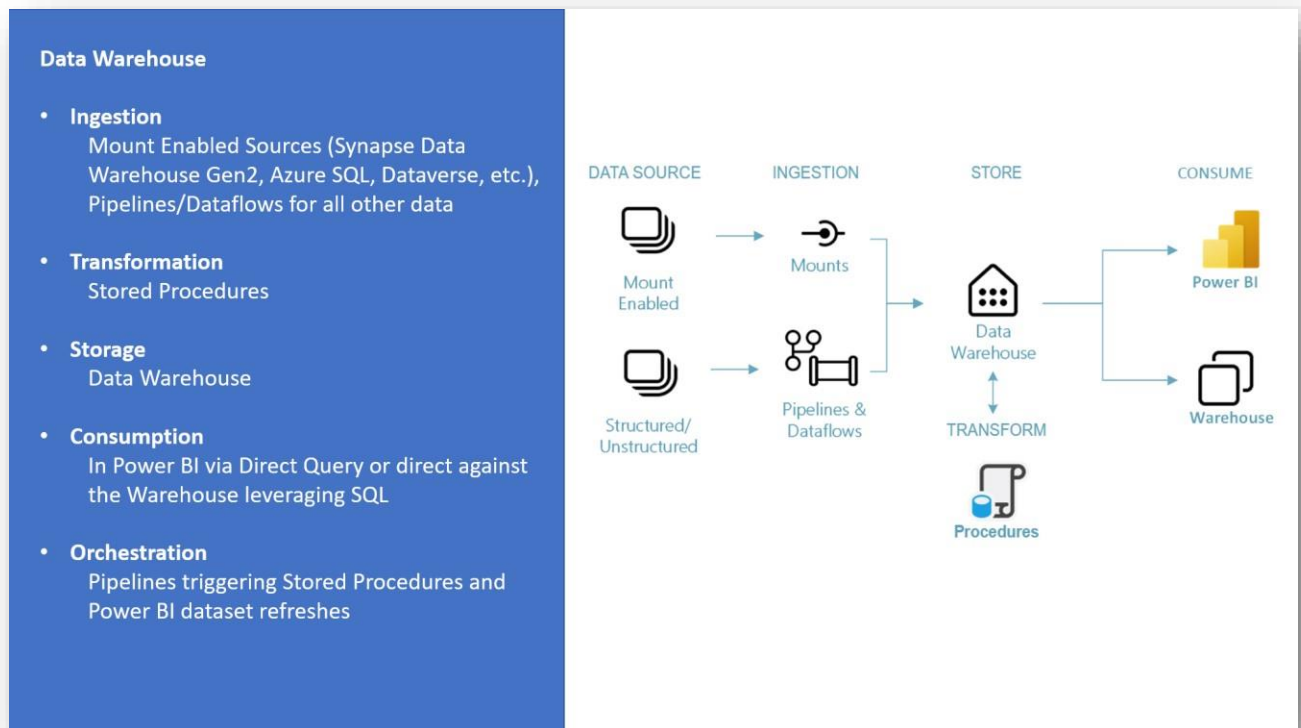


# Tutorial

## Data Warehouse

Published: June 2023



Contents

Introduction.....3

Module 1: Build your first data warehouse.....6

    Create a data warehouse .....6

    Data ingestion.....9

    Building a report .....16

Module 2: Extending the solution .....21

    Creating tables in the data warehouse .....21

    Loading data using T-SQL.....24

    Data transformation using a stored procedure .....28

    Using the visual query builder .....31

    Create a Power BI report.....36

# 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.

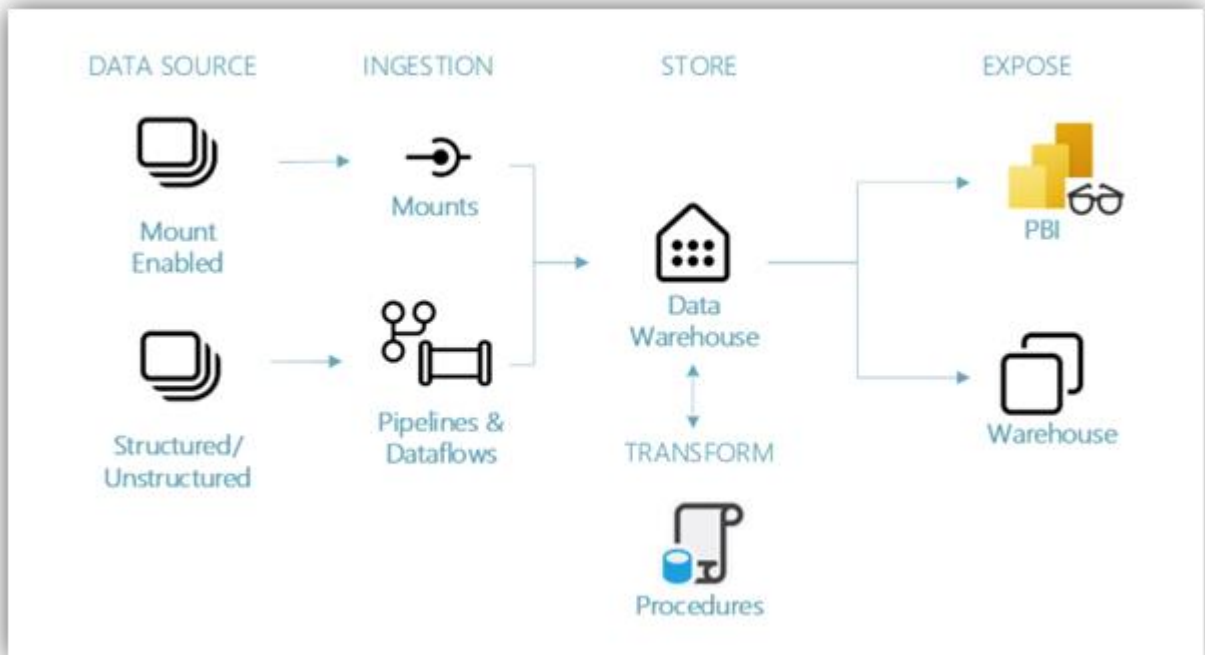
The tutorials are not intended to be a reference architecture, an exhaustive list of features and functionality, or a recommendation of specific best practices.

## 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:
  - Enable Fabric in your tenant
  - Create a Fabric workspace
  - Quickly create a data warehouse
    - Ingest data from source to the data warehouse dimensional model
    - Transform the data to create aggregated datasets using T-SQL
    - Perform orchestration, data ingestion, and data transformation with pipelines
    - Query the data warehouse using T-SQL and a visual query editor
    - 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 nocode 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

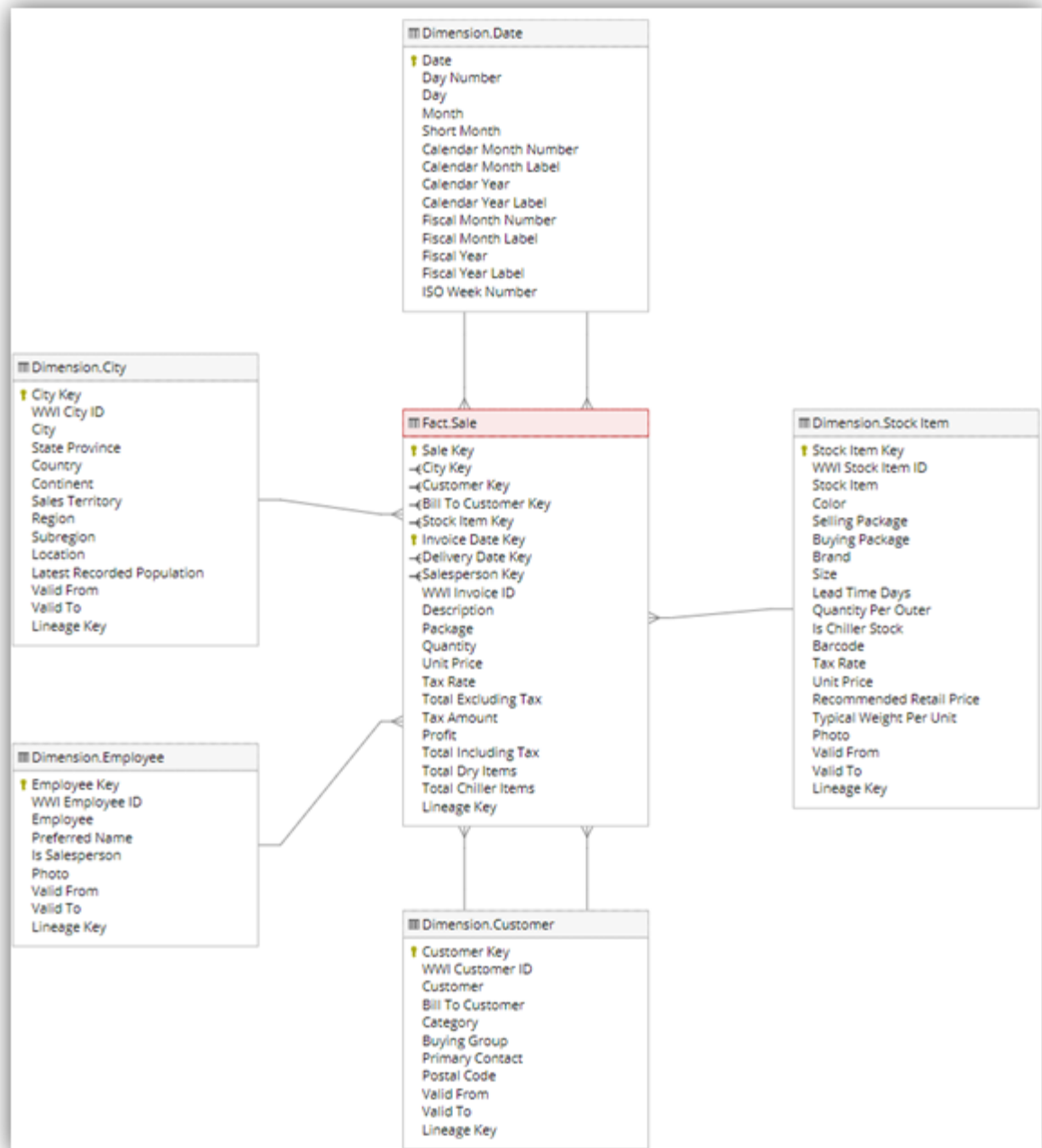
For sample data, we are going to use [Wide World Importers \(WWI\) sample database](#). For our data warehouse end-to-end scenario, we have generated sufficient data for a sneak peek into the scale and performance capabilities of the Fabric platform.

