Collibra

Collibra Data Intelligence Cloud

# Data Lineage

Version 2022.10

Collibra

# Collibra Data Intelligence Cloud - Data Lineage

Release date: October 9th, 2022

Revision date: Thu Oct 06, 2022

You can find the most up-to-date technical documentation on our Documentation Center at

https://productresources.collibra.com/docs/collibra/latest/Content/to_data-lineage.htm

# Contents

# Collibra Data Lineage

Collibra Data Lineage is a product that allows you to trace how data flows from source to destination. It consists of two components to accommodate two different personas:

- A technical lineage for Data Engineers, Data Architects and similar personas.
- A diagram with Business Summary Lineage for Business Analysts and other business users.

Technical lineage is a detailed lineage graph that shows where data objects are used and how they are transformed. A diagram with the Business Summary Lineage shows the relations between Data Assets in Data Catalog after stitching. Both map the flow of data, but a technical lineage provides a detailed overview of the data flow, while a diagram with Business Summary Lineage only provides a summary of it.

> Note Collibra Data Lineage is only a cloud-only feature.

# Technical lineage

Technical lineage is a detailed lineage graph that shows how data transforms and flows from source to destination across its entire lifecycle. It enables you to easily discover where tables and columns are used and how they relate to each other.

During the technical lineage process, relations of the type "Data Element targets / sources Data Element" are automatically created:

- Between data objects in your data source and assets from registered data sources.
- Between ingested assets from BI sources and Data Catalog assets from registered data sources.

For complete information on technical lineage, see the Collibra Data Intelligence Cloud User Guide.

# About technical lineage

Technical lineage is a detailed lineage graph that shows how data transforms and flows from source to destination across its entire lifecycle. It enables you to easily discover where tables and columns are used and how they relate to each other. You can view a technical lineage for the following asset types:

- Table
- Column
- Power BI Report
- Power BI Column
- SSRS Report
- SSRS Column
- Tableau Worksheet
- Tableau Data Attribute
- Looker Look

During the technical lineage process, relations of the type "Data Element targets / sources Data Element" are automatically created:

- Between data objects in your data source and assets from registered data sources.
- Between ingested assets from BI sources and Data Catalog assets from registered data sources.

> Tip For detailed information on how a technical lineage is created, including how the lineage harvester interacts with your data sources and the Collibra Data Lineage service, and the interaction between the Collibra Data Lineage service and Data Catalog, see the Typical workflow section, in About the lineage harvester.

## Steps to create a technical lineage

The following table shows which steps you have to take to create a technical lineage and which prerequisites you need to execute each step.

| Step | What? | Description | Prerequisites |
|---|---|---|---|
| 1 | Prepare Data Catalog physical data layer | Before you create a technical lineage, you prepare Data Catalog's physical data layer. This is necessary to automatically stitch assets in Data Catalog and the data elements in the data source for which you want to create a technical lineage.<br><br>By preparing Data Catalog's physical data layer, you create assets of the following types:<br>• System<br>• Database<br>• Schema<br>• Table<br><br>> Note If you don't prepare the Data Catalog physical data layer, you can still create a technical lineage. However, stitching will not be performed. | • You have a global role with the Catalog global permission, for example Catalog Author.<br>• You have a resource role with the following resource permissions:<br>◦ Asset: Add<br>◦ Attribute: Add<br>◦ Domain: Add<br>◦ Attachment: Add |

| Step | What? | Description | Prerequisites |
|---|---|---|---|
| 2 | Set up the lineage harvester | You use the lineage harvester to collect source code from your data sources and create new relations between data elements from your data source and existing assets into Data Catalog.<br><br>You can download the lineage harvester from the Collibra Community Downloads page. | • Java Runtime Environment version 11 or newer or OpenJDK 11 or newer.<br>• You have purchased Collibra Data Lineage.<br>• You have Collibra Data Intelligence Cloud 5.7.3 or newer.<br>• Your environment meets the hardware requirements to install and use the lineage harvester.<br>• You have added Firewall rules so that the lineage harvester can connect to:<br>◦ The host names of all databases in the lineage harvester configuration file.<br>◦ All Collibra Data Lineage service instances within your geographical location:<br>◦ 15.222.200.199 (techlin-aws-ca.-collibra.com)<br>◦ 18.198.89.106 (techlin-aws-eu.-collibra.com)<br>◦ 54.242.194.190 |

| Step | What? | Description | Prerequisites |
| --- | --- | --- | --- |
| | | | (techlin-aws-us.-collibra.com)<br>◦ 51.105.241.132 (techlin-azure-eu.-collibra.com)<br>◦ 20.102.44.39 (techlin-azure-us.-collibra.com)<br>◦ 35.197.182.41 (techlin-gcp-au.-collibra.com)<br>◦ 34.152.20.240 (techlin-gcp-ca.-collibra.com)<br>◦ 35.205.146.124 (techlin-gcp-eu.-collibra.com)<br>◦ 34.87.122.60 (techlin-gcp-sg.-collibra.com)<br>◦ 35.234.130.150 (techlin-gcp-uk.-collibra.com)<br>◦ 34.73.33.120 (techlin-gcp-us.-collibra.com) |

| Step | What? | Description | Prerequisites |
| --- | --- | --- | --- |
| | | | Note The lineage harvester connects to different instances based on your geographic location and cloud provider. If your location or cloud provider changes, the lineage harvester rescans all your data sources. You have to whitelist all Collibra Data Lineage service instances in your geographic location. In addition, we highly recommend that you always whitelist the techlin-aws-us instance as a backup, in case the lineage harvester cannot connect to other Collibra Data Lineage service instances. |

| Step | What? | Description | Prerequisites |
|---|---|---|---|
| 3 | Prepare the configuration file | You create a configuration file to determine for which data sources you want to create a technical lineage. The configuration file is used by the lineage harvester to extract information from data sources for which you want to create a technical lineage.<br><br>Tip You can use the configuration file generator to create an example configuration file with the properties of your choosing. You can easily copy this example to your configuration file and replace the values of the properties to match your data source information.<br><br>When you have created a configuration file, you can use specific commands to perform different actions on the data sources that are defined in your configuration file.<br><br>For example, you use the full-sync command to upload the source code from the data sources in the configuration file to the Collibra Data Intelligence Cloud, where they are analyzed | • You have a global role that has the Manage all resources global permission.<br>• You have a global role with the Catalog global permission, for example Catalog Author.<br>• You have the Technical lineage global permission.<br>• The lineage harvester is able to access all data sources in the configuration file.<br>• You have the necessary permissions to all database objects that the lineage harvester accesses. |

| Step | What? | Description | Prerequisites |
|---|---|---|---|
| | | and processed and where the technical lineage is created.<br><br>Tip<br>• If you want to use SQL files from a previously loaded data source, you have to download the SQL files of a data source to the lineage harvester.<br>• If you want to use a data source in an external directory, for example Informatica PowerCenter, SQL Server Integration Services or IBM InfoSphere DataStage, you have to prepare the external directory folder.<br>• If you want to use a JSON file to create a custom technical lineage, you have to prepare the JSON file. | |

| Step | What? | Description | Prerequisites |
|---|---|---|---|
| 4 | View the technical lineage. | After you created the technical lineage, you can go to a Power BI Column, Looker Look, Column or Table asset page and click the Technical lineage tab to view the technical lineage.<br><br>You can use the Browse tab pane to search for different data objects and trace their dependencies or use the Settings tab pane to edit or export the technical lineage and see the logs created by the lineage harvester. | • You have a global role with the Catalog global permission, for example Catalog Author.<br>• You have a global role with the Technical lineage global permission. |

Tip For complete information on ingesting metadata from the following BI tools and creating a technical lineage, see the dedicated sections:

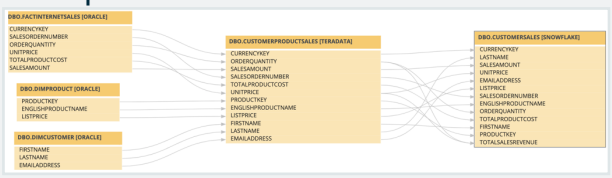
- Tableau:
  - Via the Data Catalog user interface.
  - Via the lineage harvester.
- Power BI (deprecated)
- Power BI
- Looker
- MicroStrategy
- SQL Server Reporting Services and Power BI Report Server

## Data objects

You can see two types of data objects in your technical lineage:

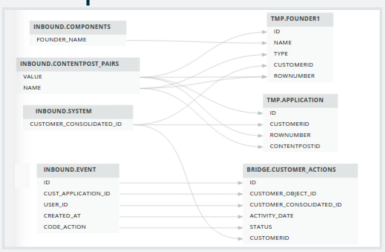
- Data objects from your data source that are stitched to assets in Data Catalog and for which you created the technical lineage. These assets have a yellow background.

Example




- Other objects, for example temporary tables and columns, that the lineage harvester collects from your data sources, but are not stitched to assets in Data Catalog. These objects have a gray background.

  Example

Note We do not support stitching for Looker or MicroStrategy assets.

## Naming convention

When you create a technical lineage, Data Catalog follows a strict naming convention for the full names of assets. Each asset has a display name and full name. You can freely edit the display name. However, you should never edit the full name, because Data Catalog needs it to refresh data sources for which you created the technical lineage and to refresh the technical lineage itself.

When you prepare the Data Catalog physical data layer and the configuration file, you should always use the full name as the name of the corresponding data object in your data source for the following assets:

- System
- Database
- Schema

> Note If you want to create a technical lineage for a Google BigQuery database, the project name in the configuration file must be the same as the full name of the Database asset.

> Warning Editing the full name of the Schema, Database and System assets may lead to errors during the technical lineage creation process.

## Transformation logic

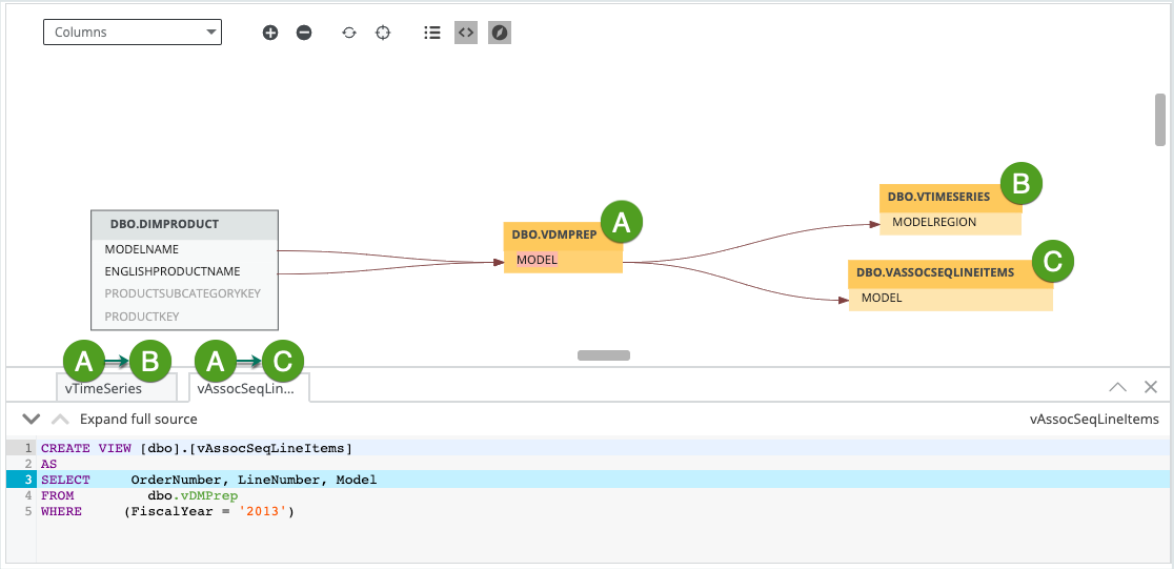
Transformation logic is used to transform source code in a technical lineage diagram that can be visualized in Data Catalog. Collibra Data Lineage supports the most commonly used transformations.

Collibra Data Lineage enables you to trace how your data flows between multiple data sources and, at the same time, see the source code of each part of your technical lineage. By following the transformations in your technical lineage, you can easily find a specific source code fragment.

Tables and columns in a technical lineage can have incoming and outgoing transformations. When you right-click on a table or column and click either Transformations (IN) or Transformations (OUT), the source code pane shows the following:

- The name of the source code fragment. On the Sources tab page, you can see the analysis log files of this source code fragment.
- If a table or column has more than one transformation, there are tabs for each source code fragment.
- The source code of the fragment. The source code that is relevant for the selected column or table is highlighted.

Example You want to see the outgoing transformations of column A to columns B and C. When you right-click column A and then click **Transformations (OUT)**, you see that there are two tabs containing source code. The first tab shows the outgoing source code from column A to column B. The second tab shows the source code from column A to column C.




## Automatic stitching for technical lineage

Stitching is a process that creates relations between assets and data objects representing the same data source. More specifically, stitching creates relations between the following assets:

- The assets that were created when you prepared Data Catalog's physical data layer for a data source; and
- The data objects in the same data source for which you created a technical lineage and that represent the assets in Data Catalog.

For Collibra Data Lineage to stitch the assets to the data objects, you must prepare the Data Catalog physical data layer to create the database > schema > table > column or system > database > schema > table > column hierarchy.

When the data sources are scanned, the Collibra Data Lineage service automatically creates and pushes new relations of the type "Data Element targets / sources Data Element":

- Between data objects in your data source and assets from registered data sources.
- Between ingested assets from BI sources and Data Catalog assets from registered data sources.

Example To clarify, in the case of Tableau integration, the Tableau Data Attribute is the target of the Column and the Column is the source of the Tableau Data Attribute.

Note If you don't prepare the Data Catalog physical data layer, Data Catalog creates a technical lineage without stitching. As a result, when you click the Technical lineage tab on any Column, Table, Tableau Data Attribute, Power BI Column or SSRS Column asset page, you get the message **The current asset doesn't have a technical lineage yet**. However, you can use the Browse tab pane to view the technical lineage of data objects in data sources for which you created the technical lineage.

## Stitching issues

To stitch assets in Data Catalog to data objects collected by the lineage harvester, the Collibra Data Lineage service looks at the full path of the assets in Data Catalog and the full path of data objects in your data source. Stitching is based on the full path of objects with the following structure: (system) > database > schema > table > column. If the full paths match, the Collibra Data Lineage automatically stitches the data objects to the existing assets in Data Catalog. To indicate this, the assets have a yellow background in the technical lineage graph.

Tip **Ingesting metadata via Edge and creating a technical lineage via the lineage harvester?**
As detailed above, stitching is based on the full path of objects with the following structure: (system) > database > schema > table > column. Toward this end, our API returns the system name for the full name of the database. If you register a data source via Edge, however, the connection name is shown in the full name of the database, not the system name. Don't worry, this does not break the stitching. The system asset that you create when you prepare the Data Catalog physical data layer is still successfully stitched, based on the system name.

If the full path of an asset in Data Catalog does not match the full path of a data object in your data source, Collibra Data Lineage cannot stitch them. To indicate this, the data objects have a gray background in your technical lineage graph. To fix stitching issues, you must check the full path of the assets in Data Catalog and make sure they match the full path of the data objects that are shown in the technical lineage graph. If you change the full path, make sure to run the lineage harvester again.

> Note We do not support stitching for Looker or MicroStrategy assets.

> Tip You can use the Stitching tab page to easily find the full path of assets in Data Catalog and data objects that were collected by the lineage harvester. The Stitching tab page also shows an overview of all assets and data objects that are stitched successfully.

# Lineage harvester versions

Collibra releases a new version of the lineage harvester every month as part of the Collibra Data Intelligence Cloud release. Check the technical lineage change log for the most important changes in each release.

| Collibra Data Intelligence Cloud version | Lineage harvester version |
|---|---|
| 2022.09 | 2022.09 |
| 2022.08 | 2022.08 |
| 2022.07 | 2022.07 |
| 2022.06 | 2022.06 |

| Collibra Data Intelligence Cloud version | Lineage harvester version |
|---|---|
| 2022.05 | 2022.05<br><br>Note Lineage harvester 2022.05 includes an internal format change to the password manager pwd.conf file. This means that if you use Lineage harvester 2022.05, you can no longer use the pwd.conf file with an older lineage harvester version. |
| 2022.04 | 2022.04 |
| 2022.03 | 2022.03 |
| 2022.02 | 2022.02 |
| 2022.01 | N/A |
| 2021.11 | 1.4.4 |
| 2021.10 | 1.4.3 |
| 2021.09 | 1.4.2 |

Important
- We highly recommend that you download and use the newest lineage harvester from the Collibra downloads page, even if you have an older version of Collibra Data Intelligence Cloud.
- Older lineage harvester versions are not supported.

# Collibra Data Lineage service

The Collibra Data Lineage service processes and analyzes the harvested metadata from supported (meta)data sources and uploads it to Data Catalog. The Collibra Data Lineage

service processes or stores only metadata, but not actual data.

When you run the lineage harvester, it firstly connects to any available Collibra Data Lineage service instance to determine your cloud provider and geographic location of your Collibra Data Intelligence Cloud environment. Then, the lineage harvester sends the harvested metadata to the Collibra Data Lineage service instance with the same cloud provider and geographic location.

Currently, your metadata can be processed on one of the following Collibra Data Lineage service instances:

| Server | IP address | DNS name |
|---|---|---|
| techlin-aws-ca | 15.222.200.199 | techlin-aws-ca.collibra.com |
| techlin-aws-eu | 18.198.89.106 | techlin-aws-eu.collibra.com |
| techlin-aws-us | 54.242.194.190 | techlin-aws-us.collibra.com |
| techlin-azure-eu | 51.105.241.132 | techlin-azure-eu.collibra.com |
| techlin-azure-us | 20.102.44.39 | techlin-azure-us.collibra.com |
| techlin-gcp-au | 35.197.182.41 | techlin-gcp-au.collibra.com |
| techlin-gcp-ca | 34.152.20.240 | techlin-gcp-ca.collibra.com |
| techlin-gcp-eu | 35.205.146.124 | techlin-gcp-eu.collibra.com |
| techlin-gcp-sg | 34.87.122.60 | techlin-gcp-sg.collibra.com |
| techlin-gcp-uk | 35.234.130.150 | techlin-gcp-uk.collibra.com |
| techlin-gcp-us | 34.73.33.120 | techlin-gcp-us.collibra.com |

> Important You have to whitelist all Collibra Data Lineage service instances in your geographic location. For example, if your data is located in Europe, you have to

whitelist the following Collibra Data Lineage service instances: techlin-aws-eu and techlin-gcp-eu. In addition, we highly recommend that you always whitelist the techlin-aws-us instances as a backup, in case the lineage harvester cannot connect to other Collibra Data Lineage service instances.

# Supported data sources for technical lineage

Collibra Data Intelligence Cloud supports many data sources and metadata sources, including JDBC data sources, ETL tools and BI tools, for which you can create a technical lineage. You use these data sources when you prepare the configuration file and Data Catalog's physical data layer.

Note Using an older version of a data source might not work as expected; however, we don't expect problems if you use a newer version.

## JDBC data sources

The following table shows the supported JDBC data sources and driver versions that have been tested. You can connect to them via a JDBC driver or by creating a folder.

| JDBC data source type | Supported versions | Connection type | Scope |
|---|---|---|---|
| Amazon Redshift | 1.2.34.1058 and newer | JDBC, Folder | SQL based input without stored procedures. |
| Azure SQL server | Newest version | JDBC, Folder | SQL based input and stored procedures. |
| Azure SQL Data Warehouse | Newest version | JDBC, Folder | SQL based input and stored procedures. |
| Azure Synapse Analytics | Newest version | JDBC, Folder | SQL based input and stored procedures. |

| JDBC data source type | Supported versions | Connection type | Scope |
|---|---|---|---|
| Google BigQuery | Newest version | JDBC, Folder | SQL based input without stored procedures. |
| Greenplum | 6.10 and newer | JDBC, Folder | SQL based input. |
| HiveQL (SQL-like statements) | 2.3.5 and newer | JDBC, Folder | SQL based input and connection via an AWS host. |
| IBM DB2 | 11.5 and newer | JDBC, Folder | SQL based input without stored procedures. |
| Oracle | 11g, 12c and newer | JDBC, Folder | SQL based input and stored procedures. |
| PostgreSQL | 9.4, 9.5 and newer | JDBC, Folder | SQL based input without stored procedures. |
| Microsoft SQL Server | 2014, 2016 and newer | JDBC, Folder | SQL based input and stored procedures. |
| MySQL | 5.7, 8 and newer | JDBC, Folder | SQL based input without stored procedures. |
| Netezza | 7.2.1.0 and newer | JDBC, Folder | SQL based input without stored procedures. |

| JDBC data source type | Supported versions | Connection type | Scope |
|---|---|---|---|
| SAP Hana | 2.00.40 and newer | JDBC, Folder | SQL based input and SAP HANA Information views, which includes attributes, analytic views and calculation views from database table or view data sources.<br><br>Script-based calculation views and stored procedures are out of scope. |
| Snowflake | Newest version | JDBC, Folder | SQL based input without stored procedures. |
| Spark SQL | 2.4.3 and newer | JDBC, Folder | SQL based input and connection via an AWS host. |
| Sybase Adaptive Server Enterprise | 16.0 SP02 and newer | JDBC, Folder | SQL based input without stored procedures. |
| Teradata | 15.0, 16.20.07.01 and newer | JDBC, Folder | SQL based input, including BTEQ scripts. |

## ETL tools

The following table shows the supported ETL tools and driver versions that have been tested. You can connect to them via an API or by creating a folder.

| ETL tool | Supported versions | Connection type | Scope |
|---|---|---|---|
| AWS Glue script annotations (beta) | N/A | Folder | Only script annotations including transformation details. |

| ETL tool | Supported versions | Connection type | Scope |
|---|---|---|---|
| IBM InfoSphere DataStage | 11.5 and newer | Folder | Commonly used DataStage ETL components including SQL overrides and transformation details.<br><br>Collibra Data Lineage supports IBM InfoSphere DataStage transformation logic.<br><br>You have to prepare a folder with all data objects that you want to process. |
| Informatica Intelligent Cloud Services, specifically Cloud Data Integration<br><br>Tip Data Integration is one of the Informatica Intelligent Cloud services. | Cloud, newest only | API | Commonly used transformations in Informatica Intelligent Cloud Services: Data Integration, including SQL overrides.<br><br>Supported data sources are locally stored flat files and databases. |
| Informatica PowerCenter | 9.6 and newer | Folder | Commonly used transformations in Informatica PowerCenter, including SQL overrides.<br><br>You have to prepare a folder with all data objects that you want to process. |

| ETL tool | Supported versions | Connection type | Scope |
|---|---|---|---|
| Matillion | Newest version | API | SQL based input without stored procedures.<br><br>The lineage harvester can only access Redshift and Snowflake projects. |
| SQL Server Integration Services (SSIS) | 2012 and newer<br><br>Package format version 6 or newer. | Folder | All commonly used transformations in SSIS, data flows and mappings, including SQL overrides.<br><br>> Important SQL statements from Excel are not supported.<br><br>You have to prepare a folder with all data objects that you want to process. |

## BI tools

The following table shows the supported BI tools.

| BI tool | Tested versions | Connection type |
|---|---|---|
| Tableau | Newest | Tableau.<br><br>You have to prepare:<br><br>• lineage harvester configuration file for Tableau ingestion.<br>• Optionally, a Tableau <source ID> configuration file. |
| Power BI (deprecated) | Newest | Existing lineage.<br><br>You have to run the Power BI harvester and the lineage harvester to ingest Power BI metadata.<br><br>> Note Integration via the Power BI harvester is deprecated. We will continue to fix issues, but the development of new features and improvements is discontinued. |
| Power BI | Newest | Power BI.<br><br>The new Power BI integration includes many enhancements, including consolidated harvesters, meaning you no longer need the Power BI harvester. You only need to prepare:<br><br>• lineage harvester configuration file for Power BI ingestion.<br>• Optionally, a Power BI <source ID> configuration file. |

| BI tool | Tested versions | Connection type |
|---|---|---|
| Looker | Newest | Looker.<br><br>Collibra Data Lineage automatically creates a technical lineage, but stitching is not available.<br><br>You have to prepare a lineage harvester configuration file for Looker ingestion. |
| SQL Server Reporting Services (SSRS) or Power BI Report Server (PBRS) | • SSRS: 2017 and newer<br>• PBRS: 2019 and newer | SSRS-PBRS.<br><br>You have to prepare:<br><br>• A lineage harvester configuration file for SSRS-PBRS ingestion.<br>• Optionally, an SSRS-PBRS <source ID> configuration file. |
| MicroStrategy | Newest | MicroStrategy<br><br>Collibra Data Lineage automatically creates a technical lineage, but stitching is not available and the technical lineage does not show the relations to columns.<br><br>You have to prepare a lineage harvester configuration file for MicroStrategy ingestion.<br><br>You can access any local or remote PostgreSQL database. The MicroStrategy Intelligence Server has an embedded PostgreSQL repository, as its default repository. For complete information on the default, embedded repository, see the MicroStrategy repository documentation. |

> Tip For complete information on ingesting metadata from the following BI tools and creating a technical lineage, see the dedicated sections:
>
> - Tableau:
>   - Via the Data Catalog user interface.
>   - Via the lineage harvester.
> - Power BI (deprecated)
> - Power BI
> - Looker
> - MicroStrategy
> - SQL Server Reporting Services and Power BI Report Server

## Custom technical lineage

You can create a custom technical lineage to include data objects from data sources that are not listed above. For more information, see Using custom technical lineage.

## Authentication

Technical lineage supports the following means of authentication:

- For all data sources, except for external directories: username and password.
- Google BigQuery data sources: username and password or a service account key file. For more information, see the Google BigQuery documentation.
- Power BI: username and password or service principal.
- Snowflake: username and password or key pair authentication.
- Tableau: username and password or token-based authentication.
- No other authentication methods are supported.

## Lineage harvester integrations available in beta

Collibra Data Intelligence Cloud supports many data sources and metadata sources, such as ETL tools or BI sources, for which you can create a technical lineage or which you can ingest.

Before Collibra releases a new lineage harvester, we test the new lineage harvester integrations extensively. However, we cannot foresee all possible use cases and scenarios. To further improve the lineage harvester, you can now test new lineage

harvester integrations in beta. After a testing period, the new lineage harvester integrations become available for all Collibra Data Lineage users

> Note Documentation is only available when the lineage harvester integrations are released. However, if you want to test new integrations, you can request testing guidelines and provide feedback.

The following table shows which integrations the lineage harvester currently supports in beta.

| Metadata source | Available in lineage harvester version | Limitations | Beta process status |
|---|---|---|---|
| AWS Glue (script annotations) | 1.4.0 and newer | The lineage harvester can process AWS Glue annotations in scripts coded in Python and Scala.<br><br>Collibra Data Lineage does not stitch the AWS Glue metadata to Amazon S3 assets created by synchronizing an S3 File system or by registering a data source using the Collibra-provided AWS Glue driver. | Open |

> Warning The lineage harvester beta integrations offer early access to new integrations. However, we can only allow a limited number of customers to test the integrations and give feedback. We will make the integrations available for all customers after processing the feedback and improving the lineage harvester.

## Testing an integration in beta

If you want to access the lineage harvester and the testing guidelines to test a lineage harvester integration in beta, do the following:

1. Create a support ticket to get access to the Technical lineage section of the Collibra Product Resources Downloads page and the testing guidelines for the lineage harvester integration.
   » You now have access to the testing guidelines.
   » You can now download and install the lineage harvester.

   > Tip If you purchased Collibra Data Lineage you already have access to the newest harvester. However, you still have to create a support ticket to access the testing guidelines.

2. Test the lineage harvester integration in beta.
3. Reach out to Collibra to provide feedback via your CSM or a support ticket.

## Supported SQL syntax

The SQL syntax used in your data sources has an impact on the technical lineage.

Technical lineage supports SQL syntax that is relevant to process data for all supported data sources. This includes:

- DML (Data Manipulation Language) statements that are used to move and transform data. For example, *INSERT*, *UPDATE* and *MERGE*.

  > Note Technical lineage supports the extraction of DML statements from supported procedures, but it does not support all SQL syntax.

- DDL (Data Definition Language) statements:
  - that impact the technical lineage. For example, *ALTER TABLE*, which you use to add or rename columns.
  - that are used to transform data. For example, *CREATE A TABLE AS SELECT*.
- Relevant syntax constructs. For example, nested subselects, aliases, different join methods, synonyms and cross-database references.

Example You want to create a technical lineage for a Teradata source that has the following SQL syntax:

- ALTER TYPE
- ALTER PROCEDURE
- CREATE/REPLACE AUTHORIZATION
- MLOAD (MultiLoad)
- RECORD (FastLoad)
- BEGIN/END QUERY LOGGING
- Functions with schema, for example schema_name.function.name(args...)
- Functions with conversation, for example function_name(args...) RETURNS VARCHAR(<number>) CHARACTER SET LATIN
- Macro argument attributes

Collibra Data Lineage will successfully parse this SQL syntax.

## Not supported SQL syntax

Technical lineage does not support the following SQL syntax:

- DML statements that you use to access data in complex structures such as JSON objects or structs.
- Triggers, foreign keys and indexes.
- Cursors, functions or dynamic queries.
- Streams queries.

Tip This is not an exhaustive list. If the SQL syntax that you use is not supported, you can add an idea in the Collibra Integrations Ideation Portal. We will evaluate the SQL syntax for inclusion.

Tip You can transform dynamic SQL statements into static ones. If the dynamic SQL can be logged at the runtime of a table, the dynamic query is transformed into a static query which can be extracted by the lineage harvester and processed without limitations.

# Supported transformation details

Collibra Data Lineage supports the most commonly used transformations in the following sources:

- Informatica PowerCenter
- Informatica Intelligent Cloud Services
- SQL Server Integration Services
- IBM DataStage (parallel job stages)

> Note The transformation is shown if the column(expression) is using at least one column from another connected transformation.

## Informatica PowerCenter transformations

The following table shows a non-exhaustive list of supported and unsupported transformations in Informatica PowerCenter.

| Supported transformations | Unsupported transformations |
|---|---|
| • Aggregator<br>• Expression<br>• Filter<br>• Joiner<br>• Lookup<br>• Mapplet<br>• Normalizer<br>• Rank<br>• Sorter<br>• Source<br>• SQL<br>• Stored Procedure<br>• Target<br>• Transaction Control | • Java<br>• Python<br>• XML |

## Informatica Intelligent Cloud Services

Collibra Data Lineage supports the following non-exhaustive list of transformations in Informatica Intelligent Cloud Services. Specifically, transformations in the Cloud Data Integration service.

- Expression
- Filter
- Joiner
- Lookup
- Mapplet
- Sequence Generator
- Source
- Target
- Union

## SQL Server Integration Services (SSIS)

Collibra Data Lineage supports the following non-exhaustive list of transformations in SQL Server Integration Services:

- Aggregate
- Cache Transform
- Conditional Split
- Data Conversion
- Derived Column
- Fuzzy Grouping
- Lookup
- Merge Join
- Multicast
- OLE DB Command
- Row Count
- Script Component
- Slowly Changing Dimension
- Sort
- Union All

Important
- Collibra Data Lineage supports SQL, but cannot parse other languages or scripts, for example SHELL and BAT scripts.
- SQL statements from Excel are not supported.
- All SQL queries must be preceded by the keyword SELECT, or else they will be skipped. Furthermore, if a comment precedes the keyword SELECT, the query will be skipped.

### IBM DataStage

Instead of transformations, IBM DataStage uses jobs with stages. IBM Datastage has three job types: parallel jobs, sequence jobs and server jobs. Collibra Data Lineage only supports the IBM DataStage stages of parallel jobs.

For a list of all job stages per job type in IBM DataStage, read the IBM documentation.

# Prepare the Data Catalog physical data layer for technical lineage

Before you can stitch data objects to the assets in Collibra Data Intelligence Cloud, if you register a data source by using a Jobserver, you must prepare the Data Catalog physical data layer to create assets and the database>schema > table > column hierarchy.

If you register a data source by using Edge, Collibra Data Intelligence Cloud creates the database > schema > table > column hierarchy. If you set the `useCollibraSystemName` property as `false` in the lineage harvester configuration file, there is no need to complete this task. If you set the `useCollibraSystemName` property as true, create a system asset.

For more information, see Automatic stitching for technical lineage and Prepare the lineage harvester configuration file.

## Prerequisites

- You have a global role with the Catalog global permission, for example Catalog Author.
- You have set up the JDBC driver of your source data, for example MySQL.

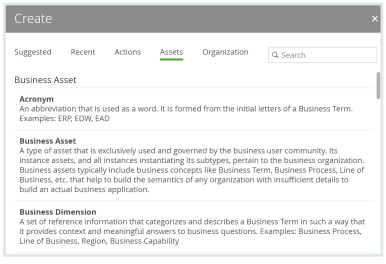
- You have registered a data source.
  If you use Jobservers in Collibra Console and there is no available Jobserver, the **Register data source** actions will be grayed out in the global create menu of Collibra Data Intelligence Cloud.
- You have a resource role with the following resource permissions on the **Schema** community if you use a Jobserver and on the **Database** community if you use Edge.
  - Asset > add
  - Attribute > add
  - Domain > add
  - Attachment > add
- You have the permissions to retrieve the metadata of the following database components through the JDBC Driver Database Metadata methods:
  - Schemas
  - Tables
  - Columns

## Steps

1. Create a System asset:

   > Tip The full name of your System asset must match the exact name of the system of the data source that you register in the configuration file.

   a. Open Catalog.
   b. In the main menu, click the **Create** (+) button.
      » The **Create** dialog box appears.
   c. Click the **Assets** tab.




   d. Click System.
      » The **Create Asset** dialog box appears.

e. Enter the required information.

| Field | Description |
|---|---|
| Type | The asset type of the asset that you are creating, in this case System. |
| Domain | The domain to which the new asset will belong. You can only create a System asset in any domain of a domain type that is assigned to a System asset type. |
| Name | The name of the System asset. This has to match the exact name of the system that you register in the configuration file as collibraSystemName .<br><br>Tip<br>You can create multiple assets in one go.<br>To do this, press `Enter` after typing a value and then type the next. Depending on the settings, asset names may have to be unique in their domain. If you type a name that already exists, it will appear in strike-through style. |

f. Click **Create**.

» A message at the top-right of your screen confirms that one or more assets are created.

2. Register a database as data source. You can register a database or an SQL directory as data source.

» After registration, the assets of the following asset types are created in Data Catalog:

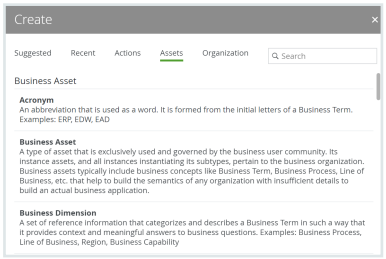
- Schema
- Table
- Column

Tip The full name of your Schema asset must match the exact name of the schema in the data source that you register in the configuration file.

3. Create a Database asset:

> Tip The full name of your Database asset must match the exact name of the database or project, in case of Google BigQuery, that you register in the configuration file.

a. Open Catalog.
b. In the main menu, click the **Create** (+) button.
   » The **Create** dialog box appears.
c. Click the **Assets** tab.




d. Click Database.
   » The **Create Asset** dialog box appears.
e. Enter the required information.

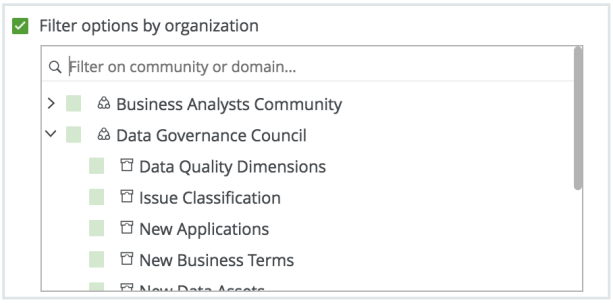
| Field | Description |
|---|---|
| Type | The asset type of the asset that you are creating, in this case Database. |
| Domain | The domain to which the new asset will belong. You can only create a Database asset in any domain of a domain type that is assigned to a Database asset type. |

| Field | Description |
| --- | --- |
| Name | The name of the Database asset. This has to match the exact name of the database that you register in the configuration file.<br><br>Tip<br>You can create multiple assets in one go.<br>To do this, press `Enter` after typing a value and then type the next. Depending on the settings, asset names may have to be unique in their domain. If you type a name that already exists, it will appear in strike-through style. |

f. Click **Create**.
   » A message at the top-right of your screen confirms that one or more assets are created.

4. Create a relation between the System asset and the Database asset using the "Technology Asset groups / is grouped by Technology Asset" relation type.
   a. In the tab pane, click **Add Characteristic**.
      » The **Add a characteristic** dialog box appears.
   b. Click **Relations**.
   c. Search for and click **groups Technology asset**.
      » The **Add groups Technology asset** dialog box appears.

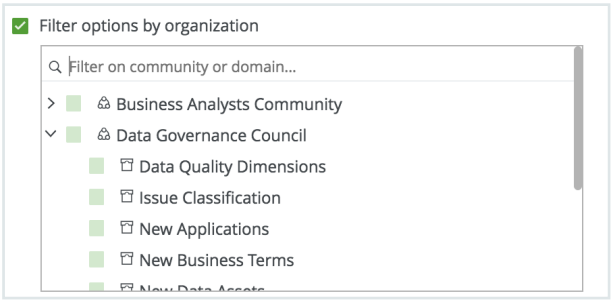
d. Enter the required information.

| Option | Description |
|---|---|
| Assets | The name of the database. |
| Filter suggested assets by organization | Option to filter the suggestions based on selected communities and domains.<br>If this option is selected, the organization tree appears. You can then filter and select domains and communities.<br>  |
| Start date | Optionally enter the date on which the relation between the assets becomes applicable. Leave this field empty to create a permanent relation. |
| End date | Optionally enter the date on which the relation between the assets is no longer applicable. Leave this field empty to create a permanent relation. |

e. Click **Save**.

5. Create a relation between the Database asset and the Schema asset using the "Technology Asset has / belongs to Schema" relation type.
   a. In the tab pane, click **Add Characteristic**.
      » The **Add a characteristic** dialog box appears.
   b. Click **Relations**.
   c. Search for and click **has schema**.
      » The **Add has schema** dialog box appears.

d. Enter the required information.

| Option | Description |
| --- | --- |
| Assets | The name of the schema. |
| Filter suggested assets by organization | Option to filter the suggestions based on selected communities and domains.<br><br>If this option is selected, the organization tree appears. You can then filter and select domains and communities.<br><br>  |
| Start date | Optionally enter the date on which the relation between the assets becomes applicable. Leave this field empty to create a permanent relation. |
| End date | Optionally enter the date on which the relation between the assets is no longer applicable. Leave this field empty to create a permanent relation. |

e. Click **Save**.

## What's next?

If you haven't created a configuration file yet, you are now required to create it.

If you created the configuration file and prepared the physical data layer, you can run the lineage harvester to start the technical lineage process.

When the technical lineage process is finished and you have the required permissions, you can go to the asset page of a Table or Column asset from the data source that you added in the configuration file and visualize the technical lineage. At the same time, new

relations of the type "Data Element targets / sources Data Element" between assets in Data Catalog are created.

The lineage harvester also uses scheduled jobs to automate the technical lineage process.

# Set up the lineage harvester

The lineage harvester is a software application that is needed to create a technical lineage and import metadata into Data Catalog.

## About the lineage harvester

You use the lineage harvester to collect source code from your data sources and create new relations between data elements from your data source and existing assets into Data Catalog.

The lineage harvester runs close to the data source and can harvest transformation logic like SQL scripts and ETL scripts from a specific location, for example a database table or a folder on a file system.

The lineage harvester connects to different Collibra Data Lineage service instances based on your geographical location and cloud provider. Ensure you have the correct system requirements before you run the lineage harvester. If your location or cloud provider changes, the lineage harvester re-harvests all your data sources.

> Note Technical lineage is created by a cloud-based service. You only connect to the cloud via an API call that is triggered by the lineage harvester.

### The lineage harvester configuration file

The lineage harvester uses a configuration file to connect to data sources, BI tools and ETL tools. The configuration file contains references to the data sources for which you want to create a technical lineage. You have to prepare the configuration file if you want to create a technical lineage and add new relations of the type "Data Element targets / sources Data Element" between existing assets in Data Catalog and "Column is target of /

is source of Data Attribute" between assets from ingested BI sources and assets in Data Catalog.

> Warning You can only use UTF-8 or ISO-8859-1 characters in all lineage harvester files.

## The lineage harvester components

The lineage harvester consists of components that harvest the metadata from the data sources specified in your configuration file and send their metadata to the Collibra Data Lineage service.

## Using the lineage harvester

Although we do not recommend it, you can use more than one lineage harvester connected to a single Collibra Data Intelligence Cloud instance, if you want to separately process data sources on different servers. In this case, all lineage harvesters must share the same configuration file. You then determine which data sources are relevant when you run the `full-sync` command. If the configuration files are not identical, then one harvester will be deleting data harvested by the other harvester.

> Note You can use different command options and arguments to perform various actions with the lineage harvester.

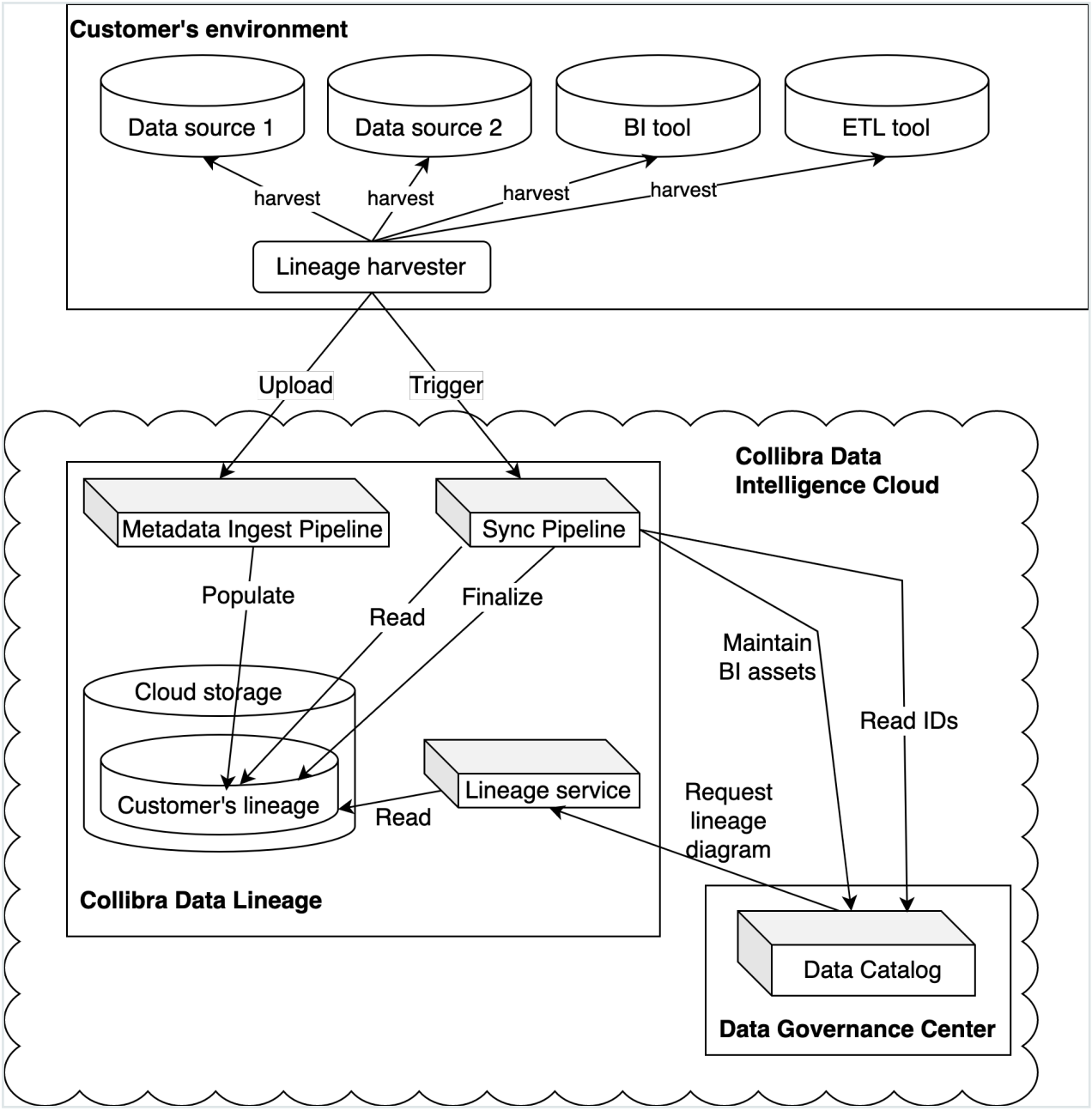
### Permissions

You need a global role with the System Administration global permission, for example Sysadmin. This role must have access to all assets in the data sources in the configuration file and be able to create new relations between these assets.

## Typical workflow

You use the lineage harvester to run the `full-sync` command. That triggers the following actions:

1. The lineage harvester:
    - Harvests the metadata from the data sources that are specified in the configuration file.
    - Uploads metadata collected from all configured data sources to Collibra Data Lineage's Metadata Ingest Pipeline.
    - Triggers the Sync Pipeline after all metadata has been completely processed.
2. The Metadata Ingest Pipeline:
    - Parses the metadata for all lineage assets and relations.
    - Stores the assets and relations in the cloud storage.
3. The Sync Pipeline:
    - Merges all partial lineages into a single data store.
    - Publishes discovered BI assets to Data Catalog.
    - Matches asset IDs from Data Catalog to the assets discovered from the metadata (stitching).
    - Stores the complete lineage in the cloud storage.
    - Publishes newly discovered relations to Data Catalog.
4. The Lineage Service:
    - Upon request, creates HTML diagrams of the lineage.
5. Data Catalog:
    - Connects to the lineage service to get the technical lineage to be shown in the technical lineage viewer.




Note The lineage harvester can only create Power BI, Tableau, Looker and other BI tool specific assets, if you included a reference to the specific BI tool in the configuration file. No other assets are created during the process. Only new relations between existing and newly created BI assets (for example between two Tableau Data Attribute assets), and between BI column and Column assets (for example between Power BI Column and Column assets) are created.

# Lineage harvester system requirements

You need to meet the system requirements to be able to install and run the lineage harvester.

## Software requirements

Java Runtime Environment version 11 or newer, or OpenJDK 11 or newer.

For Java Runtime Environment 16 or newer, or OpenJDK 16 or newer, set the `JAVA_OPTS` environment variable for the lineage harvester to function properly:

```
JAVA_OPTS='--illegal-access=deny'
```

> Note To ingest Snowflake data sources, the minimum requirement is Java Runtime Environment version 16 or newer, or OpenJDK 16 or newer.

## Hardware requirements

You need to meet the hardware requirements to install and run the lineage harvester.

### Minimum hardware requirements

You need the following minimum hardware requirements:

- 2 GB RAM
- 1 GB free disk space

### Recommended hardware requirements

The minimum requirements are most likely insufficient for production environments. We recommend the following hardware requirements:

- 4 GB RAM

  > Tip 4 GB RAM is sufficient in most cases, but more memory could be needed for larger harvesting tasks. For instructions on how to increase the maximum heap size, see Technical lineage general troubleshooting.

- 20 GB free disk space

## Network requirements

The lineage harvester uses the HTTPS protocol by default and uses port 443.

You need the following minimum network requirements:

- Firewall rules so that the lineage harvester can connect to:
  - The host names of all data sources in the lineage harvester configuration file.
  - All Collibra Data Lineage service instances in your geographic location:
    - 15.222.200.199 (techlin-aws-ca.collibra.com)
    - 18.198.89.106 (techlin-aws-eu.collibra.com)
    - 54.242.194.190 (techlin-aws-us.collibra.com)
    - 51.105.241.132 (techlin-azure-eu.collibra.com)
    - 20.102.44.39 (techlin-azure-us.collibra.com)
    - 35.197.182.41 (techlin-gcp-au.collibra.com)
    - 34.152.20.240 (techlin-gcp-ca.collibra.com)
    - 35.205.146.124 (techlin-gcp-eu.collibra.com)
    - 34.87.122.60 (techlin-gcp-sg.collibra.com)
    - 35.234.130.150 (techlin-gcp-uk.collibra.com)
    - 34.73.33.120 (techlin-gcp-us.collibra.com)

    > Note The lineage harvester connects to different Collibra Data Lineage service instances based on your geographic location and cloud provider. If your location or cloud provider changes, the lineage harvester rescans all your data sources. You have to whitelist all Collibra Data Lineage service instances in your geographic location. In addition, we highly recommend that you always whitelist the techlin-aws-us instance as a backup, in case the lineage harvester cannot connect to other Collibra Data Lineage service instances.

# Install the lineage harvester

Before you can use the lineage harvester, you need to download it and install it. You can download the lineage harvester from the Collibra Community downloads page.

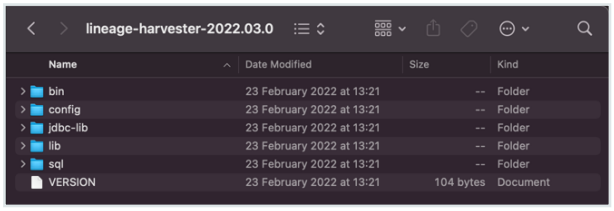
## Prerequisites

- You have purchased Collibra Data Lineage.
- You meet the minimum system requirements.
- You have added Firewall rules so that the lineage harvester can connect to:
  - The host names of all databases in the lineage harvester configuration file.
  - All Collibra Data Lineage service instances within your geographical location:
    - 15.222.200.199 (techlin-aws-ca.collibra.com)
    - 18.198.89.106 (techlin-aws-eu.collibra.com)
    - 54.242.194.190 (techlin-aws-us.collibra.com)
    - 51.105.241.132 (techlin-azure-eu.collibra.com)
    - 20.102.44.39 (techlin-azure-us.collibra.com)
    - 35.197.182.41 (techlin-gcp-au.collibra.com)
    - 34.152.20.240 (techlin-gcp-ca.collibra.com)
    - 35.205.146.124 (techlin-gcp-eu.collibra.com)
    - 34.87.122.60 (techlin-gcp-sg.collibra.com)
    - 35.234.130.150 (techlin-gcp-uk.collibra.com)
    - 34.73.33.120 (techlin-gcp-us.collibra.com)

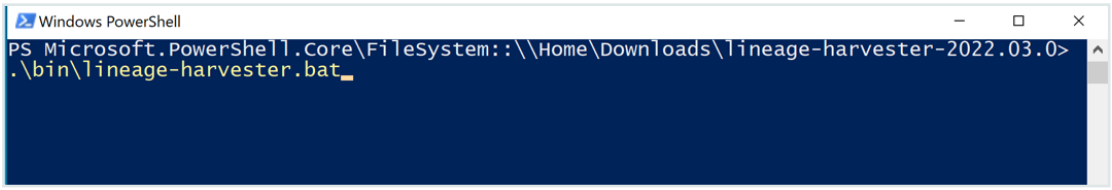
> Note The lineage harvester connects to different instances based on your geographic location and cloud provider. If your location or cloud provider changes, the lineage harvester rescans all your data sources. You have to whitelist all Collibra Data Lineage service instances in your geographic location. In addition, we highly recommend that you always whitelist the techlin-aws-us instance as a backup, in case the lineage harvester cannot connect to other Collibra Data Lineage service instances.

## Steps

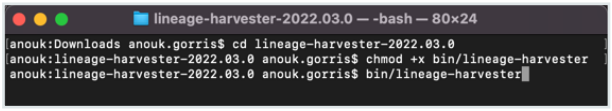
1. Download the newest lineage harvester.
2. Unzip the archive.
   » You can now access the lineage harvester folder.



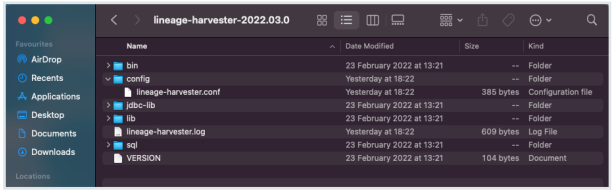

3. Run the following command line to start the lineage harvester:
   - Windows: `.\bin\lineage-harvester.bat`




   - For other operating systems: `chmod +x bin/lineage-harvester` and then `bin/lineage-harvester`




   » An empty configuration file is created in the config folder.




   » The lineage harvester is installed automatically. You can check the installation by running `./bin/lineage-harvester --help`.

## What's next?

You can now prepare the lineage harvester configuration file and run the lineage harvester again.

# Lineage harvesting app command options and arguments

After creating a configuration file, you can use the lineage harvester to perform specific actions with the data sources that are defined in your configuration file.

> Tip If you run the lineage harvester in command line, you will see an overview of possible command options and arguments that you can use. If the lineage harvester process fails, you can use the technical lineage troubleshooting guide to fix your issue.

## Typical command options and arguments

The following table shows the most commonly used command options and arguments.

> Note Although we do not recommend it, you can use more than one lineage harvester connected to a single Collibra Data Intelligence Cloud instance, if you want to separately process data sources on different servers. In this case, all lineage harvesters must share the same configuration file. You then determine which data sources are relevant when you run the `full-sync` command. If the configuration files are not identical, then one harvester will be deleting data harvested by the other harvester.

| Command | Description |
|---|---|
| `full-sync` | Uploads all of the metadata from the data sources mentioned in your configuration file to the Collibra Data Lineage service, where the metadata is then processed and uploaded to Data Catalog. |

| Command | Description |
| --- | --- |
| `-s "<ID of data source>"` | Uploads only the metadata from a specified data source. For example, `full-sync -s "myOracleDataSource"`. The specified data source must be mentioned in your configuration file.<br><br>This command allows you to process data from a newly added data source or to refresh a data source in the configuration file, without refreshing the other data sources. This reduces the time you need to upload your data sources, since you only upload specific ones without affecting the others. If you want to process multiple data sources, add `-s "ID of another data source"` per data source to the command.<br><br>> Note You can use this argument multiple times to include multiple data sources. |

| Command | Description |
| --- | --- |
| `--no-matching` | Uploads a technical lineage without stitching the data objects in your technical lineage to the corresponding Column and Table assets in Data Catalog.<br><br>> Note As a result, you won't see the technical lineage of a specific Table or Column asset, but you can still see and browse the full technical lineage. |

| Command | Description |
| --- | --- |
| `sync` | Whereas `full-sync` ingests metadata onto the Collibra Data Lineage service, processes the metadata and syncs it with assets in Data Catalog, the `sync` command only performs this last part: it syncs the metadata–as it exists on the Collibra Data Lineage service–and your assets in Data Catalog.<br><br>Tip See the following example for advice on how to use the `sync` command to add a new data source without re-harvesting all data sources.<br><br>**Example**<br>Let's say you've run `bin/lineage-harvester full-sync`, to upload from all data sources, process the metadata and sync with Data Catalog. You then decide that you want to add a new data source, but not harvest all data sources again.<br><br>1. Reference the new data source in the lineage harvester configuration file. Let's say that the new data source has the ID "MyNewSource".<br>2. Run `bin/lineage-harvester load-sources -s MyNewSource`, to load the new data source and create the ZIP file.<br>3. Run `bin/lineage-harvester` |

| Command | Description |
| --- | --- |
| | `analyze ${zip_file_from_step_2}`, to analyze the new data source on the Collibra Data Lineage service.<br>4. Run `bin/lineage-harvester sync`, to sync all of the data sources referenced in your configuration file and Data Catalog. |

| Command | Description |
| --- | --- |
| `-s "<ID of data source>"` | Syncs only the metadata on the Collibra Data Lineage service, from a specified data source. For example, `sync -s "myOracleDataSource"`. The specified data source must be mentioned in your configuration file.<br><br>This command allows you to sync data from one data source without refreshing the other data sources. You must have previously uploaded the metadata to the Collibra Data Lineage service.<br><br>Warning Only the sources you specify are synced. This means that any previously ingested metadata from non-specified sources, in Data Catalog, is deleted, along with its existing technical lineage. If this is not your intention, consider using `full-sync -s`. With `full-sync -s`, all sources are synced, regardless of which sources are specified by the `-s` command. Therefore, any previously ingested metadata from non-specified data sources remains, as do the respective technical lineages.<br><br>Note You can use this argument multiple times to include multiple data sources. |

| Command | Description |
| --- | --- |
| `load-sources` | Downloads all your data sources in a separate ZIP file, per data source, to the lineage harvester output folder. |
| `-s <ID of data source>` | Downloads only the data source with a specific ID. For example, `load-sources -s "myOracleDataSource"`.<br><br>Note You can use this argument multiple times to include multiple data sources. |
| `cat passwords.json \| ./bin/lineage-harvester <command-like-full-sync> --passwords-stdin` | Provides passwords of your Collibra Data Intelligence Cloud instance and the data sources in your configuration file to the lineage harvester without storing the passwords in the lineage harvester folder.<br><br>You can replace `cat passwords.json` by a string generated by your password manager. |
| `test-connection` | Checks the connectivity to the Collibra Data Lineage service instance and to Data Catalog. The logs will also show the IP addresses of the Collibra Data Lineage service instances that you have to whitelist.<br><br>This command is mostly used for troubleshooting purposes. |

| Command | Description |
| --- | --- |
| `--help` | Shows an overview of all supported command options and arguments that you can use in the lineage harvester. |
| `--version` | Shows the version of the lineage harvester that you are using. |
| `-Dlineage-harvester.log.dir=path/to/log/dir` | Determine the path of the log file. |

# Technical lineage password manager integration design

When you run the lineage harvester, you can either:

- Enter the passwords in the console. The passwords are then encrypted and stored in **/config/pwd.conf**.

  > Note Lineage harvester 2022.05 includes an internal format change to the password manager pwd.conf file. This means that if you use Lineage harvester 2022.05, you can no longer use the pwd.conf file with an older lineage harvester version.

- Provide the passwords via command line, in a prescribed JSON structure via stdin. This allows you to store the passwords locally in your password manager, instead of in your lineage harvester folder.

This topic provides guidance on how to structure the JSON file and which commands to use, to store the passwords locally in your password manager.

## Structure of the JSON file

If you prepare a JSON file with your passwords, you have to name the file *passwords.json*.

The JSON file must have two sections:

- The `catalogs` section defines the connection information and credentials to your Collibra Data Intelligence Cloud instance.
- The `sources` section defines the connection information and credentials to your data sources. You use the same "id" as the `id` property in the lineage harvester configuration file.

The JSON file must have the following structure:

```
{
 "catalogs": [
  {
   "url" : "<url-to-collibra-cloud>",
   "username":"<username-to-sign-in-to-collibra>",
   "password": "<password-to-sign-in-to-collibra>"
  }
 ],
 "sources": [
  {
   "id": "<id-of-your-database>",
   "username": "<database-username>",
   "password": "<database-password>"
  }
 ]
}
```

## Examples of commands

When you run the lineage harvester, you can use one of the following commands to provide the passwords:

| Passwords location | Command |
|---|---|
| a locally stored JSON file | `cat passwords.json \| ./bin/lineage-harvester full-sync --passwords-stdin` |
| a custom script, for example from a password manager | `<prepare-passwords-command> \| ./bin/lineage-harvester full-sync --passwords-stdin`<br>Note Depending on your password manager, you may need different parameters. For example, see the LastPass documentation for the parameters needed by LastPass. |

# Connecting to a proxy server

Technical lineage does not support proxy server authentication, but you can connect to a proxy server via the following commands.

## On Windows

1. Set the `-D` parameter to the `JAVA_OPTS` environment variable.

   Example
   ```
   set JAVA_OPTS=-Dhttps.proxyHost="azusquid.imf.org" -
   Dhttps.proxyPort="8080"
   ```

2. Run the lineage harvester in the same command line window: `.\bin\lineage-harvester.bat`

## On other operating systems

1. To access the hosts via a proxy server, run the following command: `bin/lineage-harvester -Dhttps.proxyHost=<Hostname or IP address of the proxy> -Dhttps.proxyPort=<port number> full-sync`

   Example  If you want to use a proxy with hostname *proxy.example.com* and port number *443*, run the following command:

   ```
   bin/lineage-harvester -Dhttps.proxyHost=proxy.example.com
   -Dhttps.proxyPort=443
   ```

2. To exclude hosts that should be accessed without going through the proxy server, add the following parameter: `-Dhttp.nonProxyHosts=<host to exclude>`. You can exclude multiple hosts by using the pipe character (|) to separate the hostnames or IP addresses to exclude. You can also use an asterisk (*) as a wildcard to match multiple hostnames or IP addresses.

Example If you want to exclude hosts with hostname *localhost* and hosts with IP address *127.0.0.1* and all IP addresses starting with *192.168**, run the following command:

```
bin/lineage-harvester -Dhttps.proxyHost=proxy.example.com
-Dhttps.proxyPort=443 -
Dhttp.nonProxyHosts=localhost|127.0.0.1|192.168*
```

Important In your configuration file, the value of the source "url" or "hostname" property (depending on the data source), and the value in your `-Dhttp.nonProxyHosts` parameter, as described above, must both be either an IP address or a host name. You will get an error if, for example, you have a host name in the "hostname" property and an IP address in the `-Dhttp.nonProxyHosts` parameter.

# Prepare the lineage harvester configuration file

Before you can visualize the technical lineage or ingest a BI source, you have to create a configuration file for the (meta)data sources that you want to process. This configuration file is used by the lineage harvester to extract data from (meta)data sources for which you want to create a technical lineage or you want to ingest.

Note

- Technical lineage only supports a limited list of (meta)data sources.
- In all lineage harvester files, you must use UTF-8 or ISO-8859-1 characters, with the exception of SQL files, which can only be UTF-8 encoded.
- Each data source has an ID property. The ID string must be unique and human readable. The ID can be anything and is only used to identify the batch of metadata that is processed on the Collibra Data Lineage service.
- The lineage harvester connects to different Collibra Data Lineage service instances based on your geographical location and cloud provider. Make sure you have the correct system requirements before you run the lineage harvester. If your location or cloud provider changes, the lineage harvester rescans all your data sources.
- Technical lineage supports the following means of authentication:
  - For all data sources, except for external directories: username and password.
  - Google BigQuery data sources: username and password or a service account key file. For more information, see the Google BigQuery documentation.
  - Power BI: username and password or service principal.
  - Snowflake: username and password or key pair authentication.
  - Tableau: username and password or token-based authentication.
  - No other authentication methods are supported.
- The lineage harvester does not support proxy server authentication, but you can manually connect to a proxy server via command line. For more information, see Connecting to a proxy server.
- Comments in the lineage harvester configuration file are not supported.
- If you upgrade to lineage harvester 1.3.0 or newer, you have to follow an upgrade procedure.

Tip For complete information on ingesting metadata from the following BI tools and creating a technical lineage, see the dedicated sections:

- Tableau:
  - Via the Data Catalog user interface.
  - Via the lineage harvester.
- Power BI (deprecated)
- Power BI
- Looker
- MicroStrategy
- SQL Server Reporting Services and Power BI Report Server

# Prerequisites

Ensure that you have completed the following tasks:

- Installed and set up the latest lineage harvester.
- Prepared the Data Catalog physical data layer for technical lineage.
- If you want to use a previously loaded data source, downloaded the SQL files of the data source to the lineage harvester.
- If you want to use an external directory, prepared a folder with data objects from the external directory.

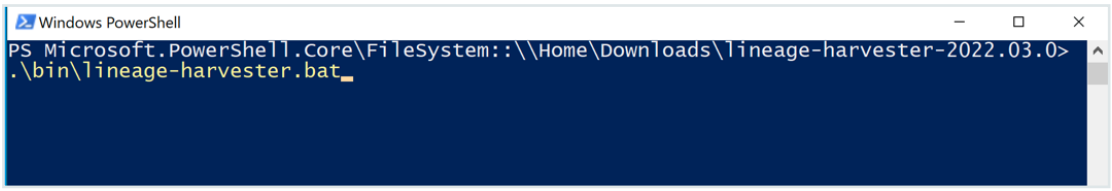
Ensure that you meet the following requirements and have the following permissions:

- Use Collibra Data Intelligence Cloud.
- A global role with the following global permissions:
  - Catalog, for example Catalog Author
  - Data Stewardship Manager
  - Manage all resources
  - System administration
  - Technical lineage
- A resource role with the following resource permission on the community level in which you created the BI Catalog domain:
  - Asset: add
  - Attribute: add
  - Domain: add
  - Attachment: add
- Necessary permissions to all database objects that the lineage harvester accesses.

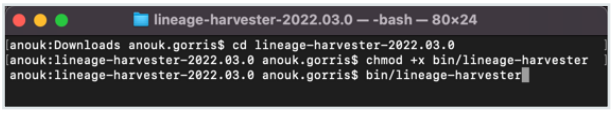
> Note For a detailed overview of the permissions that you need to access the data objects of your data sources, see the online version of this guide.

# Steps

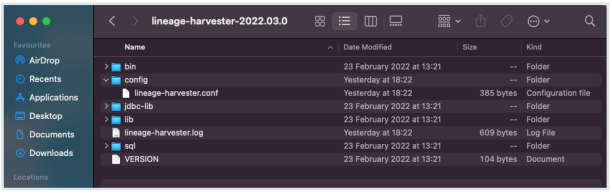
1. Run the following command line to start the lineage harvester:
   - Windows: `.\bin\lineage-harvester.bat`




   - For other operating systems: `chmod +x bin/lineage-harvester` and then `bin/lineage-harvester`




   » An empty configuration file is created in the config folder.




2. Open the configuration file and enter the values for each property.

> Tip You can use the configuration file generator to create an example configuration file with the properties of your choosing. You can easily copy this example to your configuration file and replace the values of the properties to match your data source information.

| Properties | Description |
|---|---|
| general | This section describes the connection between Collibra lineage and Data Catalog. |

| Properties | Description |
| --- | --- |
| catalog | This section contains information that is necessary to connect to Data Catalog.<br><br>> Note Versions of the lineage harvester older than 1.1.2 show `collibra` instead of `catalog`. |
| url | The URL of your Collibra environment.<br><br>> Note You can only enter the public URL of your Collibra environment. Other URLs will not be accepted. |
| username | The username that you use to sign in to Collibra. |

| Properties | Description |
| --- | --- |
| useCollibraSystemName | Indicates whether you want to use the system or server name of a data source to match to the System asset you created when you prepared the physical data layer. This is useful when you have multiple databases with the same name.<br><br>By default, the `useCollibraSystemName` property is set to `false`.<br><br>◦ If you keep the `useCollibraSystemName` property set to `false`, the lineage harvester ignores the `collibraSystemName` property in the rest of the configuration file.<br>◦ If you set the `useCollibraSystemName` property to `true`, the lineage harvester reads the value in the `collibraSystemName` property in all sections of the configuration file. It also reads the `collibraSystemName` property in the following files:<br>▪ The Informatica <source ID> configuration file<br><br>Important You must prepare a <source ID> configuration file regardless of whether the `useCollibraSystemName` property in your lineage harvester configuration files is set to `true` or `false`.<br><br>▪ The IBM DataStage or SQL Server Integration Services connection definition configuration files.<br>▪ The Informatica Intelligent Cloud Services <source ID> configuration file.<br><br>Important You must prepare a <source |

| Properties | Description |
| --- | --- |
| | ID> configuration file regardless of whether the `useCollibraSystemName` property in your lineage harvester configuration files is set to `true` or `false`.<br><br>■ The Power BI <source ID> configuration file.<br>■ The JSON files with a predefined lineage.<br><br>Note For SQL data sources, if the `useCollibraSystemName` property is:<br>◦ `false`, system or server names in table references in analyzed SQL code are ignored. This means that a table that exists in two different systems or servers is identified (either correctly or incorrectly) as a single data object, with a single asset full name.<br>◦ `true`, system or server names in table references are considered to be represented by different System assets in Data Catalog. The value of the `collibraSystemName` property is used as the default system or server name.<br><br>By default, the `useCollibraSystemName` property is set to `false`. This property is not valid for Looker integration. We recommend that you leave this property set to `false`.<br><br>Indicates whether or not you want to use the system or server name of a data source to match to the System asset in Data Catalog during automatic stitching. This is useful when you have multiple databases with the same name or if you want to |

| Properties | Description |
| --- | --- |
| | specify the Power BI workspaces from which you want to ingest.<br><br>By default, the useCollibraSystemName property is set to `False`. If you want to use it, set it to `True`.<br><br>Important<br>◦ If you set this property to `true`:<br>▪ You must provide a Power BI <source ID> configuration file that defines the system name of databases in Power BI.<br>▪ The lineage harvester reads the value of the `collibraSystemName` property in the <source-ID> configuration file.<br>◦ If you set the `useCollibraSystemName` property to `false`, the lineage harvester ignores the `collibraSystemName` property in the <source-ID> configuration file.<br><br>Indication whether you want to use the system or server name of a data source to match to the System asset you created when you prepared the physical data layer. This is useful when you have multiple databases with the same name.<br><br>By default, the `useCollibraSystemName` property is set to `false`. If you want to use it, set it to `true`.<br><br>Important<br>◦ If you set this property to `true`:<br>▪ You must provide a Tableau <source ID> configuration file that defines the system name of databases in Tableau.<br>▪ The lineage harvester reads the value of the `collibraSystemName` property in the <source-ID> configuration file. |

| Properties | Description |
| --- | --- |
| | ◦ If you set the `useCollibraSystemName` property to `false`, the lineage harvester ignores the `collibraSystemName` property in the <source-ID> configuration file.<br><br>Note If you set the `useCollibraSystemName` property to `true`, but you don't define the system name in the Tableau <source ID> configuration file, the system name in the technical lineage is DEFAULT.<br><br>Indication whether you want to use the system or server name of a data source to match to the System asset you created when you prepared the physical data layer. This is useful when you have multiple databases with the same name.<br><br>By default, the `useCollibraSystemName` property is set to `false`. If you want to use it, set it to `true`.<br><br>Important<br>◦ If you set this property to `true`:<br>▪ You must provide a SSRS-PBRS <source ID> configuration file that defines the system name of one or more databases.<br>▪ The lineage harvester reads the value of the `collibraSystemName` property in the <source-ID> configuration file.<br>◦ If you set the `useCollibraSystemName` property to `false`, the lineage harvester ignores the `collibraSystemName` property in the <source-ID> configuration file. |

| Properties | Description |
|---|---|
| | By default, the `useCollibraSystemName` property is set to `false`. This property is not valid for MicroStrategy integration. We recommend that you leave this property set to `false`.<br><br>Indicates whether or not you intend to use a Matillion <source ID> configuration file to specify the system name of a data source. This is useful if you have multiple databases with the same name, or if you want to group a number of databases under one system.<br><br>By default, the `useCollibraSystemName` property is set to `false`.<br><br>If you set this property to `true`, you must prepare a Matillion <source ID> configuration file. |
| sources | This section describes the data sources for which you want to create the technical lineage. You have to create a configuration section for each data source.<br><br>> Note You can add multiple data sources to the same configuration file. |
| <SQL directory properties> | This configuration section contains the required information of one individual SQL directory with connection type "Folder". |
| id | The unique ID of the data source. For example, `my_first_data_source`. |
| type | The kind of data source. In this case, the value has to be `SqlDirectory`. |

| Properties | Description |
| --- | --- |
| path | The full path to the SQL directory. |
| mask | The pattern of the file names in the directory. By default, this is `*`. |
| recursive | Indication of the files you want to harvest:<br>◦ `false` (default): Only harvest the files in directly under the folder in the SQL directory path.<br>◦ `true`: Harvest all files under the folder in the SQL directory path and subdirectories. |

| Properties | Description |
| --- | --- |
| dialect | The dialect of the database.<br>Tip<br>You can enter one of the following values:<br>◦ *azure*, for an Azure SQL Server data source.<br>◦ *bigquery*, for a Google BigQuery data source.<br>◦ *db2*, for an IBM DB2 data source.<br>◦ *hana*, for a SAP Hana data source.<br>◦ *hana-cviews*, for SAP Hana data calculation views.<br>◦ *hive*, for a HiveQL data source.<br>◦ *greenplum*, for a Greenplum data source.<br>◦ *mssql*, for a Microsoft SQL Server data source.<br>◦ *mysql*, for a MySQL data source.<br>◦ *netezza*, for a Netezza data source.<br>◦ *oracle*, for an Oracle data source.<br>◦ *postgres*, for a PostgreSQL data source.<br>◦ *redshift*, for an Amazon Redshift data source.<br>◦ *snowflake*, for a Snowflake data source.<br>◦ *spark*, for a Spark SQL data source.<br>◦ *sybase*, for a Sybase data source.<br>◦ *teradata*, for a Teradata data source.<br>If you want to use a Spark SQL data source, make sure that you have an AWS host. |

| Properties | Description |
| --- | --- |
| database | The name of your database, which is the full name of your Database asset.<br><br>Note You have to use the same database name as the full name of the Database asset that you create when you prepare the physical data layer in Data Catalog.<br><br>Important<br>HiveQL, MySQL and Teradata data sources don't have schemas. Therefore, HiveQL, MySQL and Teradata databases are stored in Data Catalog and technical lineage as Schema assets. The technical lineage Browse tab pane shows the following names:<br>◦ For HiveQL and Teradata:<br>▪ The database name is the name that you enter for the `collibraSystemName` property.<br>▪ The schema name is the name that you enter for the `database` property.<br>◦ For MySQL:<br>▪ The database name is the name that you enter for the `database` property. |

| Properties | Description |
| --- | --- |
| externalDbName | This property can be considered a means of database mapping, to help preserve stitching. It is relevant only for HiveQL, MySQL and Teradata data sources, specifically because they are database-less data sources.<br><br>You can add the key/value pair to the configuration file as follows: `"externalDbName": "CDATA"`<br><br>**See an example**<br>Let's say you ingest a HiveQL data source via Edge. Note that Edge gives the name "CDATA" for the database. The full path to a column is something like:<br><br>`Hive_123` (system) > `CDATA` (database) > `Hive_ABC` (schema) > `Table` > `Column`<br><br>Now, because HiveQL is database-less, the value that you give for the `database` property in your configuration file is used as the schema name in the technical lineage, and the value you give for `collibraSystemName` is used as the database name. But if `useCollibraSystemName` is set to `true`, then the value of `collibraSystemName` is also used as the system name. In that case, in the full path to the column, the system name and the database name are the same:<br><br>`Hive_123` (system) > `Hive_123` (database) > `Hive_ABC` (schema) > `Table` > `Column`<br><br>Notice the mismatch between the database names.<br><br>The `externalDbName` property tells the lineage harvester to use the value that you specify here for the database name in the technical lineage, |

| Properties | Description |
| --- | --- |
| | specifically "CDATA". This ensures that the full paths match and stitching is preserved. |
| collibraSystemName | The name of the data source's system or server. This is also the full name of your System asset in Data Catalog.<br><br>You must use the same system name as the full name of the System asset that you create when you prepare the physical data layer in Data Catalog. If you don't prepare the physical data layer, Collibra Data Lineage cannot stitch the data objects in your technical lineage to the assets in Data Catalog. |
| schema | The name of the default schema, if not specified in the data source itself. This corresponds to name of your Schema asset.<br><br>> Note You must use the same schema name as the name of the Schema asset that you create when you prepare the physical data layer in Data Catalog. |
| verbose | Indication whether you want to enable verbose logging.<br><br>By default this is set to `True`. If you don't want to use verbose logging, set it to `False`. |

| Properties | Description |
| --- | --- |
| deleteRawMetadataAfterProcessing | The lineage harvester harvests metadata from specified data sources and uploads it in a ZIP file to a Collibra Data Lineage service instance, for processing.<br><br>You can use this optional property to specify whether or not the metadata should be deleted after it has been processed.<br><br>If the property is set to `true`, the metadata is deleted after processing. If set to `false`, the metadata is stored in an Amazon S3 bucket. |
| <External directories> | This configuration section contains the required information to connect to the following data sources:<br><br>◦ Informatica PowerCenter<br>◦ SQL Server Integration Services (SSIS).<br>◦ IBM InsfoSphere DataStage<br><br>Note Make sure that you have prepared a local folder with the Informatica objects, SSIS files or DataStage files for which you want to create a technical lineage. |
| collibraSystemName | The name of the data source's system or server. If the `useCollibraSystemName` property is set to `true`, you must prepare a configuration file to provide the system information. |
| id | The unique ID of your data source. For example, *my_informatica*. |

| Properties | Description |
|---|---|
| type | The kind of data source. In this case, the value has to be *ExternalDirectory*. |
| dirType | The type of external directory. The value has to be one of the following:<br>◦ `infa`, for an Informatica PowerCenter data source.<br>◦ `ssis`, for a SQL Server Integration Service data source.<br>◦ `datastage`, for a IBM InfoSphere DataStage source. |
| path | The full path to the folder where you stored the data source. |
| mask | The pattern of the file names in the directory. By default, this is `*`. |
| recursive | Indication whether you want to use recursive queries.<br>By default, this is set to `False`. If you want to use recursive query, set it to `True`. |
| deleteRawMetadataAfterProcessing | The lineage harvester harvests metadata from specified data sources and uploads it in a ZIP file to a Collibra Data Lineage service instance, for processing.<br>You can use this optional property to specify whether or not the metadata should be deleted after it has been processed.<br>If the property is set to `true`, the metadata is deleted after processing. If set to `false`, the metadata is stored in an Amazon S3 bucket. |

| Properties | Description |
| --- | --- |
| <Informatica Intelligent Cloud Services Data Integration> | This configuration section contains the required information to enable the lineage harvester to collect and process Data Integration objects.<br><br>You can create different Informatica Intelligent Cloud Services <source ID> configuration files for a large data source to avoid errors that might occur when the lineage harvester ingests metadata from one source with a large size. You can then decrease the size of the source by separating the projects to a different source with a different <source ID> configuration file name.<br><br>**Show me the example**<br><br>`"sources" : [`<br>`{`<br>`"type" : "IICS",`<br>`"id" : "iics_source-1",`<br>`"collibraSystemName" : "iics-devel-opment",`<br>`"loginUrl" : "https://dm-us.informaticaintelligentcloud.com",`<br>`"username" : "login-iics"`<br>`"objects" : [`<br>`{`<br>`"path" : "Default/Sales",`<br>`"type" : "Project"`<br>`},`<br>`{`<br>`"path" : "My Pro-ject/Statistics",`<br>`"type" : "Project"`<br>`}`<br>`]`<br>`}`<br>`{`<br>`"type" : "IICS",`<br>`"id" : "iics_source-2",`<br>`"collibraSystemName" : "iics-devel-opment",`<br>`"loginUrl" : "https://dm-` |

| Properties | Description |
|---|---|
| | <pre>us.informaticaintelligentcloud.com",<br> "username" : "login-iics"<br> "objects" : [<br>{<br>"path" : "Finance/Task_Flows",<br>"type" : "Folder"<br>},<br>{<br>"path" : "Common/Task_Flows/tf_Cal-<br>endarDimension",<br>"type" : "Taskflow"<br>}<br> ]<br>} ]</pre><br>**Tip** Make sure you have READ permission on all data objects that you want to harvest. |
| type | The kind of data source. In this case, the value has to be `IICS`. |
| id | The unique ID that is used to identify the data source on the Collibra Data Lineage service. For example, `my_data_integration`. |
| collibraSystemName | The name of the Informatica server or system.<br><br>**Important** You must prepare a <source ID> configuration file to provide this system information. This is true regardless of whether the `useCollibraSystemName` property is set to *true* or *false*. |

| Properties | Description |
|---|---|
| loginURL | The URL of the Informatica Intelligent Cloud Services environment sign-in page. For example: `https://dm-us.informaticaintelligentcloud.com`. |
| username | The username you use to sign in to Informatica Intelligent Cloud Services. |
| objects | The objects that you want to export. Each object requires a path and a type, for example:<br>`"objects": [ { "path" : "Sales", "type" : "Project" }, { "path" : "Finance/Task_Flows", "type" : "Folder" }, { "path" : "Common/Task_Flows/tf_CalendarDimension", "type" : "Taskflow" } ]`<br>The following section provides information to identify and access Data Integration objects.<br>Tip For more information about the objects that you can export and the required information, see the Informatica documentation. |
| path | The full path to the object. |

| Properties | Description |
| --- | --- |
| type | The type of the object. For example, Taskflow.<br><br>IICS scanner's starting point is a Taskflow. Therefore the only meaningful types to export are: Taskflow, Project and Folder.<br><br>> Note The types are not case sensitive. |

| Properties | Description |
| --- | --- |
| paramFiles | The full path to the directory in which your parameter files are stored.<br><br>This is an optional parameter that allows you to harvest parameter files in Informatica Intelligent Cloud Services data sources.<br><br>Important The hierarchy of the files in the directory must be an exact match of the hierarchy of the files in your file system.<br>**Show me how to do this**<br>a. Create a directory for your parameter files. For this example, let's name the directory *my-parameter-files*.<br>b. In your lineage harvester configuration file, the value of the `paramFiles` property needs to be the full path to your parameter files directory, for example `/full/path/<my-parameter-files>/`.<br>c. Copy your parameter files to your parameter files directory.<br>Be sure to preserve the full path for each of your parameter files. For example, for parameter file */root/child/child2/paramfile.txt*, run the following commands:<br>i. `cd /full/path/<my-parameter-files>/`<br>ii. `mkdir -p root/child/child2/`<br>iii. `cp /root/child/child2/paramfile.txt root/child/child2/` |

| Properties | Description |
| --- | --- |
| deleteRawMetadataAfterProcessing | The lineage harvester harvests metadata from specified data sources and uploads it in a ZIP file to a Collibra Data Lineage service instance, for processing.<br><br>You can use this optional property to specify whether or not the metadata should be deleted after it has been processed.<br><br>If the property is set to `true`, the metadata is deleted after processing. If set to `false`, the metadata is stored in an Amazon S3 bucket. |
| <Matillion> | This section contains the required information for Matillion.<br><br>Tip When you create a new project in Matillion, you define in which group you want to create the project, the project name and the environment name. This information is needed to enable the lineage harvester to access Matillion and scan your metadata.<br><br>Important Currently, you can only create a technical lineage for Snowflake and Redshift projects in Matillion. |
| id | The unique ID that is used to identify the data source on the Collibra Data Lineage service instance. For example, `my_matillion_data_integration`. |
| type | The kind of data source. In this case, the value has to be `Matillion`. |

| Properties | Description |
| --- | --- |
| url | The URL of your Matillion environment. For example, `https://<domain name>` or `https://<IP address>`. |
| groupName | The name of your group in Matillion. |
| projectName | The name of your project in Matillion.<br><br>You can only add the name of one project. If you want to create a technical lineage for other projects within the same group, create a new section in the lineage harvester configuration file. |
| environmentName | The name of your environment in Matillion.<br><br>You can only add the name of one environment. If you want to create a technical lineage for other environments within the same project, create a new section in the lineage harvester configuration file. |
| dialect | The dialect of the database.<br><br>You can enter one of the following values:<br><br>◦ `redshift`, for an Amazon Redshift data source.<br>◦ `snowflake`, for a Snowflake data source. |

| Properties | Description |
| --- | --- |
| startTimestamp | The timestamp of tasks in Matillion. You can use this parameter to limit the amount of metadata that the lineage harvester scans.<br><br>If the `startTimestamp` field remains empty or is deleted from the configuration file, all accessible tasks are scanned.<br><br>Matillion automatically removes entries older than seven days. |
| collibraSystemName | Regardless of the value set for the `useCollibraSystemName` property, the following is true:<br>◦ You must include this property in your configuration file.<br>◦ You can leave this property empty.<br>◦ Any value that you give is ignored.<br><br>If the `useCollibraSystemName` property is set to `true`, you must prepare a Matillion <source-ID> configuration file. In that case, the `CollibraSystemName` property in the <source ID> configuration file is taken into account.<br><br>> Note This is a legacy property that will be deprecated in a future release. |
| auth | The section contains the authentication details for signing in to Matillion. |

| Properties | Description |
| --- | --- |
| type | The authentication method you want to use to sign in to Matillion.<br><br>The value must be either:<br>◦ `Basic`, for username and password authentication.<br>◦ `Token`, for token-based authentication.<br><br>Important These values are case-sensitive. |
| username | The username that you use to sign in to Matillion.<br><br>Important This property is only required if you are using the username and password authentication method. If you are using token-based authentication, do not include this property. |
| deleteRawMetadataAfterProcessing | The lineage harvester harvests metadata from specified data sources and uploads it in a ZIP file to a Collibra Data Lineage service instance, for processing.<br><br>You can use this optional property to specify whether or not the metadata should be deleted after it has been processed.<br><br>If the property is set to `true`, the metadata is deleted after processing. If set to `false`, the metadata is stored in an Amazon S3 bucket. |

| Properties | Description |
|---|---|
| <Custom lineage> | This section contains the required information to retrieve a custom lineage. To create a custom technical lineage, use this property to locate the JSON file that defines the custom technical lineage.<br><br>The JSON file must be named **lineage.json**. Ensure that you have prepared a local folder with the **lineage.json** file.<br><br>> Note In the local folder that you need to create, you can only have one JSON file. You can, however, add other files in the harvested directory and subdirectories and refer to those files from within the JSON file. |
| collibraSystemName | The system or server name of the data source. You can use this property to distinguish data objects with the same name. |
| id | The unique ID of your custom technical lineage, for example, `MyCustomLineage`. |
| type | The kind of data source. The value must be `ExternalDirectory`. |
| dirType | The type of external directory. The value is `custom-lineage`. |
| path | The full path to the folder of the JSON file that contains the custom technical lineage definition. There must be only one JSON file that defines the lineage, and the JSON file must be named **lineage.json**. |

| Properties | Description |
| --- | --- |
| deleteRawMetadataAfterProcessing | The lineage harvester harvests metadata from specified data sources and uploads it in a ZIP file to a Collibra Data Lineage service instance, for processing.<br><br>You can use this optional property to specify whether or not the metadata should be deleted after it has been processed.<br><br>If the property is set to `true`, the metadata is deleted after processing. If set to `false`, the metadata is stored in an Amazon S3 bucket. |
| <database properties> | This configuration section contains the required information of one individual data source with connection type "JDBC". |
| id | The unique ID of your data source. For example, `my_second_data_source`. |
| type | The kind of data source. In this case, the value has to be `Database`. |
| username | The username that you use to sign in to your data source. |

| Properties | Description |
| --- | --- |
| dialect | The dialect of the database.<br><br>Tip<br>You can enter one of the following values:<br>◦ *azure*, for an Azure SQL Server data source.<br>◦ *db2*, for an IBM DB2 data source.<br>◦ *hana*, for a SAP Hana data source.<br>◦ *hana-cviews*, for SAP Hana data calculation views.<br>◦ *hive*, for a HiveQL data source.<br>◦ *greenplum*, for a Greenplum data source.<br>◦ *mssql*, for a Microsoft SQL Server data source.<br>◦ *mysql*, for a MySQL data source.<br>◦ *netezza*, for a Netezza data source.<br>◦ *oracle*, for an Oracle data source.<br>◦ *postgres*, for a PostgreSQL data source.<br>◦ *redshift*, for an Amazon Redshift data source.<br>◦ *spark*, for a Spark SQL data source.<br>◦ *sybase*, for a Sybase data source.<br>◦ *teradata*, for a Teradata data source.<br>If you want to use a Spark SQL data source, make sure that you have an AWS host. |

| Properties | Description |
| --- | --- |
| databaseNames | The names or IDs of your databases.<br><br>Enter the database names of your data source between double quotes ("") and put everything between square brackets. If you want to include more than one database, separate them by a comma. For example, ["`MyFirstDatabase`", "`MySecondDatabase`"].<br><br>Note For data sources other than Oracle, ensure that you use the same database names as the full names of the Database assets that you create when you prepare the physical data layer in Data Catalog.<br><br>Warning When ingesting an Oracle data source, the value of the `databaseNames` property in your configuration file must be either the Oracle SID or service name, depending on whether you set the `connectAsServiceName` property to `true` or `false`. This means that the database in the technical lineage will have the name of the Oracle SID or service name. However, if the database asset in Data Catalog reflects the true name of the database, stitching will break. To resolve this issue and preserve stitching, you need to rename the database asset in Data Catalog to match the value you put for the `databaseNames` property. This is a known issue that we will fix in a future version of Collibra.<br><br>Important |

| Properties | Description |
| --- | --- |
| | HiveQL, MySQL and Teradata data sources don't have schemas. Therefore, HiveQL, MySQL and Teradata databases are stored in Data Catalog and technical lineage as Schema assets. The technical lineage Browse tab pane shows the following names:<br>◦ For HiveQL and Teradata:<br>▪ The database name is the name that you enter for the `collibraSystemName` property.<br>▪ The schema name is the name that you enter for the `database` property.<br>◦ For MySQL:<br>▪ The database name is the name that you enter for the `database` property. |

| Properties | Description |
| --- | --- |
| externalDbName | This property can be considered a means of database mapping, to help preserve stitching. It is relevant only for HiveQL, MySQL and Teradata data sources, specifically because they are database-less data sources.<br><br>You can add the key/value pair to the configuration file as follows: `"externalDbName": "CDATA"`<br><br>**See an example**<br>Let's say you ingest a HiveQL data source via Edge. Note that Edge gives the name "CDATA" for the database. The full path to a column is something like:<br><br>`Hive_123` (system) > `CDATA` (database) > `Hive_ABC` (schema) > `Table` > `Column`<br><br>Now, because HiveQL is database-less, the value that you give for the `database` property in your configuration file is used as the schema name in the technical lineage, and the value you give for `collibraSystemName` is used as the database name. But if `useCollibraSystemName` is set to `true`, then the value of `collibraSystemName` is also used as the system name. In that case, in the full path to the column, the system name and the database name are the same:<br><br>`Hive_123` (system) > `Hive_123` (database) > `Hive_ABC` (schema) > `Table` > `Column`<br><br>Notice the mismatch between the database names.<br><br>The `externalDbName` property tells the lineage harvester to use the value that you specify here for the database name in the technical lineage, |

| Properties | Description |
| --- | --- |
| | specifically "CDATA". This ensures that the full paths match and stitching is preserved. |
| connectAsServiceName | The option to determine whether your Oracle database uses an Oracle service name or SID.<br>◦ `True`: Connect to an Oracle database that uses an Oracle service name. Enter the service name in the `databaseNames` property.<br>◦ `False`: Connect to an Oracle database that uses an SID. Enter the SID in the `databaseNames` property.<br>Warning When ingesting an Oracle data source, the value of the `databaseNames` property in your configuration file must be either the Oracle SID or service name, depending on whether you set the `connectAsServiceName` property to `true` or `false`. This means that the database in the technical lineage will have the name of the Oracle SID or service name. However, if the database asset in Data Catalog reflects the true name of the database, stitching will break. To resolve this issue and preserve stitching, you need to rename the database asset in Data Catalog to match the value you put for the `databaseNames` property. This is a known issue that we will fix in a future version of Collibra.<br>Note This property is only valid for Oracle databases. It will be ignored for all other databases. |

| Properties | Description |
| --- | --- |
| hostname | The name of your database host. |
| collibraSystemName | The name of the data source's system or server. This is also the full name of your System asset in Data Catalog.<br><br>You must use the same system name as the full name of the System asset that you create when you prepare the physical data layer in Data Catalog. If you don't prepare the physical data layer, Collibra Data Lineage cannot stitch the data objects in your technical lineage to the assets in Data Catalog.<br><br>If the `useCollibraSystemName` property is:<br>◦ `false` (default), system or server names in table references in analyzed SQL code are ignored. This means that a table that exists in two different systems or servers is identified (either correctly or incorrectly) as a single data object, with a single asset full name.<br>◦ `true`, system or server names in table references are considered to be represented by different System assets in Data Catalog. The value of the `collibraSystemName` field is used as the default system or server name. |
| port | The port number. |

| Properties | Description |
| --- | --- |
| customConnectionProperties | An option to enable the lineage harvester to read additional connection parameters. This parameter is only required in very specific situations. If you don't need it, you can remove it from the configuration file.<br><br>Note You cannot currently use this property for Oracle data sources. |
| deleteRawMetadataAfterProcessing | The lineage harvester harvests metadata from specified data sources and uploads it in a ZIP file to a Collibra Data Lineage service instance, for processing.<br><br>You can use this optional property to specify whether or not the metadata should be deleted after it has been processed.<br><br>If the property is set to `true`, the metadata is deleted after processing. If set to `false`, the metadata is stored in an Amazon S3 bucket. |
| <Google BigQuery database> | This configuration section contains the required information for a Google BigQuery database. |
| id | The unique ID of your data source. For example, `my_third_data_source`. |
| type | The kind of data source. In this case, the value has to be `DatabaseBigQuery`. |

| Properties | Description |
| --- | --- |
| projectIDs | The IDs of your Google BigQuery project. You can add multiple projects. For example, `[ "first-project", "second-project", "third-project" ]`.<br><br>Note You have to use the same project ID as the full name of the Database asset that you create when you prepare the physical data layer in Data Catalog. |
| region | The location of your BigQuery data. This is the region that you specified when you create a data set.<br><br>You can only add one location as value. However, you can create separate BigQuery entries per location in the configuration file. As a result, you create a complete technical lineage with Google BigQuery data from different locations.<br><br>Note This property is optional. |
| auth | The path to a JSON file that contains authentication information.<br><br>Tip For more information about setting up the authentication, see the Google Big Query user guide. |

| Properties | Description |
| --- | --- |
| collibraSystemName | The name of the Google BigQuery system. This is also the full name of your System asset in Data Catalog.<br><br>You must use the same system name as the full name of the System asset that you create when you prepare the physical data layer in Data Catalog. If you don't prepare the physical data layer, Collibra Data Lineage cannot stitch the data objects in your technical lineage to the assets in Data Catalog. |
| deleteRawMetadataAfterProcessing | The lineage harvester harvests metadata from specified data sources and uploads it in a ZIP file to a Collibra Data Lineage service instance, for processing.<br><br>You can use this optional property to specify whether or not the metadata should be deleted after it has been processed.<br><br>If the property is set to `true`, the metadata is deleted after processing. If set to `false`, the metadata is stored in an Amazon S3 bucket. |
| <Snowflake database> | This configuration section contains the required information for a Snowflake database. |
| id | The unique ID of your data source. For example, `my_fourth_data_source`. |
| type | The kind of data source. In this case, the value has to be `DatabaseSnowflake`. |

| Properties | Description |
| --- | --- |
| username | The username that you use to sign in to your data source.<br><br>Note This property is deprecated. Use the **auth** property instead. The **username** and **auth** properties are mutually exclusive. |
| auth | This section indicates the authentication details to connect to the Snowflake database.<br><br>Note The **username** and **auth** properties are mutually exclusive. |
| type | The authentication method.<br><br>Specify one of the following values. The values are case-sensitive.<br><br>`Basic`<br>The username and password authentication method.<br>`KeyPair`<br>The key pair authentication method. |
| username | The user name that you use to connect to the Snowflake database. |

| Properties | Description |
| --- | --- |
| pathToPrivateKey | The path to your private key file. This property is required if you use the key pair authentication method.<br><br>Ensure that the private key matches the public key; otherwise, an error occurs indicating that the JWT token is invalid. For more information about the error, go to Snowflake JDBC driver error at login: net.snowflake.client.jdbc.SnowflakeSQLException: JWT token is invalid in Collibra Support Portal. |
| usePassword | The private key file password.<br><br>This property is required if you use the key pair authentication method. Specify one of the following values:<br><br>`true`<br>The password is required.<br>`false`<br>The password is not required. This is the default value. |
| hostname | The URL that you use to access Snowflake web console. For example, `<AccountName>.snowflakecomputing.com`. |

| Properties | Description |
| --- | --- |
| collibraSystemName | The name of the Snowflake system. This is also the full name of your System asset in Data Catalog.<br><br>You must use the same system name as the full name of the System asset that you create when you prepare the physical data layer in Data Catalog. If you don't prepare the physical data layer, Collibra Data Lineage cannot stitch the data objects in your technical lineage to the assets in Data Catalog. |
| databaseNames | The names of your databases.<br><br>Enter the database names of your data source between double quotes ("") and put everything between square brackets. If you want to include more than one database, separate them by a comma. For example, ["`MyFirstSnowflakeDatabase`", "`MySecondSnowflakeDatabase`"]<br><br>Note You have to use the same database names as the full names of the Database assets that you create when you prepare the physical data layer in Data Catalog. |
| warehouse | The name of your virtual warehouse.<br><br>Note This property is optional. |

| Properties | Description |
| --- | --- |
| customConnectionProperties | An option to enable the lineage harvester to read additional connection parameters. This parameter is only required in very specific situations. If you don't need it, you can remove it from the configuration file.<br><br>Example If you get an OSCP scan error, you can turn OSCP checking off by using the following value: `insecureMode=true`. |
| deleteRawMetadataAfterProcessing | The lineage harvester harvests metadata from specified data sources and uploads it in a ZIP file to a Collibra Data Lineage service instance, for processing.<br><br>You can use this optional property to specify whether or not the metadata should be deleted after it has been processed.<br><br>If the property is set to `true`, the metadata is deleted after processing. If set to `false`, the metadata is stored in an Amazon S3 bucket. |
| <SQL files in the lineage harvester output folder> | This configuration section contains the required information for SQL files of a data source that were previously downloaded by the lineage harvester and is stored in the lineage harvester output folder. |
| type | The kind of data source. In this case, the value has to be `LoadedSource`. |
| id | The unique ID of the data source that you uploaded to the lineage harvester folder. For example, `my_loaded_snowflake_source`. |

| Properties | Description |
|---|---|
| zipFile | The full path to the ZIP file that was created in the lineage harvester folder. |
| deleteRawMetadataAfterProcessing | The lineage harvester harvests metadata from specified data sources and uploads it in a ZIP file to a Collibra Data Lineage service instance, for processing.<br><br>You can use this optional property to specify whether or not the metadata should be deleted after it has been processed.<br><br>If the property is set to `true`, the metadata is deleted after processing. If set to `false`, the metadata is stored in an Amazon S3 bucket. |
| <Tableau> | This configuration section contains the required information for Tableau integration. |
| sources | This section contains all Tableau connection properties. |
| type | The kind of data source. In this case, the value has to be *Tableau*. |
| id | The unique ID to identify the Tableau metadata that was uploaded to the Collibra Data Lineage.<br><br>Tip This value can be anything as long as it is a unique. The lineage harvester uses the ID to identify a batch of data on the Collibra Data Lineage service instance. |
| url | The link to the data in Tableau. |

| Properties | Description |
| --- | --- |
| username | The username you use to sign in to the Tableau server.<br><br>Warning As of October 2022, Tableau is enforcing multi-factor authentication for Tableau Cloud Admin users. However, the lineage harvester doesn't support multi-factor authentication. Therefore, Tableau Cloud users with an Admin role must use token-based authentication. This does not affect Tableau Server users or Tableau Cloud users with an Explorer role.<br><br>Important If you want to use token-based authentication, you need to replace `username` with `tokenName`. You must specify either `username` or `tokenName`; if both exist, then `tokenName` is used. |
| tokenName | The lineage harvester authentication token.<br><br>Note For token-based authentication, use this property in your lineage harvester configuration file, instead of the `username` property. If both properties are present, `tokenName` is used. |

| Properties | Description |
| --- | --- |
| siteIds | The site IDs of the Tableau sites that you want to include in the ingestion process.<br><br>Warning Ensure that you specify the correct value. The correct value is the URL of the site to which you want to sign in. When you manually sign in to Tableau Server or Tableau Online, the site ID is the value that appears after **/site/** in the browser address bar. In the following example URLs, the site ID is `MarketingTeam`:<br>◦ Tableau Server: http://MyServer/#/site/MarketingTeam/projects<br>◦ Tableau Online: https://10ay.online.tableau.com/#/site/MarketingTeam/workbooks<br><br>On Tableau Server, however, the URL of the Default site does not specify the site. For example, the URL for a view named Profits, on a site named Sales, is http://localhost/#/site/sales/views/profits. The URL for this same view on the Default site is http://localhost/#/views/profits. The site name Sales does not figure in the URL. If you can't see the site ID, leave this property empty: `"siteIds": [""]`<br><br>Example If you want to ingest two Tableau sites "Site 1" and "Site 2", you can enter the following information in the siteIds property: ["site ID of Site 1", "site ID of Site 2"]. |

| Properties | Description |
| --- | --- |
| siteNames | The site names of the corresponding site IDs.<br><br>Important This property is:<br>◦ Optional for Tableau Server<br>◦ Mandatory for Tableau Online.<br><br>Warning If you have Tableau Server and you don't use this property, you must delete it from your configuration file. Don't leave the property in the configuration file without a value. |
| restOnly | Indication whether or not you would like to use both the Tableau REST API and Tableau Metadata API to harvest Tableau metadata.<br><br>◦ `false` (default): The lineage harvester will use the REST API and Metadata API to harvest Tableau metadata.<br>◦ `true`: The lineage harvester will only use the REST API to harvest Tableau metadata.<br><br>Warning If you only allow the lineage harvester to use the Tableau REST API, the harvester won't be able to process the necessary information for the technical lineage and the automatic stitching of Column assets to Tableau Data Attribute assets will not be possible. |

| Properties | Description |
| --- | --- |
| collibraSystemName | Regardless of the value set for the `useCollibraSystemName` property, the following is true:<br>◦ You must include this property in your configuration file.<br>◦ You can leave this property empty.<br>◦ Any value that you give is ignored.<br><br>If you want to specify name of the system or server, because, for example, you have multiple databases with the same name, then:<br><br>a. Set the `useCollibraSystemName` property to `true`.<br>b. Specify the system or server names in the `collibraSystemName` property in your Tableau <source ID> configuration file.<br><br>> Note This is a legacy property that will be deprecated in a future release. |
| domainId | The unique reference ID of the domain in Collibra Data Intelligence Cloud in which you want to ingest the Tableau assets.<br><br>**How do I find a domain reference ID?**<br>Open the relevant domain in Collibra. The URL looks like: https://<yourcollibrainstance>/domain/**22258f64-40b6-4b16-9c08-c95f8ec0da26**?view=00000000-0000-0000-0000-000000040001. In this example, the reference ID is in bold. |

| Properties | Description |
| --- | --- |
| excludeImages | Optional parameter for excluding the downloading of images.<br><br>To exclude the downloading of images, set this property to `true`. |
| deleteRawMetadataAfterProcessing | The lineage harvester harvests metadata from specified data sources and uploads it in a ZIP file to a Collibra Data Lineage service instance, for processing.<br><br>You can use this optional property to specify whether or not the metadata should be deleted after it has been processed.<br><br>If the property is set to `true`, the metadata is deleted after processing. If set to `false`, the metadata is stored in an Amazon S3 bucket. |

<table>
<tr><th>Properties</th><th>Description</th></tr>
<tr><td>paging</td><td>Optional parameter for customizing the Tableau API pagination settings.<br>The default values are sufficient in most cases; however, you can decrease them to help mitigate node limit errors, or increase them to speed up API calls.<br><br>The complete list of pagination settings, descriptions and default values<br><br><pre>"paging": {
   "databasesPageSize": 100,
   "tablesPageSize": 100,
   "tablesColumnsPageSize": 100,
   "tableColumnsPageSize": 1000,
   "datasourcesPageSize": 50,
   "datasourcesFieldsPageSize": 50,
   "datasourceFieldsPageSize": 100,
   "worksheetsPageSize": 100,
   "worksheetsFieldsPageSize": 100,
   "worksheetFieldsPageSize": 1000,
   "dashboardsPageSize": 100,
   "columnsLimit": 20,
   "fieldsLimit": 20
   }</pre><br>Settings per metadata type and descriptions<br><br><table>
<tr><th>Metadata type</th><th>Setting and description</th></tr>
<tr><td>Dashboard</td><td>◦ <code>dashboardsPageSize</code>: The number of dashboards per page.</td></tr>
</table></td></tr>
</table>

<table>
<tr><th>Properties</th><th>Description</th></tr>
<tr><td></td><td>
<table>
<tr><th>Metadata type</th><th>Setting and description</th></tr>
<tr><td>Worksheet</td><td>◦ <code>worksheetsPageSize</code>: The number of worksheets per page.<br>◦ <code>worksheetsFieldsPageSize</code>: The number of worksheet fields per page.</td></tr>
<tr><td>Database</td><td>◦ <code>databasesPageSize</code>: The number of databases per page.</td></tr>
<tr><td>Table</td><td>◦ <code>tablesPageSize</code>: The number of tables per page.<br>◦ <code>tablesColumnsPageSize</code>: The number of table columns per page.</td></tr>
<tr><td>Table columns</td><td>◦ <code>tableColumnsPageSize</code>: The number of table columns per page.</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><th>Properties</th><th>Description</th></tr>
<tr><td></td><td>
<table>
<tr><th>Metadata type</th><th>Setting and description</th></tr>
<tr><td>Data source</td><td>
<ul>
<li><code>datasourcesPageSize</code>: The number of data sources per page.</li>
<li><code>datasourcesFieldsPageSize</code>: The number of data source fields per page.</li>
<li><code>columnsLimit</code>: The number of data source field columns per page.</li>
<li><code>fieldsLimit</code> : The number of referenced data source fields per page.</li>
</ul>
</td></tr>
<tr><td>Data source field</td><td>
<ul>
<li><code>datasourceFieldsPageSize</code>: The number of data source fields per page.</li>
<li><code>columnsLimit</code>: The number of data source field columns per page.</li>
<li><code>fieldsLimit</code> : The number of referenced data source fields per page.</li>
</ul>
</td></tr>
</table>
</td></tr>
</table>

| Properties | Description |
| --- | --- |
| <Power BI (deprecated)> | This configuration section contains the required information for Power BI integration.<br><br>Note<br>◦ You have to purchase the Power BI connector and lineage feature. Then you need to add the Power BI connection properties to both the lineage harvester configuration file and the Power BI harvester configuration file to ingest Power BI metadata into Data Catalog.<br>◦ This integration method is deprecated. We will continue to fix issues, but the development of new features and improvements is discontinued. |
| type | The kind of data source. In this case, the value has to be `ExistingLineage`. |
| id | The unique ID of the Power BI metadata you harvested via the Power BI harvester.<br><br>You must use the same ID as the value you used in the Power BI configuration file sourceID property. |
| <Looker> | This configuration section contains the required information for Looker integration. |
| collibraSystemName | This property is deprecated for Looker integration. The lineage harvester does not take into account any value that you enter here. |

| Properties | Description |
| --- | --- |
| id | The unique ID of your Looker metadata. For example, *my_looker*.<br><br>Tip This value can be anything as long as it is unique and human readable. The ID identifies the batch of Looker metadata on the Collibra Data Lineage service. |
| type | The kind of data source. In this case, the value has to be *Looker*. |
| lookerUrl | The URL to your Looker API.<br><br>Tip There are two ways to find the Looker API URL:<br>◦ In the API Host URL field in the Looker Admin menu. If this field is empty, you can use the default Looker API URL which you can find in the interactive API documentation.<br>◦ In the interactive API documentation URL. It is the part of the URL before `/api-docs/`. |
| clientId | The username you use to access the Looker API. |
| domainId | The unique ID of the domain in Collibra Data Intelligence Cloud in which you want to ingest the Looker assets. |

| Properties | Description |
| --- | --- |
| deleteRawMetadataAfterProcessing | The lineage harvester harvests metadata from specified data sources and uploads it in a ZIP file to a Collibra Data Lineage service instance, for processing.<br><br>You can use this optional property to specify whether or not the metadata should be deleted after it has been processed.<br><br>If the property is set to `true`, the metadata is deleted after processing. If set to `false`, the metadata is stored in an Amazon S3 bucket. |
| <MicroStrategy> | This configuration section contains the required information for MicroStrategy integration. |
| type | The kind of data source. In this case, the value has to be `MicroStrategy`. |
| collibraSystemName | This property is deprecated for MicroStrategy integration. The lineage harvester does not take into account any value that you enter here. |
| id | The unique ID of your MicroStrategy metadata. For example, `my_microstrategy`.<br><br>Tip This value can be anything as long as it is unique and human readable. The ID identifies the batch of MicroStrategy metadata on the Collibra Data Lineage service instance. |
| domainId | The unique reference ID of the domain in Collibra Data Intelligence Cloud in which you want to ingest the MicroStrategy assets. |

| Properties | Description |
| --- | --- |
| username | The username that you use to sign in to MicroStrategy. |
| hostname | The endpoint that you use to access the PostgreSQL repository or remote data source, depending on where you installed the lineage harvester.<br>For example `remote.postgres.com`. |
| port | The port number. |
| databaseName | Optionally, the name of your database. For example `poc_metadata`. |
| deleteRawMetadataAfterProcessing | The lineage harvester harvests metadata from specified data sources and uploads it in a ZIP file to a Collibra Data Lineage service instance, for processing.<br>You can use this optional property to specify whether or not the metadata should be deleted after it has been processed.<br>If the property is set to `true`, the metadata is deleted after processing. If set to `false`, the metadata is stored in an Amazon S3 bucket. |
| <SQL Server Reporting Services and Power BI Report Server> | This configuration section contains the required information for SQL Server Reporting Services and Power BI Report Server integration. |

| Properties | Description |
| --- | --- |
| collibraSystemName | Regardless of the value set for the `useCollibraSystemName` property, the following is true:<br>◦ You must include this property in your configuration file.<br>◦ You can leave this property empty.<br>◦ Any value that you give is ignored.<br><br>If you want to specify name of the system or server, because, for example, you have multiple databases with the same name, then:<br><br>a. Set the `useCollibraSystemName` property to `true`.<br>b. Specify the system or server names in the `collibraSystemName` property in you SSRS-PBRS <source ID> configuration file.<br><br>Note This is a legacy property that will be deprecated in a future release. |
| id | The unique ID to identify the SSRSmetadata that was uploaded to the Collibra Data Lineage service.<br><br>Tip This value can be anything as long as it is a unique. The lineage harvester uses the ID to identify a batch of data on the Collibra Data Lineage service. |

| Properties | Description |
| --- | --- |
| type | The kind of data source. In this case, the value has to be *SSRS* or *PBIRS*.<br><br>Note There is no difference between type SSRS or PBIRS. |
| url | The URL to the server's web portal. By default, the URL is *http://<computer-name>/reports*. For example, "http://1.23.45.678/PowerBIReports". |
| username | The username you use to sign in to the web portal.<br><br>Tip If you use NTLM authentication, your username also contains the NTLM domain name. For example `MyDomain\\username`. |
| domainId | The unique ID of the domaindomain in Collibra Data Intelligence Cloud in which you want to ingest the SSRS assets.<br><br>Finding the domain ID<br>a. Open the domain.<br>b. Copy the domain ID.<br><br>Tip If you go to your domain, you can find the domain ID in the URL. The URL looks like: https://<yourcollibrainstance>/domain/**22258f64-40b6-4b16-9c08-c95f8ec0da26**?view=00000000-0000-0000-0000-000000040001. In this example, the domain ID is in bold. |

| Properties | Description |
| --- | --- |
| folderFilter | An option to exclude specific folders that contain reports or KPIs from the ingestion process.<br><br>You can add multiple folders by listing folder names, providing the full path to folders or by using a wildcard:<br><br>◦ Use folder names when the folder name is unique: ["folder 1", "folder 2"]<br>◦ Use the full path to the folder to only ingest a specific folder: ["/database1/folder1", "/database2/folder2"]<br>◦ Use a wildcard to ingest all child folders or a specific folder: ["/folder1/*", "/folder2/*"]<br>You can also use a combination of these methods. For example, ["folder 1", "/database/folder2", /folder3/*"]<br><br>Important This property must be included in your configuration file and it cannot be empty. If you want to ingest all folders, use `*`, for example: `"folderFilter":["*"]`.<br><br>Tip For more information about connecting to a SSRS or PBRS folder, see the Microsoft documentation. |

| Properties | Description |
| --- | --- |
| deleteRawMetadataAfterProcessing | The lineage harvester harvests metadata from specified data sources and uploads it in a ZIP file to a Collibra Data Lineage service instance, for processing.<br><br>You can use this optional property to specify whether or not the metadata should be deleted after it has been processed.<br><br>If the property is set to `true`, the metadata is deleted after processing. If set to `false`, the metadata is stored in an Amazon S3 bucket. |
| <Power BI> | This configuration section contains the required information for Power BI integration via the lineage harvester. |
| type | The kind of data source. In this case, the value has to be *PowerBI*. |
| id | The unique ID to identify the Power BI service metadata that was uploaded to the Collibra Data Lineage service instance. |
| tenantDomain | The Power BI tenant domain is the domain associated with the Microsoft Azure tenant.<br><br>This domain is either a default domain or a custom domain. For example, *collibrapowerbi.onmicrosoft.com*.<br><br>> Note Usually, you can find a list of Power BI tenant or server domains in your Azure Active Directory or in the top right menu. |

| Properties | Description |
| --- | --- |
| loginFlow | This section describes the authentication information for accessing your Power BI metadata.<br><br>The lineage harvester supports two authentication methods: service principal, and username and password. For complete information on your authentication options, see Authentication. |
| type | This depends on the authentication method you use.<br><br>◦ Service principle: The value should be `ServicePrincipal`.<br>◦ Username and password: The value should be `ResourceOwnerPasswordCredentials`. |
| applicationId | The unique ID of the Microsoft Azure Application (client) ID. |
| username | The email address of your Azure Active Directory user.<br><br>Tip This property only applies if you are using the username and password authentication method. |
| domainId | The reference ID of the domain in Collibra in which you want to ingest Power BI metadata. |

| Properties | Description |
| --- | --- |
| collibraSystemName | Regardless of the value set for the `useCollibraSystemName` property, the following is true:<br>◦ You must include this property in your configuration file.<br>◦ You can leave this property empty.<br>◦ Any value that you give is ignored.<br><br>If you want to specify name of the system or server, because, for example, you have multiple databases with the same name, then:<br><br>a. Set the `useCollibraSystemName` property to `true`.<br>b. Specify the system or server names in the `collibraSystemName` property in you Power BI <source ID> configuration file.<br><br>Note This is a legacy property that will be deprecated in a future release. |
| deleteRawMetadataAfterProcessing | The lineage harvester harvests metadata from specified data sources and uploads it in a ZIP file to a Collibra Data Lineage service instance, for processing.<br><br>You can use this optional property to specify whether or not the metadata should be deleted after it has been processed.<br><br>If the property is set to `true`, the metadata is deleted after processing. If set to `false`, the metadata is stored in an Amazon S3 bucket. |

3. Save the configuration file.

4. Start the lineage harvester again and do one of the following:
   - To process data from all data sources in the configuration file, run the following command:
     For windows:

     ```
     .\bin\lineage-harvester.bat full-sync
     ```

     For other operating systems:

     ```
     ./bin/lineage-harvester full-sync
     ```

   - To process data from specific data sources in the configuration file, run the following command:
     For windows:

     ```
     .\bin\lineage-harvester.bat full-sync -s "ID of the data source"
     ```

     For other operating systems:

     ```
     ./bin/lineage-harvester full-sync -s "ID of the data source"
     ```

     » The lineage harvester sends the data source information to the Collibra Data Lineage service by using Collibra REST API, where it is parsed and analyzed. As a result, the technical lineage is created and shown in Data Catalog.
5. When prompted, enter the passwords to connect to Collibra and your data sources.
   Do one of the following:
   - Enter the passwords in the console.
     » The passwords are encrypted and stored in **/config/pwd.conf**.
   - Provide the passwords via command line.
     » The passwords are stored locally and not in your lineage harvester folder.

