

Load data into a Microsoft Fabric data warehouse

1. Introduction

<https://learn.microsoft.com/en-us/training/modules/load-data-into-microsoft-fabric-data-warehouse/1-introduction>

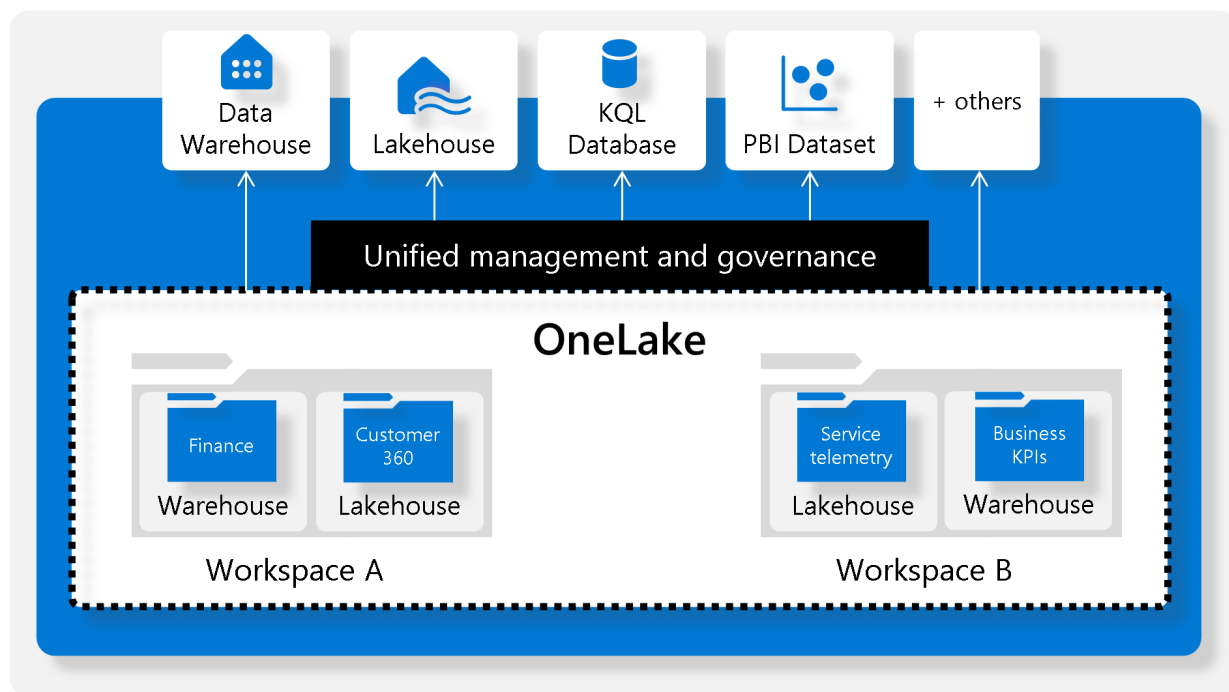
Introduction

Completed

Microsoft Fabric Data Warehouse is a complete platform for data, analytics, and AI (Artificial Intelligence). It refers to the process of storing, organizing, and managing large volumes of structured and semi-structured data.

Data warehouse in Microsoft Fabric offers a rich set of features that make it easier to manage and analyze data. It includes advanced query processing capabilities, and supports the full transactional T-SQL capabilities like an enterprise data warehouse.

Unlike a dedicated SQL pool in Synapse Analytics, a warehouse in Microsoft Fabric is centered around a single data lake. The data in the Microsoft Fabric warehouse is stored in the Parquet file format. This setup allows users to focus on tasks such as data preparation, analysis, and reporting. It takes advantage of the SQL engine's extensive capabilities, where a unique copy of their data is stored in **Microsoft OneLake**.



Understand the ETL (Extract, Transform, and Load) process

ETL provides the foundation for data analytics and data warehouse workstreams. Let's review some aspects of data manipulation in an ETL process.

	Description
Data extraction	It involves connecting to the source system and collecting necessary data for analytical processing.
Data transformation	It involves a series of steps performed on the extracted data to convert it into a standard format. Combining data from different tables, cleaning data, deduplicating data, and performing data validations.
Data loading	The extracted and transformed data are loaded into the fact and dimension tables. For an incremental load, this involves periodically applying ongoing changes as per requirement. This process often involves reformatting the data to ensure its quality and compatibility with the data warehouse schema.
Post-load optimizations	Once the data is loaded, certain optimizations can be performed to enhance the performance of the data warehouse.

All these steps in the ETL process can run in parallel depending on the scenario. As soon as some data is ready, it's loaded without waiting for the previous steps to be completed.

In the next units, we'll explore various ways of loading data in a warehouse, and how they can facilitate the tasks of building a data warehouse workload.

2. Explore data load strategies

<https://learn.microsoft.com/en-us/training/modules/load-data-into-microsoft-fabric-data-warehouse/2-explore-data-load-strategies>

Explore data load strategies

Completed

In Microsoft Fabric, there are many ways you can choose to load data in a warehouse. This step is fundamental as it ensures that high-quality, transformed, or processed data is integrated into a single repository.

Also, the efficiency of data loading directly impacts the timeliness and accuracy of analytics, making it vital for real-time decision-making processes. Investing time and resources in designing and implementing a robust data loading strategy is essential for the success of the data warehouse project.

