Cloudera Octopai Data Lineage 1.0.0

Metadata Sources

Date published: 2025-10-09
Date modified: 2025-10-20

CLOUDERA

https://docs.cloudera.com/

Legal Notice

© Cloudera Inc. 2026. All rights reserved.

The documentation is and contains Cloudera proprietary information protected by copyright and other intellectual property rights. No license under copyright or any other intellectual property right is granted herein.

Unless otherwise noted, scripts and sample code are licensed under the Apache License, Version 2.0.

Copyright information for Cloudera software may be found within the documentation accompanying each component in a particular release.

Cloudera software includes software from various open source or other third party projects, and may be released under the Apache Software License 2.0 (“ASLv2”), the Affero General Public License version 3 (AGPLv3), or other license terms. Other software included may be released under the terms of alternative open source licenses. Please review the license and notice files accompanying the software for additional licensing information.

Please visit the Cloudera software product page for more information on Cloudera software. For more information on Cloudera support services, please visit either the Support or Sales page. Feel free to contact us directly to discuss your specific needs.

Cloudera reserves the right to change any products at any time, and without notice. Cloudera assumes no responsibility nor liability arising from the use of products, except as expressly agreed to in writing by Cloudera.

Cloudera, Cloudera Altus, HUE, Impala, Cloudera Impala, and other Cloudera marks are registered or unregistered trademarks in the United States and other countries. All other trademarks are the property of their respective owners.

Disclaimer: EXCEPT AS EXPRESSLY PROVIDED IN A WRITTEN AGREEMENT WITH CLOUDERA, CLOUDERA DOES NOT MAKE NOR GIVE ANY REPRESENTATION, WARRANTY, NOR COVENANT OF ANY KIND, WHETHER EXPRESS OR IMPLIED, IN CONNECTION WITH CLOUDERA TECHNOLOGY OR RELATED SUPPORT PROVIDED IN CONNECTION THEREWITH. CLOUDERA DOES NOT WARRANT THAT CLOUDERA PRODUCTS NOR SOFTWARE WILL OPERATE UNINTERRUPTED NOR THAT IT WILL BE FREE FROM DEFECTS NOR ERRORS, THAT IT WILL PROTECT YOUR DATA FROM LOSS, CORRUPTION NOR UNAVAILABILITY, NOR THAT IT WILL MEET ALL OF CUSTOMER’S BUSINESS REQUIREMENTS. WITHOUT LIMITING THE FOREGOING, AND TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW, CLOUDERA EXPRESSLY DISCLAIMS ANY AND ALL IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY, QUALITY, NON-INFRINGEMENT, TITLE, AND FITNESS FOR A PARTICULAR PURPOSE AND ANY REPRESENTATION, WARRANTY, OR COVENANT BASED ON COURSE OF DEALING OR USAGE IN TRADE.

Contents

Databases and data warehouses.....5
Amazon Redshift.....5
Azure SQL DWH/DB & Azure Synapse Analytics.....6
DB2 iSeries.....8
Google BigQuery.....13
Configure the BigQuery region.....19
GreenPlum.....20
IBM DB2 LUW.....21
Microsoft Fabric SQL Database [Technical Preview].....23
Microsoft Fabric Warehouse [Technical Preview].....26
MySQL.....29
Netezza.....30
Oracle (DWH).....32
Oracle Autonomous Data Warehouse (OAD/ADW).....33
PostgreSQL.....35
Snowflake Tool Permissions Prerequisites.....37
SQL Server (SQLS).....46
Teradata.....48
Trino Connector for Cloudera Data Lineage.....49
How to connect to Trino with DBeaver.....54
Vertica.....55

ETL and data integration.....57
Azure Data Factory (ADF).....57
Configure dbt Cloud Connector for Cloudera Data Lineage.....62
IBM DataStage.....65
IICS (Informatica Cloud).....76
Informatica Power Center.....78
Oracle ODI.....81
SSIS (Files / Integration Services / Integration Services Catalog / Azure).....82

BI and reporting.....86
IBM Cognos Report Studio.....86
Cognos Operational Metadata Intelligence Harvester.....93
Looker.....95
MicroStrategy.....96
Oracle OBIEE / Oracle Analytics Server (OAS).....103
Power BI & Power BI Report Server.....106
Power BI (Microsoft Fabric).....119
Qlik Sense.....122
Configure Qlik Sense for Data Extraction.....124
QlikView.....136
SAP Business Objects (BO).....139
SSAS (OLAP + Tabular + Azure Analysis Services).....146
SSRS.....149
Tableau Server.....150
Tableau Intelligent Graph Connector for operational metadata intelligence.....151

Data platforms and lakehouse....154
Configure Databricks in Cloudera Data Lineage....154
Configure Databricks authentication in Cloudera Data Lineage....158
Configure Databricks Metadata Source in Cloudera Data Lineage....162
Databricks Lineage in Cloudera Data Lineage....166
Microsoft Fabric Lakehouse [Technical Preview]....169

Hive, Impala, and HMS....172
Apache Hive connector....172
Configuring Hive connector with HiveServer as the metadata source....173
Apache Impala connector....175
Configuring Impala connector with HiveServer as the metadata source....176
Apache Ranger permissions for Impala metadata extraction....179
Configuring Hive or Impala connector with HMS as the metadata source....182
Prerequisites for HMS connections....183
Configuring an HMS connection in Octopai Client....184
Machine settings for HMS Kerberos authentication....186
Running the Octopai Client Windows service with WSL....188
Configuring MariaDB or MySQL HMS access for Kerberos....188
Configuring PostgreSQL HMS access for Kerberos....190
Configuring Oracle HMS access....191

Streaming and processing....192
Configuring the Kafka Connect connector in Cloudera Data Lineage....192
Cloudera Data Lineage Connector for Apache NiFi....196
Cloudera Data Lineage Connector for Apache Spark — Installation and Setup Guide....202

Code and custom connectors....207
Code-based Lineage Connector....207
Cloudera Data Lineage Custom Lineage Connector (formerly Universal Connector) overview....210
Custom Lineage Connector....211
Custom Lineage Connector for ETLs....213
Custom Lineage Connector for Database Objects....217
Custom Lineage Connector for Reports....218
DBT Files....221
Textual Files or Folder connector....221

Databases and data warehouses

Cloudera Data Lineage extracts metadata from relational databases and data warehouses, including Amazon Redshift, Google BigQuery, Snowflake, Teradata, SQL Server, Oracle, PostgreSQL, and Trino.

Amazon Redshift

Learn how to integrate Amazon Redshift with Cloudera Data Lineage (formerly Octopai).

Note: Version supported: up to 2023

Tool Permissions Prerequisites

Warning:

Missing permissions could end up in broken lineages.

Amazon Redshift ODBC driver installed on the machine running the Octopai Client https://docs.aws.amazon.com/redshift/latest/mgmt/configure-odbc-connection.html#install-odbc-driver-windows

Open Server Port for each Redshift Database Connection

Exiting/New Redshift user (OCTOPAI_USER) for each connection with grant select permission for the following dictionary tables:

- pg_catalog.pg_database
- pg_catalog.pg_proc
- pg_catalog.pg_table_def
- pg_catalog.pg_tables
- pg_catalog.pg_views
- pg_catalog.pg_namespace
- pg_catalog.pg_attribute
- pg_catalog.pg_class
- pg_catalog.pg_constraint
- pg_catalog.svv_table_info
- pg_catalog.svv_all_tables
- pg_catalog.svv_all_columns
- pg_catalog.pg_user
- pg_catalog.svv_external_schemas
- pg_catalog.svv_external_tables
- pg_catalog.svv_external_columns

Grant select on all tables in schema dbo to the group assigned to the Redshift user.

Grant usage on schema dbo to the group assigned to the Redshift user.

Setting up Redshift Metadata Source

Metadata Sources are set on the Octopai Client

Verify extraction results

After a successful run, confirm the metadata archive in the target folder. See Verify extracted metadata files.

Troubleshoot

If an error occurs during the extraction, see Troubleshoot connector extraction failures.

Azure SQL DWH/DB & Azure Synapse Analytics

Learn how to configure and manage Azure SQL DWH/DB.

Note: Version supported: Managed \ Unmanaged

Tool Permissions Prerequisites

Warning:

Missing permissions could end up in broken lineages.

- Open Server Port for each SQL Server Database Connection
- Existing/New user (SQL Server Authentication only) for each server/database with:
  - Grant 'CONNECT' to MASTER DB
  - Grant 'SELECT' for the following dictionary tables:
    - sys.objects
    - sys.schemas
    - sys.sql_modules
    - sys.columns
    - sys.identity_columns
    - sys.computed_columns
    - sys.check_constraints
    - sys.synonyms
    - sys.indexes
    - sys.index_columns
    - sys.tables
    - sys.foreign_keys
    - sys.foreign_key_columns
- Only if previous grants are not enough:
  - USE MASTER:
  - grant connect any database to "DOMAIN_NAME\OCTOPAI_USER";
  - grant view server state to "DOMAIN_NAME\OCTOPAI_USER";
  - grant view any definition to "DOMAIN_NAME\OCTOPAI_USER";

Setting up SQL Azure DWH/DB Metadata Source

Metadata Sources are set on the Octopai Client:

Figure 1: Azure SQL DWH/DB

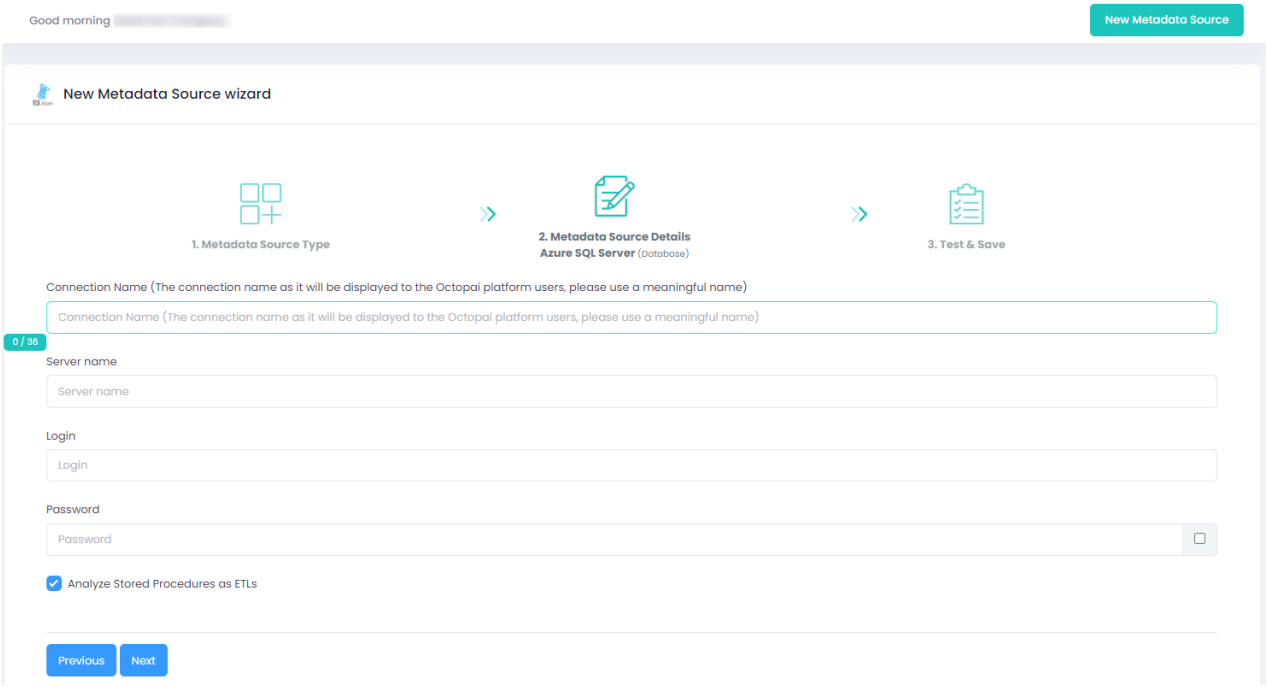
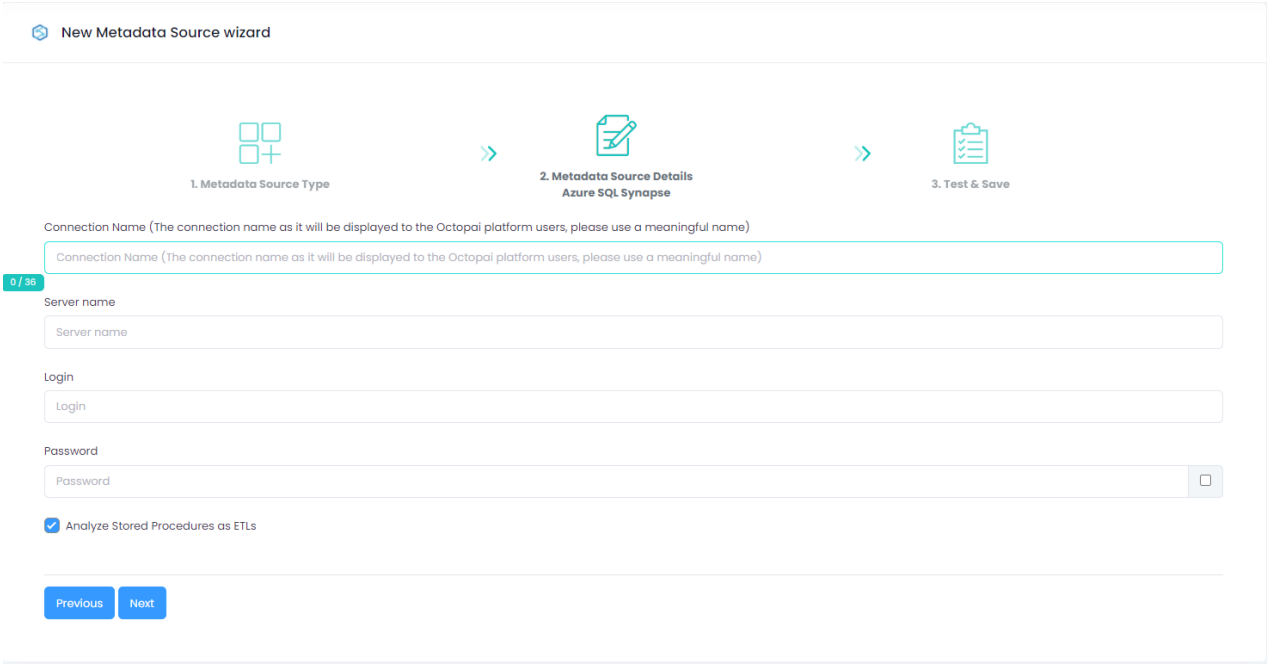


Figure 2: Azure SQL Synapse



Verify extraction results

After a successful run, confirm the metadata archive in the target folder. See Verify extracted metadata files.

Troubleshoot

If an error occurs during the extraction, see Troubleshoot connector extraction failures.

DB2 iSeries

Configure Cloudera Data Lineage (formerly Octopai) to extract metadata from IBM DB2 iSeries, from client setup through ODBC configuration and required permissions.

Tool Permissions Prerequisites

- Warning: Missing permissions could end up in broken lineages.
- Microsoft Visual C++ Redistributable 2013 x64: Essential for running applications developed with Visual C++ on Windows systems. Download from this link.
- IBM i Access Client Solutions: A suite for accessing and managing IBM iSeries systems.
- Network Configuration: Ensure the server port is open for each DB2 iSeries database connection to facilitate communication.

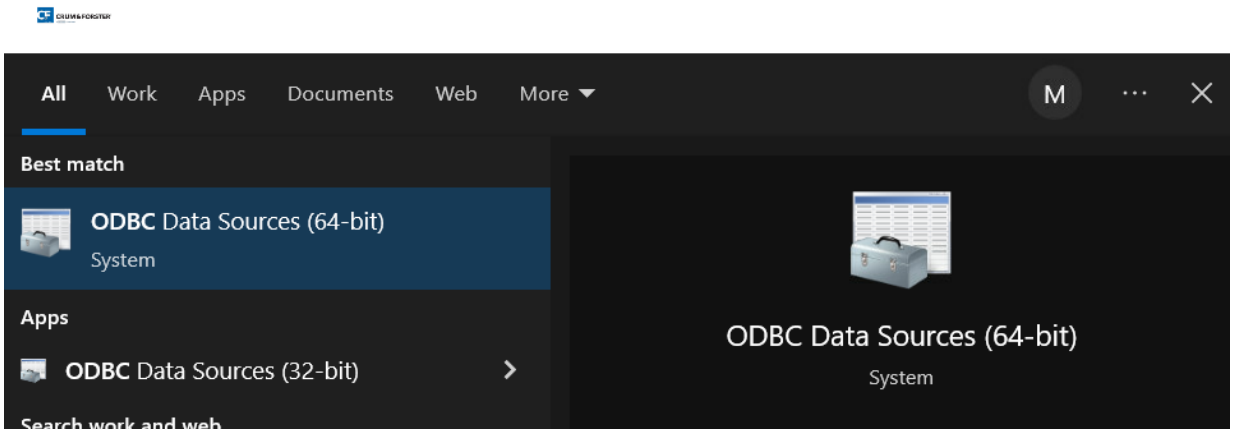
Configuring ODBC Data Sources for DB2 iSeries

IBM i Access Client Solutions Installation:

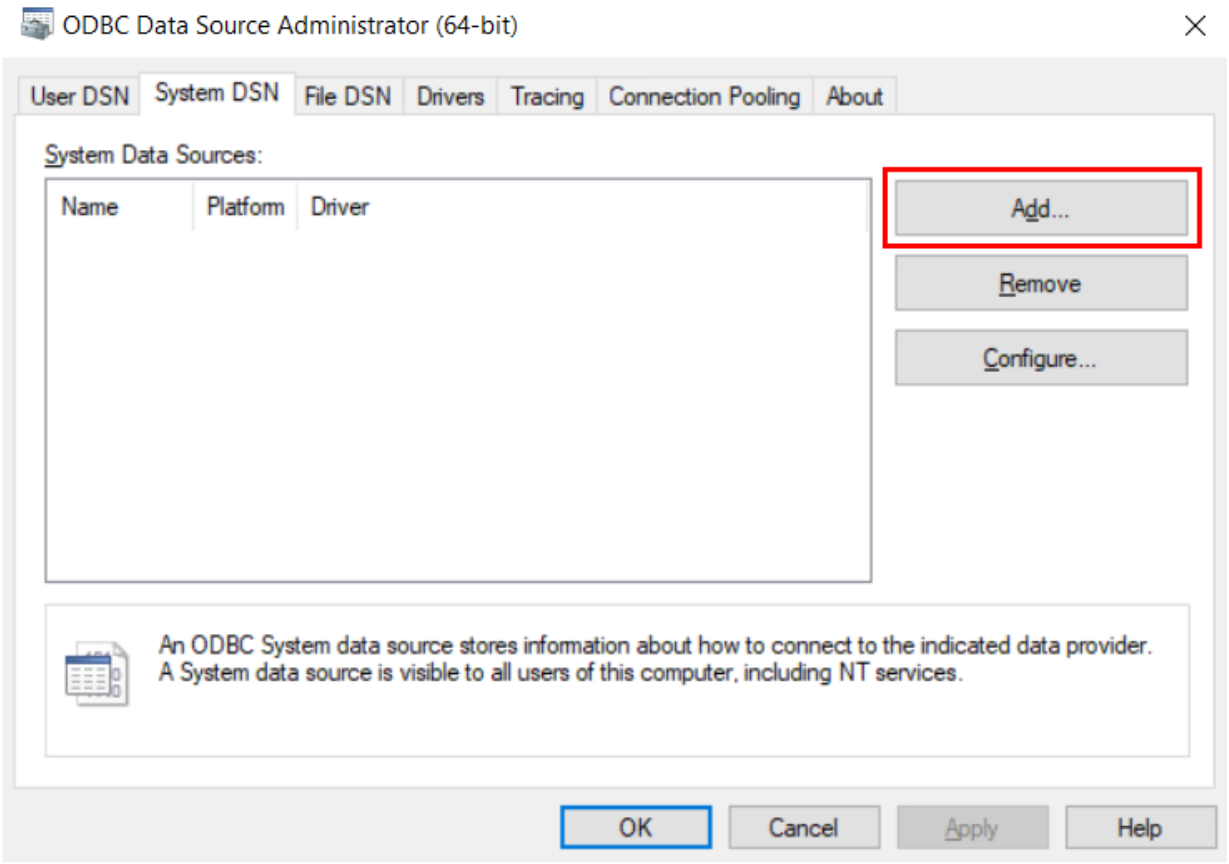
1. Navigate to the IBM website's login page, and log in using your IBM ID and password.
2. Download the "ACS Windows App Pkg English (64bit)".
3. Execute the installer located at IBMiAccess_v1r1_WindowsAP_English\Image64a\setup.exe to extract and install the ACS application.

Post-installation Configuration:

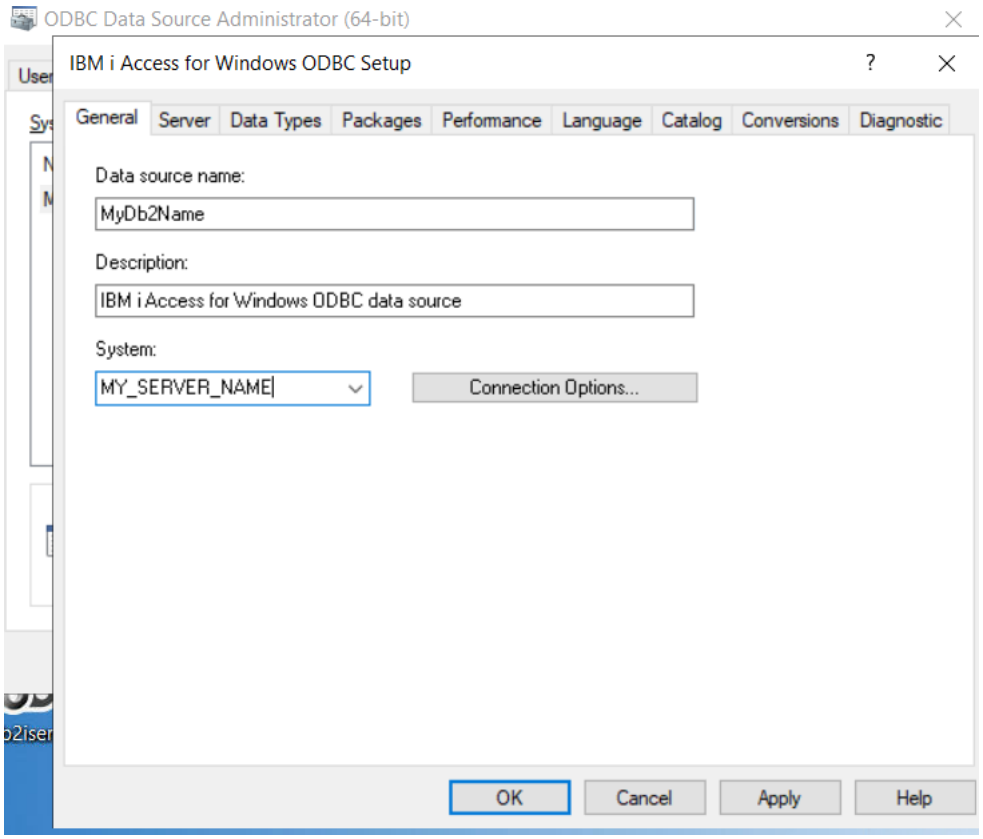
1. Access the ODBC Data Source Administrator by navigating through Control Panel > Administrative Tools > ODBC Data Sources (64-bit).



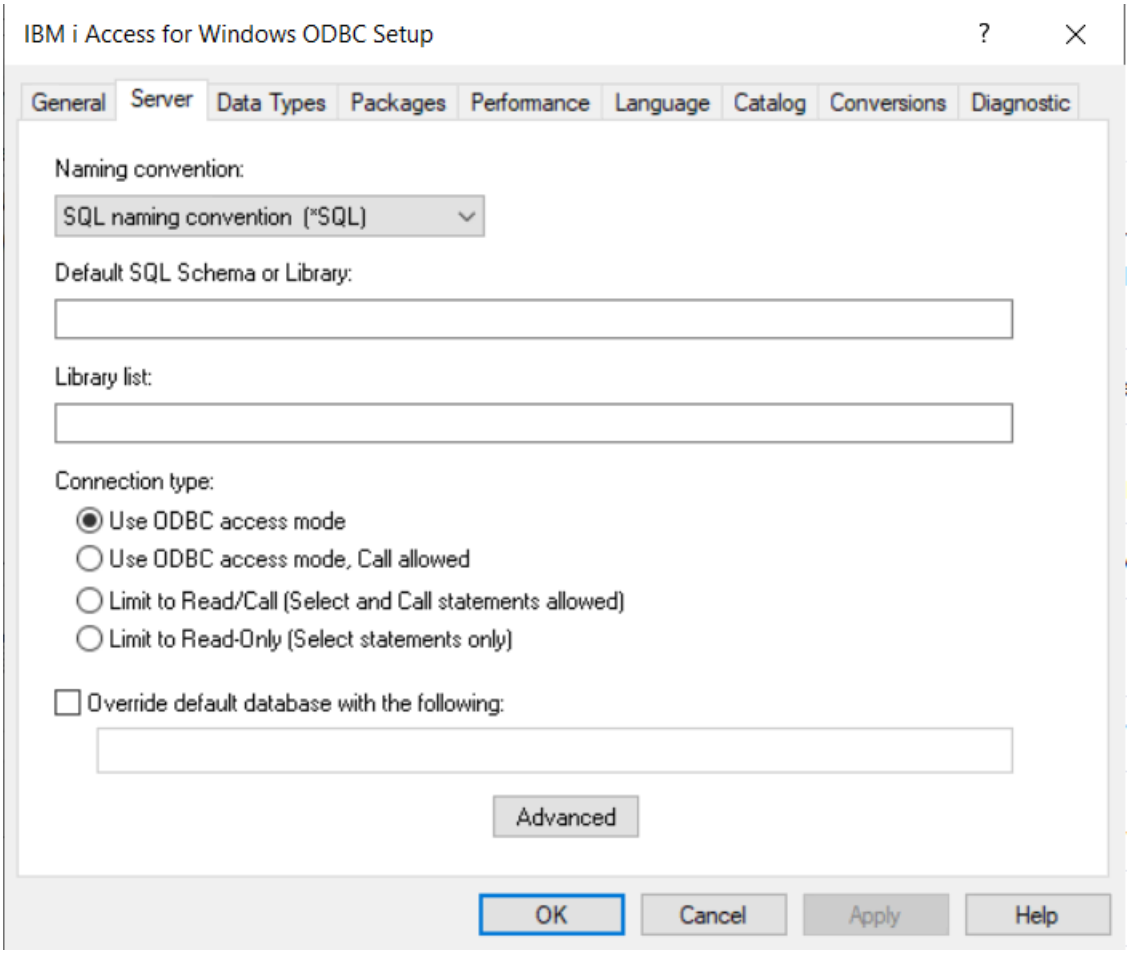
2. In the System DSN tab, click on Add to initiate the creation of a new data source.



3. Select iSeries Access ODBC Driver and click the Finish button. The General tab of the IBM i Access for Windows ODBC Setup dialog is displayed.
4. Recommended Settings for the General tab:
   - Data source name – Enter any unique name to identify the ODBC data source.
   - System – Enter the hostname where LMi resides.



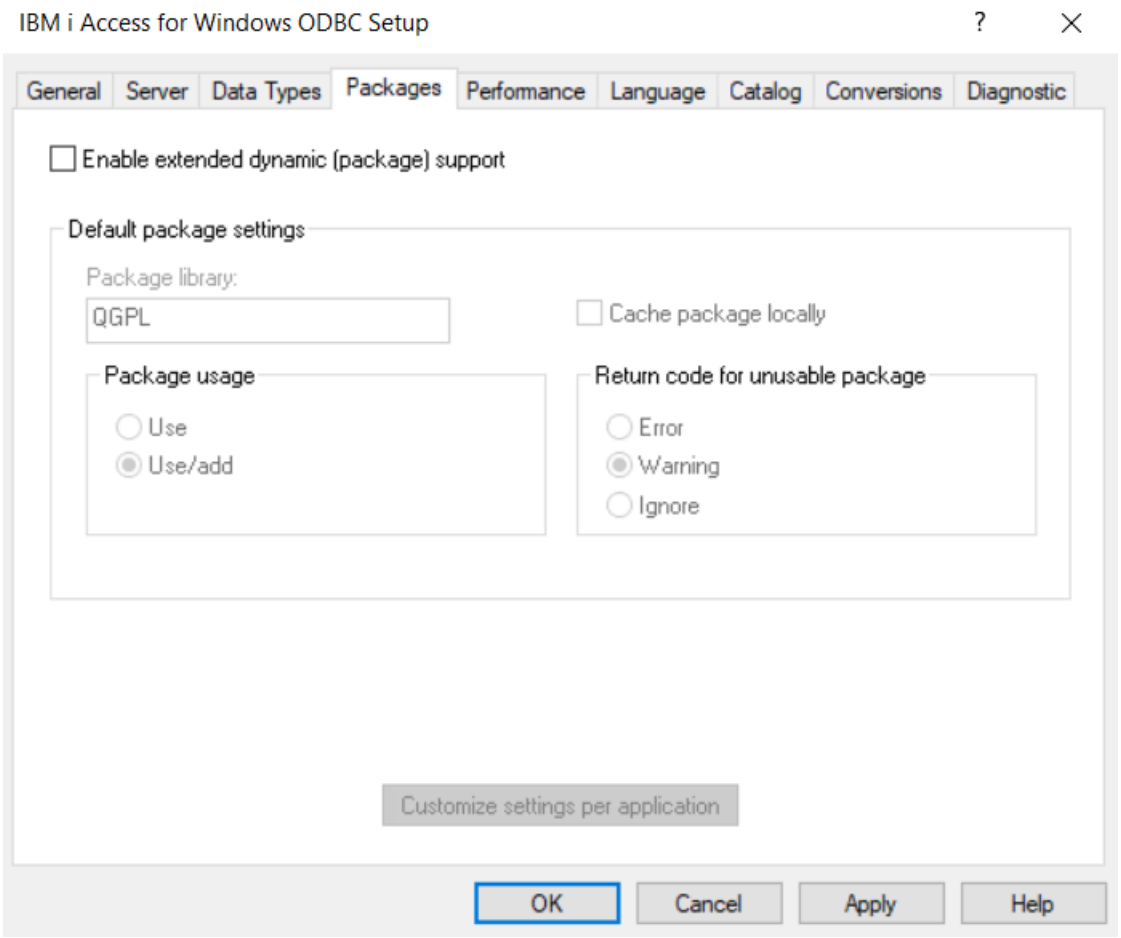
5. Click the Connection Options button to display the Connection Options dialog. Recommended settings:
   - Default user ID – Use iSeries Navigator default
   - Sign-on dialog prompting – Prompt for SQL Connect if needed
   - Security – Use the same security as the iSeries Navigator connection
6. Click OK.
7. On the IBM i Access for Windows ODBC Setup dialog box, select the Server tab. Recommended settings:
   a. In the Naming convention combo box, select SQL naming convention (*SQL).
   b. Leave Default SQL Schema or Library blank.
   c. Leave the Library list blank.
   d. For the Connection type, select Use ODBC access mode.
   e. Leave Override default database with the following unchecked.



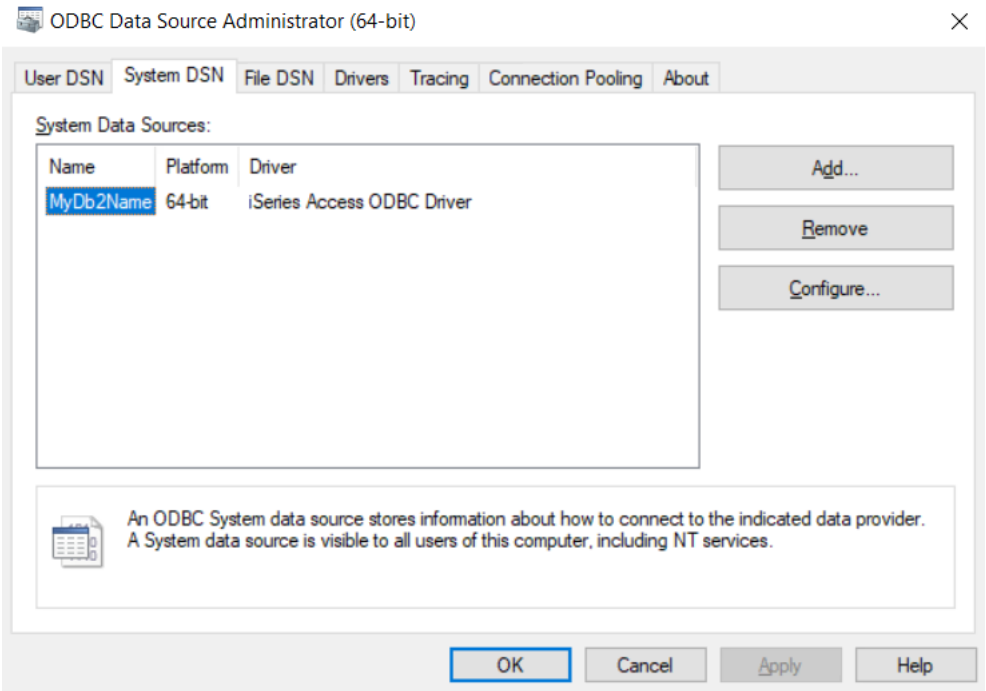
f. Click OK.

8. On the IBM i Access for Windows ODBC Setup dialog box, select the Packages tab. Packages Tab Adjustment:

In the Packages tab, it is crucial to ensure the 'Enable Extended Dynamic Support' checkbox is unchecked, deactivating other controls on the page.

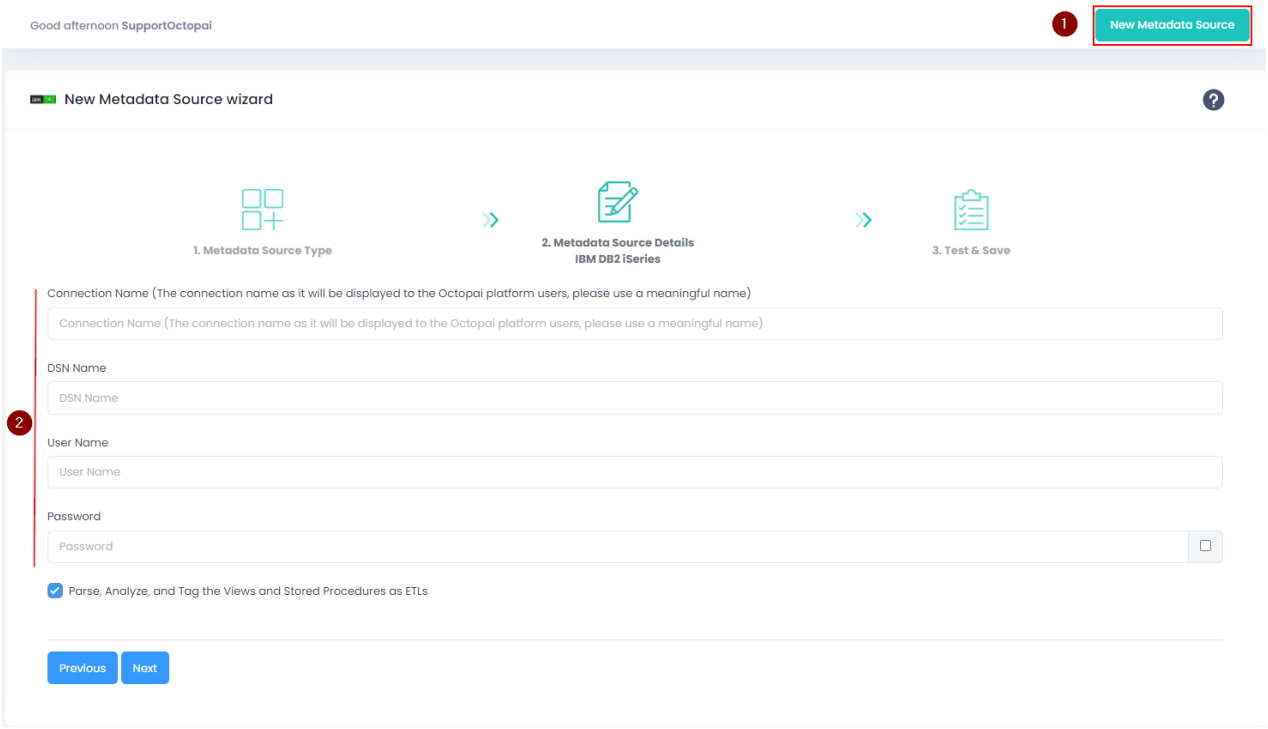


Finally, this is what we will see:



Setting up IBM Iseries Metadata Source in Octopai Client

First, press "New Metadata Source," then fill in the relevant information for your new connection.



Google BigQuery

Learn how to configure Google BigQuery as a metadata source for Cloudera Data Lineage (formerly Octopai).

Note: Version Supported: up to 2023

Tool Permissions Prerequisites

Warning:

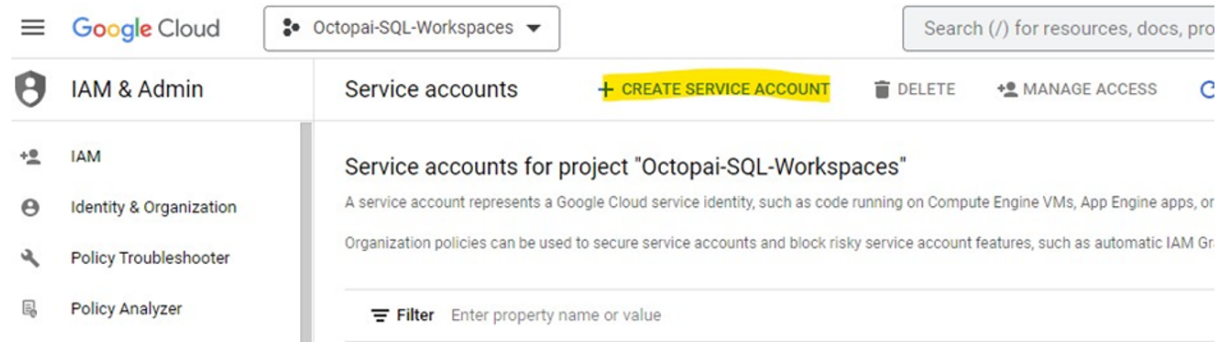
Missing permissions could end up in broken lineages.

Google Service Account (SA) integrated with Cloudera Data Lineage.

How to establish a Google Cloud Service Account

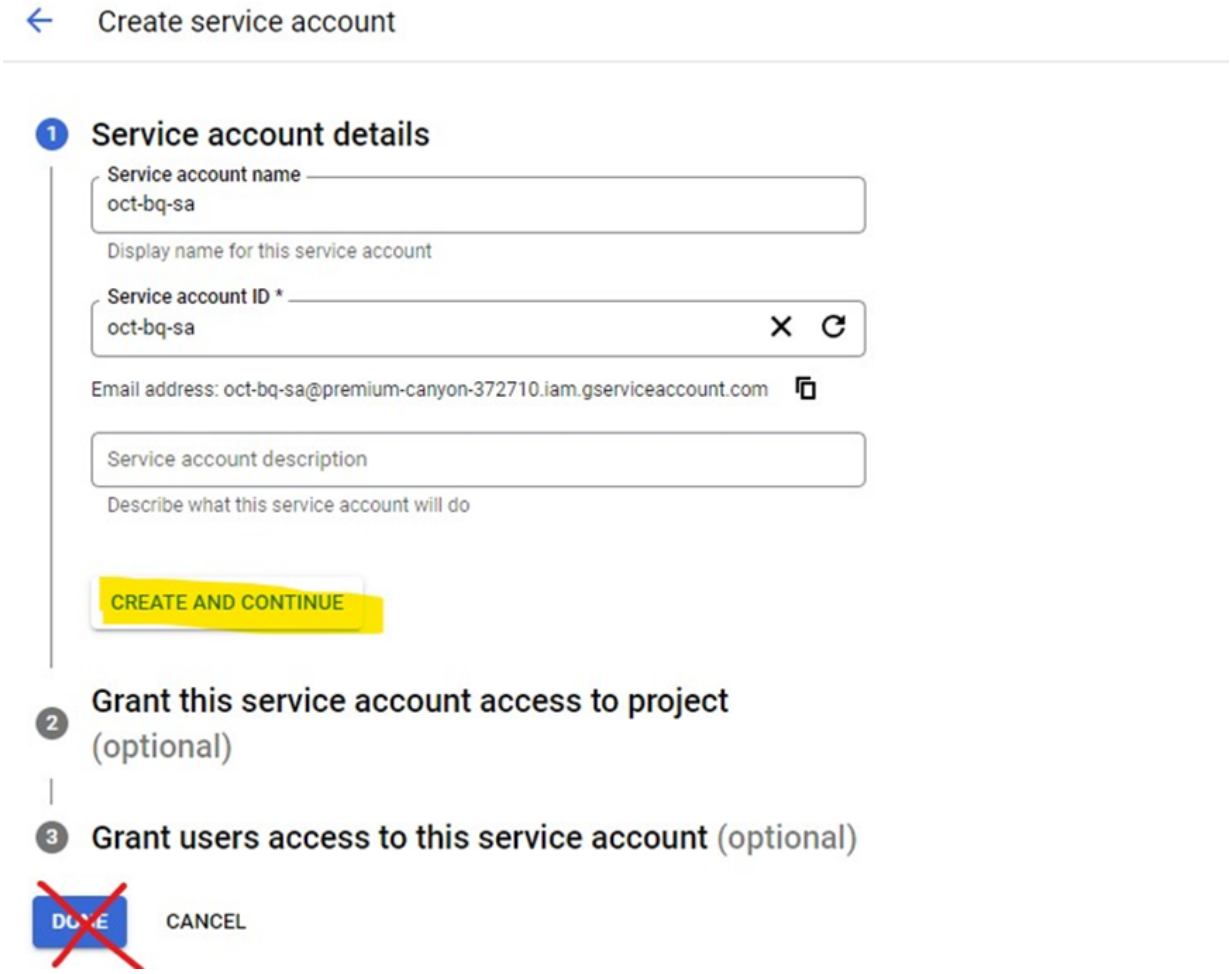
Creating a Google Cloud Service Account (SA)

1. Open the Google Cloud Console and navigate to your desired project.
2. Access the IAM & Admin menu and select 'Service Accounts'.
3. Click on 'Create Service Account' to initiate the creation process.

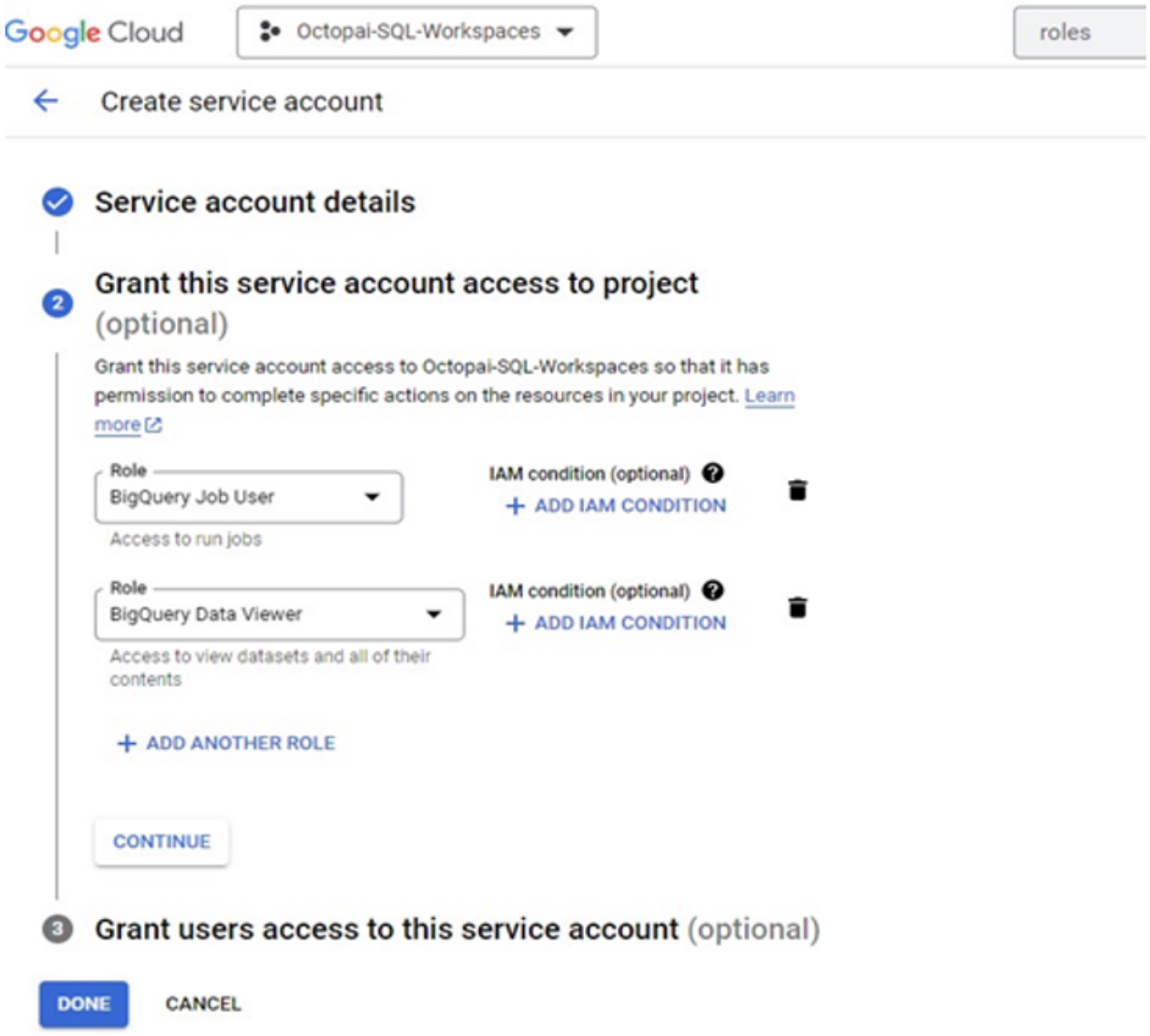


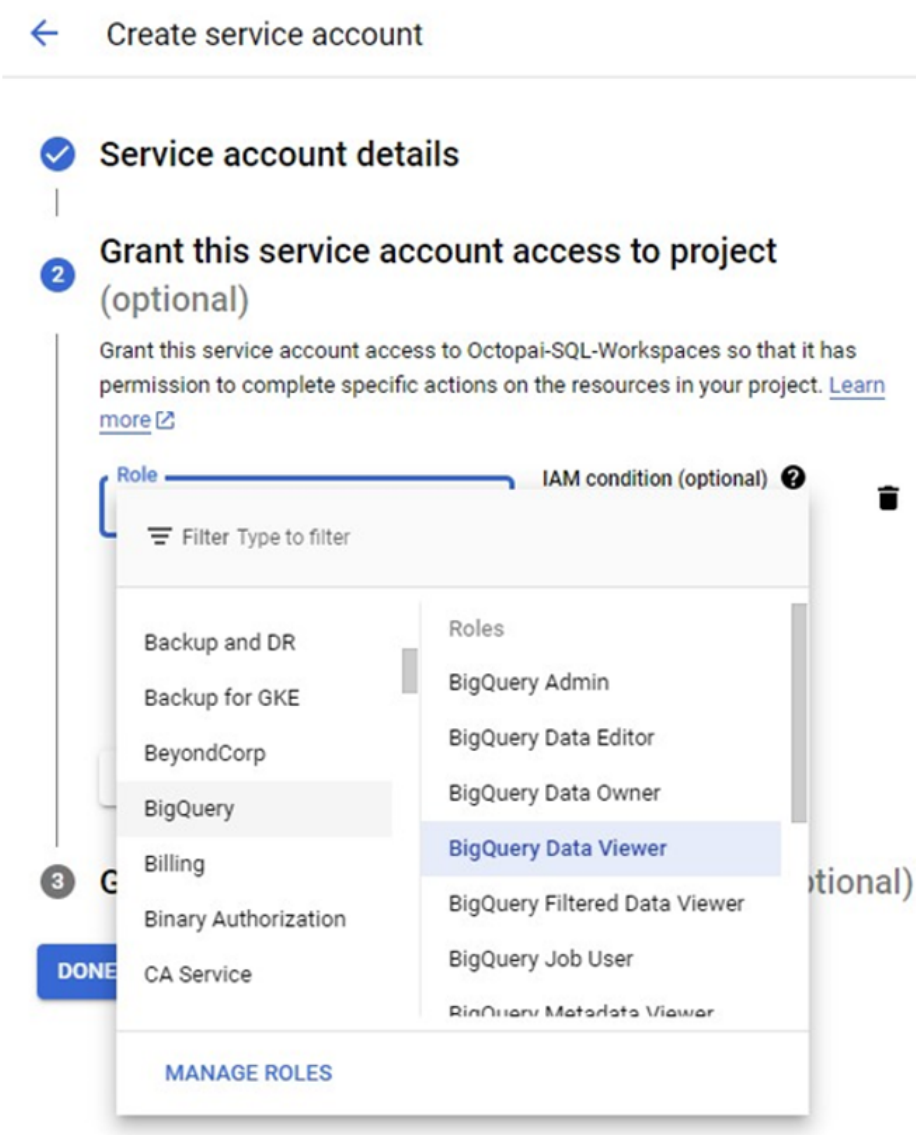
4. Assign a unique, identifiable name to your Service Account. Upon clicking 'Create and Continue', an Identity and Access Management (IAM) principal will be instantiated automatically.

Important: Important: Do not click 'Done' at this point.

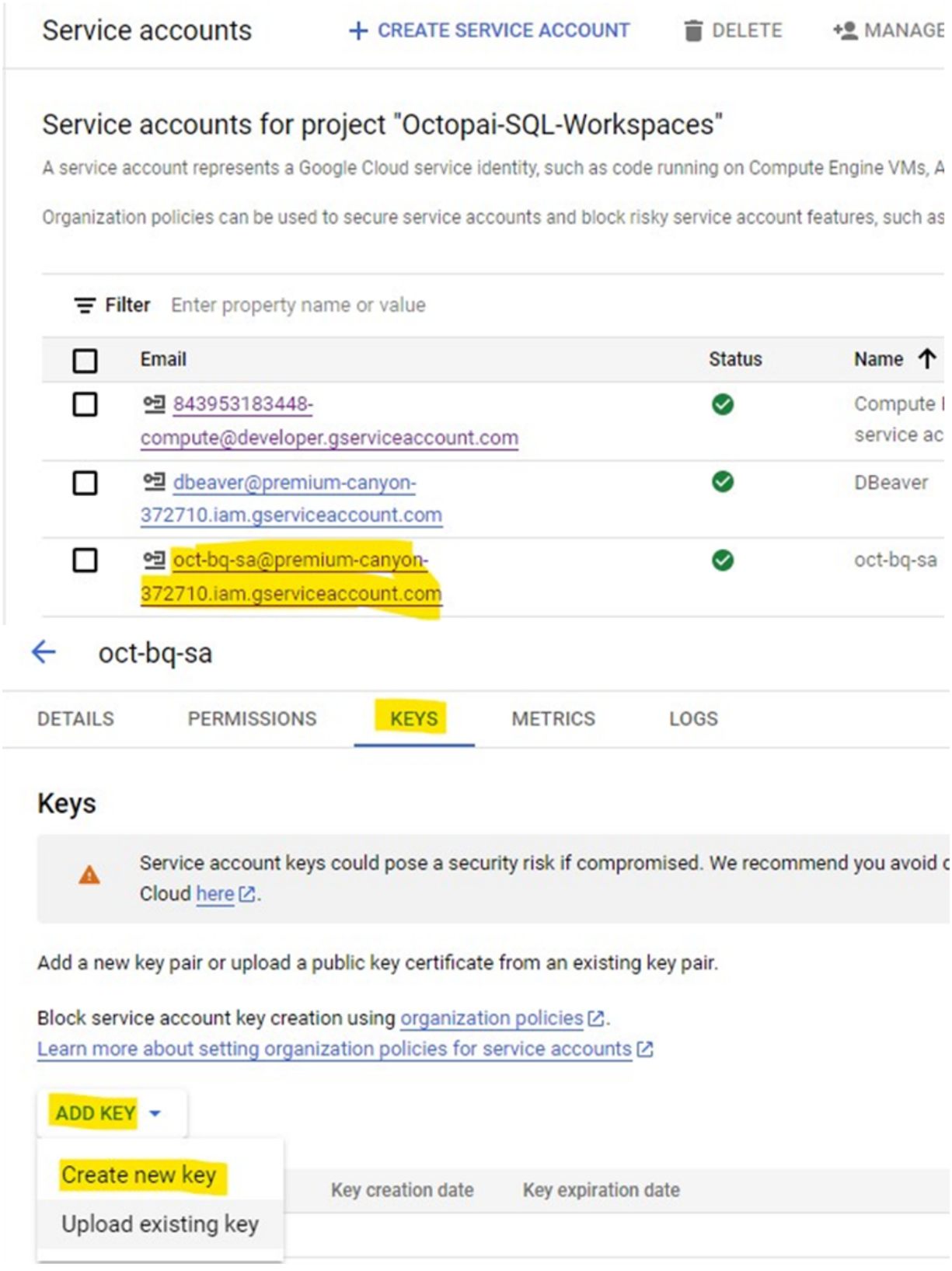


5. From the 'roles' dropdown menu, select both 'BigQuery Data Viewer' and 'BigQuery Job User' to assign necessary permissions to your Service Account.


Google Cloud
Octopai-SQL-Workspaces
roles
Create service account
Service account details
2
Grant this service account access to project (optional)
Grant this service account access to Octopai-SQL-Workspaces so that it has permission to complete specific actions on the resources in your project. Learn more
Role
BigQuery Job User
Access to run jobs
IAM condition (optional)
ADD IAM CONDITION
Role
BigQuery Data Viewer
Access to view datasets and all of their contents
IAM condition (optional)
ADD IAM CONDITION
ADD ANOTHER ROLE
CONTINUE
3
Grant users access to this service account (optional)
DONE
CANCEL

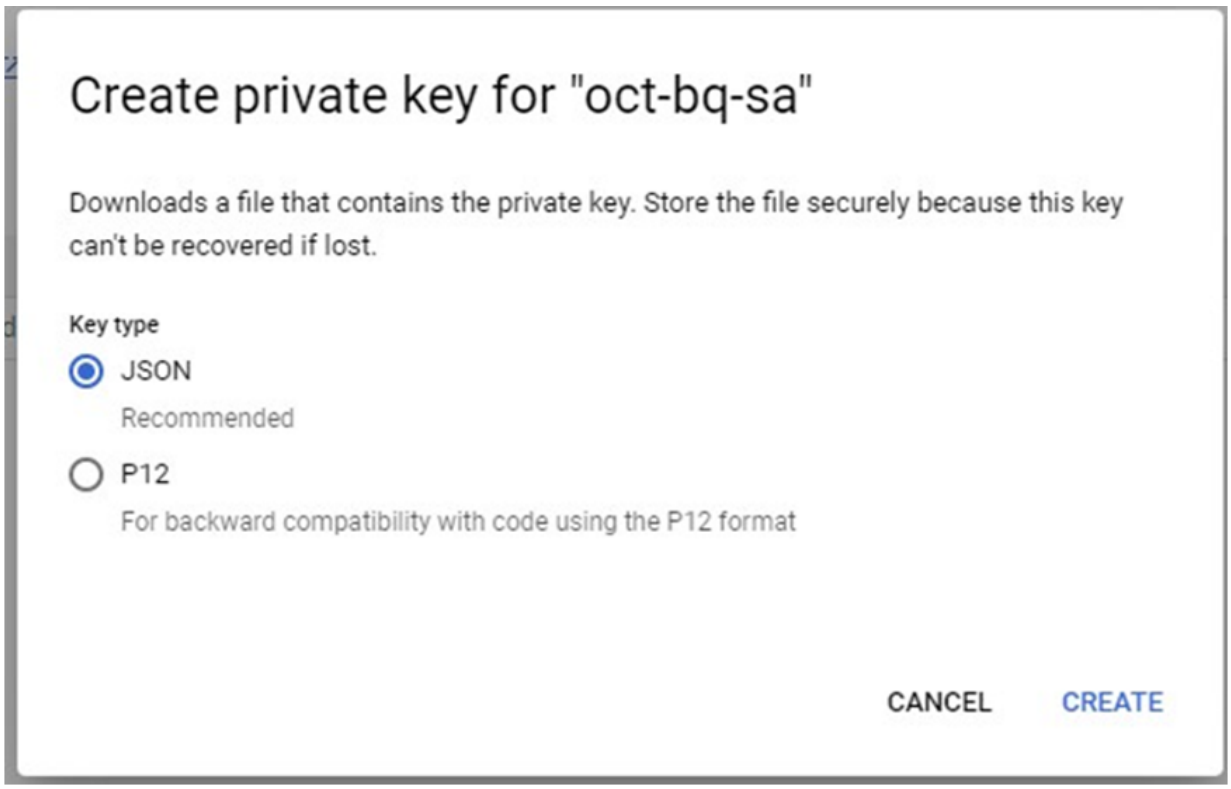


6. Complete the Service Account creation process by clicking 'Done'.
7. Access the newly created Service Account and navigate to the 'KEYS' tab.



Generating a Service Account Key

- Click on 'ADD KEY', then 'Create new key'. Make sure to select the JSON format. The key will automatically be downloaded to your local system.

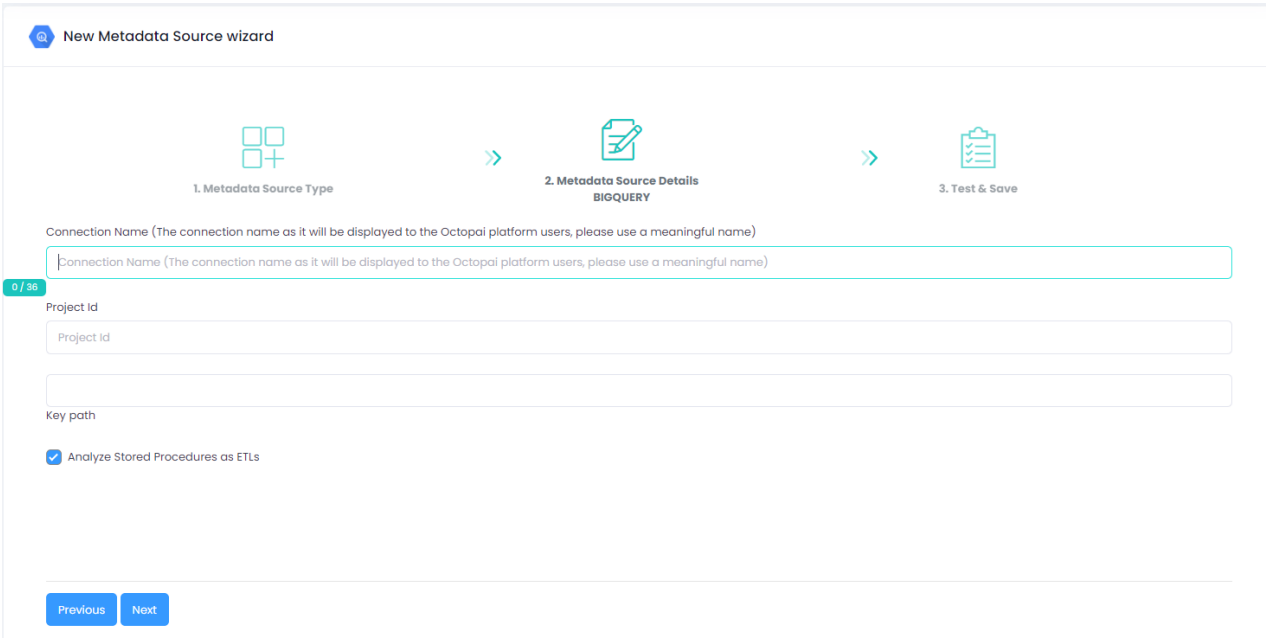


Configuring Cloudera Data Lineage with BigQuery:

1. Open Cloudera Data Lineage and start the creation of a new metadata source, choosing 'BigQuery' as your source.
2. Assign a descriptive name to this metadata source for easy reference in the future.
3. Input the Project ID associated with your Google Cloud project. This can be found within the downloaded JSON file (under the 'project_id' field) or in the project selector on the Google Cloud Console.
4. Specify the file path where your downloaded JSON key file is stored in the 'Key Path' field.
5. Save your settings and initiate the connection by clicking 'Save and Run'.

Setting up Google BigQuery Metadata Source

Metadata Sources are set on the Octopai Client



If your datasets are not in the US multi-region, see Configure the BigQuery region.

Verify extraction results

After a successful run, confirm the metadata archive in the target folder. See Verify extracted metadata files.

Troubleshoot

If an error occurs during the extraction, see Troubleshoot connector extraction failures.

Configure the BigQuery region

Learn how to configure the BigQuery region on the Octopai Client so dataset discovery matches your Google Cloud dataset location.

About this task

BigQuery dataset discovery is region-dependent. You must configure a region that matches the location of your datasets in Google Cloud.

Note: If you do not specify the region property, the default value is US.

Before you begin

The connector supports the following region values:

- US (default)
- EU

Procedure

1. Open the following file on the Octopai Client server:
   C:\Program Files (x86)\Octopai\Service\Settings\bigQuerySettings.json
2. In Google Cloud Console, go to BigQuery Dataset Dataset properties and verify your dataset location.

Important:

The configured region must match the dataset location.

If the configured region does not match the dataset location, the connection test or metadata extraction may fail or return no datasets.

3. Locate the entry whose connectionId matches your BigQuery connection in bigQuerySettings.json.
4. Set the region property to US or EU based on the dataset location.

   Use EU for the EU multi-region and European regional locations. Use US for the US multi-region and US regional locations.

   The following example shows a bigQuerySettings.json entry with the region set to EU:

```
{
  "proxy": null,
  "extractIndirect": false,
  "connectionId": 102,
  "indirectProjectId": "",
  "region": "EU"
}
```

5. Restart the Cloudera Data Lineage (formerly Octopai) Service before you run the extraction again.

GreenPlum

Learn how to configure GreenPlum as a metadata source for Cloudera Data Lineage (formerly Octopai).

Tool Permissions Prerequisites

Warning: Missing permissions could end up in broken lineages.

Add the following line to the **pg_hba.conf** file located under **/data/master/gpseg-1**.

Enable remote access, allowing the Octopai Client to perform the MetaData extraction:

TYPE	DATABASE	USER	ADDRESS	METHOD
host	databasename	octopai-user	octopai-client-ip-address	md5

Replace **"databasename"**, **"octopai-user"**, and **"octopai-client-ip-address"** with the actual values for your database and client. This line will allow the Octopai Client to access the specified database using **md5** authentication.

Ensure the following prerequisites are met:

- Postgres ODBC driver installed on the machine running the Octopai Client.
- Open Server Port for each Postgre Database Connection.
- Existing/New Postgres user (**OCTOPAI_USER**) for each connection with grant select permission for the following dictionary tables:
  - information_schema.tables
  - information_schema.views
  - information_schema.columns
  - pg_catalog.pg_proc
  - pg_catalog.pg_namespace
  - pg_catalog.pg_attribute
  - pg_catalog.pg_constraint
  - pg_catalog.pg_class

- pg_catalog.pg_database

How to set up the permissions

psqlodbc - PostgreSQL ODBC driver

Setting up GreenPlum Metadata Source

Metadata Sources are set on the Octopai Client

Verify extraction results

After a successful run, confirm the metadata archive in the target folder. See Verify extracted metadata files.

Troubleshoot

If an error occurs during the extraction, see Troubleshoot connector extraction failures.

IBM DB2 LUW

Learn how to integrate IBM DB2 LUW with Cloudera Data Lineage (formerly Octopai).

Tool Permissions Prerequisites



Warning: Missing permissions could end up in broken lineages.

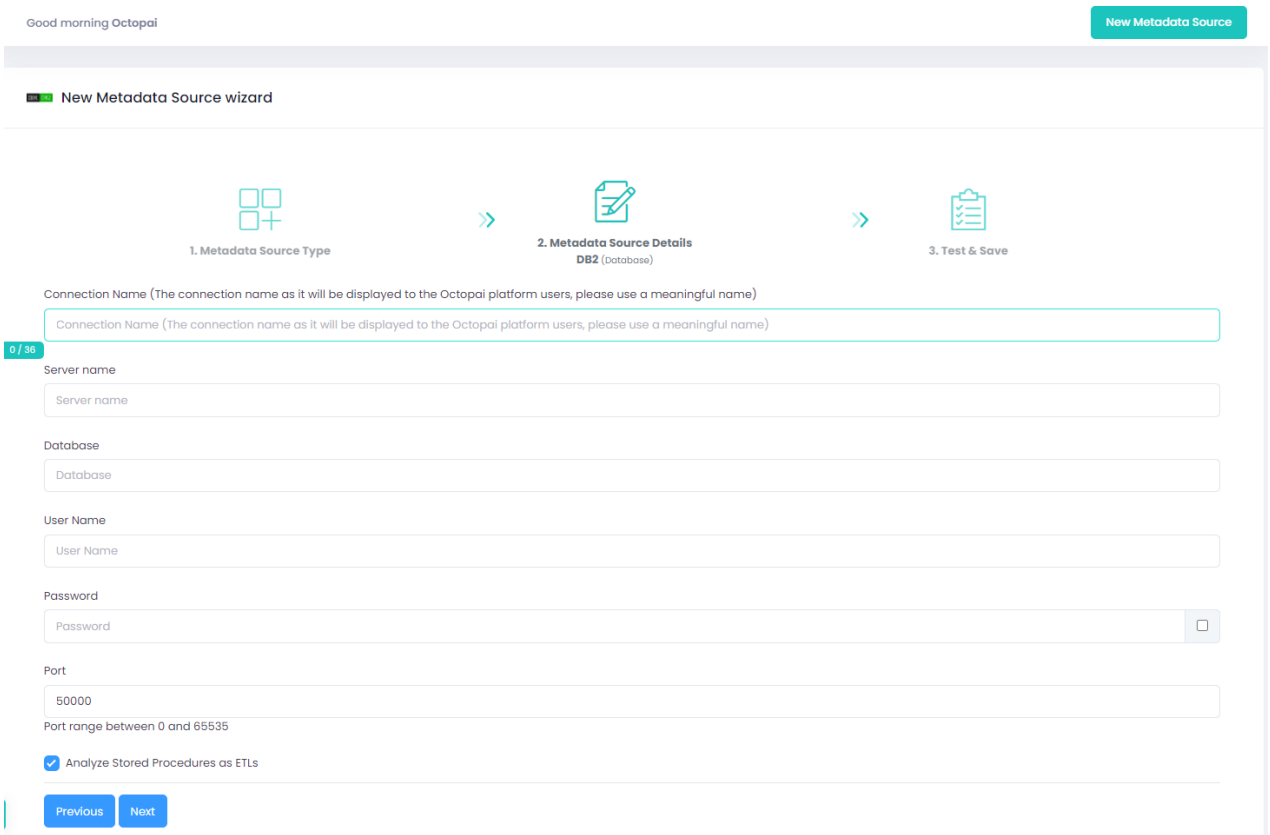
- DB2 ODBC driver installed on the machine running the Octopai Client
- Microsoft Visual C++ Redistributable - Visual Studio 2013 (VC++ 12.0) - 64X, installed on the machine running the Octopai Client
- Open Server Port for each DB2 Database Connection
- Existing/New DB2 user(OCTOPAI_USER) for each connection with:
  - Grant connect on DATABASE ;
  - Grant Execute on function SYSPROC.ENV_GET_INST_INFO;
  - Grant Execute on function SYSPROC.MON_GET_CONTAINER;
  - Grant select permission for the following dictionary tables:
    - SYSIBM.SYSTABAUTH;
    - SYSIBM.SYSCOLAUTH;
    - SYSIBM.SYSDBAUTH;
    - SYSIBMADM.DBMCFG;
    - SYSIBMADM.DBCFG;
    - SSYSIBM.SYSDBAUTH;
    - SYSCAT.PACKAGEAUTH;
    - SYSCAT.TBSPACEAUTH;
    - SSYSCAT.DBAUTH;
    - SYSCAT.SEQUENCEAUTH;
    - SYSCAT.INDEXAUTH;
    - SYSCAT.TABLES;
    - SYSIBM.SYSSCHEMAAUTH ;
    - SSYSIBM.SYSTABAUTH;
    - SYSCAT.LIBRARYAUTH;
    - SYSCAT.TABAUTH;
    - SYSIBM.SYSROUTINEAUTH;

- SYSIBM.ROUTINES;
- SYSCAT.INDEXES;
- SSYSCAT.SCHEMAAUTH;
- SYSCAT.PACKAGES;
- SYSCAT.VIEWS;
- SYSCAT.TRIGGERS;
- SYSCAT.PASSTHRUAUTH;
- SYSCAT.ROUTINEAUTH;
- SYSCAT.TABLESPACES;
- SYSCAT.SEQUENCES;
- SYSCAT.ROUTINES;
- SYSCAT.INDEXES;
- SYSCAT.PACKAGES;
- SYSCAT.SCHEMATA;
- SYSCAT.TRIGGERS;
- SYSCAT.TABLES;
- SYSCAT.VIEWS;
- SYSCAT.ROUTINES;

Setting up IBM - DB2 Metadata Source

Metadata Sources are set on the Octopai Client

Figure 3: New Metadata Source Wizard



Verify extraction results

After a successful run, confirm the metadata archive in the target folder. See Verify extracted metadata files.

Troubleshoot

If an error occurs during the extraction, see Troubleshoot connector extraction failures.

Microsoft Fabric SQL Database [Technical Preview]

The Cloudera Data Lineage (formerly Octopai) Microsoft Fabric SQL Database connector requires a Microsoft Entra service principal, Fabric admin tenant settings, Fabric API permissions, ODBC connectivity, Contributor workspace access, and the SQL Database (Microsoft Fabric) metadata source wizard.

Note: This feature is in Technical Preview and is not ready for production deployments. Cloudera recommends trying this feature in test or development environments and encourages you to provide feedback on your experiences.

Overview

Before you set up the Microsoft Fabric SQL Database connector, ensure that you have the following:

- **Microsoft Entra application credentials:** a registered service principal with a tenant ID, application (client) ID, and client secret. See Application registration (service principal).
- **Microsoft ODBC Driver 18 for SQL Server:** installed on the Windows machine that hosts the Cloudera Data Lineage platform for SQL endpoint connectivity. See SQL access (ODBC).
- **Fabric admin tenant settings:** settings that allow the service principal to call Fabric public APIs and access Fabric items. See Tenant settings (Fabric admin portal).
- **Fabric workspace permissions:** the Contributor role assigned to the service principal on the target workspace. See Workspace permissions.
- **Microsoft Fabric API permissions:** application API permissions assigned to the service principal, including SQLDatabase.Read.All and related scopes. See Required application permissions.

Application registration (service principal)

1. In the Azure portal, open Microsoft Entra ID App registrations , and select New registration.
2. Give the application a name, and select Accounts in this organizational directory only.

   Note:

   Redirect URI is optional and is not needed for client credentials.
3. Select Register.
4. From the application overview, record the Directory (tenant) ID and Application (client) ID.
5. Create a Client secret under Manage Certificates and secrets .

The Cloudera Data Lineage Microsoft Fabric SQL Database connector authenticates using a Microsoft Entra service principal. The service principal is used for:

- Microsoft Fabric REST API access
- SQL endpoint access through ODBC using Microsoft Entra authentication

You must record the following values for configuration in Cloudera Data Lineage:

- Tenant ID
- Application (client) ID
- Client Secret Value

SQL access (ODBC)

Microsoft Fabric SQL Database metadata extraction requires SQL connectivity through the SQL endpoint. Install Microsoft ODBC Driver 18 for SQL Server on the Windows machine that hosts the Cloudera Data Lineage platform.

For Windows, see Download ODBC Driver for SQL Server.

Tenant settings (Fabric admin portal)

Use the Fabric admin portal to enable the settings required by the connector:

1. Open Microsoft Fabric.
2. Select Settings, and then select Admin portal.
3. Open Tenant settings.
4. Find and enable Service principals can call Fabric public APIs.
5. Find and enable Allow apps to access Fabric items.

Workspace permissions

Assign the service principal to the target workspace with the Contributor role. Contributor access is required for complete metadata extraction coverage, including:

- Stored procedures
- Views
- Functions
- Tables
- Columns
- Schemas

The Viewer role is not sufficient for full metadata extraction because it may not expose stored procedures, views, and functions.



Important:

Cloudera recommends Contributor access for all Cloudera Data Lineage Microsoft Fabric SQL Database integrations. Lower-privilege roles result in incomplete metadata extraction.



Note:

Cloudera Data Lineage uses Contributor access in a read-only manner. The connector only reads and extracts metadata. It does not create, update, delete, or modify objects in the workspace.

Required application permissions

The application requires the following Microsoft Fabric API permissions:

- SQLDatabase.Read.All
- SQLEndpoint.Read.All
- Workspace.Read.All
- Item.Read.All

Metadata access requirements

Ensure that the service principal can access the following SQL metadata sources:

- INFORMATION_SCHEMA: returns metadata only for objects visible to the current user. Limited permissions can result in missing objects.
- **System views** (sys.*): requires SELECT permission or equivalent metadata visibility. Used for extracting views, stored procedures, and functions, including:
  - sys.objects

- sys.sql_modules
- sys.schemas

Network requirements

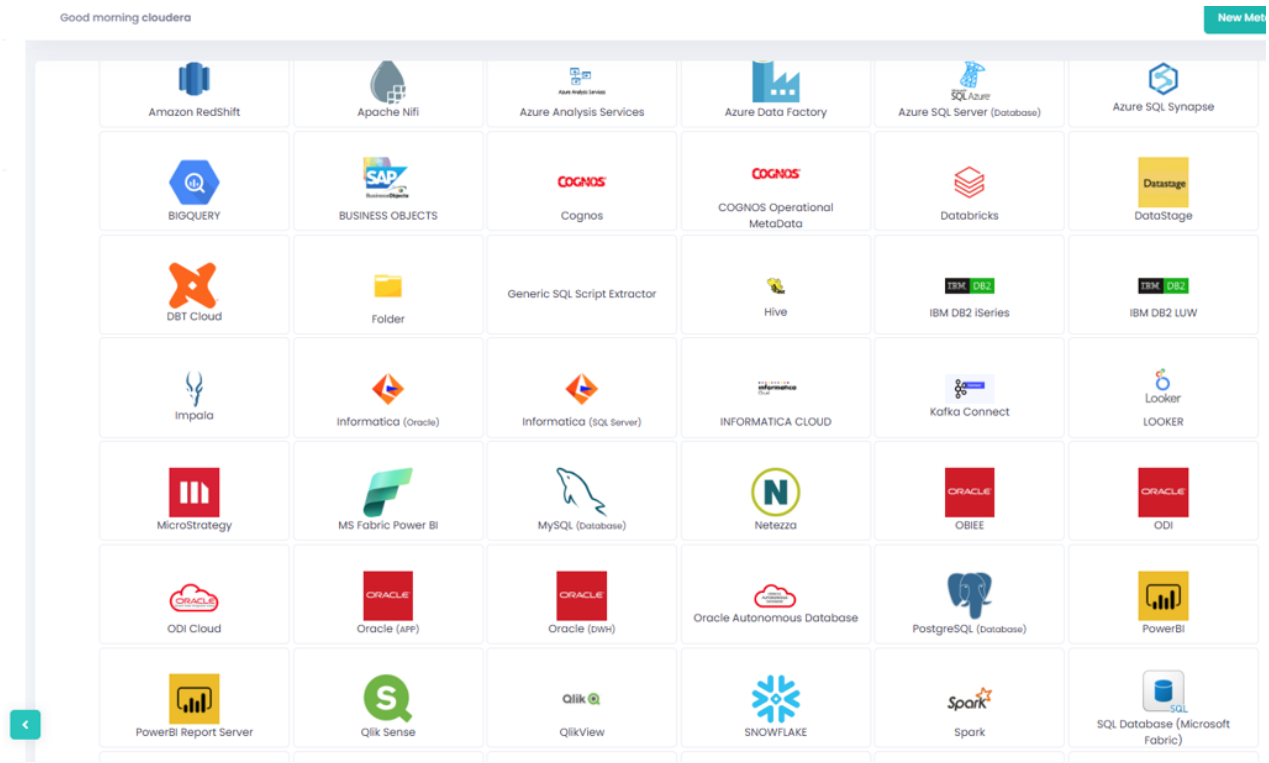
The extraction environment must allow outbound HTTPS access over port 443 to the following endpoints for authentication and Microsoft Fabric API interaction:

- api.fabric.microsoft.com
- *.fabric.microsoft.com
- login.microsoftonline.com

Setting up SQL Database (Microsoft Fabric) metadata source

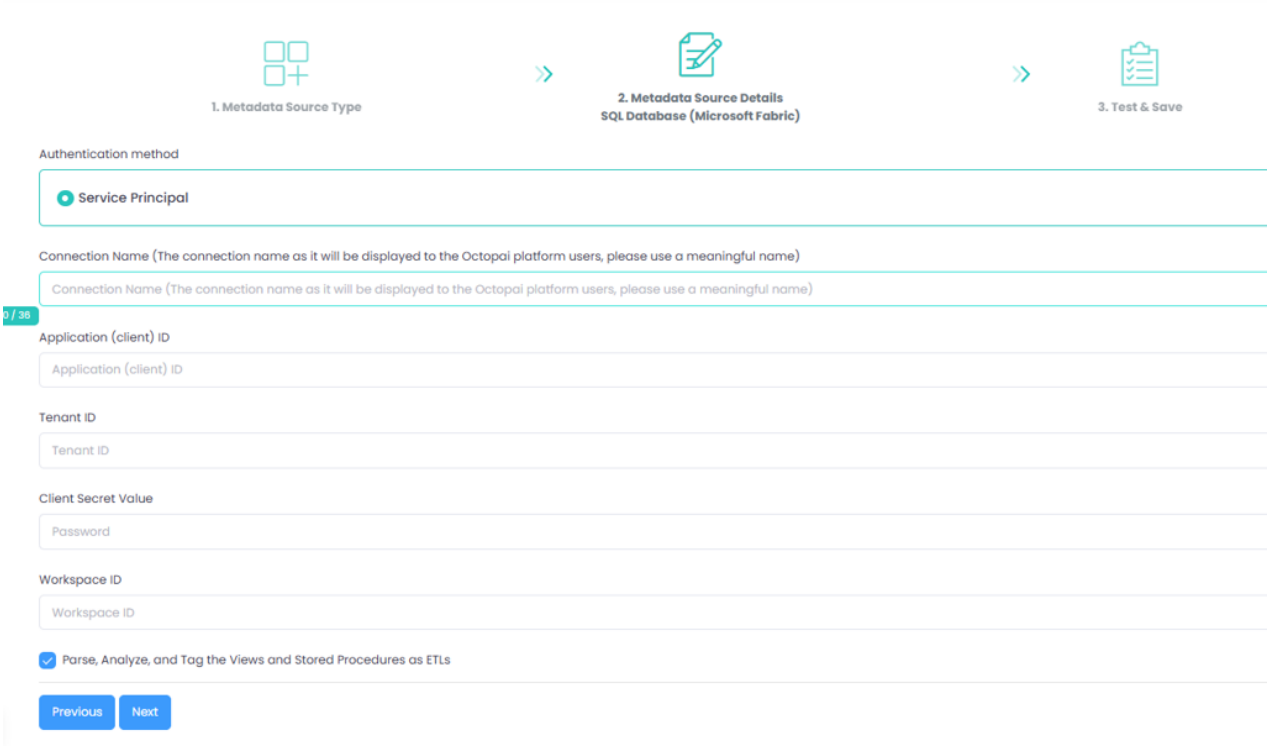
Configure the metadata source on the Octopai Client. In the New Metadata Source wizard, on Metadata Source Type, select SQL Database (Microsoft Fabric).

Figure 4: Selecting SQL Database (Microsoft Fabric) as the metadata source type



On Metadata Source Details, select Service Principal, and enter the following:

Figure 5: SQL Database (Microsoft Fabric) metadata source details



- Connection Name: Enter a meaningful name. The name is displayed to Cloudera Data Lineage platform users.
- Application (client) ID
- Tenant ID
- Client Secret Value
- Workspace ID

  Note:

  To locate your Workspace ID in Microsoft Fabric:

  1. Open Microsoft Fabric.
  2. Open the target workspace.
  3. In your browser address bar, locate the URL. The URL follows this pattern:

     ```
     https://app.fabric.microsoft.com/groups/[***WORKSPACE ID***]/list?experience=power-bi
     ```

  4. Copy the value that appears after /groups/. This value is your Workspace ID.
- Parse, Analyze, and Tag the Views and Stored Procedures as ETLs: optional. Select this option when you want Cloudera Data Lineage to treat views and stored procedures as ETL objects during analysis.

Microsoft Fabric Warehouse [Technical Preview]

Learn how to set up Microsoft Fabric Warehouse as a metadata source for Cloudera Data Lineage (formerly Octopai), including Entra ID application registration, ODBC driver installation, tenant settings, and workspace permissions.

Note: This feature is in Technical Preview and is not ready for production deployments. Cloudera recommends trying this feature in test or development environments and encourages you to provide feedback on your experiences.

Overview

Before you set up the Microsoft Fabric Warehouse connector, ensure that you have the following:

- **Microsoft Entra application credentials:** a registered service principal with a tenant ID, client ID, and client secret. See Application registration (service principal).
- **Microsoft ODBC Driver 18 for SQL Server:** installed on the Windows machine that hosts the Cloudera Data Lineage platform for Warehouse SQL endpoint connectivity. See SQL access (ODBC).
- **Microsoft Fabric API permissions:** item-level API permissions assigned to the service principal, including Ware house.Read.All and related scopes. See Required application permissions.
- **Fabric workspace permissions:** the Contributor role assigned to the service principal on the target workspace. See Workspace permissions.

Application registration (service principal)

1. In the Azure portal, open Microsoft Entra ID App registrations , and select New registration.
2. Give the application a name, and select Accounts in this organizational directory only.

   Note:

   Redirect URI is optional and is not needed for client credentials.
3. Select Register.
4. From the application overview, record the Directory (tenant) ID and Application (client) ID.
5. Create a Client secret under Manage Certificates and secrets .

The Cloudera Data Lineage Microsoft Fabric Warehouse connector authenticates using a Microsoft Entra service principal. The following values are required for configuration:

- Directory (tenant) ID
- Application (client) ID
- Client secret

SQL access (ODBC)

Microsoft Fabric Warehouse metadata extraction requires SQL connectivity through the Warehouse SQL endpoint. Install Microsoft ODBC Driver 18 for SQL Server on the Windows machine that hosts the Cloudera Data Lineage platform.

For Windows, see Download ODBC Driver for SQL Server.

Identity prerequisites

A Microsoft Entra application registration (service principal) is required and is used for:

- API authentication
- SQL connectivity through the Warehouse SQL endpoint
- Access to Fabric workspace items through Microsoft Fabric APIs

Tenant settings (Fabric admin portal)

Use the Fabric admin portal to enable the settings required by the connector:

1. Open Microsoft Fabric.
2. Select the gear icon, then select Admin portal.
3. Open Tenant settings.
4. Find and enable Service principals can call Fabric public APIs.
5. Find and enable Allow apps to access Fabric items.

Workspace permissions

Assign the service principal to the target workspace with the Contributor role. Contributor access is required for complete extraction coverage, including:

- Stored procedures
- Functions
- Complete metadata visibility (such as definitions or system views)

Important:

Cloudera recommends Contributor access for all Cloudera Data Lineage Microsoft Fabric Warehouse integrations. Lower-privilege roles result in incomplete metadata extraction.

Note:

Cloudera Data Lineage uses Contributor access in a read-only manner. The connector only retrieves workspace and item metadata, and item definitions required for discovery and analysis. The connector does not create, update, delete, publish, rebind, or change permissions in the workspace.

Required application permissions

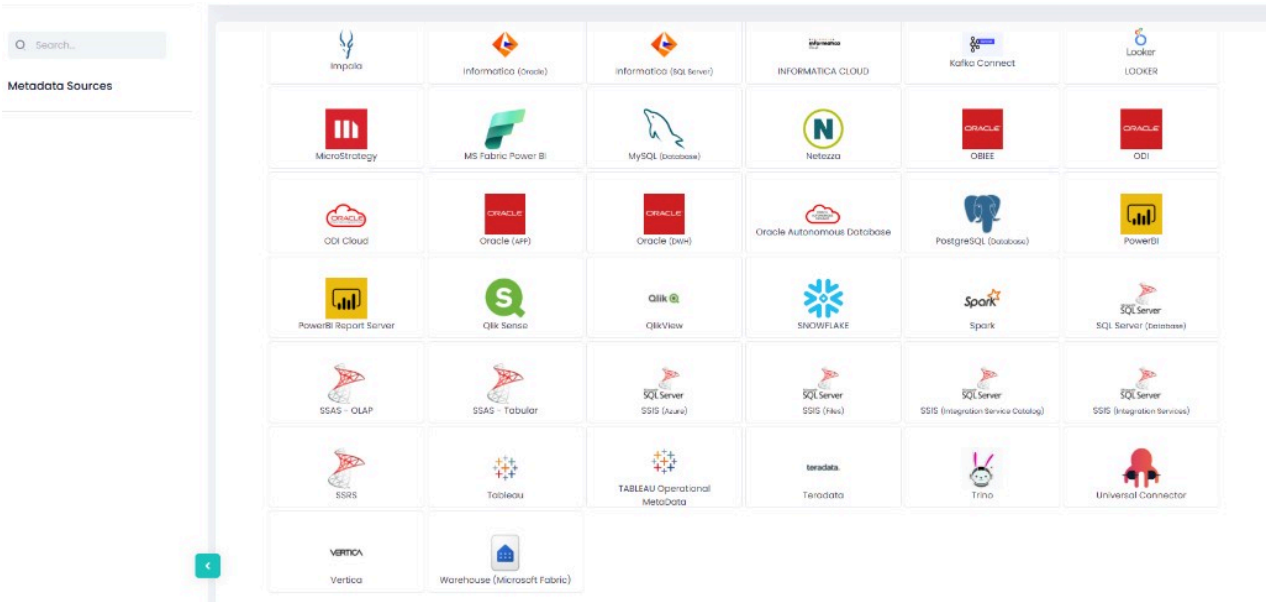
The application requires the following Microsoft Fabric API permissions:

- Warehouse.Read.All
- Workspace.Read.All
- SQLEndpoint.Read.All
- Item.Read.All

Setting up Warehouse (Microsoft Fabric) metadata source

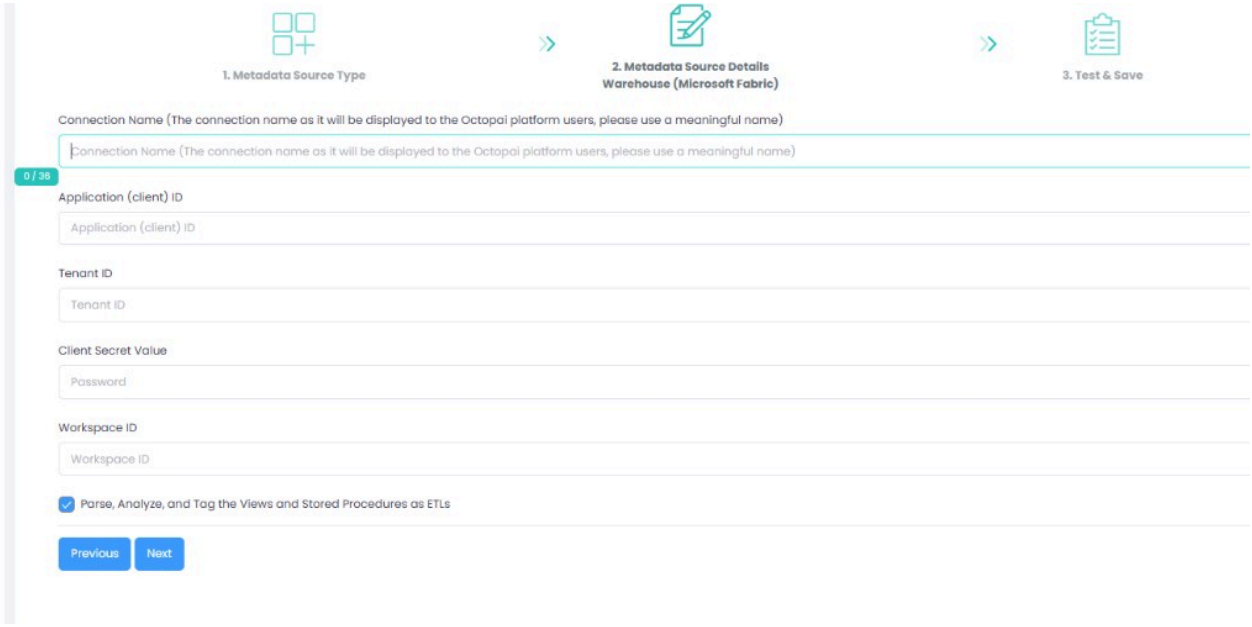
Configure the metadata source on the Octopai Client. In the New Metadata Source wizard, on Metadata Source Type, select Warehouse (Microsoft Fabric).

Figure 6: Selecting Warehouse (Microsoft Fabric) as the metadata source type



On Metadata Source Details, enter the following:

Figure 7: Warehouse (Microsoft Fabric) metadata source details



- Connection Name: enter a meaningful name (it is displayed to Cloudera Data Lineage platform users).
- Application (client) ID
- Directory (tenant) ID
- Client secret
- Workspace ID

Note:

To locate your Workspace ID in Microsoft Fabric:

1. Open Microsoft Fabric.
2. Open the target workspace.
3. In your browser address bar, locate the URL. The URL follows this pattern:

```
https://app.fabric.microsoft.com/groups/[***WORKSPACE ID***]/list?
experience=power-bi
```

4. Copy the value that appears after /groups/. This value is your Workspace ID.

MySQL

How to set up MySQL metadata sources, configure permissions, and verify extracted files.

Tool Permissions Prerequisites

Warning: Missing permissions could end up in broken lineages.

MySQL ODBC driver installed on the machine running the Octopai Client

Open Server Port for each MySQL Database Connection

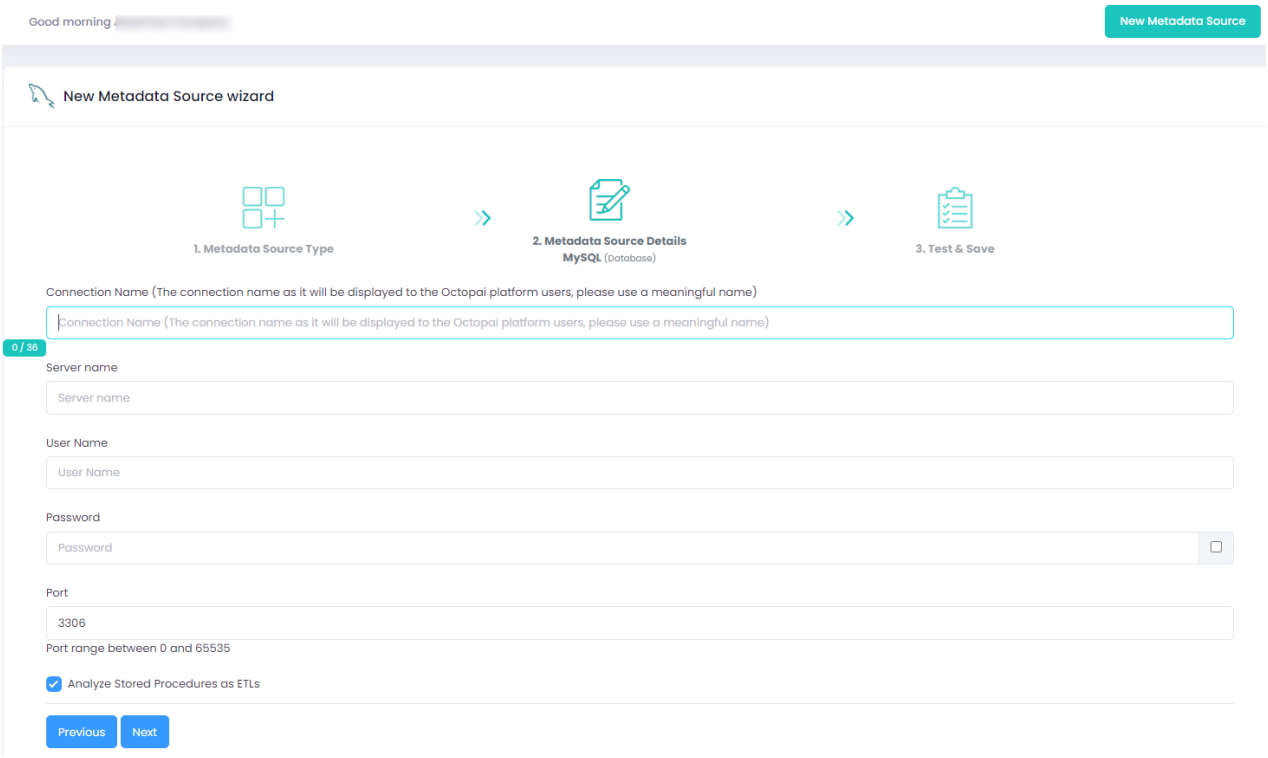
Existing/New MySQL user (OCTOPAI_USER) for each connection with grant select permission for the following dictionary tables:

- information_schema.tables;
- information_schema.columns;
- information_schema.key_column_usage;

- information_schema.table_constraints;
- information_schema.routines;

Setting up MySQL Metadata Source

Metadata Sources are set on the Octopai Client



How to set up the permissions

MySQL :: Download Connector/ODBC

Verify extraction results

After a successful run, confirm the metadata archive in the target folder. See Verify extracted metadata files.

Troubleshoot

If an error occurs during the extraction, see Troubleshoot connector extraction failures.

Netezza

Learn how to set up Netezza metadata sources with the Octopai Client.

Note: Version supported: v7.2

Tool Permissions Prerequisites

Warning:

Missing permissions could end up in broken lineages.