> Tip If the lineage harvester log shows an error message or the harvesting process fails, you can use the technical lineage troubleshooting guide to fix your issue.

## What's next?

If you prepared the physical data layer and have the required permissions, you can go to the asset page of a Table, Column Power BI Column or Looker Look asset from the data

source that you added in the configuration file and visualize the technical lineage. The technical lineage shows the data source information of data sources that have been successfully analyzed and processed.

The lineage harvester can also use scheduled jobs to synchronize the data sources on fixed times.

> Tip You can check the progress of the technical lineage creation in Activities. The **Results** field indicates how many relations were imported into Data Catalog. Go to the status page to see the log files of the SQL analysis.

## The configuration file generator

The configuration file generator helps you create your lineage harvester configuration file more easily by providing the structure of the file with the correct properties per data source.

### The lineage harvester configuration file

The lineage harvester uses a configuration file to connect to data sources, BI tools and ETL tools. The configuration file contains references to the data sources for which you want to create a technical lineage. You have to prepare the configuration file if you want to create a technical lineage and add new relations of the type "Data Element targets / sources Data Element" between existing assets in Data Catalog and "Column is target of / is source of Data Attribute" between assets from ingested BI sources and assets in Data Catalog.

> Tip You have to save the configuration file in the **config** directory in the lineage harvester folder.

### Empty configuration file

When you run the lineage harvester for the first time, it creates an empty configuration file. To create a technical lineage, you have to manually add properties and values, per data source, to this configuration file.

The following image shows an example of the empty configuration file created by the lineage harvester.

```
{
   "general" : {
       "catalog" : {
           "url" : "",
           "username" : "",
       },
       "useCollibraSystemName" : false
   },
   "sources" : [ {
       "type" : "Database",
       "id" : "MyDB",
       "hostname" : "",
       "username" : "",
       "dialect" : "",
       "collibraSystemName" : "",
       "databaseNames" : [ ],
       "port" : 1521
   } ]
}
```

## Configuration file generator

> Tip The configuration file generator is only available in the online version of this guide.

The configuration file generator creates an example configuration file with the data source properties of your choosing:

1. Scroll down to the configuration file example.
2. Paste the example in your empty configuration file in the lineage harvester**config** folder.
3. Replace the values in the example to match your actual data source information.

   > Tip Make sure you understand each property and know which values you must use to access your data source information.

4. Run the lineage harvester.

> Warning Some browser plug-ins may slow the configuration file generator down.

```
{
   "general": {
       "catalog" : {
           "url" : "https://companydomain.collibra.com",
           "username" : "my-Collibra-username"
           },
           "useCollibraSystemName" : false
   },
   "sources" : [

   {
       "collibraSystemName" : "datastage-system-name",
       "id" : "datastage_source",
       "type" : "ExternalDirectory",
       "dirType" : "DATASTAGE",
       "path" : "/path/to/the/datastage/folder/",
       "mask" : "*",
       "recursive" : false,
       "deleteRawMetadataAfterProcessing": false
   }
   {
       "collibraSystemName" : "infa-system-name",
       "id" : "informatica_source",
       "type" : "ExternalDirectory",
       "dirType" : "INFA",
       "path" : "/path/to/the/informatica/folder/",
       "mask" : "*",
       "recursive" : false,
       "deleteRawMetadataAfterProcessing": false
   }
   {
       "collibraSystemName" : "ssis-system-name",
       "id" : "datastage_source",
       "type" : "ExternalDirectory",
       "dirType" : "SSIS",
       "path" : "/path/to/the/ssis/folder/",
       "mask" : "*",
       "recursive" : false,
       "deleteRawMetadataAfterProcessing": false
   }
   {
       "type" : "IICS",
       "id" : "iics_source",
       "collibraSystemName" : "iics-development",
       "loginUrl" : "https://dm-us.informaticaintelligentcloud.com",
       "username" : "login-iics",
       "deleteRawMetadataAfterProcessing": false,
       "objects" : [
           {
               "path" : "Default/Sales",
```

```
                "type" : "Project"
            },
            {
                "path" : "My Project/Statistics",
                "type" : "Project"
            }
        ]
    }
    {
        "id" : "my-matillion-project",
        "type" : "Matillion",
        "url" : "https://my-domain",
        "groupName" : "my-matillion-group",
        "projectName" : "redshift-project",
        "environmentName" : "redshift-environment",
        "dialect" : "redshift",
        "startTimestamp" : 1594080796911,
        "collibraSystemName": "Matillion-system",
        "deleteRawMetadataAfterProcessing": false,
        "auth": {
            "type": "Basic",
            "username": "ec2-user"
        }
    }
    {
        "type": "Tableau",
        "id": "unique-ID",
        "url": "URL to Tableau server?",
        "username": "Admin",
        "siteIds": ["site ID of Tableau Site 1", "site ID of Tableau Site
2"],
        "siteNames": ["site name of Tableau Site 1", "site name of Tableau
Site 2"],
        "restOnly": false,
        "collibraSystemName": "tableau-system-name",
        "domainId": "Domain-resource-ID",
        "excludeImages": true,
        "deleteRawMetadataAfterProcessing": false,
        "paging": {
            "pagination-setting": 100,
            "pagination-setting-2": 100
        }
    }
    {
        "collibraSystemName" : "looker",
        "id" : "looker-source",
        "type" : "Looker",
        "lookerUrl" : "https://<instance-name.api.looker.com",
        "clientId" : "my-looker-api-user-name",
        "domainId" : "22258f64-40b6-4b16-9c08-c95f8ec0da26",
```

```
        "deleteRawMetadataAfterProcessing": false   }
    {
        "type" : "ExistingLineage",
        "id" : "MyPowerBISourceID",
        "deleteRawMetadataAfterProcessing": false
    }
    {
        "collibraSystemName": "",
        "id": "<unique-id>",
        "type": "SSRS",
        "url": "http://<IP address or computer name>/Reports",
        "username": "<server-api-user-name>",
        "domainId": "<domain-resource-id>",
        "folderFilter":  ["/Folder1/*", "Folder2"],
        "deleteRawMetadataAfterProcessing": false
    }
    {
        "collibraSystemName" : "custom-system-name",
        "id" : "MyCustomLineage",
        "type" : "ExternalDirectory",
        "dirType" : "custom-lineage",
        "path" : "/path/to/custom-lineage/dir",
        "deleteRawMetadataAfterProcessing": false
    }
    {
        "type" : "LoadedSource",
        "id" : "MySource",
        "zipFile" : "/path/to/source-MySource.zip",
        "deleteRawMetadataAfterProcessing": false
    }
    {
        "id" : "database_source",
        "type" : "Database",
        "username" : "MyUsername",
        "dialect" : "hive",
        "databaseNames" : ["MyDefaultDbName"],
        "hostname" : "localhost",
        "collibraSystemName" : "apache-hive-system",
        "port" : 1521,
        "deleteRawMetadataAfterProcessing": false,
        "customConnectionProperties" : ""
    }
    {
        "id" : "oracle_source",
        "type" : "Database",
        "username" : "MyUsername",
        "dialect" : "oracle",
        "databaseNames" : ["oracle-service-name"],
        "connectAsServiceName" : true,
        "hostname" : "localhost",
```

```
        "collibraSystemName" : "oracle-system-name",
        "port" : 1521,
        "deleteRawMetadataAfterProcessing": false
    }
    {
        "id" : "bigquery_source",
        "type" : "DatabaseBigQuery",
        "projectIDs" : [ "bigquery_project1", "bigquery_project2" ],
        "region": "europe-west1"
        "auth" : "/path/to/the/authentication/file.json",
        "collibraSystemName" : "bigquery-system-name",
        "deleteRawMetadataAfterProcessing": false
    }
    {
        "id" : "snowflake_source",
        "type" : "DatabaseSnowflake",
        "auth": {
            "type": "KeyPair|Basic",
            "username": "some_username",
            "pathToPrivateKey": "path_to_your_private_key_file",
            "usePassword": "true|false"
        },
        "hostname" : "MyAccountName.snowflakecomputing.com",
        "collibraSystemName" : "snowflake-system-name",
        "databaseNames" : ["MyFirstDbName","MySecondDbName"],
        "warehouse" : "MySnowflakeWarehouseName",
        "deleteRawMetadataAfterProcessing": false,
        "customConnectionProperties" : ""
    }
    {
        "type": "Microstrategy",
        "id": "microstrategy-batch",
        "collibraSystemName": "system-name",
        "domainId": "<domain-resource-id>",
        "username": "mstr",
        "hostname": "remote.postgres.com",
        "port": 5432,
        "databaseName": "poc_metadata",
        "deleteRawMetadataAfterProcessing": false
    }
    {
        "type" : "PowerBI",
        "id" : "power-bi-1",
        "tenantDomain": "collibra3.onmicrosoft.com",
        "loginFlow": {
            "type": "ServicePrincipal",
            "applicationId": "be560fac-7545-4ce2-ad9f-cbce14c59af6"
        },
        "domainId": "domain-reference-ID",
        "collibraSystemName": "collibra-system-name",
```

```
        "deleteRawMetadataAfterProcessing": false
    }
    {
        "id" : "sqldirectory_source",
        "type" : "SqlDirectory",
        "path" : "/path/to/the/sql/folder/",
        "mask" : "*",
        "recursive" : false,
        "dialect" : "db2",
        "database" : "MyDefaultDbName",
        "collibraSystemName" : "data-source-system",
        "schema" : "MyDefaultDbSchema",
        "verbose" : true,
        "deleteRawMetadataAfterProcessing": false
    } ]
}
```

## Informatica PowerCenter

The following example shows an Informatica PowerCenter <source ID> configuration file.

```
{
    "connectionDefinitions": {
        "oracle_source": {
            "dbname": "oracle-source-database-name1",
            "schema": "my Oracle source schema",
            "dialect": "oracle"
        },
        "oracle_target": {
            "dbname": "oracle-target-database-name2",
            "schema": "my other oracle target schema",
            "dialect": "oracle"
        }
    },
    "collibraSystemNames": {
        "databases": [
            {
                "dbname": "oracle-source-database-name1",
                "collibraSystemName": "oracle-system-name1"
            },
            {
                "dbname": "oracle-target-database-name2",
                "collibraSystemName": "oracle-system-name2"
            }
        ],
        "connections": [
            {
                "connectionName": "oracle-connection-name1",
                "collibraSystemName": "oracle-system-name1"
            },
```

```
            {
                "connectionName": "oracle-connection-name2",
                "collibraSystemName": "oracle-system-name2"
            }
        ]
    }
}
```

## SQL Server Integration Services

The following example shows an SQL Server Integration Services connection definitions configuration file.

```
{
  "ConnStringRegExTranslation": {

    "Data Source=dhb-sql-prod;Initial Catalog=SFG_repl_
staging;Provider=SQLNCLI11;Integrated Security=SSPI.*": {
      "dbname": "DATAHUB",
      "schema": "DBO",
      "dialect": "mssql",
      "collibraSystemName" : "WAREHOUSE"
    },

    "Server=sb-dhub;User ID=SYS_USER;Initial
Catalog=STAGEDB;Port=6306.*": {
      "dbname": "STAGEDB",
      "schema": "STAGE_OWNER",
      "dialect": "sybase",
      "collibraSystemName" : ""
    }

  }
}
```

## IBM InfoSphere DataStage

The following example shows a DataStage connection definitions configuration file.

```
{
  "OdbcDataSources": {
    "oracle-data-source": {
      "dbname": "my-oracle-database",
      "schema": "my-oracle-schema",
      "dialect": "oracle",
      "collibraSystemName": "my-system"
```

```
    },
    "mssql-data-source": {
      "dbname": "my-mssql-database",
      "schema": "my-mssql-schema",
      "dialect": "mssql",
      "collibraSystemName": "my-system"
    }
  },
  "NonOdbcConnectors": {

    "admin@database-name": {
      "dbname": "my-netezza-database",
      "schema": "my-netezza-schema",
      "dialect": "netezza",
      "collibraSystemName": "my-system"
    },
    "admin@second-database-name": {
      "dbname": "my-second-netezza-database",
      "schema": "my-second-netezza-schema",
      "dialect": "netezza",
      "collibraSystemName": "my-system"
    }
  }
}
```

## Informatica Intelligent Cloud Services

The following example shows an Informatica Intelligent Cloud Services <source ID> configuration file.

```
{
   "collibraSystemNames": {
       "connections": [
           {
               "connectionName": "DG_con_standby_cmdm_clientors",
               "collibraSystemName": "PUBLIC"
           },
           {
               "connectionName": "DG_con_dev_dg_dgiauser_su",
               "collibraSystemName": "PUBLIC"
           }
       ]
   },
   "connectionDefinitions": [
       {
           "connectionName": "DG_con_standby_cmdm_clientors",
           "databaseName": "main",
           "schemaName": "dbo",
```

```
            "dialect": "oracle"
        },
        {
            "connectionName": "DG_con_dev_dg_dgiauser_su",
            "databaseName": "main",
            "schemaName": "dbo",
            "dialect": "oracle"
        }
    ]
}
```

## Tableau

The following example shows a Tableau <source ID> configuration file.

```
{
  "collibraSystemNames": {
    "databases": [
      {
        "hostName": "database-hostname",
        "collibraSystemName": "public"
      }
    ],
    "files": [
      {"filePath": "C:\\ProgramData\\Tableau\\Tableau
Server\\data\\files\\sample.xls",
        "collibraSystemName": "sample-files"
      }
    ],
    "connectors": [
      {
        "connectorUrl": "tableau-server-connector-url.com",
        "collibraSystemName": "Oracle-connector"
      }
    ],
    "cloudFiles": [
      {
        "name": "file-name",
        "collibraSystemName": "FILE"
      }
    ]
  },
  "databaseMapping": {
    "<hostname:port>":"<actual database name>"
  },
  "filters": {
     "projects":{
       "site_name2 > project_name2": "domain-reference-id2",
       "site_name3 > project_name3 > subproject_name": "domain-
```

```
reference-id2"
        }
    }
}
```

## Looker

The following example shows a Looker <source ID> configuration file.

```
{
    "Connections": {
        "connection-object1": {
            "dialect": "mssql",
            "schema": "mssql-schema-name",
            "dbname": "mssql-database-name",
            "collibraSystemName": "mssql-system-name"
        },
        "connection-object2": {
            "dialect": "oracle",
            "schema": "oracle-schema-name",
            "dbname": "oracle-database-name",
            "collibraSystemName": "oracle-system-name"
        }
    }
    "filters":[
        {
        "domainId":"<reference ID>",
            "description":"any-description",
            "folderNames":["Folder1", "Folder2"]
        },
        {
        "domainId":"<reference ID>",
            "description":"any-description",
            "folderNames":["Folder3", "Folder4"]
        },
        {
        "domainId":"<reference ID>",
            "description":"any-description",
            "folderIds":["123xxxx", "456xxxx"]
        }
        ]
    }
}
```

## SQL Server Reporting Services and Power BI Report Server

The following example shows a SQL Server Reporting Services and Power BI Report Server <source ID> configuration file.

```
{
   "DataSources": {
       "Redshift": {
           "dbname": "redshift-database-name",
           "schema": "redshift-schema-name",
           "dialect": "redshift",
           "collibraSystemName": "redshift-system-name"
       },
       "Oracle": {
           "dbname": "oracle-database-name",
           "schema": "oracle-schema-name",
           "dialect": "oracle",
           "collibraSystemName": "oracle-system-name"
       }
   },
   "CustomDataSources":
       "/path to report/custom data souce name": {
           "dbname": "mssql-database-name",
           "dialect": "mssql"
       }
   }
}
```

## Power BI

The following example shows a Power BI <source ID> configuration file.

```
{
   "found_dbname=databasename1;found_hostname=*;found_schema=schema1":
{
       "dbname": "mssql-database-name",
       "schema": "mssql-schema-name",
       "dialect": "mssql",
       "collibraSystemName": "mssql-system-name"
   },
   "found_dbname=databasename2;found_hostname=server-
name.onmicrosoft.com;found_schema=schema2": {
       "dbname": "oracle-database-name",
       "schema": "oracle-schema-name",
       "dialect": "oracle",
       "collibraSystemName": "oracle-system-name"
   },
```

```
   "filters":[
       {
       "domainId": "<domain-ref-id>",
       "description": "FirstFilter",
       "workspaceNames": ["workspace1", "workspace2"],
       "workspaceIds": ["id3","id4"],
       "capacityNames": ["capacity1","capacity2"]
       },
       {
       "domainId": "<domain-ref-id>",
       "description": "SecondFilter",
       "workspaceNames": ["workspace3", "workspace4"],
       "capacityIds": ["id1","id2"]
       }
   ]
}
```

## Matillion

The following example shows a Matillion <source ID> configuration file.

```
{
   "collibraSystemNames":
       {
           "sources":[
               {
                   "jobName":"<name of job>",
                   "collibraSystemName":"<name>"
               },
               {
                   "jobName":"<name of job>",
                   "collibraSystemName":"<name>"
               }
           ],
           "targets":[
               {
                   "jobName":"<name of job>",
                   "collibraSystemName":"<name>"
               },
               {
                   "jobName":"<name of jobv",
                   "collibraSystemName":"<name>"
               }
           ]
       }
}
```

# Prepare an external directory folder

If you want to create a technical lineage for an external directory such as Informatica PowerCenter, SQL Server Integration Services (SSIS) or IBM InfoSphere DataStage, you must prepare a folder with the external directory's data source files.

If the external directory files do not have the necessary information, for example a database and a schema, to stitch the data sources, you have to provide the connection definitions manually via a JSON configuration file. This is required at each connection, regardless of whether the `useCollibraSystemName` property in the lineage harvester configuration file is set to `true` or `false`.

> Tip Go to the online version of the user guide for more detailed steps and examples.

> Note You can also create and configure a JSON file to define a custom technical lineage.

> Note For best technical lineage results, we recommend harvesting JDBC sources when possible, rather than using an external directory of source files. If harvesting a JDBC source is not possible, the files in your external directory need to be ordered alphabetically.

## Prerequisites

- You have IBM InfoSphere Information Server version 11.5 or newer.
- You have Informatica PowerCenter version 9.6 or newer.
- You have SQL Server Integration Services 2012 or newer with package format version 6 or newer.
- You have Microsoft Visual Studio version 2012 or newer.
- You have downloaded the lineage harvester and you have the necessary system requirements to run it.

- You have prepared the physical data layer in Data Catalog.

  > Note To stitch the data objects in the source and target data sources in external directories with Data Catalog assets, you first have to register those data sources in Data Catalog.

> Tip If you want to create a technical lineage for Informatica Intelligent Cloud Services Data Integration, you don't have to create a folder with data source files. You add your data source information directly to the lineage harvester configuration file.

## Steps to create a technical lineage for Informatica PowerCenter

1. Create a local folder.
2. Export the Informatica objects or repository for which you want to create a technical lineage to the local folder.

   > Note
   > - All XML and parameter files, for example PAR, TXT or PRM files in this folder and its subfolders are taken into account when you create a technical lineage, but Collibra Data Lineage only shows a technical lineage for workflows that have mappings with sources, transformations and targets. Collibra supports the most common Informatica PowerCenter transformations. For more information, see the Informatica PowerCenter documentation.
   > - A technical lineage is created when the following tags are present in your XML file:
   >   - <REPOSITORY>
   >   - <FOLDER>
   >   - <SOURCE> / <TARGET>
   >   - <SESSION>
   >   - <MAPPING>
   >   - <TRANSFORMATION> (within a <MAPPING> tag)

3. Put your parameter files in the right location.

| If... | Then... |
|---|---|
| all parameter files are PAR files | No action required |
| not all parameter files are PAR files | a. Create a new folder in the local folder.<br>b. Name the folder *techlin-param*.<br>c. Move all parameter files that are used by the exported XML to the **techlin-param** folder.<br>d. In the lineage harvester configuration file, set the `recursive` property to `true`.<br>Note The lineage harvester only takes into account parameter files in the **techlin-param** folder. |

4. Optionally, create a source ID configuration file with connection definitions and system names:

   Tip If you previously created a technical lineage for Informatica PowerCenter with connection definitions, the connection_definitions.conf file will still be taken into account.

   a. Create a new JSON file in the lineage harvester **config** folder.
   b. Give the JSON file the same name as the value of the `Id` property in the lineage harvester configuration file.

      Example The value of the `Id` property in the lineage harvester configuration file is `informatica-source-1`. As a result, the name of your JSON file should be *informatica-source-1.conf*.

c. For each data source, add the following content to the JSON file:

| Property | Description |
| --- | --- |
| connectionDefinitions | This section contains the connection properties to a source in Informatica PowerCenter. |
| <connectionName> | The type of your source or target data source.<br>This section contains the connection properties to a source or target in Informatica PowerCenter. |
| dbname | The name of your source or target database. |
| schema | The name of your source or target schema. |

| Property | Description |
| --- | --- |
| dialect | The dialect of the referenced database.<br><br>Tip<br>You can enter one of the following values:<br><br>■ *azure*, for an Azure SQL Server data source.<br>■ *bigquery*, for a Google BigQuery data source.<br>■ *db2*, for an IBM DB2 data source.<br>■ *hana*, for a SAP Hana data source.<br>■ *hana-cviews*, for SAP Hana data calculation views.<br>■ *hive*, for a HiveQL data source.<br>■ *greenplum*, for a Greenplum data source.<br>■ *mssql*, for a Microsoft SQL Server data source.<br>■ *mysql*, for a MySQL data source.<br>■ *netezza*, for a Netezza data source.<br>■ *oracle*, for an Oracle data source.<br>■ *postgres*, for a PostgreSQL data source.<br>■ *redshift*, for an Amazon Redshift data source.<br>■ *snowflake*, for a Snowflake data source.<br>■ *spark*, for a Spark SQL data source.<br>■ *sybase*, for a Sybase data source.<br>■ *teradata*, for a Teradata data source. |

| Property | Description |
|---|---|
| collibraSystemNames | This section contains the system or server name that is specified in your database and referenced in your connection.<br><br>Note This section is only required when the useCollibraSystemName flag in the lineage harvester configuration file is set to `true`. |
| databases | This section contains the database information. This is required to connect directly to the system or server of the database. |
| dbname | The name of the database. The database name is the same as the name you entered in the <connectionName> section. |
| collibraSystemName | The system or server name of the database. |
| connections | This section contains the connection information. This is required to reference to the system or server of the connection. |
| connectionName | The name of the connection. |
| collibraSystemName | The system or server name of the connection. |

Important If you are using variables in Informatica PowerCenter, add the value of the variable instead of the name in the connection definitions JSON file. For example, if the parameter file contains `$DBConnection_`

`dwh=DWH_EXPORT` then you add the following connection definitions to the JSON file:

```
{
   "DWH_EXPORT":

       { "dbname": "DWH", "schema": "DBO" }
}
```

5. Add a new section for Informatica PowerCenter to the lineage harvester configuration file.

## Example of the connection_definitions.conf file

```
{
   "connectionDefinitions": {
       "oracle_source": {
           "dbname": "oracle-source-database-name1",
           "schema": "my Oracle source schema",
           "dialect": "oracle"
       },
       "oracle_target": {
           "dbname": "oracle-target-database-name2",
           "schema": "my other oracle target schema",
           "dialect": "oracle"
       }
   },
   "collibraSystemNames": {
       "databases": [
           {
               "dbname": "oracle-source-database-name1",
               "collibraSystemName": "oracle-system-name1"
           },
           {
               "dbname": "oracle-target-database-name2",
               "collibraSystemName": "oracle-system-name2"
           }
       ],
       "connections": [
           {
               "connectionName": "oracle-connection-name1",
               "collibraSystemName": "oracle-system-name1"
           },
           {
               "connectionName": "oracle-connection-name2",
               "collibraSystemName": "oracle-system-name2"
           }
```

```
            ]
        }
}
```

## Steps to create a technical lineage for SQL Server Integration Services

1. Create a local folder.
2. Export the SSIS files for which you want to create a technical lineage.

   > Tip You can export them directly from the SQL Server Integration Services repository or via Microsoft Visual Studio. For more information, see the SQL Server Integration Services documentation.

3. Store the SSIS files to your local folder. Typically, the folder contains the following files:
   - SSIS package files (DTSX), containing the SQL Server Integration Services source code.
   - Connection manager files (CONMGR), containing environment and connection information.
   - Parameter files (PARAMS), if applicable.

   > Note
   > - All files in this folder and subfolders are taken into account when you create a technical lineage. The lineage harvester automatically detects data sources in the SSIS files.
   > - Not all SSIS files are processed and shown in the technical lineage. The lineage harvester retrieves all of the SSIS package files from the server, but only the files that contain lineage information, meaning those that contain a data flow, or Pipeline, are processed.

4. Optionally, configure the connection definitions:

   > Tip If the `useCollibraSystemName` in the lineage harvester configuration file is set to `true`, you must provide the connection_definitions.conf file.

   a. Create a new JSON file in the local folder.
   b. Name the JSON file *connection_definitions.conf*.

c. For each supported data source, specify the relevant translations.

| Property | Description |
| --- | --- |
| ConnStringRegExTranslation | The parent element that opens the connection definitions. |

| Property | Description |
| --- | --- |
| <regular expression> | A regular expression that must match one or more connection strings.<br><br>Note<br>Important considerations:<br>■ By default, the regular expression is not case sensitive. As a consequence, a regular expression can match with connection strings containing uppercase characters or lowercase characters.<br>■ The connection string is part of the SSIS connection manager.<br>■ SSIS connection managers are included in an SSIS package files (DTSX) or in connection manager files (CONMGR). |

| Property | Description |
| --- | --- |
| | Example<br>Regular expression: `Server=sb-dhub;User ID=SYB_USER2;Initial Catalog=STAGEDB;Port=6306.*`<br>Explanation: The first section, up to .*, is a literal, but not case-sensitive, match of the characters. The dot (.) can match any single character. The asterisk (*) means zero or more of the previous, in this case any character.<br>Match: Any connection string that starts with `Server=sb-dhub;User ID=SYB_USER2;Initial Catalog=STAGEDB;Port=6306.`<br>Example: `Server=sb-dhub;User ID=SYB_USER2;Initial Catalog=STAGEDB;Port=6306;Persist Security Info=True;Auto Translate=False;.` |
| dbname | The name of your database, to which the data source connection refers. |
| schema | The name of your schema, to which the regular expression refers. |

| Property | Description |
| --- | --- |
| dialect | The dialect of the referenced database.<br><br>Tip<br>You can enter one of the following values:<br><br>■ *azure*, for an Azure SQL Server data source.<br>■ *bigquery*, for a Google BigQuery data source.<br>■ *db2*, for an IBM DB2 data source.<br>■ *hana*, for a SAP Hana data source.<br>■ *hana-cviews*, for SAP Hana data calculation views.<br>■ *hive*, for a HiveQL data source.<br>■ *greenplum*, for a Greenplum data source.<br>■ *mssql*, for a Microsoft SQL Server data source.<br>■ *mysql*, for a MySQL data source.<br>■ *netezza*, for a Netezza data source.<br>■ *oracle*, for an Oracle data source.<br>■ *postgres*, for a PostgreSQL data source.<br>■ *redshift*, for an Amazon Redshift data source.<br>■ *snowflake*, for a Snowflake data source.<br>■ *spark*, for a Spark SQL data source.<br>■ *sybase*, for a Sybase data source.<br>■ *teradata*, for a Teradata data source. |

| Property | Description |
| --- | --- |
| collibraSystemName | The name of the referenced data source's system or server.<br><br>This property is only required when you set the useCollibraSystemName property in the lineage harvester configuration file to `true`. If this property is set to `false`, you can remove the collibraSystemName property or enter an empty string.<br><br>> Note You must use the same system name as the full name of the System asset that you create when you prepare the physical data layer in Data Catalog. If you don't prepare the physical data layer, Collibra Data Lineage cannot stitch the data objects in your technical lineage to the assets in Data Catalog.<br><br>If the “useCollibraSystemName" property is:<br><br>■ `false`, system or server names in table references in analyzed SQL code are now ignored. This means that a table that exists in two different systems or servers is identified (either correctly or incorrectly) as a single data object, with a single asset full name.<br><br>■ `true`, system or server names in table references are considered to be represented by different System assets in Data Catalog. The value of the "collibraSystemName" field is used as the default system or server name. |

5. Add a section for SQL Server Integration Services to the lineage harvester configuration file.

## Example of the connection_definitions.conf file

```
{
  "ConnStringRegExTranslation": {

    "Data Source=dhb-sql-prod;Initial Catalog=SFG_repl_
staging;Provider=SQLNCLI11;Integrated Security=SSPI.*": {
      "dbname": "DATAHUB",
      "schema": "DBO",
      "dialect": "mssql",
      "collibraSystemName" : "WAREHOUSE"
    },

    "Server=sb-dhub;User ID=SYS_USER;Initial
Catalog=STAGEDB;Port=6306.*": {
      "dbname": "STAGEDB",
      "schema": "STAGE_OWNER",
      "dialect": "sybase",
      "collibraSystemName" : ""
    }

  }
}
```

## Steps to create a technical lineage for DataStage

1. Create a local folder.
2. Export the DataStage project files (DSX) for which you want to create a technical lineage.

   > Tip You can either export a DataStage project manually or automatically via command line.

3. Store the DataStage files in your local folder.
4. Optionally, if your DataStage project uses environment variables, manually export the environment files (ENV).
5. Give the environment files the same name as the DataStage project files. For example, if your project file is named *datastage-project-1.dmx*, you have you name your environment file *datastage-project-1.env*.

6. Store the environment files in the same local folder.

> Important
> - The lineage harvester only supports DSX and ENV files.
> - You can have one DSX file per DataStage project.
> - You can have one or none ENV file per DSX file.
> - The name of the DSX file and the ENV file has to be the same.

7. Optionally, configure the connection definitions:
   a. Create a new JSON file in the local folder.
   b. Name the JSON file *connection_definitions.conf*.
   c. For each data source, specify the relevant translations:

| Property | Description |
|---|---|
| OdbcDataSources | Open Database Connectivity data sources in IBM InfoSphere DataStage for which you want to create a technical lineage. |
| <data-source-name> | The ODBC data source name that you use in your DataStage projects.<br><br>This section contains the properties to translate the database, schema and dialect. |
| dbname | The name of your database, to which the ODBC data source connection refers. |
| schema | The name of your schema, to which the ODBC data source connection refers. |

| Property | Description |
| --- | --- |
| dialect | The dialect of the referenced database.<br><br>Tip<br>You can enter one of the following values:<br><br>■ *azure*, for an Azure SQL Server data source.<br>■ *bigquery*, for a Google BigQuery data source.<br>■ *db2*, for an IBM DB2 data source.<br>■ *hana*, for a SAP Hana data source.<br>■ *hana-cviews*, for SAP Hana data calculation views.<br>■ *hive*, for a HiveQL data source.<br>■ *greenplum*, for a Greenplum data source.<br>■ *mssql*, for a Microsoft SQL Server data source.<br>■ *mysql*, for a MySQL data source.<br>■ *netezza*, for a Netezza data source.<br>■ *oracle*, for an Oracle data source.<br>■ *postgres*, for a PostgreSQL data source.<br>■ *redshift*, for an Amazon Redshift data source.<br>■ *snowflake*, for a Snowflake data source.<br>■ *spark*, for a Spark SQL data source.<br>■ *sybase*, for a Sybase data source.<br>■ *teradata*, for a Teradata data source. |

| Property | Description |
| --- | --- |
| collibraSystemName | The name of the data source's system or server.<br><br>This property is only required when you set the useCollibraSystemName property in the lineage harvester configuration file to `true`. If this property is set to `false`, you can remove the collibraSystemName property or enter an empty string.<br><br>> Note You must use the same system name as the full name of the System asset that you create when you prepare the physical data layer in Data Catalog. If you don't prepare the physical data layer, Collibra Data Lineage cannot stitch the data objects in your technical lineage to the assets in Data Catalog. |
| NonOdbcConnectors | Other data source connectors in IBM InfoSphere DataStage for which you want to create a technical lineage. For example, DB2, Oracle or Netezza.<br><br>> Note This section is optional. |

| Property | Description |
| --- | --- |
| <data-source-connector-ID> | The data source username and database of the connector that you use in your DataStage projects. This usually looks like for example *admin@database-name*. The combination of the username and database name should be unique.<br><br>The following section contains the properties to translate the database, schema and dialect. |
| dbname | The name of your database, to which the data source connection refers. |
| schema | The name of your schema, to which the data source connection refers. |

| Property | Description |
| --- | --- |
| dialect | The dialect of the referenced database.<br><br>Tip<br>You can enter one of the following values:<br><br>■ *azure*, for an Azure SQL Server data source.<br>■ *bigquery*, for a Google BigQuery data source.<br>■ *db2*, for an IBM DB2 data source.<br>■ *hana*, for a SAP Hana data source.<br>■ *hana-cviews*, for SAP Hana data calculation views.<br>■ *hive*, for a HiveQL data source.<br>■ *greenplum*, for a Greenplum data source.<br>■ *mssql*, for a Microsoft SQL Server data source.<br>■ *mysql*, for a MySQL data source.<br>■ *netezza*, for a Netezza data source.<br>■ *oracle*, for an Oracle data source.<br>■ *postgres*, for a PostgreSQL data source.<br>■ *redshift*, for an Amazon Redshift data source.<br>■ *snowflake*, for a Snowflake data source.<br>■ *spark*, for a Spark SQL data source.<br>■ *sybase*, for a Sybase data source.<br>■ *teradata*, for a Teradata data source. |

| Property | Description |
| --- | --- |
| collibraSystemName | The name of the data source's system or server.<br><br>This property is only required when you set the useCollibraSystemName property in the lineage harvester configuration file to `true`. If this property is set to `false`, you can remove the collibraSystemName property or enter an empty string.<br><br>You must use the same system name as the full name of the System asset that you create when you prepare the physical data layer in Data Catalog. If you don't prepare the physical data layer, Collibra Data Lineage cannot stitch the data objects in your technical lineage to the assets in Data Catalog. |

8. Add a section for IBM InfoSphere DataStage to the lineage harvester configuration file.

## Example of the connection_definitions.conf file

```
{
  "OdbcDataSources": {
    "oracle-data-source": {
      "dbname": "my-oracle-database",
      "schema": "my-oracle-schema",
      "dialect": "oracle",
      "collibraSystemName": "my-system"
    },
    "mssql-data-source": {
      "dbname": "my-mssql-database",
      "schema": "my-mssql-schema",
      "dialect": "mssql",
      "collibraSystemName": "my-system"
    }
  },
```

```
  "NonOdbcConnectors": {

    "admin@database-name": {
      "dbname": "my-netezza-database",
      "schema": "my-netezza-schema",
      "dialect": "netezza",
      "collibraSystemName": "my-system"
    },
    "admin@second-database-name": {
      "dbname": "my-second-netezza-database",
      "schema": "my-second-netezza-schema",
      "dialect": "netezza",
      "collibraSystemName": "my-system"
    }
  }
}
```

### What's next

You can now prepare the rest lineage harvester configuration file and run it to create a technical lineage for Informatica PowerCenterSQL Server Integration ServicesIBM InfoSphere DataStage and, optionally, other data sources.

When you run the lineage harvester, the content in your local folder is sent to the Collibra Data Lineage service for processing.

> Note For more information about the scope, see the overview of supported data sources.

## Download SQL files to the lineage harvester folder

You can download the SQL files of a data source that is stored locally and cannot be accessed via the network. The lineage harvester then stores the data source information in a ZIP file.

To create a technical lineage for these data sources, you only have to include the ID of the data source and the path to the ZIP file in the configuration file.

> Note Click here to see a list of all supported data sources.

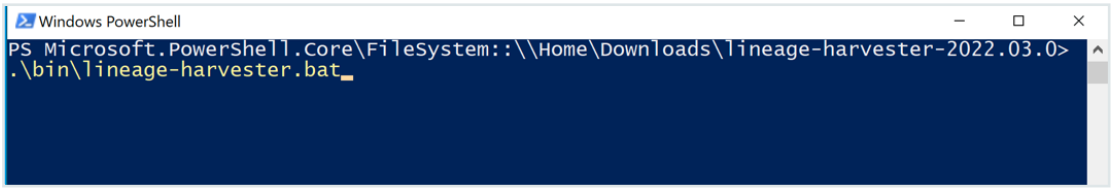
## Prerequisites

- You have downloaded the lineage harvester and you have the necessary system requirements to run it.
- You have the necessary permissions to all database objects that the lineage harvester accesses.

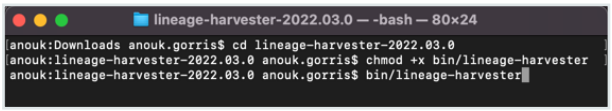
> Note For a detailed overview of the permissions that you need to access the data objects of your data sources, see the online user guide.

## Steps

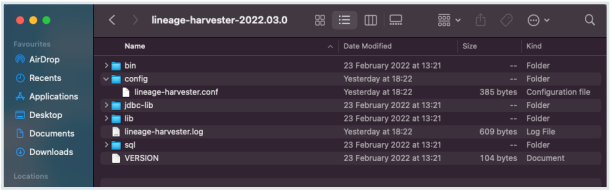
1. Run the following command line to start the lineage harvester:
   - Windows: `.\bin\lineage-harvester.bat`




   - For other operating systems: `chmod +x bin/lineage-harvester` and then `bin/lineage-harvester`




   » An empty configuration file is created in the config folder.




2. Save the configuration file in the **config** directory in the lineage harvester folder.
3. Prepare the configuration file.

> Tip Use the configuration file generator to easily create a configuration file.

4. When prompted, enter the passwords to connect to Collibra and your data sources. Do one of the following:

- Enter the passwords in the console.
  - » The passwords are encrypted and stored in **/config/pwd.conf**.
- Provide the passwords via command line.
  - » The passwords are stored locally and not in your lineage harvester folder.

5. Start the lineage harvester again and do one of the following:
   - To download the SQL files of all data sources in the configuration file, run the following command:

     ```
     ./bin/lineage-harvester load-sources
     ```

   - To download the SQL files of specific data sources in the configuration file, run the following command:

     ```
     ./bin/lineage-harvester load-sources -s "ID of the data source"
     ```

     > Tip This command allows you to download specific SQL files in the configuration file, without refreshing other SQL files. This reduces the time you need to download your SQL files, since you only download specific ones without affecting the others. If you want to download SQL files of multiple data sources, add `-s "ID of another data source"` per data source to the command.

     » The lineage harvester downloads the SQL files of the data sources and stores them in a ZIP file per data source in the lineage harvester output folder.

## What's next?

You can now prepare a configuration file for theSQL files of data sources that you want to include in your technical lineage.

# Prepare Informatica Intelligent Cloud Services <source ID> configuration file

You use the lineage harvester configuration file to access Informatica Intelligent Cloud Services Data Integration data objects. The lineage harvester processes the data objects to create a technical lineage. You also have to prepare a specific <source ID> configuration file that defines the Intelligent Cloud Services system name.

> Important You must prepare a <source ID> configuration file regardless of whether the `useCollibraSystemName` property in your lineage harvester configuration files is set to *true* or *false*.

## Prerequisites

You have Admin permission on all objects that you want to harvest.

## Steps

1. Create a new JSON configuration file in the lineage harvester **config** folder.
   If you have a data source with a large size for an Informatica Intelligent Cloud Services connection, consider creating more than one JSON file for the data source. Each JSON file must have a unique name. The contents in the JSON files are the same. In this way, you can avoid errors that might occur when the lineage harvester ingests metadata from one source with a large size.
2. Give the JSON file the same name as the value of the `Id` property in the lineage harvester configuration file.

   > Example If the value of the `Id` property in your lineage harvester configuration file is `iics-source-1`, then the name of your JSON file should be *iics-source-1.conf*.

   > Important Your JSON file must have the file extension *.conf*.
3. For each Informatica Intelligent Cloud Services connection, you can add the following content to the JSON file:

| Property | Description |
|---|---|
| collibraSystemNames | This section contains the system information for Informatica Intelligent Cloud Services. |

| Property | Description |
|---|---|
| connections | This section contains the system connection information. This is required to reference to the system or server of the connection. |
| connectionName | The name of the connection. |
| collibraSystemName | The system or server name of the connection. |
| connectionDefinitions | This section contains the database, schema and dialect information for each connection in Informatica Intelligent Cloud Services.<br><br>Note You can add connection information for each connection in the `connections` section. |
| connectionName | The name of the connection. The name must match with the name in a connection name in the `connections` section. |
| databaseName | The name of your database. |
| schemaName | The name of your schema. |

| Property | Description |
| --- | --- |
| dialect | The dialect of the connection.<br><br>You can enter one of the following values:<br><br>◦ *bigquery*<br>◦ *db2*<br>◦ *hana*<br>◦ *hive*<br>◦ *greenplum*<br>◦ *mssql*<br>◦ *mysql*<br>◦ *netezza*<br>◦ *oracle*<br>◦ *postgres*<br>◦ *redshift*<br>◦ *snowflake*<br>◦ *spark*<br>◦ *teradata* |

4. Save the configuration file.

## Example of the <source-ID>.conf file

```
{
   "collibraSystemNames": {
       "connections": [
           {
               "connectionName": "DG_con_standby_cmdm_clientors",
               "collibraSystemName": "PUBLIC"
           },
           {
               "connectionName": "DG_con_dev_dg_dgiauser_su",
               "collibraSystemName": "PUBLIC"
           }
       ]
   },
   "connectionDefinitions": [
       {
           "connectionName": "DG_con_standby_cmdm_clientors",
           "databaseName": "main",
           "schemaName": "dbo",
```

```
            "dialect": "oracle"
        },
        {
            "connectionName": "DG_con_dev_dg_dgiauser_su",
            "databaseName": "main",
            "schemaName": "dbo",
            "dialect": "oracle"
        }
    ]
}
```

# Prepare Matillion <source ID> configuration file

You use the lineage harvester configuration file to access Matillion data objects. The lineage harvester processes the data objects to create a technical lineage. However, if the `useCollibraSystemName` property in the lineage harvester configuration file is set to `true`, you also have to provide a <source ID> configuration file to define the system name for all sources and targets in the Matillion integration.

This is useful if you have multiple databases with the same name and want to distinguish between them in the technical lineage harvester by specifying the system or server specific to each.

> Note To preserve stitching, you need a System asset in Data Catalog of the same name of each system or server you specify in your <source ID> configuration file.

## Prerequisites

- The `useCollibraSystemName` in the lineage harvester configuration file is set to `true`.
- You have Admin permission on all objects that you want to harvest.

## Steps

1. Create a new JSON configuration file in the lineage harvester **config** folder.
2. Give the JSON file the same name as the value of the `Id` property in the lineage harvester configuration file.

> Example If the value of the `id` property in the lineage harvester configuration file is `matillion-source-1`, then the name of your JSON file should be *matillion-source-1.conf*.

> Important Your JSON file must have the file extension *.conf*.

3. For each Matillion connection, you can add the following content to the JSON file:

| Property | Description | Mandatory? |
|---|---|---|
| collibraSystemNames | This section contains the system information for Matillion. | Yes |
| sources | Use this section to define the system names of all sources in the Matillion job. | Yes |
| jobName | The Matillion job name. | Yes |
| collibraSystemName | The name of the Matillion source system or server. | Yes |
| targets | Use this section to define the system names of all targets in the Matillion job. | Yes |
| jobName | The Matillion job name. | Yes |
| collibraSystemName | The name of the Matillion target system or server. | Yes |

4. Save the configuration file.

## Example of the <source-ID>.conf file

```
{
   "collibraSystemNames":
```

```
        {
            "sources":[
                {
                    "jobName":"<name of job>",
                    "collibraSystemName":"<name>"
                },
                {
                    "jobName":"<name of job>",
                    "collibraSystemName":"<name>"
                }
            ],
            "targets":[
                {
                    "jobName":"<name of job>",
                    "collibraSystemName":"<name>"
                },
                {
                    "jobName":"<name of jobv",
                    "collibraSystemName":"<name>"
                }
            ]
        }
}
```

## Supported source system dialects

The dialect property in the lineage harvester configuration file refers to the target system. The following table shows the supported source system dialects.

| Matillion source dialect | Technical lineage source dialect |
|---|---|
| Amazon Redshift | redshift |
| Redshift | redshift |
| IBM DB2 | db2 |
| IBM DB2 for i | db2 |
| Microsoft SQL Server | mssql |
| MySQL | mysql |
| Netezza | netezza |

| Matillion source dialect | Technical lineage source dialect |
|---|---|
| Oracle | oracle |
| PostgreSQL | postgres |
| SAPHana | hana |
| Snowflake | snowflake |
| SQL Server (Microsoft Driver) | mssql |
| Sybase ASE | sybase |
| Teradata | teradata |

# Using a custom technical lineage

You can create a custom technical lineage to include metadata of data sources that the lineage harvester does not support or add functionality that is not supported.

To create a custom technical lineage, define the custom technical lineage in a JSON file and refer to the JSON file in the lineage harvester configuration file. The lineage harvester generates a technical lineage based on your definition in the JSON file. You can create the following custom technical lineages:

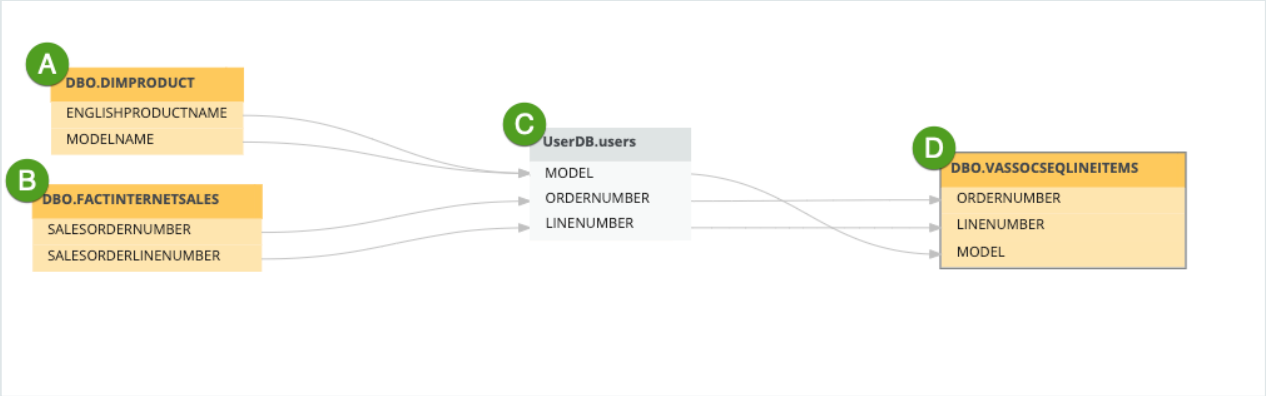
- A simple custom technical lineage, which defines a basic object hierarchy and creates a lineage between two or more data objects.
- An advanced custom technical lineage, which contains a simple custom technical lineage and uses separate source code files that define lineage transformations to create the lineage.

You can use the custom technical lineage as your only lineage source. You can also combine custom technical lineage with other lineage sources. For example, you can configure the lineage harvester to collect data objects from Oracle, Tableau and the custom technical lineage definition in the JSON file.

For the steps to create a custom technical lineage and information about configuring the JSON file, see Create custom technical lineage and Define a custom technical lineage in a JSON file.

## Example

You want to create a technical lineage that shows relations between tables and columns from system A and system B, to system C, to system D (A and B -> C -> D). System A, B and D are supported data sources, but system C is a custom application. You can create a JSON file that contains the metadata of system C and generate the following [[[Undefined variable technical-lineage.technlingraphlc]]].




## Create a custom technical lineage

To create a custom technical lineage, create and configure a JSON file that defines the custom technical lineage. Then add the properties of the JSON file to the lineage harvester configuration file.

### Prerequisites

Ensure that you've completed the following tasks:

- To stitch the data objects of your data sources with Data Catalog assets, prepare the Data Catalog physical data layer for technical lineage. When you prepare the Data Catalog physical data layer, you must register your data sources in Data Catalog and use a structure that matches the structure of ingested assets in Data Catalog.

- Determine whether you want to create a simple custom technical lineage or advanced custom technical lineage.

Ensure that you have necessary permissions for all data objects that the lineage harvester accesses. For more information, see data objects.

### Steps

1. Create a local folder.
2. Create a JSON file in the local folder and name the JSON file **lineage.json**. There must be only one JSON file in the folder; you can have other types of files in this folder. The JSON file must be named as **lineage.json**; otherwise, the process fails.
3. If you want to create an advanced custom technical lineage, store all of the source code files that you want to reference in the JSON file in the same local folder.
4. Configure the JSON file to define a simple or an advanced custom technical lineage.
5. Add a path to the folder of the JSON file in the lineage harvester configuration file.

### What's next

Complete preparing the lineage harvester configuration file and run the lineage harvester.

For more information about the commands that you can use to run the technical lineage, see Lineage harvesting app command options and arguments.

## Custom technical lineage JSON file

In the **lineage.json** file, you can define a basic data object hierarchy, a lineage between two or more data objects and transformations that create the custom technical lineage.

The following sections in the JSON file define different parts in the resulting Collibra [[[Undefined variable technical-lineage.technlingraphlc]]]:

- `tree`, which defines the data object hierarchy. The data objects are shown as nodes in the [[[Undefined variable technical-lineage.technlingraphlc]]].

- `lineages`, which defines the lineage relation. The lineage relations are shown as edges in the [[[Undefined variable technical-lineage.technlingraphlc]]]. The edges represent the data flow from a source to a target.
- `codebase_files`, which points to transformation definitions in a source code file.

If you want to create a simple custom technical lineage, specify the `tree` and `lineages` sections. You can add the transformation code in the `lineages` section.

If you want to create an advanced custom technical lineage, specify the `tree`, `lineages` and `codebase_files` sections. Add references to transformation code in source code files in the `codebase_files` section.

Transformation code in both simple and advanced custom technical lineages is displayed at the bottom part of the Collibra [[[Undefined variable technical-lineage.technlingraphlc]]].

### Sections in the `lineage.json` file

| Sections | Description |
|---|---|
| version | The version of the JSON architecture. Specify the value of `1.0`, which is the only supported version. |

| Sections | Description |
| --- | --- |
| tree | This section contains tree definitions of data objects between which lineages can be defined. The data objects are columns, tables, schemas, databases, systems, dashboards and reports.<br><br>Each node of a tree contains the name, type and optionally children or leaves properties which form a hierarchy of data objects. You must define a node only once in this section. With the nested tree format, you can reuse the properties of one node for multiple children. For example, you can define a database once and use the `children` array to define multiple tables in the database.<br><br>> Tip Usually, the structure you map is the following: system > database > schema > table > column. The system is optional, unless the `useCollibraSystemName` property is set to `true` in the lineage harvester configuration file. The Collibra Data Lineage can stitch these data objects to assets in Data Catalog. However, you can also map custom objects, for example dashboards and reports. Custom objects cannot be stitched to assets in Data Catalog. |
| lineages | This section contains the path from a source to a target and defines the transformation code or transformation references to be processed by the Collibra Data Lineage service. |
| codebase_files | This optional section defines the reference to source code files. Store the source code files that contain the transformation code in the same directory as the **lineage.json** file.<br><br>Include this section only when you create an advanced custom technical lineage. |

### tree section properties

| Properties | Description |
|---|---|
| name | The name of your data object. Specify this property with the system name, database name, schema name, table name or column name.<br><br>The following rules apply when you specify this property:<br><br>• The names are case sensitive.<br>• The names of children and leaves can be identical if the children and leaves with the same names are in different parent nodes. |
| type | The type of your data object. You can specify one of the following options: `system`, `database`, `schema`, `table`, `column`, `dashboard` or `report`. |
| children | The sub-objects that have a hierarchical relation to the defined data object.<br><br>Each child has the `name` and `type` properties and can contain `children` properties, except for the penultimate child. The penultimate `children` property must contain the `leaves` property. The `leaves` property cannot contain a `children` property.<br><br>For example, you can use the `children` property to define a table and use the `leaves` properties to define columns that have a relation to the table node. |

| Properties | Description |
|---|---|
| leaves | The sub-objects of an object that is defined in a `children` property, but cannot have sub-objects of their own.<br><br>A technical lineage is defined as relations between leaf nodes of the tree.<br><br>The value of the `type` property of the `leaves` property must be `column` or `report`. Indirect and table-level technical lineages are not supported. For the workarounds to create a table level or indirect technical lineage, see Programming considerations. |

### lineage section properties

| Properties | Required | Description |
|---|---|---|
| src_path | Yes | The hierarchical path to the source data object. This data object is defined as a leaf in the `tree` section.<br><br>This property represents where the data comes from for a transformation. |
| trg_path | Yes | The hierarchical path to the target data object. This data object is defined as a leaf in the `tree` section.<br><br>This property represents where the data flows to. |

| Properties | Required | Description |
|---|---|---|
| <data objects> | Yes | An ordered array of data object names. This array is required to define the sub-objects of the `src_path` and `trg_path` properties.<br><br>Specify the array with the data object names that start from the top of the `tree` section and finish at a leaf node.<br><br>This example shows data objects that can be stitched: system > database > schema > table > column.<br><br>This example shows data objects that cannot be stitched: dashboard > report > column. |
| mapping | No<br><br>Simple custom technical lineage only | The mapping name. This property specifies a name for the transformation code. |
| source_code | No<br><br>Simple custom technical lineage only | The transformation code, which determines how the technical lineage is constructed.<br><br>The transformation code can be a descriptive string or a SQL statement that manipulates data. |
| mapping_ref | No<br><br>Advanced custom technical lineage only | This property contains the name of the mapping reference to the transformation code in source code files. This property also contains the position and length of the transformation code to be highlighted in the [[[Undefined variable technical-lineage.technlingraphlc]]]. |

| Properties | Required | Description |
|---|---|---|
| source_code | No<br><br>Advanced custom technical lineage only | The name of the source code file that contains the transformation code. The transformation code can be a SQL statement, code that manipulates data or a descriptive string.<br><br>The source code file must be in the same directory as the **lineage.json** file. |
| mapping | No<br><br>Advanced custom technical lineage only | The unique descriptor of a part of transformation code in a source code file that is in the same directory as the **lineage.json** file.<br><br>A source code file can contain different parts of transformation code that represent different data flows. This property indicates the referenced data flow.<br><br>The value of this property is the same as the value of the `mapping_refs` property in the `codebase_files` section. |

| Properties | Required | Description |
|---|---|---|
| codebase_pos | No<br><br>Advanced custom technical lineage only | The positions indicate a string of the transformation code in a source code file to be highlighted in the bottom part of the Collibra [[[Undefined variable technical-lineage.technlingraphlc]]]. The whole lines that include the transformation code are highlighted.<br><br>The string must be a subset of the string of the transformation code that is defined by the `pos_start` and `pos_len` properties of the `mapping_refs` property in the `codebase_files` section. |
| pos_start | No<br><br>Advanced custom technical lineage only | The start position of the string of the transformation code to be highlighted. The start position is in characters, not bytes.<br><br>The value must be equal to or greater than the value of the `pos_start` property of the `mapping_refs` property in the `codebase_files` section. |

| Properties | Required | Description |
|---|---|---|
| pos_len | No<br><br>Advanced custom technical lineage only | The length of the string of the transformation code to be highlighted. The length is in characters, not bytes.<br><br>Specify a value in the following range:<br><br>• Equal to or greater than 1.<br>• Less than or equal to the length of the string that is defined by the `pos_len` property of the `mapping_refs` property in the the `codebase_files` section.<br><br>For example, if you specify `"pos_start": 10` and `"pos_len": 160` in the `codebase_files` section, specify a value for this property in the range of 0 - 149. |

### codebase_files section properties

| Properties | Description |
|---|---|
| <source code path> | The file path to source code files that contain the transformation code. The transformation code can be a SQL statement or code that manipulates data.<br><br>The source code file must be in the same directory as the **lineage.json** file. |

| Properties | Description |
|---|---|
| mapping_refs | The mapping of the transformation code and the position of the transformation code that is shown in the bottom part of the [[[Undefined variable technical-lineage.technlingraphlc]]].<br><br>This property defines a string of the transformation code in the source code file to be shown in the [[[Undefined variable technical-lineage.technlingraphlc]]]. The string must include the string that is defined by the `pos_start` and `pos_len` properties of the `mapping` property in the `lineage` section. |
| <mapping> | The unique descriptor of a part of transformation code in a source code file that is in the same directory as the **lineage.json** file.<br><br>A source code file can contain different parts of transformation code that represent different data flows. This property indicates the referenced data flow.<br><br>The value must match the value of the `mapping` property in the `lineage` section. |
| pos_start | The start position of the string of the transformation code. The start position is in characters, not bytes.<br><br>Specify a value in the following range:<br><br>• Equal to or greater than 0.<br>• Less than or equal to the value of the `pos_start` property in the `mapping` property in the `lineage` section. |

| Properties | Description |
|---|---|
| pos_len | The length of the string of the transformation code. The length is in characters, not bytes.<br><br>Specify a value in the following range:<br><br>• Greater than or equal to 1.<br>• Less than or equal to the length of the source code file minus the start position.<br><br>For example, if you specify `"pos_start": 10` and the file length is 160 characters, specify a value for this property in the range of 1 - 150. |

## Programming considerations

- As a workaround, you can specify `"type": "column"` and `"name": "*"` for the `leaves` property to create a table level or indirect technical lineage.
  With this specification, the indirect technical lineage is shown as a solid line instead of a dashed line in the Collibra [[[Undefined variable technical-lineage.technlingraphlc]]].
- The source code files must be in the same directory as the **lineage.json** file. Otherwise, an error occurs indicating that the lineage harvester cannot find the source code files.

For sample JSON files that define a simple custom technical lineage and an advanced custom technical lineage, see Custom technical lineage JSON file example.

# Custom technical lineage JSON file examples

This topic shows sample **lineage.json** files that create a simple custom technical lineage and an advanced custom technical lineage.

Each sample can be used to generate [[[Undefined variable technical-lineage.techlingraphlcs]]] in Collibra to represent the IOT_JSON and IOT_DEVICES_PER_COUNTRY tables with the following columns:

| IOT_JSON | IOT_DEVICES_PER_COUNTRY |
|---|---|
| CCA3 | COUNTRY |
| DEVICE_ID | NUMBER_DEVICES |

## Sample JSON file for a simple custom technical lineage

In the following sample **lineage.json** file, the `tree` section defines the IOT_JSON and IOT_DEVICES_PER_COUNTRY tables and columns. In this sample, the tables are in a schema named COLLIBRA. The COLLIBRA schema is in a database named COLLIBRA and a system named Databricks.

To show the transformation code at the bottom of the Collibra[[[Undefined variable technical-lineage.technlingraphlc]]] that uses a simple custom technical lineage, specify the `mapping` and `source_code` properties in the `lineages` section.

```
{ "version": "1.0",
  "tree": [
   { "name": "Databricks", "type": "system",
     "children": [
     { "name": "COLLIBRA", "type": "database",
       "children": [
           { "name": "COLLIBRA", "type": "schema",
         "children": [
         { "name": "IOT_JSON", "type": "table",
           "leaves": [
           { "name": "CCA3", "type": "column"},
           { "name": "DEVICE_ID", "type": "column"}
           ] },
           { "name": "IOT_DEVICES_PER_COUNTRY", "type": "table",
             "leaves": [
               { "name": "COUNTRY", "type": "column"},
               { "name": "NUMBER_DEVICES", "type": "column"}
             ] }
           ] }
       ] }
  ] } ],
  "lineages": [
   {"src_path": [
     {"system": "Databricks"},
     {"database": "COLLIBRA"},
     {"schema": "COLLIBRA"},
     {"table": "IOT_JSON"},
     {"column": "CCA3"}
```

```
        ],
    "trg_path": [
     {"system": "Databricks"},
     {"database": "COLLIBRA"},
     {"schema": "COLLIBRA"},
     {"table": "IOT_DEVICES_PER_COUNTRY"},
     {"column": "COUNTRY"}
       ],
    "mapping": "dev_no_bat_per_country_view",
    "source_code": "INSERT INTO ... SELECT CCA3 AS COUNTRY...FROM IOT_
JSON"

}
```

## Sample JSON file for an advanced custom technical lineage

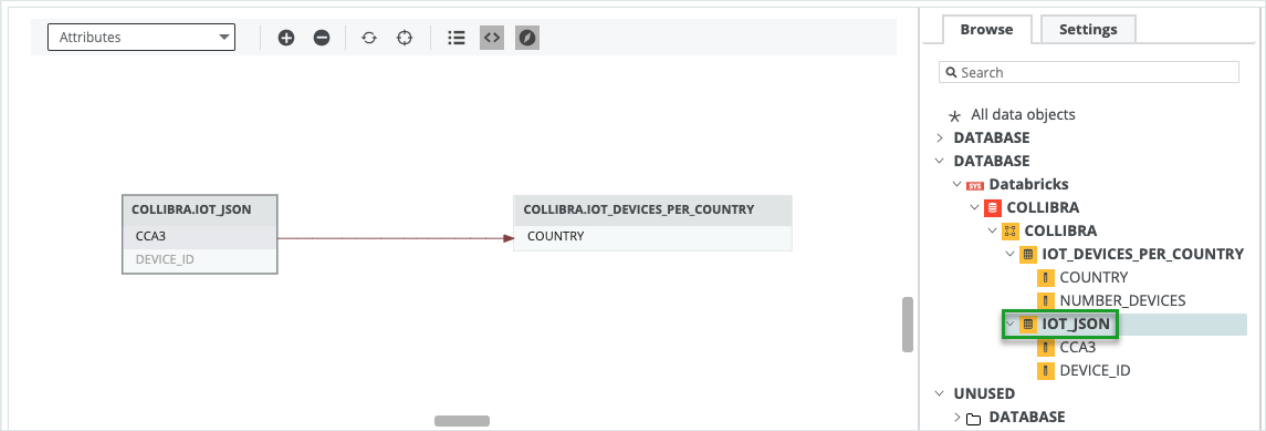
In the following sample **lineage.json** file, the `tree` section defines the IOT_JSON and IOT_DEVICES_PER_COUNTRY tables and columns. In this sample, the tables are in a schema named COLLIBRA. The COLLIBRA schema is in a database named COLLIBRA and a system named Databricks.

```
{ "version": "1.0",
  "tree": [
   { "name": "Databricks", "type": "system",
     "children": [
     { "name": "COLLIBRA", "type": "database",
       "children": [
           { "name": "COLLIBRA", "type": "schema",
         "children": [
         { "name": "IOT_JSON", "type": "table",
           "leaves": [
           { "name": "CCA3", "type": "column"},
           { "name": "DEVICE_ID", "type": "column"}
           ] },
           { "name": "IOT_DEVICES_PER_COUNTRY", "type": "table",
             "leaves": [
               { "name": "COUNTRY", "type": "column"},
               { "name": "NUMBER_DEVICES", "type": "column"}
             ] }
           ] }
       ] }
  ] } ],
  "lineages": [
   {"src_path": [
      {"system": "Databricks"},
      {"database": "COLLIBRA"},
      {"schema": "COLLIBRA"},
      {"table": "IOT_JSON"},
      {"column": "CCA3"}
```

```
        ],
      "trg_path": [
        {"system": "Databricks"},
        {"database": "COLLIBRA"},
        {"schema": "COLLIBRA"},
        {"table": "IOT_DEVICES_PER_COUNTRY"},
        {"column": "COUNTRY"}
       ],
        "mapping_ref": {
        "source_code": "transforms.sql",
        "mapping": "dev_no_bat_per_country_view",
        "codebase_pos": [ { "pos_start": 71, "pos_len": 69 } ]
       } }
      ],
   "codebase_files": {
     "transforms.sql": {
       "mapping_refs": {
         "dev_no_bat_per_country_view": {
         "pos_start": 0,
         "pos_len": 246
       } } } }
}
```

## Sample [[[Undefined variable technical-lineage.techlingraphlcs]]]

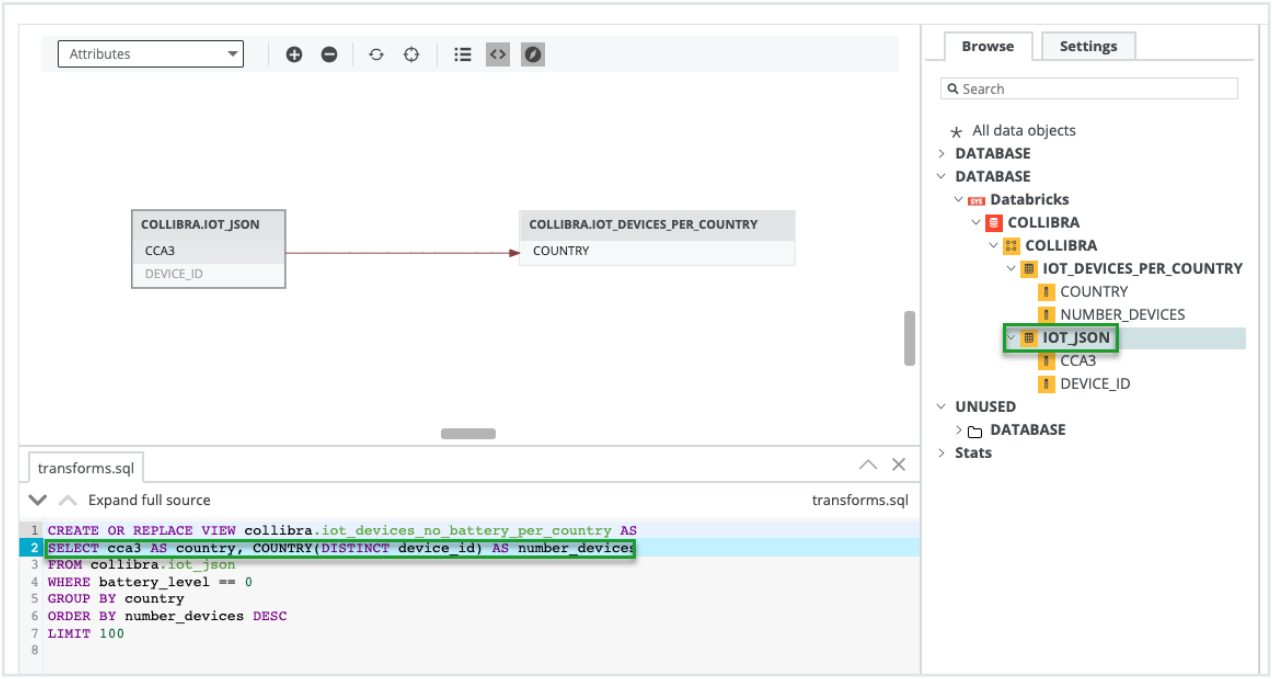
Both sample **lineage.json** files generate the following [[[Undefined variable technical-lineage.technlingraphlc]]], which contains 2 nodes and 1 edge.




The following [[[Undefined variable technical-lineage.technlingraphlc]]] is generated by using the sample **lineage.json** file for an advanced custom technical lineage. The bottom part shows the transformation code that generated the data flow.

In the `lineages` section, the `pos_start` property is specified with 71 and the `pos_len` property is specified with 69. The specifications indicate that the transformation code that

starts at position 71 and the following 69 characters are highlighted in blue. Line 2 in the [[[Undefined variable technical-lineage.technlingraphlc]]] contains the highlighted transformation code.




# Harvesting materialized views that were generated via an external script

The lineage harvester can harvest materialized views that are native to a data source—meaning the data flow is performed by SQL code stored in the data source. If, however, an external script is used to materialize views into tables, so to speak, they cannot be harvested by the lineage harvester. In this case, you could create a custom technical lineage, which requires a user-defined JSON file.

> Tip We recommend creating a script to generate a list of SQL queries to be harvested by the lineage harvester.

For each pair of source (view) and target (materialized view table), create a script as follows:

```
INSERT INTO 'dhw.sales.mv_customers'
   SELECT * FROM 'dhw.sales.v_customers';
```

The generated SQL queries then need to be harvested by the lineage harvester. There are two options for this, depending on where you choose to store the generated SQL code:

- If you store the SQL code in text files, it is harvested using an additional SqlDirectory type source.
- If you store the SQL code in a table in the data source, you need to modify the harvesting query, to harvest the table.
  In this case, actually, the generated SQL queries don't have to be stored anywhere; rather, they are generated on the fly by a harvesting query. Modify the harvesting query as follows:

  ```
  SELECT
     t.table_name,
     t.ddl as sourceCode,
     CONCAT(t.table_schema, '.', t.table_name) as groupName,
     t.table_schema as schemaName
  FROM `##PROJECT_ID##`.`##DSNAME##`.`INFORMATION_
  SCHEMA.TABLES` t
  WHERE t.table_type IN ('MATERIALIZED VIEW','VIEW')

  UNION ALL

  SELECT
     CONCAT('m', t.table_name),
     CONCAT('INSERT INTO `m', t.table_name, '` SELECT * FROM
  `', t.table_name, '`') as sourceCode,
     CONCAT('Generated m', t.table_schema, '.', t.table_name)
  as groupName,
     t.table_schema as schemaName
  FROM `##PROJECT_ID##`.`##DSNAME##`.`INFORMATION_
  SCHEMA.TABLES` t
  WHERE
     t.table_type IN ('VIEW')
     AND STARTS_WITH(t.table_name, 'v_')
  ```

  The second SELECT generates the necessary INSERT INTOs for all views in your data source that have a name starting with `v_`.

# Manage technical lineage ingestion

You can manage which data objects, for example columns and tables, are ingested in the technical lineage. By doing this, you can decide:

- Which data objects you want to visualize in the technical lineage.
- Between which columns you want to create new relations of the type "Data Element targets / sources Data Element" in Data Catalog.

You can manage technical lineage ingestion by creating a customized SQL file. In the SQL file, you can exclude data objects or change queries that are used to extract data from the database.

> Note
> - If you change queries, you can only use supported SQL syntax.
> - The customized SQL file is not supported.
> - The lineage harvester does not support proxy server authentication, but you can manually connect to a proxy server via command line. For more information, see Connecting to a proxy server.

## Steps

1. Open the lineage harvester folder.
2. Go to the **sql** folder and open the folder of the data source type of which you want to exclude tables or schemas or change queries.
3. Create a copy of the file you want to edit.
4. Rename the copy to *[original name]-custom.sql*.

   > Example You want to change the file columns.sql, so you name the copy of this file and rename it to *columns-custom.sql*.

5. Delete or edit the content of the new SQL file to include or exclude specific tables or schemas or change specific queries in the file.
6. Save the new SQL file.
   » The lineage harvester uses the new file and ignores the old one.

# Schedule jobs

You can use Task Scheduler on Windows or Crontab on Mac and Linux to make the lineage harvester run scheduled jobs at specific times, dates or intervals. In a scheduled job, the lineage harvester uploads data source information to the Collibra Data Intelligence Cloud and Data Catalog automatically creates new relations of the type "Data Element sources / targets Data Element"

- Between data objects in your data source and assets from registered data sources.
- Between ingested assets from BI sources and Data Catalog assets from registered data sources.

You can run one scheduled job for each data source that is listed in the same configuration file.

> Note If you provide the passwords to your Collibra environment and/or to your individual data sources via stdin, you have to use the correct command.

> Example You created a configuration file with two data sources. Data source A can run a scheduled job each day at 11 pm, while data source B can run a scheduled job every two days at 6 am.

# Delete the technical lineage of a data source

You can delete the technical lineage of a data source if you no longer want to see it in the technical lineage graph.

> Note You always need at least one source in your lineage harvester configuration file.

## Prerequisites

- You have a global role that has the Manage all resources global permission.
- You have a global role with the Technical lineage global permission.
- You have a global role with the Data Stewardship Manager global permission.

- You have downloaded the lineage harvester and you have the necessary system requirements to run it.
- Java Runtime Environment version 11 or newer or OpenJDK 11 or newer.
- You have added Firewall rules so that the lineage harvester can connect to:
  - The host names of all databases in the lineage harvester configuration file.
  - All Collibra Data Lineage service instances within your geographical location:
    - 15.222.200.199 (techlin-aws-ca.collibra.com)
    - 18.198.89.106 (techlin-aws-eu.collibra.com)
    - 54.242.194.190 (techlin-aws-us.collibra.com)
    - 51.105.241.132 (techlin-azure-eu.collibra.com)
    - 20.102.44.39 (techlin-azure-us.collibra.com)
    - 35.197.182.41 (techlin-gcp-au.collibra.com)
    - 34.152.20.240 (techlin-gcp-ca.collibra.com)
    - 35.205.146.124 (techlin-gcp-eu.collibra.com)
    - 34.87.122.60 (techlin-gcp-sg.collibra.com)
    - 35.234.130.150 (techlin-gcp-uk.collibra.com)
    - 34.73.33.120 (techlin-gcp-us.collibra.com)

> Note The lineage harvester connects to different instances based on your geographic location and cloud provider. If your location or cloud provider changes, the lineage harvester rescans all your data sources. You have to whitelist all Collibra Data Lineage service instances in your geographic location. In addition, we highly recommend that you always whitelist the techlin-aws-us instance as a backup, in case the lineage harvester cannot connect to other Collibra Data Lineage service instances.

## Steps

1. In the lineage harvester folder, open your lineage harvester configuration file.
2. Delete the section with connection properties of the data source you no longer want to see a technical lineage for.
3. Save the configuration file.
4. Start the lineage harvester again in the console and run the following command:
   - for Windows: `.\bin\lineage-harvester.bat full-sync`
   - for other operating systems: `./bin/lineage-harvester full-sync`
5. When prompted, enter the password to connect to your Collibra Data Intelligence Cloud and data sources in the configuration file.

» The lineage harvester uploads the metadata of the remaining data sources in the configuration file to the Collibra Data Lineage service.
» The Collibra Data Lineage service synchronizes the technical lineage and removes the deleted data source from the technical lineage graph.

# Technical lineage viewer

The technical lineage viewer shows the technical lineage and allows you to edit the view. You can access the technical lineage viewer via the Technical lineage tab on Column and Table asset pages and BI assets of the same level.

> Tip For more information about the technical lineage for Looker or Power BI, we highly advise you to read the dedicated sections in the user guide.

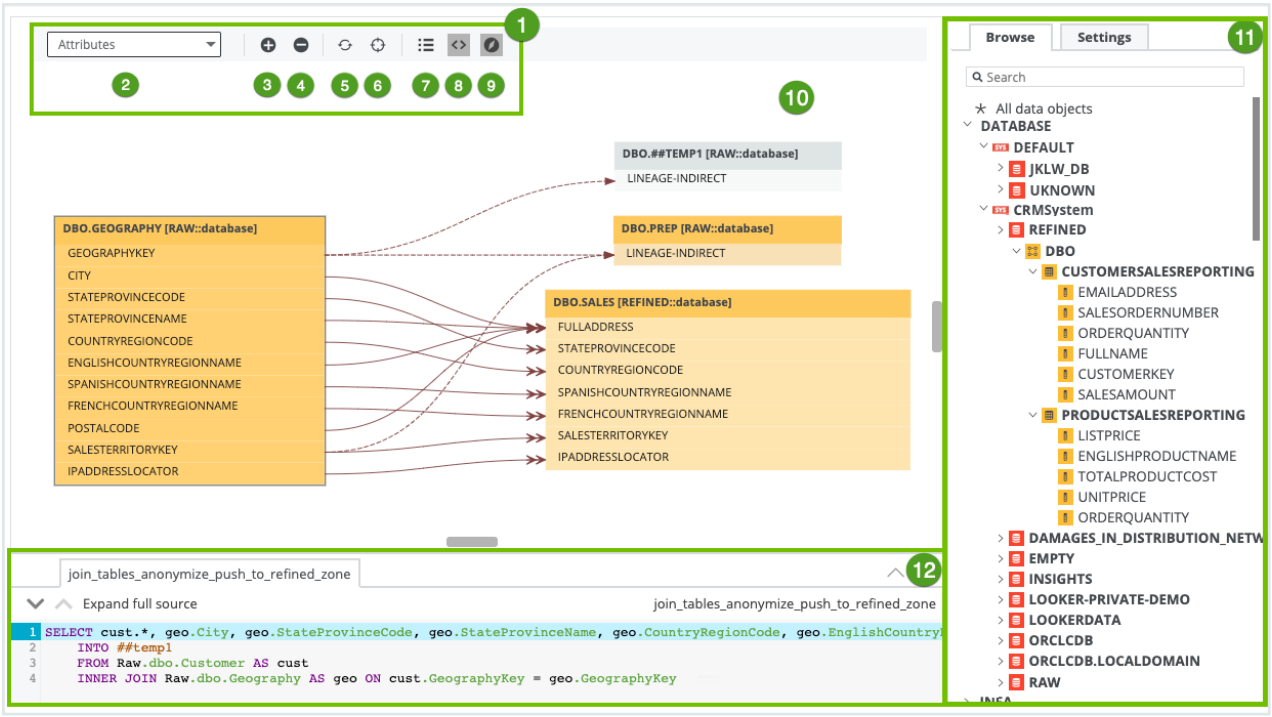
## Technical lineage tab

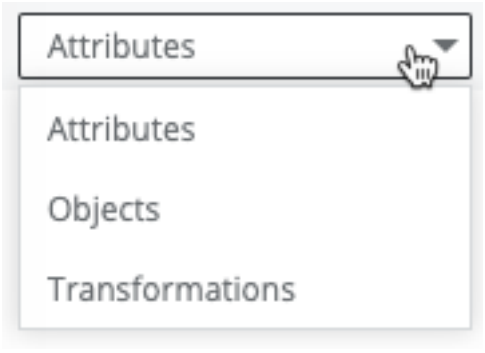
You can only see the Technical lineage tab on a Column or Table asset page when you have the following prerequisites:

- You have a global role with the Catalog global permission, for example Catalog Author.
- You have a global role with the Technical lineage global permission.

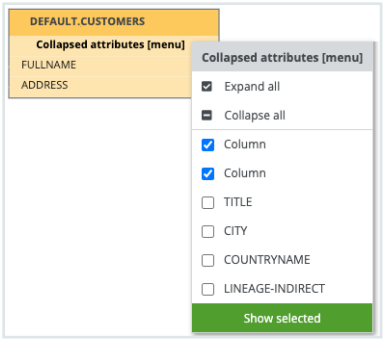
> Note View permissions are not enforced in Collibra Data Lineage. This means that anyone with the Technical Lineage global permission can see all of the assets in a technical lineage graph, regardless of their view permissions as determined at the community or domain level.

# Technical lineage viewer

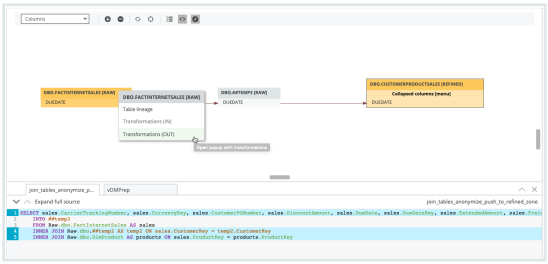



| No | Name | Description |
|---|---|---|
| 1 | Toolbar | The toolbar to work with technical lineage. The toolbar helps you to edit basic settings that apply to the entire lineage. |
| 2 |   | Drop-down list to determine which details (attributes, objects or trans-formations) you want to show in the technical lineage graph. |
| 3 | ⊕ | Button to zoom in on the technical lineage. |

| No | Name | Description |
|---|---|---|
| 4 |  | Button to zoom out on the technical lineage. |
| 5 |  | Button to refresh the technical lineage. This discards all the changes that you made to the technical lineage and restores it to the initial state. |
| 6 |  | Button to reposition the technical lineage to the starting position. |
| 7 |  | Button to show or hide the legend panel. |
| 8 |  | Button to show or hide the source code pane. |
| 9 |  | Button to show or hide the **Browse** and **Settings** tab panes. |

| No | Name | Description |
|---|---|---|
| 10 | Technical lineage graph | The actual visualization of the traceability of the current data object, according to your selection in the **Browse** tab pane.<br><br>If you select a specific column in a table with multiple columns, you can click **Collapsed columns [menu]** to show all other columns, collapse all columns or only show selected columns in the same table.<br><br>  <br><br><br>Tip Data objects that are stitched to assets in Data Catalog have a yellow background. Other data objects that the lineage harvester collected from your data source, but are not stitched and therefore are not assets in Data Catalog, have a gray background. |

| No | Name | Description |
|---|---|---|
| 11 | Tab panes | Tab panes that contain useful tools to browse through your technical lineage or determine which content is visualized in the technical lineage. |
| | **Browse** tab pane | This pane can be used to search for specific data objects or show statistics on the amount of tables and views in use. More information. |
| | **Settings** tab pane | This pane can be used to search for transformation code, edit the visualization of the technical lineage, see the status of the source code, check the stitching results or export your technical lineage to PDF, PNG or CSV. More information. |

| No | Name | Description |
|---|---|---|
| 12 | Source code pane | The source code pane shows the source code of specific data objects. It can be used to easily find issues in the data flow.<br><br>The source code pane is shown when you click <> in the toolbar or when you right-click a column or table and click **Transformations (IN)** or **Transformations (OUT)** which shows the transformation logic in the source code pane.<br><br>  |

## The technical lineage graph

The technical lineage graph consists of nodes and edges. Each node represents a corresponding object in a data source. Each edge shows a relation between nodes.

Nodes and edges in the technical lineage graph show how data flows from source to destination. Understanding the nodes and edges better, enriches your technical lineage experience.

Consider the following visual elements in the technical lineage graph:

- Relation types
- Messages
- Colors
- Icons
- Arrows

- Collapsed attributes menu
- Right-click menu

## Relation types

The technical lineage graph shows relations between columns in the graph. The Collibra Data Lineage creates and shows the following relation type between stitched assets and other data objects:

| Head | Role | Co-role | Tail | ID |
|---|---|---|---|---|
| Data Element | targets | sources | Data Element | 00000000-0000-0000-0000-000000007069 |

## Messages

The technical lineage graph might show different messages to alert you. The following messages are the most common:

| Message | Description |
|---|---|
| Nodes count exceeds the limit 350.<br><br>Edges count exceeds the limit 1,000. | The technical lineage graph exceeds the limit of 350 nodes or 1,000 edges and is too large to display. This happens, for example, if you have a table with many columns and you try to show the technical lineage of all columns in a table in one graph.<br><br>> Note You cannot manually change this limit. |
| The current asset doesn't have a technical lineage yet. | This message is shown if you didn't create a technical lineage for the data source of the asset.<br><br>Use the Browse tab pane to navigate through the data object for which a technical lineage graph is available. |

| Message | Description |
| --- | --- |
| Technical lineage cannot be shown. | The technical lineage graph cannot be shown, because there are too many data objects. This happens, for example, when you created a technical lineage for multiple data source and you click **All data objects** in the Browse tab pane.<br><br>Use the Browse tab pane to view specific parts of the technical lineage graph or click the suggested data objects to see their graph. |

## Colors

The technical lineage graph shows different colors to indicate which data objects are stitched to assets in Data Catalog and which are not.

### Background colors

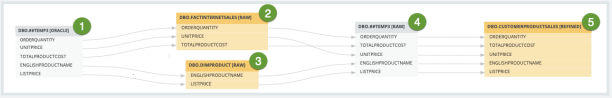
The background color of a node indicates whether or not the data object was stitched to an asset in Data Catalog, and whether something went wrong.

A node has one of three background colors:

| Color | Description |
| --- | --- |
| Yellow | Data objects from your data source that are stitched to assets in Data Catalog |
| Gray | Data objects, for example temporary tables and columns, that the lineage harvester collects from your data sources, but are not stitched to assets in Data Catalog.<br><br>> Note We do not support stitching for Looker or MicroStrategy assets. |

| Color | Description |
|---|---|
| Red | Attributes that are automatically assigned to a data object, because of missing DDL statements. If you want to remove objects with a red background, change the statements and rerun the lineage harvester. |

Since a technical lineage shows how data flows from source to destination, it is possible to see a lineage graph with both yellow, red and gray nodes.

> Example The following technical lineage graph shows two nodes with a gray background and three nodes with a yellow background. Node 1 and 4 contain data objects that are not stitched to assets in Data Catalog while nodes 2, 3 and 5 contain existing assets in Data Catalog that were stitched to the corresponding data objects when you created the technical lineage.
>
> 

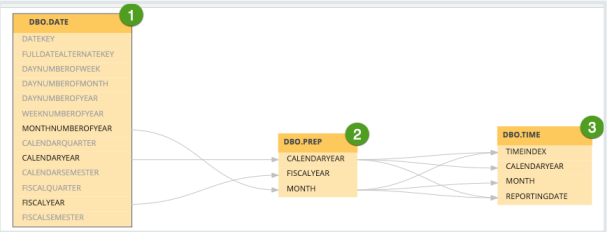
## Font colors

The font color of data objects in the technical lineage graph indicates whether or not there is a relation between this data object and one or more other data objects.

A node has one of two font colors:

| Color | Description |
|---|---|
| Black | At least one direct or indirect relation exists between the data object and another. |
| Gray | No relation exists between the data object and another. |

Example The following technical lineage graph shows three nodes. The node 1 contains data objects that have no incoming or outgoing edges to other data objects in the technical lineage. Nodes 2 and 3 only contain data objects that have a relation to other data objects in the technical lineage.

