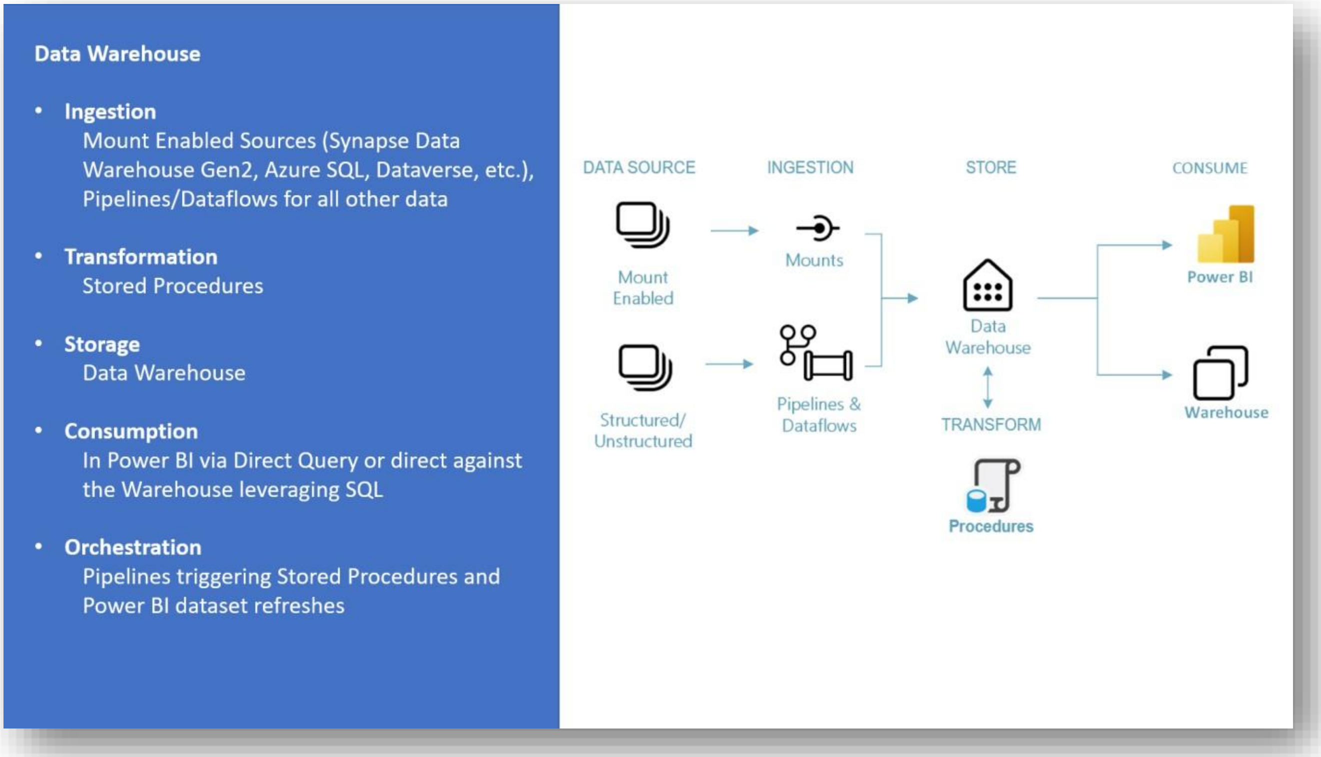
Tutorial

Data Warehouse

Published: July
2024



Data Warehouse

- **Ingestion**
  Mount Enabled Sources (Synapse Data Warehouse Gen2, Azure SQL, Dataverse, etc.), Pipelines/Dataflows for all other data
- **Transformation**
  Stored Procedures
- **Storage**
  Data Warehouse
- **Consumption**
  In Power BI via Direct Query or direct against the Warehouse leveraging SQL
- **Orchestration**
  Pipelines triggering Stored Procedures and Power BI dataset refreshes

Contents

Introduction

What is Fabric?

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 Fabric, there is 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. There is no other product on the market that offers the breadth, depth, and level of integration that Fabric offers. Additionally, Microsoft Purview is included by default in every tenant to meet compliance and governance needs.

To get an overview over the components and concepts of Fabric read Fabric - Overview and Concepts.

Purpose of this tutorial

While many concepts in 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 Fabric UX, the various workloads and their integration points, and the Fabric professional and citizen developer experiences.

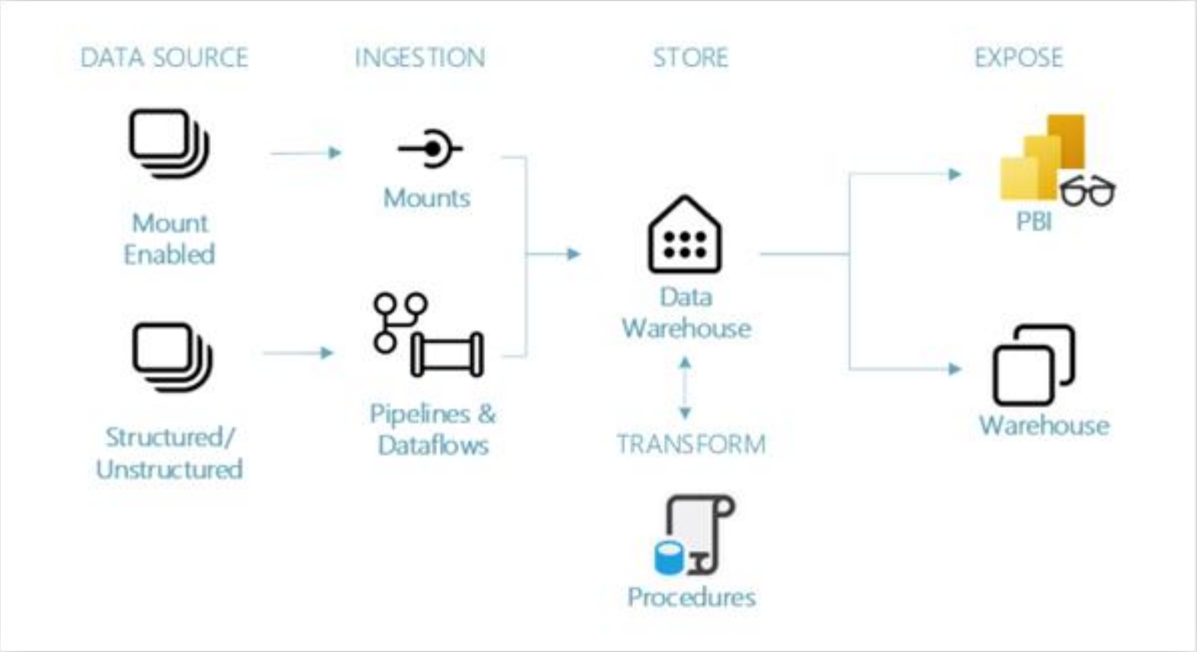
The tutorials are not intended to be a reference architecture, an exhaustive list of features and functionality, or a recommendation of specific best practices.

The data warehouse tutorial

In this tutorial, you will take on the role of a data warehouse developer at the fictional Wide World Importers company and complete the following steps:

- Sign into your Power BI online account, or if you don't have an account yet, sign up for a free trial.
- Build and implement an end to end data warehouse for your organization:
  - o Enable Fabric in your tenant
  - o Create a Fabric workspace
  - o Quickly create a data warehouse, Ingest data from source to the data warehouse dimensional model
  - o Transform the data to create aggregated datasets using T-SQL
  - o Perform orchestration, data ingestion, and data transformation with pipelines
  - o Query the data warehouse using T-SQL and a visual query editor
  - o Create Power BI report using DirectLake mode to analyze the data in place
- Cleanup resources by deleting the workspace and other items

The data warehouse end to end architecture



Data Sources – 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 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 Fabric that provides a way to connect to existing data without having to copy or move it – more details about Shortcut later in this tutorial.

Transform and Store – Fabric standardizes on Delta Lake format, that means all the engines of 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, 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 will be automatically generated at the same time with the same name of the data warehouse to start visualizing data with just a couple of mouse clicks.

The sample data

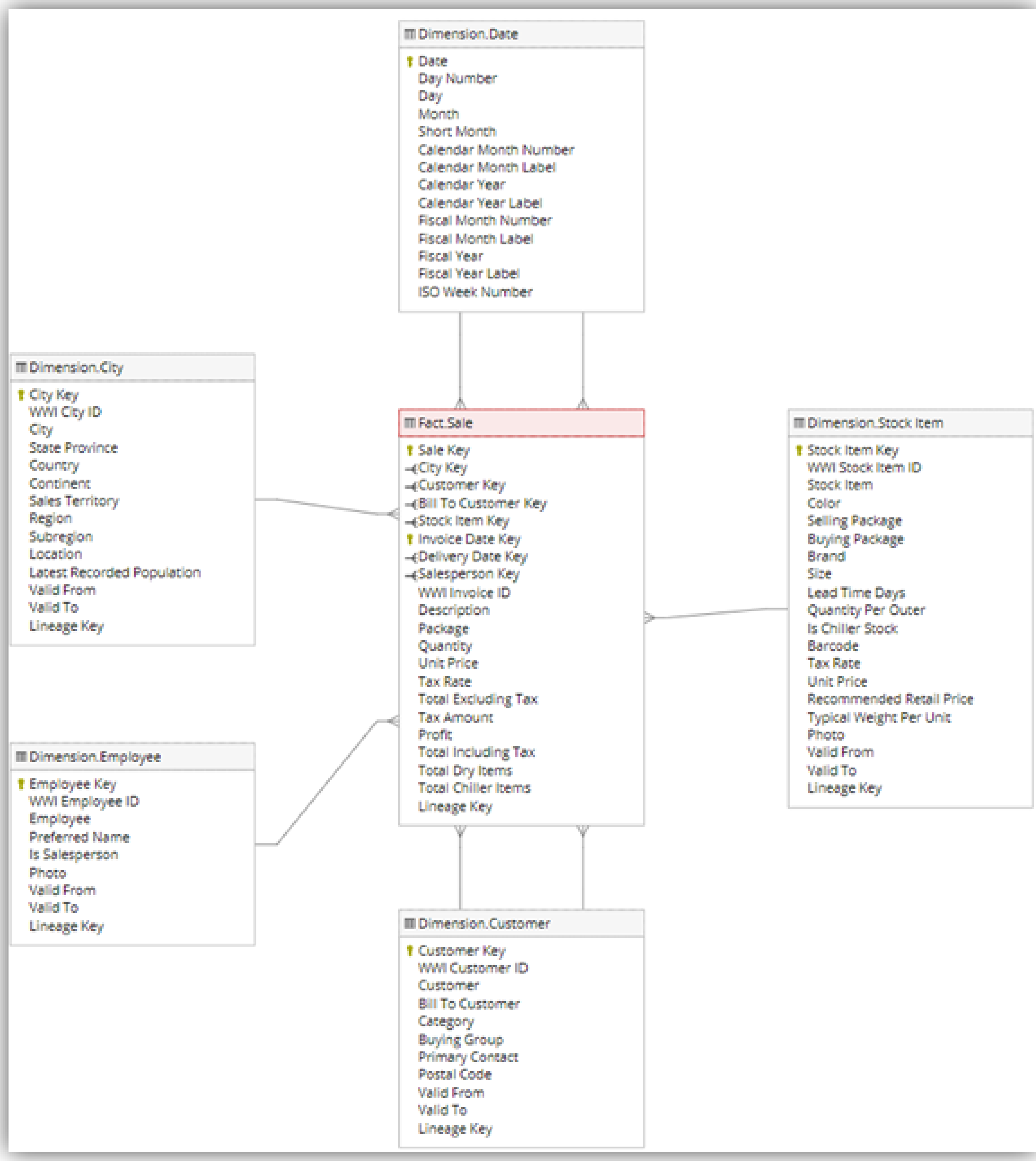
For sample data, we are going to use Wide World Importers (WWI) sample database. For our data warehouse end-to-end scenario, we have generated sufficient data for a sneak peek into the scale and performance capabilities of the Fabric platform.

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. You can learn more about their company profile and operation here.
Typically, you would bring data from transactional systems (or line of business applications) into a data lake or data warehouse staging area, however for simplicity of this tutorial, we are going to use the dimensional model provided by WWI as our initial data source. We are going to use it as the source to ingest the data into a data warehouse and transform it through T-SQL.

The data model

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



Module 1: Create a workspace (This step is not needed if the workspace is already created for every user upon provisioning of trial tenant)

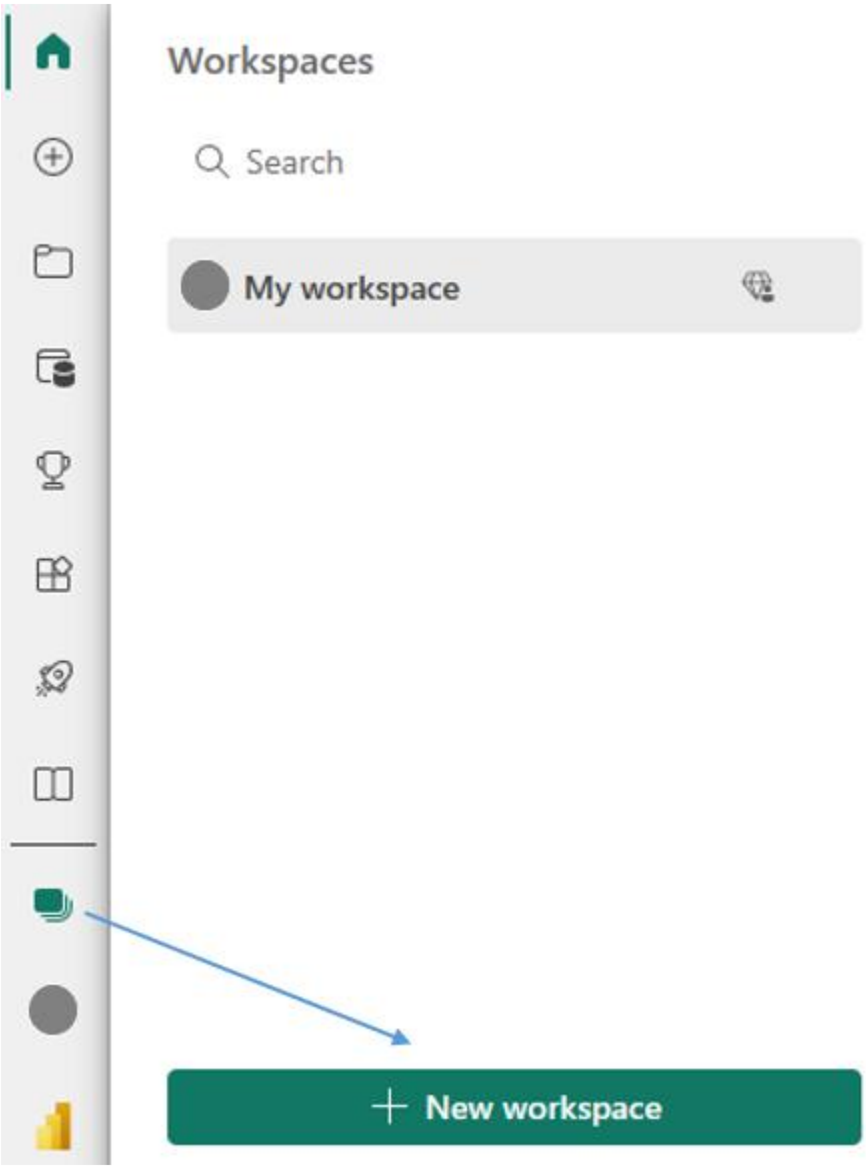
Please make sure to complete the Lakehouse tutorial first before starting this one as this tutorial use the lakehouse as one of the data sources for the exercise

Before you can begin building the warehouse, you will need to create a workspace where you will build out the remainder of the tutorial. In this module, you will learn to:

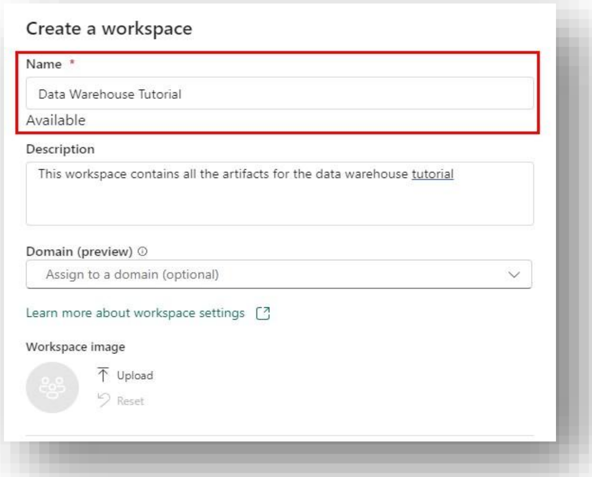
- Create a workspace

The workspace will contain all the artifacts 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.



Create a workspace

Name *

Data Warehouse Tutorial

Available

Description

This workspace contains all the artifacts for the data warehouse tutorial

Domain (preview)

Assign to a domain (optional)

4. Expand the **Advanced** section.
5. Choose **Premium capacity** in the **License Mode** section.
6. Choose a premium capacity you have access to**.**

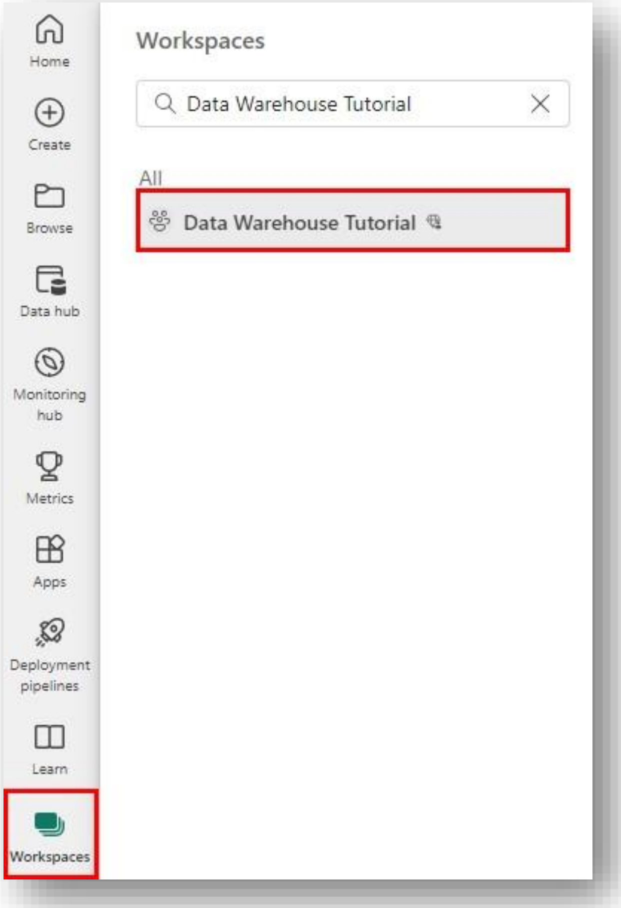
7. Select **Apply.** The workspace will be created and opened.

Module 2: Build your first data warehouse

The intent of this module is to quickly build end to end journey of building a data warehouse, ingesting data for a table and then using the data warehouse for creating a report.

Create a data warehouse

1. In the Power BI service select **Workspaces** in the left-hand menu.
2. Search for the workspace you create in Module 1 by typing in the search textbox at the top and click on your workspace to open it.



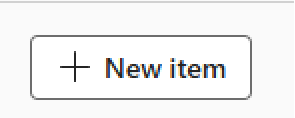
Workspaces

Data Warehouse Tutorial

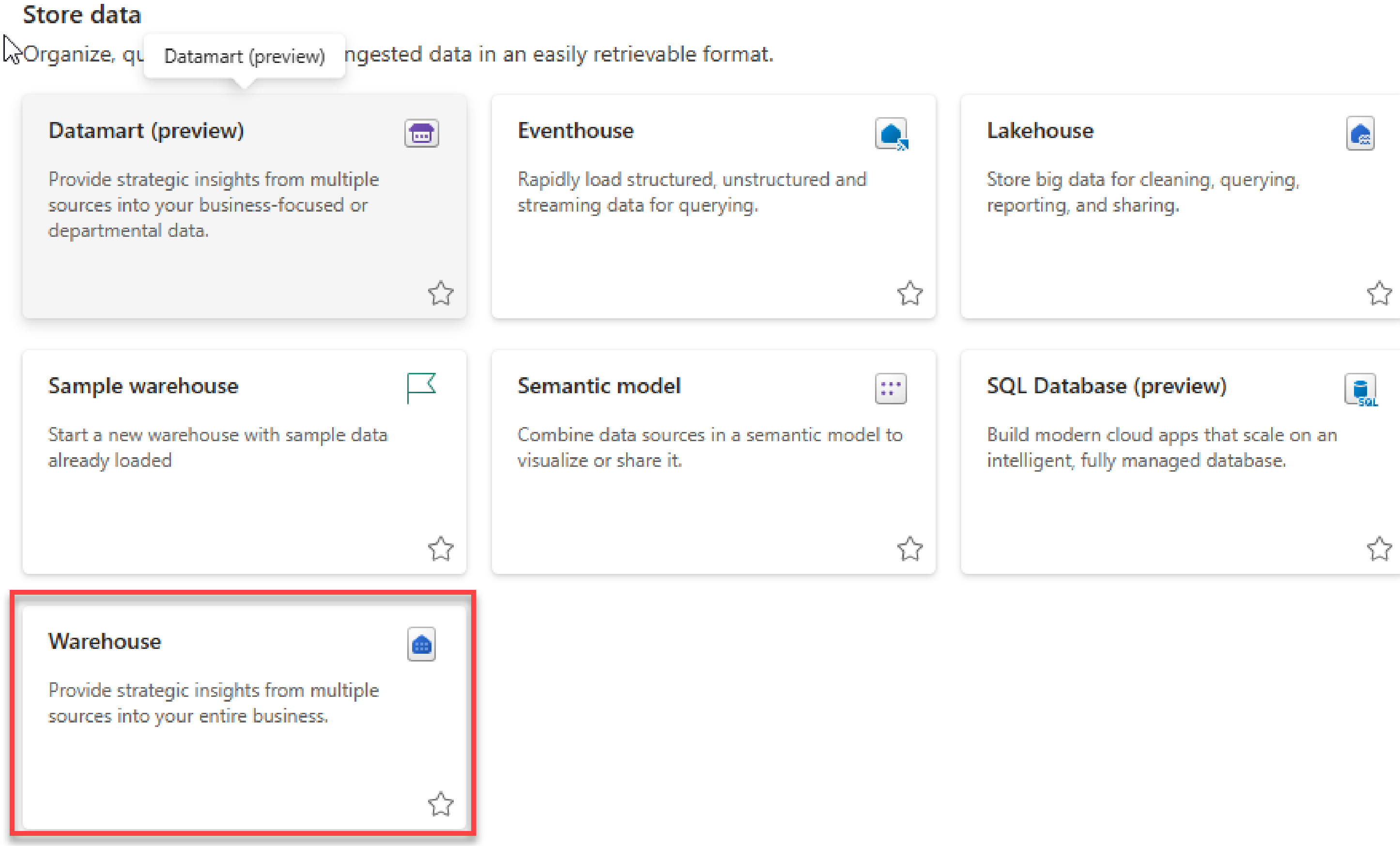
All

Data Warehouse Tutorial

3. In the upper left corner, select **New item** to display a full list of available items.



4. In the **Store Data** section, select **Warehouse**.



Store data

Organize, qu Datamart (preview) ngested data in an easily retrievable format.

Datamart (preview)
Provide strategic insights from multiple sources into your business-focused or departmental data.

Eventhouse
Rapidly load structured, unstructured and streaming data for querying.

Lakehouse
Store big data for cleaning, querying, reporting, and sharing.