Check/Install Netezza ODBC on the server of the Octopai Client

Open Server Port for each Netezza Database Connection

Admin Netezza user for each connection with grant permission for the following dictionary tables:

- SYSTEM.ADMIN(ADMIN)=> select * from _v_sys_columns;
- SYSTEM.ADMIN(ADMIN)=> select * from _v_synonym;
- SYSTEM.ADMIN(ADMIN)=> select * from _v_objects;
- SYSTEM.ADMIN(ADMIN)=> select * from _v_database;
- SYSTEM.ADMIN(ADMIN)=> select * from _v_procedure;
- SYSTEM.ADMIN(ADMIN)=> select * from _v_table;
- SYSTEM.ADMIN(ADMIN)=> select * from _v_view;

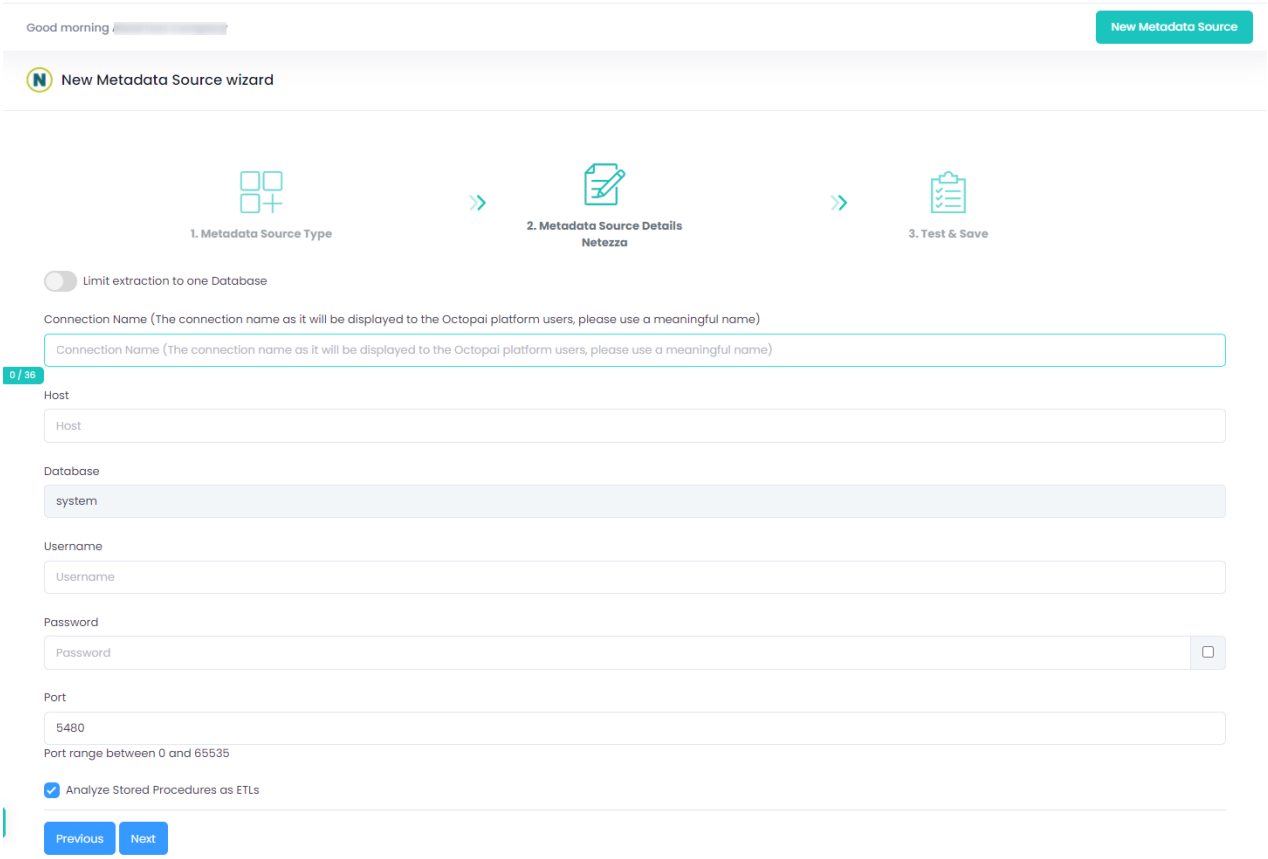
Only if previous grants are not enough:

Netezza Grant Permission Steps:

- SYSTEM.ADMIN(ADMIN)=> CREATE USER <UserName> WITH PASSWORD '<UserPassword>';
- SYSTEM.ADMIN(ADMIN)=> GRANT SELECT ON ALL.ALL.TABLE TO <UserName>;
- SYSTEM.ADMIN(ADMIN)=> GRANT SELECT ON _T_DATABASE TO <UserName>;
- SYSTEM.ADMIN(ADMIN)=> SET CATALOG <Database Name>;
- <Database Name>.ADMIN(ADMIN)=> GRANT LIST ON DATABASE to <UserName>;
- <Database Name>.ADMIN(ADMIN)=> GRANT ALL ON TABLE, FUNCTION, AGGREGATE, MATERIALIZED VIEW, PROCEDURE, SEQUENCE, SYNONYM, VIEW, SYSTEM VIEW TO <UserName>;

Setting up Netezza Metadata Source

Metadata Sources are set on the Octopai Client



Verify extraction results

After a successful run, confirm the metadata archive in the target folder. See Verify extracted metadata files.

Troubleshoot

If an error occurs during the extraction, see Troubleshoot connector extraction failures.

Oracle (DWH)

Learn how to configure and manage Oracle (DWH) metadata sources using Cloudera Data Lineage (formerly Octopai).

Note: Version supported: 11g.x, 12c, 12.1

Tool Permissions Prerequisites

Warning:

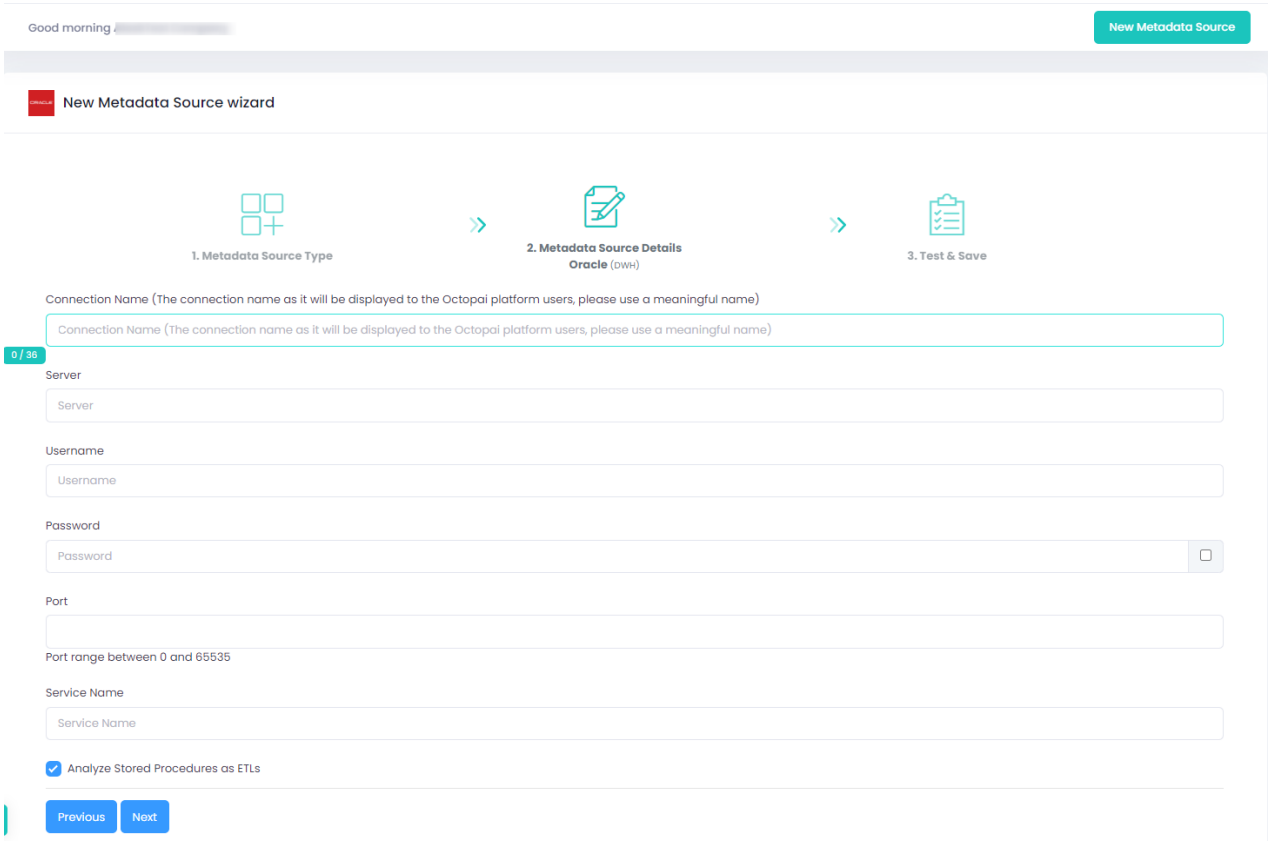
Missing permissions could end up in broken lineages.

- Open Server Port for each Oracle Database Connection
- Existing/New Oracle user for each database with 'READ' permissions for the following dictionary tables:
  - DBA_INDEXES
  - DBA_SOURCE
  - DBA_DEPENDENCIES
  - DBA_CATALOG
  - DBA_VIEWS
  - DBA_TAB_COLUMNS
  - DBA_IND_COLUMNS
  - DBA_SYNONYMS
  - DBA_MVIEWS

Setting up Oracle Metadata Source

Metadata Sources are set on the Octopai Client

Figure 8: New Metadata Source wizard



Verify extraction results

After a successful run, confirm the metadata archive in the target folder. See Verify extracted metadata files.

Troubleshoot

If an error occurs during the extraction, see Troubleshoot connector extraction failures.

Oracle Autonomous Data Warehouse (OAD/ADW)

Learn how to configure Oracle Autonomous Data Warehouse (OAD/ADW) as a metadata source for Cloudera Data Lineage (formerly Octopai).

Note: Version Supported: 19c / 20c / 21c

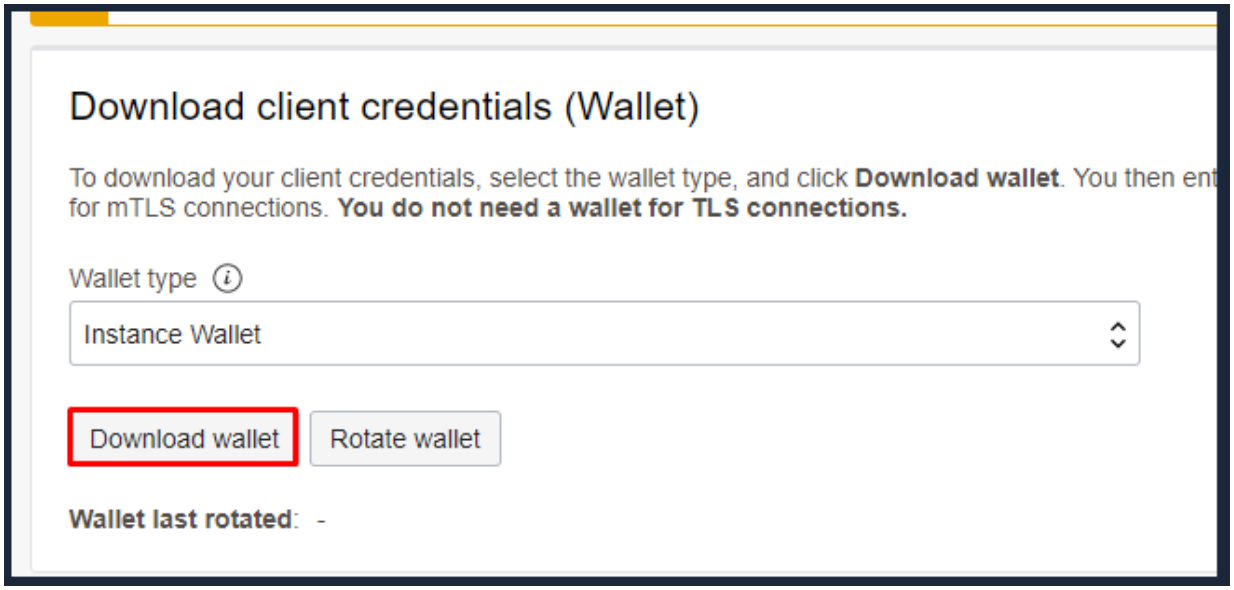
Tool Permissions Prerequisites

Warning: Missing permissions could end up in broken lineages.

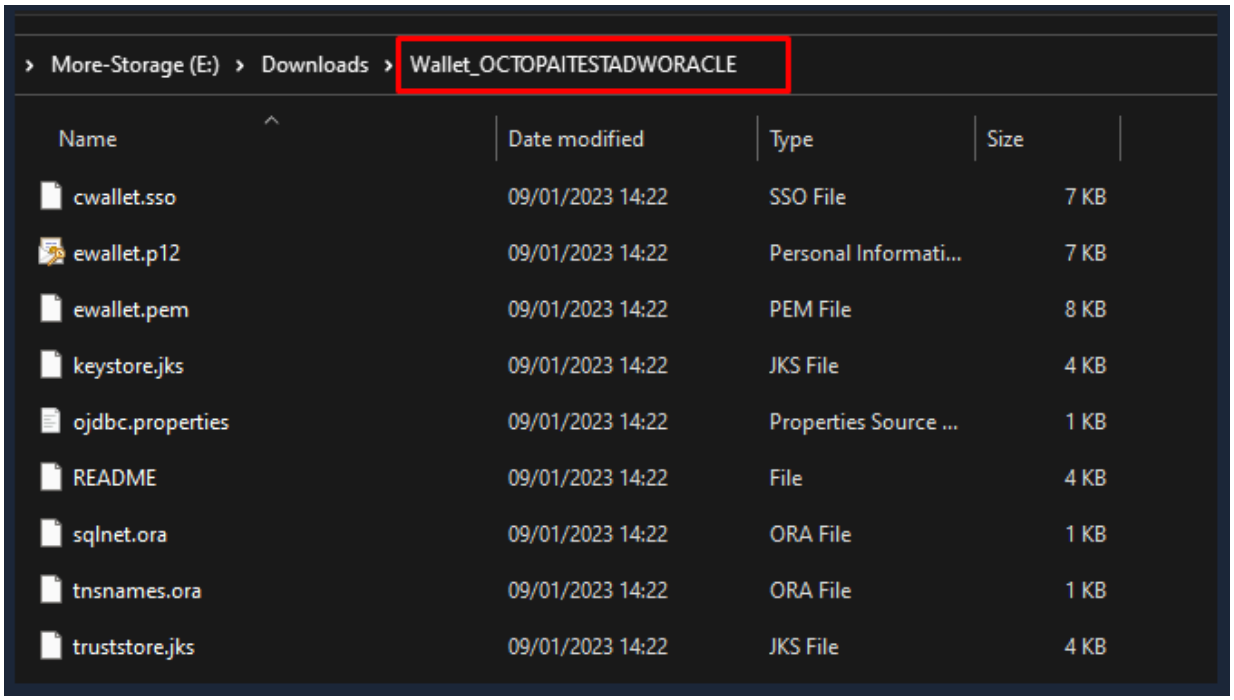
1. Log into OCI, go to databases.
2. Select the database you want to connect and click on “Database connection”.



3. Set the wallet type to “Instance Wallet”, and download the file.

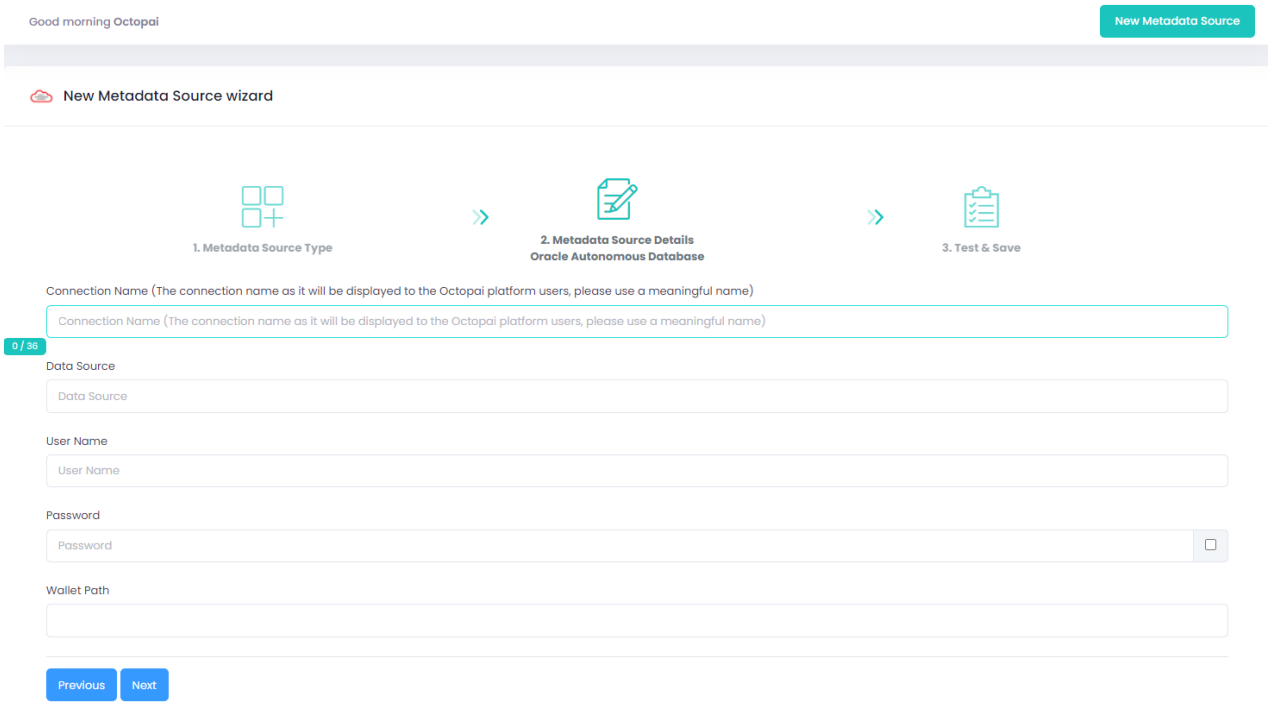


4. Extract the content of the zip into a folder, it should look like this:



Setting up OAD Metadata Source

Metadata Sources are set on the Octopai Client



Verify extraction results

After a successful run, confirm the metadata archive in the target folder. See Verify extracted metadata files.

Troubleshoot

If an error occurs during the extraction, see Troubleshoot connector extraction failures.

PostgreSQL

How to set up PostgreSQL metadata sources in Cloudera Data Lineage (formerly Octopai), including prerequisites, verification, and troubleshooting.

Note: Version supported: up to 16.x

Tool Permissions Prerequisites

Warning: Missing permissions could end up in broken lineages.

- Postgres ODBC driver installed on the machine running the Octopai Client
- Open Server Port for each Postgre Database Connection
- Existing/New Postgres user (OCTOPAI_USER) for each connection with grant select permission for the following dictionary tables:
  - information_schema.tables;
  - information_schema.views;

- information_schema.columns;
- pg_catalog.pg_proc;
- pg_catalog.pg_namespace;
- pg_catalog.pg_attribute;
- pg_catalog.pg_constraint;
- pg_catalog.pg_class;
- pg_catalog.pg_database;

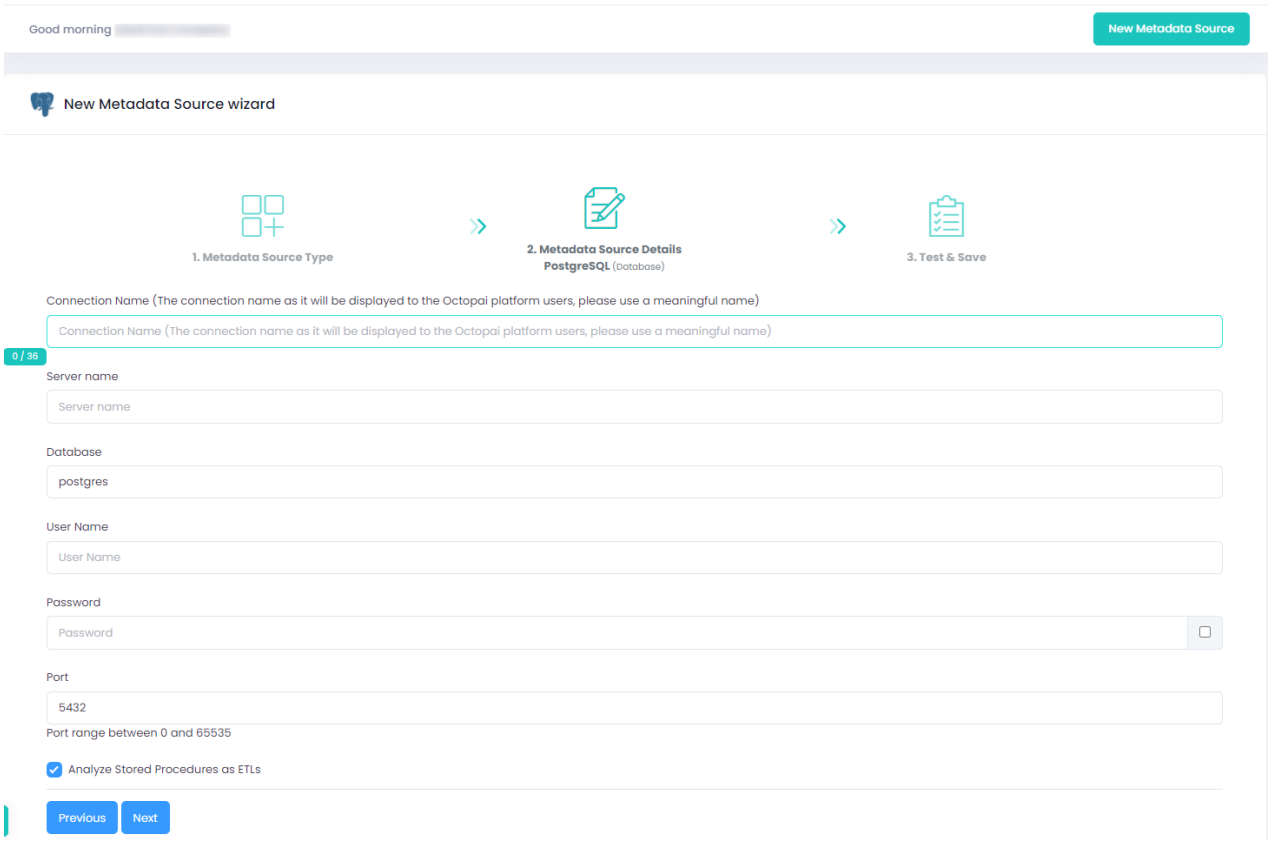
How to set up the permissions

psqlodbc - PostgreSQL ODBC driver

Setting up PostgreSQL Metadata Source

Metadata Sources are set on the Octopai Client

Figure 9: New Metadata Source wizard



Verify extraction results

After a successful run, confirm the metadata archive in the target folder. See Verify extracted metadata files.

Troubleshoot

If an error occurs during the extraction, see Troubleshoot connector extraction failures.

Snowflake Tool Permissions Prerequisites

Learn how to configure Snowflake permissions to enable seamless integration with Cloudera Data Lineage (formerly Octopai).

Note: Version supported: 2025

Permissions overview

Warning:

Missing permissions can result in incomplete or broken lineage. Proper role assignments are critical for a successful Cloudera Data Lineage–Snowflake integration.

This guide explains how to create a dedicated user and role, grant the required permissions, validate configuration, and revoke access if needed. Replace placeholder values such as *[***USERNAME***]* and *[***WAREHOUSE NAME***]* with your actual Snowflake details.

General guidelines

- Create the Snowflake user while signed in with an **ACCOUNTADMIN** role.
- Assign required roles and a **DEFAULT_WAREHOUSE** to ensure efficient query processing.
- Grant **USAGE** on all schemas in the target databases to simplify access management.

Note:

Snowflake object names and settings (including warehouses) are case sensitive. Keep capitalization consistent to avoid errors.

Database and warehouse roles

Database roles

- USAGE: Lists the database and allows metadata queries.
- CREATE SCHEMA: Creates schemas within the database.
- CREATE TABLE: Creates tables within the database.
- SELECT: Queries data from tables.
- INSERT: Inserts data into tables.
- UPDATE: Updates table data.
- DELETE: Deletes table data.
- REFERENCES: Creates foreign-key relationships.

Warehouse roles

- USAGE: Runs queries on the warehouse.
- MONITOR: Reviews warehouse usage and performance.
- OPERATE: Starts, stops, and resizes the warehouse.
- OWNERSHIP: Grants full control over the warehouse.

Set up Snowflake permissions

Follow these steps to provision a dedicated Cloudera Data Lineage user and assign the necessary Snowflake roles and privileges.

1. Create a dedicated user

Run the following SQL to create a Snowflake user for Cloudera Data Lineage metadata extraction:

```
CREATE USER [***USERNAME***]
  PASSWORD = '[***PASSWORD***]'
  DEFAULT_WAREHOUSE = '[***WAREHOUSE NAME***]'
  MUST_CHANGE_PASSWORD = false;
```

Result: the user appears in Snowflake.

Figure 10: Sample user creation output

Row	status
1	User DORA_TEST successfully created.

Note:

If you use Okta authentication, create the user without a PASSWORD clause so federated sign-on is required. You cannot create passwordless users in the Snowflake web UI; use SQL as shown below. See step 3 in Okta authentication setup overview.

```
CREATE USER [***USERNAME***]
  DEFAULT_WAREHOUSE = '[***WAREHOUSE NAME***]'
  MUST_CHANGE_PASSWORD = FALSE;
```

2. Create a dedicated role

Create a role to encapsulate Cloudera Data Lineage permissions:

```
CREATE ROLE [***ROLE NAME***];
```

Result: the role is registered in Snowflake.

Figure 11: Role creation confirmation

Row	status
1	Role OCTOPAI_SAMPLE_ROLE successfully created.

3. Assign the role to the user

Grant the role and set it as the default:

```
GRANT ROLE [***ROLE NAME***] TO USER [***USERNAME***];
```

```
ALTER USER [***USERNAME***] SET DEFAULT_ROLE = [***ROLE NAME***];
```

4. Grant warehouse usage

Allow the role to run workloads on the chosen warehouse:

```
GRANT USAGE ON WAREHOUSE [***WAREHOUSE NAME***] TO ROLE [***ROLE NAME***];
```

Verify warehouse grants when needed:

```
SHOW GRANTS ON WAREHOUSE [***WAREHOUSE NAME***];
```

Figure 12: Warehouse grants report



privilege	granted_on	name	grantee_name	granted_by
OWNERSHIP	WAREHOUSE	OCTOPAI_WH	ACCOUNTADMIN	ACCOUNTADMIN
USAGE	WAREHOUSE	OCTOPAI_WH	OCTOPAI_SAMPLE_ROLE	ACCOUNTADMIN

5. Grant imported privileges on the Snowflake database

Provide access to the shared SNOWFLAKE database:

```
GRANT IMPORTED PRIVILEGES ON DATABASE SNOWFLAKE TO ROLE [***ROLE NAME***];
```

6. Grant object-specific permissions

For metadata extraction, Cloudera Data Lineage primarily reads object metadata from SNOWFLAKE.ACCOUNT_USAGE views after you grant IMPORTED PRIVILEGES ON DATABASE SNOWFLAKE in step 5. Ensure the service role can see the required objects in Account Usage for the databases and schemas in scope.

When definitions are missing or additional object details are needed, Cloudera Data Lineage may also run supplemental commands such as Get_DDL('*[***OBJECT TYPE***]*', '*[***OBJECT NAME***]*'), SHOW STREAMS, SHOW TASKS, SHOW MATERIALIZED VIEWS, SHOW WAREHOUSES, and LIST @stage. Ensure that the service role has sufficient database- and schema-level visibility and privileges for those commands to succeed.

Important: Missing Account Usage visibility or supplemental privileges can result in partial extraction rather than a complete connector failure.

Grant the following permissions when supplemental access is required for your environment:

Functions

```
GRANT USAGE ON FUNCTION [***DATABASE NAME***].[***SCHEMA
 NAME***].[***FUNCTION NAME***]([***DATA TYPE 1***],[***DATA TYPE
 2***], [***ADDITIONAL DATA TYPES***]) TO ROLE [***ROLE NAME***];
```

Procedures

```
GRANT USAGE ON PROCEDURE [***DATABASE NAME***].[***SCHEMA
 NAME***].[***PROCEDURE NAME***]([***DATA TYPE 1***],[***DATA TYPE
 2***], [***ADDITIONAL DATA TYPES***]) TO ROLE [***ROLE NAME***];
```

Pipes

```
GRANT USAGE ON DATABASE [***DATABASE NAME***] TO ROLE [***ROLE NAME***];
GRANT USAGE ON SCHEMA [***DATABASE NAME***].[***SCHEMA NAME***] TO
ROLE [***ROLE NAME***];
```

Dynamic tables

```
GRANT USAGE ON DATABASE [***DATABASE NAME***] TO ROLE [***ROLE NAME***];
GRANT USAGE ON SCHEMA [***DATABASE NAME***].[***SCHEMA NAME***] TO
ROLE [***ROLE NAME***];
GRANT SELECT ON ALL DYNAMIC TABLES IN SCHEMA [***SCHEMA NAME***] TO
ROLE [***ROLE NAME***];
```

Optional: enable access to future dynamic tables.

```
GRANT SELECT ON FUTURE DYNAMIC TABLES IN SCHEMA [***SCHEMA NAME***] TO
ROLE [***ROLE NAME***];
```

Troubleshooting checks

Use these queries to validate role assignments and investigate issues:

- Check database-level grants:

```
SHOW GRANTS TO ROLE [***ROLE NAME***] ON DATABASE [***DATABASE NAME***];
```

- Review default warehouse and role settings:

```
SHOW USERS LIKE '%[***USERNAME***]%';
```

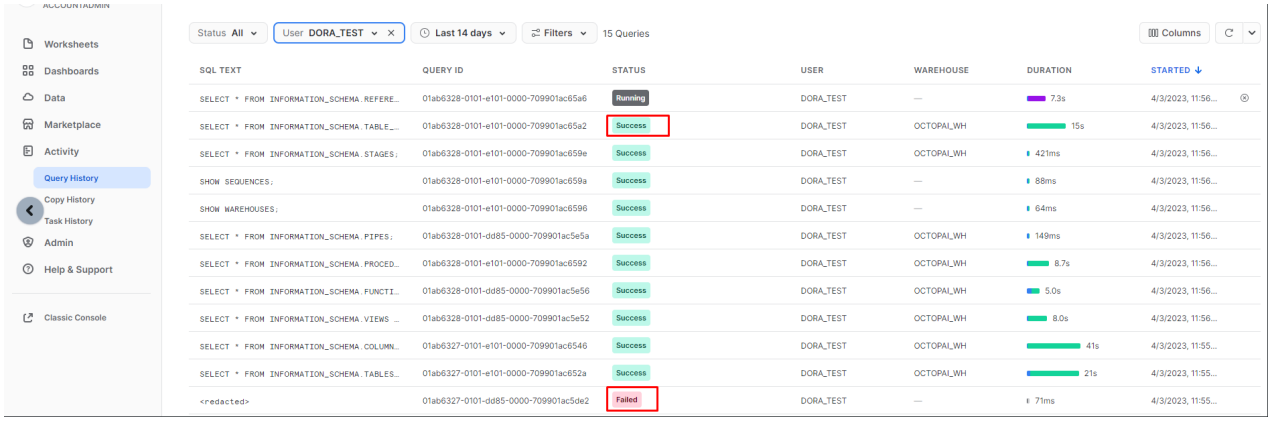
```
SHOW ROLES LIKE '%[***ROLE NAME***]%';
```

- Confirm warehouse grants:

```
SHOW GRANTS ON WAREHOUSE [***WAREHOUSE NAME***];
```

- Inspect query history logs through the Snowflake UI.

Figure 13: Query history filtered by role



Revoke warehouse permissions

Use revoke statements when access must be removed:

```
REVOKE [***PRIVILEGE***] ON WAREHOUSE [***WAREHOUSE NAME***] FROM RO
LE [***ROLE NAME***];
```

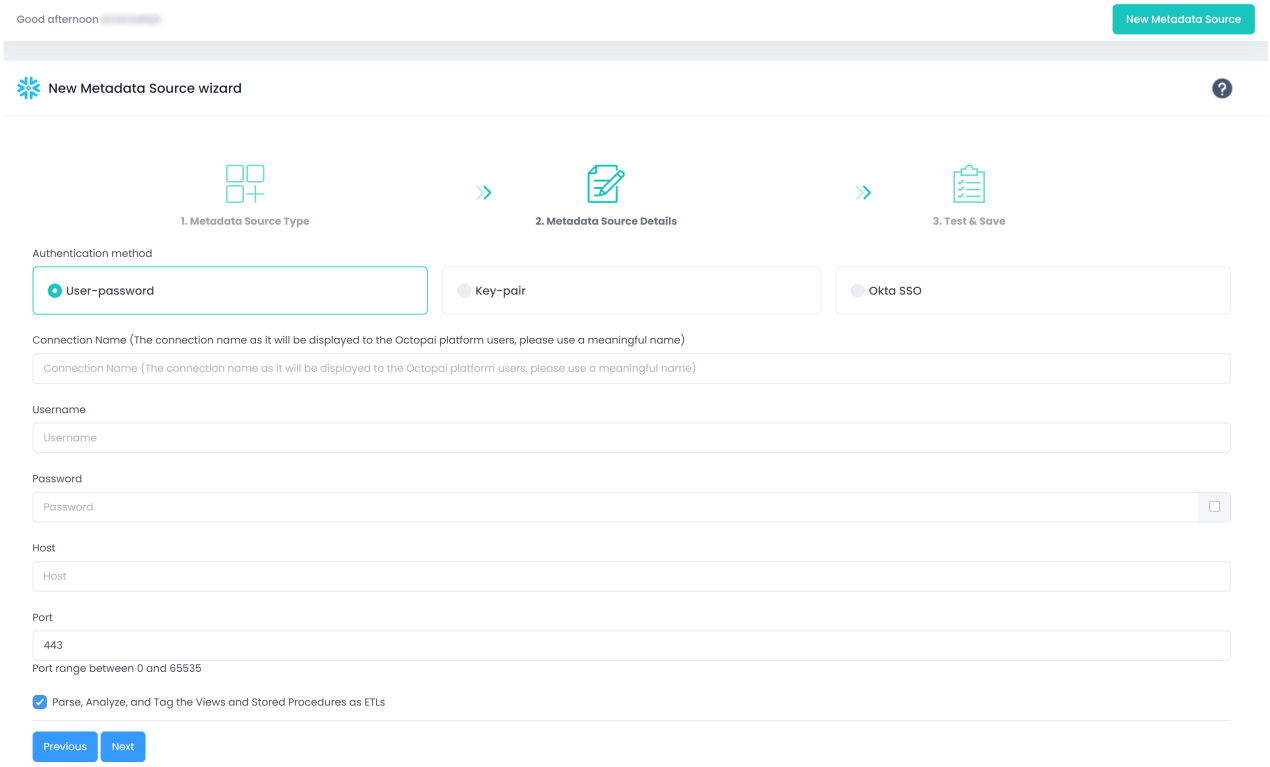
Example:

```
REVOKE MONITOR, OPERATE, USAGE ON WAREHOUSE my_warehouse FROM ROLE my_role;
```

Set up the Snowflake metadata source

Configure metadata sources through the Octopai Client.

Figure 14: Metadata source configuration in the Octopai Client



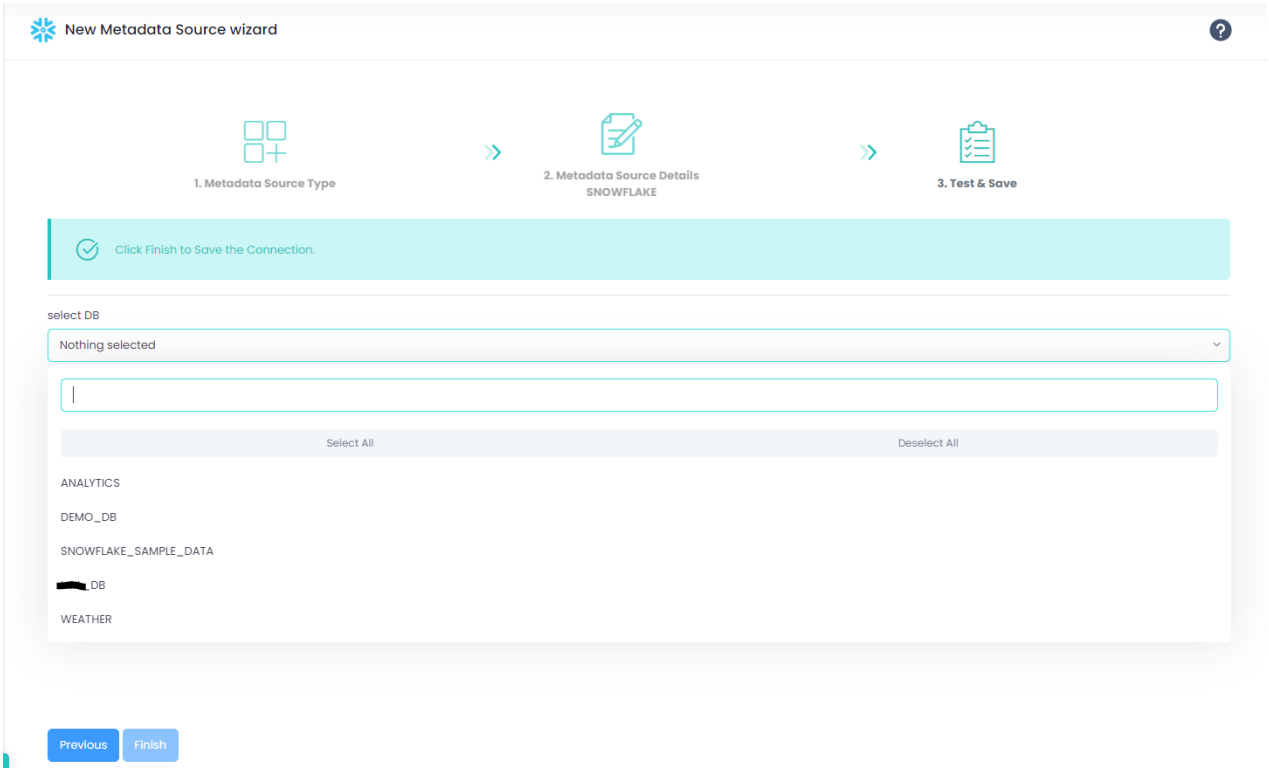
In the metadata source wizard, select an authentication method:

- User-password — enter the Snowflake user name and password for the service account.
- Key-pair — authenticate with an encrypted private key and optional passphrase. See Key-pair configuration options.
- Okta SSO — authenticate using your Okta account URL as the Snowflake authenticator parameter. See Setting up Okta SSO metadata source.

Enter the remaining connection settings for your selection, then run the connection test.

After the connection test succeeds, select the databases to scan. The list is populated from the SNOWFLAKE.ACCOUNT_USAGE.DATABASES view. The service role must have IMPORTED PRIVILEGES ON DATABASE SNOWFLAKE. See step 5.

Figure 15: Database selection list



Snowflake connector authentication options

Cloudera Data Lineage supports multiple authentication methods when connecting to Snowflake for metadata extraction:

- User-password — enter the Snowflake user name and password in the metadata source wizard.
- Key-pair — authenticate with an encrypted private key and optional passphrase instead of a Snowflake password.
- Okta SSO — authenticate through Okta SAML 2.0 federated sign-on using your Okta account URL as the Snowflake authenticator parameter.

Cloudera recommends key-pair or Okta authentication as Snowflake deprecates password-based programmatic access. See Why migrate from passwords.

Why migrate from passwords

Snowflake is retiring single-factor password authentication for many accounts:

- **November 2025:** password-only sign-ins are blocked for service and human users. See Snowflake announcement and community update.
- **March 2026:** programmatic access for password-based legacy service accounts is fully disabled. Refer to Snowflake MFA rollout documentation.

Note:

Cloudera Data Lineage continues to support user name and password authentication in the connector, but for new Snowflake connections Cloudera recommends key-pair authentication or Okta federated sign-on (SAML 2.0), especially as Snowflake deprecates password-based programmatic access.

Key-pair authentication

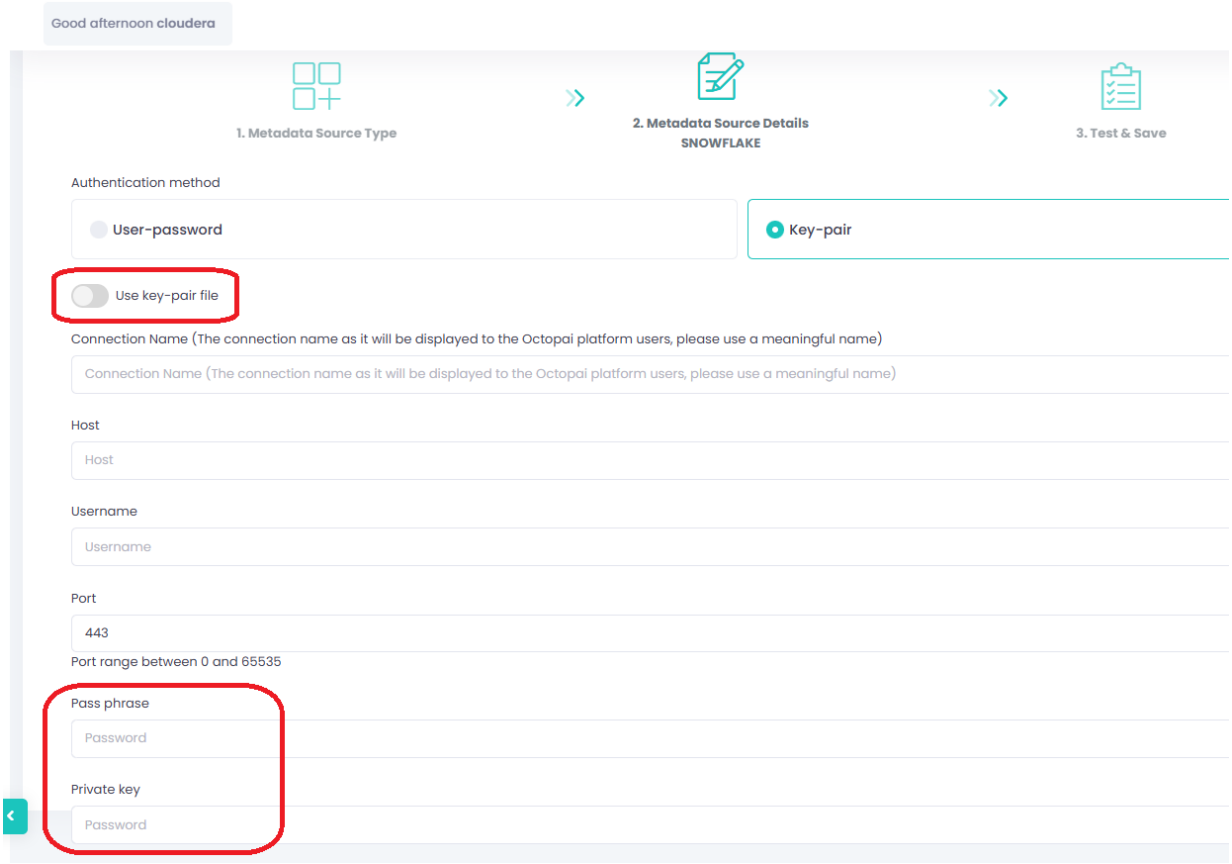
Cloudera Data Lineage supports **key-pair authentication** as a secure alternative to user name and password sign-in.

Key-pair configuration options

You can configure key-pair authentication in two ways:

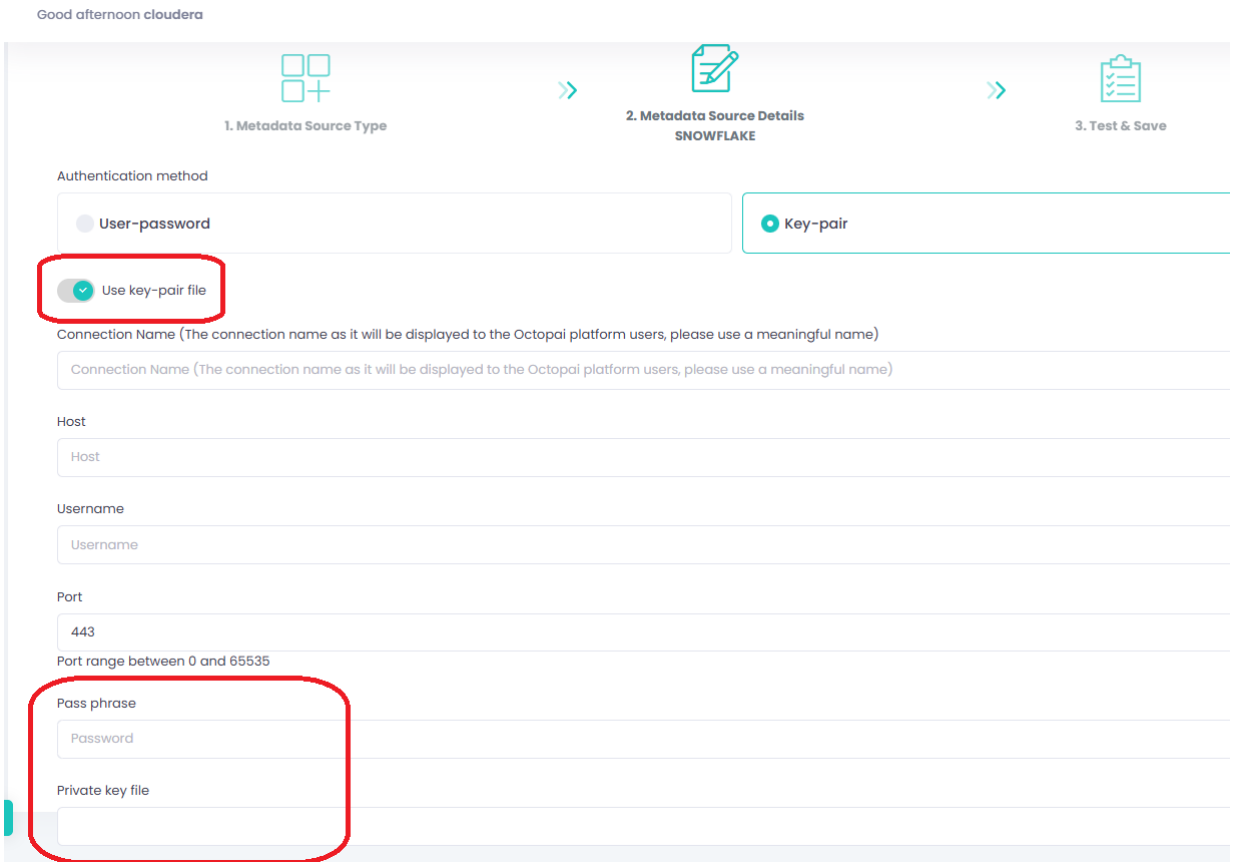
- Paste the encrypted private key and passphrase directly into the Cloudera Agent configuration.

Figure 16: Pasting an encrypted private key



- Enable Use key-pair file and provide the file path to the private key file and an optional passphrase.

Figure 17: Referencing a private key file



Both methods ensure secure authentication during metadata extraction.

Recommended actions

- Configure key-pair authentication in Cloudera Data Lineage using one of the available methods.
- Ensure the Snowflake service account is marked as a SERVICE user (not LEGACY_SERVICE).
- Plan your migration before Snowflake disables password-based programmatic access for your account, to avoid service disruptions.

Review the full setup steps in the Snowflake Key Pair Authentication Guide.

Okta authentication

Cloudera Data Lineage supports Okta as an identity provider (IdP) for Snowflake through SAML 2.0 federated authentication. When Okta is configured, the connection uses your Okta account URL as the Snowflake authenticator parameter. Select Okta SSO in the metadata source wizard to use this method.

Okta authentication requirements

Before you configure the Snowflake connector for Okta authentication, ensure the following:

- An Okta organization with SAML 2.0 federated sign-on configured for your Snowflake account.
- Snowflake federated authentication enabled through a SAML2 security integration. You must have the ACCOUNTADMIN role to create or update this integration.
- A dedicated Snowflake user for Cloudera Data Lineage metadata extraction, with the roles and warehouse access in Set up Snowflake permissions.
- The Snowflake user assigned to the Snowflake SAML application in Okta.

- For programmatic connections such as metadata extraction, native SSO through Okta. You must set the authenti cator login parameter to your Okta account URL (for example, https://*[***OKTA SUBDOMAIN***]*.okta.com). native SSO does not use a web browser and is supported only when Okta is the IdP.

Note:

If you use native SSO through Okta for programmatic access, disable Okta MFA for that user. To use MFA with Okta, configure browser-based SSO instead. For details, see Native SSO — Okta only.

Okta authentication setup overview

Configure Okta and Snowflake federated authentication before you add the metadata source in Cloudera Data Lineage:

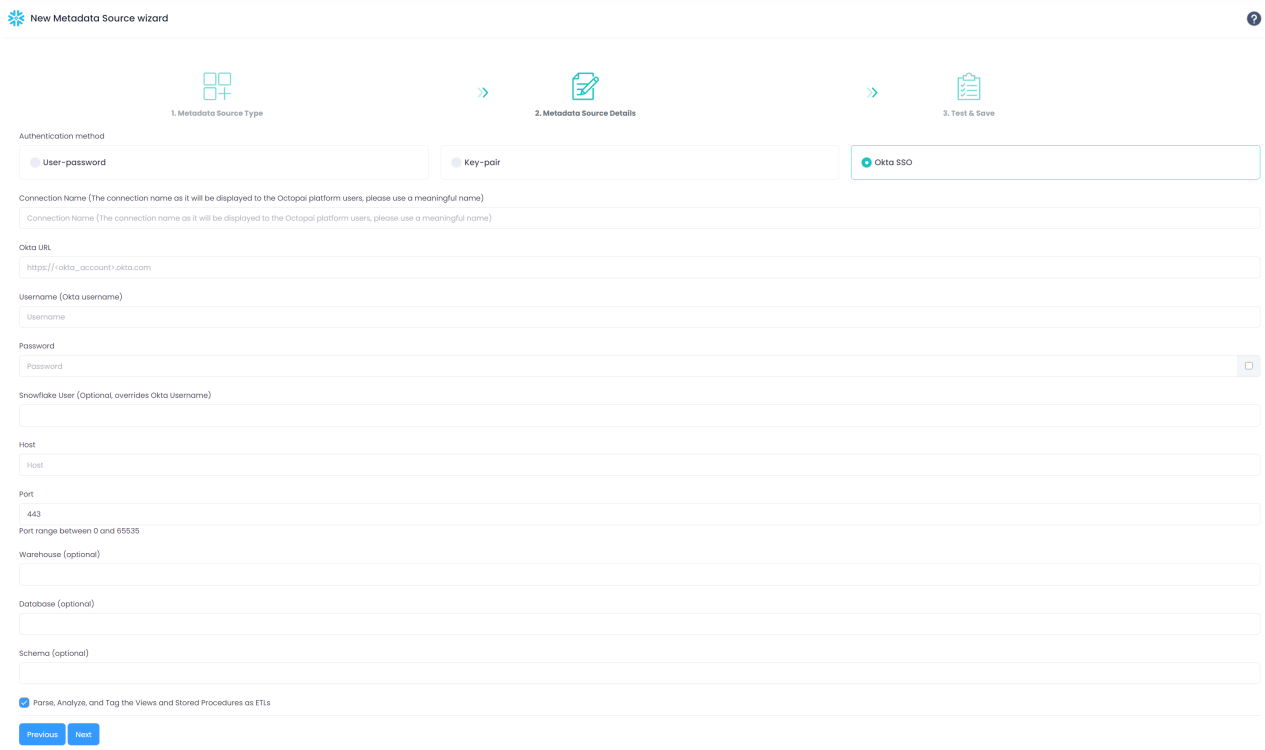
1. In Okta, configure the Snowflake SAML application and assign the service account user.
2. In Snowflake, create or update the SAML2 security integration that trusts Okta as the IdP.
3. Create the Snowflake service user without a Snowflake-managed password so that federated sign-on is required.
4. Grant the user the Snowflake roles and privileges required for metadata extraction, as described in Set up Snowflake permissions.
5. In Cloudera Data Lineage, configure the Snowflake metadata source with Okta SSO as described in Setting up Okta SSO metadata source.

Review the full setup steps in How to Configure SAML 2.0 for Snowflake and Using federated authentication: Native SSO — Okta only.

Setting up Okta SSO metadata source

Configure the metadata source on the Octopai Client. In Metadata Source Details, select Okta SSO, and enter the following:

Figure 18: Okta SSO metadata source configuration



- Connection Name: Enter a meaningful name. The name is displayed to Cloudera Data Lineage platform users.
- Okta URL: Enter your Okta account URL, for example https://*[***OKTA SUBDOMAIN***]*.okta.com.

- Username (Okta username): Enter the Okta user name for the service account.
- Password: Enter the Okta password for the service account.
- Snowflake User (Optional; overrides Okta Username): Enter the Snowflake user name when it differs from the Okta user name.
- Host: Enter the full Snowflake URL, including https:// (for example, http s://*[***ACCOUNT***].[***REGION***]*.snowflakecomputing.com).
- Port: Enter the Snowflake port (default 443).
- Warehouse (optional): Enter the Snowflake warehouse name.
- Database (optional): Enter the Snowflake database name.
- Schema (optional): Enter the Snowflake schema name.
- Parse, Analyze, and Tag the Views and Stored Procedures as ETLs: optional. Select this option when you want Cloudera Data Lineage to treat views and stored procedures as ETL objects during analysis.

Verify extraction results

After a successful run, confirm the metadata archive in the target folder. See Verify extracted metadata files.

Troubleshoot extraction issues

If extraction fails, confirm Snowflake permissions using Troubleshooting checks. For log collection and support escalation, see Troubleshoot connector extraction failures.

SQL Server (SQLS)

Learn how to configure and manage SQL Server connectors, including tool permissions prerequisites, metadata source setup, and troubleshooting steps. This guide also provides instructions for verifying extracted metadata files and accessing the Cloudera Data Lineage (formerly Octopai) Target Folder.



Note: Version supported: up to SQL 2022

Tool Permissions Prerequisites



Warning:

Missing permissions could end up in broken lineages.

- Open Server Port for each SQL Database Connection
- Existing/New user (SQL Server or Windows Authentication) for each server/database - with ‘grant SELECT’ for the following dictionary tables:
  - sys.objects
  - sys.schemas
  - sys.sql_modules
  - sys.columns
  - sys.types
  - sys.identity_columns
  - sys.computed_columns
  - sys.check_constraints
  - sys.synonyms
  - sys.indexes
  - sys.index_columns
  - sys.tables
  - sys.foreign_keys

    - sys.foreign_key_columns
    - sys.sysservers
    - sys.syslogins
    - msdb.sysjobs
    - msdb.sysjobsteps
- Grant execute on sys.sp_linkedservers
- Only if previous grants are not enough:
    - USE MASTER:
    - grant connect any database to "DOMAIN_NAME\OCTOPAI_USER";
    - grant view server state to "DOMAIN_NAME\OCTOPAI_USER";
    - grant view any definition to "DOMAIN_NAME\OCTOPAI_USER";
- USE MSDB:
    - grant select on msdb.dbo.sysjobsteps to "DOMAIN_NAME\OCTOPAI_USER";
    - grant select on msdb.dbo.sysjobs to "DOMAIN_NAME\OCTOPAI_USER";

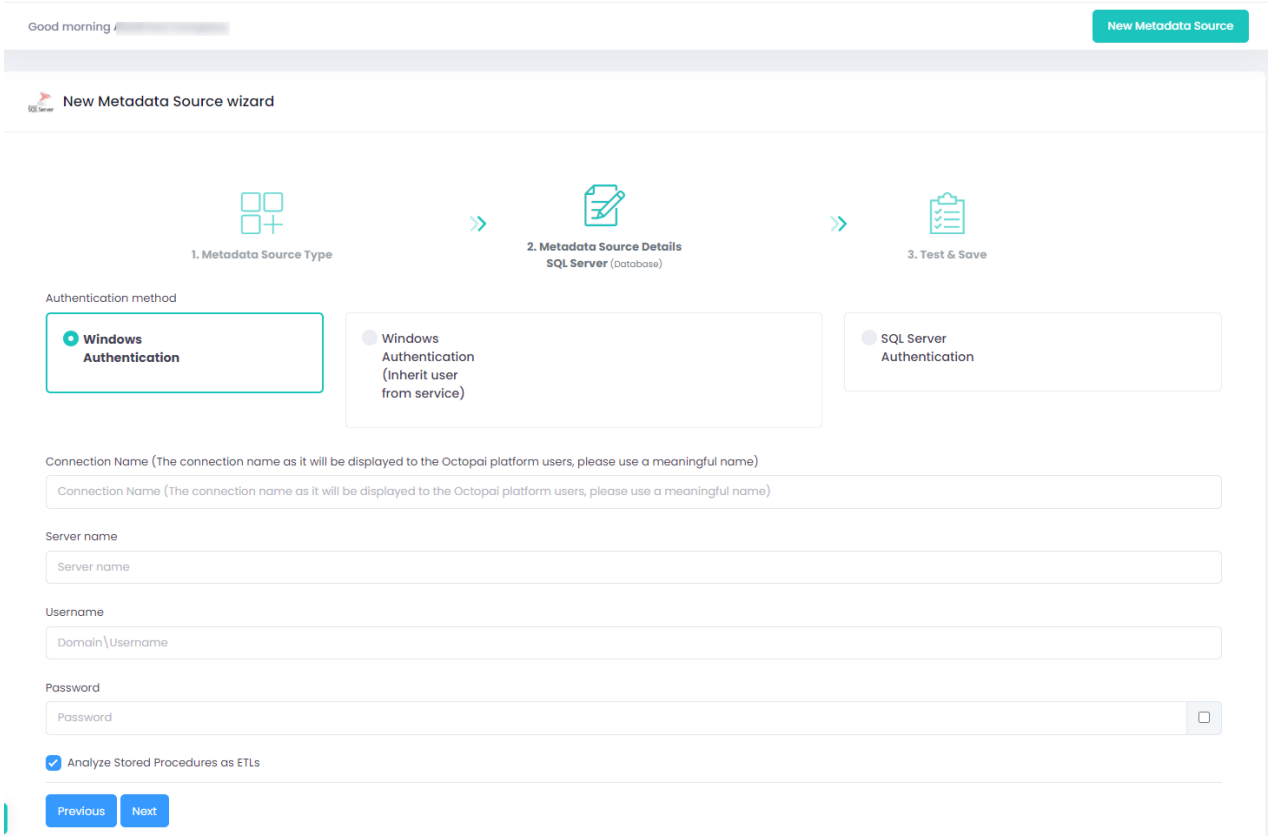
Setting up SQL Server Metadata Source

Windows Authentication: Use the server name, domain, username, and password that are recognized by the SQL Server.

Windows Authentication (inherit user from service): Run the service as another user. Important Clarification: If you choose to run the service as another user, please ensure that this user has the appropriate permissions for ALL tools used by the Octopai Client. Otherwise, the other connectors will not be able to authenticate and perform the metadata extraction.

SQL Server Authentication: Log in with an SQL Server user.

Figure 19: New Metadata Source Wizard



Verify extraction results

After a successful run, confirm the metadata archive in the target folder. See Verify extracted metadata files.

Troubleshoot

If an error occurs during the extraction, see Troubleshoot connector extraction failures.

Teradata

Learn how to set up and verify Teradata metadata sources using Cloudera Data Lineage (formerly Octopai).

Note: Version supported: up to v17 / Vantage v17

Tool Permissions Prerequisites

Warning:

Missing permissions could end up in broken lineages.

Open Server Port for each Teradata Database Connection

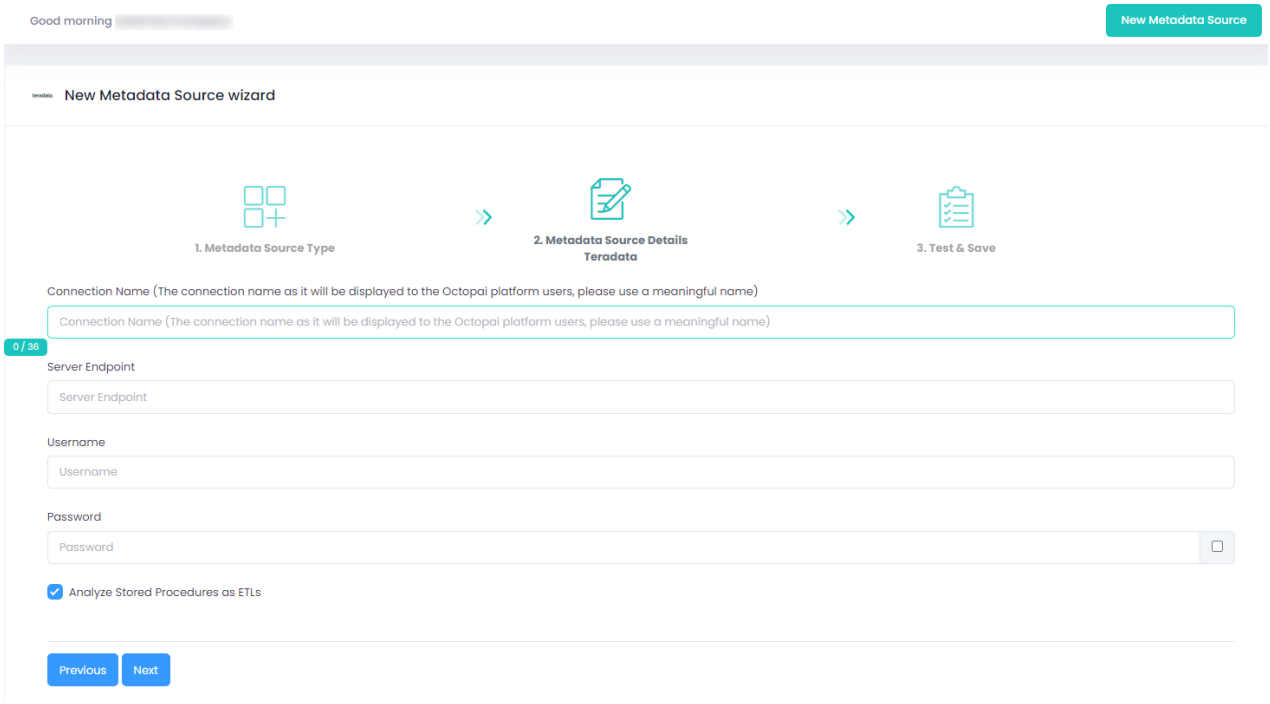
Exiting/New Teradata user (**OCTOPAI_USER***) for each connection with grant read permission for the following dictionary tables:

- dbc.tablesv
- dbc.indicesV
- dbc.tvm
- DBC.TVFields
- DBC.triggersTbl
- dbc.ColumnsV
- dbc.dbase
- dbc.texttbl
- dbc.tablesv

Setting up Teradata Metadata Source

Metadata Sources are set on the Octopai Client

Figure 20: New Metadata Source wizard



Verify extraction results

After a successful run, confirm the metadata archive in the target folder. See Verify extracted metadata files.

Troubleshoot

If an error occurs during the extraction, see Troubleshoot connector extraction failures.

Trino Connector for Cloudera Data Lineage

Learn about Trino as a metadata source for Cloudera Data Lineage (formerly Octopai), including authentication options and extracted metadata.

Overview

The Trino connector connects Cloudera Data Lineage to a Trino coordinator and collects metadata for lineage and discovery. The connector extracts metadata and configuration properties only. It does not read, query, or export table data.

Prerequisites

Before configuring the connector, verify the following requirements.

Access requirements

Ensure you have the following:

- Read access to the Trino environment.
- Network connectivity from the Octopai Client to the Trino coordinator.
- Trino host, port, and authentication credentials.

Catalog properties access

All connector settings must be defined in Trino catalog property files, typically located in **etc/catalog/**.

If you do not have access to these files or do not know their location, contact your Trino administrator for the connection details.

Kerberos requirements (if applicable)

For Kerberos authentication, you must obtain the following from your Trino administrator:

- Kerberos principal
- Keytab file
- Path to the kinit executable (KinitPath)

Define the KinitPath value in the kerberos.settings.json file.

Supported authentication methods

Cloudera Data Lineage supports the following Trino authentication types:

- LDAP (username and password)
- Kerberos (SPNEGO / Negotiate)

Connection parameters

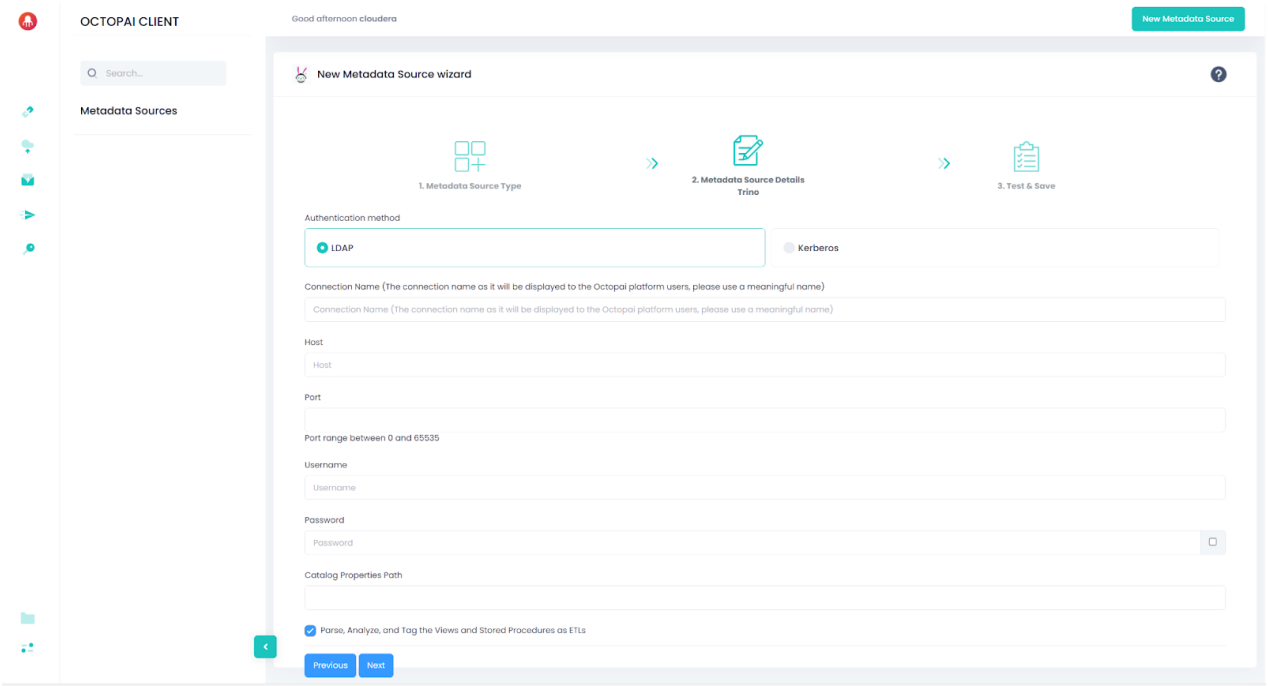
The required connection parameters depend on your authentication method.

LDAP authentication

Parameter	Description
Host	Trino coordinator hostname or IP (example: trino.example.com)
Port	Trino coordinator port (example: 443 for HTTPS)
Username	Trino user identity
Password	Trino user password
CatalogPropertiesPath	Path to directory containing Trino catalog properties files

The extractor reads catalog property files and exports non-sensitive properties only. Passwords, tokens, and keys are filtered automatically.

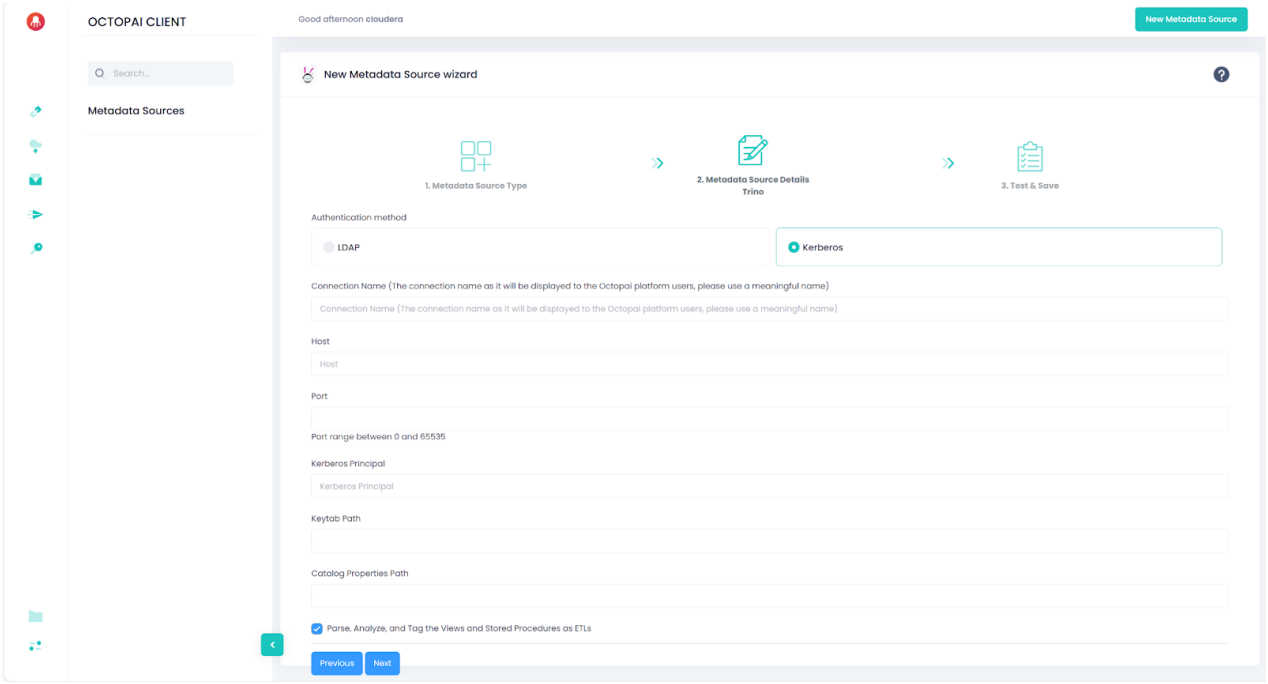
Figure 21: LDAP authentication configuration



Kerberos authentication

Parameter	Description
Host	Trino coordinator hostname or IP
Port	Trino coordinator port
KerberosPrincipal	Kerberos principal used to obtain the ticket
KeytabPath	Path to the keytab file
CatalogPropertiesPath	Path to directory containing Trino catalog properties files

Figure 22: Kerberos authentication configuration



Kerberos ticket behavior

The connector attempts to use an existing Kerberos ticket from the cache. If the ticket is expired or invalid, the extractor runs the following command to obtain a new ticket before connecting:

```
kinit -k -t [***KEYTAB PATH***] [***KERBEROS PRINCIPAL***]
```

Catalog property handling

The extractor scans catalog properties files and exports only non-sensitive configuration values. Sensitive keys are automatically removed, including:

- Passwords
- Tokens
- Encryption keys
- SSL credentials
- Cloud access credentials

If a catalog references a platform not supported by Cloudera Data Lineage, the corresponding objects appear as inferred objects in the lineage.

Supported connectors

Cloudera Data Lineage provides complete lineage only when Trino catalog properties use the required configuration formats listed in the following table.

Platform	Connector Name	Required Configuration
SQL Server	sqlserver	jdbc:sqlserver://*[***SERVER***]*:1433;databaseName=*[***DATABASE NAME***]*;encrypt=false;trustServerCertificate=true
Snowflake	snowflake	jdbc:snowflake://*[***SERVER***]*:443?db=*[***DATABASE NAME***]*
PostgreSQL	postgresql	jdbc:postgresql://*[***SERVER***]*:5432/*[***DATABASE NAME***]*

Platform	Connector Name	Required Configuration
MySQL	mysql	jdbc:mysql://*[***SERVER***]*:3306?useSSL=true&requireSSL=true&verifyServerCertificate=false
Hive	hive	hive.metastore.uri=thrift://*[***HOST***]*:9083
Google BigQuery	bigquery	bigquery.project-id=*[***YOUR GCP PROJECT ID***]*
Oracle	oracle	jdbc:oracle:thin:@*[***HOST***]*:*[***PORT***]*:*[***SID***]*
Redshift	redshift	jdbc:redshift://*[***HOST***]*:*[***PORT***]*/*[***DATABASE***]*
Iceberg (Hive)	iceberg	hive.metastore.uri=thrift://*[***HOST***]*:9083
MariaDB	mariadb	jdbc:mariadb://*[***HOST***]*:*[***PORT***]*
Kafka	kafka	kafka.table-names, kafka.nodes, kafka.config.resources
MongoDB	mongodb	mongodb.connection-url=mongodb://*[***USER***]*:*[***PASS***]*@*[***HOST***]*:*[***PORT***]*/
Delta Lake (Thrift)	delta_lake	hive.metastore.uri=thrift://*[***HOST***]*:9083
Delta Lake (Glue)	delta_lake	hive.metastore=glue
Druid	druid	jdbc:avatica:remote:url=http://*[***BROKER***]*:*[***PORT***]*/druid/v2/sql/avatica/

Metadata extracted

The Octopai agent queries Trino system metadata and exports the following objects:

Catalogs

From system.metadata.catalogs:

- Catalog name (the system catalog is excluded)

Catalog properties

The extractor scans catalog properties files and exports non-sensitive properties only.

Tables

From system.jdbc.tables:

- Catalog
- Schema
- Table name
- Table type
- Table comments

The following system schemas are excluded: information_schema, jdbc, runtime, metadata.

Views

From *[***CATALOG***]*.information_schema.views:

- Catalog
- Schema
- View name
- View definition

Columns

From *[***CATALOG***]*.information_schema.columns:

- Catalog
- Schema

- Table
- Column name
- Ordinal position
- Default value
- Nullable flag
- Data type

Results are ordered by schema, table, and ordinal position.

Comments

From system.metadata.table_comments:

- Catalog
- Schema
- Table
- Comment text

Empty comments and system schemas are excluded.

Extraction output

The connector generates XML output files in the extractor temporary folder. Files are created for each extraction type:

- Catalogs
- Tables (per catalog)
- Views (per catalog)
- Columns (per catalog)
- Comments (per catalog)
- Catalog properties

Cloudera Data Lineage uses these files to build lineage and discovery maps.

How to connect to Trino with DBeaver

Learn how to connect to Trino on Starburst using DBeaver.

Before you begin

Before you begin, ensure that you have the following:

- Access to your Starburst environment (Galaxy or Starburst Enterprise)
- Your Starburst username and password
- DBeaver installed

About this task

You can connect to a Trino environment hosted on Starburst (Galaxy or Enterprise) from DBeaver using standard username and password authentication.

Procedure

1. Open DBeaver.
2. Start a new Trino database connection.
   a) Click the plug icon (New Database Connection).
   b) In the search field, type Trino.
   c) Select Trino from the list.
   d) Click Next.

3. Configure the connection settings on the **Main** tab.

   Host

   Enter your full Starburst cluster URL.

   Examples:

   - Galaxy: *[***YOUR CLUSTER***]*.starburstdata.com
   - Enterprise: your Starburst Enterprise (SEP) host

   Port

   Use 443 for Starburst Galaxy. Starburst Enterprise typically uses 8443 for HTTPS.

   Database / Schema

   Leave this field empty for the initial connection. You can define the catalog and schema later.

   Username

   Enter your Starburst username.

   Password

   Enter your Starburst password.
4. Open the **Driver Properties** tab.
5. Set SSL to true to enable TLS for the connection to the cluster.
6. Click Test Connection to validate connectivity.
7. If the test succeeds, click Finish.

Results

You can query Trino catalogs and schemas from DBeaver using your Starburst credentials.

Vertica

Learn how to set up Vertica metadata sources with Cloudera Data Lineage (formerly Octopai).



Note: Version supported: up to v12.0

Tool Permissions Prerequisites



Warning: Missing permissions could end up in broken lineages.

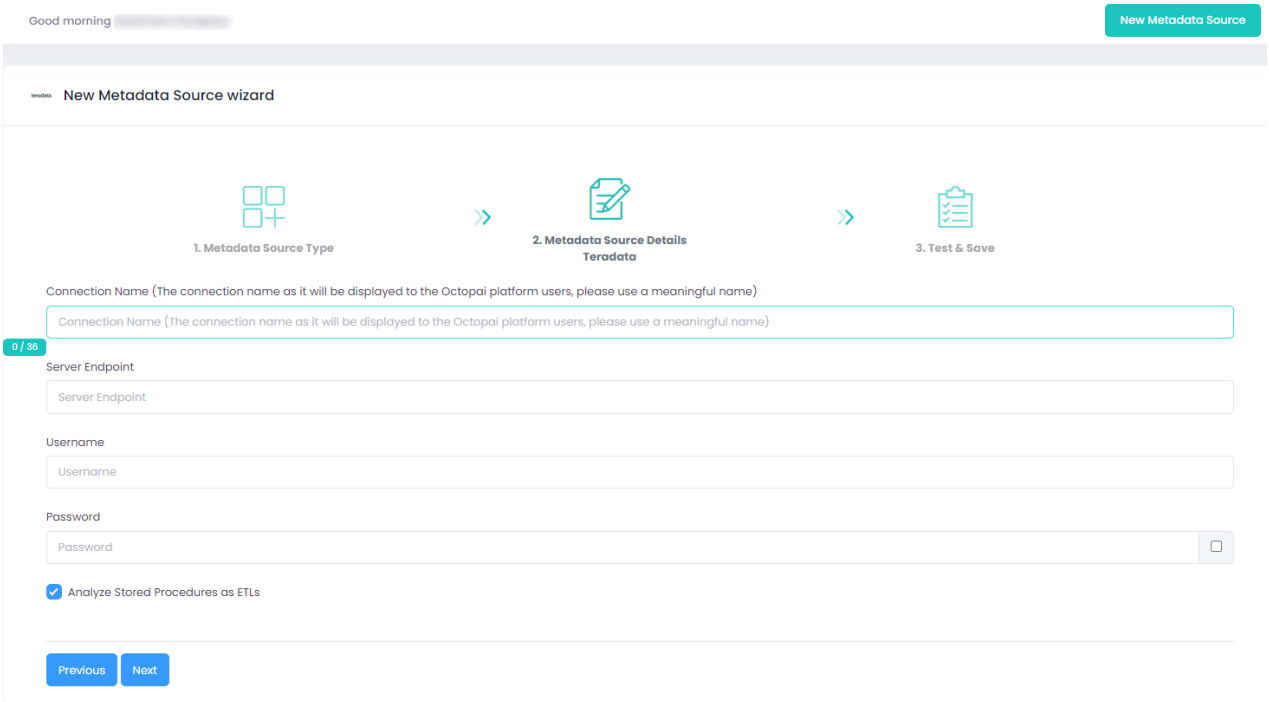
Ensure the following prerequisites are met:

- Open Server Port for each Vertica Database Connection
- Admin Vertica user for each connection with grant permission for the following dictionary tables:
  - PROJECTIONS
  - TEXT_INDICES
  - USER_FUNCTIONS
  - USER_PROCEDURES
  - USER_TRANSFORMS
  - VIEW_COLUMNS
  - VIEWS
  - VIEW_TABLES

- ALL_TABLES
- DATABASES
- COLUMNS
- COMMENTS
- HCATALOG_TABLES
- CONSTRAINT_COLUMNS
- HCATALOG_COLUMNS
- HCATALOG_SCHEMATA
- HCATALOG_TABLE_LIST
- PROJECTION_COLUMNS

Setting up Vertica Metadata Source

Metadata Sources are set on the Octopai Client.



Verify extraction results

After a successful run, confirm the metadata archive in the target folder. See Verify extracted metadata files.

Troubleshoot

If an error occurs during the extraction, see Troubleshoot connector extraction failures.

ETL and data integration

Cloudera Data Lineage extracts metadata and lineage from ETL and data integration tools, including Azure Data Factory, dbt Cloud, IBM DataStage, Informatica, Oracle ODI, and SSIS.

Azure Data Factory (ADF)

Overview of the Cloudera Data Lineage (formerly Octopai) Azure Data Factory extraction workflow.



Note: Version supported: Version 2

Tool Permissions Prerequisites

Before proceeding, verify that you meet the prerequisites for tool permissions in Azure Data Factory.

Warning: Missing permissions could end up in broken lineages.

- A dedicated application registered with the 'Reader' role assigned to the relevant Data Factories.
- Valid 'Client Secret' for authentication credentials.

How to set up the permissions

Ensure you have the necessary permissions to set up and manage Azure Data Factory.

Step 1:

Application Setup: Quickstart: Register an app in the Microsoft identity platform

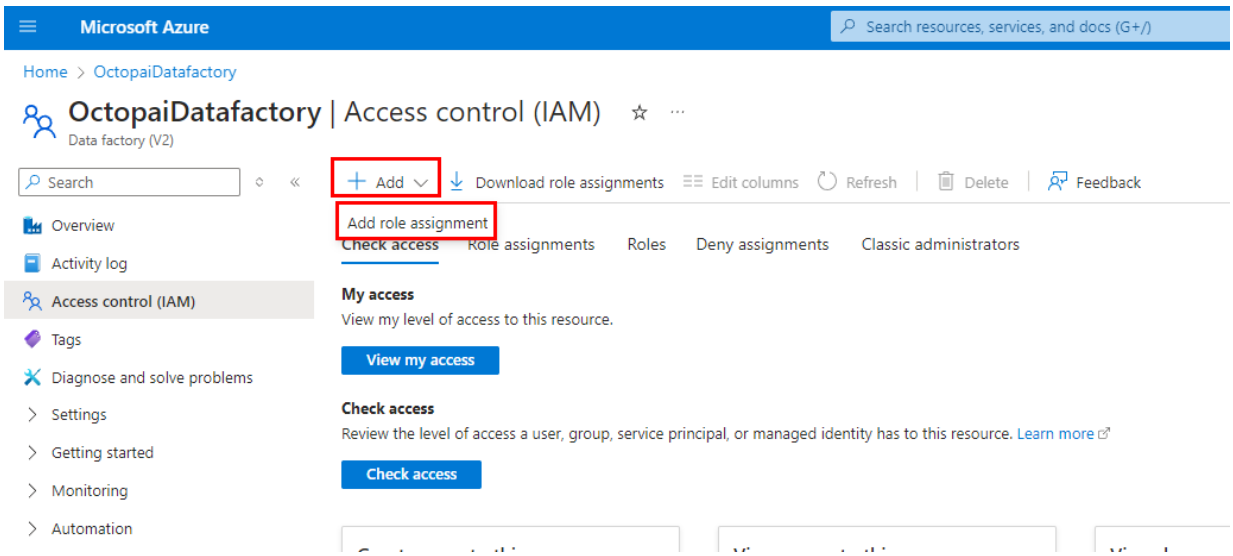
Guidelines for application setup:

- Select ‘Accounts in this organizational directory only’ when creating a new application (registration), under ‘Who can use this application or access this API?’.
- On the same page, leave 'Redirect URI empty'.
- Credentials - On the Application page, under Manage > Certificates & Secrets, use the Client Secret option for Credentials. Copy it immediately, as it won't be fully visible afterward.

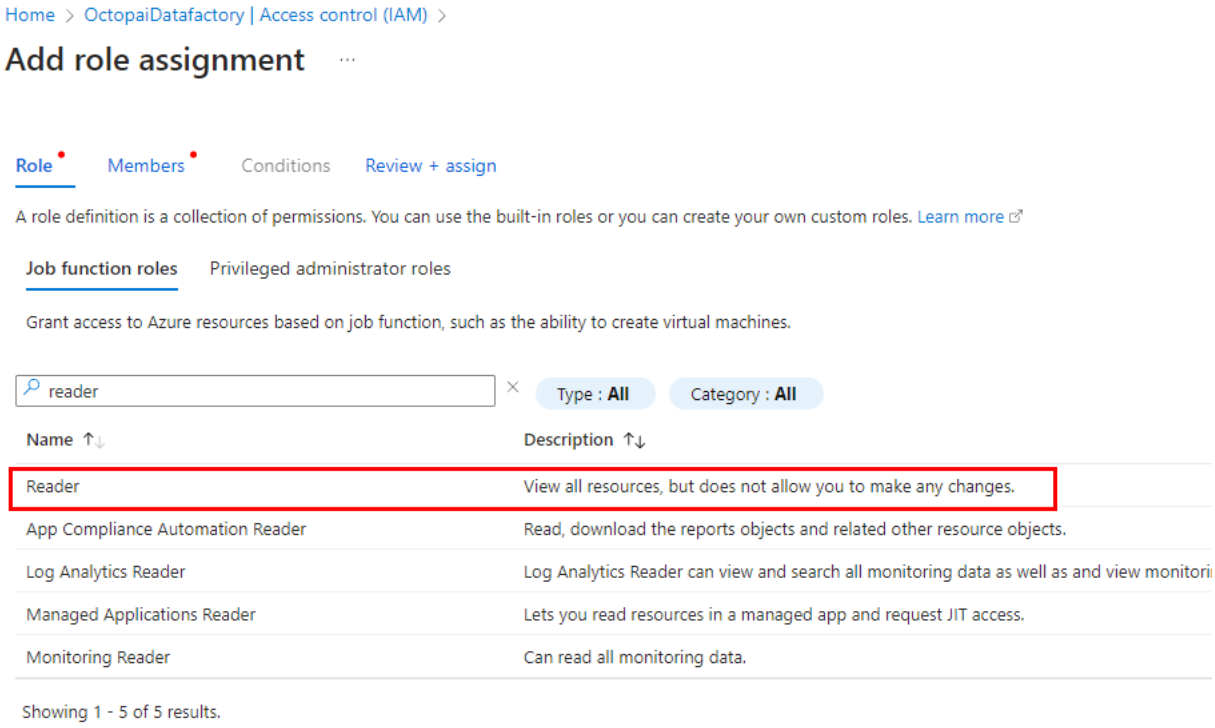
Step 2:

Assign the dedicated application a 'Reader' role to the relevant Factory/ies by following the below steps:

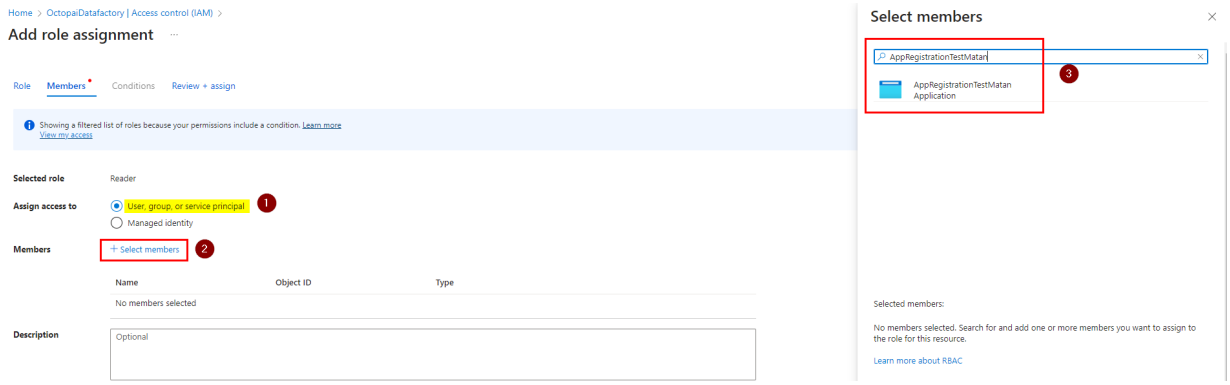
1. Under your DataFactory, go to the 'Access control (IAM)' tab and click on 'ADD > Add role assignment'.



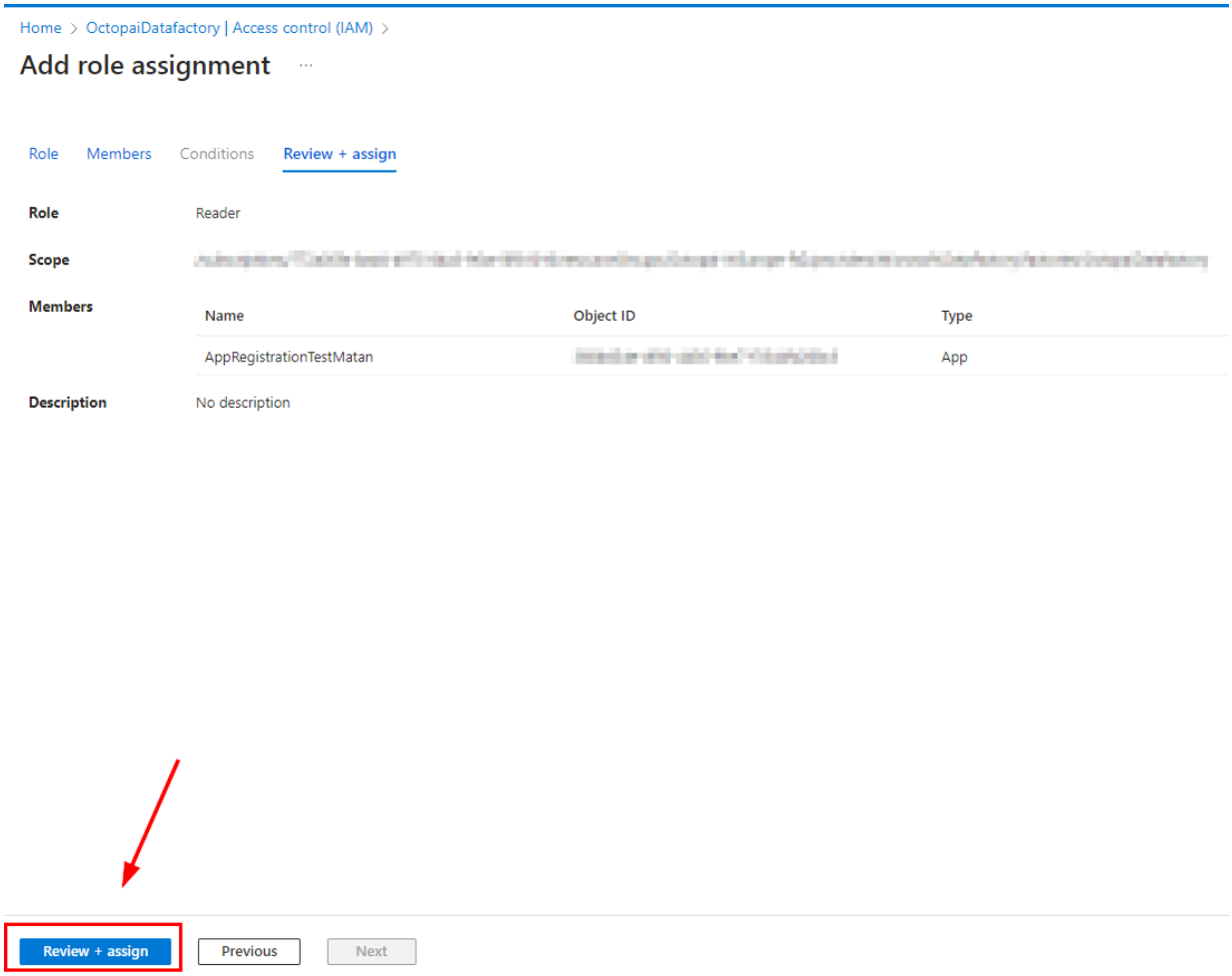
2. Look for the 'Reader' role and click it.



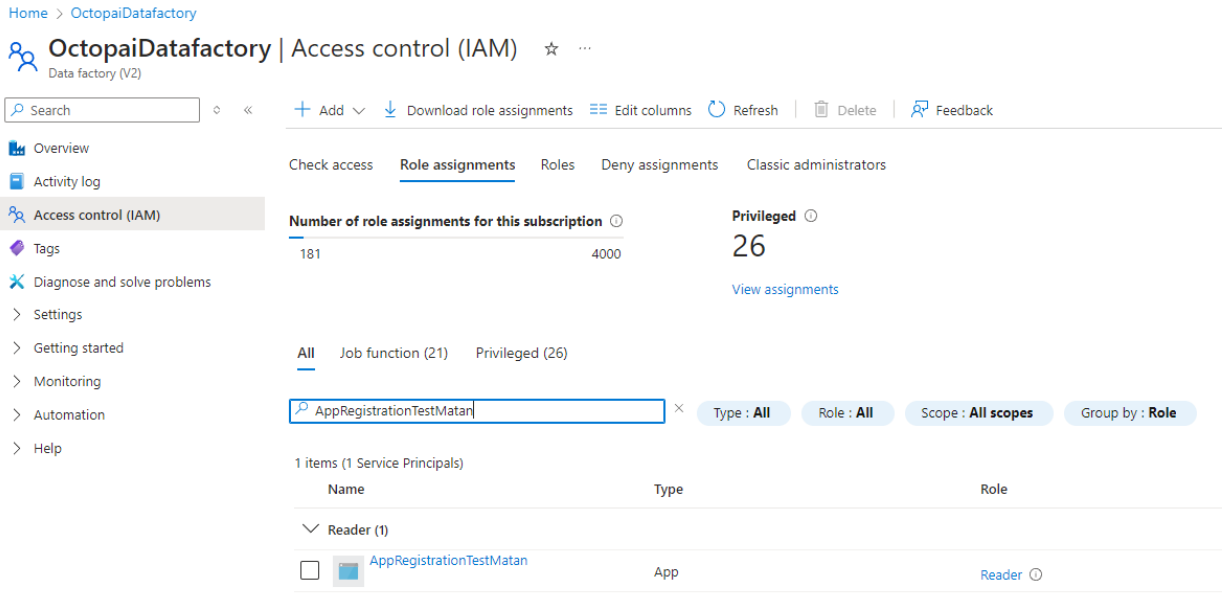
3. Under the 'Members' tab, choose 'User, group, or service principal' and click on '+ Select members', then search for your application.