Understand data ingestion and data load operations

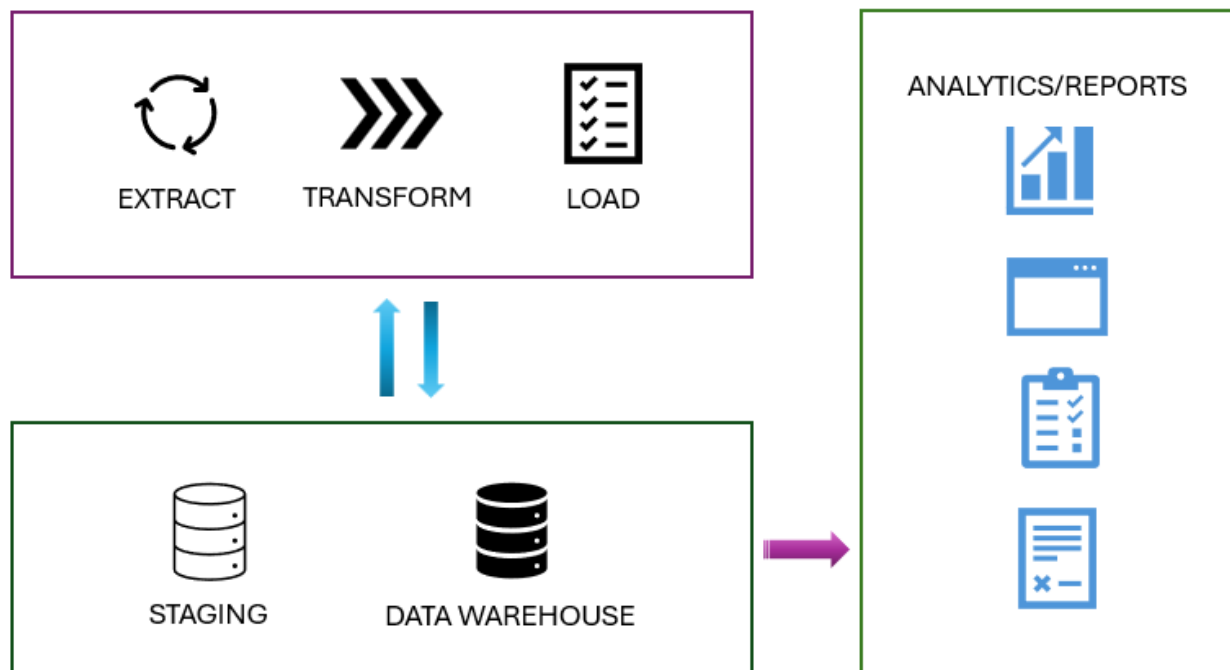
While both processes are part of the ETL (Extract, Transform, Load) pipeline in a data warehouse scenario, they usually serve different purposes. **Data ingestion/extract** is about moving raw data from various sources into a central repository. On the other hand, **data loading** involves taking the transformed or processed data and loading it into the final storage destination for analysis and reporting.

Fabric data warehouses and lakehouses automatically store their data in OneLake using the Delta Parquet format.

Stage your data

You might have to build and work with auxiliary objects involved in a load operation, such as tables, stored procedures, and functions. These auxiliary objects are commonly known as **staging**. Staging objects act as temporary storage and transformation areas. They can either share resources with a data warehouse or reside in their own storage area.

Staging serves as an abstraction layer, simplifying and facilitating the load operation to the final tables in the data warehouse.



Also, staging area provides a buffer that can help to minimize the impact of the load operation on the performance of the data warehouse. This is important in environments where the data warehouse needs to remain operational and responsive during the data loading process.

Review type of data loads

There are two types of data loads to consider when loading a data warehouse.

Load Type	Description	Operation	Duration	Complexity	Best used
Full (initial) load	The process of populating the data warehouse for the first time.	All the tables are truncated and reloaded, and the old data is lost	It may take longer to complete due to the amount of data being handled	Easier to implement as there's no history preserved	This method is typically used when setting up a new data warehouse, or when a complete refresh of the data is required
Incremental load	The process of updating the data warehouse with the changes since the last update	The history is preserved, and tables are updated with new information	Takes less time than the initial load	Implementation is more complex than the initial load	This method is commonly used for regular updates to the data warehouse, such as daily or hourly updates. It requires mechanisms to track changes in the source data since the last load.

An ETL (Extract, Transform, Load) process for a data warehouse doesn't always need both the full load and the incremental load. In some cases, a combination of both methods might be used. The choice between a full load and an incremental load depends on many factors such as the amount of data, the characteristics of the data, and the requirements of the data warehouse.

To learn more about how to perform an incremental load, see [Incremental load](#).

Understand business key and surrogate key

In a data warehouse, both surrogate keys and business keys are essential for effective data warehousing and data integration, but they serve different purposes.

- **Surrogate key:** A surrogate key is a system-generated identifier that is used to uniquely identify a record in a table within the data warehouse. It has no business meaning and is typically an integer or a unique identifier. Surrogate keys are used to maintain consistency and accuracy in the data warehouse, especially when integrating data from multiple sources. They help to avoid issues that can arise from changes in the source systems, such as reusing or changing business keys.
- **Business Key:** A business key, also known as a natural key, is an identifier that comes from the source system and has business meaning. It's used to uniquely identify a record in the source system. Examples of business keys include product codes, customer IDs, and employee numbers. Business keys are important for maintaining traceability between the data warehouse and the source systems. They help to ensure that data in the warehouse can be accurately matched to the corresponding records in the source systems.

Load a dimension table

Think of a dimension table as the "*who, what, where, when, why*" of your data warehouse. It's like the descriptive backdrop that gives context to the raw numbers found in the fact tables.

For example, if you're running an online store, your fact table might contain the raw sales data - how many units of each product were sold. But without a dimension table, you wouldn't know who bought those products, when they were bought, or where the buyer is located.

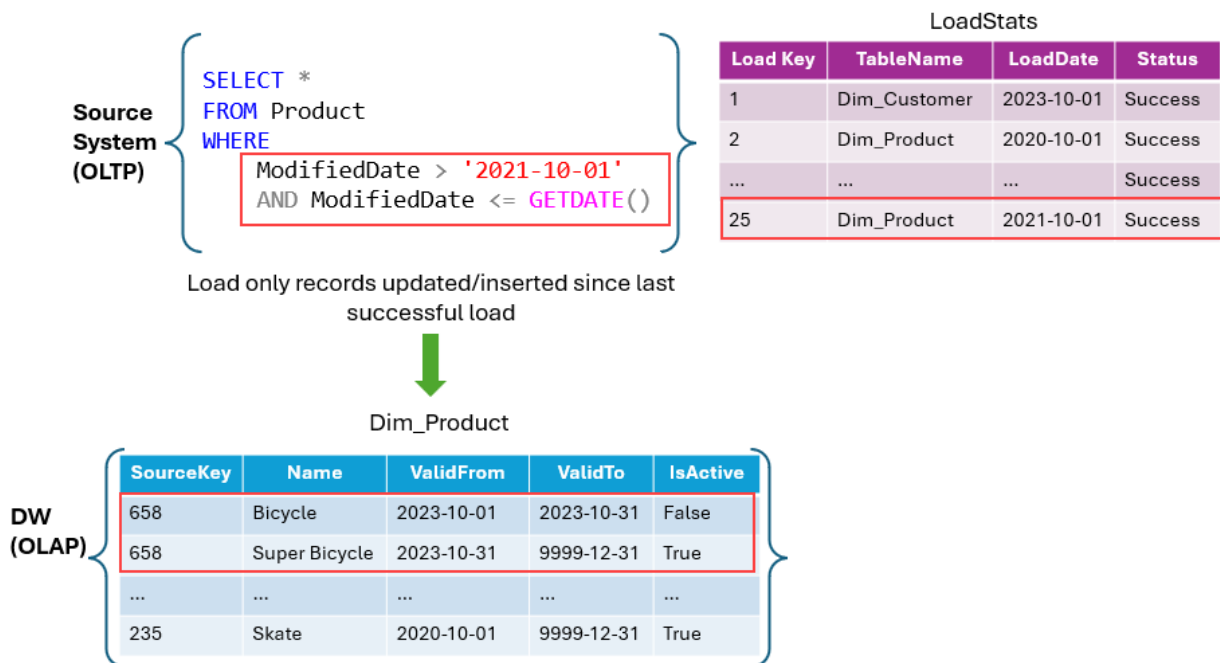
Slowly changing dimensions (SCD)

