



Data Science

Data Analytic Expression (DAX)

Asst. Prof. Dr. Santitham Prom-on

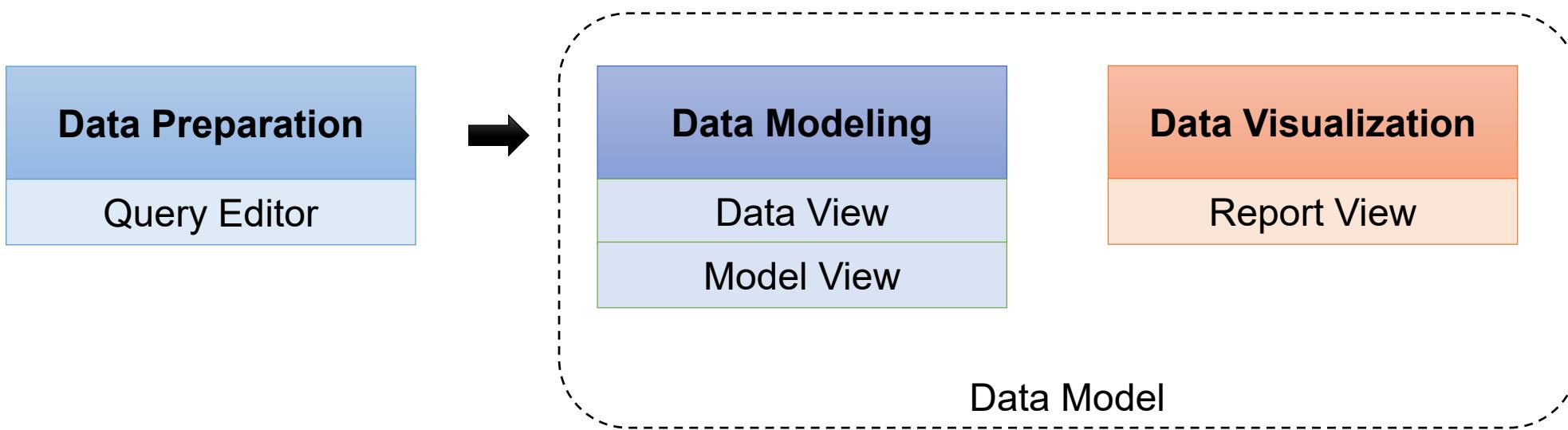
Department of Computer Engineering, Faculty of Engineering
King Mongkut's University of Technology Thonburi



Part 1

- Power BI workflow
- DAX engine and evaluations
- Basic DAX functions
 - DAX Aggregation

Power BI desktop Workflow





Microsoft Adventure Works Database

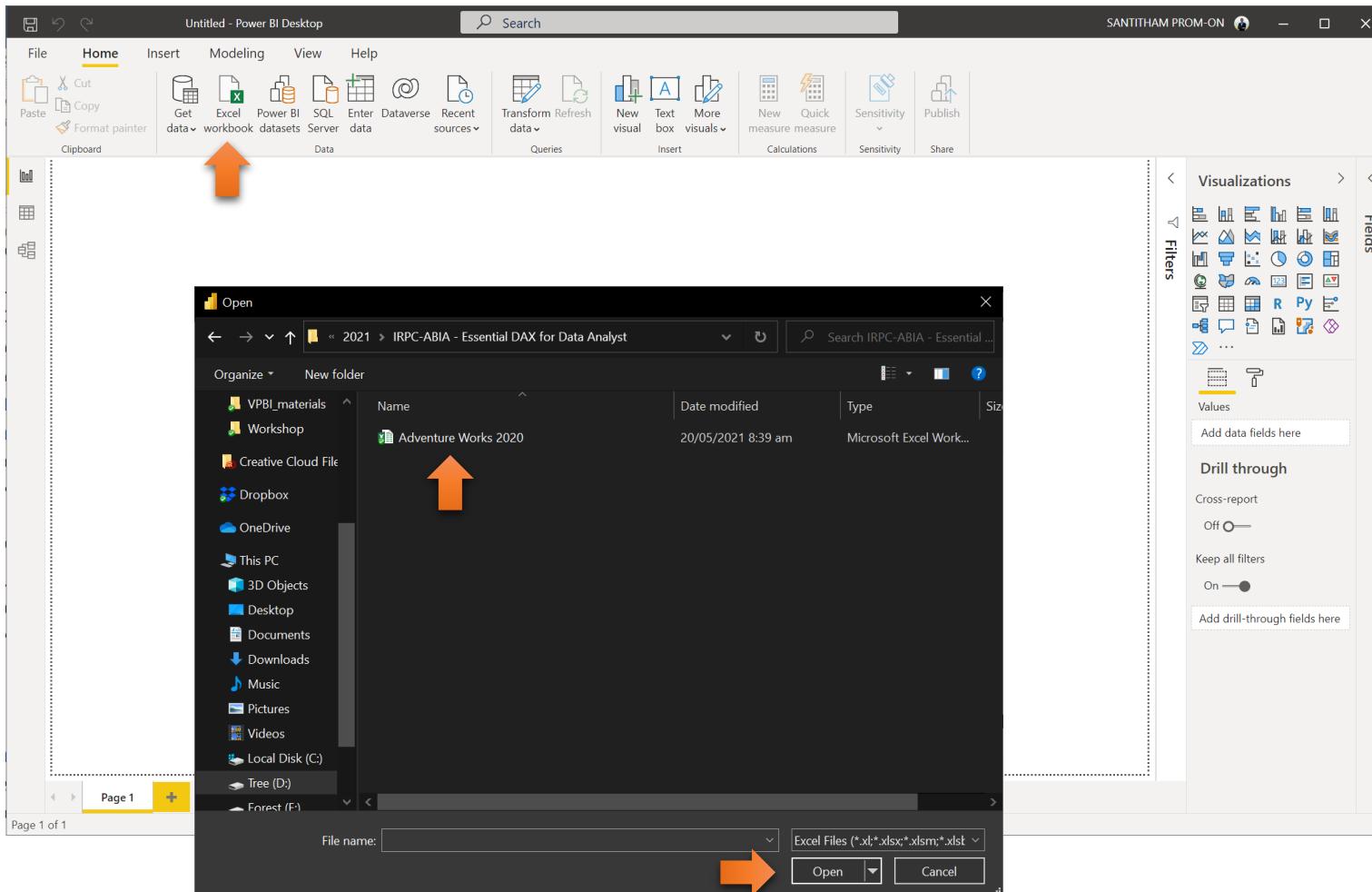
AdventureWorks is a worldwide chain of bicycle stores that has retail stores in countries including Australia, the United States, Canada, France, and the United Kingdom.

It consists of the following tables

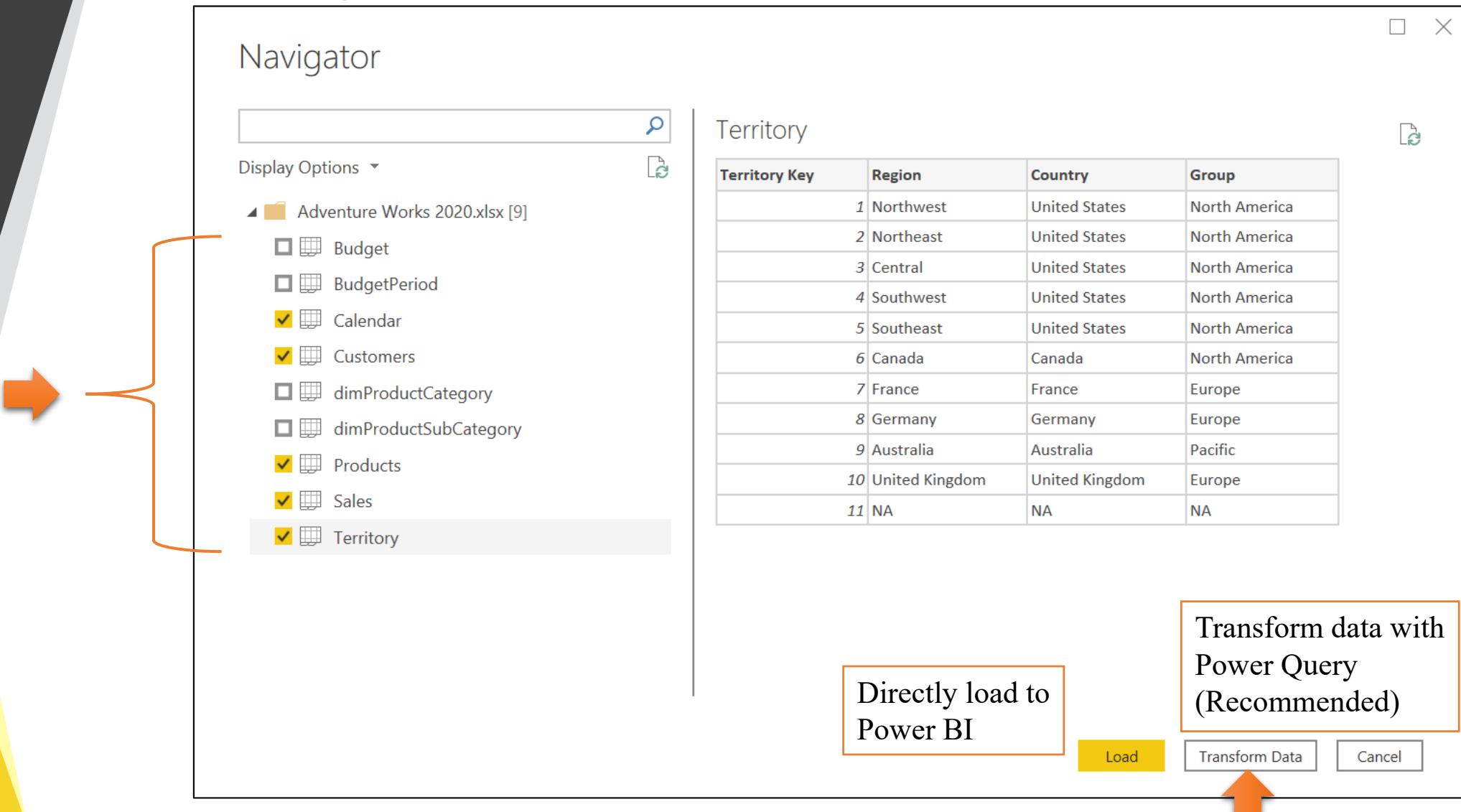
- Calendar
- Products
- Customers
- Territory
- Sales
- dimProductSubCategory
- dimProductCategory
- BudgetPeriod
- Budget

