



Informatica® Metadata Command Center
November 2025

Databricks Sources

© Copyright Informatica LLC 2023, 2025

This software and documentation are provided only under a separate license agreement containing restrictions on use and disclosure. No part of this document may be reproduced or transmitted in any form, by any means (electronic, photocopying, recording or otherwise) without prior consent of Informatica LLC.

U.S. GOVERNMENT RIGHTS Programs, software, databases, and related documentation and technical data delivered to U.S. Government customers are "commercial computer software" or "commercial technical data" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, the use, duplication, disclosure, modification, and adaptation is subject to the restrictions and license terms set forth in the applicable Government contract, and, to the extent applicable by the terms of the Government contract, the additional rights set forth in FAR 52.227-19, Commercial Computer Software License.

Informatica, Informatica Cloud, Informatica Intelligent Cloud Services, PowerCenter, PowerExchange, and the Informatica logo are trademarks or registered trademarks of Informatica LLC in the United States and many jurisdictions throughout the world. A current list of Informatica trademarks is available on the web at <https://www.informatica.com/trademarks.html>. Other company and product names may be trade names or trademarks of their respective owners.

Portions of this software and/or documentation are subject to copyright held by third parties. Required third party notices are included with the product.

The information in this documentation is subject to change without notice. If you find any problems in this documentation, report them to us at infa_documentation@informatica.com.

Informatica products are warranted according to the terms and conditions of the agreements under which they are provided. INFORMATICA PROVIDES THE INFORMATION IN THIS DOCUMENT "AS IS" WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING WITHOUT ANY WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND ANY WARRANTY OR CONDITION OF NON-INFRINGEMENT.

Publication Date: 2025-11-20

Table of Contents

Preface.	5
Chapter 1: Introduction to Databricks catalog sources.	6
Extraction and view process.	7
About the Databricks catalog source.	8
Extracted metadata.	8
Compatible functionalities.	9
Data profiling for Databricks objects.	12
Compatible connectors.	13
Chapter 2: Before you begin.	14
Verify permissions.	14
Permissions to extract metadata.	14
Permissions to run data profiles.	15
Permissions to perform data classification.	15
Permissions to perform relationship discovery.	15
Permissions to perform glossary association.	15
Create a connection.	15
Personal access token authentication.	18
OAuth machine-to-machine authentication.	18
Advanced connection properties.	18
Analytics and job cluster properties.	19
Import a relationship inference model.	19
Chapter 3: Create catalog sources in Metadata Command Center.	21
Step 1. Register a catalog source.	21
Step 2. Configure capabilities.	23
Configure metadata extraction.	23
Configure lineage discovery.	29
Configure data profiling and quality.	30
Configure data classification.	33
Configure relationship discovery.	34
Configure glossary association.	35
Step 3. Associate stakeholders and asset groups.	35
Step 4. Run or schedule the job.	37
Step 5. Assign reference catalog source connections to endpoint catalog source objects.	38
Chapter 4: View results in Data Governance and Catalog.	40
View metadata extraction results.	40
View referenced source systems.	42

View data lineage.	42
View lineage at the catalog source level.	43
View lineage at the data set level.	43
View lineage at the data element level.	44
View data profiling results	44
View data observability results	45
View classified data.	46
View glossary associations.	46

Preface

Read *Databricks Sources* to learn how to register and configure Databricks source systems as catalog sources in Metadata Command Center. After you configure a catalog source, you extract metadata and then view the results in Data Governance and Catalog.

CHAPTER 1

Introduction to Databricks catalog sources

You can use Metadata Command Center to extract metadata from a source system.

A source system is any system that contains data or metadata. For example, Databricks is a source system from which you can extract metadata through a Databricks catalog source. A catalog source is an object that represents and contains metadata from the source system.

Before you extract metadata from a source system, you first create and register a catalog source that represents the source system. Then you configure capabilities for the catalog source. A capability is a task that Metadata Command Center can perform, such as metadata extraction, lineage discovery, data profiling, data classification, or glossary association.

When Metadata Command Center extracts metadata, Data Governance and Catalog displays the extracted metadata and its attributes as technical assets. You can then perform tasks such as analyzing the assets, viewing lineage, and creating links between those assets and their business context.

The following table describes the capabilities of the catalog source:

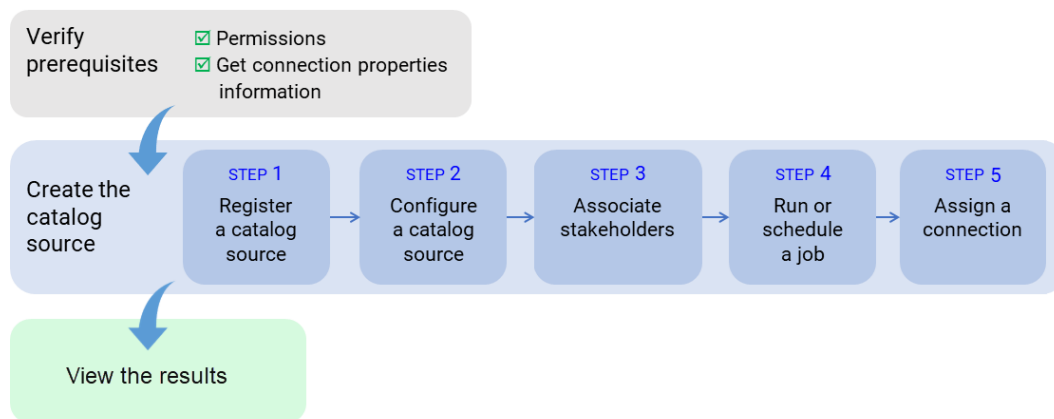
Capability	Description
Incremental metadata extraction	<p>An incremental metadata extraction extracts only the changed and new objects since the last catalog source job run. Incremental metadata extraction doesn't remove deleted objects from the catalog and doesn't extract metadata of code-based objects if applicable.</p> <p>Applicable only to Databricks Unity Catalog.</p> <p>Applies to the following Databricks Unity Catalog objects:</p> <ul style="list-style-type: none">- Table- View
Advanced Programming Language Parsing	<p>Advanced Programming Language Parsing parses the source system code in addition to extracting objects from the source system.</p>
Lineage Discovery	<p>Builds the complete lineage of a catalog source by recommending endpoint catalog source objects to assign to reference catalog source connections. When you run the catalog source job, Metadata Command Center assigns the reference catalog source connections to CLAIRE recommended endpoint catalog source objects. You can then view the list of CLAIRE recommendations and accept or reject them.</p>
Data Profiling and Quality	<ul style="list-style-type: none">- Data Profiling. Assesses source metadata and analyzes the collected statistics to discover content and structure, such as value distribution, patterns, and data types.- Data Quality. Measures the reliability of the data and enables data usage.- Data Observability. Identifies anomalies in the characteristics of the data.

Capability	Description
Data Classification	Data classification is the process of identifying and organizing data into relevant categories based on the functional meaning of the data. Classifying data can help your organization manage risks, compliance, and data security.
Glossary Association	You can associate terms that are in the glossary with technical assets to provide user-friendly business names to technical assets. Glossary Association automatically associates glossary terms with technical assets or recommends glossary terms that you can manually associate with technical assets in Data Governance and Catalog.

Extraction and view process

To extract metadata from a source system, configure the catalog source and run the catalog source job in Metadata Command Center. Then view the results in Data Governance and Catalog.

The following image shows the process to extract metadata from a source system:



After you verify prerequisites, perform the following tasks to extract metadata from Databricks:

1. Register a catalog source. Create a catalog source object, select the source system, and select the connection.
2. Configure the catalog source. Specify the runtime environment and configure parameters for metadata extraction. Optionally, add filters to include or exclude source system assets from metadata extraction. You can also configure other capabilities such as data profiling and quality.
3. Optionally, associate stakeholders. Associate users with technical assets, giving the users permission to perform actions determined by their roles.
4. Run or schedule the catalog source job.
5. Optionally, if the catalog source job generates referenced asset objects, you can assign a connection to referenced source system assets.
You can view the lineage with object references without performing connection assignment. After connection assignment, you can view the objects.
Run the catalog source again after you assign connections to referenced source system assets.

After you run the catalog source job, you view the results in Data Governance and Catalog.

About the Databricks catalog source

You can use the Databricks catalog source to extract metadata from Databricks source system.

Databricks combines data warehouses and data lakes into an AI-driven Databricks Lakehouse platform.

You can run connection-aware scans on Databricks sources.

You can use SQL warehouse or all-purpose clusters to extract metadata.

Extracted metadata

You can use the Databricks catalog source to extract metadata from Databricks source system.

You can extract AI Model Core and AI Model Core Version objects from Databricks Unity Catalog source systems. Additionally, you can retrieve lineage captured by Databricks Unity Catalog.

Note: Databricks Unity Catalog retains lineage data for 90 days.

You can extract metadata from Databricks notebooks if they use the following technologies:

- Markdown
- Python
- SQL

You can extract the following objects from a Databricks workspace:

- Calculation
- Command
- Experiment
 - Experiment Run
- Folder
- Job Parameter
- Live Table
- Live View
- Notebook Definition
- Notebook Instance
- Notebook Parameter
- Notebook Task
- Pipeline Definition (SQL based)
- Result
- Run Job Task
- Streaming Table
- Streaming View
- Task Parameter
- Workflow Job Definition
- Workflow Job Instance

You can extract the following objects from Databricks Unity Catalog:

- Column
 - Primary Key
 - Foreign Key
- Database
- External Column
- External Table
- File
- File System
- Schema
- Table
- View

You can extract the following complex data type columns along with their nested fields from Databricks source systems:

- Map
- Struct
- Array

Compatible functionalities

Databricks offers integration with a diverse range of modules and programming languages.

You can use Databricks with the following programming languages:

- Python
- SQL
- Markdown

You can use Databricks with the following Python functionalities:

- Standard language constructions
- Standard built-in functions
- Partially-compatible modules:

Note: Data Governance and Catalog processes only a subset of library functions of partially-compatible modules.

- abs
- adal
- argparse
- array
- ast
- azure
- base64
- binascii
- calendar

- codecs
- collections
- concurrent
- contextlib
- contextvars
- copy
- copyreg
- csv
- dataclasses
- datetime
- dbutils
- decimal
- delta
- difflib
- distutils
- email
- enum
- errno
- fnmatch
- fractions
- functools
- gc
- genericpath
- gettext
- glob
- graphframes
- hashlib
- heapq
- hmac
- importlib
- inspect
- io
- itertools
- json
- keyword
- locale
- logging
- math
- matplotlib

- nt
- numbers
- numpy
- operator
- os
- pandas
- pathlib
- pickle
- pkgutil
- posix
- posixpath
- pprint
- py4j
- pyodbc
- pyspark
- pytz
- random
- re
- reprlib
- requests
- seaborn
- secrets
- shutil
- simplejson
- six
- sklearn
- smtplib
- socket
- ssl
- stat
- string
- struct
- subprocess
- sys
- teradatasql
- textwrap
- threading
- time
- traceback

- types
- typing
- urllib
- urllib3
- uuid
- warnings
- weakref
- xml
- yaml
- zipfile
- zlib

- Custom libraries

Note: Custom libraries are libraries created by a user.