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:

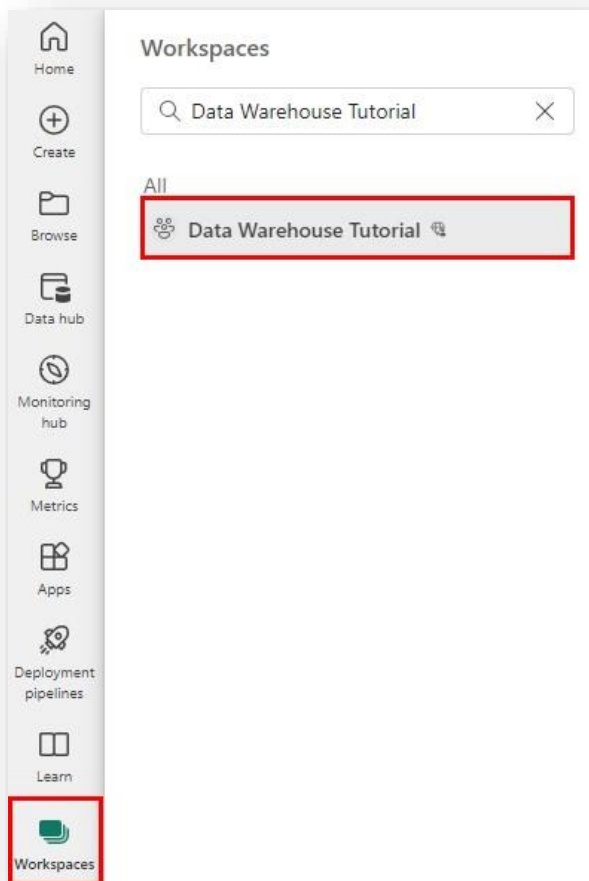


## Module 1: Build your first data warehouse

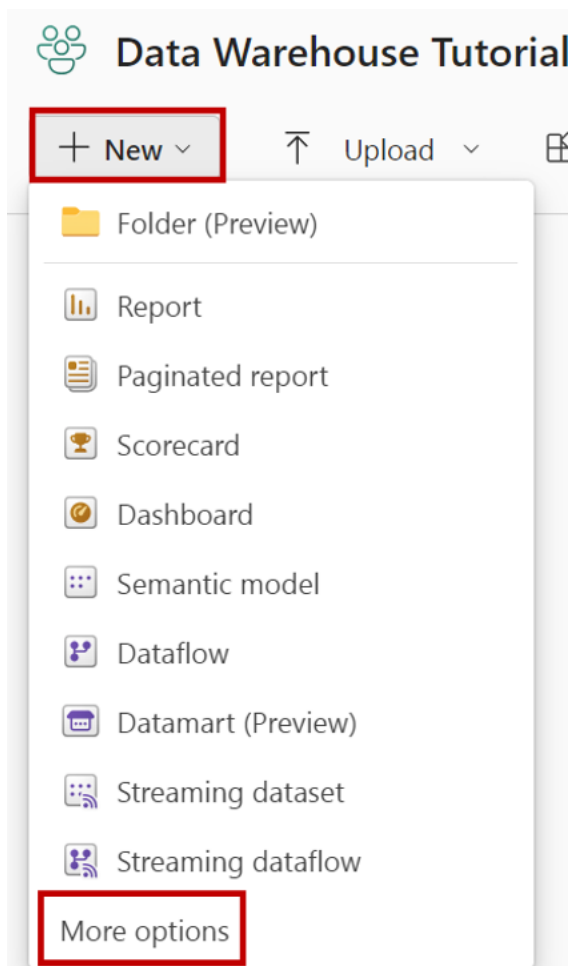
The intent of this module is to quickly build end to end journey of building a data warehouse, ingesting data for a table and then using the data warehouse for creating a report.

### Create a data warehouse

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



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



5. In the **Data warehouse** section, select **Warehouse**.

## Data Warehouse

Provide strategic insights from multiple sources into your entire business. [Learn more](#) 

### Warehouse



Provide strategic insights from multiple sources into your entire business.

6. On the **New warehouse** dialog, enter **WideWorldImporters** as the name.
7. Select **Create**.

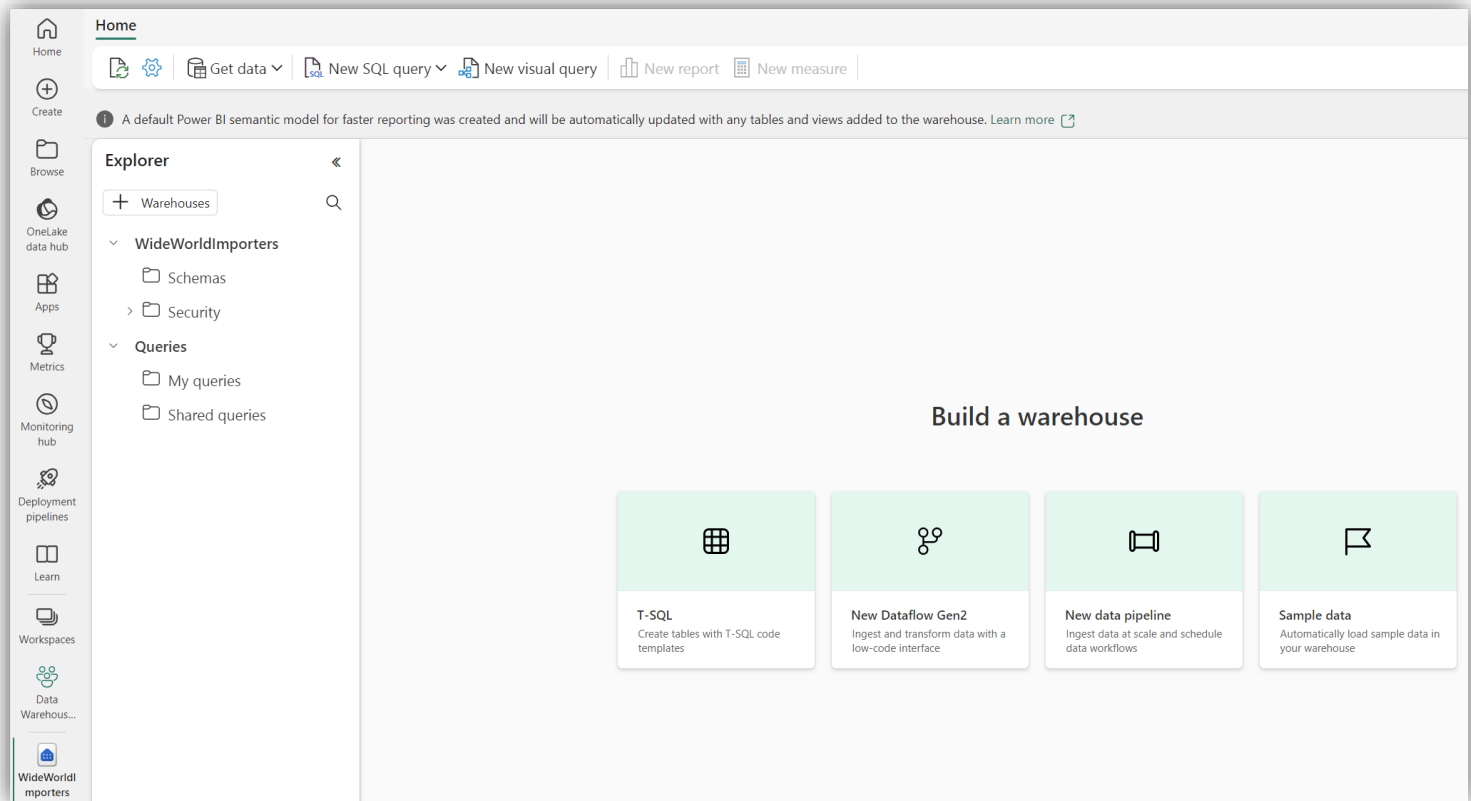
### New warehouse

Name

CreateCancel

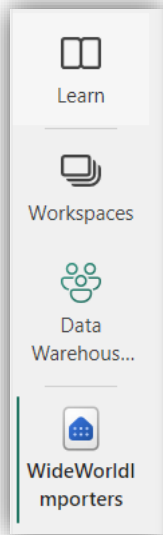
When provisioning is complete the **Build a warehouse** landing page will be shown.