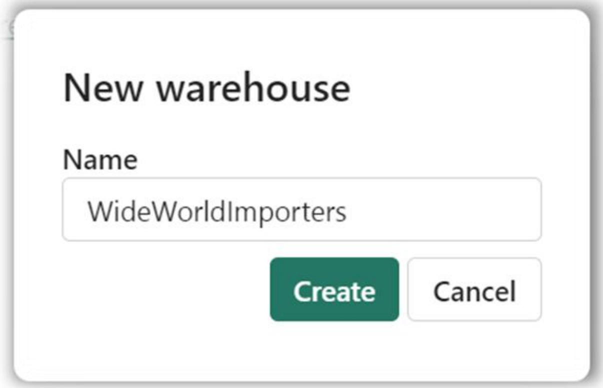
Sample warehouse
Start a new warehouse with sample data already loaded

Semantic model
Combine data sources in a semantic model to visualize or share it.

SQL Database (preview)
Build modern cloud apps that scale on an intelligent, fully managed database.

Warehouse
Provide strategic insights from multiple sources into your entire business.

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

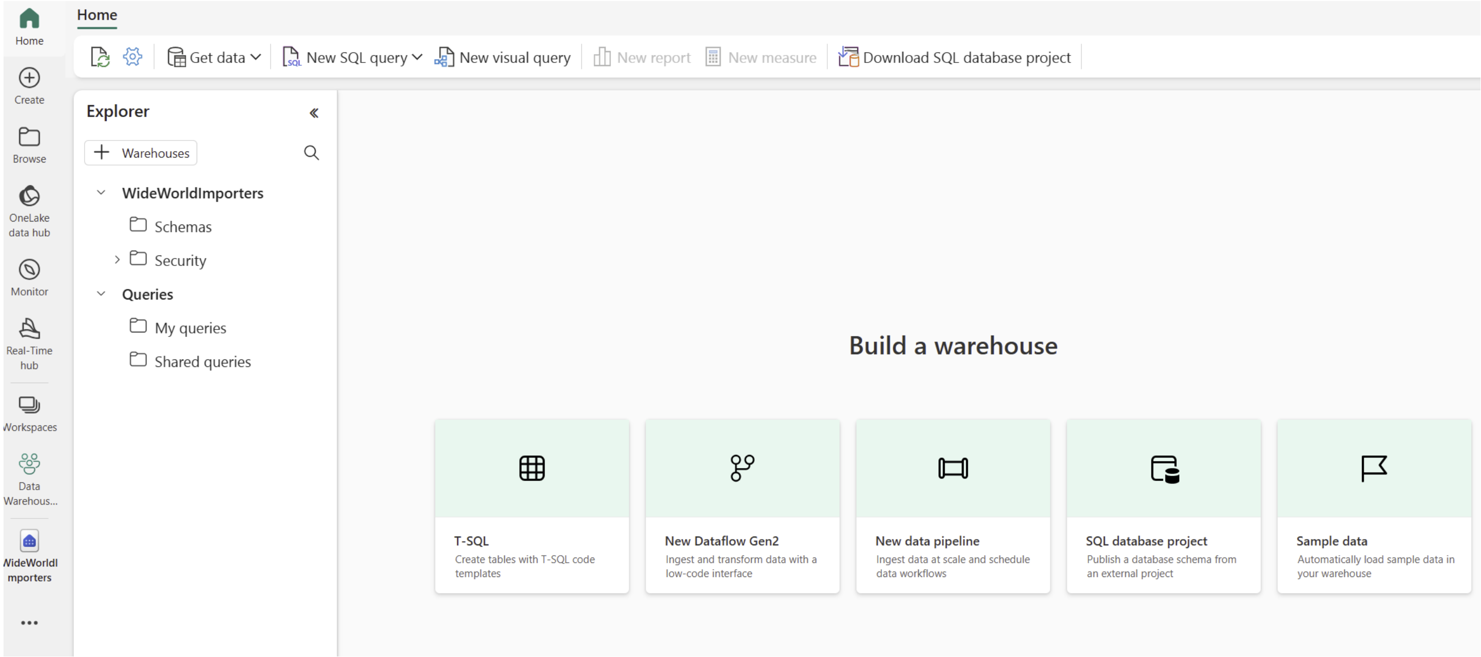


New warehouse

Name
WideWorldImporters

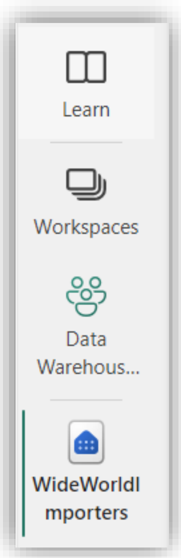
6. Select **Create**.

When provisioning is complete the **Build a warehouse** landing page will be shown.

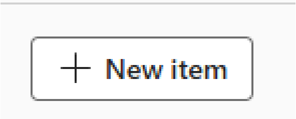


Data ingestion

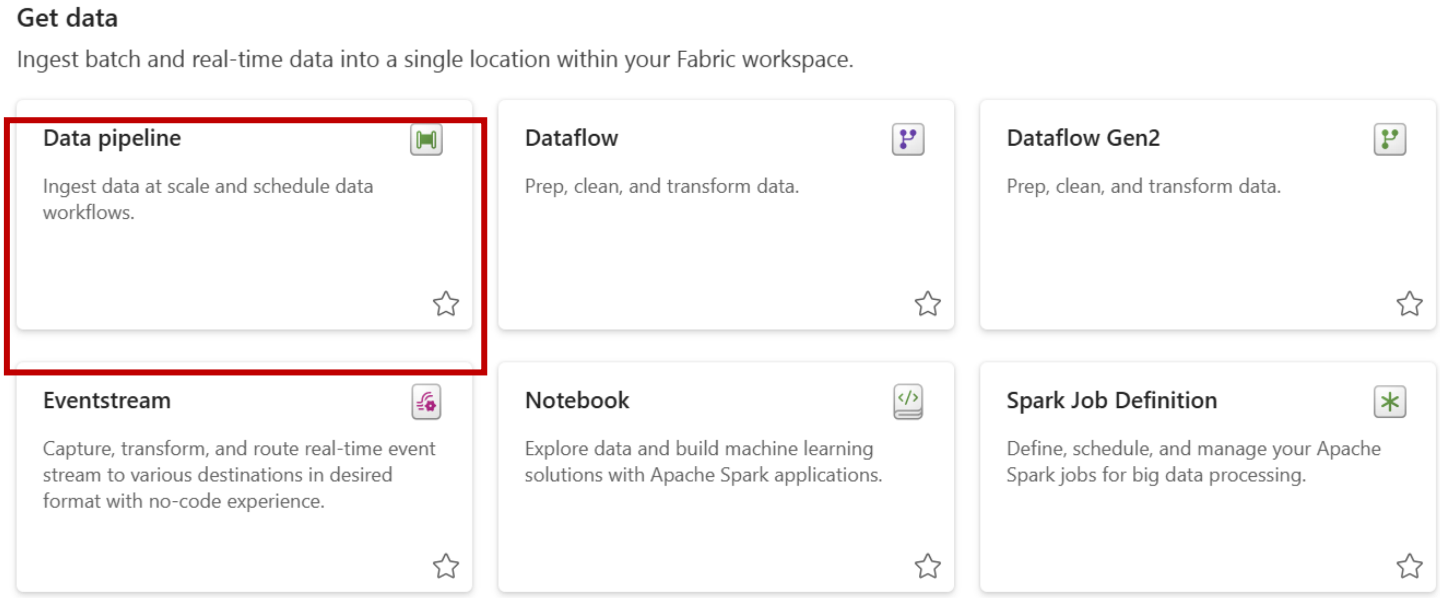
1. Select **Data Warehouse Tutorial** in the left-hand navigation menu to return to the workspace artifact view.



2. In the upper left corner, select **+ New Item** to display a full list of available items.



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



Get data

Ingest batch and real-time data into a single location within your Fabric workspace.

Data pipeline

Ingest data at scale and schedule data workflows.

Dataflow

Prep, clean, and transform data.

Dataflow Gen2

Prep, clean, and transform data.

Eventstream

Capture, transform, and route real-time event stream to various destinations in desired format with no-code experience.

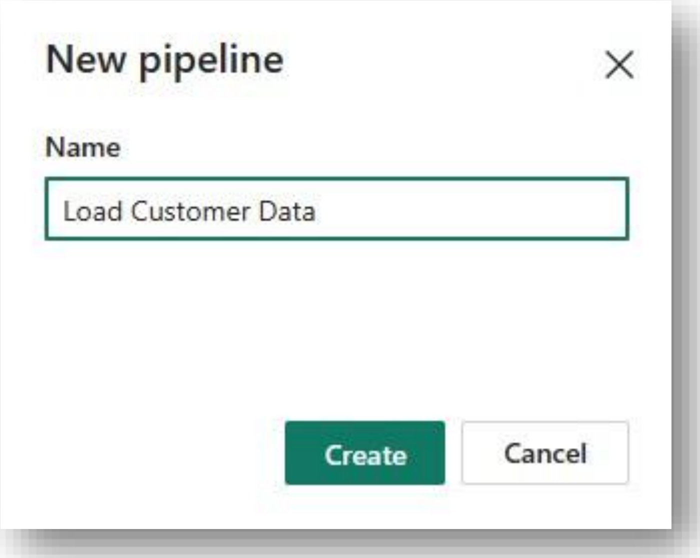
Notebook

Explore data and build machine learning solutions with Apache Spark applications.

Spark Job Definition

Define, schedule, and manage your Apache Spark jobs for big data processing.

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

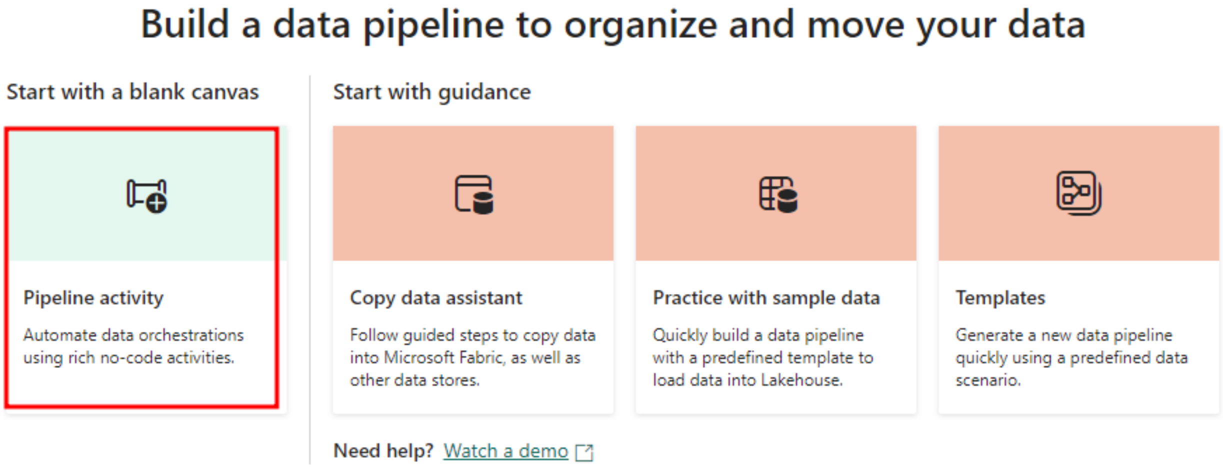


New pipeline

Name

Load Customer Data

5. Select **Create**.
6. Select P**ipeline activity** from the **Build a data pipeline to organize and move your data** landing page.



Build a data pipeline to organize and move your data

Start with a blank canvas

Pipeline activity
Automate data orchestrations using rich no-code activities.

Start with guidance

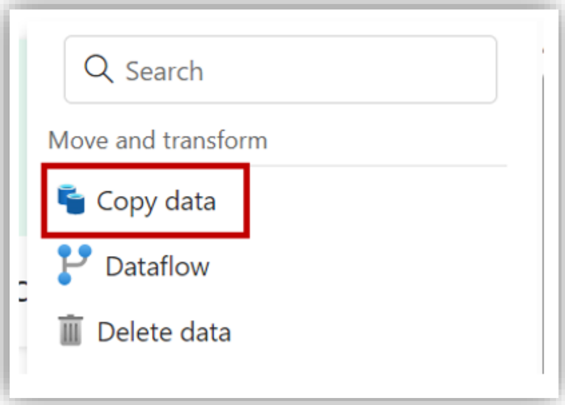
Copy data assistant
Follow guided steps to copy data into Microsoft Fabric, as well as other data stores.

Practice with sample data
Quickly build a data pipeline with a predefined template to load data into Lakehouse.

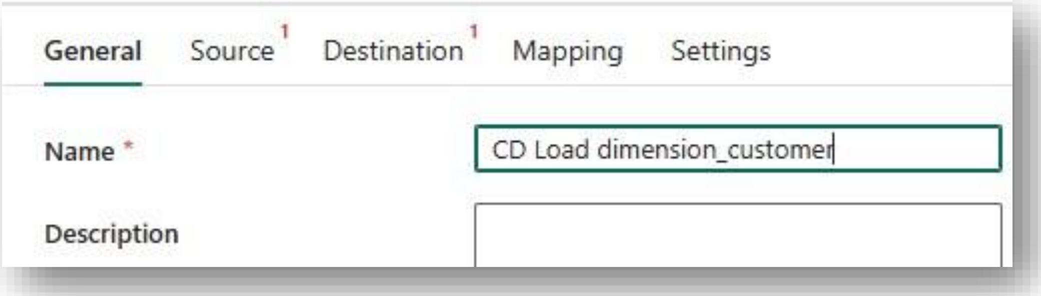
Templates
Generate a new data pipeline quickly using a predefined data scenario.

Need help? Watch a demo

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



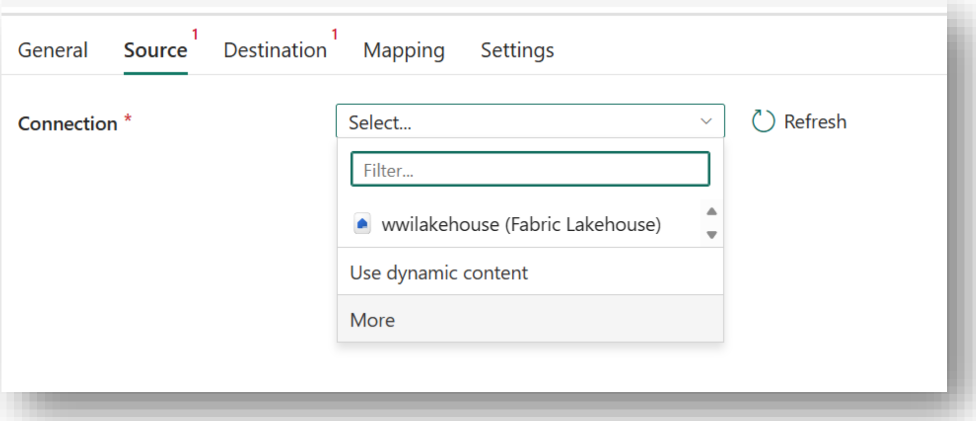
8. If necessary, select the newly created Copy data activity from the design canvas and follow the steps below to configure it.
9. On the **General** page, enter **CD Load dimension_customer** as the **Name**.



Name * CD Load dimension_customer

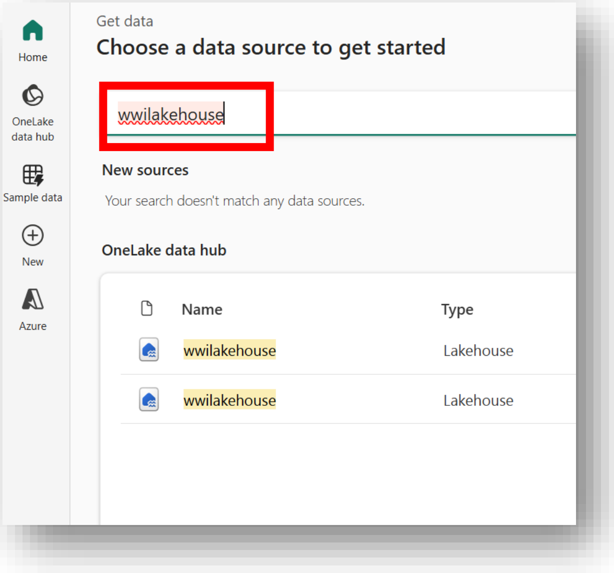
Description

10. On the **Source** page Under **Connection** , select **More**



Connection * Select...

11. Type **wwilakehouse** in the searchbox to **select the lakehouse created in the lakehouse tutorial**. Make sure you are the owner of the lakehouse and is the one you created in the prior tutorial as multiple names will show up



Get data

Choose a data source to get started

wwilakehouse

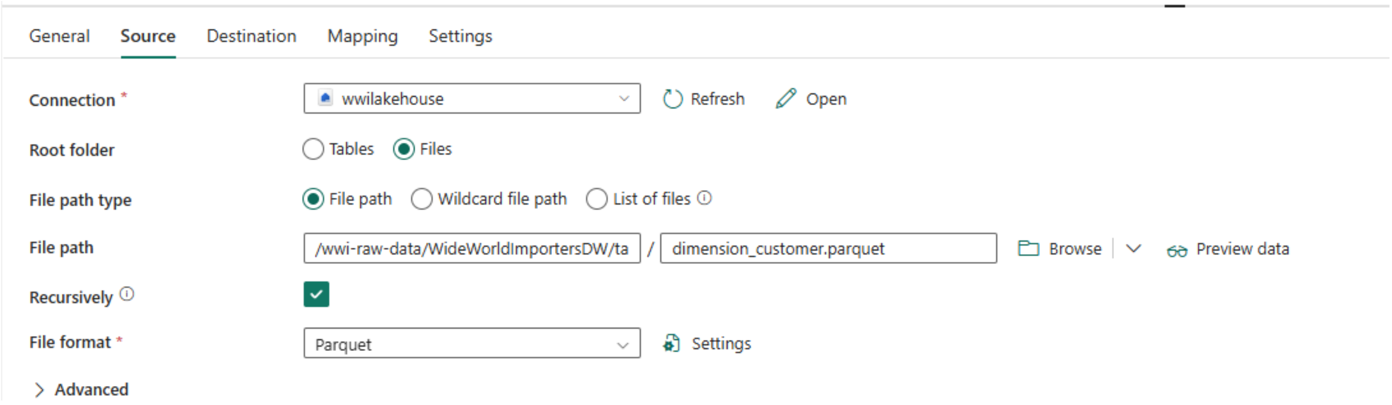
New sources

Your search doesn't match any data sources.

OneLake data hub

Name	Type
wwilakehouse	Lakehouse
wwilakehouse	Lakehouse

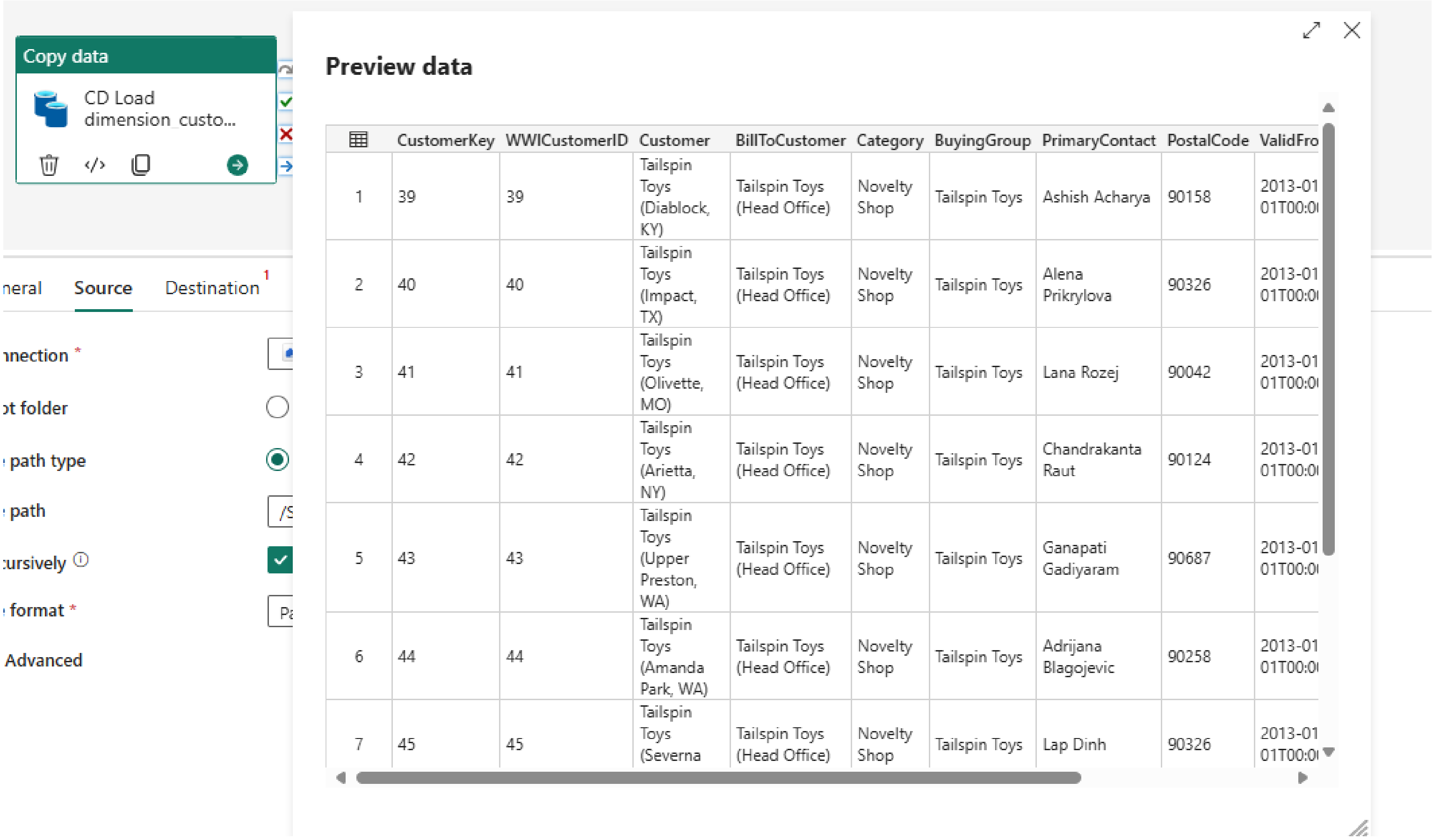
12. On the **File Path**, configure the settings as follows:



Connection *	wwilakehouse
Root folder	○ Tables ● Files
File path type	● File path ○ Wildcard file path ○ List of files
File path	/wwi-raw-data/WideWorldImportersDW/ta / dimension_customer.parquet
Recursively	☑
File format *	Parquet

File path – Directory: /wwi-raw-data/WideWorldImportersDW/tables
File path – File name: dimension_customer.parquet
File format: Parquet

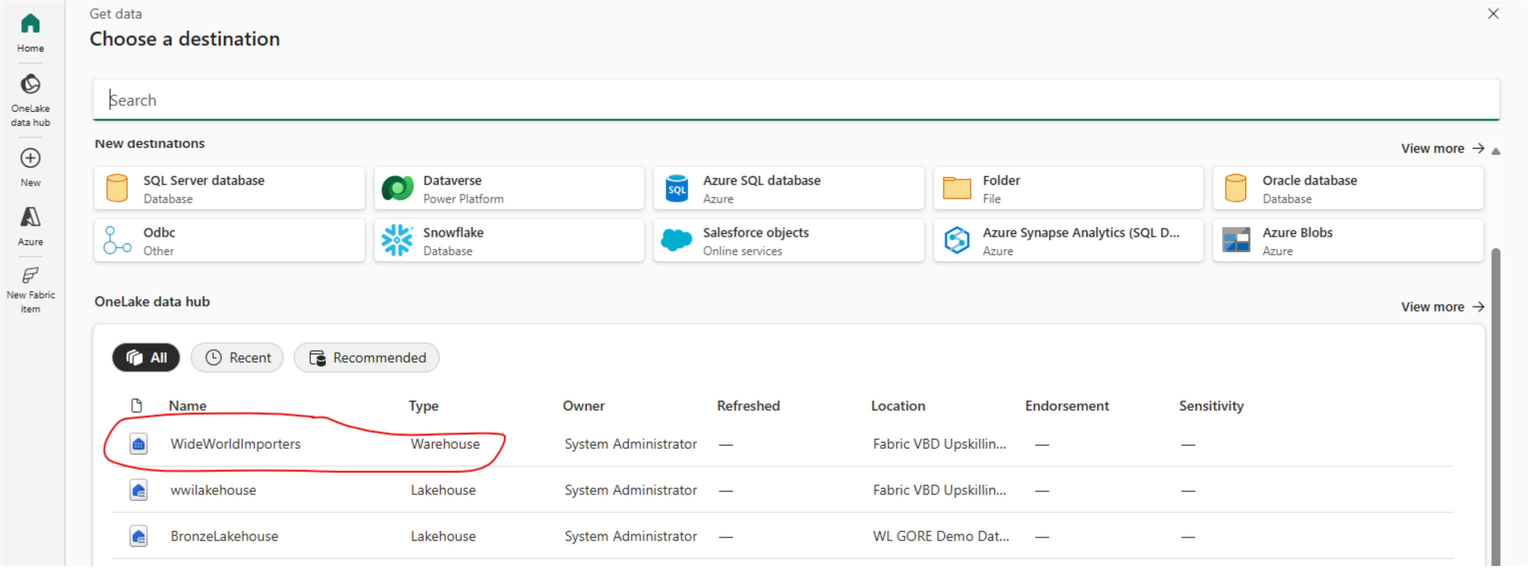
13. Select **Preview data** next to the **File path** setting to ensure there are no errors.



