TOC

Composer 24

Connect to Data in Composer 24

Connect Composer to Data Stores

Using connectors, you can connect to a wide array of data stores, from modern databases such as Hadoop, Search, Streaming, and NoSQL, to traditional sources like SQL-based stores. By default, your environment comes preconfigured with certain connectors enabled. However, additional connectors are available. If the connector you are looking for is not shown, it may be because:

- The connector is installed but is not enabled. For more information, see Manage Connectors And Connector Servers.
- The connector was not installed and needs to be downloaded separately. For more information, see Obtain Additional Connector Servers.
- The connector requires you to provide a licensed JDBC driver. For more information, see Add A JDBC Driver.

Certain connectors require a JDBC driver, which is obtained through a separate download. This allows you to select and add a driver that meets your operation needs or policies.

To connect to a data store and use its data in a Composer visual, you must first verify that a connector and its connector server have been defined in Composer for the data store. See Manage Connectors And Connector Servers. Then you must define a data store connection and a data source configuration that uses the connection. The data store connection can be defined while you are setting up the data source configuration. See Manage Data Store Connections and Manage Data Sources for more information.

Note: Composer supports only underscores and dashes in data store field names. No other special characters or white space are supported. If your data store uses special characters other than underscores and dashes in field names, please remove them before attempting to create a data source configuration.

For information about what versions of data stores are supported by the Composer connectors, see Data Connector Reference. For information about which features are supported by different connectors, see Connector Feature Support.

Manage Connectors and Connector Servers

Composer connects to a wide array of data sources available in the marketplace today–from modern databases (including Hadoop, Search, Streaming, and NoSQL) to traditional sources like SQL-based stores. Composer comes prepackaged with connectors that are automatically installed during the Composer installation process.

JDBC drivers for a few of the connectors are no longer included in the installation package. This means that if you use one of these connectors, you also need to configure a JDBC driver before it can be enabled and accessible from Composer. For additional details, including the list of connectors, see Add A JDBC Driver.

If the connector you are looking for is not shown, it may be because:

- The connector is installed but is not enabled. For more information, see Enable And Disable Connectors.
- The connector was not installed and needs to be downloaded separately. For more information, see Obtain Additional Connector Servers.
- The connector requires you to provide a licensed JDBC driver. For more information, see Add A JDBC Driver.

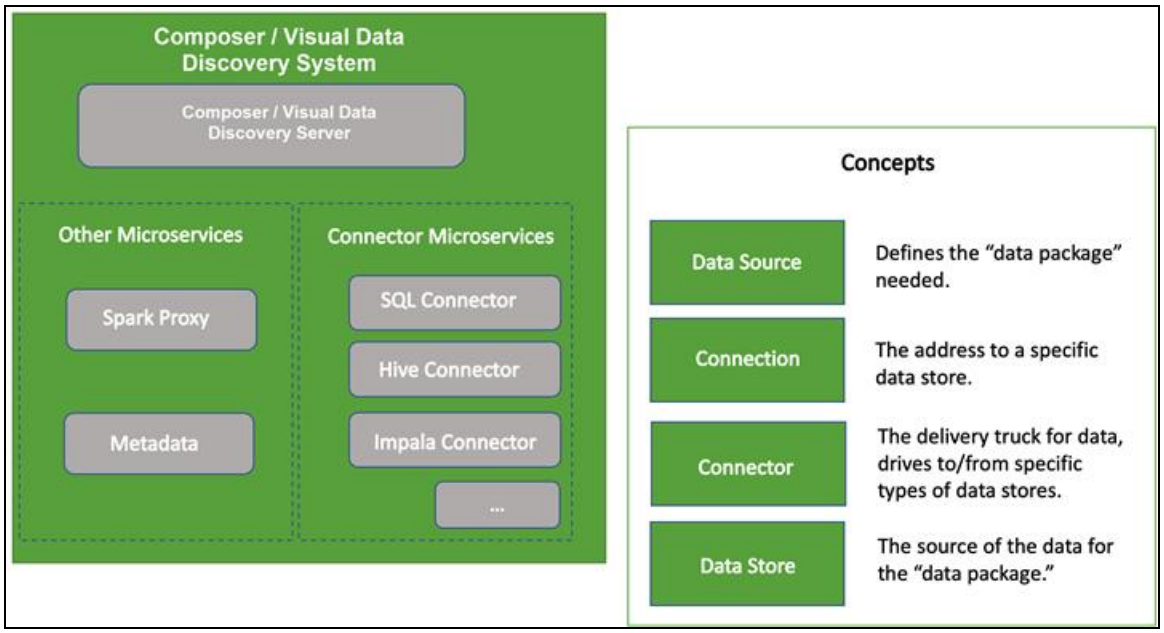
See Data Connector Reference for the full list of supported Composer connectors.

Management of connector microservices is split into two sections:

- Connector Servers
- Connectors

Each connector server runs independently in the Composer environment. You can set up a connection type for each connector server and manage the ones to be available to users in the Composer account.

This means that you are able to enable or disable any of these servers at any time, depending on the data stores that you use and need to use with Composer. The following figure provides a high level concept diagram of the Composer environment.



The setup and management of connector servers in the Composer environment is handled on the Manage Connector Services page, which is accessible to supervisors. To make the actual connection between Composer and your data source after the connector server has been configured, log into Composer as an administrator and access the Sources page, accessible from the UI menu. See Connect Composer To Data Stores.

The Manage Connector Services page (available to supervisors) lets you register or remove connector servers that are not available out-of-the-box in the Composer instance. You can also use this page to maintain, enable, and disable connector definitions based on the connector servers defined in the Composer instance. See Manage Connector Services Page.

The Connections page (available to administrators and users with appropriate privileges) lets you define the connection for a connector between Composer and a data store. See Connect Composer To Data Stores.

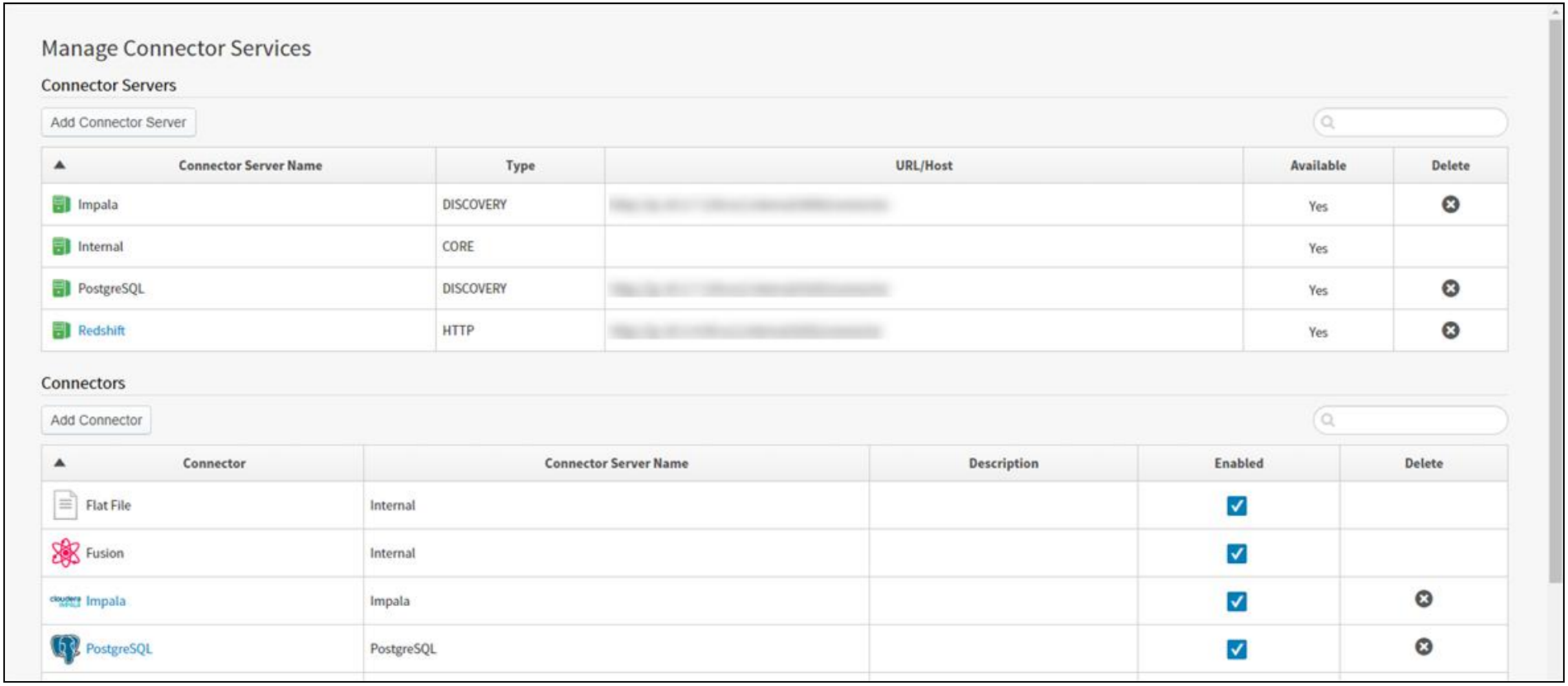
Note: If you try to delete a visual, filter snippet, dashboard, dashboard link, source, or source field, Composer displays an error message naming any objects dependent on the item you're trying to delete. You can delete the item after you've removed the association from the dependent object. See Fields Usage.

See the following topics:

- Obtain Additional Connector Servers
- Register A New Connector Server

- Modifying A Connector Server
- Delete A Connector Server
- Define A New Connector
- Modify A Connector
- Enable And Disable Connectors
- Delete A Connector
- Connector Graceful Shutdown

Manage Connector Services Page



The Manage Connector Services page is where you register a new connector server that is not available out-of-the-box to the Composer instance. It is also where you can remove connector servers available to the instance.

This page is also where you set up and enable connectors for use in your data sources. The connectors you can define depend on which connector servers have been registered. You can also use this page to delete and disable connectors in the Composer instance.

Connector Servers

The Connector Servers section lets the Composer admins and supervisors group users register and delete the connector servers available in the Composer environment. See Obtain Additional Connector Servers for information about setting up a connector server.

Note: The default **supervisor** user is no longer installed; add users to the **Supervisors** group instead.

Use the search box above the Connector Servers table to locate a connector server (for example, if your Composer environment contains a large number of connector servers).

Connector servers that you register and connect in Composer are accessible and the registration details can be edited (for example, if the server has moved or changed). Default connector servers provided by Composer are not editable.

The Connector Server table provides the following information.

Column Title	Description
Connector Server Name	The name of the connector server (data store) definition.
Type	The type of connector server. The following types are supported: ■ Discovery: uses the capability integrated into Composer to locate and set up the connector server automatically. ■ HTTP/Socket: if you manually add a connector server to the Composer environment, then it will be either an HTTP or a Socket type. ■ Core: identifies connectors that are built into the Composer server. **Note:** Type **Core** connectors , including flat files as well as HDFS and S3 buckets, do not require a dedicated connector server. These types of connectors are always available. This means they are always on, and do not require any additional network resources to keep them on.
URL/Host	The URL or host name of the connector server (data store). If more than one instance of a connector server is running, the URLs for each instance are shown, separated by commas.
Available	Indicates whether the connector server (data store) is available or not.
Delete	Provides an option to delete a connector server definition from the Composer instance. This option is not available for Core type connector servers because they are built into Composer and cannot be deleted. You can delete a connector server that you manually configured. However, you must first delete the connection definitions for the server and the data source configurations that use the connection definitions (see Delete A Data Source Configuration and Delete Data Store Connections). Then you need to delete the connector (see Delete A Connector). When the connections and the connector are all deleted, you can then delete the connector server.

See the following topics:

- Obtain Additional Connector Servers
- Register A New Connector Server
- Modifying A Connector Server
- Delete A Connector Server

Connectors

The Connectors section lists the connectors that are defined in the Composer environment. You can use these connectors to connect to a specific type of data store (such as Impala or Elasticsearch). You can use this section to add and remove connectors and to enable and disable them. A connector that is listed in this table and is enabled is visible on the Connections page in the UI (when you are logged in as a non-supervisory user or administrator and have been assigned appropriate privileges).

The Connectors table provides the following information.

Column Title	Description
Connector	The name of the connector definition.
Connector Server Name	The name of the connector server (data store) associated with the connector. If the connector server is a Core-type connector, the connector server name shows as **Internal**.
Description	An optional description of the connector. You can provide this description when you add a connector definition.
Enabled	A switch that enables or disables the connector definition in the Composer instance. An enabled connector is visible on the Connections page; a disabled connector is not.
Delete	Provides an option to delete the connector definition from the Composer instance. This option is not available for Internal connectors (Core type connector servers) because they are built into Composer and cannot be deleted. You can delete a connector that you manually configured. However, you must first delete the connection definitions for the server and the data source configurations that use the connection definitions. See Delete A Data Source Configuration and Delete Data Store Connections.

See the following topics:

- Define A New Connector
- Modify A Connector

- Delete A Connector
- Enable And Disable Connectors

Obtain Additional Connector Servers

Several data connector servers are installed by default in your Composer environment, but a number of other connector servers are also available. The new connector server can be added via HTTP or Socket protocols. For a complete list of all supported connector servers, see Data Connector Reference.

Obtain one or more of these connector servers

1. Contact Composer Technical Support. You will receive a `yum` or `apt` command that downloads and sets up the connector server in the Composer environment. After running the command, the connector microservice is installed and enabled in the Composer environment, allowing it to be discoverable as a new connector microservice.

2. Register the connector server. Connector servers are started and run as separate processes, and accept requests on a specific TCP/IP port. See Register A New Connector Server.

After you have obtained, installed, and registered the connector server, you need to add and enable at least one connector for it. See Define A New Connector and Enable And Disable Connectors.

Register a New Connector Server

Before you can register a new connector server, be sure that you have obtained and installed it. See Obtain Additional Connector Servers and contact insightsoftware Technical Support to obtain the connector server code.

Connector servers are started and run as separate processes, and accept requests on a specific TCP/IP port.

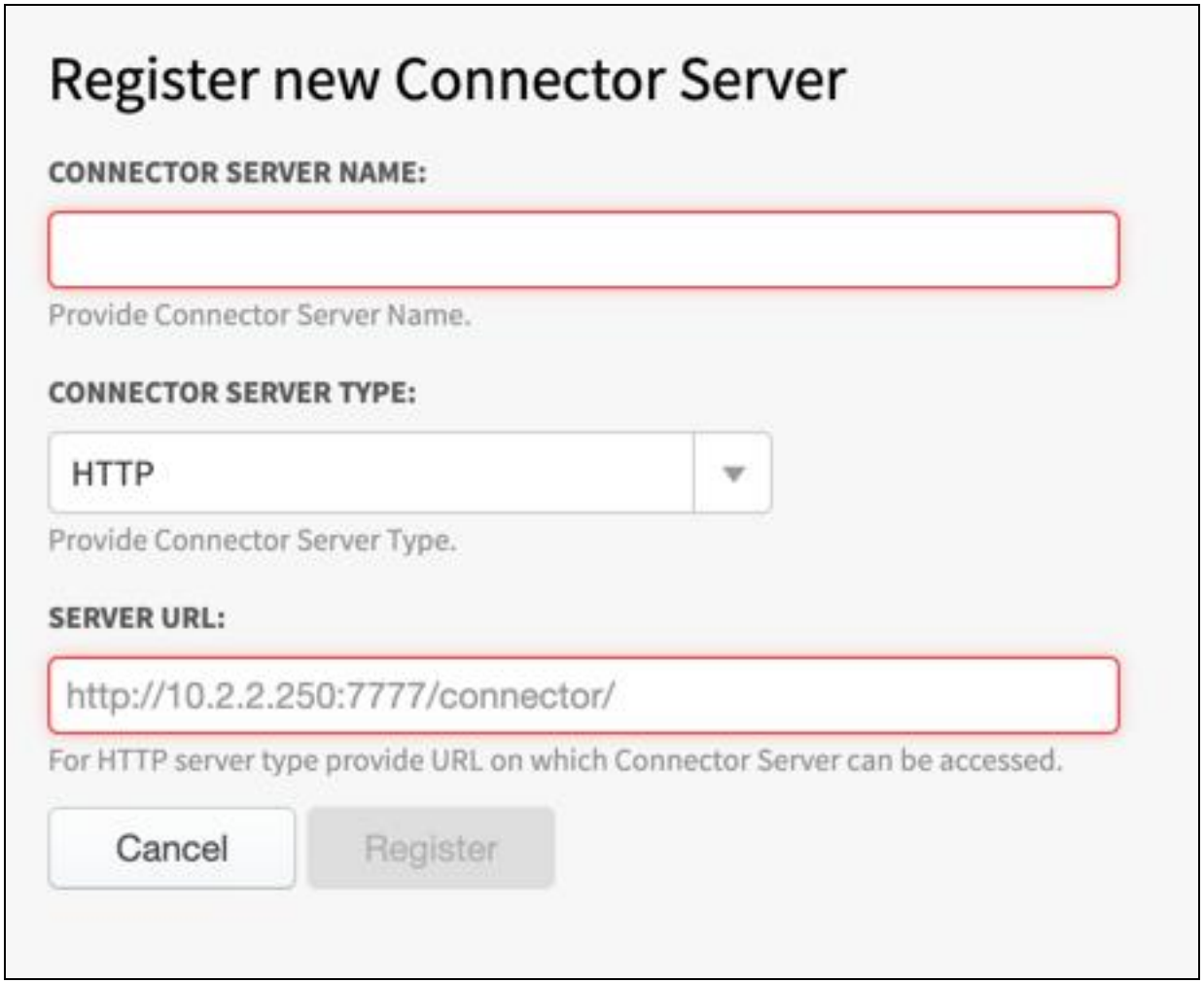
Register a new connector server in your Composer environment

1. Log in as a system admin or a member of the Supervisors group.

   Note: The default **supervisor** user is no longer installed; add users to the **Supervisors** group instead.

2. Select **Connectors** from the menu. The Managed Connector Services work area opens.

3. In the Connector Servers section of the Manage Connector Services page, select **Add Connector Server**. The Register New Connector Server page appears.



4. On the Register New Connector Server page, specify the following information in the input boxes.

Input Box	Description
Connector Server Name	Specify a unique name for the new connector server.
Connector Server Type	The new connector server can be added using HTTP or Socket protocols. Select either **HTTP** or **Socket** from drop-down menu.

Input Box	Description
Server URL	If you selected the HTTP protocol, specify the URL for the connector server. If you selected the socket protocol, specify the host and port details. For a list of default Composer ports, see Default Port Reference.

5. Select **Register**.

After the connector server is registered, add and enable at least one connector for it. See Define A New Connector and Enable And Disable Connectors.

Modifying a Connector Server

The only way to modify a connector server on the Manage Connector Services page is to delete and re-register the connector server. See Delete A Connector Server, Obtain Additional Connector Servers, and Register A New Connector Server.

Delete a Connector Server

You can delete a connector server that you manually configured. However, before deleting it, you must first delete the connections to the server from the Sources page. Then you need to delete the connector (at the bottom of the Manage Connector Services page). When the connections and the connector are deleted, you can delete the connector server.

Delete a connector server from yourComposer environment

1. Log in as a system admin or a member of the Supervisors group.

   Note: The default **supervisor** user is no longer installed; add users to the **Supervisors** group instead.

2. Select **Connectors** from the menu. The Managed Connector Services work area opens.

3. On the **Manage Connector Services** page, locate the connector server you want to delete in the Connector Servers table.

4. Select  for the connector server.

5. The connector server is removed from the Composer instance.

Note: If you try to delete a visual, filter snippet, dashboard, dashboard link, source, or source field, Composer displays an error message naming any objects dependent on the item you're trying to delete. You can delete the item after you've removed the association from the dependent object. See Fields Usage.

Define a New Connector

Connector definitions make a connector server available to authorized users on the Sources page. They make it possible for you to connect to a specific type of data store (such as Impala or Elasticsearch). More than one connector definition can be created for a connector server.

Define a new connector in the Composer environment

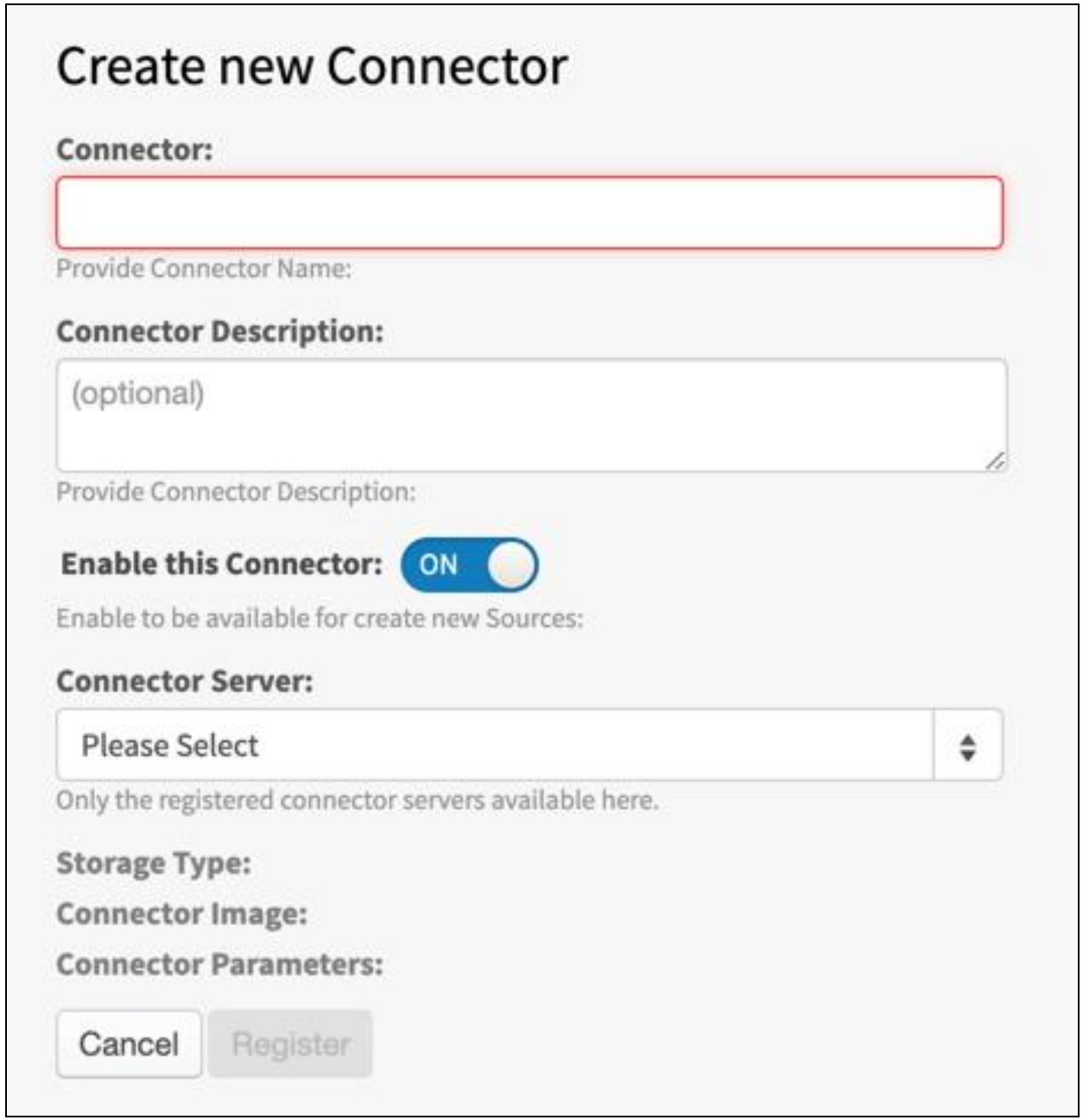
1. Log in as a system admin or a member of the Supervisors group.

   Note: The default **supervisor** user is no longer installed; add users to the **Supervisors** group instead.

2. Select **Connectors** from the menu. The Managed Connector Services work area opens.

3. In the Connector Servers section of the Manage Connector Services page, verify that a connector server has been registered for the type of connector you want to define. If it has not, register one. See Obtain Additional Connector Servers and Register A New Connector Server.

4. In the Connectors section of the Manage Connector Services page, select **Add Connector Server** The Create New Connector page appears.


Create new Connector
Connector:
Provide Connector Name:
Connector Description:
(optional)
Provide Connector Description:
Enable this Connector:
ON
Enable to be available for create new Sources:
Connector Server:
Please Select
Only the registered connector servers available here.
Storage Type:
Connector Image:
Connector Parameters:
Cancel
Register

5. On the Create New Connector page, specify the following information in the input boxes.

Input Box	Description
Connector	Specify a unique name for the new connector. A short name is recommended due to the limited space on the Sources page to display the icon and name.
Connector Description	Optionally, specify a description of the connector.
Enable this Connector	Slide this switch on or off to enable or disable the connector on the Sources page.
Connector Server	Select the connector server definition that should be used for this connector.
Storage Type	This value is automatically set by Composer after you have selected the connector server.
Connector Image	The default image is retrieved from the connector server. If you want to change the image, select **Choose File** to select a custom icon for the connection type. The requirements for the icon are as follows: ■ PNG or SVG format ■ Resolution (min/max): 72 x 72 px or 160 x 160 px ■ Max file size: 50 Kb Select **Restore Default** to restore the image to the default image for the connector server type.
Connector Parameters	The connector parameter information is generated from the selected connector server. Configure or customize the connection parameters, as needed. These are the values that appear on the Connections page or Connection tab when a user defines a new connection or a new data source configuration or edits an existing one. The following fields help you customize the parameters to meet your needs: ■ **Order:** Use the arrows to move the parameters up or down and change the order in which the parameters are listed on the Connections page (connection definition) or Connection tab (data source configuration). ■ **Required:** Select this checkbox to make the parameter required for validating the connection. Clear the checkbox if the parameter is not required. ■ **Visible:** Select this checkbox to show optional parameters on the Connections page or Connection tab. Clear the checkbox to hide the parameter. ■ **User Attribute:** Select this checkbox to use custom attributes for the parameter when you set up connections using this

Input Box	Description
	connector. See Use User Attributes For Connection Parameters. ■ **Parameter:** The internal parameter name. No changes can be made. ■ **Parameter Type:** The type of parameter. No changes can be made. ■ **Label:** Specify the label for the parameter. This is the field name that will be used for the parameter on the Connections page or Connection tab. ■ **Help Text:** Provide help text for the parameter. It will be displayed when you select  associated with the input box for the parameter on the Connections page or Connection tab. Select **Restore Default** to restore the connector parameters to the defaults for the connector server type.

6. When you have made the required changes, select **Register**. The new connector displays in the **Connector** section of the **Manage Connector Services** page. If you enabled it, it is also visible on the Sources page where you maintain data source configurations.

Modify a Connector

Modify a connector definition in yourComposer environment

1. Log in as a system admin or a member of the Supervisors group.

   Note: The default **supervisor** user is no longer installed; add users to the **Supervisors** group instead.

2. Select **Connectors** from the menu. The Managed Connector Services work area opens.

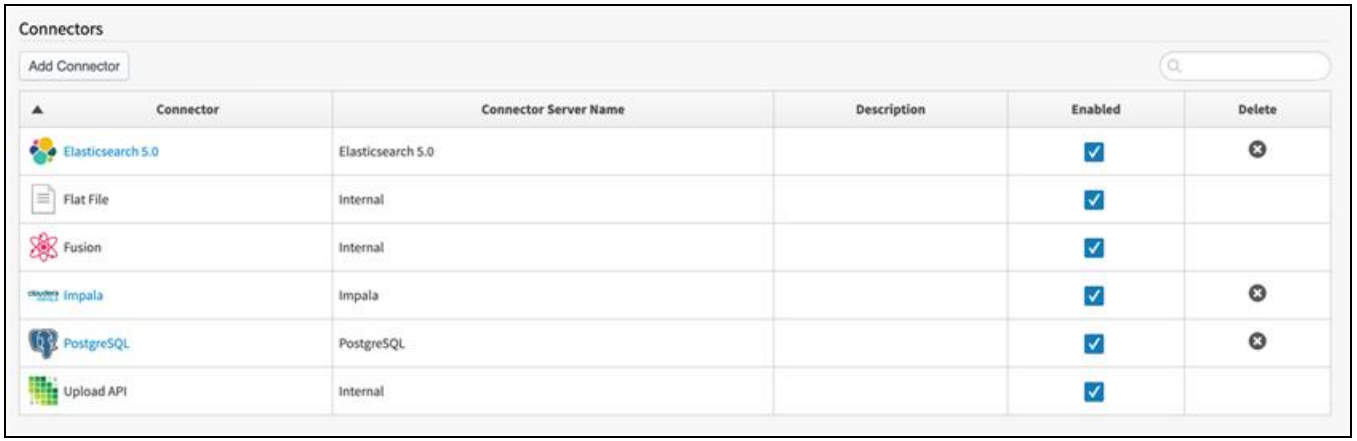
3. In the Connectors section of the Manage Connector Services page, locate the connector you want to modify and select its name. The Edit <connector-name > Connector page appears.

4. On the Edit <connector-name> Connector page, change the connector settings. These settings are described in Define A New Connector.

5. When you have made the required changes, select **Save**. The new connector changes are saved and display, as appropriate, in the rest of the UI.

Enable and Disable Connectors

A connector can be enabled or disabled at any time. Selecting or clearing the checkbox in the **Enabled** column for the target connector determines whether its representative icon appears on the Data Source page. The switch is instantaneous, removing or adding the icon on the Sources page. Existing connected data sources can still be accessed and used.



Delete a Connector

Delete a connector from yourComposer environment

1. Log in as a system admin or a member of the Supervisors group.

   Note: The default **supervisor** user is no longer installed; add users to the **Supervisors** group instead.

2. Select **Connectors** from the menu. The Managed Connector Services work area opens.

3. In the Connectors section of the Manage Connector Services page, locate the connector you want to delete,

4. Select  in the Delete column for the connector.

   The connector is removed from the Composer instance.

Note: If you try to delete a visual, filter snippet, dashboard, dashboard link, source, or source field, Composer displays an error message naming any objects dependent on the item you're trying to delete. You can delete the item after you've removed the association from the dependent object. See Fields Usage.

Connector Graceful Shutdown

Composer supports the graceful shutdown of connectors. When a connector is shut down, it gracefully completes queries that are in-flight and notifies clients that it is terminating.

Three connector properties in each connector property file support graceful shutdown:

- `connector.graceful.shutdown.enabled` indicates whether or not graceful shutdown processing should occur. Valid values are `true` (perform graceful shutdown processing) or `false` (do not perform graceful shutdown processing). The default is `true`.
- `connector.graceful.shutdown.event.propagation.timeout-sec` specifies how long (in seconds) the connector will wait to allow clients to receive the information that it is out of service. The default is 35 seconds.
- `connector.graceful.shutdown.force.kill.timeout-sec` specifies the maximum number of seconds that the connector will wait for the number of its active tasks to reach zero. The default is 30 seconds. When this time has elapsed, the connector will stop.

Manage Data Store Connections

Data store connections define the connection strings and options necessary to connect to a data store. You can manually add and maintain them. Saved and validated data store connections can be used in sources. Data store connections must be validated when they are defined. If they are not validated, they cannot be saved.

Note: You must be logged in as an administrator or as a user with the group privilege **Manage Connections** to manage data store connection definitions.

To manage existing data store connections or add new ones, select **Connections** on the top-level navigation banner or the UI menu or select the **Connections** box on the Home page. The Connections page appears.

Review the following links for information on managing your data store connection definitions.

- Connections Page
- List Data Store Connections
- Search And Filter Lists
- Add Data Store Connections
- Modify Data Store Connections
- Delete Data Store Connections
- Select A Connection For Source
- Use User Attributes For Connection Parameters
- Insert Variables For Connection Parameters

Connections Page

The **Connections** page allows you to review and maintain the connection definitions used by Composer connectors.

Note: You must be logged in as a user with the group privilege **Manage Connections** to see the Connections page.

Access the Connections page

1. Log in as an administrator or as a user with the **Manage Connections** privilege.
2. Select **Connections** on the top-level navigation banner or the UI menu ( ) or select the **Connections** box on the Home page. The Connections page appears.

The Connections page shows a table listing all the connection definitions available. Several of these columns can be used to sort the list: select the column header to sort first to last and again to sort last to first. You can search for items by the contents of several columns. See Search and Filter Lists.

Column Name	Description
Type	The logo showing the data store type to which the connection definition pertains.
Name	The name of the connection definition.
Author	The name of the user who defined the connection definition.
Modified Date	The date the connection definition was last modified.
Sources	The number of defined data sources that use the connection definition.
Actions	Allows you to delete the connection definition.

Select **Create Connection** to add a new connection definition.

If there are many connections listed, you may need to search for the connection you need. Use the search bar to search for a connection in the list. See Search And Filter Lists.

List Data Store Connections

You can list data store connection definitions on the Connections page in the UI.

Note: You must be logged in as a user with the group privilege **Manage Connections** to maintain data store connection definitions.

List your data store connection definitions:

1. Log in as a user with the group privilege **Manage Connections**.
2. Select **Connections** on the top-level navigation banner or the UI menu ( ) or select the **Connections** box on the Home page. The Connections page appears.

   The Connections page lists the data store connections you have defined and identifies how many data source configurations each connection uses.

Search and Filter Lists

If there are many items included in your work area, you may need to search and filter for what you need.

You can use the search bar to perform a search, and a filter to filter all items or your search results, using built in filters or content tags.

Use The Search Field

You can search many of the columns in your list to find what you need. Select **All** to search all searchable columns, or select an option available from the drop-down menu, such as **Author**, **Name**, or **Type**. Some work areas also allow you to search the text of a **Description** associated with an item.

Note: The columns you can search vary by work area.

Enter some characters into the search field. Composer hides items that don't match your partial search criteria. Narrow the results further by entering a complete search term or character string, or by performing a more complex search.

To perform a complex search, you can include the following special characters in the Search field:

Note: Searches are performed from left to right, with no logical preference or grouping.

Character	Use to ...
& or **:**	Conditionally AND items in the search string.
|	Conditionally OR items in the search string.

Optionally, use these column keywords in complex searches and map to columns in the list.

Note: A column keyword is not searchable in a list that does not contain that column. For example, dashboards do not include a **Type** column.

Keyword	Use to identify a search for ...
author	The name of an author in the Author column.
connection	The name of a connection in the Connection column (outside of the Connections work area).
name	The name of an item in the Name column.
type	The type of an item in the Type column.
description	Items with specific words in the Description.

For example, in the Connections work area, the following search string would search for connections with names that include the characters `impala` and that were created by user `linda`:

```
name:impala & author:linda
```

If no connections in the list meet both of these criteria, a message is returned indicating that no items match the current search.

Filter Lists or Results

You can search for specific items using the search field, and narrow your returned results by selecting a content filter at the top of your work area. These filters are available in the Sources work area, the Visual Gallery, and the Library.

Filter	Description
All	Removes any filters for the list and displays all items of this type you can access within your Composer environment.
	Displays only items that you have marked as favorites.
	Displays only items that you created and saved. Visuals created and saved by other users are hidden.
	Displays only items that other users shared with you. See Grant Permissions for a Visual.

Filter Lists and Search Results Using Tags

After you've added tags to one or more sources, visuals, or dashboards , you can filter your view of these items by selecting one or more tags (or select items that do not have a tag, specifically).

Filter your items using tags

1. Navigate to the work area you want to filter or search and filter (Sources, Visual Gallery, Library).
2. Optionally, enter a search term in the **Search** field to find items in your list that match that search query.
3. Select the filter icon for the **Tags** column header to open a drop-down selection list.
4. Select one or more tags in the list by selecting (checking) the checkbox. The list changes to reflect items that included your selected tags criteria.

5. Select any portion of the work area to hide the drop-down selection list and view your results.

6. Clear the **Search** field and the filtered tags to return your entire list of items to the work area, or simply refresh the page.

Tags Drop-Down Selection List Options

The tags shown in your list are determined by the available tags in your environment. Tags are shared among sources, Visual Gallery visuals, and dashboards . For example, if the sources in your instance only have two associated tags, while visuals and dashboards have a broader range of associated tags, you'll see all available tags in the drop-down selection list.

Two options you can use to filter are not tags. They are listed at the top of the drop-down selection list: **Select All (*n*)** and **[NONE]**.

- **Select All (*n*)**: Use Select All to quickly select or deselect all tags in the list by filling or clearing the checkbox.

- **[NONE]**: Use NONE to return all items in the list that do not have associated tags.
  - If you select NONE while any or all tags are selected, those selections are cleared, and only items with no tags are shown.
  - If you select any other tag while NONE is selected, the NONE checkbox is cleared, and only items with selected tags are shown.

Add Data Store Connections

You can add and validate data store connections on the Connections page in the UI.

Note: You must be logged in as a user with the group privilege **Manage Connections** to maintain data store connection definitions.

Add and validate a connection

1. Log in as a user with the group privilege **Manage Connections**.
2. Select **Connections** on the top-level navigation banner, the UI menu (), or select the **Connections** option on the Home page. The Connections page appears.

   The Connections page lists the data store connections you have defined, identifies the number of associated data sources, and other overview information about your existing connections.
3. Select **Create Connection** in the upper right corner of the connection list. The Select a Connection Type dialog appears.
4. Select the connection type you want to use for the connection definition. The Add <type> Connection page appears.
5. Optionally, change the default name for the connection in the **Connection Name** field.
6. The connection details required to connect to a data store vary by data store. Use the fields in Connection Details to specify the URL and other connection details and, if applicable, authentication credentials (**User Name** and **Password** fields) required to connect to the data store. Fields highlighted in red are required. Any connection requirements for a specific data store are described in the connector documentation for that data store. See the Data Connector Reference.

   You can insert variables for connection parameters, if you have defined any custom attributes in your environment. See Insert Variables for Connection Parameters. Additionally, you can select any user attributes you have defined for connection parameters using the up and down arrows in the connection parameter section. See Use User Attributes for Connection Parameters.
7. If the **Do As User** option is available, optionally specify the custom user attributes you set up to enable user delegation. See Apply User Delegation to a Connection.
8. This is an optional step.

   Each data source configuration specifies refresh settings for the data from the data store. If a data store connection requires special credentials to refresh the data source data, select the **Add an Override** button under **Scheduler Overrides** and select an override setting to use. The override settings you can specify mirror the regular data store connection settings (except for the connection definition name) and vary based on the type of data store connector used for the connection. See Data Connector Reference.

More than one override setting can be specified. Simply select the **Add an Override** button again and select a different setting and provide its input value. Repeat this process until all override settings required by the data store have been specified.

9. Select **Validate** to validate the connection. If the connection is valid, you can save the connection. If invalid, make changes, then select **Validate** again.

10. Select **Save** to save the connection.

After you create a connection, you can update the Connection Details or view Data Sources associated with this connection at any time. See Modify Data Store Connections.

Preview and Edit Schemas

Use ComposerSymphony to preivew and edit the schemas associated with your connections and the data sources.

When you preview a schema, the relationships detected from the database, stored in the metadata for your instance, are highlighted. This helps you understand to understand the relationships present, whether created by the database, or by users. Preview a schema when you create or edit a data source.

Users with appropriate permissions can also edit schemas at the connection level. Select a field in a table, then draw a link to another field to create a join. Create one-to-one or one-to-many joins as needed. When you **Save** your changes, the new relationships are added to your metadata store.

Modify Data Store Connections

You can modify data store connections.

Note: You must be logged in as an administrator or as a user with the group privilege **Manage Connections** to maintain data store connection definitions.

Modify a data store connection definition:

1. Make sure you are logged in as a user with the group privilege **Manage Connections**.
2. Select **Connections** on the top-level navigation banner or the UI menu () or select the **Connections** option on the Home page. The Connections page appears. Search the list to locate the data store connection definition you want to modify. See Search and Filter Lists.

   The Connections page lists the data store connections you have defined and identifies how many data source configurations each connection uses.
3. Select the data store connection definition you want to modify. The **Connection Details** tab opens.
4. Select the edit icon  next to the name at the top of the page to change the name of this connection. The field is now editable.

   Change the name and select **Save**.
5. Use the fields on the Connection Details tab to alter the URL and other connection details and, if applicable, the authentication credentials (**User Name** and **Password** fields) required to connect to the data store. Any connection requirements for a specific data store are described in the connector documentation for that data store. See the Data Connector Reference.

   You can insert variables for connection parameters, if you have defined any custom attributes in your environment. See Insert Variables for Connection Parameters. Additionally, you can select any user attributes you have defined for connection parameters using the up and down arrows in the connection parameter section. See Use User Attributes for Connection Parameters.
6. If the **Do As User** option is available, optionally specify the custom user attributes you set up to enable user delegation. See Apply User Delegation to a Connection.
7. This is an optional step.

   Each data source configuration specifies refresh settings for the data from the data store. If a data store connection requires special credentials to refresh the data source data, select the **Add an Override** button under **Scheduler Overrides** and select an override setting to use. The override settings you can specify mirror the regular data store connection settings (except for the connection definition name) and vary based on the type of data store connector used for the connection. See Data Connector Reference.

More than one override setting can be specified. Simply select the **Add an Override** button again and select a different setting and provide its input value. Repeat this process until all override settings required by the data store have been specified.

8. Select **Validate** to validate the connection. If the connection is valid, you can save the connection. If invalid, make changes, then select **Validate** again.
9. Select **Save** to save the connection.
10. To see the data source configuration definitions that use this connection definition, select the **Data Sources** tab.
11. Select **Back** at the top of the page to return to the Connections page that lists the connection definitions.

Delete Data Store Connections

A data store connection cannot be deleted if it is used by any data source configurations. You must first remove it from the data source configurations before you can delete it.

Note: You must be logged in as an administrator or as a user with the group privilege **Manage Connections** to maintain data store connection definitions.

Delete a data store connection:

1. Log in as an administrator or as a user with the group privilege **Manage Connections**.
2. Select **Connections** on the top-level navigation banner or the UI menu () or select the **Connections** box on the Home page. The Connections page appears.

   The Connections page lists the data store connections you have defined and identifies how many data source configurations each connection uses.
3. Highlight (hover over) the row listing the data store connection you want to delete. You can search the list to locate the connection definition. See Search And Filter Lists.
4. Select the delete icon () in the Actions column for the associated row.

   A warning dialog appears.
5. Select **Delete** on the warning dialog. The connection is deleted.

Note: If you try to delete a visual, filter snippet, dashboard, dashboard link, source, or source field, Composer displays an error message naming any objects dependent on the item you're trying to delete. You can delete the item after you've removed the association from the dependent object. See Fields Usage.

Select a Connection for Source

Data store connections are selected as part of creating sources. If you can modify and save sources, you can select or change the data store connections they use.

> **Note:** You must be logged in as an administrator or as a user with the group privilege **Manage Connections** to select or change the data store connection for a data source.

If you are logged in as an administrator or as a user with both the group privileges **Manage Connections** and **Create New Data Sources**, you select the data store connection for a source on the Source Creation tab of the source. See Source Creation Tab and Manage Data Store Connections.

Use User Attributes for Connection Parameters

User attributes (variables) can be used for the connection parameters in a connection definition. Attributes can be defined for the full connection string, the host name, the port number, the cluster name, the user name, or the password, and any other parameters supported by the connector. The attributes are passed to the connection string via custom attributes specified in the user definition or dynamically in the custom attributes specified in the SAML or LDAP configurations for your environment. This topic describes how to set up this functionality.

- Step 1: Review And Select Connector Parameters For Which Attributes Can Be Used
- Step 2: Define Custom Attributes
- Step 3: Define Connections Using User Attributes

You can also insert variables directly in the connection parameters of a connection definition. See Insert Variables For Connection Parameters.

Step 1: Review and Select Connector Parameters for Which Attributes Can Be Used

Review and select connector parameters for which user attributes can be used

1. Log in as a system admin or a member of the Supervisors group.

   Note: The default **supervisor** user is no longer installed; add users to the **Supervisors** group instead.

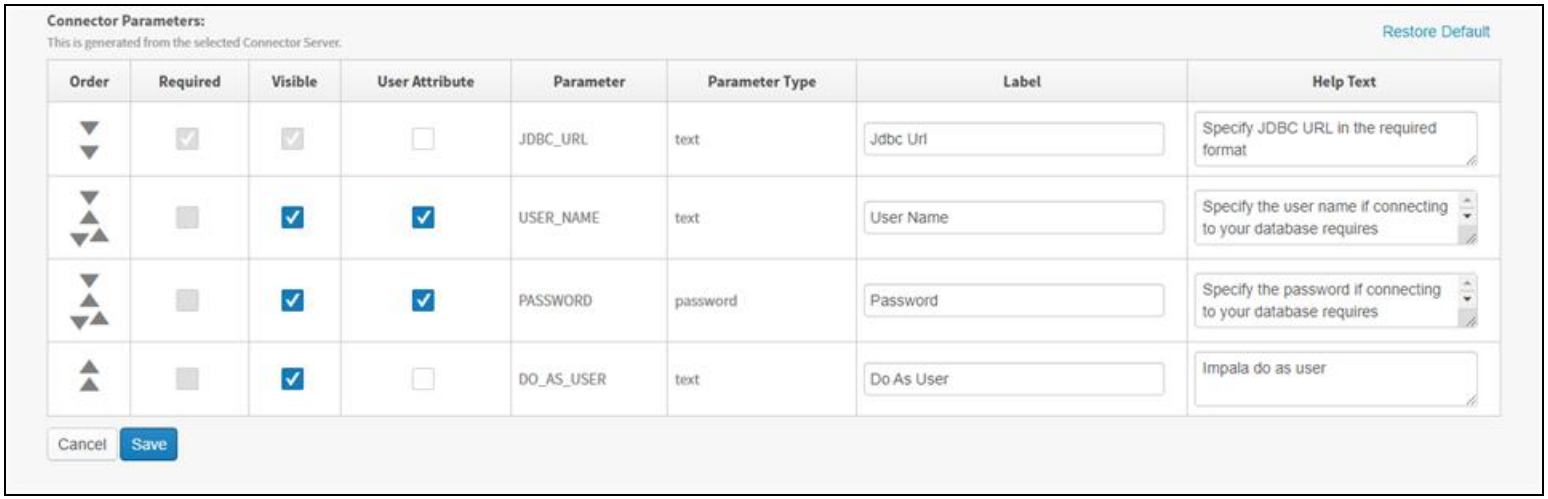
2. Select **Connectors** from the menu ( ). The Manage Connector Services page appears.

3. In the Connectors section of the page, select connector parameters to review. Alternatively, you can create a new connector definition and review and modify the parameters in the new definition. See Define A New Connector.

   Either the Edit Connector or the Create New Connector page for the connector appears.

4. Review the connector parameters that appear at the bottom of the Edit Connector or Create New Connector page.

5. In the following Impala connector definition, the **User Attribute** checkbox is selected for the USER_NAME and PASSWORD parameters. The user definitions in this instance must include custom attributes for these parameters (see Step 2) and the connection definitions for this Impala connector must select the appropriate custom attributes as user credentials (see Step 3).



6. Alter the other connector parameters, as needed.

i. If a connector parameter is required, make sure its **Required** checkbox is selected. Depending on the connector server, some parameters are already selected because they are required.

ii. If a connector parameter should be visible (especially if the parameter is required) when the connection definition is created, make sure its **Visible** checkbox is selected.

   For information about all connector parameters, see Define A New Connector.

7. Select **Save** to save the connector parameters.

Step 2: Define Custom Attributes

A custom attribute must be defined for every connector parameter for which you selected the **User Attribute** checkbox in Step 1. The only exceptions are the context variables `${User.composerUserName}`, `${User.accountId}`, and `${User.credentials}`. These built-in attributes which automatically exist and can be used connect the currently logged in user.

- `${User.composerUserName}`, `${User.accountId}`: include to insert the name or account ID of the user that is currently logged in.
- `${User.credentials}`: include to pass the session ID or trusted access token (in embedded environments) of the user.

You can define custom attributes in several ways:

- Individually for every user who needs to create data sources from a connection or using the connector. If you use this method, the variable names must be the same for each user. See Specify Custom User Attributes.

- Dynamically in the LDAP or SAML configurations for your Composer instance. See Use Lightweight Directory Access Protocol (LDAP) With ComposerSymphony and Configure ComposerSymphony To Support SAML.

Note: JavaScript and HTML Source files used for embedding must be encoded in UTF-8 without BOM.

The same custom attribute key name must be defined and used in all the user definitions of the users expected to use this connection.

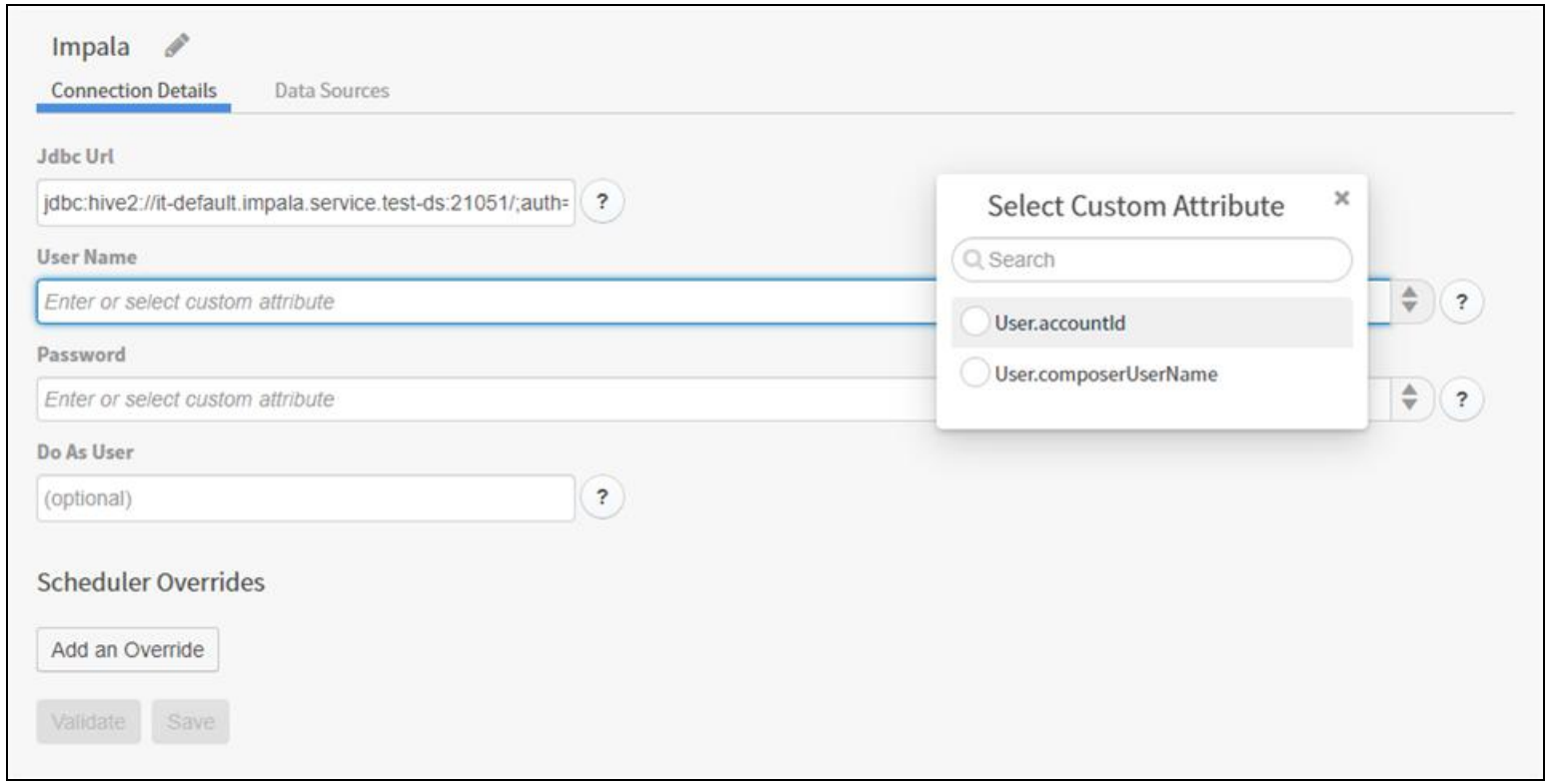
Step 3: Define Connections Using User Attributes

Define connections using user attributes

1. Log in as an system admin.

2. Create or edit a connection using the connector you updated in Step 1. See Add Data Store Connections and Modify Data Store Connections.

3. If a connection parameter is identified as a user attribute, up and down arrows appear in the connection parameter field on the Connections and Connection Details work areas. Use these arrows to select the custom attribute you want to use for the connection from list shown in the **Select Custom Attribute** drop-down menu.



Note: If the connector associated with the connection type for the connection definition has ***not*** been defined with the **User Attribute** checkbox selected (the **User Attribute** checkbox *was* selected in Step 1) for the USER_NAME or PASSWORD parameters, the Select Custom Attribute drop-down menu is not available and you must manually enter the custom attribute in `${User.<custom-attribute-name>}` format. See Insert Variables For Connection Parameters. Note that the custom attributes in this case do not use the same format as when you select them from the drop-down menu.

4. Select **Validate** to validate the connection. If the connection is valid, you can save the connection. If invalid, make changes, then select **Validate** again.
5. Select **Save** to save the connection.

Insert Variables for Connection Parameters

Variables can be inserted for any connection parameter in a connection definition. The variables are passed to the connection string via custom attributes specified in the user definition or dynamically in the custom attributes specified in the SAML or LDAP configurations for your environment.

You can also specify user attributes for use in the connection parameters of a connection definition. See Use User Attributes for Connection Parameters.

Step 1: Define Custom Attributes for the Variables

A custom attribute must be defined for every variable you want to use. The only exceptions are the Composer context variables `${User.composerUserName}`, `${User.accountId}`, and `${User.credentials}`. These built-in attributes which automatically exist and can be used connect the currently logged in user.

You can define custom attributes in several ways:

- Individually for every user. If you use this method, the variable names must be the same for every user. See Specify Custom User Attributes.
- Dynamically in the LDAP or SAML configurations for your Composer instance. See Use Lightweight Directory Access Protocol (LDAP) With ComposerSymphony and Configure ComposerSymphony To Support SAML.

Details about specifying custom attribute values are provided in Specify Custom User Attributes.

Step 2: Define Connections Using Variables

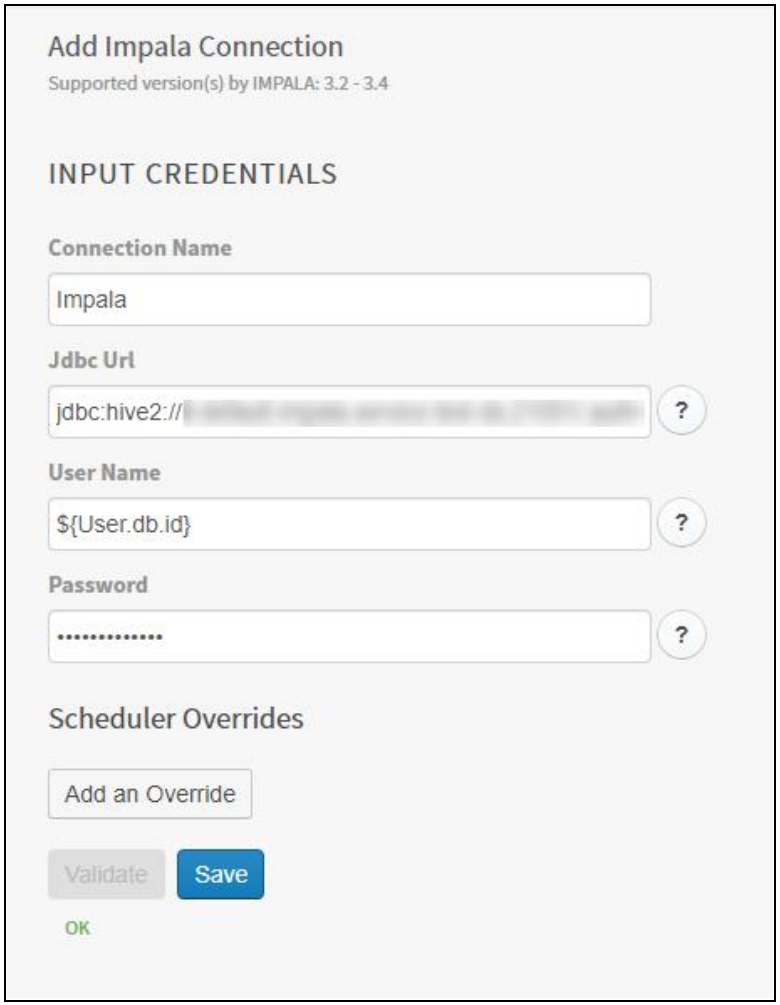
Define connections using variables

1. Log in as an system admin.
2. Create or edit a connection definition. See Add Data Store Connections and Modify Data Store Connections.
3. If custom attributes have been defined, they can be directly entered in a connection parameter field on the Connections page using the following syntax:

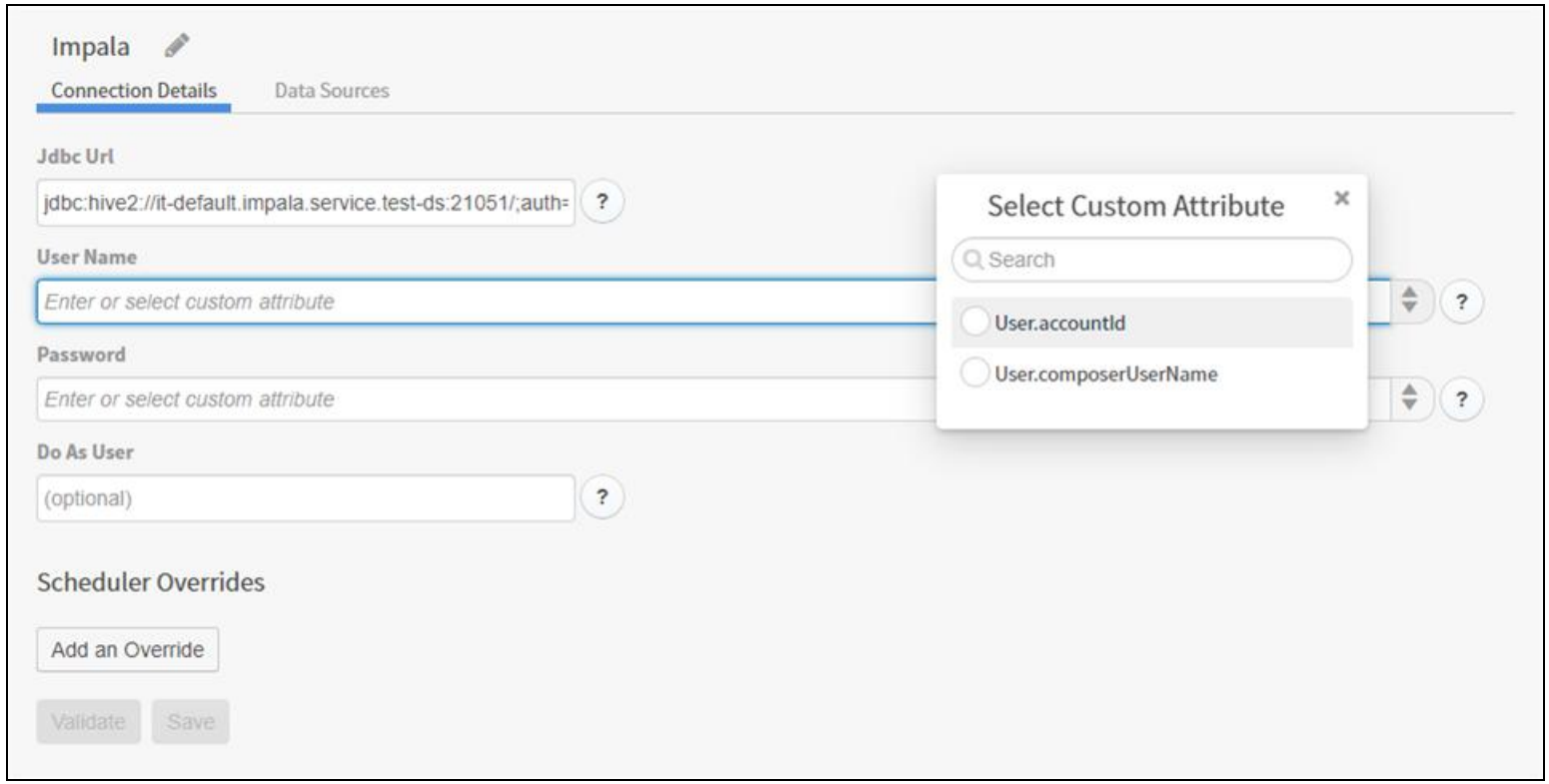
   ```
   ${User.<custom-attribute-name>}
   ```

   The same custom attribute key name must be defined and used in all the user definitions of the users expected to use this connection.

   For example:



Note: If the connector associated with the connection type for the connection definition has been defined with the **User Attribute** checkbox selected for the USER_NAME or the PASSWORD parameters, a custom user attribute ***must*** be defined for the user creating the connection and for any users using the connection. In this scenario, the Add Connection screen allows you to select the custom user attribute from a **Select Custom Attribute** drop-down menu, as shown below. Note that the custom attributes in this case are not shown using the same format as when you specify them manually. For complete information, see Use User Attributes For Connection Parameters.



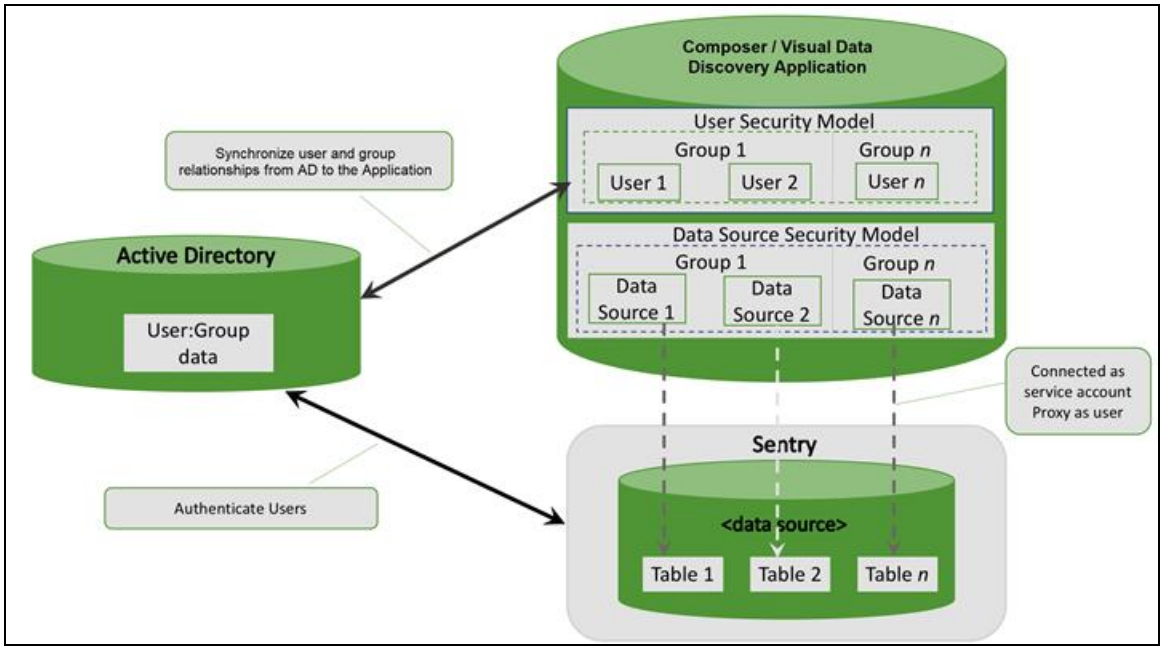
4. Select **Validate** to validate the connection. If the connection is valid, you can save the connection. If invalid, make changes, then select **Validate** again.
5. Select **Save** to save the connection.

Apply User Delegation to a Connection

Applying user delegation to a Composer data source connection definition involves setting the **Do As User** parameter in the connection definition and setting up proxy user features in your data store. Any authentication mechanism (Kerberos or LDAP) and group mapping (file system or LDAP-based) method can be used by the data store or Composer, as long as the user name assigned to the **Do As User** connector parameter is allowed appropriate authorizations (delegation) in the data store configuration.

Note: Supervisors group members enables user delegation via a custom user attribute. Administrators enable and apply user delegation to the data connection definition for a data source.

User delegation processing is depicted in the following diagram.

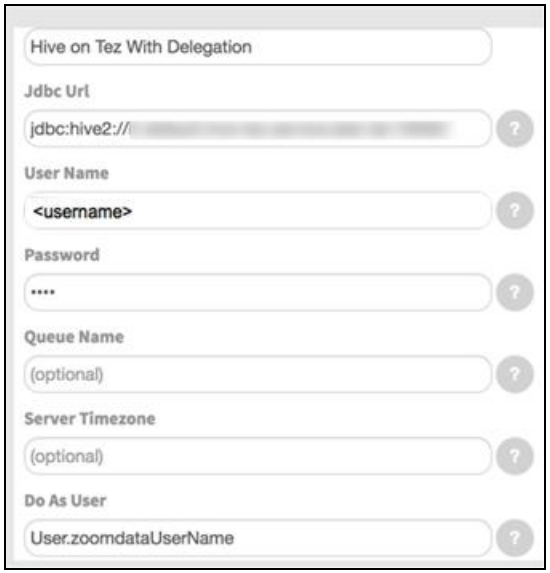


User delegation occurs in this manner:

1. The Composer supervisor group member or administrator assigns any LDAP attribute (for example, `cn`, `sAMAccountName`, `name`) to a Composer custom user attribute. This should be provided by your data store administrator. The only requirement is that this attribute must match the configuration in Sentry. See Enable User Delegation.

   The Composer custom user attribute is referenced by its name, prefaced by the word `User`. For example, if your Composer custom user attribute is named `XXXUserName`, you would reference it as `User.XXXUserName`.

2. The Composer administrator references the custom user attribute in the appropriate data source connection definition using the connection's **Do As User** box. For example:



3. When a user submits a query using the data source, the Composer connector sends the user identified by the **Do As User** parameter (or as interpreted by the setting in that parameter) to the data store when it connects on behalf of the query.

   Assuming user proxy (user delegation) features are set up properly on the data store, the data store runs the query on behalf of the user. For information on setting up user proxy, user impersonation, or user delegation features in each data store, see the following links.

Data Store	User Proxy Setup Links
Apache Drill	■ https://drill.apache.org/docs/configuring-user-impersonation/ ■ https://drill.apache.org/docs/configuring-inbound-impersonation/
Cloudera Impala	■ https://docs.cloudera.com/documentation/enterprise/latest/topics/impala_delegation.html
Cloudera Search	■ https://docs.cloudera.com/documentation/enterprise/latest/topics/impala_delegation.html ■ https://docs.cloudera.com/documentation/enterprise/5-12-x/topics/search_ha_proxy.html

Data Store	User Proxy Setup Links
	■ https://docs.cloudera.com/documentation/enterprise/5-10-x/topics/admin_hdfs_proxy_users.html
Hive	■ https://community.cloudera.com/t5/Community-Articles/Enable-DoAs-option-Hive-to-allow-users-to-runs-queries-with/ta-p/247400 ■ https://docs.cloudera.com/documentation/enterprise/5-8-x/topics/cdh_sg_hiveserver2_security.html#concept_vjq_c3x_nm ■ https://hadoop.apache.org/docs/stable/hadoop-project-dist/hadoop-common/Superusers.html

Enable User Delegation

You can use user delegation to run queries on behalf of users using a single set of credentials for a number of connectors. This allows you to share a single connection configuration among all users. User delegation can be established on a per-user or a per-group basis.

User delegation is currently supported by the following connectors: Apache Drill, Cloudera Impala, Cloudera Search, and Hive. The Oracle connector supports user delegation only via user credential pass-through.

Note: The Composer supervisor group members enables user delegation via a custom user attribute. Composer administrators enable and apply user delegation to the data connection definition for a data source.

Enable user delegation

1. Log in as a system admin or a member of the Supervisors group.

   Note: The default **supervisor** user is no longer installed; add users to the **Supervisors** group instead.

2. Select **Security** from the menu. The security tabs display.

3. Select the **LDAP Settings** tab. The LDAP Settings tab has five sections: **General Settings**, **LDAP Server**, **User Provisioning**, **Mappings**, and **Mappings to Custom User Attributes**.

   Note: If the LDAP tab cannot be selected, verify that the LDAP security service is enabled. See Use Lightweight Directory Access Protocol (LDAP) With ComposerSymphony.

4. In the **Mapping to Custom User Attributes** section, select **Add Custom User Attribute**.

5. Type any meaningful name for the custom attribute name.

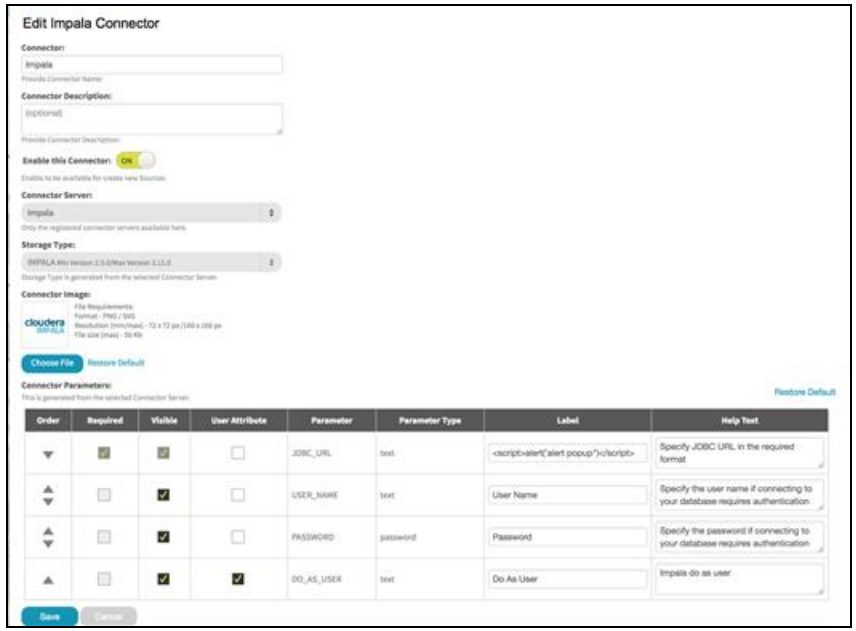
6. Match the new attribute to any LDAP attribute (for example, cn, sAMAccountName, name). This should be provided by an Impala administrator. The only requirement is that this attribute match the configuration in Sentry.

7. Select **Save** to save the attribute.

   The custom user attribute is referenced by its name, prefaced by the word `User`. For example, if your custom user attribute is named `XXXUserName`, you would reference it as `User.XXXUserName`. This reference name is shown in the **Usage** column.

8. Select **Connectors** on the supervisor menu. The Manage Connector Services page appears. This page has two tables: one for Connector Servers and one for Connectors.

9. Scroll down to the Connectors table and select the appropriate connector from the list. The connector settings page displays.



10. Scroll down to the Connector Parameters and verify that the checkbox in the **User Attribute** column for the **DO_AS_USER** parameter is selected. This ensures that the **DO_AS_USER** parameter is visible and can be set in your Impala connection.

11. Select **Save**.

To apply user delegation to a data source connection definition, see Apply User Delegation To A Connection.

Support of Nested Data Structures in Composer

Composer supports aggregations for nested (or hierarchical) data structures for some data stores.

Support for this feature by connector is shown in the following table.

Key:Y - Supported; **N** - Not Supported; N/A - not applicable

Connector	Supported?
Amazon Redshift	N/A
Amazon S3	N/A
Apache Drill	N/A
Apache Phoenix	N/A
Apache Phoenix Query Server (QS)	N/A
Apache Solr	**N**
BigQuery	N/A
Business Central Jet	N/A
Cloudera Impala	N/A
Cloudera Search	N/A
Couchbase	N/A
Dremio	N/A
Elasticsearch 7.0	**Y**
Elasticsearch 8.0	**Y**
File Upload	N/A
HDFS	N/A
Hive	N/A
Jira	**N**
MemSQL	N/A
Microsoft SQL Server	N/A
MongoDB	**Y**
MySQL	N/A
Oracle	N/A
PostgreSQL	N/A

Connector	Supported?
Python	**N**
Real Time Sales	N/A
Salesforce	**N**
SAP Hana	N/A
SAP S/4HANA	N/A
SAP IQ	N/A
Spark SQL	N/A
Snowflake	N/A
Teradata	N/A
TIBCO DV	N/A
Trino	N/A
File Upload (Upload API)	N/A
Vertica	N/A

There are two ways to store nested structure:

1. Store all hierarchy as a single document, for example, in JSON format (nested documents).

2. Store hierarchy items as separate documents and additional info on hierarchical links internally (block join).

Nested Documents

Hierarchical structure can be represented in JSON format. In MongoDB and Elasticsearch, storing such structures is supported.

Consider the following example. We need to store a hierarchy of divisions by country with two divisions in country. Also we need to store some general country information, for example, foundation year.

In this case, the following JSON is sent to the index document:

```
{
        "country":"Germany",
        "foundation year":2008,
        "divisions":[
                {
                        "city":"Berlin",
```

```
                        "sales":200,
                        "manager":{
                                "first name":"Robert",
                                "last name":"Simmons",
                                "years in company":4
                                }
                },
                {
                        "city":"Munich",
                        "sales":200,
                        "manager":{
                                "first name":"Robert",
                                "last name":"Simmons",
                                "years in company":4
                                }
                }
        ]
}
```

In MongoDB, you can store such documents and then query them as is, without any restrictions. However, the performance may be slow if the document contains a lot of arrays.

In Elasticsearch, we recommend using the "nested" type for complex objects before the document is indexed.

Block Join Support

There is another way to store hierarchical structures. All hierarchy items are stored as separate elements, with information about the hierarchical links stored internally. Apache Solr supports this approach.

Consider the following example. We need to store a hierarchy of divisions by country with two divisions in country. Also we need to store some general country information, for example, foundation year.

In this case, the following JSON is sent to the index document:

```
{
        "country":"Germany",
        "foundationYear":2008,
        "_childDocuments_":[
                {
                        "city":"Berlin",
                        "sales":200,
                        "managerFirstName":"Robert",
```

```
                        "managerLastName":"Simmons",
                        "managerYearsInCompany":4
                },
                {
                        "city":"Munich",
                        "sales":200,
                        "managerFirstName":"Robert",
                        "managerLastName":"Simmons",
                        "managerYearsInCompany":4
                }
        ]
}
```

As a result, there are three documents in the index. Information on hierarchical linking of these objects is stored internally in Solr.

```
{
        "country":"Germany",
        "foundationYear":2008
},
{
        "city":"Berlin",
        "sales":200,
        "managerFirstName":"Robert",
        "managerLastName":"Simmons",
        "managerYearsInCompany":4
},
{
        "city":"Munich",
        "sales":200,
        "managerFirstName":"Robert",
        "managerLastName":"Simmons",
        "managerYearsInCompany":4
}
```

You must specify what fields are used in parent documents. To do this, you must select the checkbox in the **Parent Field** column on the **Fields** tab while creating or modifying the data source configuration .

Use OAuth 2.0 in Connections to Cloud Data Stores

You can leverage existing authorization rules of BigQuery and Snowflake data sources by enabling OAuth 2.0 for these connectors in Composer. Users access the connected data stores, using their personalized credentials, and receive access to the data following the security rules of your data source.

Feature Support

Use OAuth to connect to these supported data sources:

- Connect to BigQuery
- Connect to Snowflake

To avoid frequent authentication requests for users, Composer operates with long-lived tokens and preemptively refreshes the tokens when they are close to expiration.

Note: Scheduled source refresh is not available when you use OAuth 2.0 authentication.

Data Connector Reference

Composer data connectors are used to connect to your data stores. Each data connector has capabilities and limitations when connected to the Composer server. This topic highlights those details and provides a link to more information about each connector. For information about setting the parameters required by a connector to connect to a data store, see Manage Data Store Connections. For information on how to download and install a connector that is not provided in the default Composer installation, see Obtain Additional Connector Servers.

Connector	Microservice	Supported Versions	Connector Port	Other Actions Required & Notes
Amazon Redshift	`zoomdata-edc-redshift`	1.0	8202	Install JDBC driver.
Amazon S3	`zoomdata-edc-s3`		8129	Separate download.
Apache Drill	`zoomdata-edc-drill`	1.14 - 1.16	8095	Separate download.
Apache Phoenix	`zoomdata-edc-phoenix-4.7`	4.7	8124	Apache Phoenix 4.4 and 4.5 require separate downloads.
Apache Phoenix Query Server (QS)	`zoomdata-edc-phoenix-4.7-queryserver`	4.7	8125	
Apache Solr	`zoomdata-edc-apache-solr`	7.4 - 8.4	8115	
BigQuery	`zoomdata-edc-bigquery`		8093	
Business Central	`zoomdata-edc-businesscentral-jet`	N/A	8156	Separate download.
Cloudera Impala	`zoomdata-edc-impala`	3.2 - 3.4	8098	
Cloudera Search	`zoomdata-edc-cloudera-search`	4.10 - 7.4	8201	
Couchbase	`zoomdata-edc-couchbase`	6.0.1	8138	Includes Couchbase Community Edition 6.0.0.
Dremio	`zoomdata-edc-dremio`	4.1 through 4.8	8142	
Elasticsearch 7.0	`zoomdata-edc-elasticsearch-7.0`	7.0 - 7.17	8139	
Elasticsearch 8.0	`zoomdata-edc-elasticsearch-8-0`	8.1 - 8.3	8147	
HDFS	`zoomdata-edc-hdfs`		8126	Separate download.
Hive	`zoomdata-edc-hive`	2.1 - 3.1	8132	

Connector	Microservice	Supported Versions	Connector Port	Other Actions Required & Notes
Jira Connector	zoomdata-edc-jira	Jira JDBC Driver 1.7.8.1002	8151	Separate download.
MemSQL	zoomdata-edc-memsql	7.1 - 7.6	8099	Install JDBC driver.
Microsoft SQL Server	zoomdata-edc-mssql	12.0 (SQL Server 2014) - 16.0 (SQL Server 2022)	8100	
MongoDB	zoomdata-edc-mongo	3.4 - 4.4	8123	
MySQL	zoomdata-edc-mysql	5.6 - 8.0	8101	Install JDBC driver.
Oracle	zoomdata-edc-oracle	11.2 - 21c	8102	Install JDBC driver.
PostgreSQL	zoomdata-edc-postgresql	9.6 - 14.0	8105	
Python	zoomdata-edc-python		8153	Install Docker image. See Manage The Python Connector.
Real Time Sales	zoomdata-edc-rts		8108	Must be enabled.
Salesforce	zoomdata-edc-salesforce	Simba Salesforce JDBC Driver 2.1.22	8152	Separate download.
SAP Hana	zoomdata-edc-saphana	2.0	8109	Separate download and install JDBC driver.
SAP S/4HANA	zoomdata-edc-saphanacloud	N/A	8205	Separate download and install JDBC driver.
SAP IQ	zoomdata-edc-sapiq	16	8121	Separate download and install JDBC driver.
Snowflake	zoomdata-edc-snowflake	whatever is currently supported in the cloud	8131	Separate download.
Spark SQL	zoomdata-edc-sparksql	2.3 - 3.0	8116	
Teradata	zoomdata-edc-teradata	16.20	8111	Separate download and install JDBC driver.
TIBCO DV	zoomdata-edc-tibcodv	8.0-8.1	8140	Separate download and install JDBC driver. The default TDV server port for JDBC connections is

Connector	Microservice	Supported Versions	Connector Port	Other Actions Required & Notes
				9401.
Trino	`zoomdata-edc-trino`	351-390	8148	Separate download.
Vertica	`zoomdata-edc-vertica`	7.2	8112	Separate download and install JDBC driver.

In addition to the official Composer connectors listed above, Composer allows you to:

- Upload a flat file for viewing in visuals and dashboards.
- Dynamically upload and stream your data in real time using the Composer Upload API.

Both the flat file uploads and the Upload API use port 8105 and require that PostgreSQL be enabled. While these processes are not strictly data connectors, they are additional methods of providing data for Composer.

Manage the Amazon Redshift Connector

The Composer Amazon Redshift connector lets you access the data available in Amazon Redshift storage using the Composer client. The Amazon Redshift connector supports Amazon Redshift version 1.0.

Before you can establish a connection from Composer to Amazon Redshift storage, a connector server needs to be installed and configured. See Manage Connectors And Connector Servers for general instructions and Connect To Amazon Redshift and Troubleshoot The Amazon Redshift Connector for details specific to the Amazon Redshift connector.

After setting up the connector, create data sources that specify the necessary connection information and identify the data you want to use. See Manage Data Sources for more information. After you set up your data sources, create dashboards and visuals from from the data in these data sources. See Create Data Discovery Dashboards.

Feature Support

Connector support for specific features is shown in the following table.

Key: Y - Supported; **N** - Not Supported; N/A - not applicable

Feature	Supported?
Admin-Defined Functions	**Y**
Box Plots	**Y**
Custom SQL Queries	**Y**
Derived Fields (Row-Level Expressions)	**Y**
Distinct Counts	**Y**
Fast Distinct Values	N/A
Group By Multiple Fields	**Y**
Group By Time	**Y**
Group By UNIX Time	**Y**
Histogram Floating Point Values	**Y**
Histograms	**Y**
Kerberos Authentication	**N**
Last Value	**Y**
Live Mode and Playback	**Y**
Multivalued Fields	N/A
Nested Fields	N/A

Feature	Supported?
Partitions	N/A
Pushdown Joins for Fusion Data Sources	**Y**
Schemas	**Y**
Text Search	N/A
TLS	**Y**
User Delegation	**N**
Wildcard Filters	**Y**
Wildcard Filters, Case-Insensitive Mode	**Y**
Wildcard Filters, Case-Sensitive Mode	**Y**

Note: Amazon Redshift returns whole numbers for aggregates on columns of type DECIMAL and NUMERIC types which have a 0 scale (in other words, 0 decimal places).

