Data warehousing documentation in Microsoft Fabric

Learn more about the two Data Warehousing experiences in Microsoft Fabric

Data warehousing

OVERVIEW

Overview

Get started with Data Warehouse

TUTORIAL

End to End tutorials

Data warehouse tutorial

Create a Warehouse quickstart

Get started with SQL Endpoint

GET STARTED

What is a Lakehouse?

Better together - the lakehouse and warehouse

Create a lakehouse with OneLake

Understand default Power BI datasets

Load data into the Lakehouse

How to copy data using Copy activity in Data pipeline

How to move data into Lakehouse via Copy assistant

Security

HOW-TO GUIDE

Data warehousing security

Connectivity

Workspace roles

SQL granular permissions

Item permissions

Ingest data

HOW-TO GUIDE

Ingest data guide

Ingest data using pipelines

Ingest data using TSQL

Ingest data using Copy

Design and Develop

CONCEPT

Datasets

Model data in the default Power BI dataset

Define relationships in data models

Reports in the Power BI service

Query

CONCEPT

Query using the SQL Query editor

Query with T-SQL

Query using visual query editor

View data in the Data preview

Manage and Monitor

HOW-TO GUIDE

Monitor

Workload Management

Statistics

Troubleshoot

Best practices

ARCHITECTURE

Security

Ingest Data

What is data warehousing in Microsoft Fabric?

Article • 05/23/2023

Applies to: SQL Endpoint and Warehouse in Microsoft Fabric

Microsoft Fabric provides customers with a unified product that addresses every aspect of their data estate by offering a complete, SaaS-ified Data, Analytics and AI platform, which is lake centric and open. The foundation of Microsoft Fabric enables the novice user through to the seasoned professional to leverage Database, Analytics, Messaging, Data Integration and Business Intelligence workloads through a rich, easy to use, shared SaaS experience with Microsoft OneLake as the centerpiece.

> **Important**
>
> Microsoft Fabric is currently in PREVIEW. This information relates to a prerelease product that may be substantially modified before it's released. Microsoft makes no warranties, expressed or implied, with respect to the information provided here.

A lake-centric SaaS experience built for any skill level

Microsoft Fabric introduces a lake centric data warehouse built on an enterprise grade distributed processing engine that enables industry leading performance at scale while eliminating the need for configuration and management. Through an easy to use SaaS experience that is tightly integrated with Power BI for easy analysis and reporting, Warehouse in Microsoft Fabric converges the world of data lakes and warehouses with a goal of greatly simplifying an organizations investment in their analytics estate. Data warehousing workloads benefit from the rich capabilities of the SQL engine over an open data format, enabling customers to focus on data preparation, analysis and reporting over a single copy of their data stored in their Microsoft OneLake.

The Warehouse is built for any skill level - from the citizen developer through to the professional developer, DBA or data engineer. The rich set of experiences built into Microsoft Fabric workspace enables customers to reduce their time to insights by having an easily consumable, always connected dataset that is integrated with Power BI in DirectLake mode. This enables second-to-none industry leading performance that ensures a customer's report always has the most recent data for analysis and reporting. Cross database querying can be leveraged to quickly and seamlessly leverage multiple data sources that span multiple databases for fast insights and zero data duplication.

Virtual warehouses with cross database querying

Microsoft Fabric provides customers with the ability to stand up virtual warehouses containing data from virtually any source by using shortcuts. Customers can build a virtual warehouse by creating shortcuts to their data wherever it resides. A virtual warehouse may consist of data from OneLake, Azure Data Lake Storage, or any other cloud vendor storage within a single boundary and with no data duplication.

Seamlessly unlock value from a variety of data sources through the richness of cross database querying in Microsoft Fabric. Cross database querying enables customers to quickly and seamlessly leverage multiple data sources for fast insights and with zero data duplication. Data stored in different sources can be easily joined together enabling customers to deliver rich insights that previously required significant effort from data integration and engineering teams.

Cross-database queries can be created through the Visual Query editor, which offers a no-code path to insights over multiple tables. The SQL Query editor, or other familiar tools such as SQL Server Management Studio (SSMS), can also be used to create cross-database queries.

Autonomous workload management

Warehouses in Microsoft Fabric leverage an industry-leading distributed query processing engine, which provides customers with workloads that have a natural isolation boundary. There are no knobs to turn with the autonomous allocation and relinquishing of resources to offer best in breed performance with automatic scale and concurrency built in. True isolation is achieved by separating workloads with different workload characteristics, ensuring that ETL jobs never interfere with their ad hoc analytics and reporting workloads.

Open format for seamless engine interoperability

Data in the Warehouse is stored in the parquet file format and published as Delta Lake Logs, enabling ACID transactions and cross engine interoperability that can be leveraged through other Microsoft Fabric workloads such as Spark, Pipelines, Power BI and Azure Data Explorer. Customers no longer need to create multiple copies of their data to enable data professionals with different skill sets. Data engineers that are accustomed to working in Python can easily leverage the same data that was modeled and served by a data warehouse professional that is accustomed to working in SQL. In parallel, BI professionals can quickly and easily leverage the same data to create a rich set of visualizations in Power BI with record performance and no data duplication.

Separation of storage and compute

Compute and storage are decoupled in a Warehouse which enables customers to scale near instantaneously to meet the demands of their business. This enables multiple compute engines to read from any supported storage source with robust security and full ACID transactional guarantees.

Easily ingest, load and transform at scale

Data can be ingested into the Warehouse through Pipelines, Dataflows, cross database querying or the COPY INTO command. Once ingested, data can be analyzed by multiple business groups through functionality such as cross database querying. Time to insights is expedited through a fully integrated BI experience through graphical data modeling easy to use web experience for querying within the Warehouse Editor.

What types of warehouses are available in Microsoft Fabric?

This section provides an overview of two distinct data warehousing experiences: the SQL Endpoint of the Lakehouse and the Warehouse.

SQL Endpoint of the Lakehouse

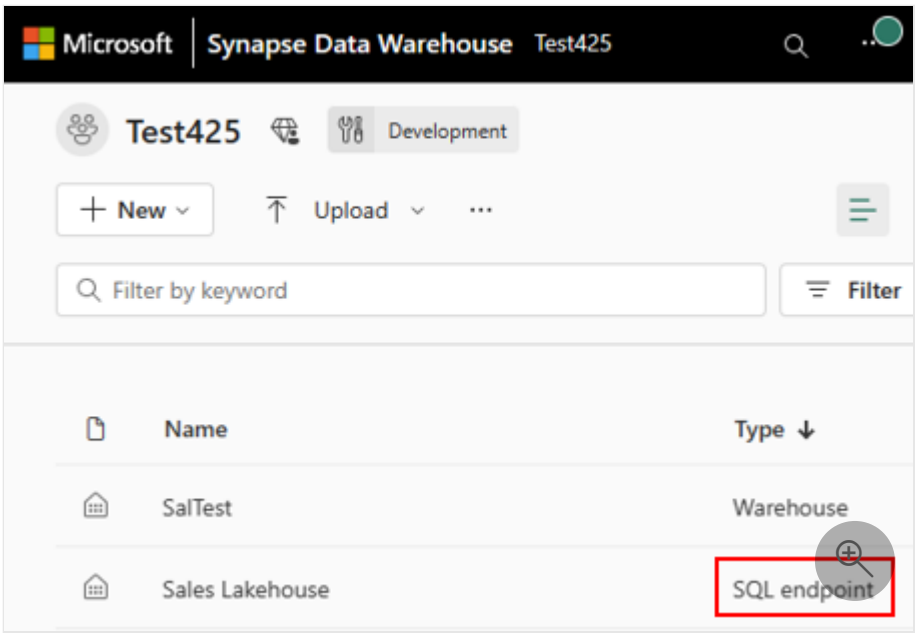
A SQL Endpoint is a warehouse that is automatically generated from a Lakehouse in Microsoft Fabric. A customer can transition from the "Lake" view of the Lakehouse (which supports data engineering and Apache Spark) to the "SQL" view of the same Lakehouse. The SQL Endpoint is read-only, and data can only be modified through the "Lake" view of the Lakehouse using Spark.

Via the SQL Endpoint of the Lakehouse, the user has a subset of SQL commands that can define and query data objects but not manipulate the data. You can perform the following actions in the SQL Endpoint:

- Query the tables that reference data in your Delta Lake folders in the lake.
- Create views, inline TVFs, and procedures to encapsulate your semantics and business logic in T-SQL.
- Manage permissions on the objects.

In a Microsoft Fabric workspace, a SQL Endpoint is labeled "SQL Endpoint" under the **Type** column. Each Lakehouse has an autogenerated SQL Endpoint that can be

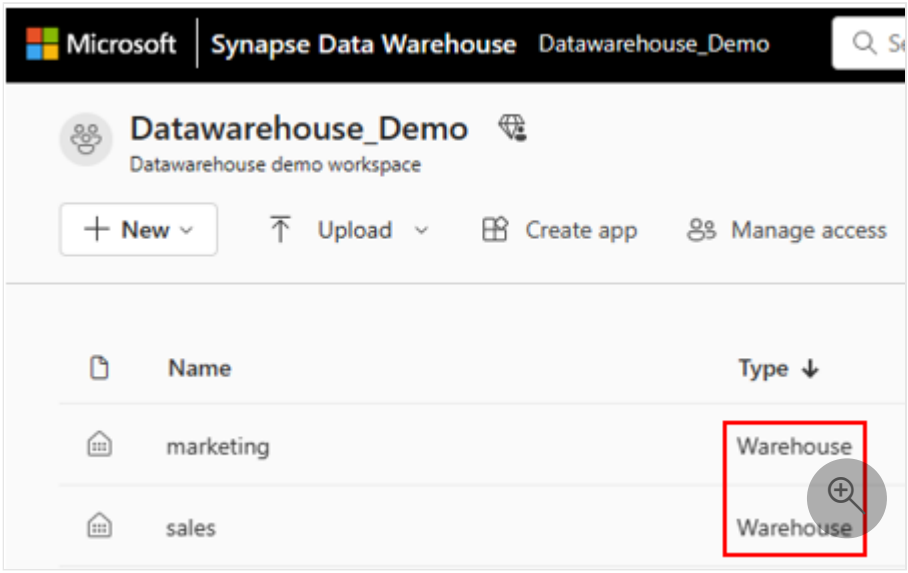
leveraged through familiar SQL tools such as SQL Server Management Studio, Azure Data Studio, the Microsoft Fabric SQL Query Editor.



To get started with the SQL Endpoint, see Better together: the lakehouse and warehouse in Microsoft Fabric.

Synapse Data Warehouse

In a Microsoft Fabric workspace, a Synapse Data Warehouse or **Warehouse** is labeled as 'Warehouse' under the **Type** column. A Warehouse supports transactions, DDL, and DML queries.



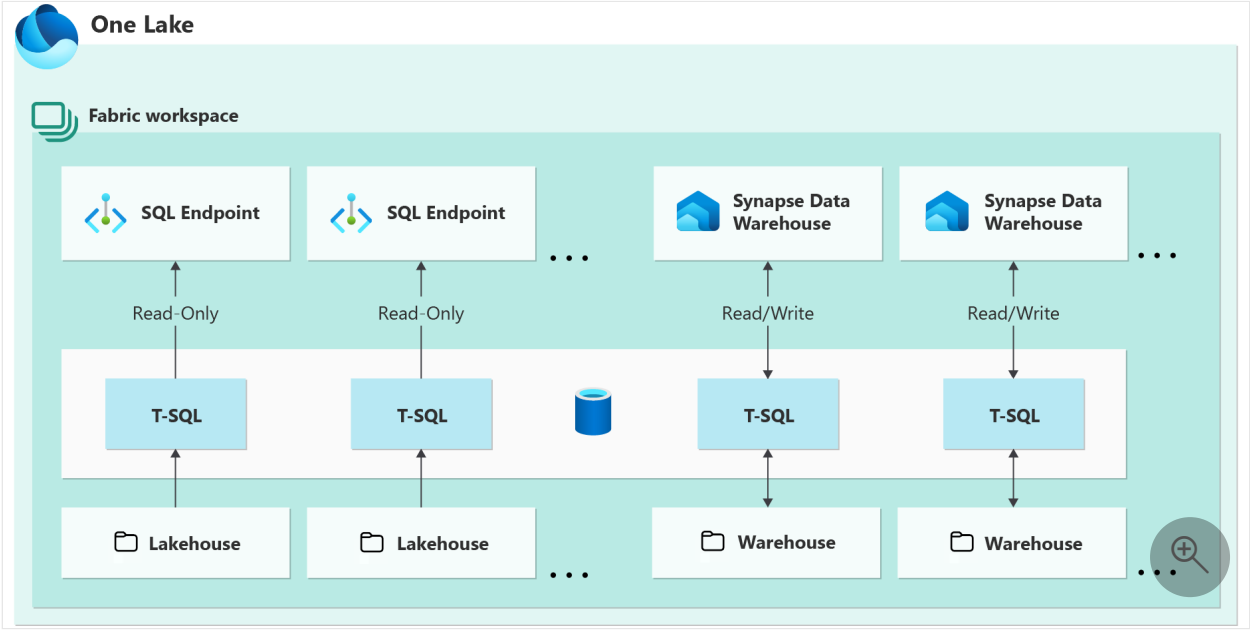
Unlike a SQL Endpoint which only supports read only queries and creation of views and TVFs, a Warehouse has full transactional DDL and DML support and is created by a customer. A Warehouse is populated by one of the supported data ingestion methods

such as COPY INTO, Pipelines, Dataflows, or cross database ingestion options such as CREATE TABLE AS SELECT (CTAS), INSERT..SELECT, or SELECT INTO.

To get started with the Warehouse, see Create a warehouse in Microsoft Fabric.

Compare the Warehouse and the SQL Endpoint of the Lakehouse

This section describes the differences between the Warehouse and SQL Endpoint in Microsoft Fabric.



The **SQL Endpoint** is a *read-only* warehouse that is automatically generated upon creation from a Lakehouse in Microsoft Fabric. Delta tables that are created through Spark in a Lakehouse are automatically discoverable in the SQL Endpoint as tables. The SQL Endpoint enables data engineers to build a relational layer on top of physical data in the Lakehouse and expose it to analysis and reporting tools using the SQL connection string. Data analysts can then use T-SQL to access Lakehouse data using the warehouse experience. Use SQL Endpoint to design your warehouse for BI needs and serving data.

The **Synapse Data Warehouse** or **Warehouse** is a 'traditional' data warehouse and supports the full transactional T-SQL capabilities like an enterprise data warehouse. As opposed to SQL Endpoint, where tables and data are automatically created, you are fully in control of creating tables, loading, transforming, and querying your data in the data warehouse using either the Microsoft Fabric portal or T-SQL commands.

For more information about querying your data in Microsoft Fabric, see Query the SQL Endpoint or Warehouse in Microsoft Fabric.

Compare different warehousing capabilities

In order to best serve your analytics use cases, there are a variety of capabilities available to you. Generally, the warehouse can be thought of as a superset of all other capabilities, providing a synergistic relationship between all other analytics offerings that provide T-SQL.

Within fabric, there are users who may need to decide between a Warehouse, Lakehouse, and even a Power BI datamart.

Microsoft Fabric offering	Warehouse	SQL Endpoint of the Lakehouse	Power BI datamart
Licensing	Fabric or Power BI Premium	Fabric or Power BI Premium	Power BI Premium only
Primary capabilities	ACID compliant, full data warehousing with transactions support in T-SQL.	Read only, system generated SQL Endpoint for Lakehouse for T-SQL querying and serving. Supports queries and views on top of Lakehouse delta tables only.	No-code data warehousing and T-SQL querying
Developer profile	SQL Developers or citizen developers	Data Engineers or SQL Developers	Citizen developer only
Recommended use case	• Data Warehousing for enterprise use		

- Data Warehousing supporting departmental, business unit or self service use
- Structured data analysis in T-SQL with tables, views, procedures and functions and Advanced SQL support for BI
- Exploring and querying delta tables from the lakehouse
- Staging Data and Archival Zone for analysis
- Medallion architecture with zones for bronze, silver and gold analysis
- Pairing with Warehouse for enterprise analytics use cases
- Small departmental or business unit warehousing use cases
- Self service data warehousing use cases
- Landing zone for Power BI dataflows and simple SQL support for BI

Development experience

- Warehouse Editor with full support for T-SQL data ingestion, modeling, development, and querying UI experiences for data ingestion, modeling, and querying
- Read / Write support for 1st and 3rd party tooling
- Lakehouse SQL Endpoint with limited T-SQL support for views, table valued functions, and SQL Queries
- UI experiences for modeling and querying
- Limited T-SQL support for 1st and 3rd party tooling
- Datamart Editor with UI experiences and queries support
- UI experiences for data ingestion, modeling, and querying
- Read-only support for 1st and 3rd party tooling

T-SQL capabilities

Full DQL, DML, and DDL T-SQL support, full transaction support

Full DQL, No DML, limited DDL T-SQL Support such as SQL Views and TVFs

Full DQL only

Data loading

SQL, pipelines, dataflows

Spark, pipelines, dataflows, shortcuts

Dataflows only

Delta table support

Reads and writes Delta tables

Reads delta tables

NA

Storage layer

Open Data Format - Delta

Open Data Format - Delta

NA

Automatically generated schema in the SQL Endpoint of the Lakehouse

The SQL Endpoint manages the automatically generated tables so the workspace users can't modify them. Users can enrich the database model by adding their own SQL schemas, views, procedures, and other database objects.

For every Delta table in your Lakehouse, the SQL Endpoint automatically generates one table.

Tables in the SQL Endpoint are created with a delay. Once you create or update Delta Lake folder/table in the lake, the warehouse table that references the lake data won't be immediately created/refreshed. The changes will be applied in the warehouse after 5-10 seconds.

For autogenerated schema data types for the SQL Endpoint, see Data types in Microsoft Fabric.

Next steps

- Better together: the lakehouse and warehouse in Microsoft Fabric
- Create a warehouse
- Create a lakehouse in Microsoft Fabric
- Introduction to Power BI datamarts
- Creating reports

Microsoft Fabric decision guide: data warehouse or lakehouse

Article • 05/23/2023

Use this reference guide and the example scenarios to help you choose between the data warehouse or a lakehouse for your workloads using Microsoft Fabric.

> **ⓘ Important**
>
> Microsoft Fabric is currently in PREVIEW. This information relates to a prerelease product that may be substantially modified before it's released. Microsoft makes no warranties, expressed or implied, with respect to the information provided here.

Data warehouse and lakehouse properties

	Data warehouse	Lakehouse	Power BI Datamart
Data volume	Unlimited	Unlimited	Up to 100 GB
Type of data	Structured	Unstructured, semi-structured, structured	Structured
Primary developer persona	Data warehouse developer, SQL engineer	Data engineer, data scientist	Citizen developer
Primary developer skill set	SQL	Spark (Scala, PySpark, Spark SQL, R)	No code, SQL
Data organized by	Databases, schemas, and tables	Folders and files, databases and tables	Database, tables, queries
Read operations	Spark, T-SQL	Spark, T-SQL	Spark, T-SQL, Power BI
Write operations	T-SQL	Spark (Scala, PySpark, Spark SQL, R)	Dataflows, T-SQL

	Data warehouse	Lakehouse	Power BI Datamart
Multi-table transactions	Yes	No	No
Primary development interface	SQL scripts	Spark notebooks, Spark job definitions	Power BI
Security	Object level (table, view, function, stored procedure, etc.), column level, row level, DDL/DML	Row level, table level (when using T-SQL), none for Spark	Built-in RLS editor
Access data via shortcuts	Yes (indirectly through the lakehouse)	Yes	No
Can be a source for shortcuts	Yes (tables)	Yes (files and tables)	No
Query across items	Yes, query across lakehouse and warehouse tables	Yes, query across lakehouse and warehouse tables; query across lakehouses (including shortcuts using Spark)	No

Scenarios

Review these scenarios for help with choosing between using a lakehouse or a data warehouse in Fabric.

Scenario 1

Susan, a professional developer, is new to Microsoft Fabric. They are ready to get started cleaning, modeling, and analyzing data but need to decide to build a data warehouse or a lakehouse. After review of the details in the previous table, the primary decision points are the available skill set and the need for multi-table transactions.

Susan has spent many years building data warehouses on relational database engines, and is familiar with SQL syntax and functionality. Thinking about the larger team, the primary consumers of this data are also skilled with SQL and SQL analytical tools. Susan decides to use a **data warehouse**, which allows the team to interact primarily with T-SQL, while also allowing any Spark users in the organization to access the data.

Scenario 2

Rob, a data engineer, needs to store and model several terabytes of data in Fabric. The team has a mix of PySpark and T-SQL skills. Most of the team running T-SQL queries are consumers, and therefore don't need to write INSERT, UPDATE, or DELETE statements. The remaining developers are comfortable working in notebooks, and because the data is stored in Delta, they're able to interact with a similar SQL syntax.

Rob decides to use a **lakehouse**, which allows the data engineering team to use their diverse skills against the data, while allowing the team members who are highly skilled in T-SQL to consume the data.

Scenario 3

Ash, a citizen developer, is a Power BI developer. They're familiar with Excel, Power BI, and Office. They need to build a data product for a business unit. They know they don't quite have the skills to build a data warehouse or a lakehouse, and those seem like too much for their needs and data volumes. They review the details in the previous table and see that the primary decision points are their own skills and their need for a self service, no code capability, and data volume under 100 GB.

Ash works with business analysts familiar with Power BI and Microsoft Office, and knows that they already have a Premium capacity subscription. As they think about their larger team, they realize the primary consumers of this data may be analysts, familiar with no-code and SQL analytical tools. Ash decides to use a **Power BI datamart**, which allows the team to interact build the capability fast, using a no-code experience. Queries can be executed via Power BI and T-SQL, while also allowing any Spark users in the organization to access the data as well.

Next steps

- What is data warehousing in Microsoft Fabric?
- Create a warehouse in Microsoft Fabric
- Create a lakehouse in Microsoft Fabric
- Introduction to Power BI datamarts

Create a Warehouse in Microsoft Fabric

Article • 05/23/2023

Applies to: Warehouse in Microsoft Fabric

This article describes how to get started with Warehouse in Microsoft Fabric using the Microsoft Fabric portal, including discovering creation and consumption of the warehouse. You learn how to create your warehouse from scratch and sample along with other helpful information to get you acquainted and proficient with warehouse capabilities offered through the Microsoft Fabric portal.

> **Note**
>
> It is important to note that much of the functionality described in this section is also available to users via a TDS end-point connection and tools such as **SQL Server Management Studio (SSMS)** or **Azure Data Studio (ADS)** (for users who prefer to use T-SQL for the majority of their data processing needs). For more information, see **Connectivity** or **Query a warehouse**.

> **Important**
>
> Microsoft Fabric is currently in PREVIEW. This information relates to a prerelease product that may be substantially modified before it's released. Microsoft makes no warranties, expressed or implied, with respect to the information provided here.

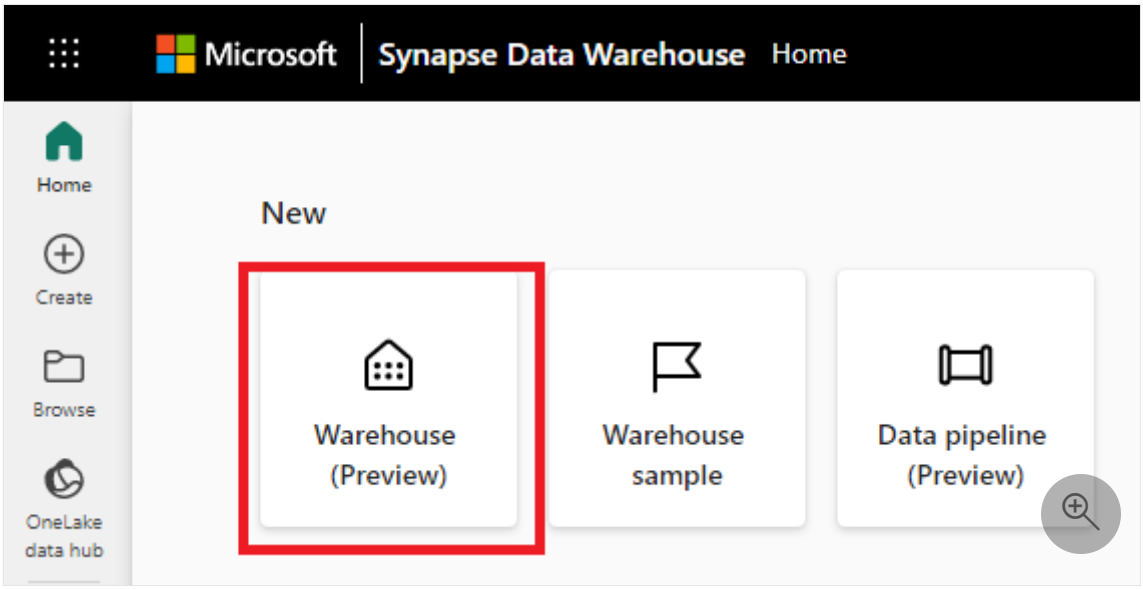
> **Tip**
>
> You can proceed with either a blank Warehouse or a sample Warehouse to continue this series of Get Started steps.

How to create a warehouse

In this section, we walk you through three distinct experiences available for creating a Warehouse from scratch in the Microsoft Fabric portal.

Create a warehouse using the Home hub

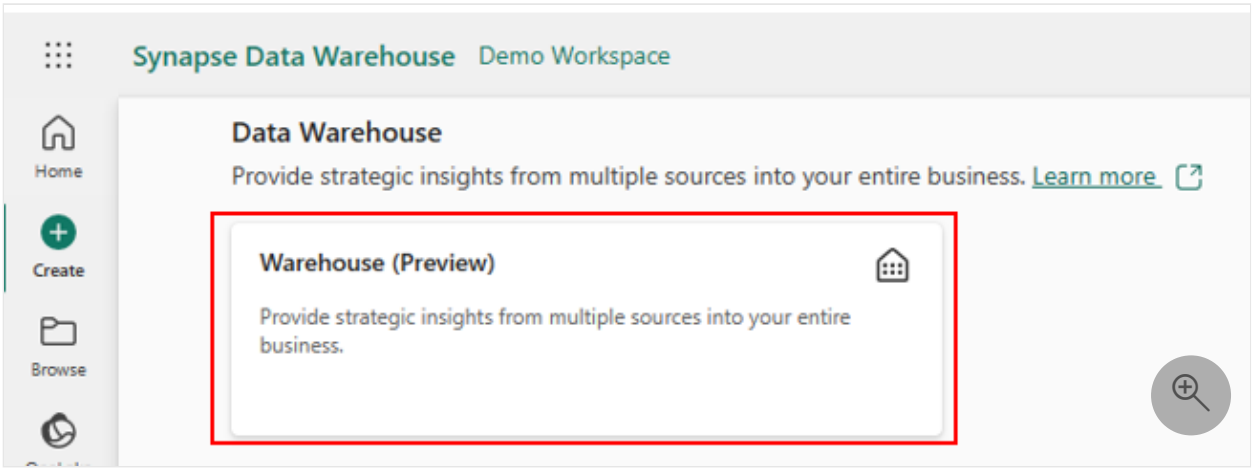
The first hub in the left navigation menus is the **Home** hub. You can start creating your warehouse from the **Home** hub by selecting the **Warehouse** card under the **New** section. An empty warehouse is created for you to start creating objects in the warehouse. You can use either a sample data set to get a jump start or load your own test data if you prefer.



Create a warehouse using the Create hub

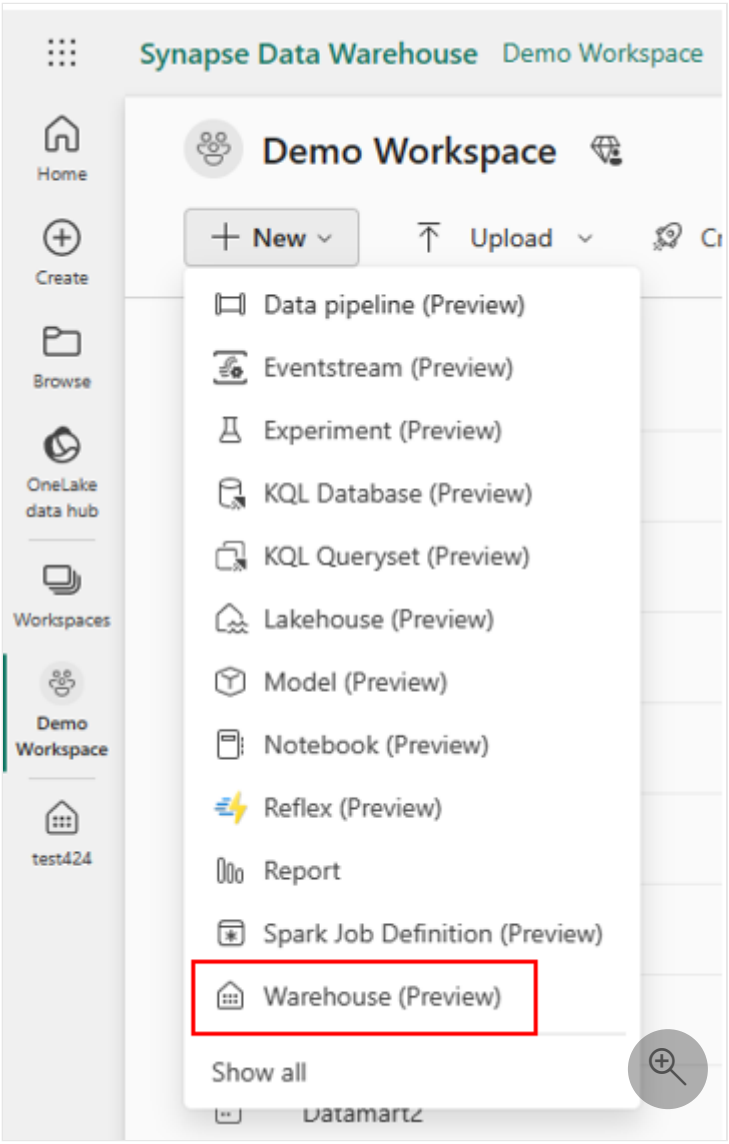
Another option available to create your warehouse is through the Create hub, which is the second hub in the left navigation menu.

You can create your warehouse from the Create hub by selecting the **Warehouse** card under the **Data Warehousing** section. When you select the card, an empty warehouse is created for you to start creating objects in the warehouse or use a sample to get started as previously mentioned.

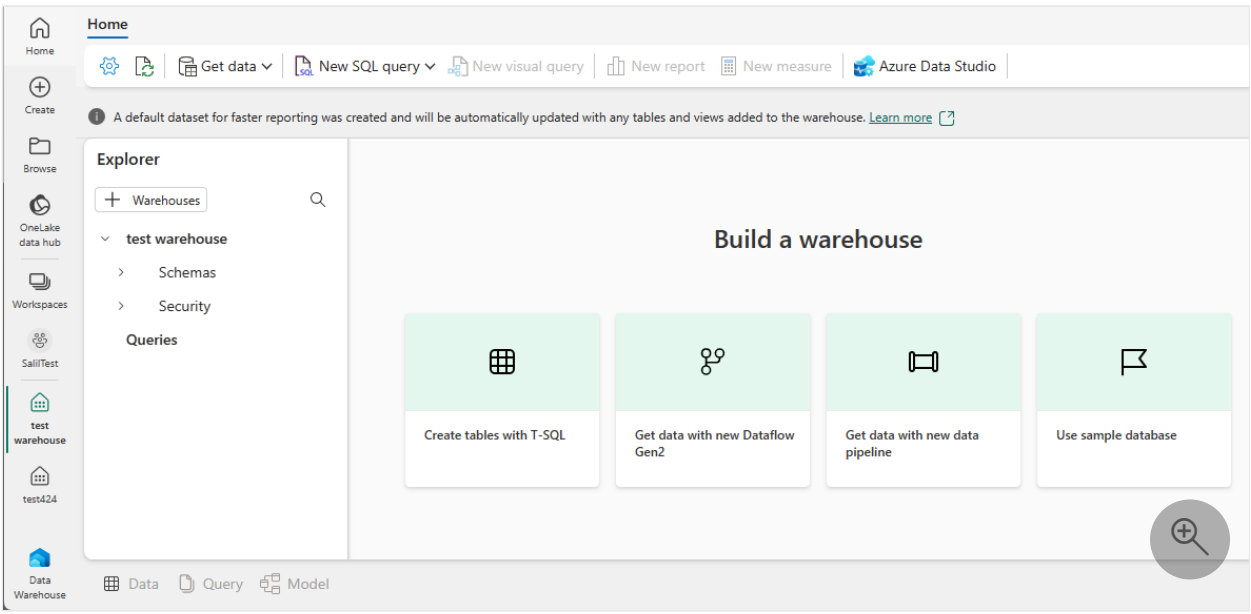


Create a warehouse from the workspace list view

To create a warehouse, navigate to your workspace, select **+ New** and then select **Warehouse** to create a warehouse.



Once initialized, you can load data into your warehouse. For more information about getting data into a warehouse, see Ingesting data.

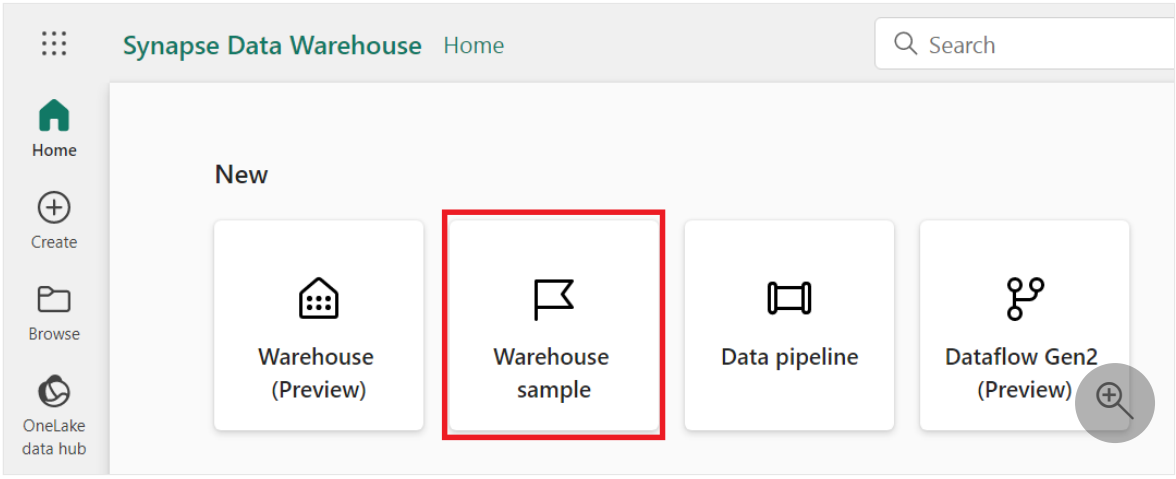


How to create a warehouse sample

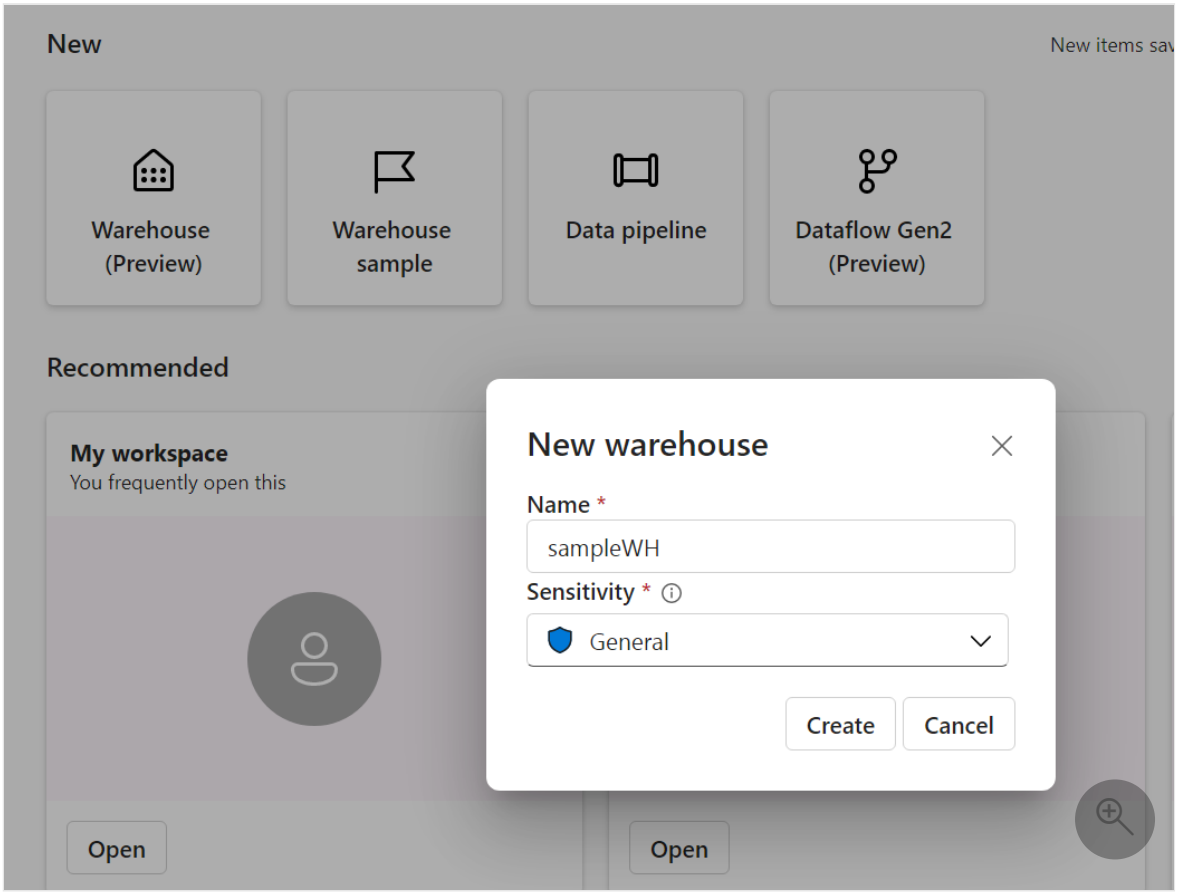
In this section, we walk you through two distinct experiences available for creating a sample Warehouse from scratch.

Create a warehouse sample using the Home hub

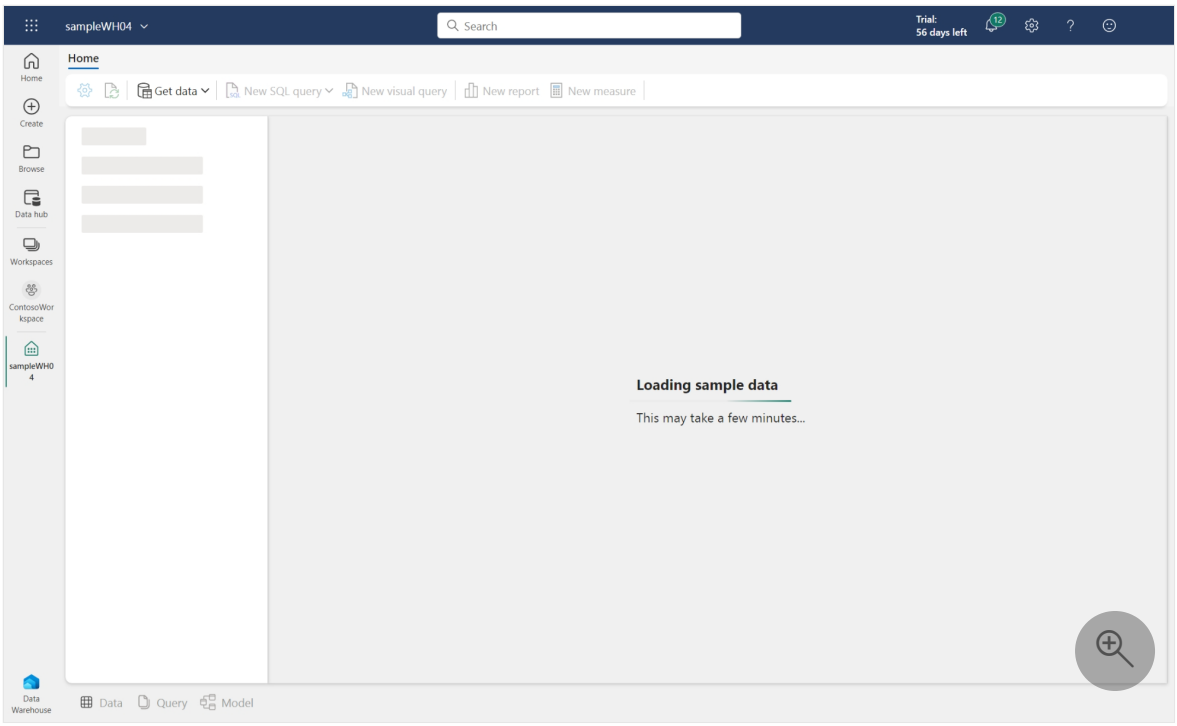
1. The first hub in the left navigation menus is the **Home** hub. You can start creating your warehouse sample from the **Home** hub by selecting the **Warehouse sample** card under the **New** section.



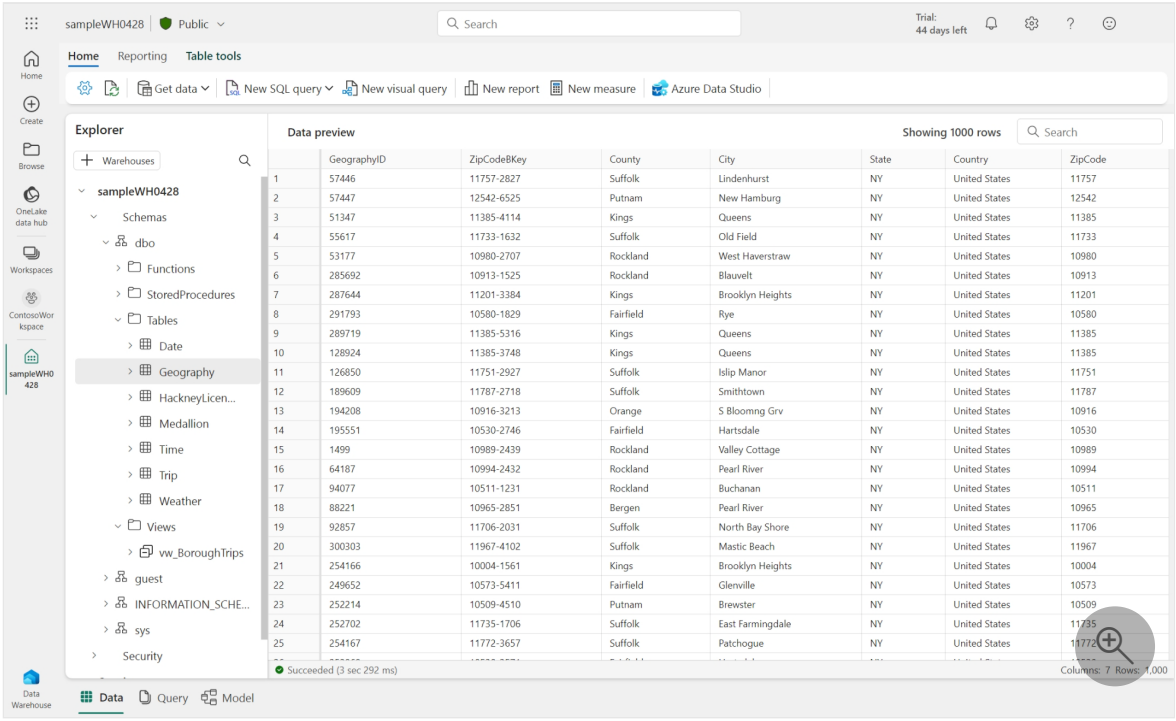
2. Provide the name for your sample warehouse and select **Create**.



3. The create action creates a new Warehouse and start loading sample data into it. The data loading takes few minutes to complete.



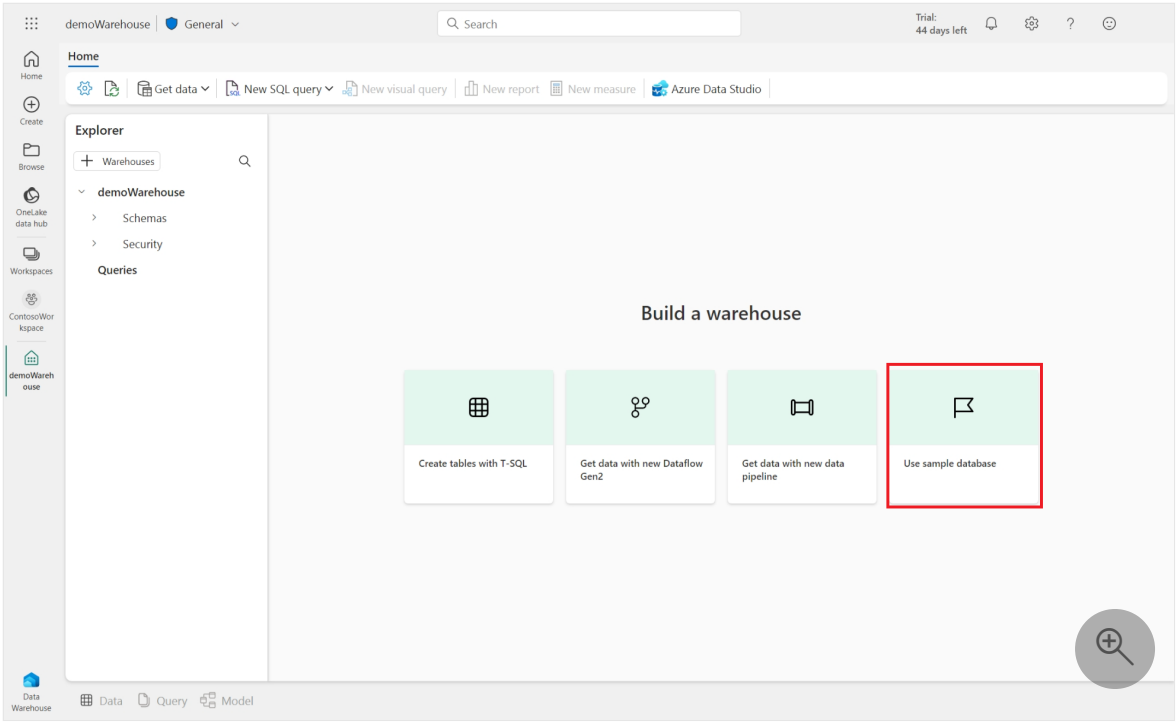
4. On completion of loading sample data, the warehouse opens with data loaded into tables and views to query.



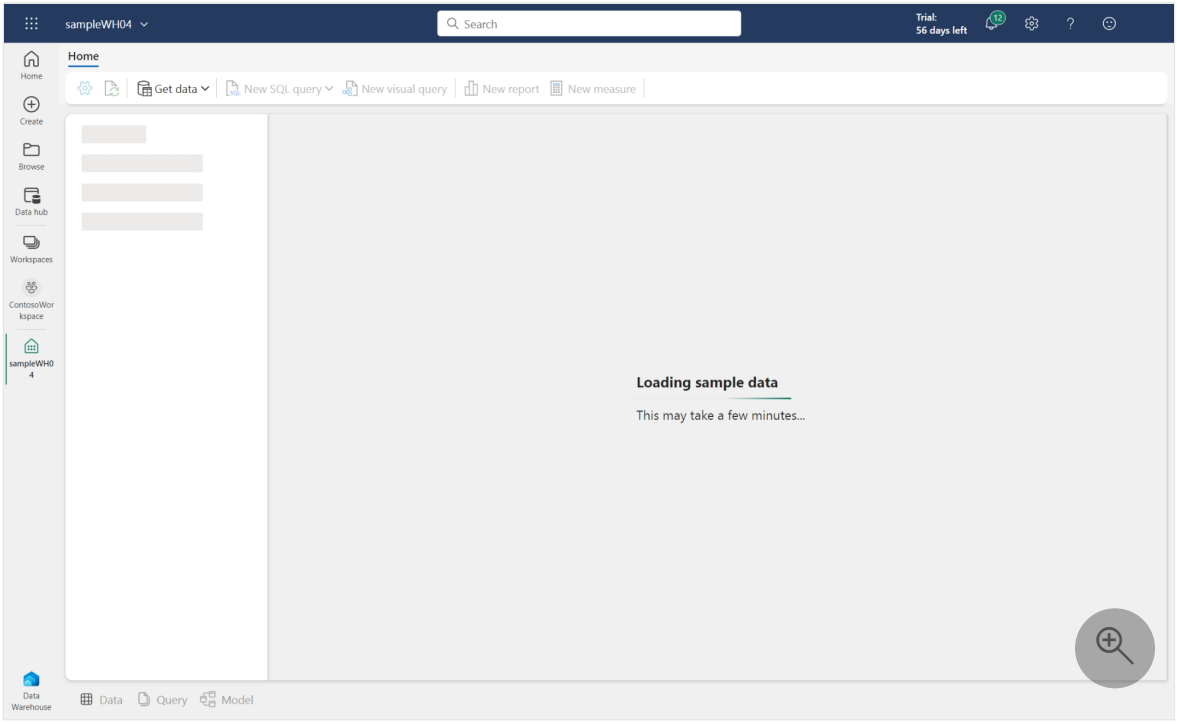
Load sample data into existing warehouse

For more information on how to create a warehouse, see Create a Synapse Data Warehouse.

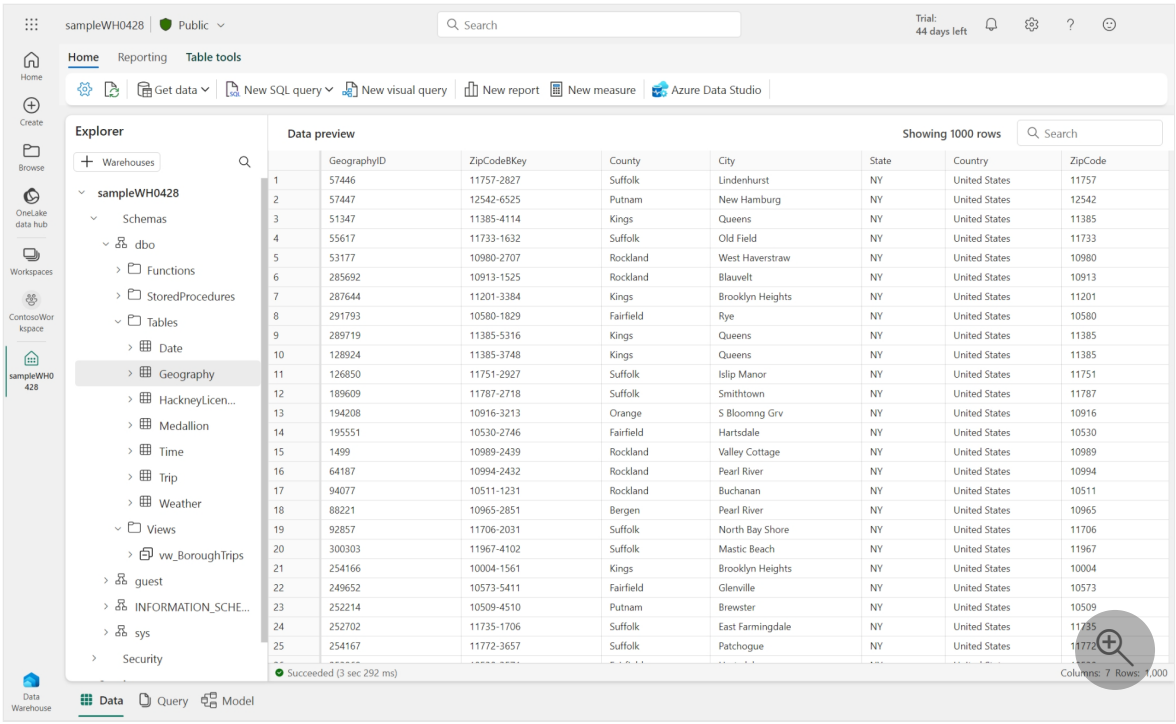
1. Once you have created your warehouse, you can load sample data into warehouse from **Use sample database** card.



2. The data loading takes few minutes to complete.



3. On completion of loading sample data, the warehouse displays data loaded into tables and views to query.



Sample scripts

SQL

```
/*********************************************
Get number of trips performed by each medallion
*********************************************/

SELECT
    M.MedallionID
    ,M.MedallionCode
    ,COUNT(T.TripDistanceMiles) AS TotalTripCount
FROM
    dbo.Trip AS T
JOIN
    dbo.Medallion AS M
ON
    T.MedallionID=M.MedallionID
GROUP BY
    M.MedallionID
    ,M.MedallionCode

/*************************************************
How many passengers are being picked up on each trip?
*************************************************/
SELECT
    PassengerCount,
    COUNT(*) AS CountOfTrips
FROM
    dbo.Trip
WHERE
    PassengerCount > 0
```

```
GROUP BY
    PassengerCount
ORDER BY
    PassengerCount

/*************************************************************************
******
What is the distribution of trips by hour on working days (non-holiday
weekdays)?
*************************************************************************
*****/
SELECT
    ti.HourlyBucket,
    COUNT(*) AS CountOfTrips
FROM dbo.Trip AS tr
INNER JOIN dbo.Date AS d
    ON tr.DateID = d.DateID
INNER JOIN dbo.Time AS ti
    ON tr.PickupTimeID = ti.TimeID
WHERE
    d.IsWeekday = 1
    AND d.IsHolidayUSA = 0
GROUP BY
    ti.HourlyBucket
ORDER BY
    ti.HourlyBucket
```

Tip

You can proceed with either a blank Warehouse or a sample Warehouse to continue this series of Get Started steps.

Next steps

Create tables in Warehouse

Create tables in the Warehouse in Microsoft Fabric

Article • 05/23/2023

Applies to: ✅ Warehouse in Microsoft Fabric

To get started, you must complete the following prerequisites:

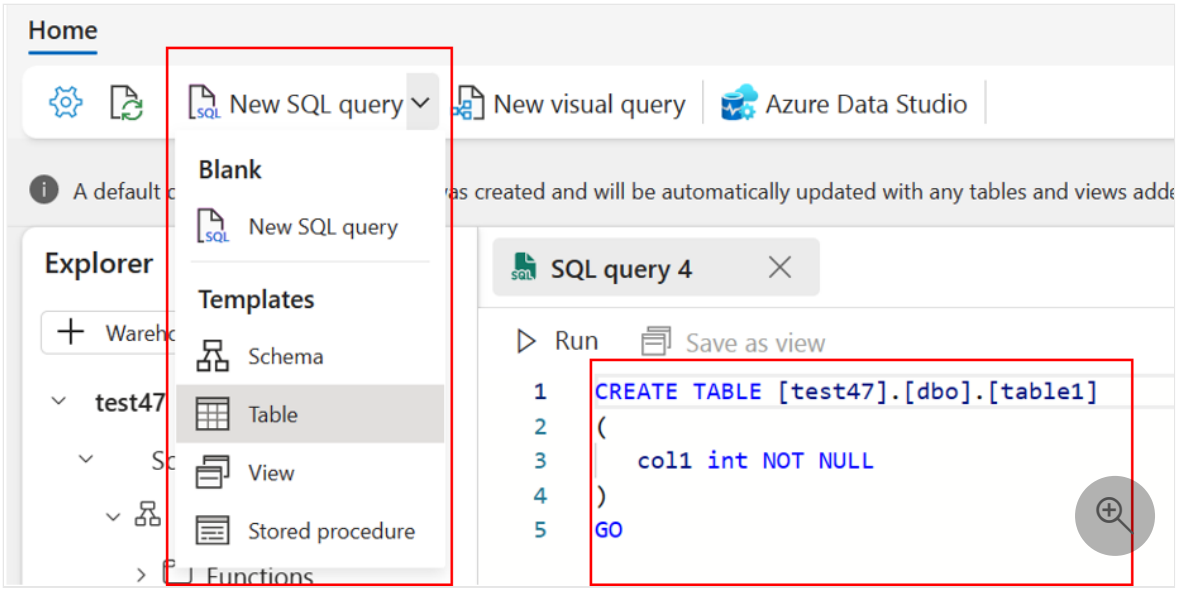
- Have access to a Warehouse within a Premium capacity workspace with contributor or above permissions.
- Choose your query tool. This tutorial features the SQL query editor in the Microsoft Fabric portal, but you can use any T-SQL querying tool.
  - Use the SQL query editor in the Microsoft Fabric portal.

For more information on connecting to your Warehouse in Microsoft Fabric, see Connectivity.

> **ⓘ Important**
>
> Microsoft Fabric is currently in PREVIEW. This information relates to a prerelease product that may be substantially modified before it's released. Microsoft makes no warranties, expressed or implied, with respect to the information provided here.

Create a new table in the SQL query editor with templates

1. In the warehouse editor ribbon, locate the **New SQL query** button.
2. Instead of selecting **New SQL query**, you can select the dropdown arrow to see **Templates** to create T-SQL objects.
3. Select **Table**, and an autogenerated CREATE TABLE script template appears in your new SQL query window, as shown in the following image.



4. Modify the `CREATE TABLE` template to suit your new table.

5. Select **Run** to create the table.

To learn more about supported table creation in Warehouse in Microsoft Fabric, see Tables in data warehousing in Microsoft Fabric and Data types in Microsoft Fabric.

Next steps

Ingest data into your Warehouse using data pipelines

Ingest data into your Warehouse using data pipelines

Article • 05/23/2023

Applies to: Warehouse in Microsoft Fabric

Data pipelines offer an alternative to using the COPY command through a graphical user interface. A data pipeline is a logical grouping of activities that together perform a data ingestion task. Pipelines allow you to manage extract, transform, and load (ETL) activities instead of managing each one individually.

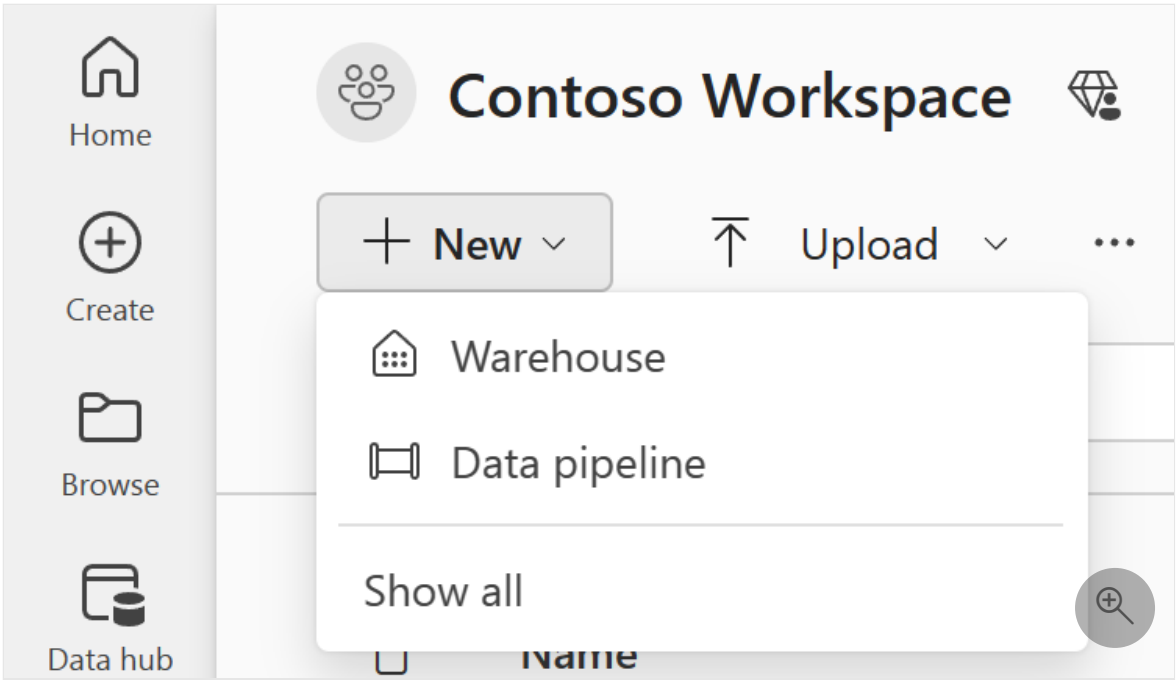
In this tutorial, you'll create a new pipeline that loads sample data into a Warehouse in Microsoft Fabric.

> **Note**
>
> Some features from Azure Data Factory are not available in Microsoft Fabric, but the concepts are interchangeable. You can learn more about Azure Data Factory and Pipelines on **Pipelines and activities in Azure Data Factory and Azure Synapse Analytics**. For a quickstart, visit **Quickstart: Create your first pipeline to copy data**.

> **Important**
>
> Microsoft Fabric is currently in PREVIEW. This information relates to a prerelease product that may be substantially modified before it's released. Microsoft makes no warranties, expressed or implied, with respect to the information provided here.

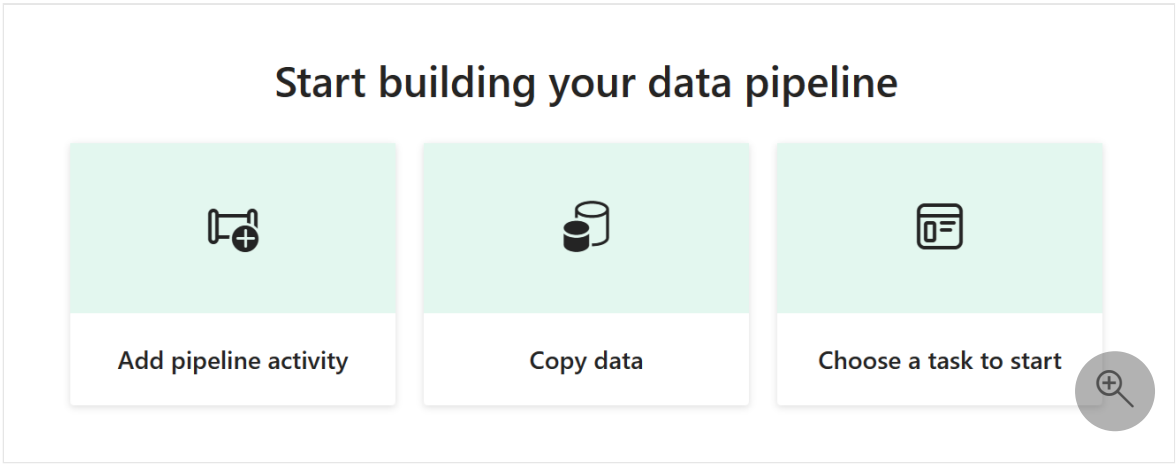
Create a data pipeline

1. To create a new pipeline navigate to your workspace, select the **+New** button, and select **Data pipeline**.



2. In the **New pipeline** dialog, provide a name for your new pipeline and select **Create**.

3. You'll land in the pipeline canvas area, where you see three options to get started: **Add a pipeline activity**, **Copy data**, and **Choose a task to start**.

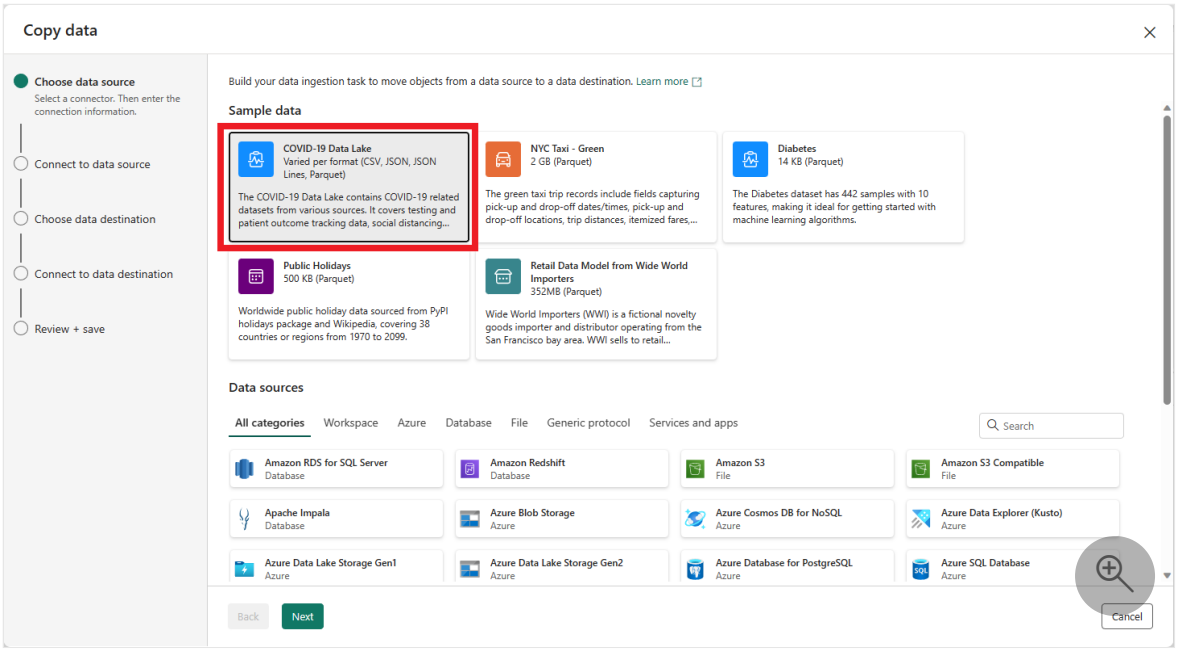


Each of these options offers different alternatives to create a pipeline:

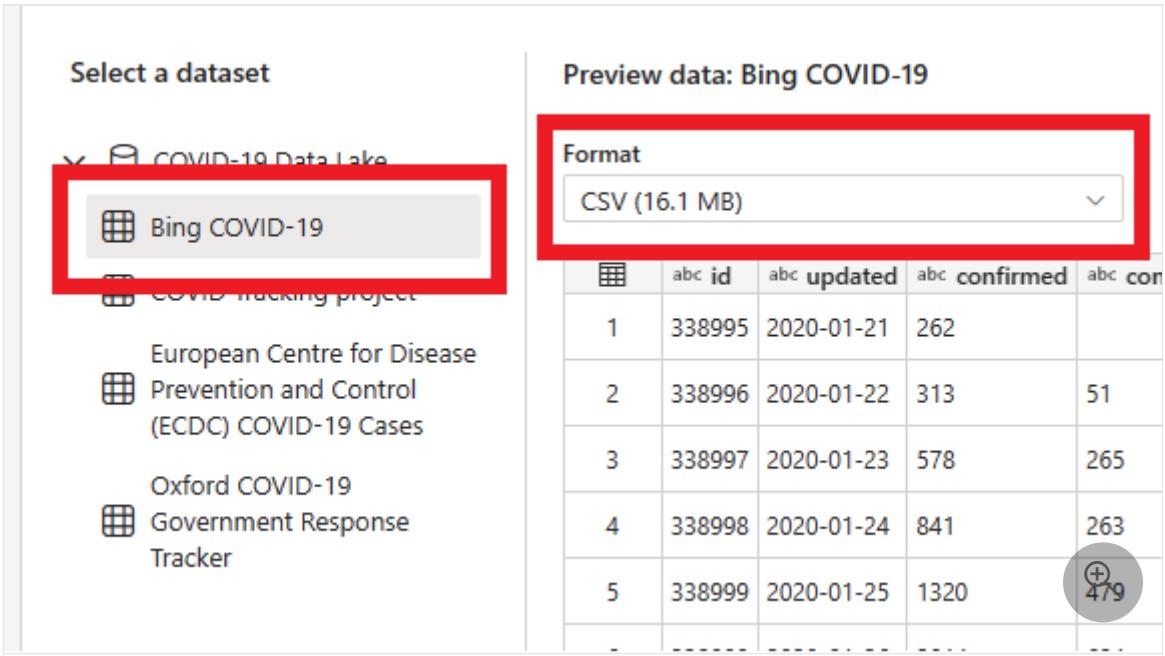
- **Add pipeline activity**: this option launches the pipeline editor, where you can create new pipelines from scratch by using pipeline activities.
- **Copy data**: this option launches a step-by-step assistant that helps you select a data source, a destination, and configure data load options such as the column mappings. On completion, it creates a new pipeline activity with a **Copy Data** task already configured for you.
- **Choose a task to start**: this option launches a set of predefined templates to help get you started with pipelines based on different scenarios.

Pick the **Copy data** option to launch the **Copy assistant**.

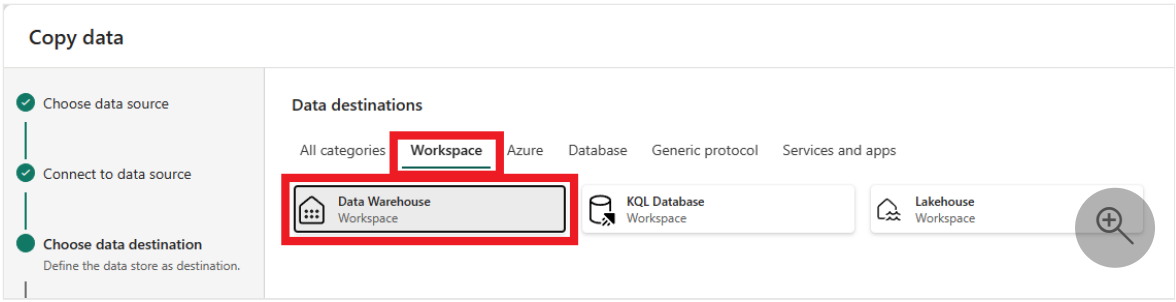
4. The first page of the **Copy data** assistant helps you pick your own data from various data sources, or select from one of the provided samples to get started. For this tutorial, we'll use the **COVID-19 Data Lake** sample. Select this option and select **Next**.