Note: If Databricks catalog source detects an incompatible function or library, it can't process the statement. It skips the statement and continues to process the next one.

You can also use DeltaLake SQL in SQL commands and PySpark calls.

Data profiling for Databricks objects

Configure data profiling to run profiles on the metadata extracted from a Databricks source system. You can use all-purpose clusters or SQL warehouse to run profiles. You can also run profiles on Databricks Unity Catalog objects. You can view the profiling statistics in Data Governance and Catalog.

You can run profiles on the following Databricks Unity Catalog objects:

- Delta Table
- External Table in Delta, Parquet, and CSV formats
- View

You can view the profiling statistics in Data Governance and Catalog. The data profiling task runs profiles on the following data types:

- Bigint
- Boolean
- Date
- Decimal
- Double
- Float
- Int
- Smallint
- String
- Tinyint
- Timestamp

Compatible connectors

Before you configure a Databricks catalog source, you must connect to Databricks source system.

Use the Databricks connector to connect to Databricks source system.

Note: To enable profiling, perform the following prerequisite steps before you configure a Databricks connector in Administrator:

1. Create a folder with the name `informatica.databricksdelta` in the `<Secure Agent installation directory>/ext/connectors/thirdparty/` folder.
2. Copy the `SparkJDBC42.jar` file to the `informatica.databricksdelta` folder. The minimum version of the Spark server must be 3.x.
3. Run the `./infaagent startup` script located in the `<Secure Agent installation directory>/apps/agentcore` folder to restart the Secure Agent.

CHAPTER 2

Before you begin

Before you create a catalog source, ensure that you have the information required to connect to the source system.

Perform the following tasks:

- Assign the required permissions.
- Configure a connection to the Databricks source system.
- Optionally, if you want to identify pairs of similar columns and relationships between tables within a catalog source, import a relationship inference model.

Verify permissions

To extract metadata and to configure other capabilities that a catalog source might include, you need account access and permissions on the source system. The permissions required might vary depending on the capability.

Permissions to extract metadata

Ensure that you have the required permissions to enable metadata extraction.

- Grant the following permissions for non-unity catalog sources:
 - show tables
 - show schemas
 - describe table extended
 - describe schema extended
 - show catalogs
- To extract Databricks Unity Catalog lineage with system tables, grant access permissions on the following system tables:
 - system.access.column_lineage
 - system.access.table_lineage

Note: If you don't have access to system tables, the metadata extraction job uses REST APIs to extract lineage data.

- To extract Databricks Unity Catalog table metadata with system tables, you need the following permissions:

- USE CATALOG permission on the Databricks Unity Catalog
- SELECT and USE SCHEMA permissions on the INFORMATION_SCHEMA schema

The metadata extraction job uses the following INFORMATION_SCHEMA schema tables to extract metadata:

- TABLES
- COLUMNS
- SCHEMATA
- VIEWS
- KEY_COLUMN_USAGE
- REFERENTIAL_CONSTRAINTS
- CATALOGS

Note: If you don't have access to the INFORMATION_SCHEMA tables, the metadata extraction job uses the `Show` and `Describe` commands to extract metadata.

Permissions to run data profiles

Ensure that you have the required permissions to run profiles.

Grant the following permissions:

- Read permission for Databricks tables in the Databricks source system.
- Access to Databricks workspace and Databricks SQL in the Databricks source system.

Permissions to perform data classification

You don't need any additional permissions to run data classification.

Permissions to perform relationship discovery

You don't need any additional permissions to run relationship discovery.

Permissions to perform glossary association

You don't need any additional permissions to run glossary association.

Create a connection

Before you configure the Databricks catalog source, create a connection object in Administrator.

1. In Administrator, select **Connections**.
2. Click **New Connection**.
3. Enter the connection details.

The following table describes the basic connection properties:

Property	Description
Connection Name	Name of the Databricks connection. Must be unique within the organization. Connection names can contain alphanumeric characters, spaces, and the following special characters: _ . + - Maximum length is 255 characters.
Description	Optional description of the connection. Maximum length is 4000 characters.
Type	Type of connection. Ensure that the type is Databricks.
Use Secret Vault	Stores sensitive credentials for this connection in the secrets manager that is configured for your organization. This property appears only if secrets manager is set up for your organization. This property is not supported by Data Ingestion and Replication. When you enable the secret vault in the connection, you can select which credentials that the Secure Agent retrieves from the secrets manager. If you don't enable this option, the credentials are stored in the repository or on a local Secure Agent, depending on how your organization is configured. For information about how to configure and use a secrets manager, see <i>Secrets manager configuration</i> in the Administrator help.

Property	Description
Runtime Environment	<p>The name of the runtime environment where you want to run tasks.</p> <p>Select a Secure agent, Hosted Agent, or serverless runtime environment.</p> <p>Hosted Agent is not applicable for mappings in advanced mode.</p> <p>You cannot run an application ingestion and replication, database ingestion and replication, or streaming ingestion and replication task on a Hosted Agent or serverless runtime environment.</p>
SQL Warehouse JDBC URL	<p>Databricks SQL Warehouse JDBC connection URL. Required to connect to a Databricks SQL warehouse. Also applies to Databricks clusters.</p> <p>Note: Databricks SQL Serverless is the recommended Databricks cluster type.</p> <p>To get the SQL Warehouse JDBC URL, go to the Databricks console and select the JDBC driver version from the JDBC URL menu.</p> <p>If you use the personal access token authentication type and select JDBC URL version 2.6.22 or earlier, use the following syntax:</p> <pre>jdbc:spark://<Databricks Host>:443/default;transportMode=http;ssl=1;AuthMech=3;httpPath=/sql/1.0/endpoints/<SQL endpoint cluster ID>;</pre> <p>If you use the personal access token authentication type and select JDBC URL version 2.6.25 or later, use the following syntax:</p> <pre>jdbc:databricks://<Databricks Host>:443/default;transportMode=http;ssl=1;AuthMech=3;httpPath=/sql/1.0/endpoints/<SQL endpoint cluster ID>;</pre> <p>Application ingestion and replication and database ingestion and replication tasks can use JDBC URL version 2.6.25 or later or 2.6.22 or earlier. The URLs must begin with the prefix <code>jdbc:databricks://</code>, as follows:</p> <pre>jdbc:databricks://<Databricks Host>:443/default;transportMode=http;ssl=1;AuthMech=3;httpPath=/sql/1.0/endpoints/<SQL endpoint cluster ID>;</pre> <p>If you use the OAuth machine-to-machine authentication type and select JDBC URL version 2.6.36 or later, use the following syntax: <code>jdbc:databricks://<Databricks Host>:443/default;transportMode=http;ssl=1;AuthMech=11;Auth_Flow=1;httpPath=/sql/1.0/endpoints/<SQL endpoint cluster ID>;</code></p> <p>Ensure that you set the required environment variables in the Secure Agent. Also specify the correct JDBC Driver Class Name under advanced connection settings.</p> <p>Note: Specify the database name in the Database Name connection property. If you specify the database name in the JDBC URL, it is not considered. The Databricks Host, Organization ID, and Cluster ID properties are not considered if you configure the SQL warehouse JDBC URL property.</p>

4. Select the authentication type to connect to Databricks and enter the required properties.

You can use the following authentication types:

- Personal Access Token
- OAuth Machine-to-Machine

5. Click **Test**.
6. Click **Save**.

Personal access token authentication

The following table describes the connection properties for personal access token authentication:

Property	Description
Databricks Token	Personal access token to access Databricks. This property is required for SQL warehouse, all-purpose cluster, and job cluster.

OAuth machine-to-machine authentication

The following table describes the connection properties for OAuth machine-to-machine authentication:

Property	Description
Client ID	The client ID of the service principal.
Client Secret	The client secret associated with the Client ID of the service principal. Log in to your Databricks account to generate the client secret.

Advanced connection properties

The following table describes the advanced connection properties:

Property	Description
JDBC Driver Class Name	The name of the JDBC driver class. Optional for SQL warehouse and Databricks cluster. For JDBC URL versions 2.6.22 or earlier, specify the driver class name as <code>com.simba.spark.jdbc.Driver</code> . For JDBC URL versions 2.6.25 or later, specify the driver class name as <code>com.databricks.client.jdbc.Driver</code> .

Analytics and job cluster properties

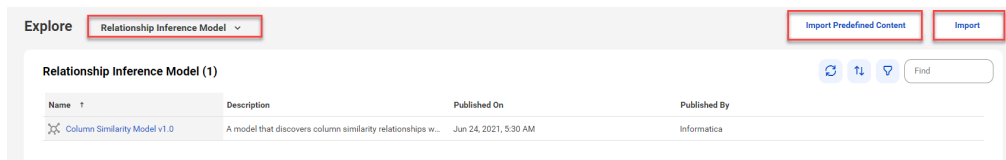
The following table describes the analytics and job cluster properties:

Property	Description
Databricks Host	<p>The host name of the endpoint the Databricks account belongs to. Required for Databricks cluster. Doesn't apply to SQL warehouse.</p> <p>You can get the Databricks Host from the JDBC URL. The URL is available in the Advanced Options of JDBC or ODBC in the Databricks all-purpose cluster.</p> <p>The following example shows the Databricks Host in JDBC URL: <code>jdbc:spark:// <Databricks Host> :443/ default;transportMode=http; ssl=1;httpPath=sql/ protocolv1/o/ <Org Id>/<Cluster ID>; AuthMech=3; UID=token; PWD=<personal-access-token></code></p> <p>The value of PWD in Databricks Host, Organization Id, and Cluster ID is always <code><personal-access-token></code>.</p>
Cluster ID	<p>The ID of the cluster. Required for Databricks cluster. Doesn't apply to SQL warehouse.</p> <p>You can get the cluster ID from the JDBC URL. The URL is available in the Advanced Options of JDBC or ODBC in the Databricks all-purpose cluster.</p> <p>The following example shows the Cluster ID in JDBC URL: <code>jdbc:spark://<Databricks Host>:443/ default;transportMode=http; ssl=1;httpPath=sql/ protocolv1/o/<Org Id>/ <Cluster ID>; AuthMech=3;UID=token; PWD=<personal-access-token></code></p>
Organization ID	<p>The unique organization ID for the workspace in Databricks. Required for Databricks cluster. Doesn't apply to SQL warehouse.</p> <p>You can get the Organization ID from the JDBC URL. The URL is available in the Advanced Options of JDBC or ODBC in the Databricks all-purpose cluster.</p> <p>The following example shows the Organization ID in JDBC URL: <code>jdbc:spark://<Databricks Host>:443/ default;transportMode=http; ssl=1;httpPath=sql/ protocolv1/o/ <Organization ID> / <Cluster ID>;AuthMech=3;UID=token; PWD=<personal-access-token></code></p>

Import a relationship inference model

Import a relationship inference model if you want to configure the relationship discovery capability. You can either import a predefined relationship inference model, or import a model file from your local machine.