Connect to Amazon Redshift

Verify the MTU Size of Composer

Before you can establish a connection between Amazon Redshift and Composer, you must verify that the size of the maximum transmission unit (MTU) on your Composer server is set to 1500.

The MTU size determines the maximum size, in bytes, of a packet that can be transferred in one Ethernet frame over your network connection. If your MTU size is too large for the connection, you might experience incomplete query results, your query might hang, or the connection might be dropped altogether. For more information, see: https://docs.aws.amazon.com/redshift/latest/mgmt/connecting-drop-issues.html.

To review the MTU value, use the `ip` command:

```
$ ip addr show eth0
```

If you need to edit the MTU value and set the size to 1500, use the following `ip` command:

```
$ ip link set dev eth0 mtu 1500
```

Configure and Reference the JDBC Driver

The Amazon Redshift connector requires a JDBC driver to be configured before you connect. You can download the driver from the vendor's site. If you are upgrading, keep in mind you need to configure the JDBC driver- see Upgrade ComposerSymphony. For more information, see Add A JDBC Driver. The JDBC Driver for Redshift has more than one jar file that needs to be downloaded: be sure to place the files in the same location to avoid any issues.

When setting up your Amazon Redshift connection, you need to specify the JDBC URL. You can find the URL on the **Configuration** tab of a cluster under **Cluster Database Properties**. The format varies slightly based on the type of database being connected. For Amazon Redshift, use the following format: `jdbc:redshift://HOSTNAME:PORT/DATABASE_NAME` . If authentication has been set up, provide the user name and password.

Troubleshoot the Amazon Redshift Connector

The Composer Amazon Redshift connector lets you access the data available in Amazon Redshift storage using the Composer client. You can troubleshoot out of memory errors that may occur when executing heavy queries against large databases.

Out Of Memory Errors

To troubleshoot out of memory errors, you'll need to set the connector log to DEBUG mode. Edit the Redshift configuration file, `edc-redshift.properties`. See Connector Properties And Property Files. Use the information available in the log to adjust your environment to prevent further errors.

After you set your log to DEBUG mode, run your queries again. Review the logs, and use one or more approaches to resolve the out of memory issues.

- Rewrite your most memory-consuming queries to return more granular results.
- Increase the RAM allocation in your Composer environment for this connector. See Configure Memory Settings.
- Increase the `wlm_query_slot` count in your AWS environment. See Troubleshooting queries - Amazon Redshift.

Manage the Amazon S3 Connector

The Composer Amazon S3 connector lets you access the data available in Amazon Simple Storage Service (S3) storage using the Composer client. This connector uses the S3A protocol.

The Composer Amazon S3 connector uses its own embedded Apache Spark functionality. It supports Apache Spark 2.2 in its implementation.

Before you can establish a connection from Composer to Amazon S3 storage, a connector server needs to be downloaded, configured and enabled. See Manage Connectors And Connector Servers for general instructions and Connect To Amazon S3 for details specific to the Amazon S3 connector.

After setting up the connector, create data sources that specify the necessary connection information and identify the data you want to use. See Manage Data Sources for more information. After you set up your data sources, create dashboards and visuals from from the data in these data sources. See Create Data Discovery Dashboards.

Feature Support

Connector support for specific features is shown in the following table.

Key: Y - Supported; **N** - Not Supported; N/A - not applicable

Feature	Supported?
Admin-Defined Functions	**Y**
Box Plots	**Y**
Custom SQL Queries	**Y**
Derived Fields (Row-Level Expressions)	**Y**
Distinct Counts	**Y**
Fast Distinct Values	N/A
Group By Multiple Fields	**Y**
Group By Time	**Y**
Group By UNIX Time	**Y**
Histogram Floating Point Values	**Y**
Histograms	**Y**
Kerberos Authentication	**N**
Last Value	**Y**
Live Mode and Playback	**Y**
Multivalued Fields	N/A

Feature	Supported?
Nested Fields	N/A
Partitions	N/A
Pushdown Joins for Fusion Data Sources	**N**
Schemas	**Y**
Text Search	N/A
TLS	**N**
User Delegation	**N**
Wildcard Filters	**Y**
Wildcard Filters, Case-Insensitive Mode	**Y**
Wildcard Filters, Case-Sensitive Mode	**Y**

Connect to Amazon S3

Scheduled reload of newly added data is not supported for the Amazon S3 connector. To add new data:

1. Navigate to the Fields tab of your data source configuration.
2. Select (check) the **Refresh Fields** checkbox on the tab. This forces the connector to reload any new data discovered at the level of the data source.

Manage the Apache Drill Connector

The Composer Apache Drill connector lets you access the data available in the Apache Drill open-source SQL query engine using the Composer client. The Composer Apache Drill connector supports Apache Drill versions 1.11 - 1.13.

Before you can establish a connection from Composer to Apache Drill, it must be downloaded, installed, configured, and enabled. See the Apache Drill website. After Apache Drill is installed, a Composer connector server for it needs to be installed, configured and enabled. See Manage Connectors And Connector Servers for general instructions.

After setting up the connector, create data sources that specify the necessary connection information and identify the data you want to use. See Manage Data Sources for more information. After you set up your data sources, create dashboards and visuals from from the data in these data sources. See Create Data Discovery Dashboards.

Feature Support

Connector support for specific features is shown in the following table.

Key: Y - Supported; **N** - Not Supported; N/A - not applicable

Feature	Supported?
Admin-Defined Functions	Y
Box Plots	Y
Custom SQL Queries	Y
Derived Fields (Row-Level Expressions)	Y
Distinct Counts	Y
Fast Distinct Values	N/A
Group By Multiple Fields	Y
Group By Time	Y
Group By UNIX Time	Y
Histogram Floating Point Values	Y
Histograms	Y
Kerberos Authentication	Y
Last Value	Y
Live Mode and Playback	Y
Multivalued Fields	N/A
Nested Fields	N/A

Feature	Supported?
Partitions	**Y**
Pushdown Joins for Fusion Data Sources	**Y**
Schemas	**Y**
Text Search	N/A
TLS	**Y**
User Delegation	**N**
Wildcard Filters	**Y**
Wildcard Filters, Case-Insensitive Mode	**Y**
Wildcard Filters, Case-Sensitive Mode	**Y**

Manage the Apache Phoenix Connector

The Composer Apache Phoenix connector lets you access the data available in your Apache Phoenix storage using the Composer client. The Composer Apache Phoenix connector supports Apache Phoenix version 4.7 and Apache Phoenix Query Server 4.7. Apache Phoenix v4.5 requires a separate download. To obtain a connector for Apache Phoenix 4.4, contact Technical Support.

Before you can establish a connection from Composer to Apache Phoenix, a Composer connector server for it needs to be installed, configured and enabled. See Manage Connectors And Connector Servers for general instructions and Connect To Apache Phoenix for details specific to the Apache Phoenix and Apache Phoenix Query Server connectors.

After setting up the connector, create data sources that specify the necessary connection information and identify the data you want to use. See Manage Data Sources for more information. After you set up your data sources, create dashboards and visuals from from the data in these data sources. See Create Data Discovery Dashboards.

Feature Support

Connector support for specific features is shown in the following table.

Key: Y - Supported; **N** - Not Supported; N/A - not applicable

Feature	Supported?	Notes
Admin-Defined Functions	**Y**	
Box Plots	**N**	
Custom SQL Queries	**Y**	If you need to access a BigQuery partition, explicitly include an alias for the built in partition column in your select clause, such as `select *, _PARTITIONTIME as pt from projectId.datasetId.tableId`.
Derived Fields (Row-Level Expressions)	**Y**	Apache Phoenix and Apache Phoenix Query Server connectors support row-level expressions (derived fields) with the following limitations: ■ The filter IS NULL does not work properly on grouped fields. ■ The LOCATE text row-level function only supports a constant as a argument. ■ A COALESCE conditional row-level function specified with and empty argument does not work properly. ■ If the CASE conditional row-level function returns a null value as a an argument of

Feature	Supported?	Notes
		another function, a NullPointerException may occur. ■ The LPAD and RPAD text row-level functions are not supported.
Distinct Counts	**Y**	
Fast Distinct Values	N/A	
Group By Multiple Fields	**Y**	
Group By Time	**Y**	
Group By UNIX Time	**Y**	
Histogram Floating Point Values	**Y**	
Histograms	**Y**	
Kerberos Authentication	**Y** / **N**	Apache Phoenix supports Kerberos, but Apache Phoenix Query Server does not. For more information, see Enable Kerberos Authentication For Apache Phoenix Connectors.
Last Value	**N**	
Live Mode and Playback	**Y**	
Multivalued Fields	N/A	
Nested Fields	N/A	
Partitions	N/A	
Pushdown Joins for Fusion Data Sources	**N**	
Schemas	**Y**	
Text Search	N/A	
TLS	**Y**	
User Delegation	**N**	
Wildcard Filters	**Y**	
Wildcard Filters, Case-Insensitive Mode	**Y**	
Wildcard Filters, Case-Sensitive Mode	**Y**	

Connect to Apache Phoenix

For Apache Phoenix, specify the JDBC URL in the following format:

```
jdbc:phoenix:<zk_quorum>:<zk_port>:<zk_hbase_path>
```

where:

- <zk_quorum> is a comma separated list of the ZooKeeper servers
- <zk_port> is the ZooKeeper port
- <zk_hbase_path> is the path used by HBase to stop information about the instance

For Apache Phoenix Query Server, specify the JDBC URL in the following format:

```
jdbc:phoenix:thin:url=<scheme>://<server-hostname>:<port>
```

where:

- <scheme> is a transport protocol for communication with the server
- <server-hostname> is the name of the host offering the microservice
- <port> is the port number on which the host is listening

Enable Kerberos Authentication for Apache Phoenix Connectors

Support for Kerberos authentication for Composer Apache Phoenix connectors is only provided for Phoenix 4.7 (and later) connectors. It is not provided for any version of the Phoenix QueryServer connector.

Enable Kerberos authentication for Apache Phoenix connectors:

1. Download `hbase-site.xml` and `core-site.xml` files from the Apache HDFS and HBase microservices. For example, for Hortonworks you can use the instructions at the following link: https://docs.cloudera.com/HDPDocuments/Ambari-2.6.2.2/bk_ambari-operations/content/downloading_client_configs.html.

2. Add the following configuration options to the `hbase-site.xml` file:

```
<property>
    <name>hbase.myclient.principal</name>
    <value>YOUR_PRINCIPAL</value>
</property>
<property>
    <name>hbase.myclient.keytab</name>
    <value>PATH_TO_YOUR_KEYTAB</value>
</property>
```

   Substitute the ID of your Kerberos principal for `YOUR_PRINCIPAL` and the path to your Kerberos keytab file for `PATH_TO_YOUR_KEYTAB`.

3. Verify that the `core-site.xml` file contains the following entry:

```
<property>
    <name>hadoop.security.authentication</name>
    <value>kerberos</value>
</property>
```

4. Make sure that the Apache Phoenix connector has access to the `hbase-site.xml` and `core-site.xml` files as well as the Kerberos keytab file you identified in `PATH_TO_YOUR_KEYTAB`. We recommend that you place these files in the `/etc/zoomdata/edc-phoenix` directory.

5. Add the following property to the `/etc/zoomdata/edc-phoenix-4.7.properties` file to direct the Apache Phoenix connector to the files you created

```
datasource.config.files-path=/etc/zoomdata/edc-phoenix
```

Note: Composer does not recommend that you provide the Kerberos principal ID and keytab file path using a JDBC URL. The Apache Phoenix driver has a bug that will not refresh a ticket after expiration.

Add Patched JAR Files to an Apache Phoenix Connector's Classpath

If your Apache Phoenix/HBase servers use patched `.jar` files, you might need to add the patched `.jar` files to the Apache Connector's classpath. If you do not, Apache Phoenix may produce an error indicating that you have outdated `.jar` files.

Add the patched `.jar` files to the Apache Phoenix connector's classpath:

1. Add the patched `.jar` files to a directory that is accessible to the connector. For example:

```
/usr/local/share/java/zoomdata/phoenix
```

2. Add or modify the following property in the `/etc/zoomdata/edc-phoenix-4.7.properties` file to specify the path to your `.jar` files as a comma-separated list in `datasource.driver-config.jar-path` property. Be sure to put the path to your `.jar` files first so you do not corrupt entries already present in this property. For example:

```
datasource.driver-config.jar-path=/usr/local/share/java/zoomdata/phoenix,lib/edc-phoenix-4.7/phoenix-core-4.7.0-HBase-1.1.jar
```

Manage the Apache Solr Connector

The Composer Apache Solr connector lets you access the data available in your Apache Solr databases using the Composer client. The Composer Apache Solr connector supports Apache Solr versions 7.4 through 8.4.

Before you can establish a connection from Composer to an Apache Solr database, a connector server needs to be installed, configured and enabled. See Manage Connectors And Connector Servers for general instructions and Connect To Apache Solr for details specific to the Apache Solr connector.

After setting up the connector, create data sources that specify the necessary connection information and identify the data you want to use. See Manage Data Sources for more information. After you set up your data sources, create dashboards and visuals from from the data in these data sources. See Create Data Discovery Dashboards.

Feature Support

Connector support for specific features is shown in the following table.

Key: Y - Supported; **N** - Not Supported; N/A - not applicable

Feature	Supported?	Notes
Admin-Defined Functions	**N**	
Box Plots	**Y**	
Custom SQL Queries	**N**	If you need to access a BigQuery partition, explicitly include an alias for the built in partition column in your select clause, such as `select *, _PARTITIONTIME as pt from projectId.datasetId.tableId`.
Derived Fields (Row-Level Expressions)	**N**	
Distinct Counts	**Y**	
Fast Distinct Values	**Y**	
Group By Multiple Fields	**Y**	
Group By Time	**Y**	
Group By UNIX Time	**Y**	
Histogram Floating Point Values	**N**	
Histograms	**Y**	
Kerberos Authentication	**Y**	
Last Value	**N**	

Feature	Supported?	Notes
Live Mode and Playback	**Y**	
Multivalued Fields	**N**	The Apache Solr JSON API does not support metrics by multivalued fields.
Nested Fields	**N**	
Partitions	N/A	
Pushdown Joins for Fusion Data Sources	**N**	
Schemas	N/A	
Text Search	**Y**	You can sort keyword searches by Best Match and Most Recent (when you select a preferred time field from the source). Filter your search results by selecting fields in the Filter modal. Select **Clear All** to clear filtered search results.
TLS	**N**	
User Delegation	**Y**	
Wildcard Filters	**Y**	
Wildcard Filters, Case-Insensitive Mode	**N**	Case-sensitivity cannot be enforced. Consequently, neither case-sensitive or case-insensitive wildcard filters are supported.
Wildcard Filters, Case-Sensitive Mode	**N**	Case-sensitivity cannot be enforced. Consequently, neither case-sensitive or case-insensitive wildcard filters are supported.

In addition, Apache Solr supports the Request Handler field on the Tables/Indices tab of the data source configuration. You can use this box at the top of the Field table on the Tables/Indices tab to specify a request handler plug-in that defines the logic used when executing a search request.

For Kerberos authentication instructions, see Connect To Apache Solr Data Stores That Use Kerberos Authentication. For instructions on using user delegation with the Apache Solr connector, see Configure User Delegation For The Apache Solr Connector.

Connect to Apache Solr

You can configure the Composer Apache Solr connector to connect to a kerberized Apache Solr data store. For more information, see Connect To Apache Solr Data Stores That Use Kerberos Authentication.

When establishing a connection to Apache Solr, you must provide the following information.

1. Select the hosting type: **Standalone** or **Cloud**.
2. Specify a Solr Base URL.
3. Specify the version of the Solr source that you are going to connect to in the following format: `<major>.<minor>.<patch>`. This field is optional.

While connecting to Solr, Composer first checks its version. If the version is not available, Composer checks the version that you have specified in this field and attempts to connect that version.

4. If authentication has been set up, provide the user name and password.

Dashboard and Visual Considerations

Distinct count and percentiles metrics return approximate values in Solr, due to the nature of the data source. The precision of the result returned by distinct count metric depends on the precision threshold setting (default value is 1000).

When you create a bar chart using an Apache Solr data source, you can search for a specific word or phrase using the search box at the top of the dashboard. The **Details** tab displays the results.

Note: Fields of time groupings with low time granularity may cause loading time issues for the data source.

Connect to Apache Solr Data Stores That Use Kerberos Authentication

A secure standalone or cloud Apache Solr can use Kerberos authentication to validate and confirm access requests. You can set up Composer to connect to the secure Solr using the following instructions.

Configure Composer Microservices

Obtain Kerberos Credentials

Each microservice must have its own unique identifier called a principal. Perform the following steps:

1. Install the Kerberos client on the CentOS or Ubuntu machine where the Composer server resides.

2. Generate the Kerberos principal and corresponding keytab for Composer microservice. Before you proceed, make sure that:

i. Composer microservice is running on a node with proper Kerberos configuration: `/etc/krb5.conf` or similar location for your Linux distribution.

ii. The Kerberos realm on your environment is the same as the realm specified in the `kdc.conf` file from the Apache Solr server.

3. Check the Kerberos configuration (that is, `krb5.conf`) and validity of the principal and keytab pair using MIT Kerberos client:

```
kinit -V -k -t <composer_principal>.keytab <composer_principal@KERBEROS.REALM>
```

4. Make the keytab accessible for Composer's Apache Solr connector:

```
sudo mkdir /etc/zoomdata
sudo mv <composer_principal>.keytab /etc/zoomdata
sudo chown zoomdata:zoomdata /etc/zoomdata/<composer_principal>.keytab
sudo chmod 600 /etc/zoomdata/<composer_principal>.keytab
```

Configure the Apache Solr Connector

1. Create or update the file named `/etc/zoomdata/edc-apache-solr.properties`. If this file already exists, verify that the information below exists in the file:

```
kerberos.krb5.conf.location=/etc/krb5.conf
kerberos.service.account.authentication=true
kerberos.service.account.principal=<composer_principal@KERBEROS.REALM>
kerberos.service.account.keytab.location=/etc/zoomdata/<composer_principal>.keytab
```

2. Restart the Apache Solr connector:

```
sudo systemctl restart zoomdata-edc-apache-solr
```

After you have obtained Kerberos credentials and configured the connector properties, follow the instructions provided in Connect To Apache Solr to complete the connection.

Configure User Delegation for the Apache Solr Connector

User delegation is supported by Composer Apache Solr connectors.

Prerequisites

A Solr Cloud cluster, version 6.4 or later, with Kerberos authentication must be available.

Configuration Steps

User delegation configuration for Solr is performed in two steps:

1. Enable Kerberos delegation tokens.

   To enable Kerberos delegation tokens, set the Solr configuration parameter `solr.kerberos.delegation.token.enabled` to `true` (see Using Delegation Tokens in the Kerberos Authentication Plugin documentation) in the `solr.in.sh` file on each Solr node.

2. Configure proxy users and delegates.

   Configuration of proxy users and delegates is performed using these parameters in the `solr.in.sh` file on each Solr node:

i.
```
solr.kerberos.impersonator.user.<USER>.users
```

ii.
```
solr.kerberos.impersonator.user.<USER>.groups
```

iii.
```
solr.kerberos.impersonator.user.<USER>.hosts.
```

Consider the following example:

```
solr.kerberos.impersonator.user.proxy_user.groups=finance,marketing
solr.kerberos.impersonator.user.proxy_user.hosts=*
```

In this configuration, user `proxy_user` can impersonate users belonging to groups `finance` or `marketing` when connecting to a Solr instance from any

host.

all these parameters should be stored in file `solr.in.sh` on each Solr node.

Manage the Aurora Connector

The Aurora connector has been deprecated. However, you can connect to your Aurora databases using our standard MySQL connector. See Manage The MySQL Connector.

Manage the BigQuery Connector

The Composer BigQuery connector lets you access the data available in Google BigQuery storage using the Composer client. The Composer BigQuery connector supports the current version of this software as a microservice (SaaS) product.

The Composer BigQuery connector is a cloud connector that connects to Google BigQuery via the BigQuery API. See Manage Connectors And Connector Servers for general instructions and Connect To BigQuery for details specific to the BigQuery connector.

After setting up the connector, create data sources that specify the necessary connection information and identify the data you want to use. See Manage Data Sources for more information. After you set up your data sources, create dashboards and visuals from from the data in these data sources. See Create Data Discovery Dashboards.

Feature Support

Connector support for specific features is shown in the following table.

Key: Y - Supported; **N** - Not Supported; N/A - not applicable

Feature	Supported?
Admin-Defined Functions	Y
Box Plots	Y
Custom SQL Queries	Y
Derived Fields (Row-Level Expressions)	Y
Distinct Counts	Y
Fast Distinct Values	N/A
Group By Multiple Fields	Y
Group By Time	Y
Group By UNIX Time	Y
Histogram Floating Point Values	Y
Histograms	Y
Kerberos Authentication	N/A
Last Value	Y
Live Mode and Playback	Y
Multivalued Fields	N/A
Nested Fields	N/A

Feature	Supported?
Partitions	Y
Pushdown Joins for Fusion Data Sources	Y
Schemas	Y
Text Search	N/A
TLS	N/A
User Delegation	N
Wildcard Filters	Y
Wildcard Filters, Case-Insensitive Mode	Y
Wildcard Filters, Case-Sensitive Mode	Y

Connect to BigQuery

When connecting to BigQuery, provide the following information:

- Key Path: you have to specify the absolute path to the file that must be available for the connector.
- Public Project IDs.

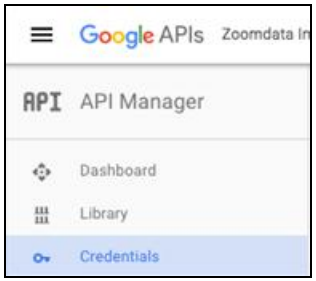
For more information about these values, refer to Google BigQuery's documentation.

Authorize the BigQuery Connection

To authorize the BigQuery connection, you need to create a security key for it. Before you can create the security key, you must access or create a BigQuery microservice account.

Create a BigQuery microservice account

1. Login to your Google API Console.
2. Select the required project from the list.
3. Make sure that current account is linked to a billing account. To check this, select the menu () icon and then select **Billing**.
4. On the API Manager page, select **Credentials**:



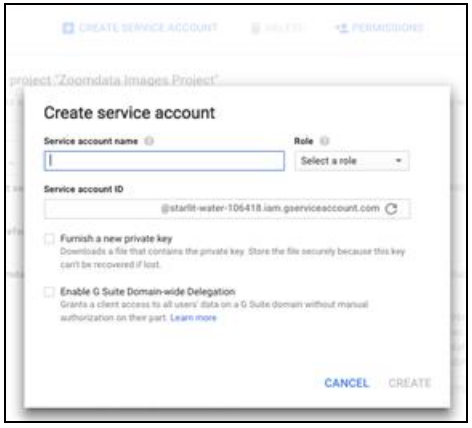
5. On the Credentials page, select **Manage service accounts**.

6. On the Service Accounts page, select **Create Service Account** and specify the following:

i. Service account name

ii. Role - grant this microservice account role based access to the project. From the list, select the **BigQuery** category and then select **BigQuery Data Viewer** and **BigQuery User** roles.

iii. Service account ID



7. Select **Create**.

After you have created an account, create a security key for it.

Create a security key

1. On the **Service Accounts** page, find the required account.
2. From the menu, select **Create key**.
3. In the Create private key dialog, select **JSON** for the key type and select **Create**. The local copy of the key is saved on your computer.

   For more information, see the following Google resource: BigQuery Introduction to Authentication.
4. Move the file with the key to the server, on which the connector is running.

Connect to BigQuery Using OAuth

To create a BigQuery connection use one of the available authentication methods:

- Key authentication flow requires a security key to be generated at BigQuery and placed to the Composer instance;
- OAuth 2.0 requires providing OAuth `client_id` and `client_secrets` generated for a user that will serve for data retrieval, such as an integration user. Users are asked to authenticate via a separate authentication form. Users provide their individual credentials when accessing the data retrieved using this connection.

If both authentication methods are selected, connection via OAuth will have higher priority over key authentication except for the scheduled overrides setup.

	Authentication Flow	
Key Path	Key Authentication	Absolute path to the key authentication file obtained from BigQuery and placed to Composer instance.
Public Project Ids		List of public project IDs that will be queried for the data.

	Authentication Flow	
OAuth 2.0 Enabled	OAuth 2.0	TRUE/FALSE
Project Id		Billing project ID that will be queried for the data. **Note:** Optional if keys authentication is used. **Important:** Mandatory if OAuth 2.0 connection is enabled.
OAuth 2.0 Client Id		`client_id`: Obtain from BigQuery. See https://cloud.google.com/bigquery/docs/authentication/end-user-installed.
OAuth 2.0 Client Secrets		`client_secrets`: Obtain from BigQuery. See https://cloud.google.com/bigquery/docs/authentication/end-user-installed.

Scheduled Override Options

To maintain Composer's ability to perform scheduled operations such as scheduled dashboard reports, alerts notifications, and more when using OAuth 2.0 authentication flow, you can setup scheduled overrides with key authentication method by providing a key path.

Additional OAuth 2.0 parameters available for override, however, already have prepopulated BigQuery values and do not require manual editing:

- `OAUTH2.AUTHORIZATION_URI`
- `OAUTH2.TOKEN_URI`
- `OAUTH2.SCOPES`

Note: Scheduled source refresh is not available when you use OAuth 2.0 authentication.

To avoid frequent authentication requests for users, Composer operates with long-lived tokens and preemptively refreshes the tokens when they are close to expiration.

Important: Users' OAuth sessions are terminated when the OAuth token is revoked, if the connection is deleted, or connection details are modified.

Manage the Business Central Jet Connector

The Composer Business Central Jet connector lets you access the data stored Business Central databases using the Composer client using the available API.

Before you can establish a connection from Composer to Business Central databases, a connector server needs to be installed and configured. See Manage Connectors And Connector Servers for general instructions and Connect To Business Central for details specific to this connector.

After setting up the connector, create data sources that specify the necessary connection information and identify the data you want to use. See Manage Data Sources for more information. After you set up your data sources, create dashboards and visuals from from the data in these data sources. See Create Data Discovery Dashboards.

Feature Support

Connector support for specific features is shown in the following table.

Key: Y - Supported; **N** - Not Supported; N/A - not applicable

Feature	Supported?
Admin-Defined Functions	**Y**
Box Plots	**Y**
Custom SQL Queries	**N**
Derived Fields (Row-Level Expressions)	**Y**
Distinct Counts	**Y**
Fast Distinct Values	N/A
Group By Multiple Fields	**Y**
Group By Time	**Y**
Group By UNIX Time	**Y**
Histogram Floating Point Values	**Y**
Histograms	**Y**
Kerberos Authentication	**N**
Last Value	**Y**
Live Mode and Playback	**N**
Multivalued Fields	N/A
Nested Fields	N/A
Partitions	**N**

Feature	Supported?
Pushdown Joins for Fusion Data Sources	**N**
Schemas	**Y**
Text Search	N/A
TLS	**Y**
User Delegation	**N**
Wildcard Filters	**Y**
Wildcard Filters, Case-Insensitive Mode	**Y**
Wildcard Filters, Case-Sensitive Mode	**Y**

Connect to Business Central

You connect to Business Central using their provided API.

When setting up a connection, provide the following:

- A name for the connection.
- The **Base Url**, for example: `api.businesscentral.dynamics.com`.
- The **Tenant Id** provided by Business Central.
- The **Client Id** and the **Client Secret** for authentication.
- **Company** - The name of the company you're connecting to.

Other optional fields are not supported at this time.

Manage Cloudera Connectors

Composer provides connectors to the following Cloudera data stores:

- Cloudera Impala is a query engine that accesses data stored in clusters running Apache Hadoop.
- Cloudera Search enables searches of data stored in Hadoop and provides a simple full-text interface to conduct those searches. Cloudera Search supports Cloudera's open source Hadoop platform - Cloudera Distributed Hadoop (CDH). Composer also connects to CDH via the HDFS connector .

Cloudera also supports a secure CDH cluster using Kerberos authentication so that access requests can be validated and confirmed. Composer can connect Connect To A Kerberized CDH Cluster.

- Manage The Impala Connector
- Connect To Impala With TLS (SSL) Enabled
- Connect To A Kerberized CDH Cluster
- Work With Distinct Counts On Cloudera Impala
- Enable User Delegation
- Apply User Delegation To A Connection
- Manage The Cloudera Search Connector

Manage the Impala Connector

The Composer Cloudera Impala™ connector allows you to visualize huge volumes of data stored in their Hadoop cluster in real time and with no ETL. Composer supports Impala versions 3.2 - 3.4.

Before you can establish a connection from Composer to Cloudera Impala storage, a connector server needs to be installed and configured. See Manage Connectors And Connector Servers for general instructions and Connect To Impala for details specific to the Cloudera Impala connector.

After setting up the connector, create data sources that specify the necessary connection information and identify the data you want to use. See Manage Data Sources for more information. After you set up your data sources, create dashboards and visuals from from the data in these data sources. See Create Data Discovery Dashboards.

This topic describes:

- Feature Support
- Impala Authentication
- Connect To Impala
- Impala Table Settings

See also:

- Work With Distinct Counts On Cloudera Impala
- Enable Data Sharpening For Cloudera Impala Data Sources

Feature Support

Connector support for specific features is shown in the following table.

Key: Y - Supported; **N** - Not Supported; N/A - not applicable

Feature	Supported?	Notes
Admin-Defined Functions	**Y**	
Box Plots	**Y**	
Custom SQL Queries	**Y**	If you need to access a BigQuery partition, explicitly include an alias for the built in partition

Feature	Supported?	Notes
		column in your select clause, such as `select *, _PARTITIONTIME as pt from projectId.datasetId.tableId`.
Derived Fields (Row-Level Expressions)	Y	
Distinct Counts	Y	Cloudera Impala connectors can receive only a single distinct count field in a query.
Fast Distinct Values	N/A	
Group By Multiple Fields	Y	
Group By Time	Y	
Group By UNIX Time	Y	
Histogram Floating Point Values	Y	
Histograms	Y	
Kerberos Authentication	Y	
Last Value	Y	
Live Mode and Playback	Y	
Multivalued Fields	N/A	
Nested Fields	N/A	
Partitions	Y	
Pushdown Joins for Fusion Data Sources	Y	
Schemas	Y	
Text Search	N/A	
TLS	Y	
User Delegation	Y	
Wildcard Filters	Y	
Wildcard Filters, Case-Insensitive Mode	Y	
Wildcard Filters, Case-Sensitive Mode	Y	

The Cloudera Impala connector also supports Progress reporting. Progress reporting support allows the connector to report the progress of a running query. On the UI, this shows as **Reading *nn*%** in the upper left corner of a visual.

Impala Authentication

Support is provided for passing along credentials for users with access privileges to Impala source. Delegation allows for Impala queries to be issued with the privileges from a specified user. This is available in the Connection page and is set as the **Do As User** list. See Enable User Delegation and Apply User Delegation

To A Connection.

Connect to Impala

When setting up an Impala connection, you need to provide the following.

1. Specify the JDBC URL. You can connect to your Impala data source using either simple user credentials authentication or Kerberos authentication with optional SSL encryption. Refer to Connecting to Impala on Kerberized CDH or Connecting to Impala with TLS (SSL) for more details on the configuration.

   Composer enables you to connect either to a single Impala node or to multiple nodes within a cluster. To connect to a single Impala node, specify a JDBC URL in the following format:

   ```
   jdbc:hive2://<impala_host>:<port>/;auth=noSasl
   ```

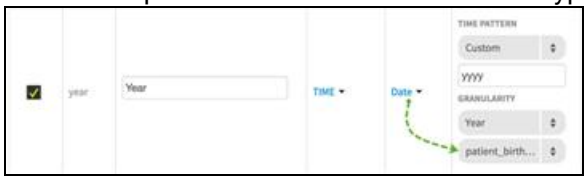
   To connect to multiple Impala nodes, specify the required JDBC URLs separated by commas. The URLs will be used in a round-robin fashion. Keep in mind that such a connection will be valid as long as there is at least one available node. If all the nodes can not be reached, then the connection won't be validated.

2. If Impala authentication has been set up, provide a user name and password.
3. To allow for Impala user delegation, select the appropriate custom user attribute from the **Do As User** drop-down list (set up by the Composer supervisor or administrator). This basically allows Composer to pass along credentials for the specified user with access rights to Impala. See Enable User Delegation and Apply User Delegation To A Connection.
4. Select **Validate**. If successfully validated, the connection is saved.

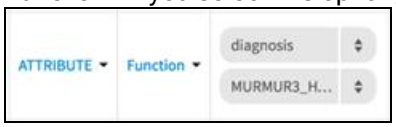
Impala Table Settings

Time-based fields can be configured for partitioning in an Impala data source configuration using the **Partition** column on the Fields tab of the data source. The following options are available:

- No (partitioning to be done)
- Date - this option is available for the Time field type. If you select this option, the list of the partitioned columns will be displayed in the Configure column.



- Function - If you select this option, the list of the partitioned columns and supported MURMUR3_HASH function will be displayed in the Configure column.



Numeric and time-based fields can be edited using the Fields tab:

- Numeric type Number - ability to select a default aggregation function
- Time fields - ability to define the default time pattern and granularity; if the time field provides granularities of hour, minute and second, then a time zone label may be applied.

Select the checkbox in the **Distinct Count** column for any fields if a distinct count is needed. For more information, see Work With Distinct Counts On Cloudera Impala.

Connect to Impala with TLS (SSL) Enabled

You can connect to the Impala data source with TLS/SSL network-level encryption to secure your data while working with your data source.

Prerequisites

For Impala:

- Before you proceed, make sure that TLS is configured for Impala using either Cloudera Manager or the Command Line interface.
- Impala's TLS configuration requires an x509 certificate that will identify the Impala daemon to clients during TLS connections. Production usage of TLS usually implies purchasing the necessary certificates from a commercial Certificate Authority (CA), while development environments can use self-signed certificates. If you have either a **rootCA** from the trusted CA or a **self-signed certificate** in PEM format you can verify your Impala TLS configuration using the `openssl` utility:

```
openssl s_client -connect <impala_host>:port -CAfile <certificate.pem>
```

For Composer Server/Impala Connector:

- There is no particular configuration related to TLS from the point of view of Composer components. However, the client must have a Java **truststore** with a correct certificate (for example, a root certificate provided by some CA) installed. This means that the **truststore** must be accessible to the Composer Server/Impala connector.
- To list all the certificates installed in the Java truststore, use the `keytool` utility:

```
keytool -v -list -keystore <path_to_truststore> -storetype jks -storepass <truststore_password>
```

After you have the Java truststore configured, enabling SSL from Composer's perspective is a matter of composing the correct JDBC URL.

Creating a JDBC URL with the TLS Parameters

To specify the TLS-related parameters, use the following template for a JDBC URL:

```
jdbc:hive2://<impala_host>:<port>/;ssl=true;sslTrustStore=<path_to_truststore>;
trustStorePassword=<truststore_password>;auth=noSasl
```

where:

- `ssl=true` is a required parameter for enabling TLS encryption.
- `path_to_truststore` is the path to a Java **truststore** which contains either a certificate issued by a trusted CA or a self-signed certificate (not recommended and shouldn't be used in a production environment).

  Note: Make sure that the Composer server/connector process has read access privileges to the **truststore** file.

- `truststore_password` is the password to access the **truststore**.
- `auth=noSasl` is a required parameter when no authentication or simple user/password authentication is used.

Use TLS Encryption with Kerberos Authentication

See Connect To A Kerberized CDH Cluster for more details on enabling Kerberos authentication. The template for a JDBC URL containing both TLS and Kerberos parameters is as follows:

```
jdbc:hive2://<impala_host>:<port>/;principal=<impala_principal>;ssl=true;
sslTrustStore=<path_to_truststore>;trustStorePassword=<truststore_password>
```

You do not need to specify the `auth=noSasl` parameter when using Kerberos authentication.

Connect to a Kerberized CDH Cluster

A secure CDH Cluster uses Kerberos authentication to validate and confirm access requests. You can set up Composer to connect to the secure CDH Cluster using the instructions provided below. Before establishing a connection to either type of cluster, review the prerequisites and be sure to obtain your Kerberos credentials.

- Obtain Kerberos Credentials
- Configure An Impala Connector
- Configure A Cloudera Search Connector
- Connect To A Kerberized Data Source
- Use TLS Encryption With Kerberos Authentication

Prerequisites

- To enable Kerberos for CDH distribution using Cloudera manager, see Cloudera's documentation Configuring Authentication in Cloudera Manager.
- Kerberos authentication requires precise time correspondence on all instances to work properly. You need to enable the Network Time Protocol service in your network. For more information, access the topic Using the Network Time Protocol to Synchronize Time.

Obtain Kerberos Credentials

Each microservice must have its own unique identifier called a principal. Perform the following steps:

1. Install the Kerberos client on the machine where Composer Impala connector is installed.
2. Generate the Kerberos principal and corresponding keytab for Composer microservice. Before you proceed, make sure that:

i. Composer or a Connector is running on a node with proper Kerberos configuration: `/etc/krb5.conf` or similar location for your Linux distribution.

ii. The Kerberos realm on your environment is the same as the realm specified in the `kdc.conf` file from Impala server.

3. Check the Kerberos configuration (that is, `krb5.conf`) and validity of the principal and keytab pair using MIT Kerberos client:

```
kinit -V -k -t zoomdata_principal .keytab zoomdata_principal@KERBEROS.REALM
```

4. Make the keytab accessible for the Composer Server or a connector:

```
sudo mkdir /etc/zoomdata
sudo mv zoomdata_principal.keytab /etc/zoomdata
sudo chown zoomdata:zoomdata /etc/zoomdata/zoomdata_principal.keytab
sudo chmod 600 /etc/zoomdata/zoomdata_principal.keytab
```

Configure an Impala Connector

1. Create or update the file named `/etc/zoomdata/edc-impala.properties`. If this file already exists, verify that the information below exists in the file:

```
kerberos.krb5.conf.location=/etc/krb5.conf
kerberos.service.account.authentication=true
kerberos.service.account.principal=zoomdata_principal@KERBEROS.REALM
kerberos.service.account.keytab.location=/etc/zoomdata/zoomdata_principal.keytab
```

2. Restart the Impala connector:

```
sudo systemctl restart zoomdata-edc-impala
```

Configure a Cloudera Search Connector

1. Create or update the file named `/etc/zoomdata/edc-cloudera-search.properties`. If this file already exists, verify that the information below exists in the file:

```
kerberos.krb5.conf.location=/etc/krb5.conf
kerberos.service.account.authentication=true
kerberos.service.account.principal=zoomdata_principal@KERBEROS.REALM
kerberos.service.account.keytab.location=/etc/zoomdata/zoomdata_principal.keytab
```

2. Restart the Cloudera Search microservice:

```
sudo systemctl restart zoomdata-edc-cloudera-search
```

Connect to a Kerberized Data Source

You are now ready to create the Cloudera Search or Impala source:

1. Open a new browser window and log into Composer.
2. Select **Sources**.
3. Select **Cloudera Search or Impala**.
4. Specify the name of your source and add a description (if desired). Select **Next**.
5. On the Connection tab, define the connection source. You can use an existing connection, if available, or create a new one. To create a new connection, select the **Input New Credentials** option button and specify the connection name and JDBC URL. Make sure that you enter the JDBC URL in the correct format.

   For Impala, specify:

   ```
   jdbc:hive2://<impala_host>:21050/;principal=<impala_principal@KERBEROS.REALM>
   ```

   For Cloudera Search, specify:

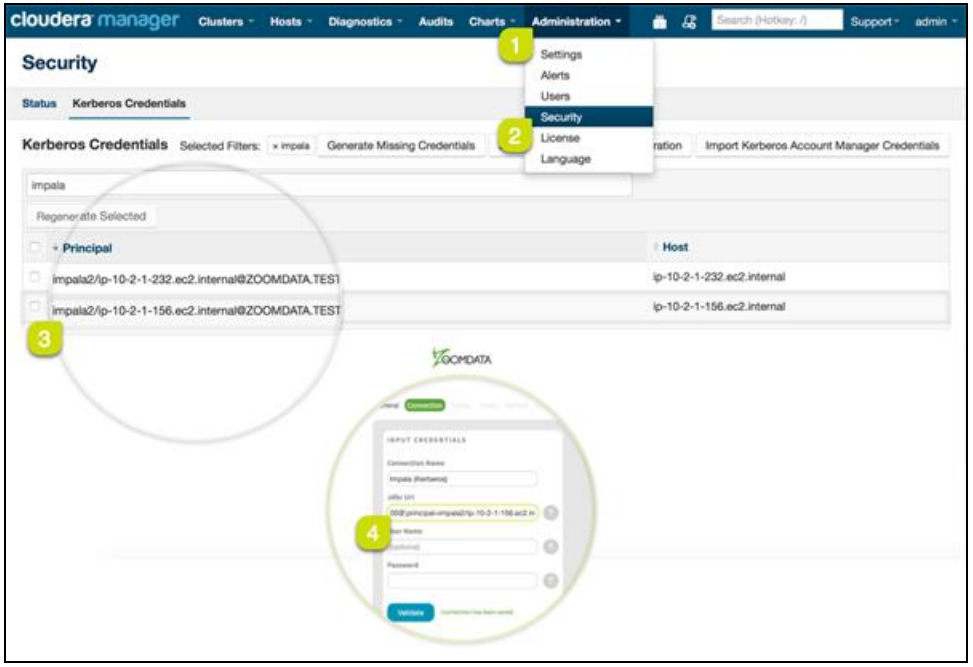
   ```
   cloudera.domain:2181/solr
   ```

   The JDBC URL for Cloudera Search needs to be the zookeeper URL of the Kerberized cluster.

   Replace the placeholders as follows:

i. For <impala_host>, enter the IP address/host name of the Impala node you are connecting to.

ii. For <impala_principal@KERBEROS.REALM>, enter the principal of the node you are connecting to. To get the list of all Impala principals, navigate to Cloudera Manager > Administration > Security > Kerberos Credentials.



6. Select **Validate**. After successful validation, the values are saved. Select **Next**.

> **Note:** If you run into connection issues, verify that the Composer Server was restarted successfully. Access the troubleshooting topic Verify The ComposerSymphony Server Restart for assistance.

You can continue configuring the data source as needed.

After you have completed the configuration, Composer begins accessing the data source using `zoomdata_principal@KERBEROS.REALM` authenticated by its keytab in `/etc/zoomdata/zoomdata_principal.keytab`.

Use TLS Encryption with Kerberos Authentication

See Connect To Impala With TLS (SSL) Enabled for more details.

Work With Distinct Counts on Cloudera Impala

Due to the structure of Cloudera Impala, you cannot build a visual using two or more metrics for which the Distinct Count option has been enabled. You can enable or disable the distinct counts for the specific fields in a data source on the Fields Tab of a data source.

Enable Data Sharpening for Cloudera Impala Data Sources

Data Sharpening works with certain partitioned Impala data sources. The partitioned field should be a time-based attribute and in a supported time format (for example, yyyy-MM-dd). Follow the steps below to set up Data Sharpening for an Impala data source.

Configure Data Sharpening for a Cloudera Impala data source configuration:

1. Log into Composer (either as an administrator or as a user who has been assigned to a group with data source management privileges).
2. Select **Sources** on the UI menu () or the top-level navigation menu, or select the **Sources** box on the Home page. The Sources page appears.
3. Select the appropriate data source configuration to edit it, then access the Fields tab of the data source.
4. Locate and select the time field you want to use as the driving time field. Select an appropriate time granularity in the **Data Details** section of the **Settings** side bar menu, then **Save** your changes. Consider the 10% rule to ensure Data Sharpening runs when you want it to. See When Data Sharpening Occurs for more information.
5. Select the **Global Settings** tab, and enable **Time Bar** if not enabled to access the data sharpening settings. See Configure Time Bar Defaults.
6. Select the time field you want to use as the driving time field in thedrop-downfor **Default Time Attribute**.
7. Enable the **Prefer Sharpening** toggle to enable sharpening and sharpening settings.
8. Optionally, use the **Max Queries** slider to specify the maximum number of queries used for Data Sharpening. The default maximum is 10 queries.
9. When your changes are complete, select **Save Settings** to save your changes.

Manage the Cloudera Search Connector

Composer supports Cloudera Search version 4.10 - 7.4.

Before you can establish a connection from Composer to Cloudera Search storage, a connector server needs to be installed and configured. See Manage Connectors And Connector Servers for general instructions and Connect To Cloudera Search for details specific to the Cloudera Search connector.

After setting up the connector, create data sources that specify the necessary connection information and identify the data you want to use. See Manage Data Sources for more information. After you set up your data sources, create dashboards and visuals from from the data in these data sources. See Create Data Discovery Dashboards.

Feature Support

Connector support for specific features is shown in the following table.

Key: Y - Supported; **N** - Not Supported; N/A - not applicable

Feature	Supported?	Notes
Admin-Defined Functions	**N**	
Box Plots	**N**	The Apache Solr versions prior to 5.3 used by Cloudera Search do not have the percentile aggregations required for box plots built in.
Custom SQL Queries	**N**	If you need to access a BigQuery partition, explicitly include an alias for the built in partition column in your select clause, such as `select *, _PARTITIONTIME as pt from projectId.datasetId.tableId`.
Derived Fields (Row-Level Expressions)	**N**	
Distinct Counts	**Y**	
Fast Distinct Values	**Y**	
Group By Multiple Fields	**N**	
Group By Time	**N**	
Group By UNIX Time	**N**	
Histogram Floating Point Values	**N**	
Histograms	**N**	
Kerberos Authentication	**Y**	
Last Value	**N**	
Live Mode and Playback	**Y**	

Feature	Supported?	Notes
Multivalued Fields	**Y**	
Nested Fields	**N**	
Partitions	N/A	
Pushdown Joins for Fusion Data Sources	**N**	
Schemas	N/A	
Text Search	**Y**	You can sort keyword searches by Best Match and Most Recent (when you select a preferred time field from the source). Filter your search results by selecting fields in the Filter modal. Select **Clear All** to clear filtered search results.
TLS	**Y**	
User Delegation	**Y**	
Wildcard Filters	**Y**	
Wildcard Filters, Case-Insensitive Mode	**N**	Case-sensitivity cannot be enforced. Consequently, neither case-sensitive or case-insensitive wildcard filters are supported.
Wildcard Filters, Case-Sensitive Mode	**N**	Case-sensitivity cannot be enforced. Consequently, neither case-sensitive or case-insensitive wildcard filters are supported.

Connect to Cloudera Search

When establishing a connection to a new Cloudera Search server, you need to provide the following:

- The connection name
- The Base URL

Manage the Couchbase Connector

The Composer Couchbase connector lets you access the data available in your Couchbase Data Platform storage using the Composer client. The Composer Couchbase connector supports Couchbase version 6.0.1 and Couchbase Community Edition 6.0.0.

Before you can establish a connection from Composer to Couchbase Data Platform, a connector server needs to be installed and configured. See Manage Connectors And Connector Servers for general instructions and Connect To Couchbase for details specific to the Couchbase connector.

After setting up the connector, create data sources that specify the necessary connection information and identify the data you want to use. See Manage Data Sources for more information. After you set up your data sources, create dashboards and visuals from from the data in these data sources. See Create Data Discovery Dashboards.

Feature Support

Connector support for specific features is shown in the following table.

Key: Y - Supported; **N** - Not Supported; N/A - not applicable

Feature	Supported?	Notes
Admin-Defined Functions	**Y**	
Box Plots	**N**	
Custom SQL Queries	**Y**	If you need to access a BigQuery partition, explicitly include an alias for the built in partition column in your select clause, such as `select *, _PARTITIONTIME as pt from projectId.datasetId.tableId`.
Derived Fields (Row-Level Expressions)	**Y**	
Distinct Counts	**Y**	
Fast Distinct Values	N/A	
Group By Multiple Fields	**Y**	
Group By Time	**Y**	
Group By UNIX Time	**Y**	
Histogram Floating Point Values	**Y**	
Histograms	**Y**	
Kerberos Authentication	N/A	
Last Value	**N**	
Live Mode and Playback	**Y**	
Multivalued Fields	N/A	

Feature	Supported?	Notes
		The Couchbase & Couchbase Community Edition connector supports multivalued fields with some limitations. See the detailed description below this table.
Nested Fields	N/A	
Partitions	**N**	
Pushdown Joins for Fusion Data Sources	**N**	
Schemas	**Y**	
Text Search	N/A	
TLS	**Y**	
User Delegation	**N**	
Wildcard Filters	**Y**	
Wildcard Filters, Case-Insensitive Mode	**Y**	
Wildcard Filters, Case-Sensitive Mode	**Y**	

The following limitations exist for multivalued field support for the Couchbase connector (including nested fields when they are also multivalued fields):

- The COUNT, SUM, or AVG metric result for a computed single-value field will be inaccurate when it is combined with any multivalued field in the same query.
- The COUNT, SUM, or AVG metric result for a computed multivalued field will be inaccurate when it is combined with any other multivalued field in the same query.
- With the exception of the TEXT_TO_TIME row-level function, multivalued fields created using row-level expressions can only be used as a visual's group or metric (and in filters when the same multivalued field is used as the visual's group or metric).

The following examples show *valid* metric combinations:

```
COUNT(single_value_field1),SUM(single_value_field2),AVG(single_value_field3),
DISTINCT_COUNT(single_value_field4), MIN(single_value_field5), MAX(single_value_field6)
```

```
COUNT(multivalue_field),SUM(multivalue_field),AVG(multivalue_field),
DISTINCT_COUNT(multivalue_field), MIN(multivalue_field), MAX(multivalue_field)
```

```
DISTINCT_COUNT(single_value_field), MIN(multivalue_field1), MAX(multivalue_field2)
```

The following examples show *invalid* combinations of metrics:

```
COUNT(single_value_field), SUM(single_value_field), AVG(single_value_field), MIN(multivalue_field)
```

```
COUNT(multivalue_field1), SUM(multivalue_field2), AVG(multivalue_field1), MAX(multivalue_field2)
```

Connect to Couchbase

To establish a connection to Couchbase, the default port is 8138. If SSL/TLS authentication is required, see Configure Couchbase TLS/SSL Support.

By default, the Couchbase connector uses ports 8091, 8093, and 11210 in unencrypted mode and ports 18091, 18093, and 11207 in encrypted mode. Please make sure that these ports are not blocked by a firewall and are accessible to the connector.

Ordinarily, the port shown after the colon in a Couchbase URL is ignored and the defaults are used (for example, port 9999 will be ignored in `couchbase://localhost:9999`). If your Couchbase installation does not use the default ports, add at least one of the following HTTP parameters and preferably one of the carrier parameters to the Couchbase connection string. An HTTP parameter is required.

Type	Property	Default Value
HTTP	`bootstrapHttpDirectPort`	8091
HTTP-SSL	`bootstrapHttpSslPort`	18091
Carrier	`bootstrapCarrierDirectPort`	11210
Carrier-SSL	`bootstrapCarrierSslPort`	11207

Here is an example of a valid connection string SSL (encrypted case) example using non-default HTTP port 18777 and non-default carrier port 11777:

```
couchbase://localhost?bootstrapHttpSslPort=18777&bootstrapCarrierSslPort=11777
```

Configure Couchbase TLS/SSL Support

Couchbase can use the TLS protocol for:

- Full encryption of client-side traffic with server authentication. See Connecting with SSL.
- Client authentication using X.509 certificates. This form of authentication is suitable for further Couchbase role-based access control. See Certificate-Based Authentication.

To support these capabilities in Composer's Couchbase connector, additional configuration is needed. See the following sections.

- Configure TLS Server Authentication
- Configure Client Certificate-Based Authentication For A Single Couchbase Connection
- Configure Client Certificate-Based Authentication For Multiple Couchbase Connections

Configure TLS Server Authentication

To enable TLS for the Couchbase server connection, add the following properties to the `edc-couchbase.properties` file:

```
com.couchbase.sslEnabled=true

com.couchbase.sslTruststoreFile=<path_to_truststore_storing_rootCertificate.jks>
com.couchbase.sslTruststorePassword=<optional_truststore_password>
```

The `com.couchbase.sslEnabled` property must be set to `true`. The path you specify for the `com.couchbase.sslTruststoreFile` property must be accessible by the Composer Couchbase connector.

For information about modifying connector properties in property files, see Connector Properties And Property Files.

Configure Client Certificate-Based Authentication for a Single Couchbase Connection

If your installation only needs client certificate-based authentication for a single Composer Couchbase connection, specify the certificate-based authentication using connector properties. You must have a keystore containing the client's identity as well as a keystore or truststore containing the root certificate required to authenticate the server. Use one of the following setup options:

- Use a password-protected Java keystore that contains the client's identity and the root certificate. If you select this option, enable client certificate-based authentication by specifying the following connector properties in the `edc-couchbase.properties` file:

```
com.couchbase.sslEnabled=true
```

```
com.couchbase.sslKeystoreFile=<path_to_keystore_storing_clientID_and_rootCertificate.jks>
com.couchbase.sslKeystorePassword=<mandatory_keystore_password>

com.couchbase.certAuthEnabled=true
```

The `com.couchbase.sslEnabled` and `com.couchbase.certAuthEnabled` properties must be set to `true`.

- Use a password-protected Java keystore that contains only the client's identity and a separate Java truststore containing the root certificate. This approach allows you to share a single universal truststore containing all required root certificates among different services.

  If you select this option, enable client certificate-based authentication by specifying the following connector properties in the `edc-couchbase.properties` file:

```
com.couchbase.sslEnabled=true

com.couchbase.sslTruststoreFile=<path_to_truststore_storing_rootCertificate_only.jks>
com.couchbase.sslTruststorePassword=<optional_truststore_password>

com.couchbase.sslKeystoreFile=<path_to_keystore_storing_clientID_only.jks>
com.couchbase.sslKeystorePassword=<mandatory_keystore_password>

com.couchbase.certAuthEnabled=true
```

  The `com.couchbase.sslEnabled` and `com.couchbase.certAuthEnabled` properties must be set to `true`.

For information about modifying connector properties in property files, see Connector Properties And Property Files.

Configure Client Certificate-Based Authentication for Multiple Couchbase Connections

If your installation needs client certificate-based authentication for multiple Composer Couchbase connections, specify the client certificate-based authentication information using Composer connection parameters in the definition of each connection. You must have a keystore containing the client's identity as well as a keystore or truststore containing the root certificate required to authenticate the server.

Two new input boxes have been added in the Composer UI when you create a Couchbase connection. Assuming your keystore contains both the required client identity and root certificate information, specifying the keystore information using these two input boxes is sufficient.

- Use the **Key Store Path** box to specify the path to the keystore. This path must be accessible by the Composer Couchbase connector, so your system administrator must upload all client keystores to the machine where the connector is running before you define the Couchbase connection in the UI.

- Use the **Key Store Password** box to specify the mandatory password for the keystore.

If your installation only uses the TLS keystore to store the client's identity and requires a separate truststore to store the root certificate, specify the following connector truststore properties in the `edc-couchbase.properties` file:

```
com.couchbase.sslEnabled=true

com.couchbase.sslTruststoreFile=<path_to_truststore_storing_rootCertificate_only.jks>
com.couchbase.sslTruststorePassword=<optional_truststore_password>
```

For information about modifying connector properties in property files, see Connector Properties And Property Files.

Manage the Dremio Connector

The Composer Dremio connector lets you access the data available in Dremio from supported versions of MySQL and MS SQL Server data stores using the Composer client. The Composer Dremio connector supports Dremio Community and Enterprise Editions version 4.1 through 4.8.

Before you can establish a connection from Composer to Dremio storage, a connector server needs to be installed and configured. See Manage Connectors And Connector Servers for general instructions and Connect To Dremio for details specific to the Dremio connector.

After setting up the connector, create data sources that specify the necessary connection information and identify the data you want to use. See Manage Data Sources for more information. After you set up your data sources, create dashboards and visuals from from the data in these data sources. See Create Data Discovery Dashboards.

Feature Support

Connector support for specific features is shown in the following table.

Key: Y - Supported; **N** - Not Supported; N/A - not applicable

Feature	Supported?
Admin-Defined Functions	**N**
Box Plots	**Y**
Custom SQL Queries	**N**
Derived Fields (Row-Level Expressions)	**N**
Distinct Counts	**Y**
Fast Distinct Values	N/A
Group By Multiple Fields	**Y**
Group By Time	**Y**
Group By UNIX Time	**Y**
Histogram Floating Point Values	**Y**
Histograms	**Y**
Kerberos Authentication	**N**
Last Value	**Y**
Live Mode and Playback	**Y**
Multivalued Fields	N/A
Nested Fields	N/A

Feature	Supported?
Partitions	N/A
Pushdown Joins for Fusion Data Sources	**N**
Schemas	**Y**
Text Search	N/A
TLS	**N**
User Delegation	**N**
Wildcard Filters	**Y**
Wildcard Filters, Case-Insensitive Mode	**Y**
Wildcard Filters, Case-Sensitive Mode	**N**

Connect to Dremio

The Dremio JDBC driver 3.0.6 is used by and included in the Dremio connector. Driver drop-in is supported, so you can replace this JDBC driver if you want. By default, the Dremio connector uses port 8142.

To create a Dremio connector, you must provide three parameters:

- The JDBC URL in a format supported by Dremio:

```
jdbc:dremio:direct=<DREMIO_COORDINATOR>:<JDBC_PORT>[;schema=<OPTIONAL_SCHMEMA>]
```

  or

```
jdbc:dremio:zk=<ZOOKEEPER_QUORUM>:<ZK_PORT>[;schema=<OPTIONAL_SCHEMA>]
```

- A valid user name
- A valid password.

Live Mode and Playback Considerations

Dremio supports reflections instead of indices. For more information about Dremio reflections, see https://docs.dremio.com/acceleration/. Because Dremio does not support indices, Composer cannot accurately determine which date and number fields are playable, so it treats ***all*** date and time fields as playable, by default. Consequently, use livemode and playback judiciously when you are working with large data sets. Switch playback and live mode on only for visuals you know that the corresponding SQL queries are covered by a reflection and can be accelerated.

If you don't use Dremio reflections and are concerned that the default playability of all date and number fields might be misused and impact performance, you can switch this feature off using the connector configuration property: `metadata.detection.mark-all-date-and-int-fields-playable`. Valid values are `true` (on) and `false` (off). The default setting for this property is `true`.

Manage the Elasticsearch Connector

The Composer Elasticsearch connector lets you access the data available in the Elasticsearch storage using the Composer client. The Composer Elasticsearch connector supports the following Elasticsearch versions.

- Elasticsearch 7.0 - 7.17
- Elasticsearch 8.1 - 8.11

Important: You cannot import or export Elasticsearch data sources (or the visuals and dashboards that use those Elasticsearch data sources) if the version of the Elasticsearch connector in the Composer environment is different from the version used by the data sources. For example, you cannot import an Elasticsearch 7 data source you have exported if your Composer environment only has an Elasticsearch 8 connector defined. When you change connector versions in your Composer environment, we recommend that you also create new data source configurations (and associated visuals and dashboards) for the newer version.

Before you can establish a connection from Composer to Elasticsearch storage, a connector server needs to be installed and configured. See Manage Connectors And Connector Servers for general instructions and Connect To Elasticsearch for details specific to the Elasticsearch connector.

After setting up the connector, create data sources that specify the necessary connection information and identify the data you want to use. See Manage Data Sources for more information. After you set up your data sources, create dashboards and visuals from from the data in these data sources. See Create Data Discovery Dashboards.

This section covers the following topics:

- Elasticsearch Last Value Processing
- Elasticsearch 7 Composite Aggregation
- Elasticsearch Source Document Storage Configurations
- Inner Hits Configuration Property
- Support Of X-Pack For Elasticsearch

Composer Elasticsearch Connector Feature Support

Connector support for specific features is shown in the following table.

Key: Y - Supported; **N** - Not Supported; N/A - not applicable

Feature	Supported?	Notes
Admin-Defined Functions	**N**	
Box Plots	**Y**	
Custom SQL Queries	**Y**	If you need to access a BigQuery partition, explicitly include an alias for the built in partition column in your select clause, such as `select *, _PARTITIONTIME as pt from projectId.datasetId.tableId`.
Derived Fields (Row-Level Expressions)	**Y**	
Distinct Counts	**Y**	
Fast Distinct Values	**Y**	
Group By Multiple Fields	**Y**	
Group By Time	**Y**	
Group By UNIX Time	**N**	
Histogram Floating Point Values	**Y**	
Histograms	**Y**	
Kerberos Authentication	**N**	
Last Value	**Y**	
Live Mode and Playback	**Y**	
Multivalued Fields	**Y**	
Nested Fields	**Y**	
Partitions	N/A	
Pushdown Joins for Fusion Data Sources	**N**	
Schemas	N/A	
Text Search	**Y**	You can sort keyword searches by Best Match and Most Recent (when you select a preferred time field from the source). Filter your search results by selecting fields in the Filter modal. Select **Clear All** to clear filtered search results.
TLS	**Y**	

Feature	Supported?	Notes
User Delegation	N	
Wildcard Filters	Y	
Wildcard Filters, Case-Insensitive Mode	N	Case-sensitivity cannot be enforced. Consequently, neither case-sensitive or case-insensitive wildcard filters are supported.
Wildcard Filters, Case-Sensitive Mode	N	Case-sensitivity cannot be enforced. Consequently, neither case-sensitive or case-insensitive wildcard filters are supported.

Connect to Elasticsearch

When establishing a connection to an Elasticsearch data store, make sure you:

1. Specify the connection string in the following format:

Protocol	Connection String Format	Example
HTTP/HTTPS	`<schema>://<host1>:<port1>,...,<hostN>:<postN>/<prefix>`	`http://ip-10-2-2-241.ec2.internal:80/es`
Transport/Transports	`<schema>://<host1>:<port1>,...,<hostN>:<portN>`	`transports://10.2.2.2:9010,10.2.2.3:9010`

   where `<schema>` is the protocol that you want to use:

- `http` or `https` (with SSL support)
- `transport` or `transports` (with SSL support)

Bear in mind that you must specify the hosts within one cluster.

> **Important:** Elasticsearch 8 does not support transports. Use an `http://` or `https://` connection when connecting to or upgrading to Elasticsearch 8 by updating your connection string as needed, for example, replace `transport://1.1.1.1:9300,2.2.2.2:9300` with `https://localhost:9200`.

2. Specify your Elasticsearch **cluster name**.
3. If required, specify your Elasticsearch **User Name** and **Password**.
4. Select **Validate** to confirm your connection.

To connect to your Elasticsearch cluster and data set secured by X-Pack, see Support Of X-Pack For Elasticsearch.

Connect to Elasticsearch with a Configured Custom Certificate

If your Elasticsearch cluster is configured with a custom certificate, you should configure a truststore for the Elasticsearch connector.

Connect to an Elasticsearch data store with a configured custom certificate

1. Copy a truststore to the machine on which the Elasticsearch connector is running.
2. Add the following lines to file the appropriate Elasticsearch `jvm` file.

i. For Linux: , `/etc/zoomdata/edc-elasticsearch-7.0.jvm`, or `/etc/zoomdata/edc-elasticsearch-8.0.jvm`. Copy these files from the `/opt/zoomdata/conf` directory if a copy is not in `/etc/zoomdata/`.

ii. For Windows: , `<install-path>/edc-elasticsearch-7.0.jvm` or `<install-path>/edc-elasticsearch-8.0.jvm`. Copy these files from the `<install-path>/conf` directory if a copy is not in `<install-path>`.

```
-Djavax.net.ssl.trustStore=<path_to_truststore>
-Djavax.net.ssl.trustStorePassword=<truststore_password>
```

Replace:

i. `<path_to_truststore>` with an absolute path to your truststore

ii. `<truststore_password>` with a password for your truststore

Connect to Elasticsearch Using Amazon Web Services Authentication

You can connect the Composer Elasticsearch connectors to your Elasticsearch data store using Amazon Web Services (AWS) credentials. After connecting, the Composer Elasticsearch connectors work with AWS Elasticsearch without any restrictions.

Important: AWS does not support Elasticsearch v8.0.

Connect the Elasticsearch connector to your Elasticsearch data store using AWS authentication:

1. Specify the AWS credentials in a standard AWS-format credentials file (see Format of the credentials file) and store the file in the Elasticsearch connector's file system.

2. Edit the Elasticsearch properties file (`edc-elasticsearch-7.0.properties`) and locate or add the `elasticsearch.aws.show-aws-connection-params` property to the file. This property indicates whether AWS-specific connection parameters should be shown when a new connector is created or the connection properties of an existing connector are refreshed. Valid values are `true` or `false`. The default is `false` (users will *not* see new AWS connection parameters).

   Set the value of this property to `true` and save the properties file. For more information about Composer properties files, see Configuration Property Files.

3. Create a new Elasticsearch connection or edit an existing one (see Add Data Store Connections).

   The AWS connection parameters appear in the UI. Supply values for them as described in the following table.

Parameter	Specify	Description
`AUTHENTICATE_WITH_AWS_CREDENTIALS`	`true`	Indicate whether authentication with AWS credentials should be used. Note that the Elasticsearch username / password specification and AWS credentials cannot be simultaneously used for authentication. Valid values are `true` or `false`; the default is `false` (AWS credentials are not used).
`AWS_REGION`	a valid region name	Specify the AWS region where the target Elasticsearch service is running. This parameter is optional when host names in the connection string have the standard format `<domain>.<region>.es.amazonaws.com` (the region name can be extracted from the host name). However, when specified, it has priority over the region name included in the host name.
`AWS_PROFILES_CONFIG_PATH`	the path to the credentials file	Specify the location of the AWS credentials file in the Elasticsearch connector's file system. This parameter is optional when the credentials file is in the default location (`~/.aws/credentials` for the user of the connector).
`AWS_PROFILE`	a valid profile name in the AWS credentials file	Specify the profile to use within the AWS credentials file. This parameter is optional if you choose to use the `default` profile in the AWS credentials file.

4. After you have specified all parameters necessary for the connection definition, save it. See Add Data Store Connections.

Elasticsearch Data Source Configuration Notes

When setting up an Elasticsearch data source configuration, select the indices and aliases to be queried, and select the fields to be handled. You can do this in three steps:

1. Select indices and aliases to be queried on the Source Creation tab.

2. Select indices **Manually** or **Automatically**.

i. If you want to get the data only from specific indices, select the **Manually** option and choose the corresponding indices from the list **Select Indices**.

ii. The **Automatically** option is more flexible. It lets you set the pattern by which the indices will be selected automatically.

For this option, you can select one of the pattern types. Note that when no indices match the pattern while querying, your visuals are returned empty.

a. **Native** - specify the pattern for index names. Use an asterisk (*) to replace one character or a set of characters.

For example, you want to get all the indices whose name starts with ***log*** and ends with ***16***. In this case, specify the following pattern:

```
log*16
```

b. **Time Based** - set the time pattern to get the matching indices. Check the supported date and time patterns.

For example, the time pattern YYYY-MM will return all the indices, whose name will match the pattern in the following examples. Note that if the Index Name includes text with the time and date pattern, you need to enclose the text portion in brackets [ ]:

Examples:

Index Name	Pattern
2022-01	YYYY-MM
2022-3	YYYY-Q
10:23:11	HH:MM:SS
logstash-2022-06-14	[logstash-]YYYY-MM-DD

Note: The fields for indexes are not refreshed. If new fields are added to your data source, they are added to Composer only after you select the **Manual Refresh** () button on the **Cache** tab of the data source configuration. If there are some changes in the existing fields (for example, if a field has been removed) they won't be applied.

Note: Filtering by type is not supported.

When you connect to your Elasticsearch data source, the additional service field **type** is added. The **type** field contains all the selected Elasticsearch types you can visualize as attributes on your visuals.

Distinct Counts and Percentiles in Elasticsearch

Distinct count and percentiles metrics return approximate values in Elasticsearch. The precision of the result returned by distinct count metric depends on the precision threshold setting (default value is 1000).

You can change the value of the precision threshold by setting the `elasticsearch.query.cardinality.precision.threshold` property in the `zoomdata.properties` file.

See Elasticsearch's documentation on the following for more information:

- For Elasticsearch version 7, see the following for percentiles and distinct count.
- For Elasticsearch version 8, see the following for percentiles and distinct count.

The table below lists all available properties that you can modify to work with Elasticsearch.

Property	Default	Use	Notes
elasticsearch.query.cardinality.precision.threshold	1000	control the level of accuracy of the distinct counts	The maximum supported value is 40000. However, Composer does not recommend to set such value as it may result in performance issues and the data source itself may return errors. For more info, refer to the Precision Control section by Elasticsearch.
elasticsearch.query.limit.nongrouped	10000	set the limit for the number of non-grouped records (per shard) to execute on.	
elasticsearch.query.limit.grouped	10000	set the limit for the number of grouped records (per shard) to execute on.	

If you need to change the default settings, add the corresponding properties (listed above) to the `zoomdata.properties` file and assign the required values. For more details, see Managing Configurations .

Tokenization in Elasticsearch

Keep in mind that Elasticsearch, by default, tokenizes or analyzes fields that are of type `text`. As a result, strings consisting of two or more words may become separate fields when connected to Composer (for example, city names like *Las Vegas*). To disable this process and ensure that a string field is not analyzed, specify its type as `keyword`:

```
City: {
       type: "keyword"
       }
```

To learn more about tokenization in Elasticsearch, see Get Trained Models API.

Elasticsearch Connector IP Address Data Type Support

The IP Address data type is supported for Elasticsearch data connectors. Fields of this type are treated as ATTRIBUTEs and can be used in:

- An Elasticsearch text search box. When searching via the text search, Composer also supports the CIDR notation for IP addresses as described in the Elasticsearch documentation (https://www.elastic.co/guide/en/elasticsearch/reference/current/ip.html).
- The Group By selection box.
- Filters, although Composer does not support CIDR notation in filters for an IP address field. An exact match is required.
- Row-level expressions. In row-level expressions, Composer treats IP addresses as strings and expect an exact match.

Elasticsearch Last Value Processing

There are situations in which the Elasticsearch connector cannot compute the Last Value metric correctly.

1. When the original value *is* available for a metric field, an error appears in either of the following situations:

i. Both metric and group fields are nested and related.

ii. Both metric and time fields are nested and related.

> **Note:** The Elasticsearch connector still may not always choose the maximum value among several values for the Last Value of a time field. This should happen only in the following cases:

i. When the metric field is nested and neither the group or time field is located in the same hierarchy (group and time fields are either at the root level or belong to another hierarchy)

ii. When the Last Value is an array.

2. When the original value *is not* available for a metric field (the result is fetched from doc values or stored fields), an error appears when the metric field is nested and the time field is not located on the same or lower level in the hierarchy.

> **Note:** The Elasticsearch connector still may not always choose the maximum value among several values for the Last Value of a time field. This should happen only when the Last Value is an array.

Elasticsearch 7 Composite Aggregation

Composite aggregations are implemented by Elasticsearch 7 connectors. This support optimizes aggregations of Elasticsearch 7 data, except for queries with:

- histograms
- time groups with WEEK granularity
- multiple groups when group fields belong to different nested contexts.

An Elasticsearch 7 configuration property `elasticsearch.query.composite-agg.max-fetch-size` in the Elasticsearch 7 configuration file (`edc-elasticsearch-7.0.properties`) can be used to specify the maximum number of buckets to return for each query within a composite aggregation. Valid values must be greater than zero; the default value is 10000. This property corresponds to the Elasticsearch setting `search.max_buckets`, that also has a default value of 10000. If you elect to increase the value of the `elasticsearch.query.composite-agg.max-fetch-size property`, be sure to correspondingly increase the value of the Elasticsearch `search.max_buckets` setting.

Elasticsearch Source Document Storage Configurations

Elasticsearch data stores generally store the original JSON source documents passed when Elasticsearch performs its document indexing in the `_source` field in the index. However, some organizations disable the `_source` field to save storage and thus do not store the original JSON source documents.

Composer Elasticsearch 7 and 8 connectors support Elasticsearch data stores with any of the following source document configurations:

- Source documents disabled. For example:

```
...
"mappings": {
   "_source": {
      "enabled": false
   },
}
...
```

- Source documents enabled, but with some source exclusions. For example:

```
...
"mappings": {
   "_source": {
      "enabled": true,
      "includes": [
         "order_*"
      ],
      "excludes": [
         "order_items.*"
      ]
   }
},
...
```

- Source documents enabled, with no exclusions.

```
...
"mappings": {
   "_source": {
      "enabled": true
```

```
    }
},
...
```

The data sources created from connections to Elasticsearch data stores with any of these configurations function almost identically.

Known Issue Summary

The following known issues exist when the `_source` field is disabled or when it is enabled with exclusions:

- **Raw data presentation will vary, depending on the source from which the raw field data is fetched.** In particular, fused data sources may be affected where Elasticsearch indices with different mappings are joined (for example, when such indices are joined by an IP field and one of them allows Composer to fetch the data from the original documents but another requires Composer to fetch the data from doc values). See Raw Data Differences .
- **The last value metric may be computed incorrectly for nested fields.** Last value metrics are computed incorrectly when a time field is higher in the nested hierarchy or when a time field does not belong to the same hierarchy as the metric field.
- **Raw data is not available for some nested fields in multi-index Elasticsearch data sources.** When a nested field exists in some indices but is absent in other indices, the field's raw data is not available. It is represented as having a NULL value in all documents when it is included in a table.
- **Some nested fields are not searchable in multi-index data sources.** When a nested field exists in some indices but is absent in other indices, it cannot be used in text search queries. Such fields will not be used for text searches.

Raw Data Differences

The following Composer functions are impacted by the source document storage configuration of your Elasticsearch data stores:

- The data source collection preview on the Indices tab of the data source configuration
- Tables
- Text search results
- Last value metric computations.

Composer fetches the raw value of a field (including its last value metric result) in the following order:

1. If the field is stored, the stored value is fetched.
2. If the field is available in the original stored document, the value in the original stored document is fetched.
3. If doc values are available for the field and the field is not a text field, the doc value is fetched. For more information, see Text Field Raw Data Considerations.

Results vary based on the source from which the value was fetched, as described in the following table:

Value Fetched From	Differences from the original stored document
The stored value	■ NULL values in arrays are excluded. ■ Arrays may be sorted, completely or partially. ■ Numeric values may be approximated. ■ IPv6 addresses are normalized.
The doc value	■ NULL values in arrays are excluded. ■ Arrays may be sorted, completely or partially. ■ Duplicates in `string (keyword)` arrays are excluded, completely or partially. ■ Numeric values may be approximated. ■ IPv6 addresses are normalized.

Text Field Raw Data Considerations

Doc values are enabled by default for all fields, except for text fields. Consequently, by default, even if the original document is not stored or some fields are excluded from it, raw data is available for all fields, except for text fields.

An alternative structure called field data can be used for text fields. However, field data contains a set of terms for a text field, not its original value.

For this reason, raw data for a text field is not available if the text field is not stored in the index and cannot be fetched from the original document.

For example, the following mapping specifies that original documents should not be stored and declares two fields: `name` of type `keyword` and `description` of type `text`:

```
{
   "mappings": {
      "_source": {
         "enabled": false
      },
      "properties": {
         "name": {
            "type": "keyword"
         },
         "description": {
            "type": "text"
         }
      }
   }
}
```

Raw data for the text field `description` is not available in the index. To make raw data available for this field, declare it stored, as shown below:

```
{
   "mappings": {
      "_source": {
         "enabled": false
      },
      "properties": {
         "name": {
            "type": "keyword"
         },
         "description": {
            "type": "text",
            "store": true
         }
      }
   }
}
```

Inner Hits Configuration Property

Use the Composer Elasticsearch `elasticsearch.inner-hits.size property` to specify the maximum number of hits to return per `inner_hits` query (used for raw data requests involving nested fields). The default value is 100.

If you specify a value that is too small, the number of values returned for a nested field in a document with a large number of sub-documents may be limited. If you specify a value that is too large, excessive memory may be consumed.

Note: To learn more about this property, see inner hits.

Support of X-Pack for Elasticsearch

Composer allows you to connect to your Elasticsearch cluster and data set secured by X-Pack.

X-Pack is an add-on offering for Elasticsearch 7 that aims at securing the data on your cluster, and is included in Elasticsearch 8. Learn more about X-Pack.

Configure Cluster or Index Privileges for a User

To connect to the Elasticsearch cluster, you need to create an Elasticsearch user and configure the access privileges for this user.

The access permissions for the Elasticsearch user determine the scope of the data available for querying by Composer users.

To work with Elasticsearch data, use X-Pack to grant the following minimal access privileges to the Elasticsearch user:

- **Monitor** privileges for Elasticsearch Cluster
- **Manage** (to get the metadata) and **Read** (to read data) privileges for Index
- **Reference** Support Matrix

After the Elasticsearch user permissions are configured, you can proceed with connecting to a data source.

Added Libraries Required to Connect Using a Transport Protocol

Composer extracts specific libraries needed to support secured connections to Elasticsearch clusters over a transport protocol. The Composer Elasticsearch connector starts and works normally without these libraries, except when you want to use secured transport connections. To use secured transport connections, you must download and enable these libraries. If you attempt to establish a secured transport connection to an Elasticsearch cluster without the required libraries, an error will occur when you try to validate the connection.

To download and enable the transport-required libraries:

1. Download the required libraries to `/opt/zoomdata/lib/edc-elasticsearch-<x.x>/` for Linux, and for `<install-path>/lib/edc-elasticsearch-<x.x>/` Windows, where `<x.x>` is the version of Elasticsearch.

   ```
   mkdir -p /opt/zoomdata/lib/edc-elasticsearch-<x.x>
   wget -P /opt/zoomdata/lib/edc-elasticsearch-<x.x> <library URL>
   ```

   The following table provides a list of the required libraries for each supported Elasticsearch version, with URLs:

Composer Connector	Library Name	Version	License	URL
Elasticsearch 7.0	org.elasticsearch.client:x-pack-transport	7.0.0	Commercial Software End User License Agreement (https://www.elastic.co/eula)	https://artifacts.elastic.co/maven/org/elasticsearch/client/x-pack-transport/7.0.0/x-pack-transport-7.0.0.jar
	org.elasticsearch.plugin:x-pack-core			https://artifacts.elastic.co/maven/org/elasticsearch/plugin/x-pack-core/7.0.0/x-pack-core-7.0.0.jar
	com.unboundid:unboundid-ldapsdk	3.2.0	GPLv2, LGPLv2.1, or UnboundID Free Use License (https://docs.ldap.com/ldap-sdk/docs/LICENSE-UnboundID-LDAPSDK.txt)	https://mvnrepository.com/artifact/com.unboundid/unboundid-ldapsdk/3.2.0

For example:

```
mkdir -p /opt/zoomdata/lib/edc-elasticsearch-7.0
wget -P /opt/zoomdata/lib/edc-elasticsearch-7.0 https://artifacts.elastic.co/maven/org/elasticsearch/client/x-pack-transport/7.0.0/x-pack-transport-7.0.0.jar

wget -P /opt/zoomdata/lib/edc-elasticsearch-7.0 https://artifacts.elastic.co/maven/org/elasticsearch/plugin/x-pack-api/7.0.0/x-pack-api-7.0.0.jar

wget -P /opt/zoomdata/lib/edc-elasticsearch-7.0 http://central.maven.org/maven2/com/unboundid/unboundid-ldapsdk/3.2.0/unboundid-ldapsdk-3.2.0.jar
```

2. Update the `datasource.driver-config.jar-path` property value in the appropriate Elasticsearch connector property file to point to the directory path containing all the libraries you downloaded or to a comma-separated list combining all library paths. For example:

```
datasource.driver-config.jar-path=/opt/zoomdata/lib/edc-elasticsearch-7.0
```

or

```
datasource.driver-config.jar-path=/opt/zoomdata/lib/edc-elasticsearch-7.0/x-pack-transport-7.0.0.jar,
/opt/zoomdata/lib/edc-elasticsearch-7.0/x-pack-api-7.0.0.jar,
/opt/zoomdata/lib/edc-elasticsearch-7.0/unboundid-ldapsdk-3.2.0.jar
```

See Connector Properties And Property Files to determine the correct Elasticsearch property file to use and where to save it.

> **Note:** insightsoftware discourages changing properties in the `/opt/zoomdata/conf` directory (Linux) or `<install-path>/conf` (Windows). Copy the files you want to change to the `/etc/zoomdata` directory (Linux) or `<install-path>/conf-modify` (Windows) and change them there. This will ensure that your changes are not overwritten when Composer is next upgraded.

Quickly determine what changes you've made to a properties file using `diff` in Linux. For example:

```
diff /opt/zoomdata/conf/edc-<connector-name>.properties /etc/zoomdata/<edc-<connector-name>.properties
```

or

```
diff /opt/zoomdata/conf/zoomdata.properties /etc/zoomdata/zoomdata.properties
```

For Windows environments, use your preferred diff utility to compare the differences between your original and updated property files.

3. Restart the Elasticsearch connector microservice. For example:

```
sudo systemctl restart zoomdata-edc-elasticsearch-7.0
```

or

```
./bootstrap-composer.ps1 -ServicesAction restart
```

Connection Via HTTP or Transport Protocol and Using SSL

You can connect to your Elasticsearch cluster using either HTTP or transport protocols. SSL is optional for the HTTP connection but is required for transport connections when connecting to an X-Pack secured Elasticsearch cluster.

Manage the HDFS Connector

Composer offers connection to Cloudera's open source Hadoop platform - Cloudera Distributed Hadoop (CDH)*. CDH provides unified batch processing, interactive SQL, interactive search, and role-based access controls. In addition, it offers enterprise-grade continuous availability. Specifically, Composer connects to CDH's fault-tolerant storage system called the Hadoop Distributed File System (HDFS).

The Composer HDFS connector uses its own embedded Apache Spark functionality. It supports Apache Spark 2.2 in its implementation.

By default, the HDFS connector is not included with Composer. You or your administrator need to download and enable it before configuring the connector.

