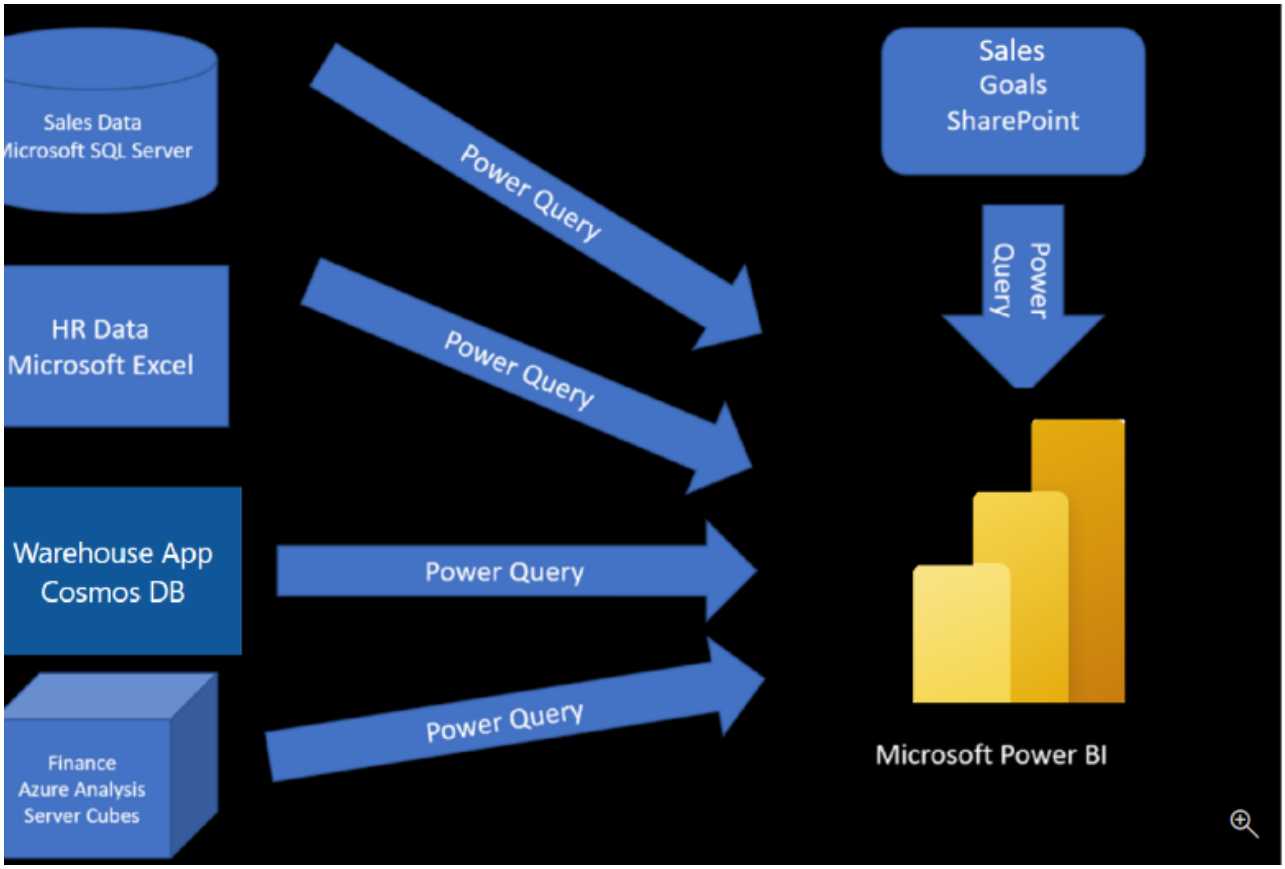
Prepare Data For Analysis

An example scenario for **learning how to get data in Power BI**

In this module's scenario you work for Tailwind Traders. You've been tasked by senior leadership to **create a suite of reports** that are **dependent on data in several different locations**. The **database that tracks sales transactions is in a Microsoft SQL Server**, a relational database that contains what items each customer bought and when. It also tracks which employee made the sale, along with the employee name and employee ID. However, that database doesn't contain the employee's hire date, their title, or who their manager is. For that information, you need to **access files that Human Resources keeps in Excel**. You've been consistently requesting that they use a SQL database, but they haven't yet had the chance to implement it.

When an item ships, the shipment is recorded in the warehousing application, which is new to the company. **The developers chose to store data in CosmosDB**, a **set of JSON documents**

Tailwind Traders has an **application that helps with financial projections**, so that they can predict what their sales will be in future months and year, based on past trends. Projections are **stored in Microsoft Azure Analysis Services**



Before you can create reports, you must **first extract data from the various data sources**. Interacting with SQL Server is different from Excel, so you should learn the nuances of both systems. After you've learned the particulars of each system, you can use **Power Query (the query engine used by Power BI and Excel) to help you clean the data**, such as renaming columns, replacing values, removing errors, and combining query results. After the data has been cleaned and organized, you are ready to build reports in Power BI. Finally, you will **publish your combined dataset and reports to Power BI service (PBIS)**. From there, other people can use your dataset

and **build their own reports or they can use the reports that you've already built**. Additionally, if someone else built a dataset that you'd like to use, you can build reports from that, too!