1. In Metadata Command Center, click **Explore** on the navigation panel.
2. Expand the menu and select **Relationship Inference Model**. The following image shows the **Explore** page with the **Relationship Inference Model** menu:



3. Select one of the following options:
 - **Import Predefined Content.** Imports a predefined relationship inference model called Column Similarity Model v1.0.

- **Import.** Imports the predefined relationship inference model from your local machine. Select this if you previously imported predefined content into your local machine and the inference model is stored on the machine.

To import a file, click **Choose File** in the **Import Relationship Inference Model** window and navigate to the model file on your local machine. You can also drag and drop the file.

The imported models appear in the list of relationship inference models on the **Relationship Discovery** tab.

CHAPTER 3

Create catalog sources in Metadata Command Center

Use Metadata Command Center to configure a catalog source for Databricks and run the catalog source job.

When you configure a catalog source, you define the source system that you want to extract metadata from. Configure filters to include or exclude source system metadata before you run the job. Optionally, configure other capabilities, such as data profiling and quality, data classification, relationship discovery, and glossary association.

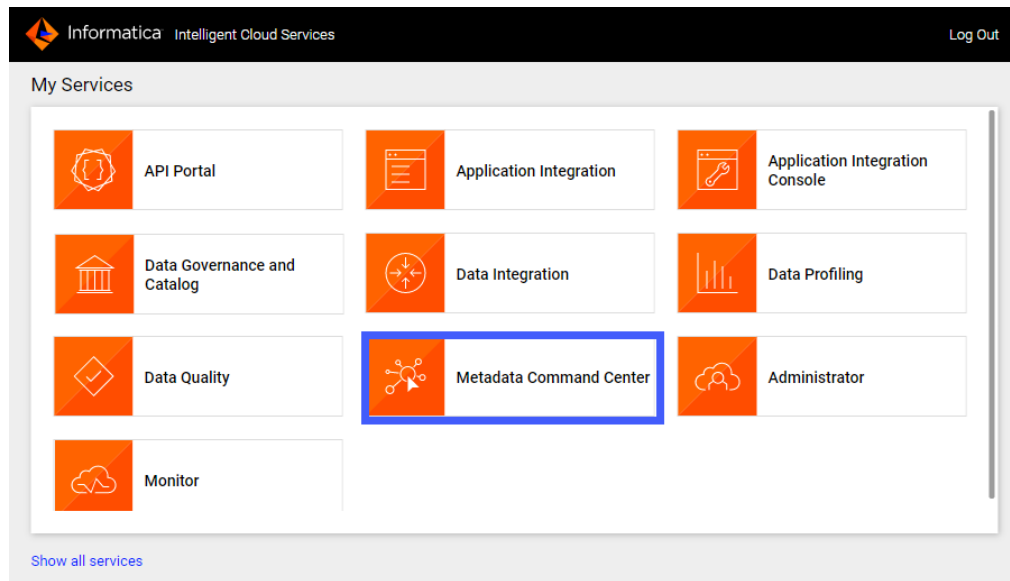
To provide stakeholders access to technical assets, you can assign access through roles. If your catalog source references other source systems, you can create a connection assignment to the endpoint catalog source to view complete lineage.

Step 1. Register a catalog source

When you register a catalog source, provide general information and connection values.

1. Log in to Informatica Intelligent Cloud Services.
The **My Services** page appears.
2. Click **Metadata Command Center**.

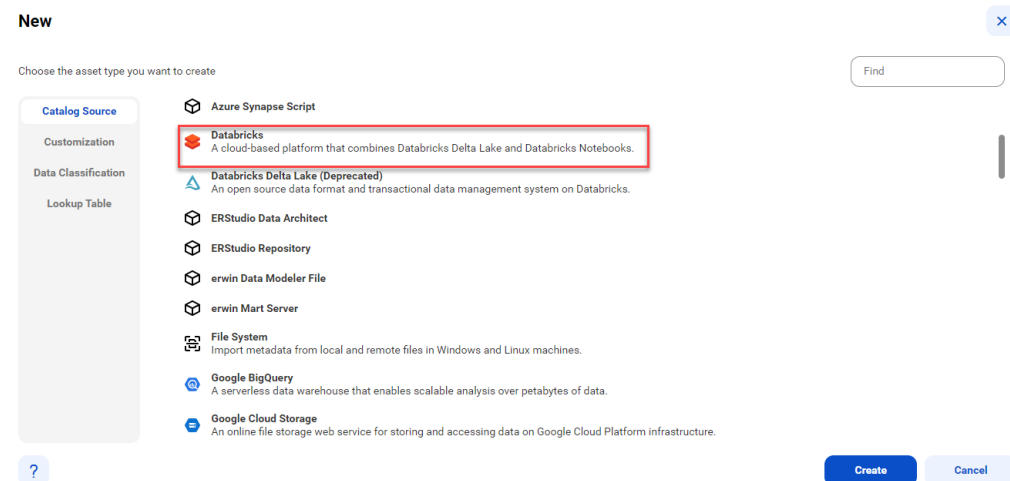
The following image shows the Metadata Command Center box on the **My Services** page:



The Metadata Command Center home page appears.

3. Click **New**.
4. Select **Catalog Source** from the list of asset types.
5. Select Databricks from the list of catalog source types.

The following image shows the Databricks catalog source:



6. Click **Create**.

The **New Catalog Source** page opens.

7. In the **General Information** section, enter a name and an optional description for the catalog source.

Note: You can rename a catalog source after you create it, but to apply the change to all associated objects you must rerun the metadata extraction job.

After you save the catalog source, you can update the description in Metadata Command Center and Data Governance and Catalog. The update appears only in the service in which you update it.

8. In the **Connection Information** area, select the connection that you created in Administrator.

Note: To create or edit a catalog source, you need permissions on the connection to the source system. Select a connection that you have access to, or ask the administrator to grant the necessary permissions to the connection that you want to use.

9. Click **Connection Properties** to expand and view the connection properties for the selected connection.
10. Click **Test Connection** to test your connection to the source system.
11. Click **Next**.

The **Configuration** page appears.

Step 2. Configure capabilities

When you configure the Databricks catalog source, you define the settings for the metadata extraction capability and other optional capabilities.

The metadata extraction capability extracts source metadata from external source systems. You can also configure other capabilities that the catalog source includes.

You can save the catalog source configuration at any point after you enter the connection information. After you save the catalog source, you can choose to run the catalog source job. To run the job once, click **Run**. To run metadata extraction and other capabilities on a recurring schedule, configure schedules on the **Schedule** tab.

Configure metadata extraction

When you configure the Databricks catalog source, you choose a runtime environment, define filters, and enter configuration parameters for metadata extraction.

Before you configure metadata extraction, configure runtime environments in the Informatica Intelligent Cloud Services Administrator.

1. In the **Connection and Runtime** area, choose a serverless runtime environment or the Secure Agent group where you want to run catalog source jobs.

Note: Serverless runtime environment options are available if the catalog source works with a serverless runtime environment.

2. Choose to retain, delete, or deprecate objects that are deleted from the source system in the catalog with the **Metadata Change Option**.
 - **Retain.** Retains objects that are deleted from the source system in the catalog. If you update or add a filter, the catalog retains objects extracted from the previous job and extracts additional objects that match the current filter. Objects deleted from the source system are not deleted from the catalog. Enrichments added on deleted objects and relationships are retained.
 - **Delete.** Deletes metadata from the catalog based on objects deleted from the source system and changes you make to the filter. Enrichments added on deleted objects and relationships are also permanently lost. Objects renamed in the source system are removed and recreated in the catalog.

- **Deprecate.** The lifecycle of objects imported into the catalog moves to Obsolete based on objects deleted from the source system and changes you make to the filter. This does not impact enrichments added on deprecated objects and relationships. Objects renamed in the source system are removed and recreated in the catalog. When you run the catalog source job again for other capabilities such as data classification, relationship discovery, or glossary association, the job doesn't consider obsolete objects. Obsolete objects remain in the catalog until they are purged when you run a **Purge Obsolete Objects** job on the **Explore** page.

Note: You can also change the configured metadata change option when you run a catalog source.

3. In the **Filters** area, define one or more filter conditions to apply for metadata extraction:
 - a. Select **Filter conditions**.
 - b. From the Include/Exclude list, choose to include or exclude metadata based on the filter parameters.
 - c. From the Object type list, select **Notebooks Path**, **Table**, **Workflow Job**, **Pipeline**, or **AI Model** depending on the object that you want to extract metadata from.

The following list describes the object types:

- **Notebooks Path** filters Databricks notebooks.
- **Table** filters Databricks Unity Catalog assets.
- **Workflow Job** filters Databricks workflows.
- **Pipeline** filters Delta Live Table pipelines.
- **AI Model** filters Databricks Unity Catalog AI Models.

The following image shows the filter condition options:

- d. From the Filter criteria list, select **Pattern**.
- e. Click **Select**.
- f. In the **Select values** dialog box, enter the path and click **OK**.

Each pattern can contain wildcards. Use a question mark to represent a single character. For example, A? matches A1, Ab. Use an asterisk to represent multiple characters or empty text. For example, A* matches A, Ab, ABC. For object hierarchies, use relevant separators, such as a dot for Table object type and '/' for Notebooks Paths object type. When you enter values for filters, enclose them in double quotes if you include spaces before or after a string value.

The Notebooks Path that you enter in the value field is relative to the Notebook Workspace Path that you enter in the Databricks parameters.

Examples:

- To include or exclude metadata from the 'customers' table located in the 'db_schema' schema that belongs to the 'catalog_dept' catalog, select **Table** as the object type and enter `catalog_dept.db_schema.customers` in the value field.

