




Power BI - Advanced

Grant Shannon

Nov 2024



Housekeeping

- Student check
- Installation and environment setup
 - Download install PBI desktop client
 - Connect to github repo
 - <https://github.com/gpsuser/PBI>
 - Get data files
 - <https://github.com/microsoft/powerbi-desktop-samples/blob/main/DAX/Adventure%20Works%20DW%202020.pbix>
- 10 min break on the hour (time permitting)
- Hands on course



Previously

- Data Modelling
 - Dimensional Modelling
 - Star Schema
- DAX
- Measures
- Report Generation

Agenda

- Introduction
- Data Modelling
 - Recap: Business motivation behind dimensional modelling
 - Revisit Star Schema
 - Connecting to a Table as a *Copy* vs. *Reference*
 - Update implications
 - Implement Star Schema from a single raw data table
 - Fact Table
 - Dimension Tables
 - Relationships
 - Types of Joins
 - Enriching DIM table with *Merge* (join)
- Aggregation inside Dimensional Model - without *DAX*
- Introduction to *M-language*
- Conclusion



Introduction

Power BI - recap

- Key Features:
 - **Data Connectivity:** Connects to a wide range of data sources, including Excel, databases, and cloud services
 - **Data Transformation:** Clean and transform data with Power QueryIt helps you connect, prepare, model, visualize, and share data from various sources.
 - **Data Modeling:** Create relationships between data sets and build complex models.
 - **Visualizations:** Create interactive reports and dashboards with a variety of visual tools.
 - **Sharing and Collaboration:** Share insights with your team through Power BI Service and mobile apps.

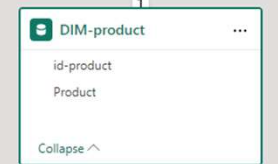
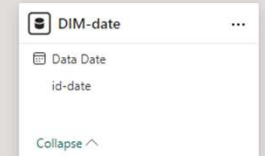
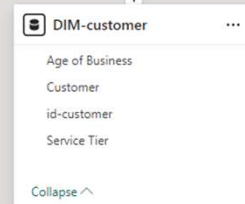
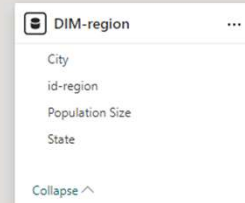
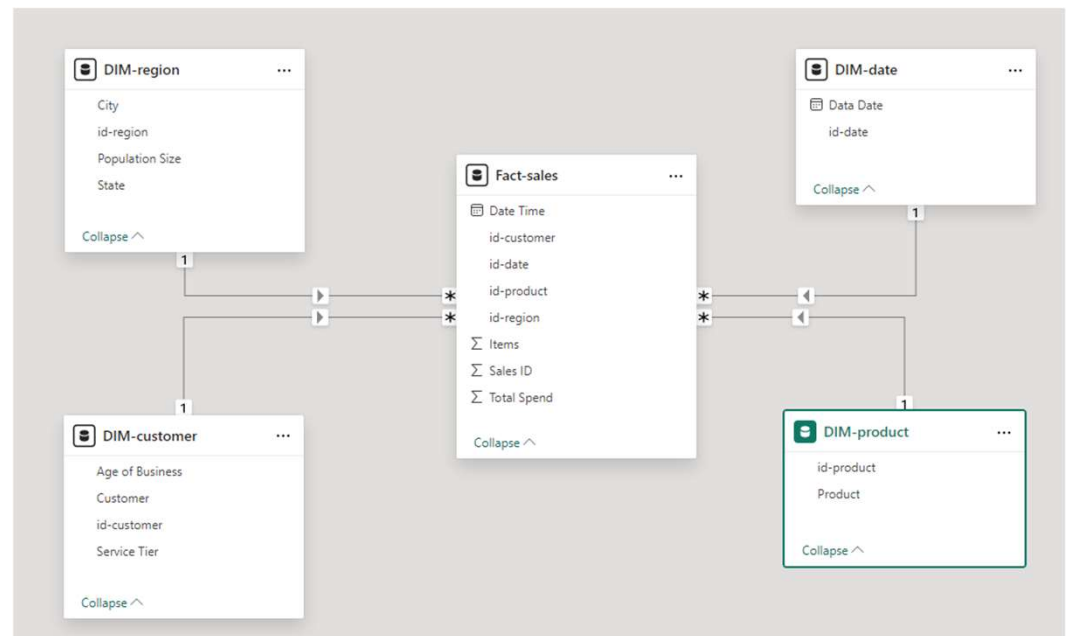
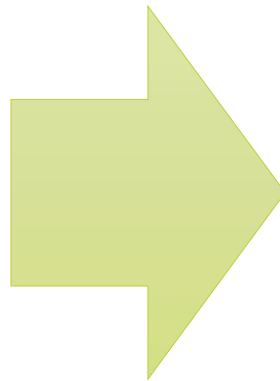
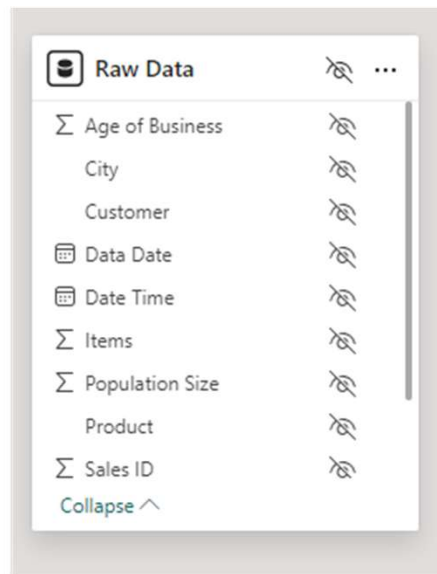
Data Modelling

Dimensional Modelling Re-cap

- *Dimensions* hold descriptive data
- *Fact* table holds quantitative / event driven data
 - Contain foreign keys that connect to DIM tables
 - Connections implemented outside of Query Editor – *Model View*

Building a dimensional model

Using a single raw data table



Getting Started

Table as reference

- First download and import data into Power BI
- Then - to get started with our dimensional model – get the *Raw Data* table as a reference

Download Data

- <https://github.com/gpsuser/PBI>
- <https://github.com/gpsuser/PBI/tree/main/data>
- data > Part3 data.zip

References vs Duplicates

- Table *References* create a link to a table
 - Changes in the original table are recorded in the original table (reference table automatically inherits these changes).
- Table *Duplicates* – creates a copy with no links
 - Copied table includes all the changes (applied steps) in the original table.
 - Changes/updates to the original table do not affect the duplicate table.
- If changes to the original query/table need to reflect in the copied table – then must create a reference table and not a duplicate table.