Preview data

	CustomerKey	WWICustomerID	Customer	BillToCustomer	Category	BuyingGroup	PrimaryContact	PostalCode	ValidFro
1	39	39	Tailspin Toys (Diablock, KY)	Tailspin Toys (Head Office)	Novelty Shop	Tailspin Toys	Ashish Acharya	90158	2013-01 01T00:0
2	40	40	Tailspin Toys (Impact, TX)	Tailspin Toys (Head Office)	Novelty Shop	Tailspin Toys	Alena Prikrylova	90326	2013-01 01T00:0
3	41	41	Tailspin Toys (Olivette, MO)	Tailspin Toys (Head Office)	Novelty Shop	Tailspin Toys	Lana Rozej	90042	2013-01 01T00:0
4	42	42	Tailspin Toys (Arietta, NY)	Tailspin Toys (Head Office)	Novelty Shop	Tailspin Toys	Chandrakanta Raut	90124	2013-01 01T00:0
5	43	43	Tailspin Toys (Upper Preston, WA)	Tailspin Toys (Head Office)	Novelty Shop	Tailspin Toys	Ganapati Gadiyaram	90687	2013-01 01T00:0
6	44	44	Tailspin Toys (Amanda Park, WA)	Tailspin Toys (Head Office)	Novelty Shop	Tailspin Toys	Adrijana Blagojevic	90258	2013-01 01T00:0
7	45	45	Tailspin Toys (Severna	Tailspin Toys (Head Office)	Novelty Shop	Tailspin Toys	Lap Dinh	90326	2013-01 01T00:0

14. On the **Destination** page, select **WideWorldImporters** from the list. If it is not available, then click on **More** to browse in OneLake data hub



Get data

Choose a destination

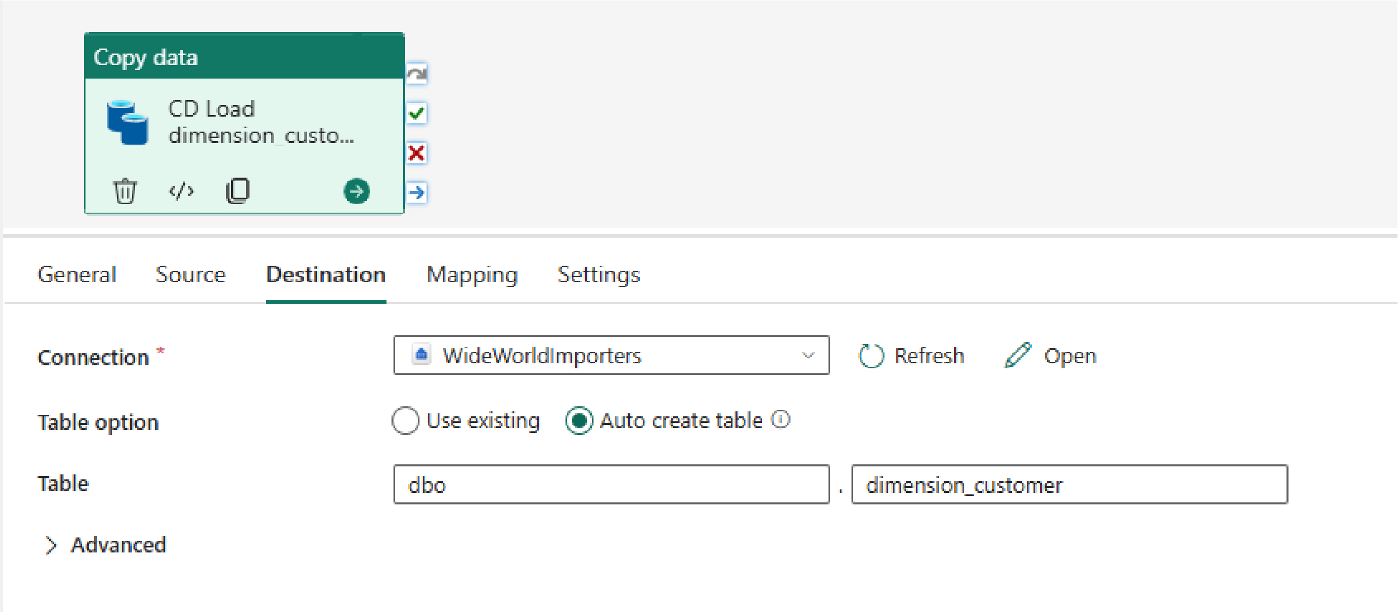
New destinations

SQL Server database Database	Dataverse Power Platform	Azure SQL database Azure	Folder File	Oracle database Database
Odbc Other	Snowflake Database	Salesforce objects Online services	Azure Synapse Analytics (SQL D... Azure	Azure Blobs Azure

OneLake data hub

Name	Type	Owner	Refreshed	Location	Endorsement	Sensitivity
WideWorldImporters	Warehouse	System Administrator	—	Fabric VBD Upskillin...	—	—
wwilakehouse	Lakehouse	System Administrator	—	Fabric VBD Upskillin...	—	—
BronzeLakehouse	Lakehouse	System Administrator	—	WL GORE Demo Dat...	—	—

15. Next to the **Table option** setting, select **Auto create table**
16. In the first box (schema name) next to the **Table** setting, enter **dbo**.
17. In the second box(table name) next to the **Table** setting, enter **dimension_customer**.

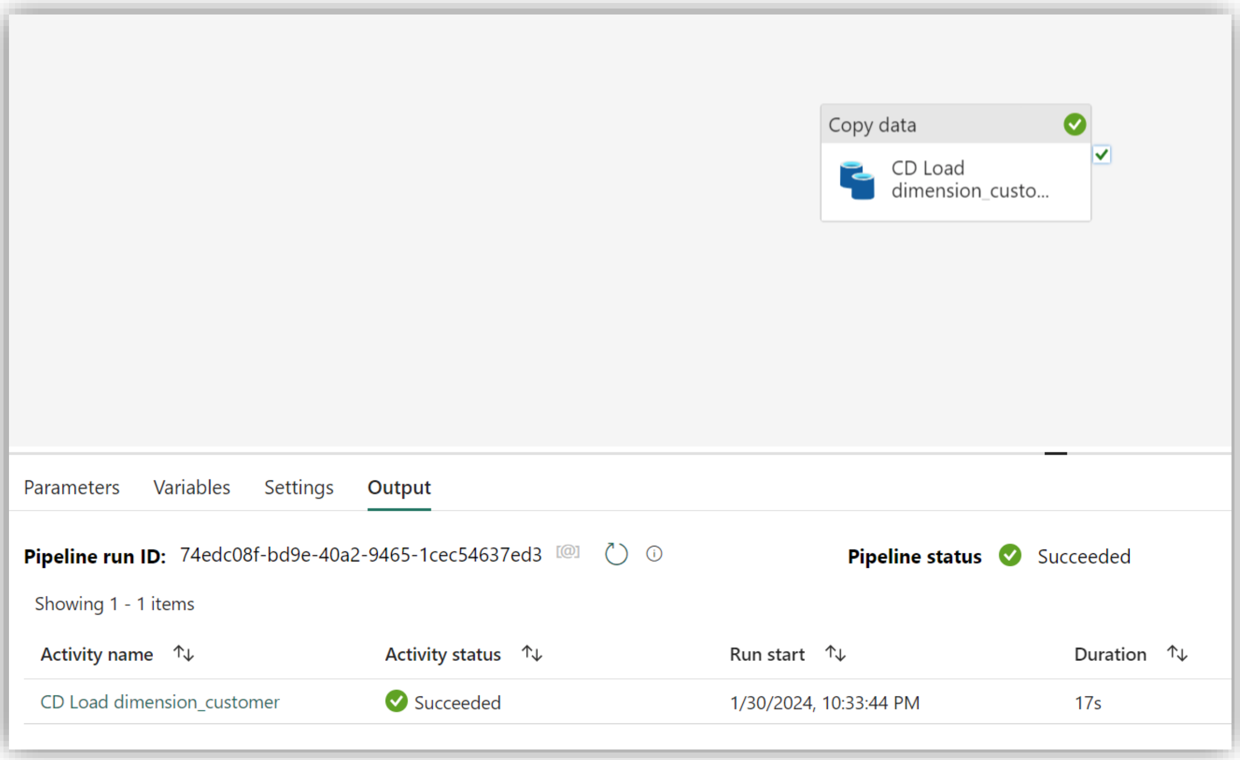


Connection * WideWorldImporters

Table option ○ Use existing ● Auto create table

Table dbo . dimension_customer

18. From the ribbon, select **Run**.
19. Select **Save and run** from the dialog box. The pipeline to load the dimension_customer table with start.
20. Monitor the copy activity's progress on the **Output** page and wait for it to complete.



Pipeline run ID: 74edc08f-bd9e-40a2-9465-1cec54637ed3

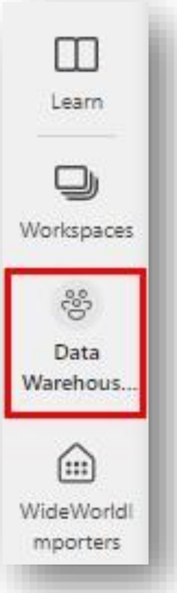
Pipeline status Succeeded

Showing 1 - 1 items

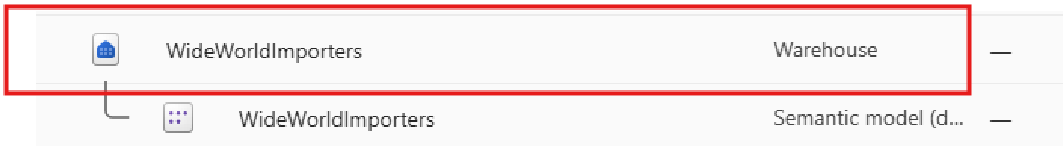
Activity name	Activity status	Run start	Duration
CD Load dimension_customer	Succeeded	1/30/2024, 10:33:44 PM	17s

Building a report

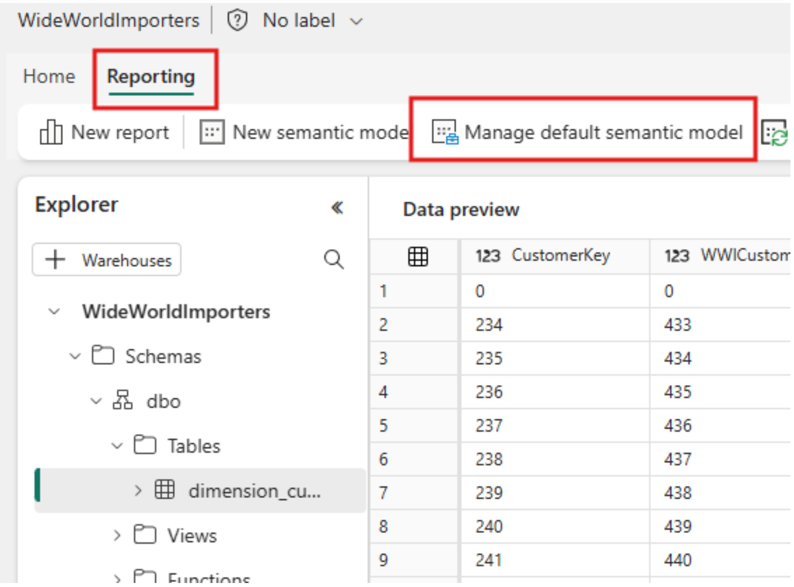
1. Select **Data Warehouse Tutorial** in the left-hand navigation menu to return to the workspace artifact view.



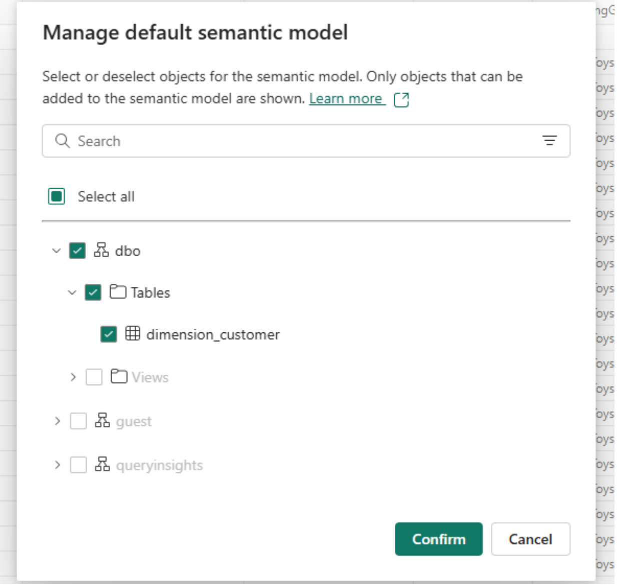
2. From the artifact list, select **WideWorldImporters** with the type of **Warehouse,** navigate to tab **Reporting** and click on **Manage default semantic model.** Select under Tables “dimension_customer” and click on **Confirm**.



WideWorldImporters	Warehouse	—
WideWorldImporters	Semantic model (d...	—



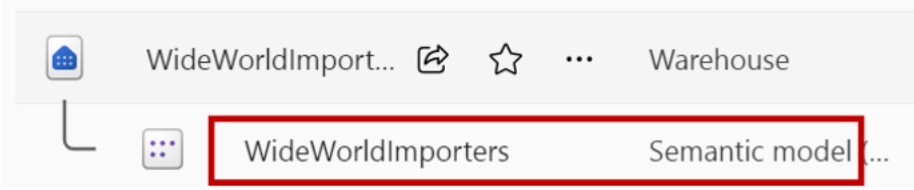
	123 CustomerKey	123 WWICustom
1	0	0
2	234	433
3	235	434
4	236	435
5	237	436
6	238	437
7	239	438
8	240	439
9	241	440

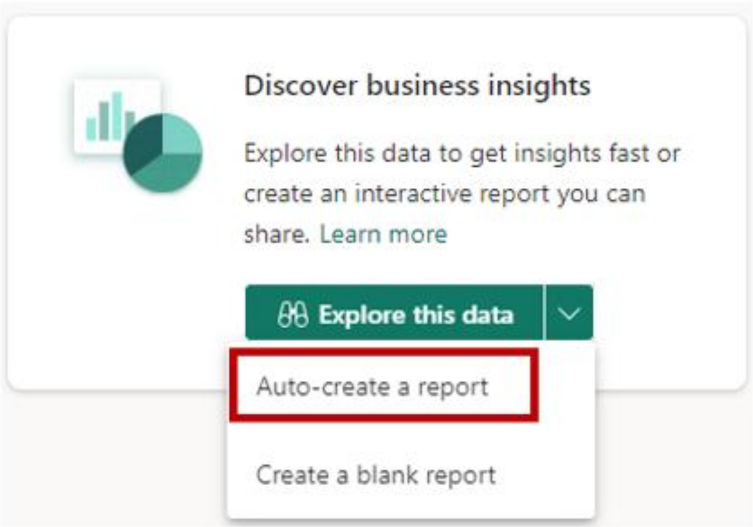


Manage default semantic model

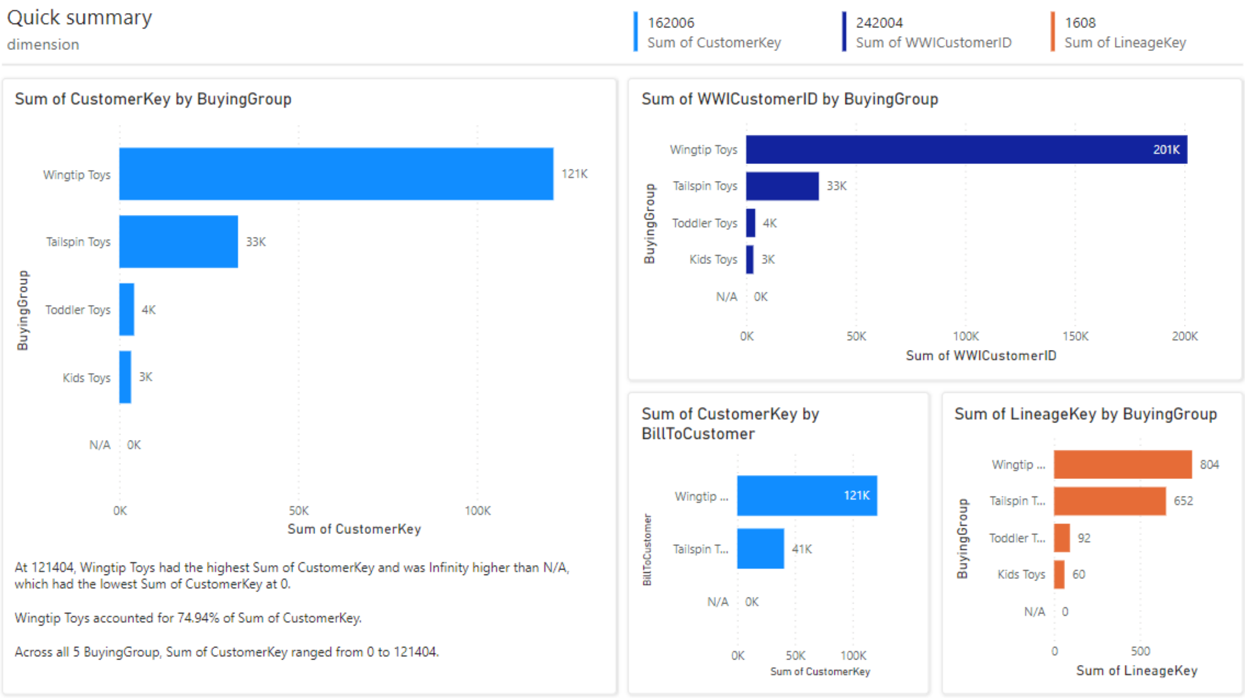
Select or deselect objects for the semantic model. Only objects that can be added to the semantic model are shown. Learn more

3. Return to the workspace artifact view, from the artifact list, select **WideWorldImporters** with the type of **Semantic Model.** In the **Discover business insights** section, select **Explore this data > Auto-create a report**. A report will be generated from the dimension_customer table that was loaded in the previous section.




Discover business insights
Explore this data to get insights fast or create an interactive report you can share. Learn more
Explore this data
Auto-create a report
Create a blank report

4. A report similar to one shown below will be generated.

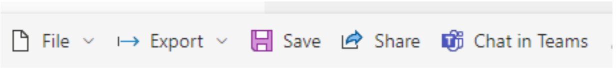


At 121404, Wingtip Toys had the highest Sum of CustomerKey and was Infinity higher than N/A, which had the lowest Sum of CustomerKey at 0.

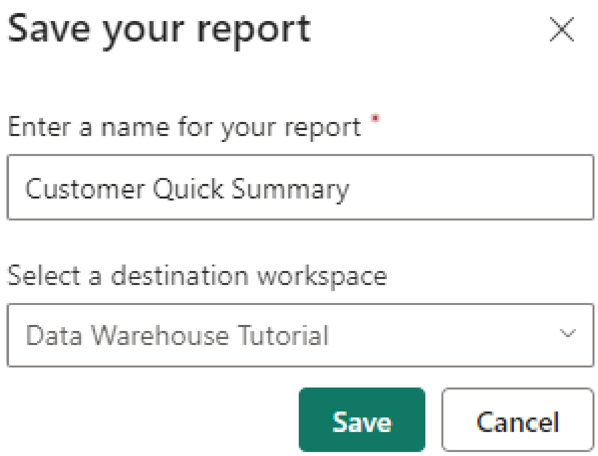
Wingtip Toys accounted for 74.94% of Sum of CustomerKey.

Across all 5 BuyingGroup, Sum of CustomerKey ranged from 0 to 121404.

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



6.
7. Enter **Customer Quick Summary** in the name box.
8. Select **Save**.



Save your report

Enter a name for your report *

Customer Quick Summary

Select a destination workspace

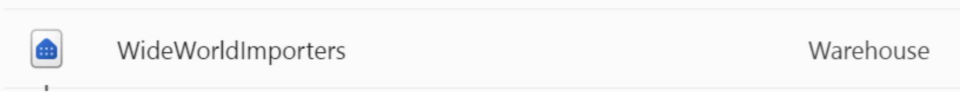
Data Warehouse Tutorial

Module 3: Extending the solution

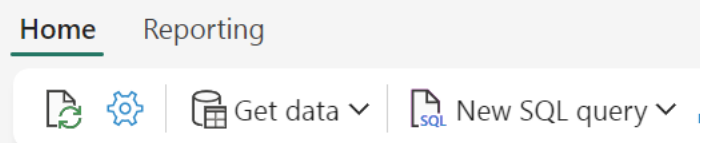
Now that you have see how to build a data warehouse, load a table, and generate a report it is time to extend the solution by exploring additional methods for loading data, querying data, and building reports.

Creating tables in the data warehouse

1. Select **Workspaces** in the left-hand menu of the Power BI service.
2. Select the workspace created in **Module 1: Getting started**, such as **Data Warehouse Tutorial**.
3. From the artifact list, select **WideWorldImporters** with the type of **Warehouse**.



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

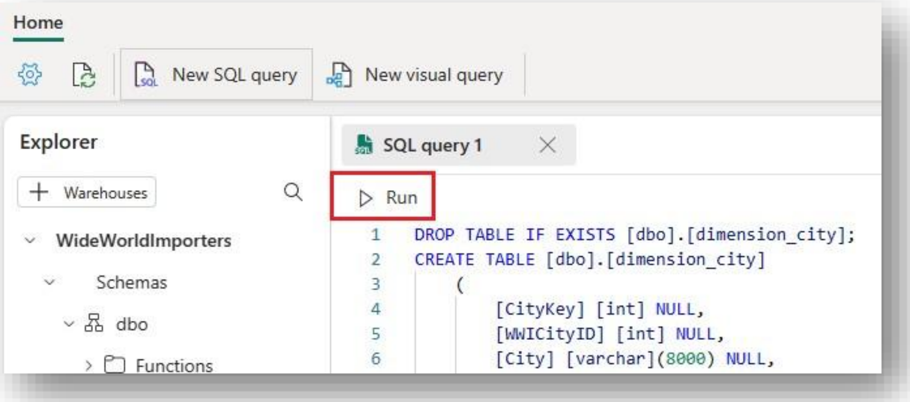


5. In the query editor, paste the code below.

   Note: In case of issues with copy/paste formatting, a text file containing the script called **Create Tables.txt** can be accessed from the Scripts folder.