This module will focus on the first step, of **getting the data from the different data sources and importing it into Power BI by using Power Query**.

By the end of this module, you'll be able to:

- Identify and connect to a data source
- Get data from a **relational database, such as Microsoft SQL Server**
- Get data from a **file, such as Microsoft Excel**
- Get data from **applications**
- Get data from **Azure Analysis Services**
- Select a storage mode
- Fix performance issues
- Resolve data import errors

Get Data from Files

Organizations often **export and store data in files**

One file format is a **flat file**

flat file – a type of file that has only one data table and every row of data is in the same structure
does not contain hierarchies

common types of flat files

- comma-separated values (.csv) files
- delimited text (.txt) files
- fixed width files

Another type of file format is an **output file**, usually created from different applications

common types of output files

- Microsoft excel workbooks (.xlsx)

Power BI allows the ability to **get data from many different types of files** – a full list is available under the Get Data drop-down in the Home bar section

Flat File Locations

Multiple options are available to **upload data from a flat file**

Cloud options:

- OneDrive for Business
- SharePoint – Team Sites

allow the ability to keep the **Power BI** data (dataset, reports, and dashboards) **in-sync**

Local options:

- Local (a .csv file saved in your computer)

if data is not changed regularly, then this is a **suitable option**

Change the Source File
In order to **keep reports up to date**, you'll need to **update the file connection paths in Power BI**

PowerQuery provides a number of ways to accomplish this task:

1. **Data Source settings**
   Manual (human-made) File path changes in the PowerQuery section in the **Data source settings** page under the **Home** section
2. **Query settings**
3. **Advanced Editor**

Get Data from Relational Data Sources

Power BI will **help you to monitor the progress of your business** and **identify trends**

Allows you to **forecast sales figures, plan budgets,** and **set performance indicators and targets**.

Connecting to data in a relational database.

1. **Get Data**
2. **SQL Server**
3. **Enter database server name and database name in the SQL Server database window**
4. **Two options in data connectivity mode**
   1. **Import** (selected by default, recommended)
   2. **DirectQuery**
5. **Sign-in with your Username and Password**
   1. **Windows** account - (Azure Active Directory credentials)
   2. **Database** - Database credentials – SQL Server has its own sign-in and authentication system that is sometimes used
   3. **Microsoft** account – Use your Microsoft account credentials. Used for Azure services

Select Data to Import

Navigator window displays the data that is available in your data source

Load – Automatically load your data into a Power BI model in its current state

Transform Data – Opens data in Microsoft Power Query
Can perform actions such as **deleting unnecessary rows or columns, grouping your data, removing errors, and many other quality tasks**

Importing Data by Writing an SQL Query

Import data by **writing an SQL query** to **specify only the tables and columns you need**

open the SQL Server database window

1. Enter your server and database name
2. **Select the arrow next to Advance options**
3. In the **SQL statement box**, write your query statement, and then select **OK**
4. An example of **Select SQL statement**
   to load the ID, NAME and SALESAMOUNT columns from the SALES table

```
SELECT

ID
, NAME
```

, SALESAMOUNT

FROM SALES

Change Data Source Settings

After creating a data source connection and load data into Power BI Desktop.

You can return and change your connection settings at any time.

Action is usually **required due to a security policy within the organization**,

- when password needs to be updated every 90 days

You can also change the **data source, edit permissions, or clear permissions**

On the **Home** tab, select **Transform data**, and then select the **Data source settings** option

Write an SQL Statement

SQL (Structured Query Language) – standardized programming language – used to **manage relational database and perform various data management operations**

SQL is beneficial allows you to:

- **load only the required set of data by specifying exact columns and rows in your SQL statement**
- **import them into your data model**
- **join different tables, run specific calculations, create logical statements, and filter data in your SQL query**

SQL examples
ID, NAME, and SALESAMOUNT are selected from the SALES table

```
SELECT
ID
, NAME
, SALESAMOUNT
FROM
SALES
```

***** (wildcard character) in the SELECT statement – e.g., SELECT * FROM SALES will **import all columns that you don't need from the specified table**
will lead to **redundant data in your data model** – leads to performance issues, additional steps to normalize data for reporting

WHERE clause. This clause will **filter the rows to pick only filtered records that you want**.

e.g., getting recent sales data after Jan 1, 2020

```
SELECT ID, NAME, SALESAMOUNT FROM SALES
WHERE OrderDate >= '1/1/2020'
```

If PowerBI uses a view, when it retrieves data, it **participates in query folding**, a **feature of PowerQuery**

PowerQuery will **optimize data retrieval according to how the data is being used later**

Get Data from a NoSQL database

A NoSQL (non-SQL, not only SQL, non-relational) database
a flexible type of database that does not use tables to store data

An example scenario:
Using **CosmosDB, a NoSQL database**, as the **data repository** for the shipping and tracking products from their warehouses

CosmosDB used to **store JSON documents**
which are **open standard file formats** that are **primarily used to transmit data between a server and web application**

Connect to a NoSQL Database (Azure Cosmos DB)

