a *Semantic Usage* of *Hierarchy with Directory* to indicate that your entity contains one or more parent-child hierarchies and has an association to a directory dimension containing a list of the hierarchies.

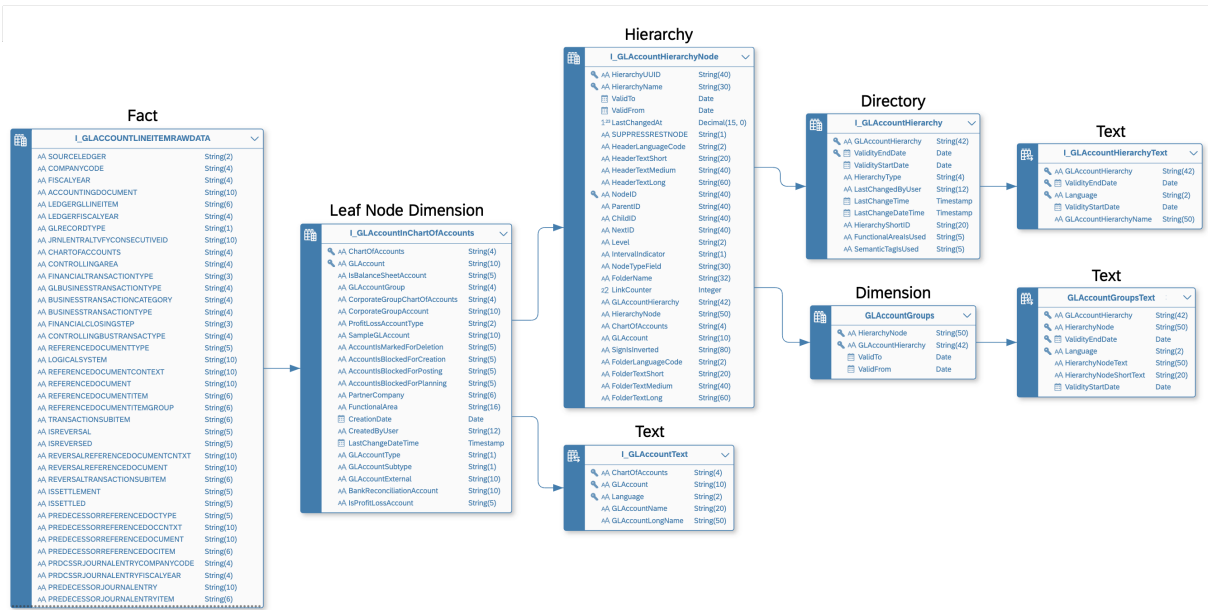
These types of hierarchy entities can include nodes from multiple dimensions (for example, country, cost center group, and cost center) and are commonly imported from SAP S/4HANA Cloud and SAP BW (including SAP BW Bridge) systems.

See also the [Hierarchy with Directory Samples](#)  Community Content page for SAP S/4HANA samples, and the blog series beginning with [An Introduction to Hierarchy with Directory in SAP Datasphere](#)  for more information and walkthroughs.

This topic contains the following sections:

- [Example: SAP S/4HANA Cloud General Ledger Account Hierarchy \[page 331\]](#)
- [Prepare the Leaf Node and Other Hierarchy Node Dimensions and Text Entities \[page 333\]](#)
- [Prepare the Hierarchy Directory Entity and Text Entities \[page 333\]](#)
- [Prepare the Hierarchy Entity \[page 334\]](#)
- [Use a Hierarchy in an Analytic Model \[page 337\]](#)
- [Use a Hierarchy in an SAP Analytics Cloud Story \[page 337\]](#)

Example: SAP S/4HANA Cloud General Ledger Account Hierarchy



Our example is created from entities imported from an SAP S/4HANA Cloud system and defines hierarchies of general ledger accounts and categories:

- I_GLACCOUNTLINEITEMRAWDATA (semantic usage: *Fact*) - Contains transactional data and has an association to:
 - I_GLAccountInChartOfAccounts (semantic usage: *Dimension*) - A dimension containing general ledger accounts, which are organized into hierarchies. It has associations to:
 - I_GLAccountText (semantic usage: *Text*) - A text entity containing translations of the general ledger account names.
 - I_GLAccountHierarchyNode (semantic usage: *Hierarchy with Directory*) - A hierarchy entity for defining parent-child relationships between general ledger accounts and categories. It has associations to:
 - I_GLAccountHierarchy (semantic usage: *Dimension*) - The directory for GLAccountHierarchyNode, which contains the names of the various hierarchies. It has an association to:
 - I_GLAccountHierarchyText (semantic usage: *Text*) - A text entity containing translations of the hierarchy names.
 - GLAccountGroups (semantic usage: *Dimension*) - A dimension containing grouping nodes that are used in the general ledger account hierarchies. It has an association to:
 - I_GLAccountGroupText (semantic usage: *Text*) - A text entity containing translations of the general ledger account group names.

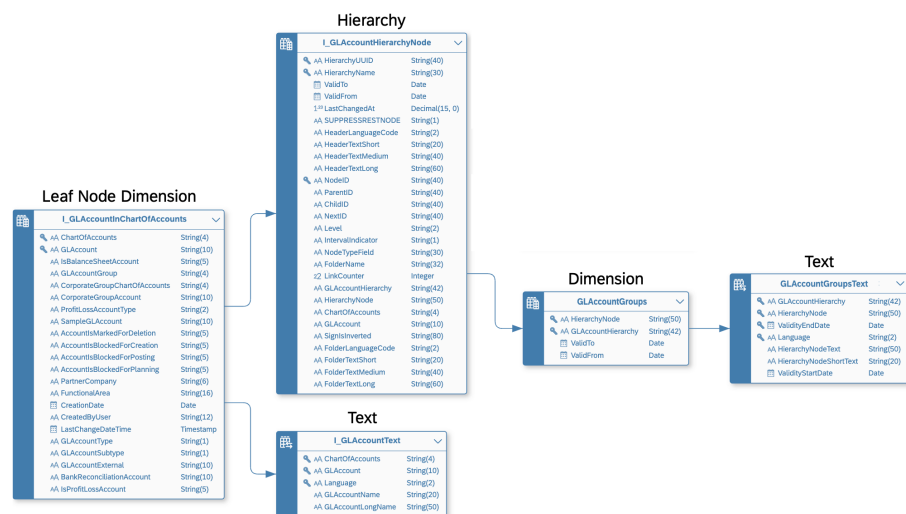
Note

In SAP S/4HANA Cloud, the GL account hierarchy grouping nodes are stored directly in the I_GLAccountHierarchyNode entity, but SAP Datasphere requires all hierarchy node types to be stored in separate dimensions. Therefore, to complete the valid definition of the hierarchy and its supporting entities we extracted the unique values for the groups into this separate view.

Prepare the Leaf Node and Other Hierarchy Node Dimensions and Text Entities

You must identify a leaf node dimension, which is the dimension that contains the lowest nodes (leaf nodes) in the hierarchy, along with any other dimensions that contribute nodes to the hierarchy. Each dimension will generally be associated with one or more text entities to provide translation of the node names.

In our example, `GLAccountInChartOfAccounts` is the leaf node dimension and `GLAccountGroups` is another dimension providing non-leaf nodes, and each has an association to an appropriate text entity:

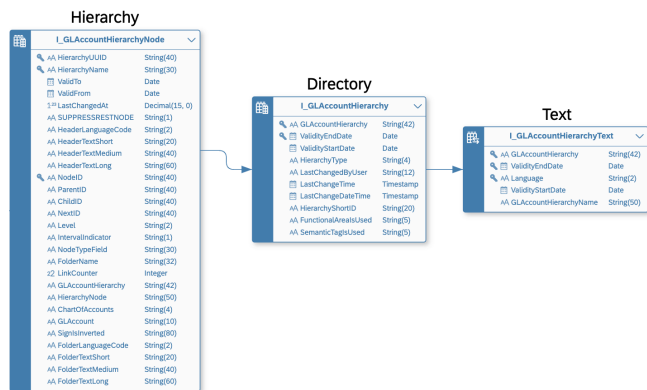


1. Open your leaf node dimension and set the semantic usage to *Dimension*. Your leaf node dimension must have a key and will contain records of members that are related in one or more parent-child hierarchies. For more information about dimensions, see [Creating a Dimension \[page 314\]](#).
2. In the *Associations* section, click **New > Hierarchy Association**, select your hierarchy entity and complete the mapping.
3. In the *Associations* section, click **New > Text Association**, select the appropriate text entity and complete the mapping. For more information about text entities, see [Create a Text Entity for Attribute Translation \[page 327\]](#).
4. [optional] If your hierarchy contains other node types, ensure that each dimension containing one of these node types is available in your space, has its semantic usage set to *Dimension*, and has text associations to appropriate text entities. In our example, `GLAccountGroups` is a dimension containing GL Account grouping nodes, which is a non-leaf node type in our hierarchy.

Prepare the Hierarchy Directory Entity and Text Entities

You must identify a hierarchy directory entity that contains hierarchy identifiers and names. A directory will generally be associated with one or more text entities to provide translation of the hierarchy names.

In our example, `I_GLAccountHierarchy` is the hierarchy directory dimension:

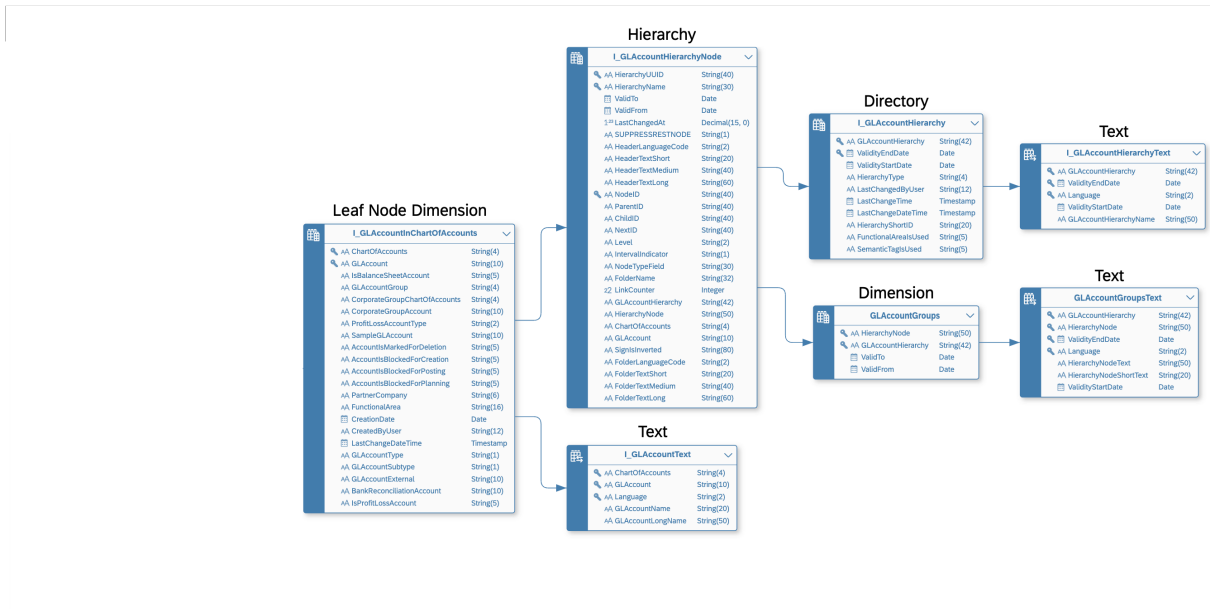


1. Open your hierarchy directory entity and set the semantic usage to *Dimension*.
2. Ensure that it contains, as a minimum, the following columns:
 - Hierarchy Identifier - (key) containing the unique identifier of a hierarchy
 - Hierarchy Name - Containing the human-readable name of a hierarchy.
A text entity may be used to provide translations of the hierarchy name
 - [optional] Time dependency-columns to control the availability of hierarchies based on the key date used in the eventual chart (see [Enable Time-Dependency for a Dimension or Text Entity \[page 324\]](#)).
3. In the *Associations* section, click **New** > *Text Association*, select the appropriate text entity and complete the mapping.
For more information about text entities, see [Create a Text Entity for Attribute Translation \[page 327\]](#).

Prepare the Hierarchy Entity

You must identify a hierarchy entity that contains parent-child hierarchy data, along with hierarchy identifiers, node types, and unique identifier columns for each type of node.

In our example `I_GLAccountHierarchyNode` is the hierarchy node:



1. Open the table or view containing the hierarchy information and set the *Semantic Usage* property to *Hierarchy with Directory*.
2. Ensure the following columns are marked as keys:
 - The column containing the identifiers of child nodes in the parent-child relationship. In our example, this is the `NodeID` column.
 - The column containing the identifier of the hierarchy to which the parent-child relationship belongs. In our example, this is the `HierarchyName` column.
3. Create an association from the hierarchy entity to the directory entity, and map the column you selected as the *Hierarchy Name Column* in the hierarchy entity to the primary key column in the directory entity. In our example, the `HierarchyName` column in the hierarchy entity `I_GLAccountHierarchyNode`, is mapped to the primary key column in the directory entity, `I_GLAccountHierarchy`.
4. [if multiple types of hierarchy nodes are specified] Create an association from the hierarchy entity to each (non-leaf) dimension containing nodes that are used in the hierarchies, and map the appropriate node type values columns identified in the hierarchy entity to the key columns in the dimensions. In our example, the columns, `GLAccountHierarchy` and `HierarchyNode` in the hierarchy entity `I_GLAccountHierarchyNode`, are mapped to the key columns in the `GLAccountGroups` dimension.
5. [optional] If time-dependency columns are available, add the appropriate semantic types (see [Enable Time-Dependency for a Dimension or Text Entity \[page 324\]](#)).
6. Click the *Hierarchy with Directory Settings* button, complete the following properties in the dialog, and then click *OK*:

Property	Description
Parent	Select the column containing the identifiers of parent nodes in the parent-child relationship. In our example, the <code>ParentID</code> column is selected.

Property	Description
Child	<p>Select the column containing the identifiers of child nodes in the parent-child relationship.</p> <p>This column must be marked as a key.</p> <p>In our example, the <code>NodeID</code> column is selected.</p>
Hierarchy Name Column	<p>Select the column containing the identifier of the hierarchy to which the parent-child relationship belongs.</p> <p>This column must be marked as a key.</p> <p>In our example, the <code>HierarchyName</code> column is selected.</p>
Hierarchy Directory Entity	<p>Displays the name of the hierarchy directory entity, once the <i>Hierarchy Name Column</i> property is set and an association is created to point from the hierarchy to the directory entity.</p>
Node Type Column	<p>Select the column containing the type of the child node in the parent-child relationship.</p> <p>In our example, the <code>NodeTypeField</code> column is selected.</p>
Node Type Values	<p>Create one entry for each value that appears in the node type column, and complete the properties as follows:</p> <ul style="list-style-type: none"> <p><i>Node Type Value</i> - Enter the value that is used to identify the type of the node in the <i>Node Type Column</i>.</p> <p>In our example:</p> <ul style="list-style-type: none"> <code>GLAccount</code> is used to identify that the node is from the <code>GLAccountInChartOfAccounts</code> dimension. <code>HierarchyNode</code> is used to identify that the node is from the <code>GLAccountGroups</code> dimension. <p><i>Set as Leaf</i> - Select this option if this type of node is at the lowest level of the hierarchy.</p> <div style="background-color: #e0e0e0; padding: 10px; margin: 10px 0;"> <p>Note</p> <p>Only one node type can be identified as the leaf, and all the hierarchies defined in the entity must have the same leaf type.</p> </div> <p>In our example, <code>GLAccount</code> is the leaf node type.</p> <ul style="list-style-type: none"> <p><i>Column 1</i> - Select the column that contains the identifiers of nodes of this type. If more than one column is used, click the <i>Add Column</i> button to add and specify each required column.</p> <p>All of the columns you identify must be mapped in the association between the hierarchy and the dimension.</p> <p>In our example:</p> <ul style="list-style-type: none"> Two columns, <code>ChartOfAccounts</code> and <code>GLAccount</code> identify nodes from the <code>GLAccountInChartOfAccounts</code> dimension. Two columns, <code>GLAccountHierarchy</code> and <code>HierarchyNode</code> identify nodes from the <code>GLAccountGroups</code> dimension.

