Part of the Power BI Bootcamp Series

Microsoft Power BI
Intermediate Bootcamp

Data Connections and Transformations

Instructors:

Nicolás Lagreste Zucchini

María Florencia Hourcouripé

O'REILLY®

Class 01

Course Agenda

Class 1: Introduction and getting started with M

- Course Framework and Introduction
- Data Sources and Connectivity
- Getting Started with M: A Step Beyond the UI
- Combining Data

Class 2: Important tools in the M language

- Understanding Object Structures and Data Types
- Intermediate Data Transformation
- Advanced Query Techniques and Error Handling
- Final Comments and Advanced Tips

Our Method on each topic:
Demo example, followed by exercises to practice!

- **DEMO – Flor & Nico**

- **EXERCISE – Your turn!**

*Students should have the latest version of **Power BI Desktop installed.**

Instructors

Follow us on LinkedIn:

 Nicolás Lagreste Zucchini

 María Florencia Hourcouripé

Let's Collaborate: Interested in a personalized mentorship or consulting session? Contact us:

- nico@analyticmood.com
- flor@analyticmood.com

Visit Our Website:

- www.analyticmood.com for resources, upcoming events, and blog articles.

Course Framework and Introduction

Course Overview

1. **Introduction to Power Query**
   - Understand the structure, workflow, and foundational concepts of Power Query.

2. **Connecting to Data Sources**
   - Learn how to connect to and access a wide range of data sources.

3. **M Language for Advanced Transformations**
   - Write custom code for complex data transformations beyond the user interface.

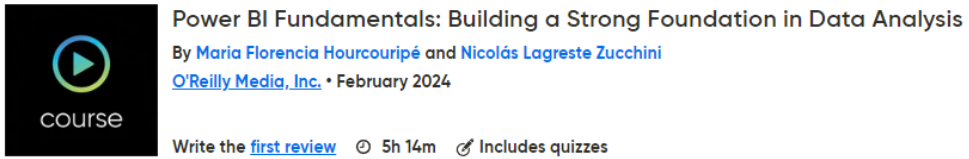
4. **Combining Data**
   - Master techniques to merge and integrate data from multiple sources and tables.

Before getting started

- We will work almost exclusively in the **Power Query** section of Power BI.
- We'll help you with concepts you've heard of but may not have worked with before.
- This is not an advanced course; it is specifically designed for **intermediate Power BI users** who are already familiar with using Power Query through the UI.
- Have **Power BI Desktop installed** (it is recommended to use the latest version available).
  - We will use January 2025 version in the Demos.
  - Microsoft web page: https://www.microsoft.com/en-us/download/details.aspx?id=58494

Before getting started

- To be better prepared, review our introductory course: "**Power BI Fundamentals** or attend our live event: "**Power BI Bootcamp**"



https://learning.oreilly.com/course/power-bi-fundamentals/0636920986317/

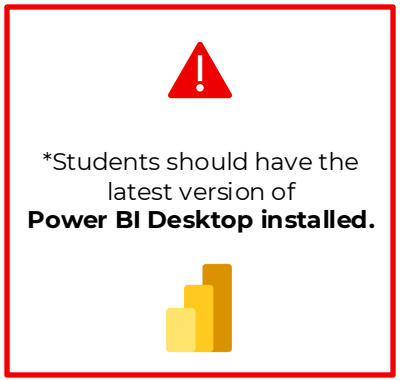
Starting Point for the Bootcamp

We will begin with the project from our previous Power BI Bootcamp.

To get started, download the following files:
Download Here: **www.analyticmood.com/resources**

1. **Power BI Project**:
    - Pbix: Exercise 1 to 10 and FinalPowerQuery

2. **DataSources**:
    - Sales_2022_2023.xlsx
    - Sales_data_2024.csv
    - Budget 2024.pdf
    - Budget 2022_2023



Ensure all files are downloaded before proceeding.

Let's do the following quiz to warm up

Question 1

What is Power Query?

a) An advanced programming language

b) An ETL (Extract, Transform, Load) tool in Excel and Power BI

c) A type of chart in Power BI

Question 2

What language is used in Power Query to transform data?

a) DAX

b) SQL

c) M

Question 3

In which format are Power Query queries saved in Power BI?

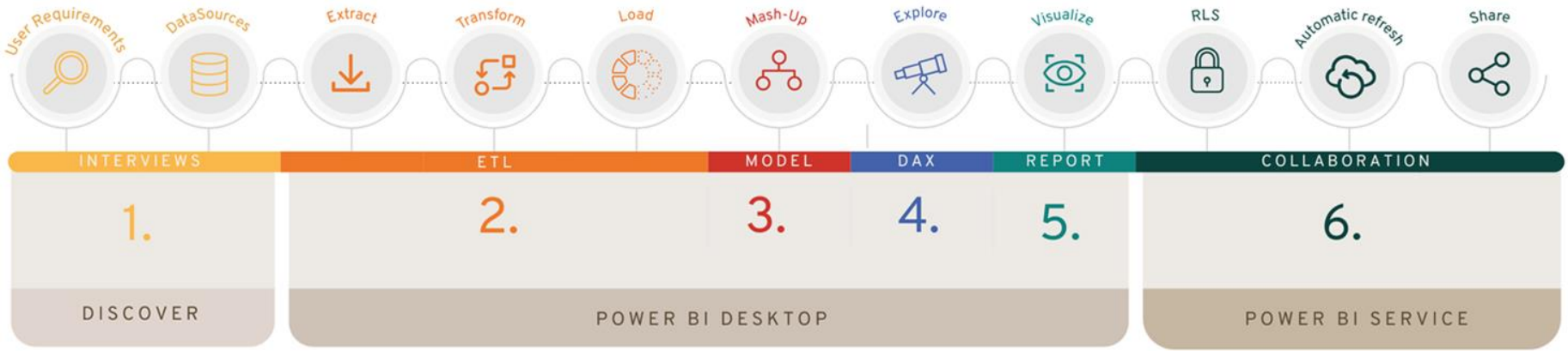
a) In a SQL database file

b) In a .pbix file

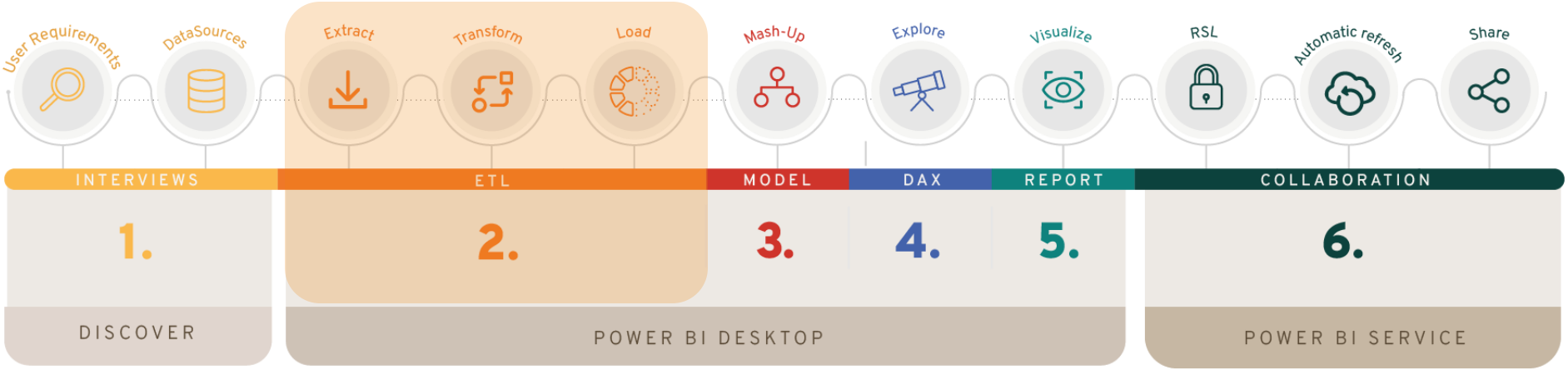
c) In a text file

Flow of Report Development

Our Method: Flow of Report Development



Our Method: Flow of Report Development

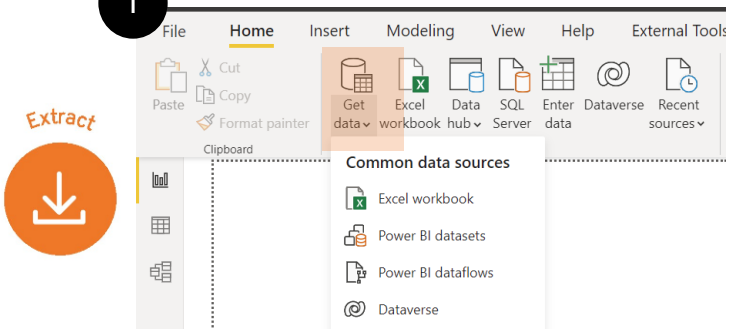


What is Power Query?

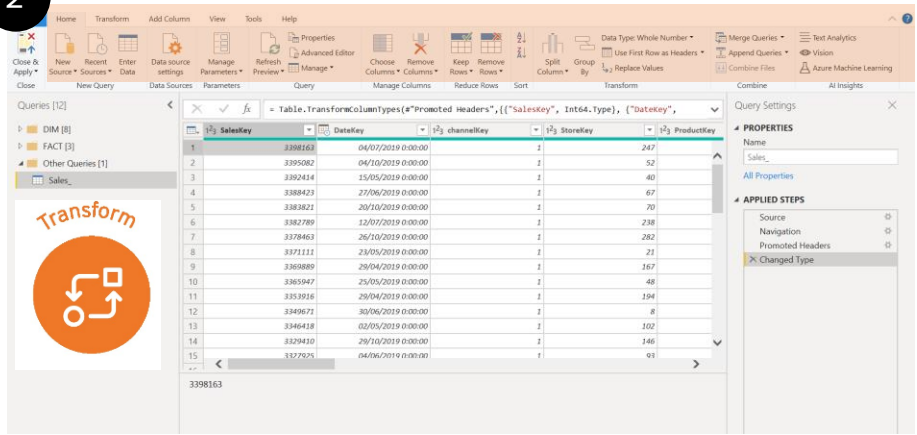
Power Query

Clean and transform data sources

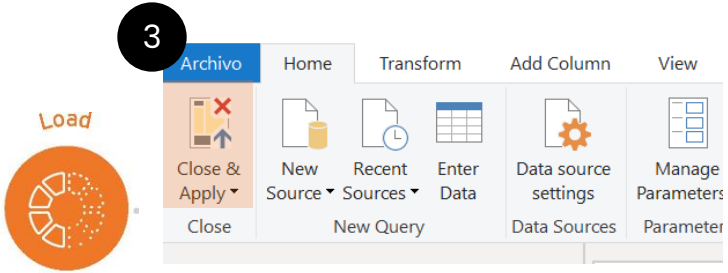
1



2



3



What is Power Query?

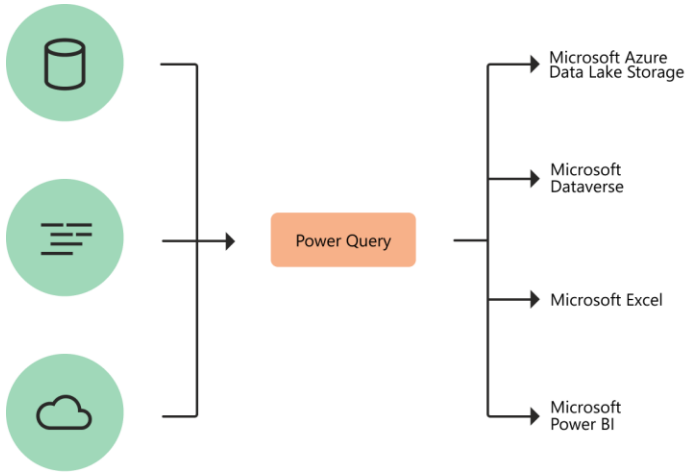
A powerful tool for **data connection, cleaning, and transformation.**

- Built-in interface with advanced capabilities using M language.
- Works seamlessly with multiple data sources.
- Ideal for automating repetitive data tasks.

How Power Query helps with data acquisition

Business users spend up to 80% of their time preparing data, which slows down analysis and decision-making.

Power Query **streamlines this process**, making it easier to connect, transform, and update data efficiently for faster insights.



DEMO 01

Change data sources

Exercise 1

Change data sources

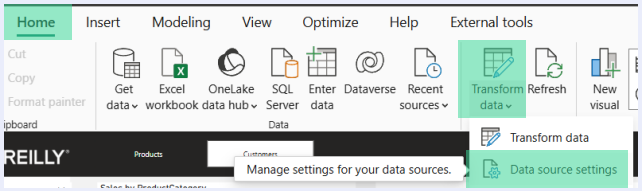
Exercise 1: Change the connections of the data sources

Let's do this together

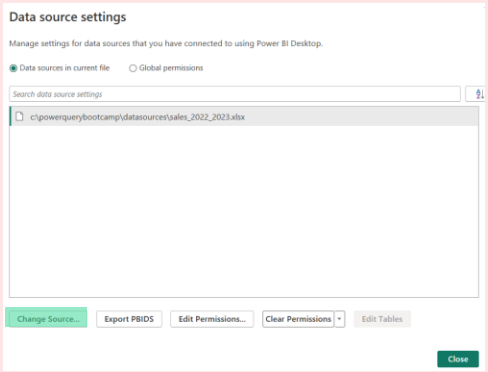
Goal: *Learn to connect data sources on your personal computer.*

Open the file ***Exercise1.pbix*** *(Location: C:\PowerQueryBootcamp\Exercise1).*

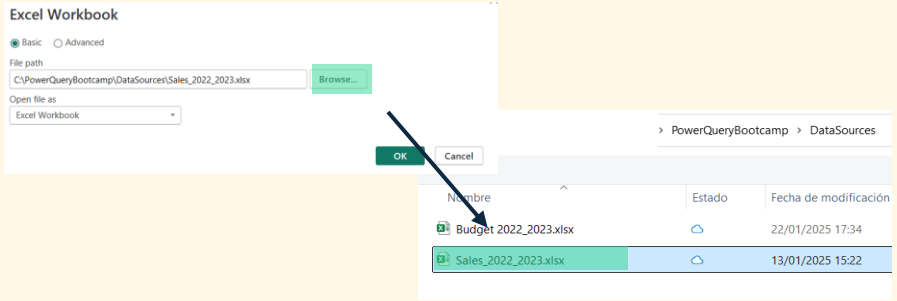
1 Click on ***Home / Transform Data / Data Source Setting***



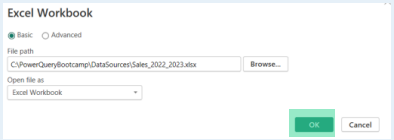
2 Click on ***Change Source*** at the bottom



3
1. *Click on **Browse** at the bottom*
2. *Go to **PowerQueryBootcamp/DataSources***
3. *Select **Sales_2022_2023***
4. *Click **OK**, then **Open***



4 In ***Excel Workbook*** window click on ***OK*** at the bottom



Overview of Power BI Data Sources

Power BI Data Sources

Get data

Power BI Data Sources

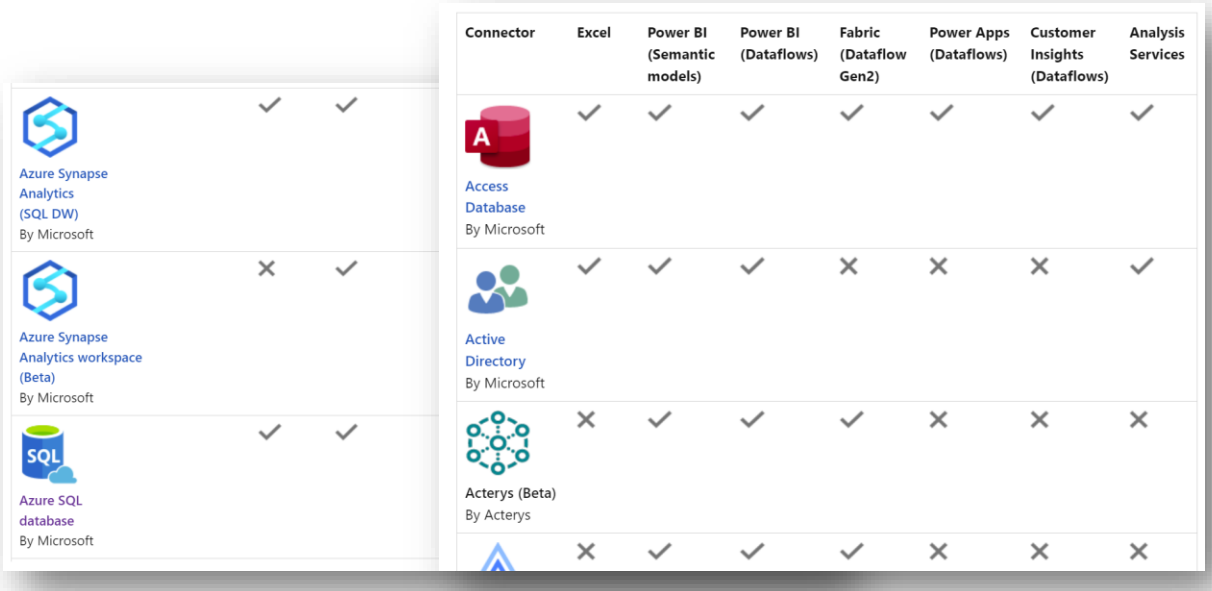
Types of Connectors	Types of Connections	Benefits of Using Connections
• Regular • Custom • Certified	• DirectQuery • Import Mode • Live Connection	• Up-to-date Information • Flexibility in Data Handling • Resource Efficiency • Scalability

Connectors

- Power Query offers a wide range of connectors for various data sources.

- For a complete list of available connectors and their details, check the link provided below.

  https://learn.microsoft.com/en-us/power-query/connectors/



Connector	Excel	Power BI (Semantic models)	Power BI (Dataflows)	Fabric (Dataflow Gen2)	Power Apps (Dataflows)	Customer Insights (Dataflows)	Analysis Services
Azure Synapse Analytics (SQL DW) By Microsoft	✓	✓					
Azure Synapse Analytics workspace (Beta) By Microsoft	✗	✓					
Azure SQL database By Microsoft	✓	✓					
Access Database By Microsoft	✓	✓	✓	✓	✓	✓	✓
Active Directory By Microsoft	✓	✓	✓	✗	✗	✗	✓
Acterys (Beta) By Acterys	✗	✓	✓	✓	✗	✗	✗
	✗	✓	✓	✓	✗	✗	✗

Connectors – M functions

- Power Query supports various file formats, utilizing M functions for efficient data retrieval and transformation. Common formats and their M functions are listed below.

File Format	M Functions
Azure Storage	AzureStorage.BlobContents, AzureStorage.Blobs, AzureStorage.DataLake, AzureStorage.DataLakeContents, AzureStorage.Tables
Binary	File.Contents
Excel	Excel.Workbook, Excel.CurrentWorkbook
Folder	Folder.Contents, Folder.Files
HDFS	Hdfs.Contents, Hdfs.Files
JSON	Json.Document
PDF	Pdf.Tables
Parquet	Parquet.Document
Text/CSV	Csv.Document
XML	Xml.Document, Xml.Tables

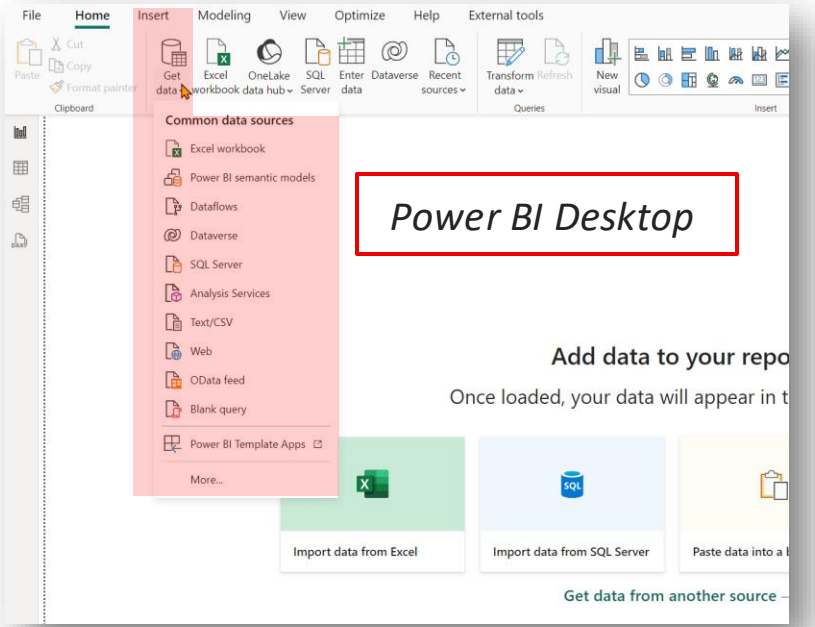
DEMO 02

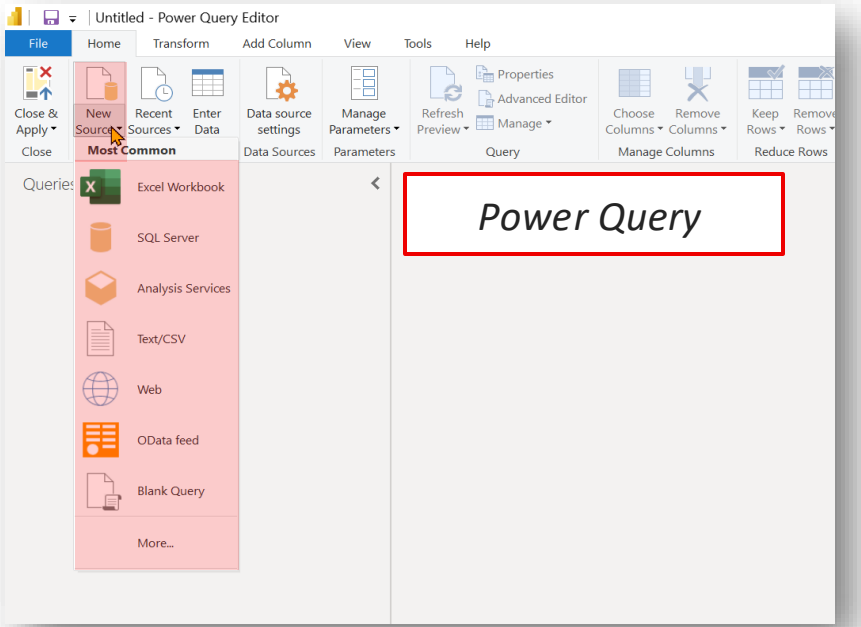
- Excel
- PDF
- SharePoint
- Azure Blob Storage
- SQL

Accessing Files

Accessing Files

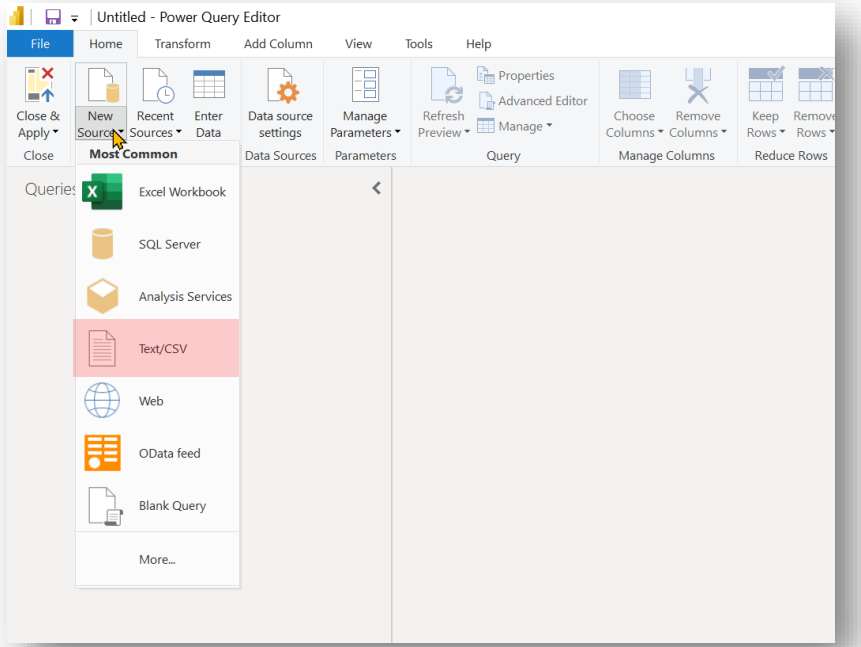
- In Power BI Desktop, you can connect directly to data sources, but within Power Query, there's also the **option to retrieve data.**
- Below, you'll see the difference between the access buttons in both interfaces.