Slowly Changing Dimensions evolve over time, but at a slow pace and unpredictably. Take, for instance, a customer's address in a retail business. When a customer relocates, their address changes. If you overwrite the old address with the new one, you lose the historical data. However, if you need to analyze historical sales data, it's crucial to know where the customer lived at the time of each sale. This is where SCDs become essential.

There are several types of slowly changing dimensions in a data warehouse, with type 1 and type 2 being the most frequently used.

- **Type 0 SCD:** The dimension attributes never change.
- **Type 1 SCD:** Overwrites existing data, doesn't keep history.
- **Type 2 SCD:** Adds new records for changes, keeps full history for a given natural key.
- **Type 3 SCD:** History is added as a new column.
- **Type 4 SCD:** A new dimension is added.
- **Type 5 SCD:** When certain attributes of a large dimension change over time, but using type 2 isn't feasible due to the dimension's large size.
- **Type 6 SCD:** Combination of type 2 and type 3.

In type 2 SCD, when a new version of the same element is brought to the data warehouse, the old version is considered expired and the new one becomes active.



The following code shows a simple example on how to handle the business key in a type 2 SCD for the `Dim_Products` table using T-SQL.

```
IF EXISTS (SELECT 1 FROM Dim_Products WHERE SourceKey = @ProductID AND IsActive = 1)
BEGIN
    -- Existing product record
    UPDATE Dim_Products
    SET ValidTo = GETDATE(), IsActive = 0
    WHERE SourceKey = @ProductID
    AND IsActive = 1;
END
ELSE
BEGIN
    -- New product record
    INSERT INTO Dim_Products (SourceKey, ProductName, StartDate, EndDate, IsActive)
    VALUES (@ProductID, @ProductName, GETDATE(), '9999-12-31', 1);
END
```

The mechanism for detecting changes in source systems is crucial for determining when records are inserted, updated, or deleted. [Change Data Capture \(CDC\)](#), [change tracking](#), and [triggers](#) are all features available for managing data tracking in source systems such as SQL Server.

Load a fact table

Typically, a standard data warehouse load operation involves loading fact tables after dimension tables. This ensures that the dimensions, which the facts reference, are already present in the data warehouse.

The staged fact data usually includes business keys for the related dimensions, so your loading logic must look up the corresponding surrogate keys. When dealing with slowly changing dimensions in the data warehouse, it's crucial to identify the appropriate version of the dimension record to ensure the correct surrogate key is used. This matches the event recorded in the fact table with the state of the dimension at the time the fact occurred.

In many cases, you can retrieve the latest "current" version of the dimension. However, sometimes you might need to find the correct dimension record based on DateTime columns that indicate the period of validity for each version of the dimension.

The following example assumes that dimension records have incrementing surrogate keys and that the most recently added version of a specific dimension instance, which will have the highest key value, might be used.

```
-- Lookup keys in dimension tables
INSERT INTO dbo.FactSales
SELECT  (SELECT MAX(DateKey)
        FROM dbo.DimDate
        WHERE FullDateAlternateKey = stg.OrderDate) AS OrderDateKey,
        (SELECT MAX(CustomerKey)
        FROM dbo.DimCustomer
        WHERE CustomerAlternateKey = stg.CustNo) AS CustomerKey,
        (SELECT MAX(ProductKey)
        FROM dbo.DimProduct
        WHERE ProductAlternateKey = stg.ProductID) AS ProductKey,
        (SELECT MAX(StoreKey)
        FROM dbo.DimStore
        WHERE StoreAlternateKey = stg.StoreID) AS StoreKey,
        OrderNumber,
        OrderLineItem,
        OrderQuantity,
        UnitPrice,
        Discount,
        Tax,
        SalesAmount
FROM dbo.StageSales AS stg
```

3. Use data pipelines to load a warehouse

<https://learn.microsoft.com/en-us/training/modules/load-data-into-microsoft-fabric-data-warehouse/3-load-data-using-data-pipeline>

Use data pipelines to load a warehouse

Completed

Microsoft Fabric's Warehouse provides integrated data ingestion tools, enabling users to load and ingest data into warehouses on a large scale through either coding or noncoding experiences.

Data pipeline is the cloud-based service for data integration, which enables the creation of workflows for data movement and data transformation at scale. You can create and schedule data pipelines that can ingest and load data from disparate data stores. You can build complex ETL, or ELT processes that transform data visually with data flows.

Most of the functionality of data pipelines in Microsoft Fabric comes from Azure Data Factory, allowing for seamless integration and utilization of its features within the Microsoft Fabric ecosystem.