Tax Invoice / Receipt Adventure Works Australia		
OrderNo:	SO57418	4/11/2018
Bill To:	Jon Yang	
Cust No.:	11000	
Item	Description	Amount
573	Touring-1000 Blue, 46	\$ 2,384.07
541	Touring Tire	\$ 28.99
530	Touring Tire Tube	\$ 4.99
214	Sport-100 Helmet, Red	\$ 34.99
488	Short-Sleeve Classic Jersey, S	\$ 53.99
Subtotal:		\$ 2,507.03
Total ex tax:		\$ 2,507.03
Tax (8%):		\$ 200.56
Total Inc tax:		\$ 2,707.59

Import data



Navigator



The screenshot shows the Power BI Navigator interface. On the left, a list of data sources from 'Adventure Works 2020.xlsx' is shown, with several items checked: Calendar, Customers, Products, Sales, and Territory. An orange arrow points from the 'Territory' checkbox to the Territory table on the right. The Territory table has columns: Territory Key, Region, Country, and Group. It lists 11 rows of data. At the bottom right of the Navigator window, there are three buttons: 'Load', 'Transform Data', and 'Cancel'. A callout box highlights the 'Transform Data' button with the text: 'Transform data with Power Query (Recommended)'. Another callout box highlights the 'Load' button with the text: 'Directly load to Power BI'.

Territory Key	Region	Country	Group
1	Northwest	United States	North America
2	Northeast	United States	North America
3	Central	United States	North America
4	Southwest	United States	North America
5	Southeast	United States	North America
6	Canada	Canada	North America
7	France	France	Europe
8	Germany	Germany	Europe
9	Australia	Australia	Pacific
10	United Kingdom	United Kingdom	Europe
11	NA	NA	NA

Navigator

Display Options ▾

- Adventure Works 2020.xlsx [9]
 - Budget
 - BudgetPeriod
 - Calendar
 - Customers
 - dimProductCategory
 - dimProductSubCategory
 - Products
 - Sales
 - Territory

Transform data with Power Query (Recommended)

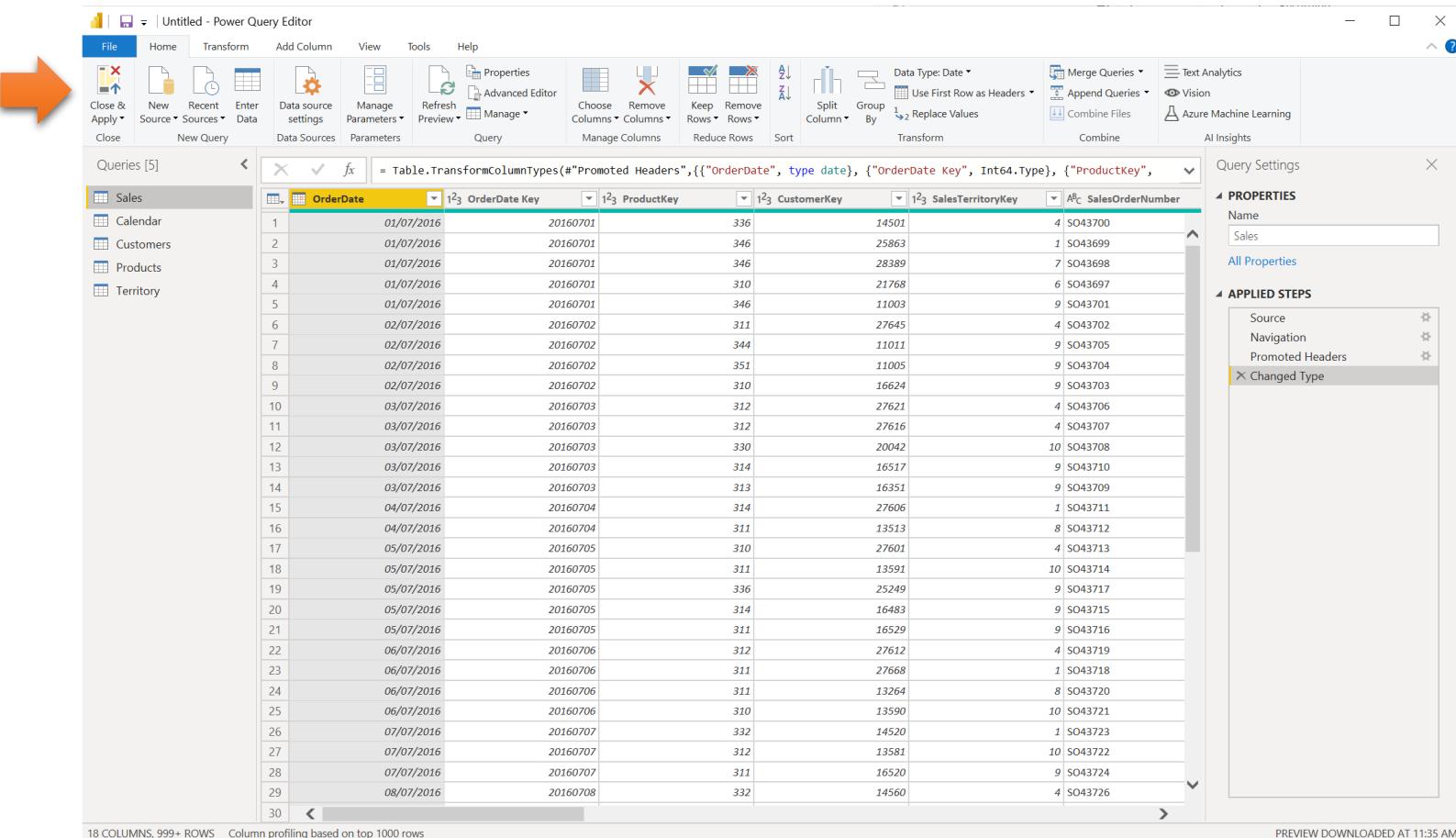
Directly load to Power BI

Load

Transform Data

Cancel

Power Query



The screenshot shows the Microsoft Power Query Editor interface. On the left, a sidebar lists five queries: Sales, Calendar, Customers, Products, and Territory. An orange arrow points to the 'Sales' query. The main area displays a table with 18 columns and over 900 rows, representing transformed data. The columns include OrderDate, OrderDate Key, ProductKey, CustomerKey, SalesTerritoryKey, and SalesOrderNumber. The 'Properties' pane on the right shows the query is named 'Sales'. The 'Applied Steps' pane at the bottom right lists the following steps:

- Source
- Navigation
- Promoted Headers
- Changed Type

At the bottom of the editor, it says '18 COLUMNS, 999+ ROWS' and 'Column profiling based on top 1000 rows'. The status bar at the bottom right indicates 'PREVIEW DOWNLOADED AT 11:35 AM'.



Untitled - Power BI Desktop

SANTITHAM PROM-ON

File Home Insert Modeling View Help

Paste Cut Copy Format painter Clipboard

Get data Excel Power BI datasets SQL Server Enter data Recent sources Data

Transform Refresh data New visual Text box More visuals Queries

New visual Text box More visuals Insert Calculations Share

Power BI Desktop interface showing the Home tab selected. A red box highlights the "Views" icon (three squares) in the ribbon. To the right, there are three main sections: Filters, Visualizations, and Fields, each with search bars and various configuration options. Below these sections are icons for Report View, Data View, and Model View, each with a corresponding yellow vertical bar to its left.

Views

Main workflow of Power BI Desktop. User can switch between (1) Report View, (2) Data View, and (3) Model View