Use a Hierarchy in an Analytic Model

If your leaf dimension is included in an analytic model, you can enable any of your defined hierarchies in the analytic preview:

1. Open your analytic model and click *Preview* to open the analytic preview.
2. In the *Available Objects* panel, enable your leaf node column as a row.
No hierarchy is enabled by default.
3. In the *Builder* panel *Rows* section, hover over your leaf node column and then click ► *More* ► *Hierarchy* ► *Select Hierarchy* ▾.
4. In the *Select Hierarchy* dialog, select your hierarchy and click *OK*
The hierarchy is shown collapsed and you can drill down into it as appropriate.

For more information about working with the analytic preview, see [Using the Data Preview \[page 350\]](#).

Use a Hierarchy in an SAP Analytics Cloud Story

Hierarchies defined in a hierarchy with directory are supported only in stories created using the Optimized Design Experience (see [Create a New Story \(Optimized Story Experience\)](#) in the *SAP Analytics Cloud* documentation).

Note

Hierarchies defined in a hierarchy with directory cannot be used in Classic Design Experience stories.

For information about using hierarchies in SAP Analytics Cloud, see [Work with Hierarchies in a Chart](#).

7.6 Creating an Analytic Model

Create an analytic model as a basis for consumption in SAP Analytics Cloud.

Context

Analytic models are the analytical foundation for making data ready for consumption in SAP Analytics Cloud. They allow you to create and define multi-dimensional models to provide data for analytical purposes to answer different business questions. Pre-defined measures, hierarchies, filters, parameters, and associations provide flexible and simple navigation through the underlying data.

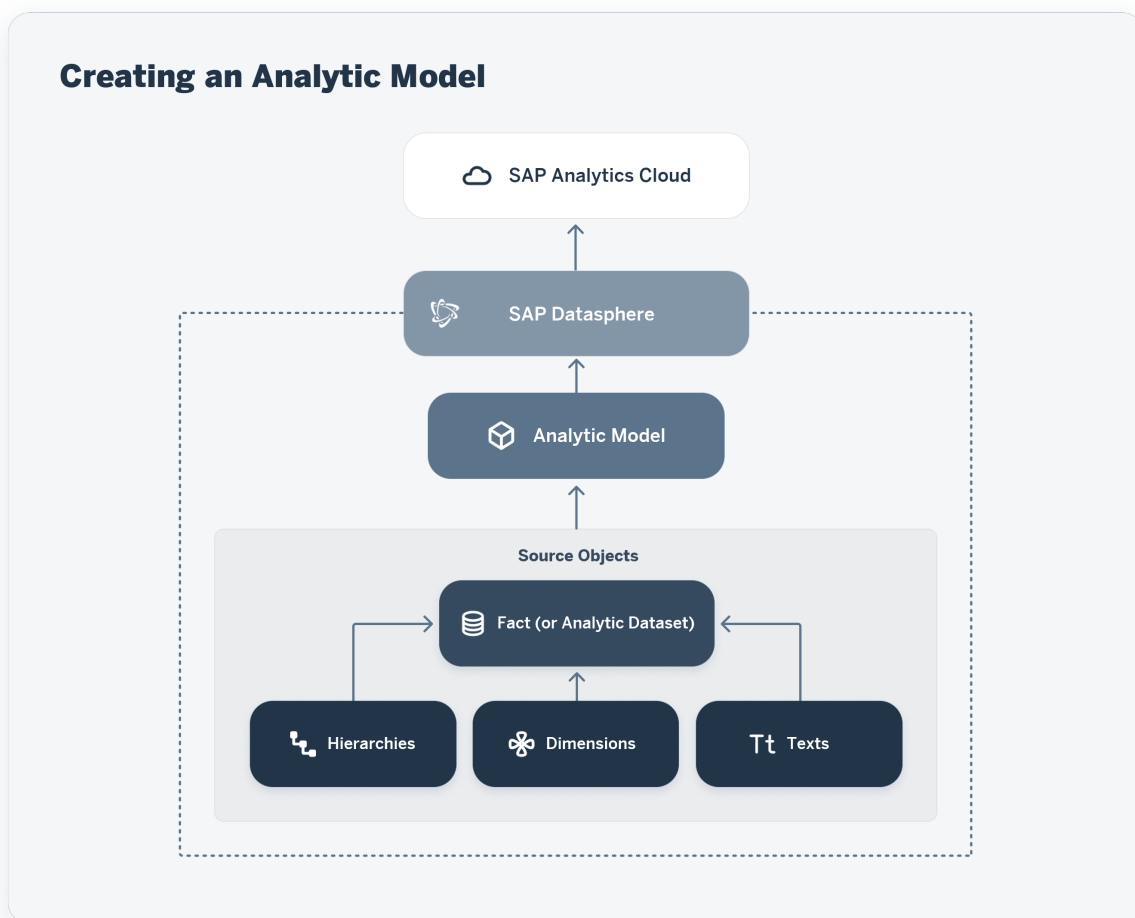
The sources for analytic models are facts (or analytical datasets), which can contain dimensions, texts, and hierarchies.

Note

Some terms and concepts are used differently in analytic models to align more closely with the terminology in SAP Analytics Cloud stories:

- "Input parameters" in facts are called "variables" in analytic models.
- "Attributes" in facts are called "dimensions" in analytic models.

This graphic shows the role of the analytic model within SAP Datasphere:



Procedure

1. From the side navigation, choose *Data Builder*, select a space if necessary, and choose *New Analytic Model* to open the editor.
2. Add a source. For more information, see [Add a Source \[page 340\]](#).
3. Click on your fact source on the canvas to select or deselect any measures, associated dimensions, or attributes in the properties panel on the right. For more information, see [Add a Dimension \[page 341\]](#).

Note

Attributes of type `LargeString` are not consumable in SAP Analytics Cloud.

4. To edit the properties of the analytic model: Click on the background of the canvas to show the analytic model's properties in the side panel. You can make the following changes here:
 1. Enter a descriptive *Name* to help users identify the object. This name can be changed at any time.
 2. Enter a new *Technical Name*. Technical names can contain only alphanumeric characters and underscores.

Note

Once the object is saved, the technical name can no longer be modified.

3. Optional: Select a package. For more information, see [Packages \[page 20\]](#).
4. You can add a measure. For more information, see [Add Measures \[page 343\]](#).
5. You can add a variable. For more information, see [Add a Variable \[page 348\]](#).
5. To edit the properties of a the fact source: Click on the fact source on the canvas to show the its properties in the side panel. You can make the following changes here:
 1. You can change the alias.
 2. You can add or deselect measures.
 3. You can add or deselect associated dimensions.
 4. You can add or deselect attributes.
6. To edit the properties of a dimension: Click on the dimension on the canvas to show its properties in the side panel. You can make the following changes here:
 1. You can change the alias.
 2. You can add or deselect associated dimensions.
 3. You can add or deselect attributes.
7. When you click on a dimension or the fact source on the canvas, you can change the alias of this item. The alias is the name that is shown in the story in SAP Analytics Cloud.

Example

In many cases you need to have a more specific name for a field than the source provides. For example, the field name is *Costcenter*, but you want to specify if it is to *Receiving Costcenter* or *Sending Costcenter*.

8. You can choose [Export](#) > [Export to CSN/JSON File](#) to export the definition of your analytic models to a CSN/JSON file. For more information, see [Exporting Objects to a CSN/JSON File \[page 50\]](#).
9. You can choose [Export](#) > [Export Information for Support](#) to export information from the runtime CSN for support to analyze it.
10. You can choose [Preview](#) to check if the data looks like expected. Here you have two options: there is the simple preview which is available via the context menu in the editor at the analytic model. For more information, see [Viewing or Previewing Data in Data Builder Objects \[page 297\]](#). And then there is the analytical preview in which you can drill down by rows and columns. For more information, see [Using the Data Preview \[page 350\]](#).

Note

When you change an analytic model for which a story has been defined, and you deploy it again, you need to open the story in SAP Analytics Cloud and save it again.

7.6.1 Create an Analytic Model Directly From a View or Table

You can create one or more analytic models directly from views or tables with a *Semantic Usage* of *Fact* (or *Analytical Dataset*).

Procedure

1. Open the appropriate editor and ensure that your table or view has a *Semantic Usage* of *Fact* (or *Analytical Dataset*).

Analytic models cannot be created from entities with other semantic usage.

2. Click *Create Analytic Model* under the *Semantic Usage* property.

The analytic model editor opens directly with the view or table added as a source, with all its properties included.

3. [optional] Remove any unwanted measures or dimensions and edit the analytic model in any appropriate manner (see [Creating an Analytic Model \[page 337\]](#)).
4. Save and deploy your analytic model to expose it for consumption.

Note

You can create as many analytic models as necessary from each view or table.

7.6.2 Add a Source

Drag a table or view from the *Repository* panel and drop it on the canvas.

Context

As a source for your analytic model, you need an object of type fact (or analytical dataset).

The data type and semantic type information is inherited from the fact (or analytical dataset), as well as the standard aggregation behaviour for a measure. For fields with semantic type "Amount with Currency" and "Quantity with Unit" you only need to add the measure in the analytic model, the quantity or unit are added automatically.

Procedure

1. Browse or search for the object you want to add. The repository shows only the objects which can be used in an analytic model: facts (and analytical datasets).
2. Drag your source from the Repository and drop it onto the canvas.
3. Select the properties you want to copy from the source. You can still take over properties into your analytic model later.
4. If the source is a view containing one or more input parameters, you must decide how each input parameter will be processed:
 - **Map To:** Map the source input parameter to a variable in your analytic model. Users of this view will need to provide a value for the variable.
 - **Set Value:** Enter a value to resolve the variable. The variable is resolved and users of this analytic model will no longer need to provide a value for it. If a value is already set by default, you can either set it or edit it by clicking the value and selecting another one in the list of available values of the **Select Member** dialog. The **Select Member** dialog is available only if a predefined default value has been defined in the input parameter.

7.6.3 Add a Dimension

You can add dimensions to your analytic model.

Context

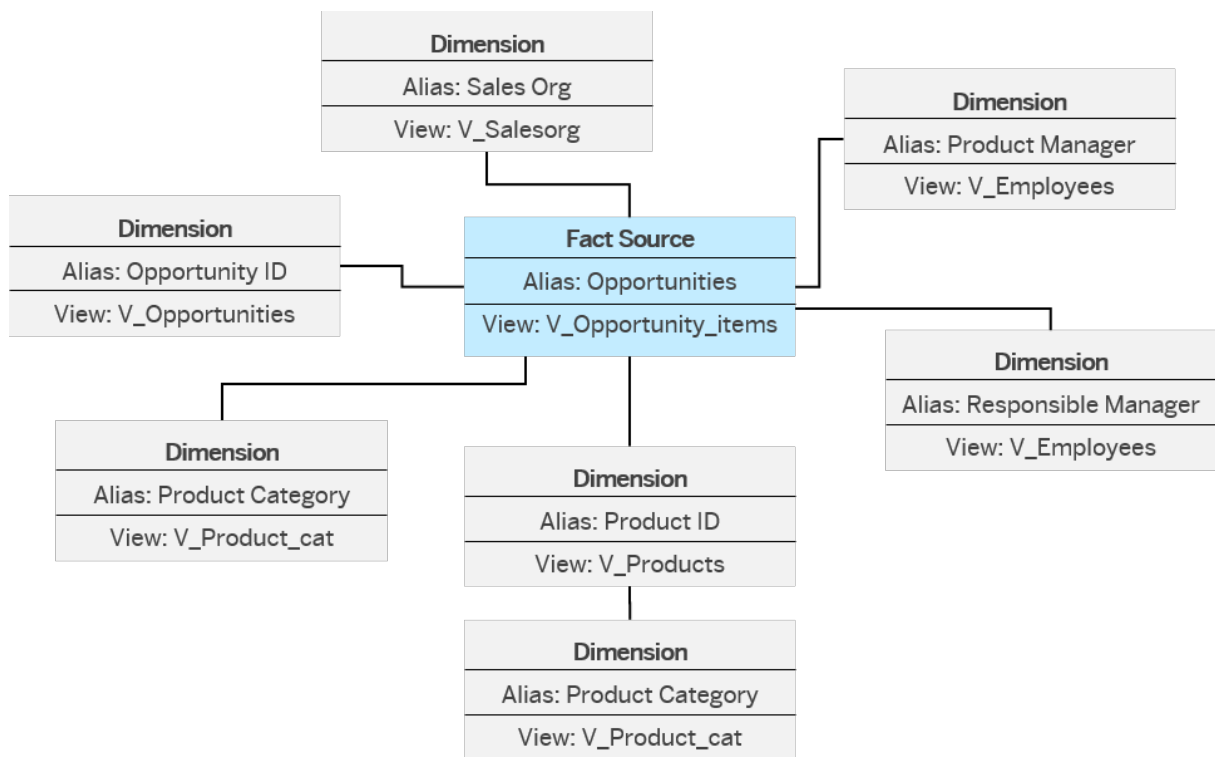
The source of your analytic model can contain associations to dimensions, texts, and hierarchies. Dimensions contain attributes that can be used to analyze and categorize measures defined in other entities. Attributes are used as dimensions in the analytic model. When you take over associated dimensions from your source, the object type is also taken over. Thus, you can easily build a multidimensional model in a star schema with the fact source in the center and the dimensions as direct associations.

These associations are supported:

- Associations to dimensions (associations to a table or a view with semantic usage Dimension)
- Associations to a text table (associations to a table or a view with semantic usage Text). They are automatically used by the analytic model if a dimension has a text association.
- Association from a dimension to a hierarchy table. They are automatically used by the analytic model if a dimension has a hierarchy association.

You can also add dimensions of dimensions.