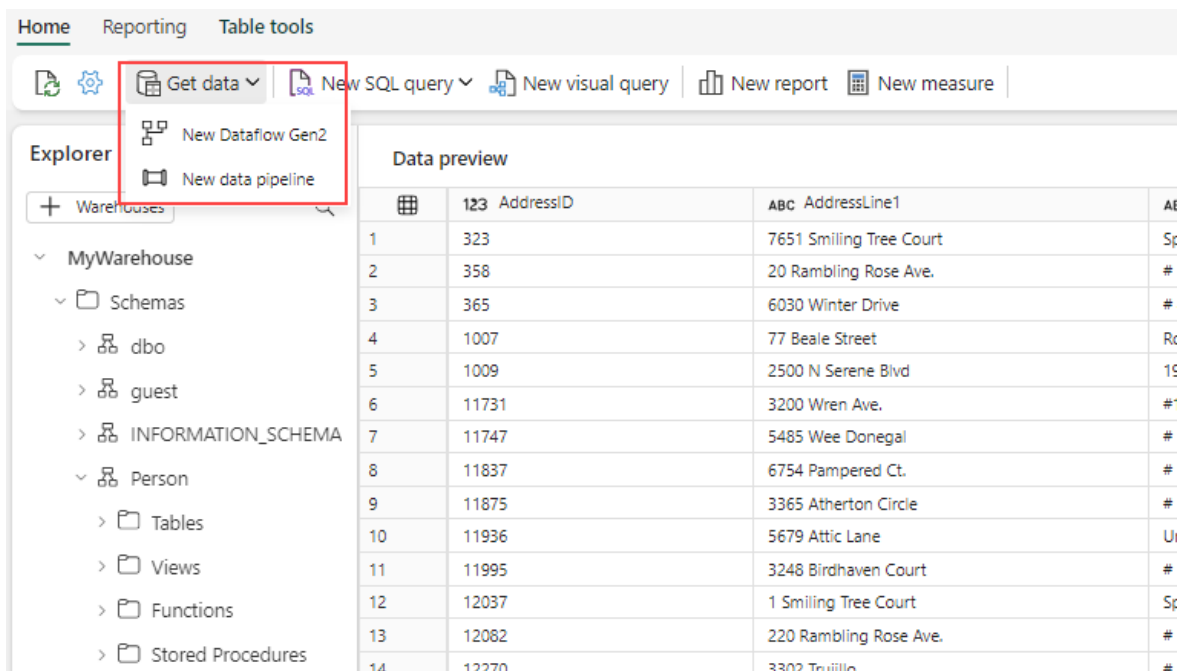
Note

All data in a Warehouse is automatically stored in the Delta Parquet format in OneLake.

Create a data pipeline

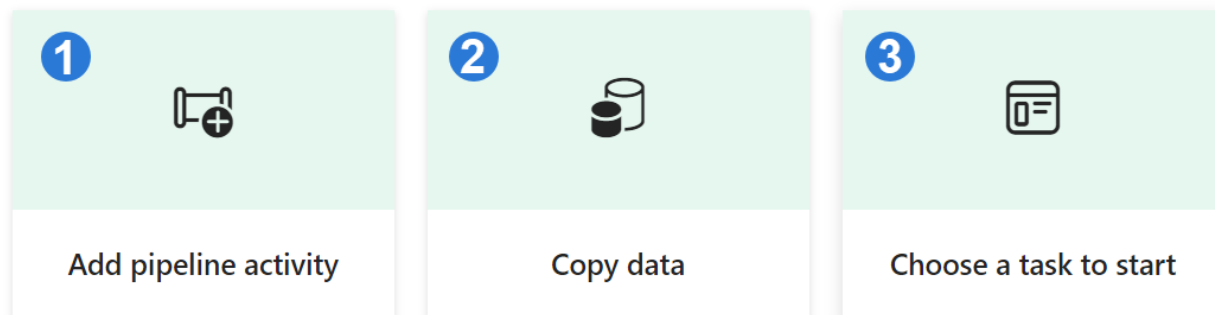
There are a few ways to launch the data pipeline editor.

- **From the workspace:** Select **+ New**, then select **Data pipeline**. If it's not visible in the list, select **More options**, then find **Data pipeline** under the **Get data** section.
- **From the warehouse asset** - Select **Get Data**, and then **New data pipeline**.



There are three options available when creating a pipeline.

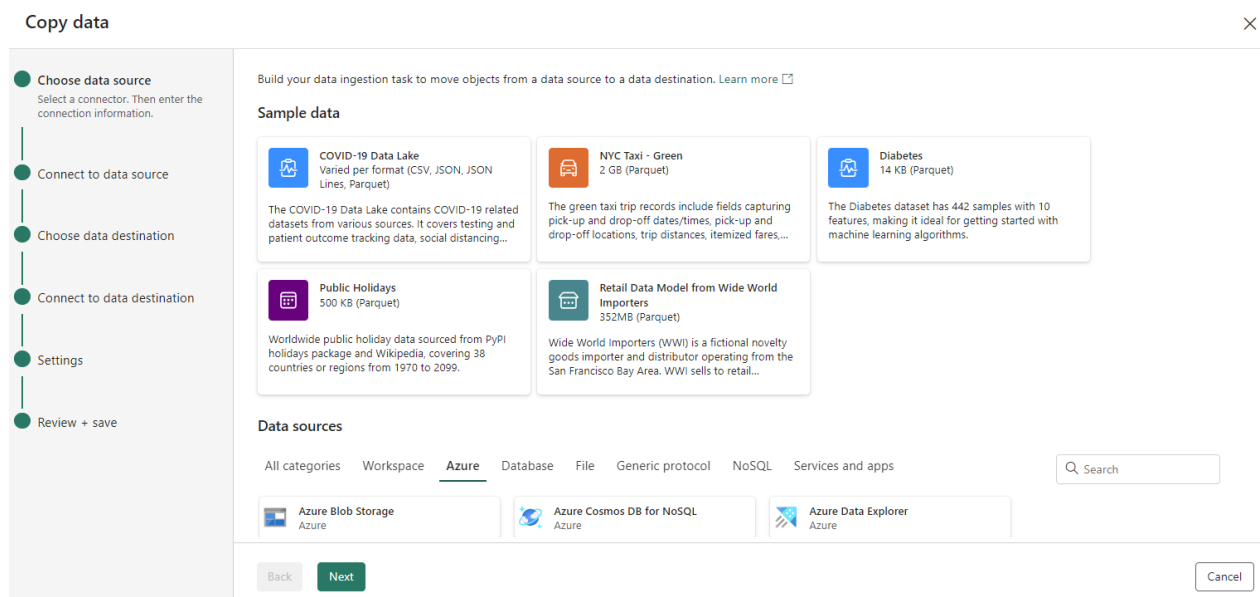
Start building your data pipeline



Option	Description
1. Add pipeline activity	Launches the pipeline editor where you can create your own pipeline.
2. Copy data	Launches an assistant to copy data from various data sources to a data destination. A new pipeline activity is generated at the end with a preconfigured Copy Data task.
3. Choose a task to start	You can choose from a collection of predefined templates to assist you in initiating pipelines based on many scenarios.

Configure the copy data assistant

The copy data assistant provides a step-by-step interface that facilitates the configuration of a **Copy Data** task.