Go to Report View

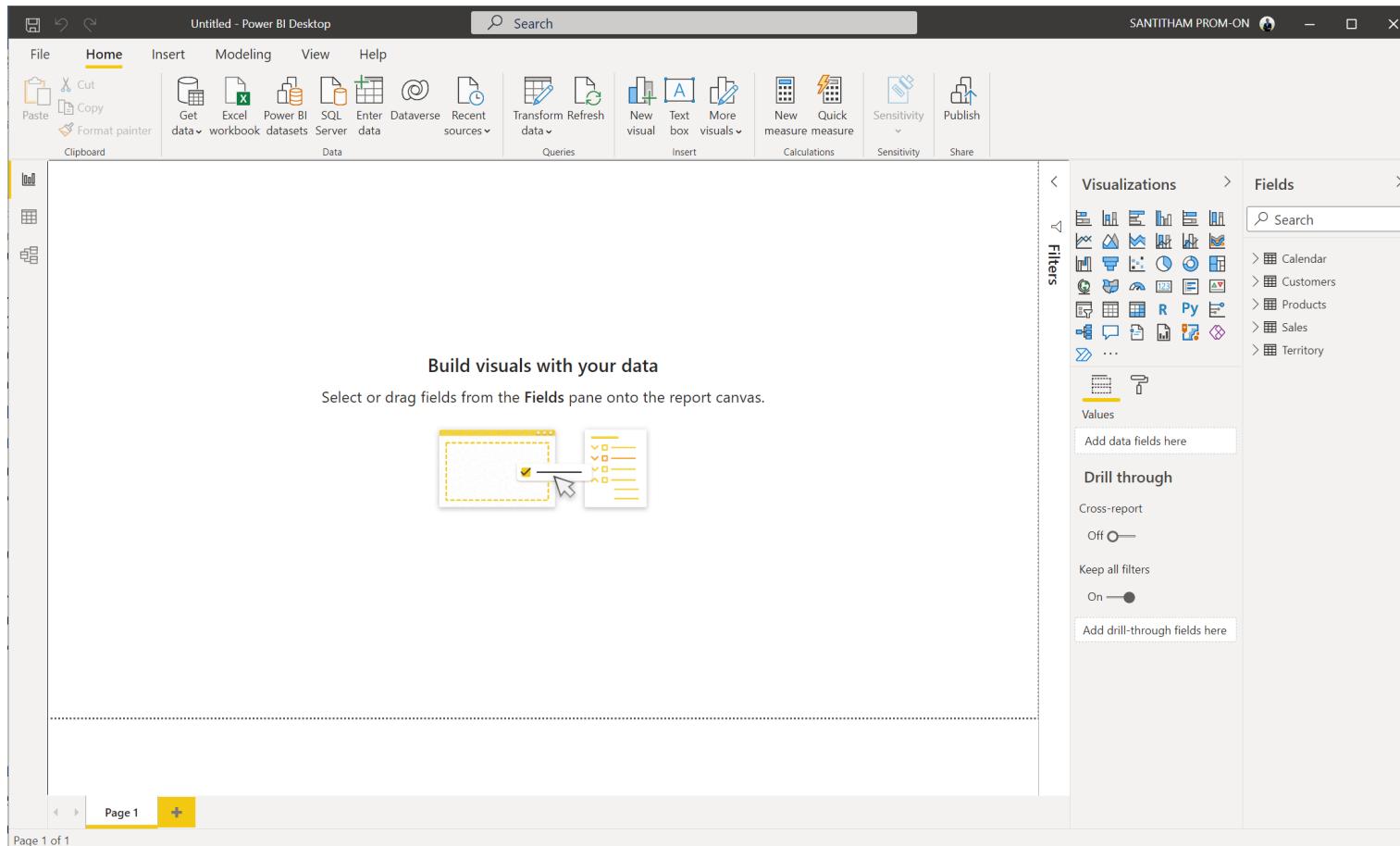
Go to Data View

Go to Model View

Report View

Report View

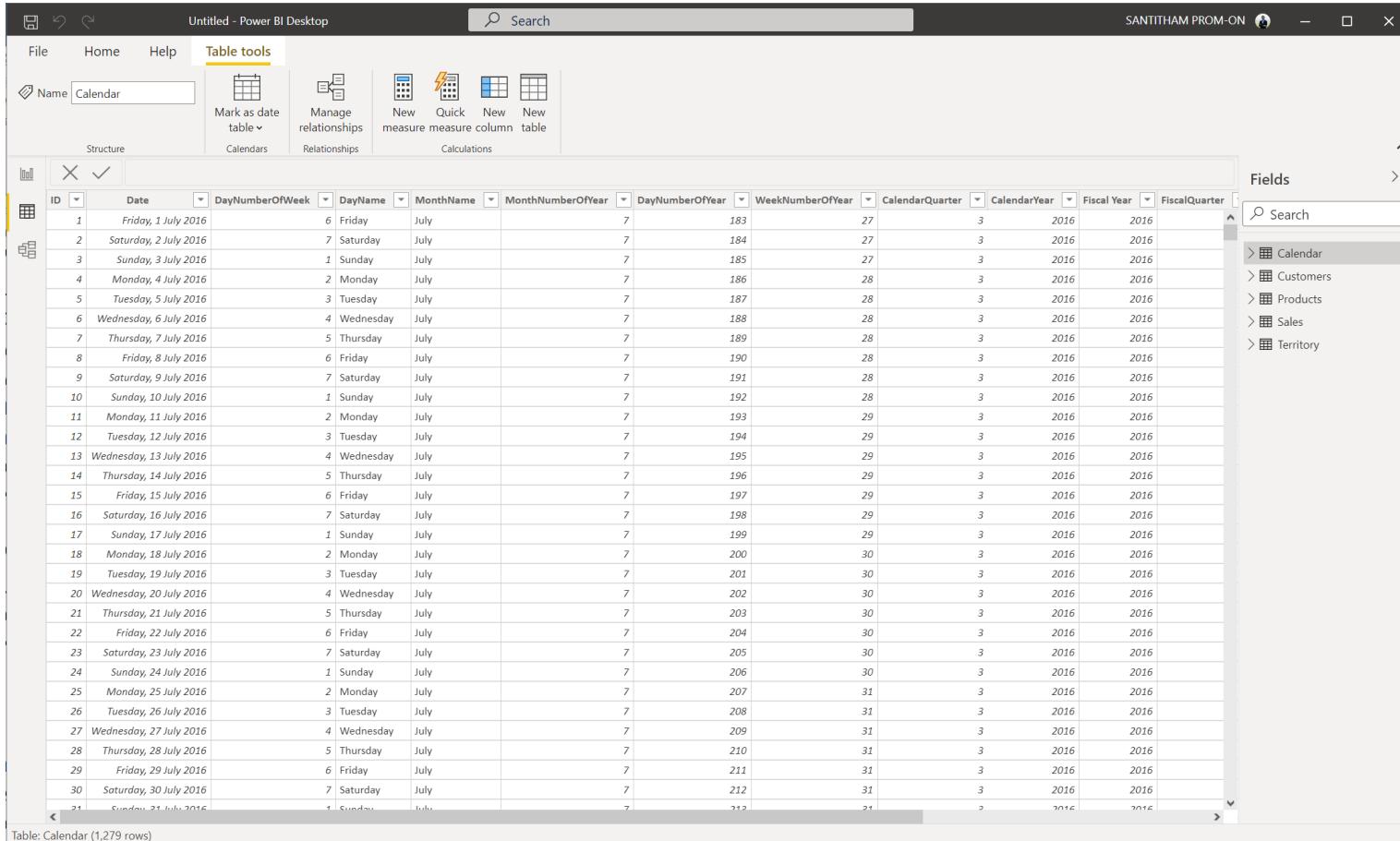
This View is used for creating visualization.
Data and Model have to be defined first.



Data View

Data View

This View shows the detail data for each table.
It also has tools to process the data.



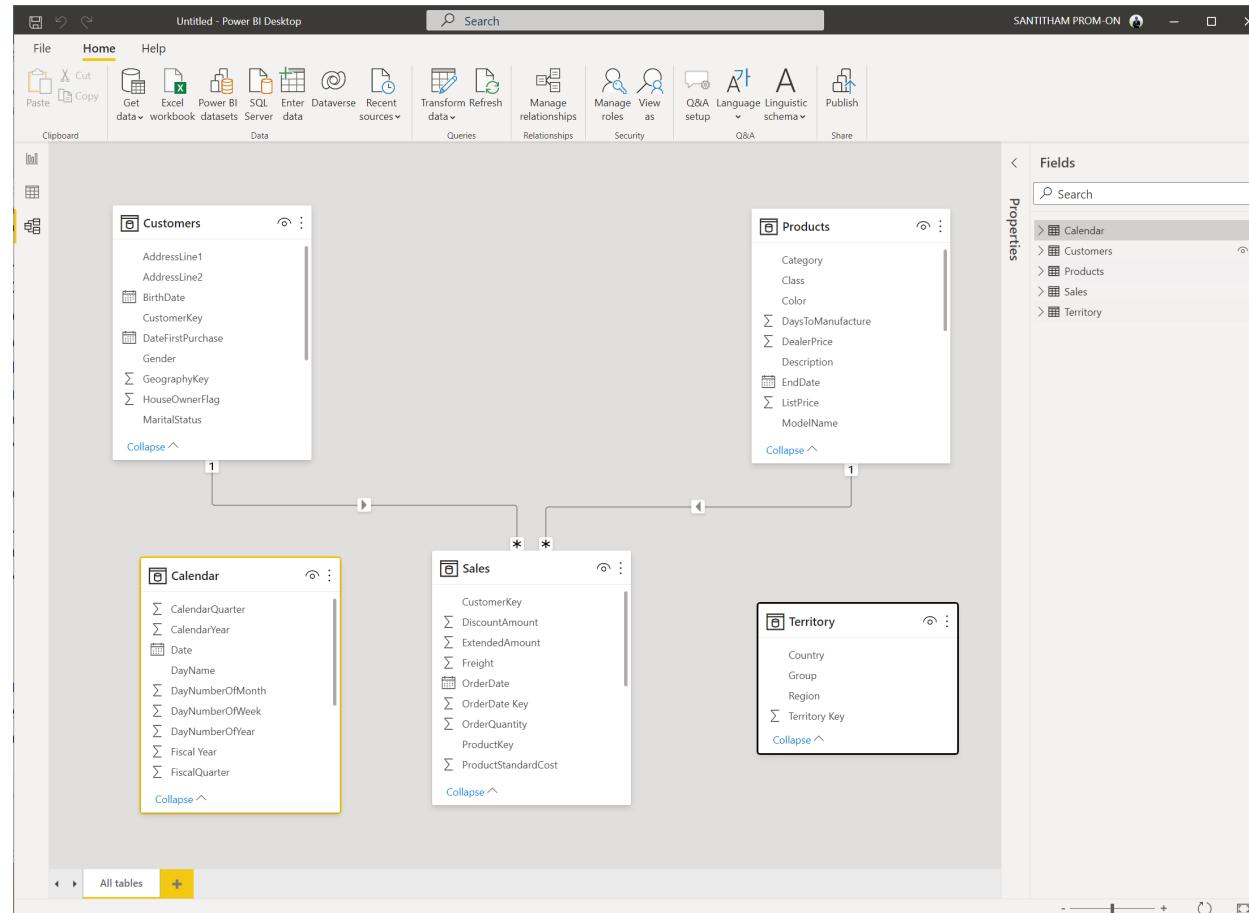
The screenshot shows the Power BI Desktop interface with the title bar "Untitled - Power BI Desktop". The ribbon menu is open at "Table tools". The "Structure" tab is selected, showing a table named "Calendar". The table contains 1,279 rows of data with columns: ID, Date, DayNumberOfWeek, DayName, MonthName, MonthNumberOfYear, DayNumberOfYear, WeekNumberOfYear, CalendarQuarter, CalendarYear, Fiscal Year, and FiscalQuarter. The data spans from July 1, 2016, to July 31, 2016. A "Fields" pane on the right lists other tables: Calendar, Customers, Products, Sales, and Territory, with "Calendar" currently selected. A search bar is also present in the Fields pane.