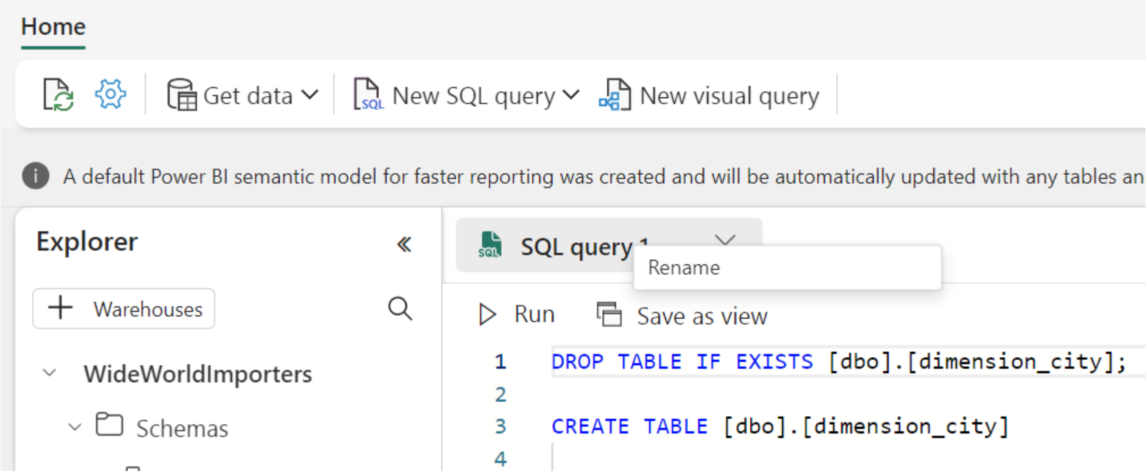
```
/*
1. Drop the dimension_city table if it already exists.
2. Create the dimension_city table.
3. Drop the fact_sale table if it already exists.
4. Create the fact_sale table.
*/


--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.



```
DROP TABLE IF EXISTS [dbo].[dimension_city];
CREATE TABLE [dbo].[dimension_city]
    (
        [CityKey] [int] NULL,
        [WWICityID] [int] NULL,
        [City] [varchar](8000) NULL,
```

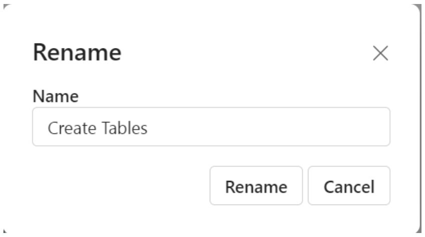
7. To save this query for reference later, right-click on the query tab just above the editor and select **Rename**.



A default Power BI semantic model for faster reporting was created and will be automatically updated with any tables an

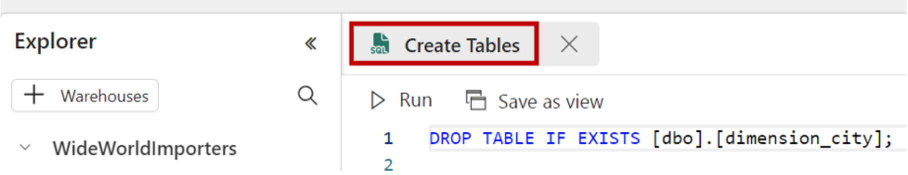
```
DROP TABLE IF EXISTS [dbo].[dimension_city];

CREATE TABLE [dbo].[dimension_city]
```

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



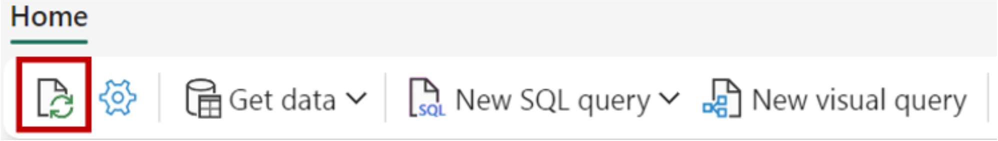
Name: Create Tables

9. Click **Rename** to save the query with a given name

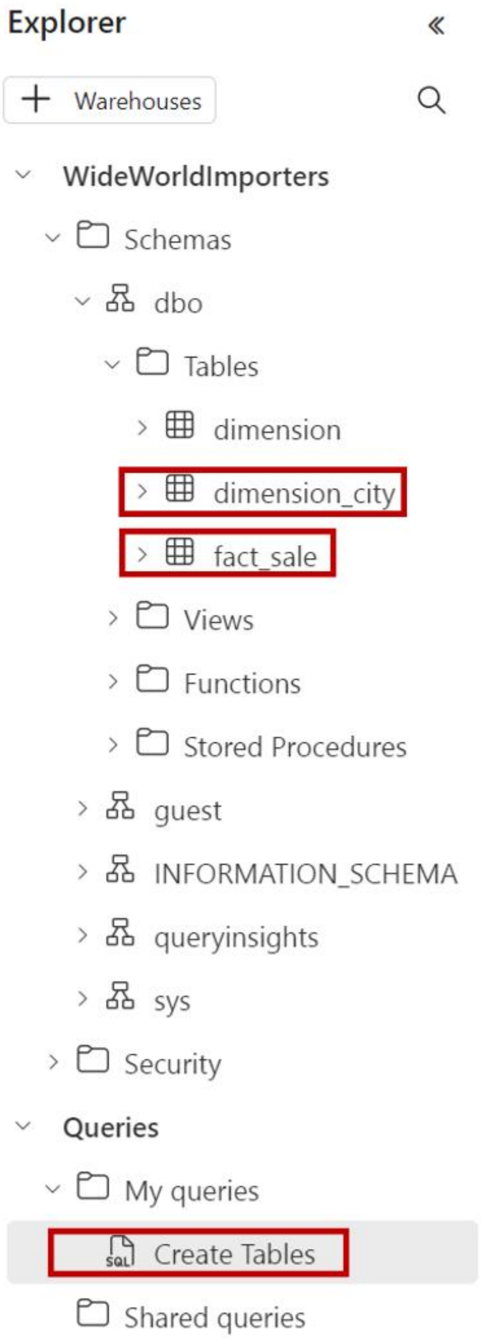