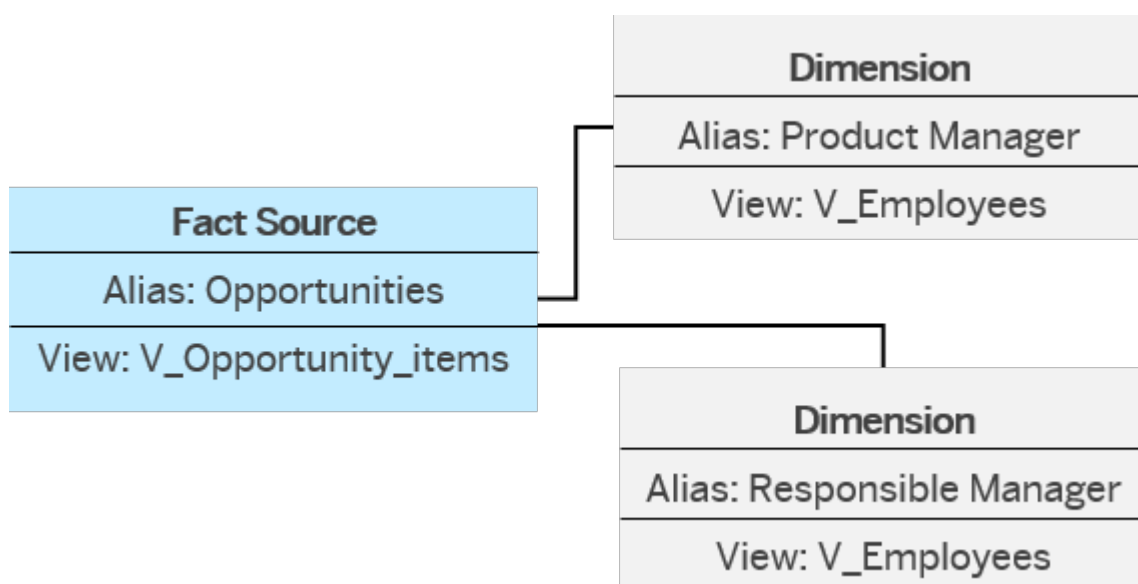
The graphic shows the star schema of the analytic model: the fact source is in the center and the dimensions are surrounding it. The dimension *Product ID* also has a dimension: *Product Category*.



Here, you can see that the same dimension can be included more than once. To be able to distinguish between the different roles of the dimension in the data preview and the story in SAP Analytics Cloud, you can give the dimensions different aliases. The alias enables you to give a dimension a more specific name for a field than the source provides.

❁ Example

The dimension `V_Employees` is included twice, but can be distinguished by the aliases *Responsible Manager* and *Product Manager*.



Procedure

There are different ways to add dimensions:

1. When you select your source, you can select the dimensions that you want to copy from the source.
2. When you click on your fact source on the canvas, you can select associated dimensions and attributes in the properties panel on the right. The attributes are also treated as dimensions in the analytic model.
3. When you click on a dimension on the canvas, you can select associated dimensions of the dimension in the properties panel on the right.
4. To edit the properties of a dimension: Click on the dimension of the canvas to show its properties in the side panel. You can make the following changes here:
 - You can change the alias of this item in the properties panel. The alias is the name that is shown in the story in SAP Analytics Cloud.
 - You can add or deselect associated dimensions.
 - You can add or deselect attributes.

7.6.4 Add Measures

Add measures to your analytic model.

Context

Measures contain numerical values that you want to analyze.

There are different types of measures to choose from:

- **calculated measure:** A calculated measure references other measures and allows the combination of measures with elementary arithmetic (operations of addition, subtraction, multiplication, and division), and also more complex functions like maximum/minimum, IF (as function), mathematical functions like round, and conversion functions like convert to decfloat.
- **restricted measure:** A restricted measure refers to another measure and allows restrictions on available dimensions.
- **count distinct measure:** A count distinct measure counts unique occurrences of attributes (e.g. in case multiple rows contain the country "Germany", it is counted only once).
- **Currency conversion measure:** With a currency conversion measure, you can convert currencies.

Procedure

1. To add fact source measures, click on your fact source on the canvas to select measures in the properties panel on the right.
2. You can create new measures by clicking the background of the canvas and choosing **+** *Add Measure* in the properties panel. Choose the type of measure.

3. For calculated measures, enter your expression in the formula editor.

The expression field contains the formula to calculate the measure.

Example

To calculate the price, the expression could be `Amount / Quantity`.

4. For restricted measures, proceed as described here: [Create a Restricted Measure \[page 346\]](#).
5. For currency conversion measures, proceed as described here: [Create a Currency Conversion Measure \[page 344\]](#).
6. For count distinct measures, select the dimensions for which this measure shall be applied.
7. For calculated measures, restricted measures, and fact source measures, you can define an exception aggregation. The exception aggregation determines how a measure is aggregated with regard to one or more dimensions. A dimension is needed for exception aggregation in order to define the granularity with which the aggregation rule is applied. For more information, see [Aggregation and Exception Aggregation \[page 346\]](#).
8. Decide whether your measure should be an *Auxiliary Measure*. An auxiliary measure can be used for further calculation but it will be hidden in the story in SAP Analytics Cloud.

7.6.4.1 Create a Currency Conversion Measure


With a currency conversion measure, you can convert currencies.

Prerequisites

The tables and view containing currencies and exchange rates `SAP.CURRENCY.TABLE.TCUR*` are available in the current space.

You can create these tables in the *Data Builder* (see [Enabling Currency Conversion with TCUR* Tables and Views \[page 52\]](#)).

Procedure

1. You are in the editor of your analytic model. You can create a new currency conversion measure by clicking the background of the canvas and choosing  *Add Measure* in the properties panel. Choose *Currency Conversion Measure*.
2. Select the source measure. The source measure has to be a fact source and has to refer to a measure in the underlying fact with semantic type *Amount with Currency*. The measure in the fact has to have an assigned currency field.
3. Define the target currency.

The target currency can be specified as:

- a constant value
 - a reference to a dimension that is mapped to an attribute of the fact with semantic type *Currency Code*.
 - a variable. For more information, see [Use Variables for Currency Conversion \[page 345\]](#).
4. Define the reference date.
- The reference date can be:
- the current date
 - a constant value, with a fixed date
 - a reference to a dimension with data type *Date*
 - a variable. For more information, see [Use Variables for Currency Conversion \[page 345\]](#).
5. Select the exchange rate type to be used on the conversion.
- The target currency can be specified as:
- a constant value
 - a reference to a dimension that is mapped to an attribute of the fact
 - a variable. For more information, see [Use Variables for Currency Conversion \[page 345\]](#).
6. Select the client. The currency conversion tables have a client field, which you can enter here.
7. Select the error handling method. This defines how the system reacts when a value cannot be converted. Possible values are:
- Set to null
 - Fail on error
 - Keep unconverted

7.6.4.1.1 Use Variables for Currency Conversion

For currency conversion measures, you can define variables for the target currency, the reference date, and the exchange rate type.

Procedure

1. In the definition of the currency conversion measure, when defining the target currency, the reference date, and the exchange rate type, you can select *Variable* as type.
2. You can then select an existing variable or create a new variable.
3. To create a new variable, select *Create new Source Variable*. The new variable is added to the variable section of the currency conversion measure.
4. Edit the properties of the new variable: choose how the variable should be filled:
 - *Manual Input*
 - *Derive Value*
5. If you want to derive the value, you need to
 1. Define a lookup entity (a view)

2. Select the column from which the value is to be derived.
3. Map the parameter of the lookup entity. If it has no input parameter, you can set a constant value or create a variable.

7.6.4.2 Create a Restricted Measure

With a restricted measure, you can restrict available dimensions.

Procedure

1. You are in the editor of your analytic model. You can create a new currency conversion measure by clicking the background of the canvas and choosing **+** *Add Measure* in the properties panel. Choose *Restricted Measure*.
2. Select the source measure.
3. When the source measure is a fact source measure the standard aggregation can be overridden by changing the *Aggregation Type*.
4. Enter your expression in the formula editor. An atomic expression usually has the format `<dimension> <operator> <value>` (e.g. `Country = 'Germany'`).
5. You can define an exception aggregation. The exception aggregation determines how a measure is aggregated with regard to one or more dimensions. A dimension is needed for exception aggregation in order to define the granularity with which the aggregation rule is applied.

Exception aggregation is only supported for restricted measures if the source measure is a fact source measure. You can use the exception aggregation to calculate the stock for a given time. You can also define no restrictions for the restricted measure, and just use it for the exception aggregation.

For more information, see [Aggregation and Exception Aggregation \[page 346\]](#).

6. You can enable constant selection for the restricted measure. You can choose to set it for all dimensions or just for selected dimensions. A measure with constant selection will not be impacted by filters or drill-downs on the constant dimensions.

Enabling constant selection is useful for comparing a single value with several different values. For example, you could create a restricted measure for sales in 2022, and then compare sales in 2022 with sales for all other years in the same table.

See also the blog post [Restricted Measures with Constant Selection in SAP Analytics Cloud](#) (published February 2, 2021) for examples.

7.6.4.3 Aggregation and Exception Aggregation

Aggregation is a process to summarize measures, for example, by grouping values of multiple rows into a single result.

The standard aggregation is used to aggregate with respect to any dimension not in query drill-down. Exception aggregation is always performed in addition to standard aggregation. It defines the exception:

It is used to aggregate with respect to the defined dimensions for exception aggregation. You can use exception aggregation if warehouse stock cannot be totaled up at the time, or if counters count the number of dimensions for a specific dimension.

A measure is calculated in the following order: first the standard aggregation, then the formula, and then the exception aggregation.

Note

Using many dimensions in an exception aggregation will have an impact on the performance. This is because more data must be processed after the main aggregation, which uses the SAP HANA build-in engines and data structures. While these engines can use optimized data storage and processes, the exception aggregation is more generic and slower than the low level aggregation.

Examples

Calculate Stock

You want to calculate the stock at the beginning of a month and the end of the month. You can do that by creating two measures: one with different exception aggregation behavior First and one with exception aggregation behavior Last.

With exception aggregation First, only the value of the first day in the month is taken and with exception aggregation Last, only the value of the last day in the month is taken.

Count Number of Customers Within a Year

Count Number of Customers per Year (2022)	NetAmount 2022	Number of Customers in 2022	Average NetAmount per Customer
Colorado	55,871.75 \$	35	1,596.34 \$
Conneticut	7,924.00 \$	5	1,584.80 \$
Delaware	35,217.10 \$	7	5,031.01 \$
Florida	10,634.35 \$	14	759.60 \$
Florida	47,895.40 \$	25	1,915.82 \$
Florida	582.85 \$	1	582.85 \$
Maryland	1,521.15 \$	4	380.29 \$
Pennsylvania	53,748.20 \$	23	2,336.88 \$

The above query shows the companies, revenue of 2022 in combination with the number of customers within this year. The special aspect here is, that the "counting of the customers" does not need to have the CUSTOMER dimension itself within the current result set. The aggregation is done over exception aggregation and over the exception aggregation dimension CUSTOMER for the counting. This allows you to leave the dimension CUSTOMER out of the result set. Based on this you can have further analysis, e.g. with a formula which calculates the average revenue per customer in the year.

7.6.5 Add a Variable

Add variables to your analytic model.

When you add a variable, the user will be prompted to enter a value in the data preview or when the analytic model is consumed in a story in SAP Analytics Cloud.

There are different types of variables to choose from:

- **source variable:** A source variable is used in a fact source to map it to input parameters of the fact.
- **restricted measure variable:** A restricted measure variable is used in the filter condition of a restricted measure (e.g. “Revenue of selected country” with **country = <user input>**). It can be applied only in restricted measures, and they do not filter the entire data.
- **filter variable:** A filter variable refers to an attribute. When a story is opened, the variable dialog shows a filter for the attribute.
- **reference date variable:** When the analytic model has associations to a time-dependent dimension or text table, you can define a reference date variable. With a reference data variable, you can filter the data using a specific date. This allows SAP Analytics Cloud users to enter a date of their choice for their story and show dimension members based on that date. For more information on time-dependent dimensions, see [Enable Time-Dependency for a Dimension or Text Entity \[page 324\]](#).

Create a Source Variable

1. You can create a new source variable by clicking the background of the canvas and choosing **+ Add Variable** in the properties panel. Choose **Source Variable**.
2. Enter a name for your variable.
3. Choose how the variable should be filled:
 - **Manual Input**
 - **Derive Value.** Derived variables are hidden in the data preview or in an SAP Analytics Cloud story.
4. If you choose **Manual Input**, you can set a default value.
5. If you want to derive the value, you need to
 1. Define a lookup entity (a view).
 2. Select the column from which the value is to be derived.
 3. Map the parameter of the lookup entity. If it has no input parameter, you can set a constant value or create a variable.

Create a Restricted Measure Variable

1. You can create a new source variable by clicking the background of the canvas and choosing **+ Add Variable** in the properties panel. Choose **Restricted Measure Variable**.
2. Choose the dimension for which the variable should be created.
3. Choose the **Filter Type**. The filter can be
 - a single value: The user should enter a single value to filter on. You can optionally specify a default value.

- multiple single values: Allows the user to enter more than one value to filter on.
 - an interval: Allows the user to enter an interval of values to filter on.
 - a range of values: Allows the user to enter a range of values to filter on.
4. You can select *Mandatory*, if the user has to enter a value in the prompt in the preview or the story in SAC.

Create a Filter Variable

You can create only one filter variable for each dimension.

1. You can create a new filter variable by clicking the background of the canvas and choosing **+** *Add Variable* in the properties panel. Choose *Filter Variable*.
2. Choose the dimension for which the variable should be created.
3. Choose the *Filter Type*. The filter can be
 - a single value: The user should enter a single value to filter on. You can optionally specify a default value.
 - multiple single values: Allows the user to enter more than one value (or range of values) to filter on.
 - an interval: Allows the user to enter an interval of values to filter on.
 - a range of values: Allows the user to enter a range of values to filter on.
4. You can set a default value.
5. You can select *Mandatory*, if the user has to enter a value in the prompt in the preview or the story in SAP Analytics Cloud.

Create a Reference Date Variable

1. You can create a new reference date variable by clicking the background of the canvas and choosing **+** *Add Variable* in the properties panel. Choose *Reference Date Variable*.
2. Choose how the variable should be filled:
 - *Manual Input*
 - *Derive Value*. Derived variables are hidden in the data preview or in an SAP Analytics Cloud story.
3. If you choose *Manual Input*, you can set a single value as a default value. The variable is always mandatory.
4. If you want to derive the value, you need to
 1. Define a lookup entity (a view).
 2. Select the column from which the value is to be derived.
 3. Map the parameter of the lookup entity. If it has no input parameter, you can set a constant value or create a variable.

Use Variables in the Analytic Model

You must use each of the variables that you create in your analytic model or you will receive an error instructing you to use or delete them.

7.6.6 Using the Data Preview

The data preview helps you to determine whether your data is correct.


As this is an analytical preview, you can navigate through the various dimensions, and view the aggregated data. This means, you can see how the data will look like in an SAP Analytics Cloud story.



You can drill down by rows and columns by clicking on the icons at the objects.

Setting Filters

If you want to set filters, choose  [Add Filter](#) and select [Measures](#) or [Dimensions](#).

Using the Builder Panel

The [Builder](#) panel is displayed at the right side of the application. You can show it or hide it by choosing  [Query Builder Designer Panel](#).

When you choose [Available Objects](#), you get a list of all available dimensions and measures in the analytic model. Here you can select dimensions and measures and assign them directly to the table's rows or columns by clicking  [Column](#) or  [Row](#).



You can also drag and drop dimensions from the [Available Objects](#) panel into the [Builder](#) panel.