- To include or exclude metadata from tables with names that start with 'cust' located in schemas with names that start with 'db_sch' and belong to catalogs with names that start with 'catalog_dep', select **Table** as the object type and enter `catalog_dep*.db_sch*.cust*` in the value field.
- To include or exclude all metadata from all notebooks located in the workspace path, select **Notebooks Path** as the object type and enter `*` in the value field.
- To include or exclude all metadata from notebooks with names that start with 'Cust' followed by a single character located in the workspace path '/Users/username@domain.com/Deltalake_Notebooks', select **Notebooks Path** as the object type and enter `/username@domain.com/Deltalake_Notebooks/Cust?` in the value field.
- To include or exclude the notebook named Cust_Notebook located in the workspace path '/Users/username@domain.com/Deltalake_Notebooks', select **Notebooks Path** as the object type and enter `/username@domain.com/Deltalake_Notebooks/Cust_Notebook` in the value field.
- To include or exclude all the jobs with names that start with 'Job1' followed by a single character, select **Workflow Job** as the object type and enter `Job1?` in the value field.
- To include or exclude all the jobs with names that start with 'Job1', select **Workflow Job** as the object type and enter `Job1*` in the value field.
- To include or exclude all pipelines with names that start with 'Pipeline1' followed by a single character, select **Pipeline** as the object type and enter `Pipeline1?` in the value field.
- To include or exclude metadata from all AI Models in the 'db_schema' schema in the 'catalog_dept' catalog source, select **AI Model** as the object type and enter `catalog_dept.db_schema.*` in the value field.
- To include or exclude metadata from the 'iris_classifier_model' AI Model in the 'db_schema' schema in the 'catalog_dept' catalog source, select **AI Model** as the object type and enter `catalog_dept.db_schema.iris_classifier_model` in the value field.

To include or exclude different object types, click the **Add** icon to add filters with the OR condition.

4. Optionally, in the **Configuration Parameters** area, enter properties to override default context values and job parameters.

Note: Click **Show Advanced** to view all configuration parameters.

The following table describes the properties that you can enter for Databricks:

Parameter	Description
Lineage from Unity	Select one of the following metadata extraction options for Databricks Unity Catalog: <ul style="list-style-type: none"> - Disabled. Default option. Doesn't extract lineage from Unity Catalog. - Enabled for filtered assets. Extracts lineage from assets selected in filter conditions. For external tables included in the input filter, the lineage extends to or from the source file on which the external table was created. For assets not selected in the filter conditions, reference assets are created. - Enabled for all assets. Extracts complete lineage including referenced resources.
Notebooks Workspace Path	The absolute path to the Databricks Notebooks that you want to extract on the remote workspace host.
Notebooks Python Modules Path	Advanced parameter. The absolute path to the Python user modules of Databricks Notebooks located on the Secure Agent machine.

Parameter	Description
Databricks Environment Initialization File Path	Advanced parameter. The absolute path to the Python code file that defines the mount points and other environment properties related to the Databricks source. Specify the file path to resolve mount points when you extract metadata from the Databricks source.
Python Default Variables Values	Python default variable values for Databricks notebooks. Use [VARIABLES], [FUNCTIONS], or [GLOBAL] sections in the form of [SECTION NAME] before you specify a variable.
Notebooks Preload Paths	Advanced parameter. The paths to the Databricks workspace folders with the notebooks that you want to preload. The paths are relative to the Notebooks Workspace Path. If you don't enter any paths, all the notebooks added in Notebook Workspace Path parameter get preloaded.

The following table describes the property that you can enter for additional settings:

Note: The **Additional Settings** section appears when you click **Show Advanced**.

Property	Description
Expert Parameters	Enter additional configuration options to be passed at runtime. Required if you need to troubleshoot the catalog source job. Caution: Use expert parameters when it is recommended by Informatica Global Customer Support.

- Click each tab to enable and configure additional capabilities for the catalog source.

Python default variables values

Provide Python default variables values when your script uses values that are not defined in the code.

Ensure that the Python default variables values include either both **VARIABLES** and **FUNCTIONS** sections or a **GLOBAL** section.

To escape special characters such as `\n` or `\t`, use the backslash (`\`). For example, to define `E:\file\tgtParameterized.csv`, enter `E:\\file\\tgtParameterized.csv`

Variables

Use the following syntax: `<VariableName>=<VariableExpression>`

Note: A variable name doesn't require quotes if it contains only standard alphanumeric characters. If a variable name contains special characters, use double quotes.

A sample variable section can have the following structure:

```
[VARIABLES]
a = 42
b = 7
c = a < b ? a + 7 : b - 7 // It's 49
d = e(1,2) + 1 // Expression use call to function.
"User::table" = 'table' || "User::tableSuffix" // It's a string table concatenated with
the value of User::tableSuffix variable
```

Functions

Use the following syntax: `<FunctionCallSignature>=<FunctionExpression>`

Consider the following rules and guidelines when you define functions:

- A function name doesn't require quotes if it contains only standard alphanumeric characters.
- Enclose a function name in double quotes if it contains special characters.
- Define the list of arguments within parentheses.
- Use a question mark for each function argument.
- Ensure that arguments consist of question marks separated by commas.

A sample function section can have the following structure:

```
[FUNCTIONS]
a(?) = 1
a(?,?) = 2
b(?) = a(1) + 1 // Expression use call to another function.
c(?,?) = d + 2 // Expression use reference to variable.
```

You can provide additional sections to match functions. To match an overloaded function, provide placeholders for its arguments. You can also reference matched function arguments inside the matched section.

A sample custom function section can have the following structure:

```
[host.db.schema.func(x)]
z=x
[host.db.schema.func(x,y)]
z=x+y
```

The following table describes the functions that you can use:

Function	Description
Hash(str, maxOutputLen)	<p>Applies the Message Digest Algorithm 5 (MD5) to an input string and produces a hash value. You can specify the length of the hash value.</p> <p>For example: Hash('abcdefgh', 4) -> 'E8DC'</p> <p>'E8DC' is the result that you get in MD5 hashing algorithm application for the input string 'abcdefgh' with a specified hash length of 4 characters.</p>
Replace(str, from, to)	<p>Replaces a string with another string. For example: Replace('abc', 'b', 'D') -> 'aDc'</p> <p>The function replaces the single occurrence of the substring 'b' in the input string 'abc' with the string 'D' and creates the modified string 'aDc'.</p>
ReplaceRegexp(str, regex, replacement)	<p>Replaces the strings that you specify with Java regular expressions. For example:</p> <ul style="list-style-type: none">- ReplaceRegexp('abcde', 'b.*', 'f') -> 'af'- ReplaceRegexp('graph_id20', 'id(\d+)', '\$1') -> 'graph_20' <p>In the first example, the substring 'bcde' matches the regular expression pattern 'b.*'. The function replaces the matched substring with the replacement string 'f'. As a result, the modified string returned by the function is 'af'.</p> <p>In the second example the substring 'id20' matches the regular expression pattern 'id(\d+)'. The function replaces the entire matched substring with the captured digits '20'. As a result, the modified string returned by the function is 'graph_20'.</p>

Function	Description
StringLengthLimit(str, limit, hashSize)	<p>Limits the length of the input string based on a specified limit. If the length of the string exceeds the limit, the function appends a hash of the remaining characters using the Hash function, where 'hashSize' specifies the size of the hash.</p> <p>For example:</p> <ul style="list-style-type: none"> - StringLengthLimit('abcc', 3, 2) -> 'a26' - StringLengthLimit('abcc', 4, 2) -> 'abcc' <p>In the first example, the input string 'abcc' is longer than the specified limit of 3 characters and it is truncated to 'a'. The remaining characters are replaced with the hash value '26'.</p> <p>In the second example, the function does not modify or truncate the input string because its length matches the specified limit of 4 characters. The function returns the original string 'abcc'.</p>
StringLengthLimit(str, limit)	<p>Limits the length of the input string based on the specified limit. If the length of the string exceeds the limit, the function appends a hash of the remaining characters using the Hash function with a default hash size of 8 characters.</p> <p>For example: StringLengthLimit('abcdabcdabcd', 10) -> 'abE340600C'</p> <p>In this example, the function limits the length of the input string 'abcdabcdabcd' based on the specified limit of 10 characters.</p>
RegexMatch(pattern, str)	<p>Tests whether the input string matches a specified pattern. For example:</p> <pre>RegexMatch('[A-Za-z]+', 'Abcd') -> TRUE</pre> <p>In this example, the function returns 'TRUE' because the input string 'Abcd' contains alphabetic characters that satisfy the pattern of one or more occurrences of alphabetic characters specified by [A-Za-z]+.</p>
Upper(str)	<p>Converts the characters in a given input string to uppercase.</p> <p>For example: Upper('Abc') -> 'ABC'</p>
Lower(str)	<p>Converts the characters in a given input string to lowercase.</p> <p>For example: Lower('aBC') -> 'abc'</p>
Date(text,format)	<p>Returns objects that represent a date in a specified format. For example:</p> <pre>Date('2017-10-31', 'yyyy-MM-dd')</pre> <p>Note: The function follows the conventions and patterns provided by the SimpleDateFormat class in Java 8.</p>
Contains/ ContainsIgnoreCase(stack, needle)	<p>Checks if a given "needle" string is present within a "stack" string.</p> <p>Examples:</p> <ul style="list-style-type: none"> - Contains('abc', 'ab') -> TRUE - Contains(' abc', 'AB') -> FALSE - ContainsIgnoreCase(' abc', 'AB') -> TRUE - Contains('stack', 'needle') -> FALSE <p>In the examples, TRUE means that the function contains a given substring and FALSE means that it doesn't.</p>

Global

You can use a GLOBAL section. It contains both variables and function definitions. If you use a GLOBAL section, don't add the VARIABLES or FUNCTIONS sections.

Use the following syntax: DefaultVariable = 'DefaultValue'

Note: A sample global section can have the following structure:

```
[GLOBAL]
Table = 'DefaultTable'
a = 42
b = 7
c = a < b ? a + 7 : b - 7
d = e(1,2) + 1a(?) = 1
```

Note: A variable name doesn't require quotes if it contains only standard alphanumeric characters. If a variable name contains special characters, use double quotes.

Simple expression

Use the Simple Expression Language to define a value for a variable or a function call. The Simple Expression Language allows you to use values from already defined variables and functions, as well as variables from the current job context.

You can use the following case-sensitive literals:

- Integer: 42
- String: 'str'
- String with quoted single quote: 'str\''
- Boolean: true, false

You can use the following features and functionalities of the Simple Expression Language:

- Equals: 42 == 7 -> FALSE
- Non equals: 42 != 7 -> TRUE
- Concatenations: 'str1' + 'str2' -> 'str1str2'
- Function calls: Replace('abc', 'b', 'd') -> 'adc'
- Ternary operators:
 - 42 == 7 ? 'a' : 'b' -> 'b'
 - 42 != 7 ? 'a' : 'b' -> 'a'

Supported comment types

Use comments to provide additional information about Python default variables values. You can use the following comment types:

- Single line: // comment
- Multi line: /* comment */

Note: Comments don't affect the job.

Configure lineage discovery

Enable the lineage discovery capability and use CLAIRE to build complete lineage by recommending endpoint catalog source objects to assign to reference catalog source connections.

1. Click the **Lineage Discovery** tab.
2. Select **Enable Lineage Discovery**.
3. In the **Filters** area, define one or more filter conditions to apply for lineage discovery.

To define filters, you can choose to select catalog source types, asset groups, or enter a catalog source name or search from a list of catalog sources.