Creating a region Dimension table

DIM-region

Open Query Editor as follows:

- Home > Transform Data > Transform Data >
- Right click on **Raw Data** Table : reference
- Change name of reference table
 - > Right click on the **Raw Data(2)** :
 - > rename to **DIM-region**
- Choose columns for the **DIM-region** table
 - > Choose Columns : Choose Columns
 - > **City, Population Size**
- Remove duplicate rows :
 - Make sure **DIM-region** (Queries) selected
 - Select the **City** Column
 - > Remove Rows > Remove Duplicates
- Add index column (make sure **DIM-region** (Queries) still selected)
 - Add Column menu > Index Column > From 1
 - Click on **Index** column and move it to front
 - Home > Close and Apply: close and apply
- Rename index column:
 - Table view
 - > Select **DIM-region**
 - > right click on **Index** column
 - > rename: **id-region**

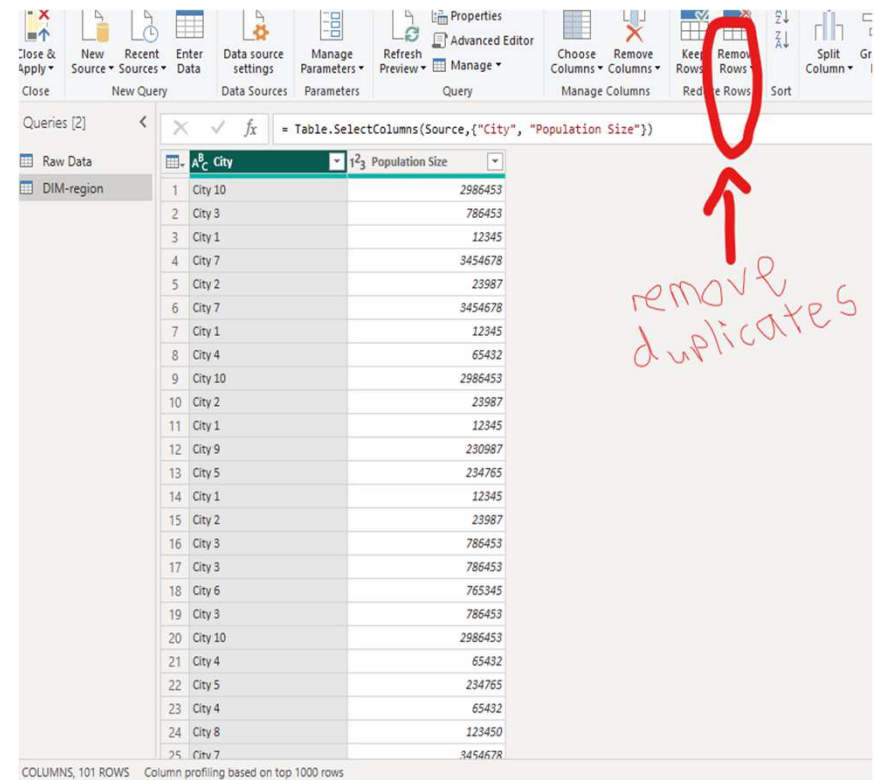


Table: SelectColumns(Source, {"City", "Population Size"})

	City	Population Size
1	City 10	2986453
2	City 3	786453
3	City 1	12345
4	City 7	3454678
5	City 2	23987
6	City 7	3454678
7	City 1	12345
8	City 4	65432
9	City 10	2986453
10	City 2	23987
11	City 1	12345
12	City 9	230987
13	City 5	234765
14	City 1	12345
15	City 2	23987
16	City 3	786453
17	City 3	786453
18	City 6	765345
19	City 3	786453
20	City 10	2986453
21	City 4	65432
22	City 5	234765
23	City 4	65432
24	City 8	123450
25	City 7	3454678

COLUMNS, 101 ROWS Column profiling based on top 1000 rows

Investigating Data Updates

Impact on DIM Tables

- Update the Raw Data table
- Monitor its impact on the *DIM-region* table (as *Reference* Table)

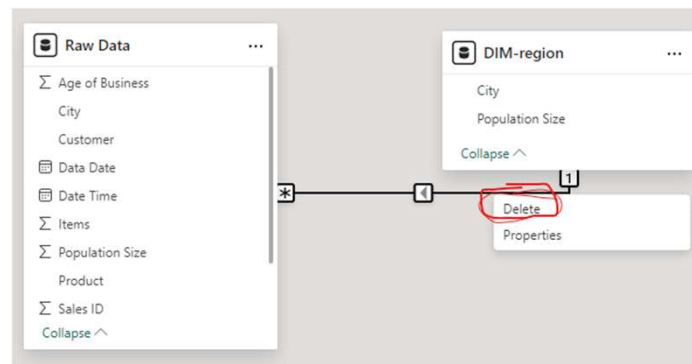
Change Raw Data

- Open *Raw Data.xlsx* and add row 101 from *extra row.xlsx*:
 - Ctrl C , Ctrl V , Ctrl Save
 - Close spreadsheets

A	B	C	D	E	F	G	H	I	J	K
Sales ID	Date Time	Items	total Spent	Product	City	Customer	Population	Size of Business	Service Tier	Data Date
97	2023-12-11 22:07:31	13	167	Product D	City 6	Customer	765345	4	Platinum	2023-12-11
98	2023-10-11 07:22:39	12	1775	Product C	City 10	Customer	2986453	8	Premium	2023-10-11
99	2023-07-09 11:21:51	15	632	Product C	City 5	Customer	234765	10	Platinum	2023-07-09
100	2023-06-28 02:51:11	12	3759	Product A	City 6	Customer	765345	15	Standard	2023-06-28
101	2023-06-16 18:20:31	9	6886	Product C	City 11	Customer	1295925	20	Platinum	2023-06-16

Check DIM table update

- Check *Raw Data.xlsx* change caused DIM-region update
 - Home > Refresh
 - Table View > Select DIM-region
- Go to Model View - and delete relationship



The screenshot shows the Power BI Table View. The table has two columns: 'City' and 'Population Size'. The data rows are as follows:

City	Population Size
City 10	2986453
City 3	786453
City 1	12345
City 7	3454678
City 2	23987
City 4	65432
City 9	230987
City 5	234765
City 6	765345
City 8	123450
City 11	1295925

The last row, 'City 11' with a population size of '1295925', is highlighted in yellow.