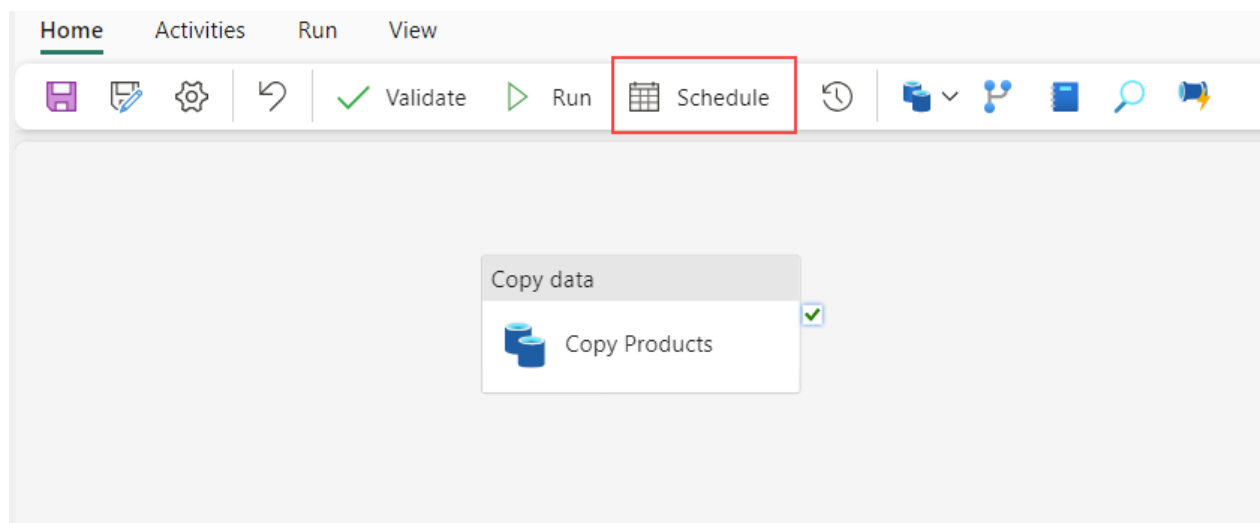


- **Choose data source:** Select a connector, and provide the connection information.
- **Connect to a data source:** Select, preview, and choose the data. This can be done from tables or views, or you can customize your selection by providing your own query.
- **Choose data destination:** Select the data store as the destination.
- **Connect to data destination:** Select and map columns from source to destination. You can load to a new or existing table.
- **Settings:** Configure other settings like staging, and default values.

After you copy the data, you can use other tasks to further transform and analyze it. You can also use the **Copy Data** task to publish transformation and analysis results for business intelligence (BI) and application consumption.

Schedule a data pipeline

You can schedule your data pipeline by selecting **Schedule** from the data pipeline editor.



You can also configure the schedule by selecting **Settings** in the **Home** menu in the data pipeline editor.

 Schedule

Scheduled run
☒ On ☐ Off

Repeat

Weekly 

Every

☐ Su ☒ Mo ☐ Tu ☒ We ☐ Th ☒ Fr ☐ Sa

Time

Enter time 



 Add a time

Start date and time

Enter start date 

End date and time

Enter end date 

Time zone

(UTC-06:00) Central Time (US and Canada) 

Apply

Discard

We recommend data pipelines for a code-free or low-code experience due to the graphical user interface. They're ideal for data workflows that run at a schedule, or that connects to different data sources.

To learn more about data pipelines, see [Ingest data into your Warehouse using data pipelines](#).

4. Load data using T-SQL

Load data using T-SQL

Completed

SQL developers or citizen developers, who are often well-versed in the SQL engine and adept at using T-SQL, will find the Warehouse in Microsoft Fabric favorable.

This is because the Warehouse is powered by the same SQL engine they're familiar with, enabling them to perform complex queries and data manipulations. These operations include filtering, sorting, aggregating, and joining data from different tables. The SQL engine's wide range of functions and operators further allows for sophisticated data analysis and transformations at the database level.

Use COPY statement

The [COPY statement](#) serves as the main method for importing data into the Warehouse. It facilitates efficient data ingestion from an external Azure storage account.

It offers flexibility, allowing you to specify the format of the source file, designate a location for storing rows that are rejected during the import process, skip header rows, among other configurable options.

The option to store rejected rows separately is useful for data cleaning and quality control. It allows you to easily identify and investigate any issues with the data that weren't successfully imported.

To connect to an Azure storage account, you need to use either Shared Access Signature (SAS) or Storage Account Key (SAK).

Handle error

The option to use a different storage account for the *ERRORFILE* location (`REJECTED_ROW_LOCATION`) allows for better error handling and debugging. It makes it easier to isolate and investigate any issues that occur during the data loading process. *ERRORFILE* only applies to CSV.

Load multiple files

The ability to specify wildcards and multiple files in the storage location path allows the COPY statement to handle bulk data loading efficiently. This is useful when dealing with large datasets distributed across multiple files.

Multiple file locations can only be specified from the same storage account and container via a comma-separated list.

```

COPY INTO my_table
FROM 'https://myaccount.blob.core.windows.net/myblobcontainer/folder0/*.csv,
    https://myaccount.blob.core.windows.net/myblobcontainer/folder1/'
WITH (
    FILE_TYPE = 'CSV',
    CREDENTIAL=(IDENTITY= 'Shared Access Signature', SECRET='<Your_SAS-Token>')
    FIELDTERMINATOR = '|'
)

```

The following example shows how to load a PARQUET file.

```

COPY INTO test_parquet
FROM 'https://myaccount.blob.core.windows.net/myblobcontainer/folder1/*.parquet'
WITH (
    CREDENTIAL=(IDENTITY= 'Shared Access Signature', SECRET='<Your_SAS-Token>')
)

```

Ensure that all the files have the same structure (that is, same columns in the same order) and that this structure matches the structure of the target table.

Load table from other warehouses and lakehouses

You can load data from various data assets in a workspace, such as other warehouses and lakehouses.

To reference the data asset, ensure that you use [three-part naming](#) to combine data from tables on these workspace assets. You can then use `CREATE TABLE AS SELECT` (CTAS) and `INSERT...SELECT` to load the data into the warehouse.

SQL Statement	Description
<code>CREATE TABLE AS SELECT</code>	Allows you to create a new table based on the output of a <code>SELECT</code> statement. This operation is often used for creating a copy of a table or for transforming and loading the results of complex queries.
<code>INSERT...SELECT</code>	Allows you to insert data from one table into another. It's useful when you want to copy data from one table to another without creating a new table.

In a scenario where an analyst needs data from both a warehouse and a lakehouse, they can use this feature to combine the data. They can then load this combined data into the warehouse for analysis. This feature is useful when data is distributed across many assets in a workspace.

The following query creates a new table in the `analysis_warehouse` that combines data from the `sales_warehouse` and the `social_lakehouse` using the `product_id` as the common key. The new table can then be used for further analysis.