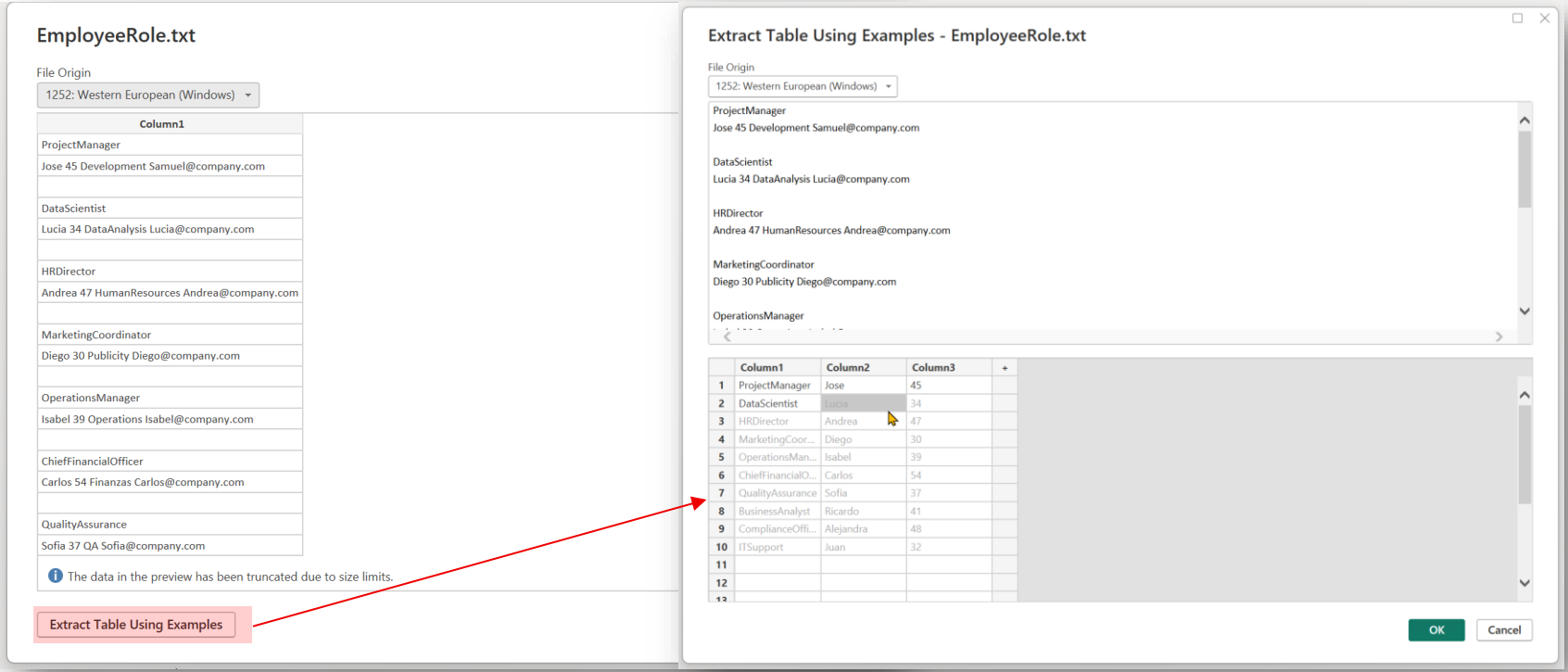
Text/CSV

- Power Query automatically treats CSVs as structured files with a comma delimiter. For text files, it will try to detect if delimiter-separated values are present and infer the delimiter for structured data import.



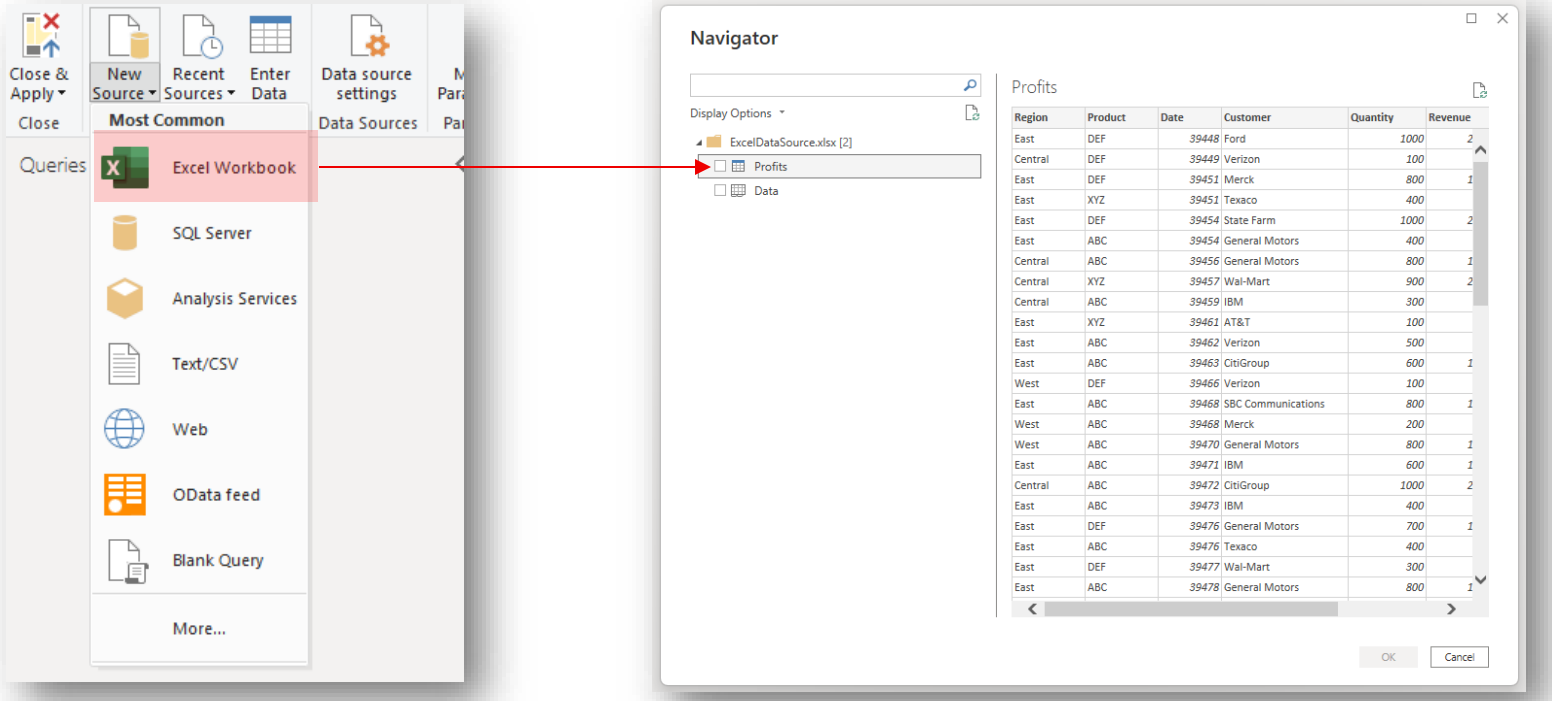
Text/CSV by Example

- This feature allows you to manually specify sample values from a Text/CSV file, and Power Query will automatically extract the rest of the data based on your input.



Excel

- Connecting Excel to Power Query in Power BI allows users to import and transform data from Excel spreadsheets. Here's a quick overview of the process:



Connecting with SharePoint

SharePoint

*Power Query enables connections to SharePoint **document libraries and lists**, allowing users to retrieve and transform SharePoint files and folders or list data.*

- **Prerequisites**: Requires appropriate access to the SharePoint **site** or **library**.
- **Capabilities Supported**: Import mode; connection to document libraries and SharePoint lists.
- **Limitations:**
  - DirectQuery is not supported.
  - Large lists may result in slower performance, and permission management can be complicated.
  - The size of files and lists can significantly impact performance, and non-Office file types may not be fully supported.

Excel in SharePoint

With Power Query, Excel files stored in SharePoint can be accessed and integrated into Power BI.

- **Prerequisites:** Requires access to the SharePoint site or Excel file with proper permissions
- **Capabilities Supported:** Import mode, Excel file manipulation
- **Limitations:**
  - DirectQuery is not supported
  - Accessing multiple large sheets may slow down queries

Utilizing Azure Blob Storage

Azure Blob Storage

The Blob Storage connector allows Power Query to retrieve data from Azure Blob Storage in various formats, such as CSV, JSON, and images.

- **Prerequisites:** Access to Azure Storage account and permissions (Account key or SAS token)
- **Capabilities Supported:** Import mode, Binary access, hierarchical navigation of Blob containers
- **Limitations:**
  - DirectQuery is not supported
  - Handling very large blobs can slow query execution, especially for complex transformations
  - Concurrent requests and request size parameters need adjustment for optimal performance

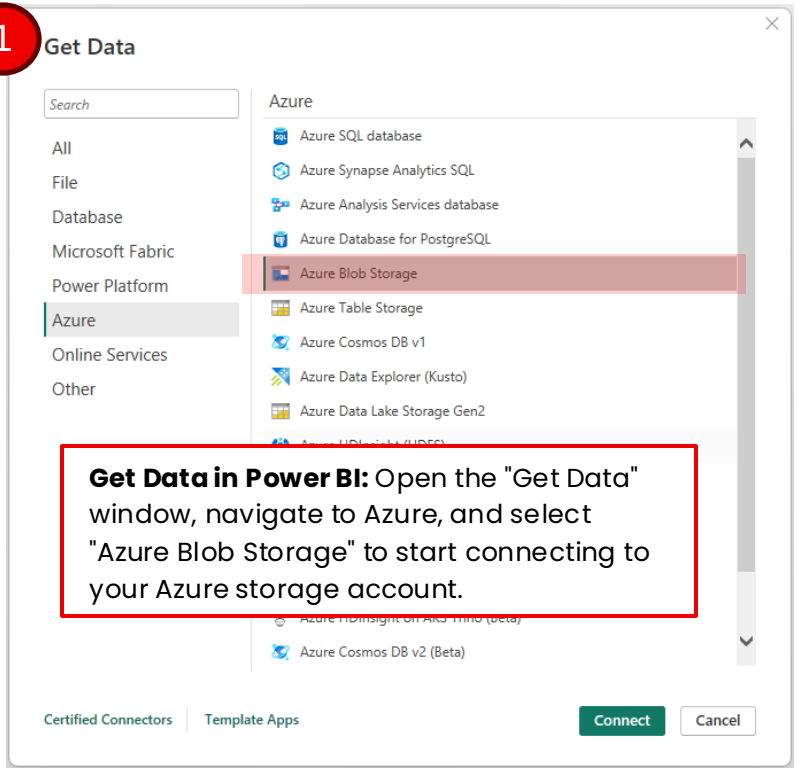
Azure Blob Storage – Benefits

- Connecting Power BI with Azure Blob Storage offers several key benefits:
  - Cost-effective data storage
  - Flexible data access
  - Scalability for large datasets
  - Seamless data refresh capabilities
  - Secure data management with encryption and access control
  - Integration with other Azure services for enhanced analytics

Azure Blob Storage

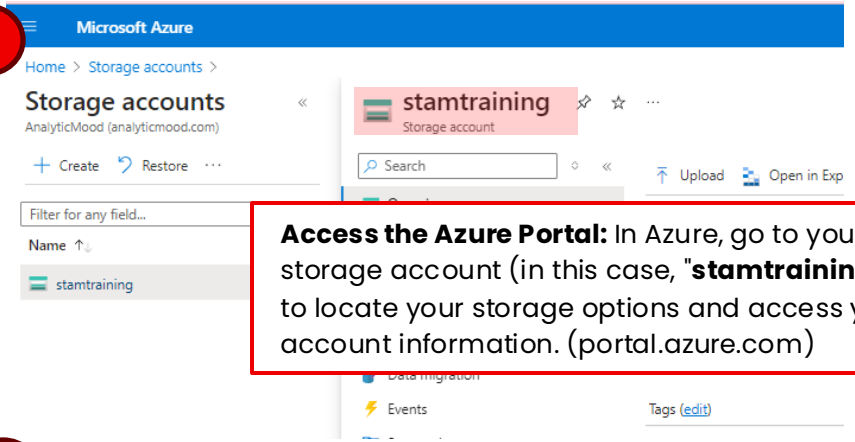
To access a Blob Storage Account, follow these steps:

1



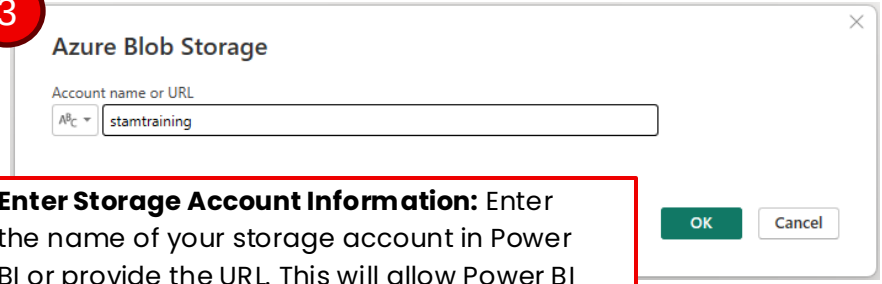
Get Data in Power BI: Open the "Get Data" window, navigate to Azure, and select "Azure Blob Storage" to start connecting to your Azure storage account.

2



Access the Azure Portal: In Azure, go to your storage account (in this case, "**stamtraining**") to locate your storage options and access your account information. (portal.azure.com)

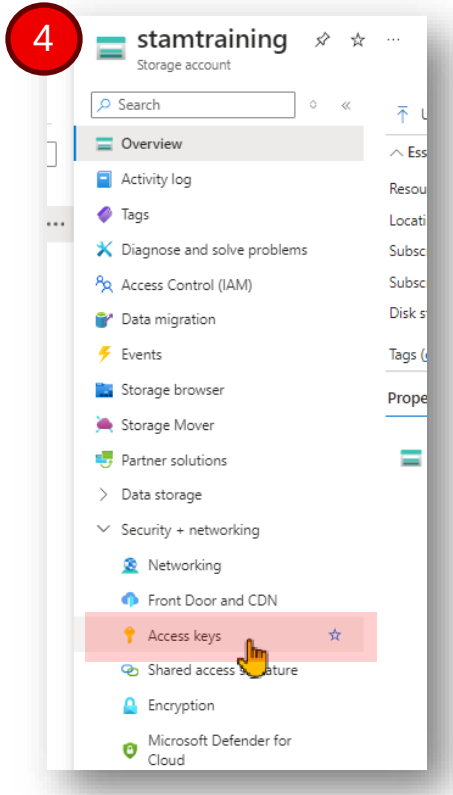
3

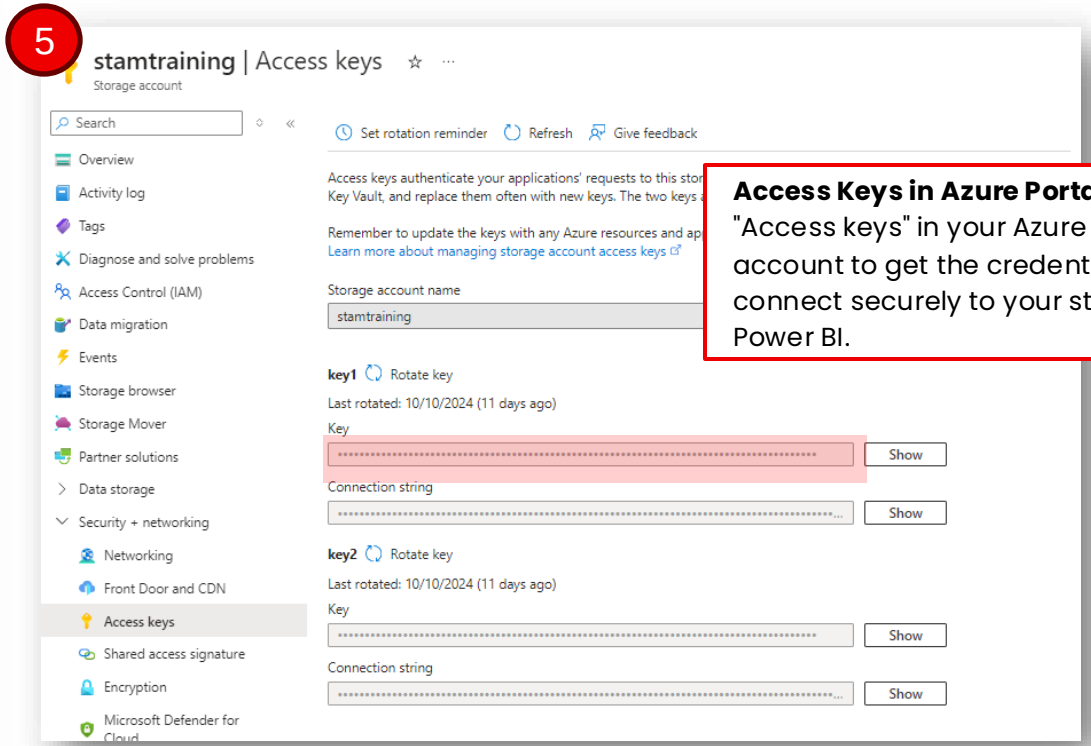


Enter Storage Account Information: Enter the name of your storage account in Power BI or provide the URL. This will allow Power BI to establish a connection to your Blob Storage account.

Azure Blob Storage

To access a Blob Storage Account, follow these steps:



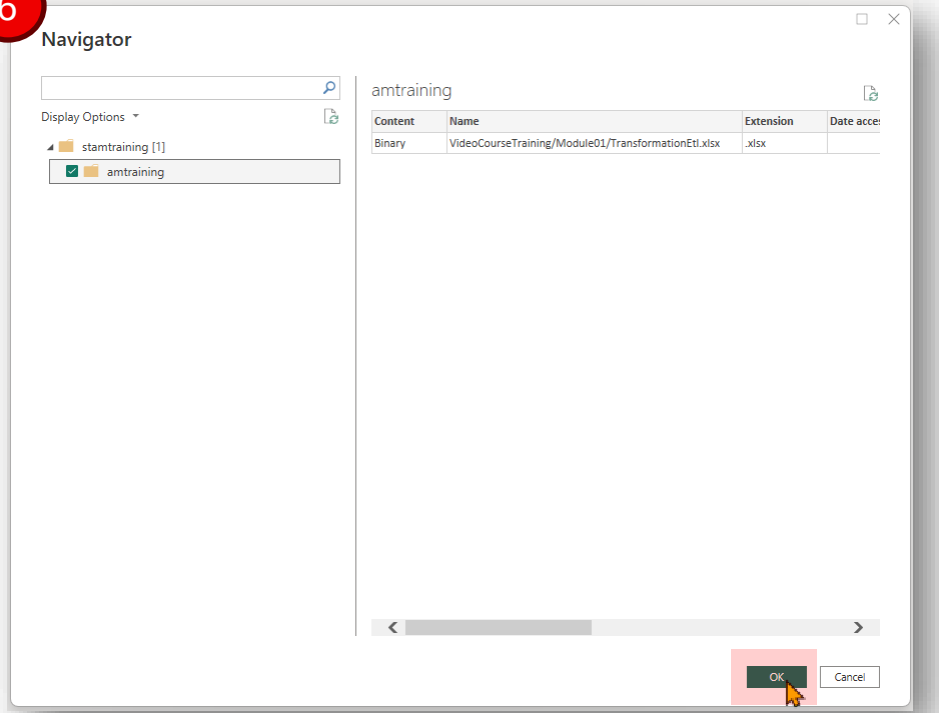


Access Keys in Azure Portal: Navigate to "Access keys" in your Azure Storage account to get the credentials required to connect securely to your storage from Power BI.

Azure Blob Storage

To access a Blob Storage Account, follow these steps:

6



Select Data in Navigator: Once connected, Power BI will show the available files in the Blob Storage. Select the folder you want to connect to and click "OK."

Connecting with Local Folders

Local Folder

Power Query connects to local or network folders, enabling you to aggregate and transform data from multiple files within a directory.