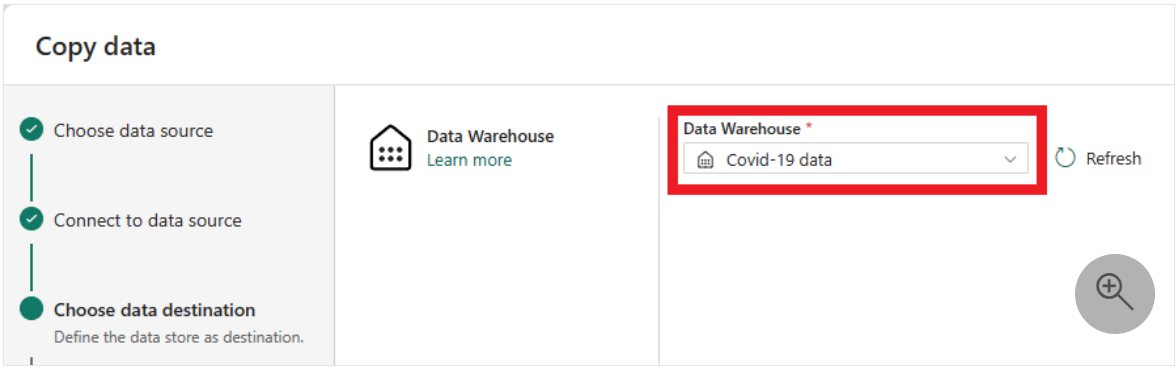
5. In the next page, you can select a dataset, the source file format, and preview the selected dataset. Select the **Bing COVID-19** dataset, the **CSV** format, and select **Next**.



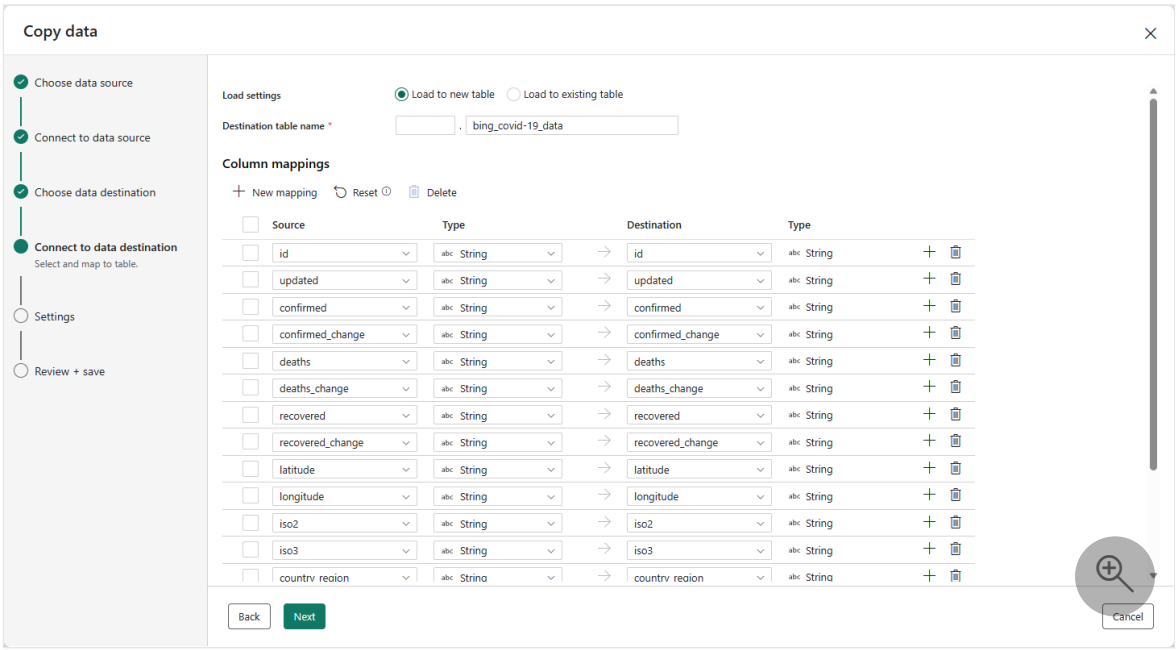
6. The next page, **Data destinations**, allows you to configure the type of the destination dataset. We'll load data into a warehouse in our workspace, so select the **Warehouse** tab, and the **Data Warehouse** option. Select **Next**.



7. Now it's time to pick the warehouse to load data into. Select your desired warehouse in the dropdown box and select **Next**.



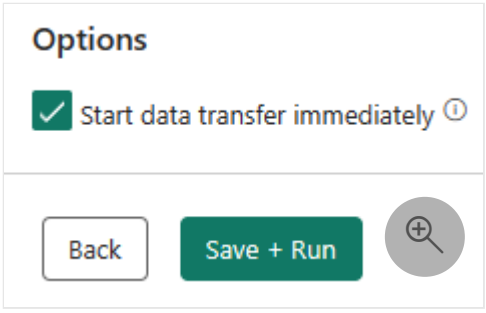
8. The last step to configure the destination is to provide a name to the destination table and configure the column mappings. Here you can choose to load the data to a new table or to an existing one, provide a schema and table names, change column names, remove columns, or change their mappings. You can accept the defaults, or adjust the settings to your preference.



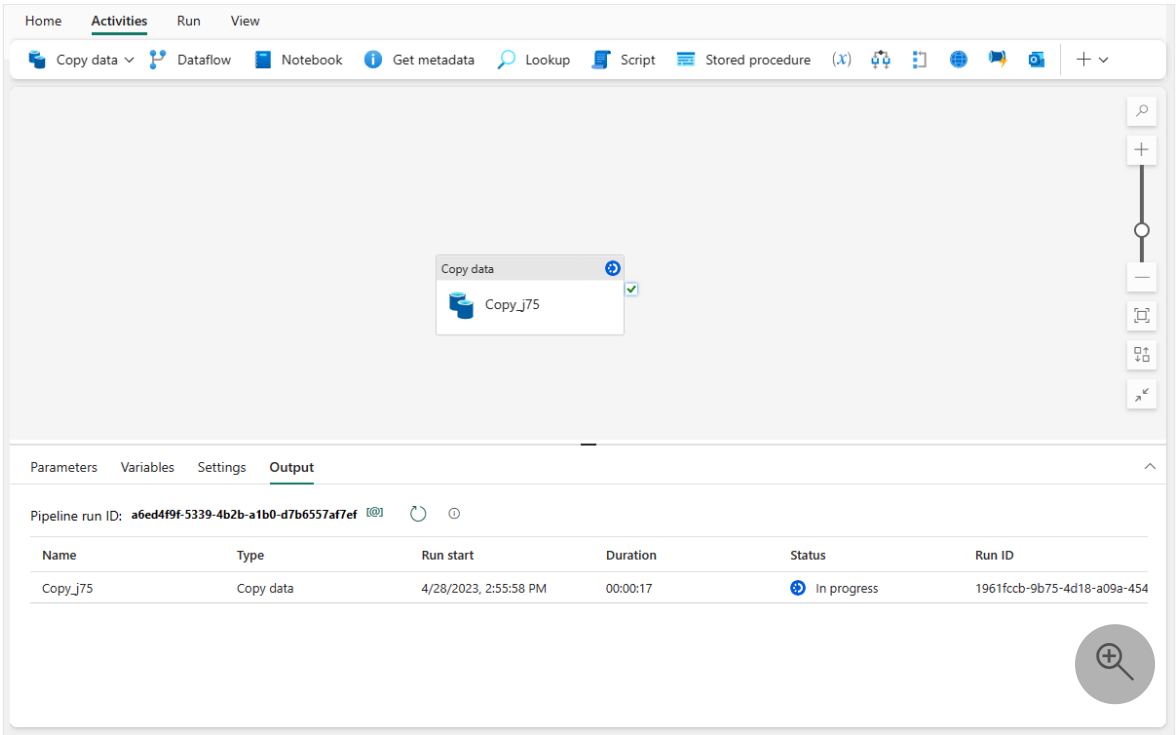
When you're done reviewing the options, select **Next**.

9. The next page gives you the option to use staging, or provide advanced options for the data copy operation (which uses the T-SQL COPY command). Review the options without changing them and select **Next**.

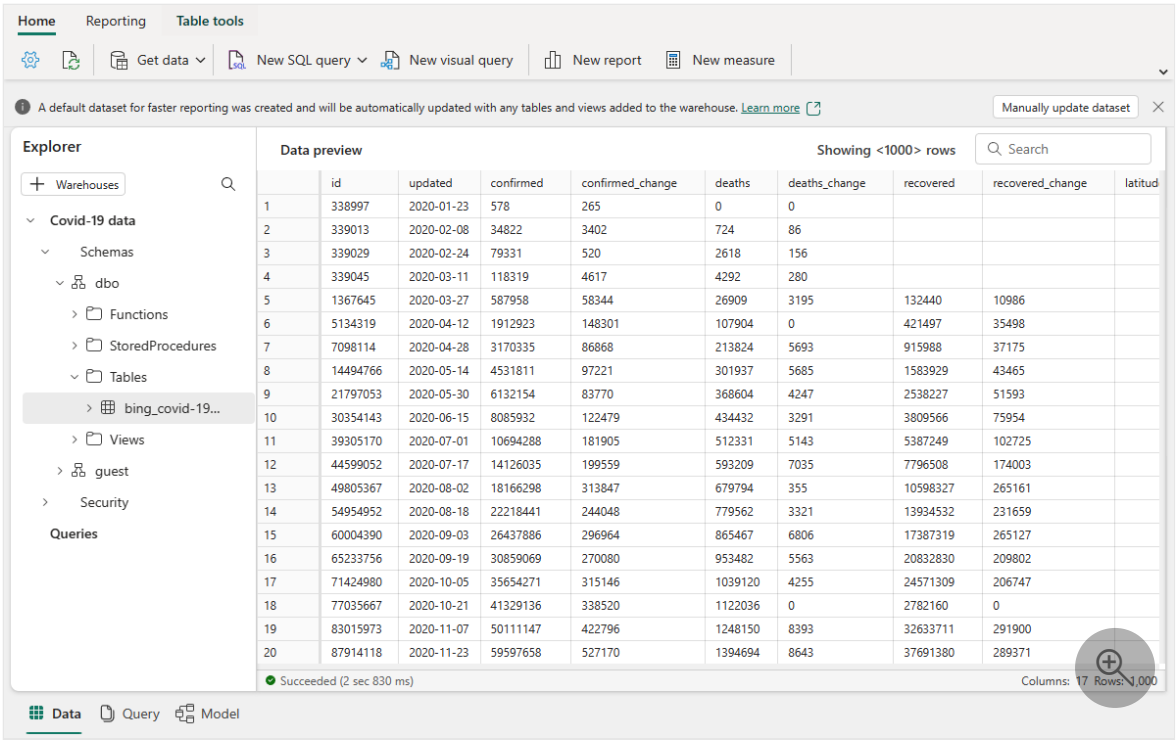
10. The last page in the assistant offers a summary of the copy activity. Select the option **Start data transfer immediately** and select **Save + Run**.



11. You are directed to the pipeline canvas area, where a new Copy Data activity is already configured for you. The pipeline starts to run automatically. You can monitor the status of your pipeline in the **Output** pane:



12. After a few seconds, your pipeline finishes successfully. Navigating back to your warehouse, you can select your table to preview the data and confirm that the copy operation concluded.



For more on data ingestion into your Warehouse in Microsoft Fabric, visit:

- Ingesting data into the Warehouse
- Ingest data into your Warehouse using the COPY statement
- Ingest data into your Warehouse using Transact-SQL

Next steps

Query the SQL Endpoint or Warehouse in Microsoft Fabric

Query the SQL Endpoint or Warehouse in Microsoft Fabric

Article • 05/23/2023

Applies to: ✅ SQL Endpoint and Warehouse in Microsoft Fabric

To get started with this tutorial, check the following prerequisites:

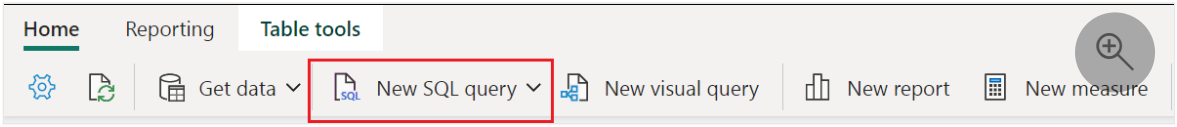
- You should have access to a SQL Endpoint or Warehouse within a Premium capacity workspace with contributor or above permissions.
- Choose your querying tool.
  - Use the SQL query editor in the Microsoft Fabric portal.
  - Use the Visual query editor in the Microsoft Fabric portal.
- Alternatively, you can use any of the below tools to connect to your SQL Endpoint or Warehouse via T-SQL connection string. For more information, see Connectivity.
  - Download SQL Server Management Studio (SSMS).
  - Download Azure Data Studio.

> **Note**
>
> Review the **T-SQL surface area** for SQL Endpoint or Warehouse in Microsoft Fabric.

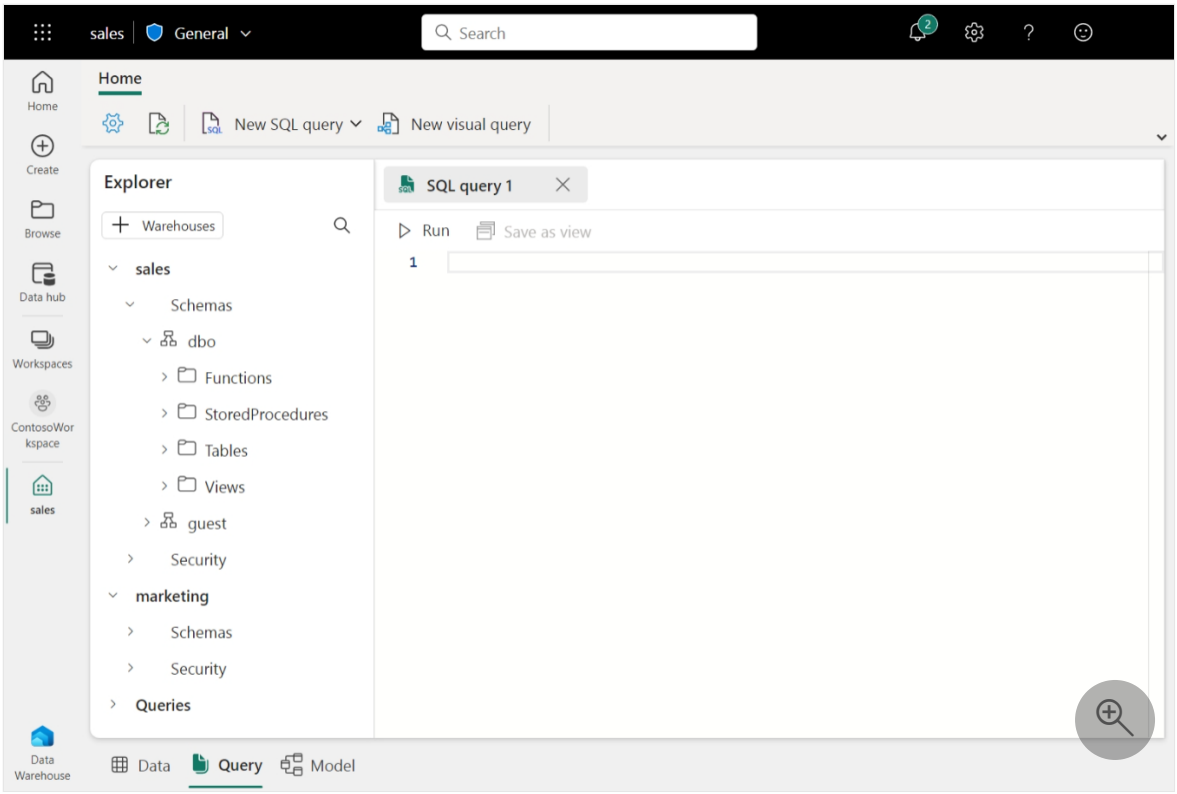
> **Important**
>
> Microsoft Fabric is currently in PREVIEW. This information relates to a prerelease product that may be substantially modified before it's released. Microsoft makes no warranties, expressed or implied, with respect to the information provided here.

Run a new query in SQL query editor

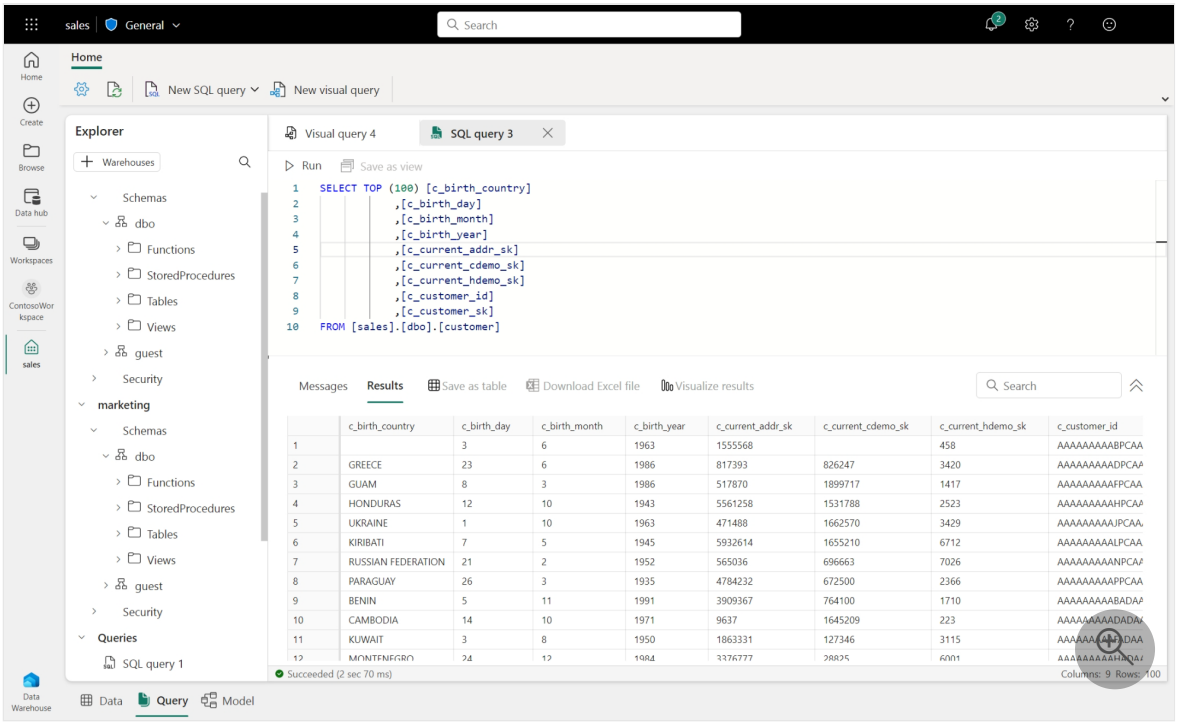
1. Open a **New SQL query** window.



2. A new tab appears for you to write a SQL query.

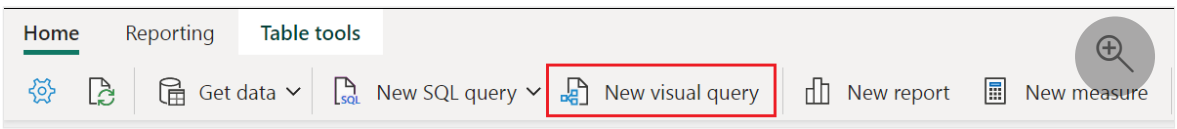


3. Write a SQL query and run it.

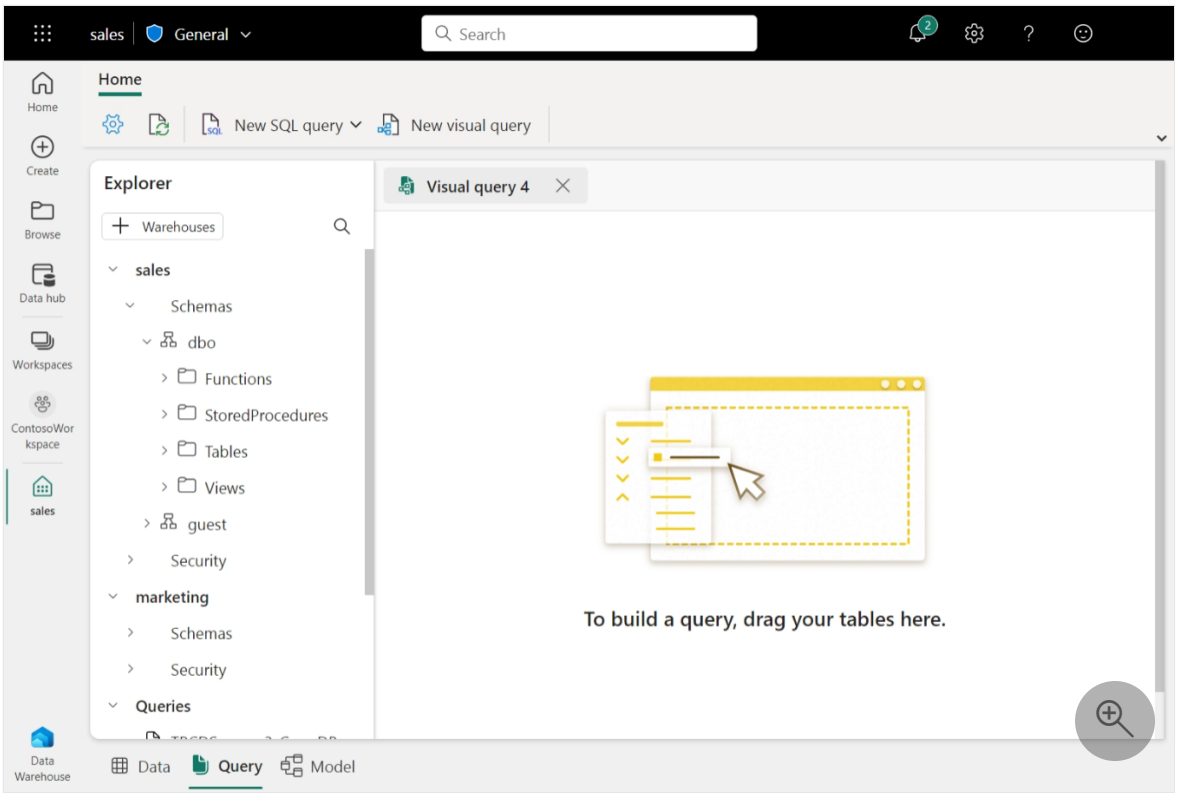


Run a new query in Visual query editor

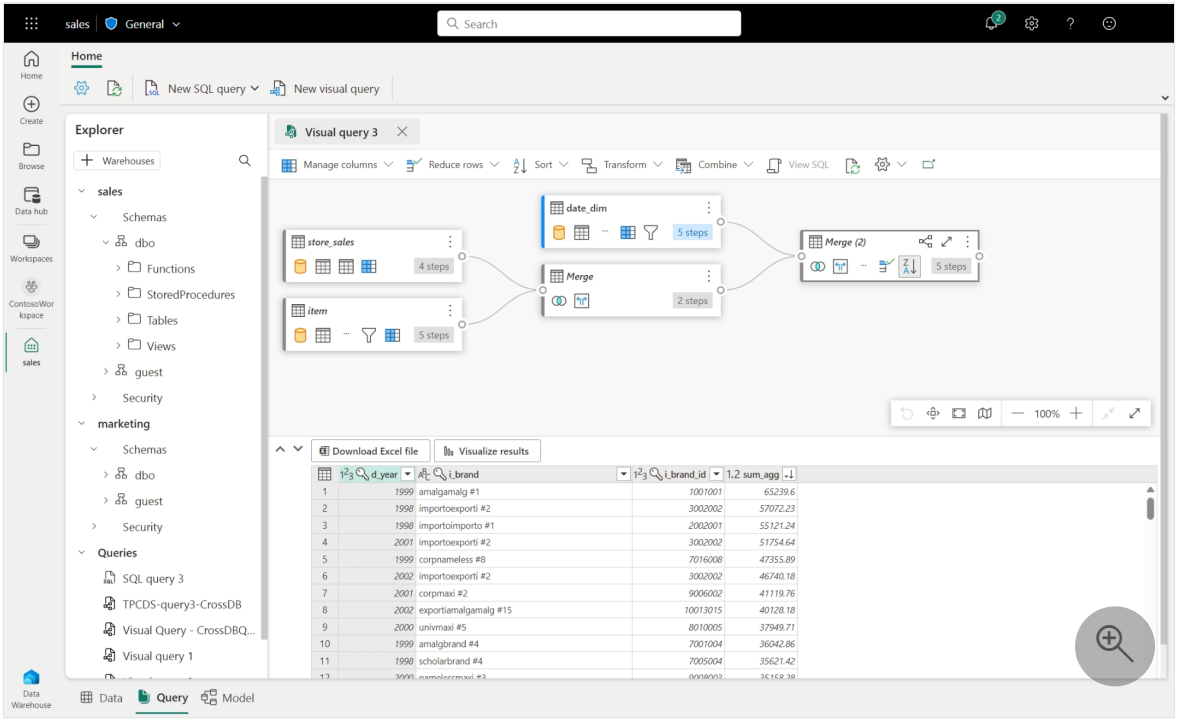
1. Open a **New visual query** window.



2. A new tab appears for you to create a visual query.



3. Drag and drop tables from the object **Explorer** to **Visual query editor** window to create a query.

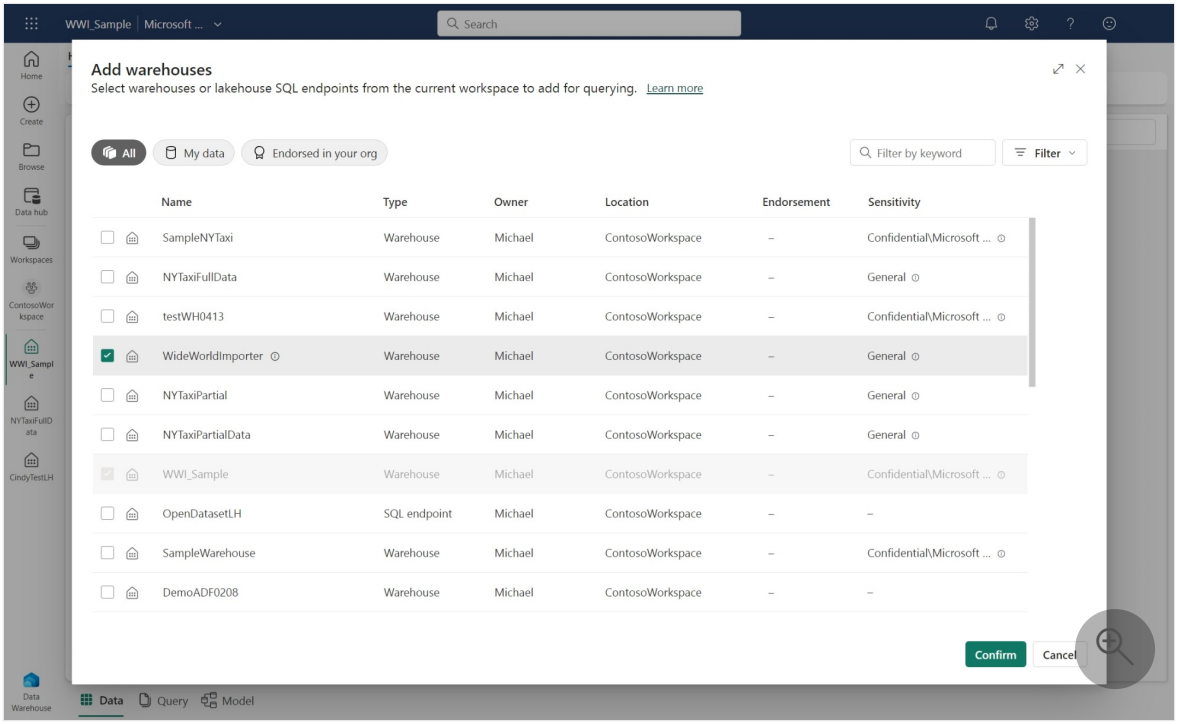


Write a cross-database query

You can write cross database queries to databases in the current active workspace in Microsoft Fabric.

There are several ways you can write cross-database queries within the same Microsoft Fabric workspace, in this section we explore examples. You can join tables or views to run cross-warehouse queries within current active workspace.

1. Add SQL Endpoint or Warehouse from your current active workspace to object **Explorer** using + **Warehouses** action. When you select SQL Endpoint or Warehouse from the dialog, it gets added into the object **Explorer** for referencing when writing a SQL query or creating Visual query.



2. You can reference the table from added databases using three-part naming. In the following example, use the three-part name to refer to `ContosoSalesTable` in the added database `ContosoLakehouse`.

SQL

```
SELECT *
FROM ContosoLakehouse.dbo.ContosoSalesTable AS Contoso
INNER JOIN Affiliation
ON Affiliation.AffiliationId = Contoso.RecordTypeID;
```

3. Using three-part naming to reference the databases/tables, you can join multiple databases.

SQL

```
SELECT *
FROM ContosoLakehouse.dbo.ContosoSalesTable AS Contoso
INNER JOIN My_lakehouse.dbo.Affiliation
ON My_lakehouse.dbo.Affiliation.AffiliationId = Contoso.RecordTypeID;
```

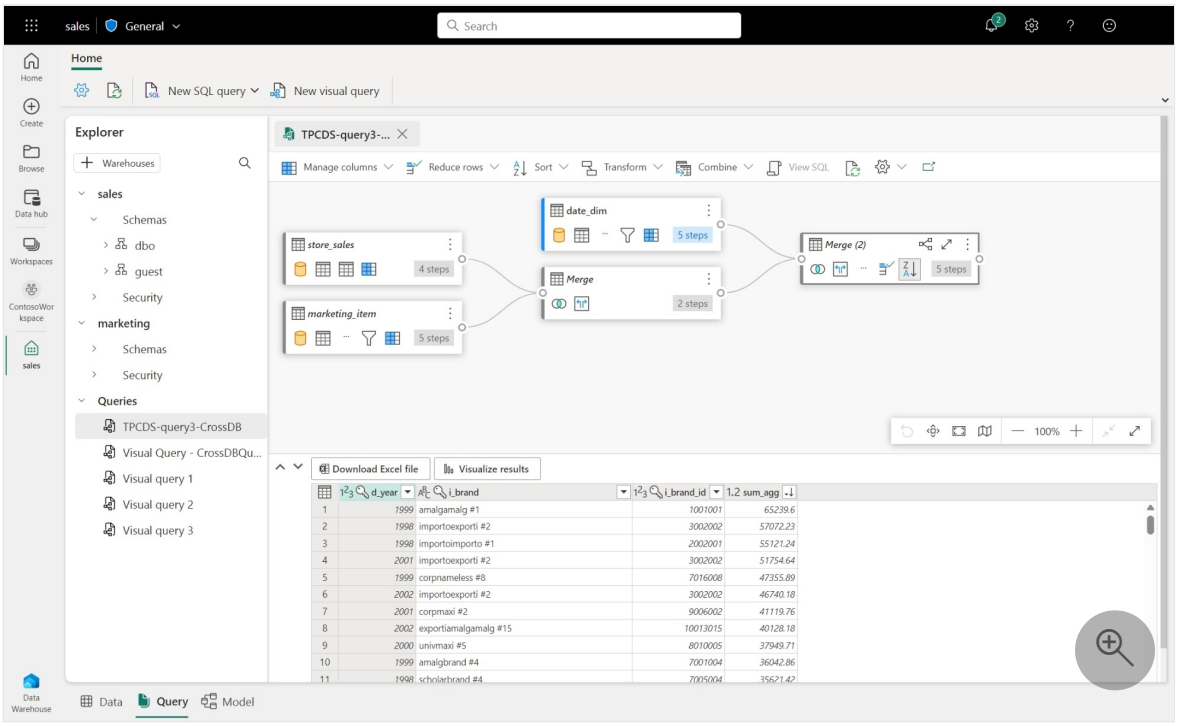
4. For more efficient and longer queries, you can use aliases.

```
SQL

SELECT *
FROM ContosoLakehouse.dbo.ContosoSalesTable AS Contoso
INNER JOIN My_lakehouse.dbo.Affiliation as MyAffiliation
ON MyAffiliation.AffiliationId = Contoso.RecordTypeID;
```

5. Using three-part naming to reference the database and tables, you can insert data from one database to another.

```
SQL

INSERT INTO ContosoWarehouse.dbo.Affiliation
SELECT *
FROM My_Lakehouse.dbo.Affiliation;
```

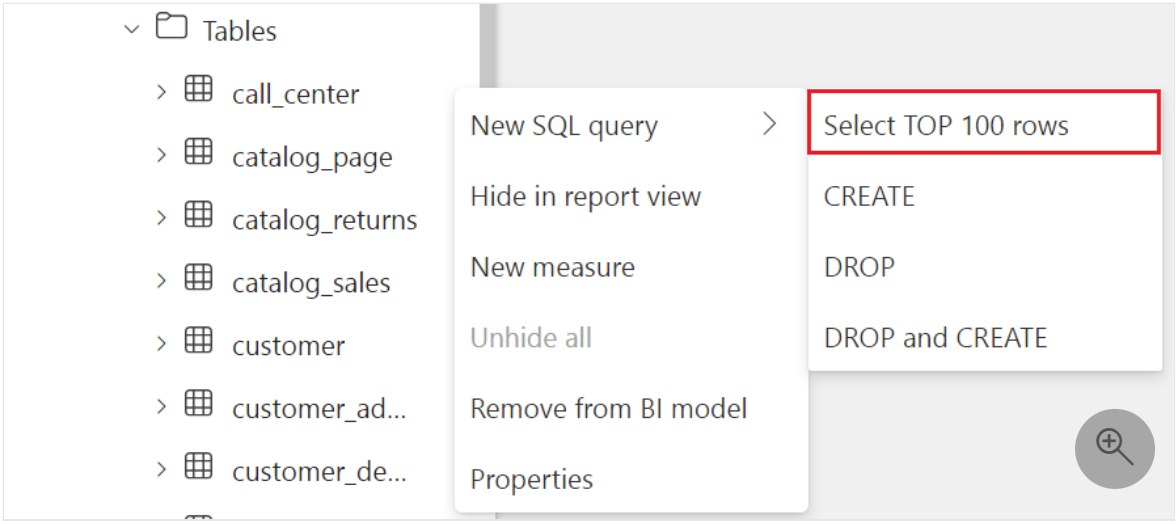
6. You can drag and drop tables from added databases to **Visual query editor** to create a cross-database query.



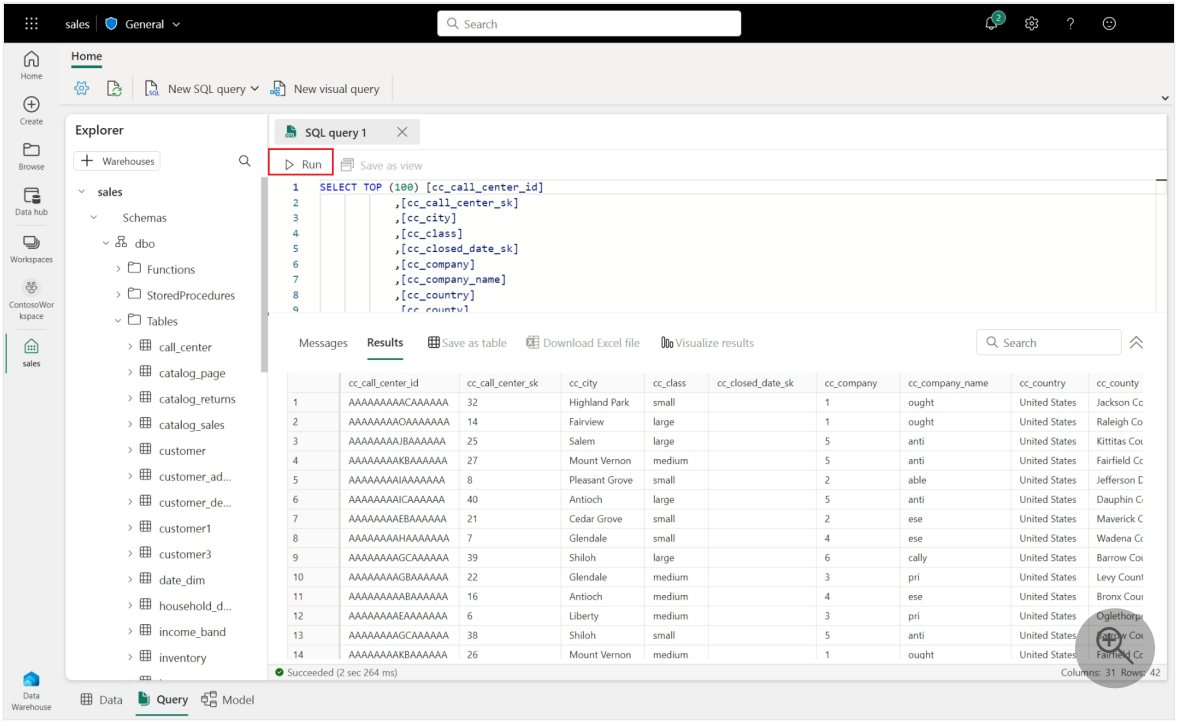
SELECT Top 100 Rows from the Explorer

1. After opening your warehouse from the workspace, expand your database, schema and tables folder in the object **Explorer** to see all tables listed.

2. Right-click on the table that you would like to query and select **Select TOP 100 rows**.



3. Once the script is automatically generated, select the **Run** button to run the script and see the results.



> **Note**
>
> At this time, there's limited T-SQL functionality. See T-SQL surface area for a list of T-SQL commands that are currently not available.

Next steps

Create reports on data warehousing in Microsoft Fabric

Create reports on data warehousing in Microsoft Fabric

Article • 05/23/2023

Applies to: SQL Endpoint and Warehouse in Microsoft Fabric

Microsoft Fabric lets you create reusable and default Power BI datasets to create reports in various ways in Power BI. This article describes the various ways you can use your Warehouse or SQL Endpoint, and their default Power BI datasets, to create reports.

For example, you can establish a live connection to a shared dataset in the Power BI service and create many different reports from the same dataset. You can create a data model in Power BI Desktop and publish to the Power BI service. Then, you and others can create multiple reports in separate .pbix files from that common data model and save them to different workspaces.

Advanced users can build reports from a warehouse using a composite model or using the SQL connection string.

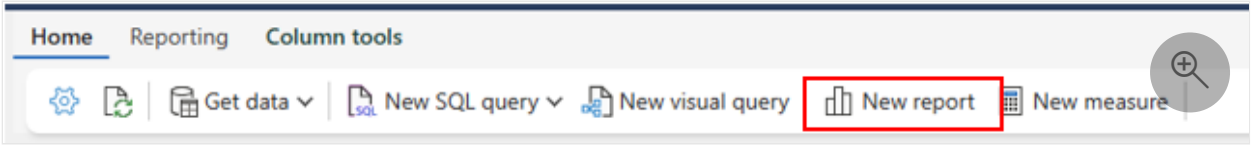
Reports that use the Warehouse or SQL Endpoint can be created in either of the following two tools:

- Power BI service
- Power BI Desktop

> **Important**
>
> Microsoft Fabric is currently in PREVIEW. This information relates to a prerelease product that may be substantially modified before it's released. Microsoft makes no warranties, expressed or implied, with respect to the information provided here.

Create reports using the Power BI service

Within the warehouse experience, using the ribbon and the main home tab, navigate to the **New report** button. This option provides a native, quick way to create report built on top of the default Power BI dataset.



If no tables have been added to the default Power BI dataset, the dialog first automatically adds tables, prompting the user to confirm or manually select the tables included in the canonical default dataset first, ensuring there's always data first.

With a default dataset that has tables, the **New report** opens a browser tab to the report editing canvas to a new report that is built on the dataset. When you save your new report you're prompted to choose a workspace, provided you have write permissions for that workspace. If you don't have write permissions, or if you're a free user and the dataset resides in a Premium capacity workspace, the new report is saved in your **My workspace**.

For more information on how to create reports using the Power BI service, see Create reports in the Power BI service.

Create reports using Power BI Desktop

You can build reports from datasets with **Power BI Desktop** using a Live connection to the default dataset. For information on how to make the connection, see connect to datasets from Power BI Desktop.

For a tutorial with Power BI Desktop, see Get started with Power BI Desktop. For advanced situations where you want to add more data or change the storage mode, see use composite models in Power BI Desktop.

You can use integrated Data hub experience in Power BI Desktop to select your SQL Endpoint or Warehouse to make a connection and build reports.

Alternatively, you can complete the following steps to connect to a warehouse in Power BI Desktop:

1. Navigate to the warehouse settings in your workspace and copy the SQL connection string. Or, right-click on the Warehouse or SQL Endpoint in your workspace and select **Copy SQL connection string**.
2. Select the **Warehouse (preview) connector** from the **Get data** or connect to the default dataset from **Data hub**.
3. Paste the SQL connection string into the connector dialog.
4. For authentication, select *organizational account*.
5. Authenticate using Azure Active Directory - MFA.
6. Select **Connect**.
7. Select the data items you want to include or not include in your dataset.

Next steps

- Data modeling in the default Power BI dataset in Microsoft Fabric
- Create reports in the Power BI service in Microsoft Fabric

Data warehouse tutorial introduction

Article • 05/23/2023

Applies to: ✅ SQL Endpoint and Warehouse in Microsoft Fabric

Microsoft Fabric provides a one-stop shop for all the analytical needs for every enterprise. It covers the complete spectrum of services including data movement, data lake, data engineering, data integration and data science, real time analytics, and business intelligence. With Microsoft Fabric, there's no need to stitch together different services from multiple vendors. Instead, the customer enjoys an end-to-end, highly integrated, single comprehensive product that is easy to understand, onboard, create and operate. No other product on the market offers the breadth, depth, and level of integration that Microsoft Fabric offers. Additionally, Microsoft Purview is included by default in every tenant to meet compliance and governance needs.

> **ⓘ Important**
>
> Microsoft Fabric is currently in PREVIEW. This information relates to a prerelease product that may be substantially modified before it's released. Microsoft makes no warranties, expressed or implied, with respect to the information provided here.

Purpose of this tutorial

While many concepts in Microsoft Fabric may be familiar to data and analytics professionals, it can be challenging to apply those concepts in a new environment. This tutorial has been designed to walk step-by-step through an end-to-end scenario from data acquisition to data consumption to build a basic understanding of the Microsoft Fabric user experience, the various experiences and their integration points, and the Microsoft Fabric professional and citizen developer experiences.

The tutorials aren't intended to be a reference architecture, an exhaustive list of features and functionality, or a recommendation of specific best practices.

Data warehouse end-to-end scenario

As prerequisites to this tutorial, complete the following steps:

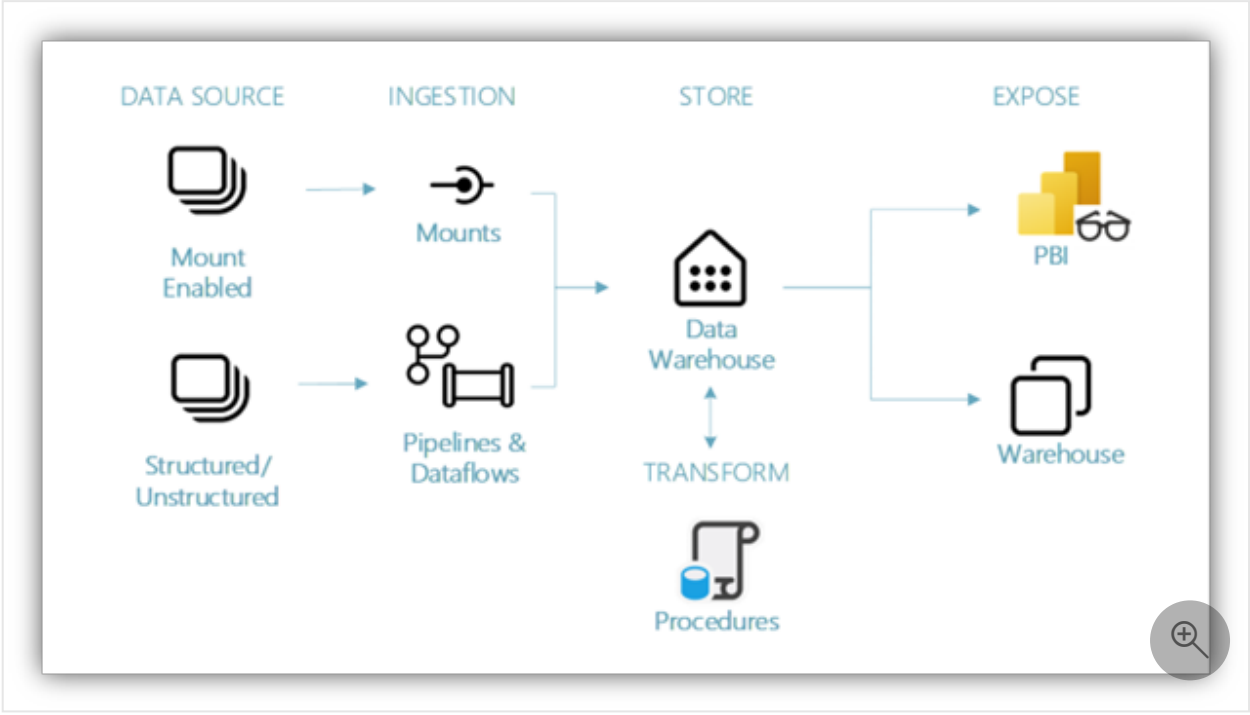
1. Sign into your Power BI online account, or if you don't have an account yet, sign up for a free trial.

2. Enable Microsoft Fabric in your tenant.

In this tutorial, you take on the role of a Warehouse developer at the fictional *Wide World Importers* company and complete the following steps in the Microsoft Fabric portal to build and implement an end-to-end data warehouse solution:

1. Create a Microsoft Fabric workspace.
2. Create a Warehouse.
3. Ingest data from source to the data warehouse dimensional model with a data pipeline.
4. Create tables in your Warehouse.
5. Load data with T-SQL with the SQL query editor.
6. Transform the data to create aggregated datasets using T-SQL.
7. Use the visual query editor to query the data warehouse.
8. Analyze data with a notebook.
9. Create and execute cross-warehouse queries with SQL query editor.
10. Create Power BI reports using DirectLake mode to analyze the data in place.
11. Build a report from the **Data Hub**.
12. Clean up resources by deleting the workspace and other items.

Data warehouse end-to-end architecture



Data sources - Microsoft Fabric makes it easy and quick to connect to Azure Data Services, other cloud platforms, and on-premises data sources to ingest data from.

Ingestion - With 200+ native connectors as part of the Microsoft Fabric pipeline and with drag and drop data transformation with dataflow, you can quickly build insights for your organization. Shortcut is a new feature in Microsoft Fabric that provides a way to connect to existing data without having to copy or move it. You can find more details about the Shortcut feature later in this tutorial.

Transform and store - Microsoft Fabric standardizes on Delta Lake format, which means all the engines of Microsoft Fabric can read and work on the same dataset stored in OneLake - no need for data duplicity. This storage allows you to build a data warehouse or data mesh based on your organizational need. For transformation, you can choose either low-code or no-code experience with pipelines/dataflows or use T-SQL for a code first experience.

Consume - Data from the data warehouse can be consumed by Power BI, the industry leading business intelligence tool, for reporting and visualization. Each data warehouse comes with a built-in TDS/SQL endpoint for easily connecting to and querying data from other reporting tools, when needed. When a data warehouse is created, a secondary item, called a default dataset, is generated at the same time with the same name. You can use the default dataset to start visualizing data with just a couple of steps.

Sample data

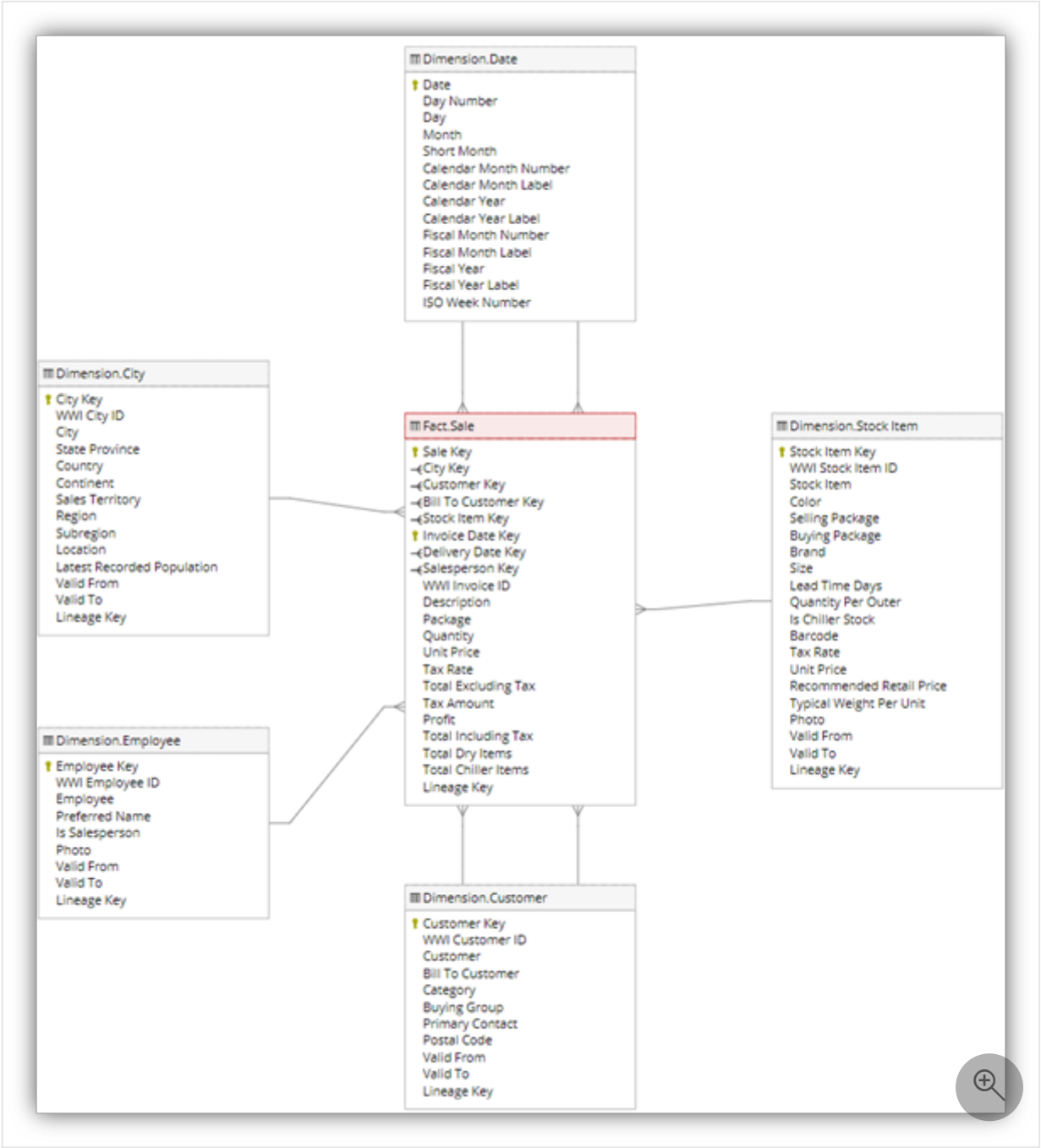
For sample data, we use the Wide World Importers (WWI) sample database. For our data warehouse end-to-end scenario, we have generated sufficient data for a sneak peek into the scale and performance capabilities of the Microsoft Fabric platform.

Wide World Importers (WWI) is a wholesale novelty goods importer and distributor operating from the San Francisco Bay area. As a wholesaler, WWI's customers are mostly companies who resell to individuals. WWI sells to retail customers across the United States including specialty stores, supermarkets, computing stores, tourist attraction shops, and some individuals. WWI also sells to other wholesalers via a network of agents who promote the products on WWI's behalf. To earn more about their company profile and operation, see Wide World Importers sample databases for Microsoft SQL.

Typically, you would bring data from transactional systems (or line of business applications) into a data lake or data warehouse staging area. However, for this tutorial, we use the dimensional model provided by WWI as our initial data source. We use it as the source to ingest the data into a data warehouse and transform it through T-SQL.

Data model

While the WWI dimensional model contains multiple fact tables, for this tutorial we focus on the Sale Fact table and its related dimensions only, as follows, to demonstrate this end-to-end data warehouse scenario:



Next steps

Tutorial: Create a Microsoft Microsoft Fabric workspace

Tutorial: Create a Microsoft Fabric workspace

Article • 05/23/2023

Applies to: SQL Endpoint and Warehouse in Microsoft Fabric

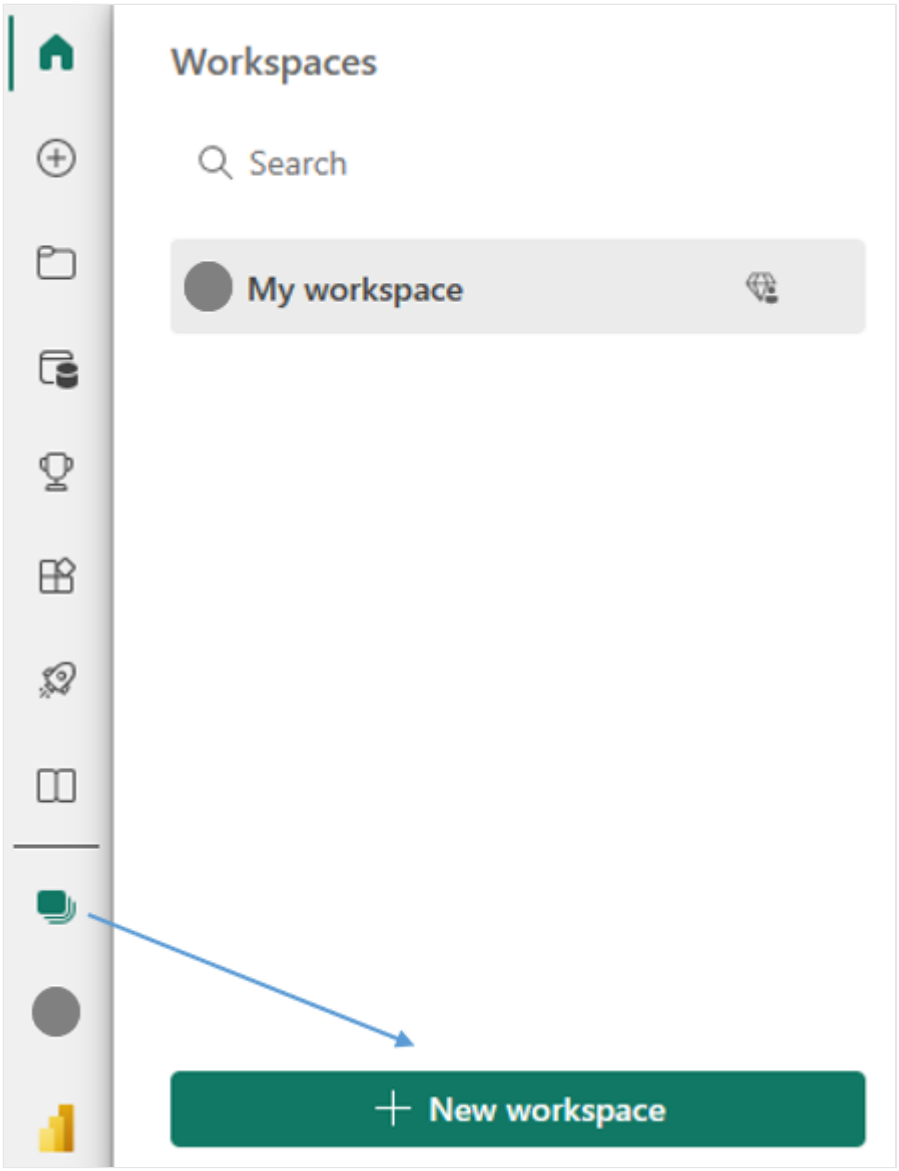
Before you can create a warehouse, you need to create a workspace where you'll build out the remainder of the tutorial.

> **Important**
>
> Microsoft Fabric is currently in PREVIEW. This information relates to a prerelease product that may be substantially modified before it's released. Microsoft makes no warranties, expressed or implied, with respect to the information provided here.

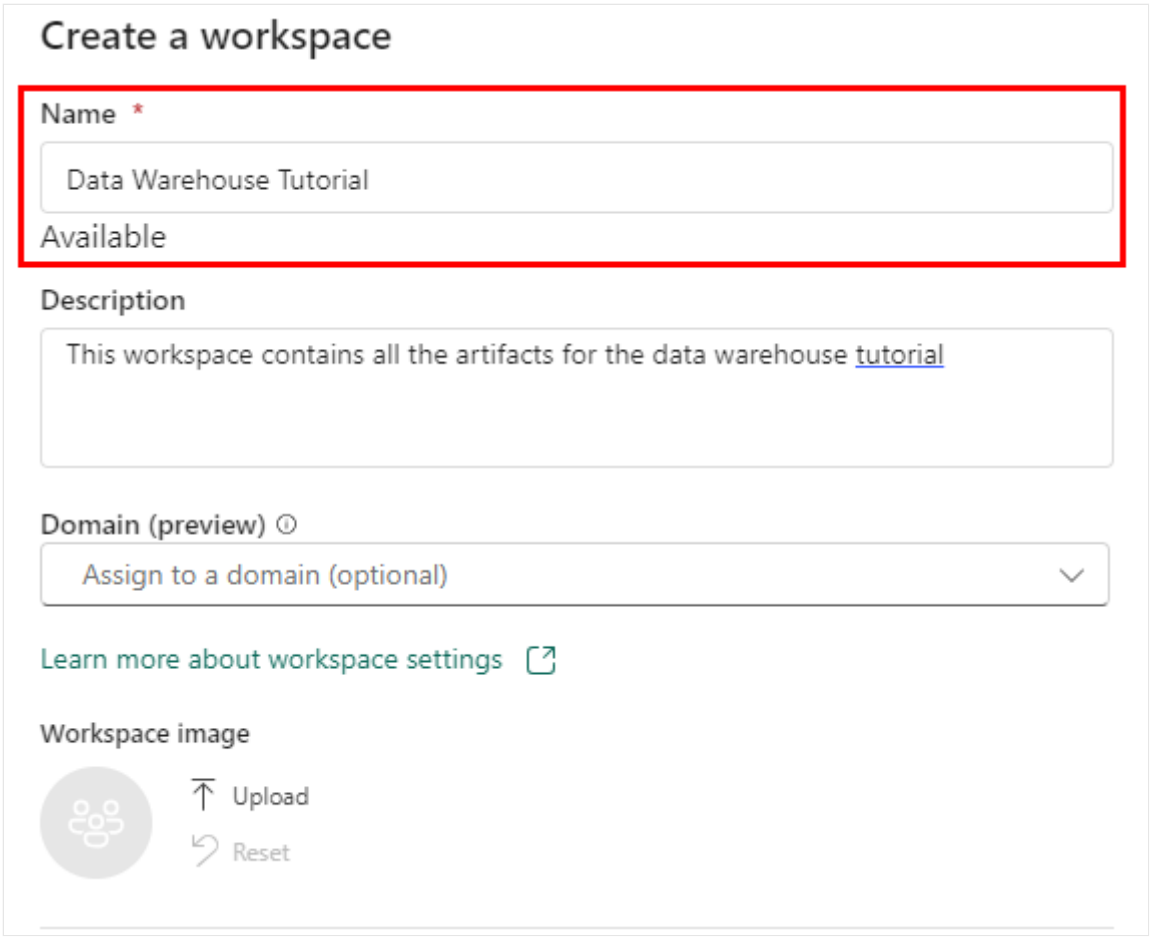
Create a workspace

The workspace contains all the items needed for data warehousing, including: Data Factory pipelines, the data warehouse, Power BI datasets, and reports.

1. Sign in to Power BI.
2. Select **Workspaces** > **New workspace**.



3. Fill out the **Create a workspace** form as follows:
    a. **Name**: Enter `Data Warehouse Tutorial`, and some characters for uniqueness.
    b. **Description**: Optionally, enter a description for the workspace.



4. Expand the **Advanced** section.

5. Choose **Fabric capacity** or **Trial** in the **License mode** section.

6. Choose a premium capacity you have access to.

7. Select **Apply**. The workspace is created and opened.

Next steps

Tutorial: Create a Microsoft Fabric data warehouse

Tutorial: Create a Warehouse in Microsoft Fabric

Article • 05/23/2023

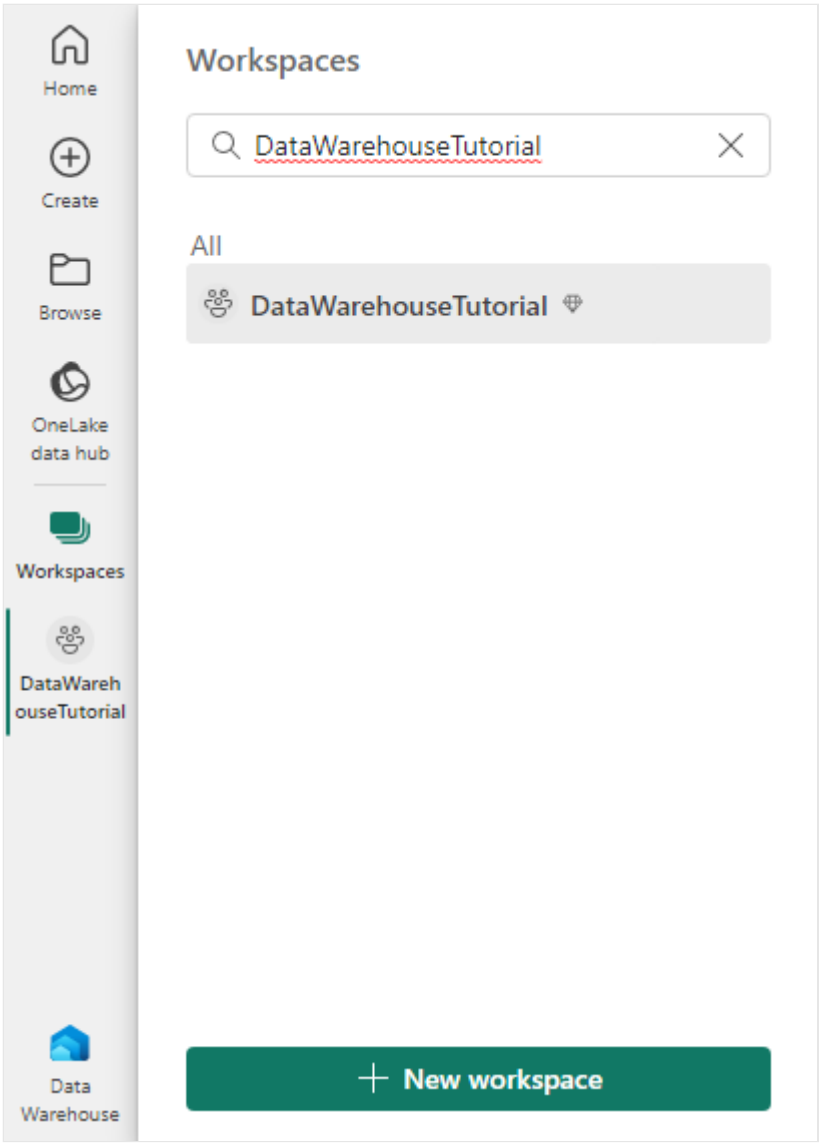
Applies to: SQL Endpoint and Warehouse in Microsoft Fabric

Now that you have a workspace, you can create your first Warehouse in Microsoft Fabric.

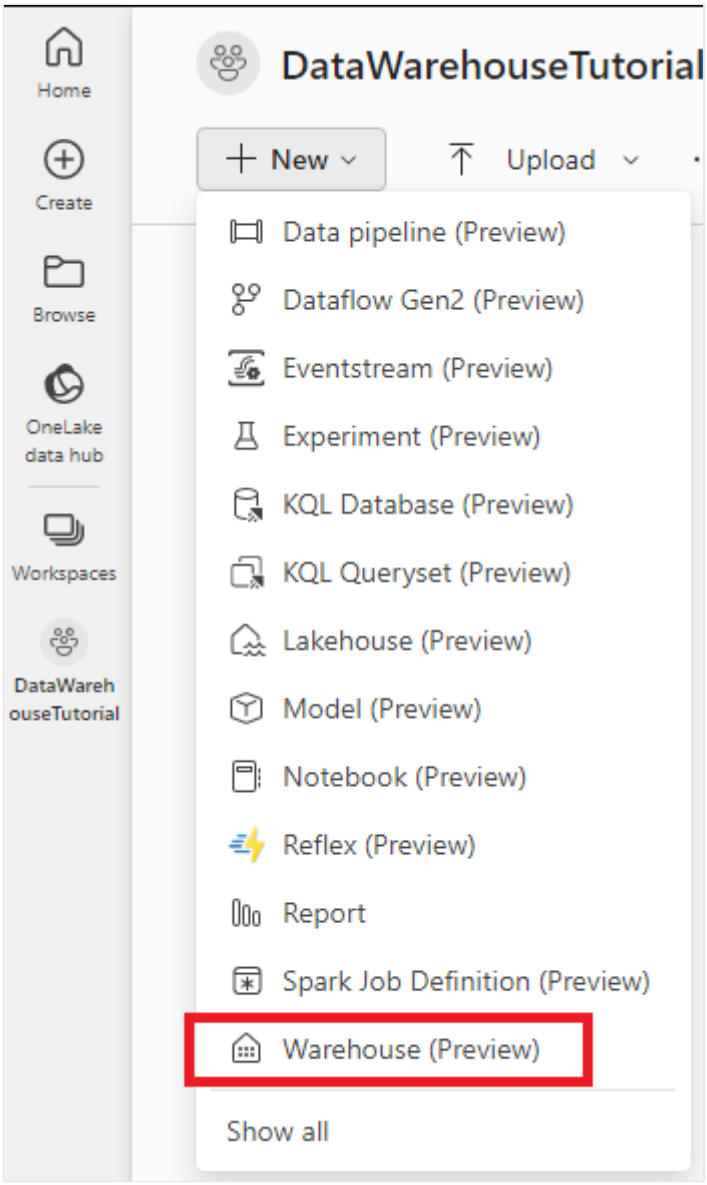
> **Important**
>
> Microsoft Fabric is currently in PREVIEW. This information relates to a prerelease product that may be substantially modified before it's released. Microsoft makes no warranties, expressed or implied, with respect to the information provided here.

Create your first Warehouse

1. Select **Workspaces** in the navigation menu.
2. Search for the workspace you created in Tutorial: Create a Microsoft Fabric workspace by typing in the search textbox at the top and selecting your workspace to open it.



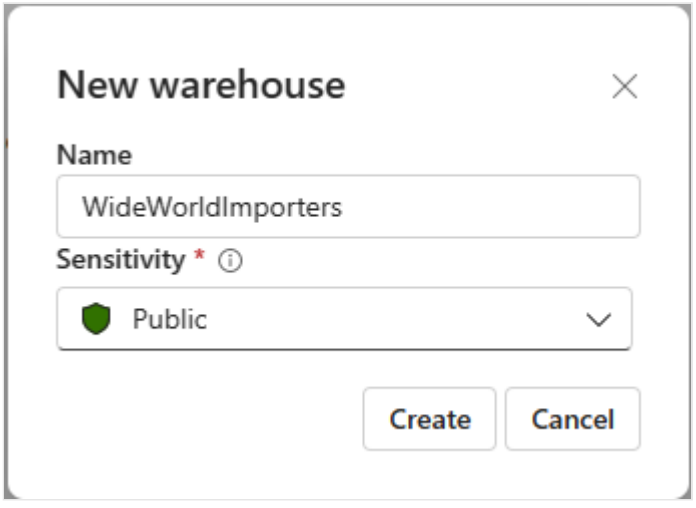
3. Select the + **New** button to display a full list of available items. From the list of objects to create, choose **Warehouse (Preview)** to create a new Warehouse in Microsoft Fabric.