After setting up the connector, create data sources that specify the necessary connection information and identify the data you want to use. See Manage Data Sources for more information. After you set up your data sources, create dashboards and visuals from from the data in these data sources. See Create Data Discovery Dashboards.

Feature Support

Connector support for specific features is shown in the following table.

Key: Y - Supported; **N** - Not Supported; N/A - not applicable

Feature	Supported?
Admin-Defined Functions	Y
Box Plots	Y
Custom SQL Queries	Y
Derived Fields (Row-Level Expressions)	Y
Distinct Counts	Y
Fast Distinct Values	N/A
Group By Multiple Fields	Y
Group By Time	Y
Group By UNIX Time	Y
Histogram Floating Point Values	Y
Histograms	Y
Kerberos Authentication	Y
Last Value	Y
Live Mode and Playback	Y
Multivalued Fields	N/A

Feature	Supported?
Nested Fields	N/A
Partitions	N/A
Pushdown Joins for Fusion Data Sources	**N**
Schemas	**Y**
Text Search	N/A
TLS	**N**
User Delegation	**N**
Wildcard Filters	**Y**
Wildcard Filters, Case-Insensitive Mode	**Y**
Wildcard Filters, Case-Sensitive Mode	**Y**

Manage the Hive Connector

The Composer Hive connector lets you access the data available in Hive storage using the Composer client. It can connect to both Hive on Tez and Hive on Tez with LLAP, depending on the JDBC URL you provide (see Connect To Hive below). The Composer Hive connector supports Hive versions 2.1 through 3.1.

Before you can establish a connection from Composer to Hive storage, a connector server needs to be installed and configured. See Manage Connectors And Connector Servers for general instructions and Connect To Hive for details specific to the Hive connector.

After setting up the connector, create data sources that specify the necessary connection information and identify the data you want to use. See Manage Data Sources for more information. After you set up your data sources, create dashboards and visuals from from the data in these data sources. See Create Data Discovery Dashboards.

Feature Support

Connector support for specific features is shown in the following table.

Key: Y - Supported; **N** - Not Supported; N/A - not applicable

Feature	Supported?
Admin-Defined Functions	Y
Box Plots	Y
Custom SQL Queries	Y
Derived Fields (Row-Level Expressions)	Y
Distinct Counts	Y
Fast Distinct Values	N/A
Group By Multiple Fields	Y
Group By Time	Y
Group By UNIX Time	Y
Histogram Floating Point Values	Y
Histograms	Y
Kerberos Authentication	Y
Last Value	Y
Live Mode and Playback	Y
Multivalued Fields	N/A
Nested Fields	N/A

Feature	Supported?
Partitions	Y
Pushdown Joins for Fusion Data Sources	Y
Schemas	Y
Text Search	N/A
TLS	Y
User Delegation	Y
Wildcard Filters	Y
Wildcard Filters, Case-Insensitive Mode	Y
Wildcard Filters, Case-Sensitive Mode	Y

Connect to Hive

To establish a connection to Hive, you must specify a JDBC URL on the Connection page of your Composer data source definition for the Hive connection.

- Specify the JDBC URL for Hive.
- If authentication has been set up, provide the user name and password.
- If required, specify the Hive/YARN queue name in the Queue Name box.
- Specify the server timezone. If the timezone of your Hive server is in UTC, leave the Server Timezone box blank. Otherwise, specify the timezone abbreviation in all caps for correct handling the time data (for example, EST, EDT, or CST).
- Select **Validate** to test the connection.

To connect to Hive LLAP, the JDBC URL you must specify is different. If you use Hortonworks Data Platform (HDP), you can copy the URL from Ambari. See https://docs.cloudera.com/HDPDocuments/HDP3/HDP-3.1.4/performance-tuning/content/hive_connect_clients_to_llap.html.

See also Connect To Hive Sources On A Kerberized HDP Cluster.

Troubleshooting

If you run into a warning message that is displayed when you try to open a dashboard based on a Hive data source, see Resolve The Hive Timeout Warning Message.

Connect to Hive Sources on A Kerberized HDP Cluster

A secure Hortonworks Data Platform (HDP) cluster uses Kerberos authentication to validate and confirm access requests. You can set up Composer to connect to the secure HDP cluster using the following instructions.

Prepare the Hive Cluster

- To enable Kerberos for HDP distribution using a Hive source, refer to Hortonwork's documentation Enabling Kerberos Authentication Using Ambari.
- Kerberos authentication requires precise time correspondence on all instances to work properly. You need to enable the Network Time Protocol service in your network. For more information, see Using the Network Time Protocol to Synchronize Time .

Configure Composer Microservices

Obtain Kerberos Credentials

Each microservice must have its own unique identifier called a principal. Perform the following steps:

1. Install the Kerberos client on the CentOS or Ubuntu machine where the Composer server resides.
2. Generate the Kerberos principal and corresponding keytab for the Composer microservice. Before you proceed, make sure that:

i. The Composer microservice is running on a node with proper Kerberos configuration: `/etc/krb5.conf` or similar location for your Linux distribution.

ii. The Kerberos realm on your environment is the same as the realm specified in the `kdc.conf` file from the Hive server.

3. Check the Kerberos configuration (that is, `krb5.conf`) and validity of the principal and keytab pair using MIT Kerberos client:

```
kinit -V -k -t <composer_principal>.keytab <composer_principal@KERBEROS.REALM>
```

4. Make the keytab accessible for Composer's Hive connector:

```
sudo mkdir /etc/zoomdata
sudo mv <composer_principal>.keytab /etc/zoomdata
```

```
sudo chown zoomdata:zoomdata /etc/zoomdata/<composer_principal>.keytab
sudo chmod 600 /etc/zoomdata/<composer_principal>.keytab
```

Configure a Hive Connector

1. Create or update the file named `/etc/zoomdata/edc-hive.properties`. If this file already exists, verify that the information below exists in the file:

```
kerberos.krb5.conf.location=/etc/krb5.conf
kerberos.service.account.authentication=true
kerberos.service.account.principal=<composer_principal@KERBEROS.REALM>
kerberos.service.account.keytab.location=/etc/zoomdata/<composer_principal>.keytab
```

2. Restart the Hive connector:

```
sudo systemctl restart zoomdata-edc-hive
```

Connect to the Kerberized Hive Source

You are now ready to create the Hive source:

1. Open a new browser window and log into Composer.
2. Select **Sources**.
3. Select **Hive**.
4. Specify the name of your source and add a description (if desired). Then select **Next**.
5. On the **Connection** page, define the connection source. You can use an existing connection, if available, or create a new one. To create a new connection, select the **Input New Credentials** option button and specify the connection name and JDBC URL. Make sure that you enter the JDBC URL in the correct format:

```
jdbc:hive2://<hive_host>:10000/;principal=<hive_principal@KERBEROS.REALM>
```

   Replace the placeholders as follows:

i. `<hive_host>`: Specify the IP address or host name of the Hive node to which you are connecting.

ii. `<hive_principal@KERBEROS.REALM>`: Enter the principal of the Hive node you are connecting to. To get the list of all Hive principals, navigate to Ambari > Admin > Kerberos > Advanced > Hive.

> **Note:** The *principal* spec contained in the JDBC URL refers to the principal of the Hive node. `<hive_principal@KERBEROS.REALM >` principal has nothing to do with the `<zoomdata_principal@KERBEROS.REALM>` principal specified for the Composer connector.

6. Select **Validate** and, after your connection is valid, select **Next**.

You can continue configuring the data source as described in Manage Data Sources.

After you have completed the configuration, Composer will begin accessing Hive using `zoomdata_principal@KERBEROS.REALM` authenticated by its keytab in `/etc/zoomdata/<composer_principal>.keytab`.

Manage the Jira Connector

The Composer Jira connector lets you access the data available in Jira. Obtain the connector server following this process: Obtain Additional Connector Servers

After setting up the connector, create data sources that specify the necessary connection information and identify the data you want to use. See Manage Data Sources for more information. After you set up your data sources, create dashboards and visuals from from the data in these data sources. See Create Data Discovery Dashboards.

- Composer Feature Support
- Connect To Jira
- Optimize Performance
- Custom SQL Optimization
- Custom Fields
- Retrieving And Calculating Story Points

Composer Feature Support

Connector support for specific features is shown in the following table.

Key: Y - Supported; **N** - Not Supported; N/A - not applicable

Feature	Supported?
Admin-Defined Functions	**N**
Box Plots	**Y**
Custom SQL Queries	**Y**
Derived Fields (Row-Level Expressions)	**Y**
Distinct Counts	**Y**
Fast Distinct Values	**N**
Group By Multiple Fields	**Y**
Group By Time	**Y**
Group By UNIX Time	**Y**

Feature	Supported?
Histogram Floating Point Values	Y
Histograms	Y
Kerberos Authentication	N
Last Value	N
Live Mode and Playback	Y
Multivalued Fields	N
Nested Fields	N
Partitions	N
Pushdown Joins for Fusion Data Sources	N
Schemas	Y
Text Search	N
TLS	N
User Delegation	N
Wildcard Filters	Y
Wildcard Filters, Case-Insensitive Mode	Y
Wildcard Filters, Case-Sensitive Mode	Y

Connect to Jira

To connect Composer to your Jira instance, provided the JDBC url, a Jira user name, and password or API token.

Input Field	Description
JDBC Url	jdbc:jira://;Host=*host_name_or_your_server*;Auth_Type=Basic Authentication
User Name	Jira user name
Password	Jira user password or API token generated for the Atlassian account.

Configure your Jira rate limit using Atlassian's guidelines to minimize rate limit errors and prevent performance issues. Rate limit are configured per user, per project; add a test user account with no rate limit to test your projects. See https://developer.atlassian.com/cloud/jira/platform/rate-limiting/.

Optimize Performance

The Jira connector and Simba JDBC driver use schema tables to map your data to a compatible JDBC format you can use when creating your sources. Learn more here: Schema Tables.

Custom SQL Optimization

Large Jira boards and projects can affect the connector's performance. To minimize this impact, you can create your data source using a custom SQL query and pushdown filters. To optimize the query, you can:

- Include pushdown filters
- Design queries to filter on columns for which folding is supported
- Limit your query using a `WHERE` clause and `TOP` for columns that don't support pushdown filters
- Narrow your data retrieval by filtering your data by epic, project, issue creation date, or completion date
- Speed your initial load time by defining a small time range on the Global Settings tab

Example:

```
SELECT S.Sprint_name, S.Sprint_startDate, S.Sprint_endDate, S.Sprint_state, I.Issues_fields_project_name, I.Issues_fields_
project_key, I.Issues_fields_status_name, I.Issues_key
FROM Agile_Board_Sprint S
JOIN Extra.Agile_Board_Issue I ON S.Sprint_id = I.Issues_fields_sprint_id
WHERE S.Sprint_startDate > 'YYY-MM-DD' AND I.Issues_fields_project_key = 'MY_PROJECT_KEY'
```

Raw Data Cache

You can reduce the number of queries Composer makes to the source and speed up data query execution by enabling raw data caching. When enabled, an **Entity Data Cache** toggle is added to the Source Creation work area. Enable and define a caching schedule. Contact Technical Support for assistance enabling this feature.

Custom Fields

See Api_Field for custom fields mappings. The field `custom_name` is type `SQL_VARCHAR(1024)` and is often represented in JSON format. Use the `SUBSTRING()` function to extract fields from the JSON.

Example:

```
SELECT Fields_Issue_Type_Name, SUBSTRING(customfield_13218, 89, 5) AS Team, SUBSTRING(customfield_11200, 105, 1) AS
Severity, Fields_Status_Name AS Status, SUBSTRING(customfield_12618, 105, 8) AS Defect_Origin, Fields_Created, customfield_
10100 AS Story_points
FROM Extra.Api_Issue E
WHERE Fields_Project_Key = 'MY_PROJECT_KEY' AND Fields_Issue_Type_Name = 'Story'
```

Retrieving and Calculating Story Points

Jira stores time estimation and story points. The story points are a custom field mapping, see Api_Field. Use this information to build custom metrics and return commonly used calculations.

- Committed story points: `SUM(story_points)`
- Completed story points: `SUM(story_points) WHERE fields_status_name = 'Done'`
- Percent completed: `(SUM(story_points) WHERE fields_status_name = 'Done') / (SUM(story_points))`

Example:

```
SELECT Issues_fields_project_name, Issues_fields_sprint_name, Issues_fields_sprint_startDate, Issues_fields_sprint_endDate,
Issues_fields_sprint_self, Issues_fields_sprint_state, Issues_key, Issues_fields_status_name, customfield_10100 AS story_
points
FROM Extra.Agile_Board_Issue
WHERE Issues_fields_project_key = 'MY_PROJECT_KEY' AND Issues_fields_sprint_name !=NULL
```

Manage the MemSQL Connector

The Composer MemSQL connector lets you access the data available in MemSQL databases using the Composer client. The Composer MemSQL connector supports MemSQL versions 7.1 - 7.6.

Before you can establish a connection from Composer to MemSQL storage, a connector server needs to be installed and configured. See Manage Connectors And Connector Servers for general instructions and Connect To MemSQL for details specific to the MemSQL connector.

After setting up the connector, create data sources that specify the necessary connection information and identify the data you want to use. See Manage Data Sources for more information. After you set up your data sources, create dashboards and visuals from from the data in these data sources. See Create Data Discovery Dashboards.

Feature Support

Connector support for specific features is shown in the following table.

Key: Y - Supported; **N** - Not Supported; N/A - not applicable

Feature	Supported?
Admin-Defined Functions	**Y**
Box Plots	**Y**
Custom SQL Queries	**Y**
Derived Fields (Row-Level Expressions)	**Y**
Distinct Counts	**Y**
Fast Distinct Values	N/A
Group By Multiple Fields	**Y**
Group By Time	**Y**
Group By UNIX Time	**Y**
Histogram Floating Point Values	**Y**
Histograms	**Y**
Kerberos Authentication	**N**
Last Value	**Y**
Live Mode and Playback	**Y**
Multivalued Fields	N/A
Nested Fields	N/A

Feature	Supported?
Partitions	N/A
Pushdown Joins for Fusion Data Sources	**Y**
Schemas	**Y**
Text Search	N/A
TLS	**Y**
User Delegation	**N**
Wildcard Filters	**Y**
Wildcard Filters, Case-Insensitive Mode	**Y**
Wildcard Filters, Case-Sensitive Mode	**N**

Connect to MemSQL

The MemSQL connector requires a JDBC driver to be configured before you can connect to your data source.

1. Download and install the latest JDBC driver from https://dev.mysql.com/downloads/connector/j/. Instructions are provided on that website.
2. After installing the driver, edit the MemSQL properties file and change the driver path and class name properties, as shown below and as described in Add A JDBC Driver:

```
datasource.driver-config.class-name=com.mysql.cj.jdbc.Driver

datasource.driver-config.jar-path=<JDBC_driver_filepath>
```

3. Save the properties file and restart the connector. See Add A JDBC Driver.

Manage the Microsoft SQL Server Connector

Composer's SQL Server connector lets you access the data available in the SQL query engine using the Composer client. The minimum version of MS SQL you can use with Composer is v12.0 (SQL Server 2014).

Before you can establish a connection from Composer to SQL Server storage, a connector server needs to be installed and configured. See Manage Connectors And Connector Servers for general instructions and Connect To Microsoft SQL Server for details specific to the Microsoft SQL Server connector.

After setting up the connector, create data sources that specify the necessary connection information and identify the data you want to use. See Manage Data Sources for more information. After you set up your data sources, create dashboards and visuals from from the data in these data sources. See Create Data Discovery Dashboards.

Feature Support

Connector support for specific features is shown in the following table.

Key: Y - Supported; **N** - Not Supported; N/A - not applicable

Feature	Supported?
Admin-Defined Functions	**Y**
Box Plots	**Y**
Custom SQL Queries	**Y**
Derived Fields (Row-Level Expressions)	**Y**
Distinct Counts	**Y**
Fast Distinct Values	N/A
Group By Multiple Fields	**Y**
Group By Time	**Y**
Group By UNIX Time	**Y**
Histogram Floating Point Values	**Y**
Histograms	**Y**
Kerberos Authentication	**N**
Last Value	**Y**
Live Mode and Playback	**Y**
Multivalued Fields	N/A
Nested Fields	N/A

Feature	Supported?
Partitions	**N**
Pushdown Joins for Fusion Data Sources	**Y**
Schemas	**Y**
Text Search	N/A
TLS	**Y**
User Delegation	**N**
Wildcard Filters	**Y**
Wildcard Filters, Case-Insensitive Mode	**Y**
Wildcard Filters, Case-Sensitive Mode	**N**

Connect to Microsoft SQL Server

When establishing a connection to SQL Server, you need to provide the following.

- Specify the connection name and JDBC URL. The JDBC URL for SQL Server database being connected, must be: `jdbc:sqlserver//MYSQLSERVERHOST:PORT`
- If authentication is required, provide the username and password.

Support for Azure Synapse Data Sources

This connector can also be used to connect to Azure Synapse data sources with the following caveats:

- If you use offset, pagination is implemented using the `ROW_NUMBER()` function.
- When you use a LIKE pattern, use square brackets [ ] as the escape characters.
- When using the IIF function, use the CASE expression.

Manage the MongoDB Connector

The Composer MongoDB connector lets you access the data available in MongoDB storage using the Composer client. The Composer MongoDB connector supports MongoDB versions 3.4 - 4.4.

The MongoDB connector supports MongoDB views.

Before you can establish a connection from Composer to MongoDB storage, a connector server needs to be installed and configured. See Manage Connectors And Connector Servers for general instructions.

After setting up the connector, create data sources that specify the necessary connection information and identify the data you want to use. See Manage Data Sources for more information. After you set up your data sources, create dashboards and visuals from from the data in these data sources. See Create Data Discovery Dashboards.

Feature Support

Connector support for specific features is shown in the following table.

Key: Y - Supported; **N** - Not Supported; N/A - not applicable

Feature	Supported?
Admin-Defined Functions	**N**
Box Plots	**N**
Custom SQL Queries	**N**
Derived Fields (Row-Level Expressions)	**Y**
Distinct Counts	**Y**
Fast Distinct Values	N/A
Group By Multiple Fields	**Y**
Group By Time	**Y**
Group By UNIX Time	**N**
Histogram Floating Point Values	**Y**
Histograms	**N**
Kerberos Authentication	**N**
Last Value	**N**
Live Mode and Playback	**Y**
Multivalued Fields	**Y**

Feature	Supported?
Nested Fields	**Y**
Partitions	N/A
Pushdown Joins for Fusion Data Sources	**N**
Schemas	**Y**
Text Search	N/A
TLS	**Y**
User Delegation	**N**
Wildcard Filters	**Y**
Wildcard Filters, Case-Insensitive Mode	**Y**
Wildcard Filters, Case-Sensitive Mode	**Y**

MongoDB connectors support derived fields created using most row-level functions, with the following exceptions:

- If you are running a version of MongoDB prior to version 4.0, the following text row-level functions are not supported (these functions work for MongoDB version 4.0 and later):
  - TEXT_TO_NUM
  - TEXT_TO_TIME
  - LTRIM
  - RTRIM
- The following restrictions apply to MongoDB 3.4:
  - You cannot use a number field with a year pattern as a date field in a row-level expression.
  - TIME_ADD intervals cannot be specified for YEAR, QUARTER, or MONTH.
  - TRUNCATE_TIME cannot truncate date-time field values to YEAR or QUARTER granularities.

Connect to MongoDB with Configured SSL

Connect to a MongoDB data store with configured SSL

1. Add parameter `ssl=true` to your connection string. For example (replace `<mongodb_host>`, `<port>`, `<database>` with your values):

```
mongodb://<mongodb_host>:<port>/<database>?ssl=true
```

2. If your MongoDB data store is configured with a custom certificate, you should configure a truststore for the MongoDB connector:

   a. Copy a truststore to the machine on which the MongoDB connector is running.

   b. Add the following lines to file `/etc/zoomdata/edc-mongo.jvm`. Copy the file `edc-mongo.jvm` from the `/opt/zoomdata/conf` directory if a copy is not in `/etc/zoomdata/`.

```
-Djavax.net.ssl.trustStore=<path_to_truststore>
-Djavax.net.ssl.trustStorePassword=<truststore_password>
```

   Replace:

   i. `<path_to_truststore>` with an absolute path to your truststore

   ii. `<truststore_password>` with a password for your truststore

3. Restart the MongoDB connector microservice. See Restart ComposerSymphony Microservices.

Manage the MySQL Connector

The Composer MySQL connector lets you access the data available in MySQL databases using the Composer client. The Composer MySQL connector supports MySQL versions 5.6 - 8.0.

Before you can establish a connection from Composer to MySQL storage, a connector server needs to be installed and configured. See Manage Connectors And Connector Servers for general instructions and Connect To MySQL for details specific to the MySQL connector.

After setting up the connector, create data sources that specify the necessary connection information and identify the data you want to use. See Manage Data Sources for more information. After you set up your data sources, create dashboards and visuals from from the data in these data sources. See Create Data Discovery Dashboards.

Feature Support

Connector support for specific features is shown in the following table.

Key: Y - Supported; **N** - Not Supported; N/A - not applicable

Feature	Supported?
Admin-Defined Functions	**Y**
Box Plots	**N**
Custom SQL Queries	**Y**
Derived Fields (Row-Level Expressions)	**Y**
Distinct Counts	**Y**
Fast Distinct Values	N/A
Group By Multiple Fields	**Y**
Group By Time	**Y**
Group By UNIX Time	**Y**
Histogram Floating Point Values	**Y**
Histograms	**Y**
Kerberos Authentication	**N**
Last Value	**Y**
Live Mode and Playback	**Y**
Multivalued Fields	N/A
Nested Fields	N/A

Feature	Supported?
Partitions	**N**
Pushdown Joins for Fusion Data Sources	**Y**
Schemas	**Y**
Text Search	N/A
TLS	**Y**
User Delegation	**N**
Wildcard Filters	**Y**
Wildcard Filters, Case-Insensitive Mode	**Y**
Wildcard Filters, Case-Sensitive Mode	**N**

Connect to MySQL

The MySQL connector requires a JDBC driver to be configured before you can connect to your data source.

1. Download and install the version latest JDBC driver from https://dev.mysql.com/downloads/connector/j/. Instructions are provided on that website.

2. After installing the driver, edit the MySQL properties file and change the driver path and class name properties, as shown below and as described in Add A JDBC Driver:

```
datasource.driver-config.class-name=com.mysql.cj.jdbc.Driver
datasource.driver-config.jar-path=<JDBC_driver_filepath>
```

3. Save the properties file and restart the connector. See Add A JDBC Driver.

Manage the Oracle Connector

The Composer Oracle connector lets you access the data available in Oracle databases using the Composer client. The Composer Oracle connector supports Oracle versions 11.2 - 21c.

Before you can establish a connection from Composer to Oracle storage, a connector server needs to be installed and configured. See Manage Connectors And Connector Servers for general instructions and Connect To Oracle for details specific to the Oracle connector.

After setting up the connector, create data sources that specify the necessary connection information and identify the data you want to use. See Manage Data Sources for more information. After you set up your data sources, create dashboards and visuals from from the data in these data sources. See Create Data Discovery Dashboards.

Feature Support

Connector support for specific features is shown in the following table.

Key: Y - Supported; **N** - Not Supported; N/A - not applicable

Feature	Supported?	Notes
Admin-Defined Functions	**Y**	
Box Plots	**Y**	
Custom SQL Queries	**Y**	If you need to access a BigQuery partition, explicitly include an alias for the built in partition column in your select clause, such as `select *, _PARTITIONTIME as pt from projectId.datasetId.tableId`.
Derived Fields (Row-Level Expressions)	**Y**	
Distinct Counts	**Y**	
Fast Distinct Values	N/A	
Group By Multiple Fields	**Y**	
Group By Time	**Y**	
Group By UNIX Time	**Y**	
Histogram Floating Point Values	**Y**	
Histograms	**Y**	
Kerberos Authentication	**N**	
Last Value	**Y**	
Live Mode and Playback	**Y**	

Feature	Supported?	Notes
Multivalued Fields	N/A	
Nested Fields	N/A	
Partitions	**N**	
Pushdown Joins for Fusion Data Sources	**Y**	
Schemas	**Y**	
Text Search	N/A	
TLS	**Y**	
User Delegation	**Y**	The Composer Oracle connector supports user delegation only via user credential pass-through.
Wildcard Filters	**Y**	
Wildcard Filters, Case-Insensitive Mode	**Y**	
Wildcard Filters, Case-Sensitive Mode	**Y**	

Connect to Oracle

The Oracle connector requires a JDBC driver to be configured before you can connect to your data source. You can download the driver from the vendor's site. If you are upgrading, keep in mind you need to configure the JDBC driver- see Upgrade ComposerSymphony for instructions. For more information and steps, see Add A JDBC Driver.

When setting up a connection to Oracle, provide the following.

- Specify the connection name and JDBC URL. The JDBC URL for Oracle database being connected must be: `jdbc:oracle:thin@//ORACLEHOST:PORT/DATABASE_NAME` or `jdbc:oracle:thin:@ORACLEHOST:PORT:SID`

  To connect to Oracle with TLS enabled, see Connect To Oracle With TLS Enabled.

- Provide the user name and password for Oracle database.

- You can use an Impersonation feature to work with Oracle data source on behalf of a proxy user. Before you begin, you must configure proxy users with the corresponding privileges in the Oracle database. To use this, select the **Impersonation Enabled** checkbox and specify the **Impersonation Username** and **Impersonation Password**. See Configure Settings To Use A Proxy User.

- If you need to use Composer's Oracle connector to access a table that uses the XML data type, complete the additional setup steps described in Enable Access To Oracle Tables That Use The XML Data Type.

Note: If there are any changes in the Oracle database, you must clear the Composer cache. See How Composer Caches Data.

Connect to Oracle with TLS Enabled

Before you attempt to connect to Oracle with TLS enabled, make sure you have first installed Java Cryptography Extension (JCE). See *https://www.oracle.com/java/technologies/javase-jce8-downloads.html*.

Connect to Oracle with TLS enabled

1. Create a JDBC URL with TLS parameters. To specify TLS-related parameters, use the following template for a JDBC URL:

```
jdbc:oracle:thin:@(DESCRIPTION=(ADDRESS=(PROTOCOL=tcps)(HOST=<oracle_host>)
(PORT=<oracle_tls_port))CONNECT_DATA=(SID=<service_id>)))
```

   where:

i. `<oracle_host>` is the host of the Oracle database.

ii. `<oracle_tls_port>` is the port of the Oracle database with TLS enabled

iii. `<service_id>` is the Oracle service ID or database to which you want to connect

   Make sure your JDBC URL uses the correct protocol. For a TLS connection, you should use `tcps`.

2. If your Oracle database is configured with a custom certificate, you should configure a truststore for the Oracle connector, as described in the following steps:

   a. Copy a truststore to the machine on which Composer's Oracle connector is running.

   b. Add the following lines to file `edc-oracle.jvm`.

```
-Djavax.net.ssl.trustStore=<path_to_truststore>
-Djavax.net.ssl.trustStorePassword=<truststore_password>
```

i. Linux: Copy the file `edc-oracle.jvm` from the `/opt/zoomdata/conf` directory if a copy is not in `/etc/zoomdata/`.

ii. Windows: Copy the file `edc-oracle.jvm` from the `<install-path>/conf` directory if a copy is not in `<install-path>/conf-modify`.

where:

i. `<path_to_truststore>` is the absolute path to your truststore

ii. `<truststore_password>` is the password for your truststore

3. Restart the Oracle connector microservice, `zoomdata-edc-oracle`. See Restart ComposerSymphony Microservices.

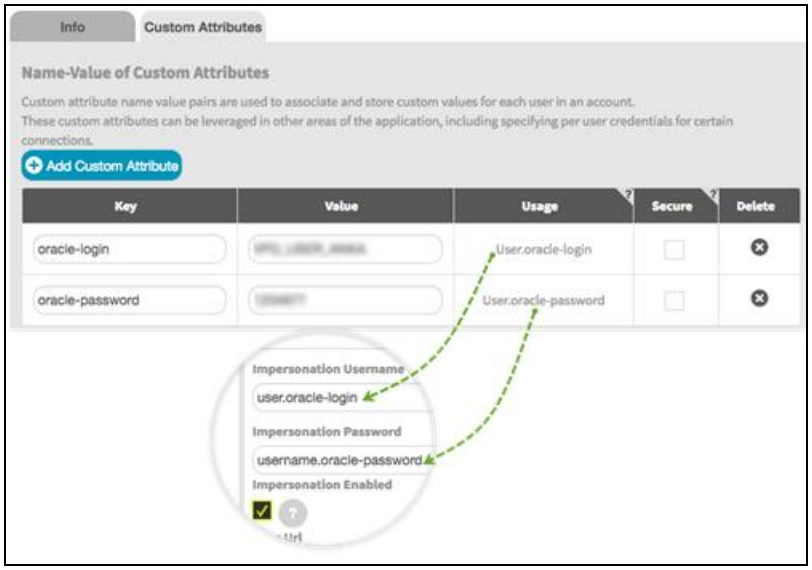
Configure Settings to Use a Proxy User

To enable a Composer user work as a proxy user, specify the user attributes of corresponding oracle user (that will be used as proxy user) in the account details of a Composer user.

You must specify the user attributes for each Composer user that will access the Oracle data source as a proxy user.

Perform the following steps:

1. Select Settings and then select **Users and Groups**.
2. Select a user from the list and select the **Custom Attributes** tab.
3. Select **Add Custom Attribute**. Specify credentials for a user as follows:
   - Key - specify the login attribute for proxy user. Corresponding reference name is listed in the **Usage** column. You have to specify the value from the **Usage** column in the **Impersonation Username** field while creating a connection.
   - Value - specify the actual name of the oracle user, that you want to use as proxy user.
   - Select the checkbox in the **Secure** column to encrypt the value of the key.
4. If the proxy user requires a password, select **Add Custom Attribute** and specify the key and value for the password. You have to specify the reference name from the **Usage** column in the **Impersonation Password** field while creating the connection.



Enable Access to Oracle Tables That Use the XML Data Type

Before you enable access to Oracle tables that use the XML data type, be sure you have set up the Oracle JDBC driver. See Add A JDBC Driver.

Enable access to Oracle tables that use the XML data type

1. Download the `xdb6.jar` and `xmlparserv2.jar` files from Oracle to the corresponding Composer instance. You can download obtain `xdb6.jar` by downloading it from https://www.oracle.com/database/technologies/jdbc-ucp-122-downloads.html. You can obtain `xmlparserve2.jar` by extracting it from the `lib` directory in the Oracle XML Developers Kit, which can be downloaded from https://www.oracle.com/downloads/.

   Place these files in the following folder:

i. Linux: `/opt/zoomdata/lib/edc-oracle/`

ii. Windows: `<install-path>/lib/edc-oracle/`

> **Note:** Note that this is the same folder where you downloaded the Oracle JDBC driver (see Add A JDBC Driver).

2. Add the following lines to the connector JVM file:

```
-Djavax.xml.parsers.DocumentBuilderFactory=org.apache.xerces.jaxp.DocumentBuilderFactoryImpl
-Djavax.xml.transform.TransformerFactory=com.sun.org.apache.xalan.internal.xsltc.trax.TransformerFactoryImpl
```

i. Linux: `/etc/zoomdata/edc-oracle.jvm`

ii. Windows: `<install-path>/conf-modify/`

3. Restart the Oracle connector microservice, `zoomdata-edc-oracle`. See Restart ComposerSymphony Microservices.

Manage the PostgreSQL Connector

The Composer PostgreSQL connector lets you access the data available in the PostgreSQL query engine using the Composer client. The Composer PostgreSQL connector supports PostgreSQL versions 9.6 through 14.0.

Before you can establish a connection from Composer to PostgreSQL storage, a connector server needs to be installed and configured. See Manage Connectors And Connector Servers for general instructions and Connect To PostgreSQL for details specific to the PostgreSQL connector.

After setting up the connector, create data sources that specify the necessary connection information and identify the data you want to use. See Manage Data Sources for more information. After you set up your data sources, create dashboards and visuals from from the data in these data sources. See Create Data Discovery Dashboards.

Feature Support

Connector support for specific features is shown in the following table.

Key: Y - Supported; **N** - Not Supported; N/A - not applicable

Feature	Supported?
Admin-Defined Functions	**Y**
Box Plots	**Y**
Custom SQL Queries	**Y**
Derived Fields (Row-Level Expressions)	**Y**
Distinct Counts	**Y**
Fast Distinct Values	N/A
Group By Multiple Fields	**Y**
Group By Time	**Y**
Group By UNIX Time	**Y**
Histogram Floating Point Values	**Y**
Histograms	**Y**
Kerberos Authentication	**N**
Last Value	**Y**
Live Mode and Playback	**Y**
Multivalued Fields	N/A
Nested Fields	N/A

Feature	Supported?
Partitions	**N**
Pushdown Joins for Fusion Data Sources	**Y**
Schemas	**Y**
Text Search	N/A
TLS	**Y**
User Delegation	**N**
Wildcard Filters	**Y**
Wildcard Filters, Case-Insensitive Mode	**Y**
Wildcard Filters, Case-Sensitive Mode	**Y**

Connect to PostgreSQL

When establishing a connection to PostgreSQL, you must provide the following.

- Specify the connection name and JDBC URL. The JDBC URL format for PostgreSQL database being connected, must be:
  `jdbc:postgresql://POSTGRESQLHOST:PORT/DATABASE_NAME`

- If authentication has been set up, provide the User Name and Password.

Manage the Python Connector

Connect your data in Composer using a Python script. Access information exposed by APIs, or other data generated by calculation or prediction models.

Important: The Python connector is available as a Docker image; you must install Docker on your server running the Python connector. See Install The Python Connector.

Python code is executed using JEP. To circumvent some Global Interpreter Lock issues in Python, some queries use processed based parallelism. Based on the request type, the connector functions in one of two ways:d

- Interpret the script in the same process for validation and describe requests.
- Interpret the script in the same process and invoke the function in a sub process for fetch data requests.

For more information, see Use The Python Connector.

Connector Feature Support

Connector support for specific features is shown in the following table.

Feature	Supported?
Admin-Defined Functions	**N**
Box Plots	**Y**
Custom SQL Queries	**Y**
Derived Fields (Row-Level Expressions)	**Y**
Distinct Counts	**Y**
Fast Distinct Values	**N**
Group By Multiple Fields	**Y**
Group By Time	**Y**
Group By UNIX Time	**Y**
Histogram Floating Point Values	**Y**
Histograms	**Y**
Kerberos Authentication	**N**
Last Value	**Y**

Feature	Supported?
Live Mode and Playback	N
Multivalued Fields	N
Nested Fields	N
Partitions	N
Pushdown Joins for Fusion Data Sources	N
Schemas	N
Text Search	N
TLS	N
User Delegation	N
Wildcard Filters	Y
Wildcard Filters, Case-Insensitive Mode	Y
Wildcard Filters, Case-Sensitive Mode	Y

Install the Python Connector

The python connector is available as a Docker image only. Docker must be installed on the Composer server running the Python connector.

Download and Install the Docker Image in a Linux Environment

To download the Docker image:

```
docker pull insightsoftware/zoomdata-edc-python:<RELEASE_TAG>
```

You can find your required `<RELEASE_TAG>` in this repo: https://hub.docker.com/r/insightsoftware/zoomdata-edc-python/tags.

Use the **docker run** command to run the Python connector.

Important: You must provide a Consul host and ensure that the connector is registered in Consul with a host that is accessible to other Composer services.

In the example below, the **docker run** command runs the setup on the same machine that has both the connector installed and other Composer services installed using the bootstrap script:

```
docker run --env DISCOVERY_REGISTRY_HOST=localhost --network=host --name=zoomdata-edc-python --detach
insightsoftware/zoomdata-edc-python:<RELEASE_TAG>
```

Note: Adjust the command to work with your specific network configuration.

- Consul host is passed to Python Connector using the `DISCOVERY_REGISTRY_HOST` environment variable.
- Because the container network in this case is connected to the host machine's network (due to `--network=host`), Consul is accessed on `localhost`.
- Use `--name param` to assign a meaningful name to the container. `--detach` runs the container in the background and prints the container ID.
- See docker documentation for more information on `docker run` arguments.

Important: The host networking driver only works in Linux environments. See Run The Python Connector On Non-Linux Servers for generic networking requirements.

Run the Python Connector on Non-Linux Servers

Define a networking configuration that:

- Runs the connector inside a container that can access the Consul host and register with Consul.
- Allows other Composer services to access the connector running inside the container.

For example, in a case when the Consul is running on a different host, `--env DISCOVERY_REGISTRY_HOST=localhost --network=host` is not suitable. You'll need to make sure that Consul listens on external port 8500, and takes its hostname. Additionally, you may need to use the `--expose 8153` argument to expose the port that the Python Connector listens on. Also, you can use the `--hostname` argument to control the value of the service address that will be registered in Consul for Python Connector. Putting it together:

```
docker run --expose 8153 --env DISCOVERY_REGISTRY_HOST=<consul-host> --hostname <connector_host> --name=zoomdata-edc-python
--detach insightsoftware/zoomdata-edc-python:<RELEASE_TAG>
```

Verify the Installation

To verify correct installation of the Python connector, run the following command shortly after Composer starts:

```
curl localhost:8500/v1/health/service/edc-python
```

All checks must return the status `passing`.

After that, log in to Composer create a new connection. Python should be available in the Connection Type list.

View Python Logs

To view the Python connector's logs use the `docker logs` command:

```
docker logs --follow zoomdata-edc-python
```

Python Packages

The Docker image is shipped with the `python3-pip` package installed. This includes preinstalled pip packages of `numpy`, `pandas`, `requests`, and `jep`. To install additional pip packages, use the `ADDITIONAL_PYTHON_LIBS` environment variable when running the container.

```
docker run --expose 8153 --env DISCOVERY_REGISTRY_HOST=<consul-host> --env ADDITIONAL_PYTHON_LIBS="boto3 python-dateutil" --detach insightsoftware/zoomdata-edc-python:<RELEASE_TAG>
```

This command runs the Python connector and installs both the `boto3` and `python-dateutil` packages inside the container.

For more information on using the Python connector and how it works, see Use The Python Connector.

Use the Python Connector

You can use the Python connector using arbitrary Python scripts as connection parameter.

Important: The Python connector functions in raw data mode only and does not support push down of aggregations. Minimize the amount of data in your request using filter operations. This will increase performance speed and improve loading time for dashboards and visuals.

Python Script Conventions

Data sources are resolved from Python script using the following conventions:

- Each function definition is a separate data source
- Private functions (that start with an underscore _) are not resolved as a data source

Conventions for return values of functions include:

- Pandas dataframes.
- Dictionaries: The key is a string (column name) and values a list: `return {"column1": [1, 2, 3], "column2": ["one", "two", "three"]}`.
- List of dictionaries: return `[{"column1": 1, "column2": "one"}, {"column1": 2, "column2": "two"}]`.
- List of lists: `return [[1, 2], [3, 4]]`. Each enclosed list resolves as a row.
- List: `return [1, 2, 3, 4]`. Resolves as a single column with index 1.
- Single value of any of supported types: `return 1` OR `return decimal.Decimal("3.14")`

Regardless of return type, you must use uniform columns of the same size, containing the same value type, to prevent unexpected behavior.

Conversion Values

The connector applies the following rules when reading values:

Python Type	Connector Field Type
str	STRING
int	INTEGER
float	DOUBLE
decimal.Decimal	DOUBLE
datetime.date	DATE
datetime.datetime	DATE
arbitray object	STRING

Python Script Writing Tips

Avoid using top level statements

Top level statement are executed in a single thread for all users. You can add function calls in a top level statement to validate a connection, but don't call functions when you save your script. See How The Connector Works.

Avoid overriding internal names

Python is used within your environment to invoke data source functions and convert data. Since this code is executed in the same namespace as your scripts, if you try to override the names listed below, you may receive unexpected results. Avoid using the following names in your code:

- `__convert`
- `__convert_list_of_dicts_to_dict_of_lists`
- `f`
- `__fork`
- `__emulate`
- `all_functions`

To use this connector, we import these modules. Attempting to use these names for variables and functions may return unexpected results.

- `pandas`
- `numbers`
- `datetime`
- `multiprocessing`
- `queue`
- `inspect`
- `types`

Your python script has limited access to the file system; the container is run as a non-root user. Edit access is still available on the folders below:

Folders with write access include:

- `/opt/zoomdata/logs`
- `/opt/zoomdata/temp`
- `/opt/zoomdata/lib`
- `/opt/zoomdata/wrappers`

How the Connector Works

Python code is executed using JEP to interpret Python code in the same process where the Java app is running. To circumvent some Global Interpreter Lock issues in Python, some queries use processed based parallelism. Based on the request type, the connector functions in one of two ways:

- Interpret the script in the same process for validation and describe requests.
- Interpret the script in the same process and invoke the function in a sub process for fetch data requests.

New sub processes are created by forking the Java process. Each request starts a new Python interpreter. Scripts are always interpreted first in one parent process. Limit top level statements to imports and function definitions for optimal performance.

Function invocation happens in a separate process, so global variables aren't available. For example:

```
x = 40

def side_effect():
       x = x + 1
       return {"result": [x]}
```

While the script is valid, using it as a connection parameter and attempting to set `side_effect` as an entity for the source will return an error such as:

```
UnboundLocalError: local variable 'x' referenced before assignment
```

Logging

Outputs of your Python scripts are not preserved. Statements such as `print("Message")` to write data to `stdout` or `stderr` will not be retained.

Python Script Conversion

Python script types conversion, in versions 23.3+ of Composer, converts all values returned from public functions to Pandas Dataframe. To resolve the type, the connector relies on DataFrame.dtypes.kind.

The following table includes the conversion rules used:

Numpy kind (character code)	Numpy kind (type name)	Field Type
b	boolean	STRING
i	signed integer	INTEGER
u	unsigned integer	INTEGER
f	floating-point	DOUBLE
c	complex floating-point	STRING
m	timedelta	STRING
M	datetime	DATE
O	object	STRING
S	(byte-)string	STRING
U	Unicode	STRING

Manage the Trino Connector

The Composer Trino connector connects to a Trino server, and verified with versions 351-390. Trino is the new brand name for Presto.

You might use the Composer Trino connector to connect to a Trino server on which the Trino connector is installed. After the connection is made, you would then be able to visualize and explore data in the PostgreSQL or AWS S3 database using the Composer client.

The Trino connector has been validated and tested with PostgreSQL or AWS S3 databases only. Composer cannot guarantee bug-free operation of the Trino connector with other databases, although you can try to use Trino to connect to them.

Before you can establish a connection from Composer to the Trino server, the Trino connector server must be installed, configured and enabled. See Manage Connectors And Connector Servers for general instructions and Connect To Trino for details specific to the Trino connector.

After setting up the connector, create data sources that specify the necessary connection information and identify the data you want to use. See Manage Data Sources for more information. After you set up your data sources, create dashboards and visuals from from the data in these data sources. See Create Data Discovery Dashboards.

Feature Support

Connector support for specific features is shown in the following table.

Key: Y - Supported; **N** - Not Supported; N/A - not applicable

Feature	Supported?
Admin-Defined Functions	**Y**
Box Plots	**Y**
Custom SQL Queries	**N**
Derived Fields (Row-Level Expressions)	**N**
Distinct Counts	**Y**
Fast Distinct Values	N/A
Group By Multiple Fields	**Y**
Group By Time	**Y**
Group By UNIX Time	**Y**
Histogram Floating Point Values	**Y**
Histograms	**Y**
Kerberos Authentication	**N**
Last Value	**Y**

Feature	Supported?
Live Mode and Playback	**Y**
Multivalued Fields	N/A
Nested Fields	N/A
Partitions	**N**
Pushdown Joins for Fusion Data Sources	**N**
Schemas	**Y**
Text Search	N/A
TLS	**Y**
User Delegation	**N**
Wildcard Filters	**Y**
Wildcard Filters, Case-Insensitive Mode	**Y**
Wildcard Filters, Case-Sensitive Mode	**Y**

Connect to Trino

Install and configure the Trino connector

1. Install a supported Trino version Refer to Trino's documentation Deploying Trino for more details.
2. Configure the Trino Server. Refer to Trino's documentation Configuring Trino for more details.
3. Make sure that the following files are available in the appropriate Trino server `/etc/` folder:
    - `config.properties`
    - `node.properties`
    - `jvm.config`
    - `log.properties`
4. Configure Trino to connect to your data source of choice, such as PostgreSQL or AWS S3.

Manage the Salesforce Connector

The Composer Salesforce connector lets you access the data available in Salesforce. Obtain the connector server following this process: Obtain Additional Connector Servers

After setting up the connector, create data sources that specify the necessary connection information and identify the data you want to use. See Manage Data Sources for more information. After you set up your data sources, create dashboards and visuals from from the data in these data sources. See Create Data Discovery Dashboards.

- Composer Feature Support
- Connect To Salesforce
- Optimize Performance
- Manage The Salesforce Connector
- Manage The Salesforce Connector
- Manage The Salesforce Connector

Composer Feature Support

Connector support for specific features is shown in the following table.

Key: Y - Supported; **N** - Not Supported; N/A - not applicable

Feature	Supported?
Admin-Defined Functions	**N**
Box Plots	**Y**
Custom SQL Queries	**Y**
Derived Fields (Row-Level Expressions)	**Y**
Distinct Counts	**Y**
Fast Distinct Values	**N**
Group By Multiple Fields	**Y**
Group By Time	**Y**

Feature	Supported?
Group By UNIX Time	Y
Histogram Floating Point Values	Y
Histograms	Y
Kerberos Authentication	N
Last Value	N
Live Mode and Playback	Y
Multivalued Fields	N
Nested Fields	N
Partitions	N
Pushdown Joins for Fusion Data Sources	N
Schemas	Y
Text Search	N
TLS	N
User Delegation	N
Wildcard Filters	Y
Wildcard Filters, Case-Insensitive Mode	Y
Wildcard Filters, Case-Sensitive Mode	Y

Connect to Salesforce

To connect Composer to your Salesforce instance, provided the JDBC url, a Salesforce user name, and password or API token.

Input Field	Description
JDBC Url	jdbc:salesforce://localhost;Endpoint=https://your_salesforce_endpoint/services/Soap/u/48.0
User Name	Your Salesforce user name
Password	Your Salesforce user password

Optimize Performance

The Salesforce connector is a raw data connector. Data retrieved from the data source is processed by the QueryEngine service. When you retrieve and manipulate large chunks of data, consider the following practices to improve performance speed:

- Create your source using custom SQL and fusion source interfaces
- Limit your query using a `WHERE` clause and `TOP`
- Narrow your data retrieval by filtering your data
- Speed your initial load time by defining a small time range on the Global Settings tab

Raw Data Cache

You can reduce the number of queries Composer makes to the source and speed up data query execution by enabling raw data caching. When enabled, an **Entity Data Cache** toggle is added to the Source Creation work area. Enable and define a caching schedule. Contact Technical Support for assistance enabling this feature.

Manage the SAP Hana Connector

The Composer SAP Hana connector lets you access the data stored within your in-memory database using the Composer client. The Composer SAP Hana connector supports SAP Hana version 2.0.

Before you can establish a connection from Composer to SAP Hana storage, a connector server needs to be installed and configured. See Manage Connectors And Connector Servers for general instructions and Connect To SAP Hana for details specific to the SAP Hana connector.

After setting up the connector, create data sources that specify the necessary connection information and identify the data you want to use. See Manage Data Sources for more information. After you set up your data sources, create dashboards and visuals from from the data in these data sources. See Create Data Discovery Dashboards.

Feature Support

Connector support for specific features is shown in the following table.

Key: Y - Supported; **N** - Not Supported; N/A - not applicable

Feature	Supported?
Admin-Defined Functions	**Y**
Box Plots	**Y**
Custom SQL Queries	**Y**
Derived Fields (Row-Level Expressions)	**Y**
Distinct Counts	**Y**
Fast Distinct Values	N/A
Group By Multiple Fields	**Y**
Group By Time	**Y**
Group By UNIX Time	**Y**
Histogram Floating Point Values	**Y**
Histograms	**Y**
Kerberos Authentication	**N**
Last Value	**Y**
Live Mode and Playback	**Y**
Multivalued Fields	N/A
Nested Fields	N/A

Feature	Supported?
Partitions	**N**
Pushdown Joins for Fusion Data Sources	**N**
Schemas	**Y**
Text Search	N/A
TLS	**Y**
User Delegation	**N**
Wildcard Filters	**Y**
Wildcard Filters, Case-Insensitive Mode	**Y**
Wildcard Filters, Case-Sensitive Mode	**Y**

Connect to SAP Hana

The SAP Hana connector requires a JDBC driver to be configured before you can connect to your data source. You can download the driver from the vendor's site. For information and steps, see Add A JDBC Driver.

When setting up a connection to SAP Hana, you need to provide the following:

- The name of the connection
- The JDBC URL. The format for the JDBC URL must be as follows: `jdbc:sap://SAPHANAHOST:PORT/DATABASE_NAME` For example, `jdbc:sap://10.2.2.135:39013.`
- The username and password, if applicable.

Manage the SAP S/4HANA Connector

The Composer SAP S/4HANA connector lets you access the data stored within your in-memory database using the Composer client.

Before you can establish a connection from Composer to SAP S/4HANA storage, a connector server needs to be installed and configured. See Manage Connectors And Connector Servers for general instructions and Connect To SAP S/4HANA for details specific to this connector.

After setting up the connector, create data sources that specify the necessary connection information and identify the data you want to use. See Manage Data Sources for more information. After you set up your data sources, create dashboards and visuals from from the data in these data sources. See Create Data Discovery Dashboards.

Feature Support

Connector support for specific features is shown in the following table.

Key: Y - Supported; **N** - Not Supported; N/A - not applicable

Feature	Supported?
Admin-Defined Functions	**Y**
Box Plots	**Y**
Custom SQL Queries	**Y**
Derived Fields (Row-Level Expressions)	**Y**
Distinct Counts	**Y**
Fast Distinct Values	N/A
Group By Multiple Fields	**Y**
Group By Time	**Y**
Group By UNIX Time	**Y**
Histogram Floating Point Values	**Y**
Histograms	**Y**
Kerberos Authentication	**N**
Last Value	**Y**
Live Mode and Playback	**Y**
Multivalued Fields	N/A
Nested Fields	N/A
Partitions	**N**

Feature	Supported?
Pushdown Joins for Fusion Data Sources	**N**
Schemas	**Y**
Text Search	N/A
TLS	**Y**
User Delegation	**N**
Wildcard Filters	**Y**
Wildcard Filters, Case-Insensitive Mode	**Y**
Wildcard Filters, Case-Sensitive Mode	**Y**

Connect to SAP S/4HANA

The SAP S/4Hana connector requires a JDBC driver to be configured before you can connect to your data source. You can download the driver from the vendor's site. For information and steps, see Add A JDBC Driver.

When setting up a connection to SAP S/4HANA, you need to provide the following:

- The name of the connection
- The JDBC URL. The format for the JDBC URL must be as follows: `jdbc:datadirect:ddhybrid://{hdp_url};hybridDataPipelineDataSource={hdp_datasource};`.
- The username and password, if applicable.

Manage the SAP IQ Connector

The Composer SAP IQ connector allows you to access the data stored within your SAP IQ database using the Composer client. The Composer connector supports SAP IQ version 16.

Before you can establish a connection from Composer to SAP IQ, a connector server needs to be installed, configured and enabled. See Manage Connectors And Connector Servers for general instructions and Connect To SAP IQ for details specific to the SAP IQ connector. If you elect to use Kerberos authentication, see Configure Kerberos Support For The SAP IQ Connector.

After setting up the connector, create data sources that specify the necessary connection information and identify the data you want to use. See Manage Data Sources for more information. After you set up your data sources, create dashboards and visuals from from the data in these data sources. See Create Data Discovery Dashboards.

Feature Support

Connector support for specific features is shown in the following table.

Key: Y - Supported; **N** - Not Supported; N/A - not applicable

Feature	Supported?
Admin-Defined Functions	Y
Box Plots	Y
Custom SQL Queries	Y
Derived Fields (Row-Level Expressions)	Y
Distinct Counts	Y
Fast Distinct Values	N/A
Group By Multiple Fields	Y
Group By Time	Y
Group By UNIX Time	Y
Histogram Floating Point Values	Y
Histograms	Y
Kerberos Authentication	Y
Last Value	Y
Live Mode and Playback	Y
Multivalued Fields	N/A
Nested Fields	N/A

Feature	Supported?
Partitions	**N**
Pushdown Joins for Fusion Data Sources	**N**
Schemas	**Y**
Text Search	N/A
TLS	**Y**
User Delegation	**N**
Wildcard Filters	**Y**
Wildcard Filters, Case-Insensitive Mode	**Y**
Wildcard Filters, Case-Sensitive Mode	**Y**

Connect to SAP IQ

Make sure that the `jConnect` system objects were installed on your Sap IQ database instance (see https://help.sap.com/viewer/a894a54d84f21015b142ffe773888f8c/16.1.3.0/en-US/3bd561266c5f10149e06d363dbe03486.html).

The SAP IQ connector requires a JDBC driver to be configured before your data source configurations can use it. You can download the driver from the vendor's site. For information and steps, see Add A JDBC Driver.

When setting up a connection to SAP IQ , you need to provide the following:

- The name of the connection
- The JDBC URL. The format for the JDBC URL must be as follows: `jdbc:sybase:Tds:<ipaddr>:<port>`. For example, `jdbc:sybase:Tds:10.1.2.3:2638/.`
- The username and password, if applicable.
- Optionally, the microservice principal name.
- Optionally, select **Request Kerberos Session** if Kerberos connection authentication will be used. See Configure Kerberos Support For The SAP IQ Connector for more information.

Configure Kerberos Support for the SAP IQ Connector

Configure Kerberos support for the connector

1. Create or update the file named `/etc/zoomdata/edc-sapiq.properties`. If this file already exists, verify that the information below exists in the file:

```
kerberos.krb5.conf.location=/etc/krb5.conf
kerberos.service.account.authentication=true
kerberos.service.account.principal=<yourcompany_principal>@KERBEROS.REALM
kerberos.service.account.keytab.location=/etc/zoomdata/<yourcompany_principal>.keytab
```

2. Restart the SAP IQ connector.

3. To connect to SAP IQ, use the following JDBC URL template:

```
jdbc:sybase:Tds:host:port/?REQUEST_KERBEROS_SESSION=true&SERVICE_PRINCIPAL_NAME=sap_iq_database@PRINCIPAL.NAME
```

Manage the Snowflake Connector

The Composer Snowflake connector lets you access the data available in Snowflake storage using the Composer client. The Composer Snowflake connector supports whatever Snowflake version is currently available in the cloud.

Before you can establish a connection from Composer to Snowflake storage, a connector server needs to be installed and configured. See Manage Connectors And Connector Servers for general instructions and Connect To Snowflake for details specific to the Snowflake connector.

After setting up the connector, create data sources that specify the necessary connection information and identify the data you want to use. See Manage Data Sources for more information. After you set up your data sources, create dashboards and visuals from from the data in these data sources. See Create Data Discovery Dashboards.

Feature Support

Connector support for specific features is shown in the following table.

Key: Y - Supported; **N** - Not Supported; N/A - not applicable

Feature	Supported?
Admin-Defined Functions	**Y**
Box Plots	**Y**
Custom SQL Queries	**Y**
Derived Fields (Row-Level Expressions)	**Y**
Distinct Counts	**Y**
Fast Distinct Values	N/A
Group By Multiple Fields	**Y**
Group By Time	**Y**
Group By UNIX Time	**Y**
Histogram Floating Point Values	**Y**
Histograms	**Y**
Kerberos Authentication	**N**
Last Value	**Y**
Live Mode and Playback	**Y**
Multivalued Fields	N/A
Nested Fields	N/A

Feature	Supported?
Partitions	**N**
Pushdown Joins for Fusion Data Sources	**Y**
Schemas	**Y**
Text Search	N/A
TLS	**Y**
User Delegation	**N**
Wildcard Filters	**Y**
Wildcard Filters, Case-Insensitive Mode	**Y**
Wildcard Filters, Case-Sensitive Mode	**Y**

Connect to Snowflake

The version 3.12.11 JDBC driver is included with the Snowflake connector, but you can download a newer version from https://repo1.maven.org/maven2/net/snowflake/snowflake-jdbc/. See Add A JDBC Driver.

When setting up a connection to Snowflake, you need to provide the following:

- The name of the connection
- The JDBC URL.
- Each Snowflake connection must be associated with a database. It may be the database specified in the JDBC URL or the default data base of the connecting user (when no database is specified in the JDBC URL).
- The username and password. Only simple username and password authentication is supported.

Snowflake officially supports each of its client versions for a minimum of two years: https://docs.snowflake.net/manuals/release-notes/requirements.html#support-policy. If the JDBC driver is not updated for two years, the Snowflake connector may stop working. Composer regularly updates the JDBC driver, however if you do not update your Snowflake connector for a long time, you may encounter problems. If this happens, you can manually update the JDBC driver yourself. Composer provides it in `/opt/zoomdata/lib/edc-snowflake` for Linux, and `<install-path>/lib/edc-snowflake` for Windows environments.

Snowflake Time Field Conversion

The Composer Snowflake connector converts date-time fields with data types of TIMESTAMP_TZ (a Snowflake data type) to Coordinated Universal Time (UTC) format. The connector also sets the session timezone to UTC format, which means that all Snowflake fields that use the Snowflake local timezone data type

TIMESTAMP_LTZ are also converted to UTC format.

Configure the Snowflake Clustering Depth Threshold

Snowflake does not have an index, but supports micro-partitions and clustering keys instead. It uses a clustering depth for a table column to indicate whether the clustering state of the column has improved or deteriorated as a result of data changes in the table. A value of 1.0 for the clustering depth indicates that the column is fully clustered. A higher clustering depth indicates that the Snowflake table is not optimally clustered. See *Understanding Snowflake Table Structures*.

To define playability of date or numeric fields, the Composer Snowflake connector uses the relative clustering depth of these fields in relation to the total number of partitions in the table, computed as a percentage using the following formula:

```
AverageClusteringDepth / MAX(TotalPartitionCount, 100) * 100
```

If the relative clustering depth of a field is equal to or less than a set threshold value, it is considered to be playable. The default clustering depth threshold is 10%, but can be changed by changing the following Snowflake configuration property in the Snowflake properties file (`edc-snowflake.properties`):

```
snowflake.metadata-detection.fast-range-queries.max-clustering-depth-percent=<nnn>
```

See Connector Properties And Property Files.

The clustering depth threshold allows Composer to enable playback and live mode for all fields that are optimally clustered and disable it for all fields that are not. Adjust the threshold value or recluster your Snowflake tables to better handle intermediate cases.

Connect to Snowflake Using OAuth

To create a Snowflake connection use one of the available authentication methods:

- Basic authentication via username and password
- OAuth 2.0

If connecting using basic authentication, provide:

- The name of the connection.
- The JDBC URL.
- Each Snowflake connection you use must be associated with a database.

  - The database can be the one specified in the JDBC URL, or
  - The default database of the connecting user (if no database is specified in the JDBC URL).

- The username and password. Only simple username and password authentication is supported.

For connecting via OAuth 2.0, fill in the specific parameters:

JDBC URL	
OAuth 2.0 Enabled	TRUE/FALSE
OAuth 2.0 Authorization URI	Obtain OAuth 2.0 connection parameters from your Snowflake instance for connection.
OAuth 2.0 Token URI	
OAuth 2.0 Client Id	
OAuth 2.0 Client Secret	

Note: Scheduled source refresh is not available when you use OAuth 2.0 authentication.

Note: If you do not want to expose OAuth 2.0 connection options to your customers, disable OAuth-related connection parameters at the connector level as a member of the Supervisors group.

Manage the Spark SQL Connector

The Composer Spark SQL connector lets you access the data available in Spark SQL databases using the Composer client. The Composer Spark SQL connector supports Spark SQL versions 2.3 through 3.0.

Before you can establish a connection from Composer to Spark SQL storage, a connector server needs to be installed and configured. See Manage Connectors And Connector Servers for general instructions and Connect To Spark SQL for details specific to the Spark SQL connector.

After setting up the connector, create data sources that specify the necessary connection information and identify the data you want to use. See Manage Data Sources for more information. After you set up your data sources, create dashboards and visuals from from the data in these data sources. See Create Data Discovery Dashboards.

Feature Support

Connector support for specific features is shown in the following table.

Key: Y - Supported; **N** - Not Supported; N/A - not applicable

Feature	Supported?	Notes
Admin-Defined Functions	Y	
Box Plots	Y	
Custom SQL Queries	Y	If you need to access a BigQuery partition, explicitly include an alias for the built in partition column in your select clause, such as `select *, _PARTITIONTIME as pt from projectId.datasetId.tableId`.
Derived Fields (Row-Level Expressions)	Y	
Distinct Counts	Y	
Fast Distinct Values	N/A	
Group By Multiple Fields	Y	
Group By Time	Y	
Group By UNIX Time	Y	
Histogram Floating Point Values	Y	
Histograms	Y	
Kerberos Authentication	Y	To enable Kerberos authentication, see Connect To Spark SQL Sources On A Kerberized HDP Cluster.
Last Value	Y	
Live Mode and Playback	Y	

Feature	Supported?	Notes
Multivalued Fields	N/A	
Nested Fields	N/A	
Partitions	**Y**	
Pushdown Joins for Fusion Data Sources	**Y**	
Schemas	**Y**	
Text Search	N/A	
TLS	**N**	
User Delegation	**N**	
Wildcard Filters	**Y**	
Wildcard Filters, Case-Insensitive Mode	**Y**	
Wildcard Filters, Case-Sensitive Mode	**Y**	

Connect to Spark SQL

When establishing a connection to Spark SQL, you need to provide the following information when setting up the partition settings.

Configure the partition settings. For the partitioned fields you can select one of the following options:

- **No**
- **Date** - this option is available for the Time field type. If you select this option, the list of the partitioned columns will be displayed in the Configure column.

Numeric and time-based fields can be edited using the Fields tab:

- Numeric type Number - ability to select a default aggregation function
- Time fields - ability to define the default time pattern and granularity; if the time field provides granularities of hour, minute and second, then a time zone label may be applied.

When you create a data source, the specific number of distinct values for the attribute fields are saved in Composer depending on the data sample from your data set. You can filter the data on your visual by these values. While editing a data source, if you want to use all distinct values in the filter (that is from whole data source), select **Refresh** in the **Statistics** column.

Connect to Spark SQL Sources on a Kerberized HDP Cluster

A secure Hortonworks Data Platform (HDP) cluster uses Kerberos authentication to validate and confirm access requests. You can set up Composer to connect to the secure HDP cluster using the following instructions.

Prepare the Spark SQL cluster

- To enable Kerberos for HDP distribution using a Spark SQL source, refer to Hortonwork's documentation Enabling Kerberos Authentication Using Ambari.
- Kerberos authentication requires precise time correspondence on all instances to work properly. You need to enable the Network Time Protocol service in your network. For more information, access the topic Using the Network Time Protocol to Synchronize Time.
- Set up a Thrift JDBC/ODBC server in your environment. See Spark documentation.

Configure Composer Microservices

Obtain Kerberos Credentials

Each microservice must have its own unique identifier called a principal. Perform the following steps:

1. Install the Kerberos client on the CentOS or Ubuntu machine where the Composer server resides.
2. Generate the Kerberos principal and corresponding keytab for the Composer microservice. Before you proceed, make sure that:

i. The Composer microservice is running on a node with proper Kerberos configuration: `/etc/krb5.conf` or similar location for your Linux distribution.

ii. The Kerberos realm on your environment is the same as the realm specified in the `kdc.conf` file from the Spark SQL server.

3. Check the Kerberos configuration (that is, `krb5.conf`) and validity of the principal and keytab pair using MIT Kerberos client:

```
kinit -V -k -t <composer_principal>.keytab <composer_principal@KERBEROS.REALM>
```

4. Make the keytab accessible for Composer's Spark SQL connector:

```
sudo mkdir /etc/zoomdata
sudo mv <composer_principal>.keytab /etc/zoomdata
sudo chown zoomdata:zoomdata /etc/zoomdata/<composer_principal>.keytab
sudo chmod 600 /etc/zoomdata/<composer_principal>.keytab
```

Configure a Spark SQL Connector

1. Create or update the file named `/etc/zoomdata/edc-sparksql.properties`. If this file already exists, verify that the information below exists in the file:

```
kerberos.krb5.conf.location=/etc/krb5.conf
kerberos.service.account.authentication=true
kerberos.service.account.principal=<composer_principal@KERBEROS.REALM>
kerberos.service.account.keytab.location=/etc/zoomdata/<composer_principal>.keytab
```

2. Restart the Spark SQL connector:

```
sudo systemctl restart zoomdata-edc-sparksql
```

Connect to the Kerberized Spark SQL Source

You are now ready to create the Spark SQL source:

1. Open a new browser window and log into Composer.
2. Select **Sources**.
3. Select **Spark SQL**.
4. Specify the name of your source and add a description (if desired). Then select **Next**.
5. On the **Connection** page, define the connection source. You can use an existing connection, if available, or create a new one. To create a new connection, select the **Input New Credentials** option button and specify the connection name and JDBC URL. Make sure that you enter the JDBC URL in the correct format:

```
jdbc:hive2://<spark-sql-host>:10000/;principal=<spark-sql-principal@KERBEROS.REALM>
```

Replace the placeholders as follows:

i. `<spark_sql_host>`: Specify the IP address or host name of the Spark SQL node to which you are connecting.

ii. `<spark_sql_principal@KERBEROS.REALM>`: Enter the principal of the Spark SQL node you are connecting to. To get the list of all Spark SQL principals, navigate to Ambari > Admin > Kerberos > Advanced > Spark SQL.

> **Note:** The `principal` spec contained in the JDBC URL refers to the principal of the Spark SQL node. `spark_sql_principal@KERBEROS.REALM` principal has nothing to do with the `zoomdata_principal@KERBEROS.REALM` principal specified for the Composer connector.

6. Select **Validate** and, after your connection is valid, select **Next**.

You can continue configuring the data source as described in Manage Data Sources.

After you have completed the configuration, Composer will begin accessing Spark SQL using `zoomdata_principal@KERBEROS.REALM` authenticated by its keytab in `/etc/zoomdata/<composer_principal>.keytab`.

Manage the Teradata Connector

The Composer Teradata connector lets you access the data available in Teradata databases using the Composer client. The Composer Teradata connector supports Teradata version 16.20.

Before you can establish a connection from Composer to Teradata storage, a connector server needs to be installed and configured. See Manage Connectors And Connector Servers for general instructions and Connect To Teradata for details specific to the Teradata connector.

After setting up the connector, create data sources that specify the necessary connection information and identify the data you want to use. See Manage Data Sources for more information. After you set up your data sources, create dashboards and visuals from from the data in these data sources. See Create Data Discovery Dashboards.

Feature Support

Connector support for specific features is shown in the following table.

Key: Y - Supported; **N** - Not Supported; N/A - not applicable

Feature	Supported?
Admin-Defined Functions	**Y**
Box Plots	**Y**
Custom SQL Queries	**Y**
Derived Fields (Row-Level Expressions)	**Y**
Distinct Counts	**Y**
Fast Distinct Values	N/A
Group By Multiple Fields	**Y**
Group By Time	**Y**
Group By UNIX Time	**Y**
Histogram Floating Point Values	**Y**
Histograms	**Y**
Kerberos Authentication	**N**
Last Value	**Y**
Live Mode and Playback	**Y**
Multivalued Fields	N/A
Nested Fields	N/A

Feature	Supported?
Partitions	**N**
Pushdown Joins for Fusion Data Sources	**Y**
Schemas	**Y**
Text Search	N/A
TLS	**Y**
User Delegation	**N**
Wildcard Filters	**Y**
Wildcard Filters, Case-Insensitive Mode	**Y**
Wildcard Filters, Case-Sensitive Mode	**Y**

Connect to Teradata

The Teradata connector requires a JDBC driver to be configured before you can connect to your data source. You can download the driver from the vendor's site. If you are upgrading, keep in mind you need to configure the JDBC driver - see Upgrade ComposerSymphony for instructions. For more information and steps, see Add A JDBC Driver.

You can limit the list of available collections in a Teradata data store to collections from a specific schema specified in the Teradata JDBC URL. Use the Teradata JDBC URL `database` property to limit the list of available schemas as well as the list of available collections to collections from the schema. For example:

```
jdbc:teradata://<url>/dbs_port=<port>,database=<schema>
```

Be sure to work with your Teradata database administrator when you do this.

Manage the TIBCO Data Virtualization (TDV) Connector

The Composer Tibco connector lets you access the data available in your TIBCO Data Virtualization (TDV) storage using the Composer client. It supports TDV version 8.0-8.1 and has been tested with the following back-end databases:

Database	Comments
MS SQL 14.0	
PostgreSQL 9.3	
Oracle 11	
MemSQL 6.7	Wildcard filters don't work when a single quote is used in the search expression.

Different data stores behave differently when comparing strings (for example, they handle white space and case sensitivity differently). For this reason, the Composer TDV connector supports the comparison behavior of the underlying data store and does not enforce a behavior of its own.

Before you can establish a connection from Composer to TDV, a connector server needs to be installed and configured. See Manage Connectors And Connector Servers for general instructions and Connect To TDV for details specific to the Couchbase connector.

After setting up the connector, create data sources that specify the necessary connection information and identify the data you want to use. See Manage Data Sources for more information. After you set up your data sources, create dashboards and visuals from from the data in these data sources. See Create Data Discovery Dashboards.

Feature Support

Connector support for specific features is shown in the following table.

Key: Y - Supported; **N** - Not Supported; N/A - not applicable

Feature	Supported?
Admin-Defined Functions	Y
Box Plots	Y
Custom SQL Queries	Y
Derived Fields (Row-Level Expressions)	Y
Distinct Counts	Y
Fast Distinct Values	N/A
Group By Multiple Fields	Y

Feature	Supported?
Group By Time	**Y**
Group By UNIX Time	**Y**
Histogram Floating Point Values	**Y**
Histograms	**Y**
Kerberos Authentication	**N**
Last Value	**Y**
Live Mode and Playback	**Y**
Multivalued Fields	N/A
Nested Fields	N/A
Partitions	N/A
Pushdown Joins for Fusion Data Sources	**Y**
Schemas	**Y**
Text Search	N/A
TLS	**N**
User Delegation	**N**
Wildcard Filters	**Y**
Wildcard Filters, Case-Insensitive Mode	**Y**
Wildcard Filters, Case-Sensitive Mode	**N**

Connect to TDV

The TDV connector requires a JDBC driver to be configured before you can connect to your data source. You can obtain the driver from the TDV installer distribution.

To connect to a TDV data store, you must specify the JDBC URL for the TDV database, and, if necessary, the username and password credentials for the TDV database. The structure of the connection URL is:

```
jdbc:compositesw:dbapi@<host>:<port>?domain=<domain>&dataSource=<datasource>
```

where

- `<host>` is the TDV server host name or IP address.
- `<port>` is the optional TDV server port for JDBC connections. The default is 9401.
- `<domain>` is the TDV domain in which the <datasource> belongs.
- `<datasource>` is the TDV data source to which you want to connect.

If you are upgrading, keep in mind you need to configure the JDBC driver - see *Upgrade The Composer V24 Server* for instructions. For more information and steps, see *Add A JDBC Driver*.

Manage the Vertica Connector

The Composer Vertica connector lets you access the data available in the Vertica SQL query engine using the Composer client. The Composer Vertica connector supports Vertica version 7.2.

Before you can establish a connection from Composer to Vertica storage, a connector server needs to be installed and configured. See Manage Connectors And Connector Servers for general instructions and Connect To Vertica for details specific to the Vertica connector.

After setting up the connector, create data sources that specify the necessary connection information and identify the data you want to use. See Manage Data Sources for more information. After you set up your data sources, create dashboards and visuals from from the data in these data sources. See Create Data Discovery Dashboards.

Feature Support

Connector support for specific features is shown in the following table.

Key: Y - Supported; **N** - Not Supported; N/A - not applicable

Feature	Supported?
Admin-Defined Functions	**Y**
Box Plots	**Y**
Custom SQL Queries	**Y**
Derived Fields (Row-Level Expressions)	**Y**
Distinct Counts	**Y**
Fast Distinct Values	N/A
Group By Multiple Fields	**Y**
Group By Time	**Y**
Group By UNIX Time	**Y**
Histogram Floating Point Values	**Y**
Histograms	**Y**
Kerberos Authentication	**N**
Last Value	**Y**
Live Mode and Playback	**Y**
Multivalued Fields	N/A
Nested Fields	N/A

Feature	Supported?
Partitions	**N**
Pushdown Joins for Fusion Data Sources	**Y**
Schemas	**Y**
Text Search	N/A
TLS	**Y**
User Delegation	**N**
Wildcard Filters	**Y**
Wildcard Filters, Case-Insensitive Mode	**Y**
Wildcard Filters, Case-Sensitive Mode	**Y**

Connect to Vertica

The Vertica connector requires a JDBC driver to be configured before you can connect to your data source. You can download the driver from the vendor's site. If you are upgrading, keep in mind you need to configure the JDBC driver - see Upgrade ComposerSymphony for instructions. For more information and steps, see Add A JDBC Driver .

Composer supports Flex tables. However, you must properly map the metadata for the Flex tables in Vertica to use these data in Composer.

The Interval data type in Vertica will be converted to the Number type in Composer. The values for the Interval data type will be converted as follows in Composer:

- INTERVAL [DAY TO SECOND] - number of seconds
- INTERVAL [YEAR TO MONTH] - number of months

The fields containing these converted data will be marked as Raw Data Only. You can view these data in the Details window on a visual or export as raw data .

Manage File Uploads

Composer can visualize data from file uploads including CSV, JSON, and TSV files.

- The maximum supported file size is 500 MB
- Data from the file upload is stored in a PostgreSQL database
- When you upload a flat data file, the first 1,000 records are used to determine if fields are imported as NUMBER or INTEGER.
  - If there are no decimal number records in the first 1,000 records, the numeric fields are imported as INTEGER instead of NUMBER: any decimal number records after row 1,000 may not upload fully.
  - To ensure all records are imported, sort your data to ensure decimal numbers are included in the first 1,000 records.

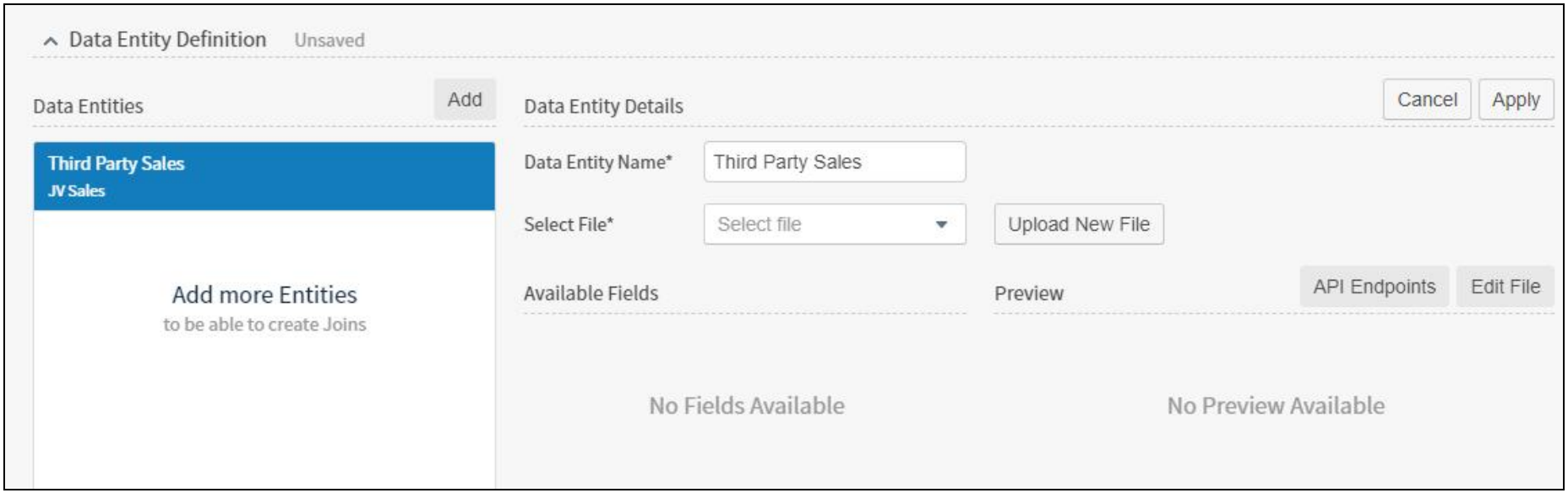
Before you can establish a connection from Composer to your file uploads storage, a connector server needs to be installed and configured. See Manage Connectors And Connector Servers for general instructions.

After the connector has been set up, create a data source configuration and upload your file or files to that source. See:

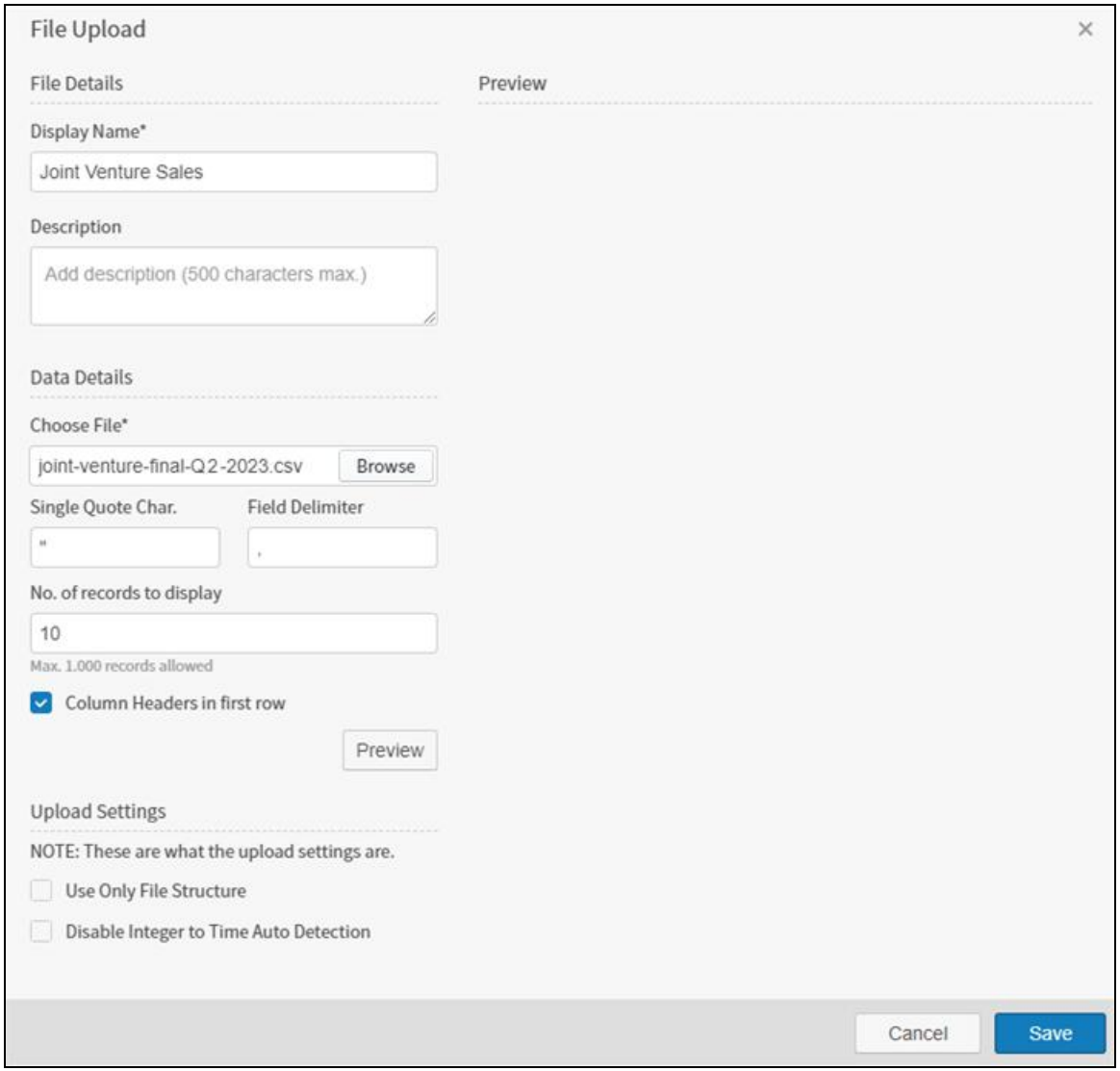
- Define A Source
- Upload A New File
- Edit An Existing File
- Delete a File
- API Endpoints

Upload a New File

1. Log in as a user with the **Administer Sources** or **Create New Data Sources** privilege.
2. Create or edit an existing data source.
3. Select **Add** to add a new data entity, then select **From File**.