ID	Date	DayNumberOfWeek	DayName	MonthName	MonthNumberOfYear	DayNumberOfYear	WeekNumberOfYear	CalendarQuarter	CalendarYear	Fiscal Year	FiscalQuarter
1	Friday, 1 July 2016	6	Friday	July		183	27	3	2016	2016	
2	Saturday, 2 July 2016	7	Saturday	July		184	27	3	2016	2016	
3	Sunday, 3 July 2016	1	Sunday	July		185	27	3	2016	2016	
4	Monday, 4 July 2016	2	Monday	July		186	28	3	2016	2016	
5	Tuesday, 5 July 2016	3	Tuesday	July		187	28	3	2016	2016	
6	Wednesday, 6 July 2016	4	Wednesday	July		188	28	3	2016	2016	
7	Thursday, 7 July 2016	5	Thursday	July		189	28	3	2016	2016	
8	Friday, 8 July 2016	6	Friday	July		190	28	3	2016	2016	
9	Saturday, 9 July 2016	7	Saturday	July		191	28	3	2016	2016	
10	Sunday, 10 July 2016	1	Sunday	July		192	28	3	2016	2016	
11	Monday, 11 July 2016	2	Monday	July		193	29	3	2016	2016	
12	Tuesday, 12 July 2016	3	Tuesday	July		194	29	3	2016	2016	
13	Wednesday, 13 July 2016	4	Wednesday	July		195	29	3	2016	2016	
14	Thursday, 14 July 2016	5	Thursday	July		196	29	3	2016	2016	
15	Friday, 15 July 2016	6	Friday	July		197	29	3	2016	2016	
16	Saturday, 16 July 2016	7	Saturday	July		198	29	3	2016	2016	
17	Sunday, 17 July 2016	1	Sunday	July		199	29	3	2016	2016	
18	Monday, 18 July 2016	2	Monday	July		200	30	3	2016	2016	
19	Tuesday, 19 July 2016	3	Tuesday	July		201	30	3	2016	2016	
20	Wednesday, 20 July 2016	4	Wednesday	July		202	30	3	2016	2016	
21	Thursday, 21 July 2016	5	Thursday	July		203	30	3	2016	2016	
22	Friday, 22 July 2016	6	Friday	July		204	30	3	2016	2016	
23	Saturday, 23 July 2016	7	Saturday	July		205	30	3	2016	2016	
24	Sunday, 24 July 2016	1	Sunday	July		206	30	3	2016	2016	
25	Monday, 25 July 2016	2	Monday	July		207	31	3	2016	2016	
26	Tuesday, 26 July 2016	3	Tuesday	July		208	31	3	2016	2016	
27	Wednesday, 27 July 2016	4	Wednesday	July		209	31	3	2016	2016	
28	Thursday, 28 July 2016	5	Thursday	July		210	31	3	2016	2016	
29	Friday, 29 July 2016	6	Friday	July		211	31	3	2016	2016	
30	Saturday, 30 July 2016	7	Saturday	July		212	31	3	2016	2016	
31	Sunday, 31 July 2016	1	Sunday	July		213	31	3	2016	2016	