- Select **Yes** to view filter options.
- From the Include/Exclude list, choose to include or exclude catalog sources for lineage discovery based on the filter parameters.
- From the filter type list, select catalog source type, catalog source name, or asset group.
- In the filter value field, select the required catalog source types, or click the Search button and select catalog sources or asset groups.

Filters can contain the asterisk wildcard to represent multiple characters or empty text.

The following image shows the filter condition options:

Examples:

- To include or exclude all Oracle catalog sources, select **Catalog Source Type** as the filter type and select `Oracle` in the filter value field.
- To include or exclude the 'Oracle_Retail' catalog source, select **Catalog Source Name** as the filter type and search for the catalog source or enter `Oracle_Retail` in the filter value field.
- To include or exclude all catalog sources with names that start with 'Oracle', select **Catalog Source Name** as the filter type and search for the catalog source or enter `Oracle*` in the filter value field.
- To include or exclude all catalog sources with names that end with 'Retail', select **Catalog Source Name** as the filter type and search for the catalog source or enter `*Retail` in the filter value field.
- To include or exclude all catalog sources with names that contain 'Ret', select **Catalog Source Name** as the filter type and search for the catalog source or enter `*Ret*` in the filter value field.
- To include or exclude all catalog sources that are part of the 'Financial Group' asset group, select **Asset Group** as the filter type and search `Financial Group` in the filter value field.

Note: You can't add more than one include or exclude filter for the same filter type.

- Optionally, to define an additional filter with an AND condition, click the **Add** icon.

For more information about lineage discovery, see *Lineage discovery* in the *Administration* help.

Configure data profiling and quality

Enable the data profiling capability to evaluate the quality of metadata extracted from the Databricks source system.

- Click the **Data Profiling and Quality** tab.
- Expand **Data Profiling** and select **Enable Data Profiling**.

Note: Ensure that you have permissions on all the staging connections that you use in your data profiling configuration. You can't run the job if you don't have permissions on the connections that you use. Select

connections that you have access to, or ask the administrator to grant the necessary permissions on the connections that you want to use.

3. In the **Connection and Runtime** area, choose the Secure Agent group where you want to run catalog source jobs.
4. Optionally, in the **Filters** area, specify additional filters in addition to metadata filters:
 - a. Select **Yes**.
 - b. From the Include/Exclude list, choose to include or exclude metadata based on the filter parameters.
 - c. From the Object type list, select **Catalog.Schema**.
 - d. Enter the path to the object as the filter value.

Examples:

- You extracted metadata from all objects from the 'Schema1' schema in the 'Catalog1' catalog and now you want to run a profile on the 'Table1' table in the schema. Select Table from the Object type list and then enter the catalog name followed by the schema name and the table name in the input field. For example, Catalog1.Schema1.Table1
- You extracted metadata from all objects from a schema and now you want to run a profile on multiple tables in the schema. Select Table from the Object type list. Click Select and then enter the catalog name, schema name and table name separated by dots. Click Add to enter additional tables.

To include or exclude multiple objects, click the **Add** icon to add filters with the OR condition.

5. In the **Parameters** area, configure the parameters.

The following table describes the parameters that you can enter:

Parameter	Description
Modes of Run	Determine the type of data that you want the data profiling task to collect. Choose one of the following options: <ul style="list-style-type: none">• Keep signatures only. Collects only aggregate information such as data types, average, standard deviation, and patterns.• Keep signatures and values. Collects both signatures and data values.
Profiling Scope	Determine whether you want to run data profiling only on the changes made to the source system since the last profile run or on the entire source system. Choose one of the following options: <ul style="list-style-type: none">• Incremental. Runs the profile only on the changed or updated metadata in the source system since the last profile.• Full. Runs the profile on the entire metadata that is extracted based on the filters applied for extraction.
Sampling Type	All Rows. Runs the profile on all rows in the metadata. Custom Query. Runs the profile on a specified percentage of rows. For example, TABLESAMPLE (BUCKET 10 OUT OF 50) , TABLESAMPLE (5 PERCENT) , LIMIT 2 , or TABLESAMPLE (5000 ROWS) .

Parameter	Description
Maximum Precision of String Fields	The maximum precision value for profiles on string data type.
Text Qualifier	The character that defines string boundaries. If you select a quote character, profiling ignores delimiters within the quotes. Select a qualifier from the list. Default is Double Quote .

- Expand **Data Quality** and select **Enable Data Quality**.

Note: You can click **Use Data Profiling Parameters** to use the same parameters as in the **Data Profiling** section.

Note: Ensure that you have permissions on all the staging and flat file connections that you use in your data quality configuration. You can't run the job if you don't have permissions on the connections that you use. Select connections that you have access to, or ask the administrator to grant the necessary permissions on the connections that you want to use.

- In the **Connection and Runtime** area, choose the Secure Agent group where you want to run catalog source jobs.
- In the **Parameters** area, configure the parameters.

The following table describes the parameters that you can enter:

Parameter	Description
Data Quality Rule Automation	Enable the option to automatically create or update rule occurrences for data elements in the catalog source. Choose one of the following options: <ul style="list-style-type: none"> • Apply on Data Elements linked with Business Dataset. Creates rule occurrences for all data elements that are linked with business data sets in the catalog source. • Apply on all Data Elements. Creates rule occurrences for all data elements in the catalog source.
Data Quality Remediation	Enable the option to specify a flat file connection to store the list of failed rows so that users can remediate poor data quality scores. Choose one of the following options: <ul style="list-style-type: none"> • No. Doesn't enable the Create Data Quality Failure Ticket option. • Yes. Shows a list of flat file connections where you write failed rows to customer-managed locations.

Parameter	Description
Data Quality Failure Ticket	<p>Specify whether you want to create data quality failure tickets for poor data quality scores based on the threshold defined for the rule occurrence in Data Governance and Catalog.</p> <p>Choose one of the following options:</p> <ul style="list-style-type: none"> • No. Doesn't automatically create data quality failure tickets when the data quality scores are poor. • Yes. Automatically creates data quality failure tickets based on the data quality threshold values you define in Data Governance and Catalog, and notifies you when a data quality score is below the threshold. <p>Note: You must configure a workflow event for the data quality failure and enable the event in Metadata Command Center.</p>
Cache Result	<p>Specify how you want to preview the rule occurrence results. Select Agent Cache if you want to generate a cache file in the runtime environment and to preview the cached results faster in subsequent data preview runs. The results are cached for seven days by default after the first run in the runtime environment. Select No Cache if you don't want to cache the preview results and view the live results.</p>
Run Rule Occurrence Frequency	<p>Specify whether you want to run data quality rules based on the frequency defined for the rule occurrence in Data Governance and Catalog.</p>
Sampling Type	<p>All Rows. Runs the data quality task on all rows in the metadata.</p> <p>Custom Query. Runs the data quality task on a specified percentage of rows. For example, TABLESAMPLE (BUCKET 10 OUT OF 50), TABLESAMPLE (5 PERCENT), LIMIT 2, or TABLESAMPLE (5000 ROWS).</p>
Maximum Precision of String Fields	<p>The maximum precision value for profiles on string data type.</p>
Text Qualifier	<p>The character that defines string boundaries. If you select a quote character, the data quality task ignores delimiters within the quotes. Select a qualifier from the list.</p> <p>Default is Double Quote.</p>

9. To enable the data observability capability, expand **Data Observability** and select **Enable Data Observability**.

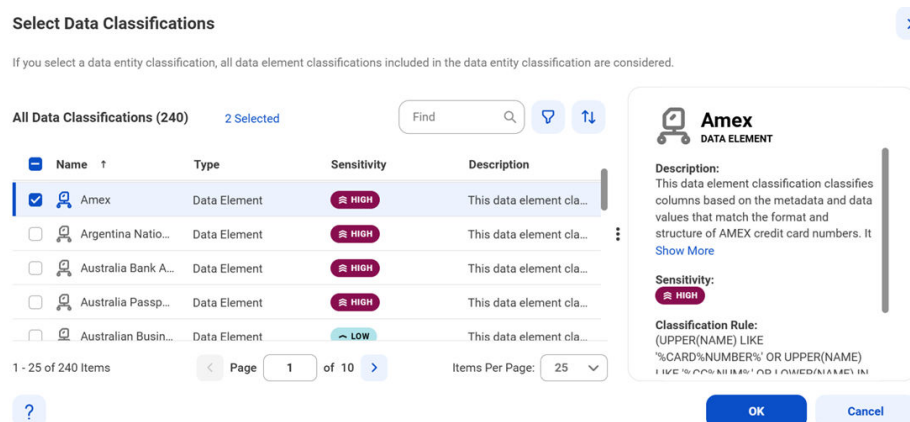
Configure data classification

Enable the data classification capability to identify and organize data into relevant categories based on the functional meaning of the data.

1. Click the **Data Classification** tab.
2. Select **Enable Data Classification**.
3. Choose one of the following options:
 - **Generated Data Classifications.** CLAIRE automatically generates data classifications for the data elements.

- **Data Classification Rules.** Choose from predefined or custom data classifications.

1. Click **Add Data Classification**. The following image shows the **Select Data Classifications** dialog box:



2. Select the data classifications that you want to use.
3. Click **OK**.

Configure relationship discovery

Enable the relationship discovery capability to identify pairs of similar columns and relationships between tables within a catalog source.

Before you configure relationship discovery, you need to import a relationship inference model. For more information about importing a relationship inference model, see [“Import a relationship inference model” on page 19](#).

1. Click the **Relationship Discovery** tab.
2. Select **Enable Relationship Discovery**.
3. In the **Column Similarity** area, select the **Relationship Inference Model**.

The relationship inference models that you imported appear in the **Relationship Inference Model** field.

4. Specify the **Confidence Score Threshold** that indicates whether the predictions made by the model for column similarity are acceptable. Specify a score from 0 to 1 to set a threshold limit. If the score is higher than the specified limit, Metadata Command Center automatically assigns a matching glossary term to the data element.

Note: A **Confidence Score Threshold** lower than 0.4 might result in a large number of false positives.

5. In the **Joinable Tables Relationship** area, specify the **Containment Score Threshold** to identify joinable table relationships within the catalog source. This score is an indicator of the data overlap between any two given columns which determines whether the tables are joinable.

A higher score means that the data in the objects is more similar and a lower score means lesser overlapping data between the two objects. A **Containment Score Threshold** lower than 0.4 might result in a large number of false positives.

Configure glossary association

Enable the glossary association capability to associate glossary terms with technical assets, or to get recommendations for glossary terms that you can manually associate with technical assets in Data Governance and Catalog.

Metadata Command Center considers all published business terms in the glossary while making recommendations to associate your technical assets.