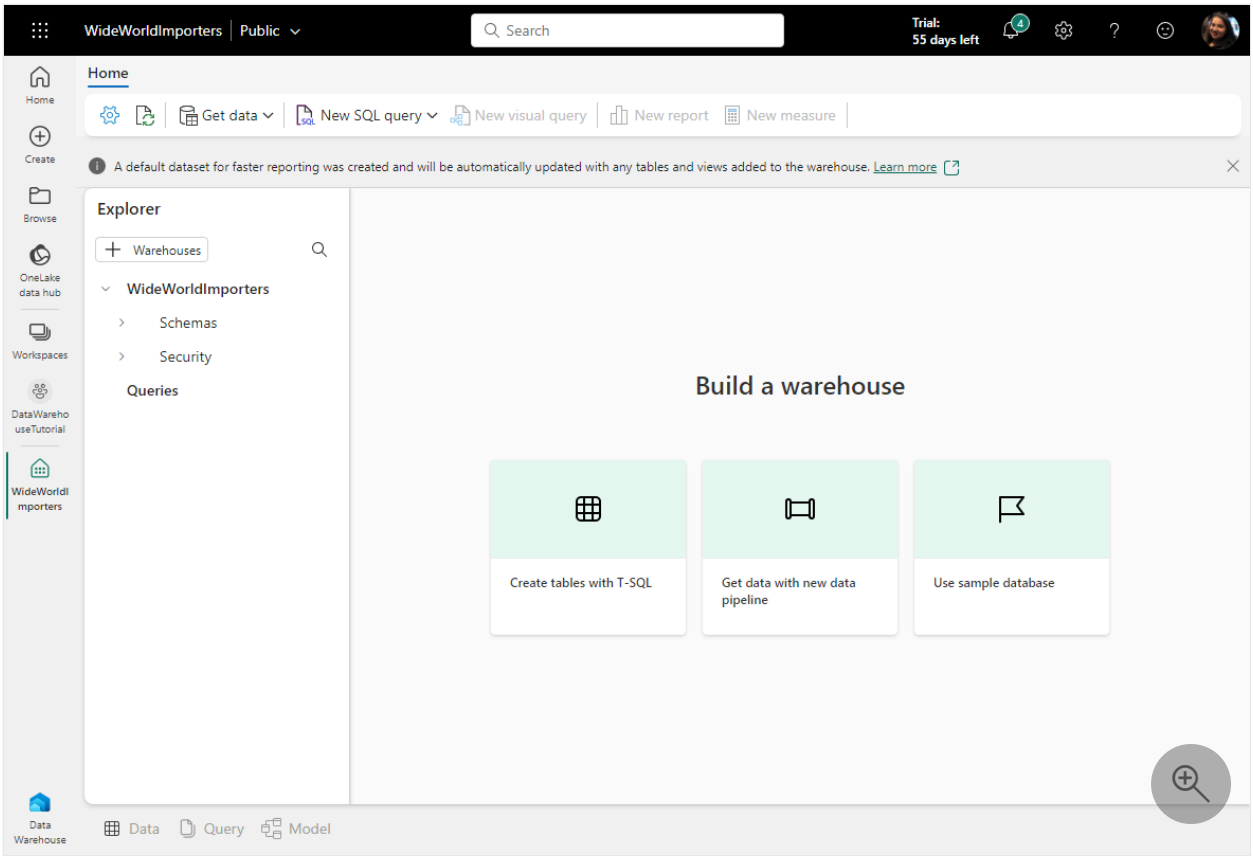
4. On the **New warehouse** dialog, enter `WideWorldImporters` as the name.

5. Set the **Sensitivity** to **Public**.

6. Select **Create**.



When provisioning is complete, the **Build a warehouse** landing page appears.



Next steps

Tutorial: Ingest data into a Microsoft Fabric data warehouse

Tutorial: Ingest data into a Warehouse in Microsoft Fabric

Article • 05/23/2023

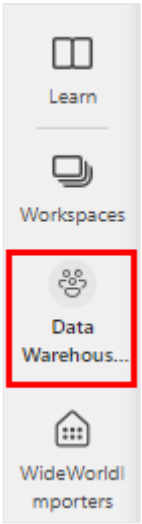
Applies to: SQL Endpoint and Warehouse in Microsoft Fabric

Now that you have created a Warehouse in Microsoft Fabric, you can ingest data into that warehouse.

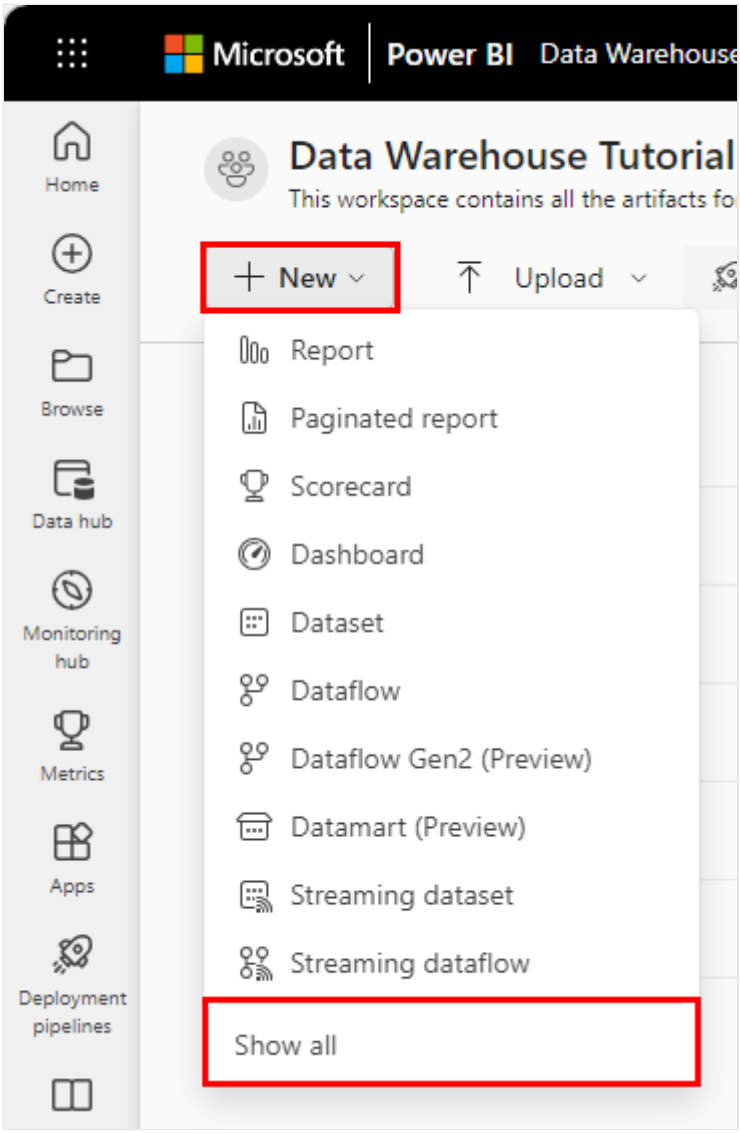
> **Important**
>
> Microsoft Fabric is currently in PREVIEW. This information relates to a prerelease product that may be substantially modified before it's released. Microsoft makes no warranties, expressed or implied, with respect to the information provided here.

Ingest data

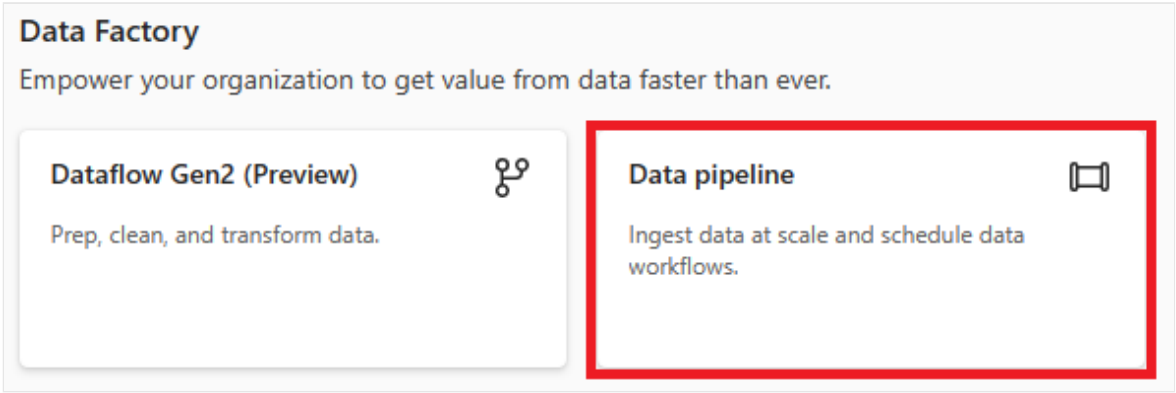
1. From the **Build a warehouse** landing page, select **Data Warehouse Tutorial** in the navigation menu to return to the workspace item list.



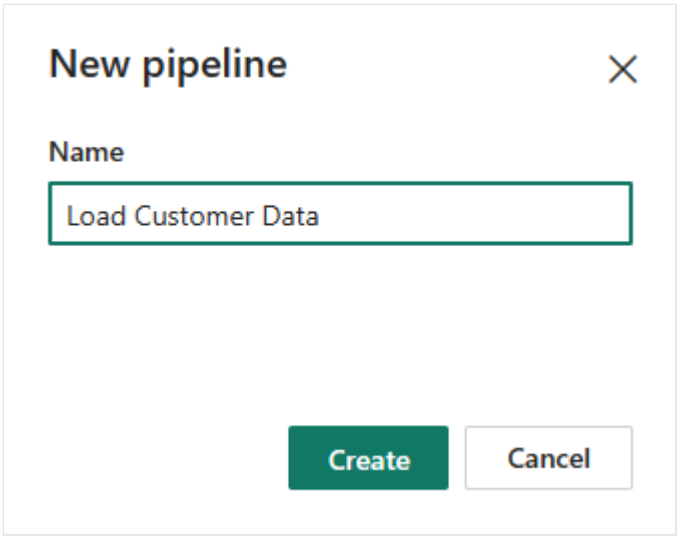
2. In the upper left corner, select **New** > **Show all** to display a full list of available items.



3. In the **Data Factory** section, select **Data pipeline**.

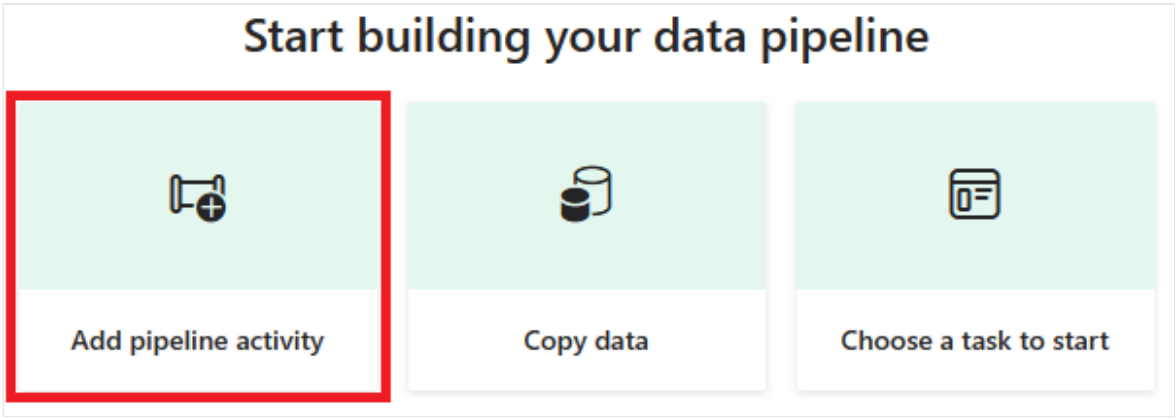


4. On the **New pipeline** dialog, enter `Load Customer Data` as the name.

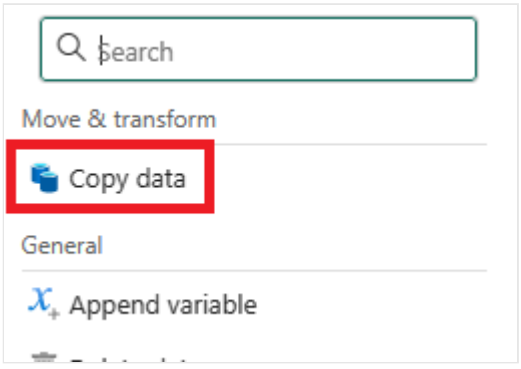


5. Select **Create**.

6. Select **Add pipeline activity** from the **Start building your data pipeline** landing page.

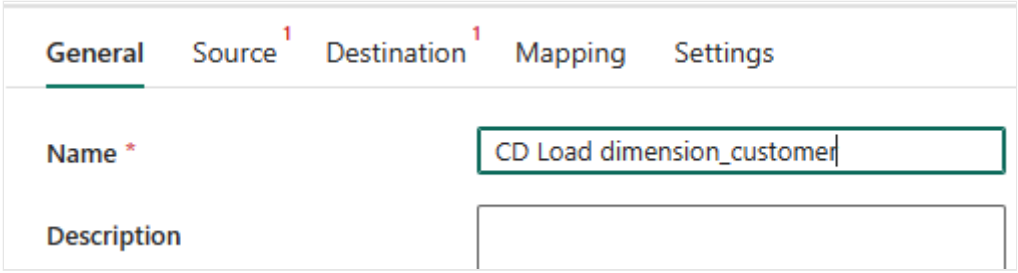


7. Select **Copy data** from the **Move & transform** section.

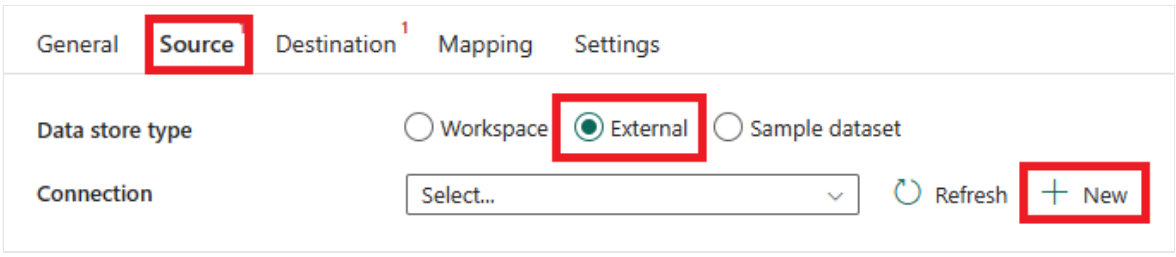


8. If necessary, select the newly created **Copy data** activity from the design canvas and follow the next steps to configure it.

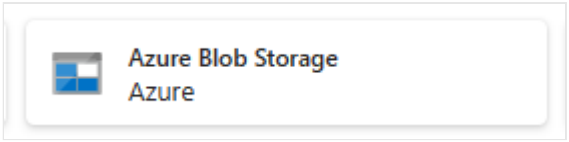
9. On the **General** page, for **Name**, enter `CD Load dimension_customer`.



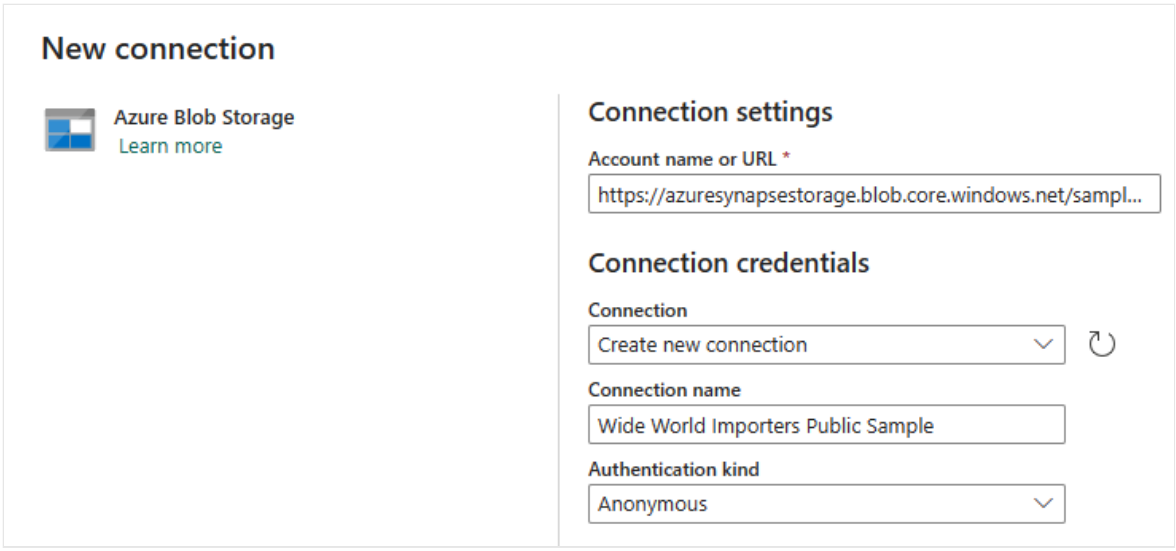
10. On the **Source** page, select **External** for the **Data store type**.
11. Next to the **Connection** box, select **New** to create a new connection.



12. On the **New connection** page, select **Azure Blob Storage** from the list of connection options.



13. Select **Continue**.
14. On the **Connection settings** page, configure the settings as follows:
    a. In the **Account name or URL**, enter `https://azuresynapsestorage.blob.core.windows.net/sampledata/`.
    b. In the **Connection credentials** section, select **Create new connection** in the dropdown for the **Connection**.
    c. For **Connection name**, enter `Wide World Importers Public Sample`.
    d. Set the **Authentication kind** to **Anonymous**.



15. Select **Create**.

16. Change the remaining settings on the **Source** page of the copy activity as follows, to reach the .parquet files in `https://azuresynapsestorage.blob.core.windows.net/sampledata/WideWorldImportersDW/parquet/full/dimension_city/*.parquet`:

    a. In the **File path** text boxes, provide:

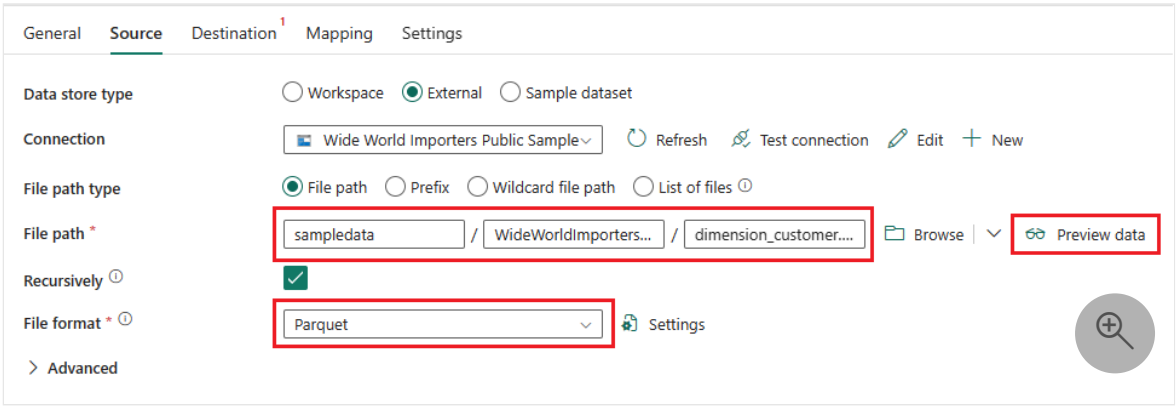
        i. **Container:** `sampledata`

        ii. **File path - Directory:** `WideWorldImportersDW/tables`

        iii. **File path - File name:** `dimension_customer.parquet`

    b. In the **File format** drop down, choose **Parquet**.

17. Select **Preview data** next to the **File path** setting to ensure there are no errors.



18. On the **Destination** page, select **Workspace** for the **Data store type**.

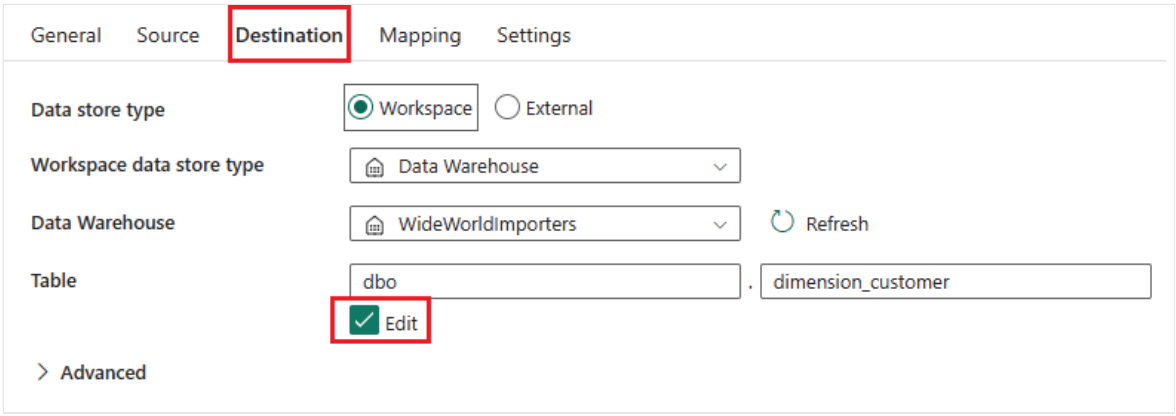
19. Select **Data Warehouse** for the **Workspace data store type**.

20. In the **Data Warehouse** drop down, select **WideWorldImporters** from the list.

21. Next to the **Table** configuration setting, check the box under the dropdown list labeled **Edit**. The dropdown changes to two text boxes.

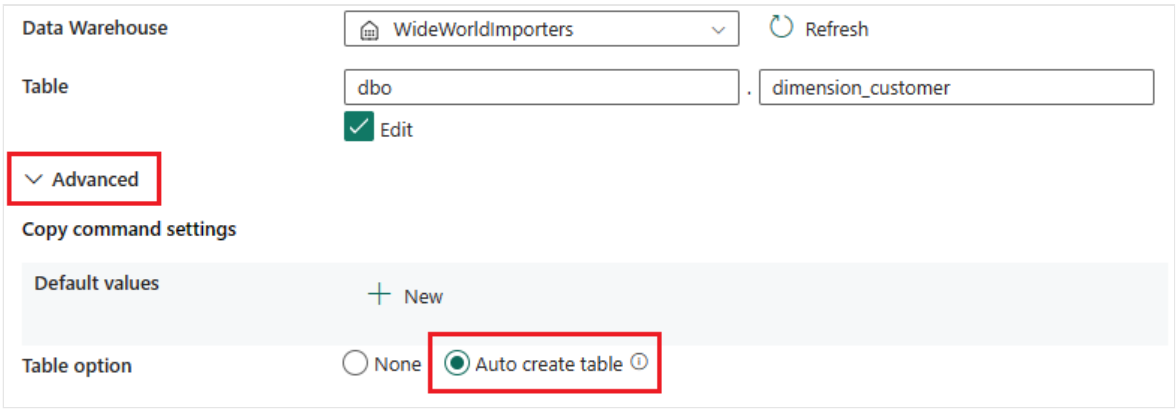
22. In the first box next to the **Table** setting, enter `dbo`.

23. In the second box next to the **Table** setting, enter `dimension_customer`.



24. Expand the **Advanced** section.

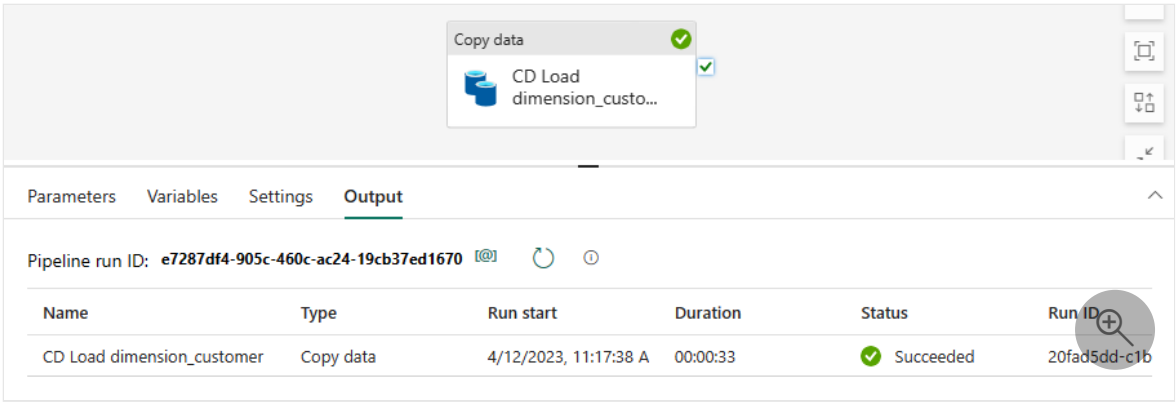
25. For the **Table option**, select **Auto create table**.



26. From the ribbon, select **Run**.

27. Select **Save and run** from the dialog box. The pipeline to load the `dimension_customer` table with start.

28. Monitor the copy activity's progress on the **Output** page and wait for it to complete.



Next steps

Tutorial: Create tables in a data warehouse

Tutorial: Create tables in a data warehouse

Article • 05/23/2023

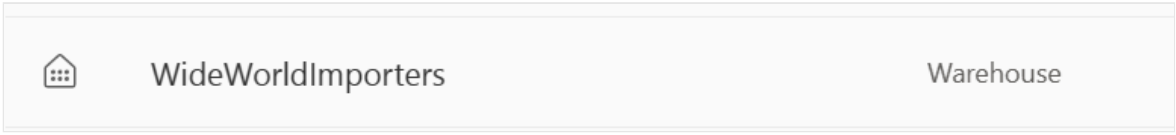
Applies to: SQL Endpoint and Warehouse in Microsoft Fabric

Learn how to create tables in the data warehouse you created in a previous part of the tutorial.

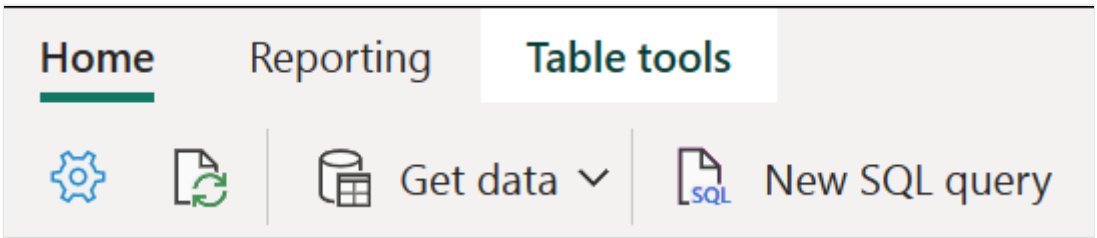
> **Important**
>
> Microsoft Fabric is currently in PREVIEW. This information relates to a prerelease product that may be substantially modified before it's released. Microsoft makes no warranties, expressed or implied, with respect to the information provided here.

Create a table

1. Select **Workspaces** in the navigation menu.
2. Select the workspace created in Tutorial: Create a Microsoft Fabric data workspace, such as **Data Warehouse Tutorial**.
3. From the item list, select `WideWorldImporters` with the type of **Warehouse**.



4. From the ribbon, select **New SQL query**.



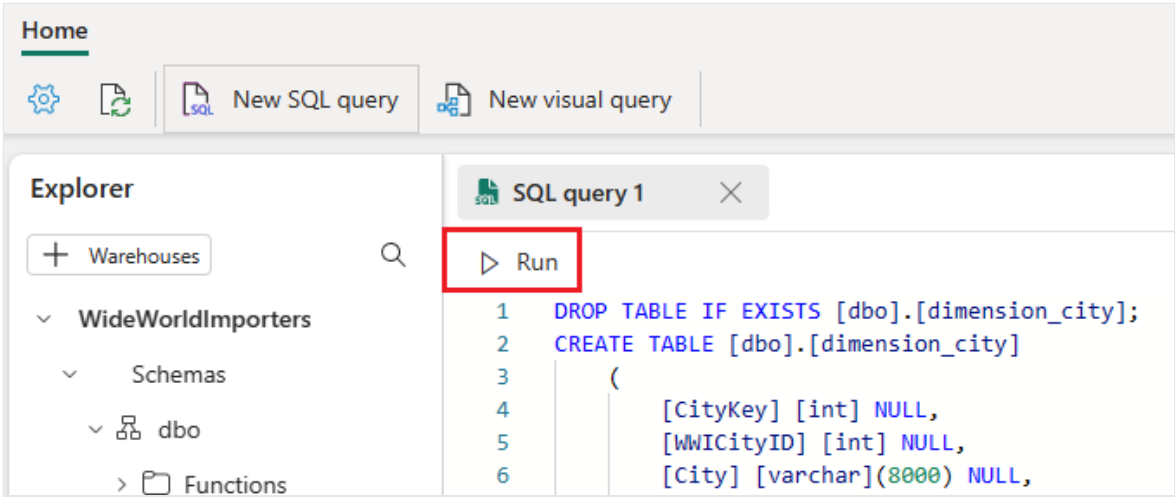
5. In the query editor, paste the following code.

```SQL
/*
1. Drop the dimension_city table if it already exists.
```

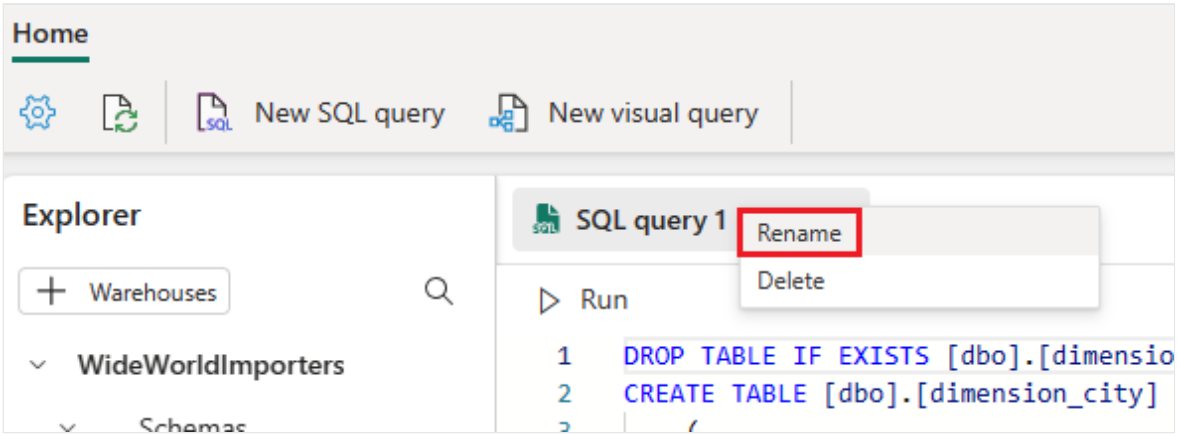
```
2. Create the dimension_city table.
3. Drop the fact_sale table if it already exists.
4. Create the fact_sale table.
*/

--dimension_city
DROP TABLE IF EXISTS [dbo].[dimension_city];
CREATE TABLE [dbo].[dimension_city]
    (
        [CityKey] [int] NULL,
        [WWICityID] [int] NULL,
        [City] [varchar](8000) NULL,
        [StateProvince] [varchar](8000) NULL,
        [Country] [varchar](8000) NULL,
        [Continent] [varchar](8000) NULL,
        [SalesTerritory] [varchar](8000) NULL,
        [Region] [varchar](8000) NULL,
        [Subregion] [varchar](8000) NULL,
        [Location] [varchar](8000) NULL,
        [LatestRecordedPopulation] [bigint] NULL,
        [ValidFrom] [datetime2](6) NULL,
        [ValidTo] [datetime2](6) NULL,
        [LineageKey] [int] NULL
    );

--fact_sale

DROP TABLE IF EXISTS [dbo].[fact_sale];

CREATE TABLE [dbo].[fact_sale]

    (
        [SaleKey] [bigint] NULL,
        [CityKey] [int] NULL,
        [CustomerKey] [int] NULL,
        [BillToCustomerKey] [int] NULL,
        [StockItemKey] [int] NULL,
        [InvoiceDateKey] [datetime2](6) NULL,
        [DeliveryDateKey] [datetime2](6) NULL,
        [SalespersonKey] [int] NULL,
        [WWIInvoiceID] [int] NULL,
        [Description] [varchar](8000) NULL,
        [Package] [varchar](8000) NULL,
        [Quantity] [int] NULL,
        [UnitPrice] [decimal](18, 2) NULL,
        [TaxRate] [decimal](18, 3) NULL,
        [TotalExcludingTax] [decimal](29, 2) NULL,
        [TaxAmount] [decimal](38, 6) NULL,
        [Profit] [decimal](18, 2) NULL,
        [TotalIncludingTax] [decimal](38, 6) NULL,
        [TotalDryItems] [int] NULL,
        [TotalChillerItems] [int] NULL,
        [LineageKey] [int] NULL,
        [Month] [int] NULL,
        [Year] [int] NULL,
```

```
        [Quarter] [int] NULL
    );
```

6. Select **Run** to execute the query.



7. To save this query for reference later, right-click on the query tab just above the editor and select **Rename**.

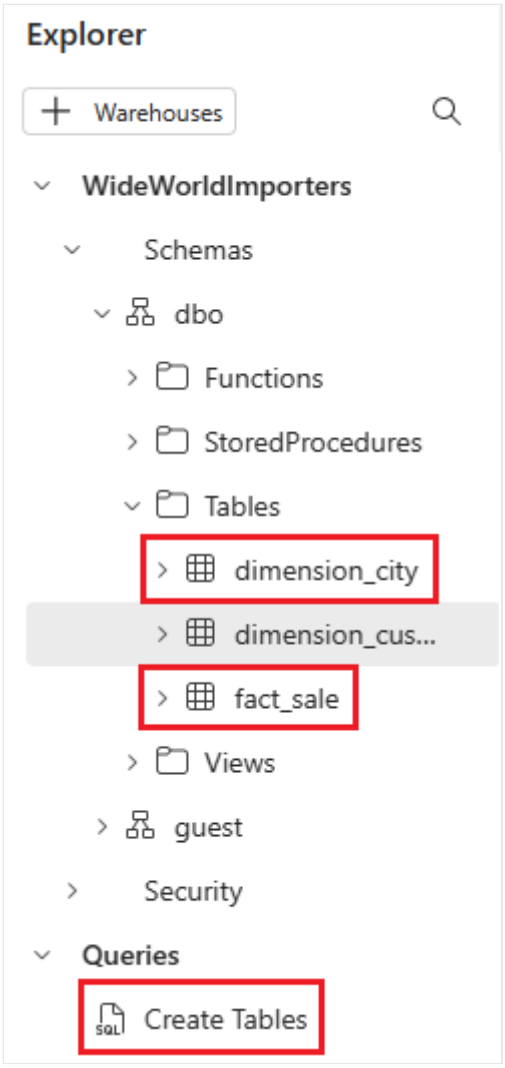


8. Type `Create Tables` to change the name of the query.

9. Press **Enter** on the keyboard or select anywhere outside the tab to save the change.

10. Validate the table was created successfully by selecting the **refresh** button on the ribbon.

11. In the **Object explorer**, verify that you can see the newly created **Create Tables** query, `fact_sale` table, and `dimension_city` table.



Next steps

Tutorial: Load data using T-SQL

Tutorial: Load data using T-SQL

Article • 05/23/2023

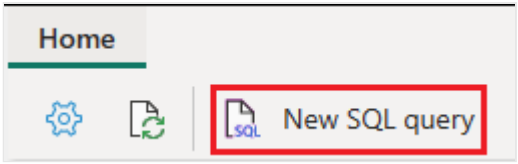
Applies to: SQL Endpoint and Warehouse in Microsoft Fabric

Now that you know how to build a data warehouse, load a table, and generate a report, it's time to extend the solution by exploring other methods for loading data.

> **Important**
>
> Microsoft Fabric is currently in PREVIEW. This information relates to a prerelease product that may be substantially modified before it's released. Microsoft makes no warranties, expressed or implied, with respect to the information provided here.

Load data with COPY INTO

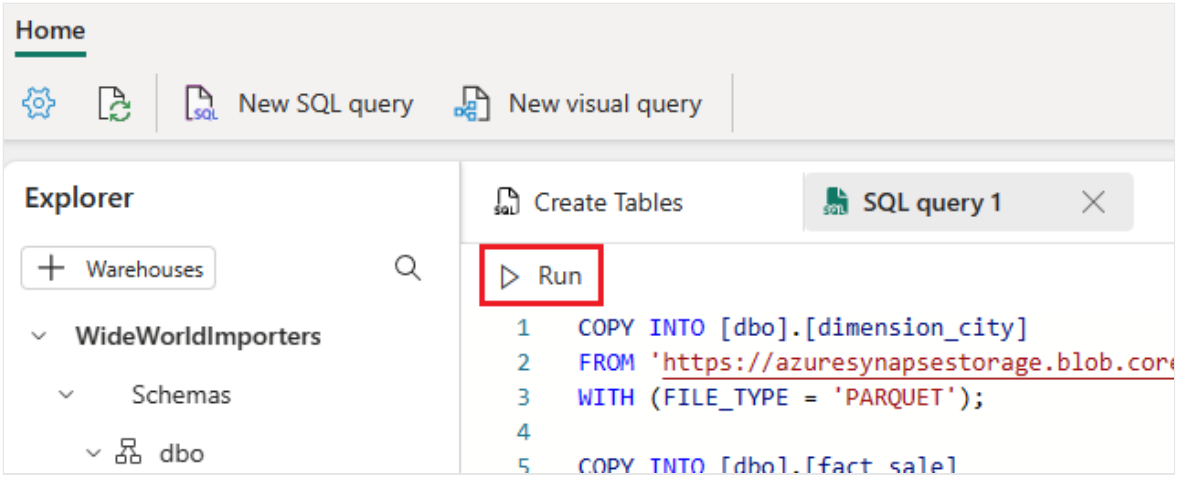
1. From the ribbon, select **New SQL query**.



2. In the query editor, paste the following code.

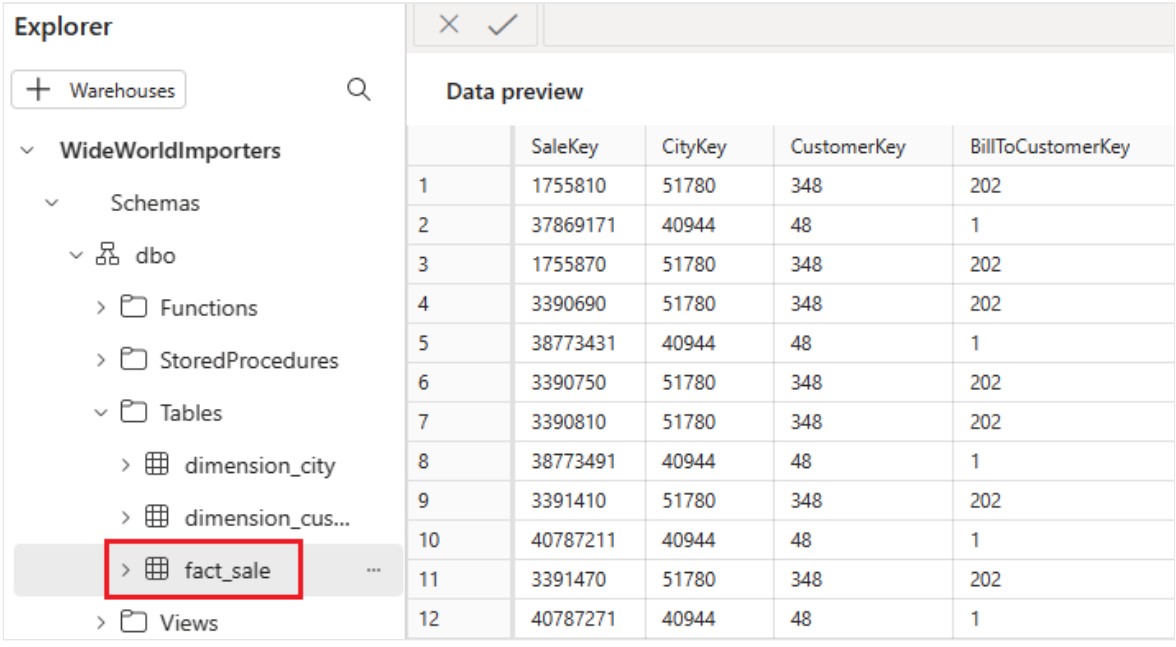
```SQL
--Copy data from the public Azure storage account to the
dbo.dimension_city table.
COPY INTO [dbo].[dimension_city]
FROM
'https://azuresynapsestorage.blob.core.windows.net/sampledata/WideWorld
ImportersDW/tables/dimension_city.parquet'
WITH (FILE_TYPE = 'PARQUET');

--Copy data from the public Azure storage account to the dbo.fact_sale
table.
COPY INTO [dbo].[fact_sale]
FROM
'https://azuresynapsestorage.blob.core.windows.net/sampledata/WideWorld
ImportersDW/tables/fact_sale.parquet'
WITH (FILE_TYPE = 'PARQUET');
```

3. Select **Run** to execute the query. The query takes between one and four minutes to execute.



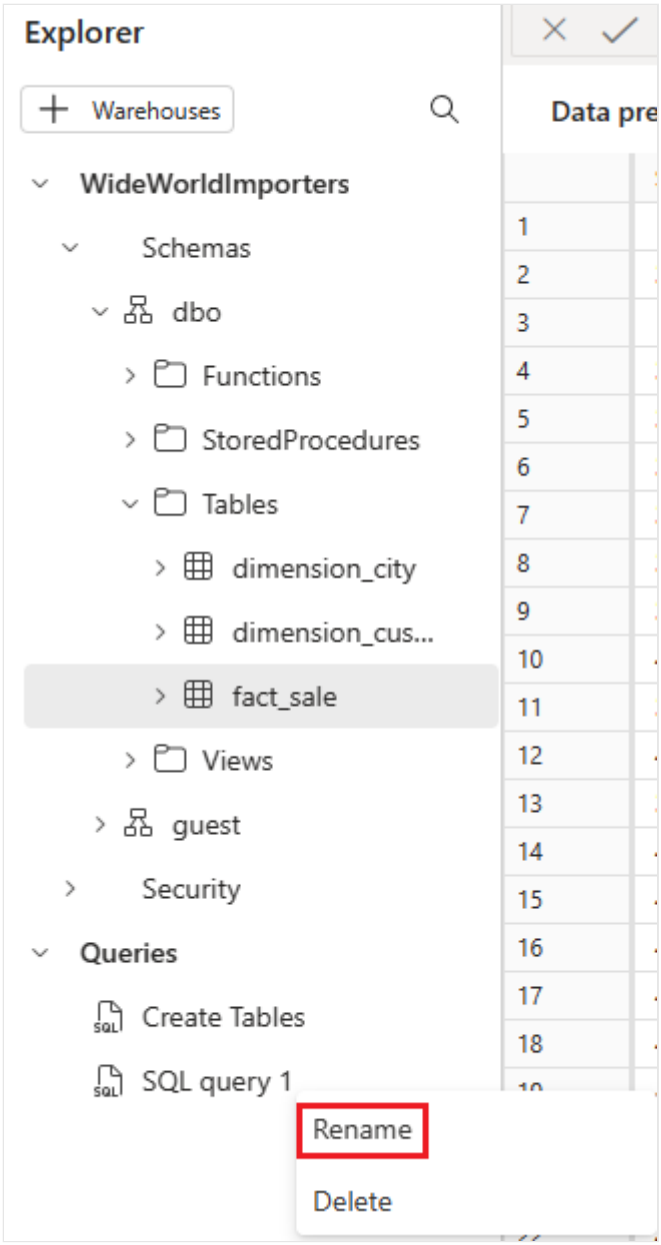
4. After the query is completed, review the messages to see the rows affected which indicated the number of rows that were loaded into the `dimension_city` and `fact_sale` tables respectively.

5. Load the data preview to validate the data loaded successfully by selecting on the `fact_sale` table in the **Explorer**.



	SaleKey	CityKey	CustomerKey	BillToCustomerKey
1	1755810	51780	348	202
2	37869171	40944	48	1
3	1755870	51780	348	202
4	3390690	51780	348	202
5	38773431	40944	48	1
6	3390750	51780	348	202
7	3390810	51780	348	202
8	38773491	40944	48	1
9	3391410	51780	348	202
10	40787211	40944	48	1
11	3391470	51780	348	202
12	40787271	40944	48	1

6. Rename the query for reference later. Right-click on **SQL query 1** in the **Explorer** and select **Rename**.



7. Type `Load Tables` to change the name of the query.

8. Press **Enter** on the keyboard or select anywhere outside the tab to save the change.

Next steps

Tutorial: Transform data using a stored procedure

Tutorial: Transform data using a stored procedure

Article • 05/23/2023

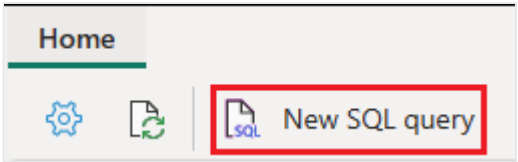
Applies to: SQL Endpoint and Warehouse in Microsoft Fabric

Learn how to create and save a new stored procedure to transform data.

> **Important**
>
> Microsoft Fabric is currently in PREVIEW. This information relates to a prerelease product that may be substantially modified before it's released. Microsoft makes no warranties, expressed or implied, with respect to the information provided here.

Transform data

1. From the **Home** tab of the ribbon, select **New SQL query**.

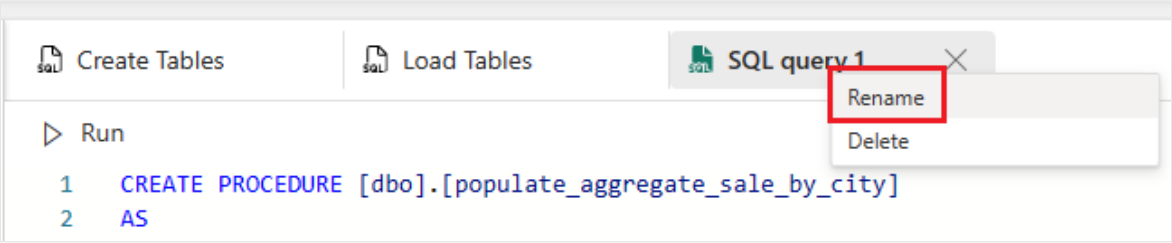


2. In the query editor, paste the following code to create the stored procedure `dbo.populate_aggregate_sale_by_city`. This stored procedure will create and load the `dbo.aggregate_sale_by_date_city` table in a later step.

```
SQL

--Drop the stored procedure if it already exists.
DROP PROCEDURE IF EXISTS [dbo].[populate_aggregate_sale_by_city]
GO

--Create the populate_aggregate_sale_by_city stored procedure.
CREATE PROCEDURE [dbo].[populate_aggregate_sale_by_city]
AS
BEGIN
    --If the aggregate table already exists, drop it. Then create the
table.
    DROP TABLE IF EXISTS [dbo].[aggregate_sale_by_date_city];
    CREATE TABLE [dbo].[aggregate_sale_by_date_city]
        (
            [Date] [DATETIME2](6),
            [City] [VARCHAR](8000),
```

```
            [StateProvince] [VARCHAR](8000),
            [SalesTerritory] [VARCHAR](8000),
            [SumOfTotalExcludingTax] [DECIMAL](38,2),
            [SumOfTaxAmount] [DECIMAL](38,6),
            [SumOfTotalIncludingTax] [DECIMAL](38,6),
            [SumOfProfit] [DECIMAL](38,2)
        );

    --Reload the aggregated dataset to the table.
    INSERT INTO [dbo].[aggregate_sale_by_date_city]
    SELECT
        FS.[InvoiceDateKey] AS [Date],
        DC.[City],
        DC.[StateProvince],
        DC.[SalesTerritory],
        SUM(FS.[TotalExcludingTax]) AS [SumOfTotalExcludingTax],
        SUM(FS.[TaxAmount]) AS [SumOfTaxAmount],
        SUM(FS.[TotalIncludingTax]) AS [SumOfTotalIncludingTax],
        SUM(FS.[Profit]) AS [SumOfProfit]
    FROM [dbo].[fact_sale] AS FS
    INNER JOIN [dbo].[dimension_city] AS DC
        ON FS.[CityKey] = DC.[CityKey]
    GROUP BY
        FS.[InvoiceDateKey],
        DC.[City],
        DC.[StateProvince],
        DC.[SalesTerritory]
    ORDER BY
        FS.[InvoiceDateKey],
        DC.[StateProvince],
        DC.[City];
END
```

3. To save this query for reference later, right-click on the query tab just above the editor and select **Rename**.



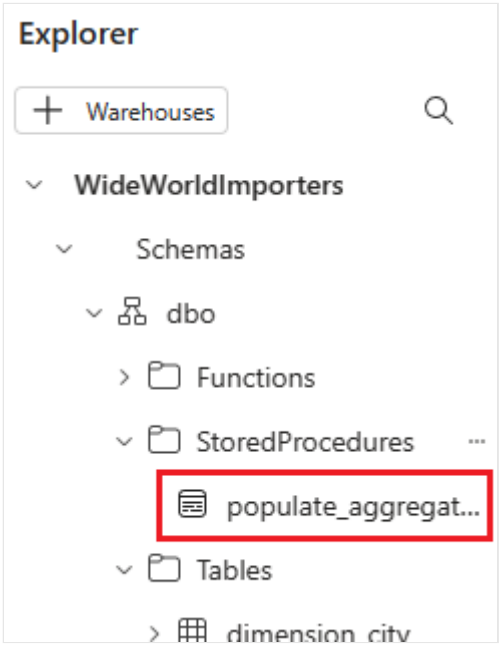
4. Type **Create Aggregate Procedure** to change the name of the query.

5. Press **Enter** on the keyboard or select anywhere outside the tab to save the change.

6. Select **Run** to execute the query.

7. Select the **refresh** button on the ribbon.

8. In the **Object explorer**, verify that you can see the newly created stored procedure by expanding the **StoredProcedures** node under the `dbo` schema.



9. From the **Home** tab of the ribbon, select **New SQL query**.

10. In the query editor, paste the following code. This T-SQL executes `dbo.populate_aggregate_sale_by_city` to create the `dbo.aggregate_sale_by_date_city` table.

```SQL
--Execute the stored procedure to create the aggregate table.
EXEC [dbo].[populate_aggregate_sale_by_city];
```

11. To save this query for reference later, right-click on the query tab just above the editor and select **Rename**.

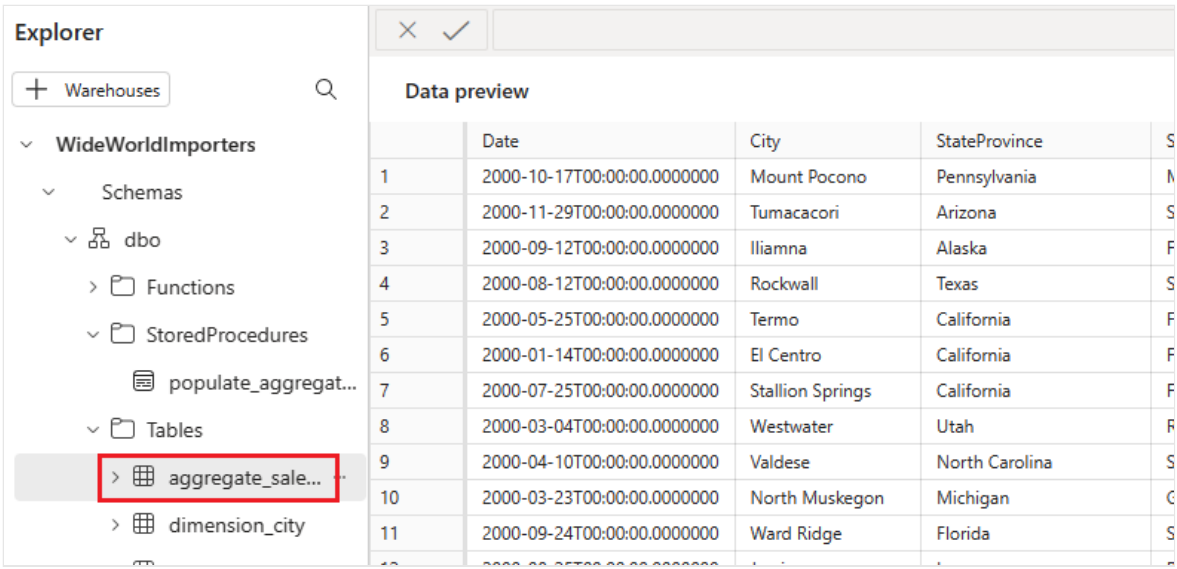
12. Type **Run Create Aggregate Procedure** to change the name of the query.

13. Press **Enter** on the keyboard or select anywhere outside the tab to save the change.

14. Select **Run** to execute the query.

15. Select the **refresh** button on the ribbon. The query takes between two and three minutes to execute.

16. In the **Object explorer**, load the data preview to validate the data loaded successfully by selecting on the `aggregate_sale_by_city` table in the **Explorer**.



Next steps

Tutorial: Create a query with the visual query builder

Tutorial: Create a query with the visual query builder

Article • 05/23/2023

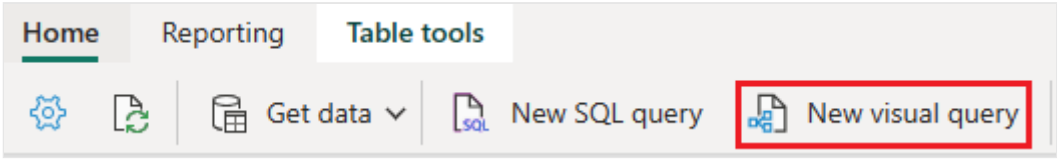
Applies to: SQL Endpoint and Warehouse in Microsoft Fabric

Create and save a query with the visual query builder in the Microsoft Fabric portal.

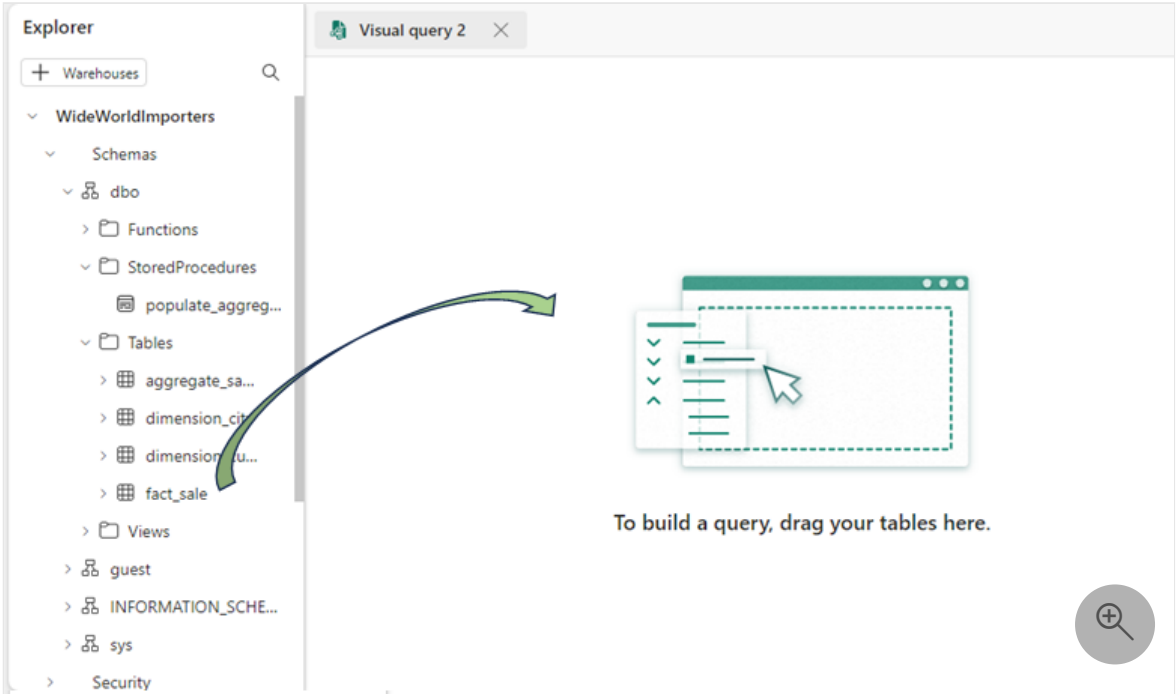
> **Important**
>
> Microsoft Fabric is currently in PREVIEW. This information relates to a prerelease product that may be substantially modified before it's released. Microsoft makes no warranties, expressed or implied, with respect to the information provided here.

Use the visual query builder

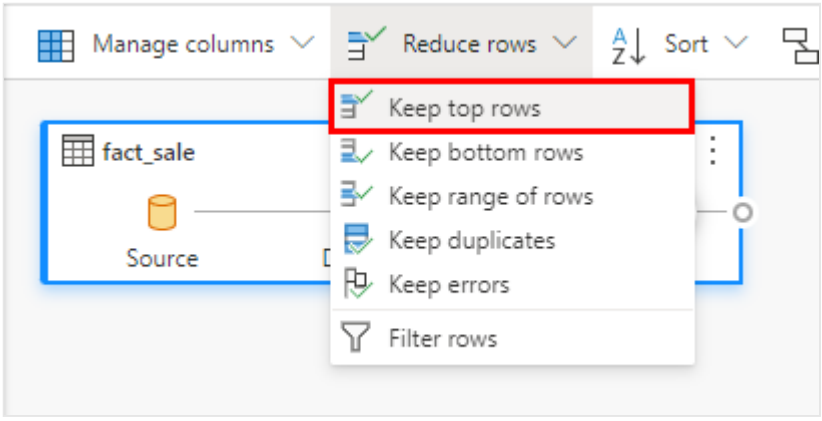
1. From the **Home** tab of the ribbon, select **New visual query**.



2. Drag the `fact_sale` table from the **Explorer** to the query design pane.



3. Limit the dataset size by selecting **Reduce rows** > **Keep top rows** from the transformations ribbon.

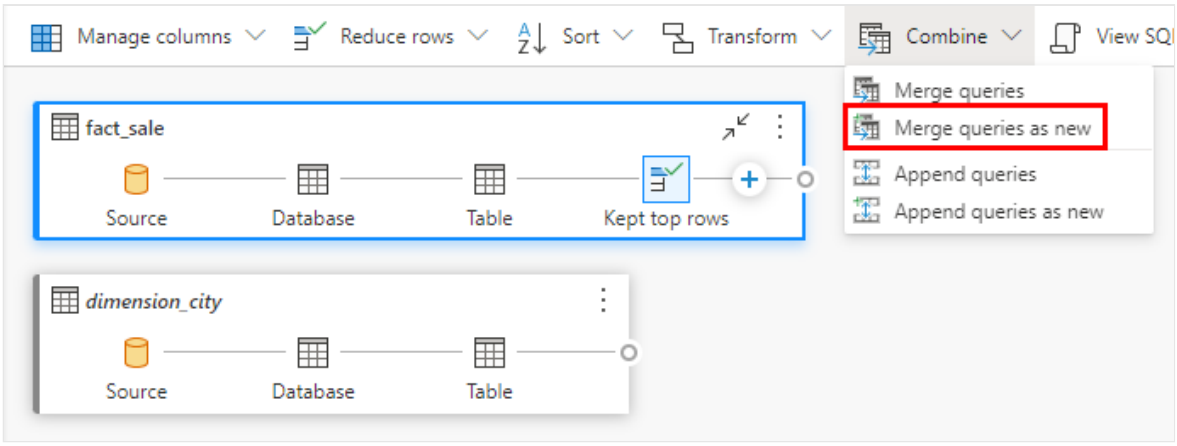


4. In the **Keep top rows** dialog, enter `10000`.

5. Select **OK**.

6. Drag the `dimension_city` table from the explorer to the query design pane.

7. From the transformations ribbon, select the dropdown next to **Combine** and select **Merge queries as new**.



8. On the **Merge** settings page:

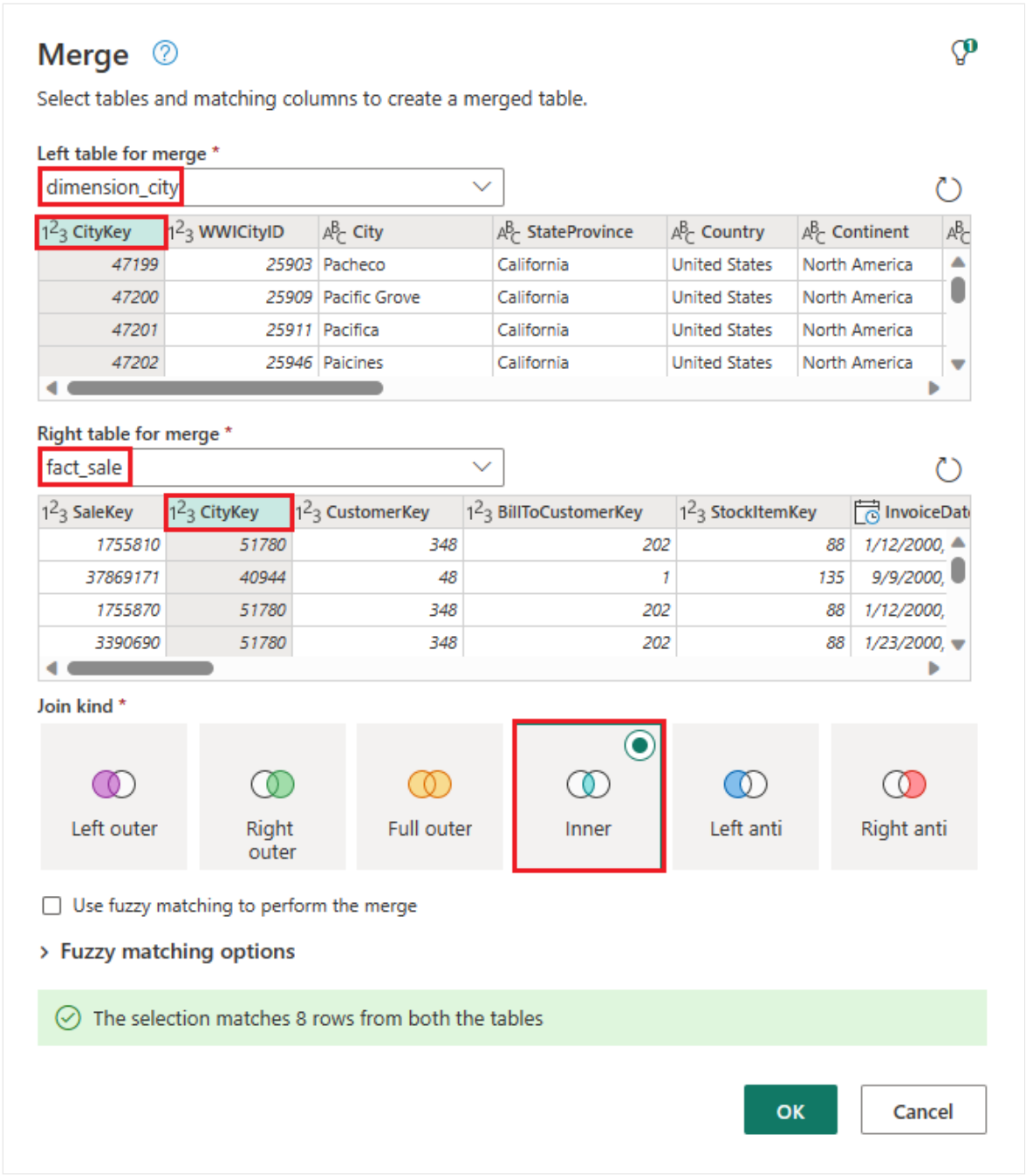
   a. In the **Left table for merge** dropdown, choose `dimension_city`

   b. In the **Right table for merge** dropdown, choose `fact_sale`

   c. Select the `CityKey` field in the `dimension_city` table by selecting on the column name in the header row to indicate the join column.

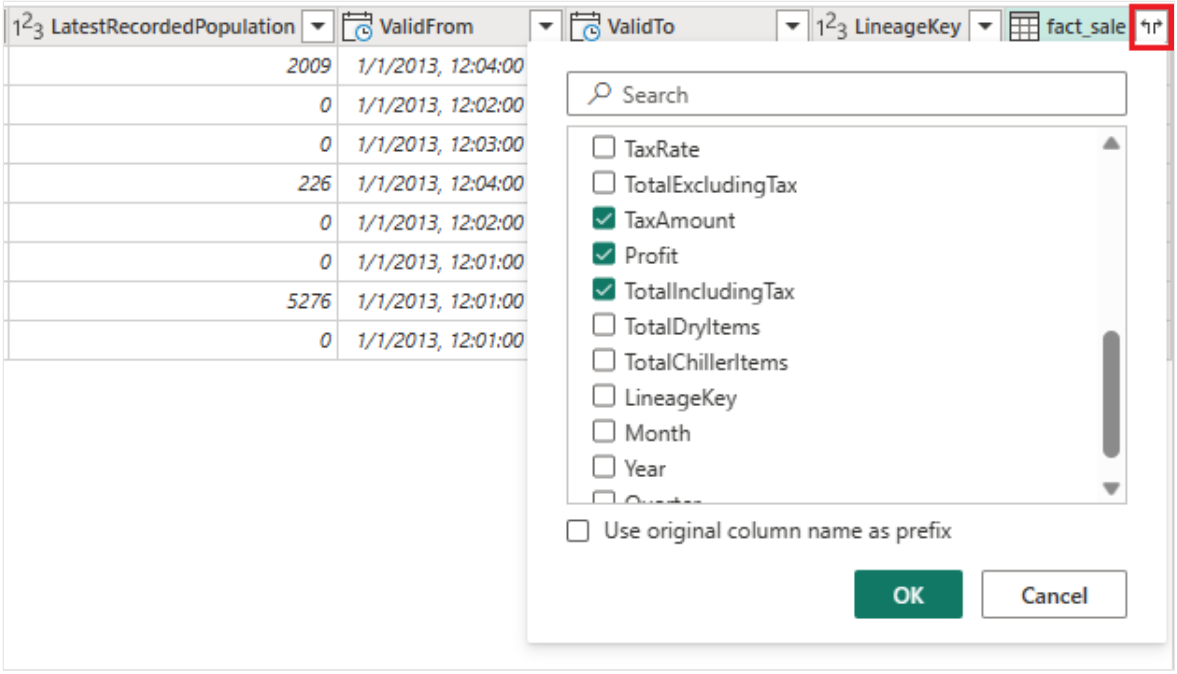
   d. Select the `CityKey` field in the `fact_sale` table by selecting on the column name in the header row to indicate the join column.

   e. In the **Join kind** diagram selection, choose **Inner**.



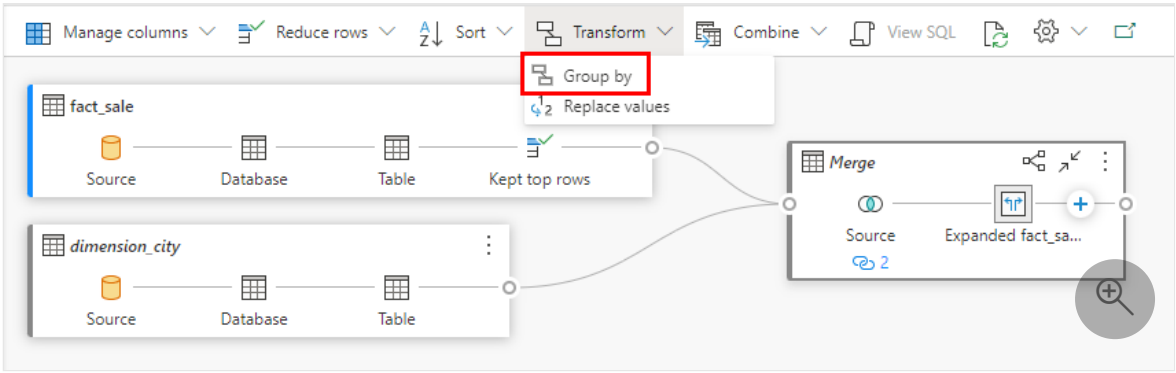
9. Select **OK**.

10. With the **Merge** step selected, select the **Expand** button next to `fact_sale` on the header of the data grid then select the columns `TaxAmount`, `Profit`, and `TotalIncludingTax`.