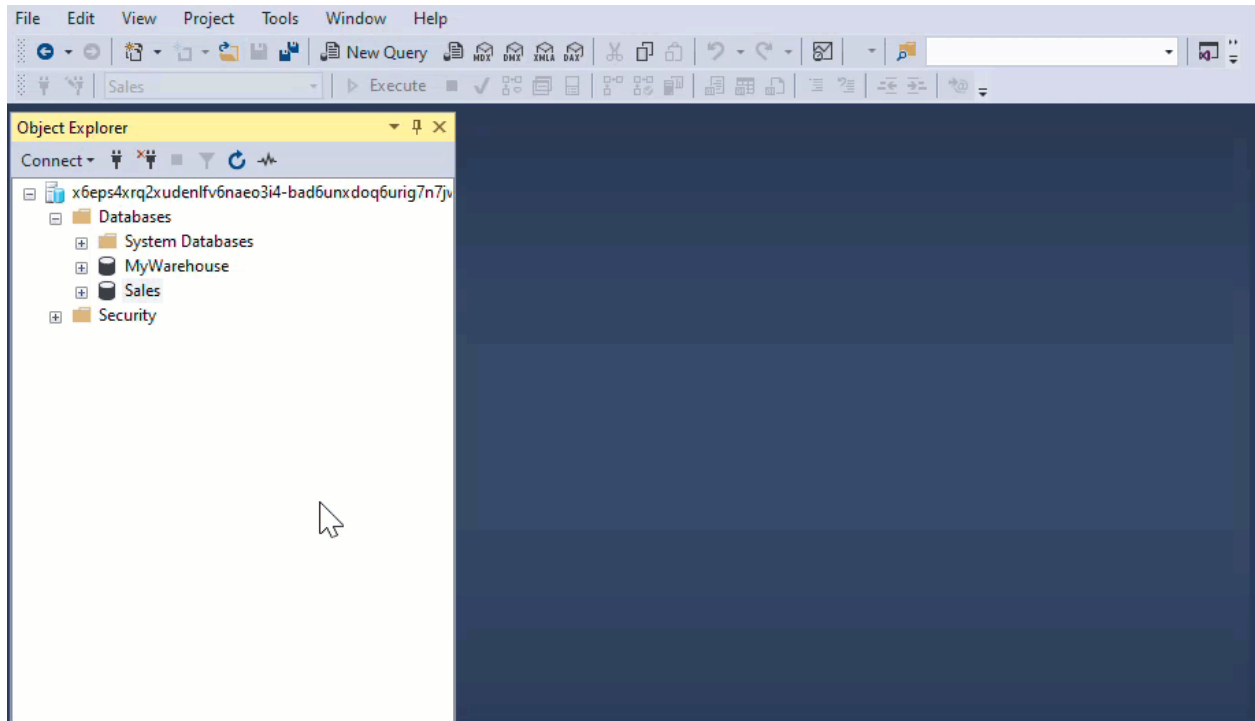
```

CREATE TABLE [analysis_warehouse].[dbo].[combined_data]
AS
SELECT *

```

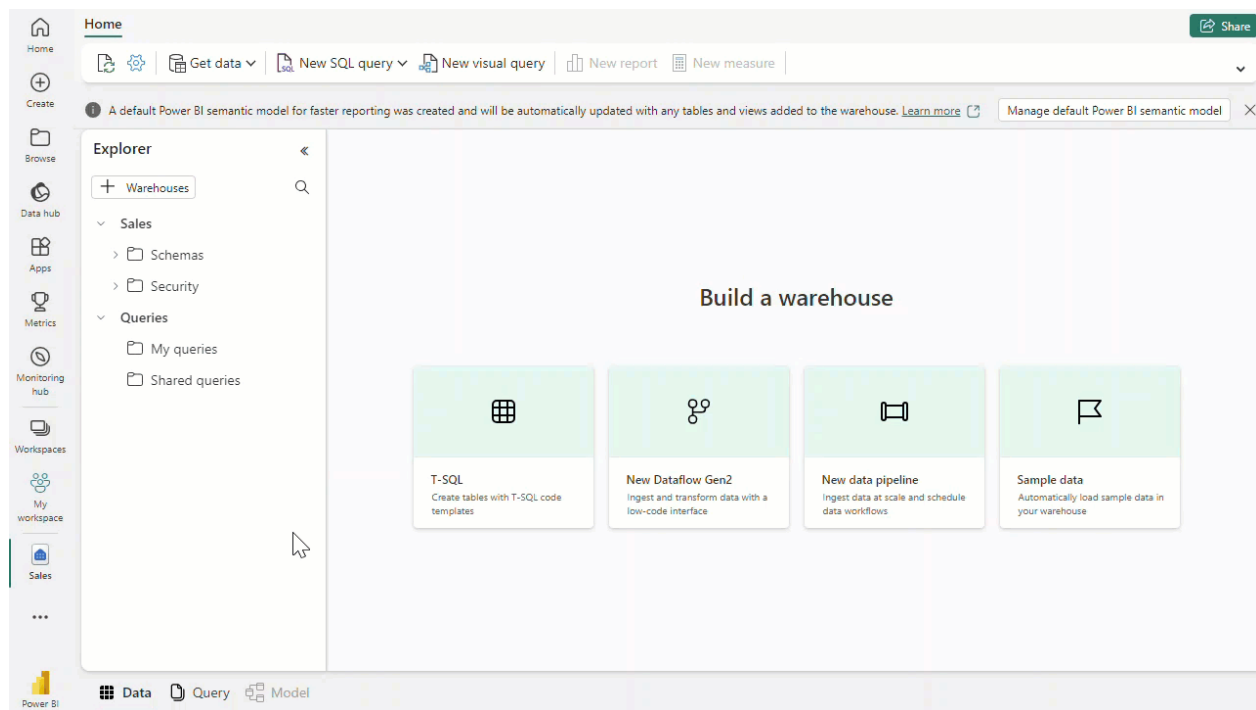
```
FROM [sales_warehouse].[dbo].[sales_data] sales
INNER JOIN [social_lakehouse].[dbo].[social_data] social
ON sales.[product_id] = social.[product_id];
```

All the Warehouses that share the same workspace are integrated into the same logical SQL server. If you use SQL client tools such as [SQL Server Management Studio](#), you can easily perform a cross-database query like in any SQL Server instance.



MyWarehouse and *Sales* are both warehouse assets that share the same workspace.

If you're using the object Explorer from the workspace to query your Warehouses, you need to add them explicitly. The warehouses added will also be visible from the Visual query editor.