TABLE JOINS

- A table join in database terms is a method of combining rows from two or more tables based on a related column between them.
- This allows for the retrieval of data that is spread across multiple tables, making it easier to analyze and work with related information
- Consider:
 - Outer Joins
 - Anti Joins
 - Inner Joins

Outer Joins

Left		Right	
Table		Table	
ID	Sales	ID	Region
A	5	A	Europe
B	15	DD	Asia
C	24	E	America

1. Outer Left Join

Table	
ID	Sales
A	5
B	15
C	24

2. Outer Full Join

Table		
ID	Sales	Region
A	5	Europe
B	15	NULL
C	24	NULL
DD	NULL	Asia
E	NULL	America

3. Outer Right Join

Table	
ID	Sales
A	5
DD	NULL
E	NULL

Anti Joins

Left		Right	
Table		Table	
ID	Sales	ID	Region
A	5	A	Europe
B	15	DD	Asia
C	24	E	America

4. Anti Left Join

Table	
ID	Sales
B	15
C	24

5. Anti Right Join

Table	
ID	Region
DD	Asia
E	America

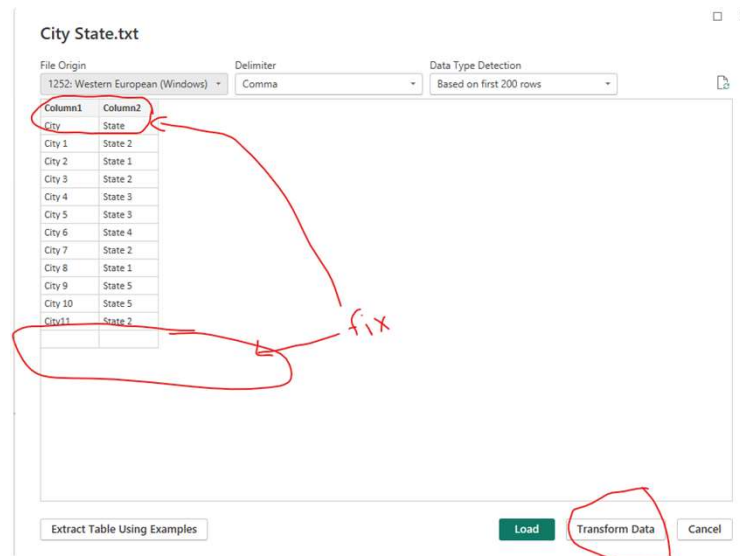
Inner Joins

Left		Right	
Table		Table	
ID	Sales	ID	Region
A	5	A	Europe
B	15	DD	Asia
C	24	E	America

6. Inner Join		
Table		
ID	Sales	Region
A	5	Europe

Enriching DIM-region table

- Want to add City- State information to DIM-region from *City State.csv*
- Home > File > Get Data > Text CSV > *City State.csv* > Open > Transform



Enriching DIM-region table

- Home > File > Get Data > Text CSV > *City State.csv* > Open > Transform
 - > Use First Row as Headers: Use first rows as headers
 - > Remove Rows > Remove Blank Rows
 - > Close and Apply : close % apply

	City	State
1	City 1	State 2
2	City 2	State 1
3	City 3	State 2
4	City 4	State 3
5	City 5	State 3
6	City 6	State 4
7	City 7	State 2
8	City 8	State 1
9	City 9	State 5
10	City 10	State 5
11	City11	State 2

Enriching DIM-region table

Merging

- Home > Transform Data: Transform data
 - > Select DIM-region (Query) > Merge Queries:
 - merge queries
 - In Merge window
 - > For DIM-region :
 - > Select Select City column (to join/merge against)
 - > For City State:
 - > Select City State
 - > Select City column
 - Join Kind
 - Left outer
 - all from first table, matching from second table
 - OK

Merge

Select a table and matching columns to create a merged table.

DIM-region

City	Population Size
City 10	2986453
City 3	786453
City 1	12345
City 7	3454678
City 2	23987

City State

City	State
City 1	State 2
City 2	State 1
City 3	State 2
City 4	State 3
City 5	State 3

Join Kind

Left Outer (all from first, matching from second)

☐ Use fuzzy matching to perform the merge

▸ Fuzzy matching options

✓ The selection matches 10 of 11 rows from the first table.

Enriching DIM-region table

Selecting merge columns

Table.NestedJoin(#"Removed Duplicates", {"City"}, #"City State", {"City"}, "City State", JoinKind.LeftOuter)

City	Population Size	City State
City 10	2986453	Table
City 3	786453	Table
City 1	12345	Table
City 7	3454678	Table
City 2	23987	Table
City 4	65432	Table
City 9	230987	Table
City 5	234765	Table
City 6	765345	Table
City 8	123450	Table
City 11	1295925	Table

Don't repeat City column – when merge

Search Columns to Expand

☒ Expand ☐ Aggregate

(Select All Columns)

☐ City

☒ State

☒ Use original column name as prefix

OK Cancel

Table.ExpandTableColumn(#"Merged Queries", "City State", {"State"})

City	Population Size	City State.State
City 10	2986453	State 5
City 1	12345	State 2
City 3	786453	State 2
City 2	23987	State 1
City 7	3454678	State 2
City 4	65432	State 3
City 5	234765	State 3
City 6	765345	State 4
City 9	230987	State 5
City 8	123450	State 1
City 11	1295925	State 2

Close and Apply

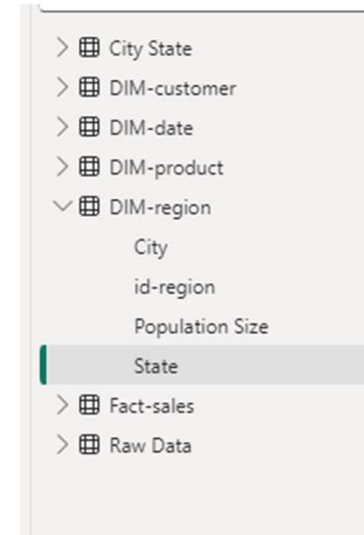
Rename State Column

- Home > Table View

From *Data* blade > *Dim-region*

> Select *City State.State* + right click

> rename: *State*





Creating the remaining dimensions

- *DIM-Customer*
- *DIM-Product*
- *DIM-Date*

Creating a customer Dimension table

DIM-customer

Open Query Editor as follows:

- Home > Transform Data > Transform Data >
- Right click on **Raw Data** Table : reference
- Change name of reference table
 - > Right click on the **Raw Data(2)** :
 - > rename to **DIM-customer**
- Choose columns for the **DIM-customer** table
 - > Choose Columns : Choose Columns
 - > **Customer, Age of Business, Service Tier**
- Remove duplicate rows :
 - Make sure **DIM-customer** (Queries) selected
 - Select the **Customer** Column
 - > Remove Rows > Remove Duplicates
- Add index column (make sure **DIM-customer** (Queries) still selected)
 - Add Column menu > Index Column > From 1
 - Click on **Index** column and move it to front
 - Home > Close and Apply: close and apply
- Rename index column:
 - Table view
 - > Select **DIM-customer**
 - > right click on **Index** column
 - > rename: **id-customer**