4. The last step will be to review your configuration and assign the role by clicking on 'Review + assign'.



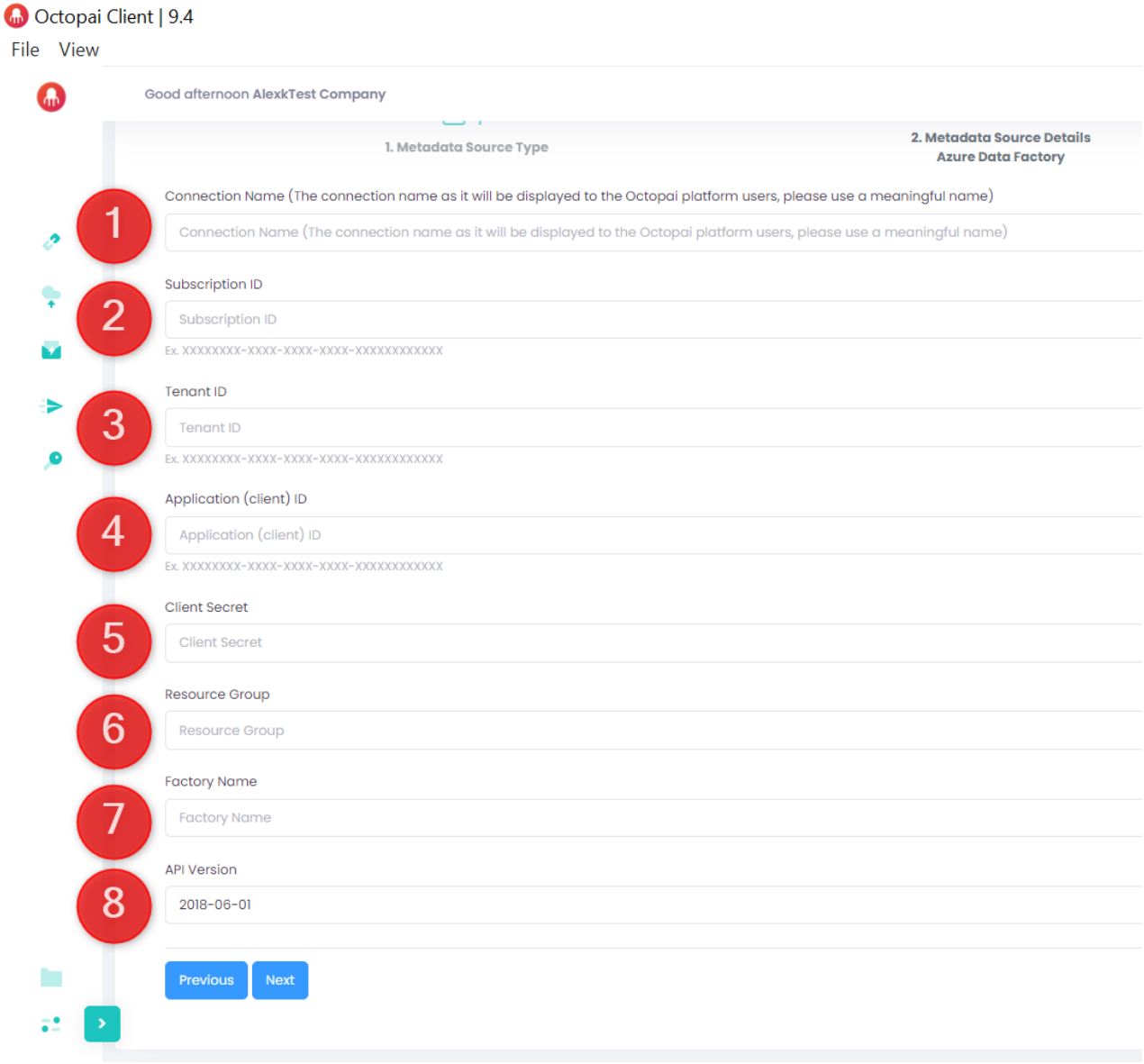
5. After completing the previous steps, go back to your DataFactory's 'Access control (IAM)' tab > 'Role assignments'. Your application should be there.



Setting up ADF Metadata Source

Follow these instructions to configure the metadata source in Azure Data Factory.

Metadata Sources are set on the Octopai Client:



Legend:

- Connection Name: Give a meaningful name, as it will be displayed to the Cloudera Data Lineage platform users.
- Subscription ID: Found in the 'Subscriptions' section of the Azure portal.
- Tenant ID: Available in the 'App registrations' section under the application you created.
- Application (Client) ID: Available in the 'App registrations' section under the application you created.
- Client Secret: Generated in 'App registrations > Certificates & secrets'.
- Resource Group: Found in the 'Resource groups' section where you created or assigned resources for your Data Factory.
- Factory Name: Listed in the 'Data Factory' section under your specific factory instance.
- API Version: Usually specified in the Azure documentation or the REST API version section related to Data Factory.

After completing all the mandatory fields, click on 'Next' > 'Finish' > and 'Run' to extract the metadata from your source.

Configure dbt Cloud Connector for Cloudera Data Lineage

Learn how to configure dbt Cloud for Cloudera Data Lineage (formerly Octopai).

Before you begin

Before you connect dbt Cloud to Cloudera Data Lineage, verify that the dbt Cloud environment, permissions, and credentials are prepared. The connector uses the dbt Cloud API and a service token to retrieve execution metadata and compiled SQL for lineage extraction.

dbt Cloud plan requirements

The connector requires API access.

You must have one of the following:

- A paid dbt Cloud plan (Starter or Enterprise).
- An active trial of a paid plan.

Note: The free developer tier does not provide API access and cannot be used because service tokens cannot be created.

Git repository requirement

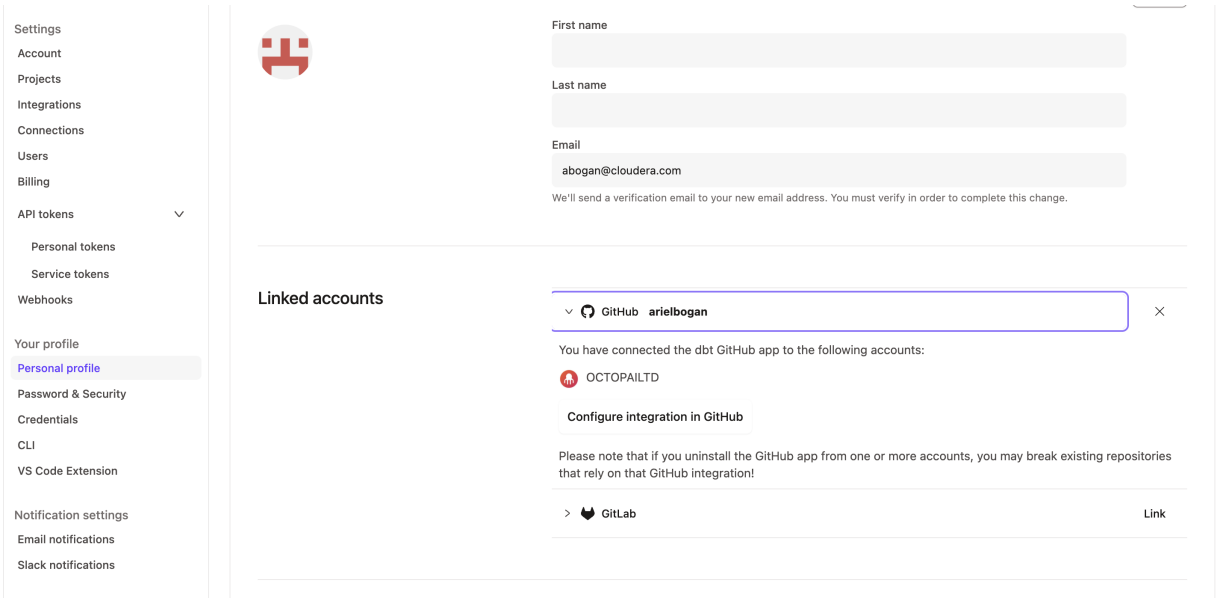
dbt Cloud requires a connected Git repository (for example, GitHub or GitLab). The connector depends on project runs generated from a version-controlled dbt project.

Procedure

1. Connect Git to dbt Cloud (if not already connected)
   a) Log in to dbt Cloud.
   b) Select the user icon (bottom left).
   c) Open Personal Profile.
   d) Connect your Git provider.

   If Git is already connected, continue to the next step.

Figure 23: Connect a Git provider in dbt Cloud



2. Create a service token

   Cloudera Data Lineage authenticates to dbt Cloud using a service token.

   Note: You must have a dbt Cloud user account with **Account Admin** permissions to create the service token.

   a) In dbt Cloud select the user icon.
   b) Select Personal Profile.
   c) Navigate to API Tokens.
   d) Select Service Tokens.
   e) Click Create Service Token.
   f) Provide a name.

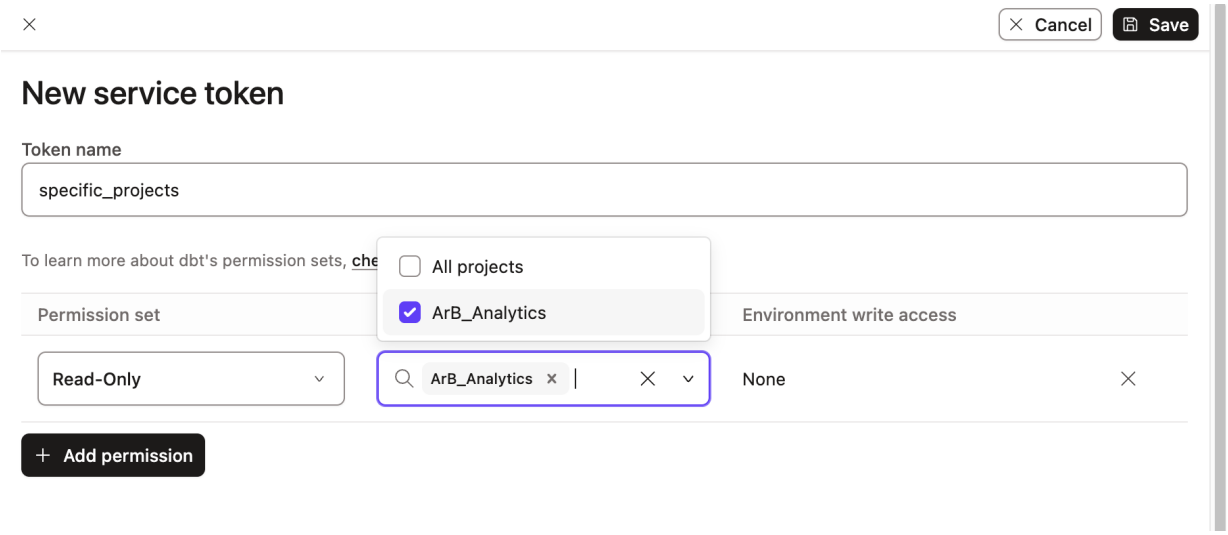
      Example: dbt_extractor

   g) Assign permissions.

      Select Read-Only.

   h) (Optional) Scope the token to specific project(s).

Figure 24: Create a service token in dbt Cloud

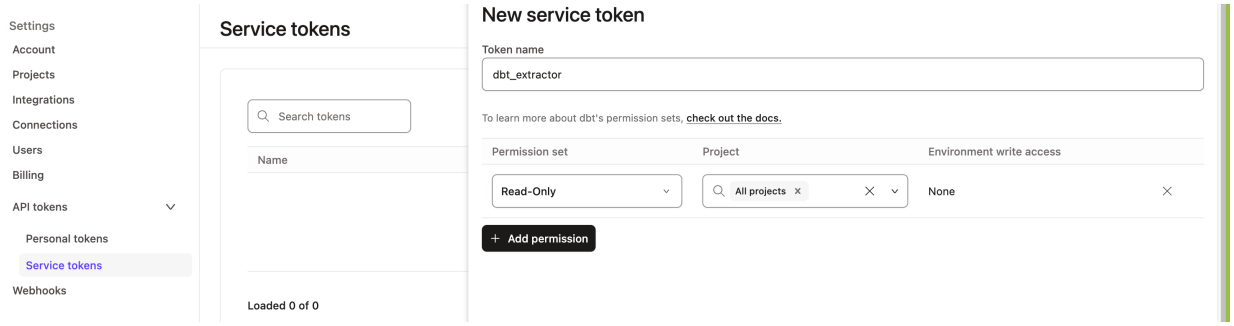


After creation:

- Copy the token immediately.
- Store it securely. You need this token to configure the dbt Cloud connection in Cloudera Data Lineage.

Important: The token is shown only once and cannot be viewed again.

Figure 25: Copy the generated service token



The integration must use a read-only service token scoped to the relevant project(s). By default, the connector discovers all dbt Cloud accounts accessible by the provided token.

3. Capture the dbt Cloud base URL

   Copy the full base URL of your dbt Cloud environment from the browser address bar.

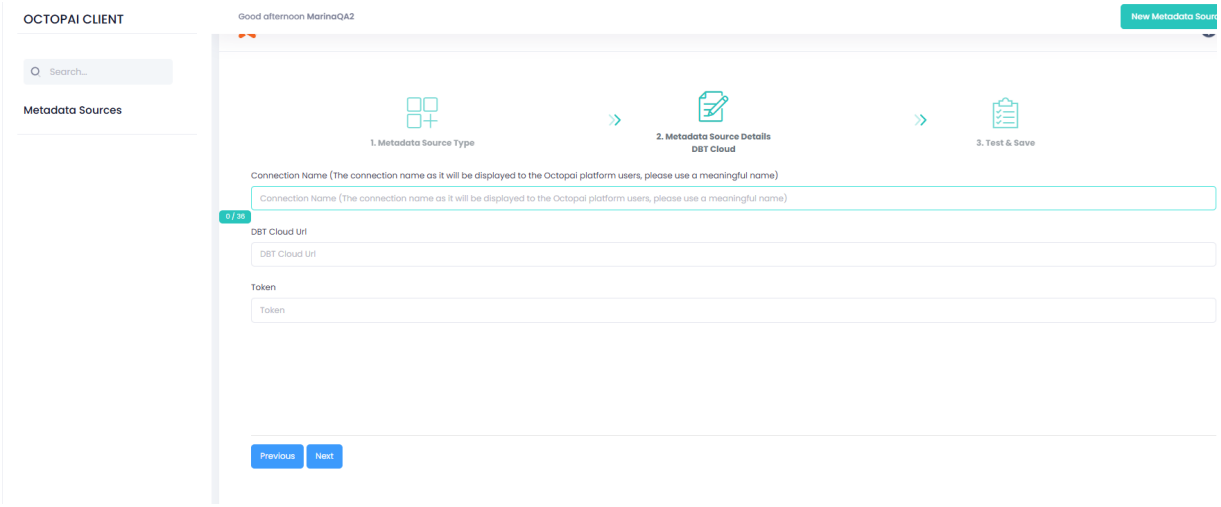
   Example: https://*[***COMPANY NAME***]*.us1.dbt.com

   You need this value when you configure the connection in Cloudera Data Lineage.

   The following parameters are required for the setup:

Parameter	Description
Connection Name	Logical name displayed in the platform
dbt Cloud Base URL	Full dbt Cloud environment URL
Service Token (Read-Only)	Token created in Step 2

Figure 26: Required setup information



What to do next

Extraction scope

The connector extracts metadata related to dbt execution activity. The following objects are discovered:

- Jobs
- Environments
- Models

Job-level SQL retrieval

For each job, Cloudera Data Lineage retrieves the SQL executed during the most recent successful run. Note the following behavior:

- The original model SQL source file is not used.
- Lineage is based on the compiled SQL generated by dbt.

Lineage parsing scope

The connector parses SQL-based models only. Python-based dbt transformations do not generate lineage. Lineage parsing is limited to database engines supported by Cloudera Data Lineage.

Limitations:

Expression representation

Expressions defined inside dbt models are not represented as expressions in the internal lineage visualization.

Python models

Python dbt models are not supported currently:

- Python models are not parsed.
- Column-level lineage is not generated for Python transformations.

Supported platforms

The connector supports dbt lineage extraction when models run on the following databases:

- Snowflake
- PostgreSQL
- Google BigQuery
- Redshift
- Databricks (requires GSP license)
- Spark SQL
- Synapse (via dbvmssql)
- Teradata

Unsupported platforms

Lineage parsing is limited to RDBMS platforms supported by Cloudera Data Lineage. The following platforms are not supported:

- Starburst
- Microsoft Fabric
- Athena

IBM DataStage

IBM DataStage integration with Cloudera Data Lineage (formerly Octopai) enables users to export and manage DataStage projects and metadata efficiently. Each project must be exported as a separate file, with specific configurations for ISX or DSX methods, ensuring compatibility with the Octopai Client.



Note: Version Supported: up to v11.7

Tool Permissions Prerequisites

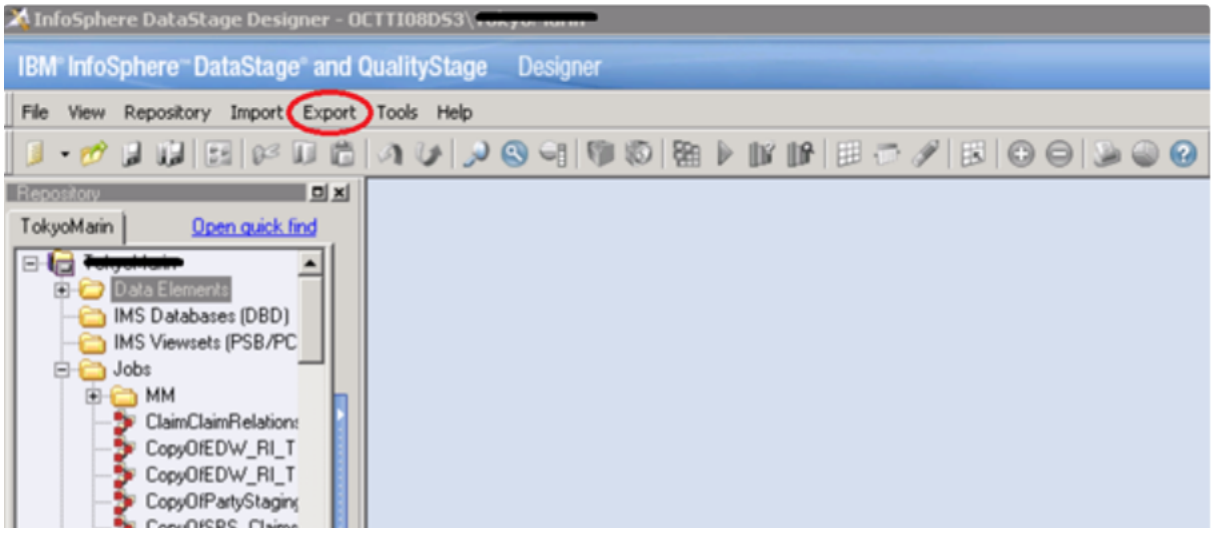
Warning: Missing permissions could end up in broken lineages.

IBM DataStage Client version 8.5 or later installed

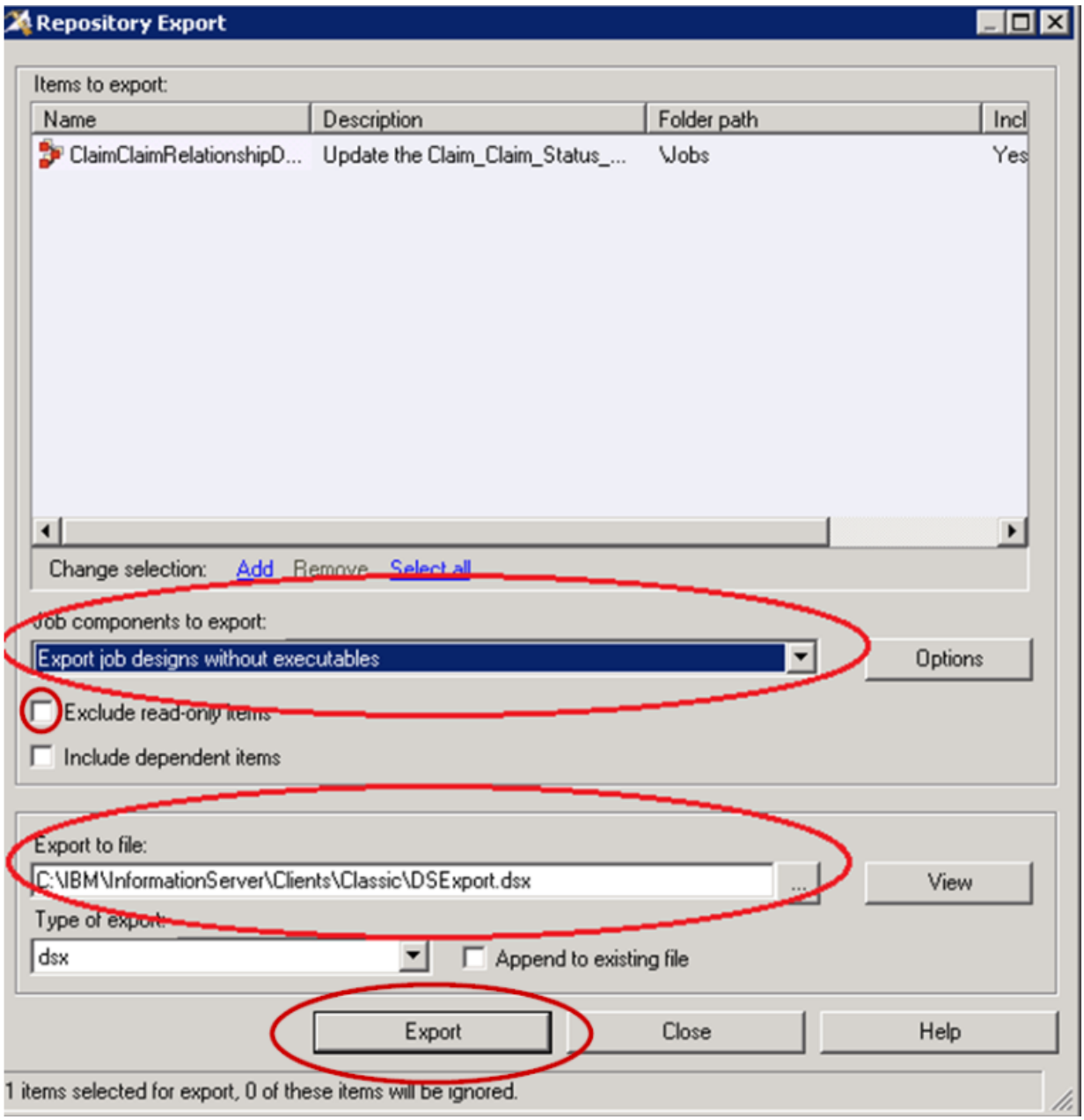
Open Server Port to DataStage Server Machine

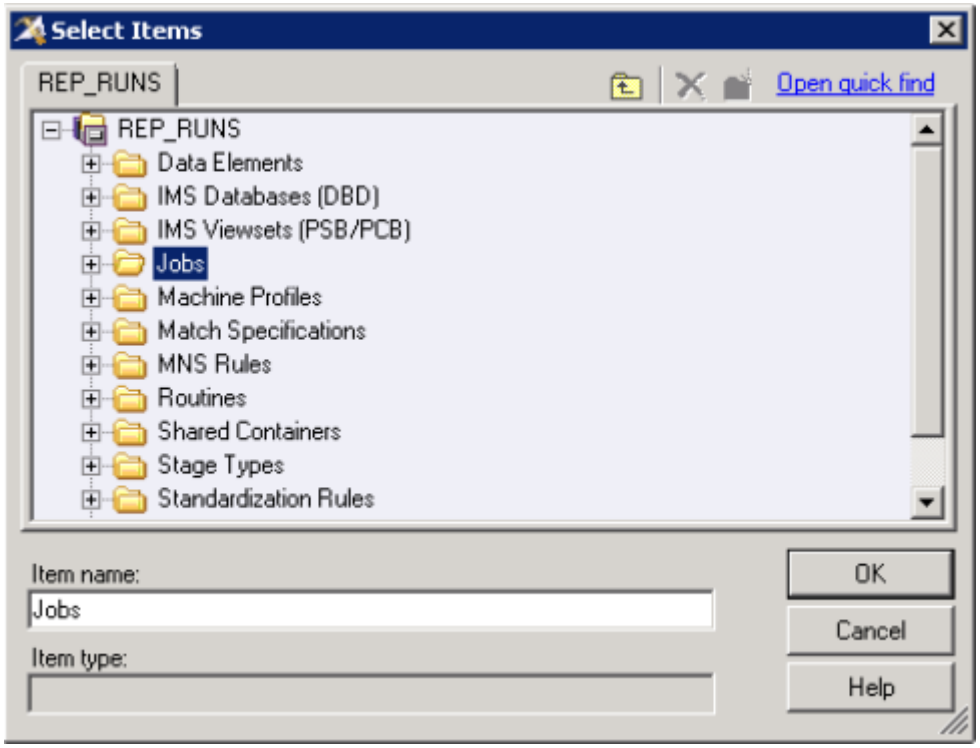
How to export IBM DataStage jobs

1. Open DataStage Designer (Client).
2. Choose project and connect.
3. Click **Export** -> **DataStage Components** ->

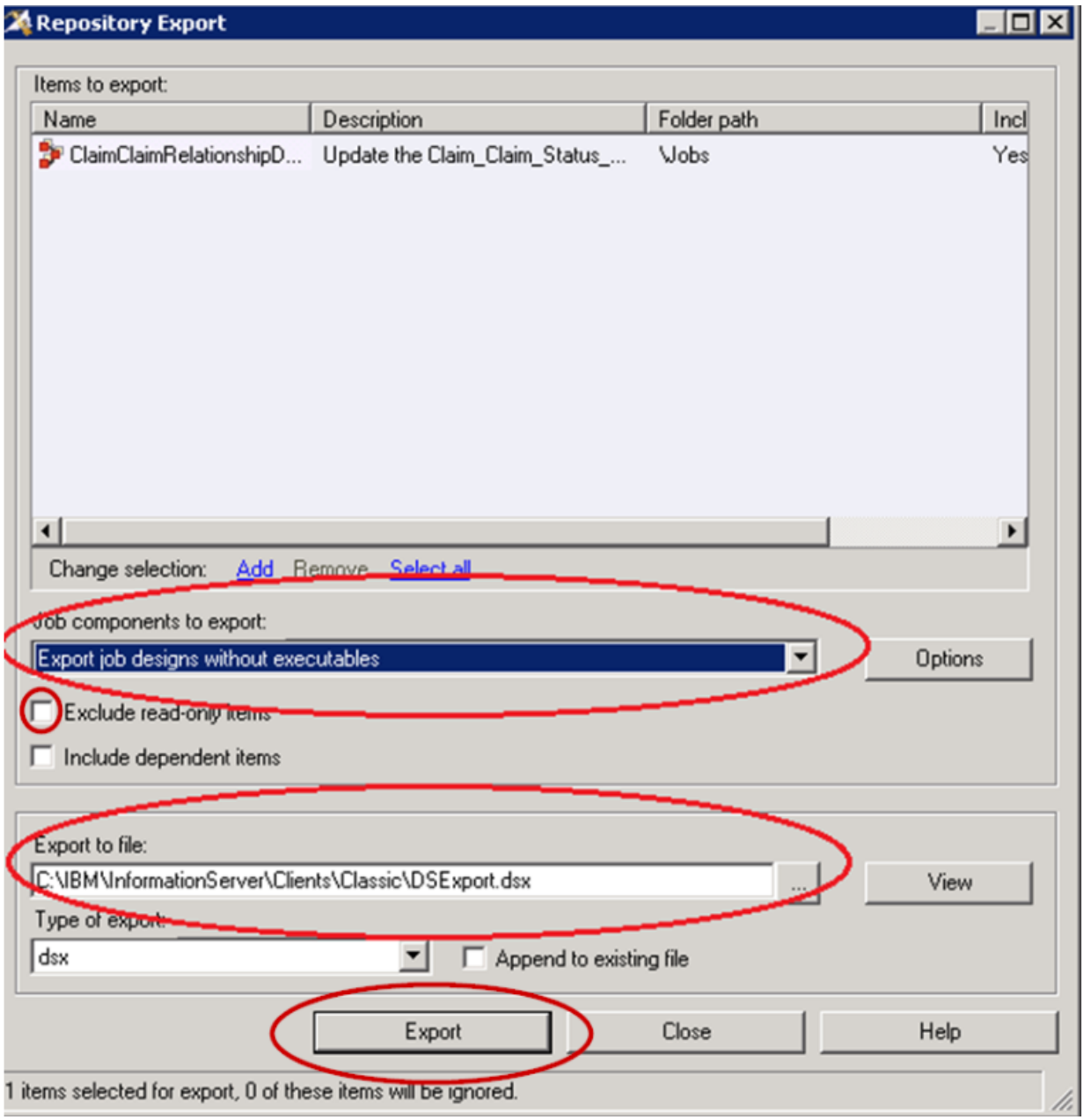


4. Click **Add** and choose folder jobs from the pop window: **Select Items**.


Repository Export
Items to export:
Name
Description
Folder path
Incl
ClaimClaimRelationshipD...
Update the Claim_Claim_Status_...
\Jobs
Yes
Change selection: Add Remove Select all
Job components to export:
Export job designs without executables
Options
Exclude read-only items
Include dependent items
Export to file:
C:\IBM\InformationServer\Clients\Classic\DSExport.dsx
View
Type of export:
dsx
Append to existing file
Export
Close
Help
1 items selected for export, 0 of these items will be ignored.



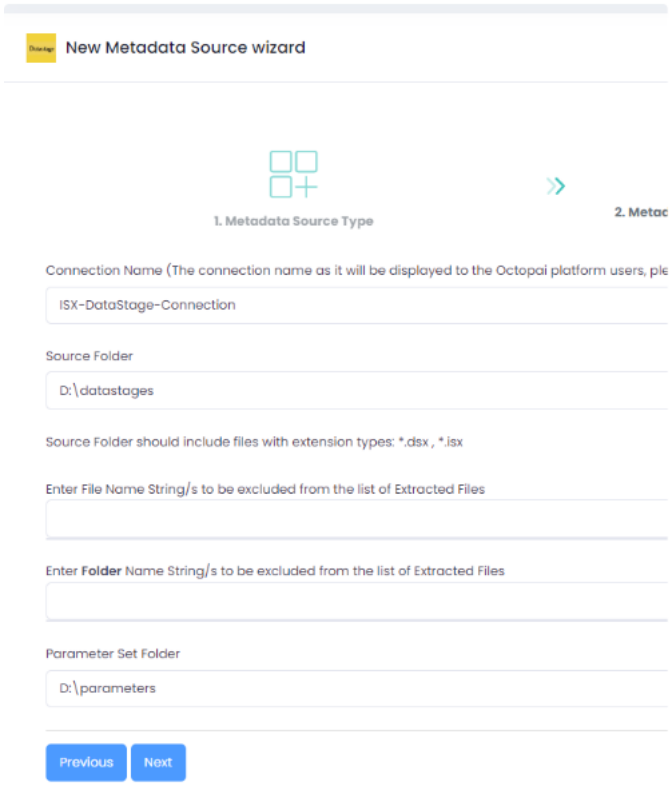
5. Configure destination file name and location:
   - At the drop-down list of the **Job components to export:**, choose **Export job designs without executables**.
   - Uncheck the **Exclude read-only items** option.
   - Choose the export file destination and type a name for the DSX file – **Export to file**.
6. Click the **Export** button.



7. The extracted file will appear in the destination defined folder. Use this file as the source for your Cloudera Data Lineage metadata DataStage connection in the Octopai Client.
8. Ensure appropriate permissions to the path to allow the Octopai Client to access the file with the user running it.

DataStage automation requires the customer to extract the DataStage project using the following methods.

ISX Method:



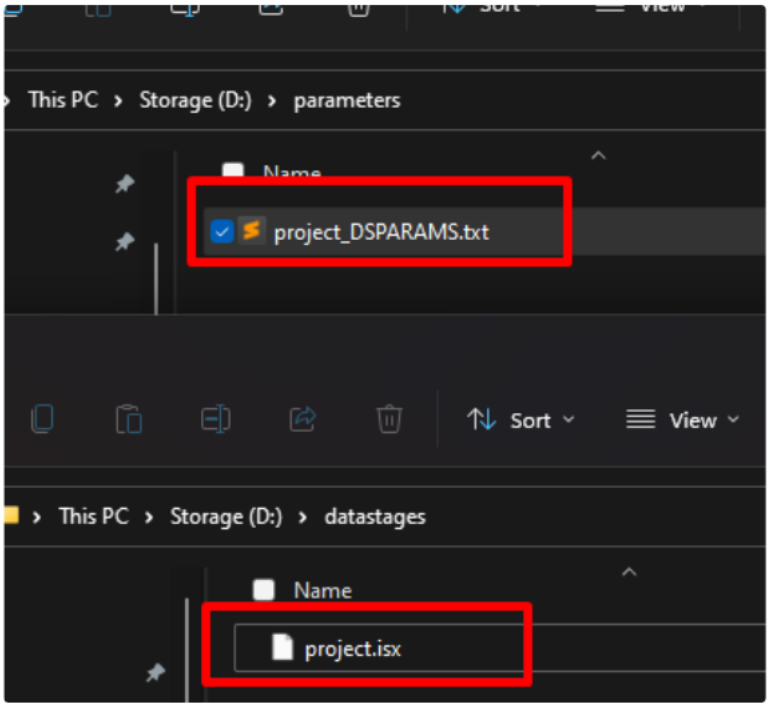
Each **project** for each **Server** needs to be in a separate zip file. Each **project** will be displayed as a different **connection name** in the Cloudera Data Lineage application:

Octopai Client - DataStage Module's fields

In the Source Folder, there needs to be only **one** ISX file that resembles **one** project.

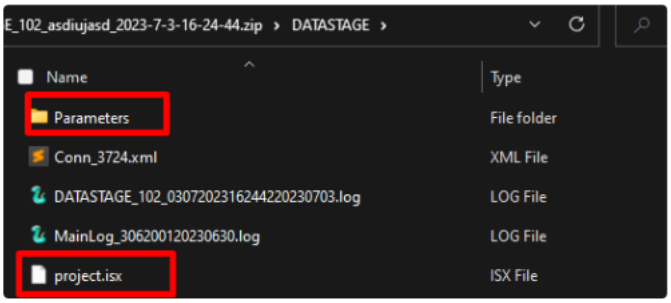
In the Parameter Set Folder, there needs to be the parameters file for that **one** project. (the format is TXT)

The Parameter Set file should follow this convention: "Isx_File_Name_DSPARAMS.txt". (For example, if we have a project named **project.isx** , the Parameter file name will be **project_DSPARAMS.txt**)

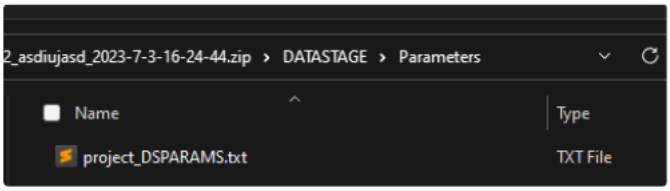


DataStage ISX file + Parameters for ISX Project File

The result from this module should look like the following:

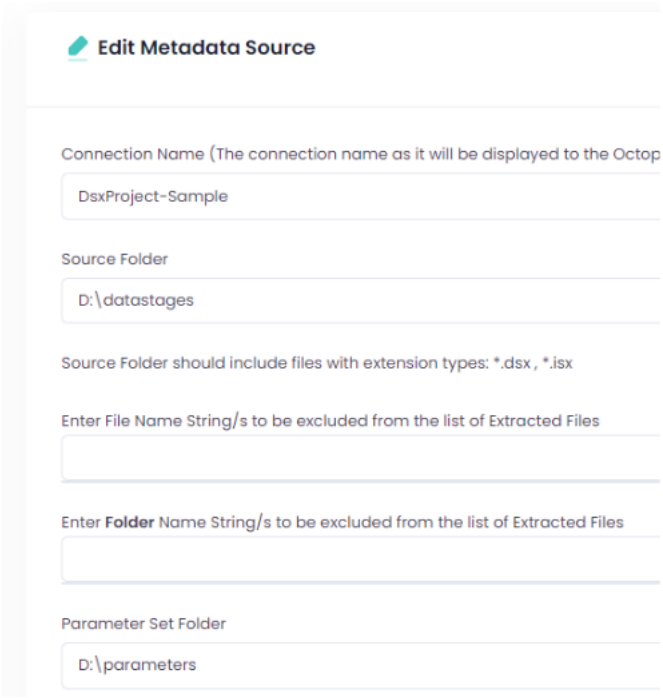


Inside the zip file



Inside the parameters folder

DSX Method:

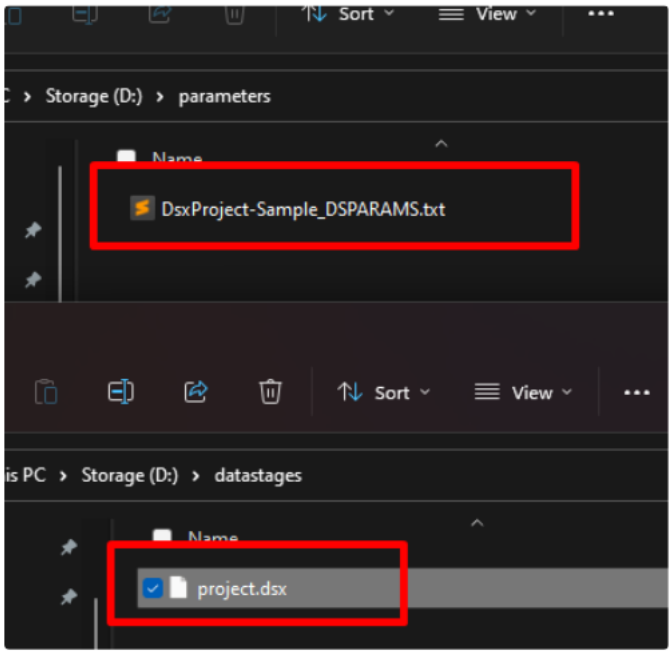


Octopai Client - DataStage Module's fields

Each **project** for each **Server** needs to be in a separate zip file. The **connection name** specified in the Octopai Client will be the **Name** of the parameter file:

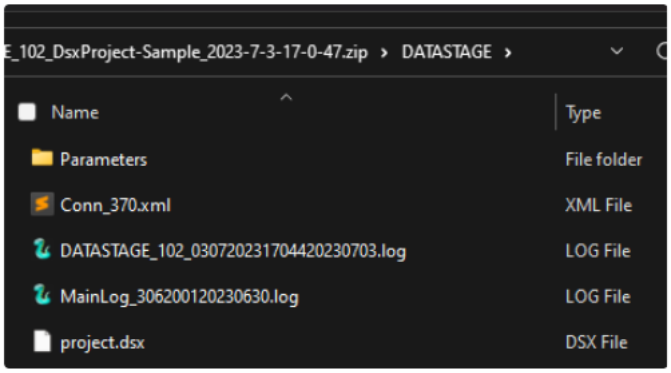
In the Source Folder, there needs to be only **one** DSX file that resembles **one** project.

In the Parameter Set Folder, there needs to be the parameters file for that **one** project. (the format is TXT) The Parameter Set file should follow this convention: "ConnectionName_DSPARAMS.txt". (For example, if the **Connection Name** given while creating this connection was **DsxProject-Sample** , the Parameter file name will be **DsxProject-Sample_DSPARAMS.txt**)

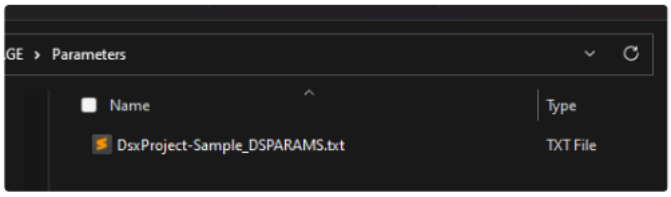


DataStage DSX file + Parameters for DSX Project File

The result from this module should look like the following:



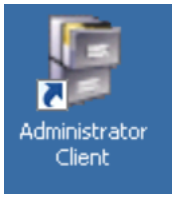
Inside the zip file



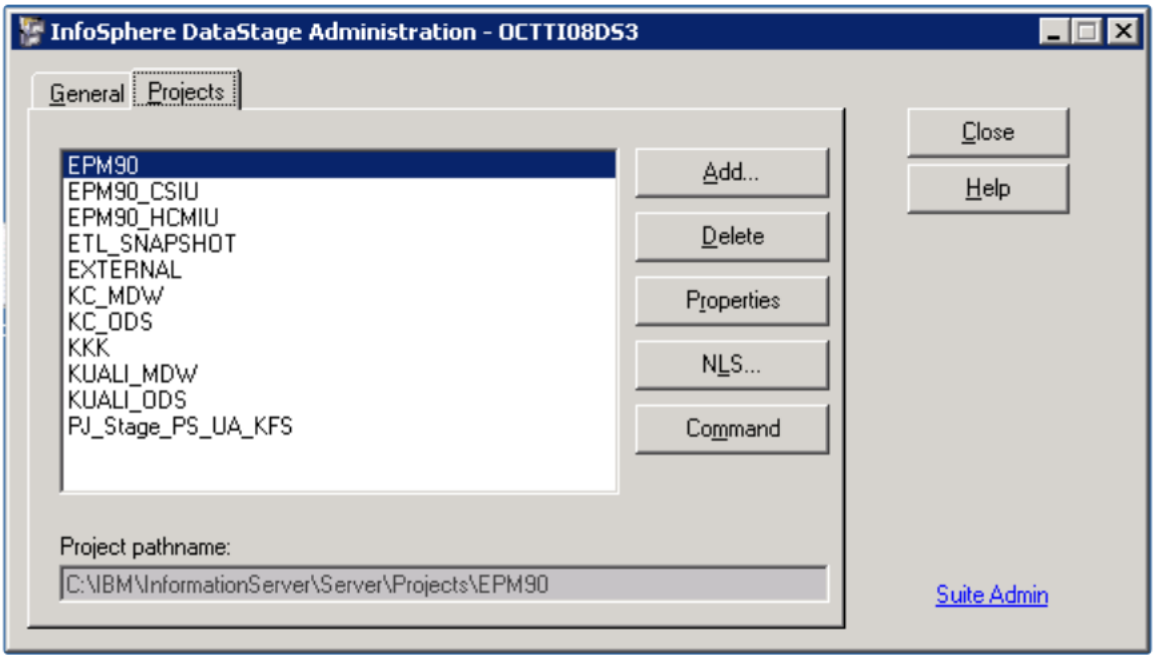
Inside the parameters folder

DataStage: Export of parameter files

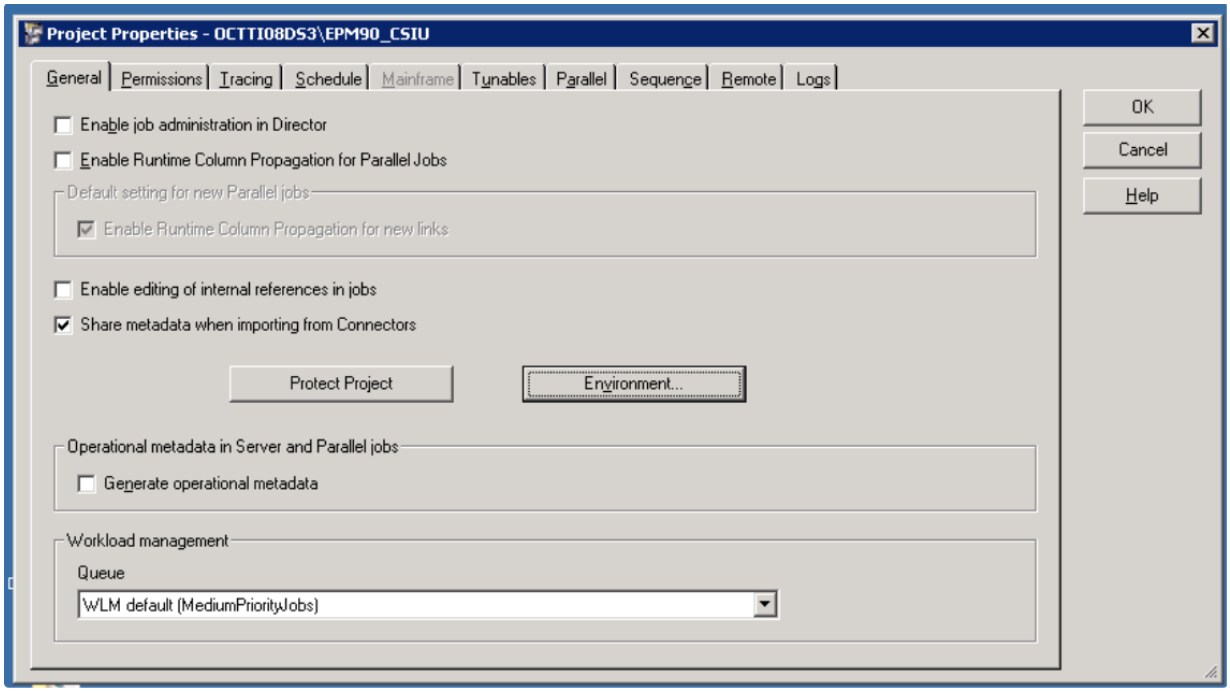
1. Go to the Administrator Client.



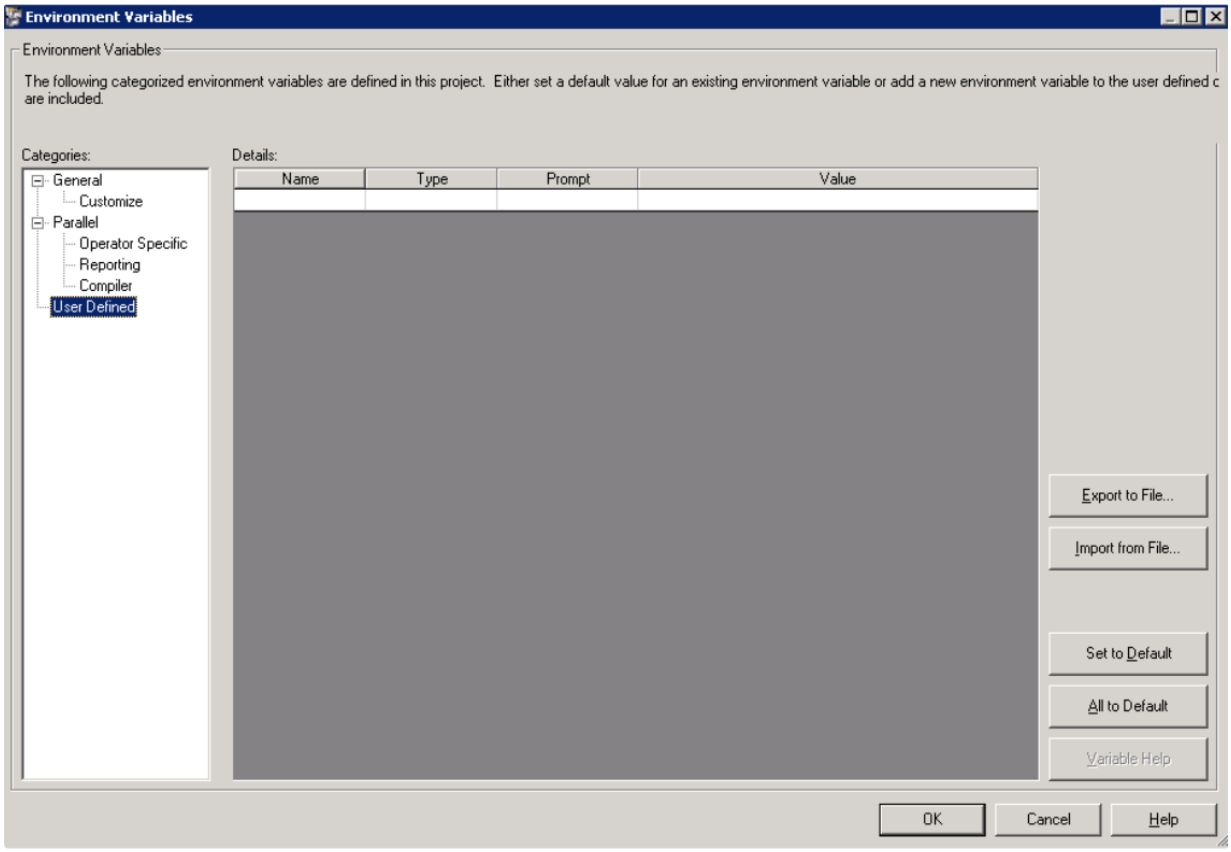
2. Choose the relevant Project and click on **Properties**.



3. Click on the **Environment** button.



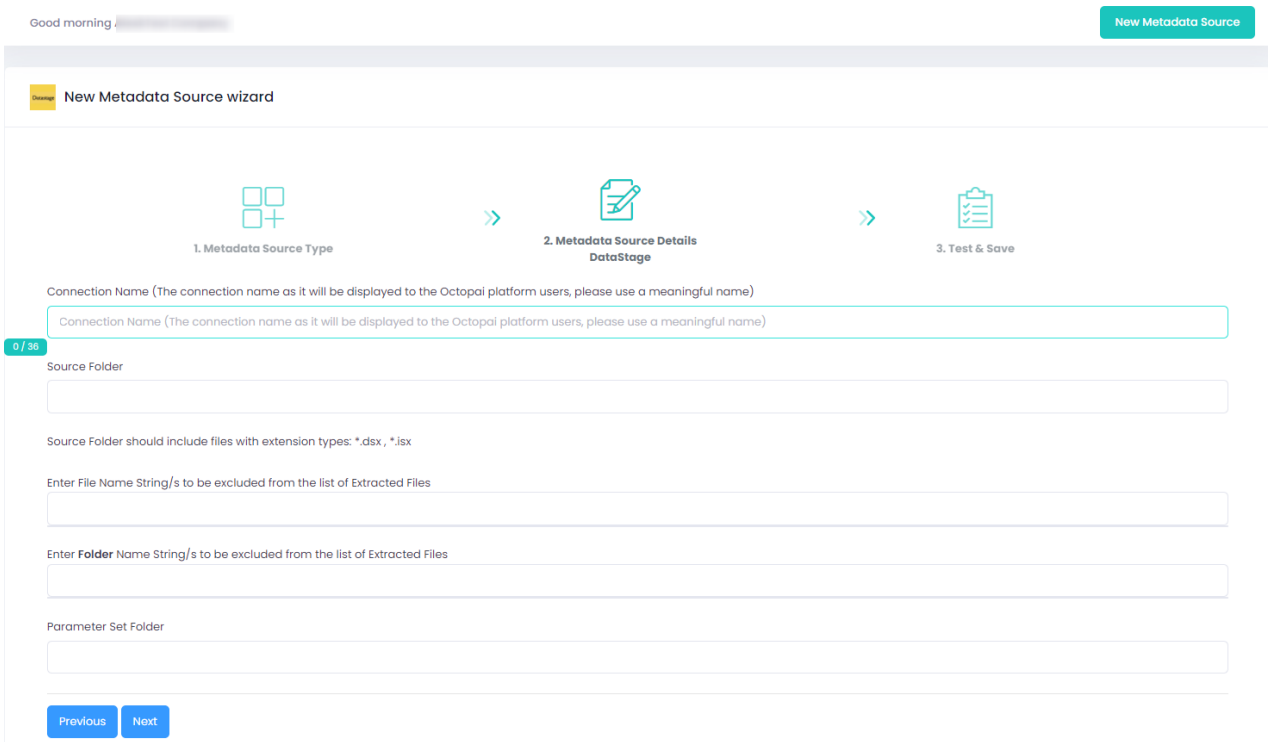
4. Go to **User Defined** and click on **Export to File**.



5. The new file name (.env) will be created - save it as **Name of Project _DSPARAMS.txt**.

Setting up DataStage Metadata Source

Metadata Sources are set on the Octopai Client



Verify extraction results

After a successful run, confirm the metadata archive in the target folder. See Verify extracted metadata files.

Troubleshoot

If an error occurs during the extraction, see Troubleshoot connector extraction failures.

IICS (Informatica Cloud)

Learn how to configure Informatica Cloud (IICS) permissions, set up metadata sources, verify files, access the Cloudera Data Lineage (formerly Octopai) target folder, and troubleshoot issues.

Before you begin

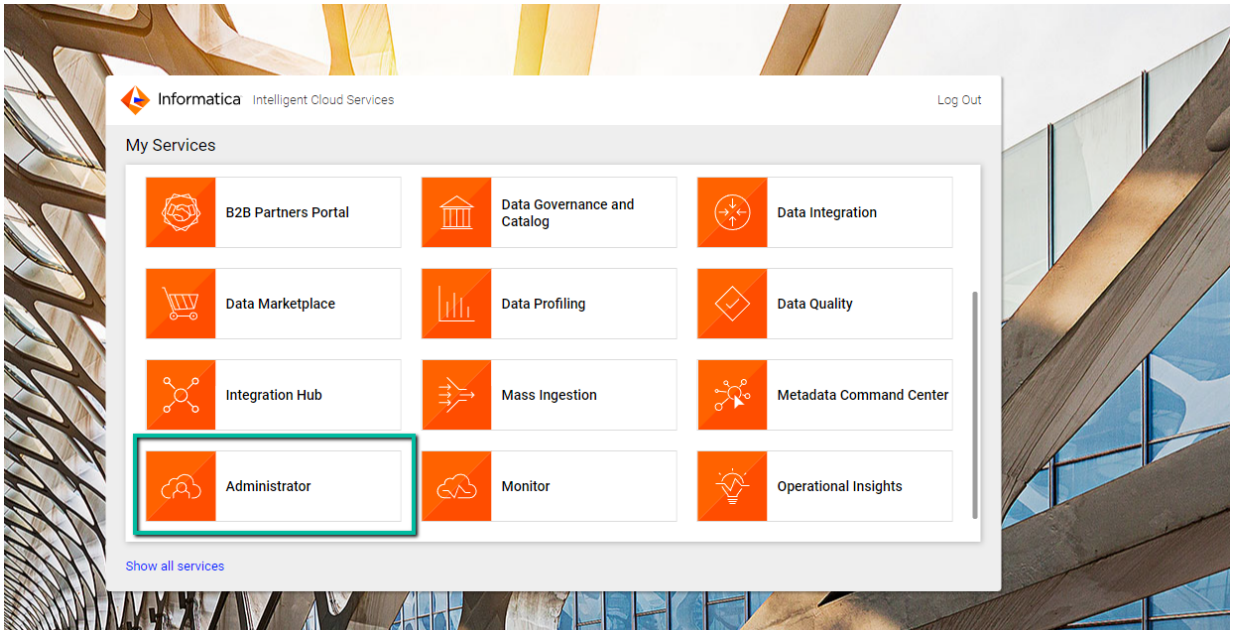
Warning: Missing permissions could end up in broken lineages.

- Add user role:

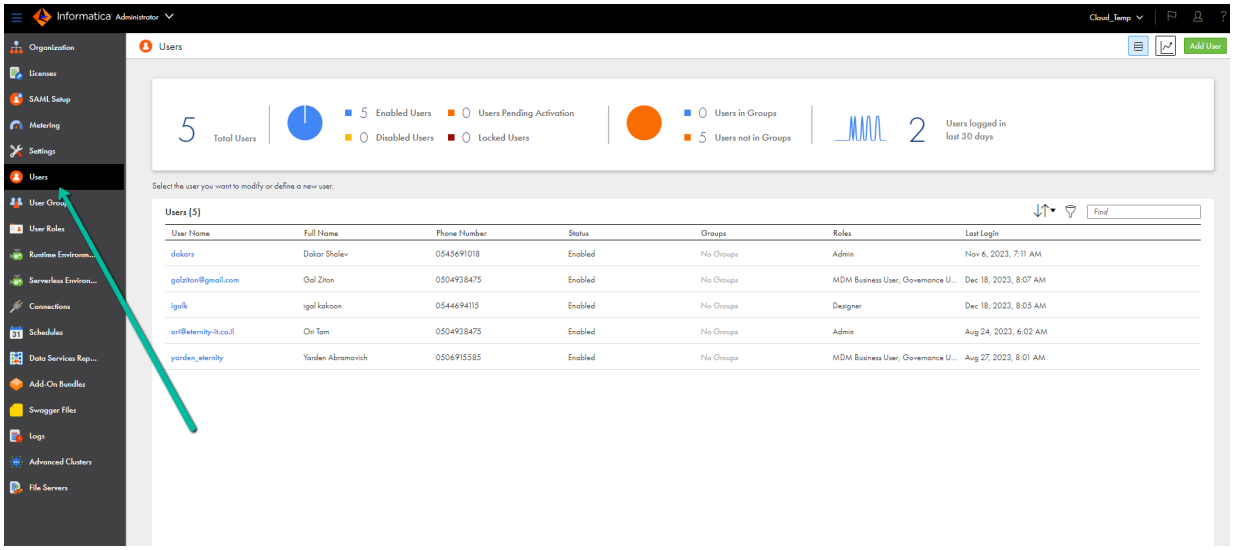
  Log in as admin or any other role that can edit the roles of other users

Procedure

1. Select the Administrator service.



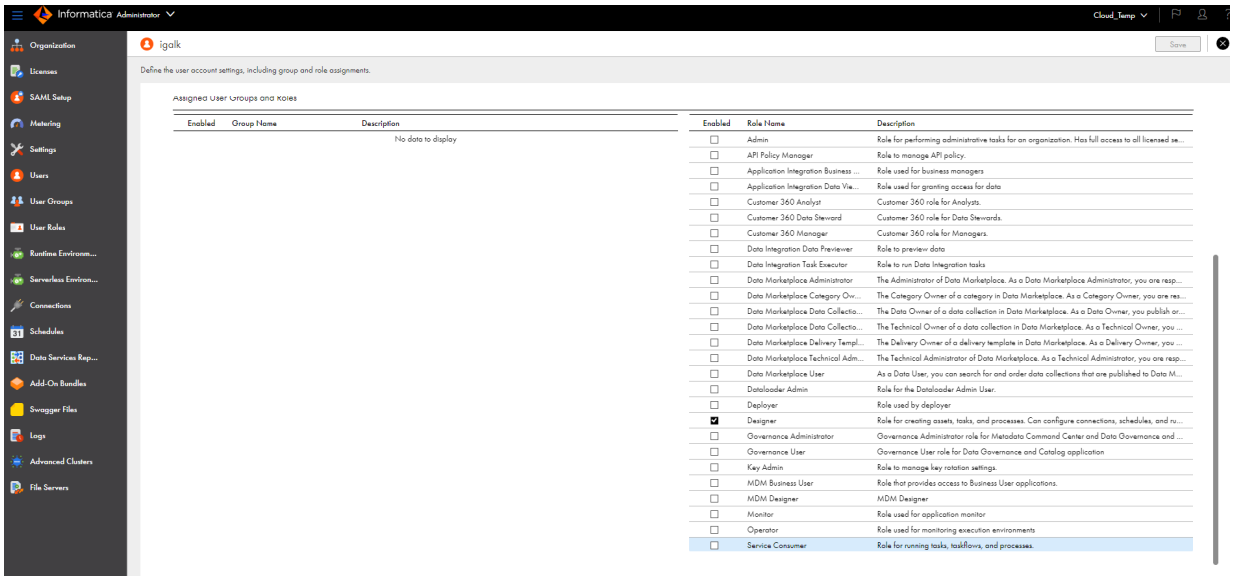
2. On the left panel, select Users.



3. Scroll down to the role list, and select the "Designer" role.

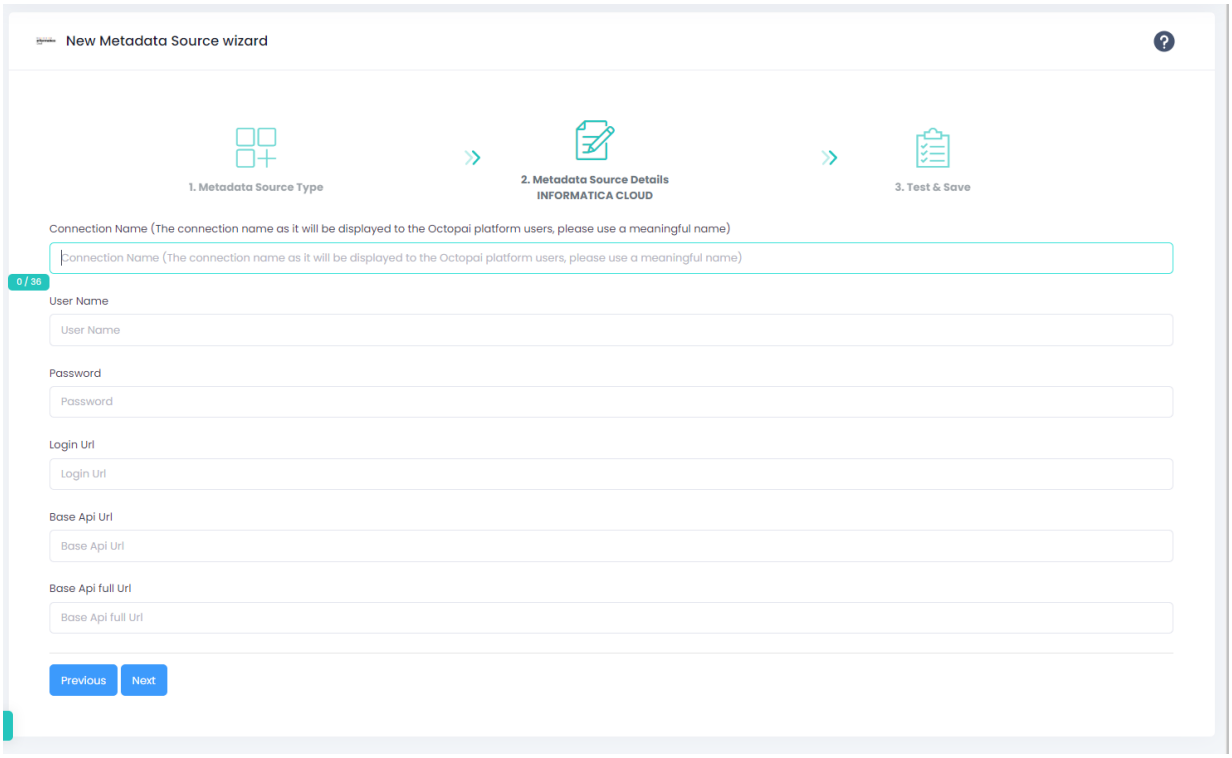
Select the user to whom you want to assign a role.

Click the Save button in the upper right screen.



4. Set up Informatica Metadata Source.

Metadata Sources are set on the Octopai Client.



baseApiUrl: The URL of Informatica Cloud. To derive the correct URL, take the application's URL and remove the first segment before the dot. For example, https://emw1.dm-em.informaticacloud.com/saas converts to https://dm-em.informaticacloud.com/saas.

baseApiUrlFull: For example, https://emw1.dm-em.informaticacloud.com/saas is the full URL of the app.

5. Verify extraction results.

   After a successful run, confirm the metadata archive in the target folder. See Verify extracted metadata files.

If an error occurs during the extraction, see Troubleshoot connector extraction failures.

Informatica Power Center

Learn how to configure and set up Informatica Power Center as a metadata source for Cloudera Data Lineage (formerly Octopai).

Note: Version Supported: up to v10.5

Tool Permissions Prerequisites

Note:

Warning: Missing permissions could end up in broken lineages.

1. Open Server Port for each Informatica repository database.
2. Existing/New (Oracle/SQL) user for each database with granted read permission for the following dictionary tables:
   - OPB_SESS_EXTNS
   - OPB_MMD_SESS_EXTNS
   - OPB_SESS_CNX_REFS

- OPB_CNX_ATTR
- OPB_CNX
- OPB_MMD_CNX
- OPB_EXPRESSION
- OPB_EXTN_ATTR
- OPB_MAPPING
- OPB_OBJECT_TYPE
- OPB_SWIDGET_ATTR
- OPB_SWIDGET_INST
- OPB_TABLE_GROUP
- OPB_TASK
- OPB_TASK_ATTR
- OPB_TASK_INST
- OPB_TASK_VAL_LIST
- OPB_WIDGET
- OPB_WIDGET_INST
- REP_ALL_MAPPINGS
- REP_ALL_MAPPLETS
- REP_ALL_SOURCE_FLDS
- REP_ALL_SOURCES
- REP_ALL_TARGET_FLDS
- REP_ALL_TARGETS
- REP_ALL_TASKS
- REP_COMPONENT
- REP_METADATA_EXTNS
- REP_SESS_WIDGET_CNXS
- REP_SUBJECT
- REP_TASK_ATTR
- REP_TASK_INST
- REP_WIDGET_ATTR
- REP_WIDGET_DEP
- REP_WIDGET_FIELD
- REP_WIDGET_INST
- REP_WORKFLOWS

If you would like to build a loop for this action, we offer to use this guide by Aron Kumar.

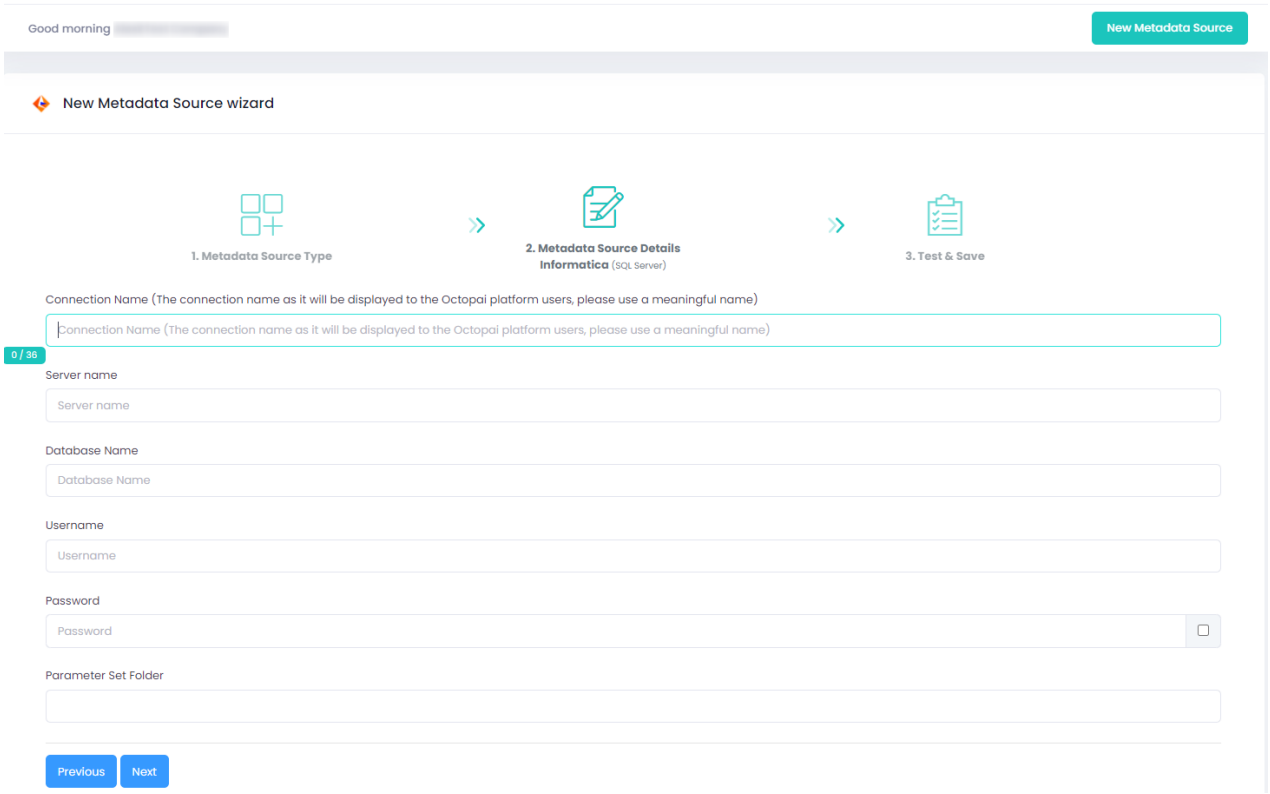
Setting up Informatica Metadata Source

Metadata Sources are set on the Octopai Client.

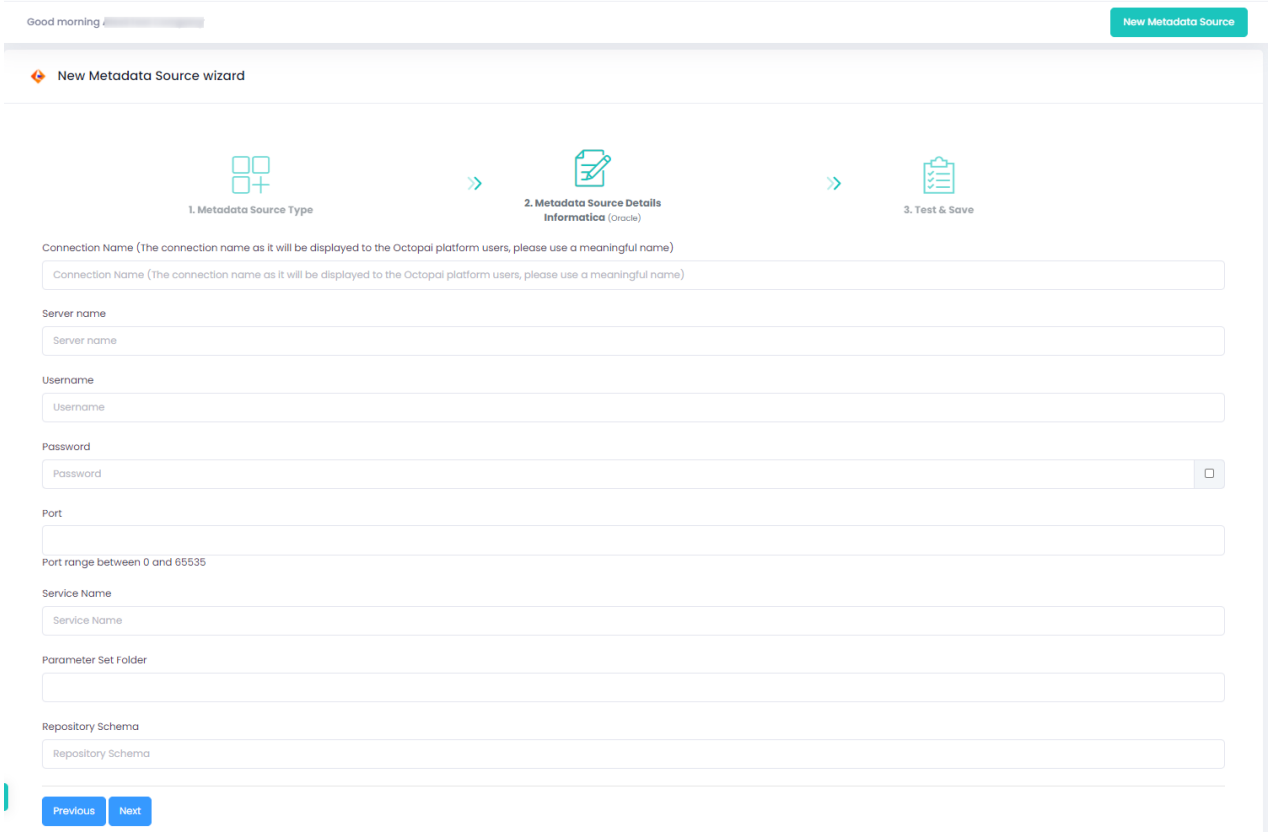


Note: If you're using parameters for Informatica workflows, it's recommended to organize them in a dedicated folder. To upload these parameters, simply navigate to the designated folder and select it from the "Parameter Set Folder" box.

SQL Server:


Good morning
New Metadata Source
New Metadata Source wizard
1. Metadata Source Type
2. Metadata Source Details
Informatica (SQL Server)
3. Test & Save
Connection Name (The connection name as it will be displayed to the Octopai platform users, please use a meaningful name)
Connection Name (The connection name as it will be displayed to the Octopai platform users, please use a meaningful name)
0 / 36
Server name
Server name
Database Name
Database Name
Username
Username
Password
Password
Parameter Set Folder
Previous
Next

Oracle:


Good morning
New Metadata Source
New Metadata Source wizard
1. Metadata Source Type
2. Metadata Source Details
Informatica (Oracle)
3. Test & Save
Connection Name (The connection name as it will be displayed to the Octopai platform users, please use a meaningful name)
Connection Name (The connection name as it will be displayed to the Octopai platform users, please use a meaningful name)
Server name
Server name
Username
Username
Password
Password
Port
Port range between 0 and 65535
Service Name
Service Name
Parameter Set Folder
Repository Schema
Repository Schema
Previous
Next

Verify extraction results

After a successful run, confirm the metadata archive in the target folder. See Verify extracted metadata files.

Troubleshoot

If an error occurs during the extraction, see Troubleshoot connector extraction failures.

Oracle ODI

Learn how to configure Oracle ODI as a metadata source for Cloudera Data Lineage (formerly Octopai).



Note: Version Supported: 12c

Tool Permissions Prerequisites

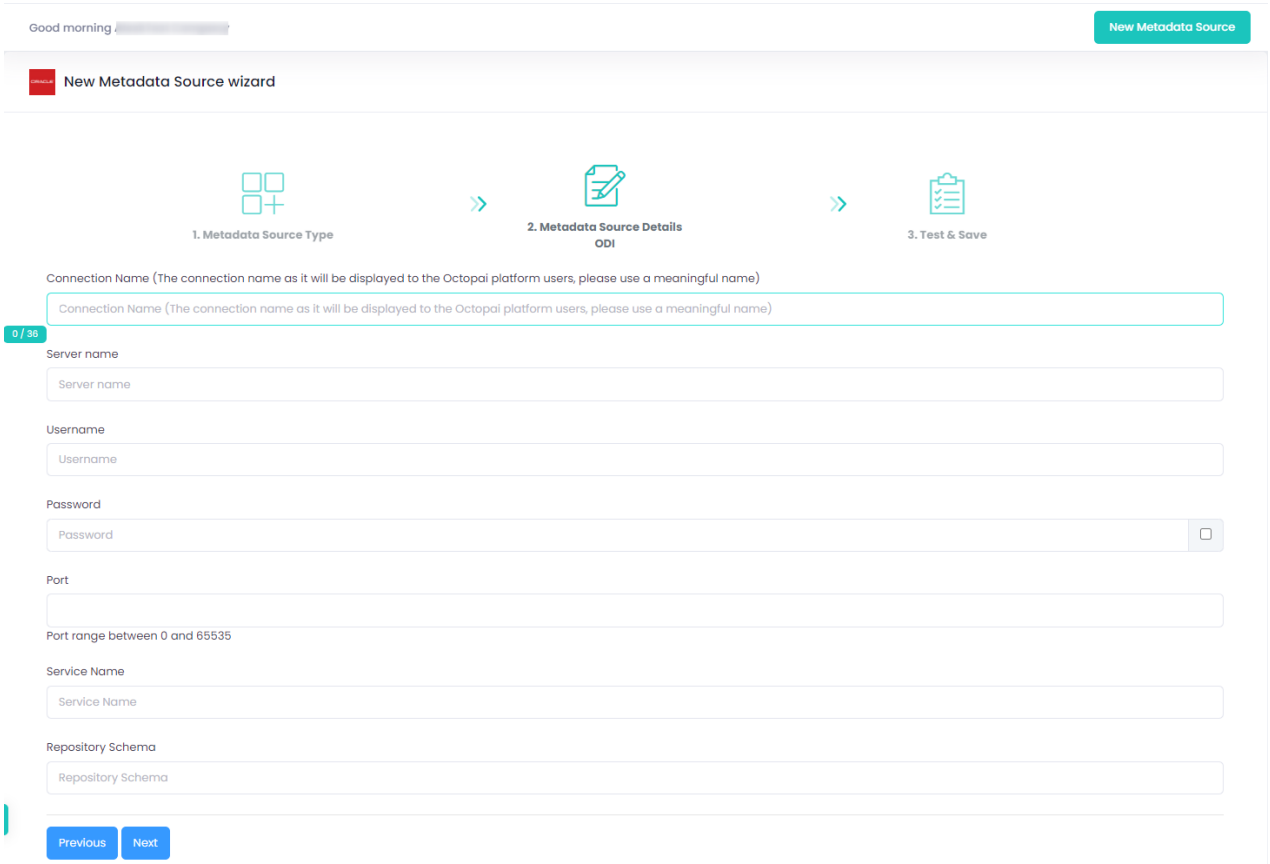


Warning: Missing permissions could end up in broken lineages.

1. Open Server Port for each ODI databases repositorie
2. Existing/New Oracle user for each database with grant read permission for the following dictionary tables:
   - SNP_MAP_CONN
   - SNP_MAP_CP
   - SNP_MAP_ATTR
   - SNP_MAP_EXPR
   - SNP_PACKAGE
   - SNP_MAP_REF
   - SNP_TABLE
   - SNP_COL
   - SNP_DEPLOY_SPEC
   - SNP_EXEC_UNIT_GRP
   - SNP_LOAD_PLAN
   - SNP_EXEC_UNIT
   - SNP_PHY_NODE
   - SNP_PHY_EXPR
   - SNP_MAP_EXPR_REF
   - SNP_MAP_PROP
   - SNP_SCEN
   - SNP_SCEN_FOLDER
   - SNP_MAPPING
   - SNP_SEQUENCE
   - SNP_MODEL
   - SNP_FOLDER
   - SNP_PROJECT
   - SNP_MAP_COMP

Setting up ODI Metadata Source

Metadata Sources are set on the Octopai Client



Connection Name: Giving a meaningful name to the connection will be useful for future references.

Verify extraction results

After a successful run, confirm the metadata archive in the target folder. See Verify extracted metadata files.

Troubleshoot

If an error occurs during the extraction, see Troubleshoot connector extraction failures.

SSIS (Files / Integration Services / Integration Services Catalog / Azure)

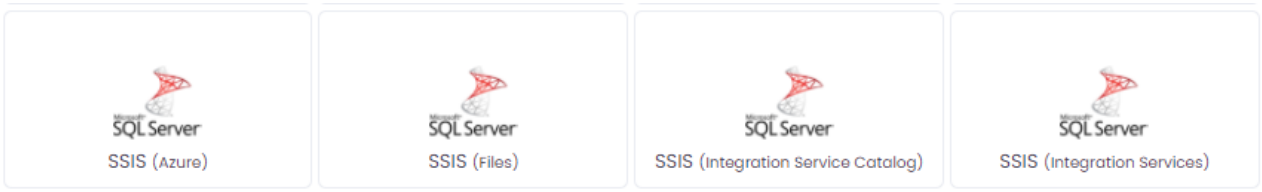
Learn how to configure SSIS (Files, Integration Services, Integration Services Catalog, Azure) as metadata sources for Cloudera Data Lineage (formerly Octopai).

Note: Supported versions: SQL Server 2012 through 2022

SSIS method

In the Octopai Client, select the SSIS method implemented in your organization.

Figure 27: Select the SSIS method in the Octopai Client



SSIS Files

Warning:

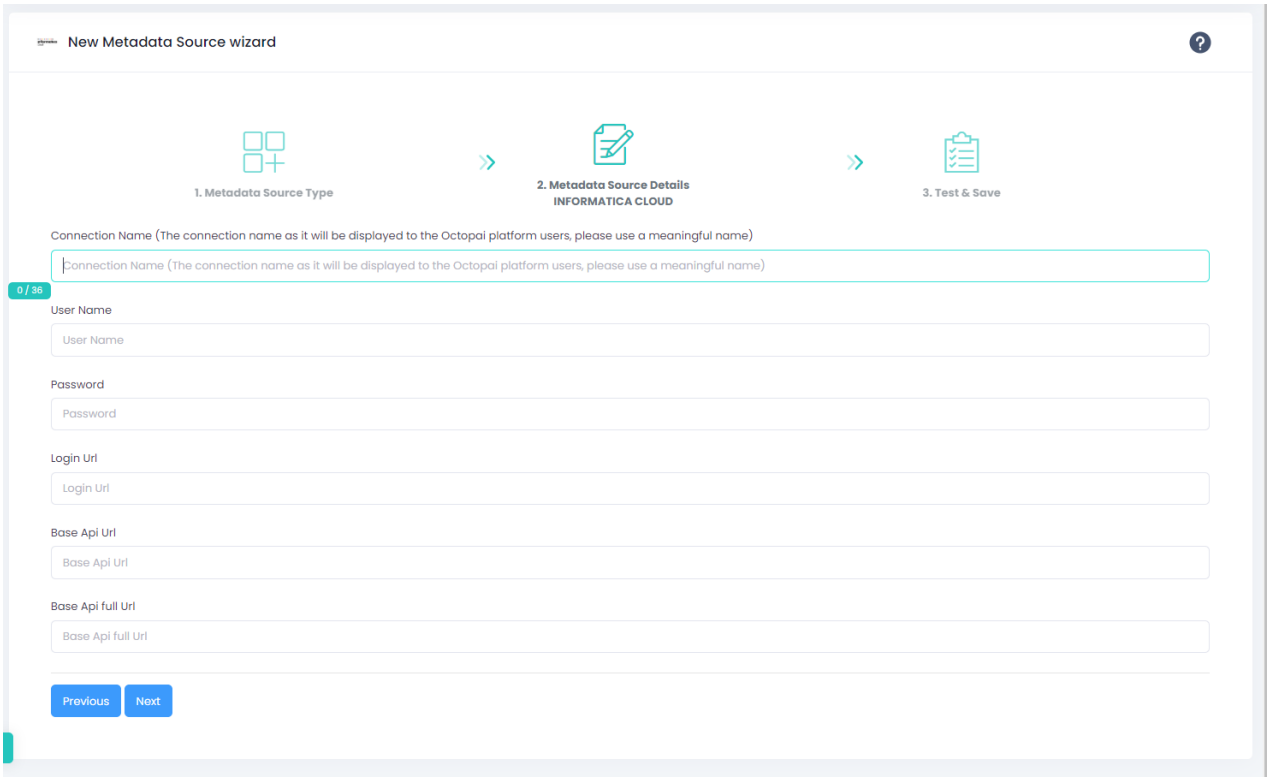
Missing permissions could end up in broken lineages.

Note:

Permissions Prerequisites:

- Read Permission for the Cloudera Data Lineage Windows NT User to the SSIS folder containing the *.dtsx files.
- Read Permission for the Cloudera Data Lineage Windows NT User to the SSIS folder containing the *.dtsx files.

Figure 28: Setting up SSIS Files Metadata Source



SSIS Integration Services

Warning:

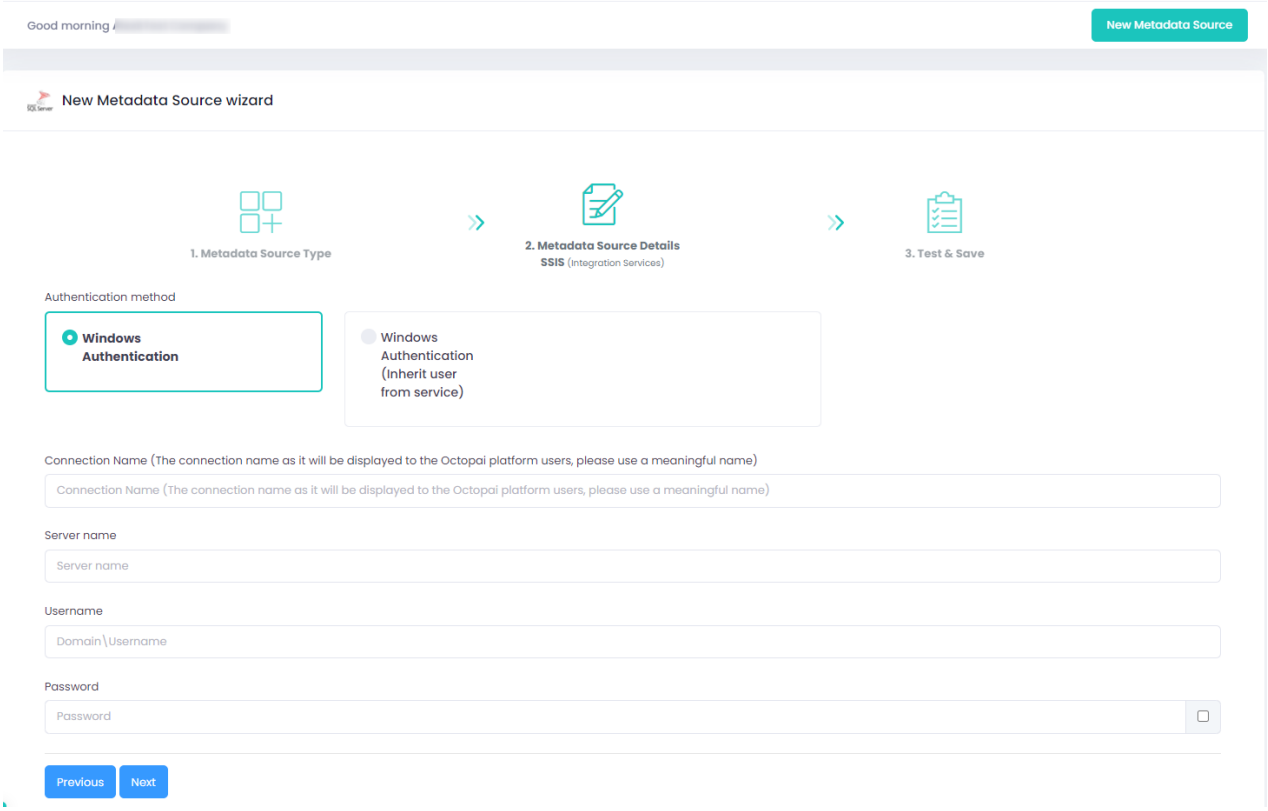
Missing permissions could end up in broken lineages.

Note:

Permissions Prerequisites (Server Type: Integration Services - MSDB):

- Map the following users to the login used for extraction:
  - Database **msdb** with Database role membership **db_datareader**.
  - Database **SSISDB** with Database role membership **public** and **ssis_admin**.

Figure 29: Setting up SSIS Integration Services Metadata Source



SSIS Integration Services Catalog

Warning:

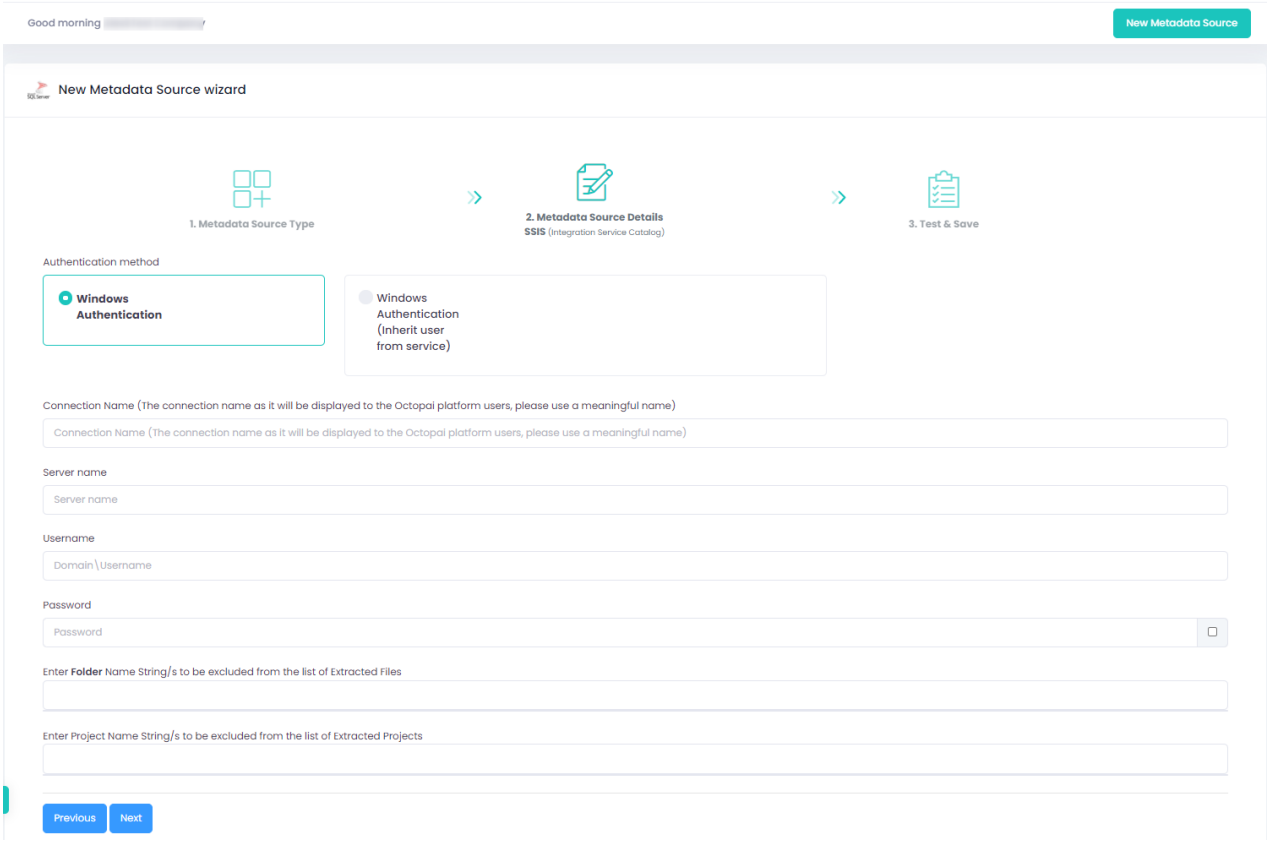
Missing permissions could end up in broken lineages.

Note:

Permissions Prerequisites:

- Grant database role membership to **ssis_admin**.
- Grant read access permission to **SSISDB.catalog.environments**.

Figure 30: Setting up SSIS Integration Services Catalog Metadata Source



SSIS Azure

Warning:

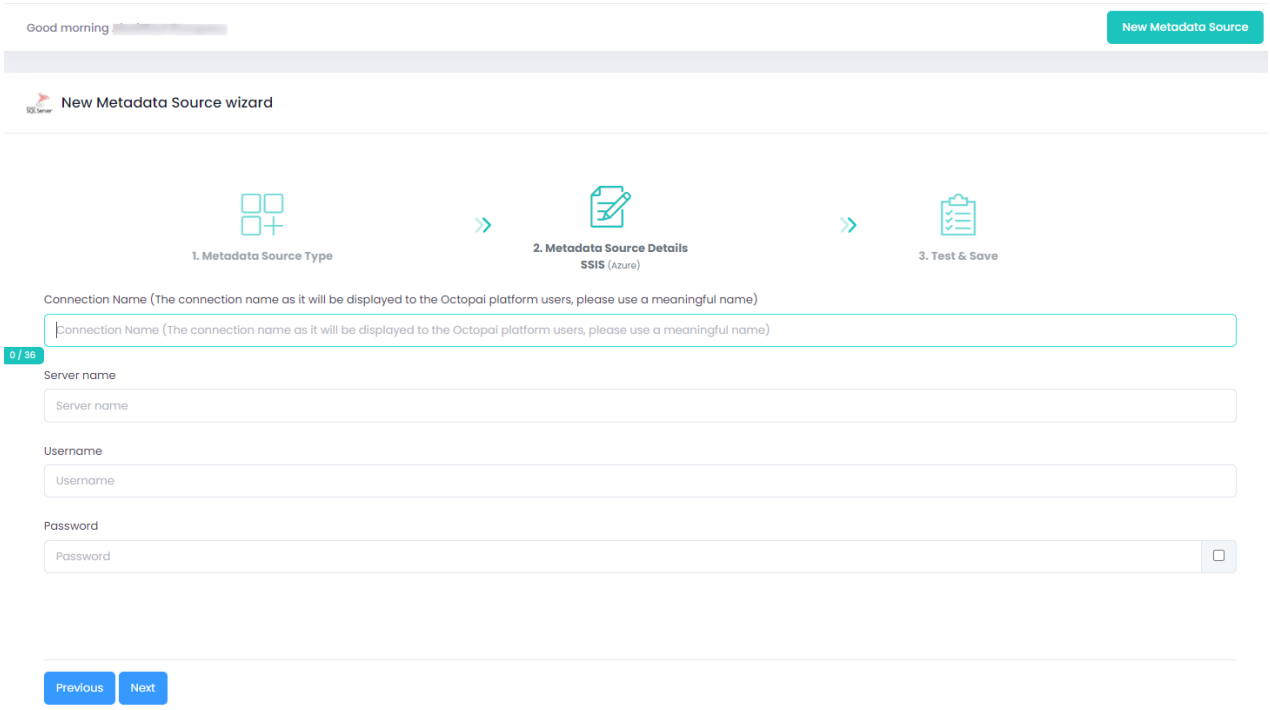
Missing permissions could end up in broken lineages.

Permissions prerequisites

- Define a user with the **db_owner** role on the SSISDB instance for the Integration Services Catalog.

Set up the metadata source

Figure 31: Configure the SSIS Azure metadata source



Verify extraction results

After a successful run, confirm the metadata archive in the target folder. See Verify extracted metadata files.

Troubleshoot

If an error occurs during the extraction, see Troubleshoot connector extraction failures.

BI and reporting

Cloudera Data Lineage extracts report metadata from business intelligence (BI) tools, including Power BI, Tableau, Qlik, Looker, MicroStrategy, IBM Cognos, SAP BusinessObjects, SSAS, and SSRS.

IBM Cognos Report Studio

Learn how to export, schedule, and manage IBM Cognos reports and packages.

Note: Version supported: up to v11.2.x

Before you begin

- Read the Permission for Cloudera Data Lineage (formerly Octopai) Windows NT User to the Cognos ZIP files folder.

Warning: Missing permissions could result in broken lineages.

About this task

Procedure

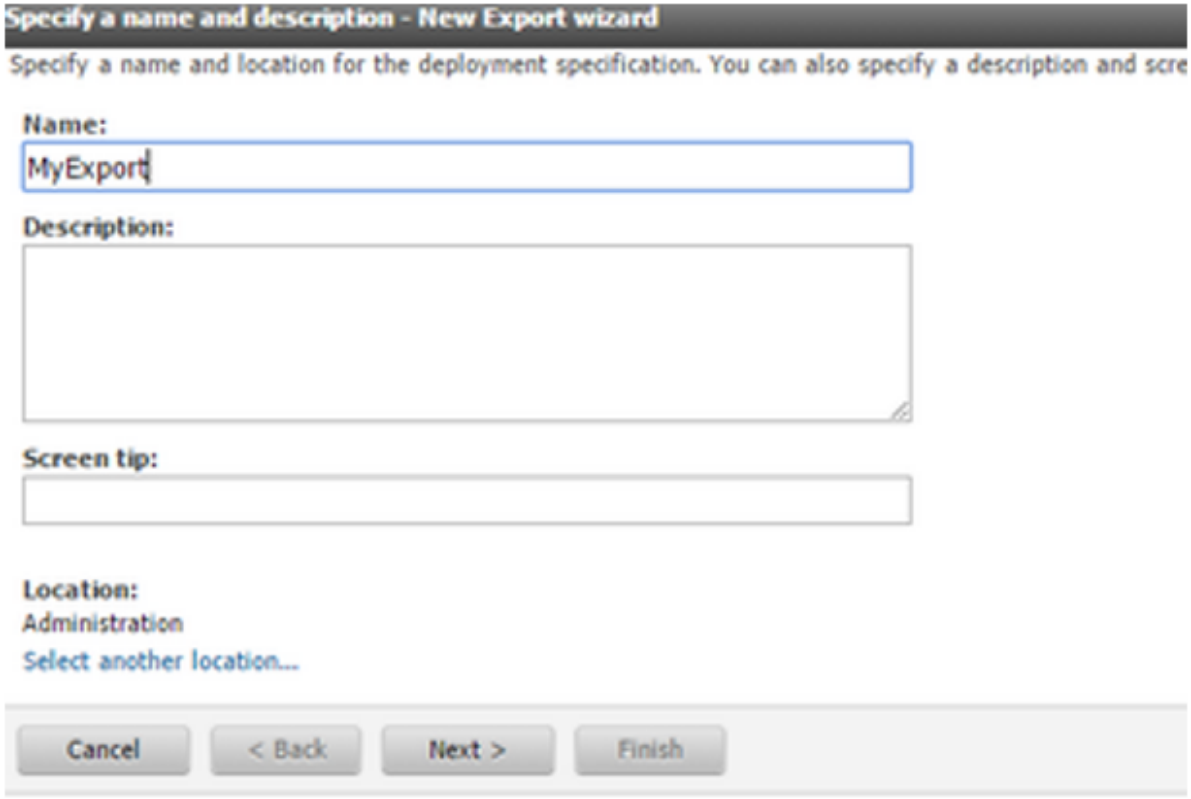
1. Export and Schedule Export of IBM Cognos Reports and Packages
   a) Open IBM Cognos Administration and go to the Configuration tab.