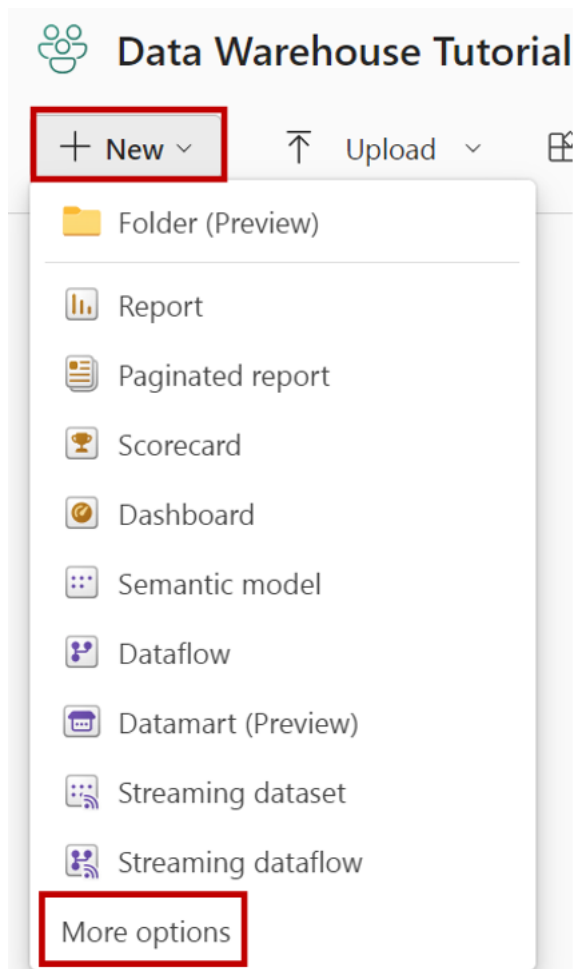


## Data ingestion

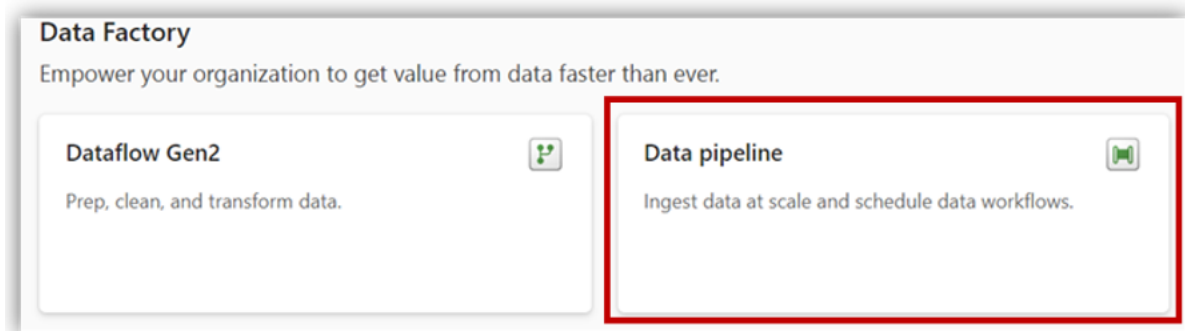
1. Select **Data Warehouse Tutorial** in the left-hand navigation menu to return to the workspace artifact view.



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



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



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

## New pipeline

×

Name


Load Customer Data

Create

Cancel

11. Select **Create**.
12. Select **Add pipeline activity** from the **Start building your data pipeline** landing page.


## Start building your data pipeline



Add pipeline activity

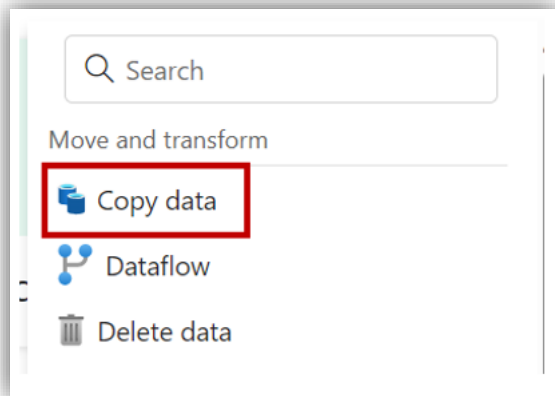


Copy data



Choose a task to start

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



14. If necessary, select the newly created Copy data activity from the design canvas and follow the steps below to configure it.
15. On the **General** page, enter **CD Load dimension\_customer** as the **Name**.

A screenshot of the 'General' configuration page for a data activity. The page has a tabbed interface with 'General', 'Source', 'Destination', 'Mapping', and 'Settings'. The 'General' tab is selected. There are two input fields: 'Name' and 'Description'. The 'Name' field contains the text 'CD Load dimension\_customer' and is highlighted with a green rectangular box. The 'Description' field is empty.

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

A screenshot of the 'Source' configuration page for a data activity. The page has a tabbed interface with 'General', 'Source', 'Destination', 'Mapping', and 'Settings'. The 'Source' tab is selected and highlighted with a red rectangular box. There are two main sections: 'Data store type' and 'Connection'. The 'Data store type' section has three radio buttons: 'Workspace', 'External', and 'Sample dataset'. The 'External' radio button is selected and highlighted with a red rectangular box. The 'Connection' section has a dropdown menu with 'Select...' and a 'Refresh' button. To the right of the 'Refresh' button is a '+ New' button, which is highlighted with a red rectangular box.

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

## New connection



All **Azure** Database File Generic protocol NoSQL Services and apps

Q Search

Azure Blob Storage  
Azure

Azure Cosmos DB for NoSQL  
Azure

Azure Data Explorer  
Azure

Azure Data Lake Storage Gen1  
Azure

Azure Data Lake Storage Gen2  
Azure

Azure Database for PostgreSQL  
Azure

Azure SQL Database  
Azure

Azure SQL Database Managed Instance  
Azure

Azure Synapse Analytics  
Azure

Azure Table Storage  
Azure

19. Select **Continue**.

20. On the **Connection settings** page, configure the settings as follows:

1. Enter **https://azuresynapsestorage.blob.core.windows.net/sampled** in the **Account name or URL**.
2. In the **Connection credentials** section, select **Create new connection** in the dropdown for the **Connection**.
3. Enter **Wide World Importers Public Sample** for the **Connection name**.
4. Set the **Authentication kind** to **Anonymous**.

### New connection

Azure Blob Storage  
[Learn more](#)

### Connection settings

Account name or URL \*

https://azuresynapsestorage.blob.core.windows.net/sampl...

### Connection credentials

Connection

Create new connection

Connection name

Wide World Importers Public Sample

Authentication kind

Anonymous

21. Click **Create**.

22. Change the remaining settings on the **Source** page of the copy activity as follows:

1. **File path – Container:** sampled
2. **File path – Directory:** WideWorldImportersDW/tables
3. **File path – File name:** dimension\_customer.parquet
4. **File format:** Parquet

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

General **Source** Destination<sup>1</sup> Mapping Settings

Data store type ☐ Workspace ☒ External ☐ Sample dataset

Connection Wide World Importers Public Sample Refresh Test connection Edit New

File path type ☒ File path ☐ Prefix ☐ Wildcard file path ☐ List of files ⓘ

File path \* sampledata / WideWorldImporters... / dimension\_customer... Browse Preview data

Recursively ⓘ ☒

File format \* ⓘ Parquet Settings

> Advanced

24. On the **Destination** page, select **Workspace** for the **Data store type**.
25. Select **Warehouse** for the **Workspace data store type**.
26. In the **Warehouse** drop down, select **WideWorldImporters** from the list.
27. Next to the **Table option** setting, select **Auto create table**
28. In the first box (schema name) next to the **Table** setting, enter **dbo**.
29. In the second box(table name) next to the **Table** setting, enter **dimension\_customer**.

General Source **Destination** Mapping Settings

Data store type ☒ Workspace ☐ External

Workspace data store type Warehouse

Warehouse WideWorldImporters Refresh Open


Table option ☐ Use existing ☒ Auto create table ⓘ

Table dbo . dimension

> Advanced

30. From the ribbon, select **Run**.
31. Select **Save and run** from the dialog box. The pipeline to load the dimension\_customer table with start.
32. Monitor the copy activity's progress on the **Output** page and wait for it to complete.

Copy data

 CD Load dimension\_custo...

ParametersVariablesSettingsOutput

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	<span>✓</span> Succeeded	1/30/2024, 10:33:44 PM	17s