4. Add a unique Data Entity Name, then select **Upload New File**. The File Upload work area opens.

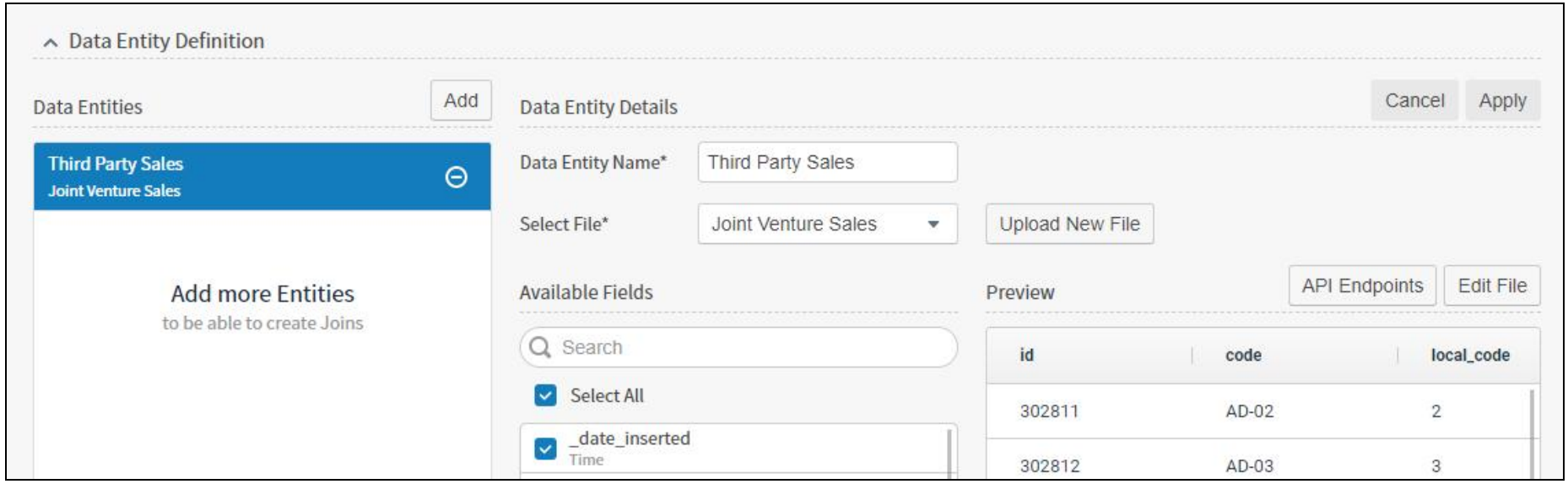


5. Enter File Details, such as a unique **Display Name**, and optional **Description**.
6. Use the **Browse** button to select a file to upload.
7. After you have selected a file, Composer may autofill the **Single Quote Char.** and Field **Delimiter** fields. Adjust if needed.

8. If needed, change the **No. of records to display** from 10 to up to 1,000 records.
9. Enable or disable the checkbox **Column Headers** in first row to match your file layout. If no column headers are in your data, Composer assigns numerical column headings, **field_1**, **field_2**, and so on.
10. Optionally, select **Preview** to preview your data.
11. Enable the checkbox **Use Only File Structure** to create the file using only the existing file structure, no data.
12. Enable the checkbox **Disable Integer to Time Auto Detection** to prevent auto detection of time fields.
13. Select **Save** to close the File Upload work area and continue creating or updating your data entity for this source.

Edit an Existing File

1. Log in as a user with the **Administer Sources** or **Create New Data Sources** privilege.
2. Edit an existing data source.

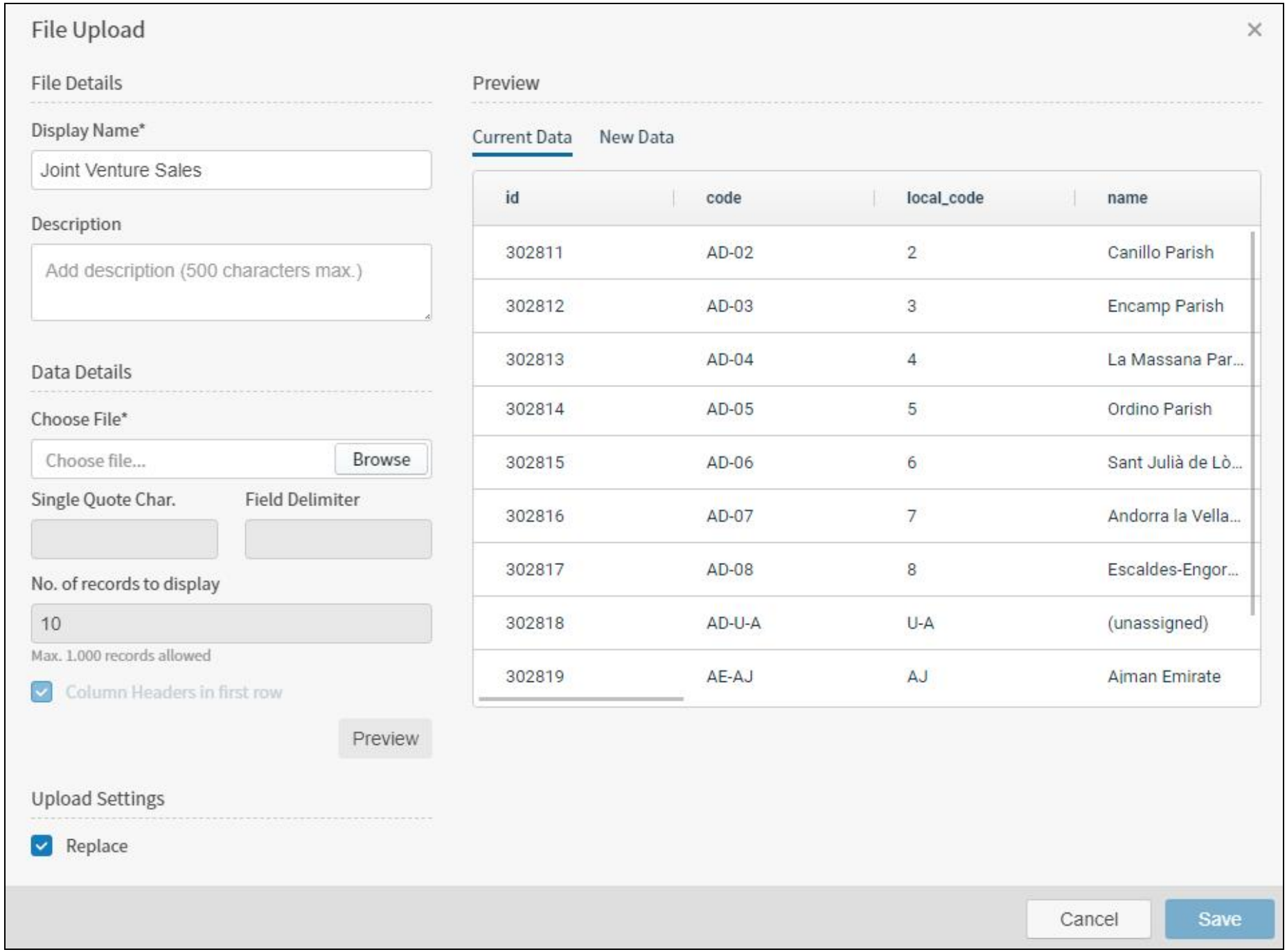


3. Select the data entity with the file you want to edit, then select **Edit File**. The File Upload work area opens.

4. **Browse** for a new file.

i. If you are replacing the existing file, your new file must use the same data file structure as the existing file. Enable the **Replace** checkbox in Upload Settings: previous data is replaced.

ii. If you are adding data to the existing file, your new file must use the same data file structure as the existing file. Disable the **Replace** checkbox in Upload Settings: previous data is appended with new rows of data.


File Upload
File Details
Display Name*
Joint Venture Sales
Description
Add description (500 characters max.)
Data Details
Choose File*
Choose file...
Browse
Single Quote Char.
Field Delimiter
No. of records to display
10
Max. 1.000 records allowed
Column Headers in first row
Preview
Upload Settings
Replace
Preview
Current Data
New Data
id
code
local_code
name
302811
AD-02
2
Canillo Parish
302812
AD-03
3
Encamp Parish
302813
AD-04
4
La Massana Par...
302814
AD-05
5
Ordino Parish
302815
AD-06
6
Sant Julià de Lò...
302816
AD-07
7
Andorra la Vella...
302817
AD-08
8
Escaldes-Engor...
302818
AD-U-A
U-A
(unassigned)
302819
AE-AJ
AJ
Ajman Emirate
Cancel
Save

5. Select **Preview** to preview your data.

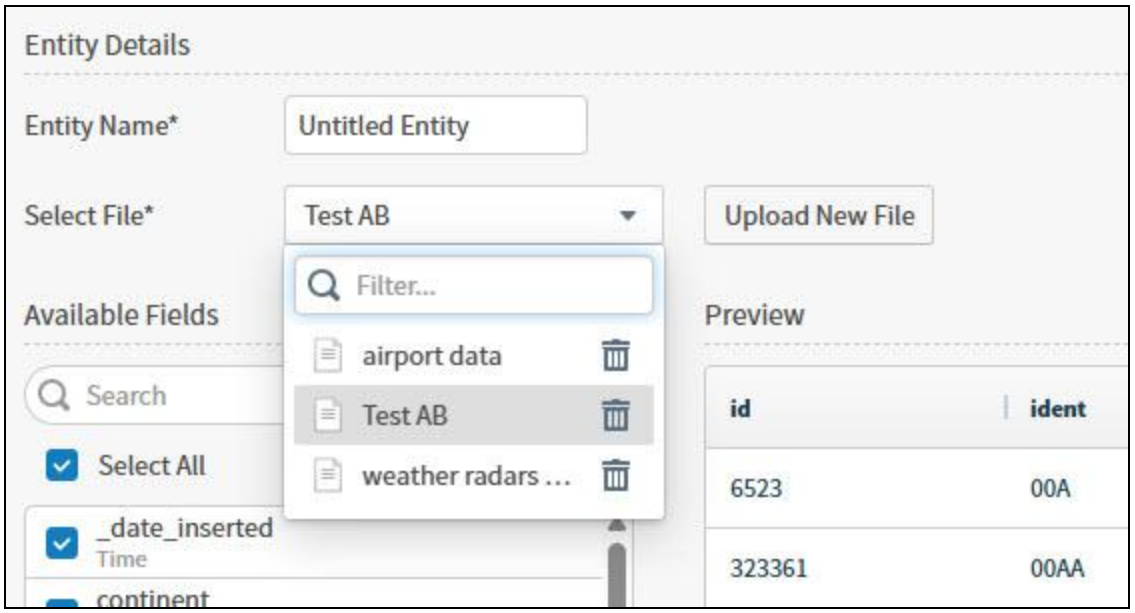
i. The Current Data tab shows the data of your existing file.

ii. The New Data shows the replaced or amended data preview.

6. Select Save to **Save** your changes or **Cancel** to discard your changes. The File Upload work area closes.

Delete a File

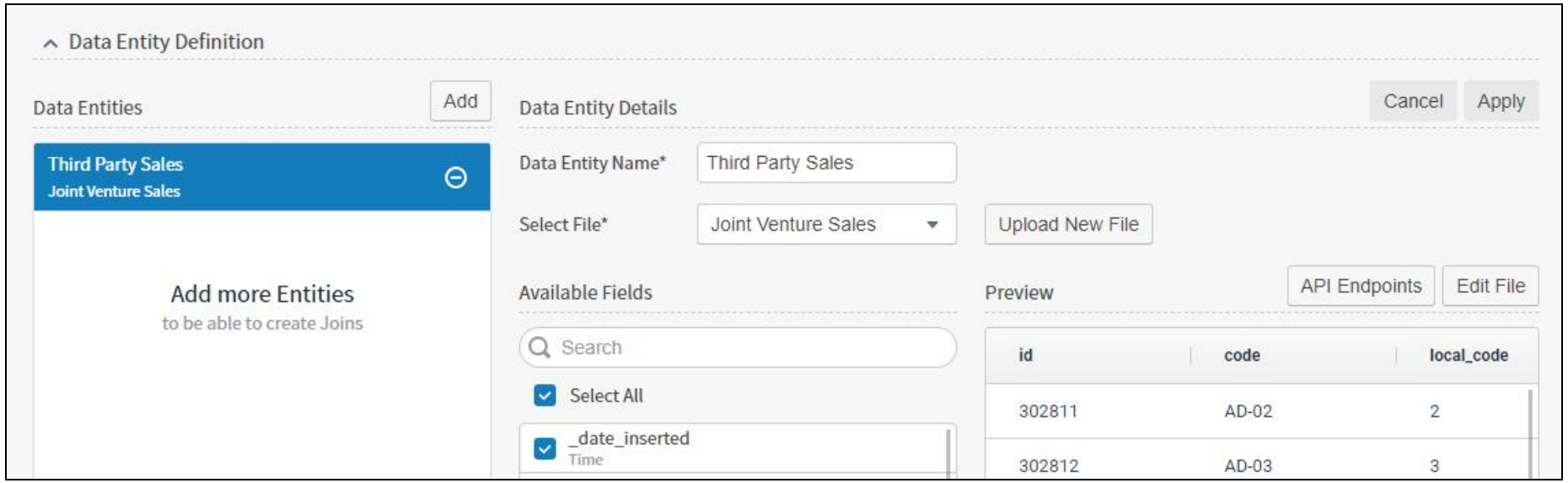
If you have the privilege to manage file uploads, you can delete uploaded files as needed. Select the trash can icon next to the file name in the **Select File** dropdown.



API Endpoints

1. Log in as a user with the **Administer Sources** or **Create New Data Sources** privilege.

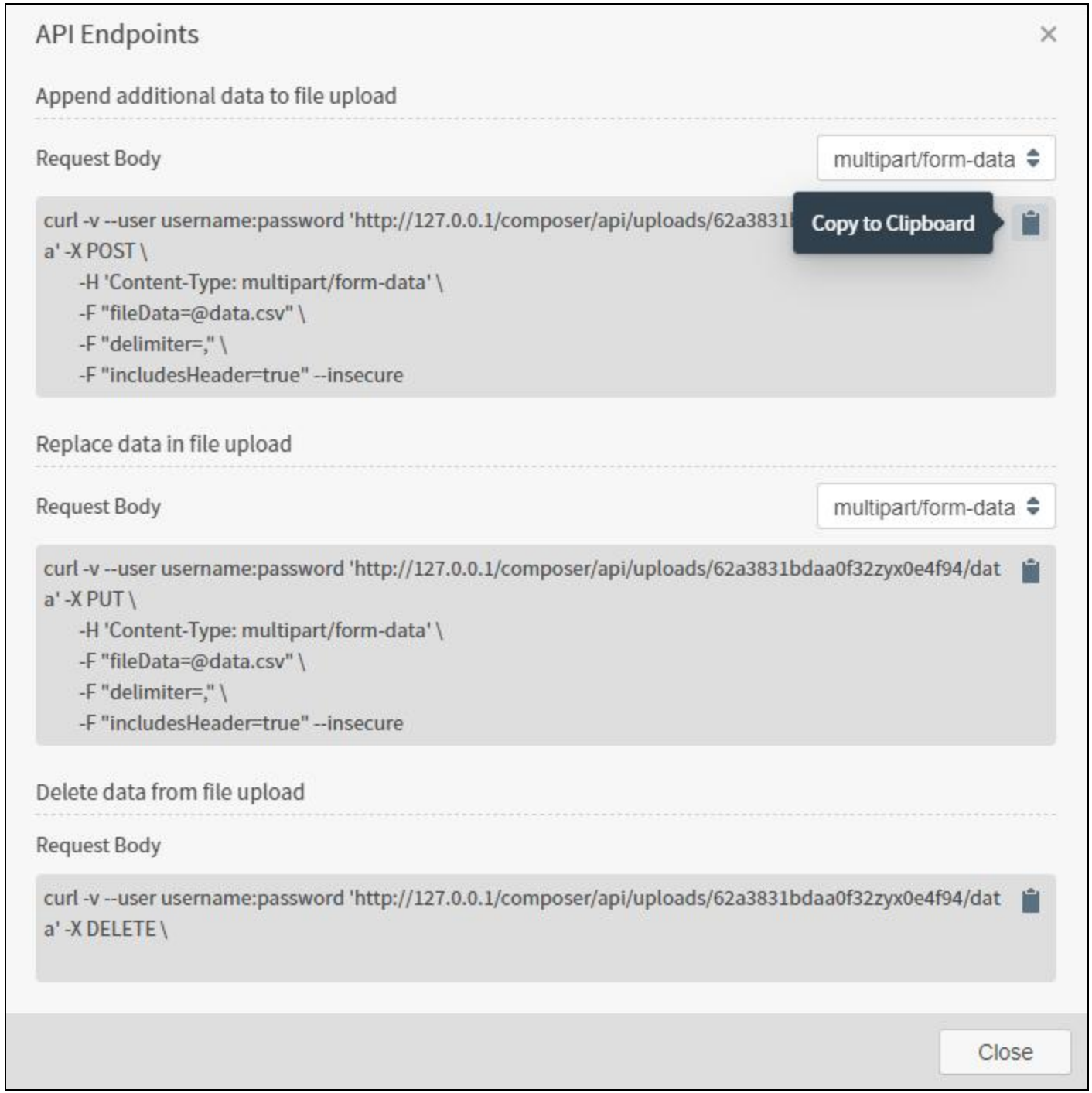
2. Edit an existing data source.



3. Select the data entity with the file you want to edit, then select **API Endpoints**. The API Endpoints dialog box opens.

   The dialog offers convenient example cURL requests but the APIs can be leveraged from your preferred development platform.

4. Copy and modify the example cURL requests to include your own Composer credentials, replacing the placeholders for username and password. Select **Close** to close the dialog.


API Endpoints
Append additional data to file upload
Request Body
multipart/form-data
curl -v --user username:password 'http://127.0.0.1/composer/api/uploads/62a3831
a' -X POST \
-H 'Content-Type: multipart/form-data' \
-F "fileData=@data.csv" \
-F "delimiter=," \
-F "includesHeader=true" --insecure
Copy to Clipboard
Replace data in file upload
Request Body
multipart/form-data
curl -v --user username:password 'http://127.0.0.1/composer/api/uploads/62a3831bdaa0f32zyx0e4f94/dat
a' -X PUT \
-H 'Content-Type: multipart/form-data' \
-F "fileData=@data.csv" \
-F "delimiter=," \
-F "includesHeader=true" --insecure
Delete data from file upload
Request Body
curl -v --user username:password 'http://127.0.0.1/composer/api/uploads/62a3831bdaa0f32zyx0e4f94/dat
a' -X DELETE \
Close

Work with the Upload API

There are two operations that can be performed using the Upload API: appending additional data and clearing previously uploaded data. The source creation page offers convenient example cURL requests but the APIs can be leveraged from your preferred development platform. Select API Endpoints on the source creation tab to edit your data.

Modify the example cURL requests to include your own Composer credentials, replacing the placeholders for username and password.

```
curl -v --user <username>:<password> <YourServer>
```

Example: Append Data

In the following example, the Upload API accepts an array of JSON objects. Note that the object field types must match those used to create the Upload API source originally. For example, if the value of the ***price*** field is a number, you can not upload new rows in which the value of the ***price*** field is a string.

```
curl -v --user <username>:<password> 'https://<Your_Composer_Server>/ composer/api/upload/<YourDataSourceId>' -X POST -H
"Content-Type: application/vnd.composer.v3+json" -d '[{"price":100.5,"venue_id":"V678","venue_name": "Pizza Barn"}]' --
insecure
```

Example: Clear Previously Uploaded Data

In the example below, the Upload API will clear all previously uploaded data from the data source with the ID of `<YourDataSourceId>`.

```
curl -v --user <username>:<password> 'https://<Your_Composer_Server>/ api/upload/<YourDataSourceId>' -X DELETE --insecure
```

Feature Support

File upload support for specific features is shown in the following table.

Key: Y - Supported; **N** - Not Supported; N/A - not applicable

Feature	Supported?
Admin-Defined Functions	**Y**
Box Plots	**Y**
Custom SQL Queries	**Y**

Feature	Supported?
Derived Fields (Row-Level Expressions)	**Y**
Distinct Counts	**Y**
Fast Distinct Values	N/A
Group By Multiple Fields	**Y**
Group By Time	**Y**
Group By UNIX Time	**Y**
Histogram Floating Point Values	**Y**
Histograms	**Y**
Kerberos Authentication	**N**
Last Value	**Y**
Live Mode and Playback	**Y**
Multivalued Fields	N/A
Nested Fields	N/A
Partitions	**N**
Pushdown Joins for Fusion Data Sources	**Y**
Schemas	**Y**
Text Search	N/A
TLS	**Y**
User Delegation	**N**
Wildcard Filters	**Y**
Wildcard Filters, Case-Insensitive Mode	**Y**
Wildcard Filters, Case-Sensitive Mode	**Y**

Manage the Real Time Sales Demo Source

A demo data source called Real Time Sales (RTS) is included as part of the Composer installation package. This data generator simulates a real-time data stream; allowing users to interact with and explore Composer's capabilities without the need to connect to a data source. However, this demo source, by default, is not available in the Composer Client and must be manually activated. This topic walks you through how to enable and disable the demo source.

Note: A known issue with real-time streaming sources such as this demo source exists. If such sources are left enabled for an extended period of time, 'Out of Memory' errors may occur in the Composer Server. To avoid this problem, disable such sources when not in use.

Before you can establish a connection from Composer to RTS, a data source configuration for the demo source needs to be enabled and set up. See Enable The Real Time Sales Demo Source and Set Up The Real Time Sales Demo Source . To manage the availability for RTS, see Manage Connectors And Connector Servers.

After setting up the connector, create data sources that specify the necessary connection information and identify the data you want to use. See Manage Data Sources for more information. After you set up your data sources, create dashboards and visuals from from the data in these data sources. See Create Data Discovery Dashboards.

Feature Support

Real Time Sales Demo support for specific features is shown in the following table.

Key: Y - Supported; **N** - Not Supported; N/A - not applicable

Feature	Supported?
Admin-Defined Functions	N/A
Box Plots	**Y**
Custom SQL Queries	N/A
Derived Fields (Row-Level Expressions)	N/A
Distinct Counts	**Y**
Fast Distinct Values	N/A
Group By Multiple Fields	**Y**
Group By Time	**Y**
Group By UNIX Time	**N**
Histogram Floating Point Values	**Y**
Histograms	**Y**
Kerberos Authentication	N/A

Feature	Supported?
Last Value	Y
Live Mode and Playback	Y
Multivalued Fields	N/A
Nested Fields	N/A
Partitions	N/A
Pushdown Joins for Fusion Data Sources	N/A
Schemas	Y
Text Search	N/A
TLS	N/A
User Delegation	N/A
Wildcard Filters	Y
Wildcard Filters, Case-Insensitive Mode	Y
Wildcard Filters, Case-Sensitive Mode	Y

Enable the Real Time Sales Demo Source

To enable the Real Time Sales demo source, use an automated script that activates the demo source in the Composer client. This script, labeled `create-rts.sh`, is included as part of the installation package. However, the script needs to be run from the Linux prompt, so administrative access to your Linux server is necessary.

RTS is installed in your server as part of the installation process. However, if an alternative installation method was used to manually install each Composer component, you should first confirm whether the RTS package was included during the installation efforts.

More specifically, running the automated RTS script will do the following (in the Composer Client):

1. Add the RTS connector server details in the Composer Client (using port 8108).
2. Define the connection type for the RTS demo source.
3. Create the connection and generate the RTS icon in the Data Source page.

Set Up the Real Time Sales Demo Source

To set up the RTS demo source, take the following steps:

1. Log out of the Composer client and close the browser window.
2. Access the Linux prompt and log into your Composer Server (via Secure Shell or SSH).
3. From your Linux prompt, run the RTS script:

```
sudo /opt/zoomdata/lib/create-rts.sh -a admin:<YourAdminPassword>
-s supervisor:<YourSupervisorPassword>
```

Enable RTS After Upgrading Composer

If you are upgrading your Composer Server to the current release version, that version's RTS demo source is deleted during the process and the current version is installed. You need to activate RTS following the same steps outlined above.

Disable the Real Time Sales Demo Data Source

To disable the Real Time Sales demo source, do the following:

- Log in as an admin user.
- Select the **Connectors** tab.
- Go to the **Connectors** list in the bottom half of the Connectors page.
- Locate the **RTS** connector in the Connectors list.
- Clear the checkbox in the **Enabled** column associated with the RTS connector.

Connector Feature Support

Composer queries each connector to better understand its data store's capabilities and behavior. The connector's response describes to Composer the Composer features that the connector and the data store can support and any limitations to that support. It identifies the type of data requests that the connector and its data store can fulfill.

To learn more about a Composer feature and the connectors that support it, select the feature from the list below, organized by feature category:

Derived Fields (Row-Level Expressions)

- Admin-Defined Functions

Advanced Visualizations

- Box Plot
- Histograms
- Histogram Floating Point Values

Group By Functionality

- Group By Multiple Fields
- Group By Time
- Group By UNIX Time

Filters

- Wildcard Filters
- Wildcard Filters Case-Insensitive

- Wildcard Filters Case-Sensitive
- Text Search

Metrics

- Distinct Counts
- Last Value

Security

- Kerberos Authentication
- TLS
- User Delegation

Custom SQL Queries

Live Mode and Playback

Multivalued Fields

Nested Fields

Schemas

Performance

- Fast Distinct Values
- Partitions

- Pushdown Joins for Fusion Data Sources

Note: Fused data sources inherit the limitations of the underlying connectors used by the fused sources. In addition, fused sources have other feature limitations. See Data Fusion Limitations.

Custom SQL Queries

Applicable only to SQL-based connectors, a data source using a connector that supports custom SQL queries can use an SQL query to select fields from the table. The custom SQL statement can be specified on the Custom SQL area of the Source Creation tab after selecting the **Custom SQL** option. Any visual you create displays fields in the order they are retrieved from the source. When you create a source using custom SQL, your field data is shown in the order you specify.

Important: Custom SQL queries are a powerful tool for performing complex data queries. However, be careful when creating custom SQL queries because it is easy to define a heavy query or a query that may overwhelm your database. Use this feature carefully.

In SQL-based sources, Composer typically wraps the query with select * from. For example, suppose the original query is this:

```
select count(*), someField from myCollection GROUP By someField
```

The resulting query that Composer uses is this:

```
select * from (select count(*), someField from myCollection GROUP By someField)
```

Support for this feature by connector is shown in the following table.

Key:Y - Supported; **N** - Not Supported; N/A - not applicable

Connector	Supported?
Amazon Redshift	**Y**
Amazon S3	**N**
Apache Drill	**Y**
Apache Phoenix	**Y**
Apache Phoenix Query Server (QS)	**Y**
Apache Solr	N/A
BigQuery	**Y**
Business Central Jet	**N**
Cloudera Impala	**Y**
Cloudera Search	N/A
Couchbase	**N**

Connector	Supported?
Dremio	**Y**
Elasticsearch 7.0	N/A
Elasticsearch 8.0	N/A
File Upload	**Y**
HDFS	**N**
Hive	**Y**
Jira	**Y**
MemSQL	**Y**
Microsoft SQL Server	**Y**
MongoDB	N/A
MySQL	**Y**
Oracle	**Y**
PostgreSQL	**Y**
Python	**Y**
Real Time Sales	**N**
Salesforce	**Y**
SAP Hana	**Y**
SAP S/4HANA	**Y**
SAP IQ	**Y**
Spark SQL	**Y**
Snowflake	**Y**
Teradata	**Y**
TIBCO DV	**Y**
Trino	**Y**
File Upload (Upload API)	**N**
Vertica	**Y**

Fast Distinct Values

Connectors that support fast distinct values can efficiently return distinct values for a field. This functionality optimizes the retrieval of distinct (unique) values in large numbers of records. If a connector supports this feature, the Filter dialog is populated with distinct values for an attribute directly from the data source, without the need to refresh the data and without retrieving or storing the distinct values in the metadata. For example, Elasticsearch keeps lists of distinct values at the ready. Features such as these make fast distinct values possible for your connector.

There is no metric that defines "fast". This value is based on the judgment of the developer.

When custom ranges or list values are requested for a field, full data scans are not performed.

For most connectors, this feature can be safely left disabled without impact.

Support for this feature by connector is shown in the following table.

Key:Y - Supported; **N** - Not Supported; N/A - not applicable

Connector	Supported?
Amazon Redshift	N/A
Amazon S3	N/A
Apache Drill	N/A
Apache Phoenix	N/A
Apache Phoenix Query Server (QS)	N/A
Apache Solr	**Y**
BigQuery	N/A
Business Central Jet	N/A
Cloudera Impala	N/A
Cloudera Search	**Y**
Couchbase	N/A
Dremio	N/A
Elasticsearch 7.0	**Y**
Elasticsearch 8.0	**Y**
File Upload	N/A
HDFS	N/A
Hive	N/A
Jira	**N**

Connector	Supported?
MemSQL	N/A
Microsoft SQL Server	N/A
MongoDB	N/A
MySQL	N/A
Oracle	N/A
PostgreSQL	N/A
Python	**N**
Real Time Sales	N/A
Salesforce	**N**
SAP Hana	N/A
SAP S/4HANA	N/A
SAP IQ	N/A
Spark SQL	N/A
Snowflake	N/A
Teradata	N/A
TIBCO DV	N/A
Trino	N/A
File Upload (Upload API)	N/A
Vertica	N/A

Group By Multiple Fields

Many connectors can group by more than one field in a query. Here is a sample SQL query:

```
select firstField, secondField, count(distinct otherField) from myCollection group by firstField, secondField
```

If multi-group querying is not supported, some visuals will be unavailable for a data source.

Support for this feature by connector is shown in the following table.

Key:Y - Supported; **N** - Not Supported; N/A - not applicable

Connector	Supported?	Notes
Amazon Redshift	**Y**	
Amazon S3	**Y**	
Apache Drill	**Y**	
Apache Phoenix	**Y**	
Apache Phoenix Query Server (QS)	**Y**	
Apache Solr	**Y**	
BigQuery	**Y**	If you need to access a BigQuery partition, explicitly include an alias for the built in partition column in your select clause, such as `select *, _PARTITIONTIME as pt from projectId.datasetId.tableId`.
Business Central Jet	**Y**	
Cloudera Impala	**Y**	
Cloudera Search	**N**	
Couchbase	**Y**	
Dremio	**Y**	
Elasticsearch 7.0	**Y**	
Elasticsearch 8.0	**Y**	
File Upload	**Y**	
HDFS	**Y**	
Hive	**Y**	
Jira	**Y**	

Connector	Supported?	Notes
MemSQL	Y	
Microsoft SQL Server	Y	
MongoDB	Y	
MySQL	Y	
Oracle	Y	
PostgreSQL	Y	
Python	Y	
Real Time Sales	Y	
Salesforce	Y	
SAP Hana	Y	
SAP S/4HANA	Y	
SAP IQ	Y	
Spark SQL	Y	
Snowflake	Y	
Teradata	Y	
TIBCO DV	Y	
Trino	Y	
File Upload (Upload API)	Y	
Vertica	Y	

Group By Time

Many connectors support grouping data by a real time field. This function is a prerequisite for all time-based visuals.

Most commonly, the data includes a date or time stamp field type that corresponds to Composer 's date field type. Here is a sample SQL query:

```
select timeField, max(otherField) from myCollection group by timeField
```

If grouping on time is not supported, some visuals will be unavailable for the data source.

Support for this feature by connector is shown in the following table.

Key:Y - Supported; **N** - Not Supported; N/A - not applicable

Connector	Supported?
Amazon Redshift	Y
Amazon S3	Y
Apache Drill	Y
Apache Phoenix	Y
Apache Phoenix Query Server (QS)	Y
Apache Solr	Y
BigQuery	Y
Business Central Jet	Y
Cloudera Impala	Y
Cloudera Search	N
Couchbase	Y
Dremio	Y
Elasticsearch 7.0	Y
Elasticsearch 8.0	Y
File Upload	Y
HDFS	Y
Hive	Y
Jira	Y
MemSQL	Y

Connector	Supported?
Microsoft SQL Server	Y
MongoDB	Y
MySQL	Y
Oracle	Y
PostgreSQL	Y
Python	Y
Real Time Sales	Y
Salesforce	Y
SAP Hana	Y
SAP S/4HANA	Y
SAP IQ	Y
Spark SQL	Y
Snowflake	Y
Teradata	Y
TIBCO DV	Y
Trino	Y
File Upload (Upload API)	Y
Vertica	Y

Group By UNIX Time

Many connectors support grouping data by an integer field that contains times in Unix (epoch) time.

Support for this feature by connector is shown in the following table.

Key:Y - Supported; **N** - Not Supported; N/A - not applicable

Connector	Supported?
Amazon Redshift	Y
Amazon S3	Y
Apache Drill	Y
Apache Phoenix	Y
Apache Phoenix Query Server (QS)	Y
Apache Solr	Y
BigQuery	Y
Business Central Jet	Y
Cloudera Impala	Y
Cloudera Search	N
Couchbase	Y
Dremio	Y
Elasticsearch 7.0	N
Elasticsearch 8.0	N
File Upload	Y
HDFS	Y
Hive	Y
Jira	Y
MemSQL	Y
Microsoft SQL Server	Y
MongoDB	N
MySQL	Y
Oracle	Y
PostgreSQL	Y

Connector	Supported?
Python	Y
Real Time Sales	N
Salesforce	Y
SAP Hana	Y
SAP S/4HANA	N
SAP IQ	Y
Spark SQL	Y
Snowflake	Y
Teradata	Y
TIBCO DV	Y
Trino	Y
File Upload (Upload API)	Y
Vertica	Y

Histogram Floating Point Values

Many connectors support the calculations necessary for histogram visuals with non-integer values, such as floating point (32-bit) and double-precision (64-bit) floating point data types.

Support for this feature by connector is shown in the following table.

Key:Y - Supported; **N** - Not Supported; N/A - not applicable

Connector	Supported?
Amazon Redshift	**Y**
Amazon S3	**Y**
Apache Drill	**Y**
Apache Phoenix	**Y**
Apache Phoenix Query Server (QS)	**Y**
Apache Solr	**N**
BigQuery	**Y**
Business Central Jet	**Y**
Cloudera Impala	**Y**
Cloudera Search	**N**
Couchbase	**Y**
Dremio	**Y**
Elasticsearch 7.0	**Y**
Elasticsearch 8.0	**Y**
File Upload	**Y**
HDFS	**Y**
Hive	**Y**
Jira	**Y**
MemSQL	**Y**
Microsoft SQL Server	**Y**
MongoDB	**Y**
MySQL	**Y**
Oracle	**Y**

Connector	Supported?
PostgreSQL	Y
Python	Y
Real Time Sales	Y
Salesforce	Y
SAP Hana	Y
SAP S/4HANA	Y
SAP IQ	Y
Spark SQL	Y
Snowflake	Y
Teradata	Y
TIBCO DV	Y
Trino	Y
File Upload (Upload API)	Y
Vertica	Y

Last Value

The last value metric in data is the last value in all the data values for a field, sorted by the time attribute selected for the time bar. If the latest date and time for the time attribute is exactly the same in multiple records, the last value for the field is the maximum value of the field in the records with the latest date and time.

When a connector supports the last value feature, it determines and uses the last value of a selected field in the data.

Although many data stores implement a last value function, the Composer last value indicates that the last value in a given field collection can be loaded and used in a visual immediately.

Support for this feature by connector is shown in the following table.

Key:Y - Supported; **N** - Not Supported; N/A - not applicable

Connector	Supported?
Amazon Redshift	**Y**
Amazon S3	**Y**
Apache Drill	**Y**
Apache Phoenix	**N**
Apache Phoenix Query Server (QS)	**N**
Apache Solr	**N**
BigQuery	**Y**
Business Central Jet	**Y**
Cloudera Impala	**Y**
Cloudera Search	**N**
Couchbase	**N**
Dremio	**Y**
Elasticsearch 7.0	**Y**
Elasticsearch 8.0	**Y**
File Upload	**Y**
HDFS	**Y**
Hive	**Y**
Jira	**N**
MemSQL	**Y**
Microsoft SQL Server	**Y**

Connector	Supported?
MongoDB	N
MySQL	Y
Oracle	Y
PostgreSQL	Y
Python	Y
Real Time Sales	Y
Salesforce	N
SAP Hana	Y
SAP S/HANA	Y
SAP IQ	Y
Spark SQL	Y
Snowflake	Y
Teradata	Y
TIBCO DV	Y
Trino	Y
File Upload (Upload API)	Y
Vertica	Y

Multivalued Fields

Some connectors support aggregation by multivalued fields, such as arrays.

Support for this feature by connector is shown in the following table.

Key:Y - Supported; **N** - Not Supported; N/A - not applicable

Connector	Supported?	Notes
Amazon Redshift	N/A	
Amazon S3	N/A	
Apache Drill	N/A	
Apache Phoenix	N/A	
Apache Phoenix Query Server (QS)	N/A	
Apache Solr	**Y**	The Apache Solr JSON API does not support metrics by multivalued fields.
BigQuery	N/A	If you need to access a BigQuery partition, explicitly include an alias for the built in partition column in your select clause, such as `select *, _PARTITIONTIME as pt from projectId.datasetId.tableId`.
Business Central Jet	N/A	
Cloudera Impala	N/A	
Cloudera Search	**Y**	
Couchbase	N/A	The Couchbase connector supports multivalued fields with some limitations. See the detailed description in Manage The Couchbase Connector.
Dremio	N/A	
Elasticsearch 7.0	**Y**	
Elasticsearch 8.0	**Y**	
File Upload	N/A	
HDFS	N/A	
Hive	N/A	
Jira	**N**	
MemSQL	N/A	
Microsoft SQL Server	N/A	
MongoDB	**Y**	Mongo DB unwinds multivalued fields which may result in incorrect metrics' results.

Connector	Supported?	Notes
MySQL	N/A	
Oracle	N/A	
PostgreSQL	N/A	
Python	**N**	
Real Time Sales	N/A	
Salesforce	**N**	
SAP Hana	N/A	
SAP S/4HANA	N/A	
SAP IQ	N/A	
Spark SQL	N/A	
Snowflake	N/A	
Teradata	N/A	
TIBCO DV	N/A	
Trino	N/A	
File Upload (Upload API)	N/A	
Vertica	N/A	

Partitions

Some connectors support partitions and pruning. This feature enables the Partition column on the Fields tab of the data source configuration, when the data source uses a supporting connector. Partitioning allows you to link a partitioned field to another field to help improve the performance of filtering operations for a data source.

Although many data stores support partitions in some form, this feature specifically tells Composer that the partitions may be used for manual pruning of result sets to increase speed.

Support for this feature by connector is shown in the following table.

Key:Y - Supported; **N** - Not Supported; N/A - not applicable

Connector	Supported?
Amazon Redshift	N/A
Amazon S3	N/A
Apache Drill	**Y**
Apache Phoenix	N/A
Apache Phoenix Query Server (QS)	N/A
Apache Solr	N/A
BigQuery	N/A
Business Central Jet	**N**
Cloudera Impala	**Y**
Cloudera Search	N/A
Couchbase	**N**
Dremio	N/A
Elasticsearch 7.0	N/A
Elasticsearch 8.0	N/A
File Upload	**N**
HDFS	N/A
Hive	**Y**
Jira	**N**
MemSQL	N/A
Microsoft SQL Server	**N**

Connector	Supported?
MongoDB	N/A
MySQL	**N**
Oracle	**N**
PostgreSQL	**N**
Python	**N**
Real Time Sales	N/A
Salesforce	**N**
SAP Hana	**N**
SAP S/4HANA	**N**
SAP IQ	**N**
Spark SQL	**Y**
Snowflake	**N**
Teradata	**N**
TIBCO DV	N/A
Trino	**N**
File Upload (Upload API)	**N**
Vertica	**N**

Schemas

When a connector supports schemas, it supports namespace, schema, or catalog notation for organizing collections. When schemas are supported, the Source Creation tab of the data source configuration displays a Schema drop-down you can use to select a schema for the data source configuration. Elasticsearch has a custom UI for displaying multiple indices.

Visualize the relationships in your schemas in supported data sources, and connections. Add more relationships in connections as needed. See Visualize Schemas and Joins

If you'd like to make a default schema available to your users, or hide a schema from your users, update the properties file of your connector. See Select Schemas .

Select Schemas

Control which data source schemas are treated as internal by Composer and are not disclosed to users during source creation. Supported schemas are included in the Schema drop-down selector in the Source Creation tab of the data source configuration.

Add or edit the `system.schemas` property to the properties file of supported connectors to specify the schemas to use. Restart the connector after making these changes. The schemas you want to make available to users are visible in the Schema drop-down.

Action	Update the properties file by:
Hide a data source schema from users	List the value of all default schemas, and add the value of the schema you want to hide to the `system.schemas` property. This hides all default schemas and the additional schema.
Show a data source schema that is hidden by default to users	List the value of all default schemas in the `system.schemas` property, except for the default schema you want to include for users. This hides all listed schemas, and displays the omitted default schema in the Schema drop-down.

Support for this feature by connector is shown in the following table.

Key:Y - Supported; **N** - Not Supported; N/A - not applicable

Connector	Supported?
Amazon Redshift	**Y**
Amazon S3	**Y**
Apache Drill	**Y**
Apache Phoenix	**Y**
Apache Phoenix Query Server (QS)	**Y**

Connector	Supported?
Apache Solr	N/A
BigQuery	Y
Business Central Jet	Y
Cloudera Impala	Y
Cloudera Search	N/A
Couchbase	Y
Dremio	Y
Elasticsearch 7.0	N/A
Elasticsearch 8.0	N/A
File Upload	Y
HDFS	Y
Hive	Y
Jira	Y
MemSQL	Y
Microsoft SQL Server	Y
MongoDB	Y
MySQL	Y
Oracle	Y
PostgreSQL	Y
Python	N
Real Time Sales	Y
Salesforce	Y
SAP Hana	Y
SAP S/4HANA	Y
SAP IQ	Y
Spark SQL	Y
Snowflake	Y
Teradata	Y
TIBCO DV	Y

Connector	Supported?
Trino	Y
File Upload (Upload API)	Y
Vertica	Y

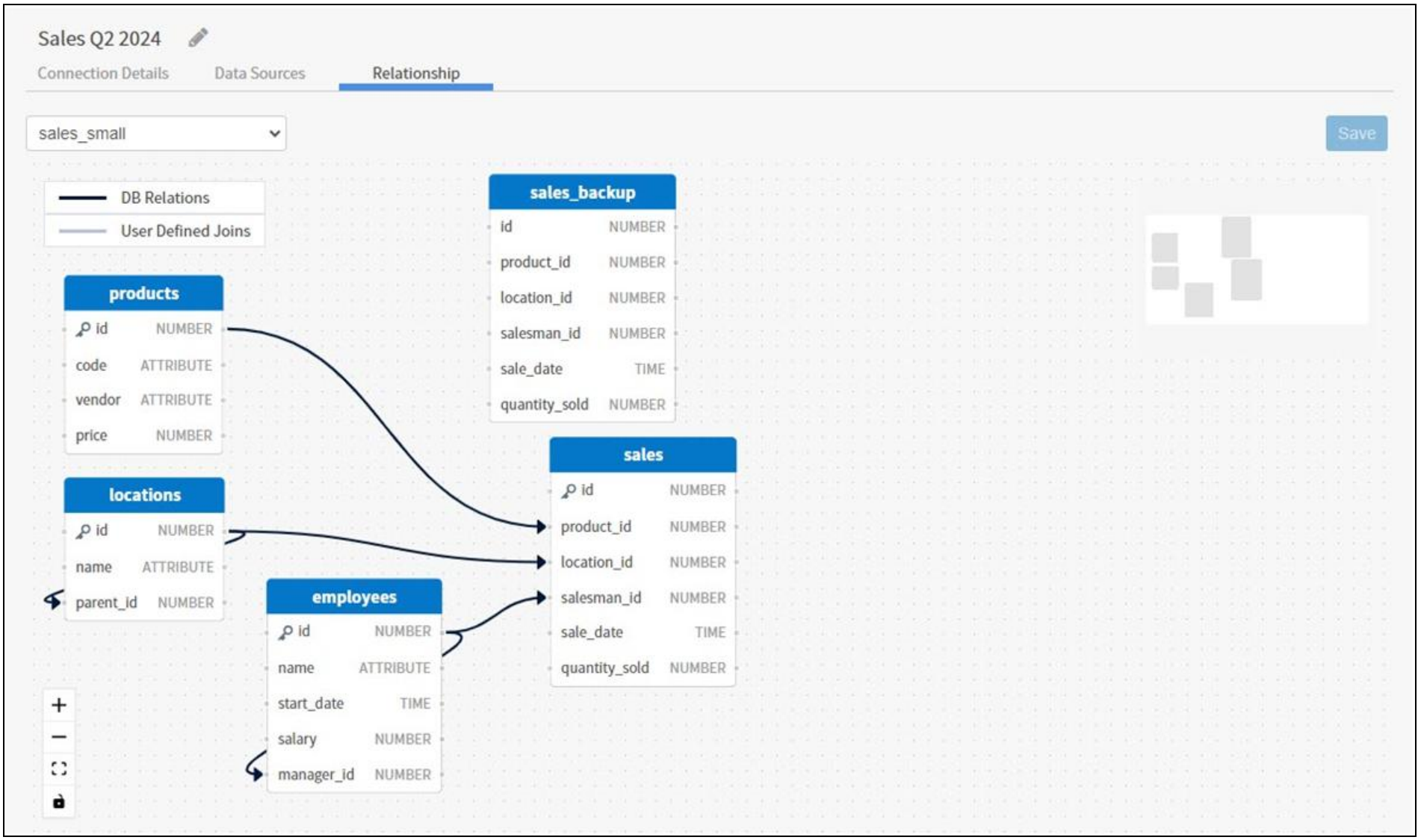
Visualize Schemas and Joins

Visualize the fields and tables in schemas included with your connection, and created by other users. Add relationships as needed. Visualize joins in your data sources.

When you create joins in a source, you can visualize and update your joins as well. See Fuse Data Sources.

Edit and View Relationships in Schemas

Users with appropriate permissions can view and edit the relationships using the Relationship tab for a supported connection. Select a schema from the list in the drop-down, then make and save any additions to the relationships as needed.



View or edit a schema for a connection

1. Open a supported connection and navigate to the **Relationship** tab. A work area opens you can use to navigate among the tables and relationships of a selected schema.

2. Select an available schema from the drop-down list. Larger data sets may take a few moments to load.
3. View the existing relationships and any user defined relationships.
4. Optionally, draw more relationships in this work area, then **Save** your changes.

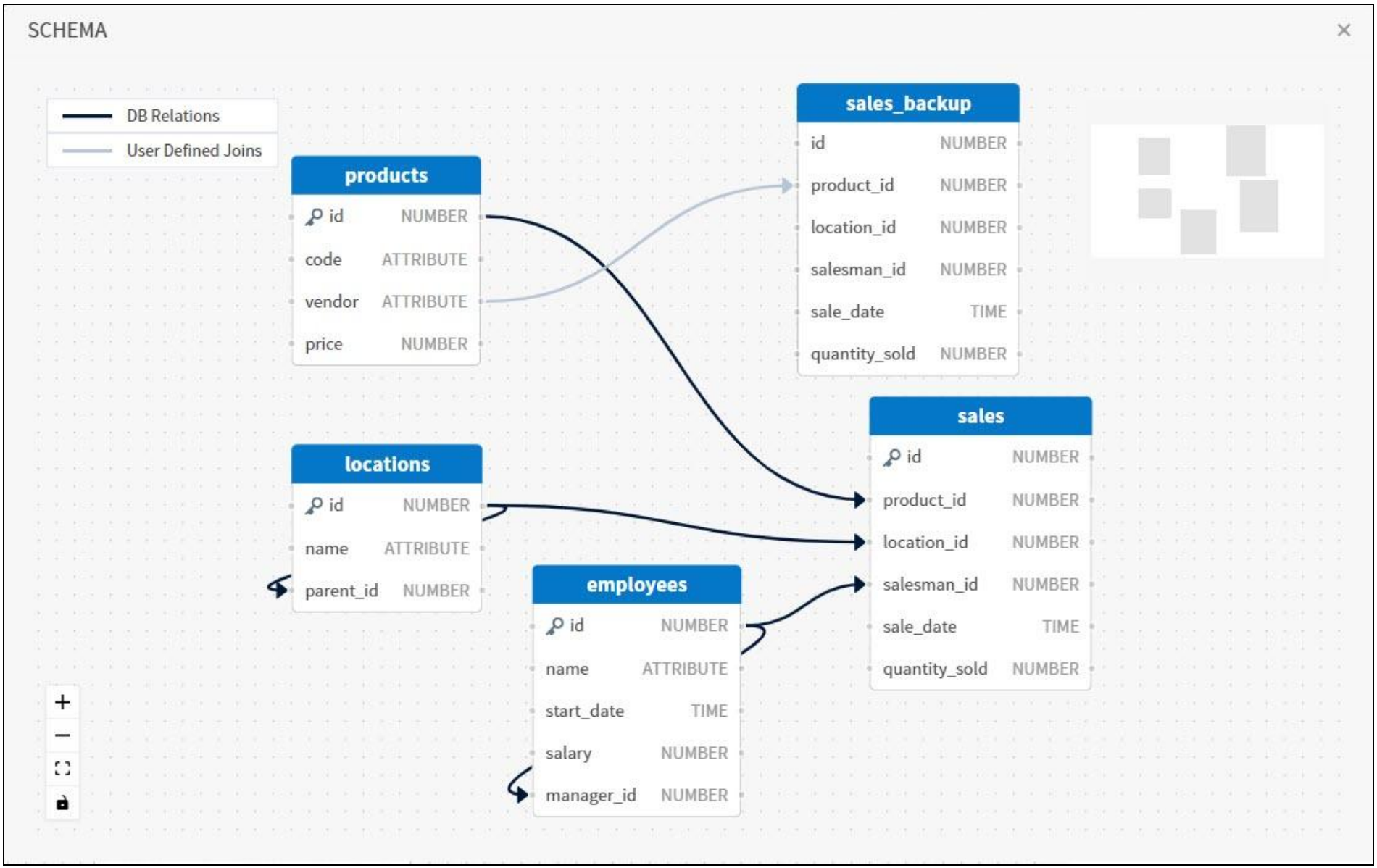
   Note: To remove a user-applied join, double-click to select the join, then select the backspace key. The relationship is removed.
5. View or edit as many schemas as you need.

Important: If you add joins that create one-to-many relationships here, Composer may return an error that prevents use of the data in a visual. For best results, when you create a one-to-many relationship with a specific left entity, any additional joins must refer to that table as the right entity. See Recommended Joins.

Note: For Postgres connections, both tables and data relationship information are read from the connection. Other supported connections include table information but do not read relationship information. See Connector Support for Schema Visualization. No information is provided for unsupported connections.

View Relationships of a Schema in a Source

You can view the relationships present in schemas you're using in your sources that use a supported connection. The Source Creation tab of the source includes a Schema drop-down you can use to select a schema for the source, with a view button that opens the work area.



View a schema in a source

1. Create or edit a source that uses a schema.
2. Select the view icon next to your selected Schema. A schema work area opens you can use to view the tables and relationships of that schema. Larger data sets may take a few moments to load.
3. View the existing relationships and any user defined joins to better understand the relationships present.
4. Close the schema or joins modal when you're done viewing the schema information, and repeat for other schemas if needed.

Connector Support for Schema Visualization

Support for this feature by connector is shown in the following table.

Key:Y - Supported; **N** - Not Supported; N/A - may not be supported.

Connector	Supported?
Amazon Redshift	**Y**
Amazon S3	**Y**
Apache Drill	N/A
Apache Phoenix	N/A
Apache Phoenix Query Server (QS)	N/A
Apache Solr	N/A
BigQuery	N/A
Cloudera Impala	**Y**
Cloudera Search	N/A
Couchbase	N/A
Dremio	N/A
Elasticsearch 7.0	**N**
Elasticsearch 8.0	N/A
File Upload	N/A
HDFS	**Y**
Hive	N/A
Jira	**Y**

Connector	Supported?
MemSQL	Y
Microsoft SQL Server	Y
MongoDB	Y
MySQL	Y
Oracle	Y
PostgreSQL	Y
Python	N/A
Real Time Sales	Y
Salesforce	Y
SAP Hana	N/A
SAP S/4HANA	N/A
SAP IQ	N/A
Spark SQL	N/A
Snowflake	Y
Teradata	N/A
TIBCO DV	N/A
Trino	N/A
File Upload (Upload API)	N/A
Vertica	N/A

Text Search

When a connector supports text searches, it can perform an efficient search on text fields. When text searches are supported, search control is enabled on dashboards using data sources that use the connector. To turn on text search, **Enable Text Search** in the Global Settings tab for your sources.

Support for this feature by connector is shown in the following table.

Key:Y - Supported; **N** - Not Supported; N/A - not applicable

Connector	Supported?
Amazon Redshift	N/A
Amazon S3	N/A
Apache Drill	N/A
Apache Phoenix	N/A
Apache Phoenix Query Server (QS)	N/A
Apache Solr	**Y**
BigQuery	N/A
Business Central Jet	N/A
Cloudera Impala	N/A
Cloudera Search	**Y**
Couchbase	N/A
Dremio	N/A
Elasticsearch 7.0	**Y**
Elasticsearch 8.0	**Y**
File Upload	N/A
HDFS	N/A
Hive	N/A
Jira	**N**
MemSQL	N/A
Microsoft SQL Server	N/A
MongoDB	N/A
MySQL	N/A
Oracle	N/A

Connector	Supported?
PostgreSQL	N/A
Python	**N**
Real Time Sales	N/A
Salesforce	**N**
SAP Hana	N/A
SAP S/4HANA	N/A
SAP IQ	N/A
Spark SQL	N/A
Snowflake	N/A
Teradata	N/A
TIBCO DV	N/A
Trino	N/A
File Upload (Upload API)	N/A
Vertica	N/A

TLS

Many connectors support SSL/TLS encryption.

Support for this feature by connector is shown in the following table.

Key:Y - Supported; **N** - Not Supported; N/A - not applicable

Connector	Supported?
Amazon Redshift	Y
Amazon S3	N
Apache Drill	Y
Apache Phoenix	Y
Apache Phoenix Query Server (QS)	Y
Apache Solr	N
BigQuery	Y
Business Central Jet	Y
Cloudera Impala	Y
Cloudera Search	Y
Couchbase	Y
Dremio	N
Elasticsearch 7.0	Y
Elasticsearch 8.0	Y
File Upload	Y
HDFS	N
Hive	Y
Jira	N
MemSQL	Y
Microsoft SQL Server	Y
MongoDB	Y
MySQL	Y
Oracle	Y
PostgreSQL	Y

Connector	Supported?
Python	**N**
Real Time Sales	N/A
Salesforce	**N**
SAP Hana	**Y**
SAP S4HANA	**Y**
SAP IQ	**Y**
Spark SQL	**N**
Snowflake	**Y**
Teradata	**Y**
TIBCO DV	**N**
Trino	**Y**
File Upload (Upload API)	**Y**
Vertica	**Y**

Timezone Conversion for Users

Displaying the source data in dashboards and visualizations in the timezone of individual users instead of the default timezone stored at the source. Additionally, you can convert a `TIME` field to a custom timezone.

Note: If you are upgrading from an earlier version of Composer, this may be a breaking change: the introduction of the system attribute `User.timeZone` may cause a conflict if you used this as a custom attribute. See Upgrade Workflow.

The functionality is available in the following data sources and for the data stored in the UTC timezone:

- MS SQL
- Snowflake
- MongoDB
- BigQuery
- Hive
- SparkSQL
- Impala
- PostgreSQL
- Redshift

Enable `TIME` Conversion To User Timezones

Before you convert a `TIME` field for use by users, define their timezone in user regional settings. Next, convert the `TIME` field in the source.

Define a User's Timezone

1. Log in as an administrator or a user who has been assigned to a group with user management privileges.

   If the user name you log in with is also associated with other Composer accounts, verify that the correct account is selected. See Switch Tenants.

2. Select **Users and Groups** on the UI menu (). The Users and Groups page work area opens.

3. Select **Users** to see a list of all the user definitions that have been defined for the account.

4. Select a user, then select the **Regional Settings** tab.

5. Select the **Time Zone** for the user from the options available in the drop-down selector.

6. Select **Save** to save the user definition.

Convert a `TIME` Field of a Source

When you convert field, your software creates a derived field that includes the `User.timeZone` system attribute used as an interpolated value, for example `to_timezone(TIME_field, '${User.timeZone|UTC}'))`. Use the created derived field in dashboards and visualizations. The data will be recalculated using the account of each individual user with the custom timezone.

1. Log in as a user with the **Administer Sources** privilege, or a user with **read** and **write** permission for the data source.

2. Select **Sources** on the UI menu (). The Sources page appears.

3. Select a source to open it, then select a time field in the **Fields** tab.

4. Select the **Settings**sidebar menu, then select **Convert** and the **Convert to User Timezone** option to convert the data type. A field conversion modal window opens.

5. In the Time to Time Zone Conversion work area, define a **Label** for the newly created field, then select **User Time Zone** for the Time Zone field if not already selected.

6. Select **Save** to create the new field.

Alternative: Convert a Timezone Once Using a Function

You can convert a field with the `TIME` data type into a selected timezone manually using the `to_timezone` function. Specify the function as shown below to return static timezone conversion.

Syntax	Example	Use Case
`to_timezone(TIME_field, 'IANA timezone identifier')`	`to_timezone(TIME_field, 'Europe/Kyiv')`	Converts a field `TIME_field` from its UTC stored timezone to the selected timezone, `Europe/Kyiv`.

Important: Conversion is available only for `TIME` fields stored in the UTC timezone at the data source. Conversions performed on the data stored in a custom timezone may be inaccurate.

Upgrade Workflow

If you are upgrading from an earlier version of Composer, this may be a breaking change: the introduction of the system attribute `User.timeZone` may cause a conflict if you used this as a custom attribute.

When you upgrade to the latest version of Composer this feature triggers the following changes:

- Custom user attributes you manually created with the name the `User.timeZone` in earlier Composer versions (23.2 and earlier) are automatically converted to the system attributes if their value corresponds to the IANA timezone standard ISO 8601 (for example, `'Europe/Kyiv', 'UTC+5'`).
- All custom user attributes `User.timeZone` that do not correspond to the IANA timezone standard are removed.

To ensure a smooth upgrade process, select an upgrade workflow ahead of updating Composer depending on your needs:

- If you want to start using your custom attribute `User.timeZone` for timezone conversion purposes, the attribute will be automatically changed to the system value at upgrade. To ensure this takes place, verify before upgrade that the values of the attribute are provided in IANA format before you upgrade Composer.
- If you want to preserve your custom attribute for other purposes, we recommend renaming the attribute before you upgrade Composer. For example, change `User.timeZone` to `User.timeZone_custom`. Don't change the value of the attributes before running the upgrade script.

API Changes

The APIs in `/api/users` has been expanded to include the `"timeZone": "string"` parameter. This displays the user's timezone formatted as an IANA timzeone identifier, ISO 8601 (for example, "Europe/Kyiv"). The default value is `UTC`.

Important: `User.timeZone` is now a reserved system attribute to support this feature. See Upgrade Workflow for alternative approaches.

Payload Changes

Composer v23.2 and earlier:

```
{
  "id": "string",
  "fullname": "string",
  "email": "string",
  "accountId": "string",
  "name": "string",
  "password": "string",
  "localeSettingsId": "string",
  "languageLocaleId": "string"
}
```

Composer v23.3 and later:

```
{
  "id": "string",
  "fullname": "string",
  "email": "string",
  "accountId": "string",
  "name": "string",
  "password": "string",
  "localeSettingsId": "string",
  "languageLocaleId": "string",
  "timeZone": "Europe/Kyiv"
}
```

Wildcard Case-Insensitive Filters

Many connectors support case-insensitive wildcard filters.

Support for this feature by connector is shown in the following table.

Key:Y - Supported; **N** - Not Supported; N/A - not applicable

Connector	Supported?	Notes
Amazon Redshift	**Y**	
Amazon S3	**Y**	
Apache Drill	**Y**	
Apache Phoenix	**Y**	
Apache Phoenix Query Server (QS)	**Y**	
Apache Solr	**N**	Apache Solr connectors support wildcard filters, but case-sensitivity cannot be enforced. Consequently, neither case-sensitive or case-insensitive wildcard filters are supported.
BigQuery	**Y**	If you need to access a BigQuery partition, explicitly include an alias for the built in partition column in your select clause, such as `select *, _PARTITIONTIME as pt from projectId.datasetId.tableId`.
Business Central Jet	**Y**	
Cloudera Impala	**Y**	
Cloudera Search	**N**	Cloudera Search connectors support wildcard filters, but case-sensitivity cannot be enforced. Consequently, neither case-sensitive or case-insensitive wildcard filters are supported.
Couchbase	**Y**	
Dremio	**Y**	
Elasticsearch 7.0	**N**	Elasticsearch connectors support wildcard filters, but case-sensitivity cannot be enforced. Consequently, neither case-sensitive or case-insensitive wildcard filters are supported.
Elasticsearch 8.0	**N**	
File Upload	**Y**	
HDFS	**Y**	
Hive	**Y**	
Jira	**Y**	
MemSQL	**Y**	
Microsoft SQL Server	**Y**	

Connector	Supported?	Notes
MongoDB	Y	
MySQL	Y	
Oracle	Y	
PostgreSQL	Y	
Python	Y	
Real Time Sales	Y	
Salesforce	Y	
SAP Hana	Y	
	N	
SAP IQ	Y	
Spark SQL	Y	
Snowflake	Y	
Teradata	Y	
TIBCO DV	Y	
Trino	Y	
File Upload (Upload API)	Y	
Vertica	Y	

Wildcard Case-Sensitive Filters

Many connectors support case-sensitive wildcard filters.

Support for this feature by connector is shown in the following table.

Key:Y - Supported; **N** - Not Supported; N/A - not applicable

Connector	Supported?	Notes
Amazon Redshift	**Y**	
Amazon S3	**Y**	
Apache Drill	**Y**	
Apache Phoenix	**Y**	
Apache Phoenix Query Server (QS)	**Y**	
Apache Solr	**N**	Apache Solr connectors support wildcard filters, but case-sensitivity cannot be enforced. Consequently, neither case-sensitive or case-insensitive wildcard filters are supported.
BigQuery	**Y**	If you need to access a BigQuery partition, explicitly include an alias for the built in partition column in your select clause, such as `select *, _PARTITIONTIME as pt from projectId.datasetId.tableId`.
Business Central Jet	**Y**	
Cloudera Impala	**Y**	
Cloudera Search	**N**	Cloudera Search connectors support wildcard filters, but case-sensitivity cannot be enforced. Consequently, neither case-sensitive or case-insensitive wildcard filters are supported.
Couchbase	**Y**	
Dremio	**N**	
Elasticsearch 7.0	**N**	Elasticsearch connectors support wildcard filters, but case-sensitivity cannot be enforced. Consequently, neither case-sensitive or case-insensitive wildcard filters are supported.
Elasticsearch 8.0	**N**	
File Upload	**Y**	
HDFS	**Y**	
Hive	**Y**	
Jira	**Y**	
MemSQL	**N**	
Microsoft SQL Server	**N**	

Connector	Supported?	Notes
MongoDB	Y	
MySQL	N	
Oracle	Y	
PostgreSQL	Y	
Python	Y	
Real Time Sales	Y	
Salesforce	Y	
SAP Hana	Y	
SAP S/4HANA	Y	
SAP IQ	Y	
Spark SQL	Y	
Snowflake	Y	
Teradata	Y	
TIBCO DV	N	
Trino	Y	
File Upload (Upload API)	Y	
Vertica	Y	

Manage Data Sources

Composer can connect to a wide array of data stores–from modern databases such as Hadoop, Search, Streaming, and NoSQL, to traditional stores like SQL-based stores.

To use the data from a data store in a Composer visual, you must first create connection to it and then create a data source configuration for it. The following topics describe how to maintain your data source configurations.

Note: Composer supports only underscores and dashes in data store field names. No other special characters or white space are supported. If your data store uses special characters other than underscores and dashes in field names, please remove them before attempting to create a data source configuration.

Note: You cannot save a data source with an invisible field, if the invisible field is referenced in a visual (charted or used in filters and keysets).

Data source configurations are managed from the Sources page.

- Data Sources Page
- Search And Filter Lists
- About Source Permissions
- Define A Source
- Edit A Data Source
- Clear The Cache For A Data Source Configuration
- Import Or Export Sources
- Restrict Access To Data Using Row Security
- Restrict Access To Fields Using Column Security
- Delete A Data Source Configuration

For information about how Composer caches data, see How Composer Caches Data. For information on ways you can manipulate the data received from your data store, see Manipulate Data In The ComposerSymphony Data Store.

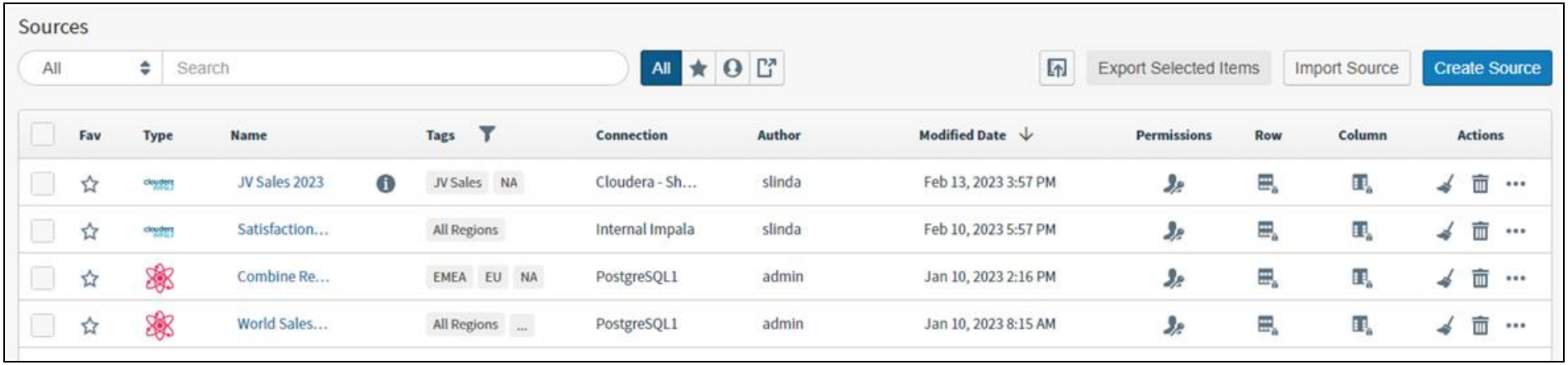
For information on configuring the time bar or search bar defaults, including the default refresh rates, for your data source, see Configure Time Bar Defaults and Configure Search Box Defaults.

Data Sources Page

Use **Sources** to create, import, export, search, review, and maintain your data sources.

All users can view the **Sources** work area.

- If you log in as a user who has not been granted permissions for any data sources, the **Sources** work area displays a message indicating that no sources are available.
- If you log in as a user who only has **read** permissions for one or more data sources, they are shown here, but Connection information is not available for the sources.
- If you log in as a user who belongs to a group that is granted the **Create New Data Sources** or **Administer Sources** privilege, you can create, import, export, and maintain data source configurations using this work area.



The Data Sources work area includes the following features:

1. **Create Source**: Select to create a new data source configuration. See Define A Source.
2. **Import Source**: Select to import a new data source configuration and connection information. See Import Or Export Sources.
3. **Export Selected Items**: Export one or more data sources by selecting the checkbox for a source to export. The **Export Selected Items** button becomes active. Select to download the sources in JSON format.

4. **Embed Sources Inventory**: Select  to generate a code snippet to embed the sources inventory in your application. See Generate A Sources Inventory HTML Snippet.

5. A table that lists the data sources you can see, sort, and favorite.

6. Options to modify source permissions, view and modify row and column security filters, refresh the cache, and manage Available Visual Types and Materialized Views.

7. A search bar at the top of the page you can use to search for a specific data source in the table.

Access the Data Sources work area

1. Log into Composer.

2. Select **Sources** on the UI menu () or the top-level navigation menu, or select the **Sources** box on the Home page. The Data Sources work area appears.

Depending on your settings, you can edit and delete data sources listed in the table. You can also clear the cache for a data source. If there are many sources listed, you may need to search for the source you need.

Search Field

You can use the search field to filter the sources in this work area by source Name, Description (if provided), Connection, or Author. For example, if you type a **C** in the search box, only sources that include the letter *C* in the selected field searched are shown in the working area. See Search And Filter Lists.

Buttons

The buttons on the page allow easy access to saved data sources, as well as other data sources created by other users in your Composer environment that you have been granted access to see. Use the options shown to create new sources, filter the data sources shown, as well as import, export, or create sources.

Button	Description
All	Removes any filters for the sources list and displays all sources available to you within your environment.
	Displays only the sources that you have marked as favorites.
	Displays only the sources that you created and saved. Sources created and saved by other users are hidden.

Button	Description
	Displays only the sources that other users shared with you.
	Select to generate an embeddable sources inventory link. See Generate A Sources Inventory HTML Snippet.
Export Selected Items	Allows you to export multiple selected items.
Import Source	Allows you to import a source. See Import Or Export Sources.
Create Source	Allows you to create a new source. See Define A Source.

The Sources List

The sources list columns are described below. Several of these columns can be used to sort the list: select the column header to sort first to last and again to sort last to first. You can search for items by the contents of several columns. See Search and Filter Lists.

Column	Description
Select (not labeled)	Select one or more items to perform bulk actions, such as export, for your resources.
Fav	Mark the source as a favorite.
Type	An icon identifying the data store type for the data source.
Name	The name assigned during data source creation.
Description (not labeled)	The description icon is visible if a description associated with a source. You can search for a source by the contents of this field.
Tags	Content tags applied to the source. Select the filter icon to open a drop down list and select tags to filter your list or to narrow your search results. If several tags are associated with an item, hover over the ellipsis to see all tags for this resource.
Filter icon	Select to filter the work area's contents by one or more content tags.
Connection	The display name of the data store connection definition connected to this data source. For flat files, you define a Display Name when you upload the file; this name is shown.
Author	The user name of the data source creator.
Modified Date	The time stamp when the data source configuration definition was last modified.
Permissions	Select the permissions icon for a data source to assign and manage its permissions. You can only define permissions for a data source if you are logged in as a user with the **Administer Sources** privilege, or as a user with the **Manage Source Permissions**

Column	Description
	privilege.
Row	Select the row security () icon for a data source to define its row security for authorization groups. You can only define row security for a data source if you are logged in as a user with the **Administer Sources** privilege, or as a user with the **Manage Source Permissions** privilege.
Column	Select the column security () icon for a data source to define its column security for different authorization groups. You can only define row security for a data source if you are logged in as a user with the **Administer Sources** privilege, or as a user with the **Manage Source Permissions** privilege.
Actions	Shows icons you can select to perform actions for the data source. ■ Select the delete () icon to delete a data source configuration. Before you delete a data source configuration, you must delete all the dashboards and visuals that use it. See Delete A Data Source Configuration. You can only delete a data source if you are logged in as, a user with the **Administer Sources** privilege, or a user with **read** and **delete** permission for the data source. ■ Select the clear cache () icon to clear the data cache for a data source configuration. Select one or more cache options to clear: ◦ Data Cache: Clears the cached query results. ◦ Statistics Cache: Clears the cache of fields statistic metadata, such as min, max, and distinct values numbers. See How Composer Caches Data and Clear The Cache For A Data Source Configuration. You can only clear the cache for a data source if you a user with the **Administer Sources** privilege, or a user with **write** permission for the data source. **Note:** If a Custom Range has been defined for a field, the minimum and maximum fields used in filters remain unchanged when you refresh source data. These fields are shown with cache actions disabled on the Cache tab. ■ Select the More menu () button to view the More menu for a data source configuration. Options include: ◦ Available Visual Types: Select to define what visuals are available for this data source configuration. See Available Visual Types.

Column	Description
	◦ Materialized Views: Select to define materialized view settings for the data source configuration. See Use Materialized Views (Experimental). Not visible if disabled. ◦ Export Source: Select to export the source information in JSON format. See Import Or Export Sources. **Note:** If you try to delete a visual, filter snippet, dashboard, dashboard link, source, or source field, Composer displays an error message naming any objects dependent on the item you're trying to delete. You can delete the item after you've removed the association from the dependent object. See Fields Usage.

Define a Source

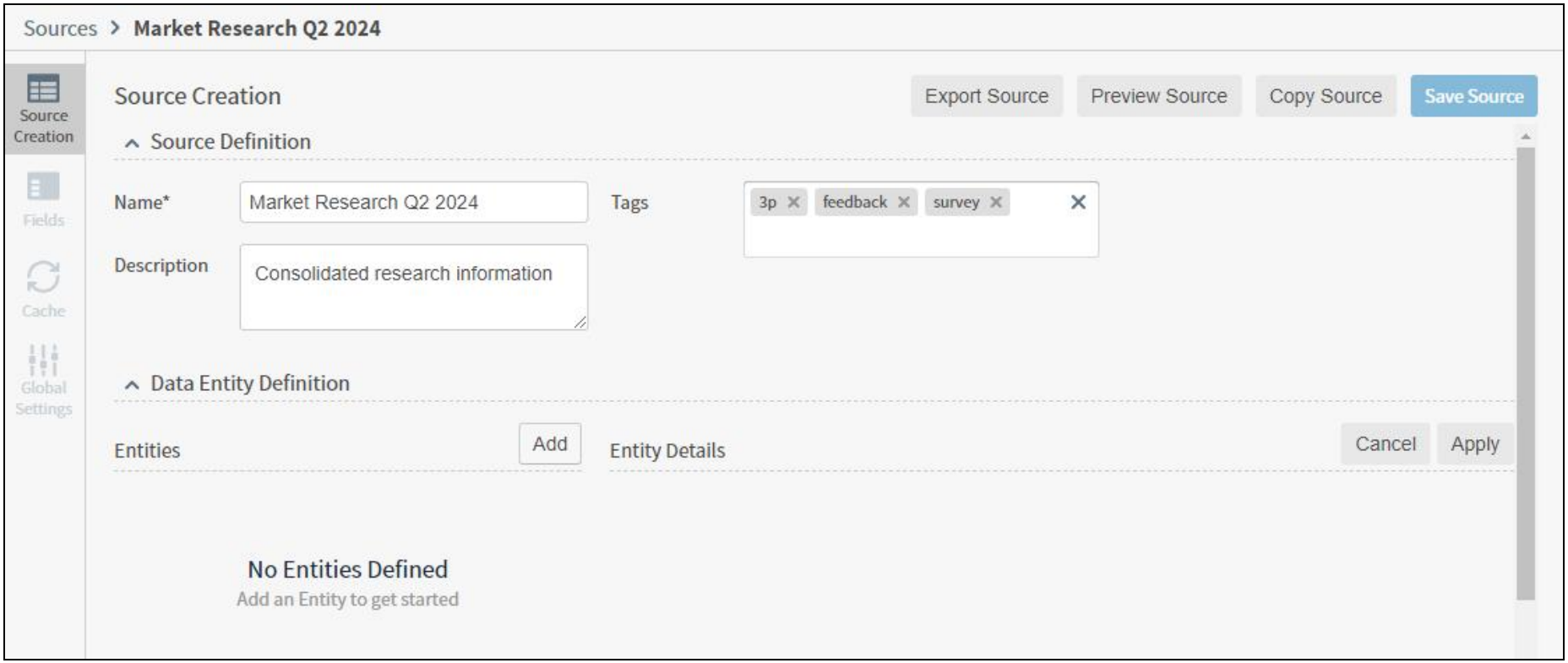
Sources define what data you and your users can access through a data connection or in an uploaded file. Use this data to create visuals and dashboards in your environment. You can create a source from an existing connection, uploaded files, or as a combination of data from multiple connections as joins or hierarchical data. Adjust the available content by selecting specific entities, associated schemas, or providing custom SQL.

- If you are creating a fusion or hierarchical source, add multiple data entities and set up a join configuration. See Create a Fusion Source, Hierarchical Fields And Structures, and Define A Hierarchical Source.
- If you are adding a file as a data entity, some options may differ. See Manage File Uploads and Data Entity Details - From File.

Define a New Source

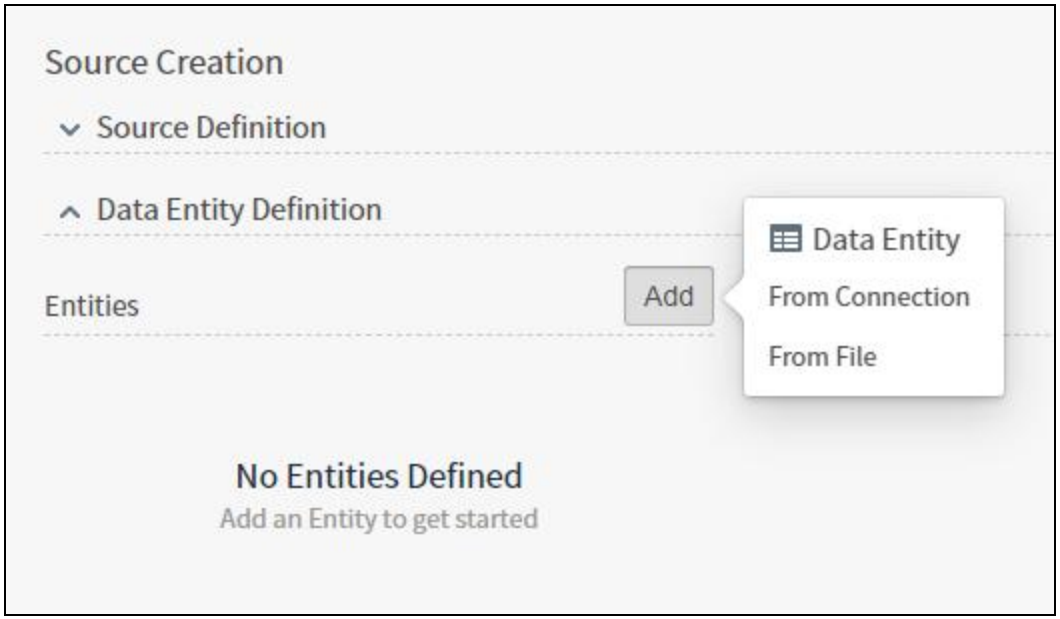
Define a new source

1. Log in as a user with the **Administer Sources** or **Create New Data Sources** privilege.
2. Select **Sources** on the UI menu ( ) or the top-level navigation menu, or select the **Sources** box on the Home page. The Sources work area appears.
3. On the Sources page, select the **Create Source** button. The Source Creation work area opens.



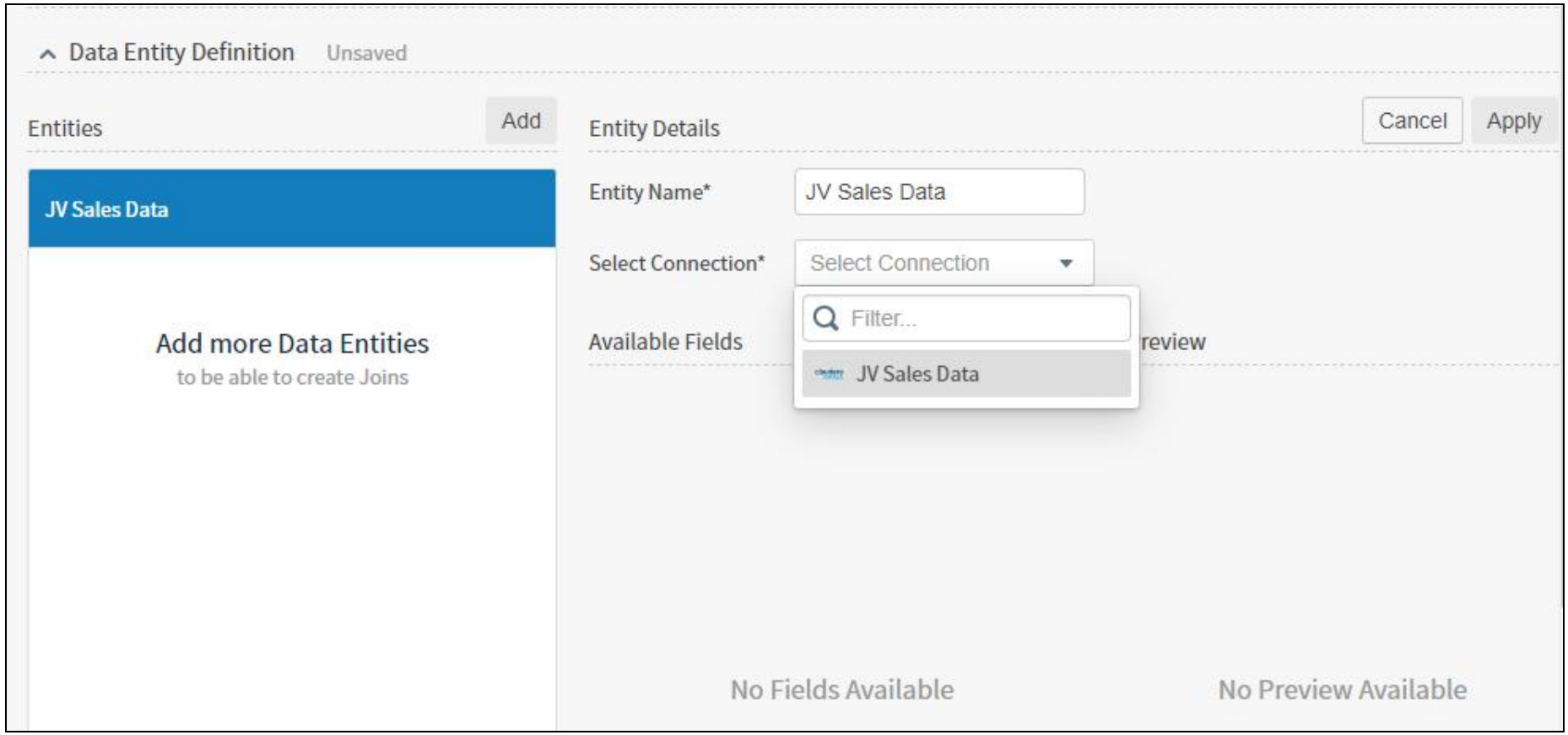
4. Enter a unique **Name** for your source and optional **Description** in the Source Definition work area. This description is searchable from the Sources page.

5. Click the **Add** button to add a data entity From Connection or From File.



6. Select **From Connection** to open the Data Entity Details work area. You will only see the connections you have read permission for. See About Source Permissions.