```
DROP TABLE IF EXISTS [dbo].[dimension_city];
```

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



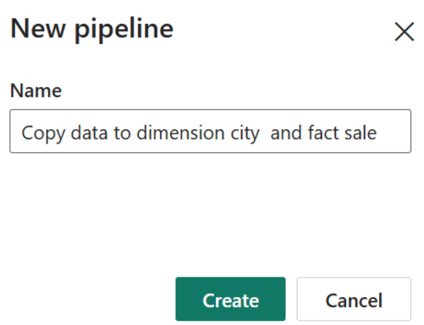
11. In the **Object explorer** verify that you can see the newly created **Create Tables** query, **fact_sale** table, and **dimension_city** table.



Loading data using Pipeline

1. From the Data factory experience , select **New Data Pipeline**

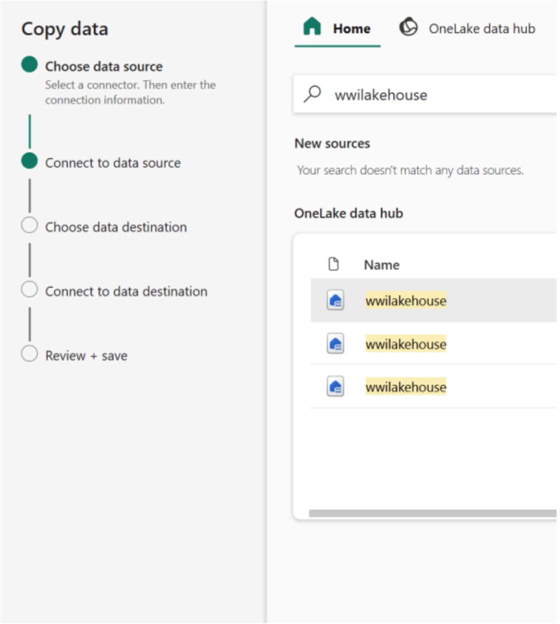
2. Name the Pipeline **Copy data to dimension city and fact sale,** once pipeline is created, choose tile 'Copy data assistant' to get into the data assistant



New pipeline

Name: Copy data to dimension city and fact sale

3. In the copy data assistant window, search for **wwilakehouse** and select the lakehouse you own as it will show multiple lakehouse



Copy data

- Choose data source — Select a connector. Then enter the connection information.
- Connect to data source
- Choose data destination
- Connect to data destination
- Review + save

wwilakehouse

New sources

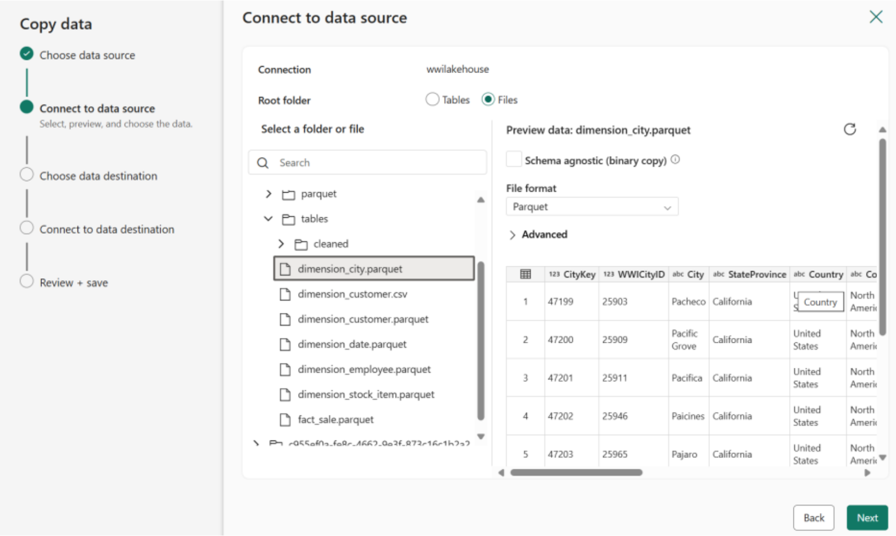
Your search doesn't match any data sources.

OneLake data hub

Name
wwilakehouse
wwilakehouse
wwilakehouse

4. Navigate to OneLake -> wwilakehouse -> files section -> /wwi-raw-data/WideWorldImportersDW/tables

Select the file **dimension_city.parquet** as the source file. Click next



Connection: wwilakehouse

Root folder: ○ Tables ● Files

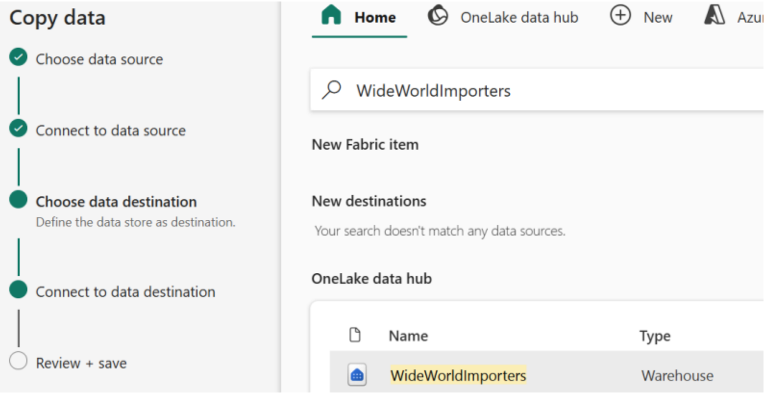
Preview data: dimension_city.parquet

□ Schema agnostic (binary copy)

File format: Parquet

	CityKey	WWICityID	City	StateProvince	Country	Co
1	47199	25903	Pacheco	California	Country	North Ameri
2	47200	25909	Pacific Grove	California	United States	North Ameri
3	47201	25911	Pacifica	California	United States	North Ameri
4	47202	25946	Paicines	California	United States	North Ameri
5	47203	25965	Pajaro	California	United States	North Ameri

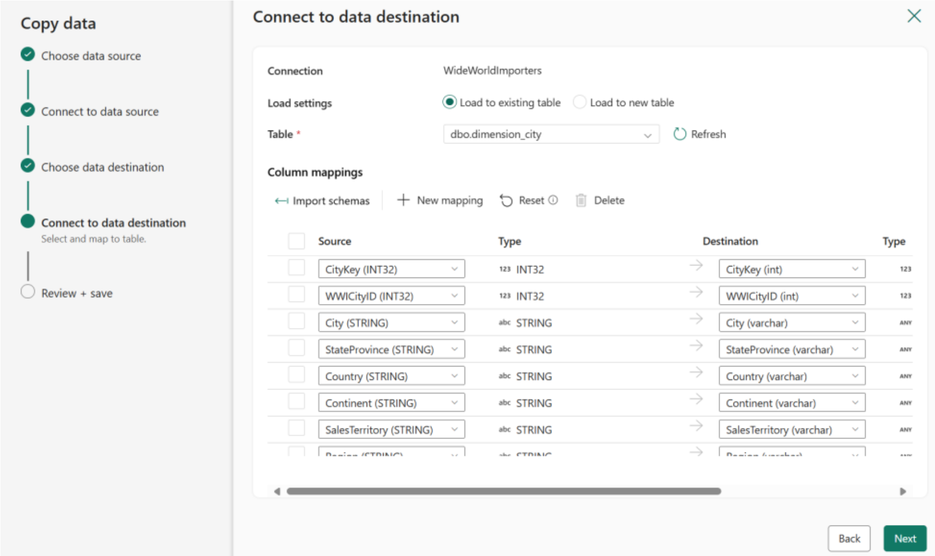
5. In destination choose warehouse **WideWorldImporters**. Select the one you own as multiple warehouses with the same name will show up



WideWorldImporters

Name	Type
WideWorldImporters	Warehouse

6. Load to existing table **dbo.dimension_city**. Click next



Copy data

- Choose data source
- Connect to data source
- Choose data destination
- Connect to data destination
  Select and map to table.
- Review + save

Connect to data destination

Connection: WideWorldImporters

Load settings: ● Load to existing table ○ Load to new table

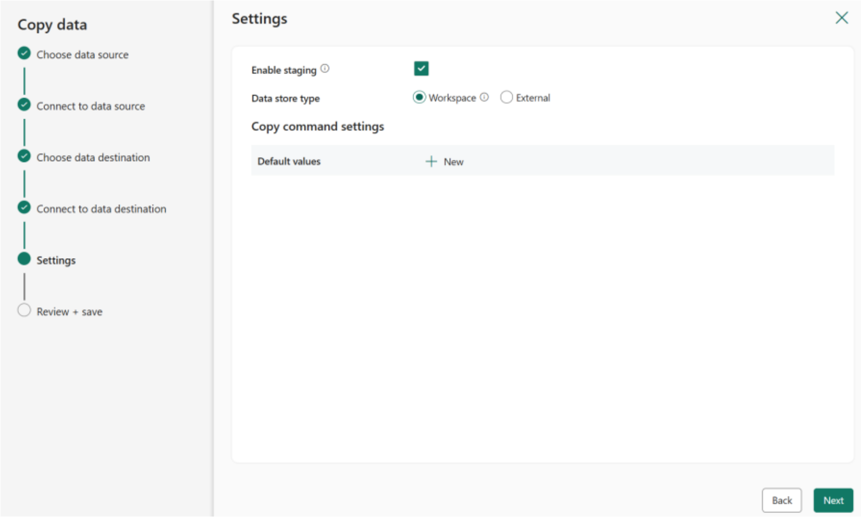
Table *: dbo.dimension_city · Refresh

Column mappings

Import schemas · New mapping · Reset · Delete

	Source	Type		Destination	Type
□	CityKey (INT32)	123 INT32	→	CityKey (int)	123
□	WWICityID (INT32)	123 INT32	→	WWICityID (int)	123
□	City (STRING)	abc STRING	→	City (varchar)	ANY
□	StateProvince (STRING)	abc STRING	→	StateProvince (varchar)	ANY
□	Country (STRING)	abc STRING	→	Country (varchar)	ANY
□	Continent (STRING)	abc STRING	→	Continent (varchar)	ANY
□	SalesTerritory (STRING)	abc STRING	→	SalesTerritory (varchar)	ANY
□	Region (STRING)	abc STRING	→	Region (varchar)	ANY

7. Make sure **enable staging** is enabled. Click next



Copy data

- Choose data source
- Connect to data source
- Choose data destination
- Connect to data destination
- Settings
- Review + save

Settings

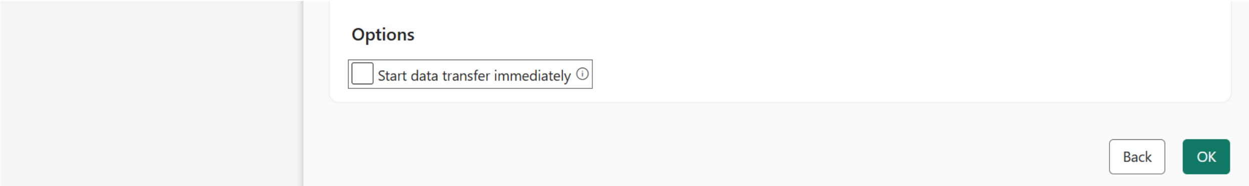
Enable staging ☑

Data store type ● Workspace ○ External

Copy command settings

Default values + New

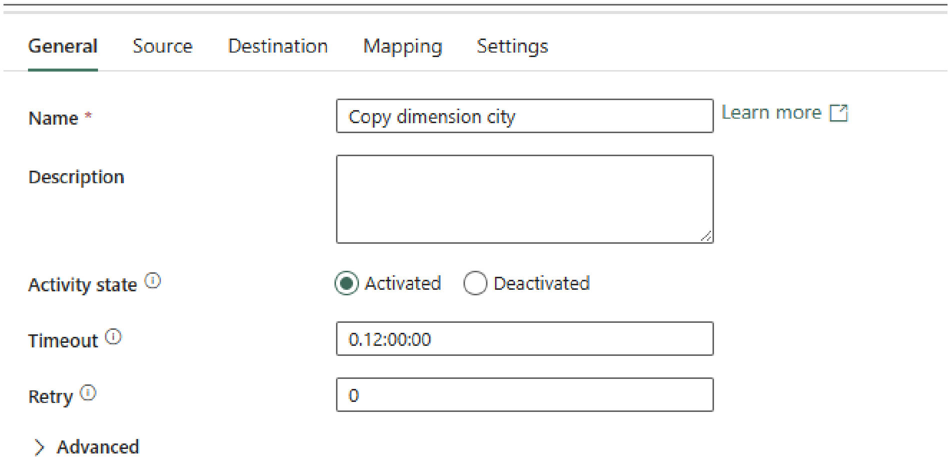
8. Make sure option **Start data transfer immediately** is unchecked as we will run it later. Click ok



Options

□ Start data transfer immediately

9. Rename the copy data activity as **Copy dimension city** in the general tab of the copy activity.



General | Source | Destination | Mapping | Settings

Name * : Copy dimension city — Learn more

Description :

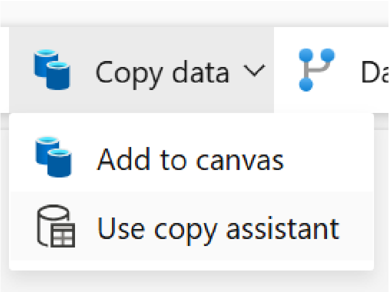
Activity state : ● Activated ○ Deactivated

Timeout : 0.12:00:00

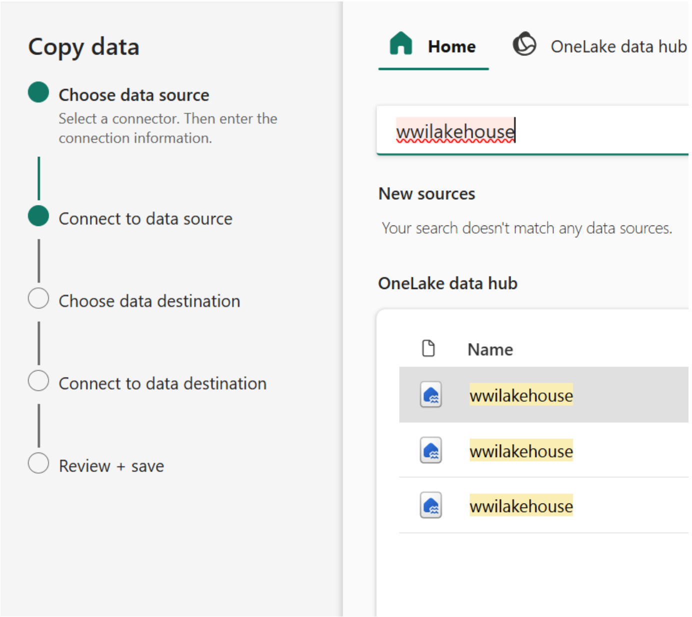
Retry : 0

> Advanced

10. Add one more copy data activity to the canvas. Choose **Use copy assistant** option



11. Select **wwilakehouse** as the lakehouse source. Make sure to select the one you own ad multiple lakehouse will show up.



Copy data

● Choose data source
Select a connector. Then enter the connection information.

● Connect to data source

○ Choose data destination

○ Connect to data destination

○ Review + save

Home | OneLake data hub

wwilakehouse

New sources

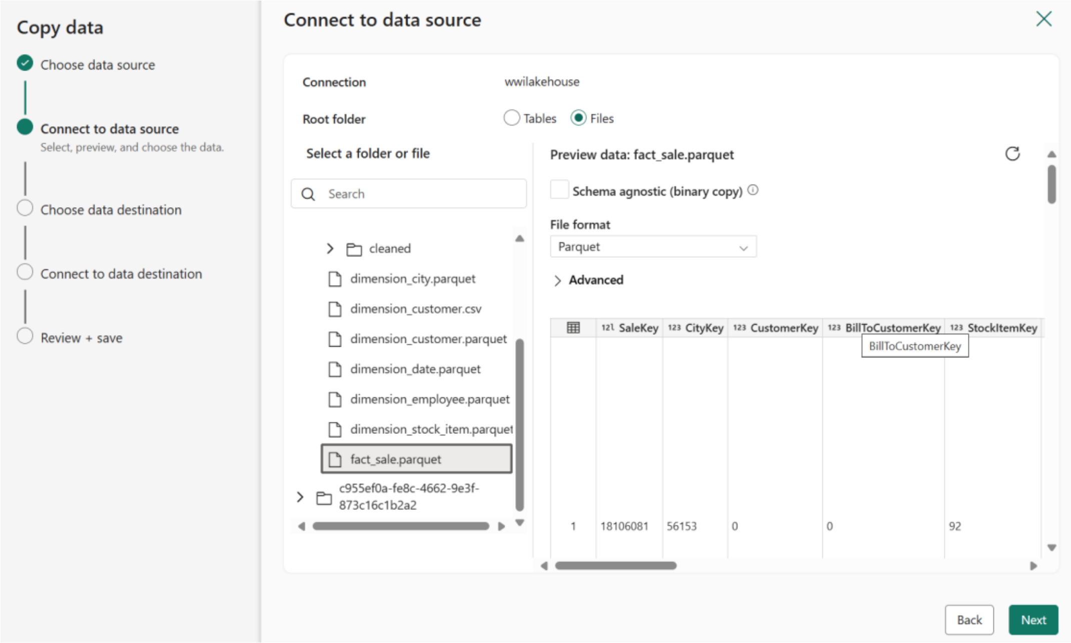
Your search doesn't match any data sources.

OneLake data hub

Name
wwilakehouse
wwilakehouse
wwilakehouse

12. Navigate to OneLake -> wwilakehouse -> files section -> /wwi-raw-data/WideWorldImportersDW/tables

    Select **fact_sale.parquet** as the source file. Click next



Copy data

- ✓ Choose data source
- ● Connect to data source
  Select, preview, and choose the data.
- ○ Choose data destination
- ○ Connect to data destination
- ○ Review + save

Connect to data source

Connection: wwilakehouse

Root folder: ○ Tables ● Files

Select a folder or file

Search

- cleaned
- dimension_city.parquet
- dimension_customer.csv
- dimension_customer.parquet
- dimension_date.parquet
- dimension_employee.parquet
- dimension_stock_item.parquet
- fact_sale.parquet
- c955ef0a-fe8c-4662-9e3f-873c16c1b2a2

Preview data: fact_sale.parquet

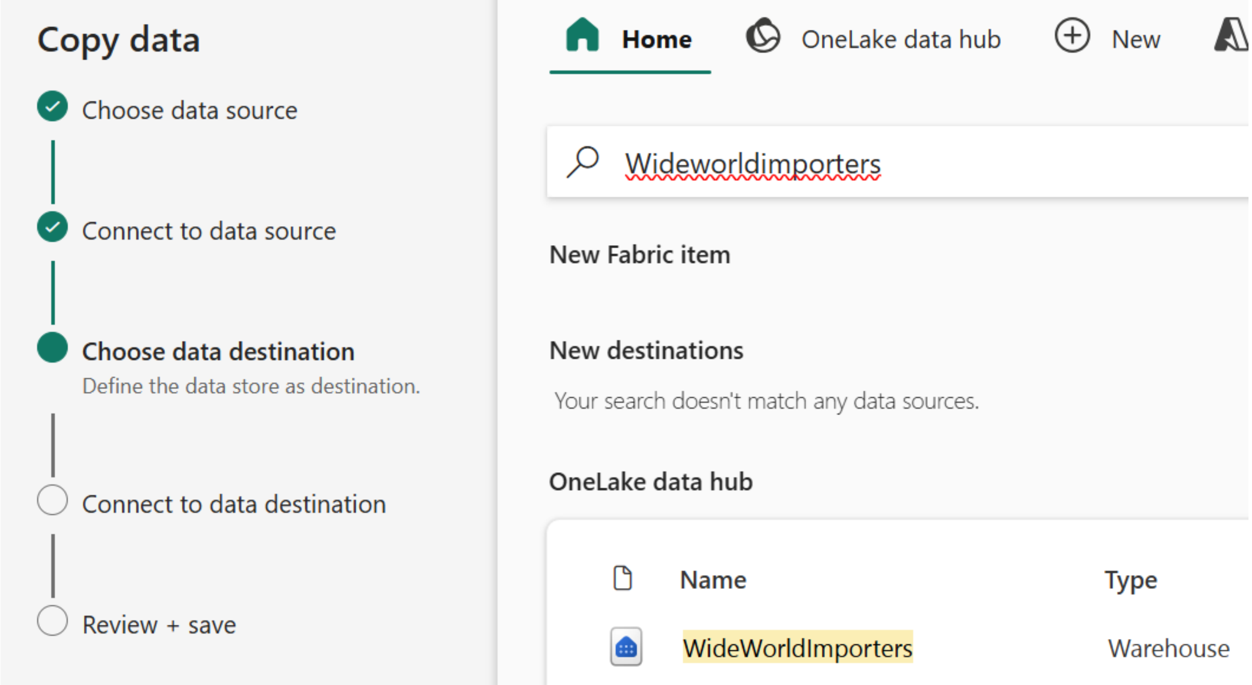
□ Schema agnostic (binary copy)

File format: Parquet

Advanced

	SaleKey	CityKey	CustomerKey	BillToCustomerKey	StockItemKey
1	18106081	56153	0	0	92

1. Choose **Wideworksimporters** warehouse as the target destination . Make sure to select the one you own as multiples will show up.



Copy data

- ✓ Choose data source
- ✓ Connect to data source
- ● Choose data destination
  Define the data store as destination.
- ○ Connect to data destination
- ○ Review + save

Wideworldimporters

New Fabric item

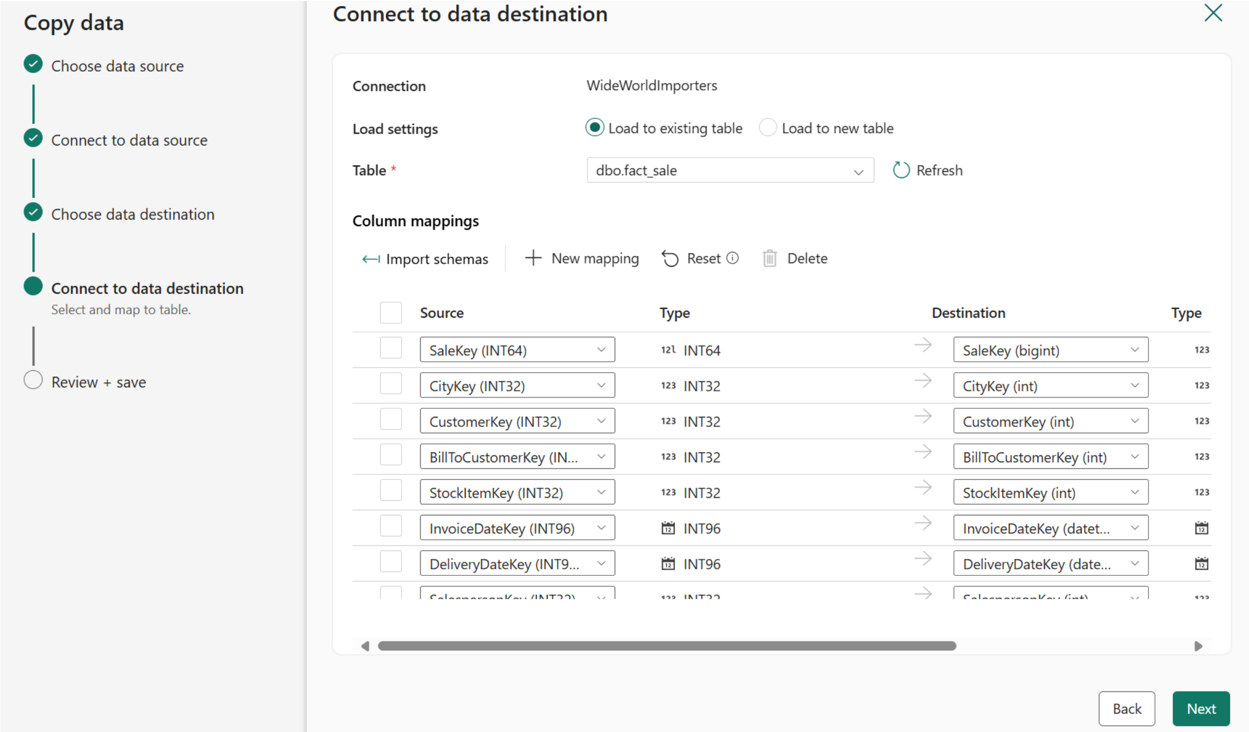
New destinations

Your search doesn't match any data sources.

OneLake data hub

Name	Type
WideWorldImporters	Warehouse

2. In the mappings tab , Make sure to delete the mapping for **Month , Year and Quarter** as the source file doesn't contain these columns. Click Next



Copy data

- ✓ Choose data source
- ✓ Connect to data source
- ✓ Choose data destination
- ● Connect to data destination
  Select and map to table.
- ○ Review + save

Connect to data destination

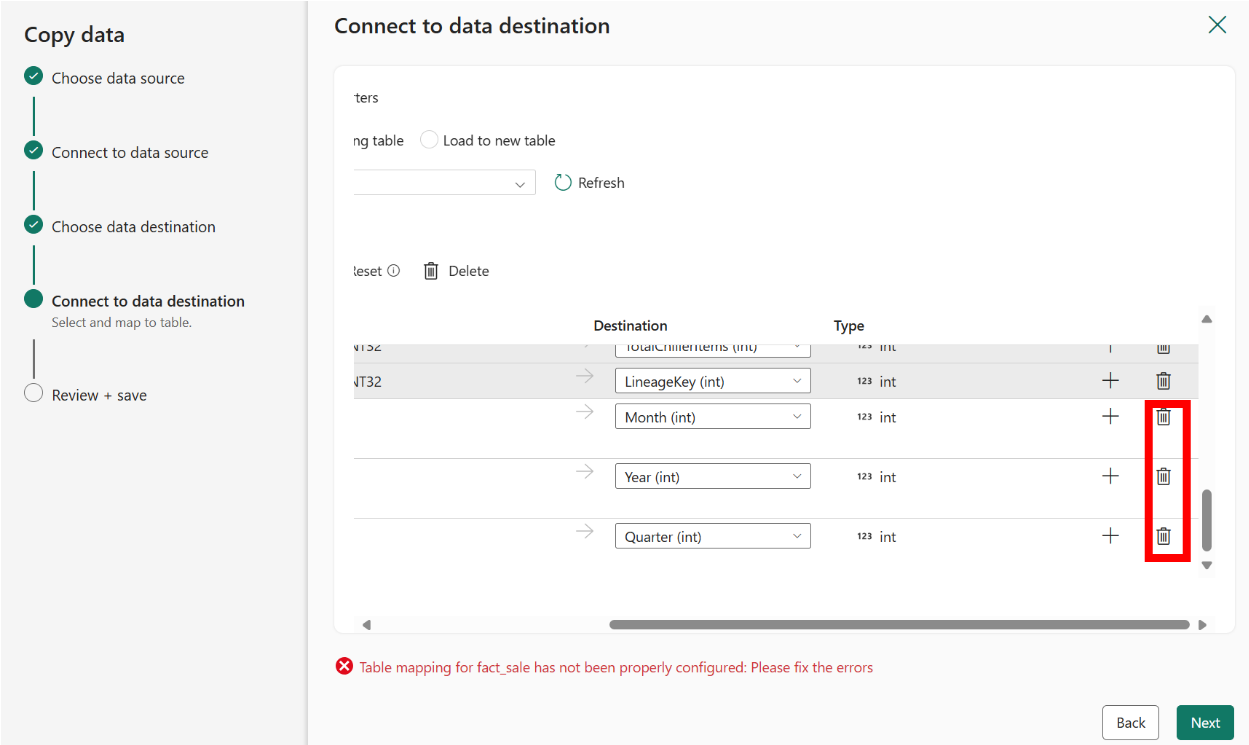
Connection: WideWorldImporters

Load settings: ● Load to existing table ○ Load to new table

Table *: dbo.fact_sale Refresh

Column mappings

□	Source	Type		Destination	Type
□	SaleKey (INT64)	INT64	→	SaleKey (bigint)	123
□	CityKey (INT32)	INT32	→	CityKey (int)	123
□	CustomerKey (INT32)	INT32	→	CustomerKey (int)	123
□	BillToCustomerKey (IN...	INT32	→	BillToCustomerKey (int)	123
□	StockItemKey (INT32)	INT32	→	StockItemKey (int)	123
□	InvoiceDateKey (INT96)	INT96	→	InvoiceDateKey (datet...	📅
□	DeliveryDateKey (INT9...	INT96	→	DeliveryDateKey (date...	📅
□	SalespersonKey (INT32)	INT32	→	SalespersonKey (int)	123



Copy data

- ✓ Choose data source
- ✓ Connect to data source
- ✓ Choose data destination
- ● Connect to data destination
  Select and map to table.
- ○ Review + save

Connect to data destination

ters

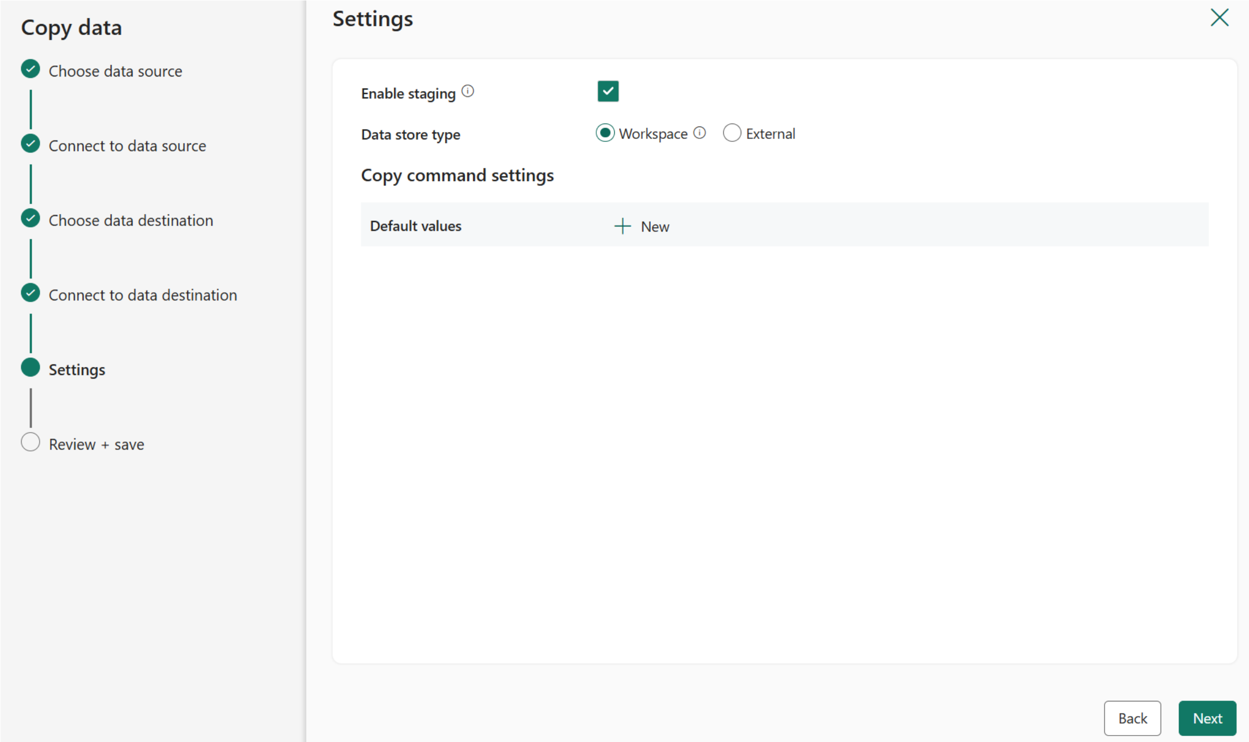
ng table ○ Load to new table

Refresh

		Destination	Type		
NT32	→	TotalChillerItems (int)	int	+	🗑
NT32	→	LineageKey (int)	int	+	🗑
	→	Month (int)	int	+	🗑
	→	Year (int)	int	+	🗑
	→	Quarter (int)	int	+	🗑

Table mapping for fact_sale has not been properly configured: Please fix the errors

3. Make sure the **enable staging option** is checked, Click Next



Copy data

- ✓ Choose data source
- ✓ Connect to data source
- ✓ Choose data destination
- ✓ Connect to data destination
- ● Settings
- ○ Review + save

Settings

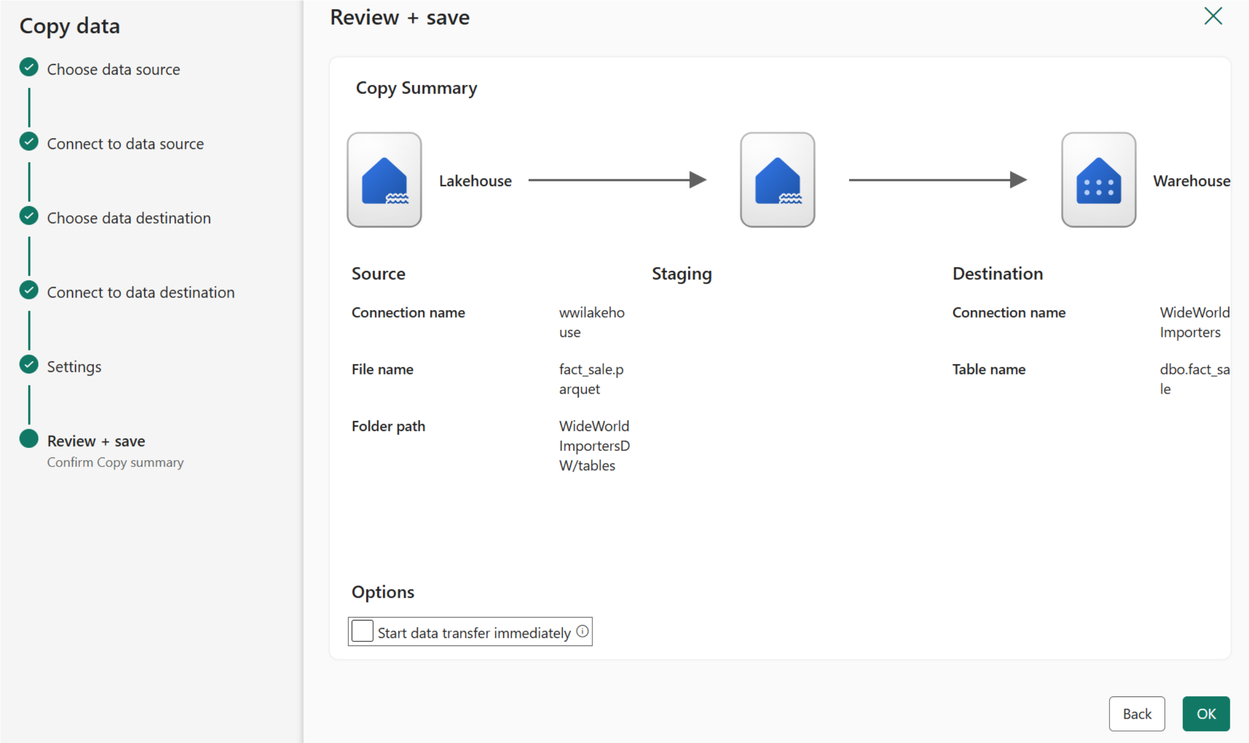
Enable staging ☑

Data store type ● Workspace ○ External

Copy command settings

Default values + New

4. Make sure **Start data transfer immediately** checkbox is unchecked , Click ok



Copy data

- ✓ Choose data source
- ✓ Connect to data source
- ✓ Choose data destination
- ✓ Connect to data destination
- ✓ Settings
- ● Review + save
  Confirm Copy summary

Review + save

Copy Summary

Lakehouse → → Warehouse

Source

Connection name wwilakehouse

File name fact_sale.parquet

Folder path WideWorldImportersDW/tables

Staging

Destination

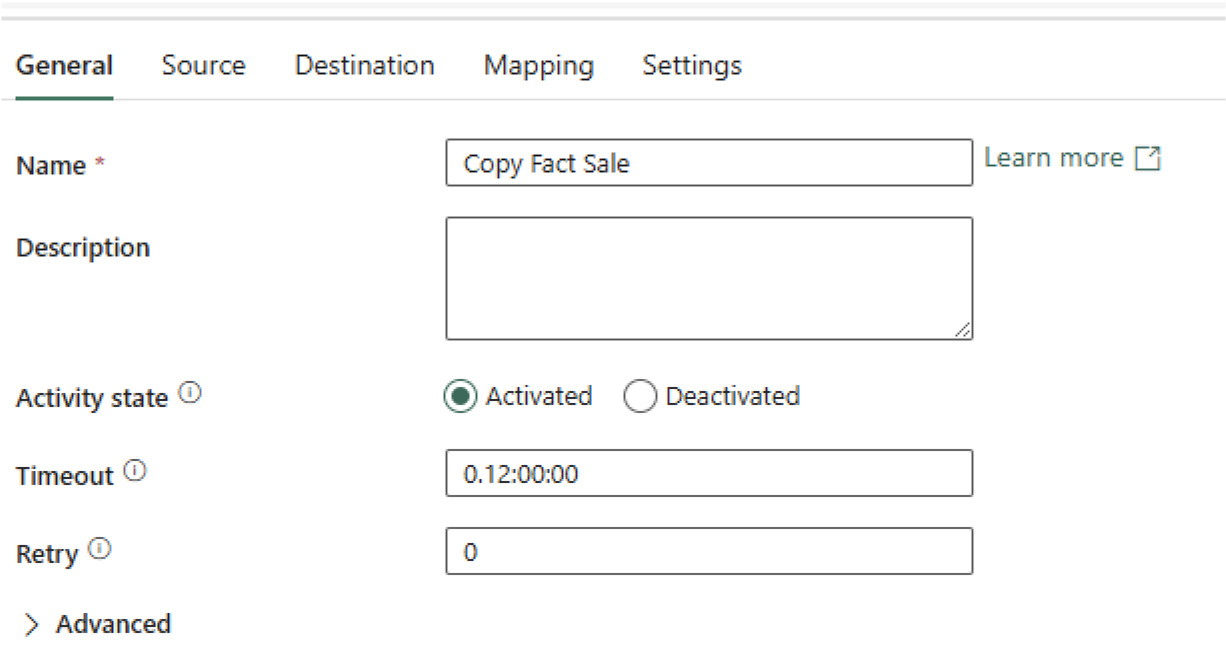
Connection name WideWorld Importers

Table name dbo.fact_sale

Options

□ Start data transfer immediately

5. Rename the copy activity as **Copy Fact Sale**



Name * Copy Fact Sale

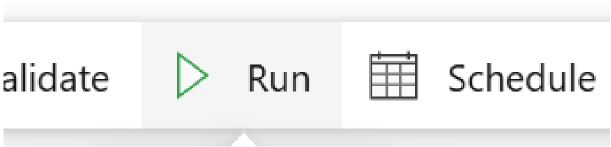
Description

Activity state ● Activated ○ Deactivated

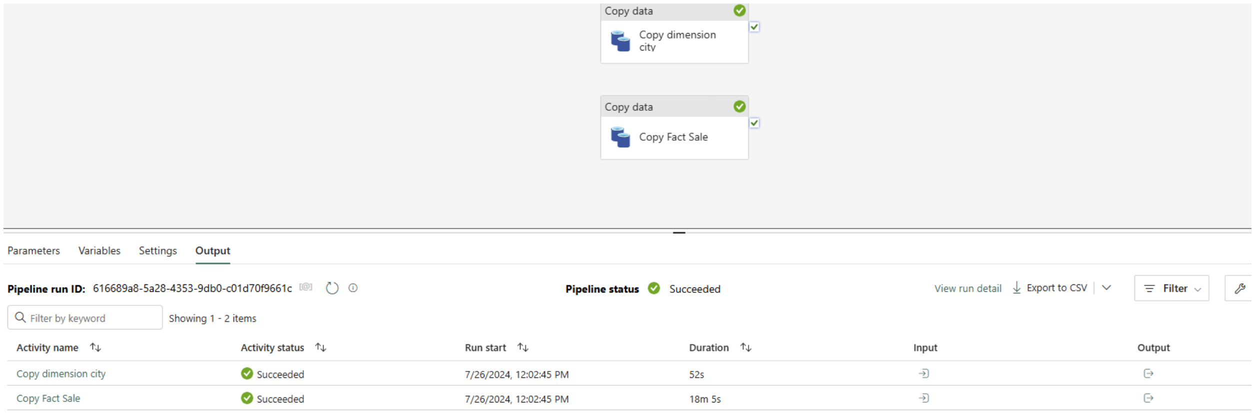
Timeout 0.12:00:00

Retry 0

6. Now that we have both the copy activity defined, Run the entire pipeline by clicking the Run option from the ribbon



7. Monitor the pipeline for successful completion.



Pipeline run ID: 616689a8-5a28-4353-9db0-c01d70f9661c

Pipeline status ✓ Succeeded

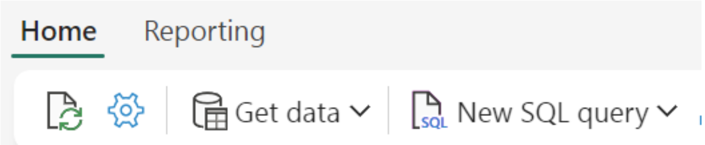
Showing 1 - 2 items

Activity name	Activity status	Run start	Duration	Input	Output
Copy dimension city	✓ Succeeded	7/26/2024, 12:02:45 PM	52s		
Copy Fact Sale	✓ Succeeded	7/26/2024, 12:02:45 PM	18m 5s		

8. Validate in the warehouse WideWorldImporters that the tables have been loaded successfully

Data transformation using a stored procedure

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

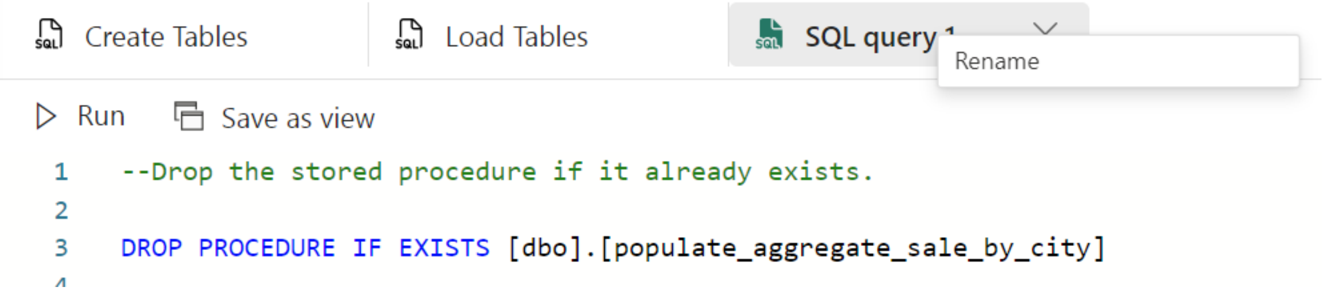


2. In the query editor, paste the code below.
   Note: In case of issues with copy/paste formatting, a text file containing the script called **Create Aggregate Procedure.txt** from the Scripts folder .