Use the **Get Data** feature in the Power BI desktop
=> Select **More..** option to **locate and connect to the type of database**
=> Select the **Azure** category
=> Select the **Azure Cosmos DB**
=> Select **Connect**

On the **Preview Connector** window
=> Select **Continue**, and then enter your database credentials
=> On the **Azure Cosmos DB** window page
=> Enter the **database details**,
you can specify the **Azure Cosmos DB** account **endpoint URL that you want to get the data from**
The URL can be found in the **Keys** blade of the **Azure** portal

IF YOU ARE CONNECTING TO THE ENDPOINT FOR THE FIRST TIME
Make sure you enter your account key,
Can find this in the **Primary Key** box in the **Read-only Keys** blade of the **Azure portal**

Import a JSON File

JSON type records **must be extracted and normalized before you can report on them**
you NEED to TRANSFORM THE DATA before loading it into Power BI

After you have **connected to the database account**
=> **Navigator** window opens, showing a list of databases under that account
=> **Select the table that you want to import**, e.g., a given table
=> Select the **Edit** button to **open the records** in **Power Query**

In Power Query, select the **Expander** button on the **right side of the Column1 header**, will display the context menu with a list of fields
=> **Select the fields (columns)** that **you want to load** into Power BI
=> Clear the **Use original column name as prefix** checkbox
=> Select **OK**
=> Review that the selected data and that you are satisfied with it, select **Close & Apply** to **load the**

data into Power BI

Data now **resembles a table with rows and columns**. Data from Cosmos DB can now be related to data from other data sources and used in a Power BI report.

Get Data from Online Services

Power BI allows for the ability to **combine the data from multiple applications to produce more meaningful insights and reports**

Example range of software applications:

- SharePoint
- OneDrive
- Dynamics 365
- Google Analytics

An example scenario:
SharePoint used to **collaborate and store sales data**. Required to establish a connection with it so that **sales goals can be used alongside other sales data** to **determine the health of the sales pipeline**

Start a similar way, selecting the **Get Data**
=> Select the option that you need from the **Online Services** category
=> Select **SharePoint Online List**
=> Select the **Connect**, then asked for the **SharePoint URL**
=> The **URL** is the one **used to sign in to the SharePoint through a web browser**
Only need to load the **Site URL,** the **full URL file path is NOT NECESSARY**
=> After entering the **SharePoint URL**, select **OK** then **authorize the connection** with a **Microsoft Account**
=> The **Navigator** window then appears
=> Window **displays the tables and entities within your SharePoint site**
=> Select the **specific lists** that you want to load
=> Take either the option to **automatically load the data into Power BI** or **launch the Power Query Editor** in order to **transform the data before loading it**

Select a Storage Mode

Three different types of storage modes:

- **Import**
- **DirectQuery**
- **Dual (Composite)**

Access storage modes => **switching to the Model view**
=> selecting a **data table**
=> in the resulting **Properties** pane, selecting which mode you want to use from the **Storage mode** drop down list

Import (ing Data into Power BI)

This means that **the data is stored in the Power BI file** and **gets published with the Power BI reports**

GIANT NOTE: The data is stored in the Power BI file and gets published along with the Power BI

reports

Biggest downfall:
For security reasons, this might not be allowed (e.g., import local copies of the data into your reports)
Solution? => **create a direct connection** to the Sales departments' data source

Automatically selected default option for creating new Power BI reports

Data refreshes can be scheduled or on-demand.

All Power BI features are available:
- Q & A
- Quick Insights

DirectQuery
A storage mode, **allows you to query the data in the data source directly and not import a copy into Power BI**

Useful for when you do not want to save local copies of your data because **data will not be cached**

You can **query the specific tables** as you are **creating a direct connection to the data source**

Useful because:
- **Ensures you are always viewing the most recent version of the data**
- **Ensures security requirements are met** – impossible to directly import a copy
- **Avoid creating performance bottlenecks** – avoids too large and take too long to load
  - Ideal for when you have large datasets to pull data from

Dual (Composite)
You can **identify some data to be directly imported**, and **other data must be queried**

Any table that is **brought in to your report is a product of both Import and DirectQuery modes**

allows Power BI to **choose the most efficient form of data retrieval**

Get Data from Azure Analysis Services

An Azure product

Allows you to **ingest data from multiple data sources, build relationships between the data, and create calculations on the data**

Calculations are built using **data analysis expressions (DAX)**

An example scenario:
A group uses **Azure Analysis Service** to store financial projection data. You've been asked to **compare this data with actual sales data in a different database**

Getting data is **similar to an SQL server**

You can:

- Authenticate to the server
- Pick the cube you want to use
- Select which tables you need

Notable differences between **Azure Analysis Services cubes** and **SQL Server** are:

- **Analysis Service cubes** have **calculations already in the cube**
- If you don't need an entire table, **you can query the data directly**.
  - **Azure Analysis Service** uses **multi-dimensional expressions (MDX)** and **data analysis expressions (DAX)** to query the data

Connect to Data in Azure Analysis Service

Start a similar way, select the **Get Data** feature
=> Select **Analysis Services** in the menu
=> Enter the **server address and database name** in the following prompts
=> Either select **Import or Connect live**

Live Connection (Connect Live)