11. Select **OK**.

12. Select **Transform** > **Group by** from the transformations ribbon.



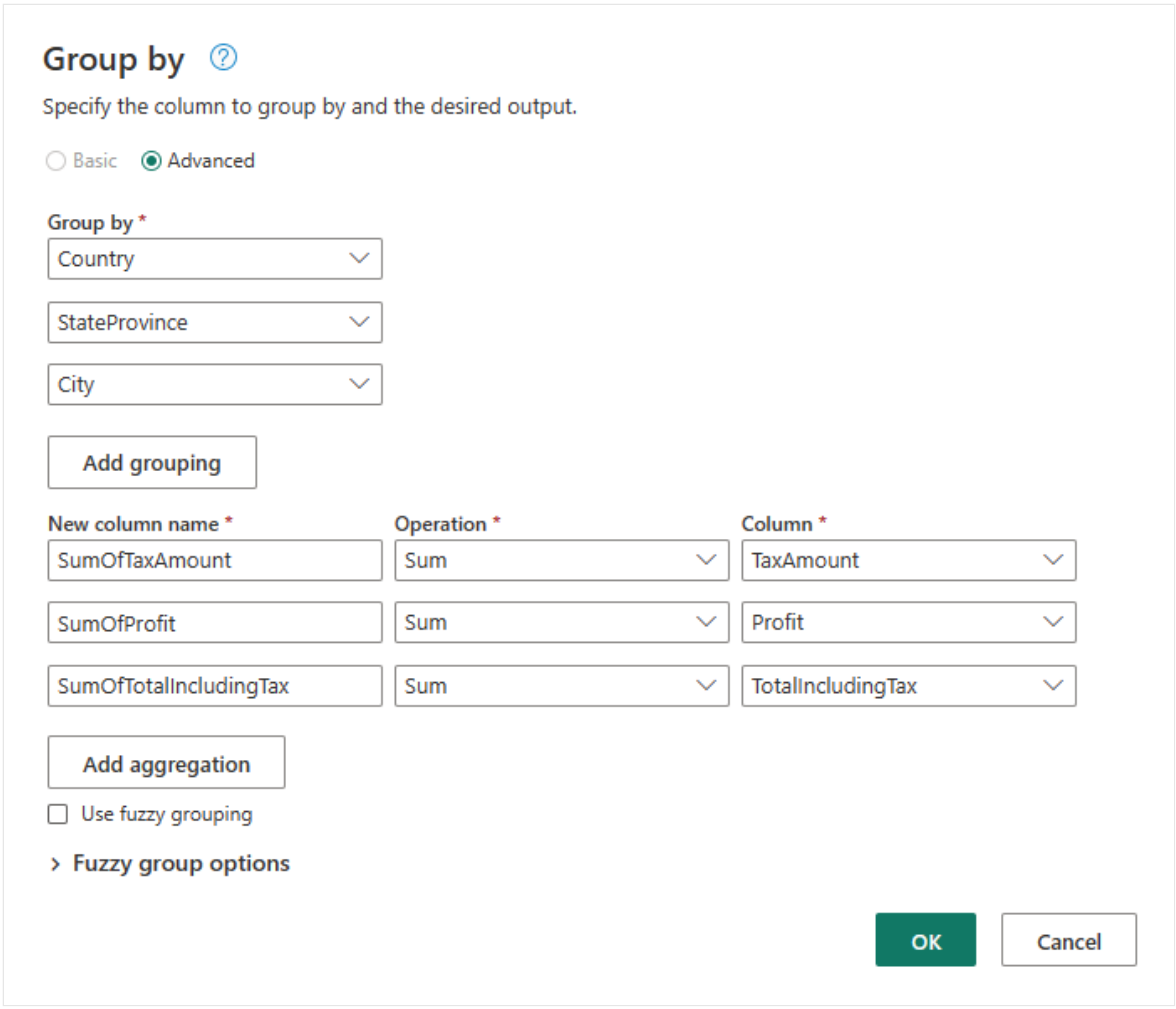
13. On the **Group by** settings page:

    a. Change to **Advanced**.

    b. **Group by** (if necessary, select **Add grouping** to add more group by columns):
        i. `Country`
        ii. `StateProvince`
        iii. `City`

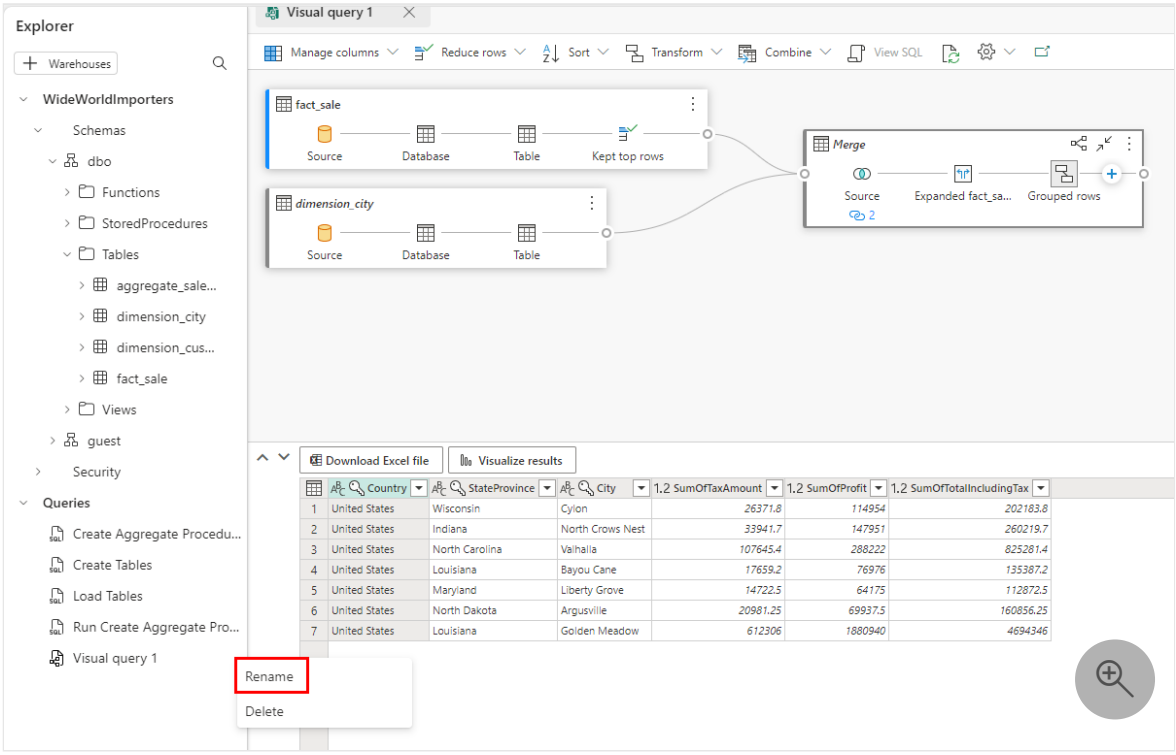
    c. **New column name** (if necessary, select **Add aggregation** to add more aggregate columns and operations):
        i. `SumOfTaxAmount`
            i. Choose **Operation** of **Sum** and **Column** of `TaxAmount`.
        ii. `SumOfProfit`
            i. Choose **Operation** of **Sum** and **Column** of `Profit`.
        iii. `SumOfTotalIncludingTax`

i. Choose **Operation** of **Sum** and **Column** of `TotalIncludingTax`.



14. Select **OK**.

15. Right-click on **Visual query 1** in the **Explorer** and select **Rename**.



16. Type `Sales Summary` to change the name of the query.

17. Press **Enter** on the keyboard or select anywhere outside the tab to save the change.

Next steps

Tutorial: Analyze data with a notebook

Tutorial: Analyze data with a notebook

Article • 05/23/2023

Applies to: ✅ SQL Endpoint and Warehouse in Microsoft Fabric

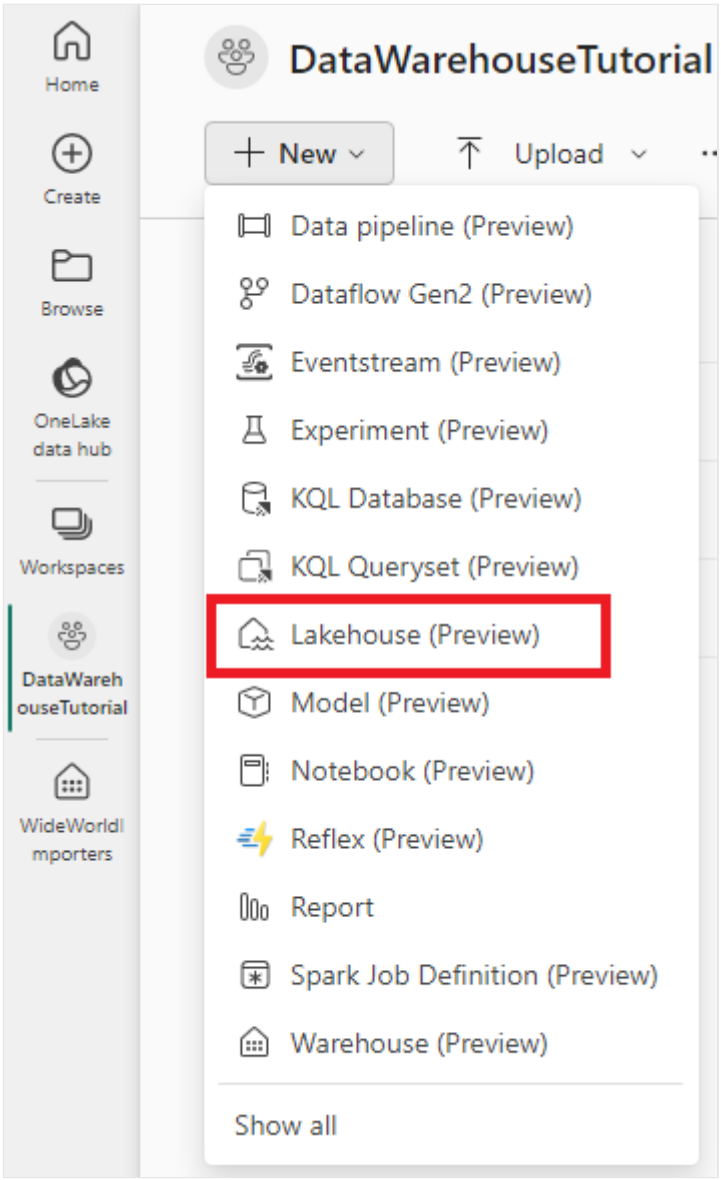
In this tutorial, learn about how you can save your data once and then use it with many other services. Shortcuts can also be created to data stored in Azure Data Lake Storage and S3 to enable you to directly access delta tables from external systems.

> **ⓘ Important**
>
> Microsoft Fabric is currently in PREVIEW. This information relates to a prerelease product that may be substantially modified before it's released. Microsoft makes no warranties, expressed or implied, with respect to the information provided here.

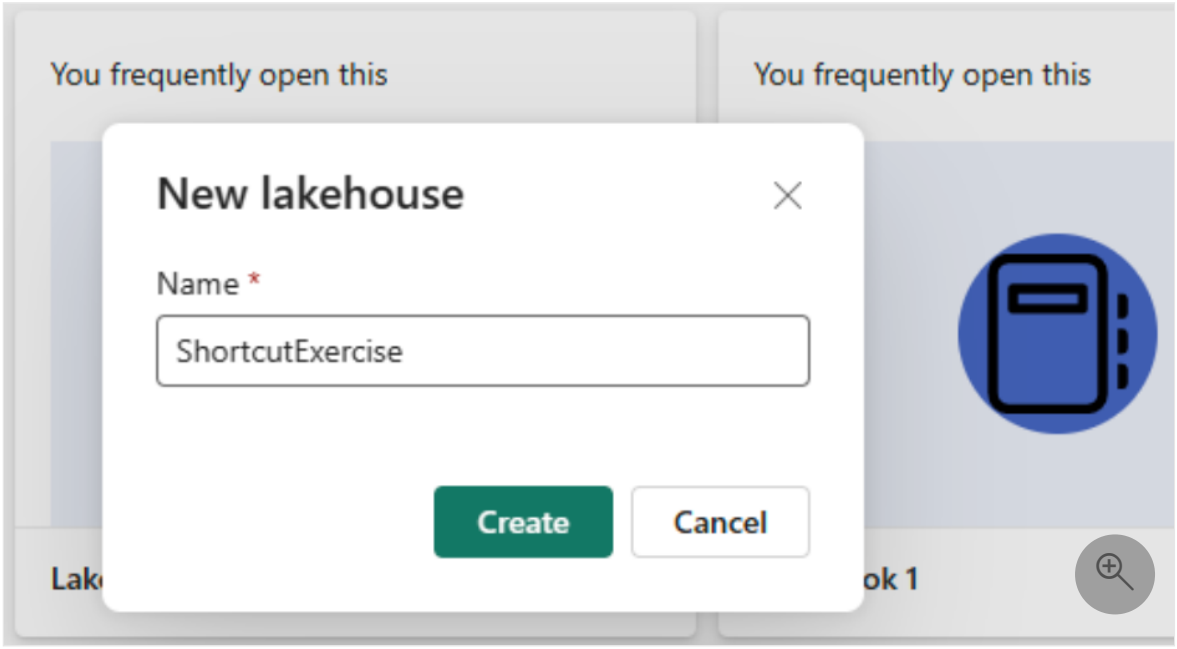
Create a lakehouse

First, we create a new lakehouse. To create a new lakehouse in your Microsoft Fabric workspace:

1. Select the `Data Warehouse Tutorial` workspace in the navigation menu.
2. Select **+ New** > **Lakehouse (Preview)**.

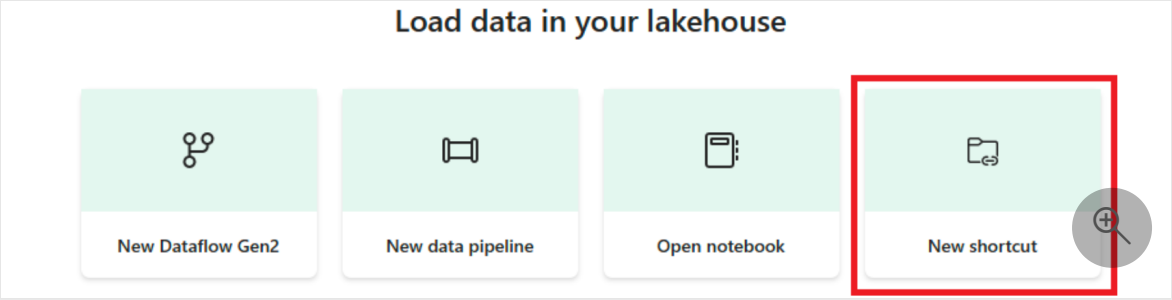


3. In the **Name** field, enter `ShortcutExercise` and select **Create**.

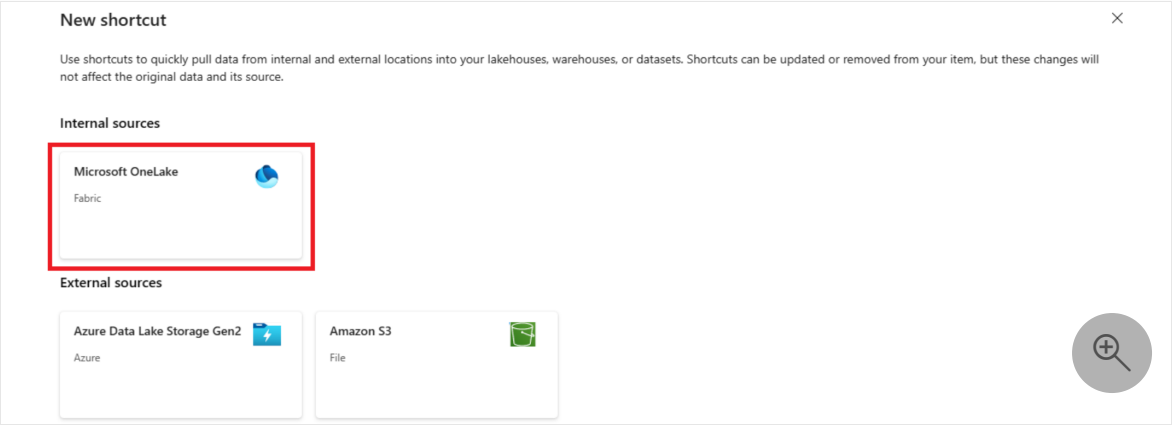


4. The new lakehouse loads and the **Explorer** view opens up, with the **Get data in your lakehouse** menu. Under **Load data in your lakehouse**, select the **New**

shortcut button.

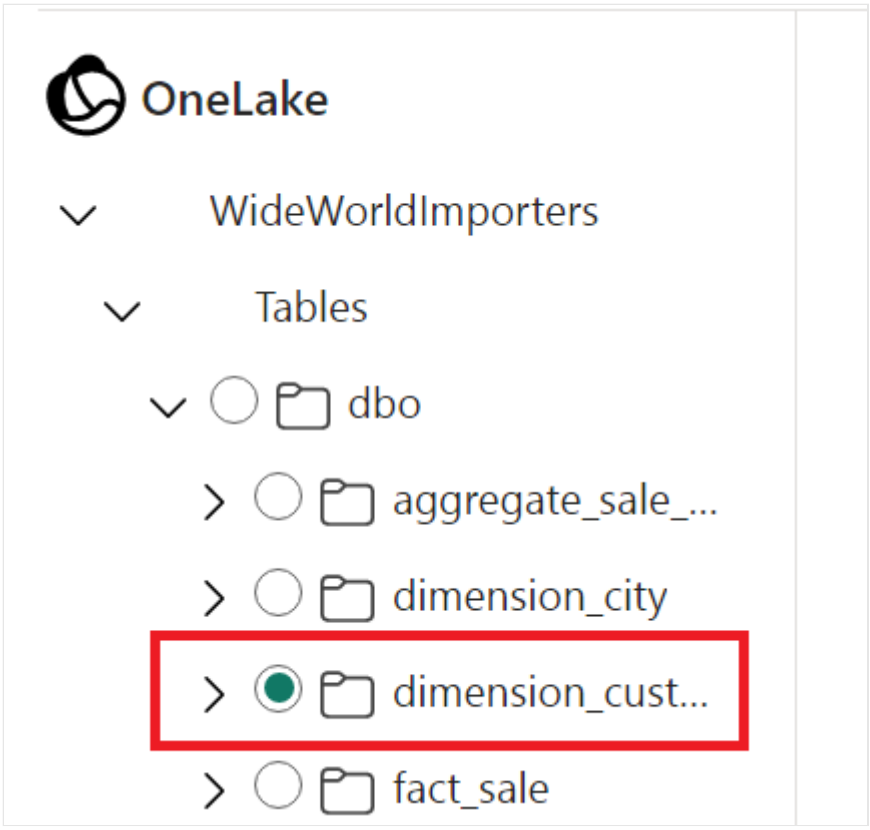


5. In the **New shortcut** window, select the button for **Microsoft OneLake**.

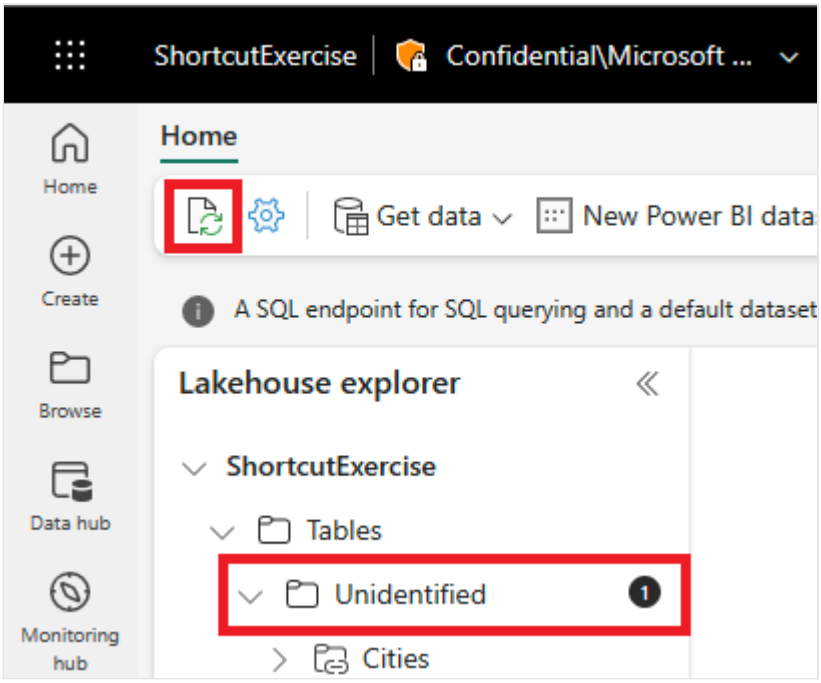


6. In the **Select a data source type** window, scroll through the list until you find the **Warehouse** named `WideWorldImporters` you created previously. Select it, then select **Next**.

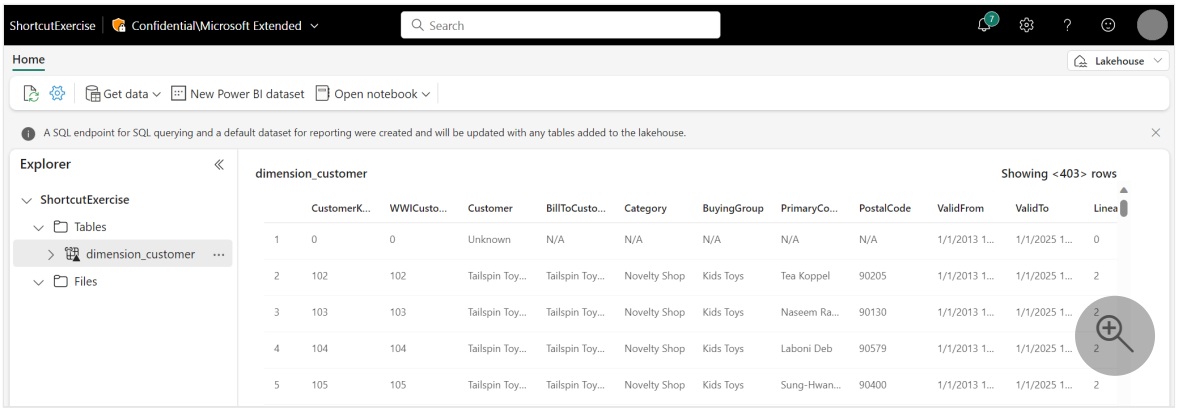
7. In the OneLake object browser, expand **Tables**, expand the `dbo` schema, and then select the radio button beside `dimension_customer`. Select the **Create** button.



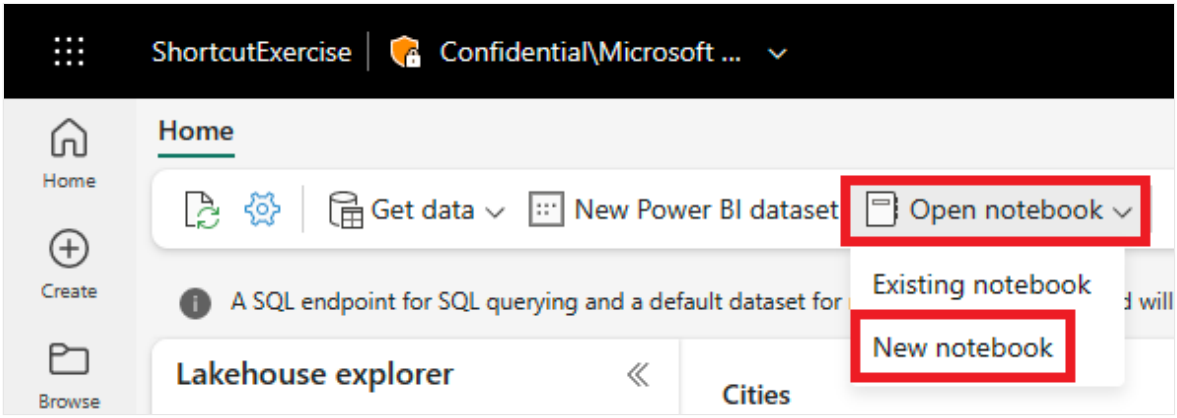
8. If you see a folder called `Unidentified` under **Tables**, select the **Refresh** icon in the horizontal menu bar.



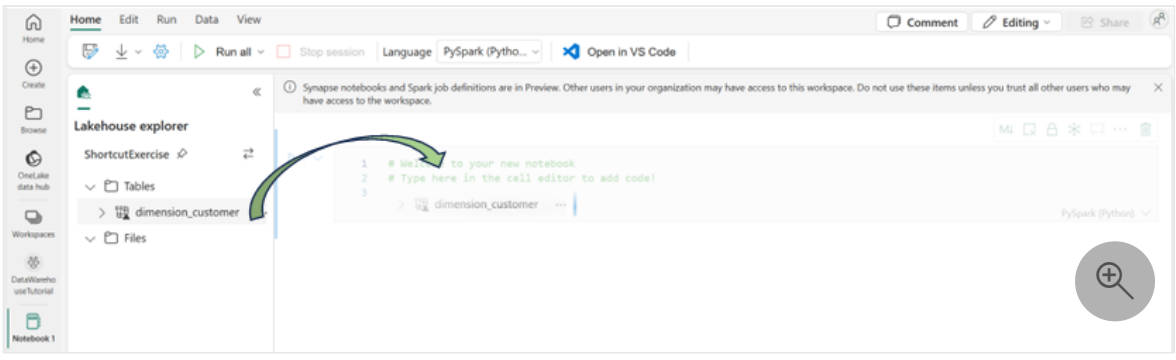
9. Select the `dimension_customer` in the **Table** list to preview the data. Notice that the lakehouse is showing the data from the `dimension_customer` table from the Warehouse!



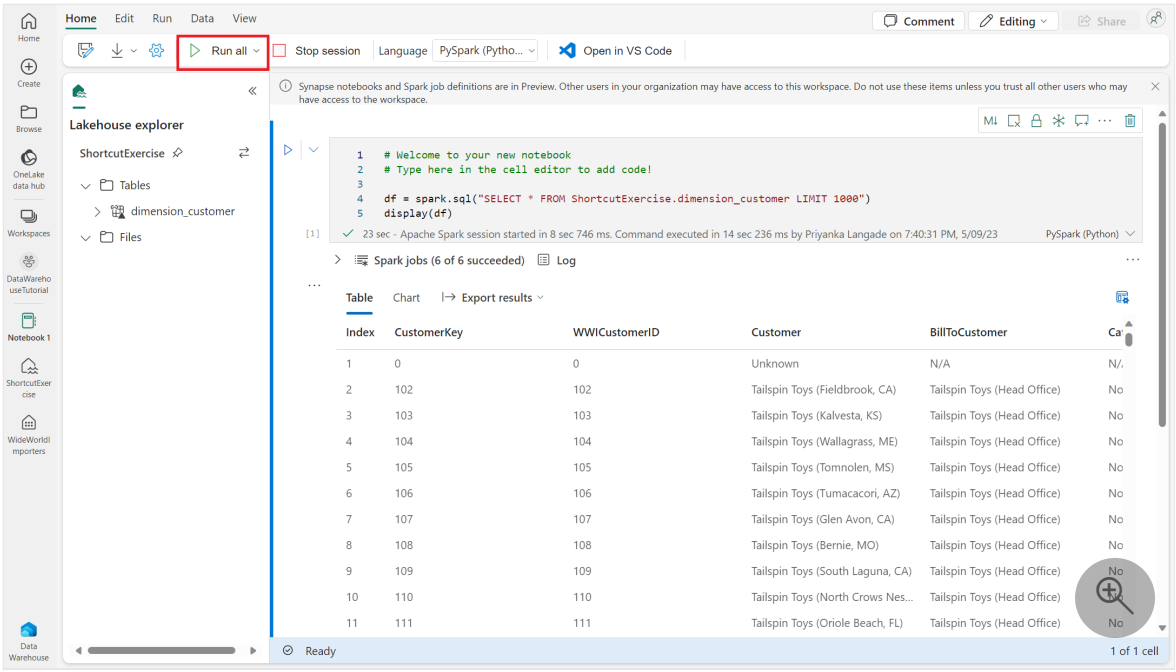
10. Next, create a new notebook to query the `dimension_customer` table. In the **Home** ribbon, select the drop down for **Open notebook** and choose **New notebook**.



11. Select, then drag the `dimension_customer` from the **Tables** list into the open notebook cell. You can see a PySpark query has been written for you to query all the data from `ShortcutExercise.dimension_customer`. This notebook experience is similar to Visual Studio Code Jupyter notebook experience. You can also open the notebook in VS Code.



12. In the **Home** ribbon, select the **Run all** button. Once the query is completed, you will see you can easily use PySpark to query the Warehouse tables!



Next steps

Tutorial: Create cross-warehouse queries with the SQL query editor

Tutorial: Create cross-warehouse queries with the SQL query editor

Article • 05/23/2023

Applies to: SQL Endpoint and Warehouse in Microsoft Fabric

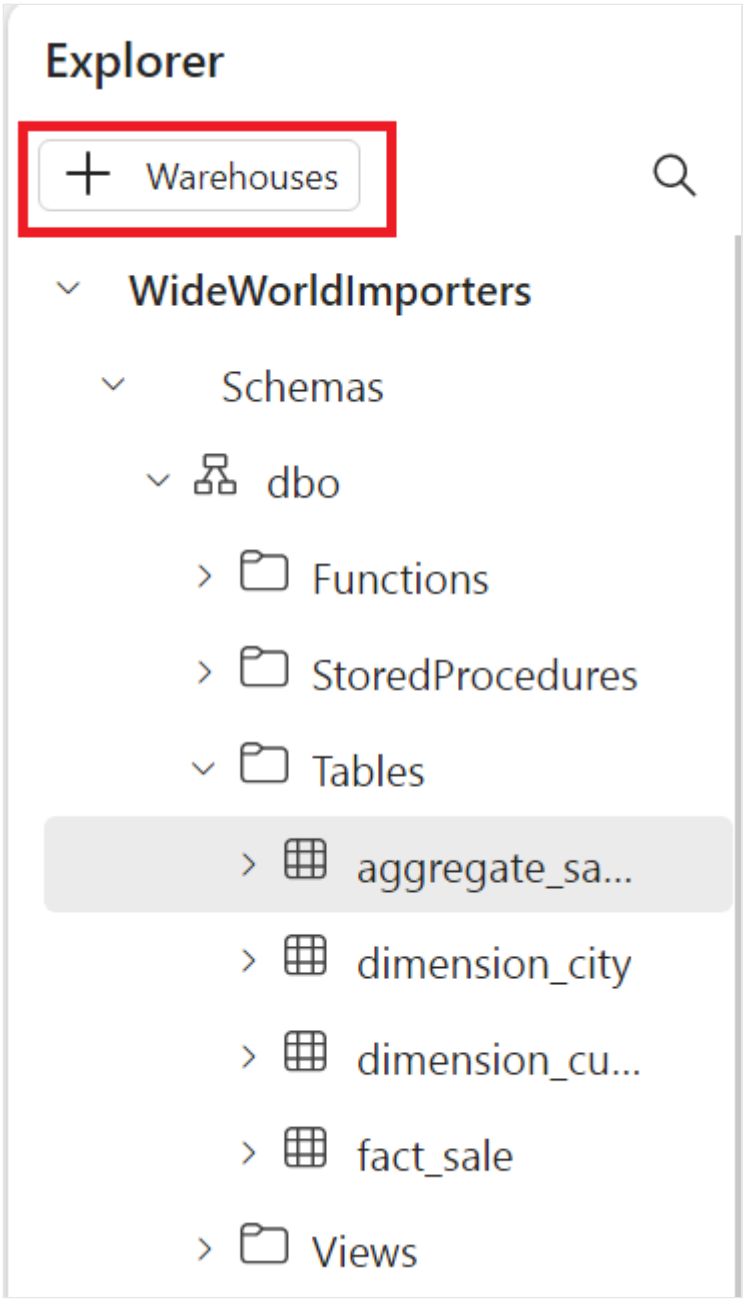
In this tutorial, learn about how you can easily create and execute T-SQL queries with the SQL query editor across multiple warehouse, including joining together data from a SQL Endpoint and a Warehouse in Microsoft Fabric.

> **Important**
>
> Microsoft Fabric is currently in PREVIEW. This information relates to a prerelease product that may be substantially modified before it's released. Microsoft makes no warranties, expressed or implied, with respect to the information provided here.

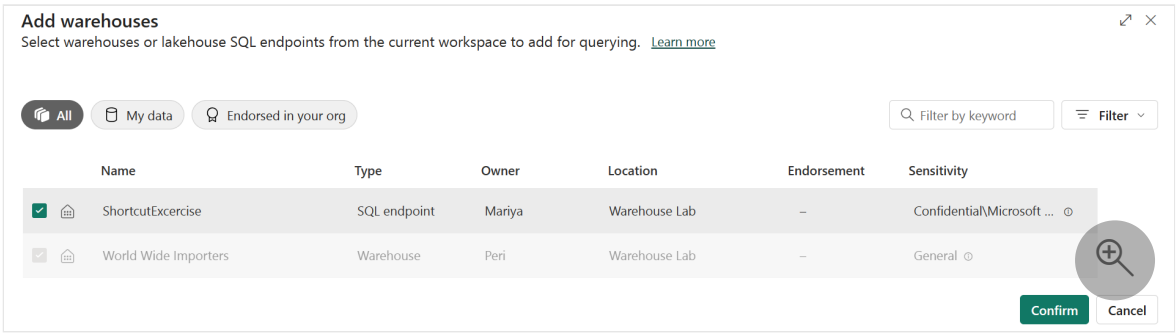
Add multiple warehouses to the Explorer

1. Select the `Data Warehouse Tutorial` workspace in the navigation menu.

2. In the **Explorer**, select the **+ Warehouses** button.



3. Select the SQL endpoint of the lakehouse you created using shortcuts previously, named `ShortcutExercise`. Both warehouse experiences are added to the query.

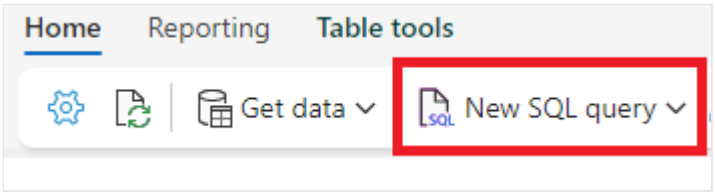


4. Your selected warehouses now show the same **Explorer** pane.

Execute a cross-warehouse query

In this example, you can see how easily you can run T-SQL queries across the `WideWorldImporters` warehouse and `ShortcutExercise` SQL Endpoint. You can write cross-database queries using three-part naming to reference the `database.schema.table`, as in SQL Server.

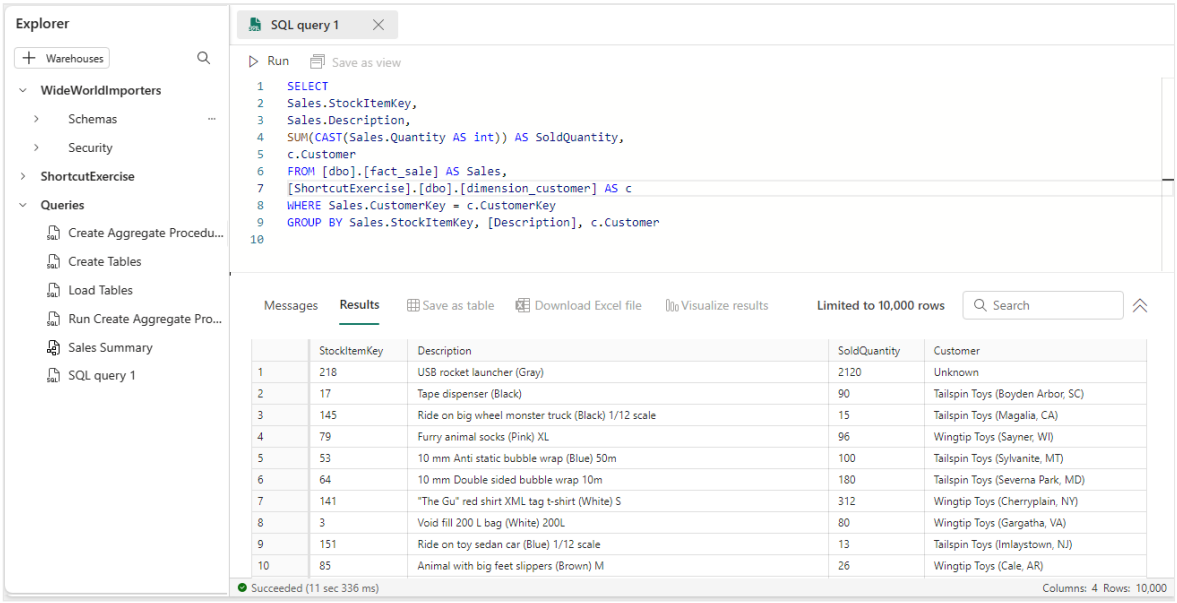
1. From the ribbon, select **New SQL query**.



2. In the query editor, copy and paste the following T-SQL code.

```SQL
SELECT Sales.StockItemKey,
Sales.Description,
SUM(CAST(Sales.Quantity AS int)) AS SoldQuantity,
c.Customer
FROM [dbo].[fact_sale] AS Sales,
[ShortcutExercise].[dbo].[dim_customer] AS c
WHERE Sales.CustomerKey = c.CustomerKey
GROUP BY Sales.StockItemKey, [Description], c.Customer;
```

3. Select the **Run** button to execute the query. After the query is completed, you will see the results.



4. Rename the query for reference later. Right-click on `SQL query 1` in the **Explorer** and select **Rename**.

5. Type `Cross-warehouse query` to change the name of the query.

6. Press **Enter** on the keyboard or select anywhere outside the tab to save the change.

Next steps

Tutorial: Create a Power BI report

Tutorial: Create Power BI reports

Article • 05/23/2023

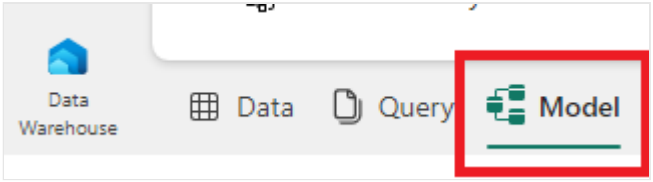
Applies to: SQL Endpoint and Warehouse in Microsoft Fabric

Learn how to create and save several types of Power BI reports.

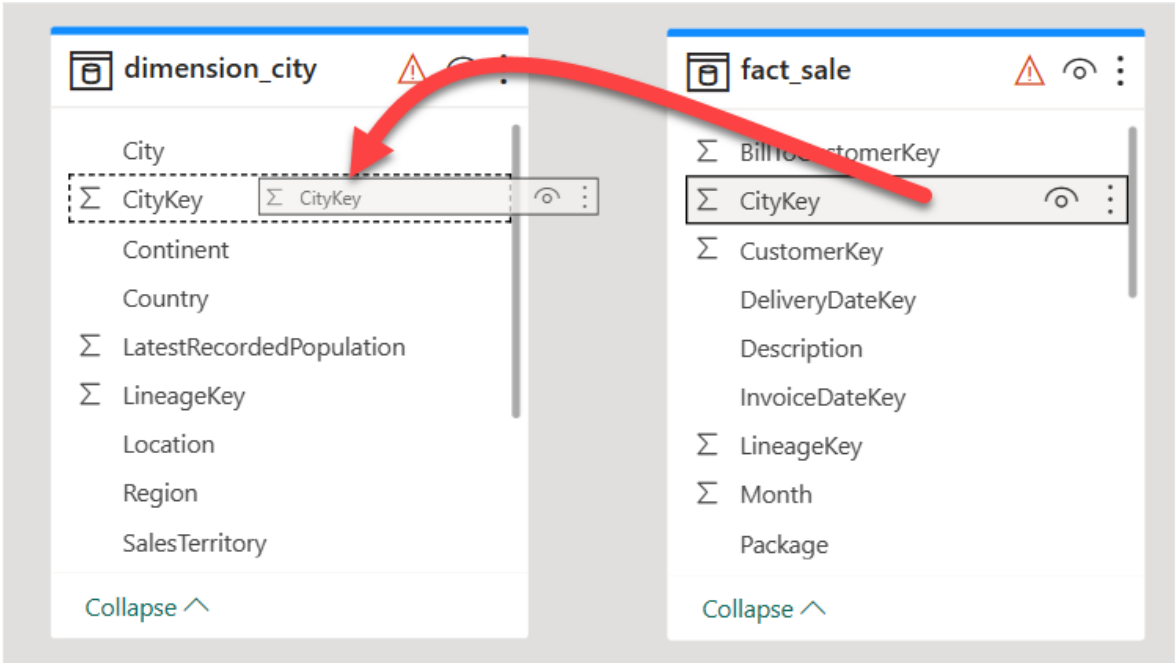
> **Important**
>
> Microsoft Fabric is currently in PREVIEW. This information relates to a prerelease product that may be substantially modified before it's released. Microsoft makes no warranties, expressed or implied, with respect to the information provided here.

Create reports

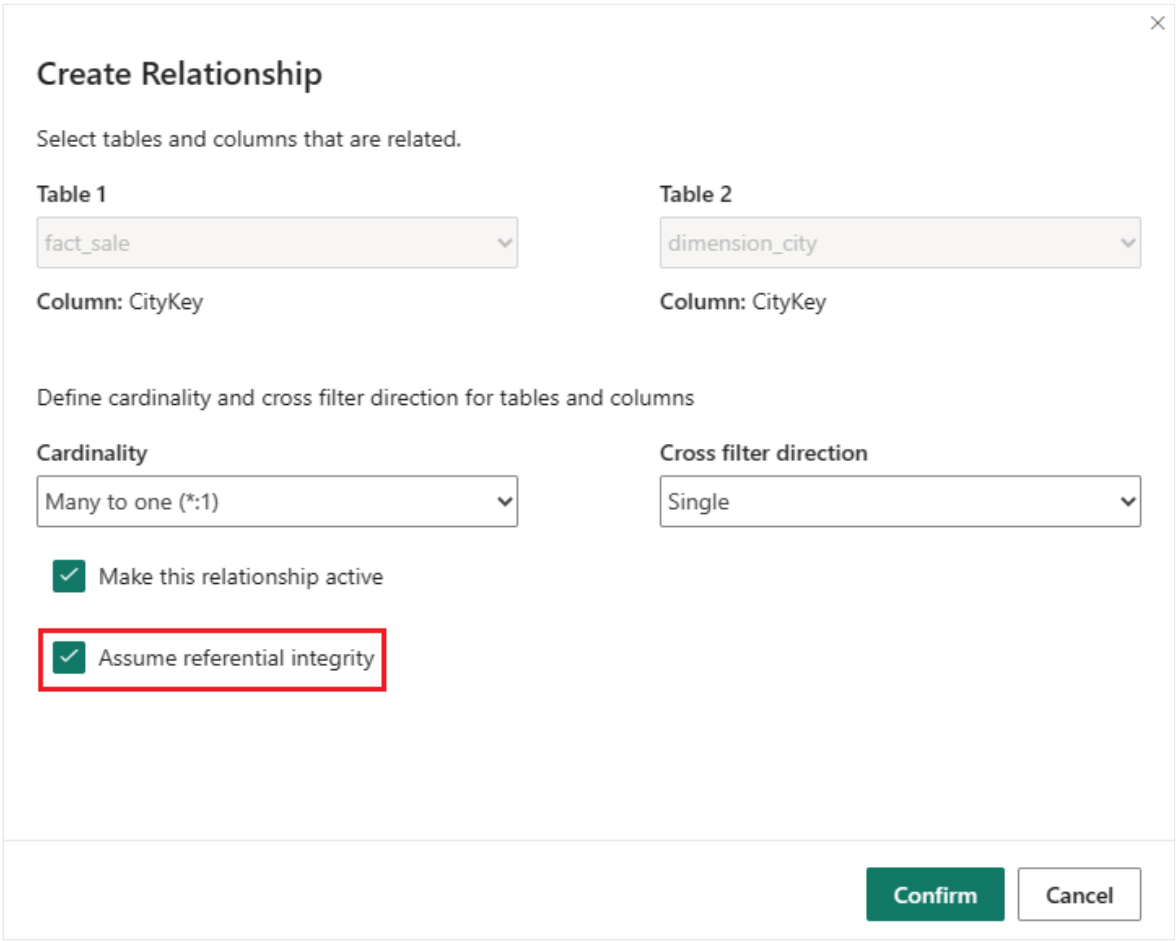
1. Select the **Model** view from the options in the bottom left corner, just outside the canvas.



2. From the `fact_sale` table, drag the `CityKey` field and drop it onto the `CityKey` field in the `dimension_city` table to create a relationship.



3. On the **Create Relationship** settings:
    a. Table 1 is populated with `fact_sale` and the column of `CityKey`.
    b. Table 2 is populated with `dimension_city` and the column of `CityKey`.
    c. **Cardinality**: select **Many to one (*:1)**.
    d. **Cross filter direction**: select **Single**.
    e. Leave the box next to **Make this relationship active** checked.
    f. Check the box next to **Assume referential integrity**.



4. Select **Confirm**.

5. From the **Home** tab of the ribbon, select **New report**.

6. Build a **Column chart** visual:

    a. On the **Data** pane, expand **fact_sales** and check the box next to **Profit**. This creates a column chart and adds the field to the Y-axis.

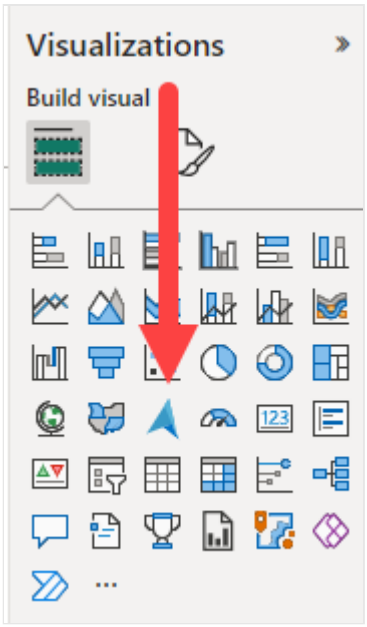
    b. On the **Data** pane, expand **dimension_city** and check the box next to **SalesTerritory**. This adds the field to the X-axis.

    c. Reposition and resize the column chart to take up the top left quarter of the canvas by dragging the anchor points on the corners of the visual.

7. Select anywhere on the blank canvas (or press the `Esc` key) so the column chart visual is no longer selected.

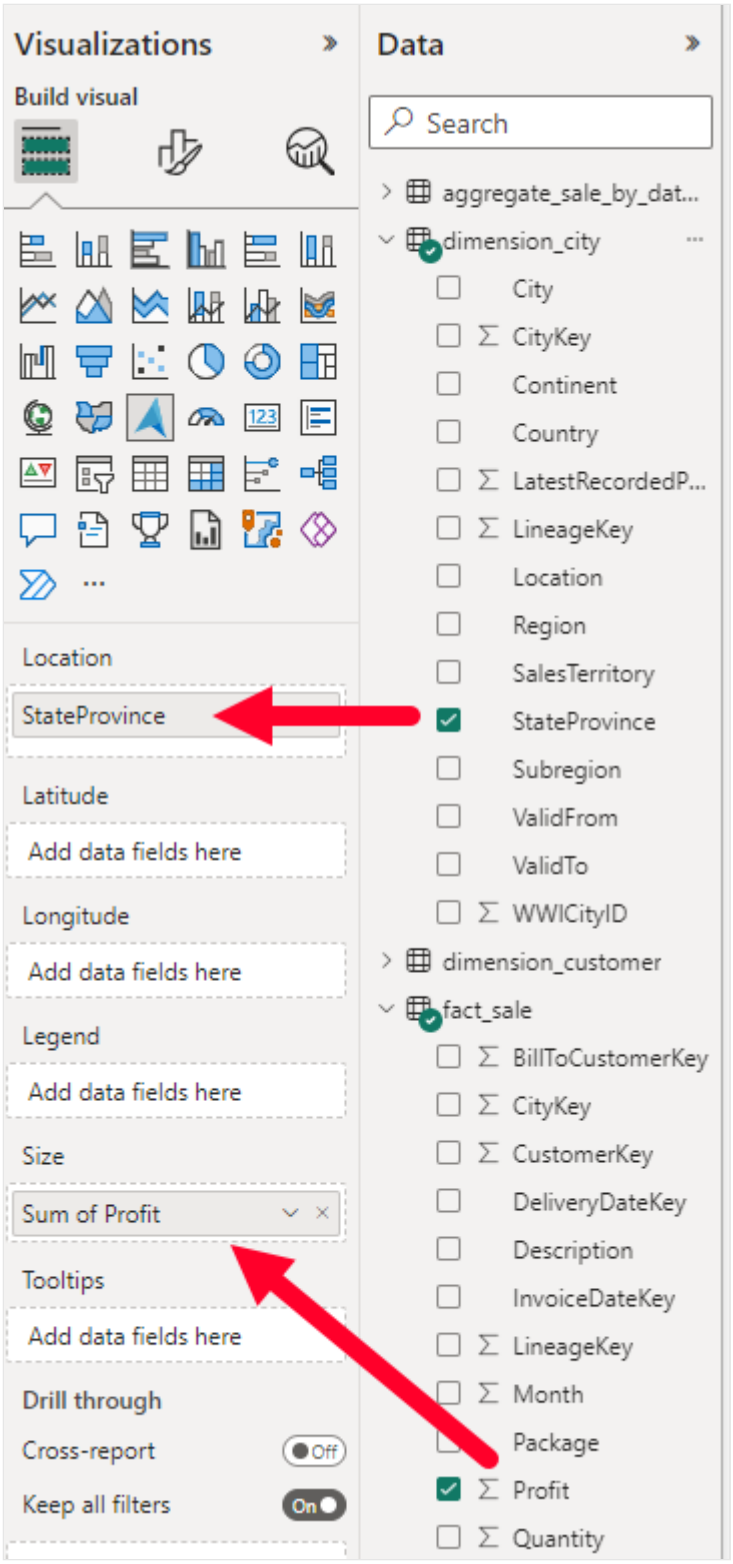
8. Build a **Maps** visual:

   a. On the **Visualizations** pane, select the **ArcGIS Maps for Power BI** visual.

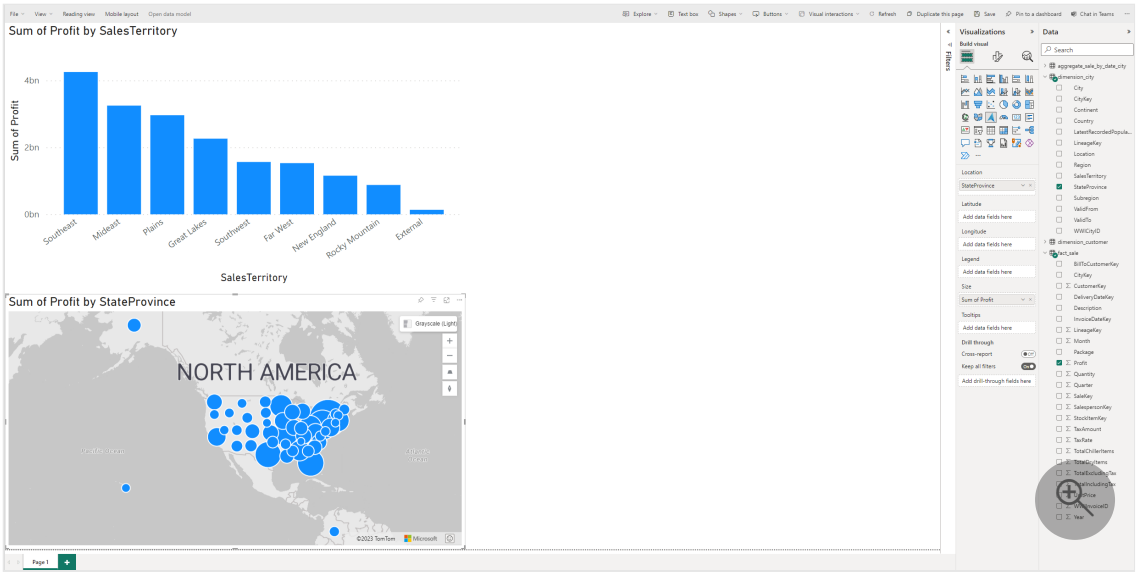


   b. From the **Data** pane, drag **StateProvince** from the `dimension_city` table to the **Location** bucket on the **Visualizations** pane.

   c. From the **Data** pane, drag **Profit** from the `fact_sale` table to the **Size** bucket on the **Visualizations** pane.



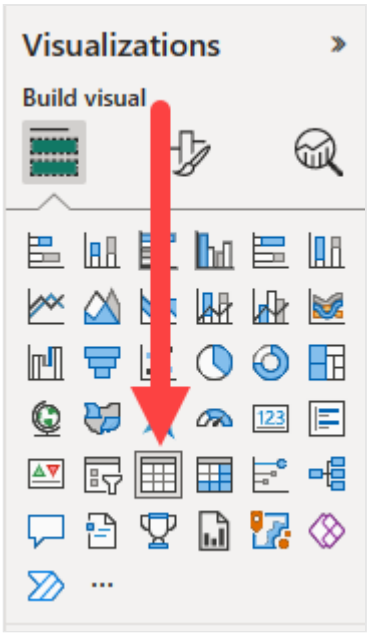
d. If necessary, reposition and resize the map to take up the bottom left quarter of the canvas by dragging the anchor points on the corners of the visual.



9. Select anywhere on the blank canvas (or press the Esc key) so the map visual is no longer selected.

10. Build a **Table** visual:

    a. On the **Visualizations** pane, select the **Table** visual.



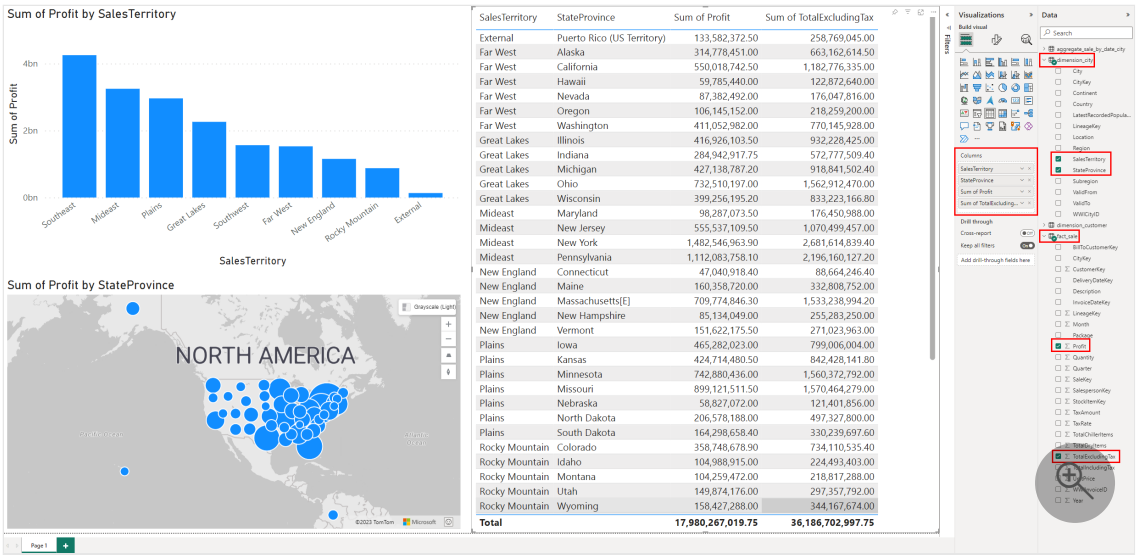
    b. From the **Data** pane, check the box next to **SalesTerritory** on the `dimension_city` table.

    c. From the **Data** pane, check the box next to **StateProvince** on the `dimension_city` table.

    d. From the **Data** pane, check the box next to **Profit** on the `fact_sale` table.

    e. From the **Data** pane, check the box next to **TotalExcludingTax** on the `fact_sale` table.

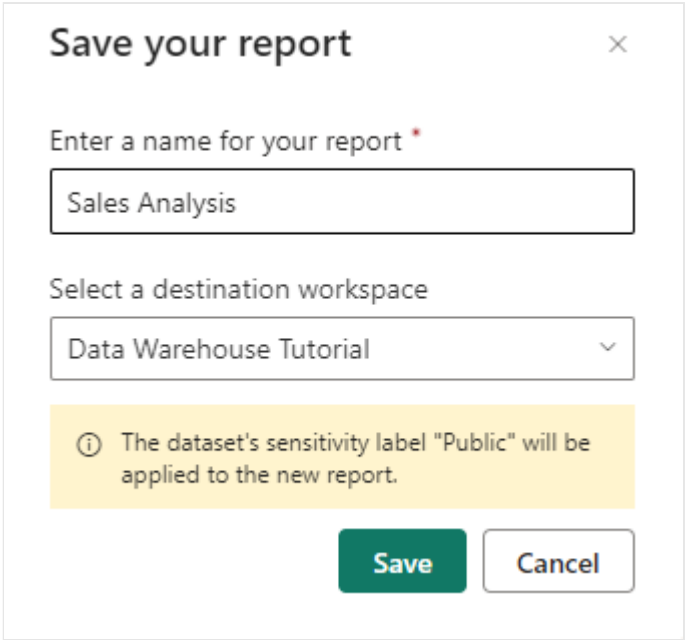
f. Reposition and resize the column chart to take up the right half of the canvas by dragging the anchor points on the corners of the visual.



11. From the ribbon, select **File** > **Save**.

12. Enter **Sales Analysis** as the name of your report.

13. Select **Save**.



Next steps

Tutorial: Build a report from the OneLake data hub

Tutorial: Build a report from the OneLake data hub

Article • 05/23/2023

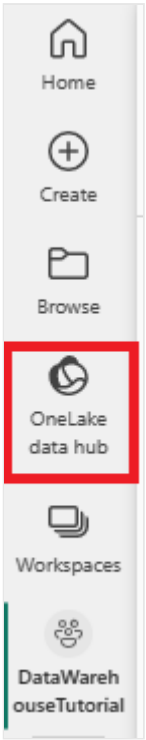
Applies to: SQL Endpoint and Warehouse in Microsoft Fabric

Learn how to build a report with the data you ingested into your Warehouse in the last step.

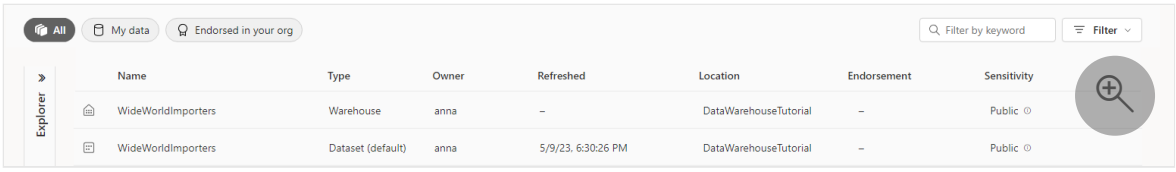
> **Important**
>
> Microsoft Fabric is currently in PREVIEW. This information relates to a prerelease product that may be substantially modified before it's released. Microsoft makes no warranties, expressed or implied, with respect to the information provided here.

Build a report

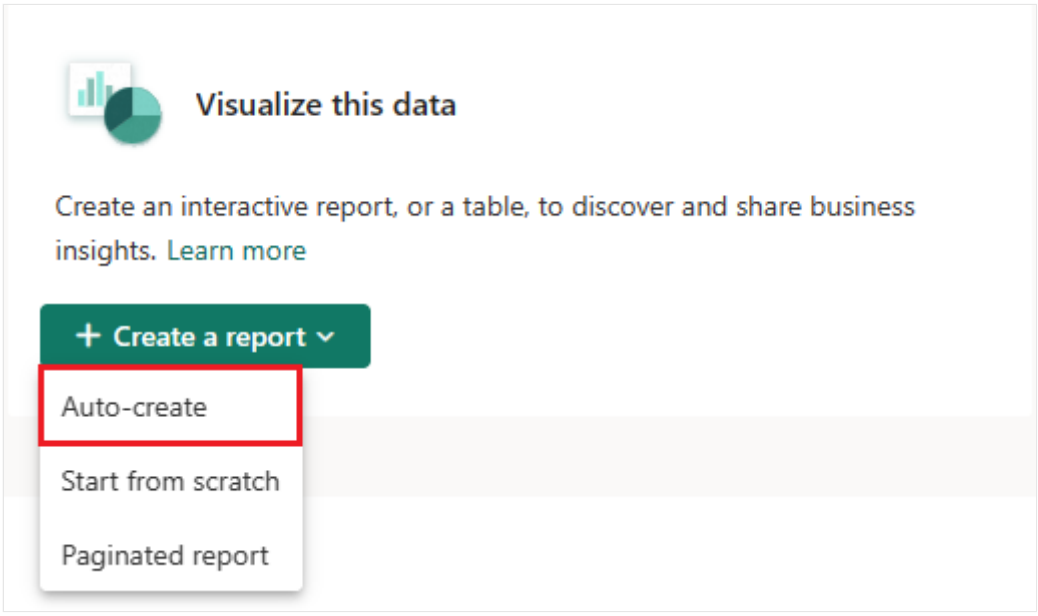
1. Select the **OneLake data hub** in the navigation menu.



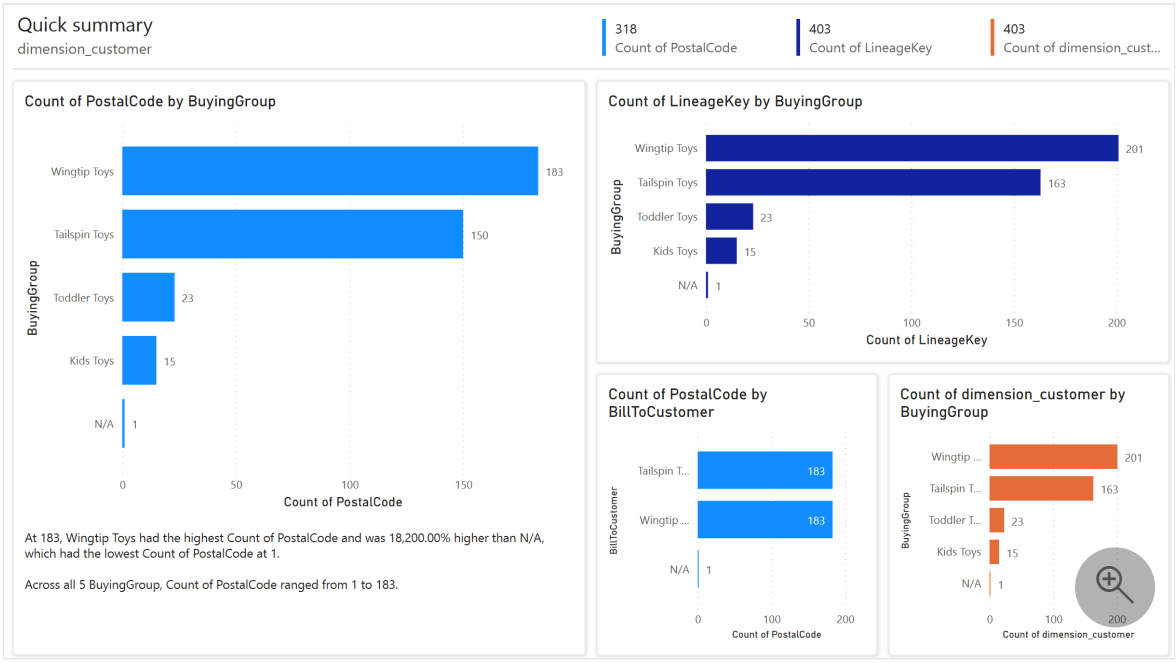
2. From the item list, select `WideWorldImporters` with the type of **Dataset (default)**.



3. In the **Visualize this data** section, select **Create a report** > **Auto-create**. A report is generated from the `dimension_customer` table that was loaded in the previous section.

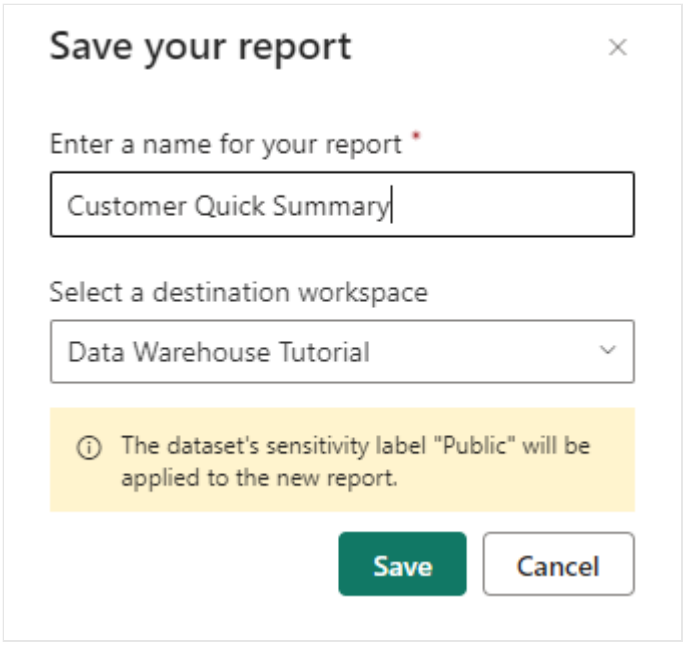


4. A report similar to the following image is generated.



5. From the ribbon, select **Save**.

6. Enter `Customer Quick Summary` in the name box. In the **Save your report** dialogue, select **Save**.



7. Your tutorial is complete!

    - Review Security for data warehousing in Microsoft Fabric.
    - Learn more about Workspace roles in Fabric data warehousing.
    - Consider Microsoft Purview, included by default in every tenant to meet important compliance and governance needs.

Next steps

Tutorial: Clean up tutorial resources

Tutorial: Clean up tutorial resources

Article • 05/23/2023

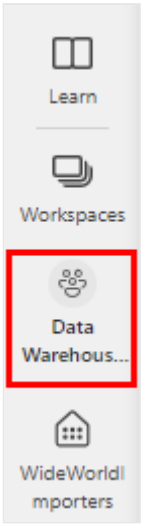
Applies to: ✅ SQL Endpoint and Warehouse in Microsoft Fabric

You can delete individual reports, pipelines, warehouses, and other items or remove the entire workspace. In this tutorial, you will clean up the workspace, individual reports, pipelines, warehouses, and other items you created as part of the tutorial.

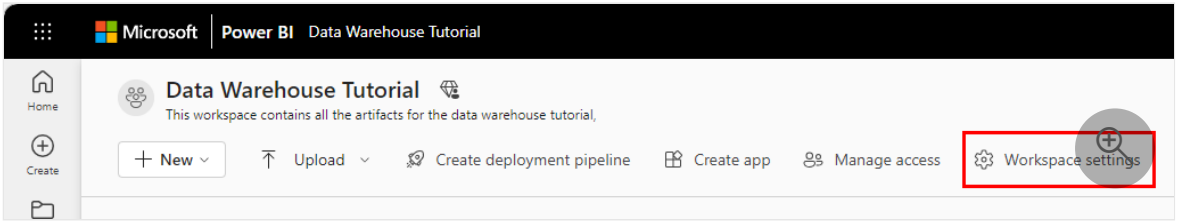
> **ⓘ Important**
>
> Microsoft Fabric is currently in PREVIEW. This information relates to a prerelease product that may be substantially modified before it's released. Microsoft makes no warranties, expressed or implied, with respect to the information provided here.

Delete a workspace

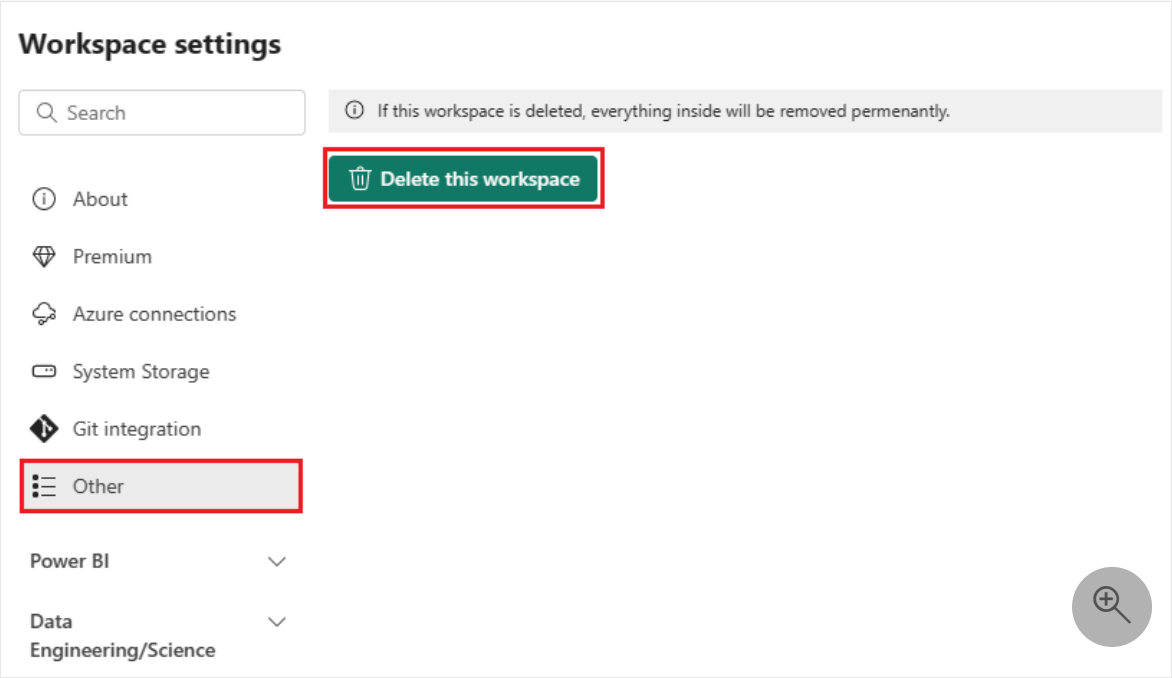
1. Select **Data Warehouse Tutorial** in the navigation menu to return to the workspace item list.



2. In the menu of the workspace header, select **Workspace settings**.



3. Select **Other** > **Delete this workspace**.



4. Select **Delete** on the warning to remove the workspace and all its contents.

Next steps

- What is data warehousing in Microsoft Fabric?