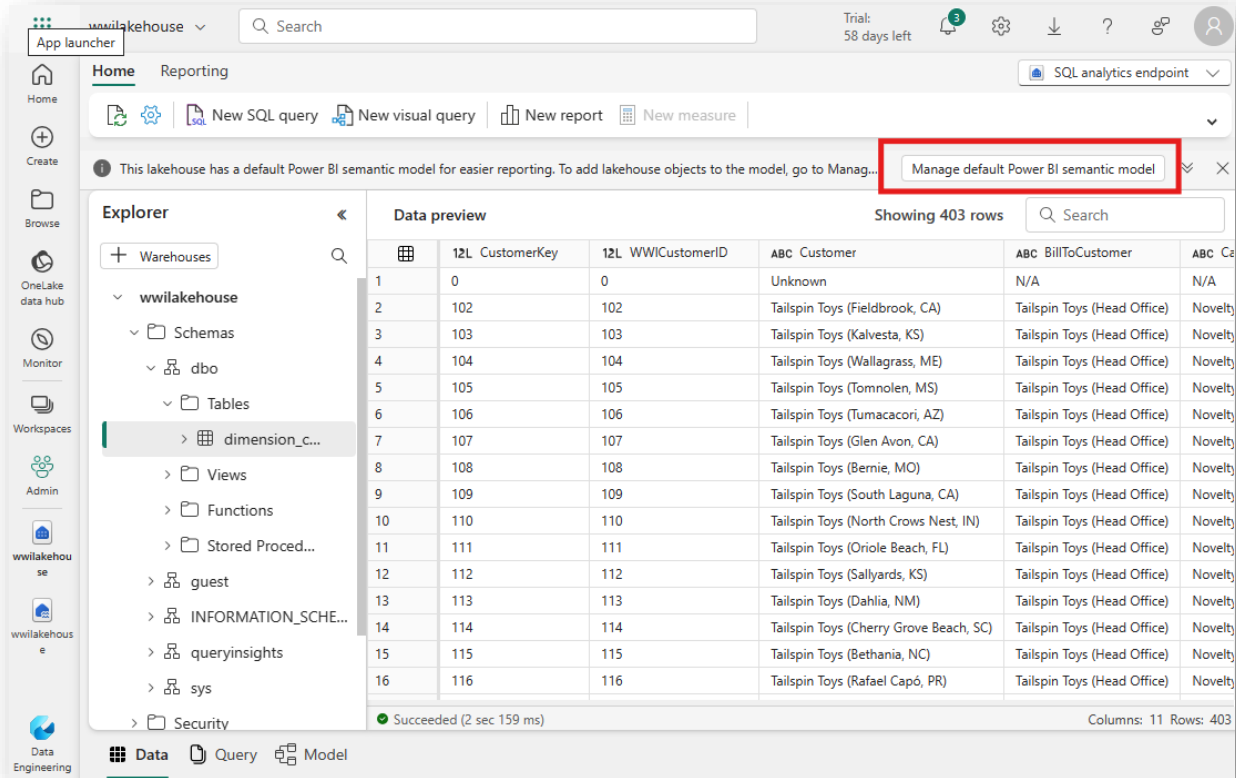
## Add table to the semantic model

In Microsoft Fabric, Power BI semantic models are a logical description of an analytical domain, with metrics, business friendly terminology, and representation, to enable deeper analysis. This semantic model is typically a star schema with facts that represent a domain, and dimensions that allow you to analyze, or slice and dice the domain to drill down, filter, and calculate different analyses.

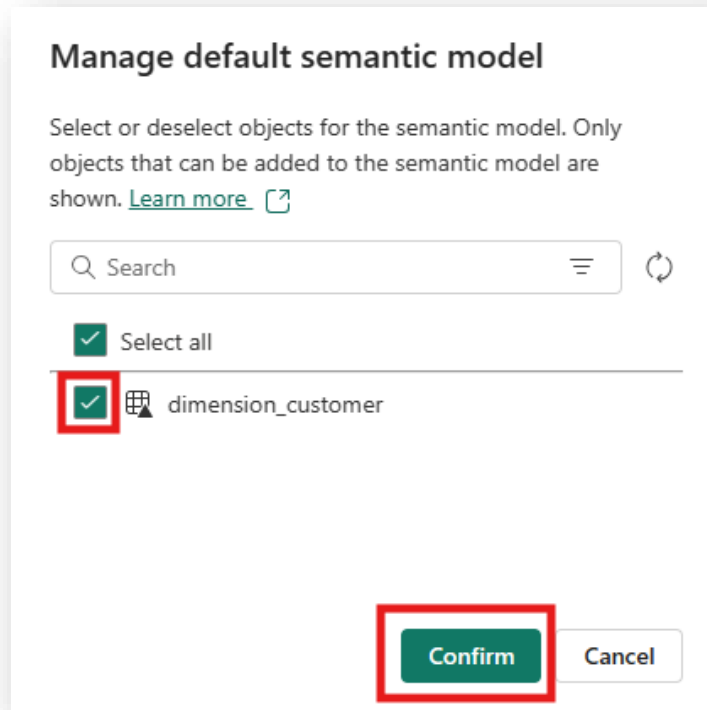
Now that you’ve ingested data into a lakehouse table, we’ll add that table to the **default semantic model** for the lakehouse.

1. From the SQL analytics Data view of the dimension\_customer table, click **Manage default Power BI semantic model** button on the upper right.



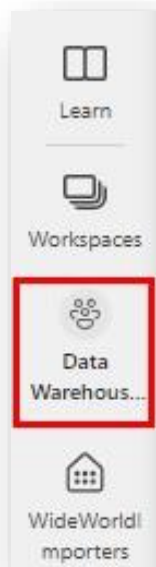


2. The **Manage default semantic model** dialog will appear. Select the `dimension_customer` table and click **Confirm**.

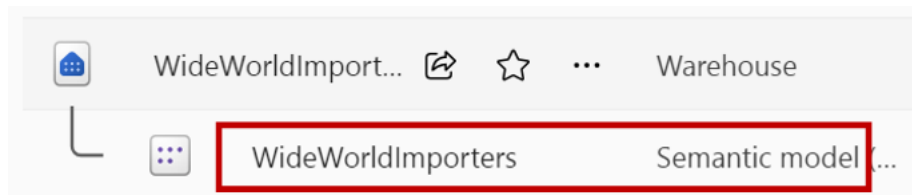


## Building a report

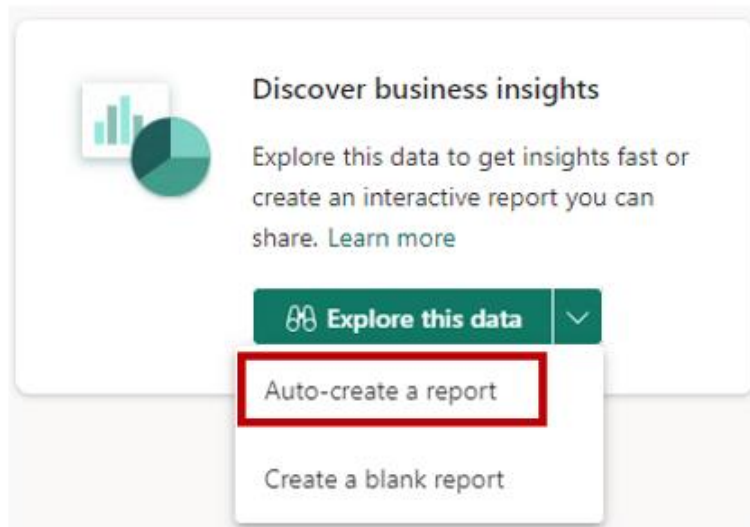
1. Select **Data Warehouse Tutorial** in the left-hand navigation menu to return to the workspace artifact view.



2. From the artifact list, select **WideWorldImporters** with the type of **Semantic Model (default)**.



3. 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.



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

## Quick summary

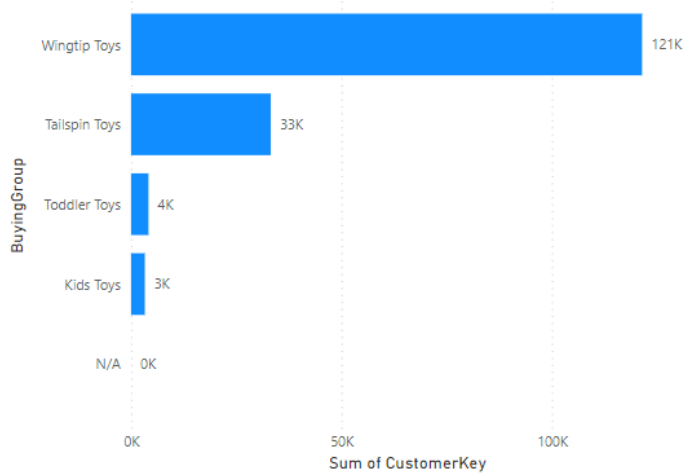
dimension

162006  
Sum of CustomerKey

242004  
Sum of WWICustomerID

1608  
Sum of LineageKey

Sum of CustomerKey by BuyingGroup

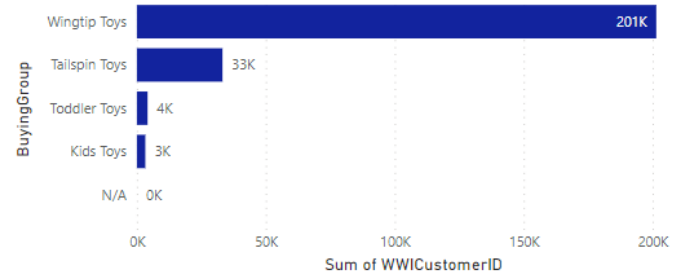


At 121404, Wingtip Toys had the highest Sum of CustomerKey and was Infinity higher than N/A, which had the lowest Sum of CustomerKey at 0.

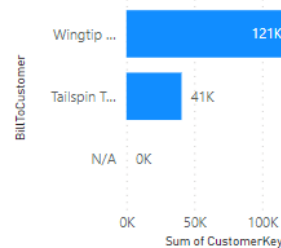
Wingtip Toys accounted for 74.94% of Sum of CustomerKey.