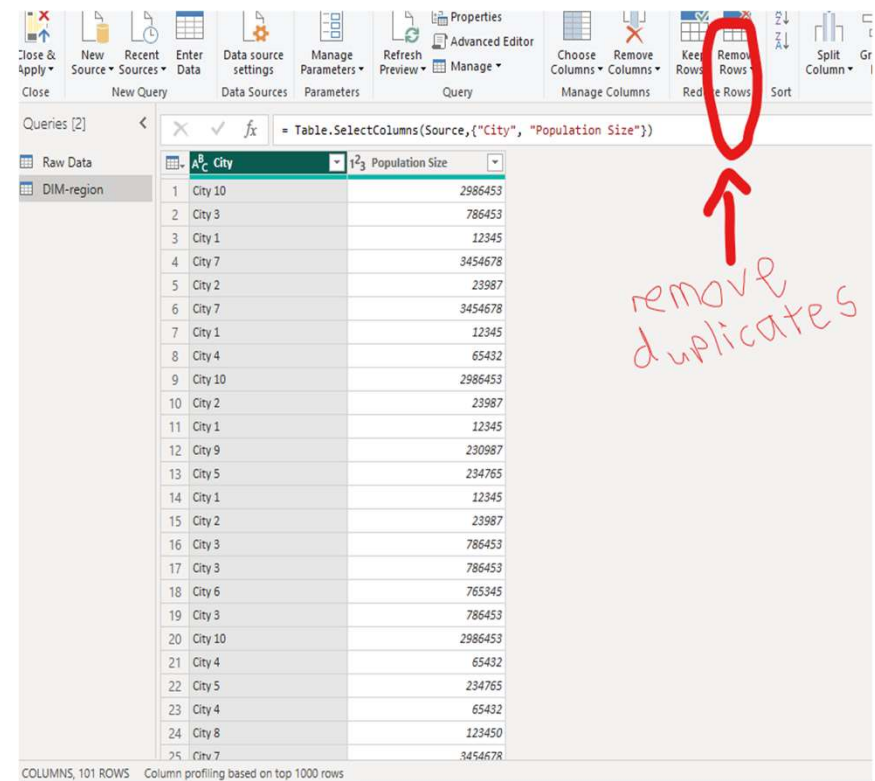


Table: SelectColumns(Source, {"City", "Population Size"})

	City	Population Size
1	City 10	2986453
2	City 3	786453
3	City 1	12345
4	City 7	3454678
5	City 2	23987
6	City 7	3454678
7	City 1	12345
8	City 4	65432
9	City 10	2986453
10	City 2	23987
11	City 1	12345
12	City 9	230987
13	City 5	234765
14	City 1	12345
15	City 2	23987
16	City 3	786453
17	City 3	786453
18	City 6	765345
19	City 3	786453
20	City 10	2986453
21	City 4	65432
22	City 5	234765
23	City 4	65432
24	City 8	123450
25	City 7	3454678

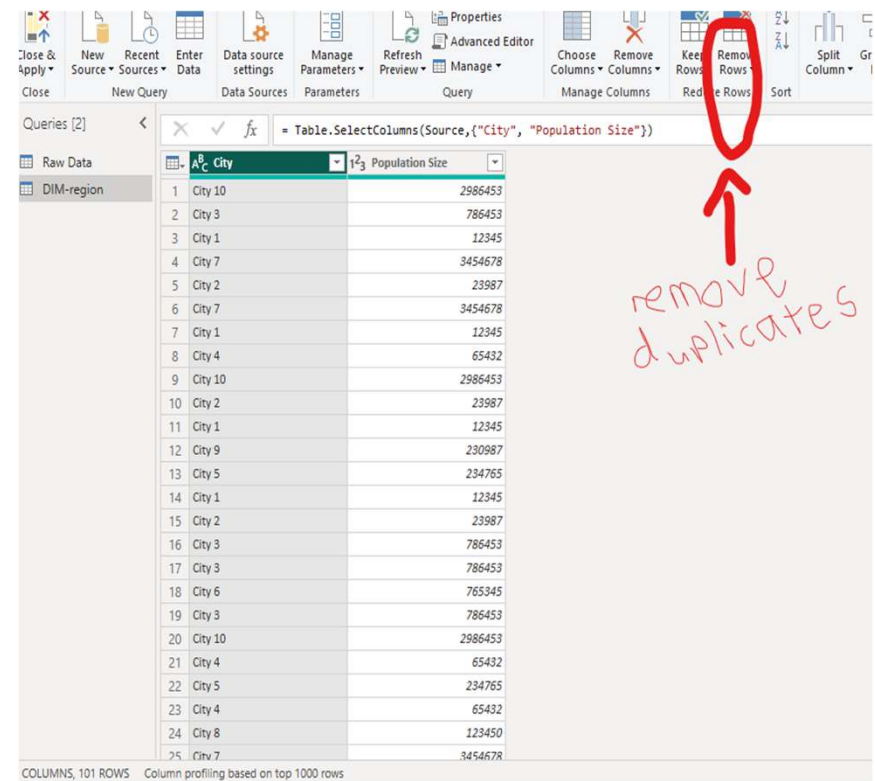
COLUMNS, 101 ROWS Column profiling based on top 1000 rows

Creating a product Dimension table

DIM-product

Open Query Editor as follows:

- Home > Transform Data > Transform Data >
- Right click on **Raw Data** Table : reference
- Change name of reference table
 - > Right click on the **Raw Data(2)** :
 - > rename to **DIM-product**
- Choose columns for the **DIM-product** table
 - > Choose Columns : Choose Columns
 - > **Product**
- Remove duplicate rows :
 - Make sure **DIM-product** (Queries) selected
 - Select the **Product** Column
 - > Remove Rows > Remove Duplicates
- Add index column (make sure **DIM-product** (Queries) still selected)
 - Add Colum menu > Index Column > From 1
 - Click on **Index** column and move it to front
 - Home > Close and Apply: close and apply
- Rename index column:
 - Table view
 - > Select **DIM-product**
 - > right click on **Index** column
 - > rename: **id-product**



The screenshot shows the Power BI Query Editor interface. The ribbon at the top includes the 'Remove Rows' button, which is circled in red. A red arrow points to this button, and the handwritten text 'remove duplicates' is written next to it. The main area displays a table with two columns: 'City' and 'Population Size'. The table contains 25 rows of data, with some cities appearing multiple times, indicating duplicates.