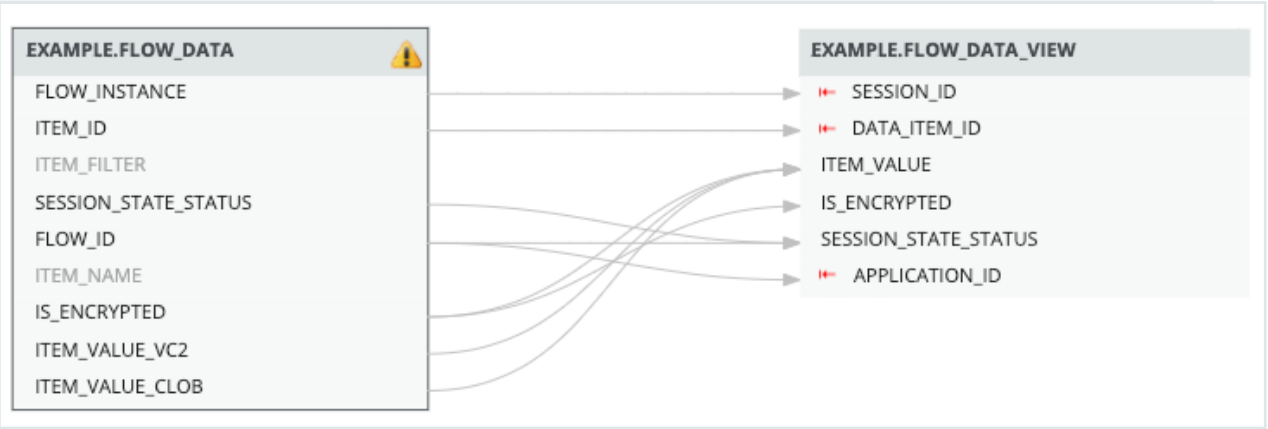



## Icons

Collibra uses various icons in the technical lineage graph.

| Icon | Description |
|---|---|
|  | The name of a table was found by the full-text search in the source code on which the analysis failed. Consequently, the lineage flow of the table is probably incomplete.<br><br>If you click **Show failed SQLs** on the right click menu of the table, the failed SQL queries appear in the source code pane at the bottom of the page. |
|  | The lineage is cyclic, for example A → B → C → A. It only appears if you enabled the only ending points option in the **Settings** tab pane. |
|  | A relation for the data objects exists, but it isn't shown, for example because you set the technical lineage flow depth to a lower value than the actual graph size. |

Example The following Technical lineage graph shows two nodes. The first node has an icon to indicate that the lineage flow you currently see is probably incomplete. The second node has three data objects that have a relation to other data objects, but the edges that represent that relation are not shown.




## Arrows

Arrows are incoming or outgoing edges that show how the data flows from source to destination. They represent relations of the type "Data Element sources / targets Data Element".

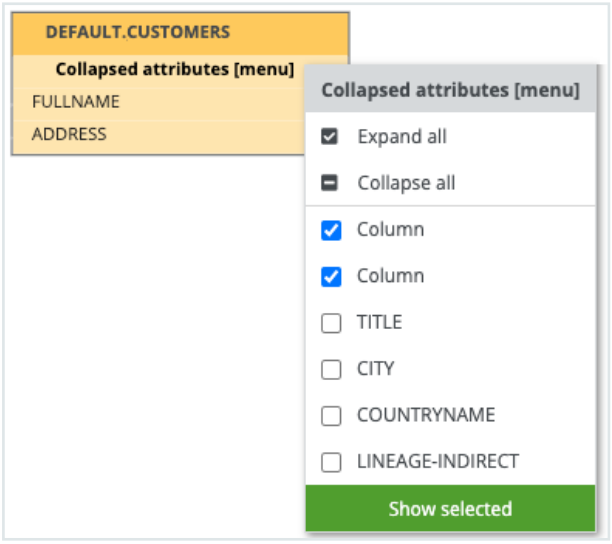
There are two ways in which an arrow can be shown:

| Arrow type | Description |
|---|---|
| Single | Shows the full lineage without skipping certain data objects. |
| Double | Shows that there are hidden data objects in the technical lineage graph. This happens when only the endpoints of the technical lineage flow are shown. |

Example The following Technical lineage graph shows three nodes. Edges with double arrows are shown between node 1 and 3. These edges indicate that there are other nodes between these nodes in the full technical lineage flow. Node 2 has outgoing edges with single arrows. These edges indicate that there is a direct relation between node 2 and 3.

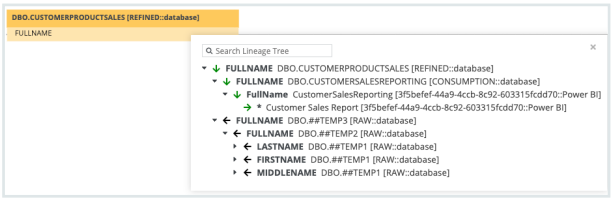

## Collapsed attributes menu

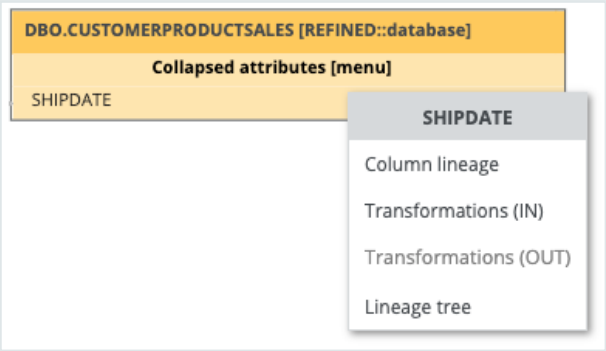
If you select a specific column in a table with multiple columns, you can click **Collapsed attributes [menu]** to show all columns, collapse all columns or only show selected columns in the same table.




## Right-click menu

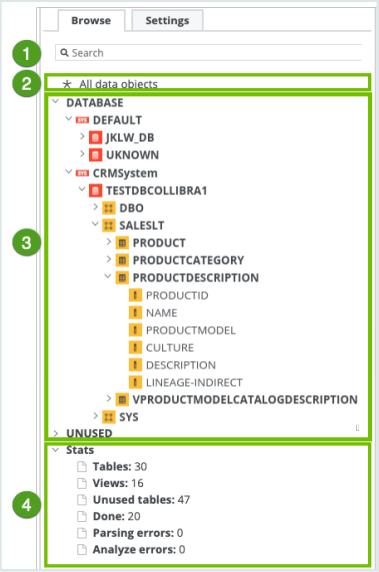
If you right-click a node, you can perform several specific actions on that node.

| Functionality | Description |
|---|---|
| Column/Table lineage | Switch to the technical lineage graph of the selected column or table. |
| Transformation (IN) | Show the transformation logic of the incoming source code fragments in the source code pane. |
| Transformation (OUT) | Show the transformation logic of the outgoing source code fragments in the source code pane. |
| Lineage tree | Show an alternative way to view the flow of data objects, called the lineage tree. The lineage tree is particularly useful if there are many nodes in a lineage. It enables you to see the entire lineage in one pop-up, which means you no longer have to scroll through the technical lineage graph to see the full lineage.<br><br>The lineage tree uses arrows to visualize the traceability of data objects:<br><br>• Green arrows represent outgoing edges.<br>• Black arrows represent incoming edges.<br><br>  |
| Custom features | When the lineage flow of the table is incomplete or there is an issue in the source code of a data object, the right-click menu shows the **Show failed SQLs** option. If you click this option, the source code pane opens and shows the SQL queries that failed. |




# Technical lineage Browse tab pane

The **Browse** tab pane allows you to navigate to and search for a specific data object within the technical lineage tree.



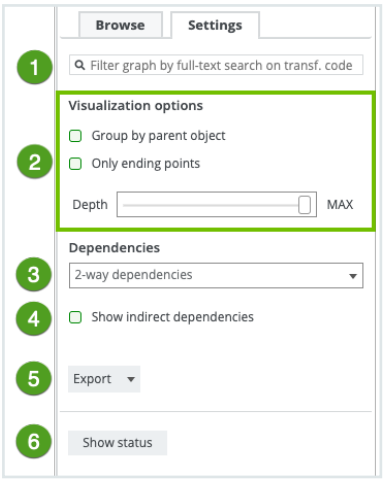

| No | Name | Description |
|---|---|---|
| 1 | Search | A search field that you can use to find a specific data object. |
| 2 | All data objects | A link to the complete technical lineage, showing all data objects in your data sources. |

| No | Name | Description |
|---|---|---|
| 3 | Navigation tree | A navigation tree in which you can search for specific data objects and visualize them in your technical lineage. The data objects are grouped by node type and have the following structure: system (if applicable) > database > schema > table > column.<br><br>Note The list of data objects contains all systems, databases, schemas, tables and columns that were collected from the data sources by the lineage harvester. If available, it also shows the technical lineage of BI sources, for example Power BI and Looker. In that case, the structure follows the existing structure in the BI source metadata.<br><br>Note The **UNUSED** branch contains data objects that were detected by Collibra Data Lineage, but are not included in any Technical lineage. |

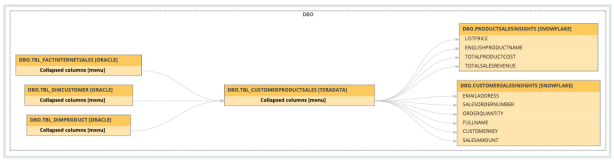
| No | Name | Description |
|---|---|---|
| 4 | Stats | Statistics that show which information is or is not visualized in the technical lineage. The statistics contain the following data:<br>• **Tables**: the amount of tables that are shown in the technical lineage.<br>• **Views**: the amount of views that are shown in the technical lineage.<br>• **Unused tables**: the amount of tables in your data source that are not shown in the technical lineage.<br>> Tip This metric is hidden when there are no unused tables.<br>• **Unused views**: the amount of views in your data source that are not shown in the technical lineage.<br>> Tip This metric is hidden when there are no unused views.<br>• **Done**: the amount of queries that were processed successfully.<br>• **Parsing errors**: the amount of queries with invalid or unidentified syntax.<br>• **Analyze errors**: the amount of columns that are not linked to a table. |

## Technical lineage Settings tab pane

The **Settings** tab pane allows you to edit the technical lineage, search for queries and export the technical lineage.




| No | Name | Description |
|---|---|---|
| 1 | Search field | A search field to find specific transformation code in a selected object or attribute. As you type, corresponding object names from the technical lineage appear in a drop-down list. If you press `Enter`, the technical lineage only shows the parts that contain your search word(s). |

| No | Name | Description |
|---|---|---|
| 2 | Visualization options | Options to define how you will see the data objects in the technical lineage. |
| | Group by parent object | Option to group tables and columns together by their hierarchical parent object.<br><br>Example A schema is the parent object of a table.<br><br>  |
| | Only ending points | Option to hide all data objects in the middle of the data flow and only show the ending points of the technical lineage. |
| | Depth | A slider that determines the maximum flow depth. The relation path length from the first node in the technical lineage graph to any other node is automatically adjusted to the maximum flow depth.<br><br>If you see in the technical lineage graph, the flow depth is set to a lower value than the actual graph size. |
| 3 | Dependencies | Drop-down to select the dependencies that you want to visualize. You can select one of the following dependencies:<br><br>• Inbound dependencies only<br>• Outbound dependencies only<br>• 2-way dependencies |
| 4 | Show indirect dependencies | Option to enable indirect dependencies. |

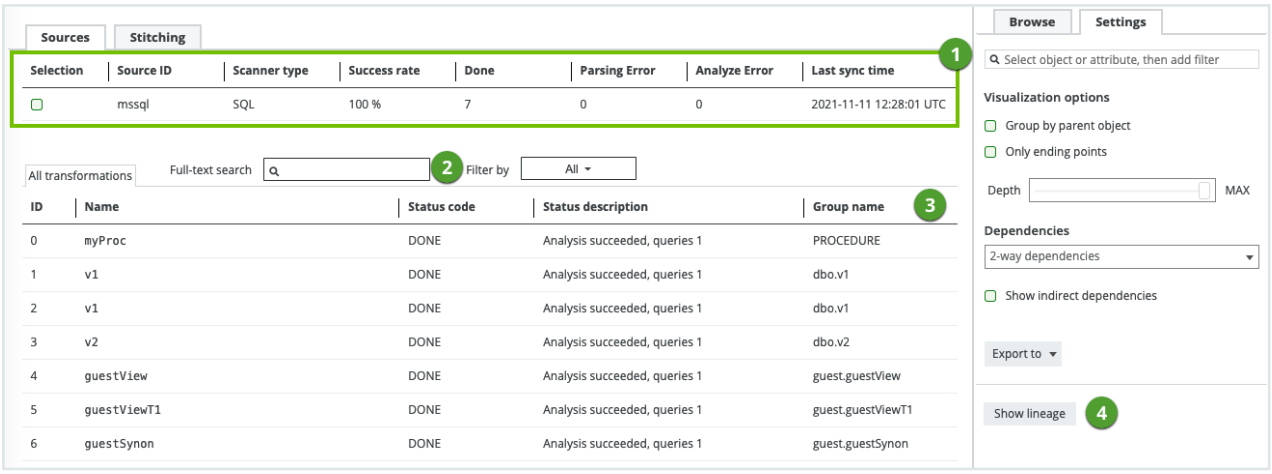
| No | Name | Description |
|---|---|---|
| 5 | Export | Button to export your technical lineage. You can choose among the following export types:<br>• PDF<br>• PNG<br>• CSV<br>• Full CSV<br>• JSON |
| 6 | Show status | Button to switch to the Sources tab page, which shows the analysis log files of your data sources and the Stitching tab page, which shows an overview of assets and data objects and shows which are stitched. |

## Technical lineage Sources tab page

When you create a configuration file and run the `full-sync` command in the lineage harvester, your data sources are uploaded to the Collibra Data Lineage service to be analyzed and processed. The Sources tab page shows the transformation details or source code that was analyzed and the results of this analysis.

You can access the Sources tab page by clicking **Show status** on the Settings tab pane.

> Note If an analyzed data source has the following result, the data source does not appear in the Sources tab page:
> ```
> Parsing errors: 0
> Analysis errors: 0
> Done: 0
> ```


Sources
Stitching
Selection
Source ID
Scanner type
Success rate
Done
Parsing Error
Analyze Error
Last sync time
mssql
SQL
100 %
7
0
0
2021-11-11 12:28:01 UTC
1
All transformations
Full-text search
2
Filter by
All
ID
Name
Status code
Status description
Group name
3
0 myProc DONE Analysis succeeded, queries 1 PROCEDURE
1 v1 DONE Analysis succeeded, queries 1 dbo.v1
2 v1 DONE Analysis succeeded, queries 1 dbo.v1
3 v2 DONE Analysis succeeded, queries 1 dbo.v2
4 guestView DONE Analysis succeeded, queries 1 guest.guestView
5 guestViewT1 DONE Analysis succeeded, queries 1 guest.guestViewT1
6 guestSynon DONE Analysis succeeded, queries 1 guest.guestSynon
Browse
Settings
Select object or attribute, then add filter
Visualization options
Group by parent object
Only ending points
Depth
MAX
Dependencies
2-way dependencies
Show indirect dependencies
Export to
Show lineage
4

| No | Name | Description |
|---|---|---|
| 1 | Summary per data source | A summary per data source. You can also select data sources to filter the results. |
| | Selected | Checkboxes to filter on a data source in the transformations table. If you select none, the transformations table contains all transformations. |
| | Source ID | The ID of your data source. You entered this ID in the configuration file. |
| | Scanner type | The type of scanner that is used to scan the queries in your data source. |
| | Success rate | The success rate of the data source analysis on the Collibra Data Lineage service. The success rate indicates how complete your technical lineage is.<br><br>Important The success rate of a technical lineage gives a good indication of the processing success. A success rate less than 100%, however, does not mean processing was unsuccessful. A parsing error, for example, which negatively affects the success rate, does not always negatively affect the completeness of the lineage. |
| | Done | The amount of queries that were scanned and analyzed. |
| | Parsing Error | The amount of parsing errors. |
| | Analyze Error | The amount of analysis errors. |
| | Last sync time | The last time the data source was uploaded to the Collibra Data Lineage service, for analysis and processing. |

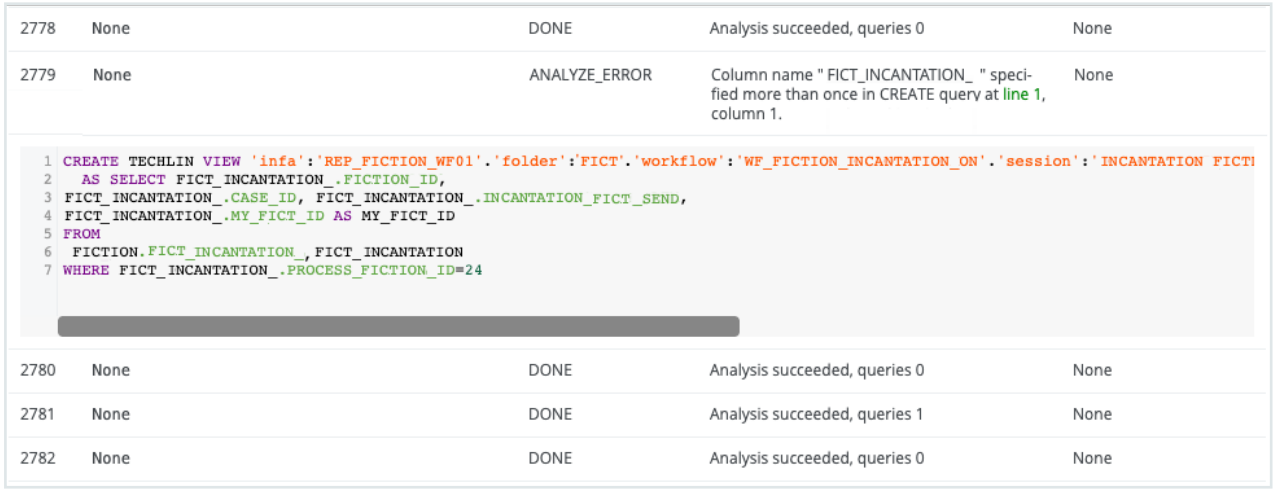
| No | Name | Description |
|---|---|---|
| 2 | Search tools | Tools to help you search for specific source code fragments. |
| | Full-text search | A search field to find specific queries in the log files. Type what you are looking for and press `Enter`. |
| | Filter by | A drop-down list to filter the source codes based on their status code. |

| No | Name | Description |
|---|---|---|
| 3 | Transformations table | The table that contains details of the transformations and source code (fragments).<br><br>You can filter the rows in the table by selecting data sources in the data source table and by using the search tools.<br><br>Tip If you click a source code fragment, you can see the log file attached to it. |
| | ID | The ID of the source code fragments or transformation details, which are assigned in chronological order. |
| | Name | The name of the specific source code fragment or transformation detail.<br><br>You can also see the source code fragment name in the source code pane in the technical lineage graph. |
| | Status code | The status of the analysis.<br><br>A source code fragment or transformation detail can have one of the following status codes:<br><br>• `DONE`: All queries are processed successfully.<br>• `ERROR`: Some queries could not be processed.<br>• `PARSING_ERROR`: The syntax of some queries is invalid or unidentified.<br>• `ANALYZE_ERROR`: Some columns are not linked to a table. |
| | Status description | The description of the status code that provides more information about the analysis and shows how many queries were processed. |

| No | Name | Description |
|---|---|---|
| | Group name | The name of the package or procedure to which the source code fragment or transformation details belongs. |
| 4 | Show lineage | Button to go back to the technical lineage graph. |

## Analysis results

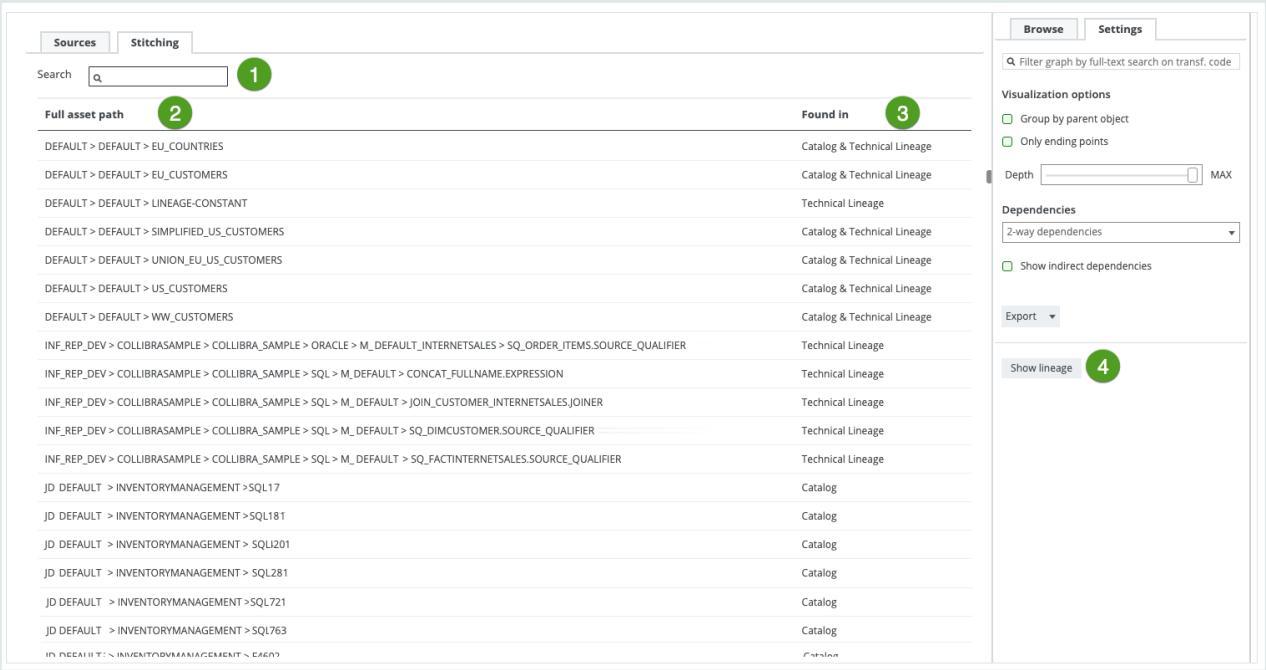
If you click one of the rows in the Transformations table, a file with the analysis results attached to the source code or transformation details opens. You can use these files to easily find errors in the source code or transformation details of your data source.




# Technical lineage Stitching tab page

The Stitching page shows the full path of assets in Data Catalog and data objects of the data sources for which you created a technical lineage. You can use it to easily check which assets are stitched and which are not.

You can access the Stitching tab page by clicking **Show status** on the Settings tab pane.




| No | Name | Description |
|---|---|---|
| 1 | Search field | A search field to find specific assets or data objects. Type what you are looking for and press `Enter`. |
| 2 | Full asset path | The full path to all data objects on the Collibra Data Lineage service and all assets in Data Catalog. |

| No | Name | Description |
|---|---|---|
| 3 | Found in | The location where the asset or data object was found. There are three possible locations:<br>• **Data Catalog**: The asset was found in Data Catalog, but it does not match the full path of a data object on the Collibra Data Lineage service. As a result, there is no technical lineage created for this asset.<br>• **Technical lineage**: The data object was found in the data source for which you created a technical lineage, but it does not match the full path of an asset in Data Catalog. As a result, the data object is shown in technical lineage with a gray background.<br>• **Data Catalog & Technical lineage**: An asset and a data object with the same full path were found in Data Catalog and on the Collibra Data Lineage service. As a result, they were stitched and are shown in technical lineage with a yellow background. |
| 4 | Show lineage | The button to go back to the technical lineage graph. |

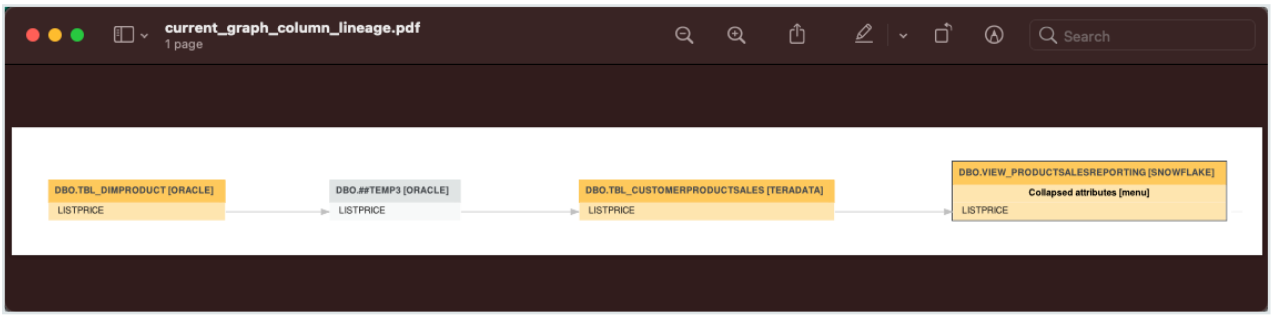
## Technical lineage export types

If you want to share a technical lineage graph of your technical lineage, you can export the information to one of the following export types, via the Settings tab pane:

- PDF
- PNG
- Graph CSV
- Full Batch CSV
- JSON Lineage

## PDF and PNG

The PDF and PNG exports show only the technical lineage graph of the selected table or column.




## Graph CSV

The CSV export option generates a ZIP file with the following CSV file:

| File name | File content |
|---|---|
| current_graph_column_lineage.csv | The technical lineage graph of the selected column or table. |

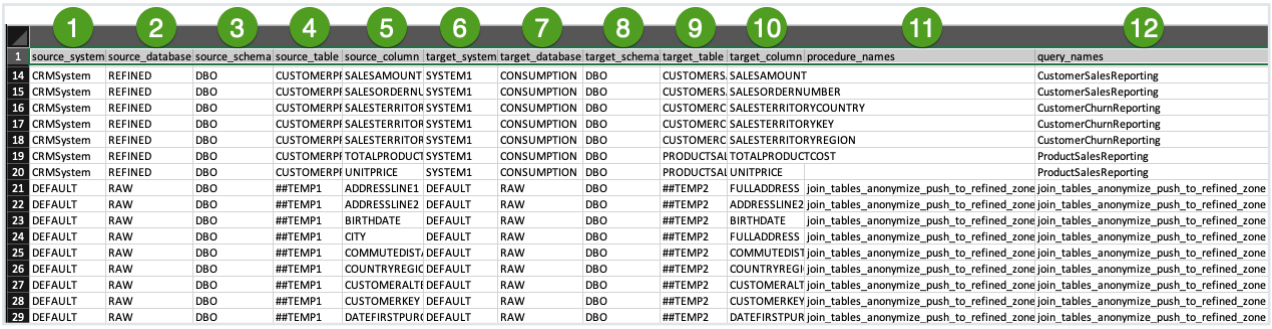
## Full Batch CSV

The Full CSV option generates a ZIP file with the following CSV files:

| File name | File content |
|---|---|
| current_graph_column_lineage.csv | The technical lineage graph of the selected column or table. |
| full_batch_column_lineage.csv | The technical lineage graph of the full technical lineage. |

## Example

The current_graph_column_lineage CSV file and the full_batch_column_lineage CSV files show the same information, but with a different scope. These files show how data flows from source to target.



| 1 | source_system | source_database | source_schema | source_table | source_column | target_system | target_database | target_schema | target_table | target_column | procedure_names | query_names |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 14 | CRMSystem | REFINED | DBO | CUSTOMERPI | SALESAMOUNT | SYSTEM1 | CONSUMPTION | DBO | CUSTOMERS | SALESAMOUNT | | CustomerSalesReporting |
| 15 | CRMSystem | REFINED | DBO | CUSTOMERPI | SALESORDERNU | SYSTEM1 | CONSUMPTION | DBO | CUSTOMERS | SALESORDERNUMBER | | CustomerSalesReporting |
| 16 | CRMSystem | REFINED | DBO | CUSTOMERPI | SALESTERRITOR | SYSTEM1 | CONSUMPTION | DBO | CUSTOMERC | SALESTERRITORYCOUNTRY | | CustomerChurnReporting |
| 17 | CRMSystem | REFINED | DBO | CUSTOMERPI | SALESTERRITOR | SYSTEM1 | CONSUMPTION | DBO | CUSTOMERC | SALESTERRITORYKEY | | CustomerChurnReporting |
| 18 | CRMSystem | REFINED | DBO | CUSTOMERPI | SALESTERRITOR | SYSTEM1 | CONSUMPTION | DBO | CUSTOMERC | SALESTERRITORYREGION | | CustomerChurnReporting |
| 19 | CRMSystem | REFINED | DBO | CUSTOMERPI | TOTALPRODUCT | SYSTEM1 | CONSUMPTION | DBO | PRODUCTSAL | TOTALPRODUCTCOST | | ProductSalesReporting |
| 20 | CRMSystem | REFINED | DBO | CUSTOMERPI | UNITPRICE | SYSTEM1 | CONSUMPTION | DBO | PRODUCTSAL | UNITPRICE | | ProductSalesReporting |
| 21 | DEFAULT | RAW | DBO | ##TEMP1 | ADDRESSLINE1 | DEFAULT | RAW | DBO | ##TEMP2 | FULLADDRESS | join_tables_anonymize_push_to_refined_zone | join_tables_anonymize_push_to_refined_zone |
| 22 | DEFAULT | RAW | DBO | ##TEMP1 | ADDRESSLINE2 | DEFAULT | RAW | DBO | ##TEMP2 | ADDRESSLINE2 | join_tables_anonymize_push_to_refined_zone | join_tables_anonymize_push_to_refined_zone |
| 23 | DEFAULT | RAW | DBO | ##TEMP1 | BIRTHDATE | DEFAULT | RAW | DBO | ##TEMP2 | BIRTHDATE | join_tables_anonymize_push_to_refined_zone | join_tables_anonymize_push_to_refined_zone |
| 24 | DEFAULT | RAW | DBO | ##TEMP1 | CITY | DEFAULT | RAW | DBO | ##TEMP2 | FULLADDRESS | join_tables_anonymize_push_to_refined_zone | join_tables_anonymize_push_to_refined_zone |
| 25 | DEFAULT | RAW | DBO | ##TEMP1 | COMMUTEDIST/ | DEFAULT | RAW | DBO | ##TEMP2 | COMMUTEDIST | join_tables_anonymize_push_to_refined_zone | join_tables_anonymize_push_to_refined_zone |
| 26 | DEFAULT | RAW | DBO | ##TEMP1 | COUNTRYREGIC | DEFAULT | RAW | DBO | ##TEMP2 | COUNTRYREGI | join_tables_anonymize_push_to_refined_zone | join_tables_anonymize_push_to_refined_zone |
| 27 | DEFAULT | RAW | DBO | ##TEMP1 | CUSTOMERALTI | DEFAULT | RAW | DBO | ##TEMP2 | CUSTOMERALT | join_tables_anonymize_push_to_refined_zone | join_tables_anonymize_push_to_refined_zone |
| 28 | DEFAULT | RAW | DBO | ##TEMP1 | CUSTOMERKEY | DEFAULT | RAW | DBO | ##TEMP2 | CUSTOMERKEY | join_tables_anonymize_push_to_refined_zone | join_tables_anonymize_push_to_refined_zone |
| 29 | DEFAULT | RAW | DBO | ##TEMP1 | DATEFIRSTPUR( | DEFAULT | RAW | DBO | ##TEMP2 | DATEFIRSTPUR | join_tables_anonymize_push_to_refined_zone | join_tables_anonymize_push_to_refined_zone |

| No | Column | Description |
|---|---|---|
| 1 | source_system | The name of the source system.<br><br>Note This column is only shown when `useCollibraSystemName` is set to `true` in the lineage harvester configuration file. |
| 2 | source_database | The name of the source database. |
| 3 | source_schema | The name of the source schema. |
| 4 | source_table | The name of the source table. |
| 5 | source_column | The name of the source column. |

| No | Column | Description |
|---|---|---|
| 6 | target_system | The name of the target system.<br><br>Note This column is only shown when `useCollibraSystemName` is set to `true` in the lineage harvester configuration file. |
| 7 | target_database | The name of the target database. |
| 8 | target_schema | The name of the target schema. |
| 9 | target_table | The name of the target table. |
| 10 | target_column | The name of the target column. |
| 11 | procedure_name | The name of the stored procedure. This column remains empty when an object in your technical lineage doesn't have stored procedure.<br><br>Warning This column is deprecated and will be removed in the future. |
| 12 | query_name | The name of the specific source code fragment or transformation detail.<br><br>You can use this name to search for more information in the **Sources** tab page. |

> Tip The names of the source and target objects indicate the full path of the object. For example, the full name of a column is (system) > database > schema > table > column. This path is used to stitch your technical lineage objects to assets in Data Catalog.

## JSON Lineage

This export option generates a JSON file that is formatted in the same manner that is required for creating a custom technical lineage.

## Export the technical lineage information

If you want to share a technical lineage graph or the transformation logic of your technical lineage, for example with colleagues who don't have access to Collibra, you can export the information. For complete details on the various export options, see Technical lineage export types.

### Steps

1. In the Technical lineage viewer, click the Settings tab.
2. Click **Export**.
3. Click the export type.


» The technical lineage information is downloaded.

# Technical lineage troubleshooting

This section describes what you can do when you encounter issues running the lineage harvester, browsing through a technical lineage or stitching data source objects in your data source to existing assets in Data Catalog.

## Technical lineage general troubleshooting

This topic contains the following information:

- Most common issues
- Testing connectivity
- Password errors

### Most common issues

The following messages or other issues can appear when you run the lineage harvester, view a technical lineage or upload the new relations to Data Catalog via Collibra Data Lineage.

> Tip For a list of all error codes and messages that the lineage harvester displays, please see the lineage harvester error codes section.

| Problem | Solution |
| --- | --- |
| You get the following error message:<br>`Could not find or load main class lineage.lineage-harvester-<version nr.>` | This error message appears when the folder path to the lineage harvester is invalid. Check the folder path and make sure that it does not contain whitespaces. |

| Problem | Solution |
| --- | --- |
| You get the following error message:<br>`Failed to load file '<file-name>'. If the file is not in UTF-8, please convert it accordingly.` | This error message appears if the lineage harvester tries to read a non-UTF-8 SQL file of a data source with connection type `SqlDirectory`. To solve this issue, convert all SQL files to UTF-8 and rerun the lineage harvester. |
| The lineage harvester does not connect to hosts using a proxy server. | Technical lineage does not support proxy server authentication, but you can connect to a proxy server. For complete details, including the necessary commands, see Connecting to a proxy server. |

| Problem | Solution |
| --- | --- |
| You get the following error message or a similar certificate error:<br>`Source '<data source name> failed with exception: javax.net.ssl.SSLHandshakeException: General SSLEngine problem` | This message appears when the proxy server sends an unexpected certificate to the lineage harvester or when the default Java TrustStore is empty or outdated.<br>First update Java and rerun the lineage harvester to see if that resolves the issue. If the same error message is shown, try the following:<br>**On Windows**<br>> Note In the following example commands, we refer to the techlin-gcp-us instance. You should refer to the correct Collibra Data Lineage service instance in the geographic location of your Collibra Data Intelligence Cloud environment.<br>1. Run the following command to extract the certificate from the Tableau server:<br>`keytool -printcert -rfc -sslserver techlin-gcp-us.-collibra.com:443 > tableau-cert.crt` |

| Problem | Solution |
| --- | --- |
| | Tip Replace the URL techlin-gcp-us.collibra.com with the URL for your Tableau server, which you specify in the lineage harvester configuration file. This will create a file named tableau-cert.crt in the folder where you run this command.<br><br>2. Run the following command to find the location of your JAVA_HOME:<br>`echo %JAVA_HOME%`<br>» The location path will be something like the following: `C:\Program Files\Java\jdk-17.0.2`<br>3. Use the location path of your JAVA_HOME in the following command, to import the tableau-cert.crt file into the cacerts file found above.<br>`keytool -importcert -file tableau-cert.crt -alias "TableauProdServerCert" -keystore "C:\Program Files\Java\jdk-17.0.2\cacerts"`<br>Note You can specify a different alias, if you want.<br><br>4. Run the following command:<br>`keytool -list -keystore "C:\Program Files\Java\jdk-17.0.2\lib\security\cacerts"` |

| Problem | Solution |
| --- | --- |
| | `\| findstr "Tableau"`<br>5. Enter the keystore password.<br>> Tip The password is typically `changeit`.<br>» A list of all certificates that match the Tableau string in the "C:\Program Files\Java\jdk-17.0.2\cacerts" file is shown.<br>> Tip In the list of certificates, look for the one that you imported in step 3. If it's listed, it means the "C:\Program Files\Java\jdk-17.0.2\cacerts" file has the certificate needed to validate the Tableau server.<br>6. Run the following command to have the lineage harvester use the cacerts file that you just updated.<br>`set JAVA_OPTS=-Djavax.net.ssl.trustStore="C:\Program Files\Java\jdk-17.0.2\lib\security\cacerts" -Djavax.net.ssl.trustStorePassword="changeit"`<br>7. Run the following command to test the synchronization:<br>`./lineage-harvester.bat full-sync -s tableau`<br>**On Linux** |

| Problem | Solution |
| --- | --- |
| | Note<br>• In the following example commands, we refer to the techlin-gcp-us instance. You should refer to the correct Collibra Data Lineage service instance in the geographic location of your Collibra Data Intelligence Cloud environment.<br>• If you want to add an existing certificate to the Java TrustStore, instead of creating a new Keystore, replace "<your keystore name>" in steps 2 and 3, with the path to the `cacerts` file in your Java installation, for example *%JAVA_HOME%\jre\lib\cacerts*.<br><br>1. Use the following command to get a certificate from the corresponding techlin-gcp-us.com site, which is part of the CollibraData Lineage infrastructure:<br>`openssl x509 -in <(openssl s_client -connect techlin-gcp-us.collibra.com:443 -prexit 2>/dev/null) -out techlin-gcp-us.crt`<br><br>Tip If you already have a correctly formatted certificate on the server, you can skip this step. |

| Problem | Solution |
| --- | --- |
| | 2. Add the certificate to the Java TrustStore:<br>`keytool -importcert -file techlin-gcp-us.crt -alias techlin-gcp-us -keystore <your keystore name> -store-pass changeit`<br>3. Run the lineage harvester and use the new TrustStore using the following parameter:<br>`-Djavax.net.ssl.trustStore-e=<your keystore name>`<br>Example To synchronize your data sources again, run the following command:<br>`./bin/lineage-harvester full-sync -Djavax.net.ssl.trustStore=mykeystore` |

| Problem | Solution |
| --- | --- |
| You get the following error messages:<br><br>In the lineage harvester log file:<br><br>`java.lang.Exception: No native library found for os.name=Linux, os.arch=x86_64, paths= [/org/sqlite/native/Linux/x86_64:/usr/java/packages/lib/amd64:/usr/lib64:/lib64:/lib:/usr/lib]`<br><br>In the console:<br><br>`Failed to load native library:<sqlite-file-name>. osinfo: Linux/x86_64 java.lang.UnsatisfiedLinkError: /tmp/<sqlite-file-name>: failed to map segment from shared object: Operation not permitted` | The lineage harvester uses a temporary file containing an SQLite database as a cache file. That means that you need write permission to the `/tmp` folder.<br><br>If this action failed, you can specify another directory with write permissions using `-Dorg.sqlite.tmpdir=<path to a temp directory>`.<br><br>Example You have a temporary directory with write permissions. The path to this directory is `custom/temp`. Run the lineage harvester with the following command:<br><br>`./bin/lineage-harvester -Dorg.sqlite.tmpdir=custom/temp full-sync` |
| You get the following error message:<br><br>`Technical lineage is not enabled for this Catalog instance.` | First make sure that there are no spelling errors in the `Data Catalog` section of the configuration file. If your configuration file is configured correctly, but the issue is not solved, create a support ticket to enable Technical lineage for your Collibra Data Intelligence Cloud instance in Salesforce. |

| Problem | Solution |
| --- | --- |
| You get the following error message:<br>`The size of the import file is too large (max size: 10 MB).` | The file you are trying to upload exceeds the size limit for uploaded files.<br>Contact Collibra support to increase the maximum file size. |
| You get the following error message:<br>`Source 'X' was never successfully processed..` | This message appears when a source that is specified in the lineage harvester configuration file has never been successfully processed by the Collibra Data Lineage service.<br>You can either:<br>• Remove source 'X' from the configuration file, and then run the command again.<br>• Run a full-sync of source X, and then re-run the command that previously failed. |
| Technical lineage is unavailable because the selected table does not contain columns. | Technical lineage only includes tables that have columns. Add a relation of the type "Table contains/is part of Column" between your Table asset and Column assets. |

| Problem | Solution |
| --- | --- |
| You get the following message in your technical lineage:<br>`The current asset doesn't have a technical lineage yet.` | This message appears if one or more of the following situations apply:<br>• The data source of the current asset is not included in the configuration file. If you want a technical lineage for this asset, add its data source to the configuration file.<br>• You have upgraded to the lineage harvester 1.3.0 or newer or you created a technical lineage for the first time. In this case, you may need to restart your DGC service before you can see the technical lineage.<br>• You see parsing errors. For more information, see the Sources tab page.<br>• The full name of one or more relevant assets does not match any of the names of the assets in the configuration file, which causes automatic stitching to fail. Make sure that the information in the configuration file and the Data Catalog physical data layer matches:<br>◦ The relevant assets have relations between each other, for example *Technology asset groups/is grouped by Technology asset> → <Database asset> contains/is part of <Schema asset> contains/is part of <Table asset> contains/is part of <Column asset>*. |

| Problem | Solution |
| --- | --- |
| | ◦ The full name of your System asset matches the name of your system or the name you used in the configuration file.<br>◦ The full name of your Database asset matches the name of your database or, for Google BigQuery your project, or the name you used in the configuration file.<br>◦ The full name of your Schema asset matches the name of the Schema of the data source or the name you used in the configuration file.<br>Tip Make sure that the full path of each asset in Data Catalog matches the full path of the corresponding data object from your data source on the Stitching tab page. |

| Problem | Solution |
| --- | --- |
| You get one of the following messages:<br>• `Nodes count exceeds the limit 350.`<br>• `Edges count exceeds the limit 1000.` | This message appears when the technical lineage graph exceeds the limit of 350 nodes or 1,000 edges, and is too large to build. This happens, for example, if you have a table with many columns and you try to show the technical lineage of all columns in a table in one graph.<br><br>If you see this message, we recommend that you browse through the technical lineage graph on the object level or select a single column in the Browse tab pane.<br><br>> Note You cannot manually change these limits. |

| Problem | Solution |
| --- | --- |
| You get the following error message in your technical lineage for a Microsoft SQL Server data source: "`Oops, no data flow found in your SQL scripts. Make sure you upload DML queries like insert, update, merge that moves data between the tables.`" | This error message appears when you run the lineage harvester to create a technical lineage for a Microsoft SQL Server data source without having the correct permissions to the SQL Server. As a result, the lineage harvester processes empty files and there is no technical lineage available for this data source.<br><br>Make sure you have at least the VIEW DEFINITION permission or sysadmin role in Microsoft SQL Server.<br><br>> Note If you use multiple users, make sure that each one of them has the proper permissions. |

| Problem | Solution |
| --- | --- |
| You get the following error message:<br><br>`net.snowflake.client.jdbc.SnowflakeSQLLoggedException: JDBC driver internal error: Fail to retrieve row count for first arrow chunk: sun.misc.Unsafe or java.nio.DirectByteBuffer.<init>(long, int) not available.` | The issue is relate to the Arrow library, a dependency of the Snowflake JDBC driver. The issue cannot be resolved. Updating the Snowflake JDBC driver, for example, doesn't help. However, the following workaround does work.<br><br>**On Windows**<br><br>1. Run the following command:<br>`set JAVA_OPTS="--add-opens=java.base/java.nio=ALL-UNNAMED"`<br>2. In the same command line, run:<br>`.\bin\lineage-harvester.bat full-sync`<br><br>**On Linux**<br>Run the following command:<br>`JAVA_OPTS="--add-opens=java.base/java.nio=ALL-UNNAMED" ./bin/lineage-harvester full-sync` |

| Problem | Solution |
| --- | --- |
| You get the following error message:<br>`Source 'SnowflakeInfo' failed with exception: net.snowflake.client.jdbc.SnowflakeSQLException: SQL compilation error:`<br>`Database 'SNOWFLAKE' does not exist or not authorized.` | To access the Snowflake shared read-only database, you need a user that has the Default role.<br>Tip You can use the `customConnectionProperties` property in the lineage harvester configuration file to assign the Default role to the user, if the role is not assigned in Snowflake. For example: `"customConnectionProperties": "role=default"` |
| The import job fails.<br>Note If the import job fails during import and the failing job is rolled back, you can have both old and new relations. The old relations were created during the first job and the new relations are created after the rollback. If more than one job is triggered, only the failed job is rolled back. | First, check the following:<br>• The asset ID must exist.<br>• The structure of the data must be correct.<br>• The cardinality of relation types between asset types.<br>Then, rerun the import of relations. |
| Relations are not changed as expected. | Check whether the lineage harvester refreshed the data source via a scheduled job. If the import job failed, then the data source was not refreshed and the previously created relations stay the same. If that happened, rerun the lineage harvester to import again. |

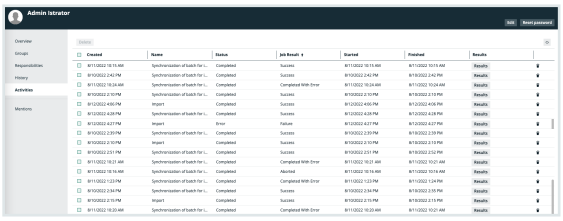
| Problem | Solution |
| --- | --- |
| Manual relations are overwritten. | We recommend that you do not manually add relations of the type "Data Element targets / sources Data Element" between asset types that are imported via the scheduled jobs. These relations are overwritten every time the scheduled job synchronizes the data source. |
| Ingesting Looker or Power BI assets fails. | For more information, see the following sections:<br>• Looker troubleshooting.<br>• Power BI troubleshooting |

| Problem | Solution |
| --- | --- |
| You get the following error message:<br>`java.lang.OutOfMemoryError: Java heap space` | This error message indicates that Java does not have enough memory allocated to finish the task. This error can happen anytime during Harvester run. Follow these steps to increase the maximum heap size.<br><br>> Note 4 GB RAM is sufficient in most cases, but more memory could be needed for larger harvesting tasks.<br><br>**On Windows**<br>Run the following command:<br>`set JAVA_OPTS=-Xmx4g && .\bin\lineage-harvester.bat full-sync`<br><br>In this example, `4g` means 4 GB.<br><br>> Tip If you want to check the default maximum heap size, run the following command:<br>`java -XX:+PrintFlagsFinal -version \| findstr MaxHeapSize`<br><br>**On Linux**<br>Run the following command:<br>`JAVA_OPTS=-Xmx4g ./bin/lineage-harvester full-sync`<br>In this example, `4g` means 4 GB. |

| Problem | Solution |
| --- | --- |
| | Tip If you want to check the default maximum heap size, run the following command: `java -XX:+PrintFlagsFinal -version \| grep MaxHeapSize` |
| You get the following error message:<br>`java.lang.NoSuch.MethodError` | This error message indicates that the JAVA_HOME was not specified; therefore, the harvester was using a previous version of Java. With the following commands, you can specify the Java version to 11, which is needed to successfully run the lineage harvester:<br>• `export JAVA_HOME=/us`<br>• `echo $JAVA_HOME` |

| Problem | Solution |
| --- | --- |
| You get the following error message:<br><br>`Error: A JNI error has occurred, please check your installation and try again`<br><br>`Exception in thread "main" java.lang.UnsupportedClassVersionError: harvester/Harvester has been compiled by a more recent version of the Java Runtime (class file version 55.0), this version of the Java Runtime only recognizes class file versions up to 52.0 ...` | In this error message, `class file versions up to 52.0` indicates that Java 8 was used; however, lineage harvester requires Java version 11 or newer.<br><br>If there are multiple versions of Java installed, the lineage harvester might pick Java 8 instead of Java 11. You can run the command `java -version` to check your Java version.<br><br>To resolve this issue, set the path to the correct Java installation directory, in the `JAVA_HOME` environment variable. To do so, run the lineage harvester with the following command:<br><br>On Windows:<br>`set JAVA_HOME=\path\to\java_11_dir && .\bin\lineage-harvester.bat full-sync`<br><br>On Linux:<br>`JAVA_HOME=/path/to/java_11_dir ./bin/lineage-harvester full-sync` |

| Problem | Solution |
| --- | --- |
| You get a `NegativeArraySizeException` error. | A `NegativeArraySizeException` error is shown if your Java Virtual Machine (JVM) has the string compaction feature disabled. In that case, calling the `getBytes` results in an attempt to allocate triple the size of the string's value, which can exceed the size limit.<br><br>To resolve this error, try running the lineage harvester with the string compaction feature enabled, by running the following command:<br><br>`JAVA_OPTS='-XX:+CompactStrings -Xmx8g' ./lineage-harvester full-sync -s <source ID>` |

| Problem | Solution |
| --- | --- |
| Synchronization of a data source fails completely or with some errors. | A synchronization job that is completed without any errors has the Success status. Other possible statuses, Completed With Error, Aborted and Failure, are determined in part by the value of the "Number of failed commands before stopping import job" setting, in Collibra Console.<br><br>For complete information, see Synchronization: Continue on error option.<br><br>You can view the results of a synchronization job in the Activities list.<br><br>  <br><br>In the Activities list, click **Results** in the relevant row to view the details of a synchronization job. The details are intended to help you resolve the errors. To help reduce the chance of an aborted synchronization job, consider increasing the value of the "Number of failed commands before stopping import job" setting. |

## Testing connectivity

You can check whether the lineage harvester can connect to the Collibra Data Lineage service instance and Data Catalog.

1. Run the lineage harvester in command line.
2. Run the following command: `test-connection`.
   » The result shows if the lineage harvester can connect to the Collibra Data Lineage service instance and Data Catalog.

The logs will also show the IP addresses of the Collibra Data Lineage service instances that you have to whitelist.

## Password errors

If you mistyped the password or want to change an existing password, go to the lineage harvester folder > **config/pwd.conf** and delete the lines below. As a result, the lineage harvester will ask for the password again.

> Tip If you have the lineage harvester version 1.3.0 or newer, you can also provide your passwords via stdin or a password manager.

```
{
   "url" : "<URL>",
   "userName" : "<user>",
   "password" : "<password>"
}
```

# Technical lineage known issues and limitations

The following table shows known issues and limitations in the current lineage harvester version.

> Important The success rate of a technical lineage, as shown in the Sources tab page, gives a good indication of the processing success. A success rate less than 100%, however, does not mean processing was unsuccessful. A parsing error, for example, which negatively affects the success rate, does not always negatively affect the completeness of the lineage.

| Known issue | Description |
| --- | --- |
| The lineage harvester currently does not support Java version 16. | If you have Java version 16 and run the lineage harvester with the `full-sync` command, the harvester fails during the API key retrieval process.<br><br>As a workaround, we recommend the following:<br><br>1. Set the JAVA_OPTS to the following:<br><br>`JAVA_OPTS='--illegal-access=deny'`<br><br>2. Run the lineage harvester in the same command line window. |
| Collibra Data Lineage does not reuse the database model or DDL statements from other sources in the lineage harvester configuration file. | Currently, all sources in the lineage harvester configuration file are analyzed separately. As a result, the database model and DDL statements that are used for one source are not taken into account when analyzing another source.<br><br>As a workaround, we recommend that you make sure that each source has all DDL statements that it needs to be processed properly.<br><br>> Tip Saving the DDL statements in separate files and adding the preview "_" before their names might speed up the analysis of the DDL statements. |

| Known issue | Description |
| --- | --- |
| Harvesting an Amazon Redshift data source fails when using a CDATA JDBC driver. | If you use a CDATA JDBC driver to harvest metadata from an Amazon Redshift data source, you have to set the `queryPassthrough` property in the connection configuration to `true`, otherwise the driver fails to execute the query.<br><br>> Note The `queryPassthrough` property is only required if you are using the CDATA Redshift driver, for ingestion via Edge. The lineage harvester via CLI uses the native Redshift driver, which does not require the `QueryPassthrough` property. |

## Lineage harvester messages

A message code is shown in the lineage harvester logs when something goes wrong during the lineage harvester process. The message code indicates which part of the harvesting process was skipped or failed and provides steps to resolve it.

# General lineage harvester messages

| Message code | Description |
|---|---|
| MSG-LIN-1001 | The current asset does not have a technical lineage yet.<br><br>Only assets that are processed and stitched by Collibra Data Lineage have a Technical lineage.<br><br>Look for the asset name in the navigation tree of the Browse tab pane, to see if the asset was processed.<br><br>• If the asset name is not shown in the navigation tree, ensure that the data source of the asset is included in the configuration file.<br>• If the asset name is shown in the navigation tree, ensure that you correctly prepared the Data Catalog physical data layer for technical lineage before you run the harvester. Specifically, the full path of each asset in Data Catalog must match the full path of the corresponding data object from your data source on the Stitching tab page.<br><br>Less likely factors, such as your lineage harvester version and parsing errors can also lead to this error.<br><br>For complete troubleshooting information, see Technical lineage general troubleshooting. |
| MSG-LIN-3000 | This is an unknown or unclassified lineage harvester error. Create a support ticket to report the issue. |

| Message code | Description |
| --- | --- |
| MSG-LIN-3001 | The lineage harvester was able to successfully connect to the Collibra Data Lineage service instances, but received HTTP client error response.<br><br>If the error message contains **Technical lineage is not enabled for this Catalog instance**, do the following:<br><br>• Make sure that the URL to your Collibra Data Intelligence Cloud in the `catalog` section of the lineage harvester configuration file is correct.<br>• Make sure that the username and password you use to sign in to Collibra are correct.<br>• Make sure that Collibra Data Lineage is enabled for your Collibra environment.<br><br>If the error message contains **Enter a valid URL**, do the following:<br><br>• This error is caused by an invalid URL. Make sure that the URL to your Collibra Data Intelligence Cloud in the `catalog` section of the lineage harvester configuration file is correct.<br><br>If the issue persists, please contact Collibra support or your customer success manager. |
| MSG-LIN-3002 | The lineage harvester was able to successfully connect to an instance of the Collibra Data Lineage service, but received an HTTP server error response.<br><br>Wait a few minutes and then run the lineage harvester again. If the issue persists, please contact Collibra support or your customer success manager. |

| Message code | Description |
| --- | --- |
| MSG-LIN-3003 | The lineage harvester failed to retrieve the API key of your Collibra Data Intelligence Cloud environment with Data Catalog from the Collibra Data Lineage service instances due to network connectivity issues.<br><br>To resolve this issue, do the following:<br><br>• Check your network connectivity.<br>• Make sure you have whitelisted the IP addresses of all Collibra Data Lineage service instances.<br>• Check your proxy settings.<br><br>Tip You can test your connectivity using the `test-connectivity` command. |
| MSG-LIN-3004 | Unable to determine the geographic location of your Collibra Data Intelligence Cloud environment.<br><br>When you run the lineage harvester, it firsts connects to any available Collibra Data Lineage service instance to determine your cloud provider and geographic location of your Collibra environment. Then, the lineage harvester sends the harvested metadata to the Collibra Data Lineage service instance with the same cloud provider and geographic location.<br><br>In this case, the geographic location of your Collibra environment could not be determined. If the issue persists, please contact Collibra support or your customer success manager. |

| Message code | Description |
| --- | --- |
| MSG-LIN-3005 | Connection error due to Snowflake DB Client.<br><br>The lineage harvester encountered an error through the Snowflake database connector SDK. This issue is specific to the Snowflake connector, not the lineage harvester.<br><br>To resolve this issue, try the following:<br><br>• In JDK16, run the lineage harvester with the following command: `-Djdk.module.illegalAccess=permit`<br>• In JDK17, run the lineage harvester with the following command: `--add-opens jdk.unsupported/sun.misc=ALL-UNNAMED`<br><br>`JAVA_OPTS="--add-opens=java.base/java.nio=ALL-UNNAMED" ./bin/lineage-harvester load-sources` |
| MSG-LIN-4000 | The Collibra Data Lineage service instance is unable to connect to Data Catalog.<br><br>To resolve this issue, try the following:<br><br>• Check your network connectivity.<br>• Make sure that the URL to your Collibra Data Intelligence Cloud in the `catalog` section of the lineage harvester configuration file is correct.<br>• Make sure the host names of all databases in the lineage harvester configuration file are correct.<br><br>If the issue persists, please contact Collibra support or your customer success manager. |

| Message code | Description |
|---|---|
| MSG-LIN-19001 | Connection not defined in config file.<br><br>This means that a connection to the system or server could not be established.<br><br>To resolve this issue, try to ensure that you have correctly prepared your Informatica Intelligent Cloud Services <source ID> configuration file. |
| MSG-LIN-19002 | Taskflow failed to process because of missing connection definition.<br><br>A taskflow could not be processed because one of the mappings in the taskflow refers to a connection that could not be extracted from the <source ID> configuration file.<br><br>To resolve this issue, try to ensure that you have correctly prepared your Informatica Intelligent Cloud Services <source ID> configuration file. |
| MSG-LIN--19003 | Both the **auth** and **username** properties exist in the lineage harvester configuration file. Lineage harvester processing ends.<br><br>The **username** property is deprecated. Take the following steps:<br><br>1. Update the lineage harvester configuration file by using the **auth** property only.<br>2. Run the lineage harvester again. |

## SQL scanner messages

| Message code | Steps to resolve the issue |
|---|---|
| MSG-LIN-5001 | This is an unexpected error. Create a support ticket to report your issue. |

| Message code | Steps to resolve the issue |
| --- | --- |
| MSG-LIN-5002 | <Object> not found, please provide DDL or object definition.<br><br>The scanner for SQL statements couldn't successfully complete its analysis, due to a missing object definition. The error message includes the name of the object.<br><br>This happens, for example, in a scenario whereby a SQL statement such as "CREATE TABLE TMP AS SELECT * FROM ACCOUNTS" is uploaded, but the definition for the table ACCOUNTS was not uploaded.<br><br>In this case, it's impossible to extract lineage information, as the structure of the table ACCOUNTS is unknown. Therefore "*" cannot be expanded to an actual list of columns, which results in the error.<br><br>To resolve this issue, if you are uploading SQL statements as files, you need to ensure that you provide DDL for all objects. You can inspect the lineage harvester output to identify the queries that are using the object in the error. |

## Synchronization: Continue on error option

This option allows you to continue the processing of an import or synchronization job even if one or more commands fail. With this option disabled, calls to the Import and Sync APIs either fully succeed or fully fail. You might wait for a lengthy import or synchronization job to complete, only to have it fail completely because of a single error.

With this option enabled, commands that have validation errors and those that failed to execute are skipped, allowing the processing of valid commands to continue until the job is complete or until an error threshold is met. The error threshold is determined by the "Number of failed commands before stopping import job" setting in Collibra Console. The default value is 100.

This feature applies to the following APIs:

- Full Sync (/Import/Synchronize)
- Batch Sync (/Import/Synchronize/Batch)
- Import (/Import)

For more information, see the Import API Documentation in the Collibra Developer Portal.

## Enabling and disabling

This option is enabled by default. You can disable it via the **continueOnError** parameter in your API call.

## Benefits of this option

- Errors are skipped and valid commands are processed, instead of immediate and complete failure of the job.
- All errors are identified at once, reducing the chances of running a job multiple times, only to discover additional errors.
- Complete error information, including the resource identifier, to quickly identify the source and reason for errors.

> Important Importing or synchronizing data with this feature enabled can result in partial import or synchronization results, leading to data inconsistencies between Collibra and the external system.

## Job results

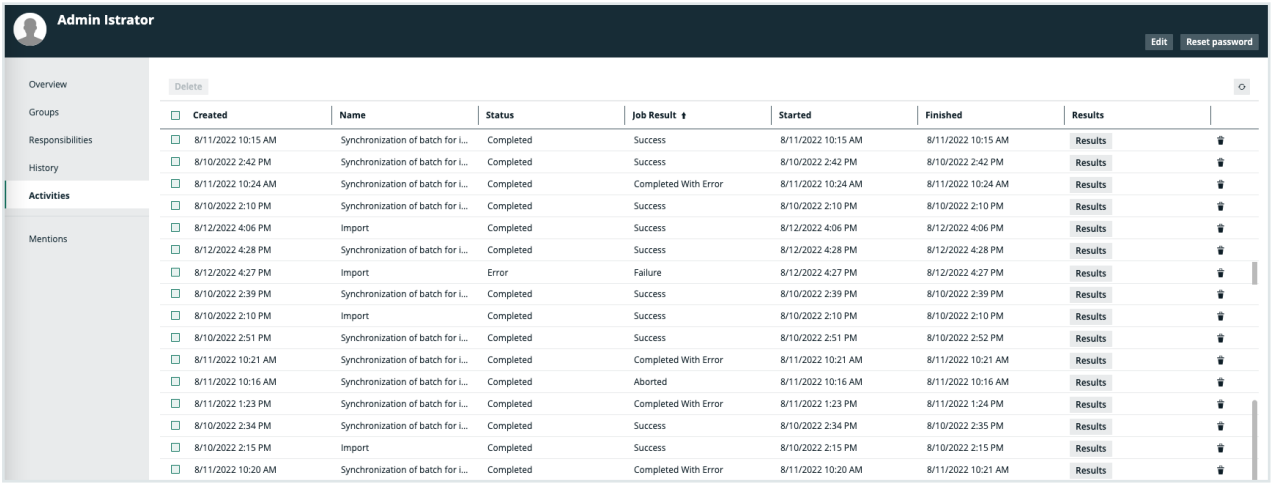
The following table shows the four possible job results for an import or synchronization job:

| Job result | Description |
|---|---|
| Success | The job was completed without errors. |
| Completed With Error | Errors were detected, but the error threshold was not reached and the job was completed. |

| Job result | Description |
|---|---|
| Aborted | The error threshold was exceeded, at which point, the job was stopped. All commands that were executed before the stoppage stay committed. |
| Failure | The job was stopped and any executed commands were rolled back. |

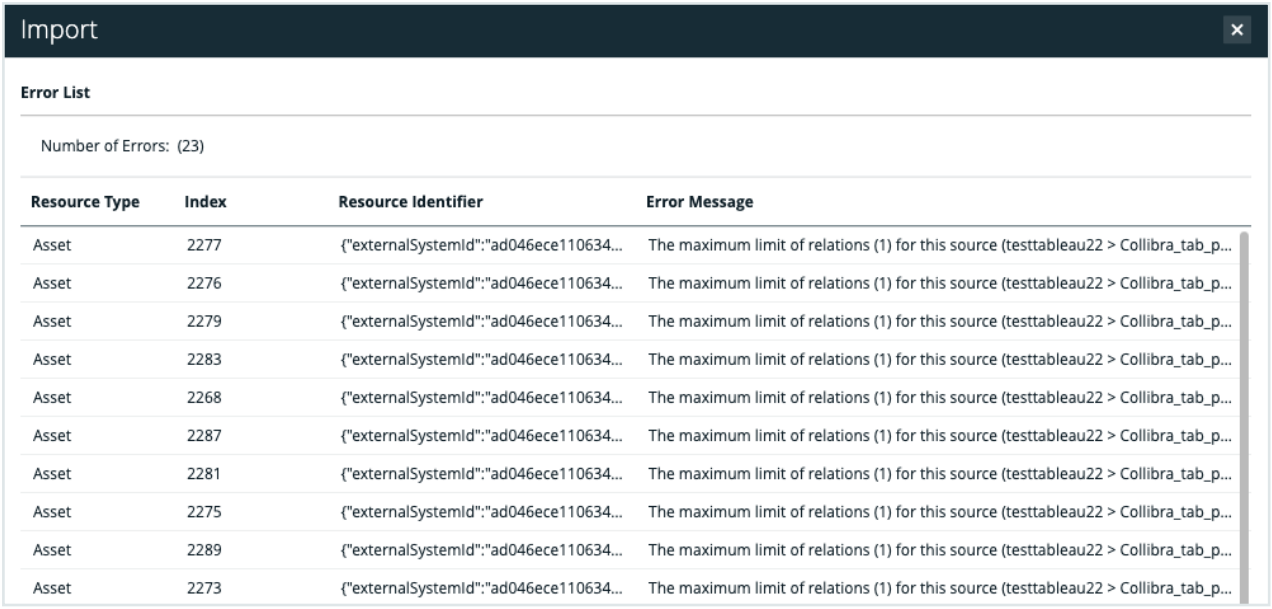
## List of errors

You can view the results of a synchronization job in the Activities list.




When you click **Results** in the relevant row, a dialog box opens, showing a general summary of the job. For jobs with the job result Completed With Error, Aborted, or Failure, the dialog box includes a link to a list of errors. The list of errors includes the following information:

- The resource type.
- The index number.
- The resource identifier.
- An error message.




# Upgrade the lineage harvester

Each new lineage harvester adds features and enhancements to the previous version. We highly recommend that you always use the newest lineage harvester available.

If you have created a technical lineage using an older lineage harvester, you can easily upgrade to the newest lineage harvester and reuse your configuration file.

> Tip For a list of differences between lineage harvester versions, see the lineage harvester change log.

## Steps

1. Download the newest lineage harvester from the Collibra Downloads page.
2. Install the lineage harvester.
3. Copy the code from your old configuration file and paste it in the new configuration file, in the lineage harvester folder included in your new download.
4. Optionally, edit the lineage harvester configuration file to suit your needs.
5. Use the `full-sync` command to synchronize all data sources in your configuration file.
   » The lineage harvester synchronizes your data sources on the Collibra Data Lineage service and refreshes your technical lineage.

# The lineage harvester change log

Collibra Data Lineage is updated and improved on a regular basis. On this page, you can see the most important changes between different versions of the lineage harvester. For a complete list, see the release notes.

> Note In the documentation, we assume that you have the most recent version of the lineage harvester. We highly recommend to download and use the newest lineage harvester from the Collibra downloads page even if you are on an older version of Collibra Data Intelligence Cloud.

> Warning If you upgrade to lineage harvester 1.3.0 or newer, you have to follow an upgrade procedure.

The following list contains the most important changes to the lineage harvester and its configuration file.

| Changed in version | New lineage harvester improvements |
| --- | --- |
| 2022.10 | • The lineage harvester now supports the following IBM DB2 constructs: PREVVAL FOR <sequence>, PREVIOUS VALUE FRO <sequence>, NEXTVAL FOR <sequence> and NEXT VALUE FOR <sequence>.<br>• You can now use the new optional "deleteRawMetadataAfterProcessing" property in your lineage harvester configuration file. With this property, you can delete your raw metadata from the Collibra Data Lineage service after processing. This property is applicable for all supported data sources.<br>• When you specify a Data Catalog URL in the lineage harvester configuration file, it no longer matters whether you include a trailing slash (/) in the URL.<br>• The Collibra Data Lineage service now supports the following transformations: Table.FromRecords and Table.IsEmpty.<br>• Collibra Data Lineage now supports key-pair authentication when ingesting Snowflake data sources.<br>• The PostgreSQL JDBC Driver is upgraded to version 42.4.1.<br>• The Collibra Data Lineage service can now compute indirect lineage from set queries, which are queries with the UNION keyword with the ORDER BY clause.<br>• When you integrate Power BI, the lineage harvester is now more resilient to OutOfMemory errors.<br>• When you integrate Tableau and filter on a sub-project, the metadata of the parent project is no longer ingested in Collibra. However, the parent Tableau Project asset is created in the default domain, to preserve the hierarchy required for stitching.<br>• Looker integration no longer fails if the "collibraSystemName" property is not included in the lineage harvester configuration file. If you want to specify the system name of a database in Looker, use the "collibraSystemName" property in the Looker source ID configuration file. If you don't specify a system name in the source ID configuration file, the system name in the technical lineage |

| Changed in version | New lineage harvester improvements |
| --- | --- |
| | graph will be Default.<br>• In the case of a lookup procedure when ingesting Informatica Intelligent Cloud Services data sources, if the CONNECTIONSUBTYPE parameter is empty, the Collibra Data Lineage service now looks to the CONNECTIONREFERENCE parameter for the name. If that is also empty, then the name in the VARIABLE parameter is used. The ensures the correct detection of the SQL dialect.<br>• Fixed an issue related to dialect extraction when ingesting Informatica Intelligent Cloud Services data sources. |

| Changed in version | New lineage harvester improvements |
| --- | --- |
| 2022.09 | • Previously, when you created a technical lineage for Power BI, SQL Server Reporting Services (SSRS) or Power BI Report Server (PBRS), the nodes in the technical lineage graph had a gray background, even if the data objects from your data source were stitched to assets in Data Catalog. Data objects now have the intended yellow background when creating a technical lineage for Power BI, SSRS or PBRS. We introduced this enhancement for Tableau and Looker in Collibra 2022.07.<br>• When you integrate Tableau, for every Tableau Workbook that you have permission to ingest, all Tableau Dashboards in the Workbooks are now correctly shown in the technical lineage graph. If you do not have permission on the Workbook or Dashboard level, the metadata of these data objects is not ingested.<br>• When integrating Power BI, the ownership information (email address only) for reports is now ingested in Collibra. The new Owner in source attribute is included on Power BI Report asset pages.<br>• The lineage harvester now uses Looker 4.0 APIs, with paging options.<br>• When you integrate Power BI, the lineage harvester is now more resilient against OutOfMemory errors.<br>• When you integrate Tableau and use domain mapping, subprojects are now ingested in the domains of their parent projects.<br>• The Collibra Data Lineage service instances now benefit from the following parsing enhancements when integrating Snowflake data sources:<br>◦ Support for the COLLATE keyword.<br>◦ Support for EXTERNAL TABLE syntax.<br>• When integrating Power BI, the descriptions of Data Set Tables and Data Set Columns in Power BI are now harvested.<br>• Fixed an issue that was resulting in a processing error when a column referenced in an ORDER BY clause references a repeated column in the SELECT column list. |

| Changed in version | New lineage harvester improvements |
| --- | --- |
| | • When integrating Tableau, you can now ingest sub-projects for which you have permission to ingest, even if you don't have permission to ingest the parent projects. |

| Changed in version | New lineage harvester improvements |
| --- | --- |
| 2022.08 | • Previously, when you created a technical lineage for a supported BI tool, the nodes in the technical lineage graph had a gray background, even if the data objects from your data source were stitched to assets in Data Catalog. Data objects now have the intended yellow background when creating a technical lineage for Power BI. This enhancement was introduced for Tableau or Looker in Collibra 2022.07. Soon, the enhancement will also apply to SSRS and PBRS.<br>• When synchronizing Tableau, the synchronization no longer fails if two data sources in the same project with the same name are returned from the Tableau API. The assets of both data sources are now synchronized in Collibra.<br>• You can now filter on the Tableau project level.<br>• When integrating Power BI, you can now ingest measures and show them in the technical lineage. Measures are included as the value in the Role in Report attribute on Power BI Column asset pages.<br>• When attempting to integrate Power BI with invalid Power BI credentials, the lineage harvester log file now provides a more helpful error message.<br>• When you specify the Power BI workspaces for ingestion, the filters are not case sensitive now.<br>• When integrating Looker, the ownership information (email address only) for folders, Looks and dashboards is now ingested in Collibra. The new Owner in source attribute is included on Looker Folder, Looker Look and Looker Dashboard asset pages.<br>• When integrating Power BI, the ownership information (email address only) for data sets and workspaces is now ingested in Collibra. The new Owner in source attribute is included on Power BI Data Model and Power BI Workspace asset pages.<br>• The lineage harvester log file now identifies whether you are using Tableau Online or Tableau Server, and the version of your Tableau environment. |

| Changed in version | New lineage harvester improvements |
| --- | --- |
| 2022.07 | • The lineage harvester now retries to get a batch status again if the first HTTP call failed due to a network error.<br>• Fixed an issue that was causing custom SQL queries to be identified as belonging to two different Tableau data sources. This resulted in a "Unique constraint failed" error.<br>• Fixed an issue that was resulting in the No asset matches the specified criteria error.<br>• When the lineage harvester fetches an access key for a data store, only active records are now fetched. Inactive records are ignored.<br>• The lineage harvester is more resilient against authorization expiration when ingesting Looker metadata.<br>• The lineage harvester log file now includes the following information:<br>◦ Your Tableau environment type: Tableau Online or Tableau Server type<br>◦ The version of your Tableau environment |

| Changed in version | New lineage harvester improvements |
| --- | --- |
| 2022.06 | • When synchronizing Power BI, the last sync time is now correctly shown in the Sources tab page.<br>• Fixed an issue that was causing the processing of harvested metadata batches to run without coming to completion.<br>• When ingesting Power BI, if there are Oracle data sources, the Oracle service name is now used, instead of the database name.<br>• When processing Tableau metadata, the Collibra Data Lineage servers no longer replace ">>" by "<}", which was resulting in parsing errors.<br>• Fixed an [SQLITE_ERROR] issue that was breaking the technical lineage when attempting to synchronize a data source.<br>• When processing Power BI metadata, SQL statements are now in upper case.<br>• When creating a technical lineage for Tableau, any unnecessary brackets “][“ in the names of schemas are now removed.<br>• When integrating Power BI, you can now ingest measures without DAX. They are shown as attribute type Role in Report on Power BI Column asset pages. |

| Changed in version | New lineage harvester improvements |
| --- | --- |
| 2022.05 | Warning The lineage harvester 2022.05 includes an internal format change to the password manager pwd.conf file. This means that if you use Lineage harvester 2022.05, you can no longer use the pwd.conf file with an older harvester.<br>• You can now integrate Power BI in Data Catalog via the lineage harvester, meaning you no longer need to use the Power BI harvester. Additional benefits include the following:<br>◦ Support for Power BI Data Flows.<br>◦ Descriptions of Power BI Reports.<br>◦ Statuses of Power BI Workspaces.<br>◦ Filtering and domain mapping.<br>Note The new Power BI integration method is specifically for new integrations. For those who have been ingesting Power BI via the Power BI harvester, we will soon release a migration script.<br>• Collibra Data Lineage now also supports the following BI integrations:<br>◦ MicroStrategy<br>◦ SQL Server Reporting Services and Power BI Report Server.<br>• You can now use token-based authentication when creating a technical lineage for Matillion.<br>Warning This enhancement is not backwards compatible. You must update your configuration file.If you use the lineage harvester 2022.05, you can no longer use the pwd.conf file with an older harvester.<br>• The `useCollibraSystemName` property is now solely used for the configuration of the system name.<br>• If you set the `useCollibraSystemName` property to `true` in your lineage harvester configuration file, but don't define the system |

| Changed in version | New lineage harvester improvements |
| --- | --- |
| | name in the Tableau <source ID> configuration file, the system name in the Tableau technical lineage shows DEFAULT as the system name.<br>• If using a Tableau <source ID> configuration file:<br>◦ You can now use wildcards throughout the file.<br>◦ The hostName and connectorUrl properties are no longer case-sensitive.<br>• The PostgreSQL JDBC driver is now upgraded from from 42.3.2 to 42.3.3.<br>• The Apache Hive JDBC driver is now upgraded from 2.6.17.1020 to 2.6.19.2022.<br>• The lineage harvester no longer hangs when harvesting metadata from certain data sources.<br>• The lineage harvester automatically refreshes Tableau tokens.<br>• You can now use the optional `concurrencyLevel` property in the lineage harvester configuration file, to specify the internal sizing, meaning the amount of tasks that can be executed at the same time. |
| 2022.04 | • You can now use the `databaseMapping` property in your Tableau <source ID> configuration file, to map a Tableau technical database name to the real database name.<br>• When providing connection definitions for Informatica PowerCenter, the `dbname` property is no longer case-sensitive.<br>• When integrating Informatica PowerCenter data sources, Collibra Data Lineage now correctly creates a technical lineage when `useCollibraSystemName` is set to `true`. |

| Changed in version | New lineage harvester improvements |
|---|---|
| 2022.03 | • By default, the lineage harvester no longer harvests images. If you want to include images, include the optional `excludeImages` property in your configuration file and set the value to `false`.<br>• When ingesting Tableau metadata, you can now leave empty the `collibraSystemName` property in your configuration file, even if the `useCollibraSystemName` property is set to `true`.<br>• The lineage harvester now correctly shows the help overview when you run the `--help` command.<br>• Hive source now skips harvesting DDL of exclusively locked tables.<br>• When you change the domain reference ID in the lineage harvester configuration file, Tableau assets are now successfully deleted from the previous domain and recreated in the new domain.<br>• You no longer see a Fiber Failed error while running the lineage harvester.<br>• Protobuf is upgraded to version 3.19.3.<br>• Fixed an issue that was causing incomplete technical lineage and stitching issues when using custom SQL in Tableau.<br>• Fixed an issue that resulted in a TableauHarvesterError when ingesting Tableau metadata via the linage harvester.<br>• Fixed a NullPointerException when no column data type is harvested.<br>• Fixed an issue that was causing the ingestion of Looker metadata to fail.<br>• Fixed an issue that was causing a JsonParseError when ingesting Tableau metadata. |
| 2022.02 | |

| Changed in version | New lineage harvester improvements |
|---|---|
| 1.4.4 | The lineage harvester now supports:<br>• Technical lineage for Matillion. Redshift and Snowflake projects in Matillion are supported.<br>• Snowflake syntax for the CONNECT BY clause. |
| 1.4.3 | The lineage harvester log output now includes Collibra Data Lineage server processing information. |
| 1.4.2 | Collibra Data Lineage has improved Teradata parsing. |

# Business Summary Lineage

The Business Summary Lineage is a representation of relations of the type "Data Element sources / targets Data Element" in a business diagram. It is not a separate diagram view, but refers to any diagram that contains that relation type. It allows you to trace data flows between registered databases and, as such, provides a summary of a technical lineage.

> Note Click here for an overview of the differences between Technical lineage and a diagram with Business Summary Lineage.

You can create a new diagram view including the Business Summary Lineage or you can select one of the existing diagram views that shows the relation "Data Element sources / targets Data Element" between Column assets of registered data sources and between BI assets and assets of registered data sources.

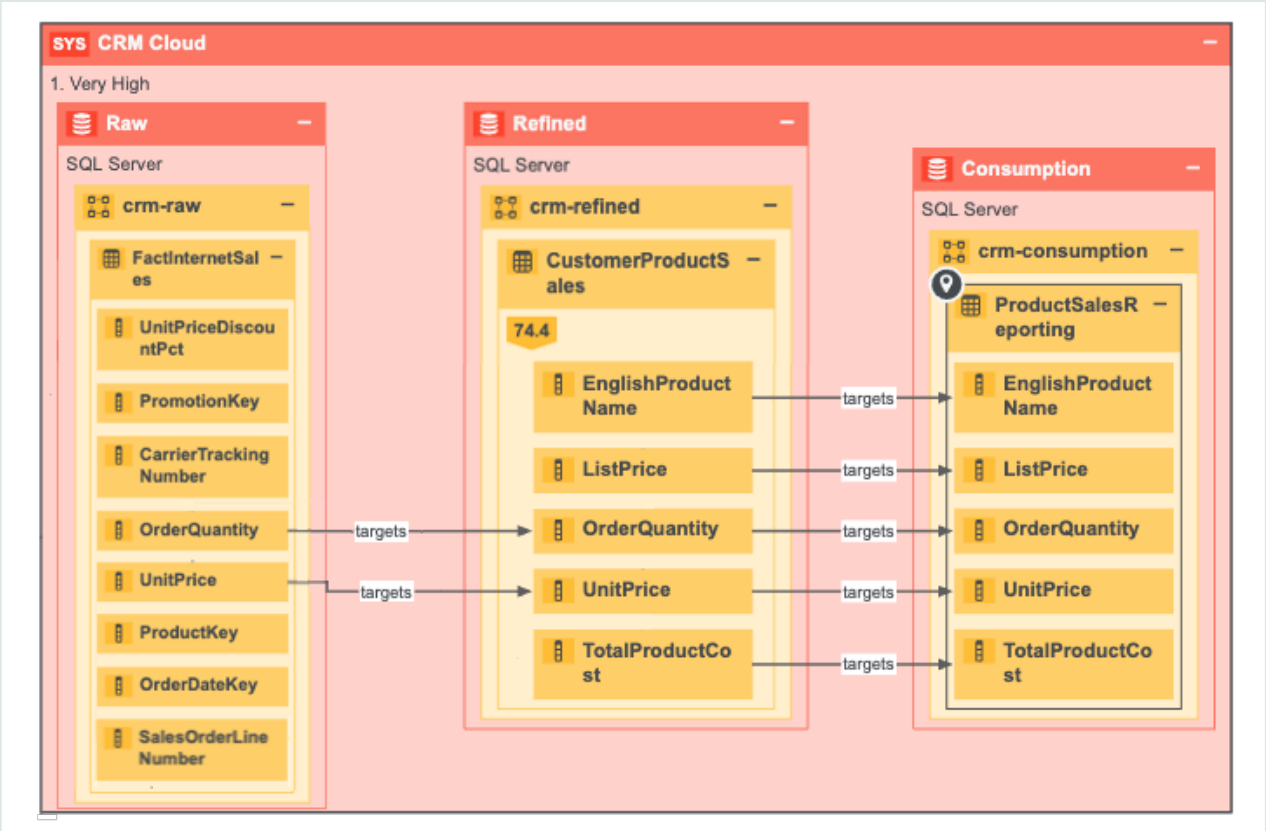
Before you can view a diagram with Business Summary Lineage, you have to:

- Register the data sources that you want to see in a diagram with Business Summary Lineage.
- Prepare a configuration file to create a technical lineage.
- Use the lineage harvester to upload the data sources in your configuration file to the Collibra Data Lineage service where they are scanned and processed.

Once the data sources are scanned, the Collibra Data Lineage service automatically pushes relations of the type "Data Element sources / targets Data Element" to Collibra Data Intelligence Cloud.

# Example of a diagram with Business Summary Lineage

In this business diagram, you see that the Column assets of the Table asset CustomerProductSales have a relation of the type "Data Element sources / targets Data Element" to Column assets of other Table assets.



# Differences between Technical lineage and diagrams with Business Summary Lineage

Technical lineage is a detailed lineage graph that shows where data objects are used and how they are transformed. A diagram with the Business Summary Lineage shows the relations between Data Assets in Data Catalog after stitching. Both map the flow of data, but a technical lineage provides a detailed overview of the data flow, while a diagram with Business Summary Lineage only provides a summary of it.

The Business Summary Lineage and a technical lineage are both visual representations of nodes. However, there are some key differences between them.

> Tip For information on the steps required to create a technical lineage, including how to prepare the Data Catalog physical data layer, see About technical lineage.

| Business Summary Lineage | Technical lineage |
|---|---|
| A diagram with a Business Summary Lineage helps Business Analysts and other business users to understand their data by providing a summary of the technical lineage. | A technical lineage helps Data Engineers, Data Architects and similar personas to easily navigate to data objects in the data flows and find relevant source code fragments by providing a detailed lineage graph. |
| A diagram containing Business Summary Lineage is accessible via the Diagram tab pane of all assets. | A technical lineage is accessible via the tab pane of all Table assets and Column assets. You can view a technical lineage via the tab pane of Table assets and Column assets if you added their database as data sources in the configuration file. |

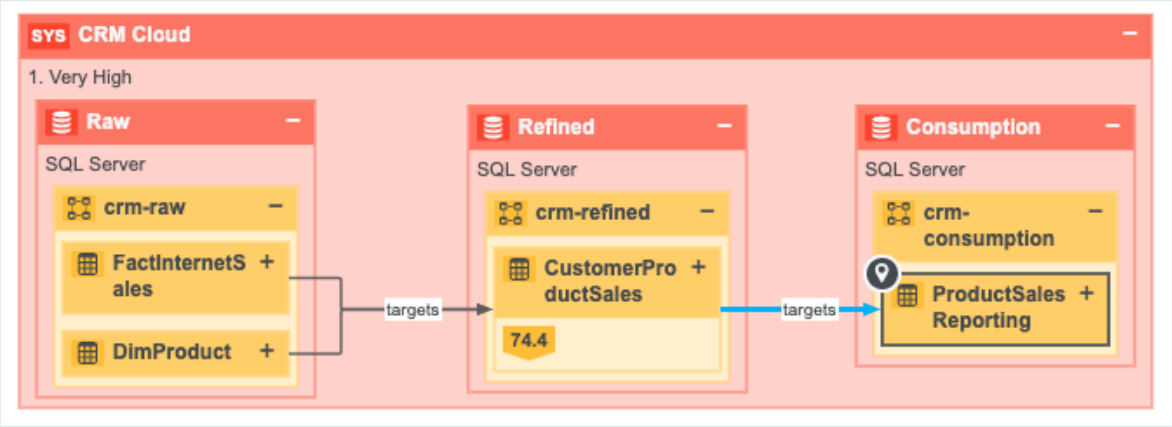
| Business Summary Lineage | Technical lineage |
| --- | --- |
| A diagram shows assets and relations as defined in its diagram view. In the case of a Business Summary Lineage, the diagram shows, amongst others, relations of the type "Data Element targets / sources Data Element" between assets that exist in Data Catalog. Relations of this type are automatically created as part of the technical lineage process. | A technical lineage shows relations of the type "Data Element targets / sources Data Element" between all data objects in the data source. Relations of this type are automatically created as part of the technical lineage process.<br><br>Note The data objects that you see in the technical lineage are:<br>• Data Element assets for which you created the technical lineage,<br>• Other objects, for example temporary tables and columns, that the lineage scanner collected from your data sources, but are not assets in Data Catalog. |
| A diagram with a Business Summary Lineage shows how registered data sources relate to each other. | Technical lineage shows how all data sources for which you create a technical lineage relate to each other. If the data source, or a part of the data source, is not registered in Data Catalog, the dependencies between the data elements in the data sources are still shown. |

Example
You have created a technical lineage for four different databases:

- The first database, *Oracle*, is not ingested in Data Catalog and therefore has no assets in Data Catalog.
- The second database, *Raw*, contains tables that are ingested in Data Catalog, but also tables that are not ingested and therefore are not assets.
- The third and fourth database, *Refined* and *Consumption*, only contains data objects that are also assets in Data Catalog.

Technical lineage shows the data flow from all data objects in the first database, to the second, the third, and the fourth. Databases or data objects that are not ingested in Data Catalog and therefore are not assets, have a gray background.

A diagram with Business Summary Lineage only shows the relations between data objects that are also assets in Data Catalog, which means the data flow from assets in the second database to assets in the third, to assets in the fourth. The first database, which wasn't ingested, will not be shown on the diagram.




# Dependencies

A dependency is a data object that is targeted by another data object. This is represented by a relation of the type "Data Element targets / sources Data Element", where the dependency is the tail.

There are two type of dependencies:

- a direct dependency: a data object that is the tail of a relation of the type "Data Element targets / sources Data Element".

  > Example  If column A targets column B, then column B is the direct dependency of column A.

- an indirect dependency: a data object that is the target of a direct or another indirect dependency.

  > Example  Column A targets column B, which on its turn targets column C. This means that column A indirectly targets column C, so column C is the indirect dependency of column A.

# Working with Tableau

Tableau is business intelligence software that helps people see and understand their data. Integrating Tableau in Collibra Data Intelligence Cloud enables you to see metadata from Tableau Server and Tableau Online in CollibraData Catalog.

In this section, we describe how you can ingest Tableau metadata in Collibra Data Catalog and synchronize the metadata using the lineage harvester, a standalone Java application.

Warning As of October 2022, Tableau is enforcing multi-factor authentication for Tableau Cloud Admin users. However, the lineage harvester doesn't support multi-factor authentication. Therefore, Tableau Cloud users with an Admin role must use token-based authentication. This does not affect Tableau Server users or Tableau Cloud users with an Explorer role.

Important Please note the following important points regarding this integration method:

- It is a cloud-only feature.
- The new Tableau operating model is only available in Collibra versions 2021.10 and newer.
- The two Tableau integration methods—Tableau integration via the Data Catalog and the new integration method via lineage harvester—coexist, and you are free to use the method of your choosing.

# Features and limitations of Tableau integration via the lineage harvester

This section describes the features and limitations of using the lineage harvester to create Tableau assets and data in Data Catalog.

> Important Data Catalog uses Tableau's REST API to get metadata information and follows Tableau's requirements regarding authentication methods. As such, you need a Tableau user with access to the relevant Tableau sites. For more information, see the Tableau documentation.

## Features

The following table shows the features specific to the two integration methods.

| Feature | Integration via Data Catalog UI | Integration via the lineage harvester |
|---|---|---|
| Catalog ingestion | ✓ | ✓ |
| Technical lineage | | ✓ |
| Automatic stitching | | ✓ |
| Embedded data source connectivity | | ✓ |
| Custom SQL parsing | | ✓ |

| Feature | Integration via Data Catalog UI | Integration via the lineage harvester |
|---|---|---|
| On-prem credential storage | | ✓ |
| Ingestion via Explorer role (with the Data Management Add-On) | | ✓ |

# Limitations

Currently, there are still a couple of limitations to integrating Tableau metadata via the lineage harvester:

- You ingest Tableau via the lineage harvester, instead of via the Data Catalog UI. All changes to the Tableau ingestion must be configured in the lineage harvester configuration file, instead of Data Catalog.
- The **Stitch** button is part of the legacy Tableau integration (via Data Catalog) and does not work when integrating Tableau via the lineage harvester, as stitching is done automatically via this method.
- We partially support Unions and Joins. For example, Unions created via the Tableau UI are not represented in Data Catalog. Tableau Data Sources created via custom SQL are supported.

# Tableau terminology

The following table shows the Tableau terminology and corresponding asset types and terminology in Collibra Data Intelligence Cloud.

| Tableau term | Description | Collibra equivalent |
|---|---|---|
| Site | A site is a stand-alone collection of content, such as projects, workbooks and users. Each site has its own URL and its own set of users. | Subcommunity and Tableau Site asset |

| Tableau term | Description | Collibra equivalent |
|---|---|---|
| Project | A project organizes related content resources. Content resources are workbooks, views and data sources. | Tableau Project asset |
| Workbook | A workbook is a collection of views. | Tableau Workbook asset |
| Dashboard | A dashboard is a collection of views from multiple worksheets. | Tableau Dashboard asset |
| Worksheet | A worksheet contains a single view, along with shelves, legends, and the Data pane. | Tableau Worksheet asset |
| Tableau data source | Tableau Data Sources consist of metadata that describe the connection information, information about how to access or refresh the data and customizations. | Tableau Data Model asset |
| Dimension | Dimensions contain qualitative values (such as names, dates, or geographical data). | Attribute type Role in Report on a Tableau Data Attribute asset page |
| Measure | Measures contain numeric, quantitative values that you can measure. | Attribute type Role in Report on a Tableau Data Attribute asset page |
| Tableau data attribute | Tableau Data Attributes define a property of a Tableau data entity. | Tableau Data Attribute asset |

| Tableau term | Description | Collibra equivalent |
|---|---|---|
| Tableau data entity | Tableau Data Entities are an abstraction of the physical implementation of database tables, used for Tableau report creation. | Tableau Data Model asset |
| Tableau data model | Tableau Data Models are an abstraction for the physical implementation of databases, schemas, files, etc., used for Tableau report creation. | Tableau Data Model asset |
| Tableau server | A Tableau server is a server on which Tableau users can publish data sources, as a means to share the data connections they've defined. | Tableau Server asset |

# Tableau asset types and domain types

The Tableau integration of Collibra Data Intelligence Cloud uses a specific subset of asset types and domain types. All of these come out of the box with your software.

The following table shows the asset and domain types that are used for the Tableau integration. Above each asset type you can see the parent asset types in the breadcrumbs.

| Asset type | Description | Domain type |
| --- | --- | --- |
| Business Asset ▸ Business Dimension ▸ BI Folder ▸ Tableau Project | Collection of Tableau workbooks and data sources. | BI Catalog |
| Business Asset ▸ Business Dimension ▸ BI Folder ▸ Tableau Site | Collection of content (workbooks, data sources, users, …) that's walled off from any other content on that instance of Tableau Server. | BI Catalog |
| Business Asset ▸ Report ▸ BI Report ▸ Tableau View ▸ Tableau Dashboard | A collection of several worksheets and supporting information, shown on a single screen, so that you can simultaneously compare and monitor a variety of data. | BI Catalog |
| Business Asset ▸ Report ▸ BI Report ▸ Tableau View ▸ Tableau Worksheet | A worksheet is a single sheet on which you can build views of your data. | BI Catalog |
| Business Asset ▸ Report ▸ BI Report ▸ Tableau Workbook | Collection of sheets. A sheet can be a worksheet, a dashboard or a story. | BI Catalog |

| Asset type | Description | Domain type |
| --- | --- | --- |
| Data Asset ▸ Data Element ▸ Data Attribute ▸ BI Data Attribute ▸ Tableau Data Attribute | A specification that defines a property of a Tableau data entity.<br>Examples: CustomerBirthDate, EmployeeFirstName. | BI Catalog |
| Data Asset ▸ Data Structure ▸ Data Model ▸ BI Data Model ▸ Tableau Data Model | An abstraction from the physical implementation of database, schema, file, etc., used for Tableau report creation. | BI Catalog |
| Technology Asset ▸ Server ▸ BI Server ▸ Tableau Server | A visual analytics platform for creating interactive dashboards and rich visualisations | BI Catalog |

# Tableau operating model

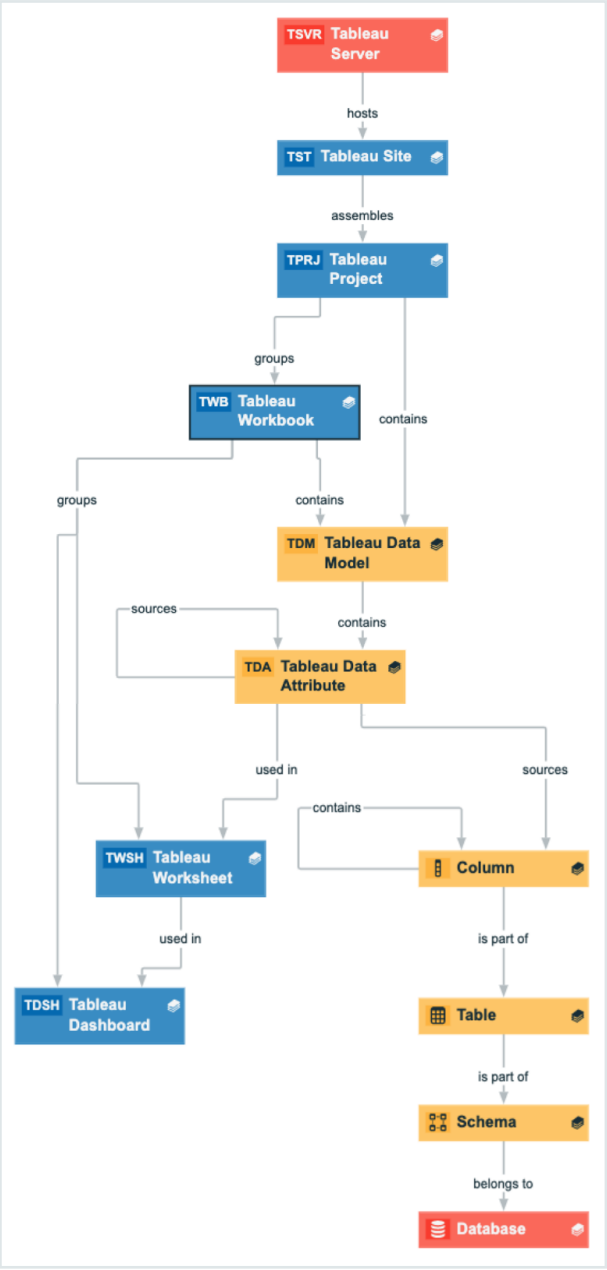
Synchronizing Tableau data means ingesting metadata from Tableau to your Collibra Data Intelligence Cloud environment. The metadata is represented as assets of specific types and their characteristics.

Note

- The assets have the same names as their counterparts in Tableau.
- Some asset types are only created if the Tableau user has specific permissions.
- Relations that were created between Tableau assets and other assets via a relation type in the Tableau operating model, are deleted upon synchronization. The same is true of any attribute types in the operating model that you add to Tableau assets. To ensure that the characteristics you add to Tableau assets are not deleted upon synchronization, be sure to use characteristics that are not part of the Tableau operating model.

The following image shows the relations between Tableau asset types.

You can easily recreate this diagram view in your Collibra environment. See Create a Tableau operating model diagram view.




# Harvested metadata per asset type

This table shows the metadata for each Tableau asset type and the resource ID for each asset type and metadata.

| Asset type | Synchronized metadata | Resource ID |
|---|---|---|
| Tableau Server<br><br>Resource ID: 00000000-0000-0000-0000-110000000005 | Description | 00000000-0000-0000-0000-000000003114 |
| | Server hosts / is hosted in Business Dimension | 00000000-0000-0000-0000-120000000000 |
| | URL: The link to the data in Tableau | 00000000-0000-0000-0000-000000000258 |
| Tableau Site<br><br>Resource ID: 00000000-0000-0000-0000-110000000000 | BI Folder assembles / Is assembled in BI Folder | 00000000-0000-0000-0000-120000000001 |
| | Description | 00000000-0000-0000-0000-000000003114 |
| | Server hosts / is hosted in Business Dimension | 00000000-0000-0000-0000-120000000000 |
| | URL: The link to the data in Tableau | 00000000-0000-0000-0000-000000000258 |

| Asset type | Synchronized metadata | Resource ID |
|---|---|---|
| Tableau Project<br><br>Resource ID: 00000000-0000-0000-0000-110000000001 | Description | 00000000-0000-0000-0000-000000003114 |
| | Owner in source<br><br>• The only harvested metadata are email addresses. To harvest this metadata, you need to enable the Metadata API by setting the `restOnly` property in your lineage harvester configuration file to `false`. | 00000000-0000-0000-0000-200000000001 |
| | BI Folder assembles / is assembled in BI Folder | 00000000-0000-0000-0000-120000000001 |
| | Business Dimension groups / is grouped into Report | 00000000-0000-0000-0000-120000000002 |
| | BI Folder contains / contained in Data Asset | 00000000-0000-0000-0000-120000000014 |

| Asset type | Synchronized metadata | Resource ID |
|---|---|---|
| Tableau Workbook<br><br>Resource ID: 00000000-0000-0000-0000-110000000002 | Description | 00000000-0000-0000-0000-000000003114 |
| | Certified | 00000000-0000-0000-0001-000500000001 |
| | Document creation date | 00000000-0000-0000-0000-000000000260 |
| | Document modification date | 00000000-0000-0000-0000-000000000261 |
| | Document size | 00000000-0000-0000-0000-000000000259 |
| | Owner in source<br><br>• The only harvested metadata are email addresses. To harvest this metadata, you need to enable the Metadata API by setting the `restOnly` property in your lineage harvester configuration file to `false`. | 00000000-0000-0000-0000-200000000001 |
| | Report Image | 00000000-0000-0000-0000-000000000262 |
| | URL: The link to the data in Tableau | 00000000-0000-0000-0000-000000000258 |
| | Report groups / is grouped into Report | 00000000-0000-0000-0000-120000000004 |

| Asset type | Synchronized metadata | Resource ID |
|---|---|---|
| | Tableau Workbook contains / contained in Tableau Data Model | 00000000-0000-0000-0000-120000000020 |
| | Business Dimension groups / is grouped into Report | 00000000-0000-0000-0000-120000000002 |

| Asset type | Synchronized metadata | Resource ID |
|---|---|---|
| Tableau Dashboard<br><br>Resource ID: 00000000-0000-0000-0001-110000000301<br><br>Assets of this type are only created if the Tableau user has the Download/Save As permission on the workbook. | Certified | 00000000-0000-0000-0001-000500000001 |
| | Document creation date | 00000000-0000-0000-0000-000000000260 |
| | Document modification date | 00000000-0000-0000-0000-000000000261 |
| | Report image: The image of the report.<br><br>Images are downloaded and stored in Data Catalog. You can configure the maximum file size and content types of the Tableau images in the Collibra DGC service settings. | 00000000-0000-0000-0000-000000000262 |
| | URL: The link to the data in Tableau | 00000000-0000-0000-0000-000000000258 |
| | Visible on server | 00000000-0000-0000-0000-000000000265 |
| | Report groups / is grouped into Report | 00000000-0000-0000-0000-120000000004 |
| | Report uses / used in Data Attribute | 00000000-0000-0000-0000-120000000021 |
| | Report uses / used in Report | 00000000-0000-0000-0000-120000000007 |

| Asset type | Synchronized metadata | Resource ID |
|---|---|---|
| Tableau Worksheet<br><br>Resource ID: 00000000-0000-0000-0001-110000000300<br><br>Assets of this type are only created if the Tableau user has the Download/Save As permission on the workbook. | Certified | 00000000-0000-0000-0001-000500000001 |
| | Document creation date | 00000000-0000-0000-0000-000000000260 |
| | Document modification date | 00000000-0000-0000-0000-000000000261 |
| | Report image: The image of the report.<br><br>Images are downloaded and stored in Data Catalog. You can configure the maximum file size and content types of the Tableau images in the Collibra DGC service settings. | 00000000-0000-0000-0000-000000000262 |
| | URL: The link to the data in Tableau | 00000000-0000-0000-0000-000000000258 |
| | Visible on server | 00000000-0000-0000-0000-000000000265 |
| | Report groups / is grouped into Report | 00000000-0000-0000-0000-120000000004 |
| | Report uses / used in Data Attribute | 00000000-0000-0000-0000-120000000021 |
| | Report uses / used in Report | 00000000-0000-0000-0000-120000000007 |

| Asset type | Synchronized metadata | Resource ID |
|---|---|---|
| Tableau Data Attribute<br><br>Resource ID: 00000000-0000-0000-0000-110000000010<br><br>Assets of this type are only created if the Tableau user has the Download/Save As permission on the data source. | Calculation Rule | 00000000-0000-0000-0000-000000003117 |
| | Data Type: The data type of a data asset, as it is declared by the data source. | 00000000-0000-0000-0001-000500000005 |
| | Role in Report | 00000000-0000-0000-0000-000000000266 |
| | BI Data Model contains / is part of BI Data Attribute | 00000000-0000-0000-0000-000000007196 |
| | Data Element targets / sources Data Element | 00000000-0000-0000-0000-000000007069 |
| | Report uses / used in Data Attribute | 00000000-0000-0000-0000-120000000021 |

| Asset type | Synchronized metadata | Resource ID |
|---|---|---|
| Tableau Data Model<br><br>Resource ID: 00000000-0000-0000-0000-110000000008<br><br>Assets of this type are only created if the Tableau user has the Download/Save As permission on the data source. | Certified | 00000000-0000-0000-0001-000500000001 |
| | Document creation date | 00000000-0000-0000-0000-000000000260 |
| | Document modification date | 00000000-0000-0000-0000-000000000261 |
| | Original Name: The name of the data source in Tableau | 00000000-0000-0000-0001-000500000032 |
| | Owner in source<br><br>• The only harvested metadata are email addresses. To harvest this metadata, you need to enable the Metadata API by setting the `restOnly` property in your lineage harvester configuration file to `false`. | 00000000-0000-0000-0000-200000000001 |
| | BI Data Model contains / is part of BI Data Attribute | 00000000-0000-0000-0000-000000007196 |
| | Business Dimension source / is source of System | 00000000-0000-0000-0000-120000000003 |
| | Tableau Workbook contains / contained in Tableau Data Model | 00000000-0000-0000-0000-120000000020 |

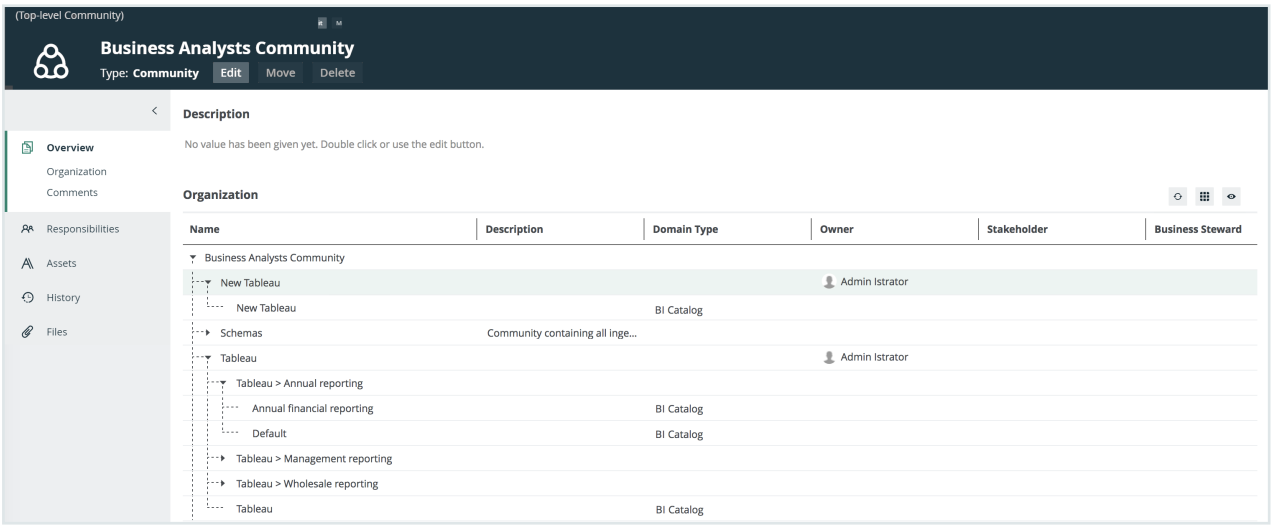
# Additional information

For the Owner in source attribute, the following rules apply:

- If the system creates a Tableau data object and the Tableau data object does not have a user ID, the Owner in source attribute is shown as System on the asset page.
- If the user who created a Tableau data object no longer exists, the Owner in source attribute is shown as empty on the asset page.