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

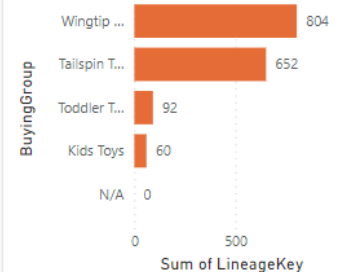
Sum of WWICustomerID by BuyingGroup



Sum of CustomerKey by BillToCustomer



Sum of LineageKey by BuyingGroup



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

File Export Save Share Chat in Teams

- 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



Save

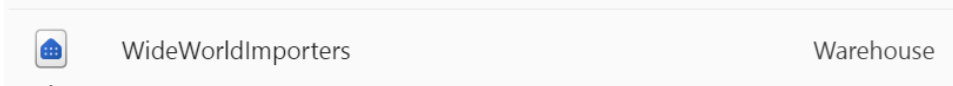
Cancel

## Module 2: Extending the solution

Now that you have seen 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,
```

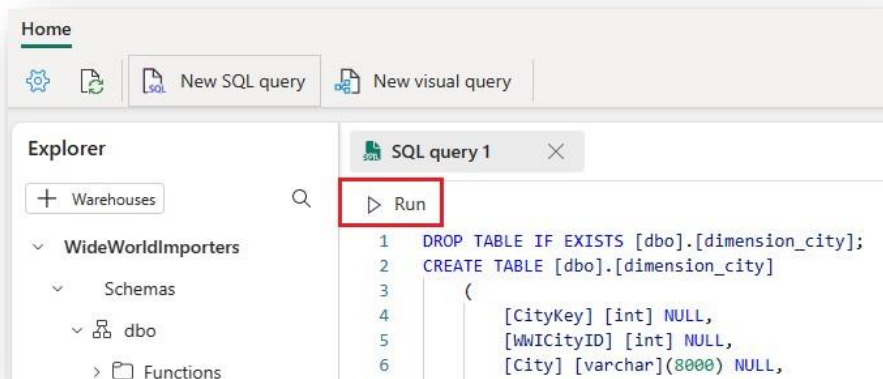
```

[ValidFrom] [datetime2](6) NULL,
[ValidTo] [datetime2](6) NULL,
[LineageKey] [int] NULL
);

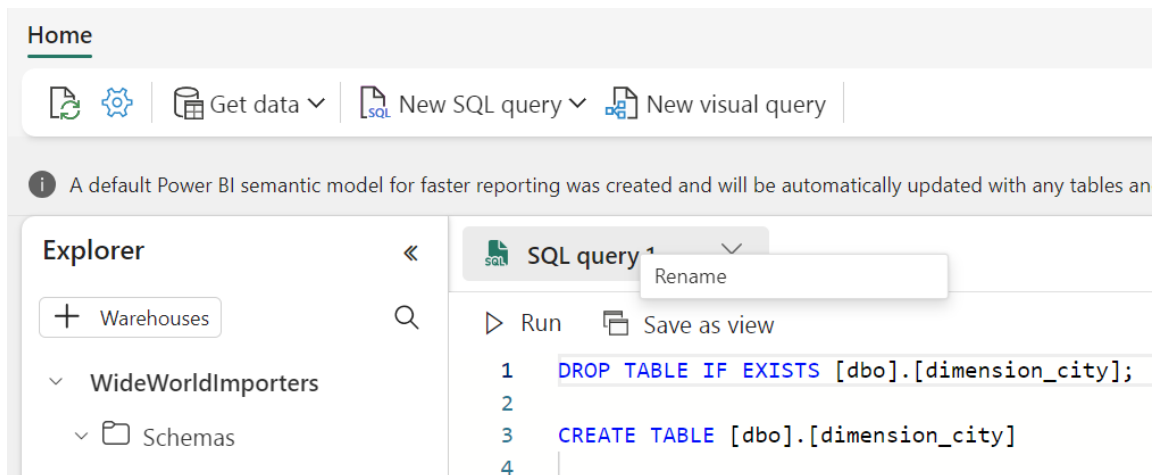
--fact_sale
DROP TABLE IF EXISTS [dbo].[fact_sale];
CREATE TABLE [dbo].[fact_sale]
(
    [SaleKey] [bigint] NULL,
    [CityKey] [int] NULL,
    [CustomerKey] [int] NULL,
    [BillToCustomerKey] [int] NULL,
    [StockItemKey] [int] NULL,
    [InvoiceDateKey] [datetime2](6) NULL,
    [DeliveryDateKey] [datetime2](6) NULL,
    [SalespersonKey] [int] NULL,
    [WWInvoiceID] [int] NULL,
    [Description] [varchar](8000) NULL,
    [Package] [varchar](8000) NULL,
    [Quantity] [int] NULL,
    [UnitPrice] [decimal](18, 2) NULL,
    [TaxRate] [decimal](18, 3) NULL,
    [TotalExcludingTax] [decimal](29, 2) NULL,
    [TaxAmount] [decimal](38, 6) NULL,
    [Profit] [decimal](18, 2) NULL,
    [TotalIncludingTax] [decimal](38, 6) NULL,
    [TotalDryItems] [int] NULL,
    [TotalChillerItems] [int] NULL,
    [LineageKey] [int] NULL,
    [Month] [int] NULL,
    [Year] [int] NULL,
    [Quarter] [int] NULL
);

```

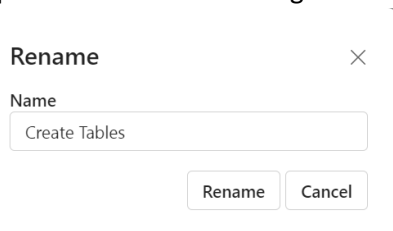
6. Select **Run** to execute the query.



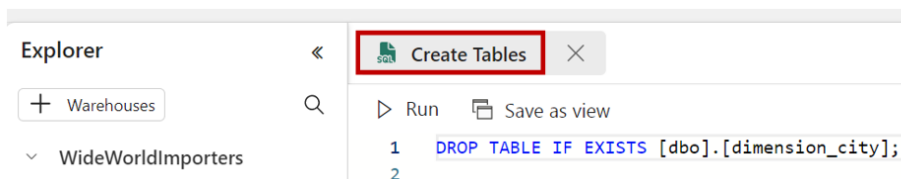
7. To save this query for reference later, right-click on the query tab just above the editor and select **Rename**.



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



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



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



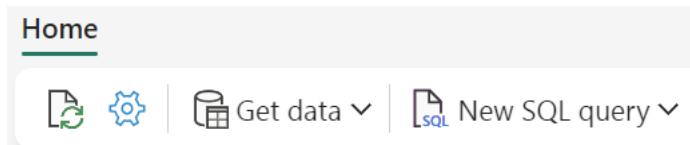
11. In the **Object explorer** verify that you can see the newly created **Create Tables** query, **fact\_sale** table, and **dimension\_city** table.



### Loading data using T-SQL

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





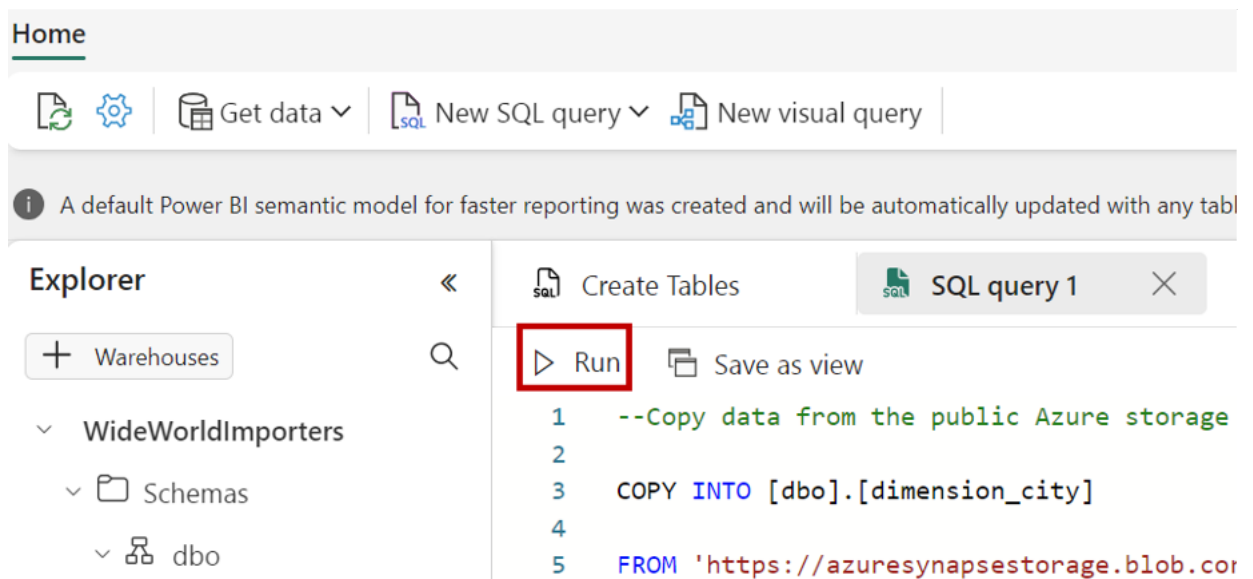
- In the query editor, paste the code below.

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

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

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

- Select **Run** to execute query. The query will take between 1 and 4 minutes to execute.



- After the query is completed, review the messages to see the rows affected which indicated the number of rows that were loaded into the dimension\_city and fact\_sale tables respectively.



- Load the data preview to validate the data loaded successfully by clicking on the **fact\_sale** table in the **Explorer**.

Explorer

Warehouses

WideWorldImporters

Schemas

dbo

Tables

dimension

dimension\_city

fact\_sale

Data preview

	12L SaleKey	123 CityKey	123 CustomerKey	123 BillToCustomerKey
1	17988584	39898	194	1
2	17988644	39898	194	1
3	17988704	39898	194	1
4	17989304	39898	194	1
5	17989364	39898	194	1
6	17989424	39898	194	1
7	17990024	39898	194	1
8	17990084	39898	194	1
9	17990144	39898	194	1
10	17988525	39898	194	1

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

Explorer

Warehouses

WideWorldImporters

Schemas

dbo

Tables

dimension

dimension\_city

fact\_sale

Views

Functions

Stored Procedures

guest

INFORMATION\_SCHEMA

queryinsights

sys

Security

Queries

My queries

Create new query

Data preview

25

26

27

28

29

30

31

32

33

34

35

36

37

38

39

40

41

42

Duplicate

Rename

Delete

Move to My queries

Move to Shared queries

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

Rename

×

Name

Load Tables

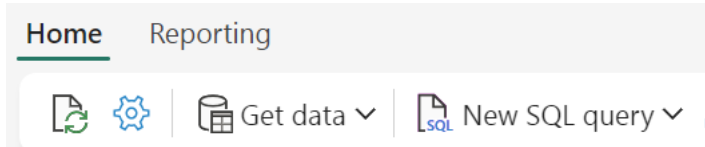
Rename

Cancel

8. Click **Rename** to change the name of the query

## 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)
    );

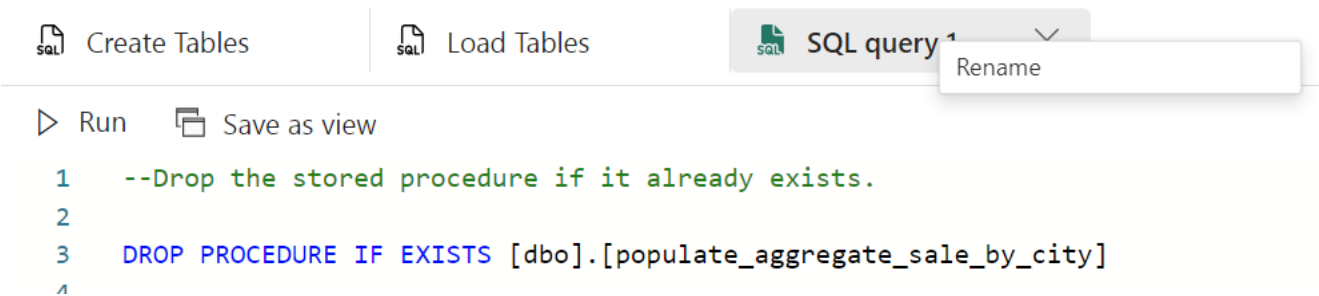
    --Reload the aggregated dataset to the table.
    INSERT INTO [dbo].[aggregate_sale_by_date_city]
    SELECT
        FS.[InvoiceDateKey] AS [Date],
        DC.[City],
        DC.[StateProvince],
        DC.[SalesTerritory],
        SUM(FS.[TotalExcludingTax]) AS [SumOfTotalExcludingTax],
        SUM(FS.[TaxAmount]) AS [SumOfTaxAmount],
        SUM(FS.[TotalIncludingTax]) AS [SumOfTotalIncludingTax],
        SUM(FS.[Profit]) AS [SumOfProfit]
```

```

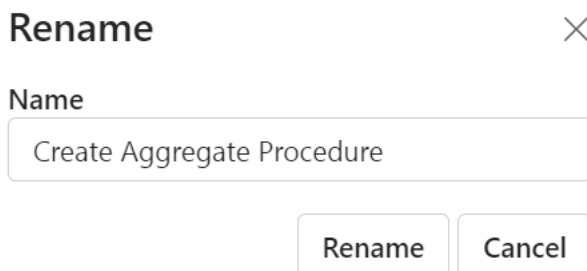
FROM [dbo].[fact_sale] AS FS
INNER JOIN [dbo].[dimension_city] AS DC
    ON FS.[CityKey] = DC.[CityKey]
GROUP BY
    FS.[InvoiceDateKey],
DC.[City],
    DC.[StateProvince],
    DC.[SalesTerritory]
ORDER BY
    FS.[InvoiceDateKey],
    DC.[StateProvince],
    DC.[City];
END