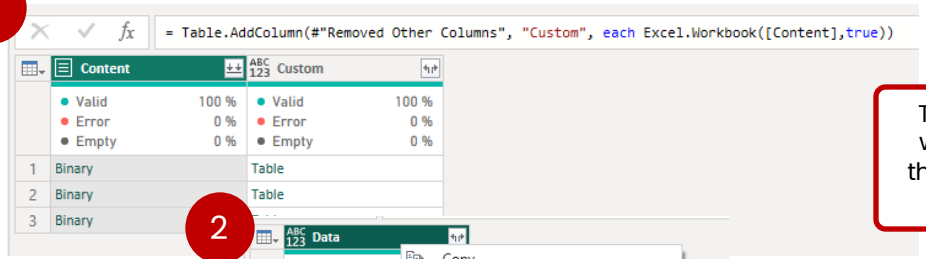
- **Prerequisites:** Local or network access to the folder; permissions to read the files.
- **Capabilities Supported:** Import mode, file combination.
- **Limitations:**
  - No DirectQuery support
  - Network paths may require additional permission setups, and inconsistent file structures can complicate transformations
  - Large datasets may take longer to load, and the connector doesn't work well with very deeply nested folder structures

Custom Functions to Combine Files

QUICK TIPS

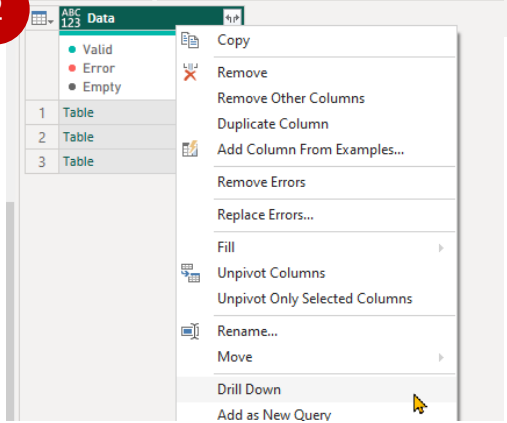
Dynamically combine multiple Excel files from a folder, leveraging functions **like Excel.Workbook** to extract and promote headers, and **Table.Combine** to merge data from each file seamlessly.

1



The Excel.Workbook function is used with the true parameter to promote the headers automatically from each file.

2



Before combining the data, we need to generate a list of tables from the files. To do this, drill-down by clicking on the header of the column.

Custom Functions to Combine Files

QUICK TIPS

Dynamically combine multiple Excel files from a folder, leveraging functions **like Excel.Workbook** to extract and promote headers, and **Table.Combine** to merge data from each file seamlessly.

3

```
= #"Removed Other Columns1"[Data]
```

List
Table
Table
Table

This list will be used as a parameter in the **Table.Combine** function.

4

```
= Table.Combine(#"Removed Other Columns1"[Data])
```

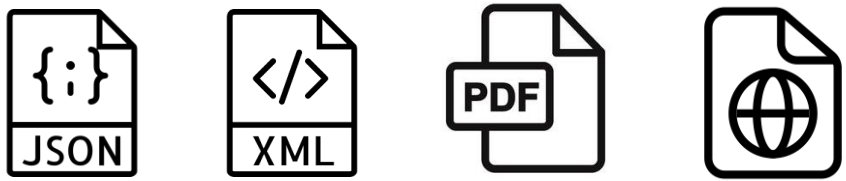
	Name	ProductNumber	Color	StandardCost
	Valid 100 % Error 0 % Empty 0 %	Valid 100 % Error 0 % Empty 0 %	Valid 83 % Error 0 % Empty 17 %	Valid 100 % Error 0 % Empty 0 %
1	HL Mountain Frame - Black, 46	FR-M94B-46	Black	739,041
2	HL Mountain Frame - Black, 38	FR-M94B-38	Black	739,041
3	HL Mountain Frame - Silver, 38	FR-M94S-38	Silver	
4	Sport-100 Helmet, Blue	HL-U509-B	Blue	
5	AWC Logo Cap	CA-1098	Multi	
6	Long-Sleeve Logo Jersey, S	LJ-0192-S	Multi	
7	Long-Sleeve Logo Jersey, M	LJ-0192-M	Multi	
8	Long-Sleeve Logo Jersey, L	LJ-0192-L	Multi	38,4923
9	Long-Sleeve Logo Jersey, XL	LJ-0192-X	Multi	38,4923
10	HL Road Frame - Red, 62	FR-R92R-62	Red	868,6342
11	HL Road Frame - Red, 44	FR-R92R-44	Red	868,6342

The **Table.Combine** function is used to merge all data without hardcoding the column structure.

Connecting to Other File Types (JSON, PDF, Web Content)

Connecting to Other File Types

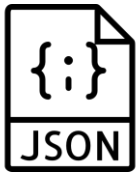
- Many users work with M by accessing files and folders from cloud storage, network drives, or even locally.
- Power Query supports a variety of formats like **JSON**, **XML**, **PDF**, and more, each with its own M function to make data extraction and transformation easy and efficient.



JSON

Power Query can retrieve and transform data from JSON files, typically used for APIs or data exchange in structured format.

- **Prerequisites:** JSON file or access to an API providing JSON data.
- **Capabilities Supported:** Import mode, deep parsing of nested structures.
- **Limitations:**
  - Complex or deeply nested JSON structures can slow down performance
  - API rate limits can affect how much data you can pull



PDF

The PDF connector in Power Query extracts structured data (tables) from PDF files, making it easy to pull information from scanned or formatted documents.

- **Prerequisites:** Requires access to PDF files.
- **Capabilities Supported:** Import mode, table extraction.
- **Limitations:**
  - Only basic table structures are accurately extracted
  - Complex layouts may require manual adjustments
  - Not all PDFs are parsed correctly, especially for multi-page or complexly formatted documents
  - Performance decreases with larger files

Web Content

Power Query allows users to retrieve data from web pages or APIs, supporting dynamic web content extraction.

- **Prerequisites**: Public or API URL; for API, may require authentication.
- **Capabilities Supported**: Import mode, handling HTML and JSON content.
- **Limitation**:
  - Performance depends heavily on website or API response time.
  - Parsing HTML structures can be inconsistent, depending on the site's format.
  - Websites with dynamic content (JavaScript-generated data) are not supported unless APIs are used.

Connecting to SQL Database and Other Connectivity Strategies

Azure SQL

Azure SQL connector provides access to cloud-based SQL databases, allowing for both import and DirectQuery modes in Power BI.

- **Prerequisites:** Requires an Azure SQL Database and connection credentials (login, password, or AAD authentication).
- **Capabilities Supported:** Import mode or DirectQuery providing query folding.
- **Limitations:**
  - Complex queries in DirectQuery mode can affect performance
  - Query folding may not occur for advanced transformations
  - Complex joins and nested queries may perform slowly under DirectQuery. It's recommended to optimize SQL queries at the database level

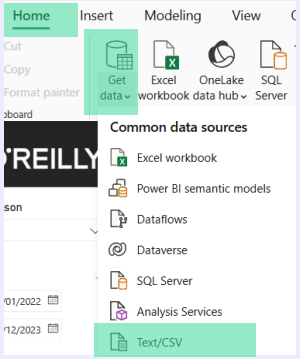
Exercise 2

Load new files

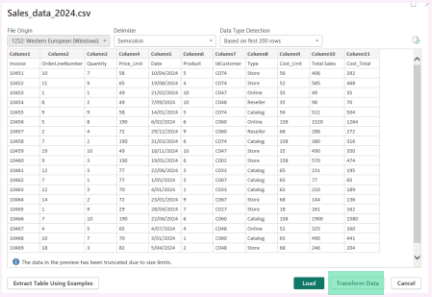
Exercise 2: Load new files

Goal: *Learn to load different types of files.*

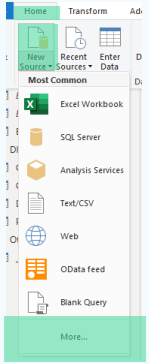
1 **Load 'Sales_ data_2024' table:** Select *Home,* ***later*** *Text/SCV*



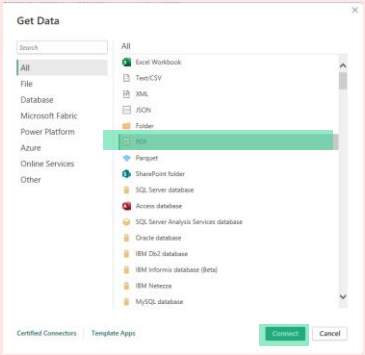
2 Select the path where the file **Sales_data_2024** is stored and click Open, then *Transform Data*.



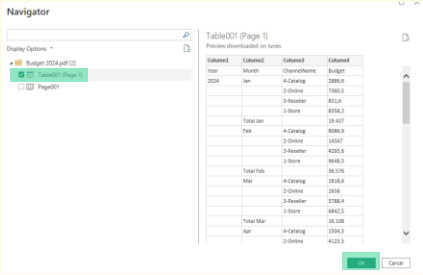
3 **Load 'Budget_2024'** Select *Home, New Source,* and *More*



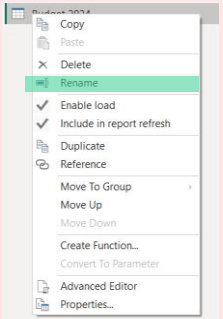
4 In window *Get Data*, select *PDF* and later *Connect* button



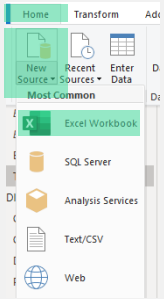
5 Browse to the location where the file **Budget_2024.pdf** is stored and click Open. After that, check 'Table 001 (page 1)' and click OK.



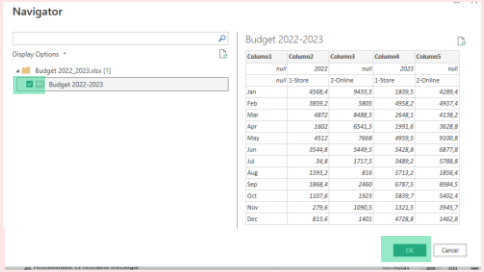
6 Change the table name from 'Table 001 (page 1)' to 'Budget 2024'



7 **Load 'Budget_2022_2023'** Select *Home, New Source/Excel Workbook*



8 Select the path where the file Budget_ 2022_2023 is stored and click *Open*. Check 'Budget 2022-2023', and *Ok* button



DEMO 03

- Organizing Queries

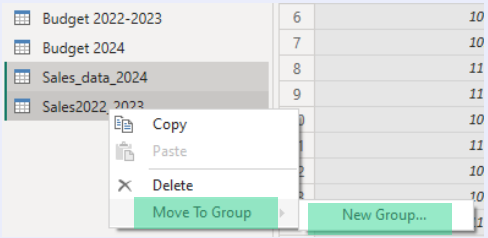
Exercise 3

Organizing queries

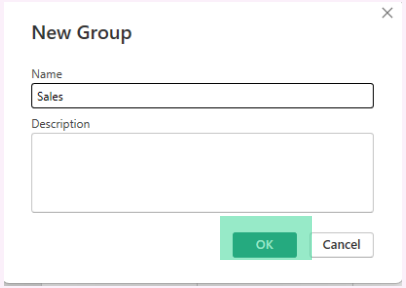
Exercise 3: Organize the project, before starting to work

Goal: *Organize the project tables into folders.*

1
- Go to Home > Transform Data
- Hold down **Ctrl** and select 'Sales 2022_2023' and 'Sales_data_2024'
- Right-click, choose Move to Group > New Group.

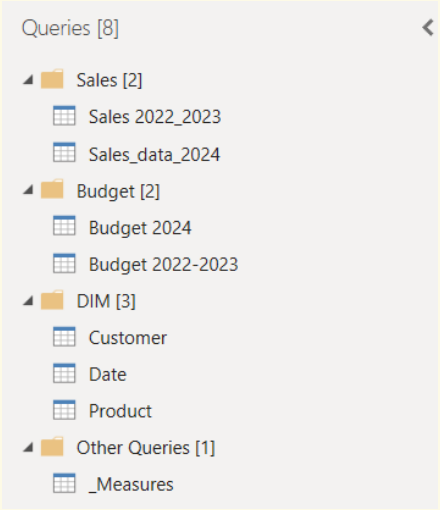


2 Name the folder 'Sales' and click *OK*



Automatically, those tables will be move in the new folder 'Sales'

3
- Select the queries 'Budget 2022-2023' and 'Budget 2024'.
- Repeat Step 1 and 2, and rename the new folder to 'Budget'.
- Next, select the tables Customer, Date, and Product.
- Create a new folder and name it 'DIM'.



Break time

Getting Started with M

The powerful M-language

- **Power Query is often seen as a no-code or low-code tool.**

- However, **to build more complex queries** and fully leverage Power Query's capabilities, **understanding its formula language, M, is essential.**

What is M?

- A **Functional language** that is widely used across **Microsoft platforms.**
- Integrated into **Power BI Desktop, Excel, Dataflows, Power Platform, SQL Server, and Dynamics.**
- Allows you to **extract, transform and load data** from multiple sources like databases, Excel files, and web services.

M in Power Query (Power BI Desktop)

- During this training, **we will focus on how M operates within Power Query in Power BI Desktop.**

- Every action performed through the UI (merging tables, filtering, etc.) generates M code in the background.

Flexibility, Customization & Efficiency with M

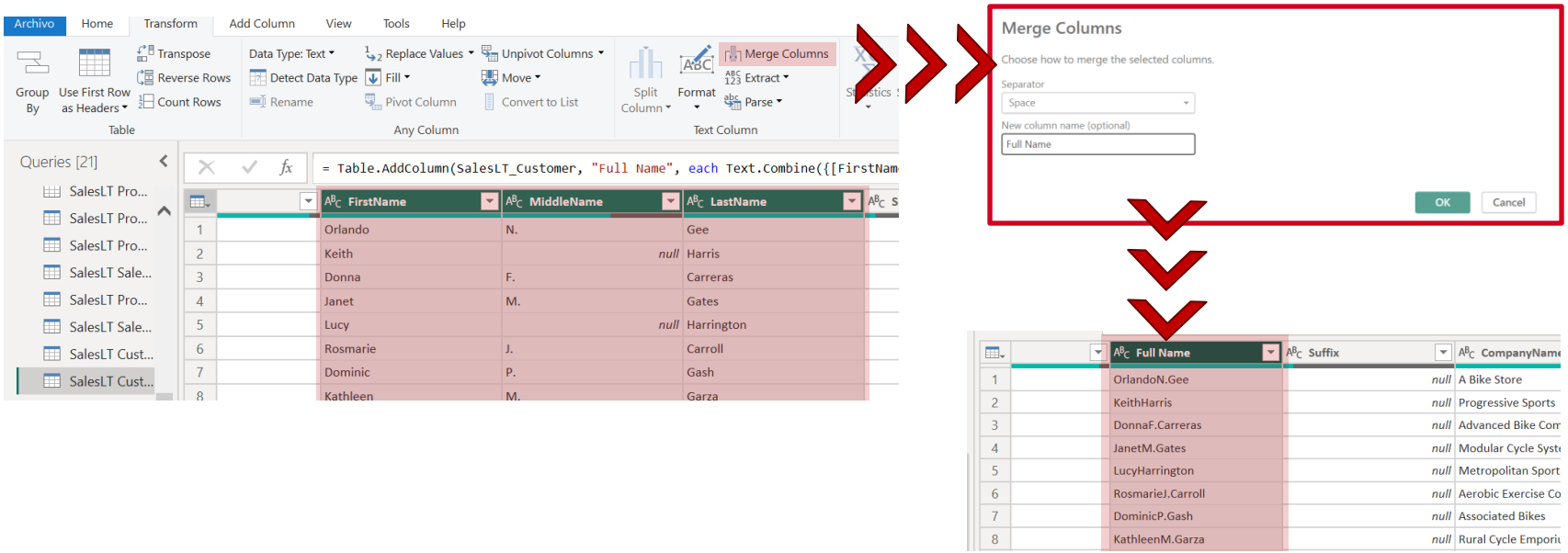
Why Learn M?

- **Tailor data transformations to meet specific needs.**
- Writing efficient M code can **significantly improve the performance of your queries.**

Understanding the M Code Behind Your Steps (COPILOT)

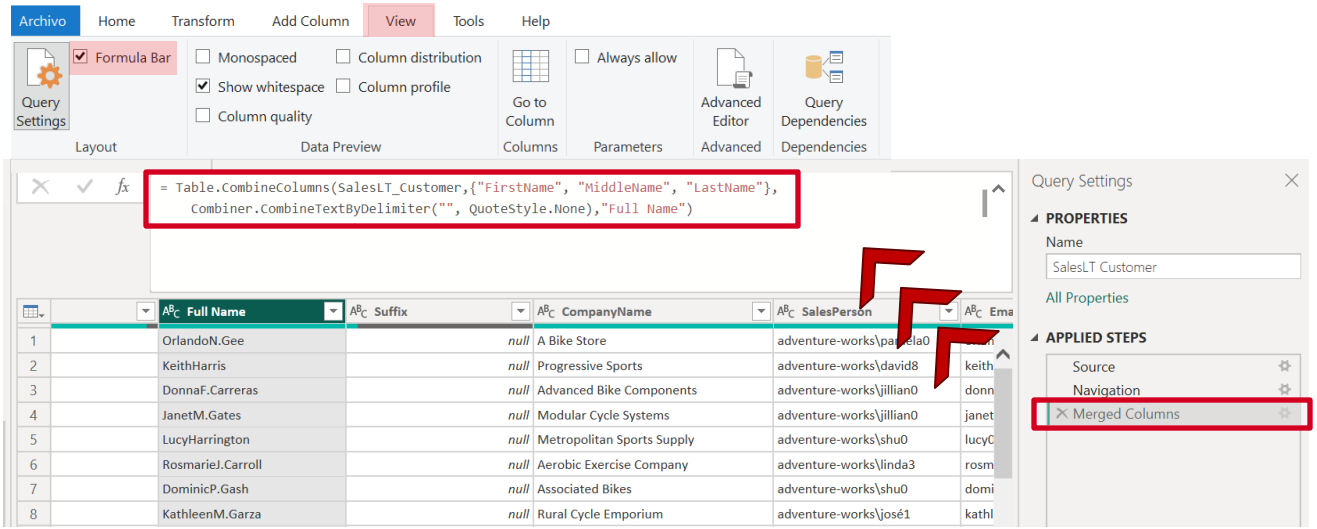
Experience-Generated Code (UI) in Power Query

Power Query's UI enables us to perform complex data transformations, with just a few clicks—no coding required.



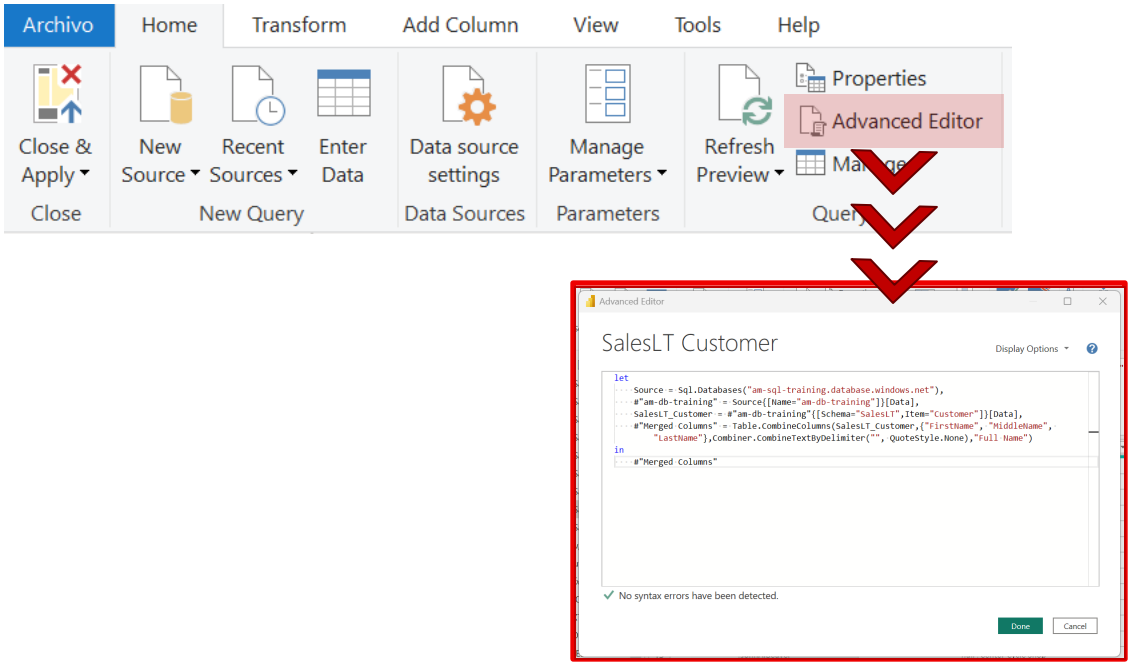
The M Code Behind Your Steps

Each step in Power Query generates a corresponding variable in M code.



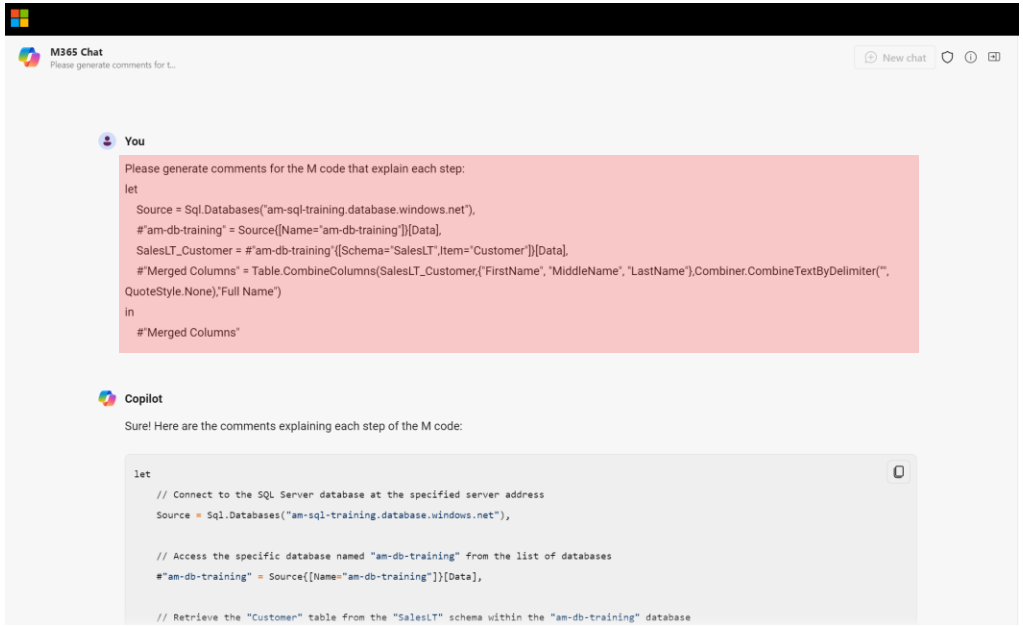
Advanced Editor

To view the full M code generated by your query steps, **click on 'Advanced Editor'** in Power Query. This allows you to see and modify the code behind the transformations , giving us flexibility to perform advanced transformations and customize our data processing.



Copilot to comment your code

Adding comments to your code is crucial for understanding and maintaining your queries, especially in complex transformations.