# Example of ingested Tableau metadata

The following image shows an example structure after synchronizing Tableau.




# Recommended hierarchy within a domain

You can enable hierarchies for the domain (or domains) in which your Tableau assets were ingested. Doing so makes it easier to understand the relation between your Tableau assets, when viewing the assets on the domain page.

Follow these steps to enable and configure the recommended hierarchy.

## Steps

1. Open the domain page of the relevant BI Catalog domain.
2. In the content toolbar, click .
   » The **Configure Hierarchy** dialog box appears.
3. Select **Enable Hierarchy**.
4. Select **Single path**.
5. Start typing and select each of the following relation types:
   - Server **hosts** Business Dimension
   - BI Folder **assembles** BI Folder
   - Business Dimension **groups** Report
   - Report **groups** Report
   - Report **uses** Report
   - Report **uses** Data Attribute
   - BI Data Attribute **is part of** BI Data Model
6. Click **Apply**.

> Note In an asset view, if any asset is deleted, for example via synchronization or manual deletion, the view is recreated and the hierarchy is lost. In this case, you can again enable and configure the recommended hierarchy.

# Create a Tableau operating model diagram view

You can create a Tableau-specific diagram view, to visualize the operating model. The following procedure provides instruction on how to quickly create a new diagram view by copying and pasting the JSON code in the diagram view text editor.

## Steps

1. Open an asset page.
2. In the tab pane, click **Diagram**.
   » The diagram appears in the default diagram view.
3. Click + to add a new view.
4. Click the **Text** tab, to switch to the diagram view text editor.
5. Click **Show me the JSON code** below this procedure, to expand the code.
6. Paste the code in diagram view text editor.

7. Click **Save**.
8. Edit the name and description of the diagram view, to suit your needs.

### Show me the JSON code


```
{
   "nodes": [
        {
            "id": "Tableau Workbook",
            "type": {
            "id": "00000000-0000-0000-0000-110000000002"
            },
            "layoutRegion": "context"
        },
        {
            "id": "Tableau Dashboard",
            "type": {
              "id": "00000000-0000-0000-0001-110000000301"
            },
            "layoutRegion": "context"
        },
        {
            "id": "Tableau Worksheet",
            "type": {
              "id": "00000000-0000-0000-0001-110000000300"
            },
            "layoutRegion": "context"
        },
        {
            "id": "Tableau Data Model",
            "type": {
              "id": "00000000-0000-0000-0000-110000000008"
            },
            "layoutRegion": "context"
        },
        {
            "id": "Tableau Project",
            "type": {
              "id": "00000000-0000-0000-0000-110000000001"
            },
            "layoutRegion": "context"
        },
        {
            "id": "Tableau Site",
            "type": {
              "id": "00000000-0000-0000-0000-110000000000"
            },
            "layoutRegion": "context"
        },
        {
```

```
            "id": "Tableau Server",
            "type": {
              "id": "00000000-0000-0000-0000-110000000005"
            },
            "layoutRegion": "context"
        },
        {
            "id": "Tableau Data Attribute",
            "type": {
              "id": "00000000-0000-0000-0000-110000000010"
            },
            "layoutRegion": "context"
        },
        {
            "id": "Column",
            "type": {
              "id": "00000000-0000-0000-0000-000000031008"
            },
            "layoutRegion": "context"
        },
        {
            "id": "Table",
            "type": {
              "id": "00000000-0000-0000-0000-000000031007"
            },
            "layoutRegion": "context"
        },
        {
            "id": "Schema",
            "type": {
              "id": "00000000-0000-0000-0001-000400000002"
            },
            "layoutRegion": "context"
        },
        {
            "id": "Database",
            "type": {
              "id": "00000000-0000-0000-0000-000000031006"
            },
            "layoutRegion": "context"
        }
    ],
    "edges": [
        {
            "from": "Tableau Project",
            "to": "Tableau Workbook",
            "label": "",
            "style": "boxing",
            "type": {
              "id": "00000000-0000-0000-0000-120000000002"
```

```
            },
            "roleDirection": true
        },
        {
            "from": "Tableau Site",
            "to": "Tableau Project",
            "label": "",
            "style": "boxing",
            "type": {
              "id": "00000000-0000-0000-0000-120000000001"
            },
            "roleDirection": true
        },
        {
            "from": "Tableau Server",
            "to": "Tableau Site",
            "label": "",
            "style": "boxing",
            "type": {
              "id": "00000000-0000-0000-0000-120000000000"
            },
            "roleDirection": true
        },
        {
            "from": "Tableau Data Model",
            "to": "Tableau Data Attribute",
            "label": "",
            "style": "boxing",
            "type": {
              "id": "00000000-0000-0000-0000-000000007196"
            },
            "roleDirection": true
        },
        {
            "from": "Tableau Data Attribute",
            "to": "Tableau Data Attribute",
            "label": "",
            "style": "arrow",
            "type": {
              "id": "00000000-0000-0000-0000-000000007069"
            },
            "roleDirection": false
        },
        {
            "from": "Tableau Workbook",
            "to": "Tableau Data Model",
            "label": "",
            "style": "boxing",
            "type": {
              "id": "00000000-0000-0000-0000-120000000020"
```

```
            },
            "roleDirection": true
        },
        {
            "from": "Tableau Project",
            "to": "Tableau Data Model",
            "label": "",
            "style": "arrow",
            "type": {
              "id": "00000000-0000-0000-0000-120000000014"
            },
            "roleDirection": true
        },
        {
            "from": "Column",
            "to": "Column",
            "label": "",
            "style": "boxing",
            "type": {
              "id": "00000000-0000-0000-0000-000000007042"
            },
            "roleDirection": false
        },
        {
            "from": "Column",
            "to": "Table",
            "label": "",
            "style": "boxed",
            "type": {
              "id": "00000000-0000-0000-0000-000000007042"
            },
            "roleDirection": true
        },
        {
            "from": "Table",
            "to": "Schema",
            "label": "",
            "style": "boxed",
            "type": {
              "id": "00000000-0000-0000-0000-000000007043"
            },
            "roleDirection": false
        },
        {
            "from": "Schema",
            "to": "Database",
            "label": "",
            "style": "boxed",
            "type": {
              "id": "00000000-0000-0000-0000-000000007024"
```

```
            },
            "roleDirection": false
        },
        {
            "from": "Tableau Data Attribute",
            "to": "Tableau Worksheet",
            "label": "",
            "style": "arrow",
            "type": {
              "id": "00000000-0000-0000-0000-120000000021"
            },
            "roleDirection": false
        },
        {
            "from": "Tableau Workbook",
            "to": "Tableau Worksheet",
            "label": "",
            "style": "boxing",
            "type": {
              "id": "00000000-0000-0000-0000-120000000004"
            },
            "roleDirection": true
        },
        {
            "from": "Tableau Workbook",
            "to": "Tableau Dashboard",
            "label": "",
            "style": "boxing",
            "type": {
              "id": "00000000-0000-0000-0000-120000000004"
            },
            "roleDirection": true
        },
        {
            "from": "Tableau Worksheet",
            "to": "Tableau Dashboard",
            "label": "",
            "style": "arrow",
            "type": {
              "id": "00000000-0000-0000-0000-120000000007"
            },
            "roleDirection": false
        },
        {
            "from": "Tableau Data Attribute",
            "to": "Column",
            "label": "",
            "style": "arrow",
            "type": {
              "id": "00000000-0000-0000-0000-000000007069"
```

```
            },
            "roleDirection": false
        }
    ],
    "showOverview": false,
    "enableFilters": true,
    "showLabels": true,
    "showFields": true,
    "showLegend": true,
    "showPreview": true,
    "visitStrategy": "directed",
    "layout": "HierarchyLeftRight",
    "maxNodeLabelLength": 50,
    "maxEdgeLabelLength": 30,
    "layoutOptions": {
        "compactGroups": false,
        "componentArrangementPolicy": "topmost",
        "edgeBends": true,
        "edgeBundling": true,
        "edgeToEdgeDistance": 5,
        "minimumLayerDistance": "auto",
        "nodeToEdgeDistance": 5,
        "orthogonalRouting": true,
        "preciseNodeHeightCalculation": true,
        "recursiveGroupLayering": true,
        "separateLayers": true,
        "webWorkers": true,
        "nodePlacer": {
            "barycenterMode": true,
            "breakLongSegments": true,
            "groupCompactionStrategy": "none",
            "nodeCompaction": false,
            "straightenEdges": true
        }
    }
}
```

# Supported data sources in Tableau

Tableau is business intelligence software that can integrate with various data sources. When you ingest Tableau metadata, Collibra Data Lineage tries to automatically stitch the metadata to data sources registered in Data Catalog. It also creates a Technical lineage that shows where metadata is used and how it transforms.

The following table shows the supported data sources in Tableau that have been tested, and whether or not technical lineage and stitching is supported for the data source.

We cannot guarantee that stitching works as expected for other data sources or versions.

> Tip For stitching, you must correctly prepare the Data Catalog physical data layer.

| Data source | Version | Support for technical lineage | Support for stitching |
|---|---|---|---|
| Amazon Redshift | 1.2.34.1058 and newer | Yes | Yes |
| Azure SQL server | Newest version | Yes | Yes |
| Azure SQL Data Warehouse | Newest version | Yes | Yes |
| Azure Synapse Analytics | Newest version | Yes | Yes |
| Dremio | 20.0.0 | Yes | Yes |
| Google BigQuery | Newest version | Yes | Yes |
| Greenplum | 6.10 and newer | Yes | Yes |
| HiveQL (SQL-like statements) | 2.3.5 and newer | Yes | Yes |
| IBM DB2 | 11.5 and newer | Yes | Yes |
| Oracle | 11g, 12c and newer | Yes | Yes |
| PostgreSQL | 9.4, 9.5 and newer | Yes | Yes |

| Data source | Version | Support for technical lineage | Support for stitching |
|---|---|---|---|
| Microsoft SQL Server | 2014, 2016 and newer | Yes | Yes |
| MySQL | 5.7, 8 and newer | Yes | Yes |
| Netezza | 7.2.1.0 and newer | Yes | Yes |
| SAP Hana | 2.00.40 and newer | Yes | Yes |
| Snowflake | Newest version | Yes | Yes |
| Spark SQL | 2.4.3 and newer | Yes | Yes |
| Sybase Adaptive Server Enterprise | 16.0 SP02 and newer | Yes | Yes |
| Teradata | 15.0, 16.20.07.01 and newer | Yes | Yes |

# Automatic stitching

Stitching is a process that creates relations between database columns that are Column assets in Collibra Data Intelligence Cloud and BI assets representing the same database. Specifically, stitching creates relations between the following assets:

- The assets that are created when you ingest Tableau.
- The assets that are created when you register a data source or import assets.

For Collibra Data Lineage to stitch the assets to the data objects, you must prepare the Data Catalog physical data layer to create the database > schema > table > column or system > database > schema > table > column hierarchy. For more information, see Prepare the Data Catalog physical data layer for Tableau stitching.

The lineage harvester harvests the Tableau source code and sends it to the Collibra Data Lineage service. The Collibra Data Lineage also collects the full names of assets ingested in Data Catalog and stitches them to data objects collected from Tableau. After processing the metadata, the Collibra Data Lineage service ingests the Tableau assets and their characteristics in Data Catalog. Tableau assets that are stitched now show a relation of the type "Data Element targets / sources Data Element" to the stitched asset. This relation type is also visualized in a technical lineage.

To clarify, the Tableau Data Attribute is the target of the Column, and the Column is the source of the Tableau Data Attribute

> Note
> - If the Column asset is from a data source that is not supported by technical lineage, a standard SQL parser is used to try to visualize the column in a technical lineage, but the technical lineage might not be complete.
> - When you ingest Tableau metadata, a technical lineage for Tableau Data Attribute assets is automatically created.

# Stitching: matching the full paths of assets

To stitch assets in Data Catalog to data objects collected by the lineage harvester, the Collibra Data Lineage service looks at the full path of the assets in Data Catalog and the full path of Tableau assets. If the full paths match, the Collibra Data Lineage service automatically stitches them.

> Note Ensure that you correctly prepare the Data Catalog physical data layer.

The Technical lineage Stitching tab page shows the full paths of assets in Data Catalog and data objects collected from Tableau. To fix stitching issues, you can look up the full paths and make sure they match.

# Technical lineage for Tableau

When you ingest Tableau metadata in Data Catalog, a technical lineage for Tableau Data Attribute assets is automatically created.

# Permissions

You can see the technical lineage of Tableau assets by clicking on the Technical lineage tab on the asset page of any Table or Column asset in Data Catalog when you have a Data Catalog global role with the Catalog and Technical lineage global permissions.

# Technical lineage graph

The technical lineage graph shows relations of the type "Data Element sources / targets Data Element" between Tableau assets and other data objects in the data flow, for example between a Column asset and a Tableau Data Attribute asset. These relations are created during the Tableau ingestion process as a result of automatic stitching.

> Note If you use a Tableau <source ID> configuration file and don't specify a value for the relevant `collibraSystemName` property, the designation "UNDEFINED" will be shown in the technical lineage.

## Example

The following technical lineage shows how data flows from a PostgreSQL data source to Tableau. It shows relations of the type "Data Element sources / targets Data Element" between the Column assets of the database and Tableau Data Attribute assets in Tableau. For example, Column asset *DEPARTMENT_NAME* has a relation of the type "Data Element sources / targets Data Element" to the Tableau Data Attribute asset *department_name*.

# Sources tab page

The Sources tab page shows the transformation details that the Collibra Data Lineage service analyzed and processed and the results of this analysis. The success rate of the analysis indicates how complete the technical lineage is. There are a few limitations that prevent the Collibra Data Lineage service from processing all Tableau metadata.

> Important The Collibra Data Lineage service might not be able to process all complex Tableau metadata. This means that the success rate of a Tableau ingestion might not be 100%.

# Overview Tableau integration steps

The Tableau integration enables you to harvest Tableau metadata and create new Tableau assets in Data Catalog. Collibra Data Intelligence Cloud analyzes and processes the metadata and presents it as specific asset types, retaining their original names.

## Steps

The table below shows the steps and prerequisites required to integrate Tableau in Collibra via the lineage harvester.

| Step | What? | Description | Prerequisites |
|---|---|---|---|
| 1 | Set up Tableau. | Before you start the Tableau integration in Data Catalog, make sure that the lineage harvester can reach the Tableau metadata. Perform these tasks before you start the actual Tableau ingestion process.<br><br>Warning Because these tasks are performed outside of Collibra, it is possible that the content changes without us knowing. We strongly recommend that you carefully read the source documentation. | • You have a Tableau subscription. |
| 2 | Create a new domain. | Before you can ingest Tableau metadata, you have to create a new domain or choose an existing domain to store the new Tableau assets.<br><br>Warning If you are using Collibra Data Intelligence Cloud 2021.11 or older, you have to add all Tableau attributes in the operating model to a scope and create a scoped assignment before you ingest Tableau via the lineage harvester. For complete information and step-by-step instruction, see Tableau general troubleshooting. | You have a resource role with the following resource permissions:<br><br>• Domain: Add |

| Step | What? | Description | Prerequisites |
|---|---|---|---|
| 3 | Prepare the physical data layer. | You prepare Data Catalog's physical data layer to enable Data Catalog to automatically stitch the Tableau assets to existing assets in Data Catalog.<br><br>Important In the global assignment of each asset type included in the Tableau operating model, ensure that none of the characteristics that are in the operating model have a maximum cardinality of “0”. If the maximum cardinality is set to “0” for any such characteristics, ingestion will fail. | • You have a global role with the Catalog global permission, for example Catalog Author.<br>• You have set up the JDBC driver of your source data, for example Snowflake.<br>• You have a resource role with the following resource permissions on the **Schema** community:<br>◦ Asset > add<br>◦ Attribute > add<br>◦ Domain > add<br>◦ Attachment > add<br>• You have the permissions to retrieve the metadata of the following database components through the JDBC Driver |

| Step | What? | Description | Prerequisites |
|---|---|---|---|
| | | | Database Metadata methods:<br>◦ Schemas<br>◦ Tables<br>◦ Columns |
| 4 | Download and install the lineage harvester | You use the lineage harvester to trigger the creation of Tableau assets, their relations and a technical lineage in Data Catalog.<br><br>You can download the lineage harvester from the Collibra Product Resource Downloads page. | • Your environment meets the system requirements to install and use the lineage harvester. |

| Step | What? | Description | Prerequisites |
|---|---|---|---|
| 5 | Prepare the lineage harvester configuration file and run the lineage harvester. | You create a lineage harvester configuration file with Tableau connection information and run the lineage harvester to import the results of the Tableau integration and the technical lineage for Tableau into Data Catalog.<br><br>As a result, Collibra creates new Tableau assets in Data Catalog and imports relations between these assets. It also creates a technical lineage for Tableau assets and other data sources in the lineage harvester configuration file. | • You have downloaded the lineage harvester version 2022.02 or newer.<br>• Your environment meets the system requirements to install and run the lineage harvester.<br>• You have a global role with the Catalog global permission, for example Catalog Author.<br>• You have a global role with the Technical lineage global permission.<br>• You have a global role with the Data Stewardship Manager global permission.<br>• A resource role with the following |

| Step | What? | Description | Prerequisites |
| --- | --- | --- | --- |
| | | | resource permission on the community level in which you created the BI Data Catalog domain:<br>◦ Asset: add<br>◦ Attribute: add<br>◦ Domain: add<br>◦ Attachment: add |

| Step | What? | Description | Prerequisites |
|---|---|---|---|
| 6 | View the Tableau assets and technical lineage | After the Tableau metadata is ingested in Data Catalog, you can go to the domain where you ingested Tableau and see the list of ingested Tableau assets. These assets are automatically stitched to existing assets in Data Catalog.<br><br>You can also view the Tableau technical lineage.<br><br>Warning When you run the lineage harvester, Collibra Data Lineage creates all Tableau assets in a single BI Catalog domain. We highly recommend that you do not move these assets to another domain. If you move assets to another domain, they will be deleted and recreated in the initial BI Catalog domain when you synchronize Tableau. As a consequence, all manually added data of those assets is lost. | You have a Data Catalog global role with the Catalog and Technical lineage global permissions. |

# Naming convention

When you synchronize Tableau, Collibra follows a strict naming convention for the names of the new assets. Each asset has a display name and full name. The full name represents the asset path from asset to the database it belongs to. You can freely edit the display name. However, you should never edit the full name, because Data Catalog may need it to synchronize and stitch data sources. This may cause unexpected results and break the synchronization process.

> Warning We strongly recommend that you not edit the full names of any Tableau assets. Doing so will likely lead to errors during the synchronization process.

# Set up Tableau

Before you ingest Tableau metadata in Data Catalog, set up Tableau.

# Tableau versions and licenses

Before you ingest Tableau metadata in Data Catalog via the lineage harvester, you must ensure that the lineage harvester can access and harvest the Tableau metadata.

> Important If you want to create a technical lineage and stitch your Tableau assets to assets in Data Catalog, you must enable the Tableau metadata API in Tableau.

## Supported versions

- 2020.2
- 2020.3
- 2020.4
- 2021.1
- 2021.2
- 2021.3
- 2021.4
- 2022.01

## License

License the Data Management Add-on if you use the Add-on. For more information about licensing the Data Management Add-on, see the Tableau documentation.

# Tableau roles and permissions

The lineage harvester uses the Tableau Rest APIs and Tableau Metadata API to ingest the Tableau metadata. You need at least minimum permissions in Tableau to enable the lineage harvester to access the Tableau metadata and ingest it in Data Catalog.

## Permissions on metadata

Permissions control who is allowed to see and manage external assets and which metadata (for both Tableau content and external assets) is shown through lineage.

> Note  If Tableau Online or Tableau Server is not licensed with the Data Management Add-on, then by default, only admins can see database and table metadata through the Tableau Metadata API. You can turn on "derived permissions", to allow users to see metadata on external assets for the content that they own, or for the content that is published to a project for which they are a project leader or project owner. For complete information, see the Tableau documentation.

## Minimum roles and permissions in Tableau

You need to following minimum roles and permissions to harvest Tableau metadata:

- You have a View permission on Tableau projects, workbooks and data sources you want to ingest.
- You have a Viewer or Explorer (can publish) role with access to the Tableau REST API.

## Recommended roles and permissions in Tableau

For a full ingestion, we recommend the following roles and permissions in Tableau:

- You have at least a View permission on Tableau projects, workbooks and data sources you want to ingest.
- You have the Explorer role with the Data Management Add-on.

If you use the Explorer role and you have access to a subproject, but not the parent project, the parent project is ingested with the Tableau UUID, to maintain the hierarchy of assets.

> Tip Tableau users with a Server Administrator role have access to the entire Tableau Server. Tableau users with a Site Administrator role can only be assigned to specific Tableau sites. As a result, if you have the Site Administrator role, only metadata from specific Tableau sites can be ingested in Data Catalog.

# Tableau data sources

You can create data sources in Tableau when you connect to data. After you set up the data sources in Tableau, you can publish data sources as standalone resources, or you can publish workbooks with the data sources embedded in.

Unless you take actions to publish the data source separately, the data source is published as embedded in a workbook by default. For more information, see Publishing data separately or embedded in workbooks.

In Collibra, Collibra Data Lineage ingests metadata of data sources as assets of the Tableau Data Model asset type, regardless of the way the data sources are published.

## eTDM and pTDM

When you ingest a Tableau data source in Collibra, each asset is identified as eTDM or pTDM with [eTDM] or [pTDM] added to the asset name.

eTDM stands for embedded Tableau Data Model, which indicates that the asset represents the data source that is embedded in a workbook in Tableau. pTDM stands for published Tableau Data Model, which indicates that the asset represents the data source that is published separately in Tableau.

For a data source that is both published separately and embedded in a workbook, Collibra Data Lineage ingests the metadata in one of the following ways:

- If the metadata of the embedded data source matches that of the published data source, Collibra Data Lineage ingests the metadata only from the published data source to avoid duplication.

- If the metadata of the embedded data source contains more fields than that of the published data source, Collibra Data Lineage ingests metadata from both the published and embedded data sources.

As a result, a Tableau workbook can have one of the following relations:

- To the published and embedded data source.
- To the published data source only.

# Tableau ingestion results

The following tables shows the ingestion results based on Tableau permissions.

By default, the lineage harvester uses both the Tableau REST API and the Tableau Metadata API, but you can limit the ingestion by allowing the lineage harvester to use only the Tableau REST API.

> Note If you ingest a Tableau dataset that doesn't have any attributes, asterisks (*) are shown as the Tableau Data Attribute asset names in Collibra.

| Tableau site role | Metadata API in Tableau | Result in Data Catalog |
|---|---|---|
| Viewer | Disabled | Tableau reports and data sources are ingested into Data Catalog, but with a limited scope.<br><br>Resulting asset types:<br><br>• Tableau Dashboard<br>• Tableau Data Model<br>• Tableau Project<br>• Tableau Server<br>• Tableau Site<br>• Tableau Workbook<br>• Tableau Worksheet<br><br>Important We cannot retrieve lineage information or perform automatic stitching. |
| Viewer | Enabled | Tableau reports and data sources are ingested into Data Catalog, but with a limited scope.<br><br>Resulting asset types:<br><br>• Tableau Dashboard<br>• Tableau Data Attribute<br>• Tableau Data Model<br>• Tableau Server<br>• Tableau Site<br>• Tableau Project<br>• Tableau Workbook<br>• Tableau Worksheet<br><br>Important We cannot retrieve lineage information or perform automatic stitching. |

| Tableau site role | Metadata API in Tableau | Result in Data Catalog |
|---|---|---|
| Explorer, without the Data Management Add-on | Disabled | Tableau reports and data sources are ingested into Data Catalog, but with a limited scope.<br><br>Resulting asset types:<br><br>• Tableau Server<br>• Tableau Site<br>• Tableau Project<br>• Tableau Dashboard<br>• Tableau Data Model<br>• Tableau Workbook<br>• Tableau Worksheet<br><br>Important We cannot retrieve lineage information or perform automatic stitching. |
| Explorer, without the Data Management Add-on | Enabled | Tableau reports and data sources are ingested into Data Catalog, but with a limited scope.<br><br>Resulting asset types:<br><br>• Tableau Server<br>• Tableau Site<br>• Tableau Project<br>• Tableau Dashboard<br>• Tableau Data Model<br>• Tableau Data Attribute<br>• Tableau Workbook<br>• Tableau Worksheet<br><br>Important We cannot retrieve lineage information or perform automatic stitching. |

| Tableau site role | Metadata API in Tableau | Result in Data Catalog |
|---|---|---|
| One of the following:<br>• Tableau Server Administrator<br>• Tableau Site Administrator<br>• Explorer, with the Data Management Add-on | Disabled | Data Catalog creates new assets according to your content in Tableau using metadata in Tableau databases and tables.<br><br>Resulting asset types:<br>• Tableau Server<br>• Tableau Site<br>• Tableau Project<br>• Tableau Data Model<br>• Tableau Workbook<br>• Tableau Dashboard<br>• Tableau Worksheet<br><br>Important We cannot retrieve lineage information or perform automatic stitching. |

| Tableau site role | Metadata API in Tableau | Result in Data Catalog |
|---|---|---|
| One of the following:<br>• Tableau Server Administrator<br>• Tableau Site Administrator<br>• Explorer, with the Data Management Add-on | Enabled | Data Catalog creates new assets according to your content in Tableau using metadata in Tableau databases and tables.<br><br>Resulting asset types:<br>• Tableau Server<br>• Tableau Site<br>• Tableau Project<br>• Tableau Data Model<br>• Tableau Data Attribute<br>• Tableau Workbook<br>• Tableau Dashboard<br>• Tableau Worksheet<br><br>Note If Tableau Online or Tableau Server is not licensed with the Data Management Add-on, then by default, only admins can see database and table metadata through the Tableau Metadata API. You can turn on "derived permissions", to allow users to see metadata on external assets for the content that they own, or for the content that is published to a project for which they are a project leader or project owner. For complete information, see the Tableau documentation. |

# Prepare a domain for Tableau ingestion

During Tableau integration, Tableau assets are ingested in one or more specified domains in Collibra Data Intelligence Cloud. You then include the domain reference ID (or IDs) in the appropriate configuration file.

# Prerequisites

- You have a resource role with the Domain > Add resource permission.

# Steps

1. In the main menu, click the **Create** (+) button.
   » The **Create** dialog box appears.
2. Click the **Organization** tab.
3. Click a domain type from the list.
   If you clicked the wrong domain type here, you can change it in the **Type** field in the next screen.
   » The **Create Domain** dialog box appears.
4. Enter the required information.

| Field | Description |
|---|---|
| Type | The domain type of the domain you are creating. In this case, you need to select *BI Catalog*. |
| Community | The community under which the domain will be located. |
| Name | The name of the new domain or domains.<br><br>Tip<br>You can create multiple domains in one go.<br>To do this, press `Enter` after typing a value and then type the next. Domain names have to be unique in their parent community. If you type a name that already exists, it will appear in strike-through style. |

5. Click **Create**.
6. Open your domain. If you created multiple domains, open each of them in turn.

7. Copy the reference ID of each domain you created.

   Tip If you go to your domain, you can find the domain ID in the URL. The URL looks like: https://<yourcollibrainstance>/domain/**22258f64-40b6-4b16-9c08-c95f8ec0da26**?view=00000000-0000-0000-0000-000000040001. In this example, the domain ID is in bold.

8. Paste the domain reference ID (or IDs) in the appropriate configuration file, depending on whether you want to ingest Tableau assets in a single domain or multiple domains.
   For complete information on which properties and which configuration files to use, see the `domainId` property description in Prepare the lineage harvester configuration file for Tableau.

Warning When you run the lineage harvester, Collibra Data Lineage creates all Tableau assets in a single BI Catalog domain. We highly recommend that you do not move these assets to another domain. If you move assets to another domain, they will be deleted and recreated in the initial BI Catalog domain when you synchronize Tableau. As a consequence, all manually added data of those assets is lost.

Warning If you are using Collibra 2021.11 or older, you have to add all Tableau attributes in the operating model to a scope and create a scoped assignment before you ingest Tableau via the lineage harvester. For complete information and step-by-step instruction, see Tableau general troubleshooting.

# Prepare the Data Catalog physical data layer for Tableau stitching

Before you can stitch data objects to the assets in Collibra Data Intelligence Cloud, if you register a data source by using a Jobserver, you must prepare the Data Catalog physical data layer to create assets and the database>schema > table > column hierarchy.

If you register a data source by using Edge, Collibra Data Intelligence Cloud creates the database > schema > table > column hierarchy. If you set the `useCollibraSystemName`

property as `false` in the lineage harvester configuration file, there is no need to complete this task. If you set the `useCollibraSystemName` property as true, create a system asset.

For more information, see Automatic stitching and Prepare the lineage harvester configuration file for Tableau.

> Important In the global assignment of each asset type included in the Tableau operating model, ensure that none of the characteristics that are in the operating model have a maximum cardinality of “0”. If the maximum cardinality is set to “0” for any such characteristics, ingestion will fail.

# Prerequisites

- You have a global role with the Catalog global permission, for example Catalog Author.
- You have a role with the following resource permissions on the **Schema** community:
  - Asset: add
  - Attribute: add
  - Domain: add
  - Attachment: add

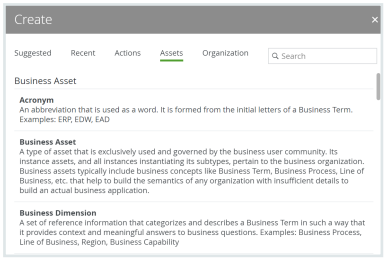
# Steps

1. Register a database as data source.
   » After registration, the assets of the following asset types are created in Data Catalog:
   - Schema
   - Table
   - Column
2. Create a Database asset.

   > Tip We strongly recommend to use the name as your original data source, so that the name of the Database asset matches Tableau's naming convention.

   1. Open Catalog.
   2. In the main menu, click the **Create** (+) button.

3. Click the **Assets** tab.



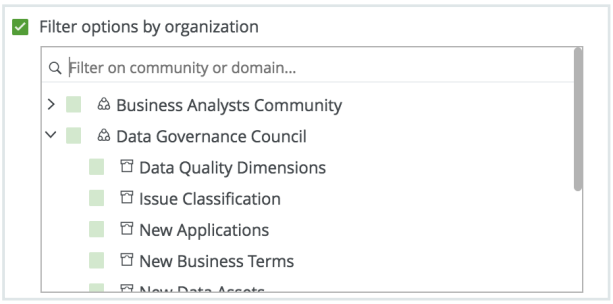

4. Click Database.
   » The **Create Asset** dialog box appears.
5. Enter the required information.

| Field | Description |
|---|---|
| Type | The asset type of the asset that you are creating, in this case Database. |
| Domain | The domain to which the new asset will belong. You can only create a asset type in any domain of a domain type that is assigned to a Database asset type. |
| Name | The name of the Database asset. This has to match the name of the Tableau Data Model.<br><br>Tip<br>You can create multiple assets in one go.<br>To do this, press `Enter` after typing a value and then type the next. Depending on the settings, asset names may have to be unique in their domain. If you type a name that already exists, it will appear in strike-through style. |

6. Click **Create**.
   » A message at the top-right of your screen confirms that one or more assets are created.

3. Create a relation between the Database asset and the Schema asset using the Technology Asset has / belongs to Schema relation type.
   a. In the tab pane, click **Add Characteristic**.
      » The **Add a characteristic** dialog box appears.

b. Click **Relations**.
c. Search for and click **has schema**.
   » The **Add has schema** dialog box appears.
d. Enter the required information.

| Option | Description |
|---|---|
| Assets | The name of the schema. |
| Filter suggested assets by organization | Option to filter the suggestions based on selected communities and domains.<br>If this option is selected, the organization tree appears. You can then filter and select domains and communities.<br>  |
| Start date | Optionally enter the date on which the relation between the assets becomes applicable. Leave this field empty to create a permanent relation. |
| End date | Optionally enter the date on which the relation between the assets is no longer applicable. Leave this field empty to create a permanent relation. |

e. Click **Save**.

4. Check that the following relations are created for all Column assets that you want to stitch to Tableau assets:
   - Schema contains / is part of Table
   - Column is part of / contains Table

# What's next?

Set up the lineage harvester for Tableau ingestion.

# Set up the lineage harvester for Tableau ingestion

The lineage harvester is a software application that is required to collect your Tableau metadata and send it to the Collibra Data Lineage service, where the metadata is processed and new Tableau assets and relations are created.

> Note To ingest Tableau metadata into Data Catalog, you need lineage harvester 2022.02 or newer. We strongly recommend that you use the latest version of the lineage harvester.

# Lineage harvester system requirements

You need to meet the system requirements to be able to install and run the lineage harvester.

## Software requirements

Java Runtime Environment version 11 or newer, or OpenJDK 11 or newer.

For Java Runtime Environment 16 or newer, or OpenJDK 16 or newer, set the `JAVA_OPTS` environment variable for the lineage harvester to function properly:

```
JAVA_OPTS='--illegal-access=deny'
```

> Note To ingest Snowflake data sources, the minimum requirement is Java Runtime Environment version 16 or newer, or OpenJDK 16 or newer.

# Hardware requirements

You need to meet the hardware requirements to install and run the lineage harvester.

## Minimum hardware requirements

You need the following minimum hardware requirements:

- 2 GB RAM
- 1 GB free disk space

## Recommended hardware requirements

The minimum requirements are most likely insufficient for production environments. We recommend the following hardware requirements:

- 4 GB RAM

  > Tip 4 GB RAM is sufficient in most cases, but more memory could be needed for larger harvesting tasks. For instructions on how to increase the maximum heap size, see Technical lineage general troubleshooting.

- 20 GB free disk space

# Network requirements

The lineage harvester uses the HTTPS protocol by default and uses port 443.

You need the following minimum network requirements:

- Firewall rules so that the lineage harvester can connect to:
  - The host names of all data sources in the lineage harvester configuration file.
  - All Collibra Data Lineage service instances in your geographic location:
    - 15.222.200.199 (techlin-aws-ca.collibra.com)
    - 18.198.89.106 (techlin-aws-eu.collibra.com)
    - 54.242.194.190 (techlin-aws-us.collibra.com)
    - 51.105.241.132 (techlin-azure-eu.collibra.com)
    - 20.102.44.39 (techlin-azure-us.collibra.com)

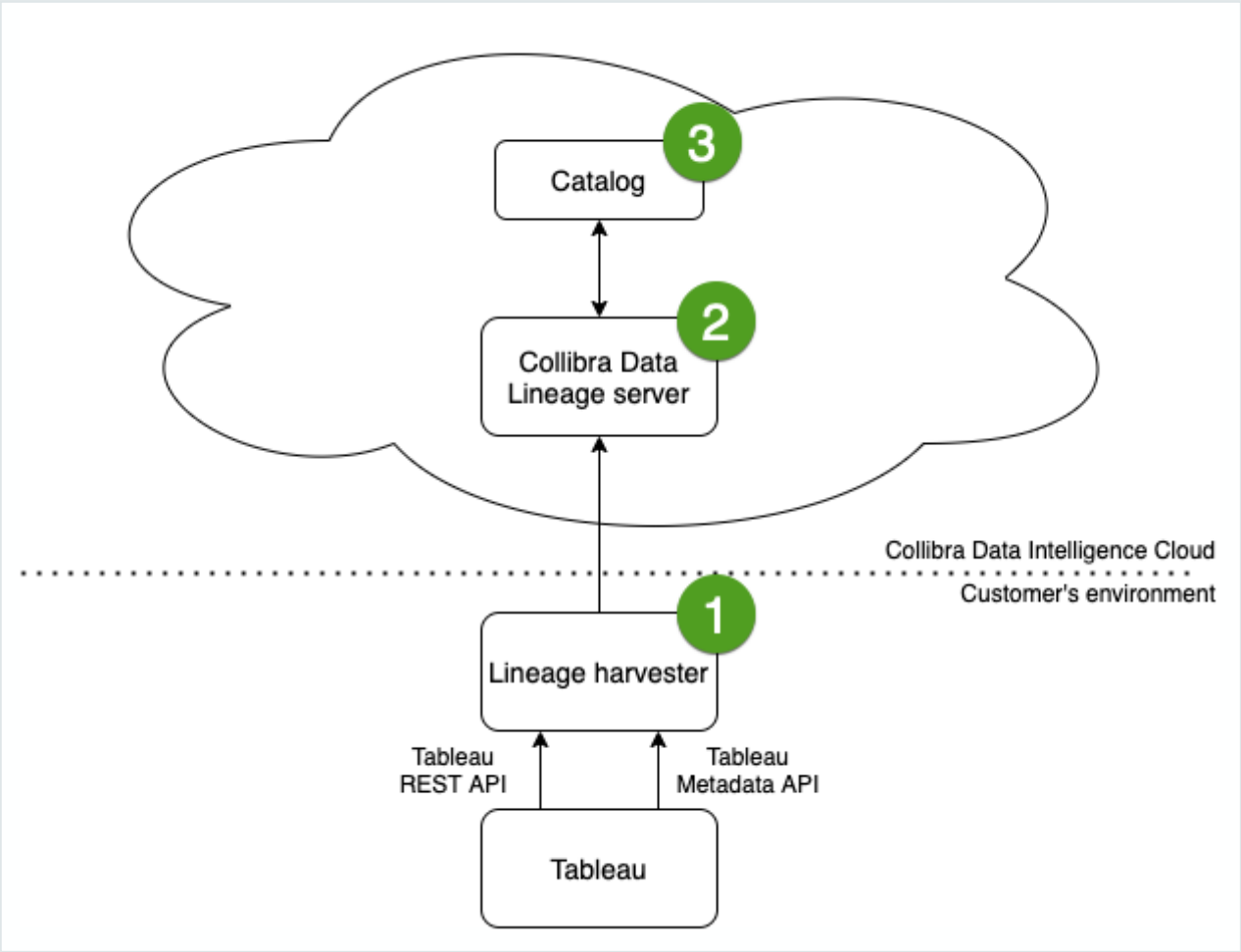
- 35.197.182.41 (techlin-gcp-au.collibra.com)
- 34.152.20.240 (techlin-gcp-ca.collibra.com)
- 35.205.146.124 (techlin-gcp-eu.collibra.com)
- 34.87.122.60 (techlin-gcp-sg.collibra.com)
- 35.234.130.150 (techlin-gcp-uk.collibra.com)
- 34.73.33.120 (techlin-gcp-us.collibra.com)

> Note The lineage harvester connects to different Collibra Data Lineage service instances based on your geographic location and cloud provider. If your location or cloud provider changes, the lineage harvester rescans all your data sources. You have to whitelist all Collibra Data Lineage service instances in your geographic location. In addition, we highly recommend that you always whitelist the techlin-aws-us instance as a backup, in case the lineage harvester cannot connect to other Collibra Data Lineage service instances.

# Tableau ingestion workflow

You run the lineage harvester to start the Tableau ingestion workflow. When you initiate Tableau ingestion, each workflow component performs the following actions:

1. The lineage harvester:
   - Communicates with Tableau.
   - Harvests the Tableau metadata that will be ingested to Data Catalog.
   - Sends the Tableau metadata to the Collibra Data Lineage service.
2. The Collibra Data Lineage service:
   - Analyzes the Tableau metadata.
   - Creates new assets and relations.
   - Stitches existing assets in Data Catalog to Tableau assets.
   - Imports new Tableau assets and their relations in Data Catalog.
3. Data Catalog:
   - Shows new Tableau assets
   - Shows a Technical lineage for Tableau assets.
   - Shows stitching results between Tableau Data Attribute assets and Column assets.




Note This is the recommended workflow. If you do not want to use the Tableau Metadata API, you can disable it via the configuration file.

# Install the lineage harvester for Tableau integration

Before you can use the lineage harvester, you need to download it and install it. You can download the lineage harvester from the Collibra Community downloads page.

Tip
- Install the lineage harvester close to your data source or on the same server.
- The lineage harvester uses port 443.

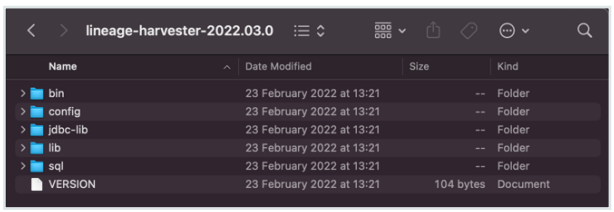
# Prerequisites

- You have purchased Collibra Data Lineage.
- You meet the minimum system requirements.
- You have added Firewall rules so that the lineage harvester can connect to:
  - The host names of all databases in the lineage harvester configuration file.
  - All Collibra Data Lineage service instances within your geographical location:
    - 15.222.200.199 (techlin-aws-ca.collibra.com)
    - 18.198.89.106 (techlin-aws-eu.collibra.com)
    - 54.242.194.190 (techlin-aws-us.collibra.com)
    - 51.105.241.132 (techlin-azure-eu.collibra.com)
    - 20.102.44.39 (techlin-azure-us.collibra.com)
    - 35.197.182.41 (techlin-gcp-au.collibra.com)
    - 34.152.20.240 (techlin-gcp-ca.collibra.com)
    - 35.205.146.124 (techlin-gcp-eu.collibra.com)
    - 34.87.122.60 (techlin-gcp-sg.collibra.com)
    - 35.234.130.150 (techlin-gcp-uk.collibra.com)
    - 34.73.33.120 (techlin-gcp-us.collibra.com)

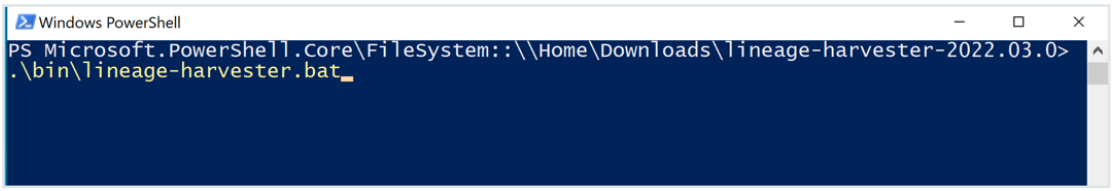
> Note The lineage harvester connects to different instances based on your geographic location and cloud provider. If your location or cloud provider changes, the lineage harvester rescans all your data sources. You have to whitelist all Collibra Data Lineage service instances in your geographic location. In addition, we highly recommend that you always whitelist the techlin-aws-us instance as a backup, in case the lineage harvester cannot connect to other Collibra Data Lineage service instances.

# Steps

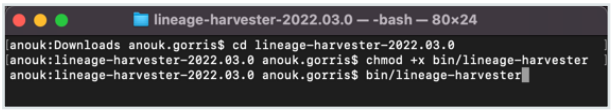
1. Download the newest lineage harvester.
2. Unzip the archive.
   » You can now access the lineage harvester folder.



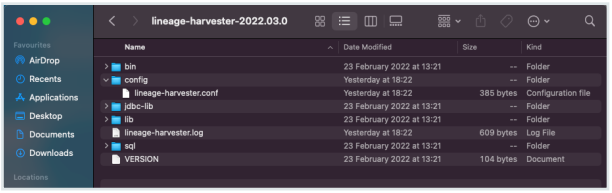
3. Run the following command line to start the lineage harvester:
   - Windows: `.\bin\lineage-harvester.bat`




   - For other operating systems: `chmod +x bin/lineage-harvester` and then `bin/lineage-harvester`




   » An empty configuration file is created in the config folder.




   » The lineage harvester is installed automatically. You can check the installation by running `./bin/lineage-harvester --help`.

## What's next?

You can now prepare the lineage harvester configuration file.

# Prepare the lineage harvester configuration file for Tableau

You have to prepare a configuration file before you run the lineage harvester. The lineage harvester collects your Tableau metadata and sends it to the Collibra Data Lineage service, where it is processed and analyzed. Collibra Data Intelligence Cloud then imports the Tableau assets and relations to Data Catalog.

## Prerequisites

Ensure that you have completed the following tasks:

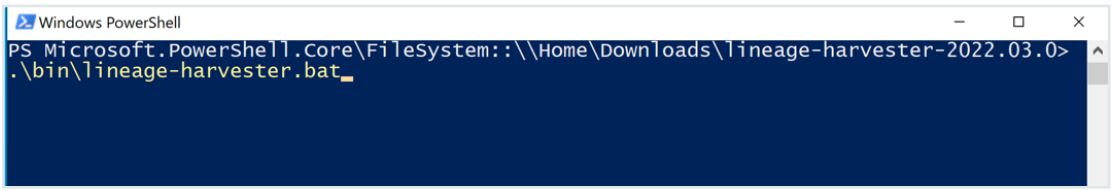
- Installed and set up the latest lineage harvester.
- Tested your connectivity with the Tableau server.
- Prepared the Data Catalog physical data layer for technical lineage.
- Created one or more BI Catalog domains in which you want to ingest the Tableau assets.

Ensure that you meet the following requirements and have the following permissions:

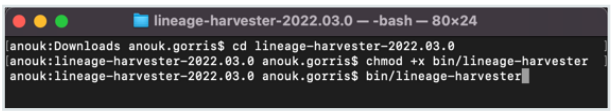
- Use Collibra Data Intelligence Cloud.
- A global role with the following global permissions:
  - Catalog, for example Catalog Author
  - Data Stewardship Manager
  - Manage all resources
  - System administration
  - Technical lineage
- A resource role with the following resource permission on the community level in which you created the BI Catalog domain:
  - Asset: add
  - Attribute: add
  - Domain: add
  - Attachment: add

## Steps

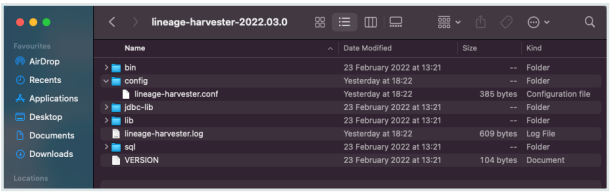
1. Run the following command line to start the lineage harvester:
   - Windows: `.\bin\lineage-harvester.bat`

   - For other operating systems: `chmod +x bin/lineage-harvester` and then `bin/lineage-harvester`

» An empty configuration file is created in the config folder.




2. Open the lineage-harvester.conf file and enter the values for each property.
   **Watch a video on how to do this**

| Properties | Description |
|---|---|
| general | This section describes the connection information between the lineage harvester and Data Catalog. |

| Properties | Description |
| --- | --- |
| catalog | This section contains information that is necessary to connect to Data Catalog. |
| url | The URL of your Collibra Data Intelligence Cloud environment.<br><br>Note You can only enter the public URL of your Collibra DGC environment. Other URLs will not be accepted. |
| username | The username that you use to sign in to Collibra. |

| Properties | Description |
| --- | --- |
| useCollibraSystemName | Indication whether you want to use the system or server name of a data source to match to the System asset you created when you prepared the physical data layer. This is useful when you have multiple databases with the same name.<br><br>By default, the `useCollibraSystemName` property is set to `false`. If you want to use it, set it to `true`.<br><br>Important<br>◦ If you set this property to `true`:<br>▪ You must provide a Tableau <source ID> configuration file that defines the system name of databases in Tableau.<br>▪ The lineage harvester reads the value of the `collibraSystemName` property in the <source-ID> configuration file.<br>◦ If you set the `useCollibraSystemName` property to `false`, the lineage harvester ignores the `collibraSystemName` property in the <source-ID> configuration file.<br><br>Note If you set the `useCollibraSystemName` property to `true`, but you don't define the system name in the Tableau <source ID> configuration file, the system name in the technical lineage is DEFAULT. |

| Properties | Description |
| --- | --- |
| useSharedDbModel | Optional property to enable the sharing of metadata batches from multiple SQL data sources. Set this property to `true`, to help avoid potential analysis errors on the Collibra Data Lineage service.<br><br>To use this property, you need lineage harvester 2022.07 or newer.<br><br>If you set this property to `true`, you have to run the lineage harvester twice. Read the following details about the issue and solution.<br><br>**See details about the issue and solution**<br>Normally, when you run the lineage harvester to harvest metadata from two or more data sources, the metadata from each source is processed independently. This means that the metadata from one data source cannot access the metadata of another.<br><br>Let's say, for example, you specify the following two SQL data sources in your lineage harvester configuration file:<br>◦ A database source that retrieves the database model.<br>◦ An SqlDirectory source with Data Manipulation Language (DML) statements that reference data in the database source.<br><br>Because these data sources are processed independently, there is a good chance that the DML statements will fail during analysis. Any wildcards in the DML statements, for example, would fail because the SqlDirectory source can't access the referenced database source. |

| Properties | Description |
| --- | --- |
| | **The solution**<br><br>The shared database model allows for computed results from a “main” batch. Although multiple data sources are still processed independently, the metadata from each data source is merged into a main batch. Then, before analyzing the next batch, a check is done to see if a preceding main batch exists. If one does, the analyzer retrieves the database model and the DML statements successfully pass analysis.<br><br>This means, however, that you have to run the lineage harvester twice. On the first run, the harvested metadata is merged in a main batch. Then, when you run the lineage harvester again, using the full-sync command, the subsequent batches are able to successfully reference the metadata in the main batch.<br><br>In a future version of Collibra, this property will be enabled by default and you won't need to run the lineage harvester twice. |
| sources | This section contains all of the Tableau connection properties. |
| type | The kind of data source. In this case, the value has to be *Tableau*. |

| Properties | Description |
| --- | --- |
| id | The unique ID to identify the Tableau metadata that was uploaded to the Collibra Data Lineage.<br><br>Warning In the `sources` section of your lineage harvester configuration file, you can only specify one `id` property per Tableau server or Tableau online account. If you have multiple `id` properties for a single Tableau server or Tableau online account, ingestion will fail. If you have multiple `id` properties in the configuration file, it means you intend to ingest from multiple unique Tableau servers or Tableau online accounts.<br><br>Tip This value can be anything as long as it is a unique. The lineage harvester uses the ID to identify a batch of data on the Collibra Data Lineage service. |
| url | The link to the data in Tableau. |

| Properties | Description |
| --- | --- |
| username | The username you use to sign in to the Tableau server.<br><br>Warning As of October 2022, Tableau is enforcing multi-factor authentication for Tableau Cloud Admin users. However, the lineage harvester doesn't support multi-factor authentication. Therefore, Tableau Cloud users with an Admin role must use token-based authentication. This does not affect Tableau Server users or Tableau Cloud users with an Explorer role.<br><br>Important If you want to use token-based authentication, you need to replace `username` with `tokenName`. You must specify either `username` or `tokenName`; if both exist, then `tokenName` is used. |
| tokenName | The lineage harvester authentication token.<br><br>Note For token-based authentication, use this property in your lineage harvester configuration file, instead of the `username` property. If both properties are present, `tokenName` is used. |

| Properties | Description |
| --- | --- |
| siteIds | The site IDs of the Tableau sites that you want to include in the ingestion process.<br><br>Important Ensure that you specify the correct value. The correct value is the URL of the site to which you want to sign in. When you manually sign in to Tableau Server or Tableau Online, the site ID is the value that appears after **/site/** in the browser address bar. In the following example URLs, the site ID is `MarketingTeam`:<br>◦ Tableau Server: http://MyServer/#/site/MarketingTeam/projects<br>◦ Tableau Online: https://10ay.online.tableau.com/#/site/MarketingTeam/workbooks<br><br>On Tableau Server, however, the URL of the Default site does not specify the site. For example, the URL for a view named Profits, on a site named Sales, is http://localhost/#/site/sales/views/profits. The URL for this same view on the Default site is http://localhost/#/views/profits. The site name Sales does not figure in the URL. If you can't see the site ID, leave this property empty: `"siteIds": [""]`<br><br>Example If you want to ingest two Tableau sites "Site 1" and "Site 2", you can enter the following information in the siteIds property: ["site ID of Site 1", "site ID of Site 2"]. |

| Properties | Description |
| --- | --- |
| siteNames | The site names of the corresponding site IDs.<br><br>Important This property is:<br>◦ Optional for Tableau Server<br>◦ Mandatory for Tableau Online.<br><br>Warning If you have Tableau Server and you don't use this property, you must delete it from your configuration file. Don't leave the property in the configuration file without a value. |
| restOnly | Indication whether or not you would like to use both the Tableau REST API and Tableau Metadata API to harvest Tableau metadata.<br><br>◦ `false` (default): The lineage harvester will use the REST API and Metadata API to harvest Tableau metadata.<br>◦ `true`: The lineage harvester will only use the REST API to harvest Tableau metadata.<br><br>Note This property must be set to `false`, to:<br>◦ Enable technical lineage and the automatic stitching of Column assets to Tableau Data Attribute assets.<br>◦ Harvest owner information for Tableau projects, workbooks and data models. |

| Properties | Description |
| --- | --- |
| collibraSystemName | Regardless of the value set for the `useCollibraSystemName` property, the following is true:<br>◦ You must include this property in your configuration file.<br>◦ You can leave this property empty.<br>◦ Any value that you give is ignored.<br><br>If you want to specify name of the system or server, because, for example, you have multiple databases with the same name, then:<br><br>a. Set the `useCollibraSystemName` property to `true`.<br>b. Specify the system or server names in the `collibraSystemName` property in your Tableau <source ID> configuration file.<br><br>> Note This is a legacy property that will be deprecated in a future release. |
| domainId | The unique reference ID of the domain in Collibra Data Intelligence Cloud in which you want to ingest the Tableau assets.<br><br>**How do I find a domain reference ID?**<br>Open the relevant domain in Collibra. The URL looks like: https://<yourcollibrainstance>/domain/**22258f64-40b6-4b16-9c08-c95f8ec0da26**?view=00000000-0000-0000-0000-000000040001. In this example, the reference ID is in bold. |

| Properties | Description |
| --- | --- |
| excludeImages | Optional property for excluding the downloading of images.<br><br>To exclude the downloading of images, set this property to `true`.<br><br>To indicate the projects that you want to ingest in different domains, specify the `filters` section in your Tableau <source ID> configuration file.<br><br>> Note The maximum number of images that can be uploaded to Collibra per day is determined by the configuration of the file upload service, in Collibra Console. For complete details, see the Upload configuration settings in DGC service configuration: options. |
| concurrencyLevel | This optional property is intended to help if you are experiencing HTTP 401 Unauthorized errors due to too many concurrent HTTP calls, using the same token. It allows you to specify the internal sizing, meaning the amount of tasks that can be executed at the same time.<br><br>The default value is "10", meaning as many as 10 HTTP requests can take place in parallel. Consider reducing the value if you are experiencing HTTP 401 Unauthorized errors. Setting the value to "1" effectively disables the concurrency level, so that HTTP requests will be run in a synchronous manner, instead of in parallel. |

| Properties | Description |
| --- | --- |
| deleteRawMetadataAfterProcessing | The lineage harvester harvests metadata from specified data sources and uploads it in a ZIP file to a Collibra Data Lineage service instance, for processing.<br><br>You can use this optional property to specify whether or not the metadata should be deleted after it has been processed.<br><br>If the property is set to `true`, the metadata is deleted after processing. If set to `false`, the metadata is stored in an Amazon S3 bucket. |

<table>
<tr><th>Properties</th><th>Description</th></tr>
<tr><td>paging</td><td>Optional property for customizing the Tableau API pagination settings.<br>The default values are sufficient in most cases; however, you can decrease them to help mitigate node limit errors, or increase them to speed up API calls.<br><br>The complete list of pagination settings, descriptions and default values<br><br><pre>"paging": {
   "databasesPageSize": 100,
   "tablesPageSize": 100,
   "tablesColumnsPageSize": 100,
   "tableColumnsPageSize": 1000,
   "datasourcesPageSize": 50,
   "datasourcesFieldsPageSize": 50,
   "datasourceFieldsPageSize": 100,
   "worksheetsPageSize": 100,
   "worksheetsFieldsPageSize": 100,
   "worksheetFieldsPageSize": 1000,
   "usersPageSize": 100,
   "dashboardsPageSize": 100,
   "columnsLimit": 20,
   "fieldsLimit": 20
   }</pre><br>Settings per metadata type and descriptions<br><br><table>
<tr><th>Metadata type</th><th>Setting and description</th></tr>
<tr><td>Dashboard</td><td>◦ <code>dashboardsPageSize</code>: The number of dashboards per page.</td></tr>
</table></td></tr>
</table>

<table>
<tr><th>Properties</th><th>Description</th></tr>
<tr><td></td><td>
<table>
<tr><th>Metadata type</th><th>Setting and description</th></tr>
<tr><td>Worksheet</td><td>◦ <code>worksheetsPageSize</code>: The number of worksheets per page.<br>◦ <code>worksheetsFieldsPageSize</code>: The number of worksheet fields per page.</td></tr>
<tr><td>Database</td><td>◦ <code>databasesPageSize</code>: The number of databases per page.</td></tr>
<tr><td>Table</td><td>◦ <code>tablesPageSize</code>: The number of tables per page.<br>◦ <code>tablesColumnsPageSize</code>: The number of table columns per page.</td></tr>
<tr><td>Table columns</td><td>◦ <code>tableColumnsPageSize</code>: The number of table columns per page.</td></tr>
<tr><td>Users</td><td>◦ <code>usersPageSize</code>: The number of users per page.</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><th>Properties</th><th>Description</th></tr>
<tr><td></td><td>
<table>
<tr><th>Metadata type</th><th>Setting and description</th></tr>
<tr><td>Data source</td><td>
<ul>
<li><code>datasourcesPageSize</code>: The number of data sources per page.</li>
<li><code>datasourcesFieldsPageSize</code>: The number of data source fields per page.</li>
<li><code>columnsLimit</code>: The number of data source field columns per page.</li>
<li><code>fieldsLimit</code> : The number of referenced data source fields per page.</li>
</ul>
</td></tr>
<tr><td>Data source field</td><td>
<ul>
<li><code>datasourceFieldsPageSize</code>: The number of data source fields per page.</li>
<li><code>columnsLimit</code>: The number of data source field columns per page.</li>
<li><code>fieldsLimit</code> : The number of referenced data source fields per page.</li>
</ul>
</td></tr>
</table>
</td></tr>
</table>

3. Save the configuration file.
4. Start the lineage harvester again in the console and run the following command:
   - for Windows: `.\bin\lineage-harvester.bat full-sync`
   - for other operating systems: `./bin/lineage-harvester full-sync`
5. When prompted, enter the password or client secret to connect to your Collibra Data Intelligence Cloud and Tableau environment.
   » The passwords are encrypted and stored in **/config/pwd.conf**.

## Example

```
{
 "general": {
   "catalog": {
     "url": "https://<organization>.collibra.com",
     "username": "<your-collibra-username>"
   },
   "useCollibraSystemName": false,
   "useSharedDbModel": true
 },
 "sources": [
  {
   "type": "Tableau",
   "id": "unique-ID",
   "url": "URL to Tableau server",
   "username": "Admin",
   "siteIds": ["site ID of Tableau Site 1", "site ID of Tableau Site
2"],
   "siteNames": ["site name of Tableau Site 1", "site name of
Tableau Site 2"],
   "restOnly": false,
   "collibraSystemName": "tableau-system-name",
   "domainId": "Domain-resource-ID",
   "excludeImages": true,
   "concurrencyLevel": 1,
   "deleteRawMetadataAfterProcessing": true,
   "paging": {
     "pagination-setting": 100,
     "pagination-setting-2": 100
   }
  }
 ]
}
```

## What's next?

The lineage harvester triggers Collibra to import Tableau assets and their relations and create a technical lineage for Tableau Data Attribute assets.

If issues occur during the Tableau ingestion process, check the Tableau troubleshooting section to solve your problems.

To refresh the Tableau metadata, you can run the lineage harvester again or schedule jobs to run them automatically.

> Tip You can check the progress of the Tableau ingestion in Activities. The results field indicates how many relations were imported into Data Catalog.

# Prepare the Tableau <source ID> configuration file

The lineage harvester uses the configuration file to connect to Tableau. However, you may need to provide additional information via a Tableau <source ID> configuration file. You use the Tableau <source ID> configuration file to:

- Define your Tableau operating model.
- Provide additional information about databases and files in Tableau. For example, you can define the system name of databases in Tableau.
- Map a Tableau technical database name to the real database name, to preserve stitching. See the databaseMapping property.
- Define in which domains in Collibra you want to ingest assets from your Tableau sites and Tableau projects. See the filters property.

## Steps

**Watch a video on how to do this**

1. Create a new JSON file in the lineage harvester **config** folder.
2. Give the JSON file the same name as the value of the `Id` property in the lineage harvester configuration file.

> Example If the value of the `Id` property in the lineage harvester configuration file is `tableau-source-1`, then the name of your JSON file should be *tableau-source-1.conf*.

> Important Your JSON file must have the file extension *.conf*.

3. For each database in Tableau, add the following content to the JSON file:

> Tip You can use wildcards to capture multiple string combinations for any of these properties.
> **Show me the supported wildcards**

| Pattern | Description |
|---|---|
| * | Matches everything. |
| ? | Matches any single character. |
| [seq] | Matches any character in "seq". |
| [!seq] | Matches any character not in "seq". |

| Property | Description |
|---|---|
| collibraSystemNames | This section contains the system information for different Tableau data sources. Depending on the kind of data source or connection, you have to specify how to connect to this data source.<br><br>Tip For more information, see the Tableau documentation. We also recommend to check the list of supported connectors in Tableau. |
| databases | This section contains connection information to one or more databases in Tableau.<br><br>Tip<br>◦ If you do not have databases in Tableau, you can remove this section.<br>◦ The values that you specify for this property are not case sensitive. |

| Property | Description |
| --- | --- |
| hostname | The host name of the database. |
| collibraSystemName | The system name of the database. |
| files | This section contains connection information to one or more files in Tableau.<br><br>Tip If you do not have files in Tableau, you can remove this section. |
| filePath | The full path to the file. For example, the path to a JSON file. |
| collibraSystemName | The system name of the file. |
| connectors | This section contains connection information to one or more connectors in Tableau.<br><br>Tip<br>◦ If you do not have connectors in Tableau, you can remove this section.<br>◦ The values that you specify for this property are not case sensitive. |
| connectorUrl | The URL of the connector. For example, the URL to Google Analytics. |
| collibraSystemName | The system name of the connector. |

| Property | Description |
| --- | --- |
| cloudFiles | This section contains connection information to one or more cloud files in Tableau's input data.<br><br>Tip If you do not have cloud files in Tableau, you can remove this section. |
| name | The name of the file. For example, the name of a Zendesk file. |
| collibraSystemName | The system name of the cloud file. |
| databaseMapping | The Tableau API returns a technical database name based on the hostname, instead of the actual database name, which breaks stitching. The values that you specify for this property are not case sensitive.<br><br>This property allows you to map a Tableau technical database name to the real database name, for example:<br><br>`"databaseMapping": { "<hostname:port>":"<actual database name>" }`<br><br>Including the port, as shown in the example, is optional. |

| Property | Description |
| --- | --- |
| filters (Beta)<br><br>This section is a beta feature. | This section defines the following ingestion rules:<br><br>◦ The Tableau projects and sub-projects from which you want to ingest metadata.<br>◦ The domains in Collibra into which you want to ingest.<br>Filtering is transitive, which means that all resources in a specified project, such as Tableau workbooks and all sub-projects, are ingested.<br><br>Tableau assets that are not mapped to the specified domains, for example the Tableau Server assets and the parent projects (if you specify their sub-projects), are ingested in the default domain. For more information about the default domain, see the `domainId` property in the lineage harvester configuration file.<br><br>Note<br>◦ If you want to ingest all assets in a Tableau site, use the `domainMapping` section.<br>◦ The `domainMapping` and `filters` sections are mutually exclusive. Do not include both `domainMapping` and `filters` sections in your JSON file. |
| projects | Specifies the Tableau projects to be ingested and the domain in which you want to ingest metadata from the Tableau projects or sub-projects. |

| Property | Description |
| --- | --- |
| site_name > project_ name : domain_id | The `site_name` should be the Tableau site name. The `project_name` should be the Tableau project name.<br><br>The `domain_id` should be the unique reference ID of the domain in Collibra in which you want to ingest metadata.<br><br>When you specify the site and project names, the following rules apply:<br><br>◦ Add spaces before and after >. The spaces are separators between the site and project.<br>◦ Specify the full exact site and project names. Do not use wildcards.<br><br>When you specify a Tableau project, all assets in the project are ingested in the specified domain. If you want to ingest assets from different Tableau projects in one domain, you can specify the same value for `domain id` for different projects.<br><br>**Example**<br><br>`"Collibra_tab_partner_site > JB_Test_ 2812": "d224a1a5-43b4-43b2-8df0- ddf8f2726b82"` |

| Property | Description |
| --- | --- |
| site_name > project_name > sub-project_name : domain_id | The `site_name` should be the Tableau site name. The `project_name` should be the Tableau project name. Optionally, use `sub-project_name` to specify the Tableau sub-project name.<br><br>The `domain_id` property should be the unique reference ID of the domain in Collibra in which you want to ingest metadata.<br><br>When you specify the site, project and sub-project names, the following rules apply:<br><br>◦ Add spaces before and after >. The spaces are separators between the site and project.<br>◦ Specify the full exact site and project names. Do not use wildcards.<br>**Example**<br><br>`"Collibra_tab_partner_site > JB_Test_2812 > ProjectJJ2": "d224a1a5-43b4-43b2-8df0-ddf8f2726b82"` |

| Property | Description |
| --- | --- |
| domainMapping | This section defines in which domains in Collibra you want to ingest assets from your Tableau sites and Tableau projects.<br><br>Important<br>◦ Use this property only if you want to ingest Tableau assets into multiple domains in Collibra Data Intelligence Cloud. If you want to ingest into a single domain, use only the `domainID` property in the lineage harvester configuration file.<br>◦ The `domainID` property in the lineage harvester configuration file represents the default domain. Tableau assets that are not mapped to specific domains via this `domainMapping` section, for example Tableau Server assets, are ingested in that default domain.<br><br>Domain mapping is transitive, meaning that all resources, such as Tableau workbooks and data attributes in a parent Tableau site, project or sub-project, are ingested in the same domain as the parent.<br><br>Note<br>◦ If you want to ingest all assets in a Tableau site, use the `domainMapping` section.<br>◦ The `domainMapping` and `filters` sections are mutually exclusive. Do not include both `domainMapping` and `filters` sections in your JSON file. |

| Property | Description |
| --- | --- |
| | **Show me an example**<br>Let's say that you have a Tableau site named "Site-1". You want to ingest all Tableau projects in "Site-1" in a domain named "Domain-1" in Collibra, with the exception of one Tableau project named "Project-Default", which you want to ingest in "Domain-2". You should configure the `domainMapping` section as follows.<br>`"domainMapping": { "<Site-1>": "reference-id-of-Domain-1", "<Site-1> > <Project-Default>": "reference-id-of-Domain-2" }`<br>If you wanted to specify a domain for a sub-project of "Project-Default", you would use the `<site name> > <project name> > <sub-project name>` property, as described below.<br>Tip For the properties in this `domainMapping` section, ensure that you maintain the spaces before and after ">", for example `"Site-1 > Project-Default"`. The spaces serve as a separator between the site and the projects. |

| Property | Description |
| --- | --- |
| site name | The Tableau site name, followed by the unique reference ID of the domain in Collibra in which you want to ingest resources from the Tableau site.<br><br>Important In the configuration file, use the actual site name, along with the domain reference ID, for example: `"Collibra_tab_partner_site": "afc8cfb0-91f1-4075-a3e5-7ce6d1f9bcc9"` |
| site name > project name | The Tableau project name, preceded by the name of the Tableau site to which it belongs, and followed by the unique reference ID of the domain in Collibra in which you want to ingest resources from the Tableau project.<br><br>Important In the configuration file, use the actual site and project names, along with the domain reference ID, for example: `"Collibra_tab_partner_site > JB_Test_2812": "d224a1a5-43b4-43b2-8df0-ddf8f2726b82"` |

| Property | Description |
| --- | --- |
| site name > project name > sub-project name | The Tableau sub-project name, preceded by the name of the Tableau site and project to which it belongs, and followed by the unique reference ID of the domain in Collibra in which you want to ingest resources from the Tableau sub-project.<br><br>Important In the configuration file, use the actual site, project and sub-project names, along with the domain reference ID, for example: `"Collibra_tab_partner_site > JB_Test_2812 > ProjectJJ2": "d224a1a5-43b4-43b2-8df0-ddf8f2726b82"` |

| Property | Description |
| --- | --- |
| . | |

```
{
  "collibraSystemNames": {
    "databases": [
      {
        "hostName": "database-hostname",
        "collibraSystemName": "public"
      }
    ],
    "files": [
      {"filePath": "C:\\ProgramData\\Tableau\\Tableau
Server\\data\\files\\sample.xls",
        "collibraSystemName": "sample-files"
      }
    ],
    "connectors": [
      {
        "connectorUrl": "tableau-server-connector-url.com",
        "collibraSystemName": "Oracle-connector"
      }
    ],
    "cloudFiles": [
      {
        "name": "file-name",
        "collibraSystemName": "FILE"
      }
    ]
  },
  "databaseMapping": {
    "<hostname:port>":"<actual database name>"
  },
  "filters": {
     "projects":{
       "site_name2 > project_name2": "domain-reference-id2",
       "site_name3 > project_name3 > subproject_name":
"domain-reference-id2"
       }
   }
}
```

4. Save the <source ID> configuration file.

# Schedule Tableau ingestion jobs

You can use Task Scheduler on Windows or Crontab on Mac and Linux to make the lineage harvester run scheduled jobs. In a scheduled job, the lineage harvester uploads the Tableau metadata information to Collibra.

Collibra automatically creates new Tableau assets and stitches the Tableau assets to existing data sources in Data Catalog at specific times, dates or intervals, using the information in your configuration file.

> Warning When you run the lineage harvester, Collibra Data Lineage creates all Tableau assets in a single BI Catalog domain. We highly recommend that you do not move these assets to another domain. If you move assets to another domain, they will be deleted and recreated in the initial BI Catalog domain when you synchronize Tableau. As a consequence, all manually added data of those assets is lost.

> Warning Relations that were manually created between Tableau assets and other assets via a relation type in the Tableau operating model, are deleted after a refresh of the Tableau metadata.

# Migrating Tableau assets to the new Tableau operating model

A key feature of the Collibra Data Intelligence Cloud 2022.02 release was the ability to ingest Tableau metadata in Collibra Data Catalog and synchronize the metadata using the lineage harvester. However, this new integration method was only available to customers who did not need to migrate existing Tableau assets to the new operating model. A migration script now eliminates that limitation.

In this section, we provide an overview of:

- How to integrate Tableau metadata via the lineage harvester.
- How to use the lineage harvester to migrate your existing Tableau assets to the new operating model.

# About the Tableau migration

This section describes the terminology and methodology for migrating your existing Tableau assets to the new Tableau operating model.

## Terminology

| Term | Description |
|---|---|
| Tableau integration v1 | The process of integrating and synchronizing Tableau metadata via the Data Catalog UI, including:<br>• The Tableau assets that were created in the process.<br>• Any custom asset types, attribute types and relation types.<br>• Any customizations to the Tableau asset types.<br>• Any customizations to your Tableau assets, for example added attributes and relations.<br>• Any tags that you added to your Tableau assets.<br>• The specific Tableau ingestion results, which differ from the v2 ingestion results. |
| Tableau integration v2 | The process of integrating and synchronizing Tableau metadata via the lineage harvester, including:<br>• The Tableau assets that were created in the process.<br>• The specific Tableau ingestion results, which differ from the v1 ingestion results. |
| Migration script | A specific set of lineage harvester commands used to migrate your custom asset types, attribute types and relation types that were created as part of Tableau integration v1.<br>Note You need lineage harvester version 2022.03.0-5 or newer. We recommend that you use the newest lineage harvester. |

# Methodology

The following is our methodology for migrating Tableau integration v1 metadata to the new operating model. For greater detail see Overview: Tableau integration v2 and migration.

> Note The purpose of this document is to guide you through the migration of assets that were created via step 1 in the table below. That step is included here merely to present the complete context, from ingesting assets via Tableau integration v1, through migration.

| No. | Step | Details |
|---|---|---|
| 1 | Integrate and synchronize Tableau metadata via Tableau integration v1. | Over time, you have likely customized the Tableau asset types, created custom attribute types and relation types, and added attributes and relations to your Tableau v1 assets.<br><br>When you switch to the harvester integration, you want to ensure that you won't lose any of those customizations. All manually created asset types, attribute types and relation types will be migrated. |
| 2 | Integrate the same Tableau metadata, but this time via Tableau integration v2. | After successful integration, you will have:<br><br>• A single BI Catalog domain in Collibra with custom Tableau integration v1 assets and their custom attributes and relations.<br>• A single BI Catalog domain in Collibra with Tableau integration v2 assets.<br><br>> Important The new Tableau operating model is only available in Collibra versions 2021.10 and newer. |

| No. | Step | Details |
|---|---|---|
| 3 | Run the migration script. | The full name of each Tableau integration v1 asset is compared to the full name of the same assets from the Tableau integration v2. When the names match, all of the custom characteristics of the v1 assets are saved to the respective v2 assets.<br><br>Assets of custom v1 asset types are recreated in the specified domain.<br><br>Specifically:<br><br>• The following elements are migrated:<br>◦ Your custom v1 asset types, attribute types and relation types.<br>◦ All assets of your custom v1 asset types.<br>◦ The custom attributes and relations of your custom v1 assets.<br>◦ Any tags that you added to your v1 assets.<br><br>• The following elements are ignored during the migration:<br>◦ All assets of out-of-the-box v1 asset types:<br>■ Their custom attributes and relations, however, are migrated and saved to their respective v2 assets.<br>■ With the exception of Tableau Data Entity, Tableau Report Attribute and Tableau View assets, which are also ignored, but so too are the attributes and relations of such assets.<br>◦ Any attribute types and relation types that are included in the operating model. |
| 4 | Verify the migration results. | Compare your Tableau integration v2 assets to the respective Tableau integration v1 assets. Look to see that the metadata that you manually added to your integration v1 assets has been added to your integration v2 assets. |

| No. | Step | Details |
| --- | --- | --- |
| 5 | Delete your Tableau integration v1 assets and custom assets. | If you've reviewed the migration results and everything looks fine, you can delete your Tableau integration v1 assets and any assets of custom asset types. |

# Overview: Tableau integration v2 and migration

The Tableau integration v2 enables you to harvest Tableau metadata and create new Tableau assets in Data Catalog. Collibra Data Intelligence Cloud analyzes and processes the metadata and presents it as specific asset types, retaining their original names.

## Steps

The following table shows the steps and prerequisites required to ingest metadata in Collibra via lineage harvester (Tableau integration v2) and run the migration script.

> Note
> - This overview assumes that you have already ingested Tableau assets via Tableau integration v1.
> - In the commands that you enter to run the migration, you need to specify which custom asset types, attribute types and relation types you want to migrate.

| Step | What? | Description | Prerequisites |
| --- | --- | --- | --- |
| 1 | Set up Tableau. | Before you start the Tableau integration in Data Catalog, make sure that the lineage harvester can reach the Tableau metadata. Perform these tasks before you start the actual Tableau ingestion process.<br><br>Warning Because these tasks are performed outside of Collibra, it is possible that the content changes without us knowing. We strongly recommend that you carefully read the source documentation. | • You have a Tableau sub-scription. |

| Step | What? | Description | Prerequisites |
| --- | --- | --- | --- |
| 2 | Create a new domain. | Before you can ingest Tableau metadata, you have to create a new domain or choose an existing domain to store the new Tableau assets.<br><br>Warning If you are using Collibra Data Intelligence Cloud 2021.11 or older, you have to add all Tableau attributes in the operating model to a scope and create a scoped assignment before you ingest Tableau via the lineage harvester. For complete information and step-by-step instruction, see Tableau general troubleshooting. | You have a resource role with the following resource permissions:<br><br>• Domain: Add |

| Step | What? | Description | Prerequisites |
|---|---|---|---|
| 3 | Prepare the physical data layer. | You prepare Data Catalog's physical data layer to enable Data Catalog to automatically stitch the Tableau assets to existing assets in Data Catalog. | • You have a global role with the Catalog global permission, for example Catalog Author.<br>• You have set up the JDBC driver of your source data, for example Snowflake.<br>• You have a resource role with the following resource permissions on the **Schema** community:<br>◦ Asset > add<br>◦ Attribute > add<br>◦ Domain > add<br>◦ Attachment > add<br>• You have the permissions to retrieve the metadata of the following database components through the JDBC Driver Database Metadata methods:<br>◦ Schemas<br>◦ Tables<br>◦ Columns |

| Step | What? | Description | Prerequisites |
| --- | --- | --- | --- |
| 4 | Download and install the lineage harvester | You use the lineage harvester to trigger the creation of Tableau assets, their relations and a technical lineage in Data Catalog.<br><br>You can download the lineage harvester from the Collibra Product Resource Downloads page.<br><br>For a list of lineage harvester installation requirements, see About the lineage harvester installation. | • Your environment meets the system requirements to install and use the lineage harvester. |

| Step | What? | Description | Prerequisites |
|---|---|---|---|
| 5 | Prepare the lineage harvester configuration file and run the lineage harvester. | You create a lineage harvester configuration file with Tableau connection information and run the lineage harvester to import the results of the Tableau integration and the technical lineage for Tableau into Data Catalog.<br><br>As a result, you now have a duplicate of your Tableau metadata in Collibra. | • You have downloaded the lineage harvester version 2022.03 or newer.<br>• Your environment meets the system requirements to install and run the lineage harvester.<br>• You have a global role with the Catalog global permission, for example Catalog Author.<br>• You have a global role with the Technical lineage global permission.<br>• You have a global role with the Data Stewardship Manager global permission.<br>• A resource role with the following resource permission on the community level in which you created the BI Data Catalog domain:<br>◦ Asset: add<br>◦ Attribute: add |

| Step | What? | Description | Prerequisites |
|---|---|---|---|
| | | | ◦ Domain: add<br>◦ Attachment: add |
| 6 | Run the migration script | The migration script is triggered by a lineage harvester command. You then use arguments to migrate your customized asset types and custom attribute types and relation types.<br><br>> Note You need lineage harvester version 2022.03.0-5 or newer. We recommend that you use the newest lineage harvester. | Same prerequisites as for the previous step. |
| 7 | Verify the migration results | Compare your Tableau integration v2 assets to the respective Tableau integration v1 assets. Look to see that the metadata that you manually added to your integration v1 assets has been added to your integration v2 assets. | None |

| Step | What? | Description | Prerequisites |
|---|---|---|---|
| 8 | Delete your Tableau integration v1 metadata. | If you've reviewed the migration results and everything looks fine, you can delete your Tableau integration v1 assets and any assets of custom asset types. | • You have a global role with the Catalog global permission, for example Catalog Author.<br>• You have a resource role with the following resource permission on the community level in which you created the BI Data Catalog domain:<br>◦ Asset: Remove<br>◦ Domain: Remove |

## Naming convention

When you synchronize Tableau, Collibra follows a strict naming convention for the names of the new assets. Each asset has a display name and full name. The full name represents the asset path from asset to the database it belongs to. You can freely edit the display name. However, you should never edit the full name, because Data Catalog needs it for a successful migration. Changing the full name may also break the synchronization process.

> Warning We highly recommend that you not edit the full names of any Tableau assets. Doing so will likely lead to errors during the migration and synchronization process.

# Run the migration script

The migration script is triggered by a lineage harvester command. You then use arguments to migrate your customized asset types and custom attribute types and relation types.

## Prerequisites

- You have Collibra Data Intelligence Cloud 2022.01 or newer.
- You have downloaded lineage harvester version 2022.03 or newer and you have the necessary system requirements to run it.
- You have a global role that has the Manage all resources global permission.
- You have a global role with the Catalog global permission, for example Catalog Author.
- You have a global role with the Technical lineage global permission.
- You have a global role with the Data Stewardship Manager global permission.
- You have a resource role with the following resource permission on the community level in which you created the BI Data Catalog domain:
  - Asset: Add
  - Attribute: Add
  - Domain: Add
  - Attachment: Add
- You have tested your connectivity with the Tableau server.

## Steps

1. Run the following command to start the lineage harvester and trigger the migration:
   - Windows: `.\bin\lineage-harvester migrate-tableau <v1_tableau_server_asset_id> <v2_source_id>`
   - for other operating systems: `./bin/lineage-harvester migrate-tableau <v1_tableau_server_asset_id> <v2_source_id>`
2. Use the following arguments to migrate:
   - Customized asset types: `-a <customAssetTypeId>`
   - Custom attribute types: `-t <customAttributeTypeId>`
   - Custom relation types: `-r <customRelationTypeId>`

> Tip You can migrate multiple asset types, attribute types and relation types by repeating the relevant command. In the following example, two asset types are migrated, one after the other, by repeating the `-a` command, followed by the relevant ID of each asset type.

Example

```
./bin/lineage-harvester migrate-tableau 7cc9f692-bbe4-
467f-8ffb-f43545465fcf testtableau22 \
  -a asd13io2-sda2-sdi2-jsd9-asdoi124io12 \
  -a ard86co4-sea5-sc4r-hk39-kjsv9she3hs9 \
  -t 3ffafa8e-029c-4d01-a3c9-1c36e43c2655 \
  -r d0086c90-98e6-4782-b07a-40fcb43845a3
```

## What's next?

- The following elements are migrated:
  - Your custom v1 asset types, attribute types and relation types.
  - All assets of your custom v1 asset types.
  - The custom attributes and relations of your custom v1 assets.
  - Any tags that you added to your v1 assets.
- The following elements are ignored during the migration:
  - All assets of out-of-the-box v1 asset types:
    - Their custom attributes and relations, however, are migrated and saved to their respective v2 assets.
    - With the exception of Tableau Data Entity, Tableau Report Attribute and Tableau View assets, which are also ignored, but so too are the attributes and relations of such assets.
  - Any attribute types and relation types that are included in the operating model.

> Tip You can check the progress of the migration in Activities.

To refresh the Tableau integration v2 metadata, you can run the lineage harvester again using the `full-sync` command, or schedule jobs to run them automatically.

# Soft deletion of your Tableau integration v1 assets

If you've reviewed the migration results and everything looks fine, you can delete your Tableau integration v1 assets and any assets of custom asset types. You can either manually delete the assets or use a lineage harvester argument to perform a soft delete of the assets. Technically speaking, the soft delete does not delete the assets from your Collibra environment; rather, it changes the status of the assets to Obsolete. You can then create an asset filter to view all assets with the status Obsolete, and then manually delete them.

## Prerequisites

- You have Collibra Data Intelligence Cloud 2022.01 or newer.
- You have downloaded lineage harvester version 2022.03 or newer and you have the necessary system requirements to run it.
- You have a global role that has the Manage all resources global permission.
- You have a global role with the Catalog global permission, for example Catalog Author.
- You have a global role with the Technical lineage global permission.
- You have a global role with the Data Stewardship Manager global permission.
- You have a resource role with the following resource permission on the community level in which you created the BI Data Catalog domain:
  - Asset: Update Status

## Steps

1. Run the following command to start the lineage harvester and trigger the migration:
   - Windows: `.\bin\lineage-harvester migrate-tableau --delete <v1_tableau_server_asset_id> <v2_source_id>`
   - for other operating systems: `./bin/lineage-harvester migrate-tableau --delete <v1_tableau_server_asset_id> <v2_source_id>`

Example

```
./bin/lineage-harvester migrate-tableau --delete 7cc9f692-
bbe4-467f-8ffb-f43545465fcf testtableau22
```

Tip You can check the progress of the migration in Activities.

# Tableau general troubleshooting

The following messages and issues can appear when you run the lineage harvester, view a technical lineage or upload the new relations to Data Catalog via Collibra Data Lineage.

| Problem | Solution |
| --- | --- |
| You get connectivity issues with a 401001 error code.<br><br>Unfortunately, 401001 is a very general error code, returned by a Tableau API, that can refer to many issues, including but not limited to the following:<br><br>• The lineage harvester configuration file was configured with the wrong password or Tableau site ID.<br>• SSO authentication was used, which is not supported. | Ensure that the user/token that you intend to use to ingest Tableau assets can authenticate to your Tableau APIs via the command line, from the server on which you intend to install and run the lineage harvester.<br><br>You can test your ability to authenticate by making the signin API call, using a cURL command.<br><br>You can also try checking the login request that the lineage harvester is sending to the Tableau server.<br><br>For complete information and guidance on how to test your ability to connect to the Tableau server and authenticate, see Test connectivity with the Tableau server.<br><br>The authentication token is automatically refreshed in the case of an HTTP 401 Unauthorized error. If the token is refreshed, the log file includes the following line: `Refreshing token.`<br><br>If a refreshed token doesn't resolve the issue, the problem could be that there are too many concurrent HTTP calls, using the same token. Try using the optional concurrencyLevel property in your lineage harvester configuration file, to indicate that you want HTTP requests to be run in a synchronous manner, instead of in parallel. |
| The lineage harvester does not connect to hosts using a proxy server. | Technical lineage does not support proxy server authentication, but you can connect to a proxy server. For complete details, including the necessary commands, see Connecting to a proxy server. |

| Problem | Solution |
| --- | --- |
| You get a TCP timeout error. | To avoid TCP timeout errors, try configuring the Linux TCP keepalive setting:<br>1. Edit your /etc/sysctl.conf file:<br>`# vi /etc/sysctl.conf`<br>2. Add the following settings:<br>`net.ipv4.tcp_keepalive_time = 60`<br>`net.ipv4.tcp_keepalive_intvl = 10`<br>`net.ipv4.tcp_keepalive_probes = 6`<br>3. To load the settings, run the following command:<br>`# sysctl -p` |
| The designation "UNDEFINED" is shown in the technical lineage. | If you are using a Tableau <source ID> configuration file, ensure that you have specified a value for the relevant `col-libraSystemName` property. |

| Problem | Solution |
| --- | --- |
| You get the following error message or a similar certificate error:<br>`Source '<data source name> failed with exception: javax.net.ssl.SSLHandshakeException: General SSLEngine problem` | This message appears when the proxy server sends an unexpected certificate to the lineage harvester or when the default Java TrustStore is empty or outdated.<br>First update Java and rerun the lineage harvester to see if that resolves the issue. If the same error message is shown, try the following:<br>**On Windows**<br>> Note In the following example commands, we refer to the techlin-gcp-us instance. You should refer to the correct Collibra Data Lineage service instance in the geographic location of your Collibra Data Intelligence Cloud environment.<br>1. Run the following command to extract the certificate from the Tableau server:<br>`keytool -printcert -rfc -sslserver techlin-gcp-us.collibra.com:443 > tableau-cert.crt`<br>> Tip Replace the URL techlin-gcp-us.collibra.com with the URL for your Tableau server, which you specify in the lineage harvester configuration file. This will create a file named tableau-cert.crt in the folder where you run this command.<br>2. Run the following command to find the location of your JAVA_HOME:<br>`echo %JAVA_HOME%`<br>» The location path will be something like the following:<br>`C:\Program Files\Java\jdk-17.0.2`<br>3. Use the location path of your JAVA_HOME in the following command, to import the tableau-cert.crt file into the cacerts file found above.<br>`keytool -importcert -file tableau-cert.crt -` |

| Problem | Solution |
| --- | --- |
| | `alias "TableauProdServerCert" -keystore "C:\Program Files\Java\jdk-17.0.2\cacerts"`<br>> Note You can specify a different alias, if you want.<br>4. Run the following command:<br>`keytool -list -keystore "C:\Program Files\Java\jdk-17.0.2\lib\security\cacerts" \| findstr "Tableau"`<br>5. Enter the keystore password.<br>> Tip The password is typically `changeit`.<br>» A list of all certificates that match the Tableau string in the "C:\Program Files\Java\jdk-17.0.2\cacerts" file is shown.<br>> Tip In the list of certificates, look for the one that you imported in step 3. If it's listed, it means the "C:\Program Files\Java\jdk-17.0.2\cacerts" file has the certificate needed to validate the Tableau server.<br>6. Run the following command to have the lineage harvester use the cacerts file that you just updated.<br>`set JAVA_OPTS=-Djavax.net.ssl.trustStore="C:\Program Files\Java\jdk-17.0.2\lib\security\cacerts" -Djavax.net.ssl.trustStorePassword="changeit"`<br>7. Run the following command to test the synchronization:<br>`./lineage-harvester.bat full-sync -s tableau`<br>**On Linux** |

| Problem | Solution |
| --- | --- |
| | Note<br>• In the following example commands, we refer to the techlin-gcp-us instance. You should refer to the correct Collibra Data Lineage service instance in the geographic location of your Collibra Data Intelligence Cloud environment.<br>• If you want to add an existing certificate to the Java TrustStore, instead of creating a new Keystore, replace "<your keystore name>" in steps 2 and 3, with the path to the `cacerts` file in your Java installation, for example *%JAVA_HOME%\jre\lib\cacerts*.<br><br>1. Use the following command to get a certificate from the corresponding techlin-gcp-us.com site, which is part of the CollibraData Lineage infrastructure:<br>`openssl x509 -in <(openssl s_client -connect techlin-gcp-us.collibra.com:443 -prexit 2>/dev/null) -out techlin-gcp-us.crt`<br>Tip If you already have a correctly formatted certificate on the server, you can skip this step.<br>2. Add the certificate to the Java TrustStore:<br>`keytool -importcert -file techlin-gcp-us.crt -alias techlin-gcp-us -keystore <your key-store name> -storepass changeit`<br>3. Run the lineage harvester and use the new TrustStore using the following parameter:<br>`-Djavax.net.ssl.trustStore=<your keystore name>` |

| Problem | Solution |
| --- | --- |
| | Example To synchronize your data sources again, run the following command:<br>`./bin/lineage-harvester full-sync -Djavax.net.ssl.trustStore=mykeystore` |

| Problem | Solution |
| --- | --- |
| You get an external system ID mapping error. | The error message looks similar to the following:<br><br>`PROCESSING ERROR: "syncer.domain.DgcSyncError: java.lang.Exception: Unexpected DGC job status: ERROR Error message: { "type" : "MESSAGE", "message" : "A mapping for the external system id 'd0f3a21a2324fa117112409b-dea6ade7' and resource '59cf9293-fca1-4f78-99ab-31c150a23626' already exists." } Caused by: java.lang.Exception: Unexpected DGC job status: ERROR Error message: { "type" : "MESSAGE", "message" : "A mapping for the external system id 'd0f3a21a2324fa117112409b-dea6ade7' and resource '59cf9293-fca1-4f78-99ab-31c150a23626' already exists." }"`<br><br>Please create a support ticket and provide your answers to the following two questions.<br><br>Note Refer to the example error message above and replace the IDs of the external system and the mapped resource with those in the error message you received.<br><br>• What is the asset type of the mapped asset? In this example, the asset with ID 59cf9293-fca1-4f78-99ab-31c150a23626?<br><br>Tip To view the asset type, go to the following URL: <your-collibra-platform-url>/asset/59cf9293-fca1-4f78-99ab-31c150a23626<br><br>• What is the mapping definition? |

| Problem | Solution |
| --- | --- |
| | Tip To view the mapping definition, go to the following URL: <your-collibra-platform-url>/rest/2.0/mappings/externalSystem/d0f3a21a2324fa117112409bdea6ade7/mappedResource/59cf9293-fca1-4f78-99ab-31c150a23626 |

<table>
<tr><th>Problem</th><th>Solution</th></tr>
<tr><td>If you are using Collibra Data Intelligence Cloud 2021.11 or older, you have to add all Tableau attributes in the operating model to a scope and create a scoped assignment before you ingest Tableau via the lineage harvester.</td><td>

**Show me how to add attributes to a scope and create a scoped assignment**

## Prerequisites

- You are using Collibra 2021.11 or older.
- You have a global role that has the System administration global permission.

## Steps

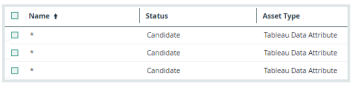
1. On the main menu, click ⠿, then ⚙ **Settings**.
   » The Collibra settings page opens.
2. In the tab pane, click **Scopes**.
3. Above the table, to the right, click **Add**.
   » The **Create Scope** dialog box appears.
4. Enter the required information.

| Field | Description |
|---|---|
| Name | The name of the scope. |
| Description | The description of the scope, for example to add extra details. |

5. Click **Save**.
6. Open a Tableau asset type:
   a. On the main menu, click ⠿, then ⚙ **Settings**.
      » The Collibra settings page opens.
   b. In the tab pane, click **Asset Types**.
      » The asset type table appears.
   c. In the overview of asset types, click an asset type.
   » The **Asset type** editor opens.

</td></tr>
</table>

| Problem | Solution |
| --- | --- |
| | Important You need to do this for each of the following asset types:<br>◦ Tableau Project<br>◦ Tableau Site<br>◦ Tableau Workbook<br>◦ Tableau Data Attribute<br>◦ Tableau Server<br>There are other Tableau asset types, but they do not require the scoped assignment.<br><br>7. In the tab pane, click **Add assignment**.<br>» The **Select scope for this assignment** dialog box appears.<br>8. Select the custom scope that you have created for Tableau assets.<br>Note You can only add one scope at a time.<br><br>9. Click **Add assignment**.<br>» The settings of the global assignment are copied into the selected scope.<br><br>Warning After you've created the scoped assignment, do not change the assignment itself. The sole purpose of the scoped assignment is to ingest read-only attributes for which you normally need a system user.<br><br>Note If you ingest Tableau metadata in a Collibra version 2021.09 or older, you must also manually create two new relation types and add them to the Tableau <source ID> configuration file. |

| Problem | Solution |
| --- | --- |
| In Collibra, the name of a Tableau Data Attribute name is an asterisk (*).<br>  <br> | If you ingest a Tableau dataset that doesn't have any attributes, asterisks (*) are shown as the Tableau Data Attribute asset names in Collibra. |

| Problem | Solution |
| --- | --- |
| You get the following error message:<br><br>`Can't show all data because the timeout limit PT1M has been exceeded. Use pagination, additional filtering, or both in the query, and try again.` | To resolve this error, try lowering the relevant paging option value (or values) for the failing query. Consider the following failure summary:<br><br>`harvester.AbstractHarvester - Failure summary:`<br>`  Source 'tabemeat' failed with exception: harvester.error.TableauHarvesterError: Failed to process Tableau source: Failed to execute metadata query. Errors: [{"message":"Can't show all data because the timeout limit PT1M has been exceeded. Use pagination, additional filtering, or both in the query, and try again."}].`<br>`  Query: query worksheets($cursor: String, $pageSize: Int, $nestedPageSize: Int) {`<br>`  sheetsConnection(first: $pageSize, after: $cursor) {`<br>`  nodes {`<br>`  id`<br>`  name`<br>`  luid`<br>`  ....`<br><br>In this example, the query name is worksheets:<br><br>`Query: query worksheets($cursor: String, $pageSize: Int, $nestedPageSize: Int)`<br><br>Try lowering one or both of the paging option values for worksheets:<br><br>• worksheetsPageSize (default value is 100)<br>• worksheetsFieldsPageSize (default value is 100)<br><br>**Show me all of the paging options and default values for each query name**<br><br>Query name: dashboards — Paging options (default value): dashboardsPageSize (100)<br>Query name: databases — Paging options (default value): databasesPageSize (100) |

<table>
<tr><th>Problem</th><th>Solution</th></tr>
<tr><td></td><td>
<table>
<tr><th>Query name</th><th>Paging options (default value)</th></tr>
<tr><td>datasourceFields</td><td>datasourceFieldsPageSize (100)<br><br>Plus the following limits:<br><br>• columnsLimit (20)<br>• fieldsLimit (20)</td></tr>
<tr><td>datasources</td><td>• datasourcesPageSize (50)<br>• datasourcesFieldsPageSize (50)<br><br>Plus the following limits:<br><br>• columnsLimit (20)<br>• fieldsLimit (20)</td></tr>
<tr><td>tableColumns</td><td>tableColumnsPageSize (1000)</td></tr>
<tr><td>tables</td><td>• tablesPageSize (100)<br>• tablesColumnsPageSize (100)</td></tr>
<tr><td>worksheetFields</td><td>worksheetFieldsPageSize (1000)</td></tr>
<tr><td>worksheets</td><td>• worksheetsPageSize (100)<br>• worksheetsFieldsPageSize (100)</td></tr>
</table>
</td></tr>
</table>

# Test your connectivity with the Tableau server

Before you run the lineage harvester, you need to test your connectivity with the Tableau server.

Connectivity requires authentication. The user/token that you intend to use to ingest Tableau assets must be able to authenticate to your Tableau APIs via the command line, from the server on which you intend to install and run the lineage harvester.

> Warning As of October 2022, Tableau is enforcing multi-factor authentication for Tableau Cloud Admin users. However, the lineage harvester doesn't support multi-factor authentication. Therefore, Tableau Cloud users with an Admin role must use token-based authentication. This does not affect Tableau Server users or Tableau Cloud users with an Explorer role.