Some functions that make it easier for you to work with the panel:

- You can search for items.
- You can change the items' display between [ID](#) and [Description](#) as well as their sort order by clicking [More](#).
- You can resize the width of the left side panel for [Available Objects](#). This way you can display long dimension names.

Under [Rows](#) and [Columns](#), you see all measures and dimensions that are displayed in the table.

For the [Rows](#) and [Columns](#), you are offered a context menu that you can open by clicking [More](#). Here you can use the following functions:

- [Totals](#): You can set the display of the totals.
- [Zero Suppression](#): You can suppress zeros in rows and/or columns.
-  [Query > Swap Axes](#): You can exchange rows and columns.
-  [Query > Set Variables](#): You can change the value for the variable.

Using the Style Panel

To change the number formatting of your measures, switch to the [Style Panel](#). Here you can find several formatting options.

Using the Table's Context Menu

Depending on the context menu of a measure or dimension, you are offered different options.

7.7 Analytical Datasets (Deprecated)

The *Analytical Dataset* semantic usage is now deprecated and we recommend that you use the new *Fact* semantic usage instead.

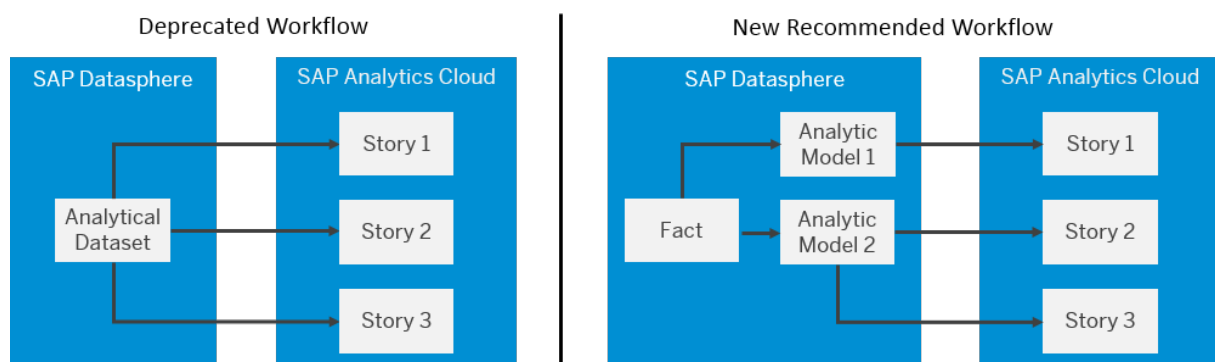
This topic contains the following sections:

- [Use Facts with Analytic Models to Expose Data to SAP Analytics Cloud](#) [page 351]
- [Migrate Your Analytical Datasets to Facts](#) [page 352]

Use Facts with Analytic Models to Expose Data to SAP Analytics Cloud

Facts (see [Creating a Fact](#) [page 305]) allow you to model data in all the ways familiar from analytical datasets, but they are not intended to be consumed directly in SAP Analytics Cloud.

The preferred way to expose data to SAP Analytics Cloud is now to identify your measures in a table or view with a semantic usage of *Fact* and then to use this fact in one or more analytic models, each of which can be consumed by one or more stories (see [Creating an Analytic Model](#) [page 337]).



This new workflow provides the following benefits:

Feature	Analytical Dataset	Fact + Analytic Model
Aggregations	Standard aggregations only, such as SUM, COUNT.	Standard and post-aggregation measures (calculated measures, restricted measures, and count distinct measures), including support for exception aggregations. Move post-aggregation measures from your SAP Analytics Cloud stories into your analytic models to benefit from real-time previewing of results and to promote re-use.
Measures and attribute selection	All measures and attributes and first-level dimension always exposed to SAP Analytics Cloud. More remote dimensions cannot be included.	Model as many measures, attributes, and associations to dimensions as appropriate in a fact, and then choose to expose only those that you need in each analytic model. You can select attributes from any dimension that is associated with the fact, whether it is a first-level dimension (directly associated with the fact) or any dimension that can be reached by following further associations.
Data viewer	Relational data viewer shows lists of records	Rich analytic viewer, based on the SAP Analytics Cloud Data Analyzer, supports measure and attribute selection, filtering and pivoting, and hierarchy support.

Where previously you would set the *Semantic Usage* of a view to *Analytical Dataset*, we now recommend that you choose *Fact* and then use your fact and its associated dimensions as sources for an analytic model.

Migrate Your Analytical Datasets to Facts

The *Analytical Dataset* semantic usage is deprecated, but existing views can continue to be consumed, at present, by SAP Analytics Cloud. However, we encourage you to migrate your analytical datasets to facts.

If your analytical dataset is consumed by one or more SAP Analytics Cloud stories, to minimize disruption:

1. Open your analytical dataset and click *Create Analytic Model* under the *Semantic Usage* property to create an analytic model with all the same measures, attributes and associated dimensions as your view.
2. Save and deploy your analytic model.
3. In your SAP Analytics Cloud story, add your analytic model as a new data source.
4. Copy relevant pages of your story to work on.
5. For each affected chart or table on your copied page, switch the measure and attribute references to those provided by your analytic model.
6. Compare the charts and tables between the original and copied pages and, once everything is the same, delete the original page.
7. When all of your stories that consume a particular analytical dataset are fully converted:
 1. Open your analytical dataset view and change the semantic usage to *Fact*.
 2. Deploy your view.
 3. Deploy your analytic model.

Now you can benefit from all the advantages of the analytic model:

- Enhance your analytic model by removing unnecessary measures and dimensions and adding analytic measures as needed.
- Move post-aggregation measures from your SAP Analytics Cloud stories into your analytic models to benefit from real-time previewing of results and to promote re-use.
- Create additional analytic models on your fact as needed.

7.7.1 Create a Story Filter (Deprecated)

Add a story filter to a view column to prompt users to filter on it when they consume your view in SAP Analytics Cloud.

Context

ⓘ Note

Story filters are deprecated and are not supported by entities with a *Semantic Usage* of *Fact*. You should instead, create a filter variable in your analytic model to trigger the display of the *Set Variables* dialog in SAP Analytics Cloud (see [Add a Variable \[page 348\]](#)).

Story filters that you add to your view will trigger the display of the *Set Variables* dialog when an SAP Analytics Cloud user creates a story with your view as a data source (see [Setting Story Variables](#)).

ⓘ Note

In order to add story filters, you must:

- Set the *Semantic Usage* to *Analytical Dataset*.
- Enable the *Expose for Consumption* switch.

Procedure

1. Select the output node of your view to display its properties in the side panel and scroll down to the *Attributes* section.
2. Hover over the column which you want to set the filter on, then click **⋮ (Menu) ▶▶ Add Story Filter ▶** to open the filter properties in the side panel.
3. Set the following properties as appropriate.

Property	Description
Type	Select the type of value that you want to allow or require the user to filter on. You can choose from: <ul style="list-style-type: none"> <i>Single</i> - The user should enter a single value to filter on. You can optionally specify a default value. <i>Range</i> -The user should enter <i>From</i> and <i>To</i> values to specify the range to filter on. You can optionally specify default values.
Mandatory	Requires the user to enter a value.
Multiple Entries	Allows the user to enter more than one value (or range of values) to filter on.

- Click the view name in the breadcrumbs at the top of the side panel to return to the view properties.

The column displays a filter icon to show that it contains a story filter.

Note

To edit or remove the filter, hover over the column and click **⋮ (Menu) ▶ ▶ Edit Story Filter ▶** or **Remove Story Filter**.

8 Modeling Data in the Business Builder

Users with the *DW Modeler* role can use the *Business Builder* editors to combine, refine, and enrich *Data Builder* objects and expose lightweight, tightly-focused perspectives for consumption by SAP Analytics Cloud and MS Excel.

This topic contains the following sections:

- [Consume Data From the Data Builder in Business Entities \[page 355\]](#)
- [Combine Business Entities in Fact Models and Consumption Models \[page 355\]](#)
- [Expose Data in Perspectives \[page 356\]](#)
- [Import SAP BW/4HANA Queries \[page 356\]](#)

Consume Data From the Data Builder in Business Entities

Each business entity created in the *Business Builder* consumes data from a *Data Builder* entity. As you can, at any time, switch the data source of a business entity to a different *Data Builder* entity, this loose coupling allows you to maintain stable business entities for reporting, even as your physical data sources change.

- You can create a business entity by selecting a *Data Builder* entity as its source (see [Creating a Business Entity \[page 358\]](#)).
- You can remove unneeded measures and attributes to simplify your business entity for a particular reporting need.
- You can enrich your business entity with new measures (including derived and calculated measures), attributes, and other properties.
- You can change the data source of your business entity to a new *Data Builder* entity if necessary.

Combine Business Entities in Fact Models and Consumption Models

Combine your business entities into star-schemas to prepare them for consumption (see [Creating a Consumption Model \[page 376\]](#)).

You can use a single business entity in multiple consumption models and modify it by adding and removing measures and attributes as appropriate for a particular reporting context.

You can, optionally, combine your business entities into an intermediate fact model and then use this as a source for multiple consumption models (see [Creating a Fact Model \[page 372\]](#)).

Expose Data in Perspectives

Create perspectives from a consumption model for exposure to SAP Analytics Cloud and other BI clients, MS Excel, and other apps and tools (see [Define Perspectives \[page 380\]](#)).

Import SAP BW/4HANA Queries

Import an SAP BW4/HANA query, along with its supporting InfoObjects and CompositeProviders to SAP Datasphere (see [Importing SAP BW/4HANA Models \[page 386\]](#)).

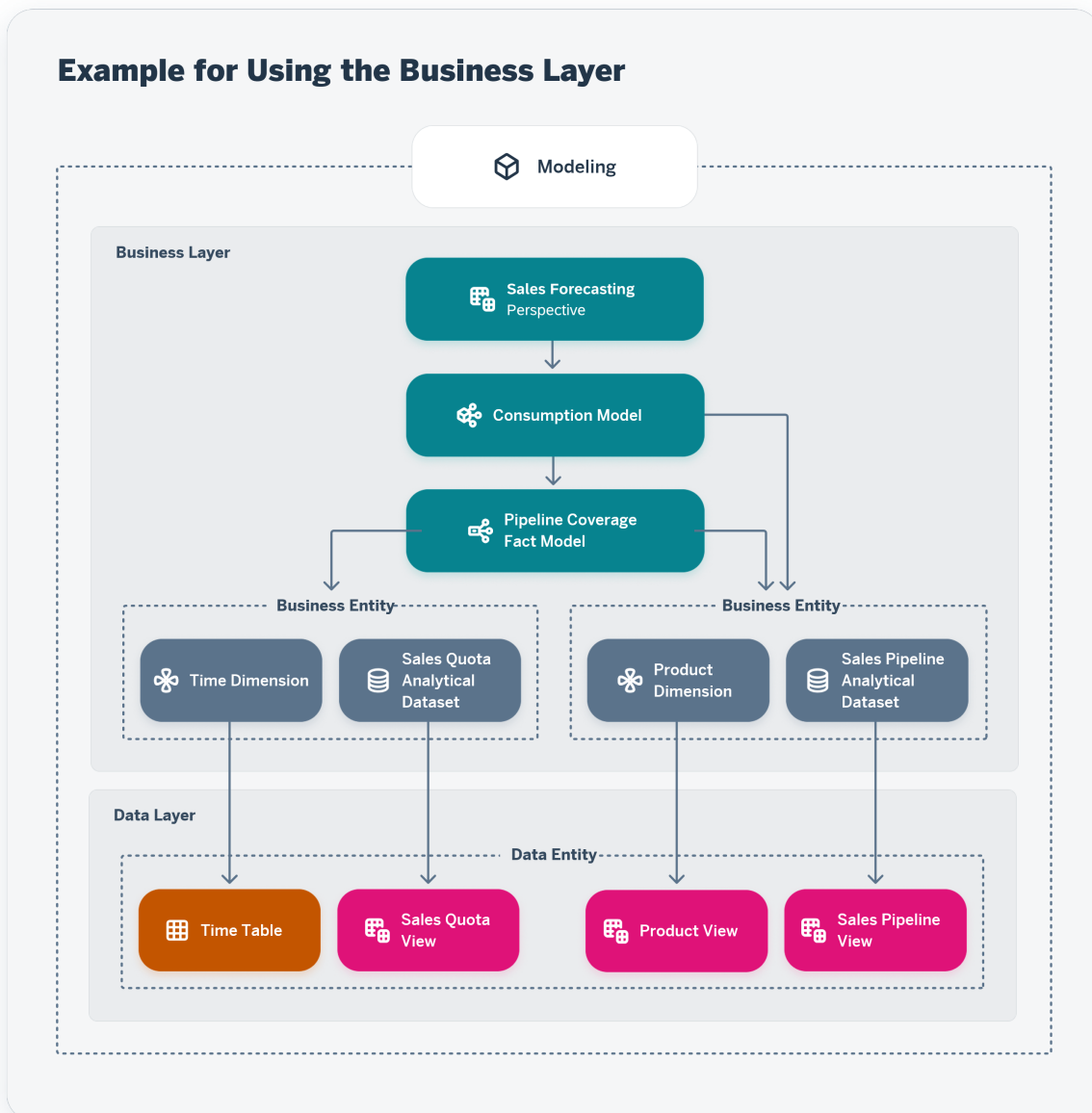
8.1 Example for Using the Business Builder

Here is a simple example for a scenario using the Business Builder.

Let's look at the objects you might need to create in order to analyze the sales performance of a given product range for a given month.

In the Data Builder, your data engineer has created the tables and views with the data you need. In the Business Builder, you first create the business entities *time*, *sales quota*, *product* and *sales pipeline* and connect them to the objects in the Data Builder. Since you want to reuse this information for a different analysis, you then create a fact model on *pipeline coverage* on top. On top of that, you create your consumption model with the *sales forecasting* perspective, which contains all the data that is relevant for your analysis.

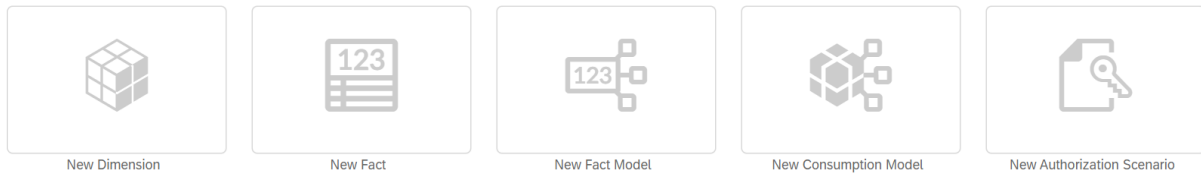
Example for Using the Business Layer



8.2 Business Builder Start Page

The Business Builder start page gives you access to the different editors.

You can access the editors for consumption models, fact models, analytical datasets and dimensions, and create authorization scenarios.



Apart from that, there are several functions which you can access here directly:

- You can search for objects.
- You can filter and sort the object list via the context menu.
- You can create new objects directly by choosing **+** (*Add*).
- You can create new folders to organize your objects. You can move objects to different folders by selecting the object and choosing **□** (*Folder*) and then *Move*. You can then select the target folder.
- You can import entities from other systems choosing **↗** (*Import*) and then *Import from connection*. And you can check the logs choosing **📄** (*Import*) and then *Import logs*.
- When you select an object, you can edit it directly by choosing **✎** (*Edit*).
- You can duplicate fact and consumption models by selecting them and choosing **📄** (*Copy model*).
- When you select one or several objects, you can delete them by choosing **🗑** (*Delete*).