	City	Population Size
1	City 10	2986453
2	City 3	786453
3	City 1	12345
4	City 7	3454678
5	City 2	23987
6	City 7	3454678
7	City 1	12345
8	City 4	65432
9	City 10	2986453
10	City 2	23987
11	City 1	12345
12	City 9	230987
13	City 5	234765
14	City 1	12345
15	City 2	23987
16	City 3	786453
17	City 3	786453
18	City 6	765345
19	City 3	786453
20	City 10	2986453
21	City 4	65432
22	City 5	234765
23	City 4	65432
24	City 8	123450
25	City 7	3454678

Creating a date Dimension table

DIM-date

Open Query Editor as follows:

- Home > Transform Data > Transform Data >
- Right click on **Raw Data** Table : reference
- Change name of reference table
 - > Right click on the **Raw Data(2)** :
 - > rename to **DIM-date**
- Choose columns for the **DIM-date** table
 - > Choose Columns : Choose Columns
 - > **Data Date**
- Remove duplicate rows :
 - Make sure **DIM-date** (Queries) selected
 - Select the **Data Date** Column
 - > Remove Rows > Remove Duplicates
- Add index column (make sure **DIM-date** (Queries) still selected)
 - Add Column menu > Index Column > From 1
 - Click on **Index** column and move it to front
 - Home > Close and Apply: close and apply
- Rename index column:
 - Table view
 - > Select **DIM-date**
 - > right click on **Index** column
 - > rename: **id-date**

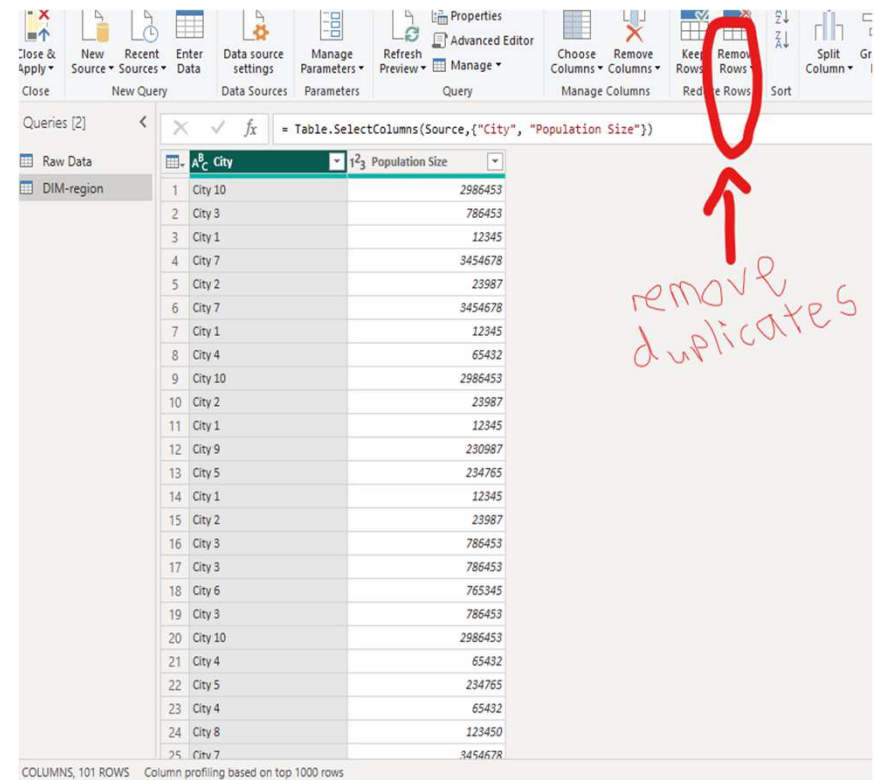


Table: Table.SelectColumns(Source,{'City', 'Population Size'})

	City	Population Size
1	City 10	2986453
2	City 3	786453
3	City 1	12345
4	City 7	3454678
5	City 2	23987
6	City 7	3454678
7	City 1	12345
8	City 4	65432
9	City 10	2986453
10	City 2	23987
11	City 1	12345
12	City 9	230987
13	City 5	234765
14	City 1	12345
15	City 2	23987
16	City 3	786453
17	City 3	786453
18	City 6	765345
19	City 3	786453
20	City 10	2986453
21	City 4	65432
22	City 5	234765
23	City 4	65432
24	City 8	123450
25	City 7	3454678

COLUMNS, 101 ROWS Column profiling based on top 1000 rows

Creating a Fact table

Fact-sales

Open Query Editor as follows:

- Home > Transform Data > Transform Data >
- Right click on *Raw Data* Table : reference
- Change name of reference table
 - > Right click on the *Raw Data(2)* :
 - > rename to *FACT-sales*
- Next - Need to bring in ID columns from DIM tables

Merge product-id into Fact-sales

Using Query Editor - bring in *product-id*, using Merge - make sure *Fact-sales* (Queries) is selected
Merge Queries: Merge Queries

- > Select *Product* column in *Fact-Sales*
- > From dropdown choose: *Dim-product*
 - > Select *Product* column in *DIM-product*
 - > Left Outer Join
 - > OK

Expand *DIM-product* (click on two diverging arrows icon)

- > Select expand
- > Select *id-product*
- > De-select *use original column name as prefix*
- > OK
- > Delete relevant *Product* columns in *Fact-sales*
 - > Select *Product* column
 - > Right-click: remove columns
- > Close and Appl: Close and Apply

Merge customer-id into Fact-sales

Using Query Editor - bring in *customer-id*, using Merge - make sure *Fact-sales* (Queries) is selected
Merge Queries: Merge Queries

- > Select *Customer* column in *Fact-Sales*
- > From dropdown choose: *Dim-customer*
 - > Select *Customer* column in *DIM-customer*
 - > Left Outer Join
 - > OK

Expand *DIM-customer* (click on two diverging arrows icon)

- > Select expand
- > Select *id-customer*
- > De-select *use original column name as prefix*
- > OK
- > Delete relevant *Customer* columns in *Fact-sales*
 - > Select *Customer* + *Ctrl* + other columns to delete (*Age of Business, Service Tier*)
 - > Right-click: remove columns

Merge region-id into Fact-sales

Using Query Editor - bring in *region-id*, using Merge - make sure *Fact-sales* (Queries) is selected
Merge Queries: Merge Queries

- > Select *City* column in *Fact-Sales*

- > From dropdown choose: *Dim-region*
 - > Select *City* column in *DIM-region*
 - > Left Outer Join
 - > OK

Expand *DIM-region* (click on two diverging arrows icon)

- > Select expand
- > Select *id-region*
- > De-select *use original column name as prefix*
- > OK
- > Delete relevant *Region* columns in *Fact-sales*
 - > Select *City* + *Ctrl* + other to delete (*Population Size*)
 - > Right-click: remove columns

Merge date-id into Fact-sales

Using Query Editor - bring in *date-id*, using Merge - make sure *Fact-sales* (Queries) is selected
Merge Queries: Merge Queries

> Select *Data Date* column in *Fact-Sales*

> From dropdown choose: *Dim-date*

> Select *Data Date* column in *DIM-date*

> Left Outer Join

> OK

Expand *DIM-date* (click on two diverging arrows icon)