To ensure that you can authenticate and connect to the Tableau server, try the following procedures.

## Make the signin API call using a cURL command

1. Create a JSON file called "signin.json".
   The file should contain the following:
   - For username/password authentication:

```
{
    "credentials": {
        "name": "YOUR_USER",
        "password": "YOUR_PASSWORD",
        "site": {
            "contentUrl": "YOUR_SITE_ID"
        }
    }
}
```

   - For personal token-based authentication:

```
{
    "credentials": {
        "personalAccessTokenName": "YOUR_TOKEN_NAME",
        "personalAccessTokenSecret": "YOUR_TOKEN_SECRET",
        "site": {
            "contentUrl": "YOUR_SITE_ID"
        }
    }
}
```

2. Test this on your machine by running the following command:
   *curl "https://YOUR_TABLEAU_URL/api/3.7/auth/signin" -H "Content-Type: application/json" -X POST -d @signin.json*

> Tip To test on a Windows machine, you need to:
> a. Download and install the cURL Command-Line Tool.
> b. In Windows, click **Start** > **Run**, and then enter *cmd* in the **Run** dialog box.
> c. Run the following command:
> *curl "https://YOUR_TABLEAU_URL/api/3.7/auth/signin" -H "Content-Type: application/json" -X POST -d @signin.json*

## Check the login request that the lineage harvester sends to the Tableau server

1. Run the lineage harvester with the following parameters:

   ```
   bin/lineage-harvester load-sources -Dakka.http.client.log-unencrypted-network-bytes=1024 -Dakka.loglevel=DEBUG
   ```

   This generates many logs. In the log file, search for “signin”. The entry for “signin” will resemble the following log snippet, in which the login request is shown between curly brackets "{}":

   ```
   [DEBUG] [11/08/2021 14:03:18.411] [default-akka.act-
   or.default-dispatcher-4] [akka.stream.Log(akka://de-
   fault/system/StreamSupervisor-1)] [client-plain-text ToNet
   ] Element: SendBytes  ByteString(375 bytes)
   50 4F 53 54 20 2F 61 70 69 2F 33 2E 37 2F 61 75  | POST /ap-
   i/3.7/au
   74 68 2F 73 69 67 6E 69 6E 20 48 54 54 50 2F 31  |
   th/signin HTTP/1
   ```

2. Verify that the request is the same as the one you used in the signin.json file.

# Working with Power BI service

Power BI service is a cloud business intelligence software that helps you see and understand your data. You can ingest Power BI metadata in Data Catalog and create a technical lineage.

The Power BI service integration in Collibra Data Intelligence Cloud is not the same as the Power BI Report Server integration. If you want to ingest Power BI Report Server metadata in Collibra Data Intelligence Cloud, please read the Power BI Report Server section in the Documentation Center.

If you want to ingest Power BI metadata in Data Catalog, you have to purchase the Power BI connector and lineage feature.

> Tip If you previously integrated Power BI metadata via the Power BI harvester, you can now migrate your existing Power BI assets to the new integration method.

## Features

Collibra Data Lineage currently supports two means by which to integrate Power BI in Data Catalog. The following table shows the features specific to the two integration methods.

| Feature | Integration via the Power BI harvester (deprecated) | Integration via the lineage harvester |
|---|---|---|
| Catalog ingestion | ✓ | ✓ |
| Technical lineage | ✓ | ✓ |
| Automatic stitching | ✓ | ✓ |
| Uses only one harvester | | ✓ |
| No Windows dependency | | ✓ |

| Feature | Integration via the Power BI harvester (deprecated) | Integration via the lineage harvester |
|---|---|---|
| Uses new Power BI APIs | | ✓ |
| Workspace filtering for high-volume data | | ✓ |
| Mapping to target domains | | ✓ |
| Data flow support | | ✓ |

> Tip After we have successfully switched to the new Power BI APIs, we will have more opportunities to improve the integration. You can add your ideas for product enhancements and new features in the Collibra Integrations Ideation Portal.

# Power BI terminology

Before you ingest Power BI, read more about the Power BI terminology and how it maps with the Collibra Data Intelligence Cloud asset types.

> Note For more information, see the Power BI documentation.

| Power BI term | Description | Asset type in Collibra |
|---|---|---|
| Capacity | A resource that hosts Power BI Workspaces. | Power BI Capacity |

| Power BI term | Description | Asset type in Collibra |
|---|---|---|
| Dashboard | A collection of Power BI tiles with metrics from one or more Reports and Data Models. | Power BI Dashboard |
| Dataflow | A collection of tables that are created and managed in workspaces in the Power BI service. | Power BI Data Flow |
| Data Set | A collection of data that is used to create a Power BI report. | Power BI Data Model |
| Data Set Column | A column in a Power BI Data Model. | Power BI Column |
| Data Set Table | A table in a Power BI Data Model. | Power BI Table |
| Report | A detailed view of a Power BI Data Model, with visualizations of findings and insights. | Power BI Report |
| Server or Tenant | A visual analytics platform for creating and storing Power BI Reports and Data Models. | Power BI Server |
| Tile | An element representing data on the Power BI Dashboard. | Power BI Tile |
| Workspace | A collection of Power BI Dashboards, Reports and Data Models. | Power BI Workspace |

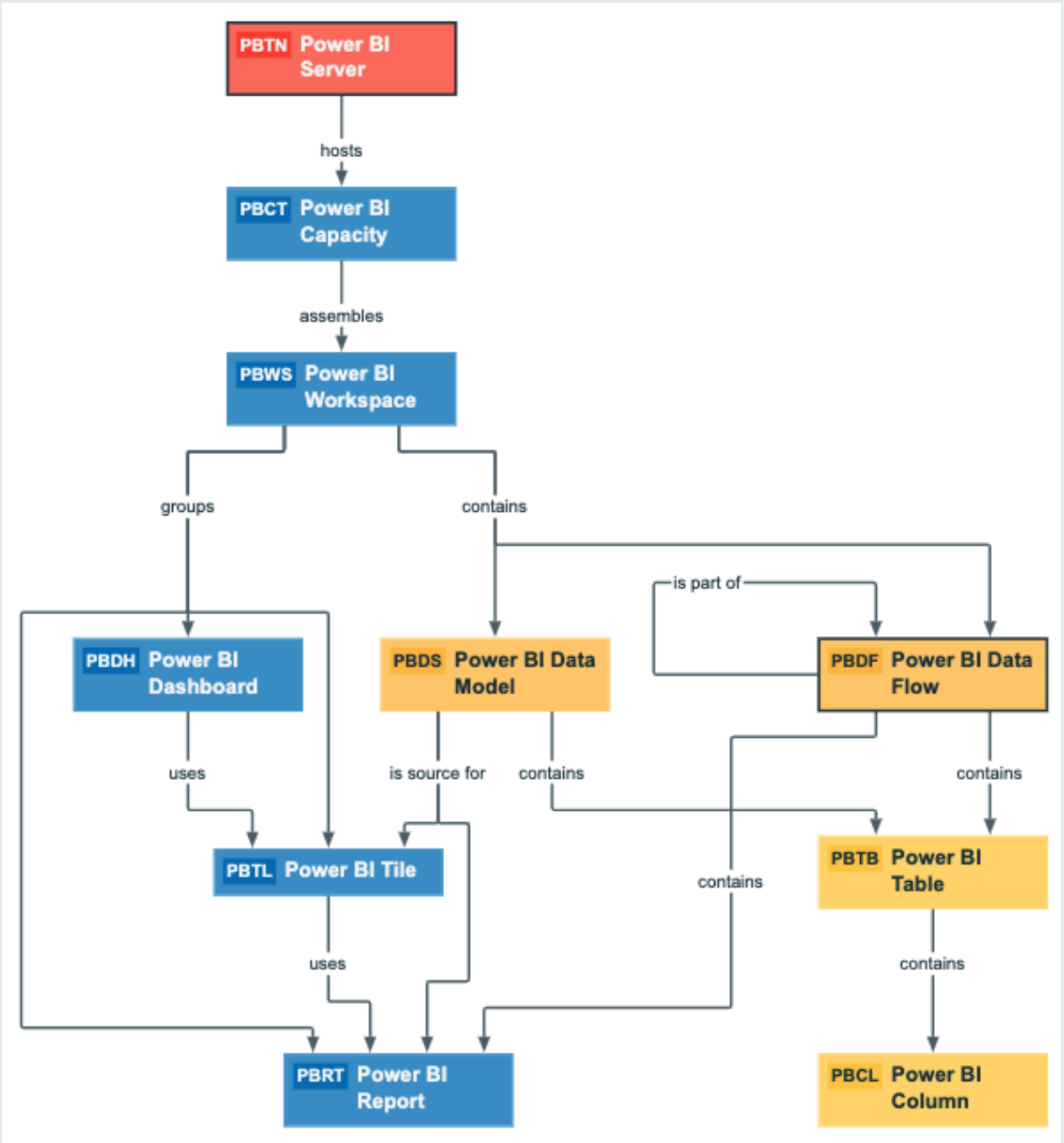
# Power BI operating model

The lineage harvester collects Power BI metadata and sends it to the Collibra Data Lineage service. Collibra processes the metadata and creates new Power BI assets and relations in Data Catalog. You can see them on the asset page overview or visualize them in a diagram or in a technical lineage.

> Note
> - The assets have the same names as their counterparts in Power BI. Full names and Display names cannot be changed in Data Catalog.
> - Asset types are only created if you have all specific Power BI and Data Catalog permissions.
> - The Power BI assets are created in the domain (or domains) that you specify in the lineage harvester configuration file.
> - Relations that were created between Power BI assets and other assets via a relation type in the Power BI operating model, are deleted upon synchronization. The same is true of any attribute types in the operating model that you add to Power BI assets. To ensure that the characteristics you add to Power BI assets are not deleted upon synchronization, be sure to use characteristics that are not part of the Power BI operating model.

# Power BI metadata overview

The following image shows the relations between Power BI asset types.




# Harvested metadata per asset type

This table shows the harvested Power BI metadata for each Power BI asset type. This table also shows the resource ID for each asset type, attribute, and relation.

| Asset type | Synchronized metadata | Resource ID |
|---|---|---|
| Power BI Capacity<br>Resource ID: 00000000-0000-0000-0000-100000000002 | Full name | |
| | Display name | |
| | Server hosts / is hosted in Business Dimension | 00000000-0000-0000-0000-120000000000 |
| | BI Folder assembles / is assembled in BI Folder | 00000000-0000-0000-0000-120000000001 |
| Power BI Column<br>Resource ID: 00000000-0000-0000-0000-100000000008 | Full name | |
| | Display name | |
| | Description | 00000000-0000-0000-0000-000000003114 |
| | Role in Report | 00000000-0000-0000-0000-000000000266 |
| | Technical Data Type | 00000000-0000-0000-0000-000000000219 |
| | BI Data Model contains / is part of BI Data Attribute | 00000000-0000-0000-0000-000000007196 |
| | Data Element targets / sources Data Element | 00000000-0000-0000-0000-000000007069 |
| | Data Entity contains / is part of Data Attribute | 00000000-0000-0000-0000-000000007047 |

| Asset type | Synchronized metadata | Resource ID |
|---|---|---|
| Power BI Dashboard<br>Resource ID: 00000000-0000-0000-0000-100000000004 | Full name | |
| | Display name | |
| | URL | 00000000-0000-0000-0000-000000000258 |
| | Data asset is source / Source for BI Report | 00000000-0000-0000-0000-120000000013 |
| | Report uses / used in Report | 00000000-0000-0000-0000-120000000007 |
| | Report related to / impacted by Business Asset | 00000000-0000-0000-0000-120000000006 |
| Power BI Data Flow<br>Resource ID: 00000000-0000-0000-0000-100000000010 | Full name | |
| | Display name | |
| | BI Data Model contains / is part of BI Data Attribute | 00000000-0000-0000-0000-000000007196 |
| | BI Folder contains / contained in Data Asset | 00000000-0000-0000-0000-120000000014 |
| | Data Entity is part of / contains Data Model | 00000000-0000-0000-0000-000000007046 |

| Asset type | Synchronized metadata | Resource ID |
|---|---|---|
| Power BI Data Model<br><br>Resource ID: 00000000-0000-0000-0000-100000000007 | Full name | |
| | Display name | |
| | Owner in source<br><br>The only harvested metadata are email addresses. | 00000000-0000-0000-0000-200000000001 |
| | BI Data Model contains / is part of BI Data Attribute | 00000000-0000-0000-0000-000000007196 |
| | BI Folder contains / contained in Data Asset | 00000000-0000-0000-0000-120000000014 |
| | Data Asset is source for / source BI report | 00000000-0000-0000-0000-120000000013 |
| | Data Entity is part of / contains Data Model | 00000000-0000-0000-0000-000000007046 |

| Asset type | Synchronized metadata | Resource ID |
|---|---|---|
| Power BI Report<br>Resource ID: 00000000-0000-0000-0000-100000000006 | Full name | |
| | Display name | |
| | Description | 00000000-0000-0000-0000-000000003114 |
| | Owner in source<br>The only harvested metadata are email addresses. | 00000000-0000-0000-0000-200000000001 |
| | URL | 00000000-0000-0000-0000-000000000258 |
| | Business Dimension groups / is grouped into Report | 00000000-0000-0000-0000-120000000002 |
| | Data Asset is source for / source BI Report | 00000000-0000-0000-0000-120000000013 |
| | Report related to / impacted by Business Asset | 00000000-0000-0000-0000-120000000006 |
| | Report uses / used in Report | 00000000-0000-0000-0000-120000000007 |
| Power BI Server<br>Resource ID: 00000000-0000-0000-0000-100000000001 | Full name | |
| | Display name | |
| | Server hosts / is hosted in Business Dimension | 00000000-0000-0000-0000-120000000000 |

| Asset type | Synchronized metadata | Resource ID |
|---|---|---|
| Power BI Table<br>Resource ID: 00000000-0000-0000-0000-100000000009 | Full name | |
| | Display name | |
| | Description | 00000000-0000-0000-0000-000000003114 |
| | Data Entity contains / is part of Data Attribute | 00000000-0000-0000-0000-000000007047 |
| | Data Entity is part of / contains Data Model | 00000000-0000-0000-0000-000000007046 |
| Power BI Tile<br>Resource ID: 00000000-0000-0000-0000-100000000005 | Full name | |
| | Display name | |
| | URL | 00000000-0000-0000-0000-000000000258 |
| | Business Dimension groups / is grouped into Report | 00000000-0000-0000-0000-120000000002 |
| | Data Asset is source for / source BI Report | 00000000-0000-0000-0000-120000000013 |
| | Report related to / impacted by Business Asset | 00000000-0000-0000-0000-120000000006 |
| | Report uses / used in Report | 00000000-0000-0000-0000-120000000007 |

| Asset type | Synchronized metadata | Resource ID |
|---|---|---|
| Power BI Workspace<br>Resource ID: 00000000-0000-0000-0000-100000000003 | Full name | |
| | Display name | |
| | Description | 00000000-0000-0000-0000-000000003114 |
| | State | 00000000-0000-0000-0000-000000000227 |
| | Owner in source<br>The only harvested metadata are email addresses. | 00000000-0000-0000-0000-200000000001 |
| | BI Folder assembles / is assembled in BI Folder | 00000000-0000-0000-0000-120000000001 |
| | BI Folder contains / contained in Data Asset | 00000000-0000-0000-0000-120000000014 |
| | Business Dimension groups / is grouped into Report | 00000000-0000-0000-0000-120000000002 |

Note The metadata that is shown on the assets' pages depends on the asset type's assignment. As a result, you might not see all harvested metadata on the asset's page by default.

# Additional information

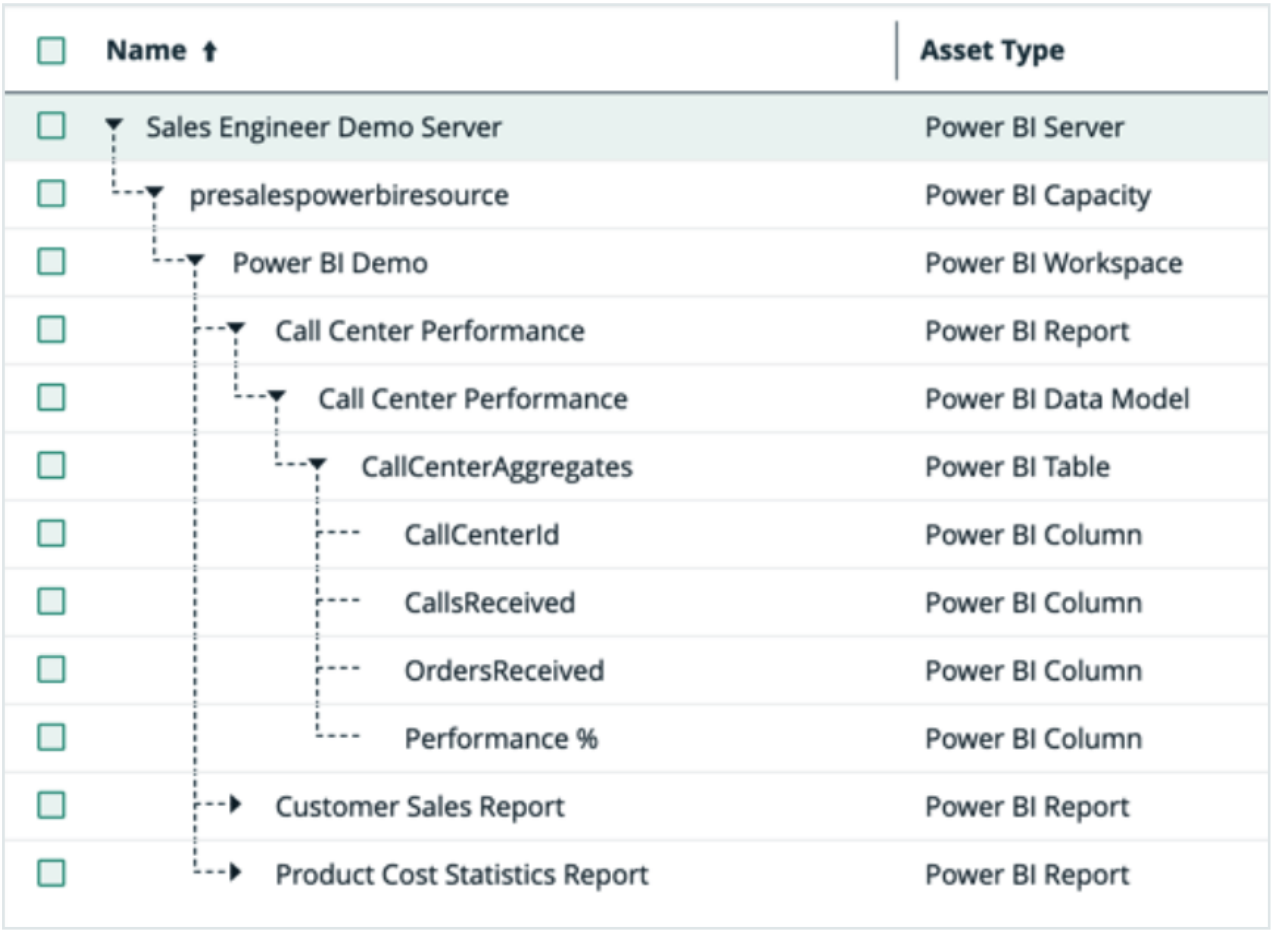
For the Owner in source attribute, the following rules apply:

- If the system creates a Power BI data object and the Power BI data object does not have a user ID, the Owner in source attribute is shown as System on the asset page.

- If the user who created a Power BI data object no longer exists, the Owner in source attribute is shown as empty on the asset page.

# Example of ingested Power BI metadata

The following image shows an example structure after Power BI ingestion.




# Recommended hierarchy within a domain

You can enable hierarchies for the domain (or domains) in which your Power BI assets were ingested. Doing so makes it easier to understand the relation between your Power BI assets, when viewing the assets on the domain page.

Follow these steps to enable and configure the recommended hierarchy.

## Steps

1. Open the domain page of the relevant BI Catalog domain.
2. In the content toolbar, click .
   » The **Configure Hierarchy** dialog box appears.
3. Select **Enable Hierarchy**.
4. Select **Single path**.
5. Start typing and select each of the following relation types:
   - Server **hosts** Business Dimension
   - BI Folder **assembles** BI Folder
   - Business Dimension **groups** Report
   - BI Report **source** Data Asset
   - Data Model **contains** Data Entity
   - Data Entity **contains** Data Attribute
6. Click **Apply**.

> Note In an asset view, if any asset is deleted, for example via synchronization or manual deletion, the view is recreated and the hierarchy is lost. In this case, you can again enable and configure the recommended hierarchy.

# Create a Power BI operating model diagram view

You can create a Power BI-specific diagram view, to visualize the operating model. The following procedure provides instruction on how to quickly create a new diagram view by copying and pasting the JSON code in the diagram view text editor.

## Steps

1. Open an asset page.
2. In the tab pane, click **Diagram**.
   » The diagram appears in the default diagram view.
3. Click + to add a new view.
4. Click the **Text** tab, to switch to the diagram view text editor.
5. Click **Show me the JSON code** below this procedure, to expand the code.
6. Paste the code in diagram view text editor.

7. Click **Save**.
8. Edit the name and description of the diagram view, to suit your needs.

### Show me the JSON code


```
{
   "nodes": [
        {
            "id": "Power BI Server",
            "type": {
              "id": "00000000-0000-0000-0000-100000000001"
        },
            "fields": []
        },
        {
            "id": "Power BI Capacity",
            "type": {
              "id": "00000000-0000-0000-0000-100000000002"
            }
        },
        {
            "id": "Power BI Workspace",
            "type": {
              "id": "00000000-0000-0000-0000-100000000003"
            }
        },
        {
            "id": "Power BI Dashboard",
            "type": {
              "id": "00000000-0000-0000-0000-100000000004"
            }
        },
        {
            "id": "Power BI Report",
            "type": {
              "id": "00000000-0000-0000-0000-100000000006"
            }
        },
        {
            "id": "Power BI Tile",
            "type": {
              "id": "00000000-0000-0000-0000-100000000005"
            }
        },
        {
            "id": "Power BI Data Model",
            "type": {
              "id": "00000000-0000-0000-0000-100000000007"
            }
        },
```

```
        {
            "id": "Power BI Data Flow",
            "type": {
              "id": "00000000-0000-0000-0000-100000000010"
            }
        },
        {
            "id": "Power BI Table",
            "type": {
              "id": "00000000-0000-0000-0000-100000000009"
            }
        },
        {
            "id": "Power BI Column",
            "type": {
              "id": "00000000-0000-0000-0000-100000000008"
            }
        }
    ],
    "edges": [
        {
            "from": "Power BI Server",
            "to": "Power BI Capacity",
            "label": "",
            "style": "arrow",
            "type": {
              "id": "00000000-0000-0000-0000-120000000000"
            },
            "roleDirection": true
        },
        {
            "from": "Power BI Capacity",
            "to": "Power BI Workspace",
            "label": "",
            "style": "arrow",
            "type": {
              "id": "00000000-0000-0000-0000-120000000001"
            },
            "roleDirection": true
        },
        {
            "from": "Power BI Workspace",
            "to": "Power BI Dashboard",
            "label": "",
            "style": "arrow",
            "type": {
              "id": "00000000-0000-0000-0000-120000000002"
            },
            "roleDirection": true
        },
```

```
        {
            "from": "Power BI Workspace",
            "to": "Power BI Report",
            "label": "",
            "style": "arrow",
            "type": {
              "id": "00000000-0000-0000-0000-120000000002"
            },
            "roleDirection": true
        },
        {
            "from": "Power BI Workspace",
            "to": "Power BI Tile",
            "label": "",
            "style": "arrow",
            "type": {
              "id": "00000000-0000-0000-0000-120000000002"
            },
            "roleDirection": true
        },
        {
            "from": "Power BI Workspace",
            "to": "Power BI Data Model",
            "label": "",
            "style": "arrow",
            "type": {
              "id": "00000000-0000-0000-0000-120000000014"
            },
            "roleDirection": true
        },
        {
            "from": "Power BI Workspace",
            "to": "Power BI Data Flow",
            "label": "",
            "style": "arrow",
            "type": {
            "id": "00000000-0000-0000-0000-120000000014"
            },
            "roleDirection": true
        },
        {
            "from": "Power BI Dashboard",
            "to": "Power BI Tile",
            "label": "",
            "style": "arrow",
            "type": {
              "id": "00000000-0000-0000-0000-120000000007"
            },
            "roleDirection": true
        },
```

```
        {
            "from": "Power BI Data Model",
            "to": "Power BI Tile",
            "label": "",
            "style": "arrow",
            "type": {
              "id": "00000000-0000-0000-0000-120000000013"
            },
            "roleDirection": true
        },
        {
            "from": "Power BI Data Model",
            "to": "Power BI Report",
            "label": "",
            "style": "arrow",
            "type": {
              "id": "00000000-0000-0000-0000-120000000013"
            },
            "roleDirection": true
        },
        {
            "from": "Power BI Data Model",
            "to": "Power BI Table",
            "label": "",
            "style": "arrow",
            "type": {
              "id": "00000000-0000-0000-0000-000000007196"
            },
            "roleDirection": true
        },
        {
            "from": "Power BI Data Flow",
            "to": "Power BI Data Flow",
            "label": "",
            "style": "arrow",
            "type": {
              "id": "00000000-0000-0000-0000-000000007046"
            },
            "roleDirection": true
        },
        {
            "from": "Power BI Data Flow",
            "to": "Power BI Table",
            "label": "",
            "style": "arrow",
            "type": {
              "id": "00000000-0000-0000-0000-120000000014"
            },
            "roleDirection": true
        },
```

```
        {
            "from": "Power BI Tile",
            "to": "Power BI Report",
            "label": "",
            "style": "arrow",
            "type": {
             "id": "00000000-0000-0000-0000-120000000007"
            },
            "roleDirection": true
        },
        {
            "from": "Power BI Table",
            "to": "Power BI Column",
            "label": "",
            "style": "arrow",
            "type": {
              "id": "00000000-0000-0000-0000-000000007047"
            },
            "roleDirection": true
        },
        {
            "from": "Power BI Data Flow",
            "to": "Power BI Report",
            "label": "",
            "style": "arrow",
            "type": {
              "id": "00000000-0000-0000-0000-000000007196"
            },
            "roleDirection": true
        }
    ],
    "showOverview": false,
    "enableFilters": true,
    "showLabels": true,
    "showFields": true,
    "showLegend": true,
    "showPreview": true,
    "visitStrategy": "directed",
    "layout": "HierarchyTopBottom",
    "maxNodeLabelLength": 50,
    "maxEdgeLabelLength": 30,
    "layoutOptions": {
        "compactGroups": false,
        "componentArrangementPolicy": "topmost",
        "edgeBends": true,
        "edgeBundling": true,
        "edgeToEdgeDistance": 5,
        "minimumLayerDistance": "auto",
        "nodeToEdgeDistance": 5,
        "orthogonalRouting": true,
```

```
        "preciseNodeHeightCalculation": true,
        "recursiveGroupLayering": true,
        "separateLayers": true,
        "webWorkers": true,
        "nodePlacer": {
            "barycenterMode": true,
            "breakLongSegments": true,
            "groupCompactionStrategy": "none",
            "nodeCompaction": false,
            "straightenEdges": true
        }
    }
}
```

# Power BI asset and domain types

The Power BI integration in Collibra Data Intelligence Cloud uses a specific subset of packaged asset types and domain types.

The following table contains the asset and domain types that are used for the Power BI integration. You can see the parent asset types in the breadcrumbs above each asset type.

| Asset type | Description | Domain type |
| --- | --- | --- |
| Business Asset ▸ Business Dimension ▸ BI Folder ▸ Power BI Capacity | A resource that hosts Power BI Workspaces. | BI Catalog |

| Asset type | Description | Domain type |
| --- | --- | --- |
| Business Asset ▸ Business Dimension ▸ BI Folder ▸ Power BI Workspace | A collection of Power BI Dashboards, Reports and Data Models. | BI Catalog |
| Business Asset ▸ Report ▸ BI Report ▸ Power BI Dashboard | A collection of Power BI tiles with metrics from one or more Reports and Data Models. | BI Catalog |
| Business Asset ▸ Report ▸ BI Report ▸ Power BI Report | A detailed view of a Power BI Data Model, with visualizations of findings and insights. | BI Catalog |
| Business Asset ▸ Report ▸ BI Report ▸ Power BI Tile | An element representing data on the Power BI Dashboard. | BI Catalog |
| Data Asset ▸ Data Element ▸ Data Attribute ▸ BI Data Attribute ▸ Power BI Column | A column in a Power BI Data Model. | BI Catalog |

| Asset type | Description | Domain type |
| --- | --- | --- |
| Data Asset ▸ Data Structure ▸ Data Entity ▸ BI Data Entity ▸ Power BI Table | A table in a Power BI Data Model. | BI Catalog |
| Data Asset ▸ Data Structure ▸ Data Model ▸ BI Data Model ▸ Power BI Data Flow | A collection of tables that are created and managed in workspaces in the Power BI service. | BI Catalog |
| Data Asset ▸ Data Structure ▸ Data Model ▸ BI Data Model ▸ Power BI Data Model | A collection of data that is used to create a Power BI report. | BI Catalog |
| Technology Asset ▸ Server ▸ BI Server ▸ Power BI Server | A visual analytics platform for creating and storing Power BI Reports and Data Models. | BI Catalog |

# Overview Power BI integration steps

The Power BI integration enables you to harvest Power BI metadata and create new Power BI assets in Data Catalog. Collibra analyzes and processes the BI metadata and presents it as specific asset types, retaining their original names.

# Steps

The table below shows the steps and prerequisites required to integrate Power BI in Data Catalog. These steps are best practices, which means that some of them might be optional, but highly recommended.

> Important In the global assignment of each asset type included in the Power BI operating model, ensure that none of the characteristics that are in the operating model have a maximum cardinality of “0”. If the maximum cardinality is set to “0” for any such characteristics, ingestion will fail.

| Step | What? | Description | Prerequisites |
|---|---|---|---|
| 1 | Set up a Power BI application. | Before you start the Power BI integration in Data Catalog, make sure that the lineage harvester can reach the Power BI metadata. Perform these tasks before you start the actual Power BI ingestion process:<br>• The authentication process.<br>• The registration of your Power BI application in Microsoft Azure.<br>• The Power BI roles and dedicated capacities for Power BI workspaces.<br>• The required Power BI subscription.<br>Warning Because these tasks are performed outside of Collibra, it is possible that the content changes without us knowing. We strongly recommend that you carefully read the source documentation. | • You have a Power BI subscription. |

| Step | What? | Description | Prerequisites |
|---|---|---|---|
| 2 | Prepare one or more new domains. | Before you can ingest Power BI metadata, you have to designate a domain for storing the new Power BI assets. You can choose an existing domain or create one or more new domains.<br><br>Note Make note of the reference ID of the domain. You need to mention the reference ID in the lineage harvester configuration file. | • You have a resource role with the following resource permissions:<br>◦ Domain: Add |
| 3 | Optionally, assign the attribute type State to the global assignment of the Power BI Workspace asset type. | On Power BI Workspace asset pages, you can include the attribute type State, to show the state of ingested Power BI workspaces. To do so, you have to edit the global assignment of the Power BI Workspace asset type and assign the attribute type State.<br><br>If you delete a Power BI workspace, the workspace is maintained for a 90-day grace period. During the grace period, the workspace has the state Deleted. When you ingest Power BI metadata in Data Catalog, this deleted workspace is ingested.<br><br>For complete information on Power BI workspaces and possible states, see the Microsoft Power BI documentation. | You have a global role that has the System administration global permission. |

| Step | What? | Description | Prerequisites |
| --- | --- | --- | --- |
| 4 | Download and install the lineage harvester. | You use the lineage harvester to trigger the creation of Power BI assets, their relations and a technical lineage in Data Catalog.<br><br>We highly recommend that you always install and use the newest lineage harvester. You can download the lineage harvester from the Collibra Product Resource Downloads page. | • Your environment meets the system requirements to install and use the lineage harvester. |

| Step | What? | Description | Prerequisites |
|---|---|---|---|
| 5 | Prepare the lineage harvester configuration file and run the lineage harvester. | You create a lineage harvester configuration file with Power BI connection information and run the lineage harvester to import the results of the Power BI integration and the technical lineage for Power BI into Data Catalog.<br><br>As a result, Collibra creates new Power BI assets in Data Catalog and imports relations between these assets. It also creates a technical lineage for Power BI assets and other data sources in the lineage harvester configuration file.<br><br>Important If the `useCollibraSystemName` property in the lineage harvester configuration file is set to `true`, you also have to provide a <source ID> configuration file that defines the system name of databases in Power BI.<br><br>Tip For more information about the lineage harvester, see the Collibra Data Lineage documentation. | • You have downloaded lineage harvester version 2022.05 or newer. We highly recommend that you always install and use the newest lineage harvester.<br>• Your environment meets the system requirements to install and run the lineage harvester.<br>• You have prepared a lineage harvester configuration file.<br>• You have a global role with the Catalog global permission, for example Catalog Author.<br>• You have a global role with the Technical lineage global permission.<br>• You have a global role with the Data Stewardship Manager global permission.<br>• A resource role with |

| Step | What? | Description | Prerequisites |
|---|---|---|---|
| | | | the following resource permission on the community level in which you created the BI Data Catalog domain:<br>◦ Asset: add<br>◦ Attribute: add<br>◦ Domain: add<br>◦ Attachment: add |
| 6 | Prepare the Power BI <source ID> configuration file. | If the `useCollibraSystemName` property in the lineage harvester configuration file is set to `true`, you have to provide a <source ID> configuration file that defines the system name of databases in Power BI. Collibra Data Lineage uses the system names to match the structure of databases in Power BI to assets in Data Catalog. | You know the names or IDs of the capacities or workspaces you want to ingest. |

| Step | What? | Description | Prerequisites |
|---|---|---|---|
| 7 | Manually refresh your Power BI datasets. | Important Carry out this step only if this is the first time you're integrating Power BI in Data Catalog.<br><br>The first time you integrate Power BI, you need to make sure that the data in your Power BI datasets is up-to-date. After that, Microsoft automatically refreshes the datasets every 90 days.<br><br>For complete information, see:<br>• The Microsoft documentation.<br>• The Microsoft Power BI Blog. | See Power BI pre-requisites. |

| Step | What? | Description | Prerequisites |
|---|---|---|---|
| 8 | Run the lineage harvester again | **Important** Carry out this step only if this is the first time you're integrating Power BI in Data Catalog.<br><br>Start the lineage harvester again in the console and run the following command:<br>• for Windows: `.\bin\lineage-harvester.bat full-sync`<br>• for other operating systems: `./bin/lineage-harvester full-sync`<br><br>When prompted, enter the passwords to connect to your Collibra environment. The password is encrypted and stored in **/config/pwd.conf** | Same as for step 5. |

| Step | What? | Description | Prerequisites |
|---|---|---|---|
| 9 | View the Power BI assets and technical lineage | After the Power BI metadata is ingested in Data Catalog, you can go to the domain where you ingested Power BI and see the list of ingested Power BI assets. These assets are automatically stitched to existing assets in Data Catalog.<br><br>You can go to a Power BI Column asset page and click the Technical lineage lineage tab to view the technical lineage.<br><br>Note If you ingest Power BI for the first time or if you change your geolocation or cloud provider, you have to restart the DGC service before you can see your technical lineage.<br><br>Warning When you run the lineage harvester, Collibra Data Lineage creates all Power BI assets in the Data Catalog BI domain (or domains) you specified in the Power BI <source ID> configuration file. We highly recommend that you do not move these assets to other domains. If you move assets to other domains, they will be deleted and recreated in the initial Data Catalog BI domains when you | • Catalog Experience is enabled in Collibra Console.<br>• You have a Data Catalog global role with the Technical lineage global permission. |

| Step | What? | Description | Prerequisites |
|---|---|---|---|
| | | synchronize Power BI. As a result, all manually added characteristics of those assets are lost. | |

# Power BI ingestion considerations and limitations

There are a few considerations and limitations that you should be aware of when you use the Power BI metadata connector and lineage feature.

## General considerations

- The assets have the same names as their counterparts in Power BI. Full names and Display names cannot be changed in Data Catalog.
- Asset types are only created if you have all specific Power BI and Data Catalog permissions.
- The Power BI assets are created in the domain (or domains) that you specify in the Power BI <source ID> configuration file.
- Relations that were created between Power BI assets and other assets via a relation type in the Power BI operating model, are deleted upon synchronization. The same is true of any attribute types in the operating model that you add to Power BI assets. To ensure that the characteristics you add to Power BI assets are not deleted upon synchronization, be sure to use characteristics that are not part of the Power BI operating model.

## Supported subscriptions

You need one of the following subscriptions to ingest Power BI metadata in Data Catalog. The metadata collected by the lineage harvester is the same, regardless of your subscription.

- Power BI Pro.
- Power BI Premium.
- Power BI Premium Per User.

Other Power BI subscriptions are currently not supported.

# Power BI metadata

The lineage harvester can only partially access metadata of the following Power BI elements:

- Classic Power BI workspaces, which include My Workspace. Only a full ingestion of new Power BI workspaces is supported.
- Descriptions of most Power BI elements.
- Power BI apps are not ingested. They can, however, be ingested as Power BI Reports.

  > Note The prefix "[App]" in the name of a Power BI Report asset indicates that the report is distributed as part of an app, in Power BI. Direct links to such reports in Power BI don't work, therefore Power BI Report asset pages for such reports do not include the URL attribute.

The lineage harvester cannot access metadata of the following Power BI elements:

- Tile subtitles.
- Data from external sources supplying the input for the Power Query expressions in Power BI.

> Important The Collibra Data Lineage service can process most, but not all, complex Power BI metadata. This means that the success rate of a Power BI ingestion can be very high, but almost never 100%.

# Known issues

The following table presents the known issues of the Power BI integration in Collibra Data Intelligence Cloud.

| Known issue | Description |
| --- | --- |
| The data set *Report Usage Metrics Model* cannot be ingested. | The *Report Usage Metrics Model* is a data set that is automatically created by Power BI. This data set does not contain actual data, which means that they contain nothing to ingest into Data Catalog.<br><br>However, the lineage harvester still tries to access the metadata and, since there is nothing to access, shows an error message. All error messages about the *Report Usage Metrics* can be ignored. |
| Power BI assets that are moved to a different domain are deleted after synchronization. | Warning When you run the lineage harvester, Collibra Data Lineage creates all Power BI assets in the Data Catalog BI domain (or domains) you specified in the Power BI <source ID> configuration file. We highly recommend that you do not move these assets to other domains. If you move assets to other domains, they will be deleted and recreated in the initial Data Catalog BI domains when you synchronize Power BI. As a result, all manually added characteristics of those assets are lost. |
| You have successfully ingested Power BI metadata, but calculated tables and columns are not shown in the Technical lineage or in the browse tab pane. | Calculated columns are virtually the same as a non-calculated columns, with one exception: their values are calculated using DAX formulas and values from other columns. Collibra Data Lineage currently does not support internal transformations via DAX language, and any data objects derived via DAX are not shown in the technical lineage or in the browse tab pane. Currently, only M Query/Power Query expressions are supported. |

| Known issue | Description |
| --- | --- |
| You get an error message that mentions one of the following:<br>• "... function not implemented"<br>• "invalid lexical element" | This means that the specific integration feature is not currently supported.<br>Tip You can add your ideas for product enhancements and new features in the Collibra Integrations Ideation Portal. |

# Supported data sources in Power BI

Power BI is business intelligence software that can integrate with various data sources. When you ingest Power BI metadata, Collibra Data Lineage tries to automatically stitch this metadata to data sources registered in Data Catalog. It also creates a technical lineage that shows where metadata is used and how it transforms.

The following table shows the supported data source types in Power BI that have been tested.

Warning Although the following data sources have been tested extensively, there still may be some issues caused by unsupported elements within the data source or limitations in the Power BI integration process.

| Power BI data source | Connection type | Technical lineage | Stitching to registered data sources in Data Catalog |
| --- | --- | --- | --- |
| Amazon Redshift | Import | Yes | Yes |
| Azure Databricks | Import | Yes | Yes |

| Power BI data source | Connection type | Technical lineage | Stitching to registered data sources in Data Catalog |
|---|---|---|---|
| Google BigQuery | Import | Yes | Yes |
| ODBC | Import | Yes | Yes<br><br>Important You need to use a Power BI <source ID> configuration file to provide the true system names of the ODBC databases in Power BI. For more information, see Providing ODBC database names in Power BI. |
| Oracle | Import | Yes | Yes |
| Snowflake | Import | Yes | Yes |
| SQL Server | Import | Yes | Yes |
| Sybase | Import | Yes | Yes |

Note We cannot guarantee that other data sources in Power BI can be stitched successfully.

# Providing ODBC database names in Power BI

You can create a technical lineage for ODBC data sources in Power BI. However, ODBC database names often can't be determined. When a database name can't be determined, it's given a substitute name, which is the ODBC connection string.

This substitute name can be seen in the technical lineage, but it is merely a placeholder that doesn't carry any meaning if you're trying to identify the database it represents in the technical lineage. A bigger problem is that if you want to stitch the ODBC database to assets in Data Catalog, the substitute name won't match with any ingested databases, so stitching won't work.

To ensure that the true database names appear in the technical lineage, and to ensure successful stitching, you can use a Power BI <source ID> configuration file to provide the true system names of the ODBC databases in Power BI.

> Tip The name "<source ID>" refers to the value of the `sourceId` property in the lineage harvester configuration file. If, for example, the value of the `sourceId` property in the lineage harvester configuration file is `power-bi-source-1`, then the name of your <source ID> configuration file should be *power-bi-source-1.conf*.

## Example of the <source ID> configuration file

For each ODBC database in Power BI, add the following content to the JSON file:

```
{
   "found_dbname=DSN_MYDATABASE;found_hostname=ODBC": {
       "dbname": "DB001",
       "schema": "MYSCHEMA",
       "dialect": "oracle",
       "collibraSystemName": "oracle-system-name"
   }
}
```

| Property | Description |
|---|---|
| found_ dbname=<substitute database name>;found_ hostname=<server name> | `found_dbname` is the substitute database name. You need to convert it to uppercase and replace every non-alphanumeric character by an underscore (_). In this example, the substitute name is “dsn=MYDATABASE”, so you should use "DSN_MYDATABASE".<br><br>Note The substitute name is the ODBC connection string, which can be lengthy when it includes the driver and parameters in full.<br><br>`found_hostname` should be “ODBC”, but you can also use an asterisk (*). |
| dbname | The true system name of the ODBC database in Power BI. |
| schema | The name of the default schema of the ODBC database in Power BI.<br><br>If no schema is specified and the lineage harvester fails to find a specific schema, it uses the default schema. |
| dialect | The dialect of the ODBC connection.<br><br>The dialect must be one of the supported SQL dialects. If no dialect is specified, “mssql” is used, by default.<br><br>Tip<br>You can enter one of the following values:<br>• *azure*, for an Azure SQL Server data source.<br>• *bigquery*, for a Google BigQuery data source.<br>• *mssql*, for a Microsoft SQL Server data source.<br>• *oracle*, for an Oracle data source.<br>• *redshift*, for an Amazon Redshift data source.<br>• *snowflake*, for a Snowflake data source.<br>• *sybase*, for a Sybase data source. |

| Property | Description |
| --- | --- |
| collibraSystemName | The system or server name of a database.<br><br>Important Because you are using a <source ID> configuration file only for the purpose of providing the true system name of an ODBC database in Power BI, you are not required to:<br>• Set the useCollibraSystemName property in the lineage harvester configuration file to `true`.<br>• Specify a Collibra system name in the <source ID> configuration file.<br>However, if the useCollibraSystemName property is set to `true` in the lineage harvester configuration file, then you must specify a Collibra system name in the <source ID> configuration file. |

For complete information on working with <source ID> configuration files, see Power BI <source ID> configuration file.

# Supported Power Query M functions

Power Query is a data transformation and preparation engine. It uses a scripting language called Power Query M formula language–also known as M–for all transformations.

M is considered a "mashup" language. The Power Query engine filters and combines data from supported data sources. The "mashed up" data is then expressed using M. M is used by Power BI. It is not relevant to other integrations in Collibra.

The Collibra Data Lineage service perform lexical and syntax analysis of M. With regard to syntax analysis, the Collibra Data Lineage service instances currently support the following functions.

For complete information on these functions, see the Microsoft documentation on accessing data functions.

- **Backend-accessing data functions that impact the lineage diagram**
  - File.Contents
  - Web.Contents
  - Csv.Document
  - Excel.Workbook
  - Sql.Database
  - Sql.Databases
  - PostgreSQL.Database
  - Sybase.Database
  - Oracle.Database
  - AmazonRedshift.Database
  - GoogleBigQuery.Database
  - Snowflake.Database
  - Databricks.Contents
  - Odbc.Query
  - Odbc.DataSource
- **Transformations that impact the lineage diagram**
  - Replacer.ReplaceText
  - Replacer.ReplaceValue
  - Table.AddColumn
  - Table.AddIndexColumn
  - Table.DuplicateColumn
  - Table.ExpandTableColumn
  - Table.FromRecords
  - Table.FromRows
  - Table.NestedJoin
  - Table.PromoteHeaders
  - Table.RemoveColumns
  - Table.RenameColumns
  - Table.ReorderColumns
  - Table.ReplaceValue
  - Table.SelectColumns
  - Table.SplitColumn
  - Table.Unpivot
  - Table.UnpivotOtherColumns
  - Table.TransformColumnNames
- **Transformations that don't impact the lineage diagram**

- Table.AddKey
- Table.AlternateRows
- Table.Buffer
- Table.Distinct
- Table.ExpandListColumn
- Table.FillDown
- Table.FillUp
- Table.FindText
- Table.FirstN
- Table.InsertRows
- Table.IsEmpty
- Table.LastN
- Table.MaxN
- Table.MinN
- Table.Range
- Table.RemoveFirstN
- Table.RemoveLastN
- Table.RemoveMatchingRows
- Table.RemoveRows
- Table.RemoveRowsWithErrors
- Table.Repeat
- Table.ReplaceErrorValues
- Table.ReplaceKeys
- Table.ReplaceMatchingRows
- Table.ReplaceRows
- Table.ReverseRows
- Table.SelectRows
- Table.SelectRowsWithErrors
- Table.Skip
- Table.Sort
- Table.TransformColumns
- TableTransformColumnTypes
- Table.First
- Table.Last
- Table.Max
- Table.Min
- Table.SingleRow

- **Unsupported transformations**

  > Note Using unsupported transformations can cause parsing errors.

  - Table.FromRecords
  - SharePoint.Tables
  - Folder.Files
  - PowerBIRESTAPI.Navigation
  - DB2.Database
  - Table.ExpandRecordColumn
  - Table.Group

# Power BI prerequisites

Before you start the Power BI integration process, you have to perform a number of tasks in Power BI and Microsoft Azure. These tasks, which are performed outside of Collibra, are needed to enable the lineage harvester to reach your Power BI application and collect its metadata.

The tasks include the following:

- Attain authentication.
- Register your Power BI application in Microsoft Azure and set permissions.
- Fulfill the Power BI dedicated capacities and roles requirements for Power BI workspaces.

The metadata harvesting process explains in detail the prerequisites for enabling the lineage harvester to collect the Power BI metadata.

> Note There are some limitations to the metadata harvesting process. Ensure that you understand these limitations before you start the harvesting process.

> Warning Because these tasks are performed outside of Collibra, it is possible that the content changes without us knowing. We strongly recommend that you carefully read the source documentation.

# Supported Power BI subscriptions

You need one of the following subscriptions to ingest Power BI metadata in Data Catalog. The metadata collected by the lineage harvester is the same, regardless of your subscription.

- Power BI Pro.
- Power BI Premium.
- Power BI Premium Per User.

> Tip We highly recommend you to have a Power BI Premium subscription.

# Authentication

You have to attain authentication to access Power BI metadata. Your authentication method determines how you retrieve the metadata. The lineage harvester supports two authentication methods:

- Username and password
- Service principal

The metadata harvesting process is different for each authentication method. Therefore, different configurations in Microsoft Azure and Power BI are required.

> Note We highly recommend that you use the service principal authentication, as detailed metadata scanning in Power BI is designed for use with service principal authentication.

Tip
You can use a cURL command to check whether or not you can use username and password authentication.

**Show me how**
Run the following command, where the bolded text refers to your information:

```
curl -v "https://login.microsoftonline.com/<your environment>.onmicrosoft.com/oauth2/v2.0/token" -F client_id=<your ID> -F "username=<your username>" -F "password=<your password>" -F "scope=https://analysis.windows.net/powerbi/api/.default" -F grant_type=password
```

To check on Windows, follow these steps:

1. Download and install the cURL Command-Line Tool.
2. In Windows, click **Start** > **Run**, and then enter *cmd* in the **Run** dialog box.
3. Run the following command, where the bolded text refers to your information:

```
"https://login.microsoftonline.com/<your environment>.onmicrosoft.com/oauth2/v2.0/token" -F client_id=<your ID> -F "username=<your username>" -F "password=<your password>" -F "scope=https://analysis.windows.net/powerbi/api/.default" -F grant_type=password
```

Note  To ingest Power BI dataflows:

- You need access to the Power BI environment in which the data flow is stored.
- The data set in the data flow must exist in a premium workspace.

## Username and password

The username and password authentication method relies on the username, in the form of an email address, and a password you provide to access the Power BI metadata.

To use the username and password authentication method, you need to be an Azure Active Directory user with a Power BI admin role in Power BI.

When you become an Azure Active Directory user, a new email address is created. This email address is the username you use to sign in to Power BI. You can store the username and password you use to sign in to Power BI in the Technical lineage configuration file.

> Note Only Azure Administrators can create users and require them to authenticate via username and password. The Azure Administrator also assigns the user the Power BI admin role. This user is only created for the purpose of Power BI integration in Collibra Data Intelligence Cloud. The user in Azure should have a Member user type.

## Service principal

The service principal authentication method lets an Azure Active Directory automatically access Power BI.

Service principal authentication relies on the Power BI Tenant ID and the Azure Active Directory application ID that you provide in the lineage harvester configuration file. The password you need to access Power BI is the client secret key of the Azure Active Directory application.

To use service principal authentication, you need to embed Power BI content with a Service Principal and an application secret. This entails the following steps:

- In the Power BI Admin portal:
  - Enable the **Allow service principals to use read-only Power BI admin APIs** option.
  - Enable the **Allow service principal to use Power BI APIs** option in the Developer settings.
  - Enable the **Enhance admin APIs responses with detailed metadata** option.
  - Enable the **Enhance admin APIs responses with DAX and mashup expressions** option.

  > Note You need Power BI administrator rights to access the Power BI Admin portal.

> Tip Do not confuse the **Allow service principals to use read-only Power BI admin APIs** option with the **Allow service principal to use Power BI APIs** option. You need to enable both options.

# Register Power BI in Microsoft Azure and set permissions

Before you set up the lineage harvester, make sure that the harvester can reach Power BI by registering Power BI in Azure and setting the necessary permission to harvest the metadata.

We highly recommend that you read about supported authentication methods before you register Power BI in Microsoft Azure.

> Warning This procedure is performed outside of Collibra. A third-party might change the software without notification, which can render this documentation out-of-date. We highly recommend that you carefully read the source documentation.

## Steps

> Tip The content in this topic is different for the username / password authentication method or service principal authentication method. We highly recommend that you read the following instructions carefully before you register Power BI in Microsoft Azure:
>
> - Service principal instructions
> - Username / password instructions

1. Register Power BI in the Azure Portal using the following settings:

| Setting | Description |
|---|---|
| Name | The name of your Power BI application. |
| Supported account types | The type of tenant. This indicates who can access the Power BI application.<br><br>In this case, the supported account type must be *Single tenant*. |
| Redirect URI | The location to which a user's client is redirected and where security tokens are sent after a successful authorization.<br><br>In this case, the redirected URI must be *Web*, but you do not have to specify any web location. |

» When you have registered Power BI, the Azure portal creates two important IDs that you need in the Technical lineage configuration file:
   - The Application (client) ID
   - The Directory (tenant) ID

> Note We highly recommend that you store these IDs for further use. You can find the IDs in the **Overview** pane on the Azure portal or in the top right menu.

2. Create a user with the Power BI Administrator role (only for username / password authentication).
   a. Grant the Power BI application in Microsoft Azure administrator rights (such as Office 365 Global Administrator or Power BI Service Administrator). (only for username / password authentication)

      > Note Delegated permissions are supported.

3. In the Azure portal, go to the **Authentication** pane and do the following:
   a. Go to the **Advanced settings** section.
   b. Set the **Treat application as a public client** to *Yes*.

> Note Ensure that the admin consent workflow is not enabled for this application. For more information, see Configure the admin consent workflow.

4. Go to the **API permissions** pane and do the following:
   a. Select **Delegated permissions** as permission type.
   b. Grant the Power BI application in Microsoft Azure the Microsoft Graph `User.Read` permission.
   c. Grant the Power BI application in Microsoft Azure all Power BI Service permissions (only for username / password authentication).
   d. Set **Admin consent required** for `Tenant.ReadAll` permission to *Yes* (only for username / password authentication).

   » The user now has the following permissions:
   - Microsoft Graph
     - `User.Read`
   - Power BI Service (only for username / password authentication)
     - `App.Read.All`
     - `Capacity.Read.All`
     - `Dashboard.Read.All`
     - `Dataflow.Read.All`
     - `Group.Read.All`
     - `Report.Read.All`
     - `Workspace.Read.All`
     - `Tenant.Read.All`, with **Admin consent required** set to `Yes`.
   - Power BI Service (only for username / password authentication)
     - `Tenant.Read.All` or `Tenant.ReadWrite.All`, with **Admin consent required** set to `No`.
5. In the Power BI Admin portal, do the following (only for service principal authentication):
   a. Enable the **Allow service principals to use read-only Power BI admin APIs** option.
   b. Enable the **Allow service principal to use Power BI APIs** option in the Developer settings.
   c. Enable the **Enhance admin APIs responses with detailed metadata** option.
   d. Enable the **Enhance admin APIs responses with DAX and mashup expressions** option.

e. Apply the option to specific security groups.
f. Enter the name of the security group to which you want to add the service principal.

> Warning The Power BI APIs do not support mail-enabled security groups.

> Note You need Power BI administrator rights to access the Power BI Admin portal.

6. In the Power BI Admin portal, do the following(Only for username / password authentication):

> Note Apply the integration setting to the entire organization (default) or to the specific security group to which your workspaces belong.

a. Enable the **Enhance admin APIs responses with detailed metadata** option.
b. Enable the **Enhance admin APIs responses with DAX and mashup expressions** option.

# Add the Power BI certificate to the Java TrustStore

Before you integrate Power BI in Data Catalog, ensure that the Power BI certificate exists in the Java TrustStore and that's it's in the correct directory. If the certificate doesn't exist, you need to generate one.

## Check if the Power BI certificate exists

Run the following command:

```
keytool.exe -list -keystore %JAVA_HOME%\jre\lib\cacerts
```

> Important Ensure that the certificate is in the correct directory: `%JAVA_HOME\jre\lib\cacerts`

If the certificate doesn't exist, you need to generate one.

# Generate a certificate and add it to the Java TrustStore

## On Windows

> Note In the following example commands, we refer to the techlin-gcp-us instance. You should refer to the correct Collibra Data Lineage service instance in the geographic location of your Collibra Data Intelligence Cloud environment.

1. Run the following command to extract the certificate from the Power BI Service:
   ```
   keytool -printcert -rfc -sslserver techlin-gcp-us.-
   collibra.com:443 > powerbi-cert.crt
   ```
   > Tip Replace the URL techlin-gcp-us.collibra.com with the URL for your Power BI server, which you specify in the lineage harvester configuration file. This will create a file named *powerbi-cert.crt* in the folder where you run this command.

2. Run the following command to find the location of your JAVA_HOME:
   ```
   echo %JAVA_HOME%
   ```
   » The location path will be something like the following: `C:\Program Files\Java\jdk-17.0.2`
3. Use the location path of your JAVA_HOME in the following command, to import the *powerbi-cert.crt* file into the cacerts file found above.
   ```
   keytool -importcert -file powerbi-cert.crt -alias "Power-
   BIProdServerCert" -keystore "C:\Program Files\Java\jdk-
   17.0.2\cacerts"
   ```
   > Note You can specify a different alias, if you want.

4. Run the following command:
   ```
   keytool -list -keystore "C:\Program Files\Java\jdk-17.0.2\lib\se-
   curity\cacerts" | findstr "PowerBI"
   ```
5. Enter the keystore password.
   > Tip The password is typically `changeit`.

   » A list of all certificates that match the PowerBI string in the "C:\Program

Files\Java\jdk-17.0.2\cacerts" file is shown.

> Tip In the list of certificates, look for the one that you imported in step 3. If it's listed, it means the "C:\Program Files\Java\jdk-17.0.2\cacerts" file has the certificate needed to validate the PowerBI server.

6. Run the following command to have the lineage harvester use the cacerts file that you just updated.

```
set JAVA_OPTS=-Djavax.net.ssl.trustStore="C:\Program
Files\Java\jdk-17.0.2\lib\security\cacerts" -
Djavax.net.ssl.trustStorePassword="changeit"
```

### On Linux

> Note In the following example commands, we refer to the techlin-gcp-us instance. You should refer to the correct Collibra Data Lineage service instance in the geographic location of your Collibra Data Intelligence Cloud environment.

1. Run the following command to get a certificate from the corresponding techlin-gcp-us.com site, which is part of the CollibraData Lineage infrastructure:

```
openssl x509 -in <(openssl s_client -connect techlin-gcp-
us.collibra.com:443 -prexit 2>/dev/null) -out techlin-gcp-us.crt
```

> Tip If you already have a correctly formatted certificate on the server, you can skip this step.

2. Add the certificate to the Java TrustStore:

```
keytool -importcert -file techlin-gcp-us.crt -alias techlin-gcp-
us -keystore %JAVA_HOME%\jre\lib\cacerts -storepass changeit
```

## Power BI workspaces

Power BI workspaces represent the most used metadata in Power BI. It contains for example reports and data sets. If you want a full ingestion, you have to make sure that the lineage harvester can access all metadata in your Power BI workspaces. Consider the following:

- Depending on the authentication type, you must have specific roles and permissions to access the metadata in the Power BI workspaces.
- You can only fully ingest new Power BI workspaces. This means that classic workspaces and My Workspace in Power BI are not supported.

> Tip Use the Power BI <source ID>_filter configuration file to filter on Power BI workspaces.

> Note To ingest Power BI dataflows:
>
> - You need access to the Power BI environment in which the data flow is stored.
> - The data set in the data flow must exist in a premium workspace.

## Filtering Power BI workspaces

Filtering, in this context, means specifying from which Power BI workspaces and/or capacities you want to ingest metadata. Filtering allows you to ingest only the metadata that matters most to you.

You use the filters section in the Power BI <source ID> configuration file to configure workspace filtering. You can filter on workspace names, workspace IDs, capacity names or capacity IDs.

The lineage harvester accesses the metadata of all Power BI workspaces. Filtering then determines which metadata is sent from the Collibra Data Lineage service to Data Catalog. This means that if you don't use filtering, all workspaces are ingested in Collibra.

> Tip In the Power BI <source ID> configuration file, you can also specify the domain (or domains) in which you want to ingest, to help structure your Power BI assets in Collibra.

### Filtering best practice

You can filter on a capacity to ingest the metadata from all workspaces in that capacity. Let's say, for example, that you have 50,000 workspaces but you only want to ingest metadata from the workspaces related to a specific department in your organization. You

could specify each of the relevant workspaces in the configuration file, but that could be tedious if there are lots of workspaces. Furthermore, if someone in your organization creates a new workspace, it will have to be added to your configuration file. Instead, you can filter on a capacity. Then, when a new workspace is created, ensure that it is added to the department' s capacity and metadata from that workspace will be automatically ingested, without having to update the configuration file.

## Workspace states

On Power BI Workspace asset pages, you can include the attribute type State, to show the state of ingested Power BI workspaces, for example Active, Orphaned or Deleted. To do so, you have to edit the global assignment of the Power BI Workspace asset type and assign the attribute type State.

For complete information on Power BI workspaces and possible states, see the Microsoft Power BI documentation.

> Tip If you only want to see Power BI workspaces that have the state Active:
>
> 1. Ensure that the attribute type State is assigned to the Power BI Workspace asset type via the global assignment.
> 2. Go to the Global view, and then create an advance filter and filter by the following clauses:
>    a. Asset type equals Power BI Workspace
>    b. Characteristic State equals Active.

## Deleted workspaces

If you delete a Power BI workspace, the workspace is maintained for a 90-day grace period, during which a Power BI administrator can restore the workspace. During the grace period, the workspace has the state Deleted. When you ingest Power BI metadata in Data Catalog, this deleted workspace is ingested.

When the grace period elapses, the state of the workspace becomes Removing, for a short time, while it is being permanently removed. The state then becomes Not found. At this point, as the workspace no longer exists in Power BI, the Power BI Workspace asset in Collibra will also be deleted upon the next synchronization.

## Why are deleted workspaces ingested?

Let's image that you ingest a Power BI workspace with the Active state and that over time, you add comments, tags and characteristics to the asset in Collibra. Now let's imagine that the workspace is deleted in Power BI and we do not ingest the deleted workspace. In this case, the Power BI Workspace asset in Collibra is deleted upon the next synchronization. But what if the Power BI administrator decides, during the 90-day grace period, to restore the workspace in Power BI? Upon the next synchronization, a new Power BI Workspace asset is created in Collibra, but all of the comments, tags and characteristics that were part of the deleted asset are lost.

By ingesting deleted Power BI workspaces, we safeguard against losing any of the additional information on the Power BI Workspace asset, in case a Power BI administrator decides to restore a workspace during the grace period.

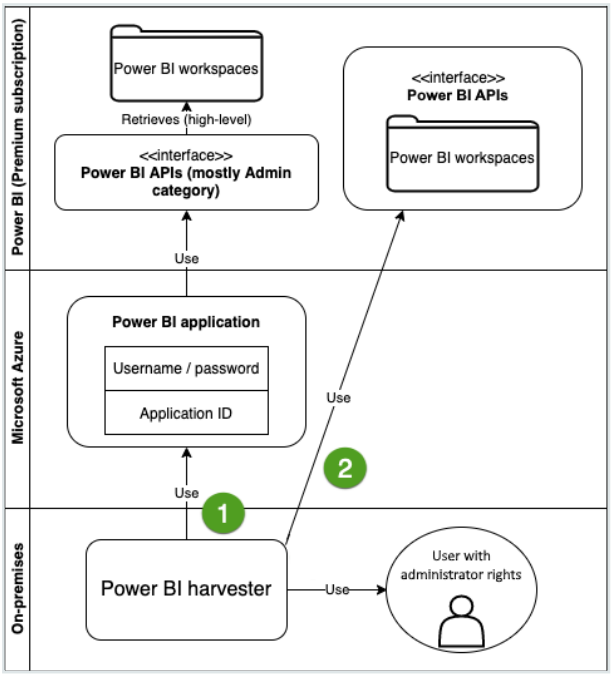
# The metadata harvesting process

Collibra uses Power BI REST APIs to harvest Power BI metadata.

To enable the lineage harvester to access metadata in Power BI workspaces, you must have the correct configurations in Microsoft Azure.

> Note There are some limitations to the metadata harvesting process. Ensure that you understand these limitations before you start the harvesting process.

# Overview of the metadata harvesting process with username / password authentication

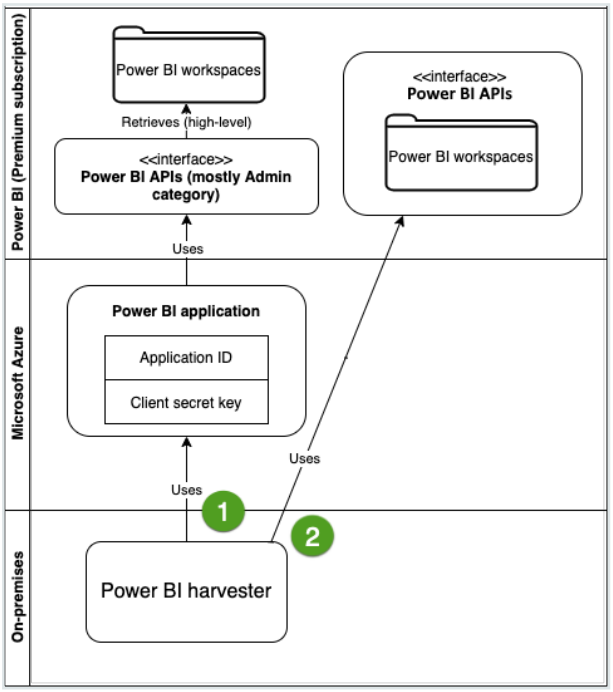



| Step | Description |
|---|---|
| 1 | The lineage harvester uses the username, password and application ID to access the Power BI APIs. These APIs retrieve basic Power BI metadata, for example metadata in the Power BI tenant or server and reports. |
| 2 | The lineage harvester uses Power BI API calls to retrieve more specific metadata, for example Power BI columns and lineage. |

Important The Power BI application in Microsoft Azure must be granted administrator rights, such as Office 365 Global Administrator or Power BI Service Administrator. Delegated permissions are supported.

Note The lineage harvester accesses the metadata of all Power BI workspaces. If you don't use filtering, all workspaces are ingested in Collibra. We recommend that you use filtering and domain mapping to structure your Power BI assets in Collibra.

# Overview of the metadata harvesting process with service principal authentication




| Step | Description |
|---|---|
| 1 | The lineage harvester uses the application ID and the client secret key of the Azure Active Directory application to access the Power BI APIs. These APIs retrieve basic Power BI metadata, for example metadata in the Power BI tenant or server and reports. |
| 2 | The lineage harvester uses Power BI API calls to retrieve more specific metadata, for example Power BI columns and lineage. |

> Note The lineage harvester accesses the metadata of all Power BI workspaces. If you don't use filtering, all workspaces are ingested in Collibra. We recommend that you use filtering and domain mapping to structure your Power BI assets in Collibra.

# Prepare a domain for Power BI ingestion

Before you ingest Power BI metadata, you have to designate a domain for storing the new Power BI assets. You can choose an existing domain or create one or more new domains. You will include the domain reference ID (or IDs) in the lineage harvester configuration file. Collibra uses the domains to ingest all Power BI assets during the Power BI integration process.

The following procedure guides you through the creation of one or more domains.

## Prerequisites

You have a resource role with the Domain > Add resource permission.

## Steps

1. In the main menu, click the **Create** (+) button.
   » The **Create** dialog box appears.
2. Click the **Organization** tab.
3. Click a domain type from the list.
   If you clicked the wrong domain type here, you can change it in the **Type** field in the next screen.
   » The **Create Domain** dialog box appears.
4. Enter the required information.

| Field | Description |
|---|---|
| Type | The domain type of the domain you are creating. In this case, you need to select *BI Catalog*. |
| Community | The community under which the domain will be located. |

| Field | Description |
| --- | --- |
| Name | The name of the new domain or domains.<br><br>Tip<br>You can create multiple domains in one go.<br>To do this, press `Enter` after typing a value and then type the next. Domain names have to be unique in their parent community. If you type a name that already exists, it will appear in strike-through style. |

5. Click **Create**.
6. Open your domain. If you created multiple domains, open each of them in turn.
7. Copy the reference ID of each domain you created.

   Tip If you go to your domain, you can find the reference ID in the URL. The URL looks like: https://<yourcollibrainstance>/domain/**22258f64-40b6-4b16-9c08-c95f8ec0da26**?view=00000000-0000-0000-0000-000000040001. In this example, the reference ID is in bold.

8. Paste a reference ID in the `domainId` property in your lineage harvester configuration file. This is your default domain.
9. Optionally, if you want to ingest the contents of specific workspaces into specific domains in Collibra, paste the relevant domain reference IDs in the `filters` section of your Power BI <source ID> configuration file.

Warning When you run the lineage harvester, Collibra Data Lineage creates all Power BI assets in the Data Catalog BI domain (or domains) you specified in the Power BI <source ID> configuration file. We highly recommend that you do not move these assets to other domains. If you move assets to other domains, they will be deleted and recreated in the initial Data Catalog BI domains when you synchronize Power BI. As a result, all manually added characteristics of those assets are lost.

# Collibra Data Lineage service instances

The Collibra Data Lineage service processes and analyzes the harvested metadata and uploads it to Data Catalog. The Collibra Data Lineage service never processes actual data.

Based on your geographical location and cloud provider, the lineage harvester sends metadata to one of the following Collibra Data Lineage service instances:

- 15.222.200.199 (techlin-aws-ca.collibra.com)
- 18.198.89.106 (techlin-aws-eu.collibra.com)
- 54.242.194.190 (techlin-aws-us.collibra.com)
- 51.105.241.132 (techlin-azure-eu.collibra.com)
- 20.102.44.39 (techlin-azure-us.collibra.com)
- 35.197.182.41 (techlin-gcp-au.collibra.com)
- 34.152.20.240 (techlin-gcp-ca.collibra.com)
- 35.205.146.124 (techlin-gcp-eu.collibra.com)
- 34.87.122.60 (techlin-gcp-sg.collibra.com)
- 35.234.130.150 (techlin-gcp-uk.collibra.com)
- 34.73.33.120 (techlin-gcp-us.collibra.com)

> Important You have to whitelist all Collibra Data Lineage service instances in your geographic location. For example, if your data is located in Europe, you have to whitelist the following Collibra Data Lineage service instances: techlin-aws-eu and techlin-gcp-eu. In addition, we highly recommend that you always whitelist the techlin-aws-us instances as a backup, in case the lineage harvester cannot connect to other Collibra Data Lineage service instances.

# Set up the lineage harvester for Power BI ingestion

The lineage harvester is a software application that is needed to collect your Power BI metadata and send it to the Collibra Data Lineage service, where the metadata is processed and a technical lineage and new Power BI assets and relations are created. Collibra Data Intelligence Cloud then import those assets and relations into Data Catalog.

For more information about the lineage harvester, read the Collibra Data Lineage documentation.

> Note You need the lineage harvester 2022.05 or newer to ingest Power BI metadata into Data Catalog.

# Lineage harvester system requirements

You need to meet the system requirements to be able to install and run the lineage harvester.

## Software requirements

Java Runtime Environment version 11 or newer, or OpenJDK 11 or newer.

For Java Runtime Environment 16 or newer, or OpenJDK 16 or newer, set the `JAVA_OPTS` environment variable for the lineage harvester to function properly:

```
JAVA_OPTS='--illegal-access=deny'
```

> Note To ingest Snowflake data sources, the minimum requirement is Java Runtime Environment version 16 or newer, or OpenJDK 16 or newer.

## Hardware requirements

You need to meet the hardware requirements to install and run the lineage harvester.

### Minimum hardware requirements

You need the following minimum hardware requirements:

- 2 GB RAM
- 1 GB free disk space

## Recommended hardware requirements

The minimum requirements are most likely insufficient for production environments. We recommend the following hardware requirements:

- 4 GB RAM

  > Tip 4 GB RAM is sufficient in most cases, but more memory could be needed for larger harvesting tasks. For instructions on how to increase the maximum heap size, see Technical lineage general troubleshooting.

- 20 GB free disk space

> Important If you have more than 50,000 actively used workspaces, we recommend increasing the heap size to at least 6 GB.

## Certificate requirements

Ensure that the Power BI certificate exists in the Java TrustStore. For complete information, see Add the Power BI certificate to the Java TrustStore.

## Network requirements

The lineage harvester uses the HTTPS protocol by default and uses port 443.

You need the following minimum network requirements:

- Firewall rules so that the lineage harvester can connect to:
  - Collibra Data Intelligence Cloud.
  - All Collibra Data Lineage service instances in your geographic location:
    - 15.222.200.199 (techlin-aws-ca.collibra.com)
    - 18.198.89.106 (techlin-aws-eu.collibra.com)
    - 54.242.194.190 (techlin-aws-us.collibra.com)
    - 51.105.241.132 (techlin-azure-eu.collibra.com)
    - 20.102.44.39 (techlin-azure-us.collibra.com)
    - 35.197.182.41 (techlin-gcp-au.collibra.com)
    - 34.152.20.240 (techlin-gcp-ca.collibra.com)
    - 35.205.146.124 (techlin-gcp-eu.collibra.com)

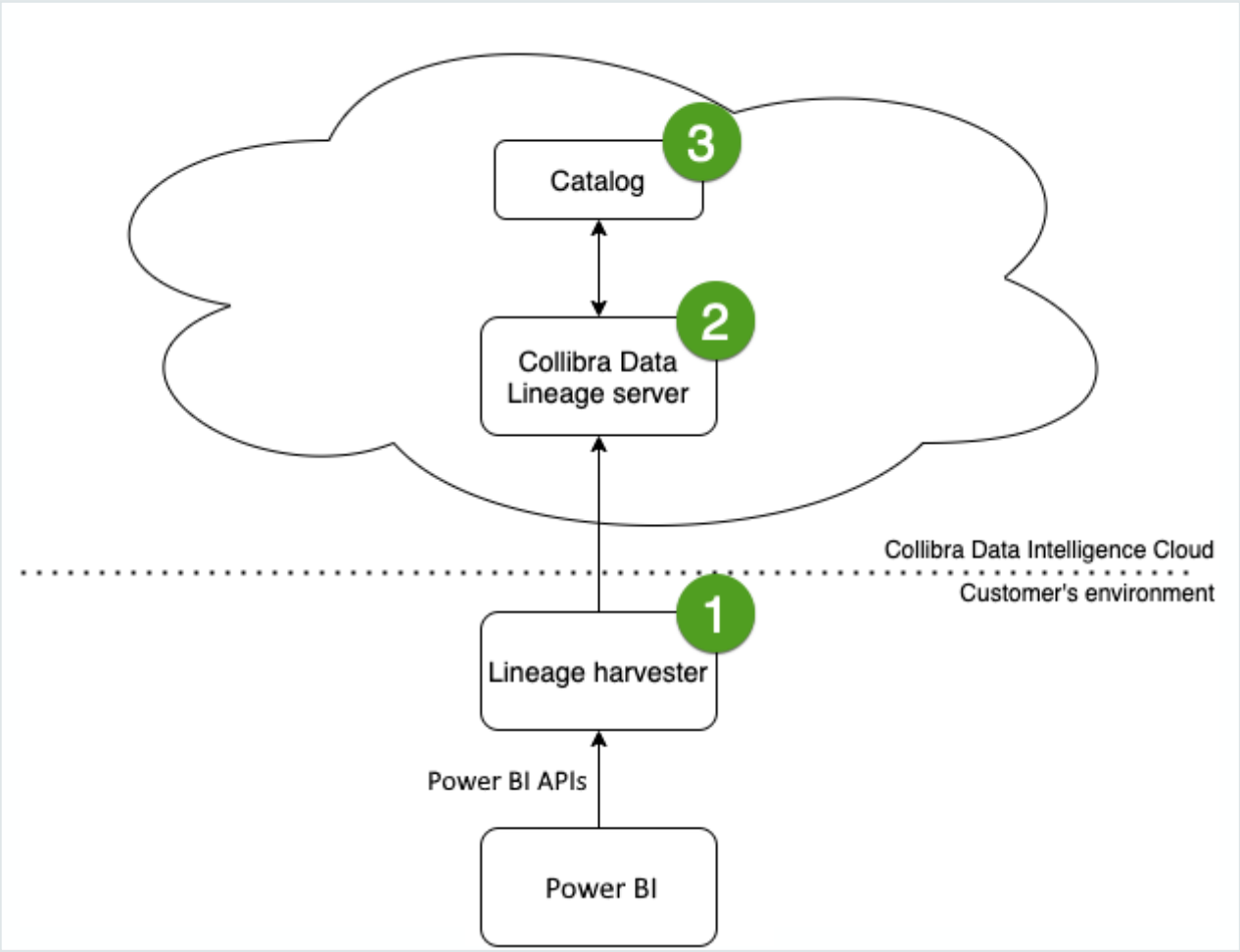
- 34.87.122.60 (techlin-gcp-sg.collibra.com)
- 35.234.130.150 (techlin-gcp-uk.collibra.com)
- 34.73.33.120 (techlin-gcp-us.collibra.com)

> Note The Power BI harvester connects to different Collibra Data Lineage service instances based on your geographic location and cloud provider. If your location or cloud provider changes, the Power BI harvester rescans all your Power BI metadata. You have to whitelist all Collibra Data Lineage service instances in your geographic location. In addition, we highly recommend that you always whitelist the techlin-aws-us instance as a backup, in case the Power BI harvester cannot connect to other Collibra Data Lineage service instances.

# Power BI ingestion workflow

You run the lineage harvester to start the Power BI ingestion workflow. When you initiate Power BI ingestion, each workflow component performs the following actions:

1. The lineage harvester:
   - Communicates with Power BI.
   - Harvests the Power BI metadata that will be ingested to Data Catalog.
   - Sends the Power BI metadata to the Collibra Data Lineage service.
2. The Collibra Data Lineage service:
   - Analyzes the Power BI metadata.
   - Creates new assets and relations.
   - Stitches existing assets in Data Catalog to Power BI assets.
   - Imports new Power BI assets and their relations in Data Catalog.
3. Data Catalog:
   - Shows the new Power BI assets.
   - Shows a Technical lineage tab on Power BI Column pages.
   - Shows stitching results between Power BI Column assets and Column assets.




# Install the lineage harvester for Power BI integration

Before you can use the lineage harvester, you need to download it and install it. You can download the lineage harvester from the Collibra Community downloads page.

Tip
- Install the lineage harvester close to your data source or on the same server.
- The lineage harvester uses port 443.

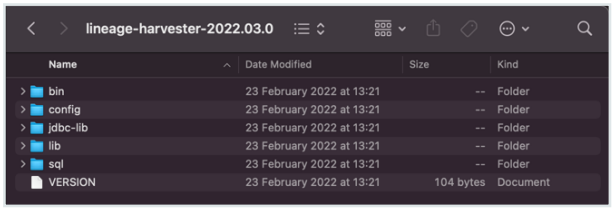
# Prerequisites

- You have purchased the Power BI metadata connector and lineage feature.
- You have Collibra Data Intelligence Cloud 2020.11 or newer.
- You meet the minimum system requirements.
- You have added Firewall rules so that the lineage harvester can connect to:
  - The host names of all databases in the lineage harvester configuration file.
  - All Collibra Data Lineage service instances within your geographical location:
    - 15.222.200.199 (techlin-aws-ca.collibra.com)
    - 18.198.89.106 (techlin-aws-eu.collibra.com)
    - 54.242.194.190 (techlin-aws-us.collibra.com)
    - 51.105.241.132 (techlin-azure-eu.collibra.com)
    - 20.102.44.39 (techlin-azure-us.collibra.com)
    - 35.197.182.41 (techlin-gcp-au.collibra.com)
    - 34.152.20.240 (techlin-gcp-ca.collibra.com)
    - 35.205.146.124 (techlin-gcp-eu.collibra.com)
    - 34.87.122.60 (techlin-gcp-sg.collibra.com)
    - 35.234.130.150 (techlin-gcp-uk.collibra.com)
    - 34.73.33.120 (techlin-gcp-us.collibra.com)

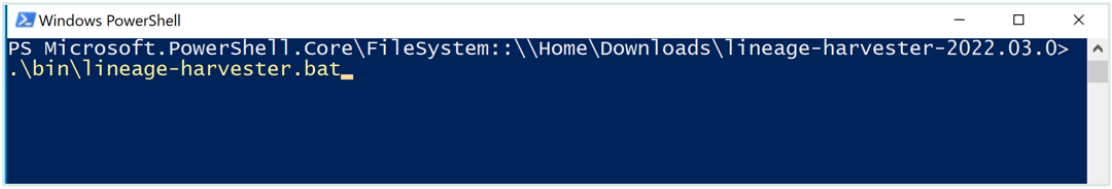
> Note The lineage harvester connects to different instances based on your geographic location and cloud provider. If your location or cloud provider changes, the lineage harvester rescans all your data sources. You have to whitelist all Collibra Data Lineage service instances in your geographic location. In addition, we highly recommend that you always whitelist the techlin-aws-us instance as a backup, in case the lineage harvester cannot connect to other Collibra Data Lineage service instances.

# Steps

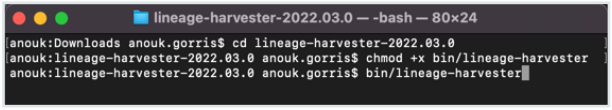
1. Download the newest lineage harvester.
2. Unzip the archive.
   » You can now access the lineage harvester folder.



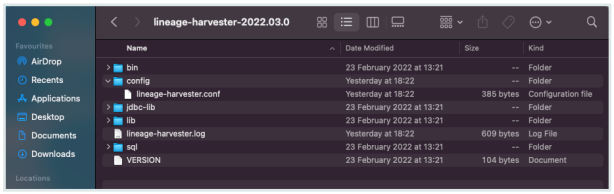

3. Run the following command line to start the lineage harvester:
   - Windows: `.\bin\lineage-harvester.bat`




   - For other operating systems: `chmod +x bin/lineage-harvester` and then `bin/lineage-harvester`




   » An empty configuration file is created in the config folder.




   » The lineage harvester is installed automatically. You can check the installation by running `./bin/lineage-harvester --help`.

# What's next?

You can now prepare the lineage harvester configuration file.

# Prepare the lineage harvester configuration file for Power BI

You have to prepare a technical lineage configuration file and run the lineage harvester to retrieve metadata from Power BI and send it to the Collibra Data Lineage service to be scanned, processed and analyzed.

> Tip For more information, see Collibra Data Lineage.

## Prerequisites

Ensure that you have completed the following tasks:

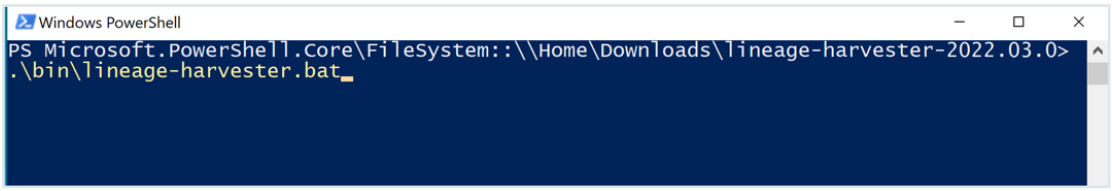
- Installed and set up the latest lineage harvester.
- Created one or more BI Catalog domains in which you want to ingest the Power BI assets.
- Manually refreshed your datasets in Power BI.

Ensure that you meet the following requirements and have the following permissions:

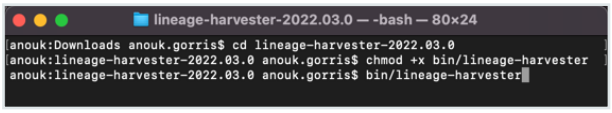
- Use Collibra Data Intelligence Cloud.
- A global role with the following global permissions:
  - Catalog, for example Catalog Author
  - Data Stewardship Manager
  - Manage all resources
  - System administration
  - Technical lineage
- A resource role with the following resource permission on the community level in which you created the BI Catalog domain:
  - Asset: add
  - Attribute: add
  - Domain: add
  - Attachment: add

# Steps

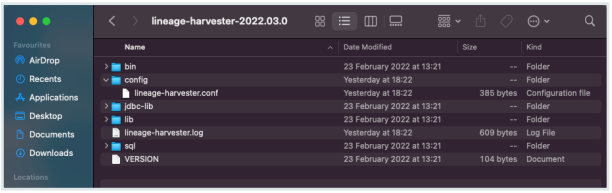
1. Run the following command line to start the lineage harvester:
   - Windows: `.\bin\lineage-harvester.bat`

   - For other operating systems: `chmod +x bin/lineage-harvester` and then `bin/lineage-harvester`

   » An empty configuration file is created in the config folder.

2. Open the configuration file and enter the values for each property.
   **Watch a video on how to do this**

| Properties | Description |
| --- | --- |
| general | This section describes the connection information between the lineage harvester and Data Catalog. |
| catalog | This section contains information that is necessary to connect to Data Catalog. |

| Properties | Description |
| --- | --- |
| url | The URL of your Collibra Data Intelligence Cloud environment.<br><br>> Note You can only enter the public URL of your Collibra DGC environment. Other URLs are not accepted. |
| username | The username that you use to sign in to Collibra. |

| Properties | Description |
| --- | --- |
| useCollibraSystemName | Indicates whether or not you want to use the system or server name of a data source to match to the System asset in Data Catalog during automatic stitching. This is useful when you have multiple databases with the same name or if you want to specify the Power BI workspaces from which you want to ingest.<br><br>By default, the useCollibraSystemName property is set to `False`. If you want to use it, set it to `True`.<br><br>Important<br>◦ If you set this property to `true`:<br>▪ You must provide a Power BI <source ID> configuration file that defines the system name of databases in Power BI.<br>▪ The lineage harvester reads the value of the `collibraSystemName` property in the <source-ID> configuration file.<br>◦ If you set the `useCollibraSystemName` property to `false`, the lineage harvester ignores the `collibraSystemName` property in the <source-ID> configuration file. |

| Properties | Description |
| --- | --- |
| useSharedDbModel | Optional property to enable the sharing of metadata batches from multiple SQL data sources. Set this property to `true`, to help avoid potential analysis errors on the Collibra Data Lineage service.<br><br>To use this property, you need lineage harvester 2022.07 or newer.<br><br>If you set this property to `true`, you have to run the lineage harvester twice. Read the following details about the issue and solution.<br><br>**See details about the issue and solution**<br>Normally, when you run the lineage harvester to harvest metadata from two or more data sources, the metadata from each source is processed independently. This means that the metadata from one data source cannot access the metadata of another.<br><br>Let's say, for example, you specify the following two SQL data sources in your lineage harvester configuration file:<br>◦ A database source that retrieves the database model.<br>◦ An SqlDirectory source with Data Manipulation Language (DML) statements that reference data in the database source.<br><br>Because these data sources are processed independently, there is a good |

| Properties | Description |
| --- | --- |
| | chance that the DML statements will fail during analysis. Any wildcards in the DML statements, for example, would fail because the SqlDirectory source can't access the referenced database source.<br><br>**The solution**<br><br>The shared database model allows for computed results from a "main" batch. Although multiple data sources are still processed independently, the metadata from each data source is merged into a main batch. Then, before analyzing the next batch, a check is done to see if a preceding main batch exists. If one does, the analyzer retrieves the database model and the DML statements successfully pass analysis.<br><br>This means, however, that you have to run the lineage harvester twice. On the first run, the harvested metadata is merged in a main batch. Then, when you run the lineage harvester again, using the full-sync command, the subsequent batches are able to successfully reference the metadata in the main batch.<br><br>In a future version of Collibra, this property will be enabled by default and you won't need to run the lineage harvester twice. |

| Properties | Description |
| --- | --- |
| sources | This section describes the data sources for which you want to create the technical lineage. You have to create a configuration section for each data source.<br><br>Note You can add multiple data sources to the same configuration file. |
| type | The kind of data source. In this case, the value has to be *PowerBI*. |
| id | The unique ID to identify the Power BI service metadata that was uploaded to the Collibra Data Lineage service.<br><br>Warning In the `sources` section of your lineage harvester configuration file, you can only specify one `id` property per Power BI service. If you have multiple `id` properties for a single Power BI service, ingestion will fail. If you have multiple `id` properties in the configuration file, it means you intend to ingest from multiple unique Power BI services. |

| Properties | Description |
| --- | --- |
| tenantDomain | The Power BI tenant domain is the domain associated with the Microsoft Azure tenant.<br><br>This domain is either a default domain or a custom domain. For example, *collibrapowerbi.onmicrosoft.com*.<br><br>> Note Usually, you can find a list of Power BI tenant or server domains in your Azure Active Directory or in the top right menu. |
| loginFlow | This section describes the authentication information for accessing your Power BI metadata.<br><br>The lineage harvester supports two authentication methods: service principal, and username and password. For complete information on your authentication options, see Authentication. |
| type | This depends on the authentication method you use.<br><br>◦ Service principle: The value should be `ServicePrincipal`.<br>◦ Username and password: The value should be `ResourceOwnerPasswordCredentials`. |

| Properties | Description |
| --- | --- |
| applicationId | The unique ID of the Microsoft Azure Application (client) ID. |
| username | The email address of your Azure Active Directory user.<br><br>Tip This property only applies if you are using the username and password authentication method. |
| domainId | The reference ID of the domain in Collibra in which you want to ingest Power BI metadata. |

| Properties | Description |
| --- | --- |
| collibraSystemName | Regardless of the value set for the `useCollibraSystemName` property, the following is true:<br>◦ You must include this property in your configuration file.<br>◦ You can leave this property empty.<br>◦ Any value that you give is ignored.<br><br>If you want to specify name of the system or server, because, for example, you have multiple databases with the same name, then:<br><br>a. Set the `useCollibraSystemName` property to `true`.<br>b. Specify the system or server names in the `collibraSystemName` property in you Power BI <source ID> configuration file.<br><br>> Note This is a legacy property that will be deprecated in a future release. |

| Properties | Description |
| --- | --- |
| deleteRawMetadataAfterProcessing | The lineage harvester harvests metadata from specified data sources and uploads it in a ZIP file to a Collibra Data Lineage service instance, for processing.<br><br>You can use this optional property to specify whether or not the metadata should be deleted after it has been processed.<br><br>If the property is set to `true`, the metadata is deleted after processing. If set to `false`, the metadata is stored in an Amazon S3 bucket. |

See an example.

```
{
  "general": {
   "catalog": {
     "url": "https://catalog-instance.collibra.com",
     "username": "Admin"
     },
   "useCollibraSystemName": false,
   "useSharedDbModel": true
   },
   "sources": [ {
     "type": "PowerBI",
     "id": "power-bi-id",
     "tenantDomain": "collibrapowerbi.onmicrosoft.com",
     "loginFlow": {
       "type": "ServicePrincipal",
       "applicationId": "ab123cde-1234-1234-1234-abcd12e34fg5",
     },
     "domainId": "domain-reference-ID",
     "collibraSystemName": "collibra-system-name",
     "deleteRawMetadataAfterProcessing": true
   } ]
}
```

3. Save the configuration file.

4. Start the lineage harvester again in the console and run the following command:
   - for Windows: `.\bin\lineage-harvester.bat full-sync`
   - for other operating systems: `./bin/lineage-harvester full-sync`
5. If the lineage harvester prompts for credentials, enter the Power BI secret key value.
   » The secret key value is encrypted and stored in the **/config/pwd.conf** file.

## What's next?

The lineage harvester triggers Collibra to import Power BI assets and their relations and create a technical lineage for Power BI Column assets. Collibra also stitches the new Power BI assets to existing assets in Data Catalog.

To refresh the Power BI metadata in Data Catalog, you can run the lineage harvester again or schedule jobs to run them automatically.

> Tip You can check the progress of the Power BI ingestion and technical lineage creation in Activities. The **Results** field indicates how many relations were imported into Data Catalog.

> Warning When you run the lineage harvester, Collibra Data Lineage creates all Power BI assets in the Data Catalog BI domain (or domains) you specified in the Power BI <source ID> configuration file. We highly recommend that you do not move these assets to other domains. If you move assets to other domains, they will be deleted and recreated in the initial Data Catalog BI domains when you synchronize Power BI. As a result, all manually added characteristics of those assets are lost.

# Prepare Power BI <source ID> configuration file

The lineage harvester uses a lineage harvester configuration file to collect the Power BI data objects. It then sends the metadata to the Collibra Data Lineage service. However, if the `useCollibraSystemName` property in the lineage harvester configuration file is set to `true`, you also have to provide a <source ID> configuration file that defines the system name of databases in Power BI.

Collibra Data Lineage uses the system names to match the structure of databases in Power BI to assets in Data Catalog.

> Tip
> - You can also include a filters section in your <source ID> configuration file, to specify the Power BI workspaces from which you want to ingest metadata.
> - The name "<source ID>" refers to the value of the `sourceId` property in the lineage harvester configuration file.

## Steps

### Watch a video on how to do this

1. Create a new JSON file in the lineage harvester **config** folder.
2. Give the JSON file the same name as the value of the `sourceId` property in the lineage harvester configuration file.

> Example The value of the `sourceId` property in the lineage harvester configuration file is `power-bi-source-1`. Therefore, the name of your JSON file should be *power-bi-source-1.conf*.

> Important Your JSON file must have the file extension *.conf*.

3. For each database in Power BI, add the following content to the JSON file:

| Property | Description | Mandatory? |
|---|---|---|
| found_dbname=<database name>;found_hostname=<server name>;found_schema=<schema name> | The database information of supported data sources in Power BI that is typically collected by the lineage harvester. It allows you to specify the name of the database (`found_dbname`), on which server a database is running (`found_hostname`), and optionally, the name of the schema (`found_schema`).<br><br>Tip<br>You can use wildcards to capture multiple connection string combinations:<br><br>**Show me the supported wildcards**<br><br>Pattern / Description:<br>* — Matches everything.<br>? — Matches any single character.<br>[seq] — Matches any character in "seq".<br>[!seq] — Matches any character not in "seq". | Yes |
| dbname | The name of the database of a supported data source in Power BI. | No |

| Property | Description | Mandatory? |
|---|---|---|
| schema | The name of the default schema of a supported data source in Power BI.<br><br>If the lineage harvester fails to find a specific schema, it uses the default schema. | No |
| dialect | The dialect of the supported data source in Power BI.<br><br>Tip<br>You can enter one of the following values:<br>◦ *azure*, for an Azure SQL Server data source.<br>◦ *bigquery*, for a Google BigQuery data source.<br>◦ *mssql*, for a Microsoft SQL Server data source.<br>◦ *oracle*, for an Oracle data source.<br>◦ *redshift*, for an Amazon Redshift data source.<br>◦ *snowflake*, for a Snowflake data source.<br>◦ *sybase*, for a Sybase data source. | No |

| Property | Description | Mandatory? |
| --- | --- | --- |
| collibraSystemName | The system or server name of a database.<br><br>If you don't specify a value for this property, "DEFAULT" is shown in the technical lineage harvester.<br><br>**Warning** The value of this property must exactly match the name of your System asset in Collibra.<br><br>**Important** If you are using a <source ID> configuration file for the purpose of providing the true system name of an ODBC database in Power BI, you are not required to:<br>◦ Set the `useCollibraSystemName` property in the lineage harvester configuration file to `true`.<br>◦ Specify a Collibra system name in the <source ID> configuration file.<br>However, if the `useCollibraSystemName` property is set to `true` in the lineage harvester configuration file, then you must specify a Collibra system name in the <source ID> configuration file. | Yes (unless you are using a <source ID> file to provide the true system names of ODBC databases in Power BI.) |

<table>
<tr><th>Property</th><th>Description</th><th>Mandatory?</th></tr>
<tr><td>filters</td><td>This section allows you to specify the Power BI workspaces from which you want to ingest metadata.<br><br>Warning If you don't want to specify the Power BI workspaces from which to ingest, you must completely remove this filters section.<br><br>Note The filters work as "workspace AND workspace AND capacity AND capacity", meaning that if you specify a capacity, all of the workspaces in that capacity are also ingested.<br><br>Tip<br>You can use wildcards to capture multiple connection string combinations:<br><br>Show me the supported wildcards<br><table><tr><th>Pattern</th><th>Description</th></tr><tr><td>*</td><td>Matches everything.</td></tr><tr><td>?</td><td>Matches any single character.</td></tr><tr><td>[seq]</td><td>Matches any character in "seq".</td></tr><tr><td>[!seq]</td><td>Matches any character not in "seq".</td></tr></table></td><td>No</td></tr>
</table>

| Property | Description | Mandatory? |
|---|---|---|
| domainId | The unique resource ID of the domain (or domains), in Collibra Data Intelligence Cloud, in which you want to ingest the Power BI assets.<br><br>Tip You can find the domain ID by clicking the domain type. Then look in the URL of your browser to find the ID. The URL looks like https://<yourcollibrainstance>/domain/<domain ID>?<view>. | Yes |
| description | Any description, as you see fit. | Yes |
| workspaceNames | The names of Power BI workspaces from which you want to ingest metadata.<br><br>Important Any meta-characters in the name of a workspace must be enclosed in square brackets "[ ]". For example, a workspace with the name "Sale and Marketing [automobiles]" should be formatted as follows:<br>`Sale and Marketing []automobiles[]]` | No |
| workspaceIds | The IDs of Power BI workspaces from which you want to ingest metadata. | No |
| capacityNames | The names of capacities on which you want to filter. | No |

| Property | Description | Mandatory? |
|---|---|---|
| capacityIds | The IDs of capacities on which you want to filter.<br><br>**Warning** Any letters in a capacity ID must be in upper case. | No |

4. Save the <source ID> configuration file.

## Example of the <source ID>.conf file

```
{
   "found_dbname=databasename1;found_hostname=*;found_schema=schema1":
{
        "dbname": "mssql-database-name",
        "schema": "mssql-schema-name",
        "dialect": "mssql",
        "collibraSystemName": "mssql-system-name"
   },
   "found_dbname=databasename2;found_hostname=server-name.on-
microsoft.com;found_schema=schema2": {
        "dbname": "oracle-database-name",
        "schema": "oracle-schema-name",
        "dialect": "oracle",
        "collibraSystemName": "oracle-system-name"
   },
   "filters":[
        {
        "domainId": "<domain-ref-id>",
        "description": "FirstFilter",
        "workspaceNames": ["workspace1", "workspace2"],
        "workspaceIds": ["id3","id4"],
        "capacityNames": ["capacity1","capacity2"]
        },
        {
        "domainId": "<domain-ref-id>",
        "description": "SecondFilter",
        "workspaceNames": ["workspace3", "workspace4"],
        "capacityIds": ["id1","id2"]
        }
   ]
}
```

# Lineage harvesting app command options and arguments

After creating a configuration file, you can use the lineage harvester to perform specific actions with the data sources that are defined in your configuration file.

> Tip If you run the lineage harvester in command line, you will see an overview of possible command options and arguments that you can use. If there lineage harvester process fails, you can use the technical lineage troubleshooting guide to fix your issue.

## Typical command options and arguments

The following table shows the most commonly used command options and arguments.

> Note Although we do not recommend it, you can use more than one lineage harvester connected to a single Collibra Data Intelligence Cloud instance, if you want to separately process data sources on different servers. In this case, all lineage harvesters must share the same configuration file. You then determine which data sources are relevant when you run the `full-sync` command. If the configuration files are not identical, then one harvester will be deleting data harvested by the other harvester.

| Command | Description |
|---|---|
| `full-sync` | Uploads all your data sources to the Collibra Data Lineage service instance where the data source metadata is processed and uploaded to Data Catalog. |

| Command | Description |
| --- | --- |
| `-s "<ID of data source>"` | Uploads only the data source with the specified ID. For example, `full-sync -s "myOracleDataSource"`.<br><br>This command allows you to process data from a newly added data source or to refresh a data source in the configuration file, without refreshing the other data sources. This reduces the time you need to upload your data sources, since you only upload specific ones without affecting the others. If you want to process multiple data sources, add `-s "ID of another data source"` per data source to the command.<br><br>> Note You can use this argument multiple times to include multiple data sources. |
| `--no-matching` | Uploads a technical lineage without stitching the data objects in your technical lineage to the corresponding Column and Table assets in Data Catalog.<br><br>> Note As a result, you won't see the technical lineage of a specific Table or Column asset, but you can still see and browse the full technical lineage. |
| `load-sources` | Downloads all your data sources in a separate ZIP file, per data source, to the lineage harvester output folder. |
| `-s <ID of data source>` | Downloads only the data source with a specific ID. For example, `load-sources -s "myOracleDataSource"`.<br><br>> Note You can use this argument multiple times to include multiple data sources. |

| Command | Description |
| --- | --- |
| `cat passwords.json \| ./bin/lineage-harvester <command-like-full-sync> --passwords-stdin` | Provides passwords of your Collibra Data Intelligence Cloud instance and the data sources in your configuration file to the lineage harvester without storing the passwords in the lineage harvester folder.<br>You can replace `cat passwords.json` by a string generated by your password manager. |
| `test-connection` | Checks the connectivity to the Collibra Data Lineage service instance and to Data Catalog. The logs will also show the IP addresses of the Collibra Data Lineage service instances that you have to whitelist.<br>This command is mostly used for troubleshooting purposes. |
| --timeout <seconds> | Determines the network timeout. |
| `--help` | Shows an overview of all supported command options and arguments that you can use in the lineage harvester. |
| `--version` | Shows the version of the lineage harvester that you are using. |

# Automatic stitching

Stitching is a process that creates relations between database columns that are Column assets in Collibra Data Intelligence Cloud and BI assets representing the same database. Specifically, stitching creates relations between the following assets:

- The assets that are created when you ingest Power BI.
- The assets that are created when you register a data source.