8.3 Creating a Business Entity

You use business entities to build your consumption model for analysis and reporting.

Prerequisites

Views or tables have been created in the Data Builder.


Context

Business entities can define measures or attributes. Measures are quantifiable values that refer to an aggregable field of the underlying model. An attribute is a descriptive element of a business entity and provides meaningful business insights into measures. The underlying model is a view or a table, which has been created in the Data Builder.

In order to model a meaningful consumption model, business entities define associations between each other. All potential association targets can be pre-defined on the data layer in order to provide business users a variety of modeling options to choose from when preparing their use case-specific consumption model.

Business entities can be modeled as a dimension or as a fact. The definition doesn't really differ but rather the intended usage within the consumption model. Dimensions are generally used to contain master data and must have a key defined. Facts are generally used to contain transactional data and must have at least one measure defined.

Procedure

1. In the side navigation area, click  (*Business Builder*), select a space if necessary, and click *New Dimension* or *New Fact* to open the editor.
2. Select a source. You can use views or tables which have been created in the Data Builder as a source.
3. Select the properties you want to copy from the source. You can copy attributes, measures, input parameters and key definitions.

Note

Associations cannot be copied yet.

4. Enter a descriptive name to help users identify the object. This name can be changed at any time. By default, the name of the source is taken.
5. A technical name is generated.

To override the default technical name, enter a new one in the field. Technical names can contain only alphanumeric characters and underscores.

Note

Once the object is saved, the technical name can no longer be modified.

6. Choose *Public Data Access* if you want to allow unrestricted access for this business entity.
Keep in mind that in most cases data restrictions need to be applied. To do this, select an appropriate authorization scenario. More information: [Authorization Scenario \[page 368\]](#)
7. Define the version of your business entity. More information: [Create Versions of an Object \[page 384\]](#)
8. If you feel that you have chosen the wrong type for your purpose, you can still convert your dimension into a fact, or vice versa.
9. Save your entries.
10. Define the measures of your business entity. More information: [Define Measures \[page 360\]](#)
11. Define the attributes of your business entity. More information: [Define Attributes \[page 364\]](#)
12. Define the keys of your business entity. You need to define at least one unique key in order to use the business entity as a dimension later on. More information: [Define a Key \[page 365\]](#)
In case you have taken over the key definition from the source, you need to verify the uniqueness of your key. Go to the *Key Definitions* tab and choose *Verify*.
13. Associate the dimensions. More information: [Define Associations \[page 366\]](#)
14. You can assign an authorization scenario, if you want to restrict the access to this business entity. More information: [Assigning an Authorization Scenario \[page 370\]](#)
15. You can check your business entity in the data preview. A data preview is only possible if the underlying object in the Data Builder is deployed.
16. Once you are satisfied with the definition of your business entity, you can set the status of your version to *Ready to Use*. For more information on versions, see [Create Versions of an Object \[page 384\]](#).

Results

You can use your business entity in fact models and consumption models.

8.3.1 Define Measures

A measure is a quantifiable value that refers to an aggregatable field of the underlying model.

Context

Measures can only be used in a consumption model if the business entity is used as a fact source.

Procedure

1. Go to the *Measures* tab.
2. To define one measure, choose **+** *Add*.
To define multiple measures: Choose **≡** *Add Measures*. Select your measures and choose *Apply*. By default, these measure are of type *Aggregation*. To change details for the measure, choose **>** *Details*.
3. Give your measure a meaningful and business-ready name. This can contain a maximum of 120 characters/special characters.
4. Select a type of measure and define the properties. The properties depend on the type of measure. The following types are available: [Measure Types \[page 361\]](#)
5. For derived measures, you can define restrictions. In the restrictions editor, you can choose between the simple and the advanced mode. In the simple mode, you are guided by a wizard. In the advanced mode, you have more options and are able to create more complex criteria.
6. For measures of type *Aggregation* set the aggregation type (sum, average, count, max, min).
7. Beside the standard aggregation, you can define an exception aggregation. For more information, see [Aggregation and Exception Aggregation \[page 363\]](#).
The exception aggregation will be visible in stories in SAP Analytics Cloud, but not in the data preview.
8. Decide whether your measure should have a currency or unit assigned. The unit or currency can have a constant value or can be derived from another field.
9. Decide whether your measure should be an *Auxiliary Measure*. An auxiliary measure can be used for further calculation upon within the given business entity but is not exposed to the consumption model. For instance, general measures that are exposed via differently restricted derived measures might not be required in the consumption model itself.
10. You can choose if null values shall be treated as NULL or zero in the data preview and story consumption.
11. Save your entries.
12. To duplicate measures, choose **□** *Duplicate existing measure*.

8.3.1.1 Measure Types

Depending on the type of model, the following measure types are available:

Type of measure	Description
Aggregation	You need to set the aggregation type (sum, average, count, max, min).
Derived Measure	A derived measure refers to another measure and allows restrictions on available attributes.
Count Distinct	Counts unique (tuple) occurrences of attributes (e.g. in case multiple rows contain the country "Germany", it is counted only once).
Calculation	A calculated measure references other measures and allows the combination of measures with elementary arithmetic (operations of addition, subtraction, multiplication, and division), and also more complex functions like maximum/minimum, IF (as function), mathematical functions like round, and conversion functions like convert to decfloat.
Fixed Value	Define a fixed value that can be used for further calculation.
Converted Currency	You can define all the parameters you need to convert currency values into another currency.

8.3.1.2 Use Currency Conversion

You can convert currency values into another currency.

Prerequisites

To enable currency conversion, you create the necessary objects in your space. For more information, see [Enabling Currency Conversion with TCUR* Tables and Views \[page 52\]](#).

Context

Currency conversion on calculated measures enables you to have a common currency for reporting, e.g. company code or cost area currency.

You can use currency conversion for measures in a business entity or a consumption model.

Procedure

1. Go to the *Measures* tab. Define your measure.
2. Set the properties for your source in the *Source* section.

Property	Description
Measure Type	Choose <i>Converted Currency</i> .
Source Measure	Select the <i>Source Measure</i> . It has to have the type <i>Amount with Currency</i> .
Client ID	Enter a three character string that is used to separate tenants within ERP system tables. This is used in the conversion tables to select the correct rows for each user.
Target Currency	Select your target currency.

3. Set the properties in the *Reference Date* section.

Property	Description
Current Date	The current date is used.
Constant Value	Select a date on the calendar.
Field Reference	This is a reference to a field in the row indicating which date to use.

4. Select the *Conversion Type*. The conversion types have been defined in your source system. Contact your system administrator for details, if you are unsure which one to take.
5. Select the *Error Handling* method you would like to have applied when a row could not be converted.

You have the following options:

Property	Description
Set to null	If any error happens during the currency conversion, the target value is set to NULL.
Fail on error	The conversion fails with an error.
Keep unconverted	The input value is returned.

6. You can open or refresh the Data Preview panel to review the results of your currency conversion.

Results

When you use the measure with currency conversion in a consumption model, you can see the converted currency in a story in SAP Analytics Cloud.

8.3.1.3 Aggregation and Exception Aggregation

For measures, you can define a standard aggregation and an exception aggregation.

The aggregation behavior determines how measure values are aggregated in perspectives that use different attributes. In order to calculate the values of measures, the data from the perspective has to be aggregated to the detail level of the query. Now we take a look at the differences between standard aggregation and an exception aggregation

Standard Aggregation

The basic form of aggregation is standard aggregation. When performing standard aggregation, the system can aggregate measures according to the following rules: SUM (calculate the total value), MIN (calculate the minimum value), MAX (calculate the maximum value), AVG (calculate the average value), and COUNT (calculate the number of distinct values). The standard aggregation is independent from a specific attribute. The system performs standard aggregation over all attributes that are not in the filter, irrespective of their sequence. This is the most well-known aggregation behavior.

Exception Aggregation

In addition to standard aggregation, there is also exception aggregation. The aggregation rules for exception aggregation depend on an attribute. You specify how a measure is aggregated with regard to one or more attribute. An attribute is needed for exception aggregation in order to define the granularity with which the aggregation rule is applied. The system aggregates the measure to this detail level according to the rules of standard aggregation.

Exception aggregation is also always performed in addition to standard aggregation. It is not an alternative to standard aggregation.

⚠ Caution

Be aware that the settings in the consumption model or fact model concerning aggregation override the settings in the layer below. This means, that settings in the business entity will be overridden by the settings in the consumption model, and also settings in the fact model will be overridden by the settings in the consumption model.

Example

Let us take aggregation rule (AVG) as an example: The system calculates the average from the sum of all values divided by the number of entries. To define the number of entries, a granularity of data must be defined to which the system is to aggregate the multi-dimensional data according to the rules of standard aggregation. The result of the standard aggregation is the starting point of the exception aggregation, as the basis for the calculation according to the *Average of All Values* rule.

From a business point of view for example, exception aggregation arises if warehouse stock cannot be totaled up at the time, or if counters count the number of characteristics for a specific characteristic.

8.3.2 Define Attributes

An attribute is a descriptive element of a business entity.

Context

It provides meaningful business insight into corresponding measures.

Procedure

1. Go to the *Attributes* tab.
2. To define one attribute: Choose **+** *New attribute*.
To define multiple attributes: Choose **≡** *Add Attributes*. Select your attributes and choose *Apply*. To define the details for an attribute, choose **>** *Details*.
3. Select the attribute type. More information: [Attribute Types \[page 365\]](#)
4. You can define an identifier for an attribute.

An identifier is an attribute which is closely connected to the attribute. Usually its the technical identifier. When you have an identifier defined, all the filters on this attribute will be applied to the identifier, so that non-unique texts or texts, which are changing are filtered out.

Example

For the attribute *Customer Name* you define *Customer ID* as identifier. In the data preview, you can then choose if it should be displayed as <customer name> (<customer ID>), as <customer name>, or as <customer ID>.

5. You can define attributes with texts and language dependent texts. This enables you to display your data with ID and text, or with a language dependent text, when you preview or analyze your data. For language dependent texts, the language in the user settings is taken.
 - To display a text: Select *Text*. Then select the text attribute.
 - To display a language dependent text: Select *Language Dependent Text*. Select the text table. The text table has to be defined in the Data Builder.

Note

Only one column can be defined as *Text*. By default, the first column with the semantic type *Text* from your text table is used. To use a different column, either reassign the semantic type *Text* in the table, or create a View, selecting the required column.

6. Decide whether your attribute should be an *Auxiliary Attribute* meaning an attribute which can be used further within the business entity, but is not exposed to consumption models.
7. Save your entries.

8.3.2.1 Attribute Types

There are different types of attributes to be used in business entities.

Attribute Types

Attribute Type	Description
Source Attribute	Attribute from the source model.
Calculated Attribute	<p>A calculated attribute is based on a formula which is specified as an SQL expression and must match the chosen data type. The calculation evaluates on rows - access to values in other rows (cell-reference) is not possible.</p> <p>An automatic validation takes place to provide immediate feedback.</p> <p>For more information on syntax and available functions, please refer to the SAP HANA SQL Functions Reference and Operators Reference.</p>

8.3.3 Define a Key

Keys unambiguously identify a single record in a business entity.

Context

Keys need to be unique. A business entity can have more than one key and you can define a key with multiple members.

Caution

Don't use fields of SAP HANA datatype BIGINT as key of a dimension, as this can lead to problems with displaying large values in SAP Analytics Cloud.

Procedure

1. Go to the [Key Definitions](#) tab.
2. Choose **+** [Add](#).
3. Enter a title for your key.
4. Select one or several of the fields as key. A key can have multiple members. The key is validated by the system and you get a message telling you how many entries within the respective business entity there are and how many of them match to the defined key. This information shall help you evaluate whether the set configuration makes sense.
5. For compound keys, you can define a representative key, and the sequence of keys.

When using compound keys it is very important to consider the order of the key fields as this order will define the compounding sequence and the representative key field. The compounding sequence is defined from top to bottom (ideally from “more general” to “specific”). The representative key field will be the last and ideally most specific field in the list. The order of key fields cannot be changed anymore when the key is used in associations.

When a perspective is deployed the compounding information for a dimension will only be deployed for the primary key. The primary key is the first key in the list of keys that matches all fields which are used in the association to the dimension.

6. Enter a description for the key field.
7. Save your entries.

8.3.4 Define Associations

An association describes the relation between business entities.

Context

The associated business entities will be available as dimensions when you create a consumption model.

Procedure

1. Go to the [Associations](#) tab.
2. Choose **+** [Create](#).
3. Select your target business entity. The information [Referential Integrity Ensured](#) is used for performance optimizations during querying. If switched on, it indicates that each data record of the currently opened business entity finds a match in the selected target business entity with the configured key mapping. If there is no match found for every data record but [Referential Integrity Ensured](#) is switched on, it can impact the result during querying.

4. Define which fields of the business entity shall function as keys.
 1. Select a key offered by the target business entity via the drop-down menu.
 2. Map the corresponding target key field to the corresponding field of the current business entity.
 3. You get a message on the key mapping validation: This provides you feedback on how many entries within the current business entity there are and how many of them match to entries of the target business entity. This information shall help you evaluate whether the set configuration makes sense.
5. Via *Association Context* you can provide information on the context of the association within the business entity. This allows for multiple associations between the same business entities (in different contexts) and provides additional business meaning to the association. The context will be available as additional information on every usage of the association, e.g. in a consumption model. To define a context, you can provide:
 - a title (max. 120 characters & special characters allowed)
 - a description (max. 5000 characters & special characters allowed)

8.3.5 Define Input Parameters

You have to create input parameters, if the underlying view has input parameters defined.

Context

Input parameters defined for business entities are inherited to the fact and consumption models, but can also be overwritten.

For more information on input parameters, see [Create an Input Parameter \[page 231\]](#).

Procedure

1. Go to the *Input Parameters* tab.
2. To define input parameters: Choose **+** (*New*).
3. Choose the source for the input parameter.
4. You can expose the input parameter to the fact models and consumption models, which are created on top of the business entity.

In order to expose it to fact models and consumption models, you need to define a default value.

8.3.6 Add an External Hierarchy

You can add one or more external parent-child hierarchies which have been defined in the source view in the Data Builder.

Context

External hierarchies are not created within a business entity; instead, they are stored in separate hierarchy tables and can be used in views.

Procedure

1. Go to the *Hierarchies* tab.
2. To add a hierarchy, choose **+** (*Add*).
3. Select your hierarchy and choose *Apply*. The key mapping and the hierarchy structure will be prefilled. The hierarchy structure cannot be changed here, only in the hierarchy view in the Business Builder.
4. Enter a *Business Name* for the hierarchy.
5. Choose *Verify* to check the key definition. If the hierarchy contains more values than the object in the Business Builder, you can still save it, and the entries which do not fit are ignored.
6. Save the hierarchy.

8.4 Authorization Scenario

Authorization scenarios allow modelers to define which data is relevant to a user's context. They are made available through business entities and can be used in consumption models for specific use-cases.