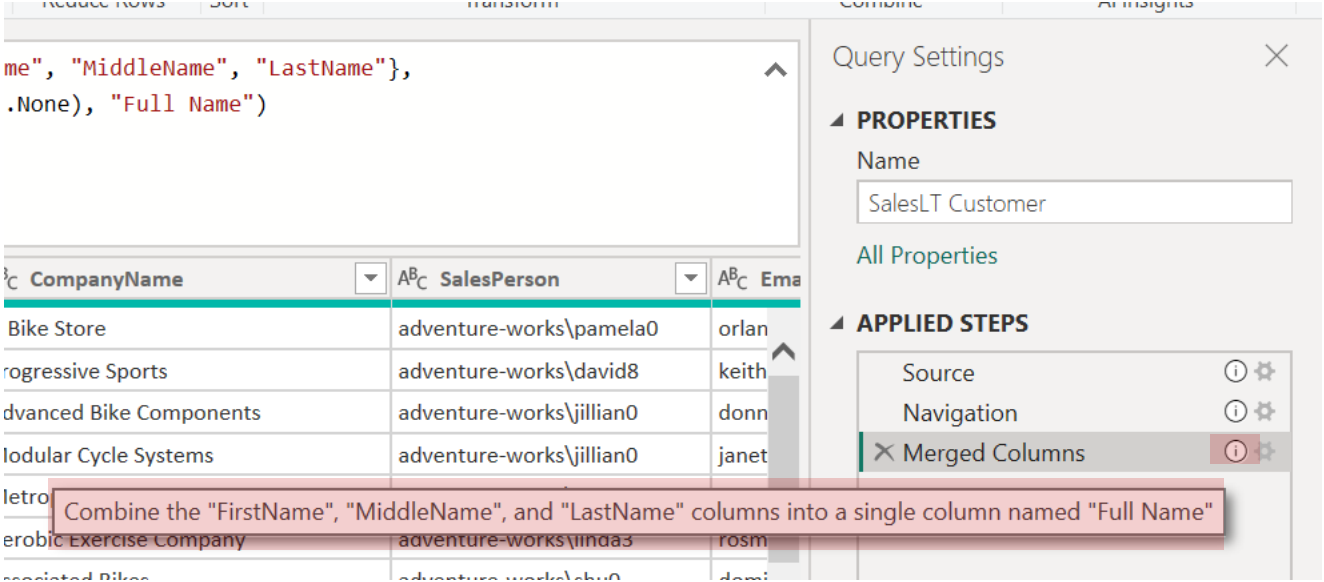


You can ask Copilot to generate comments for you:

1. Copy the code from the Advanced Editor
2. Request Copilot to add comments
3. Paste the updated code back into the Advanced Editor

Copilot to comment your code

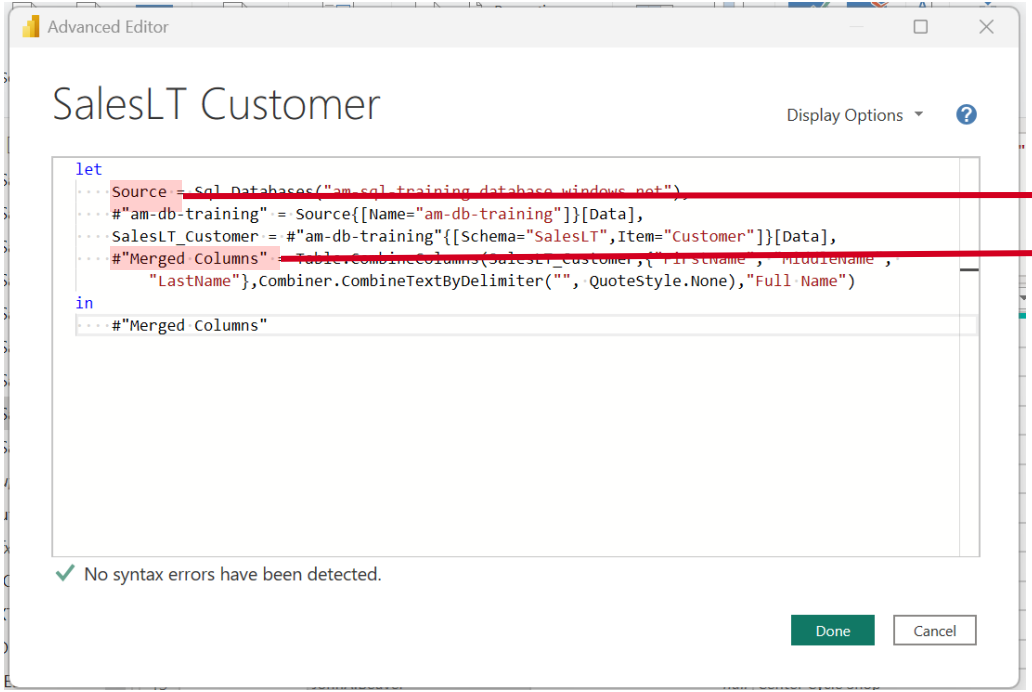
You can **view the comments for each step** by clicking the **'i' (information)** icon next to each step.



Effective Naming and Commenting in M

Naming the steps

Using clear, descriptive step names—whether with or without spaces—improves the understanding and maintenance of your code, especially in complex queries.

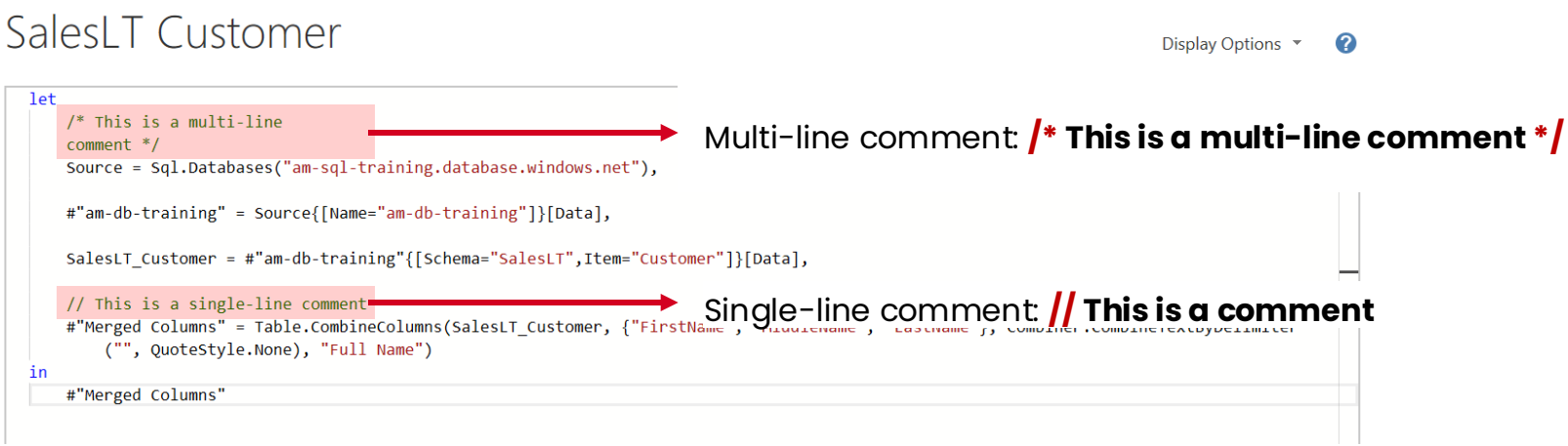


Without spaces:

With spaces: **#"Step With Spaces"**

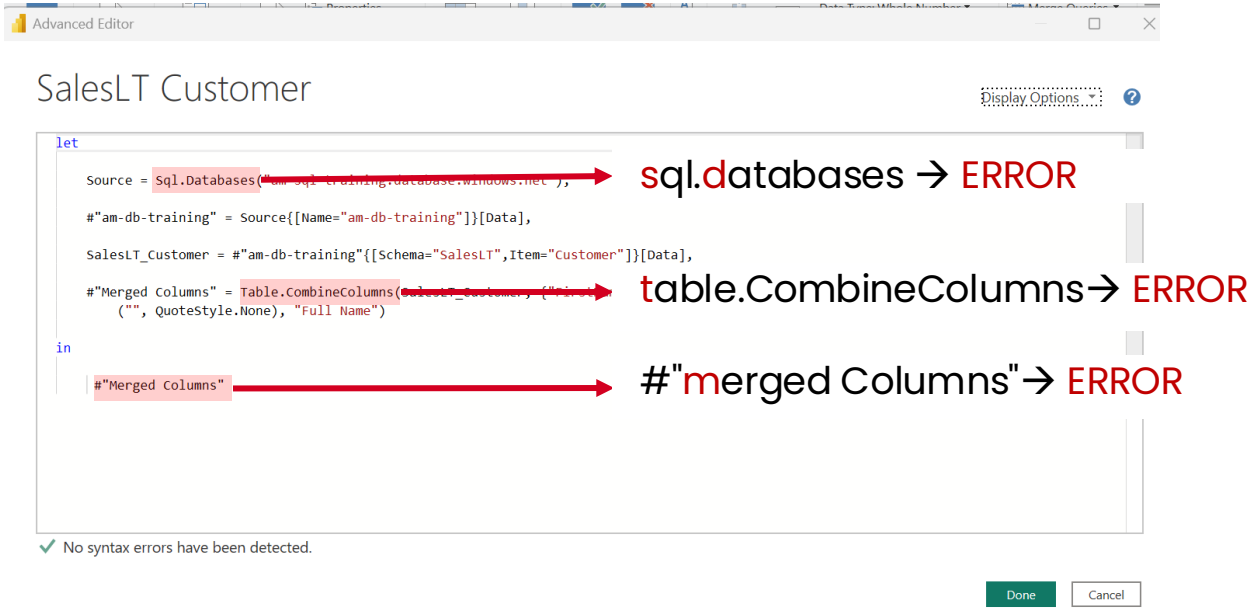
How to Comment in M

Use comments to make your M code easier to understand and maintain.



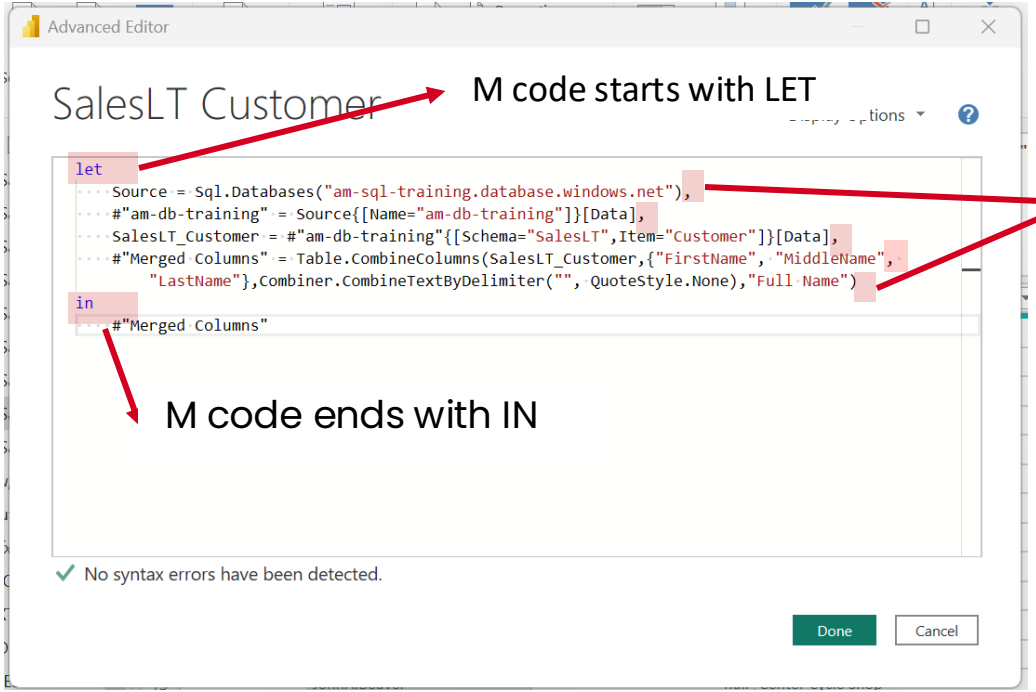
Case Sensitivity in M

M is case-sensitive, meaning 'Step1' and 'step1' are treated as two separate variables. This applies to all functions, expressions, and variable names.



Exploring the M Code Structure

Commas & Structure



Every step in M ends with a comma, except the last one.

M Steps as Variables

Each step is a variable that stores the result of a calculation or transformation.
These steps can return tables, records, or values, which can be reused in subsequent steps.

SalesLT Customer

Display Opti

```
let
    Source = Sql.Databases("am-sql-training.database.windows.net"),
    #"am-db-training" = Source{[Name="am-db-training"]}[Data],
    SalesLT_Customer = #"am-db-training"{[Schema="SalesLT",Item="Customer"]}[Data],
    #"Merged Columns" = Table.CombineColumns(SalesLT_Customer, {"FirstName", "MiddleName", "LastName"}, Combiner.CombineTextByDelim
        ("", QuoteStyle.None), "Full Name")
in
    Source
```

	Name	Data	Kind
1	am-db-training	Table	Database

SalesLT Customer

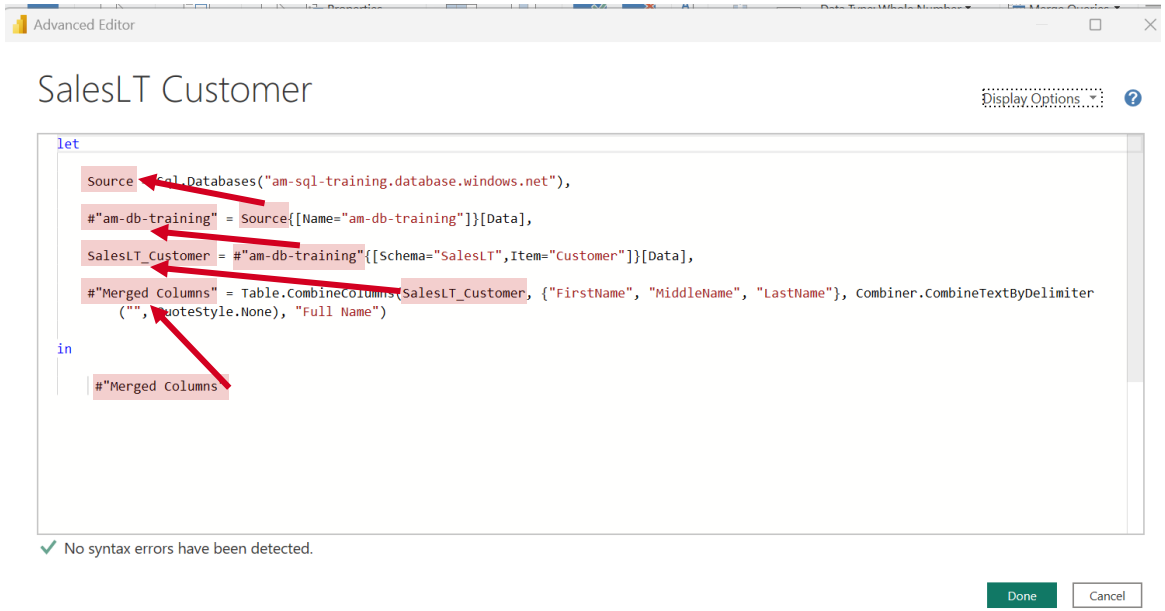
Display Op

```
let
    Source = Sql.Databases("am-sql-training.database.windows.net"),
    #"am-db-training" = Source{[Name="am-db-training"]}[Data],
    SalesLT_Customer = #"am-db-training"{[Schema="SalesLT",Item="Customer"]}[Data],
    #"Merged Columns" = Table.CombineColumns(SalesLT_Customer, {"FirstName", "MiddleName", "LastName"}, Combiner
        ("", QuoteStyle.None), "Full Name")
in
    #"Merged Columns"
```

	CustomerID	NameStyle	Title	Full Name	Suffix	
1	1	FALSE	Mr.	Orlando N. Gee	null	
2	2	FALSE	Mr.	Keith Harris	null	
3	3	FALSE	Ms.	Donna F. Carreras	null	
4	4	FALSE	Ms.	Janet M. Gates	null	
5	5	FALSE	Mr.	Lucy Harrington	null	
6	6	FALSE	Ms.	Rosmarie J. Carroll	null	
7	7	FALSE	Mr.	Dominic P. Gash	null	
8	10	FALSE	Ms.	Kathleen M. Garza	null	

Step Order & Dependencies

The IN statement references the output step, which is typically the final transformation. **This step can reference earlier steps,** and those steps can further reference others**, creating a chain of dependencies.**



QUIZ TIME

Let's do a quick quiz to reinforce our knowledge.

Question 1

Which of the following statements is true about Power Query's formula language, M?

A) M is only used for simple data transformations and cannot handle complex queries.

B) M is essential for creating custom functions and advanced data transformations in Power Query.

C) M is a programming language that is only used for data visualization.

D) M is automatically generated by Power Query, so users never need to interact with it.

Question 2

Why should you use comments in your M code in Power Query?

A) To improve the performance of your queries

B) To prevent errors in the query

C) To make your M code easier to understand and maintain

D) To automatically generate the final results of the query

Question 3

Which of the following M code fragments contains an error based on the specified rules?

A.

```
LET
    x = 5,
    y = x + 3,
IN
    y
```

B.

```
LET
    x = 5,
    y = x + 3,
    z = y * 2
IN
    z
```

C.

```
LET
    x = 5
    y = x + 3,
IN
    y
```

DEMO 04

- Promote headers
- Fill
- Unpivot
- Transposing
- Merge columns
- Filter
- Replace

Combining Queries

- Business analysts need to perform complex transformations that usually involve **a combination of multiple queries.**
- Combining queries is essential for performing complex transformations in scenarios where different tables or files store fragmented pieces of information.
- Users typically need to **merge data horizontally** to enrich a table with **additional columns** that are not available within the main query when it is loaded from a data source or append tables to gather all relevant information.

- In **Power Query** you can transform data and also combine queries in two ways:

Merge

Allows users to combine two or more queries by matching records based on one or more common fields (keys), effectively adding new columns from another table.

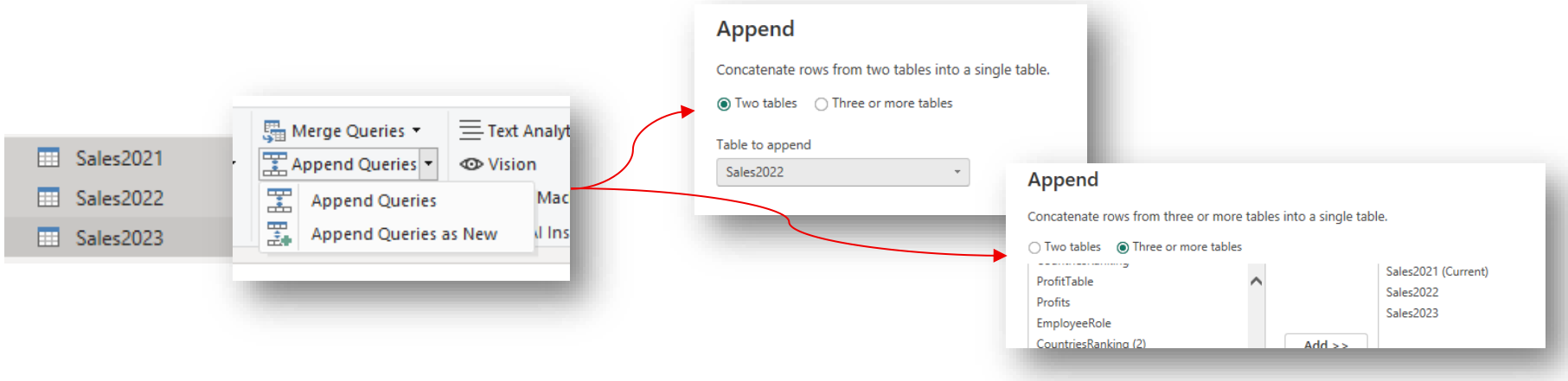
Append

Enables users to **combine data vertically** by stacking rows from different tables into one. It is especially useful when datasets share the same structure but contain different periods, locations, or categories.

Appending Queries

- Appending queries enables users to combine data vertically, stacking multiple tables into one.
- Use Case:
  - Appending is used when you need to combine datasets with the same structure (columns).

Example: appending monthly or yearly sales reports from different regions into one master file. Each region's data has the same columns, and appending helps you consolidate these datasets.



Unpivoting and Transposing

- Pivoting, unpivoting, and transposing data are powerful methods for reshaping tables in Power Query.
- Use Case:
  - **Unpivoting** is useful when you have column headers that contain values, and you need to convert them into rows.

For instance, transforming columns like "January Sales" and "February Sales" into a single "Month" column with a corresponding "Sales" column.

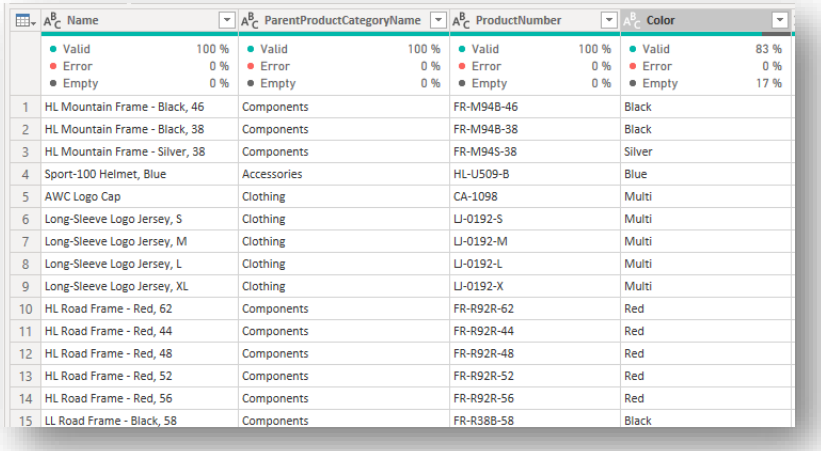
  - **Transpose** is helpful when you need to switch rows and columns.

Example: A wide table of monthly sales per city can be unpivoted to turn months into rows for a cleaner time-series analysis.

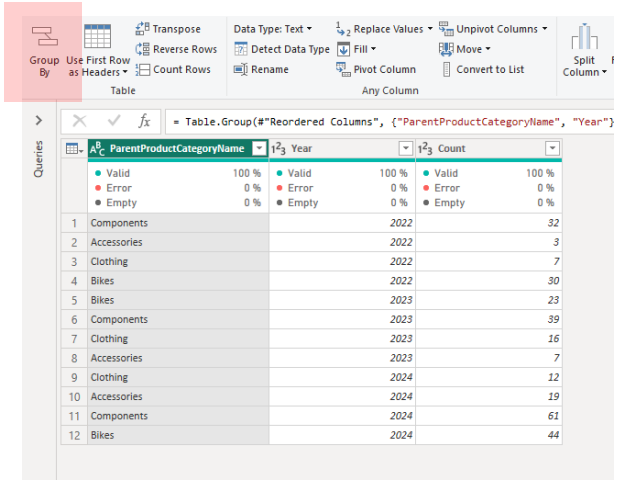
Grouping Data

- Grouping data is a key functionality for summarizing and analyzing **large datasets**.
- Use Case:
  - Summarize data by specific categories, like summing up total sales by region or counting the number of orders by customer.