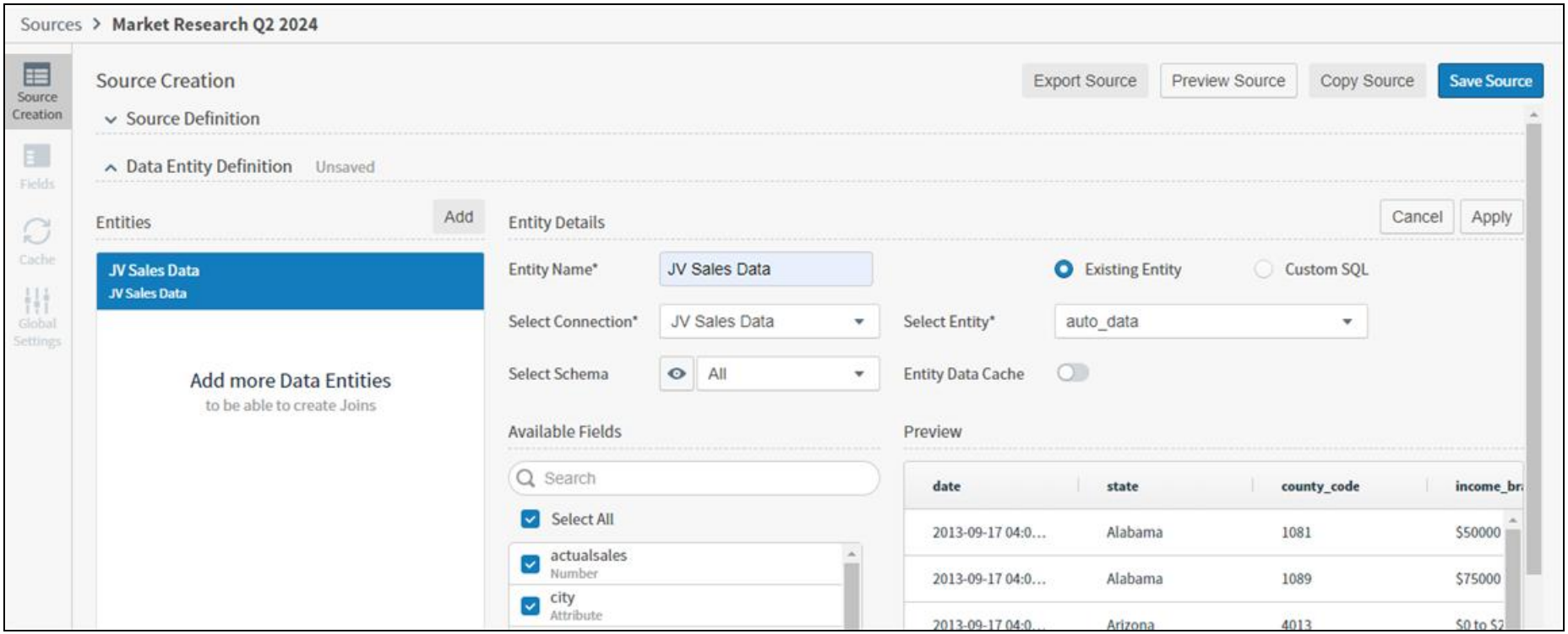
7. Enter a unique Data Entity Name, then select an available connection in the **Select Connection** list.



8. Depending on the connection you select, you'll need to select an existing entity or provide Custom SQL, then provide other information as needed to add the data entity.

   By default, all Available Fields for your source are included: clear the appropriate check box to exclude the field from the source. Select **Apply** to save the data entity or **Cancel** to discard your changes. Select **Save Source** to save this source. As needed, update the default settings on the Fields tab, Cache tab, or Global Settings tab.

   Table visuals and Detail dialogs display fields in the order they are retrieved from the source. When you create a source using custom SQL, your fields are shown in the order you specify.

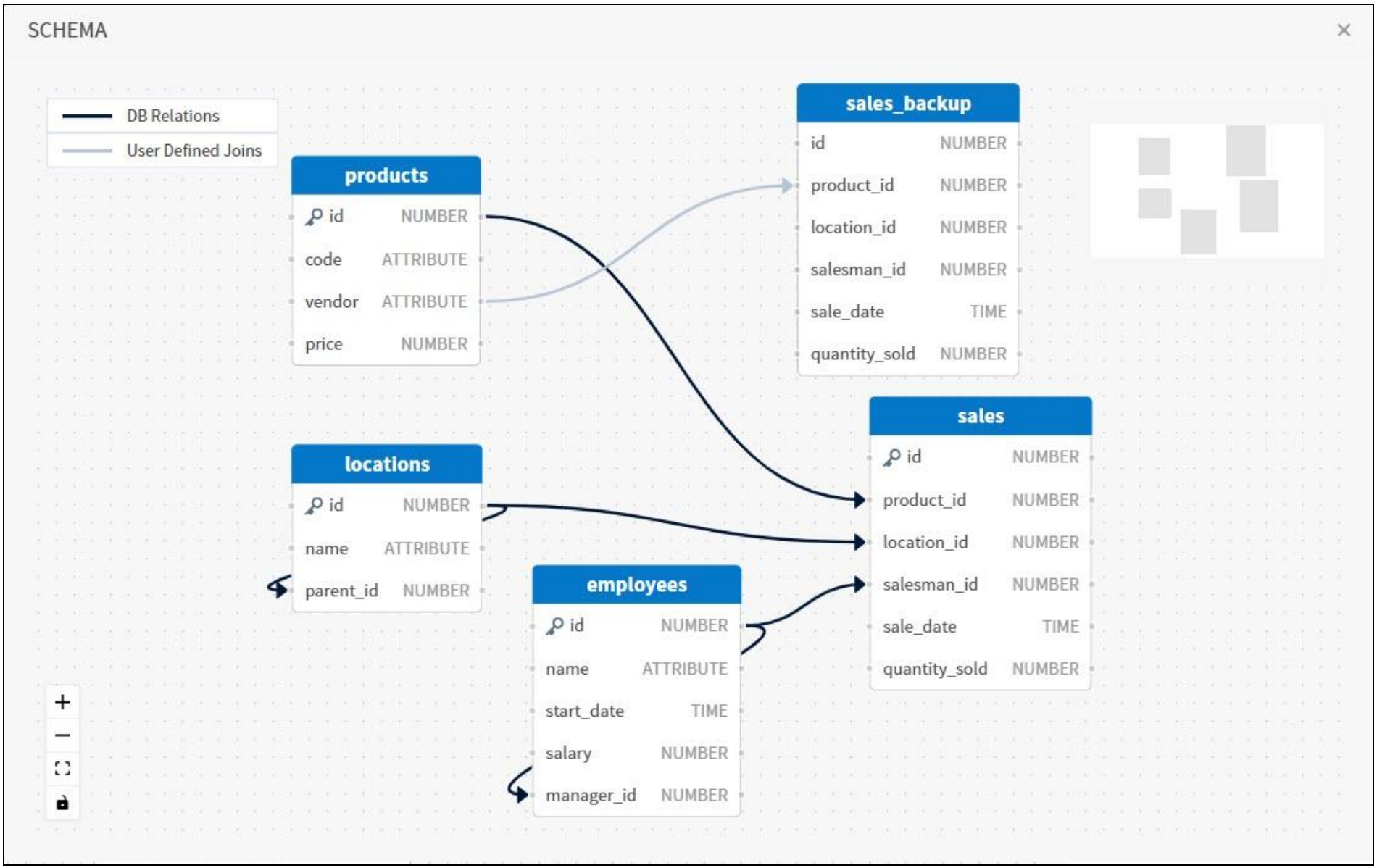


9. Once saved, your new source is added to the list on the Sources page.

View Relationships for a Schema in a Source

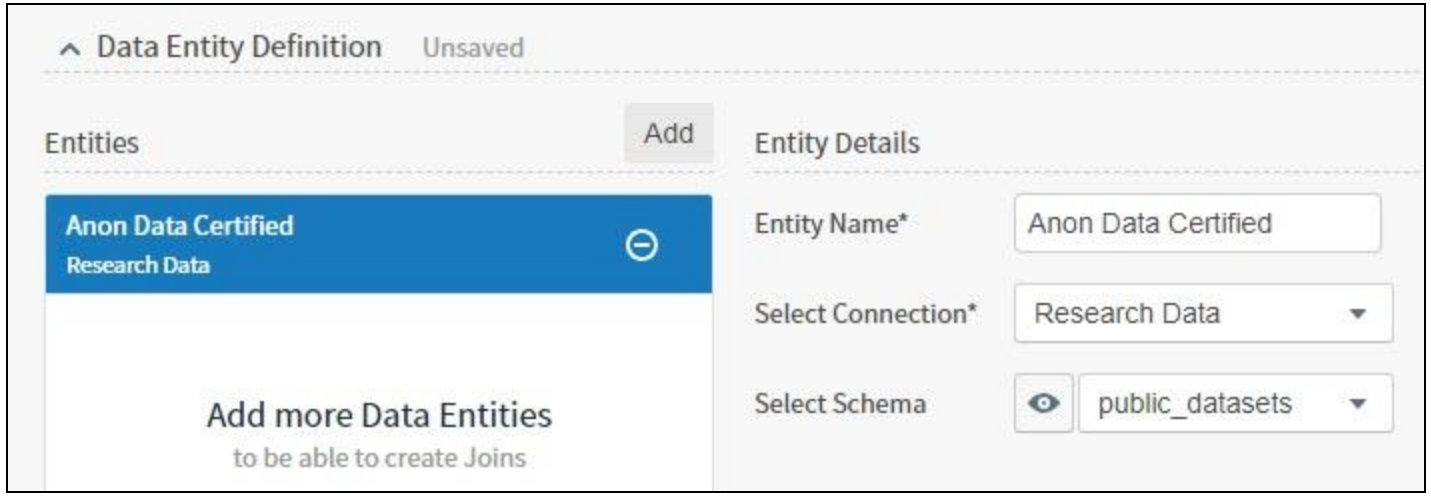
You can view the relationships for the schemas you add to your data sources to better understand the relationships present. Add a schema and select the view icon to see how the data in your tables are connected. To add more relationships to a schema, edit it in the connection directly.

Zoom in or out in this work area, or use the mini map to navigate among the various tables that make up your schema.



View a schema in a source

1. Create or edit a source that uses a schema.
2. Select the view icon next to your selected Schema. A schema work area opens you can use to view the tables and relationships of that schema. Larger data sets may take a few moments to load.

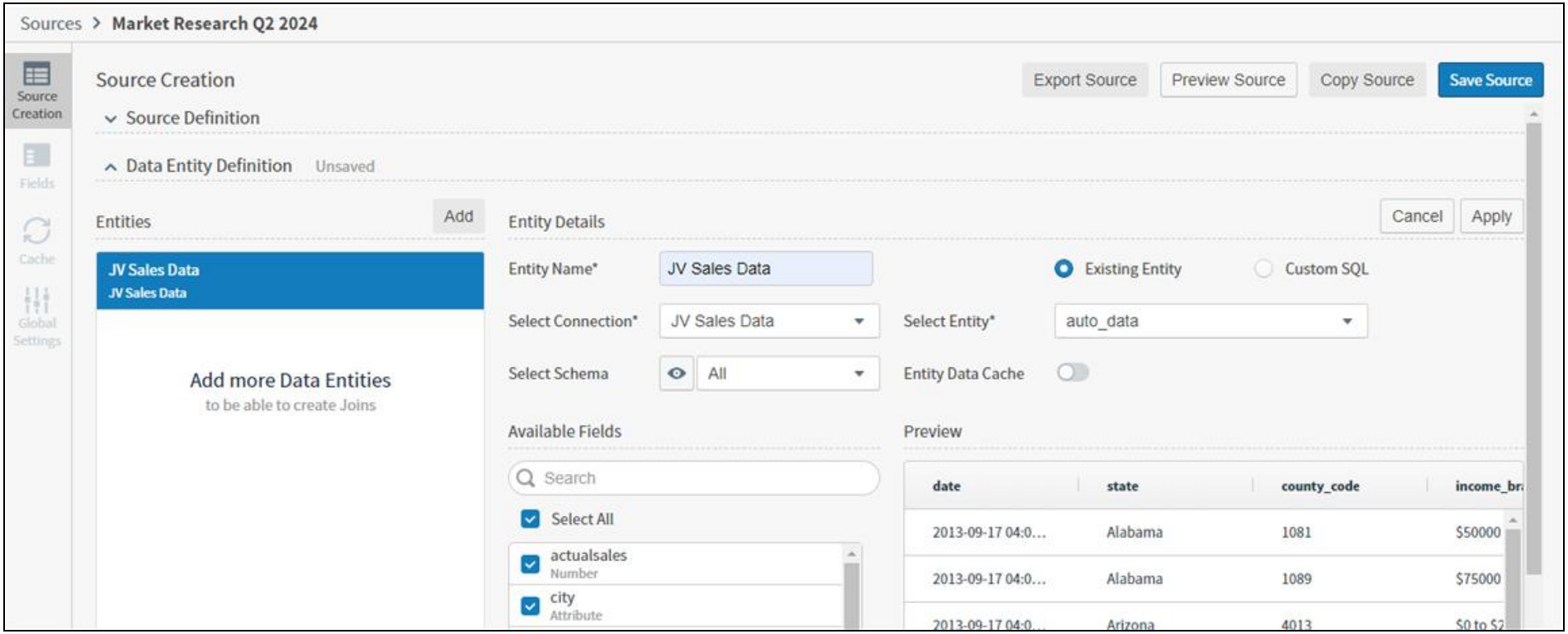


3. View the existing relationships and any user defined joins to better understand the relationships present.
4. Close the work area when you're done viewing the schema information, and repeat for other schemas if needed.

Note: For Postgres connections, both tables and data relationship information are read from the connection. Other supported connections include table information but do not read relationship information. See Connector Support for Schema Visualization. No information is provided for unsupported connections.

Source Creation Tab

Use the Source Creation tab to define new sources, edit sources, and define the data entity or entities that make up your source.



Note: You must be logged in as user with the **Administer Sources** or **Create New Data Sources** privilege to see the Source Creation tab, or have **Read** and **Write** permissions on the source.

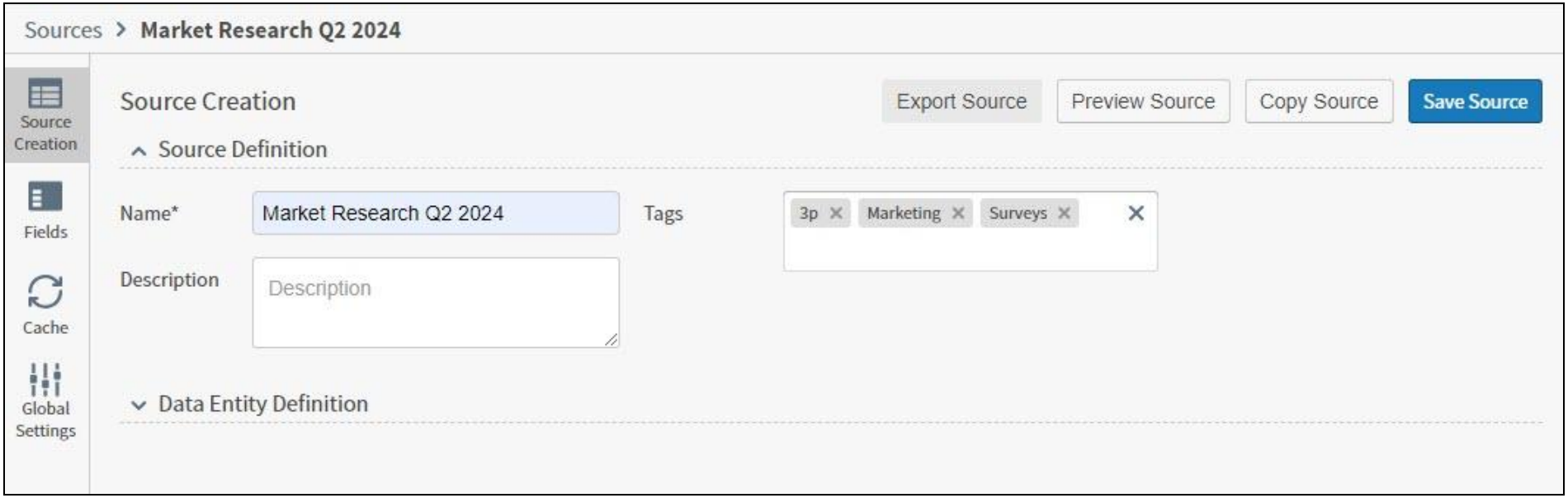
Note: When you upload a flat file, Select Schema and Select Entity fields are not present, and the option to create Custom SQL is not available. You can instead Select File, Edit File, and configure API Endpoints.

The general structure of this tab includes:

- In the upper portion of the tab, you can edit and update the Name and Description of this source in the Source Definition work area.
- In the lower portion of the tab, you can add and manage data entities in the Source Definition work area.
- In the bottom portion of the tab, you can manage Joins and Join Settings. The Join Definition work area is visible after you've added multiple data entities to create a Fusion source.

Source Creation Tab

Define basic information about this source.

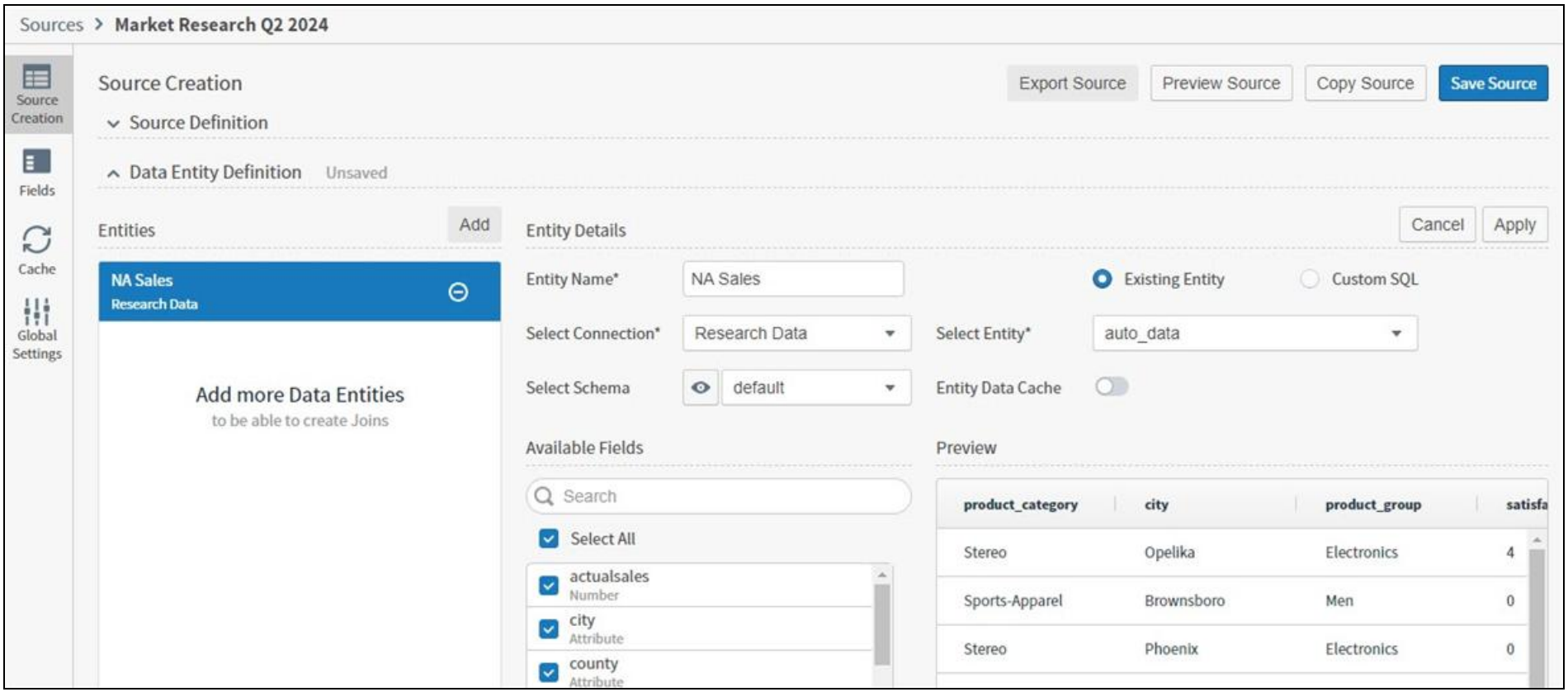


Source Definition

Add and edit the unique **Name** and optional **Description** of your source.

Data Entity Definition

Add and edit data entities for this source. After adding a data entity or making changes, select **Save Source** to save your changes, or **Preview Source** to preview your data.



Entities

Select **Add** to add a new data entity from a connection or an uploaded file.

- Select **From Connection** to add a data entity from an existing source.
- Select **From File** to add a data entity from a file.

After you've entered a data entity, you can add a filter values entity to provide an alternative source of metadata values. Use these entities in a dynamic override on Filter Values tab in Fields to improve the filter experience of heavy data entities.

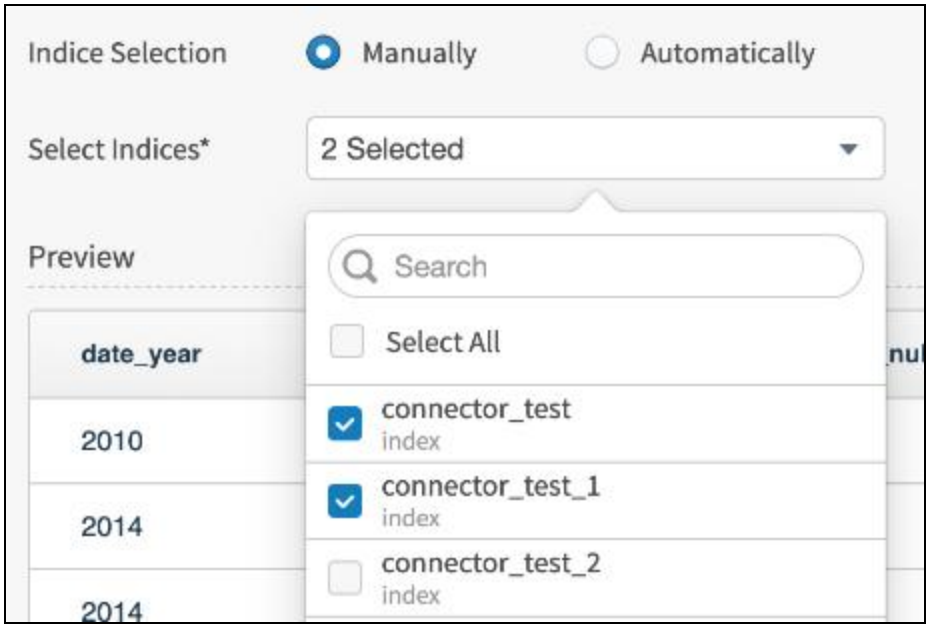
Entity Details - From Connection

Define a unique **Data Entity Name**, then define the details for how the information is accessed and presented.

Note: Depending on the your source, different options are available to define it.

When you select **From Connection**, details you can define can include:

- **Entity Name** - The unique name for this data entity.
- **Select Connection**: Select a connection for this data entity. You will only see connections you have access to.
- **Select Schema**: If a Schema is available for your connection, you can select a schema to filter a list of entities for this connection. See Connector Feature Support. Visualize your schema by selecting the view option. See View Relationships of a Schema in a Source.
- **Select Indices**: ElasticSearch connections support Indices: select one or more to include.
  - **Indice Selection**: Select **Manually** to create a merged list of fields from selected indices, or **Automatically** to select a pattern that automatically selects indices.

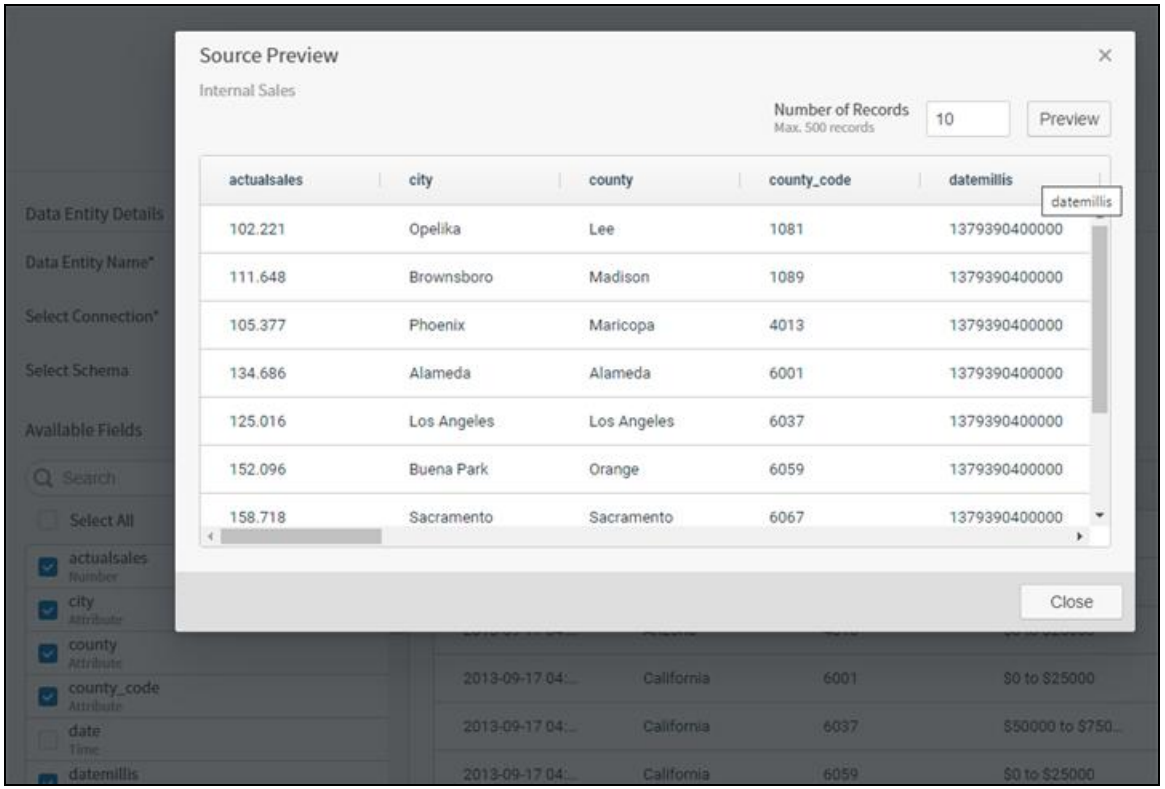


- **Existing Entity** and **Custom SQL**:

- Select **Existing Entity** to use an available entity from your source.
- **Custom SQL**: Select to define a custom SQL to retrieve the data you want from your source. **Select Entity** is not available if you select Custom SQL.

- **Select Entity**: Select an available entity. A list of all native fields from this entity populates **Available Fields** and the **Preview** table.
  - **Available Fields**: All fields available from this source are included by default. Disable (uncheck) specific fields to exclude them from the source. The fields, when disabled, are not included in any work areas of the source. If you attempt to remove a field in use by a visual or other object, Composer will prevent you from saving your changes to the source configuration.
  - Data Entity **Preview** table: A preview of the data from the source for all fields, even if deselected in **Available Fields**.
  - Source **Preview** table: A preview of data from the source as defined in **Available Fields**. Select **Preview Source** to preview your selected data.

    In the example shown here, the **date** field is deselected in Available Fields. It is visible in the Data Entity Preview table, but not the Source Preview table.



Select **Apply** to apply your changes, or add another data entity to create joins for a Fusion source.

Custom SQL

When you select this option, a Custom SQL editing pane opens that you can use to write and run your SQL query. After you have run a successful query, the results populates **Available Fields** and the **Preview** table, and you can **Apply** your changes. You can't save invalid SQL.

Table visuals and Details dialogs display fields in the order they are retrieved from the source. When you create a source using custom SQL, your fields are shown in the order you specify.

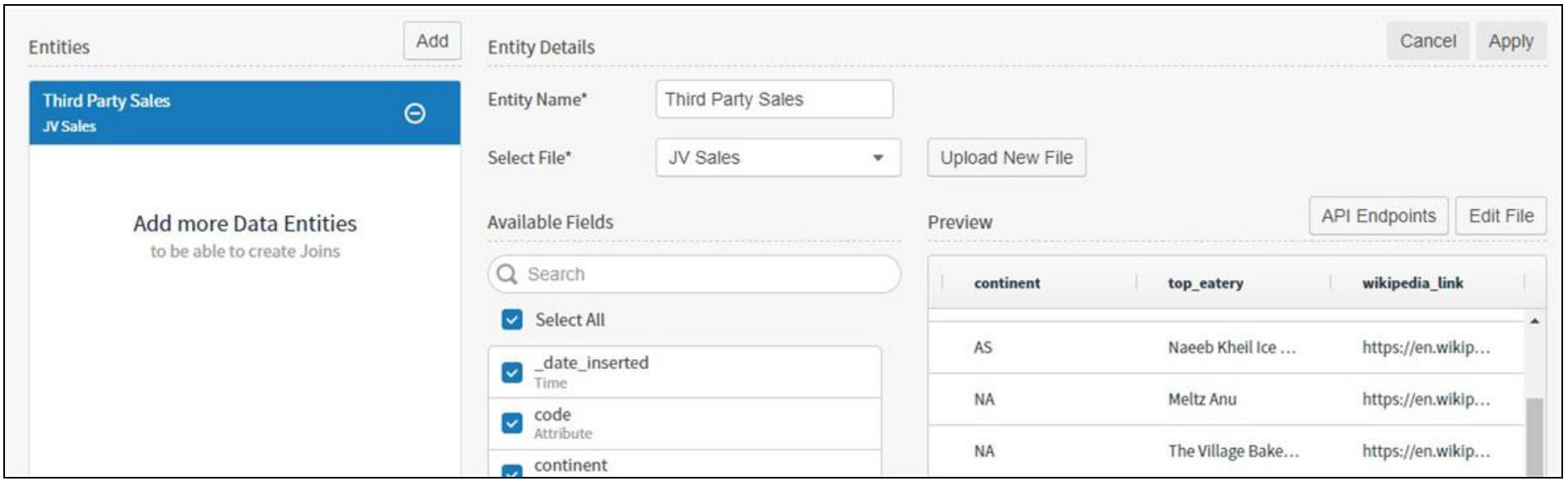
Important: Custom SQL queries are a powerful tool for performing complex data queries. However, be careful when creating custom SQL queries because it is easy to define a heavy query or a query that may overwhelm your database. Use this feature carefully.

Variables (specified as custom user attributes) can be inserted in custom SQL. See Specify Custom User Attributes. In addition, you can use a vertical bar (|) in the SQL to separate the custom attribute name from a default value used for user definitions that do not have the custom user attribute defined. For example, the following custom SQL uses the value of the `state` customer user attribute to filter source data for records from whatever state the user's `state` custom user attribute is set to. If a `state` custom user attribute is not defined for a user, a default of Alabama is used.

```
SELECT * FROM Orders WHERE state = '${User.state|Alabama}'
```

Data Entity Details - From File

Define a unique **Data Entity Name**, then define the details for how the information is accessed and presented.



When you select **From File**, details you can define can include:

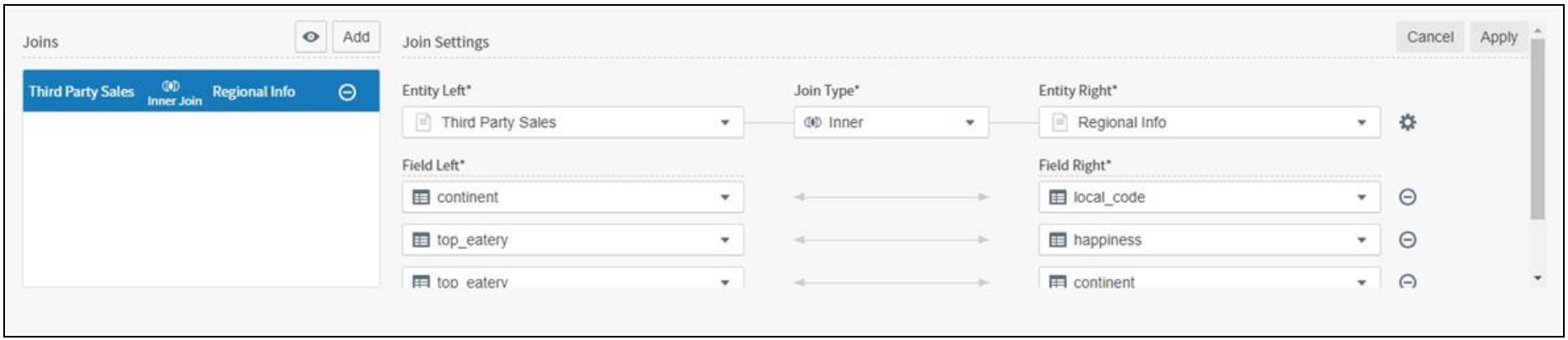
- **Data Entity Name** - The unique name for this data entity.
- **Select File**: Select an available uploaded file for this data entity. A list of all native fields from this entity populates **Available Fields** and the **Preview** table. Use **Upload New File** to add files to this source. See Manage File Uploads.
- API Endpoints: Select for more information about appending data, replacing data, or deleting data from your file upload.
- **Edit File**: Select to edit the **File Details** of your uploaded file.

Select **Apply** to apply your changes, or add another data entity to create joins for a Fusion source.

Join Definition

Create joins between pairs of data entities to create a Fusion source. You must have at least two data entities in source to create a join. If you have more than one data entity in your source, all entities must be used in a join.

To create a new join, select **Add**, then define the entities, join type, and fields. Select **Apply** to finish creating the join.

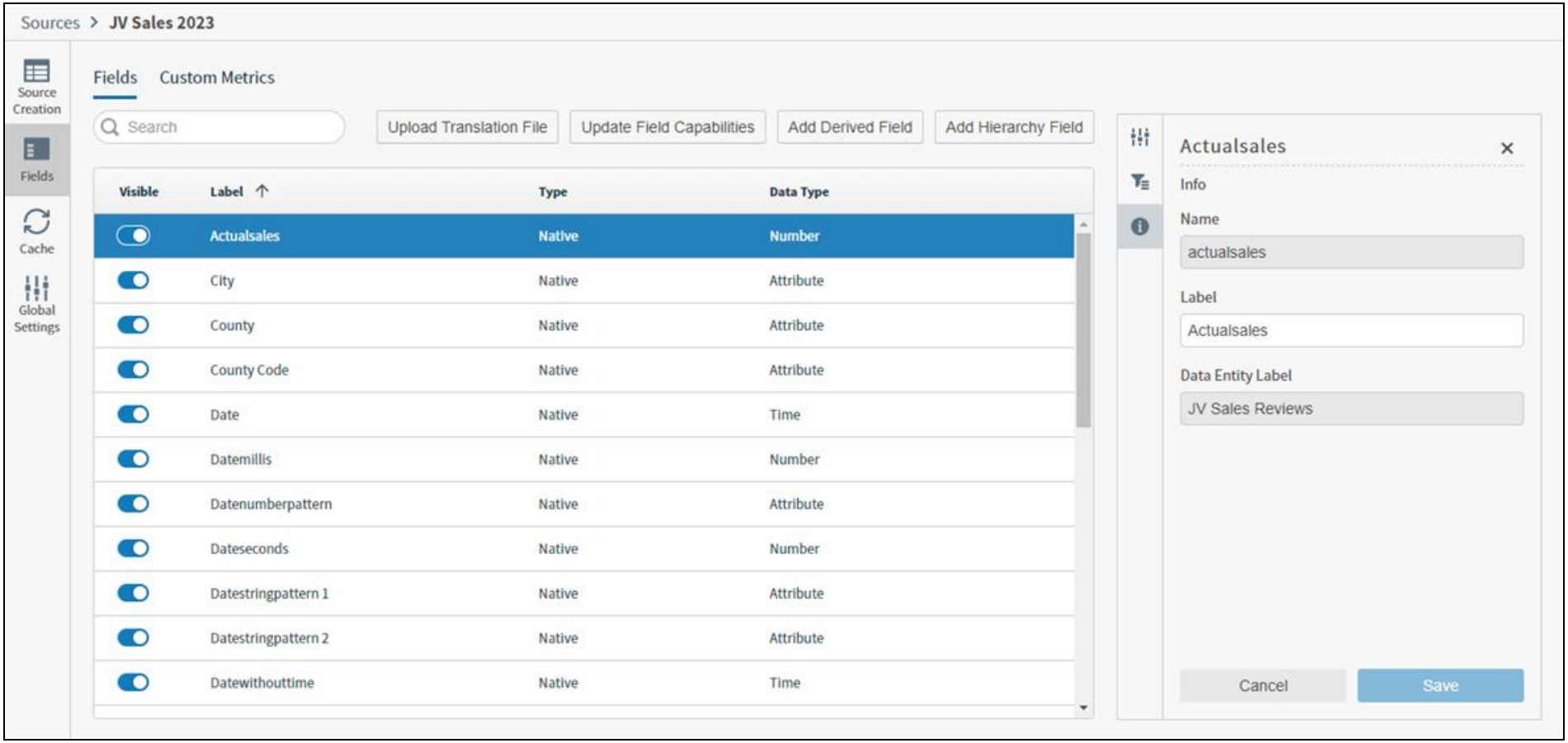


When you create a join, settings you can define can include:

- **Entity Left**: Select an available entity from the data entities you defined.
- **Join Type**: Select Left, Inner, or Full Outer.
- **Entity Right**: Select an available entity from the data entities you defined.
- **Enable Dimension Entity**: Select the gear ( ) icon to enable dimension for one or both entities. This improves the performance of queries execution by removing unused data entities from the join.
- **Field Left**: Select at least one field from this entity. You can add multiple fields by selecting the add field button.
- **Field Right**: Select at least one field from this entity. You can add multiple fields by selecting the add field button.

You can also view the relationships of your joins and add more joins in a visualization. See Create a Fusion Source.

Fields Tab



Use the Fields tab to configure settings for fields in your data source. You can search for a field by Label, upload a translation file, update field capabilities, add derived fields, add hierarchy fields, or select the Custom Metrics tab add a custom metric.

The tailored and defined fields here are used as attributes and metrics for visuals that use this data source configuration. Table visuals and Details dialogs display fields in the order they are retrieved from the source; if you add a field here, it is added to the end of the fields list.

Note: You must be logged in as a user with the **Administer Sources** or **Create New Data Sources** privilege, or write permission for this source.

The Fields tab is split into two main tabs with data tables and sidebar menus you can use to manage individual field settings.

The Fields Table

The field tables lists all the fields in the records of the data source collection or table you selected on the Source Creation tab and allows you to configure them. To define the field metadata, 1,000 records are sampled. Select **Update Field Capabilities** to make bulk changes to the fields in your source. Select **Add Derived Field** to add a derived field to this table.

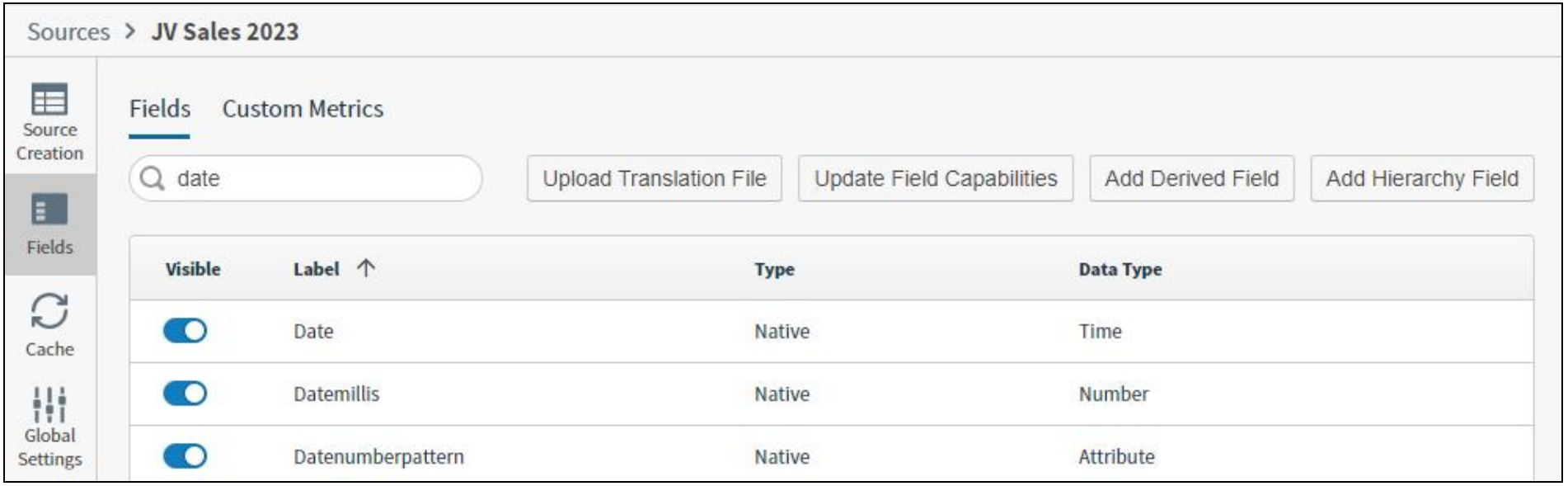
The following table describes the settings you can alter for individual data fields. To refresh your data source metadata, select the refresh button for Manual Refresh on the Cache tab. See Trigger Refresh Jobs for more information.

Column	Description
Visible	By default, all fields are visible, and you can include data from these fields in your visuals. If you want to hide specific fields, select the toggle to hide the field. Hidden fields can be added to and used in derived fields and custom metrics, but not visuals. See Hide Fields. If a hidden native field is the default metric for a new visual, another metric is used in its place.
Label	By default, the name of the field as defined in the data from the data store. To change, edit the Label field in the Settings side panel.
Type	Shows the field type. Fields from your data source are **Native**. Derived fields are **Derived**.

Column	Description
Data Type	The data type for each field is defined, by default, by Composer and shown here. If available, select **Convert** in the Data Details section of the Settings side panel to create a derived field of this data as another data type. Options include Time, Number, or Attribute. In environments where alternate calendars are enabled, a new Time derived field based on the selected calendar is created. See Use Fiscal Calendars.
Actions	Shows what actions, if any, you can take for this field. You can only delete derived fields here. **Note:** If you try to delete a visual, filter snippet, dashboard, dashboard link, source, or source field, Composer displays an error message naming any objects dependent on the item you're trying to delete. You can delete the item after you've removed the association from the dependent object. See Fields Usage.

Data Options - Fields Table

Use the search field positioned above the Fields table to find specific fields by label. Select other available options to supplement or adjust your data. Depending on your environment, not all options shown here may be available.



- **Search** - Search for a field by Label name.
- **Upload Translation File** - Upload alternative field names in your users' preferred language. See Upload a Translation File For a Source.
- **Update Field Capabilities** - Adjust the predefined capabilities of native and derived fields. See Update Field Capabilities.
- **Add Derived Field** - Create and test derived fields for your data. See Create and Modify Derived Fields.
- **Add Hierarchy Field** - define a hierarchy field from your hierarchical data source. See Define a Hierarchy Field for Your Source.

Settings Sidebar Menu - Fields Tab

Select a field in the fields table to edit and update information for each field. Depending on the field selected, available options change. Select **Save** to save any changes you make here.

Setting	Description
Expression	The expression used to define this derived field. Select the edit  button to change.
Data Details	Depending on the Data Type of the field, different options are available you can adjust. ■ Data Type: Convert to an available data type. The original field is not converted: a new derived field with the new data type is created. Data types that may be available are Time, Number, or Attribute. ■ Default Aggregation: Select an available default aggregation type. For explanations, see Metric Aggregation Functions. ◦ For numeric data types, select from: Sum, Avg, Min, Max, Count, or Distinct Count. ◦ For attribute data types, select from Count or Distinct Count. ■ Granularity: Select a default smallest increment level of time granularity to include in visuals. Valid granularity options are **Year**, **Quarter**, **Month**, **Week**, **Day**, **Hour**, **Minute**, **Second**, or **Millisecond**. You can edit a visual to include a larger granularity increment than the default for this field, but not a smaller increment. ■ Time Zone: ◦ Select a time zone to present data in a specific time zone with the time zone labeled.

Setting	Description
	◦ Select **User Time Zone** to present data in user's time zone, labeled. ◦ Select **Not Specified** to display time information with no time zone data label. Your selection here also affects exported data.
Format	The number or time format for this field. Select the edit  button to change the format type and edit display details for field. ■ Number field options: **Plain Number**, **Percentage**, **Money**, **Storage**, or **Scientific Notation**. See Configure Number Formatting - Data Sources. ■ Time field options: Adjust the display format of any available date or time field. See Configure Date and Time Formatting - Data Sources.
URL Formatting	Enable to display a hyperlink associated with this Attribute or Number field as a link in the Data Details table for a visual. When selected, the URL opens in a new browser tab. Select the **Interpolated Expression** add icon  and enter the link address, such as `https://www.website.com/${value}`.
Partition	If you are using Cloudera Impala, Apache Drill, Hive, or Spark SQL as your data source, the Partition section shows if fields within your source are partitioned. Enable to optionally adjust the Partition Field and Partition Function of the partition.

Filter Values Menu - Fields Tab

Select a field in the fields table to define filter values for each field. Choose a static override option if you want to specify Custom Value and Range. Depending on the field selected, available options change. Select **Reset** to clear unsaved changes and return to previously saved settings. Select **Save** to save any changes you make here.

Source of Filter Values	Setting	Description
Default		Select to inherit values from the data source. Select **Reset** to reset any changes you've made, even saved changes, to **Default**.
Dynamic Override		Select values from predefined entities. Apply to Attribute, Number, or Time fields. ■ Data Entity: Select an available data entity by name from the list.

Source of Filter Values	Setting	Description
		■ Field: Select field. ■ Preview: Select the number of rows to preview (10-100).
Static Override		Select to manually enter filter values. You can enable **Custom Value**, **Custom Range**, or both. If the values you define are invalid, you can't save your settings.
	Custom Range	Enable to define a custom range from available values. Apply to Number and Time fields. Adjust **Min** and **Max** to suit your needs. Applies to all visuals. These fields must defined.
	Custom Value	Enable to define custom values for this field at the source level. Apply to Attribute and Number fields. ■ Enter a custom value or variable, then select **Add** to add the value. Repeat as needed. Select delete icon  to delete a value or variable. ■ Alternatively, set no values or variables. Appropriate users can define values at the visual level.

Info Sidebar Menu - Fields Tab

Select a field in the fields table to review information about or update the label for each field. Depending on the field selected, available options change. Select **Save** to save any changes you make here.

Setting	Description
Name	The name of the field as defined in the data from the data store. Unique. When you add a derived field, Composer generates the Name using information in the Label field. The Name must be unique; Composer appends a number if needed. **Note:** Composer supports underscores and periods in data store field Names. No other special characters or white spaces are supported. Names can start with a letter or an underscore, followed by letters, digits, underscores, and periods.
Label	The Label information for the field as defined in the data store, or that you define when you create a derived field. Edit as needed for use in your visuals and dashboards. 255 characters. The Label field does not need to be unique.
Data Entity Label	The name of the data entity source for this field.

The Custom Metrics Table

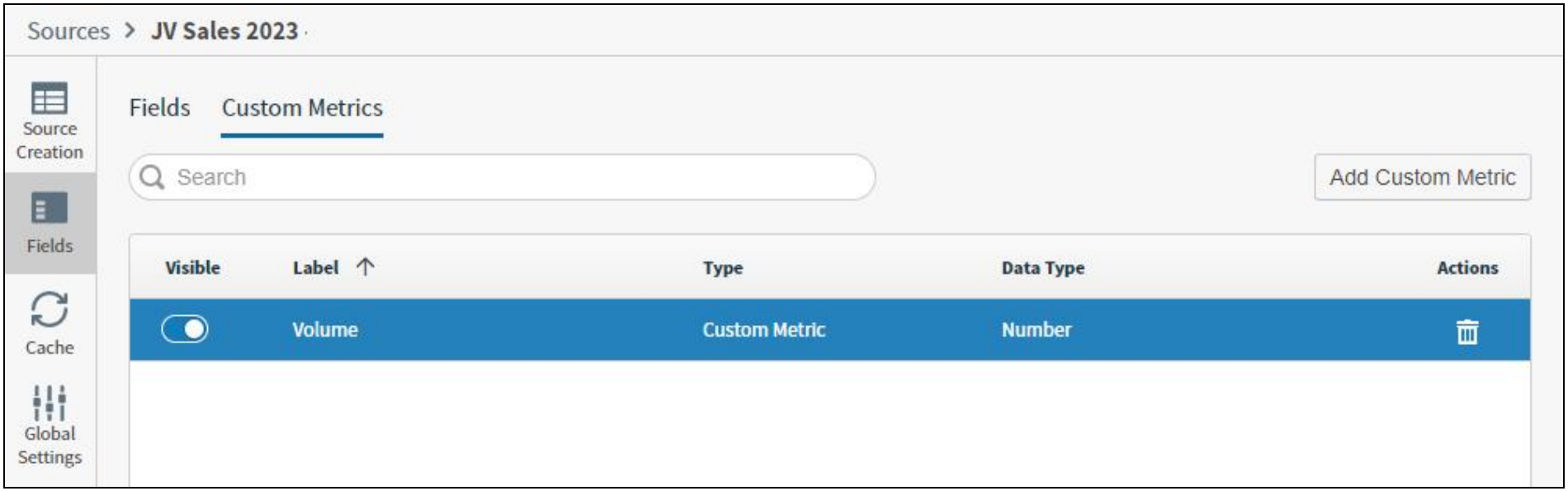
The Custom Metrics table lists custom metrics you have defined for the data and allows you to define others. You can also select **Add Custom Metric** to add a custom metric to this table.

- When you create a new source, Composer presents Volume as a custom metric, using the **Count(*)** expression. Edit or delete as needed to reflect your information as desired.

- When you upgrade Composer from an older release, volume metrics for existing sources are converted to a custom metric.

Column	Description
Visible	By default, all the fields are selected and are visible. This means that you can visualize the data from these fields on your visuals. If you want to hide specific fields, select the toggle to hide the field. Hidden fields can be used in custom metrics. However, they cannot be used in dashboard visuals. See Hide Fields. If a hidden native field is the default metric for a new visual, another metric is used in its place.
Label	The Label information for the field you define when you create a custom metric. To change, edit the Label field in the Settings side panel.
Type	Shows the field type. Custom metrics are always Custom Metric.
Data Type	Shows the data type. For custom metrics, this is always Number.
Actions	Shows what actions, if any, you can take for this field. Generally, you can only delete custom metrics.

Data Options - Custom Metrics Table



- **Search** - Search for a custom metric field by Label name.
- **Add Custom Metric** Open the Custom Metrics Editor. See Create and Modify Custom Metrics.

Settings Sidebar Menu - Custom Metrics Tab

Select a field in the custom metrics table to edit and update information for each metric. Select **Save** to save any changes you make here.

Setting	Description
Expression	The expression used to define this custom metric. Select the edit button to change.
Format	The number format for this field. Select the edit button to change the format type and edit display details for the format type. Options are **Plain Number**, **Percentage**, **Money**, **Storage**, or **Scientific Notation**. See Configure Number Formatting - Data Sources.

Info Sidebar Menu - Custom Metrics Tab

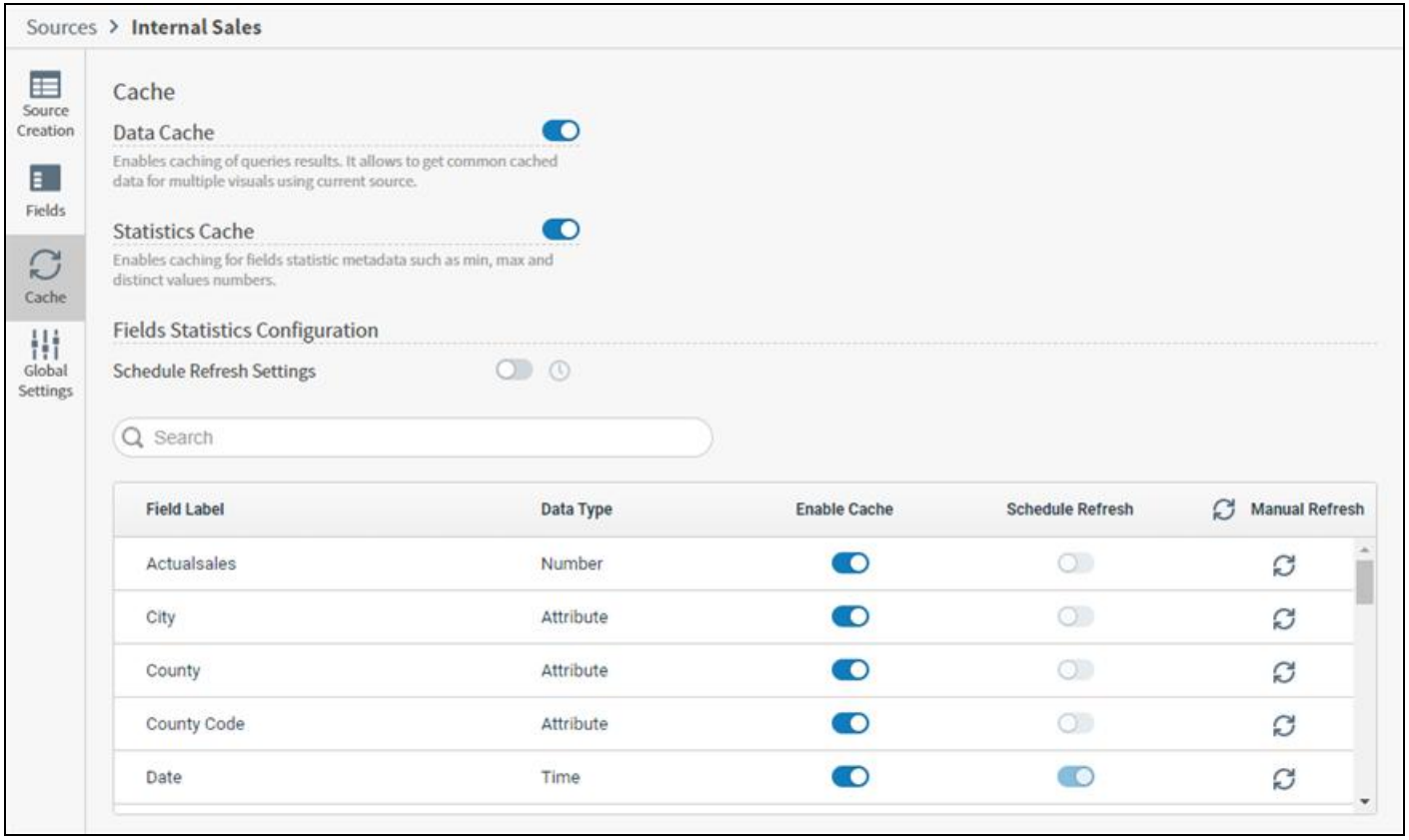
Select a field in the Custom Metrics table to review information about or update the label for each field. Depending on the field selected, available options change. Select **Save** to save any changes you make here.

Setting	Description
Name	The name of the metric as you define during custom metric creation. Unique. When you add a custom metric, Composer generates the Name using information in the Label field. The Name must be unique; Composer appends a number if needed. **Note:** Composer supports underscores and periods in data store field Names. No other special characters or white spaces are supported. Names can start with a letter or an underscore, followed by letters, digits, underscores, and periods.
Label	The Label information for the field you define when you create a custom metric. Edit as needed for use in your visuals and dashboards. 255 characters. The Label field does not need to be unique.

Cache Tab

Note: The data source configuration wizard and Refresh tab have been removed from Composer. See Define A Source.

Composer maintains data source metadata and, optionally, a cached result set of the data and statistics from the data store for each data source configuration you define.



Use the Cache tab to:

- Enable or disable **Data Cache** for this source. Any change you make to this section is automatically saved.
  - When you enable Data Cache, Composer caches query results and common cached data for multiple visuals that use this source.
  - When disabled, this metadata is not cached.

- Enable or disable **Statistics Cache** for this source. Any change you make to this section is automatically saved.
  - When you enable Statistics Cache, field metadata, such as minimum, maximum, and distinct values numbers are cached. You can enable and disable caching for individual field statistics when enabled.
  - When disabled, the **Fields Statistics Configuration** work area is disabled. If you disable this after setting up **Schedule Refresh Settings**, your schedule is deleted.

- Manage caching for individual fields in the **Field Statistics Configuration** work area. Any change you make to this section is automatically saved.
  - You can enable or disable caching for each field, scheduled refreshing for each field, or manually refresh each field if needed. Fields that include a statistics override that prevents refreshing are indicated by an exclamation point in a triangle.

- Use **Schedule Refresh Settings** to define Periodic or Advanced refreshing of fields with **Schedule Refresh** enabled. If you disable Schedule Refresh Settings, any schedule you had set up previously is deleted.

- You can perform several bulk functions related to caching using menus in the header of the fields table.
  - Select the Enable Cache menu (☰) button to quickly enable or disable caching for all fields.
  - Select the Schedule Refresh menu (☰) button to quickly enable or disable scheduled refreshing for all fields. This is available only if you have enabled **Schedule Refresh Settings** and defined a frequency.
  - Select the refresh (⟳) button for Manual Refresh to trigger a manual refresh for all fields.

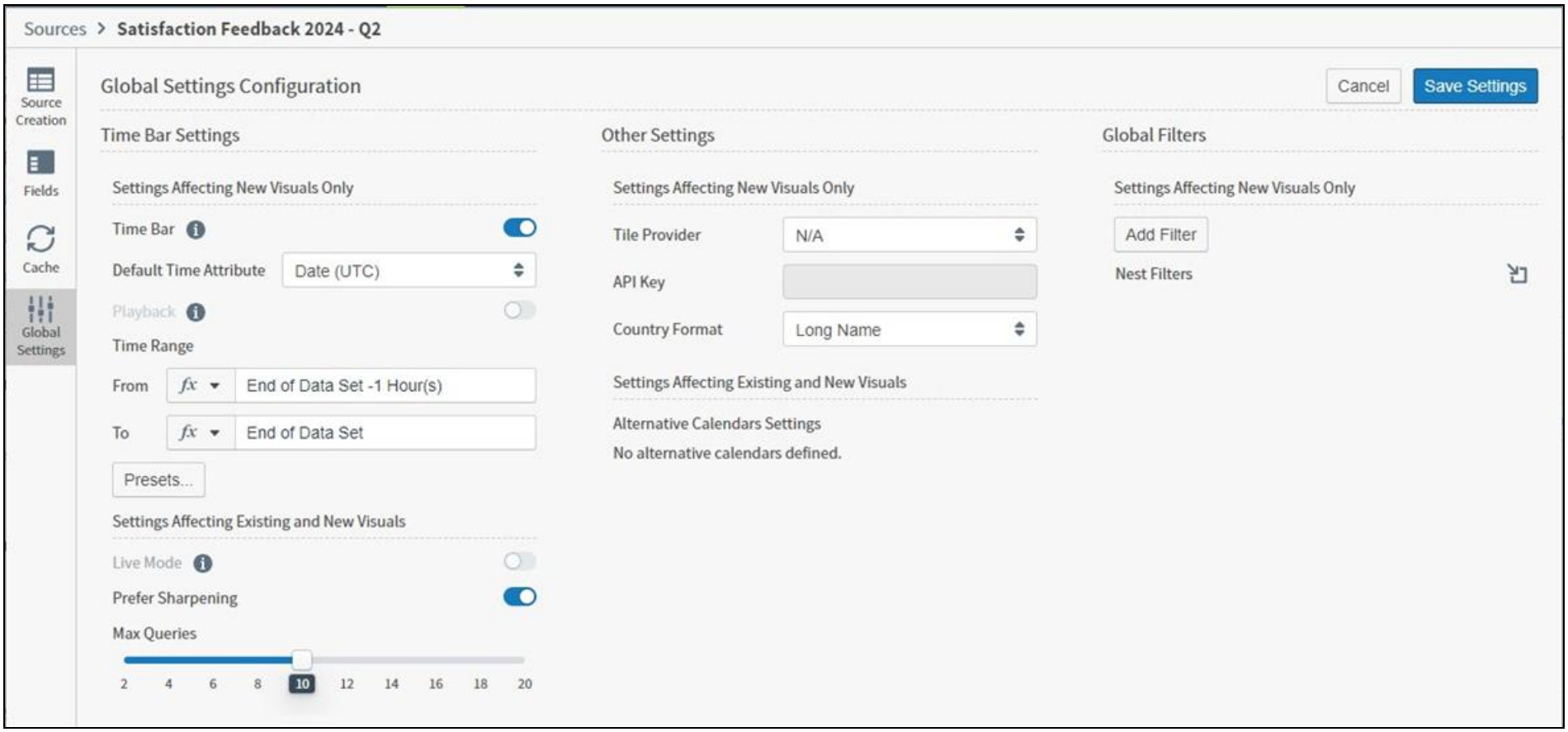
> **Note:** When you add a new field to a data source, the scheduled refresh is not enabled for the new fields by default. Quickly enable scheduled refresh for all fields using the bulk update option in the Schedule Refresh menu on the Cache tab, or enable each field for scheduled refresh manually on the Cache tab.

You can refresh the entire data source, all the fields in a data source, or select fields in a data source. For more information, see:

- Trigger Refresh Jobs
- Set Up A Data Source Refresh Job
- Review Refresh Jobs

Global Settings Tab

Use the Global Settings tab to configure settings for new visuals for this data source. If you have the **Create New Data Sources** privilege, or write permission to an existing source, you can update these settings as needed.



Depending on the connection type or source definition, not all configuration options are available for all data sources.

- Time Bar Settings
- Other Settings
- Global Filters

Time Bar Settings

If time fields are available in this data source, you can adjust the global settings here. The following table describes the time bar settings you can alter for new and existing visuals.

Setting	Description	New Visuals	Existing Visuals
Time Bar	All settings for Time Bar Settings here are disabled if the Time Bar toggle is disabled. Enable to allow setting of time bar related values. ■ Enable to allow setting of time bar related global settings here, and to enable the time bar on visuals by default. Users can disable the time bar for individual visuals as needed. ■ Disable to prevent setting of time bar related global settings here, and to disable the time bar on visuals by default. Users can enable the time bar for individual visuals as needed.	Yes	No
Default Time Attribute	Select an available time field to use, by default, for new visuals. Options vary based on the time field in your data source.	Yes	No
Playback	Enable to allow playback for optimal time fields, such as time fields with indexes, partitions, or other query optimizations from your data source.	Yes	No
Time Range	Define the time range for the time bar. By default, the time range runs from the end of the data set minus one hour to the end of the data set as dynamic time. If you don't want to use the default, you can select a preset value from **Presets...** or design your own Conditions for the From and To fields. See Configure Time Bar Defaults.	Yes	No
Live Mode	Enable to allow live stream updates from data sources that are not file-based data sources. When enabled, you can adjust live settings and enable a delay as needed.	Yes	Yes
Refresh Rate	Use Refresh Rate to specify the data refresh rate of the Default Time Attribute for this data source. The time granularity for this refresh rate is defined as Granularity on the Fields tab. For information about using the REST API to identify and modify refresh rates, see Configure Data Source Refresh Rates Using The API.	Yes	Yes
Delay By	Use Delay By to specify the delay time when playing data in live mode.	Yes	Yes
Prefer Sharpening	Enable to allow Data Sharpening for data sources that support it.	Yes	Yes
Max Queries	When Prefer Sharpening is enabled, you can use this slider to adjust the maximum number of queries for data. See Enable Data Sharpening And Configure Its Defaults	Yes	Yes

Other Settings

The following table describes the other settings you can alter for new and existing visuals.

The following table describes the other settings you can alter for new and existing visuals.

Setting	Description	New Visuals	Existing Visuals
Tile Provider	Select to define a default tile provider. Supported providers include OpenStreetMap, MapQuest, and MapBox.	Yes	No
Tile Provider URL	Provided for OpenStreetMap only. Use the default URL, or customize with your own URL.	Yes	No
API Key	Enter the API key for your tile provider, if needed.	Yes	No
Country Format	Select a country format for visuals: **Long Name**, **Formal English Name**, **ISO 2 Symbols**, or **ISO 3 Symbols**.	Yes	No
Enable Text Search	Enable to allow text search. Available only for supported data sources, such as ElasticSearch/Cloudera sources. See Configure Search Box Defaults.	Yes	Yes
Alternative Calendars Settings	Select an available alternative calendar to use for this source. Deselect a calendar to prevent the creation of new derived fiscal time fields based on a calendar. See Use Fiscal Calendars.	Yes	Yes

Global Filters

The following table describes the filter settings you can alter for new visuals. If you define initial filters, you can increase the performance of new visuals the first time you load them.

Setting	Description
Add Filter	Select **Add Filter** to add a filter to new visuals created using this source.
Nest Filter	Select the Nest Filters ( ) button to nest your filters for new visuals created using this source.

Available Visual Types

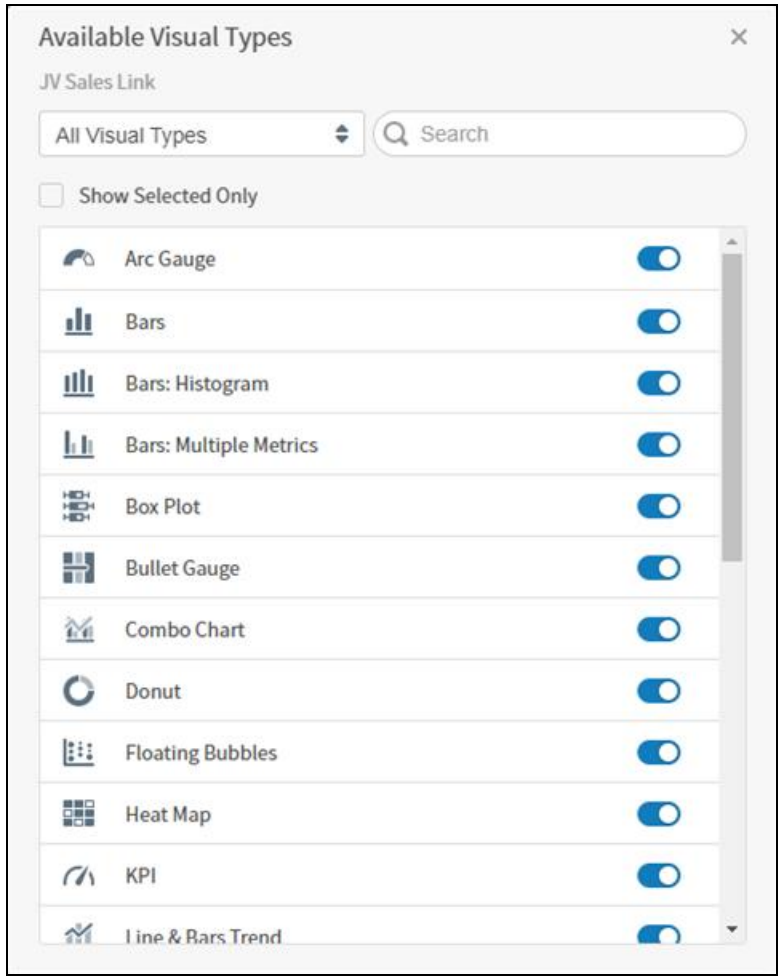
You can use the Available Visual Types work area to make specific visual styles available for a data source.

Note: If you have the **Administer Initial Visuals** privilege, you can update available visual types as needed.

Define visual types for a source

1. Make sure you are logged in as a user with the **Administer Initial Visuals** privilege.
2. Select **Sources** on the UI menu ( ). The Sources page appears.
3. On the Sources page, locate a data source configuration to edit, and select the more menu ( ) button.
4. Select **Available Visual Types**. The Available Visual Types work area for this source opens.

5. Select to enable and disable the visuals you want user to be able to use for this source. All Visual Types are shown by default: select Standard Visual Types or Custom Visual Types to edit those lists only, or use the search field to find a specific visual.



If a visual type exists for this source that you later disable, existing visuals remain, but new visuals of that type can't be made. For example, disable Bar visual types for a source to prevent users from creating Bar visual types from that source. Existing Bar visuals from that source remain unchanged.

6. After completing your changes, close the work area to save your changes for this source. All available data fields are automatically included in these default visual settings.

Access and update settings common for all visual styles on the Global Settings Tab. Global default settings allow you to set time bar and search box settings that apply to all visual styles for the data source. See Configure Time Bar Defaults, Configure Search Box Defaults, and Enable Data Sharpening And Configure Its Defaults.

For information on the specific settings available for a standard visual style, select it below.

- Arc Gauges
- Bar Chart Styles (includes standard, histogram, and multiple metric charts)
- Box Plots
- Bullet Gauges
- Combo Charts
- Donut Charts
- Heat Maps
- KPI Charts
- Line Charts (includes line & bar, attribute value, and multiple metric charts)
- Maps (includes marker, US region, and world maps)
- Pie Charts
- Pivot Tables
- Scatter (Bubble) Charts (includes floating bubble, packed bubble, and scatter plot charts)
- Sunburst
- Tables
- Tree Maps
- Waterfall
- Word Clouds

Import or Export Sources

You can import or export one or more sources in JSON format, including related connection information. When you import visuals or dashboards, their sources are also imported; this process allows you to import only sources and related connections.

Note: Special characters are not supported for import or export. If the name of a source contains special characters, change that name before continuing.

All users can view the **Sources** page.

- To import sources, you must log in as a user who belongs to a group with the **Create New Data Sources** or **Administer Sources** privilege as well as the **Manage Connections** privilege.
- To export sources, you must log in as a user who has **read** permissions for the sources and associated connection or connections.
- If you're importing and exporting a source using the import export API, you can define and use a unique key to identify the source.

  API documentation is provided with your Composer installation at this link: *https://<composer-URL>/composer/swagger-ui.html*.

Import

Import one or more sources

1. Log in as a user with the **Manage Connections** privilege andthe **Create New Data Sources** or **Administer Sources** privilege. If you are logged in as a tenant admin, verify you're in or switch to the appropriate tenant.
2. Select **Sources** on the top-level navigation banner or the UI menu, or select the **Sources** box on the Home page. The Sources work area opens.
3. Select **Import Source** in the data sources work area. The Import Source dialog opens.
4. Browse to and choose the `json` file for the sources you want to import, then select **Open**.

   The Import Sources dialog populates with information about the objects that make up your sources and the settings you can use to define how your software

inserts each object.

5. Add and remove tenants by selecting the **Tenants** field. Add or remove them from the list or field.

   Note: Only system admins or members of the Content Distributors group see the Tenants field. If this field is not shown, the content is imported into the tenant you are currently working in.

6. Optionally, enable or disable **Ignore Warnings**.

   When you enable **Ignore Warnings**, a Tags field is added to the Import work area. Add or create tags to apply to objects that do not import cleanly.

i. If errors occur during import, your software adds the tags you select to the affected objects.

ii. Use the tags to find objects you need to fix.

   Note: When you enable Ignore Warnings, items that can be imported with warnings are imported and tagged. Use these tags to find and fix the warnings in tagged objects. When disabled, no objects are imported, and errors are returned to aid in troubleshooting.

7. Select an **Insertion Strategy** for each group of objects.

   Note: The groups of objects varies based on what objects are in your JSON file.

i. **Always create objects**: Select to create an object every time, even if an existing object exists with the same name or unique ID.

ii. **Reuse existing objects**: Select to create an object if no object with the same name exists. If an object with the same name or unique ID exists, the original object is reused.

iii. **Update existing objects**: Select to update (overwrite) an existing object with the same name or unique ID. If an object with the same name does not exist, an object is created.

8. Use the default **Matching Strategy** or select the appropriate strategies for your sources in the order you want the strategies to be processed. See Matching Strategies.

9. Enable **Share Default Access With All Users** to immediately give your users access to the items you import.

10. After you've confirmed your choices, select **Import**. The visuals are imported and a success message is returned if objects import successfully or with accepted warnings. Any items imported with warnings have your selected tags applied

Note: If you import an exported source that has an associated translation file, you must re-upload the translation for that source.

Matching Strategies

When you import objects into Composer, combine these matching strategies with your selected insertion strategies to meet your organization's needs. The strategies are applied in the order you select. When you create new objects, matching strategies are not used.

Sources

Strategy	Notes
By Name	The default strategy used if no other strategies are selected.
By Origin ID	

Connections

Strategy	Notes
By Id, Type, and Parameters	A default strategy used if no other strategies are selected. Used with **By Type and Parameters** if it's not deselected.
By Type and Parameters	A default strategy used if no other strategies are selected. Used with **By Id, Type, and Parameters**.
By Name	
By Name and Type	
By Origin ID	
By Type and Parameter Keys	

Export

Export one or more sources

1. Log in a user who has **read** permissions and the **Manage Connections** privilege for the sources you want to export..
2. Select to export one or more sources by selecting the checkbox for a source to export. The **Export Selected Items** button becomes active.
3. Select **Export Selected Items**. You browser downloads the selected items in JSON format, placing them in the location you select or the default location for your browser downloads.

Edit a Data Source

You can only edit a data source configuration if you are logged in as a user with the **Administer Sources** privilege, or a user with **read** and **write** permission for the data source.

You can add additional data entities to a source to convert into a fusion data source at any time. See Create a Fusion Source.

Edit a data source configuration

1. Log in as a user with the **Administer Sources** privilege, or a user with **read** and **write** permission for the data source.
2. Select **Sources** on the UI menu (). The Sources page appears.
3. On the Sources page, locate and select the data source configuration you want to edit. The Source Creation work area opens.
4. Select and alter the settings on the tabs, as appropriate. Some changes can include, but are not limited to:

i. Source Creation Tab

a. Add or change data entities, files, or connections. If fields from the original source are in use in a visual, you can make a change if the same field is present in the new entity, file, or connection.

b. Add or remove fields. Fields can't be removed if in use in a visual, but can be hidden on the Fields tab.

ii. Fields Tab

a. Hide fields, edit Settings of fields, derived fields, and custom metrics.

b. Upload a translation file.

c. Update field capabilities for your fields in bulk.

d. Add or edit derived fields and custom metrics.

e. Add hierarchy fields.

iii. Cache Tab

a. Edit cache settings, including scheduling refresh jobs.

iv. Global Settings Tab

a. Make changes to new and existing visuals.

See also Define a Source.

5. When your changes are complete, select **Save Source**.

Important: Applied filter values that later have **Filtering** disabled to not automatically mask or hide those fields. You must recreate the filter that uses these values.

Note: When you add a new field to a data source, the scheduled refresh is not enabled for the new fields by default. Quickly enable scheduled refresh for all fields using the bulk update option in the Schedule Refresh menu on the Cache tab, or enable each field for scheduled refresh manually on the Cache tab.

If you attempt make changes that remove a field currently in use by a visual, you can't save your changes to the source unless specific conditions are met:

- You remove the visuals using affected fields.

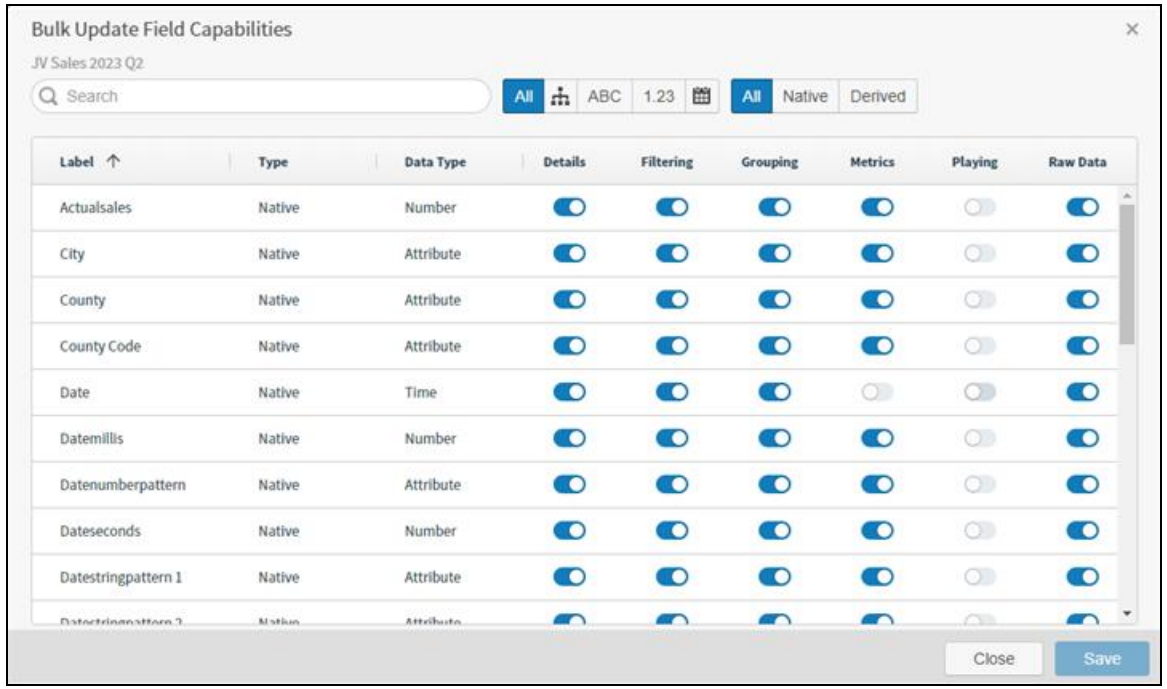
For information on configuring the time bar or search bar defaults, including the refresh rate settings, for your data source, see Configure Time Bar Defaults and Configure Search Box Defaults.

Update Field Capabilities

When you create your source or create fields in a source, Composer defines default field capabilities for your data. You can adjust the predefined capabilities of native and derived fields on the Fields tab of your source, enabling or disabling capabilities as needed.

Update the field capabilities for your source

1. Access the Fields tab for your source and select **Update Field Capabilities**. The Bulk Update Field Capabilities work area opens.



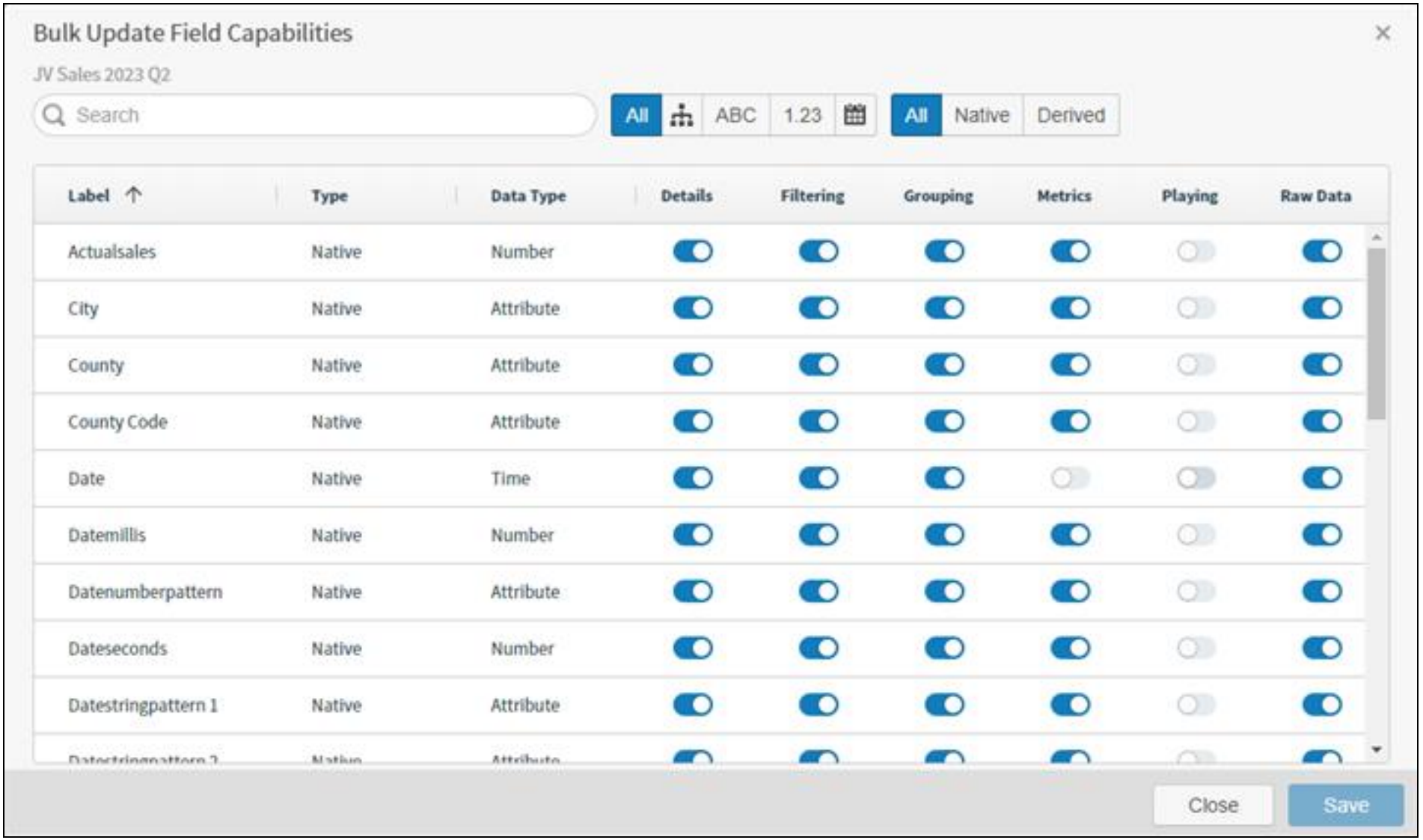
2. Only visible fields are shown in this work area. **Search** for an individual field by name, or filter the fields by data type (attribute, numeric, date) or type (native, derived).

3. Enable or disable the field capabilities for all visible fields in the source. Capabilities include: **Details**, **Filtering**, **Grouping**, **Metrics**, **Playing**, and **Raw Data**. See Field Capabilities Options.

4. **Save** your changes.

Field Capabilities

When you create your source or create fields in a source, Composer defines default field capabilities for your data. You can adjust the predefined capabilities of native and derived fields on the Fields tab of your source, enabling or disabling capabilities as needed. If you update a source, any field capabilities you set remain unchanged.