1. Click the **Glossary Association** tab.
2. Select **Enable Glossary Association**.
3. Select **Enable auto-acceptance** to automatically accept glossary association recommendations.
4. Specify the **Confidence Score Threshold for Auto-Acceptance** to set a threshold limit based on which the glossary association capability automatically accepts the recommended glossary terms.
Note: Specify a percentage from 80 to 100. If the score is higher than the specified limit, the glossary association capability automatically assigns a matching glossary term to the data element.
5. Select **Enable Below-threshold Recommendations** to receive glossary association recommendations below the auto-acceptance threshold. If you enable auto-acceptance, you can enable below-threshold recommendations to receive glossary recommendations below the auto-acceptance threshold.
6. Specify the **Confidence Score Threshold for Recommendations** to set a threshold based on which the glossary association capability makes recommendations
If you enable auto-acceptance, specify a percentage from 80 to the selected auto-acceptance threshold. You can accept or reject the recommended glossary terms that fall within this range in Data Governance and Catalog.
If you disable auto-acceptance, specify a percentage from 80 to 100 inclusive.
7. Choose to automatically assign business names and descriptions to technical assets. You can then choose to retain existing assignments and only assign business names and descriptions to assets that don't have assignments, or allow overwrite of existing assignments.
By default, existing assignments are retained.
8. Optional. Choose to ignore specific parts of data elements when making recommendations. Select **Yes** and enter prefix and suffix keyword values as needed.
Click **Select** to enter a keyword. You can enter multiple unique prefix and suffix keywords. Keyword values are case insensitive.
9. Optional. Choose specific top-level business glossary assets to associate with technical assets. Selecting a top-level asset selects its child assets as well. Select **Top-level Glossary Assets** and specify the assets on the **Select Assets** page.
10. Optional. Choose to use abbreviations and synonym definitions from lookup tables for accurate glossary association. Select **Yes** to enable, and then click **Select** to upload a lookup table.
11. Click **Next**.
The **Associations** page appears.

Step 3. Associate stakeholders and asset groups

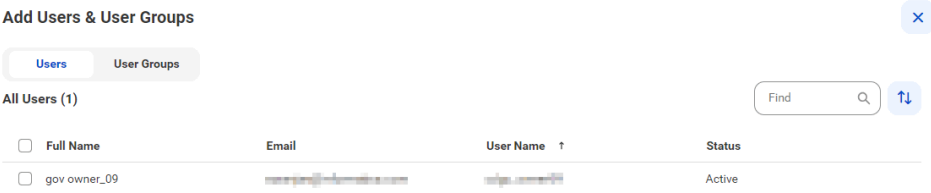
Associate users or user groups within a stakeholder role as stakeholders for technical assets in Data Governance and Catalog. Also, you can choose to assign technical assets extracted from the catalog source

to asset groups. You can then use access policies to control permissions on assets that are assigned to asset groups.

Verify that the administrator assigned users and user groups to the stakeholder role that you want to associate with technical assets.

1. To associate users or user groups as stakeholders with technical assets extracted from the catalog source, perform the following steps:
 - a. On the **Associations** page, click **Stakeholders**.
 - b. Select **Assign Stakeholders**.
 - c. Select a stakeholder role.
 - d. Click **Select** to add users and user groups from the stakeholder role as stakeholders for the technical assets.

The **Add Users & User Groups** dialog box displays a list of users and user groups assigned to the selected stakeholder role.



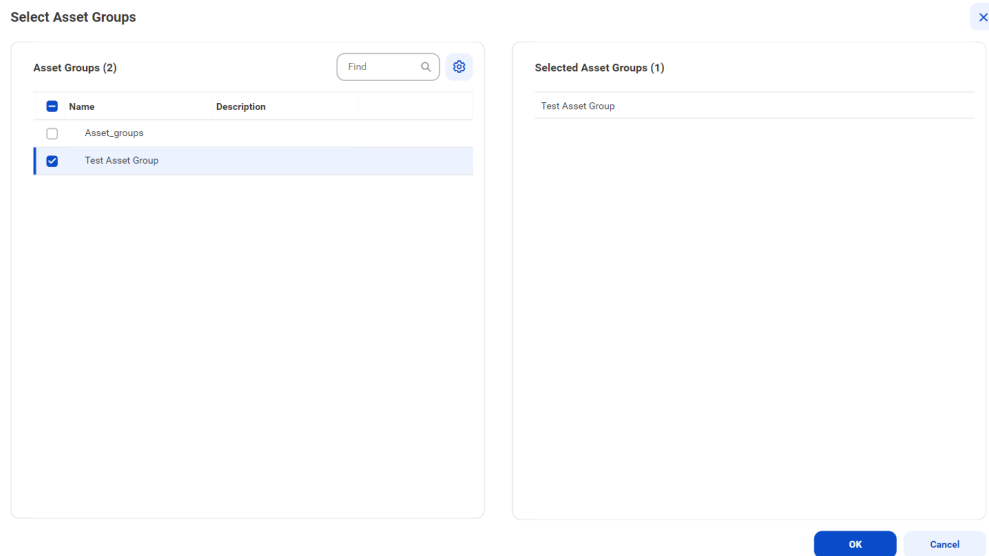
- e. Select one or more users or user groups to assign as stakeholders for the technical assets, and click **OK**.

Only the selected users and user groups belonging to the specified stakeholder role are granted the permissions to technical assets.
 - f. To assign users or user groups from another stakeholder role, click **Add** and then repeat the steps.
2. To assign asset groups to technical assets extracted from the catalog source, perform the following steps:
 - a. On the **Associations** page, click **Asset Groups**.
 - b. Select **Assign Asset Groups**.
 - c. Click **Select**.

The **Select Asset Groups** dialog box displays the list of asset groups.

If you enabled an access policy that includes an asset group, you can only view assets that belong to that asset group.

3. Select the asset groups to which you want to assign technical assets extracted from the catalog source, and click **OK**.



4. Choose to save and run the job or to schedule a recurring job.
 - To save and run the job, click **Save** and then **Run**.
 - To schedule a recurring job, click **Next** to open the **Schedule** page.

Step 4. Run or schedule the job

Choose to run a catalog source job manually, or configure it to run on schedule.

Note: You can't run multiple jobs simultaneously.

You can choose to perform a full or an incremental metadata extraction. A full metadata extraction extracts all objects from the source to the catalog. An incremental metadata extraction extracts only the changed and new objects since the last successful catalog source job run. Incremental metadata extraction doesn't remove deleted objects from the catalog and doesn't extract metadata of code-based objects if applicable.

When you run an incremental metadata extraction job with a filter to include metadata from objects, the job extracts only the objects that have the latest timestamp since the last successful job.

Note: The incremental extraction option appears if it is available for the catalog source.

Run the job manually

Click **Save** to save the catalog source and click **Run**. On the **Run Catalog Source Job** window, click **Run** to run the job.

You can override the capabilities that you selected while configuring your catalog source on the **Configuration** page. The first time you run the catalog source job, the metadata extraction capability is mandatory. From the second run onwards, you can choose to override the configured metadata change option. You can retain, delete, or deprecate objects that are deleted from the source in the catalog. For subsequent runs of the catalog source job, the metadata extraction capability is optional.

Note: You can choose incremental metadata extraction for subsequent runs only after one full metadata extraction job completes successfully. Incremental metadata extraction jobs run with the **Retain** metadata change option even if you set the option to **Delete** or **Deprecate** in the catalog source.

Note: To run a catalog source job, you need permissions on the connection to the source system. To run a catalog source job for catalog sources that reference other source systems, you need permissions on the connections for all the reference source systems.

Run the job on a schedule

You can choose to run metadata extraction and other capabilities on a recurring schedule. You can't choose incremental metadata extraction and full metadata extraction in the same schedule. To create a schedule for incremental metadata extraction, you must have completed at least one full metadata extraction job successfully. If not, first create a schedule for a full metadata extraction.

If an incremental metadata extraction is scheduled to run when the last run details aren't available, the job first performs a full metadata extraction, followed by incremental metadata extraction on subsequent runs.

For example, this can happen in the following scenarios:

- You create schedules for both incremental metadata extraction and full metadata extraction, but schedule the incremental extraction to run before the first full metadata extraction job.
- You create schedules for both incremental metadata extraction and full metadata extraction, but delete the full metadata extraction schedule before its first run.

1. On the **Schedule** tab, select **Run on Schedule**.
The **Schedule** configuration page opens.
2. Click the checkbox corresponding to each capability that you want to include in the schedule.
3. Enter the start date, time zone, and the interval at which you want to run the job.
4. You can manage additional schedules using the following options:
 - To create a new schedule, click the **Add** button.
 - To delete a schedule, click the **Delete** button.
 - To enable or disable a schedule, click the **Enable Schedule** toggle button.

Note: You can create a maximum of one schedule per capability that you enable. If you purged a catalog source or did not run the metadata extraction job, the catalog source job runs metadata extraction before running other scheduled capabilities.

Note: To create a schedule, you need permissions on the connection to the source system. If you lose permissions on the connection after you create a schedule, the scheduled jobs continue to run.

5. Click **Save** to save the schedule.

Monitor job status

After the job runs, you can monitor the status of the job on the **Overview** page of the job.

For more information about job monitoring, see *Administration*.

Step 5. Assign reference catalog source connections to endpoint catalog source objects

When you run the catalog source job, if the catalog source references another source system, a reference catalog source and connection get created that point to the reference source system. To view the complete lineage for your catalog source, you can perform connection assignment from the reference catalog source

connection to the objects in the reference source system. A reference source system might be a database, such as Databricks.

Before you assign a connection, ensure that you have created and run an endpoint catalog source for each reference source system.

Note: If the source schema contains case-sensitive tables or if the reference objects contain multiple objects with the same name in different cases, perform case-sensitive connection assignment to get correct lineage.

If you enabled the lineage discovery capability for your catalog source, you can either curate the CLAIRE recommended endpoint objects on the **Related Catalog Sources** tab or assign connections manually.

For more information about related catalog sources and lineage discovery, see *Lineage discovery* in the *Administration* help.

1. On the **Configure** page, select the **Lineage** tab, and then select the **Lineage Discovery** tab. On the **Catalog Sources** panel, select the required catalog source and click the **Assign Connections** tab.

The **Assign Connections** tab displays a list of assigned and unassigned connections along with details for each connection. Use filters to view the connections based on the connection names. Click the **Add Filter** menu to add filters.

2. Select the connection to the reference source system and click **Assign**.

The connection name appears prefixed to the reference catalog source name on the **Hierarchy** tab of your catalog source in Data Governance and Catalog.

The **Assign Connection** dialog box appears with a list of recommended objects from the endpoint catalog sources. Click **All** to view all endpoint catalog source objects.

3. In the **Assign Connections** dialog box, select one or more objects from the endpoint catalog sources and click **Assign**.

You can filter the list in the **Assign Connections** dialog box by name, type, or endpoint.

When you click **Assign**, Metadata Command Center creates links between matching objects in the connected catalog sources, and it calculates the percentage of matched and unmatched objects. The higher the percentage of matched objects, the more accurate the lineage that you view in Data Governance and Catalog.