```

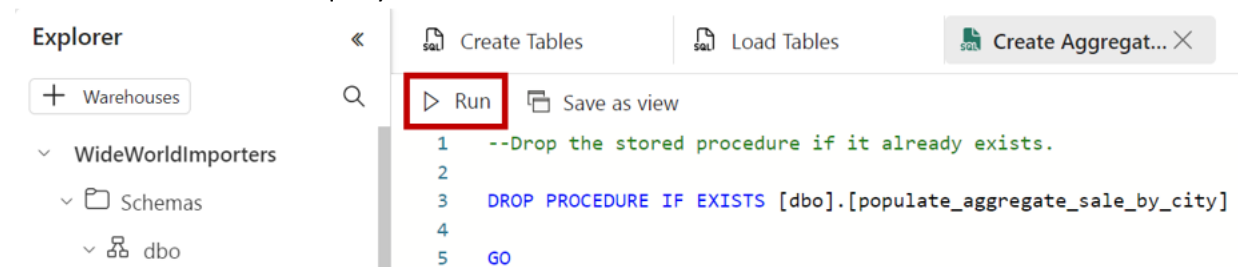
1. To save this query for reference later, right-click on the query tab just above the editor and select **Rename**.



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

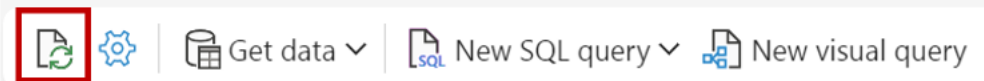


3. Click on **Rename** to save the query with given name
4. Select **Run** to execute the query.



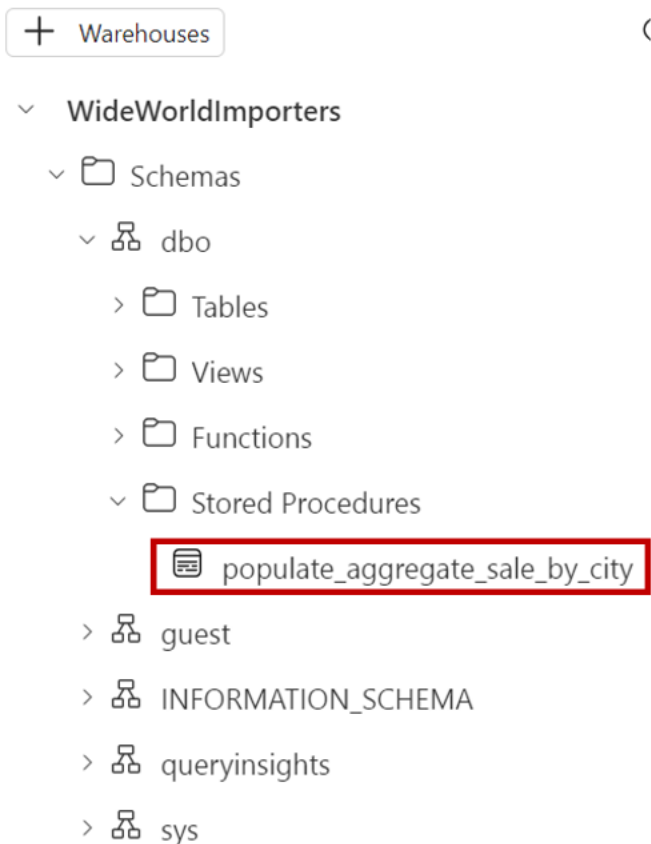
5. Click the **refresh** button on the ribbon.

## Home



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

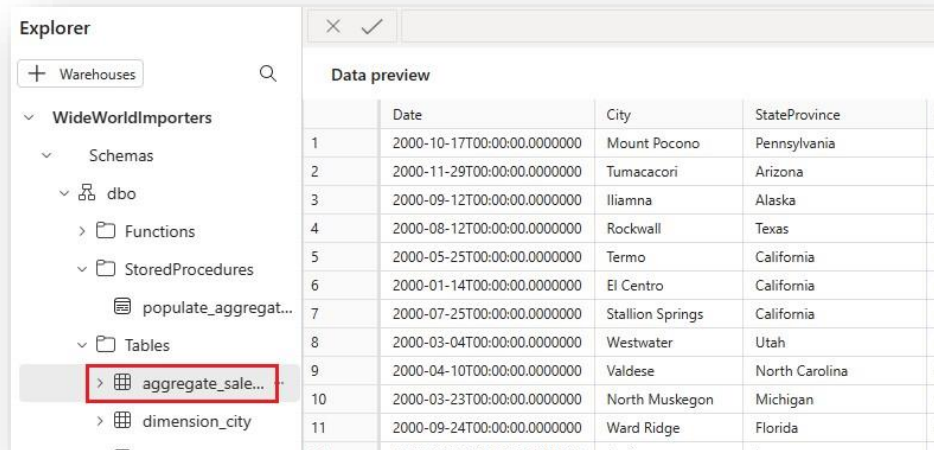
## Explorer



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 .

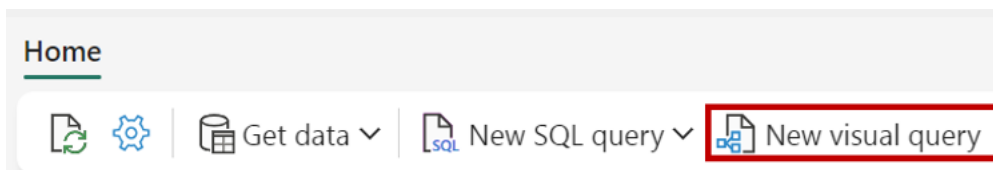
```
--Execute the stored procedure to create the aggregate table.  
EXEC [dbo].[populate_aggregate_sale_by_city];
```

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**.