Connectivity to data warehousing in Microsoft Fabric

Article • 05/23/2023

Applies to: ✅ SQL Endpoint and Warehouse in Microsoft Fabric

In Microsoft Microsoft Fabric, a Lakehouse SQL Endpoint or Warehouse is accessible through a Tabular Data Stream, or TDS endpoint, familiar to all modern web applications that interact with a SQL Server endpoint. This is referred to as the SQL Connection String within the Microsoft Fabric user interface.

This article provides a how-to on connecting to your SQL Endpoint or Warehouse.

> **ⓘ Important**
>
> Microsoft Fabric is currently in PREVIEW. This information relates to a prerelease product that may be substantially modified before it's released. Microsoft makes no warranties, expressed or implied, with respect to the information provided here.

To get started, you must complete the following prerequisites:

- You need access to a SQL Endpoint or a Warehouse within a Premium capacity workspace with contributor or above permissions.

Authentication to warehouses in Fabric

In Microsoft Fabric, two types of authenticated users are supported through the SQL connection string:

- Azure Active Directory (Azure AD) user principals, or user identities
- Azure Active Directory (Azure AD) service principals

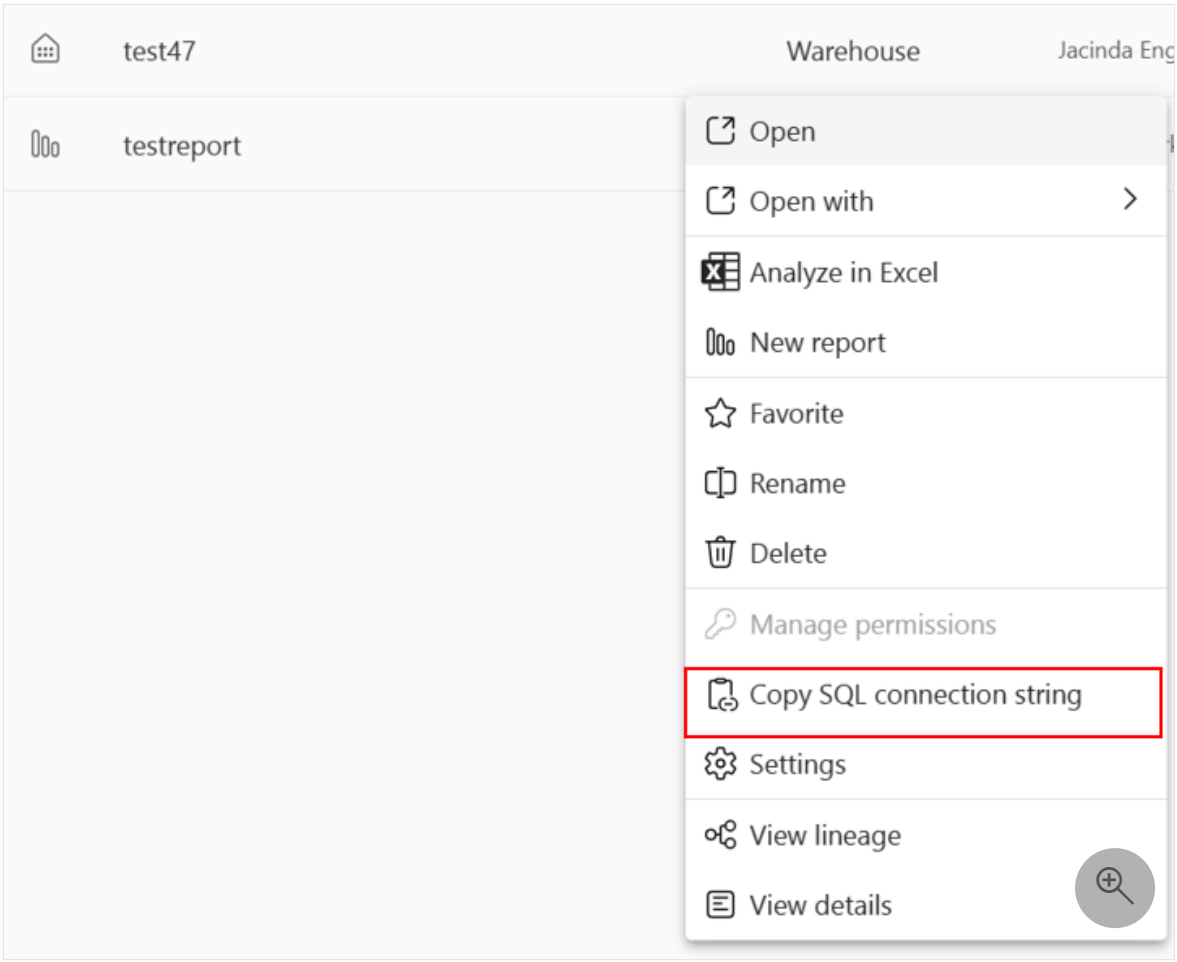
The SQL connection string requires TCP port 1433 to be open. TCP 1433 is the standard SQL Server port number. The SQL connection string also respects the Warehouse or Lakehouse SQL Endpoint security model for data access. Data can be obtained for all objects to which a user has access.

Retrieve the SQL connection string

To retrieve the connection string, follow these steps:

1. Navigate to your workspace, select the Warehouse, and select **More options**.

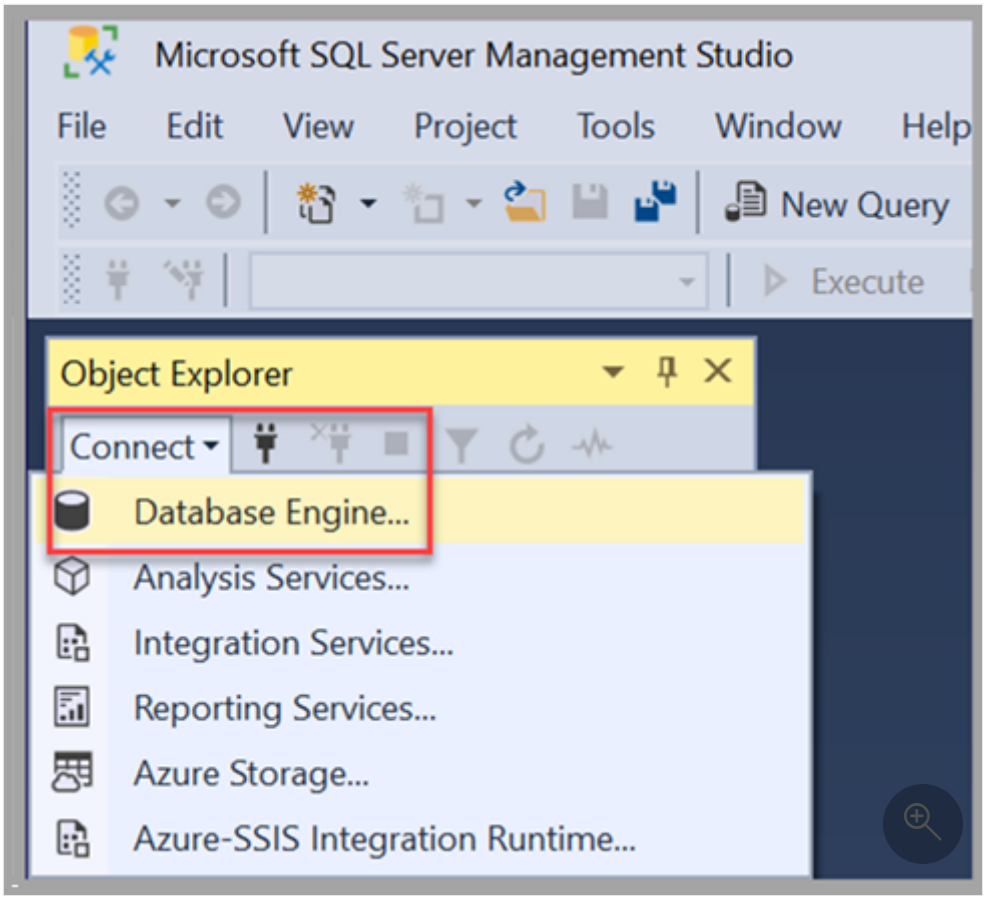
2. Select **Copy SQL connection string** to copy the connection string to your clipboard.



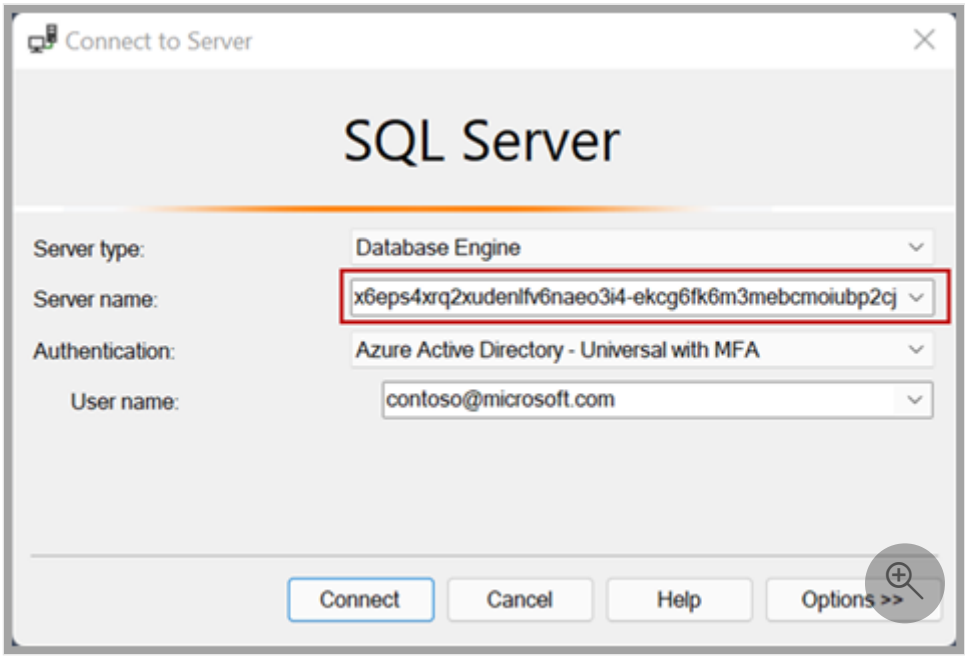
Get started with SQL Server Management Studio (SSMS)

The following steps detail how to start at the Microsoft Fabric workspace and connect a warehouse to SQL Server Management Studio (SSMS).

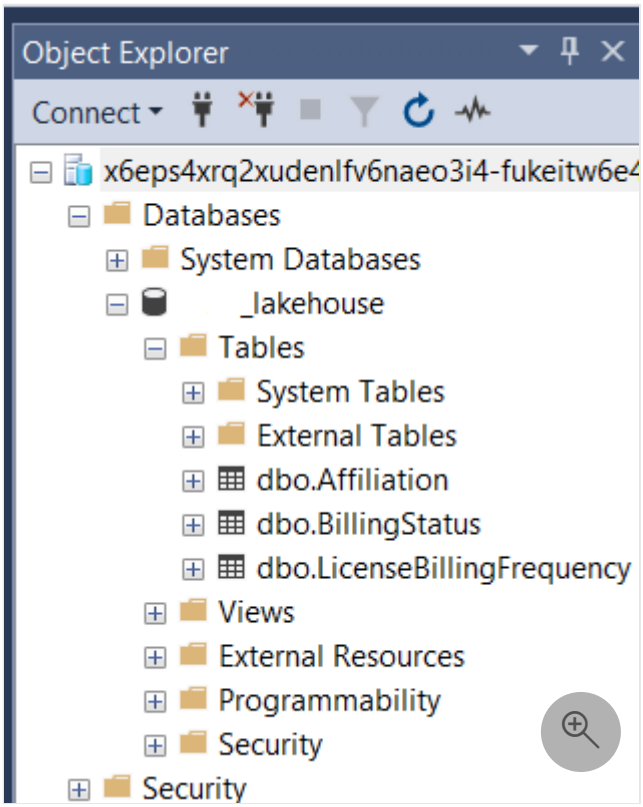
1. When you open SSMS, the **Connect to Server** window appears. If already open, you can connect manually by selecting **Object Explorer** > **Connect** > **Database Engine**.



2. Once the **Connect to Server** window is open, paste the connection string copied from the previous section of this article into the **Server name** box. Select **Connect** and proceed with the appropriate credentials for authentication. Remember that only **Azure Active Directory - MFA** authentication is supported.



3. Once the connection is established, Object Explorer displays the connected warehouse from the workspace and its respective tables and views, all of which are ready to be queried.



When connecting via SSMS (or ADS), you see both a SQL Endpoint and Warehouse listed as warehouses, and it's difficult to differentiate between the two item types and their functionality. For this reason, we strongly encourage you to adopt a naming convention that allows you to easily distinguish between the two item types when you work in tools outside of the Microsoft Fabric portal experience.

Connect using Power BI

A Warehouse or Lakehouse SQL Endpoint is a fully supported and native data source within Power BI, and there is no need to use the SQL Connection string. The Data Hub exposes all of the warehouses you have access to directly. This allows you to easily find your warehouses by workspace, and:

1. Select the Warehouse
2. Choose entities
3. Load Data - choose a data connectivity mode: import or DirectQuery

For more information, see Create reports in Microsoft Microsoft Fabric.

Connect using OLE DB

We support connectivity to the Warehouse or SQL Endpoint using OLE DB. Make sure you're running the latest Microsoft OLE DB Driver for SQL Server.

Connect using ODBC

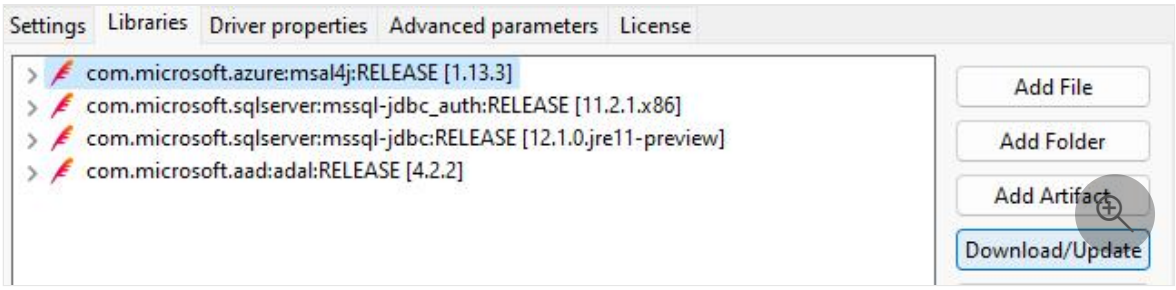
Microsoft Microsoft Fabric supports connectivity to the Warehouse or SQL Endpoint using ODBC. Make sure you're running the latest ODBC Driver for SQL Server. Use Azure Active Directory (Azure AD) authentication.

Connect using JDBC

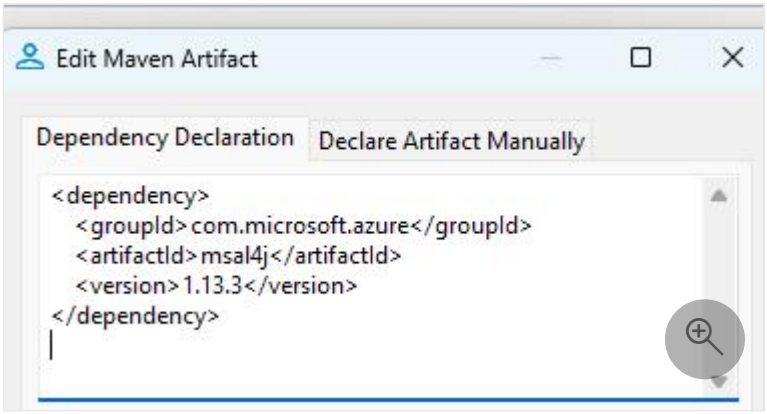
Microsoft Microsoft Fabric also supports connectivity to the Warehouse or SQL Endpoint using a Java database connectivity (JDBC) driver.

When establishing connectivity via JDBC, check for the following dependencies:

1. Add artifacts, choose **Add Artifact** and add the following four dependencies in the window like this, then select **Download/Update** to load all dependencies.



2. Select **Test connection**, and **Finish**.



XML

```
<dependency>
   <groupId>com.microsoft.azure</groupId>
   <artifactId>msal4j</artifactId>
   <version>1.13.3</version>

</dependency>

<dependency>
   <groupId>com.microsoft.sqlserver</groupId>
```

```
    <artifactId>mssql-jdbc_auth</artifactId>
    <version>11.2.1.x86</version>
</dependency>

 <dependency>
    <groupId>com.microsoft.sqlserver</groupId>
    <artifactId>mssql-jdbc</artifactId>
    <version>12.1.0.jre11-preview</version>
</dependency>

 <dependency>
    <groupId>com.microsoft.aad</groupId>
    <artifactId>adal</artifactId>
    <version>4.2.2</version>
</dependency>
```

Connect using DBT

Users typically use DBT adapters to connect DBT projects to a target datastore. DBT adapters are built specifically for each data source. Users who would like to connect to Synapse Data Warehouse in Microsoft Microsoft Fabric from DBT project must use the `dbt-synapsevnext` DBT adapter. Similarly, the Azure Synapse Analytics dedicated SQL pool data source has its own adapter, `dbt-synapse`.

The DBT Fabric DW Adapter uses the `pyodbc` library to establish connectivity with the Warehouse. The `pyodbc` library is an ODBC implementation in Python language that uses Python Database API Specification v2.0. The `pyodbc` library directly passes connection string to the database driver through SQLDriverConnect in the `msodbc` connection structure to Microsoft Fabric using a TDS (Tabular Data Streaming) proxy service.

The DBT Fabric DW adapter supports Azure Active Directory (Azure AD) authentication and allows developers to use `az cli authentication` using the dbt-synapsevnext adapter. SQL Authentication is not supported.

Connectivity by other means

Any third-party tool can use the SQL Connection string via ODBC or OLE DB drivers to connect to a Microsoft Microsoft Fabric Warehouse or SQL Endpoint, using Azure AD authentication.

Custom applications

In Microsoft Fabric, a Warehouse and a Lakehouse SQL Endpoint provide a SQL connection string. Data is accessible from a vast ecosystem of SQL tooling, provided they can authenticate using Azure AD. For more information, see Connection libraries for Microsoft SQL Database.

Considerations and limitations

- SQL Authentication is not supported.
- Multiple Active Result Sets (MARS) is unsupported for Microsoft Fabric Warehouse. MARS is disabled by default, however if `MultipleActiveResultSets` is included in the connection string, it should be removed or set to false.
- On connection to a warehouse, you may receive an error that "The token size exceeded the maximum allowed payload size". This may be due to having a large number of warehouses within the workspace or being a member of a large number of Azure AD groups. For most users, the error typically would not occur until approaching beyond 80 warehouses in the workspace. In event of this error, work with the Workspace admin to clean up unused Warehouses and retry the connection, or contact support if the problem persists.

Next steps

- Create a warehouse in Microsoft Microsoft Fabric
- Better together: the lakehouse and warehouse in Microsoft Microsoft Fabric

Better together: the lakehouse and warehouse

Article • 05/23/2023

Applies to: ✅ SQL Endpoint and Warehouse in Microsoft Fabric

This article explains the data warehousing experience with the SQL Endpoint of the Lakehouse, and scenarios for use of the Lakehouse in data warehousing.

> **ⓘ Important**
>
> Microsoft Fabric is currently in PREVIEW. This information relates to a prerelease product that may be substantially modified before it's released. Microsoft makes no warranties, expressed or implied, with respect to the information provided here.

What is a Lakehouse SQL Endpoint?

In Fabric, when you create a lakehouse, a Warehouse is automatically created.

The SQL Endpoint enables you to query data in the Lakehouse using T-SQL language and TDS protocol. Every Lakehouse has one SQL Endpoint, and each workspace can have more than one Lakehouse. The number of SQL Endpoints in a workspace matches the number of Lakehouse items.

- The SQL Endpoint is automatically generated for every Lakehouse and exposes Delta tables from the Lakehouse as SQL tables that can be queried using the T-SQL language.
- Every delta table from a Lakehouse is represented as one table. Data should be in delta format.
- The default Power BI dataset is created for every SQL Endpoint and it follows the naming convention of the Lakehouse objects.

There's no need to create a SQL Endpoint in Microsoft Fabric. Microsoft Fabric users can't create a SQL Endpoint in a workspace. A SQL Endpoint is automatically created for every Lakehouse. To get a SQL Endpoint, create a lakehouse and a SQL Endpoint will be automatically created for the Lakehouse.

> **ⓘ Note**

Behind the scenes, the SQL Endpoint is using the same engine as the **Warehouse** to serve high performance, low latency SQL queries.

Automatic Metadata Discovery

A seamless process reads the delta logs and from the files folder and ensures SQL metadata for tables, such as statistics, is always up to date. There's no user action needed, and no need to import, copy data, or set up infrastructure. For more information, see Automatically generated schema in the SQL Endpoint.

Scenarios the Lakehouse enables for data warehousing

In Fabric, we offer one warehouse.

The Lakehouse, with its SQL Endpoint, powered by the Warehouse, can simplify the traditional decision tree of batch, streaming, or lambda architecture patterns. Together with a warehouse, the lakehouse enables many additive analytics scenarios. This section explores how to leverage a Lakehouse together with a Warehouse for a best of breed analytics strategy.

Analytics with your Fabric Lakehouse's gold layer

One of the well-known strategies for lake data organization is a medallion architecture where the files are organized in raw (bronze), consolidated (silver), and refined (gold) layers. A SQL Endpoint can be used to analyze data in the gold layer of medallion architecture if the files are stored in `Delta Lake` format, even if they're stored outside the Microsoft Fabric OneLake.

You can use OneLake shortcuts to reference gold folders in external Azure Data Lake storage accounts that are managed by Synapse Spark or Azure Databricks engines.

Warehouses can also be added as subject area or domain oriented solutions for specific subject matter that may have bespoke analytics requirements.

If you choose to keep your data in Fabric, it will **always be open** and accessible through APIs, Delta format, and of course T-SQL.

Query as a service over your delta tables from Lakehouse and other items from OneLake Data Hub

There are use cases where an analyst, data scientist, or data engineer may need to query data within a data lake. In Fabric, this end to end experience is completely SaaSified.

OneLake is a single, unified, logical data lake for the whole organization. OneLake is OneDrive for data. OneLake can contain multiple workspaces, for example, along your organizational divisions. Every item in Fabric makes it data accessible via OneLake.

Data in a Microsoft Fabric Lakehouse is physically stored in OneLake with the following folder structure:

- The `/Files` folder contains raw and unconsolidated (bronze) files that should be processed by data engineers before they're analyzed. The files might be in various formats such as CSV, Parquet, different types of images, etc.
- The `/Tables` folder contains refined and consolidated (gold) data that is ready for business analysis. The consolidated data is in Delta Lake format.

A SQL Endpoint can read data in the `/tables` folder within OneLake. Analysis is as simple as querying the SQL Endpoint of the Lakehouse. Together with the Warehouse, you also get cross-database queries and the ability to seamless switch from read-only queries to building additional business logic on top of your OneLake data with Synapse Data Warehouse.

Data Engineering with Spark, and Serving with SQL

Data-driven enterprises need to keep their back-end and analytics systems in near real-time sync with customer-facing applications. The impact of transactions must reflect accurately through end-to-end processes, related applications, and online transaction processing (OLTP) systems.

In Fabric, you can leverage Spark Streaming or Data Engineering to curate your data. You can use the Lakehouse SQL Endpoint to validate data quality and for existing T-SQL processes. This can be done in a medallion architecture or within multiple layers of your Lakehouse, serving bronze, silver, gold, or staging, curated, and refined data. You can customize the folders and tables created through Spark to meet your data engineering and business requirements. When ready, you can then leverage a Warehouse to serve all of your downstream business intelligence applications and other analytics use cases, without copying data, using Views or refining data using CREATE TABLE AS SELECT (CTAS), stored procedures, and other DML / DDL commands.

Integration with your Open Lakehouse's gold layer

A SQL Endpoint is not scoped to data analytics in just the Fabric Lakehouse. A SQL Endpoint enables you to analyze lake data in any lakehouse, using Synapse Spark, Azure Databricks, or any other lake-centric data engineering engine. The data can be stored in Azure Data Lake Storage or Amazon S3.

This tight, bi-directional integration with the Fabric Lakehouse is always accessible through any engine with open APIs, the Delta format, and of course T-SQL.

Data Virtualization of external data lakes with Shortcuts

You can use OneLake shortcuts to reference gold folders in external Azure Data Lake storage accounts that are managed by Synapse Spark or Azure Databricks engines, as well as any delta table stored in Amazon S3.

Any folder referenced using a shortcut can be analyzed from a SQL Endpoint and a SQL table is created for the referenced dataset. The SQL table can be used to expose data in externally managed data lakes and enable analytics on them.

This shortcut acts as a virtual warehouse that can leveraged from a warehouse for additional downstream analytics requirements, or queried directly.

Use the following steps to analyze data in external data lake storage accounts:

1. Create a shortcut that references a folder in Azure Data Lake storage or Amazon S3 account. Once you enter connection details and credentials, a shortcut is shown in the Lakehouse.
2. Switch to the SQL Endpoint of the Lakehouse and find a SQL table that has a name that matches the shortcut name. This SQL table references the folder in ADLS/S3 folder.
3. Query the SQL table that references data in ADLS/S3. The table can be used as any other table in the SQL Endpoint. You can join tables that reference data in different storage accounts.

> **Note**
>
> If the SQL table is not immediately shown in the SQL Endpoint, you might need to wait a few minutes. The SQL table that references data in external storage account is created with a delay.

Analyze archived, or historical data in a data lake

Data partitioning is a well-known data access optimization technique in data lakes. Partitioned data sets are stored in the hierarchical folders structures in the format `/year=<year>/month=<month>/day=<day>`, where `year`, `month`, and `day` are the partitioning columns. This allows you to store historical data logically separated in a format that allows compute engines to read the data as needed with performant filtering, versus reading the entire directory and all folders and files contained within.

Partitioned data enables faster access if the queries are filtering on the predicates that compare predicate columns with a value.

A SQL Endpoint can easily read this type of data with no configuration required. For example, you can use any application to archive data into a data lake, including SQL Server 2022 or Azure SQL Managed Instance. After you partitioning data and land it in a lake for archival purposes with external tables, a SQL Endpoint can read partitioned Delta Lake tables as SQL tables and allow your organization to analyze them. this reduces the total cost of ownership, reduces data duplication, and lights up big data, AI, other analytics scenarios.

Data virtualization of Fabric data with shortcuts

Within Fabric, workspaces allow you to segregate data based on complex business, geographic, or regulatory requirements.

A SQL Endpoint enables you to leave the data in place and still analyze data in the Warehouse or Lakehouse, even in other Microsoft Fabric workspaces, via a seamless virtualization. Every Microsoft Fabric Lakehouse stores data in OneLake.

Shortcuts enable you to reference folders in any OneLake location.

Every Microsoft Fabric Warehouse stores table data in OneLake. If a table is append-only, the table data is exposed as Delta Lake datasets in OneLake. Shortcuts enable you to reference folders in any OneLake where the Warehouse tables are exposed.

Cross workspace sharing and querying

While workspaces allow you to segregate data based on complex business, geographic, or regulatory requirements, sometimes you need to facilitate sharing across these lines for specific analytics needs.

A Lakehouse SQL Endpoint can enable easy sharing of data between departments and users, where a user can bring their own capacity and warehouse. Workspaces organize departments, business units, or analytical domains. Using shortcuts, users can find any

Warehouse or Lakehouse's data. Users can instantly perform their own customized analytics from the same shared data. In addition to helping with departmental chargebacks and usage allocation, this is a zero-copy version the data as well.

The SQL Endpoint enables querying of any table and easy sharing. The added controls of workspace roles and security roles that can be further layered to meet additional business requirements.

Use the following steps to enable cross-workspace data analytics:

1. Create a OneLake shortcut that references a table or a folder in a workspace that you can access.
2. Choose a Lakehouse or Warehouse that contains a table or Delta Lake folder that you want to analyze. Once you select a table/folder, a shortcut is shown in the Lakehouse.
3. Switch to the SQL Endpoint of the Lakehouse and find the SQL table that has a name that matches the shortcut name. This SQL table references the folder in another workspace.
4. Query the SQL table that references data in another workspace. The table can be used as any other table in the SQL Endpoint. You can join the tables that reference data in different workspaces.

> **Note**
>
> If the SQL table is not immediately shown in the SQL Endpoint, you might need to wait a few minutes. The SQL table that references data in another workspace is created with a delay.

Analyze partitioned data

Data partitioning is a well-known data access optimization technique in data lakes. Partitioned data sets are stored in the hierarchical folders structures in the format `/year=<year>/month=<month>/day=<day>`, where `year`, `month`, and `day` are the partitioning columns. Partitioned data sets enable faster data access if the queries are filtering data using the predicates that filter data by comparing predicate columns with a value.

A SQL Endpoint can represent partitioned Delta Lake data sets as SQL tables and enable you to analyze them.

Next steps

- What is a lakehouse?
- Create a lakehouse with OneLake
- Understand default Power BI datasets
- Load data into the lakehouse
- How to copy data using Copy activity in Data pipeline
- Tutorial: Move data into lakehouse via Copy assistant
- Connectivity
- Warehouse of the lakehouse
- Query the Warehouse

Create a sample Warehouse in Microsoft Fabric

Article • 05/23/2023

Applies to: Warehouse in Microsoft Fabric

This article describes how to get started with sample Warehouse using the Microsoft Fabric portal, including creation and consumption of the warehouse.

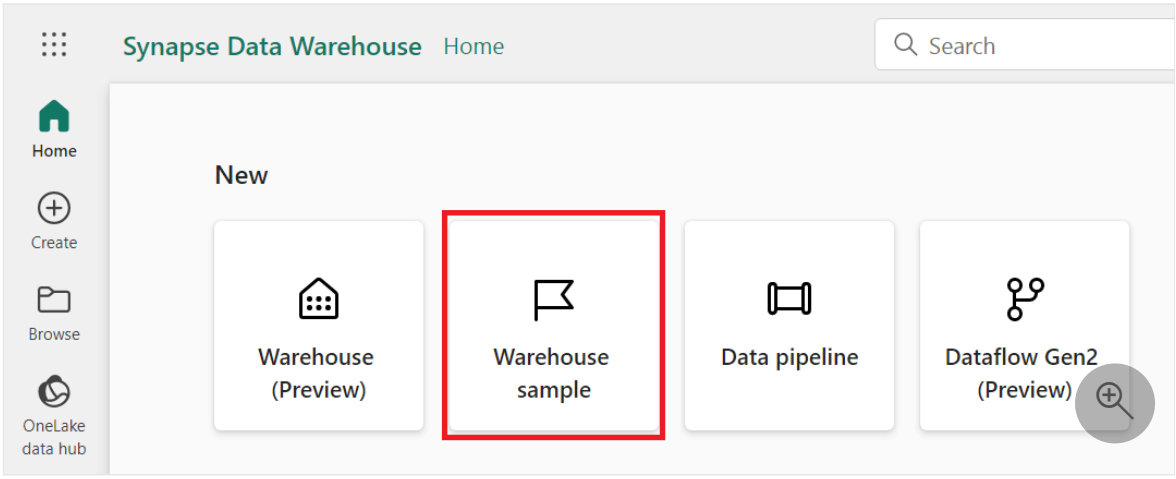
> **Important**
>
> Microsoft Fabric is currently in PREVIEW. This information relates to a prerelease product that may be substantially modified before it's released. Microsoft makes no warranties, expressed or implied, with respect to the information provided here.

How to create a warehouse sample

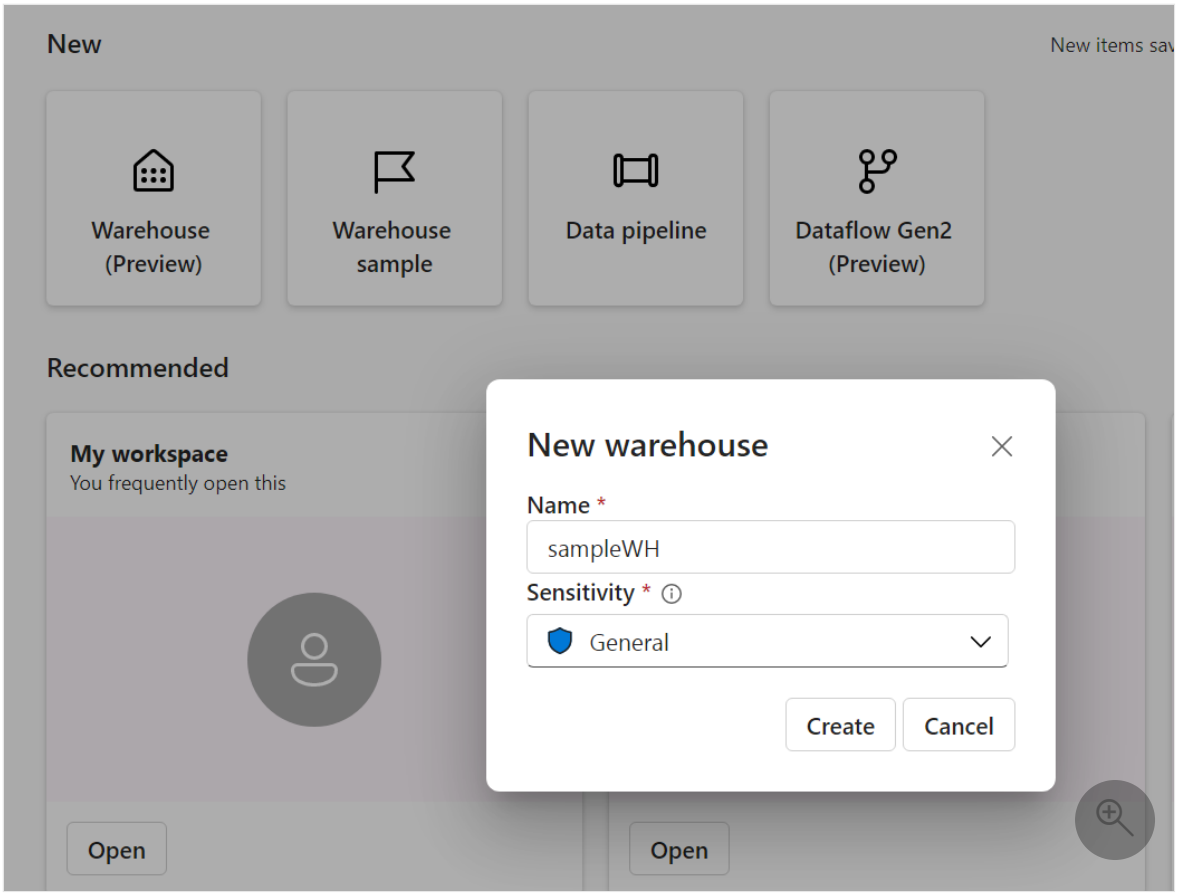
In this section, we walk you through two distinct experiences available for creating a sample Warehouse from scratch.

Create a warehouse sample using the Home hub

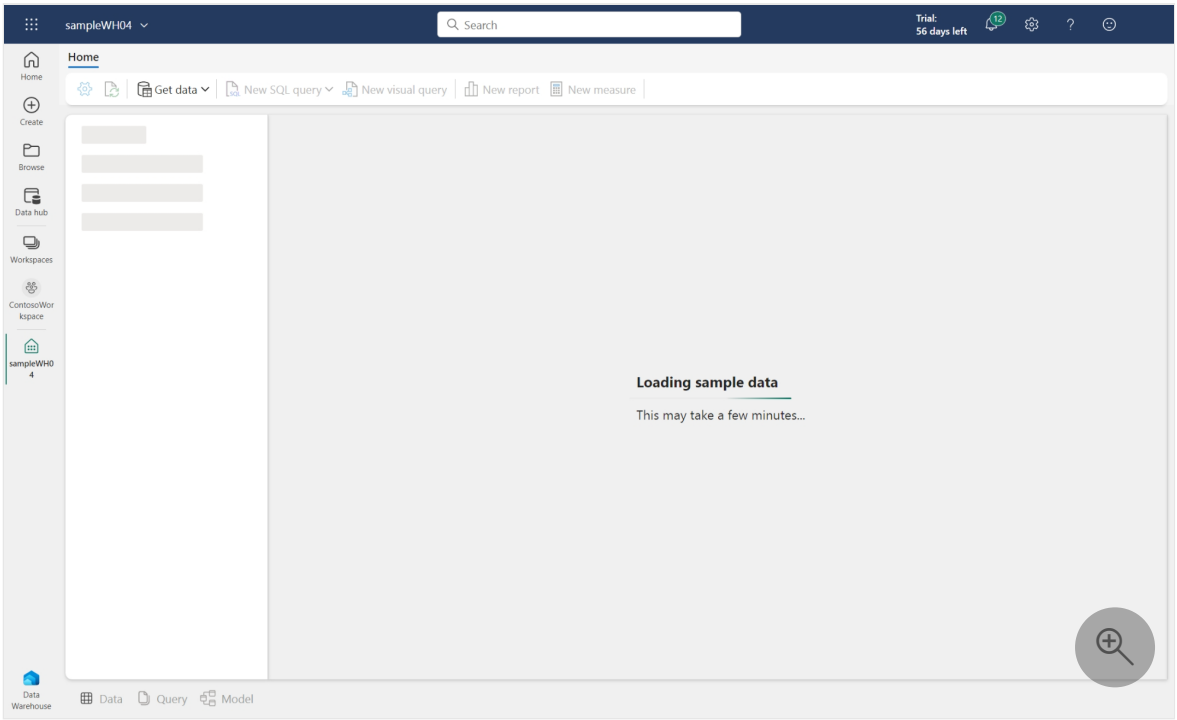
1. The first hub in the left navigation menus is the **Home** hub. You can start creating your warehouse sample from the **Home** hub by selecting the **Warehouse sample** card under the **New** section.



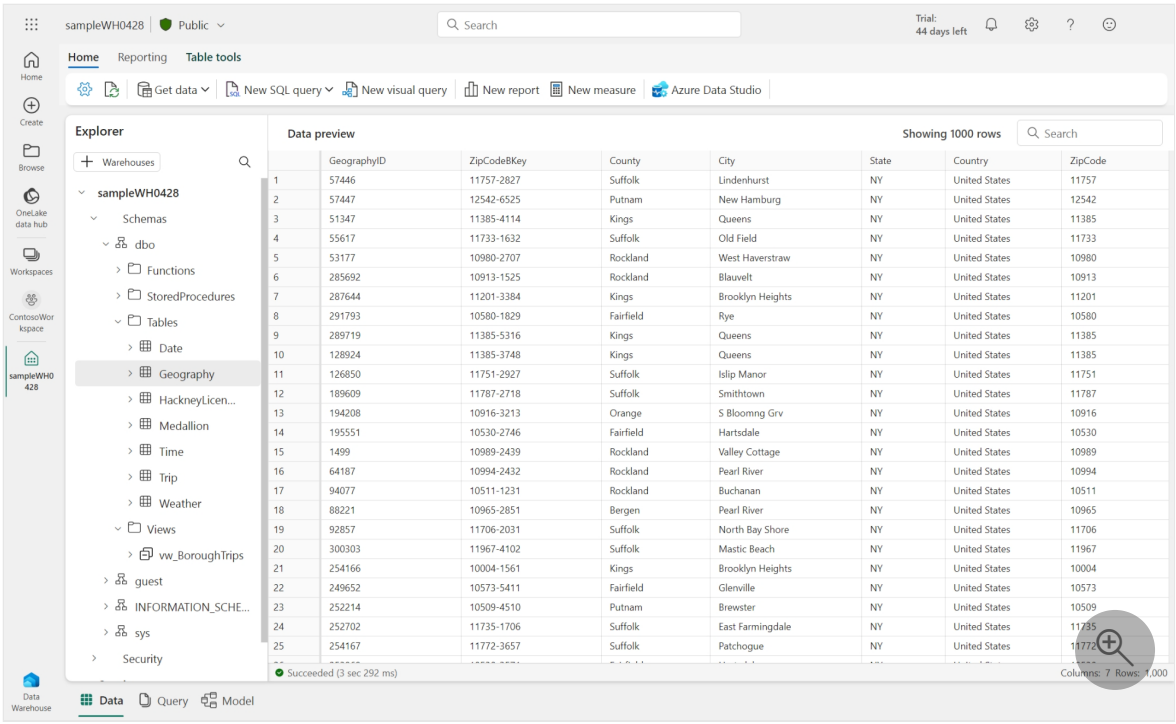
2. Provide the name for your sample warehouse and select **Create**.



3. The create action creates a new Warehouse and start loading sample data into it. The data loading takes few minutes to complete.



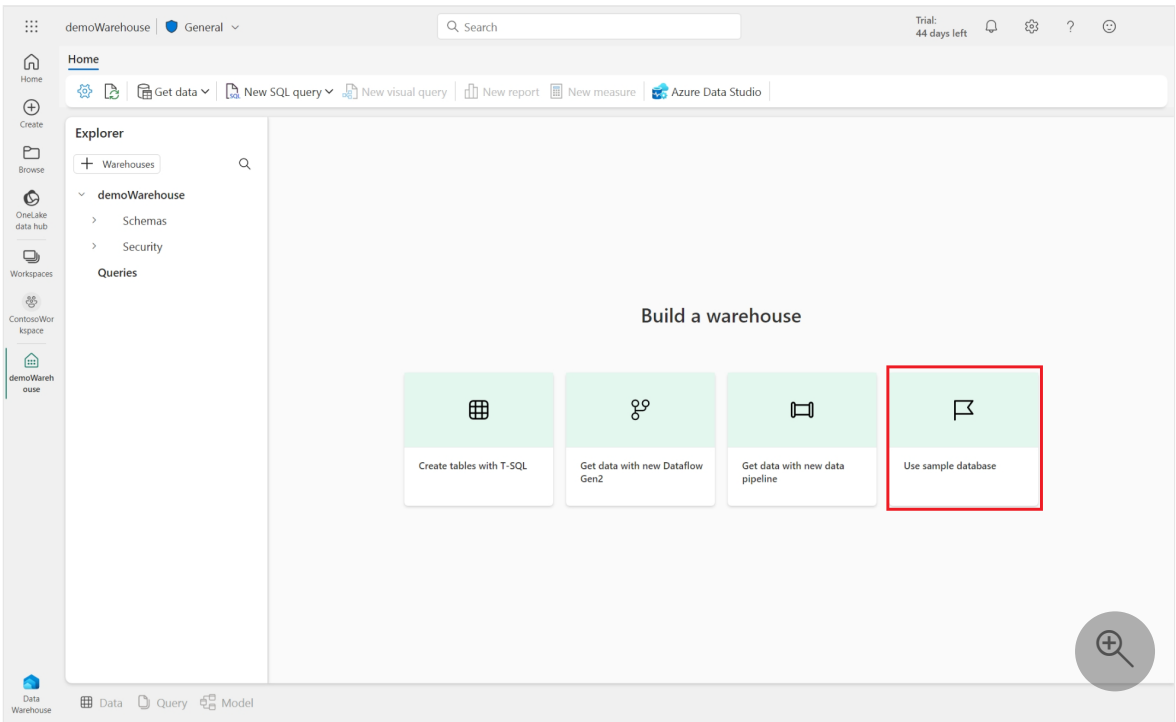
4. On completion of loading sample data, the warehouse opens with data loaded into tables and views to query.



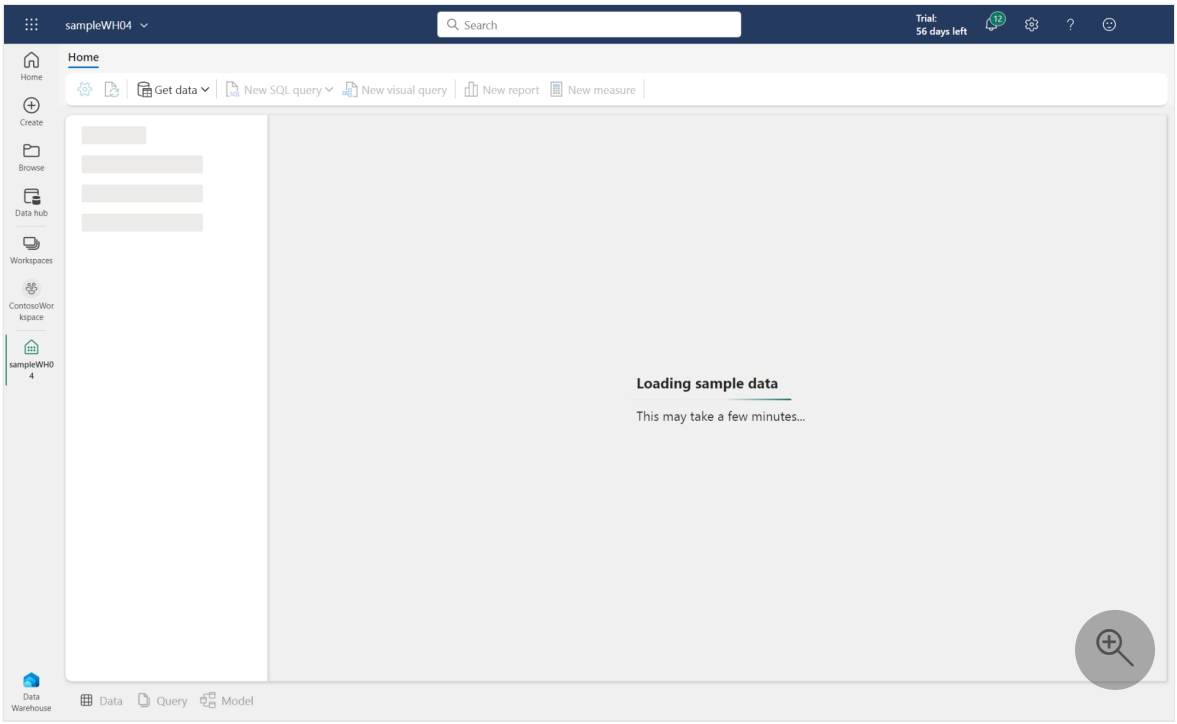
Load sample data into existing warehouse

For more information on how to create a warehouse, see Create a Warehouse.

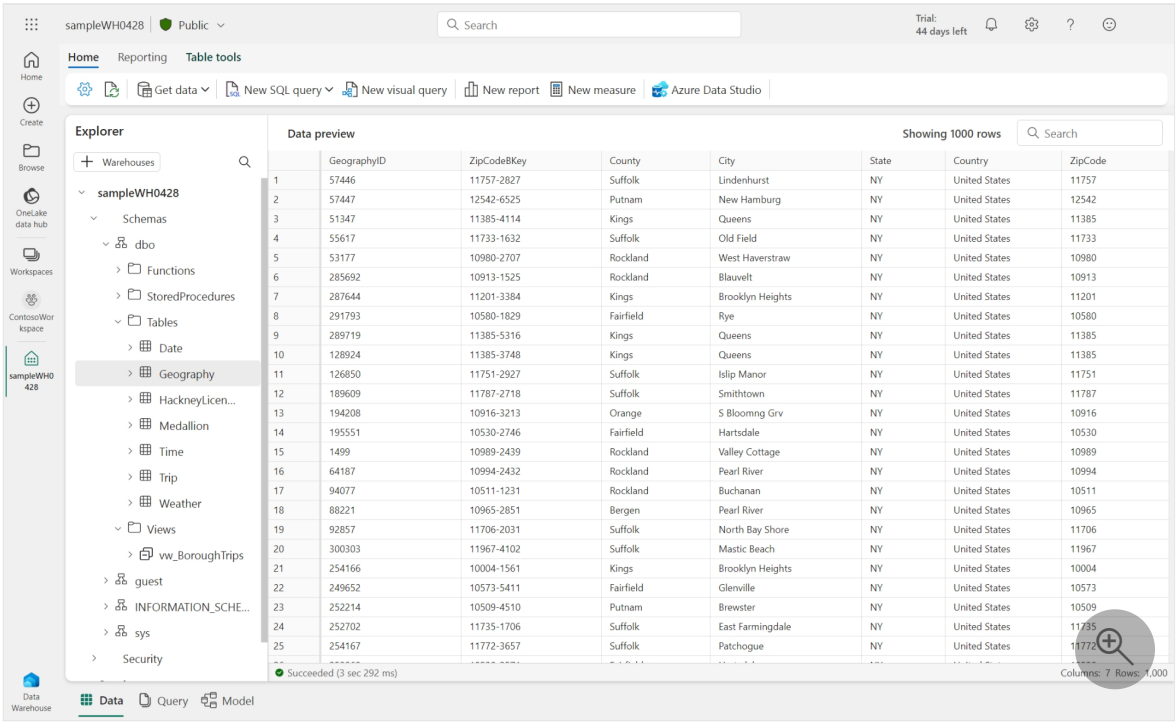
1. Once you have created your warehouse, you can load sample data into warehouse from **Use sample database** card.



2. The data loading takes few minutes to complete.



3. On completion of loading sample data, the warehouse displays data loaded into tables and views to query.



Sample scripts

SQL

```
/*********************************************
Get number of trips performed by each medallion
*********************************************/
```

```
SELECT
    M.MedallionID
    ,M.MedallionCode
    ,COUNT(T.TripDistanceMiles) AS TotalTripCount
FROM
    dbo.Trip AS T
JOIN
    dbo.Medallion AS M
ON
    T.MedallionID=M.MedallionID
GROUP BY
    M.MedallionID
    ,M.MedallionCode

/**************************************************
How many passengers are being picked up on each trip?
**************************************************/
SELECT
    PassengerCount,
    COUNT(*) AS CountOfTrips
FROM
    dbo.Trip
WHERE
    PassengerCount > 0
GROUP BY
    PassengerCount
ORDER BY
    PassengerCount

/**************************************************************************
******
What is the distribution of trips by hour on working days (non-holiday
weekdays)?
**************************************************************************
*****/
SELECT
    ti.HourlyBucket,
    COUNT(*) AS CountOfTrips
FROM dbo.Trip AS tr
INNER JOIN dbo.Date AS d
    ON tr.DateID = d.DateID
INNER JOIN dbo.Time AS ti
    ON tr.PickupTimeID = ti.TimeID
WHERE
    d.IsWeekday = 1
    AND d.IsHolidayUSA = 0
GROUP BY
    ti.HourlyBucket
ORDER BY
    ti.HourlyBucket
```

Next steps

- Query warehouse
- Warehouse settings and context menus

Tables in data warehousing in Microsoft Fabric

Article • 05/23/2023

Applies to: Warehouse in Microsoft Fabric

This article details key concepts for designing tables in Microsoft Fabric.

In tables, data is logically organized in a row-and-column format. Each row represents a unique record, and each column represents a field in the record.

- In Warehouse, tables are database objects that contain all the transactional data.

> **Important**
>
> Microsoft Fabric is currently in PREVIEW. This information relates to a prerelease product that may be substantially modified before it's released. Microsoft makes no warranties, expressed or implied, with respect to the information provided here.

Determine table category

A star schema organizes data into fact and dimension tables. Some tables are used for integration or staging data before moving to a fact or dimension table. As you design a table, decide whether the table data belongs in a fact, dimension, or integration table. This decision informs the appropriate table structure.

- **Fact tables** contain quantitative data that are commonly generated in a transactional system, and then loaded into the data warehouse. For example, a retail business generates sales transactions every day, and then loads the data into a data warehouse fact table for analysis.

- **Dimension tables** contain attribute data that might change but usually changes infrequently. For example, a customer's name and address are stored in a dimension table and updated only when the customer's profile changes. To minimize the size of a large fact table, the customer's name and address don't need to be in every row of a fact table. Instead, the fact table and the dimension table can share a customer ID. A query can join the two tables to associate a customer's profile and transactions.

- **Integration tables** provide a place for integrating or staging data. For example, you can load data to a staging table, perform transformations on the data in staging, and then insert the data into a production table.

A table stores data in OneLake overview as part of the Warehouse. The table and the data persist whether or not a session is open.

Tables in the Warehouse

To show the organization of the tables, you could use `fact`, `dim`, or `int` as prefixes to the table names. The following table shows some of the schema and table names for WideWorldImportersDW sample data warehouse.

WideWorldImportersDW Source Table Name	Table Type	Data Warehouse Table Name
City	Dimension	`wwi.DimCity`
Order	Fact	`wwi.FactOrder`

- Table names are case sensitive.
- Table names can't contain `/` or `\`.

Create a table

For Warehouse, you can create a table as a new empty table. You can also create and populate a table with the results of a select statement. The following are the T-SQL commands for creating a table.

T-SQL Statement	Description
CREATE TABLE	Creates an empty table by defining all the table columns and options.
CREATE TABLE AS SELECT	Populates a new table with the results of a select statement. The table columns and data types are based on the select statement results. To import data, this statement can select from an external table.

This example creates a table with two columns:

SQL

```
CREATE TABLE MyTable (col1 int, col2 int );
```

Schema names

Warehouse supports the creation of custom schemas. Like in SQL Server, schemas are a good way to group together objects that are used in a similar fashion. The following code creates a user-defined schema called `wwi`.

```
SQL

CREATE SCHEMA wwi;
```

Data types

Microsoft Fabric supports the most commonly used T-SQL data types.

- For more about data types, see Data types in Microsoft Fabric.
- When you create a table in Warehouse, review the data types reference in CREATE TABLE (Transact-SQL).
- For a guide to create a table in Warehouse, see Create tables.

Collation

Currently, `Latin1_General_100_BIN2_UTF8` is the default and only supported collation for both tables and metadata.

Statistics

The query optimizer uses column-level statistics when it creates the plan for executing a query. To improve query performance, it's important to have statistics on individual columns, especially columns used in query joins. Warehouse supports automatic creation of statistics.

Statistical updating doesn't happen automatically. Update statistics after a significant number of rows are added or changed. For instance, update statistics after a load. For more information, see Statistics.

Primary key, foreign key, and unique key

For Warehouse, PRIMARY KEY and UNIQUE constraint are only supported when NONCLUSTERED and NOT ENFORCED are both used.

FOREIGN KEY is only supported when NOT ENFORCED is used.

- For syntax, check ALTER TABLE.
- For more information, see Primary keys, foreign keys, and unique keys in Warehouse in Microsoft Fabric.

Align source data with the data warehouse

Warehouse tables are populated by loading data from another data source. To achieve a successful load, the number and data types of the columns in the source data must align with the table definition in the data warehouse.

If data is coming from multiple data stores, you can port the data into the data warehouse and store it in an integration table. Once data is in the integration table, you can use the power of data warehouse to implement transformation operations. Once the data is prepared, you can insert it into production tables.

Limitations

Warehouse supports many, but not all, of the table features offered by other databases.

The following list shows some of the table features that aren't currently supported. During preview, this list is subject to change.

- Computed columns
- Indexed views
- Sequence
- Sparse columns
- Surrogate keys on number sequences with Identity columns
- Synonyms
- Triggers
- Unique indexes
- User-defined types
- Temporary tables

Next steps

- What is data warehousing in Microsoft Fabric?
- What is data engineering in Microsoft Fabric?
- Create a Warehouse
- Query a warehouse

- OneLake overview
- Create tables in Warehouse
- Transactions and modify tables

Data types in Microsoft Fabric

Article • 05/23/2023

Applies to: ✅ SQL Endpoint and Warehouse in Microsoft Fabric

Tables in Microsoft Fabric support the most commonly used T-SQL data types.

- For more information on table creation, see Tables.

> **ⓘ Important**
>
> Microsoft Fabric is currently in PREVIEW. This information relates to a prerelease product that may be substantially modified before it's released. Microsoft makes no warranties, expressed or implied, with respect to the information provided here.

Data types in Warehouse

Warehouse supports a subset of T-SQL data types:

Category	Supported data types
Exact numerics	• bit • bigint • int • smallint • decimal • numeric
Approximate numerics	• float • real
Date and time	• date • datetime2 • time
Character strings	• char • varchar
Binary strings	• uniqueidentifer

> **Note**
>
> The precision for datetime2 and time is limited to 6 digits of precision on fractions of seconds.

The uniqueidentifier data type is a T-SQL data type, without a matching data type in Parquet. As a result, it's stored as a binary type. Warehouse supports storing and reading uniqueidentifier columns, but these values can't be read on the SQL Endpoint. Reading uniqueidentifier values in the lakehouse displays a binary representation of the original values. As a result, features such as cross-joins between Warehouse and SQL Endpoint using a uniqueidentifier column doesn't work as expected.

For more information about the supported data types including their precisions, see data types in CREATE TABLE reference.

Unsupported data types

For T-SQL data types that aren't currently supported, some alternatives are available. Make sure you evaluate the use of these types as precision and query behavior may vary:

Unsupported data type	Alternatives available
money and smallmoney	Use decimal, however note that it can't store the monetary unit.
datetime and smalldatetime	Use datetime2.
nchar and nvarchar	Use char and varchar respectively, as there's no similar unicode data type in Parquet. Char and varchar types in a UTF-8 collation may use more storage than nchar and nvarchar to store unicode data. To understand the impact on your environment, see Storage differences between UTF-8 and UTF-16.
text and ntext	Use varchar.
image	Use varbinary.

Unsupported data types can still be used in T-SQL code for variables, or any in-memory use in session. Creating tables or views that persist data on disk with any of these types isn't allowed.

For a guide to create a table in Warehouse, see Create tables.

Autogenerated data types in the SQL Endpoint

The tables in SQL Endpoint are automatically created whenever a table is created in the associated lakehouse. The column types in the SQL Endpoint tables are derived from the source Delta types.

The rules for mapping original Delta types to the SQL types in SQL Endpoint are shown in the following table:

Delta Data Type	SQL Data Type (Mapped)
Long | BIGINT	bigint
BOOLEAN | BOOL	bit
INT | INTEGER	int
TINYINT | BYTE | SMALLINT | SHORT	smallint
DOUBLE	float
FLOAT | REAL	real
DATE	date
TIMESTAMP	datetime2
CHAR(n)	varchar(n) with `Latin1_General_100_BIN2_UTF8` collation.
STRING | VARCHAR(n)	varchar(n) with `Latin1_General_100_BIN2_UTF8` collation. STRING/VARCHAR(MAX) is mapped to varchar(8000).
BINARY	varbinary(n).
DECIMAL | DEC | NUMERIC	decimal(p,s)

The columns that have the types that aren't listed in the table aren't represented as the table columns in the SQL Endpoint.

Next steps

- T-SQL Surface Area in Microsoft Fabric

T-SQL surface area in Microsoft Fabric

Article • 05/23/2023

Applies to: ✅ SQL Endpoint and Warehouse in Microsoft Fabric

This article covers the T-SQL language syntax capabilities of Microsoft Fabric, when querying the SQL Endpoint or Warehouse.

> **Important**
>
> Microsoft Fabric is currently in PREVIEW. This information relates to a prerelease product that may be substantially modified before it's released. Microsoft makes no warranties, expressed or implied, with respect to the information provided here.

T-SQL surface area

- Creating, altering, and dropping tables, and insert, update, and delete are only supported in Warehouse in Microsoft Fabric, not in the SQL Endpoint of the Lakehouse.
- You can create your own T-SQL views, functions, and procedures on top of the tables that reference your Delta Lake data in the SQL Endpoint of the Lakehouse.
- For more about CREATE/DROP TABLE support, see Tables.
- For more about data types, see Data types.

Limitations

At this time, the following list of commands is NOT currently supported. Don't try to use these commands because even though they may appear to succeed, they could cause issues to your warehouse.

- ALTER TABLE ADD/ALTER/DROP COLUMN
- BULK LOAD
- CREATE ROLE
- CREATE SECURITY POLICY - Row Level Security (RLS)
- CREATE USER
- GRANT/DENY/REVOKE
- Hints
- Identity Columns
- Manually created multi-column stats

- MASK and UNMASK (Dynamic Data Masking)
- MATERIALIZED VIEWS
- MERGE
- OPENROWSET
- PREDICT
- Queries targeting system and user tables
- Recursive queries
- Result Set Caching
- Schema and Table names can't contain / or \
- SELECT - FOR (except JSON)
- SET ROWCOUNT
- SET TRANSACTION ISOLATION LEVEL
- `sp_showmemo_xml`
- `sp_showspaceused`
- `sp_rename`
- Temp Tables
- Triggers
- TRUNCATE

Next steps

- Data types in Microsoft Fabric
- Limitations and known issues in Microsoft Fabric

Primary keys, foreign keys, and unique keys in Warehouse in Microsoft Fabric

Article • 05/23/2023

Applies to: Warehouse in Microsoft Fabric

Learn about table constraints in Warehouse in Microsoft Fabric, including the primary key, foreign keys, and unique keys.

> **Important**
>
> To add or remove primary key, foreign key, or unique constraints, use ALTER TABLE.

> **Important**
>
> Microsoft Fabric is currently in PREVIEW. This information relates to a prerelease product that may be substantially modified before it's released. Microsoft makes no warranties, expressed or implied, with respect to the information provided here.

Table constraints

Warehouse in Microsoft Fabric supports these table constraints:

- PRIMARY KEY is only supported when NONCLUSTERED and NOT ENFORCED are both used.
- UNIQUE constraint is only supported when NONCLUSTERED and NOT ENFORCED is used.
- FOREIGN KEY is only supported when NOT ENFORCED is used.

For syntax, check ALTER TABLE.

- Warehouse doesn't support default constraints at this time.
- For more information on tables, see Tables in data warehousing in Microsoft Fabric.

Remarks

Having primary key, foreign key and/or unique key allows Warehouse in Microsoft Fabric to generate an optimal execution plan for a query.

> **ⓘ Important**
>
> After creating a table with primary key or unique constraint in Warehouse in Microsoft Fabric, users need to make sure all values in those columns are unique. A violation of that may cause the query to return inaccurate result. Foreign keys are not enforced.

This example shows how a query may return inaccurate result if the primary key or unique constraint column includes duplicate values.

SQL

```
 -- Create table t1
CREATE TABLE t1 (a1 INT NOT NULL, b1 INT)

-- Insert values to table t1 with duplicate values in column a1.
INSERT INTO t1 VALUES (1, 100)
INSERT INTO t1 VALUES (1, 1000)
INSERT INTO t1 VALUES (2, 200)
INSERT INTO t1 VALUES (3, 300)
INSERT INTO t1 VALUES (4, 400)

-- Run this query.  No primary key or unique constraint.  4 rows returned.
Correct result.
SELECT a1, COUNT(*) AS total FROM t1 GROUP BY a1

/*
a1          total
----------- -----------
1           2
2           1
3           1
4           1

(4 rows affected)
*/

-- Add unique constraint
ALTER TABLE t1 ADD CONSTRAINT unique_t1_a1 unique NONCLUSTERED (a1) NOT
ENFORCED

-- Re-run this query.  5 rows returned.  Incorrect result.
SELECT a1, count(*) AS total FROM t1 GROUP BY a1

/*
a1          total
----------- -----------
2           1
4           1
1           1
3           1
```

```
1           1

(5 rows affected)
*/

-- Drop unique constraint.
ALTER TABLE t1 DROP CONSTRAINT unique_t1_a1

-- Add primary key constraint
ALTER TABLE t1 add CONSTRAINT PK_t1_a1 PRIMARY KEY NONCLUSTERED (a1) NOT
ENFORCED

-- Re-run this query.  5 rows returned.  Incorrect result.
SELECT a1, COUNT(*) AS total FROM t1 GROUP BY a1

/*
a1          total
----------- -----------
2           1
4           1
1           1
3           1
1           1

(5 rows affected)
*/

-- Manually fix the duplicate values in a1
UPDATE t1 SET a1 = 0 WHERE b1 = 1000

-- Verify no duplicate values in column a1
SELECT * FROM t1

/*
a1          b1
----------- -----------
2           200
3           300
4           400
0           1000
1           100

(5 rows affected)
*/

-- Add unique constraint
ALTER TABLE t1 ADD CONSTRAINT unique_t1_a1 unique NONCLUSTERED (a1) NOT
ENFORCED

-- Re-run this query.  5 rows returned.  Correct result.
SELECT a1, COUNT(*) as total FROM t1 GROUP BY a1

/*
a1          total
----------- -----------
```

```
2           1
3           1
4           1
0           1
1           1

(5 rows affected)
*/

-- Drop unique constraint.
ALTER TABLE t1 DROP CONSTRAINT unique_t1_a1

-- Add primary key contraint
ALTER TABLE t1 ADD CONSTRAINT PK_t1_a1 PRIMARY KEY NONCLUSTERED (a1) NOT
ENFORCED

-- Re-run this query.  5 rows returned.  Correct result.
SELECT a1, COUNT(*) AS total FROM t1 GROUP BY a1

/*
a1          total
----------- -----------
2           1
3           1
4           1
0           1
1           1

(5 rows affected)
*/
```

Examples

Create a Warehouse in Microsoft Fabric table with a primary key:

SQL

```
CREATE TABLE PrimaryKeyTable (c1 INT NOT NULL, c2 INT);

ALTER TABLE PrimaryKeyTable ADD CONSTRAINT PK_PrimaryKeyTable PRIMARY KEY
NONCLUSTERED (c1) NOT ENFORCED;
```

Create a Warehouse in Microsoft Fabric table with a unique constraint:

SQL

```
CREATE TABLE UniqueConstraintTable (c1 INT NOT NULL, c2 INT);
```

```
ALTER TABLE UniqueConstraintTable ADD CONSTRAINT UK_UniqueConstraintTablec1
UNIQUE NONCLUSTERED (c1) NOT ENFORCED;
```

Create a Warehouse in Microsoft Fabric table with a foreign key:

SQL

```
CREATE TABLE ForeignKeyReferenceTable (c1 INT NOT NULL);

ALTER TABLE ForeignKeyReferenceTable ADD CONSTRAINT
PK_ForeignKeyReferenceTable PRIMARY KEY NONCLUSTERED (c1) NOT ENFORCED;

CREATE TABLE ForeignKeyTable (c1 INT NOT NULL, c2 INT);

ALTER TABLE ForeignKeyTable ADD CONSTRAINT FK_ForeignKeyTablec1 FOREIGN KEY
(c1) REFERENCES ForeignKeyReferenceTable (c1) NOT ENFORCED;
```

Next steps

- Design tables in Warehouse in Microsoft Fabric
- Data types in Microsoft Fabric
- What is data warehousing in Microsoft Fabric?
- What is data engineering in Microsoft Fabric?
- Warehouse in Microsoft Fabric
- Create a Warehouse
- Query a warehouse

Transactions in Warehouse tables in Microsoft Fabric

Article • 05/23/2023

Applies to: ✅ SQL Endpoint and Warehouse in Microsoft Fabric

Similar to their behavior in SQL Server, transactions allow you to control the commit or rollback of read and write queries.

You can modify data that is stored in tables in a Warehouse using transactions to group changes together.

- For example, you could commit inserts to multiples tables, or, none of the tables if an error arises. If you're changing details about a purchase order that affects three tables, you can group those changes into a single transaction. That means when those tables are queried, they either all have the changes or none of them do. Transactions are a common practice for when you need to ensure your data is consistent across multiple tables.

> **ⓘ Important**
>
> Microsoft Fabric is currently in PREVIEW. This information relates to a prerelease product that may be substantially modified before it's released. Microsoft makes no warranties, expressed or implied, with respect to the information provided here.

Transactional capabilities

The same transactional capabilities are supported in the SQL Endpoint in Microsoft Fabric, but for read-only queries.

Transactions can also be used for sequential SELECT statements to ensure the tables involved all have data from the same point in time. As an example, if a table has new rows added by another transaction, the new rows don't affect the SELECT queries inside an open transaction.

> **ⓘ Important**
>
> Only the snapshot isolation level is supported in Microsoft Fabric. If you use T-SQL to change your isolation level, the change is ignored at Query Execution time and

snapshot isolation is applied.

Cross-database query transaction support

Warehouse in Microsoft Fabric supports transactions that span across databases that are within the same workspace including reading from the SQL Endpoint of the Lakehouse. Every Lakehouse has one SQL Endpoint and each workspace can have more than one lakehouse.

DDL support within transactions

Warehouse in Microsoft Fabric supports DDL such as CREATE TABLE inside user-defined transactions.

Locks for different types of statements

This table provides a list of what locks are used for different types of transactions, all locks are at the table level:

Statement type	Lock taken
SELECT	Schema-Stability (Sch-S)
INSERT	Intent Exclusive (IX)
DELETE	Intent Exclusive (IX)
UPDATE	Intent Exclusive (IX)
COPY INTO	Intent Exclusive (IX)
DDL	Schema-Modification (Sch-M)

These locks prevent conflicts such as a table's schema being changed while rows are being updated in a transaction.

You can query locks currently held with the dynamic management view (DMV) sys.dm_tran_locks.

Conflicts from two or more concurrent transactions that update one or more rows in a table are evaluated at the end of the transaction. The first transaction to commit completes successfully and the other transactions are rolled back with an error returned. These conflicts are evaluated at the table level and not the individual parquet file level.

INSERT statements always create new parquet files, which means fewer conflicts with other transactions except for DDL because the table's schema could be changing.