You can connect to the following source system:

- Amazon S3
- Microsoft Azure Data Lake Storage Gen2
- Databricks
- Microsoft Azure Synapse Data Warehouse
- Oracle

To assign connections to a catalog source, the target catalog source must belong to the Database class type.

4. Run the catalog source job again. If you configured the catalog source job to run on a regular schedule, the next scheduled run picks up the updated details. If you didn't configure a schedule, run the catalog source job again to view complete lineage.

When you click **Assign**, Metadata Command Center creates links between matching objects in the connected catalog sources, and it calculates the percentage of matched and unmatched objects. The higher the percentage of matched objects, the more accurate the lineage that you view in Data Governance and Catalog.

CHAPTER 4

View results in Data Governance and Catalog

After Metadata Command Center runs a job, you can view the results in Data Governance and Catalog where the catalog source and its elements are called technical assets. You can view a catalog source as a hierarchy. Expand each technical asset to see its components.

When referenced source systems are connected to a catalog source, you can expand the hierarchy to see details about the technical asset's component elements.

You can view the data lineage of an asset contained within a catalog source to see individual elements such as data sources, calculations, and filters. When you view data lineage, you can see the individual upstream elements that contribute data or expressions to each component of a data flow or catalog source.

View metadata extraction results

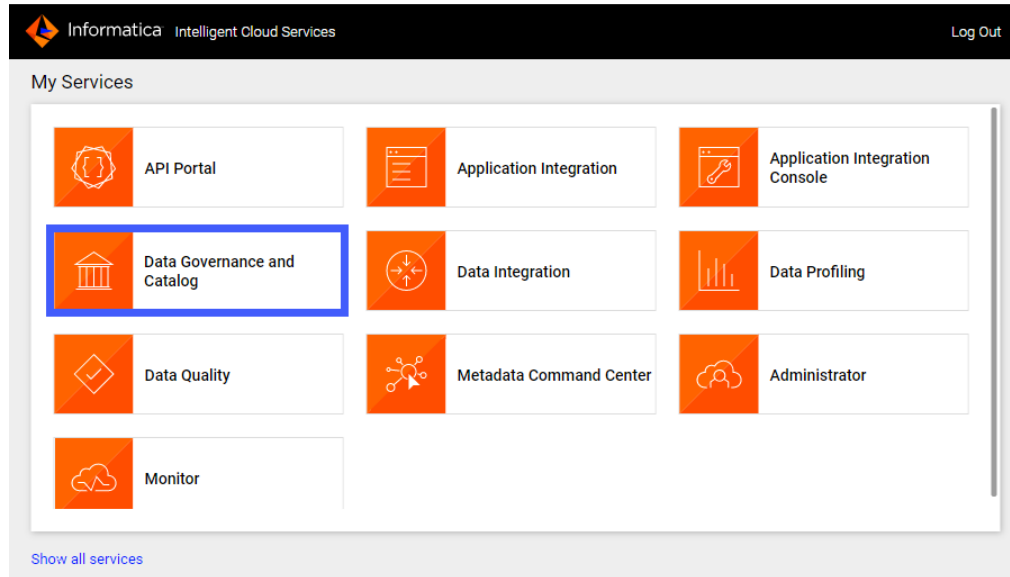
After a job runs in Metadata Command Center, view the results in Data Governance and Catalog. You can view details about source system contents in a hierarchical structure and trace data lineage.

1. Log in to Informatica Intelligent Cloud Services.

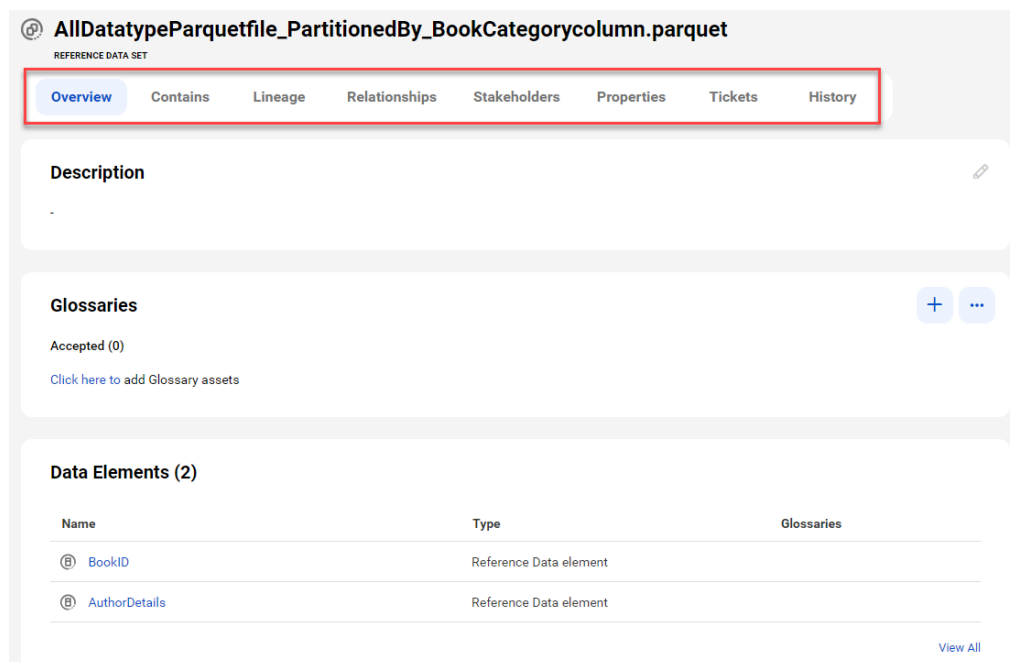
The **My Services** page appears.

2. Click Data Governance and Catalog.

The following image shows the Data Governance and Catalog box on the **My Services** page:



3. On the Data Governance and Catalog home page, click the number in the **Technical Assets** panel. The **Technical Assets** page opens.
 4. Select **Catalog Source** in the **Filter** list. The list of catalog sources opens.
 5. Search for the catalog source from which you extracted metadata, and click the name. The **Overview** tab of the asset opens.
- The following image shows a sample Databricks catalog source asset page:



6. View the asset from different perspectives by clicking on the tabs.

Note: You can view the calculation properties such as expression, control conditions, and calculation complexity on the **Overview** tab of a calculation asset.

Note: If a new record is added to the source system and you run an incremental metadata extraction job, no objects are extracted because the metadata has not changed. If an incremental metadata extraction job includes a view, the job extracts the corresponding tables even if the tables were extracted in previous runs.

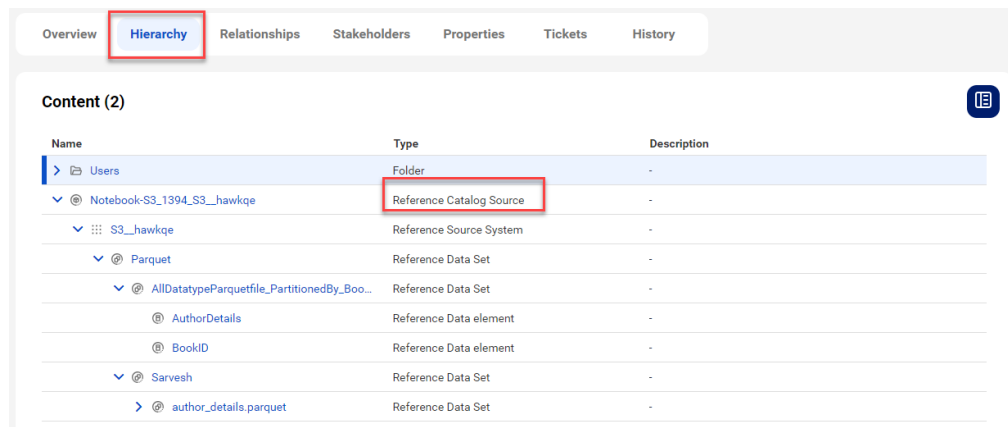
For more information about working with assets, see *Working with Assets in Data Governance and Catalog* help.

View referenced source systems

Drill down into the source systems that are referenced by catalog sources.

1. Browse to the catalog source that you want to view.
2. Click the **Hierarchy** tab to view the catalog source as a hierarchy.
3. Find an asset with the type **Reference Catalog Source**.

The following image shows an expanded view of a referenced catalog source:



4. Click to expand the referenced source to see its components.

View data lineage

Data lineage is a visual representation of the flow of data across the systems in your organization. Lineage depicts how the data flows from the system of its origin to the system of its destination.

Data lineage views are available for technical assets in the catalog source. You can view lineage at the catalog source, data set, or data element level.

The lineage at the catalog source level shows how data flows from one catalog source to another. The lineage at the data set and the data element levels show how other technical assets such as files or tables contribute to the selected asset.

If linking catalog sources is available for your catalog source, you can use Metadata Command Center to generate data lineage based on rules or by generating automated lineage with CLAIRE. You can choose source and target catalog sources and objects to link and generate lineage.

To determine whether linking catalog sources is available for your catalog source, navigate to the **Configuration** tab of the **Link Catalog Sources** page. The catalog source must appear in the list of source and target catalog sources.

For information about linking catalog sources, see *Link catalog sources* in the Administration help.

View lineage at the catalog source level

The catalog source level shows how data flows from one catalog source to another with the lineage aggregating data from the data set and data element levels.

To view data lineage at the catalog source level, open a technical asset, click the **Lineage** tab, and then verify that the level is set to **Catalog Source Level**.

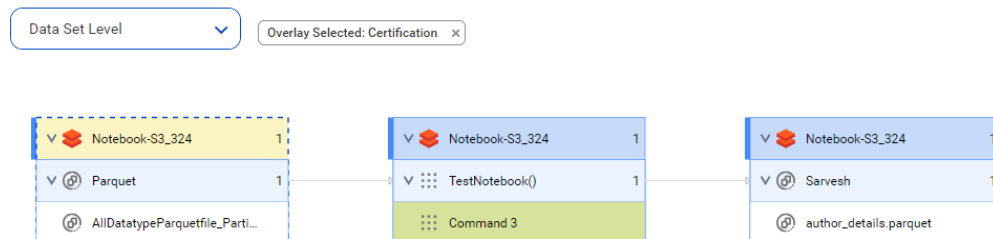
View lineage at the data set level

The data set level is a view that shows individual sets of data in the data flow.

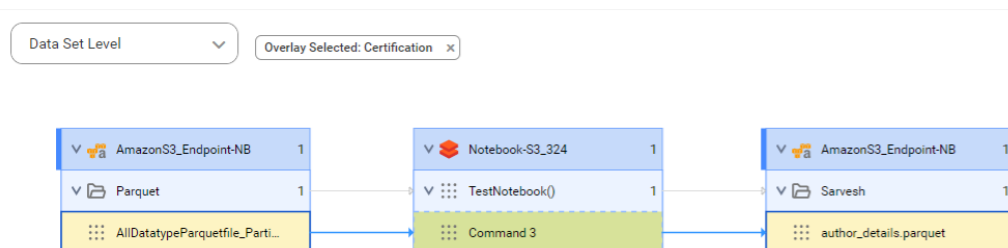
To view lineage at the data set level, open a technical asset, click the **Lineage** tab, and then verify that the level is set to **Data Set Level**.