> Select expand

> Select *id-date*

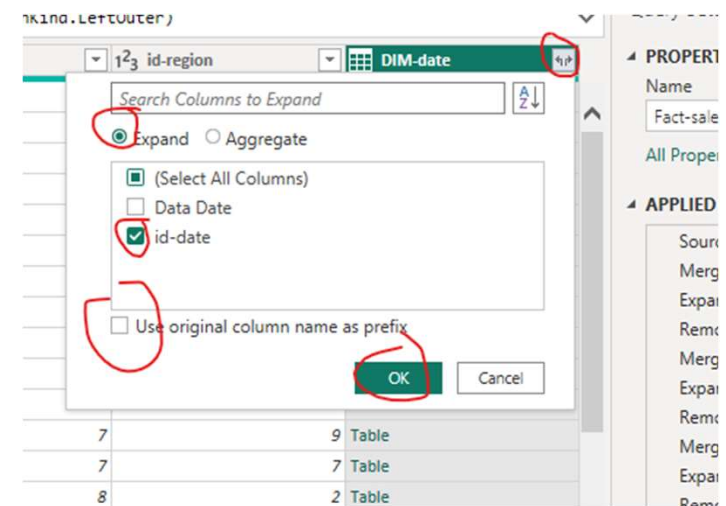
> De-select *use original column name as prefix*

> OK

> Delete relevant *Date* columns in *Fact-sales*

> Select *Data Date*

> Right-click: remove

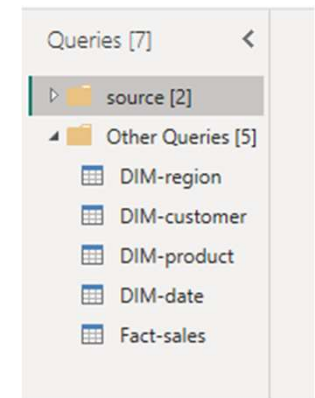
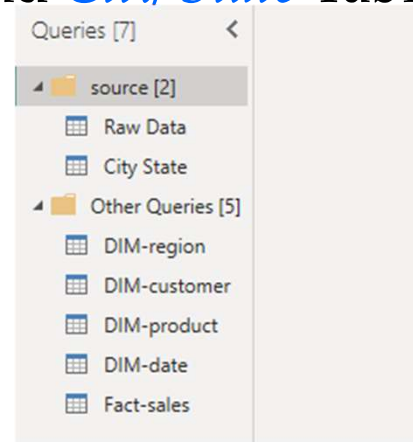


Create folder for source data

Home > Transform data : Transform data (takes you to: *Query Editor*)

- Right click on *Queries* panel > *New Group* > *Name: source*
- Click on and drag/Move *Raw Data* and *City State* Tables into *source* folder

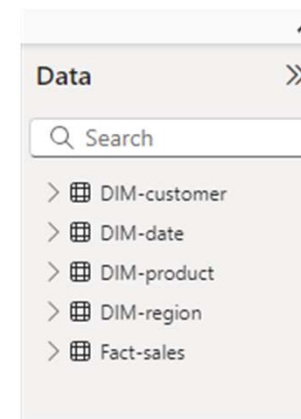
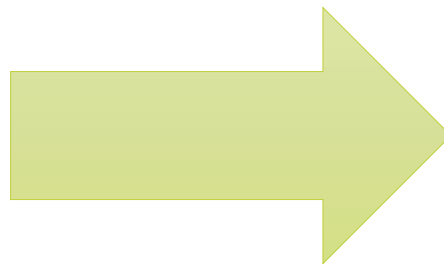
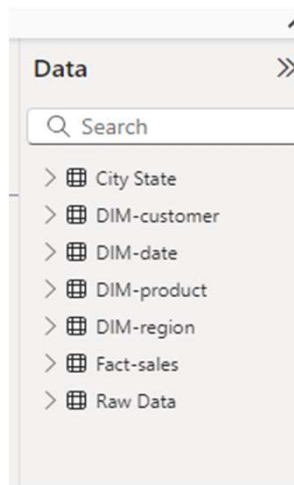
> Close and Apply: close and apply



Hide source data

Home > Data blade

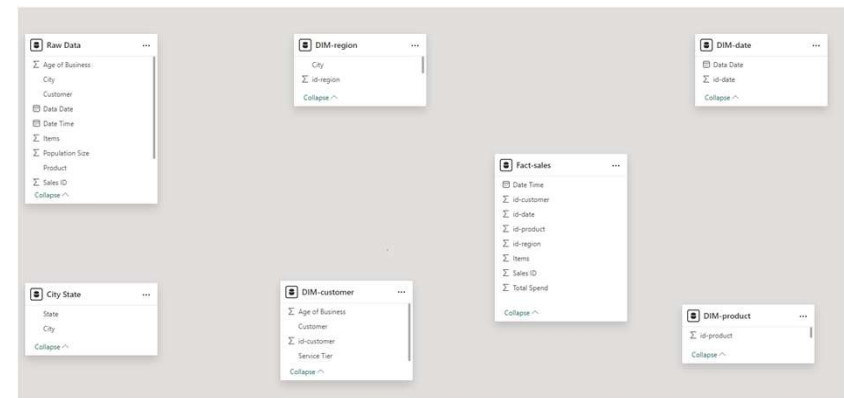
- Right click on *City State* > Hide
- Right click on *Raw Data* > Hide



Setting up relationships in Model View

Home

Delete all connections to all tables

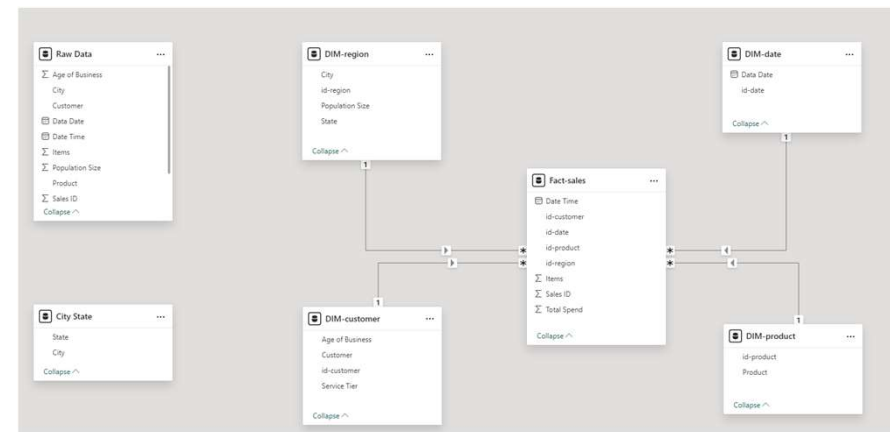
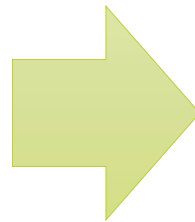
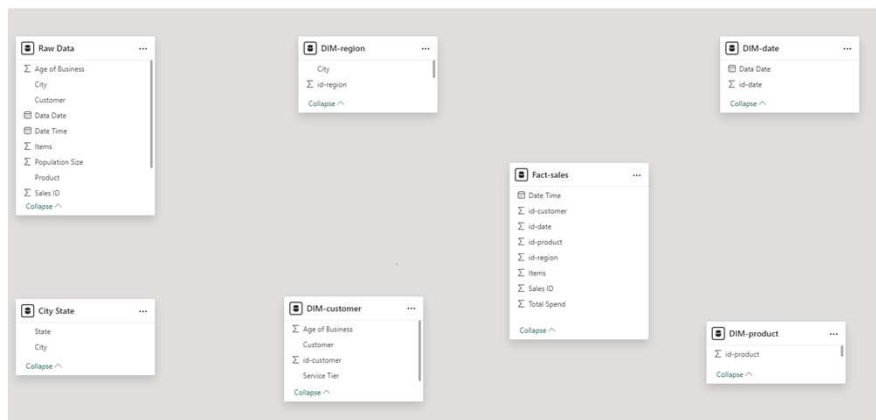