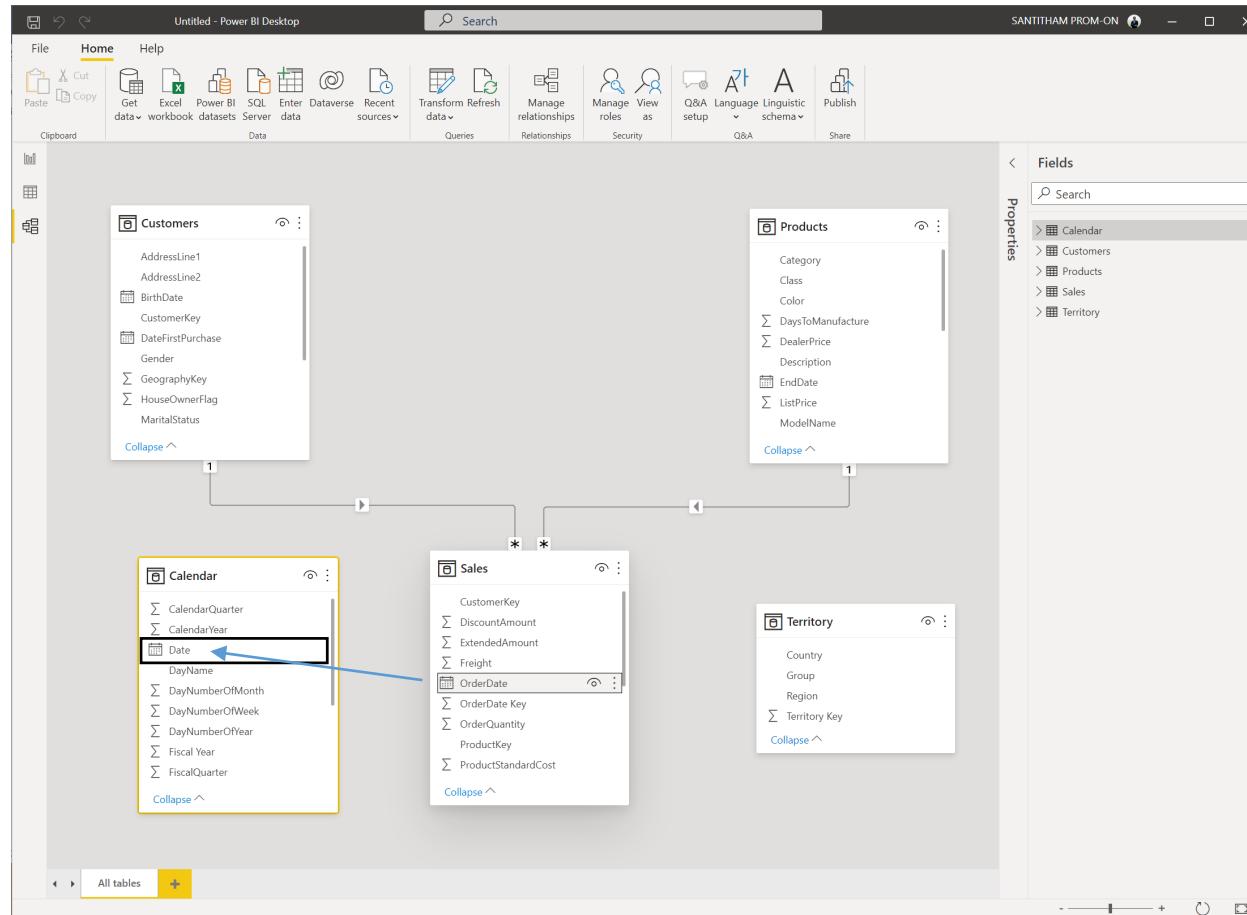
Model View

Model View

This view shows the relationships between each tables



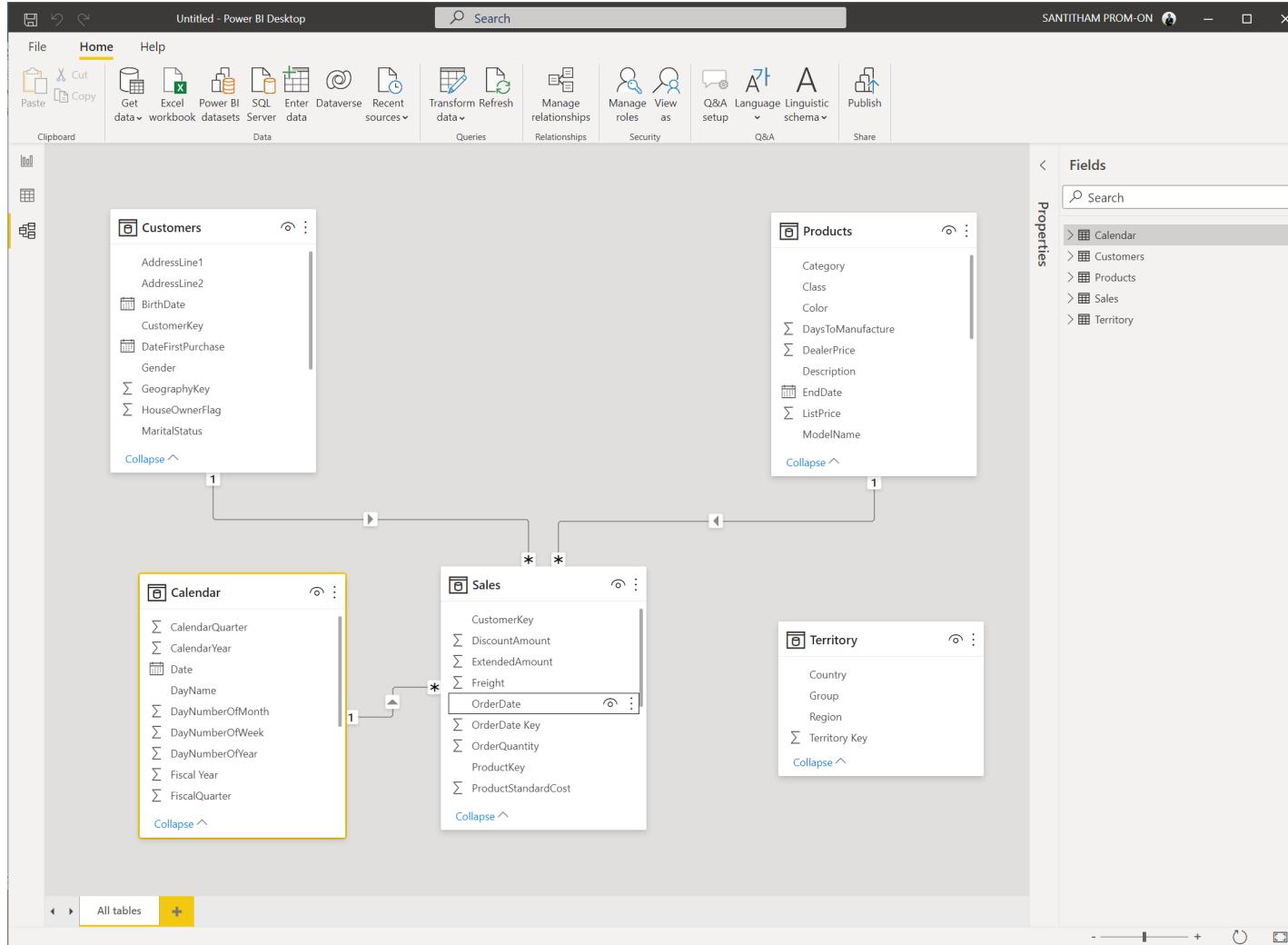
Create relationship





Relationships

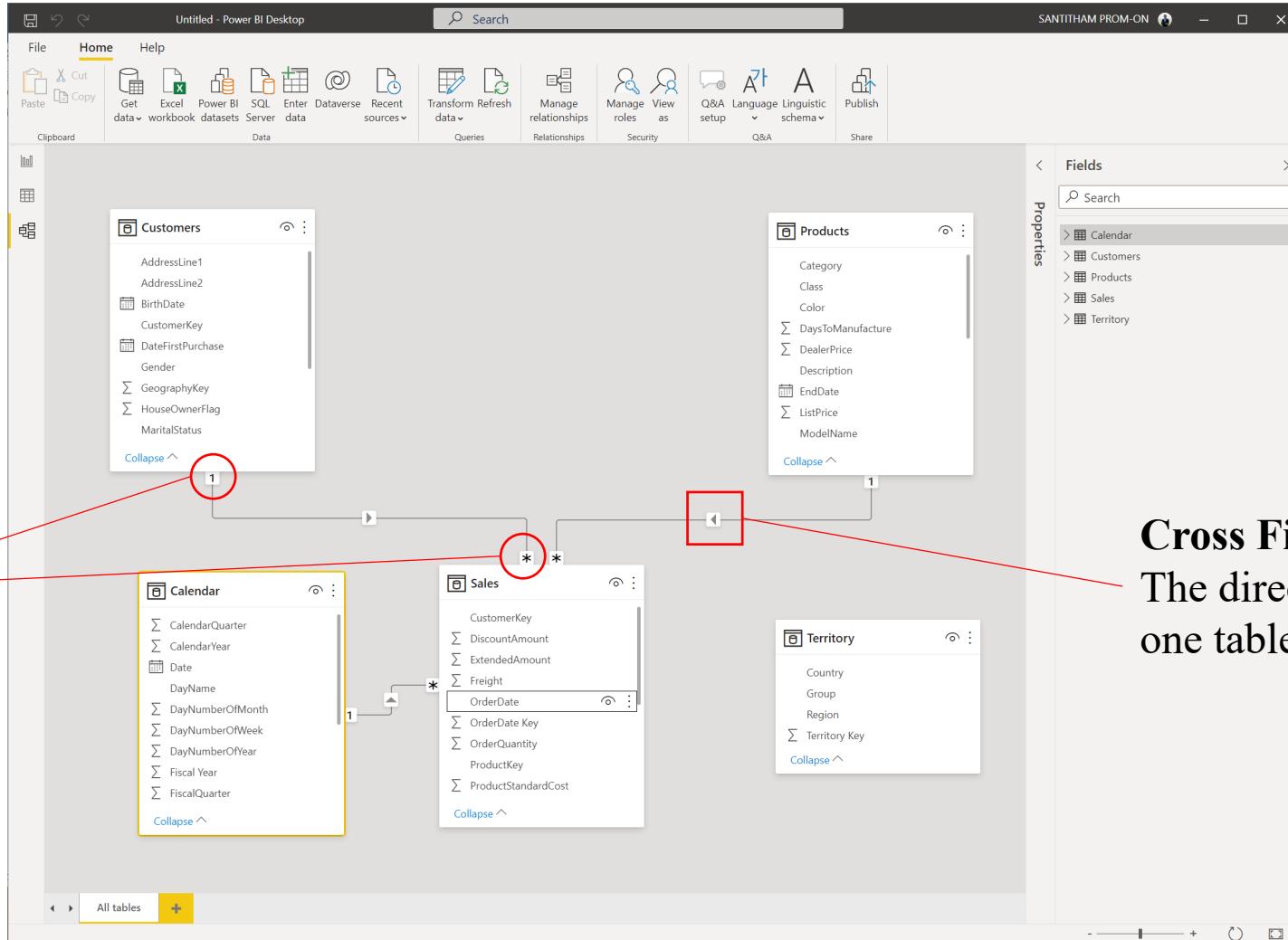
(More explanation in Module 2)





Relationships

(More explanation in Module 2)



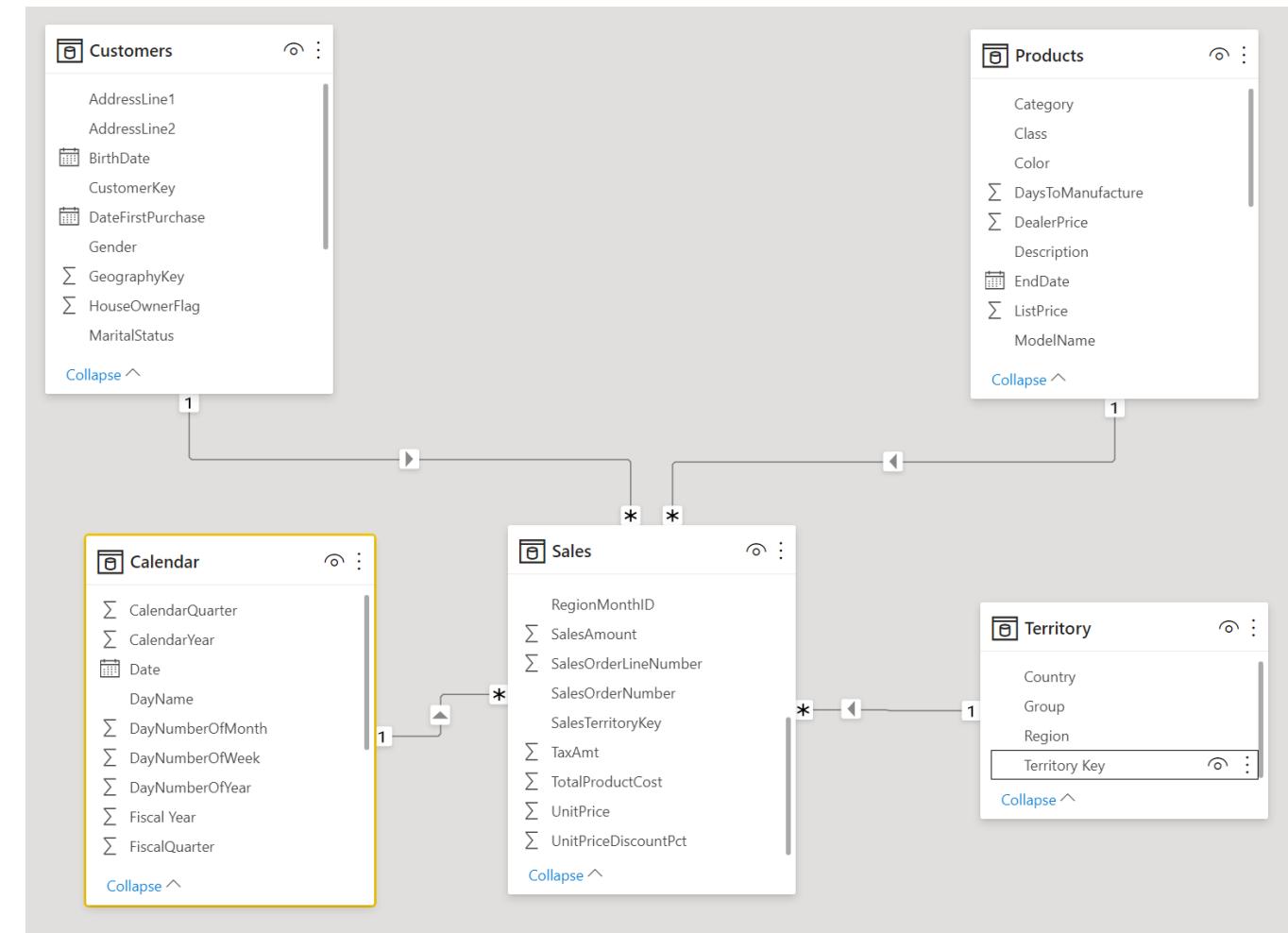
Cardinality
Types of relation

Cross Filter Direction
The direction that the data from one table filter another table.

Activity – Adding relationship

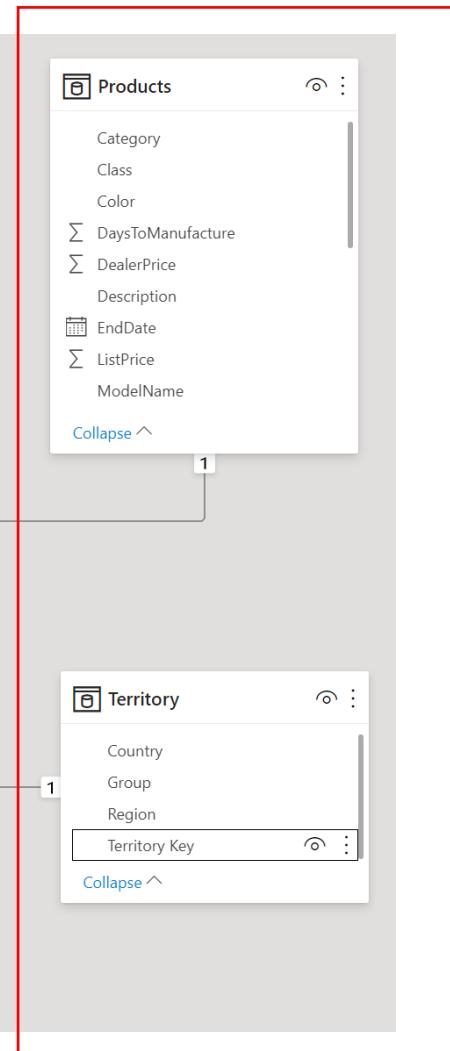
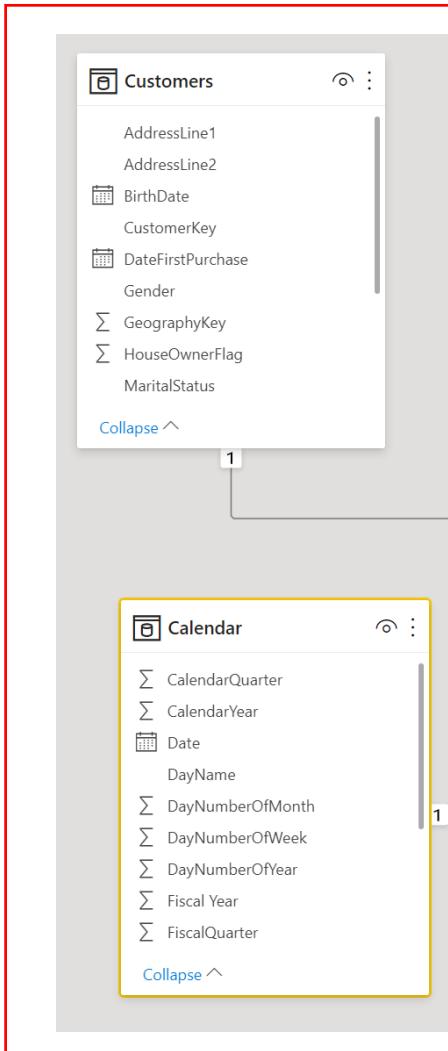
Add relationship
between “Territories”
and “Sales” using