Transaction logging

Transaction logging in Warehouse in Microsoft Fabric is at the parquet file level because parquet files are immutable (they can't be changed). A rollback results in pointing back to the previous parquet files. The benefits of this change are that transaction logging and rollbacks are faster.

Limitations

- Distributed transactions are not supported.
- Save points are not supported.
- Named transactions are not supported.
- Marked transactions are not supported.
- At this time, there's limited T-SQL functionality in the warehouse. See TSQL surface area for a list of T-SQL commands that are currently not available.
- If a transaction has data insertion into an empty table and issues a SELECT before rolling back, the automatically generated statistics may still reflect the uncommitted data, causing inaccurate statistics. Inaccurate statistics can lead to unoptimized query plans and execution times. If you roll back a transaction with SELECTs after a large INSERT, you may want to update statistics for the columns mentioned in your SELECT.

Next steps

- Query the Warehouse
- Tables in Warehouse

Warehouse settings and context menus

Article • 05/23/2023

Applies to: ✅ SQL Endpoint and Warehouse in Microsoft Fabric

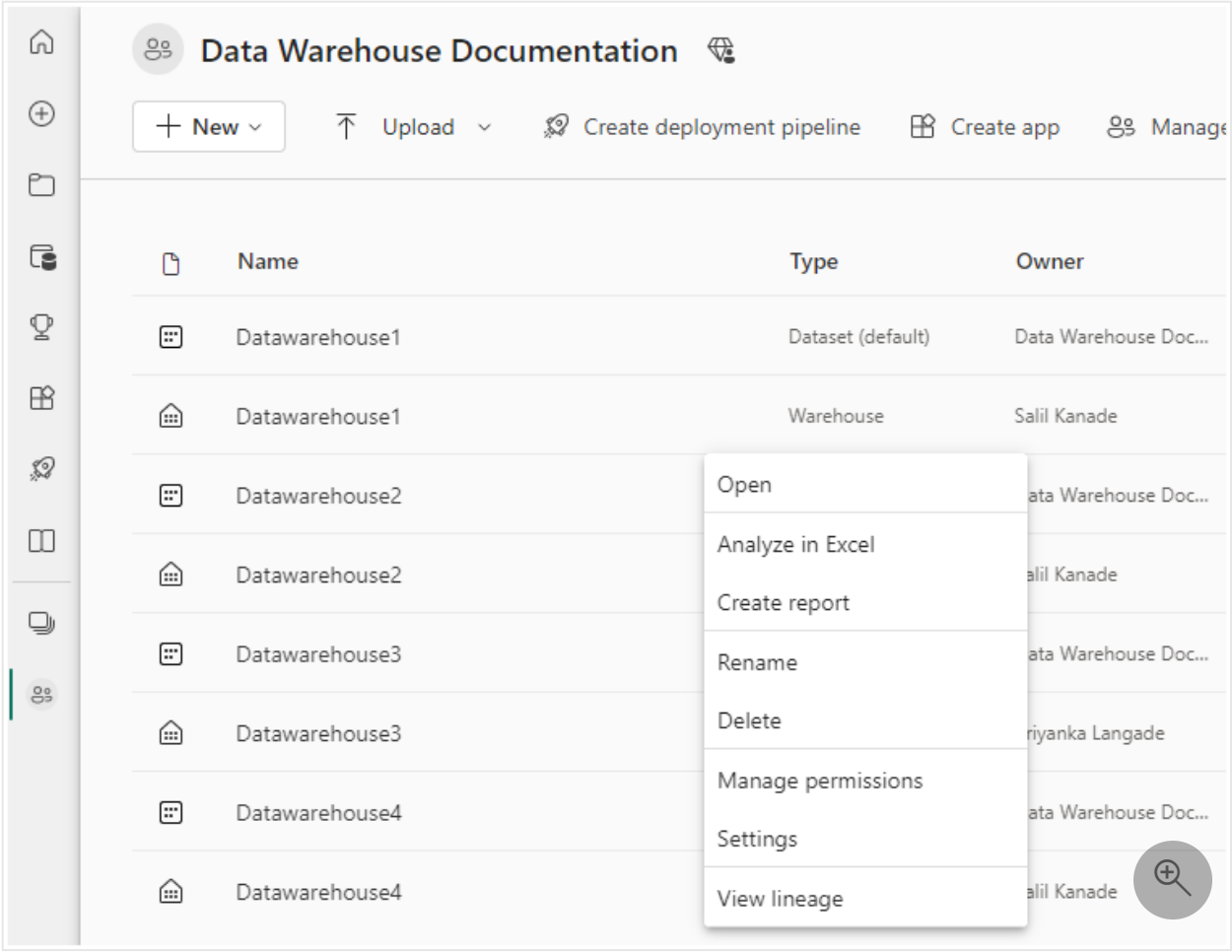
Settings are accessible from the context menu or from the Settings icon in the ribbon when you open the item. There are some key differences in the actions you can take in settings depending on if you're interacting with the SQL Endpoint or a data warehouse.

> **ⓘ Important**
>
> Microsoft Fabric is currently in PREVIEW. This information relates to a prerelease product that may be substantially modified before it's released. Microsoft makes no warranties, expressed or implied, with respect to the information provided here.

Settings options

This section describes and explains the settings options available based on the item you're working with and its description.

The following image shows the warehouse settings menu.



The following table is a list of settings available for each warehouse.

Setting	Detail	Editable for
Name	Lets user read/edit name of the warehouse.	Warehouse
Warehouse description	Lets users add metadata details to provide descriptive information about a warehouse.	Warehouse
Owned by	Name of the user who owns the warehouse.	
Last modified by	Name of the user who modified the warehouse recently.	
SQL connection string	The SQL connection string for the workspace. You can use the SQL connection string to create a connection to the warehouse using various tools, such as SSMS/Azure Data Studio.	

The following table shows settings for the default Power BI dataset.

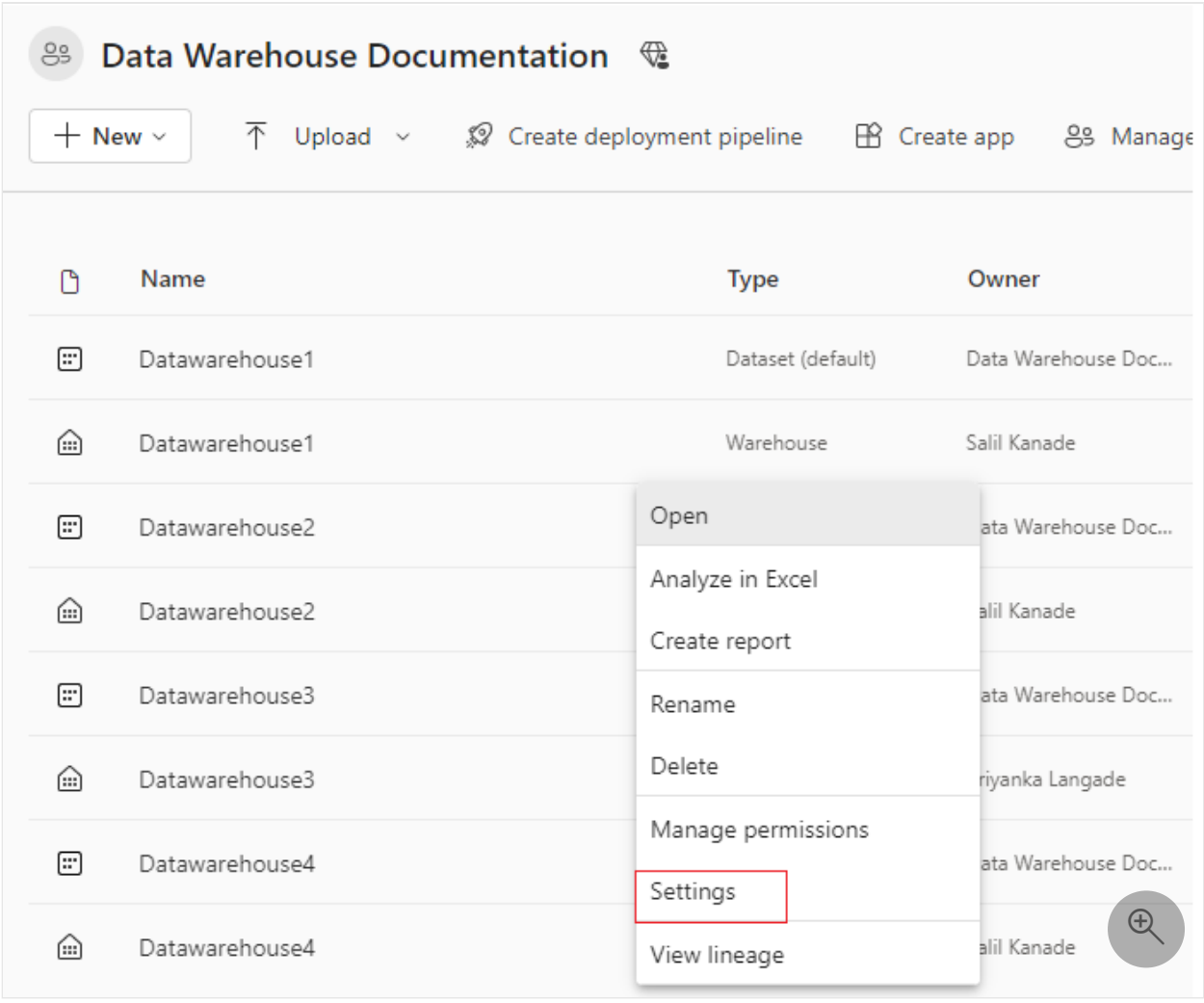
Setting	Details
Request access	Request access to the default Power BI dataset.

Setting	Details
Q&A	Use natural language to ask question on your data.
Query caching	Turn on or off caching query results for speeding up reports by using previously saved query results.
Server settings	The XMLA connection string of the default dataset.
Endorsement and discovery	Endorse default dataset and make it discoverable in your org.

Context menus

Applies to: ✅ Warehouse in Microsoft Fabric

Warehouse offers an easy experience to create reports and access supported actions using its context menus.



The following table describes the warehouse context menu options:

Menu option	Option description
Open	Opens warehouse to explore and analyze data.
Open with	Opens warehouse in Azure Data Studio.
Share	Lets users share the warehouse to build content based on the underlying default Power BI dataset, query data using SQL or get access to underlying data files. Shares the warehouse access (SQL- connect only, and autogenerated dataset) with other users in your organization. Users receive an email with links to access the detail page where they can find the SQL connection string and can access the default dataset to create reports based on it.
Analyze in Excel	Uses the existing Analyze in Excel capability on default Power BI dataset. Learn more: Analyze in Excel.
Create report	Build a report in DirectQuery mode. Learn more: Get started creating in the Power BI service
Rename	Updates the warehouse with the new name. Does not apply to (Lakehouse) SQL endpoint.
Delete	Delete warehouse from workspace. A confirmation dialog notifies you of the impact of the delete action. If the Delete action is confirmed, then the warehouse and related downstream items are deleted. Does not apply to (Lakehouse) SQL endpoint
Manage permissions	Enables users to add other recipients with specified permissions, similar to allowing the sharing of an underlying dataset or allowing to build content with the data associated with the underlying dataset.
Settings	Learn more about warehouse settings in the previous section.
View lineage	This option shows the end-to-end lineage of warehouse from the data sources to the warehouse, the default Power BI dataset, and other datasets (if any) that were built on top of the warehouse, all the way to deports, dashboards and apps.
View details	Opens up warehouse details in Data hub.

Next steps

- Warehouse in Microsoft Fabric
- Data modeling in the default Power BI dataset
- Create reports in the Power BI service
- Admin portal

Ingest data into the Warehouse

Article • 05/23/2023

Applies to: Warehouse in Microsoft Fabric

Warehouse in Microsoft Fabric offers built-in data ingestion tools that allow users to ingest data into warehouses at scale using code-free or code-rich experiences.

> **Important**
>
> Microsoft Fabric is currently in PREVIEW. This information relates to a prerelease product that may be substantially modified before it's released. Microsoft makes no warranties, expressed or implied, with respect to the information provided here.

Data ingestion options

You can ingest data into a Warehouse using one of the following options:

- **COPY (Transact-SQL)**: the COPY statement offers flexible, high-throughput data ingestion from an external Azure storage account. You can use the COPY statement as part of your existing ETL/ELT logic in Transact-SQL code.
- **Data pipelines**: pipelines offer a code-free or low-code experience for data ingestion. Using pipelines, you can orchestrate robust workflows for a full Extract, Transform, Load (ETL) experience that includes activities to help prepare the destination environment, run custom Transact-SQL statements, perform lookups, or copy data from a source to a destination.
- **Dataflows**: an alternative to pipelines, dataflows enable easy data preparation, cleaning, and transformation using a code-free experience.
- **Cross-warehouse ingestion**: data ingestion from workspace sources is also possible. This scenario may be required when there's the need to create a new table with a subset of a different table, or as a result of joining different tables in the warehouse and in the lakehouse. For cross-warehouse ingestion, in addition to the options mentioned, Transact-SQL features such as **INSERT...SELECT**, **SELECT INTO**, or **CREATE TABLE AS SELECT (CTAS)** work cross-warehouse within the same workspace.

Decide which data ingestion tool to use

To decide which data ingestion option to use, you can use the following criteria:

- Use the **COPY (Transact-SQL)** statement for code-rich data ingestion operations, for the highest data ingestion throughput possible, or when you need to add data ingestion as part of a Transact-SQL logic. For syntax, see COPY INTO (Transact-SQL).
- Use **data pipelines** for code-free or low-code, robust data ingestion workflows that run repeatedly, at a schedule, or that involves large volumes of data. For more information, see Ingest data using Data pipelines.
- Use **dataflows** for a code-free experience that allow custom transformations to source data before it's ingested. These transformations include (but aren't limited to) changing data types, adding or removing columns, or using functions to produce calculated columns. For more information, see Dataflows.
- Use **cross-warehouse ingestion** for code-rich experiences to create new tables with source data within the same workspace. For more information, see Ingest data using Transact-SQL and Write a cross-database query.

> ⓘ **Note**
>
> The COPY statement in Warehouse supports only data sources on Azure storage accounts, with authentication using to Shared Access Signature (SAS), Storage Account Key (SAK), or accounts with public access. For other limitations, see **COPY (Transact-SQL)**.

Supported data formats and sources

Data ingestion for Warehouse in Microsoft Fabric offers a vast number of data formats and sources you can use. Each of the options outlined includes its own list of supported data connector types and data formats.

For **cross-warehouse ingestion**, data sources must be within the same Microsoft Fabric workspace. Queries can be performed using three-part naming for the source data.

As an example, suppose there's two warehouses named Inventory and Sales in a workspace. A query such as the following one creates a new table in the Inventory warehouse with the content of a table in the Inventory warehouse, joined with a table in the Sales warehouse:

SQL

```
CREATE TABLE Inventory.dbo.RegionalSalesOrders
AS
SELECT s.SalesOrders, i.ProductName
FROM Sales.dbo.SalesOrders s
```

```
JOIN Inventory.dbo.Products i
WHERE s.ProductID = i.ProductID
    AND s.Region = 'West region'
```

The COPY (Transact-SQL) statement currently supports the PARQUET and CSV file formats. For data sources, currently Azure Data Lake Storage (ADLS) Gen2 and Azure Blob Storage are supported.

Data pipelines and **dataflows** support a wide variety of data sources and data formats. For more information, see Data pipelines and Dataflows.

Best practices

The COPY command feature in Warehouse in Microsoft Fabric uses a simple, flexible, and fast interface for high-throughput data ingestion for SQL workloads. In the current version, we support loading data from external storage accounts only.

You can also use TSQL to create a new table and then insert into it, and then update and delete rows of data. Data can be inserted from any database within the Microsoft Fabric workspace using cross-database queries. If you want to ingest data from a Lakehouse to a warehouse, you can do this with a cross database query. For example:

SQL

```
INSERT INTO MyWarehouseTable
SELECT * FROM MyLakehouse.dbo.MyLakehouseTable;
```

- Avoid ingesting data using singleton **INSERT** statements, as this causes poor performance on queries and updates. If singleton **INSERT** statements were used for data ingestion consecutively, we recommend creating a new table by using **CREATE TABLE AS SELECT (CTAS)** or **INSERT...SELECT** patterns, dropping the original table, and then creating your table again from the table you created using **CREATE TABLE AS SELECT (CTAS)** or **INSERT...SELECT**.
- When working with external data on files, we recommend that files are at least 4 MB in size.
- For large compressed CSV files, consider splitting your file into multiple files.
- Azure Data Lake Storage (ADLS) Gen2 offers better performance than Azure Blob Storage (legacy). Consider using an ADLS Gen2 account whenever possible.
- For pipelines that run frequently, consider isolating your Azure storage account from other services that could access the same files at the same time.
- Explicit transactions allow you to group multiple data changes together so that they're only visible when reading one or more tables when the transaction is fully

committed. You also have the ability to roll back the transaction if any of the changes fail.

- If a SELECT is within a transaction, and was preceded by data insertions, the automatically generated statistics may be inaccurate after a rollback. Inaccurate statistics can lead to unoptimized query plans and execution times. If you roll back a transaction with SELECTs after a large INSERT, you may want to update statistics for the columns mentioned in your SELECT.

Next steps

- Ingest data using Data pipelines
- Ingest data using the COPY statement
- Ingest data using Transact-SQL
- Create your first dataflow to get and transform data
- COPY (Transact-SQL)
- CREATE TABLE AS SELECT (Transact-SQL)
- INSERT (Transact-SQL)

Ingest data into your Warehouse using data pipelines

Article • 05/23/2023

Applies to: Warehouse in Microsoft Fabric

Data pipelines offer an alternative to using the COPY command through a graphical user interface. A data pipeline is a logical grouping of activities that together perform a data ingestion task. Pipelines allow you to manage extract, transform, and load (ETL) activities instead of managing each one individually.

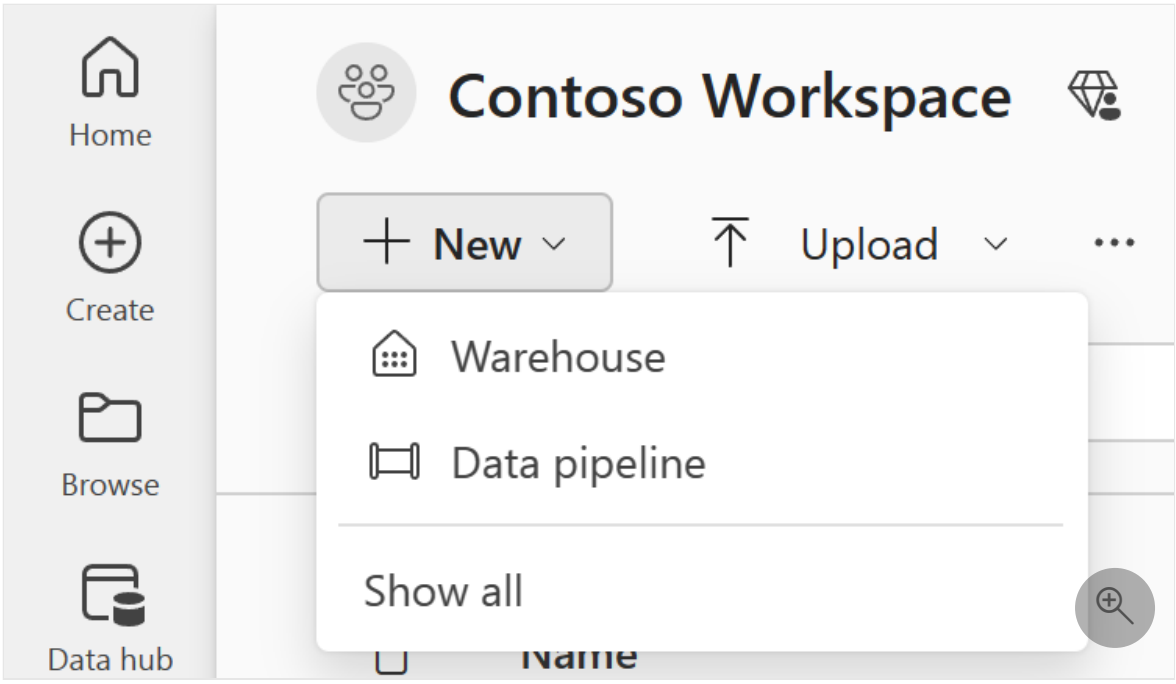
In this tutorial, you'll create a new pipeline that loads sample data into a Warehouse in Microsoft Fabric.

> **Note**
>
> Some features from Azure Data Factory are not available in Microsoft Fabric, but the concepts are interchangeable. You can learn more about Azure Data Factory and Pipelines on **Pipelines and activities in Azure Data Factory and Azure Synapse Analytics**. For a quickstart, visit **Quickstart: Create your first pipeline to copy data**.

> **Important**
>
> Microsoft Fabric is currently in PREVIEW. This information relates to a prerelease product that may be substantially modified before it's released. Microsoft makes no warranties, expressed or implied, with respect to the information provided here.

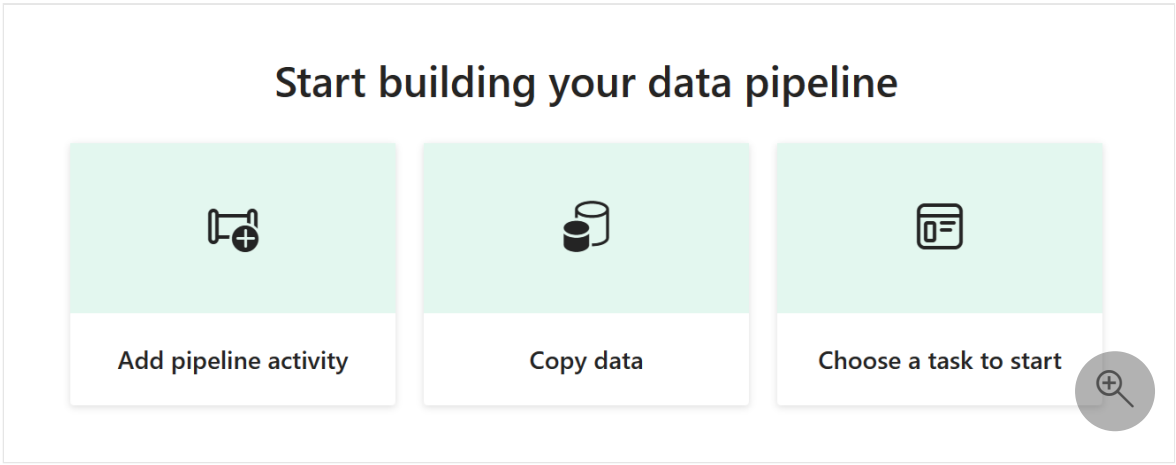
Create a data pipeline

1. To create a new pipeline navigate to your workspace, select the **+New** button, and select **Data pipeline**.



2. In the **New pipeline** dialog, provide a name for your new pipeline and select **Create**.

3. You'll land in the pipeline canvas area, where you see three options to get started: **Add a pipeline activity**, **Copy data**, and **Choose a task to start**.

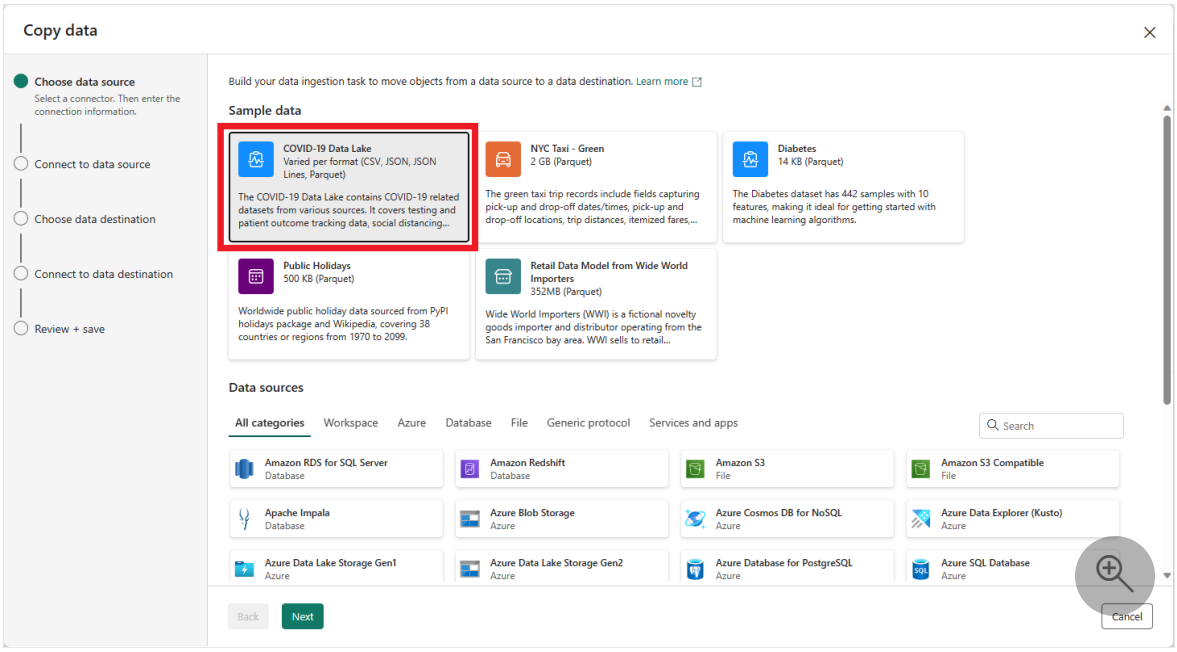


Each of these options offers different alternatives to create a pipeline:

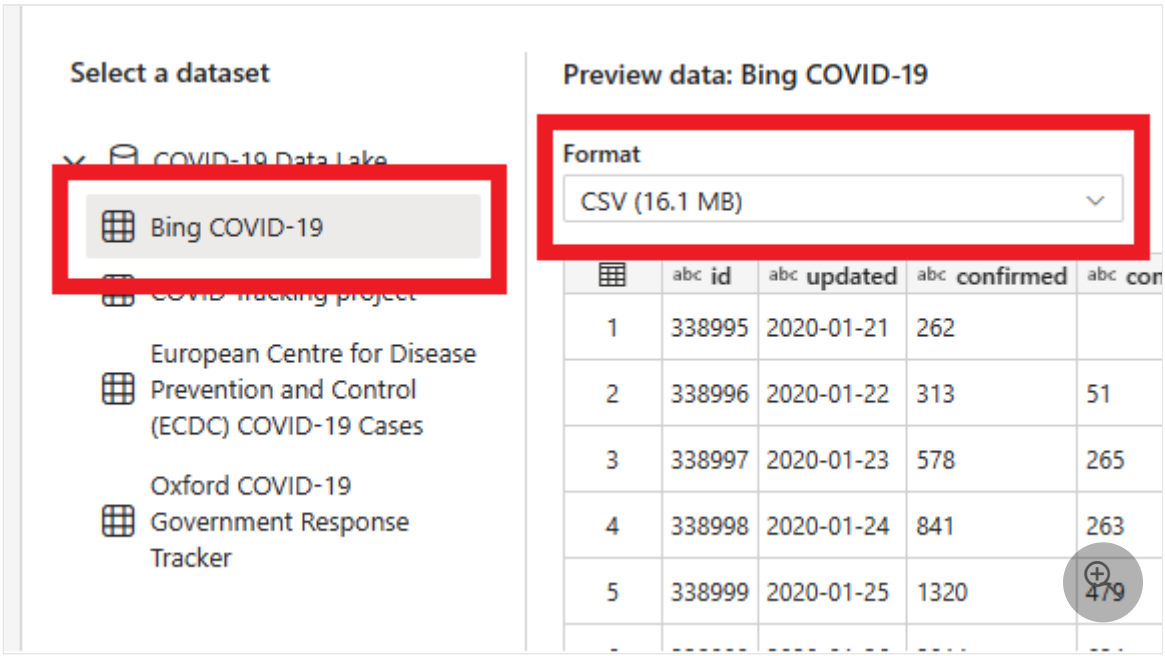
- **Add pipeline activity**: this option launches the pipeline editor, where you can create new pipelines from scratch by using pipeline activities.
- **Copy data**: this option launches a step-by-step assistant that helps you select a data source, a destination, and configure data load options such as the column mappings. On completion, it creates a new pipeline activity with a **Copy Data** task already configured for you.
- **Choose a task to start**: this option launches a set of predefined templates to help get you started with pipelines based on different scenarios.

Pick the **Copy data** option to launch the **Copy assistant**.

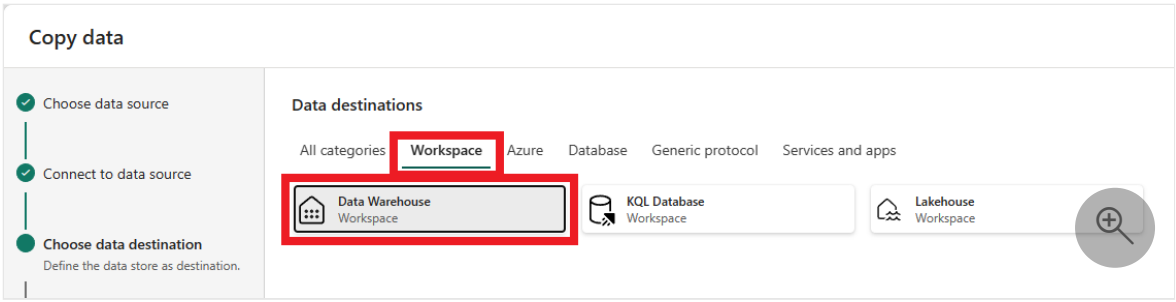
4. The first page of the **Copy data** assistant helps you pick your own data from various data sources, or select from one of the provided samples to get started. For this tutorial, we'll use the **COVID-19 Data Lake** sample. Select this option and select **Next**.



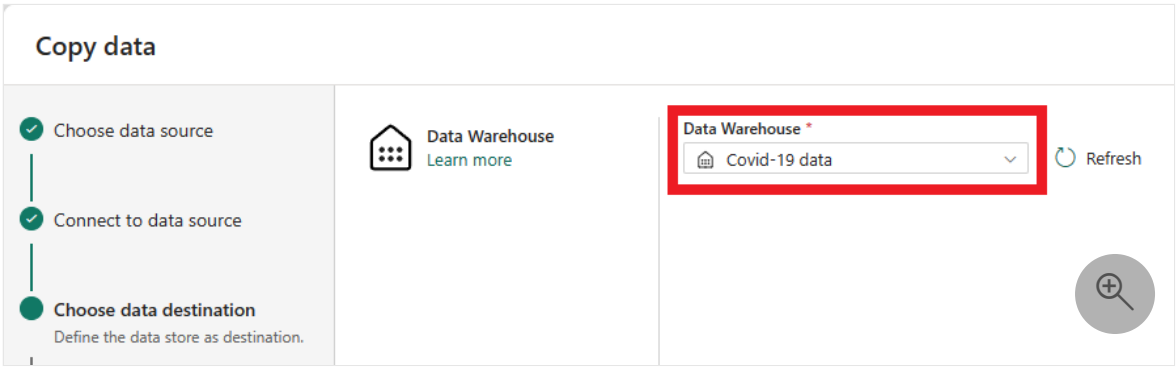
5. In the next page, you can select a dataset, the source file format, and preview the selected dataset. Select the **Bing COVID-19** dataset, the **CSV** format, and select **Next**.



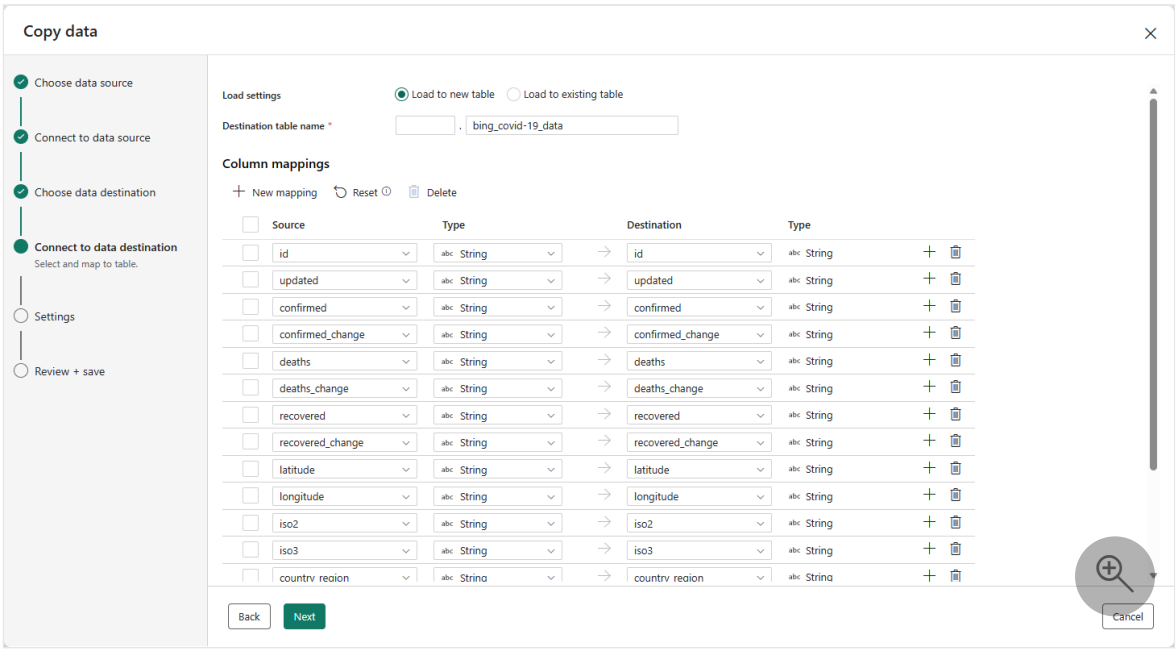
6. The next page, **Data destinations**, allows you to configure the type of the destination dataset. We'll load data into a warehouse in our workspace, so select the **Warehouse** tab, and the **Data Warehouse** option. Select **Next**.



7. Now it's time to pick the warehouse to load data into. Select your desired warehouse in the dropdown box and select **Next**.



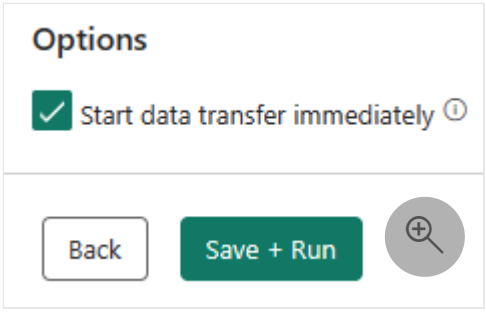
8. The last step to configure the destination is to provide a name to the destination table and configure the column mappings. Here you can choose to load the data to a new table or to an existing one, provide a schema and table names, change column names, remove columns, or change their mappings. You can accept the defaults, or adjust the settings to your preference.



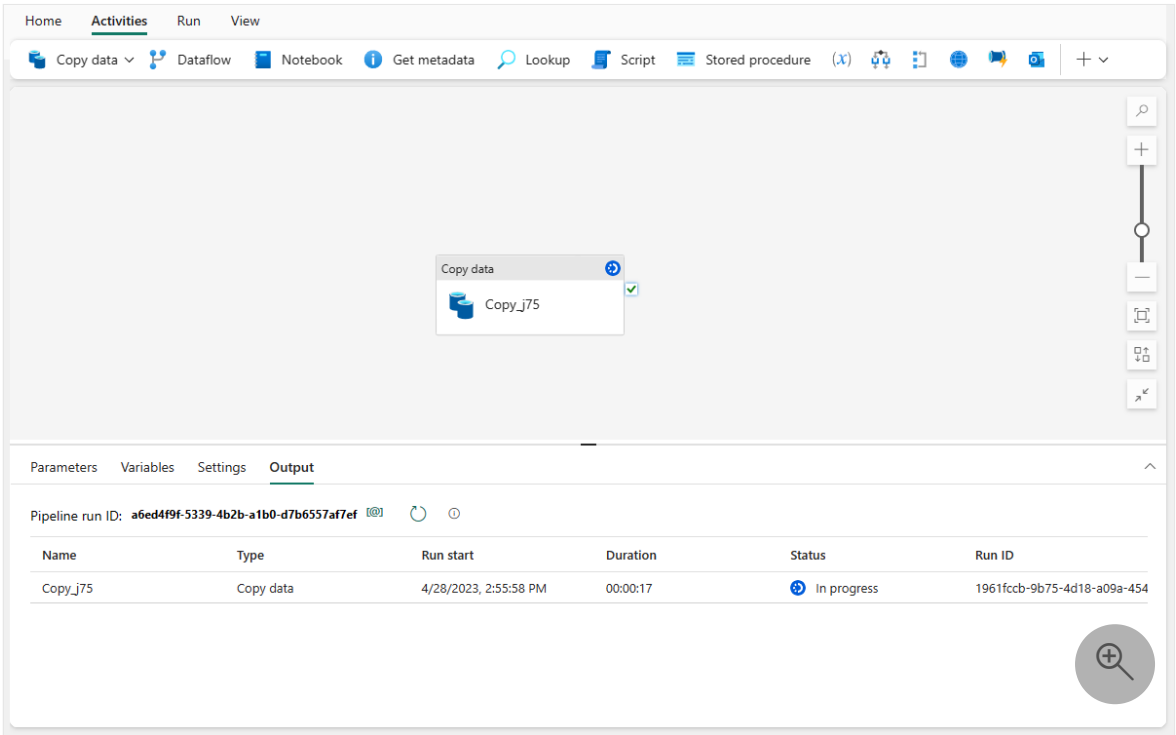
When you're done reviewing the options, select **Next**.

9. The next page gives you the option to use staging, or provide advanced options for the data copy operation (which uses the T-SQL COPY command). Review the options without changing them and select **Next**.

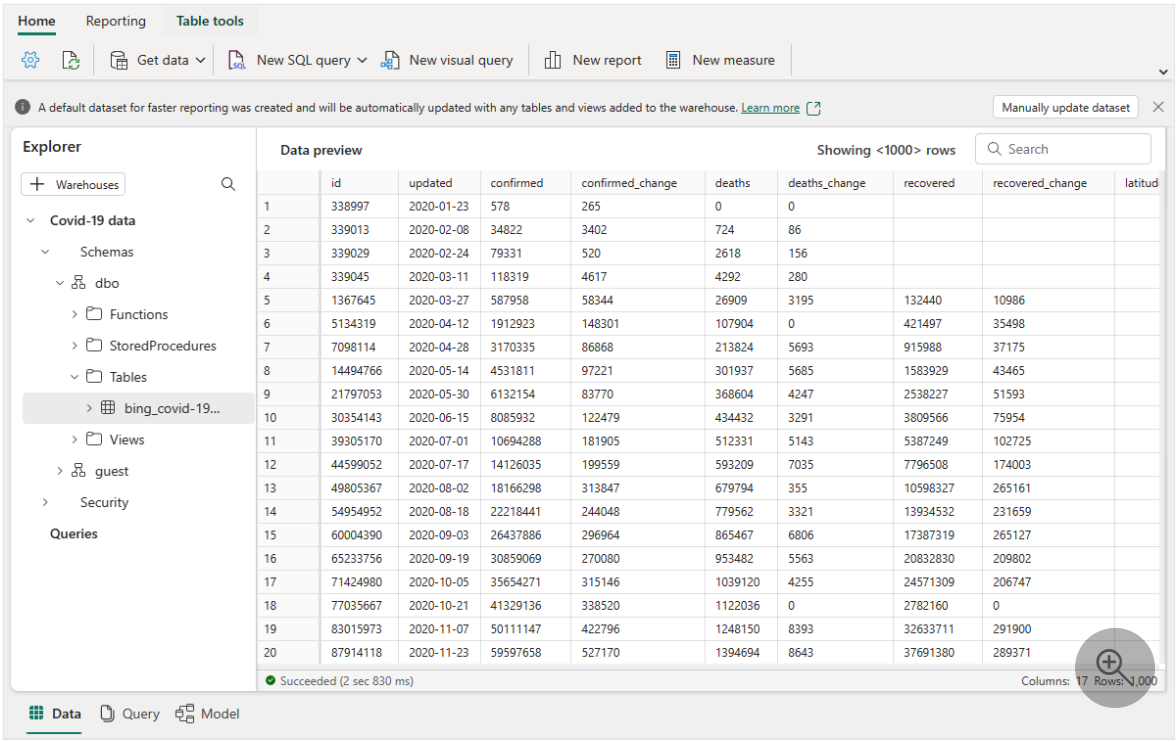
10. The last page in the assistant offers a summary of the copy activity. Select the option **Start data transfer immediately** and select **Save + Run**.



11. You are directed to the pipeline canvas area, where a new Copy Data activity is already configured for you. The pipeline starts to run automatically. You can monitor the status of your pipeline in the **Output** pane:



12. After a few seconds, your pipeline finishes successfully. Navigating back to your warehouse, you can select your table to preview the data and confirm that the copy operation concluded.



For more on data ingestion into your Warehouse in Microsoft Fabric, visit:

- Ingesting data into the Warehouse
- Ingest data into your Warehouse using the COPY statement
- Ingest data into your Warehouse using Transact-SQL

Next steps

Query the SQL Endpoint or Warehouse in Microsoft Fabric

Ingest data into your Warehouse using the COPY statement

Article • 05/23/2023

Applies to: Warehouse in Microsoft Fabric

The COPY statement is the primary way to ingest data into Warehouse tables. COPY performs high high-throughput data ingestion from an external Azure storage account, with the flexibility to configure source file format options, a location to store rejected rows, skipping header rows, and other options.

This tutorial shows data ingestion examples for a Warehouse table using the T-SQL COPY statement. It uses the Bing COVID-19 sample data from the Azure Open Datasets. For details about this dataset, including its schema and usage rights, see Bing COVID-19.

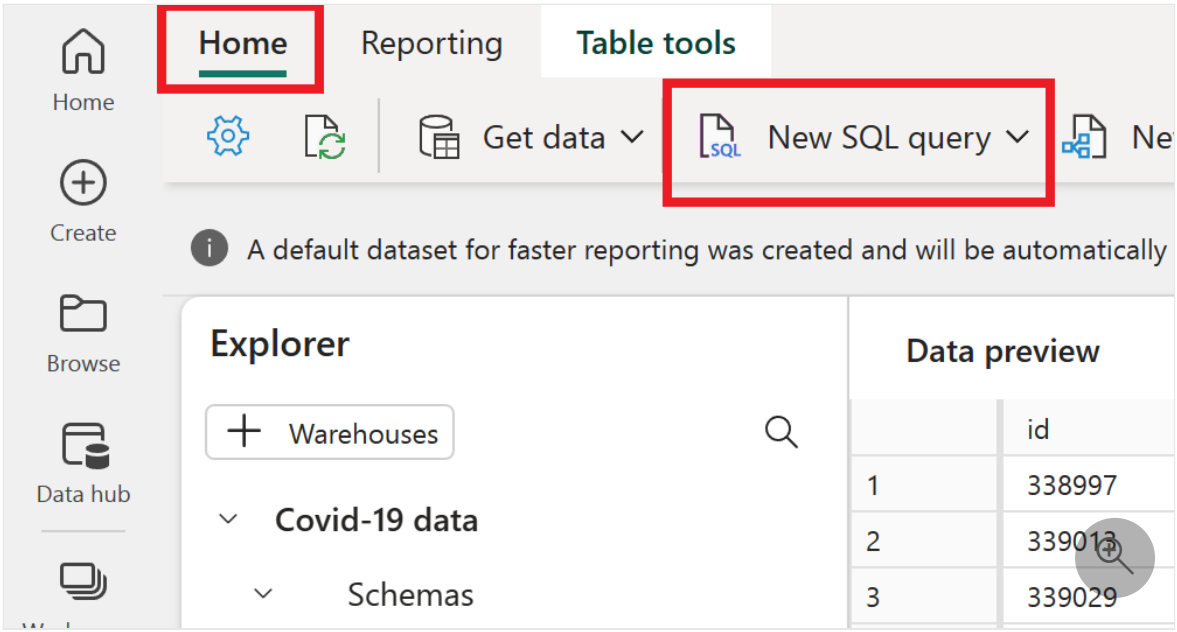
> **Note**
>
> To learn more about the T-SQL COPY statement including more examples and the full syntax, see **COPY (Transact-SQL)**.

> **Important**
>
> Microsoft Fabric is currently in PREVIEW. This information relates to a prerelease product that may be substantially modified before it's released. Microsoft makes no warranties, expressed or implied, with respect to the information provided here.

Create a table

Before you use the COPY statement, the destination table needs to be created. To create the destination table for this sample, use the following steps:

1. In your Microsoft Fabric workspace, find and open your warehouse.
2. Switch to the **Home** tab and select **New SQL query**.



3. To create the table used as the destination in this tutorial, run the following code:

```SQL
CREATE TABLE [dbo].[bing_covid-19_data]
(
    [id] [int] NULL,
    [updated] [date] NULL,
    [confirmed] [int] NULL,
    [confirmed_change] [int] NULL,
    [deaths] [int] NULL,
    [deaths_change] [int] NULL,
    [recovered] [int] NULL,
    [recovered_change] [int] NULL,
    [latitude] [float] NULL,
    [longitude] [float] NULL,
    [iso2] [varchar](8000) NULL,
    [iso3] [varchar](8000) NULL,
    [country_region] [varchar](8000) NULL,
    [admin_region_1] [varchar](8000) NULL,
    [iso_subdivision] [varchar](8000) NULL,
    [admin_region_2] [varchar](8000) NULL,
    [load_time] [datetime2](6) NULL
);
```

Ingest Parquet data using the COPY statement

In the first example, we load data using a Parquet source. Since this dataset is publicly available and doesn't require authentication, you can easily copy this data by specifying the source and the destination. No authentication details are needed. You'll only need to specify the `FILE_TYPE` argument.

Use the following code to run the COPY statement with a Parquet source:

```
SQL

COPY INTO [dbo].[bing_covid-19_data]
FROM 'https://pandemicdatalake.blob.core.windows.net/public/curated/covid-19/bing_covid-19_data/latest/bing_covid-19_data.parquet'
WITH (
    FILE_TYPE = 'PARQUET'
);
```

Ingest CSV data using the COPY statement and skipping a header row

It's common for comma-separated value (CSV) files to have a header row that provides the column names representing the table in a CSV file. The COPY statement can copy data from CSV files and skip one or more rows from the source file header.

If you ran the previous example to load data from Parquet, consider deleting all data from your table:

```
SQL

DELETE FROM [dbo].[bing_covid-19_data];
```

To load data from a CSV file skipping a header row, use the following code:

```
SQL

COPY INTO [dbo].[bing_covid-19_data]
FROM 'https://pandemicdatalake.blob.core.windows.net/public/curated/covid-19/bing_covid-19_data/latest/bing_covid-19_data.csv'
WITH (
    FILE_TYPE = 'CSV',
    FIRSTROW = 2
);
```

Checking the results

The COPY statement completes by ingesting 4,766,736 rows into your new table. You can confirm the operation ran successfully by running a query that returns the total number of rows in your table:

```
```

SQL

```
SELECT COUNT(*) FROM [dbo].[bing_covid-19_data];
```

If you ran both examples without deleting the rows in between runs, you'll see the result of this query with twice as many rows. While that works for data ingestion in this case, consider deleting all rows and ingesting data only once if you're going to further experiment with this data.

Next steps

- Ingest data using Data pipelines
- Ingest data into your Warehouse using Transact-SQL
- Ingesting data into the Warehouse

Ingest data into your Warehouse using Transact-SQL

Article • 05/23/2023

Applies to: Warehouse in Microsoft Fabric

The Transact-SQL language offers options you can use to load data at scale from existing tables in your lakehouse and warehouse into new tables in your warehouse. These options are convenient if you need to create new versions of a table with aggregated data, versions of tables with a subset of the rows, or to create a table as a result of a complex query. Let's explore some examples.

> **Important**
>
> Microsoft Fabric is currently in PREVIEW. This information relates to a prerelease product that may be substantially modified before it's released. Microsoft makes no warranties, expressed or implied, with respect to the information provided here.

Creating a new table with the result of a query by using CREATE TABLE AS SELECT (CTAS)

The **CREATE TABLE AS SELECT (CTAS)** statement allows you to create a new table in your warehouse from the output of a `SELECT` statement. It runs the ingestion operation into the new table in parallel, making it highly efficient for data transformation and creation of new tables in your workspace.

> **Note**
>
> The examples in this article use the Bing COVID-19 sample dataset. To load the sample dataset, follow the steps in **Ingest data into your Warehouse using the COPY statement** to create the sample data into your warehouse.

The first example illustrates how to create a new table that is a copy of the existing `dbo.[bing_covid-19_data_2023]` table, but filtered to data from the year 2023 only:

```
SQL
```

```
CREATE TABLE [dbo].[bing_covid-19_data_2023]
AS
SELECT *
FROM [dbo].[bing_covid-19_data]
WHERE DATEPART(YEAR,[updated]) = '2023';
```

You can also create a new table with new `year`, `month`, `dayofmonth` columns, with values obtained from `updated` column in the source table. This can be useful if you're trying to visualize infection data by year, or to see months when the most COVID-19 cases are observed:

SQL

```
CREATE TABLE [dbo].[bing_covid-19_data_with_year_month_day]
AS
SELECT DATEPART(YEAR,[updated]) [year], DATEPART(MONTH,[updated]) [month],
DATEPART(DAY,[updated]) [dayofmonth], *
FROM [dbo].[bing_covid-19_data];
```

As another example, you can create a new table that summarizes the number of cases observed in each month, regardless of the year, to evaluate how seasonality may affect spread in a given country/region. It uses the table created in the previous example with the new `month` column as a source:

SQL

```
CREATE TABLE [dbo].[infections_by_month]
AS
SELECT [country_region],[month], SUM(CAST(confirmed as bigint))
[confirmed_sum]
FROM [dbo].[bing_covid-19_data_with_year_month_day]
GROUP BY [country_region],[month];
```

Based on this new table, we can see that the United States observed more confirmed cases across all years in the month of `January`, followed by `December` and `October`. `April` is the month with the lowest number of cases overall:

SQL

```
SELECT * FROM [dbo].[infections_by_month]
WHERE [country_region] = 'United States'
ORDER BY [confirmed_sum] DESC;
```

Messages | Results | Save as table | Download Excel file | Visualize results

	country_region	month	confirmed_sum
1	United States	1	16511858227
2	United States	12	14042979570
3	United States	10	12951531137
4	United States	11	12924724969
5	United States	9	11694012835
6	United States	8	11405942208
7	United States	2	11210892235
8	United States	7	10806680093
9	United States	5	10342580051
10	United States	6	9863022596
11	United States	3	9742104939
12	United States	4	9413291389

For more examples and syntax reference, see CREATE TABLE AS SELECT (Transact-SQL).

Ingesting data into existing tables with T-SQL queries

The previous examples create new tables based on the result of a query. To replicate the examples but on existing tables, the **INSERT...SELECT** pattern can be used. For example, the following code ingests new data into an existing table:

SQL

```
INSERT INTO [dbo].[bing_covid-19_data_2023]
SELECT * FROM [dbo].[bing_covid-19_data]
WHERE [updated] > '2023-02-28';
```

The query criteria for the `SELECT` statement can be any valid query, as long as the resulting query column types align with the columns on the destination table. If column names are specified and include only a subset of the columns from the destination table, all other columns are loaded as `NULL`. For more information, see Using INSERT INTO...SELECT to Bulk Import data with minimal logging and parallelism.

Ingesting data from tables on different warehouses and lakehouses

For both **CREATE TABLE AS SELECT** and **INSERT...SELECT**, the `SELECT` statement can also reference tables on warehouses that are different from the warehouse where your destination table is stored, by using **cross-warehouse queries**. This can be achieved by using the three-part naming convention `[warehouse_or_lakehouse_name.][schema_name.]table_name`. For example, suppose you have the following workspace assets:

- A lakehouse named `cases_lakehouse` with the latest case data.
- A warehouse named `reference_warehouse` with tables used for reference data.
- A warehouse named `research_warehouse` where the destination table is created.

A new table can be created that uses three-part naming to combine data from tables on these workspace assets:

SQL

```
CREATE TABLE [research_warehouse].[dbo].[cases_by_continent]
AS
SELECT
FROM [cases_lakehouse].[dbo].[bing_covid-19_data] cases
INNER JOIN [reference_warehouse].[dbo].[bing_covid-19_data] reference
ON cases.[iso3] = reference.[countrycode];
```

To learn more about cross-warehouse queries, see Write a cross-database SQL Query.

Next steps

- Ingesting data into the Warehouse
- Ingest data using the COPY statement
- Ingest data using Data pipelines
- Write a cross-database SQL Query

Default Power BI datasets in Microsoft Fabric

Article • 05/23/2023

Applies to: SQL Endpoint and Warehouse in Microsoft Fabric

In Microsoft Fabric, Power BI datasets are a semantic model with metrics; a logical description of an analytical domain, with business friendly terminology and representation, to enable deeper analysis. This semantic model is typically a star schema with facts that represent a domain, and dimensions that allow you to analyze, or slice and dice the domain to drill down, filter, and calculate different analyses. With the default dataset, the dataset is created automatically for you, and the aforementioned business logic gets inherited from the parent lakehouse or warehouse respectively, jump-starting the downstream analytics experience for business intelligence and analysis with an item in Microsoft Fabric that is managed, optimized, and kept in sync with no user intervention.

Visualizations and analyses in **Power BI reports** can now be built completely in the web - or in just a few steps in Power BI desktop - saving users time, resources, and by default, providing a seamless consumption experience for end-users. The default Power BI dataset follows the naming convention of the Lakehouse.

Power BI datasets represent a source of data ready for reporting, visualization, discovery, and consumption. Power BI datasets provide:

- The ability to expand warehousing constructs to include hierarchies, descriptions, relationships. This allows deeper semantic understanding of a domain.
- The ability to catalog, search, and find Power BI dataset information in the Data Hub.
- The ability to set bespoke permissions for workload isolation and security.
- The ability to create measures, standardized metrics for repeatable analysis.
- The ability to create Power BI reports for visual analysis.
- The ability discover and consume data in Excel.
- The ability for third party tools like Tableau to connect and analyze data.

For more on Power BI, see Power BI guidance.

> **Important**
>
> Microsoft Fabric is currently in PREVIEW. This information relates to a prerelease product that may be substantially modified before it's released. Microsoft makes no

warranties, expressed or implied, with respect to the information provided here.

Understand what's in the default Power BI dataset

When you create a Lakehouse, a default Power BI dataset is created with the SQL Endpoint. The default dataset is represented with the *(default)* suffix. For more information, see Default datasets.

The default dataset is queried via the SQL Endpoint and updated via changes to the Lakehouse. You can also query the default dataset via cross-database queries from a Warehouse.

By default, all tables and views in the warehouse are automatically added to the default Power BI dataset. Users can also manually select tables or views from the warehouse they want included in the model for more flexibility. Objects that are in the default Power BI dataset are created as a layout in the model view.

The background sync that includes objects (tables and views) waits for the downstream dataset to not be in use to update the dataset, honoring bounded staleness. Users can always go and manually pick tables they want or no want in the dataset.

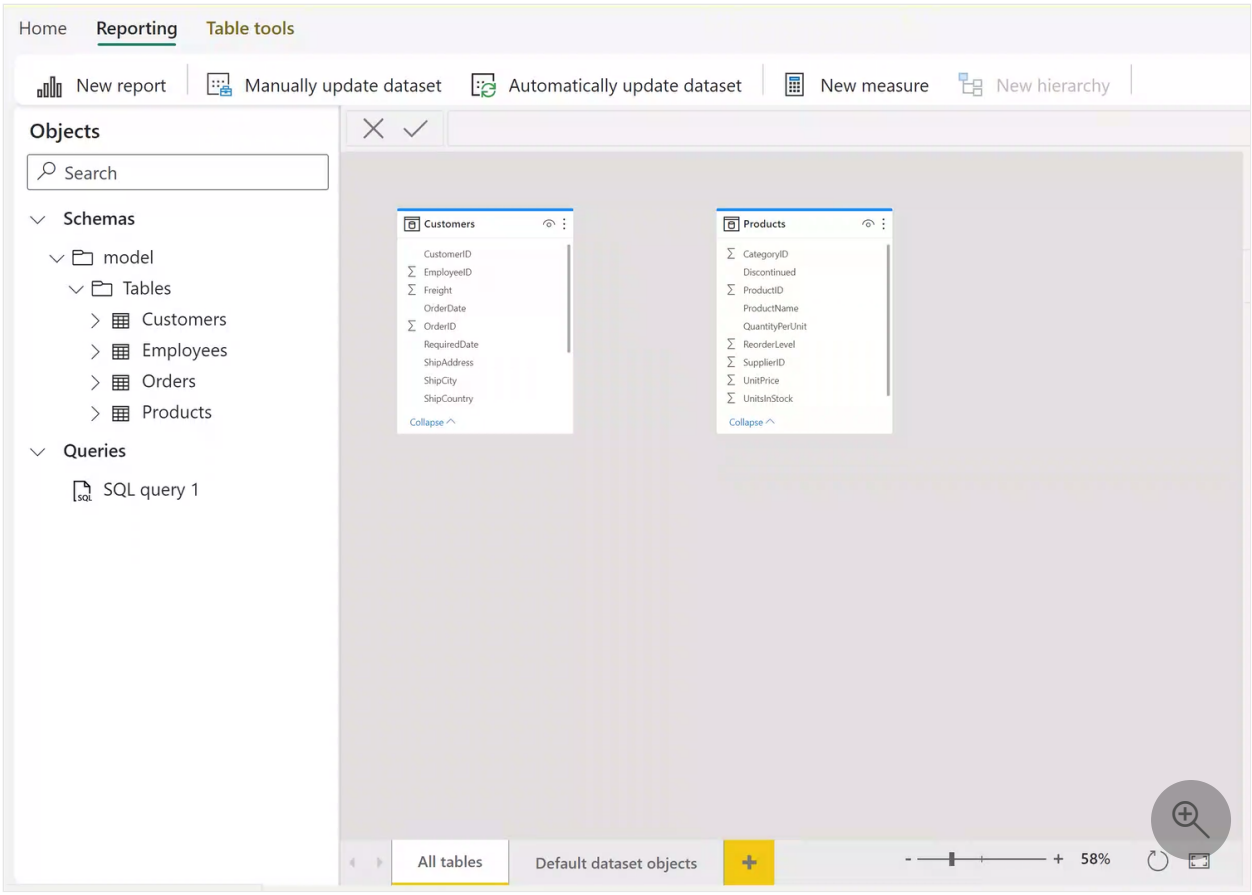
Manually update the default Power BI dataset

Once there are objects in the default Power BI dataset, there are two ways to validate or visually inspect the tables:

1. Select the **Manually update** dataset button in the ribbon.

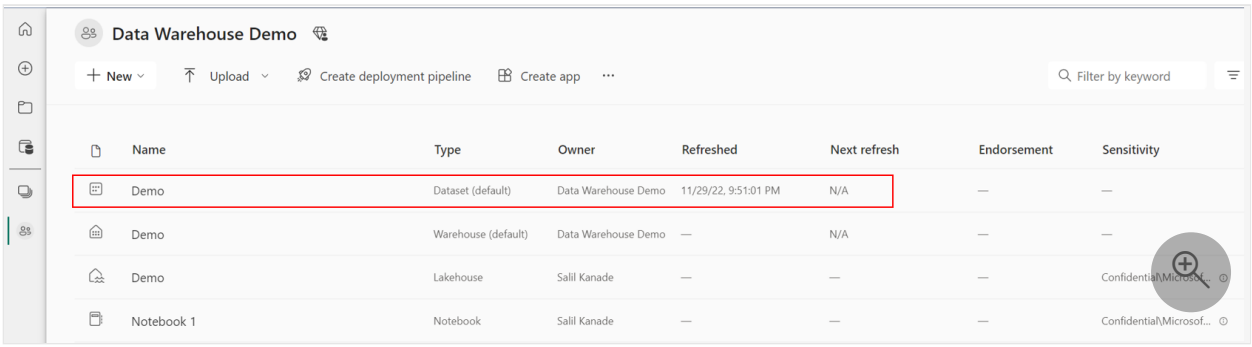
2. Review the default layout for the default dataset objects.

The default layout for BI enabled tables persists in the user session and is generated whenever a user navigates to the model view. Look for the **Default dataset objects** tab.

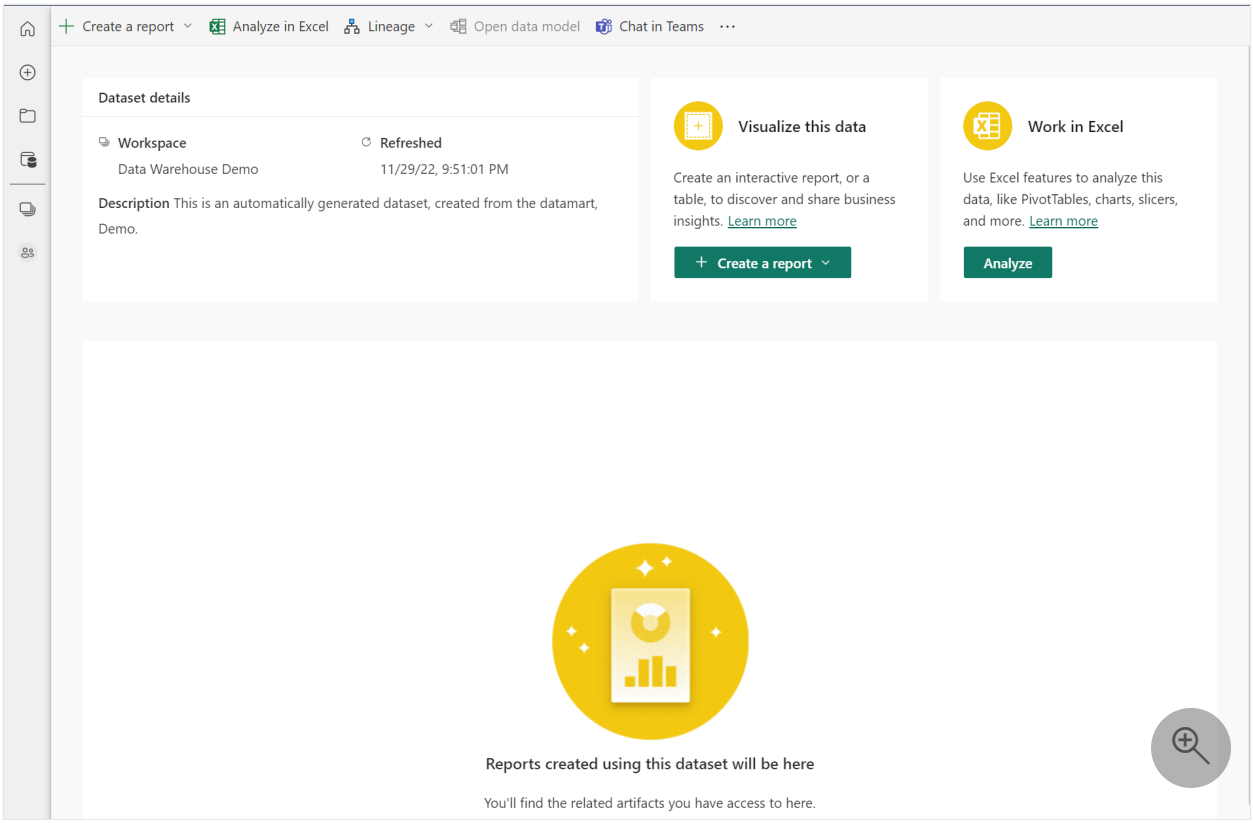


Access the default Power BI dataset

To access default Power BI datasets, go to your workspace, and find the dataset that matches the name of the desired Lakehouse. The default Power BI dataset follows the naming convention of the Lakehouse.



To load the dataset, select the name of the dataset.

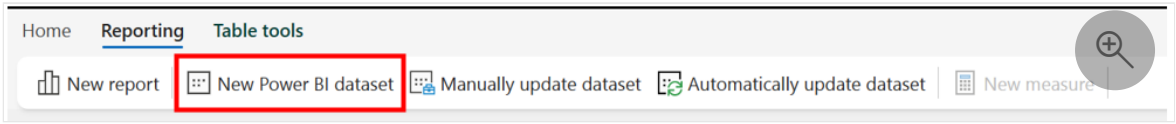


Create a new Power BI dataset

There are some situations where your organization may need to create additional Power BI datasets based off SQL endpoint or Warehouse data. To create a Power BI dataset from a warehouse, follow these steps:

1. Open the warehouse, and then switch to the **Reporting** ribbon.

2. In the **Reporting** ribbon, select **New Power BI dataset**, and then in the **New dataset** dialog, select tables to be included, and then select **Confirm**.



3. Power BI automatically saves the dataset in the workspace based on the name of your warehouse, and then opens the dataset in Power BI.

4. Select **Open data model** to open the Power BI Web modeling experience where you can add table relationships and DAX measures.

To learn more on how to edit data models in the Power BI service, see Edit Data Models.

Limitations

Default Power BI datasets follow the current limitations for datasets in Power BI. Learn more:

- Azure Analysis Services resource and object limits | Microsoft Learn
- Data types in Power BI Desktop - Power BI | Microsoft Learn

If the parquet, Apache Spark, or SQL data types can't be mapped to one of the above types, they are dropped as part of the sync process. This is in line with current Power BI behavior. For these columns, we recommend that you add explicit type conversions in their ETL processes to convert it to a type that is supported. If there are data types that are needed upstream, users can optionally specify a view in SQL with the explicit type conversion desired. This will be picked up by the sync or can be added manually as previously indicated.

Next steps

- Define relationships in data models
- Data modeling in the default Power BI dataset

Data modeling in the default Power BI dataset in Microsoft Fabric

Article • 05/23/2023

Applies to: SQL Endpoint and Warehouse in Microsoft Fabric

The default Power BI dataset inherits all relationships between entities defined in the model view and infers them as Power BI dataset relationships, when objects are enabled for BI (Power BI Reports). Inheriting the warehouse's business logic allows a warehouse developer or BI analyst to decrease the time to value towards building a useful semantic model and metrics layer for analytical business intelligence (BI) reports in Power BI, Excel, or external tools like Tableau that read the XMLA format.

While all constraints are translated to relationships, currently in Power BI, only one relationship can be active at a time, whereas multiple primary and foreign key constraints can be defined for warehouse entities and are shown visually in the diagram lines. The active Power BI relationship is represented with a solid line and the rest is represented with a dotted line. We recommend choosing the primary relationship as active for BI reporting purposes.

> **Important**
>
> Microsoft Fabric is currently in PREVIEW. This information relates to a prerelease product that may be substantially modified before it's released. Microsoft makes no warranties, expressed or implied, with respect to the information provided here.

Data modeling properties

The following table provides a description of the properties available when using the model view diagram and creating relationships:

Column name	Description
FromObjectName	Table/View name "From" which relationship is defined.
ToObjectName	Table/View name "To" which a relationship is defined.
TypeOfRelationship	Relationship cardinality, the possible values are: None, OneToOne, OneToMany, ManyToOne, and ManyToMany.

Column name	Description
SecurityFilteringBehavior	Indicates how relationships influence filtering of data when evaluating row-level security expressions and is a Power BI specific semantic. The possible values are: OneDirection, BothDirections, and None.
IsActive	A Power BI specific semantic, and a boolean value that indicates whether the relationship is marked as Active or Inactive. This defines the default relationship behavior within the semantic model
RelyOnReferentialIntegrity	A boolean value that indicates whether the relationship can rely on referential integrity or not.
CrossFilteringBehavior	Indicates how relationships influence filtering of data and is Power BI specific. The possible values are: 1 - OneDirection, 2 - BothDirections, and 3 - Automatic.

Add or remove objects to the default Power BI dataset

In Power BI, a dataset is always required before any reports can be built, so the default Power BI dataset enables quick reporting capabilities on top of the warehouse. Within the warehouse, a user can add warehouse objects - tables or views to their default Power BI dataset. They can also add other semantic modeling properties, such as hierarchies and descriptions. These properties are then used to create the Power BI dataset's tables. Users can also remove objects from the default Power BI dataset.

To add objects such as tables or views to the default Power BI dataset, you have options:

1. Automatically add objects to the dataset, which happens by default with no user intervention needed.

2. Manually add objects to the dataset.

The auto detect experience determines any tables or views and opportunistically adds them.