Example: If you have a dataset with daily transactions, you could group by "CustomerID" to get the total spending per customer, or by "Region" to summarize sales per region.



	Name	ParentProductCategoryName	ProductNumber	Color
	Valid 100 %, Error 0 %, Empty 0 %	Valid 100 %, Error 0 %, Empty 0 %	Valid 100 %, Error 0 %, Empty 0 %	Valid 83 %, Error 0 %, Empty 17 %
1	HL Mountain Frame - Black, 46	Components	FR-M94B-46	Black
2	HL Mountain Frame - Black, 38	Components	FR-M94B-38	Black
3	HL Mountain Frame - Silver, 38	Components	FR-M94S-38	Silver
4	Sport-100 Helmet, Blue	Accessories	HL-U509-B	Blue
5	AWC Logo Cap	Clothing	CA-1098	Multi
6	Long-Sleeve Logo Jersey, S	Clothing	LJ-0192-S	Multi
7	Long-Sleeve Logo Jersey, M	Clothing	LJ-0192-M	Multi
8	Long-Sleeve Logo Jersey, L	Clothing	LJ-0192-L	Multi
9	Long-Sleeve Logo Jersey, XL	Clothing	LJ-0192-X	Multi
10	HL Road Frame - Red, 62	Components	FR-R92R-62	Red
11	HL Road Frame - Red, 44	Components	FR-R92R-44	Red
12	HL Road Frame - Red, 48	Components	FR-R92R-48	Red
13	HL Road Frame - Red, 52	Components	FR-R92R-52	Red
14	HL Road Frame - Red, 56	Components	FR-R92R-56	Red
15	LL Road Frame - Black, 58	Components	FR-R38B-58	Black



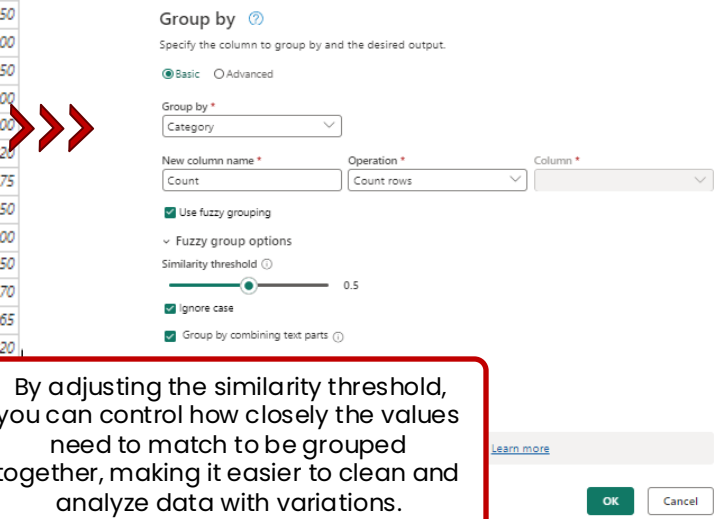
	ParentProductCategoryName	Year	Count
	Valid 100 %, Error 0 %, Empty 0 %	Valid 100 %, Error 0 %, Empty 0 %	Valid 100 %, Error 0 %, Empty 0 %
1	Components	2022	32
2	Accessories	2022	3
3	Clothing	2022	7
4	Bikes	2022	30
5	Bikes	2023	23
6	Components	2023	39
7	Clothing	2023	16
8	Accessories	2023	7
9	Clothing	2024	12
10	Accessories	2024	19
11	Components	2024	61
12	Bikes	2024	44

Fuzzy grouping in Power Query ONLINE

ONLY

- Fuzzy grouping in Power Query (Only ONLINE) allows you to group similar but not exactly identical text values. It helps in scenarios where you need to consolidate data with slight differences, such as misspelled categories or product names.

	ProductID	ProductName	Category	Price
1	1	Blue Mountain Bike	Bicycles	450
2	2	Mountain Bike	Bicycle	480
3	3	Mountain Bike Pro	Bicycles	550
4	4	Road Bike	Bikes	300
5	5	Red Road Bike	Road Bicycle	350
6	6	BMX Bike	BMX	200
7	7	Off-road Bike	Offroad Bicycles	500
8	8	Classic Road Bike	Bikes	320
9	9	Cruiser Bike	Bycicle	275
10	10	Trail Bike	Bickles	450
11	11	Speed Bike	Speed Cycle	400
12	12	Offroad Mountain	Off-Road Bicycles	550
13	13	Hybrid Bike	Hybrid Bicycles	370
14	14	Hybrid Bicycle	Hybrids	365
15	15	Touring Bike	Touring Bicycles	420
16	16	Smartphone	Electronics	
17	17	Mobile Phone	Electronics	
18	18	Laptop	Computers	
19	19	Notebook	Electronics	
20	20	Gaming Laptop	Electronic Devices	
21	21	Tablet	Electronics	
22	22	Desktop PC	Computers	900
23	23	Smartwatch	Wearable Devices	300
24	24	Fitness Tracker	Wearables	250



By adjusting the similarity threshold, you can control how closely the values need to match to be grouped together, making it easier to clean and analyze data with variations.

	Category	Count
1	Bicycles	12
2	Hybrid Bicycles	2
3	Electronics	5
4	Wearable Devices	2
5	Computers	2
6	BMX	1

	Category	Count
1	Bicycles	15
2	Electronics	5
3	Wearable Devices	2
4	Computers	2

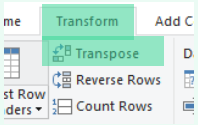
Exercise 4

Applying M language: *Working with a budget table*

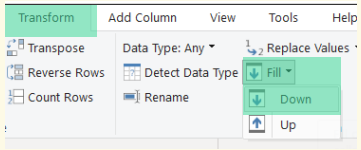
Exercise 4: Applying Unpivoting and Transposing Data

Goal: *Learn how to use the Unpivoting and Transposing function.*

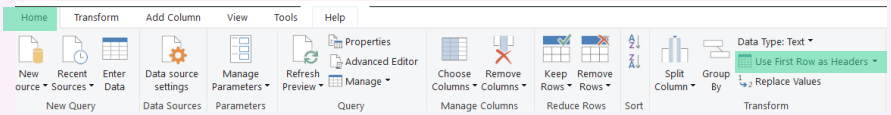
1 Go to Home > Transform data
Select '*Budget 2022_2023*'.
Click on *Transform* and then *Transpose*



2 Select 'Column 1', click on *Transform* > *Fill* > *Down*

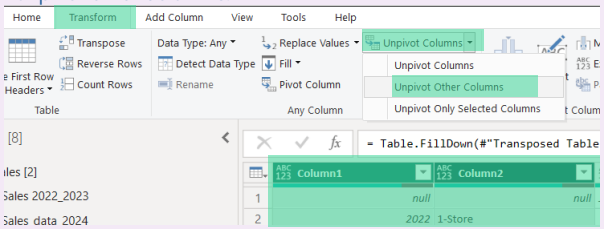


3 Select *Home* and *Use First Row as Headers*



4 **Apply unpivot:**

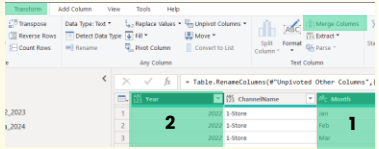
Select the columns 1 and 2, Click on Transform > Unpivot columns > Unpivot other columns.



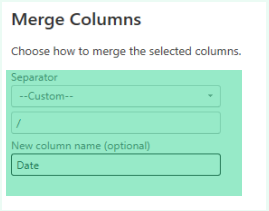
5 **Rename the columns as follows:**

- Column1 → Year
- Column2 → ChannelName
- Attribute → Month
- Value → Budget

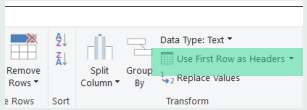
6 **Merge Columns**: Select first column 'Month' and then 'Year'. *Select Transform/Merge Columns.*



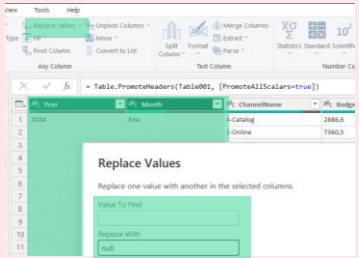
7 Separator = Custom
Type '/'
New column = Date

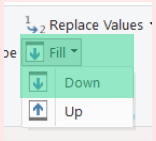


8 **Select '*Budget 2024*'.**
Click on *Home* and *Use First Row as Headers.*

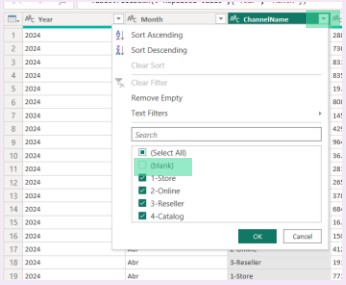


9 **a.** Select 'Year' and 'Month'. *Transform* / *Replace Values*
b. In the *Replace Values* Window, write 'null' in the *Replace With* field
c. Fill Down





10 **Remove blank** values from the **'Channel name' column:**



11 **Repeat step 6 and 7, now for query 'Budget 2024'.**

O'REILLY®

Class 02

Course Agenda

Class 1: Introduction and getting started with M

- Course Framework and Introduction
- Data Sources and Connectivity
- Getting Started with M: A Step Beyond the UI
- Combining data

Class 2: Important tools in the M language

- Understanding Object Structures and Data Types.
- Intermediate Data Transformation
- Advanced Query Techniques and Error Handling
- Final comments and Advanced tips

Our Method on each topic:
Demo example, followed by exercises to practice!

- **DEMO – Flor & Nico**

- **EXERCISE – Your turn!**

*Student should have the latest version of **Power BI Destkop installed.**

Instructors

Follow us on LinkedIn:

Nicolás Lagreste Zucchini

María Florencia Hourcouripé

Let's Collaborate: Interested in a personalized mentorship or consulting session? Contact us:

- nico@analyticmood.com
- flor@analyticmood.com

Visit Our Website:

- www.analyticmood.com for resources, upcoming events, and blog articles.

DEMO 05

- Append

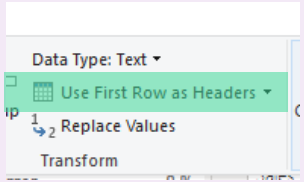
Exercise 5

Organizing and Applying Append

Exercise 5: Append process

Goal: *Learn how to use the append function*

1 Go to Home > Transform data
Select '*Sales_ data_2024*'.
Home > Use first rows as headers:

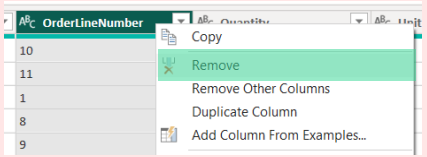


2 **Compare the queries 'Sales_data_2024' and 'Sales 2022_2023 ' to ensure that the columns are matching:**

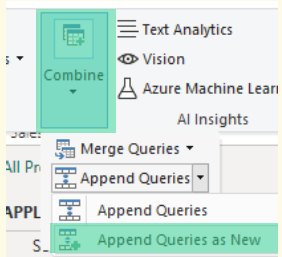
a In the table '**Sales_data_2024',** rename the following columns to match those in Sales_2022_2023:

- Price_Unit → Unit Price
- Cost_Unit → Unit Cost
- Cost_Total → Total Cost
- Product → IdProduct

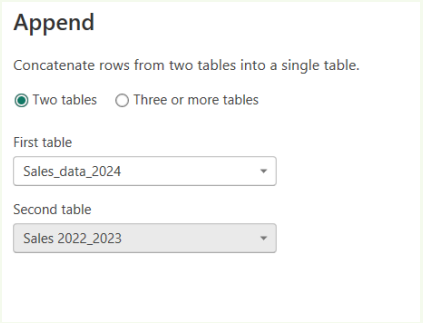
b Remove column 'OrderLineNumber': First select the column, later click right



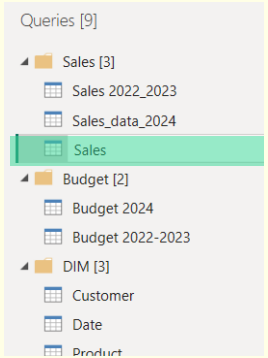
3 *Home, Combine,* and *Append query as a new*



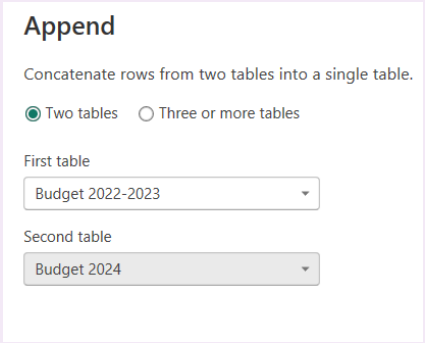
4 In *Append* Window select table 'Sales 2022_2023' as *Second Table*



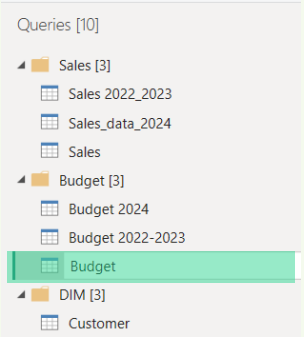
5 **Rename the new query to 'Sales'**



6 Repeat steps 3 and 4 to append 'Budget 2022_2023' and 'Budget_2024'.



7 **Rename the new query to 'Budget'**



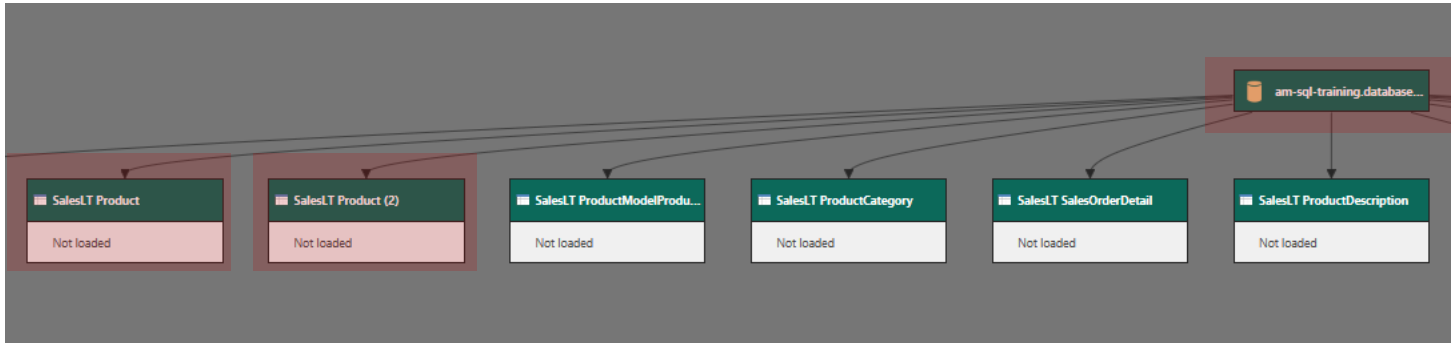
DEMO 06

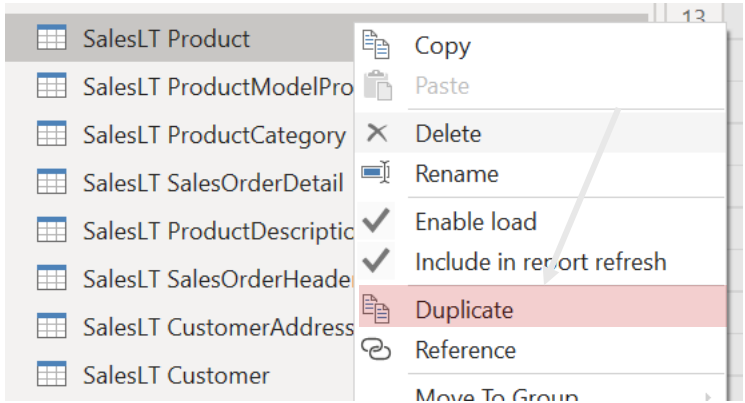
- Remove duplicates
- Reference
- Duplicates
- Split
- Rename
- Merge

References and Duplicates

Duplicates

Duplicate will duplicate the code of the query. It copies a query with all the applied steps of it as a new query.

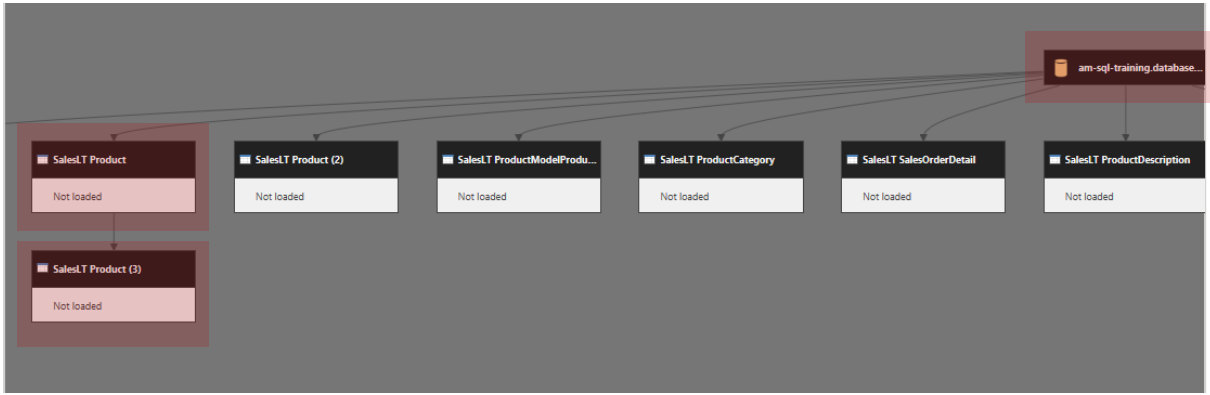


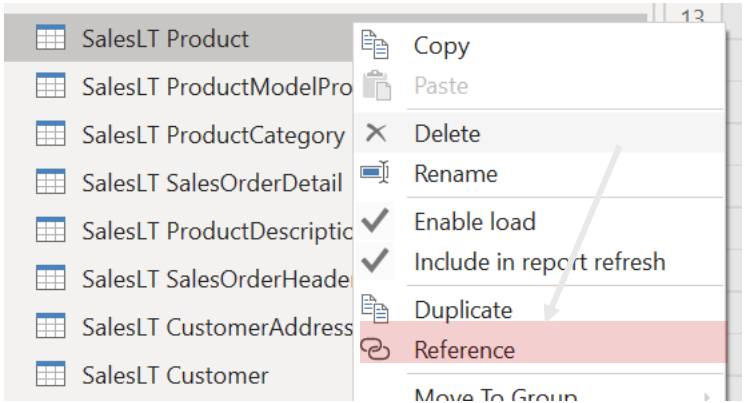


You can make changes in the original or the new query, and they will NOT affect each other.

References

When you reference a query, **the new query will have only one step: sourcing from the original query.**

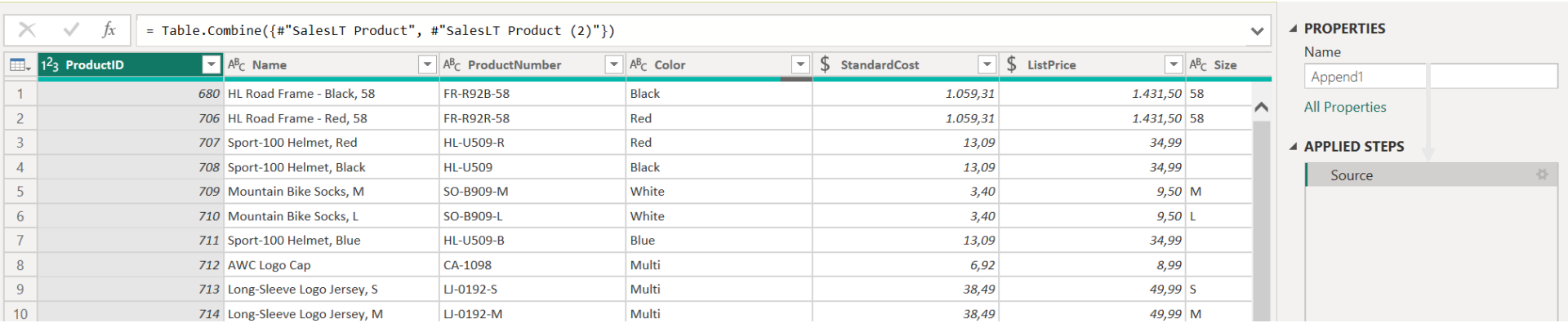




If you make a change in the original query, the new query will be impacted.