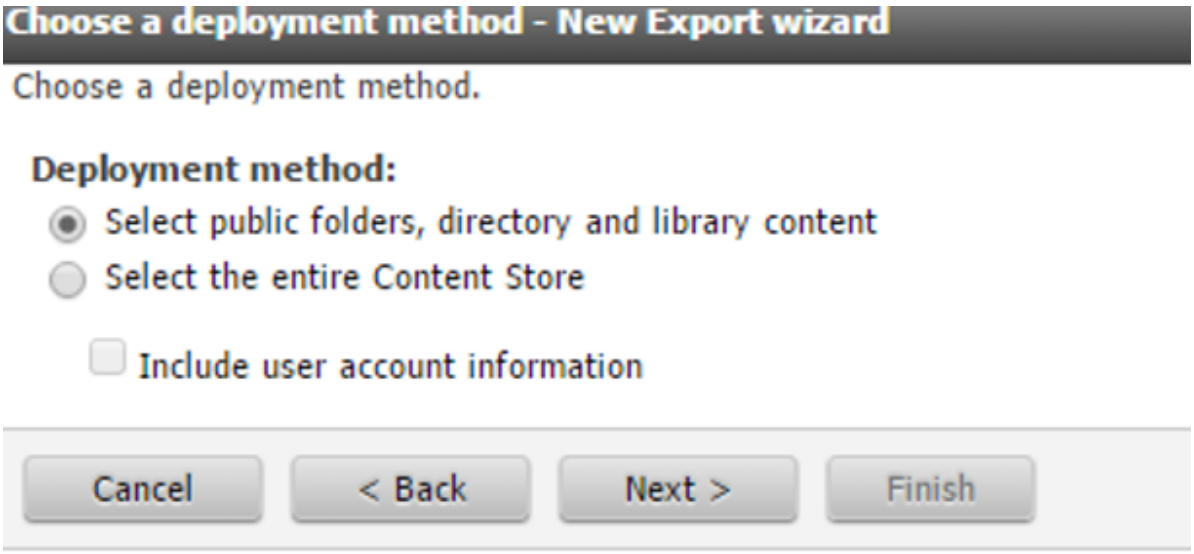
   b) Click the New export button.



   c) Name the export.



   d) Select the deployment method by choosing the first option.



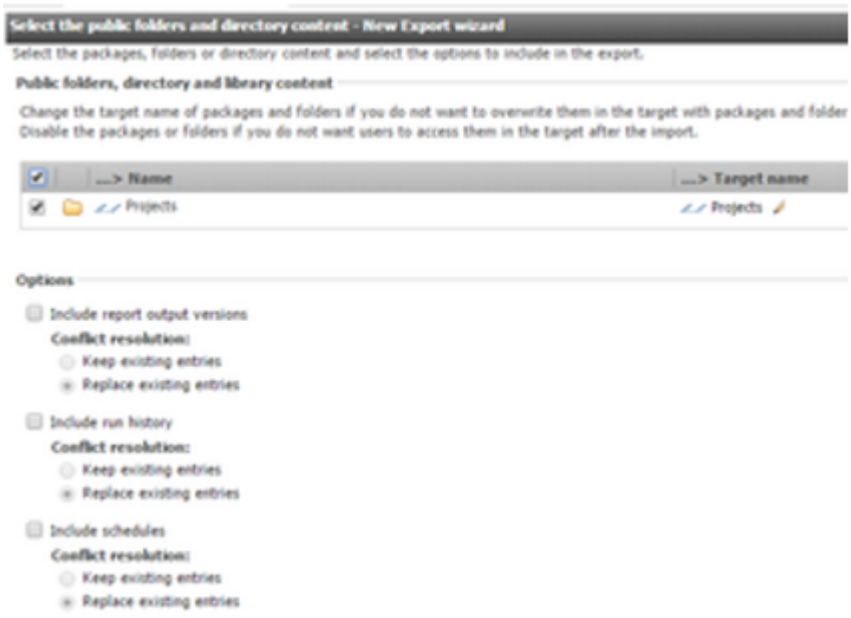
e) Click Add.



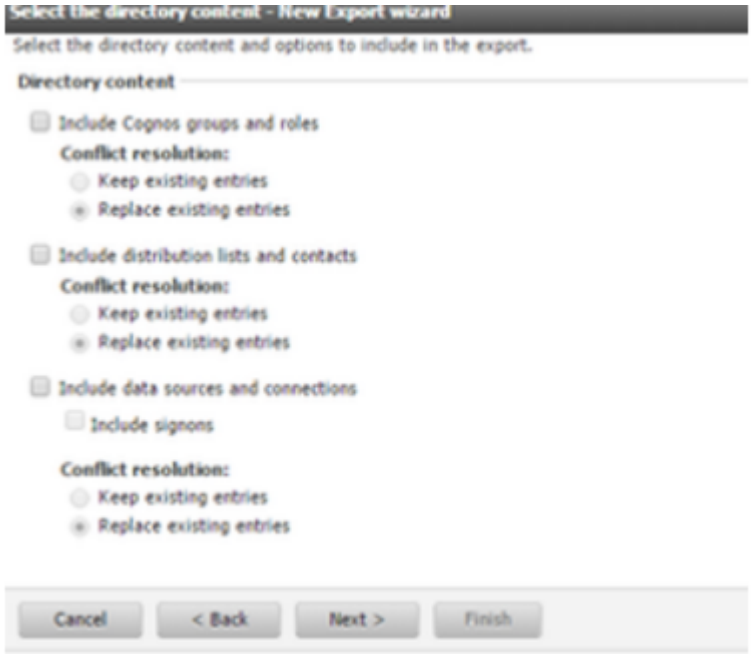
f) Add packages and reports you want by dragging them from left to right and then click OK.



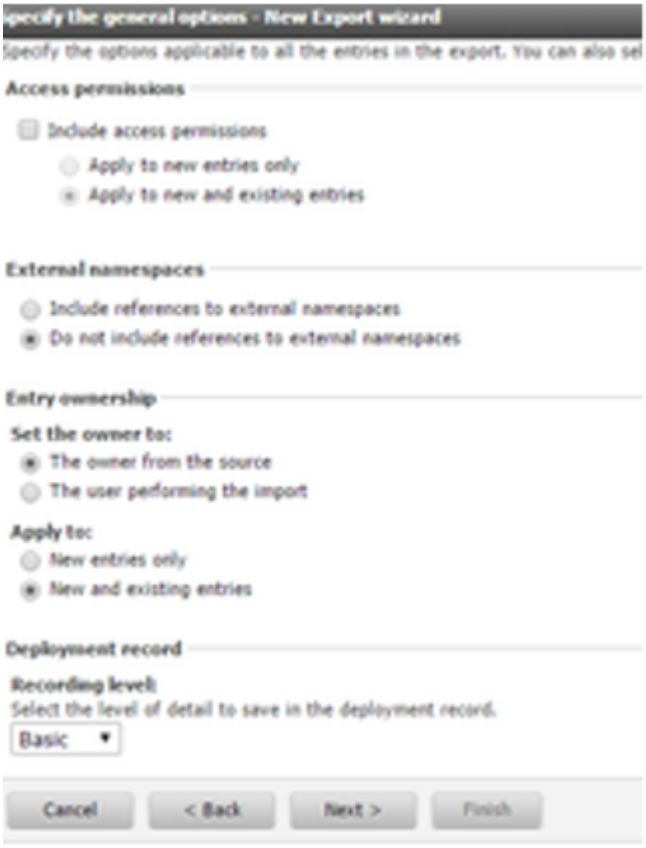
g) Select all the checkbox objects and then click Next.



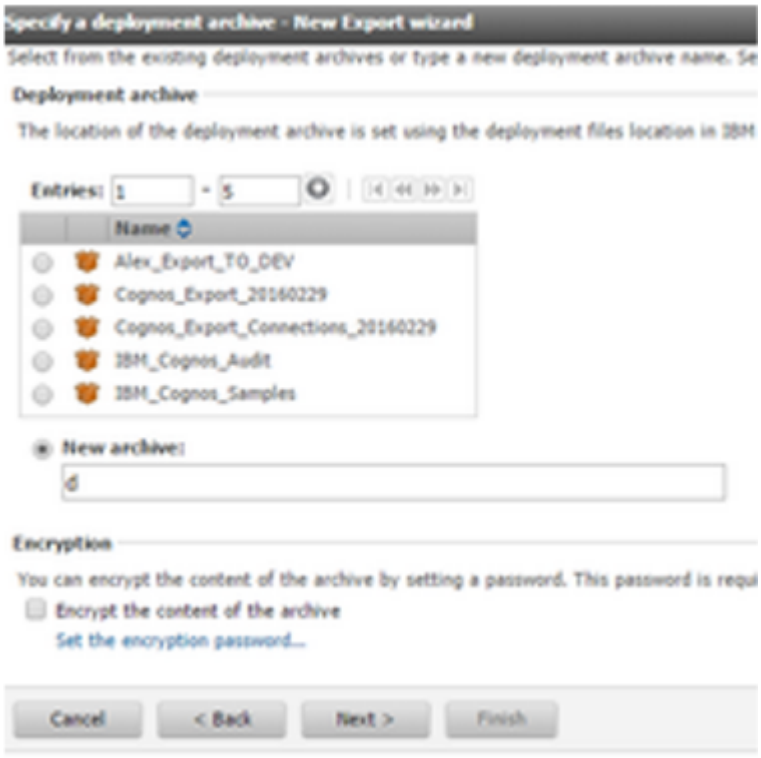
h) Click Next.



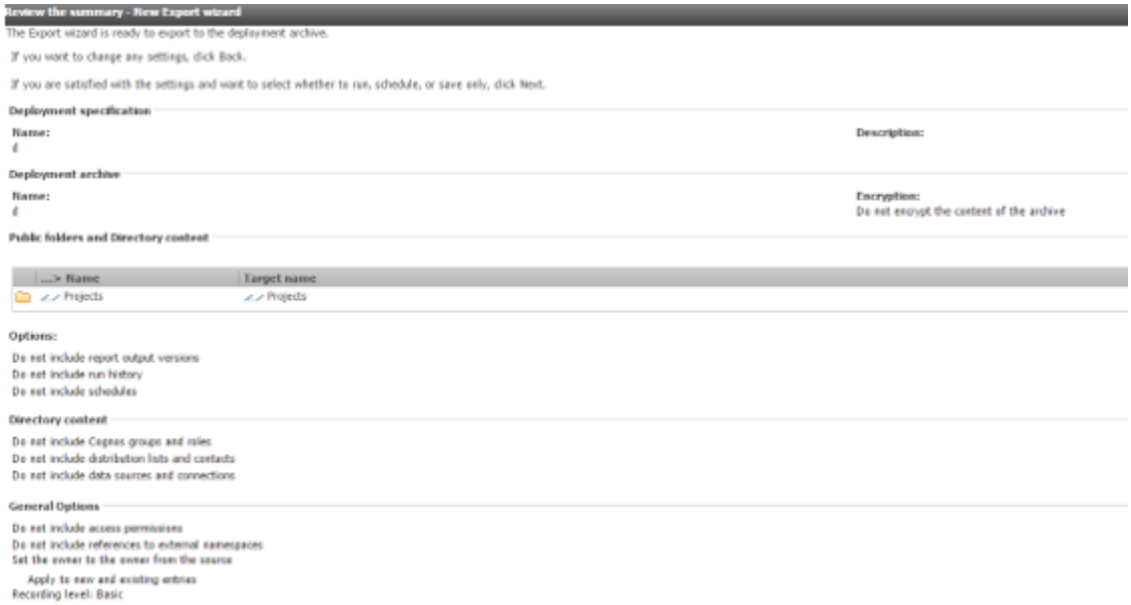
i) Click Next.



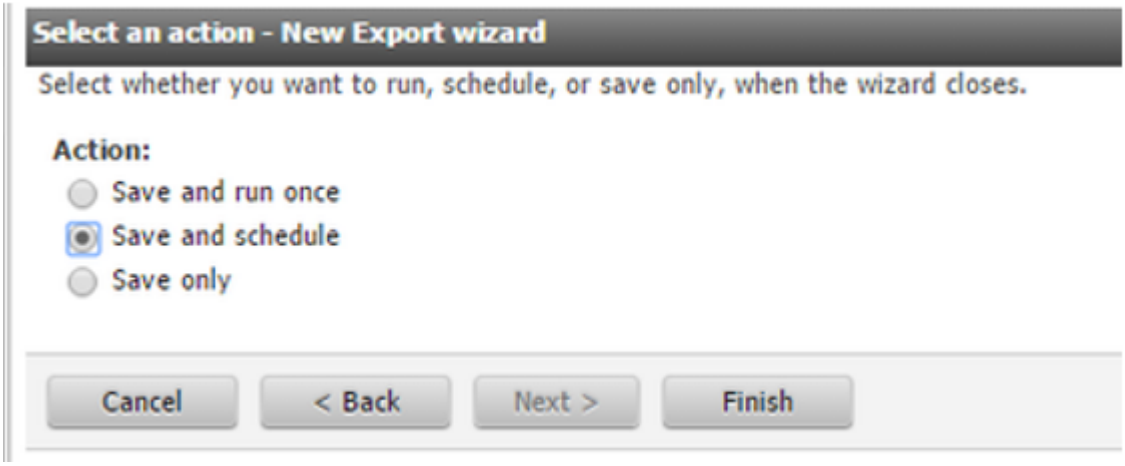
j) Click Next.



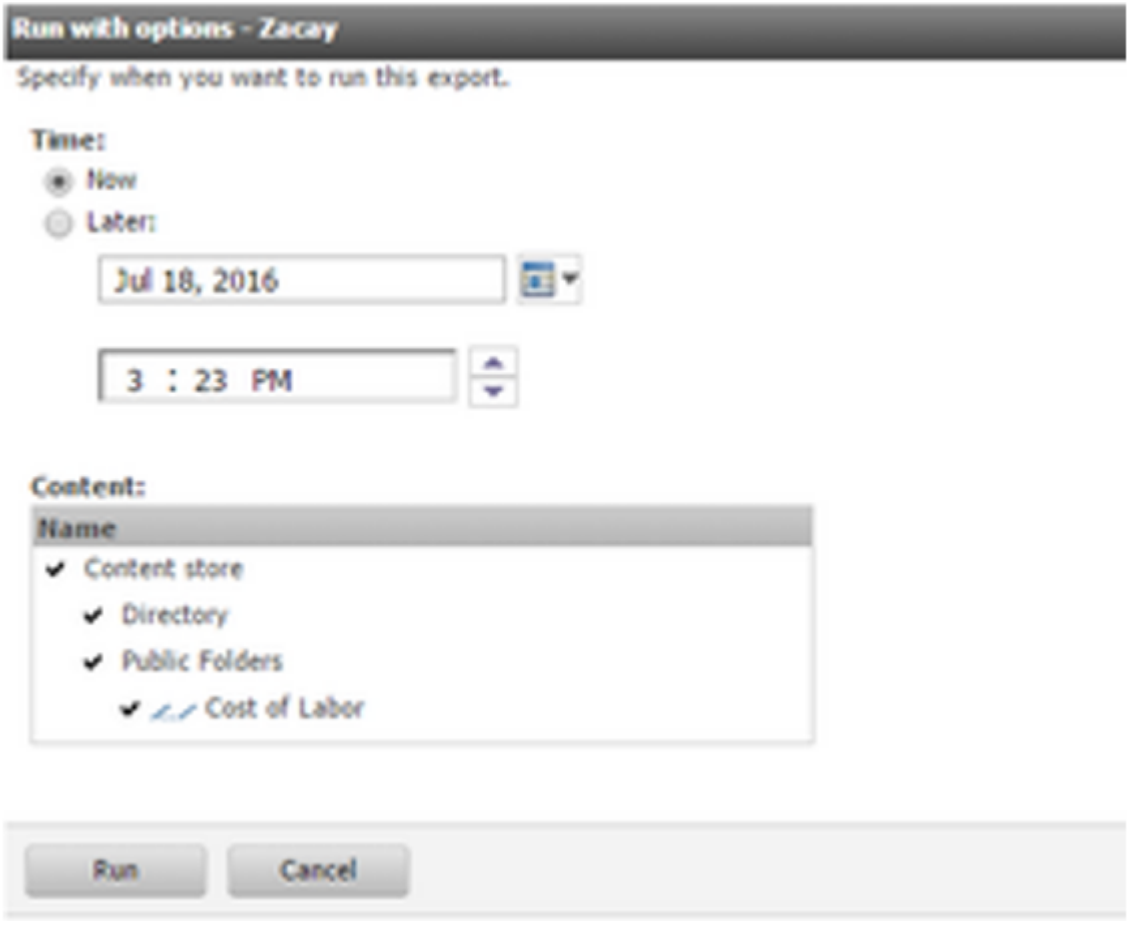
k) Click Next.



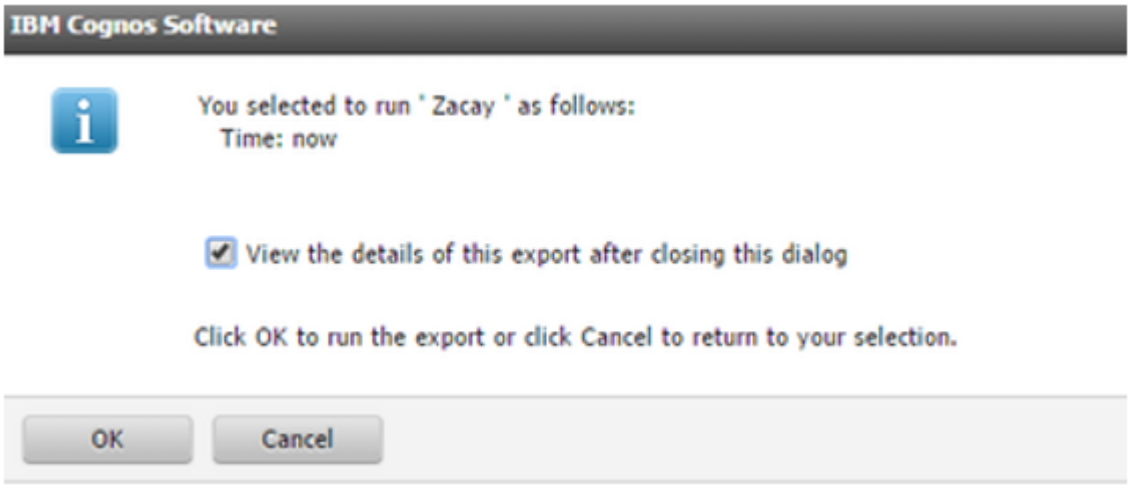
l) Choose the Save and run once or the Save and schedule checkbox and click Finish.



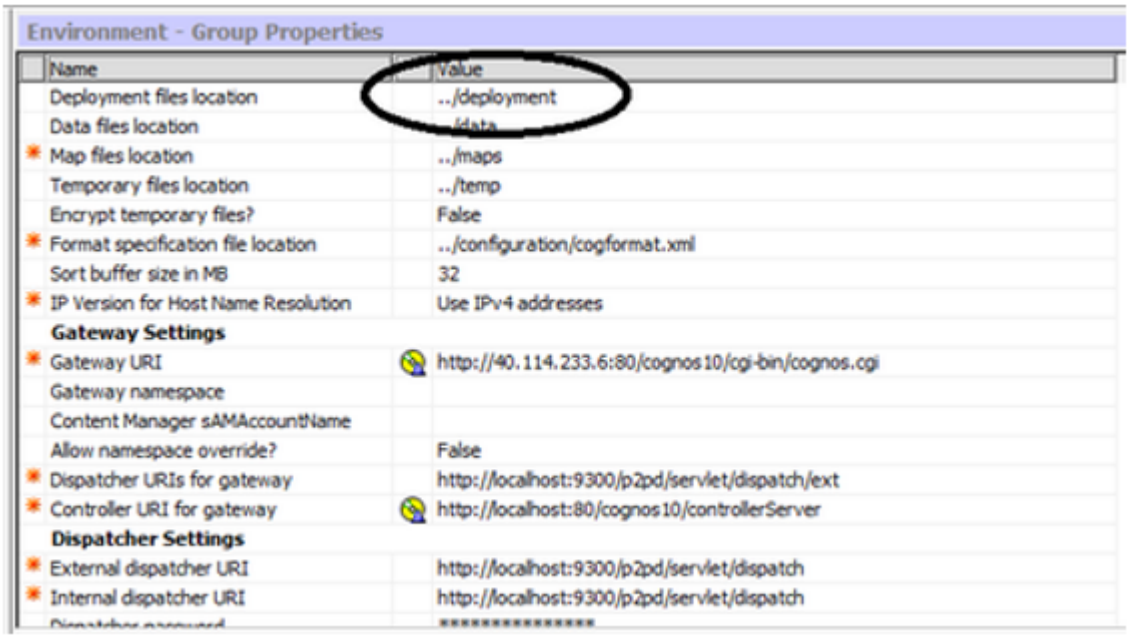
m) If you chose Save and schedule, choose when to schedule and then click Run.



n) Click OK.



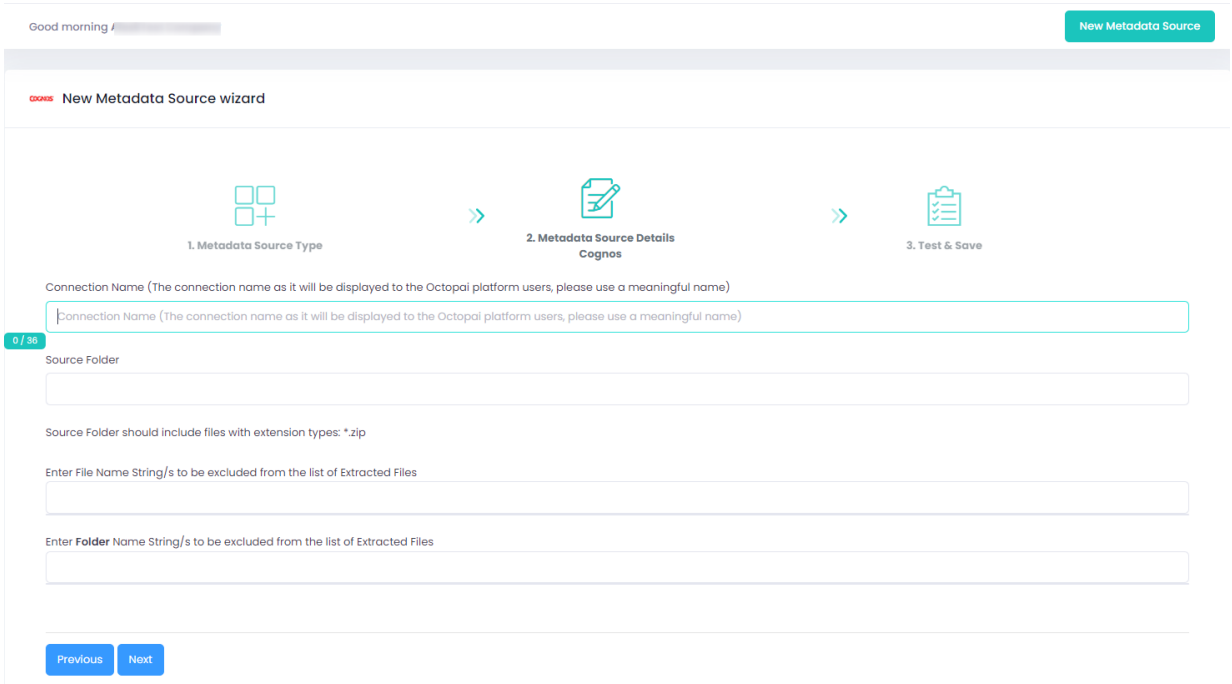
The file is saved under the deployment folder on the Cognos server. Usually the installation folder is at C:\Program Files\ibm\cognos\c10_64\deployment. The extracted file will appear in the destination defined folder.



2. Zip the metadata result file and use it as the source for your Cloudera Data Lineage metadata Cognos connection in the Octopai Client.

   Note: Ensure that you have appropriate permissions to the path so that the Octopai Client can access the file with the user running it.

3. Upload the zip file to the prospect folder in the portal.
4. Share the password of the Cognos package or framework file that defined in the guideline.
5. Share the prospect IBM Cognos version.
6. Set up the IBM Cognos Metadata Source.



7. Verify extraction results.

   After a successful run, confirm the metadata archive in the target folder. See Verify extracted metadata files.

If an error occurred during the extraction, see Troubleshoot connector extraction failures.

Cognos Operational Metadata Intelligence Harvester

Learn how to integrate Cognos metadata into Cloudera Data Lineage (formerly Octopai) by configuring permissions, setting up the metadata source, and verifying the extracted files for enhanced analysis.

Tool Permissions Prerequisites

Ensure the following prerequisites are met before proceeding with the configuration:

1. **API Enablement**:
   - Confirm that the **Cognos Operational Metadata API** is enabled through your Cognos license.
2. **Database Login Permissions**:
   - Secure login credentials with **SELECT** permissions to the relevant Cognos databases and tables.

   From the Audit Database:

   - COGIPF_RUNREPORT
   - COGIPF_USERLOGON

   From the Content Share Database:

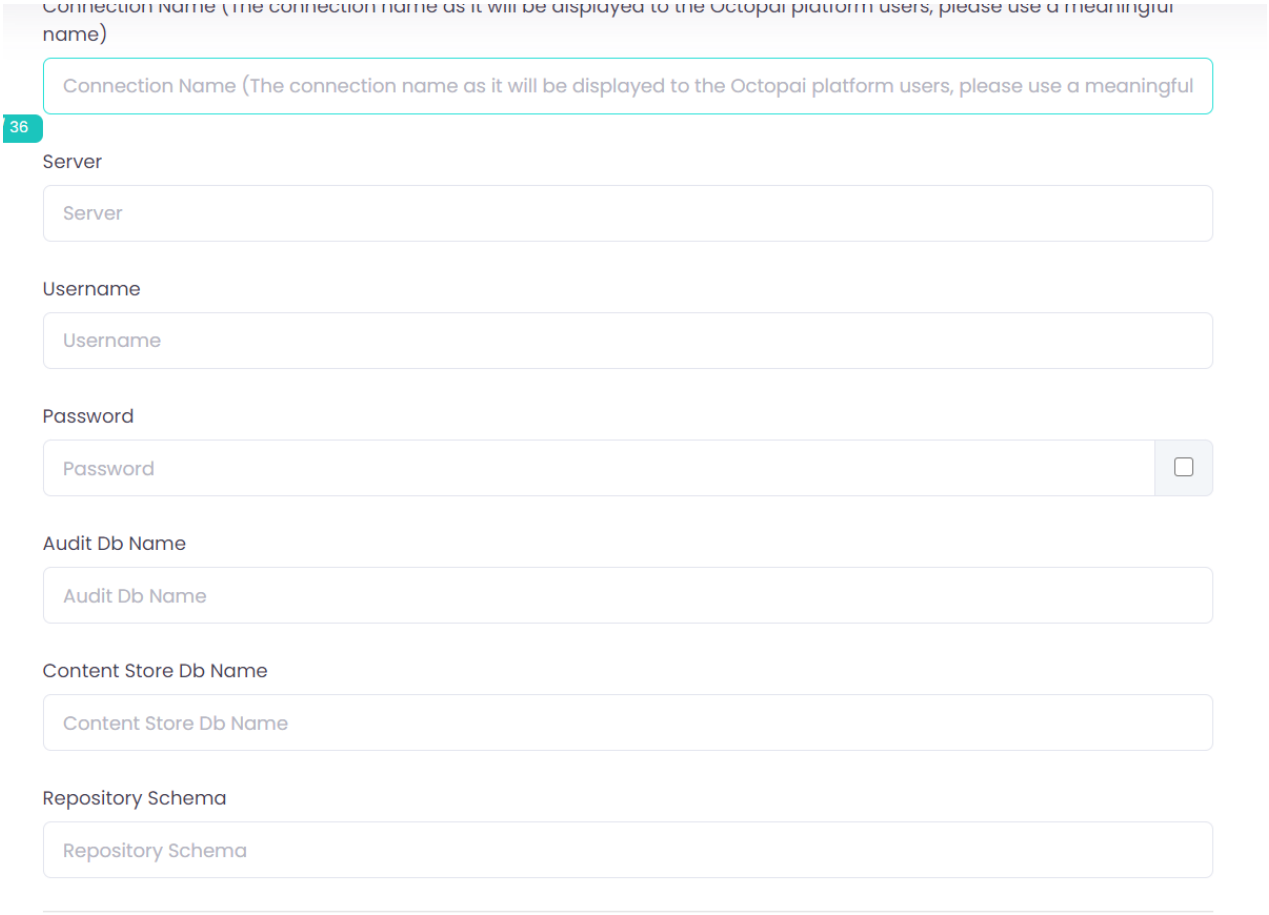
   - CMOBJECTS
   - CMCLASSES
   - CMOBJNAMES
   - CMREFNOORD1
   - CMREFNOORD2
   - CMOBJPROPS33
3. **Database Details**:

- Relevant SQL Server instance details.
- Username and password with appropriate permissions.

Setting Up Cognos Metadata Source in Cloudera Data Lineage

Open Octopai Client:

- Access the **Octopai Client** installed on your server.



Metadata Source Configuration:

- Set the Cognos Metadata Source within the client interface.

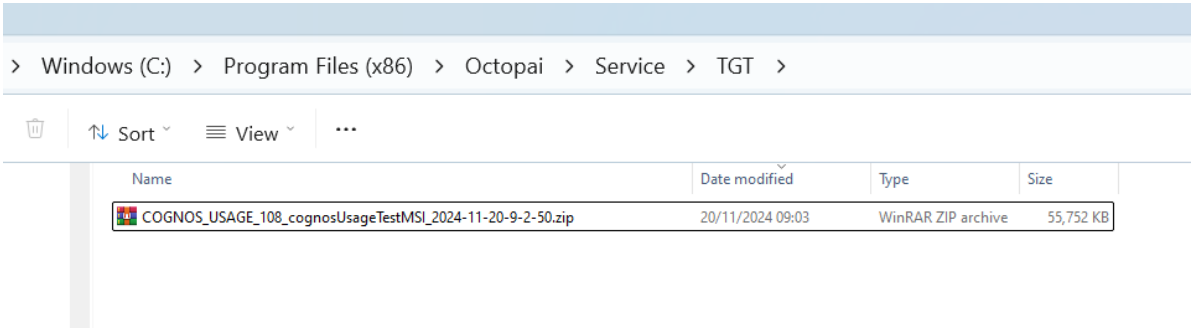
  The **Repository Schema** field is optional. If not specified, it defaults to dbo.

Verifying the Extracted Metadata File

Once the metadata extraction process is complete, verify the extracted file as follows:

1. **Locate the Target Folder**:
   - Navigate to the **Octopai Target Folder (TGT)** on the server where the Octopai Client is installed. Default path:

     C:\Program Files (x86)\Octopai\Service\TGT



2. **Open the Extracted File**:
   - Find the ZIP file corresponding to the connector name (e.g., Cognos_Metadata.zip).
3. **Validate File Content**:
   - Check the quantity and quality of the inner files to ensure all expected metadata has been successfully extracted.

By completing the above steps, Cognos Operational Metadata will be integrated into Cloudera Data Lineage for enhanced metadata intelligence and analysis.

Looker

Learn how to configure Looker as a metadata source in Cloudera Data Lineage (formerly Octopai).

Note: Version supported: up to 22.x

Tool Permissions Prerequisites

Warning: Missing permissions could end up in broken lineages.

Dedicated API3 Credentials (Client ID and Client Secret)

Looker API authentication | Google Cloud with the following permissions:

- access_data
- see_looks
- explore
- see_sql
- see_lookml
- use_sql_runner
- see_lookml_dashboards
- see_looks
- see_queries
- see_datagroups
- develop

We recommended creating a dedicated Permission Set and Role to associate with the credentials.

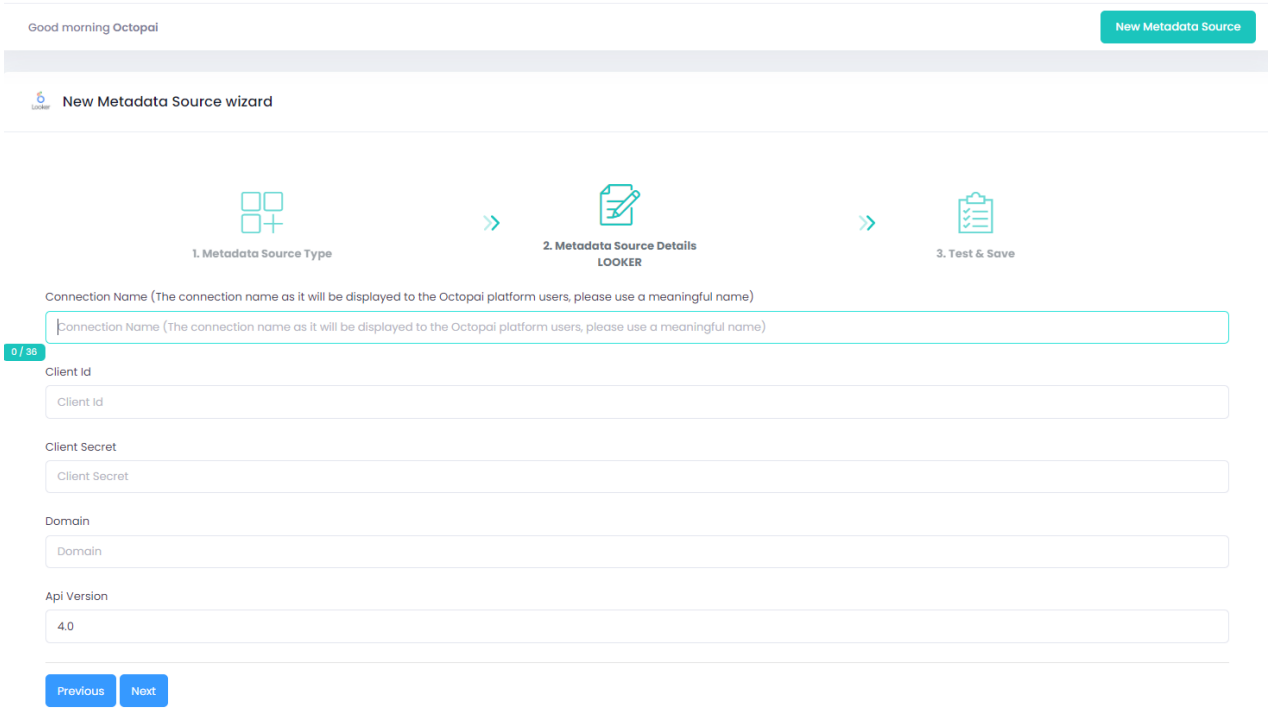
Supported API version 4.0 & up

How to set up the permissions

Looker API authentication | Google Cloud

Setting up Looker Metadata Source

Metadata Sources are set on the Octopai Client



Verify extraction results

After a successful run, confirm the metadata archive in the target folder. See Verify extracted metadata files.

Troubleshoot

If an error occurs during the extraction, see Troubleshoot connector extraction failures.

MicroStrategy

Executing the Project Merge executable, projectmerge.exe (Part of MicroStrategy Tools)

Note: Version supported: up to v2023 + Cloud Version

Tool Permissions Prerequisites

Warning: Missing permissions could end up in broken lineages.

Ensure that the Cloudera Data Lineage (formerly Octopai) Windows NT User has Read Permission to the MicroStrategy *.mmp or backup files folder.

How to Automate Extraction of MMP Files

The MMP files contain the necessary metadata for the analysis by Cloudera Data Lineage .

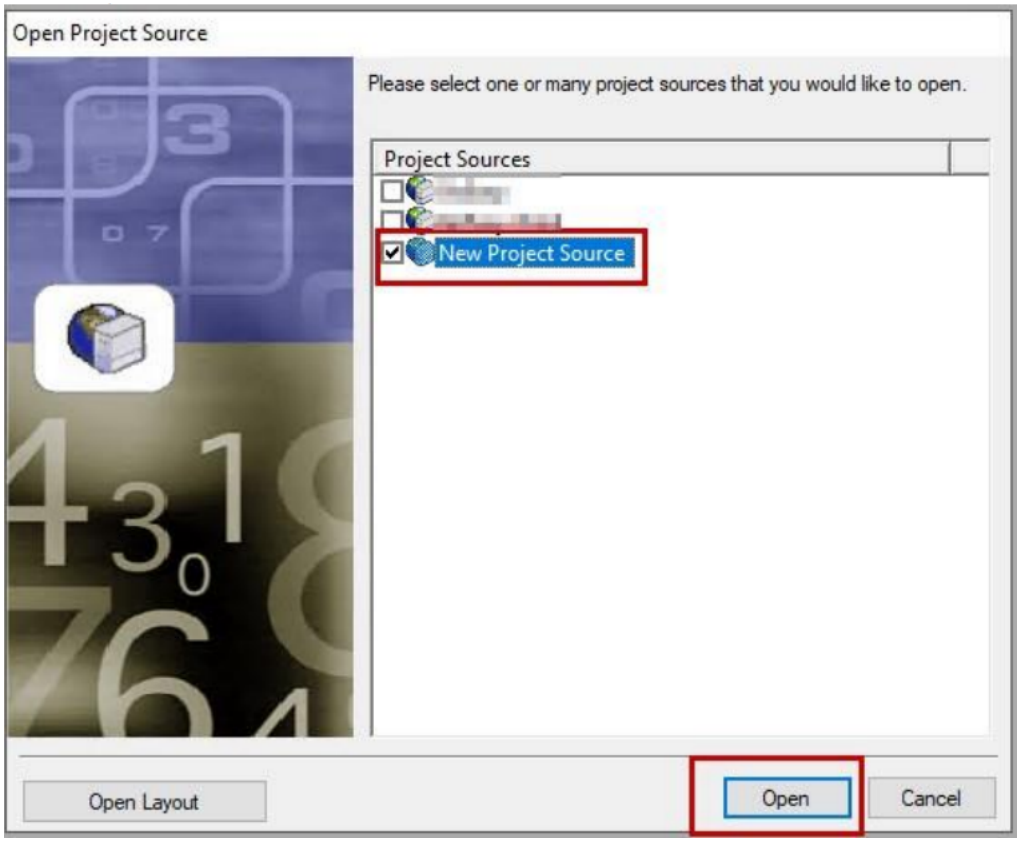
Note: To automate creating the MMP files you will need the projectmerge.exe installed (part of the MicroStrategy tools)

Follow these steps to create MMP files:

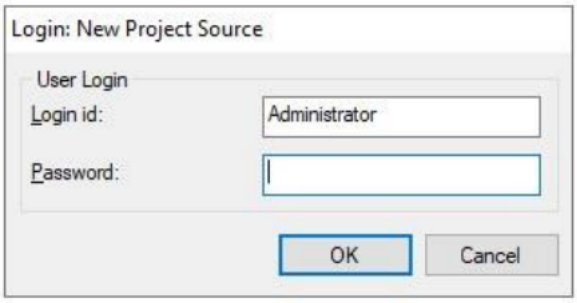
1. Open the MicroStrategy Object Manager



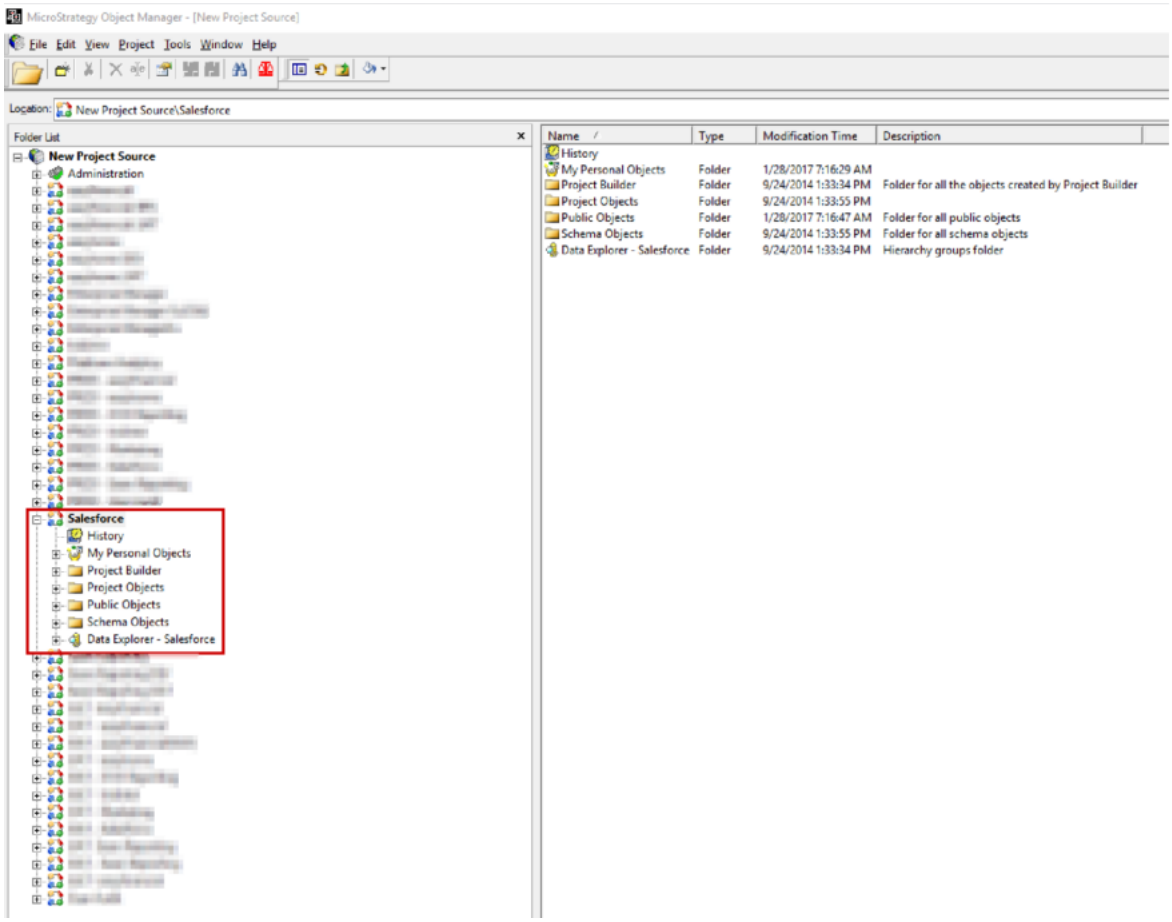
2. Select Project Source and click on Open



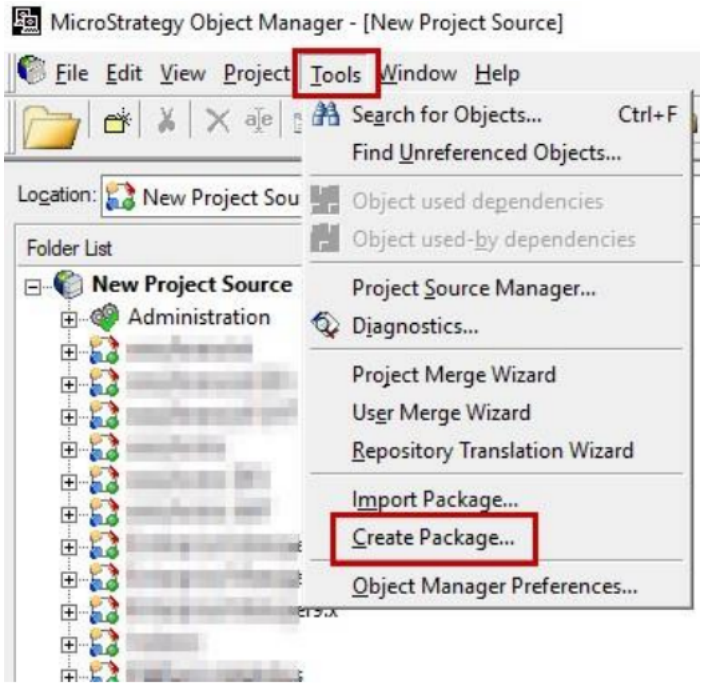
3. Enter login credentials (with admin user)



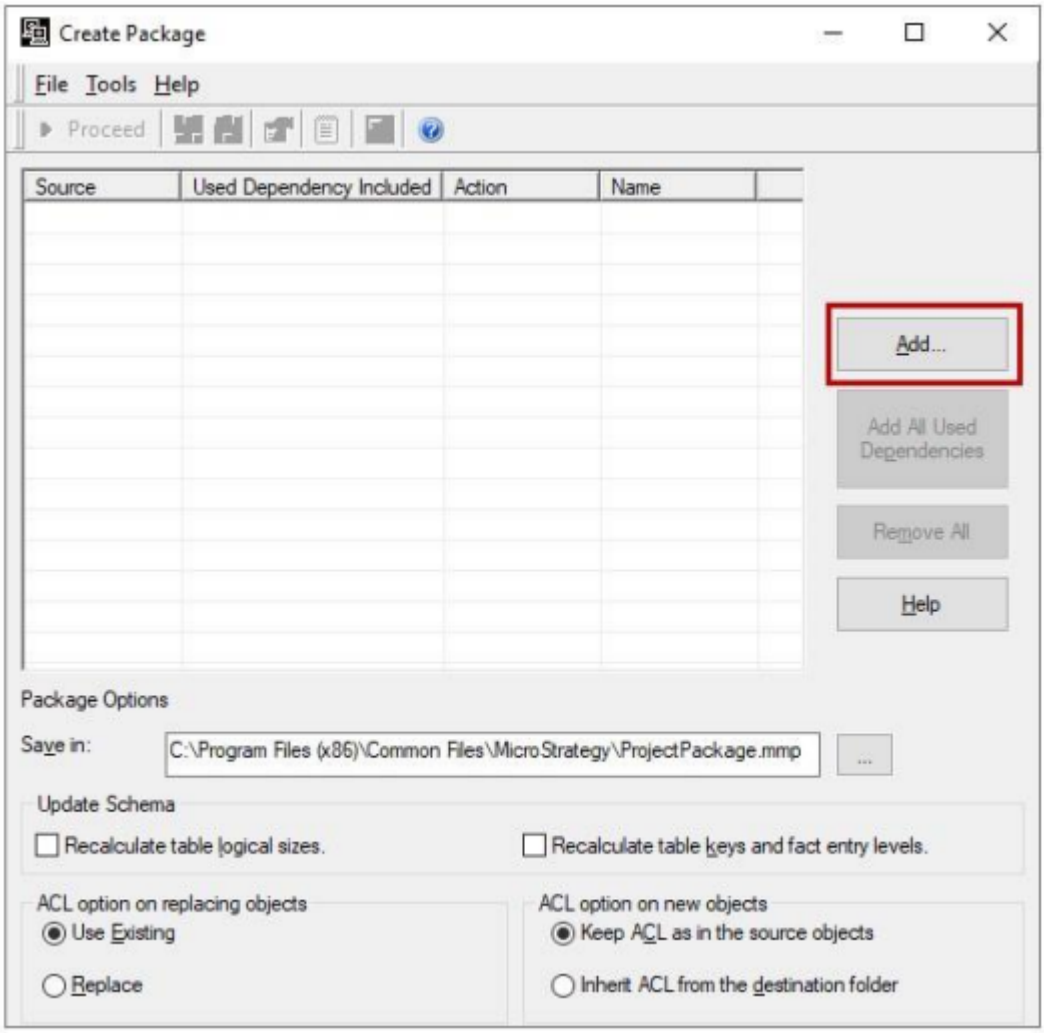
4. Select Project from the left side panel



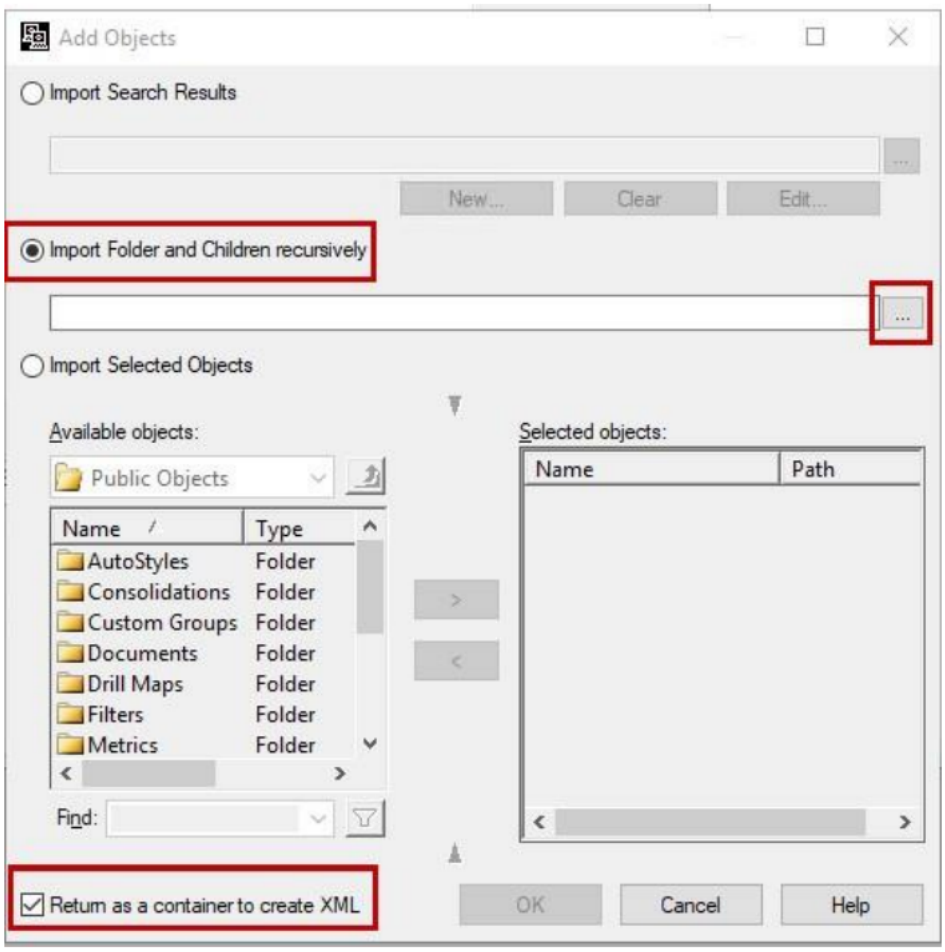
5. Click on Tools --> Create Package



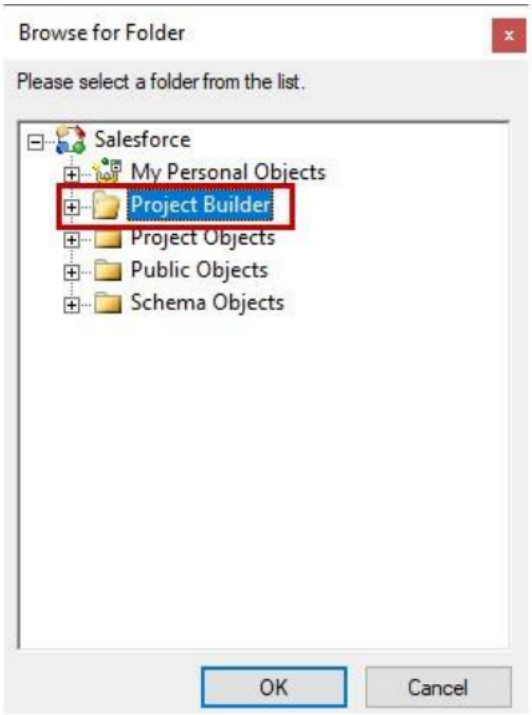
6. Click on Add



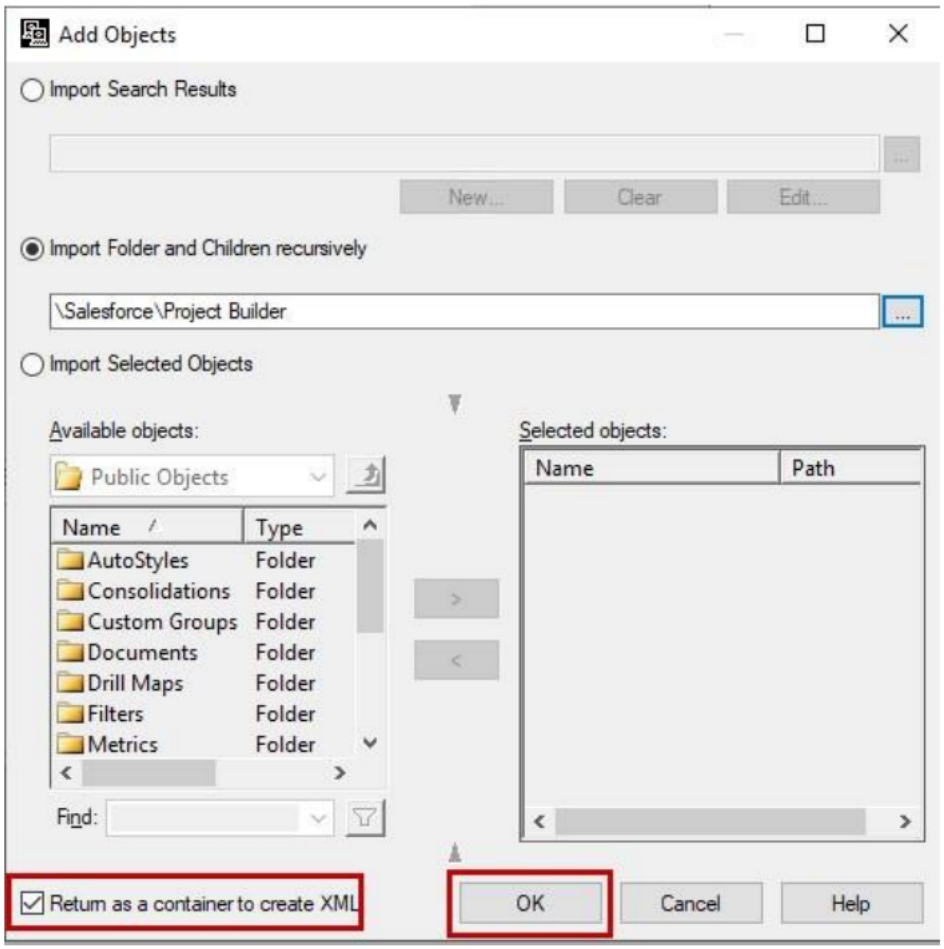
7. Check Import Folder and Children recursively



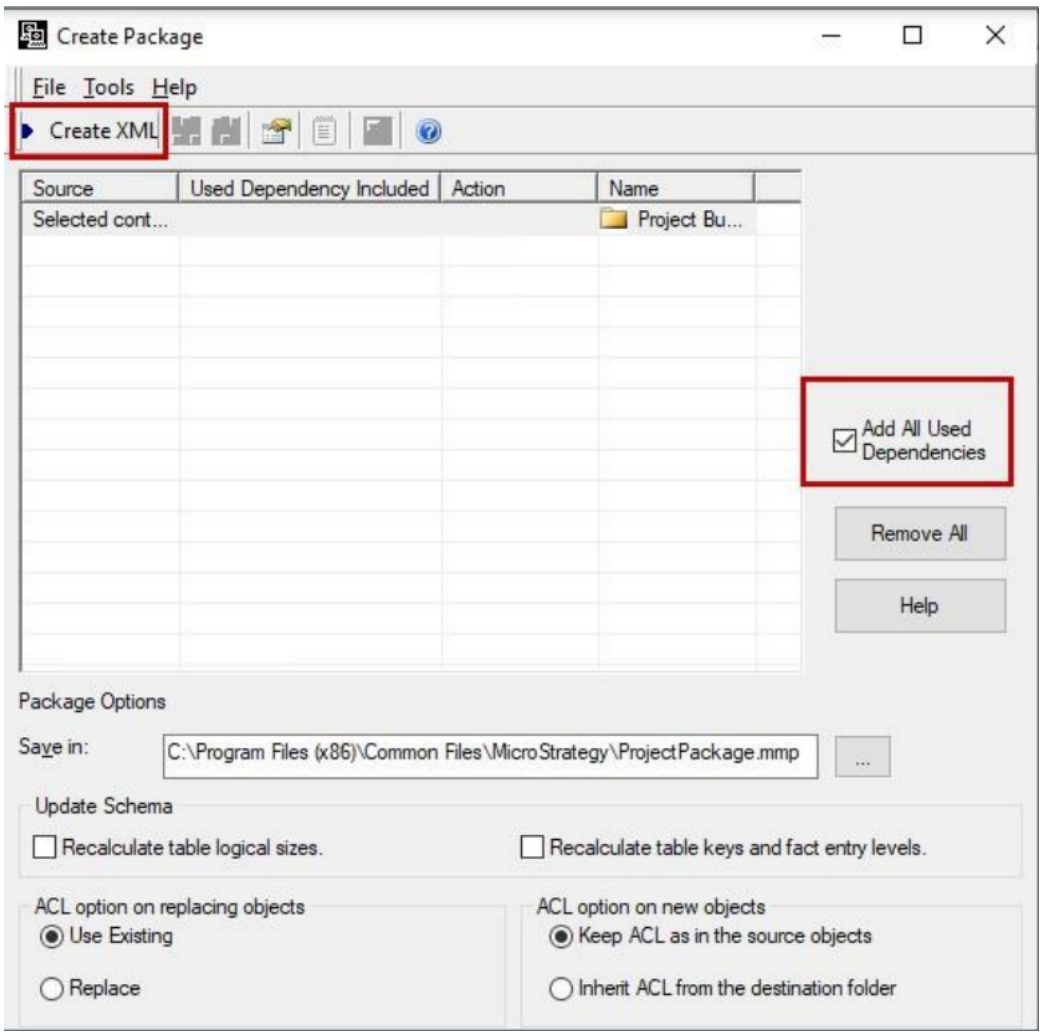
8. Select one folder and click OK



9. Check Return as a container to create XML and click OK



10. Check “Add All Used Dependencies,” change the Save in path, rename the target file as necessary, and click on Create XML



11. Repeat steps 8-10 for each folder within the project (Normally at least 4 folders per project).

12. Repeat steps 4-11 for each project.

Using the XML file to create an updated package:

Executing the Project Merge executable, projectmerge.exe (Part of MicroStrategy Tools)

Use the user and password that you use in order to log in to the Object Manager.

Enter the following command in a batch file: c:\Users\projectmerge -f\Filename.xml -sp -sup

```
C:\Users\octopai>projectmerge -fC:\temp\T.xml -sp******** -sup
```

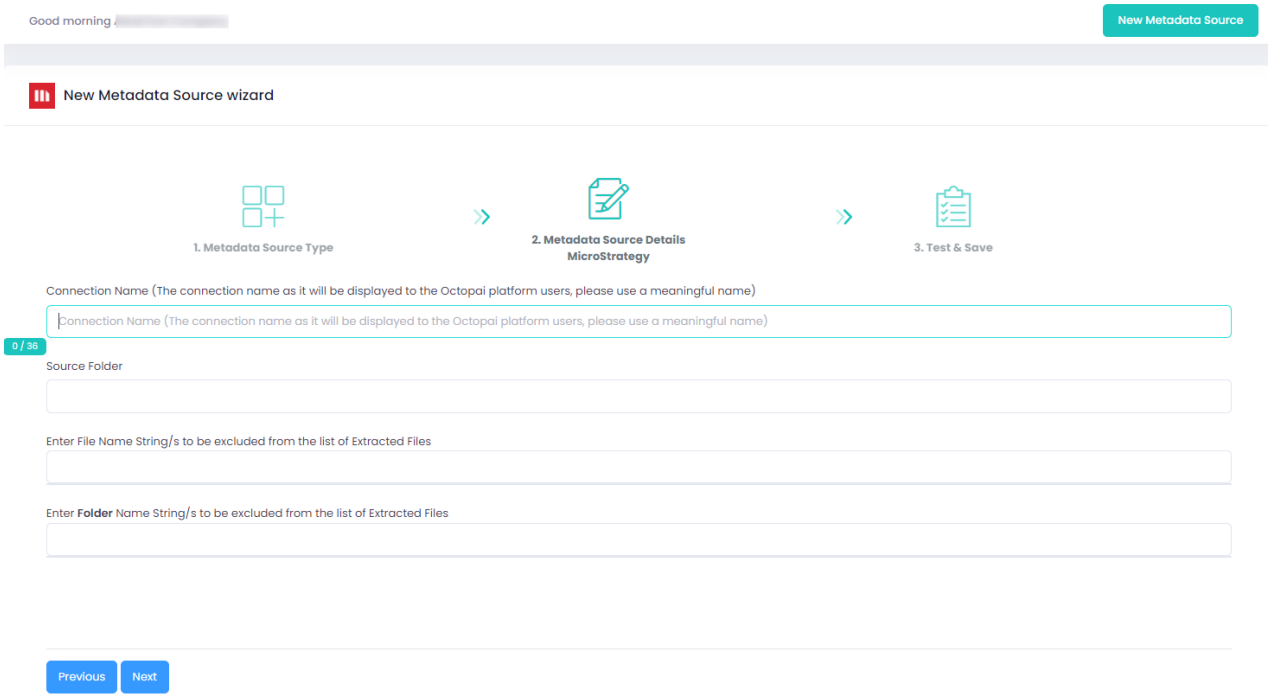
Note: Creating a package from the command line locks the project metadata for the duration of the package creation. Other users cannot make any changes to the project until it becomes unlocked.

Note: If the export gets stuck hanging, split the export to smaller files by creating separate files within each folder by selecting subfolders in step 8.

Note: Schedule the batch files one after the other with 10 minute intervals between each file. The file (*.MMP) will appear in the defined destination folder. Use these files for your Cloudera Data Lineage metadata source in the Octopai Client.

Setting up MicroStrategy Metadata Source

Metadata Sources are set on the Octopai Client :



Oracle OBIEE / Oracle Analytics Server (OAS)

Learn how to extract and configure Oracle OBIEE metadata files for analysis by Cloudera Data Lineage (formerly Octopai).

Before you begin

- Supported version: up to 12c / 2023
- Read the Permission for Cloudera Data Lineage Windows NT User to the OBIEE RDP and Catalog files folder.

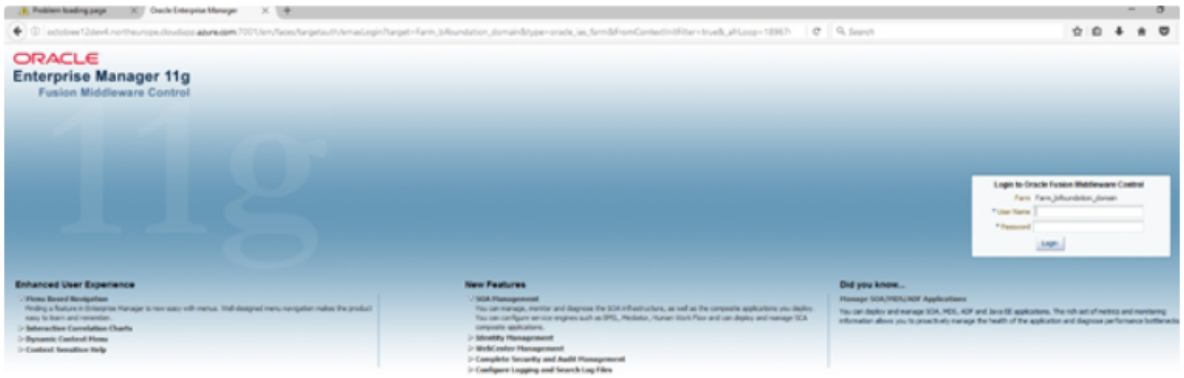
Warning: Missing permissions could result in broken lineages.

Procedure

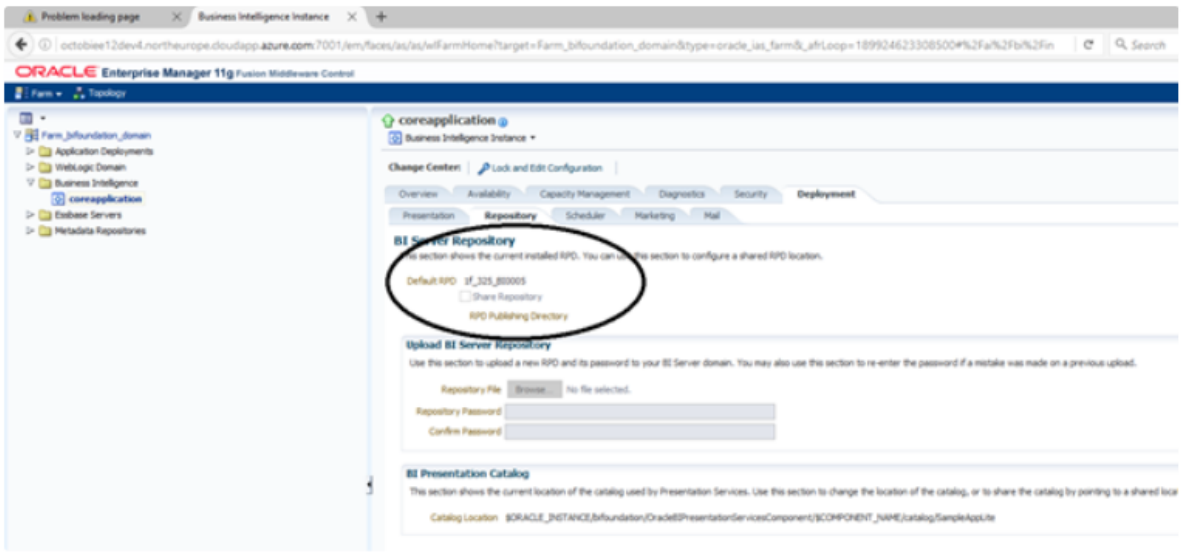
1. **Note:** The rpd and catalog files contain the necessary metadata for the analysis by Cloudera Data Lineage.

   Retrieve the repository rpd file

   a) Log in to Oracle OBIEE enterprise manager (em) at http://obiee12c:9500/em/.



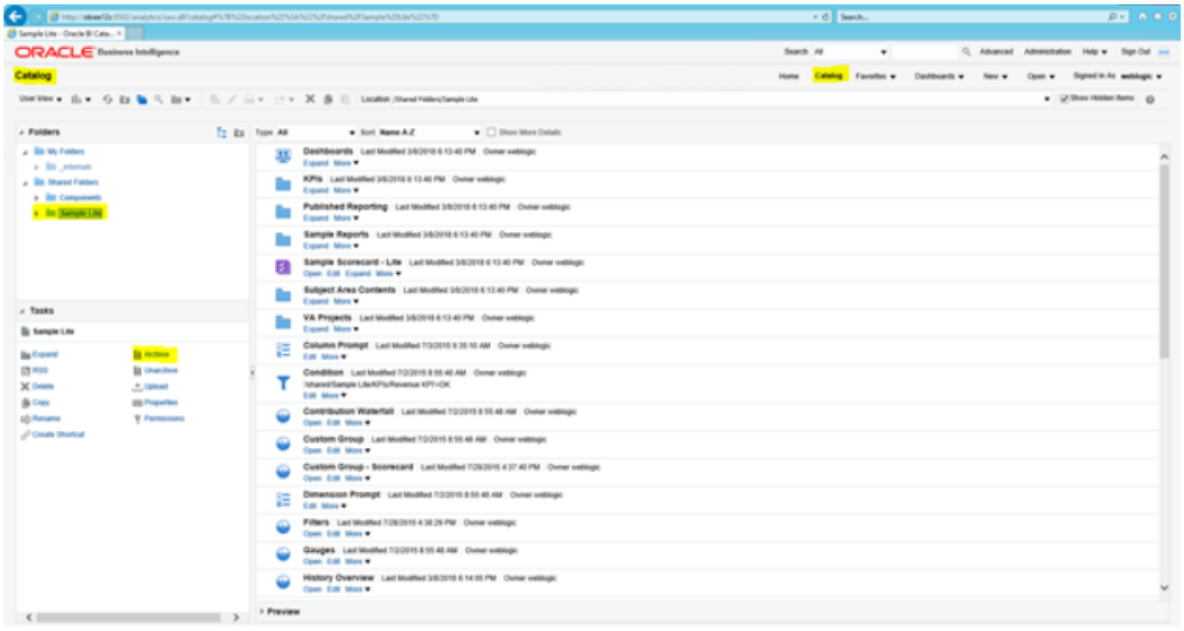
b) Go to Business Intelligence coreapplication Repository .



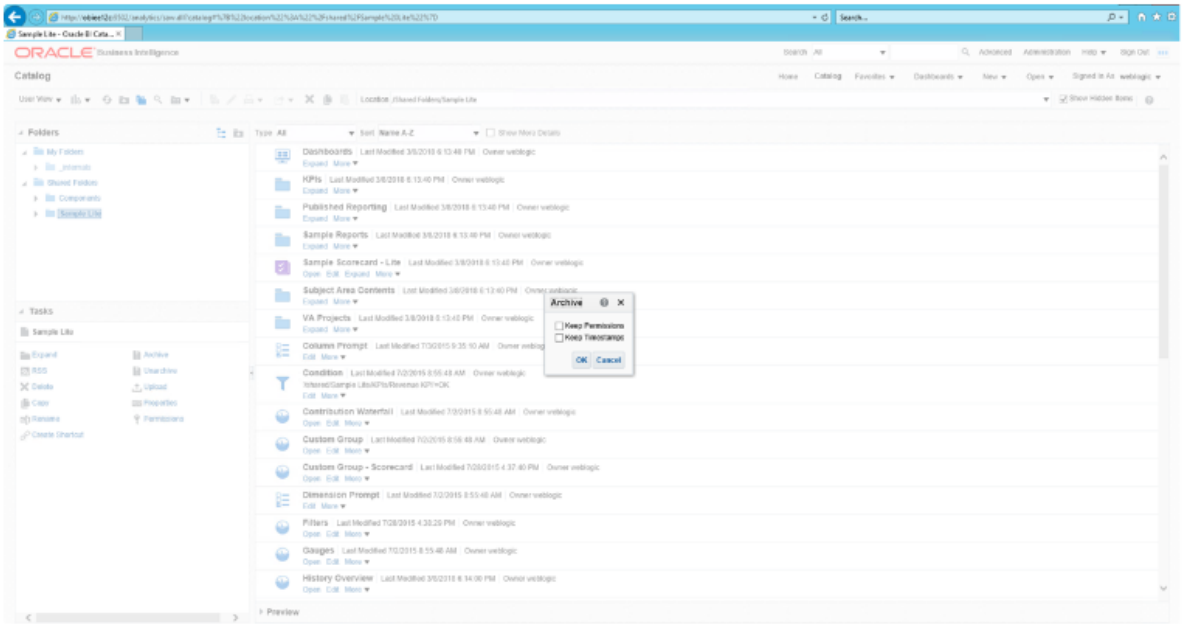
The rpd file can be found at D:\Oracle\Middleware\ user_projects\domains\bi\bidata\service_instances\ssi \metadata\datamodel\customizations.

c) Extract the OBIEE reports that is the catalog file.

1. Log in to OBIEE analytics at http://<Server Name>:9502/analytics/saw.dll?bieehome.
2. Click on Catalog at the top right corner.
3. Select the folder you want to archive, and then click on the Archive button.



d) In the task box select archive to zip the files.

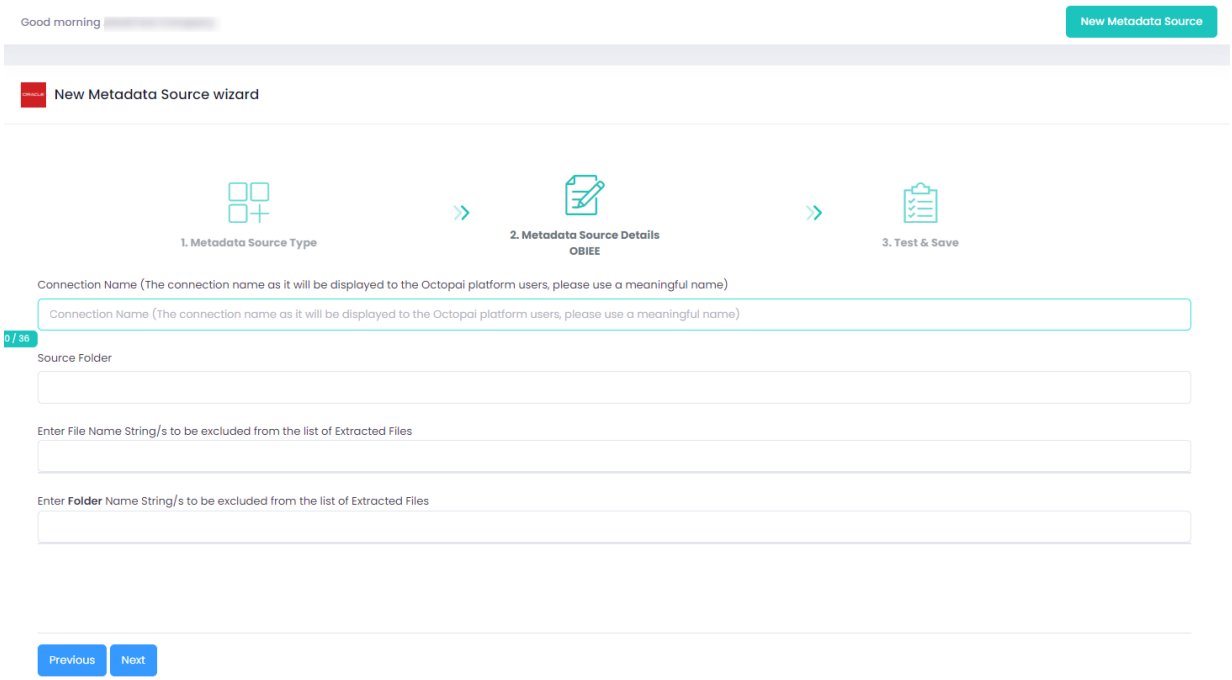


e) Deselect the Keep Permissions and Keep Timestamps checkboxes.
f) Click OK.
g) Save the file.

Note:

- The files will appear in the defined destination folder. Use this file as source for your Cloudera Data Lineage metadata OBIEE connection in the Octopai Client.
- Ensure that you have appropriate permissions to the path so that the Octopai Client can access the file with the user running it.

2. Set up the Oracle OBIEE metadata source on the Octopai Client.



3. Verify extraction results.

   After a successful run, confirm the metadata archive in the target folder. See Verify extracted metadata files.

If an error occurred during the extraction, see Troubleshoot connector extraction failures.

Power BI & Power BI Report Server

Learn how to configure Power BI and Power BI Report Server to integrate with Cloudera Data Lineage (formerly Octopai).

Note: Version supported: up to 2023

Tool Permissions Prerequisites

Note: Missing permissions could end up in broken lineages.

On-Prem Version:

- Local Folder - Read Permission for Cloudera Data Lineage Windows NT User on Power BI local folder - *.pbix files.
- Shared folder (SMB/Remote) - Full Control permission for Cloudera Data Lineage Windows NT User on Power BI shared folder (SMB/Remote) - *.pbix files.
- Power BI Web (Cloud App) - Azure Registered Application with relevant permissions (Please use the provided Power BI Settings for the Cloudera Data Lineage Extraction file) and a Power BI Premium user.
- **Note:** Multi-Factor Authentication (MFA) is not supported.

CLOUD Version:

- Power BI Web (Cloud App) - Client Secret Authentication (Open the Power BI Settings for Cloud Version Guide).
- Azure Registered Application with relevant permissions (Open the Power BI Web App Registration Guide).
- Tenant ID.
- Client Secret.

Power BI Report Server:

- 'Browse' and 'Content Manager' Role assigned for reports you would like extracted.

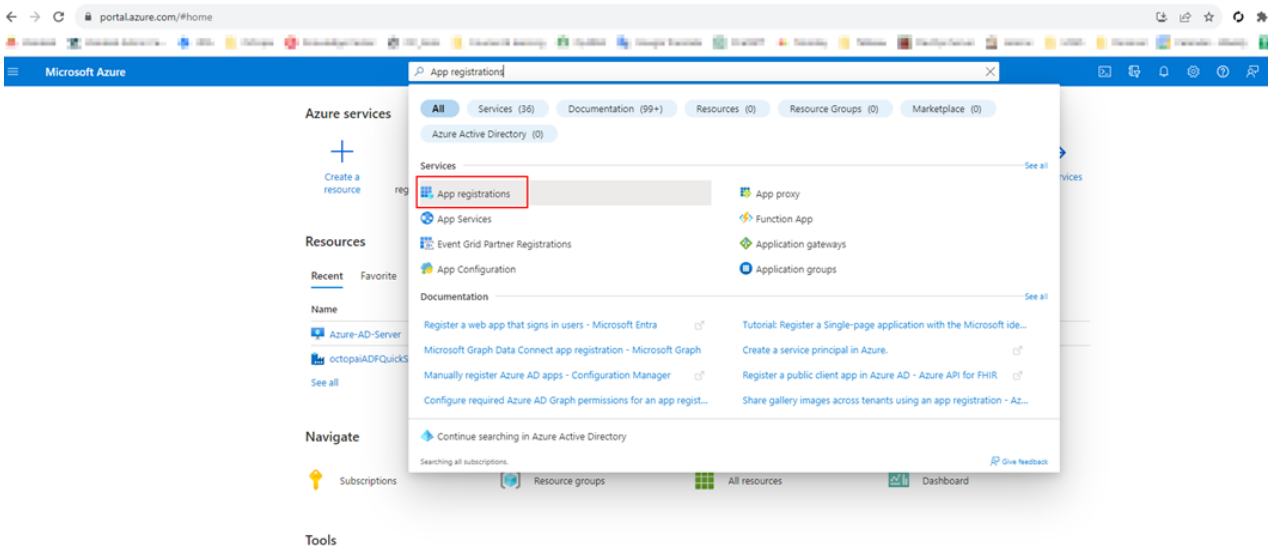
Permissions and configurations setup in Microsoft Azure

Important: Important clarification ahead of this guide:

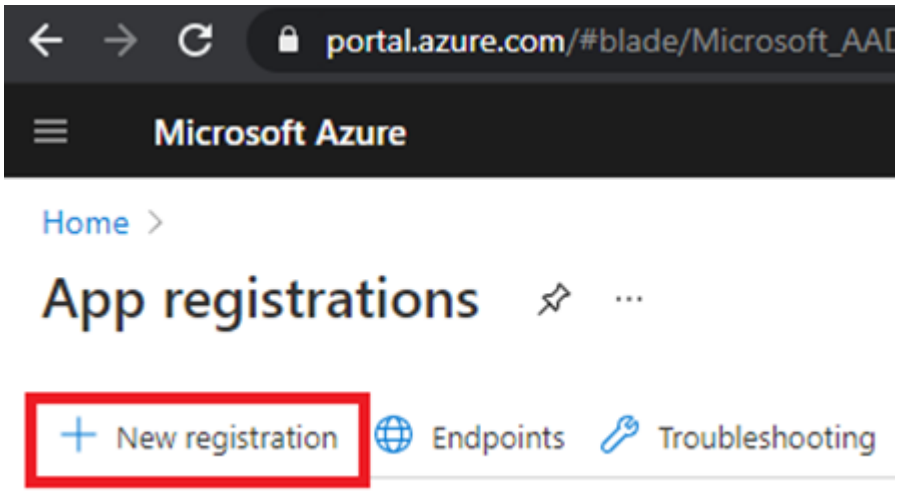
*** Changes in Azure and PowerBI portals typically take 15-30 minutes to apply but may take up to 24 hours in some cases.**

Go to https://portal.azure.com/

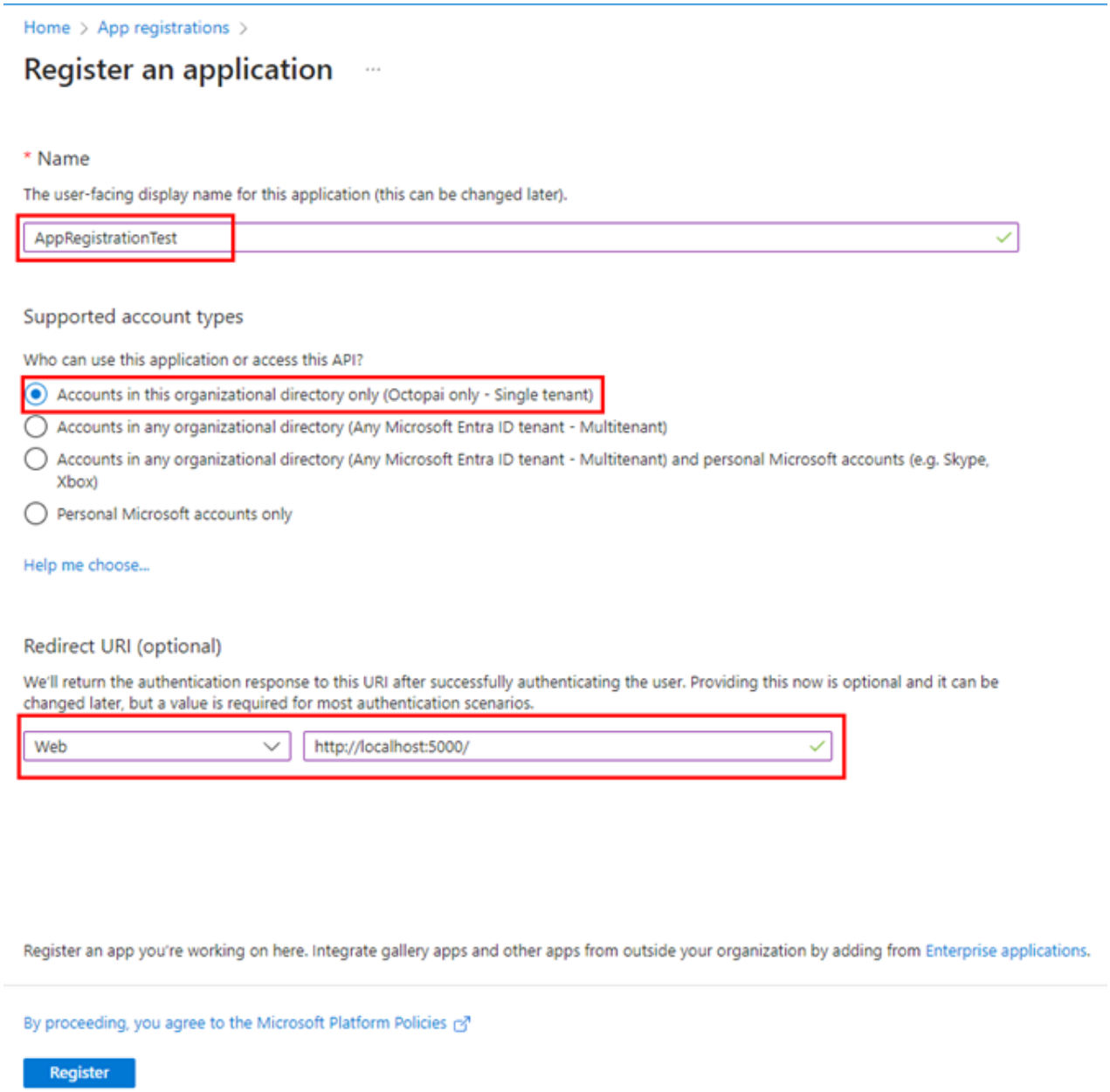
Then, go to App registrations.



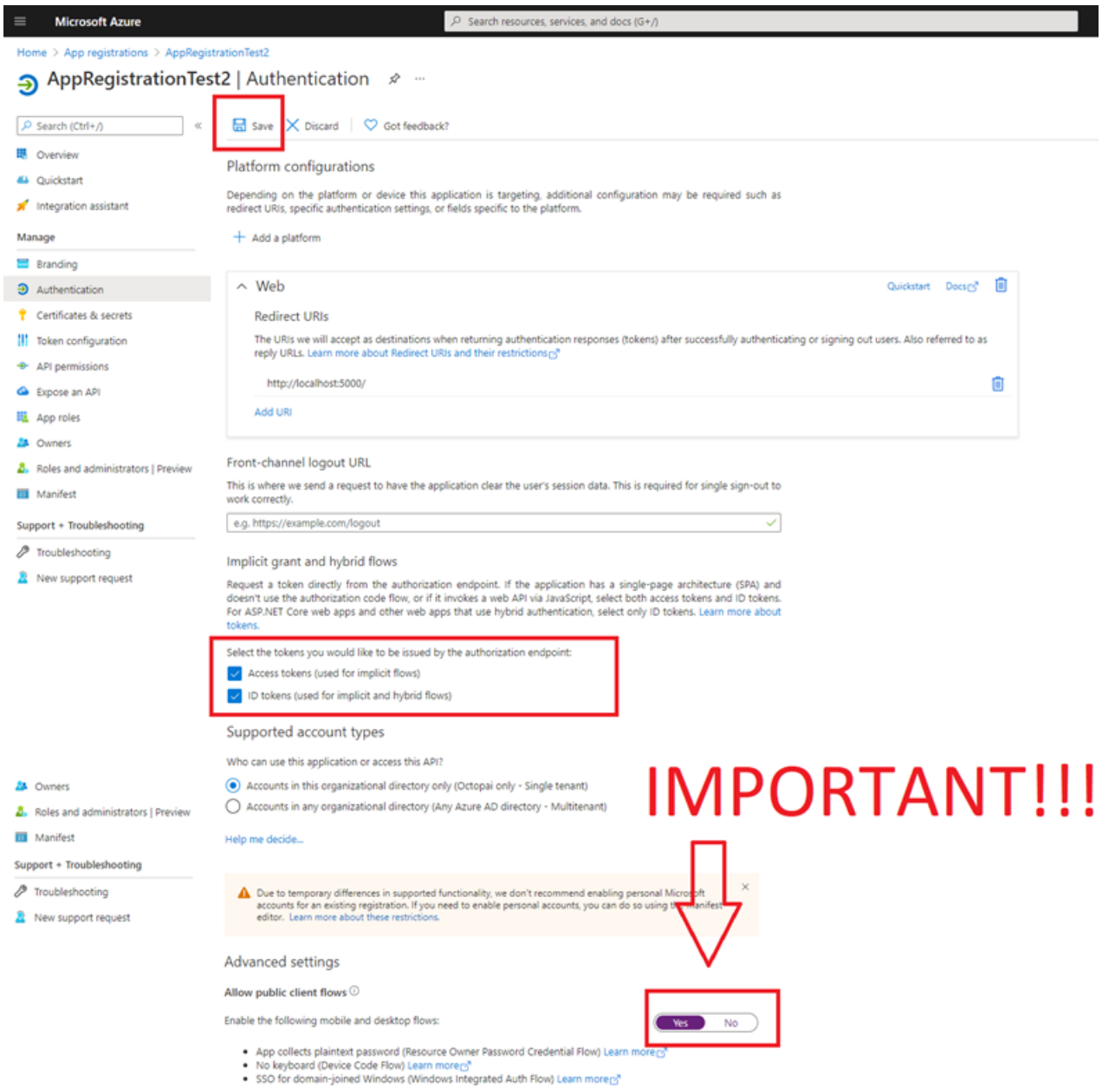
Click New Registration.



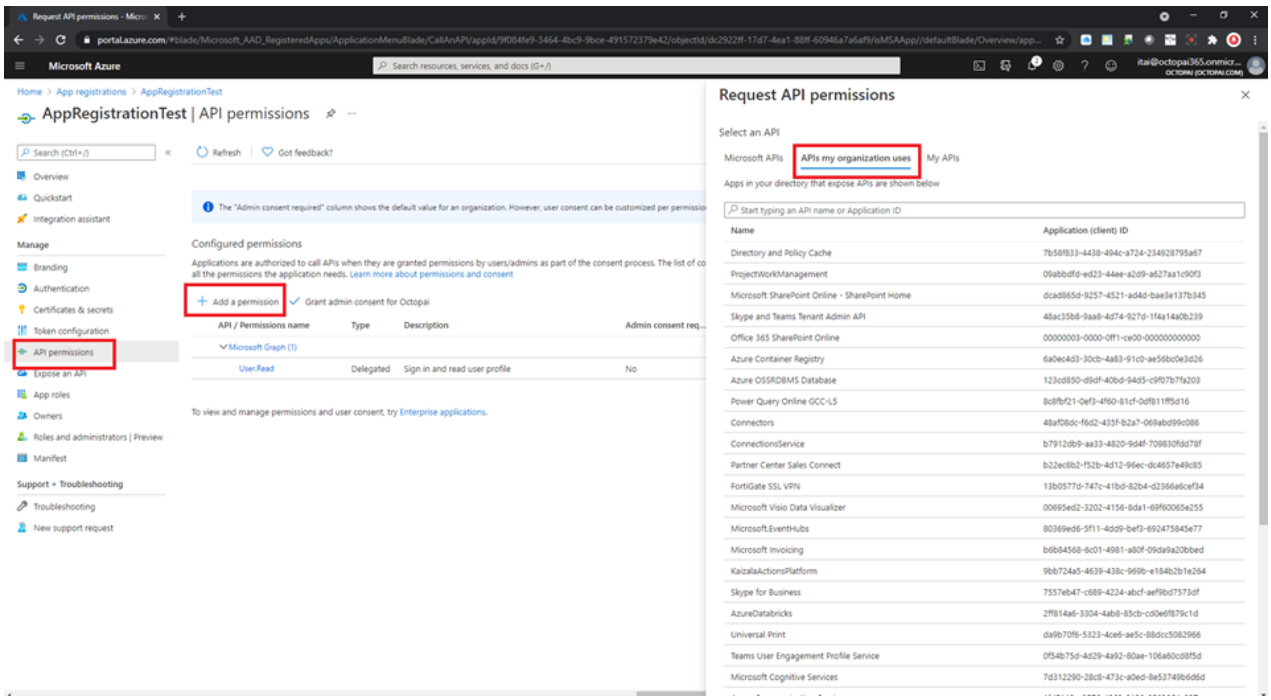
Configure as below and click Register.



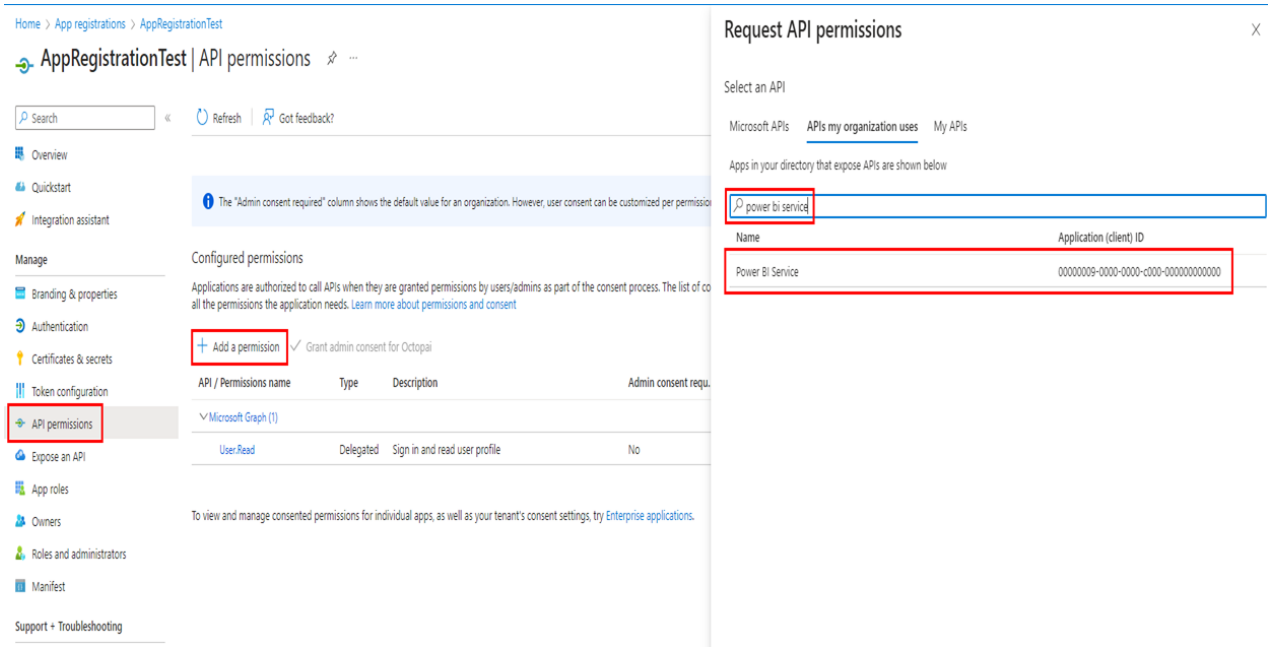
Go to Authentication tab > Configure as following > Click Save



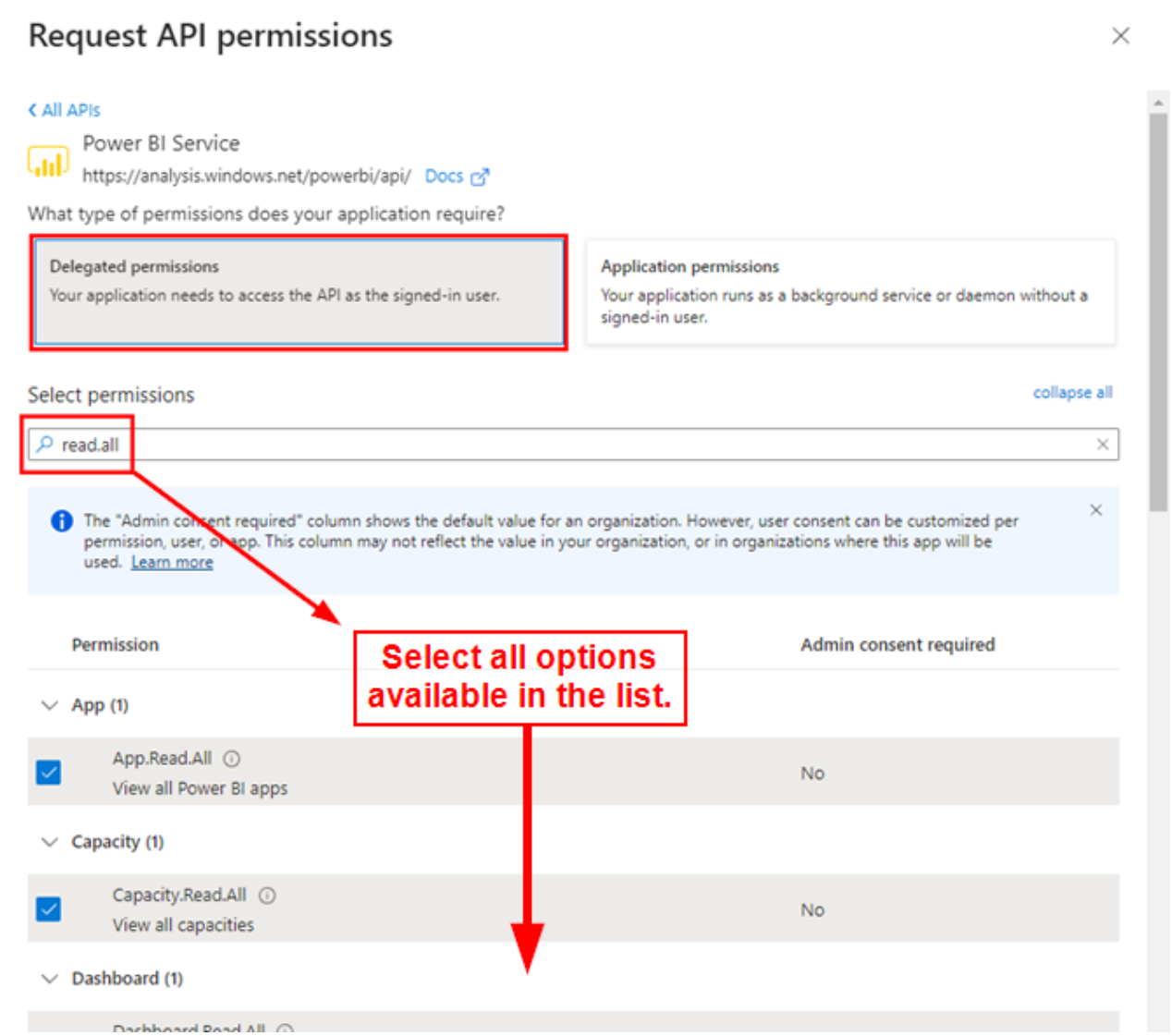
Click the API Permissions tab > Click Add permission > Click APIs my organization uses.