A new option in **Azure Analysis Services**

Helps you **keep the data** and **DAX calculations in their original location**

Also lets you have a **fast refresh schedule –** when data is refreshed in the service, **Power BI reports will be immediately updated**

In most cases, you will most likely import the data directly into Power BI

An acceptable alternative and **RECOMMENDED**
=> **Import the data** (Excel, SQL Server, and so on) **into the Azure Analysis Services model**
=> Use a live connection

Using this approach allows **the data modelling and DAX measures to be all performed in one place** and leads to a **simpler and easier way to maintain solutions**

Fix Performance Issues

Addressing performance issues when running reports can be done through the **Performance Analyser tool** in order to **help fix problems and streamline the process**

Optimize Performance in Power Query

Performance in Power Query depends on the performance at the data source level

e.g., extracting data from a Microsoft SQL server should follow the performance tuning guidelines for the product: index creation, hardware upgrades, execution plan testing, data compression

Power Query takes advantage of **good performance at the data source** through a technique called **Query Folding**

Query Folding

the process by which the **transformations and edits that you make in Power Query Editor are simultaneously tracked as native queries, or simple Select SQL statements**, while you are making active transformations

Ensures that **these transformations can take place in the original data source server** and **do not overwhelm Power BI computing resources**

Increases the performance of Power BI reports

Power Query Editor allows you to:

- renaming / deleting columns
- appending
- parsing
- filter
- grouping your data

Benefits of Query Folding include:

- **More efficiency in data refreshes and incremental refreshes**
  - Power BI is able to better allocate resources and refresh the data faster because Power BI does not have to run through each transformation locally
- **Automatic compatibility with DirectQuery and Dual storage modes**
  - **All DirectQuery and Dual Storage mode data source** must have the **back-end server processing abilities to create a direct connection**

Basically **different changes in the Power Query Editor to a given table CAN BE SUMMED UP to a SQL Select statement**

The SQL query can be accessed by looking at the **Applied Steps window**, right-clicking a change and clicking on **View Native Query**

View Native Query is not possible for the following transformations:
adding an index column
merging and appending columns of different tables with two different sources
changing the data type of a column

A good guideline to remember:
If you can translate a transformation into a Select SQL statement

which includes operators and clauses such as GROUP BY, SORT BY, WHERE, UNION ALL, and JOIN, **you can use query folding**

Query Diagnostics

This feature allows you to **determine what bottlenecks (if any) exist while loading and transforming your data, refreshing your data in Power Query, running SQL statements in Query Editor, and so on**

In order to access this
Go to **Tools** in the **Home** ribbon, when you are ready to begin transforming your data or making other edits in Power Query Editor
=> Select **Start Diagnostics** in the **Session Diagnostics** section
=> When finished, select **Stop Diagnostics**

Selecting **Diagnose Steps** shows you **the length of time that it takes to run that step**

Tells you **if a step takes longer to complete than others**, which then **serves as a starting point for further investigation**

Useful for when **you want to analyse performance on the Power Query side for tasks**

Other Techniques to Optimize Performance

- **Process as much data as possible in the original data source**
  - As too much processing power can also be used during Power Query and Power Query Editor
- **Separate date and time, if bound together**
  - **Separating them into distinct columns BEFORE IMPORTING THEM INTO POWER BI** will increase compression abilities

Resolving Data Import Errors

You may encounter errors from factors such as

- Power BI imports from numerous data sources
- Each data source might have dozens (and sometimes hundreds) of different error messages.
- Other components can cause errors, such as hard drives, networks, software services, and operating systems.
- Data can often not comply with any specific schema.

Query Timeout Expired

As relational source systems usually have **many people who are concurrently using the same data in the same database**

Some relational systems seek to limit a user from **monopolizing all hardware resources** by **setting a query timeout**

Power BI Query Error: Timeout Expired

this error **indicates that you've pulled too much data** according to the organization's policies

This error can be resolved by **pulling fewer columns or rows from a single table**.

While you are writing SQL statements, include **groupings and aggregation**

The solution:

Combine half the columns in **one query**
Combine the other half in a **different query**
Use **Power Query** to **merge those two queries back together after you're finished**

We Couldn't Find Any Data Formatted as a Table

Power BI expected to find a **data formatted as a table from Excel**. But **did not find it**

Solved by:
Error message tells you how to solve it

Could Not Find File

Usually caused by **file moving locations** or the **permission to the file changing**

Data Type Errors

This error occurs when **interpreting the data type in Power BI**. Resolution to the error **unique to the data source**

An example scenario:
Importing data from an SQL server and **see blank columns**, you could try to **convert to the correct data type in the query**

e.g., instead of using the query
SELECT CustomerPostalCode FROM Sales.Customers;

to using

SELECT CAST (CustomerPostalCode as varchar(10)) FROM Sales.Customers

By **specifying the correct type at the data source**, you **eliminate many of these common data source errors**

Clean, Transform, and Load Data in Power BI

The example scenario

You have imported data from several different sources into Power BI. Data is **not prepared for analysis**

Several issues, include:

- a column called **Employment status** only contains **numerals**
- **Several columns contains errors**
- **Some columns contains null values**
- The **customer ID** in some columns **appear as if it was duplicated repeatedly**
- A single **address column** has **combined street address, city, state, and zip code**

This leads to created visuals on reports being filled with incorrect results, bad data and simple reports about sales totals are wrong

Dirty data can be **overwhelming**

Clean data has the following **advantages**:

- Measures and columns produce more accurate results when they perform aggregations and calculations.
- Tables are organized, where users can find the data in an intuitive manner.
- Duplicates are removed, making data navigation simpler. It will also produce columns that can be used in slicers and filters.
- A complicated column can be split into two, simpler columns. Multiple columns can be combined into one column for readability.
- Codes and integers can be replaced with human readable values.