```
--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
```

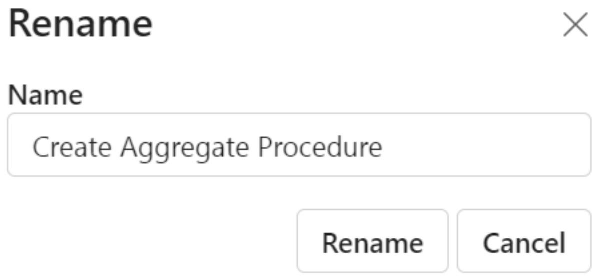
```
        FS.[InvoiceDateKey],
DC.[City],
        DC.[StateProvince],
        DC.[SalesTerritory]
    ORDER BY
        FS.[InvoiceDateKey],
        DC.[StateProvince],
        DC.[City];
END
```

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



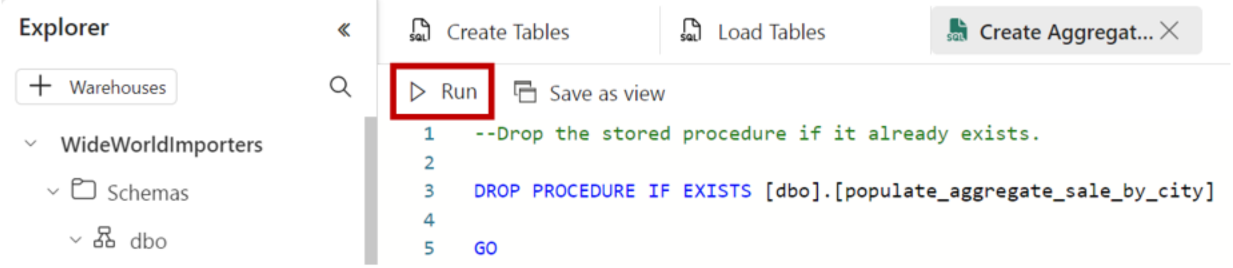
```
--Drop the stored procedure if it already exists.

DROP PROCEDURE IF EXISTS [dbo].[populate_aggregate_sale_by_city]
```

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



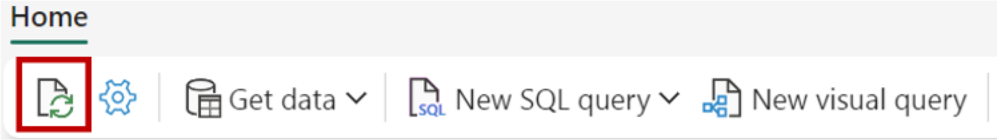
Name: Create Aggregate Procedure

3. Click on **Rename** to save the query with given name
4. Select **Run** to execute the query.

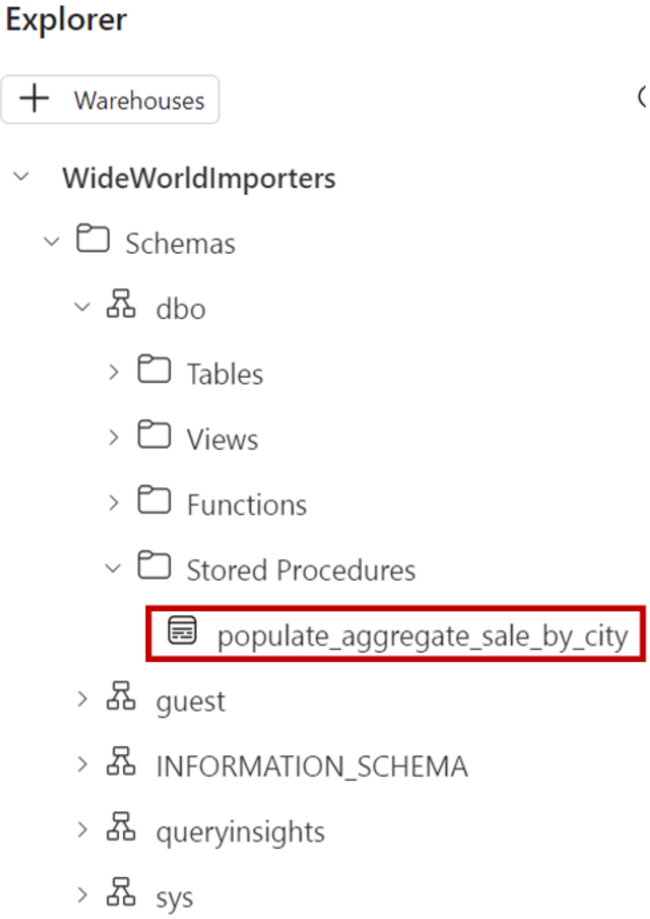


```
--Drop the stored procedure if it already exists.

DROP PROCEDURE IF EXISTS [dbo].[populate_aggregate_sale_by_city]