The lineage harvester collects the Power BI source code and sends it to the Collibra Data Lineage service for analysis. The source code is then pushed to Data Catalog where relations are created between Power BI assets in Data Catalog.

At the same time, Collibra analyzes other metadata of data sources that you registered in Data Catalog and creates new relations of the type "Data Element targets / sources Data Element" between Power BI Column assets and Column assets in Data Catalog. It also creates a data flow between data objects, which is visualized in a technical lineage.

To clarify, the Power BI Column is the target of the Column, and the Column is the source of the Power BI Column

> Note When you ingest Power BI, a technical lineage for Power BI Column assets is automatically created.

## Stitching: matching the full paths of assets

To stitch assets in Data Catalog to data object collected by the lineage harvester, the Collibra Data Lineage service looks at the full path of the assets in Data Catalog and the full path of Power BI assets. If the full paths match, the Collibra Data Lineage automatically stitches them.

> Tip You can use the Stitching tab page to easily find the full path of assets in Data Catalog and data objects that were collected by the lineage harvester.

# Migrating your existing Power BI assets to the new integration method

A key feature of the Collibra Data Intelligence Cloud 2022.05 release was the ability to ingest Power BI metadata in Data Catalog via the lineage harvester. However, the new integration method was only available to customers who did not need to migrate existing Power BI assets. The following process now allows you to migrate your existing Power BI assets, making integration via the lineage harvester available to all Power BI customers.

# Steps

The table below shows the steps and prerequisites required to integrate Power BI in Data Catalog. These steps are best practices, which means that some of them might be optional, but highly recommended.

| Step | What? | Description | Prerequisites |
|---|---|---|---|
| 1 | Download and install the newest versions of:<br>• The Power BI harvester.<br>• The lineage harvester. | You need to run both harvesters, to synchronize the Power BI metadata via the original ingestion method, as described in Working with Power BI service.<br>You can download both harvesters from the Collibra Product Resource Downloads page. | See:<br>• Install the Power BI harvester.<br>• Install the lineage harvester. |

| Step | What? | Description | Prerequisites |
|---|---|---|---|
| 2 | Prepare both configuration files. | Prepare the Power BI configuration file and the lineage harvester configuration file, for ingestion via the original method, and then run the harvesters.<br><br>Empty configuration files are included in the respective folders, after installing each harvester. | See:<br>• Prepare the Power BI configuration file.<br>• Prepare the lineage harvester configuration file. |
| 3 | Update permissions. | Ensure that you have completed all of the prerequisite tasks ingesting Power BI metadata via the lineage harvester. | |
| 4 | Make random call to the Metadata API. | Make a call to the Metadata API, to ensure that you can communicate with the Power BI server. For example:<br><br>`GET https://api.powerbi.com/v1.0/myorg/admin/workspaces/modified` | |

| Step | What? | Description | Prerequisites |
|---|---|---|---|
| 5 | Manually refresh your Power BI datasets. | Manually refresh your Power BI datasets to ensure that the data in your Power BI datasets is up-to-date. This is necessary because the Power BI APIs use cached results.<br><br>For complete information, see:<br><br>• The Microsoft documentation.<br>• The Microsoft Power BI Blog. | See Power BI pre-requisites. |
| 6 | Edit the lineage harvester configuration file. | Edit the lineage harvester configuration file so that it includes the required properties for ingesting via the lineage harvester. See Prepare the lineage harvester configuration file for Power BI.<br><br>Warning The value of the `id` property in your lineage harvester configuration file must be the same as the value of the `sourceId` property in your Power BI configuration file. | See Prepare the lineage harvester configuration file for Power BI. |
| 7 | Run the lineage harvester with the `full-sync` command | Start the lineage harvester again in the console and run the following command:<br><br>• for Windows: `.\bin\lineage-harvester.bat full-sync`<br>• for other operating systems: `./bin/lineage-harvester full-sync`<br><br>This triggers the creation of Power BI assets, their relations and a technical lineage in Data Catalog.<br><br>We highly recommend that you always install and use the newest lineage harvester. | See Prepare the lineage harvester configuration file for Power BI. |

# Migrate your existing Power BI assets to the new integration method

The following process allows you to migrate Power BI assets that you integrated via the Power BI harvester, to the new integration method.

## Prerequisites

- You have Collibra Data Intelligence Cloud 2020.11 or newer.
- You have downloaded the newest Power BI harvester and lineage harvester from the Downloads page, and have the system requirements to install and use them.
- You have purchased the Power BI metadata connector and lineage feature.
- You meet the minimum system requirements.
- You have completed all of the prerequisite tasks.
    - You have registered Power BI in Microsoft Azure.
    - You have enabled the service principal option in the Power BI Admin portal.
- You have added Firewall rules so that the Power BI harvester and lineage harvester can connect to the Collibra Data Lineage service instance with one of the following IP addresses:
    - 15.222.200.199 (techlin-aws-ca.collibra.com)
    - 18.198.89.106 (techlin-aws-eu.collibra.com)
    - 54.242.194.190 (techlin-aws-us.collibra.com)
    - 51.105.241.132 (techlin-azure-eu.collibra.com)
    - 20.102.44.39 (techlin-azure-us.collibra.com)
    - 35.197.182.41 (techlin-gcp-au.collibra.com)
    - 34.152.20.240 (techlin-gcp-ca.collibra.com)
    - 35.205.146.124 (techlin-gcp-eu.collibra.com)
    - 34.87.122.60 (techlin-gcp-sg.collibra.com)
    - 35.234.130.150 (techlin-gcp-uk.collibra.com)
    - 34.73.33.120 (techlin-gcp-us.collibra.com)
- You have a global role with the Catalog global permission, for example Catalog Author.
- You have a global role with the Technical lineage global permission.
- You have a global role with the Data Stewardship Manager global permission.

- A resource role with the following resource permission on the community level in which you created the BI Data Catalog domain:
  - Asset: add
  - Attribute: add
  - Domain: add
  - Attachment: add
- Java Runtime Environment version 11 or newer or OpenJDK 11 or newer.
- You have a global role that has the Manage all resources global permission.
- You have created a BI Catalog domain (or domains) in which you want to ingest the Power BI assets.
- For a full ingestion, we highly recommend to have a Power BI Premium subscription.

## Steps

1. Download and install:
   - The Power BI harvester.
   - The lineage harvester.
2. Prepare the Power BI configuration file and the lineage harvester configuration file, and then run the harvesters.
3. Ensure that you have completed all of the prerequisite tasks to ingest Power BI metadata via the lineage harvester.
4. Make a random call to the Metadata API, to ensure that you can communicate with the Power BI server. For example:
   `GET https://api.powerbi.com/v1.0/myorg/admin/workspaces/modified`
5. Manually refresh your Power BI datasets, to ensure that the data is up-to-date. For complete information, see:
   - The Microsoft documentation.
   - The Microsoft Power BI Blog.
6. Edit the lineage harvester configuration file so that it includes the required properties to ingest Power BI metadata via the lineage harvester.

   > Warning  The value of the `id` property in your lineage harvester configuration file must be the same as the value of the `sourceId` property in your Power BI configuration file.

7. Start the lineage harvester again in the console and run the following command:
    - for Windows: `.\bin\lineage-harvester.bat full-sync`
    - for other operating systems: `./bin/lineage-harvester full-sync`

# Working with Power BI service

Power BI service is a cloud business intelligence software that helps you see and understand your data. You can ingest Power BI metadata in Data Catalog and create a technical lineage.

The Power BI service integration in Collibra Data Intelligence Cloud is not the same as the Power BI Report Server integration. If you want to ingest Power BI Report Server metadata in Collibra Data Intelligence Cloud, please read the Power BI Report Server section of the user guide.

> Note If you want to ingest Power BI metadata in Data Catalog, you have to purchase the Power BI connector and lineage feature.

## Features

Collibra Data Lineage currently supports two methods by which to integrate Power BI in Data Catalog. The following table shows the features specific to the two integration methods.

| Feature | Power BI via the Power BI harvester | Power BI via the lineage harvester |
|---|---|---|
| Catalog ingestion | ✓ | ✓ |
| Technical lineage | ✓ | ✓ |
| Automatic stitching | ✓ | ✓ |
| Uses only one harvester | | ✓ |

| Feature | Power BI via the Power BI harvester | Power BI via the lineage harvester |
|---|---|---|
| No Windows dependency | | ✓ |
| Uses new Power BI APIs | | ✓ |
| Workspace filtering for high-volume data | | ✓ |
| Mapping to target domains | | ✓ |
| Data flow support | | ✓ |

> Note Power BI via the Power BI harvester integration method is deprecated. We will continue to fix issues, but the development of new features and improvements is discontinued. For complete information on integrating Power BI via the lineage harvester and migrating existing Power BI assets, see the following topics:
>
> - Working with Power BI service
> - Migrating your existing Power BI assets to the new integration method

# Power BI terminology

Before you ingest Power BI, read more about the Power BI terminology and how it maps with the Collibra Data Intelligence Cloud asset types.

> Note For more information, see the Power BI documentation.

| Power BI term | Description | Asset type in Collibra |
|---|---|---|
| Capacity | A resource that hosts Power BI Work-spaces. | Power BI Capacity |
| Dashboard | A collection of Power BI tiles with metrics from one or more Reports and Data Models. | Power BI Dashboard |
| Data Set | A collection of data that is used to create a Power BI report. | Power BI Data Model |
| Data Set Column | A column in a Power BI Data Model. | Power BI Column |
| Data Set Table | A table in a Power BI Data Model. | Power BI Table |
| Report | A detailed view of a Power BI Data Model, with visualizations of findings and insights. | Power BI Report |
| Server or Tenant | A visual analytics platform for creating and storing Power BI Reports and Data Models. | Power BI Server |
| Tile | An element representing data on the Power BI Dashboard. | Power BI Tile |
| Workspace | A collection of Power BI Dashboards, Reports and Data Models. | Power BI Workspace |

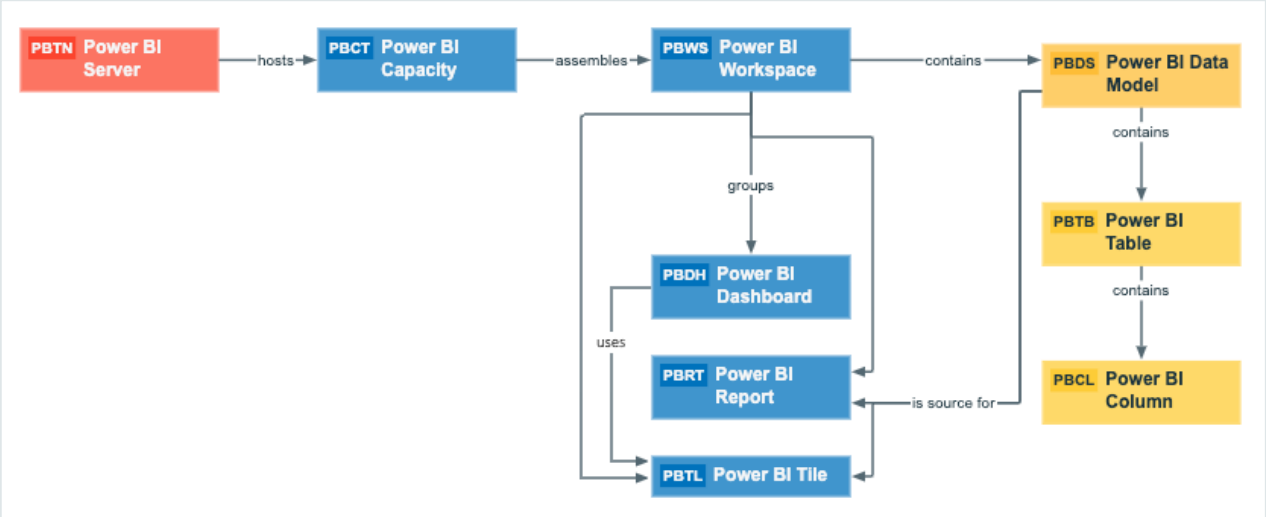
# Power BI operating model

The Power BI harvester collects Power BI metadata and sends it to the Collibra Data Lineage server. Collibra processes the metadata and creates new Power BI assets and relations in Data Catalog. You can see them on the asset page overview or visualize them in a diagram or in a technical lineage.

> Note
> - The assets have the same names as their counterparts in Power BI. Full names and Display names cannot be changed in Data Catalog.
> - Depending on your Power BI subscription, it could be that not all asset types are created.
> - Asset types are only created if you have all specific Power BI and Data Catalog permissions.
> - All Power BI asset types are created in the same domain.
> - Relations that were manually created between Power BI assets and other assets in accordance with the relation types in the Power BI operating model, are deleted after a refresh of the Power BI metadata.

# Power BI metadata overview

The following image shows the relations between Power BI asset types.



# Harvested metadata per asset type

This table shows the harvested Power BI metadata for each Power BI asset type, assuming you have the necessary subscriptions and configurations for a full ingestion. This table also shows the Resource IDs for each asset type and metadata.

| Asset type | Synchronized metadata | Resource ID |
|---|---|---|
| Power BI Capacity<br>Resource ID: 00000000-0000-0000-0000-100000000002 | Full name | |
| | Display name | |
| | Server hosts / is hosted in Business Dimension | 00000000-0000-0000-0000-120000000000 |
| | BI Folder assembles / is assembled in BI Folder | 00000000-0000-0000-0000-120000000001 |
| Power BI Column<br>Resource ID: 00000000-0000-0000-0000-100000000008 | Full name | |
| | Display name | |
| | Description | 00000000-0000-0000-0000-000000003114 |
| | Technical Data Type | 00000000-0000-0000-0000-000000000219 |
| | BI Data Model contains / is part of BI Data Attribute | 00000000-0000-0000-0000-000000007196 |
| | Data Element targets / sources Data Element | 00000000-0000-0000-0000-000000007069 |
| | Data Entity contains / is part of Data Attribute | 00000000-0000-0000-0000-000000007047 |

| Asset type | Synchronized metadata | Resource ID |
|---|---|---|
| Power BI Dashboard<br>Resource ID: 00000000-0000-0000-0000-100000000004 | Full name | |
| | Display name | |
| | Business Dimension groups / is grouped into Report | 00000000-0000-0000-0000-120000000002 |
| | Report uses / used in Report | 00000000-0000-0000-0000-120000000007 |
| | Report related to / impacted by Business Asset | 00000000-0000-0000-0000-120000000006 |
| Power BI Data Model<br>Resource ID: 00000000-0000-0000-0000-100000000007 | Full name | |
| | Display name | |
| | BI Data Model contains / is part of BI Data Attribute | 00000000-0000-0000-0000-000000007196 |
| | BI Folder contains / contained in Data Asset | 00000000-0000-0000-0000-120000000014 |
| | Data Asset is source for / source BI report | 00000000-0000-0000-0000-120000000013 |
| | Data Entity is part of / contains Data Model | 00000000-0000-0000-0000-000000007046 |

| Asset type | Synchronized metadata | Resource ID |
|---|---|---|
| Power BI Report<br>Resource ID: 00000000-0000-0000-0000-100000000006 | Full name | |
| | Display name | |
| | Business Dimension groups / is grouped into Report | 00000000-0000-0000-0000-120000000002 |
| | Data Asset is source for / source BI Report | 00000000-0000-0000-0000-120000000013 |
| | Report related to / impacted by Business Asset | 00000000-0000-0000-0000-120000000006 |
| | Report uses / used in Report | 00000000-0000-0000-0000-120000000007 |
| Power BI Server<br>Resource ID: 00000000-0000-0000-0000-100000000001 | Full name | |
| | Display name | |
| | Server hosts / is hosted in Business Dimension | 00000000-0000-0000-0000-120000000000 |
| Power BI Table<br>Resource ID: 00000000-0000-0000-0000-100000000009 | Full name | |
| | Display name | |
| | Description | 00000000-0000-0000-0000-000000003114 |
| | Data Entity contains / is part of Data Attribute | 00000000-0000-0000-0000-000000007047 |
| | Data Entity is part of / contains Data Model | 00000000-0000-0000-0000-000000007046 |

| Asset type | Synchronized metadata | Resource ID |
|---|---|---|
| Power BI Tile<br>Resource ID: 00000000-0000-0000-0000-100000000005 | Full name | |
| | Display name | |
| | Business Dimension groups / is grouped into Report | 00000000-0000-0000-0000-120000000002 |
| | Data Asset is source for / source BI Report | 00000000-0000-0000-0000-120000000013 |
| | Report related to / impacted by Business Asset | 00000000-0000-0000-0000-120000000006 |
| | Report uses / used in Report | 00000000-0000-0000-0000-120000000007 |
| Power BI Workspace<br>Resource ID: 00000000-0000-0000-0000-100000000003 | Full name | |
| | Display name | |
| | Description | 00000000-0000-0000-0000-000000003114 |
| | BI Folder assembles / is assembled in BI Folder | 00000000-0000-0000-0000-120000000001 |
| | BI Folder contains / contained in Data Asset | 00000000-0000-0000-0000-120000000014 |
| | Business Dimension groups / is grouped into Report | 00000000-0000-0000-0000-120000000002 |

> Note The metadata that is shown on the assets' pages depends on the asset type's assignment. As a result, you might not see all harvested metadata on the asset's page by default.

# Recommended hierarchy within a domain

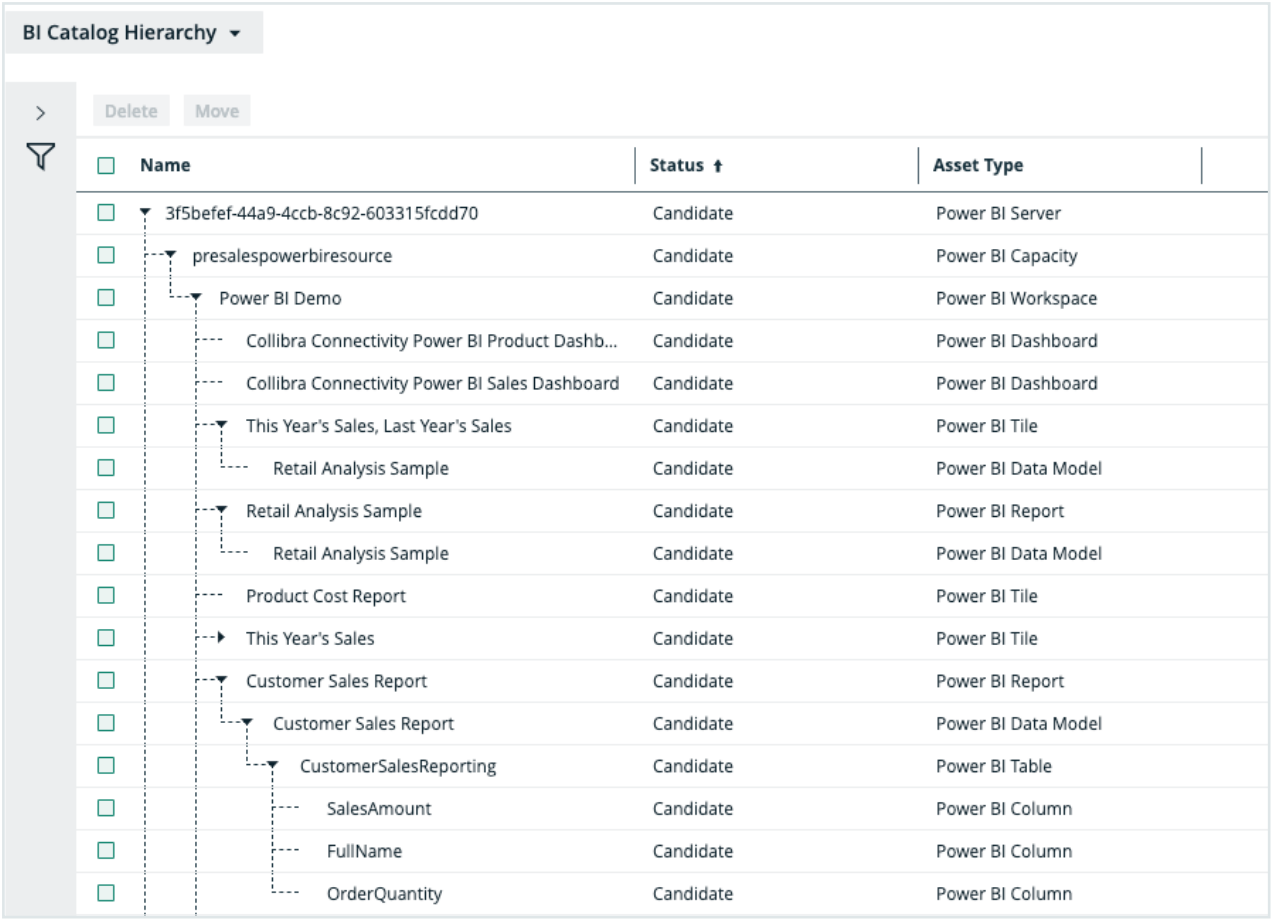
You can enable hierarchies for the domain (or domains) in which your Power BI assets were ingested. Doing so makes it easier to understand the relation between your Power BI assets, when viewing the assets on the domain page.

Follow these steps to enable and configure the recommended hierarchy.

## Steps

1. Open the domain page of the relevant BI Catalog domain.
2. In the content toolbar, click .
   » The **Configure Hierarchy** dialog box appears.
3. Select **Enable Hierarchy**.
4. Select **Single path**.
5. Start typing and select each of the following relation types:
   - Server **hosts** Business Dimension
   - BI Folder **assembles** BI Folder
   - Business Dimension **groups** Report
   - BI Report **source** Data Asset
   - Data Model **contains** Data Entity
   - Data Entity **contains** Data Attribute
6. Click **Apply**.

The following image shows an example of a BI Catalog domain with hierarchies enabled.




> Note In an asset view, like the one shown in the previous image, if any asset is deleted, for example via synchronization or manual deletion, the view is recreated and the hierarchy is lost. In this case, you can again enable and configure the recommended hierarchy.

# Power BI asset and domain types

The Power BI integration in Collibra Data Intelligence Cloud uses a specific subset of packaged asset types and domain types.

The following table contains the asset and domain types that are used for the Power BI integration. You can see the parent asset types in the breadcrumbs above each asset type.

| Asset type | Description | Domain type |
| --- | --- | --- |
| Business Asset ▸ Business Dimension ▸ BI Folder ▸ Power BI Capacity | A resource that hosts Power BI Workspaces. | BI Catalog |
| Business Asset ▸ Business Dimension ▸ BI Folder ▸ Power BI Workspace | A collection of Power BI Dashboards, Reports and Data Models. | BI Catalog |
| Business Asset ▸ Report ▸ BI Report ▸ Power BI Dashboard | A collection of Power BI tiles with metrics from one or more Reports and Data Models. | BI Catalog |
| Business Asset ▸ Report ▸ BI Report ▸ Power BI Report | A detailed view of a Power BI Data Model, with visualizations of findings and insights. | BI Catalog |
| Business Asset ▸ Report ▸ BI Report ▸ Power BI Tile | An element representing data on the Power BI Dashboard. | BI Catalog |

| Asset type | Description | Domain type |
| --- | --- | --- |
| Data Asset ▸ Data Element ▸ Data Attribute ▸ BI Data Attribute ▸ Power BI Column | A column in a Power BI Data Model. | BI Catalog |
| Data Asset ▸ Data Structure ▸ Data Entity ▸ BI Data Entity ▸ Power BI Table | A table in a Power BI Data Model. | BI Catalog |
| Data Asset ▸ Data Structure ▸ Data Model ▸ BI Data Model ▸ Power BI Data Flow | A collection of tables that are created and managed in workspaces in the Power BI service. | BI Catalog |
| Data Asset ▸ Data Structure ▸ Data Model ▸ BI Data Model ▸ Power BI Data Model | A collection of data that is used to create a Power BI report. | BI Catalog |
| Technology Asset ▸ Server ▸ BI Server ▸ Power BI Server | A visual analytics platform for creating and storing Power BI Reports and Data Models. | BI Catalog |

# Overview Power BI integration steps

The Power BI integration enables you to harvest Power BI metadata and create new Power BI assets in Data Catalog. Collibra analyzes and processes the BI metadata and presents it as specific asset types, retaining their original names.

> Tip To ingest Power BI metadata in Data Catalog, you need to run two different harvesters: the Power BI harvester and the lineage harvester. The order in which you run the harvesters is important. You first have to run the Power BI harvester to collect the metadata from your Power BI application and then run the lineage harvester to import new Power BI assets and their relations in Data Catalog. The Power BI ingestion workflow explains which roles the harvesters play in the Power BI ingestion process.

## Steps

The table below shows the steps and prerequisites required to integrate Power BI in Data Catalog. These steps are best practices, which means that some of them might be optional, but highly recommended.

| Step | What? | Description | Prerequisites |
|---|---|---|---|
| 1 | Set up a Power BI application. | Before you start the Power BI integration in Data Catalog, make sure that the Power BI harvester can reach the Power BI metadata. Perform these tasks before you start the actual Power BI ingestion process:<br>• The authentication process.<br>• The registration of your Power BI application in Microsoft Azure.<br>• The Power BI roles and dedicated capacities for Power BI workspaces.<br>• The required Power BI subscription.<br>Warning Because these tasks are performed outside of Collibra, it is possible that the content changes without us knowing. We strongly recommend that you carefully read the source documentation. | You have a Power BI subscription. |
| 2 | Create a new domain. | Before you can ingest Power BI metadata, you have to create a new domain or choose an existing domain to store the new Power BI assets. | You have a resource role with the following resource permissions:<br>• Domain: Add |

| Step | What? | Description | Prerequisites |
|---|---|---|---|
| 3 | Optionally, assign the attribute type State to the global assignment of the Power BI Workspace asset type | On Power BI Workspace asset pages, you can include the attribute type State, to show the state of ingested Power BI workspaces. To do so, you have to edit the global assignment of the Power BI Workspace asset type and assign the attribute type State.<br><br>If you delete a Power BI workspace, the workspace is maintained for a 90-day grace period. During the grace period, the workspace has the state Deleted. When you ingest Power BI metadata in Data Catalog, this deleted workspace is ingested.<br><br>For complete information on Power BI workspaces and possible states, see the Microsoft Power BI documentation. | You have a global role that has the System administration global permission. |
| 4 | Ingest or import assets from supported JDBC data sources. | The Collibra Data Lineage server connects to Data Catalog and reads the full paths of existing assets. When the full path matches the full path of assets in Power BI, the Collibra Data Lineage server automatically stitches them. | Permissions depend on how you ingest or import the assets. |

| Step | What? | Description | Prerequisites |
|---|---|---|---|
| 5 | Download and install the Power BI harvester. | You use the Power BI harvester to collect metadata from Power BI and upload it to Collibra where the metadata is scanned, processed and analyzed.<br><br>You can download the Power BI harvester from the Collibra Product Resource Downloads page. The installer file contains the following:<br><br>• a config folder with an empty configuration file.<br>• a bin folder.<br>• a TXT file with more information about the configuration file.<br>• a BAT file that you use to run the harvester. | • You have Collibra Data Intelligence Cloud 2020.11 or newer.<br>• You have access to the Power BI harvester on the Downloads page.<br>• Your environment meets the system requirements to install and use the Power BI harvester.<br>• You have added Firewall rules so that the Power BI harvester can connect to the Collibra Data Lineage server with one of the following IP addresses:<br>  • 15.222.200.199 (techlin-aws-ca.-collibra.com)<br>  • 18.198.89.106 (techlin-aws-eu.-collibra.com)<br>  • 54.242.194.190 (techlin-aws-us.-collibra.com)<br>  • 51.105.241.132 (techlin-azure-eu.-collibra.com)<br>  • 20.102.44.39 (tech- |

| Step | What? | Description | Prerequisites |
| --- | --- | --- | --- |
| | | | lin-azure-us.-collibra.com)<br>• 35.197.182.41 (techlin-gcp-au.-collibra.com)<br>• 34.152.20.240 (techlin-gcp-ca.-collibra.com)<br>• 35.205.146.124 (techlin-gcp-eu.-collibra.com)<br>• 34.87.122.60 (techlin-gcp-sg.-collibra.com)<br>• 35.234.130.150 (techlin-gcp-uk.-collibra.com)<br>• 34.73.33.120 (techlin-gcp-us.-collibra.com)<br><br>Important<br>Ingestion results vary according to your Power BI subscription. |

| Step | What? | Description | Prerequisites |
|---|---|---|---|
| 6 | Prepare the Power BI configuration file and run the Power BI harvester. | You create a configuration file to provide the connection information that you need to connect your Power BI application to Collibra and to the domain in which you want to ingest the Power BI assets.<br><br>You can access an empty configuration file in the Power BI harvester installation folder. When you have created and saved the configuration file, you can run the Power BI harvester which uploads the Power BI metadata to Collibra. | • You have access to the Power BI harvester on the Downloads page.<br>• You have completed all prerequisite tasks.<br>• You have a dedicated domain to ingest the Power BI assets.<br>• You have a global role with the Catalog global permission, for example Catalog Author.<br>• You have a global role with the Technical lineage global permission.<br>• You have a global role with the Data Stewardship Manager global permission.<br>• A resource role with the following resource permission on the community level in which you created the BI Data Catalog domain:<br>  ◦ Asset: add<br>  ◦ Attribute: add<br>  ◦ Domain: add<br>  ◦ Attachment: add<br>• Your environment |

| Step | What? | Description | Prerequisites |
|---|---|---|---|
| | | | meets the system requirements to run the Power BI harvester and the lineage harvester.<br><br>Tip For a full ingestion, we highly recommend to have a Power BI Premium subscription. |
| 7 | Download and install the lineage harvester. | You use the lineage harvester to trigger the creation of Power BI assets, their relations and a technical lineage in Data Catalog.<br><br>You can download the lineage harvester from the Collibra Product Resource Downloads page. | Your environment meets the system requirements to install and use the lineage harvester. |

| Step | What? | Description | Prerequisites |
|---|---|---|---|
| 8 | Prepare the lineage harvester configuration file and run the lineage harvester. | You create a lineage harvester configuration file with Power BI connection information and run the lineage harvester to import the results of the Power BI integration and the technical lineage for Power BI into Data Catalog.<br><br>As a result, Collibra creates new Power BI assets in Data Catalog and imports relations between these assets. It also creates a technical lineage for Power BI assets and other data sources in the lineage harvester configuration file.<br><br>Tip For more information about the lineage harvester, see the Collibra Data Lineage documentation. | • You have downloaded the lineage harvester version 1.2.1 or newer.<br>• Your environment meets the system requirements to install and run the lineage harvester.<br>• You have prepared a Power BI harvester configuration file.<br>• You have a global role with the Catalog global permission, for example Catalog Author.<br>• You have a global role with the Technical lineage global permission.<br>• A resource role with the following resource permission on the community level in which you created the BI Data Catalog domain:<br>◦ Asset: add<br>◦ Attribute: add<br>◦ Domain: add<br>◦ Attachment: add |

| Step | What? | Description | Prerequisites |
|---|---|---|---|
| 9 | View the Power BI assets and technical lineage | After the Power BI metadata is ingested in Data Catalog, you can go to the domain where you ingested Power BI and see the list of ingested Power BI assets. These assets are automatically stitched to existing assets in Data Catalog.<br><br>You can go to a Power BI Column asset page and click the Technical lineage lineage tab to view the technical lineage.<br><br>Note If you ingest Power BI for the first time or if you change your geolocation or cloud provider, you have to restart the DGC service before you can see your technical lineage.<br><br>Warning When you run the harvesters, Collibra Data Lineage creates all Power BI assets in the same Data Catalog BI domain. We highly recommend that you do not move these assets to another domain. If you move assets to another domain, they will be deleted and recreated in the initial Data Catalog BI domain when you synchronize | You have a Data Catalog global role with the Technical lineage global permissions. |

| Step | What? | Description | Prerequisites |
|---|---|---|---|
| | | Power BI. As a consequence, all manually added data of those assets is lost. | |

Note The order in which you run the harvesters is important. You first have to run the Power BI harvester to collect the metadata from your Power BI application and then run the lineage harvester to import new Power BI assets and their relations in Data Catalog.

# Ingestion results based on Power BI subscriptions

You can only ingest Power BI metadata to which the Power BI user has access to. Your level of access is determined by your Power BI subscriptions.

The following table gives an overview of the minimum required subscriptions and the results in Data Catalog.

Note

- The assets have the same names as their counterparts in Power BI. Full names and Display names cannot be changed in Data Catalog.
- Depending on your Power BI subscription, it could be that not all asset types are created.
- Asset types are only created if you have all specific Power BI and Data Catalog permissions.
- All Power BI asset types are created in the same domain.
- Relations that were manually created between Power BI assets and other assets in accordance with the relation types in the Power BI operating model, are deleted after a refresh of the Power BI metadata.

| Minimum required subscription | Result in Data Catalog |
|---|---|
| Power BI Pro | Power BI Workspaces are not assigned to a dedicated capacity in Power BI.<br><br>The following asset types are created in Data Catalog:<br><br>• Power BI Server<br>• Power BI Workspace<br>• Power BI Dashboard<br>• Power BI Tile<br>• Power BI Report<br>• Power BI Data Model<br><br>Technical lineage is unavailable. |
| Power BI Pro with Power BI Embedded Capacity subscription in Microsoft Azure | Power BI Workspaces are assigned to a embedded capacity in Azure.<br><br>The following asset types are created in Data Catalog:<br><br>• Power BI Server<br>• Power BI Capacity<br>• Power BI Workspace<br>• Power BI Dashboard<br>• Power BI Tile<br>• Power BI Report<br>• Power BI Data Model<br>• Power BI Table<br>• Power BI Column<br><br>A technical lineage is created for all Power BI Column assets. |

| Minimum required subscription | Result in Data Catalog |
| --- | --- |
| Premium | Power BI Workspaces are assigned to a dedicated capacity in Power BI.<br><br>The following asset types are created in Data Catalog:<br><br>• Power BI Server<br>• Power BI Capacity<br>• Power BI Workspace<br>• Power BI Dashboard<br>• Power BI Tile<br>• Power BI Report<br>• Power BI Data Model<br>• Power BI Table<br>• Power BI Column<br><br>A technical lineage is created for all Power BI Column assets. |

| Minimum required subscription | Result in Data Catalog |
| --- | --- |
| Premium Per User | Power BI Workspaces are assigned to a dedicated capacity in Power BI.<br><br>The following asset types are created in Data Catalog:<br><br>• Power BI Server<br>• Power BI Capacity<br>• Power BI Workspace<br>• Power BI Dashboard<br>• Power BI Tile<br>• Power BI Report<br>• Power BI Data Model<br>• Power BI Table<br>• Power BI Column<br><br>A technical lineage is created for all Power BI Column assets.<br><br>> Note The Power BI Premium Per User license is a new license type that is released for general availability, but is still in its preview period. For more information, see the Microsoft documentation. |

> Note We highly recommend you to have a Power BI Premium subscription. Power BI Premium also provides additional features and a better speed and performance.

> Warning When you run the harvesters, Collibra Data Lineage creates all Power BI assets in the same Data Catalog BI domain. We highly recommend that you do not move these assets to another domain. If you move assets to another domain, they will be deleted and recreated in the initial Data Catalog BI domain when you synchronize Power BI. As a consequence, all manually added data of those assets is lost.

# Power BI ingestion limitations

There are a few considerations and limitations that you must take into account when you use the Power BI metadata connector and lineage feature.

## Supported subscriptions

> Important Your subscription determines which Power BI metadata the Power BI harvester can collect.

You need one of the following subscriptions to ingest Power BI metadata in Data Catalog:

- Power BI Pro. To ensure a full ingestion, you also need a Power BI Embedded Capacity subscription in Microsoft Azure.
- Power BI Premium.
- Power BI Premium Per User.

  > Note The Power BI Premium Per User license is a new license type that is released for general availability, but is still in its preview period. For more information, see the Microsoft documentation.

Other Power BI subscriptions are currently not supported.

## Power BI metadata

The Power BI harvester can only partially access metadata of the following Power BI elements:

- Classic Power BI workspaces, which include My Workspace. Only a full ingestion of new Power BI workspaces is supported.
- Power BI workspaces that are not part of a dedicated capacity.
- Descriptions of most Power BI elements.
- Power BI apps.They can be ingested as Power BI Reports, but there is no easy way to distinguish them from real Power BI reports.

The Power BI harvester cannot access metadata of the following Power BI elements:

- Dataflows.
- Tile subtitles.
- Data from external sources supplying the input for the Power Query expressions in Power BI.

> Important The Collibra Data Lineage server can process most, but not all, complex Power BI metadata. This means that the success rate of a Power BI ingestion can be very high, but almost never 100%.

# Known issues

The following table presents the known issues of the Power BI integration in Collibra Data Intelligence Cloud.

| Known issue | Description |
|---|---|
| The Power BI harvester shows an `Internal Server Error` because of the Power BI workspace filter. | If you want to ingest a lot of Power BI data and you use the WorkspaceFilter in the Power BI configuration file, the Power BI API can go in timeout and, as a result, you get an `Internal Server Error`.<br><br>If you get this error, we highly advise to not use the workspace filter.<br><br>> Note If the error message indicates that the issue is an internal server error, the problem is caused by the Power BI REST API, not the Power BI harvester itself. |
| The data set *Report Usage Metrics Model* cannot be ingested. | The *Report Usage Metrics Model* is a data set that is automatically created by Power BI. This data set does not contain actual data, which means that they contain nothing to ingest into Data Catalog.<br><br>However, the Power BI harvester still tries to access the metadata and, since there is nothing to access, shows an error message. All error messages about the *Report Usage Metrics* can be ignored. |

| Known issue | Description |
| --- | --- |
| The `IN` operator is not supported. | Currently, the `IN` operator is not supported. As a result, you cannot use `IN` to filter the `workspaceFilter` property on specific Power BI workspace names in the Power BI harvester configuration file.<br><br>For example, you want to filter on two Power BI workspace names. In the configuration file, you can enter the following value of the `workspaceFilter` property to ingest only workspaces with the name "workspace1" or "workspace2<br><br>`"workspaceFilter": "name eq 'workspace1' or name eq 'workspace2'"`<br><br>However, you cannot ingest Power BI workspaces that have "workspace1" or "workspace2" in their name, because the IN operator is currently **not supported**:<br><br>`"name in ('workspace1', 'workspace2')"`<br><br>Tip For more syntax examples that you can use in the `workspaceFilter` property, see the README file attached to the Power BI harvester or see the Microsoft documentation. |
| Power BI assets that are moved to a different domain are deleted after synchronization. | When you run the harvesters, Collibra Data Lineage creates all Power BI assets in the same Data Catalog BI domain. We highly recommend that you do not move these assets to another domain. If you move assets to another domain, they will be deleted and recreated in the initial Data Catalog BI domain when you synchronize Power BI. As a consequence, all manually added data of those assets is lost. |

| Known issue | Description |
| --- | --- |
| You have successfully ingested Power BI metadata, but calculated tables and columns are not shown in the Technical lineage or in the browse tab pane. | Calculated columns are virtually the same as a non-calculated columns, with one exception: their values are calculated using DAX formulas and values from other columns. Collibra Data Lineage currently does not support internal transformations via DAX language, and any data objects derived via DAX are not shown in the technical lineage or in the browse tab pane. Currently, only M Query/Power Query expressions are supported. |
| You are experiencing Power Query parsing errors when ingesting Power BI metadata. | The Power Query parser does not currently support parsing for:<br>• The following functions:<br>◦ MicrosoftAzureConsumptionInsights.Tables<br>◦ Table.ExpandRecordColumn<br>◦ Dates Query<br>• Transact-SQL statements |

# Supported data sources in Power BI

Power BI is business intelligence software that can integrate with various data sources. When you ingest Power BI metadata, Collibra Data Lineage tries to automatically stitch this metadata to data sources registered in Data Catalog. It also creates a technical lineage that shows where metadata is used and how it transforms.

The following table shows the supported data source types in Power BI that have been tested.

> Warning Although the following data sources have been tested extensively, there still may be some issues caused by unsupported elements within the data source or limitations in the Power BI integration process.

| Power BI data source | Connection type | Technical lineage | Stitching to registered data sources in Data Catalog |
|---|---|---|---|
| Amazon Redshift | Import | Yes | Yes |
| Azure Databricks | Import | Yes | Yes |
| Google BigQuery | Import | Yes | Yes |
| ODBC | Import | Yes | Yes<br><br>Important You need to use a Power BI <source ID> configuration file to provide the true system names of the ODBC databases in Power BI. For more information, see Providing ODBC database names in Power BI. |
| Oracle | Import | Yes | Yes |
| Snowflake | Import | Yes | Yes |
| SQL Server | Import | Yes | Yes |
| Sybase | Import | Yes | Yes |

> Note We cannot guarantee that other data sources in Power BI can be stitched successfully.

# Providing ODBC database names in Power BI

You can create a technical lineage for ODBC data sources in Power BI. However, ODBC database names often can't be determined. When a database name can't be determined, it's given a substitute name, which is the ODBC connection string.

This substitute name can be seen in the technical lineage, but it is merely a placeholder that doesn't carry any meaning if you're trying to identify the database it represents in the technical lineage. A bigger problem is that if you want to stitch the ODBC database to assets in Data Catalog, the substitute name won't match with any ingested databases, so stitching won't work.

To ensure that the true database names appear in the technical lineage, and to ensure successful stitching, you can use a Power BI <source ID> configuration file to provide the true system names of the ODBC databases in Power BI.

> Tip The name "<source ID>" refers to the value of the `sourceId` property in the Power BI configuration file. If, for example, the value of the `sourceId` property in the Power BI configuration file is `power-bi-source-1`, then the name of your <source ID> configuration file should be *power-bi-source-1.conf*.

## Example of the <source ID> configuration file

For each ODBC database in Power BI, add the following content to the JSON file:

```
{
   "found_dbname=DSN_MYDATABASE;found_hostname=ODBC": {
       "dbname": "DB001",
       "schema": "MYSCHEMA",
       "dialect": "oracle",
       "collibraSystemName": "oracle-system-name"
   }
}
```

| Property | Description |
|---|---|
| found_ dbname=<substitute database name>;found_ hostname=<server name> | `found_dbname` is the substitute database name. You need to convert it to uppercase and replace every non-alphanumeric character by an underscore (_). In this example, the substitute name is “dsn=MYDATABASE”, so you should use "DSN_MYDATABASE".<br><br>Note The substitute name is the ODBC connection string, which can be lengthy when it includes the driver and parameters in full.<br><br>`found_hostname` should be “ODBC”, but you can also use an asterisk (*). |
| dbname | The true system name of the ODBC database in Power BI. |
| schema | The name of the default schema of the ODBC database in Power BI.<br><br>If no schema is specified and the Power BI harvester fails to find a specific schema, it uses the default schema. |
| dialect | The dialect of the ODBC connection.<br><br>The dialect must be one of the supported SQL dialects. If no dialect is specified, “mssql” is used, by default.<br><br>Tip<br>You can enter one of the following values:<br>• *azure*, for an Azure SQL Server data source.<br>• *bigquery*, for a Google BigQuery data source.<br>• *mssql*, for a Microsoft SQL Server data source.<br>• *oracle*, for an Oracle data source.<br>• *redshift*, for an Amazon Redshift data source.<br>• *snowflake*, for a Snowflake data source.<br>• *sybase*, for a Sybase data source. |

| Property | Description |
| --- | --- |
| collibraSystemName | The system or server name of a database.<br><br>Important Because you are using a <source> configuration file only for the purpose of providing the true system name of an ODBC database in Power BI, you are not required to:<br>• Set the useCollibraSystemName property in the Power BI configuration file to `true`.<br>• Specify a Collibra system name in the <source ID> configuration file.<br>However, if the useCollibraSystemName property is set to `true` in the Power BI configuration file, then you must specify a Collibra system name in the <source ID> configuration file. |

For complete information on working with <source ID> configuration files, see Power BI <source ID> configuration file.

# Power BI prerequisites

Before you start the Power BI integration process, you have to perform a number of tasks in Power BI and Microsoft Azure. These tasks, which are performed outside of Collibra, are needed to enable the Power BI harvester to reach your Power BI application and collect its metadata.

The tasks include the following:

- The authentication process.
- The registration of your Power BI application in Microsoft Azure.
- The Power BI dedicated capacities and roles for Power BI workspaces.

The metadata harvesting process explains in detail which prerequisites you need to enable the Power BI harvester to collect the Power BI metadata.

Note There are some limitations to the metadata harvesting process. Ensure that you understand these limitations before you start the harvesting process.

Warning Because these tasks are performed outside of Collibra, it is possible that the content changes without us knowing. We strongly recommend that you carefully read the source documentation.

# Supported Power BI subscriptions

Important Your subscription determines which Power BI metadata the Power BI harvester can collect.

You need one of the following subscriptions to ingest Power BI metadata in Data Catalog:

- Power BI Pro. To ensure a full ingestion, you also need a Power BI Embedded Capacity subscription in Microsoft Azure.
- Power BI Premium.
- Power BI Premium Per User.

  Note The Power BI Premium Per User license is a new license type that is released for general availability, but is still in its preview period. For more information, see the Microsoft documentation.

Tip We highly recommend you to have a Power BI Premium subscription.

# Authentication

You have to be authenticated to access Power BI metadata. Your authentication method determines how you retrieve the metadata.The Power BI harvester supports two types of authentication:

- Username and password
- Service principal authentication

The metadata harvesting process is different for each authentication method. As a result, different configurations in Microsoft Azure and Power BI are required.

> Note We recommend that you use the service principal authentication.

## Username and password

The username and password authentication method relies on the username, in the form of an email address, and a password you provide to access the Power BI metadata.

To use the username and password authentication, you need to be an Azure Active Directory user with a Power BI admin role in Power BI and have a Contributor role in the Power BI workspaces that you want to ingest into Data Catalog.

When you become an Azure Active Directory user, a new email address is created. You use this email address to sign in to Power BI.

The email address that is created in Microsoft Azure is the username that you use to sign in to Power BI. You can store the username and password you use to sign in to Power BI in the Power BI configuration file.

In the Power BI Tenant settings in Power BI, you have to enable the **Allow XMLA endpoints and Analyze in Excel with on-premises datasets**. This setting has to be applied to the entire organization (default) or to the specific security group to which your workspaces belong.

> Note Only Azure Administrators can create users and require them to authenticate via username and password. The Azure Administrator also assigns the user the Power BI admin role. This user is only created for the purpose of Power BI integration in Collibra Data Intelligence Cloud. The user in Azure should have a Member user type.

## Service principal

The Service Principal authentication method lets an Azure Active Directory automatically access Power BI.

The Service Principal authentication relies on the Power BI Tenant ID and the Azure Active Directory application ID that you provide in the configuration file. The password you need to access Power BI is the client secret key of the Azure Active Directory application.

To use the Service principal authentication, you need to embed Power BI content with a Service Principal and an application secret. This means that you do the following:

- Create an Azure AD security group.
- Add the security group in the Power BI Tenant settings in Power BI.
- In the Power BI Admin portal, you also do the following :
  - Enable the **Allow service principals to use read-only Power BI admin APIs (preview)** option.
  - Enable the **Allow service principal to use Power BI APIs** option in the Developer settings.
  - Apply the option to specific security groups.
  - Enter the name of the security group to which you want to add the service principal.
  - Enable the **Allow XMLA endpoints and Analyze in Excel with on-premises datasets**. This setting has to be applied to the entire organization (default) or to the specific security group to which your workspaces belong.

> Note  You need Power BI administrator rights to access the Power BI Admin portal.

- Assign the Contributor role to the security group in the Power BI workspaces you want to ingest.

> Tip  Do not confuse the **Allow service principals to use read-only Power BI admin APIs (preview)** option with the **Allow service principal to use Power BI APIs** option. You need to enable both options.

# Register Power BI in Microsoft Azure and set permissions

Before you set up the Power BI harvester, make sure that the harvester can reach Power BI by registering Power BI in Azure and setting the necessary permission to harvest the

metadata.

We highly recommend that you read about supported authentication methods before you register Power BI in Microsoft Azure.

> Warning This procedure is performed outside of Collibra. A third party may change the software without notification, which can render this documentation out of date. We highly recommend that you carefully read the source documentation.

## Steps

> Tip The content in this topic is different for the username / password authentication method or service principal authentication method. We highly recommend that you read the following instructions carefully before you register Power BI in Microsoft Azure:
>
> - Service principal instructions
> - Username / password instructions

1. Register Power BI in the Azure Portal using the following settings:

| Setting | Description |
|---|---|
| Name | The name of your Power BI application. |
| Supported account types | The type of tenant. This indicates who can access the Power BI application.<br><br>In this case, the supported account type must be *Single tenant*. |
| Redirect URI | The location to which a user's client is redirected and where security tokens are sent after a successful authorization.<br><br>In this case, the redirected URI must be *Web*, but you do not have to specify any web location. |

   » When you have registered Power BI, the Azure portal creates two important IDs that you need in the Power BI configuration file:
   - The Application (client) ID
   - The Directory (tenant) ID

   > Note We highly recommend that you store these IDs for further use. You can find the IDs in the **Overview** pane on the Azure portal or in the top right menu.

2. Create a user with the Power BI Administrator role (only for username / password authentication).
3. In the Azure portal, go to **Authentication** pane and do the following:
   a. Go to the **Advanced settings** section.
   b. Set the **Treat application as a public client** to *Yes*.
4. Go to the **API permissions** pane and do the following:
   a. Select **Delegated permissions** as permission type.
   b. Grant the Power BI application in Microsoft Azure the Microsoft Graph `User.Read` permission.
   c. Grant the Power BI application in Microsoft Azure all Power BI Service permissions (only for username / password authentication).

d. Set **Admin consent required** for `Tenant.ReadAll` permission to *Yes* (only for username / password authentication).

» The user now has the following permissions:

- Microsoft Graph
  - `User.Read`
- Power BI Service (only for username / password authentication)
  - `App.Read.All`
  - `Capacity.Read.All`
  - `Dashboard.Read.All`
  - `Dataflow.Read.All`
  - `Group.Read.All`
  - `Report.Read.All`
  - `Tenant.Read.All`, with **Admin consent required** set to `Yes`.
  - `Workspace.Read.All`

5. In the Power BI Admin portal, do the following (only for service principal authentication):
   a. Enable the **Allow service principals to use read-only Power BI admin APIs (preview)** option.
   b. Enable the **Allow service principal to use Power BI APIs** option in the Developer settings.
   c. Apply the option to specific security groups.
   d. Enter the name of the security group to which you want to add the service principal.
   e. Enable the **Allow XMLA endpoints and Analyze in Excel with on-premises datasets**.
   f. Apply the integration setting to the entire organization (default) or to the specific security group to which your workspaces belong.

> Note You need Power BI administrator rights to access the Power BI Admin portal.

6. In the Power BI Admin portal, do the following (only for username / password authentication):
   a. Enable the **Allow XMLA endpoints and Analyze in Excel with on-premises datasets**.

b. Apply the integration setting to the entire organization (default) or to the specific security group to which your workspaces belong.

## What's next?

You can add your Power BI workspaces to a dedicated capacity.

# Power BI workspaces

Power BI workspaces represent the most used metadata in Power BI. They contain, for example, reports and data sets. If you want a full ingestion, you have to make sure that the Power BI harvester can access all metadata in your Power BI workspaces. Consider the following:

- Depending on the authentication type, you must have specific roles and permissions to access the metadata in the Power BI workspaces.
- You can only fully ingest new Power BI workspaces. This means that classic workspaces and My Workspace in Power BI are not supported.

> Tip You can filter on Power BI workspaces in the Power BI configuration file.

## Roles and permissions

Depending on the authentication type that you want to use, you also require additional permissions in the Power BI workspaces to access the Power BI metadata.

- In case of username / password authentication, the Azure Active Directory user with a Power BI admin role in Power BI must have the Contributor role in the Power BI workspaces you want to ingest.
- In case of Service Principal authentication, you have to add the Active Directory security group to which you added the Service Principal to your Power BI workspaces. The Power BI workspaces you want to ingest must have the Contributor role in the Power BI security group.

# Ingesting deleted workspaces

If you delete a Power BI workspace, the workspace is maintained for a 90-day grace period, during which a Power BI administrator can restore the workspace. During the grace period, the workspace has the state Deleted. When you ingest Power BI metadata in Data Catalog, this deleted workspace is ingested.

When the grace period elapses, the state of the workspace becomes Removing, for a short time, while it is being permanently removed. The state then becomes Not found. At this point, as the workspace no longer exists in Power BI, the Power BI Workspace asset in Collibra will also be deleted upon the next synchronization.

## Why are deleted workspaces ingested?

Let's image that you ingest a Power BI workspace with the Active state and that over time, you add comments, tags and characteristics to the asset in Collibra. Now let's imagine that the workspace is deleted in Power BI and we do not ingest the deleted workspace. In this case, the Power BI Workspace asset in Collibra is deleted upon the next synchronization. But what if the Power BI administrator decides, during the 90-day grace period, to restore the workspace in Power BI? Upon the next synchronization, a new Power BI Workspace asset is created in Collibra, but all of the comments, tags and characteristics that were part of the deleted asset are lost.

By ingesting deleted Power BI workspaces, we safeguard against losing any of the additional information on the Power BI Workspace asset, in case a Power BI administrator decides to restore a workspace during the grace period.

## Viewing workspace states in Collibra

On Power BI Workspace asset pages, you can include the attribute type State, to show the state of ingested Power BI workspaces. To do so, you have to edit the global assignment of the Power BI Workspace asset type and assign the attribute type State.

For complete information on Power BI workspaces and possible states, see the Microsoft Power BI documentation.

# The metadata harvesting process

Collibra uses two methods to harvest Power BI metadata: via REST API calls and via XMLA endpoints. The REST API retrieves basic metadata, and XMLA endpoints retrieve more specific metadata.

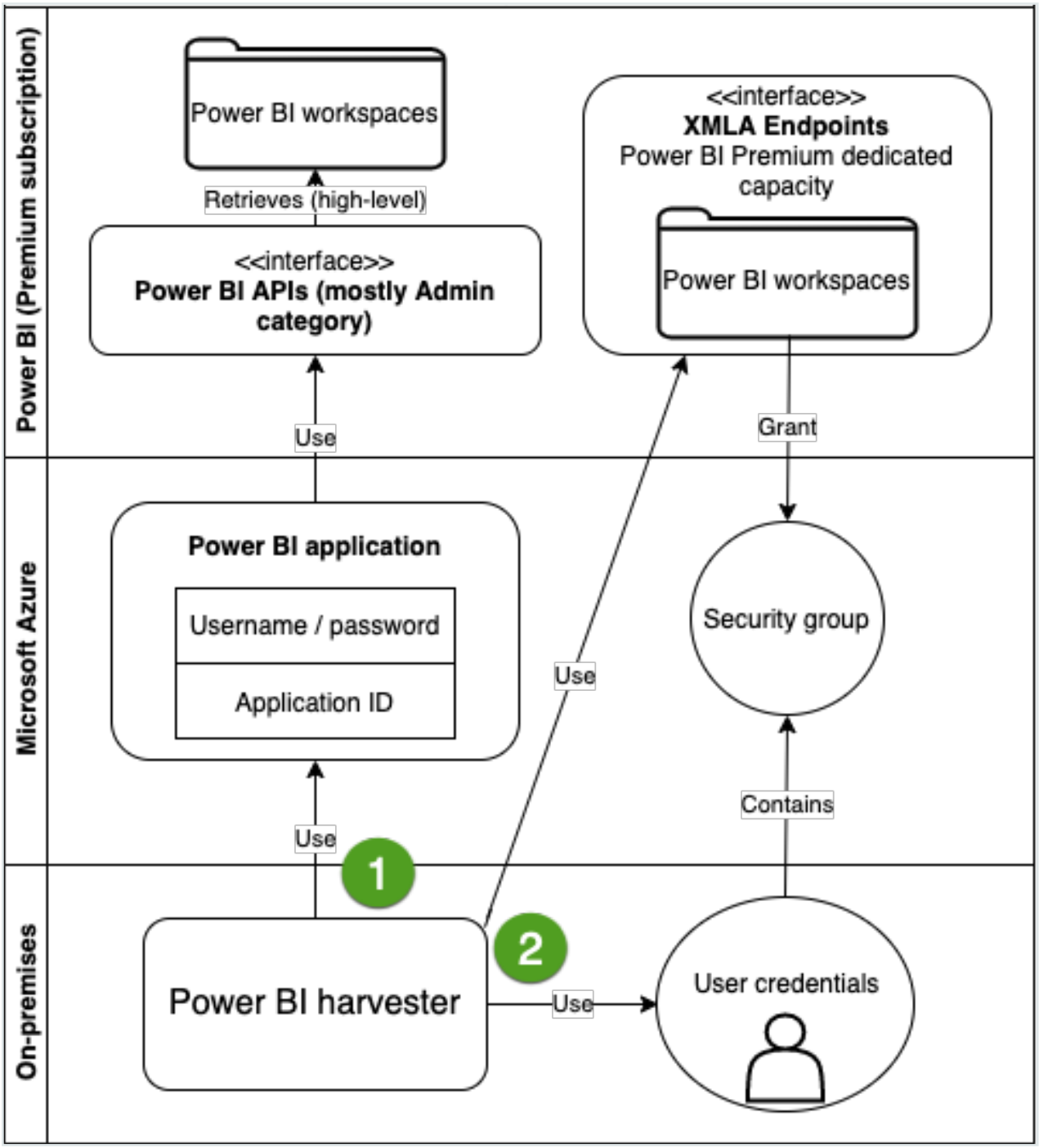
To enable the lineage harvester to access metadata in Power BI workspaces, you must add the workspaces to a Power BI Premium dedicated capacity and have the correct configurations in Microsoft Azure.

> Note There are some limitations to the metadata harvesting process. Ensure that you understand these limitations before you start the harvesting process.

The following table shows which metadata the Power BI harvester retrieves and how.

| Metadata about... | is retrieved using... |
|---|---|
| Reports | Microsoft Azure Admin Power BI REST API calls. |
| Data set columns and lineage | XMLA (Queries or M-queries) endpoints |

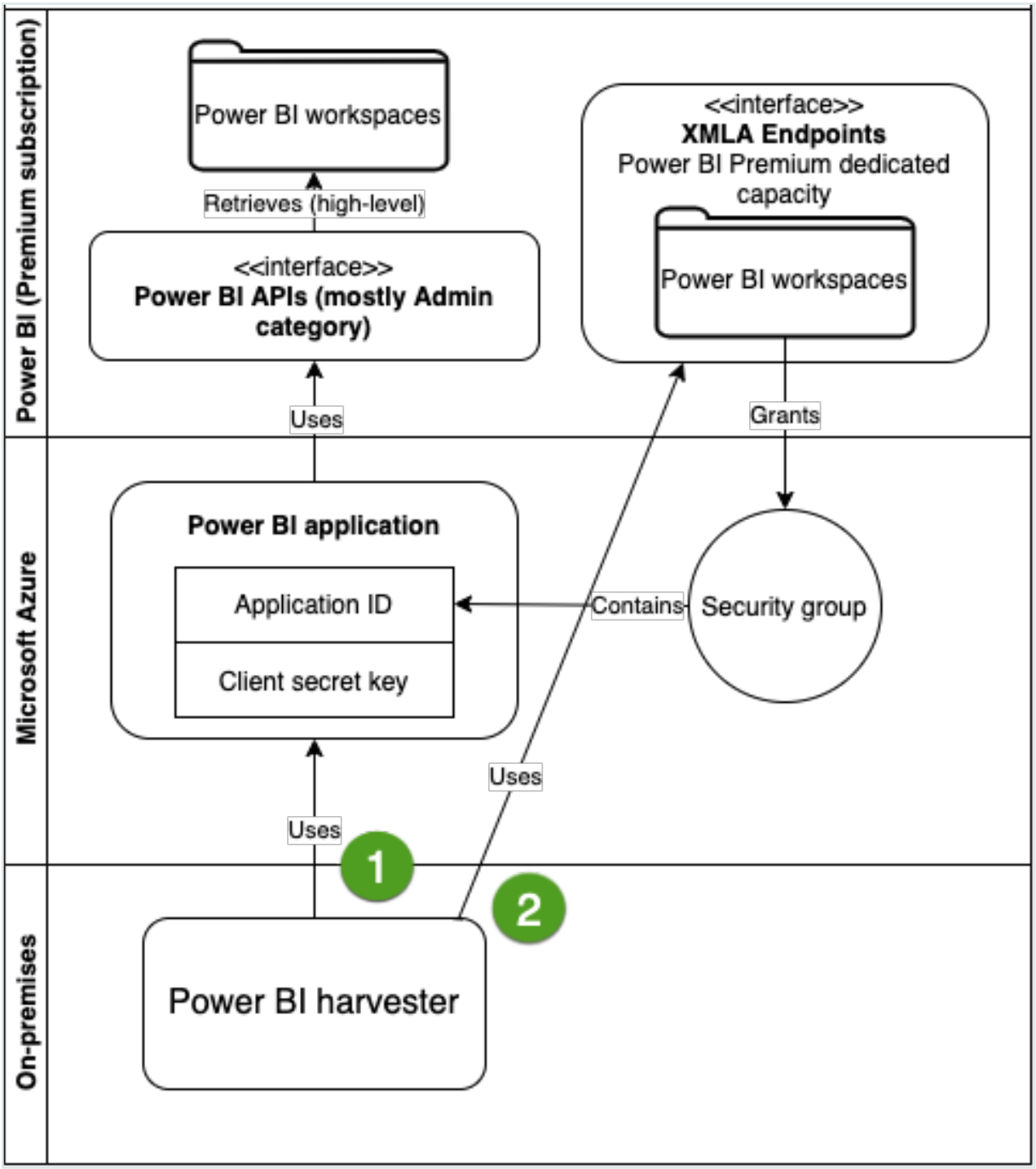
# Overview of the metadata harvesting process with username / password authentication



| Step | Retrieved via | Description |
|---|---|---|
| 1 | Power BI API calls | The Power BI harvester uses the username, password and application ID to access the Power BI APIs. These APIs retrieve basic Power BI metadata, for example metadata in the Power BI tenant or server and reports. |
| 2 | XMLA | You add the Azure Active Directory user with a Power BI admin role in Power BI to a security group and grant him the Contributor role in Power BI workspaces. You add the Power BI workspaces that you want to ingest to the same security group. As a result, the Power BI harvester uses XMLA endpoints to retrieve more specific metadata, for example Power BI columns and lineage. Specific metadata from Power BI workspaces is only harvested if you added the Power BI workspaces to the Power BI dedicated capacity and you have the necessary permissions to harvest the metadata.. |

> Note Make sure that all necessary dedicated capacities are running and accessible to the Power BI harvester. If not, creating assets for Power BI data sets and your technical lineage may fail.

## Overview of the metadata harvesting process with service principal authentication



| Step | Retrieved via | Description |
|---|---|---|
| 1 | Power BI API calls | The Power BI harvester uses the application ID and the client secret key of the Azure Active Directory application to access the Power BI APIs. These APIs retrieve basic Power BI metadata, for example metadata in the Power BI tenant or server and reports. |
| 2 | XMLA | You add the service Principal to a security group and grant it the Contributor role in the Power BI workspaces. As a result, the Power BI harvester uses XMLA endpoints to retrieve more specific metadata, for example in Power BI columns and lineage. Specific metadata from Power BI workspaces is only harvested if you add the Power BI workspaces to the dedicated capacity and you have the necessary permissions to harvest the metadata. |

> Note Make sure that all necessary dedicated capacities are running and accessible to the Power BI harvester. If not, creating assets for Power BI data sets and your technical lineage may fail.

# Prepare a domain for Power BI ingestion

You can create a new domain for your Power BI asset and use the domain ID in the Power BI harvester configuration file. As a result, Collibra uses this domain to ingest all Power BI assets during the Power BI integration process.

## Prerequisites

- You have a resource role with the Domain > Add resource permission.

# Steps

1. In the main menu, click the **Create** (+) button.
   » The **Create** dialog box appears.
2. Click the **Organization** tab.
3. Click a domain type from the list.
   If you clicked the wrong domain type here, you can change it in the **Type** field in the next screen.
   » The **Create Domain** dialog box appears.
4. Enter the required information.

| Field | Description |
|---|---|
| Type | The domain type of the domain you are creating. In this case, you need to select *BI Catalog*. |
| Community | The community under which the domain will be located. |
| Name | The name of the new domain. |

5. Click **Create**.
6. Open your domain.
7. Copy the domain ID.

> Tip If you go to your domain, you can find the domain ID in the URL. The URL looks like: https://<yourcollibrainstance>/domain/**22258f64-40b6-4b16-9c08-c95f8ec0da26**?view=00000000-0000-0000-0000-000000040001. In this example, the domain ID is in bold.

8. Paste the domain ID in the Power BI configuration file.

> Warning When you run the harvesters, Collibra Data Lineage creates all Power BI assets in the same Data Catalog BI domain. We highly recommend that you do not move these assets to another domain. If you move assets to another domain, they will be deleted and recreated in the initial Data Catalog BI domain when you synchronize Power BI. As a consequence, all manually added data of those assets is lost.

# Power BI and lineage harvester set-up

To ingest Power BI metadata in Data Catalog, you need to run two different harvesters:

- The Power BI harvester
- The lineage harvester.

The order in which you run the harvesters is important. You first have to run the Power BI harvester to collect the metadata from your Power BI application and then run the lineage harvester to import new Power BI assets and their relations in Data Catalog. The Power BI ingestion workflow explains which roles the harvesters play in the Power BI ingestion process.

> Note You need to purchase the Power BI metadata connector and lineage feature to access the Power BI and lineage harvesters.

> Warning If you upgrade from Power BI harvester 1.0.0.0 to Power BI harvester 1.0.0.1 or newer, you have to follow an upgrade procedure.

# Power BI ingestion workflow

To ingest Power BI metadata into Data Catalog, you use two types of harvesters:

- A Power BI harvester
- A lineage harvester

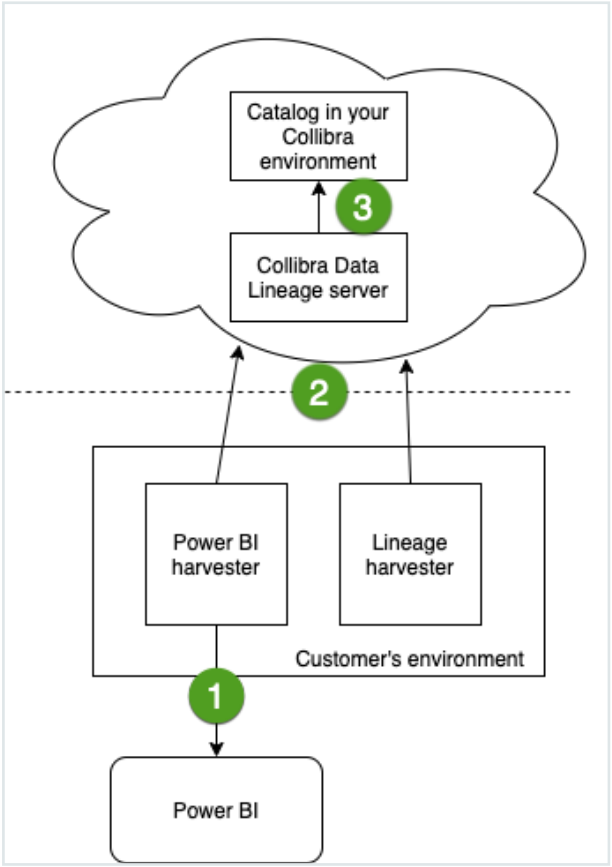
> Note The harvesters can run on the same or on different machines. However, the Power BI harvester must run on a Windows machine.

When the Power BI harvester initiates the Power BI integration, each workflow component performs the following actions:

1. The Power BI harvester:
   - Communicates with Power BI.
   - Harvests Power BI metadata for ingestion and lineage.
   - Sends the Power BI metadata to the Collibra cloud environment.

   > Note The Power BI harvester only harvests the metadata, it does not change it.

2. The lineage harvester:
   - Triggers a new synchronization of the metadata in Collibra to create a technical lineage for Power BI and new relations between Power BI assets.
   - Sends the Power BI ingestion results to Data Catalog.
   - Sends the lineage results to Data Catalog.
3. Data Catalog (via the Collibra Data Lineage server):
   - Shows the new Power BI assets.
   - Shows a Technical lineage tab on Power BI Column pages.



# Collibra Data Lineage servers

A Collibra Data Lineage server processes and analyzes the harvested metadata and uploads it to Data Catalog. Collibra Data Lineage servers never process actual data.

Based on your geographical location and cloud provider, the Power BI harvester sends metadata to one of the following Collibra Data Lineage servers:

- 15.222.200.199 (techlin-aws-ca.collibra.com)
- 18.198.89.106 (techlin-aws-eu.collibra.com)
- 54.242.194.190 (techlin-aws-us.collibra.com)
- 51.105.241.132 (techlin-azure-eu.collibra.com)
- 20.102.44.39 (techlin-azure-us.collibra.com)
- 35.197.182.41 (techlin-gcp-au.collibra.com)
- 34.152.20.240 (techlin-gcp-ca.collibra.com)
- 35.205.146.124 (techlin-gcp-eu.collibra.com)
- 34.87.122.60 (techlin-gcp-sg.collibra.com)
- 35.234.130.150 (techlin-gcp-uk.collibra.com)
- 34.73.33.120 (techlin-gcp-us.collibra.com)

> Important You have to whitelist all Collibra Data Lineage service instances in your geographic location. For example, if your data is located in Europe, you have to whitelist the following Collibra Data Lineage service instances: techlin-aws-eu and techlin-gcp-eu. In addition, we highly recommend that you always whitelist the techlin-aws-us instances as a backup, in case the lineage harvester cannot connect to other Collibra Data Lineage service instances.

# Set up the Power BI harvester

The Power BI harvester is a standalone console application that runs on a Windows machine. You use it to extract data from the Power BI REST API and XMLA endpoints and send it to the Collibra Data Lineage server in Collibra's cloud environment for analysis.

The metadata harvesting process explains in detail which prerequisites you need to enable the Power BI harvester to collect the Power BI metadata.

> Note There are some limitations to the metadata harvesting process. Ensure that you understand these limitations before you start the harvesting process.

# Power BI harvester system requirements

You need to meet the following system requirements to install and run the Power BI harvester on your Windows machine.

> Note If you want to successfully ingest Power BI metadata into Data Catalog, you need to meet both the system requirements to run the Power BI harvester, and also the system requirements to run the lineage harvester.

## Software requirements

You need to meet the software requirements to install and run the Power BI harvester.

### Minimum software requirements

You need the following minimum software requirements:

- Microsoft .NET Framework 4.7.2.
- One of the following:
  - Client operating system: Windows 7 SP1, 8.1 or 10, version 1607.
  - Server operating system: Windows Server 2008 R2 SP1.

> Note .NET Framework 4.7.2 is available as a system update.

### Recommended software requirements

The minimum software requirements are most likely insufficient for production environments. We recommend you meet the following software requirements:

- Microsoft .NET Framework 4.7.2 or higher.
- Client operating system: Windows 10 April 2018 update, version 1803 or newer.
- Server operating system: Windows Server 2016 version 1803 or newer.

## Hardware requirements

You need to meet the hardware requirements to install and run the Power BI harvester.

### Minimum hardware requirements

You need the following minimum hardware requirements:

- 2 GB RAM
- 1 GB free disk space

### Recommended hardware requirements

The minimum requirements are most likely insufficient for production environments. We recommend you meet the following hardware requirements:

- 4 GB RAM
- 20 GB free disk space

## Network requirements

You have firewalls rule to have access to:

- The Microsoft API.
- A Collibra Data Lineage server with IP address:
    - 15.222.200.199 (techlin-aws-ca.collibra.com)
    - 18.198.89.106 (techlin-aws-eu.collibra.com)
    - 54.242.194.190 (techlin-aws-us.collibra.com)
    - 51.105.241.132 (techlin-azure-eu.collibra.com)
    - 20.102.44.39 (techlin-azure-us.collibra.com)
    - 35.197.182.41 (techlin-gcp-au.collibra.com)
    - 34.152.20.240 (techlin-gcp-ca.collibra.com)
    - 35.205.146.124 (techlin-gcp-eu.collibra.com)

- 34.87.122.60 (techlin-gcp-sg.collibra.com)
- 35.234.130.150 (techlin-gcp-uk.collibra.com)
- 34.73.33.120 (techlin-gcp-us.collibra.com)

> Note The Power BI harvester connects to different servers based on your geographic location and cloud provider. If your location or cloud provider changes, the Power BI harvester rescans all your Power BI metadata. You have to whitelist all Collibra Data Lineage servers in your geographic location. In addition, we highly recommend that you always whitelist the techlin-aws-us server as a backup, in case the Power BI harvester cannot connect to other Collibra Data Lineage servers.

> Note The Power BI harvester uses port 443 (tcp only).

## Install the Power BI harvester

Before you can use the Power BI harvester, you need to download it and install it on your Windows machine. You can download the Power BI harvester from the Collibra Product Resource Center downloads page.

> Warning If you upgrade to Power BI harvester 1.0.0.1 or newer, you have to follow an upgrade procedure.

### Prerequisites

- You have Collibra Data Intelligence Cloud 2020.11 or newer.
- You have access to the Power BI harvester on the Downloads page.
- Your environment meets the system requirements to install and use the Power BI harvester.
- You have added Firewall rules so that the Power BI harvester can connect to the Collibra Data Lineage server with one of the following IP addresses:
  - 15.222.200.199 (techlin-aws-ca.collibra.com)
  - 18.198.89.106 (techlin-aws-eu.collibra.com)
  - 54.242.194.190 (techlin-aws-us.collibra.com)
  - 51.105.241.132 (techlin-azure-eu.collibra.com)
  - 20.102.44.39 (techlin-azure-us.collibra.com)

- 35.197.182.41 (techlin-gcp-au.collibra.com)
- 34.152.20.240 (techlin-gcp-ca.collibra.com)
- 35.205.146.124 (techlin-gcp-eu.collibra.com)
- 34.87.122.60 (techlin-gcp-sg.collibra.com)
- 35.234.130.150 (techlin-gcp-uk.collibra.com)
- 34.73.33.120 (techlin-gcp-us.collibra.com)

### Steps

1. Download the Power BI harvester.
2. Unzip the archive.
3. Open the Power BI harvester folder.
   » The Power BI harvester folder contains two folders, a BAT file that you use to run the harvester and a TXT file with information about the configuration file.
   » An empty Power BI configuration file is available in the **config** folder.

### What's next?

You can now prepare the Power BI connection properties in the configuration file and run the Power BI harvester.

> Note We highly recommend that you run the Power BI harvester via command line. This enables you to follow the metadata upload and see possible errors that may occur.

## Prepare the Power BI configuration file

You create a configuration file for the Power BI metadata that you want to ingest. This configuration file is used by the Power BI harvester to retrieve metadata from Power BI and send it to Collibra to be scanned, processed and analyzed.

### Prerequisites

- You have access to the Power BI harvester on the Downloads page.
- You have completed all prerequisite tasks.

- You have a dedicated domain to ingest the Power BI assets.
- You have a global role with the Catalog global permission, for example Catalog Author.
- You have a global role with the Technical lineage global permission.
- You have a global role with the Data Stewardship Manager global permission.
- A resource role with the following resource permission on the community level in which you created the BI Data Catalog domain:
  - Asset: add
  - Attribute: add
  - Domain: add
  - Attachment: add
- Your environment meets the system requirements to run the Power BI harvester and the lineage harvester.

> Tip For a full ingestion, we highly recommend to have a Power BI Premium subscription.

## Steps

1. In the Power BI harvester folder, open the empty configuration file.
2. Enter the values for each property.

| Properties | Description | Mandatory |
|---|---|---|
| powerbi | This section contains information that is necessary to connect to your Power BI application. | Yes |

| Properties | Description | Mandatory |
|---|---|---|
| tenantDomain | The Power BI tenant domain is the domain associated with the Microsoft Azure tenant.<br><br>This domain is either a default domain or a custom domain. For example, *collibrapowerbi.onmicrosoft.com*.<br><br>> Note Usually, you can find a list of Power BI tenant or server domains in your Azure Active Directory or in the top right menu. | Yes |
| applicationId | The unique ID of the Microsoft Azure Application (client) ID. | Yes |
| userName | The username that you use to access Power BI.<br><br>Depending on your authentication type, the username should have a different value:<br><br>◦ For username and password authentication, you enter the username that you use when you sign in to Power BI.<br>◦ For Service Principal authentication, you leave this field empty.<br><br>> Tip If you cannot store your username in the configuration file for security or other reasons, delete this field and provide the username via command line or when prompted by the Power BI harvester. | No |

| Properties | Description | Mandatory |
|---|---|---|
| password | The password or client secret key that you use to access Power BI.<br><br>Depending on your authentication type, the password needs a different value:<br><br>◦ For username and password authentication, you enter the password that you use when you sign in to Power BI.<br>◦ For Service Principal authentication, you enter the Power BI application client secret key.<br>In case the password is an empty string, leave this field empty.<br><br>Tip If you cannot store your password in the configuration file for security or other reasons, delete this field and provide the password via command line or when prompted by the Power BI harvester. | No |

| Properties | Description | Mandatory |
|---|---|---|
| workspaceFilter | An option to exclude specific Power BI workspaces from the ingestion process. You can add multiple workspaces. For example "workspace1, workspace2, workspace3".<br><br>If the workspaceFilter field remains empty or is deleted from the configuration file, all accessible Power BI workspaces are processed and ingested.<br><br>Tip For more information about the query options to filter Power BI workspaces, see the Microsoft documentation. Be aware that the "IN" operator is currently not supported.<br><br>Important If you use Power BI harvester older than version 1.1.0.0, the `workspaceFilter` property is named `groupFilter`. This change is backward compatible. However, if you download a new Power BI harvester, we highly recommend to update your configuration file. | No |
| techlin | This section contains information to identify your Power BI metadata on the Collibra Data Lineage server. | Yes |

| Properties | Description | Mandatory |
|---|---|---|
| sourceId | The unique ID of your Power BI metadata.<br><br>The lineage harvester uses this ID to locate the Power BI metadata on the Collibra Data Lineage server.<br><br>Tip This value can be anything as long as it is a unique, human readable ID and the same as the value of the `Id` property in the lineage harvester configuration file. The Power BI and lineage harvesters use the ID to identify a batch of data on the Collibra Data Lineage server. | Yes |
| catalog | This section contains information that is necessary to connect to Data Catalog. | Yes |
| domainId | The unique resource ID of the domain in Collibra Data Intelligence Cloud in which you want to ingest the Power BI assets.<br><br>Tip You can find the domain ID by clicking the domain type. Then look in the URL of your browser to find the ID. The URL looks like https://<yourcollibrainstance>/domain/<domain ID>?<view>. | Yes |

| Properties | Description | Mandatory |
|---|---|---|
| url | The URL of your Collibra Data Intelligence Cloud instance.<br><br>Note You can only enter the public URL of your Collibra Data Intelligence Cloud environment. Other URLs will not be accepted. | Yes |
| userName | The username that you use to sign in to Collibra.<br><br>Tip If you cannot store your username in the configuration file for security or other reasons, delete this field and provide the username via command line or when prompted by the Power BI harvester. | No |
| password | The password that you use to sign in to Collibra.<br><br>Tip If you cannot store your passwordin the configuration file for security or other reasons, delete this field and provide the password via command line or when prompted by the Power BI harvester. | No |

| Properties | Description | Mandatory |
|---|---|---|
| useCollibraSystemName | Indication whether you want to use the system or server name of a data source to match to the System asset in Data Catalog during automatic stitching. This is useful when you have multiple databases with the same name.<br><br>By default, the useCollibraSystemName property is set to `false`. If you want to use it, set it to `true`.<br><br>◦ If you set the useCollibraSystemName property to `true`, the Power BI harvester reads the <source-ID> configuration file and takes the value in the collibraSystemName property into account.<br>◦ If you set the useCollibraSystemName property to `false`, the Power BI harvester ignores the collibraSystemName property in the <source-ID> configuration file.<br><br>Warning Unless you have multiple databases with the same name, we highly recommend that you keep the default value. | Yes |

3. Save the configuration file.
4. Trigger the Power BI harvester to upload the Power BI metadata:
   - Run the following command line if your configuration file is in its default location: `.\powerbi-harvester.bat`
   - Launch the path to the Power BI configuration file if you moved the configuration file to a different location: `.\bin\powerbi-harvester.exe .\config\powerbi-harvester.conf`

> Note We highly recommend that you run the Power BI harvester via command line. This enables you to follow the metadata upload and see possible errors that may occur.

5. If the Power BI harvester prompts for credentials, enter them or use command line options to provide them.

   > Note Credentials provided via command line overwrite the credentials in the configuration file.

   » The Power BI harvester collects the Power BI metadata and sends it to the Collibra Data Lineage server. Collibra scans and analyzes the metadata.

> Tip If you want to ingest multiple Power BI applications, create a new configuration file using a unique ID and repeat these steps. In the lineage harvester configuration file, you can add multiple Power BI sections that each refer to a different ID.

> Note If you are not able to run the Power BI harvester, go to the troubleshooting section to resolve your issues.

## Example

This example shows a configuration file with the username / password authentication method.

```
{
 "powerbi": {
  "tenantDomain": "<organization.onmicrosoft.com>",
  "applicationId": "<microsoft-azure-id>",
  "userName": "<your-power-bi-email-address>",
  "password": "<password-to-access-power-bi>",
  "workspaceFilter": "workspace-name1", "workspace-name2"
 },
 "techlin": {
  "sourceId" : "<unique-power-bi-ID>"
  },
 "catalog": {
  "domainId": "<your-catalog-domain>",
  "url": "<url-to-collibra>",
```

```
  "userName": "<my-collibra-username>",
  "password": "<my-collibra-password>"
 },
 "useCollibraSystemName": false
}
```

This example shows a configuration file with the service principal authentication method.

```
{
 "powerbi": {
  "tenantDomain": "<organization.onmicrosoft.com>",
  "applicationId": "<microsoft-azure-id>",
  "userName": "",
  "password": "<secret-key>",
  "workspaceFilter": "<filter-workspace-name>"
 },
 "techlin": {
  "sourceId" : "<unique-power-bi-ID>"
  },
 "catalog": {
  "domainId": "<your-catalog-domain>",
  "url": "<url-to-collibra>",
  "userName": "<my-collibra-username>",
  "password": "<my-collibra-password>"
 },
 "useCollibraSystemName": false
}
```

> Warning If you are ingesting a large amount of Power BI data and you use the workspace filter (workspaceFilter), the Power BI harvester might time out, resulting in an Internal Server Error. If you get this error, we highly advise you to not use the workspace filter. See the known issues in Power BI ingestion limitations.

## What's next?

You can now download and install the lineage harvester and prepare the lineage harvester configuration file. The lineage harvester triggers Collibra to create new Power BI assets, stitch them and show a technical lineage for them.

To refresh the Power BI metadata in Data Catalog, you can run the Power BI harvester and lineage harvester again or schedule jobs to run them automatically.

# Prepare Power BI <source ID> configuration file

The Power BI harvester uses a Power BI configuration file to collect the Power BI data objects. It then sends the metadata to the Collibra Data Lineage server. However, if the `useCollibraSystemName` in the Power BI configuration file is set to `true`, you also have to provide a specific <source ID> configuration file that defines the system name of databases in Power BI.

Collibra Data Lineage uses the system names to match the structure of databases in Power BI to assets in Data Catalog.

> Tip The name "<source ID>" refers to the value of the `sourceId` property in the Power BI configuration file.

## Prerequisites

- The `useCollibraSystemName` in the Power BI harvester configuration file is set to `true`.

> Note This is not a prerequisite if you are using a <source ID> configuration file for the purpose of providing the true system names of the ODBC databases in Power BI. In that case, you can set the `useCollibraSystemName` property in the Power BI harvester configuration file to `true`, but it is not mandatory.

## Steps

1. Create a new JSON file in the Power BI harvester **config** folder.
2. Give the JSON file the same name as the value of the `sourceId` property in the Power BI configuration file.

   > Example The value of the `sourceId` property in the Power BI configuration file is `power-bi-source-1`. Therefore, the name of your JSON file should be *power-bi-source-1.conf*.

3. For each database in Power BI, add the following content to the JSON file:

<table>
<tr><th>Property</th><th>Description</th><th>Mandatory?</th></tr>
<tr><td>found_dbname=<database name>;found_hostname=<server name></td><td>The database information of supported data sources in Power BI that is typically collected by the Power BI harvester. It describes on which server the database is running (<code>found_hostname</code>) and what the name of the database is (<code>found_dbname</code>).<br><br>Tip<br>You can use wildcards to capture multiple connection string combinations:<br><br>Show me the supported wildcards<table><tr><th>Pattern</th><th>Description</th></tr><tr><td>*</td><td>Matches everything.</td></tr><tr><td>?</td><td>Matches any single character.</td></tr><tr><td>[seq]</td><td>Matches any character in "seq".</td></tr><tr><td>[!seq]</td><td>Matches any character not in "seq".</td></tr></table></td><td>Yes</td></tr>
<tr><td>dbname</td><td>The name of the database of a supported data source in Power BI.</td><td>No</td></tr>
</table>

| Property | Description | Mandatory? |
| --- | --- | --- |
| schema | The name of the default schema of a supported data source in Power BI.<br><br>If the Power BI harvester fails to find a specific schema, it uses the default schema. | No |
| dialect | The dialect of the supported data source in Power BI.<br><br>Tip<br>You can enter one of the following values:<br><br>◦ *azure*, for an Azure SQL Server data source.<br>◦ *bigquery*, for a Google BigQuery data source.<br>◦ *mssql*, for a Microsoft SQL Server data source.<br>◦ *oracle*, for an Oracle data source.<br>◦ *redshift*, for an Amazon Redshift data source.<br>◦ *snowflake*, for a Snowflake data source.<br>◦ *sybase*, for a Sybase data source. | No |

| Property | Description | Mandatory? |
| --- | --- | --- |
| collibraSystemName | The system or server name of a database.<br><br>**Warning** The value of this property must exactly match the name of your System asset in Collibra.<br><br>**Important** If you are using a <source> configuration file for the purpose of providing the true system name of an ODBC database in Power BI, you are not required to:<br>◦ Set the useCollibraSystemName property in the Power BI configuration file to `true`.<br>◦ Specify a Collibra system name in the <source ID> configuration file.<br>However, if the useCollibraSystemName property is set to `true` in the Power BI configuration file, then you must specify a Collibra system name in the <source ID> configuration file. | Yes<br><br>(unless you are using a <source ID> file to provide the true system names of ODBC databases in Power BI.) |

4. Save the <source ID> configuration file.

## Example of the <source ID>.conf file

```
{
   "found_dbname=databasename1;found_hostname=*": {
       "dbname": "mssql-database-name",
       "schema": "mssql-schema-name",
```

```
        "dialect": "mssql",
        "collibraSystemName": "mssql-system-name"
    },
    "found_dbname=databasename2;found_hostname=server-name.on-
microsoft.com": {
        "dbname": "oracle-database-name",
        "schema": "oracle-schema-name",
        "dialect": "oracle",
        "collibraSystemName": "oracle-system-name"
    }
}
```

# Ingest multiple Power BI applications

You can ingest more than one Power BI application in Collibra. For each Power BI application, you create a separate Power BI configuration file, and then add a section in the lineage harvester configuration file.

## Prerequisites

- You have access to the Power BI harvester on the Downloads page.
- You have completed all prerequisite tasks.
- You have a dedicated domain to ingest the Power BI assets.
- You have a global role with the Catalog global permission, for example Catalog Author.
- You have a global role with the Technical lineage global permission.
- You have a global role with the Data Stewardship Manager global permission.
- A resource role with the following resource permission on the community level in which you created the BI Data Catalog domain:
  - Asset: add
  - Attribute: add
  - Domain: add
  - Attachment: add
- Your environment meets the system requirements to run the Power BI harvester and the lineage harvester.

> Tip For a full ingestion, we highly recommend to have a Power BI Premium subscription.

## Steps

1. Prepare the Power BI configuration file for one Power BI application.
2. Run the Power BI harvester.
3. For each additional Power BI application, do the following:
   a. Prepare a new configuration file with the information of the next Power BI application.
      i. Optionally, create a new domain in Data Catalog to ingest the assets of this Power BI application.
      ii. Enter a new source ID that is different from the source IDs of existing Power BI configuration files.
   b. Run the Power BI harvester again.

   > Note Make sure that you refer to the path of this configuration file when you run the Power BI harvester.

   » The Power BI harvester collects the Power BI metadata of each Power BI application and sends it to the Collibra Data Lineage server.
   » Collibra scans and analyzes the metadata.
4. In the lineage harvester configuration file, create a Power BI section for each Power BI application. Use the source ID of each Power BI configuration file as the ID of the Power BI section in the lineage harvester configuration file.
5. Run the lineage harvester to ingest the Power BI metadata in Collibra.
   » The Power BI metadata is ingested in the domain that you specified in the Power BI configuration file.

## Example

You have two Power BI applications that you want to ingest. The first Power BI configuration file has source ID `power-bi-app-a`, the second Power BI configuration file has source ID `power-bi-app-b`. The lineage harvester configuration file contains two Power BI sections that each refer to a different source ID.

```
{
   "general": {
       "catalog" : {
           "url" : "https://companydomain.collibra.com",
           "username" : "my-Collibra-username"}
```

```
    },
    "sources" : [
     {
        "type" : "ExistingLineage",
        "id" : "power-bi-app-a"
    }
    {
        "type" : "ExistingLineage",
        "id" : "power-bi-app-b"
    }]
}
```

## What's next?

To refresh the Power BI metadata in Data Catalog, you can run the Power BI harvester and lineage harvester again or schedule jobs to run them automatically. You can schedule to synchronize Power BI applications at different times.

# Command options and arguments

After creating a Power BI harvester configuration file, you can use the command line to provide the Power BI harvester with additional information or perform specific actions.

> Note Credentials provided via command line overwrite the credentials in the configuration file.

## Typical command options and arguments

The following table shows the most commonly used command options and arguments.

| Command | Description |
| --- | --- |
| `--catalog-password "<Collibra password>"` | Your Collibra password.<br><br>If you don't want to add your password in the Power BI harvester configuration file, you can provide it via command line.<br><br>> Note If you added an API key, the Data Catalog credentials will not be used. |
| `--catalog-user"<Collibra username>"` | Your Collibra username.<br><br>If you don't want to add your password in the Power BI harvesterconfiguration file, you can provide it via command line.<br><br>> Note If you added an API key, the Data Catalog credentials will not be used. |
| `--output-file <file path>` | Save your harvested Power BI metadata to a specified directory. The output file is a ZIP file. |
| `--from-file <file path>` | Upload Power BI metadata that was already harvested and saved to a specified file. |
| `--timeout <seconds>` | Increase the timeout duration to specify a longer timeout for remote API calls.<br><br>> Example If you want the Power BI harvester to wait 15 minutes before canceling a remote API connection, you can use `--timeout 900`. |

# Set up the lineage harvester for Power BI ingestion

The lineage harvester is a software application that is needed to collect your Power BI metadata and send it to the Collibra Data Lineage server, where the metadata is processed and a technical lineage and new Power BI assets and relations are created. Collibra Data Intelligence Cloud then import those assets and relations into Data Catalog.

For more information about the lineage harvester, read the Collibra Data Lineage documentation.

> Note You need the lineage harvester 1.2.1 or newer to ingest Power BI metadata into Data Catalog.

## Lineage harvester system requirements

You need to meet the system requirements to be able to install and run the lineage harvester.

### Software requirements

Java Runtime Environment version 11 or newer, or OpenJDK 11 or newer.

For Java Runtime Environment 16 or newer, or OpenJDK 16 or newer, set the `JAVA_OPTS` environment variable for the lineage harvester to function properly:

```
JAVA_OPTS='--illegal-access=deny'
```

> Note To ingest Snowflake data sources, the minimum requirement is Java Runtime Environment version 16 or newer, or OpenJDK 16 or newer.

### Hardware requirements

You need to meet the hardware requirements to install and run the lineage harvester.

### Minimum hardware requirements

You need the following minimum hardware requirements:

- 2 GB RAM
- 1 GB free disk space

### Recommended hardware requirements

The minimum requirements are most likely insufficient for production environments. We recommend the following hardware requirements:

- 4 GB RAM

  > Tip 4 GB RAM is sufficient in most cases, but more memory could be needed for larger harvesting tasks. For instructions on how to increase the maximum heap size, see Technical lineage general troubleshooting.

- 20 GB free disk space

## Network requirements

The lineage harvester uses the HTTPS protocol by default and uses port 443.

You need the following minimum network requirements:

- Firewall rules so that the lineage harvester can connect to:
  - Collibra Data Intelligence Cloud.
  - All Collibra Data Lineage service instances in your geographic location:
    - 15.222.200.199 (techlin-aws-ca.collibra.com)
    - 18.198.89.106 (techlin-aws-eu.collibra.com)
    - 54.242.194.190 (techlin-aws-us.collibra.com)
    - 51.105.241.132 (techlin-azure-eu.collibra.com)
    - 20.102.44.39 (techlin-azure-us.collibra.com)
    - 35.197.182.41 (techlin-gcp-au.collibra.com)
    - 34.152.20.240 (techlin-gcp-ca.collibra.com)
    - 35.205.146.124 (techlin-gcp-eu.collibra.com)
    - 34.87.122.60 (techlin-gcp-sg.collibra.com)

- 35.234.130.150 (techlin-gcp-uk.collibra.com)
- 34.73.33.120 (techlin-gcp-us.collibra.com)

> Note The Power BI harvester connects to different Collibra Data Lineage service instances based on your geographic location and cloud provider. If your location or cloud provider changes, the Power BI harvester rescans all your Power BI metadata. You have to whitelist all Collibra Data Lineage service instances in your geographic location. In addition, we highly recommend that you always whitelist the techlin-aws-us instance as a backup, in case the Power BI harvester cannot connect to other Collibra Data Lineage service instances.

# Install the lineage harvester for Power BI ingestion

Before you can use the lineage harvester, you need to download it and install it. You can download the lineage harvester from the Collibra Community downloads page.

> Tip
> - Install the lineage harvester close to your data source or on the same server.
> - The lineage harvester uses port 443.

## Prerequisites

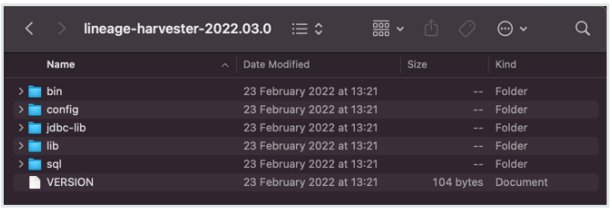
- You have purchased the Power BI metadata connector and lineage feature.
- You have Collibra Data Intelligence Cloud 2020.11 or newer.
- You meet the minimum system requirements.
- You have added Firewall rules so that the lineage harvester can connect to:
  - The host names of all databases in the lineage harvester configuration file.
  - All Collibra Data Lineage service instances within your geographical location:
    - 15.222.200.199 (techlin-aws-ca.collibra.com)
    - 18.198.89.106 (techlin-aws-eu.collibra.com)
    - 54.242.194.190 (techlin-aws-us.collibra.com)
    - 51.105.241.132 (techlin-azure-eu.collibra.com)
    - 20.102.44.39 (techlin-azure-us.collibra.com)
    - 35.197.182.41 (techlin-gcp-au.collibra.com)
    - 34.152.20.240 (techlin-gcp-ca.collibra.com)
    - 35.205.146.124 (techlin-gcp-eu.collibra.com)

- 34.87.122.60 (techlin-gcp-sg.collibra.com)
- 35.234.130.150 (techlin-gcp-uk.collibra.com)
- 34.73.33.120 (techlin-gcp-us.collibra.com)

> Note The lineage harvester connects to different instances based on your geographic location and cloud provider. If your location or cloud provider changes, the lineage harvester rescans all your data sources. You have to whitelist all Collibra Data Lineage service instances in your geographic location. In addition, we highly recommend that you always whitelist the techlin-aws-us instance as a backup, in case the lineage harvester cannot connect to other Collibra Data Lineage service instances.

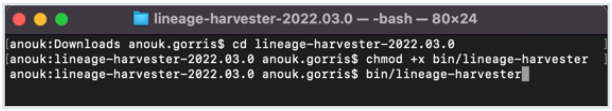
## Steps

1. Download the newest lineage harvester.
2. Unzip the archive.
   » You can now access the lineage harvester folder.

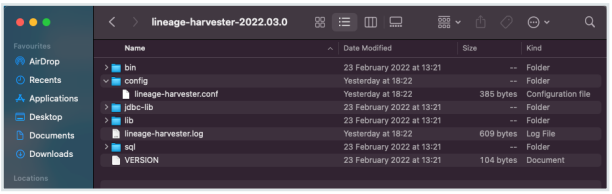



3. Run the following command line to start the lineage harvester:
   - Windows: `.\bin\lineage-harvester.bat`


   - For other operating systems: `chmod +x bin/lineage-harvester` and then `bin/lineage-harvester`



» An empty configuration file is created in the config folder.




» The lineage harvester is installed automatically. You can check the installation by running `./bin/lineage-harvester --help`.

## What's next?

You can now prepare the lineage harvester configuration file.

# Prepare the lineage harvester configuration file for Power BI

You have to prepare a technical lineage configuration file and run the lineage harvester to fetch the Power BI analysis results on the Collibra Data Lineage server and sent them as an import job to your Collibra Data Intelligence Cloud.

> Note Comments in the lineage harvester configuration file are not supported.

> Tip For more information, see Collibra Data Lineage.

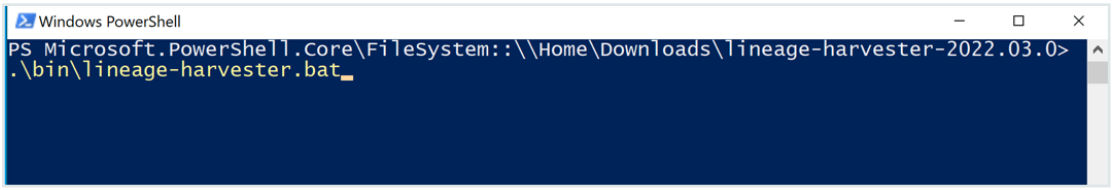
## Prerequisites

- You have prepared the Power BI configuration file and executed the Power BI harvester.
- You have a global role that has the Manage all resources global permission.
- You have a global role with the Catalog global permission, for example Catalog Author.
- You have the Technical lineage global permission.
- You have created a BI Catalog domain in which you want to ingest the Power BI assets.
- A resource role with the following resource permission on the community level in which you created the BI Data Catalog domain:

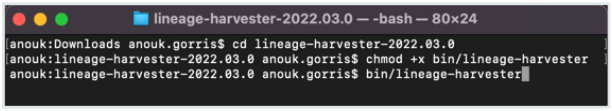
    - Asset: add
    - Attribute: add
    - Domain: add
    - Attachment: add
- You have downloaded lineage harvester version 2022.05 or newer. We highly recommend that you always install and use the newest lineage harvester.

## Steps

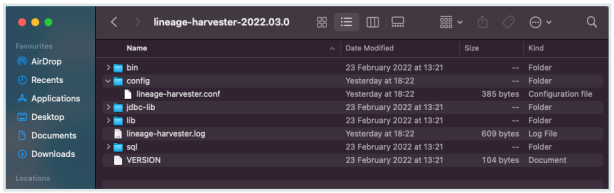
1. Run the following command line to start the lineage harvester:
    - Windows: `.\bin\lineage-harvester.bat`




    - For other operating systems: `chmod +x bin/lineage-harvester` and then `bin/lineage-harvester`




    » An empty configuration file is created in the config folder.




2. Open the configuration file and enter the values for each property.

| Properties | Description |
|---|---|
| general | This section describes the connection information between the lineage harvester and Data Catalog. |
| catalog | This section contains information that is necessary to connect to Data Catalog. |

| Properties | Description |
| --- | --- |
| url | The URL of your Collibra Data Intelligence Cloud environment.<br><br>Note You can only enter the public URL of your Collibra DGC environment. Other URLs will not be accepted. |
| username | The username that you use to sign in to Collibra. |
| sources | This section describes the data sources for which you want to create the technical lineage. You have to create a configuration section for each data source.<br><br>Note You can add multiple data sources to the same configuration file. |
| type | The kind of data source. In this case, the value has to be *ExistingLineage*. |
| id | The unique ID to identify the Power BI service metadata that was uploaded to the Collibra Data Lineage server. The value has to be the same as the value you used in the `sourceId` property in the Power BI configuration file.<br><br>Tip This value can be anything as long as it is a unique ID and the same as the value of the `sourceId` property in the Power BI configuration file. The Power BI and lineage harvesters use the ID to identify a batch of data on the Collibra Data Lineage server. |

> Tip If you want to ingest multiple Power BI applications, create a separate Power BI configuration file for each Power BI application each with a unique source ID. Duplicate the Power BI section in the lineage harvester configuration file and enter the source ID in the ID property.