In this module, you will learn to:

- Resolve inconsistencies, unexpected or null values, and data quality issues.
- Apply user-friendly value replacements.
- Profile data so you can learn more about a specific column before using it.
- Evaluate and transform column data types.
- Apply data shape transformations to table structures.
- Combine queries.
- Apply user-friendly naming conventions to columns and queries.
- Edit M code in the Advanced Editor.

Shape the Initial Data

Power BI allows you to **shape (transform) your imported data**

You can accomplish actions such as:

- **renaming columns**
- **renaming tables**
- **changing text to numbers**
- **removing rows**
- **setting the first row as headers**

When you work in Power Query Editor, all **steps that you can take to shape your data are recorded**

each time **the query connects to the data source**, **it automatically applies your step**, **so your data is always shaped**

Power Query Editor only makes changes to **a particular view of the data**

So you can **feel confident about changes that are made to your original data source**

Identifying Column Headers and Name

The first step, **identify the column headers and names** within the data
=> bad data migration can lead to columns and headers to not be migrated properly
=> you can **make changes to reorganize data**

Promote Headers

When a table is created in Power BI Desktop, **Power Query Editor assumes that all data belongs in table rows**.

However, **a data source might have a first row that contains column names**

To correct this **inaccuracy**, you need to **promote the first table row into column headers**

This can be done through **two ways**:

- selecting **Use First Row as Headers** option on the **Home tab**
- selecting the **drop-down button next to Column1** and then selecting **Use First Row as Headers**

Rename Columns

Done when **one or more columns:**

- have the wrong headers
- a header has a spelling error
- the header naming convention is not consistent or user-friendly.

Renaming can be done in **two ways**:

- **Right-click the header**, select **Rename**, edit the name, and then **press Enter**
- You can **double-click the column header** and **overwrite the name with the correct name**

Remove Top Rows

When shaping data, there might be a need to **remove some of the top rows**,
e.g., they contain a blank, or unneeded data

In order to do this:

- Select **Remove Rows** and then select **Remove Top Rows** on the **Home tab**

Remove Columns

This is a **key step when transforming your data**

KEY

- **remove unnecessary columns as early as possible**
- helps focus on the data you need
- helps improve the performance of Power BI Desktop datasets and reports

IF YOU DON'T PLAN ON USING THE DATA (THE COLUMN) IN A REPORT
=> THEN THE COLUMN ADDS NO VALUE TO THE DATA MODEL

There are **multiple ways to do this**:
At the source

- **Limit the column when you get data from the data source**
  - e.g., when extracting a data from a relational database using SQL, you can limit it by **using a column list in the SELECT statement**

In Power BI Desktop

- **Select the columns that you want to remove**, and then, on the **Home tab**, select **Remove Columns**
- Alternatively, **select the columns that you want to keep**, and then, on the **Home tab**, select **Remove Columns => Remove Other Columns**

Unpivot Columns

a useful feature in Power BI

Mostly used when importing data from Excel

Separating a specific set of data into a more streamlined version

The way to do this is:

- **Select the column headers** that you wish to unpivot, select the **Transform tab** in **Power Query**, and then **select Unpivot**

Unpivoting streamlines the process of creating DAX measures on the data

It also **creates a similar way of slicing the data** with the **Year** and the **Month** columns

Pivot Columns

If the **data you are shaping is flat**

Flat – has a lot of detail but is not organized or grouped in any way

the lack of structure can complicate your ability to identify patterns in the data

You can use the **Pivot Column** feature to **convert your flat data into a table that contains an aggregate value for each unique value**

This might be good for **using different math functions such as Count, Minimum, Maximum, Median, Average or Sum**

Basically, when you Pivot, you take two original columns and create a new attribute-value pair that represents an intersection point of the new columns

On the **Transform tab**, select **Transform => Pivot Columns**
=> On the **Pivot Column window** that displays
=> Select a column from the **Values Column** list, such as **Sales**
=> Expand the **advanced options** and select an option from the **Aggregate Value Function** list, such as **Count (All)** and then select **OK**

Close & Apply

Power Query Editor records all the steps taken to **shape the data**

All the steps taken to **transform the data** is recorded in the **Query Settings pane**

After making all the changes, **select Close & Apply** to **apply the changes to the data model**

Simplify the Data Structure

Simplifying your data structure, e.g., **change table and column names** – to ensure a **consistent format** can be done through the **Power Query Editor**

Rename a Query

Good practice

change uncommon/unhelpful query names to **names that are more obvious or that the user is more familiar with**

e.g., changing **from FactProductTable** to **Product** as a query name

e.g., changing **from TargetSales** query to **TargetSales_2022** query as it is a query you will have every year