Other new queries

Append Queries as New / or Merge Queries as New is a Reference action



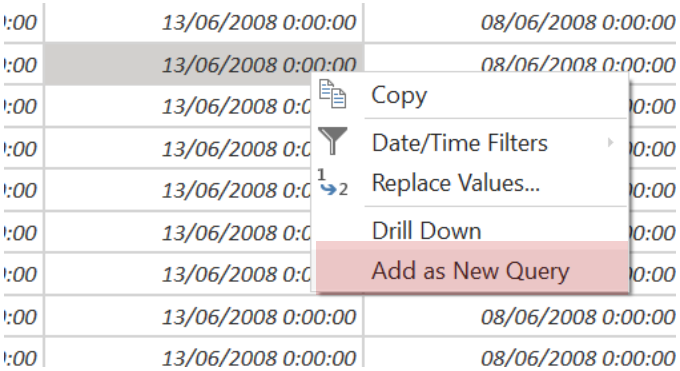
```
= Table.Combine({#"SalesLT Product", #"SalesLT Product (2)"})
```

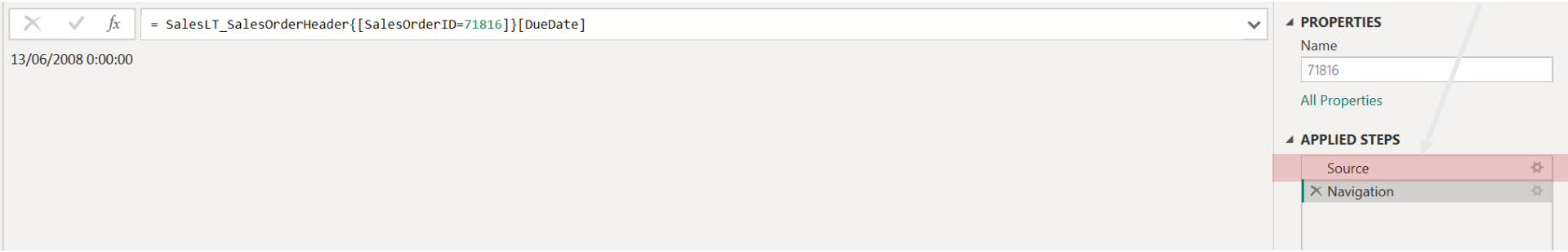
	ProductID	Name	ProductNumber	Color	StandardCost	ListPrice	Size
1	680	HL Road Frame - Black, 58	FR-R92B-58	Black	1.059,31	1.431,50	58
2	706	HL Road Frame - Red, 58	FR-R92R-58	Red	1.059,31	1.431,50	58
3	707	Sport-100 Helmet, Red	HL-U509-R	Red	13,09	34,99	
4	708	Sport-100 Helmet, Black	HL-U509	Black	13,09	34,99	
5	709	Mountain Bike Socks, M	SO-B909-M	White	3,40	9,50	M
6	710	Mountain Bike Socks, L	SO-B909-L	White	3,40	9,50	L
7	711	Sport-100 Helmet, Blue	HL-U509-B	Blue	13,09	34,99	
8	712	AWC Logo Cap	CA-1098	Multi	6,92	8,99	
9	713	Long-Sleeve Logo Jersey, S	LJ-0192-S	Multi	38,49	49,99	S
10	714	Long-Sleeve Logo Jersey, M	LJ-0192-M	Multi	38,49	49,99	M

* https://radacad.com/reference-vs-duplicate-in-power-bi-power-query-back-to-basics

Other new queries

Add as New Query is a Duplicate action





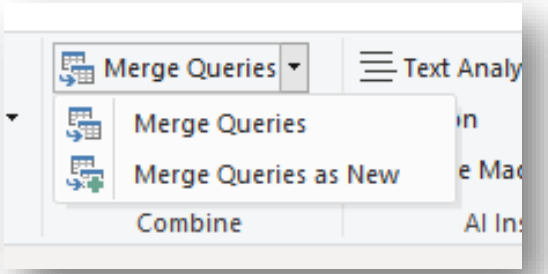
* https://radacad.com/reference-vs-duplicate-in-power-bi-power-query-back-to-basics

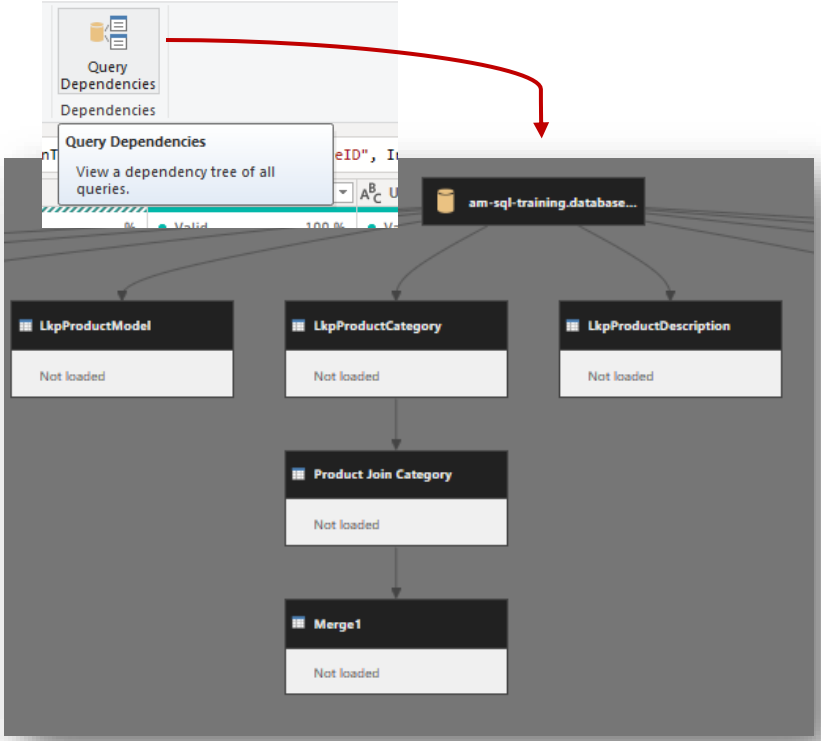
Merging Data from Multiple Sources

Combining Queries – Merge

- A Merge query combines two existing queries by matching rows based on a common column. This operation adds columns from the secondary table to the primary table.

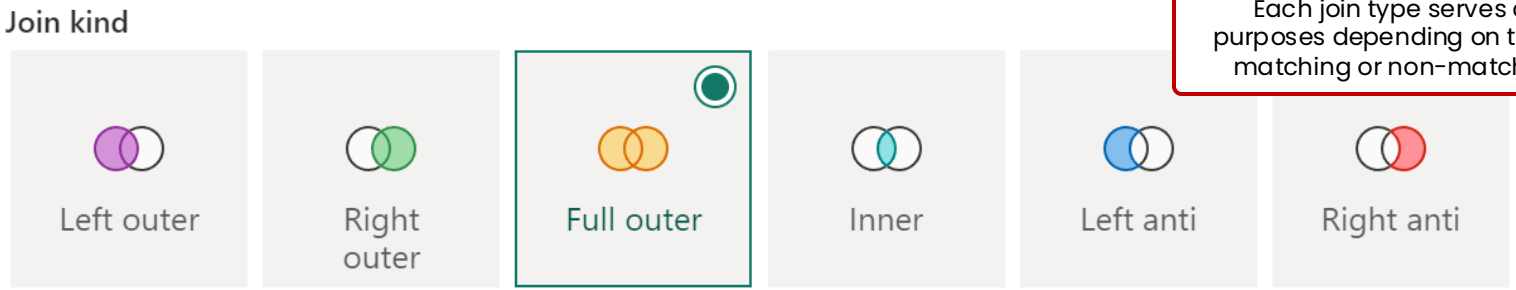
- There are two types:
  - **Inline Merge**: Merges data within the same query, creating a new step.
  - **Intermediate Merge**: Creates a new query for each merge operation.





Combining Queries – Merge

- There are several types of **joins** used to combine data from two tables based on common columns:

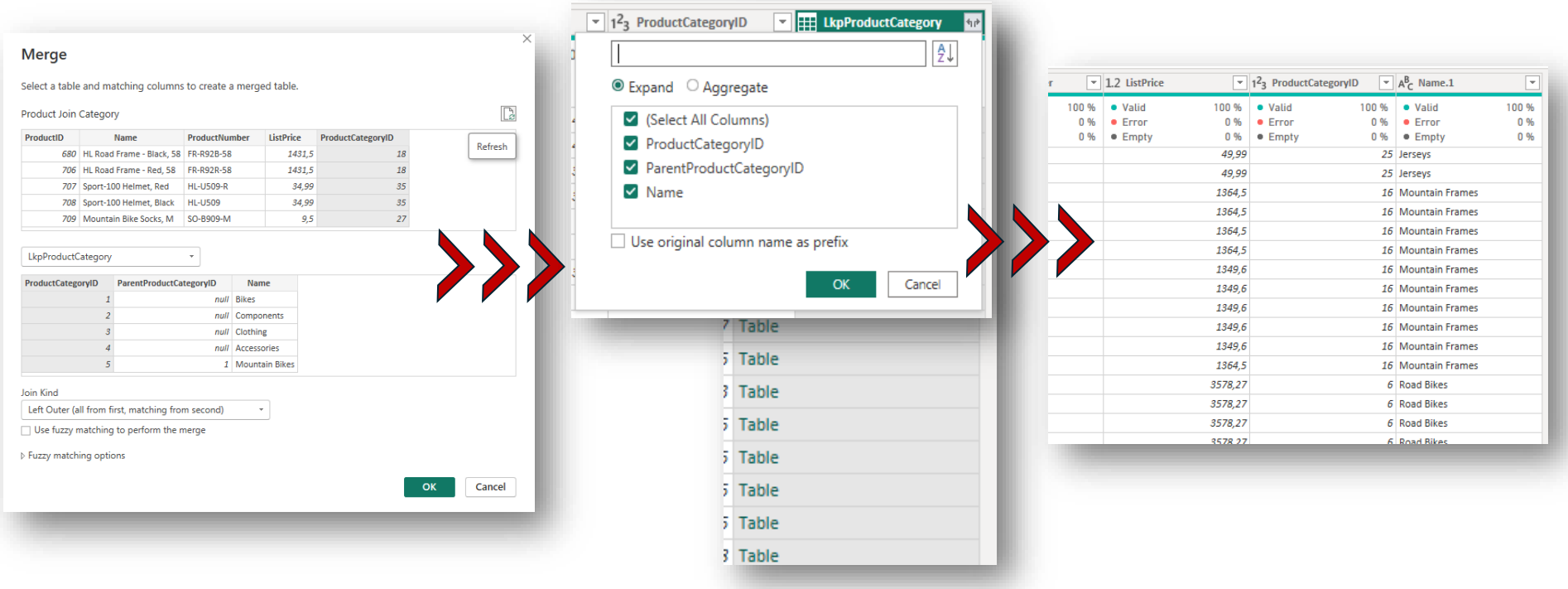


Each join type serves different purposes depending on the need for matching or non-matching rows.

1. **Left Outer**: Returns all rows from the first (left) table, and matching rows from the second (right) table. If no match is found, null values are returned for the right table's columns.
2. **Right Outer**: Returns all rows from the right table and matching rows from the left. Non-matching left table rows will return null.
3. **Full Outer** : Returns all rows from both tables. Where no match is found, null values are returned.
4. **Inner**: Returns only matching rows from both tables.
5. **Anti** (Left/Right): Returns rows from one table where no match is found in the other.

Combining Queries – Merge

- The related table contains all rows that match each row from a common column value in the primary table.



Merging Data - Semi Joins

Combining Queries – Left and Right Semi Join

- **Left Semi-Join** returns only the rows from the left table that have matching rows in the right table, but it doesn't bring in any columns from the right table. It effectively filters the left table to only include rows that exist in the right table based on the join column(s).
- **Right Semi-Join**: Keeps only rows from the right table that have matches in the left table, no columns from the left table are added.

- **Use Case:**
  - This is useful when you want to filter your data to only include records that have a match in another table, but you don't need any information from the matching table.

Combining Queries – Left and Right Semi Join

QUICK TIPS

In this example, we're using a **Left Semi Join** to return only the employees that have a corresponding department in the **Department** table.

Employee

	IdEmployee	Name	Age	City	IdDepartment
	Valid 100 % Error 0 % Empty 0 %	Valid 100 % Error 0 % Empty 0 %	Valid 100 % Error 0 % Empty 0 %	Valid 100 % Error 0 % Empty 0 %	Valid 100 % Error 0 % Empty 0 %
1	1	Alice	25	New York	8
2	2	Bob	30	Chicago	1
3	3	Carol	28	Boston	3
4	4	Dave	35	Seattle	6
5	5	Eve	22	Miami	10

Deparment

	Id	Department	Country
	Valid 100 % Error 0 % Empty 0 %	Valid 100 % Error 0 % Empty 0 %	Valid 100 % Error 0 % Empty 0 %
1	1	HR	USA
2	3	Marketing	USA
3	4	Finance	USA
4	6	IT	Canada
5	7	Sales	USA

```
= Table.NestedJoin(#"Changed Type", {"IdDepartment"}, Department, {"Id"}, "Department", JoinKind.LeftSemi)
```

	IdEmployee	Name	Age	City	IdDepartment	Department
	Valid 100 % Error 0 % Empty 0 %	Valid 100 % Error 0 % Empty 0 %	Valid 100 % Error 0 % Empty 0 %	Valid 100 % Error 0 % Empty 0 %	Valid 100 % Error 0 % Empty 0 %	Valid 100 % Error 0 % Empty 0 %
1	2	Bob	30	Chicago	1	Table
2	3	Carol	28	Boston	3	Table
			35	Seattle	6	Table

Left Semi Join keeps the rows from the left table (**Employee**) only where a match is found in the right table (**Department**), but it doesn't return the data from the right table.

Manual adjustment in the Power Query M code to use the JoinKind.LeftSemi

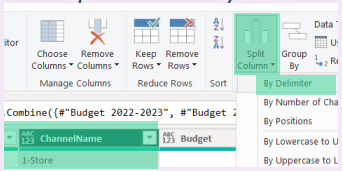
Exercise 6

References and Duplicates

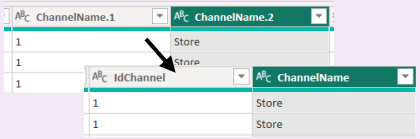
Exercise 6: Create Dimensions

Goal: *Learn a way to create dimensions*

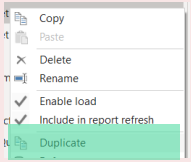
1 Go to Home > Transform data
Select *'Budget'*
Select 'ChannelName'
Home>Split Column>By Delimiter



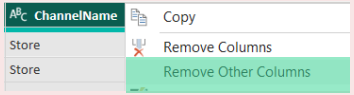
Rename Columns:
- ChannelName.1 → IdChannel
- ChannelName.2.→ ChannelName



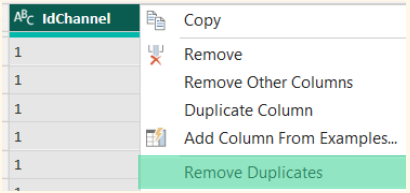
2 **Duplicate query 'Budget':**
Right-click on the name of query Budget / *Duplicate*



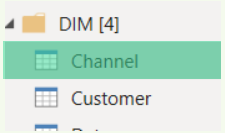
3 **Remove columns** 'Date' and 'Budget' using 'Remove other columns'



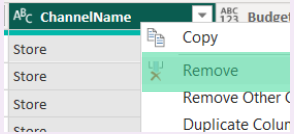
4 **Remove Duplicates:** Right-click on the column 'IdChannel'/*Remove Duplicates*



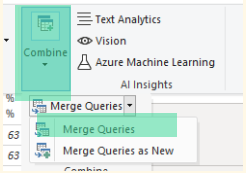
5 **Rename:** Rename the query to Channel and move the query to DIM folder



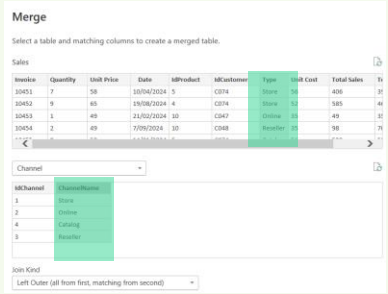
6 **Select 'Budget' and remove 'ChannelName'**



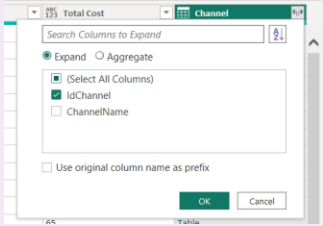
7 **Select 'Sales' and click 'Merge Queries"**



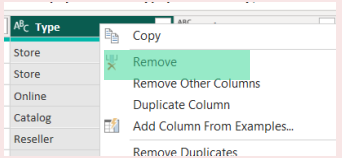
8 Select query 'Channel' and:
- Column 'Type' in Sales
- Column 'ChannelName' in 'Channel'.



9 Expand 'IdChannel'



10 Remove column 'Type'



DEMO 07

- Understanding Object Structures
- Data Types Examples
- Using Locale

Understanding Object Structures and Data Types

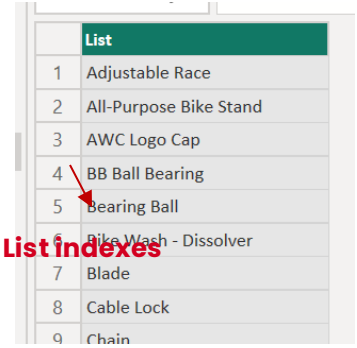
Understanding Object Structures and Data Types

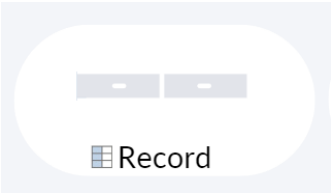
- **Structured/Abstract types**

  Power Query organizes data into **five distinct value structures.**

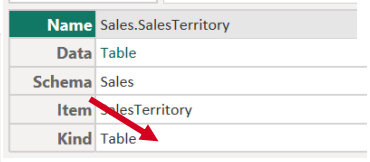
  This doesn't refer to the data type but rather to how the data itself is structured and stored.

A single columnar object with no column header



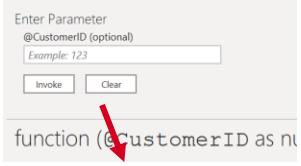


A row of a table with colu header



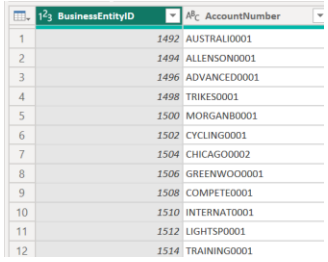
Record column headers

Pre-built capabilities that can be used to transform objects

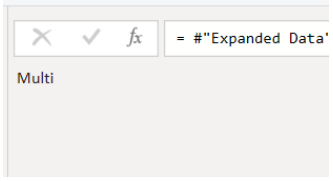


We use functions when we have a repeating process in data manipulation.

Combination of multiples row and multiple columns

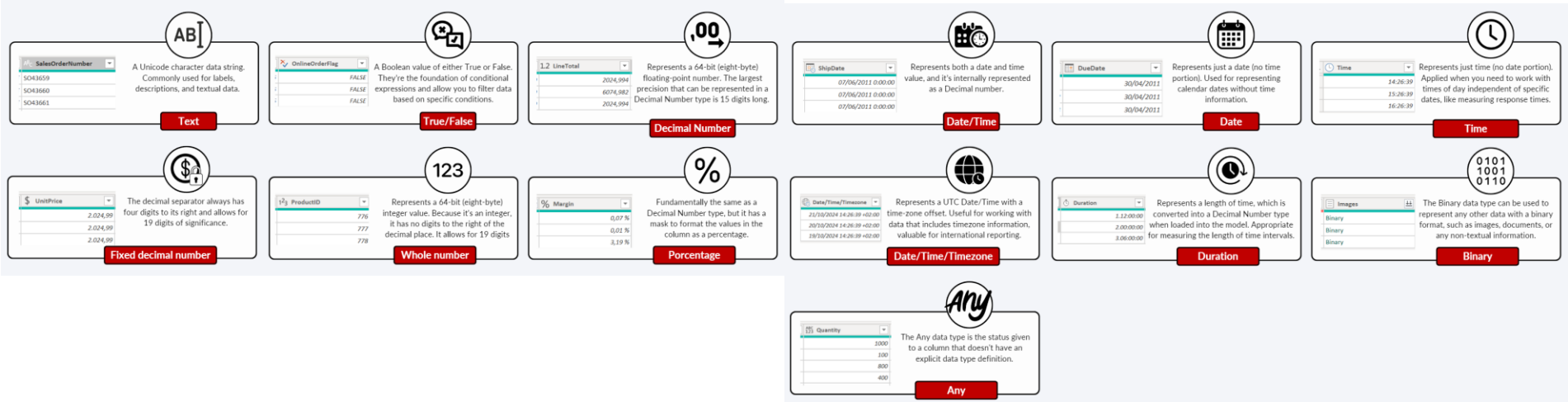


Any single part data type considered as a primitive value



Understanding Object Structures and Data Types

- **Primitive/Non-Abstract types**
  Data types define the **nature of the information contained within each column**, facilitating data organization and analysis.



Non-Abstract Types: Data Types

Data types

Data types define the **nature of the information contained within each column**, facilitating data organization and analysis.

Every column (or field) in a dataset is assigned a specific data type (such as text, number, date, etc.). All the values within that column must match or conform to this assigned data type.

Data types

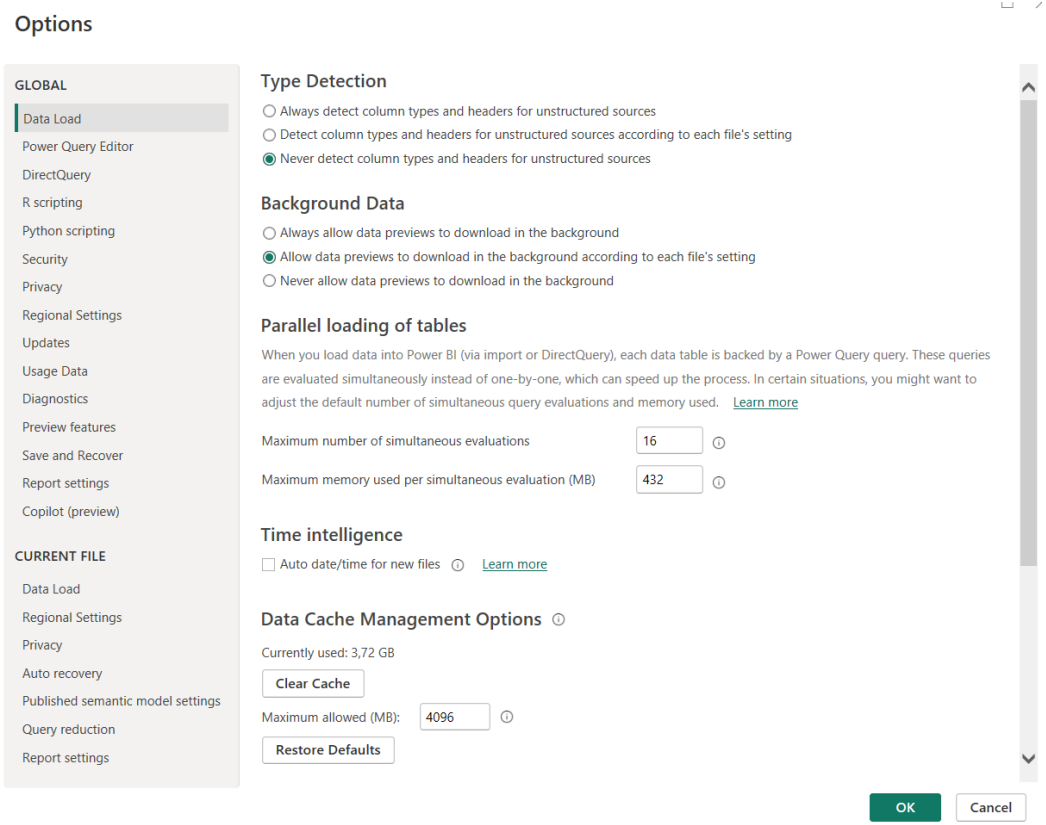
Primitive types are the fundamental data types from which all other data types are created.

1. Data types play an important role in **data validation** and **preventing errors**.
2. Defining data types can make your **queries run faster**. When you set types for columns, Power Query optimizes how it stores and retrieves that data, cutting down on memory use.
3. By specifying a data type, Power Query can **better handle errors**. If Power Query encounters a value that doesn't fit the assigned type, it either rejects it or converts it, depending on your expression.
4. Data types can be important when i**ntegrating with other systems or when exporting data.** Certain systems require specific data types for their operations or interfaces

Turn Off Automatically Detect Column Types

1. By default, Power Query automatically adds query steps that promote headers and change data types in your query after certain transformations are performed.