Use **Search** to find specific fields by **Label**, or use the provided filtering options to narrow the list of visible fields:

- Filter by data type: Show **All** data type fields: Attribute (**ABC**), Number (**1.23**) or Time ().
- Filter by type: **All**, **Native**, or **Derived**.

Note: You can only update the fields you have made visible in this source.

Select the toggle for any of the available options to enable or disable the field capabilities for that field. See Update Field Capabilities.

Note: If your source data does not contain enough data for a field capability, enabling or disabling it will have no effect on the field. For example, insufficient data can prevent display of details in detailed data views, even if the Details option is enabled.

Field Capabilities Options

Capability	Description
Details	When enabled, the field is visible in detailed data views. ■ **Native fields:** Enabled by default if **Raw Data** is not enabled for a field. If Raw Data is enabled, you can enable Details if needed. ■ **Derived fields:** Enabled by default.
Filtering	When enabled, you can filter data using this field in filters and filter snippets. When disabled, you can't use this field in filters. You can use this field to filter data in filter snippets when you pair it with a Display Column. This also masks the data users see, and masks the data in exports of the data. See Filter Data With Masked Fields. ■ **Native fields:** Enabled by default. ■ **Derived fields:** Enabled by default. **Important:** Applied filter values that later have **Filtering** disabled to not automatically mask or hide those fields. You must recreate the filter that uses these values.
Grouping	When enabled, the field can be included in grouping.

Capability	Description
	■ **Native fields:** Enabled by default if a `GROUP_ONLY` is not defined for a field. ■ **Derived fields:** Enabled by default. See Using The Raw Data Capability for more information on using Grouping and Raw Data together.
Metrics	When enabled, metrics are provided for this field. ■ **Native fields:** Enabled by default if the `GROUP_ONLY` flag is not defined for a field. ■ **Derived fields:** Enabled by default.
Playing	When enabled, the data can be played in the visual. ■ **Native fields:** Enabled by default if a `PLAYABLE` is defined for a field. ■ **Derived fields:** Disabled by default.
Raw Data	When enabled, the data is displayed in table visuals and can be exported with the visual to CSV and XLSX formats, using the Raw Data export option in visuals and details menus. When disabled, field data is hidden and not included in any raw data exports, including dashboard reports, and cannot be included in keysets. See Using The Raw Data Capability for more information. ■ **Native fields:** Enabled by default. ■ **Derived fields:** Enabled by default. Not supported by hierarchical fields.

Note: Meta flags are defined by the connector for each field during source creation. The connector performs data sampling to define some technical details and create the flags. These flags are not visible in the user interface. Manage using `/api/sources/{sourceId}/fields`.

Using the Raw Data Capability

Use the **Raw Data** option to control the visibility of raw data for your fields. This allows you to prevent export or use of specific fields that may contain sensitive information. It's enabled by default; disable to prevent export, views, and use of data fields as needed.

Note: If you disable Raw Data for a field used in an existing visual, the field remains visible in the visual, but cannot be exported with the visual in any format. Once you remove a field from a visual that has Raw Data disabled, you can't add it back in.

Field Capability Options	Field Behaviors
Raw Data: Enabled	Fields are available for use and access in: ■ The Details menu of all visuals, including Export options ■ Raw and Visual data exports to CSV and XLSX format for all visuals ■ Visual data exports to PDF and PNG for the raw data table visual ■ Dashboard exports to XLSX format ■ Scheduled dashboard reports for table visuals ■ Available when creating a keyset
Raw Data: Disabled	Fields are hidden from: ■ The raw data table visual, and cannot be readded from the settings menu ■ The Details menu of all visuals, including Export options ■ Raw and Visual data exports to CSV and XLSX format for all visuals ■ Visual data exports to PDF and PNG for the raw data table visual ■ Dashboard exports to XLSX format ■ Scheduled dashboard reports for table visuals ■ Not visible when creating a new keyset Keysets:

Field Capability Options	Field Behaviors
	■ Disabled data fields can still be used to filter by the full value of the field ■ You can filter a visual using existing keysets that use the disabled field Calculation Builder: ■ The field is hidden from the Calculation Builder for derived fields and custom metrics unless opened in the source editor ■ You can edit an existing derived field or custom metric that includes the hidden field ■ You can filter a visual using existing keysets that use the disabled field
Raw Data: Disabled **Grouping: Disabled**	When Grouping and Raw Data is disabled on a field, the field is additionally hidden from: ■ All other visuals ■ Visual data exports for all visuals to PDF and PNG formats ■ Dashboard exports for all visuals to PDF and PNG formats ■ Scheduled dashboard reports for all visuals

Important: To ensure a field is excluded from all exports, disable Details, Grouping, and Raw Data for that field. The field data can be exposed if Filtering capability is enabled. To prevent this, either turn off filtering, or hide the field but filter by its value by defining a custom value for the field. See Filter Values Menu - Fields Tab.

Upload a Translation File For a Source

Provide localization support by uploading metadata dictionaries for your data sources by uploading a translation file in CSV format for each data source you create. Use this translation file to define label translation for fields and custom metrics for your users in their preferred language.

Translation File Prerequisites

- Build your metadata dictionary as a comma separated values file (.csv) as shown below, using appropriate Locale labels for each language.
- Include as many or as few of the fields as you need from your data source, in any order. Composer displays the appropriate translation with each included field in the language defined by the user's settings. If no translation is provided, the original field name is shown.
- Provide a translation entry for each language represented. If your file is missing a language entry for a field, Composer rejects the file upload.

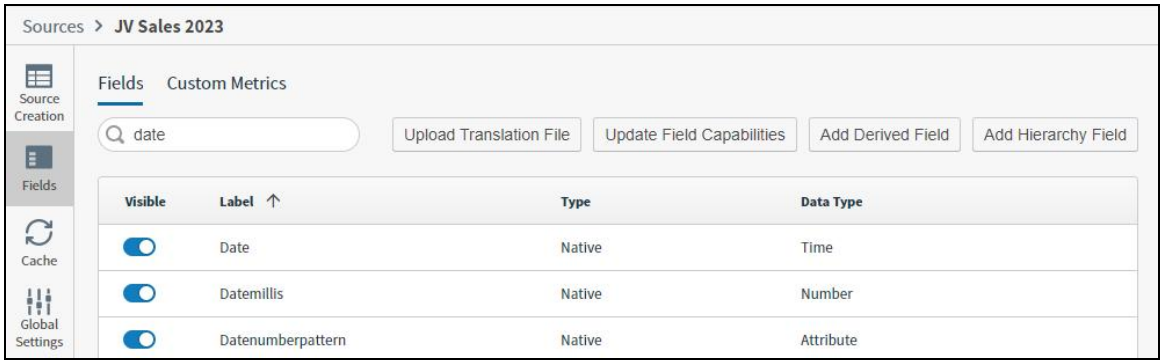
```
Field Label,ua_UK,en_GB
city,місто,city
county,округ,county
zip code,поштовий індекс,postcode
date,дата,date
income bracket,рівень доходу,income range
product category,категорія продукту,product category
satisfaction,задоволення,satisfaction
year,рік,year
```

Note: If you import an exported source that has an associated translation file, you must re-upload the translation for that source.

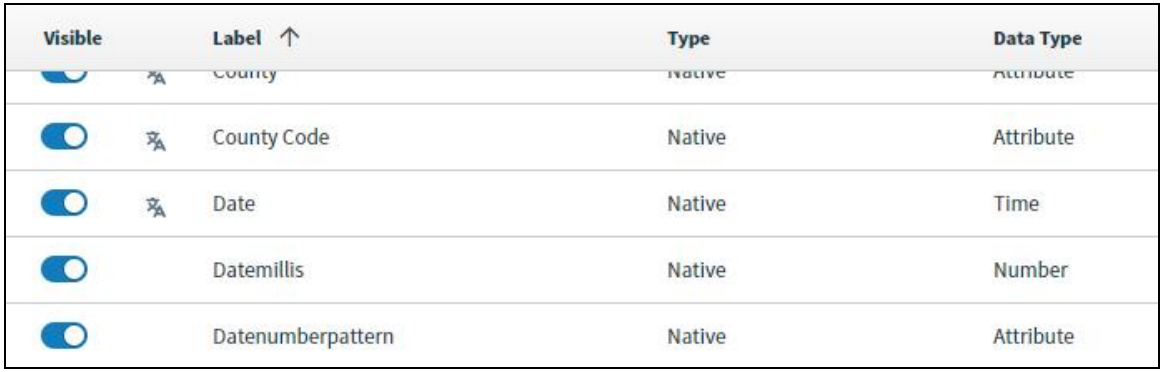
Upload or Replace a Translation File

Upload a translation file

1. Log in as a user with the **Administer Sources** privilege, or a user with **read** and **write** permission for the data source.
2. Select the Fields tab for the source, then select **Upload Translation File**. Composer prompts you to upload your file from your system.



3. Browse to your file, then select **Open**. Composer returns a success message. Translated fields are indicated by the translation icon ( ) in the appropriate table.



Replace a translation file

1. Log in as a user with the **Administer Sources** privilege, or a user with **read** and **write** permission for the data source.
2. Select the Fields tab for the source, then select **Upload Translation File**. Composer prompts you to upload your file from your system.
3. Browse to your revised file, then select **Open**. Composer overwrites the existing metadata dictionary and returns a success message. Translated fields are indicated by the translation icon ( ) in the appropriate table.

About Source Permissions

As a Composer user assigned to a group with the **Administer Sources** privilege or with the **Manage Source Permissions** privilege, you can enable users to work with data sources by enabling **Data Access**, **Read**, **Write**, and **Delete** permissions for sources.

> **Note:** If you try to delete a visual, filter snippet, dashboard, dashboard link, source, or source field, Composer displays an error message naming any objects dependent on the item you're trying to delete. You can delete the item after you've removed the association from the dependent object. See Fields Usage.

Users who create a data source can always modify or remove it, unless their permissions are revoked. Users who belong to a group with the **Administer Sources** privilege enabled have **Data Access**, **Read**, **Write**, and **Delete** permissions for any source in Composer.

You can grant data source access to users who do not belong to a group with privileges enabled by defining **Data Access**, **Read**, **Write**, and **Delete** permissions for individual sources.

Data Access is a separate permission for sources. It can be set directly on sources for users, groups, and tenants, and is enabled for users, groups, and tenants when you assign **Read** permission for a visual that uses that source. Unless they are granted Read permission to the source as well, they can't see the source listed on the Source page, or select the source to create a new visual (for users with the **Create Visuals** or **Administer Visuals** privilege).

Privilege Considerations

To manage permission settings for a source, a Composer user must meet ***one*** of the following criteria:

- The user is an administrator, belonging to the **Administrators** group.
- The user belongs to a group with the **Administer Sources** (ROLE_ADMINISTER_SOURCES) privilege enabled.
- The user belongs to a group with the **Manage Source Permissions** (ROLE_PERMISSION_SOURCES) privilege enabled. If a user only has this privilege (and *not* the **Administer Sources** privilege), they can only manage permissions for sources they can read.

  In addition, you may be restricted in which permissions you can assign. You can only assign permissions equivalent to your own. For example, if your user account has read permission for a source, you can grant and revoke the read option available on the Source Permissions panel. If you have write permission for a source, you can grant and revoke the write option on the Source Permissions panel.

  > **Note:** If your user account does not have read permission for a source, you can't see the source on the Sources page.

Source permissions are determined using a most permissive model. For more information, see How Source Permissions Are Determined.

Data Store Connection Considerations

Users with write permissions for a data source are automatically able to read the connection definitions for a data source. However, connection definitions can only be maintained by Composer administrators or users belonging to groups that have been granted the **Manage Connections** privilege.

Row and Column Security Considerations

Row and column security filters can be maintained for a data source by:

- an administrator.
- User in a group that has been granted the **Administer Sources** privilege.
- User in a group that has been granted the **Manage Source Permissions** privilege who also has **read** permission for the data source.

Security filters will not be applied to users with the privileges mentioned above. Source administrators can manage security filters for regular users but not for other source administrators.

For specific information about source permissions, see the following topics:

- Grant Permissions For A Source
- Modify Permissions For A Data Source
- Revoke Permissions For A Data Source
- How Source Permissions Are Determined

Data source permissions can also be managed using the API endpoints `GET /api/sources/{sourceId}/acls`, `PATCH and PUT /api/sources/{sourceId}/acls/bulk`, `GET /api/user/permissions/sources/{sourceId}`, `GET /api/user/permissions/sources`, and `GET /api/inventory/SOURCE/{id}`.

When you use the `GET /api/sources/{sourceId}/acls` endpoint, you can read the source data. Use `PATCH` and `PUT` to restrict the list to specific users, groups, or tenants using the `sidTypes` parameter. In addition, you can use the `returnSids` parameter to restrict the list so it retrieves only users, groups, or tenants with access to the sources or to only users, groups, or tenants without access.

API documentation is provided with your Composer installation at this link: *`https://<composer-URL>/composer/swagger-ui.html`*.

Permissions for imported objects

When you import dashboards, associated resources such as visuals, sources, and connections are imported as well. You can quickly grant default access levels to all imported and associated objects in your tenants by enabling **Share Default Access With All Users** at import time. Users are granted Data Access to Sources and Read access to Visuals and Dashboards.

How Source Permissions Are Determined

By default, the creator of a source configuration always has **Data Access**, **Read**, **Write**, and **Delete** permissions until those permissions are changed by an administrator or someone with appropriate authorization to change source permissions. If a user is removed from your software environment, sources created by that user are retained. The system admin becomes the creator of these orphaned data sources.

Note: The default **supervisor** user is no longer installed; add users to the **Supervisors** group instead.

Data Access is a separate permission for sources. It can be set directly on sources for users, groups, and accounts, and is enabled for users, groups, and accounts when you assign **Read** permission for a visual that uses that source. Unless they are granted Read permission to the source as well, they can't see the source listed on the Source page, or select the source to create a new visual (for users with the **Create Visuals** or **Administer Visuals** privilege).

If conflicting source permissions are specified for a tenant, the group within a tenant, and the user within a tenant, the permissions granted to the users are determined using a most permissive model. Users are granted the highest level of permission specified for the tenant, group, and user. For example, if the tenant is granted read and write permissions, but Group A is granted write and delete permissions, users in Group A will be able to read, write, and delete the source. However, users in any other groups in the tenant will only be able to read and write the data source.

Here's another example. If the tenant is granted data access, read, write, and delete permissions, but the groups in the tenant are only granted data access permissions, all users in the tenant will have data access, read, write, and delete permissions for the data source.

Note: If you try to delete a visual, filter snippet, dashboard, dashboard link, source, or source field, Composer displays an error message naming any objects dependent on the item you're trying to delete. You can delete the item after you've removed the association from the dependent object. See Fields Usage.

Permissions for imported objects

When you import dashboards, associated resources such as visuals, sources, and connections are imported as well. You can quickly grant default access levels to all imported and associated objects in your tenants by enabling **Share Default Access With All Users** at import time. Users are granted Data Access to Sources and Read access to Visuals and Dashboards.

Grant Permissions for a Source

You can grant read, write, or delete data source configuration permissions for your tenant, groups in your tenant, or specific users in your tenant.

Grant permissions for a data source

1. Log into Composer as an administrator or a user belonging to a group that includes the **Administer Sources** or the **Manage Source Permissions** privilege. If you are logged in as a tenant admin, verify you're in or switch to the appropriate tenant.
2. Select **Sources** on the top-level navigation banner or the UI menu, or select the **Sources** box on the Home page. The Sources work area opens.
3. Locate the row for the data source configuration in the list and select icon in its **Permissions** column. The Source Permissions dialog appears.

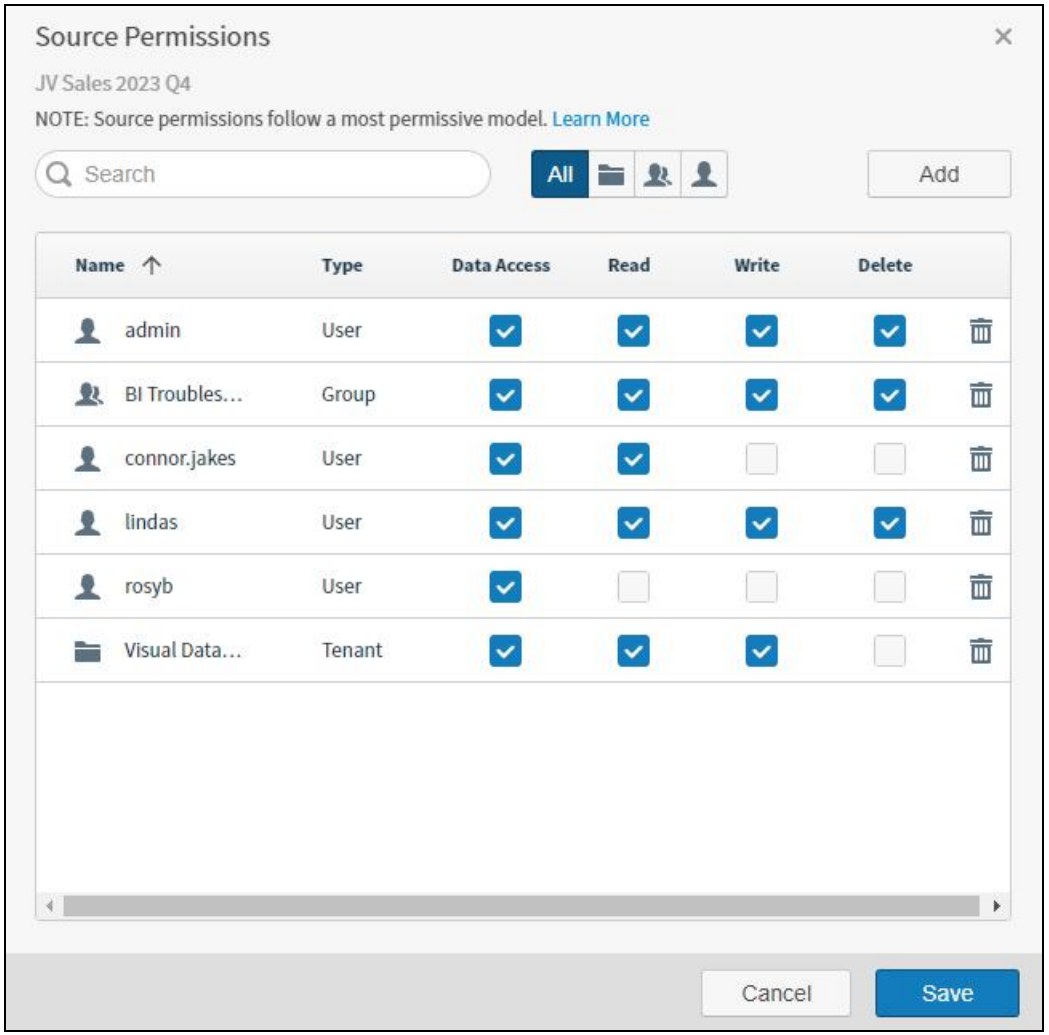
   Initially, this dialog lists only the creator of the data source.
4. Select **Add** on the Source Permissions dialog and then select **Groups**, **Users**, or **Tenant** from the drop-down menu.

i. If you select **Groups**, the Add Groups dialog appears, listing all the groups available in your tenant. The supplied groups are not shown; permissions can not be changed for those groups.

ii. If you select **Users**, the Add Users dialog appears, listing all the users available in your tenant.

iii. If you select **Tenant**, Read permission is selected for your tenant on the Source Permissions dialog.

5. Select the tenant or any specific groups or users you want to permit to read, write, or delete the data source and select **Apply**. The Source Permissions dialog lists your selections.



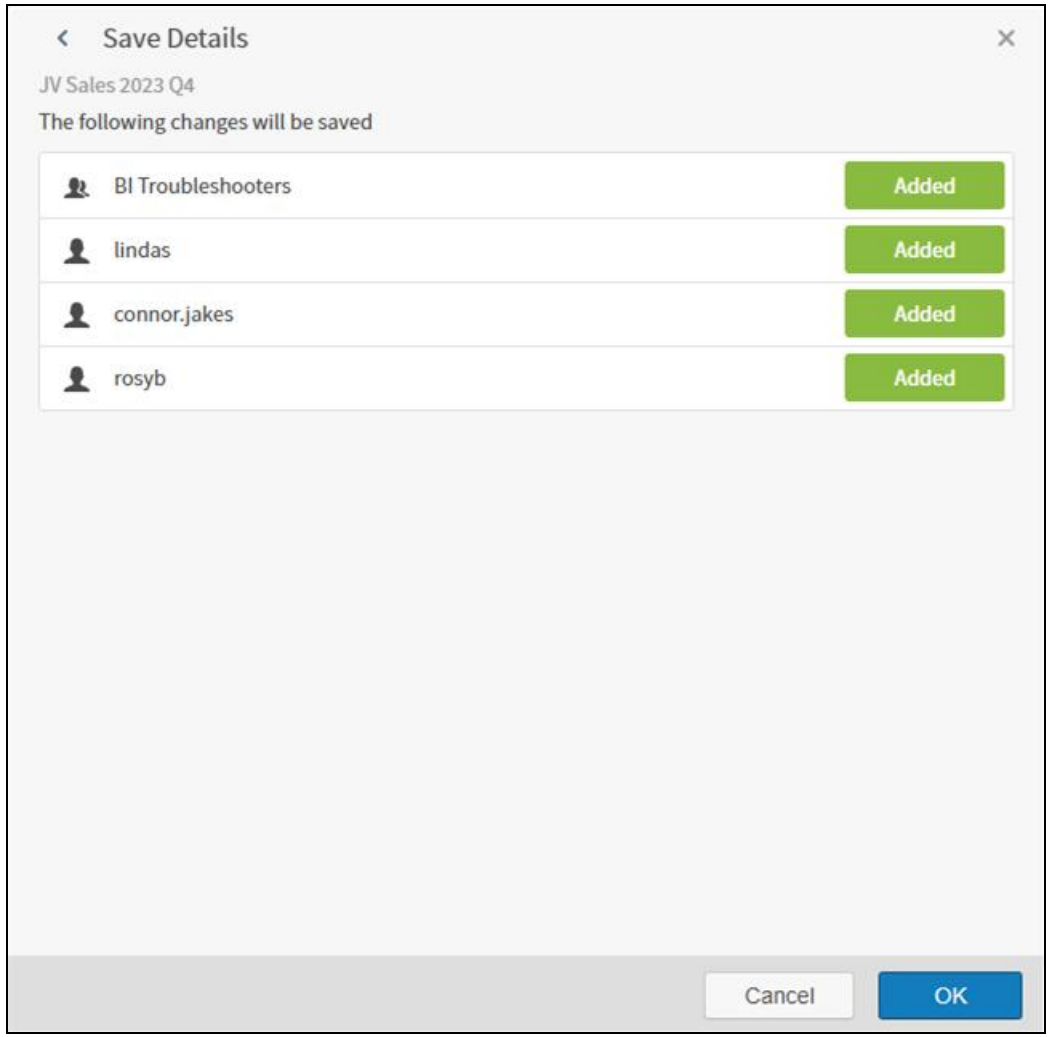
i. Members of the Administrators group have data access, read, write, and delete permissions for every data source in the tenant.

ii. The user who creates a data source is automatically selected and has **Data Access**, **Read**, **Write**, and **Delete** permissions.

6. Select the **Data Access**, **Read**, **Write**, or **Delete** checkboxes for the tenant, groups, or users to indicate what users in them can do with the data source. **Data Access** permission is assumed and is always selected. If you clear (uncheck) the check box (revoke **Data Access** permission), permission for the entire data

source is revoked for the tenant, group, or user after you save.

7. Select **Save**. The Save Details dialog appears, listing the changes that you made.



8. Review the changes and select **OK**. The source authorization permissions are set.

Permissions for imported objects

When you import dashboards, associated resources such as visuals, sources, and connections are imported as well. You can quickly grant default access levels to all imported and associated objects in your tenants by enabling **Share Default Access With All Users** at import time. Users are granted Data Access to Sources and Read access to Visuals and Dashboards.

Modify Permissions for a Data Source

You can modify the data source permissions you granted to your tenant, to groups in your tenant, or to specific users in your tenant.

Modify permissions for a data source

1. Log into Composer as an administrator or a user belonging to a group that includes the **Administer Sources** or the **Manage Source Permissions** privilege. If you are logged in as a tenant admin, verify you're in or switch to the appropriate tenant.
2. Select **Sources** on the top-level navigation banner or the UI menu, or select the **Sources** box on the Home page. The Sources work area opens.
3. Locate the row for the data source in the list and select the permissions icon in the **Permissions** column. The Source Permissions dialog appears.
4. If you want to add permissions for all users in your tenant or for additional groups or users in your tenant, select **Add** on the Source Permissions dialog and then select **Groups**, **Users**, or **Tenant** from the drop-down menu.

i. If you select **Groups**, the Add Groups dialog appears, listing all the groups available in your tenant. The supplied groups are not shown; permissions can not be changed for those groups.

ii. If you select **Users**, the Add Users dialog appears, listing all the users available in your tenant.

iii. If you select **Tenant**, Read permission is selected for your tenant on the Source Permissions dialog.

iv. Members of the Administrators group have read, write, and delete permissions for every source in the tenant.

v. The user who created the source is automatically selected and has**Data Access**, **Read**, **Write**, and **Delete** permissions unless you revoke these permissions.

5. Select the **Data Access**, **Read**, **Write**, or **Delete** checkboxes for the tenant, groups, or users to indicate what users in them can do with the data source. **Data Access** permission is assumed and is always selected. If you clear (uncheck) the check box (revoke **Data Access** permission), permission for the entire data source is revoked for the tenant , group, or user after you save.
6. Select **Save**. The Save Details dialog appears, listing the changes that you made.
7. Review the changes and select **OK**. The source authorization permissions are set.

Revoke Permissions for a Data Source

You can revoke the data source permissions you previously granted to your tenant, to groups in your tenant, or to specific users in your tenant.

Revoke permissions for a data source

1. Log into Composer as an administrator or a user belonging to a group that includes the **Administer Sources** or the **Manage Source Permissions** privilege.
2. Select **Sources** on the top-level navigation banner or the UI menu, or select the **Sources** box on the Home page. The Sources work area opens.
3. Locate the row for the data source configuration in the list and select the permissions icon in the **Permissions** column. The Source Permissions dialog opens.
4. To completely revoke all source permissions for the tenant or for a group or user, locate the row for the tenant, group or user on the Source Permissions dialog and select the delete icon. The tenant , group, or user is removed from the dialog.

   You can also revoke specific permissions by changing the checkbox selections for the tenant or group on the Source Permissions dialog. If you clear (uncheck) the **Data Access** box (revoke **Data Access** permission), permission for the entire data source is revoked for the tenant, group, or user after you save. See Modify Permissions For A Data Source.
5. Select **Save**. The Save Details dialog appears, listing the changes that you made.
6. Review the changes and select **OK**. The source authorization permissions are set.

Restrict Access to Fields Using Column Security

Users with appropriate permissions can manually restrict the fields in a Composer data source that can be viewed or used by the members of one or more groups in visuals and dashboards. By default, all data fields are available for groups.

Column security filters can be maintained for a data source by:

- an administrator.
- User in a group that has been granted the **Administer Sources** privilege.
- User in a group that has been granted the **Manage Source Permissions** privilege who also has **read** permission for the data source.

Security filters will not be applied to users with the privileges mentioned above. Source administrators can manage security filters for regular users but not for other source administrators.

If a user is included in more than one column security filter for the same data source, a most permissive model for the filter is used. For example, suppose a user is a member of two groups, Group A and Group B and that column filters have been created so Group A is restricted from using Field A and Group B is restricted from using Field B. The user will be able to use both Field A and Field B because they are in both groups and Group A can use Field B and Group B can use Field A. Likewise, if Group A is restricted from using Field A, but Group B has no restrictions, the user can use Field A.

Important: Users for whom column security filters have been applied in a data source will receive an **Invalid Visual Configuration** error for a dashboard based on the data source if the dashboard shows any of the fields the user is restricted from seeing.

Column security is supported by the API endpoint `/api/sources/<source-id>/security/attributes`.

API documentation is provided with your Composer installation at this link: *`https://<composer-URL>/composer/swagger-ui.html`*.

This section covers the following topics:

- Add Column Security Definitions
- Modify Column Security Definitions
- Remove Column Security Definitions

Add Column Security Definitions

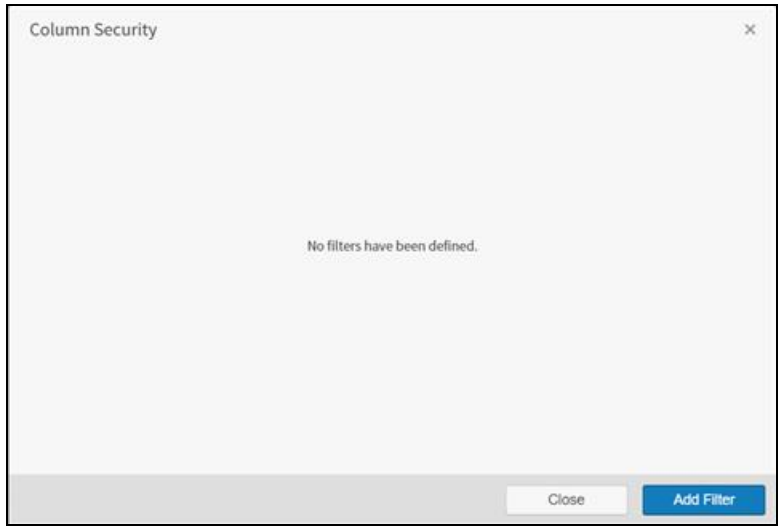
Add column security to restrict the data source fields that can be viewed or used by the group.

1. Log into Composer as a user in a group that has been granted the **Administer Sources** privilege, or a user in a group that has been granted the **Manage Source Permissions** privilege *and* who also has **read** permission for the data source.

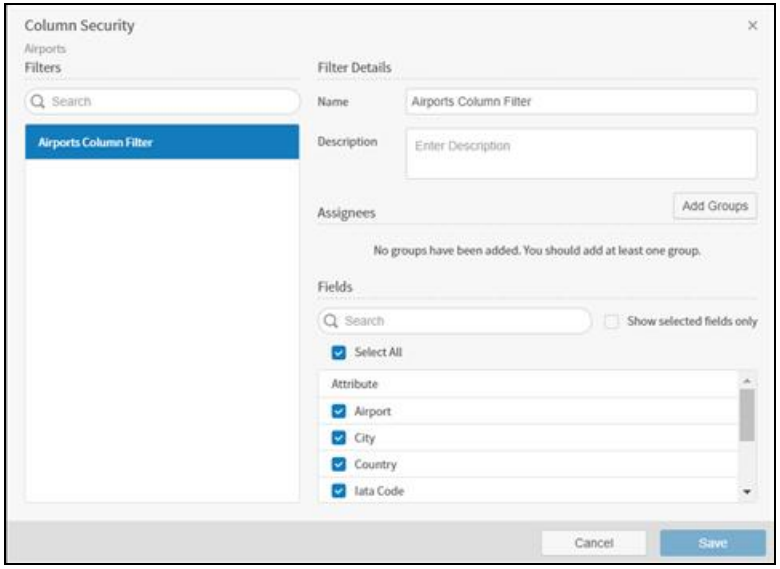
   If the user name you log in with is also associated with other Composer accounts, verify that the correct account is selected. See Switch Tenants.

2. Select **Sources** on the top-level navigation banner or in the UI menu ( ). The Sources page appears.

3. Locate the data source for which you want to restrict field access and select in the **Column** column for the data source. The Column Security dialog appears.



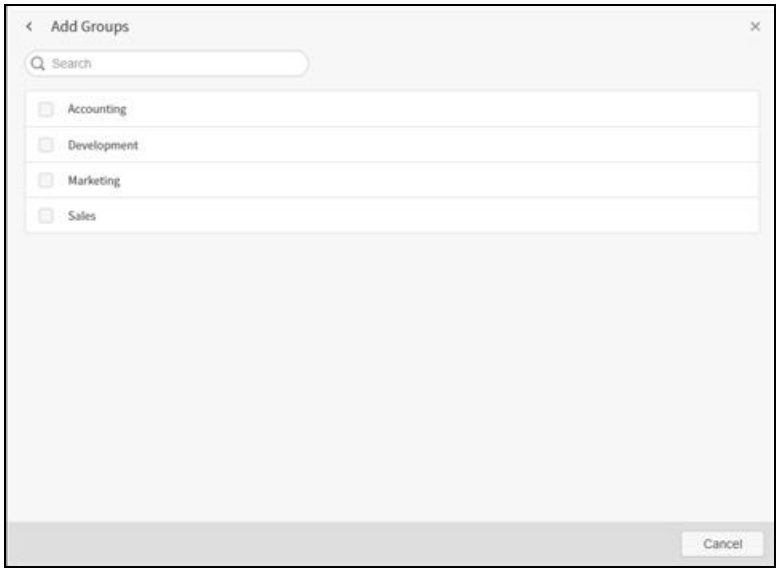
4. Select **Add Filter**. The Column Security dialog fills with information about the data source you selected.



5. Specify a name for the column security definition in the **Name** field. This name will be used to distinguish one column security definition from another in the Column Security dialog.

6. Optionally, use the **Description** field to supply a description for the column security definition.

7. Select **Add Groups** to select one or more groups to which the column security definition applies. The Add Groups dialog appears.



8. Select one or more groups for the column security filter. You can search for group names using the search box at the top of the dialog. When you have finished selecting groups, select **Apply**.

   The groups appear under **Assignees** on the Column Security dialog. If you want to remove a group from the filter, select next to the group name in the **Assignees** section.

9. Select fields in the **Fields** list to be visible to the members of the selected groups. By default **Select All** is selected. To restrict the fields visible to group members, clear (uncheck) **Select All** and manually select fields in the list (**Select ALL** is automatically unselected when you clear (uncheck) a field in the list). Use the Search bar to search for fields in the list.

   To see only the fields you have selected in the list, select **Show selected fields only**. To see all fields in the list (including fields you have not selected), clear (uncheck) **Show selected fields only**.

10. When you are finished selecting data source fields that can be visible to the groups, select **Save** to save the column security definition.

11. Repeat Steps 4-9 to add more column security definitions.

12. When all column security definition modifications have been made, select **Close** to close the Column Security dialog.

Modify Column Security Definitions

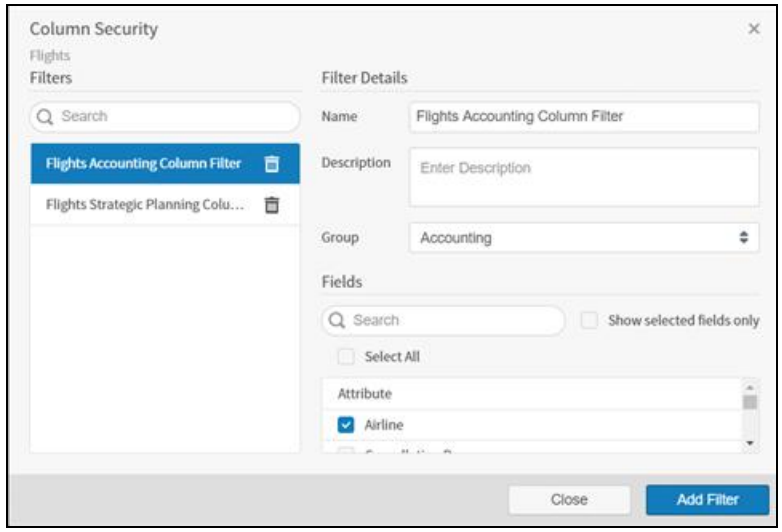
Modify column security to restrict the data source fields that can be viewed or used by the group.

1. Log into Composer as a user in a group that has been granted the **Administer Sources** privilege, or a user in a group that has been granted the **Manage Source Permissions** privilege *and* who also has **read** permission for the data source.

   If the user name you log in with is also associated with other Composer accounts, verify that the correct account is selected. See Switch Tenants.

2. Select **Sources** on the top-level navigation banner or in the UI menu ( ) The Sources page appears.

3. Locate the data source and select in the **Column** column for the data source. The Column Security dialog appears.



4. To modify a column security definition, select it on the left side of the Column Security dialog. The settings for the definition appear in the Filter Details on the right side of the dialog and can be modified.

5. Modify any of the information for the column security definition, as described in Add Column Security Definitions. When you are finished, select **Save** to save the column security settings.

6. When all column security definition modifications have been made, select **Close** to close the Column Security dialog.

Remove Column Security Definitions

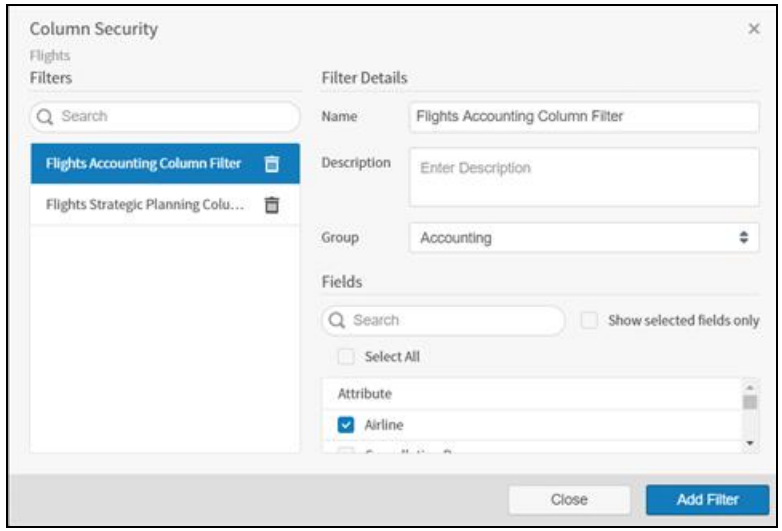
Remove column security for a data source

1. Log into Composer as a user in a group that has been granted the **Administer Sources** privilege, or a user in a group that has been granted the **Manage Source Permissions** privilege *and* who also has **read** permission for the data source.

   If the user name you log in with is also associated with other Composer accounts, verify that the correct account is selected. See Switch Tenants.

2. Select **Sources** on the top-level navigation banner or in the UI menu ( ). The Sources page appears.

3. Locate the data source and select in the **Column** column for the data source. The Column Security dialog appears.



4. Locate the column security definition you want to remove (delete) on the left side of the Column Security dialog and select next to its name.

5. Select **Delete** on the confirmation dialog. The column security definition is removed.

6. When all column security definition modifications have been made, select **Close** to close the Column Security dialog.

Restrict Access to Data Using Row Security

Users with appropriate permissions can manually restrict the data in a data source configuration that can be viewed or used by group, tenant, or user. By default, all data in a data source is available.

Row security filters allow you to secure potentially confidential data within a data source. Selected users or group or account members would only be able to view limited information within the data it collects.

Row security filters can be maintained for a data source by:

- an administrator.
- User in a group that has been granted the **Administer Sources** privilege.
- User in a group that has been granted the **Manage Source Permissions** privilege who also has **read** permission for the data source.

Security filters will not be applied to users with the privileges mentioned above. Source administrators can manage security filters for regular users but not for other source administrators.

If a user is included in more than one row security filter for the same data source (via group, user, or account specifications), an error message appears when the user tries to view a dashboard using the data source. This occurs because of the row restriction conflicts set by the different row security filters. Note that if a user is included more than once in a single row security filter (either as an explicit user, a member of more than one group, or as a member of the account), no error occurs because it is a single row security filter.

A "Data Unavailable" message appears when an error occurs obtaining data from a data source for a row security filter. In addition, if a row security filter returns no results, a "No search results" message appears.

Row security is supported by the API endpoint `/api/sources/<source-id>/security/filters`.

API documentation is provided with your Composer installation at this link: *`https://<composer-URL>/composer/swagger-ui.html`*.

This section covers the following topics:

- Add Row Security Definitions
- Modify Row Security Definitions
- Remove Row Security Definitions

Add Row Security Definitions

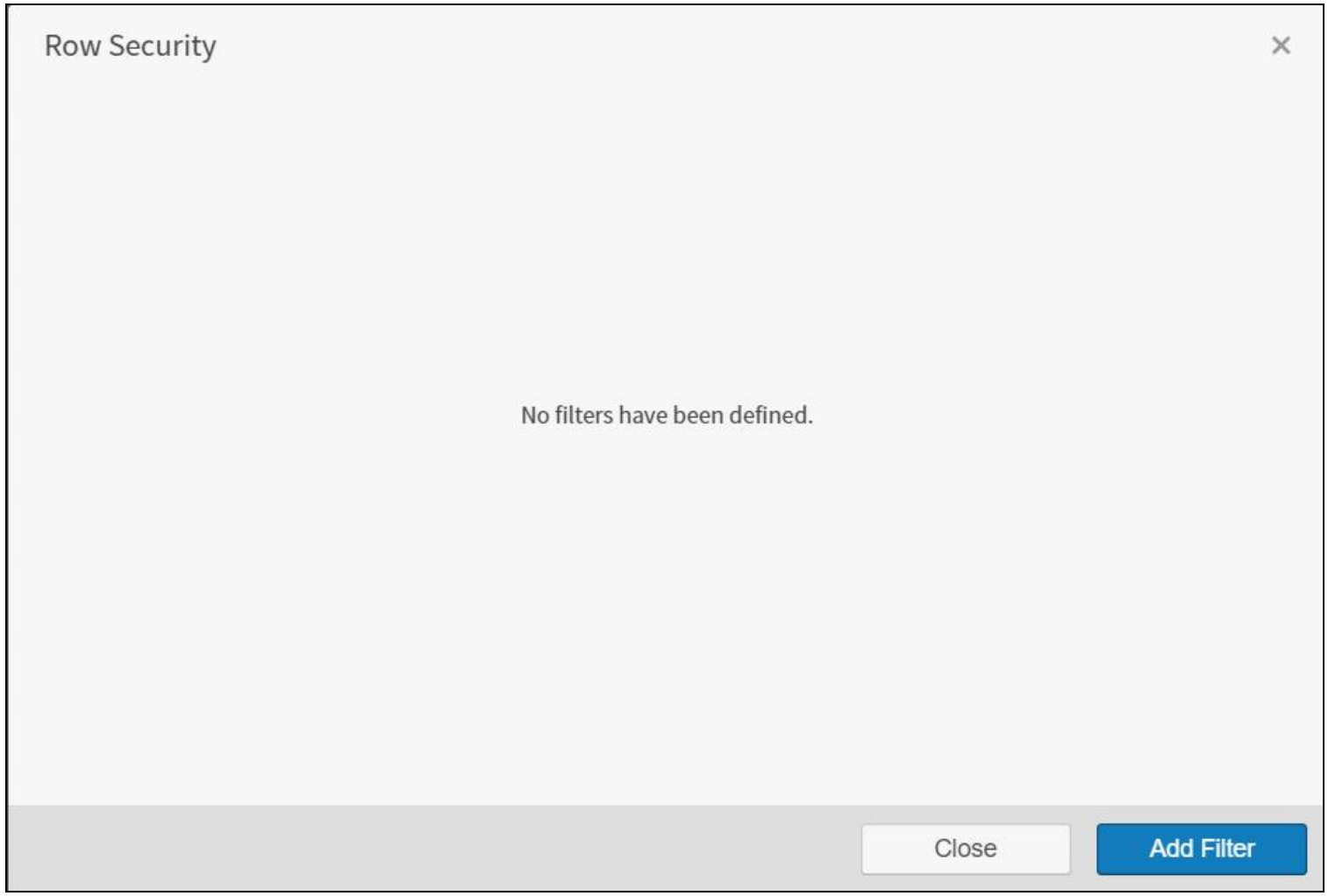
Add row security to restrict the data source data that can be viewed or used by a group, user, or account.

1. Log into Composer as a user in a group that has been granted the **Administer Sources** privilege, or a user in a group that has been granted the **Manage Source Permissions** privilege *and* who also has **read** permission for the data source.

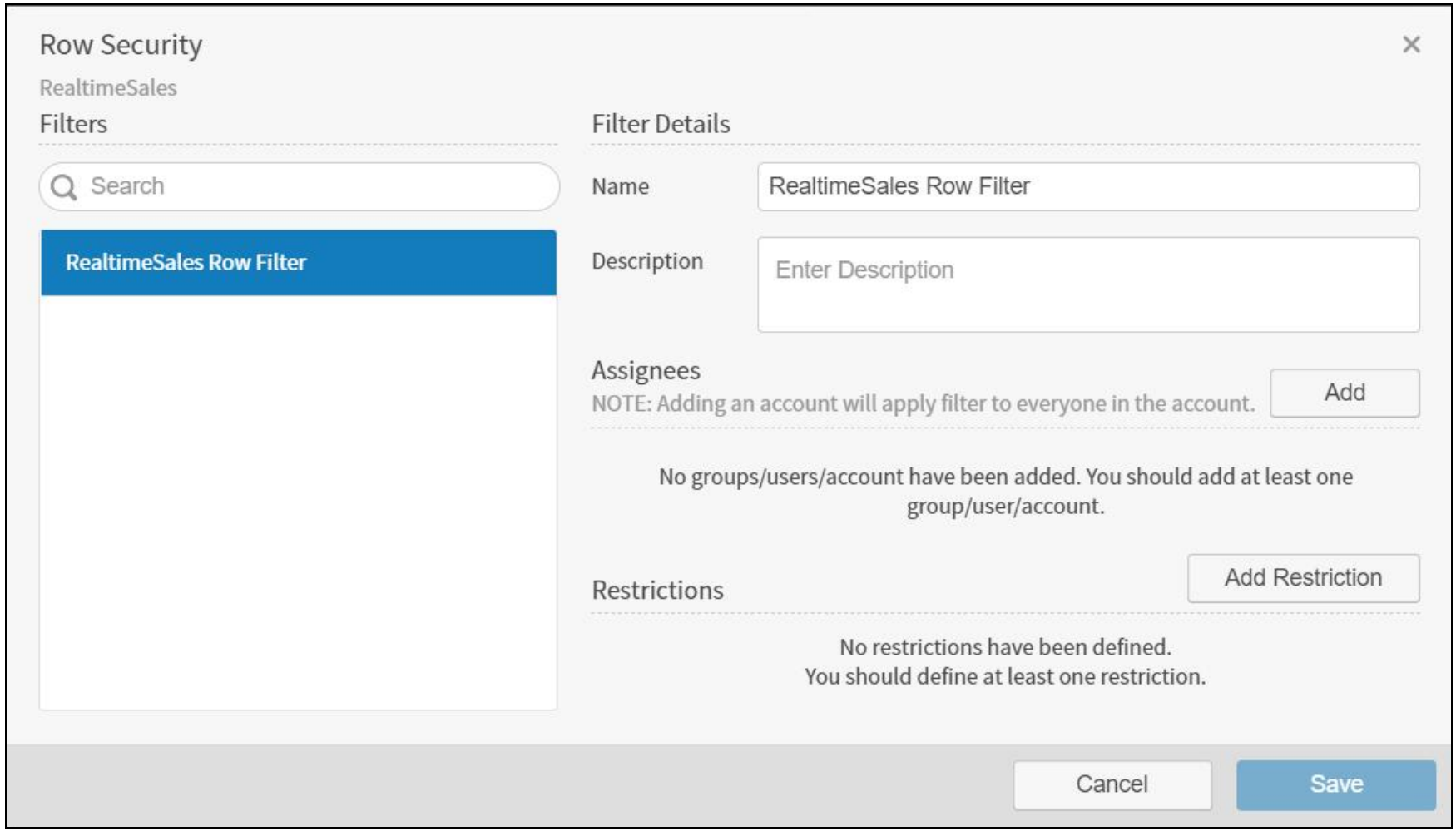
   If the user name you log in with is also associated with other Composer accounts, verify that the correct account is selected. See Switch Tenants.

2. Select **Sources** on the top-level navigation banner or in the UI menu ( ). The Sources page appears.

3. Locate the data source for which you want to restrict data access and select in the **Row** column for the data source. The Row Security dialog appears.



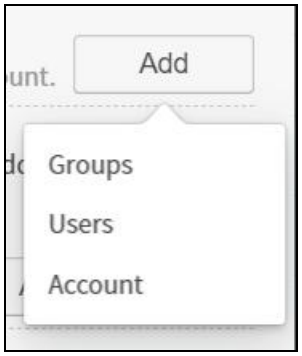
4. Select **Add Filter**. The Row Security dialog fills with information about the data source you selected.



5. Specify a name for the row security definition in the **Name** field. This name will be used to distinguish one row security field definition from another in the Row Security dialog.

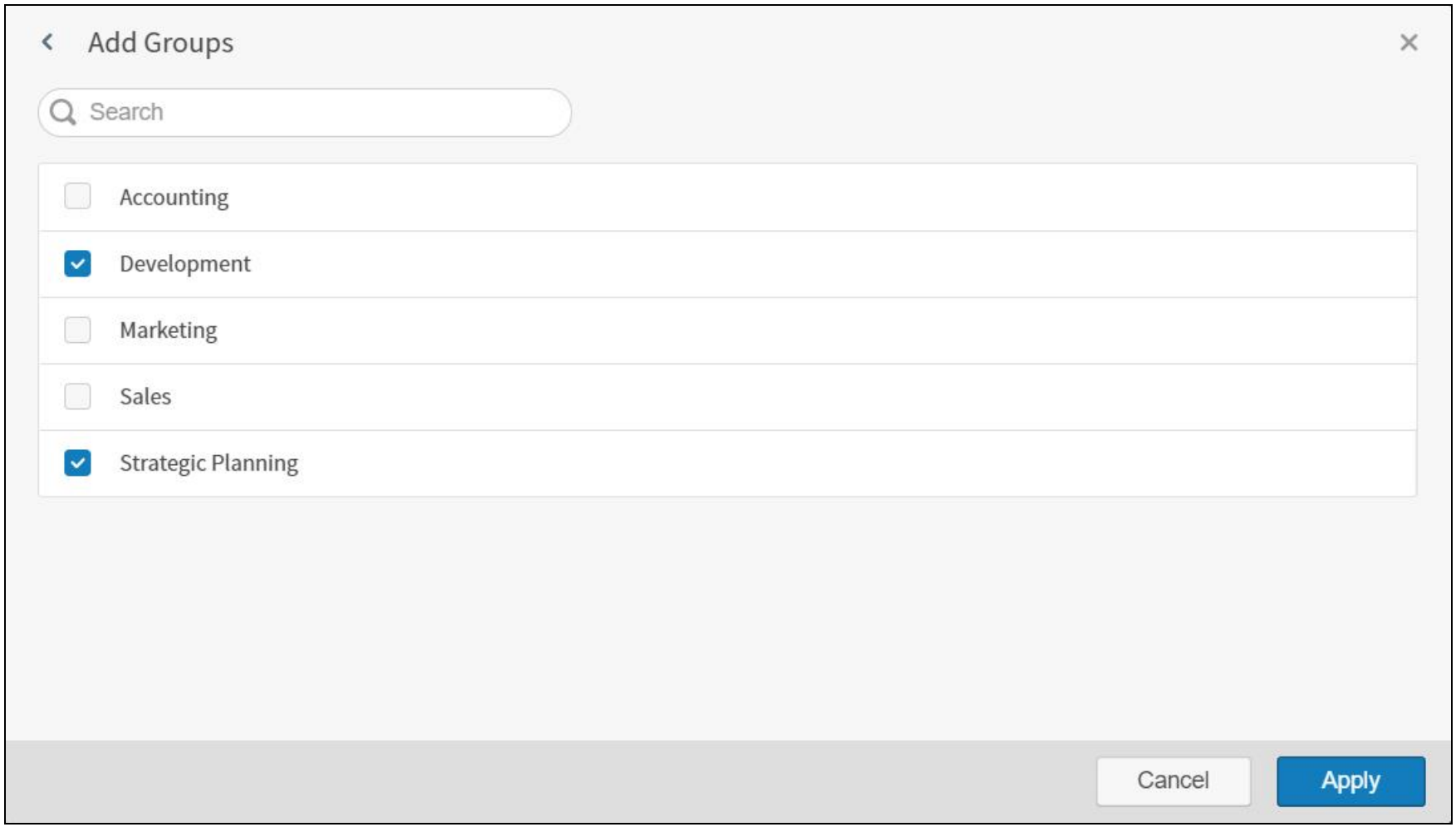
6. Optionally, use the **Description** field to supply a description for the row security definition.

7. Select **Add** to select accounts, groups, or users to which the row security definition applies. Then select **Groups**, **Users**, or **Account** from the drop-down list.



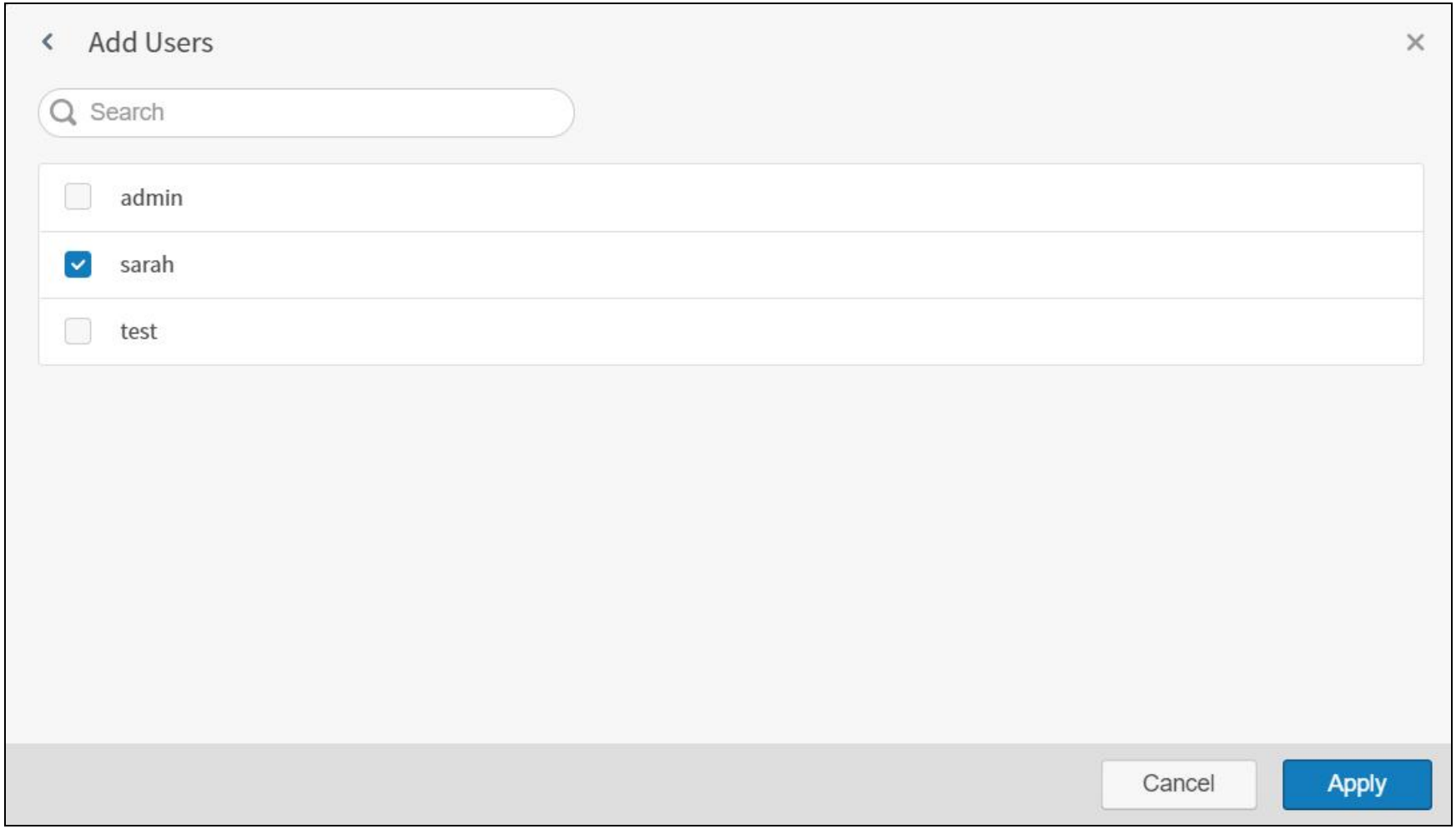
The behavior of the dialog varies depending on what you select.

i. If you select **Groups**, the Add Groups panel appears.



Select at least one group on the Add Groups panel and select **Apply**. The Add Groups panel closes and the groups you selected are added to the Row Security dialog.

ii. If you select **Users**, the Add Users panel appears.

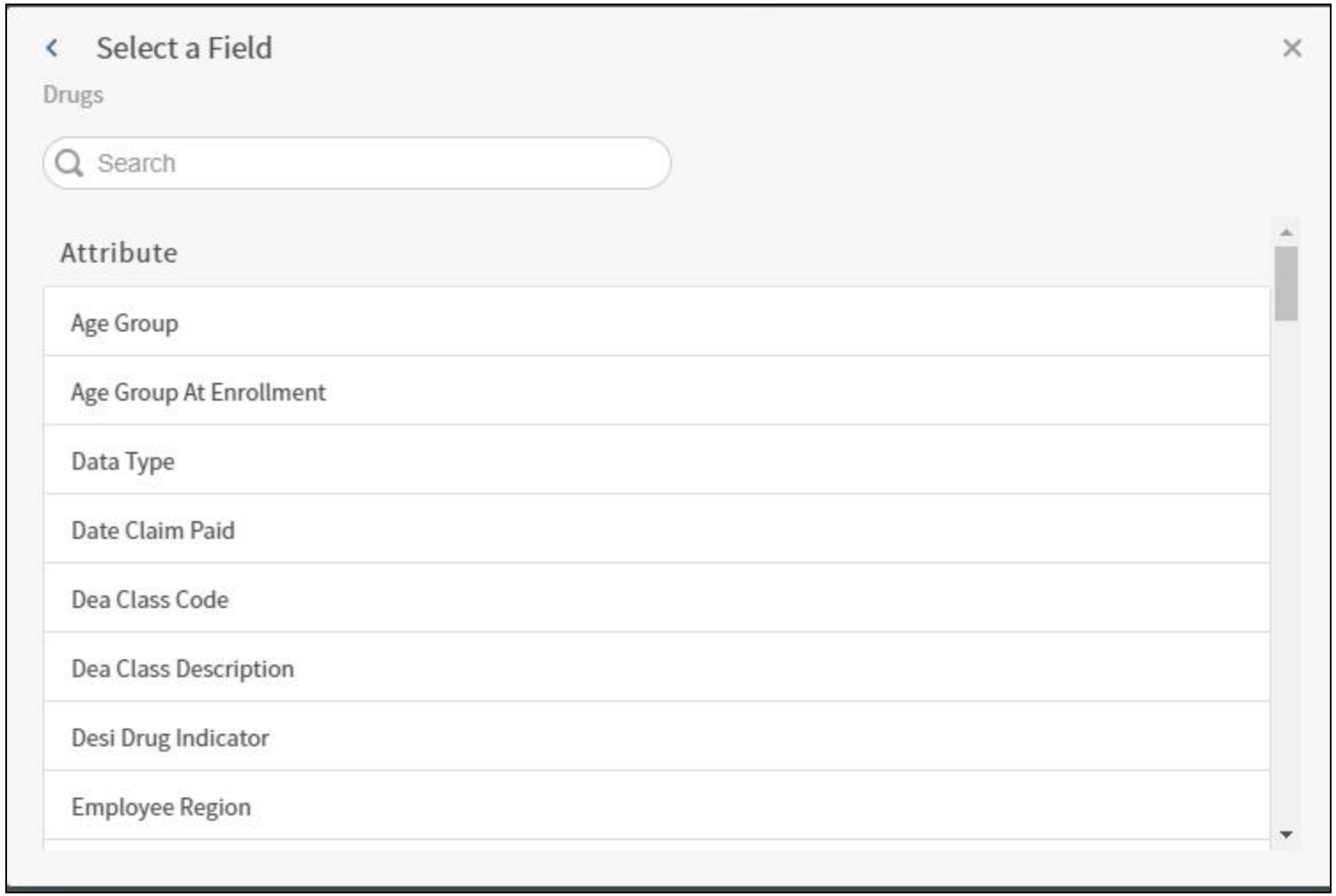


Select at least one user on the Add Users panel and select **Apply**. The Add Users panel closes and the users you selected are added to the Row Security dialog.

iii. If you select **Account**, the account in which you are working is added to the Row Security dialog. You can only add your current account to the row security definition. After that, the **Account** option is disabled.

8. Repeat Step 7 until all users, accounts, and groups are selected for the row security filter.

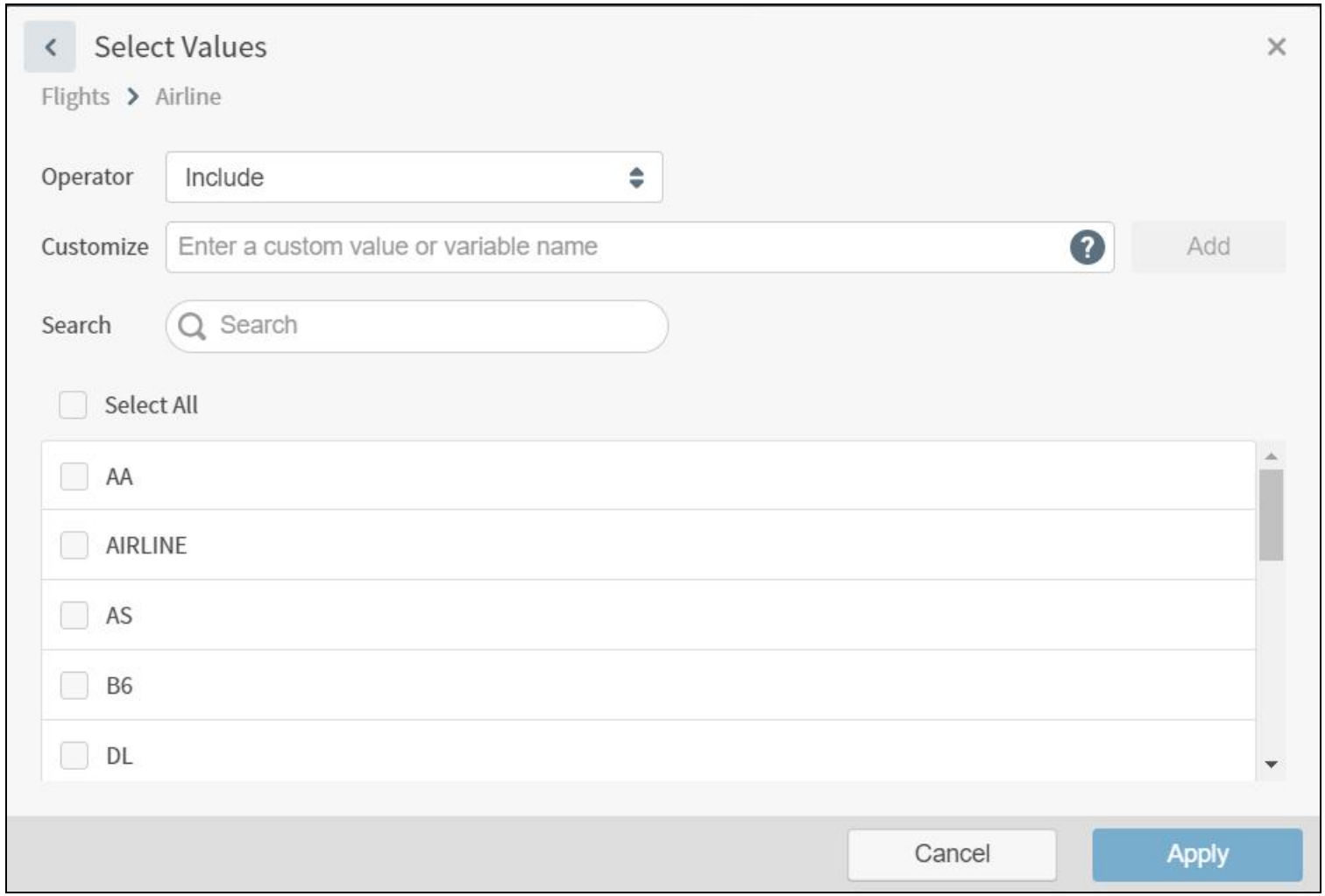
9. Select **Add Restriction** to add at least one restriction to the row security definition. The Select a Field dialog appears.



10. Select a field for the restriction filter on the Select a Field dialog. A Select Values dialog appears with fields that vary, depending on the type of field you selected: attribute, number, or time. Derived fields are included in the list of fields and can be selected for a row security filter.

Attribute Fields

If the field you selected is an attribute, the Select Values dialog looks something like this:



It allows you to:

i. Select the filter operator (**Include** or **Exclude**, depending on whether you want to include or exclude the value from the data).

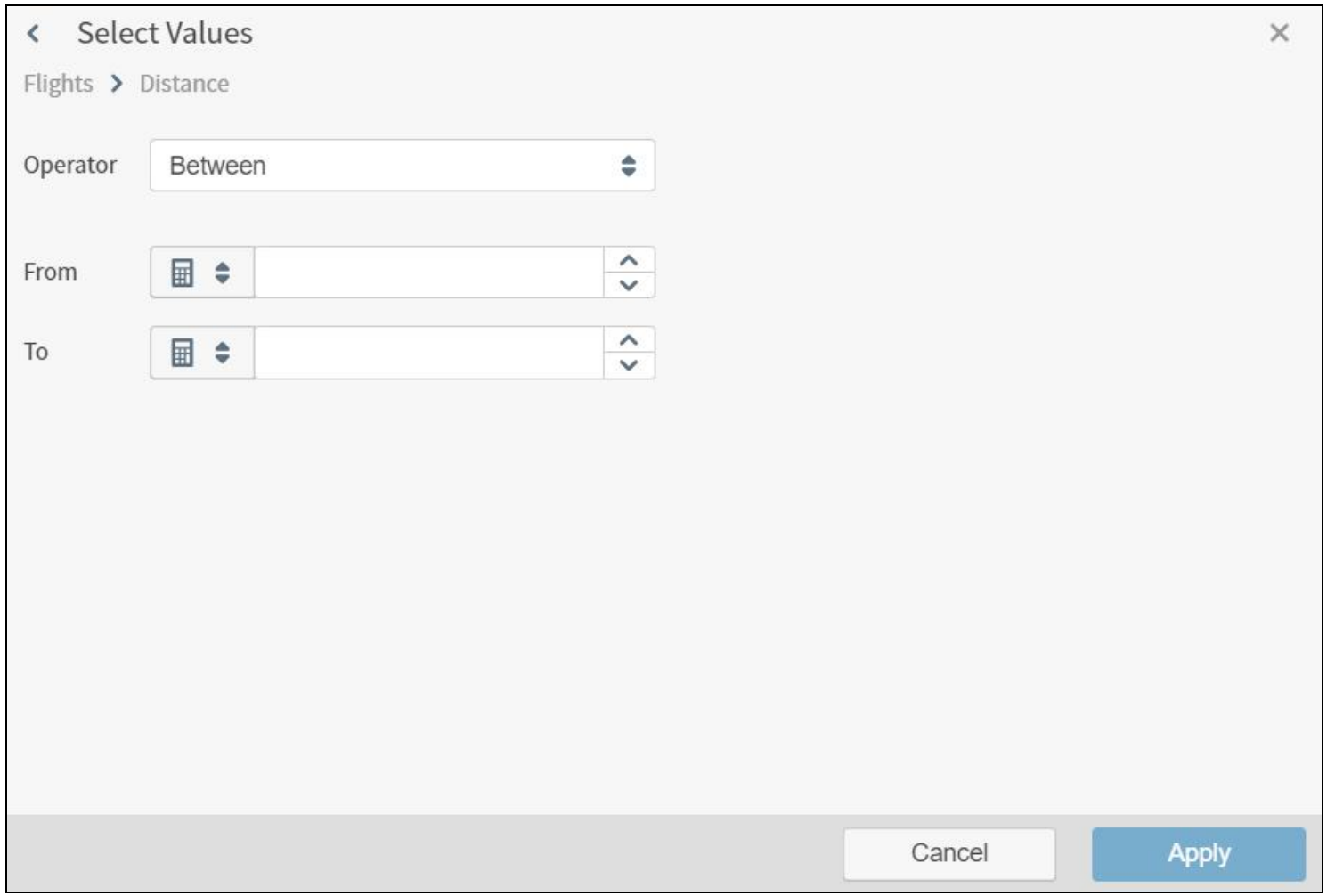
ii. Specify an optional custom value. To create and select a custom value, enter the value in the **Customize** field and select **Add**. Your custom field is added and selected in the list of possible values. To remove the custom value, uncheck it in the list of possible values. It is removed from the filter and from the list of

possible values for the filter. You can insert variables as values for the attribute filter. See Insert Variables For Row Security Restriction Filters.

iii. Select one or more values from the list of available values for the attribute you selected. To select all values, select **Select All**.

Number Fields

If the field you select is a numeric field, the Select Values dialog looks something like this:



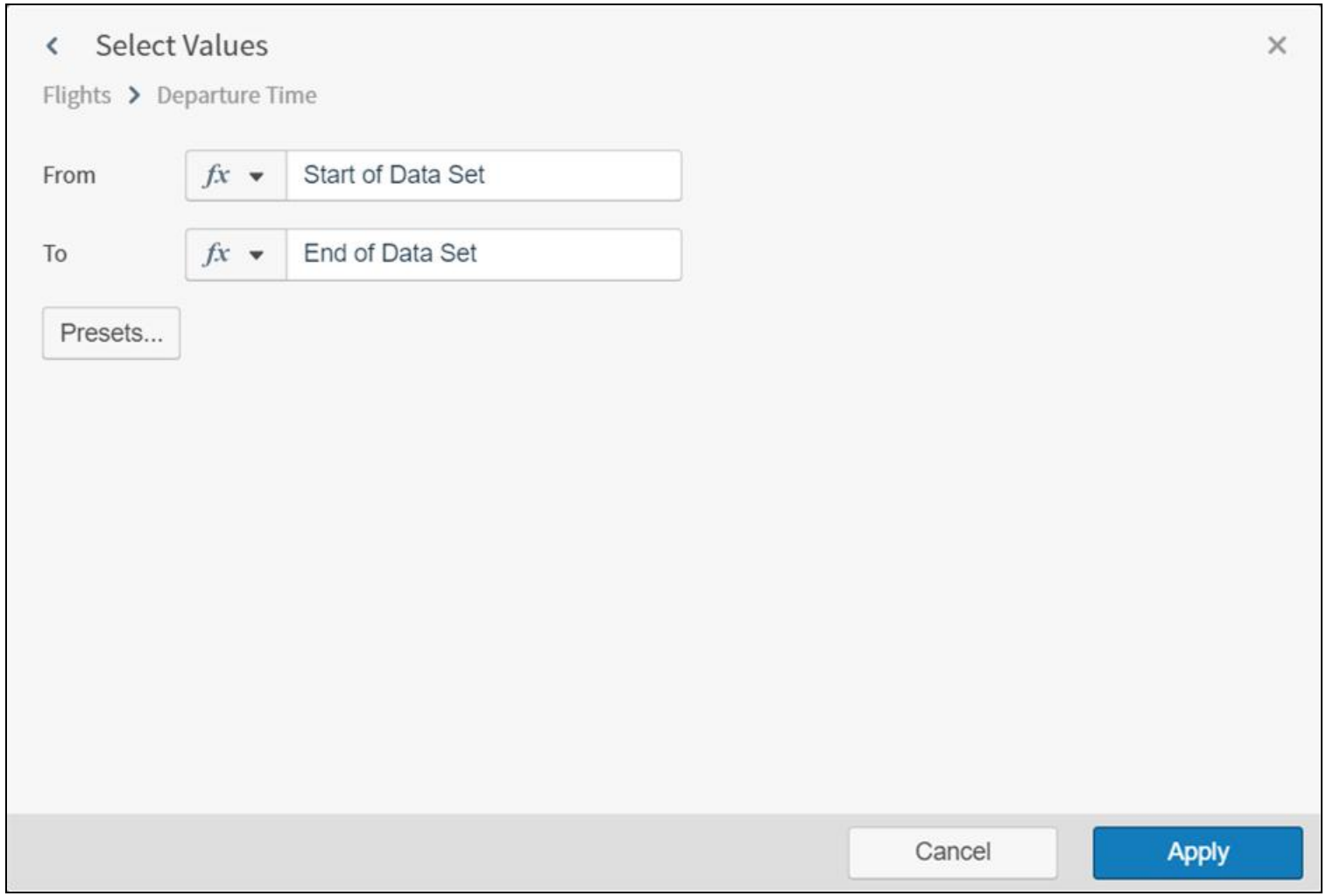
It allows you to:

i. Select a relational comparison operator in the **Operator** selection box. Data is included in the visual when the data in the filter field meets the condition set by the relational operator and the numeric values you specify. Valid numeric operators are described in Operators.

ii. Use the arrows in the **From** and **To** boxes to increase and decrease the maximum and minimum values. You can insert variables as values for the numeric filter. See Insert Variables For Row Security Restriction Filters.

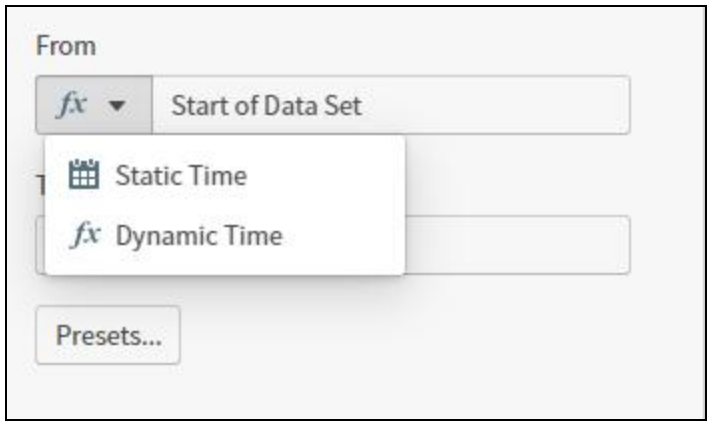
Time Fields

If the field you select is a time field, the Select Values dialog looks something like this:



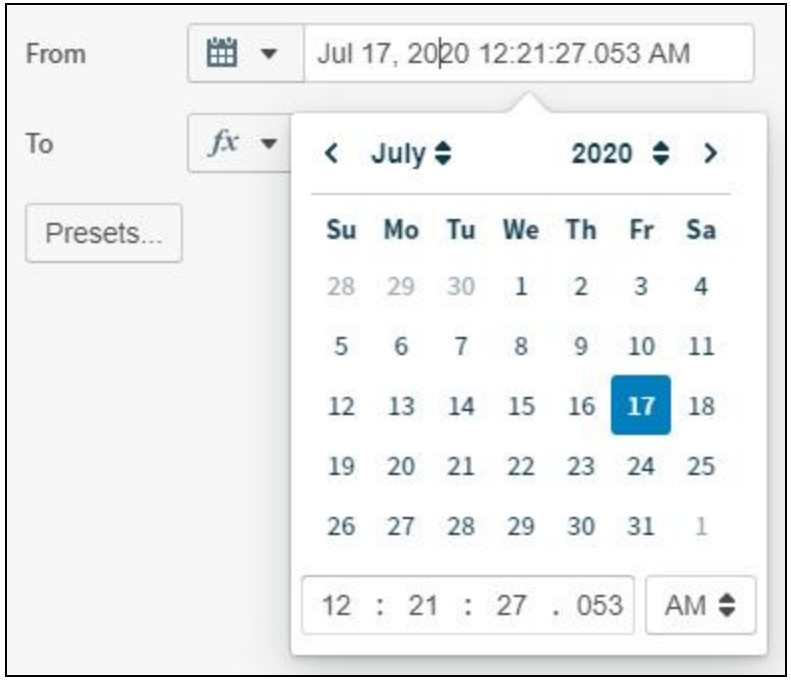
It allows you to use the **From** and **To** boxes to specify the time range for the filter. You can set the range in static time or dynamic time, or use preset ranges provided with Composer.

i. Select **Static Time**, **Dynamic Time**, or **Variables** in the **fx** drop-down menu.

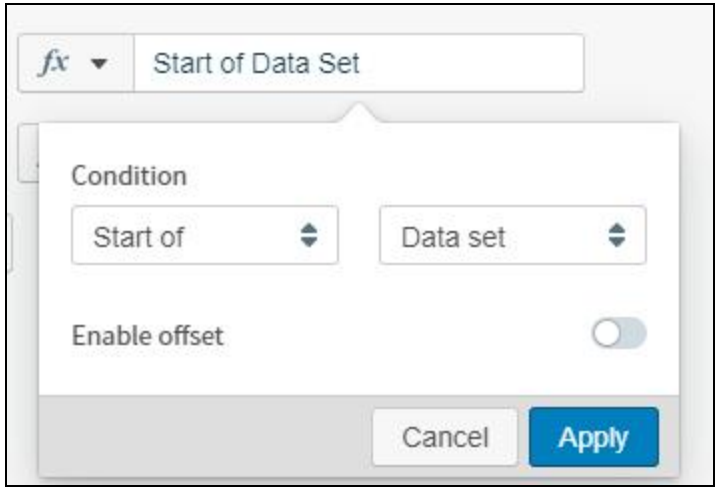


If you select **Variables**, specify a variable to use for the time filter value. See Insert Variables For Row Security Restriction Filters.

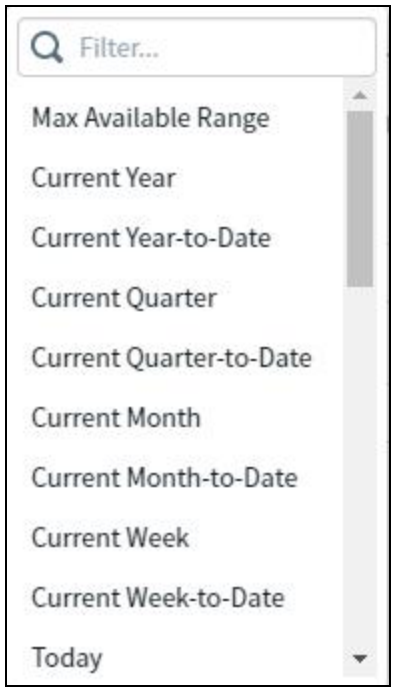
If you select **Static Time**, the **From** and **To** boxes are filled with default dates and times. Use the boxes to select specific from and to times.



If you select **Dynamic Time**, the **From** and **To** boxes are filled with **Start of Data Set** and **End of Data Set** automatically and a Condition dialog appears. Use the boxes on the Condition dialog to select different dynamic from and to times:



ii. Alternatively, select Presets... to fill the **From** and **To** boxes with predefined time ranges provided by Composer.



Use the filter box at the top of the presets list to locate the preset setting you want. Descriptions of each of the preset options are provided in Preset Time Ranges.

11. To remove a restriction from the security definition, select ⊖ next to the restriction.
12. Repeat Steps 9-10 if additional restrictions are needed. All restrictions for a row security definition are listed on the Row Security dialog.
13. When you are finished data restrictions for the group, select **Save** to save the row security definition.
14. Repeat Steps 4-13 to add more row security definitions for the data source.
15. When all row security definition modifications have been made, select **Close** to close the Row Security dialog.

Modify Row Security Definitions

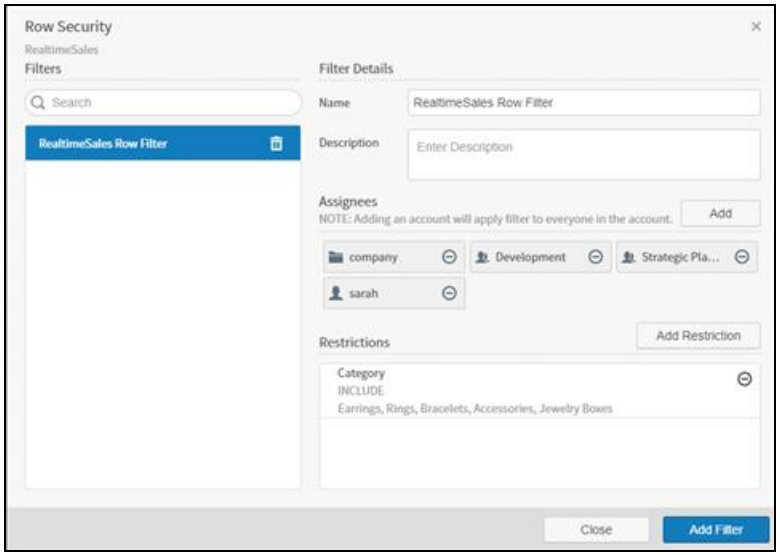
Modify row security for a data source

1. Log into Composer as a user in a group that has been granted the **Administer Sources** privilege, or a user in a group that has been granted the **Manage Source Permissions** privilege *and* who also has **read** permission for the data source.

   If the user name you log in with is also associated with other Composer accounts, verify that the correct account is selected. See Switch Tenants.

2. Select **Sources** on the top-level navigation banner or in the UI menu ( ). The Sources page appears.

3. Locate the data source for which you want to restrict data access and select in the **Row** column for the data source. The Row Security dialog appears.



4. To modify a row security definition, select it on the left side of the Row Security dialog. The settings for the definition appear in the Filter Details on the right side of the dialog and can be modified.

5. Modify any of the information for the row security definition, as described in Add Row Security Definitions. When you are finished, select **Save** to save the row security settings.