Data can be efficiently loaded into a warehouse in Microsoft Fabric through the COPY statement, or from other warehouses and lakehouses within the same workspace, allowing for seamless data management and analysis.

5. Load and transform data with Dataflow Gen2

<https://learn.microsoft.com/en-us/training/modules/load-data-into-microsoft-fabric-data-warehouse/5-load-data-using-dataflow>

Load and transform data with Dataflow Gen2

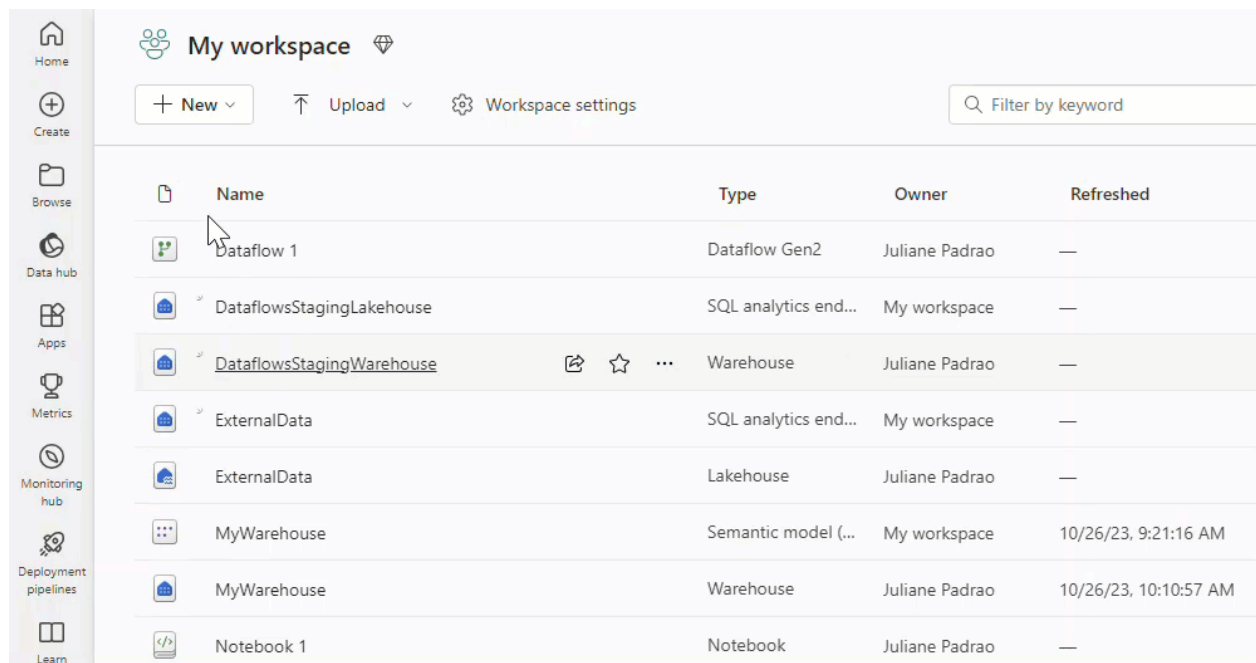
Completed

Dataflow Gen2 is the new generation of dataflows. It provides a comprehensive Power Query experience, guiding you through each step of importing data into your dataflow. The process of creating dataflows has been simplified, reducing the number of steps involved.

You can use dataflows in data pipelines to ingest data into a lakehouse or warehouse, or to define a dataset for a Power BI report.

Create a dataflow

To create a new dataflow, navigate to your workspace, then select **+ New**. If **Dataflow Gen2** isn't visible in the list, select **More options**, then find **Dataflow Gen2** under the **Data Factory** section.

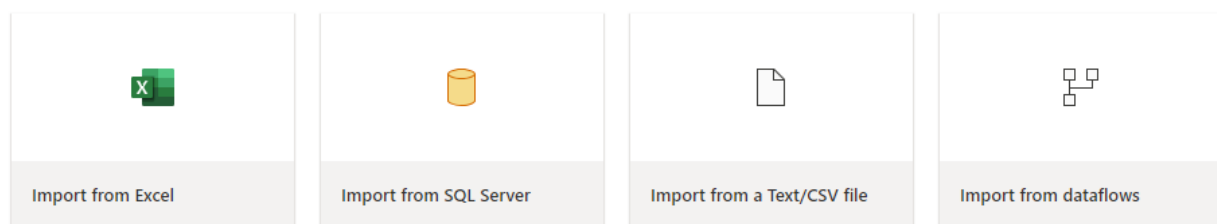


The screenshot shows the 'My workspace' interface with a sidebar on the left containing navigation icons for Home, Create, Browse, Data hub, Apps, Metrics, Monitoring hub, Deployment pipelines, and Learn. The main area displays a table of data assets with columns: Name, Type, Owner, and Refreshed. A search bar at the top right allows filtering by keyword. The table lists several assets, including 'Dataflow 1', 'DataflowsStagingLakehouse', 'DataflowsStagingWarehouse', 'ExternalData', 'MyWarehouse', and 'Notebook 1'.

Name	Type	Owner	Refreshed
Dataflow 1	Dataflow Gen2	Juliane Padrao	—
DataflowsStagingLakehouse	SQL analytics end...	My workspace	—
DataflowsStagingWarehouse	Warehouse	Juliane Padrao	—
ExternalData	SQL analytics end...	My workspace	—
ExternalData	Lakehouse	Juliane Padrao	—
MyWarehouse	Semantic model (...)	My workspace	10/26/23, 9:21:16 AM
MyWarehouse	Warehouse	Juliane Padrao	10/26/23, 10:10:57 AM
Notebook 1	Notebook	Juliane Padrao	—

Import data

Once the Dataflow Gen2 launches, there are many options to load your data available.



[Get data from another source →](#)

[Import from a Power Query template](#)

You can load different file types with just a few steps. For example, to load a text or CSV file from your local computer.

Get data

Connect to data source

Text/CSV

File

[Learn more](#)

Connection settings

☐ Link to file

☒ Upload file (Preview) ⓘ

customer-churn 1.csv

Upload successful (481.6 KB)

Connection credentials

Connection

https://microsoft-my.sharepoint.com/personal/jupadra... ▾

↻

Authentication kind: Organizational account [Edit connection](#)

Back

Cancel

Next

Once the data is imported you can start authoring your dataflow, you might decide to clean your data, reshape, remove columns, and create new ones. All the steps you perform are saved.