GO
```

5. Click the **refresh** button on the ribbon.



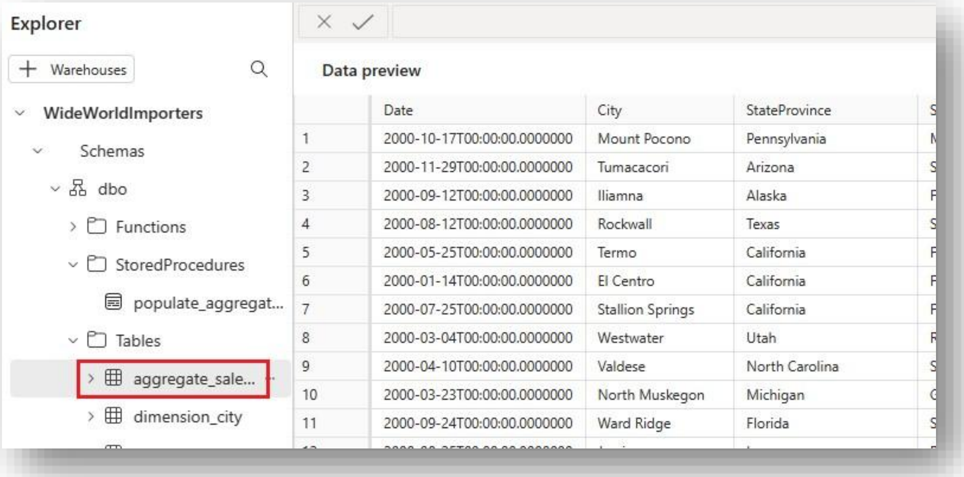
6. In the **Object explorer** verify that you can see the newly created stored procedure by expanding the **StoredProcedures** node under the **dbo** schema.



7. From the **Home** tab of the ribbon, select **New SQL query**.
8. In the query editor, paste the code below.
   Note: In case of issues with copy/paste formatting, a text file containing the script called **Run Aggregate Procedure.txt** can be accessed from the Scripts folder .

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

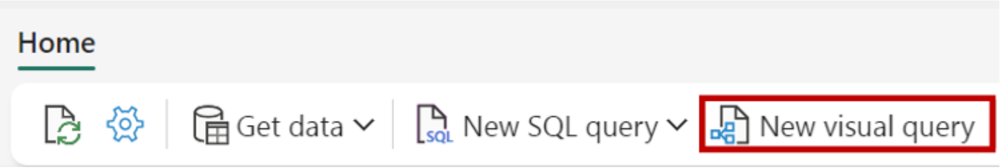
9. To save this query for reference later, right-click on the query tab just above the editor and select **Rename**.
10. Type **Run Create Aggregate Procedure** to change the name of the query.
11. Select **Run** to execute the query.
12. Click the **refresh** button on the ribbon. The query will take between 2 and 3 minutes to execute.
13. In the **Object explorer**, load the data preview to validate the data loaded successfully by clicking on the **aggregate_sale_by_city** table in the **Explorer**.



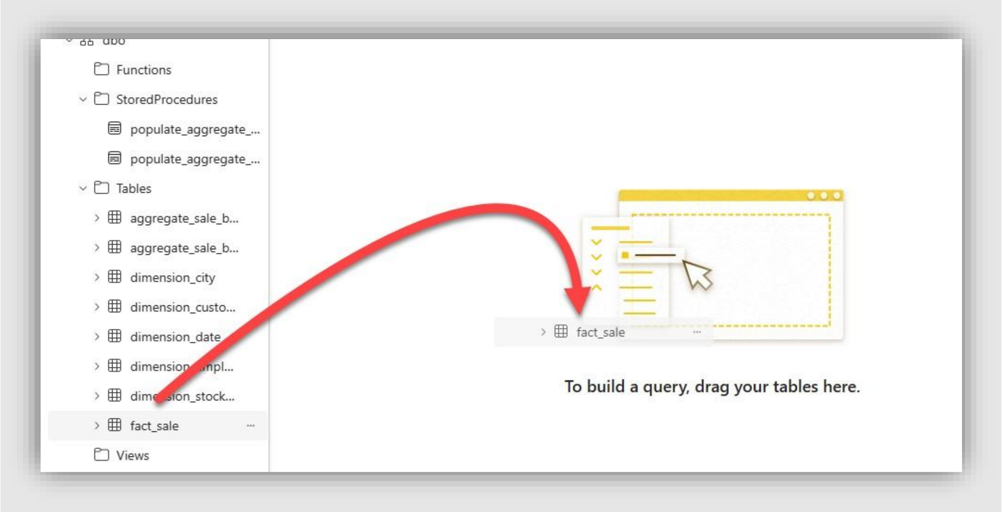
	Date	City	StateProvince	S
1	2000-10-17T00:00:00.0000000	Mount Pocono	Pennsylvania	M
2	2000-11-29T00:00:00.0000000	Tumacacori	Arizona	S
3	2000-09-12T00:00:00.0000000	Iliamna	Alaska	F
4	2000-08-12T00:00:00.0000000	Rockwall	Texas	S
5	2000-05-25T00:00:00.0000000	Termo	California	F
6	2000-01-14T00:00:00.0000000	El Centro	California	F
7	2000-07-25T00:00:00.0000000	Stallion Springs	California	F
8	2000-03-04T00:00:00.0000000	Westwater	Utah	F
9	2000-04-10T00:00:00.0000000	Valdese	North Carolina	S
10	2000-03-23T00:00:00.0000000	North Muskegon	Michigan	C
11	2000-09-24T00:00:00.0000000	Ward Ridge	Florida	S

Using the visual query builder

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

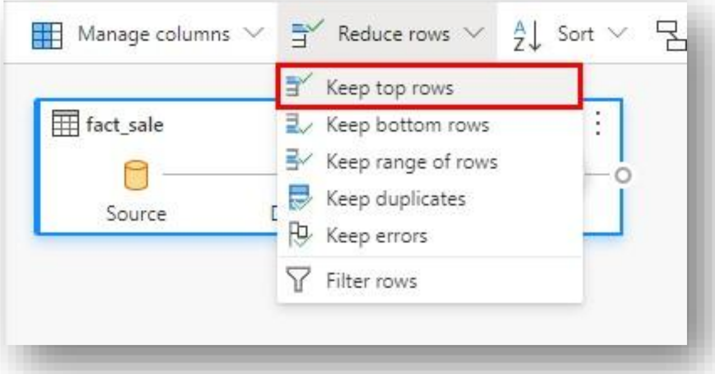


2. Drag the **fact_sale** table from the explorer to the query design pane.

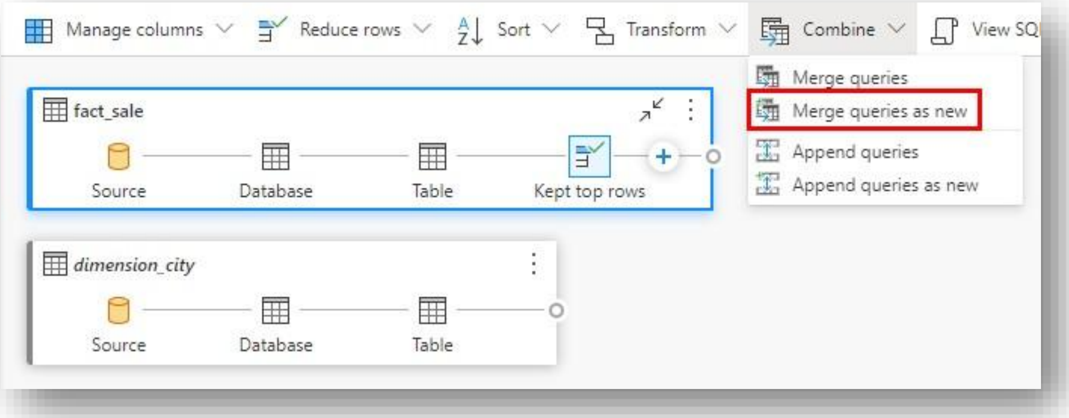


To build a query, drag your tables here.

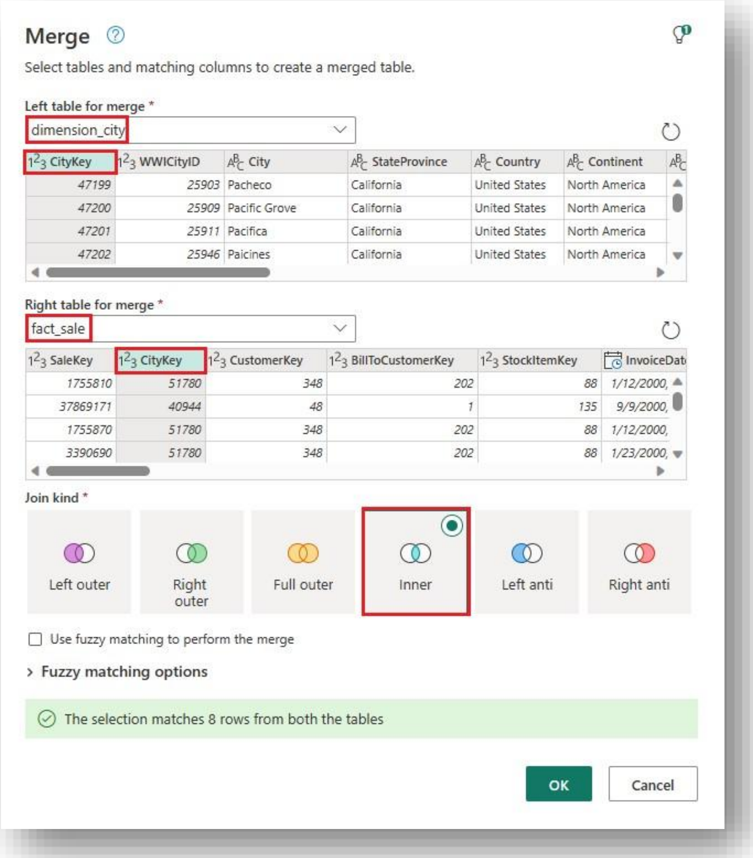
3. Limit the dataset size by selecting **Reduce rows > Keep top rows** from the transformations ribbon.



4. In the **Keep top rows** dialog enter **10,000**.
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. **Left table for merge:** dimension_city
    b. **Right table for merge:** fact_sale
    c. Select the **CityKey** field in the **dimension_city** table by clicking on the column name in the header row to indicate the join column.
    d. Select the **CityKey** field in the **fact_sale** table by clicking on the column name in the header row to indicate the join column.
    e. **Join kind:** Inner



Merge

Select tables and matching columns to create a merged table.

Left table for merge *

dimension_city

1²3 CityKey	1²3 WWICityID	ABC City	ABC StateProvince	ABC Country	ABC Continent
47199	25903	Pacheco	California	United States	North America
47200	25909	Pacific Grove	California	United States	North America
47201	25911	Pacifica	California	United States	North America
47202	25946	Paicines	California	United States	North America

Right table for merge *

fact_sale

1²3 SaleKey	1²3 CityKey	1²3 CustomerKey	1²3 BillToCustomerKey	1²3 StockItemKey	InvoiceDat
1755810	51780	348	202	88	1/12/2000,
37869171	40944	48	1	135	9/9/2000,
1755870	51780	348	202	88	1/12/2000,
3390690	51780	348	202	88	1/23/2000,

Join kind *

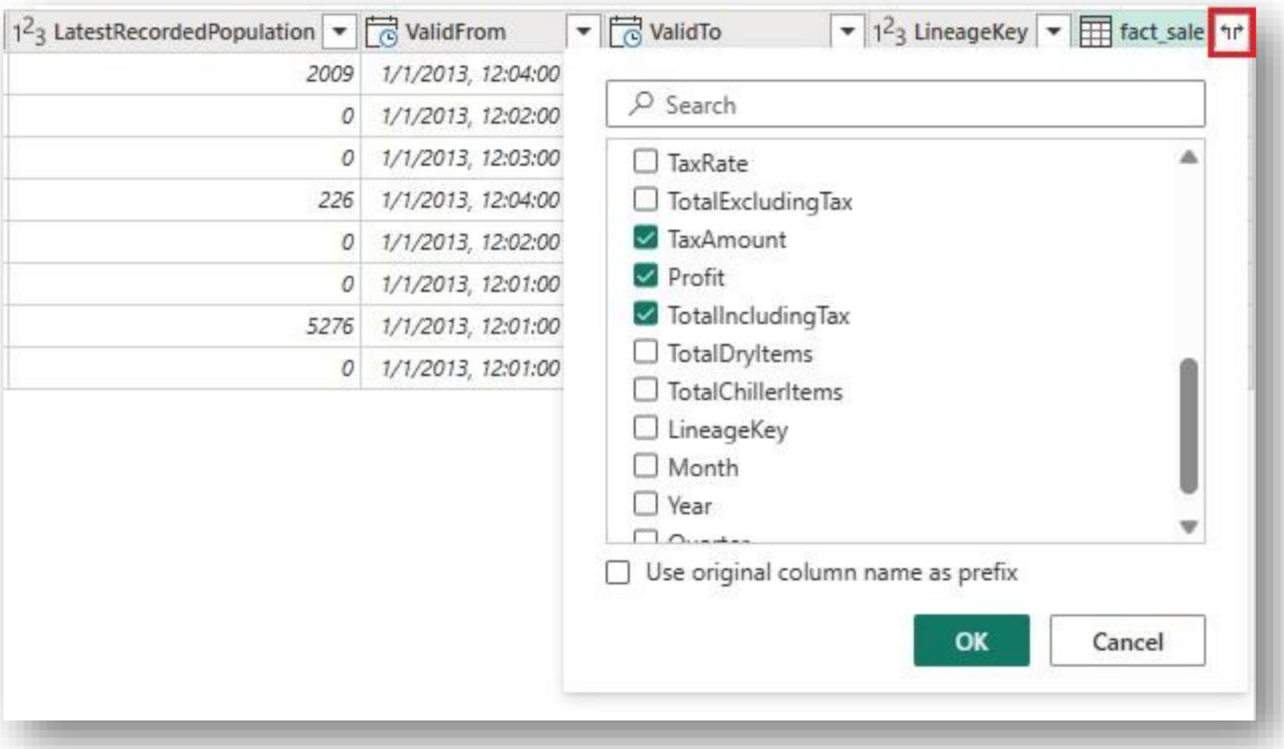
□ Use fuzzy matching to perform the merge

› Fuzzy matching options

The selection matches 8 rows from both the tables

Select **OK**.

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



LatestRecordedPopulation	ValidFrom	ValidTo	LineageKey	fact_sale
2009	1/1/2013, 12:04:00			
0	1/1/2013, 12:02:00			
0	1/1/2013, 12:03:00			
226	1/1/2013, 12:04:00			
0	1/1/2013, 12:02:00			
0	1/1/2013, 12:01:00			
5276	1/1/2013, 12:01:00			
0	1/1/2013, 12:01:00			

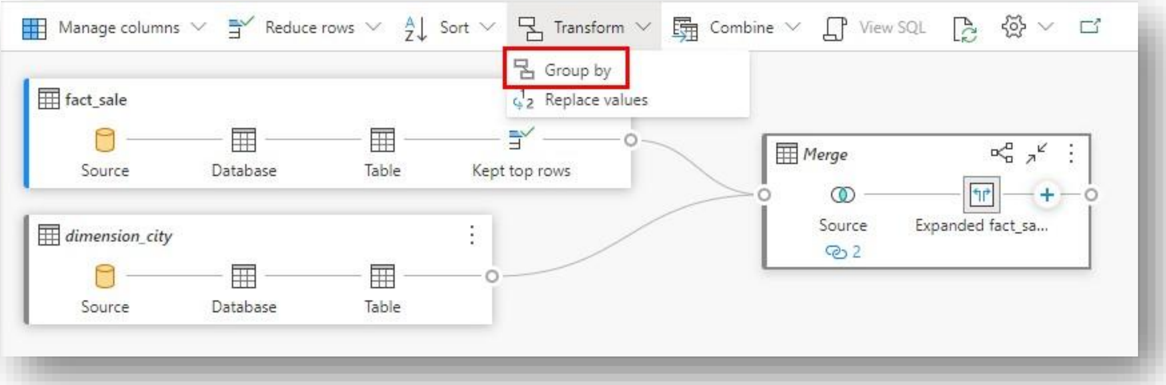
Search

□ TaxRate
□ TotalExcludingTax
☑ TaxAmount
☑ Profit
☑ TotalIncludingTax
□ TotalDryItems
□ TotalChillerItems
□ LineageKey
□ Month
□ Year
□ Quarter

□ Use original column name as prefix

OK Cancel

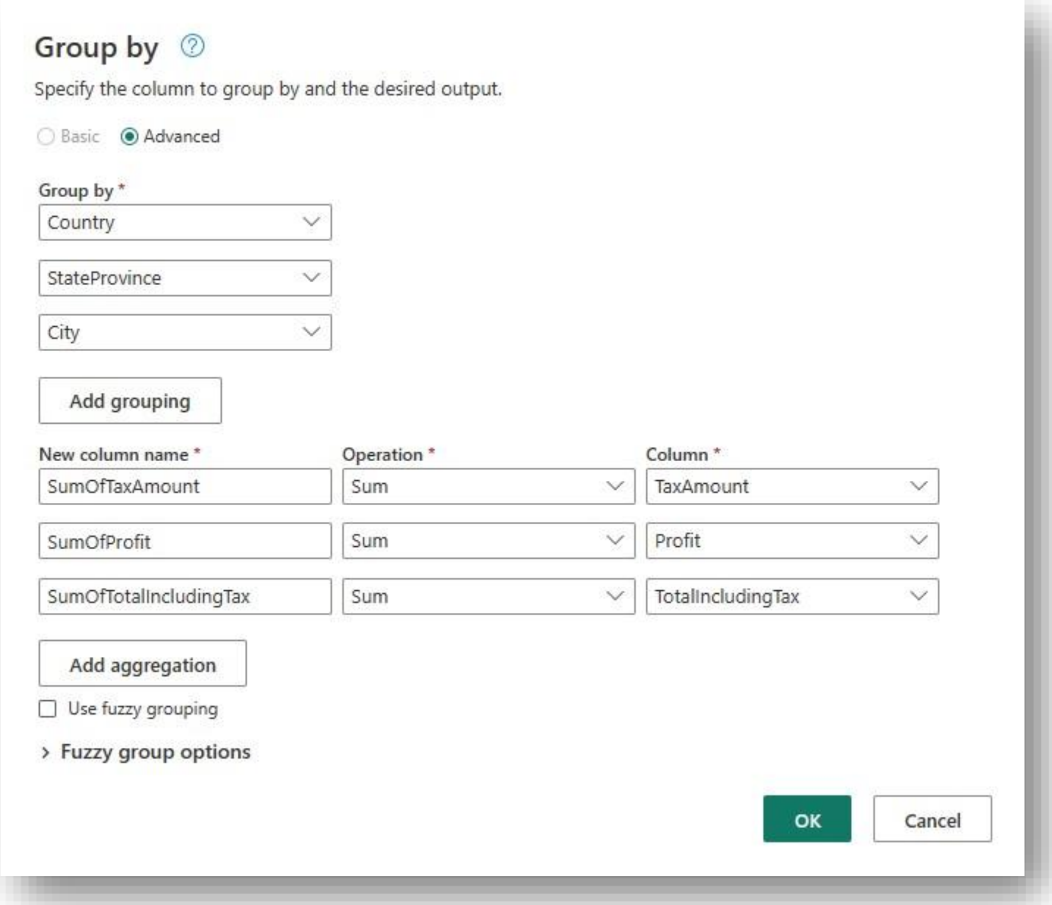
9. Select **OK**.
10. Select **Transform > Group by** from the transformations ribbon.



11. On the **Group by** settings page:
    a. Change to **Advanced**.
    b. **Group by** (if necessary, select **Add grouping** to add additional group by columns)**:** i. Country
        ii. StateProvince
        iii. City
    c. **New column name** (if necessary, select **Add aggregation** to add additional aggregate columns and operations)**:**

i. **SumOfTaxAmount** with **Operation** of **Sum** and **Column** of **TaxAmount**

ii. **SumOfProfit** with **Operation** of **Sum** and **Column** of **Profit** iii.
SumOfTotalIncludingTax with **Operation** of **Sum** and **Column** of **TotalIncludingTax**



Group by

Specify the column to group by and the desired output.

○ Basic ● Advanced

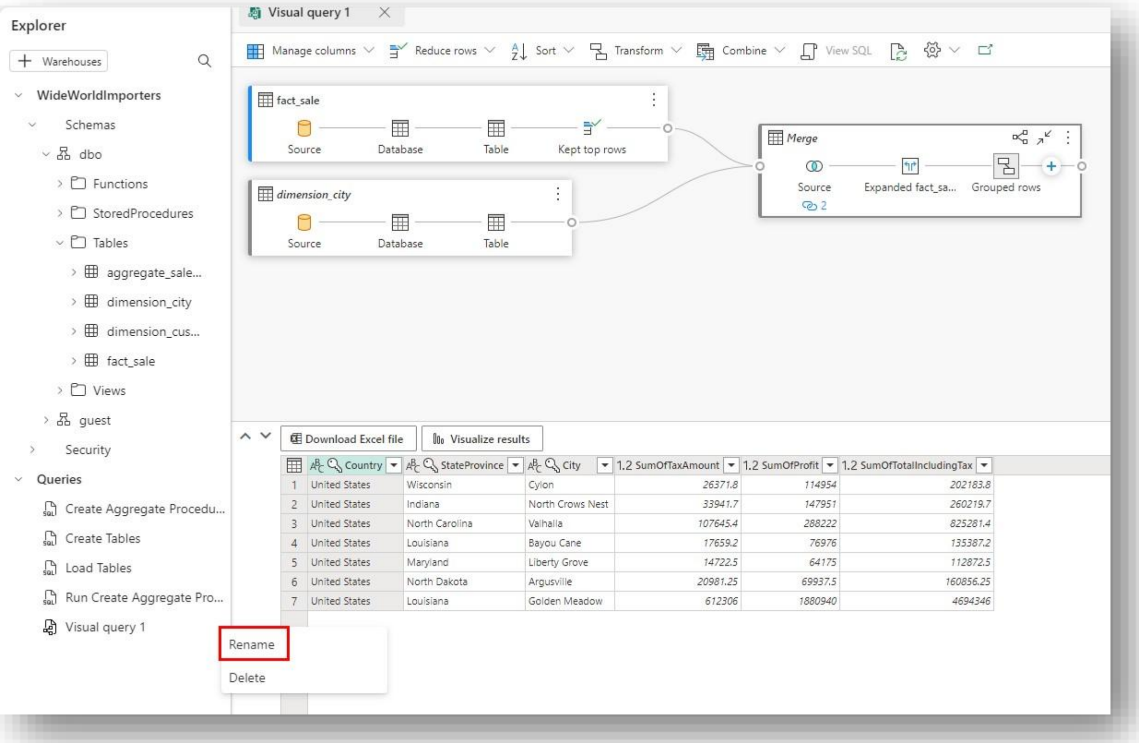
Group by *: Country, StateProvince, City

New column name *	Operation *	Column *
SumOfTaxAmount	Sum	TaxAmount
SumOfProfit	Sum	Profit
SumOfTotalIncludingTax	Sum	TotalIncludingTax

□ Use fuzzy grouping

Select **OK**.

12. Right-click on **Visual query 1** in the explorer and select **Rename**.

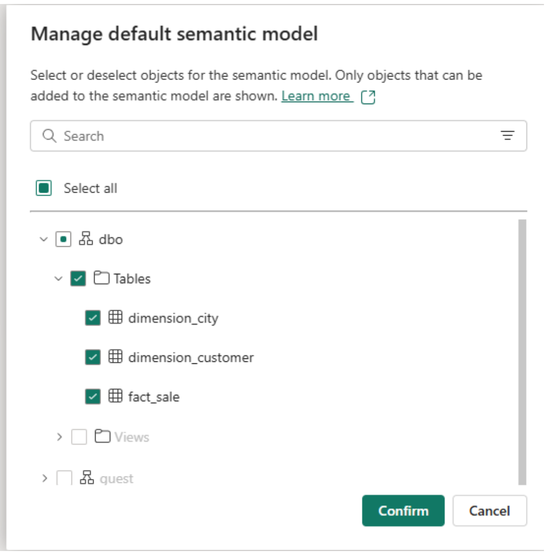


	Country	StateProvince	City	1.2 SumOfTaxAmount	1.2 SumOfProfit	1.2 SumOfTotalIncludingTax
1	United States	Wisconsin	Cylon	26371.8	114954	202183.8
2	United States	Indiana	North Crows Nest	33941.7	147951	260219.7
3	United States	North Carolina	Valhalla	107645.4	288222	825281.4
4	United States	Louisiana	Bayou Cane	17659.2	76976	135387.2
5	United States	Maryland	Liberty Grove	14722.5	64175	112872.5
6	United States	North Dakota	Argusville	20981.25	69937.5	160856.25
7	United States	Louisiana	Golden Meadow	612306	1880940	4694346

13. Type **Sales Summary** to change the name of the query.
14. Press **Enter** on the keyboard or click off anywhere outside the tab to save the change.

Create a Power BI report

1. Navigate to tab **Reporting,** update **Manage default semantic model** to add Tables "dimension_city" & "fact_sale" & click **Confirm**. .



Manage default semantic model

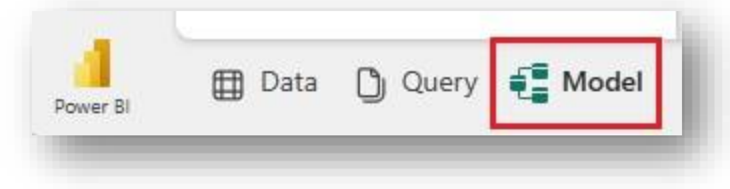
Select or deselect objects for the semantic model. Only objects that can be added to the semantic model are shown. Learn more

Search

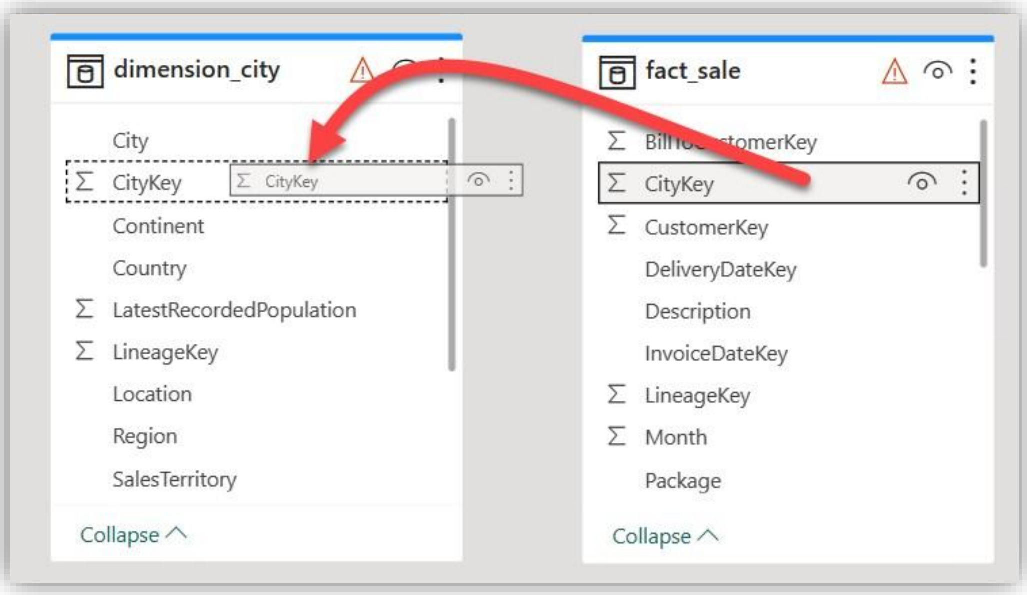
■ Select all

■ dbo
☑ Tables
☑ dimension_city
☑ dimension_customer
☑ fact_sale
□ Views
□ guest

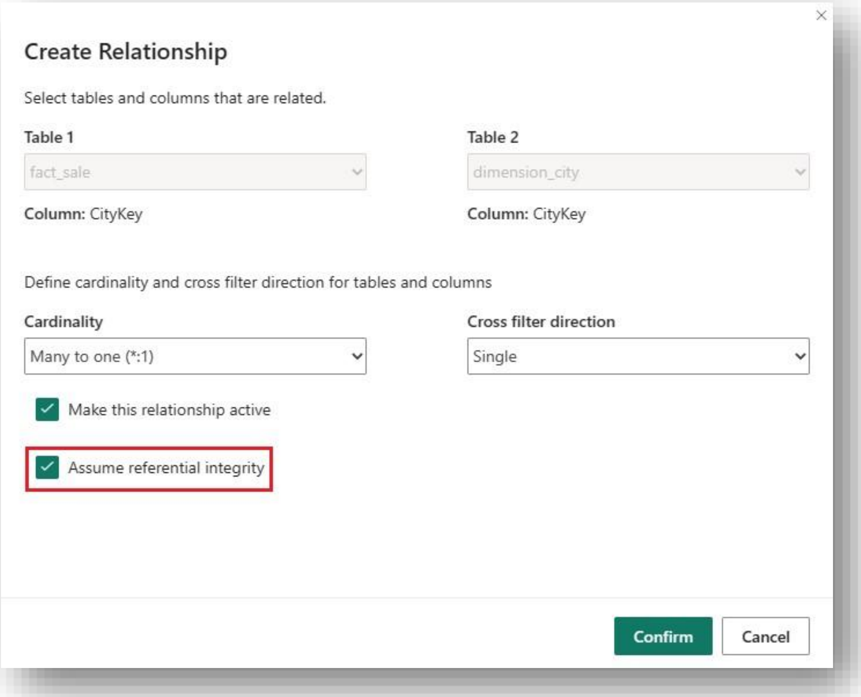
Select the **Model** view from the options in the bottom left corner, just outside the canvas



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



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



Create Relationship

Select tables and columns that are related.

Table 1: fact_sale

Column: CityKey

Table 2: dimension_city

Column: CityKey

Define cardinality and cross filter direction for tables and columns

Cardinality: Many to one (*:1)

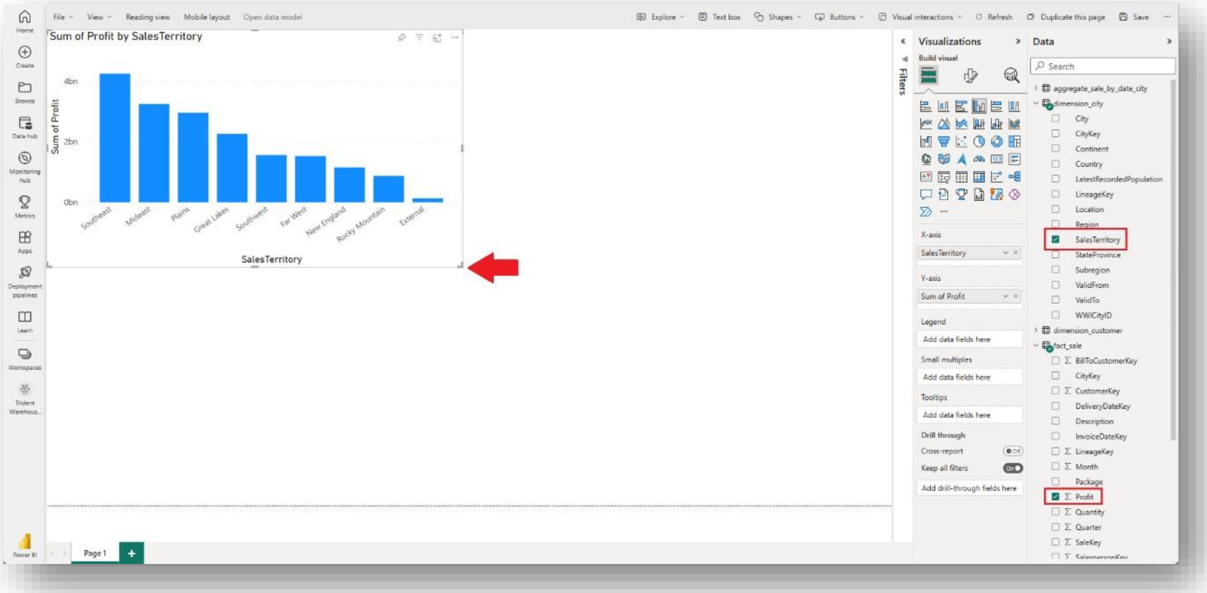
Cross filter direction: Single

☑ Make this relationship active

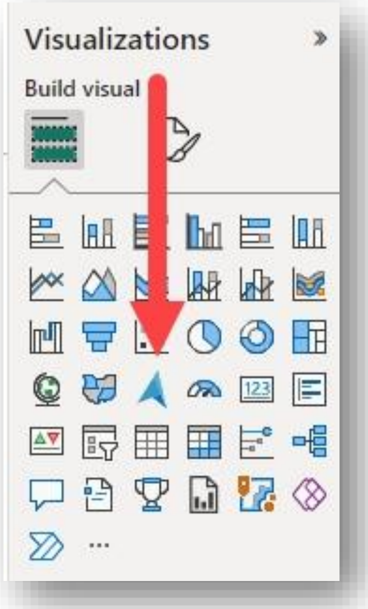
☑ Assume referential integrity

Select **Confirm**.

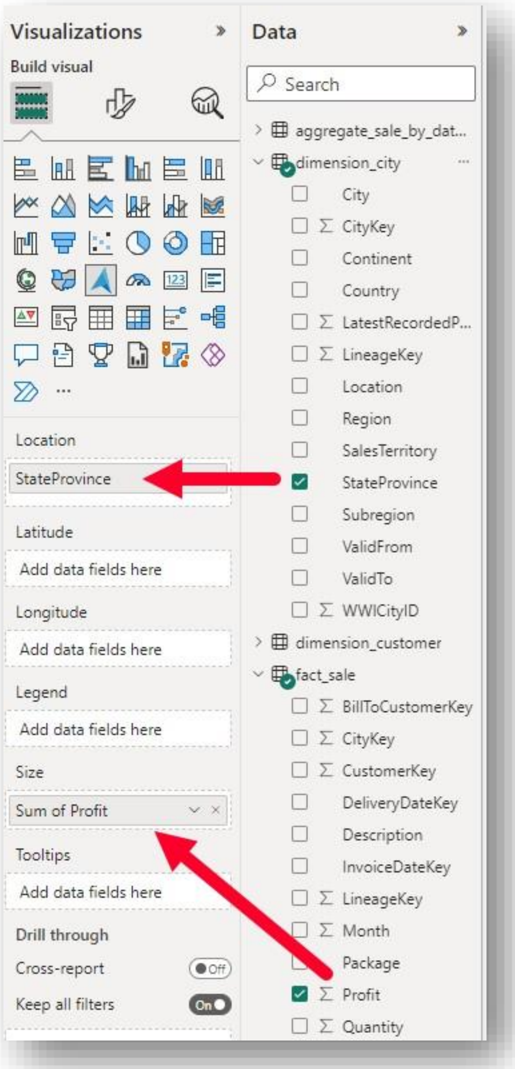
4. From the **Home** tab of the ribbon, select **New report**.
5. Build a column chart visual:
   a. On the **Data** pane, expand **fact_sales** and check the box next to **Profit**. This will create a column chart and add the field to the Y-axis.
   b. On the **Data** pane, expand **dimension_city** and check the box next to **SalesTerritory**. This will add 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.



6. Click anywhere on the blank canvas (or press the Esc key) so the column chart visual is no longer selected.
7. Build a map visual:
   a. On the **Visualizations** pane, select the **Azure Map for Power BI** visual. **Azure Map** visual needs to be enabled by PowerBI admin



   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.



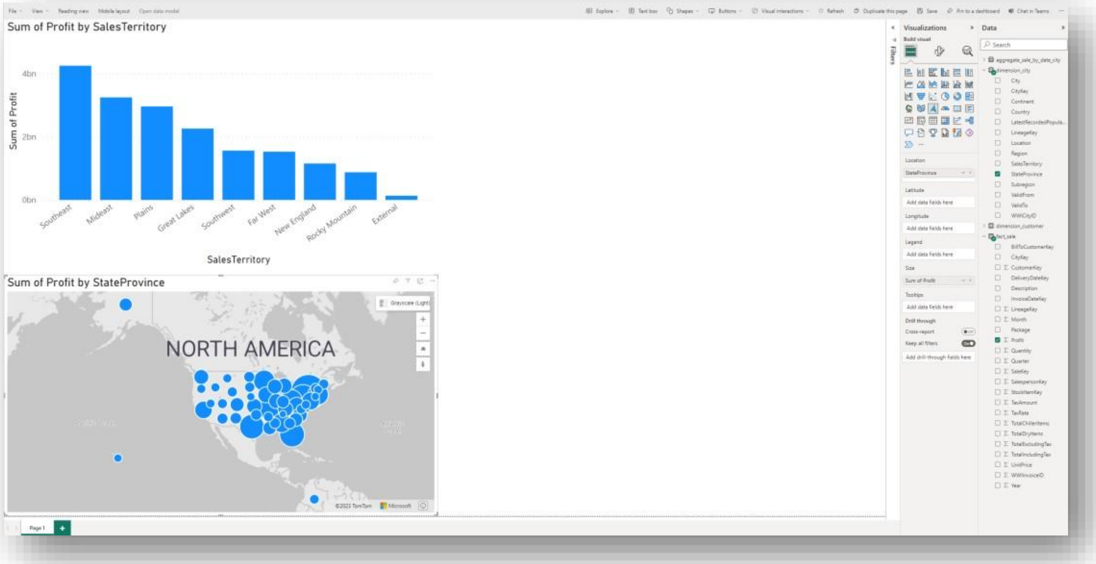
Location: StateProvince

Size: Sum of Profit

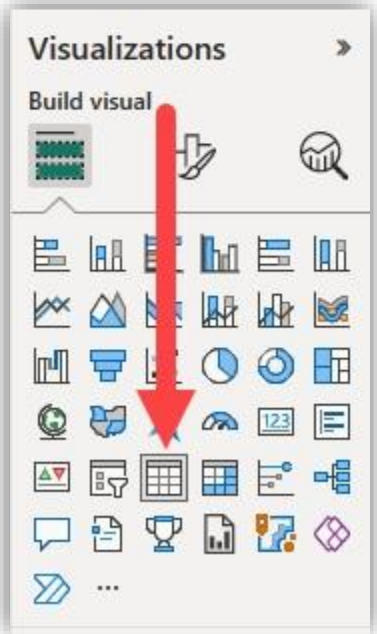
Cross-report: Off

Keep all filters: On

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.

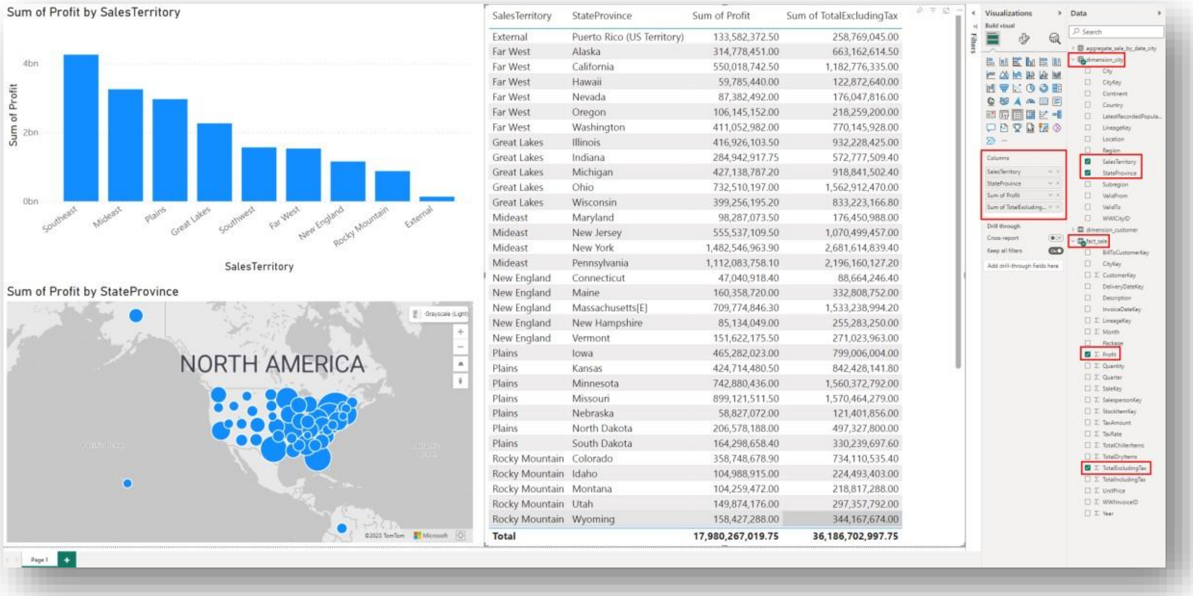


8. Click anywhere on the blank canvas (or press the Esc key) so the map visual is no longer selected.
9. Build a table visual:
    a. On the **Visualizations** pane, select the **Table** visual.



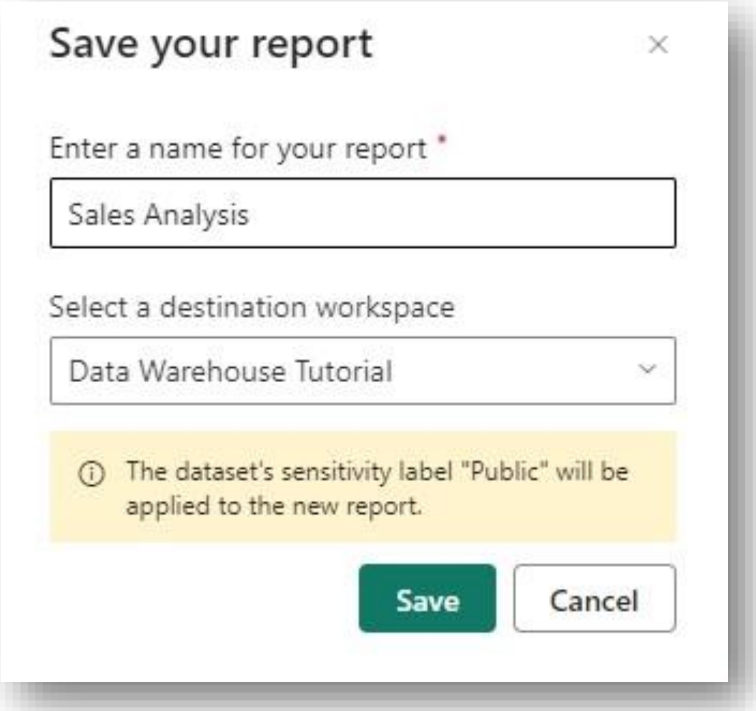
    b. From the **Data** pane, check the box next to **SalesTerritory** on the **dimension_city** table.
    c. From the **Data** pane, check the box next to **StateProvince** on the **dimension_city** table.
    d. From the **Data** pane, check the box next to **Profit** on the **fact_sale** table.
    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.



SalesTerritory	StateProvince	Sum of Profit	Sum of TotalExcludingTax
External	Puerto Rico (US Territory)	133,582,372.50	258,769,045.00
Far West	Alaska	314,778,451.00	663,162,614.50
Far West	California	550,018,742.50	1,182,776,335.00
Far West	Hawaii	59,785,440.00	122,872,640.00
Far West	Nevada	87,382,492.00	176,047,816.00
Far West	Oregon	106,145,152.00	218,259,200.00
Far West	Washington	411,052,982.00	770,145,928.00
Great Lakes	Illinois	416,926,103.50	932,228,425.00
Great Lakes	Indiana	284,942,917.75	572,777,509.40
Great Lakes	Michigan	427,138,787.20	918,841,502.40
Great Lakes	Ohio	732,510,197.00	1,562,912,470.00
Great Lakes	Wisconsin	399,256,195.20	833,223,166.80
Mideast	Maryland	98,287,073.50	176,450,988.00
Mideast	New Jersey	555,537,109.50	1,070,499,457.00
Mideast	New York	1,482,546,963.90	2,681,614,839.40
Mideast	Pennsylvania	1,112,083,758.10	2,196,160,127.20
New England	Connecticut	47,040,918.40	88,664,246.40
New England	Maine	160,358,720.00	332,808,752.00
New England	Massachusetts[E]	709,774,846.30	1,533,238,994.20
New England	New Hampshire	85,134,049.00	255,283,250.00
New England	Vermont	151,622,175.50	271,023,963.00
Plains	Iowa	465,282,023.00	799,006,004.00
Plains	Kansas	424,714,480.50	842,428,141.80
Plains	Minnesota	742,880,436.00	1,560,372,792.00
Plains	Missouri	899,121,511.50	1,570,464,279.00
Plains	Nebraska	58,827,072.00	121,401,856.00
Plains	North Dakota	206,578,188.00	497,327,800.00
Plains	South Dakota	164,298,658.40	330,239,697.60
Rocky Mountain	Colorado	358,748,678.90	734,110,535.40
Rocky Mountain	Idaho	104,988,915.00	224,493,403.00
Rocky Mountain	Montana	104,259,472.00	218,817,288.00
Rocky Mountain	Utah	149,874,176.00	297,357,792.00
Rocky Mountain	Wyoming	158,427,288.00	344,167,674.00
Total		**17,980,267,019.75**	**36,186,702,997.75**

10. From the ribbon, select **File > Save**.
11. Enter the name of your report as **Sales Analysis**.
12. Select **Save**.



Save your report

Enter a name for your report *

Sales Analysis

Select a destination workspace

Data Warehouse Tutorial

The dataset's sensitivity label "Public" will be applied to the new report.

Time Travel in Data Warehouse

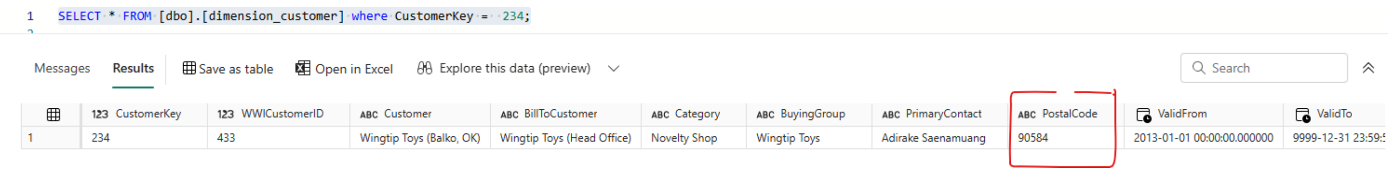
What is time travel?

Time travel in a data warehouse is a low-cost and efficient capability to quickly query prior versions of data. Microsoft Fabric currently allows retrieval with default retention period of 30 days.

The guide below will demonstrate Time travel using the OPTION clause to specify the *FOR TIMESTAMP AS OF* query hint. The column "PostalCode" will be updated twice and then one can travel back in time to query the prior state of data as persisted in past

1) Execute the following query in a new query editor. Current postal code for customer key (234) is 90584