- “Territory Key” in Table “Territories”
- “SalesTerritoryKey” in Sales



Type of Tables

DIM table
Master table
Lookup table



FACT table
Transaction table
Data table

DIM table
Master table
Lookup table



M language and DAX (Data Analysis eXpression)

M

★ DAX

Description	Where to Use
Power Query Formula Language	Data Preparation
Data Transformation	Before Data Model
Data Analysis Expression Language	Create Insights
Analytical Data Calculations	In Data Model
Comparable to Excel Function	



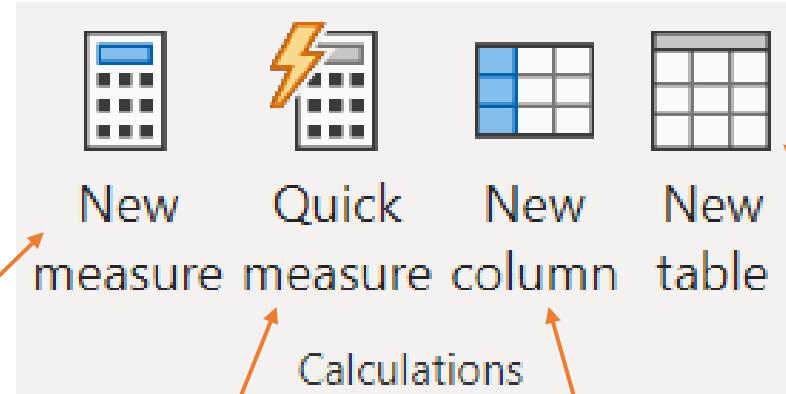
Best visual for developing DAX Matrix

Report View

A screenshot of the Power BI Desktop application. The ribbon is visible at the top with tabs like File, Home, Insert, etc. The main area shows a matrix visual with several columns and rows. On the right side, there are three panes: "Filters", "Visualizations", and "Fields". The "Visualizations" pane is open, showing various visualization icons, with the matrix icon highlighted by a red box and a red arrow pointing to it from the text "Matrix Icon". The "Fields" pane lists categories like Calendar, Customers, Products, Sales, and Territory. The bottom of the screen shows navigation controls for pages and a status bar indicating "Page 1 of 1".

Matrix Icon

Types of DAX based on output



Return a value

Build measure
using Wizard

Return a column

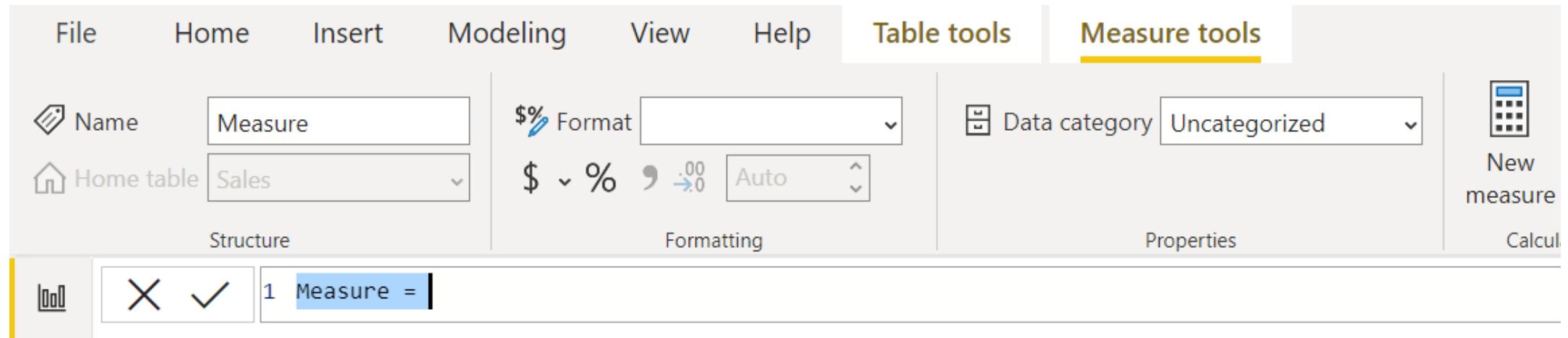
Return a table

Measure

- Measures have been around for many years in the enterprise versions of Microsoft BI tools, such as SQL Server Analysis Services.
- Measures have now made it into the world of business users who want to create Power BI reports.
- There is nothing confusing or hard to learn about measures.
- A measure is simply a DAX formula that instructs Power BI to do a calculation on data.

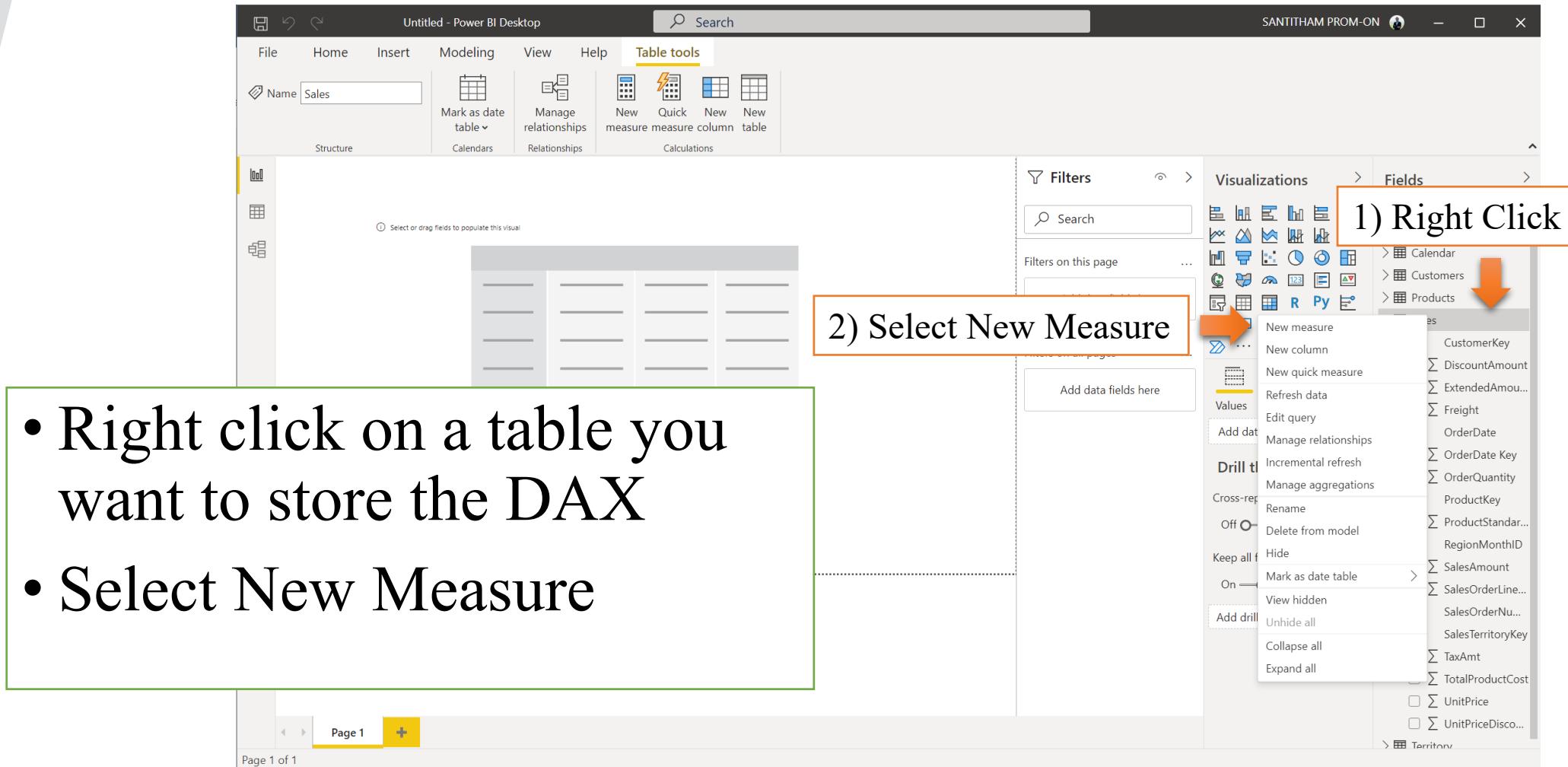
Technique for writing DAX measure

You can write measures in the formula bar



The formula bar is not visible unless you have a measure selected in the Fields pane on the right-hand side of Power BI.

Best practice in creating DAX



The screenshot shows the Power BI Desktop interface with the 'Table tools' ribbon selected. A context menu is open over a table named 'Sales'. The menu is divided into sections: Values, Drill to, Cross-report, Off, On, and Keep all fields. The 'Values' section is highlighted with a green box and contains the steps 1) Right Click and 2) Select New Measure. The 'On' section is highlighted with an orange box and contains the steps 1) Right Click and 2) Select New Measure.