3. Save the configuration file.
4. Start the lineage harvester again in the console and run the following command:
   - for Windows: `.\bin\lineage-harvester.bat full-sync`
   - for other operating systems: `./bin/lineage-harvester full-sync`
5. When prompted, enter the passwords to connect to your Collibra Data Intelligence Cloud environment.
   » The password is encrypted and stored in **/config/pwd.conf**

## What's next?

The lineage harvester triggers Collibra to import Power BI assets and their relations and create a technical lineage for Power BI Column assets. Collibra also stitches the new Power BI assets to existing assets in Data Catalog.

To refresh the Power BI metadata in Data Catalog, you can run the Power BI harvester and lineage harvester again or schedule jobs to run them automatically.

> Tip You can check the progress of the Power BI ingestion and technical lineage creation in Activities. The **Results** field indicates how many relations were imported into Data Catalog.

> Warning When you run the harvesters, Collibra Data Lineage creates all Power BI assets in the same Data Catalog BI domain. We highly recommend that you do not move these assets to another domain. If you move assets to another domain, they will be deleted and recreated in the initial Data Catalog BI domain when you synchronize Power BI. As a consequence, all manually added data of those assets is lost.

# Power BI business logic

Power BI business users work with Power BI dashboards and reports to make business decisions. Collibra's Power BI connector and lineage feature offers business users several advantages:

- Easily find certified Power BI content.
- Shop for Power BI reports.
- Trace Power BI metadata to metadata of other data sources.
- Find where content is stored in Power BI.
- Get information about a Power BI Report in a single location.

# Power BI asset pages

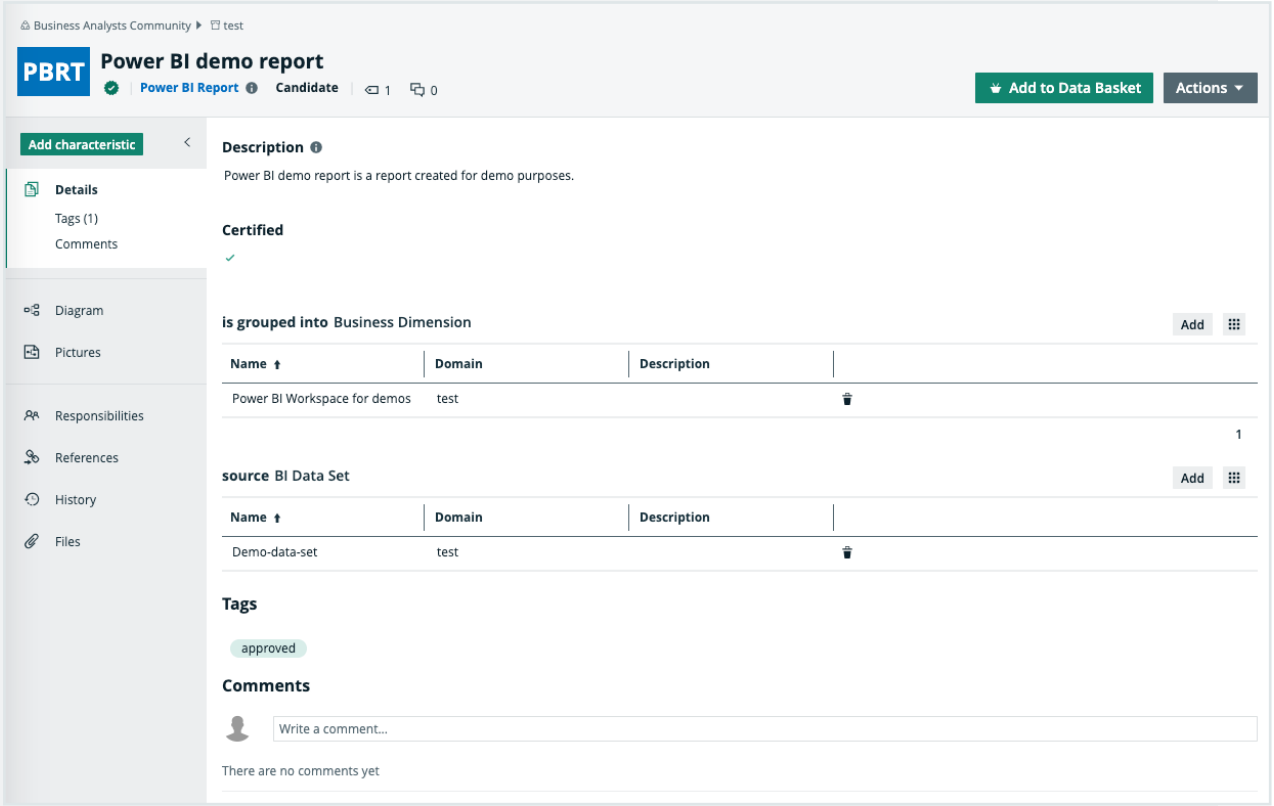
Depending on the Power BI asset type, the asset page shows different information ingested from Power BI. You can find a specific Power BI asset page using Data Catalog search or via the Data Catalog BI domain in which you ingested the Power BI metadata.

## Details

Asset pages show attributes and relations to other assets. This information is synchronized with the Power BI service. However, you can add additional characteristics, tags or comments.

If you want to use a Power BI Data Model or a Power BI Report, you can add it to the Data Basket and check it out.

Example The following Power BI Report asset page shows in which Power BI Workspace the report is stored and which Power BI Data Set it uses. This asset has a clear description and is certified.

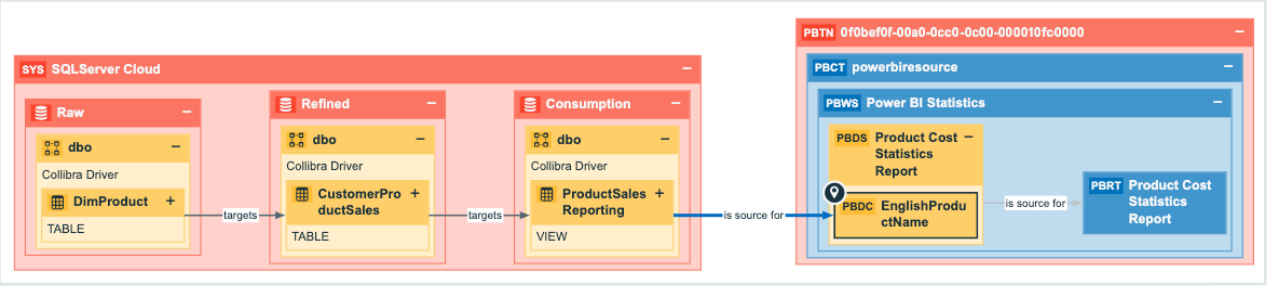



# Business diagrams

The business diagram is a feature to show and interact with many assets and relations in an easy-to-read diagram. The business diagram helps you to quickly see to which other assets a specific asset is related. As such, the diagram can show a high-level presentation of a Power BI Report. This enables you, for example, to see:

- In which Power BI Workspace the Power BI Report is stored.
- In which Power BI Capacity the Power BI Workspace is stored.
- Which Power BI Data Model assets the Power BI Report uses.
- Which Table assets and Column assets from other data sources are the source of a Power BI Column asset.

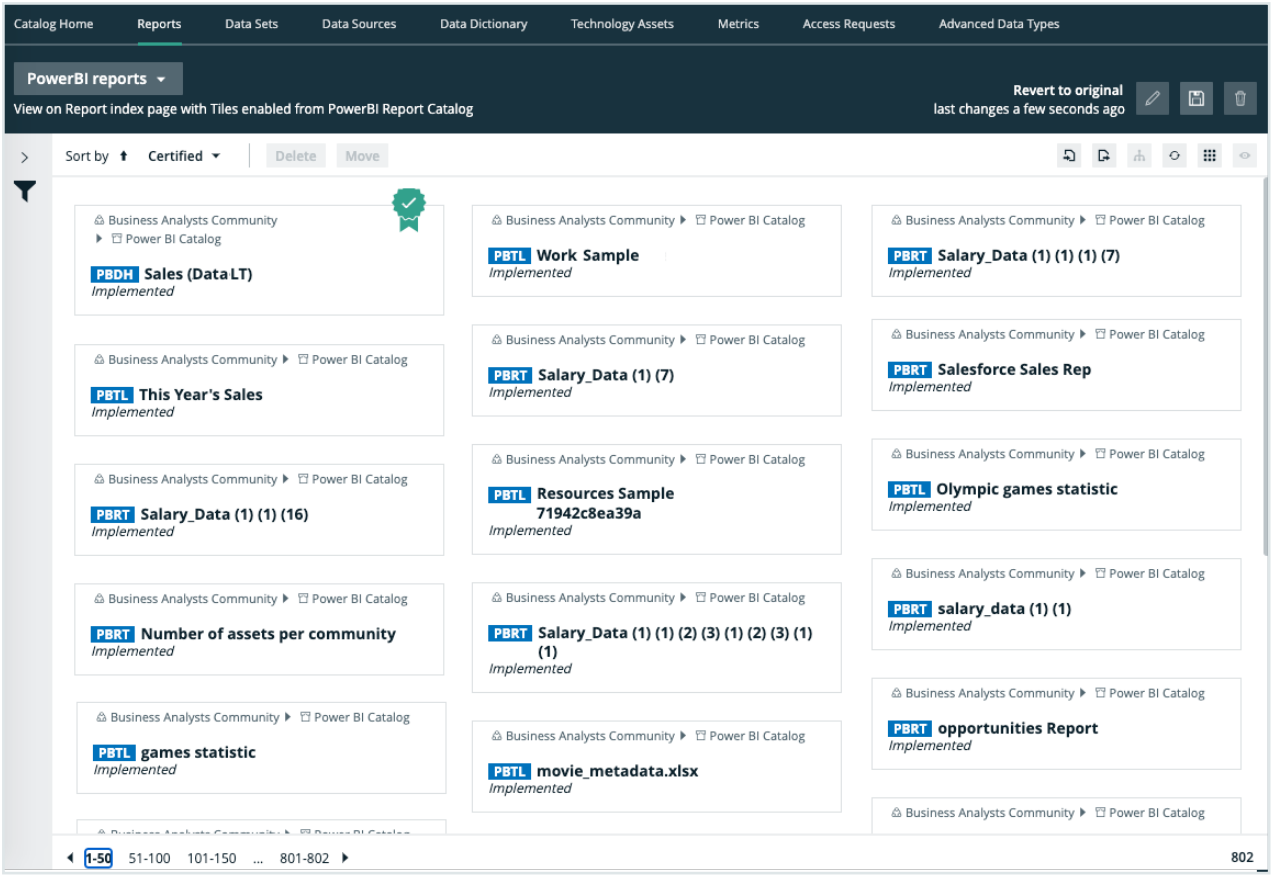
Example The following business diagram shows the *Product Cost Statistics Report* Power BI Report, which is stored in the *Power BI Statistics* Power BI Workspace. The report uses the *Product Cost - Statistics Report* Power BI Data Model. This data set contains data from the *SQL Server Cloud* source.




# Report views

The Power BI connector and lineage feature enables you to find all ingested Power BI Reports and children of the Power BI Report asset type in a single location.

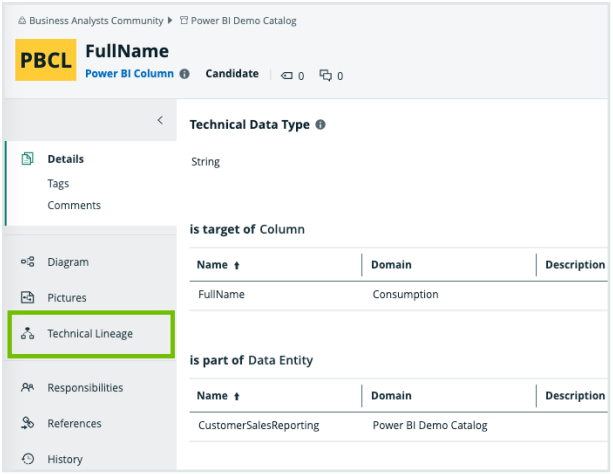
In the **Reports** tab page in Data Catalog you can see an overview of all Report assets and their children. Optionally, you can create a view with a filter to only show Power BI Reports. This is useful if you quickly want to find a report or if you want to know which reports are certified.




# Technical lineage for Power BI service

When you ingest Power BI metadata in Data Catalog, you automatically create a technical lineage for Power BI Column assets. Each Power BI Column asset page has a Technical lineage tab page that shows the technical lineage of that Power BI Column asset.

> Note If you ingest Power BI for the first time or if you change your geolocation or cloud provider, you have to restart the DGC service before you can see your technical lineage.



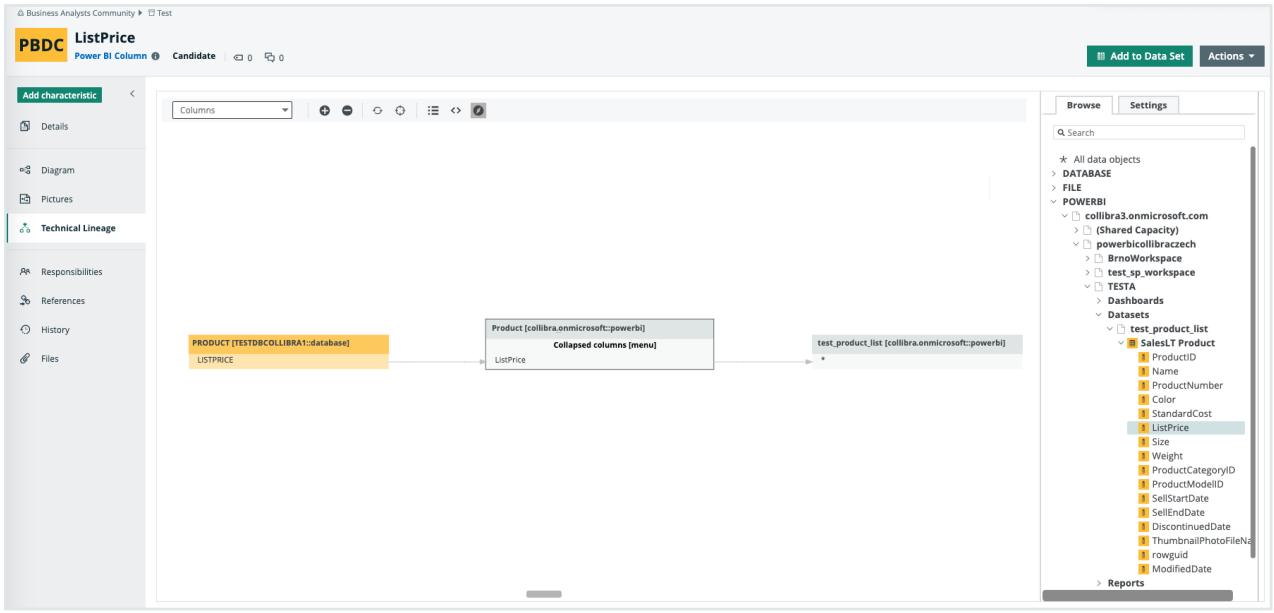

# Technical lineage graph

The technical lineage graph shows relations of the type "Data Element targets / sources Data Element" between BI assets and other data objects in the data flow, for example Column assets or Power BI Column assets. These relations are created during the Power BI ingestion process as a result of automatic stitching.

For more information about the technical lineage, see the Collibra Data Lineage section in the user guide.

## Example

The following technical lineage shows the relation of the type "Data Element targets / sources Data Element" between the Column asset *LISTPRICE* and the Power BI Column asset *ListPrice*.



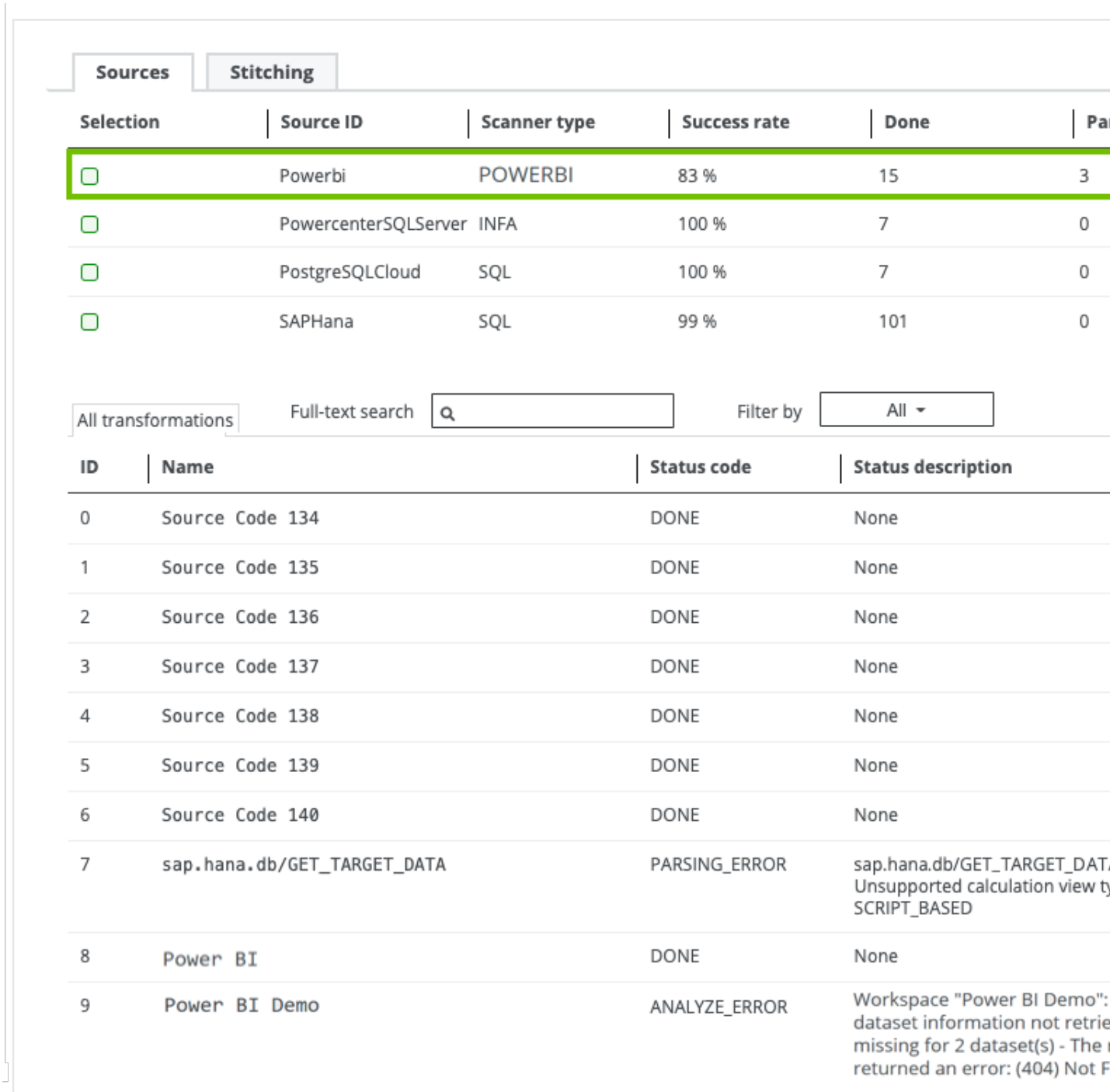

# Sources tab page

The Sources tab page shows the transformation details that were analyzed and processed on the Collibra Data Lineage server and the results of this analysis. The success rate of the analysis indicates how complete the technical lineage is. There are a few limitations that prevent the Collibra Data Lineage server from processing all Power BI metadata.

> Important The Collibra Data Lineage server can process most, but not all, complex Power BI metadata. This means that the success rate of a Power BI ingestion can be very high, but almost never 100%.

## Example

The following Sources tab page shows that you have created a technical lineage for four data sources. Power BI has a success rate of 83%. When you use the transformation logs to investigate the errors, you see that the Collibra Data Lineage server couldn't process some elements of the Power BI metadata, for example because they are not supported or there is an issue in the configuration file or the Power BI setup.




# Automatic stitching

Stitching is a process that creates relations between database columns that are Column assets in Collibra Data Intelligence Cloud and BI assets representing the same database,

specifically between:

- The assets that are created when you ingest Power BI.
- The assets that are created when you register a data source.

The Power BI Harvester collects the Power BI source code and sends it to Collibra for analyzing. The lineage harvester then pushes it to the Data Catalog and creates the relation between Power BI assets in Data Catalog.

At the same time, Collibra analyzes other metadata of data sources that you registered in Data Catalog and creates new relations of the type "Data Element targets / sources Data Element" between Power BI Column assets and Column assets in Data Catalog. It also creates a data flow between data objects, which is visualized in a technical lineage.

> Note When you ingest Power BI, you automatically create a technical lineage for Power BI Column assets.

# Stitching issues

To stitch assets in Data Catalog to data object collected by the lineage harvester, the Collibra Data Lineage server looks at the full path of the assets in Data Catalog and the full path of Power BI assets. If the full paths match, the Collibra Data Lineage automatically stitches them.

> Tip You can use the Stitching tab page to easily find the full path of assets in Data Catalog and data objects that were collected by the Power BI harvester and the lineage harvester.

# Schedule jobs

You can use scheduled jobs to run the Power BI harvester and lineage harvester at specific times automatically.

Since you need both the Power BI harvester and the lineage harvester to successfully ingest Power BI metadata in Data Catalog, we highly recommend that you schedule the Power BI harvester job before you schedule the lineage harvester job.

Warning When you run the harvesters, Collibra Data Lineage creates all Power BI assets in the same Data Catalog BI domain. We highly recommend that you do not move these assets to another domain. If you move assets to another domain, they will be deleted and recreated in the initial Data Catalog BI domain when you synchronize Power BI. As a consequence, all manually added data of those assets is lost.

# Schedule Power BI harvester jobs

You can use the Windows Task Scheduler to make the Power BI harvester run scheduled jobs periodically. In a scheduled job, the Power BI harvester automatically uploads Power BI metadata to Collibra.

Scheduled jobs only work if you add the correct credentials to the Power BI configuration file or if you use a tool to automatically provide the credentials each time the Power BI harvester job is scheduled.

# Schedule lineage harvester jobs

You can use Task Scheduler on Windows or Crontab on Mac and Linux to make the lineage harvester run scheduled jobs periodically. In a scheduled job, the lineage harvester uploads Power BI metadata to your Collibra Data Intelligence Cloud environment and Data Catalog automatically creates new Power BI assets and relations at specific times, dates or intervals. Collibra also creates a technical lineage for Power BI Column assets.

Warning Relations that were manually created between Power BI assets and other assets via a relation type in the Power BI operating model, are deleted after a refresh of the Power BI metadata.

Example You created a Power BI configuration file and added the required properties to the lineage harvester configuration file. You schedule the Power BI harvester job each Monday at 6 am and the lineage harvester job at 6 pm. As a result, your Power BI metadata is automatically refreshed on a weekly basis.

# Harvesters upgrade

Each new Power BI harvester and lineage harvester adds features and enhancements to the previous version. We highly recommend that you always use the newest harvester available.

# Upgrade to Power BI harvester 1.0.0.1 or newer and lineage harvester 1.3.0 or newer

The Power BI harvester 1.0.0.1 enables you to connect to a Collibra Data Lineage server, based on your geolocation and cloud provider.

You only have to follow this upgrade procedure when you upgrade from Power BI harvester 1.0.0.0 to Power BI harvester 1.0.0.1 and newer or if the server's geolocation or cloud provider changes.

> Tip We highly recommend that you always use the newest Power BI harvester and lineage harvester.

## Steps

1. If you have strict firewall rules, whitelist one of the following IP addresses, based on your Collibra geolocation and cloud provider:
    - 15.222.200.199 (techlin-aws-ca.collibra.com)
    - 18.198.89.106 (techlin-aws-eu.collibra.com)
    - 54.242.194.190 (techlin-aws-us.collibra.com)
    - 51.105.241.132 (techlin-azure-eu.collibra.com)
    - 20.102.44.39 (techlin-azure-us.collibra.com)
    - 35.197.182.41 (techlin-gcp-au.collibra.com)
    - 34.152.20.240 (techlin-gcp-ca.collibra.com)
    - 35.205.146.124 (techlin-gcp-eu.collibra.com)
    - 34.87.122.60 (techlin-gcp-sg.collibra.com)

- 35.234.130.150 (techlin-gcp-uk.collibra.com)
- 34.73.33.120 (techlin-gcp-us.collibra.com)

> Note IP address 15.222.200.199 is only available for Power BI harvester 1.0.0.2 and lineage harvester 1.3.1 and newer.

2. Download Power BI harvester 1.0.0.1 or newer, from the Collibra Downloads page.
3. Install the new Power BI harvester.
4. Migrate your Power BI connection information in your old configuration file to the configuration file in the new Power BI harvester folder.
5. Trigger the Power BI harvester to upload the Power BI metadata to the Collibra Data Lineage server with the new IP address:
   - Run the following command line if your configuration file is in its default location: `.\powerbi-harvester.bat`
   - Launch the path to the Power BI configuration file if you moved the configuration file to a different location: `.\bin\powerbi-harvester.exe .\config\powerbi-harvester.conf`

> Note We highly recommend that you run the Power BI harvester via command line. This enables you to follow the metadata upload and see possible errors that may occur.

6. Download lineage harvester 1.3.0 or newer, from the Collibra Downloads page.
7. Install the new lineage harvester.
8. Migrate the data sources in your old configuration file to the configuration file in the new lineage harvester folder.
9. Run the lineage harvester with the `full-sync`command.
   » The lineage harvester uploads your data sources to the Collibra Data Lineage server with the new IP address.
10. Restart the DGC service in Collibra Console.

> Tip For more information about Power BI and the Power BI harvester, see Power BI.

## What's next?

Collibra now synchronizes your Power BI assets and relations. You can also access the technical lineage via a Power BI Column asset page.

# Power BI troubleshooting

It is possible that you encounter problems during the Power BI ingestion process.

> Note You can also encounter problems due to Power BI ingestion limitations.

The following table lists possible problems and offer a solution.

| Problem | Description |
|---|---|
| You have made a mistake in the Power BI harvester configuration file or the lineage harvester configuration file. | Make sure to check all properties and values before you run the Power BI harvester.<br><br>If any of the values of the required configuration properties are missing, invalid or incorrect, the Power BI harvester or lineage harvester fails with an error or the Power BI ingestion will be incorrect.<br><br>The Power BI harvester and lineage harvester should have the same value in the `url` and `(source)Id` property. |
| You don't have the correct Power BI permissions or not all prerequisites have been met before you start the Power BI integration process. | Make sure you have read and performed all prerequisites. The prerequisites are slightly different if you choose for username / password or service principal authentication. |

| Problem | Description |
| --- | --- |
| The Power BI harvester failed to retrieve Power BI capacities and shows status code "Unauthorized". | This is a common mistake when you use the service principal authentication method. To solve this issue, make sure that you have enabled the **Allow service principals to use read-only Power BI admin APIs (preview)** option in the Power BI Admin portal,.<br><br>Tip Do not confuse the **Allow service principals to use read-only Power BI admin APIs (preview)** option with the **Allow service principal to use Power BI APIs** option. You need to enable both options. |
| You have network or remote API issues. | Web services providing the API interfaces that the Power BI harvester uses may sometimes experience problems, or there may be problems with network access to these resources. If the Power BI harvester fails unexpectedly, check the following network resources and make sure they work properly:<br><br>• Power BI REST API endpoints<br>• XMLA endpoints<br>• Technical lineage API<br><br>Considering the nature of these remote resources, the cause of the problem can often be out of your control. Please wait until the issue is resolved or escalate the issue with the respective authority. |

| Problem | Description |
| --- | --- |
| You cannot retrieve information for individual Power BI dashboards or data sets. | To retrieve metadata of individual Power BI dashboards or data sets, you require permissions to access them. However, sometimes the Power BI dashboards and data sets are in a problematic state or you cannot reach them due to Power BI-related issues.<br><br>When you execute the Power BI harvester, a summary of all encountered problems is printed. To reduce the number of problems, you can use the group filters in the Power BI configuration file to restrict the set of harvested Power BI workspaces.<br><br>Depending on the type of issue, you may need to solve them one by one. |
| The Power BI harvester cannot retrieve certain workspaces in the `workspaceFilter` property. | Make sure the syntax in the `workspaceFilter` property in the configuration file is correct and you don't use the "IN" operator.<br><br>> Note Currently, the "IN" operator is not supported. As a result, you cannot use "IN" to filter on specific Power BI workspaces in the `workspaceFilter` property in the Power BI configuration file. For more information, see the Power BI limitations. |
| The Power BI harvester failed to connect to the Microsoft API. | Usually, this is a timeout issue. We highly recommend that you increase the timeout duration. Use the following command line option to set the timeout duration: `--timeout <seconds>`. For example, if you want the Power BI harvester to wait 15 minutes for the connection, you can use `--timeout 900`. |

| Problem | Description |
| --- | --- |
| You get an error message related to Usage Metrics. | If you see errors related to Usage Metrics, you can ignore them, because they do not cause Power BI ingestion to fail.<br><br>Usage Metrics are reports that are automatically created in Power BI, but they do not represent any Power BI assets or technical lineage information. |
| The technical lineage is missing or incomplete. | If the technical lineage is missing, you must add your Power BI workspaces to a dedicated capacity to allow the Power BI harvester to extract data from XMLA endpoints.<br><br>Harvesting metadata via Power BI REST API does not require the dedicated capacity. As a result, the Power BI harvester can only reach limited Power BI metadata and won't create a technical lineage.<br><br>If the technical lineage is incomplete, certain aspects of the Power BI ingestion job may not be supported.<br><br>> Note You can only ingest new Power BI workspaces. This means that classic workspaces and My Workspace in Power BI is not supported. Also read the other limitations of the Power BI ingestion process to understand why technical lineage is missing or incomplete. |
| Some Power BI metadata is missing in Data Catalog. | Do the following:<br><br>• Use new Power BI workspaces if you want a full ingestion.<br>• Add your Power BI workspaces to a dedicated capacity to allow the Power BI harvester to extract data from XMLA endpoints.<br>• Grant the Power BI workspaces the Contributor role in the Power BI security group. |

| Problem | Description |
| --- | --- |
| You have successfully ingested Power BI metadata, but calculated tables and columns are not shown in the Technical lineage or in the browse tab pane. | Calculated columns are virtually the same as a non-calculated columns, with one exception: their values are calculated using DAX formulas and values from other columns. Collibra Data Lineage currently does not support internal transformations via DAX language, and any data objects derived via DAX are not shown in the technical lineage or in the browse tab pane. Currently, only M Query/Power Query expressions are supported. |

# Power BI ingestion tests

If you want to test the Power BI ingestion, we recommend that you use the `workspaceFilter` property in the Power BI configuration file to limit the Power BI ingestion to one or two Power BI workspaces.

For more information about the query options to filter Power BI workspaces, see the Microsoft documentation.

> Example If you want to limit the Power BI ingestion to one Power BI workspace with the name PowerBIWorkspace1, you can set the `workspaceFilter` value to "*Name eq 'PowerBIWorkspace1*".

# Power BI harvester messages

When something goes wrong during the Power BI metadata harvesting process, the Power BI harvester logs show a message code that provides a link to more information. The message code indicates which part of the harvesting process failed or was skipped, and provides steps to resolve it.

> Tip Make sure that you understand the Power BI metadata harvesting process and the typical Power BI ingestion workflow.

| Message code | Description |
|---|---|
| MSG-LIN-7000 | This message is a reminder to follow all steps to ingest the Power BI metadata in Data Catalog.<br><br>This message is always shown after the Power BI harvester successfully uploads the Power BI metadata to the Collibra Data Lineage server. Next, you have to create a lineage harvester configuration file and successfully run the lineage harvester to create Power BI assets and relations in Data Catalog. |
| MSG-LIN-7001 | An unexpected problem occurred at the local machine. The error can be caused by an invalid path name, not enough storage space or other unexpected issues. |
| MSG-LIN-7002 | There is a problem with the Power BI harvester configuration file or the source ID configuration file.Make sure that all information in the configuration file(s) and the path to the configuration file(s) is correct. |
| MSG-LIN-7003 | The Power BI harvester could not retrieve tenant information, because the Microsoft API did not return a response that the Power BI harvester could process.<br>To solve this problem, we recommend that you check your network settings and rerun the Power BI harvester. If the issue persists, please contact Collibra support or your customer success manager. |

| Message code | Description |
| --- | --- |
| MSG-LIN-7004 | The Power BI harvester could not communicate with the Collibra Data Lineage server, likely because of one of the following scenarios:<br>• The remote API did not return a response that the Power BI harvester could process.<br>• The API call returned a 401 (Unauthorized) error because an invalid userKey token was used.<br>To solve this problem, we recommend that you:<br>• Ensure that the Power BI harvester is connecting to the correct Collibra Data Lineage server.<br>• Check your network settings and rerun the Power BI harvester.<br>If the issue persists, please contact Collibra support or your customer success manager. |
| MSG-LIN-7005 | The Power BI harvester could not retrieve Power BI metadata, because the Power BI service did not return a response that the Power BI harvester could process.<br>To solve this problem, we recommend that you check your network settings and rerun the Power BI harvester. If the issue persists, please contact Collibra support or your customer success manager.<br>> Note If the error message indicates that the issue is an internal server error, the problem is caused by the Power BI REST API. |

| Message code | Description |
| --- | --- |
| MSG-LIN-7006 | The Power BI harvester could not communicate with a remote server, because the server did not return a response within an expected time interval and, as a result, the Power BI harvester aborted the process.<br><br>To solve this problem, we recommend that you do the following:<br><br>• Check your network settings.<br>• Check the amount of metadata that is processed. If it is a large amount, use the `--timeout` command line option to specify a longer timeout for remote API calls.<br><br>If the problem persists or the remote server does not respond within a reasonable time period, create a support ticket or contact your customer success manager. |
| MSG-LIN-7007 | This problem occurs when the Power BI service returns inconsistent data. As a result, the Power BI harvester cannot successfully process the data to create a consistent result data set.<br><br>The Power BI harvester uses multiple API calls to retrieve Power BI metadata. If something in the Power BI service changed during the harvesting process, the metadata can be inconsistent. We recommend to run the Power BI harvester again. If the issue persists, create a support ticket or contact your customer success manager. |
| MSG-LIN-7008 | The Power BI harvester cannot access XMLA endpoints for some Power BI dedicated capacities with harvested workspaces, because the capacities are currently not running. As a result, the Collibra Data Lineage server cannot create a technical lineage for these workspaces.<br><br>To solve this problem, check if the Power BI workspace is part of a running dedicated capacity and you meet the necessary prerequisites to access and export it. |

| Message code | Description |
| --- | --- |
| MSG-LIN-7009 | The Power BI authentication failed. This problem can be caused by an error in the Power BI credentials.<br><br>To solve this problem, check your Power BI login credentials in the Power BI harvester configuration file or reenter them via command line. |
| MSG-LIN-7010 | The connection between the Power BI harvester and the Collibra Data Lineage server failed. This problem can be caused by an error in the Collibra Data Intelligence Cloud credentials or Collibra Data Intelligence Cloud host address.<br><br>To solve this problem, check your Collibra Data Intelligence Cloud credentials in the Power BI harvester configuration file or reenter them via command line. |
| MSG-LIN-7011 | The Power BI harvester could not retrieve the tenant domain information.<br><br>To solve this problem, check that you have the correct tenant domain ID in the Power BI harvester configuration file. |

| Message code | Description |
| --- | --- |
| MSG-LIN-7012 | The Power BI API failed. This can be caused by an error in the syntax of the `workspaceFilter` field in the Power BI harvester configuration file.<br><br>Important The `workspace filter` operations use OData syntax and are processed by the Power BI service, not the Power BI harvester.<br><br>Examples of supported workspace filter operations:<br><br>• `name eq 'Workspace1' or name eq 'Workspace2'` only harvests workspaces with the specified names.<br>• `not endswith(name, 'Test')` only harvests workspaces whose names don't end in `Test`.<br>• `tolower(capacityId) eq '01234567-89ab-cdef-0123-456789abcdef'` only harvests workspaces hosted on the specified dedicated capacity.<br>• `reports/any(d:contains(d/name, 'Sales'))` only harvests workspaces with reports whose names contain `Sales`.<br><br>If you do not want to filter on specific workspaces, leave the `workspaceFilter` field in the Power BI harvester configuration file empty.<br><br>Tip For more information about the query options to filter Power BI workspaces, see the Microsoft documentation. We cannot guarantee that other group filter operations work correctly. For example, the `IN` operator is currently not supported. |

| Message code | Description |
|---|---|
| MSG-LIN-7013 | You do not have the required permissions to harvest the Power BI metadata. Check that the user is a Power BI Administrator and that the Power BI application has all required permissions.<br><br>To solve this problem, check that you have correctly registered your Power BI application in Microsoft Azure. |
| MSG-LIN-7014 | You do not have the required permissions to harvest the Power BI metadata. Enable the **Allow service principals to use read-only Power BI admin APIs (preview)** option in the Power BI Admin Console.<br><br>To solve this problem, check that you meet the prerequisites to use the service principal. |
| MSG-LIN-7015 | You do not have the required permissions to harvest the Power BI metadata. Enable the **Allow service principal to use Power BI APIs** option in the Power BI Admin Console.<br><br>To solve this problem, check that you meet the prerequisites to use the service principal. |
| MSG-LIN-7016 | The harvested Power BI workspaces are not assigned to a dedicated capacity. As a result, Data Catalog cannot ingest details about tables and columns and technical lineage are not available.<br><br>> Note You do not have to assign less important to a dedicated capacity, for example personal workspaces. However, if there are no workspaces on a dedicated capacity, the harvested Power BI metadata is very limited. |

| Message code | Description |
| --- | --- |
| MSG-LIN-7017 | The Power BI harvester could not access XMLA endpoints for any Power BI workspaces to retrieve detailed information about data sets. As a result, technical lineage is be available.<br><br>To solve this issue, check that you meet the prerequisites to access XMLA endpoints for all Power BI workspaces that you want to ingest in Data Catalog. |
| MSG-LIN-7018 | Batch processing failed at Collibra server.<br><br>The harvested batch was uploaded to a Collibra Data Lineage server, but the server could not process the batch.<br><br>Review the error message that accompanies this error code. It might identify a problem that you can resolve, for example if you used an unsupported version of the harvester. If the error message does not identify the problem or if you're unable to resolve it on your own, create a support ticket or contact your customer success manager. |

# Working with SSRS and PBRS

SQL Server Reporting Services (SSRS) and Power BI Report Server (PBRS) are server-based report generating applications created by Microsoft that helps you see and understand your data.

PBRS is included in the licensing of the SQL Server Enterprise Edition or as a free extension of Power BI premium. SSRS and PBRS are closely related and both use the same API to communicate to the lineage harvester. As a result, Collibra created one operating model that contains data from both SSRS and PBRS. Whether you use SSRS, PBRS or both, you only need one integration in the lineage harvester configuration file.

> Note While SSRS and PBRS use the same API, we can access less information from PBRS reports than from SSRS data. As a result, we do not support stitching and lineage information for PBRS reports.



# SSRS and PBRS asset and domain types

The SQL Server Reporting Services (SSRS) and Power BI Report Server (PBRS) integration in Collibra Data Intelligence Cloud uses a specific subset of asset types and domain types.

The following table shows the asset types and domain types that are used for the SSRS and PBRS integration. You can see the parent asset types in the breadcrumbs above each asset type.

| Asset type | Description | Domain type |
| --- | --- | --- |
| Business Asset ▸ Business Dimension ▸ BI Folder ▸ SSRS Folder | A collection of SQL Server Reporting Services and Power BI Report Server Reports and Data Sets. | BI Catalog |
| Business Asset ▸ Report ▸ BI Report ▸ SSRS KPI | A key performance indicator of SQL Server Reporting Services. | BI Catalog |
| Business Asset ▸ Report ▸ BI Report ▸ SSRS Report | A detailed view of an SQL Server Reporting Services Data Set, with visualizations of findings and insights. | BI Catalog |
| Data Asset ▸ Data Element ▸ Data Attribute ▸ BI Data Attribute ▸ SSRS Column | A column in an SQL Server Reporting Services Report Data Set. | BI Catalog |
| Data Asset ▸ Data Element ▸ Report Attribute ▸ BI Report Attribute ▸ SSRS Parameter | A column that is part of an SQL Server Reporting Services Data Set and that is used in a KPI. | BI Catalog |

| Asset type | Description | Domain type |
|---|---|---|
| Data Asset ▸ Data Set ▸ BI Data Set ▸ SSRS Data Model | A collection of data that is used to create an SQL Server Reporting Services Report. | BI Catalog |
| Data Asset ▸ Data Element ▸ Data Attribute ▸ BI Data Attribute ▸Power BI Table▸ SSRS Table | A table in an SQL Server Reporting Services Report Data Set. | BI Catalog |
| Technology Asset ▸ Server ▸ BI Server ▸ SSRS Server | A visual analytics platform for creating and storing SQL Server Reporting Services and Power BI Report Server Reports and Data Sets. | BI Catalog |

# SSRS and PBRS terminology

The following table shows the SQL Server Reporting Services (SSRS) and Power BI Report Server (PBRS) terminology and how it maps to the Collibra Data Intelligence Cloud asset types.

| Term | Description | Asset type in Collibra |
|---|---|---|
| Column | A column in an SQL Server Reporting Services Report Data Set. | SSRS Column |

| Term | Description | Asset type in Collibra |
|---|---|---|
| Data Set | A collection of data that is used to create an SQL Server Reporting Services Report. | SSRS Data Model |
| Folder | A collection of SQL Server Reporting Services and Power BI Report Server Reports and Data Sets. | SSRS Folder |
| KPI | A key performance indicator of SQL Server Reporting Services. | SSRS KPI |
| Mobile report | A detailed view of an SQL Server Reporting Services Data Set, with visualizations of findings and insights. | SSRS Report |
| Paginated report | A detailed view of an SQL Server Reporting Services Data Set, with visualizations of findings and insights. | SSRS Report |
| Parameter | A column that is part of an SQL Server Reporting Services Data Set and that is used in a KPI. | SSRS Parameter |

| Term | Description | Asset type in Collibra |
|---|---|---|
| Power BI Report Server report | A detailed view of a Power BI Data Model, with visualizations of findings and insights. | Power BI Report |
| SQL Server Reporting Services or Power BI Report Server server or tenant | A visual analytics platform for creating and storing SQL Server Reporting Services and Power BI Report Server Reports and Data Sets. | SSRS Server |
| Table | A table in an SQL Server Reporting Services Report Data Set. | SSRS Table |

# SSRS and PBRS operating model

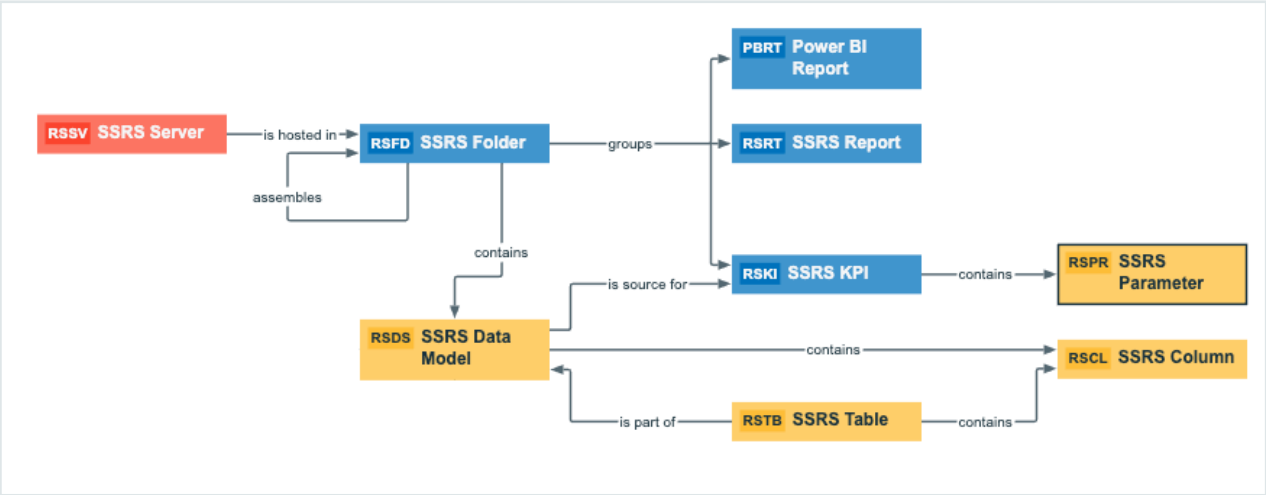
The lineage harvester collects SQL Server Reporting Services (SSRS) metadata and sends it to the Collibra Data Lineage service. Collibra processes the metadata and creates new SSRS assets and relations in Data Catalog. You can see them on the asset page overview or visualize them in a diagram or in a technical lineage.

Note

- The assets have the same names as their counterparts in SSRS and Power BI Report Server (PBRS). Full names and Names cannot be changed in Data Catalog.
- Assets ingested from SSRS and PBRS are called SSRS assets in Data Catalog, except for PBRS reports which are called Power BI Report assets.
- Asset types are only created if you have all specific Data Catalog permissions.
- All SSRS and PBRS assets are created in the same domain.
- Relations that were manually created between SSRS assets or PBRS assets and other assets via a relation type in the SSRS and PBRS operating model, are deleted after synchronizing the metadata.

# SSRS and PBRS metadata overview

The following image shows the relations between SSRS asset types and the Power BI Report asset type.




# Harvested metadata per asset type

This table shows the harvested SSRS and PBRS metadata for each SSRS asset type and Power BI Report asset type, assuming you have the necessary subscriptions and configurations for a full ingestion. This table also shows the resource ID for each asset type and metadata.

| Asset type | Synchronized metadata | Resource ID |
|---|---|---|
| SSRS Column<br>Resource ID: 00000000-0000-0000-0000-100000000029 | Full name | |
| | Display name | |
| | Description | 00000000-0000-0000-0000-000000003114 |
| | Technical Data Type | 00000000-0000-0000-0000-000000000219 |
| | BI Data Model contains / is part of BI Data Attribute | 00000000-0000-0000-0000-000000007196 |
| | Data Element targets / sources Data Element | 00000000-0000-0000-0000-000000007069 |
| | Data Entity contains / is part of Data Attribute | 00000000-0000-0000-0000-000000007047 |

| Asset type | Synchronized metadata | Resource ID |
| --- | --- | --- |
| SSRS Data Model<br><br>Resource ID: 00000000-0000-0000-0000-100000000028 | Full name | |

| Asset type | Synchronized metadata | Resource ID |
| --- | --- | --- |
| | Display name | |

| Asset type | Synchronized metadata | Resource ID |
|---|---|---|
| | Certified | 00000000-0000-0000-0001-000500000001 |
| | Description | 00000000-0000-0000-0000-000000003114 |
| | Document creation date | 00000000-0000-0000-0000-000000000260 |
| | Document modification date | 00000000-0000-0000-0000-000000000261 |
| | Document size | 00000000-0000-0000-0000-000000000259 |
| | Location | 00000000-0000-0000-0000-000000000203 |
| | URL | 00000000-0000-0000-0000-000000000258 |
| | Visible on server | 00000000-0000-0000-0000-000000000265 |
| | BI Data Model contains / is part of BI Data Attribute | 00000000-0000-0000-0000-000000007196 |
| | BI Folder contains / contained in Data Asset | 00000000-0000-0000-0000-120000000014 |
| | Data Asset is source for / source BI report | 00000000-0000-0000-0000-120000000013 |

| Asset type | Synchronized metadata | Resource ID |
| --- | --- | --- |
| | Data Entity is part of / contains Data Model | 00000000-0000-0000-0000-000000007046 |

| Asset type | Synchronized metadata | Resource ID |
|---|---|---|
| SSRS Folder<br>Resource ID: 00000000-0000-0000-0000-100000000024 | Full name | |
| | Display name | |
| | Description | 00000000-0000-0000-0000-000000003114 |
| | Document creation date | 00000000-0000-0000-0000-000000000260 |
| | Document modification date | 00000000-0000-0000-0000-000000000261 |
| | Location | 00000000-0000-0000-0000-000000000203 |
| | URL | 00000000-0000-0000-0000-000000000258 |
| | Visible on server | 00000000-0000-0000-0000-000000000265 |
| | Business Dimension groups / is grouped into Report | 00000000-0000-0000-0000-120000000002 |
| | BI Folder assembles / is assembled in BI Folder | 00000000-0000-0000-0000-120000000001 |
| | BI Folder contains / contained in Data Asset | 00000000-0000-0000-0000-120000000014 |
| | Server hosts / is hosted in Business Dimension | 00000000-0000-0000-0000-120000000000 |

| Asset type | Synchronized metadata | Resource ID |
| --- | --- | --- |
| SSRS KPI<br><br>Resource ID: 00000000-0000-0000-0000-100000000026 | Full name | |

| Asset type | Synchronized metadata | Resource ID |
| --- | --- | --- |
| | Display name | |

| Asset type | Synchronized metadata | Resource ID |
|---|---|---|
| | Certified | 00000000-0000-0000-0001-000500000001 |
| | Description | 00000000-0000-0000-0000-000000003114 |
| | Document creation date | 00000000-0000-0000-0000-000000000260 |
| | Document modification date | 00000000-0000-0000-0000-000000000261 |
| | Document size | 00000000-0000-0000-0000-000000000259 |
| | Location | 00000000-0000-0000-0000-000000000203 |
| | URL | 00000000-0000-0000-0000-000000000258 |
| | Visible on server | 00000000-0000-0000-0000-000000000265 |
| | Business Dimension groups / is grouped into Report | 00000000-0000-0000-0000-120000000002 |
| | Data Asset is source for / source BI Report | 00000000-0000-0000-0000-120000000013 |
| | Report Attribute contained in / contains Report | 00000000-0000-0000-0000-000000007058 |

| Asset type | Synchronized metadata | Resource ID |
|---|---|---|
| | Report related to / impacted by Business Asset | 00000000-0000-0000-0000-120000000006 |
| SSRS Parameter<br><br>Resource ID: 00000000-0000-0000-0000-100000000027 | Full name | |
| | Display name | |
| | Description | 00000000-0000-0000-0000-000000003114 |
| | Business Asset represents / represented by Data Asset | 00000000-0000-0000-0000-000000007038 |
| | Report Attribute contained in / contains Report | 00000000-0000-0000-0000-000000007058 |
| | Report Attribute sourced from / is source of Data Attribute | 00000000-0000-0000-0000-120000000010 |

| Asset type | Synchronized metadata | Resource ID |
| --- | --- | --- |
| SSRS Report<br><br>Resource ID: 00000000-0000-0000-0000-100000000025 | Full name | |

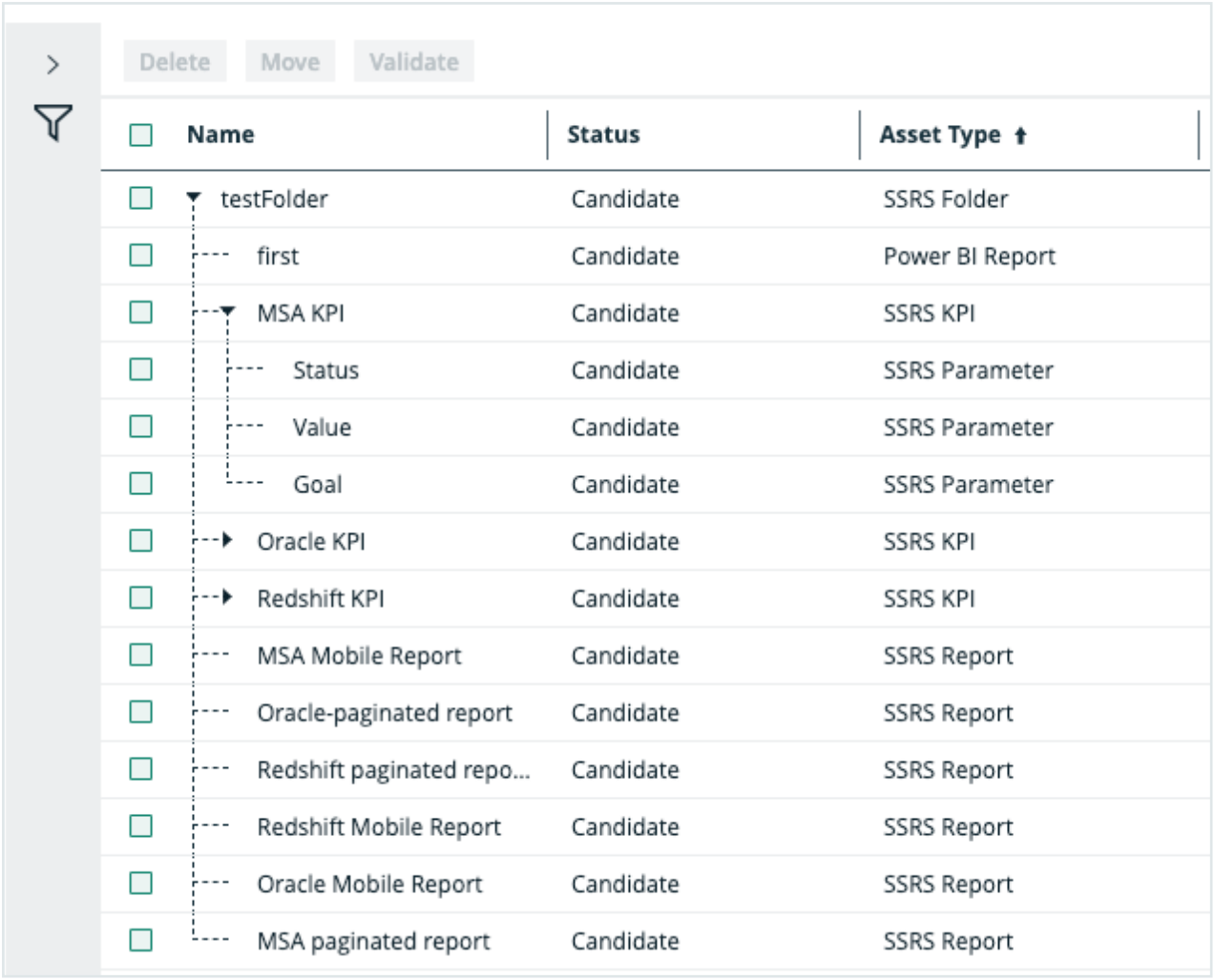
| Asset type | Synchronized metadata | Resource ID |
| --- | --- | --- |
| | Display name | |

| Asset type | Synchronized metadata | Resource ID |
|---|---|---|
| | Certified | 00000000-0000-0000-0001-000500000001 |
| | Description | 00000000-0000-0000-0000-000000003114 |
| | Document creation date | 00000000-0000-0000-0000-000000000260 |
| | Document modification date | 00000000-0000-0000-0000-000000000261 |
| | Document size | 00000000-0000-0000-0000-000000000259 |
| | Location | 00000000-0000-0000-0000-000000000203 |
| | URL | 00000000-0000-0000-0000-000000000258 |
| | Visible on server | 00000000-0000-0000-0000-000000000265 |
| | Business Dimension groups / is grouped into Report | 00000000-0000-0000-0000-120000000002 |
| | Data Asset is source for / source BI Report | 00000000-0000-0000-0000-120000000013 |
| | Report related to / impacted by Business Asset | 00000000-0000-0000-0000-120000000006 |

| Asset type | Synchronized metadata | Resource ID |
|---|---|---|
| | Report uses / used in Report | 00000000-0000-0000-0000-120000000007 |
| SSRS Server<br>Resource ID: 00000000-0000-0000-0000-100000000023 | Full name | |
| | Display name | |
| | Description | 00000000-0000-0000-0000-000000003114 |
| | Server hosts / is hosted in Business Dimension | 00000000-0000-0000-0000-120000000000 |
| SSRS Table<br>Resource ID: 00000000-0000-0000-0000-100000000030 | Full name | |
| | Display name | |
| | Description | 00000000-0000-0000-0000-000000003114 |
| | Data Entity contains / is part of Data Attribute | 00000000-0000-0000-0000-000000007047 |
| | Data Entity is part of / contains Data Model | 00000000-0000-0000-0000-000000007046 |

# Example of ingested SSRS and PBRS metadata

The following image shows an example structure after SSRS and PBRS ingestion.




# Create an SSRS and PBRS operating model diagram view

You can create a diagram view for SSRS and PBRS to visualize the operating model. Complete the following steps to create a new diagram view by copying and pasting the JSON code in the diagram view text editor.

## Steps

1. Open an asset page.
2. In the tab pane, click **Diagram**.
   » The diagram appears in the default diagram view.

3. Click + to add a new view.
4. Click the **Text** tab, to switch to the diagram view text editor.
5. Click **Show me the JSON code** below this procedure, to expand the code.
6. Paste the code in diagram view text editor.
7. Click **Save**.
8. Edit the name and description of the diagram view, to suit your needs.

### Show me the JSON code

```
{
  "nodes": [
   {
      "id": "SSRS Column",
     "type": {
       "id": "00000000-0000-0000-0000-100000000029"
      }
    },
    {
      "id": "SSRS Data Model",
      "type": {
        "id": "00000000-0000-0000-0000-100000000028"
      }
    },
    {
      "id": "SSRS Table",
      "type": {
        "id": "00000000-0000-0000-0000-100000000030"
       }
     },
     {
       "id": "SSRS KPI",
       "type": {
         "id": "00000000-0000-0000-0000-100000000026"
       }
     },
     {
       "id": "SSRS Parameter",
       "type": {
         "id": "00000000-0000-0000-0000-100000000027"
       }
     },
     {
       "id": "SSRS Folder",
       "type": {
         "id": "00000000-0000-0000-0000-100000000024"
       }
     },
```

```
        {
          "id": "Power BI Report",
          "type": {
            "id": "00000000-0000-0000-0000-100000000006"
          }
        },
        {
           "id": "SSRS Report",
          "type": {
            "id": "00000000-0000-0000-0000-100000000025"
          }
        },
        {
           "id": "SSRS Folder 2",
           "type": {
             "id": "00000000-0000-0000-0000-100000000024"
           }
         },
         {
           "id": "SSRS Server",
           "type": {
             "id": "00000000-0000-0000-0000-100000000023"
           }
         },
         {
           "id": "Column",
           "type": {
             "id": "00000000-0000-0000-0000-000000031008"
           }
         },
         {
           "id": "Table",
           "type": {
             "id": "00000000-0000-0000-0000-000000031007"
           }
         },
         {
           "id": "Schema",
           "type": {
             "id": "00000000-0000-0000-0001-000400000002"
           }
         },
         {
           "id": "Database",
           "type": {
             "id": "00000000-0000-0000-0000-000000031006"
           }
         }
      ],
       "edges": [
```

```
{
  "from": "SSRS Data Model",
  "to": "SSRS Column",
  "label": "",
  "style": "arrow",
  "type": {
    "id": "00000000-0000-0000-0000-000000007196"
  },
  "roleDirection": true
 },
 {
    "from": "SSRS Table",
    "to": "SSRS Column",
    "label": "",
    "style": "arrow",
    "type": {
      "id": "00000000-0000-0000-0000-000000007047"
    },
    "roleDirection": true
  },
  {
    "from": "SSRS Data Model",
    "to": "SSRS Table",
    "label": "",
    "style": "arrow",
    "type": {
      "id": "00000000-0000-0000-0000-000000007046"
    },
    "roleDirection": true
  },
  {
    "from": "SSRS Data Model",
    "to": "SSRS KPI",
    "label": "",
    "style": "arrow",
    "type": {
      "id": "00000000-0000-0000-0000-120000000013"
    },
    "roleDirection": true
  },
  {
    "from": "SSRS KPI",
    "to": "SSRS Parameter",
    "label": "",
    "style": "arrow",
    "type": {
      "id": "00000000-0000-0000-0000-120000000014"
   },
    "roleDirection": true
  },
```

```
        {
          "from": "SSRS Folder",
          "to": "SSRS Data Model",
          "label": "",
          "style": "arrow",
          "type": {
            "id": "00000000-0000-0000-0000-120000000014"
          },
          "roleDirection": true
        },
        {
          "from": "SSRS Folder",
          "to": "Power BI Report",
          "label": "",
          "style": "boxing",
          "type": {
            "id": "00000000-0000-0000-0000-120000000002"
          },
          "roleDirection": true
        },
        {
          "from": "SSRS Folder",
          "to": "SSRS Report",
          "label": "",
          "style": "boxing",
          "type": {
            "id": "00000000-0000-0000-0000-120000000002"
          },
          "roleDirection": true
        },
        {
          "from": "SSRS Folder",
          "to": "SSRS KPI",
          "label": "",
          "style": "boxing",
          "type": {
            "id": "00000000-0000-0000-0000-120000000004"
          },
          "roleDirection": true
        },
        {
          "from": "SSRS Server",
          "to": "SSRS Folder 2",
          "label": "",
          "style": "boxing",
          "type": {
            "id": "00000000-0000-0000-0000-120000000000"
          },
          "roleDirection": false
        },
```

```
        {
          "from": "SSRS Folder 2",
          "to": "SSRS Folder",
          "label": "",
          "style": "boxing",
          "type": {
            "id": "00000000-0000-0000-0000-120000000001"
          },
          "roleDirection": true
        },
        {
          "from": "SSRS Folder",
          "to": "SSRS Server",
          "label": "",
          "style": "boxed",
          "type": {
            "id": "00000000-0000-0000-0000-120000000000"
          },
          "roleDirection": false
        },
        {
          "from": "SSRS Report",
          "to": "SSRS Data Model",
          "label": "",
          "style": "arrow",
          "type": {
          "id": "00000000-0000-0000-0000-120000000013"
          },
          "roleDirection": false
        },
        {
          "from": "SSRS Column",
          "to": "Column",
          "label": "",
          "style": "arrow",
          "type": {
            "id": "00000000-0000-0000-0000-000000007069"
          },
          "roleDirection": false
        },
        {
          "from": "Column",
          "to": "Table",
          "label": "",
          "style": "boxed",
          "type": {
            "id": "00000000-0000-0000-0000-000000007042"
          },
          "roleDirection": true
        },
```

```
      {
        "from": "Table",
        "to": "Schema",
        "label": "",
        "style": "boxed",
        "type": {
          "id": "00000000-0000-0000-0000-000000007043"
        },
        "roleDirection": false
      },
      {
        "from": "Schema",
        "to": "Database",
        "label": "",
        "style": "boxed",
        "type": {
          "id": "00000000-0000-0000-0000-000000007024"
        },
        "roleDirection": false
      }
    ],
    "showOverview": false,
    "enableFilters": true,
    "showLabels": false,
    "showFields": true,
    "showLegend": true,
    "showPreview": true,
    "visitStrategy": "directed",
    "layout": "HierarchyLeftRight",
    "maxNodeLabelLength": 50,
    "maxEdgeLabelLength": 30,
    "layoutOptions": {
      "compactGroups": false,
      "componentArrangementPolicy": "topmost",
      "edgeBends": true,
      "edgeBundling": true,
      "edgeToEdgeDistance": 5,
      "minimumLayerDistance": "auto",
      "nodeToEdgeDistance": 5,
      "orthogonalRouting": true,
      "preciseNodeHeightCalculation": true,
      "recursiveGroupLayering": true,
      "separateLayers": true,
      "webWorkers": true,
      "nodePlacer": {
        "barycenterMode": true,
        "breakLongSegments": true,
        "groupCompactionStrategy": "none",
        "nodeCompaction": false,
        "straightenEdges": true
```

```
            }
          }
        }
```

# Automatic stitching

SQL Server Reporting Services (SSRS) and Power BI Report Server (PBRS) is business intelligence software that can integrate with various data sources. When you ingest metadata, Collibra Data Lineage tries to automatically stitch this metadata to data sources registered in Data Catalog.

Stitching is a process that creates relations between database columns that are Column assets in Collibra Data Intelligence Cloud and BI assets representing the same database, specifically between:

- The assets that are created when you ingest SSRS and PBRS.
- The assets that are created when you register a data source.

When the full name of Column assets in Data Catalog matches the full name of SSRS Column assets collected from SSRS, the Collibra Data Lineage service stitches them by creating a relation of the type "Data Element targets / sources Data Element".

> Note To clarify, the SSRS Column is the target of the Column, and the Column is the source of the SSRS Column.

# Stitching: matching the full paths of assets

To stitch assets in Data Catalog to data object collected by the lineage harvester, the Collibra Data Lineage service looks at the full path of the assets in Data Catalog and the full path of SSRS-PBRS assets. If the full paths match, the Collibra Data Lineage automatically stitches them.

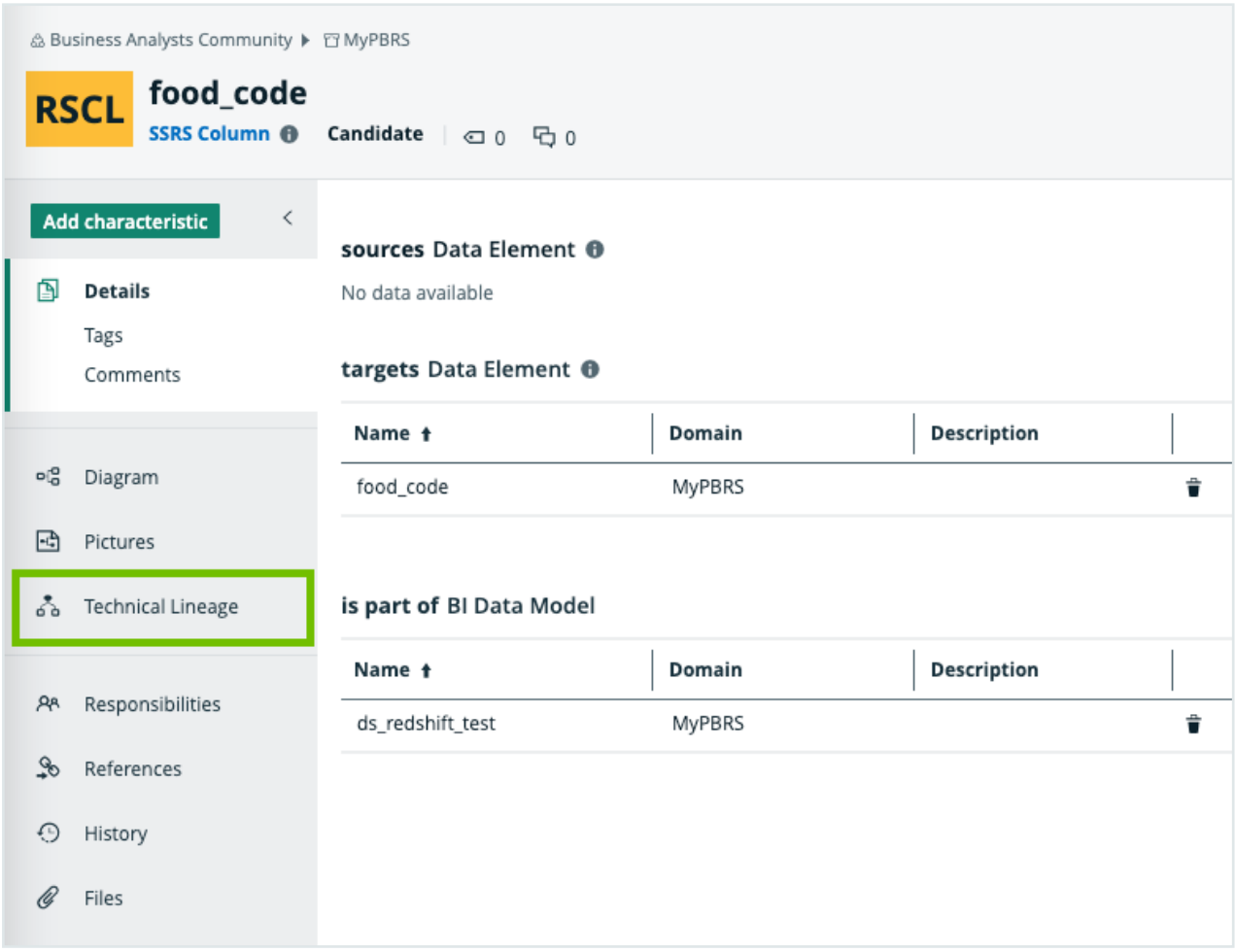
> Tip You can use the Stitching tab page to easily find the full path of assets in Data Catalog and data objects that were collected by the lineage harvester.

# Technical lineage for SSRS and PBRS

When you ingest SQL Server Reporting Services (SSRS) and Power BI Report Server (PBRS) metadata in Data Catalog, you automatically create a technical lineage for SSRS Column assets. Each SSRS Column asset page has a Technical lineage tab page that shows the technical lineage of that asset Column asset.

We cannot access PBRS lineage information. As a result, you can only create a technical lineage for SSRS Column assets.

> Note If you ingest SSRS and PBRS for the first time, or if you change your geolocation or cloud provider, you might have to restart the DGC service before you can see your technical lineage.



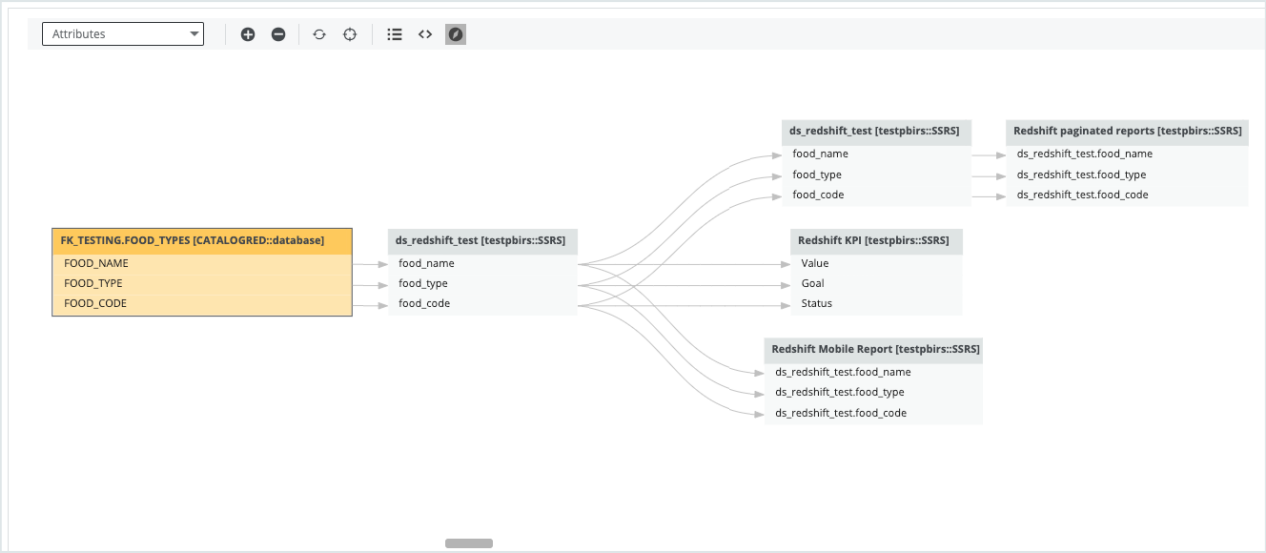
# Technical lineage graph

The technical lineage graph shows relations of the type "Column is source for / is target of Data Attribute" between BI assets and other data objects in the data flow, for example Column assets or Power BI Column assets. These relations are created during the ingestion process as a result of automatic stitching.

For more information about the technical lineage, see the Collibra Data Lineage section in the documentation.

## Example

The following technical lineage shows the relation of the type "Data Element sources / targets Data Element" between the Column assets *FOOD_NAME*, *FOOD_TYPE* and *FOOD_CODE* and the SSRS Column assets *food_name*, *food_type* and *food_code*.




# Sources tab page

The Sources tab page shows the transformation details that the Collibra Data Lineage service analyzed and processed and the results of this analysis. The success rate of the analysis indicates how complete the technical lineage is.

> Important The Collibra Data Lineage service can process most, but not all complex metadata. This means that the success rate of an ingestion job can be very high, but might not be 100%.

# Overview of SSRS and PBRS steps

The SQL Server Reporting Services (SSRS) and Power BI Report Server (PBRS) integration in Collibra Data Intelligence Cloud enables you to harvest SSRS and PBRS metadata and create new SSRS and Power BI assets in Data Catalog. Collibra analyzes and processes the BI metadata and presents it as assets of specific types, retaining their original names.

> Important In the global assignment of each asset type included in the SSRS-PBRS operating model, ensure that none of the characteristics that are in the operating model have a maximum cardinality of “0”. If the maximum cardinality is set to “0” for any such characteristics, ingestion will fail.

> Note While SSRS and PBRS use the same API, we can access less information from PBRS reports than from SSRS data. As a result, we do not support stitching and lineage information for PBRS reports.

# Roles and permissions in SSRS or PBRS.

To ingest SSRS and PBRS metadata into Data Catalog, the lineage harvester connects to the SSRS or PBRS web portal. You need a role with user access to the server from which you want to ingest:

- You have a system-level role, which is at least a System user role.
- You have an item-level role, which is at least a Content Manager role.

> Note You can ingest SSRS and PBRS with any report server subscription. We highly recommend to install and use the latest version of SSRS and PBRS .

# Steps

The table below shows the steps and prerequisites required to integrate SSRS in Data Catalog.

| Step | What? | Description | Prerequisites |
|---|---|---|---|
| 1 | Create a new domain. | Before you can ingest SSRS and PBRS metadata, you have to create a new domain or choose an existing domain to store the new assets. | • You have a resource role with the following resource permissions:<br>◦ Domain: Add |

| Step | What? | Description | Prerequisites |
|---|---|---|---|
| 2 | Download and install the lineage harvester. | You use the lineage harvester to collect metadata from SSRS and upload it to the Collibra Data Lineage service where the metadata is scanned, processed and analyzed.<br><br>You can download the lineage harvester from the Downloads section of the Collibra Product Resource Center. | • You have Collibra Data Intelligence Cloud 2021.09 or newer.<br>• You have access to the lineage harvester. We highly recommend that you always install and use the newest lineage harvester.<br>• Your environment meets the system requirements to install and use the lineage harvester.<br>• You have added firewall rules so that the lineage harvester can connect to the Collibra Data Lineage service instances with the following IP addresses:<br>  • 15.222.200.1-99 (techlin- |

| Step | What? | Description | Prerequisites |
| --- | --- | --- | --- |
| | | | aws-ca.-collibra.com)<br>• 18.198.89.10-6 (techlin-aws-eu.-collibra.com)<br>• 54.242.194.1-90 (techlin-aws-us.-collibra.com)<br>• 51.105.241.1-32 (techlin-azure-eu.-collibra.com)<br>• 20.102.44.39 (techlin-azure-us.-collibra.com)<br>• 35.197.182.4-1 (techlin-gcp-au.-collibra.com)<br>• 34.152.20.24-0 (techlin-gcp-ca.-collibra.com)<br>• 35.205.146.1-24 (techlin-gcp-eu.-collibra.com)<br>• 34.87.122.60 (techlin-gcp-sg.-collibra.com) |

| Step | What? | Description | Prerequisites |
| --- | --- | --- | --- |
| | | | • 35.234.130.1-50 (techlin-gcp-uk.-collibra.com)<br>• 34.73.33.120 (techlin-gcp-us.-collibra.com) |

| Step | What? | Description | Prerequisites |
|---|---|---|---|
| 3 | Prepare the lineage harvester configuration file and run the lineage harvester. | You create a configuration file to provide the connection information that you need to connect your SSRS application to the Collibra Data Lineage service and to the Collibra Data Intelligence Cloud domain in which you want to ingest the SSRS assets.<br><br>You can access an empty configuration file in the lineage harvester installation folder. When you have created and saved the configuration file, you can run the lineage harvester to upload the SSRS metadata to Collibra.<br><br>`{ "general": { "catalog": { "url": "https://<organization>.collibra.com", "username": "<your-collibra-username>" }, "useCollibraSystemName": false }, "sources": { "collibraSystemName" : "", "id": "<unique-id>", "type": "<SSRS or PBRS>", "url": "http://<IP address>/Reports", "username": "<user-name>", "domainId": "<domain-resource-id>", "folderFilter": ["/Folder1/*", "Folder2"] } }` | • You have a dedicated domain.<br>• You have a global role with the Catalog global permission, for example Catalog Author.<br>• You have a global role with the Technical lineage global permission.<br>• You have a global role with the Data Stewardship Manager global permission.<br>• You have a resource role with the following resource permissions:<br>◦ Asset: Add<br>◦ Attribute: Add<br>◦ Attachment: Add<br>• Your environment meets the system requirements to run the lineage |

| Step | What? | Description | Prerequisites |
|---|---|---|---|
| | | | harvester. |
| 4 | If required, create a <source ID> configuration file. | If the `useCollibraSystemName` in the lineage harvester configuration file is set to `true`, you have to provide additional information about the used data sources in SSRS. | • You have created a lineage harvester configuration file. |
| 5 | View the SSRS and PBRS ingestion results. | After the SSRS and PBRS metadata is ingested in Data Catalog, you can go to the domain where you ingested the results and see the list of ingested SSRS assets and Power BI Report assets.<br>Warning When you run the lineage harvester, Collibra Data Lineage creates all assets in the same Data Catalog BI domain. We highly recommend that you do not move these assets to another domain. If you move assets to another domain, they will be deleted and recreated in the initial Data Catalog BI domain when you synchronize the metadata. As a consequence, all manually added data of those assets is lost. | • Catalog Experience is enabled in Collibra Console.<br>• You have a global role with the Catalog global permission, for example Catalog Author. |

# The lineage harvester setup for SSRS and PBRS

The lineage harvester is a software application that is needed to collect your SQL Server Reporting Services (SSRS) and Power BI Report Server (PBRS) metadata and send it to the Collibra Data Lineage service, where the metadata is processed and new assets and

relations are created. Collibra Data Intelligence Cloud then import those assets and relations into Data Catalog.

For more information about the lineage harvester, read the Collibra Data Lineage section.

> Note We highly recommend that you always install and use the newest lineage harvester. You can download the harvester via the Downloads section of the Collibra Product Resource Center.

# Lineage harvester system requirements

You need to meet the system requirements to be able to install and run the lineage harvester.

## Software requirements

Java Runtime Environment version 11 or newer, or OpenJDK 11 or newer.

For Java Runtime Environment 16 or newer, or OpenJDK 16 or newer, set the `JAVA_OPTS` environment variable for the lineage harvester to function properly:

```
JAVA_OPTS='--illegal-access=deny'
```

> Note To ingest Snowflake data sources, the minimum requirement is Java Runtime Environment version 16 or newer, or OpenJDK 16 or newer.

## Hardware requirements

You need to meet the hardware requirements to install and run the lineage harvester.

### Minimum hardware requirements

You need the following minimum hardware requirements:

- 2 GB RAM
- 1 GB free disk space

### Recommended hardware requirements

The minimum requirements are most likely insufficient for production environments. We recommend the following hardware requirements:

- 4 GB RAM

  > Tip 4 GB RAM is sufficient in most cases, but more memory could be needed for larger harvesting tasks. For instructions on how to increase the maximum heap size, see Technical lineage general troubleshooting.

- 20 GB free disk space

## Network requirements

The lineage harvester uses the HTTPS protocol by default and uses port 443.

You need the following minimum network requirements:

# Prepare a domain for SSRS and PBRS ingestion

You can create a new domain for your SSRS and Power BI Report assets and use the domain ID in the lineage harvester configuration file. As a result, Collibra uses this domain to ingest all SSRS and Power BI assets during the integration process.

## Prerequisites

You have a resource role with the Domain > Add resource permission.

## Steps

1. In the main menu, click the **Create** (+) button.
   » The **Create** dialog box appears.
2. Click the **Organization** tab.

3. Click a domain type from the list.
   If you clicked the wrong domain type here, you can change it in the **Type** field in the next screen.
   » The **Create Domain** dialog box appears.
4. Enter the required information.

| Field | Description |
|---|---|
| Type | The domain type of the domain you are creating. In this case, you need to select *BI Catalog*. |
| Community | The community under which the domain will be located. |
| Name | The name of the new domain. |

5. Click **Create**.
6. Open your domain.
7. Copy the domain ID.

> Tip If you go to your domain, you can find the domain ID in the URL. The URL looks like: https://<yourcollibrainstance>/domain/**22258f64-40b6-4b16-9c08-c95f8ec0da26**?view=00000000-0000-0000-0000-000000040001. In this example, the domain ID is in bold.

8. Paste the domain ID in the lineage harvester configuration file.

> Warning When you run the lineage harvester, Collibra Data Lineage creates all SSRS and Power BI assets in the same Catalog BI domain. We highly recommend that you do not move these assets to another domain. If you move assets to another domain, they will be deleted and recreated in the initial Catalog BI domain when you synchronize SSRS or PBRS. As a consequence, all manually added data of those assets is lost.

# Install the lineage harvester for SSRS-PRBS integration

Before you can use the lineage harvester, you need to download it and install it. You can download the lineage harvester from the Collibra Community downloads page.

> Tip
> - Install the lineage harvester close to your data source or on the same server.
> - The lineage harvester uses port 443.

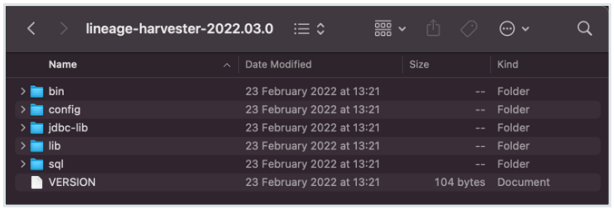
## Prerequisites

- You meet the minimum system requirements.
- You have added Firewall rules so that the lineage harvester can connect to:
  - The host names of all databases in the lineage harvester configuration file.
  - All Collibra Data Lineage service instances within your geographical location:
    - 15.222.200.199 (techlin-aws-ca.collibra.com)
    - 18.198.89.106 (techlin-aws-eu.collibra.com)
    - 54.242.194.190 (techlin-aws-us.collibra.com)
    - 51.105.241.132 (techlin-azure-eu.collibra.com)
    - 20.102.44.39 (techlin-azure-us.collibra.com)
    - 35.197.182.41 (techlin-gcp-au.collibra.com)
    - 34.152.20.240 (techlin-gcp-ca.collibra.com)
    - 35.205.146.124 (techlin-gcp-eu.collibra.com)
    - 34.87.122.60 (techlin-gcp-sg.collibra.com)
    - 35.234.130.150 (techlin-gcp-uk.collibra.com)
    - 34.73.33.120 (techlin-gcp-us.collibra.com)

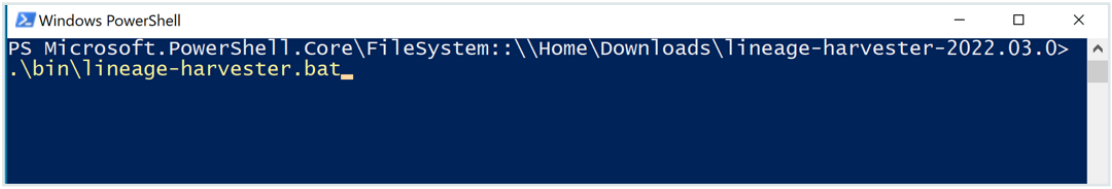
    > Note The lineage harvester connects to different instances based on your geographic location and cloud provider. If your location or cloud provider changes, the lineage harvester rescans all your data sources. You have to whitelist all Collibra Data Lineage service instances in your geographic location. In addition, we highly recommend that you always whitelist the techlin-aws-us instance as a backup, in case the lineage harvester cannot connect to other Collibra Data Lineage service instances.

# Steps

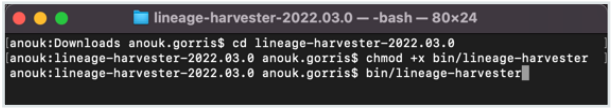
1. Download the newest lineage harvester.
2. Unzip the archive.
   » You can now access the lineage harvester folder.



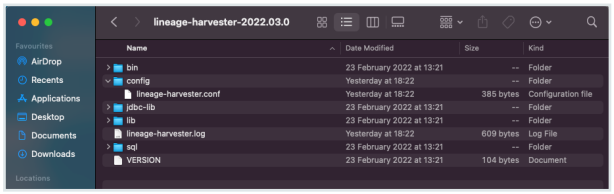

3. Run the following command line to start the lineage harvester:
   - Windows: `.\bin\lineage-harvester.bat`




   - For other operating systems: `chmod +x bin/lineage-harvester` and then `bin/lineage-harvester`




   » An empty configuration file is created in the config folder.




   » The lineage harvester is installed automatically. You can check the installation by running `./bin/lineage-harvester --help`.

# What's next?

You can now prepare the lineage harvester configuration file.

# Prepare the lineage harvester configuration file for SSRS and PBRS integration

You have to prepare a configuration file before you run the lineage harvester. The lineage harvester collects your SQL Server Reporting Services (SSRS) and Power BI Report Server (PBRS) metadata and sends it to Collibra Data Intelligence Cloud, where it is processed and analyzed. Collibra then imports the SSRS and PBRS assets and relations to Data Catalog.

> Tip We recommend that you use the configuration file generator to make sure your configuration file is valid.

## Prerequisites

Ensure that you have completed the following tasks:

- Installed and set up the latest lineage harvester.
- Created a BI Catalog domain in which you want to ingest the SSRS assets.
- Prepared a domain in Collibra.
- Prepared a SSRS and PBRS <source ID> configuration file, if necessary.

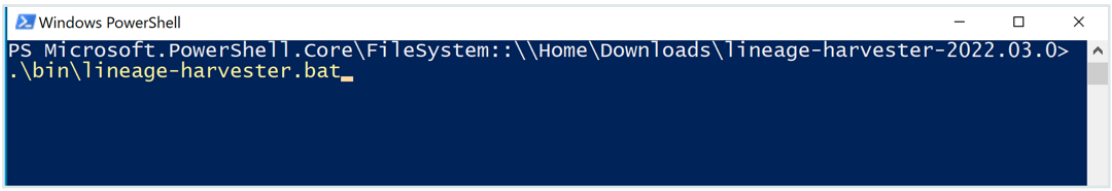
Ensure that you meet the following requirements and have the following permissions:

- Use Collibra Data Intelligence Cloud.
- A global role with the following global permissions:
  - Catalog, for example Catalog Author
  - Data Stewardship Manager
  - Manage all resources
  - System administration
  - Technical lineage
- A resource role with the following resource permission on the community level in which you created the BI Catalog domain:
  - Asset: add
  - Attribute: add
  - Domain: add
  - Attachment: add

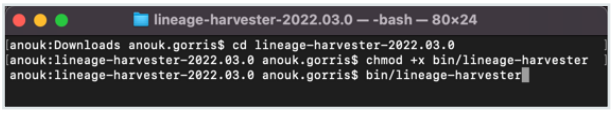
- Use SSRS version 17 or newer.
- A user that has the Content Manager role at the root folder level in SSRS and PBRS.

## Steps

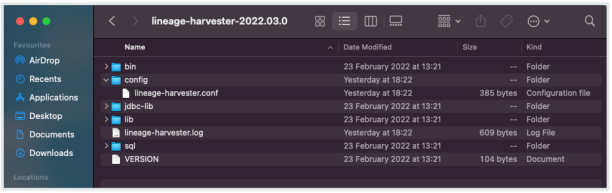
1. Run the following command line to start the lineage harvester:
   - Windows: `.\bin\lineage-harvester.bat`

   - For other operating systems: `chmod +x bin/lineage-harvester` and then `bin/lineage-harvester`

   » An empty configuration file is created in the config folder.

2. Open the lineage-harvester.conf file and enter the values for each property.

| Properties | Description |
|---|---|
| general | This section describes the connection information between the lineage harvester and Data Catalog. |
| catalog | This section contains information that is necessary to connect to Data Catalog. |

| Properties | Description |
| --- | --- |
| url | The URL of your Collibra Data Intelligence Cloud environment.<br><br>> Note You can only enter the public URL of your Collibra Data Intelligence Cloud environment. Other URLs will not be accepted. |
| username | The username that you use to sign in to Collibra. |

| Properties | Description |
| --- | --- |
| useCollibraSystemName | Indication whether you want to use the system or server name of a data source to match to the System asset you created when you prepared the physical data layer. This is useful when you have multiple databases with the same name.<br><br>By default, the `useCollibraSystemName` property is set to `false`. If you want to use it, set it to `true`.<br><br>Important<br>◦ If you set this property to `true`:<br>▪ You must provide a SSRS-PBRS <source ID> configuration file that defines the system name of one or more databases.<br>▪ The lineage harvester reads the value of the `collibraSystemName` property in the <source-ID> configuration file.<br>◦ If you set the `useCollibraSystemName` property to `false`, the lineage harvester ignores the `collibraSystemName` property in the <source-ID> configuration file. |

| Properties | Description |
| --- | --- |
| useSharedDbModel | Optional property to enable the sharing of metadata batches from multiple SQL data sources. Set this property to `true`, to help avoid potential analysis errors on the Collibra Data Lineage service.<br><br>To use this property, you need lineage harvester 2022.07 or newer.<br><br>If you set this property to `true`, you have to run the lineage harvester twice. Read the following details about the issue and solution.<br><br>**See details about the issue and solution**<br>Normally, when you run the lineage harvester to harvest metadata from two or more data sources, the metadata from each source is processed independently. This means that the metadata from one data source cannot access the metadata of another.<br><br>Let's say, for example, you specify the following two SQL data sources in your lineage harvester configuration file:<br>◦ A database source that retrieves the database model.<br>◦ An SqlDirectory source with Data Manipulation Language (DML) statements that reference data in the database source.<br><br>Because these data sources are processed independently, there is a good |

| Properties | Description |
| --- | --- |
| | chance that the DML statements will fail during analysis. Any wildcards in the DML statements, for example, would fail because the SqlDirectory source can't access the referenced database source.<br><br>**The solution**<br><br>The shared database model allows for computed results from a "main" batch. Although multiple data sources are still processed independently, the metadata from each data source is merged into a main batch. Then, before analyzing the next batch, a check is done to see if a preceding main batch exists. If one does, the analyzer retrieves the database model and the DML statements successfully pass analysis.<br><br>This means, however, that you have to run the lineage harvester twice. On the first run, the harvested metadata is merged in a main batch. Then, when you run the lineage harvester again, using the full-sync command, the subsequent batches are able to successfully reference the metadata in the main batch.<br><br>In a future version of Collibra, this property will be enabled by default and you won't need to run the lineage harvester twice. |

| Properties | Description |
| --- | --- |
| sources | This section contains all SSRS connection properties. |
| collibraSystemName | Regardless of the value set for the `useCollibraSystemName` property, the following is true:<br>◦ You must include this property in your configuration file.<br>◦ You can leave this property empty.<br>◦ Any value that you give is ignored.<br><br>If you want to specify name of the system or server, because, for example, you have multiple databases with the same name, then:<br><br>a. Set the `useCollibraSystemName` property to `true`.<br>b. Specify the system or server names in the `collibraSystemName` property in you SSRS-PBRS <source ID> configuration file.<br><br>Note This is a legacy property that will be deprecated in a future release. |

| Properties | Description |
| --- | --- |
| id | The unique ID to identify the SSRSmetadata that was uploaded to the Collibra Data Lineage service.<br><br>Tip This value can be anything as long as it is a unique. The lineage harvester uses the ID to identify a batch of data on the Collibra Data Lineage service.<br><br>Warning In the `sources` section of your lineage harvester configuration file, you can only specify one `id` property per SQL Server Reporting Service (SSRS) or Power BI Report Server (PBRS). If you have multiple `id` properties for a single SSRS or PBRS, ingestion will fail. If you have multiple `id` properties in the configuration file, it means you intend to ingest from multiple unique SSRS or PBRS. |
| type | The kind of data source. In this case, the value has to be *SSRS* or *PBIRS*.<br><br>Note There is no difference between type SSRS or PBIRS. |
| url | The URL to the server's web portal. By default, the URL is *http://<computer-name>/reports*. For example, "http://1.23.45.678/PowerBIReports". |

| Properties | Description |
| --- | --- |
| username | The username you use to sign in to the web portal.<br><br>Tip If you use NTLM authentication, your username also contains the NTLM domain name. For example `MyDomain\\username`. |
| domainId | The unique ID of the domaindomain in Collibra Data Intelligence Cloud in which you want to ingest the SSRS assets.<br><br>Finding the domain ID<br>a. Open the domain.<br>b. Copy the domain ID.<br><br>Tip If you go to your domain, you can find the domain ID in the URL. The URL looks like: https://<yourcollibrainstance>/domain/**22258f64-40b6-4b16-9c08-c95f8ec0da26**?view=00000000-0000-0000-0000-000000040001. In this example, the domain ID is in bold. |

| Properties | Description |
| --- | --- |
| folderFilter | An option to exclude specific folders that contain reports or KPIs from the ingestion process.<br><br>You can add multiple folders by listing folder names, providing the full path to folders or by using a wildcard:<br><br>◦ Use folder names when the folder name is unique: ["folder 1", "folder 2"]<br>◦ Use the full path to the folder to only ingest a specific folder: ["/database1/folder1", "/database2/folder2"]<br>◦ Use a wildcard to ingest all child folders or a specific folder: ["/folder1/*", "/folder2/*"]<br>You can also use a combination of these methods. For example, ["folder 1", "/database/folder2", /folder3/*"]<br><br>Important This property must be included in your configuration file and it cannot be empty. If you want to ingest all folders, use `*`, for example: `"folderFilter":["*"]`.<br><br>Tip For more information about connecting to a SSRS or PBRS folder, see the Microsoft documentation. |

| Properties | Description |
| --- | --- |
| deleteRawMetadataAfterProcessing | The lineage harvester harvests metadata from specified data sources and uploads it in a ZIP file to a Collibra Data Lineage service instance, for processing.<br><br>You can use this optional property to specify whether or not the metadata should be deleted after it has been processed.<br><br>If the property is set to `true`, the metadata is deleted after processing. If set to `false`, the metadata is stored in an Amazon S3 bucket. |

3. Save the configuration file.
4. Start the lineage harvester again in the console and run the following command:
   - for Windows: `.\bin\lineage-harvester.bat full-sync`
   - for other operating systems: `./bin/lineage-harvester full-sync`
5. When prompted, enter the passwords to connect to Collibra and SSRS. Do one of the following:
   - Enter the passwords in the console.
     » The passwords are encrypted and stored in **/config/pwd.conf**.
   - Provide the passwords via command line.
     » The passwords are stored locally and not in your lineage harvester folder.

## Example

```
{
 "general": {
   "catalog": {
     "url": "https://<organization>.collibra.com",
     "username": "<your-collibra-username>"
   },
   "useCollibraSystemName": false,
   "useSharedDbModel": true
 },
  "sources": {
    "collibraSystemName": "",
    "id": "<unique-id>",
```

```
    "type": "SSRS",
    "url": "http://<IP address or computer name>/Reports",
    "username": "<server-api-user-name>",
    "domainId": "<domain-resource-id>",
    "folderFilter":  ["/Folder1/*", "Folder2"],
   "deleteRawMetadataAfterProcessing": true
  }
}
```

> Note There is no difference between type SSRS or PBIRS.

## What's next?

The lineage harvester triggers Collibra to import SSRS and PBRS assets and their relations and create a technical lineage for SSRS assets.Collibra also stitches the new SSRS assets to existing assets in Data Catalog.

> Note We can only access stitching and lineage information for SSRS assets, but not for Power BI reports in SSRS and PBRS.

If issues occur during the ingestion process, check the Collibra Data Lineage troubleshooting section to solve your problems.

To synchronize the SSRS and PBRS metadata, you can run the lineage harvester again or schedule jobs to run them automatically.

> Tip You can check the progress of the ingestion in Activities. The results field indicates how many relations were imported into Data Catalog.

# Prepare a SSRS and PBRS <source ID> configuration file

The lineage harvester uses the lineage harvester configuration file to collect the SQL Server Reporting Services (SSRS) and Power BI Report Server (PBRS) data objects and sends them to the Collibra Data Lineage service. However, if the `useCollibrasystemName` in the lineage harvester configuration file is set to `true`, you

also have to provide a specific <source ID> configuration file that defines the system name of databases in SSRS and Power BI Report Server. This is necessary to enable the Collibra Data Lineage service to process multiple databases with the same name.

The <source ID> configuration file can also be used to provide additional information about databases in SSRS and Power BI Report Server, which is necessary if the databases do not contain all information to process the SQL source code correctly.

## Prerequisites

The `useCollibraSystemName` in the lineage harvester configuration file is set to `true`.

## Steps

1. Create a new JSON file in the lineage harvester **config** folder.
2. Give the JSON file the same name as the value of the `Id` property in the lineage harvester configuration file.

   > Example The value of the `Id` property in the lineage harvester configuration file is `ssrs-source-1`. As a result, the name of your JSON file should be *ssrs-source-1.conf*.

   > Important Your JSON file must have the file extension *.conf*.

3. For each database in SQL Server Reporting Services and Power BI Report Server, add the following content to the JSON file:

| Property | Description | Required? |
|---|---|---|
| DataSources | This section contains all connections for which you want to create a technical lineage.<br><br>The `DataSources` section refers to shared data sources in SSRS and PBRS. For more information about shared data sources, see the Microsoft documentation. | Yes |
| <data source type> | The name of a connection object in SSRS and PBRS. | Yes |
| dbname | The name of the database of a supported data source in SSRS and PBRS. | No |
| schema | The name of the default schema of a supported data source in SSRS and PBRS. | No |
| dialect | The dialect of the supported data source in SSRS and PBRS. | No |
| collibraSystemName | The system or server name of the database. | Yes |

| Property | Description | Required? |
|---|---|---|
| CustomDataSources | You can use custom data processing extensions that are used to support embedded data sources of which the data source definition is specified locally in a report or embedded data set.<br><br>The `CustomDataSources` section refers to embedded data sources in SSRS and PBRS. For more information about embedded data sources, see the Microsoft documentation. | No |
| <path to report>/<custom data source name> | The full path to the report and the custom data source name.<br><br>You can use wildcards to match multiple folders, reports or data sets. The connection information is this section is used to add missing information or to overwrite parsed information. | No |
| dbname | The name of the database of a custom data source in SSRS and PBRS.. | No |
| schema | The name of the schema of a custom data source in power. If you don't provide the schema name, the default schema is used. | No |

| Property | Description | Required? |
| --- | --- | --- |
| dialect | The dialect of the custom data source in SSRS and PBRS.. | No |

| Property | Description | Required? |
| --- | --- | --- |
| | Tip<br>You can enter one of the following values:<br>◦ *azure*, for an Azure SQL Server data source.<br>◦ *bigquery*, for a Google BigQuery data source.<br>◦ *db2*, for an IBM DB2 data source.<br>◦ *hana*, for a SAP Hana data source.<br>◦ *hive*, for a HiveQL data source.<br>◦ *greenplum*, for a Greenplum data source.<br>◦ *mssql*, for a Microsoft SQL Server data source.<br>◦ *mysql*, for a MySQL data source.<br>◦ *netezza*, for a Netezza data source.<br>◦ *oracle*, for an Oracle data source.<br>◦ *postgres*, for a PostgreSQL data source.<br>◦ *redshift*, for an Amazon Redshift data source.<br>◦ *snowflake*, for a Snowflake data source.<br>◦ *spark*, for a Spark SQL data source.<br>◦ *sybase*, for a Sybase data source.<br>◦ *teradata*, for a Teradata data source. | |

| Property | Description | Required? |
| --- | --- | --- |
| . | | |

```
{
   "DataSources": {
       "Redshift": {
           "dbname": "redshift-database-name",
           "schema": "redshift-schema-name",
           "dialect": "redshift",
           "collibraSystemName": "redshift-system-name"
       },
       "Oracle": {
           "dbname": "oracle-database-name",
           "schema": "oracle-schema-name",
           "dialect": "oracle",
           "collibraSystemName": "oracle-system-name"
       }
   },
   "CustomDataSources":
       "/path to report/custom data souce name": {
           "dbname": "mssql-database-name",
           "dialect": "mssql"
       }
   }
}
```

4. Save the <source ID> configuration file.

# Schedule SSRS and PBRS ingestion jobs

You can use Task Scheduler on Windows or Crontab on Mac and Linux to enable the lineage harvester to run scheduled jobs. In a scheduled job, the lineage harvester uploads the SQL Server Reporting Services (SSRS) and Power BI Report Server (PBRS) metadata to Collibra.