The manually detect option in the ribbon allows fine grained control of which object(s), such as tables and/or views, should be added to the default Power BI dataset:

- Select all
- Filter for tables or views
- Select specific objects

To remove objects, a user can use the manually select button in the ribbon and:

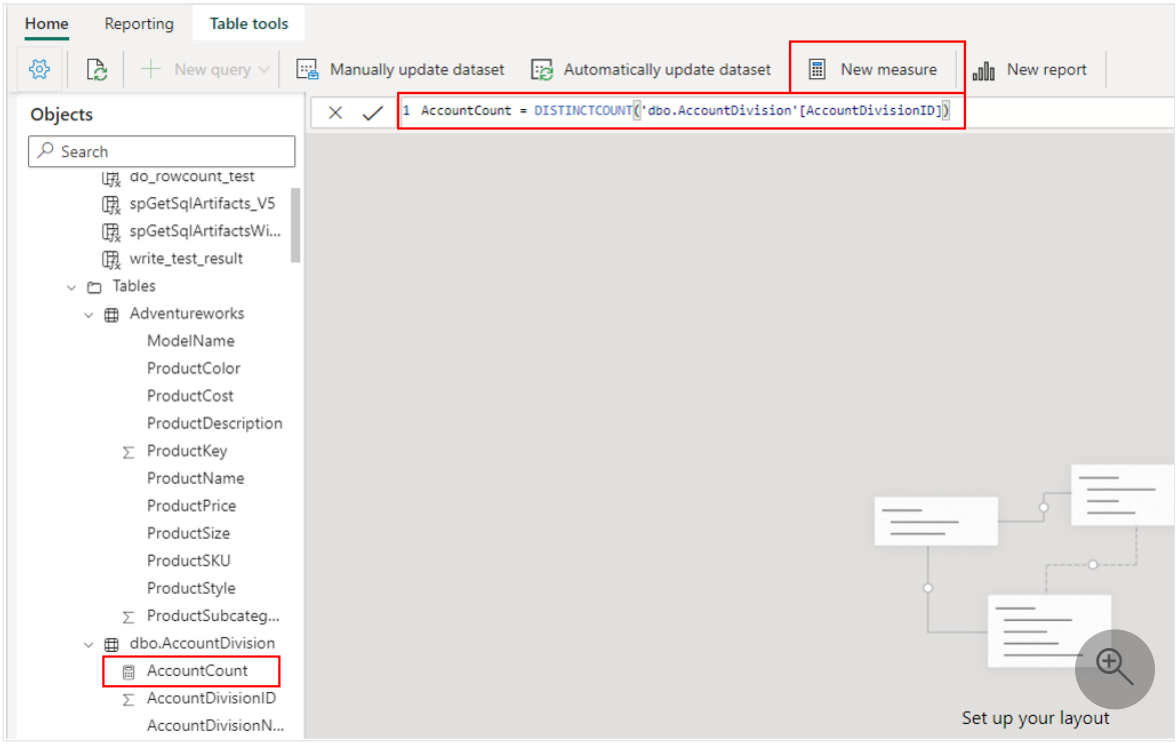
- Un-select all
- Filter for tables or views
- Un-select specific objects

> **Tip**
>
> We recommend reviewing the objects enabled for BI and ensuring they have the correct logical relationships to ensure a smooth downstream reporting experience.

Create a measure

A measure is a collection of standardized metrics. Similar to Power BI Desktop, the DAX editing experience in warehouse presents a rich editor complete with autocomplete for formulas (IntelliSense). The DAX editor enables you to easily develop measures right in warehouse, making it a more effective single source for business logic, semantics, and business critical calculations.

1. To create a measure, select the **New Measure** button in the ribbon, as shown in the following image.



2. Enter the measure into the formula bar and specify the table and the column to which it applies. The formula bar lets you enter your measure. For detailed

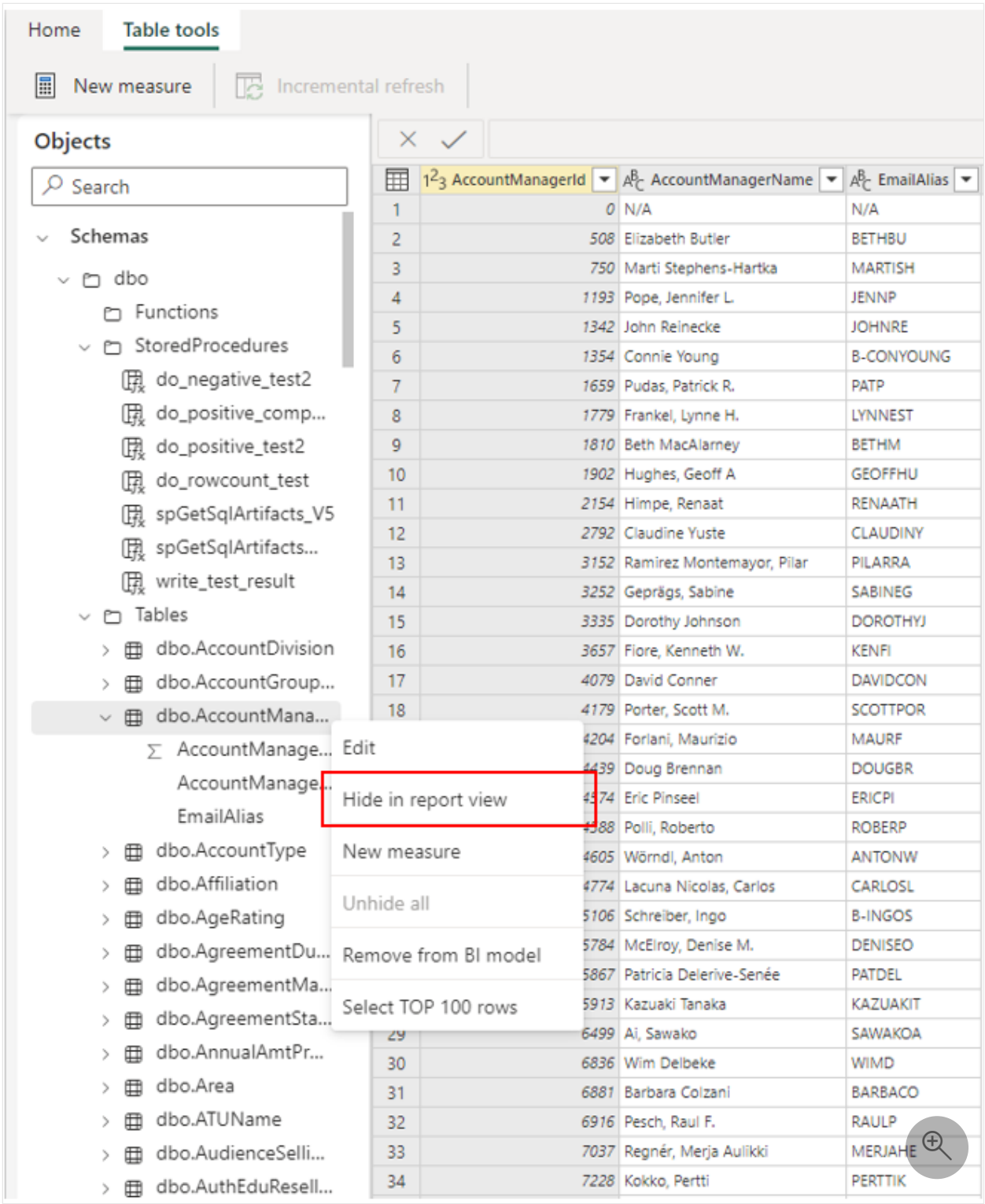
information on measures, see Tutorial: Create your own measures in Power BI Desktop.

3. You can expand the table to find the measure in the table.

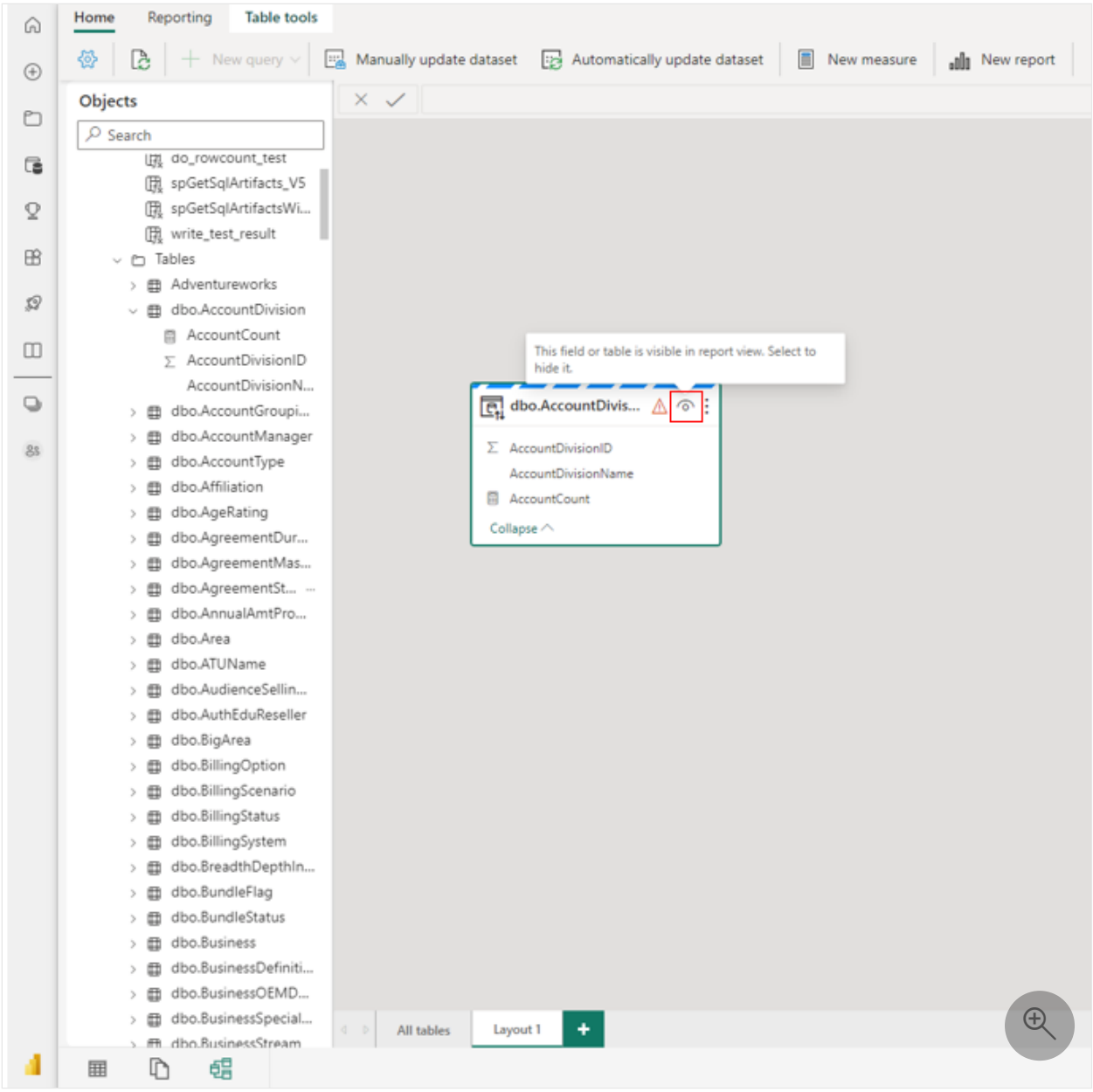
Hide elements from downstream reporting

You can hide elements of your warehouse from downstream reporting by right-clicking on the column or table you want to hide from the object explorer.

Select **Hide** in **Report view** from the menu that appears to hide the item from downstream reporting.



You can also hide the entire table and individual columns by using the **Model view** canvas options, as shown in the following image.



Next steps

- Define relationships in data models
- Create reports in the Power BI service

Define relationships in data models for data warehousing in Microsoft Fabric

Article • 05/23/2023

Applies to: SQL Endpoint and Warehouse in Microsoft Fabric

A well-defined data model is instrumental in driving your analytics and reporting experiences. In a Warehouse in Microsoft Fabric, you can easily build and change your data model with a few simple steps in our visual editor. You need to have at least a small sample of data loaded before you can explore these concepts further; tables may be empty, but the schemas (their structures) need to be defined.

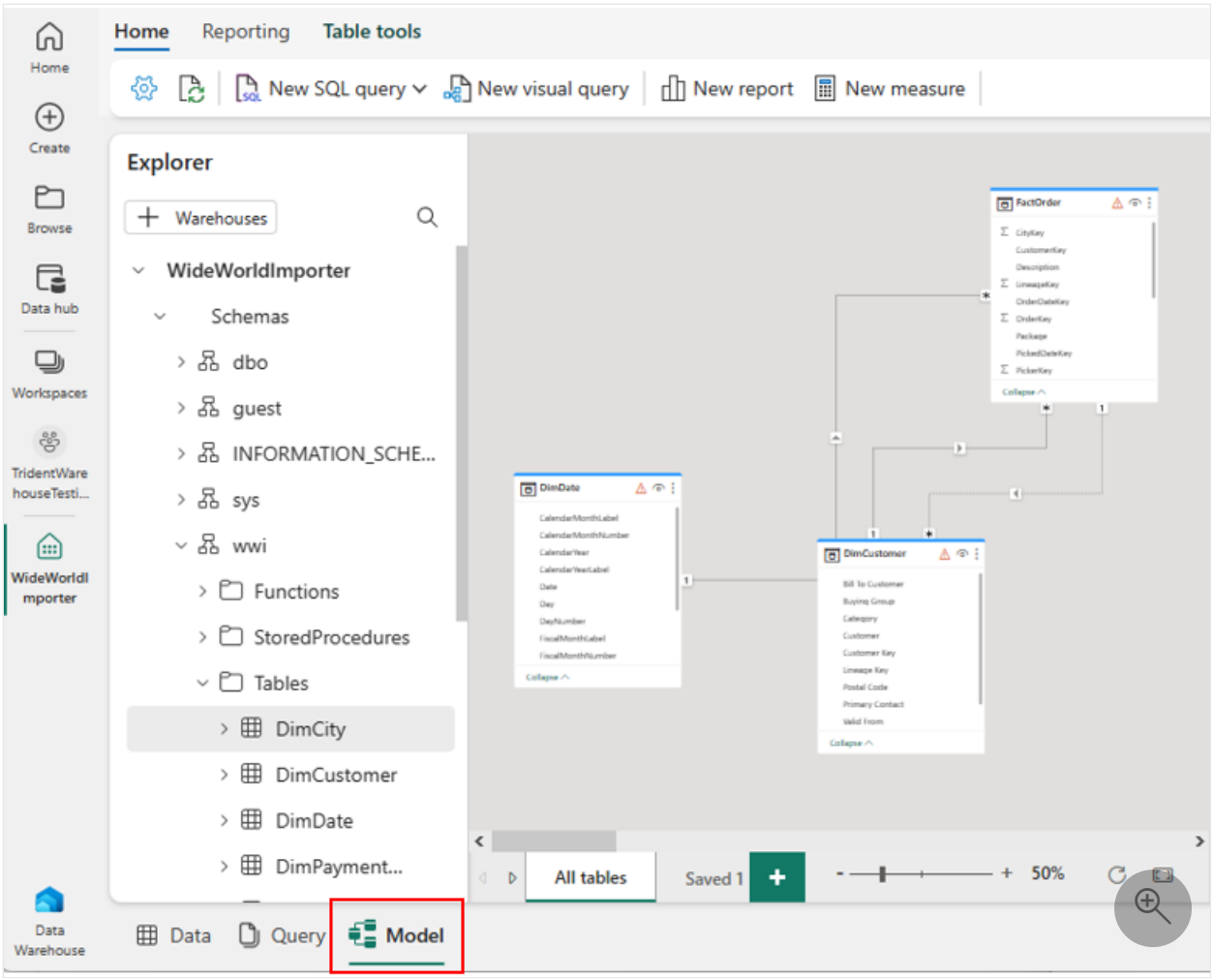
> **Important**
>
> Microsoft Fabric is currently in PREVIEW. This information relates to a prerelease product that may be substantially modified before it's released. Microsoft makes no warranties, expressed or implied, with respect to the information provided here.

Warehouse modeling

Modeling the warehouse is possible by setting primary and foreign key constraints and setting identity columns on the model view within the data warehouse UX. After you navigate the model view, you can do this in a visual entity relationship diagram that allows a user to drag and drop tables to infer how the objects relate to one another. Lines visually connecting the entities infer the type of physical relationships that exist.

How to model data and define relationships

To model your data, navigate to **Model view** by selecting the **Model view** icon at the bottom left of the window, as shown in the following image.



In the model view, users can model their warehouse and the canonical autogenerated default Power BI dataset. We recommend modeling your data warehouse using traditional Kimball methodologies, using a star schema, wherever possible. There are two types of modeling possible:

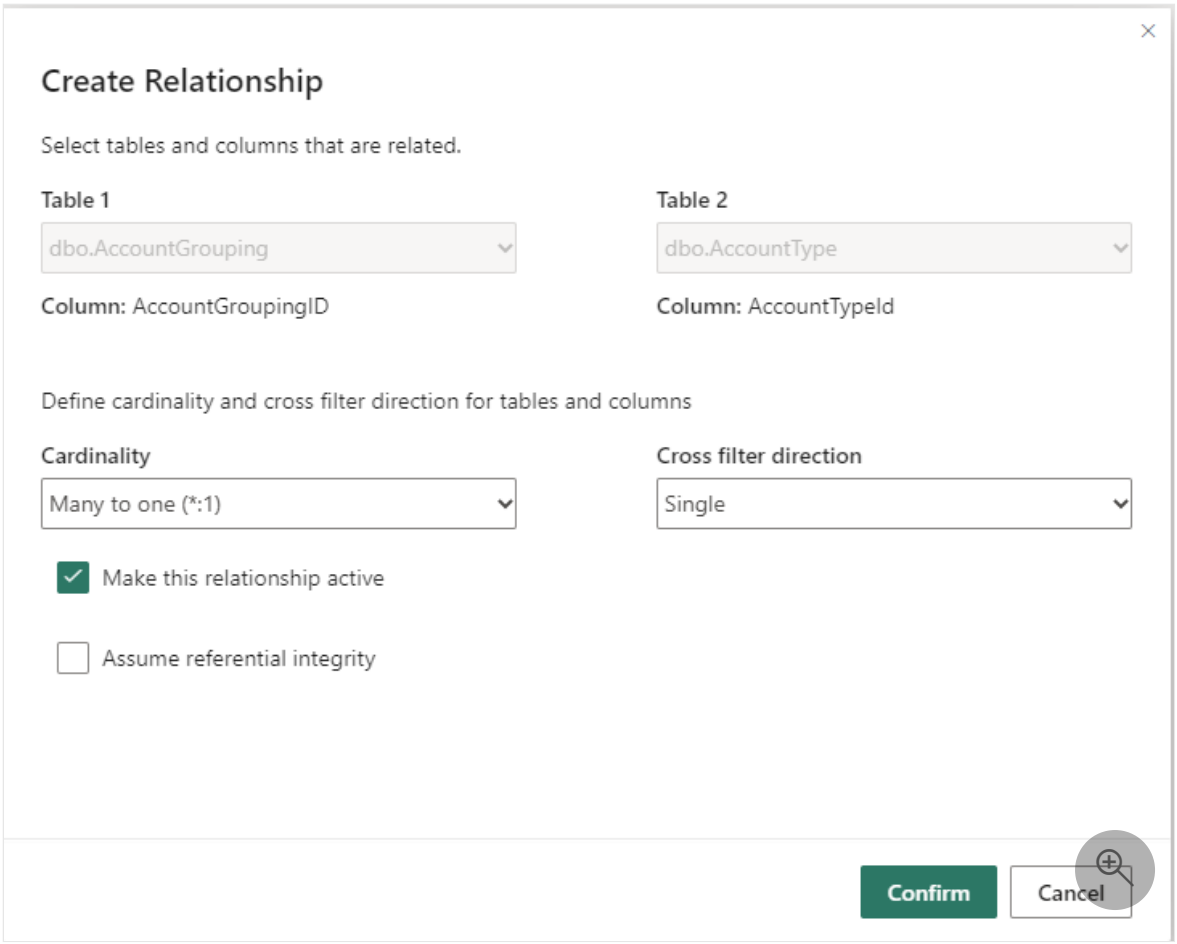
1. Warehouse modeling - the physical relationships expressed as primary and foreign keys and constraints

2. Default Power BI dataset modeling - the logical relationships expressed between entities

Modeling automatically keeps these definitions in sync, enabling powerful warehouse and semantic layer development simultaneously.

Define physical and logical relationships

1. To create a logical relationship between entities in a warehouse and the resulting primary and foreign key constraints, select the **Model view** and select your warehouse, then drag the column from one table to the column on the other table to initiate the relationship. In the window that appears, configure the relationship properties.

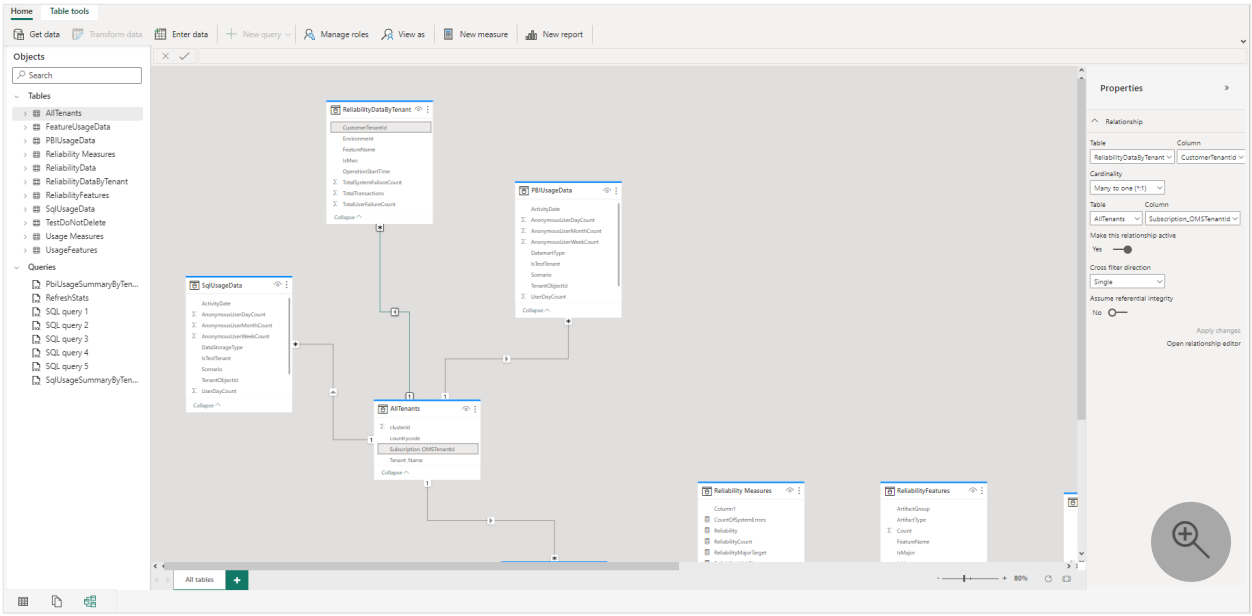


2. Select the **Confirm** button when your relationship is complete to save the relationship information. The relationship set will effectively:
   a. Set the physical relationships - primary and foreign key constraints in the database
   b. Set the logical relationships - primary and foreign key constraints in the default Power BI dataset

Edit relationships using different methods

Using drag and drop and the associated **Edit relationships dialog** is a more guided experience for editing relationships in Power BI.

In contrast, editing relationships in the **Properties** pane is a streamlined approach to editing relationships:



You only see the table names and columns from which you can choose, you aren't presented with a data preview, and the relationship choices you make are only validated when you select **Apply changes**. Using the **Properties** pane and its streamlined approach reduces the number of queries generated when editing a relationship, which can be important for big data scenarios, especially when using DirectQuery connections. Relationships created using the **Properties** pane can also use multi-select relationships in the **Model** view diagram layouts. Pressing the **Ctrl** key and select more than one line to select multiple relationships. Common properties can be edited in the **Properties** pane and **Apply changes** processes the changes in one transaction.

Single or multi-selected relationships can also be deleted by pressing **Delete** on your keyboard. You can't undo the delete action, so a dialog prompts you to confirm deleting the relationships.

Using model view layouts

During the session, users may create multiple tabs in the model view to depict say, data warehouse schemas or further assist with database design. Currently the model view layouts are only persisted in session. However the database changes are persisted. Users can use the auto-layout whenever a new tab is created to visually inspect the database design and understand the modeling.

Next steps

- Data modeling in the default Power BI dataset

Create reports in the Power BI service in Microsoft Fabric and Power BI Desktop

Article • 05/23/2023

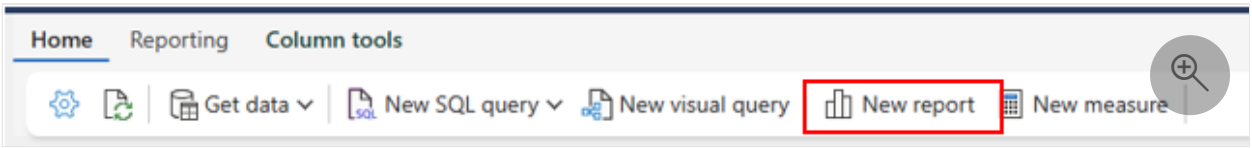
Applies to: SQL Endpoint and Warehouse in Microsoft Fabric

This article describes three different scenarios you can follow to create reports in the Power BI service.

> **Important**
>
> Microsoft Fabric is currently in PREVIEW. This information relates to a prerelease product that may be substantially modified before it's released. Microsoft makes no warranties, expressed or implied, with respect to the information provided here.

Create a report from the warehouse editor

From within the warehouse experience, using the ribbon and the main home tab, navigate to the **New report** button. This option provides a native, quick way to create report built on top of the default Power BI dataset.

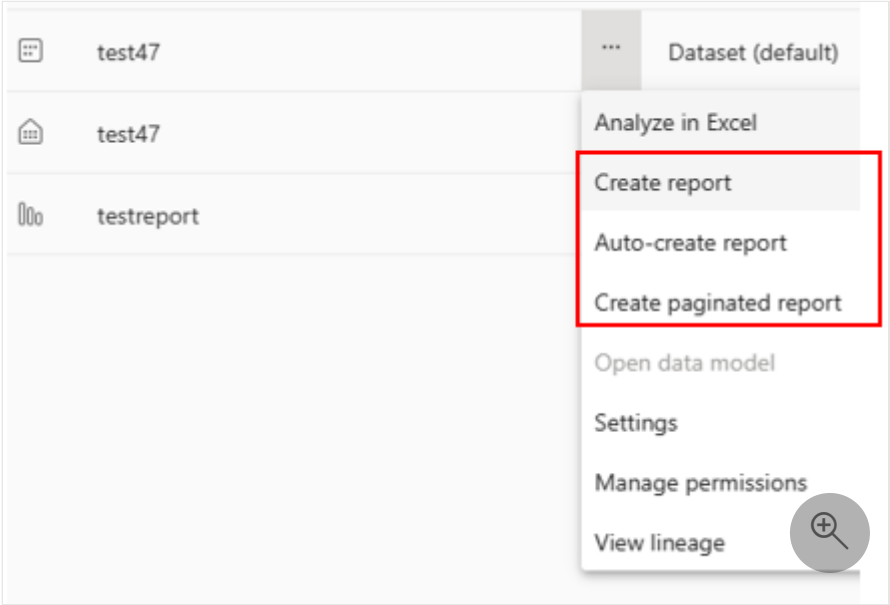


If no tables have been added to the default Power BI dataset, the dialog first automatically adds tables, prompting the user to confirm or manually select the tables included in the canonical default dataset first, ensuring there's always data first.

With a default dataset that has tables, the **New report** opens a browser tab to the report editing canvas to a new report that is built on the dataset. When you save your new report you're prompted to choose a workspace, provided you have write permissions for that workspace. If you don't have write permissions, or if you're a free user and the dataset resides in a Premium capacity workspace, the new report is saved in your **My workspace**.

Use default Power BI dataset within workspace

Using the default dataset and action menu in the workspace: In the Microsoft Fabric workspace, navigate to the default Power BI dataset and select the **More** menu (...) to create a report in the Power BI service.



Select **Create report** to open the report editing canvas to a new report on the dataset. When you save your new report, it's saved in the workspace that contains the dataset as long as you have write permissions on that workspace. If you don't have write permissions, or if you're a free user and the dataset resides in a Premium capacity workspace, the new report is saved in your **My workspace**.

Use Data hub

Using the default Power BI dataset and dataset details page. In the workspace list, select the default dataset's name to get to the **Dataset** details page, where you can find details about the dataset and see related reports. You can also create a report directly from this page. To learn more about creating a report in this fashion, see Dataset details.

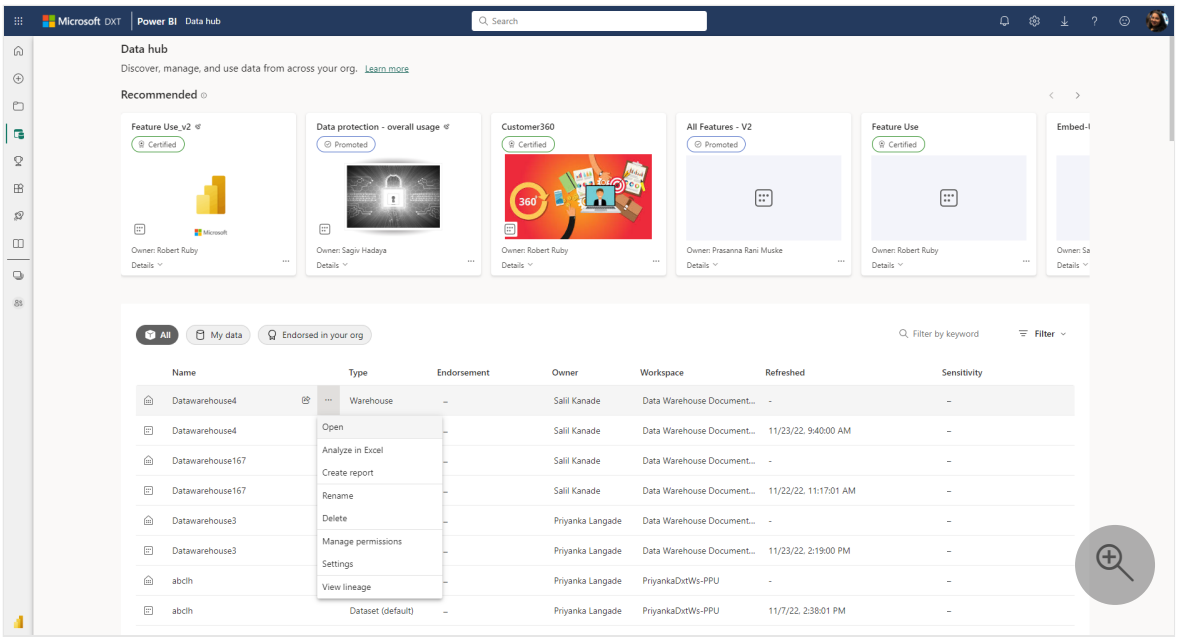
In the **Data hub**, you see warehouse and their associated default datasets. Select the warehouse to navigate to the warehouse details page. You can see the warehouse metadata, supported actions, lineage and impact analysis, along with related reports created from that warehouse. Default datasets derived from a warehouse behave the same as any dataset.

To find the warehouse, you begin with the **Data hub**. The following image shows the **Data hub** in the Power BI service:

1. Select a warehouse to view its warehouse details page.

2. Select the **More** menu (...) to display the options menu.

3. Select **Open** to open the warehouse.



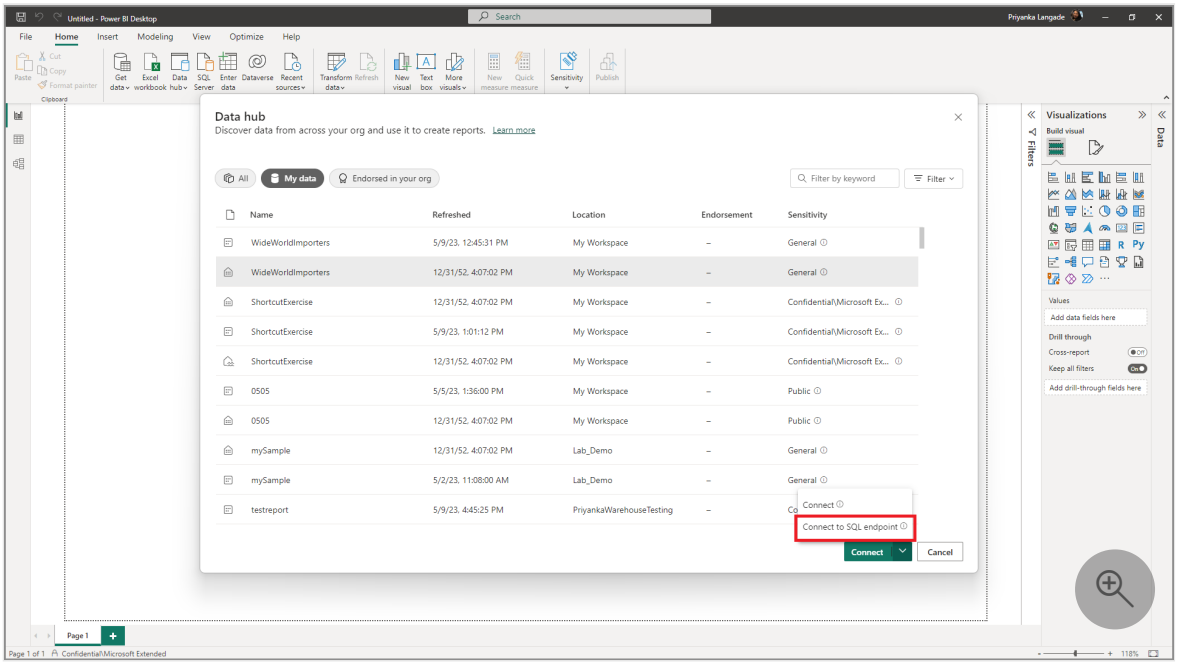
Create reports in the Power BI Desktop

The Data hub integration in Power BI Desktop lets you connect to the Warehouse or SQL endpoint of Lakehouse in easy steps.

1. Use **Data hub** menu in the ribbon to get list of all items.

2. Select the warehouse that you would like to connect

3. From the drop down on **Connect** button, select **Connect to SQL endpoint**.



Next steps

- Connectivity
- Create reports
- Tutorial: Get started creating in the Power BI service

Security for data warehousing in Microsoft Fabric

Article • 05/23/2023

Applies to: SQL Endpoint and Warehouse in Microsoft Fabric

This article covers security topics for securing the SQL Endpoint of the lakehouse and the Warehouse in Microsoft Fabric.

> **Important**
>
> Microsoft Fabric is currently in PREVIEW. This information relates to a prerelease product that may be substantially modified before it's released. Microsoft makes no warranties, expressed or implied, with respect to the information provided here.

For information on Microsoft Fabric security, see Security in Microsoft Fabric.

For information on connecting to the SQL Endpoint and Warehouse, see Connectivity.

Warehouse access model

Microsoft Fabric permissions and granular SQL permissions work together to govern Warehouse access and the user permissions once connected.

- Warehouse connectivity is dependent on being granted the Microsoft Fabric Read permission, at a minimum, for the Warehouse.
- Microsoft Fabric item permissions enable the ability to provide a user with SQL permissions, without needing to grant those permissions within SQL.
- Microsoft Fabric workspace roles provide Microsoft Fabric permissions for all warehouses within a workspace.
- Granular user permissions can be further managed via T-SQL.

Workspace roles

Workspace roles are used for development team collaboration within a workspace. Role assignment determines the actions available to the user and applies to all items within the workspace.

- For an overview of Microsoft Fabric workspace roles, see Roles in workspaces.
- For instructions on assigning workspace roles, see Give Workspace Access.

See Workspace roles in Fabric data warehousing for details on the specific Warehouse capabilities provided through Workspace roles.

Object-level security

Workspace roles and item permissions provide an easy way to assign coarse permissions to a user for the entire warehouse. However, in some cases, more granular permissions are needed for a user. To achieve this, standard T-SQL constructs can be used to provide specific permissions to users.

See SQL granular permissions for details on the managing granular permissions in SQL.

Guidance

When evaluating the permissions to assign to a user, consider the following guidance:

- Only team members who are currently collaborating on the solution should be assigned to Workspace roles (Admin, Member, Contributor), as this provides them access to all Items within the workspace.
- If they primarily require read only access, assign them to the Viewer role and grant read access on specific objects through T-SQL. For more information, see Manage SQL granular permissions.
- If they are higher privileged users, assign them to Admin, Member or Contributor roles. The appropriate role is dependent on the other actions that they will need to perform.
- Other users, who only need access to an individual warehouse or require access to only specific SQL objects, should be given Fabric Item permissions and granted access through SQL to the specific objects.
- You can manage permissions on Azure Activity Directory groups, as well, rather than adding each specific member.

Next steps

- Connectivity

Workspace roles in Fabric data warehousing

Article • 05/23/2023

Applies to: SQL Endpoint and Warehouse in Microsoft Fabric

This article details the permissions that workspace roles provide in SQL Endpoint and Warehouse. For instructions on assigning workspace roles, see Give Workspace Access.

> **Important**
>
> Microsoft Fabric is currently in PREVIEW. This information relates to a prerelease product that may be substantially modified before it's released. Microsoft makes no warranties, expressed or implied, with respect to the information provided here.

Workspace roles

Assigning users to the various workspace roles provides the following capabilities:

Workspace role	Description
Admin	Grants the user CONTROL access for each Warehouse and SQL Endpoint within the workspace, providing them with full read/write permissions and the ability to manage granular user SQL permissions. Allows the user to see workspace-scoped session, monitor connections and requests in DMVs via TSQL, and KILL sessions.
Member	Grants the user CONTROL access for each Warehouse and SQL Endpoint within the workspace, providing them with full read/write permissions and the ability to manage granular user SQL permissions.
Contributor	Grants the user CONTROL access for each Warehouse and SQL Endpoint within the workspace, providing them with full read/write permissions and the ability to manage granular user SQL permissions.
Viewer	Grants the user CONNECT permissions for each Warehouse and SQL Endpoint within the workspace. Viewers can be granted granular SQL permissions to read data from tables/views using T-SQL. For more information, see Manage SQL granular permissions.

Next steps

- Security for data warehousing in Microsoft Fabric
- SQL granular permissions
- Manage item permissions in Microsoft Fabric
- Connectivity
- Monitoring connections, sessions, and requests using DMVs

SQL granular permissions in Microsoft Fabric

Article • 05/23/2023

Applies to: ✅ SQL Endpoint and Warehouse in Microsoft Fabric

When the out-of-the box permissions provided by assignment to workspace roles or granted through item permissions are insufficient, standard SQL constructs are available for more granular control.

For SQL Endpoint and Warehouse:

- Object-level-security can be managed using GRANT, REVOKE, and DENY syntax.
  - For more information, see T-SQL syntax for GRANT, REVOKE, and DENY.
- Users can be assigned to SQL roles, both custom and built-in database roles.

> **ⓘ Important**
>
> Microsoft Fabric is currently in PREVIEW. This information relates to a prerelease product that may be substantially modified before it's released. Microsoft makes no warranties, expressed or implied, with respect to the information provided here.

User granular permissions

- In order for a user to connect to the database, the user must be assigned to a Workspace role or assigned the item **Read** permission. Without **Read** permission at a minimum, the connection fails.
- If you'd like to set up a user's granular permissions prior to allowing them to connect to the warehouse, permissions can first be set up within SQL. Then, they can be given access by assigning them to a Workspace role or granting item permissions.

Limitations

- CREATE USER cannot be explicitly executed currently. When GRANT or DENY is executed, the user will be created automatically.
- Row-level security is currently not supported.
- Dynamic data masking is currently not supported.

View my permissions

When a user connects to the SQL connection string, they can view the permissions available to them using the sys.fn_my_permissions function.

User's database scoped permissions:

```
SQL

SELECT *
FROM sys.fn_my_permissions(NULL, "Database")
```

User's schema scoped permissions:

```
SQL

SELECT *
FROM sys.fn_my_permissions("<schema-name>", "Schema")
```

User's object-scoped permissions:

```
SQL

SELECT *
FROM sys.fn_my_permissions("<schema-name>.<object-name>", "Object")
```

View permissions granted explicitly to users

When connected via the SQL connection string, a user with elevated permissions can query the permissions that have been granted by using system views. This doesn't show the users or user permissions that are given to users by being assigned to workspace roles or assigned item permissions.

```
SQL

SELECT DISTINCT pr.principal_id, pr.name, pr.type_desc,
 pr.authentication_type_desc, pe.state_desc, pe.permission_name
FROM sys.database_principals AS pr
JOIN sys.database_permissions AS pe
 ON pe.grantee_principal_id = pr.principal_id;
```

Restrict row access by using views

Row level security is currently not supported. As a workaround, views and system functions can be used to limit a user's access to the data. This can be achieved in the following way:

1. Provide the user with the Fabric Read permission only - This will grant them CONNECT permissions only for the Warehouse.

2. Optionally, create a custom role and add the user to the role, if you'd like to restrict access based on roles.

   SQL

   ```
   CREATE ROLE PrivilegedRole

   ALTER ROLE PrivilegedRole ADD MEMBER [userOne@contoso.com]
   ```

3. Create a view that queries the table for which you'd like to restrict row access

4. Add a WHERE clause within the VIEW definition, using the SUSER_SNAME() or IS_ROLEMEMBER() system functions, to filter based on user name or role membership. Below is an example of providing access to certain rows to users based on region data within the row. The first condition provides access to rows, of a specific region, to one specific user, while the second condition provides access to rows, of a specific region, to any member of the PrivilegedRole custom role.

   SQL

   ```
   CREATE VIEW dbo.RestrictedAccessTable as
   select *
   from dbo.SampleTable
   WHERE
   ( SUSER_SNAME() = 'userTwo@contoso.com' AND test_region =
   '<region_one_name>')
   OR
   ( IS_ROLEMEMBER('PrivilegedRole', SUSER_SNAME()) = 1 AND test_region =
   '<region_two_name')
   ```

5. Grant access to the view:

   SQL

   ```
   GRANT SELECT ON dbo.RestrictedAccessTable TO [userOne@contoso.com]
   ```

Next steps

- Security for data warehousing in Microsoft Fabric
- Manage item permissions in Microsoft Fabric
- GRANT, REVOKE, and DENY

Query using the visual query editor

Article • 05/23/2023

Applies to: SQL Endpoint and Warehouse in Microsoft Fabric

You can query the data in your warehouse with multiple tools, including the visual query editor and the SQL query editor. This article describes how to use the visual query editor to quickly and efficiently write queries, and suggestions on how best to see the information you need.

> **Important**
>
> Microsoft Fabric is currently in PREVIEW. This information relates to a prerelease product that may be substantially modified before it's released. Microsoft makes no warranties, expressed or implied, with respect to the information provided here.

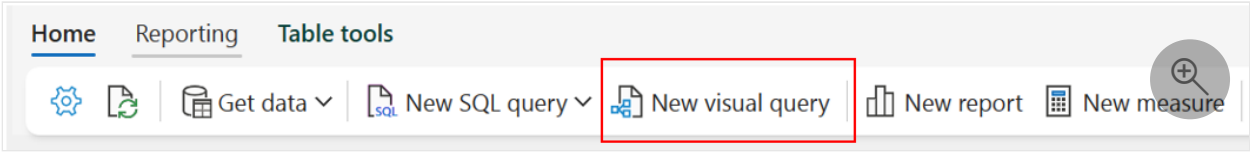
Visual query editor in the Fabric portal

The visual query editor provides an easy visual interface to write queries against the data in your warehouse.

Once you've loaded data into your warehouse, you can use the visual query editor to create queries to analyze your data. You can use the visual query editor for a no-code experience to create your queries.

There are two ways to get to the visual query editor:

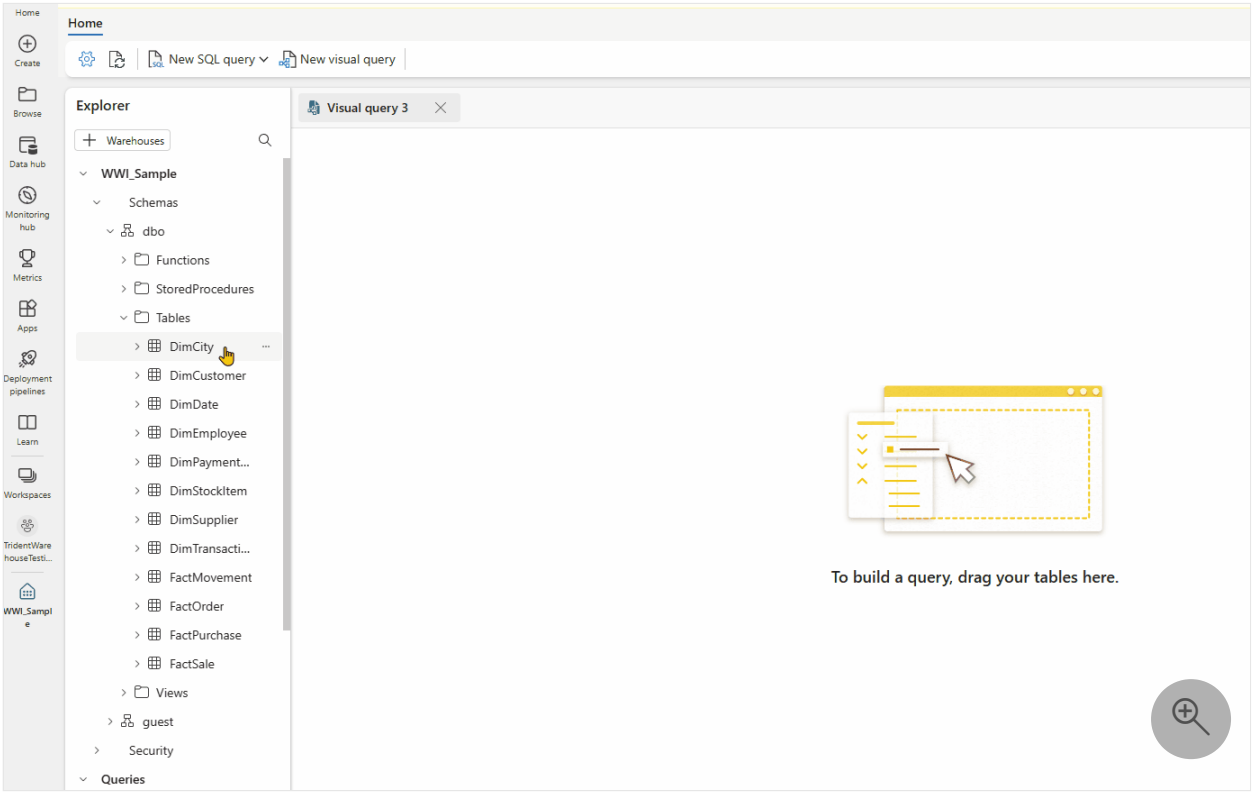
In the ribbon, create a new query using the **New visual query** button, as shown in the following image.



To create a query, drag and drop tables from the **Object explorer** on the left onto the canvas. Once you drag one or more tables onto the canvas, you can use the visual experience to design your queries. The warehouse editor uses the Power Query diagram view experience to enable you to easily query and analyze your data. Learn more about Power Query diagram view.

As you work on your visual query, the queries are automatically saved every few seconds. A "saving indicator" appears in your query tab to indicate that your query is being saved.

The following animated gif shows the merging of two tables using a no-code visual query editor. First, the `DimCity` then `FactSale` are dragged from the **Explorer** into the visual query editor. Then, the **Merge** Power Query operator is used to join them on a common key.

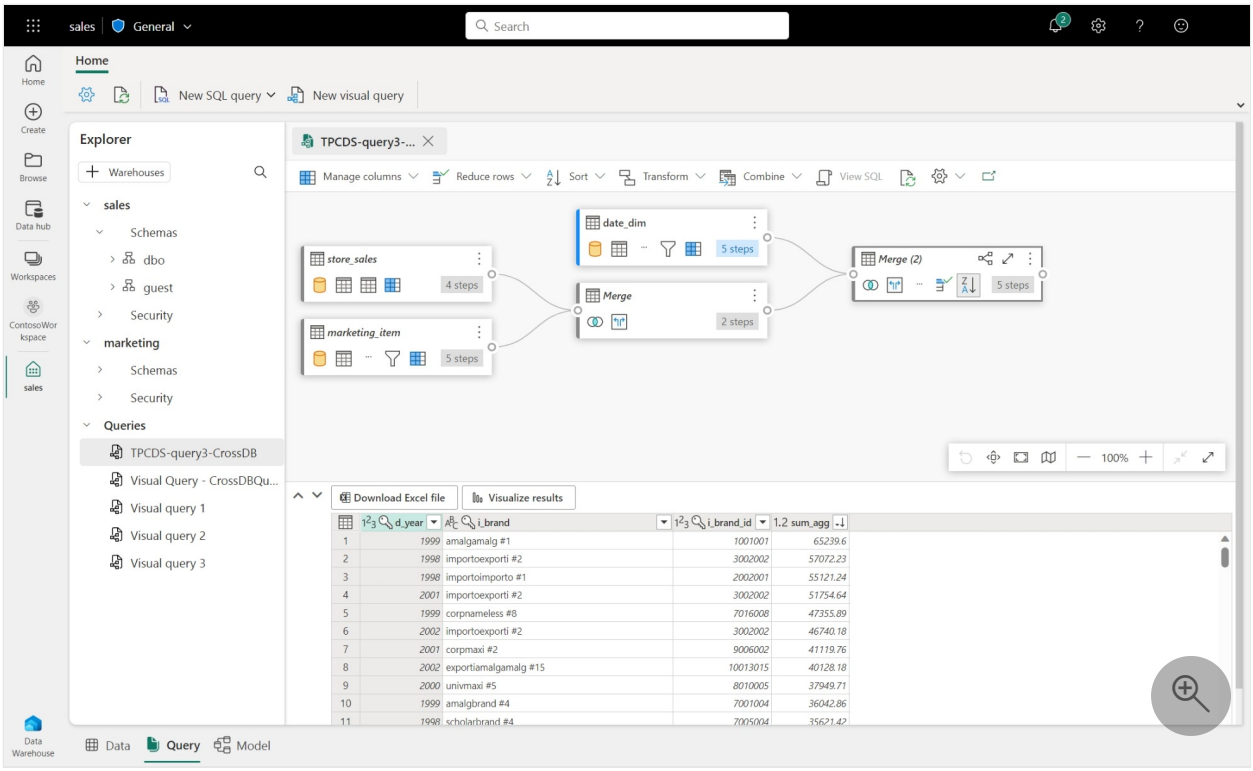


When you see results, you can use **Download Excel file** to view results in Excel or **Visualize results** to create report on results.

Create a cross-warehouse query in visual query editor

For more information on cross-warehouse querying, see Cross-warehouse querying.

- To create a cross-warehouse query, drag and drop tables from added warehouses and add merge activity. For example, in the following image example, `store_sales` is added from `sales` warehouse and it's merged with `item` table from `marketing` warehouse.



Limitations with visual query editor

- In the visual query editor, you can only run DQL (Data Query Language) or read-only SELECT statements. DDL or DML are not supported.
- Only a subset of Power Query operations that support Query folding are currently supported.

Next steps

- How-to: Query the Warehouse
- Query using the SQL Query editor

Query using the SQL query editor

Article • 05/23/2023

Applies to: ✅ SQL Endpoint and Warehouse in Microsoft Fabric

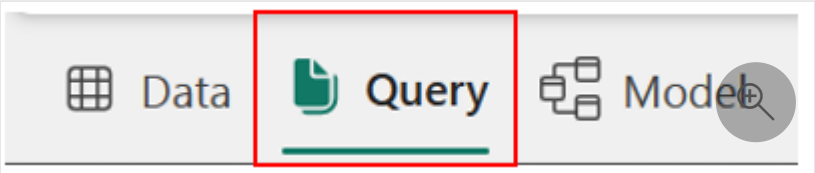
You can query the data in your warehouse with multiple tools, including the Visual query editor and the SQL query editor in the Microsoft Fabric portal. This article describes how to use the SQL query editor to quickly and efficiently write queries, and suggestions on how best to see the information you need.

> **ⓘ Important**
>
> Microsoft Fabric is currently in PREVIEW. This information relates to a prerelease product that may be substantially modified before it's released. Microsoft makes no warranties, expressed or implied, with respect to the information provided here.

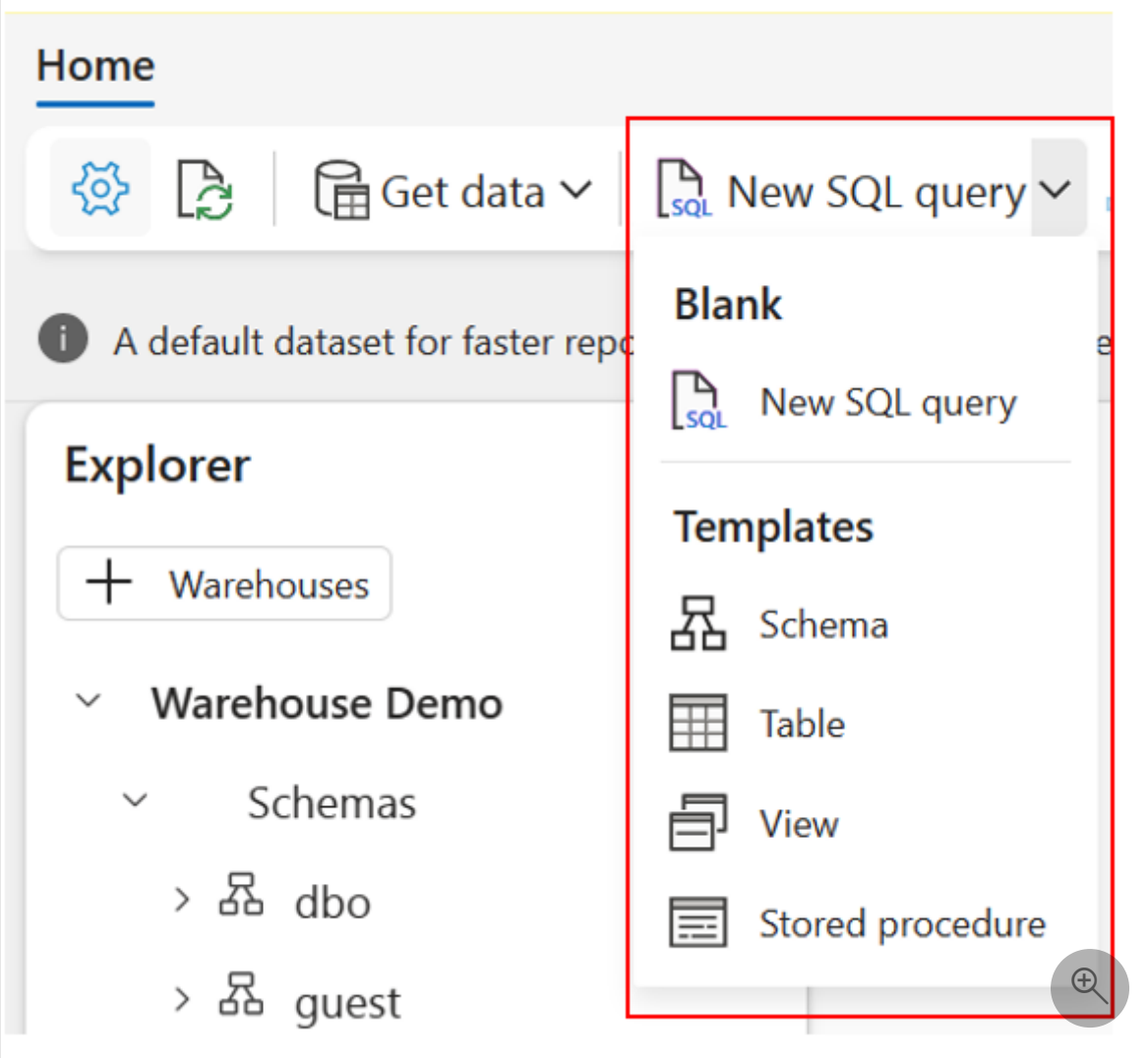
SQL query editor in the Fabric portal

The SQL query editor provides a text editor to write queries using T-SQL.

- To access the built-in SQL query editor, select the **Query** icon located at the bottom of the warehouse editor window.



- Alternatively, in the warehouse editor ribbon, create a new query using the **New SQL query** button. If you select the dropdown, you can easily create T-SQL objects with code templates that will populate in your SQL query window, as shown in the following image.



The SQL query editor provides support for IntelliSense, code completion, syntax highlighting, client-side parsing, and validation. You can run Data Definition Language (DDL), Data Manipulation Language (DML) and Data Control Language (DCL) statements. For more information on limitations for Transaction Control Language statement runs, see Limitations.

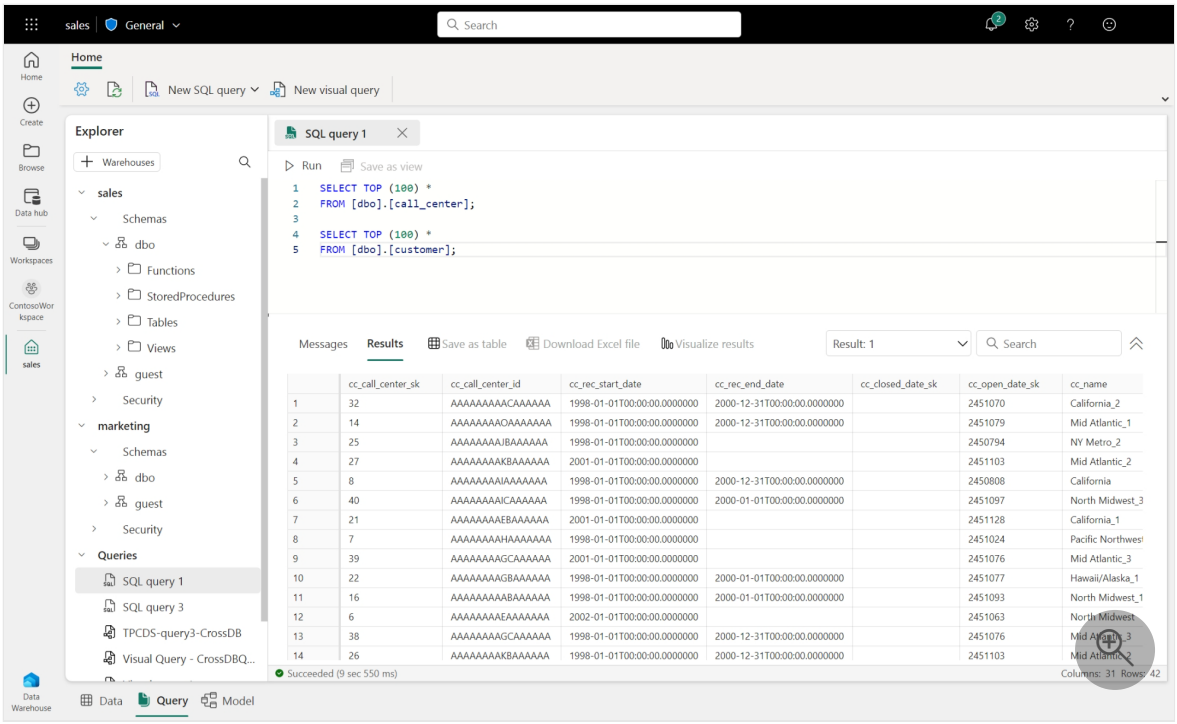
View query results

Once you've written the T-SQL query, select **Run** to execute the query.

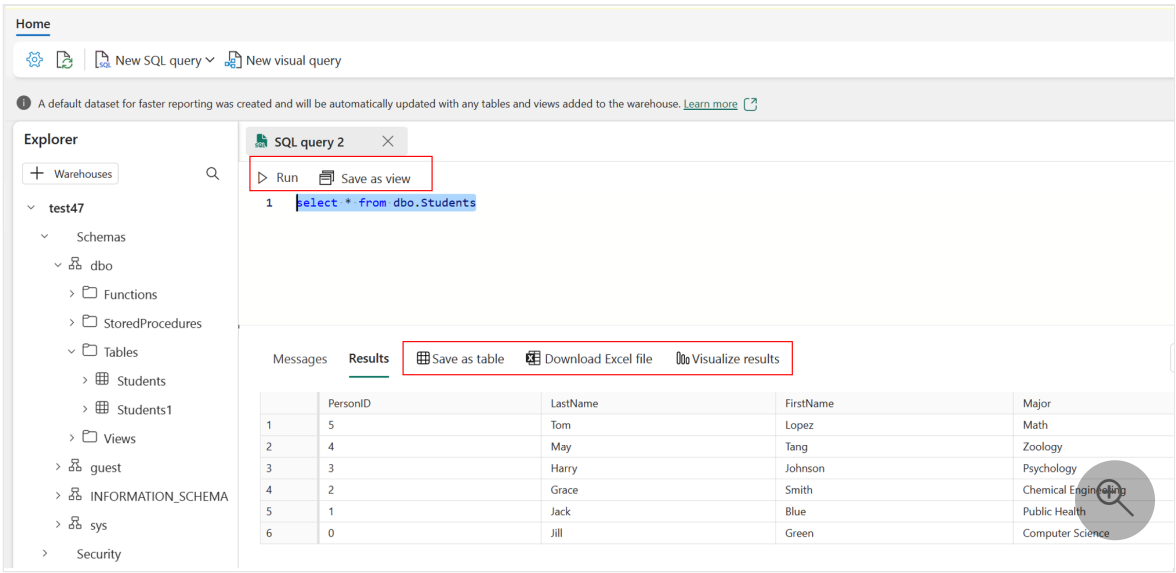
The **Results** preview is displayed in the **Results** section. If number of rows returned is more than 10,000 rows, the preview is limited to 10,000 rows. You can search string within results grid to get filtered rows matching search criteria. The **Messages** tab shows SQL messages returned when SQL query is run.

The status bar indicates the query status, duration of the run and number of rows and columns returned in results.

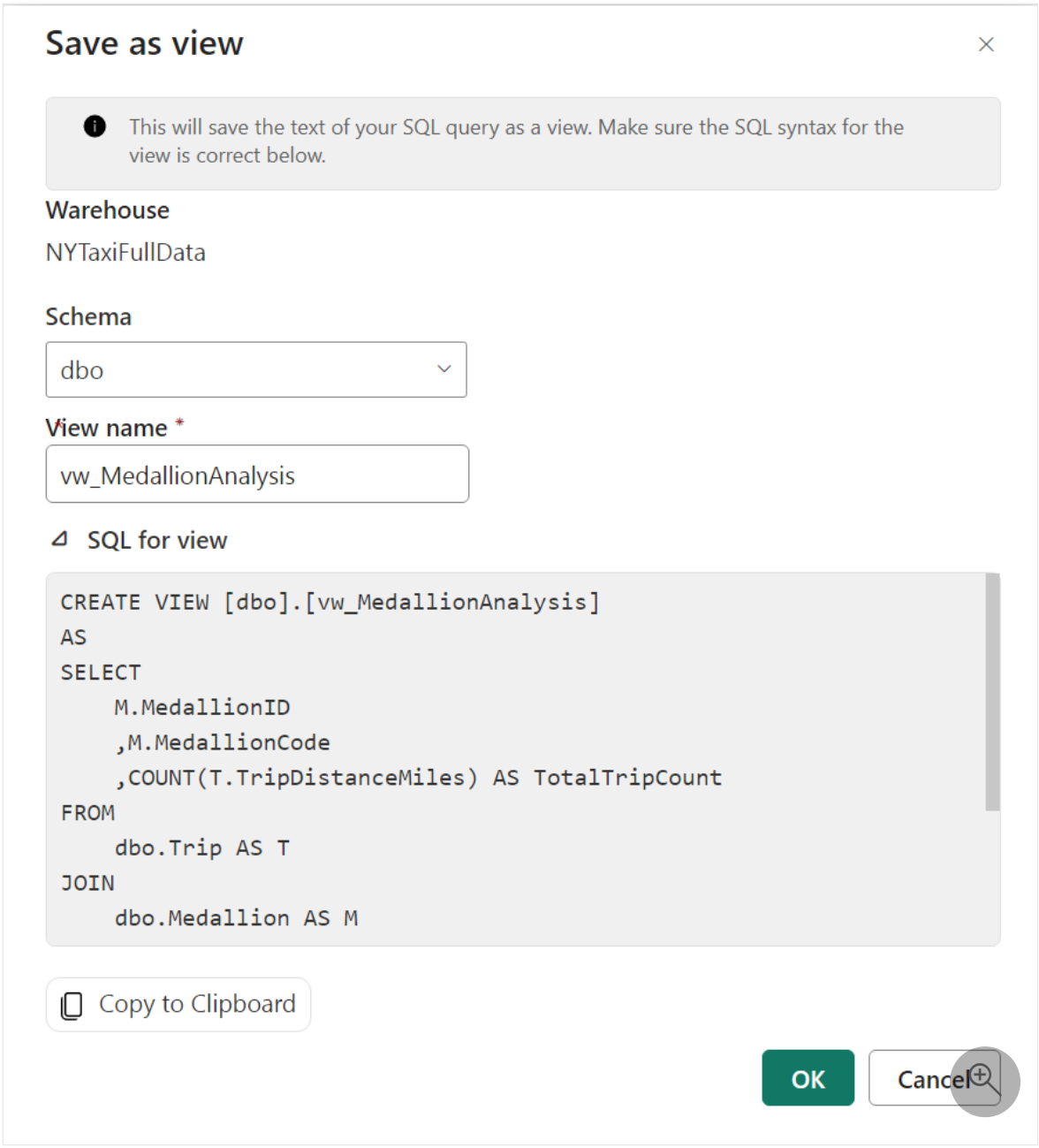
- When you run multiple queries and those return multiple results, you can select results drop down to see individual results.



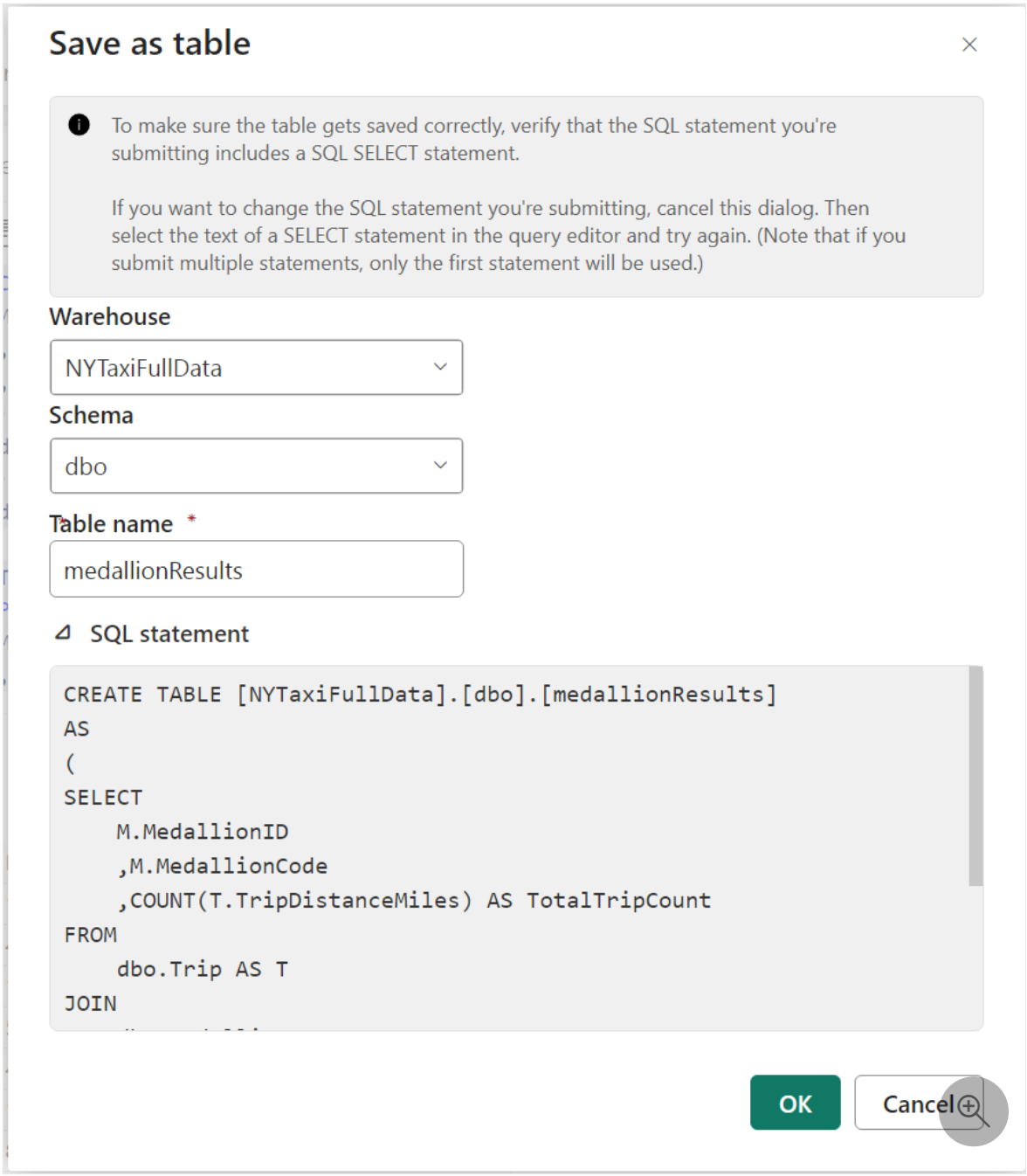
- To enable **Save as view**, **Save as table**, **Download Excel file**, and **Visualize results** menus, select the SQL statement containing SELECT statement in the SQL query editor.