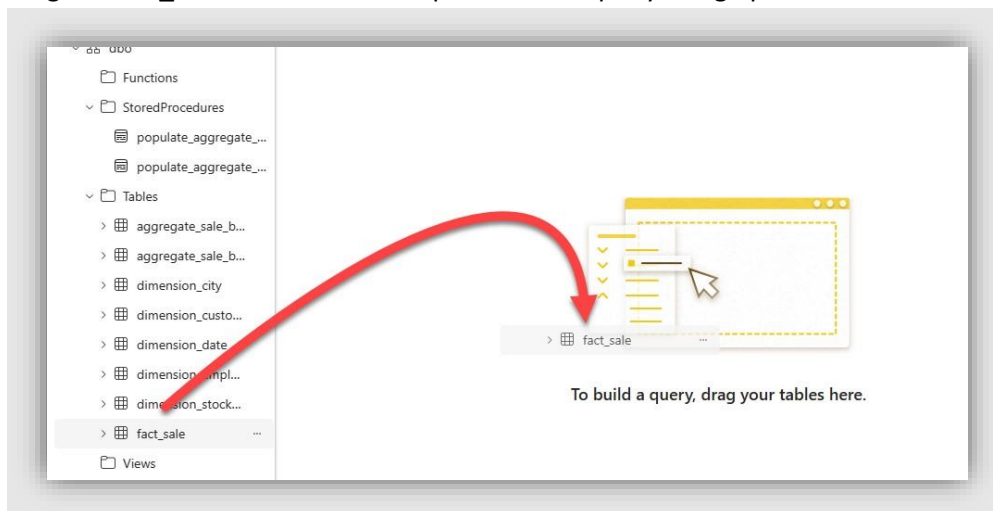


## Using the visual query builder

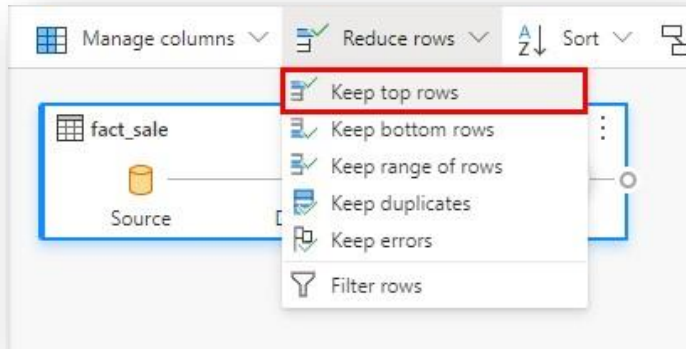
1. From the **Home** tab of the ribbon, select **New visual query**.



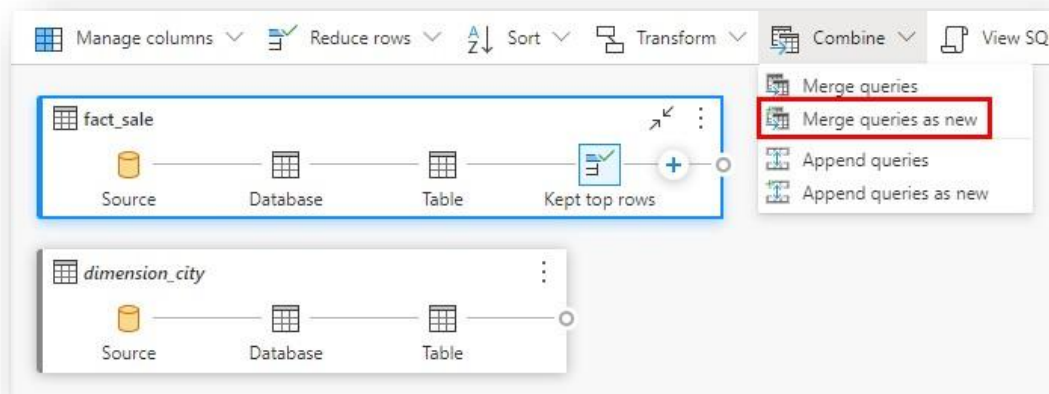
2. Drag the **fact\_sale** table from the explorer to the query design pane.



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

1-2-3 CityKey	1-2-3 WWCityID	1-2-3 City	1-2-3 StateProvince	1-2-3 Country	1-2-3 Continent	1-2-3
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	Palmdale	California	United States	North America	

Right table for merge \*

fact\_sale


1-2-3 SaleKey	1-2-3 CityKey	1-2-3 CustomerKey	1-2-3 BillToCustomerKey	1-2-3 StockItemKey	InvoiceDate
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 \*

☐ Left outer
 ☐ Right outer
 ☐ Full outer
 ☒ Inner
 ☐ Left anti
 ☐ Right anti

☐ Use fuzzy matching to perform the merge

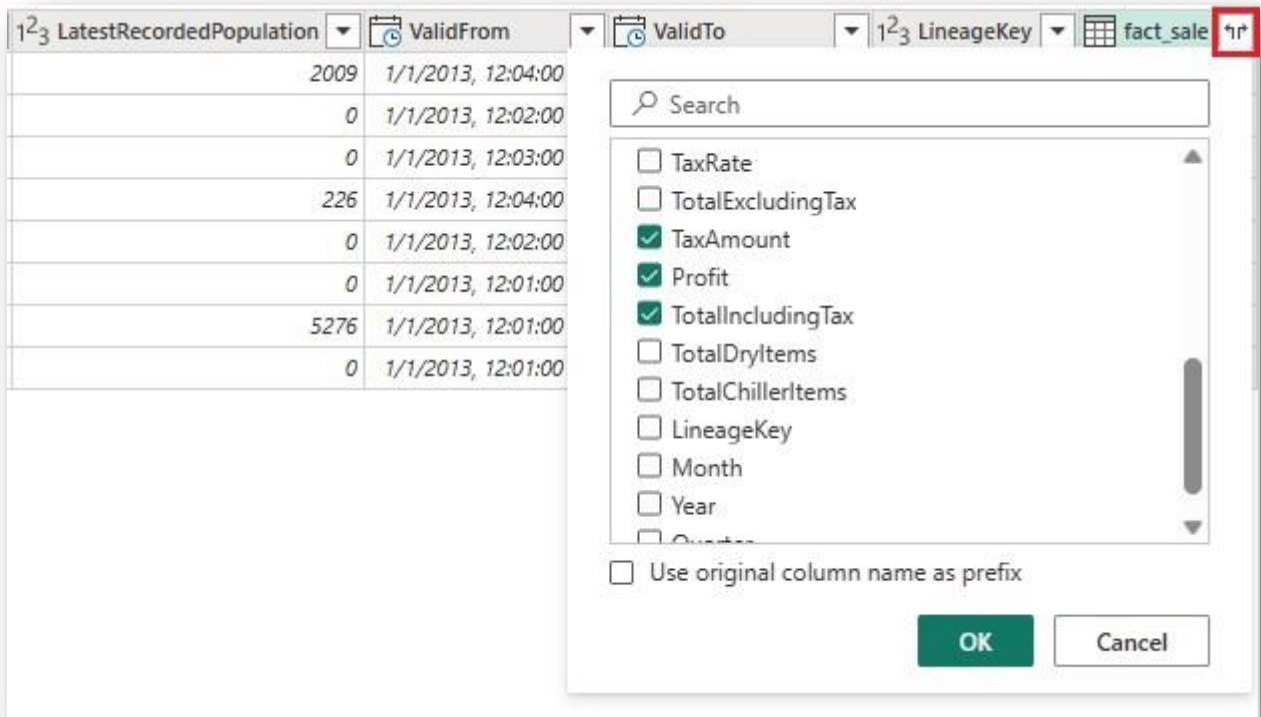
> Fuzzy matching options

 The selection matches 8 rows from both the tables

OK Cancel

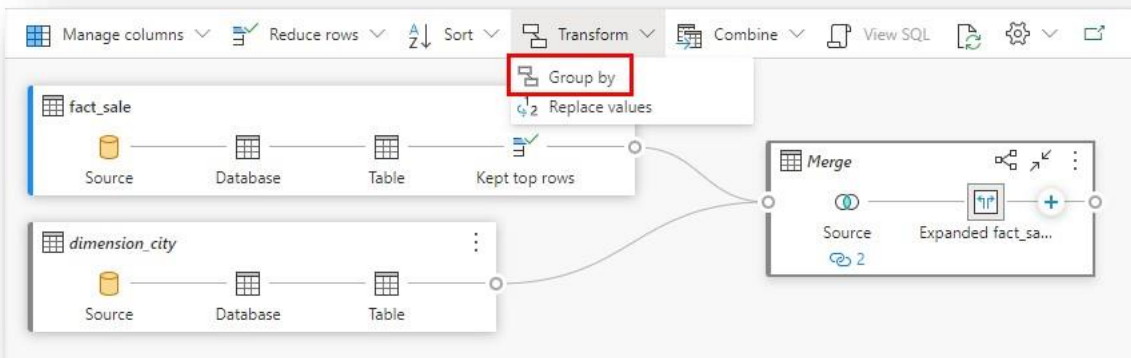
Select **OK**.

- 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**.



Select **OK**.