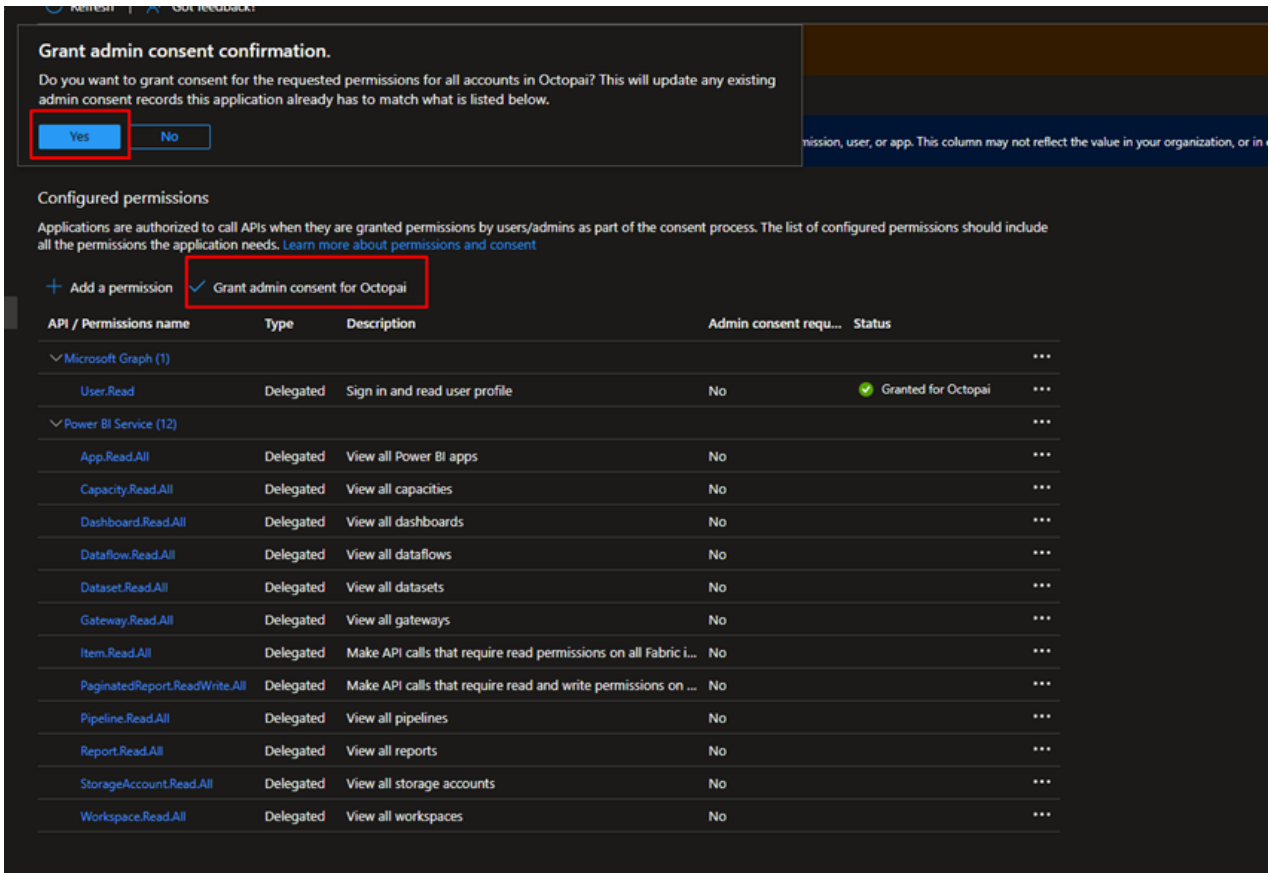
Search for Power bi Service > Click on Power bi Service.



Click Delegated permissions > Search “Read.All” and check ALL related permissions > click Add permission.



Click Grant admin consent for <Your domain name> > Click Yes



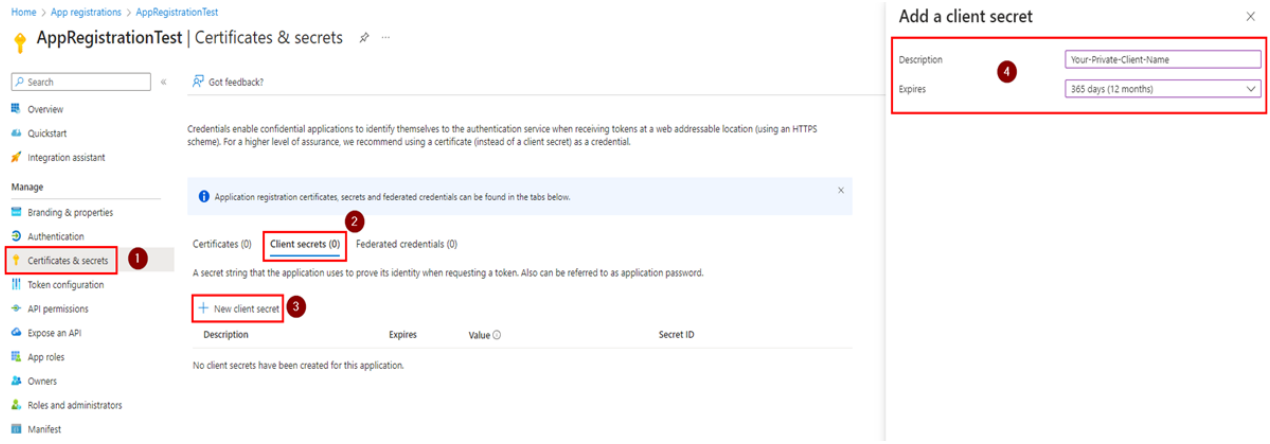
That summarizes the guidelines for creating the AppRegistration and granting it the proper permissions.

Security Group and PowerBI Portal configuration setup

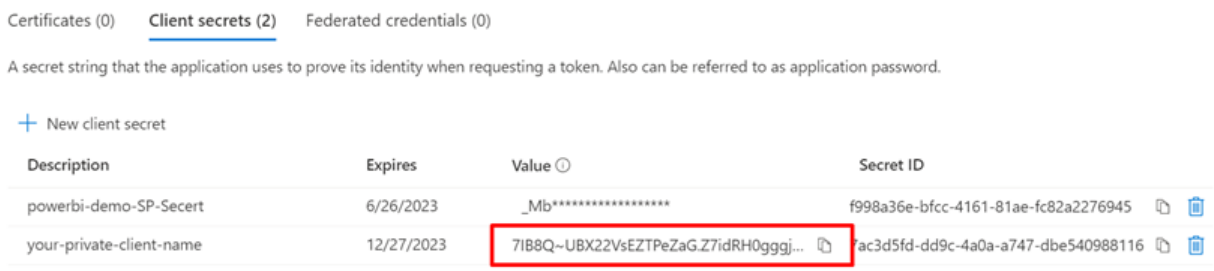
Important clarification ahead of this guide:

- Please make sure you completed **Permissions and configurations setup in Microsoft Azure**.
- The workspace license mode must be Premium Per User.

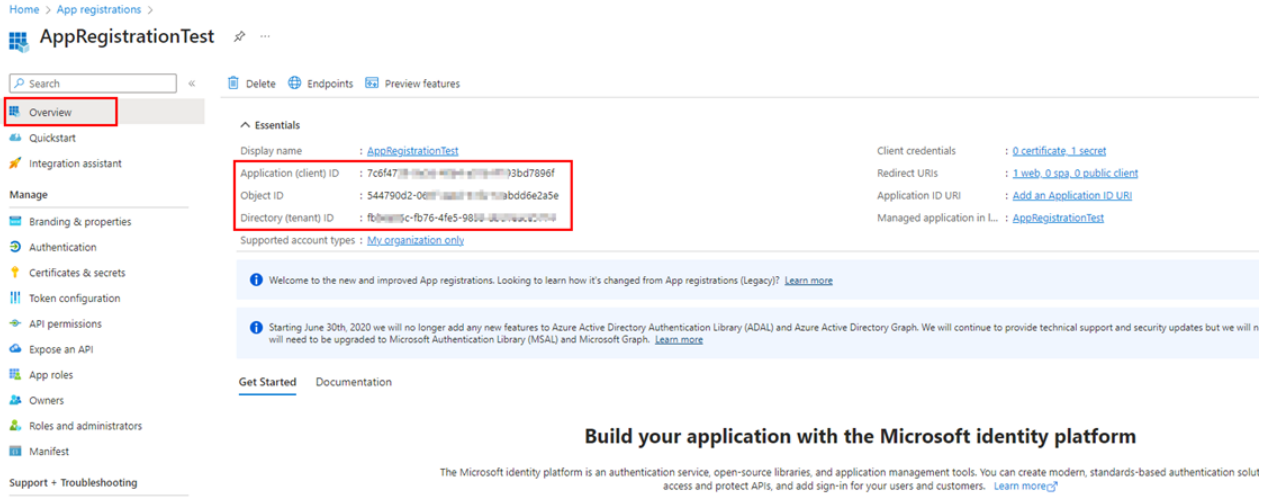
In the Azure portal, go to Certificates & secrets > New client secret.



Please copy the Secret value, we will need it later for the OC authentication.

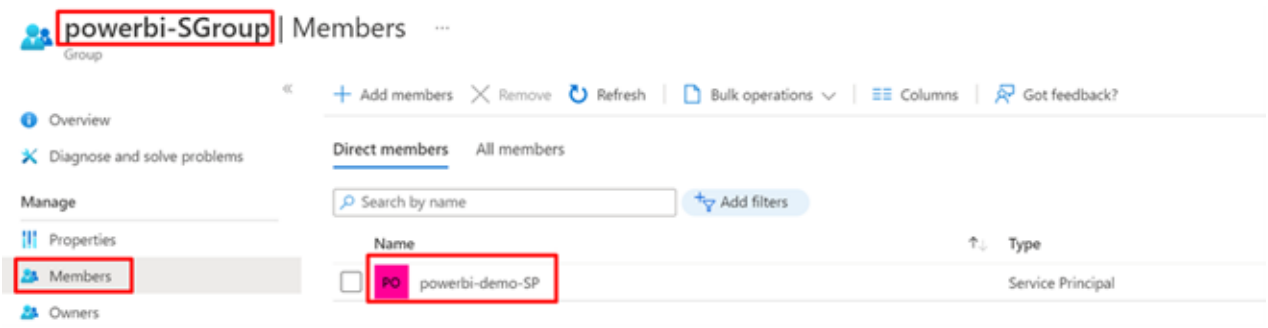


In the Overview tab, we will see the relevant information for the Octopai Client.



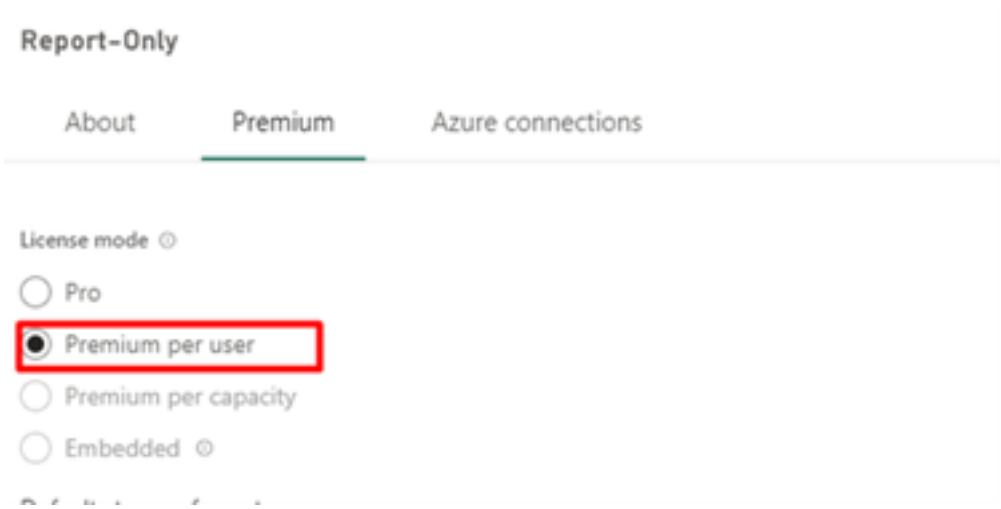
Create Security Group: We will need to create a group in https://portal.azure.com/ under groups OR ADD your Service Principal to an existing group.

(App Registration name = Principal service name)



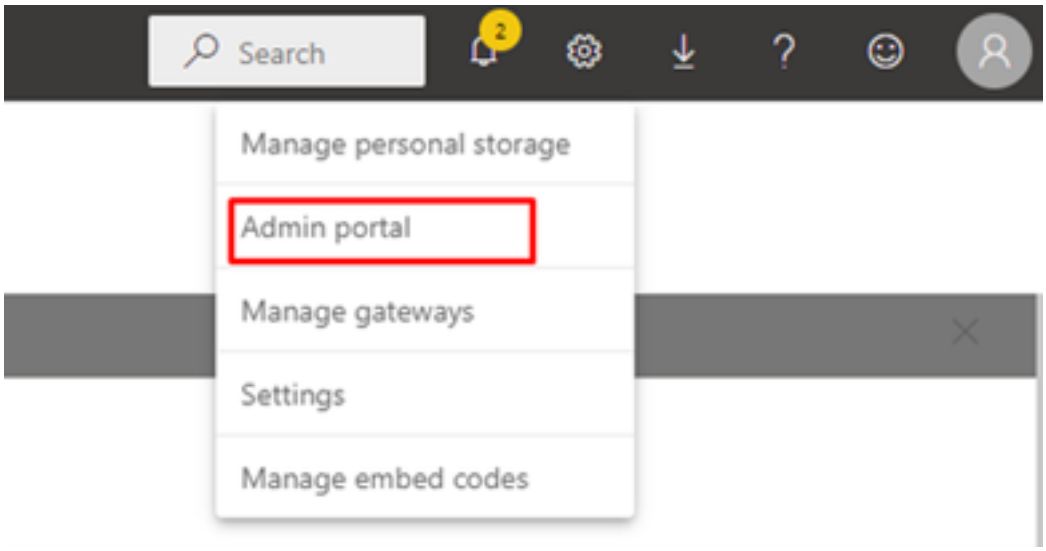
Power BI Settings (https://app.powerbi.com/home):

Required license type: Workspace License mode must be Premium Per User.



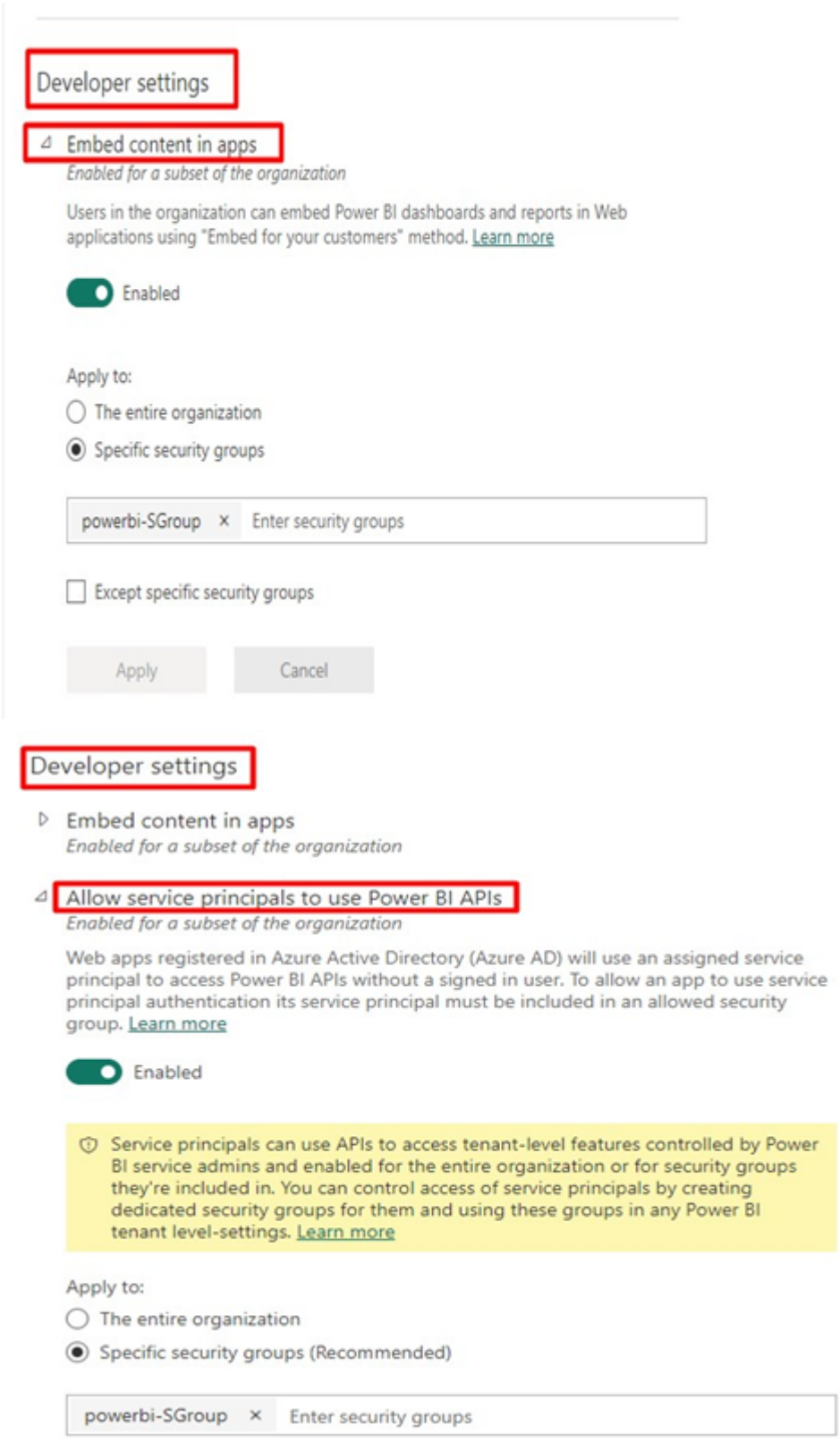
Admin Tenant settings: https://app.powerbi.com/home

Go to Settings - > Admin portal # Tenant settings.

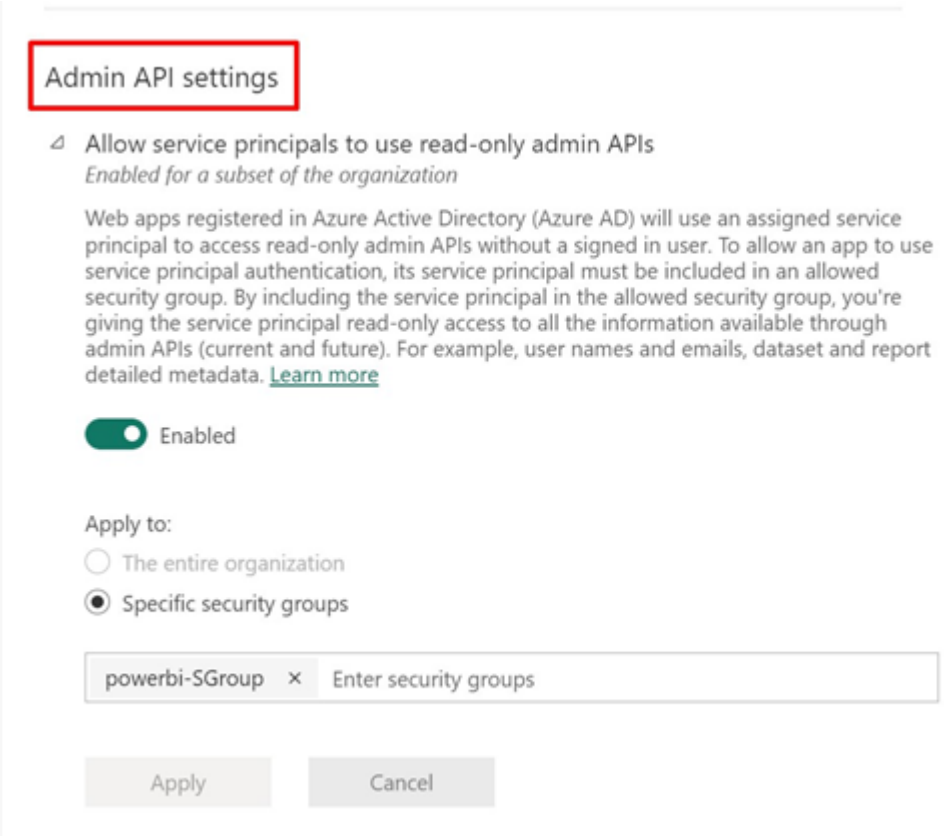


Note: Note: Changing the following steps takes approximately 15 min to be refreshed.

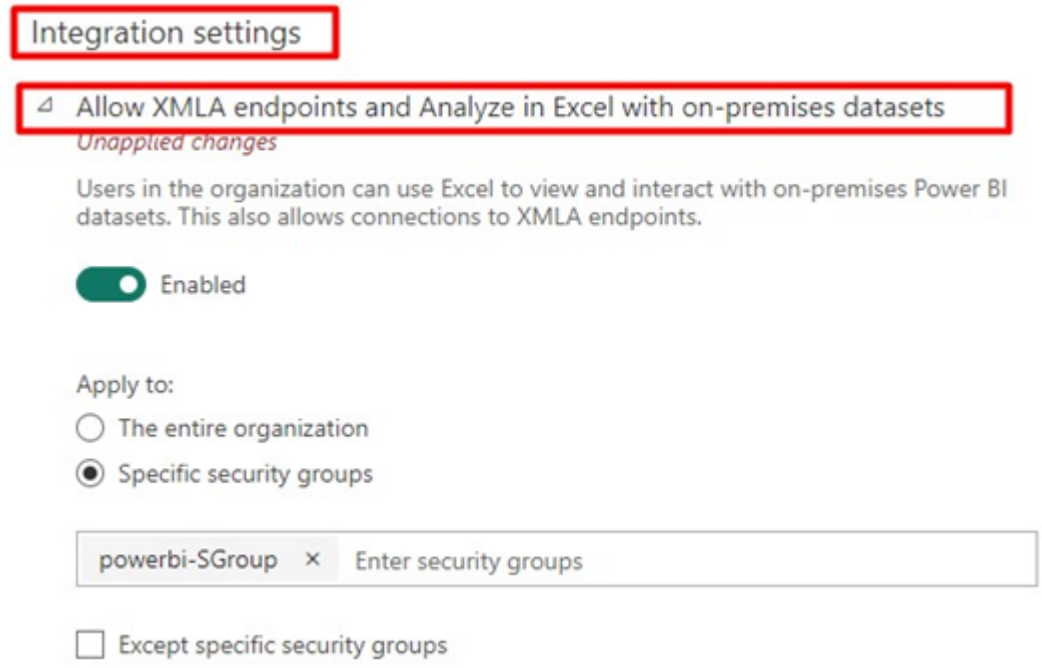
We need to enable the following: Under Tenant settings go to Developer Settings. Enable “Embed content in apps” and “Allow service principals to use Power BI APIs”. Choose the security group that the Service Principal is in.



Under Admin API settings enable and add the specific group that the Service Principal is in.

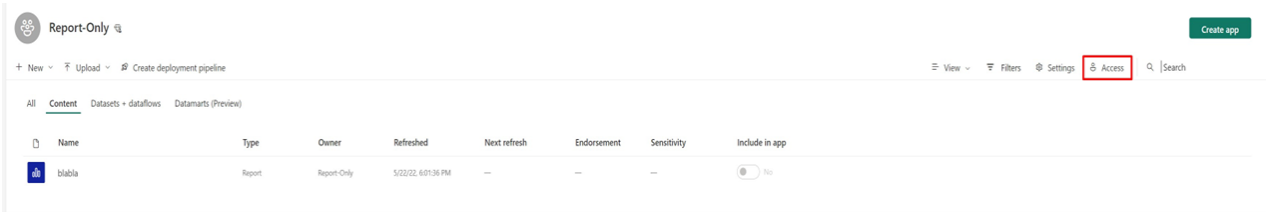


Under Integration settings enable and add the specific group that the Service Principal is in.

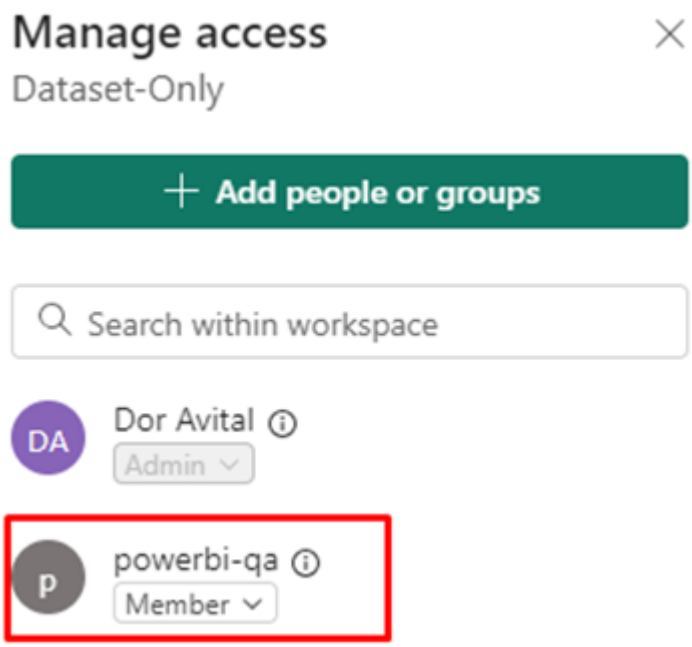


Workspace settings:

Go to your desired workspace and click Access.



Add the security group you've created or the Service Principal to the workspace and give it a Member role.



Note: Changes in Azure and PowerBI portals typically take 15-30 minutes to apply but may take up to 24 hours in some cases.

If you are experiencing trouble with extracting your metadata using the Octopai Client, please ensure you have followed all steps in this guide and wait for Microsoft to apply the new configurations across your organization.

Power BI Report Server

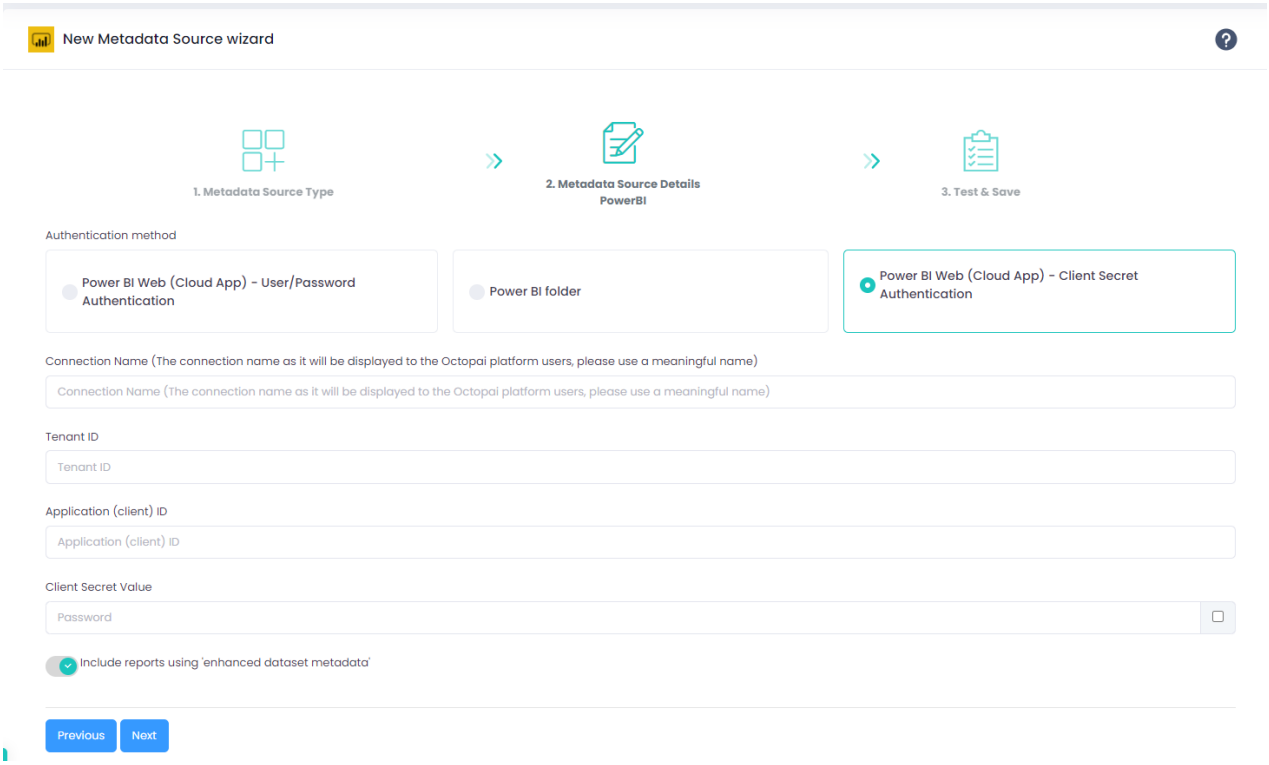
Please assign your (application) user with 'Browse' and 'Content Manager' roles for reports you would like to see in Cloudera Data Lineage.

For the Octopai Client connector, use the following URL structure:

http:\\comapny-url.com OR http:\\<IP>

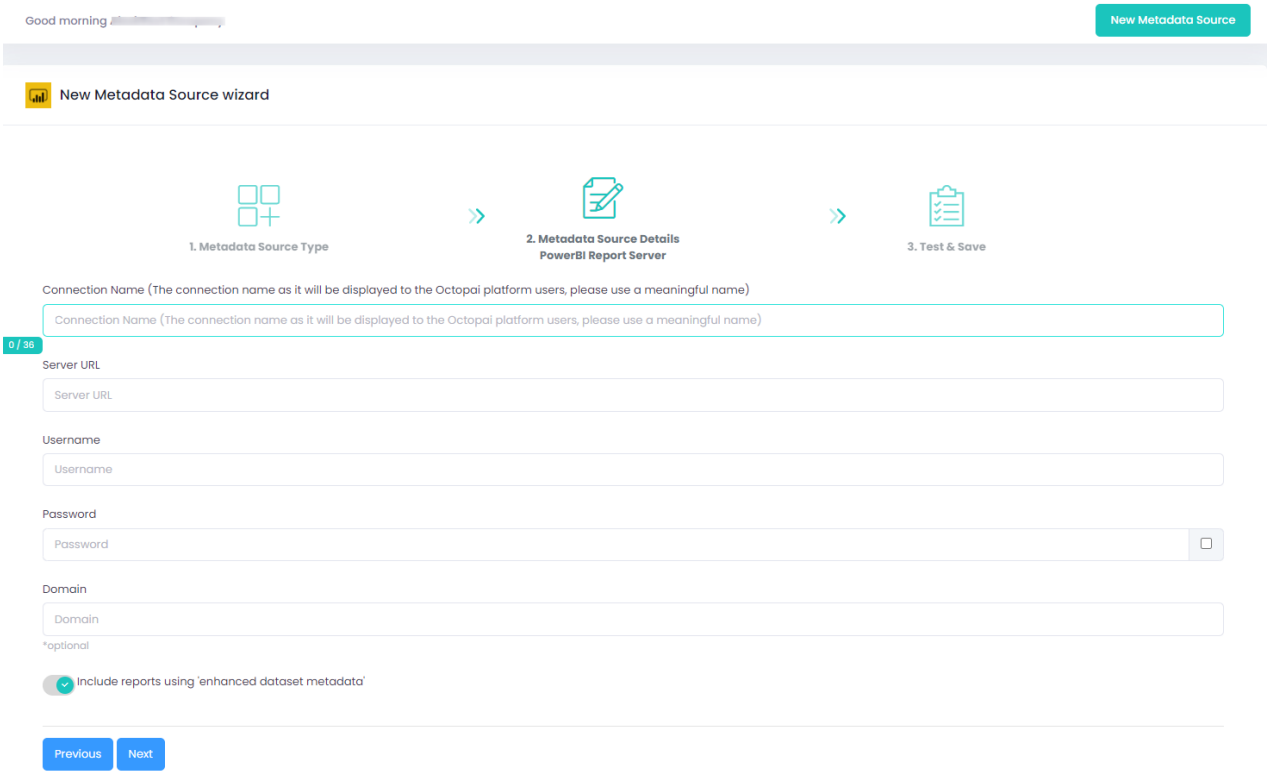
Setting up Power BI Metadata Source

Metadata Sources are set on the Octopai Client



1. Connection Name - Give a meaningful name, as it will be displayed to the Cloudera Data Lineage platform users.
2. Tenant ID - Available in the 'App registrations' section under the application you created.
3. Application (Client) ID - Available in the 'App registrations' section under the application you created.
4. Client Secret - Generated in 'App registrations > Certificates & secrets'.

Figure 32: Power BI Report Server



1. Connection Name - Give a meaningful name, as it will be displayed to the Cloudera Data Lineage platform users.
2. Server URL - The Report Server's URL.
3. Username - The user that is granted the needed roles.
4. Password - The user's password.
5. Domain (optional) - Only if the user is in a domain.

Power BI (Microsoft Fabric)

The Cloudera Data Lineage (formerly Octopai) Power BI (Microsoft Fabric) connector requires a Microsoft Entra service principal, Fabric admin tenant settings, Power BI Service API permissions, and Contributor workspace access.

Overview

Before you set up the Power BI (Microsoft Fabric) connector, ensure that you have the following:

- **Microsoft Entra application credentials:** a registered service principal with a tenant ID, application (client) ID, and client secret. See Application registration (service principal).
- **Fabric admin tenant settings:** settings that allow the service principal to call Fabric public APIs and access Fabric items. See Tenant settings (Fabric admin portal).
- **Power BI Service API permissions:** delegated API permissions for the service principal, including Dataset.Read .All and related scopes. See Tenant-level permissions.
- **Fabric workspace permissions:** the Contributor role assigned to the service principal on the target workspace. See Workspace permissions.

How this integration works

- Cloudera Data Lineage connects to a **single** Microsoft Fabric workspace, identified by **Workspace ID**, in your Microsoft Entra ID (Azure AD) tenant.
- Extraction uses **service principal (client credentials)** only.
- API access is scoped by workspace access.
- Cloudera Data Lineage calls Microsoft Fabric APIs and can also read Power BI content in the same workspace.



Note: API availability: The Power BI REST API must be available to your tenant. Availability depends on your Microsoft Fabric or Power BI license.

Required components on the customer side

You must have all of the following:

- **Microsoft tenant:** A Microsoft Entra tenant.
- **Microsoft Fabric or Power BI enabled:** Microsoft Fabric enabled in the tenant, or at a minimum enabled for the relevant capacity or workspace path.

Minimum permissions model

Because Cloudera Data Lineage connects to one workspace at a time, you must configure **two** permission layers:

1. **Tenant-level permissions** allows the service principal call Fabric and Power BI APIs.
2. **Workspace-level permissions** allows the service principal access the target workspace.

Application registration (service principal)

1. In the Azure portal, open Microsoft Entra ID App registrations , and select New registration.
2. Give the application a name, and select Accounts in this organizational directory only.

Note:

Redirect URI is optional and is not needed for client credentials.

3. Select Register.
4. From the application overview, record the Directory (tenant) ID and Application (client) ID.
5. Create a Client secret under Manage Certificates and secrets .

The Cloudera Data Lineage Power BI (Microsoft Fabric) connector authenticates using a Microsoft Entra service principal. The service principal is used for:

- Microsoft Fabric REST API access
- Power BI Service API access for workspace and item metadata

You must record the following values for configuration in Cloudera Data Lineage:

- Tenant ID
- Application (client) ID
- Client Secret Value

Tenant settings (Fabric admin portal)

In the Fabric admin portal, enable the following settings:

1. Open Microsoft Fabric.
2. Select Settings, and then select Admin portal.
3. Open Tenant settings.
4. Find and enable Service principals can call Fabric public APIs.
5. Find and enable Allow apps to access Fabric items.

Tenant-level permissions

In the app registration, go to Manage API permissions , select **Power BI Service**, and add the following delegated permissions:

- Dataflow.Read.All
- Dataset.Read.All
- Report.Read.All
- SemanticModel.Read.All
- Workspace.Read.All

Workspace permissions

Enabling tenant settings does not grant workspace access. You must also assign the service principal to the target workspace with the Contributor role. You need Contributor access for complete metadata extraction coverage, including:

- Reports
- Datasets and semantic models
- Dataflows
- Tables, measures, and lineage definitions

For metadata extraction (tables, measures, and lineage), you need Build and XMLA access, available with Contributor or equivalent permissions. The Viewer role is not sufficient for full metadata extraction.

Important:

Cloudera recommends Contributor access for all Cloudera Data Lineage Power BI (Microsoft Fabric) integrations. Lower-privilege roles result in incomplete metadata extraction.

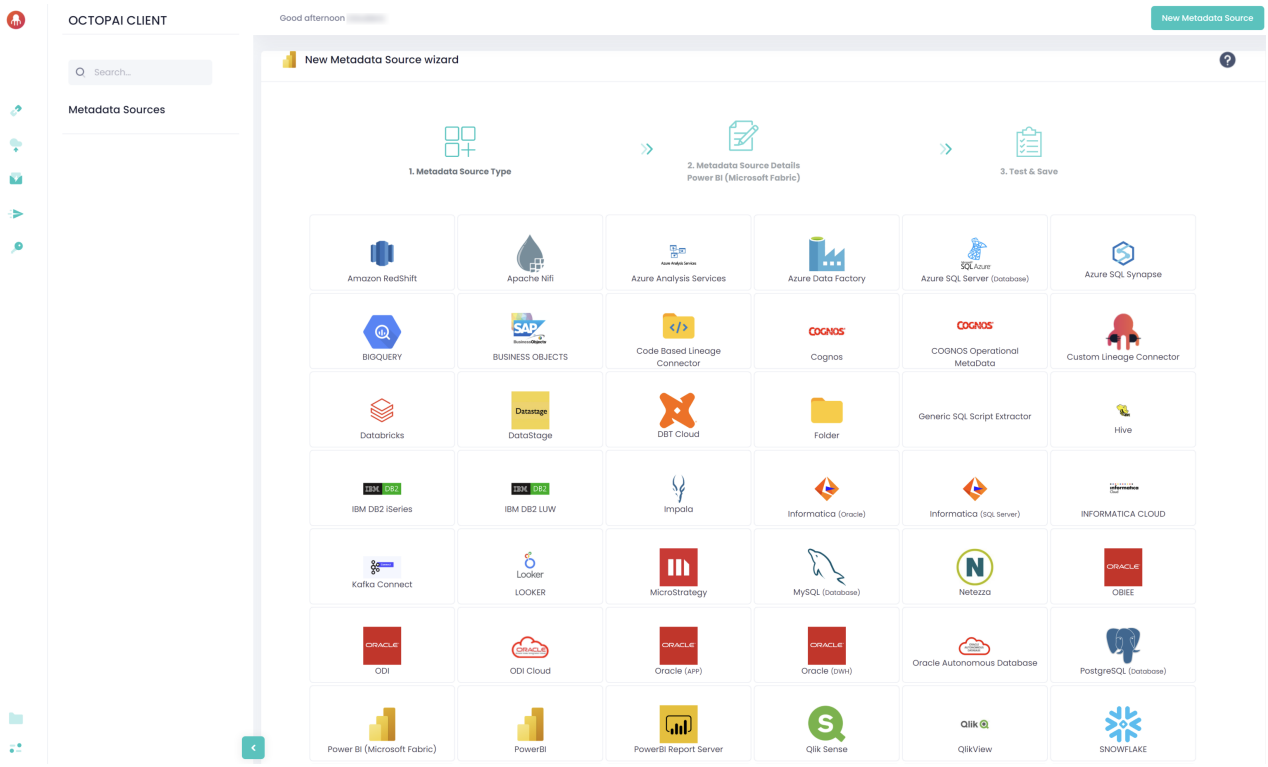
Note:

Cloudera Data Lineage uses Contributor access for read-only metadata extraction. Extraction includes workspace and item metadata and item definitions required for discovery and analysis. Cloudera Data Lineage does not create, update, delete, publish, rebind, or change permissions in the workspace.

Setting up Power BI (Microsoft Fabric) metadata source

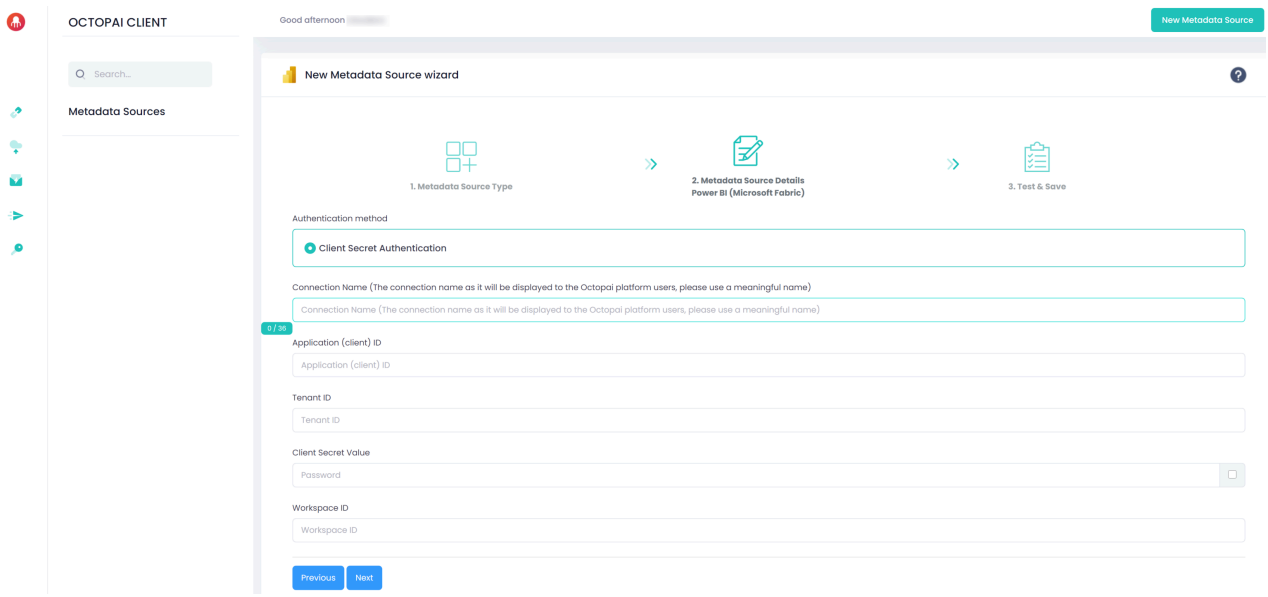
Configure the metadata source on the Octopai Client. In the New Metadata Source wizard, on Metadata Source Type, select Power BI (Microsoft Fabric).

Figure 33: Selecting Power BI (Microsoft Fabric) as the metadata source type



On Metadata Source Details, select Client Secret Authentication, and enter the following:

Figure 34: Power BI (Microsoft Fabric) metadata source details



- Connection Name: Enter a meaningful name. The name is displayed to Cloudera Data Lineage platform users.
- Application (client) ID
- Tenant ID
- Client Secret Value
- Workspace ID

Note:

To locate your Workspace ID in Microsoft Fabric:

1. Open Microsoft Fabric.
2. Open the target workspace.
3. In your browser address bar, locate the URL. The URL follows this pattern:

```
https://app.fabric.microsoft.com/groups/[***WORKSPACE ID***]/list?experience=power-bi
```

4. Copy the value that appears after /groups/. This value is your Workspace ID.

Qlik Sense

Configure the Cloudera Data Lineage (formerly Octopai) Qlik Sense connector, including metadata setup, user allocation, and required permissions.

Note: Version supported: 2023

Tool Permissions Prerequisites

Warning: Missing permissions could end up in broken lineages.

- Login permissions from the server running the Octopai Client to the Qlik Sense Server (User and password with permissions to the Qlik Sense REST API, which needs to be enabled, and permissions to all apps you wish to extract).

- Read Permission for Cloudera Data Lineage Windows NT User on Qlik Sense Log files folder (Default location for log files:C:\ProgramData\Qlik\Sense\Log\Script).
- Allocated licensed user with permissions to streams you would like analyzed by Cloudera Data Lineage, use the following steps if needed:
  - Create Custom property Creating a custom property # Qlik Sense for administrators
    - Name ‘Octopai_Group'
    - Resource type - ‘Users’, ‘Streams’
    - Value 'Octopai’
  - Assign Property ‘Octopai_ Group' and value 'Octopai’ to each stream you would like Cloudera Data Lineage to analyze.
  - Assign Property ‘Octopai_ Group' and value 'Octopai’ to the user that will be used for extraction
  - Add Security rule Creating security rules # Qlik Sense for administrators
    - Name - Octopai_Stream_Rule
    - Resource filter - Stream_*
    - Action - Read
    - Condition - ((user.@Octopai_Group=resource.@Octopai_Group))
  - Proxy port values most contain default values Editing proxies # Qlik Sense for administrators
    - Service listen port HTTPS (default) - 443
    - Authentication listen port - 4244
    - REST API listen port - 4243

How to allocate a user in Qlik Sense



Note: The allocated user is necessary for a successful extraction from Qlik Sense.

Prerequisites:

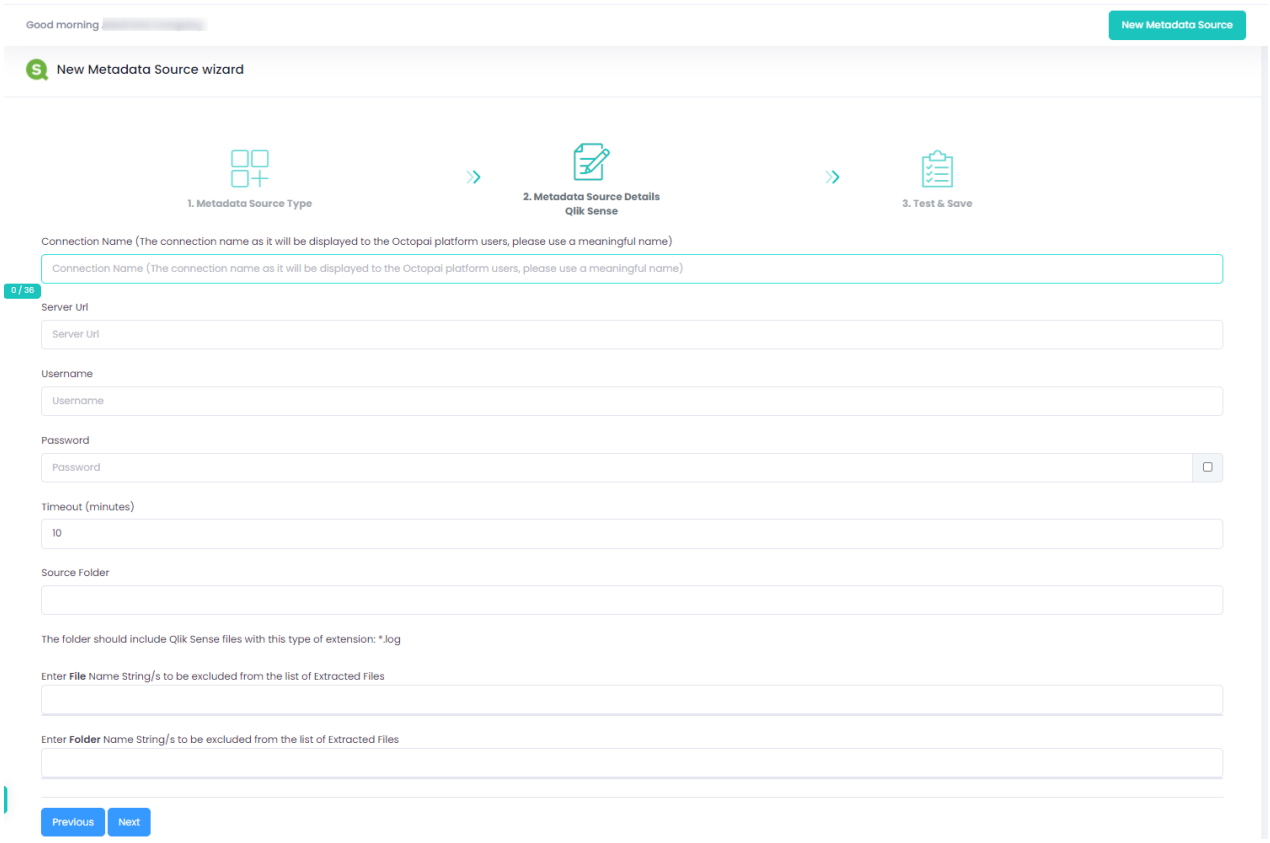
- Login permissions from the server running the Octopai Client to the Qlik Sense Server (User and password with permissions to the Qlik Sense REST API, which needs to be enabled and permissions to all apps you would like to extract).
- Read Permission for Cloudera Data Lineage Windows NT User on Qlik Sense Log files folder.
- Default location for log files: C:\ProgramData\Qlik\Sense\Log\Script.
- Dedicated Header Authentication: Configuring header authentication.

Grant permission to all relevant streams to an allocated user:

- Create Custom property: Creating a custom property.
  - Name ‘Octopai_Group'
  - Resource type - ‘Users’, ‘Streams’
  - Value 'Octopai’
- Assign Property ‘Octopai_ Group' and value 'Octopai’ to each stream you would like Cloudera Data Lineage to analyze.
- Assign Property ‘Octopai_ Group' and value 'Octopai’ to the user that will be used for extraction
- Add Security rule: Creating security rules.
- Name - Octopai_Stream_Rule
- Action - Read
- Condition - User@Octopai_Group = #Stream@Octopai_Group

Setting up Qlik Sense Metadata Source

Metadata Sources are set on the Octopai Client

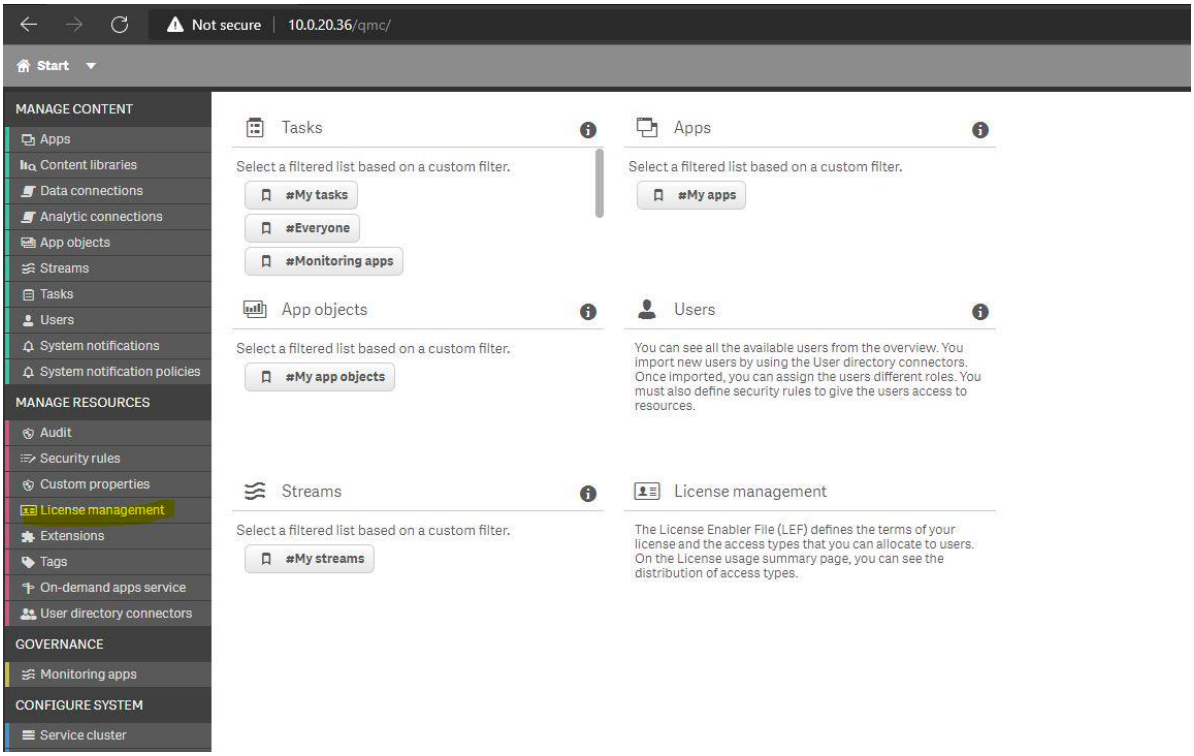


Configure Qlik Sense for Data Extraction

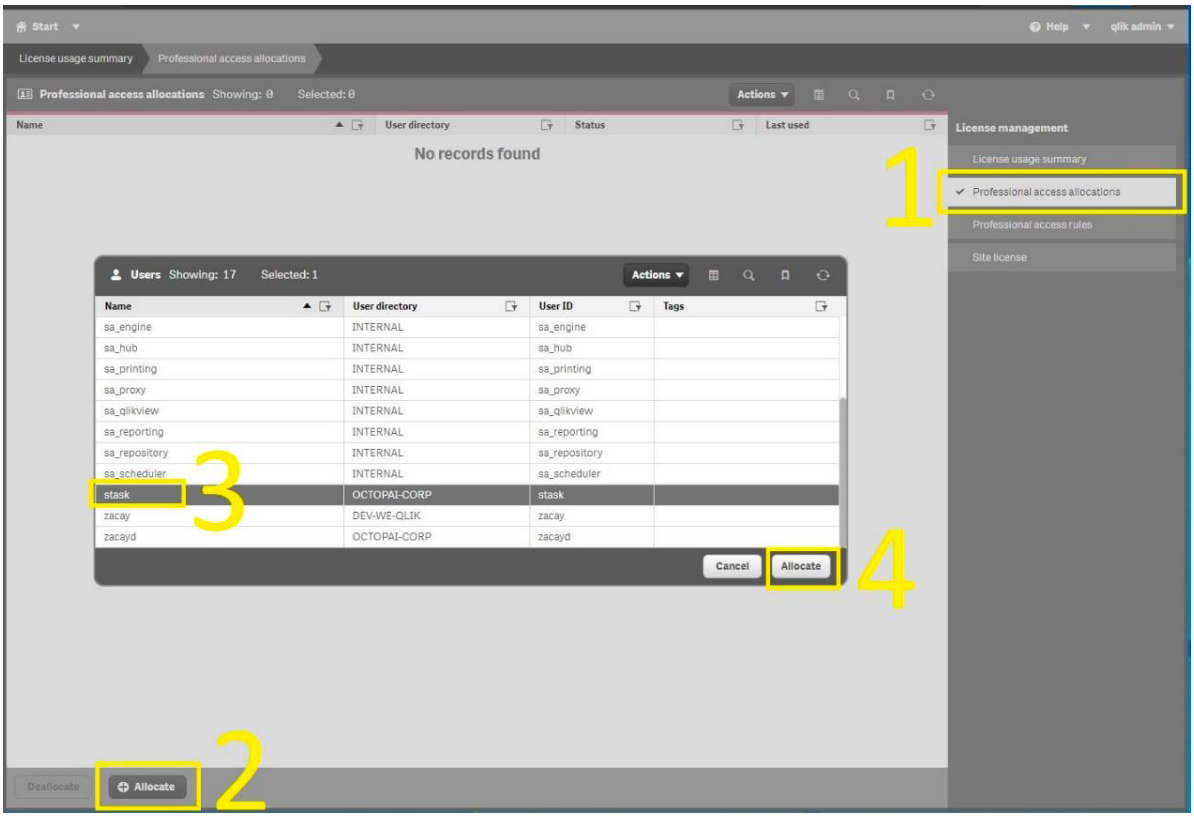
Learn how to configure Qlik Sense for data extraction with Cloudera Data Lineage (formerly Octopai).

Procedure

1. Assign a license to the Qlik user used for the data extraction.
   a) Open the Qlik Management Console at https://qlik-server-address:/qmc.
   b) Navigate to the License management menu.

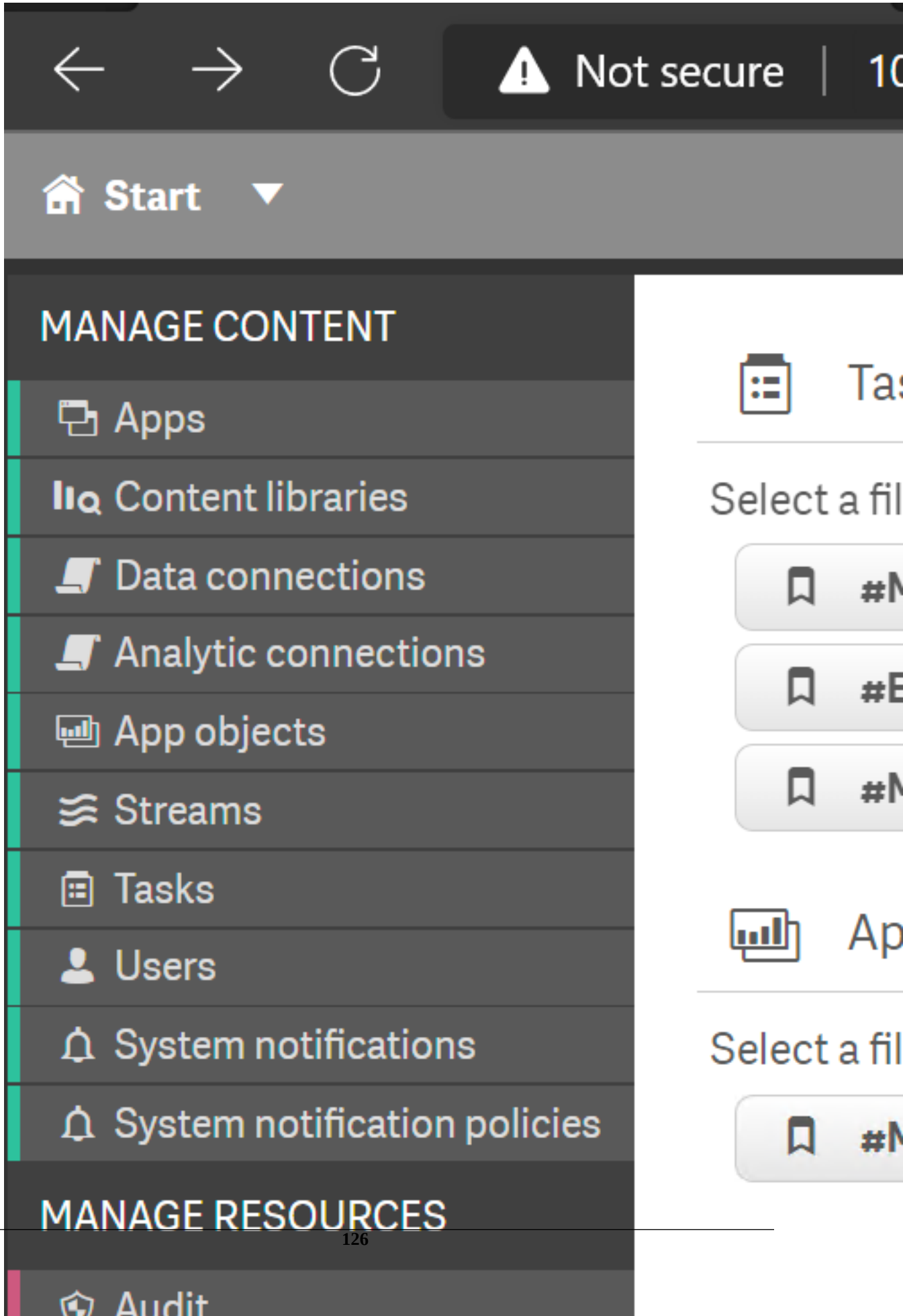


c) Click Professional license allocation, select the target user, and click Allocate.

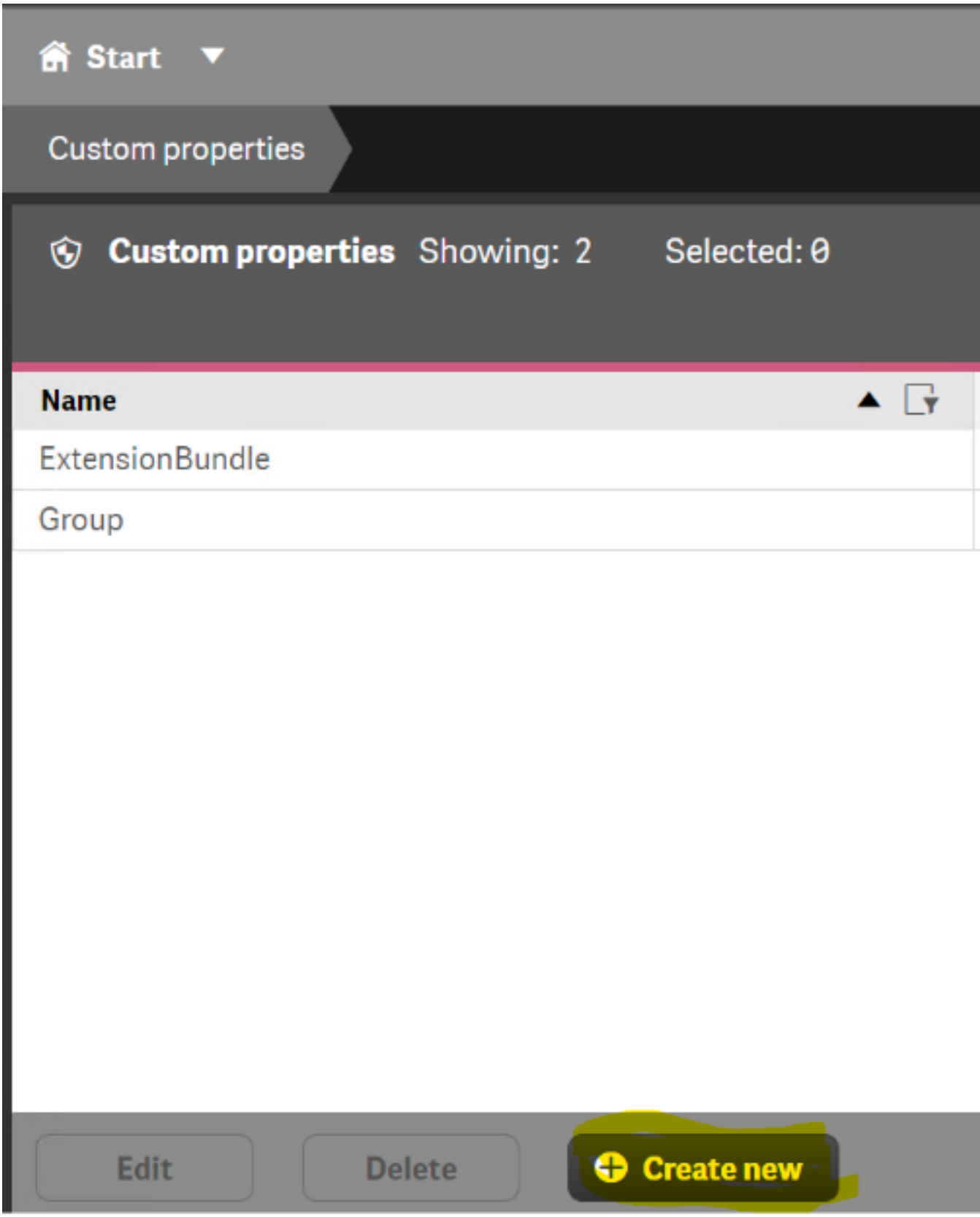


2. Create a custom property to tag specific users and streams.

a) Go to the Custom properties menu.


Not secure
Start
MANAGE CONTENT
Apps
Content libraries
Data connections
Analytic connections
App objects
Streams
Tasks
Users
System notifications
System notification policies
MANAGE RESOURCES
Audit

b) Click Create new and add the following properties:



Name

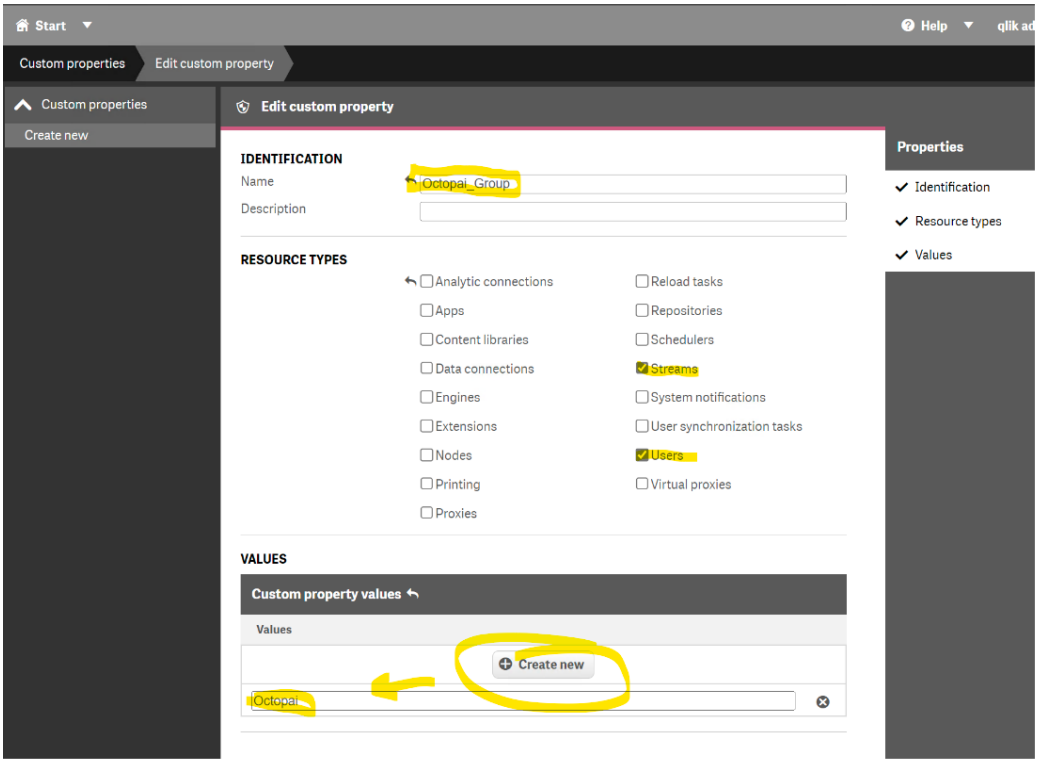
Octopai_Group

Resource types

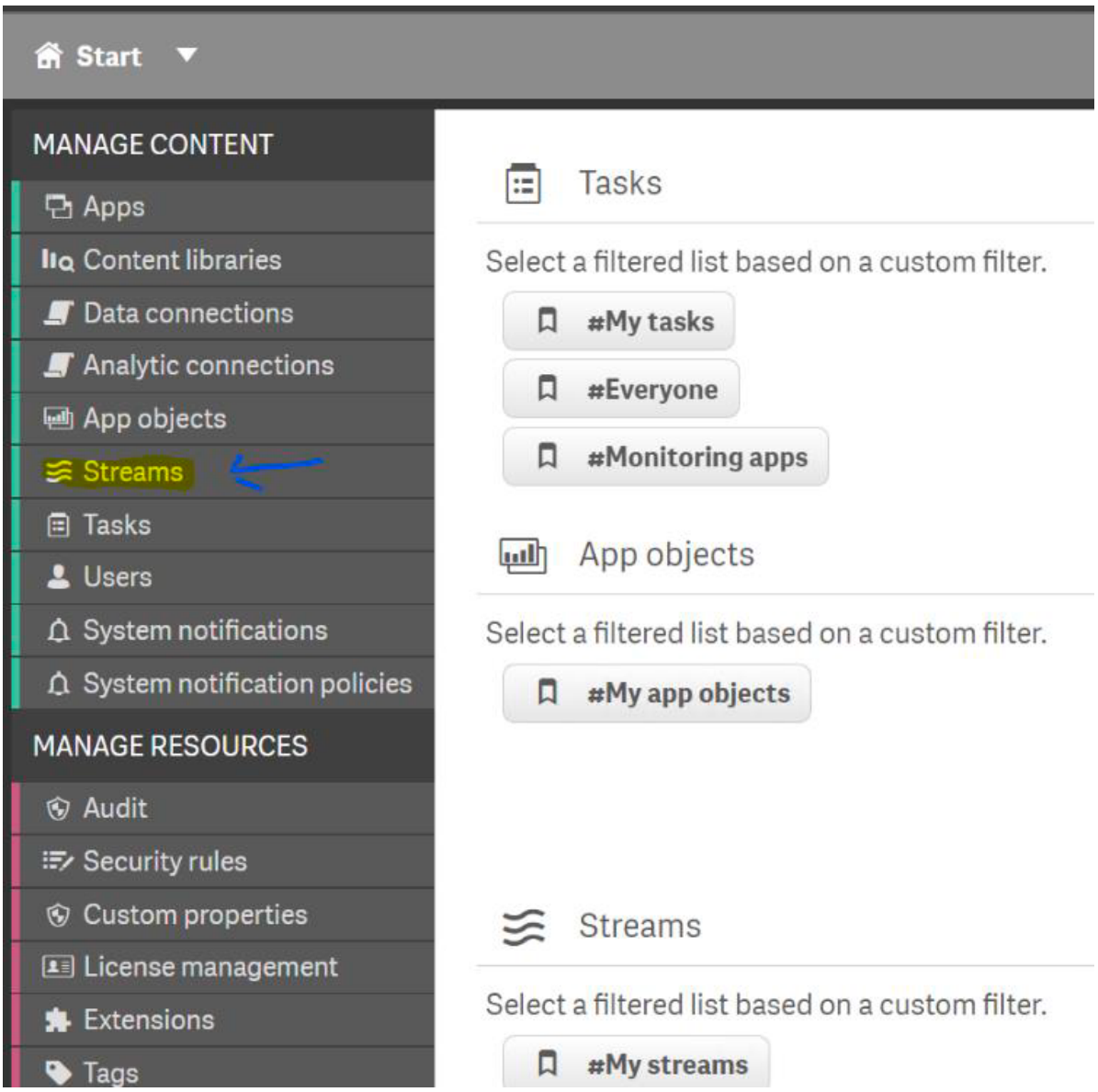
Streams, Users

Value

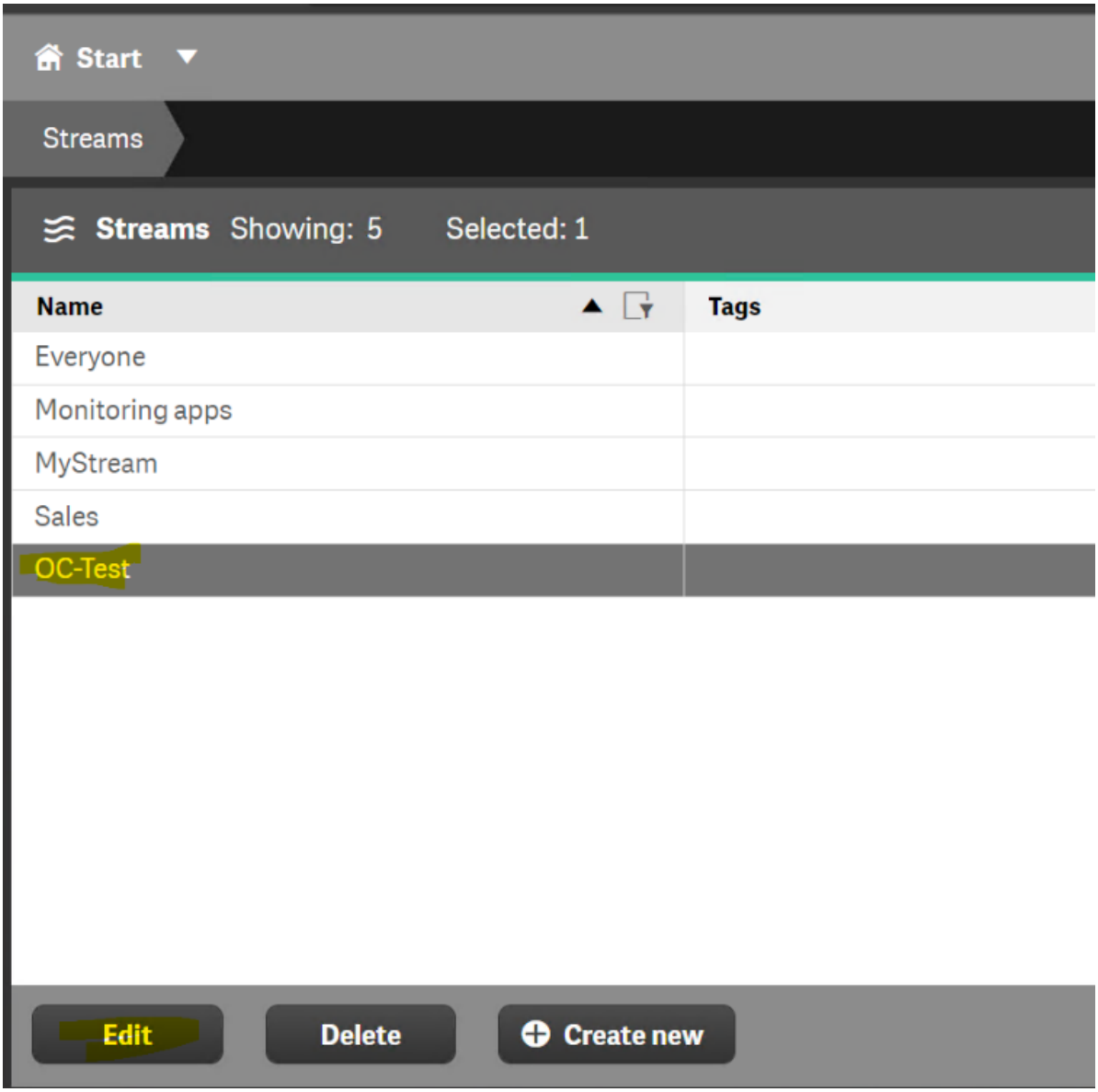
Octopai



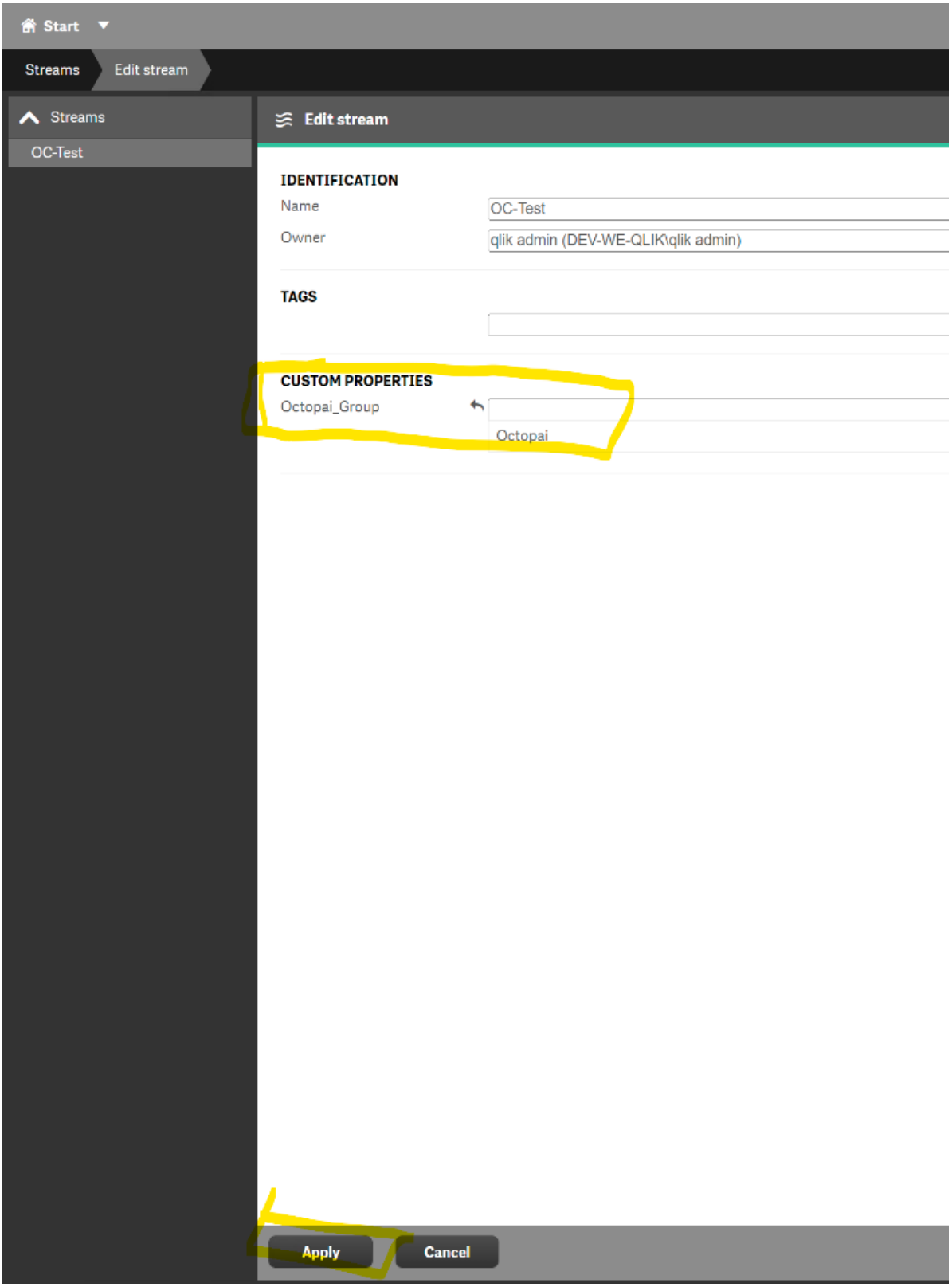
3. Assign the custom property to the targeted streams.
   a) Go to the Streams menu.



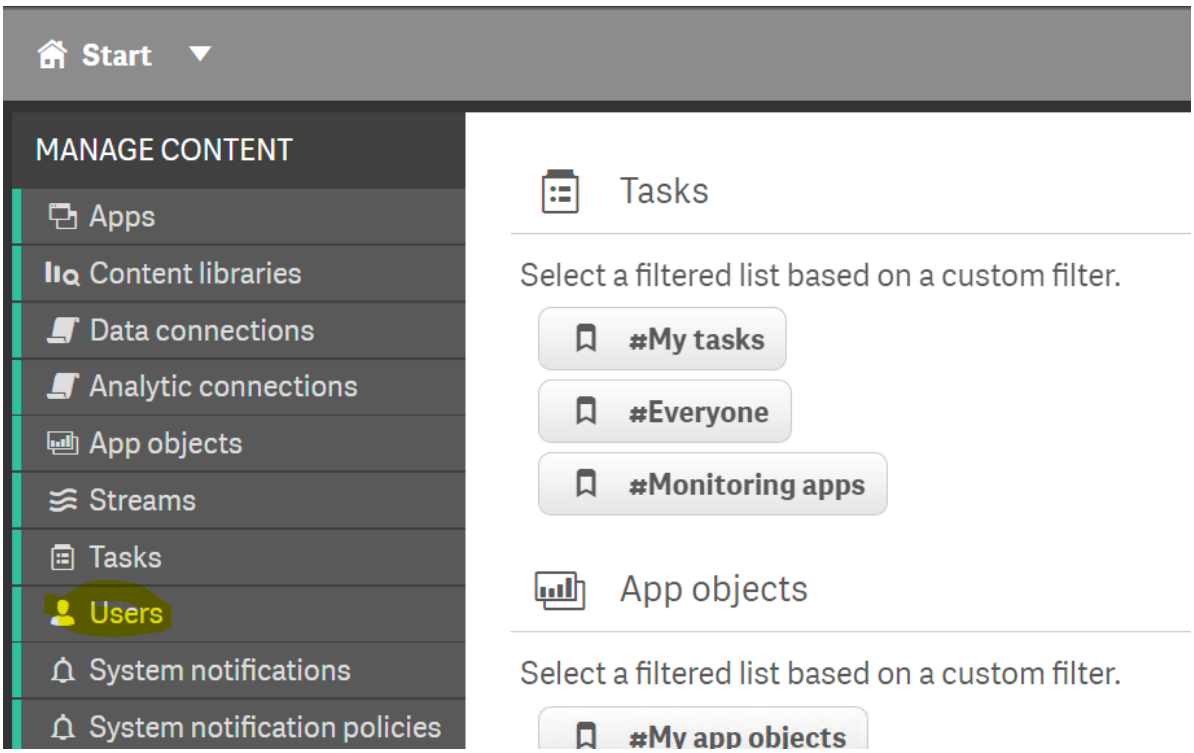
b) Select the desired stream and click Edit.



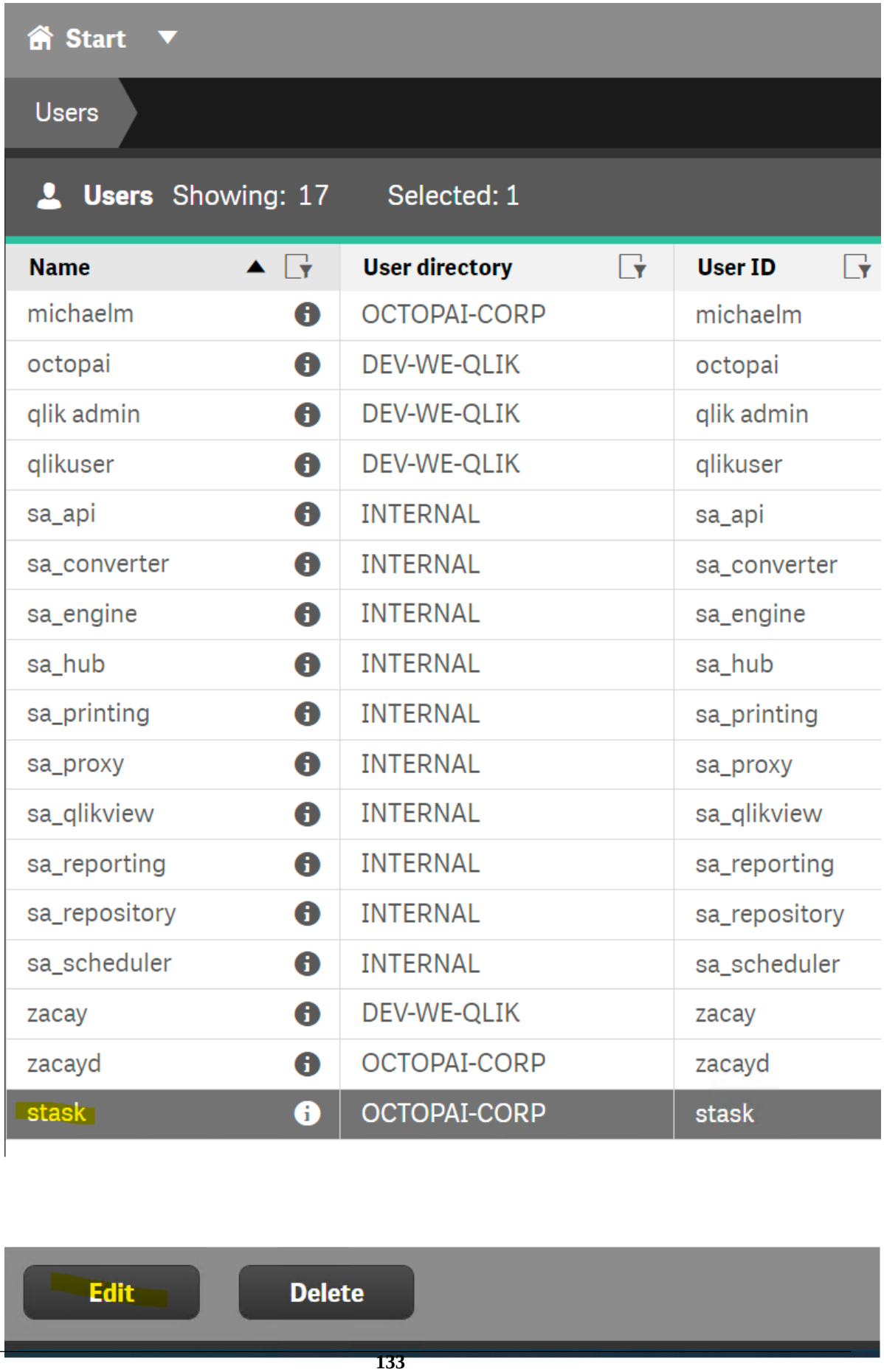
c) Under Custom properties, find Octopai_Group and select Octopai from the dropdown menu.



4. Assign the custom property to the user used for extraction.
   a) Go to the Users menu.



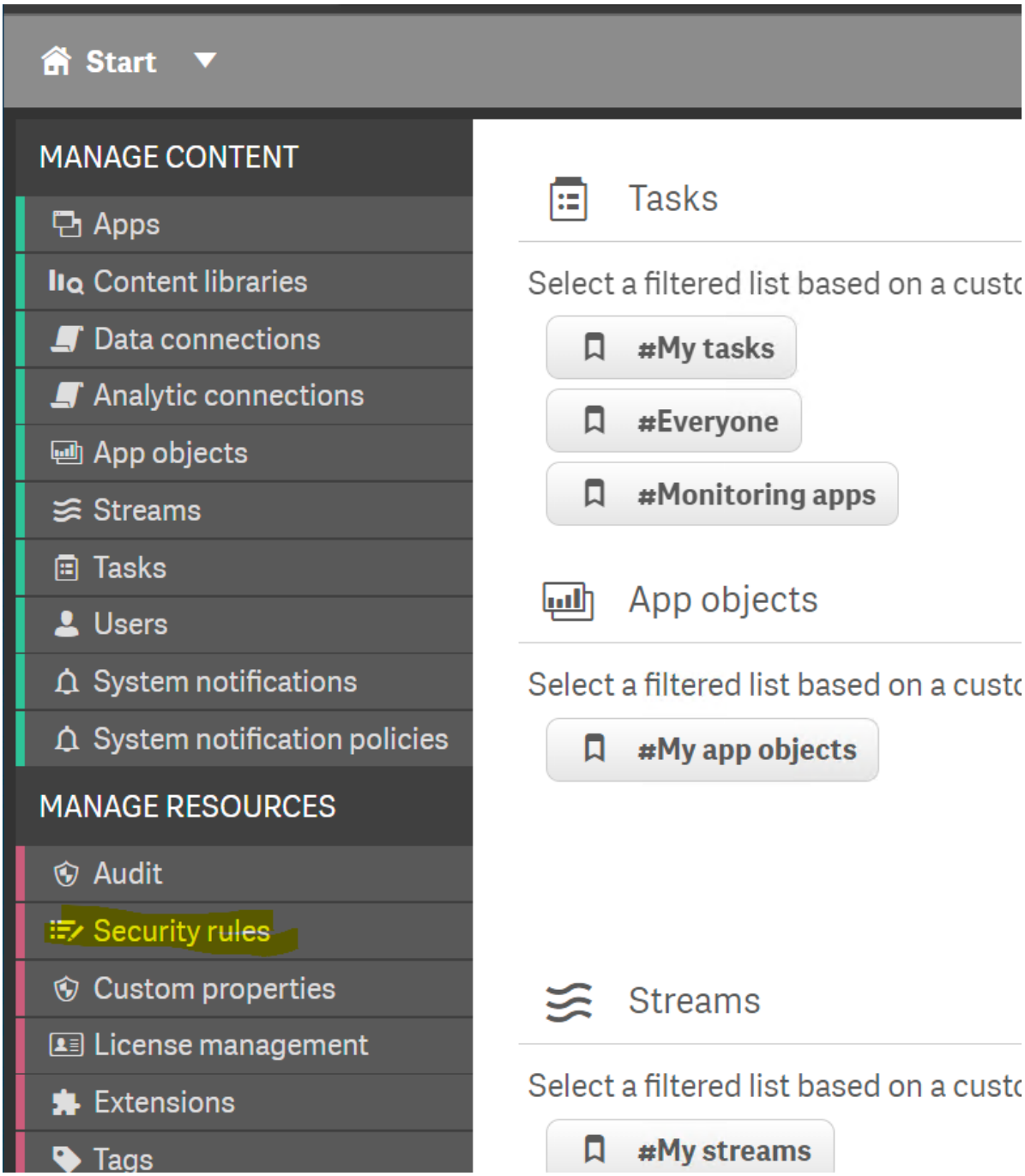
b) Select the intended user and click Edit.


Start
Users
Users Showing: 17 Selected: 1
Name
User directory
User ID
michaelm OCTOPAI-CORP michaelm
octopai DEV-WE-QLIK octopai
qlik admin DEV-WE-QLIK qlik admin
qlikuser DEV-WE-QLIK qlikuser
sa_api INTERNAL sa_api
sa_converter INTERNAL sa_converter
sa_engine INTERNAL sa_engine
sa_hub INTERNAL sa_hub
sa_printing INTERNAL sa_printing
sa_proxy INTERNAL sa_proxy
sa_qlikview INTERNAL sa_qlikview
sa_reporting INTERNAL sa_reporting
sa_repository INTERNAL sa_repository
sa_scheduler INTERNAL sa_scheduler
zacay DEV-WE-QLIK zacay
zacayd OCTOPAI-CORP zacayd
stask OCTOPAI-CORP stask
Edit
Delete

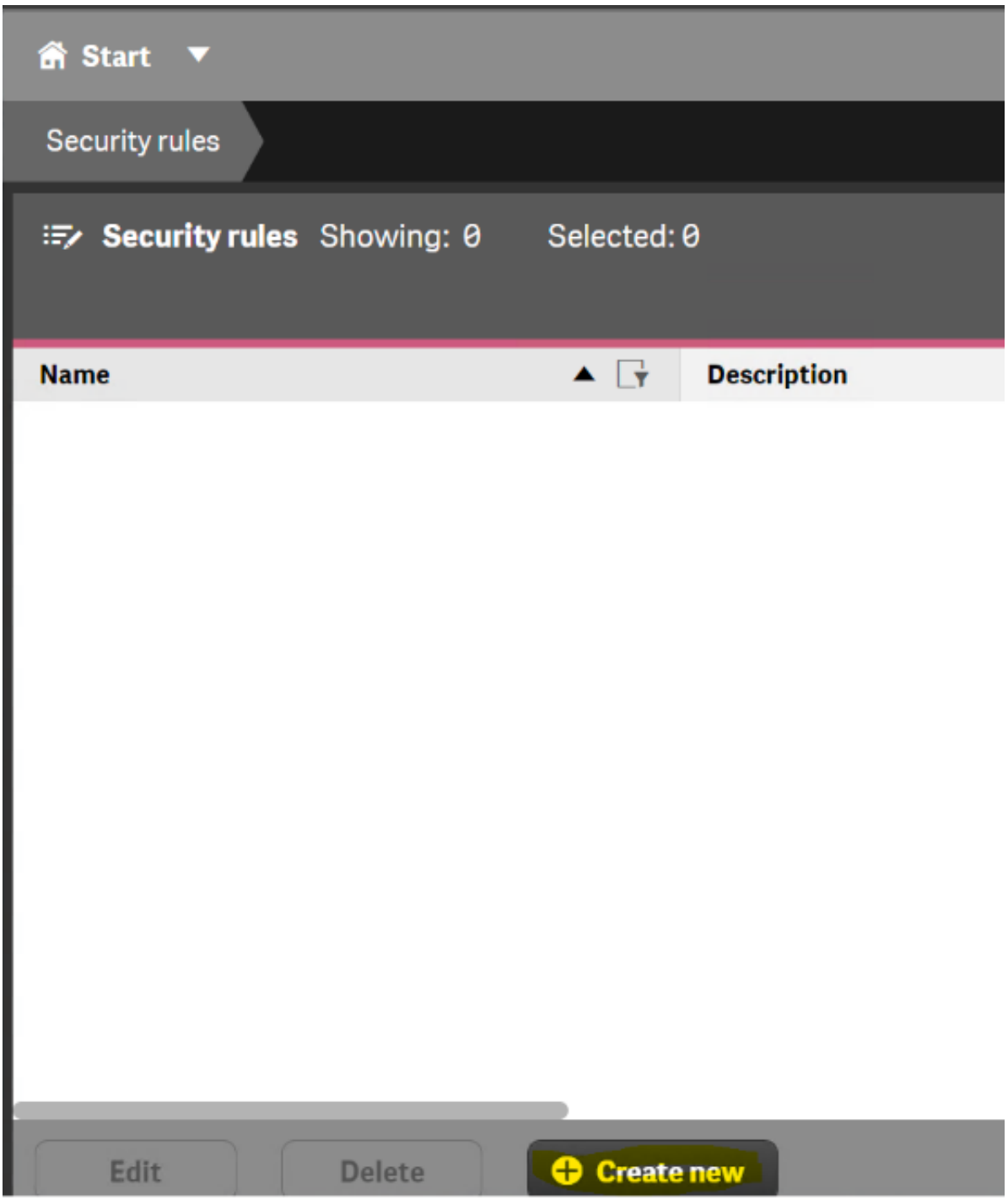
c) Under Custom properties, find Octopai_Group and select Octopai from the dropdown menu.