In the Business Builder, modelers can create, assign, and consume authorization scenarios.

Related Information

[Creating an Authorization Scenario \[page 369\]](#)

[Assigning an Authorization Scenario \[page 370\]](#)

[Using an Authorization Scenario in a Consumption Model \[page 371\]](#)

[Securing Data with Data Access Controls](#)

[Create a "Single Values" Data Access Control](#)

[Apply a Data Access Control \[page 245\]](#)

8.4.1 Creating an Authorization Scenario

Authorization scenarios help you control data access for business entities leveraging data access controls.

Context

Authorization scenarios help you filter the data available to specific business entities and their subsets based on existing data access controls.

For example, a manager wants to access their employees' data. They only should see employees from their team. An authorization scenario can help them make sure they only see data that is relevant to the team they manage.

Procedure

1. Go to the [Business Builder](#) and select [New Authorization Scenario](#).
2. Select a data access control. Make sure to select a deployed data access control to have access to data preview.
3. Click [Create](#).
4. Go to the [Data Restrictions](#) tab and click [+ Data Restrictions](#).
5. Select a [Business Entity](#).
6. You have access to the business entity's key definitions. It gives you access to key members. For each business entity used as data restriction, you need to map each key member to the data access control's criteria column.
To learn more about defining keys for a business entity, see [Define a Key \[page 365\]](#).
7. Click [Save](#) to save your authorization scenario.

You can define several restrictions on several business entities within the same authorization scenario.

Results

Your new authorization scenario is ready to use. You can add as many [Data Restrictions](#) as you need within your authorization scenario.

Data Preview helps you make sure that the authorization scenario works. You can see the business entity's keys used as data restrictions and the data allowed by the underlying data access control that is relevant to the user.

Related Information

[Assigning an Authorization Scenario \[page 370\]](#)


8.4.2 Assigning an Authorization Scenario

Once you've created an authorization scenario, you can assign it to a business entity to tailor data access to different business contexts.


Context

Authorization scenarios help you define different business authorization contexts that act as data filters. A manager wants to access their employees' data. They only want to see employees from their team. An authorization scenario can help them make sure they only see data that is relevant to the team they manage.

Procedure

1. In the *Business Builder*, select your business entity.
2. In the *Authorization Scenario* tab, click **+** *Add* and select your authorization scenario. You can add one or several authorization scenarios and use the ones that are relevant to your use-case.
3. Under *Data Restriction Context*, click  to open the *Select Context* dialog. You can see all available association contexts. They resolve your business entity defined as a restriction in the authorization scenario. Select your context and *Save*.
4. In the *General* tab under *Public Data Access*, you can choose to make your model public or not. You can combine it to an authorization scenario. If you've switched on *Public Data Access* for your business entity, using an authorization scenario is optional. If you've switched off *Public Data Access* for your business entity, you have to use an authorization scenario. If your model has no authorization scenario or *Public Data Access*, you won't be able to use our model.

Results

You can see the different authorization scenarios you've created for your business entity in the data preview. In the Data Preview side panel, click . Under *Authorization Scenario*, you can directly switch from one authorization scenario to another. Authorizations are automatically applied on data.

Related Information

[Creating a Consumption Model \[page 376\]](#)

[Authorization Scenario \[page 368\]](#)

[Creating an Authorization Scenario \[page 369\]](#)

[Using an Authorization Scenario in a Consumption Model \[page 371\]](#)

8.4.3 Using an Authorization Scenario in a Consumption Model

Choose from your business entities which authorization scenarios to use in a consumption model.

Context

Consumption models are use-case specific. You can take advantage of the authorization scenarios that you've previously created and assigned to business entities to refine the ways in which users can or cannot access data.

Procedure

1. Select a consumption model.
2. Go to the *General* tab. Under *Authorization Scenario*, next to *Supported Authorization Scenarios*, click **+**.
3. The *Authorization Scenarios* dialog opens. The listed authorizations scenarios come from the business entities used as Fact Source in the consumption model. Select the ones you want to use in your consumption model. Click *Select*.

You can only select the authorization scenarios that are shared between all Fact Source. If a Fact Source allows Public Data Access, it doesn't need to share an authorization scenario:

- Allow Public Data Access: switch it on if this business entity and its data should be accessible for everyone.
 - Authorization Scenario: if your business entity shouldn't be public, an authorization scenario must be selected to allow consumption.
4. Click *Save*.

Results

The authorization scenario is visible in the consumption model's data preview. You can chose to use authorization scenarios or not depending on the use-case you dedicate your consumption model to.

Note

- If Pulic Data Access is turned on on a consumption model, you are able to to select “without authorization scenario public access” in the Data Preview.

- You can define perspectives in a consumption model. To learn more, see [Define Perspectives \[page 380\]](#).

Related Information

[Creating a Business Entity \[page 358\]](#)

[Creating an Authorization Scenario \[page 369\]](#)

[Assigning an Authorization Scenario \[page 370\]](#)

8.5 Creating a Fact Model

Fact models are reusable models you can use to streamline the creation of other models within the same business context.

Prerequisites


You already have analytical datasets, dimensions, or fact models defined.

Context

Fact models let you predefine an array of measures, attributes, and filters that are relevant to a specific business context. Fact models can then be reused in several use cases within that business context.

As an example, you have five consumption models that require the same ten calculated attributes. Instead of defining each of the ten calculated attributes five times, you can create a single fact model that includes those calculated attributes you need and reuse it in each consumption model where it's relevant.

Procedure

1. In the side navigation area, click  (*Business Builder*), select a space if necessary, and click [New Fact Model](#) to open the editor.
2. Enter a title for your fact model.
3. Choose the initial fact source you want to use in your fact model: you can choose analytical datasets, dimensions or fact models.
4. Enter an alias for your initial fact source and click [Create](#).

At this point, your fact model is ready to use. You can also choose to include or remove measures and attributes relevant to several use cases within a same business context.

5. Define the *Measures* of your fact model. For more information, see [Define Measures \[page 373\]](#).
6. Define the *Attributes* of your fact model. For more information see [Define Attributes \[page 374\]](#).
7. Define the *Dimension Sources* which should be exposed for consumption. For more information see [Expose Dimension Sources \[page 374\]](#).
8. Define the *Filters* of your fact model. For more information see [Define Filters \[page 375\]](#).

8.5.1 Define Measures

You can define measures for your fact model.

Context

A measure is a quantifiable value that refers to an aggregatable field of the model. Business metrics such as profits and sales are common measures.

Procedure

1. Go to the *Measures* tab.
2. To define one measure, choose **+** *New Measure*.
To define multiple measures: Choose **≡** *Add Fact Source Measures*. Select your measures and choose *Apply*. To change details for the measure, choose **>** *Details*.
3. Give your measure a meaningful and business-ready name. This can contain a maximum of 120 characters/special characters.
4. Select a type of measure and define the properties. The properties depend on the type of measure. The following types are available: [Measure Types \[page 361\]](#)
5. Beside the standard aggregation, you can define an exception aggregation. For more information, see [Aggregation and Exception Aggregation \[page 363\]](#).
The exception aggregation will be visible in stories in SAP Analytics Cloud, but not in the data preview.
6. For derived measures, you can define restrictions. In the restrictions editor, you can choose between the simple and the advanced mode. In the simple mode, you are guided by a wizard. In the advanced mode, you have more options and are able to create more complex criteria.
7. Decide whether your measure should have a currency or unit assigned. The unit or currency can have a constant value or can be derived from another field.
8. Decide whether your measure should be an *Auxiliary Measure*. An auxiliary measure can be used for further calculation upon within the given business entity but is not exposed to the consumption model. For instance, general measures that are exposed via differently restricted derived measures might not be required in the consumption model itself.

9. Save your entries.
10. To duplicate measures, choose  [Duplicate existing measure](#).

8.5.2 Define Attributes




You can define attributes in your fact models.

Context

Attributes in fact models can either reference attributes from the included fact or dimension sources or be calculated from other attributes of the current fact model.

For more information on the different attribute types, see [Attribute Types \[page 378\]](#).

Procedure

1. Go to the [Attributes](#) tab.
2. To define one attribute: Choose  [New attribute](#).
To define multiple attributes: Choose  [Add Dimension Source Attributes](#). Select your attributes and choose [Apply](#). To define the details for an attribute, choose  [Details](#).
3. Select the attribute type. More information: [Attribute Types \[page 378\]](#)
4. Decide whether your attribute should be an [Auxiliary Attribute](#) meaning an attribute which can be used further within the fact model, but is not exposed to consumption models.
5. Save your entries.

8.5.3 Expose Dimension Sources

You can expose dimension sources in you fact models.

Prerequisites

You already have mapped dimension sources defined in your fact model.

Context

The associations that you defined for the business entity are available as dimension sources in the fact model. You can expose these dimension sources, so that they can be used in the consumption model.

Procedure

1. Go to the *Exposed Dimension Sources* tab.
2. Choose **+** *Add* and select the dimension source you want to expose.
3. Enter a name and a description.
4. Save your entries.

8.5.4 Define Filters

You can define filters for your fact models.

Context

You can filter the data that is visible depending on your business needs. For instance, you only want to include values over a certain threshold.

Procedure

1. Go to the *Filters* tab.
2. Create your filter formula using values, elements and logical operators.
3. Save your filter.

8.6 Creating a Consumption Model

Consumption models are the basis to consume your data.

Prerequisites

Analytical datasets, fact models or dimensions have already been defined.

Context



You build your consumption models on top of business entities or on top of fact models, which have some elements of the consumption model already predefined.

These consumption models focus on specific analytical requirements and enrich the model accordingly. As an example, this could mean setting measures and attributes from different fact and dimension sources or fact models into relation by defining context-dependent navigations to compose use case-specific consumption models.

For instance, if we consider the case of Controlling - Profitability Analysis (COPA) in a business context, it becomes clear that three fact sources storing the measures *Actuals*, *Budget*, and *Forecast* are connected to different dimension sources via associations (*Material*, *Profit Center*, *SAP Mastercode*, and *Time*). All four dimensions from the first level are connected to the three fact sources. Given that the link between the different business entities was already established, the information on *Company Code* is automatically available through the *Profit Center* dimension source.

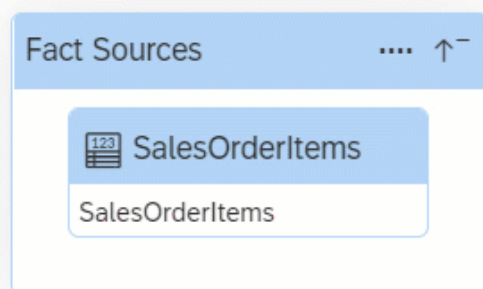
As a result, the consumption model allows harmonized reporting on different measures and attributes of all the included fact and dimension sources. The object used for reporting is the perspective which is defined within the consumption model editor.

Procedure

1. In the side navigation area, click  (*Business Builder*), select a space if necessary, and click *New Consumption Model* to open the editor.
2. Enter a title for your consumption model. Choose *Step 2*.
3. Choose the fact source you want to use in your consumption model: you can choose analytical datasets or fact models. Choose *Step 3*.
4. Define a *Source Alias* and choose *Create*. You are taken to the graphical representation of your model. At the center of the diagram, you can see your fact source.
5. You can zoom into your fact source and add associated business entities for consumption by selecting your fact source and choosing  (*Show Source Graph*).

If your consumption model contains just one fact source, you will see all associations from the selected fact by default.

6. To add dimension sources, choose **+** [Add Dimension Sources](#) and select your business entities. They then appear under *Dimensions* in the right half of the screen.



7. Define the authorizations for the consumption model. Go to the *General* tab and choose either *Allow public data access* or choose an authorization scenario.
8. Define the measures of your consumption model. More information: [Define Measures \[page 379\]](#)
9. Define the attributes of your consumption model. More information: [Define Attributes \[page 377\]](#)
10. Define the filters for your consumption model. More information: [Define Filters \[page 380\]](#)
11. Define the perspectives for your consumption model. More information: [Define Perspectives \[page 380\]](#)

8.6.1 Define Attributes

You can define attributes for your consumption model.

Context

Attributes in consumption models can either reference attributes from the included fact or dimension sources or be calculated from other attributes of the current consumption model.

Procedure

1. Go to the *Attributes* tab.
2. To define an attribute: Choose **+** *New attribute*.
To add multiple attributes: Choose **≡** *Add Attributes*. Select your attributes and choose *Apply*. To change details for an attribute, choose **>** *Details*.
3. Select the attribute type. More information: [Attribute Types \[page 378\]](#)
4. Decide whether your attribute should be an *Auxiliary Attribute* meaning an attribute which can be used further within the consumption model, but is not exposed to other consumption models or the perspective.

Note

In stories in SAP Analytics Cloud, auxiliary attributes are nevertheless displayed with the prefix [Hidden]. The same applies to attributes, which are used indirectly, for example in restrictions.

5. Save your entries.

8.6.1.1 Attribute Types

There are different types of attributes to be used in fact models and consumption models.

Attribute Types

Attribute Type	Description
Dimension Source Attribute	References an attribute from a dimension source in the consumption model.
Calculated Attribute	<p>A calculated attribute is based on a formula which is specified as an SQL expression and must match the chosen data type. The calculation evaluates on rows - access to values in other rows (cell-reference) is not possible.</p> <p>An automatic validation takes place to provide immediate feedback.</p> <p>For more information on syntax and available functions, please refer to the SAP HANA SQL Functions Reference and Operators Reference.</p>
Fact Source Attribute Mapping/Fact Source Attribute	Allows mapping of common attributes across the fact sources in the source model. Fact sources that do not provide an applicable attribute may default to NULL or to a constant value. You only have to map fact sources when your consumption model contains more than one fact source. If your consumption model contains only one fact source, the attribute type is called <i>Fact Source Attribute</i> .



8.6.2 Define Measures

You can define measures for your consumption model.

Context

A measure is a quantifiable value that refers to an aggregatable field of the underlying model.

Procedure

1. Go to the [Measures](#) tab.
2. To define a measure, choose **+** [New measure](#).
To add multiple fact source measures, choose  [Add Measures](#). Select your measures and choose [Apply](#).
To change details for a measure, choose **>** [Details](#).
3. Select a type of measure and define the properties. The properties depend on the type of measure. More information: [Measure Types \[page 361\]](#)
4. Beside the standard aggregation, you can define an exception aggregation. For more information, see [Aggregation and Exception Aggregation \[page 363\]](#).
The exception aggregation will be visible in stories in SAP Analytics Cloud, but not in the data preview.
5. Give your measure a meaningful and business-ready name. This can contain a maximum of 120 characters/special characters.
6. Decide whether your measure should be an [Auxiliary Measure](#). An auxiliary measure can be used for further calculation upon within the given business entity, but is not exposed to the consumption model. For instance, general measures that are exposed via differently restricted derived measures might not be required in the consumption model itself.
7. Save your entries.
8. To duplicate measures, choose  [Duplicate existing measure](#).

8.6.3 Add an External Hierarchy

You can add one or more external parent-child hierarchies to a fact or a dimension.

Procedure

1. Go to the [Hierarchies](#) tab.

2. To add a hierarchy, choose **+** (*Add*). If you don't see the **+** (*Add*) icon, this means that there are no hierarchies available. Either all the available hierarchies have been added to the consumption model, or there are no hierarchies defined in the facts or dimensions.
3. Select your hierarchy.
4. Enter a *Business Name* for the hierarchy and save it.