6. When all row security definition modifications have been made, select **Close** to close the Row Security dialog.

Remove Row Security Definitions

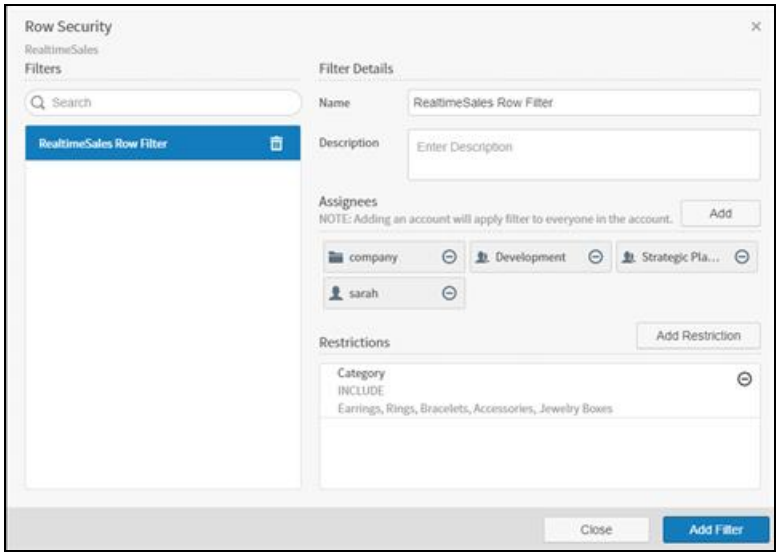
Remove row security for a data source

1. Log into Composer as a user in a group that has been granted the **Administer Sources** privilege, or a user in a group that has been granted the **Manage Source Permissions** privilege *and* who also has **read** permission for the data source.

   If the user name you log in with is also associated with other Composer accounts, verify that the correct account is selected. See Switch Tenants.

2. Select **Sources** on the top-level navigation banner or in the UI menu ( ). The Sources page appears.

3. Locate the data source and select in the **Row** column for the data source. The Row Security dialog appears.



4. Locate the row security definition you want to remove (delete) on the left side of the Row Security dialog and select next to its name.

5. Select **Delete** on the confirmation dialog. The row security definition is removed.

6. When all row security definition modifications have been made, select **Close** to close the Row Security dialog.

Insert Variables for Row Security Restriction Filters

Variables can be inserted as values for any restriction filter in a row security definition. The variables are passed to the connection string via custom attributes specified in the user definition or dynamically in the custom attributes specified in the SAML or LDAP configurations for your Composer installation.

You can also specify user attributes for use in the connection parameters of a connection definition. See Use User Attributes for Connection Parameters.

Note: If a you use a variable in a row security definition, but don't define a corresponding custom attribute the user, an error message appears when the user attempts to view a dashboard on which the row security is applied.

Step 1: Define Custom Attributes for the Variables

A custom attribute must be defined for every variable you want to use. The only exceptions are the Composer context variables `${User.composerUserName}`, `${User.accountId}`, and `${User.credentials}`. These built-in attributes which automatically exist and can be used connect the currently logged in user.

You can define custom attributes in several ways:

- Individually for every user. If you use this method, the variable names must be the same for every user. See Specify Custom User Attributes.
- Dynamically in the LDAP or SAML configurations for your Composer instance. See Use Lightweight Directory Access Protocol (LDAP) With ComposerSymphony and Configure ComposerSymphony To Support SAML.

Details about specifying custom attribute values are provided in Specify Custom User Attributes.

Step 2: Using Variables in Row Security

Use variables in row security

1. Log into Composer as a user in a group that has been granted the **Administer Sources** privilege, or a user in a group that has been granted the **Manage Source Permissions** privilege *and* who also has **read** permission for the data source.
2. Follow the instructions in Restrict Access To Data Using Row Security to add or modify a row security definition. When you get to the step where you select values for the restriction filter, specify the custom attribute (variable) you defined in Step 1 as a value for the filter. If custom attributes are defined, they can be directly entered using the following syntax:

```
${User.<custom-attribute-name>}
```

3. Save the row security definition and close the Row Security dialog, as described in Restrict Access to Data Using Row Security.

Row Security Filter Errors

When a custom user attribute, used as a variable in a row security filter, is invalid (for example, it cannot be parsed as a value of a required type), a generic error message is given and a detailed message is logged describing what is wrong with the row security filter.

Use Materialized Views (Experimental)

Note: Materialized view functionality is disabled by default. To enable, contact technical support for assistance. The Materialized Views API is deprecated and will be removed in a future release.

Materialized views allow you to use pre-aggregated query results to speed up query processing in certain scenarios, especially when processing a query with heavily aggregated data. This boosts your visual rendering time.

Important: This is an experimental feature.

Materialized views only work if the metrics and groups in the materialized view definitions match those used by your visuals. For example, no benefit from using materialized views will occur if you define a materialized view for the Planned Sales metric in a data source, but none of the visuals in your environment use the Planned Sales metric. Consequently, it is a good idea to understand what data in your data source is used (or will be used) in visuals before you define a materialized view.

Note: A Count metric is required in all materialized view definitions.

Support for materialized views is performed using the REST API endpoint `/api/materialized-views`, described in Materialized Views API (Experimental). You can also specify materialized view settings for a data source configuration using the UI. See the following topics:

- List Materialized View Definitions
- Add A Materialized View Definition
- Edit A Materialized View Definition
- Delete A Materialized View Definition
- Enable And Disable Materialized View Definitions

List Materialized View Definitions

Note: Materialized view functionality is disabled by default. To enable, contact technical support for assistance. The Materialized Views API is deprecated and will be removed in a future release.

Important: This is an experimental feature.

List the materialized view definitions for a data source configuration

1. Make sure you are logged in as a user with the **Administer Sources** privilege or that you have **read** and **write** permission for the data source.
2. Select **Sources** on the UI menu () or the top-level navigation menu, or select the **Sources** box on the Home page. The Sources page appears.
3. Locate the data source configuration on the Sources page and select the More menu () icon in the **Actions** column to view the **More** menu.
4. Select **Materialized Views** from the menu.

   The Materialized View dialog appears, listing the materialized views defined for the data source configuration.

Add a Materialized View Definition

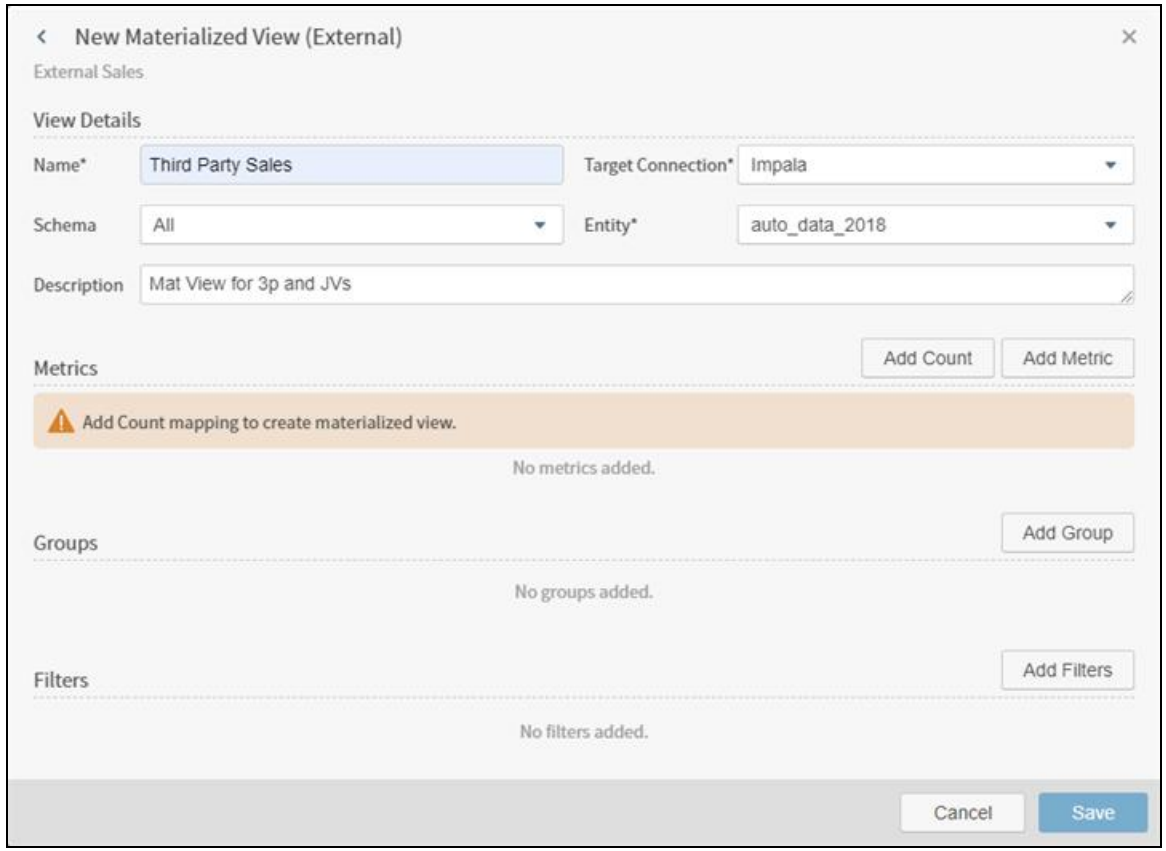
Note: Materialized view functionality is disabled by default. To enable, contact technical support for assistance. The Materialized Views API is deprecated and will be removed in a future release.

Important: This is an experimental feature.

Important: Pre-aggregated data is not managed by Composer, so it should be maintained manually and kept up to date by the owner of the data.

Add a materialized view definition to a data source using the UI

1. Make sure you are logged in as a user with the **Administer Sources** privilege or that you have **read** and **write** permission for the data source. If you create in Composer, you have **read**, **write**, and **delete** permissions for that source by default.
2. List the materialized views for the data source to which you want to add a materialized view. See List Materialized View Definitions.
3. Select **Add View**. The **New Materialized View (External)** dialog appears. This dialog is used to identify where the aggregated result from a query is stored so it can be quickly recalled in visuals.



4. Specify a name for the materialized view in the **Name** field. This is required.

5. Select an external target connection in the **Target Connection** field. This is required. The connections listed for this field match the list of connections defined for the Composer instance. The target connection is not required to be the same as the connection of the data source; it can be any other connection configured in Composer.

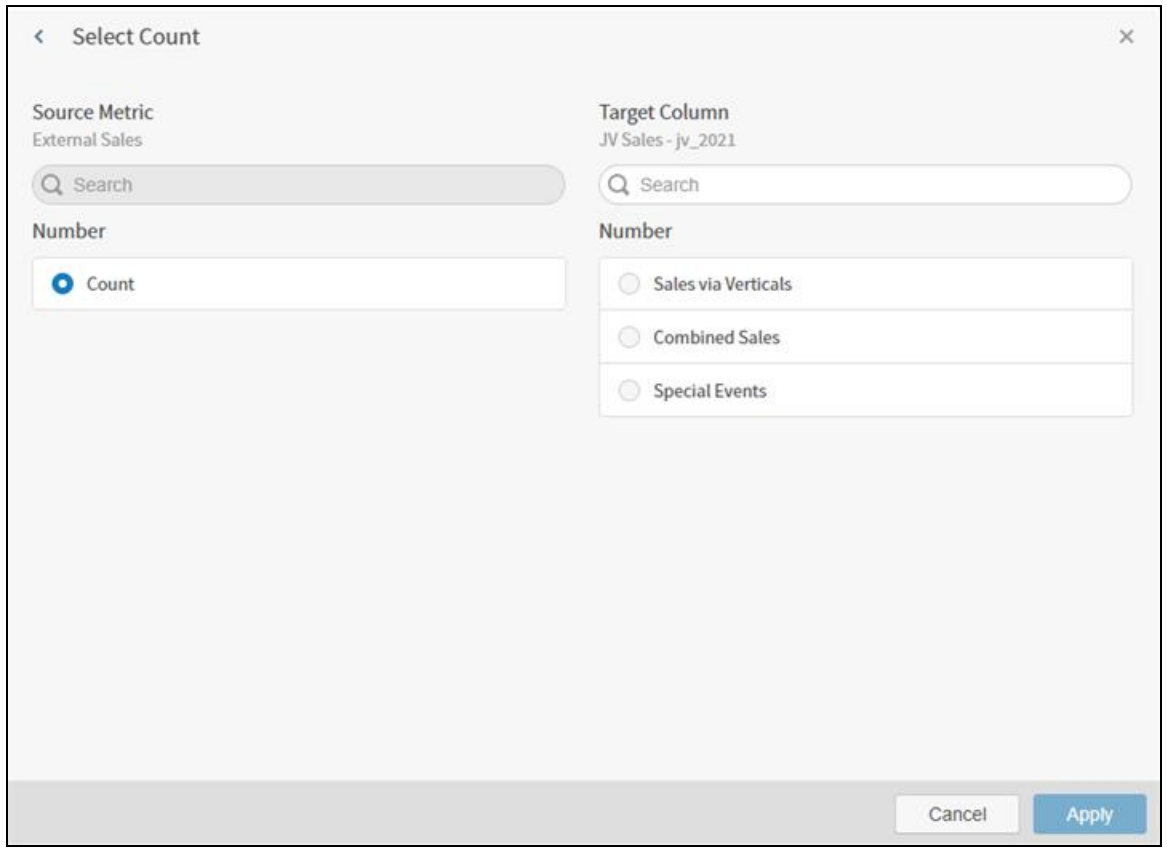
   Note: If you do not have read permissions for the target data store, an error message appears.

6. Use the **Schema** and **Entity** fields to identify the target entity that will store the aggregated data. The target entity is required, while the schema is not. However, selecting a schema filters the list of entities so you can more quickly find the entity you want.

   As soon as an entity is specified, the **Add Count**, **Add Metric**, **Add Group**, and **Add Filters** buttons become available on the dialog.

7. Optionally supply a description in the **Description** field.

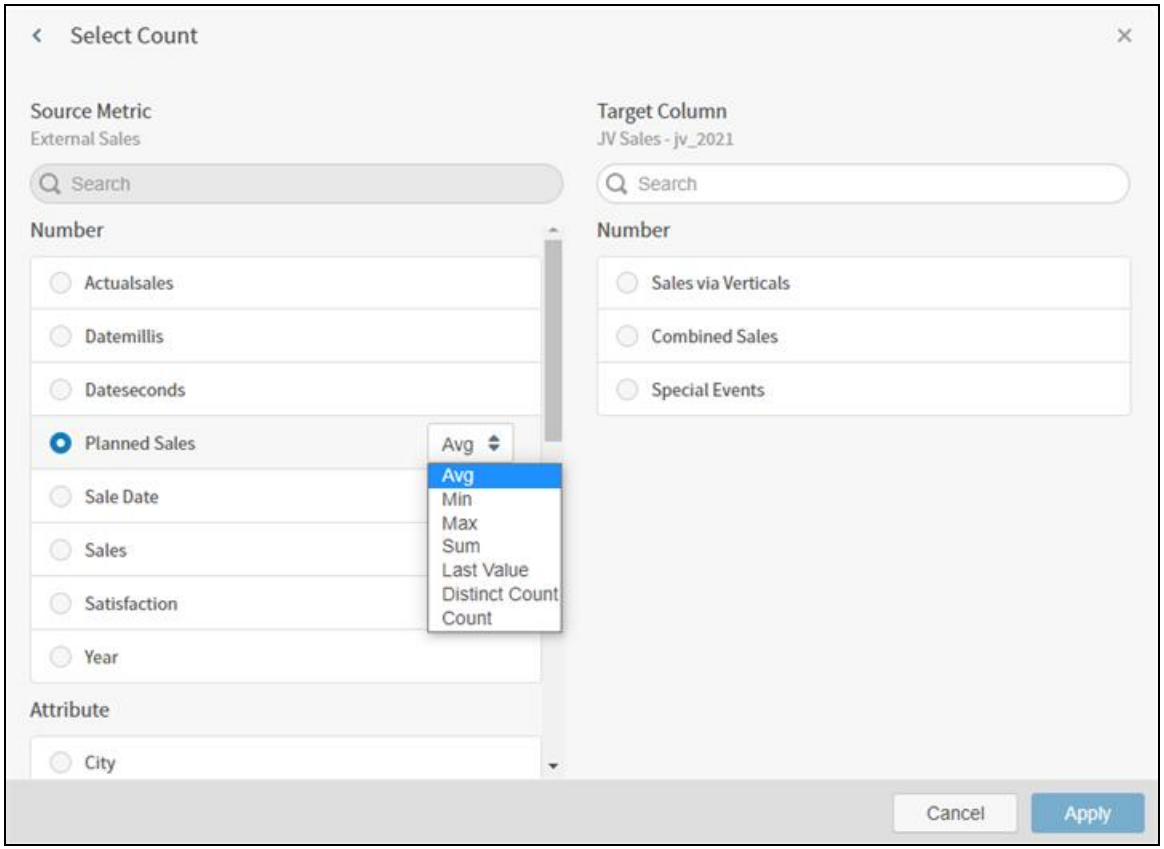
8. Select **Add Count** to open the Select Count work area. Select a source field in the Source Metric list and a Target Column to list which source field should be used, then select **Apply** to apply your changes.



9. Select **Add Metrics** to add metrics to your materialized view definition. The Count metric must be added because it is required. Other metrics are optional, but must match the metrics used in your visuals.

> **Note:** Materialized views only work if you specify the metrics and groups used by your visuals in the materialized view definition. For example, if you use Sales (SUM) as your metric and State as your group in a visual, be sure to add these metrics and groups to your materialized view definition. If they don't match, Composer will not use the materialized view to boost your visual rendering time.

The Select Metric dialog appears.



Select a source field in the Source Metric list and a target column in the Target Column list to which the source field should be mapped. Aggregated data for the source field will be taken from the target column to provide the results of the query.

When you select a source field (other than Volume), you can also select a metric function to be used to aggregate the field data. Use the drop-down list to select the metric function. See Metric Aggregation Functions.

Use the search boxes at the top of the Source Metric and Target Column to quickly locate a source metric or target column in their lists.

Select **Apply** to apply the mapping to the materialized view definition.

Repeat this step for as many metrics as needed for the definition.

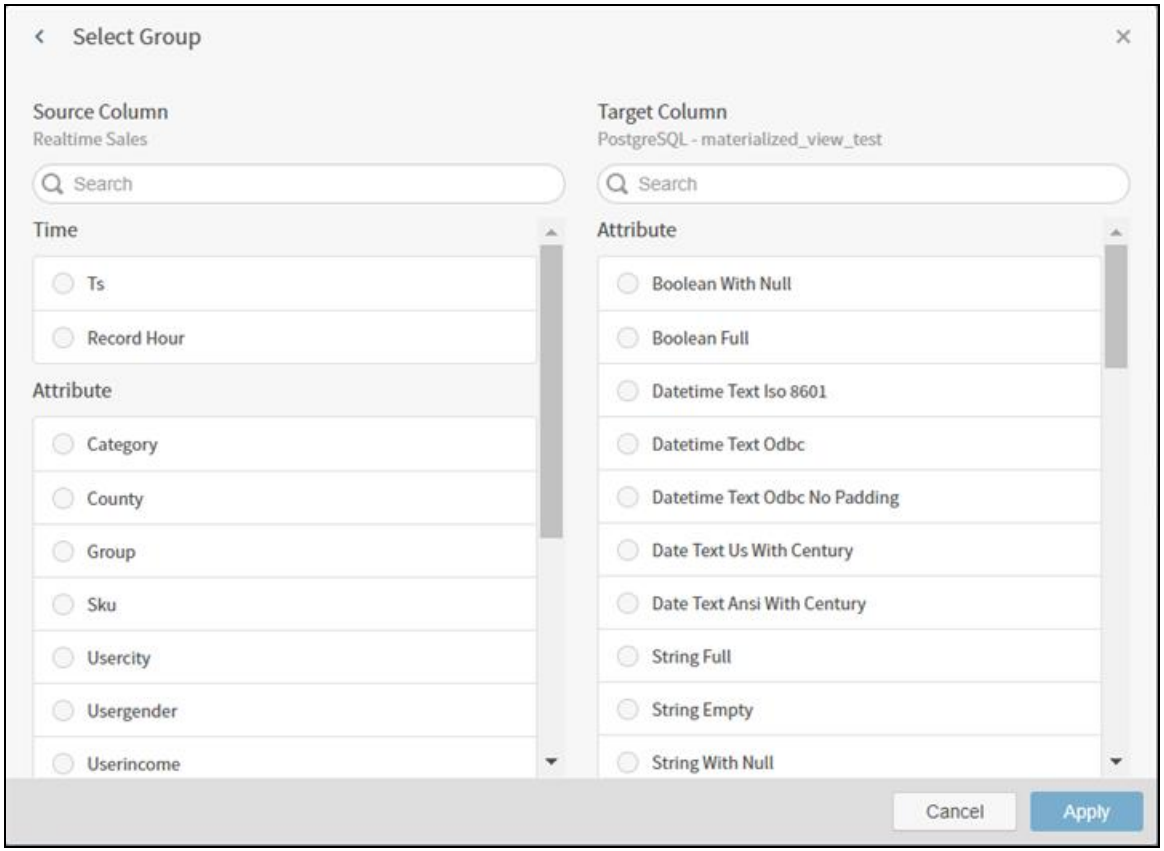
Note: If you have two visuals that differ only in their metrics, and you have an externally stored table containing pre-aggregated data for both visuals, you can specify all of the metrics for both visuals in a single materialized view. When the visuals are rendered, both visuals will be matched by the same materialized view.

10. Select **Add Groups** to add groups to your materialized view definition. The groups must match the groups used in your visuals.

Important: Be sure that the target entity contains correct data for this materialized view. If there are more pre-aggregated group columns in the target entity than configured groups in the materialized view, the data for the query will be taken as-is from the target entity. This might result in incorrect data shown on the visual, (for example, non-unique group values and incorrect metrics).

Note: Materialized views only work if you specify the metrics and groups used by your visuals in the materialized view definition. For example, if you use Sales (SUM) as your metric and State as your group in a visual, be sure to add these metrics and groups to your materialized view definition. If they don't match, Composer will not use the materialized view to boost your visual rendering time.

The Select Group dialog appears.



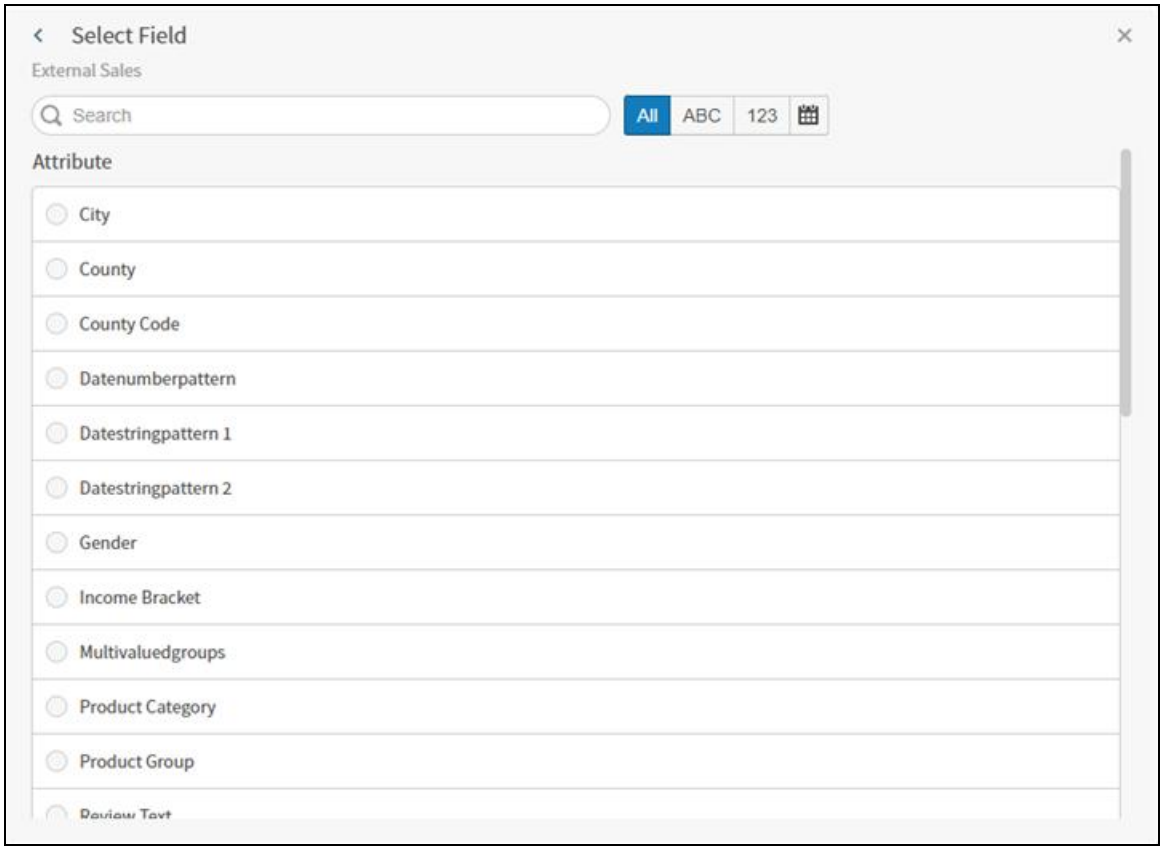
Select a source field in the Source Column list and a target column in the Target Column list to which the source field should be mapped. Aggregated data for the source field will be taken from the target column to provide the results of the query.

Use the search boxes at the top of the Source Column and Target Column to quickly locate a source field or target column in their lists.

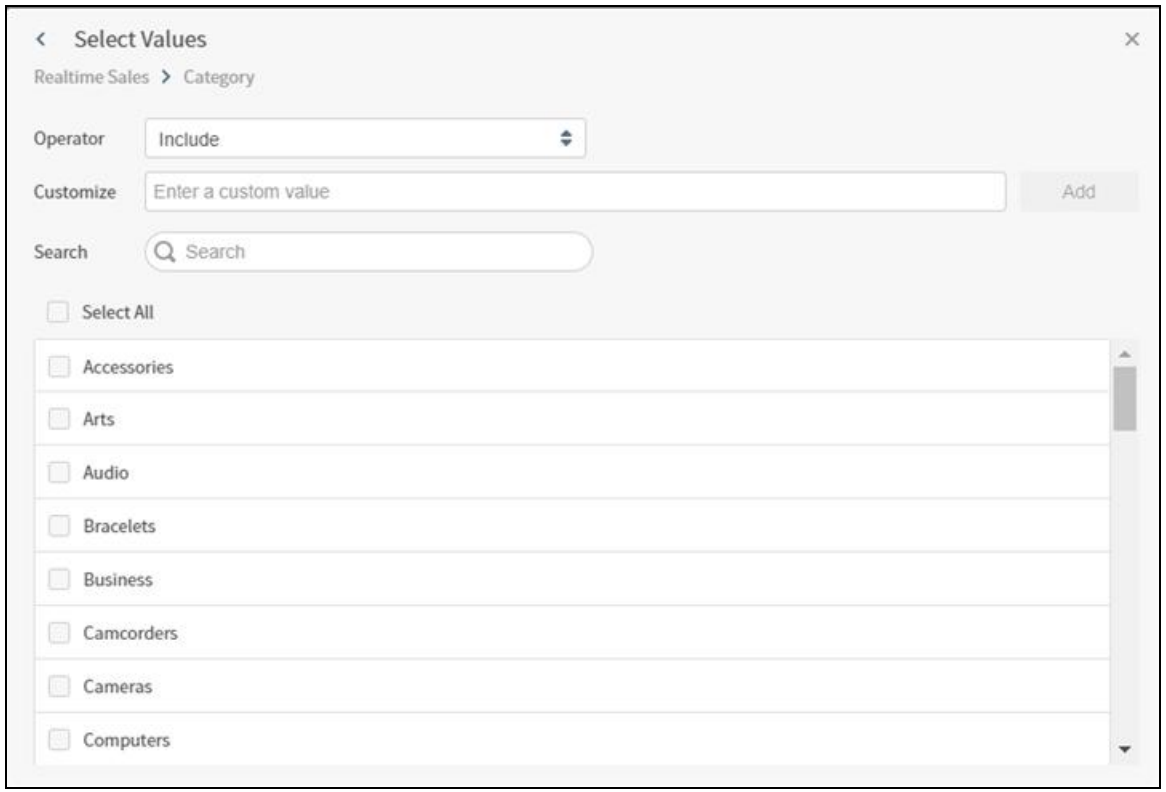
Select **Apply** to apply the mapping to the materialized view definition. Repeat this step for as many groups as needed for the definition.

11. Select **Add Filters** to add filters for the data that is stored in the target entity. Only requests that match the specified filters in the materialized view definition will be processed using the data from this materialized view.

    The Select Field dialog appears.


Select Field
External Sales
Search
All
ABC
123
Attribute
City
County
County Code
Datenumberpattern
Datestringpattern 1
Datestringpattern 2
Gender
Income Bracket
Multivaluedgroups
Product Category
Product Group

i. Select a source field for the filter. Use the search box at the top of the list to quickly locate a source field in the list. The Select Values dialog appears.



ii. Select an operator for the filter in the Operator drop-down menu.

iii. Optionally enter a custom value in the Customize field. The field name you supply is added and selected in the value list on the dialog.

iv. Select one or more values in the value list. Use the search box at the top of the list to quickly locate a value in the list.

12. Select **Save** to save the materialized view definition.

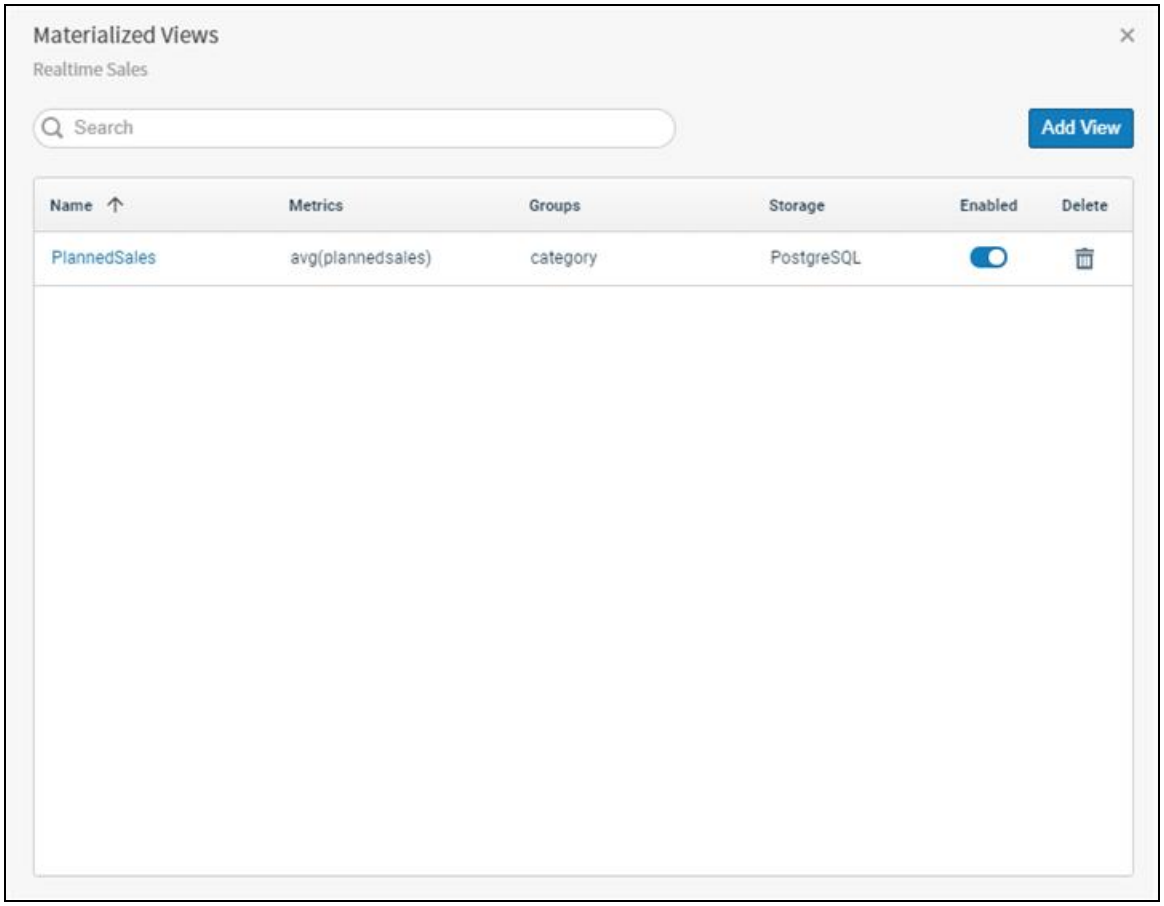
Edit a Materialized View Definition

Note: Materialized view functionality is disabled by default. To enable, contact technical support for assistance. The Materialized Views API is deprecated and will be removed in a future release.

Important: This is an experimental feature.

Edit a materialized view definition using the UI

1. Make sure you are logged in as a user with the **Administer Sources** privilege or that you have **read** and **write** permission for the data source.
2. List the materialized views for the data source. See List Materialized View Definitions.



3. Select the materialized view name in the list of materialized view definitions. The definition opens in a new dialog.

4. Modify any field in the definition. You can change the definition name, description and target settings. You can also add and remove metrics, groups, and filters from the definition. Remove metrics, group, and filter specifications by selecting ⊖ in the **Delete** column of the Metrics, Groups, and Filters tables.

   > **Note:** Materialized views only work if you specify the metrics and groups used by your visuals in the materialized view definition. For example, if you use Sales (SUM) as your metric and State as your group in a visual, be sure to add these metrics and groups to your materialized view definition. If they don't match, Composer will not use the materialized view to boost your visual rendering time.

   See Add A Materialized View Definition for explanations of each field in the definition.

5. Select **Save** to save the materialized view definition.

Delete a Materialized View Definition

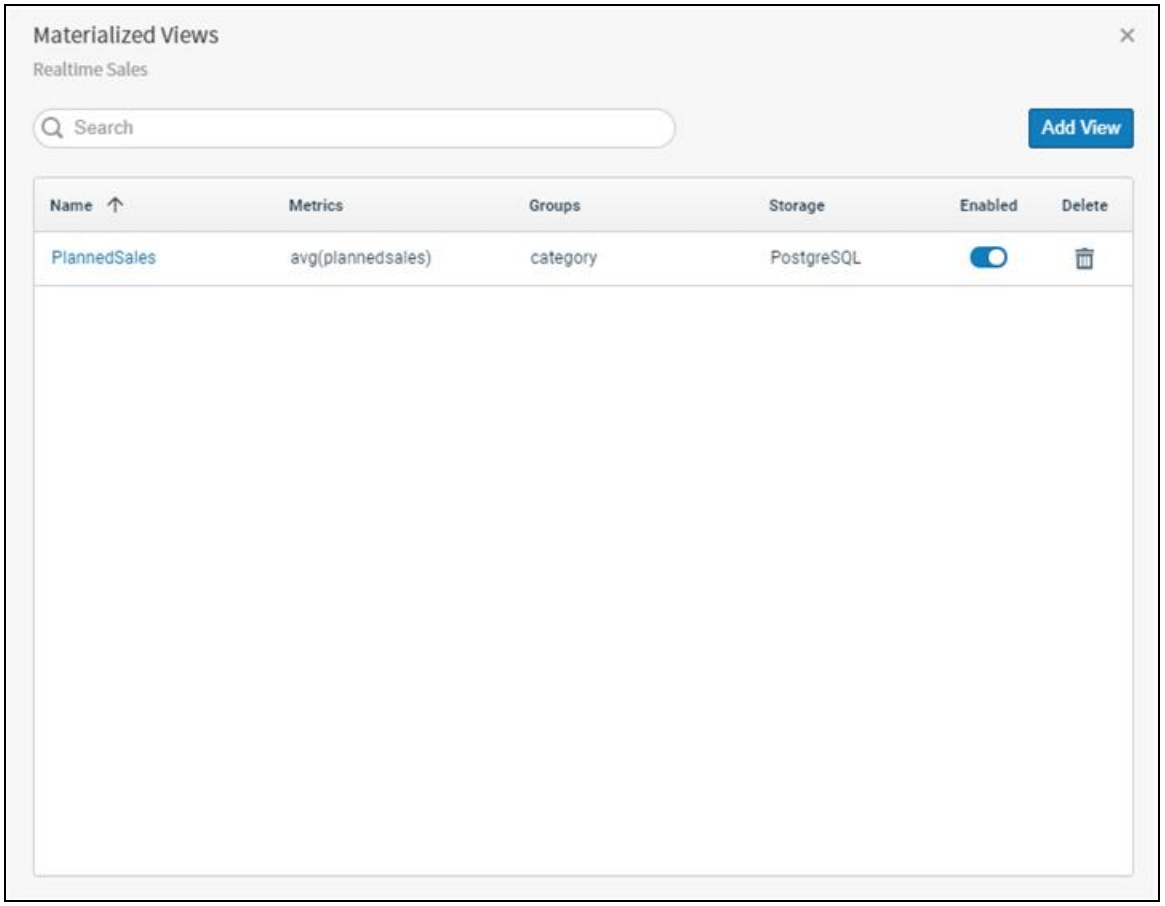
Note: Materialized view functionality is disabled by default. To enable, contact technical support for assistance. The Materialized Views API is deprecated and will be removed in a future release.

Important: This is an experimental feature.

Deleting a materialized view does not affect the underlying pre-aggregated data. When that data is not used any more, it should be deleted manually from you external storage by the data owner.

Delete a materialized view definition using the UI

1. Make sure you are logged in as a user with the **Administer Sources** privilege or that you have **read**, **write**, and permission for the data source.
2. List the materialized views for the data source. See List Materialized View Definitions.



3. Locate the materialized view definition you want to delete and select in the Delete column for the definition.

   A warning dialog appears, prompting you to confirm the deletion.

4. Select **Delete** on the warning dialog.

   The definition is deleted.

Enable and Disable Materialized View Definitions

Note: Materialized view functionality is disabled by default. To enable, contact technical support for assistance. The Materialized Views API is deprecated and will be removed in a future release.

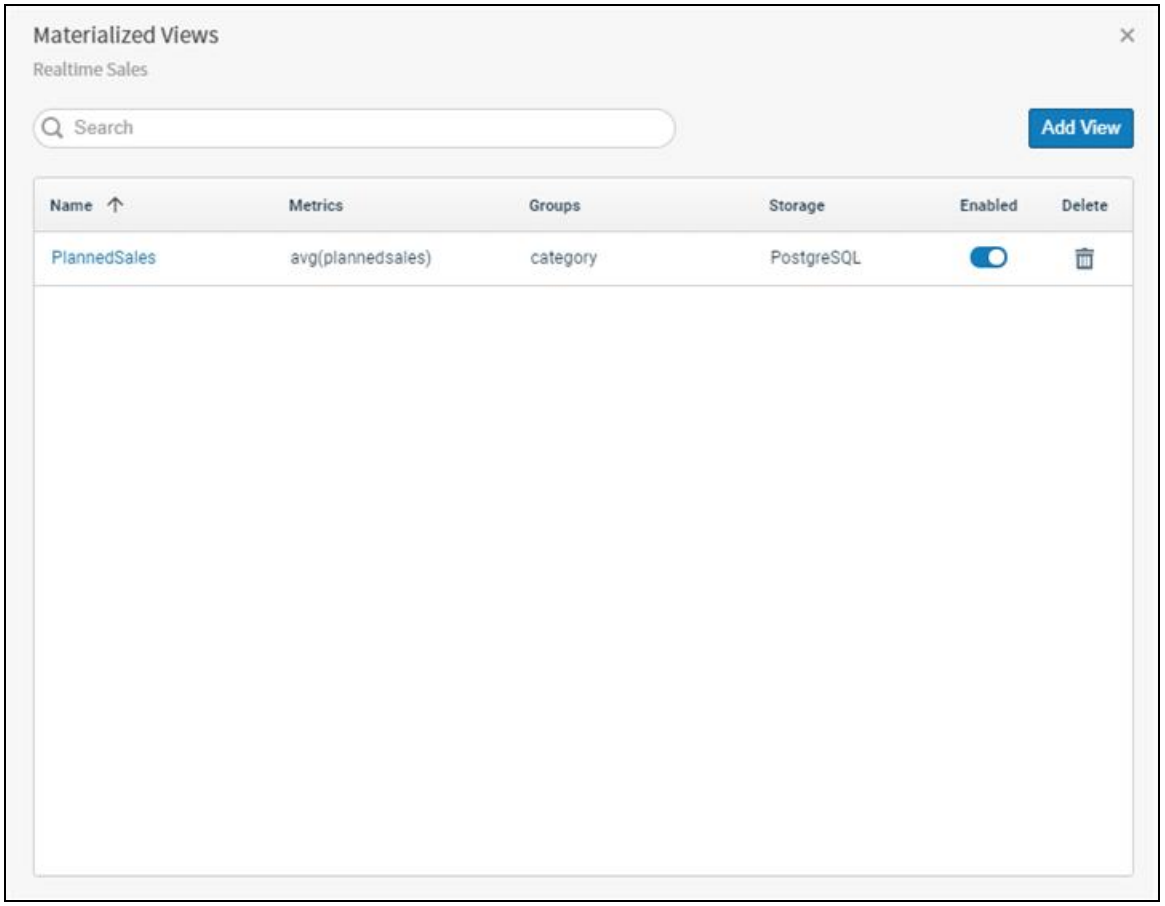
You can enable and disable individual materialized view definitions. When disabled, the stored query data for the materialized view is not used in visuals for the data source. Disabling a materialized view definition allows you to remove the effect of the definition without deleting it.

By default, individual materialized view definitions are enabled when they are defined.

Important: This is an experimental feature.

Disable a materialized view definition using the UI

1. Make sure you are logged in as a user with the **Administer Sources** or the **Create New Data Sources** privilege.
2. List the materialized views for the data source. See List Materialized View Definitions.

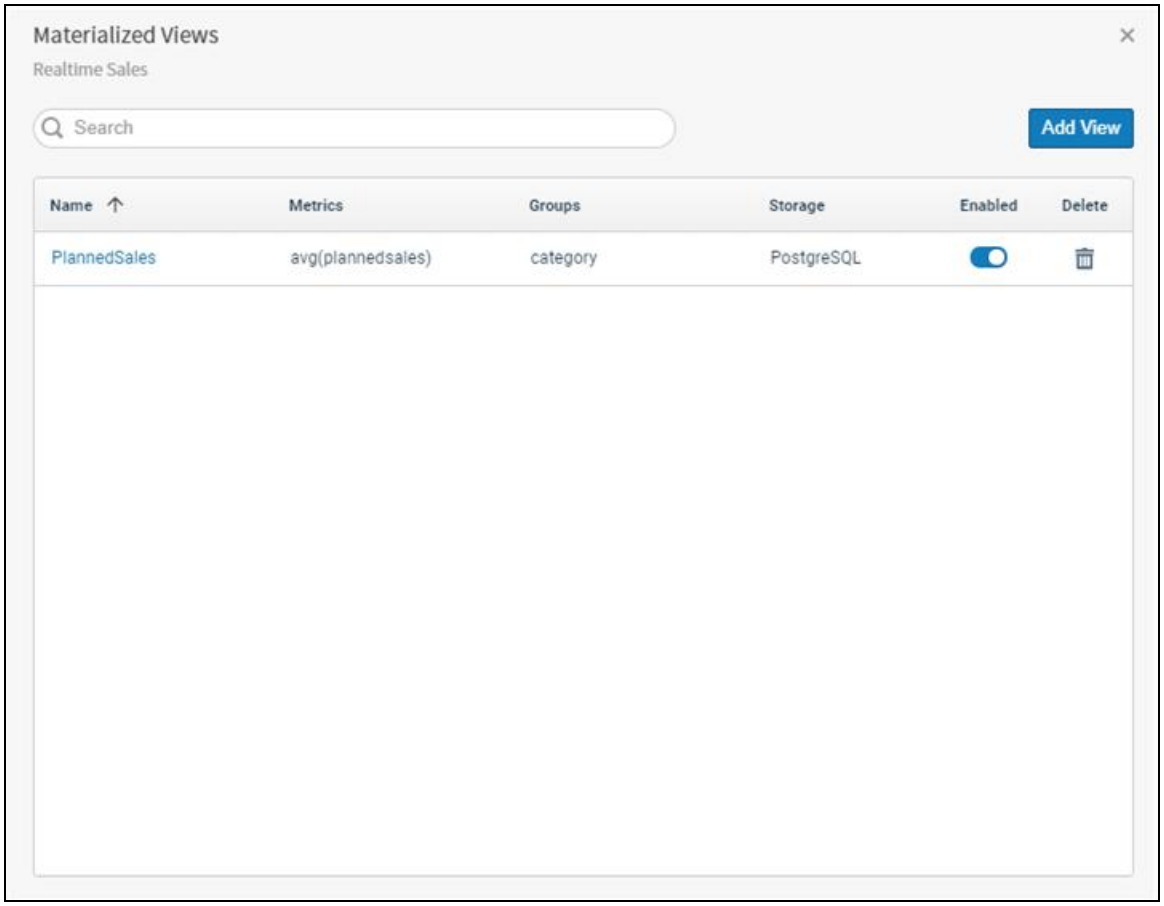


3. Locate the materialized view in the list of materialized view definitions and slide the switch in the **Enabled** column for the definition to the left (off).

   The definition is disabled.

Enable a materialized view definition using the UI

1. Make sure you are logged in as a user with the **Administer Sources** or the **Create New Data Sources** privilege.
2. List the materialized views for the data source. See List Materialized View Definitions.



3. Locate the materialized view in the list of materialized view definitions and slide the switch in the **Enabled** column for the definition to the right (on).

   The definition is enabled.

Delete a Data Source Configuration

You cannot delete a data source configuration if it has been used in any visuals on any dashboard. You must first remove it from the visuals. In addition, you can only delete a data source if you are logged in as a user with the **Administer Sources** privilege, or a user with **delete** permission for the data source.

Delete a data source configuration

1. Make sure you are logged in as a Composer a user with the **Administer Sources** privilege, or a user with **delete** permission for the data source.
2. Select **Sources** on the UI menu ( ) or the top-level navigation menu, or select the **Sources** box on the Home page. The Sources page appears.
3. In the table on the Sources page, locate the row displaying the data source configuration you want to delete.
4. Select in the **Actions** column for the row. A warning dialog requests confirmation of your delete request.
5. Select **Delete** on the warning dialog.

   The data source configuration is deleted.

Hide Fields

By default, all fields in a data source are visible and available for use in custom metrics, derived fields, , and in your dashboard visuals. However, there may be source fields, derived fields, or custom metrics in the data that you prefer to hide.

Hidden fields can be used in custom metrics, dashboard visuals, and derived fields.

When to Hide Fields

You can hide a field already used in a derived field, custom metric, or visual. The next time a user opens a visual that uses that hidden field, they are prompted to re-render the visual by selecting another available field.

How to Hide a Field

Hide source fields, derived fields, or custom metrics using the Fields tab of a data source configuration.

Hide a field

1. Make sure you are logged in as a user with the **Administer Sources** privilege, or a user with **read** and **write** permission for the data source.
2. Select **Sources** on the UI menu () or the top-level navigation menu, or select the **Sources** box on the Home page. The Sources page appears.
3. Select the **Fields** tab.
4. To hide a field or derived field, disable (toggle off) the corresponding **Visible** toggle in the Fields table. By default, all included fields for a data source are enabled and visible.
5. To hide a custom metric, disable (toggle off) the corresponding **Visible** toggle in the Custom Metrics table. By default, all custom metrics for a data source are enabled and visible.
6. Exit the source data configuration work area when you've finished defining field visibility.

Configure Time Bar Defaults

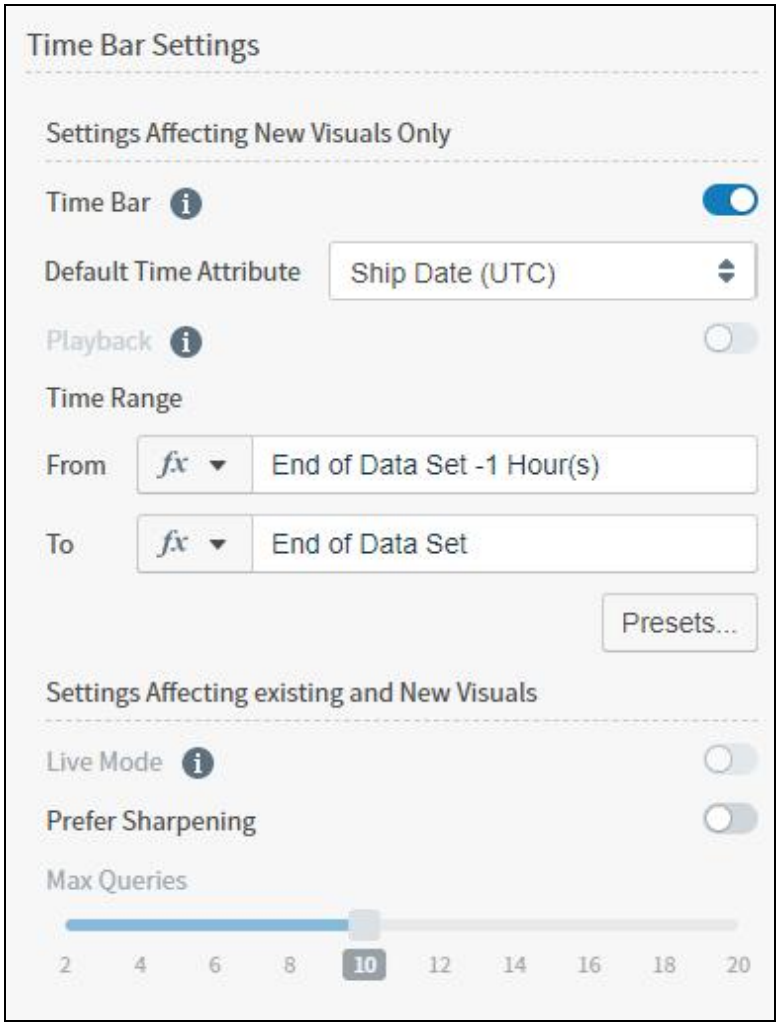
The time bar feature is available for all visual styles. When enabled, it appears at the bottom of a dashboard. You can use the time bar to filter the data in your visuals by a specified time attribute. If more than one data source is used for visuals on a dashboard, the time bar shown on the dashboard depends on the visual that is selected in the dashboard. For information on using the time bar on the dashboard, see Use the Time Bar.

You can set time bar defaults for data sources used in your Composer environment. These defaults are specified in data source configurations and are applied to visuals that are created using the data sources. Specifically, you can specify:

- The default time field used for the time bar.
- Whether playback and live mode should be available time bar features.
- The default time range used for the time bar.

Specify default time bar settings for a data source configuration

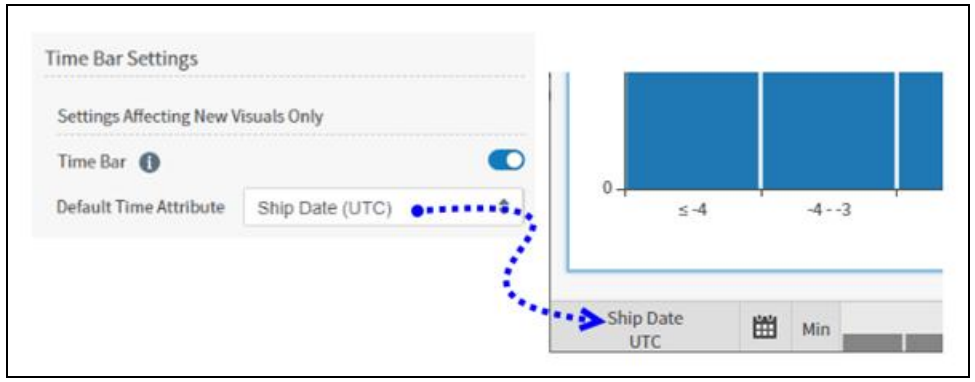
1. Log in as a user with the **Administer Sources** privilege, or write permission for this source.
2. Select **Sources** on the UI menu (). The Sources page appears.
3. Select the appropriate data source configuration to edit it, then access the Global Settings tab of the data source.



4. Enable Time Bar in Time Bar Settings. This allows you to edit all available time bar settings for new visuals. See Global Settings Tab

5. Select the default time field to use in the **Default Time Attribute** box.

   The attribute you select is used as default and is displayed on the time bar after you create and open a new visual:



If you select a **Default Time Attribute** that has the time zone information disabled, only the field name appears on the time bar and in this work area.

6. If the time attribute you select is *playable*, the **Enable Playback** and **Enable Live Mode** settings can be changed.

   Live mode and historical playback (also know as Data DVR) allow you to get the most from data sources using connectors that support live mode and playback (see Live Mode and Historical Playback). The only difference between live mode and historical playback is the time range that is selected:

i. Live mode refreshes field data on your visuals for fields that are indexed as playable. In live mode, your data plays forever without an end date.

ii. Historical playback (Data DVR) shows the historical record of field data for fields that are indexed as playable. Playback can show up to the last moment before the current period in your data.

   Live mode and historical playback require an index or partition field. The data store should be capable of receiving new or updated data, that is, data that is not static like flat files. If the data store does not support indexing or partitioning, then live mode and historical playback are not available. For most data stores, indexing is the default. For more information about the requirements for playback, see Live Mode and Historical Playback.

i. Enable the **Playback** toggle to show the Play button (Data DVR functionality) on the time bar.

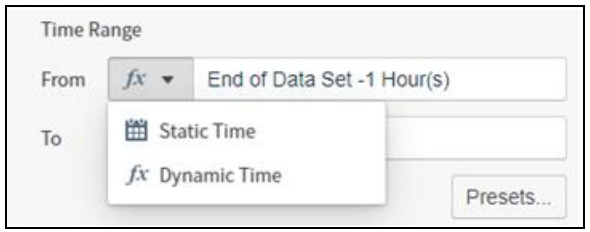
ii. Enable the **Live Mode** toggle to enable playing data in live mode.

   If you enable **Live Mode**, **Playback** is also enabled by default.

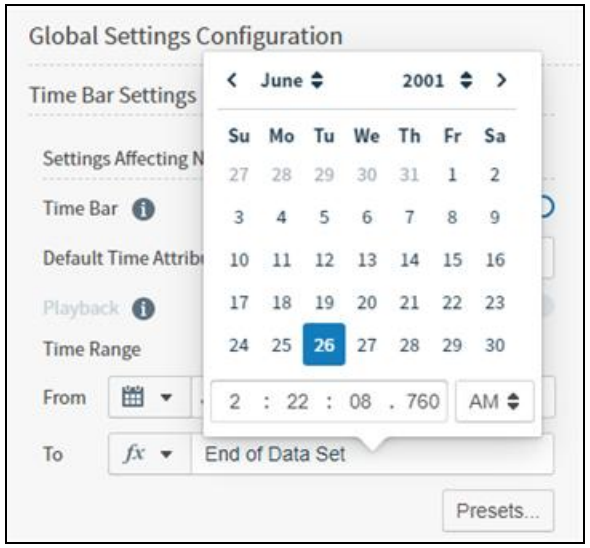
7. Use the **Time Range From** and **To** boxes to specify the default time bar range.

   You can set the range in static time or dynamic time, or use preset ranges provided with Composer.

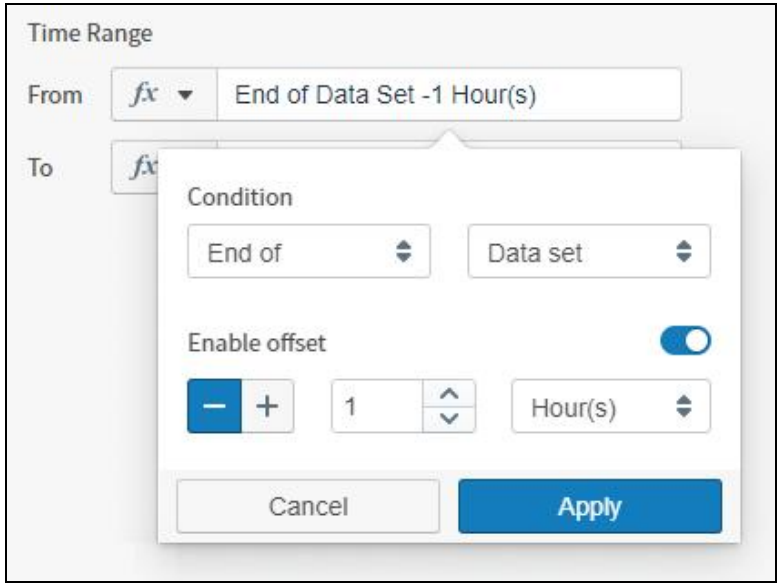
i. Select **Static Time** or **Dynamic Time** in the **fx** drop-down menu.



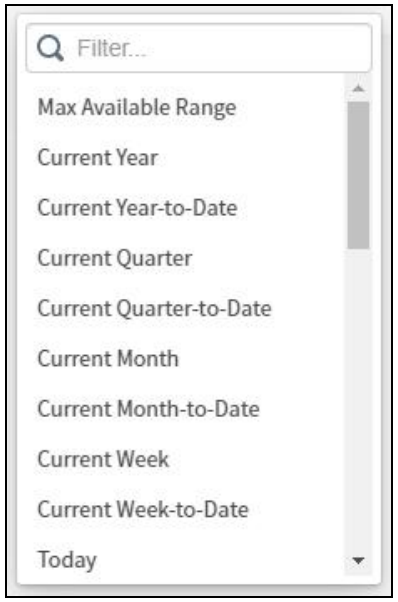
If you select **Static Time**, the **From** and **To** boxes are filled with default dates and times. Use the boxes to select specific from and to times:



If you select **Dynamic Time**, the **From** and **To** boxes are filled with **Start of data** and **End of data** automatically. Use the boxes to select different dynamic From and To times:

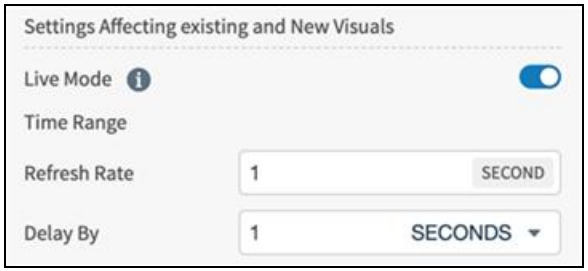


ii. Alternatively, select **Presets...** to fill the **From** and **To** boxes with predefined time ranges provided by Composer:



Use the filter box at the top of the presets list to locate the preset setting you want. Descriptions of each of the preset options are provided in Preset Time Ranges.

8. If you enable live mode for a data source (**Enable Live Mode** checkbox), you can set the refresh rate and delay time. The time bar defaults expand to show these settings.



i. Use the **Refresh Rate** box to specify the data refresh rate for the data source. The time granularity for the time field's refresh rate is defined in the Settings sidebar menu on the Fields tab of the data source.

   For information about using the REST API to identify and modify refresh rates, see Configure Data Source Refresh Rates Using the API.

ii. Use the **Delay By** box to specify the delay time when playing data in live mode.

9. When your changes are complete, select **Save Settings** to save your changes.

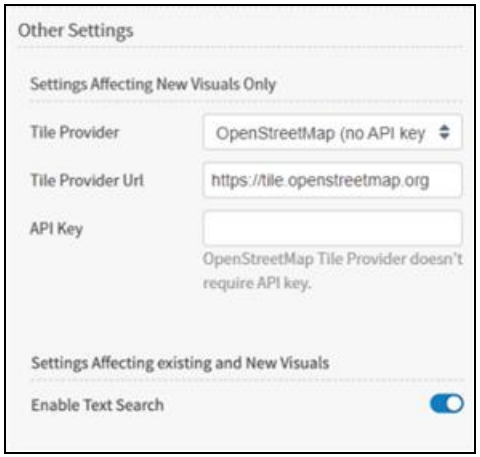
Configure Search Box Defaults

The search box is available for Apache Solr, Cloudera Search, and Elasticsearch data sources and is enabled by default for visuals created using these sources. It allows you to enter keywords to quickly filter the data. You can disable the search box, if needed, in the data source configuration.

Disable the Search Box

Disable the search box

1. Log in as a user with the **Administer Sources** privilege, or write permission for this source.
2. Select **Sources** on the UI menu ( ). The Sources page appears.
3. Select the appropriate data source configuration to edit it, then access the Global Settings tab of the data source.
4. Toggle off **Enable Text Search** in the **Other Settings** work area if you want to disable the text search option for new and existing visuals.

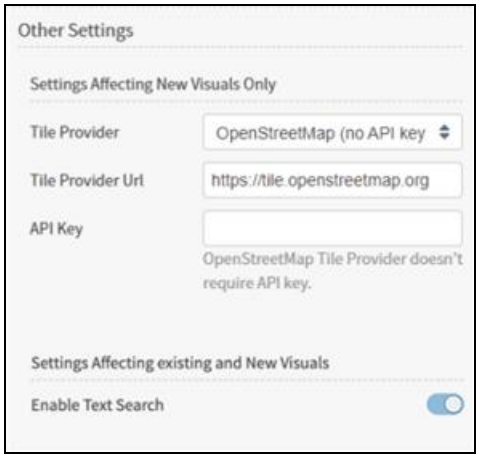


Note: The availability of this setting depends on the data source you have selected. The Search feature is available for Apache Solr, Cloudera Search, and Elasticsearch data sources.

Enable the Search Box

Enable the search box

1. Log in as a user with the **Administer Sources** privilege, or write permission for this source.
2. Select **Sources** on the UI menu ( ). The Sources page appears.
3. Select the appropriate data source configuration to edit it, then access the Global Settings tab of the data source.
4. Toggle on **Enable Text Search** in the **Other Settings** work area if you want to enable the text search option for new and existing visuals.



Note: The availability of this setting depends on the data source you have selected. The Search feature is available for Apache Solr, Cloudera Search, and Elasticsearch data sources.

Trigger Refresh Jobs

Composer maintains data source metadata and, optionally, a cached result set of the data from the data store for each data source configuration you define. When you initially connect to your data store using a data source configuration, a sampling of the data is collected to determine:

- Distinct values for all fields with ATTRIBUTE and NUMBER field types.
- The minimum and maximum values for all fields with number or time data types for which Custom Ranges have not been defined.

Composer administrators can define refresh jobs for this cached data. The following table identifies the types of refresh jobs that are currently supported, the triggers for the jobs, and the activities that occur when the job is run.

Refresh Type	Trigger	Activities
Source Refresh	An initial connection to a data source is saved. See Define A Source. Changes to the Source Creation Tab or Fields tab in the data source configuration are saved. See Edit A Data Source. The scheduled refresh time (set on the Cache tab in the data source configuration) occurs. See Set Up A Data Source Refresh Job.	The Composer cache is cleared and the minimum and maximum values for non-attribute fields and distinct values for attribute and number fields are refreshed.
Field Refresh	▪ To refresh the entire list of fields in a data source configuration, access the Cache tab and select the refresh (⟳) button for Manual Refresh in the table heading to trigger a manual refresh for all fields. This triggers a manual refresh of all the field data. ▪ To refresh a specific field in a data source, access the Cache tab and select the Manual Refresh button (⟳) for the field you want to refresh. This triggers an immediate manual refresh for the selected field.	The minimum and maximum for non-attribute fields and distinct values for attribute and number fields are refreshed.
After Cache Cleanup	When a user selects an available Cache Cleanup option from the Actions column for a data source on the Sources page. See Clear The Cache For A Data Source Configuration.	The data is freshly loaded into the Composer cache the next time it is requested from the data source.

Note: If a Custom Range has been defined for a field, the minimum and maximum fields used in filters remain unchanged when you refresh source data. These fields are shown with cache actions disabled on the Cache tab.

Note: When you add a new field to a data source, the scheduled refresh is not enabled for the new fields by default. Quickly enable scheduled refresh for all fields using the bulk update option in the Schedule Refresh menu on the Cache tab, or enable each field for scheduled refresh manually on the Cache tab.

The status of refresh jobs can be reviewed on the Console of Refreshing Jobs. See Review Refresh Jobs.

Set Up a Data Source Refresh Job

Data source refresh jobs are defined on the Cache tab of a data source configuration. Only one refresh job can be defined for a data source, however you can manually trigger refresh jobs on the Cache tab of the data source configuration. See Trigger Refresh Jobs.

A refresh job requires you to specify a schedule for the job and, optionally, identify fields for the refresh.

By default, no data refresh jobs are scheduled for a data source. Select **Schedule Refresh Settings** on the Cache tab to set up Periodic or Advanced refresh jobs. An initial data source refresh job is run after the data source has been successfully created and saved. After that, no additional refreshes of the data occur, although you can manually trigger a refresh of the field data. See Trigger Refresh Jobs.

- **Periodic**: Select this option to define a refresh job that runs at standard periods.
- **Advanced**: Select this option to define a refresh job using cron expressions. Use this option to define refresh jobs that run on a more complicated schedule.

In addition to the refresh job schedule options, you can select specific fields for the refresh job.

For specific instructions, see one of the following topics:

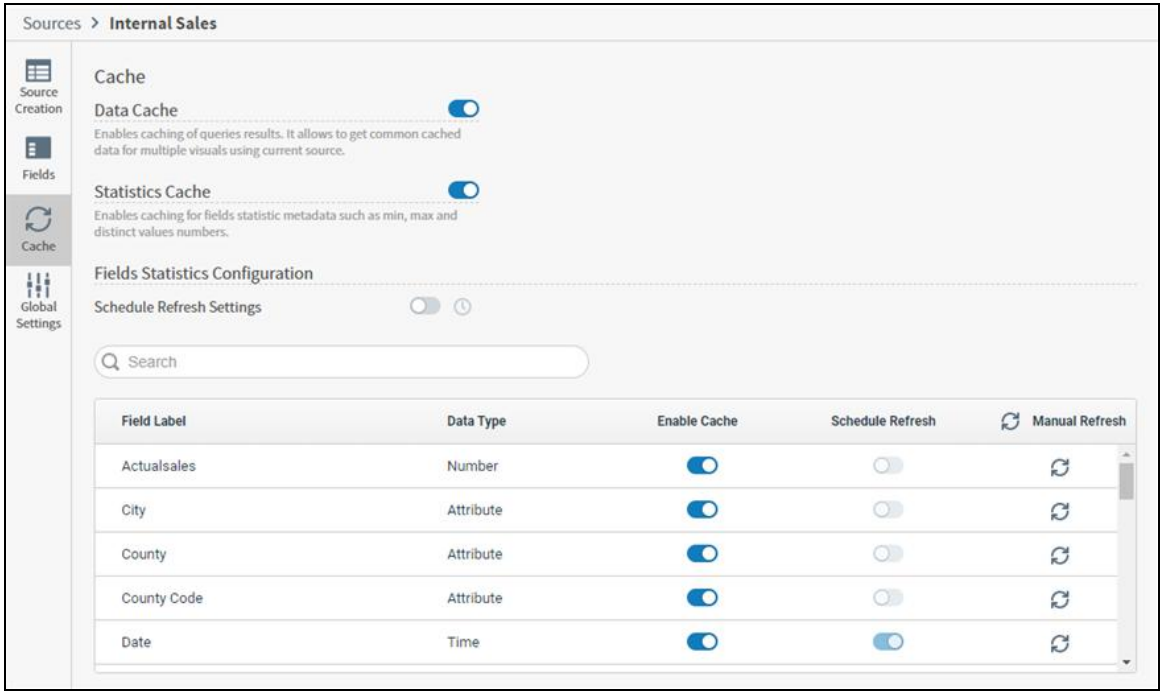
- Configure a Periodic Refresh Job
- Configure an Advanced Refresh Job
- Identify Specific Fields for a Refresh Job

Configure a Periodic Refresh Job

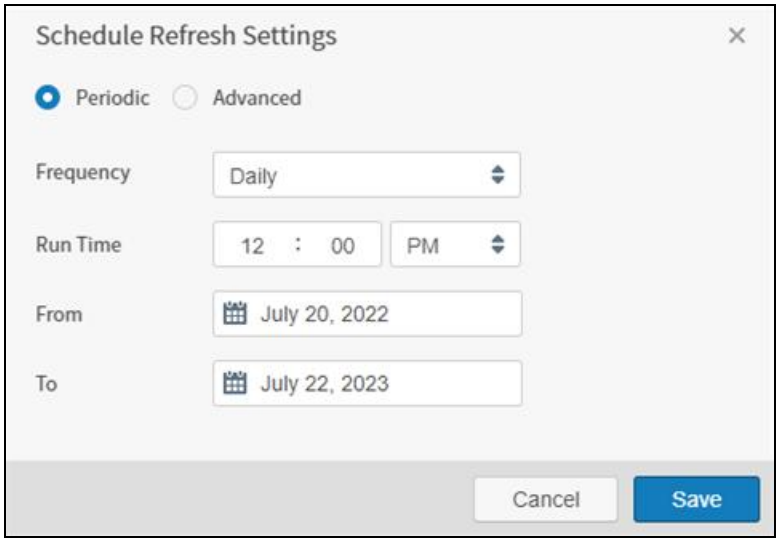
You can configure a periodic refresh job for the data cached for a data source configuration.

Configure a periodic refresh job for a data source configuration

1. Log in as a user with the **Administer Sources** privilege, or a user with **read** and **write** permission for the data source.
2. Select **Sources** on the UI menu ( ). The Sources page appears.
3. On the Sources page, locate and select the data source configuration you want to edit. The Source Creation work area opens.
4. Select the **Cache** tab. All the fields from your data source are listed.



5. Select to enable **Schedule Refresh Settings** and select **Periodic** in the pop up. The settings work area for a periodic refresh job appear.



6. Select the time interval for the job to be run using the **Frequency** list. Options **Daily**, **Weekly**, and **Monthly**. Depending on the **Runs** option you select, corresponding options become available in the **Run every** section:

Runs Selection	Run every Specification
Daily	Specify the time of day in **Run Time**, and a start and end date for daily runs at that time using the calendar options in **From** and **To**. The first job runs on the date and time at which you save the data source. Subsequent jobs continue daily until the last date and time occur.
Weekly	Select the day of the week for the job to be run in **Run on**, time of day in **Run Time**, and a start and end date for weekly runs at the time using the calendar options in **From** and **To**. The first job runs on the date and time at which you save the data source. Subsequent jobs continue weekly until the last date and time occur.
Monthly	Specify the number of months between job runs in **Run on**, time of day in **Run Time**, and a start and end date for monthly interval runs at the time using the calendar options in **From** and **To**. The first job runs on the date and time at which you save the data source. Subsequent jobs run at the **Run Time** time after the number of months you specify have passed.

Runs Selection	Run every Specification
	For example, if you set your job to run every three months and the **Start on** time is April 4, 2024 at 5:00 a.m., the subsequent job runs on July 4, 2024 at 5:00 a.m.

7. Select **Save** to save your changes, then exit the source data configuration when you've finished defining your periodic refresh jobs. A summary of the configuration appears below **Schedule Refresh Settings.**

Note: If you disable **Schedule Refresh Settings**, any schedule you had set up previously is deleted.

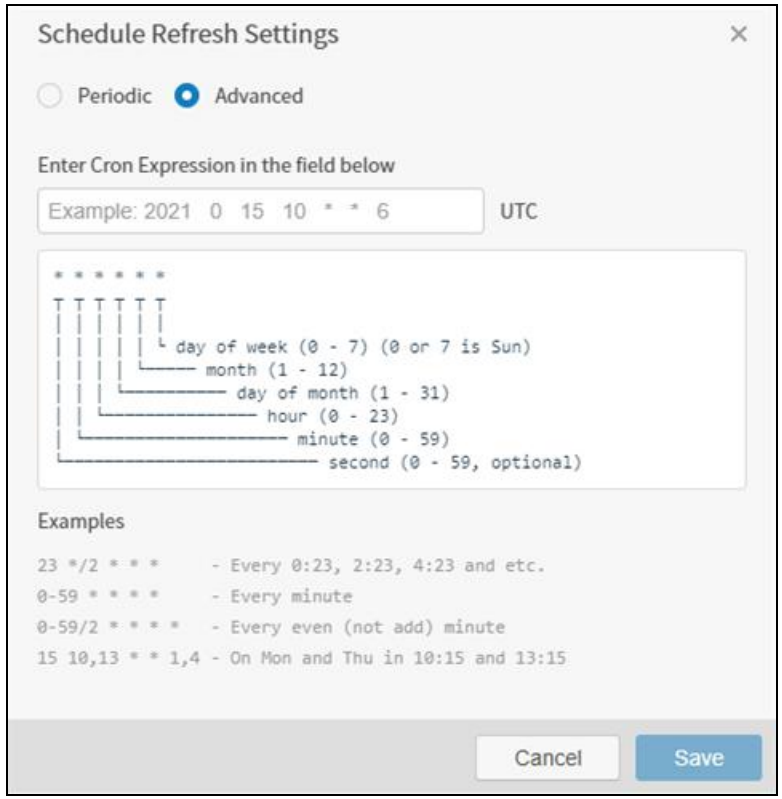
Select the Schedule Refresh menu (☰) button to quickly enable or disable scheduled refreshing for all fields. This is available only if you have enabled **Schedule Refresh Settings** and defined a frequency.

Configure an Advanced Refresh Job

Using cron expressions, you can specify a more complicated schedule for refresh jobs for the data cached for a data source configuration.

Configure an advanced refresh job for a data source configuration using a cron expression

1. Log in as a user with the **Administer Sources** privilege, or a user with **read** and **write** permission for the data source.
2. Select **Sources** on the UI menu ( ). The Sources page appears.
3. On the Sources page, locate and select the data source configuration you want to edit. The Source Creation work area opens.
4. Select the **Cache** tab, then enable enable **Schedule Refresh Settings** and select **Advanced** in the pop up. The settings work area for an Advanced refresh job appear.



5. In the box provided, enter a cron expression that sets the schedule using a string of six fields, each separated by a blank space. The format for the cron expression is:

```
<seconds> <minutes> <hours> <days of the month> <months> <days of the week>
```

The standard values supported by each field include:

Field	Allowed Values	Additional Characters
<seconds>	0-59	, - * /
<minutes>	0-59	, - * /
<hours>	0-23	, - * /
<days of the month>	1-31	, - * / ? L W
<months>	1-12 or Jan-Dec	, - * /
<days of the week>	1-7 or Sun-Sat	, - * / ? L W #

When creating the cron expression, keep the following requirements in mind:

i. Specifying `<seconds>` is optional, but defaults to zero seconds, which ensures that only one refresh job is run. If you specify seconds, the refresh job is rerun after that number of seconds has elapsed.

ii. Either the `<days of the month>` or `<days of the week>` field is needed, but both should not be specified. If one of these fields is marked with an asterisk (*), the other is automatically set to a question mark in the cron expression. In addition, an error is returned if you try to specify both of these fields.

> **Note:** If you save an existing data source configuration that has both fields specified, no error is returned, but warnings are written to the log when Composer starts.

iii. Names for the `<month>` and `<days of the week>` fields are not case sensitive. For example, `FRI` and `fri` are both acceptable formats.

Special characters that can be specified are described in the table below.

Special Character	What It Means
*	All values. Represents all the values within the specified field. In the following example, an asterisk is used in the <minutes> field, indicating that the job will run every minute:

Special Character	What It Means
	`0 * 0 0 0 0`
?	No specific value. Used as a placeholder when no value is needed in the field. In the following example, the <months> field value is 6 (June) and the <days of the week> field value is a question mark, indicating that job runs in June, regardless of the day of the week: `0 0 0 0 6 ?`
-	Used to specify a range of values. For example, specifying `3-6` in the <hours> field (as shown in the following example) means the job will run at 3:00, 4:00, 5:00 and 6:00 am: `0 0 3-6 0 0 0`
,	Used to specify a series of values. Use the comma to identify all the values for the field. In the following example, a comma is used in the <days of the week> field to run the job on Wednesdays, Thursdays, and Fridays: `0 0 0 0 0 Wed,Thur,Fri`
/	Used to specify the starting time value and the incremental increase of time. In the following example, specifying 0/5 in the <minutes> field means that the job runs immediately and then every 5 minutes. `0 0/5 0 0 0 0`
L	Last. Used only in the <days of the month> and <days of the week> fields. When this character is used standalone, it indicates the last day of the month or the last day of the week (Saturday). However, when it is used with a value (such as 5L) in the <days of the month> field, it means the last Friday of the month (5 is Friday, L indicates the last Friday). `0 0 0 5L 0 0`
W	Weekday. Used only in the <days of the month> and <days of the week> fields.

Special Character	What It Means
	Identifies the weekday closest to the given day. For example, 15W means the closest weekday to the 15th of the month. The following results are possible: ■ If the 15th falls on a Saturday, the result returned would be Friday the 14th ■ If the 15th falls on a Sunday, the result is Monday the 16th ■ If the 15th falls on a weekday, that specific day is returned.
#	Number sign. Used only with the <days of the week> field. Identifies the specific day of the month. For example, both `Wed#2` and `3#2` identify the second Wednesday of the month. `0 0 0 0 0 Wed#2`

The following table provides some cron expression examples:

cron Expression	Meaning
`0 0 12 * * ?`	Noon every day
`0 30 20 ? * *`	8:30 p.m. every night
`0 0/10 17 * * ?`	Every 10 minutes starting at 5 p.m. and ending at 5:50 p.m., every day
`0 15-30 20 * * ?`	Every minute starting at 8:15 p.m. and ending at 8:30 p.m., every day
`0 45 20 ? * Mon,Wed,Fri`	8:45 p.m. every Monday, Wednesday and Friday
`0 0 20 3/3 * ?`	8 p.m. every 3 days in every month, starting on the third day of the month

6. Select **Save** to save your changes, then exit the source data configuration when you've finished defining your periodic refresh jobs. A summary of the configuration appears below **Schedule Refresh Settings.**

Note: If you disable **Schedule Refresh Settings**, any schedule you had set up previously is deleted.

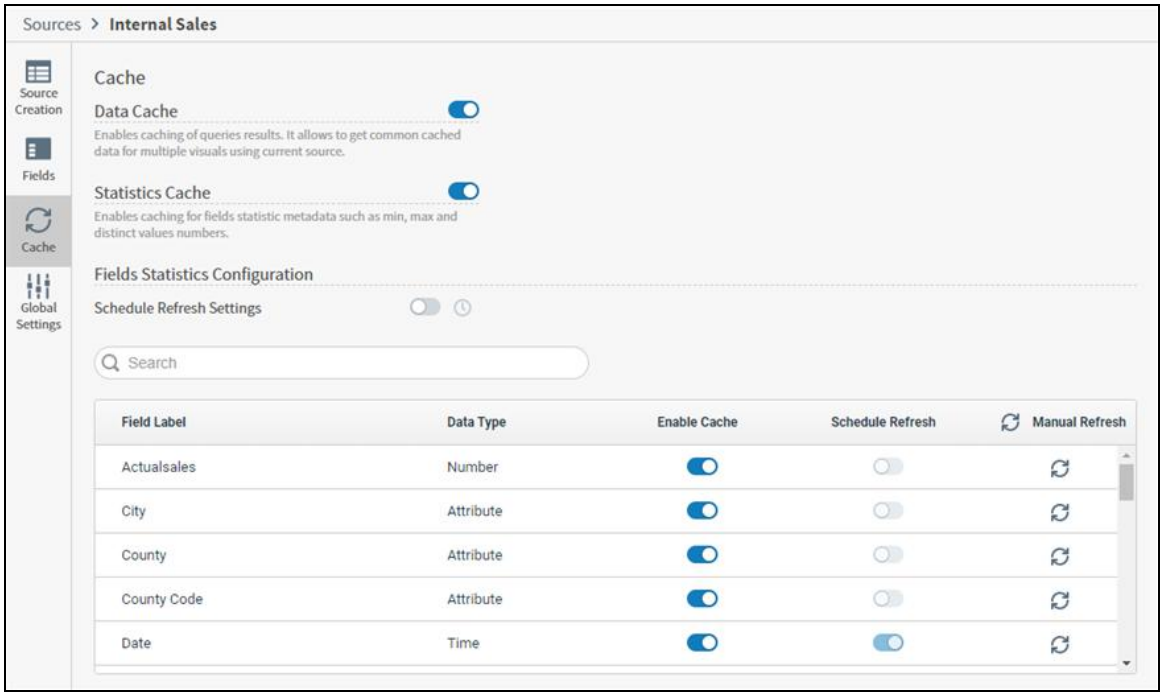
Select the Schedule Refresh menu (☰) button to quickly enable or disable scheduled refreshing for all fields. This is available only if you have enabled **Schedule Refresh Settings** and defined a frequency.

Identify Specific Fields for a Refresh Job

You can select specific fields to be refreshed by a data source refresh job using the Configuration section of the Refresh tab.

Select specific fields for a refresh job for a data source configuration

1. Log in as a user with the **Administer Sources** privilege, or a user with **read** and **write** permission for the data source.
2. Select **Sources** on the UI menu ( ). The Sources page appears.
3. On the Sources page, locate and select the data source configuration you want to edit. The Source Creation work area opens.
4. Select the **Cache** tab. All the fields from your data source are listed.



5. To refresh all the fields in your data source, select the refresh ( ) button for Manual Refresh to trigger a manual refresh for all fields.
6. Otherwise, select the refresh ( ) button for specific fields. Exit the source data configuration when you've finished triggering all of your manual refreshes.

Review Refresh Jobs

Administrators can monitor data source refresh jobs using the Console of Refreshing Jobs. Data on this page automatically refreshes every 15 seconds.

Access the Console of Refreshing Jobs

1. Log in as a Composer administrator.
2. Select **Console** on the UI menu (). The Console of Refreshing Jobs page appears. It shows a table of refresh jobs that are in progress or that have completed.

Refresh jobs are identified in the table by the data source configuration name. The following information is provided:s

Column	Description
Data Source	The name of the data source configuration to which the job applies.
Job Type	The type of refresh job that was run. Possible types are **Source Refresh** (the data source metadata and all fields were refreshed) and **Field Refresh** (an individual field was refreshed). Field refresh jobs also identify the field that was refreshed in the job type (for example, **Field Refresh [condition_code]** indicates that the `condition_code` field was refreshed).
Status	The status of the job. Possible statuses include: ▪ **Complete**: The job completed successfully. ▪ **In Progress**: The job is running or is scheduled to be run. ▪ **Incomplete**: The job only partially completed. For example, the minimum and maximum values were successfully refreshed, but the distinct values were not refreshed. ▪ **Failed**: The job could not run or could not be completed due to some error in the system For example, there may be connection problems with the data store. Select the arrow to view details on the issues that occurred while running the job.
Last Finished	The date and time when the most recent job completed.
Next Run	The date and time when the next job is scheduled.
Job History	Select  in this column to list all the completed refresh jobs of the selected refresh job type that have been run for the selected data source. The list appears on a pop-up dialog. The dialog shows the start time, finish time and status of the refresh jobs.