In order to rename, in the **Queries pane** to the left of the data
=> **Select the query you want to rename**
=> **Right-click the query**, and select **Rename**
=> Edit the current name, and **press Enter**

Replace Values

The **replace values** feature can be used to **replace any value with another value in a selected column**

e.g., correcting a misspelled value in a column (Dezember instead of December)

In order to do this, **select the column that contains the value that you want to replace**
=> Select **Replace Values** on the **Transform tab**
=> In the **Value to Find box**, enter the name of the value that you want to replace
=> In the **Replace With box**, enter the correct value name and then **select OK**

NOTE:
Power Query DOES NOT allow selecting ONE CELL and CHANGING THE ONE VALUE
e.g., like in Excel

Replace Null Values

Occasionally, null values are present within the data.

This could be a **problem**
e.g., **when calculating the averages will not be calculated correctly**.

This is because **AVERAGE** takes **the total and divides by the number of non-null values**.
If **NULL** is **synonymous with zero in the data**, the **average will be different from the accurate average**

One solution would be:

- Change the **nulls to 0s** – this will produce **more accurate results when calculating with the column's values**
  - This step can be done **the same as replace values**

Remove Duplicates

It is also possible to **remove duplicates from columns** to **only keep unique names in a select column** by using the **Remove Duplicates** feature in **Power Query**

This is usually done when you **want to create a table with unique categories** and **use it in your data model**

This can be done through, **selecting a column**
=> **Right-clicking on the header of a column**
=> **Selecting** the **Remove Duplicate** option

Consider **copying the table BEFORE removing the duplicates**

=> The **Copy option** is at the **top of the context menu**

Best Practice for Naming Tables, Columns, and Values

General Recommendation

- Use the **language and abbreviations** that are **commonly used within your organization** and that **everyone agrees on and considers them as common terminology**.
- Give your **tables, columns, and measures – descriptive business terms**
- Replace underscores (_) with **spaces**
- Be **consistent with abbreviations, prefaces, and words** like “number” and “ID”
- **Excessively short abbreviations** can **cause confusion**
- **Remove prefixes and suffixes** that you might use in table names and name them in a **simple format**
- **Avoid acronyms in values**
- Consider the value length, **too long is difficult to read and fit into a visual**, and **too short might be difficult to interpret**

Evaluate and Change Column Data Types

When importing a table from data source, **Power BI scans the first 1,000 rows** and tries to **detect the type of data in the columns**

Incorrect data types leads to **performance issues**

More common for flat files (.csv and .xlsx) files as their data is manually entered.

Ideally **determining/evaluating the data type** is better done in **Power Query Editor**

In order to change a column data type, first **select the column you wish to change the data type**
=> **Right click the column header**
=> Select the **Change Type**
=> Select the **Data Type you wish the column to now be made of**.

Implications of Incorrect Data Types

Incorrect Data Types will prevent you from **creating certain calculations, deriving hierarchies, or creating proper relationships** with other tables

e.g., an incorrect data type (**Text instead of Date**) will not allow you to use **certain calculations (YTD – a time based calculations)**

YTD Formula:
Quantity of Orders YTD = TOTALYTD(SUM(‘Sales’[OrderQty]), ‘Sales’[OrderDate])

another e.g., an **incorrect data type applied on a date field** => the **inability to create a data hierarchy**

Data Hierarchy – Allows you to **analyse your data** on a **yearly, monthly, or weekly basis**

Best Practice

- Use a **date table** and **turn off the auto date/time** to **get rid of the auto generated hierarchy**

Change the Column Data Type

You can **change the data type of a column** in **two places**:

- One way, **select the column that has the issue**,
  - => select **Date Type** in the **Transform tab**, and then select the **correct data type from the list**

Change the Column Data Type in Power Query Editor

Two ways to **change the data type**:

- select **Data Type** in the **Transform tab**, then select the **correct data type**
- select the **Data Type icon** next to the column header, then **select the correct data type from the list**

Combine Multiple Tables into a Single Table

Ability to combine queries is powerful because it **allows you to append or merge different tables or queries together**

Reasons to combine multiple tables into a single table:

- Too many tables exist, making it difficult to navigate an overly-complicated data model
- Several tables have a similar role
- A table has only a column or two that can fit into a different table
- You want to use several columns from different tables in a custom column

You can **combine the tables in TWO DIFFERENT WAYS:**

- **Merging**
- **Appending**

e.g., importing data from 3 different tables, the problem then becomes **how to merge all these data from multiple tables** and **CREATE ONE SOURCE-OF-TRUTH TABLE to create a report from**

Power BI allows you to create and merge queries into a single table

Append Queries

Appending Queries is more or less **adding rows of data to another table or query**

e.g., two tables, one with **300 rows**, and another with **100 rows**, and **when you append queries, you will end up with 400 rows**.