Results

You can now add the hierarchy to the perspective.

8.6.4 Define Filters

You can define a filter on fields.

Context

You can filter your data so that only a specific part is visible in the perspective.

Procedure

1. Go to the *Filters* tab.
2. You can choose between the simple and the advanced editor.
3. Advanced editor: you can define your syntax by adding elements, values, and operators.
4. Simple editor: you are taken through the steps by a wizard.

8.6.5 Define Perspectives

Perspectives are reusable configurations that contain a subset of a consumption models attributes, measures and parameters.

Context

You can create multiple perspectives for your consumption model which allows for different variants to enable quick validation while modeling. Perspectives can be used to create stories in SAP Analytics Cloud.

You can create a perspective in the *Perspectives* tab of your consumption model or by choosing *Data Preview* and saving your preview as a perspective.

Note

Older perspectives use internal naming conventions for dimensions that are not stable when transporting between tenants. If your perspective has this issue, the *Update Perspective for Transport* option is available at the bottom of its property sheet.

To make your perspective safe for transport, click this option and then click *Yes* to update its internal dimension names.

Updating the perspective will break any existing SAP Analytics Cloud stories that consume it, and you will need to reselect the dimension manually in each story.

Define Perspectives From the Data Preview

Procedure

1. In your consumption model, choose *Data Preview*.
2. The context menu **...** on a field has various options:
 - To remove fields from the output but keep it in the perspective: choose *Remove from Output*.
 - To remove or add fields: choose *Remove from Perspective* or *Add to Perspective*.
 - To sort your fields: choose *Sort*.
 - To define a filter on the fields: choose *Filter*.
 - To move a field to a different position: choose *Move*.
 - The *Association Context* shows you where the field comes from.
3. You can also add or remove fields via drag&drop.
4. To define a filter you can also choose **+** *Add Filter* the bar above the data preview. Here you can choose between the simple and the advanced editor.
5. You can also select more than one field by choosing **☰** *Multi Select* and then perform the action for multiple fields.
6. You can also remove a field from the output by choosing **👁** *Remove from Output*.
7. You can define settings for your perspective by choosing **⚙** *Settings*:
Define your preferences for the
 - authorizations: this is only available if an authorization scenario has been assigned. You can only assign one authorization scenario per perspective. Assigning it to a perspective allows the authorization scenario to become effective. More information: [Authorization Scenario \[page 368\]](#)
 - preview mode: you can choose between the *Validate* and the *Explore* mode.
 - refresh mode: the data can be refreshed automatically or manually.

- row limit: you can specify the number of rows to be retrieved.
 - attribute display: you can choose if the text, the technical name or the ID is displayed.
8. Save your perspective by choosing [Save New](#). Enter a title and save it.
 9. When you make changes to your perspective, you can save the changes or you can save this changed perspective as a new perspective by choosing [Save As](#). You can undo your changes by choosing [Reset](#).
 10. Switching to other perspectives: When you click on the title of your perspective, you can see the titles of the existing perspectives for this consumption model. From here, you can also open these perspectives.
 11. In order to make your perspective consumable, you need to deploy it. Go back to your consumption model, choose the [Perspectives](#) tab and choose [↻ Deploy](#). A perspective needs to be deployed in order to be used in a story.

Define Perspectives on the Perspectives Tab

Procedure

1. Go to the [Perspectives](#) tab and choose [+](#) ([New Perspective](#)) .
2. Enter a business purpose.
3. Enter an authorization scenario, if required.
4. Select your measures by choosing [+](#) ([New](#))
5. Select your attributes by choosing [+](#) ([New](#))
6. Select your hierarchies by choosing [+](#) ([New](#))
7. You can select [Run in Analytical Mode](#). When you select this option, the `USE_OLAP_PLAN` hint is sent to the SQL optimizer.

This may improve view performance, particularly if a union is performed. It is only available if [Expose for Consumption](#) is enabled.

For more information, see [HINT Details](#) in the *SAP HANA Cloud, SAP HANA Database SQL Reference Guide*.

8. Save your perspective by choosing [Save](#).
9. When you make changes to your perspective, you can save the changes or you can save this changed perspective as a new perspective by choosing [Save As](#). You can undo your changes by choosing [Reset](#).
10. In order to make your perspective consumable, you need to deploy it. Go back to the [Perspectives](#) tab and choose [↻ Deploy](#). A perspective needs to be deployed in order to be used in a story.

Results

The perspective can be found in the [Repository Explorer](#). You can use your perspective in stories in SAP Analytics Cloud.

Related Information

[Previewing Data in Business Builder Objects \[page 385\]](#)

8.7 Change the Data Source of a Business Entity

You can change the source of your business entity.

Context

For example, the source may have to be changed as your IT department provides you with more recent data.

Procedure

1. Go to the Business Builder. First, you need to select the space you would like to work in.
2. Select the business entity you need to change.
3. On the tab *General*, select your new source under *Dataset*.
4. Save your entries.
5. You may now have to adjust some settings of your business entity, due to a different structure of your new source. If that is the case, you get some error messages, which you have to resolve. The error message contains a link which takes you to the object in question. Make your changes and save your entries.

Results

In the story on top of the consumption model, which contains this business entity, you just need to refresh and you will see the new data.

8.8 Create Versions of an Object

You can create different versions of an object in the Business Builder.

Context

To guarantee a smooth roll-out of updates to any object without immediately affecting or even breaking fact models, consumption models, or perspectives, you can derive versions of objects.

The versions exist in parallel and are mostly independent from each other. The definition gets cloned once a new version is derived. As a consequence, changes in the new version have no impact on older versions. Older versions can still be adjusted, but likewise, changes to these are not automatically carried over to newer versions.

When several versions of an object exist, you are informed which properties are version-specific, which means that they apply only to the current version, e.g. the mapping of fields for the key definition, or if they are cross-version, which means that they are valid across all version, e.g. the title.

When you add an object to a model, a selection for the version is offered.

There is no activation process for versions which means that models are not auto-upgraded to a newly created version. Instead, you can decide whether and when a model adopts the new version.

Each version has its own life-cycle depicted by a status. This status in combination with the version's description indicates if a version is recommended for usage and signals the user to switch to another version. A wizard assists you in the migration process.

Procedure

1. Provide an easy-to-understand title for the new version.
2. Describe the updates that have been applied in comparison to the former version.
3. Select the status of the current version:
 - In Process: The object is being created.
 - In Validation: The object is being validated.
 - Ready to Use: The object is validated and ready-to-use.
 - Deprecated: The object is deprecated.
 - Discontinued: The object shall not be used any more. Nevertheless, it will be technically available as long as it is in use.

8.9 Previewing Data in Business Builder Objects

You can check the data in your models in the data preview.

Note

Users with the standard *DW Modeler* role can preview data in any object in their space. Users with the *DW Viewer* role can only preview data for fact models and consumption models. For more information, see [Roles and Privileges by App and Feature](#).

The data preview is an analytic preview. It is accessible from every model in the Business Builder, but only when you access it for a consumption model, you can save it as a perspective. The functions available are mainly the same as in the perspective.

A data preview is only possible if the underlying object in the Data Builder has been deployed.

When you open the data preview from a business entity or a fact model (only for single fact models), you can save the data preview as a consumption model. A consumption model and a perspective is generated with the attributes and measures chosen in the data preview. This can be used as a visual method of modeling.

Related Information

[Define Perspectives \[page 380\]](#)





8.10 Saving and Deploying Data Objects

When you save an object, it is stored in the SAP Datasphere repository, which contains the design-time definitions of all your objects. When you deploy it, you are creating a run-time version for use in the SAP Datasphere database.

Design-time objects do not contain any data, but you can preview the data they will contain when they are deployed to the run-time environment. You can save an object even if it contains validation errors.

In the Business Builder, all objects apart from the perspective are just saved.

To review the current status of a perspective, open it in its editor and look for the *Deployment Status* property, which can have the following values:


-  (*Not Deployed*) - The object has never been deployed and exists only as a design-time artifact.
-  (*Deployed*) - The object is deployed to the run-time database and its design-time and run-time versions are identical.
-  (*Changes to Deploy*) - The design-time version of the object contains changes that need to be deployed to make them available in the run-time.
-  (*Design-Time Error*) - The deployment process failed and you should try to deploy again.

When you deploy a perspective, you create a run-time version, which can be used by other run-time objects and consumed in stories in SAP Analytics Cloud, or any other BI client.

You can always save your changes but you can deploy only if your Space isn't locked. Your Space Administrator can unlock the Space.

The run-time database server might be unavailable under exceptional circumstances. In this case, a message strip is displayed informing you of the database status and the following features are disabled:

- [Import from Connection](#)
- [Deploy](#)
- [Preview Data](#)
- [Log Viewer](#)

You can solve this problem by waiting a few minutes and refreshing your web browser. If the problem persists, [open a support ticket](#) .

8.11 Importing SAP BW/4HANA Models

You can import existing analytic queries from SAP BW/4HANA into SAP Datasphere in order to build new models on top of them or enhance them.

Note

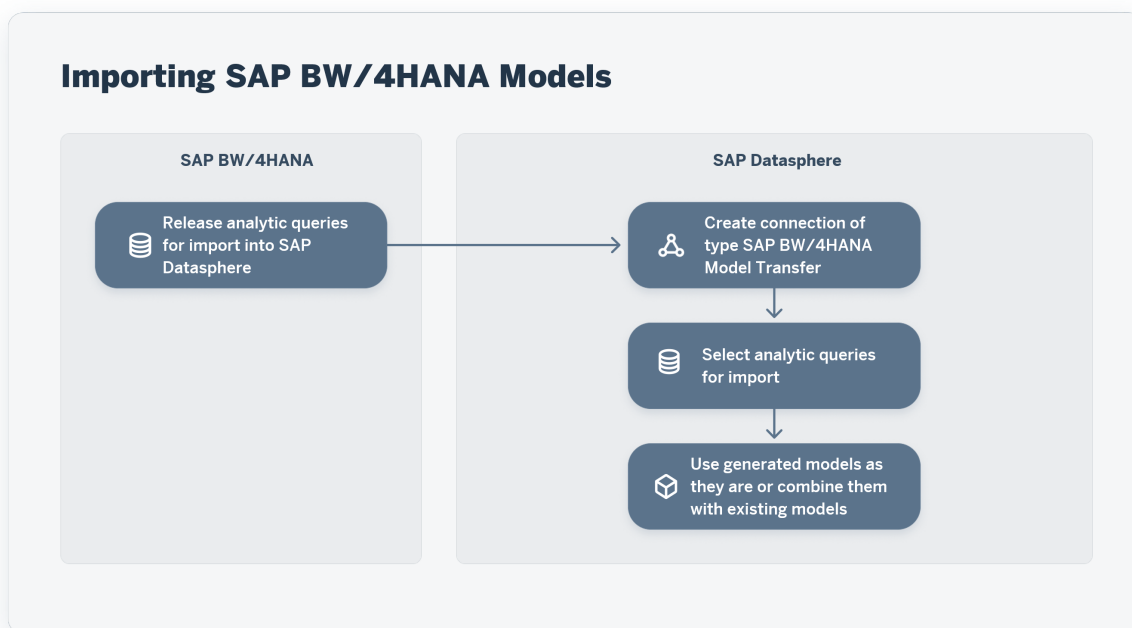
The import of SAP BW/4HANA models is now also integrated in the Import Entities wizard, which can be accessed from the Data Builder and the Repository Explorer. When you use the Import Entities wizard to import SAP BW/4HANA models, you can now also import InfoProviders besides analytic queries. Note that this is only supported for SAP BW/4HANA 2.0.

The import of SAP BW/4HANA models enables you to re-use existing metadata and data from SAP BW/4HANA systems so that you can include this metadata and data in SAP Datasphere without having to rebuild them manually. To enable interaction between SAP Datasphere and SAP BW/4HANA, we need to ensure that SAP Datasphere and SAP BW/4HANA understand each other. This requires a transformation of metadata of entities from SAP BW/4HANA to the definition of entities known in SAP Datasphere.

With the import, objects are generated in the Business Builder and the Data Builder. Data access to SAP BW/4HANA is realized through these objects. The data is retrieved directly from the SAP BW/4HANA system via remote tables.

You can then use these models as they are or enhance the data in SAP Datasphere on each level of your model - for example by adding master data to a view of type dimension.

The graphic shows the process flow for importing queries:



Prerequisites

You have a SAP BW/4HANA system installed.

In your SAP BW/4HANA system, you need to release the analytic queries for transferring into SAP Datasphere. For more information, see: [Releasing Analytic Queries for Use in SAP Datasphere](#)

In SAP Datasphere, You have created a connection of type *SAP BW/4HANA Model Transfer* to connect to your SAP BW/4HANA system. For more information, see [SAP BW/4HANA Model Transfer Connections](#).

Procedure

1. In the side navigation area, click *Business Builder*. Select a space if necessary.
2. Choose (*Import*) *SAP BW/4HANA* *Model Transfer Wizard*.
3. In the *Model Transfer Wizard*, search for and select the SAP BW/4HANA connection that you would like to use. Choose *Next*. The system will show you all available analytic queries.

Note

Only analytic queries that have been exposed in the source system for consumption in SAP Datasphere can be imported. For more information see [Releasing Analytic Queries for Use in SAP Datasphere](#).

4. Use the *Create Business Builder Objects* setting to control which objects to create:

- *Business Entities and Consumption Models* [default] - Create all possible *Data Builder* and *Business Builder* objects.
 - *Business Entities Only* - Create *Data Builder* objects and *Business Builder* business entities.
 - *None* - Create only *Data Builder* objects. Queries are imported as SAP Datasphere analytic models, which can be consumed directly in SAP Analytics Cloud. The analytic model inherits all the properties of the analytic query, like associations, global filter, restricted and calculated measure etc..
5. Select the analytic query that you would like to transfer. In the pane on the right, you will see the list of objects which will be generated in the *Business Builder* and in the *Data Builder*. The *Import Status* tells you if the objects already exist and will be overwritten or if they are new.

Note

When re-importing a model, changes in the column definitions of source entities that generate remote tables will automatically result in updating and redeploying the remote tables. This is only possible for remote tables that directly access data live in the source. Before doing the re-import, you need to stop snapshot and real-time replication for any affected remote tables. This is necessary because of the inconsistent changes in the source.

If there is a remote table that needs to be updated but has been set up for replication (snapshot or real-time replication), you will be notified and asked to stop replication first.

6. [optional] To import the analysis authorizations associated with the data in the query, click the *Import Permissions* button, and then click *OK*. Additional objects will be added to the list of objects to be imported on the *Data Builder/Data Access Controls* tab to create the data access control and apply it to the fact.

Note

If you do not import the analysis authorizations in this wizard, you can do it later in the *Data Access Controls* app (see [Import SAP BW and SAP BW/4HANA Analysis Authorizations](#)).

7. Choose *Import*.

Note

The system will generate notifications about the import process.

If the import was successful, the generated objects will appear in the Business Builder and Data Builder.

You can check this in detail by choosing [\(Import\) >> SAP BW/4HANA > Show Logs >](#).