5. Create a security rule to connect the user and the streams.

a) Go to the Security rules menu.



b) Click Create new and use the following values:



Name

Octopai_Stream_Rule

Resource filter

Stream_*

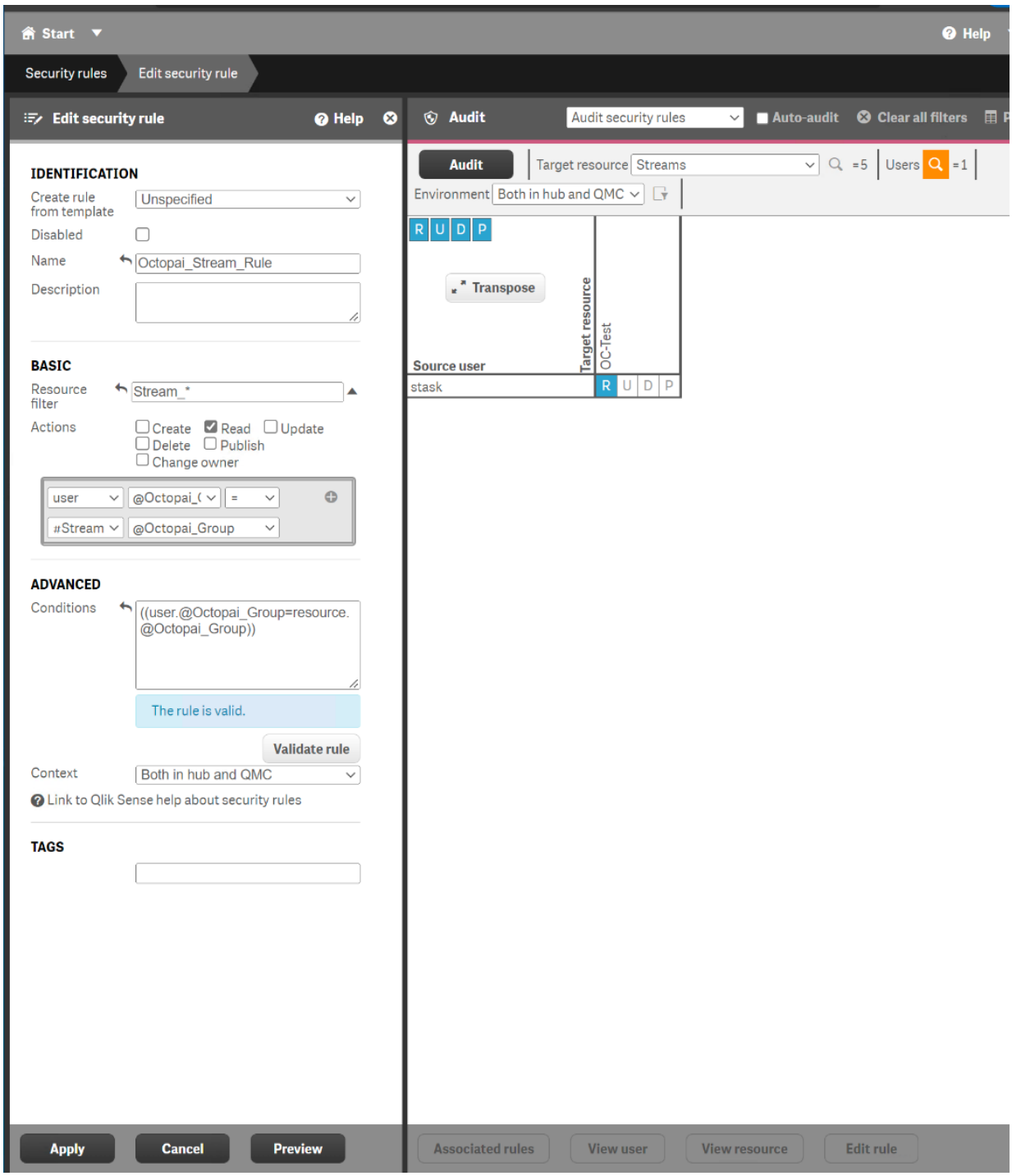
Actions

Read

Conditions

((user.@Octopai_Group=resource.@Octopai_Group))

c) (Optional) Click Validate rule to verify the configuration.



QlikView

How to configure the Cloudera Data Lineage (formerly Octopai) QlikView connector.

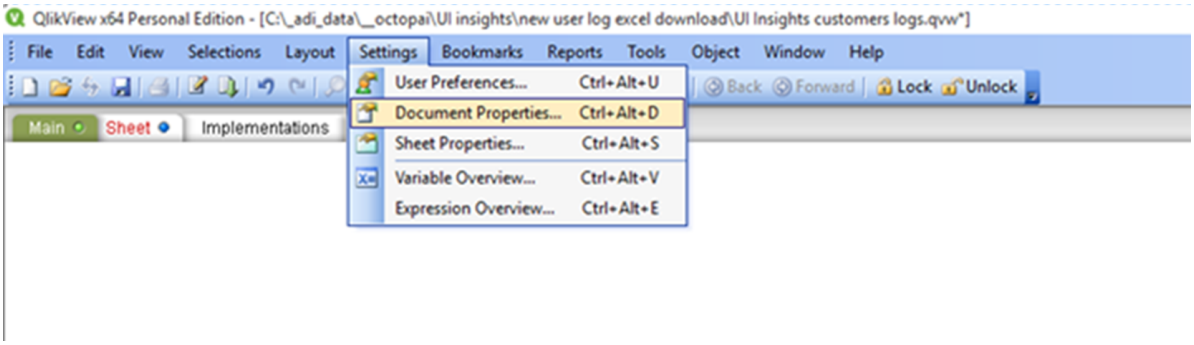
Before you begin

- **Note:** Supported version: 2022
- Read Permission for Cloudera Data Lineage Windows NT User to the QlikView Log files folder.

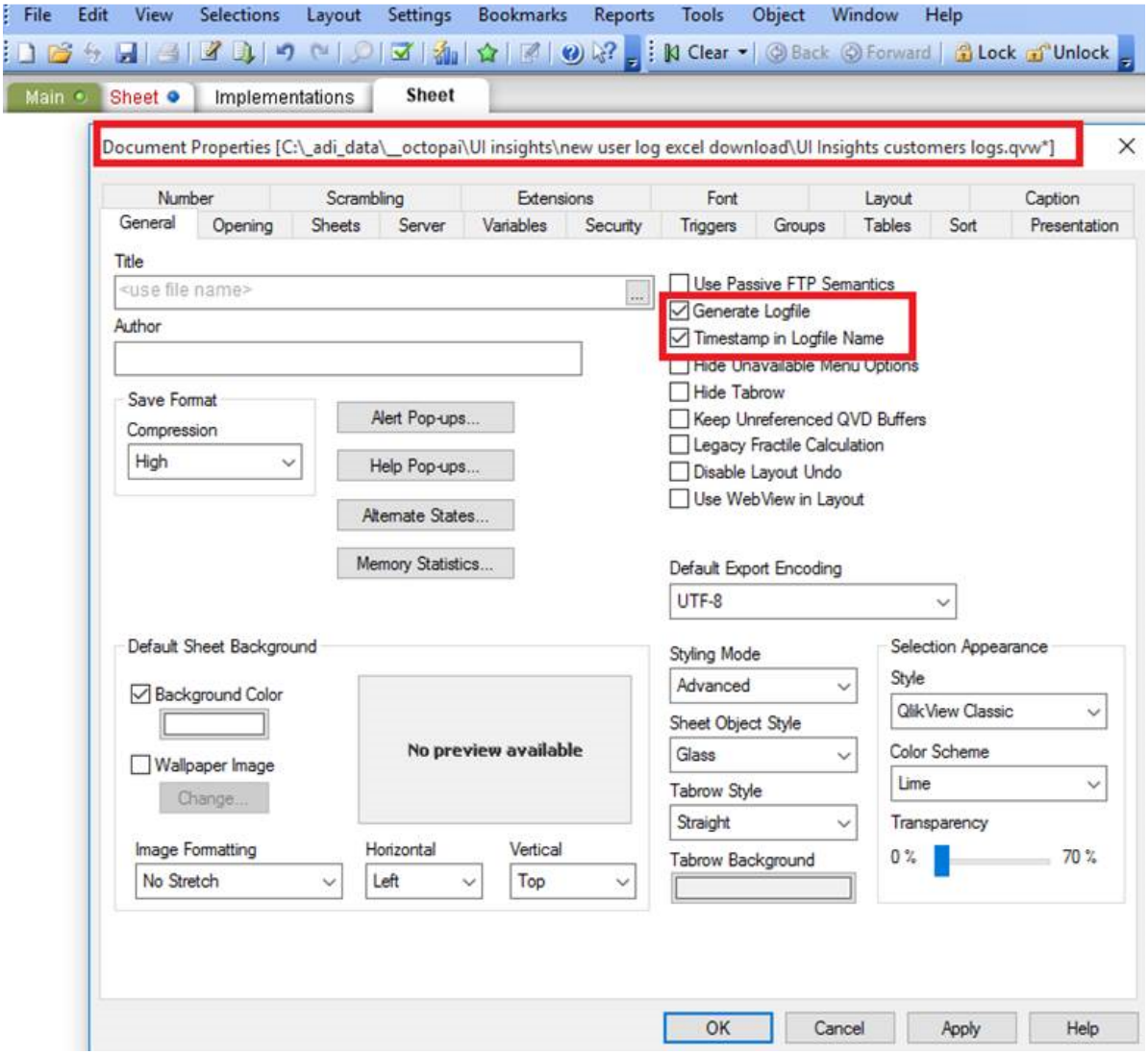
Warning: Missing permissions could end up in broken lineages.

Procedure

1. Generate QlikView log files
   a) Open the QlikView Model that needs to be uploaded to Cloudera Data Lineage.
   b) Choose “Settings” > “Document Properties”.



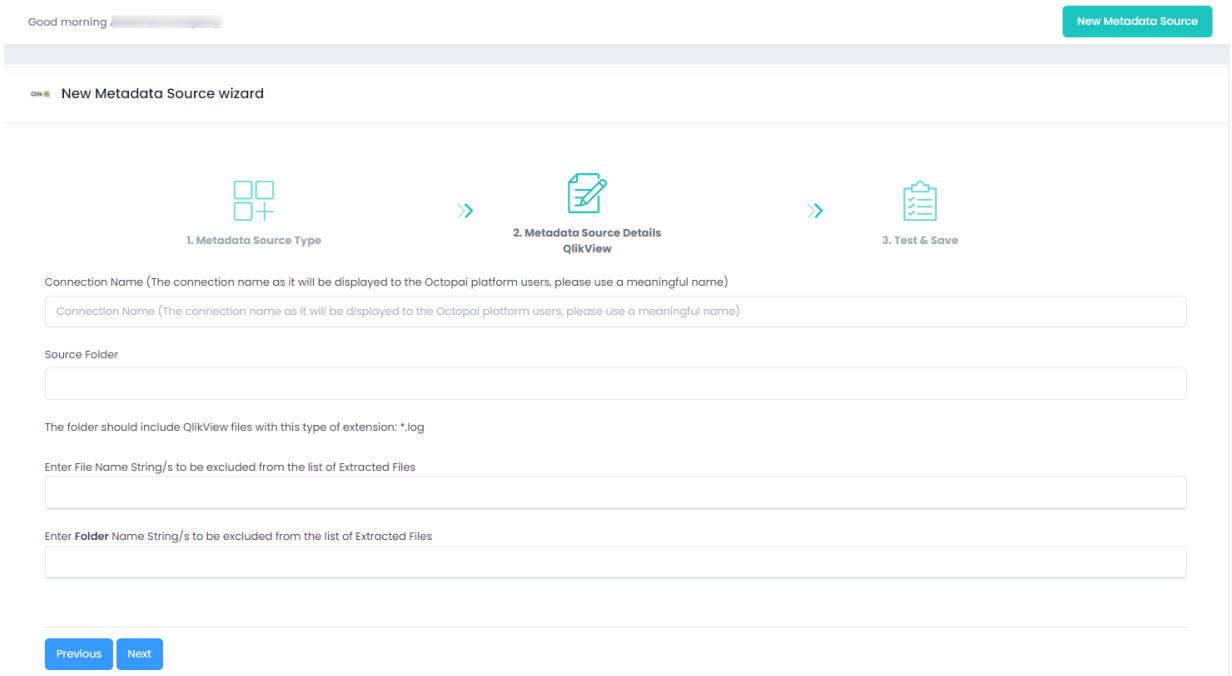
   c) Check the boxes “Generate Logfile” and “Timestamp in Logfile Name”.
   d) Copy and save the Header – This is the location of the log files that will be used for the Cloudera Data Lineage Application.



e) Click on Apply and OK.

f) Use the generated Path (header) for the log file location in the Octopai Client.

2. Set up QlikView Metadata Source



If an error occurred during the setup or extraction, see Troubleshoot connector extraction failures.

SAP Business Objects (BO)

Extract LCMBIAR files from SAP Business Objects (BO) for Cloudera Data Lineage (formerly Octopai) integration using the Central Management Console or Promotion Management Wizard.

Note: Version supported: up to 4.3

Tool Permissions Prerequisites

Warning: Missing permissions could end up in broken lineages.

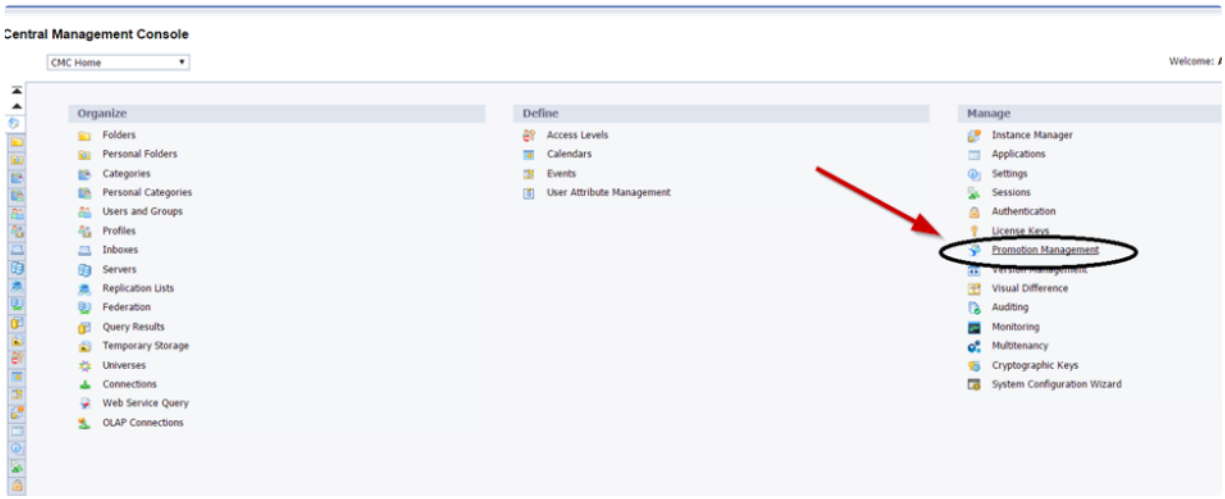
Enable read permission for Cloudera Data Lineage Windows NT User to the Business Objects BIAR files folder.

How to extract LCMBIAR files

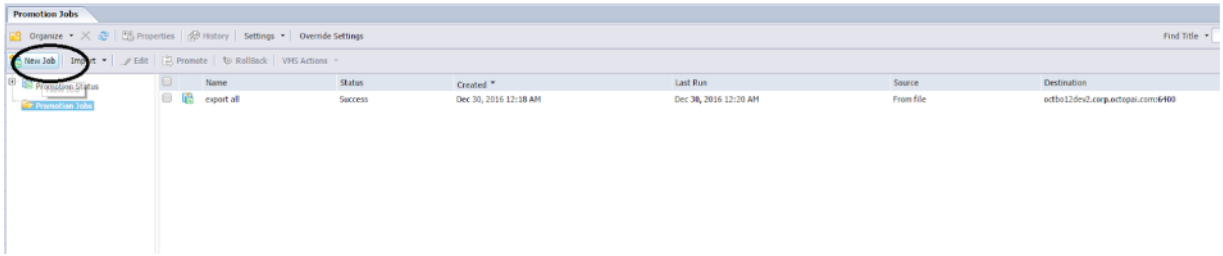
You can extract LCMBIAR files using one of the following methods:

Method 1: Export LCMBIAR with the Central Management Console

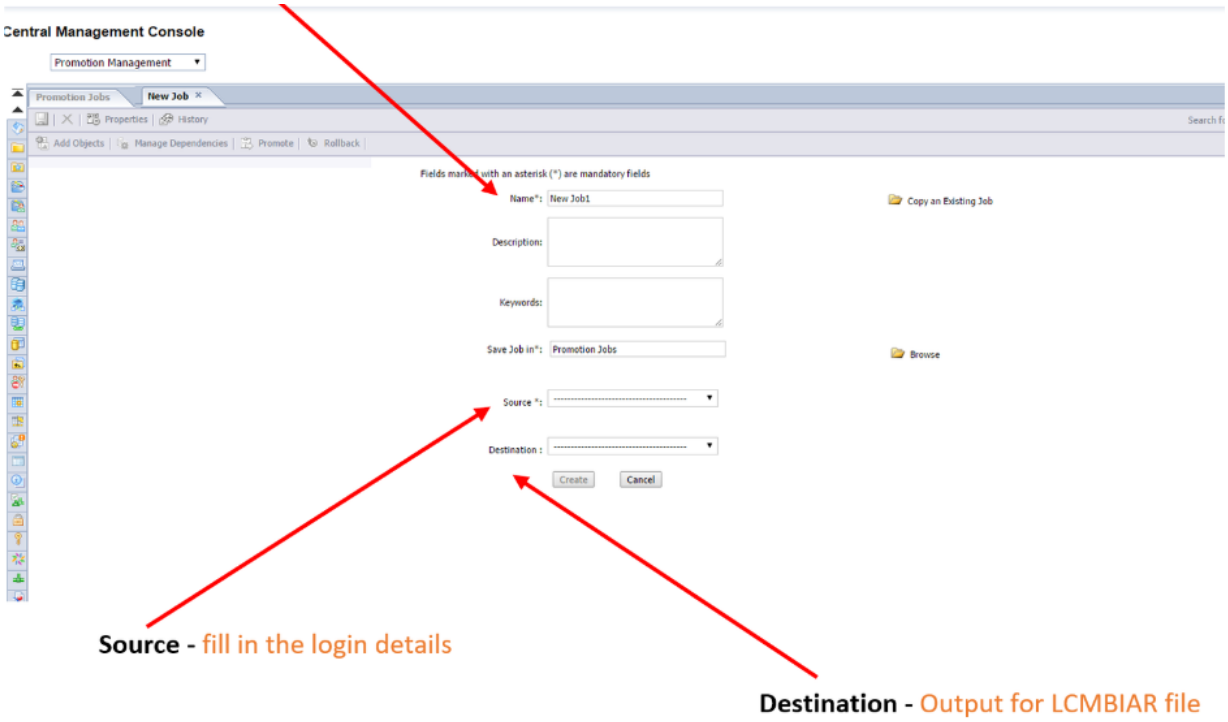
1. Log in to CMC: http://bo-server-name.com:8080/BOE/CMC
2. Click on promotion management:



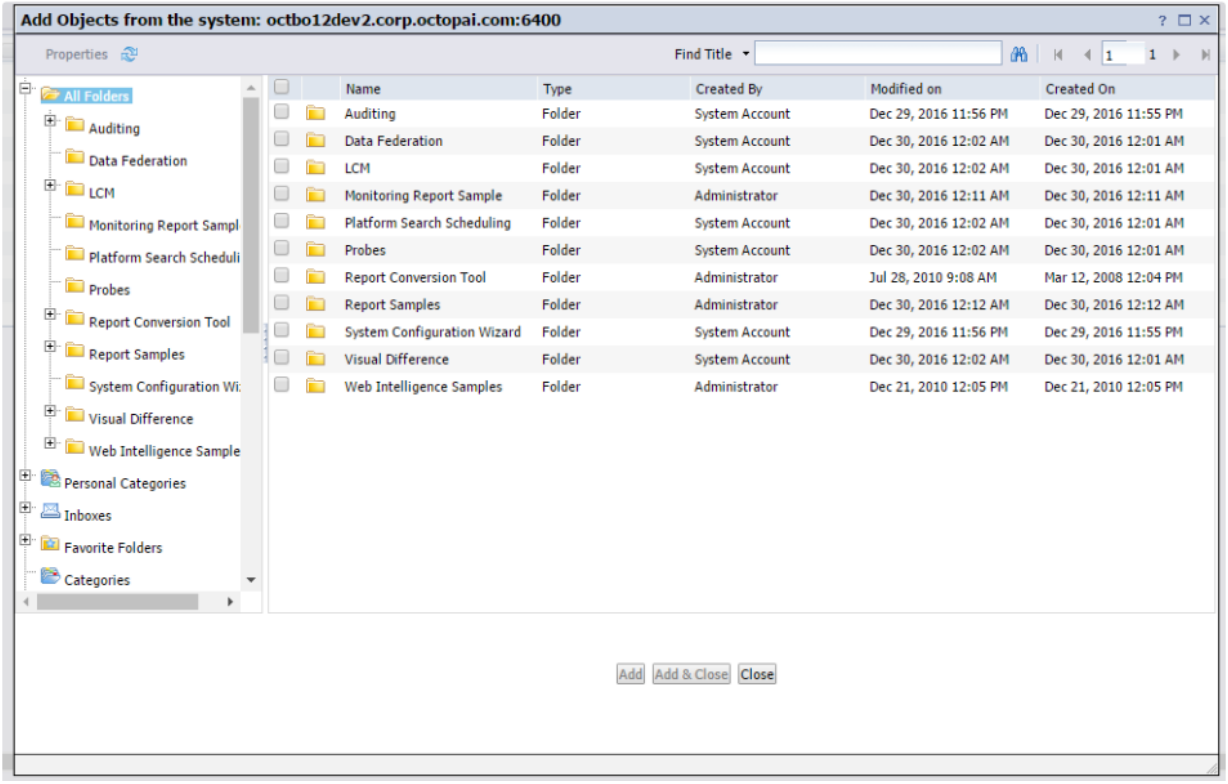
3. Create a new job:



4. Fill in a job name and details:



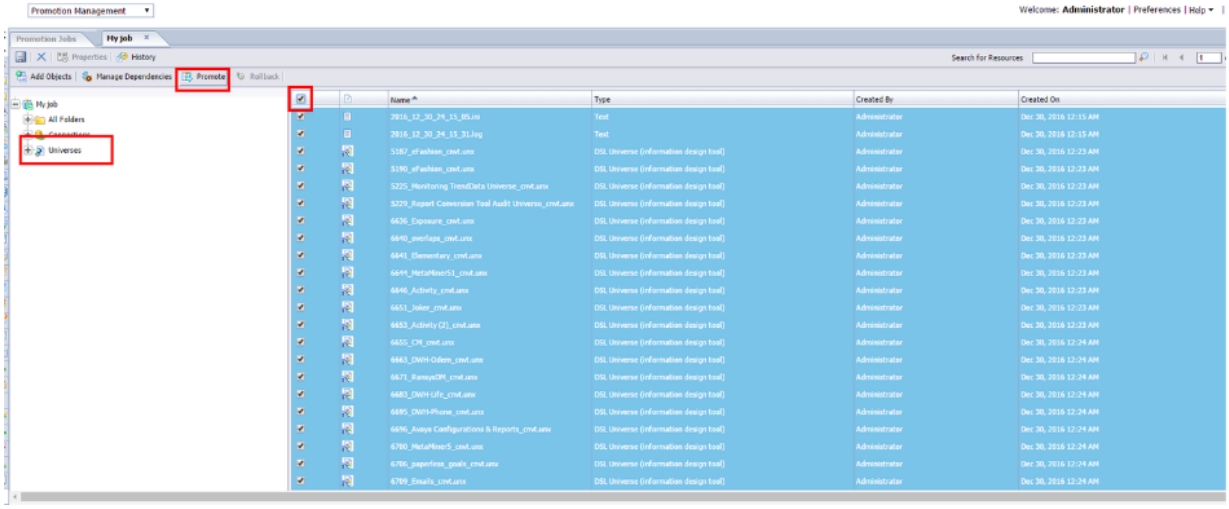
5. Click on "Create":



6. On the left side panel choose Universes and then select all by clicking the upper checkbox
7. Click add
8. Repeat steps from #6 above for **All Connections**
9. Repeat steps from #6 above for **All folders (reports)**

Note: There is a 500 MB file size limit for LCMBIAR files, In the event report files are larger than 500 MB please divide this step as necessary by breaking up the selection of folders and creating separate output files

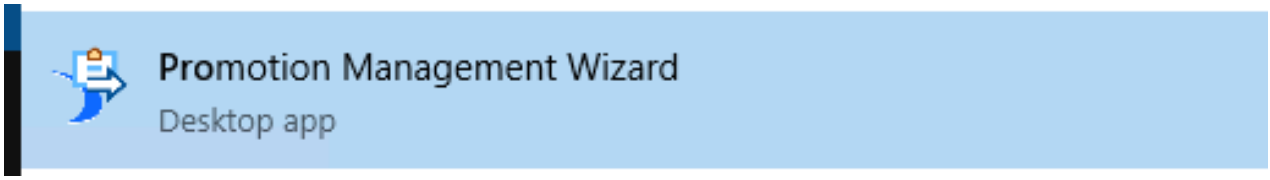
10. Click on close.
11. Select all and click on the promote button, then choose - **export now**



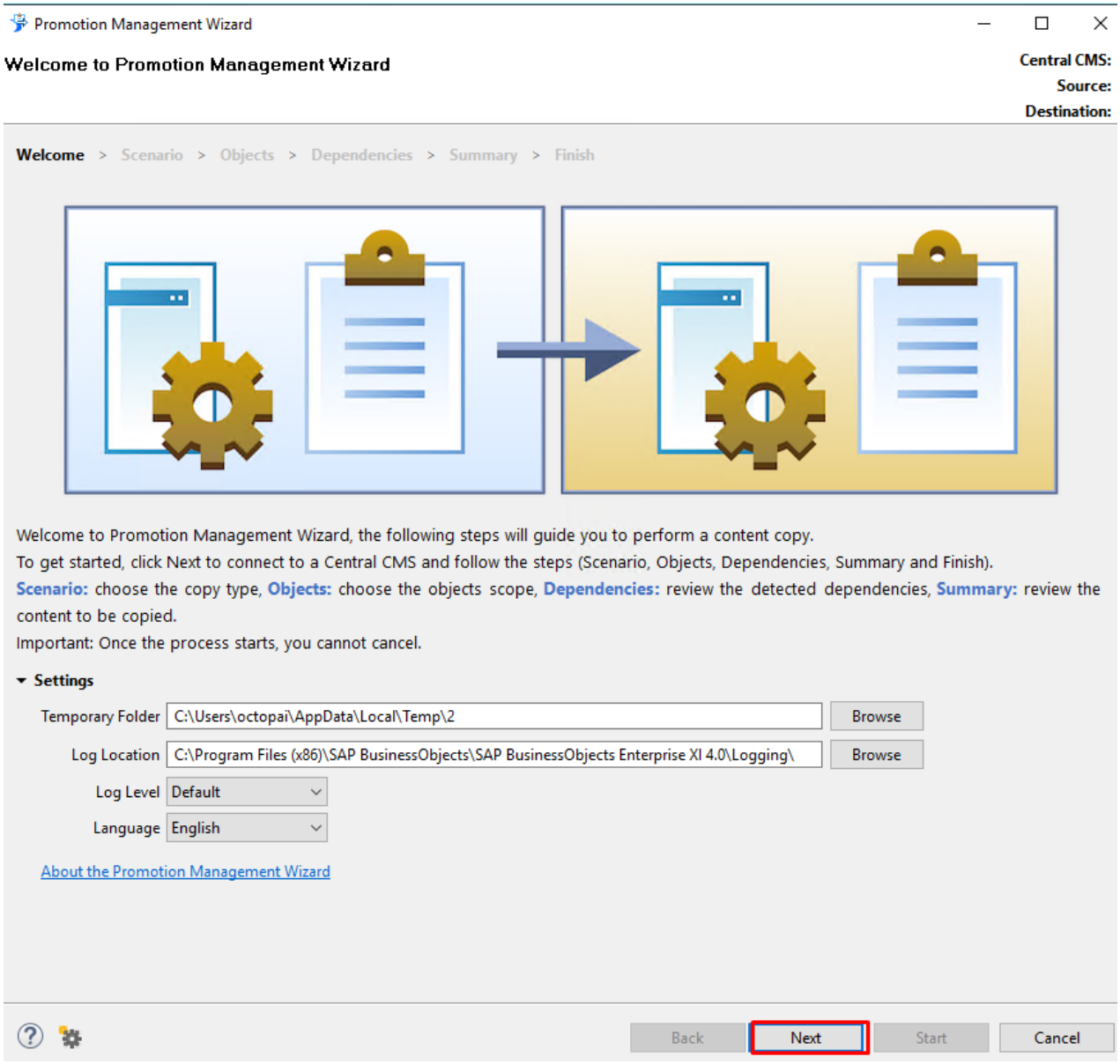
Note: The file (*.lcmbiar) will appear in the destination-defined folder.Use this file as a source for your Cloudera Data Lineage metadata BO connection in the Octopai Client.

Method 2: Export LCMBIAR with the Promotion Management Wizard

Go into the BO Server and open the program called " **Promotion Management Wizard** ".



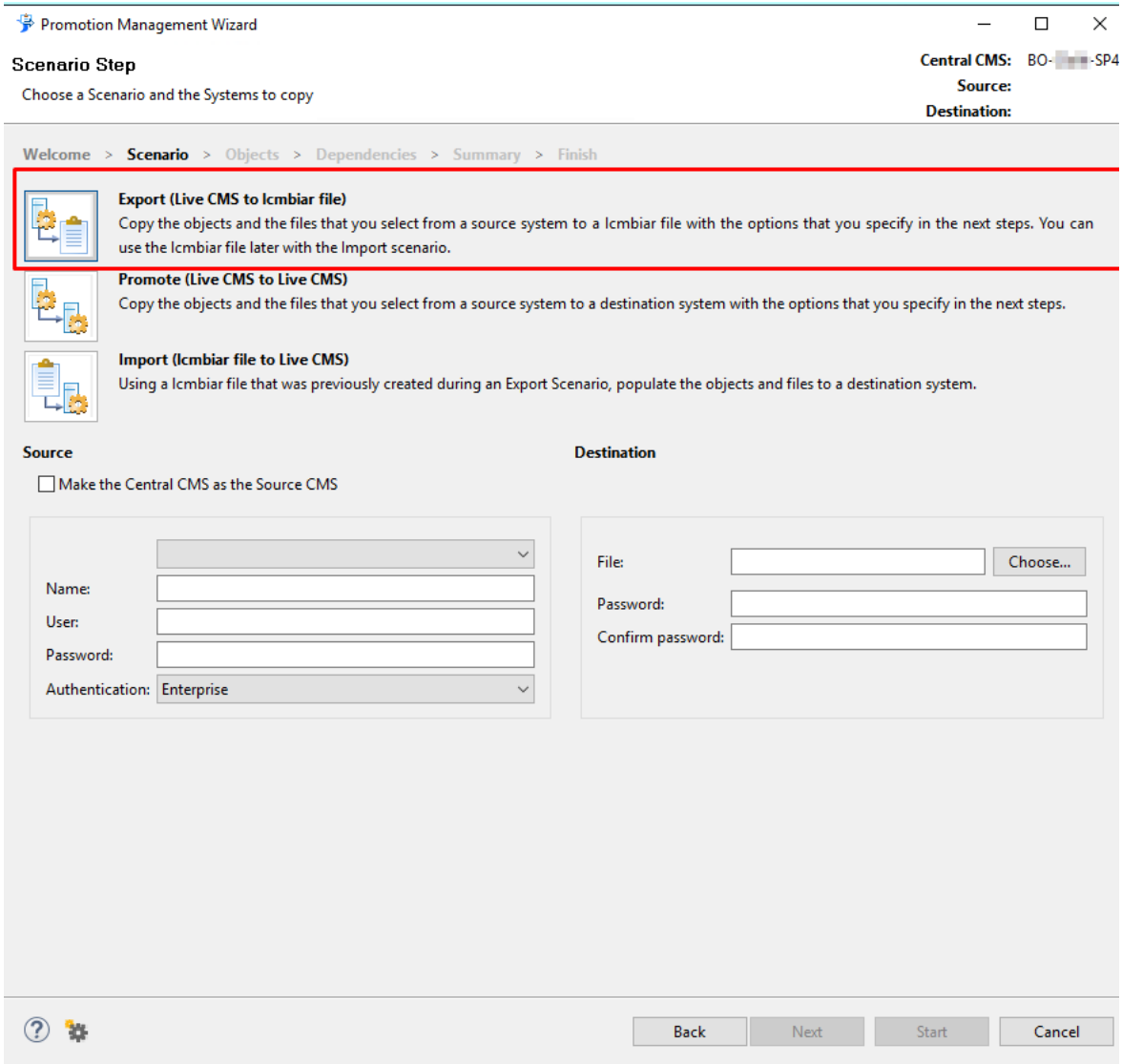
When the following window opens, click **Next**



Login to the CMS (BO Server) that you want to extract from.

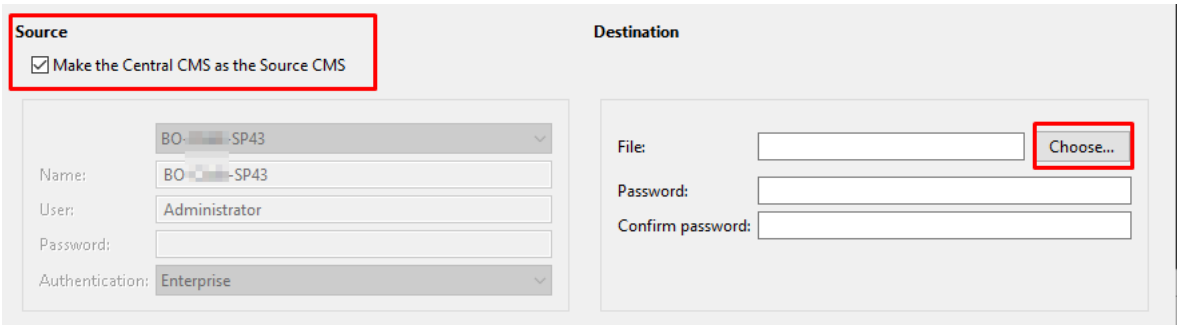


Click on " **Export (Live CMS to lcmbiar file)** "

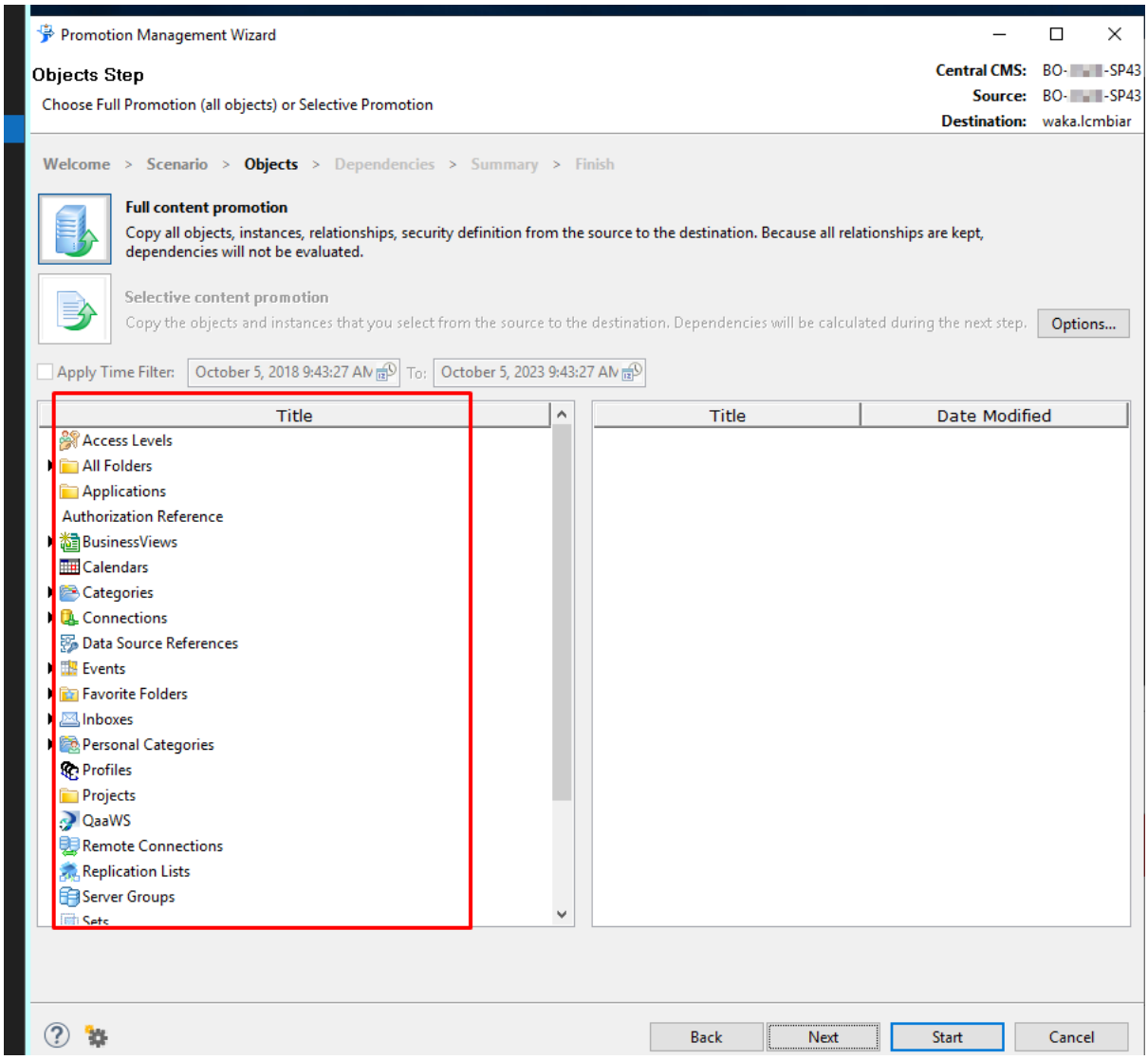


Fill up the information:

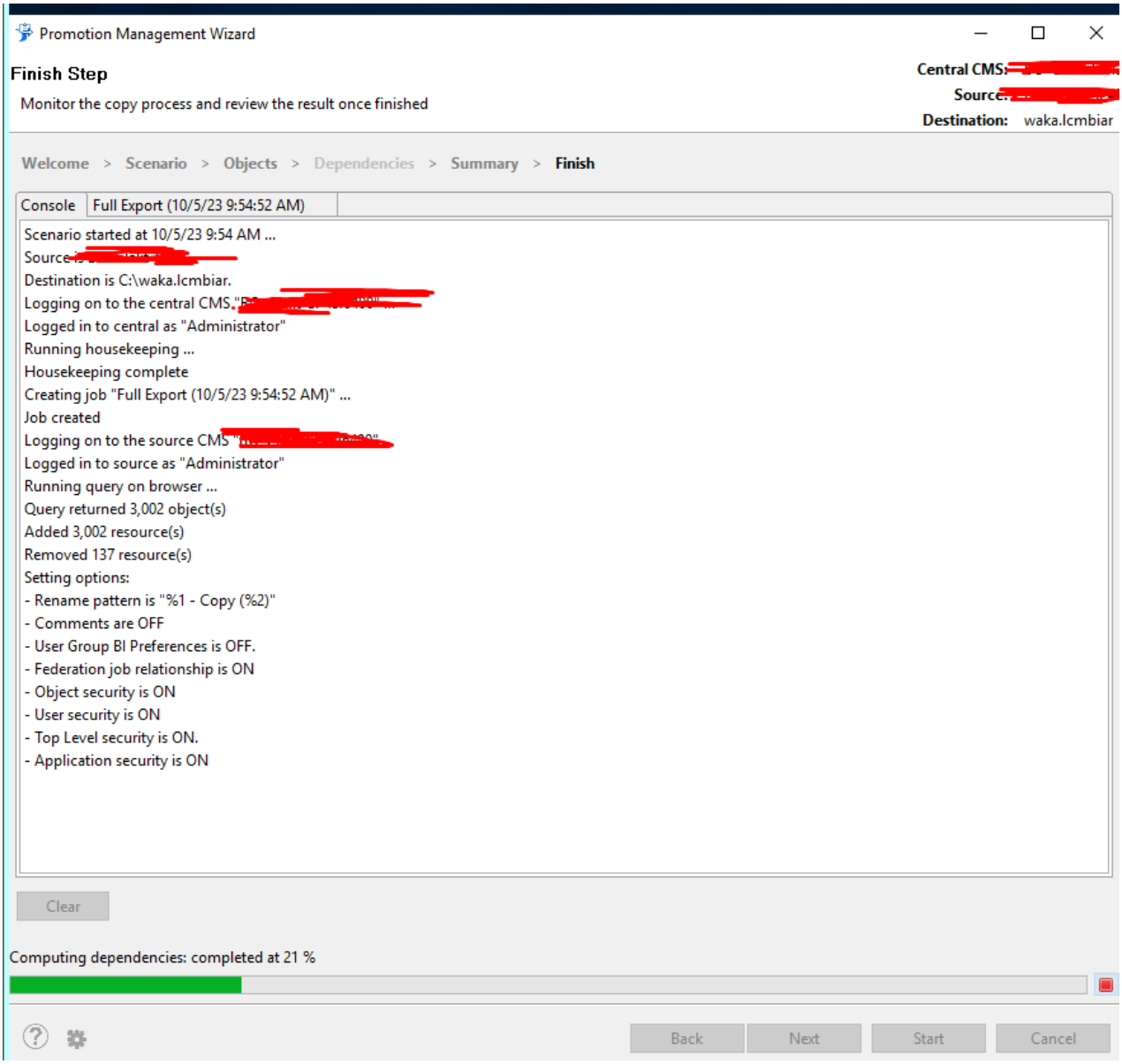
- Click on the Source tab to **“Make the Central CMS as the Source CMS”** which means to use the local CMS as our source.
- In the Destination tab choose where to save the file and give it a password. **(Password is Mandatory).**



Choose the objects you want to export whether it’s Universes, Reports, Connections.



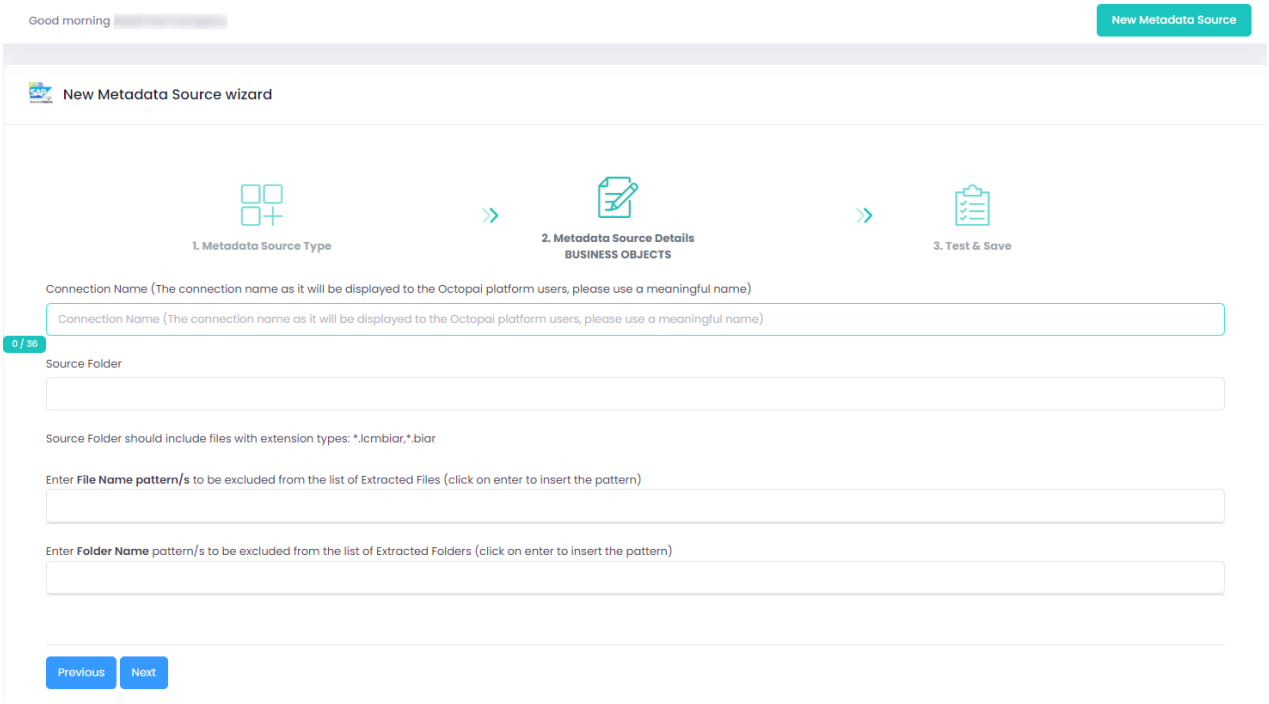
Then afterward select **Start** .



All done!Once the file has been created, upload the file to the Cloudera Data Lineage Portal.

Setting up SAP-BO Metadata Source

Metadata Sources are set on the Octopai Client



Verify extraction results

After a successful run, confirm the metadata archive in the target folder. See Verify extracted metadata files.

Troubleshoot

If an error occurs during the extraction, see Troubleshoot connector extraction failures.

SSAS (OLAP + Tabular + Azure Analysis Services)

Learn how to set up and manage SSAS (OLAP, Tabular, and Azure Analysis Services).

Note: Version supported:

- Tabular: up to 2022
- OLAP: up to 2022
- Tabular on Azure: up to 2022

Tool Permissions Prerequisites

Warning: Missing permissions could end up in broken lineages.

Missing permissions could end up in broken lineages.

SSAS - OLAP (On Prem or hosted on Azure): NT User with “Read definition” for each DB on the server. (Windows Authentication).

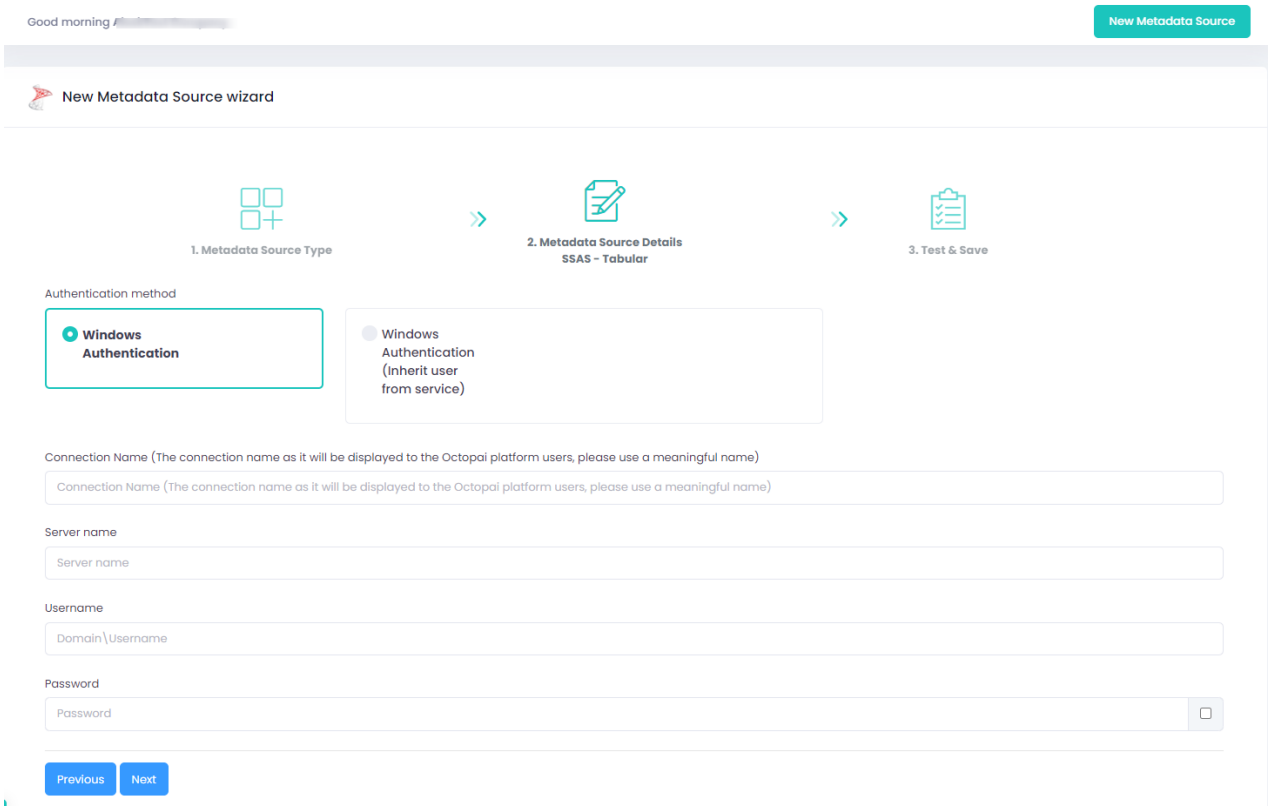
SSAS - Tabular: NT User with “Full control” for each model on the server. (Windows Authentication).

SSAS - Tabular on Azure Analysis Services (managed PaaS): 'Active Directory Password Authentication' User with “Full control” for each model on the server.

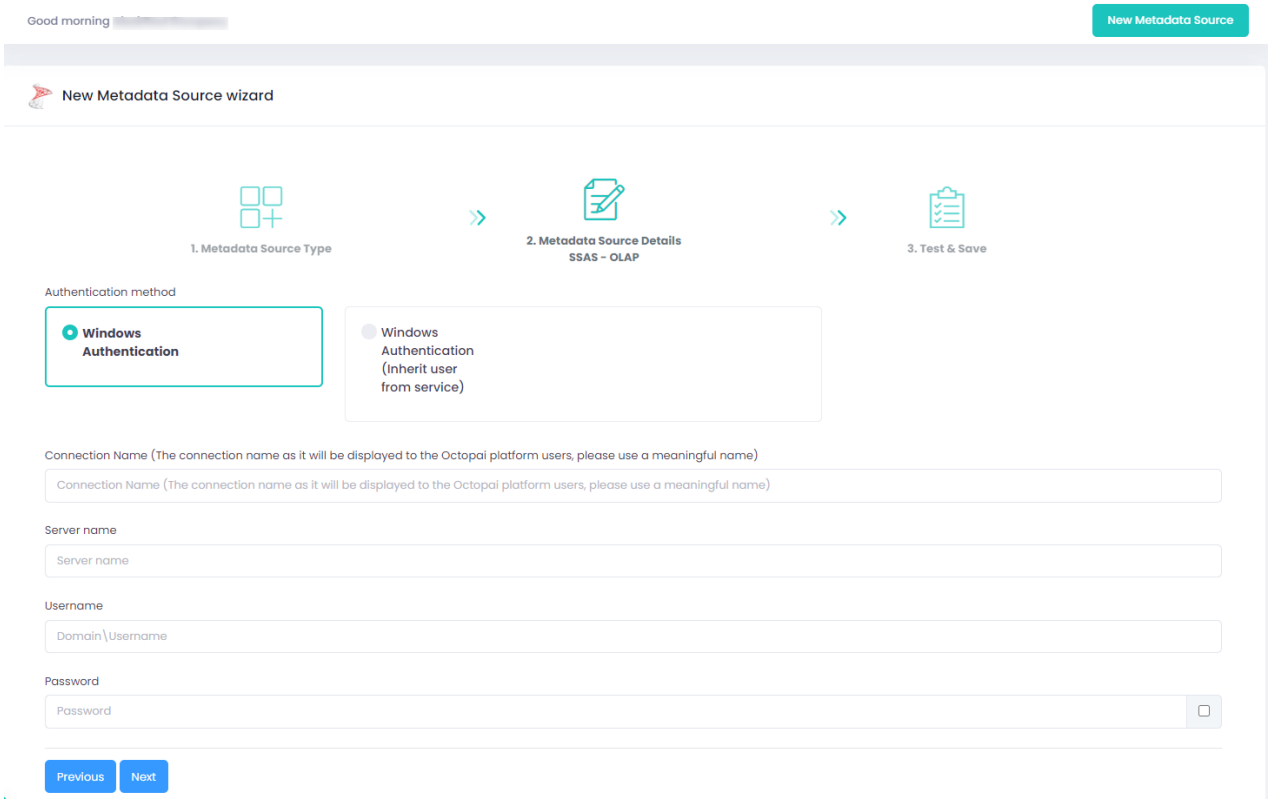
Setting up SSAS (OLAP + Tabular + Azure Analysis Services) Metadata Source

Metadata Sources are set on the Octopai Client

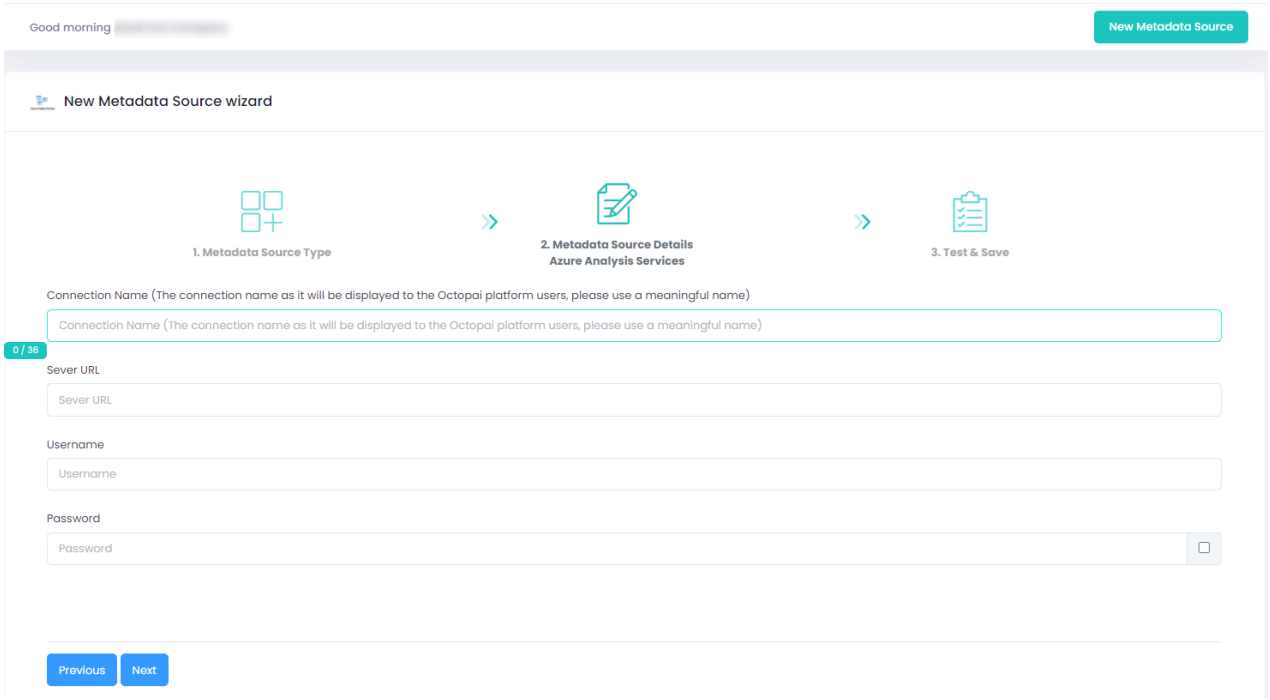
Tabular:



OLAP:



Azure Analysis Services:



Verify extraction results

After a successful run, confirm the metadata archive in the target folder. See Verify extracted metadata files.

Troubleshoot

If an error occurs during the extraction, see Troubleshoot connector extraction failures.

SSRS

Learn how to set up and manage SSRS metadata sources using Cloudera Data Lineage (formerly Octopai).

Note: Version supported: up to SQL 2022

Tool Permissions Prerequisites

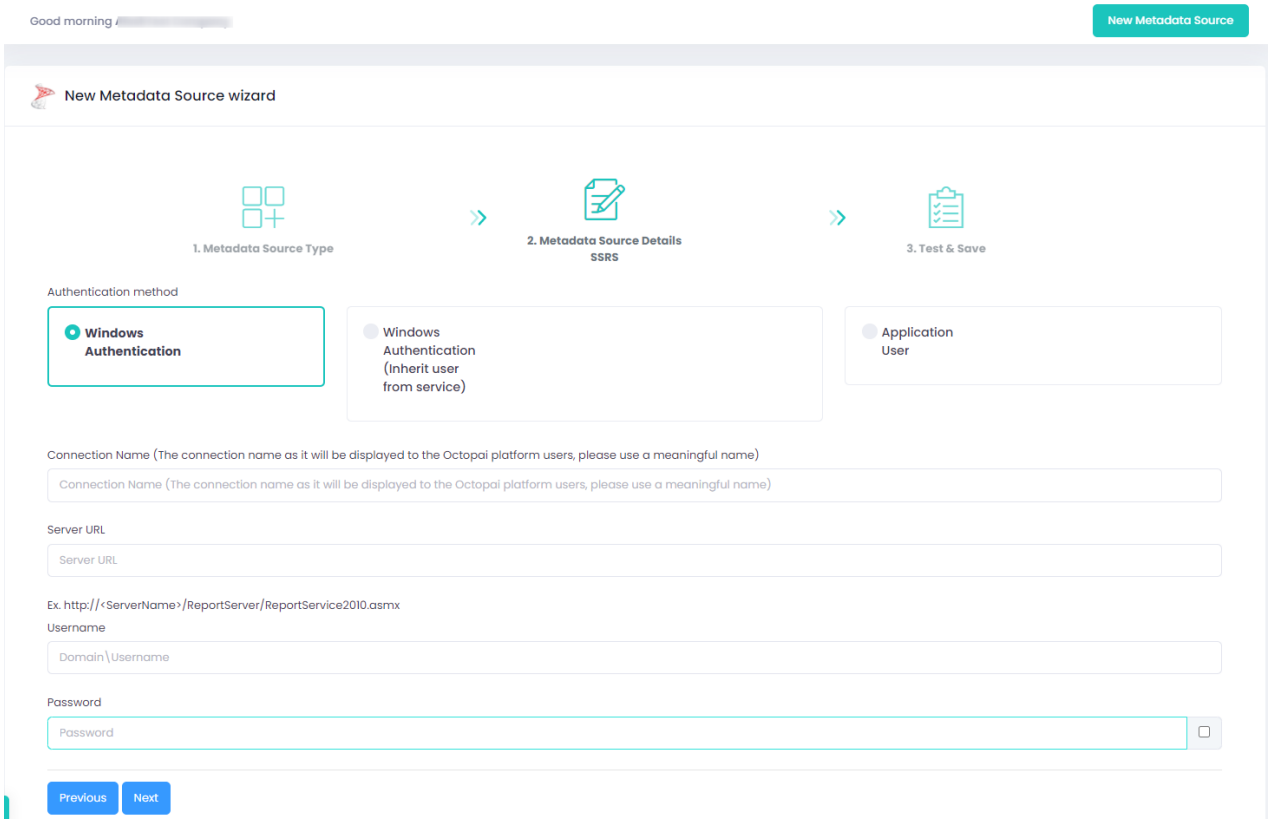
Warning: Missing permissions could end up in broken lineages.

User and password with permissions to the SSRS API (Azure Active Directory authentication not supported)

- The user needs to have “Content Manager” permissions on the report’s folder or the report itself.
  - http://<ServerName>/Reportserver/ReportService2010.asmx * Native Mode Min Permission: Content Manager
  - http://<sp server name>/_vti_bin/ReportServer/ReportService2010.asmx * Sharepoint Integrated Mode Min Permission: Owners

Setting up SSRS Metadata Source

Metadata Sources are set on the Octopai Client



Verify extraction results

After a successful run, confirm the metadata archive in the target folder. See Verify extracted metadata files.

Troubleshoot

If an error occurs during the extraction, see Troubleshoot connector extraction failures.

Tableau Server

Learn how to configure Tableau Server for metadata extraction.



Note: Version supported: up to 2023

Tool Permissions Prerequisites



Warning: Missing permissions could end up in broken lineages.

Tableau Server/Online – Supported (Tableau Desktop - Not Supported by automation)

- Tableau server from version 10.3 and up
- API of Tableau Server is Enabled (by Default)
- Login permissions to the Tableau server using one of the following options:
  - Tableau Personal Access Token (PAT) associated with a user that has Server/Site Admin permissions to the Tableau API & the relevant Sites (Recommended). More about PAT can be found here: Personal Access Tokens
  - Tableau User and password with Server/Site Admin permissions to the Tableau API & the Relevant Sites.

The error message displayed when API is disabled:

```
System.Net.Http.HttpRequestException: Method: POST, RequestUri: 'http://<Ser
ver>/api/<api_version>/auth/signin', Version: 1.1, Content…
```

Check and run Tableau API:

Tableau server version 10.3 or lower:

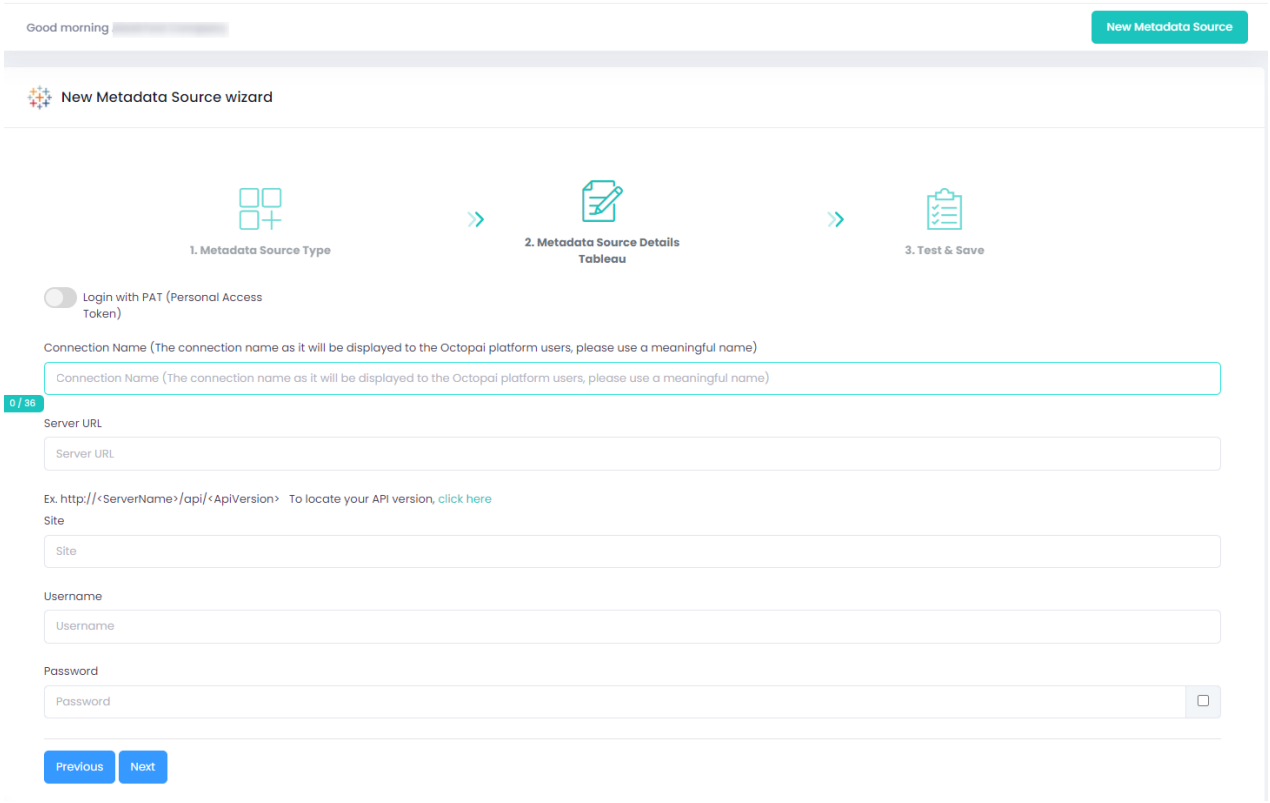
- Check if the default Rest API of Tableau is enabled --> tabadmin get api.server.enabled
- If it isn’t enabled (false): tabadmin set api.server.enabled
- For this, one of the following pre-requisites is required:
  - tabadmin stop
  - tabadmin start
  - tabadmin config

Tableau server versions later than 10.3:

- Check if the default Rest API of Tableau is enabled: tsm configuration get --key api.server.enabled
- If it isn’t enabled (false): tsm configuration set --key api.server.enabled true
- For this, one of the following pre-requisites is required:
  - tsm stop
  - tsm start

Setting up Tableau Metadata Source

Metadata Sources are set on the Octopai Client



Verify extraction results

After a successful run, confirm the metadata archive in the target folder. See Verify extracted metadata files.

Troubleshoot

If an error occurs during the extraction, see Troubleshoot connector extraction failures.

Tableau Intelligent Graph Connector for operational metadata intelligence

Learn about setting up Tableau Intelligent Graph Connector to extract operational metadata for usage, performance and auditing data.

Permissions prerequisites

The following permissions and prerequisites are valid for setting up Tableau Intelligent Graph Connector:

- **API License** – Ensure that the API of Tableau Usage is enabled through your Tableau license. Refer to your Tableau administrator or support if you need help enabling the API.
- **Login Permissions** – Ensure you have login permissions to the Tableau Usage PostgreSQL server.
- **Database Access** – You will need PostgreSQL server and port information, along with a username and password. Ensure that the SELECT permissions are granted for the following public data tables:
  - _sites
  - users
  - views
  - hist_comments

- hist_views
- _http_requests
- historical_events
- projects
- hist_users
- workbooks
- system_users
- _sessions

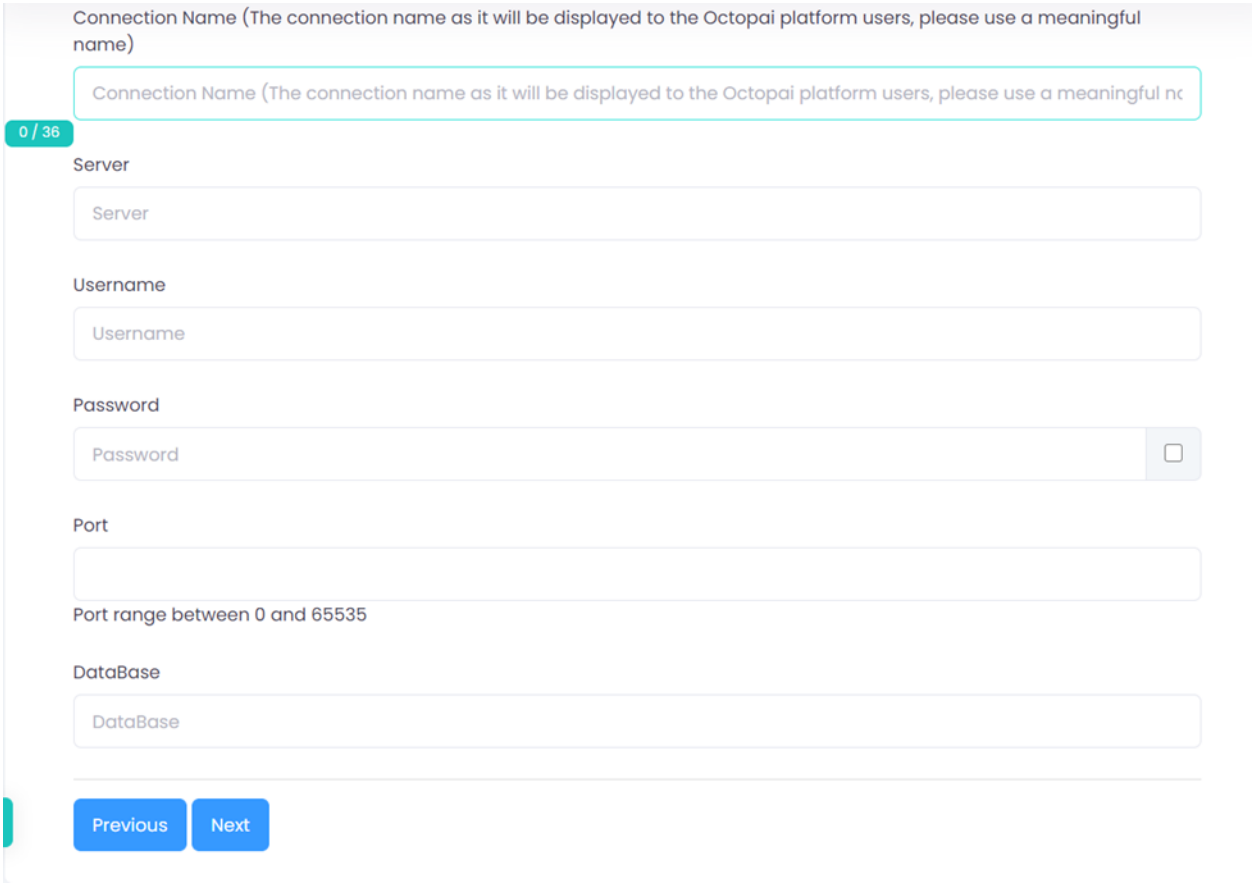
Setting up Tableau Metadata source

To configure Tableau Metadata in Cloudera Data Lineage (formerly Octopai):

1. **Open Octopai Client.**

   Launch the **Octopai Client** on your system.
2. **Navigate to the metadata source setup.**
   - In the Octopai Client interface, go to the **Metadata Sources** section.
   - Locate and select **Tableau** as the metadata source.
3. **Input the server credentials.**
   - Enter the relevant PostgreSQL server details, including **Server** , **Port** , **User** , and **Password** . This information must correspond to the Tableau PostgreSQL database.
   - Ensure you provide the correct login credentials and that the required permissions are in place (as listed in the prerequisites).
4. **Select data tables.**
   - Select the relevant tables for which you have permissions, such as those mentioned in the prerequisites.
5. **Save the configuration.**
   - Once you have input the necessary details, save your configuration. The Octopai Client will now have access to Tableau's Operational Metadata.

Figure 35: Tableau metadata in Cloudera Data Lineage configuration

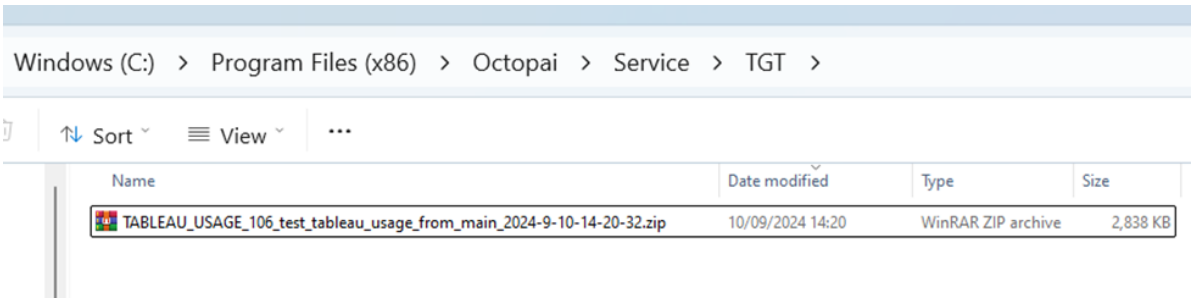


Verifying the extracted metadata file

Once the metadata extraction is complete, follow these steps to verify the extracted files:

1. **Access the target folder.**
   - On the server where the **Octopai Client** is installed, navigate to the **Target (TGT) Folder** .
   - **Default Path** : C:\Program Files (x86)\Octopai\Service\TGT
2. **Locate the connector file.**
   - Inside the TGT folder, you will find a **.zip** file named after the **Tableau Connector** .
   - Example: Tableau_Metadata.zip
3. **Open the zip file.**
   - Extract the contents of the **.zip** file.

   Figure 36: Open the ZIP file



4. **Verify file contents.**

- Ensure the quantity of inner files matches what you expect based on your metadata extraction process.
- Review the **quality** of the files by checking the format and ensuring that the data is consistent with what was pulled from the Tableau server.

Expected files and quality check

- The extracted .zip must contain several files representing different metadata components, for example views, projects, and historical events.
- Ensure that all necessary metadata files are present. Any missing or corrupted files could indicate an issue with the extraction process.

Data platforms and lakehouse

Cloudera Data Lineage connects to Databricks through Unity Catalog, Hive Metastore, or hybrid deployments, and to Microsoft Fabric Lakehouse as a technical preview.

Configure Databricks in Cloudera Data Lineage

Learn how to integrate Databricks with Cloudera Data Lineage (formerly Octopai) based on your catalog type, including Unity Catalog, Hive Metastore, or hybrid (Unity Catalog and Hive Metastore) deployments.

About this task

Before configuring Databricks in Cloudera Data Lineage, review the prerequisites that apply to your catalog environment. The configuration requirements vary depending on whether you use Unity Catalog, Hive Metastore, or a hybrid deployment combining both.

Requirements for Unity Catalog

Unity Catalog environments require system table access and Databricks SQL connectivity. To extract lineage, Cloudera Data Lineage must authenticate with a service principal and query Unity Catalog lineage system tables.



Note: Databricks admin permissions are required to view and manage Unity Catalog system tables and configure the access controls needed for lineage extraction.

Requirements for Hive Metastore

For environments using only Hive Metastore, ensure that the user or machine identity meets the following requirements:

- Permission to view and access the workspace folders containing the notebooks.
- Read access to the projects or directories selected for metadata extraction.
- Can view the relevant Hive Metastore objects referenced by the notebooks.

When working with Hive Metastore, ensure that a cluster is active and running during the extraction process. If the cluster is stopped or unavailable, Spark cannot execute the metadata queries, and metadata retrieval fails.

Requirements for Unity Catalog and Hive Metastore Hybrid Deployments

In hybrid environments, the following requirements must be met:

- Unity Catalog prerequisites, including SQL Warehouse access and permissions.
- Hive Metastore assets must also be included.
- Cloudera Data Lineage combines metadata sources to provide extended lineage coverage.

Perform the following steps to configure Databricks in Cloudera Data Lineage:

Procedure

1. Create a service principal (required)

   Unity Catalog lineage extraction requires a machine identity with access to governed metadata.

   You must ensure the following:

   - Create a Databricks-managed service principal.
   - Enable Workspace access and Databricks SQL access.

2. Enable or create an SQL Warehouse (required)

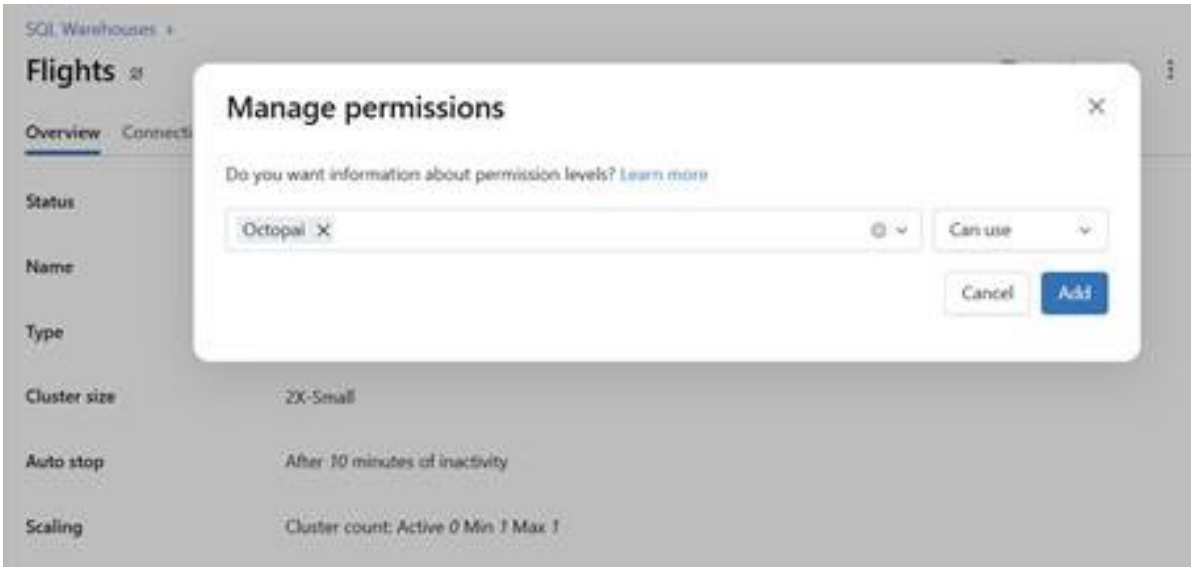
   Cloudera Data Lineage relies on querying Databricks system tables, which requires a running SQL Warehouse.

   You must ensure the following:

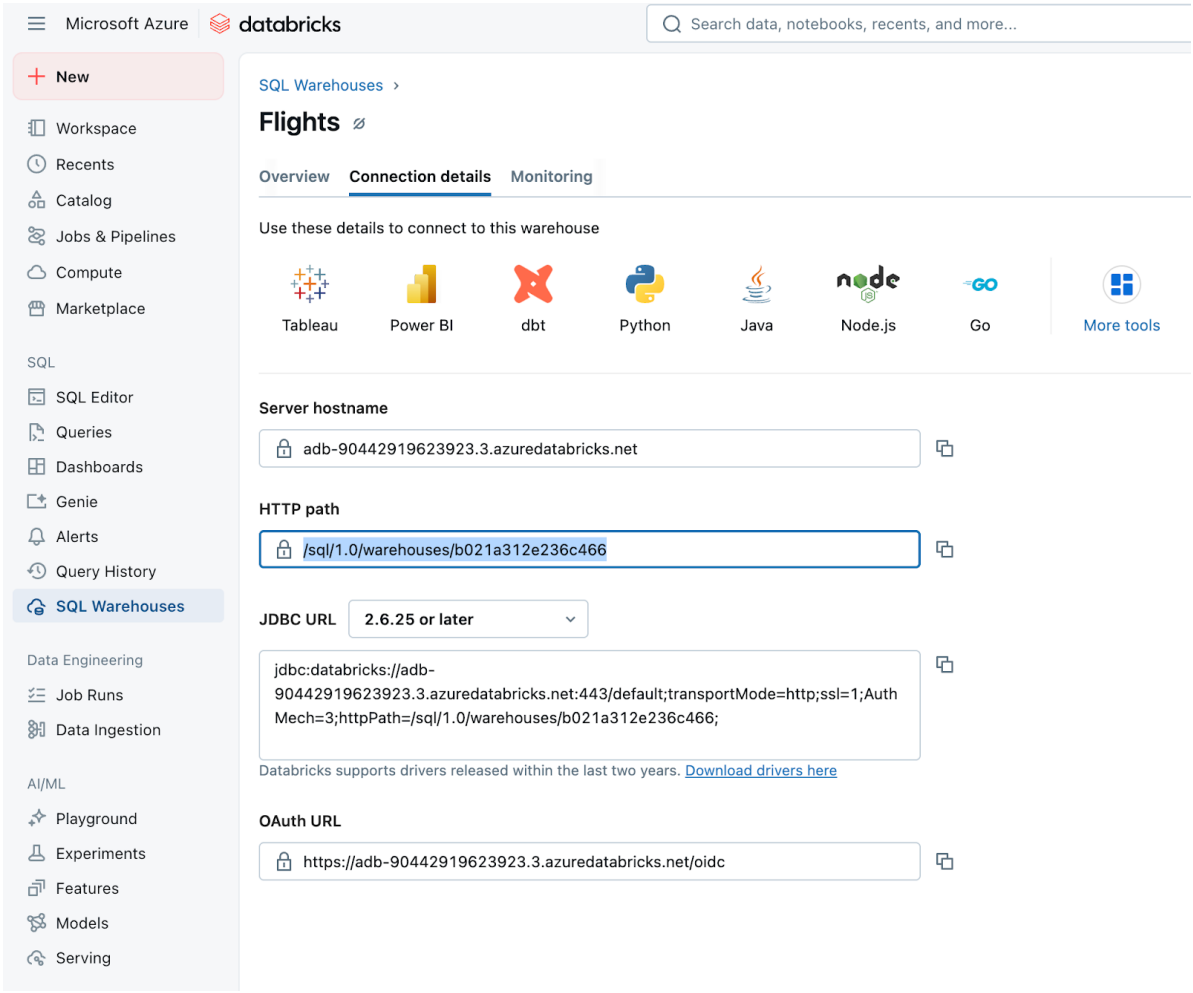
   - Create or enable a Databricks SQL Warehouse.
   - Allow access to required system schemas.

   Perform the following steps:

   a. In Databricks, go to the SQL Warehouses tab.
   b. If no SQL Warehouse exists,, click Create SQL Warehouse and configure it as required.
   c. Assign the service principal Manager permissions to the warehouse by selecting Can use.



   d. Open the SQL Warehouse and go to Connection details.
   e. Copy the HTTP path. You will need this path for the integration process.



3. Ensure Unity Catalog–Enabled Compute
   - Unity Catalog must be enabled at the workspace/account level.
   - A cluster that supports Unity Catalog access must be available.
4. Grant Unity Catalog Lineage Permissions (required)

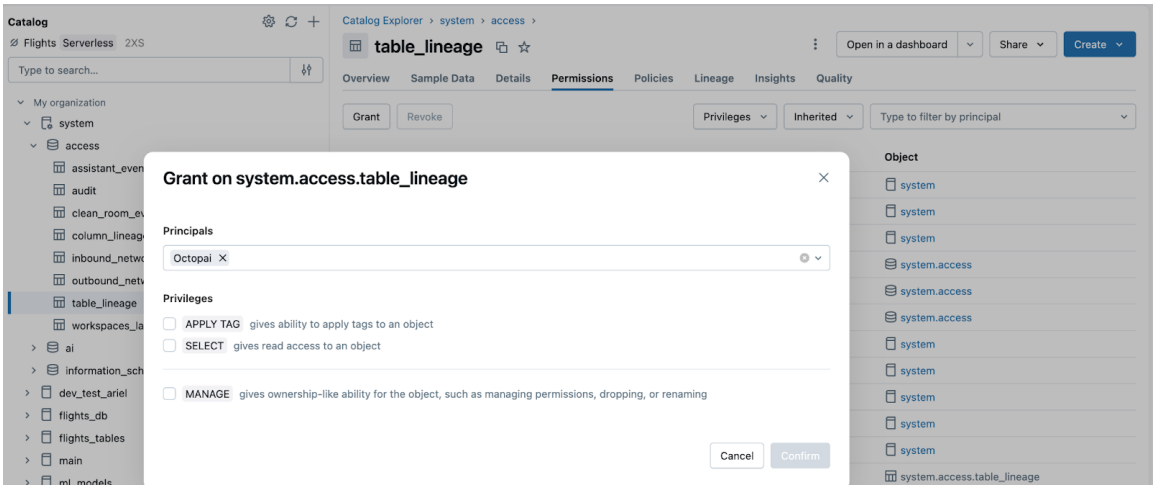
   The service principal must have SELECT access on the system lineage tables (system.access.table_lineage and system.access.column_lineage) and read access on relevant catalogs and schemas.

   a. Open the Catalog in Databricks.
   b. Search for:
      - Catalog: system
      - Schema: access
      - Tables: table_lineage and column_lineage

   c. The tables are automatically created by Databricks.

   Note: You must have admin permissions to view and manage the tables.

   a. For each table, perform the following steps:
      - Open the Permissions tab.
      - Click Grant.
      - Select the service principal created earlier.
      - Enable Select Permission.



5. Download the ODBC driver
   - Download and install the Simba ODBC driver for Databricks from the Simba ODBC driver archive page: https://www.databricks.com/spark/odbc-drivers-archive#simba_odbc
   - Select the appropriate version for your operating system (Windows or Linux).
6. Collect the required workspace information:
   - Workspace URL
   - Workspace ID (required)
   - Account ID (optional)

   a) Find the Databricks Account ID

   The account ID is available in the Databricks Account Console.

   1. Open the account console: https://accounts.cloud.databricks.com/
   2. Log in using your organization's credentials (SSO may be required).
   3. In the top-right corner, select your username/email to open the dropdown menu.
   4. Databricks displays the Account ID as a UUID value, for example: 55eb1a01-48d5-4008-8dbd-03dd8447a595
   5. Copy this value.

   b) Find the Databricks Workspace ID

   The workspace ID is embedded directly in your Databricks workspace URL.

   1. Open your Databricks workspace in the browser, for example:

      ```
      https://adb-90442919623923.3.azuredatabricks.net/
      ```

      or

      ```
      https://adb-90442919623923.3.azuredatabricks.net/?o=90442919623923
      ```

   2. Locate the parameter ?o= in the URL, for example:

      https://mycompany.cloud.databricks.com/?o=90442919623923 # Workspace ID = 90442919623923
   3. If you do not find the ?o= parameter, go to Sidebar Data Science & Engineering .

      The URL will update to include the workspace ID:

      ```
      https://mycompany.cloud.databricks.com/?o=90442919623923#workspace/
      ```

   This verifies the workspace ID value.

Related Information

Databricks Lineage in Cloudera Data Lineage

Configure Databricks authentication in Cloudera Data Lineage

Learn about the authentication methods available for connecting Cloudera Data Lineage (formerly Octopai) to Databricks, including machine-to-machine authentication using service principals and user authentication using Personal Access Tokens.

Cloudera Data Lineage supports two authentication approaches for Databricks integration, applicable to Unity Catalog, Hive Metastore, or hybrid (Unity Catalog and Hive Metastore) deployment.

Option 1: Machine-to-machine authentication (service principal)

This is the recommended approach for production deployments.

For environments using Unity Catalog, you must configure a service principal. This is because Cloudera Data Lineage must authenticate using an identity with permission to query Unity Catalog system lineage tables.

To configure Databricks authentication, ensure the following:

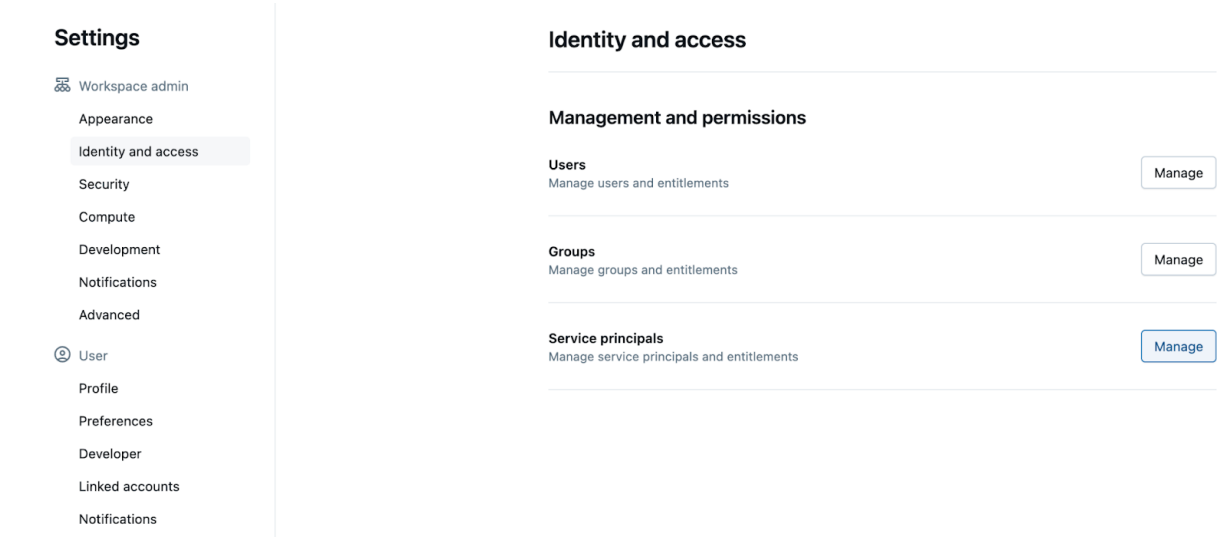
- Create a Databricks service principal.
- Assign it to the workspace.
- Grant the required Unity Catalog and system table permissions.
- Generate OAuth credentials for secure access.

This method enables automated extraction without relying on a personal user account.

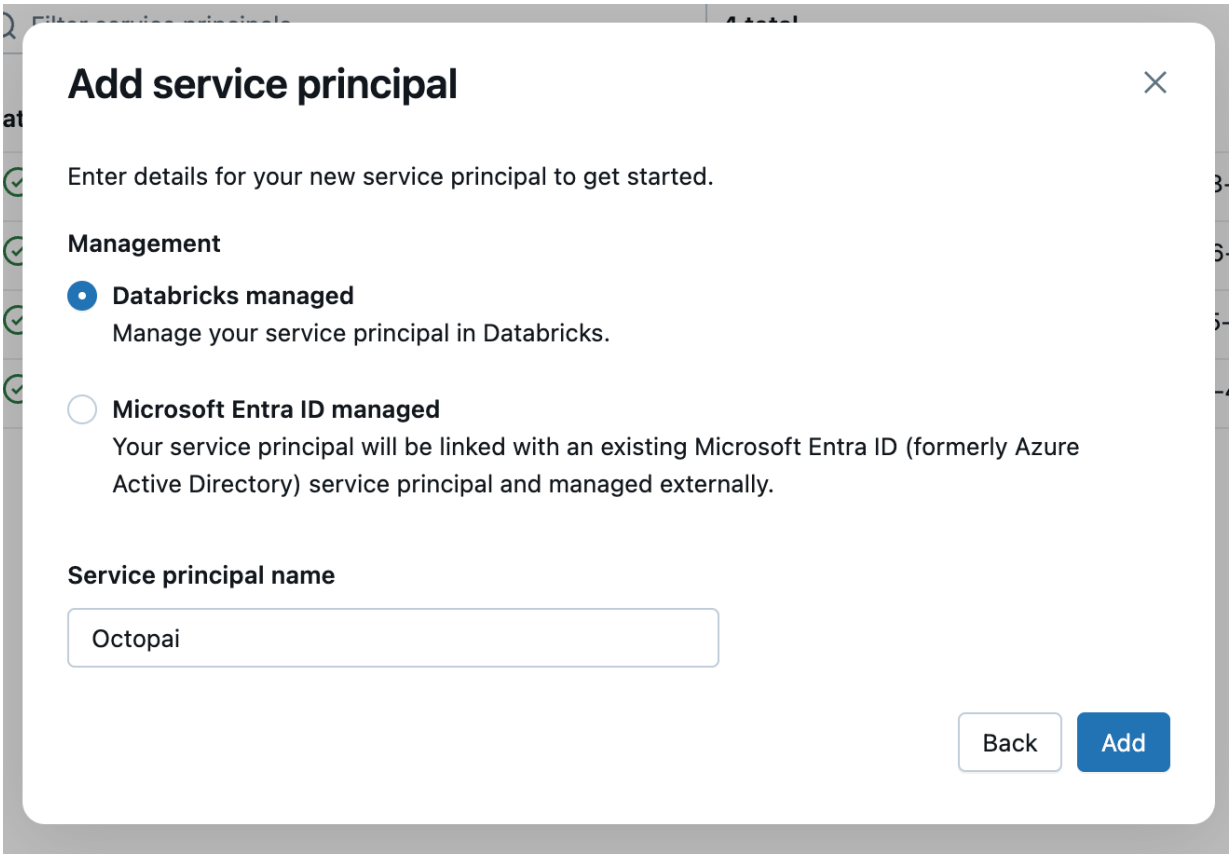
Create a dedicated service principal

To create a dedicated service principal, perform the following steps:

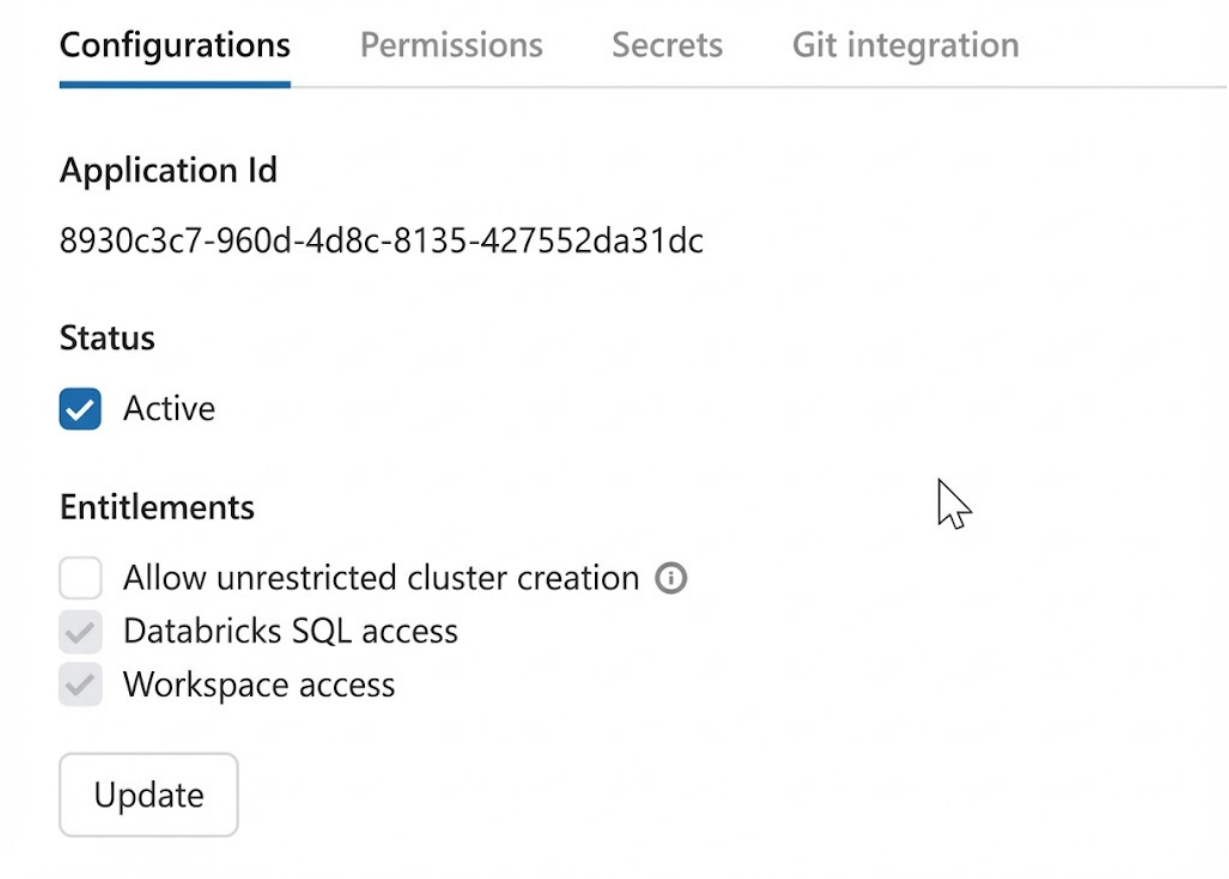
1. In the Databricks workspace, go to Settings.
2. Go to Identity and access Service principals .