Setting up relationships in Model View

Create Fact and DIM connections

Home > Open Model View

- Create many to one connections by:
 - Connecting (select, drag, drop) *Fact* table *id* columns (FOREIGN KEYS) to respective *DIM* table *id* columns (PRIMARY KEYS)



Aggregation - no DAX

Group By

In Query Editor

- Queries pane > Select Fact-sales > Right click: *Reference*

Rename Reference Table

> Click *Fact-sales (2)* > Right click: *rename: TotSalesByCustomerByProduct*

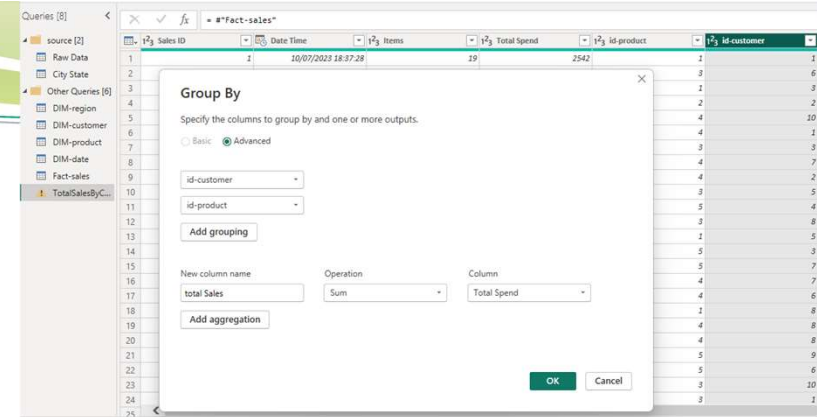
Apply Aggregation: Group By

In *TotalSalesByCustomerByProduct* select the column to group by

> Select *id-customer*

> Transform Tab > Group By > Advanced > Add Grouping > *id-product*

> New Column name: *Total Sales* > Operation: *Sum* > Column: *Total Spend* > *OK*



Aggregation - no DAX

Joins - for labeling *Customer*

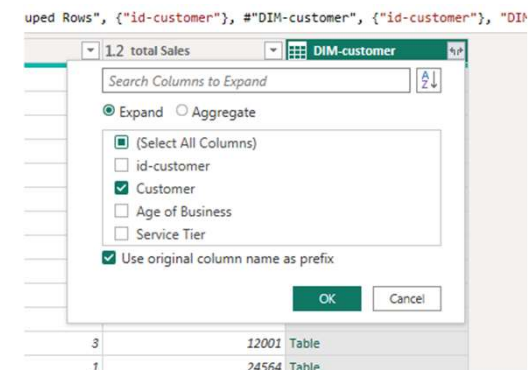
In Query Editor > *Home* tab with *TotalSalesByCustomerByProduct* (Queries) selected > Select *id-customer* column

Join (with Merge)

Merge Queries : merge queries > select *id-customer* for *TotalSalesByCustomerByProduct*

In dropdown > select *DIM-customer* > *id-customer*
> Left outer join

Expand *DIM-customer* > Select *Customer*
> De-select *Use original column name as prefix* > OK



Aggregation - no DAX

Joins - for labeling *Product*

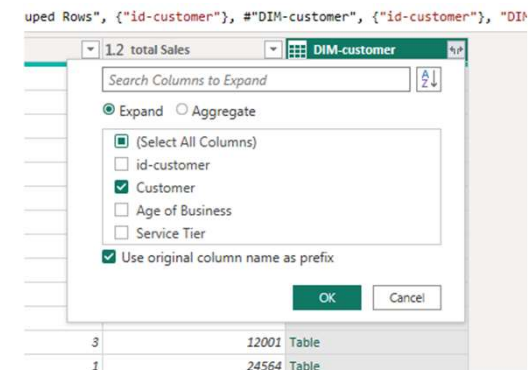
In Query Editor > *Home* tab with *TotalSalesByCustomerByProduct* (Queries) selected > Select *id-product* column

Join (with Merge)

Merge Queries : merge queries > select *id-product* for *TotalSalesByCustomerByProduct*

In dropdown > select *DIM-product* > *id-product*
> Left outer join

Expand *DIM-productr* > Select *Product*
> De-select *Use original column name as prefix* > OK



Delete and Move Columns

- *TotalSalesByCustomerByProduct* (Queries) selected
- Home tab
- Select *id-customer* + *Ctrl* Select *id-product* > right-click: remove columns
- Select Customer + *Ctrl* Select Product > Select Transform tab > Move > Left
- Home > Close and Apply: close and apply

Delete *TotalSalesByCustomerByProduct* relationships/connections in model View



M Language

- M language is used in the Power Query Editor (mostly auto generated code)
 - Power Query formula language (data preparation)
 - Data transformation (before loading data into data model)
- Dax
 - Data Analysis Expression Language (create insights)
 - Analytical Data Calculations (inside data model)
 - Similar to Excel functions

Creating a column with M - language

- There is a visual representation of the transformation steps/instructions you apply in the Power Query Editor
 - You can reverse/undo these steps
- M language is the script behind these transformation steps/instructions
 - If you select a step - you notice the formula bar shows the relevant M language script for that step
 - (expand the formula bar with the down arrow)
 - Manually add/change code in the M language formula bar
- The advanced editor exposes all the M language code behind a Power Query table

M language - behind each step

The screenshot displays the Microsoft Power BI Desktop interface. The top ribbon shows the 'Query' tab, with the 'M Language' section highlighted. The query editor shows the following M language formula:

```
= Table.ReorderColumns("#Removed Columns",{"DIM-customer.Customer", "Product", "total Sales"})
```

The formula is circled in red, and the text 'M Language' is written in red below it. The data view shows a table with the following columns: DIM-customer.Customer, Product, and total Sales. The table contains 15 rows of data.