8.11.1 Imported Objects

Overview of metadata mapping and the deployment status after model transfer

SAP BW/4HANA		SAP Datasphere	
Model	Sub-Model	Data Builder	Business Builder
InfoObject	Master Data	Remote table	
	Text	Remote table	
	InfoObject	View with prefix DL	Dimension
	InfoObject with text	Additional view with prefix TA	
CompositeProvider	Fact	Remote table	
	InfoProvider	View with prefix DL (with associations to dimension views) (fact)	Fact (with associations to dimensions)
Query	Query (Restricted Key Figure, Calculated Key Figure, Filter)	Analytic Model	Consumption Model
	Query (visible part of query for consumption)		Perspective for consumption in stories in SAP Analytics Cloud

9 SQL and SQLScript Reference

SAP Datasphere views use a subset of the SAP HANA Cloud SQL and SQLScript syntax.

9.1 SQL Reference

SAP Datasphere views support a subset of the SQL syntax supported by SAP HANA Cloud.

SAP Datasphere primarily supports the SQL syntax listed in the `SELECT` statement (see [SELECT Statement \(Data Manipulation\)](#)) and related sections of the *SAP HANA Cloud, SAP HANA Database SQL Reference Guide*.

SAP Datasphere supports the following:

- Operators (see [SQL Operators \[page 390\]](#))
- Predicates (see [SQL Predicates \[page 391\]](#))
- Expressions (see [Expressions \[page 392\]](#))
- Functions (see [SQL Functions Reference \[page 257\]](#))

→ Tip

An alternative to using `TOP`, which is not supported in SAP Datasphere is to use `limit`.

So, instead of using:

```
SELECT TOP 3 "SMALL_INT" FROM "ALLDATASV0" ORDER BY "SMALL_INT" DESC
```

You can use:

```
SELECT "SMALL_INT" FROM "ALLDATASV0" ORDER BY "SMALL_INT" DESC limit 3
```

SQL Operators

SAP Datasphere supports the following operators:

Operator	Description
+	Addition
-	Subtraction
*	Multiplication
/	Division

Operator	Description
AND	Logical conjunction
OR	Logical disjunction
NOT	Logical negation
	Concatenation
<	Less than
>	Greater than
=	Equal
!=	Not equal
<=	Less than or equal
>=	Greater than or equal
LIKE	Value similar to
IS NULL	Value does not exist
BETWEEN	Value between two other values
()	Parentheses

For more information, see [Operators](#) in the *SAP HANA Cloud, SAP HANA Database SQL Reference Guide*.

SQL Predicates

SAP Datasphere supports the following predicates:

Predicate	Description
ANY, SOME, ALL	Compares values using the specified comparison operator and returns true, false, or unknown.
BETWEEN	Compares a value with a list of values within the specified range and returns true or false.
CONTAINS	Matches a search string with the results of a subquery.
EXISTS	Tests for the presence of a value in a set and returns either true or false.
IN	Searches for a value in a set of values and returns true or false.
LIKE	Performs a comparison to see if a character string matches a specified pattern.
MEMBER OF	Determines whether a value is a member of an array.
NULL	Performs a comparison of the value of an expression with NULL.

For more information, see [Predicates](#) in the *SAP HANA Cloud, SAP HANA Database SQL Reference Guide*.

Expressions

SAP Datasphere supports the `CASE` expression:

Expression	Description
<code>CASE WHEN THEN ELSE END</code>	Provides if, then, else logic.

For more information, see [Expressions](#) in the *SAP HANA Cloud, SAP HANA Database SQL Reference Guide*.

9.2 SQL Functions Reference

SAP Datasphere views support a subset of the SQL functions supported by SAP HANA Cloud.

This topic contains the following sections:

- [Array Functions \[page 392\]](#)
- [Data Type Conversion Functions \[page 392\]](#)
- [Datetime Functions \[page 393\]](#)
- [Fulltext Functions \[page 394\]](#)
- [Miscellaneous Functions \[page 395\]](#)
- [Numeric Functions \[page 395\]](#)
- [Security Functions \[page 396\]](#)
- [String Functions \[page 396\]](#)
- [Window Functions \[page 397\]](#)
- [Example: Converting Currency Values with CONVERT_CURRENCY \[page 398\]](#)

Array Functions

SAP Datasphere supports the following array functions, which take arrays as input:

- `SUBARRAY`
- `TRIM_ARRAY`

For detailed documentation of these functions, see [Array Functions](#) in the *SAP HANA Cloud, SAP HANA Database SQL Reference Guide*.

Data Type Conversion Functions

SAP Datasphere supports the following data type conversion functions, which convert data from one data type to another:

- CAST
- TO_BIGINT
- TO_BINARY
- TO_BLOB
- TO_BOOLEAN
- TO_CLOB
- TO_DATE
- TO_DATS
- TO_DECIMAL (DECFLOAT)
- TO_DOUBLE (DOUBLE)
- TO_FIXEDCHAR
- TO_INT (INT)
- TO_INTEGER (INT)
- TO_NCLOB
- TO_NVARCHAR
- TO_REAL (FLOAT)
- TO_SECONDSDATE
- TO_SMALLDECIMAL
- TO_SMALLINT
- TO_TIME
- TO_TIMESTAMP
- TO_TINYINT
- TO_VARBINARY
- TO_VARCHAR

For detailed documentation of these functions, see [Data Type Conversion Functions](#) in the *SAP HANA Cloud, SAP HANA Database SQL Reference Guide*.

Datetime Functions

SAP Datasphere supports the following datetime functions, which perform operations on date and time data types or return date or time information:

- ADD_DAYS
- ADD_MONTHS
- ADD_MONTHS_LAST
- ADD_NANO100
- ADD_SECONDS
- ADD_WORKDAYS
- ADD_YEARS
- CURRENT_DATE
- CURRENT_DATE ()

- CURRENT_TIME
- CURRENT_TIME ()
- CURRENT_TIMESTAMP
- CURRENT_TIMESTAMP ()
- DAYNAME
- DAYOFMONTH
- DAYOFYEAR
- DAYS_BETWEEN
- HOUR
- ISOWEEK
- LAST_DAY
- LOCALTOUTC
- MINUTE
- MONTH
- MONTHNAME
- MONTHS_BETWEEN
- NEXT_DAY
- NOW
- QUARTER
- SECOND
- SECONDS_BETWEEN
- UTCTOLOCAL
- WEEK
- WEEKDAY
- WORKDAYS_BETWEEN
- YEAR
- YEARS_BETWEEN

For detailed documentation of these functions, see [Datetime Functions](#) in the *SAP HANA Cloud, SAP HANA Database SQL Reference Guide*.

Fulltext Functions

SAP Datasphere supports the following fulltext functions, which perform operations on data that has a fulltext index:

- SCORE

For detailed documentation of this function, see [SCORE Function](#) in the *SAP HANA Cloud, SAP HANA Database SQL Reference Guide*.

Miscellaneous Functions

SAP Datasphere supports the following miscellaneous functions, which return system values and perform various operations on values, expressions, and return values of other functions:

- COALESCE
- CONVERT_CURRENCY - See [Example: Converting Currency Values with CONVERT_CURRENCY \[page 398\]](#)
- CONVERT_UNIT
- GREATEST (MAX)
- HASH_SHA256
- IFNULL
- LEAST (MIN)
- SESSION_CONTEXT
- SESSION_USER
- SYSUUID
- WIDTH_BUCKET
- XMLEXTRACT
- XMLEXTRACTVALUE

For detailed documentation of these functions, see [Miscellaneous Functions](#) in the *SAP HANA Cloud, SAP HANA Database SQL Reference Guide*.

Numeric Functions

SAP Datasphere supports the following numeric functions, which perform mathematical operations on numerical data types or return numeric information:

- ABS
- ACOS
- ASIN
- ATAN
- ATAN2
- BITAND
- BITCOUNT
- BITNOT
- BITOR
- BITXOR
- CEIL
- COS
- COSH
- COT
- EXP
- FLOOR

- LN (LOG)
- LOG (LOG10)
- MOD (%)
- NDIV0
- POWER (**)
- RAND
- ROUND
- SIGN
- SIN
- SINH
- SQRT
- TAN
- TANH
- UMINUS

For detailed documentation of these functions, see [Numeric Functions](#) in the *SAP HANA Cloud, SAP HANA Database SQL Reference Guide*.

Security Functions

SAP Datasphere supports the following security functions, which provide special functionality for security purposes:

- ESCAPE_DOUBLE_QUOTES
- ESCAPE_SINGLE_QUOTES
- IS_SQL_INJECTION_SAFE

For detailed documentation of these functions, see [Security Functions](#) in the *SAP HANA Cloud, SAP HANA Database SQL Reference Guide*.

String Functions

SAP Datasphere supports the following string functions, which perform extraction and manipulation on strings, or return information about strings:

- ABAP_ALPHANUM
- ABAP_LOWER
- ABAP_NUMC
- ABAP_UPPER
- ASCII
- BINTOHEX
- BINTOSTR
- CHAR

- CONCAT
- HEXTOBIN
- LCASE
- LEFT
- LENGTH
- LOCATE
- LOWER
- LPAD
- LTRIM
- NCHAR
- REPLACE
- REPLACE_REGEXPR
- RIGHT
- RPAD
- RTRIM
- SOUNDEX
- STRTOBIN
- SUBSTR_AFTER
- SUBSTR_BEFORE
- SUBSTRING
- UCASE
- UNICODE
- UPPER

For detailed documentation, see [String Functions](#) in the *SAP HANA Cloud, SAP HANA Database SQL Reference Guide*.

Window Functions

SAP Datasphere supports the following window and window aggregation functions, which allow you to perform analytic operations over a set of input rows:

- AVG
- BINNING
- CORR
- CORR_SPEARMAN
- COUNT
- CUBIC_SPLINE_APPROX
- CUME_DIST
- DENSE_RANK
- FIRST_VALUE
- LAG

- LAST_VALUE
- LEAD
- MAX
- MEDIAN
- MIN
- NTH_VALUE
- NTILE
- PERCENT_RANK
- RANDOM_PARTITION
- RANK
- ROW_NUMBER
- SERIES_FILTER
- STDDEV
- SUM
- VAR
- WEIGHTED_AVG

Note

In the graphical view editor, window functions can only be used in a *Calculated Columns* node (and not in *Filter* or *Aggregation* nodes).

The following types of functions and syntaxes are not supported in SAP Datasphere views:

- Inverse distribution functions
- COLLATE
- SERIES
- WITHIN GROUP

For detailed documentation of these functions, see [Window Functions and the Window Specification](#) in the *SAP HANA Cloud, SAP HANA Database SQL Reference Guide*.

Example: Converting Currency Values with `CONVERT_CURRENCY`

When using the `CONVERT_CURRENCY` function you must, as a minimum, complete the following parameters:

- `AMOUNT` - Specify the column containing the currency value to be converted
- `CLIENT` - Specify the three character string identifying the tenant in your SAP system
- `SOURCE_UNIT` - Specify the column containing the source currency or enter the source currency code
- `TARGET_UNIT` - Specify the column containing the target currency or enter the target currency code
- `REFERENCE_DATE` - Specify the date to use when obtaining the conversion rate
- `SCHEMA` - Specify the space name.

In our example, which is in a space named `SALES`, we are converting the value in the `sale_Amount` column from Euros to US Dollars using the conversion rate from 30 September 2021:

```
CONVERT_CURRENCY (
```

```
"AMOUNT" => "Sale_Amount",  
"SOURCE_UNIT" => 'EUR',  
"TARGET_UNIT" => 'USD',  
"SCHEMA" => 'SALES',  
"REFERENCE_DATE" => '2021-09-30',  
"CLIENT" => '000'  
)
```

For full documentation of this function, see [CONVERT_CURRENCY](#) in the *SAP HANA Cloud, SAP HANA Database SQL Reference Guide*.

9.3 SQLScript Reference

When you set the SAP Datasphere SQL view editor *Language* property to *SQLScript (Table Function)*, it creates an SAP HANA Cloud table user-defined function.



SAP Datasphere supports the SQLScript syntax documented for table user-defined functions in [User-Defined Functions](#) and associated sections of the [SAP HANA Cloud, SAP HANA SQLScript Reference](#).

Important Disclaimers and Legal Information

Hyperlinks

Some links are classified by an icon and/or a mouseover text. These links provide additional information.

About the icons:

- Links with the icon : You are entering a Web site that is not hosted by SAP. By using such links, you agree (unless expressly stated otherwise in your agreements with SAP) to this:
 - The content of the linked-to site is not SAP documentation. You may not infer any product claims against SAP based on this information.
 - SAP does not agree or disagree with the content on the linked-to site, nor does SAP warrant the availability and correctness. SAP shall not be liable for any damages caused by the use of such content unless damages have been caused by SAP's gross negligence or willful misconduct.
- Links with the icon : You are leaving the documentation for that particular SAP product or service and are entering an SAP-hosted Web site. By using such links, you agree that (unless expressly stated otherwise in your agreements with SAP) you may not infer any product claims against SAP based on this information.

Videos Hosted on External Platforms

Some videos may point to third-party video hosting platforms. SAP cannot guarantee the future availability of videos stored on these platforms. Furthermore, any advertisements or other content hosted on these platforms (for example, suggested videos or by navigating to other videos hosted on the same site), are not within the control or responsibility of SAP.

Beta and Other Experimental Features

Experimental features are not part of the officially delivered scope that SAP guarantees for future releases. This means that experimental features may be changed by SAP at any time for any reason without notice. Experimental features are not for productive use. You may not demonstrate, test, examine, evaluate or otherwise use the experimental features in a live operating environment or with data that has not been sufficiently backed up.

The purpose of experimental features is to get feedback early on, allowing customers and partners to influence the future product accordingly. By providing your feedback (e.g. in the SAP Community), you accept that intellectual property rights of the contributions or derivative works shall remain the exclusive property of SAP.

Example Code

Any software coding and/or code snippets are examples. They are not for productive use. The example code is only intended to better explain and visualize the syntax and phrasing rules. SAP does not warrant the correctness and completeness of the example code. SAP shall not be liable for errors or damages caused by the use of example code unless damages have been caused by SAP's gross negligence or willful misconduct.

Bias-Free Language

SAP supports a culture of diversity and inclusion. Whenever possible, we use unbiased language in our documentation to refer to people of all cultures, ethnicities, genders, and abilities.

© 2024 SAP SE or an SAP affiliate company. All rights reserved.

No part of this publication may be reproduced or transmitted in any form or for any purpose without the express permission of SAP SE or an SAP affiliate company. The information contained herein may be changed without prior notice.

Some software products marketed by SAP SE and its distributors contain proprietary software components of other software vendors. National product specifications may vary.

These materials are provided by SAP SE or an SAP affiliate company for informational purposes only, without representation or warranty of any kind, and SAP or its affiliated companies shall not be liable for errors or omissions with respect to the materials. The only warranties for SAP or SAP affiliate company products and services are those that are set forth in the express warranty statements accompanying such products and services, if any. Nothing herein should be construed as constituting an additional warranty.

SAP and other SAP products and services mentioned herein as well as their respective logos are trademarks or registered trademarks of SAP SE (or an SAP affiliate company) in Germany and other countries. All other product and service names mentioned are the trademarks of their respective companies.

Please see <https://www.sap.com/about/legal/trademark.html> for additional trademark information and notices.