Collibra automatically creates new assets and relations at specific times, dates or intervals, using the information in the lineage harvester configuration file.

> Example  You created a lineage harvester configuration file with connection information to your SSRS or PBRS environment. You schedule the lineage harvester job to run each Sunday at 23:00. As a result, your SSRS and PBRS metadata is automatically refreshed on a weekly basis.

Warning When you run the lineage harvester, Collibra Data Lineage creates all SSRS and Power BI assets in the same Catalog BI domain. We highly recommend that you do not move these assets to another domain. If you move assets to another domain, they will be deleted and recreated in the initial Catalog BI domain when you synchronize SSRS or PBRS. As a consequence, all manually added data of those assets is lost.

Warning Relations that were manually created between SSRS and Power BI Report assets and other assets via a relation type in the operating model, are deleted after a refresh of the metadata.

# Working with Looker

Looker is a business intelligence software that helps people see and understand their data.

For more information about Looker, see the Looker documentation.

> Note When you ingest Looker metadata, you automatically create a technical lineage for Looker.



# Looker terminology

Before you ingest Looker, read more about the Looker terminology and how it maps with the Collibra Data Intelligence Cloud asset types.

> Note For more information, see the Looker documentation.

| Looker term | Description | Asset type in Collibra |
|---|---|---|
| Dashboard | A collection of Looker tiles with metrics from one or more Looker Looks. | Looker Dashboard |
| Explore | A collection of data that is used to define Looker Dimensions and Measures. | Looker Data Set |
| Dimensions, Measures | An atomic unit of data that is used in a Looker Look or Looker Tile. It represents a column in a Looker Data Set. | Looker Data Set Column |
| Folder or Space | A container that stores Looker Looks, Dashboards and other folders. | Looker Folder |
| Look | A detailed view of a Looker Data Set, with visualizations of findings and insights. | Looker Look |
| Dimensions, Measures | An atomic unit of data that is used in a Looker Look or Looker Tile. It represents the actual use a Looker Data Set Column. | Looker Report Attribute |
| Query | A query that creates a simple report in a Looker Tile or Looker Look. | Looker Query |
| Looker instance | A platform to create Looker Dashboards and rich visualizations. | Looker Tenant |
| Tile or Dashboard element | An element that represents data on the Looker Dashboard. | Looker Tile |

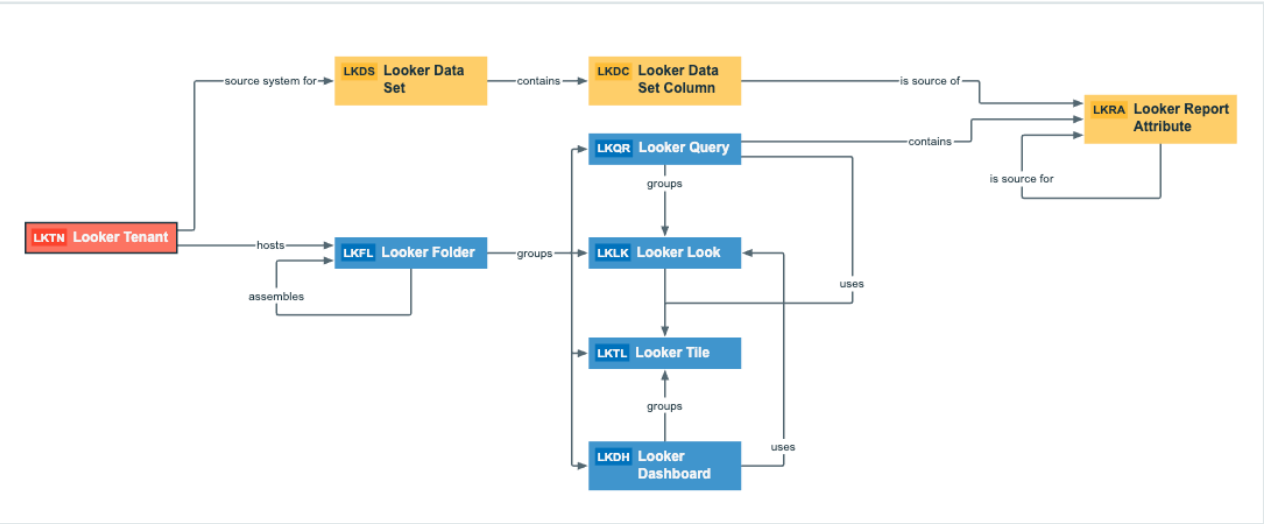
# Looker operating model

The Looker scanner collects Looker metadata and sends it to the Collibra Data Lineage service. Collibra processes the metadata and creates new Looker assets and relations in Data Catalog. You can see them on the asset page overview or visualize them in a diagram or in a technical lineage.

> Note
> - The assets have the same names as their counterparts in Looker. Full names and Display names cannot be changed in Data Catalog.
> - Asset types are only created if you have all specific Looker and Data Catalog permissions.
> - All Looker asset types are created in the same domain.
> - Relations that were manually created between Looker assets and other assets via a relation type in the Looker operating model are deleted after a refresh of the Looker metadata.

# Looker metadata overview

The following image shows the relations between Looker asset types.



# Harvested metadata per asset type

The following table shows the harvested Looker metadata for each Looker asset type. This table also shows the resource ID for each asset type and metadata.

| Asset type | Synchronized metadata | Resource ID |
| --- | --- | --- |
| Looker Dashboard<br><br>Resource ID: 00000000-0000-0000-0000-100000000013 | Full name | |

| Asset type | Synchronized metadata | Resource ID |
| --- | --- | --- |
| | Display name | |

| Asset type | Synchronized metadata | Resource ID |
|---|---|---|
| | Description | 00000000-0000-0000-0000-000000003114 |
| | Document creation date | 00000000-0000-0000-0000-000000000260 |
| | Document last accessed date | 00000000-0000-0000-0000-000000000268 |
| | Favorites count | 00000000-0000-0000-0000-000000000269 |
| | Owner in source<br>The only harvested metadata are email addresses. | 00000000-0000-0000-0000-200000000001 |
| | Technical Data Type | 00000000-0000-0000-0000-000000000219 |
| | URL | 00000000-0000-0000-0000-000000000258 |
| | Visit count | 00000000-0000-0000-0000-000000000264 |
| | Business Dimension groups / is grouped into Report | 00000000-0000-0000-0000-120000000002 |
| | Report groups / is grouped into Report | 00000000-0000-0000-0000-120000000004 |
| | Report related to / impacted by Business Asset | 00000000-0000-0000-0000-120000000006 |

| Asset type | Synchronized metadata | Resource ID |
|---|---|---|
| | Report uses / used in Report | 00000000-0000-0000-0000-120000000007 |
| Looker Data Set<br>Resource ID: 00000000-0000-0000-0000-100000000017 | Full name | |
| | Display name | |
| | Description | 00000000-0000-0000-0000-000000003114 |
| | Data Set contains / is part of Data Element | 00000000-0000-0000-0000-000000007062 |
| | Technology Asset source system for / source system Data Asset | 00000000-0000-0000-0000-000000007050 |
| Looker Data Set Column<br>Resource ID: 00000000-0000-0000-0000-100000000018 | Full name | |
| | Display name | |
| | Description | 00000000-0000-0000-0000-000000003114 |
| | Data Set contains / is part of Data Element | 00000000-0000-0000-0000-000000007062 |
| | Report Attribute sourced from / is source of Data Attribute | 00000000-0000-0000-0000-120000000010 |

| Asset type | Synchronized metadata | Resource ID |
|---|---|---|
| Looker Folder<br><br>Resource ID: 00000000-0000-0000-0000-100000000012 | Full name | |
| | Display name | |
| | Document creation date | 00000000-0000-0000-0000-000000000260 |
| | Owner in source<br><br>The only harvested metadata are email addresses. | 00000000-0000-0000-0000-200000000001 |
| | BI Folder assembles / is assembled in BI Folder | 00000000-0000-0000-0000-120000000001 |
| | Business Dimension groups / is grouped into Report | 00000000-0000-0000-0000-120000000002 |
| | Server hosts / is hosted in Business Dimension | 00000000-0000-0000-0000-120000000000 |

| Asset type | Synchronized metadata | Resource ID |
| --- | --- | --- |
| Looker Look<br><br>Resource ID: 00000000-0000-0000-0000-100000000014 | Full name | |

| Asset type | Synchronized metadata | Resource ID |
| --- | --- | --- |
| | Display name | |

| Asset type | Synchronized metadata | Resource ID |
|---|---|---|
| | Description | 00000000-0000-0000-0000-000000003114 |
| | Document creation date | 00000000-0000-0000-0000-000000000260 |
| | Document last accessed date | 00000000-0000-0000-0000-000000000268 |
| | Document modification date | 00000000-0000-0000-0000-000000000261 |
| | Favorites count | 00000000-0000-0000-0000-000000000269 |
| | Owner in source<br>The only harvested metadata are email addresses. | 00000000-0000-0000-0000-200000000001 |
| | Report image | 00000000-0000-0000-0000-000000000262 |
| | URL | 00000000-0000-0000-0000-000000000258 |
| | Visit count | 00000000-0000-0000-0000-000000000264 |
| | Business Dimension groups / is grouped into Report | 00000000-0000-0000-0000-120000000002 |
| | Report groups / is grouped into Report | 00000000-0000-0000-0000-120000000004 |

| Asset type | Synchronized metadata | Resource ID |
|---|---|---|
| | Report uses / used in Report | 00000000-0000-0000-0000-120000000007 |
| Looker Report Attribute<br>Resource ID: 00000000-0000-0000-0000-100000000019 | Full name | |
| | Display name | |
| | Report Attribute contained in / contains Report | 00000000-0000-0000-0000-000000007058 |
| | Report Attribute sourced from / is source of Data Attribute | 00000000-0000-0000-0000-120000000010 |
| Looker Query<br>Resource ID: 00000000-0000-0000-0000-100000000016 | Full name | |
| | Display name | |
| | URL | 00000000-0000-0000-0000-000000000258 |
| | Business Dimension groups / is grouped into Report | 00000000-0000-0000-0000-120000000002 |
| | Report Attribute contained in / contains Report | 00000000-0000-0000-0000-000000007058 |
| | Report uses / used in Report | 00000000-0000-0000-0000-120000000007 |

| Asset type | Synchronized metadata | Resource ID |
|---|---|---|
| Looker Tenant<br>Resource ID: 00000000-0000-0000-0000-100000000011 | Full name | |
| | Display name | |
| | Description | 00000000-0000-0000-0000-000000003114 |
| | Server hosts / is hosted in Business Dimension | 00000000-0000-0000-0000-120000000000 |
| | Technology Asset source system for / source system Data Asset | 00000000-0000-0000-0000-000000007050 |
| Looker Tile<br>Resource ID: 00000000-0000-0000-0000-100000000015 | Full name | |
| | Display name | |
| | Business Dimension groups / is grouped into Report | 00000000-0000-0000-0000-120000000002 |
| | Report uses / used in Report | 00000000-0000-0000-0000-120000000007 |

Note The metadata that is shown on the assets' pages depends on the asset type's assignment. As a result, you might not see all harvested metadata on the asset's page by default.

# Additional information

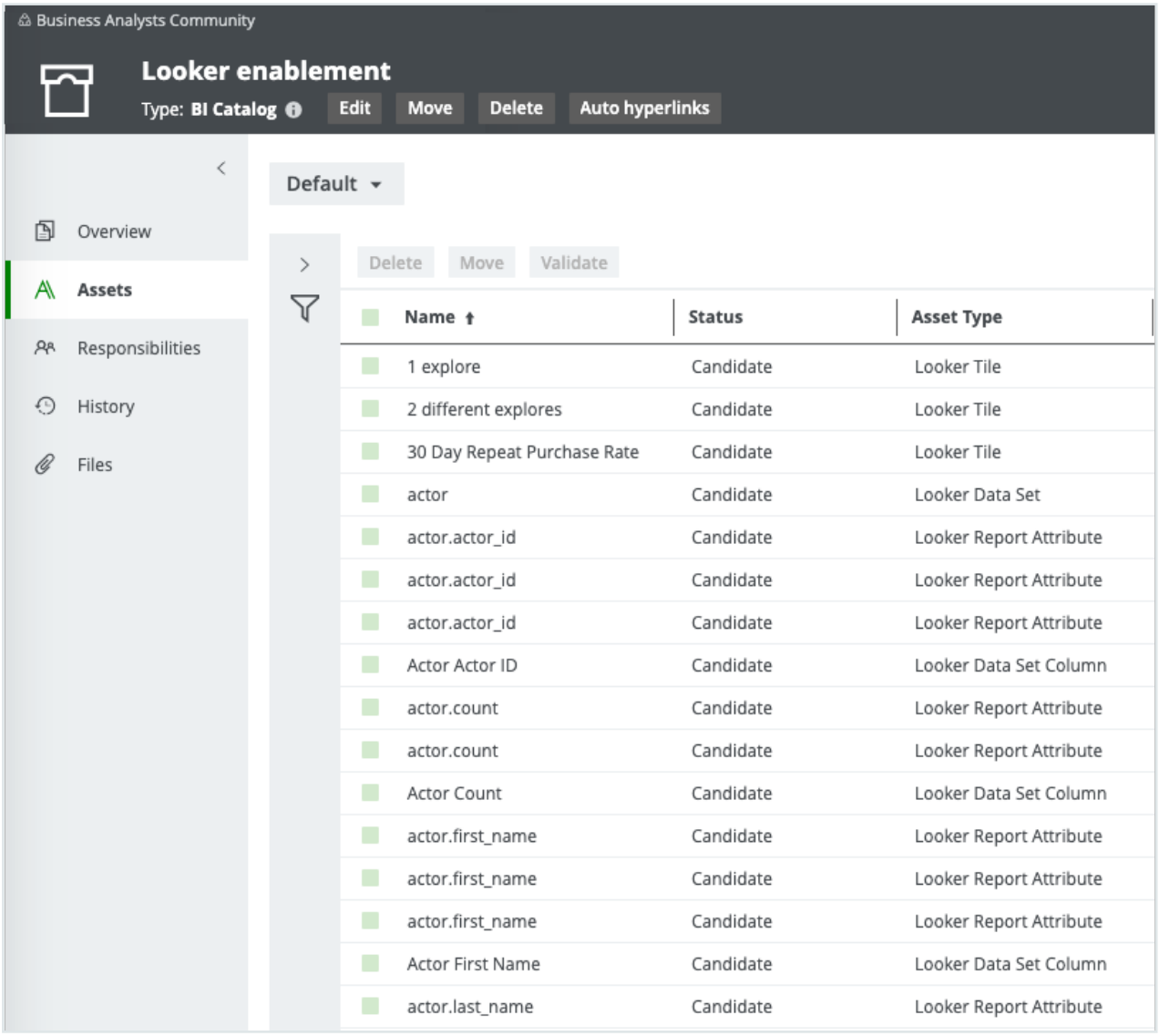
For the Owner in source attribute, the following rules apply:

- If the system creates a Looker data object and the Looker data object does not have a user ID, the Owner in source attribute is shown as System on the asset page.

- If the user who created a Looker data object no longer exists, the Owner in source attribute is shown as empty on the asset page.

# Example of ingested Looker metadata

The following image shows an example structure after Looker ingestion.



# Looker asset and domain types

The Looker integration in Collibra Data Intelligence Cloud uses a specific subset of asset types and domain types. All of these come out of the box with your software.

The following table contains the asset and domain types that are used for the Looker integration. Above each asset type you can see the parent asset types in the breadcrumbs.

| Asset type | Description | Domain type |
|---|---|---|
| Business Asset ▸ Business Dimension ▸ BI Folder ▸ Looker Folder | A container that stores Looker Looks, Dashboards and other folders. | BI Catalog |
| Business Asset ▸ Report ▸ BI Report ▸ Looker Dashboard | A collection of Looker tiles with metrics from one or more Looker Looks. | BI Catalog |
| Business Asset ▸ Report ▸ BI Report ▸ Looker Look | A detailed view of a Looker Data Set, with visualizations of findings and insights. | BI Catalog |
| Business Asset ▸ Report ▸ BI Report ▸ Looker Query | A query that creates a simple report in a Looker Tile or Looker Look. | BI Catalog |

| Asset type | Description | Domain type |
| --- | --- | --- |
| Business Asset ▸ Report ▸ BI Report ▸ Looker Tile | An element that represents data on the Looker Dashboard. | BI Catalog |
| Data Asset ▸ Data Element ▸ Data Attribute ▸ BI Data Attribute ▸ Looker Data Set Column | An atomic unit of data that is used in a Looker Look or Looker Tile. It represents a column in a Looker Data Set. | BI Catalog |
| Data Asset ▸ Data Element ▸ Report Attribute ▸ BI Report Attribute ▸ Looker Report Attrib-ute | An atomic unit of data that is used in a Looker Look or Looker Tile. It represents the actual use a Looker Data Set Column. | BI Catalog |
| Data Asset ▸ Data Set ▸ BI Data Set ▸ Looker Data Set | A collection of data that is used to define Looker Dimensions and Measures. | BI Catalog |
| Technology Asset ▸ Server ▸ BI Server ▸ Looker Tenant | A platform to create Looker Dashboards and rich visualizations. | BI Catalog |

# Overview Looker integration steps

The Looker integration enables you to harvest Looker metadata and create new Looker assets in Data Catalog. Collibra analyzes and processes the Looker metadata and presents it as specific asset types, retaining their original names.

> Tip To ingest Looker metadata in Data Catalog, you need to run the lineage harvester. The Looker ingestion workflow explains the role of the lineage harvester in the Looker ingestion process.

## Steps

The table below shows the steps and prerequisites required to integrate Looker in Data Catalog.

> Important In the global assignment of each asset type included in the Looker operating model, ensure that none of the characteristics that are in the operating model have a maximum cardinality of “0”. If the maximum cardinality is set to “0” for any such characteristics, ingestion will fail.

| Step | What? | Description | Prerequisites |
|---|---|---|---|
| 1 | Set up Looker authentication. | Before you start the Looker integration, you have to enable Collibra to access your Looker metadata. | • You have a Looker subscription. |
| 2 | Create a new domain. | Before you can ingest Looker metadata, you have to create a new domain or choose an existing domain to store the new Looker assets. | • You have a resource role with the following resource per-missions:<br>◦ Domain: Add |

| Step | What? | Description | Prerequisites |
|---|---|---|---|
| 3 | Download and install the lineage harvester and prepare a configuration file with Looker connection properties. | You use the lineage harvester to collect metadata from Looker and upload it to Collibra, where the metadata is scanned, processed and analyzed.<br><br>When you download the lineage harvester, you can access the configuration file. You prepare a configuration file with Looker connection properties.<br><br>> Note You need the lineage harvester 1.3.0 or newer to ingest Looker metadata into Data Catalog | • You have access to the lineage harvester 1.3.0 or newer..<br>• Your environment meets the system requirements to install and use the lineage harvester.<br>• You have a global role that has the Manage all resources global permission.<br>• You have a global role with the Catalog global permission, for example Catalog Author.<br>• You have a global role with the Technical lineage global permission.<br>• You have a global role with the Data Stewardship Manager global permission.<br>• A resource role with the following resource permission on the community level in which you created the BI Data Catalog domain:<br>  ◦ Asset: add<br>  ◦ Attribute: add |

| Step | What? | Description | Prerequisites |
| --- | --- | --- | --- |
| | | | ◦ Domain: add<br>◦ Attachment: add |

| Step | What? | Description | Prerequisites |
|---|---|---|---|
| 4 | Run the lineage harvester | You run the lineage harvester to start the ingestion process.<br><br>Collibra creates new Looker assets in Data Catalog and imports relations between these assets. It also creates a technical lineage for Looker Look assets.<br><br>You can create a lineage harvester job to schedule automatic Looker ingestion and synchronization. | • You have Collibra Data Intelligence Cloud 2020.12 or newer.<br>• Your environment meets the system requirements to run the lineage harvester.<br>• You have added Firewall rules so that the lineage harvester can connect to Collibra Data Lineage service instances with the following IP addresses:<br>◦ 15.222.200.199 (techlin-aws-ca.collibra.com)<br>◦ 18.198.89.106 (techlin-aws-eu.collibra.com)<br>◦ 54.242.194.190 (techlin-aws-us.collibra.com)<br>◦ 51.105.241.132 (techlin-azure-eu.collibra.com)<br>◦ 20.102.44.39 (techlin-azure-us.collibra.com)<br>◦ 35.197.182.41 (techlin-gcp-au.collibra.com)<br>◦ 34.152.20.240 |

| Step | What? | Description | Prerequisites |
| --- | --- | --- | --- |
| | | | (techlin-gcp-ca.-collibra.com)<br>◦ 35.205.146.124 (techlin-gcp-eu.-collibra.com)<br>◦ 34.87.122.60 (techlin-gcp-sg.-collibra.com)<br>◦ 35.234.130.150 (techlin-gcp-uk.-collibra.com)<br>◦ 34.73.33.120 (techlin-gcp-us.-collibra.com) |

| Step | What? | Description | Prerequisites |
|---|---|---|---|
| 4 | View the Looker assets and technical lineage | After the Looker metadata is ingested in Data Catalog, you can go to the domain where you ingested Looker and see the list of ingested Looker assets.<br><br>You can go to a Looker Look asset page and click the Technical lineage lineage tab to view the technical lineage.<br><br>Warning When you run the lineage harvester, Collibra Data Lineage creates all Looker assets in the specified domain (or domains) in Collibra. We highly recommend that you do not move these assets to other domains. If you move assets to other domains, they will be deleted and recreated in the initial Data Catalog BI domain (or domains) when you synchronize Looker. As a consequence, all manually added data of those assets is lost. | • You have a global role with the Technical lineage global permission.<br>• You have a global role with the Catalog global permission, for example Catalog Author. |

# Authentication

The Looker integration process uses a Looker API. To access the Looker metadata, the Looker API uses API3 credentials for authorization and access control.

## Prerequisite

- You have the necessary permissions in Looker to see the Looker data.

## Steps

1. Create a user with the Admin role.

   > Tip Only a user with a role that has the Admin permission set can create API3 credentials. Some Looker API calls also require a role that has the Admin permission set.

2. Create the API3 credentials.
3. Use the API3 credentials in the configuration file.

> Note API3 credentials are always linked to a Looker user account. As a result, calls to the API only return data that the user is allowed to see.

> Tip For more information, see the Looker documentation.

# Prepare a domain for Looker ingestion

You can create one or more domains for your Looker assets. You then specify:

- One domain reference ID in the lineage harvester configuration file. This domain is the default domain.
- If you want to ingest the contents of specific Looker Folders into specific domains in Collibra, you specify the domain reference IDs in the filters section of the Looker <source ID> configuration file.

Collibra uses the specified domain (or domains) to ingest all Looker assets during the Looker integration process.

# Prerequisites

- You have a resource role with the Domain > Add resource permission.

# Steps

1. In the main menu, click the **Create** (+) button.
   » The **Create** dialog box appears.
2. Click the **Organization** tab.
3. Click a domain type from the list.
   If you clicked the wrong domain type here, you can change it in the **Type** field in the next screen.
   » The **Create Domain** dialog box appears.
4. Enter the required information.

| Field | Description |
|---|---|
| Type | The domain type of the domain you are creating. In this case, you need to select *BI Catalog*. |
| Community | The community under which the domain will be located. |
| Name | The name of the new domain or domains.<br><br>Tip<br>You can create multiple domains in one go.<br>To do this, press `Enter` after typing a value and then type the next. Domain names have to be unique in their parent community. If you type a name that already exists, it will appear in strike-through style. |

5. Click **Create**.
6. Open your domain. If you created multiple domains, open each of them in turn.

7. Copy the reference ID of each domain you created.

   > Tip If you go to your domain, you can find the reference ID in the URL. The URL looks like: https://<yourcollibrainstance>/domain/**22258f64-40b6-4b16-9c08-c95f8ec0da26**?view=00000000-0000-0000-0000-000000040001. In this example, the reference ID is in bold.

8. Paste a reference ID in the `domainId` property in your lineage harvester configuration file. This is your default domain.

   > Note Metadata from Looker data sets is ingested in the default domain, even if you configure filtering.

9. If you want to ingest the contents of specific Looker Folders into specific domains in Collibra, you specify the domain reference IDs in the filters section of the Looker <source ID> configuration file.

> Warning When you run the lineage harvester, Collibra Data Lineage creates all Looker assets in the specified domain (or domains) in Collibra. We highly recommend that you do not move these assets to other domains. If you move assets to other domains, they will be deleted and recreated in the initial Data Catalog BI domain (or domains) when you synchronize Looker. As a consequence, all manually added data of those assets is lost.

# The lineage harvester setup for Looker

The lineage harvester is a software application that is needed to collect your Looker metadata and send it to the Collibra Data Lineage service, where the metadata is processed and new Looker assets and relations are created. Collibra Data Intelligence Cloud then import those assets and relations into Data Catalog.

For more information about the lineage harvester, read the Collibra Data Lineage section.

If you purchased Collibra Data Lineage, you have access to the lineage harvester on the Collibra downloads page.

For more information about the lineage harvester, read the Collibra Data Lineage section.

> Note You need the lineage harvester 1.3.0 or newer to ingest Looker metadata into Data Catalog

# Lineage harvester system requirements

You need to meet the system requirements to be able to install and run the lineage harvester.

## Software requirements

Java Runtime Environment version 11 or newer, or OpenJDK 11 or newer.

For Java Runtime Environment 16 or newer, or OpenJDK 16 or newer, set the `JAVA_OPTS` environment variable for the lineage harvester to function properly:

```
JAVA_OPTS='--illegal-access=deny'
```

> Note To ingest Snowflake data sources, the minimum requirement is Java Runtime Environment version 16 or newer, or OpenJDK 16 or newer.

## Hardware requirements

You need to meet the hardware requirements to install and run the lineage harvester.

### Minimum hardware requirements

You need the following minimum hardware requirements:

- 2 GB RAM
- 1 GB free disk space

## Recommended hardware requirements

The minimum requirements are most likely insufficient for production environments. We recommend the following hardware requirements:

- 4 GB RAM

  > Tip 4 GB RAM is sufficient in most cases, but more memory could be needed for larger harvesting tasks. For instructions on how to increase the maximum heap size, see Technical lineage general troubleshooting.

- 20 GB free disk space

# Network requirements

The lineage harvester uses the HTTPS protocol by default and uses port 443.

You need the following minimum network requirements:

- Firewall rules so that the lineage harvester can connect to:
  - Your Collibra Data Intelligence Cloud instance version 2020.12 or newer.
  - All Collibra Data Lineage service instances in your geographic location:
    - 15.222.200.199 (techlin-aws-ca.collibra.com)
    - 18.198.89.106 (techlin-aws-eu.collibra.com)
    - 54.242.194.190 (techlin-aws-us.collibra.com)
    - 51.105.241.132 (techlin-azure-eu.collibra.com)
    - 20.102.44.39 (techlin-azure-us.collibra.com)
    - 35.197.182.41 (techlin-gcp-au.collibra.com)
    - 34.152.20.240 (techlin-gcp-ca.collibra.com)
    - 35.205.146.124 (techlin-gcp-eu.collibra.com)
    - 34.87.122.60 (techlin-gcp-sg.collibra.com)
    - 35.234.130.150 (techlin-gcp-uk.collibra.com)
    - 34.73.33.120 (techlin-gcp-us.collibra.com)

> Note The lineage harvester connects to different Collibra Data Lineage service instances based on your geographic location and cloud provider. If your location or cloud provider changes, the lineage harvester rescans all your data sources. You have to whitelist all Collibra Data Lineage service instances in your geographic location. In addition, we highly recommend that you always whitelist the techlin-aws-us instance as a backup, in case the lineage harvester cannot connect to other Collibra Data Lineage service instances.

# Install the lineage harvester for Looker integration

Before you can use the lineage harvester, you need to download it and install it. You can download the lineage harvester from the Collibra Community downloads page.

> Tip
> - Install the lineage harvester close to your data source or on the same server.
> - The lineage harvester uses port 443.

## Prerequisites

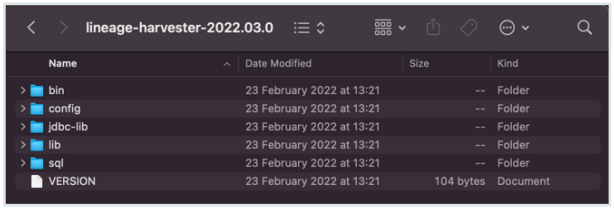
- You have purchased the Looker metadata connector and lineage feature.
- You have Collibra Data Intelligence Cloud 2020.12 or newer.
- You meet the minimum system requirements.
- You have added Firewall rules so that the lineage harvester can connect to:
  - The host names of all databases in the lineage harvester configuration file.
  - All Collibra Data Lineage service instances within your geographical location:
    - 15.222.200.199 (techlin-aws-ca.collibra.com)
    - 18.198.89.106 (techlin-aws-eu.collibra.com)
    - 54.242.194.190 (techlin-aws-us.collibra.com)
    - 51.105.241.132 (techlin-azure-eu.collibra.com)
    - 20.102.44.39 (techlin-azure-us.collibra.com)
    - 35.197.182.41 (techlin-gcp-au.collibra.com)
    - 34.152.20.240 (techlin-gcp-ca.collibra.com)
    - 35.205.146.124 (techlin-gcp-eu.collibra.com)

- 34.87.122.60 (techlin-gcp-sg.collibra.com)
- 35.234.130.150 (techlin-gcp-uk.collibra.com)
- 34.73.33.120 (techlin-gcp-us.collibra.com)

> Note The lineage harvester connects to different instances based on your geographic location and cloud provider. If your location or cloud provider changes, the lineage harvester rescans all your data sources. You have to whitelist all Collibra Data Lineage service instances in your geographic location. In addition, we highly recommend that you always whitelist the techlin-aws-us instance as a backup, in case the lineage harvester cannot connect to other Collibra Data Lineage service instances.

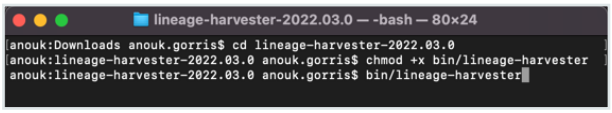
## Steps

1. Download the newest lineage harvester.
2. Unzip the archive.
   » You can now access the lineage harvester folder.

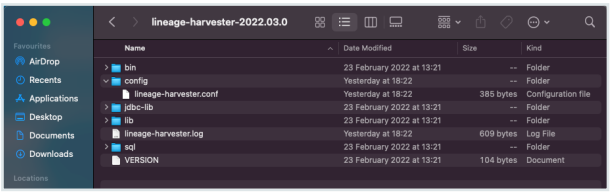



3. Run the following command line to start the lineage harvester:
   - Windows: `.\bin\lineage-harvester.bat`


   - For other operating systems: `chmod +x bin/lineage-harvester` and then `bin/lineage-harvester`



» An empty configuration file is created in the config folder.




» The lineage harvester is installed automatically. You can check the installation by running `./bin/lineage-harvester --help`.

## What's next?

You can now prepare the lineage harvester configuration file.

# Prepare the lineage harvester configuration file for Looker

You have to prepare a configuration file before you run the lineage harvester. The lineage harvester collects your Looker metadata and sends it to the Collibra Data Lineage service, where it is processed and analyzed. Collibra Data Intelligence Cloud then imports the Looker assets and relations to Data Catalog.

## Prerequisites

Ensure that you have completed the following tasks:

- Installed and set up the latest lineage harvester.
- Created one or more BI Catalog domains in which you want to ingest the Looker assets.

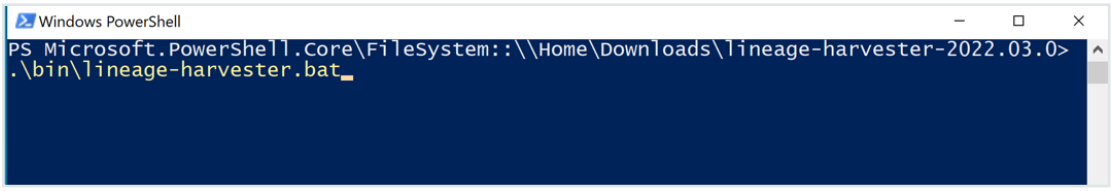
Ensure that you meet the following requirements and have the following permissions:

- Use Collibra Data Intelligence Cloud.
- A global role with the following global permissions:
  - Catalog, for example Catalog Author
  - Data Stewardship Manager
  - Manage all resources

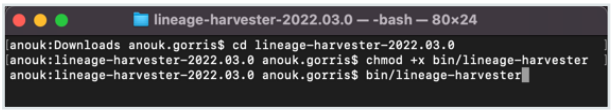
    - System administration
    - Technical lineage
- A resource role with the following resource permission on the community level in which you created the BI Catalog domain:
    - Asset: add
    - Attribute: add
    - Domain: add
    - Attachment: add

## Steps

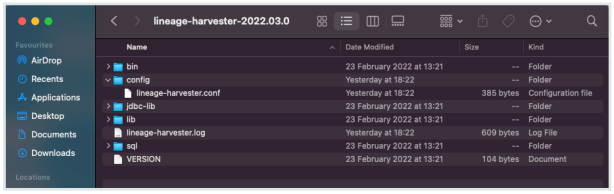
1. Run the following command line to start the lineage harvester:
    - Windows: `.\bin\lineage-harvester.bat`




    - For other operating systems: `chmod +x bin/lineage-harvester` and then `bin/lineage-harvester`




    » An empty configuration file is created in the config folder.




2. Open the lineage-harvester.conf file and enter the values for each property.

| Properties | Description |
|---|---|
| general | This section describes the connection information between the lineage harvester and Data Catalog. |

| Properties | Description |
| --- | --- |
| catalog | This section contains information that is necessary to connect to Data Catalog. |
| url | The URL of your Collibra Data Intelligence Cloud environment.<br><br>Note You can only enter the public URL of your Collibra DGC environment. Other URLs will not be accepted. |
| username | The username that you use to sign in to Collibra. |
| useCollibraSystemName | By default, the `useCollibraSystemName` property is set to `false`. This property is not valid for Looker integration. We recommend that you leave this property set to `false`. |

| Properties | Description |
| --- | --- |
| useSharedDbModel | Optional property to enable the sharing of metadata batches from multiple SQL data sources. Set this property to `true`, to help avoid potential analysis errors on the Collibra Data Lineage service.<br><br>To use this property, you need lineage harvester 2022.07 or newer.<br><br>If you set this property to `true`, you have to run the lineage harvester twice. Read the following details about the issue and solution.<br><br>**See details about the issue and solution**<br>Normally, when you run the lineage harvester to harvest metadata from two or more data sources, the metadata from each source is processed independently. This means that the metadata from one data source cannot access the metadata of another.<br><br>Let's say, for example, you specify the following two SQL data sources in your lineage harvester configuration file:<br>◦ A database source that retrieves the database model.<br>◦ An SqlDirectory source with Data Manipulation Language (DML) |

| Properties | Description |
| --- | --- |
| | statements that reference data in the database source.<br><br>Because these data sources are processed independently, there is a good chance that the DML statements will fail during analysis. Any wildcards in the DML statements, for example, would fail because the SqlDirectory source can't access the referenced database source.<br><br>**The solution**<br><br>The shared database model allows for computed results from a "main" batch. Although multiple data sources are still processed independently, the metadata from each data source is merged into a main batch. Then, before analyzing the next batch, a check is done to see if a preceding main batch exists. If one does, the analyzer retrieves the database model and the DML statements successfully pass analysis.<br><br>This means, however, that you have to run the lineage harvester twice. On the first run, the harvested metadata is merged in a main batch. Then, when you run the lineage harvester again, using the full-sync command, the subsequent batches are able to |

| Properties | Description |
| --- | --- |
| | successfully reference the metadata in the main batch.<br><br>In a future version of Collibra, this property will be enabled by default and you won't need to run the lineage harvester twice. |
| sources | This section contains all Looker connection properties. |
| collibraSystemName | This property is deprecated for Looker integration. The lineage harvester does not take into account any value that you enter here. |

| Properties | Description |
| --- | --- |
| id | The unique ID of your Looker metadata. For example, *my_looker*.<br><br>Tip This value can be anything as long as it is unique and human readable. The ID identifies the batch of Looker metadata on the Collibra Data Lineage service.<br><br>Warning In the `sources` section of your lineage harvester configuration file, you can only specify one `id` property per Looker instance. If you have multiple `id` properties for a single Looker instance, ingestion will fail. If you have multiple `id` properties in the configuration file, it means you intend to ingest from multiple unique Looker instances. |
| type | The kind of data source. In this case, the value has to be *Looker*. |

| Properties | Description |
| --- | --- |
| lookerUrl | The URL to your Looker API.<br><br>Tip There are two ways to find the Looker API URL:<br>◦ In the API Host URL field in the Looker Admin menu. If this field is empty, you can use the default Looker API URL which you can find in the interactive API documentation.<br>◦ In the interactive API documentation URL. It is the part of the URL before `/api-docs/`.<br><br>Note Looker 3.1 APIs are deprecated; however, the API3 credentials for authorization and access control remain valid. |
| clientId | The username you use to access the Looker API. |

| Properties | Description |
| --- | --- |
| domainId | The unique ID of the domain in Collibra Data Intelligence Cloud in which you want to ingest the Looker assets.<br><br>This is the default domain.<br><br>If you want to ingest the contents of specific Looker Folders into specific domains in Collibra, you specify the domain reference IDs in the filters section of the Looker <source ID> configuration file. |
| deleteRawMetadataAfterProcessing | The lineage harvester harvests metadata from specified data sources and uploads it in a ZIP file to a Collibra Data Lineage service instance, for processing.<br><br>You can use this optional property to specify whether or not the metadata should be deleted after it has been processed.<br><br>If the property is set to `true`, the metadata is deleted after processing. If set to `false`, the metadata is stored in an Amazon S3 bucket. |

3. Save the configuration file.
4. Start the lineage harvester again in the console and run the following command:
   - for Windows: `.\bin\lineage-harvester.bat full-sync`
   - for other operating systems: `./bin/lineage-harvester full-sync`

5. When prompted, enter the password or client secret to connect to your Collibra Data Intelligence Cloud and Looker environment.
   » The passwords are encrypted and stored in **/config/pwd.conf**.

## Example

```
{
 "general": {
  "catalog": {
   "url": "https://<organization>.collibra.com",
   "userName": "<your-collibra-username>"
  },
  "useCollibraSystemName": false,
  "useSharedDbModel": true
 },
 "sources": [{
  "collibraSystemName" : "",
  "id": "<looker-id>",
  "type": "Looker",
  "lookerUrl": "<https://<instance-name>.api.looker.com",
  "clientId": "<looker-api-user-name>",
  "clientSecret": "<looker-api-userkey",
  "domainId": "<domain-resource-id>",
  "deleteRawMetadataAfterProcessing": true
 }]
}
```

## What's next?

The lineage harvester triggers Collibra to import Looker assets and their relations and create a technical lineage for Looker Look assets.

Currently, Looker assets are not yet stitched to other assets in Data Catalog.

If issues occur during the Looker ingestion process, check the Looker troubleshooting section to solve your problems.

To refresh the Looker metadata, you can run the lineage harvester again or schedule jobs to run them automatically.

> Tip You can check the progress of the Looker ingestion in Activities. The results field indicates how many relations were imported into Data Catalog.

# Prepare Looker <source ID> configuration file

The lineage harvester uses the lineage harvester configuration file to collect the Looker data objects and sends them to the Collibra Data Lineage service. However, if the `useCollibraSystemName` in the lineage harvester configuration file is set to `true`, you also have to provide a specific <source ID> configuration file that defines the system name of databases in Looker.

Collibra Data Lineage uses the system names to match the structure of databases in Looker to assets in Data Catalog.

> Tip The name <source ID> configuration file refers to the value of the `Id` property in the lineage harvester configuration file.

## Prerequisites

- The `useCollibraSystemName` in the lineage harvester configuration file is set to `true`.

## Steps

1. Create a new JSON file in the lineage harvester **config** folder.
2. Give the JSON file the same name as the value of the `Id` property in the lineage harvester configuration file.

   > Example The value of the `Id` property in the lineage harvester configuration file is `looker-source-1`. As a result, the name of your JSON file should be *looker-source-1.conf*.

   > Important Your JSON file must have the file extension *.conf*.

3. For each database in Looker, add the following content to the JSON file:

| Property | Description | Mandatory? |
|---|---|---|
| Connections | This section contains all Looker connections for which you want to create a technical lineage. | Yes |
| <connection name> | The name of a connection object in Looker. | Yes |
| dialect | The dialect of the supported data source in Looker. | No |
| schema | The name of the default schema of a supported data source in Looker.<br>If the lineage harvester fails to find a specific schema, it uses the default schema. | No |
| dbname | The name of the database of a supported data source in Looker. | No |
| collibraSystemName | The system or server name of a database. | Yes |

| Property | Description | Mandatory? |
| --- | --- | --- |
| filters | Optionally, use this section to specify the Looker folders from which you want to ingest metadata.<br><br>Let's say, for example, you filter on folder B. A Looker Folder asset is created in the specified domain in Collibra, and all of the metadata in folder B is ingested. If folder B has a parent folder A, then a Looker Folder asset is created (in the domain specified for folder B) to preserve the hierarchy, but no metadata from folder A is ingested.<br><br>You can specify more than one Looker Folder for ingestion into a single domain in Collibra.<br><br>Looker Data Sets are ingested in the default domain, regardless of any specified filtering.<br><br>Warning If you don't want to filter on Looker Folders, you must completely remove this filters section. | No |

<table>
<tr><th>Property</th><th>Description</th><th>Mandatory?</th></tr>
<tr><td></td><td>
Tip<br>You can use wildcards to capture multiple connection string combinations:
Show me the supported wildcards
<table>
<tr><th>Pattern</th><th>Description</th></tr>
<tr><td>*</td><td>Matches everything.</td></tr>
<tr><td>?</td><td>Matches any single character.</td></tr>
<tr><td>[seq]</td><td>Matches any character in "seq".</td></tr>
<tr><td>[!seq]</td><td>Matches any character not in "seq".</td></tr>
</table>
</td><td></td></tr>
<tr><td>domainId</td><td>
The unique resource ID of the domain (or domains), in Collibra, in which you want to ingest data objects from one or more Looker Folders.
Tip  You can find the domain ID by clicking the domain type. Then look in the URL of your browser to find the ID. The URL looks like https://<yourcollibrainstance>/domain/<domain ID>?<view>.
</td><td></td></tr>
<tr><td>description</td><td>Any description, as you see fit.</td><td></td></tr>
</table>

| Property | Description | Mandatory? |
|---|---|---|
| folderNames | The name (or names) of the Looker Folders from which you want to ingest.<br><br>Note You must specify either a folder name, a folder ID, or both. | |
| folderIds | The ID (or IDs) of the Looker Folder you want to ingest.<br><br>Note You must specify either a folder ID, a folder name, or both. | |

4. Save the <source ID> configuration file.

## Example of the <source ID>.conf file

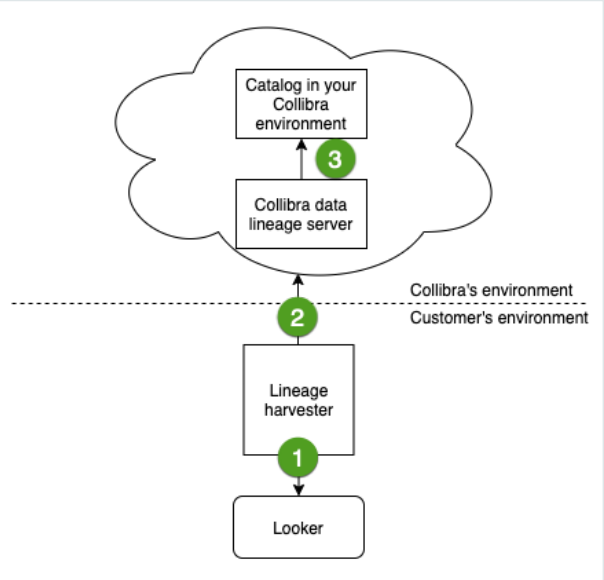
```
{
   "Connections": {
       "connection-object1": {
           "dialect": "mssql",
           "schema": "mssql-schema-name",
           "dbname": "mssql-database-name",
           "collibraSystemName": "mssql-system-name"
       },
       "connection-object2": {
           "dialect": "oracle",
           "schema": "oracle-schema-name",
           "dbname": "oracle-database-name",
           "collibraSystemName": "oracle-system-name"
       }
   }
   "filters":[
       {
       "domainId":"<reference ID>",
           "description":"any-description",
           "folderNames":["Folder1", "Folder2"]
       },
       {
       "domainId":"<reference ID>",
           "description":"any-description",
```

```
            "folderNames":["Folder3", "Folder4"]
        },
        {
        "domainId":"<reference ID>",
            "description":"any-description",
            "folderIds":["123xxxx", "456xxxx"]
        }
        ]
    }
}
```

# Looker ingestion workflow

You run the lineage harvester to start the Looker ingestion workflow. When you initiate Looker ingestion, each workflow component performs the following actions:

1. The lineage harvester:
    - Communicates with Looker.
    - Harvests the Looker metadata that will be ingested into Data Catalog.
    - Sends the Looker metadata to Collibra.
2. Collibra Data Intelligence Cloud:
    - Analyzes the Looker metadata.
    - Creates new assets and relations.
    - Imports new Looker assets and their relations in Data Catalog.
3. Data Catalog, via the Collibra Data Lineage server:
    - Shows new Looker assets.
    - Shows a technical lineage tab on Looker Look asset pages.



# Collibra Data Lineage service

The Collibra Data Lineage service processes and analyzes the harvested metadata and uploads it to Data Catalog. The Collibra Data Lineage service never processes actual data.

Based on your geographical location and cloud provider, the lineage harvester sends metadata to one of the following Collibra Data Lineage service instances:

- 15.222.200.199 (techlin-aws-ca.collibra.com)
- 18.198.89.106 (techlin-aws-eu.collibra.com)
- 54.242.194.190 (techlin-aws-us.collibra.com)
- 51.105.241.132 (techlin-azure-eu.collibra.com)
- 20.102.44.39 (techlin-azure-us.collibra.com)
- 35.197.182.41 (techlin-gcp-au.collibra.com)
- 34.152.20.240 (techlin-gcp-ca.collibra.com)
- 35.205.146.124 (techlin-gcp-eu.collibra.com)
- 34.87.122.60 (techlin-gcp-sg.collibra.com)
- 35.234.130.150 (techlin-gcp-uk.collibra.com)
- 34.73.33.120 (techlin-gcp-us.collibra.com)

> Important You have to whitelist all Collibra Data Lineage service instances in your geographic location. For example, if your data is located in Europe, you have to whitelist the following Collibra Data Lineage service instances: techlin-aws-eu and techlin-gcp-eu. In addition, we highly recommend that you always whitelist the techlin-aws-us instances as a backup, in case the lineage harvester cannot connect to other Collibra Data Lineage service instances.

# Schedule Looker ingestion jobs

You can use Task Scheduler on Windows or Crontab on Mac and Linux to make the lineage harvester run scheduled jobs. In a scheduled job, the lineage harvester uploads the Looker data source information to Collibra.

Collibra automatically creates new assets and relations of the type "Data Element targets / sources Data Element" at specific times, dates or intervals, using the information in your configuration file.

Warning When you run the lineage harvester, Collibra Data Lineage creates all Looker assets in the specified domain (or domains) in Collibra. We highly recommend that you do not move these assets to other domains. If you move assets to other domains, they will be deleted and recreated in the initial Data Catalog BI domain (or domains) when you synchronize Looker. As a consequence, all manually added data of those assets is lost.

Warning Relations that were manually created between Looker assets and other assets via a relation type in the Looker operating model, are deleted after a refresh of the Looker metadata.

Example You created a configuration file with connection information to your Looker environment. You schedule the lineage harvester job to run each Sunday at 23:00. As a result, your Looker metadata is automatically refreshed on a weekly basis.

# Looker business logic

Looker business users usually work with Looker dashboards and Looker looks to make business decisions. Collibra's Looker connector and lineage feature, offers business users several advantages:

- Easily find certified Looker content.
- Shop for Looker Looks.
- Find where content is stored in Looker.
- Get information about a Looker Look and other Looker report details in a single location.

Note Due to limitations of the Looker REST API, Data Catalog cannot stitch Looker assets and corresponding assets in Data Catalog. The Looker REST API does not provide transformations in Looker that are needed for stitching.

# Looker asset pages

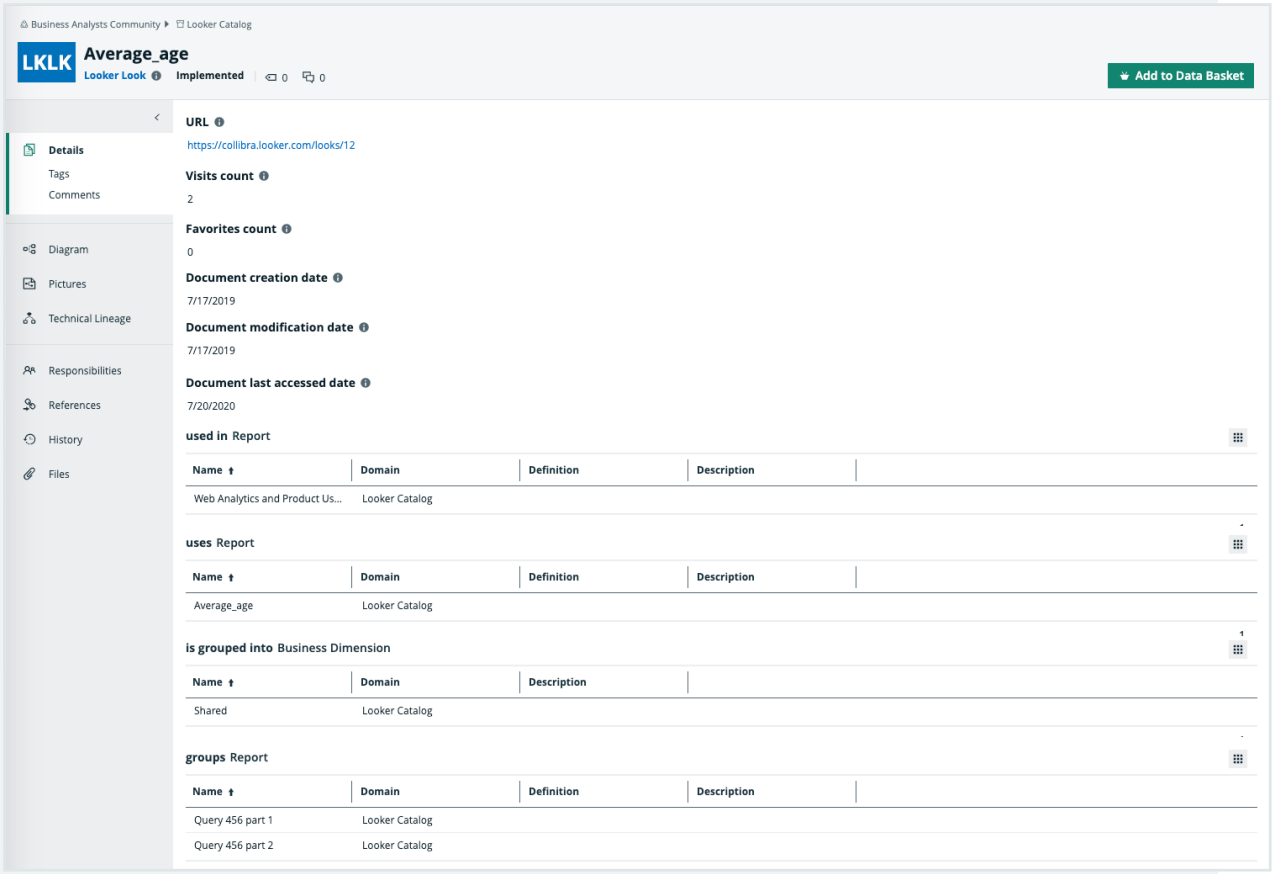
Depending on the Looker asset type, the asset page shows different information ingested from Looker. You can find a specific Looker asset page using Data Catalog search or via the Data Catalog BI domain in which you ingested the Looker metadata.

## Details

An asset page contains attributes and relations to other assets. This information is synchronized from Looker. However, you can add additional characteristics, tags or comments.

If you want to use a Looker Look, you can add it to the Data Basket and check it out.

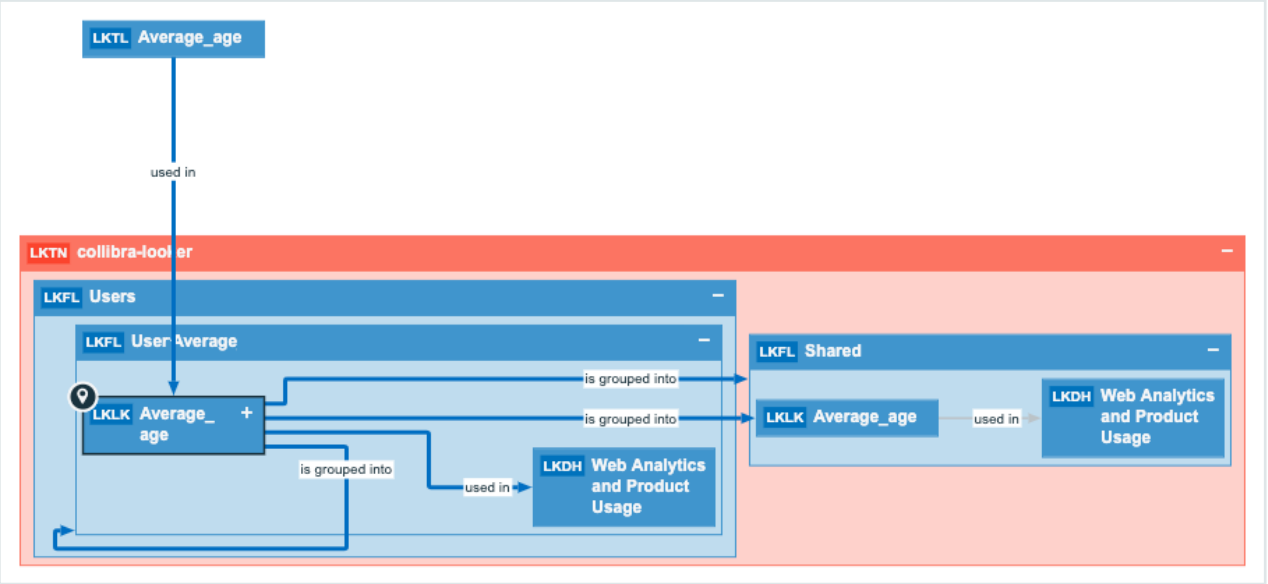
Example The following Looker Look asset shows in which Looker Folder it is stored, in which Looker Dashboard it is shown, which Looker Tiles it uses and which Looker Queries it groups. This asset has a number of attributes that give more information about the Looker Look.




# Business diagrams

The business diagram is a feature to show and interact with many assets and relations in an easy-to-read diagram. The business diagram helps you to quickly see to which other assets a specific asset is related. As such, the diagram can show a high-level presentation of a Looker Look. This enables you to see how the Looker Look relates to other Looker assets.

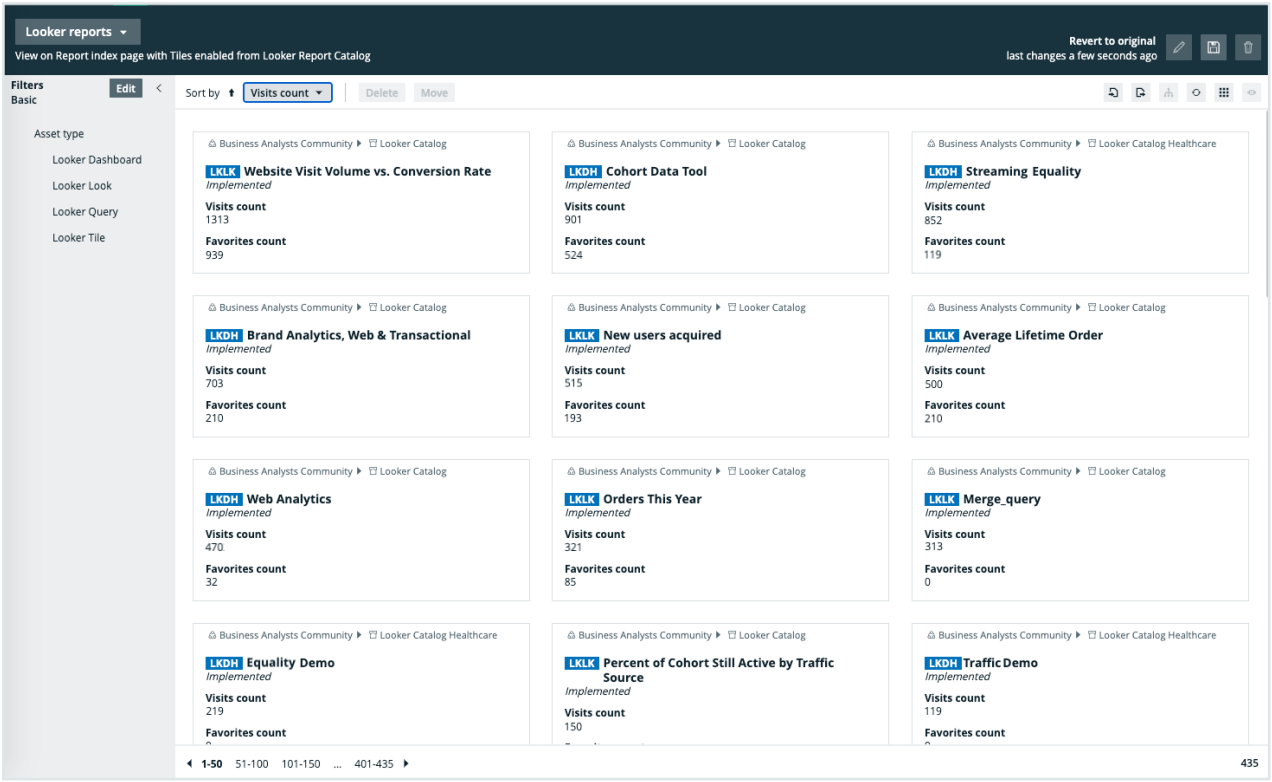
Example The following business diagram shows the *Average_age* Looker Look, which is stored in the *UserAverage* Looker Folder, but is also grouped into the *Shared* Looker Folder.




# Report views

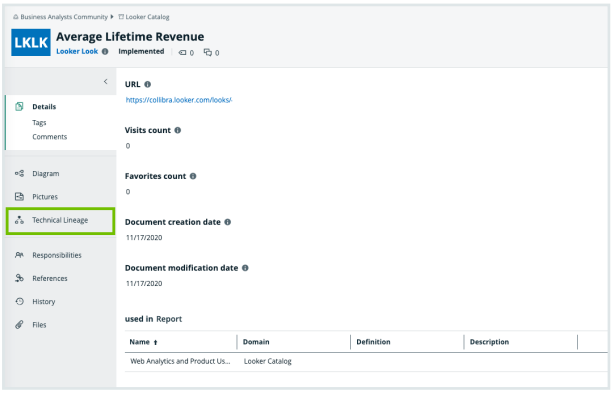
The Looker connector and lineage feature enables you to find all ingested Looker Look, Looker Dashboard, Looker Tile and Looker Query asset types in a single location.

In the **Reports** tab page in Data Catalog you can see an overview of all Report assets and their children. Optionally, you can create a view with a filter to only show Looker assets. This is useful if you quickly want to see all reports or if you want find specific reports for example certified reports or reports that are visited the most.




# Technical lineage for Looker

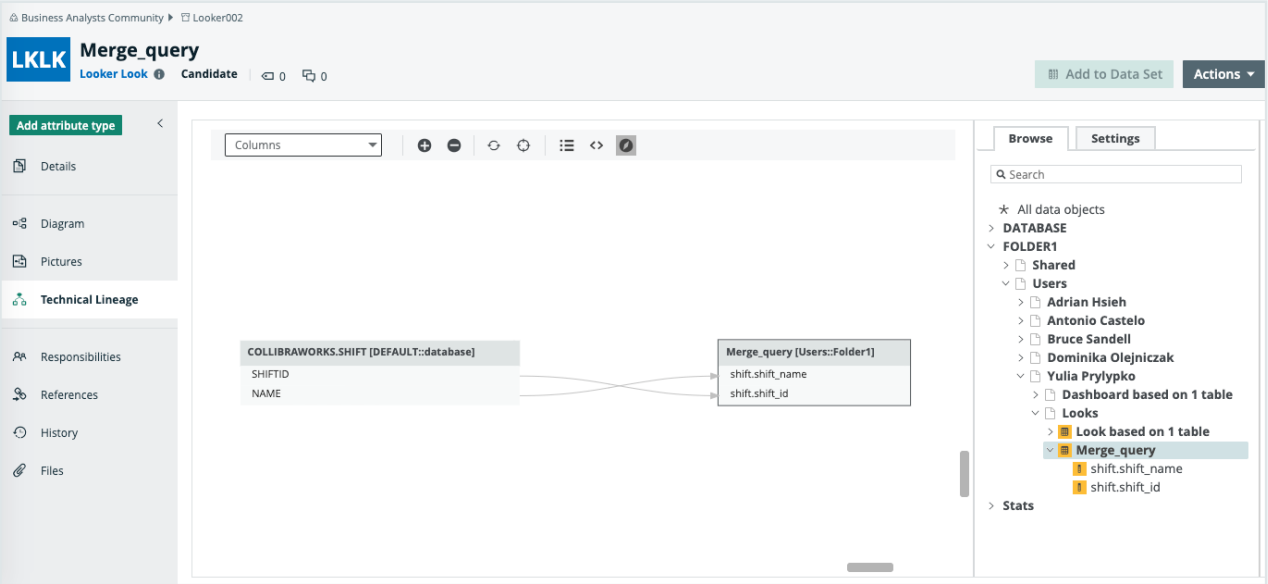
When you ingest Looker metadata, you automatically create a technical lineage for Looker Look assets. If you have the right permissions to view the technical lineage, you can go to a Looker Look asset page and click the Technical lineage tab, which allows you to access the technical lineage.



> Note Due to the limitations of the Looker REST API, we cannot stitch Looker assets and corresponding assets in Data Catalog. The Looker REST API does not provide transformations in Looker that are needed for stitching. As a result, the technical lineage only shows Looker metadata as it exists on the Collibra Data Lineage service and not as assets in Data Catalog.

# Example

The following technical lineage graph shows the technical lineage of Looker objects.




# Troubleshooting

If the connection between the lineage harvester and your Looker instance fails, you must try to sign in to Looker as an Admin user on the same machine that runs the lineage harvester. Open the interactive API documentation. If you are not able to open the API page, try one of the following:

- Check that you have network access to the API URL.
- Check that you have the correct credentials to sign in to the interactive API documentation. If necessary, create new API3 keys and try again. If you are now able to access the interactive API documentation, use the new Client ID and Client Secret in the configuration file.

- Sign in to the interactive API documentation with your API3 credentials and test the API calls. If your test is successful, compare API URL in the Request URL section to the `lookerUrl` value in the configuration file.

> Tip There are two ways to find the Looker API URL:
>
> - In the API Host URL field in the Looker Admin menu. If this field is empty, you can use the default Looker API URL which you can find in the interactive API documentation.
> - In the interactive API documentation URL. It is the part of the URL before `/api-docs/`.

# Working with MicroStrategy

MicroStrategy Intelligence Server is business intelligence software that connects to data sources to create and store layers of objects in the MicroStrategy metadata.

For more information on MicroStrategy, see the MicroStrategy documentation.

> Note
> - You can access any local or remote PostgreSQL database. The MicroStrategy Intelligence Server has an embedded PostgreSQL repository, as its default repository. For complete information on the default, embedded repository, see the MicroStrategy repository documentation.
> - Collibra Data Lineage automatically creates a technical lineage for MicroStrategy, but stitching is not available and the technical lineage does not show the relations to columns.



# MicroStrategy terminology

The following table shows the MicroStrategy terminology and how it maps to the Collibra Data Intelligence Cloud asset types.

| MicroStrategy term | Description | Asset type in Collibra |
|---|---|---|
| Attribute | A detailed view of a MicroStrategy visualization, with findings and insights. | MicroStrategy Report Attribute |

| MicroStrategy term | Description | Asset type in Collibra |
|---|---|---|
| Column | A column in a MicroStrategy data model. | MicroStrategy Column |
| Dataset | A collection of data that is used to create MicroStrategy reports. | MicroStrategy Data Model |
| Dossier | A collection of MicroStrategy chapters and pages. | MicroStrategy Dossier |
| Folder | A collection of MicroStrategy reports and data models. | MicroStrategy Folder |
| Project | A collection of MicroStrategy visualizations, report attributes and tables. | MicroStrategy Project |
| Report | A detailed view of a MicroStrategy data model, with visualizations of findings and insights. | MicroStrategy Report |
| Server | A visual analytics platform for creating and storing MicroStrategy reports and data models. | MicroStrategy Server |

# MicroStrategy asset and domain types

The MicroStrategy integration in Collibra Data Intelligence Cloud uses a specific subset of asset types and domain types.

The following table shows the asset and domain types that are used for the MicroStrategy integration. Above each asset type you can see the parent asset types in the breadcrumbs.

| Asset type | Description | Domain type |
| --- | --- | --- |
| Business Asset ▸ Business Dimension ▸ BI Folder ▸ MicroStrategy Folder | A collection of MicroStrategy reports and data models. | BI Catalog |
| Business Asset ▸ Business Dimension ▸ BI Folder ▸ MicroStrategy Project | A collection of MicroStrategy visualizations, report attributes and tables. | BI Catalog |
| Business Asset ▸ Report ▸ BI Report ▸ MicroStrategy Dossier | A collection of MicroStrategy chapters and pages. | BI Catalog |
| Business Asset ▸ Report ▸ BI Report ▸ MicroStrategy Report | A detailed view of a MicroStrategy data model, with visualizations of findings and insights. | BI Catalog |
| Data Asset ▸ Data Element ▸ Data Attribute ▸ BI Data Attribute ▸ MicroStrategy Column | A column in a MicroStrategy data model. | BI Catalog |

| Asset type | Description | Domain type |
| --- | --- | --- |
| Data Asset ▸ Data Element ▸ Report Attribute ▸ BI Report Attribute ▸ MicroStrategy Report Attribute | A detailed view of a MicroStrategy visualization, with findings and insights. | BI Catalog |
| Data Asset ▸ Data Structure ▸ Data Model ▸ BI Data Model ▸ MicroStrategy Data Model | A collection of data that is used to create MicroStrategy reports. | BI Catalog |
| Technology Asset ▸ Server ▸ BI Server ▸ MicroStrategy Server | A visual analytics platform for creating and storing MicroStrategy reports and data models. | BI Catalog |

# MicroStrategy operating model

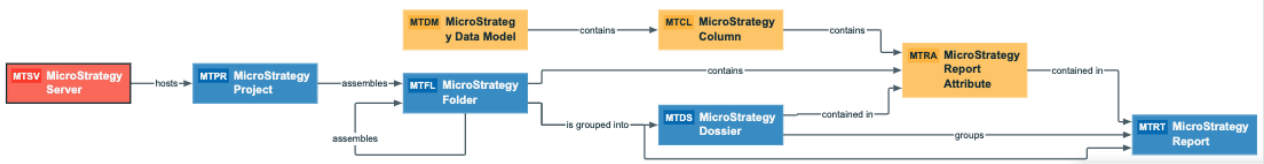
The harvester collects MicroStrategy metadata and sends it to the Collibra Data Lineage service. Collibra processes the metadata and creates new MicroStrategy assets and relations in Data Catalog. You can see them on the asset page overview or visualize them in a diagram or in a technical lineage.

Note

- The assets have the same names as their counterparts in MicroStrategy. You cannot edit Full names and Names in Data Catalog.
- Asset types are only created if you have all specific MicroStrategy and Data Catalog permissions.
- All MicroStrategy assets are created in the same domain.
- Relations that were manually created between MicroStrategy assets and other assets via a relation type in the MicroStrategy operating model, are deleted after synchronizing the MicroStrategy metadata.

# MicroStrategy metadata overview

The following image shows the relations between MicroStrategy asset types.




# Harvested metadata per asset type

This table shows the harvested MicroStrategy metadata for assets of each MicroStrategy asset type, assuming you have the necessary subscriptions and configurations for a full ingestion.

| Asset type | Harvested MicroStrategy metadata in Data Catalog | Resource ID |
|---|---|---|
| MicroStrategy Column<br><br>Resource ID: 00000000-0000-0000-0000-100000000047 | Description | 00000000-0000-0000-0000-000000003114 |
| | BI Data Model contains / is part of BI Data Attribute | 00000000-0000-0000-0000-000000007196 |
| | Report Attribute sourced from / is source of Data Attribute | 00000000-0000-0000-0000-120000000010 |
| MicroStrategy Data Model<br><br>Resource ID: 00000000-0000-0000-0000-100000000046 | Certified (not available yet) | 00000000-0000-0000-0001-000500000001 |
| | Description | 00000000-0000-0000-0000-000000003114 |
| | BI Data Model contains / is part of BI Data Attribute | 00000000-0000-0000-0000-000000007196 |
| MicroStrategy Dossier<br><br>Resource ID: 00000000-0000-0000-0000-100000000043 | Certified (not available yet) | 00000000-0000-0000-0001-000500000001 |
| | Description | 00000000-0000-0000-0000-000000003114 |
| | Business Dimension groups / is grouped into Report | 00000000-0000-0000-0000-120000000002 |
| | Report Attribute contained in / contains Report | 00000000-0000-0000-0000-000000007058 |
| | Report groups / is grouped into Report | 00000000-0000-0000-0000-120000000004 |

| Asset type | Harvested MicroStrategy metadata in Data Catalog | Resource ID |
|---|---|---|
| MicroStrategy Folder<br><br>Resource ID: 00000000-0000-0000-0000-100000000042 | Description | 00000000-0000-0000-0000-000000003114 |
| | BI Folder assembles / is assembled in BI Folder | 00000000-0000-0000-0000-120000000001 |
| | BI Folder contains / contained in Data Asset | 00000000-0000-0000-0000-120000000014 |
| | Business Dimension groups / is grouped into Report | 00000000-0000-0000-0000-120000000002 |
| MicroStrategy Project<br><br>Resource ID: 00000000-0000-0000-0000-100000000041 | Description | 00000000-0000-0000-0000-000000003114 |
| | Document creation date | 00000000-0000-0000-0000-000000000260 |
| | Document modification date | 00000000-0000-0000-0000-000000000261 |
| | BI Folder assembles / is assembled in BI Folder | 00000000-0000-0000-0000-120000000001 |
| | Server hosts / is hosted in Business Dimension | 00000000-0000-0000-0000-120000000000 |

| Asset type | Harvested MicroStrategy metadata in Data Catalog | Resource ID |
|---|---|---|
| MicroStrategy Report<br><br>Resource ID: 00000000-0000-0000-0000-100000000044 | Certified (not available yet) | 00000000-0000-0000-0001-000500000001 |
| | Description | 00000000-0000-0000-0000-000000003114 |
| | Business Dimension groups / is grouped into Report | 00000000-0000-0000-0000-120000000002 |
| | Report Attribute contained in / contains Report | 00000000-0000-0000-0000-000000007058 |
| | Report groups / is grouped into Report | 00000000-0000-0000-0000-120000000004 |
| MicroStrategy Report Attribute<br><br>Resource ID: 00000000-0000-0000-0000-100000000045 | Description | 00000000-0000-0000-0000-000000003114 |
| | Role in Report | 00000000-0000-0000-0000-000000000266 |
| | BI Folder contains / contained in Data Asset | 00000000-0000-0000-0000-120000000014 |
| | Report Attribute contained in / contains Report | 00000000-0000-0000-0000-000000007058 |
| | Report Attribute sourced from / is source of Data Attribute | 00000000-0000-0000-0000-120000000010 |

| Asset type | Harvested MicroStrategy metadata in Data Catalog | Resource ID |
|---|---|---|
| MicroStrategy Server<br><br>Resource ID: 00000000-0000-0000-0000-100000000040 | Description | 00000000-0000-0000-0000-000000003114 |
| | Server hosts / is hosted in Business Dimension | 00000000-0000-0000-0000-120000000000 |

# MicroStrategy integration steps

The MicroStrategy integration in Collibra Data Intelligence Cloud enables you to harvest MicroStrategy Intelligence Server metadata and create new MicroStrategy assets in Data Catalog. Collibra analyzes and processes the BI metadata and presents it as assets of specific types, retaining their original names.

> Important If you have a MicroStrategy on-premises environment, you can install the lineage harvester on a server that has access to the MicroStrategy server or on the MicroStrategy server itself. If you use MicroStrategy cloud, you have to install the lineage harvester where it can access the local or remote PostgreSQL repository. If you need to connect to the local PostgreSQL repository, you have to install the lineage harvester on the MicroStrategy server.

# Roles, privileges and permissions in MicroStrategy

To ingest MicroStrategy metadata in Data Catalog, the lineage harvester connects to the MicroStrategy Intelligence Server or a remote PostgreSQL database, depending on where you install the lineage harvester. You must have a role with user access to the relevant server and be able to access the metadata that is stored there.

# Steps

The table below shows the steps and prerequisites required to ingest MicroStrategy assets in Data Catalog.

> Important In the global assignment of each asset type included in the MicroStrategy operating model, ensure that none of the characteristics that are in the operating model have a maximum cardinality of “0”. If the maximum cardinality is set to “0” for any such characteristics, ingestion will fail.

| Step | What? | Description | Prerequisites |
|---|---|---|---|
| 1 | Create a new domain. | Before you can ingest MicroStrategy metadata, you have to create a new domain or choose an existing domain to store the new MicroStrategy assets. | • You have a resource role with the following resource permissions:<br>◦ Domain: Add |

| Step | What? | Description | Prerequisites |
|---|---|---|---|
| 2 | Download and install the lineage harvester. | You use the lineage harvester to collect metadata from MicroStrategy and upload it to the Collibra Data Lineage service where the metadata is scanned, processed and analyzed.<br><br>You can download the lineage harvester from the Downloads section of the Collibra Product Resource Center.<br><br>Where you choose to install the lineage harvester depends on the database that the lineage harvester will access to harvest the metadata. You can install it either:<br>• On the MicroStrategy server, to access the local PostgreSQL database.<br>• Close to your data source, to access a remote PostgreSQL database. | • You have access to the lineage harvester. We highly recommend that you always install and use the newest lineage harvester.<br>• Your environment meets the system requirements to install and use the lineage harvester.<br>• You have added firewall rules so that the lineage harvester can connect to the Collibra Data Lineage service instances with the following IP addresses:<br>  • 15.222.200.1-99 (techlin-aws-ca.-collibra.com)<br>  • 18.198.89.10-6 (techlin-aws-eu.-collibra.com) |

| Step | What? | Description | Prerequisites |
|---|---|---|---|
| | | | • 54.242.194.1-90 (techlin-aws-us.-collibra.com)<br>• 51.105.241.1-32 (techlin-azure-eu.-collibra.com)<br>• 20.102.44.39 (techlin-azure-us.-collibra.com)<br>• 35.197.182.4-1 (techlin-gcp-au.-collibra.com)<br>• 34.152.20.24-0 (techlin-gcp-ca.-collibra.com)<br>• 35.205.146.1-24 (techlin-gcp-eu.-collibra.com)<br>• 34.87.122.60 (techlin-gcp-sg.-collibra.com)<br>• 35.234.130.1-50 (techlin-gcp-uk.-collibra.com)<br>• 34.73.33.120 (techlin-gcp- |

| Step | What? | Description | Prerequisites |
| --- | --- | --- | --- |
| | | | us.-collibra.com) |

| Step | What? | Description | Prerequisites |
|---|---|---|---|
| 3 | Prepare the lineage harvester configuration file and run the lineage harvester. | You create a configuration file to provide the connection information that you need to connect your MicroStrategy server and remote data source to the Collibra Data Lineage service and to the Collibra Data Intelligence Cloud domain in which you want to ingest the MicroStrategy assets.<br><br>You can access an empty configuration file in the lineage harvester installation folder. When you have created and saved the configuration file, you can run the lineage harvester to upload the MicroStrategy metadata to Collibra.<br><br>`{ "general": { "catalog": { "url": "https://<organization>.collibra.com", "userName": "<your-collibra-username>" }, "useSharedDbModel": true, "useCollibraSystemName": false }, "sources": { "type": "Microstrategy", "id": "microstrategy-batch", "collibraSystemName": "system-name", "domainId": "<domain-resource-id>", "username": "mstr", "hostname": "remote.-postgres.com", "port": 5432, "databaseName": "poc_metadata", "deleteRawMetadataAfter-Processing": true } }` | • You have Collibra Data Intelligence Cloud 2021.07 or newer.<br>• You have a dedicated domain to ingest the MicroStrategy assets.<br>• You have a global role with the Catalog global permission, for example Catalog Author.<br>• You have a global role with the Technical lineage global permission.<br>• You have a global role with the Data Stewardship Manager global permission.<br>• You have a resource role with the following resource permissions:<br>◦ Asset: Add |

| Step | What? | Description | Prerequisites |
|---|---|---|---|
| | | | ◦ Attribute: Add<br>◦ Attachment: Add<br>• Your environment meets the system requirements to run the lineage harvester. |
| 4 | View the MicroStrategy ingestion results. | After the MicroStrategy metadata is ingested in Data Catalog, you can go to the domain where you ingested MicroStrategy and see the list of ingested MicroStrategy assets.<br>Warning When you run the lineage harvester, Collibra Data Lineage creates all MicroStrategy assets in the same Data Catalog BI domain. We highly recommend that you do not move these assets to another domain. If you move assets to another domain, they will be deleted and recreated in the initial Data Catalog BI domain when you synchronize MicroStrategy. As a consequence, all manually added data of those assets is lost. | • You have Collibra Data Intelligence Cloud 2021.07 or newer.<br>• Catalog Experience is enabled in Collibra Console.<br>• You have a global role with the Catalog global permission, for example Catalog Author. |

# The lineage harvester setup for MicroStrategy

The lineage harvester is a software application that is needed to collect your MicroStrategy metadata and send it to the Collibra Data Lineage service, where the metadata is processed and new MicroStrategy assets and relations are created. Collibra Data Intelligence Cloud then import those assets and relations into Data Catalog.

> Note We highly recommend that you always install and use the newest lineage harvester. You can download the harvester via the Downloads section of the Collibra Product Resource Center.

## Lineage harvester system requirements

You need to meet the system requirements to be able to install and run the lineage harvester.

### Software requirements

Java Runtime Environment version 11 or newer, or OpenJDK 11 or newer.

For Java Runtime Environment 16 or newer, or OpenJDK 16 or newer, set the `JAVA_OPTS` environment variable for the lineage harvester to function properly:

```
JAVA_OPTS='--illegal-access=deny'
```

> Note To ingest Snowflake data sources, the minimum requirement is Java Runtime Environment version 16 or newer, or OpenJDK 16 or newer.

## Hardware requirements

You need to meet the hardware requirements to install and run the lineage harvester.

### Minimum hardware requirements

You need the following minimum hardware requirements:

- 2 GB RAM
- 1 GB free disk space

### Recommended hardware requirements

The minimum requirements are most likely insufficient for production environments. We recommend the following hardware requirements:

- 4 GB RAM

  > Tip 4 GB RAM is sufficient in most cases, but more memory could be needed for larger harvesting tasks. For instructions on how to increase the maximum heap size, see Technical lineage general troubleshooting.

- 20 GB free disk space

## Network requirements

The lineage harvester uses the HTTPS protocol by default and uses port 443.

You need the following minimum network requirements:

# Collibra Data Lineage service instances

The Collibra Data Lineage service processes and analyzes the harvested metadata from supported (meta)data sources and uploads it to Data Catalog. The Collibra Data Lineage service never processes or stores actual data, only metadata.

When you run the lineage harvester, it first connects to any available Collibra Data Lineage service instance to determine your cloud provider and geographic location of your

Collibra Data Intelligence Cloud environment. Then, the lineage harvester sends the harvested metadata to the Collibra Data Lineage service instance with the same cloud provider and geographic location.

Currently, your metadata can be processed on one of the following Collibra Data Lineage service instances:

| Server | IP address | DNS name |
|---|---|---|
| techlin-aws-ca | 15.222.200.199 | techlin-aws-ca.collibra.com |
| techlin-aws-eu | 18.198.89.106 | techlin-aws-eu.collibra.com |
| techlin-aws-us | 54.242.194.190 | techlin-aws-us.collibra.com |
| techlin-azure-eu | 51.105.241.132 | techlin-azure-eu.collibra.com |
| techlin-azure-us | 20.102.44.39 | techlin-azure-us.collibra.com |
| techlin-gcp-au | 35.197.182.41 | techlin-gcp-au.collibra.com |
| techlin-gcp-ca | 34.152.20.240 | techlin-gcp-ca.collibra.com |
| techlin-gcp-eu | 35.205.146.124 | techlin-gcp-eu.collibra.com |
| techlin-gcp-sg | 34.87.122.60 | techlin-gcp-sg.collibra.com |
| techlin-gcp-uk | 35.234.130.150 | techlin-gcp-uk.collibra.com |
| techlin-gcp-us | 34.73.33.120 | techlin-gcp-us.collibra.com |

Important You have to whitelist all Collibra Data Lineage service instances in your geographic location. For example, if your data is located in Europe, you have to whitelist the following Collibra Data Lineage service instances: techlin-aws-eu and techlin-gcp-eu. In addition, we highly recommend that you always whitelist the techlin-aws-us instances as a backup, in case the lineage harvester cannot connect to other Collibra Data Lineage service instances.

# Prepare a domain for MicroStrategy ingestion

You can create a new domain for your MicroStrategy assets and use the domain ID in the lineage harvester configuration file. As a result, Collibra uses this domain to ingest all MicroStrategy assets during the MicroStrategy integration process.

## Prerequisites

You have a resource role with the Domain > Add resource permission.

## Steps

1. In the main menu, click the **Create** (+) button.
   » The **Create** dialog box appears.
2. Click the **Organization** tab.
3. Click a domain type from the list.
   If you clicked the wrong domain type here, you can change it in the **Type** field in the next screen.
   » The **Create Domain** dialog box appears.
4. Enter the required information.

| Field | Description |
|---|---|
| Type | The domain type of the domain you are creating. In this case, you need to select *BI Catalog*. |
| Community | The community under which the domain will be located. |
| Name | The name of the new domain. |

5. Click **Create**.
6. Open your domain.

7. Copy the domain ID.

   > Tip If you go to your domain, you can find the domain ID in the URL. The URL looks like: https://<yourcollibrainstance>/domain/**22258f64-40b6-4b16-9c08-c95f8ec0da26**?view=00000000-0000-0000-0000-000000040001. In this example, the domain ID is in bold.

8. Paste the domain ID in the lineage harvester configuration file.

> Warning When you run the lineage harvester, Collibra Data Lineage creates all MicroStrategy assets in the same Data Catalog BI domain. We highly recommend that you do not move these assets to another domain. If you move assets to another domain, they will be deleted and recreated in the initial Data Catalog BI domain when you synchronize MicroStrategy. As a consequence, all manually added data of those assets is lost.

# Install the lineage harvester

Before you can use the lineage harvester, you have to download it and install it. You can download the lineage harvester from the Collibra Community downloads page.

> Note We highly recommend that you always install and use the newest lineage harvester.

Where you choose to install the lineage harvester depends on the database that the lineage harvester will access to harvest the metadata. You can install it either:

- On the MicroStrategy server, to access the local PostgreSQL database.
- Close to your data source, to access a remote PostgreSQL database.

> Important If you have a MicroStrategy on-premises environment, you can install the lineage harvester on a server that has access to the MicroStrategy server or on the MicroStrategy server itself. If you use MicroStrategy cloud, you have to install the lineage harvester where it can access the local or remote PostgreSQL repository. If you need to connect to the local PostgreSQL repository, you have to install the lineage harvester on the MicroStrategy server.

# Install the lineage harvester on the MicroStrategy server

For local database access, only PostgreSQL databases are supported. The MicroStrategy Intelligence Server has an embedded PostgreSQL repository, as its default repository. For complete information, see the MicroStrategy repository documentation.

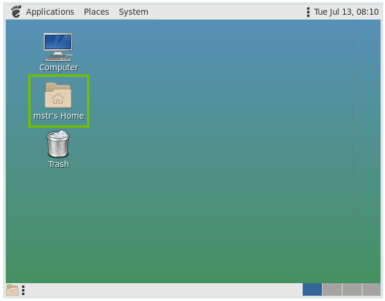
## Prerequisites

- You have Collibra Data Intelligence Cloud 2021.07 or newer.
- You have MicroStrategy 2021 or newer.
- You meet the minimum lineage harvester system requirements.
- Java Runtime Environment version 11 or newer or OpenJDK 11 or newer.
- You have added Firewall rules so that the lineage harvester can connect to:
  - The host names of all databases in the lineage harvester configuration file.
  - All Collibra Data Lineage service instances within your geographical location:
    - 15.222.200.199 (techlin-aws-ca.collibra.com)
    - 18.198.89.106 (techlin-aws-eu.collibra.com)
    - 54.242.194.190 (techlin-aws-us.collibra.com)
    - 51.105.241.132 (techlin-azure-eu.collibra.com)
    - 20.102.44.39 (techlin-azure-us.collibra.com)
    - 35.197.182.41 (techlin-gcp-au.collibra.com)
    - 34.152.20.240 (techlin-gcp-ca.collibra.com)
    - 35.205.146.124 (techlin-gcp-eu.collibra.com)
    - 34.87.122.60 (techlin-gcp-sg.collibra.com)
    - 35.234.130.150 (techlin-gcp-uk.collibra.com)
    - 34.73.33.120 (techlin-gcp-us.collibra.com)

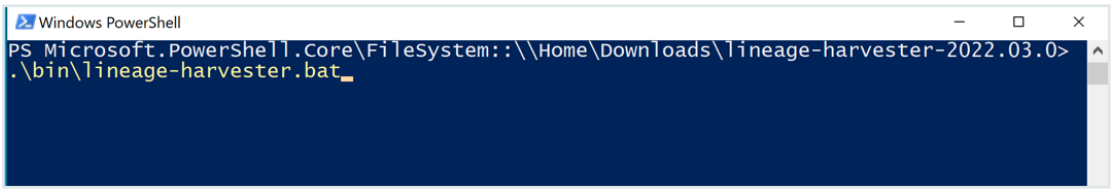
> Note The lineage harvester connects to different instances based on your geographic location and cloud provider. If your location or cloud provider changes, the lineage harvester rescans all your data sources. You have to whitelist all Collibra Data Lineage service instances in your geographic location. In addition, we highly recommend that you always whitelist the techlin-aws-us instance as a backup, in case the lineage harvester cannot connect to other Collibra Data Lineage service instances.

## Steps

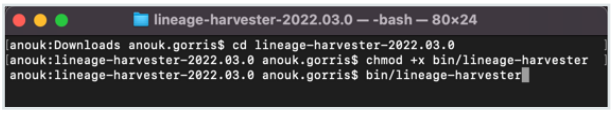
1. Download the lineage harvester.
2. Sign in to the MicroStrategy web portal.
3. Click **Remote Desktop Gateway**.
4. Sign in to **Apache Guacamole**.
5. Click **Platform Instance VNC**.
6. Copy the lineage harvester ZIP file to the Platform Instance VNC home directory.



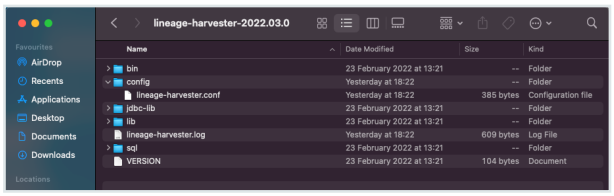

7. Unzip the archive.
   » You can now access the lineage harvester folder.
8. Run the following command line to start the lineage harvester:
   - Windows: `.\bin\lineage-harvester.bat`




   - For other operating systems: `chmod +x bin/lineage-harvester` and then `bin/lineage-harvester`




   » An empty configuration file is created in the config folder.




   » The lineage harvester is installed automatically. You can check the installation by running `./bin/lineage-harvester --help`.

# Install the lineage harvester close to your data source

To access a remote PostgreSQL database, install the lineage harvester close to the data source.

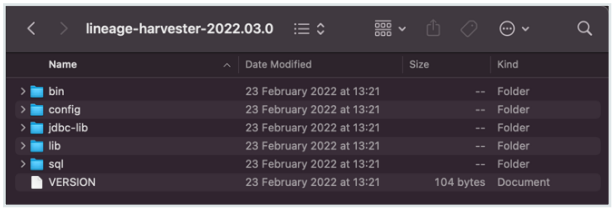
## Prerequisites

- You have Collibra Data Intelligence Cloud 2021.07 or newer.
- You have MicroStrategy 2021 or newer.
- You meet the minimum lineage harvester system requirements.
- Java Runtime Environment version 11 or newer or OpenJDK 11 or newer.
- You have added Firewall rules so that the lineage harvester can connect to:
  - The host names of all databases in the lineage harvester configuration file.
  - All Collibra Data Lineage service instances within your geographical location:
    - 15.222.200.199 (techlin-aws-ca.collibra.com)
    - 18.198.89.106 (techlin-aws-eu.collibra.com)
    - 54.242.194.190 (techlin-aws-us.collibra.com)
    - 51.105.241.132 (techlin-azure-eu.collibra.com)
    - 20.102.44.39 (techlin-azure-us.collibra.com)
    - 35.197.182.41 (techlin-gcp-au.collibra.com)
    - 34.152.20.240 (techlin-gcp-ca.collibra.com)
    - 35.205.146.124 (techlin-gcp-eu.collibra.com)
    - 34.87.122.60 (techlin-gcp-sg.collibra.com)
    - 35.234.130.150 (techlin-gcp-uk.collibra.com)
    - 34.73.33.120 (techlin-gcp-us.collibra.com)

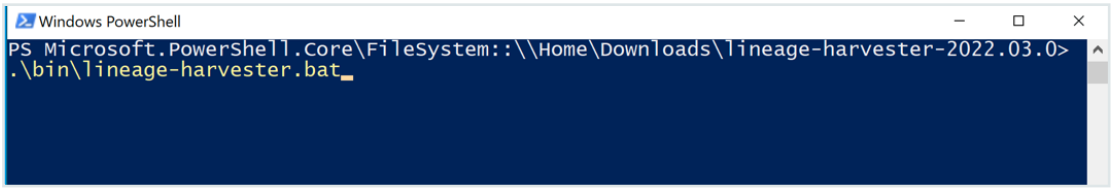
> Note The lineage harvester connects to different instances based on your geographic location and cloud provider. If your location or cloud provider changes, the lineage harvester rescans all your data sources. You have to whitelist all Collibra Data Lineage service instances in your geographic location. In addition, we highly recommend that you always whitelist the techlin-aws-us instance as a backup, in case the lineage harvester cannot connect to other Collibra Data Lineage service instances.

## Steps

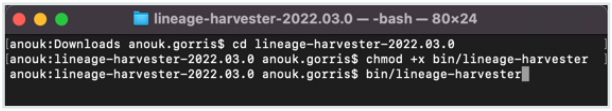
1. Download the newest lineage harvester.
2. Unzip the archive.
   » You can now access the lineage harvester folder.



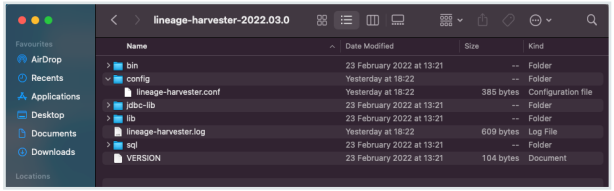

3. Run the following command line to start the lineage harvester:
   - Windows: `.\bin\lineage-harvester.bat`




   - For other operating systems: `chmod +x bin/lineage-harvester` and then `bin/lineage-harvester`




   » An empty configuration file is created in the config folder.




   » The lineage harvester is installed automatically. You can check the installation by running `./bin/lineage-harvester --help`.

# Prepare the lineage harvester configuration file for MicroStrategy

You have to prepare a configuration file before you run the lineage harvester. The lineage harvester collects your MicroStrategy metadata and sends it to the Collibra Data Lineage

service, where it is processed and analyzed. Collibra Data Intelligence Cloud then imports the MicroStrategy assets and relations to Data Catalog.

## Prerequisites

Ensure that you have completed the following tasks:

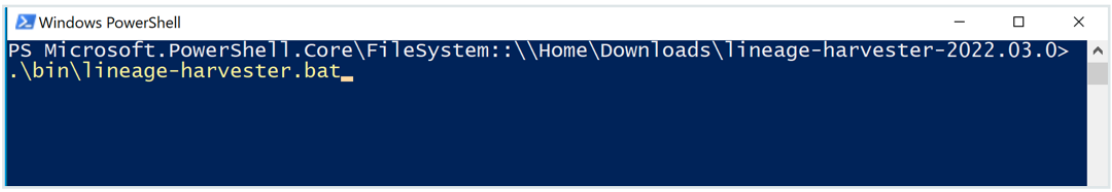
- Installed and set up the latest lineage harvester.
- Created a BI Catalog domain in which you want to ingest the MicroStrategy assets.

Ensure that you meet the following requirements and have the following permissions:

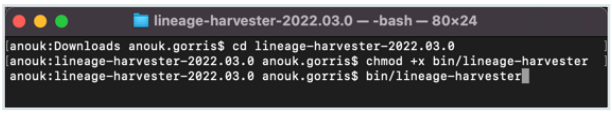
- Use Collibra Data Intelligence Cloud.
- A global role with the following global permissions:
  - Catalog, for example Catalog Author
  - Data Stewardship Manager
  - Manage all resources
  - System administration
  - Technical lineage
- A resource role with the following resource permission on the community level in which you created the BI Catalog domain:
  - Asset: add
  - Attribute: add
  - Domain: add
  - Attachment: add

## Steps

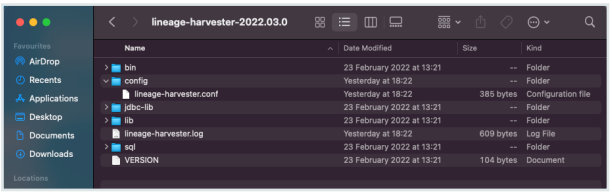
1. Run the following command line to start the lineage harvester:
   - Windows: `.\bin\lineage-harvester.bat`



- For other operating systems: `chmod +x bin/lineage-harvester` and then `bin/lineage-harvester`




» An empty configuration file is created in the config folder.




2. Open the lineage-harvester.conf file and enter the values for each property.

| Properties | Description |
|---|---|
| general | This section describes the connection information between the lineage harvester and Data Catalog. |
| catalog | This section contains information that is necessary to connect to Data Catalog. |
| url | The URL of your Collibra Data Intelligence Cloud environment.<br><br>Note You can only enter the public URL of your Collibra DGC environment. Other URLs will not be accepted. |
| username | The username that you use to sign in to Collibra. |

| Properties | Description |
| --- | --- |
| useCollibraSystemName | By default, the `useCollibraSystemName` property is set to `false`. This property is not valid for MicroStrategy integration. We recommend that you leave this property set to `false`. |

| Properties | Description |
| --- | --- |
| useSharedDbModel | Optional property to enable the sharing of metadata batches from multiple SQL data sources. Set this property to `true`, to help avoid potential analysis errors on the Collibra Data Lineage service.<br><br>To use this property, you need lineage harvester 2022.07 or newer.<br><br>If you set this property to `true`, you have to run the lineage harvester twice. Read the following details about the issue and solution.<br><br>**See details about the issue and solution**<br>Normally, when you run the lineage harvester to harvest metadata from two or more data sources, the metadata from each source is processed independently. This means that the metadata from one data source cannot access the metadata of another.<br><br>Let's say, for example, you specify the following two SQL data sources in your lineage harvester configuration file:<br>◦ A database source that retrieves the database model.<br>◦ An SqlDirectory source with Data Manipulation Language (DML) |

| Properties | Description |
| --- | --- |
| | statements that reference data in the database source.<br><br>Because these data sources are processed independently, there is a good chance that the DML statements will fail during analysis. Any wildcards in the DML statements, for example, would fail because the SqlDirectory source can't access the referenced database source.<br><br>**The solution**<br><br>The shared database model allows for computed results from a "main" batch. Although multiple data sources are still processed independently, the metadata from each data source is merged into a main batch. Then, before analyzing the next batch, a check is done to see if a preceding main batch exists. If one does, the analyzer retrieves the database model and the DML statements successfully pass analysis.<br><br>This means, however, that you have to run the lineage harvester twice. On the first run, the harvested metadata is merged in a main batch. Then, when you run the lineage harvester again, using the full-sync command, the subsequent batches are able to |

| Properties | Description |
| --- | --- |
| | successfully reference the metadata in the main batch.<br><br>In a future version of Collibra, this property will be enabled by default and you won't need to run the lineage harvester twice. |
| sources | This section contains all MicroStrategy connection properties. |
| type | The kind of data source. In this case, the value has to be `MicroStrategy`. |
| collibraSystemName | This property is deprecated for MicroStrategy integration. The lineage harvester does not take into account any value that you enter here. |

| Properties | Description |
| --- | --- |
| id | The unique ID of your MicroStrategy metadata. For example, `my_microstrategy`.<br><br>**Warning** In the `sources` section of your lineage harvester configuration file, you can only specify one `id` property per MicroStrategy Intelligence Server. If you have multiple `id` properties for a single MicroStrategy Intelligence Server, ingestion will fail. If you have multiple `id` properties in the configuration file, it means you intend to ingest from multiple unique MicroStrategy Intelligence Servers.<br><br>**Tip** This value can be anything as long as it is unique and human readable. The ID identifies the batch of MicroStrategy metadata on the Collibra Data Lineage service. |
| domainId | The unique reference ID of the domain in Collibra Data Intelligence Cloud in which you want to ingest the MicroStrategy assets. |

| Properties | Description |
| --- | --- |
| username | The username that you use to sign in to MicroStrategy. |
| hostname | The endpoint that you use to access the PostgreSQL repository or remote data source, depending on where you installed the lineage harvester.<br>For example `remote.postgres.com`. |
| port | The port number. |
| databaseName | Optionally, the name of your database. For example `poc_metadata`. |
| deleteRawMetadataAfterProcessing | The lineage harvester harvests metadata from specified data sources and uploads it in a ZIP file to a Collibra Data Lineage service instance, for processing.<br>You can use this optional property to specify whether or not the metadata should be deleted after it has been processed.<br>If the property is set to `true`, the metadata is deleted after processing. If set to `false`, the metadata is stored in an Amazon S3 bucket. |

3. Save the configuration file.

4. Start the lineage harvester again in the console and run the following command:
   - for Windows: `.\bin\lineage-harvester.bat full-sync`
   - for other operating systems: `./bin/lineage-harvester full-sync`
5. When prompted, enter the password or client secret to connect to your Collibra Data Intelligence Cloud and MicroStrategy environment.
   » The passwords are encrypted and stored in **/config/pwd.conf**

## Example

The following example shows a configuration file for MicroStrategy.

```
{
 "general": {
   "catalog": {
     "url": "https://<organization>.collibra.com",
     "userName": "<your-collibra-username>"
   },
   "useSharedDbModel": true,
   "useCollibraSystemName": false
 },
  "sources": {
    "type": "Microstrategy",
    "id": "microstrategy-batch",
    "collibraSystemName": "system-name",
    "domainId": "<domain-resource-id>",
    "username": "mstr",
    "hostname": "remote.postgres.com",
    "port": 5432,
    "databaseName": "poc_metadata",
    "deleteRawMetadataAfterProcessing": true
  }
}
```

# Schedule MicroStrategy ingestion jobs

You can use Task Scheduler on Windows or Crontab on Mac and Linux to make the lineage harvester run scheduled jobs. In a scheduled job, the lineage harvester uploads the MicroStrategy data source information to Collibra.

Collibra automatically creates new assets and relations between the MicroStrategy assets at specific times, dates or intervals, using the information in your configuration file.

Example You created a configuration file with connection information to your MicroStrategy environment. You schedule the lineage harvester job to run each Sunday at 23:00. As a result, your MicroStrategy metadata is automatically refreshed on a weekly basis.

Warning When you run the lineage harvester, Collibra Data Lineage creates all MicroStrategy assets in the same Data Catalog BI domain. We highly recommend that you do not move these assets to another domain. If you move assets to another domain, they will be deleted and recreated in the initial Data Catalog BI domain when you synchronize MicroStrategy. As a consequence, all manually added data of those assets is lost.

Warning Relations that were manually created between MicroStrategy assets and other assets via a relation type in the MicroStrategy operating model, are deleted after a refresh of the MicroStrategy metadata.