Note: To improve wildcard lineage at the directory or file level for Databricks assets, perform connection assignment, and run the Databricks catalog source again. These wildcards can refer to files in Amazon S3 and Microsoft Azure Data Lake Storage Gen2.

The following image shows how the target `author_details.parquet` file gets data from a Parquet file stored in the Databricks source system before connection assignment:



The following image shows how the target `author_details.parquet` file gets data from a Parquet file stored in the Databricks source system after connection assignment:

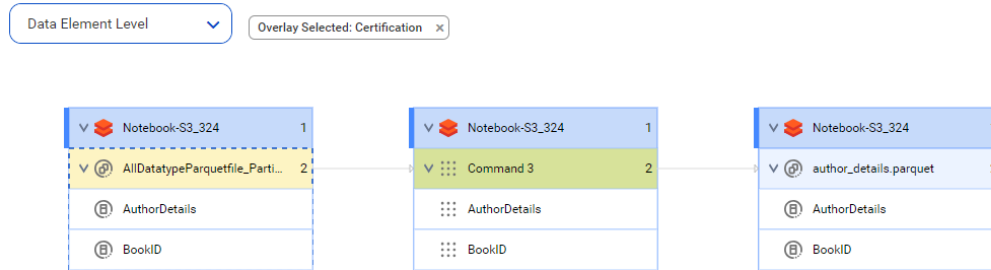


View lineage at the data element level

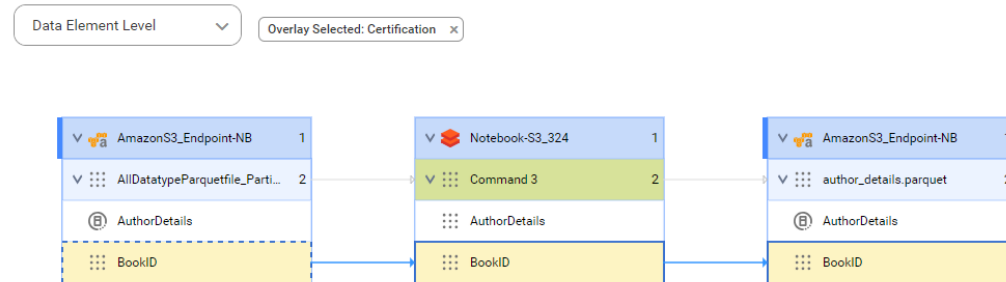
The data element level displays details of the data set level. At the data element level, you can see the input sources for expressions or commands and calculations or transformations on the data.

To view data lineage at the data element level, open a technical asset, click the **Lineage** tab, and then verify that the level is set to **Data Element Level**.

The following image shows how the target AuthorDetails column gets data from the AuthorDetails column of a Parquet file stored in Databricks source system before connection assignment:



The following image shows how the target AuthorDetails column gets data from the AuthorDetails column of a Parquet file stored in Databricks source system after connection assignment:

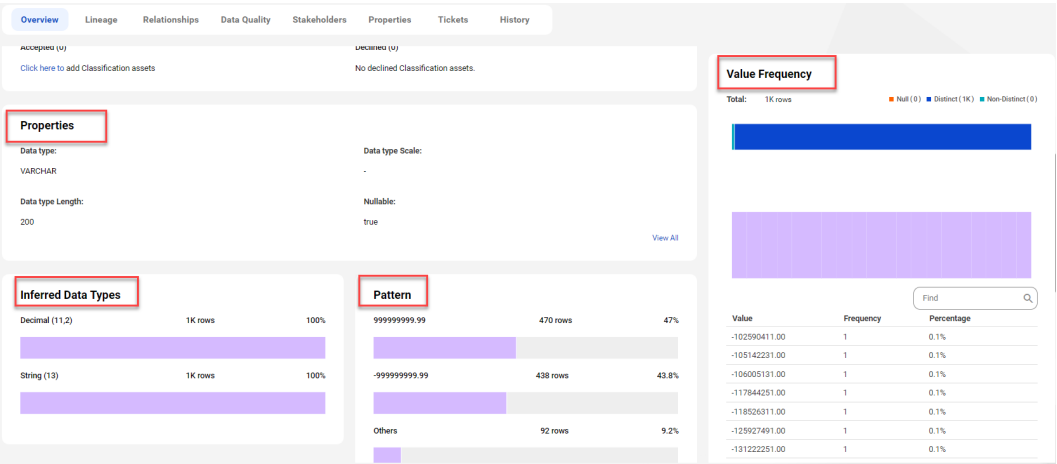


View data profiling results

When you enable the data profiling task for a catalog source in Metadata Command Center, the system runs a profile to evaluate the quality of the metadata extracted from the source system. The profiling statistics appear in Data Governance and Catalog when you open the technical assets.

The scope of profiling statistics that Data Governance and Catalog displays depends on the data profiling configuration parameters that you set when you configured the catalog source in Metadata Command Center.

The following image shows the data profiling statistics that appear on a column asset page in Data Governance and Catalog:



For more information about data profiling results, see *Asset Details* in the Data Governance and Catalog help.

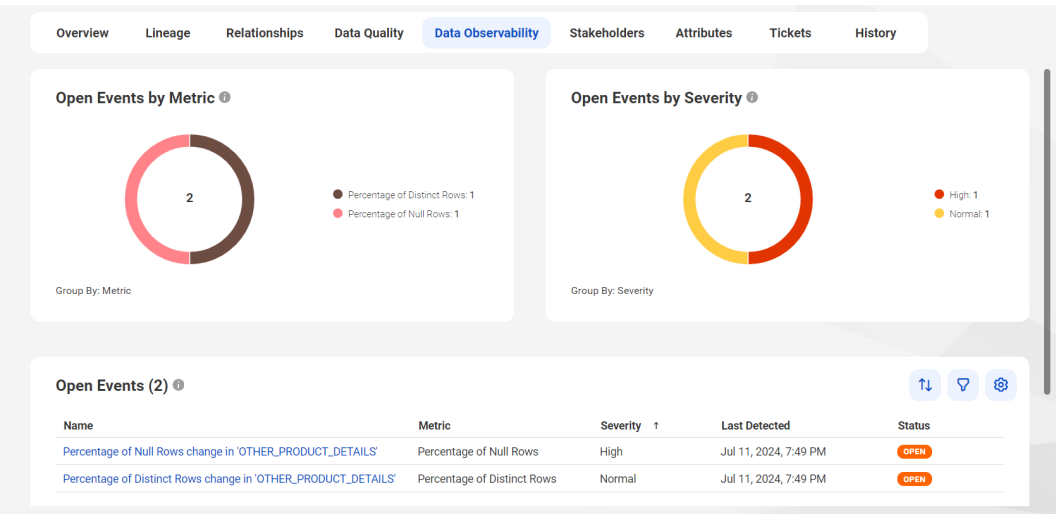
View data observability results

When you enable data observability for a catalog source in Metadata Command Center, you can view and evaluate the events that it generates in Data Governance and Catalog. These events indicate anomalies identified in the characteristics of the profiled data in your source system.

You can view the events that data observability generates for anomalies identified for catalog sources, technical data sets, and data elements. You can then take appropriate actions for the generated events.

Note: The administrator of the catalog source might have applied filters to the data to narrow down the data elements that are applicable for business users in Data Governance and Catalog. The data for which users receive anomaly notifications depend on the filters that are configured for the catalog source.

The following image shows the open events for a column asset in Data Governance and Catalog:

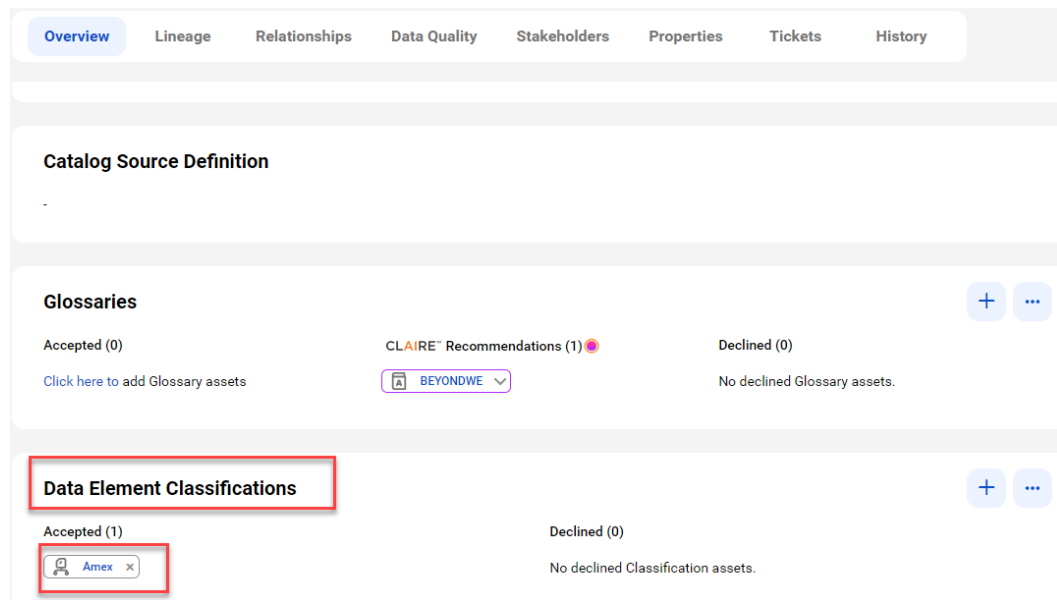


For more information about data observability results, see *Working With Assets* in the Data Governance and Catalog help.

View classified data

When you add data classification rules to a catalog source in Metadata Command Center, the system identifies the columns and tables that match the rules and displays one or more matched data classifications on the column or table asset pages in Data Governance and Catalog.

The following image shows a column asset page with the inferred data element classifications that match the column data and metadata:



For more information about data classification assets, see *Asset Details* in the Data Governance and Catalog help.

View glossary associations

When you enable the glossary association capability for a catalog source in Metadata Command Center, you can view the accepted glossary assets in Data Governance and Catalog.

The **Overview** tab for a technical asset in the catalog source displays glossary assets in the Accepted and CLAIRE Recommendations sections.

The **Glossaries** panel shows the automatically accepted and CLAIRE® recommended terms.

The following image shows a sample asset page:

OverviewLineageRelationshipsData QualityStakeholdersPropertiesTicketsHistory

Catalog Source Definition

Glossaries

Accepted (0)
[Click here to add Glossary assets](#)

CLAIRE™ Recommendations (1)

BEYONDWE

Declined (0)
No declined Glossary assets.

+...