When you **merge queries**, you will be **adding rows from one table (or query) into another**

In order to **merge two tables**, you **MUST HAVE a column that is the KEY between the two tables**

NOTE: The **KEY must be unique for every entry (row) in the appended final table**

NOTE: The **NUMBER OF COLUMNS** for the tables that **will be appended** should **BE THE**

SAME and HOLD THE SAME COLUMN HEADERS

Remove extraneous columns that you don't need
Only keep relevant columns

The combined query will **have more rows while keeping the same number of columns**

In order to append queries you can do this through the **Home tab** on the **Power Query Editor**
=> Select the **drop-down list** for **Append Queries**
=> You can select **Append Queries as New**, which means that **the output of the appending will result in a new query or table**
=> You can select **Append Queries** which will **add rows from an existing table into another**

Creating a **MASTER TABLE** (combination of multiple tables into one)
Can be by selecting **Append Queries as New**. This selection will bring you to a window
=> In this window, **add the tables that you want to append** from **Available Tables** to **Tables to Append**
=> After **you finish adding all the tables that you want to append**, select **OK**
=> You will **be routed to a new query that CONTAINS ALL ROWS FROM ALL YOUR SELECTED TABLES**

Merge Queries

Merging queries is similar to **combining the data from multiple tables into one based on a column that is common between the tables**

This process is **similar to the JOIN clause in SQL**

An example scenario:
Sales Team wants to consolidate orders and their corresponding details (which are currently in two tables) **into a single table**

NOTE: **A COMMON COLUMN SHARED BETWEEN BOTH TABLES IS REQUIRED TO MERGE**

This can be done through going to **Home** on the **Power Query Editor Ribbon** and selecting **Merge Queries drop down menu**, you can then select **Merge Queries as New**
=> This selection will **Open a New Window**
=> Where you can **choose the tables you want to merge from the drop-down list**
=> Then **SELECTING THE COLUMN THAT IS MATCHING BETWEEN THE TABLES**, e.g., OrderID

You can also **choose how to join the two tables together**, a process that is also **similar to JOIN statements in SQL**:

- **Left Outer** – Displays **all rows from the first table** and **only matching rows from the second**
- **Full Outer** – Displays **all rows from both tables**
- **Inner** – Displays the **matched rows between the two tables**

Profile Data in Power BI

Profiling data is about **studying the nuances of the data**:

- determining anomalies

- examining and developing the underlying data structures
- querying data statistics such as
  - row counts
  - value distributions
  - minimum and maximum values
  - averages

Underlying Data Structure

In order to **view the current data model**, go to the **Model tab** under **Power BI Desktop**

On the **Model tab**, you can **edit specific column and table properties**
and you can **transform the data** using the **Transform Data button**

You can also **manage, create, edit, and delete relationships between different tables** using **Manage Relationships**, which is located on the ribbon

Find Data Anomalies and Data Statistics

Data anomalies are outliers within your data

Determining **what those anomalies are** can help you **identify what the normal distribution of your data looks like**

Power Query Editor determines data anomalies by using the **Column Distribution feature**

In order to access this, select **View** on the **ribbon**, and under **Data Preview**, you can choose a few options

In order **to understand data anomalies and statistics**, select the **Column Distribution, Column Quality, and Column Profile** options

Column Quality

shows you the **percentages of data that is valid, in error, and empty**

In an **ideal situation**, **100% of the data should be VALID**

Column Distribution

shows you the **distribution of data within the column and the counts of distinct and unique values**, both of which can **tell you details about the data counts**

Distinct Values
all **the different values in a column, including duplicates and null values**

the total count of how many values are present

Unique Values
how many values are present only once

Column Profile

Gives you a more **in-depth look into the statistics within the columns for the first 1,000 rows of data**

This column provides several different values:

- **Count of rows** – important when verifying whether the importing of your data was successful
  - e.g., if the original database had 100 rows, **row count to verify that 100 rows were, in fact, imported correctly**
  - Will also show you **how many rows Power BI has deemed as being**
    - **outliers**
    - **empty rows, and strings**
    - **the min and max**
      - tells you the **smallest and largest values in a column**
      - This distinction is especially important as it will **immediately notify you if you have a maximum value** that is **beyond what your business identifies as a 'maximum'**
      - In the case where the data is in a **text column**
        - the **minimum value is the first value** when in alphabetical order
        - the **maximum value is the last value** when in alphabetical order

Value Distribution Graph

tells you **the counts for each distinct value in that specific column**

e.g., a large number of appearances by a single value **could be an outlier**. This tool **helps you identify whether a possible data point could be an outlier or not**

e.g., on how to determine whether they're an outlier or not, the value appears far more often than other values in a column

Allows you to pinpoint a place to begin your investigation as to why

Column Statistics (the left box chart – when column profile is turned on)

will also **include**

- **how many zeroes and null values exist**
- **the average value in the column**
- the **standard deviation of the values in the column**
- **how many even and odd values are in the column**

These statistics give you an idea of the distribution of data within the column

IMPORTANT as they **summarize the data in the column and serve as a starting point to determine what the outliers are**.