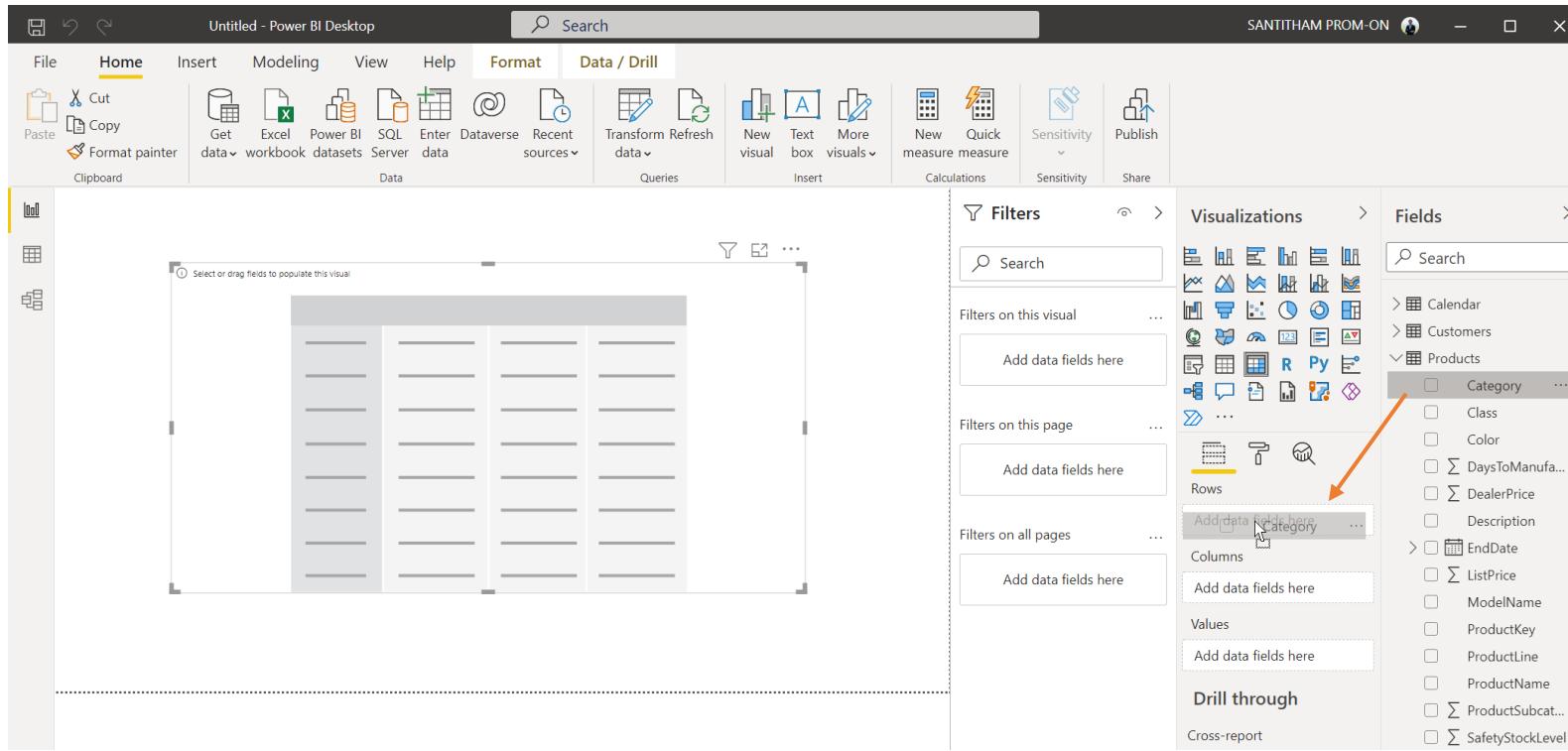
- Right click on a table you want to store the DAX
- Select New Measure

1) Right Click

2) Select New Measure

Let's write our first DAX

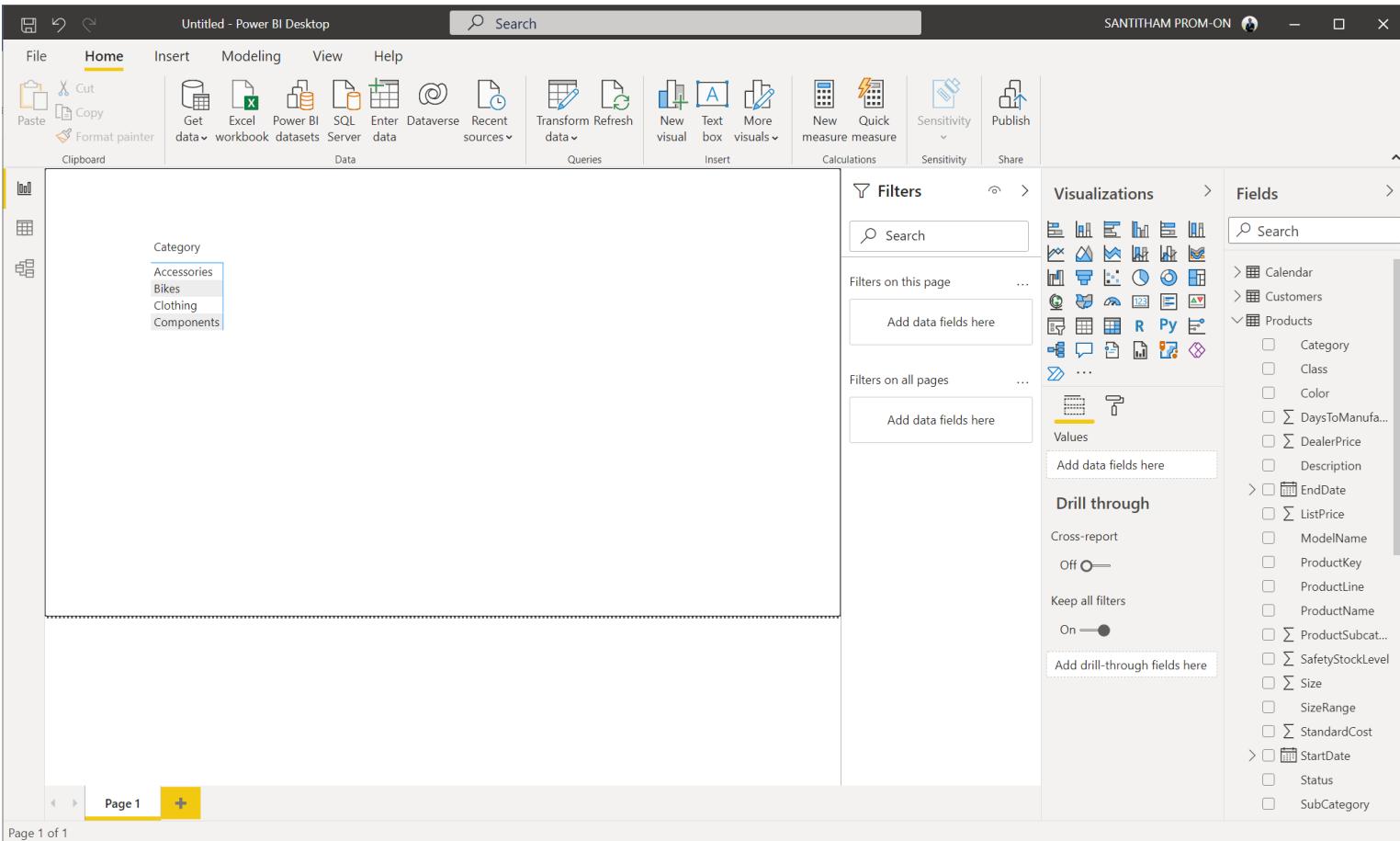
Build a matrix with Product[Category] at Rows



The screenshot shows the Power BI Desktop interface with the following details:

- Home tab selected:** The ribbon bar shows the Home tab is active.
- Visual area:** A matrix visual is being created, indicated by the placeholder text "Select or drag fields to populate this visual".
- Filters pane:** The left sidebar displays the "Filters" pane, which includes sections for "Filters on this visual", "Filters on this page", and "Filters on all pages".
- Visualizations pane:** The middle sidebar displays the "Visualizations" pane, showing various visualization icons.
- Fields pane:** The right sidebar displays the "Fields" pane, listing fields from the "Products" table:
 - Category (selected)
 - Class
 - Color
 - Σ DaysToManufa...
 - Σ DealerPrice
 - Description
 - EndDate
 - Σ ListPrice
 - modelName
 - ProductKey
 - ProductLine
 - ProductName
 - Σ ProductSubcat...
 - Σ SafetyStockLevel
- Drag-and-Drop:** An orange arrow points from the "Category" field in the Fields pane to the "Rows" section in the Filters pane, indicating the action of dragging the field to the visual.