2. You can modify the Power Query options settings to prevent this from happening.



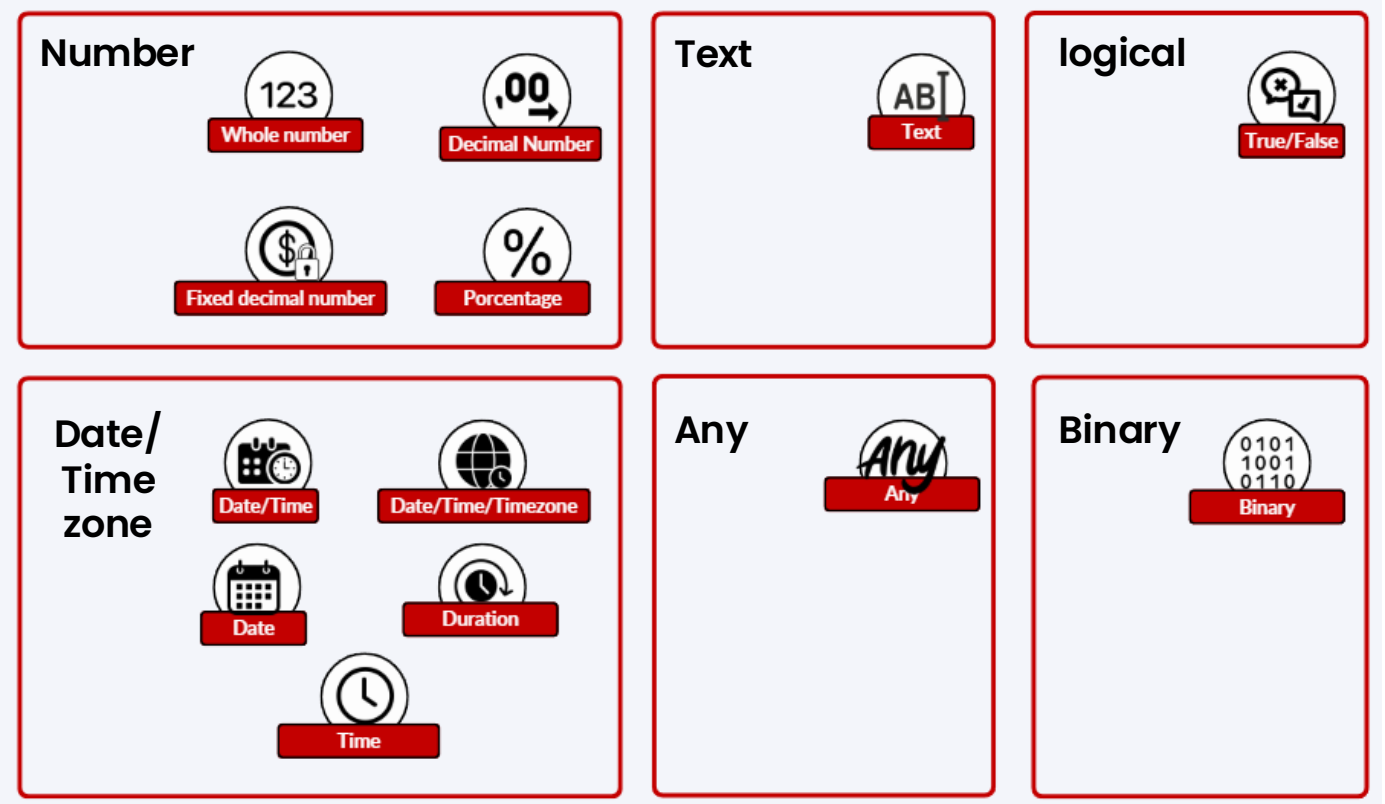
How to define a column data type

The data type of a column is displayed on the left side of the column heading with an icon that symbolizes the data type.

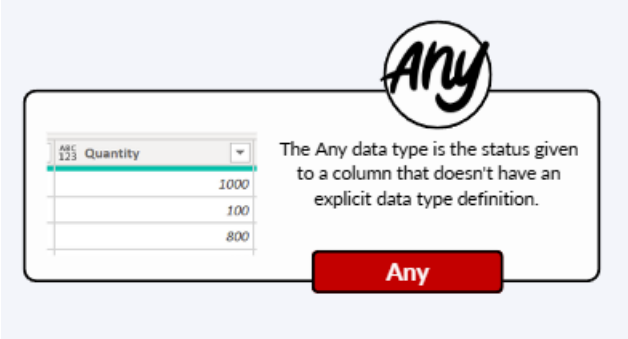
How to define a column data type

- Data type detection
- Transform / Data Type
- By selecting the icon on the left side of the column heading.

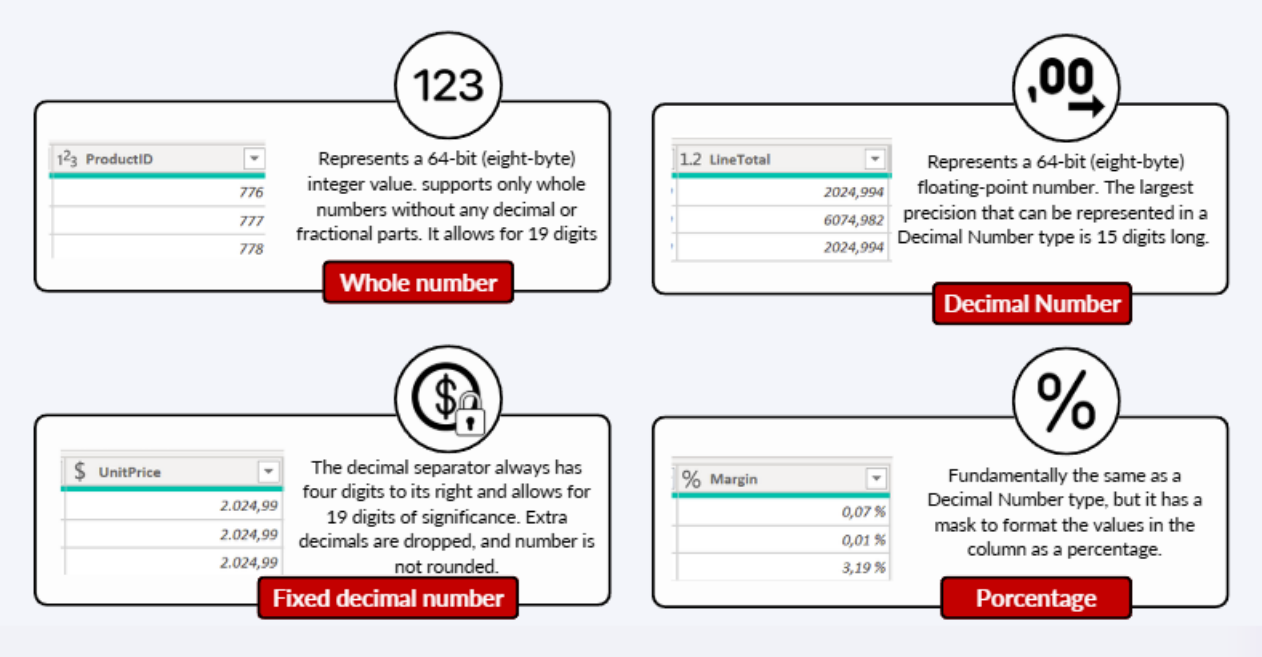
Data types


Number
123
Whole number
,00
Decimal Number
$
Fixed decimal number
%
Porcentage
Text
AB
Text
logical
True/False
Date/ Time zone
Date/Time
Date/Time/Timezone
Date
Duration
Time
Any
Any
Binary
0101
1001
0110
Binary

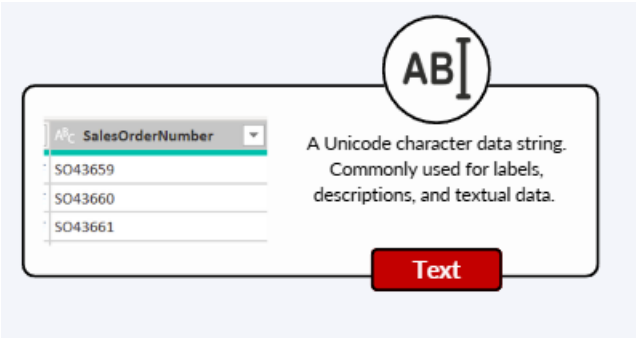
Data types: Any



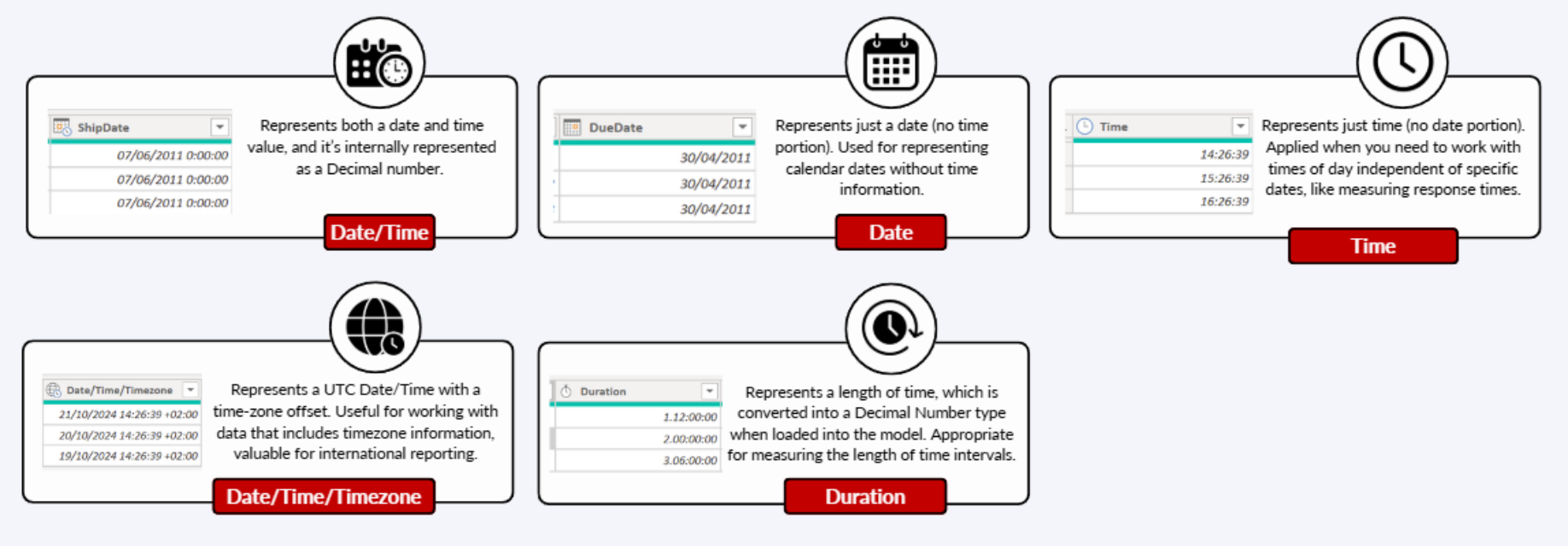
Data types: Number



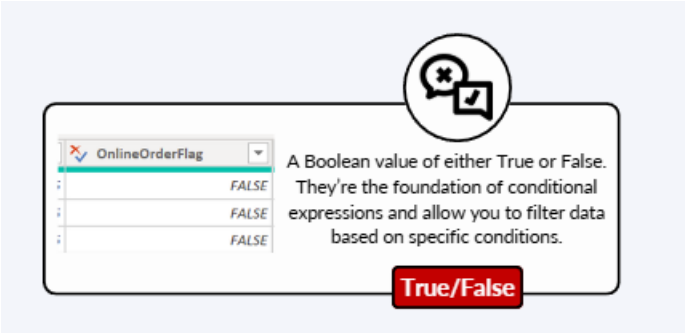
Data types: Text



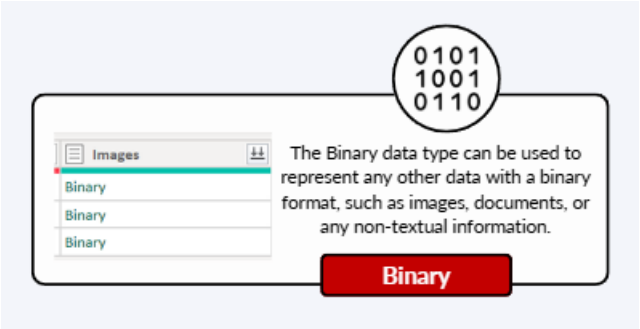
Data types: Date/Time zone



Data types: logical/boolean



Data types: Binary



Data type vs Format

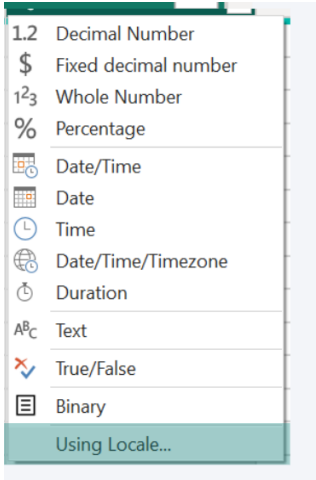
Data Type: Defines the kind of data (text, number, date) and determines how much memory is allocated to store values. It directly influences how data is stored and used in calculations.

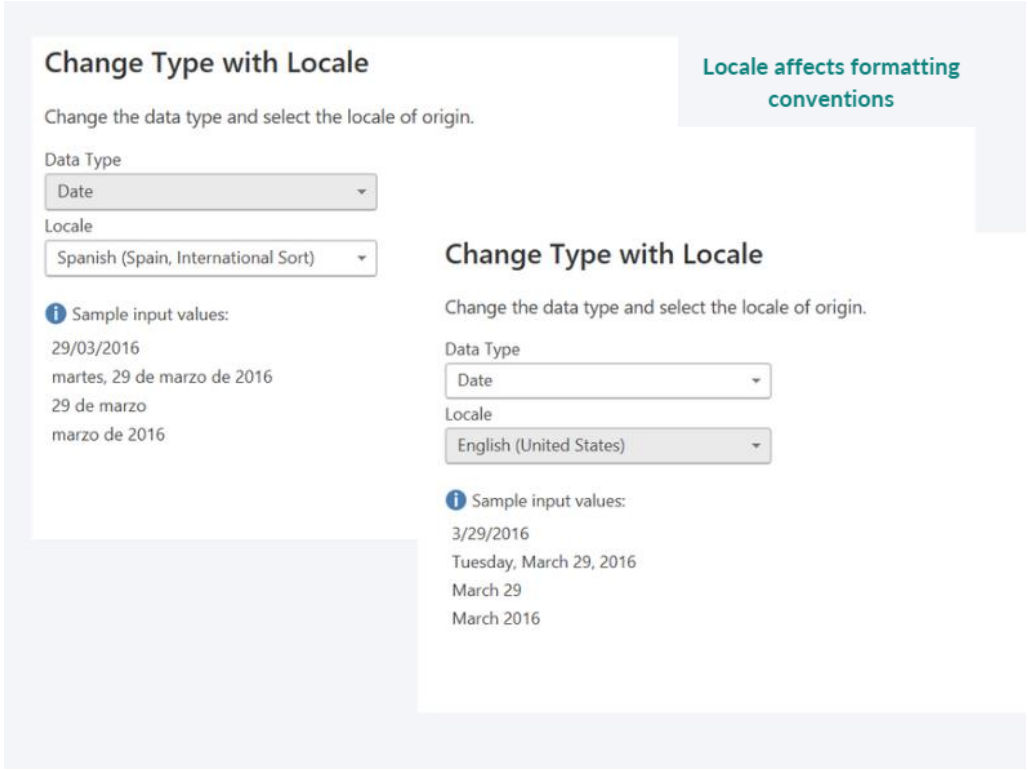
Format: Controls haw a value is displayed without affecting their underlying data structure or storage.

***Key Difference: Data types affect the backend (storage and memory usage), while formats only influence the frontend (visual presentation)**

Effect of locale/culture

Different regions around the world follow varied data conventions





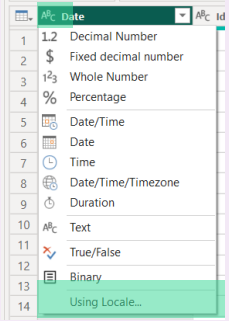
Exercise 7

Change Data Type

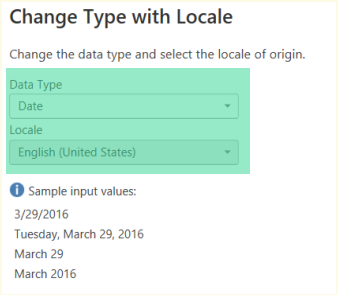
Exercise 7: Change data types

Goal: *Change data types*

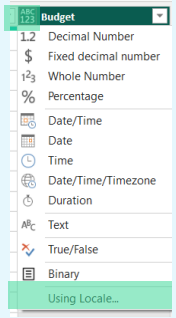
1 Go to Home > Transform Data
Select 'Budget' query.
Select 'Date' column. Click on the data type and select *Using Local*



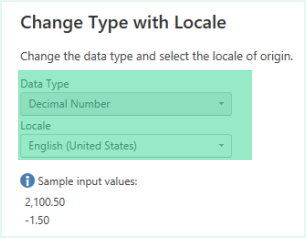
2 In 'Data type', select 'Date' and in 'Locale' select 'English (United States)'



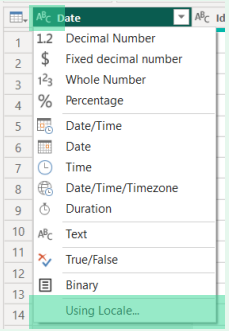
3 Select 'Budget' column. Click on the data type and select Using Locale



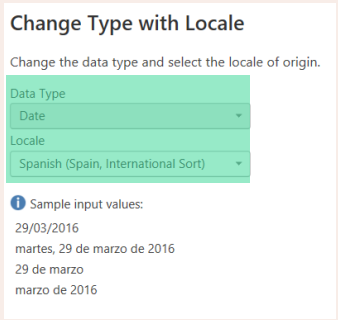
4 In 'Data type', select 'Decimal Number' and in 'Locale' select 'English (United States)'



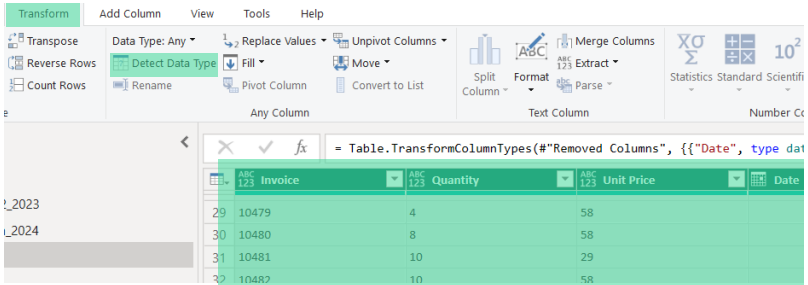
5 Select 'Sales' query.
Select 'Date' column. Click on the data type and select *Using Local*



6 In 'Data type', select 'Date' and in 'Locale' select 'Spanish(Spain, International Sort)'



7 Since there are no more decimal values or dates in the table, we will select all columns in the table by pressing Ctrl + A, then go **to Transform / Detect Data Type.**



Break time

DEMO 08

- Overview of Abstract Types
- Utilizing Parameters

Utilizing Parameters

Parameters

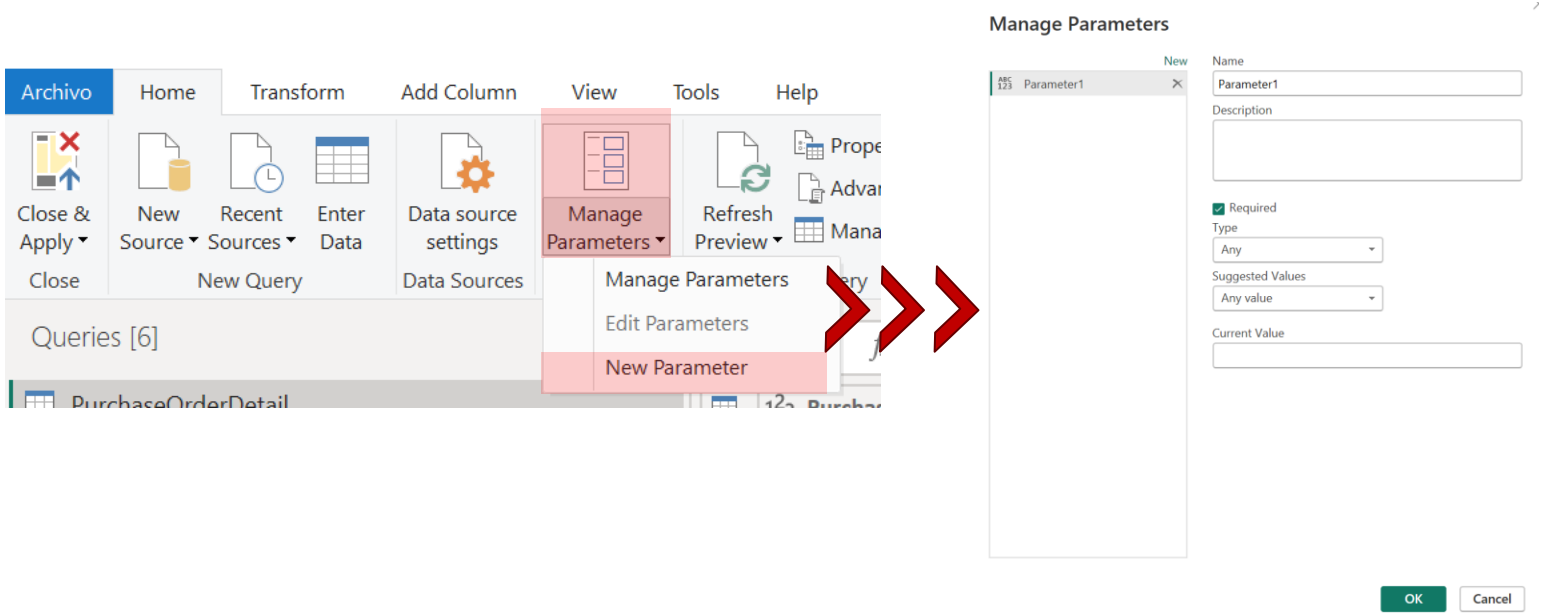
Parameters in Power Query play an important role in making queries dynamic.

- **They allow for the input of scalar values like dates, numbers, or text.**

- **This design enables you to make external modifications** to a single parameter, which then automatically propagates updates across multiple queries or steps within a query.