Use Advanced Editor to Modify Power Query M

Each time you create a step in Power Query, you **create a step in the Power Query process**

these **steps** can be **reordered, deleted, and modified**

Power Query uses the **M Language behind the scenes**

COMBINED STEPS are **read using the Power Query Advanced Editor**

The **M language** is **always available to be read** and **modified directly**

NOT REQUIRED to **use M code** to **take advantage of Power Query**

BASICALLY – **EACH STEP IN POWER QUERY is WRITTEN in M CODE**

Accessed by selecting the **View ribbon** of **Power Query**
=> select **Advanced Editor**

M-Code is written TOP DOWN – Hence, order matters
BE CAREFUL when **reordering these steps** as it **could ruin the statement dependencies**

Write to a query formula step by using the in statement

The **last query step** is used as the **in final data set** result

Power BI – PowerQuery Editor

allows for the ability to **review and clean your data before loading it into the Power BI model**

Can be loaded in through the Transform Data selection when attempting to load in a set of data

Tools and usability in the PowerQuery editor:

- **Retrieving Database from Different Sources**
  - Can be done through the Power BI Desktop app and the Get Data selection on the top bar, different sources (e.g., SQL servers and Excel files to retrieve data) => **PowerQuery** can be used to transform the data.
- **Determine which columns / merged query's columns are kept**
  - Click on the arrow button (the one for the drop-down menu) and deselect values that you wish to not be included in the merge/database
- **Applied Steps**
  - View all of the changes performed on the database, pull an old version of the database.
- **Change entire columns' data-type**
  - Right clicking the column header and selecting the proper data-type from the Change Type selection.
- **Join tables by merging queries**
  - **Select the query you wish to merge into** then on the top bar of the home section click on Merge Queries

If the value inside a column / row / cell contains **Table** or **Value** links
=> these tables **represent relationships to other tables in the database**. They can be used to **join tables together**

By clicking on the **View ribbon tab**
=> Then selecting the **Column quality checkbox** in the **Data Preview section**

Column Quality checkbox
allows you to easily determine the **percentage of valid, error, or empty values found in each**

corresponding column in an associated selected table

Privacy Levels

Can be **configured to determine whether data can be shared between sources.**

Organizational - **allows them to share data if necessary**.

Private – **can never be shared with other data sources**
DOES NOT MEAN THAT IT CAN NEVER BE SHARED
It means that **the Power Query engine cannot share data between the sources**

Query Settings => All Properties => Uncheck Enable Load to Report

Disabling the load means **it will not load as a table to the data model**

Usually done when a query was MERGED WITH ANOTHER QUERY

Making the query be able to be modified, but will not be processed into the final report.

Distinct and Unique Values

Distinct Values

indicates how many **different values there are in the specific column**

Unique values

indicate how many values **occur only once**

Column Distribution checkbox
allows you to easily determine **unique and distinct values in each cell of the entire column**

When **distinct count === unique count**
Columns contain **all unique values**

When modelling, it is important that some model table have unique columns.

These **unique columns can be used to create one-to-many relationships**

Many-to-Many Relationships

Can be done when modelling data

An example scenario of modelling a **many-to-many relationship** is:

- relating **many regions to a single employee**
- some **employers manage one, two, or possibly more regions**
  - hence, a many-to-many relationship can be modelled

Things that Will be Taught

- In the **Load Data in Power BI Desktop** – joining of multiple tables/queries together
- In the **Load Data in Power BI Desktop** – relabel rows and cells (e.g., from misspellings (Ware House instead of Warehouse) in order to **prevent a data quality issue**
- In the **Load Data in Power BI Desktop** – you'll apply transformations to **fill in missing**

values by using the product standard cost, which is stored in the related DimProduct table in order to **prevent a data quality issue**

- In the **Load Data in Power BI Desktop** – achieve a **different shaped result** consisting of **only three columns**
  - Date
  - EmployeeKey
  - TargetAmount
- In the **Load Data in Power BI Desktop** – learn to **integrate the data** with another **DimProduct query data**

- In the **Model Data in Power BI Desktop** – learn how to create a **hierarchy** in order **to support analysis at region, country, or group level**
  - e.g., Creating a hierarchy from regions which are assigned to a country, and countries which are assigned to groups

Transform and Load Data with Power BI Desktop

When a the values **under a column header equals to Value** , this means that they are **links to another related table**

You can **select specific columns**, and a **transformation will be applied** to **join to the table**

When **selecting specific columns** to be added from **related table links** – **Uncheck** the **Use Original Column Name as Prefix** checkbox

NOTE: **Query Column Names must always be unique**. If left **checked**
the **Use Original Column Name as Prefix** option will prefix each column with the expanded column name
e.g., Product.Colour instead of just Colour

IF WE KNOW THAT the **SELECTED COLUMN NAMES DO NOT** collide with **column header names** in the **PRIMARY TABLE**

Otherwise, if unsure that **SELECTED COLUMNS WILL COLLIDE WITH COLUMN HEADER NAMES IN THE PRIMARY TABLE** then keep the **Use Original Column Name as Prefix selected**

When **selecting specific columns to be added to a table** from a **related table link** – the **column containing the links to the related table will be removed and replaced with the selected columns to be added**

General Notes

Configuring the correct data type is important

When the column **contains numeric value** – **IMPORTANT TO CHOOSE THE CORRECT TYPE FOR PERFORMING CALCULATIONS**

Fixed Decimal Number

Fixed decimal number data type stores values **with full precision**, and so **requires more storage space that decimal number**

IMPORTANT to use fixed decimal number for **financial values** or **rates** (like **exchange rates**)