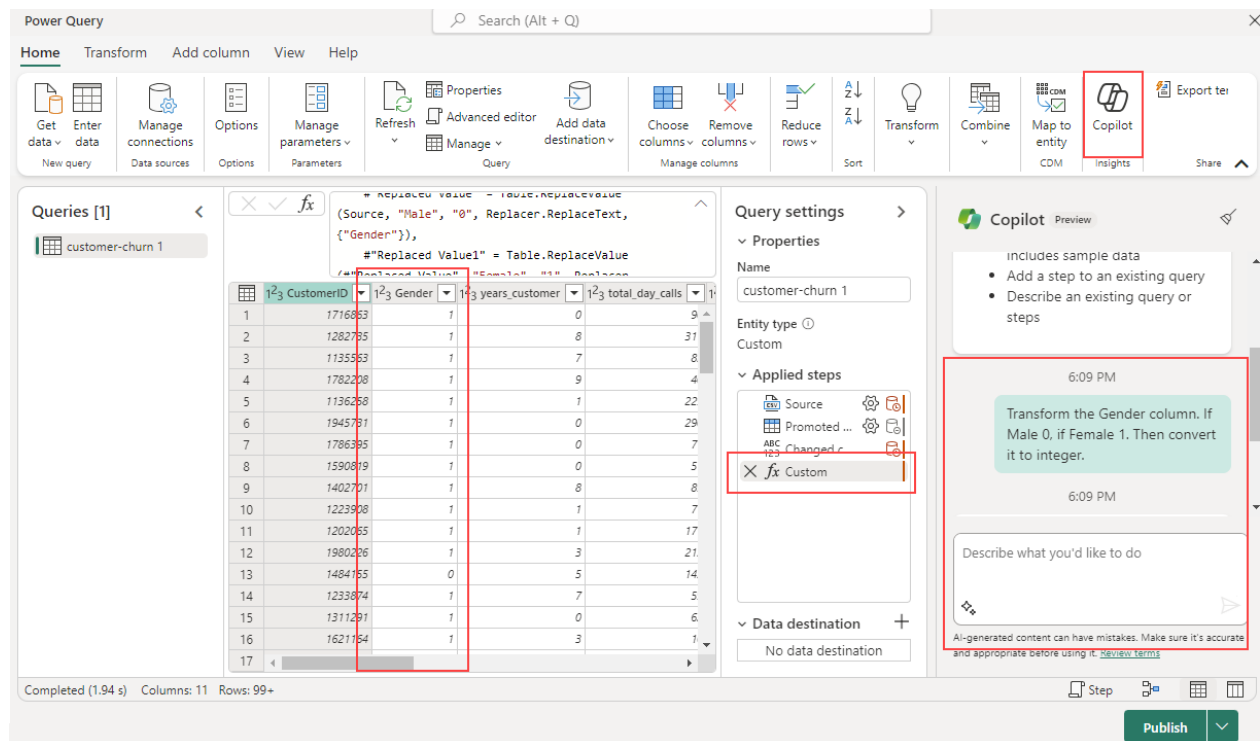
Transform data with Copilot

Copilot can be a valuable tool for assisting with dataflow transformations. Let's say we have a *Gender* column that contains 'Male' and 'Female' and we want to transform it.

The first step is to activate Copilot within your dataflow. Once that's done, you can then provide specific instructions on the transformation you want to perform.

For instance, you might input the following command: *"Transform the Gender column. If Male 0, if Female 1. Then convert it to integer."*

17 / 21



Copilot adds a new step automatically, and you can always revert it if you want, or continue to build on it for further transformations.

Add a data destination

With the **Add data destination** feature, you can separate your ETL logic and destination storage. This separation can lead to cleaner, more maintainable code and can make it easier to modify either the ETL process or the storage configuration without affecting the other.

Once the data is transformed, the next step is to add a destination step. On the **Query settings** tab, select **+** to add a destination step in your dataflow.

Query settings

▼ Properties

Name

customer-churn 1

Entity type ⓘ

Custom

▼ Applied steps

Source

Promoted headers

ABC
123 Changed column type

Custom

▼ Data destination

No data destination

The following destination options are available.

- Azure SQL Database
- Lakehouse
- Azure Data Explorer (Kusto)
- Azure Synapse Analytics (SQL DW)
- Warehouse

Data that's loaded into a destination like a warehouse can be easily accessed and analyzed using various tools. This improves the accessibility of your data and allows for more flexible and comprehensive data analysis.

When you select a warehouse as a destination, you can choose the following update methods.



- **Append:** Add new rows to an existing table.

- **Replace:** Replace the entire content of a table with a new set of data.

Publish a dataflow

After you choose your update method, the final step is to publish your dataflow.

Publishing makes your transformations and data loading operations live, allowing the dataflow to be executed either manually or on a schedule. This process encapsulates your ETL operations into a single and reusable unit, streamlining your data management workflow.

Any changes made in the dataflow take effect when it's published. So, always ensure to publish your dataflow after making any relevant modifications.

6. Exercise: Load data into a warehouse in Microsoft Fabric

<https://learn.microsoft.com/en-us/training/modules/load-data-into-microsoft-fabric-data-warehouse/6-exercise-load-data-into-warehouse>

Exercise: Load data into a warehouse in Microsoft Fabric

Completed

Now it's your chance to load data into a warehouse in Microsoft Fabric.

In this exercise, you'll learn how to load data into a warehouse using T-SQL.

Note

You need a Microsoft Fabric trial license with the Fabric preview enabled in your tenant. See [Getting started with Fabric](#) to enable your Fabric trial license.

Launch the exercise and follow the instructions.

[Launch Exercise](#)

7. Module assessment

<https://learn.microsoft.com/en-us/training/modules/load-data-into-microsoft-fabric-data-warehouse/7-knowledge-check>

Module assessment

Completed

8. Summary

<https://learn.microsoft.com/en-us/training/modules/load-data-into-microsoft-fabric-data-warehouse/8-summary>

Summary

Completed

There's no one-size-fits-all solution for loading your data. The best approach depends on the specifics of your business requirement and the question you're trying to answer.

When it comes to loading data in a data warehouse, there are several considerations to keep in mind.

	Description
Load volume & frequency	Assess data volume and load frequency to optimize performance.
Governance	Any data that lands in OneLake is governed by default.
Data mapping	Manage mapping from source to staging to warehouse.
Dependencies	Understand dependencies in the data model for loading dimensions.
Script design	Design efficient import scripts considering column names, filtering rules, value mapping, and database indexing.

For additional reading, you can refer to the following URLs:

- [Create a Warehouse in Microsoft Fabric](#)
- [Ingest data into the Warehouse](#)
- [Compare the Warehouse and the SQL analytics endpoint of the Lakehouse](#)