3. Click Manage, then select Add service principal.
4. Choose Databricks managed and assign a descriptive name (for example, octopai).



5. Open the created service principal and go to the Configurations tab.
6. Select Databricks SQL access and Workspace access.



Option 2: User authentication token (Personal Access Token)

Alternatively, you can authenticate using a Databricks user token (applicable only for HMS).

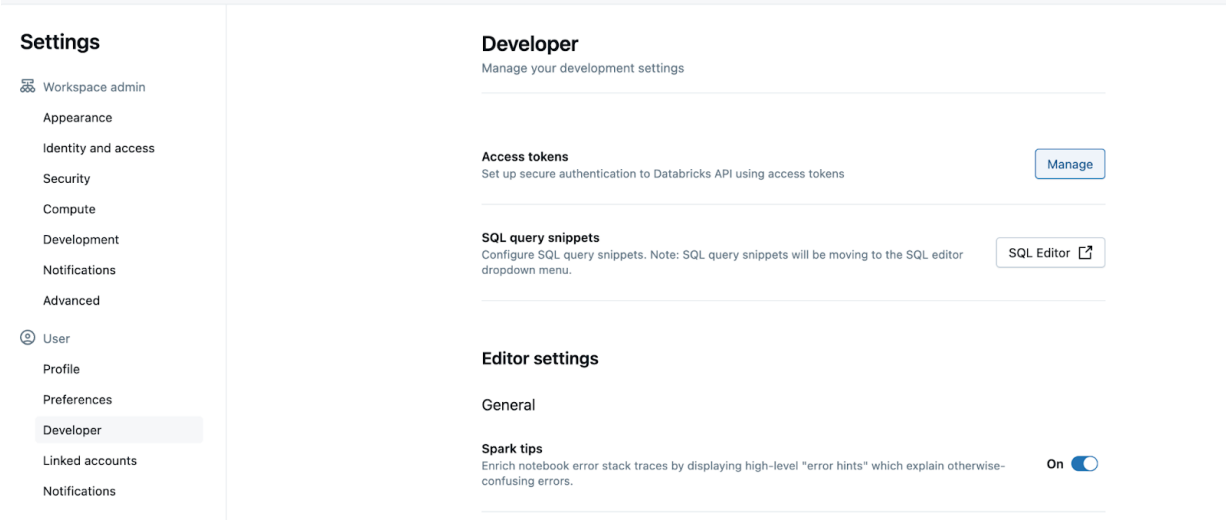
You must ensure the following:

- Generate a Personal Access Token (PAT).
- Provide the token during the Cloudera Data Lineage setup.

Generate a Personal Access Token

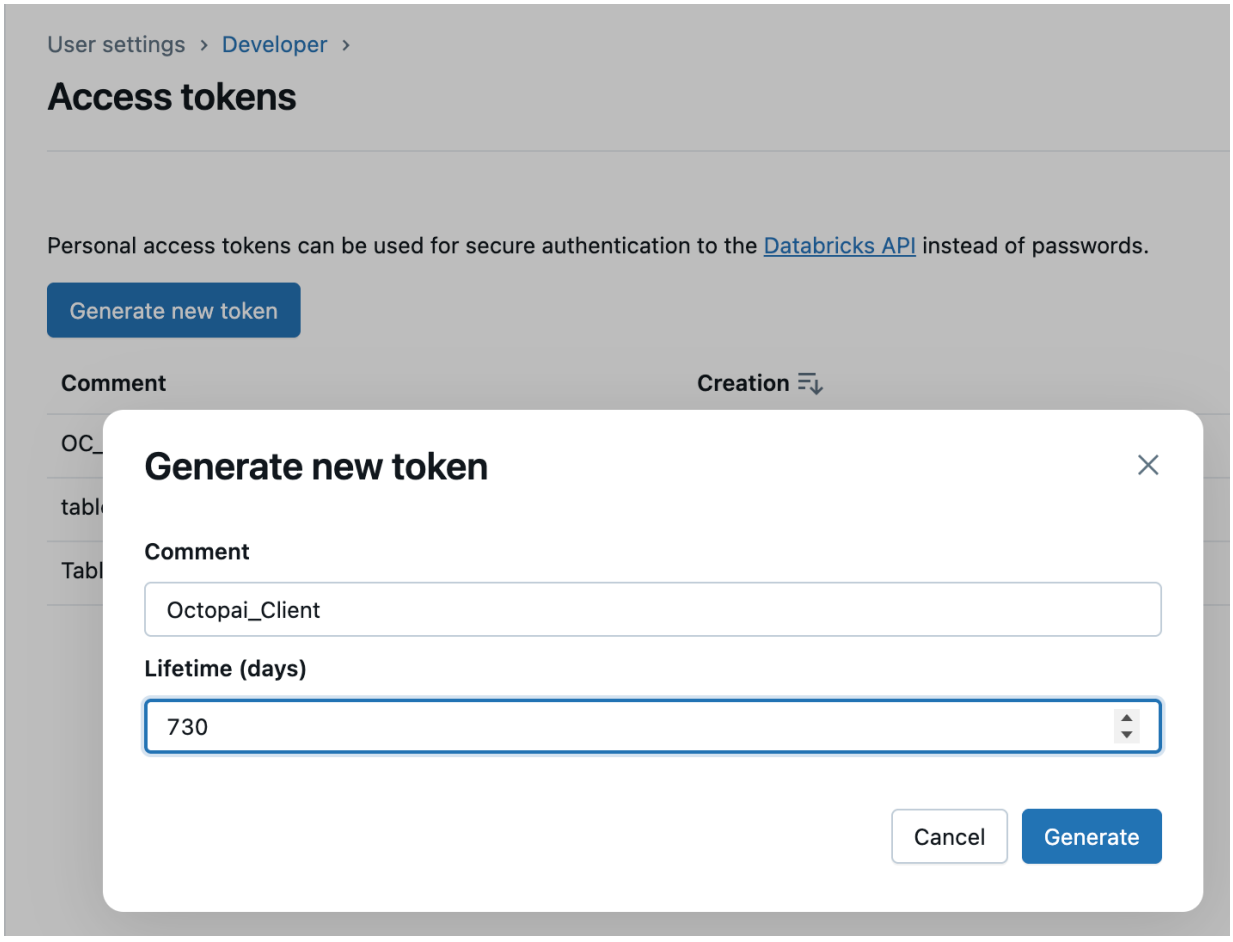
Perform the following steps to generate a Personal Access Token:

1. In Databricks, go to Settings Developer , then select the Manage button next to Access tokens.



2. Click Generate new token.
3. Set the maximum lifespan for the token.

Note: The token must be periodically regenerated.



Related Information

Databricks Lineage in Cloudera Data Lineage

Configure Databricks Metadata Source in Cloudera Data Lineage

Learn how to configure the Databricks Metadata Source in Cloudera Data Lineage (formerly Octopai) using either user authentication with Personal Access Tokens or machine-to-machine authentication with service principals.

Cloudera Data Lineage supports two authentication methods for connecting to Databricks:

- User authentication using a Personal Access Token
- Machine-to-machine (M2M) authentication using a service principal

Important: Select one of the methods when configuring your connection.

Option 1: User authentication token (Personal Access Token)

Figure 37: Databricks metadata source configuration for Hive Metastore using Personal Access Token

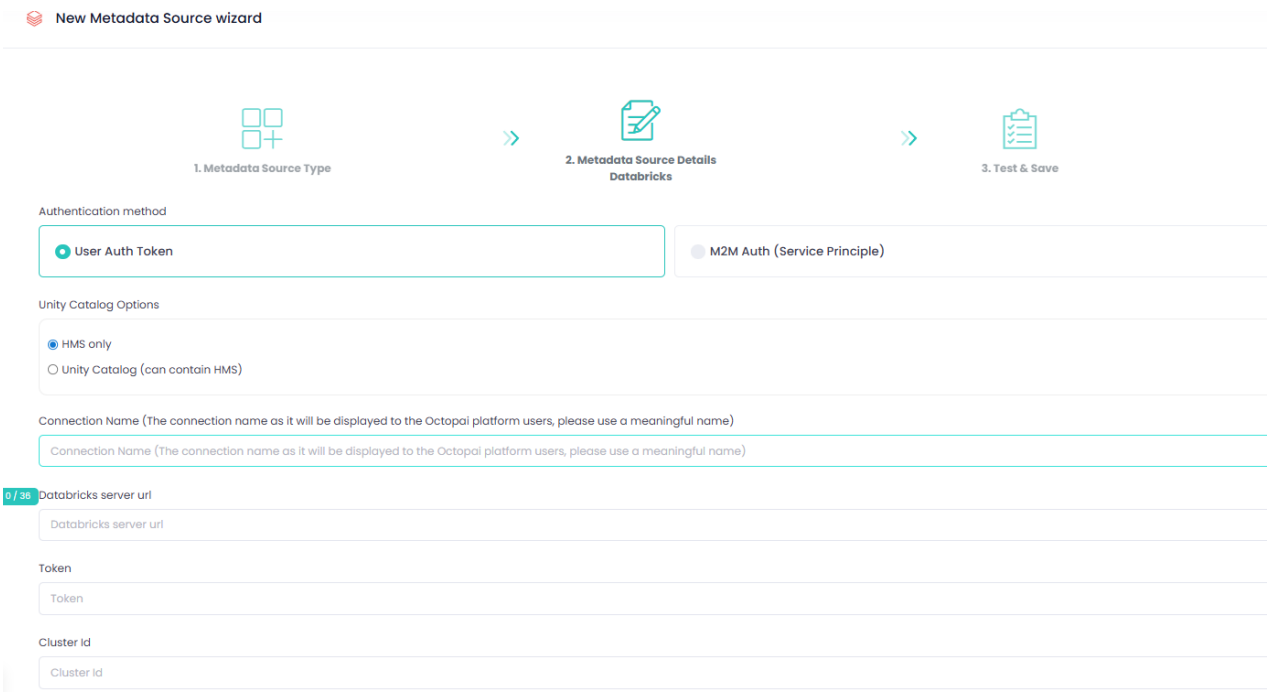
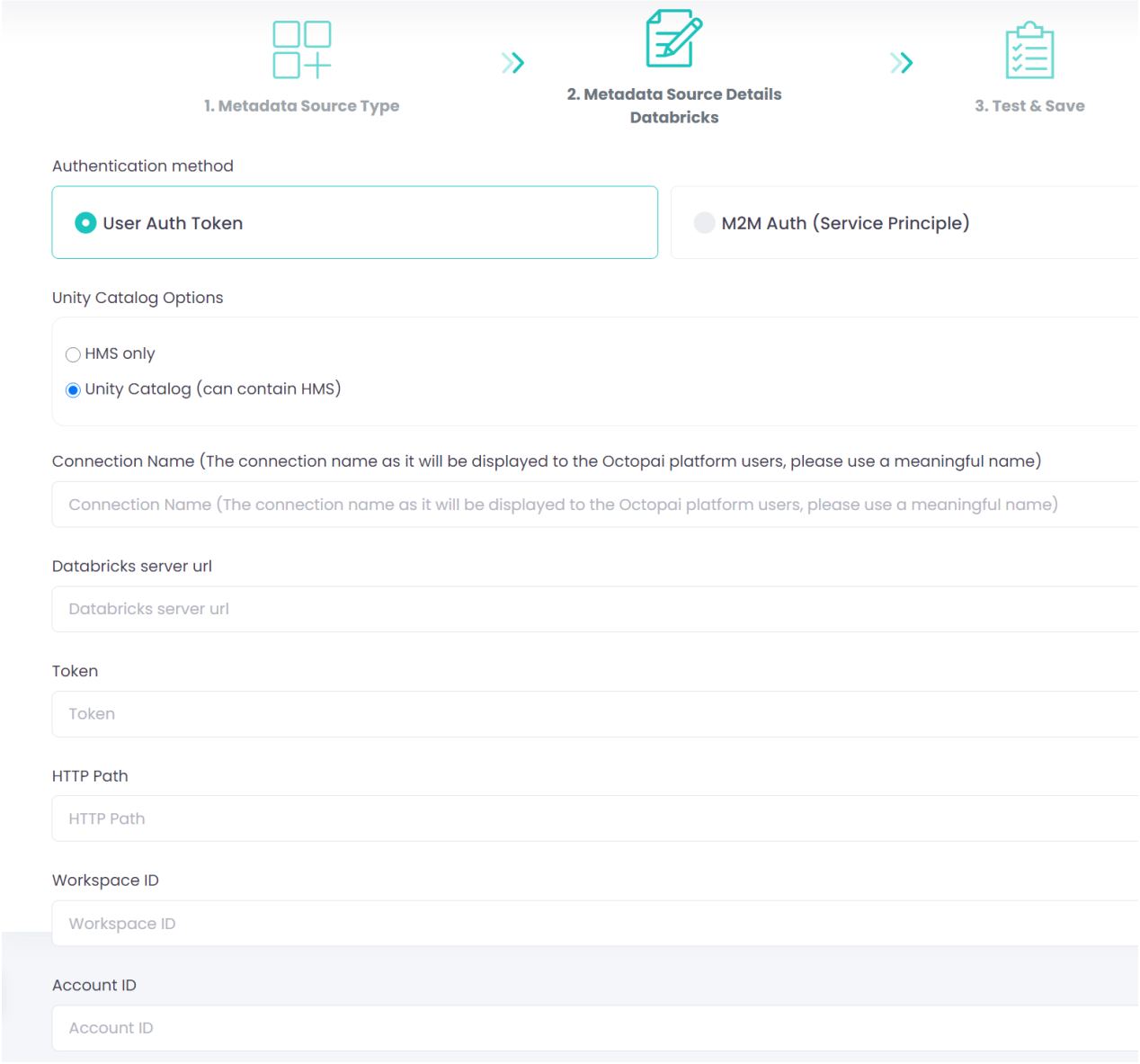


Figure 38: Databricks metadata source configuration for Unity Catalog using Personal Access Token



Configure the following settings when using the Personal Access Token authentication method:

1. Unity Catalog Options
   - HMS only – when Databricks uses Hive Metastore without Unity Catalog.
   - Unity Catalog (can contain HMS) – when Databricks uses Unity Catalog. Hive Metastore can also be used (not mandatory).
2. Connection Name

   Assign a clear and meaningful name for the connection. This name will appear to users within the Cloudera Data Lineage platform.
3. Databricks Server URL

   Enter the customer's Databricks workspace URL.

   Example: https://abc-1234.5.azuredatabricks.net
4. Token

   Enter the Personal Access Token generated under SettingsDeveloperAccess Tokens (Manage) in Databricks.
5. HTTP Path (for Unity Catalog only)

Paste the HTTP Path copied from the Databricks SQL Warehouse Connection Details field.

Example: /sql/1.0/warehouses/abc123xyz

6. Workspace ID (for Unity Catalog only)
7. Account ID (for Unity Catalog only, optional)
8. Cluster ID (for Hive Metastore only, optional)

The Cluster ID identifies the compute context where metadata queries run. Cloudera recommends that you provide a running cluster for full metadata extraction.

If you do not supply a running cluster (and Cluster ID), Cloudera Data Lineage can still generate lineage; however, it cannot retrieve table-level metadata stored in the Hive Metastore (such as table definitions and detailed schema information). As a result, Cloudera Data Lineage displays lineage with limited table metadata.

To retrieve the Cluster ID:

a. Navigate to Compute.
b. Select the relevant cluster.
c. Copy the Cluster ID from the URL.

Example:

https://abc-1234.5.azuredatabricks.net/compute/clusters/123-11568975-2zabcde?o=71234567896

The Cluster ID in this example is 123-11568975-2zabcde.

Option 2: Machine-to-machine authentication (service principal)

Figure 39: Databricks metadata source configuration for Hive Metastore using service principal

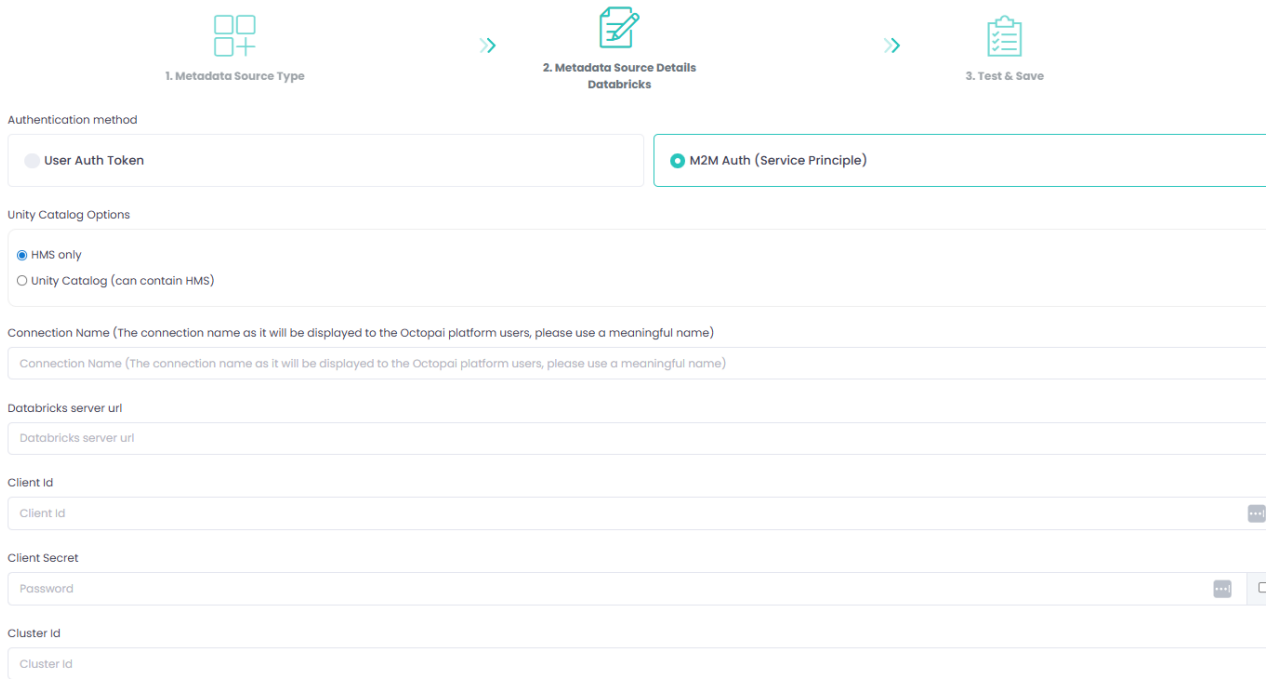
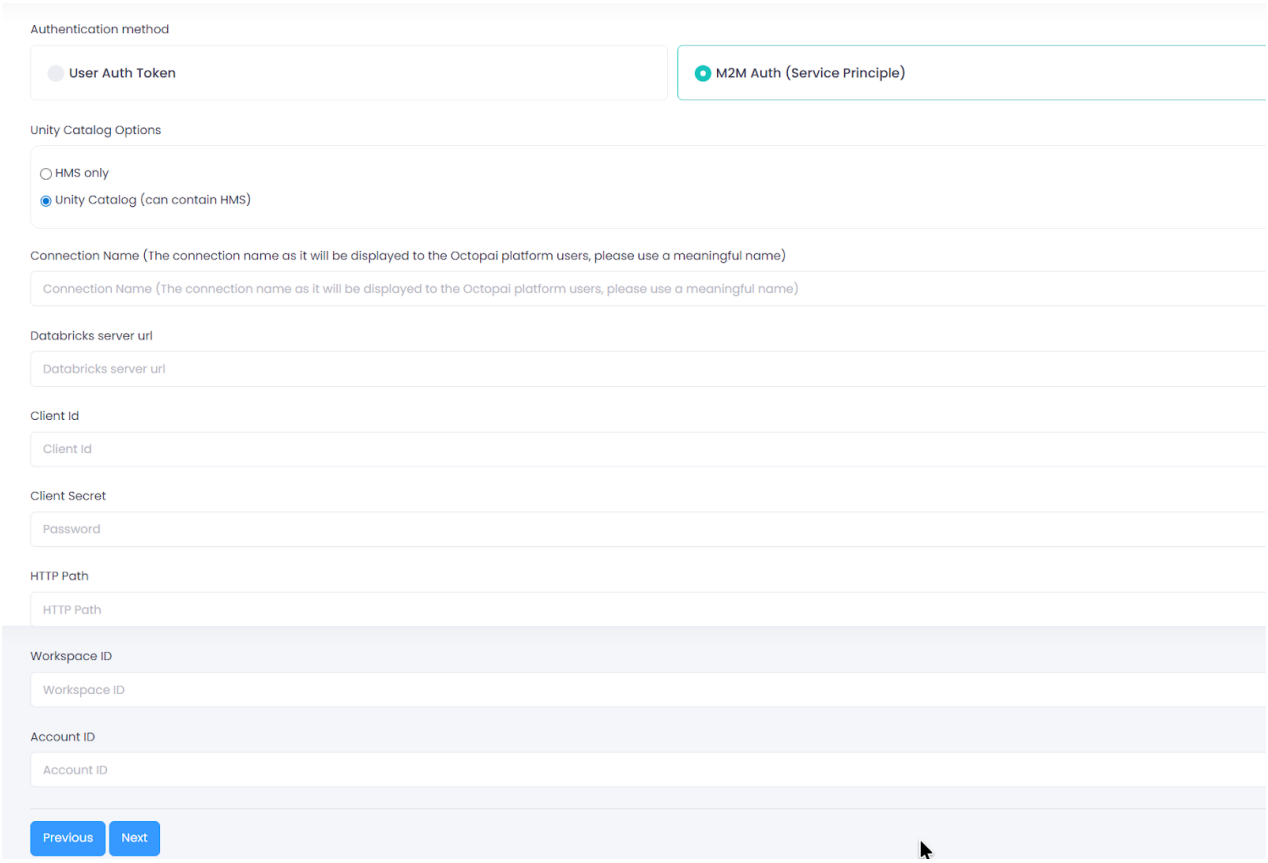


Figure 40: Databricks metadata source configuration for Unity Catalog using service principal



Configure the following settings when using the service principal authentication method:

1. Unity Catalog Options
   - HMS only – when Databricks uses Hive Metastore without Unity Catalog.
   - Unity Catalog (may include HMS) – when Databricks uses Unity Catalog. Hive Metastore can also be used but is not mandatory.
2. Connection Name

   Assign a clear and meaningful name for the connection. This name will appear to users within the platform.
3. Databricks Server URL

   Enter the customer's Databricks workspace URL.

   Example: https://abc-1234.5.azuredatabricks.net
4. Client ID

   Enter the Client ID of the service principal created in Databricks.
5. Client Secret

   Enter the secret token generated for the service principal.
6. HTTP Path (for Unity Catalog only)

   Paste the HTTP Path copied from the Databricks SQL Warehouse Connection Details field.

   Example: /sql/1.0/warehouses/abc123xyz
7. Workspace ID (for Unity Catalog only)
8. Account ID (for Unity Catalog only, optional)
9. Cluster ID (for Hive Metastore only, optional)

   The Cluster ID identifies the compute context where metadata queries run. Cloudera recommends that you provide a running cluster for full metadata extraction.

If you do not supply a running cluster (and Cluster ID), Cloudera Data Lineage can still generate lineage; however, it cannot retrieve table-level metadata stored in the Hive Metastore (such as table definitions and detailed schema information). As a result, Cloudera Data Lineage displays lineage with limited table metadata.

To retrieve the Cluster ID:

a. Navigate to Compute.
b. Select the relevant cluster.
c. Copy the Cluster ID from the URL.

Example:

https://abc-1234.5.azuredatabricks.net/compute/clusters/123-11568975-2zabcde?o=71234567896

The Cluster ID in this example is 123-11568975-2zabcde.

Related Information

Databricks Lineage in Cloudera Data Lineage

Databricks Lineage in Cloudera Data Lineage

Cloudera Data Lineage (formerly Octopai) provides automated data lineage and metadata visibility for Databricks environments, supporting Unity Catalog, Hive Metastore, and hybrid configurations. It extracts lineage from system metadata and notebook parsing, offering insights into data flows across pipelines, notebooks, and downstream systems.

Overview

Cloudera Data Lineage automates data lineage and metadata visibility for Databricks environments. Support for Databricks in Cloudera Data Lineage accommodates various catalog architectures, including Unity Catalog, Hive Metastore, and hybrid deployments that combine Unity Catalog and Hive Metastore.

Cloudera Data Lineage extracts lineage directly from Databricks system metadata and enhances it with advanced parsing capabilities for notebook-based workloads. This enables organizations to gain deeper insights into data flows across Databricks pipelines, notebooks, and downstream systems.

Supported Databricks catalog configurations

Cloudera Data Lineage supports lineage extraction from the following Databricks catalog setups:

Unity Catalog

Unity Catalog is Databricks' centralized governance layer. When Unity Catalog is enabled, Cloudera Data Lineage extracts lineage using Databricks system lineage tables and catalog metadata, as well as other parsing capabilities.

Unity Catalog is most suitable for customers using it as their primary metastore.

Hive Metastore

For Databricks workspaces still using the legacy Hive Metastore, Cloudera Data Lineage supports metadata and lineage extraction without relying on Unity Catalog system tables.

Hive Metastore is most suitable for customers not yet migrated to Unity Catalog.

Hybrid: Unity Catalog and Hive Metastore

Some environments operate with both Unity Catalog and Hive Metastore simultaneously. Cloudera Data Lineage supports these hybrid deployments and can provide lineage across both catalogs, offering broader visibility compared to native Databricks capabilities.

Hybrid configurations are most suitable for customers transitioning from Hive Metastore to Unity Catalog.

Lineage coverage and supported workloads

Cloudera Data Lineage supports lineage extraction from Databricks environments using either Unity Catalog system metadata, notebook parsing, or a combination of both. Lineage behavior is determined by the catalog configuration of the customer's workspace.

Supported notebook languages

Cloudera Data Lineage supports SQL, Python, and PySpark notebook lineage consistently across Databricks environments. Lineages for Scala and R are available only when Unity Catalog system metadata provides lineage records.

This table provides an overview of the lineage capabilities available in different Databricks catalog configurations, including Unity Catalog, Hive Metastore, and hybrid (Unity Catalog and Hive Metastore) environments.

Catalog Configuration	SQL	Python	PySpark	Scala	R
Unity Catalog	Supported	Supported	Supported	Supported when Databricks provides lineage	Supported when Databricks provides lineage
Hive Metastore	Supported	Supported	Supported	Not supported	Not supported
Unity Catalog and Hive Metastore (hybrid)	Supported	Supported	Supported	Unity Catalog-only when Databricks provides lineage	Unity Catalog-only when Databricks provides lineage

Lineage behavior by catalog type

Cloudera Data Lineage Databricks lineage extraction varies based on whether the workspace uses Unity Catalog, Hive Metastore, or both.

Unity Catalog lineage (including hybrid Unity Catalog and Hive Metastore deployments)

In Unity Catalog environments, Cloudera Data Lineage integrates system lineage metadata with notebook parsing to deliver comprehensive coverage. It uses two complementary sources:

- Databricks Unity Catalog system lineage tables
- Notebook-level parsing for supported scripts, including SQL, Python, and PySpark

This feature provides the following benefits:

- Authoritative lineage recorded natively by Databricks
- Additional lineage relationships derived from notebook code analysis

Native Unity Catalog lineage:

Unity Catalog system lineage captures persistent, governed operations, including reads and writes to managed tables.

Operations that do not produce persistent table or storage writes, such as intermediate DataFrame transformations, in-memory processing, or pandas-based manipulations, may not appear in Unity Catalog lineage metadata.

Notebook parsing enhancement:

Cloudera Data Lineage also parses notebook scripts to enrich lineage coverage, including cases where native system lineage may be incomplete.

Temporary views or non-persistent transformations may not appear in Unity Catalog system metadata, but may still be partially reflected through parsing where possible.

Hybrid (Unity Catalog and Hive Metastore) environments:

In hybrid environments, Cloudera Data Lineage uses the same Unity Catalog extraction approach while also including Hive Metastore assets. This approach offers broader visibility across both catalog types compared to the Databricks native lineage UI.

Hive Metastore lineage support (Hive Metastore only):

In Hive Metastore-only environments, Cloudera Data Lineage derives lineage primarily through notebook script parsing.

Cloudera Data Lineage supports lineage extraction from the following notebook types:

- Python notebooks
- PySpark notebooks
- SQL notebooks

Lineage is determined based on the notebook code.

In Hive Metastore-only workspaces, a compute resource must be available to execute metadata queries. The Cluster ID specifies the compute context where these commands execute. The cluster serves as the execution engine that queries the Hive Metastore and returns the results.

If no cluster is running, Spark cannot execute metadata queries, and you cannot retrieve metadata. Therefore, Hive Metastore-only environments require a running cluster to access metadata programmatically.

Script-based parsing limitations:

In Hive Metastore environments, lineage is inferred from notebook scripts, resulting in the following limitations:

- Highly dynamic transformations (such as code-generated queries, function-driven logic, user-defined functions (UDFs), loops that write files programmatically, or indirect write operations) can limit lineage resolution or prevent full identification of sources and targets..
- Lineage resolution requires explicit table and column references in the code.
- If a table is referenced without a fully qualified database or schema name, Cloudera Data Lineage might not be able to resolve the database context. For example, the query SELECT * FROM sales_table might appear in the lineage without its associated database.
- Fully qualified references, such as db.schema.sales_table, provide the most complete lineage results.

In Hive Metastore environments, Cloudera Data Lineage does not currently support lineage for the following cases:

- Scala notebooks
- R notebooks
- Databricks pipelines or jobs

If you do not supply a running cluster (and Cluster ID), Cloudera Data Lineage can still generate lineage; however, it cannot retrieve table-level metadata stored in the Hive Metastore (such as table definitions and detailed schema information). As a result, Cloudera Data Lineage displays lineage with limited table metadata.

Column-level lineage support (Unity Catalog and Hive Metastore):

Column-level lineage is supported when column mappings are available in the metadata source:

- In Unity Catalog environments, column lineage is captured primarily from Databricks system lineage tables. It may also be supplemented by notebook parsing if column references are explicitly defined.
- In Hive Metastore environments, columns are captured only when explicitly referenced in notebook scripts (for example, within SQL queries).

Shared platform limitations

The following limitations apply across both Unity Catalog and Hive Metastore environments:

Pipelines and jobs not represented as entities

Databricks jobs and pipelines are not currently modeled as standalone entities in Cloudera Data Lineage lineage. In Unity Catalog environments, lineage generated by jobs or pipelines is captured. However, the lineage is reported through the underlying notebook activity instead of appearing as a separate job or pipeline asset.

Cloudera Data Lineage captures lineage at the notebook execution level and focuses on the underlying data transformations, rather than modeling orchestration constructs (such as jobs or pipelines) as standalone lineage entities.

Unity Catalog environments provide the most complete lineage coverage. Hive Metastore environments rely entirely on notebook parsing and require explicit table and column references.

Microsoft Fabric Lakehouse [Technical Preview]

The Cloudera Data Lineage (formerly Octopai) Microsoft Fabric Lakehouse connector is used to integrate Microsoft Fabric Lakehouse environments into the Cloudera Data Lineage platform, enabling centralized metadata management and comprehensive data lineage.



Note: This feature is in Technical Preview and is not ready for production deployments. Cloudera recommends trying this feature in test or development environments and encourages you to provide feedback on your experiences.

Overview

Before you set up the Microsoft Fabric Lakehouse connector, ensure that you have the following requirements:

- **Microsoft Entra application credentials**: a registered service principal with a tenant ID, application (client) ID, and client secret. See Application registration (service principal).
- **Microsoft ODBC Driver 18 for SQL Server**: installed on the Windows machine that hosts the Cloudera Octopai Client for SQL endpoint connectivity. See SQL access (ODBC).
- **Microsoft Fabric API permissions**: application API permissions assigned to the service principal, including Lakehouse.Read.All and related scopes. See Required application permissions.
- **Fabric workspace permissions**: the Contributor role assigned to the service principal on the target workspace. See Workspace permissions.
- **OneLake access**: required for extracting metadata from Lakehouse files, folders, and file shortcuts through the Microsoft OneLake Distributed File System (DFS) endpoint.
- **Shortcut access**: required when the Lakehouse contains shortcuts[*] to another workspace or to external storage.

Microsoft Fabric Lakehouse metadata extraction uses both the Lakehouse SQL endpoint and OneLake APIs. The SQL endpoint is used for Delta tables, columns, schemas, views, stored procedures, and functions. OneLake access is used for files, folders, and file shortcuts.

Application registration (service principal)

Perform the following steps to register a Microsoft Entra service principal.

1. In the Azure portal, open Microsoft Entra ID App registrations , and select New registration.
2. Provide a name for the application, and select Accounts in this organizational directory only.



Note: Redirect URI is optional and is not needed for client credentials.

3. Select Register.
4. From the application overview page, record the Directory (tenant) ID and Application (client) ID.
5. Create a Client secret under Manage Certificates and secrets .

The Cloudera Data Lineage Microsoft Fabric Lakehouse connector authenticates using a Microsoft Entra service principal. The service principal is used for:

[*] The Microsoft Fabric Lakehouse connector supports data lineage for OneLake shortcuts.

- Microsoft Fabric REST API access
- SQL endpoint access through ODBC using Microsoft Entra authentication

The following values are required while configuring the connector:

- Tenant ID
- Application (client) ID
- Client Secret Value

SQL access (ODBC)

Microsoft Fabric Lakehouse metadata extraction requires SQL connectivity through the Lakehouse SQL endpoint. Install Microsoft ODBC Driver 18 for SQL Server on the Windows machine that hosts the Octopai Client. For Windows, see Download ODBC Driver for SQL Server.

Identity prerequisites

A Microsoft Entra application registration (service principal) is required and is used for:

- API authentication
- SQL connectivity through the Lakehouse SQL endpoint
- Access to Fabric workspace items through Microsoft Fabric APIs
- Access to Lakehouse files and folders through OneLake APIs

Tenant settings (Fabric admin portal)

Use the Fabric admin portal to enable the settings required by the connector:

1. Open Microsoft Fabric.
2. Select Settings, and then select Admin portal.
3. Open Tenant settings.
4. Find and enable the Service principals can call Fabric public APIs option.
5. Find and enable the Allow apps to access Fabric items option.
6. Find and enable the Users can access data stored in OneLake with apps external to Fabric option.

Workspace permissions

Assign the service principal to the target workspace with the Contributor role. Contributor access is required for complete metadata extraction coverage, including:

- Stored procedures
- Views
- Functions
- Tables
- Columns
- Schemas

The Viewer role is not sufficient for full metadata extraction because it may not expose stored procedures, views, and functions.



Important: Cloudera recommends Contributor access for all Cloudera Data Lineage Microsoft Fabric Lakehouse integrations. Lower-privilege roles result in incomplete metadata extraction.



Note: Cloudera Data Lineage uses Contributor access in a read-only manner. The connector only reads and extracts metadata. It does not create, update, delete, or modify objects in the workspace.

Required application permissions

The application requires the following Microsoft Fabric API permissions:

- Lakehouse.Read.All
- Workspace.Read.All
- SQLEndpoint.Read.All
- Item.Read.All

Network requirements

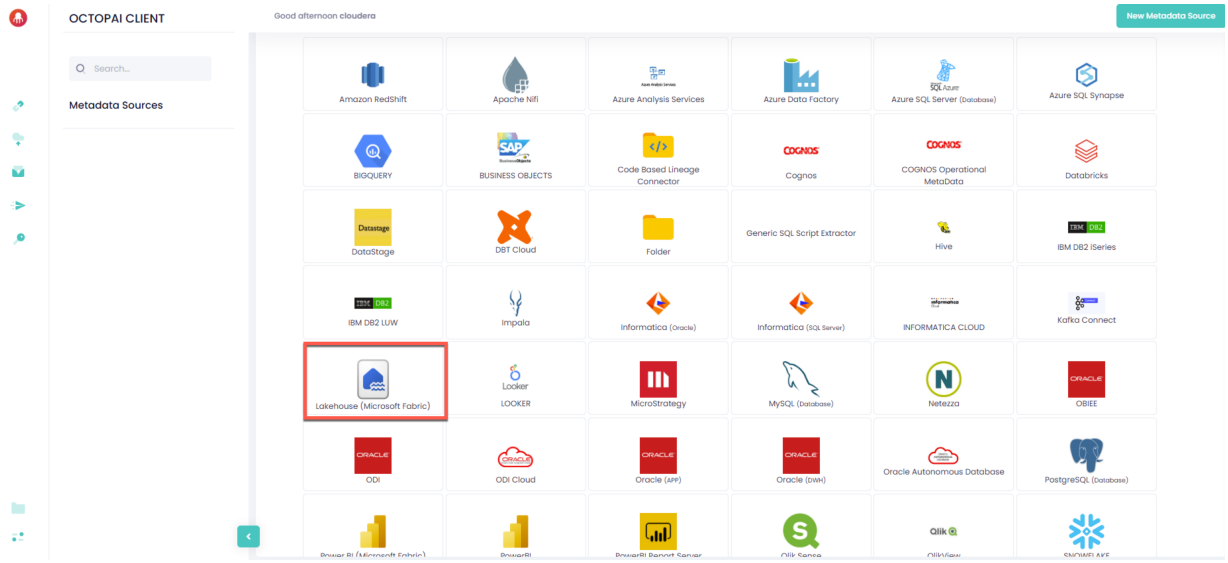
The extraction environment must allow outbound network access from the Windows machine that hosts the Cloudera Data Lineage platform to the following endpoints:

```
api.fabric.microsoft.com
*.fabric.microsoft.com
*.datawarehouse.fabric.microsoft.com
onelake.dfs.fabric.microsoft.com
login.microsoftonline.com
```

Setting up Lakehouse (Microsoft Fabric) metadata source

Metadata sources are set on the Octopai Client.

1. From the Octopai Client, click the Lakehouse (Microsoft Fabric) connector.



2. In the **New Metadata Source** wizard, select the Service Principal authentication method and enter the following details:
   - Connection Name: Enter a meaningful name. The name is displayed to Cloudera Data Lineage platform users.
   - Application (client) ID
   - Tenant ID
   - Client Secret Value
   - Workspace ID

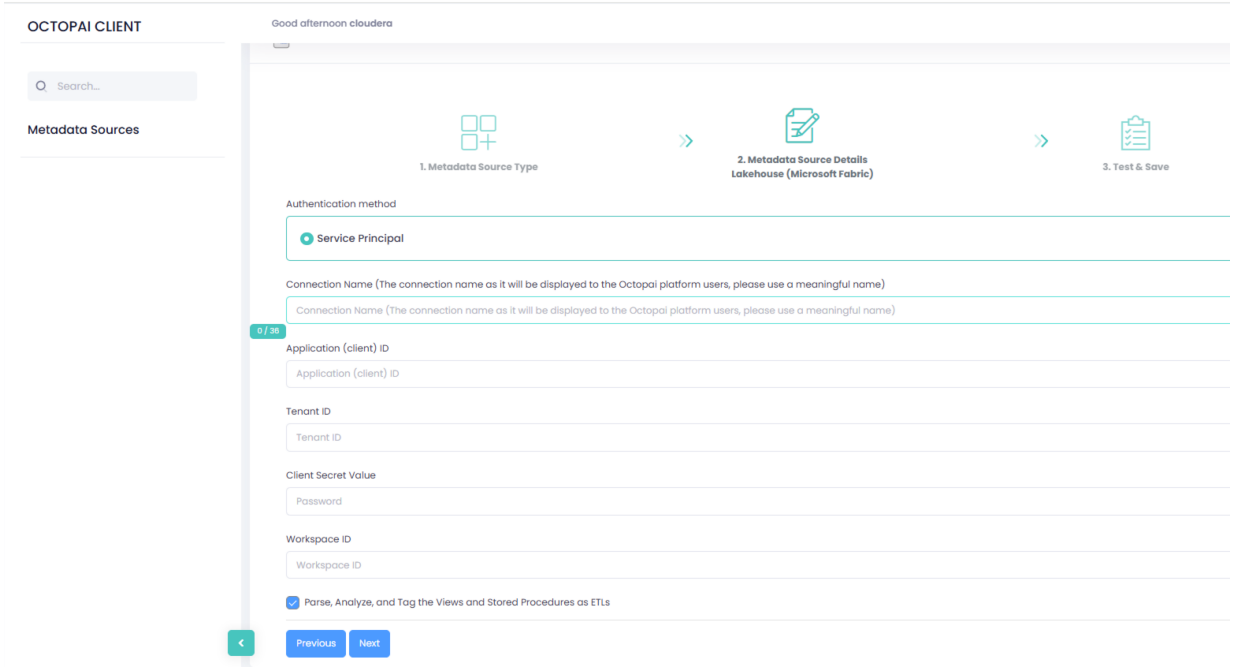
Note: To locate your Workspace ID in Microsoft Fabric:

a. Open Microsoft Fabric.
b. Open the target workspace.
c. In your browser address bar, locate the URL. The URL follows this pattern:

```
https://app.fabric.microsoft.com/groups/[***WORKSPACE ID***]/list?experience=power-bi
```

d. Copy the value that appears after /groups/. This value is your Workspace ID.

- Parse, Analyze, and Tag the Views and Stored Procedures as ETLs: (optional) Select this option when you want Cloudera Data Lineage to treat views and stored procedures as ETL objects during analysis.



Hive, Impala, and HMS

Cloudera Data Lineage extracts Apache Hive and Apache Impala metadata through HiveServer or the Hive Metastore (HMS), using Kerberos or username and password authentication.

Apache Hive connector

The Apache Hive connector extracts database, table, column, and view metadata from Apache Hive into Cloudera Data Lineage (formerly Octopai), using either HiveServer or the Hive Metastore (HMS) as the metadata source.

The Apache Hive connector reads metadata from Apache Hive so that Cloudera Data Lineage can build lineage and catalog information. You can choose one of two metadata sources, depending on your environment:

HiveServer

Connects to HiveServer2 through an Open Database Connectivity (ODBC) data source name (DSN), using username and password or Kerberos authentication. For more information, see Configuring Hive connector with HiveServer as the metadata source

Hive Metastore

Connects directly to the HMS database that backs Hive and Impala For more information, see Configuring Hive connector with HMS as the metadata source.

Configuring Hive connector with HiveServer as the metadata source

Configure the Apache Hive connector in Octopai Client to extract metadata through HiveServer2 over an ODBC connection, using username and password or Kerberos authentication.

About this task

In HiveServer mode, Octopai Client connects to HiveServer2 through an Open Database Connectivity (ODBC) data source name (DSN). You configure the DSN once, choose either username and password or Kerberos authentication, and then add a matching connection in Octopai Client.

Before you begin



Warning: Missing permissions can result in broken connections or metadata extraction failures.

Ensure the following are available before you configure the connector:

- A Hive ODBC driver that matches your Octopai Client architecture (32-bit or 64-bit). You can download the driver from the Hive ODBC driver download page.
- For Kerberos authentication, MIT Kerberos for Windows. You can download the releveant distribution from MIT Kerberos Distribution. The default installation path is C:\Program Files\MIT\Kerberos\. Verify that C:\Program Files\MIT\Kerberos\bin\kinit.exe exists.
- For Kerberos authentication, the Kerberos configuration file krb5.ini from your administrator, placed at C:\ProgramData\MIT\Kerberos5\krb5.ini, with your realm and key distribution center (KDC), for example:

```
[libdefaults]
default_realm = YOUR.REALM
dns_lookup_realm = false
dns_lookup_kdc = false
ticket_lifetime = 24h
renew_lifetime = 7d
forwardable = true

[realms]
YOUR.REALM = {
  kdc = kdc.example.com
  admin_server = kdc.example.com
}

[domain_realm]
.example.com = YOUR.REALM
```

- For Kerberos authentication, a keytab file with the extension .keytab for your service principal, stored securely on the Octopai Client machine, for example C:\Octopai\keytabs\hive.keytab.

Procedure

1. Install the Hive ODBC driver, choosing the 32-bit or 64-bit version that matches your environment.
2. Open the ODBC Data Source Administrator and create a new system DSN.
   - Search for ODBC Data Source in the Start menu, and open the 32-bit or 64-bit version that matches the installed driver.
   - On the System DSN tab, click Add, select the Hive ODBC driver, and click Finish.
3. Configure the basic DSN settings.
   - Data Source Name: a descriptive name, for example Hive_Prod.

- Host: the hostname or IP address of the HiveServer2 service, for example hive.example.com.
- Port: the HiveServer2 port. The default is 10000; confirm the value in your cluster configuration.
- Database: a default database, for example default.

4. Configure the authentication for the DSN.
   - For username and password, set the authentication mechanism to User Name and Password and enter the database credentials.
   - For Kerberos, set the authentication mechanism to Kerberos and enter the service name (hive), the realm, the fully qualified domain name (FQDN) of the HiveServer2 host, and the Kerberos configuration path C:\ProgramData\MIT\Kerberos5\krb5.ini.
5. If the cluster requires secure connectivity, enable SSL or TLS and configure the truststore path and password according to the security policy of the cluster.
6. Click Test to test the DSN, and then click OK to save it.

   For a Kerberos DSN, a valid Kerberos ticket might be required for the test to succeed.
7. In Octopai Client, start the **New Metadata Source wizard**, select the Hive metadata source type, and on the **Metadata Source Details** step, under Metadata Source, select Hive Server.
8. For Authentication method, select the method that matches the DSN, and enter the connection details.
   - For Username and Password, enter a Connection Name, the DSN Name that you created, the UserName, and the Password.
   - For Kerberos, enter a Connection Name, the DSN Name that you created, the Keytab Path (for example, C:\Octopai\keytabs\hive.keytab), and the Kerberos Principal (for example, hive@YOUR.REALM).

Figure 41: Hive Server metadata source with Username and Password authentication

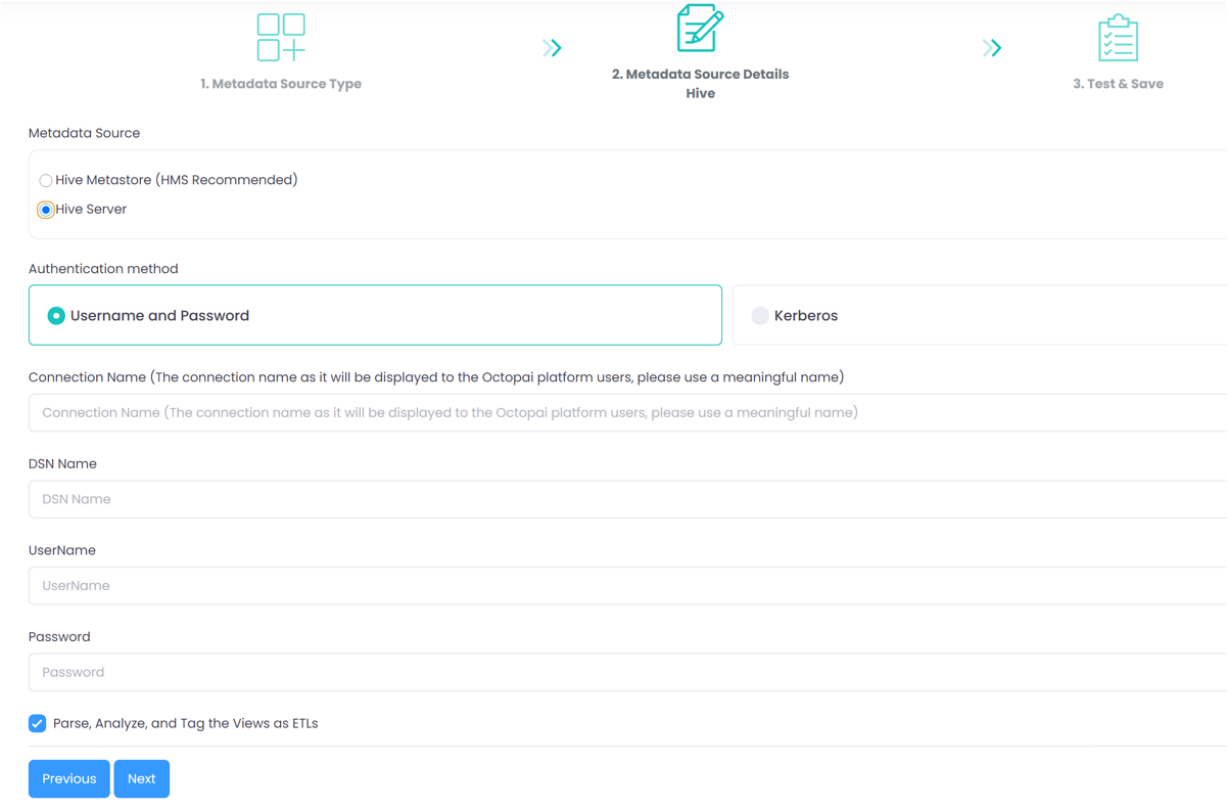
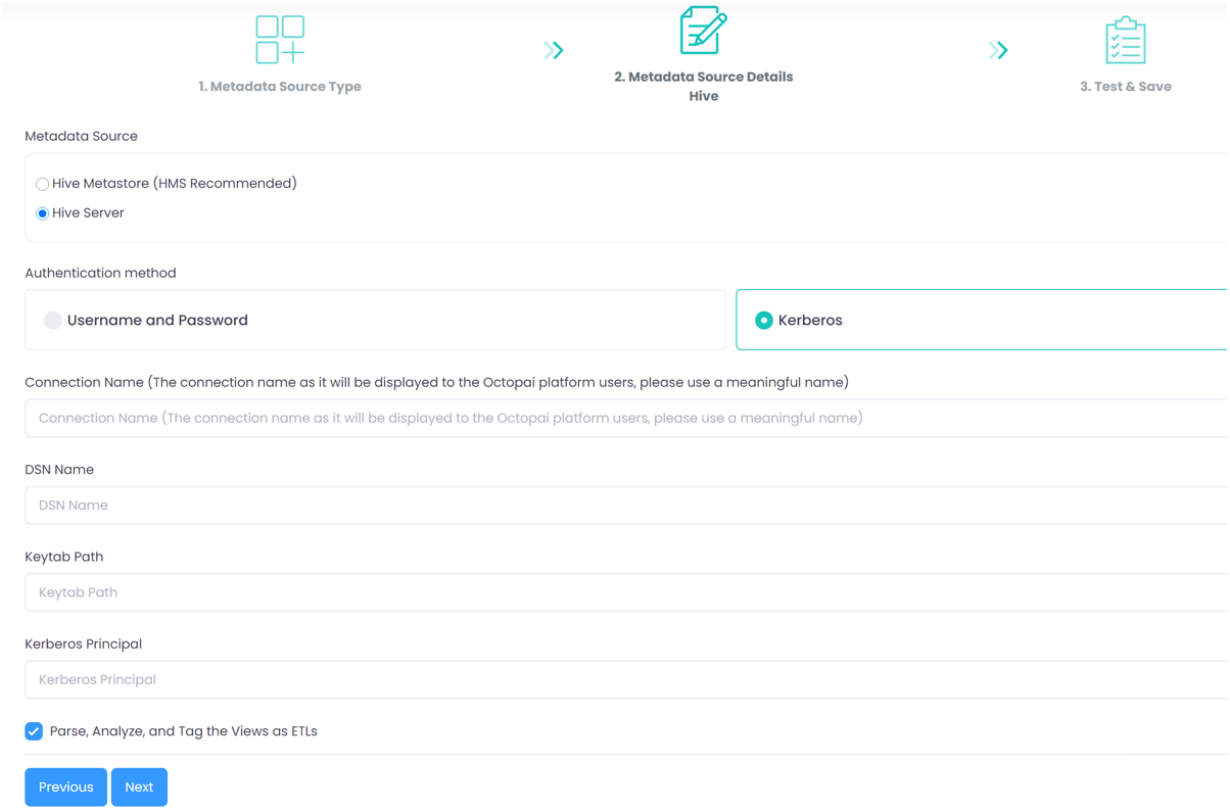


Figure 42: Hive Server metadata source with Kerberos authentication



9. Select the Parse, Analyze, and Tag the Views as ETLs check box to process views as ETLs.
10. Click Next, test the connection, and save it.

    For Kerberos, Octopai Client acquires a ticket by using `kinit` and the keytab, connects to Hive, and displays the connection status.

Results

To verify the extracted metadata, navigate to the target folder (TGT) on the server where Octopai Client is installed. The default location is C:\Program Files (x86)\Octopai\Service\TGT. Locate the compressed file for the Hive connector, for example Hive_Metadata_Export.zip, and verify the quantity and quality of the extracted files.

If the connection or extraction fails, verify the following and try again:

- Permissions on the Hive server and the ODBC connection.
- The DSN configuration, including the correct hostname, port, and authentication.
- For Kerberos, the krb5.ini configuration, the keytab, and the file permissions.
- That the HiveServer2 service is running and reachable.

If the problem persists, see Troubleshoot connector extraction failures.

Apache Impala connector

The Apache Impala connector extracts database, table, column, and view metadata from Apache Impala into Cloudera Data Lineage (formerly Octopai), using either HiveServer or the Hive Metastore (HMS) as the metadata source.

The Apache Impala connector reads metadata from Apache Impala so that Cloudera Data Lineage can build lineage and catalog information. You can choose one of two metadata sources, depending on your environment:

HiveServer

Connects to the Impala service through an Open Database Connectivity (ODBC) data source name (DSN), using username and password or Kerberos authentication. For more information, see Configuring Impala connector with HiveServer as the metadata source.

HMS

Connects directly to the Hive Metastore (HMS) database that backs Hive and Impala. For more information, see Configuring Impala connector with HMS as the metadata source.

Before you extract metadata, ensure that the required permissions are in place. For more information, see Apache Ranger permissions for Impala metadata extraction on page 179.

Configuring Impala connector with HiveServer as the metadata source

Configure the Apache Impala connector in Octopai Client to extract metadata through the Impala service over an ODBC connection, using username and password or Kerberos authentication.

About this task

In HiveServer mode, Octopai Client connects to the Impala service through an Open Database Connectivity (ODBC) data source name (DSN). You configure the DSN once, choose either username and password or Kerberos authentication, and then add a matching connection in Octopai Client.

Before you begin

Warning: Missing permissions can result in broken connections or metadata extraction failures. For the required Apache Ranger permissions, see Apache Ranger permissions for Impala metadata extraction.

Ensure the following are available before you configure the connector:

- An Impala ODBC driver that matches your Octopai Client architecture (32-bit or 64-bit). You can download the driver from the Impala ODBC driver download page.
- For Kerberos authentication, MIT Kerberos for Windows. You can download the releveant distribution from MIT Kerberos Distribution. The default installation path is C:\Program Files\MIT\Kerberos\. Verify that C:\Program Files\MIT\Kerberos\bin\kinit.exe exists.
- For Kerberos authentication, the Kerberos configuration file krb5.ini from your administrator, placed at C:\ProgramData\MIT\Kerberos5\krb5.ini, with your realm and key distribution center (KDC), for example:

```
[libdefaults]
default_realm = YOUR.REALM
dns_lookup_realm = false
dns_lookup_kdc = false
ticket_lifetime = 24h
renew_lifetime = 7d
forwardable = true

[realms]
YOUR.REALM = {
  kdc = kdc.example.com
  admin_server = kdc.example.com
}

[domain_realm]
.example.com = YOUR.REALM
```

- For Kerberos authentication, a keytab file with the extension .keytab for your service principal, stored securely on the Octopai Client machine, for example C:\Octopai\keytabs\impala.keytab.

Procedure

1. Install the Impala ODBC driver, choosing the 32-bit or 64-bit version that matches your environment.
2. Open the ODBC Data Source Administrator and create a new system DSN.

- Search for ODBC Data Source in the Start menu, and open the 32-bit or 64-bit version that matches the installed driver.
- On the System DSN tab, click Add, select the Impala ODBC driver, and click Finish.

3. Configure the basic DSN settings.
   - Data Source Name: a descriptive name, for example Impala_Prod.
   - Host: the hostname or IP address of the Impala service, for example impala.example.com.
   - Port: the Impala port. The default is 21050; confirm the value in your cluster configuration.
   - Database: a default database, for example default.
4. Configure the authentication for the DSN.
   - For username and password, set the authentication mechanism to User Name and Password and enter the database credentials.
   - For Kerberos, set the authentication mechanism to Kerberos and enter the service name (impala), the realm, the fully qualified domain name (FQDN) of the Impala service host, and the Kerberos configuration path C:\ProgramData\MIT\Kerberos5\krb5.ini.
5. If the cluster requires secure connectivity, enable SSL or TLS and configure the truststore path and password according to the security policy of the cluster.
6. Click Test to test the DSN, and then click OK to save it.

   For a Kerberos DSN, a valid Kerberos ticket might be required for the test to succeed.
7. In Octopai Client, start the **New Metadata Source wizard**, select the Impala metadata source type, and on the **Metadata Source Details** step, under Metadata Source, select Hive Server.
8. For Authentication method, select the method that matches the DSN, and enter the connection details.
   - For Username and Password, enter a Connection Name, the DSN Name that you created, the UserName, and the Password.
   - For Kerberos, enter a Connection Name, the DSN Name that you created, the Keytab Path (for example, C:\Octopai\keytabs\impala.keytab), and the Kerberos Principal (for example, impala@YOUR.REALM).

Figure 43: Hive Server metadata source with Username and Password authentication

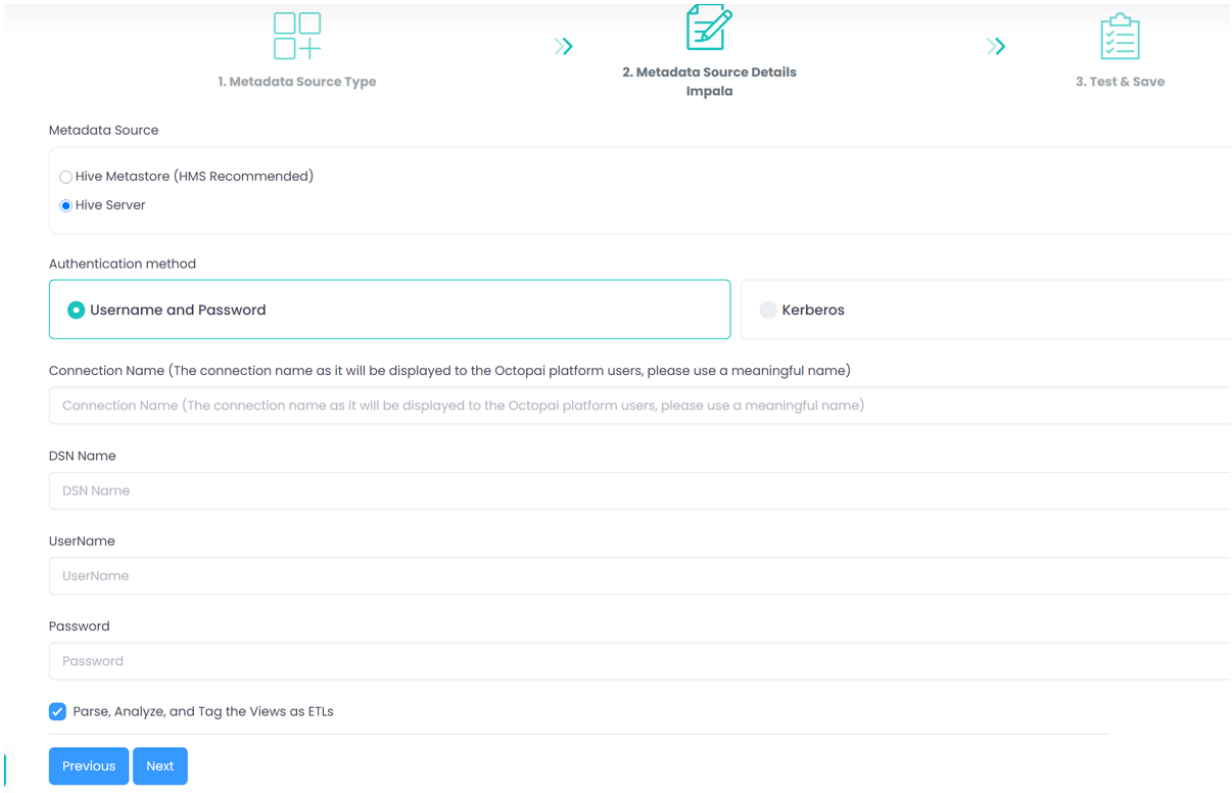
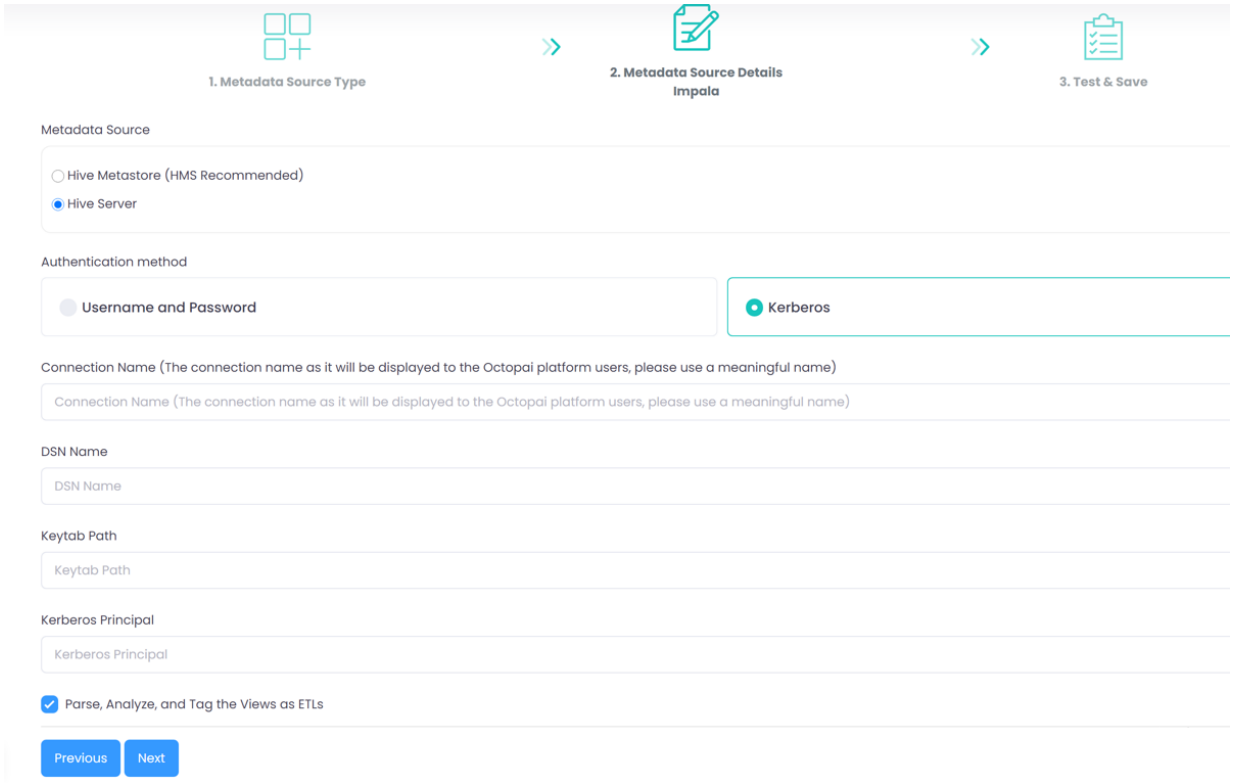


Figure 44: Hive Server metadata source with Kerberos authentication



9. Select the Parse, Analyze, and Tag the Views as ETLs check box to process views as ETLs.
10. Click **Next**, test the connection, and save it.

For Kerberos, Octopai Client acquires a ticket by using `kinit` and the keytab, connects to Impala, and displays the connection status.

Results

To verify the extracted metadata, navigate to the target folder (TGT) on the server where Octopai Client is installed. The default location is C:\Program Files (x86)\Octopai\Service\TGT. Locate the compressed file for the Impala connector, for example Impala_Metadata_Export.zip, and verify the quantity and quality of the extracted files.

If the connection or extraction fails, verify the following and try again:

- Permissions on the Impala server and the ODBC connection.
- The DSN configuration, including the correct hostname, port, and authentication.
- For Kerberos, the krb5.ini configuration, the keytab, and the file permissions.
- That the Impala service is running and reachable.

If the problem persists, see Troubleshoot connector extraction failures.

Apache Ranger permissions for Impala metadata extraction

The Cloudera Data Lineage (formerly Octopai) Impala extractor requires specific Apache Ranger permissions to extract database and table metadata.

Overview

The Cloudera Data Lineage Impala extractor requires specific Apache Ranger permissions to extract metadata from Impala databases and tables, including the minimum required permissions based on the extractor's operational scope.

To install the Impala ODBC driver and configure the DSN, see Impala.

Warning: If permissions are missing or insufficient, you might see broken connections or metadata extraction failures.

Recommended permissions

Grant the extractor principal the Impala REFRESH privilege on the required databases and tables in Apache Ranger. The REFRESH privilege is the least-privilege option that allows the extractor to run SHOW DATABASES, SHOW TABLES, DESCRIBE FORMATTED, SHOW CREATE TABLE, and SHOW CREATE VIEW without granting data-read access.

Extractor Scope

The Cloudera Data Lineage Impala extractor performs metadata-only SQL operations against Impala and Hive objects. It does not issue SELECT statements against table data.

The extractor runs the following SQL statements:

- SHOW DATABASES lists all databases.
- SHOW TABLES IN *[***DATABASE NAME***]* lists tables within a database.
- DESCRIBE FORMATTED *[***DATABASE NAME***].[***TABLE NAME***]* retrieves table metadata.
- SHOW CREATE TABLE *[***DATABASE NAME***].[***TABLE NAME***]* retrieves table DDL and location.
- SHOW CREATE VIEW *[***DATABASE NAME***].[***VIEW NAME***]* retrieves view definitions.

Permission Mapping

The following table maps extractor actions to the minimum required Impala privileges and recommended Apache Ranger grants.

Table 1: Extractor actions, statements, and grants

Extractor action	Impala statement	Minimum privilege	Recommended grant
List databases	SHOW DATABASES	Any privilege on visible objects. The SHOW statement only returns objects the user can view.	REFRESH
List tables in a database	SHOW TABLES IN *[***DATABASE NAME***]*	Any privilege on visible objects. Output is filtered by authorization.	REFRESH
Retrieve table metadata	DESCRIBE FORMATTED *[***DATABASE NAME***].[***TABLE NAME***]*	SELECT, INSERT, or REFRESH on the table or view	REFRESH
Retrieve DDL and location	SHOW CREATE TABLE *[***DATABASE NAME***].[***TABLE NAME***]*	SELECT, INSERT, or REFRESH on the table or view	REFRESH
Retrieve view definition	SHOW CREATE VIEW *[***DATABASE NAME***].[***VIEW NAME***]*	SELECT, INSERT, or REFRESH on the view	REFRESH

Least-privilege Best Practices

Use the following guidance when you configure Ranger permissions for the Impala extractor:

Use REFRESH as the primary privilege

Based on the Impala authorization matrix, REFRESH is sufficient for DESCRIBE FORMATTED, SHOW CREATE TABLE, and SHOW CREATE VIEW statements, and it also satisfies the broader visibility requirement for SHOW statements.

Grant at database scope

When the extractor must inventory all tables in a database, grant the privilege at the database scope. This approach is simpler operationally and avoids partial visibility caused by table-by-table omissions.

Avoid granting SELECT

Do not grant SELECT unless REFRESH does not work in your specific cluster policy model or product version. The SELECT privilege exposes data-read capability that the extractor does not require.

Ranger Policy Configuration Examples

The following examples show how to configure Apache Ranger policies for different metadata extraction scenarios.

Use this configuration when the extractor should access only specific databases.

Table 2: Example A: Scan selected databases

Ranger resource	Value	Permission
Database	finance / sales / hr (example)	REFRESH
Table	*	REFRESH
Column	* or not applicable	None required

Use this configuration when the extractor should access all databases in the cluster.

Table 3: Example B: Scan all databases

Ranger resource	Value	Permission
Database	*	REFRESH

Ranger resource	Value	Permission
Table	*	REFRESH
Column	* or not applicable	None required

Important Considerations

Security-filtered SHOW statements

If a database or table does not appear in the extraction results, the extractor principal likely does not have sufficient privileges on that object. The SHOW statements automatically filter output based on the user's authorization level.

Column-level grants

Avoid using column-level grants for the extractor principal. Column-level grants can lead to partial metadata visibility, while the extractor expects full table definitions.

LOCATION extraction

The extractor retrieves the LOCATION property by parsing SHOW CREATE TABLE output. Because it only reads DDL text, no HDFS read permission is required or implied.

System databases

The system databases _impala_builtins, sys, and information_schema are automatically excluded by the extractor and do not require explicit permissions.

Validating the Ranger Policy

After applying the Apache Ranger policy, validate that the extractor principal has proper access by running the following test commands.



Tip: Run these commands using the same principal that the extractor will use to ensure accurate validation results.

Table 4: Commands to validate Ranger access

Test command	Expected result
SHOW DATABASES;	Returns all authorized databases
SHOW TABLES IN `*[***DATABASE NAME***]*`;	Returns all visible tables in the specified database
DESCRIBE FORMATTED `*[***DATABASE NAME***]*`.`*[***TABLE NAME***]*`;	Returns complete table metadata including schema, partitions, and properties
SHOW CREATE TABLE `*[***DATABASE NAME***]*`.`*[***TABLE NAME***]*`;	Returns DDL statement including LOCATION when applicable
SHOW CREATE VIEW `*[***DATABASE NAME***]*`.`*[***VIEW NAME***]*`;	Returns complete view definition

Summary

Configure an Apache Ranger policy that grants the Impala extractor principal the REFRESH privilege on the required database and table scope. This represents the lowest practical permission set for metadata extraction operations.

Grant the SELECT privilege only if cluster-specific behavior or policy requirements show that REFRESH alone is insufficient.

Configuring Hive or Impala connector with HMS as the metadata source

HMS mode connects Octopai Client directly to the Hive Metastore (HMS) database to read databases, tables, columns, and views, using Kerberos or username and password authentication.

Apache Hive and Impala store their database and table definitions in a relational database called the Hive Metastore (HMS). In HMS mode, also called direct metastore access, Octopai Client reads metadata tables such as DBS and TBLS directly from the database that backs the metastore.

The HMS extraction method supports four database providers: PostgreSQL, MariaDB, MySQL, and Oracle. MariaDB and MySQL share the same setup path.

HMS mode compared to Hive Server connections

HMS mode is different from connecting to HiveServer2 or Impala through an Open Database Connectivity (ODBC) data source name (DSN). For ODBC connections, use the following Hive or Impala connector topics instead of this guide:

- Configuring Hive connector with HiveServer as the metadata source
- Configuring Impala connector with HiveServer as the metadata source

Supported HMS databases

In the **New Metadata Source wizard**, select the required Hive Hms Type depending on the backend database that your metastore uses. The following table lists the supported databases:

Table 5: Supported HMS database backends

HMS type	Typical use on Cloudera	Default port
MariaDB or MySQL	Very common for Hive HMS on Cloudera	3306
PostgreSQL	Some clusters use PostgreSQL for HMS	5432
Oracle	Less common; enterprise deployments	1521

Supported authentication methods

Table 6: Supported authentication methods

Method	What you need	Best for
Kerberos	Kerberos principal, keytab file, and cluster krb5 configuration	Production Cloudera clusters with Kerberos enabled (recommended)
Username and password	Database user and password from your administrator	Non-Kerberos environments, or clusters that allow SQL password login to the HMS database

HMS database permissions

HMS mode uses read-only access to the HMS database. Octopai Client does not create, update, or delete HMS data.



Important: The database user, or the Kerberos-mapped user must have read-only access to the HMS tables and be able to run the SELECT query on the HMS metadata tables. Grant SELECT permission on the following HMS tables used by Octopai. No write permissions are required.

- DBS
- TBLS
- SDS
- COLUMNS_V2

The connection test checks access to DBS. Full metadata extraction also reads TBLS, SDS, and COLUMNS_V2.

Ask your administrator to grant read-only access to the HMS tables, using statements similar to the following examples. Replace the database name, schema name, and user name with the values from your environment.

MariaDB or MySQL

```
GRANT SELECT ON hms.DBS TO 'octopai_user'@'%';
GRANT SELECT ON hms.TBLS TO 'octopai_user'@'%';
GRANT SELECT ON hms.SDS TO 'octopai_user'@'%';
GRANT SELECT ON hms.COLUMNS_V2 TO 'octopai_user'@'%';
```

PostgreSQL

```
GRANT SELECT ON TABLE public."DBS" TO octopai_user;
GRANT SELECT ON TABLE public."TBLS" TO octopai_user;
GRANT SELECT ON TABLE public."SDS" TO octopai_user;
GRANT SELECT ON TABLE public."COLUMNS_V2" TO octopai_user;
```

Oracle

```
GRANT SELECT ON HIVE.DBS TO OCTOPAI_USER;
GRANT SELECT ON HIVE.TBLS TO OCTOPAI_USER;
GRANT SELECT ON HIVE.SDS TO OCTOPAI_USER;
GRANT SELECT ON HIVE.COLUMNS_V2 TO OCTOPAI_USER;
```



Note: On Oracle, the HMS tables typically reside in the HIVE schema (uppercase), rather than in a database named hms or a PostgreSQL public schema. Confirm the actual schema name with your administrator before you run the GRANT statements.

Where credentials are stored

Enter per-connection credentials, such as the principal, keytab, or username and password, in the Octopai Client connection. The kerberos.settings.json file on the Octopai Client machine holds machine-level settings only, such as the path to `kinit`, the Windows Subsystem for Linux (WSL) Kerberos configuration, and the Oracle network folder. It does not hold per-connection passwords.

Related Information

Troubleshooting HMS connections

Prerequisites for HMS connections

Before you configure a Hive or Impala HMS connection in Octopai Client, confirm network access to the metastore database. For Kerberos authentication, also confirm MIT Kerberos, the krb5 configuration, a keytab, and — for MariaDB, MySQL, or PostgreSQL — WSL.

Ensure the items in the following two lists are available before you configure an HMS connection. Use only the values that your administrator provides; the names in the examples are illustrative.

Required for all HMS connections

- Network access to the HMS database host, which is often the same host as Hive or Impala, or a dedicated metastore database host. A virtual private network (VPN) might be required.
- The HMS connection details from your administrator: hostname, port, and database name.
- Credentials for the authentication method that you plan to use: either a Kerberos principal and keytab, or a SQL username and password.

Required only for Kerberos authentication

- MIT Kerberos for Windows. You can download the relevant distribution from MIT Kerberos Distribution. The default installation path is C:\Program Files\MIT\Kerberos\. Verify that C:\Program Files\MIT\Kerberos\bin \kinit.exe exists. This path is used by kerberos.settings.json unless explicitly changed.

- The Kerberos configuration file krb5.ini from your administrator, placed at C:\ProgramData\MIT \Kerberos5\krb5.ini. It must include your realm and key distribution center (KDC), for example:

```
[libdefaults]
default_realm = YOUR.REALM
dns_lookup_realm = false
dns_lookup_kdc = false
ticket_lifetime = 24h
renew_lifetime = 7d
forwardable = true

[realms]
YOUR.REALM = {
  kdc = kdc.example.com
  admin_server = kdc.example.com
}

[domain_realm]
.example.com = YOUR.REALM
```

- A keytab file with the extension .keytab for your service principal, stored securely on the Octopai Client machine, for example C:\Octopai\keytabs\my-service.keytab.
- Windows Subsystem for Linux (WSL) 2 with Ubuntu — required only for Kerberos authentication against MariaDB, MySQL, or PostgreSQL. Oracle HMS Kerberos runs natively on Windows and does not use WSL. Octopai Client uses Linux tools inside WSL for Kerberos database access on Windows. To install WSL, run wsl --install -d Ubuntu.

Configuring an HMS connection in Octopai Client

Configure a Hive or Impala HMS connection in Octopai Client by selecting the Hive Metastore metadata source, entering the metastore database details, and choosing Kerberos or password authentication.

About this task

These steps are common to all HMS database backends. For backend-specific Kerberos setup, see the MariaDB or MySQL, PostgreSQL, and Oracle topics.

Before you begin

Complete the items in the HMS connection prerequisites, and obtain the host, port, database name, and credentials from your administrator.

Procedure

1. In Octopai Client, start the **New Metadata Source wizard** and select the Hive or Impala metadata source type.
2. On the **Metadata Source Details** step, under Metadata Source, select Hive Metastore (HMS Recommended).
3. For Authentication method, select Username and Password or Kerberos.
4. Enter a Connection Name.
5. From the Hive Hms Type list, select the backend database: Postgres, MariaDB, MySql, or Oracle.
6. Enter the Host, Port, and Database Name.

   The port must be in the range 0 to 65535.
7. Enter the authentication details.
   - For Username and Password, enter the UserName and Password.
   - For Kerberos, enter the Keytab Path and Kerberos Principal.

Figure 45: HMS metadata source with Username and Password authentication

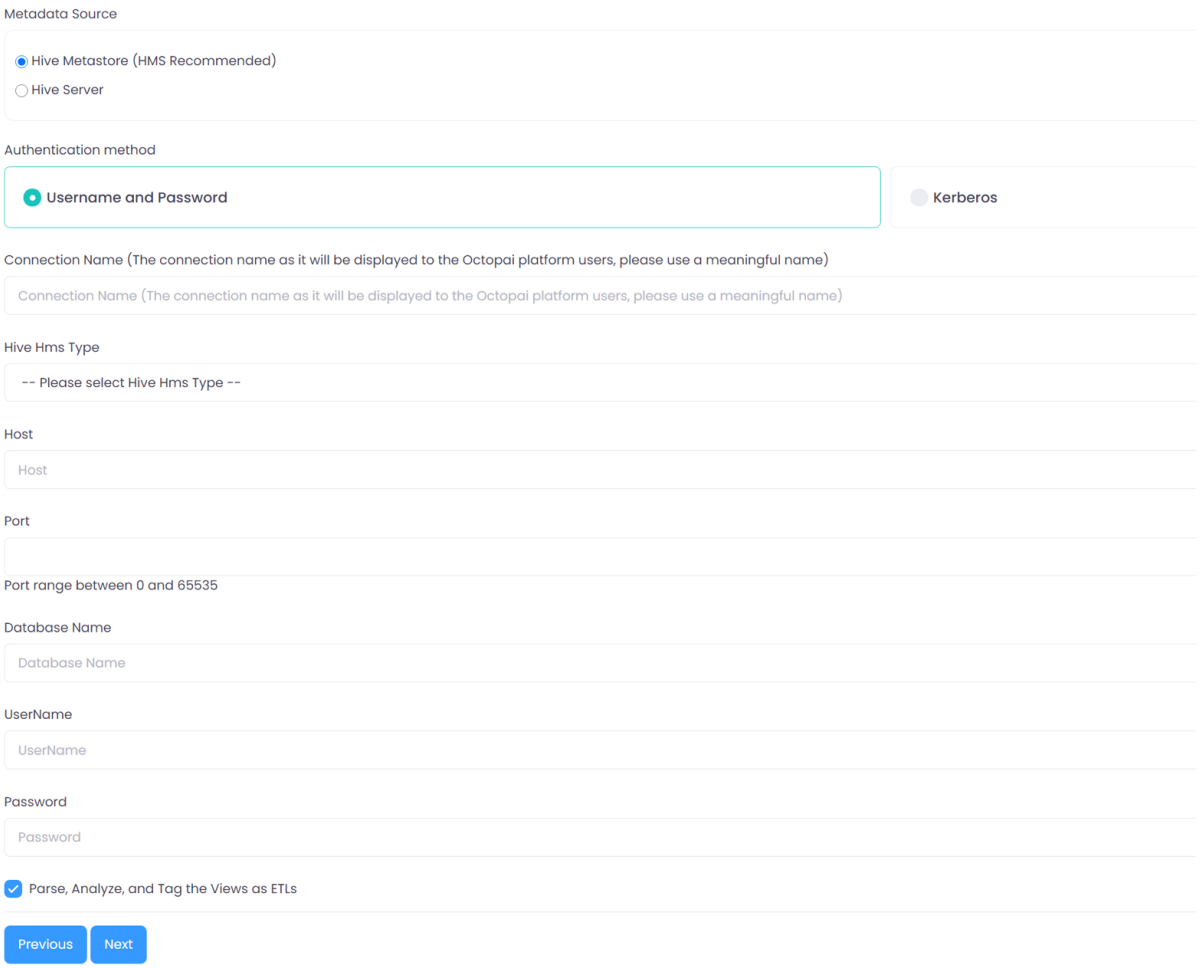
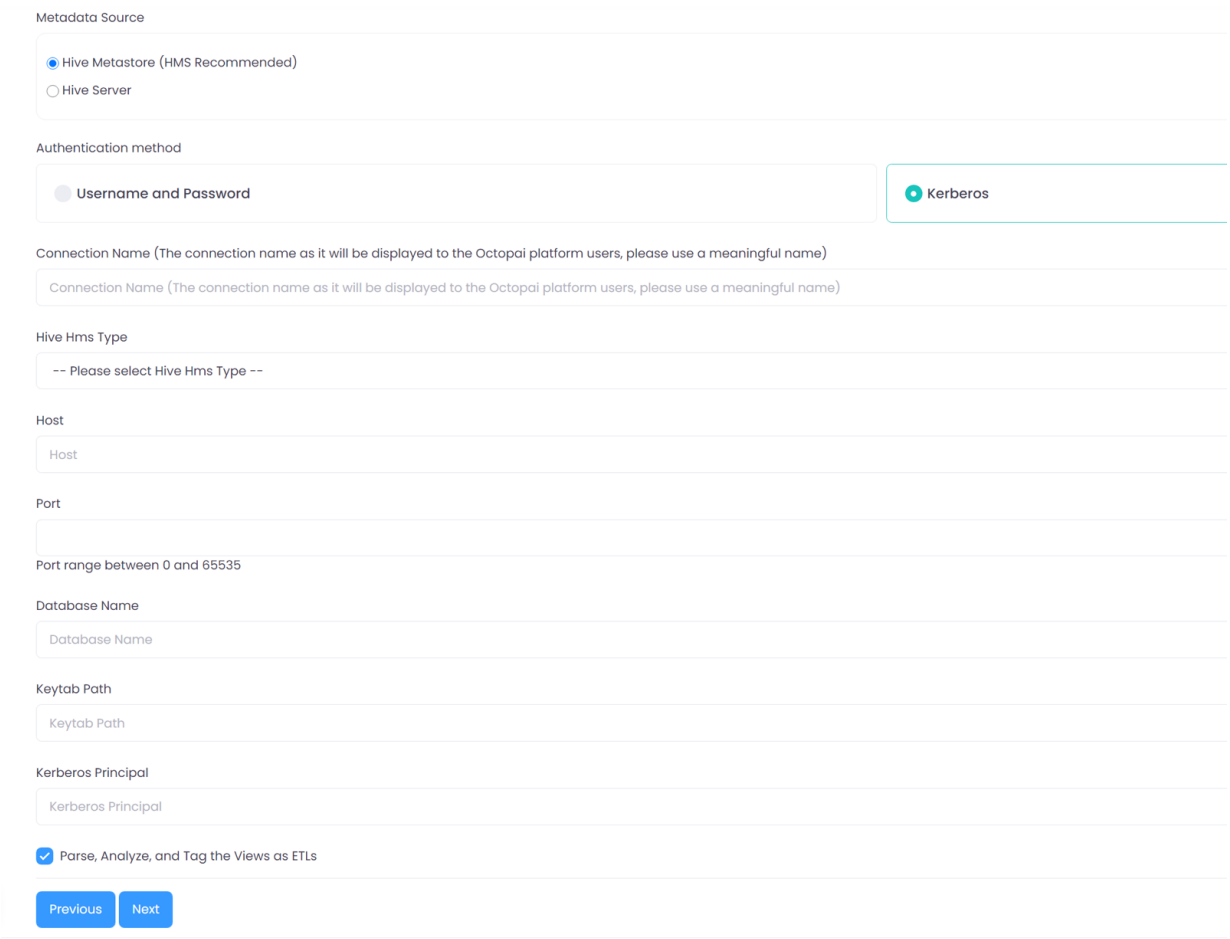


Figure 46: HMS metadata source with Kerberos authentication



8. Select the Parse, Analyze, and Tag the Views as ETLs check box to process views as ETLs.
9. Click Next, test the connection, and save it.

Results

When you test the connection, Octopai Client does the following:

- For Kerberos, acquires a ticket by using `kinit` and your keytab.
- Connects to the HMS database with the settings you entered.
- Runs a simple query against the metastore tables, for example a row count from DBS.
- Shows a success or an error message.

The test confirms that Octopai Client can reach the HMS database and read the metastore tables. It does not run a full metadata extraction. If the test fails, see Troubleshooting HMS connections.

Machine settings for HMS Kerberos authentication

The kerberos.settings.json file on the Octopai Client host defines machine-level Kerberos settings and one enabled profile per HMS database type.

Top-level entries

Edit Data\kerberos.settings.json on the Octopai Client host and enable the one profile that matches your HMS database type. Do not provide your keytab path or principal here for HMS connections; those belong in each connection.

```
{
```

```
  "Kerberos": {
    "KinitPath": "C:\\Program Files\\MIT\\Kerberos\\bin\\kinit.exe",
    "DefaultCacheName": "FILE:C:\\Octopai\\krb5cc",
    "Krb5ConfigPath": ""
  }
}
```

Setting	Description
KinitPath	Path to `kinit.exe` from your MIT Kerberos installation.
DefaultCacheName	Ticket cache used for Hive Server ODBC Kerberos connections. HMS Kerberos on MariaDB or PostgreSQL uses WSL instead. For Oracle, this value must match SQLNET.KERBEROS5_CC_NAME in sqlnet.ora if you set one.
Krb5ConfigPath	Leave empty to use the default C:\ProgramData\MIT \Kerberos5\krb5.ini.

Profile: MariaDB or MySQL (HmsMariaDbMySql)

```
"HmsMariaDbMySql": {
  "Enabled": true,
  "WslKrb5ConfigPath": "/etc/krb5.conf",
  "MysqlClientBinary": "mariadb",
  "MysqlGssapiPluginDir": "/usr/lib/x86_64-linux-gnu/libmariadb3/plugin"
}
```

Setting	Description
Enabled	Set to true when this Octopai Client host uses MariaDB or MySQL HMS with Kerberos.
WslKrb5ConfigPath	Path to the Kerberos configuration inside WSL, usually /etc/krb5.conf.
MysqlClientBinary	Name of the SQL client command in WSL. Use mariadb, the default on Ubuntu. Set it to mysql only if your WSL installation provides that binary instead.
MysqlGssapiPluginDir	Folder in WSL that contains auth_gssapi_client.so, the Generic Security Services Application Program Interface (GSSAPI) Kerberos plugin. Locate this on your machine after you install the packages.



Important: If Octopai Client runs as a Windows service, it must run as a local administrator user account that owns the WSL profile, not as Local System, Network Service, or a separate service account. WSL and Kerberos setup must be completed while logged from that user account. For more information, see Running the Octopai Client Windows Service with WSL.

Profile: PostgreSQL (HmsPostgres)

```
"HmsPostgres": {
  "Enabled": true,
  "WslKrb5ConfigPath": "/etc/krb5.conf"
}
```

To install the client, run sudo apt install -y krb5-user postgresql-client.

PostgreSQL HMS Kerberos also uses WSL. For more information, see Running the Octopai Client Windows Service with WSL.

Profile: Oracle (HmsOracle)

```
"HmsOracle": {
```

```
  "Enabled": true,
  "OracleTnsAdminPath": "C:\\Octopai\\oracle\\network\\admin"
}
```

The HmsOracle profile is one part of the Oracle Kerberos setup. It only tells Octopai Client where the local Oracle network folder is. OracleTnsAdminPath is a Windows folder on the Octopai Client machine that must contain a file named sqlnet.ora, and Octopai Client uses this folder as TNS_ADMIN when it connects with Kerberos. The actual Kerberos behavior still comes from sqlnet.ora, MIT Kerberos on the client host, the keytab, and the connection fields — see Configuring Oracle HMS access.

HmsOracle is not required for Oracle password authentication. For password-only Oracle HMS connections, ensure that HmsOracle.Enabled is set to false or exclude the profile.



Note: OracleTnsAdminPath must be on the Windows host where Octopai Client runs. Your administrator provides the folder (often a zip file containing sqlnet.ora) for that machine.

Running the Octopai Client Windows service with WSL

MariaDB and PostgreSQL HMS Kerberos access uses WSL, which is installed per Windows user, so the Octopai Client Windows service must run as the local administrator user that owns the WSL profile.

MariaDB and PostgreSQL HMS Kerberos run `kinit` on Windows and then run `mariadb` or `psql` inside WSL. WSL is installed per Windows user, so it is not available to built-in service accounts in the same way. Oracle HMS Kerberos does not use WSL, so this requirement applies only to MariaDB and PostgreSQL.

When you install Octopai Client as a Windows service, rather than only opening it as a desktop application, you must register the service to log on as a local administrator user account, for example .\octopai-admin. Do not use any of the following:

- Local System
- Network Service
- A dedicated Windows service account that does not have a WSL profile

Recommended setup order

To set up scheduled extractions that use MariaDB or PostgreSQL HMS Kerberos, complete the setup in the following order:

1. Create or choose the Windows user that runs the Octopai Client service. The user must be a member of the local Administrators group.
2. Log in as that user on the Octopai Client machine.
3. Install WSL, the packages, and /etc/krb5.conf under that user.
4. Install MIT Kerberos, the keytab, and kerberos.settings.json on that machine.
5. Install or reconfigure the Octopai Client Windows service to run as that same user, with the password stored for the service.
6. Restart the service and run Test Connection.

Configuring MariaDB or MySQL HMS access for Kerberos

Configure Kerberos access to a MariaDB or MySQL Hive Metastore database by setting up WSL, the Kerberos client, and the GSSAPI plugin on the Octopai Client host.

Before you begin

Your administrator must provide the following permissions and details:

- The HMS database name.
- For Kerberos, a database user or role mapped to your Kerberos principal.
- For password authentication, a SQL user and password allowed from your client IP address.
- The Kerberos service principal in the form mariadb/<hostname>@<REALM>.

Table 7: MariaDB or MySQL HMS connection fields

Field	Example	Notes
Hive Hms Type	MariaDB or MySql	Match the cluster backend
Host	hms-db.example.com	From your administrator
Port	3306	From your administrator
Database Name	hms	From your administrator
Kerberos Principal	my-service@YOUR.REALM	Kerberos only
Keytab Path	C:\Octopai\keytabs\my-service.keytab	Kerberos only
UserName and Password	From your administrator	Password authentication only

About this task

For Kerberos, Octopai Client runs `kinit` on Windows and then uses the `mariadb` client inside WSL with Kerberos (GSSAPI). You must complete WSL setup and enable HmsMariaDbMySql in kerberos.settings.json.

If Octopai runs as a Windows Service, it must use a local administrator user account—not Local System or a service account without WSL.

Perform the following steps to set up Kerberos authentication inside WSL:

Procedure

1. Install WSL 2 and Ubuntu if they are not already installed.

   In PowerShell running as Administrator, run wsl --install -d Ubuntu, and then reboot if prompted.
2. Open Ubuntu from the Start menu and install the required packages.

   ```
   sudo apt update
   sudo apt install -y krb5-user mariadb-client libmariadb3
   ```

   When prompted for the default Kerberos realm, enter the site realm from your administrator.
3. Create or copy /etc/krb5.conf in WSL so that it matches the Windows C:\ProgramData\MIT\Kerberos5\krb5.ini, using the same realm and KDC hosts.
4. Find the GSSAPI plugin directory.

   ```
   find /usr -name auth_gssapi_client.so 2>/dev/null
   ```

   Use the directory that contains the file, not the file path, as MysqlGssapiPluginDir. On Ubuntu, this is often /usr/lib/x86_64-linux-gnu/libmariadb3/plugin.
5. In kerberos.settings.json, set HmsMariaDbMySql.Enabled to true and add the paths from the previous steps.

Results

To verify Kerberos access from WSL before you test the connection, acquire a ticket and query the metastore:

```
kinit -kt /mnt/c/Octopai/keytabs/my-service.keytab my-service@YOUR.REALM
klist

mariadb --plugin-dir=/usr/lib/x86_64-linux-gnu/libmariadb3/plugin \
  -h hms-db.example.com -P 3306 \
  -u my-service hms \
  -e "SELECT COUNT(*) FROM DBS;"
```

Replace my-service, keytab path, host, and database name with the values that you received from your administrator. The -u user must match the MariaDB user mapped to your Kerberos principal.

What to do next

For password authentication, no WSL or Kerberos profile is required. Enter the username and password in the connection, and ensure that the database server allows password login from your machine.

Configuring PostgreSQL HMS access for Kerberos

Configure Kerberos access to a PostgreSQL Hive Metastore database by setting up WSL and the PostgreSQL client on the Octopai Client host.

Before you begin

Your administrator must provide the following permissions and details:

- The HMS database name.
- For Kerberos, a PostgreSQL role mapped to your Kerberos principal in pg_ident.conf.
- For password authentication, a role and password, with your client IP address allowed in pg_hba.conf.
- The Kerberos service principal in the form postgres/<hostname>@<REALM>.

Table 8: PostgreSQL HMS connection fields

Field	Example	Notes
Hive Hms Type	Postgres	N/A
Host	hms-db.example.com	From your administrator
Port	5432	From your administrator
Database Name	hms	From your administrator
Kerberos Principal	my-service@YOUR.REALM	Kerberos only
Keytab Path	C:\Octopai\keytabs\my-service.keytab	Kerberos only
UserName and Password	From your administrator	Password authentication only; not used for Kerberos

For Kerberos connections, you enter only the principal and keytab. Octopai Client derives the PostgreSQL login role from the principal: the part before @, with hyphens changed to underscores. For example, my-service@REALM becomes the role my_service. Your administrator must create or map that role to match your principal.

About this task

For Kerberos, Octopai Client runs `kinit` on Windows and then uses the `psql` client inside WSL. If Octopai runs as a Windows Service, it must use a local administrator user account—not Local System or a service account without WSL.