The 'Query Settings' pane on the right shows the query name 'TotalSalesByCustomerByProduct'. Under the 'APPLIED STEPS' section, the following steps are listed:

- Source
- Grouped Rows
- Merged Queries
- Expanded DIM-customer
- Merged Queries1
- Expanded DIM-product
- Removed Columns
- Reordered Columns** (highlighted with a red box and a red circle)
- Renamed Columns

A red bracket groups the last four steps (Removed Columns, Reordered Columns, and Renamed Columns) with the word 'steps' written in red next to it.

	DIM-customer.Customer	Product	total Sales
1	Customer A	Product A	2542
2	Customer D	Product A	14925
3	Customer A	Product B	21543
4	Customer J	Product B	16382
5	Customer J	Product C	4722
6	Customer E	Product D	24692
7	Customer D	Product D	19892
8	Customer H	Product E	12748
9	Customer G	Product B	5309
10	Customer B	Product D	6841
11	Customer C	Product B	19415
12	Customer F	Product D	12001
13	Customer B	Product A	24564
14	Customer D	Product E	597
15	Customer C	Product E	10074

Use M to create column

Power Query Editor

Select *TotalSalesByCustomerByProduct* (Queries)

> Add Column tab

> Select last Step in *Applied Steps*

> Custom Column

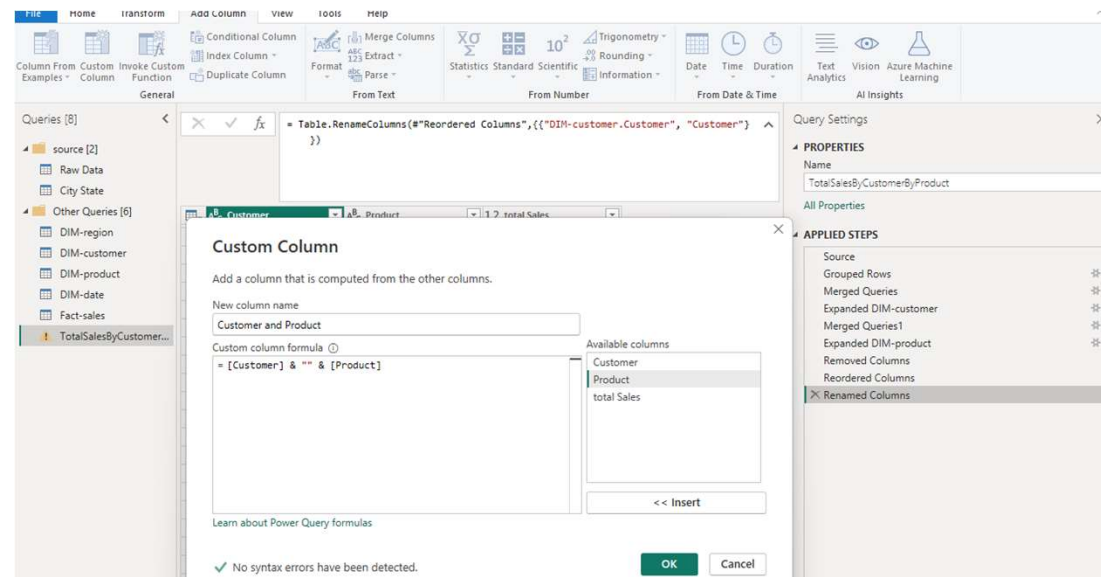
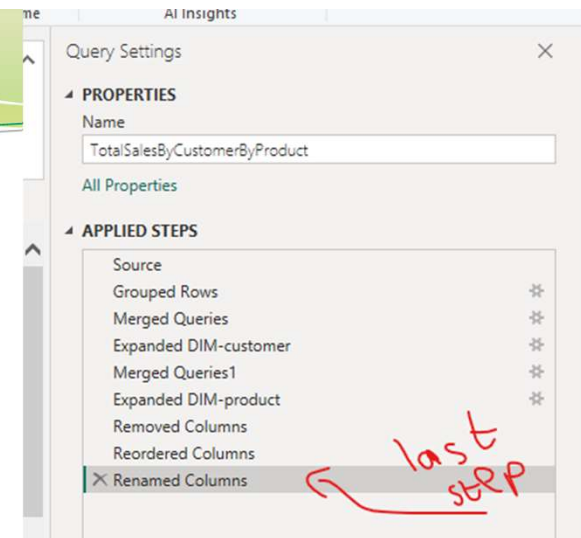
> New Column Name:

Customer and Product

> Custom column formula

= [Customer] & " " & [Product]

> OK



M language script – to create column

```
= Table.AddColumn("#Renamed Columns", "Customer and Product", each [Customer] & " " & [Product])
```

The screenshot displays the Microsoft Power BI Desktop interface. The top ribbon shows various data transformation options. The central area is the M language script editor, which contains the following script:

```
= Table.AddColumn("#Renamed Columns", "Customer and Product", each [Customer] & " " & [Product])
```

Below the script editor, a data table is shown with the following columns: Customer, Product, total Sales, and Customer and Product. The table contains 14 rows of data.

	Customer	Product	total Sales	Customer and Product
1	Customer A	Product A	2542	Customer AProduct A
2	Customer D	Product A	14925	Customer DProduct A
3	Customer A	Product B	21543	Customer AProduct B
4	Customer J	Product B	16382	Customer JProduct B
5	Customer J	Product C	4722	Customer JProduct C
6	Customer E	Product D	24692	Customer EProduct D
7	Customer D	Product D	19892	Customer DProduct D
8	Customer H	Product E	12748	Customer HProduct E
9	Customer G	Product B	5309	Customer GProduct B
10	Customer B	Product D	6841	Customer BProduct D
11	Customer C	Product B	19415	Customer CProduct B
12	Customer F	Product D	12001	Customer FProduct D
13	Customer B	Product A	24564	Customer BProduct A
14	Customer D	Product E	597	Customer DProduct E

On the right side of the interface, the 'Query Settings' pane is open, showing the 'PROPERTIES' tab with the query name 'TotalSalesByCustomerByProduct' and the 'APPLIED STEPS' tab showing the sequence of transformations applied to the data.



Conclusion

Conclusion

Single most important rule:

- *Practice*

Introduction

Data Modelling

Recap: Business motivation behind dimensional modelling

Revisit Star Schema

Connecting to a Table as a *Copy* vs. *Reference*

Update implications

Implement Star Schema from a single raw data table

Fact Table

Dimension Tables

Relationships

Types of Joins

Enriching DIM table with *Merge* (join)

Aggregation inside Dimensional Model - without *DAX*

Introduction to *M-language*

Conclusion