```
SELECT * FROM [dbo].[dimension_customer] where CustomerKey =  234;
```

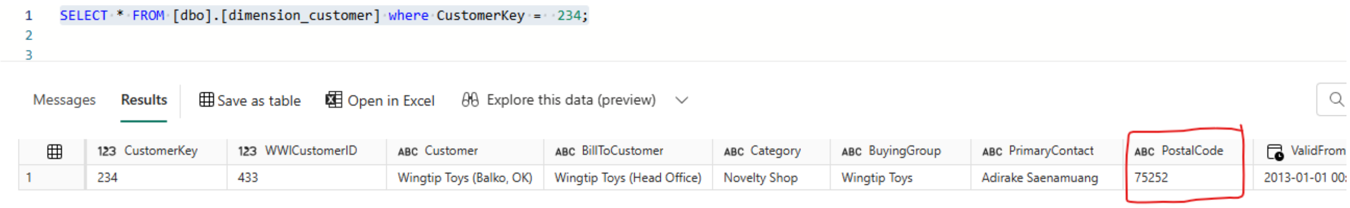


```
SELECT * FROM [dbo].[dimension_customer] where CustomerKey = 234;
```

	CustomerKey	WWICustomerID	Customer	BillToCustomer	Category	BuyingGroup	PrimaryContact	PostalCode	ValidFrom	ValidTo
1	234	433	Wingtip Toys (Balko, OK)	Wingtip Toys (Head Office)	Novelty Shop	Wingtip Toys	Adirake Saenamuang	90584	2013-01-01 00:00:00.000000	9999-12-31 23:59:

2) Update PostalCode for customer key (234) to 75252

```
update [dbo].[dimension_customer]
set PostalCode = 75252
where CustomerKey =  234;
```

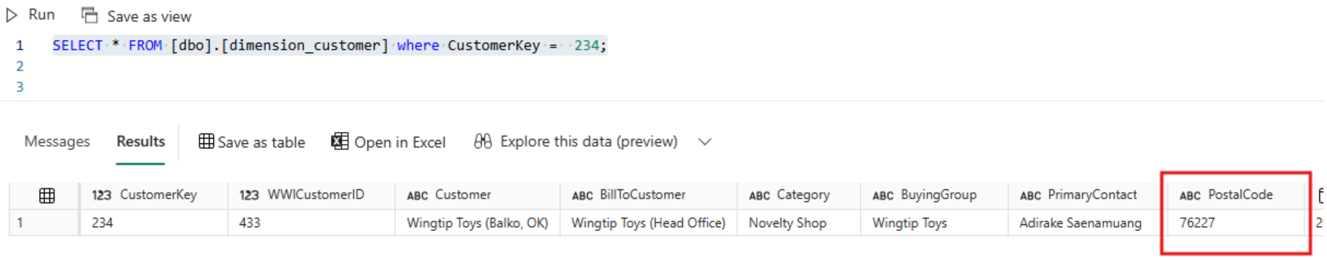


```
SELECT * FROM [dbo].[dimension_customer] where CustomerKey = 234;


```

	CustomerKey	WWICustomerID	Customer	BillToCustomer	Category	BuyingGroup	PrimaryContact	PostalCode	ValidFrom
1	234	433	Wingtip Toys (Balko, OK)	Wingtip Toys (Head Office)	Novelty Shop	Wingtip Toys	Adirake Saenamuang	75252	2013-01-01 00:

3) Update PostalCode for customer key (234) to 76227

```
update [dbo].[dimension_customer]
set PostalCode = 76227
where CustomerKey =  234;
```

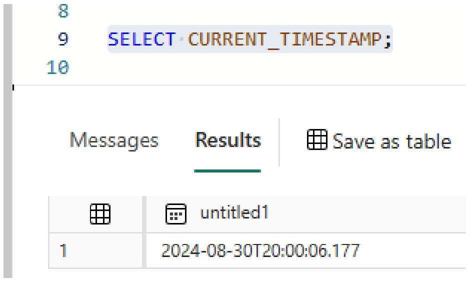


```
SELECT * FROM [dbo].[dimension_customer] where CustomerKey = 234;


```

	CustomerKey	WWICustomerID	Customer	BillToCustomer	Category	BuyingGroup	PrimaryContact	PostalCode
1	234	433	Wingtip Toys (Balko, OK)	Wingtip Toys (Head Office)	Novelty Shop	Wingtip Toys	Adirake Saenamuang	76227

4) Get current time as shown in screenshot below. Copy the timestamp value

```
SELECT CURRENT_TIMESTAMP;
```



```
SELECT CURRENT_TIMESTAMP;
```

	untitled1
1	2024-08-30T20:00:06.177

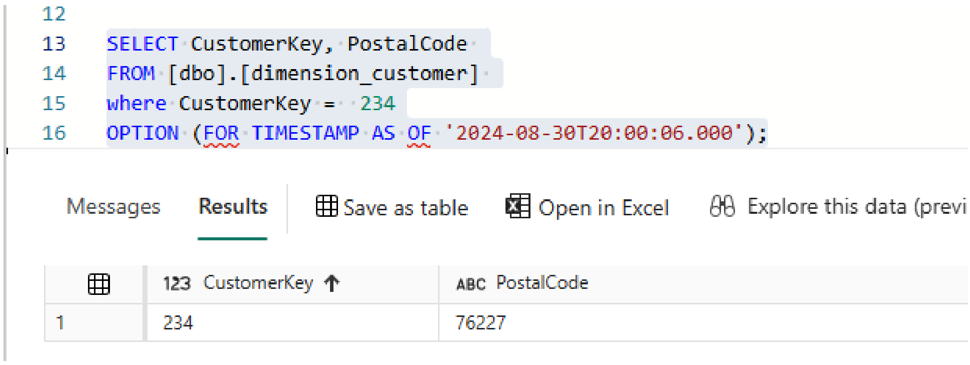
Copy the timestamp value and for simpler querying of prior versions, make the millisecond component to 000 as shown below.

2024-08-30T20:00:06.177 ----------------→ 2024-08-30T20:00:06.**000**

5) Time travel using queries below, and observe the prior values for field "PostalCode"

Latest Status as of 20:00:06

```
SELECT CustomerKey, PostalCode
FROM [dbo].[dimension_customer]
where CustomerKey =  234
OPTION (FOR TIMESTAMP AS OF '2024-08-30T20:00:06.000');
```

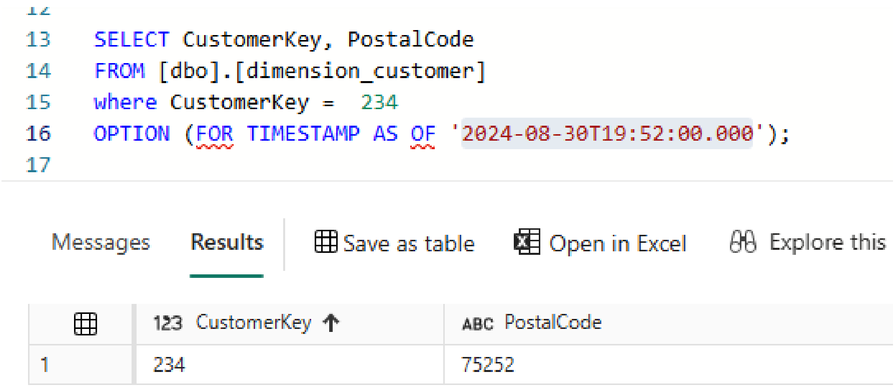


```
SELECT CustomerKey, PostalCode
FROM [dbo].[dimension_customer]
where CustomerKey =  234
OPTION (FOR TIMESTAMP AS OF '2024-08-30T20:00:06.000');
```

	123 CustomerKey ↑	ABC PostalCode
1	234	76227

Prior Status as of 19:52:00

```
SELECT CustomerKey, PostalCode
FROM [dbo].[dimension_customer]
where CustomerKey =  234
OPTION (FOR TIMESTAMP AS OF '2024-08-30T19:52:00.000');
```

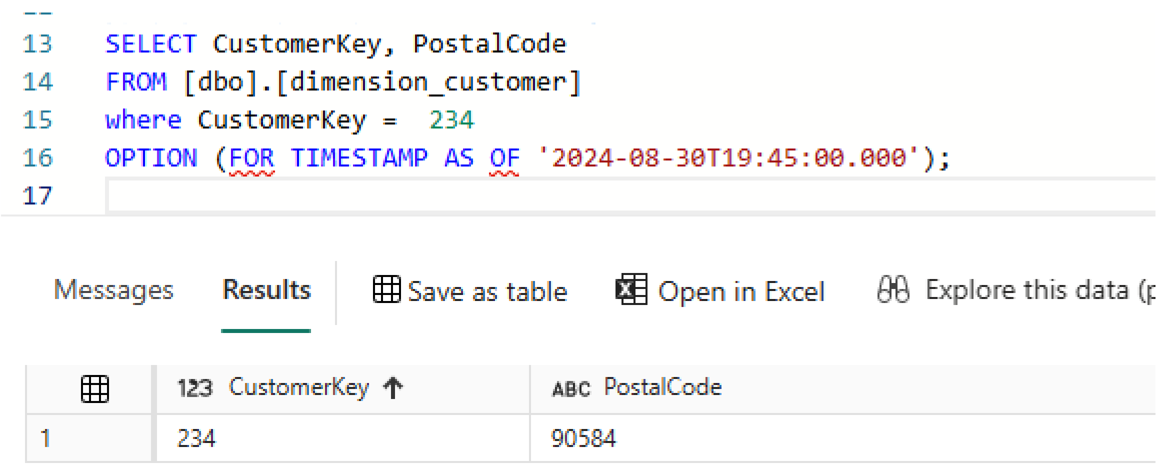


```
SELECT CustomerKey, PostalCode
FROM [dbo].[dimension_customer]
where CustomerKey =  234
OPTION (FOR TIMESTAMP AS OF '2024-08-30T19:52:00.000');
```

	123 CustomerKey ↑	ABC PostalCode
1	234	75252

Prior Status as of 19:45:00

SELECT CustomerKey, PostalCode
FROM [dbo].[dimension_customer]
where CustomerKey = 234
OPTION (FOR TIMESTAMP AS OF '2024-08-30T19:45:00.000');



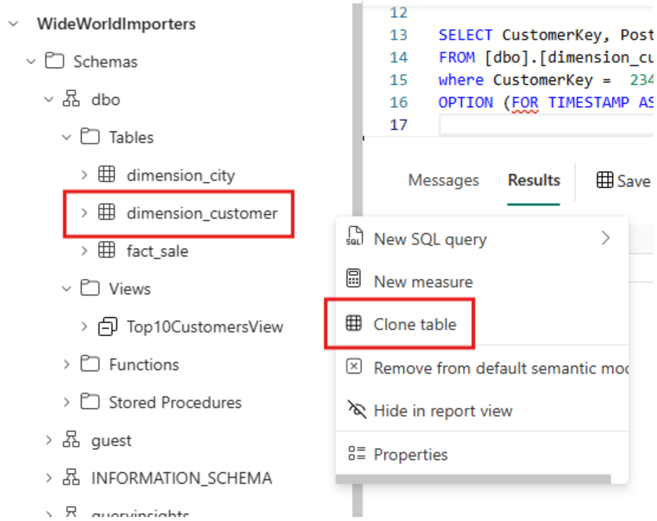
```
SELECT CustomerKey, PostalCode
FROM [dbo].[dimension_customer]
where CustomerKey =  234
OPTION (FOR TIMESTAMP AS OF '2024-08-30T19:45:00.000');
```

	123 CustomerKey ↑	ABC PostalCode
1	234	90584

Clone Table in Data Warehouse

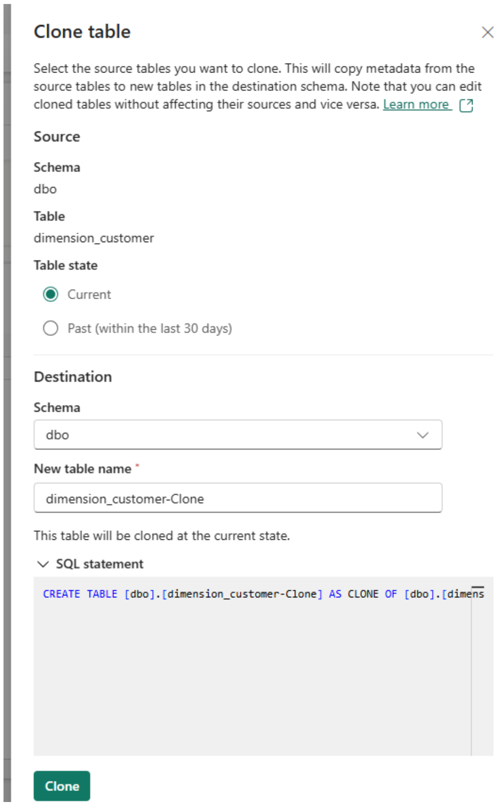
Microsoft Fabric offers the capability to create near-instantaneous zero-copy clones with minimal storage costs. A zero-copy clone creates a replica of the table by copying the metadata, while still referencing the same data files in OneLake. The metadata is copied while the underlying data of the table stored as parquet files is not copied. The creation of a clone is similar to creating a table within a Warehouse in Microsoft Fabric.

1) Clone the “dimension_customer” table. When you select the table, and select on more options, you get the Clone table menu. This menu is also available via Table tools in the ribbon.



```
SELECT CustomerKey, Post
FROM [dbo].[dimension_cu
where CustomerKey =  234
OPTION (FOR TIMESTAMP AS
```

2) On clone table pane, you can see the source table schema and name is already populated. The table state as current, creates clone of the source table as of its current state. You can also clone table from a past point in time. You can choose destination schema and edit pre-populated destination table name. You can also see the generated T-SQL statement when you expand SQL statement section. When you select the Clone button, a clone of the table is generated and you can see it in Explorer.



Clone table

Select the source tables you want to clone. This will copy metadata from the source tables to new tables in the destination schema. Note that you can edit cloned tables without affecting their sources and vice versa. Learn more

Source

Schema
dbo

Table
dimension_customer

Table state

● Current

○ Past (within the last 30 days)

Destination

Schema
dbo

New table name *
dimension_customer-Clone

This table will be cloned at the current state.

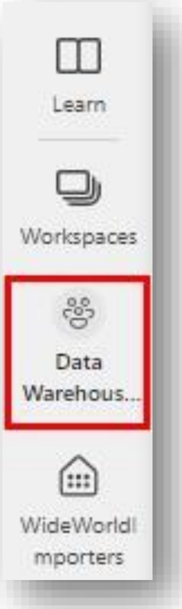
SQL statement

```
CREATE TABLE [dbo].[dimension_customer-Clone] AS CLONE OF [dbo].[dimens
```

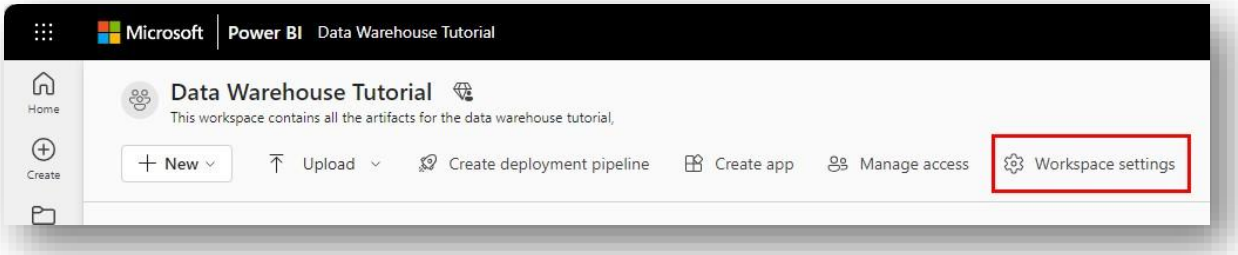
Module 4: Clean up resources

You can delete individual reports, pipelines, warehouses, and other items or remove the entire workspace.

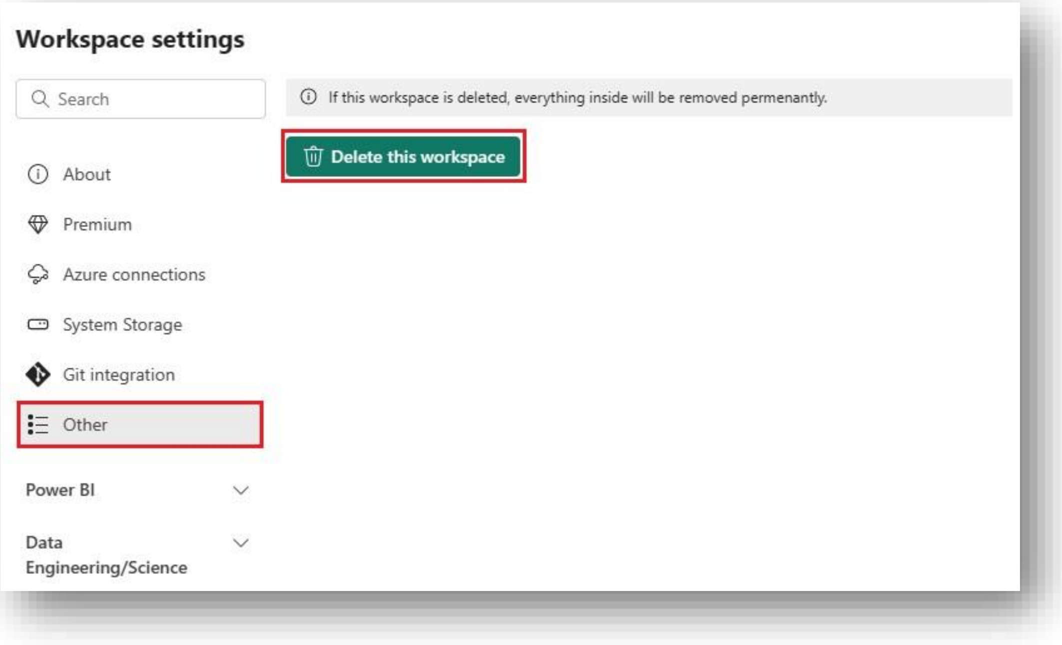
1. Select **Data Warehouse Tutorial** in the left-hand navigation menu to return to the workspace artifact view.



2. Below the workspace name and description at the top of the workspace header, select **Workspace settings**.



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



Workspace settings

If this workspace is deleted, everything inside will be removed permenantly.

4. Select **Delete** on the warning.