Perform the following steps to set up Kerberos authentication inside WSL:

Procedure

1. Install WSL 2 and Ubuntu if they are not already installed.

   In PowerShell running as Administrator, run wsl --install -d Ubuntu, and then reboot if prompted.
2. Open Ubuntu from the Start menu and install the required packages.

   ```
   sudo apt install -y krb5-user postgresql-client
   ```
3. Create or copy /etc/krb5.conf in WSL so that it matches the Windows C:\ProgramData\MIT\Kerberos5\krb5.ini, using the same realm and KDC hosts.
4. In kerberos.settings.json, set HmsPostgres.Enabled to true.

Results

To verify Kerberos access from WSL before you test the connection, acquire a ticket and query the metastore:

```
kinit -kt /mnt/c/Octopai/keytabs/my-service.keytab my-service@YOUR.REALM

psql "host=hms-db.example.com port=5432 dbname=hms user=my_service" \
  -c 'SELECT COUNT(*) FROM public."DBS";'
```

What to do next

For password authentication, use a SQL username and password. Password authentication must be enabled in pg_hba.conf for your client IP address. If login fails with a correct password, ask your administrator to verify that the pg_hba.conf authentication settings match how the password was created.

Configuring Oracle HMS access

Oracle HMS supports both password and Kerberos authentication. Both modes use the Oracle .NET driver that Octopai Client ships, and neither requires Oracle Database software on the client. Kerberos additionally requires MIT Kerberos and a local folder containing sqlnet.ora, configured through OracleTnsAdminPath. Oracle HMS Kerberos does not use WSL.

Before you begin

Your administrator must provide the following permissions and details:

- The Oracle service name for the pluggable database (PDB) — required for both authentication modes.
- The metastore schema and tables — required for both authentication modes.
- For password authentication, an Oracle user and password.
- For Kerberos authentication, an Oracle user mapped to your principal, the service principal oracle/<hostname >@<REALM>, and a sqlnet.ora folder for the Octopai Client host.

You do not need Oracle Database software on the Octopai Client machine for either authentication mode; Octopai Client ships the Oracle .NET driver. The two authentication modes differ as follows:

- Password authentication needs only the shipped Oracle .NET driver and the database credentials.
- Kerberos authentication additionally needs MIT Kerberos on the Octopai Client host and a local folder containing sqlnet.ora, configured through OracleTnsAdminPath in kerberos.settings.json. It runs natively on Windows — no WSL.

The following table lists the connection fields:

Table 9: Oracle HMS connection fields

Field	Example	Notes
Hive Hms Type	Oracle	N/A
Host	oracle-hms.example.com	HMS database host, from your administrator
Port	1521	From your administrator
Database Name	ORCLPDB1	Oracle service name, from your administrator
Kerberos Principal	my-service@YOUR.REALM	Kerberos only
Keytab Path	C:\Octopai\keytabs\my-service.keytab	Kerberos only
UserName and Password	From your administrator	Password authentication only

About this task

The following steps apply only when you use Kerberos authentication. For password authentication, skip the steps and see the postrequisite at the end of this topic.

Oracle HMS Kerberos runs on Windows and does not use WSL. Octopai Client uses `kinit` and the Oracle driver with Kerberos.

The HmsOracle profile in kerberos.settings.json is one part of the Oracle Kerberos setup, not a separate mechanism. It only tells Octopai Client where the local Oracle network folder is; the actual Oracle Kerberos behavior still comes from sqlnet.ora inside that folder, MIT Kerberos on the client host, the keytab, and the connection fields.

Procedure

1. Obtain the TNS admin folder from your administrator and place it on the Octopai Client host, for example C:\Octopai\oracle\network\admin.

   Your administrator might provide this folder as a compressed file to extract.
2. Ensure the folder contains a sqlnet.ora file configured for your Kerberos realm and ticket cache.

   ```
   SQLNET.AUTHENTICATION_SERVICES = (KERBEROS5)
   SQLNET.AUTHENTICATION_KERBEROS5_SERVICE = oracle
   SQLNET.KERBEROS5_CONF = C:\ProgramData\MIT\Kerberos5\krb5.ini
   SQLNET.KERBEROS5_CONF_MIT = TRUE
   SQLNET.KERBEROS5_CC_NAME = FILE:C:/Octopai/krb5cc
   ```

   SQLNET.KERBEROS5_CC_NAME must match where MIT Kerberos stores tickets on Windows.
3. In kerberos.settings.json, set HmsOracle.Enabled to true and set OracleTnsAdminPath to the folder that contains sqlnet.ora.

What to do next

For password authentication, enter the Oracle username and password in the connection. No HmsOracle profile in kerberos.settings.json is required; leave HmsOracle.Enabled at false (or omit the profile) unless the same Octopai Client host also uses Oracle HMS Kerberos.

Streaming and processing

Cloudera Data Lineage captures lineage from Apache Kafka Connect, Apache NiFi flows, and Apache Spark jobs through a Spline-based connector.

Configuring the Kafka Connect connector in Cloudera Data Lineage

Learn how to configure the Kafka Connect connector in Octopai Client using Kerberos authentication (SASL/GSSAPI).

Before you begin

Before configuring the Kafka Connect connector in Octopai Client, ensure the following components are available and properly configured:

- **Kerberos infrastructure:** Active Kerberos Key Distribution Center (KDC), valid Kerberos realm configuration, and network connectivity from the Octopai Client host to the KDC.
- **MIT Kerberos for Windows:** Install MIT Kerberos for Windows on the machine running Octopai Client. The default installation path is C:\Program Files\MIT\Kerberos\bin\kinit.exe. Octopai Client uses kinit to acquire Kerberos tickets.

- **Kerberos configuration file:** The Kerberos configuration file must exist at C:\ProgramData\MIT \Kerberos5\krb5.ini.

  Example configuration:

```
[libdefaults]
                              default_realm = ROOT.COMOPS.SITE
                              dns_lookup_realm = false
                              dns_lookup_kdc = false
                              ticket_lifetime = 24h
                              renew_lifetime = 7d
                              forwardable = true

                              [realms]
                              ROOT.COMOPS.SITE = {
                              kdc = ccycloud-1.cdp.root.comops.site
                              admin_server = ccycloud-1.cdp.root.comops.site
                              }

                              [domain_realm]
                              .root.comops.site = ROOT.COMOPS.SITE
                              root.comops.site = ROOT.COMOPS.SITE
```

- **Kerberos credentials:** Obtain a Kerberos principal (for example, kafka-user@REALM) and its associated keytab file (for example, C:\octopai\kafka-user.keytab). Ensure the keytab file is securely stored and accessible.
- **Kafka cluster configuration:** Ensure the Kafka cluster is configured with SASL/GSSAPI enabled, the Kafka service principal is configured on the brokers (for example, kafka/hostname@REALM), and the broker hostnames are resolvable using fully qualified domain names.

Procedure

1. Create a keytab file.

   On a Kerberos administration server, create a keytab file for the Kerberos principal used by Octopai Client using this command:

```
ktutil
                           addent -password -p kafka-user@REALM -k 1 -e aes2
56-cts
                           wkt /path/to/kafka-user.keytab
                           quit
```

   Or request the keytab file from your Kerberos administrator using this command:

```
kadmin -q "ktadd -k /path/to/kafka-user.keytab kafka-user@REALM"
```

   Copy the keytab file securely to the Windows server running the Octopai Client.
2. Verify Kerberos authentication before configuring the connector.
   a) Open PowerShell.
   b) Navigate to the Kerberos binaries directory:

```
cd "C:\Program Files\MIT\Kerberos\bin"
```

   c) Obtain a Kerberos ticket:

```
.\kinit.exe -kt "C:\octopai\kafka-user.keytab" kafka-user@REALM
```

   d) Confirm the validity of the ticket:

```
.\klist.exe
```

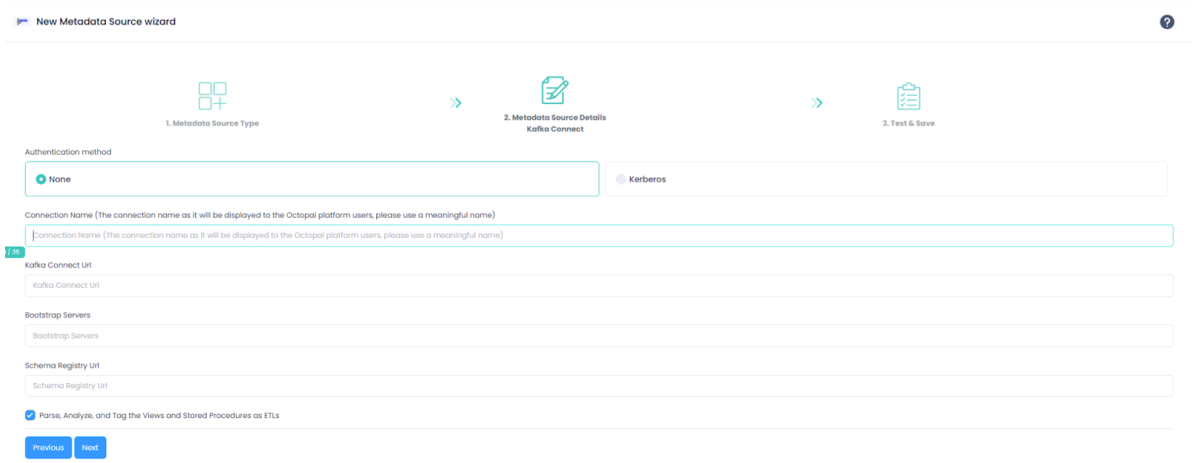
Note: A valid ticket confirms that Kerberos is configured correctly.

3. Configure the Kafka Connect connector in Octopai Client.
   a) Start a new connection:
      1. Open Octopai Client.
      2. In the **New Metadata Source** wizard, on Metadata Source Type, select Kafka Connect.
   b) Select the authentication method: None or Kerberos.

Note: The fields available on the form depend on the authentication method you select.

   c) Complete the connection fields:

Figure 47: Kafka Connect metadata source details with no authentication method selected



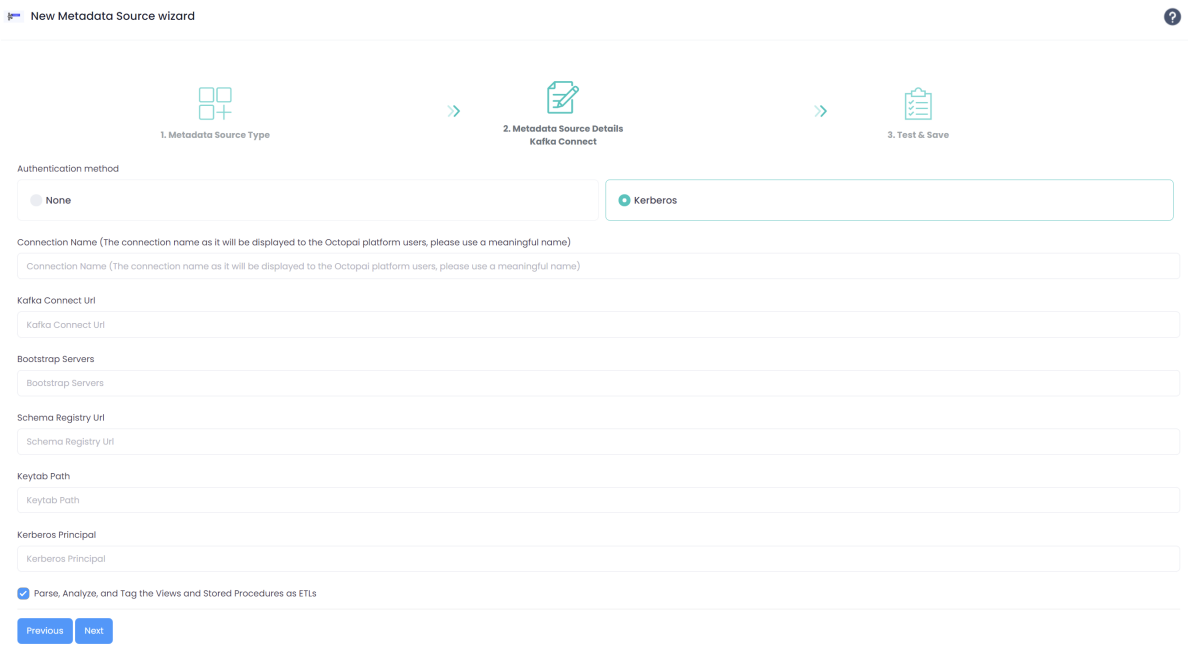
- Connection Name (Required): Enter a meaningful name. The name is displayed to Cloudera Data Lineage (formerly Octopai) platform users.
- Kafka Connect URL (Required): Kafka Connect REST endpoint.
- Bootstrap Servers (Required): Kafka broker hostnames and ports (for example, kafka1.example.com:9092).
- Schema Registry URL (Optional): Schema Registry endpoint (for example, http://schema-registry.example.com:8081).
- Parse, Analyze, and Tag the Views and Stored Procedures as ETLs (Optional): Select this option when you want Cloudera Data Lineage to treat views and stored procedures as ETL objects during analysis. This option is enabled by default.

Note:

The connector retrieves topic metadata from Kafka using Bootstrap Servers, and all other metadata from Kafka Connect using the Kafka Connect URL.

   d) If you selected Kerberos, enter the following additional Kerberos settings.

Figure 48: Kafka Connect metadata source details with Kerberos authentication selected



- Keytab Path (Required): Full path to the keytab file (for example, C:\octopai\kafka-user.keytab).
- Kerberos Principal (Required): for example, kafka-user@REALM.

Important:

- Realm names are case-sensitive.
- The keytab file must be readable by the Octopai Client service account.
- Broker addresses must use fully qualified domain names.

e) Click Next to advance to the Test & Save step.
f) To test the connection, click Test Connection.
g) Click Save to store the connection.

4. Verify extraction results.

   After a successful run, confirm the metadata archive in the target folder. See Verify extracted metadata files.

Kerberos authentication errors:

- Verify the principal format: username@REALM
- Confirm the keytab path and permissions.
- Validate the realm configuration in krb5.ini

Kafka connectivity issues:

- Verify the network connectivity.
- Use fully qualified domain names.
- Confirm the SASL listener configuration.

Ticket expiration:

- Verify the ticket_lifetime and renew_lifetime values.
- Adjust the renewal configuration, if required.

Clock skew:

Synchronize the system time:

```
w32tm /resync /force
```

Error during the extraction:

See Troubleshoot connector extraction failures.

Cloudera Data Lineage Connector for Apache NiFi

Learn how the Cloudera Data Lineage (formerly Octopai) connector for Apache NiFi enables visibility into data movement across systems by capturing and visualizing lineage derived from NiFi flows.

Overview

Apache NiFi is a core orchestration platform in modern data architectures, responsible for ingesting, routing, transforming, and delivering data across heterogeneous environments. The Cloudera Data Lineage connector for Apache NiFi extracts metadata from NiFi flows and constructs lineage that exposes how data moves between systems, technologies, and platforms.

The connector enables the following capabilities:

- Building cross-system, inner system and end-to-end column lineage for NiFi flows.
- Populating the Knowledge Hub assets automatically and enabling users to discover NiFi assets.
- Enabling governance, impact analysis, and operational visibility across enterprise data pipelines.

Why Apache NiFi data lineage matters

NiFi often serves as the integration layer that connects files, databases, object stores, streaming platforms, and cloud services. Understanding these flows is critical for several reasons.

Visibility into data movement

NiFi connects diverse sources and targets. Lineage reveals how data enters, moves through, and exits the platform.

Cross-system complexity

NiFi commonly bridges legacy, hybrid, and cloud environments. Cross-system lineage enables teams to track data as it moves across technologies and organizational boundaries.

Operational insight

Understanding dependencies between systems helps teams troubleshoot failures, assess the impact of change, and reduce risk during migrations or platform modernization initiatives.



Supported NiFi versions

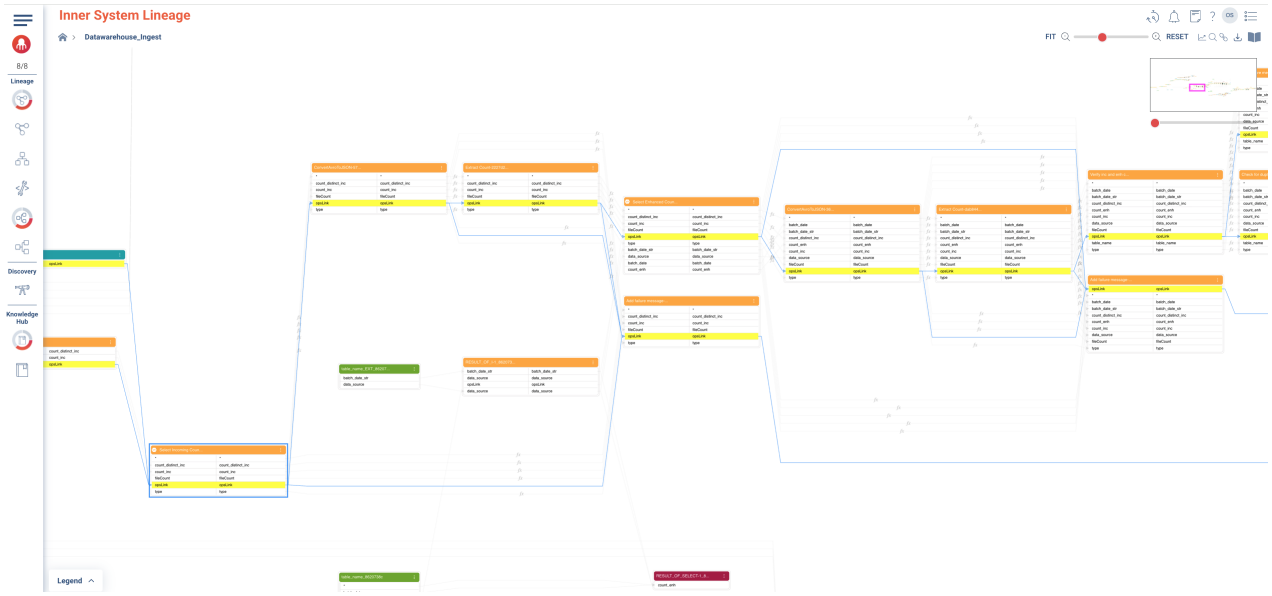
The connector is compatible with Apache NiFi versions 1.2.8 through 2.7.2.

Using supported versions ensures consistent metadata extraction, stable API behavior, and reliable lineage generation.

Lineage model overview

The connector builds lineage in layers, starting with operational flow relationships and extending to system-level source and target context.

Processor-level operational lineage (opsLink)



The foundation of NiFi lineage is the processor-level operational link, referred to as an *opsLink*.

An opsLink represents a direct execution relationship between two NiFi components:

- A source processor and a target processor connected by a NiFi connection.

- InputPorts and OutputPorts are treated as processors for lineage purposes.

opsLinks are derived by parsing the ProcessorGroup configuration JSON and capturing:

- Processors, InputPorts, and OutputPorts.
- Connections between components.
- Associated source or target context when available, such as database, table, topic, bucket, or file location.

This processor-level lineage forms the operational flow graph of a NiFi ProcessGroup and serves as the backbone for cross-system lineage views.

Nested ProcessGroups behavior

When a ProcessorGroup contains other nested ProcessorGroups, inner lineage is scoped strictly to the selected ProcessorGroup:

- Only processors, InputPorts, and OutputPorts that belong directly to the current ProcessorGroup are displayed in the inner lineage view.
- Nested ProcessorGroups are not expanded or traversed as part of inner lineage.

When one ProcessorGroup is connected to another ProcessorGroup:

- The relationship between ProcessorGroups is not shown in inner lineage.
- These relationships are visualized only in end-to-end lineage or cross-system lineage views.

This separation ensures that inner lineage remains focused on execution flow within a single ProcessGroup, while cross-group dependencies are handled at higher-level lineage views.

Supported NiFi components

The connector supports a broad set of NiFi processors for lineage extraction.

For each supported processor, the documentation lists:

- Component Type (FQCN)
- NiFiProcessorType

Table 10: Supported components

Component Type (FQCN)	NiFiProcessorType
org.apache.nifi.processors.standard.QueryDatabaseTable	QueryDatabaseTable
org.apache.nifi.processors.kafka.pubsub.ConsumeKafka_1_0	ConsumeKafka
org.apache.nifi.kafka.processors.ConsumeKafka	ConsumeKafka
org.apache.nifi.processors.kafka.pubsub.ConsumeKafkaRecord_2_6	ConsumeKafkaRecord
org.apache.nifi.processors.kafka.pubsub.PublishKafka_1_0	PublishKafka
org.apache.nifi.processors.kafka.pubsub.PublishKafka_2_0	PublishKafka
org.apache.nifi.processors.kafka.pubsub.PublishKafka_2_6	PublishKafka
org.apache.nifi.processors.kafka.pubsub.PublishKafkaRecord_2_0	PublishKafkaRecord
org.apache.nifi.processors.kafka.pubsub.PublishKafkaRecord_2_6	PublishKafkaRecord
org.apache.nifi.processors.kudu.PutKudu	PutKudu
org.apache.nifi.processors.standard.FlattenJson	FlattenJson
org.apache.nifi.processors.attributes.UpdateAttribute	UpdateAttribute
org.apache.nifi.processors.aws.s3.PutS3Object	PutS3Object
org.apache.nifi.processors.standard.PutSQL	PutSQL
org.apache.nifi.processors.standard.RouteOnAttribute	RouteOnAttribute

Component Type (FQCN)	NiFiProcessorType
org.apache.nifi.processors.standard.RouteOnContent	RouteOnContent
org.apache.nifi.processors.aws.s3.ListS3	ListS3
org.apache.nifi.processors.standard.ExecuteStreamCommand	ExecuteStreamCommand
org.apache.nifi.processors.standard.ReplaceText	ReplaceText
org.apache.nifi.processors.hadoop.DeleteHDFS	DeleteHDFS
org.apache.nifi.processors.standard.GenerateFlowFile	GenerateFlowFile
org.apache.nifi.processors.aws.s3.FetchS3Object	FetchS3Object
org.apache.nifi.processors.parquet.PutParquet	PutParquet
org.apache.nifi.processors.standard.InvokeHTTP	InvokeHTTP
org.apache.nifi.processors.standard.GenerateTableFetch	GenerateTableFetch
org.apache.nifi.processors.standard.ConvertRecord	ConvertRecord
org.apache.nifi.processors.hadoop.FetchHDFS	FetchHDFS
org.apache.nifi.processors.standard.EvaluateJsonPath	EvaluateJsonPath
org.apache.nifi.csv.CSVReader	CSVReader
org.apache.nifi.processors.standard.AttributesToJSON	AttributesToJSON
org.apache.nifi.processors.standard.ExecuteSQL	ExecuteSQL
org.apache.nifi.processors.standard.ExecuteScript	ExecuteScript
org.apache.nifi.processors.script.ExecuteScript	ExecuteScript
org.apache.nifi.processors.standard.LogMessage	LogMessage
org.apache.nifi.processors.standard.ExecuteSQLRecord	ExecuteSQLRecord
org.apache.nifi.processors.hive.PutHive3QL	PutHive3QL
org.apache.nifi.processors.office.ConvertExcelToCSVProcessor	ConvertExcelToCSVProcessor
org.apache.nifi.processors.cdp.objectstore.PutCDPObjectStore	PutCDPObjectStore
org.apache.nifi.processors.cdp.objectstore.DeleteCDPObjectStore	DeleteCDPObjectStore
org.apache.nifi.csv.CSVRecordSetWriter	CSVRecordSetWriter
org.apache.nifi.processors.standard.MergeContent	MergeContent
org.apache.nifi.processors.standard.QueryRecord	QueryRecord
org.apache.nifi.processors.standard.ExtractText	ExtractText
org.apache.nifi.dbcp.DBCPConnectionPool	DBCPConnectionPool
org.apache.nifi.processors.standard.DistributeLoad	DistributeLoad
org.apache.nifi.processors.standard.SplitJson	SplitJson
org.apache.nifi.processors.standard.JoltTransformJSON	JoltTransformJSON
org.apache.nifi.processors.standard.JoltTransformRecord	JoltTransformRecord
org.apache.nifi.processors.mongodb.GetMongo	GetMongo
org.apache.nifi.processors.mongodb.PutMongo	PutMongo
org.apache.nifi.processors.hive.SelectHive3QL	SelectHive3QL
org.apache.nifi.processors.standard.SplitText	SplitText
org.apache.nifi.processors.standard.FetchSFTP	FetchSFTP
org.apache.nifi.processors.avro.ConvertAvroToJSON	ConvertAvroToJSON

Component Type (FQCN)	NiFiProcessorType
org.apache.nifi.processors.standard.UpdateRecord	UpdateRecord
org.apache.nifi.processors.parquet.ConvertAvroToParquet	ConvertAvroToParquet
org.apache.nifi.processors.enrich.JoinEnrichment	JoinEnrichment
org.apache.nifi.processors.enrich.ForkEnrichment	ForkEnrichment
org.apache.nifi.processors.standard.MergeRecord	MergeRecord
org.apache.nifi.processors.standard.InferAvroSchema	InferAvroSchema
org.apache.nifi.processors.kite.InferAvroSchema	InferAvroSchema
org.apache.nifi.processors.aws.s3.DeleteS3Object	DeleteS3Object
org.apache.nifi.processors.standard.SplitRecord	SplitRecord
org.apache.nifi.processors.standard.ConvertJSONToAvro	ConvertJSONToAvro
org.apache.nifi.processors.standard.PutSFTP	PutSFTP
org.apache.nifi.processors.hadoop.PutHDFS	PutHDFS
org.apache.nifi.processors.standard.PutDatabaseRecord	PutDatabaseRecord
org.apache.nifi.processors.standard.ConvertJSONToSQL	ConvertJSONToSQL
org.apache.nifi.processors.iceberg.PutIceberg	PutIceberg
org.apache.nifi.processors.standard.PutFile	PutFile
com.demoulas.nifi.processors.GetSFTPFileInfo	GetSFTPFileInfo
com.demoulas.nifi.processors.MoveSFTP	MoveSFTP
org.apache.nifi.processors.standard.CompressContent	CompressContent
org.apache.nifi.processors.standard.GetSFTP	GetSFTP
org.apache.nifi.processors.standard.Notify	Notify
org.apache.nifi.processors.standard.RetryFlowFile	RetryFlowFile
org.apache.nifi.processors.standard.Wait	Wait

Note: Processors that are not listed in the table are handled using basic pass-through logic.

Knox configuration

The connector supports environments where NiFi is deployed behind an Apache Knox Gateway.

When you configure connectivity through Knox, use the correct topology and endpoint.

Knox topology behavior

With Knox service auto-discovery enabled, two NiFi-related endpoints may appear:

Table 11: NiFi-related Knox topologies and endpoints

Topology	Endpoint	Description
cdp-proxy	/nifi-app/nifi	NiFi UI (browser access)
cdp-proxy-api	/nifi	NiFi API (programmatic access)

Required configuration for Cloudera Data Lineage

Cloudera Data Lineage metadata extraction requires the following Knox path:

- Topology: cdp-proxy

- Endpoint: /nifi

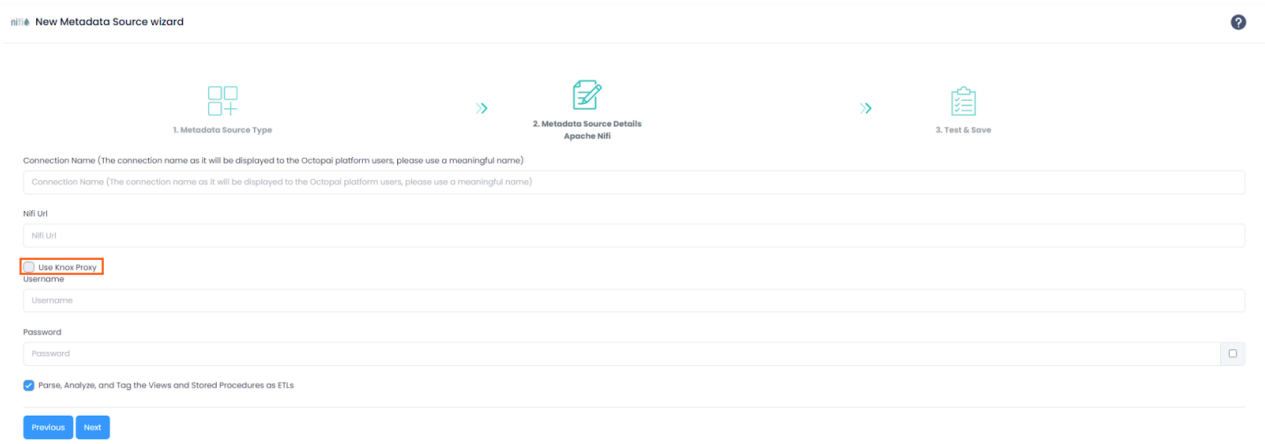
Important:

The /nifi-app/nifi path is for UI access in the browser. For metadata extraction, use the /nifi endpoint instead of /nifi-app/nifi.

Using the wrong endpoint can result in connection failures, incomplete metadata extraction, or authentication or routing issues.

Use Knox Proxy disabled (default)

Figure 49: Use Knox Proxy option in the New Metadata Source wizard



If Use Knox Proxy is unchecked in the New Metadata Source wizard:

- Authentication uses NiFi native token-based authentication.
- The extractor sends credentials to the NiFi API endpoint: POST /nifi-api/access/token
- NiFi returns a JWT bearer token.
- Subsequent API requests use the token in the Authorization header.
- A direct connection to NiFi is used; standard NiFi API endpoint configuration applies.

Use Knox Proxy enabled

If Use Knox Proxy is checked in the New Metadata Source wizard:

- Authentication uses HTTP Basic Authentication through Knox.
- Credentials are sent with each request in the Authorization header.
- Knox validates credentials and forwards authenticated requests to NiFi.
- No token exchange is performed.

When to use Knox Proxy

Use Knox Proxy when:

- NiFi is accessed through an Apache Knox Gateway.
- Authentication is centrally managed by Knox.
- The NiFi API is exposed through a Knox proxy URL.

Do not use Knox Proxy when:

- Connecting directly to NiFi without a gateway.
- Using NiFi native authentication.
- The NiFi URL points directly to the NiFi server.

Limitations

The following limitations apply:

- Dynamic parameters embedded inside table names or query identifiers, such as ${db.table.fullname}, are not resolved.
- Site-to-site connections are not currently supported.

Installation and setup

Installation and enablement are performed as part of Cloudera environment configuration.

For assistance with configuration or enablement, contact your Cloudera representative or Cloudera Data Lineage Support.

Roadmap direction

The connector will continue to evolve with enhancements including:

- Expanded processor coverage.

Cloudera Data Lineage Connector for Apache Spark — Installation and Setup Guide

Learn how to install and configure the Cloudera Data Lineage (formerly Octopai) Connector for Apache Spark, based on Spline technology, to enable automated metadata extraction and lineage tracking.

About this task

The Cloudera Data Lineage Connector for Apache Spark supports both non-secured Spark clusters and Spark clusters secured with Kerberos authentication.

In Kerberos-secured environments, lineage capture relies on WebHDFS delegation tokens acquired by the Octopai Client running on Windows. Kerberos authentication and delegation token acquisition occur within a Linux environment running on Windows Subsystem for Linux (WSL). This Linux layer is required because the Hadoop and WebHDFS security tools necessary for delegation tokens are available only in Linux.

The Octopai Client, running on Windows, orchestrates the process but relies on Linux (through WSL) to authenticate with the Kerberos KDC and securely access HDFS.

Before you begin

Before starting the installation, ensure the following:

- Spark must be included in your Cloudera Data Lineage license.
- A running Spark Cluster (Spark 2.x or 3.x).
- Relevant Spline Agent JAR file based on your Spark and Scala versions. Contact Cloudera Support or your dedicated Cloudera Account Team for the appropriate package.
- Access to HDFS for storing lineage files generated by the Spline agent.
- Cloudera Manager or similar access to configure Spark cluster properties.
- Access permissions to upload JAR files to HDFS and edit Spark configuration files, such as spark-defaults.conf.
- Cloudera Manager or similar administrative access to configure Spark cluster properties.

Additional prerequisites for Kerberos-secured environments only

Kerberos-secured Spark clusters require additional client-side setup to enable secure HDFS access.

Windows Requirements (Octopai Client)

- A Windows host for running the Octopai Client service. This host acts as the control plane for lineage ingestion.

- MIT Kerberos for Windows installed, version 4.1 or later. This provides Windows-side Kerberos tooling and ticket validation.
- A Kerberos configuration file (krb5.ini) provided by the customer. This file defines realms, KDCs, and domain mappings.
- A Spark service principal keytab stored securely on the Windows host. This keytab is used to authenticate non-interactively against the Kerberos KDC.
- Network access from the Windows host to the following:
  - Port 88 (TCP and UDP) for Kerberos KDC access.
  - Port 749 (TCP) for Kerberos Admin Server access.
  - The WebHDFS endpoint on the configured port (for example, port 20101).

Linux Requirements (WSL Ubuntu)

- A Linux environment is mandatory for Kerberos-secured Spark clusters.
- WSL enabled, with Ubuntu installed. This Linux environment runs alongside Windows and is used exclusively for authentication and token handling.
- Kerberos utilities installed inside Ubuntu, including:
  - krb5-user
  - curl
  - jq
- A Linux Kerberos configuration file located at /etc/krb5.conf. This file is copied from the krb5.ini file on Windows to ensure consistent realm configuration.
- Ability to run Kerberos commands such as kinit and klist inside the Linux environment. These commands are used to authenticate, validate tickets, and troubleshoot authentication issues.
- Linux is required because Hadoop WebHDFS delegation tokens cannot be generated using Windows-native tooling. The Linux Kerberos and Hadoop security stack is required to securely acquire and manage these tokens.

About this task

What is Supported

1. **Lineage based on the Spline agent**: Lineage capture is based on what Spline is capable of parsing from Spark SQL execution plans.
2. **Running jobs only**: Only currently running or newly executed jobs are captured for lineage.
3. **Persistent actions only**: Actions that involve reading from or writing to persistent storage (such as tables or files) are captured.
4. **Cluster configuration**: Spark cluster must be configured to include Spline-specific properties in spark-defaults.conf.
5. **Explicit Application Name**: For job names to appear, jobs must define the application name explicitly:

```
spark = SparkSession.builder \
                        .appName("Spark UDF Example") \
                        .getOrCreate()
```

6. **Deployment inside Customer Environment**: Spark clusters must be deployed and managed by the customer.
7. **Authentication modes**: Non-secured Spark clusters, using standard HDFS authentication, are supported without additional setup. Kerberos-secured Spark clusters are supported using WebHDFS delegation tokens acquired by the Octopai Client through Linux running on WSL.

About this task

Limitations

1. **Lineage for Successfully Completed Jobs Only**: Lineage is captured only for Spark jobs that complete successfully.

2. **Persistent Storage Only**: Only persistent read/write actions to tables or file systems are captured. In-memory DataFrame operations that are not written to storage are not captured.
3. **Explicit Job Naming Required**: If no job name is explicitly set in the Spark code, the lineage record will not include a meaningful job name.
4. **Kerberos Authentication Scope**:
   - Kerberos authentication is supported through delegation tokens.
   - Direct Kerberos authentication inside Spark executors is not supported.
   - Keytabs and Kerberos credentials are not deployed into Spark containers.
5. **Limited Parsing Scope**: Only operations that Spline supports and can parse from the Spark SQL execution plan are included in lineage.
6. **Streaming Jobs Not Supported**: Spark Structured Streaming (e.g., Kafka read/write) is not captured.
7. **Partial Code Execution**: Only the parts of the code that are executed (e.g., code within true condition branches) will produce lineage.
8. **UDFs**: User-Defined Functions (UDFs) are not parsed. Their usage appears in the plan, but the internal logic is not captured.

About this task

Important Usage Note



Important: Lineage is captured only for data operations that are persisted to storage. DataFrames that are not saved to persistent storage will not have lineage captured. As a result, there may be cases where the lineage appears incomplete or broken if intermediate DataFrames are used in-memory without being written to a persistent target.

In Kerberos-secured environments, successful lineage generation depends on the Octopai Client successfully acquiring a valid WebHDFS delegation token. Failure to acquire a token or token expiration will prevent the Spark job from writing lineage files.

Procedure

1. Obtain the Spline Agent JAR file from your Cloudera account team. For example, for Spark 3.5, the Spline Agent JAR is spark-3.5_spark-2.12.jar.
2. Upload the JAR file to HDFS

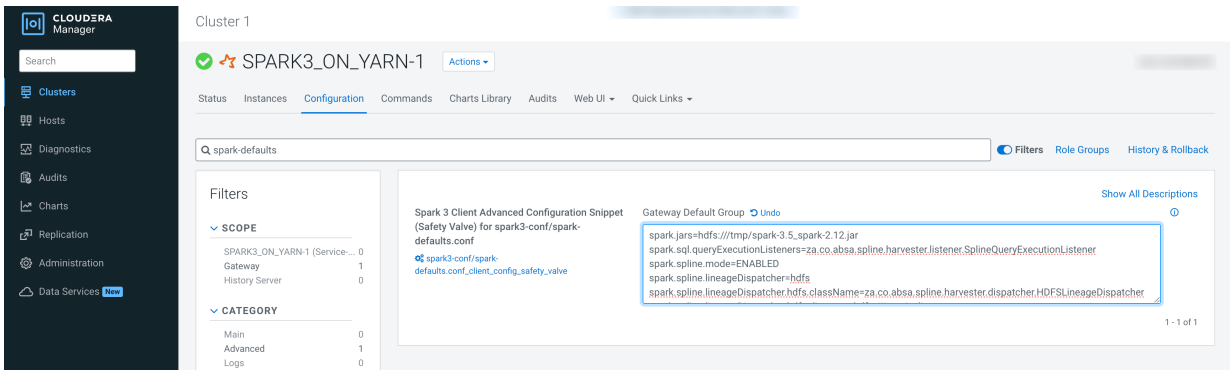
   Upload the Spline Agent JAR file to your HDFS /tmp folder:

   ```
   hdfs dfs -put spark-3.5_spark-2.12.jar /tmp/
   ```

3. Configure Spark Defaults

   Add the following properties to your Spark cluster configuration (spark-defaults.conf) through Cloudera Manager or equivalent:

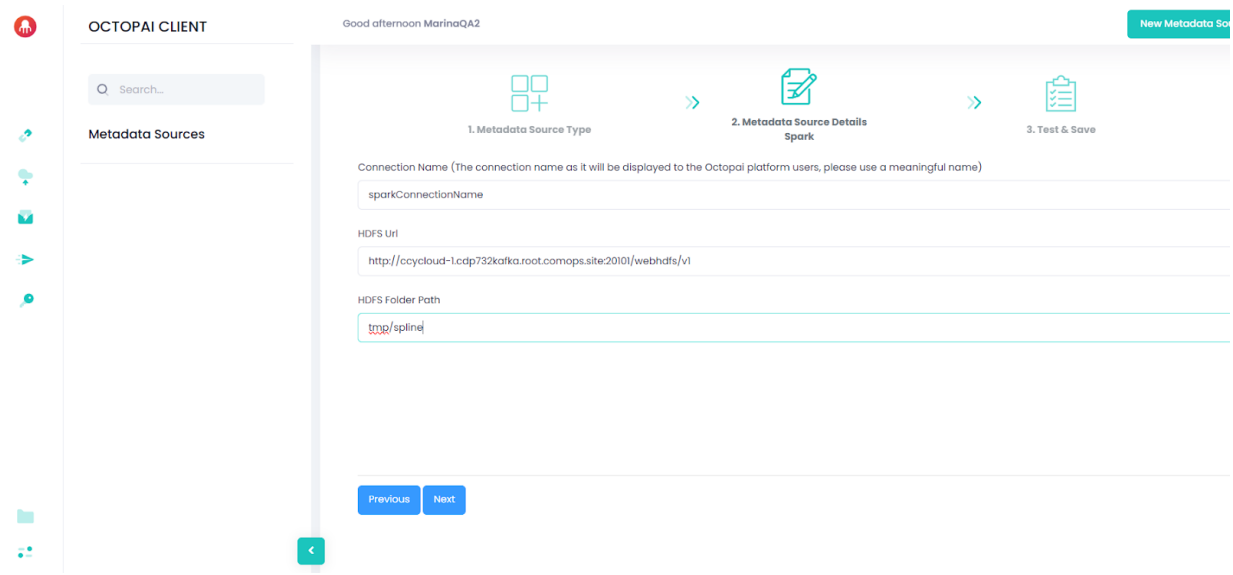
   ```
   spark.jars=hdfs:///tmp/spark-3.5_spark-2.12.jar
   spark.sql.queryExecutionListeners=za.co.absa.spline.harvester.listener.S
   plineQueryExecutionListener
   spark.spline.mode=ENABLED
   spark.spline.lineageDispatcher=hdfs
   spark.spline.lineageDispatcher.hdfs.className=za.co.absa.spline.harvester.
   dispatcher.HDFSLineageDispatcher
   spark.spline.lineageDispatcher.hdfs.directory=hdfs:///tmp/spline
   spark.driver.memory=4g
   ```



4. Create the HDFS Lineage Directory

   Create the directory where lineage files will be written and set permissions:

```
hdfs dfs -mkdir /tmp/spline
hdfs dfs -chown hive /tmp/spline
```



5. Set Permissions

   Ensure the user running the Spark jobs has permission to write lineage files to /tmp/spline.

   This can typically be done by ensuring the Spark job runs under a user who has write access to /tmp/spline in HDFS.

What to do next

Post-Installation Checklist

- Spline Agent JAR is uploaded to HDFS
- Spark cluster configuration is updated with Spline properties
- /tmp/spline folder is created and write-access is configured
- Spark cluster is restarted or the configuration is refreshed
- Test Spark jobs are producing lineage files in /tmp/spline
- For Kerberos-secured environments:
  - Kerberos authentication is validated on Windows.
  - Kerberos authentication is validated inside Linux (WSL).
  - WebHDFS delegation tokens are successfully generated.

- The Octopai Client service is running.

Important Notes

- Reach out to your Cloudera Account Team or Cloudera Support and provide the Spark, Scala, and Java version details used in your environment, and obtain the appropriate Spline connector JARs.
- Ensure the Spline Agent bundle matches your Spark and Scala versions.

Spark / Scala version compatibility matrix

	Scala 2.11	Scala 2.12
Spark 2.2	(no SQL; no codeless init)	—
Spark 2.3	(no Delta support)	—
Spark 2.4	Yes	Yes
Spark 3.0 or newer	—	Yes

- Only successful jobs with persistent outputs will generate lineage.
- If no lineage appears, verify:
  - The job reads and writes to and from persistent sources
  - The Spark job includes the correct configuration parameters. Check the Spark logs
  - The JAR file was correctly uploaded to HDFS and accessible

Appendix A. Sample krb5.ini Configuration:

This appendix provides a reference Kerberos configuration file example. Use this example to validate realm, KDC, and domain mappings. The actual values must be provided by your organization.

```
[libdefaults]
default_realm = ROOT.COMOPS.SITE
dns_lookup_realm = false
dns_lookup_kdc = false
ticket_lifetime = 24h
renew_lifetime = 7d
forwardable = true
[realms]
ROOT.COMOPS.SITE = {
  kdc = ccycloud-1.cdp.root.comops.site
  admin_server = ccycloud-1.cdp.root.comops.site
}

[domain_realm]
.root.comops.site = ROOT.COMOPS.SITE
root.comops.site = ROOT.COMOPS.SITE
```

On Windows, the file must be placed under C:\ProgramData\MIT\Kerberos5\krb5.ini

On Linux, the file must be placed under /etc/krb5.conf

Correct realm and domain mappings are required for successful Kerberos authentication and delegation token acquisition.

Appendix B. Kerberos Verification and Expected Results:

This appendix describes how to verify that the Kerberos authentication chain is functioning correctly on both Windows and Linux. These checks must be completed before the lineage can be written to HDFS in Kerberos-secured environments.

Windows Verification

Run the following command from a PowerShell window:

```
"C:\Program Files\MIT\Kerberos\bin\kinit.exe" -kt C:\Octopai\spark.keytab sp
ark@ROOT.COMOPS.SITE
```

Expected result: No output indicates successful authentication.

If an error occurs, verify the keytab, principal name, realm configuration, and network connectivity to the KDC.

Linux Verification (WSL Ubuntu)

Run the following commands from PowerShell:

```
wsl kinit -kt /mnt/c/Octopai/spark.keytab spark@ROOT.COMOPS.SITE
wsl klist
```

Expected result: The klist output should display a valid Kerberos ticket, including fields such as Valid starting, Expires, and Service principal.

If no ticket is shown, verify that the krb5.conf file exists under /etc, the keytab path is correct, and the Linux environment can access the KDC.

After all Windows and Linux verifications are complete, start the Octopai Client service:

```
Start-Service OctopaiClient
```

Completing these steps successfully verifies that Kerberos authentication and WebHDFS delegation token acquisition are configured correctly.

Code and custom connectors

Code-based, custom, and file-based connectors load metadata from systems that out-of-the-box connectors do not cover, including code files, dbt projects, CSV templates, and folders.

Code-based Lineage Connector

The Cloudera Data Lineage (formerly Octopai) Code Based Lineage connector allows you to upload code (non-SQL) through files to include them in the automated data lineage analysis. You can set this connector up by configuring metadata sources and validating permissions.

Supported file types

The following file types are supported:

- Upload (Discovery display): python file (.py), scala file (.scala)
- Fully Analyzed (All modules supported - AI parsed): python file (.py), scala file (.scala)

Tool permissions prerequisites

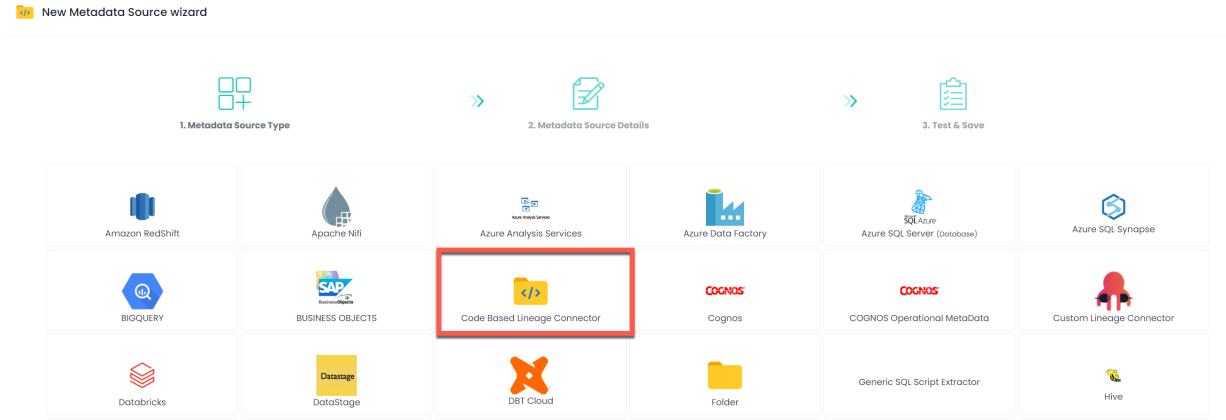
Warning: Missing permissions could result in broken lineages

Enable read permission for Cloudera Data Lineage Windows NT user to the folder (which contains the code-based files).

Setting up Code Based Lineage metadata source

Metadata sources are set on the Octopai Client.

1. From the Octopai Client, click the Code Based Lineage connector.



2. In the **New Metadata Source** wizard, choose the authentication method between a Local Folder or a Github Folder, and enter the appropriate information:

   - **Local Folder**: Choose the Local Folder option if you want to upload files from a local directory for data lineage analysis.
   - **Github Folder**: Choose the Github Folder option if you want to connect to your Github repository to extract the required files for data lineage analysis.

For Local Folder

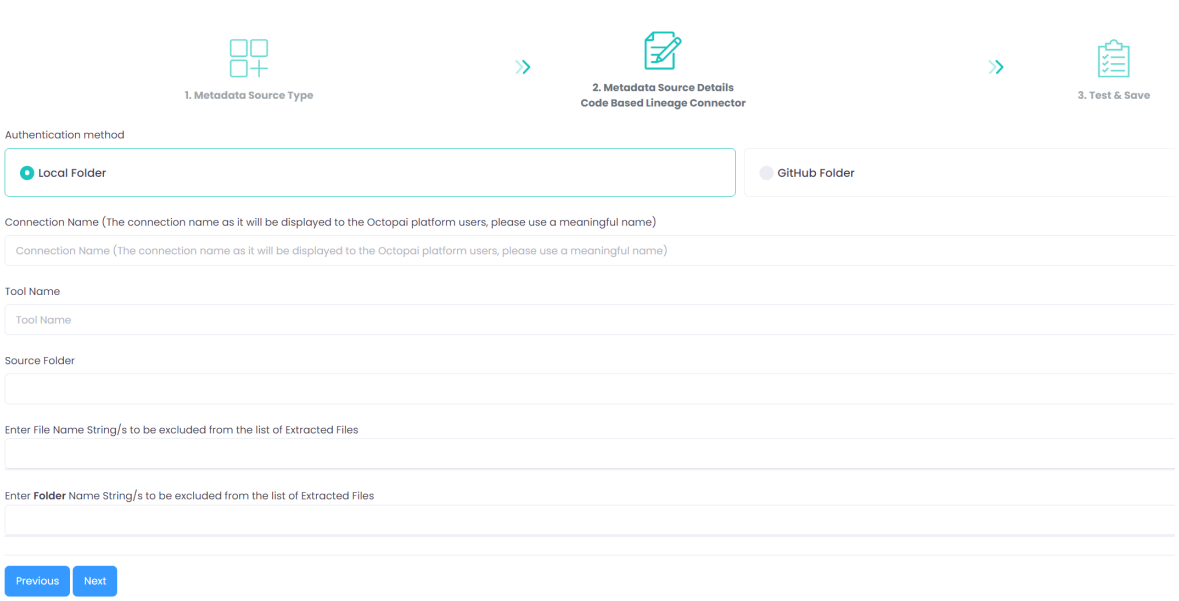
a. Connection Name: Provide a meaningful name for this connection to help you easily identify it later in the Cloudera Data Lineage application.

b. Tool Name: Specify the tool or language environment you are using. For example, enter "Python" or "Scala" depending on the language you are uploading.

c. Source Folder: Click the blank field to select the folder containing the files you want to upload.

d. Exclude Files or Strings: Enter file names, suffixes, strings, or patterns you want to exclude from the export (for example, test scripts, local configuration files).

e. Exclude Folders: Specify folder names to exclude from the export if you have multiple subfolders (for example, venv, __pycache__, .git).



For Github Folder

a. Connection Name: Provide a meaningful name for this connection to help you easily identify it later in the Cloudera Data Lineage application.

b. Tool Name: Specify the tool or language environment you are using. For example, enter "Python" or "Scala" depending on the language you are uploading.

c. Personal Access Token (PAT): Enter your Github Personal Access Token to authenticate and authorize the connector to access your repository.

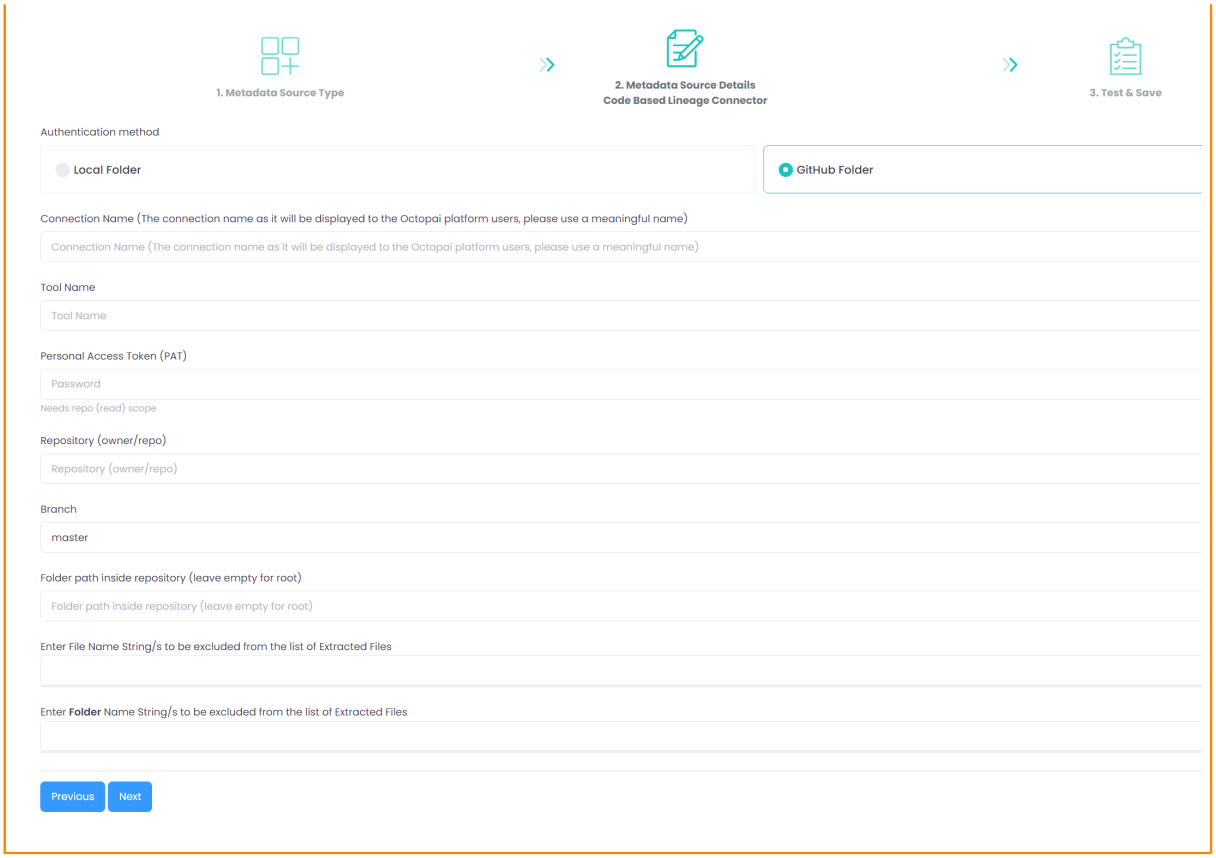
d. Repository (owner/repo): Enter the repository name in the format 'owner/repository-name'.

e. Branch: Specify the branch within the repository from which you want to extract files (for example, 'main').

f. Folder path inside the repository: Provide the specific path to the directory within the repository that contains the files you want to include.

g. Exclude Files or Strings: Enter file names, suffixes, strings, or patterns you want to exclude from the export (for example, test scripts, local configuration files).

h. Exclude Folders: Specify folder names to exclude from the export if you have multiple subfolders (for example, venv, __pycache__, .git).



3. Click Next Finish Run .

Cloudera Data Lineage Custom Lineage Connector (formerly Universal Connector) overview

The Cloudera Data Lineage (formerly Octopai) Custom Lineage Connector lets you load metadata from systems that are not covered by out-of-the-box connectors so that the Data Intelligence Platform can show lineage, discovery, and catalog coverage across your full data landscape in conjunction with the rest of the tools in your landscape.

As data demands evolve, data teams need a clearer picture of the data ecosystem. Cloudera Data Lineage continues to expand out-of-the-box coverage, yet many organizations still rely on custom processes, niche databases, or reporting layers that need the same visibility as first-class integrations.

The Cloudera Data Lineage Custom Lineage Connector fills those gaps. You supply metadata in structured CSV templates, while ingestion and analysis on the platform side are automated so you can connect lineage across systems that would otherwise stay invisible and cause lineage to break.

How it is done

Use the Cloudera Data Lineage CSV templates below. After you upload completed templates, processing on the platform is automated.

What Cloudera Data Lineage offers

This metadata, together with metadata from out-of-the-box supported systems, is analyzed using proprietary algorithms. Cloudera Data Lineage can then show end-to-end column-level lineage, inner-system lineage, cross-system lineage, data discovery, and a data catalog of your full data landscape for users across the organization.

The benefits

- No blind spots – perform changes with confidence.
- Get a clear picture of data transformations.
- Increase visibility of the organization's complete data ecosystem.
- Future-proof your expanding data landscape by providing access to unlimited data systems.
- Add links to our out-of-the-box technologies.

Template guides

Refer to the guide that matches the metadata you are loading:

- Custom Lineage Connector ETL guide
- Custom Lineage Connector databases guide
- Custom Lineage Connector Reporting Tool guide

How to set up the Custom Lineage Connector

For step-by-step setup instructions, see Custom Lineage Connector.

Custom Lineage Connector

Instructions for configuring Cloudera Data Lineage (formerly Octopai) Custom Lineage Connector (formerly Universal Connector) metadata source and required permissions.

Tool Permissions Prerequisites



Warning: Missing permissions could end up in broken lineages.

Enable read permission for Cloudera Data Lineage Windows NT User to the folder (which contains the filled template files).

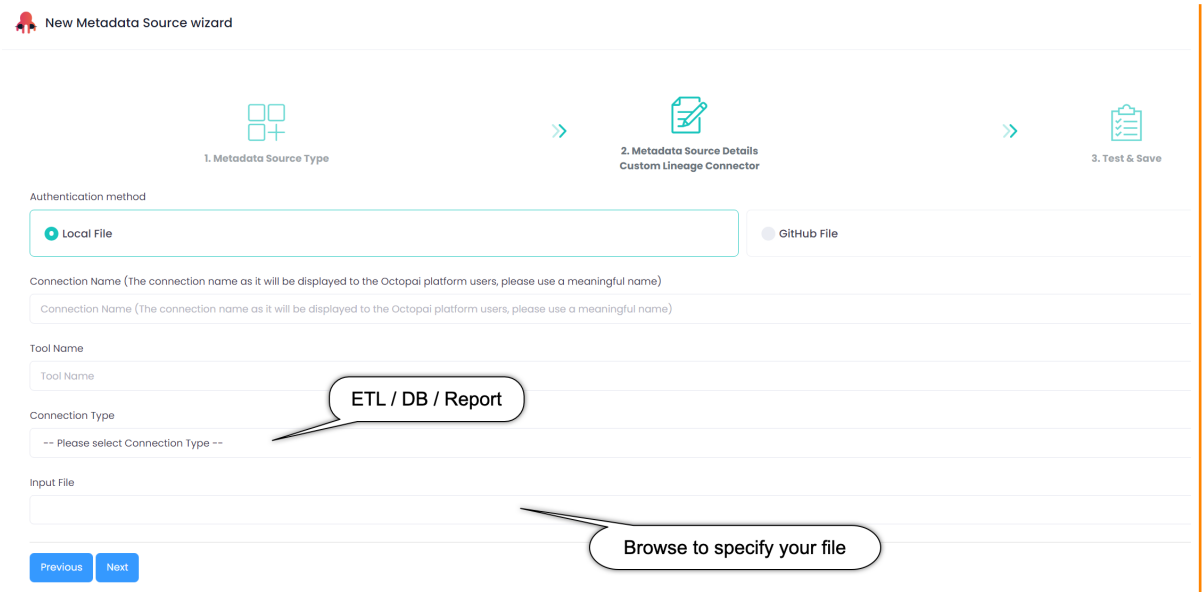
Setting up Custom Lineage Connector Metadata Source

Metadata Sources are set on the Octopai Client.

1. From the Octopai Client, click the Custom Lineage Connector.
2. In the **New Metadata Source** wizard, choose the authentication method between a Local File or a Github File, and enter the appropriate information:
   - **Local File**: Choose the Local File option if you want to upload files from a local directory for data lineage analysis.
   - **Github File**: Choose the Github File option if you want to connect to your Github repository to extract the required files for data lineage analysis.

For Local File

a. Connection Name: Provide a meaningful name for this connection to help you easily identify it later in the Cloudera Data Lineage application.
b. Tool Name: Specify the tool or language environment you are using.
c. Connection Type: Specify if the custom source is a database object, custom ETL process, or a report.
d. Input File: Select the input CSV template file.



For Github File

a. Connection Name: Provide a meaningful name for this connection to help you easily identify it later in the Cloudera Data Lineage application.

b. Tool Name: Specify the tool or language environment you are using.

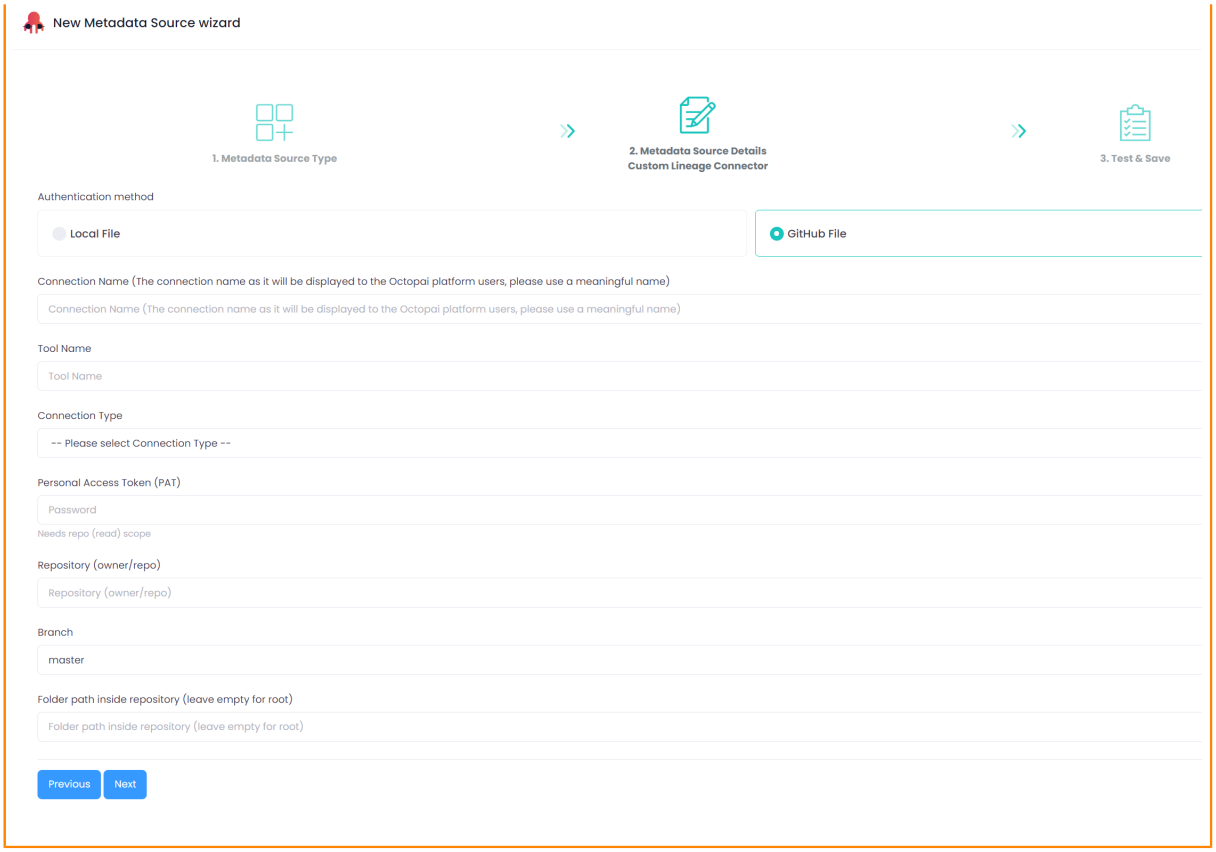
c. Connection Type: Specify if the custom source is a database object, custom ETL process, or a report.

d. Personal Access Token (PAT): Enter your Github Personal Access Token to authenticate and authorize the connector to access your repository.

e. Repository (owner/repo): Enter the repository name in the format 'owner/repository-name'.

f. Branch: Specify the branch within the repository from which you want to extract files (for example, 'main').

g. Folder path inside the repository: Provide the specific path to the directory within the repository that contains the files you want to include.



3. Click Next Finish Run .

Related Information

Cloudera Data Lineage Custom Lineage Connector (formerly Universal Connector) overview

Custom Lineage Connector for ETLs

Use the Custom Lineage Connector (formerly Universal Connector) Links CSV template to describe ETL processes, tasks, and column-level data flows so that Cloudera Data Lineage (formerly Octopai) can show lineage for custom or unsupported ETL and orchestration tools.

The Cloudera Data Lineage platform supports data movement and transformation scenarios across many tools. The Custom Lineage Connector Links CSV template integrates metadata from custom or unsupported ETL and orchestration tools beyond native connectors. You can build a more complete view of your data ecosystem, with data lineage, discovery, and catalog content aligned and connected with the rest of your environment.

This guide explains the Custom Lineage Connector Links CSV structure. It complements database object metadata. When you also need tables and columns defined outside the ETL graph, see Custom Lineage Connector for Database Objects.

How ETL objects are consolidated (object identity)

An ETL object is uniquely identified by the combination of its name fields: Process Name, Process Path, Task Name, and Task Path. Rows that share the same values in all these fields are consolidated into a single object. If any of these fields differ between rows (for example, a different Process Path), Cloudera Data Lineage treats them as separate objects.

To show rows as separate objects, use different values in one or more of these fields.

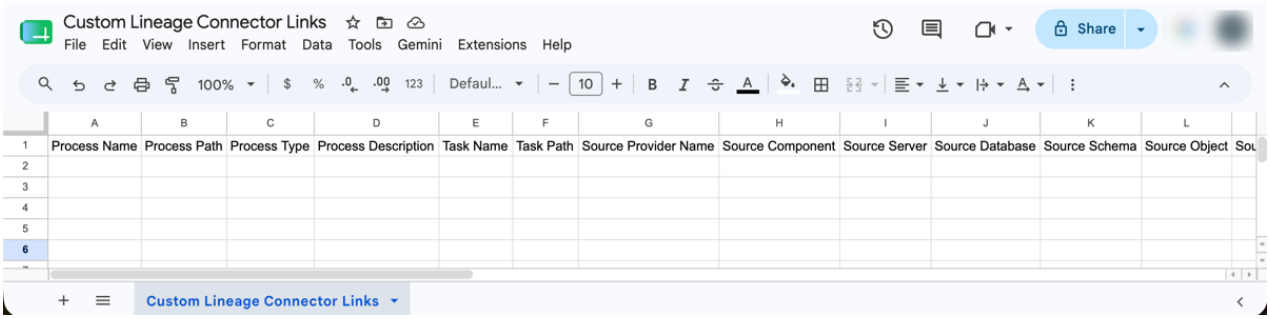
To consolidate rows into a single object while keeping the same Task Name, use one of the following approaches:

1. Populate only Task Name and leave Process Path and the other name fields blank. All rows merge into one object. The path is not visible for that object.

2. Use the same Process Path across all rows with the same Task Name. All rows merge into one object while keeping a Process Path populated. Use this approach when a visible path matters.

How to use the ETLs template file

1. Download the template file: Cloudera Data Lineage Custom Lineage Connector Links
2. Fill in the required fields using the table below.



Column Name	Description	Required
Process Name	Name of the process that wraps the task, for example “Workflow” in Informatica or “Package” in SSIS. This field is part of the object's identity; differing values create separate objects.	No
Process Path	Path of the process – for example, the path where the SSIS package is stored, including the package name and suffix (aaa\bbb\ccc \Package Name.dtsx). This field is part of the object's identity; differing values create separate objects.	No
Process Type	The type of process – job, map, package, and so forth.	Yes
Process Description	Short process description to be identified clearly in the lineages.	No
Task Name	The task name – the atomic unit that holds the data flow within the process. This field is part of the object's identity; differing values create separate objects.	Yes
Task Path	The path of the task – the location of the atomic unit that runs the process (for example, aaa\bbb\ccc\Package Name\container\Task Name). This field is part of the object's identity; differing values create separate objects.	No
Source Provider	The type of database that the source object connects to (for example, Oracle or SQL Server). See Supported script parsing providers.	Yes (required when Source Sql is populated)
Source Component	The name of the logic component in the ETL tool. Example: for Informatica, the name of the aggregator in the map. When there is no component, enter the table name.	No
Source Server	Server name of the source object.	No
Source Database	Database name of the source object.	Yes (optional when Source Sql is populated and specifies the database)
Source Schema	Schema name of the source object.	Yes (optional when Source Sql is populated and specifies the schema)

Column Name	Description	Required
Source Object	Name of the source object. Leave empty when Source Sql is populated; any value entered is ignored.	Yes (leave empty when Source Sql is populated)
Source Column	When Source Sql is populated, the column name parsed from the source SQL that maps to the target column on the same row, and can be left empty. Otherwise, the column name in the source object.	Yes (optional when Source Sql is populated)
Source Data Type	Data type of the column.	No
Source Precision	Precision of the column.	No
Source Scale	Scale of the column.	No
Source Type	Type of source object, such as Table or View. Optional when you use a source SQL script; use this field only to distinguish components that share the same name. Required when Sour ce Sql is not populated.	Yes (optional when Source Sql is populated)
Source Sql	The SQL query that retrieves the source data. When populated, Cloudera Data Lineage parses the script regardless of values in other source fields. If the script specifies schema, database, or server explicitly, those values take precedence. Otherwise, populated template columns are required and used.	No
Target Provider	The type of database that the target object connects to (for example, Oracle or SQL Server). See Supported script parsing providers.	Yes (required when Source Sql is populated)
Target Component	Name of the logic component in the ETL tool. Example: for Informatica, the name of the aggregator in the map. When there is no component, enter the table name.	No
Target Server	Server name of the target object.	No
Target Database	Database name of the target object.	Yes
Target Schema	Schema name of the target object.	Yes
Target Object	Name of the target object.	Yes
Target Column	Column name in the target object.	Yes
Target Data Type	Data type of the column.	No
Target Precision	Precision of the column.	No
Target Scale	Scale of the column.	No
Target Type	The type of target object, such as Table, View, or Stored Procedure (SP).	Yes
Expression	Formula or transformation between source column and target column.	No
Link Type	DataFlow or ImpactAnalysis.	No (default = DataFlow)
Link Description	Documentation about the link.	No (default = empty string)

Important:

- All parameters are string values.
- Ensure the structure of the template files remains unchanged.
- Column names must match the template header row exactly, including capitalization and spacing.
- Values are not case sensitive.

Troubleshoot

Match the symptom to a cause and resolution.

Symptom	Cause	Resolution
One logical ETL object appears as multiple or duplicate objects	Process Name, Process Path, Task Name, and Task Path are not identical across rows that should consolidate. A differing Process Path or Task Path is the most common cause.	Verify that all four fields match across those rows. To merge rows into one object, see How ETL objects are consolidated (object identity).

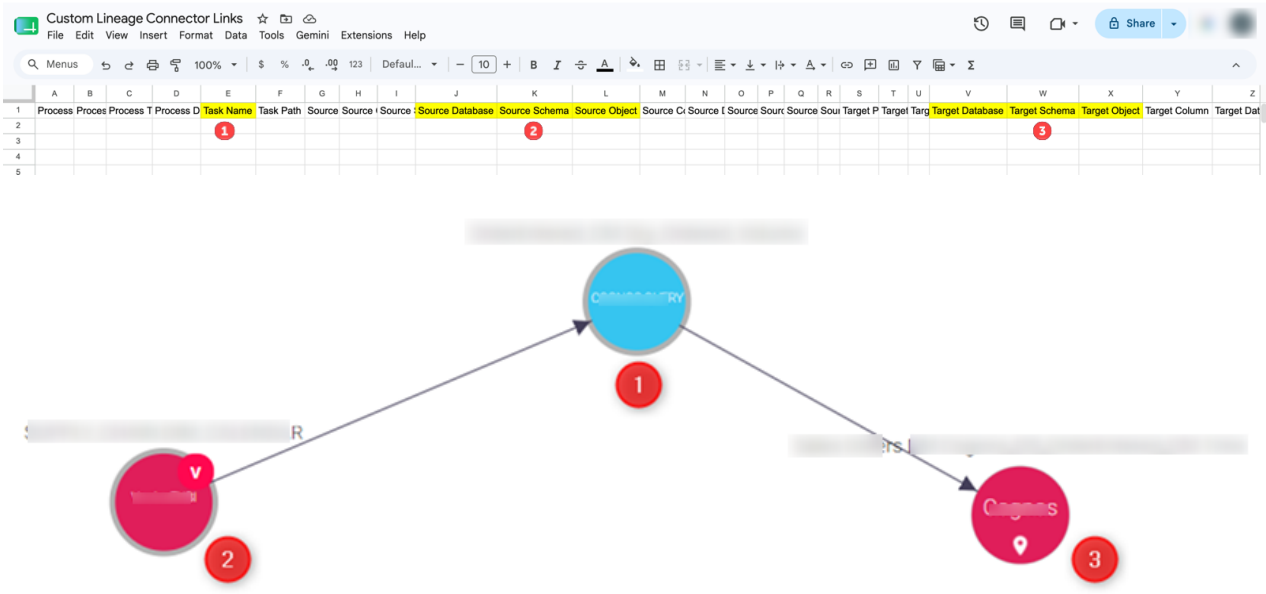
Supported script parsing providers

Supported parsing providers for scripts:

- DB2
- HIVE
- IMPALA
- MSSQL
- SYNAPSE (MSSQL)
- NETEZZA
- ORACLE
- POSTGRESQL
- REDSHIFT
- TERADATA
- SNOWFLAKE
- VERTICA
- BIGQUERY
- HANA

Example for ETL process on cross-system lineage

The Custom Lineage Connector links the source and the target for the task name as the main object.



Custom Lineage Connector for Database Objects

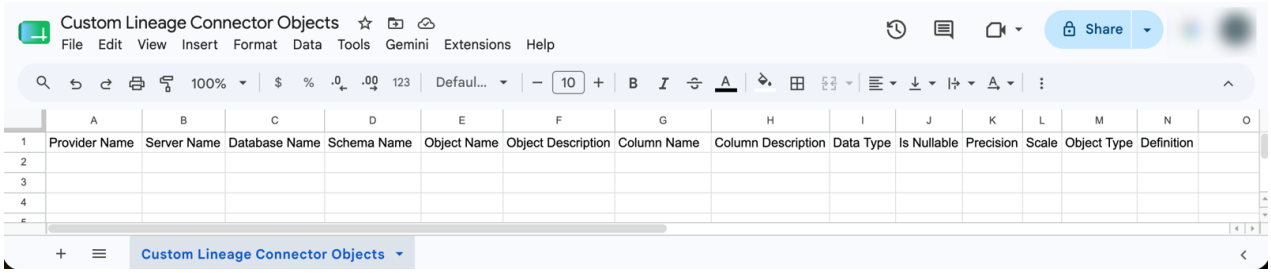
Use the Custom Lineage Connector (formerly Universal Connector) Objects CSV template to register databases, schemas, objects, and columns so that Cloudera Data Lineage (formerly Octopai) can enrich the catalog and lineage for systems without a native connector.

The Custom Lineage Connector Objects template integrates metadata from databases and catalog objects in systems that are not covered by native connectors. You can build a more complete view of your data ecosystem, with data lineage, discovery, and catalog content aligned and connected with the rest of your environment.

This guide explains the Custom Lineage Connector Objects CSV template structure. It complements ETL process and task metadata. When you also need transformations and column-level data flows defined inside the ETL graph, see Custom Lineage Connector for ETLs.

How to use the Database Objects template file

1. Download the template file: Custom Lineage Connector Objects
2. Fill in the required fields using the table below.



Column Name	Description	Required
Provider Name	The type of database that the object connects to (for example, Oracle or SQL Server). See Supported script parsing providers.	No
Server Name	Server name of the object.	No
Database Name	Database name of the object.	Yes
Schema Name	Schema name of the object.	Yes
Object Name	Name of the source object.	Yes
Object Description	Documentation about the object.	No (default = empty string)
Column Name	Column name in the source object.	Yes
Column Description	Short column description.	No (default = empty string)
Data Type	Data type of the column.	No
Is Nullable	Indicates whether the column accepts null values.	No
Precision	Precision of the column.	No
Scale	Scale of the column.	No
Object Type	Type of object – table, view, file, and so on.	Yes
Definition	The definition text for the object (for example DDL or SQL).	No

Important:

Note:

- All parameters are string values.
- Ensure the structure of the template files remains unchanged.
- Column names must match the template header row exactly, including capitalization and spacing.
- Values are not case sensitive.

Supported script parsing providers

Supported parsing providers for scripts:

- DB2
- HIVE
- IMPALA
- MSSQL
- SYNAPSE (MSSQL)
- NETEZZA
- ORACLE
- POSTGRESQL
- REDSHIFT
- TERADATA
- SNOWFLAKE
- VERTICA
- BIGQUERY
- HANA

Custom Lineage Connector for Reports

Use the Custom Lineage Connector (formerly Universal Connector) Reports CSV template to describe reporting assets and their connections to data sources so that Cloudera Data Lineage (formerly Octopai) can integrate metadata from reporting and BI tools beyond native connectors.

The Cloudera Data Lineage platform supports expansive analysis and visualization scenarios. The Custom Lineage Connector for Reporting Tools integrates metadata from a diverse range of systems that are not covered by native connectors (for example Tableau, Power BI, Cognos, MicroStrategy, Qlik, Looker, SSRS, and similar tools). You can build a more complete view of your data ecosystem, with data lineage, discovery, and catalog content aligned and connected with the rest of your environment.

This guide explains the Custom Lineage Connector Reports CSV structure.

How to use the Reports template file

1. Download the template file: Custom Lineage Connector Reports
2. Fill in the required fields in the template file using the information provided in the table below.

Column Name	Description	Required
Model Name	The name of the model, such as the name of the package in Cognos, the RPD name in OBIEE, or the Universe name in Business Object.	Yes
Report Path	The unique path of the report, per ConnectionID.	Yes
Report Name	The name of the report.	Yes
Source Provider	The type of database that the report connects to. See Supported script parsing providers.	Yes (required when Source Sql is populated)

Column Name	Description	Required
Source Layer Name	The name of the source layer or semantic object that contains the source object or column.	No
Source Server	The server name of the source object.	No
Source Schema	The schema name of the source object.	Yes (optional when Source Sql is populated and specifies the schema)
Source Db	The database name of the source object.	Yes (optional when Source Sql is populated and specifies the database)
Source Object	The name of the source object. Leave empty when Source Sql is populated; any value entered is ignored.	Yes (leave empty when Source Sql is populated)
Source Column	When Source Sql is populated, the column name parsed from the source SQL that maps to the target column on the same row, and can be left empty. Otherwise, the column name in the source object.	Yes (optional when Source Sql is populated)
Source Data Type	The data type of the column.	No
Source Type	The type of source object, such as Table, View, or Stored Procedure (SP). Optional when you use a source SQL script; use this field only to distinguish components that share the same name. Required when Source Sql is not populated.	Yes (optional when Source Sql is populated)
Source Precision	The precision of the column.	No
Source Scale	The scale of the column.	No
Source Sql	The SQL query that retrieves the report's source data. When populated, Cloudera Data Lineage parses the script regardless of values in other source fields. If the script specifies schema, database, or server explicitly, those values take precedence. Otherwise, populated template columns are required and used.	No
Target Server	The server name of the target object.	No
Target Layer Name	The name of the component in the report, such as a table, cross-tab, chart, etc.	Yes
Target Schema	The schema name of the target object.	No
Target Db	The database name of the target object.	No
Target Table	The table name in the target object.	No
Target Column	The column name in the target object.	Yes
Target Data Type	The data type of the column.	No
Target Precision	The precision of the column.	No
Target Scale	The scale of the column.	No
Link Description	Documentation about the link. Default = Null.	No
Link Type	Data Flow or Impact Analysis. Default = Data Flow.	No
Expression	The formula or transformation between the source column and target column.	No

Important:

Note:

- All parameters are string values.
- Ensure the structure of the template files remains unchanged.
- Column names must match the template header row exactly, including capitalization and spacing.
- Values are not case sensitive.

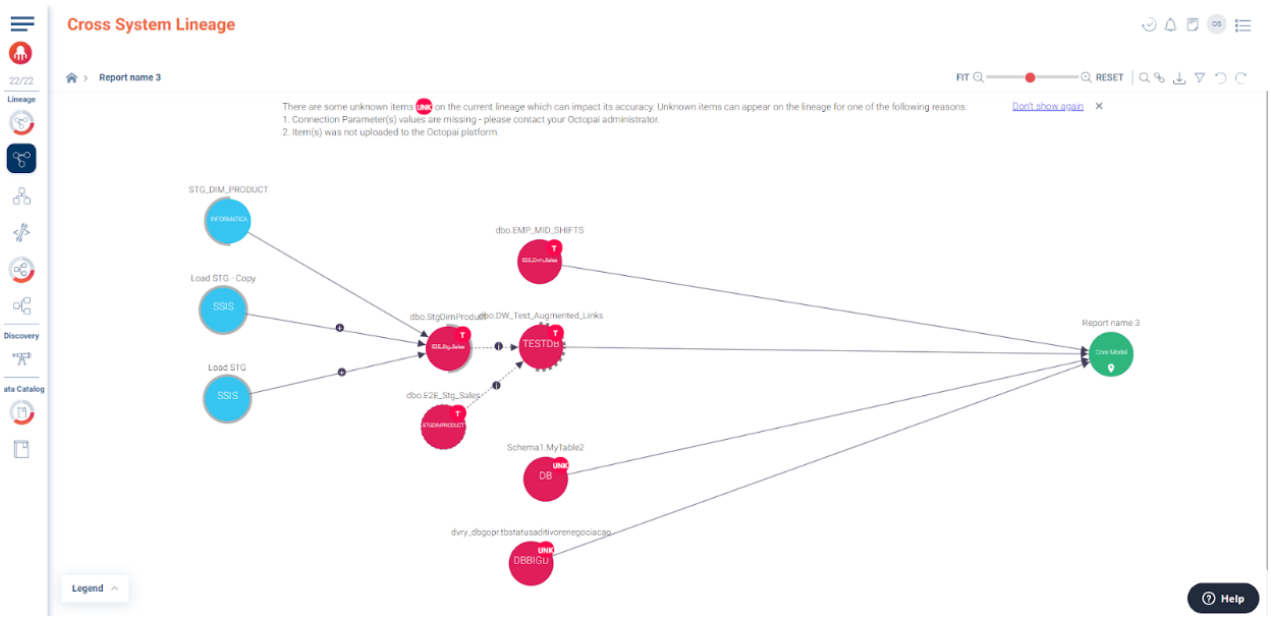
Supported script parsing providers

Supported parsing providers for scripts:

- DB2
- HIVE
- IMPALA
- MSSQL
- SYNAPSE (MSSQL)
- NETEZZA
- ORACLE
- POSTGRESQL
- REDSHIFT
- TERADATA
- SNOWFLAKE
- VERTICA
- BIGQUERY
- HANA

Example for report cross-system lineage

Example of a report cross-system lineage created as part of an upload of the Custom Lineage Connector:



DBT Files

Set up DBT project files as a textual metadata source in Cloudera Data Lineage (formerly Octopai) and review the permissions required for ingestion.

Tool Permissions Prerequisites

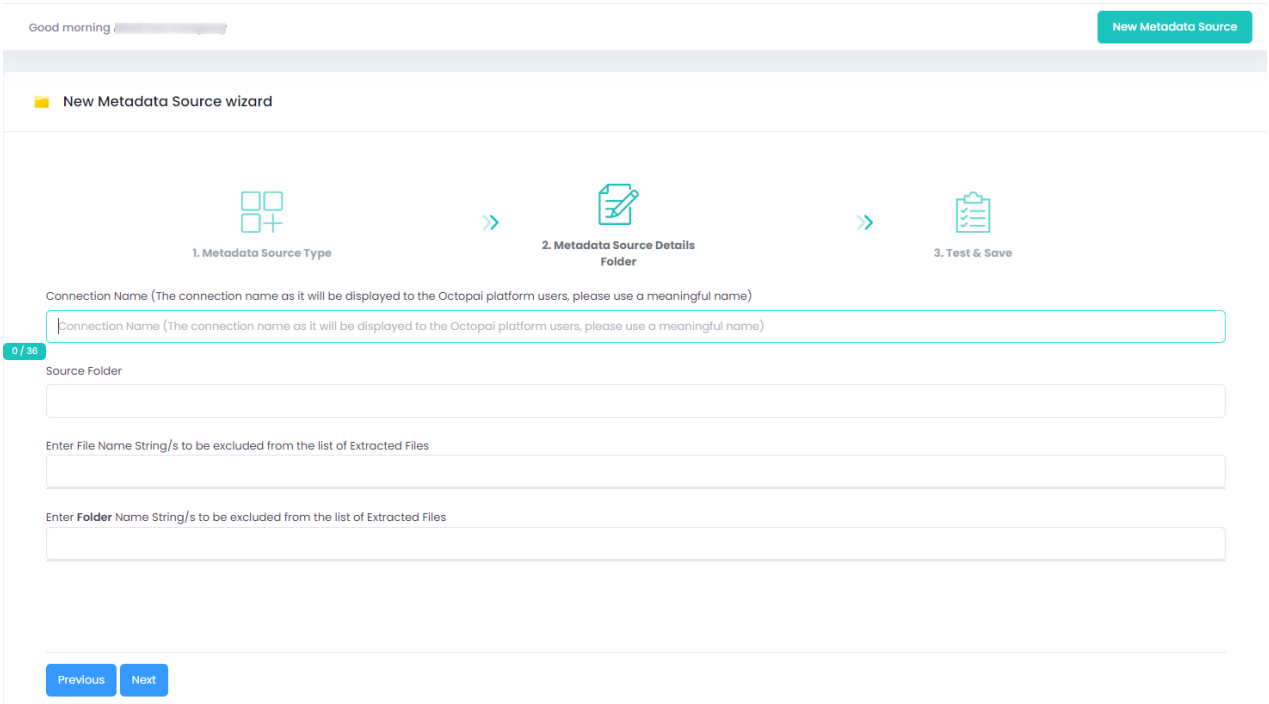
Warning: Missing permissions could end up in broken lineages.

- Enable read permission for Cloudera Data Lineage Windows NT User to the following path: \run\<projectName>\models
- The above folder should contain the DBT *.sql files.

Setting up Textual Files Metadata Source

Metadata Sources are set on the Octopai Client

Please use the Textual Files metadata source and fill in the \run<projectName>\models*.sql path on the source folder attribute



Textual Files or Folder connector

The Textual Files or Folder connector allows you to extract file metadata from directories. You can set this connector up by configuring metadata sources and validating permissions.

Supported file types

The following file types are supported:

- Upload (Discovery Display): text file, CSV file, SQL file, log file, XML file, JSON file, bash file, output file, BTEQ file, BTQ file, control file, report file, Perl file, KornShell file, C# file, Siddhi file
- Fully Analyzed (All Modules Supported): SQL file, BTEQ file, BTQ file, control file, Siddhi file

Limitations

The following limitations apply to the Textual Files or Folder connector:

- Archive files, including .zip and other archive formats, are not supported. The connector does not extract archives, and nested archives (for example, a .zip inside another .zip) are also not extracted.
- Only files placed directly in the configured Source Folder are analyzed. Files that remain inside an archive are ignored, even if the archive itself is in the source folder.
- Uploading a .zip or other archive through the Cloudera Data Lineage (formerly Octopai) portal does not trigger connector processing. The archive must be fully extracted, and the individual files placed in the source folder, before the connector runs.

Tool Permissions Prerequisites



Warning: Missing permissions could end up in broken lineages.

Enable read permission for Cloudera Data Lineage Windows NT User to the folder (which contains the textual files).

Setting up Textual Files Metadata Source

Metadata Sources are set on the Octopai Client

1. From the Octopai Client, click the Folder connector.
2. In the **New Metadata Source** wizard, choose the authentication method between a Local Folder or a Github Folder, and enter the appropriate information:
   - **Local Folder**: Choose the Local Folder option if you want to upload files from a local directory for data lineage analysis.
   - **Github Folder**: Choose the Github Folder option if you want to connect to your Github repository to extract the required files for data lineage analysis.

For Local Folder

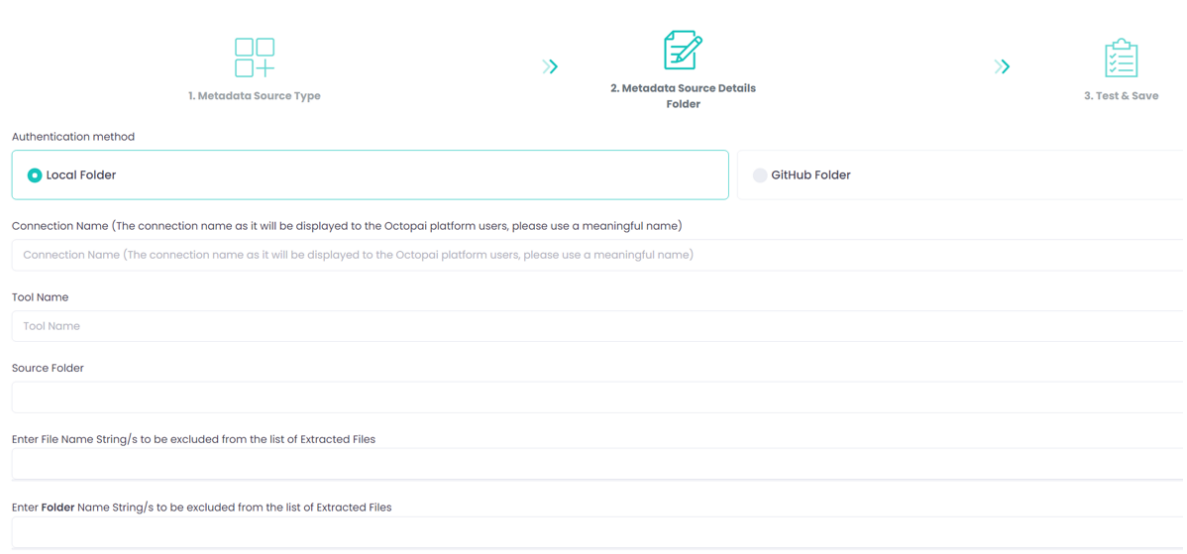
a. Connection Name: Provide a meaningful name for this connection to help you easily identify it later in the Cloudera Data Lineage application.

b. Tool Name: Specify the tool or language environment you are using. For example, enter "JSON" if you are uploading JSON scripts.

c. Source Folder: Click the blank field to select the folder containing the files you want to analyze. The folder must contain individual, extracted files of a supported type. Archives (for example, .zip files) and nested archives are not extracted. Extract them first and place the raw files (for example, .sql) directly in this folder. Remove any archives and unrelated files from the source folder.

d. Exclude Files or Strings: Enter file names, suffixes, strings, or patterns you want to exclude from the export.

e. Exclude Folders: Specify folder names to exclude from the export if you have multiple subfolders.



For Github Folder

a. Connection Name: Provide a meaningful name for this connection to help you easily identify it later in the Cloudera Data Lineage application.

b. Tool Name: Specify the tool or language environment you are using. For example, enter "JSON" if you are uploading JSON scripts.

c. Personal Access Token (PAT): Enter your Github Personal Access Token to authenticate and authorize the connector to access your repository.

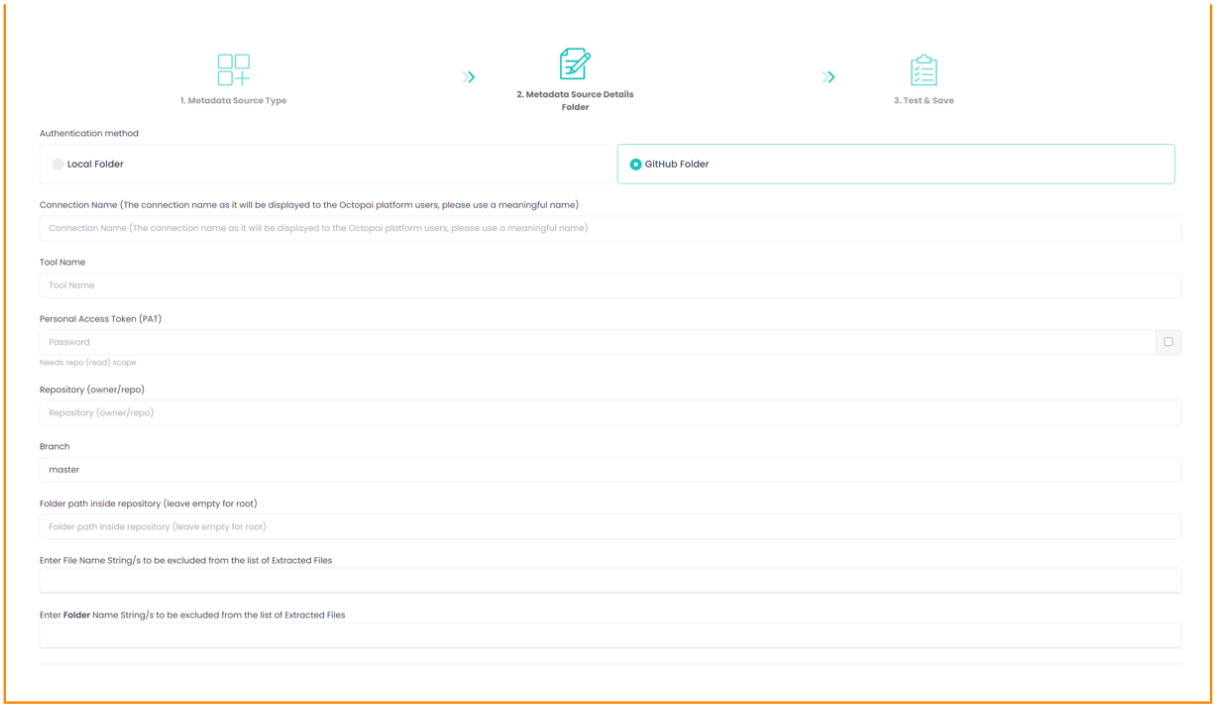
d. Repository (owner/repo): Enter the repository name in the format 'owner/repository-name'.

e. Branch: Specify the branch within the repository from which you want to extract files (for example, 'main').

f. Folder path inside the repository: Provide the specific path to the directory within the repository that contains the files you want to include. The path must contain individual, extracted files of a supported type — archives (for example, .zip files) are not extracted.

g. Exclude Files or Strings: Enter file names, suffixes, strings, or patterns you want to exclude from the export.

h. Exclude Folders: Specify folder names to exclude from the export if you have multiple subfolders.



3. Click Next Finish Run .

Troubleshooting

The connector run completes successfully but returns no scripts (empty result). This usually means the Source Folder did not contain any individually extracted, supported files at the time of the run. To resolve this issue:

1. Confirm the source folder contains individual files of a supported type (for example, .sql) and not .zip or other archives. Fully extract any archives, including nested archives, and place the raw files directly in the source folder.
2. Remove archives and unrelated files from the source folder.
3. Verify that the Cloudera Data Lineage Windows NT user has read permission on the Source Folder and its contents (see Tool Permissions Prerequisites).
4. Rerun the connector.