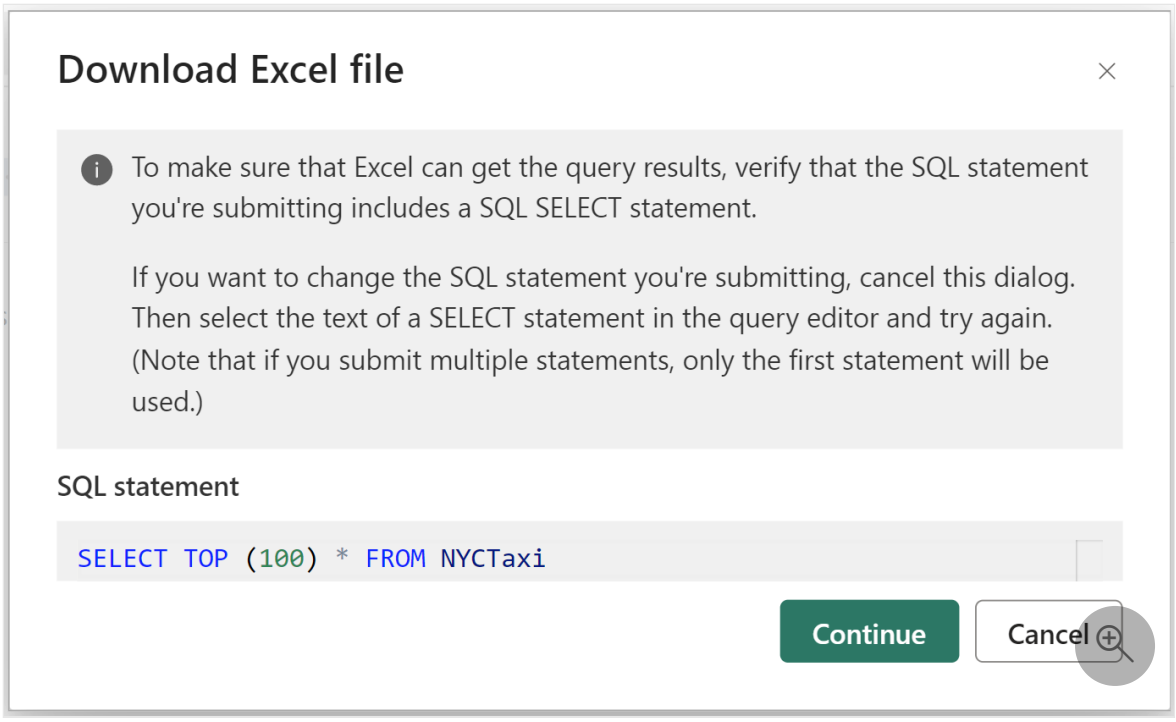
- You can select the query and save your query as a view using the **Save as view** button. Select the schema name, provide name of view and verify the SQL statement before confirming creating view. When view is successfully created, it will appear in the Explorer.



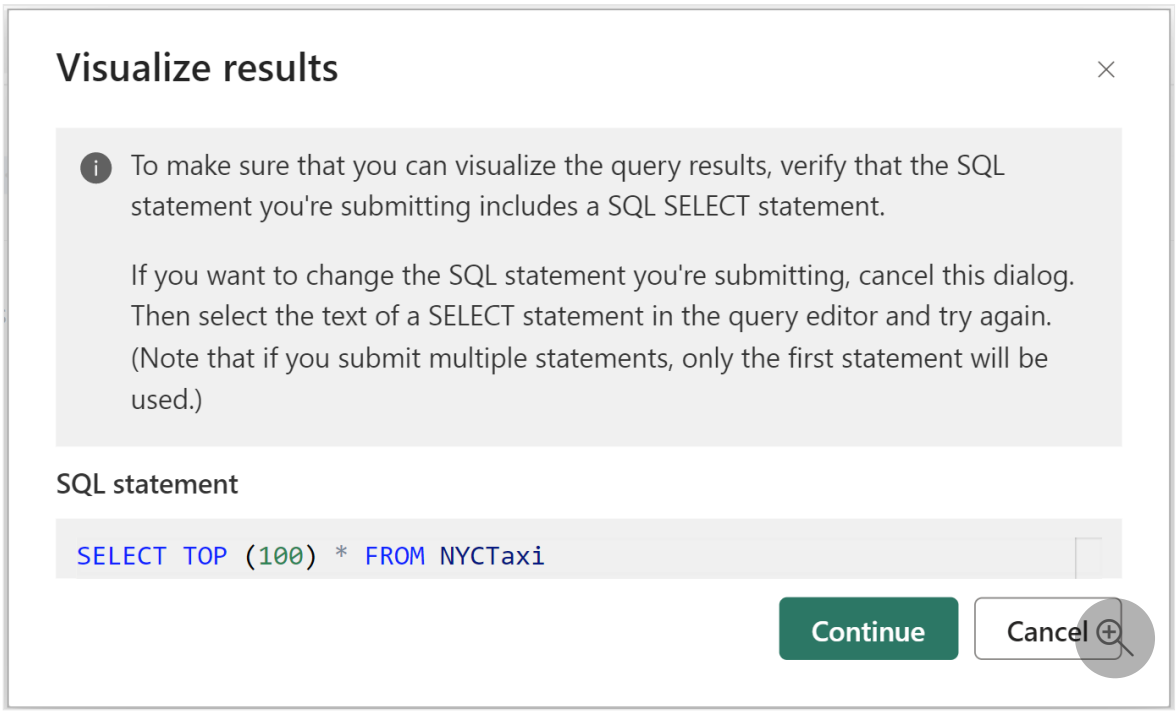
- You can use **Save as table** to save your query results into a table. Select the warehouse in which you would like to save results, select schema and provide table name to load results into the table using CREATE TABLE AS SELECT statement. When table is successfully created, it will appear in the Explorer.



- The **Download Excel file** button opens the corresponding T-SQL Query to Excel and executes the query, enabling you to view the results in Excel.



- **Visualize results** allows you to create reports from your query results within the SQL query editor.



As you work on your SQL query, the queries are automatically saved every few seconds. A "saving" indicator appears in your query tab at the bottom to indicate that your query is being saved.

Cross-warehouse querying

For more information on cross-warehouse querying, see Cross-warehouse querying.

You can write a T-SQL query with three-part naming convention to refer to objects and join them across warehouses, for example:

```sql
SELECT
    emp.Employee
    ,SUM(Profit) AS TotalProfit
    ,SUM(Quantity) AS TotalQuantitySold
FROM
    [SampleWarehouse].[dbo].[DimEmployee] as emp
JOIN
    [WWI_Sample].[dbo].[FactSale] as sale
ON
    emp.EmployeeKey = sale.SalespersonKey
WHERE
    emp.IsSalesperson = 'TRUE'
GROUP BY
    emp.Employee
ORDER BY
    TotalProfit DESC;
```

Keyboard shortcuts

Keyboard shortcuts provide a quick way to navigate and allow users to work more efficiently in SQL query editor. The table in this article lists all the shortcuts available in SQL query editor in the Microsoft Fabric portal:

Function	Shortcut
New SQL query	Ctrl + Q
Close current tab	Ctrl + Shift + F4
Run SQL script	Ctrl + Enter, Shift +Enter
Cancel running SQL script	Alt+Break
Search string	Ctrl + F
Replace string	Ctrl + H
Undo	Ctrl + Z
Redo	Ctrl + Y
Go one word left	Ctrl + Left arrow key
Go one word right	Ctrl + Right arrow key

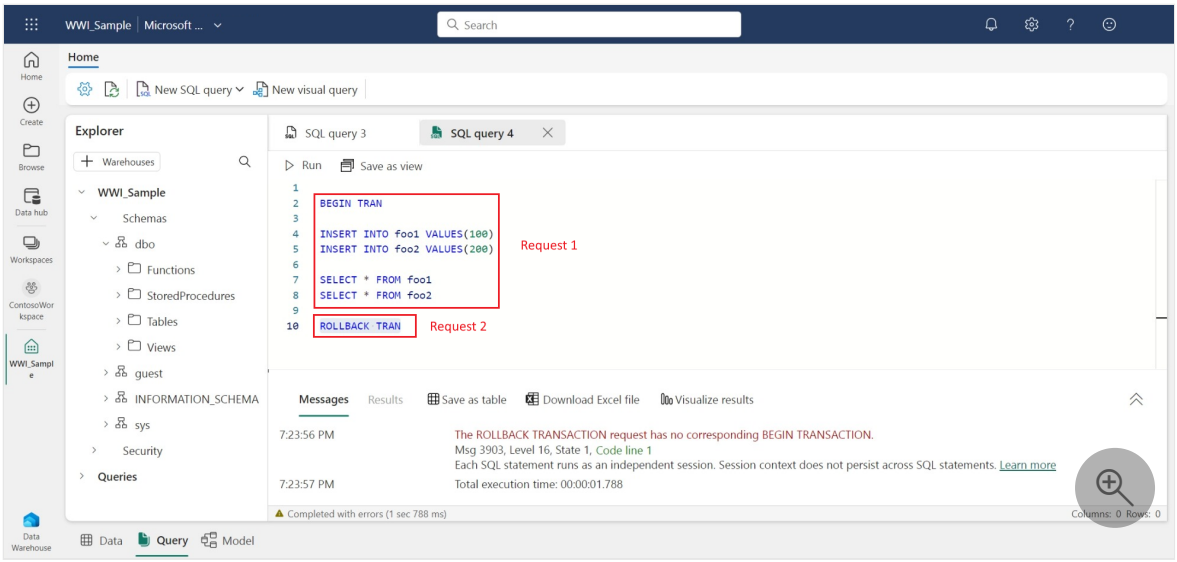
Function	Shortcut
Indent increase	Tab
Indent decrease	Shift + Tab
Comment	Ctrl + K, Ctrl + C
Uncomment	Ctrl + K, Ctrl + U
Move cursor up	↑
Move cursor down	↓
Select All	Ctrl + A

Limitations

- In SQL query editor, every time you run the query, it opens a separate session and closes it at the end of the execution. This means if you set up session context for multiple query runs, the context is not maintained for independent execution of queries.

- In SQL query editor, when you select **Run** button, you are submitting an independent batch request to execute. The SQL query editor does not support `sp_set_session_context`. Each **Run** action in the SQL query editor is a batch request, and a session only exists per batch. Each execution of code in the same query window will run in a different batch and session.

  For example, when independently executing transaction statements, session context is not retained. In the following screenshot, `BEGIN TRAN` was executed in the first request, but since the second request was executed in a different session, there is no transaction to commit, resulting into the failure of commit/rollback operation. If the SQL batch submitted does not include a COMMIT TRAN, the changes applied after `BEGIN TRAN` will not commit.



Similarly, in the SQL query editor, the `GO` SQL command creates a new independent batch in a new session.

- When you are running a SQL query with USE, you need to submit the SQL query with `USE` as one single request.
- The following table summarizes the expected behavior will not match with SQL Server Management Studio/Azure Data Studio:

Scenario	Supported in SSMS/ADS	Supported in SQL query editor in Fabric portal
Using SET Statements (Transact-SQL) to set properties for session	Yes	No
Using sp_set_session_context (Transact-SQL) for multiple batch statements runs	Yes	No
Transactions (Transact-SQL) (unless executed as a single batch request)	Yes	No

Next steps

- How-to: Query the Warehouse
- Query using the Visual Query editor

View data in the Data preview in Microsoft Fabric

Article • 05/23/2023

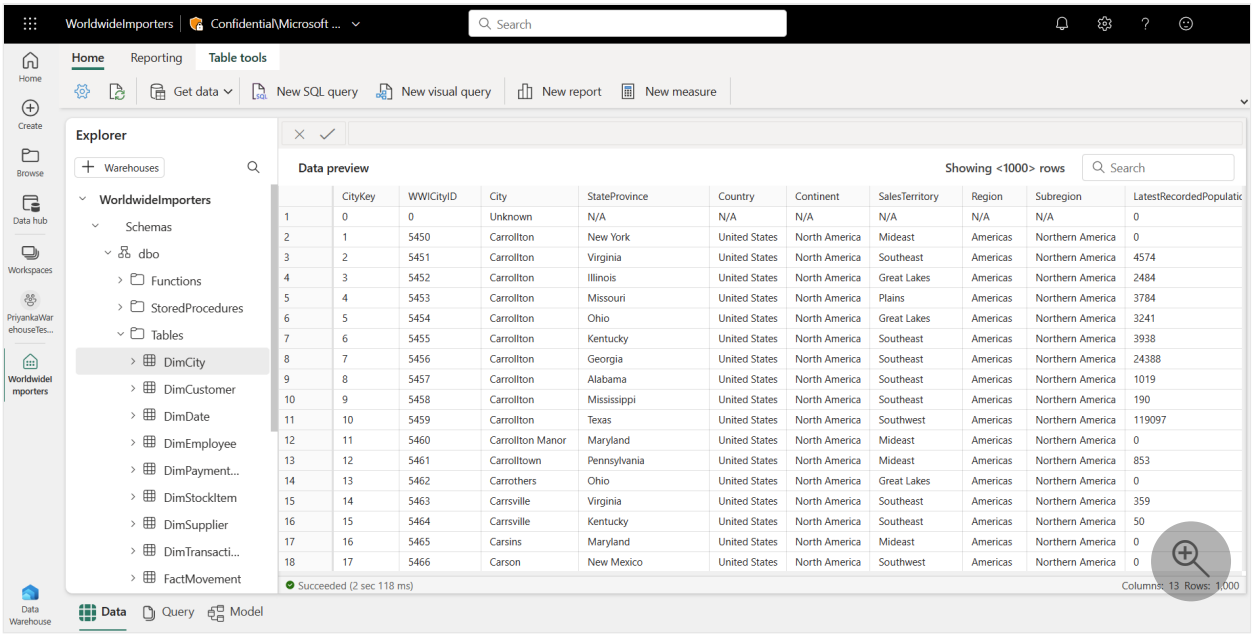
Applies to: SQL Endpoint and Warehouse in Microsoft Fabric

The **Data preview** is one of the three switcher modes along with the Query editor and Model view within the warehouse experience that provides an easy interface to view the data within your tables or views to preview sample data (top 1000 rows).

> **Important**
>
> Microsoft Fabric is currently in PREVIEW. This information relates to a prerelease product that may be substantially modified before it's released. Microsoft makes no warranties, expressed or implied, with respect to the information provided here.

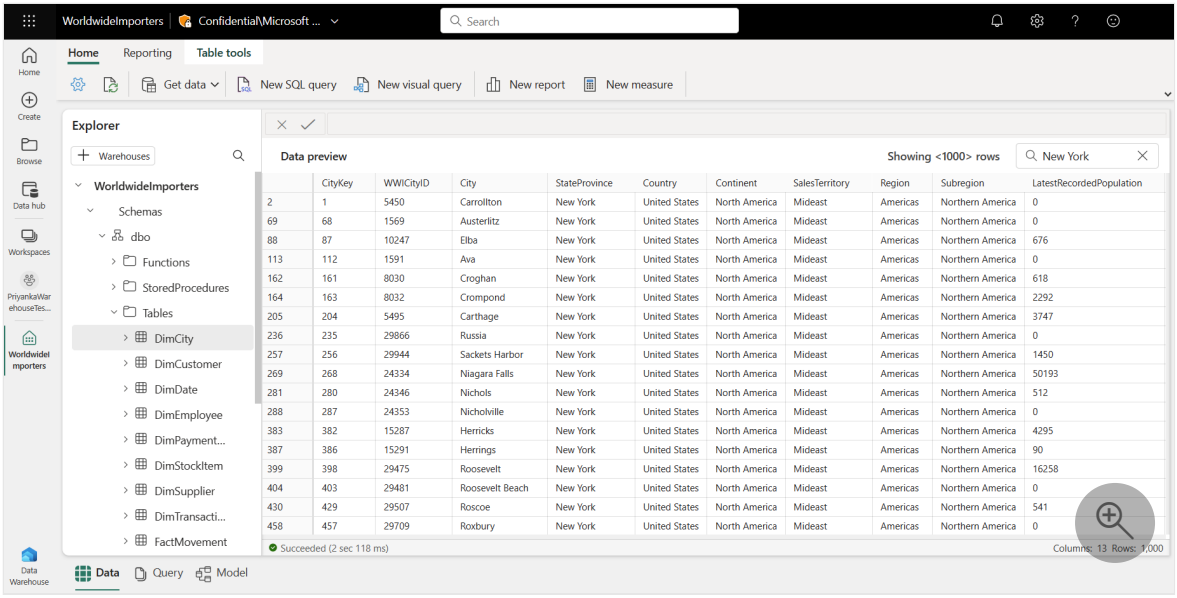
Get started

After creating a warehouse and ingesting data, select the **Data** tab. Choose a specific table or view you would like to display in the data grid of the Data preview page.

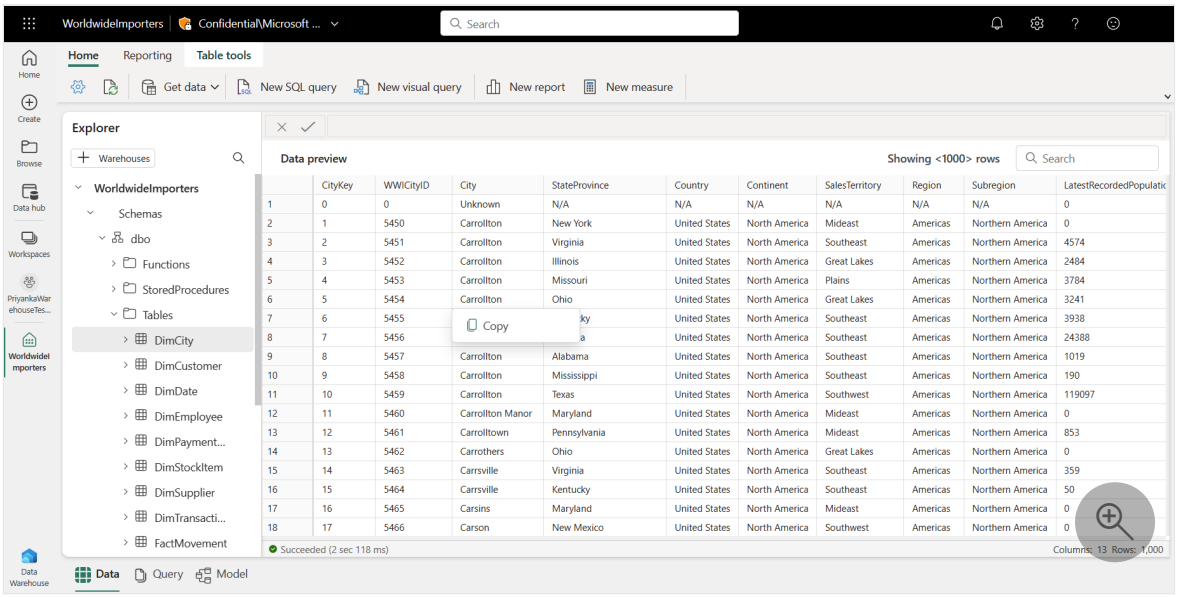


- **Search value** – Type in a specific keyword in the search bar and rows with that specific keyword will be filtered. In this example, "New York" is the keyword and only rows containing this keyword are shown. To clear the search, select on the `X` inside the search bar.

- **Sort columns (alphabetically or numerically)** – Hover over the column title and select on the up/down arrow that appears next to the title.



- **Copy value** – Right-click a cell within the table and a **Copy** option will appear to copy the specific selection.



Considerations and limitations

- Only the top 1000 rows can be shown in the data grid of the Data preview.
- The Data preview view will change depending on how the columns are sorted or if there's a keyword that is searched.

Next steps

- Define relationships in data models for data warehousing

- Data modeling in the default Power BI dataset

Delta Lake logs in Warehouse in Microsoft Fabric

Article • 05/23/2023

Applies to: Warehouse in Microsoft Fabric

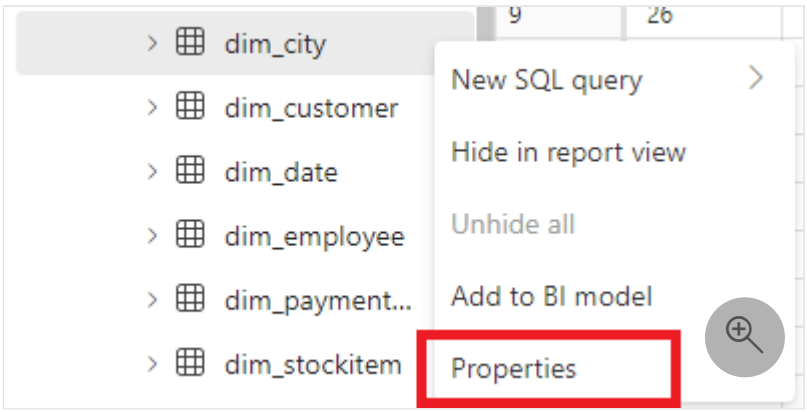
Warehouse in Microsoft Fabric is built up open file formats. User tables are stored in parquet file format, and Delta Lake logs are published for all user tables.

The Delta Lake logs opens up direct access to the warehouse's user tables for any engine that can read Delta Lake tables. This access is limited to read-only to ensure the user data maintains ACID transaction compliance. All inserts, updates, and deletes to the data in the tables must be executed through the Warehouse. Once a transaction is committed, a system background process is initiated to publish the updated Delta Lake log for the affected tables.

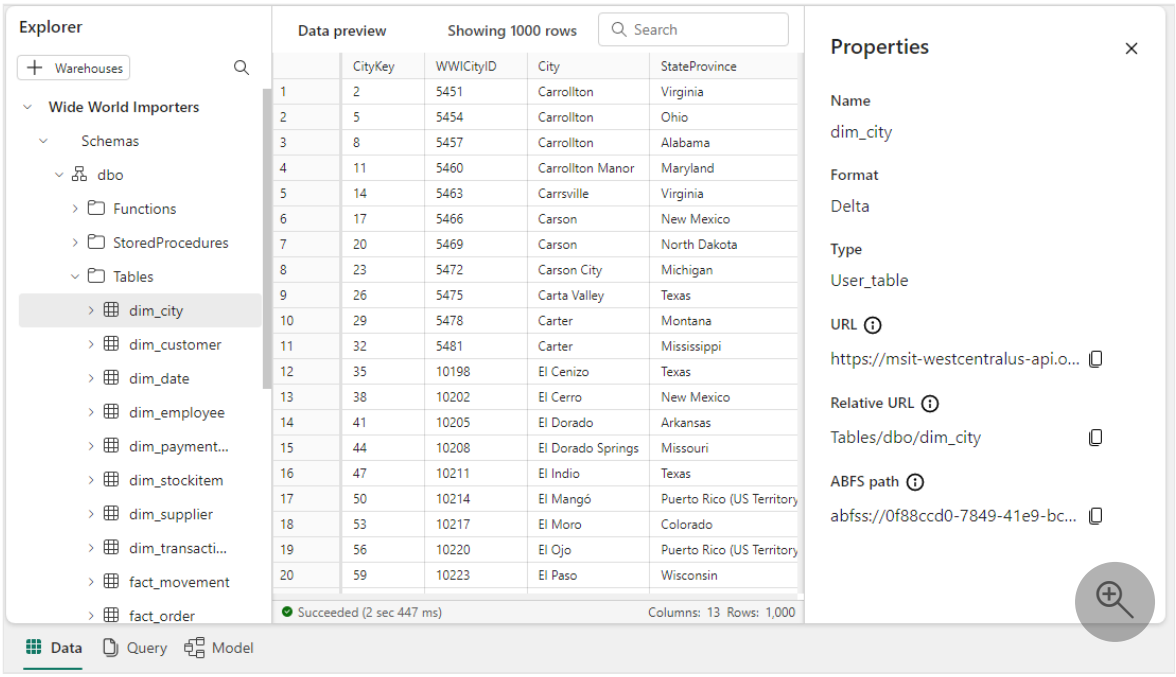
How to get OneLake path

The following steps detail how to get the OneLake path from a table in a warehouse:

1. Open **Warehouse** in your Microsoft Fabric workspace.
2. In the **Object Explorer**, you find more options **(...)** on a selected table in the **Tables** folder. Select the **Properties** menu.



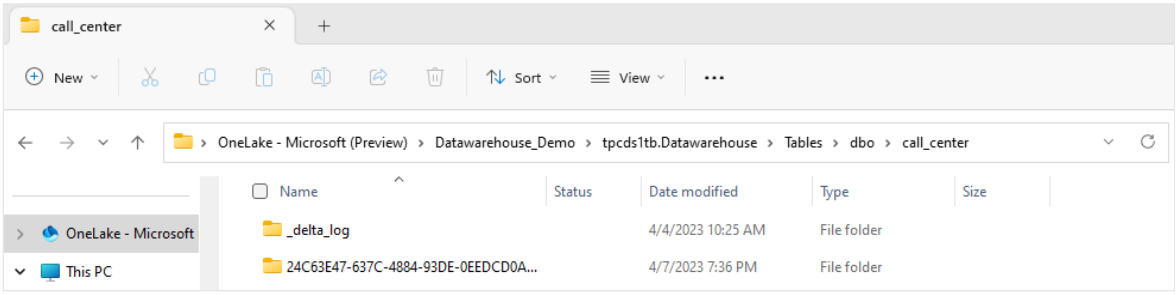
3. On selection, the **Properties** pane shows the following information:
   a. Name
   b. Format
   c. Type
   d. URL
   e. Relative path
   f. ABFS path



How to get Delta Lake logs path

You can locate Delta Lake logs via the following methods:

- Delta Lake logs can be queried through shortcuts created in a lakehouse. You can view the files using a Microsoft Fabric Spark Notebook or the Lakehouse explorer in Synapse Data Engineering in the Microsoft Fabric portal.
- Delta Lake logs can be found via Azure Storage Explorer, through Spark connections such as the Power BI Direct Lake mode, or using any other service that can read delta tables.
- Delta Lake logs can be found in the `_delta_log` folder of each table through the OneLake Explorer (Preview) in Windows, as shown in the following screenshot.



Limitations

- Currently, tables with inserts only are supported.
- Currently, Delta Lake log checkpoint and vacuum functions are unavailable.

- Table Names can only be used by Spark and other systems if they only contain these characters: A-Z a-z 0-9 and underscores.
- Column Names that will be used by Spark and other systems cannot contain:
  - spaces
  - tabs
  - carriage returns
  - [
  - ,
  - ;
  - {
  - }
  - (
  -)
  - =
  -]

Next steps

- Query the Warehouse
- How to use Microsoft Fabric notebooks
- OneLake overview
- Accessing shortcuts
- Navigate the Fabric Lakehouse explorer

Statistics in Fabric data warehousing

Article • 05/23/2023

Applies to: ✅ SQL Endpoint and Warehouse in Microsoft Fabric

The Warehouse in Microsoft Fabric uses a query engine to create an execution plan for a given SQL query. When you submit a query, the query optimizer tries to enumerate all possible plans and choose the most efficient candidate. To determine which plan would require the least overhead (I/O and memory), the engine needs to be able to evaluate the amount of work or rows that might be processed at each operator. Then, based on each plan's cost, it chooses the one with the least amount of estimated work. Statistics are objects that contain relevant information about your data, to allow query optimizer to estimate these costs.

> **ⓘ Important**
>
> Microsoft Fabric is currently in PREVIEW. This information relates to a prerelease product that may be substantially modified before it's released. Microsoft makes no warranties, expressed or implied, with respect to the information provided here.

How to leverage statistics

To achieve optimal query performance, it is important to have accurate statistics. Microsoft Fabric currently supports the following paths to provide relevant and up-to-date statistics:

- User-defined statistics
  - User issues DDL to create, update, and drop statistics as needed
- Automatic statistics
  - Engine automatically creates statistics at querytime

Manual statistics for all tables

The traditional option of maintaining statistics health is available in Microsoft Fabric. Users can create, update, and drop histogram-based single-column statistics with CREATE STATISTICS, UPDATE STATISTICS, and DROP STATISTICS, respectively. Users can also view the contents of histogram-based single-column statistics with DBCC SHOW_STATISTICS. Currently, a limited version of these statements is supported.

- If creating statistics manually, consider focusing on those heavily used in your query workload (specifically in GROUP BYs, ORDER BYs, filters, and JOINs).
- Consider updating column-level statistics regularly after data changes that significantly change rowcount or distribution of the data.

Examples of manual statistics maintenance

To create statistics on the `dbo.DimCustomer` table, based on all the rows in a column `CustomerKey`:

SQL

```
CREATE STATISTICS DimCustomer_CustomerKey_FullScan
ON dbo.DimCustomer (CustomerKey) WITH FULLSCAN;
```

To manually update the statistics object `DimCustomer_CustomerKey_FullScan`, perhaps after a large data update:

SQL

```
UPDATE STATISTICS DimCustomer_CustomerKey_FullScan (CustomerKey) WITH
FULLSCAN;
```

To show information about the statistics object:

SQL

```
DBCC SHOW_STATISTICS ("dbo.DimCustomer",
"DimCustomer_CustomerKey_FullScan");
```

To show only information about the histogram of the statistics object:

SQL

```
DBCC SHOW_STATISTICS ("dbo.DimCustomer", "DimCustomer_CustomerKey_FullScan")
WITH HISTOGRAM;
```

To manually drop the statistics object `DimCustomer_CustomerKey_FullScan`:

SQL

```
DROP STATISTICS dbo.DimCustomer.DimCustomer_CustomerKey_FullScan;
```

The following T-SQL objects can also be used to check both manually created and automatically created statistics in Microsoft Fabric:

- sys.stats catalog view
- sys.stats_columns catalog view
- STATS_DATE system function

Automatic statistics at query

Whenever you issue a query and query optimizer requires statistics for plan exploration, Microsoft Fabric will automatically create those statistics if they don't already exist. Once statistics have been created, query optimizer can utilize them in estimating the plan costs of the triggering query. Because this creation is done synchronously, you can expect the first query run to include this statistics creation time.

To verify automatic statistics creation at querytime

There are various cases where you can expect some type of statistics to be automatically created. The most common are histogram-based system statistics, which are often created for columns referenced in GROUP BYs, JOINs, DISTINCT clauses, filters (WHERE clauses), and ORDER BYs. For example, if you want to see the automatic creation of these statistics, a query will trigger creation if statistics for `COLUMN_NAME` do not yet exist. For example:

SQL

```
SELECT <COLUMN_NAME>
FROM <YOUR_TABLE_NAME>
GROUP BY <COLUMN_NAME>;
```

In this case, you should expect that statistics for `COLUMN_NAME` to have been created. If the column was also a varchar column, you would also see average column length statistics created. If you'd like to validate statistics were automatically created, you can run the following query:

SQL

```
select
    object_name(s.object_id) AS [object_name],
    c.name AS [column_name],
    s.name AS [stats_name],
    s.stats_id,
    STATS_DATE(s.object_id, s.stats_id) AS [stats_update_date],
```

```
    s.auto_created,
    s.user_created,
    s.stats_generation_method_desc
FROM sys.stats AS s
INNER JOIN sys.objects AS o
ON o.object_id = s.object_id
INNER JOIN sys.stats_columns AS sc
ON s.object_id = sc.object_id
AND s.stats_id = sc.stats_id
INNER JOIN sys.columns AS c
ON sc.object_id = c.object_id
AND c.column_id = sc.column_id
WHERE o.type = 'U' -- Only check for stats on user-tables
    AND s.auto_created = 1
    AND o.name = '<YOUR_TABLE_NAME>'
ORDER BY object_name, column_name;
```

This query only looks for column-based statistics. If you'd like to see all statistics that exist for this table, remove the JOINs on `sys.stats_columns` and `sys.columns`.

Now, you can find the `statistics_name` of the automatically generated histogram statistic (should be something like `_WA_Sys_00000007_3B75D760`) and run the following T-SQL:

SQL

```
DBCC SHOW_STATISTICS ('<YOUR_TABLE_NAME>', '<statistics_name>');
```

For example:

SQL

```
DBCC SHOW_STATISTICS ('sales.FactInvoice', '_WA_Sys_00000007_3B75D760');
```

The `Updated` value in the result set of DBCC SHOW_STATISTICS should be a date (in UTC) similar to when you ran the original GROUP BY query.

> **Note**
>
> Microsoft Fabric does not currently support the automatic update of statistics at querytime.

Types of automatically generated statistics

In Microsoft Fabric, there are multiple types of statistics that are automatically generated by the engine to improve query plans. Currently, they can be found in sys.stats although not all are actionable:

- Histogram statistics
  - Created per column needing histogram statistics at querytime
  - These objects contain histogram and density information regarding the distribution of a particular column. Similar to the statistics automatically created at querytime in Azure Synapse Analytics dedicated pools.
  - Name begins with `_WA_Sys_`.
  - Contents can be viewed with DBCC SHOW_STATISTICS
- Average column length statistics
  - Created for character columns (char and varchar) needing average column length at querytime.
  - These objects contain a value representing the average row size of the varchar column at the time of statistics creation.
  - Name begins with `ACE-AverageColumnLength_`.
  - Contents cannot be viewed and are nonactionable by user.
- Table-based cardinality statistics
  - Created per table needing cardinality estimation at querytime.
  - These objects contain an estimate of the rowcount of a table.
  - Named `ACE-Cardinality`.
  - Contents cannot be viewed and are nonactionable by user.

Limitations

- Only single-column histogram statistics can be manually created and modified.
- Multi-column statistics creation is not supported.
- Statistics (of any kind) are not currently supported for varchar(max).
- Other statistics objects may show under sys.stats aside from manually created statistics and automatically created statistics. These objects are not used for query optimization.

Next steps

- Monitoring connections, sessions, and requests using DMVs

Workload management

Article • 05/23/2023

Applies to: SQL Endpoint and Warehouse in Microsoft Fabric

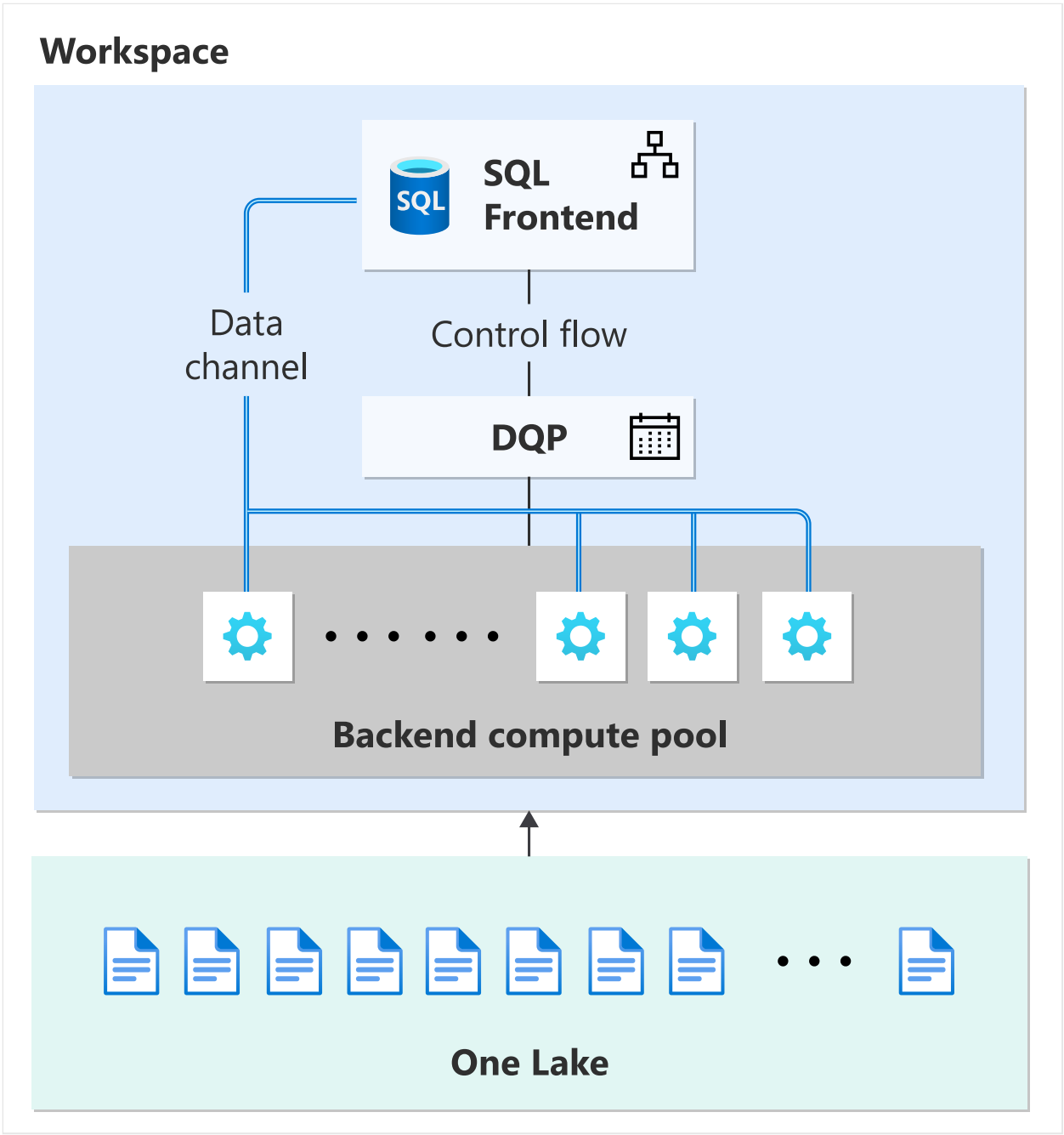
This article describes the architecture and workload management behind data warehousing in Microsoft Fabric.

> **Important**
>
> Microsoft Fabric is currently in PREVIEW. This information relates to a prerelease product that may be substantially modified before it's released. Microsoft makes no warranties, expressed or implied, with respect to the information provided here.

Data processing

The Warehouse and SQL Endpoint share the same underlying processing architecture. As data is retrieved or ingested, it leverages a distributed engine built for both small and large-scale data and computational functions.

The processing system is serverless in that backend compute capacity scales up and down autonomously to meet workload demands.

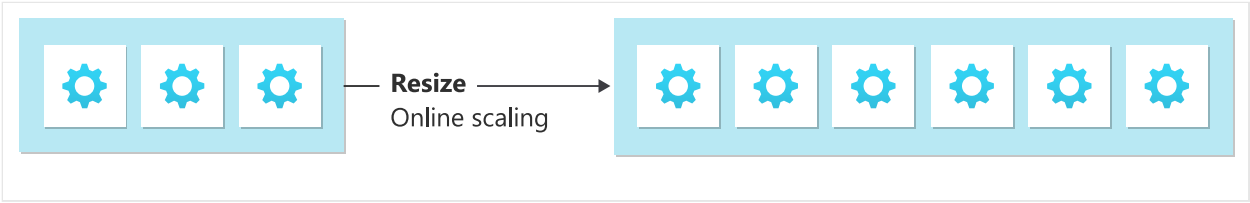


When a query is submitted, the SQL frontend (FE) performs query optimization to determine the best plan based on the data size and complexity. Once the plan is generated, it is given to the Distributed Query Processing (DQP) engine. The DQP orchestrates distributed execution of the query by splitting it into smaller queries that are executed on backend compute nodes. Each small query is called a **task** and represents a distributed execution unit. It reads file(s) from OneLake, joins results from other tasks, groups, or orders data retrieved from other tasks. For ingestion jobs, it also writes data to the proper destination tables.

When data is processed, results are returned to the SQL frontend for serving back to the user or calling application.

Elasticity and resiliency

Backend compute capacity benefits from a fast provisioning architecture. Although there is no SLA on resource assignment, typically new nodes are acquired within a few seconds. As resource demand increases, new workloads leverage the scaled-out capacity. Scaling is an online operation and query processing goes uninterrupted.



The system is fault tolerant and if a node becomes unhealthy, operations executing on the node are redistributed to healthy nodes for completion.

Scheduling and resourcing

The distributed query processing scheduler operates at a **task** level. Queries are represented to the scheduler as a directed acyclic graph (DAG) of tasks. This concept is familiar to Spark users. A DAG allows for parallelism and concurrency as tasks that do not depend on each other can be executed simultaneously or out of order.

As queries arrive, their tasks are scheduled based on first-in-first-out (FIFO) principles. If there is idle capacity, the scheduler may use a "best fit" approach to optimize concurrency.

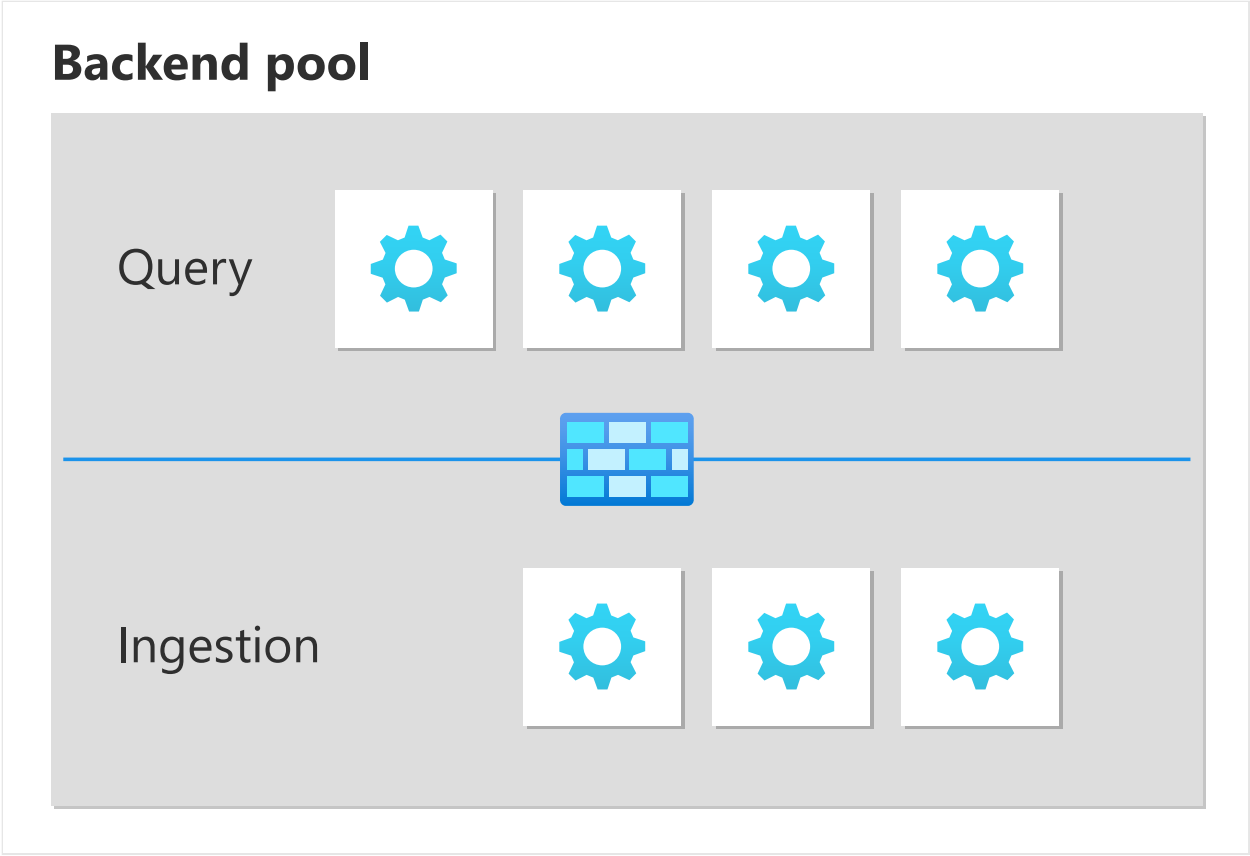
When the scheduler identifies resourcing pressure, it invokes a scale operation. Scaling is managed autonomously and backend topology grows as concurrency increases. As it takes a few seconds to acquire nodes, the system is not optimized for consistent subsecond performance of queries that require distributed processing.

When pressure subsides, backend topology scales back down and releases resource back to the region.

Ingestion isolation

Applies to: ✅ Warehouse in Microsoft Fabric

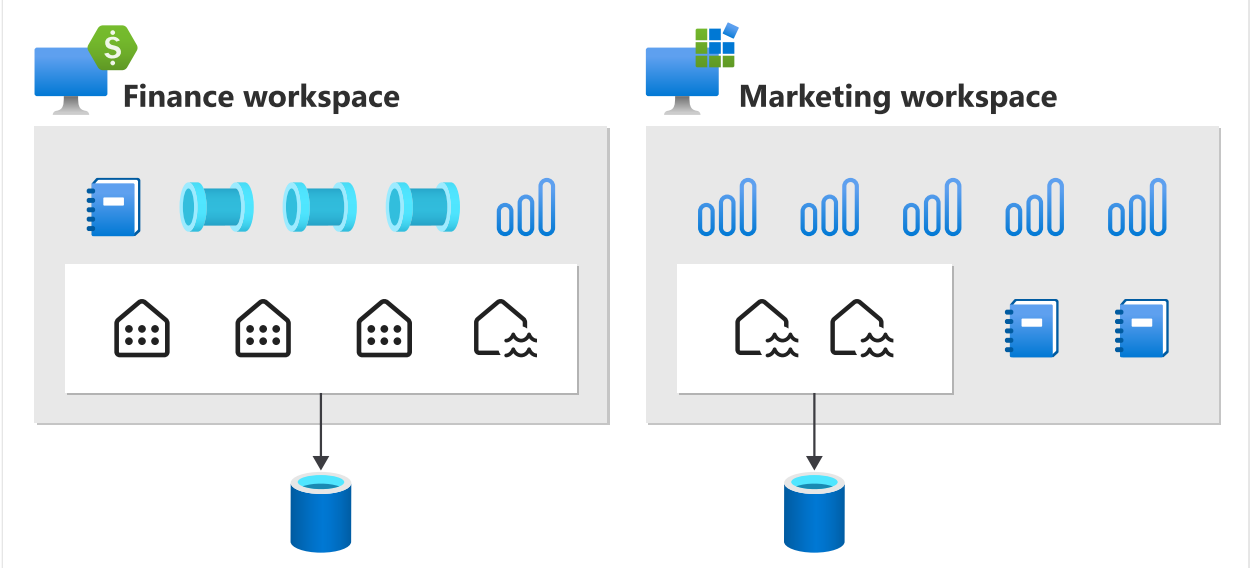
In the backend compute pool of Warehouse in Microsoft Fabric, loading activities are provided resource isolation from analytical workloads. This improves performance and reliability, as ingestion jobs can run on dedicated nodes that are optimized for ETL and do not compete with other queries or applications for resources.



Best practices

The Microsoft Fabric workspace provides a natural isolation boundary of the distributed compute system. Workloads can take advantage of this boundary to manage both cost and performance.

OneLake shortcuts can be used to create read-only replicas of tables in other workspaces to distribute load across multiple sql engines creating an isolation boundary.



Next steps

- OneLake overview
- Data warehousing
- Better together: the lakehouse and warehouse in Microsoft Fabric

Monitor connections, sessions, and requests using DMVs

Article • 05/23/2023

Applies to: SQL Endpoint and Warehouse in Microsoft Fabric

You can use existing dynamic management views (DMVs) to monitor connection, session, and request status in Microsoft Fabric. For more information about the tools and methods of executing T-SQL queries, see Query the Warehouse.

> **Important**
>
> Microsoft Fabric is currently in PREVIEW. This information relates to a prerelease product that may be substantially modified before it's released. Microsoft makes no warranties, expressed or implied, with respect to the information provided here.

How to monitor connections, sessions, and requests using query lifecycle DMVs

For the current version, there are three dynamic management views (DMVs) provided for you to receive live SQL query lifecycle insights.

- sys.dm_exec_connections
  - Returns information about each connection established between the warehouse and the engine.
- sys.dm_exec_sessions
  - Returns information about each session authenticated between the item and engine.
- sys.dm_exec_requests
  - Returns information about each active request in a session.

These three DMVs provide detailed insight on the following scenarios:

- Who is the user running the session?
- When was the session started by the user?
- What's the ID of the connection to the data Warehouse and the session that is running the request?
- How many queries are actively running?
- Which queries are long running?

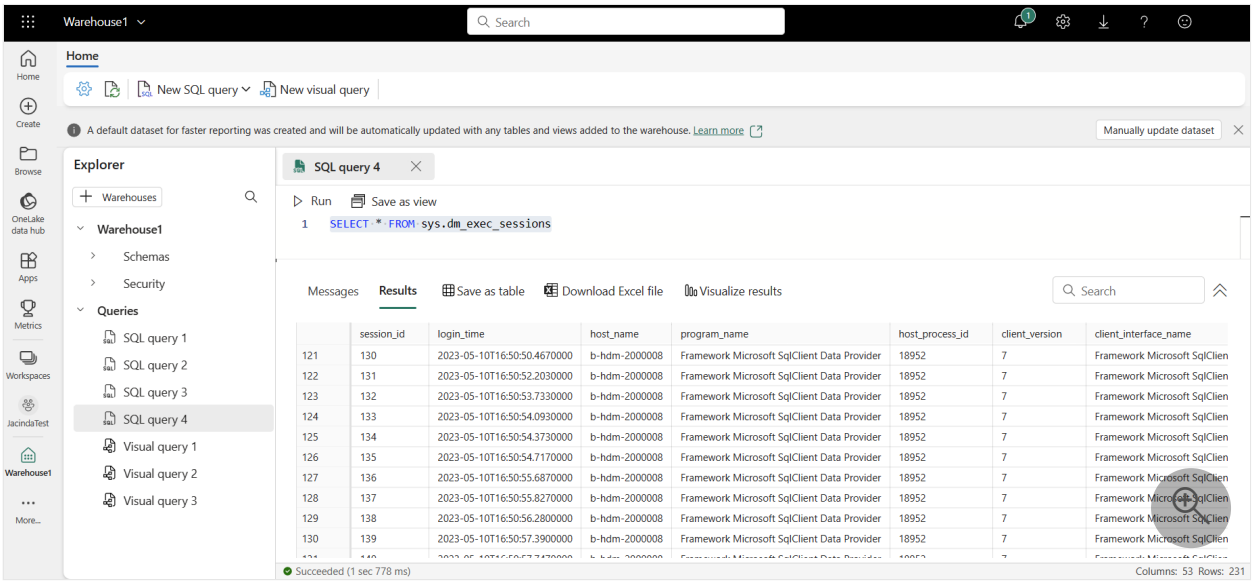
In this tutorial, learn how to monitor your running SQL queries using dynamic management views (DMVs).

Example DMV queries

The following example queries `sys.dm_exec_sessions` to find all sessions that are currently executing.

SQL

```
SELECT *
FROM sys.dm_exec_sessions;
```



Find the relationship between connections and sessions

The following example joins `sys.dm_exec_connections` and `sys.dm_exec_sessions` to the relationship between the active session in a specific connection.

SQL

```
SELECT connections.connection_id,
 connections.connect_time,
 sessions.session_id, sessions.login_name, sessions.login_time,
sessions.status
FROM sys.dm_exec_connections AS connections
INNER JOIN sys.dm_exec_sessions AS sessions
ON connections.session_id=sessions.session_id;
```

Identify and KILL a long-running query

This first query identifies the list of long-running queries in the order of which query has taken the longest since it has arrived.

```SQL
SELECT request_id, session_id, start_time, total_elapsed_time
FROM sys.dm_exec_requests
WHERE status = 'running'
ORDER BY total_elapsed_time DESC;
```

This second query shows which user ran the session that has the long-running query.

```SQL
SELECT login_name
FROM sys.dm_exec_sessions
WHERE 'session_id' = 'SESSION_ID WITH LONG-RUNNING QUERY';
```

This third query shows how to use the KILL command on the `session_id` with the long-running query.

```SQL
KILL 'SESSION_ID WITH LONG-RUNNING QUERY'
```

For example

```SQL
KILL '101'
```

Permissions

- An Admin has permissions to execute all three DMVs (`sys.dm_exec_connections`, `sys.dm_exec_sessions`, `sys.dm_exec_requests`) to see their own and others' information within a workspace.
- A Member, Contributor, and Viewer can execute `sys.dm_exec_sessions` and `sys.dm_exec_requests` and see their own results within the warehouse, but does not have permission to execute `sys.dm_exec_connections`.
- Only an Admin has permission to run the `KILL` command.

Next steps

- Query using the SQL Query editor
- Query the SQL Endpoint or Warehouse in Microsoft Fabric

Troubleshoot the Warehouse

Article • 05/23/2023

Applies to: Warehouse in Microsoft Fabric

This article provides guidance in troubleshooting common issues in Warehouse in Microsoft Fabric.

> **Important**
>
> Microsoft Fabric is currently in PREVIEW. This information relates to a prerelease product that may be substantially modified before it's released. Microsoft makes no warranties, expressed or implied, with respect to the information provided here.

Transient connection errors

A transient error, also known as a transient fault, has an underlying cause that soon resolves itself. If a connection to Warehouse used to work fine but starts to fail without changes in user permission, firewall policy, and network configuration, try these steps before contacting support:

1. Check the status of Warehouse and ensure it's not paused.
2. Don't immediately retry the failed command. Instead, wait for 5 to 10 minutes, establish a new connection, then retry the command. Occasionally Azure system quickly shifts hardware resources to better load-balance various workloads. Most of these reconfiguration events finish in less than 60 seconds. During this reconfiguration time span, you might have issues with connecting to your databases. Connection could also fail when the service is being automatically restarted to resolve certain issues.
3. Connect using a different application and/or from another machine.

Query failure due to tempdb space issue

The `tempdb` is a system database used by the engine for various temporary storage needs during query execution. It can't be accessed or configured by users. Queries could fail due to `tempdb` running out of space. Take these steps to reduce `tempdb` space usage:

1. Refer to the article about statistics to verify proper column statistics have been created on all tables.

2. Ensure all table statistics are updated after large DML transactions.
3. Queries with complex JOINs, GROUP BY, and ORDER BY and expect to return large result set use more `tempdb` space in execution. Update queries to reduce the number of GROUP BY and ORDER BY columns if possible.
4. Check for data skew in base tables.
5. Rerun the query when there's no other active queries running to avoid resource constraint during query execution.
6. Pause and resume the service to flush `tempdb` data.

Query performance seems to degrade over time

Many factors can affect a query's performance, such as changes in table size, data skew, workload concurrency, available resources, network, etc. Just because a query runs slower doesn't necessarily mean there's a query performance problem. Take following steps to investigate the target query:

1. Identify the differences in all performance-affecting factors among good and bad performance runs.
2. Refer to the article about statistics to verify proper column statistics have been created on all tables.
3. Ensure all table statistics are updated after large DML transactions.
4. Check for data skew in base tables.
5. Pause and resume the service. Then, rerun the query when there's no other active queries running. You can monitor the warehouse workload using DMV.

Query fails after running for a long time. No data is returned to the client.

A SELECT statement could have completed successfully in the backend and fails when trying to return the query result set to the client. Try following steps to isolate the problem:

1. Use different client tools to rerun the same query.

    - SQL Server Management Studio (SSMS)
    - Azure Data Studio
    - The SQL query editor in the Microsoft Fabric portal
    - The Visual Query editor in the Microsoft Fabric portal

- SQLCMD utility (for authentication via Azure AD Universal with MFA, use parameters `-G -U`)

2. If step 1 fails, run a CTAS command with the failed SELECT statement to send the SELECT query result to another table in the same warehouse. Using CTAS avoids query result set being sent back to the client machine. If the CTAS command finishes successfully and the target table is populated, then the original query failure is likely caused by the warehouse front end or client issues.

What to collect before contacting Microsoft support

- Provide the workspace ID of Warehouse.
- Provide the Statement ID and Distributed request ID. They're returned as messages after a query completes or fails.
- Provide the text of the exact error message.
- Provide the time when the query completes or fails.

Next steps

- Monitoring connections, sessions, and requests using DMVs

Limitations and known issues in Microsoft Fabric

Article • 05/23/2023

Applies to: ✅ SQL Endpoint and Warehouse in Microsoft Fabric

This article details the current limitations and known issues in Microsoft Fabric.

> **ⓘ Important**
>
> Microsoft Fabric is currently in PREVIEW. This information relates to a prerelease product that may be substantially modified before it's released. Microsoft makes no warranties, expressed or implied, with respect to the information provided here.

Limitations

Data Warehousing in Microsoft Fabric is currently in preview. The focus of this preview is on providing a rich set of SaaS features and functionality tailored to all skill levels. The preview delivers on the promise of providing a simplified experience through an open data format over a single copy of data. This release is not focused on performance, concurrency, and scale. Additional functionality will build upon the world class, industry-leading performance and concurrency story, and will land incrementally as we progress towards General Availability of data warehousing in Microsoft Fabric.

General product limitations for Data Warehousing in Microsoft Fabric are listed in this article, with feature level limitations called out in the corresponding feature article.

- **IMPORTANT** At this time, there's limited T-SQL functionality, and certain T-SQL commands can cause warehouse corruption. See T-SQL surface area for a list of T-SQL command limitations.
- Warehouse Recovery capabilities are not available during Preview.

For more limitations information in specific areas, see:

- Data types in Microsoft Fabric
- Datasets
- Delta lake logs
- Statistics
- Transactions
- The Visual Query editor

- Connectivity
- Tables

Known issues for querying

- Queries with PIVOT operator fail if there's a GROUP BY on the nonpivot column output by PIVOT. As a workaround, remove the nonpivot column from the GROUP BY. Query results will be the same, as this GROUP BY clause is duplicate.
- Warehouse explorer doesn't list all objects of the same name but different cases.

Limitations of the SQL Endpoint

The following limitations apply to SQL Endpoint automatic schema generation and metadata discovery.

- Data should be in Delta Parquet format to be auto-discovered in the SQL Endpoint. Delta Lake is an open-source storage framework that enables building Lakehouse architecture.

- Tables with renamed columns aren't supported in the SQL Endpoint.

- Delta tables created outside of the `/tables` folder aren't available in the SQL Endpoint.

  If you don't see a Lakehouse table in the warehouse, check the location of the table. Only the tables that are referencing data in the `/tables` folder are available in the warehouse. The tables that reference data in the `/files` folder in the lake aren't exposed in the SQL Endpoint. As a workaround, move your data to the `/tables` folder.

- Some columns that exist in the Spark Delta tables might not be available in the tables in the SQL Endpoint. Refer to the Data types for a full list of supported data types.

Next steps

- Get Started with Warehouse

Transact-SQL reference (Database Engine)

Article • 02/17/2023

Applies to: SQL Server Azure SQL Database Azure SQL Managed Instance Azure Synapse Analytics Analytics Platform System (PDW)

This article gives the basics about how to find and use the Microsoft Transact-SQL (T-SQL) reference articles. T-SQL is central to using Microsoft SQL products and services. All tools and applications that communicate with a SQL Server database do so by sending T-SQL commands.

T-SQL compliance with the SQL standard

For detailed technical documents about how certain standards are implemented in SQL Server, see the Microsoft SQL Server Standards Support documentation.

Tools that use T-SQL

Some of the Microsoft tools that issue T-SQL commands are:

- SQL Server Management Studio (SSMS)
- Azure Data Studio
- SQL Server Data Tools (SSDT)
- sqlcmd

Locate the Transact-SQL reference articles

To find T-SQL articles, use search at the top right of this page, or use the table of contents on the left side of the page. You can also type a T-SQL key word in the Management Studio Query Editor window, and press F1.

Find system views

To find the system tables, views, functions, and procedures, see these links, which are in the Using relational databases section of the SQL documentation.

- System catalog Views
- System compatibility views

- System dynamic management views
- System functions
- System information schema views
- System stored procedures
- System tables

"Applies to" references

The T-SQL reference articles encompass multiple versions of SQL Server, starting with 2008, and the other Azure SQL services. Near the top of each article, is a section that indicates which products and services support subject of the article.

For example, this article applies to all versions, and has the following label.

Applies to: ✅ SQL Server ✅ Azure SQL Database ✅ Azure SQL Managed Instance ✅ Azure Synapse Analytics ✅ Analytics Platform System (PDW)

Another example, the following label indicates an article that applies only to Azure Synapse Analytics and Parallel Data Warehouse.

Applies to: ✅ Azure Synapse Analytics ✅ Analytics Platform System (PDW)

In some cases, the article is used by a product or service, but all of the arguments aren't supported. In this case, other **Applies to** sections are inserted into the appropriate argument descriptions in the body of the article.

Get help from Microsoft Q & A

For online help, see the Microsoft Q & A Transact-SQL Forum.

See other language references

The SQL docs include these other language references:

- XQuery Language Reference
- Integration Services Language Reference
- Replication Language Reference
- Analysis Services Language Reference

Next steps

- Tutorial: Writing Transact-SQL Statements
- Transact-SQL Syntax Conventions (Transact-SQL)

Microsoft Learn documentation contributor guide overview

Article • 02/16/2023

Welcome to the Microsoft Learn documentation contributor guide!

Sharing your expertise with others on Microsoft Learn helps everyone achieve more. Use the information in this guide to publish a new article to Microsoft Learn or make updates to an existing published article.

Several of the Microsoft documentation sets are open source and hosted on GitHub. Not all document sets are completely open source, but many have public-facing repos where you can suggest changes via pull requests (PR). This open-source approach streamlines and improves communication between product engineers, content teams, and customers, and it has other advantages:

- Open-source repos *plan in the open* to get feedback on what docs are most needed.
- Open-source repos *review in the open* to publish the most helpful content on our first release.
- Open-source repos *update in the open* to make it easier to continuously improve the content.

The user experience on Microsoft Learn integrates GitHub workflows directly to make it even easier. Start by editing the document you're viewing. Or help by reviewing new topics or creating quality issues.

> **Important**
>
> All repositories that publish to Microsoft Learn have adopted the **Microsoft Open Source Code of Conduct** or the **.NET Foundation Code of Conduct**. For more information, see the **Code of Conduct FAQ**. Contact **opencode@microsoft.com** or **conduct@dotnetfoundation.org** with any questions or comments.
>
> Minor corrections or clarifications to documentation and code examples in public repositories are covered by the **learn.microsoft.com Terms of Use**. New or significant changes generate a comment in the PR, asking you to submit an online Contribution License Agreement (CLA) if you're not a Microsoft employee. We need you to complete the online form before we can review or accept your PR.

Quick edits to documentation

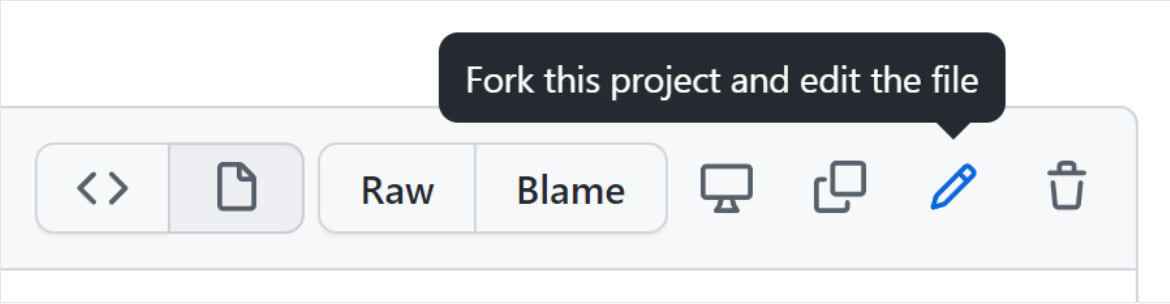
Quick edits streamline the process to report and fix small errors and omissions in documentation. Despite all efforts, small grammar and spelling errors *do* make their way into our published documents. While you can create issues to report mistakes, it's faster and easier to create a PR to fix the issue, when the option is available.

1. *Some* docs pages allow you to edit content directly in the browser. If so, you'll see an **Edit** button like the one shown below. Choosing the **Edit** (or equivalently localized) button takes you to the source file on GitHub.

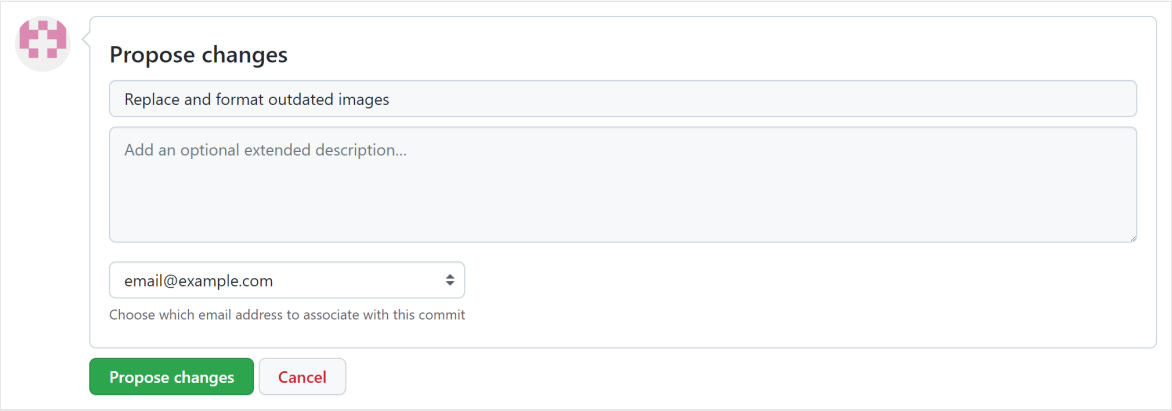
   If the **Edit** button isn't present, it means the content isn't open to public contributions. Some pages are generated (for example, from inline documentation in code) and must be edited in the project they belong to.

2. Select the pencil icon to edit the article. If the pencil icon is grayed out, you need to either log in to your GitHub account or create a new account.

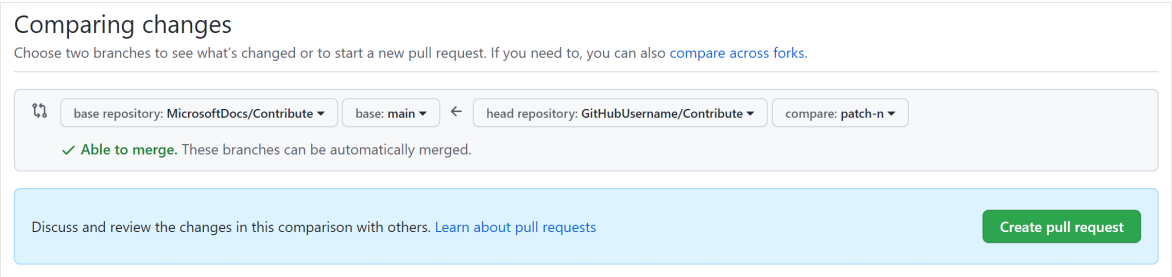
   

3. Edit the file in the web editor. Choose the **Preview** tab to check the formatting of your changes.

4. When you're finished editing, scroll to the bottom of the page. In the **Propose changes** area, enter a title and optionally a description for your changes. The title will be the first line of the commit message. Select **Propose changes** to create a new branch in your fork and commit your changes:



5. Now that you've proposed and committed your changes, you need to ask the owners of the repository to "pull" your changes into their repository. This is done using something called a "pull request" (PR). When you select **Propose changes**, a new page similar to the following is displayed:



   Select **Create pull request**. Next, enter a title and a description for the PR, and then select **Create pull request**. If you're new to GitHub, see About pull requests for more information.

6. That's it! Content team members will review your PR and merge it when it's approved. You may get feedback requesting changes.

The GitHub editing UI responds to your permissions on the repository. The preceding images are for contributors who don't have write permissions to the target repository. GitHub automatically creates a fork of the target repository in your account. The newly created fork name has the form `GitHubUsername/RepositoryName` by default. If you have write access to the target repository, such as your fork, GitHub creates a new branch in the target repository. The branch name has the default form **patch-`n`**, using a numeric identifier for the patch branch.

We use PRs for all changes, even for contributors who have write access. Most repositories protect the default branch so that updates must be submitted as PRs.

The in-browser editing experience is best for minor or infrequent changes. If you make large contributions or use advanced Git features (such as branch management or advanced merge conflict resolution), you need to fork the repo and work locally.

> **Note**
>
> Most localized documentation doesn't offer the ability to edit or provide feedback through GitHub. To provide feedback on localized content, use **https://aka.ms/provide-feedback** form.

Review open PRs

You can read new topics before they're published by checking the open PR queue. Reviews follow the GitHub flow process. You can see proposed updates or new articles in public repositories. Review them and add your comments. Look at any of our Microsoft Learn repositories, and check the open PRs for areas that interest you. Community feedback on proposed updates helps the entire community.

Create quality issues

Our docs are a continuous work in progress. Good issues help us focus our efforts on the highest priorities for the community. The more detail you can provide, the more helpful the issue. Tell us what information you sought. Tell us the search terms you used. If you can't get started, tell us how you want to start exploring unfamiliar technology.

Many of Microsoft's documentation pages have a **Feedback** section at the bottom of the page where you can choose to leave **Product feedback** or **Content feedback** to track issues that are specific to that article.

Issues start the conversation about what's needed. The content team will respond to these issues with ideas for what we can add, and ask for your opinions. When we create a draft, we'll ask you to review the PR.

Get more involved

Other topics in this guide help you get started productively contributing to Microsoft Learn. They explain working with GitHub repositories, Markdown tools, and extensions used in the Microsoft Learn content.