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



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

- Change to **Advanced**.
- Group by** (if necessary, select **Add grouping** to add additional group by columns):
  - Country
  - StateProvince
  - City
- 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

Add grouping

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

Add aggregation

☐ Use fuzzy grouping

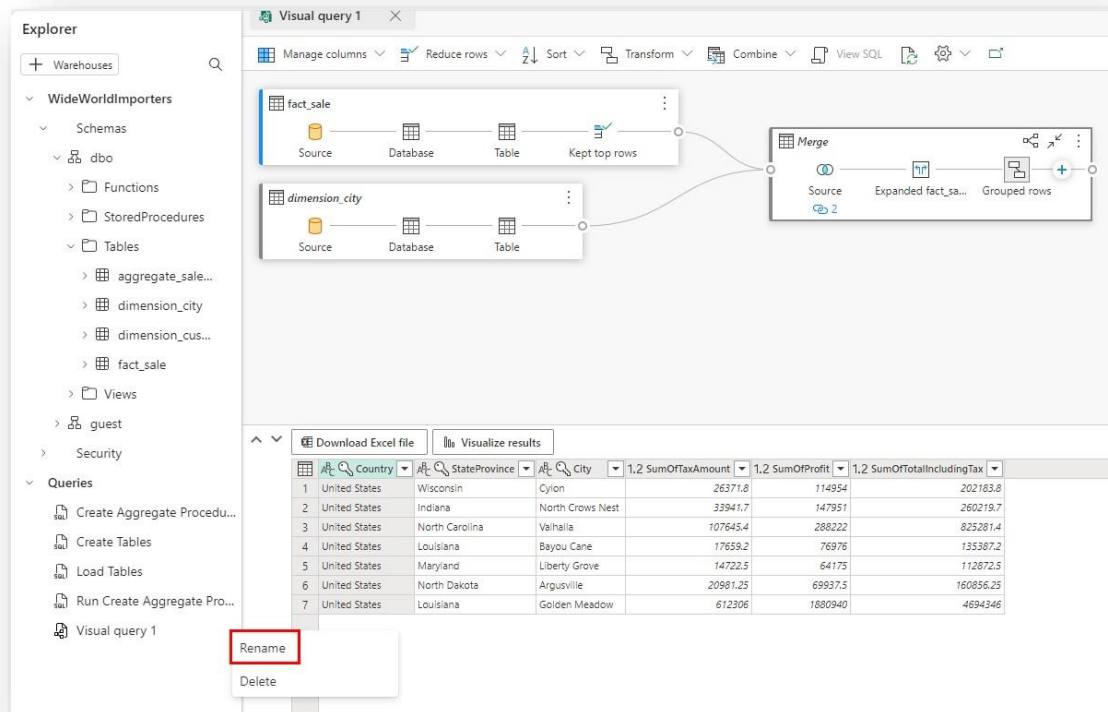
> Fuzzy group options

OK

Cancel

Select **OK**.

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



13. Type **Sales Summary** to change the name of the query.

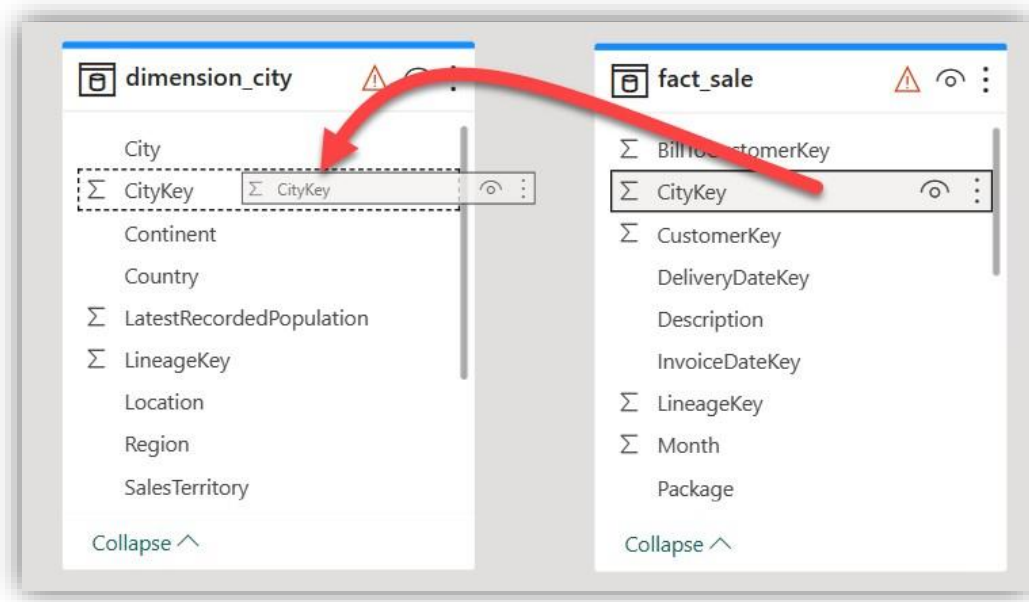
14. Press **Enter** on the keyboard or click off anywhere outside the tab to save the change.

## Create a Power BI report

1. 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

**Table 2**  
dimension\_city

**Column:** CityKey

**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

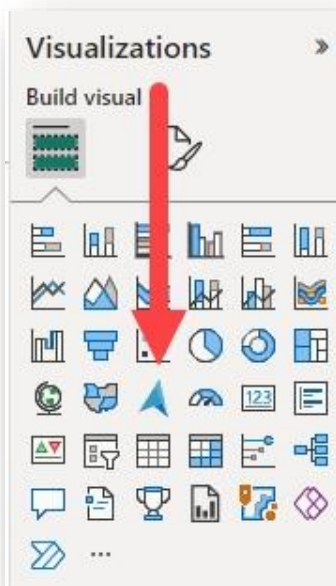
**Confirm** **Cancel**

Select **Confirm**.

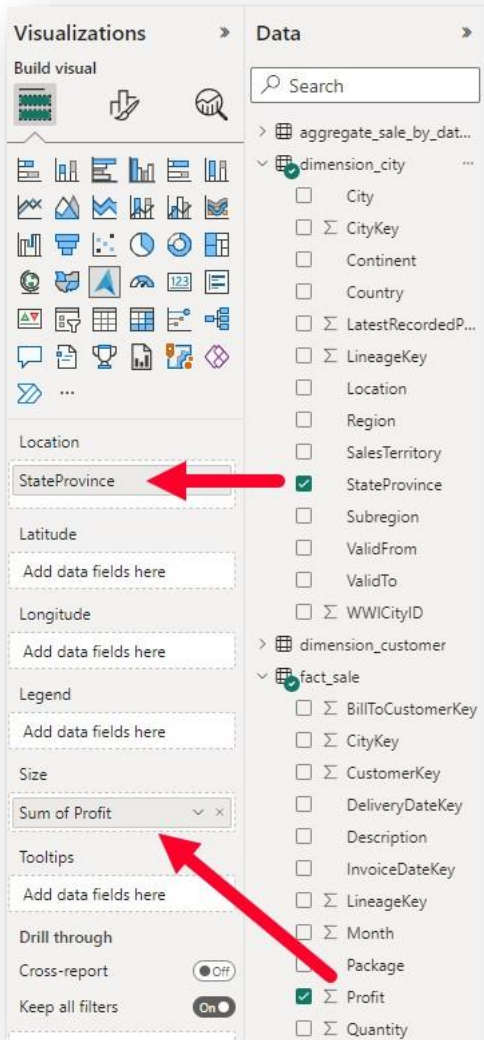
4. From the **Home** tab of the ribbon, select **New report**.
5. Build a column chart visual:
  - a. On the **Data** pane, expand **fact\_sales** and check the box next to **Profit**. This will create a column chart and add the field to the Y-axis.
  - b. On the **Data** pane, expand **dimension\_city**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

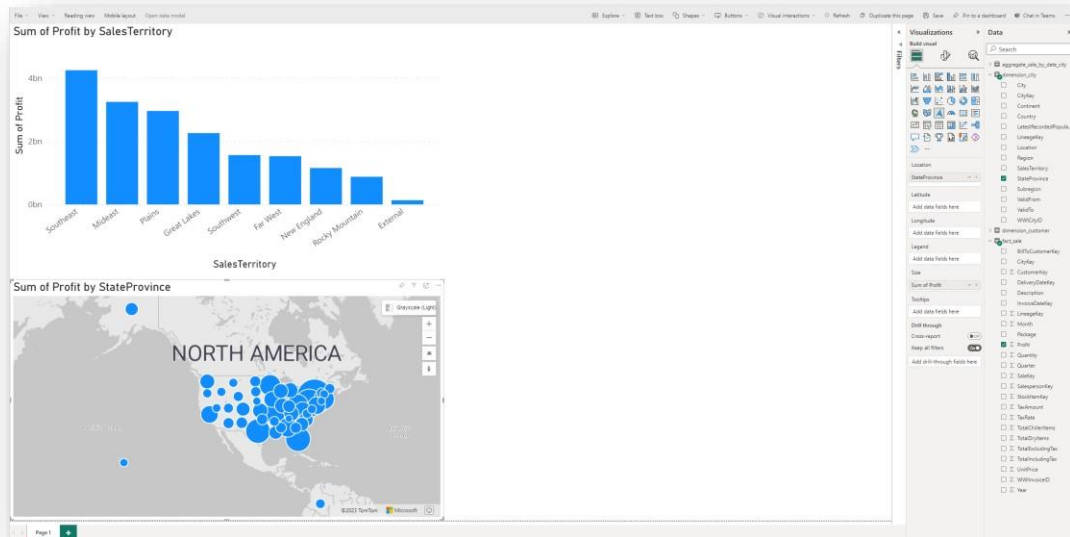


- b. From the **Data** pane, drag **StateProvince** from the **dimension\_city** table to the **Location** bucket on the **Visualizations** pane.
  - c. From the **Data** pane, drag **Profit** from the **fact\_sale** table to the **Size** bucket on the **Visualizations** pane.

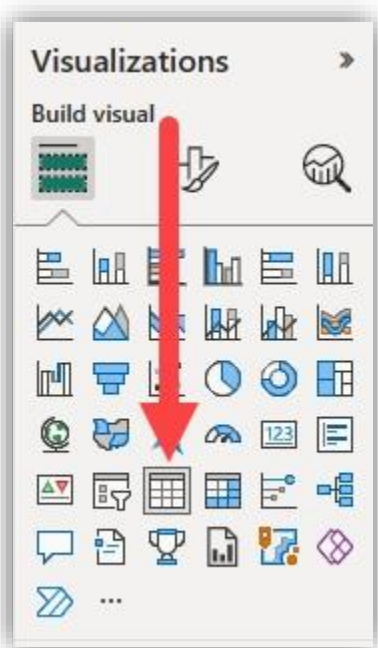


- d. If necessary, reposition and resize the map to take up the bottom left quarter of the canvas by dragging the anchor points on the corners of the visual.





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.



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 \*

Select a destination workspace

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

Save
Cancel