Create Parameters in Power Query

- **Manage Parameters:**



Parameters

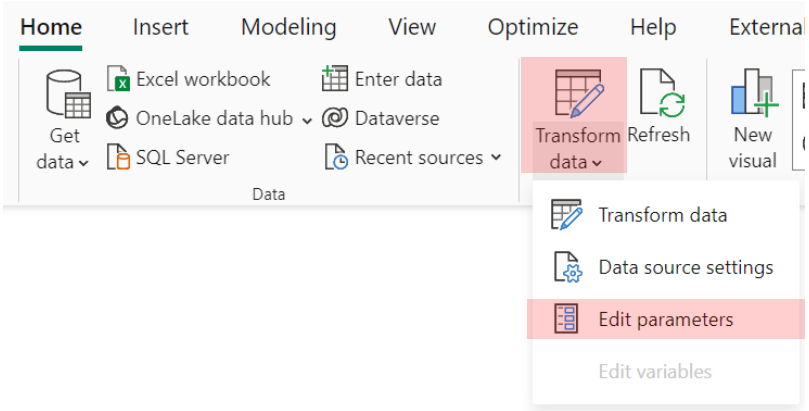
Parameters in Power Query play an important role in making queries dynamic.

- **They allow for the input of scalar values like dates, numbers, or text.**

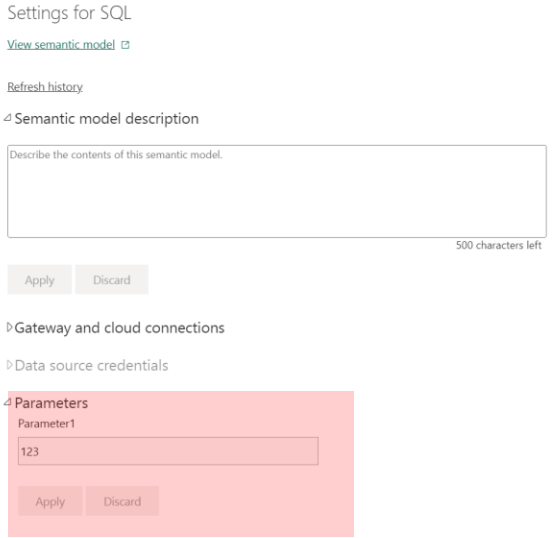
- **This design enables you to make external modifications** to a single parameter, which then automatically propagates updates across multiple queries or steps within a query.

Edit Parameters in Power BI Desktop or Power BI Service

Power BI Desktop



Power BI Service



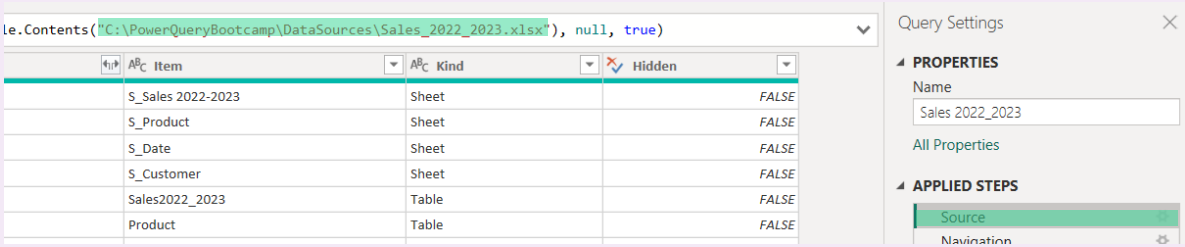
Exercise 8

Creating Parameters

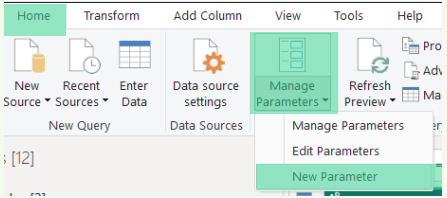
Exercise 8: Creating Parameters

Goal: *Learn to create parameters*

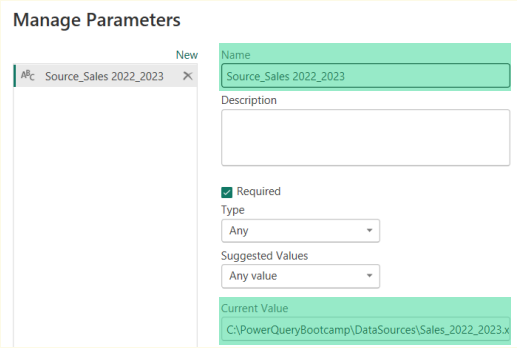
1. Go to the Source step and copy the URL from the 'Sales2022_2023' table



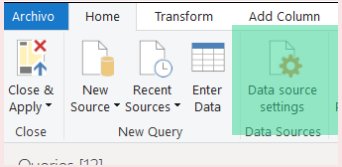
2. **Create a new parameter:** Click on Home > Manage Parameter > New Parameter.



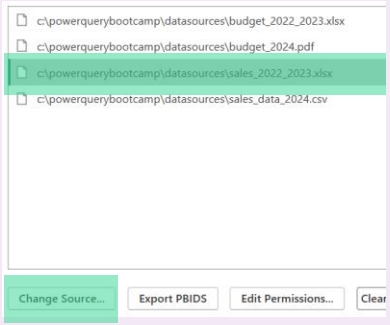
3. Name the parameter 'Source_sales 2022_2023' and paste the URL you copied in step1



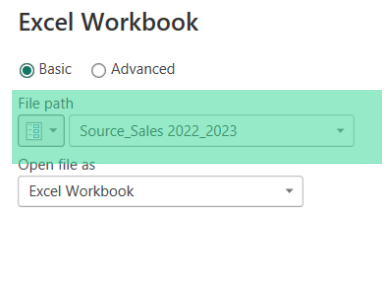
4. Click on *Home* > *Data Source Setting*



5. Select URL from 'sales_2022_2023'. Click 'Change Source'



6. Select *Parameter* option

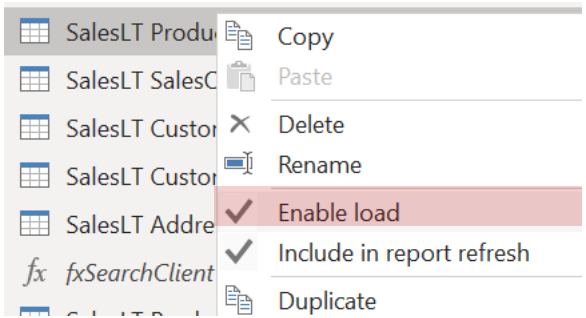


DEMO 09

- Data Loading

Minimize loaded queries

Limit the number of queries loaded into memory to optimize performance.



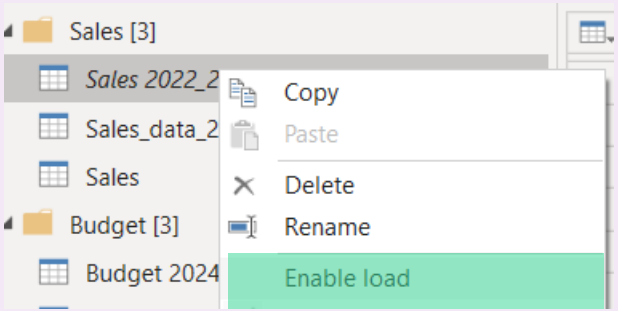
Exercise 9

Minimize Loaded Queries

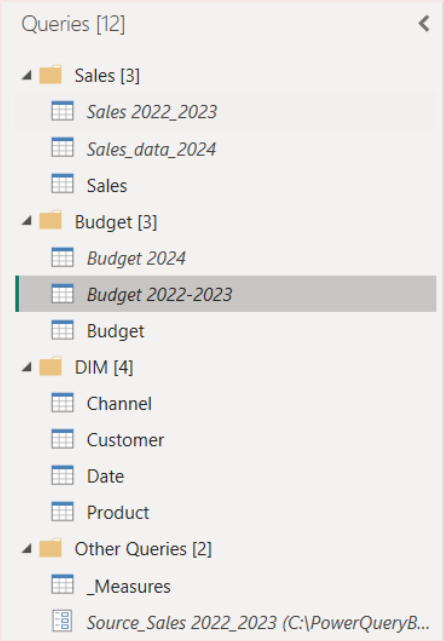
Exercise 9: Minimize loaded queries

Goal: *Learn to disable queries*

1. Select query 'Sales_data_2024'
   Right-click on the selected query, uncheck Enable Load



2. Repeat the process for queries:
   - Sales_data_2024
   - *Budget 2024*
   - *Budget 2022-2023*



**Notice that the query names are in italics, which means they will not be loaded into the final model*

DEMO 10

- Understanding Query Folding
- Types of Errors in Power Query

Understanding Query Folding

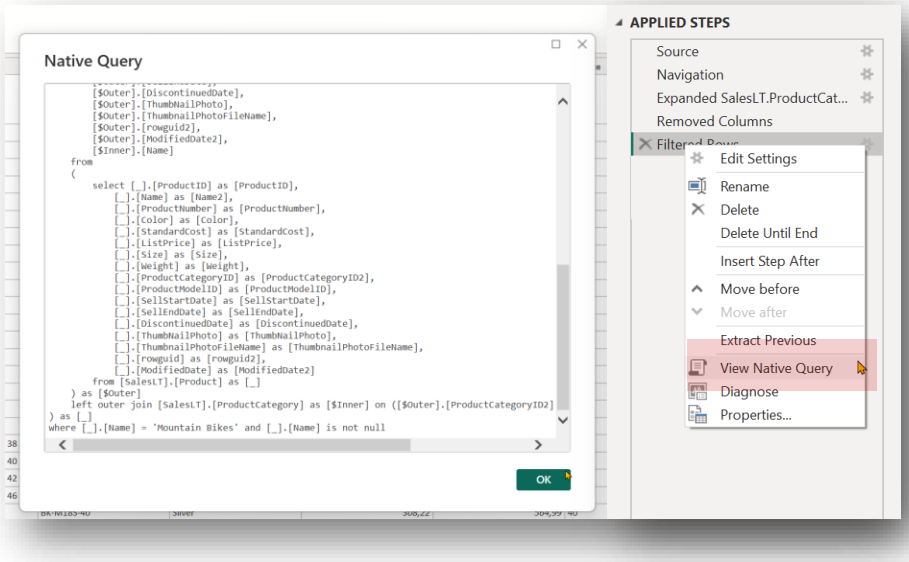
Query Folding

Understanding Query Folding

- Converts M code into a query language.
- Offloads complex processing to the data source.
- Generates a single SQL query executed directly on the data source for better performance!

Benefits

- Enhances data refresh efficiency
- Essential for DirectQuery/Dual mode, ensuring queries fold properly
- Makes incremental refresh work as intended



Query Folding

- Query folding is supported by sources like relational databases, OData, SSAS, and SharePoint lists, allowing transformations to push down to the data source.

- However, **not all file types support query folding**, such as Excel and text files.

Importantly, **not every data transformation will fold:**

- ✅ Removing or renaming columns
- ✅ Filtering rows
- ✅ Grouping and summarizing
- ✅ Merging or appending queries from the same data source
- ✅ Basic custom columns
- ✅ Pivoting and unpivoting

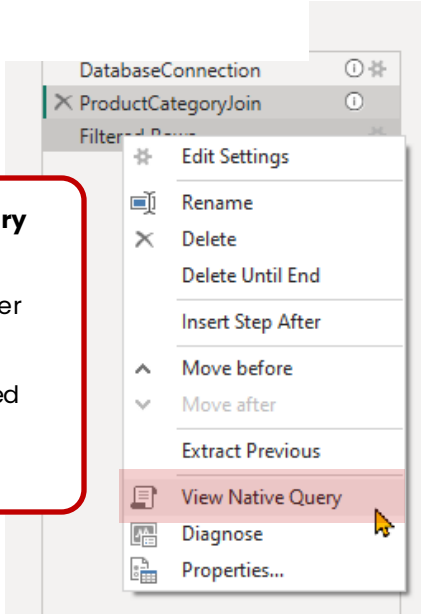
- 🚫 Merging or appending queries across **different sources**
- 🚫 Complex custom columns
- 🚫 Adding index columns
- 🚫 Changing data types (this may vary depending on the transformation)

Query Folding – Native Query

- Use **Value.NativeQuery** function
  - The goal of this process is to execute the following SQL code, and to apply more transformations with Power Query that can be folded back to the source.

```
ProductCategoryJoin = Value.NativeQuery(DatabaseConnection, "
    SELECT
        p.ProductID,
        p.Name AS ProductName,
        p.ProductNumber,
        p.Color,
        p.StandardCost,
        p.ListPrice,
        p.Size,
        p.Weight,
        c.Name AS CategoryName,
        p.SellStartDate,
        p.SellEndDate,
        p.DiscontinuedDate
    FROM
        SalesLT.Product p
    JOIN
        SalesLT.ProductCategory c
    ON
        p.ProductCategoryID = c.ProductCategoryID"
        , null, [EnableFolding = true]),
```

This example uses the **Value.NativeQuery** function to run a SQL query and enable query folding with the optional parameter **[EnableFolding = true].** This allows transformations like filtering to be pushed back to the database, optimizing performance.

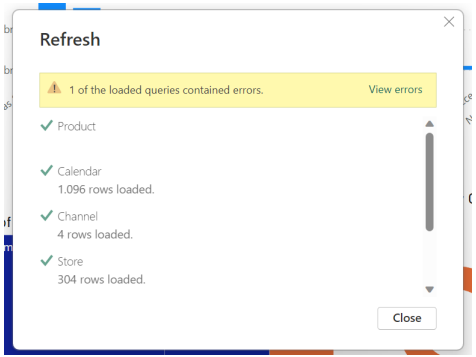


Types of Errors in Power Query

Types of Errors

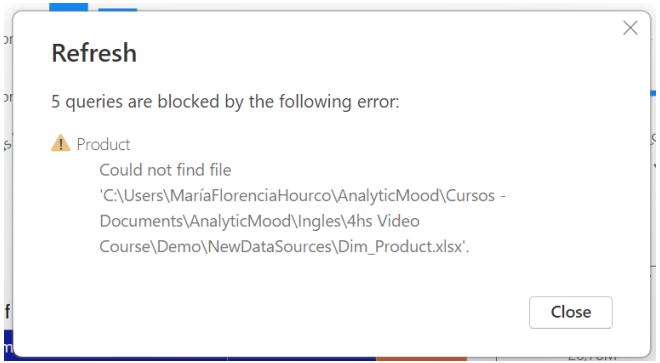
Row level Errors

The model gets updated, and the cells with errors are replaced by blank values.



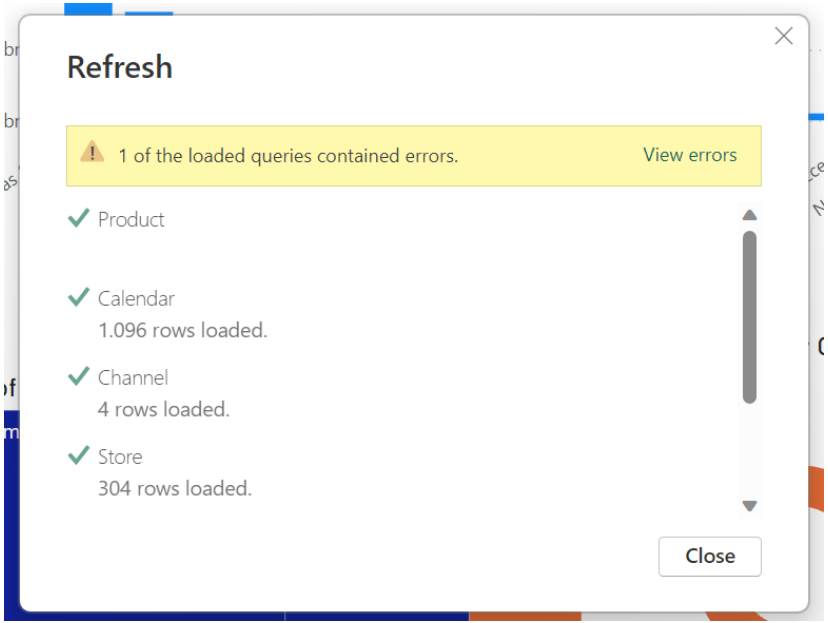
Step Level Errors

The model won't refresh until the errors are resolved.



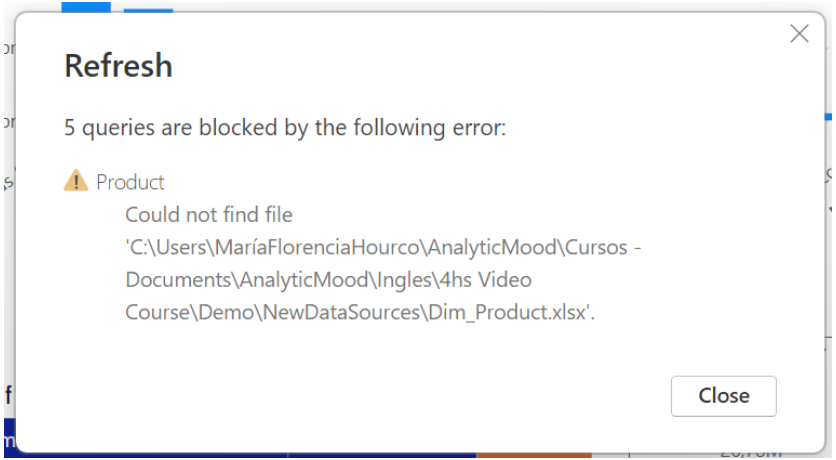
Row-Level Errors

- The model gets updated, and the cells with errors are replaced by blank values.
- Row-level errors in Power Query can be caused by:
  - A cell value conversion issue to the specified data type in the column.
  - Incorrect formatting of one of the columns involved in creating an operational column.



Step-Level Errors

- The model won't refresh until the errors are resolved.
- Step-level errors in Power Query can be caused by:
  - Can't find the source
  - The column of the table wasn't found
  - Invalid formulas
  - Duplicates or blanks in a table one to many



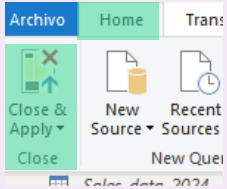
Exercise 10

Other Types of Errors

Exercise 10: Other types of errors

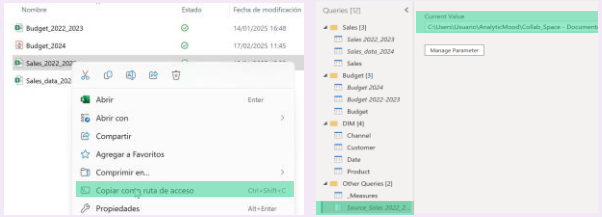
Goal: *Understand Different Types of Errors*

1 **Load the model into Power BI:** Go to Home > Close and apply

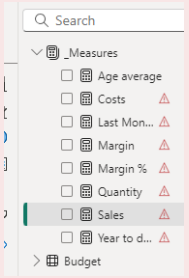


* If you are working with Exercise 10 downloaded

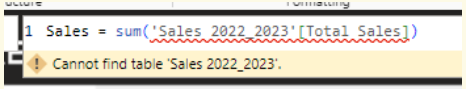
1. Go to the folder where your data sources are saved
2. Right-click on Sales_2022_2023 > Copy as path
3. Open the query Source_Sales_2022_2023 in PowerQuery
4. Paste the copied path into Current Value
5. Remove the quotation marks (")



2 Once the model is loaded into Power Bi, go to Measures and notice that all the measures that were created show an error



3 Select each of the measures that show an error



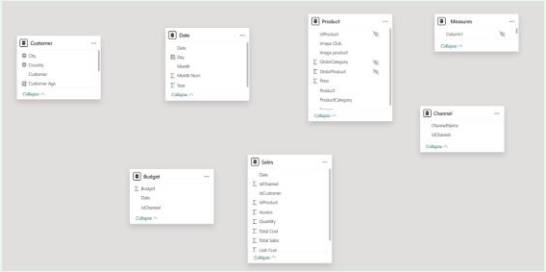
4 Change the data source to the new query we have created called 'Sales'

```
1 Sales = sum('Sales'[Total Sales])
```

Repeat the process with:

- Costs
- Quantity

5 It's necessary to create relations again, because the sales query has changed.



Data Modeling for optimal structure & DAX for calculations

Quiz

Let's do a final quiz!

Question 1

What is the main purpose of Query Folding in Power BI?

A. It converts M code into a query language and offloads complex processing to the data source.
B. It combines all the queries into a single M script for faster processing.
C. It generates a single SQL query executed directly on the data source for better performance.
D. It allows for manual editing of the SQL query to optimize performance.

Question 2

Which of the following transformations are supported by Query Folding in Power BI?

A. Removing or renaming columns
B. Filtering rows, grouping and summarizing, merging or appending queries from the same data source
C. Creating complex custom columns
D. D. Using Excel or text files as data sources

Question 3

The model gets updated, and the cells with errors are replaced by blank values. Row-level errors in Power Query can be caused by:

A. A cell value conversion issue to the specified data type in the column.
B. Incorrect formatting of one of the columns involved in creating an operational column.
C. Both A and B.
D. None of the above.

Question 4

In Power BI Desktop, an error message appears indicating that some values have been replaced by nulls. In the Power BI web service, no error is displayed when the dataset updates, even though there are row-level errors. What could be happening?

A. The Power BI web service cannot handle row-level errors.
B. The Power BI web service ignores row-level errors and proceeds with the update.
C. Power BI Desktop and Power BI Web always show the same errors.
D. Row-level errors are automatically resolved in the Power BI web service.

Recommendations for Next Steps

Part of the Power BI Bootcamp Series

Advanced Learning Path:

Aug 2, 9, 16 & 23
2pm-6pm CET

1) Power BI Bootcamp: Building the Foundations for Power BI

This Bootcamp

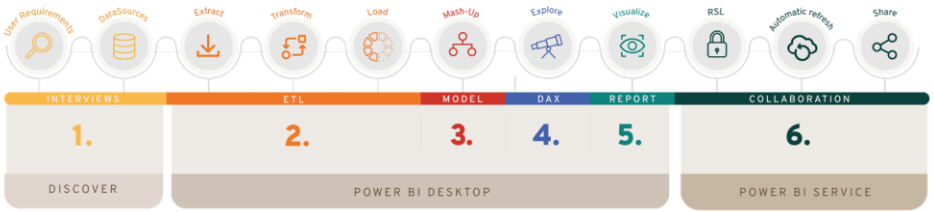
2) Power Query: Data Connections and Transformations

3) Data Modeling for optimal structure & DAX for calculations

COMING SOON

4) Visualizations for impactful reporting & Power BI Service

COMING SOON



Stay Connected

Follow us on LinkedIn:

 Nicolás Lagreste Zucchini

 María Florencia Hourcouripé

Let's Collaborate: Interested in a personalized mentorship or consulting session? Contact us:

- 📧 nico@analyticmood.com
- 📧 flor@analyticmood.com

Visit Our Website:

- www.analyticmood.com for resources, upcoming events, and blog articles.

Stay Tuned for More: **Upcoming courses and events to take your skills to the next level!**

analytic mood

Thank you!


O'REILLY®