By default, the table is sorted in order by the Last Finished date and time. You can sort the table by selecting any of the following column headings: **Data Source**, **Job Type**, **Status**, or **Last Finished**. If you select a heading once, the table data sorts in reverse lexicographical order by the selected column heading. Select the heading a second time to sort the data in lexicographical order. The table sort order is automatically reset to the default order every time the table is refreshed.

You can filter the list using any of the column headings. Select the filter icon  in the column heading to select for specify an appropriate filter on a pop-up dialog. For example, the pop-up dialog for the Data Source column allows you to select data sources you want to see in the list.

How Composer Caches Data

Composer uses a visual cache to enhance performance in scenarios where large numbers of users are concurrently viewing the same shared visuals. A metadata cache is also used to store field statistics. This cached data is shared between users only if they have the same data access permissions and security context.

Caching is enabled by default for all data sources. Composer does not requery the data source for data unless you manually clear the cache, or define a refresh schedule. See Cache Tab and Trigger Refresh Jobs.

Available cache settings and options include:

- **Data Cache**: Composer caches query results and common cached data for multiple visuals that use this source. When you create a visual that uses this source, the first data request is sent to the Composer cache. Composer returns the cached data, if available.

  If the data is not available in the cache, Composer next queries the data source. The results are returned and stored in the cache, and the visual displays the returned data.

- **Statistics Cache**: When enabled, field metadata, such as minimum, maximum, and distinct values numbers are cached. This toggle also controls the availability of the Field Statistics Configuration option and scheduled refresh settings.

- **Field Statistics Configuration**: When enabled, you can enable or disable caching and scheduling for individual fields, or manually refresh the cached data for individual fields. Fields that include a statistics override that prevents refreshing are indicated by an exclamation point in a triangle.

- **Schedule Refresh Settings**: Enable and define Periodic or Advanced refreshing of fields with **Schedule Refresh** enabled. If you disable Schedule Refresh Settings, any schedule you had set up previously is deleted.

Note: If a Custom Range has been defined for a field, the minimum and maximum fields used in filters remain unchanged when you refresh source data. These fields are shown with cache actions disabled on the Cache tab.

You can refresh the entire data source, all the fields in a data source, or select fields in a data source. For more information, see:

- Clear The Cache For A Data Source Configuration
- Disable Data Caching For A Data Source
- Trigger Refresh Jobs

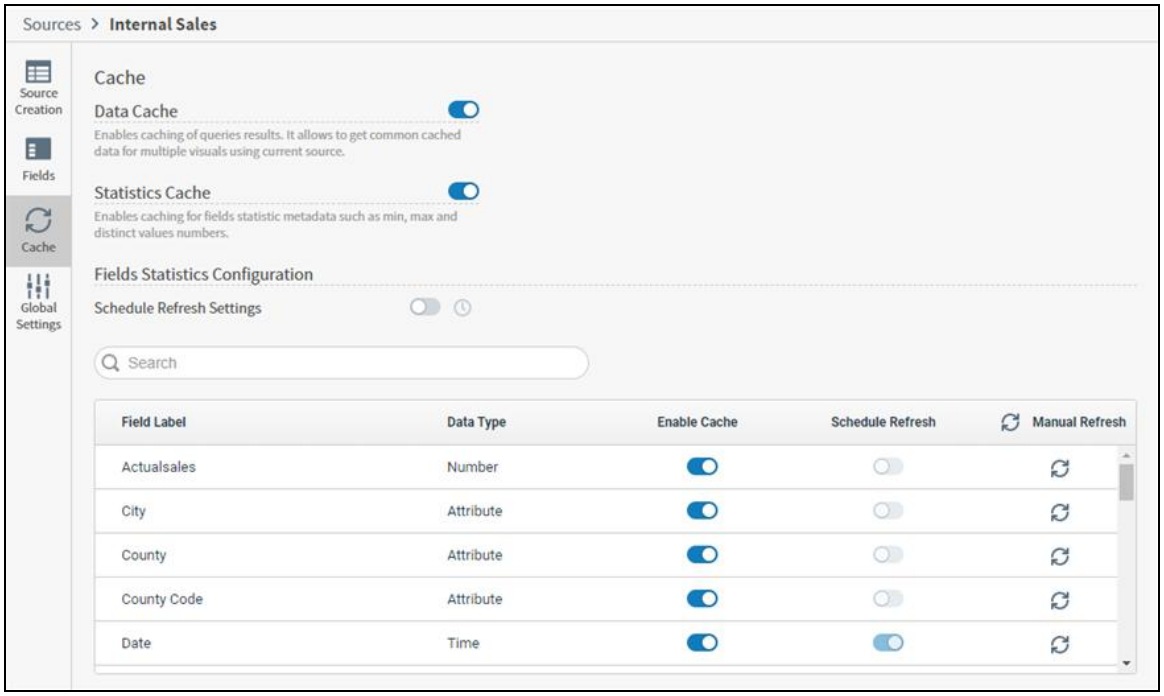
Note: You can force Composer to bypass the cache and to query the underlying data source by selecting **Refresh All** from a Composer dashboard menu.

Disable Data Caching for a Data Source

The Composer data cache is a temporary storage area containing the aggregated data from your data sources. Two caches are supported: the visual (visual data) cache and the metadata (field statistics) cache. By default, caching is enabled for all data sources using both caches. However, you can disable all caching if your data source is constantly being updated, or you do not want to allocate the required RAM or if the performance of your data source is so high you do not need to store the aggregated queries.

Disable data caching

1. Make sure you are logged in as a user with the **Administer Sources** privilege, or a user with **read** and **write** permission for the data source.
2. Select **Sources** on the UI menu ( ). The Data Sources page appears.
3. On the Sources page, locate and select the data source configuration you want to edit. The Source Creation work area opens.
4. Select the **Cache** tab. All the fields from your data source are listed.



5. Disable (toggle off) **Data Cache**. Data caching is disabled for the data source configuration. Data will be freshly loaded into the Composer cache the next time it is requested from the data source.

6. Optionally, Disable (toggle off) **Statistics Cache**. Field metadata caching is disabled for the data source configuration. Data will be freshly loaded into the Composer cache the next time it is requested from the data source. Data will be freshly loaded into the Composer cache the next time it is requested from the data source.

   If you have set up refresh jobs for this data source, a warning appears that the jobs are disabled and the schedule deleted for this data source. Select **Ok** to continue disabling Statistics Cache and delete the schedule.

7. Exit the source data configuration when you've finished making your changes.

Fuse Data Sources

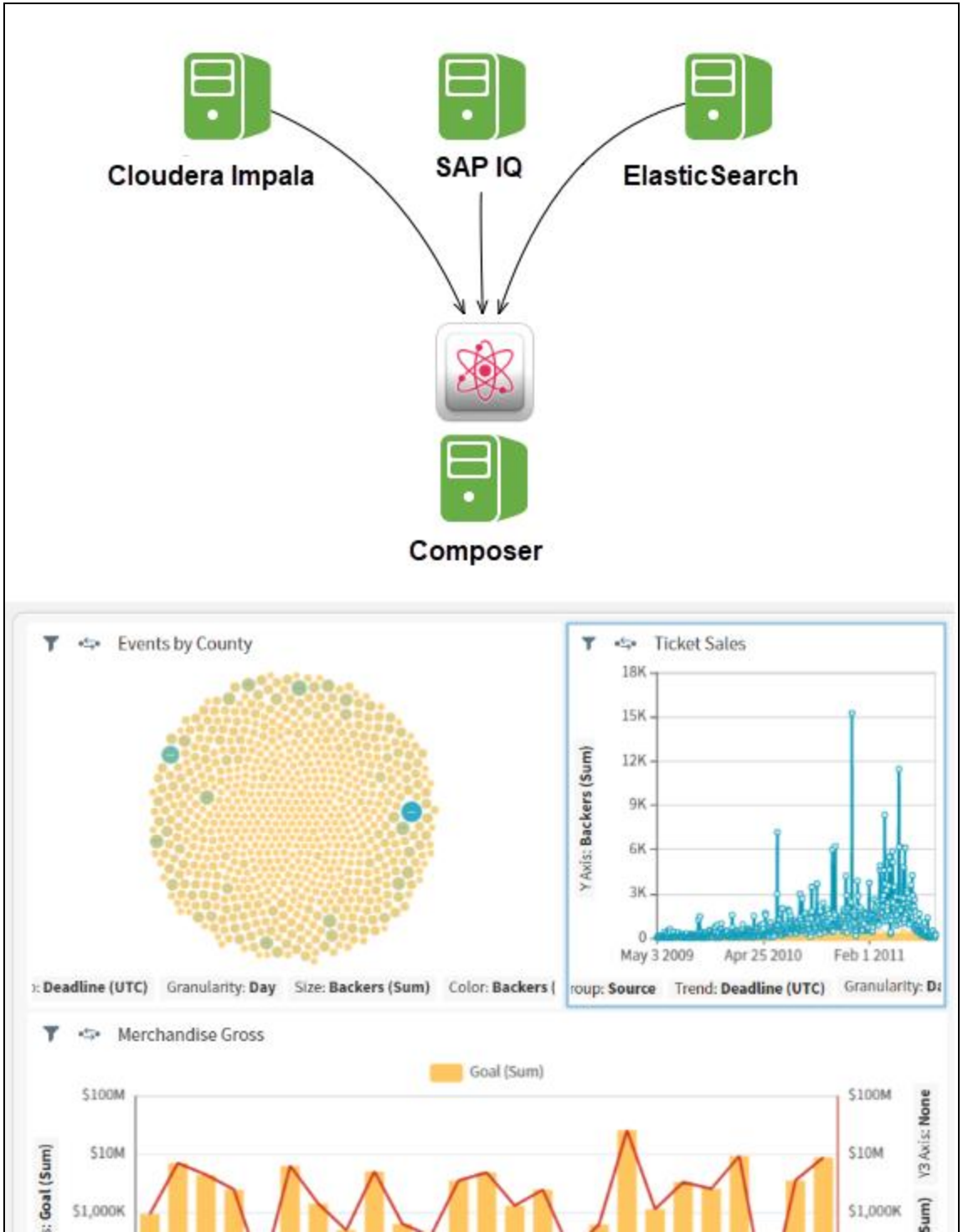
Data fusion is the concept of tying together data from two or more connections or flat files into a single source for exploration and analysis. After Composer joins the data, you can visualize the combined results in visuals and dashboards. Composer supports data fusion between all data connections it supports.

Fused data can generate more meaningful insights than what might be available in the data from a single data store. For example, suppose a data warehouse stores ticketing sales and events data in the following different data stores:

- Information about buyers and sellers stored in SAP IQ
- Events data stored in Elasticsearch
- Ticket sales stored in Cloudera Impala

You can create a Fusion source to fuse the data in these data stores together. After you create a connection to each data store, create data entities for each during source creation, you can join the data to use the combined data for more in-depth and complete analysis and exploration.

The following diagram depicts the basic concept of Composer data fusion.


Cloudera Impala
SAP IQ
ElasticSearch
Composer
Events by County
Deadline (UTC)
Granularity: Day
Size: Backers (Sum)
Color: Backers (
Ticket Sales
18K
15K
12K
9K
6K
3K
0
Y Axis: Backers ($um)
May 3 2009
Apr 25 2010
Feb 1 2011
roup: Source
Trend: Deadline (UTC)
Granularity: Da
Merchandise Gross
Goal (Sum)
$100M
$10M
$1,000K
Y3 Axis: None

Data fusion is available through Composer's familiar and intuitive user interface. Step-by-step instructions are provided in Create A Fusion Source.

Fused data sets are stored as a Fusion source (). Access Fusion data sources in the same way as other Composer data sources. Visualize the fused data in standard or custom charts.

For more information, see:

- Data Fusion Limitations
- Data Fusion Processing
- Data Fusion Join Rules
- Filter Fused Data
- Data Fusion Table Structures
- Data Fusion Use Cases
- Create A Fusion Source
- Optimize Joins

Data Fusion Limitations

Fusion data sources have the following limitations:

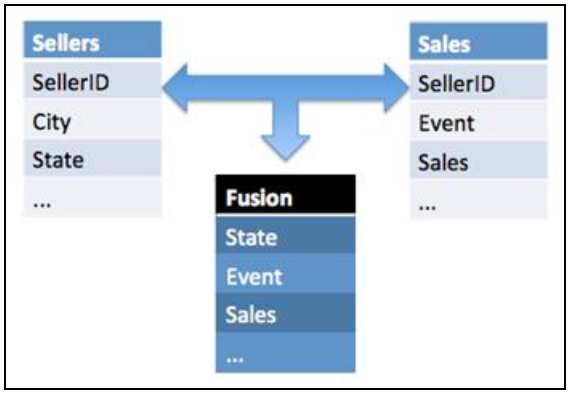
- Fused data can be used in tables (raw data). Composer recommends that you initially use a subset of fields from Fusion data sources on tables to limit the load on Composer's query engine and improve its performance. You can change the subset in data source configurations and add additional fields, later, as needed while working on a dashboard.
- Text Search and Facet filtering are not supported for fused data, even if one or all the data sources defining the fused source support it.
- The COUNT, COUNTD, TableCOUNT, TableCOUNTD, WindowCOUNT, and WindowCOUNTD aggregate functions are supported for Fusion data sources. However, they normally ignore null values for the specified field. Consequently, the result of these aggregate functions may not be the same as the actual number of records in the data. Use the wildcard character (*) for `<field>` to include null values for the field in the count.
- Live mode and historical playback are now supported for fused data.
- Data Sharpening is not supported for Fusion data sources.
- Two levels of top-of-the-top multi-group fusion are supported. If you have custom charts that require more than two levels, you cannot use fused data sources.

  Note: Multi-group fusion provides multiple levels of grouping under a single header. A multi-group variable is a stringified array of grouped queries. Each of the listed groups is described in the same manner as a group in the Query Configuration Object.

Data Fusion Processing

With data fusion, Composer can perform Group By operations using fields that are available across tables. A variety of table structures residing in data repositories can be fused, including lookup tables, fact tables, star and snowflake schema structures. See Data Fusion Table Structures.

For example, if a table in one data repository contains the IDs and address information of sellers and another table in another data store contains IDs, events, and sales information for sellers, these disparate fields can be fused into one sellers table with the three fields joined and accessible (as shown below).



Using data fusion, you can create data entities to join disparate data repositories that are connected to Composer. Multiple data entities (three or more) can be fused into a single Fusion data source.

To fuse these disparate data entities together, you must join matching fields from the different data sources on the Source Creation tab of the Fusion data source. This is the key step to data fusion and must adhere to specific rules.

Joins are usually performed in-memory. However, if a data connector supports pushdown joins and the data to be joined comes from the same data connection, Composer pushes the join operation to the underlying data engines and allows those data stores to join the data instead. In addition, if the joins are inner joins and aggregate functions SUM, MIN, MAX, COUNT, DISTINCT COUNT, and aggregations are used in the data, the Composer engine intelligently pushes the aggregate queries to the underlying data engines, thus reducing the amount of data that needs to be processed. This aggregate pushdown occurs when joining data from the same or from different data connections. For more information about optimizing joins in your Fusion data connections, see Optimize Joins.

Because most joins are performed in-memory, a configurable limit has been placed on the number of records that can be processed from each joined source. This limit is initially set at 1,000,000 records per joined data source and can be configured by your Composer administrator or supervisor using the `qe.zengine.edc.rows.limit` property in the `query-engine.properties` file. See Manage The ComposerSymphony Query Engine. When this threshold is exceeded, no data is shown on the visuals containing the fused data and a message appears indicating that the threshold (maximum row number) is exceeded. If you find you are hitting this limit, use filters on the visual or dashboard to reduce the number of records processed and shown.

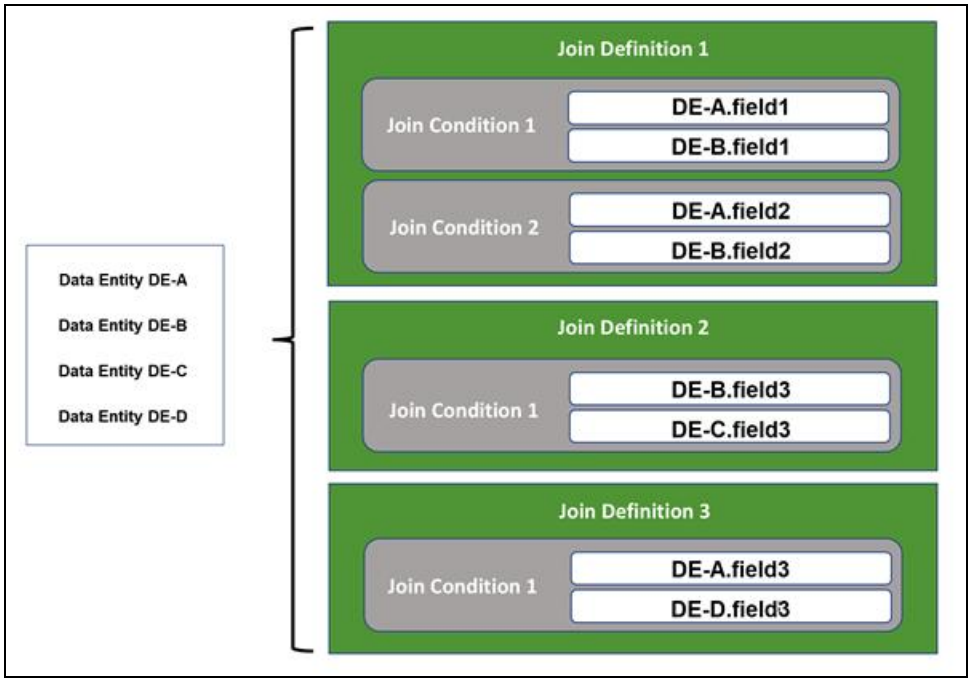
After you have joined the necessary fields from the data entities and saved your Fusion data source, you can visualize and explore the fused data in visuals and dashboards.

Data Fusion Join Rules

Data fusion joins are key to the success of your attempts to fuse data. So they must adhere to the rules described in this section.

You can explicitly specify the type of join that occurs for fusion: inner join, left outer join, or full outer join. Right outer joins are not supported. The join type can be selected for each pair of mapped fields in a join condition.

The following diagram depicts the relationship between the data entities you include in a Fusion data source and the join definitions and join conditions (mappings) you define in a Fusion data source.



The following rules must be adhered to for data fusion joins:

1. Joins are processed in the order in which they are specified in the UI. This affects the resulting data and the performance of the join.

2. Only a single join definition is allowed between two data entites. The join definition can contain multiple join conditions (mappings), previously called *forms*. Each mapping must contain exactly two fields. The fields used in the mapping must be of the same type and should contain the same kind of data.

   If there are more than two data entities in a Fusion data source definition, a single join definition can be specified for each unique combination of the included data entities. For example, if you have four data entities in your Fusion data source definition (sources A, B, C, and D), you can specify six join definitions, one each for these data entity combinations: A+B, A+C, A+D, B+C, B+D, and C+D.

3. Every data entity in a Fusion data source must be connected to at least one other data entity in the fused source. The data entities in a Fusion data source must be interconnected in some way.

   For example, if you have four data entities (A, B, C, and D) in your Fusion data source definition, you cannot define only a join between sources A and B and a second join between entities C and D. In addition, all the data entities must be connected in some way to one of the others. So the following join sequence would be correct because every data entity is included: A+B, B+D, D+C. However, the following join sequence would be incorrect because data entity C is entirely omitted: A+B, A+D, B+D.

4. Each subsequent join in the Fusion data source configuration must use a data entity from one of the previously defined joins.

   For example, if you have four data entities (A, B, C, and D) in your Fusion data source definition, if a join between entities A and B is the first join defined, the second join defined for the fused source must include either entity A or entity B and one of the other entities (C or D). So the following join sequence would be correct: A+B, B+D, D+C. However, the following join sequence would be incorrect because entity C and D are introduced before they have been linked to either entity A or entity B: A+B, D+C, B+D.

5. The data in mapped time fields must have the same granularity. Composer assumes that the granularity of a time field correctly matches the granularity of its data. For example, if one time field contains data in days and another time field contains data in seconds, they cannot be joined. The only way to join these two time fields would be to modify the granularity of the underlying data in the data store.

Filter Fused Data

Fused data can also be filtered in visuals and dashboards. Filtering fused data sets is possible in the following scenarios:

- Different visuals contain metrics from different fact tables but have common attributes.
- Filtering on a common attribute in one data fusion visual can be applied to other visuals in the dashboard.

Data Fusion Table Structures

The following table structures are used when data is fused using Composer's data fusion feature. For examples of these used in data fusion, see Data Fusion Use Cases.

Note: You can flag one or both entities used in creating a join for a fused data source as a dimensional entity. See Create A Fusion Source.

Lookup Table

A lookup table contains description fields that describe dimensional data. For example, a lookup table may contain information about sellers that includes the seller's username, first name, last name and contact information. But this table may not contain pertinent sales data linked to the seller. This table may need to be joined with a fact table to provide more insightful information. The following is an example of a lookup table.

Fact Table

A fact table stores measurements, metrics and other analytical information. A fact table typically contains two types of columns:

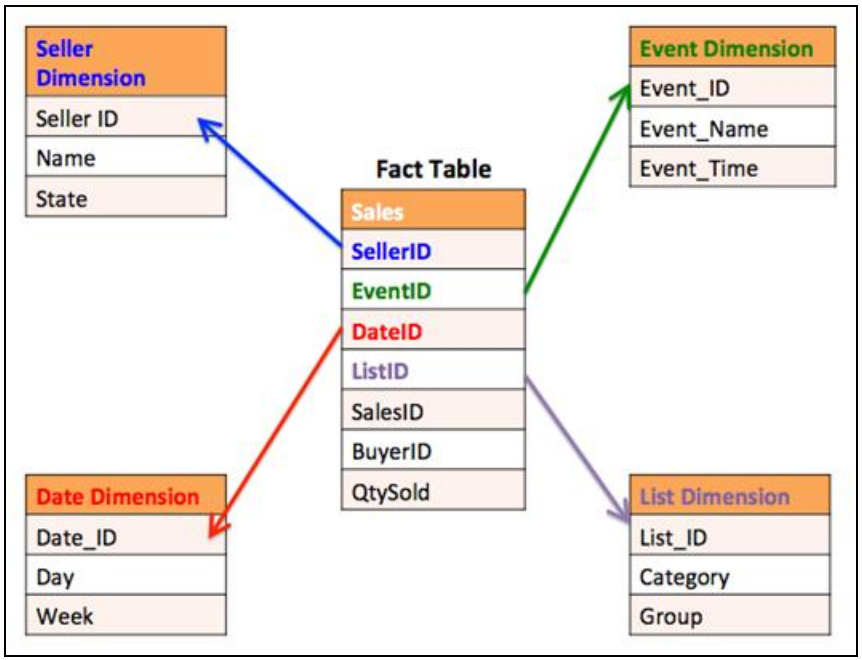
- Fact columns that contain the measures to analyze
- Dimensional keys to analyze the facts using different attribute contexts.

For example, analyzing quantities sold (fact) and price of tickets (fact) by event would be one context. Another context might be to analyze the same facts by seller. Fact tables may be missing the descriptions needed for categorizing and analyzing the data. Combining fact tables with lookup tables may be necessary to conduct proper data analysis.

The following figure illustrates a sample fact table containing sales information that includes the identifiers for each sales transaction along with the related event (and other information).

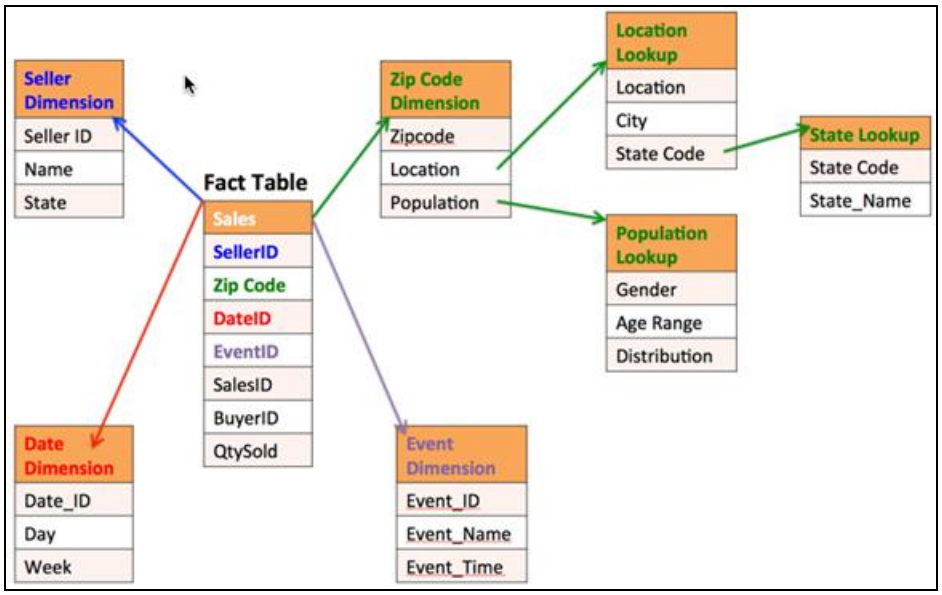
Star Schema

A star schema consists of one or more fact tables with references to dimension tables. Dimension tables store descriptions for key identifiers in the fact table as well as parent attributes that can allow aggregation of metric data to higher levels. This type of table structure is useful for rolling up and analyzing metric data using various dimensional attributes. The following figure depicts a sample star schema.



Snowflake Schema

A snowflake schema consists of one or more fact tables that have references to multiple dimension tables which in turn may connect to additional dimension tables. The logical arrangement of tables in a multidimensional database when displayed in a diagram resembles a snowflake shape. Like a star schema, dimension tables store descriptions for key identifiers in the fact table as well as parent attributes that can allow aggregation of metric data to higher levels. However, the difference is that the dimension table themselves may connect to additional dimension tables providing further information. The following figure depicts a sample snowflake schema.


Seller Dimension
Seller ID
Name
State
Fact Table
Sales
SellerID
Zip Code
DateID
EventID
SalesID
BuyerID
QtySold
Zip Code Dimension
Zipcode
Location
Population
Location Lookup
Location
City
State Code
State Lookup
State Code
State_Name
Population Lookup
Gender
Age Range
Distribution
Date Dimension
Date_ID
Day
Week
Event Dimension
Event_ID
Event_Name
Event_Time

Data Fusion Use Cases

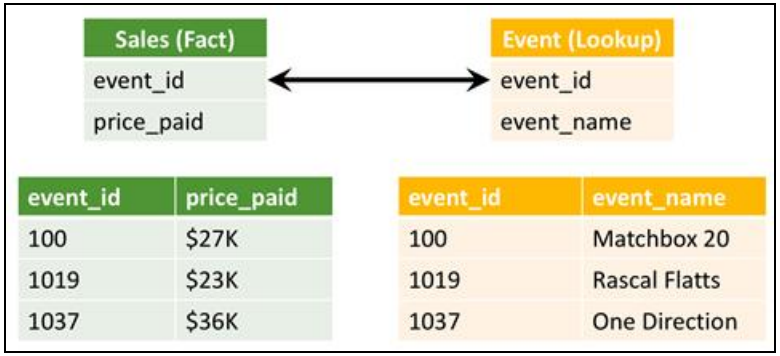
Composer supports the following data fusion use cases:

- Looking up descriptions from lookup tables
- Aggregating data using dimensional tables in a star or snowflake schema
- Joining multiple fact tables on common keys

We'll explore each use case in greater detail and then demonstrate how you can combine these different scenarios together to build unique data fusion data sets for your research and analysis.

Fact-to-Lookup Table Use Case

The lookup description is the simplest use case since it is joining a fact table to a lookup table using a common identity key. In this scenario, the description from a lookup table can be visualized along with metrics information from a fact table. The following diagram illustrates an example of a join between the event (lookup table) and the price paid. The common field between the two disparate tables is **event_id**.



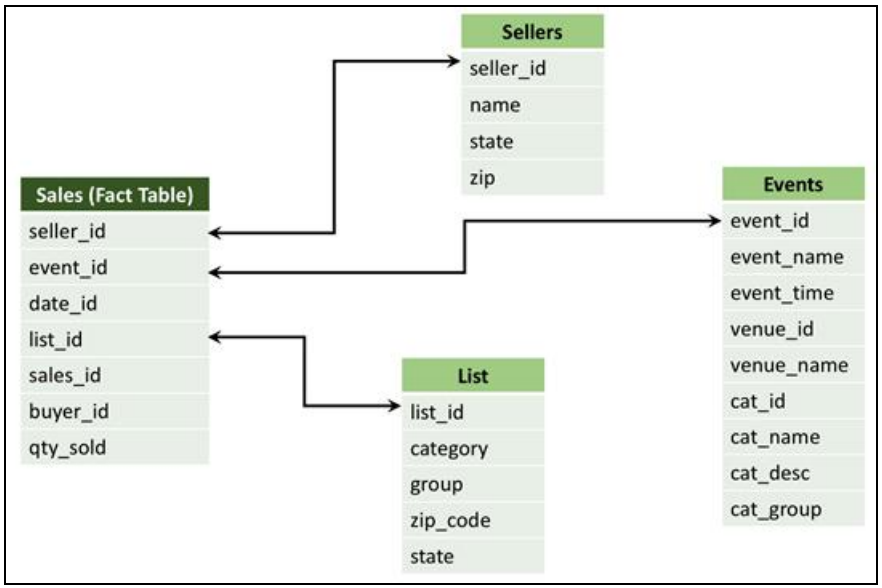
The resulting fused data includes the attributes from the lookup table and the metrics from the fact table. Any additional group-by attributes that are available from the lookup table are displayed as well.

Star Schema Table Use Case

A star schema table extends the capability of the lookup description use case to dimensional tables that contain attribute descriptions and higher level group-by attributes than on fact table. The fact table can be joined to several dimensional tables using common ID keys (limited to one key per join). This allows you to visualize the fused data set using group-bys and filtering on dimensional attributes sourced from the dimension table in conjunction with metrics from the fact table.

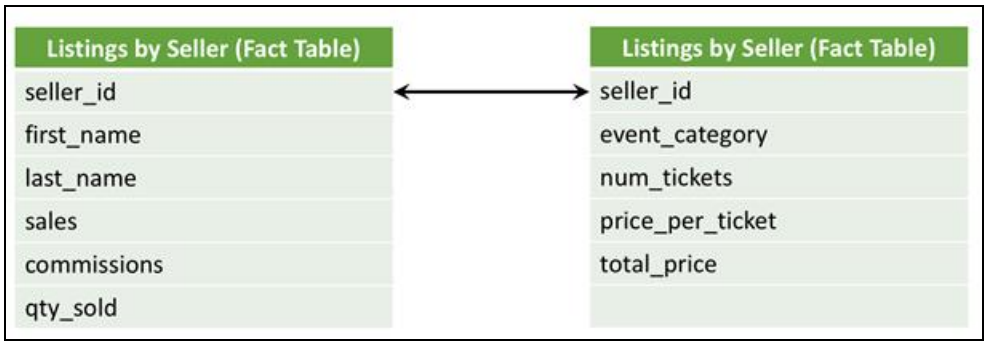
For example, the following diagram illustrates a star schema table. The Sales table is the fused data set resulting from the following disparate data repositories: seller details from the Sellers data entity, event information from the Events data entity, and listing details from the List data entity.

For example, the following diagram illustrates a star schema table. The Sales table is the fused data set resulting from the following disparate data sources: seller details from the Sellers data source, event information from the Events data source, and listing details from the List data source.



Multiple Fact Table Use Case

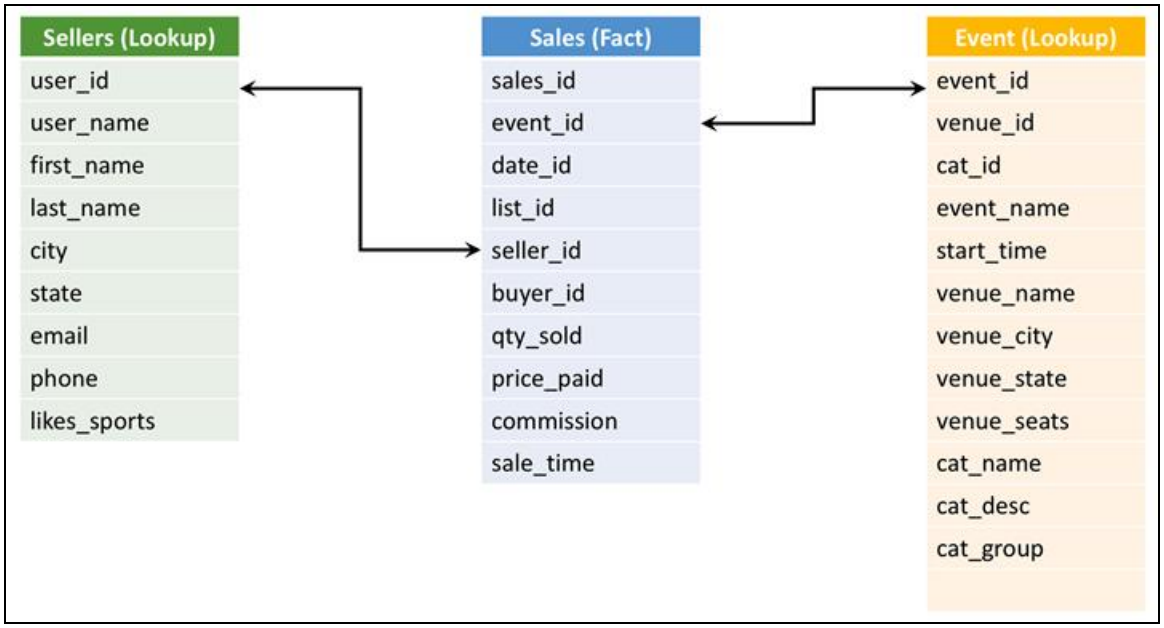
Composer supports the capability to join different fact tables together. Similar to the lookup description use case, multiple fact tables are joined using a common key. As a result, metrics from different tables can be displayed on the same visual with common keys as the Group By keys. The following diagram shows two disparate fact tables being joined under the common attribute - **seller_id**.



If a join is attempted between tables that have a one-to-many or many-to-many relationship, the table metrics will be duplicated.

Combination Use Case

You can also fuse data using a combination of these use cases. For example, you can create a lookup to fact to lookup join (as shown in the diagram below) to explore the sellers, their ticket sales and event information all on one visual. In this example, seller information is located in a lookup table, sales data is housed in a fact table, and event details are stored in a lookup table. The common keys connecting these data sets are **user_id**, **seller_id**, and **event_id**.



Create a Fusion Source

Fusing data in a source in Composer is very similar to adding other data sources. Add multiple entities to a new or existing source, then use the Join Definition work area to identify and specify the data from those entities you want to join.

For an overview of Composer's data fusion capability, see Fuse Data Sources.

Important: You must log in as a user with the **Administer Sources** or **Create New Data Sources** privilege, or write permission for the source for which you want to create a join.

Before You Start

Before you attempt to create a fusion source, verify you can access the connections. If not, create them. See Manage Data Store Connections.

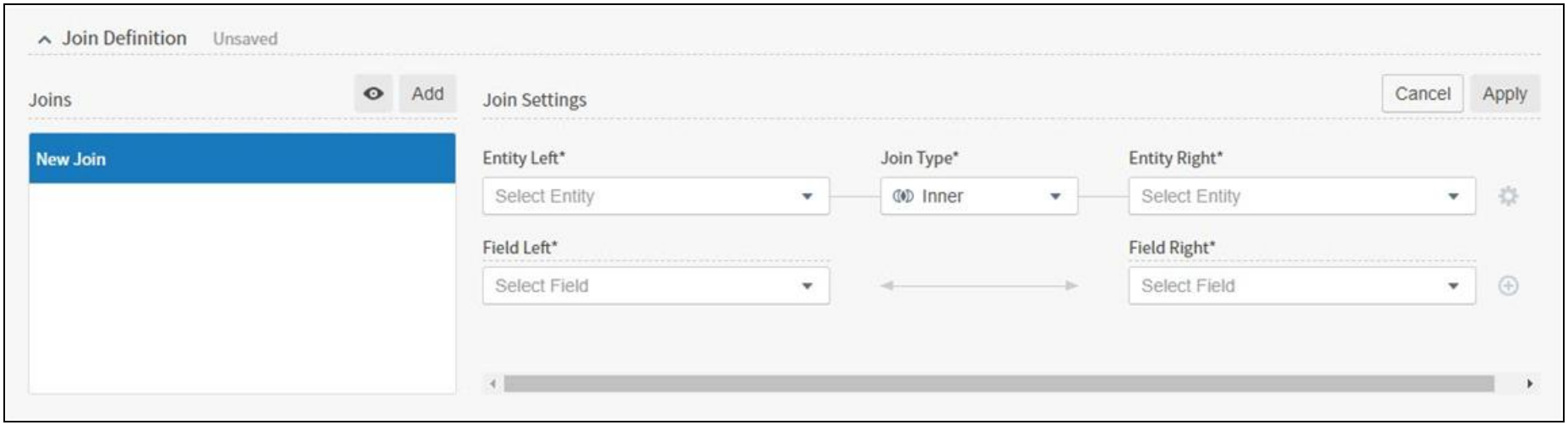
Configure a Fusion Source

To create a Fusion source, add multiple data entities to a new or existing source, then use the Join Definition work area to specify the fields and joins included in your fused source.

Add Joins to a New or Existing Source

1. Log in as a user with the **Administer Sources** or **Create New Data Sources** privilege, or write permission for the source for which you want to create a join.
2. Select **Sources** on the UI menu (). The Sources page appears.
3. Create or edit an existing source, adding multiple entities **From Connection** or **From File**. You will only see the connections you have read permission for. See About Source Permissions.

   Note: If your source contains multiple data entities, you must use all entities in a join to save the source.
4. Select **Add** in the **Join Definition** work area. A Join work area opens.



5. Define your entities, join type, and fields.

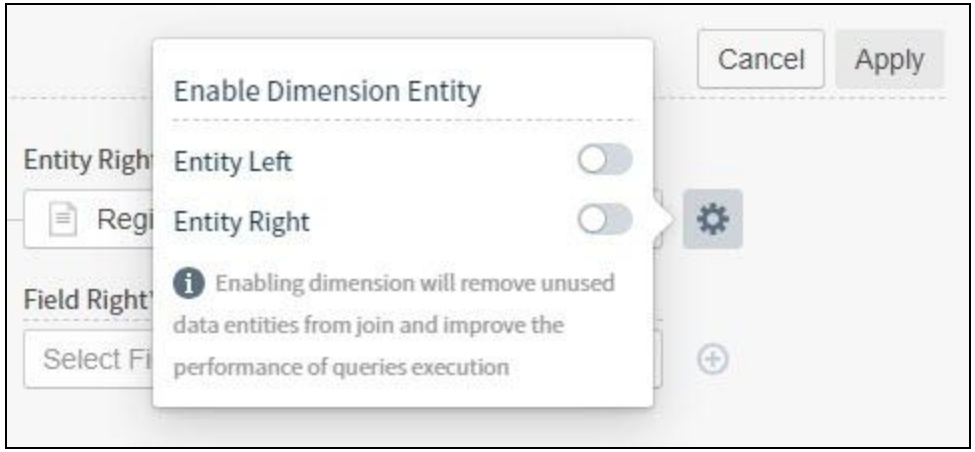
> **Note:** Ensure you have matching fields across the data sources you are joining.

i. **Entity Left**: Select an available entity from the data entities you defined.

ii. **Join Type**: Select Left, Inner, or Full Outer.

iii. **Entity Right**: Select an available entity from the data entities you defined.

iv. **Enable Dimension Entity**: Select the settings icon to enable dimension for one or both entities. This improves the performance of queries execution by removing unused data entities from the join.



v. **Field Left**: Select at least one field from this entity. You can add multiple fields by selecting the add field icon, or remove fields by selecting the remove icon.

vi. **Field Right**: Select at least one field from this entity. You can add multiple fields by selecting the add field icon, or remove fields by selecting the remove icon.

6. Select **Apply** to finish creating the join. Composer names the join and adds it to the Joins list. Remove joins by selecting the remove icon.

You can also view the relationships of your joins and add more joins in a visualization. See Visualize Joins.

7. After creating your joins, select **Preview Source** to preview your data, or select **Save Source** to save your updated source.

8. As needed, update the default settings on the Fields tab, Cache tab, or Global Settings tab.

Fields Tab for Fusion Sources

Use the Fields tab to manage your fused source data: rename field Labels, add derived fields, or select the custom metrics tab add a custom metric. If your fused data sources include duplicate field names, Composer appends a number to the duplicate field name.

Cache Tab for Fusion Sources

Use the Cache tab to enable or disable caching of aggregated results of queries for this source. See How Composer Caches Data.

Global Settings Tab for Fusion Sources

Use the Global Settings tab to configure settings for new visuals for this fused source. Not all visual styles are available for Fusion data sources. See Data Fusion Limitations.

Visualize Joins

You can use the provided work area to create joins, or view and create joins in a move visual way.

Zoom in or out in this work area, or use the mini map to navigate among the various tables that make up your joins.

View or edit a joins for a fused data source

1. Create or edit a fusion source that uses a join.

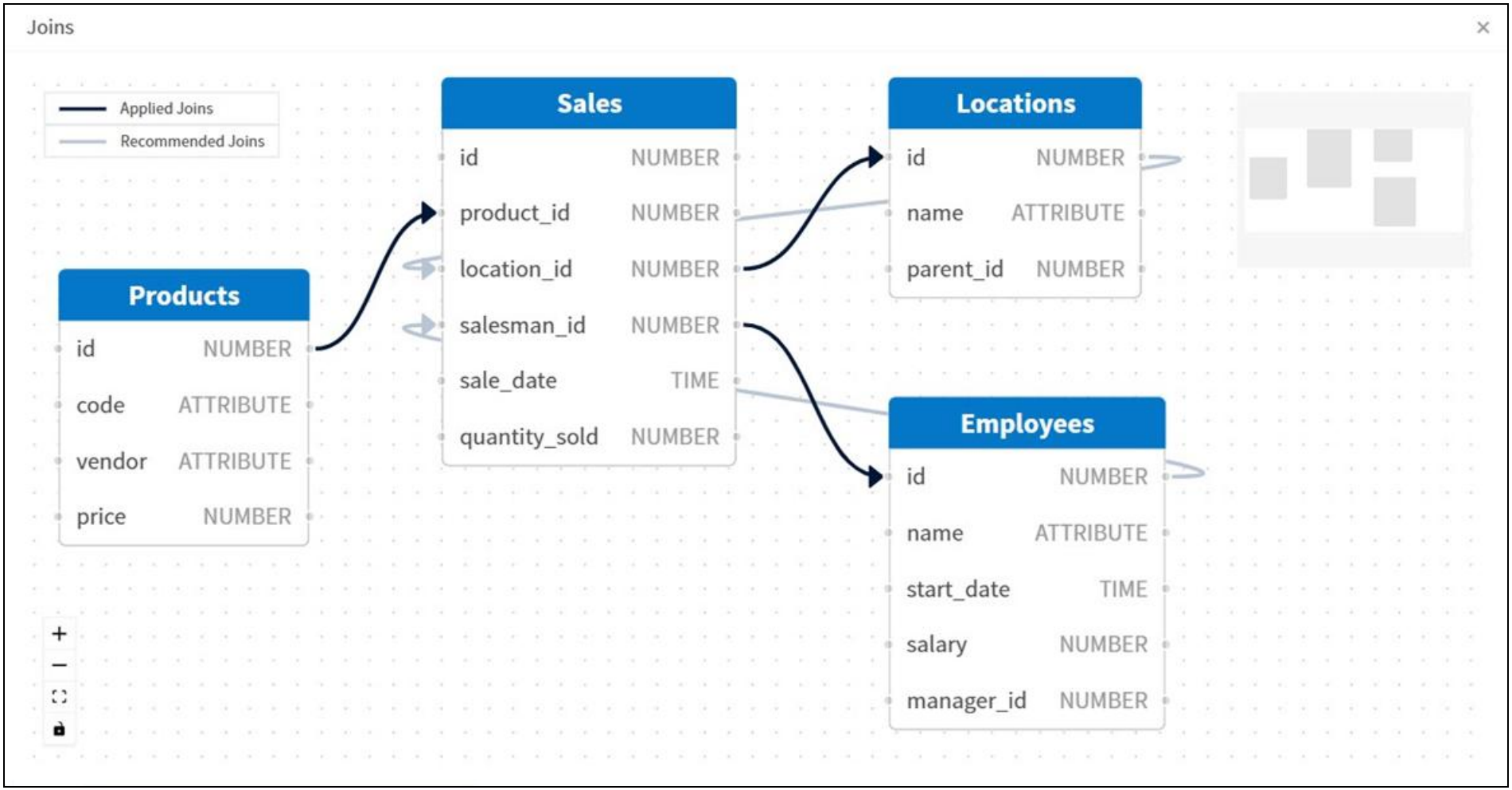
2. Select the view icon in the Joins work area to open a visualization of existing joins.

3. Optionally, draw more joins in this work area, then **Save** your changes. New joins are added to the list of those in Join Settings.

   Note: To remove a user-applied join, double-click to select the join, then select the backspace key. The relationship is removed.

Important: If you add joins that create one-to-many relationships here, Composer may return an error that prevents use of the data in a visual. For best results, when you create a one-to-many relationship with a specific left entity, any additional joins must refer to that table as the right entity. See Recommended Joins.

Note: For Postgres connections, both tables and data relationship information are read from the connection. Other supported connections include table information but do not read relationship information. See Connector Support for Schema Visualization. No information is provided for unsupported connections.



Recommended Joins

Composer provides some visual guidance for recommended joins. You can add these suggested joins, or create your own.

Note: Recommended Joins may present one-to-many or many-to-one relationships that when implemented return an error. You can still implement the recommended joins: when you create a one-to-many relationship with a specific left entity, any additional joins must refer to that table as the right entity.

For example:

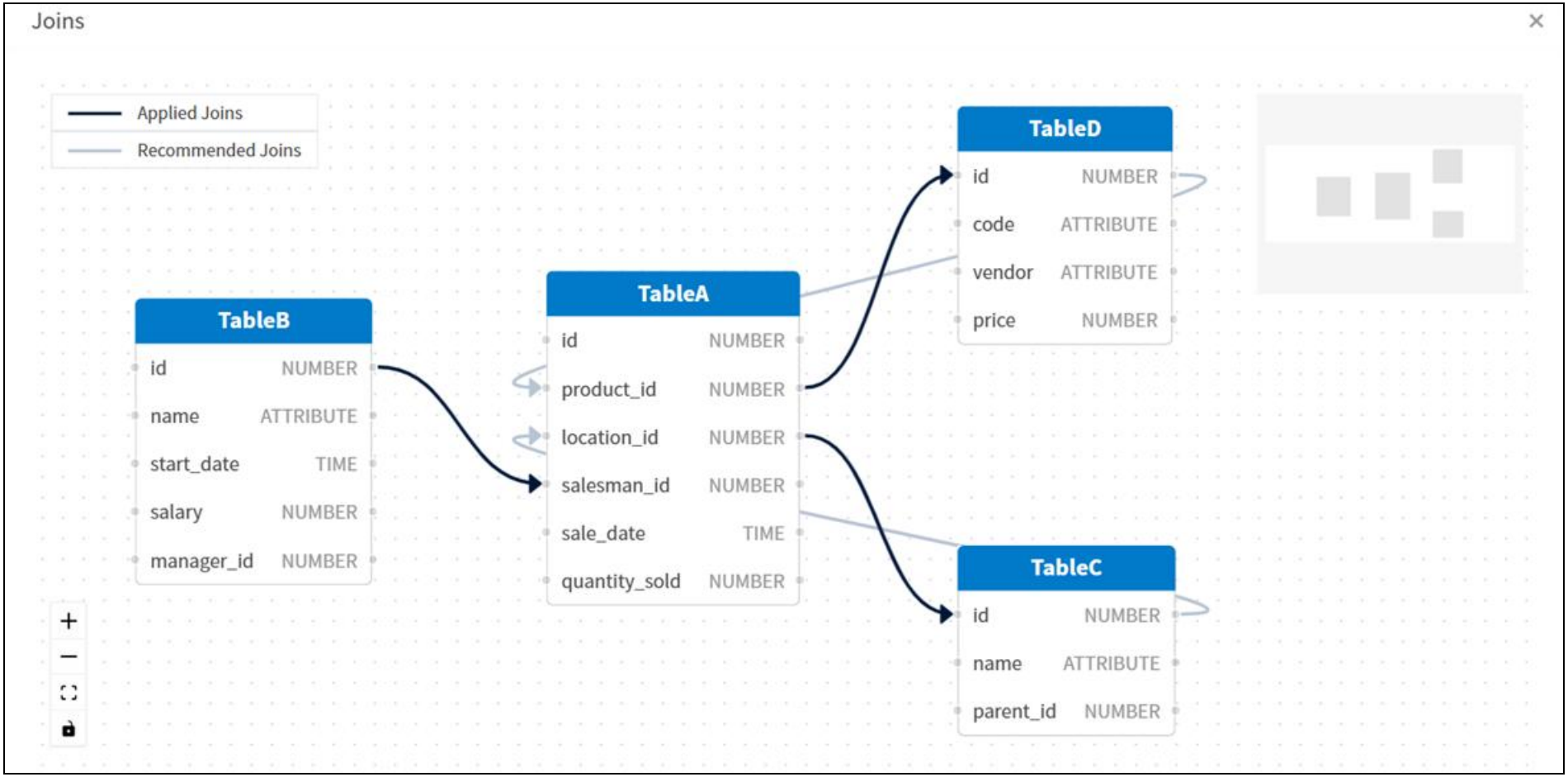


Table A has relationships with **Table B**, **Table C**, and **Table D**.

- After defining a relationship with **Table A** as the left entity, include **Table A** as the right entity in additional joins.

- If you are using the visual interface, position only one of the other tables (**Table B**, **Table C**, or **Table D**) on the left side of **Table A**. Remaining tables should be placed to the right of **Table A**.

Optimize Joins

Data fusion joins are processed in the order in which they are specified in the UI. This affects the resulting data and the performance of the join. In addition, the type of join you select affects whether fusion processing time is optimized.

Joins are usually performed in-memory. However, when join processing can be pushed down to the data connectors to perform, fusion processing time is greatly reduced. Composer supports pushdown join processing in the following ways.

- If a data connector supports pushdown joins and if the data to be joined comes from the same data source connection, Composer pushes the join operation to the underlying data connectors and allows them to join the data instead. Several examples are given later.

- If the type of join is an inner join and aggregate functions SUM, MIN, MAX, or COUNT are used in the data, the Composer engine intelligently pushes the aggregate queries to the underlying data connectors, thus reducing the amount of data that needs to be processed. In these cases, the aggregation is performed first before the data is joined. This aggregate pushdown occurs when joining data from the same or from different data sources.

Because most joins are performed in-memory, a configurable limit has been placed on the number of records that can be processed from each joined source. This limit is initially set at 1,000,000 records per joined data source and can be configured by your administrator or supervisor using the `qe.zengine.edc.rows.limit` property in the `query-engine.properties` file. See Manage The ComposerSymphony Query Engine. When this threshold is exceeded, no data is shown on the visuals containing the fused data and a message appears indicating that the threshold (maximum row number) is exceeded. If you find you are hitting this limit, use filters on the visual or dashboard to reduce the number of records processed and shown.

Support for this feature by connector is shown in the following table.

Key:Y - Supported; **N** - Not Supported; N/A - not applicable

Connector	Supported?
Amazon Redshift	**Y**
Amazon S3	**N**
Apache Drill	**Y**
Apache Phoenix	**N**
Apache Phoenix Query Server (QS)	**N**
Apache Solr	**N**
BigQuery	**Y**
Business Central Jet	**N**
Cloudera Impala	**Y**
Cloudera Search	**N**

Connector	Supported?
Couchbase	**N**
Dremio	**N**
Elasticsearch 7.0	**N**
Elasticsearch 8.0	**N**
File Upload	**Y**
HDFS	**N**
Hive	**Y**
Jira	**N**
MemSQL	**Y**
Microsoft SQL Server	**Y**
MongoDB	**N**
MySQL	**Y**
Oracle	**Y**
PostgreSQL	**Y**
Python	**N**
Real Time Sales	N/A
Salesforce	**N**
SAP Hana	**N**
SAP S/4HANA	**N**
SAP IQ	**N**
Spark SQL	**Y**
Snowflake	**Y**
Teradata	**Y**
TIBCO DV	**Y**
Trino	**N**
File Upload (Upload API)	**Y**
Vertica	**Y**

Examples

Example 1

In the following two fusion data sources, **Fusion Data Source 1** will be pushed to the Impala connector to perform, whereas **Fusion Data Source 2** will be performed in-memory because the two data sources use different Impala connections.

```
        Fusion Data Source 1 join:
Impala-Data-Source1-using-Impala-Connection-1
Impala-Data-Source2-using-Impala-Connection-1
```

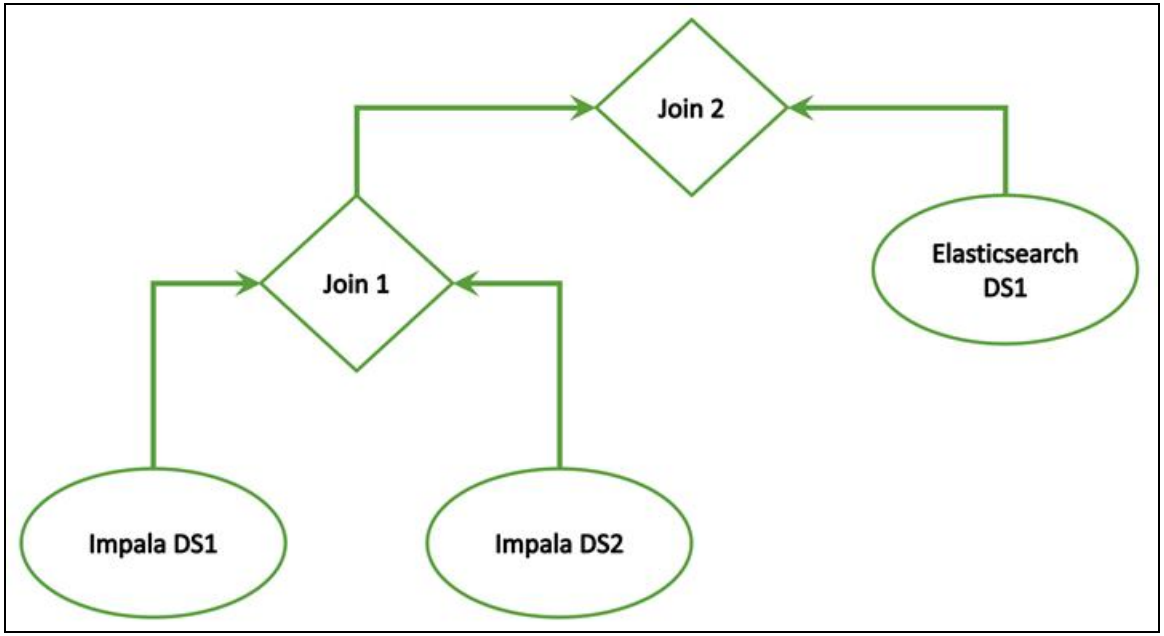
```
        Fusion Data Source 2 join:
Impala-Data-Source1-using-Impala-Connection-1
Impala-Data-Source3-using-Impala-Connection-2
```

Example 2

The following multisource fusion example has more than one join defined. Assuming both joins are inner joins, **join 1** will be performed by the Impala connector and **join 2** will be performed in-memory.

```
        Fusion Data Source
inner join 1:
   Impala-Data-Source1-using-Impala-Connection-1
   Impala-Data-Source2-using-Impala-Connection-1
inner join 2:
   Impala-Data-Source1-using-Impala-Connection-1
   Elasticsearch-Data-Source1-using Elasticsearch-Connection-1
```

The following diagram depicts the relationship of the joins in the fused data source:

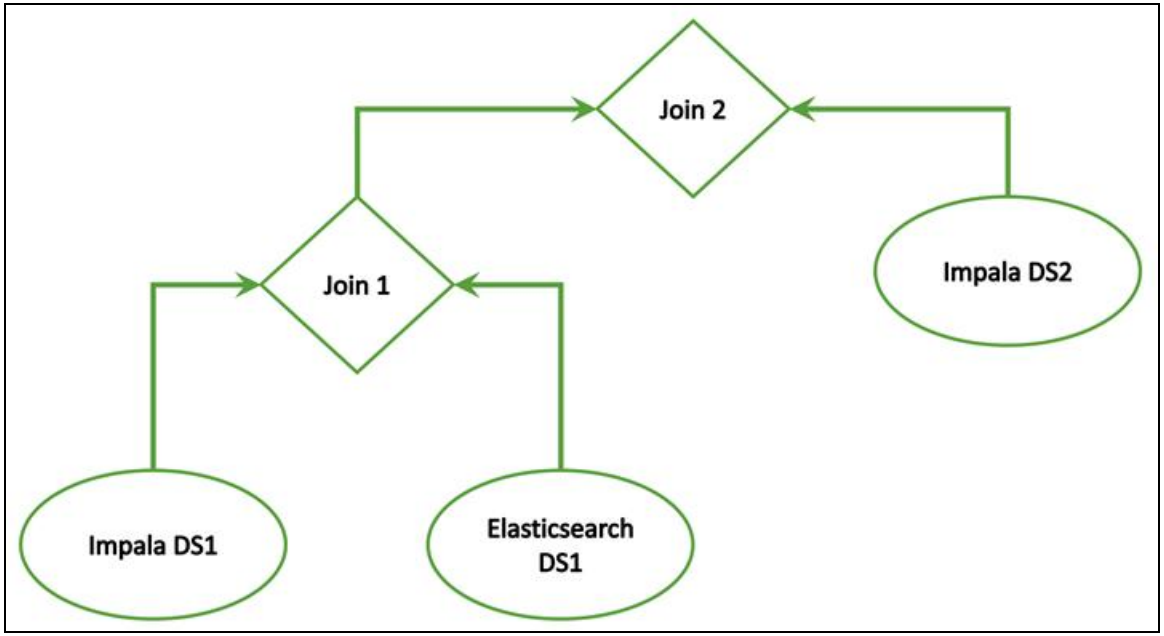


Example 3

If the join order from Example 2 is switched as shown below and if the first join is changed to a left join, neither join can be performed by data connectors. They are both performed in-memory.

```
        Fusion Data Source
left join 1:
   Elasticsearch-Data-Source1-using-Elasticsearch-Connection-1
   Impala-Data-Source1-using-Impala-Connection-1
inner join 2:
   Impala-Data-Source1-using-Impala-Connection-1
   Impala-Data-Source2-using-Impala-Connection-1
```

The following diagram depicts the relationship of the joins in the fused data source:


Join 2
Join 1
Impala DS2
Impala DS1
Elasticsearch
DS1

Hierarchical Fields and Structures

Note: Hierarchical fields are enabled by default at the server level. Work with Technical Support to disable.

Composer now supports common hierarchical structures you can use to visualize and organize your data. Hierarchies create parent-child tree-based, and field-level hierarchical structures using your source data.

When you organize your data in Composer in hierarchies, it's easier to access for analytical processing, for data traversal, and keyword searching. To view your data in a hierarchical format, create a data source that defines a hierarchy using data in an adjacency list structure, add one or more hierarchy fields to your source, and use the hierarchical field in the context of a group or filter on pivot table visuals or other visuals.

See the following topics for more information:

- Define A Hierarchical Source
- Edit A Hierarchical Source
- Define A Hierarchy Field For Your Source
- Filter by Hierarchy Field
- Apply Hierarchical Groups
- Apply Hierarchical Filters To A Pivot Table Visual
- Apply Hierarchical Filters To Visuals
- Sunburst

Limitations

- Cycles in hierarchies are not supported.
- Hierarchical groups are available in the Pivot Table visual as a single hierarchical group in rows.

- Hierarchical groups are available as multiple groups in the Sunburst visual.
- When you use a hierarchical group, simple metrics are calculated for all hierarchy levels by default. Specify **Use Rollup** to roll data up to parent levels as needed in pivot tables.
- Data sharpening and viewing data changes using the time bar are not available.
- By default, simple metrics are included; enable **Use Rollup** to roll data up to the parent levels in pivot tables.
- Aggregate filters are not available.
- Pagination is not supported in pivot table visuals that use hierarchical groups. To view your entire hierarchy, adjust the rows per page limit to accommodate all items of the hierarchy.

Define a Hierarchical Source

Note: Hierarchical fields are enabled by default at the server level. Work with Technical Support to disable.

Create or edit a source to include a facts table and at least one lookup table to create your hierarchical source.

Define a New Hierarchical Source

1. Log in as a user with the **Administer Sources** or **Create New Data Sources** privilege.
2. Select **Sources** on the UI menu ( ) or the top-level navigation menu, or select the **Sources** box on the Home page The Sources page appears.
3. On the Sources page, select the **Create Source** button. The Source Creation work area opens.
4. Name your source, then add the first data entity, including appropriate connection, schema, and entity options or Custom SQL.

   Note:
   The first data entity can be, but does not have to be, the facts table.

5. Add the second data entity, your hierarchy lookup table, including appropriate connection, schema, and entity options or Custom SQL.
6. Select **Add** in the Join Definition work area to connect the hierarchy lookup table to the facts table.
7. If only one hierarchy is used, we recommend you use a left join from a hierarchy lookup table to a facts table. To configure multiple hierarchical tables on a single source, contact Technical Support.

i. Select a left join to preserve the contents and structure of the hierarchical tree, regardless of the contents of the facts table.

ii. If you select an inner join, the hierarchy is limited to data in the facts table. This may break your hierarchy structure, separating nested items from parents, making the nest items top-level items.

Note: You can visualize your joins by selecting the view icon in the Joins work area. See Visualize Joins.

8. Select **Apply** to create the join, then **Save Source** to save your source.

Next, define a hierarchy field for your source. SeeDefine a Hierarchy Field for Your Source.

Edit a Hierarchical Source

Note: Hierarchical fields are enabled by default at the server level. Work with Technical Support to disable.

You can edit an existing source to pair a facts table and at least one lookup table to use hierarchical data.

Edit a source

1. Log in as a user with the **Administer Sources** or **Create New Data Sources** privilege.
2. Select **Sources** on the UI menu ( ) or the top-level navigation menu, or select the **Sources** box on the Home page. The Sources page appears.
3. On the Sources page, select a source to edit. The Source Creation work area opens.
4. In this example, the existing data entity is the fact table. Select **Add** to add a data entity to use as a hierarchical lookup table.
5. Select **Add** in the Join Definition work area to connect the hierarchy lookup table to the facts table.
6. If only one hierarchy is used, we recommend you use a left join from a hierarchy lookup table to a facts table.

i. Select a left join to preserve the contents and structure of the hierarchical tree, regardless of the contents of the facts table.

ii. If you select an inner join, the hierarchy is limited to data in the facts table. This may break your hierarchy structure, separating nested items from parents, making the nest items top-level items.

Note: You can visualize your joins by selecting the view icon in the Joins work area. See Visualize Joins.

7. Select **Apply** to create the join, then **Save Source** to save your source.
8. Next, define a hierarchy field for your source.

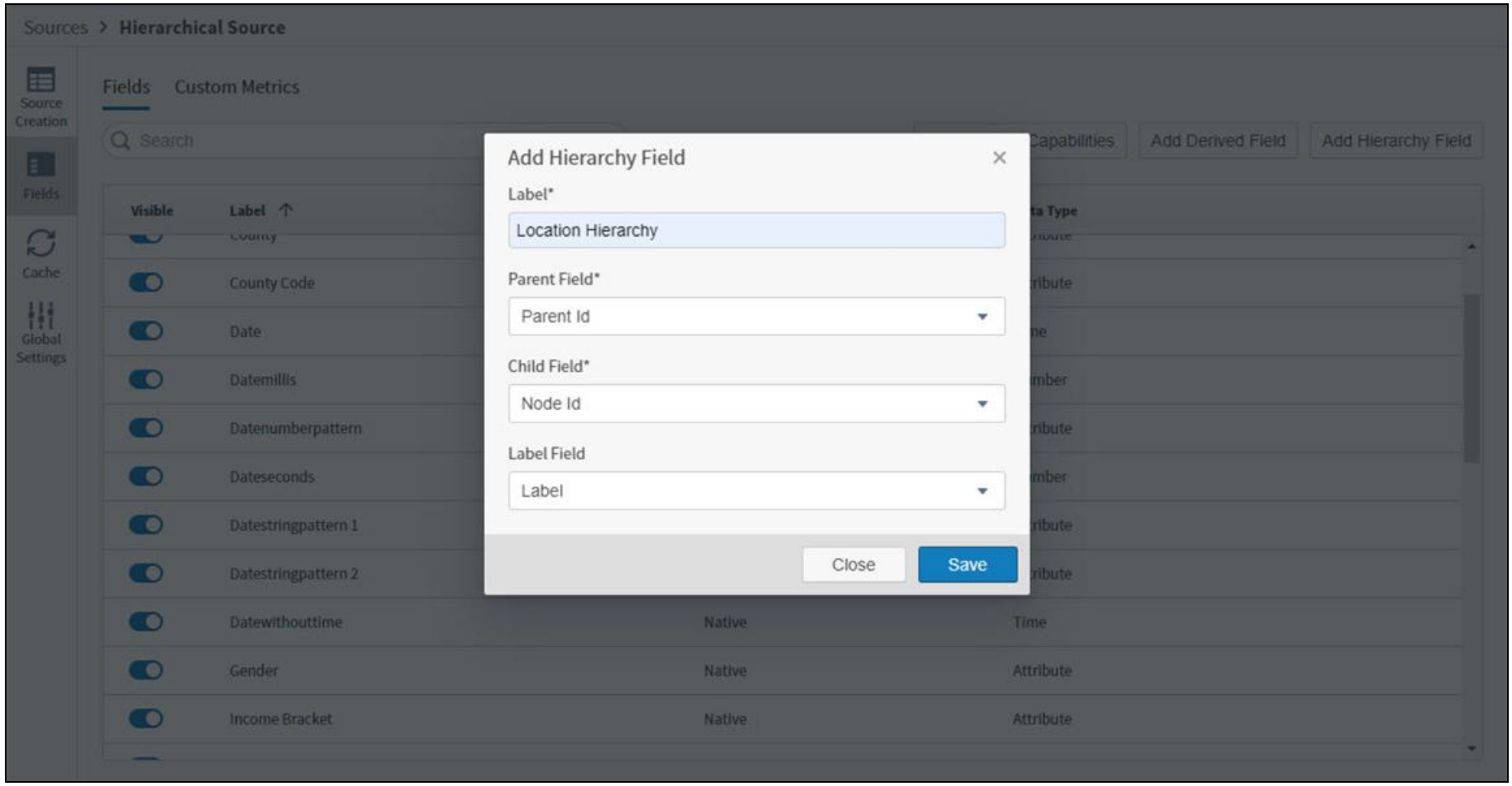
Define a Hierarchy Field for Your Source

Note: Hierarchical fields are enabled by default at the server level. Work with Technical Support to disable.

After you have created your source, define a hierarchy field for your source. Once you've defined a hierarchy field, you can create a table visual or pivot table visual that uses your hierarchical data.

Add a Hierarchy Field

1. Open the Fields tab of your hierarchical source.
2. Select **Add Hierarchy Field**. The Add Hierarchy Field work area opens.



3. Select a **Parent Field**, a **Child Field**, and optionally select a **Label Field** if applicable. If you do not select a Label Field, the value selected for Child Field is used.

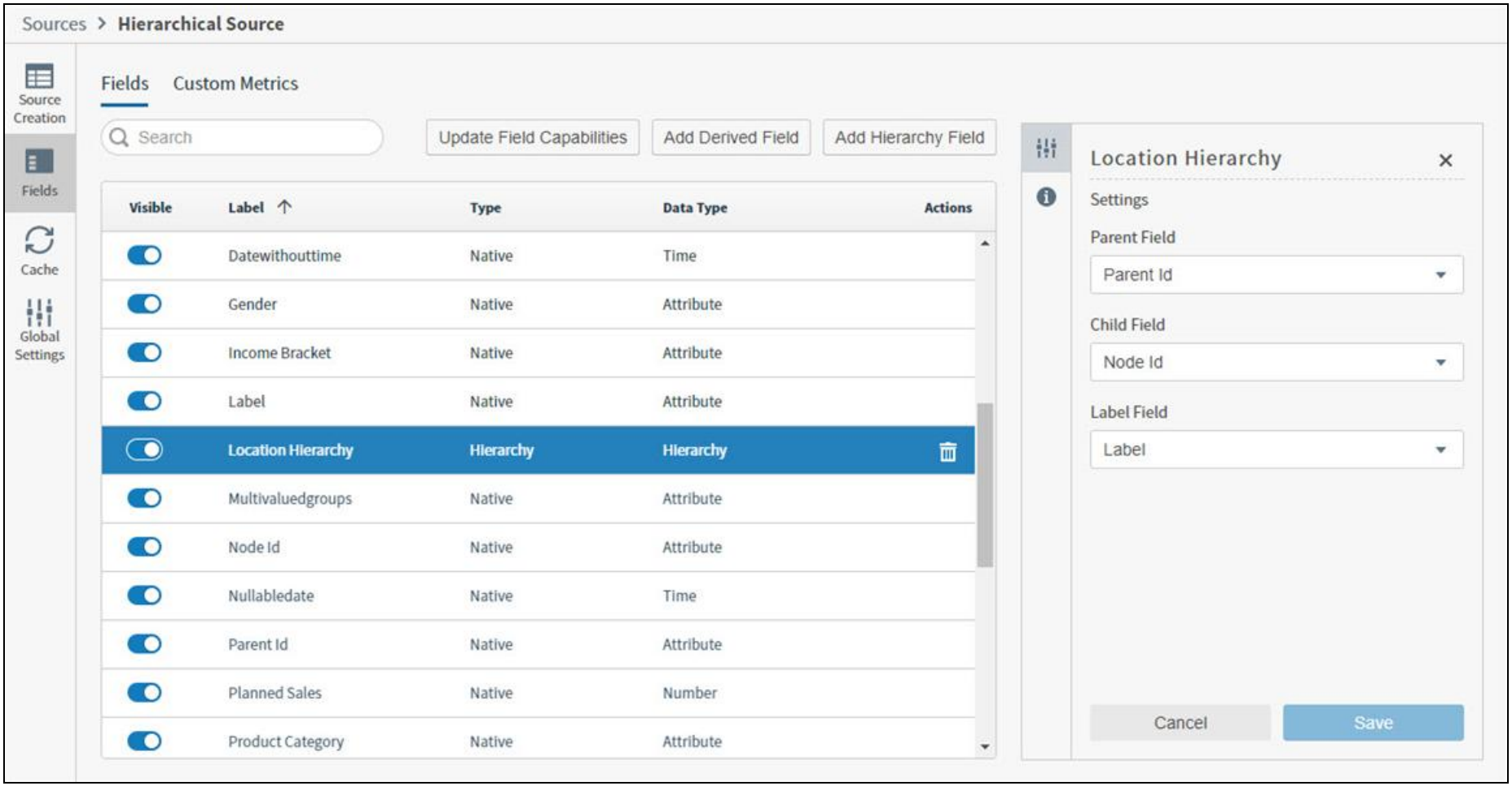
i. A parent field references another row from the same table using a unique identifier or unique name. This establishes the parent-child relationship. The parent and child fields must be the same field type, either an attribute or number.

ii. A child field contains a unique identifier or unique name of an element in the hierarchical tree, unique within the scope of the table. The parent and child fields

must be the same field type, either an attribute or number.

iii. A label field contains a user friendly name of a hierarchy element. This optional field is visible on the visual if configured. This field does not need to be unique.

4. Select **Save** to save your new hierarchy field.



5. Optionally, disable the Time Bar on the General Settings tab, and select **Save Settings** to save your changes.

After you've defined a hierarchical field, you can:

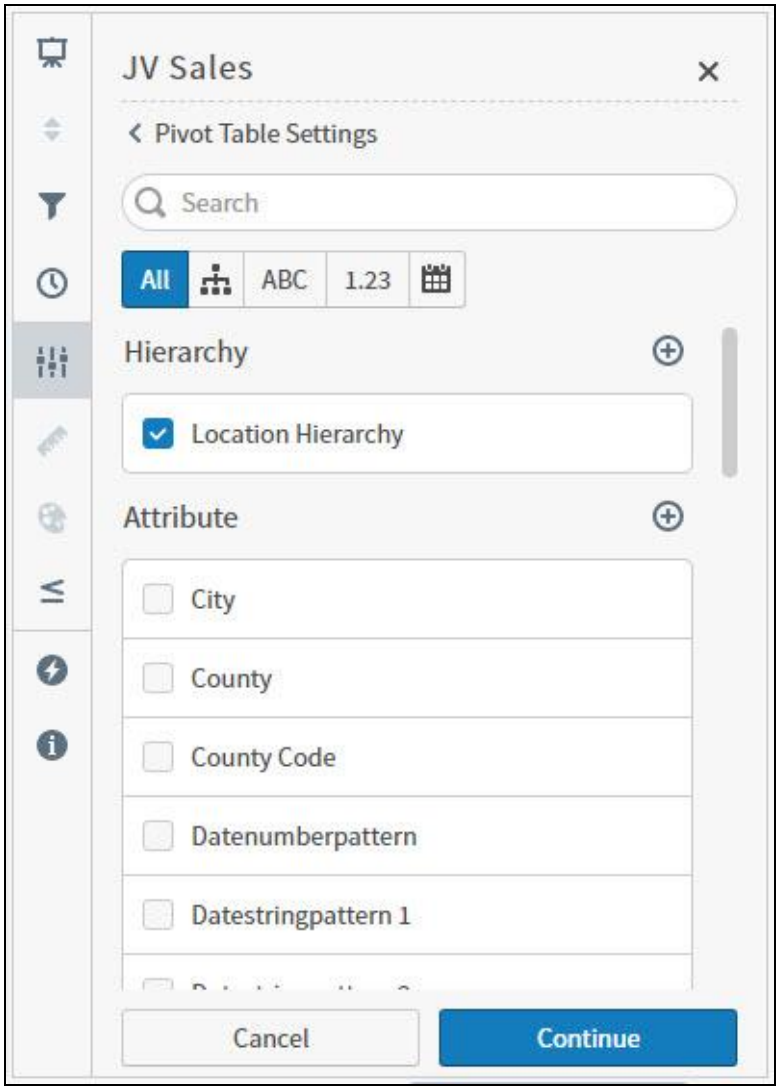
- Create a table visual or pivot table using the hierarchical data referenced by this hierarchy field.
- Filter data using this hierarchical field in a filter or filter snippet.

Apply Hierarchical Groups

Note: Hierarchical fields are enabled by default at the server level. Work with Technical Support to disable.

Use a Hierarchical Group in a Pivot Table

1. Create a pivot table visual from your hierarchical data source.
2. Select the Settings sidebar menu, then select **Edit Row Groups** ().



3. Select a hierarchical field to use, then select **Continue** to define column groups and metrics for your visual.

> **Note:** You can't use other fields in row groups if you have selected a hierarchical field. The hierarchical field will be the only row in the rows dimension.

4. Optionally, select **Edit Column Group** () to add a field or fields to column groups.

5. Select metrics to use in your pivot table. Optionally, enable **Rollup** for the metrics, if supported. This shows the rolled up value of all child values at each hierarchy level.

6. Select **Apply** to apply your changes. You can now expand and collapse the hierarchical data you defined in your visual.

> **Note:** Not all data in your sources may display in your visual due to rows per page settings. The default display of rows is 200, including the parents and root node. Adjust the **Rows per Page** in the Display Settings.

Apply Hierarchical Filters to a Pivot Table Visual

Note: Hierarchical fields are enabled by default at the server level. Work with Technical Support to disable.

Apply a Hierarchical Filter to a Pivot Table

1. Open the Filter sidebar menu of a pivot table that uses a hierarchical group.
2. Select **Add Filter**. The Add Filter work area opens.
3. On the Row tab, select an available Hierarchy to filter. The Select Values work area opens.

   The default Operator, **Equals or Descendants Of**, is selected. When used in this filter, Composer selects data for nodes equal to the selected node, and its descendants. You can select multiple nodes in a hierarchical filter, but not deselect child nodes.

   If you select the Operator **includes**, Composer selects data for nodes equal to the selected node, and its descendants. You can select multiple nodes in a

hierarchical filter, and optionally deselect child nodes.

4. Select one or more nodes to filter your data. At least one filter value is required.

i. Use the **Search** box to find specific nodes.

ii. Select the node indicators to expand and collapse nodes, or select and deselect **Expand All** to expand and collapse all nodes.

5. Select **Continue** to apply your changes.

6. Optionally, add any other filters you need.

7. Select **Apply** to apply your filters to the pivot table.

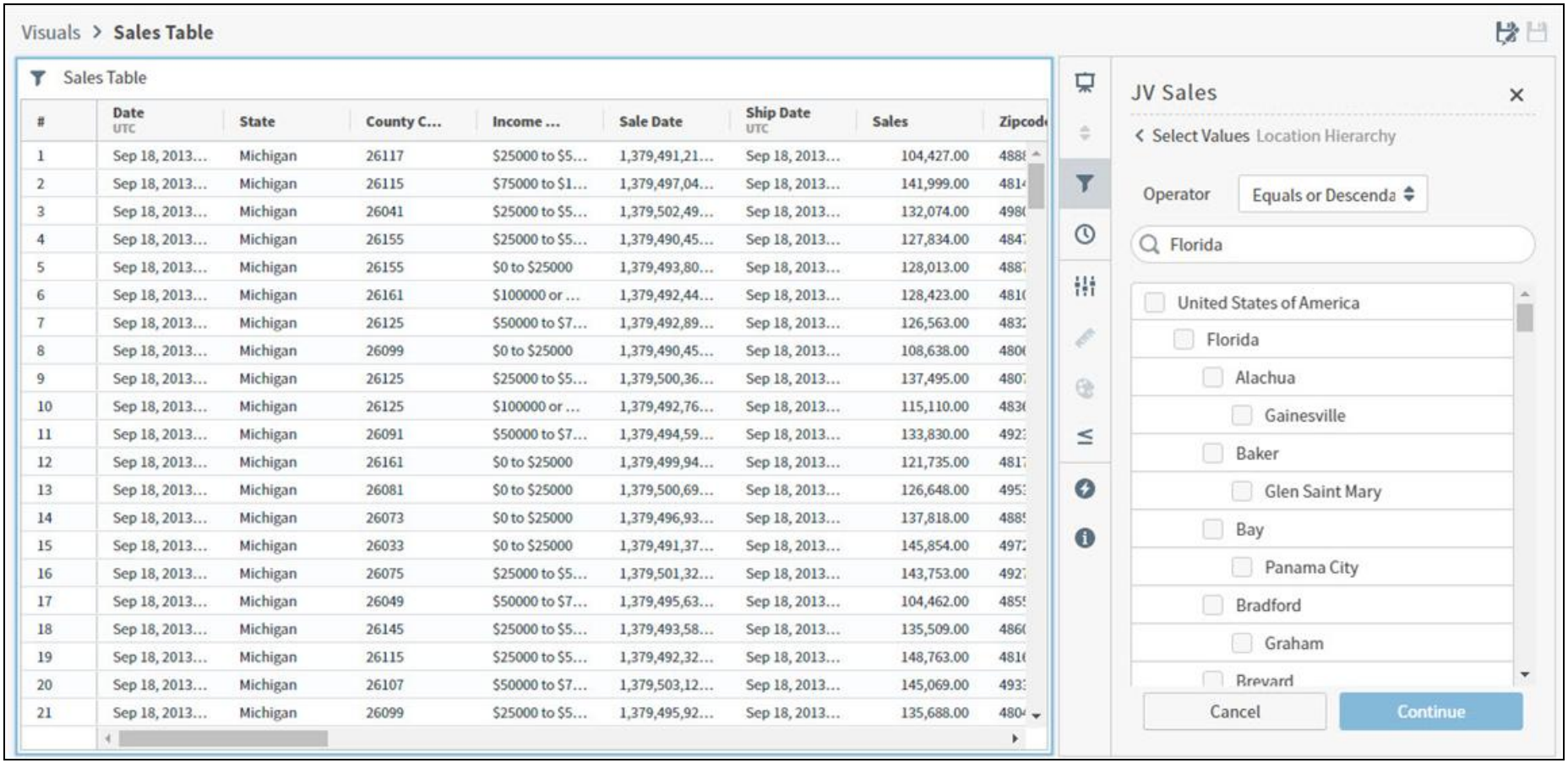
Apply Hierarchical Filters to Visuals

Note: Hierarchical fields are enabled by default at the server level. Work with Technical Support to disable.

Apply a Hierarchical Filter to a Table Visual

1. Open the Filter sidebar menu of a visual that uses a hierarchical data source.
2. Select **Add Filter**. On the row tab, select an available Hierarchy to filter. The Select Values work area opens.

   The default Operator, **Equals or Descendants Of**, is selected. When used in this filter, Composer selects data for nodes equal to the selected node, and its descendants.



3. Select one or more nodes to filter your data. At least one filter value is required.

i. Use the **Search** box to find specific nodes.

ii. Select the node indicators to expand and collapse nodes, or select and deselect **Expand All** to expand and collapse all nodes.

4. Select **Continue** to apply your changes.

5. Optionally, add any other filters you need.

6. Select **Apply** to apply your filters to the visual.