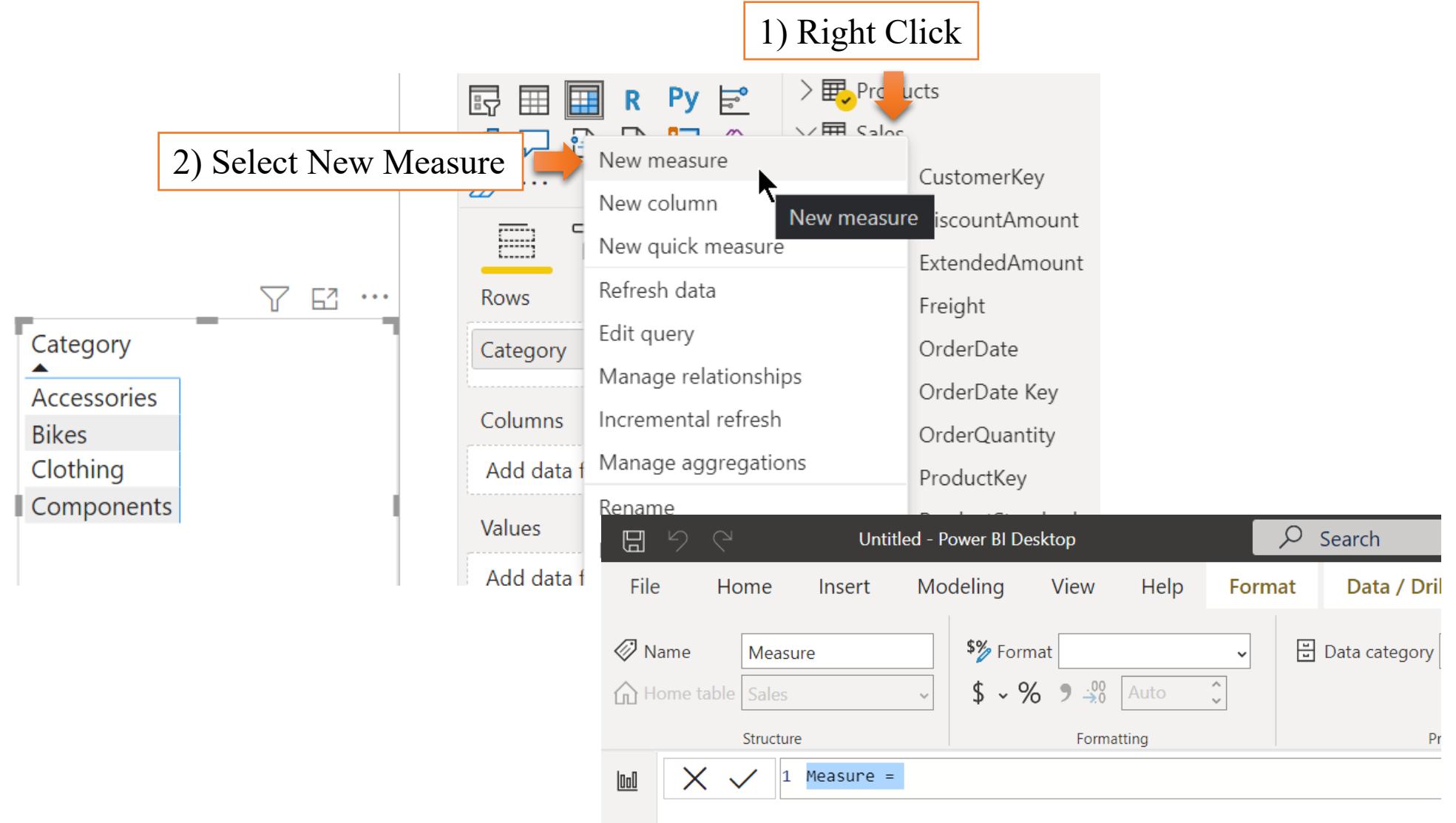
Matrix



The screenshot shows the Power BI Desktop interface with the following details:

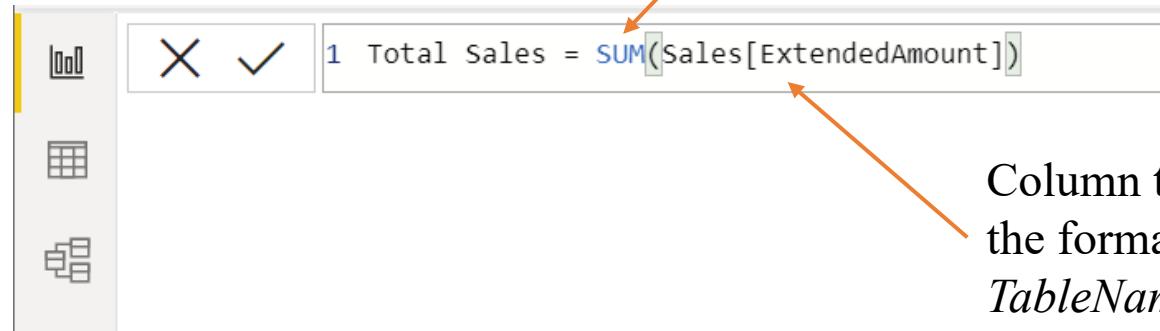
- Home Tab:** Selected.
- Clipboard:** Contains a list of categories: Accessories, Bikes, Clothing, Components.
- Data Tab:** Contains options for Get data, Excel workbook, Power BI datasets, SQL Server, Enter data, Dataverse, and Recent sources.
- Queries Tab:** Contains options for Transform data, Refresh data, New visual, Text box, More visuals, Insert, Calculations, Sensitivity, and Share.
- Visualizations Tab:** Contains icons for various visual types like bar charts, line charts, maps, etc.
- Fields Tab:** Shows a search bar and a list of fields under Products category:
 - Category
 - Class
 - Color
 - \sum DaysToManufacture
 - \sum DealerPrice
 - Description
 - EndDate
 - \sum ListPrice
 - modelName
 - ProductKey
 - ProductLine
 - ProductName
 - \sum ProductSubcategory
 - \sum SafetyStockLevel
 - \sum Size
 - SizeRange
 - \sum StandardCost
 - StartDate
 - Status
 - SubCategory
- Filters Panel:** Shows sections for Filters on this page and Filters on all pages, both with "Add data fields here" buttons.
- Drill through Panel:** Shows Cross-report set to Off, Drill-through set to Off, and Keep all filters set to On.
- Page Navigation:** Shows Page 1 of 1.

Add New measure



Add Total Sales

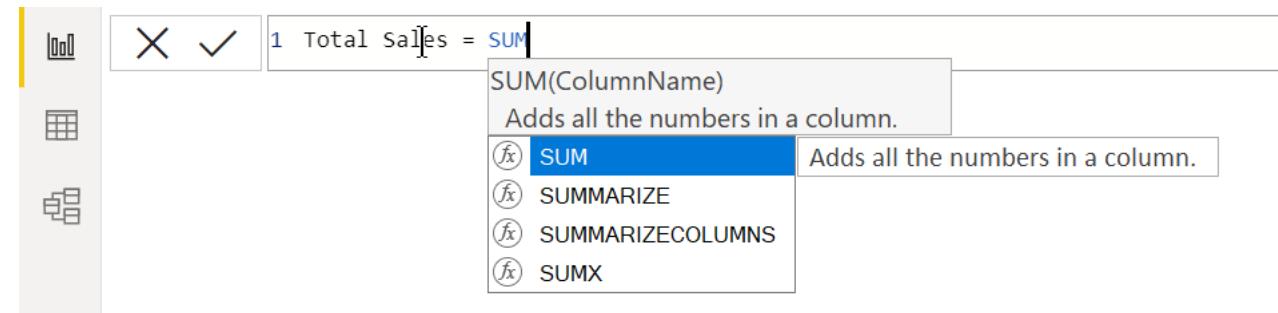
Measure Name



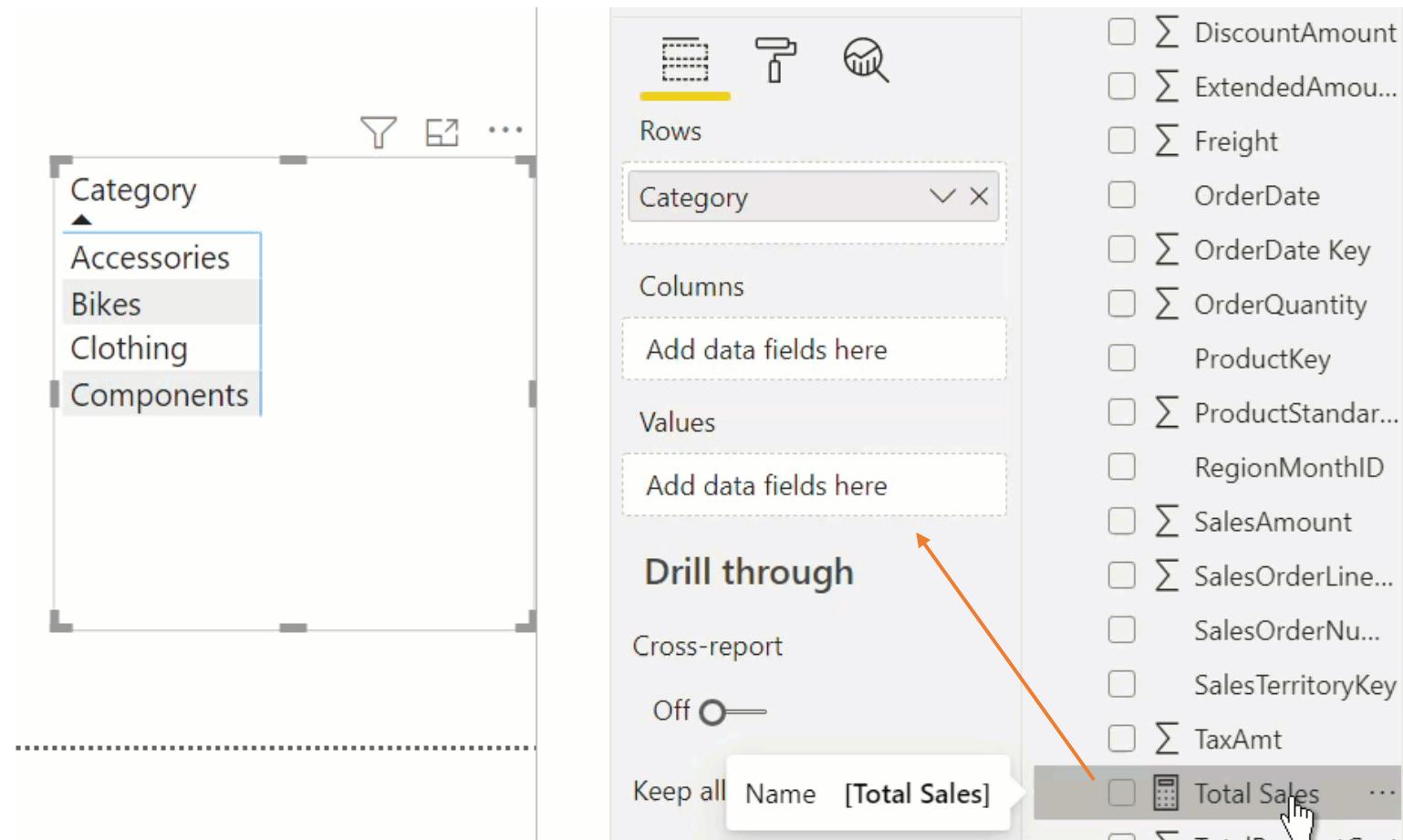
Aggregate Function

Column to be aggregated in
the format
TableName[ColumnName]

IntelliSense



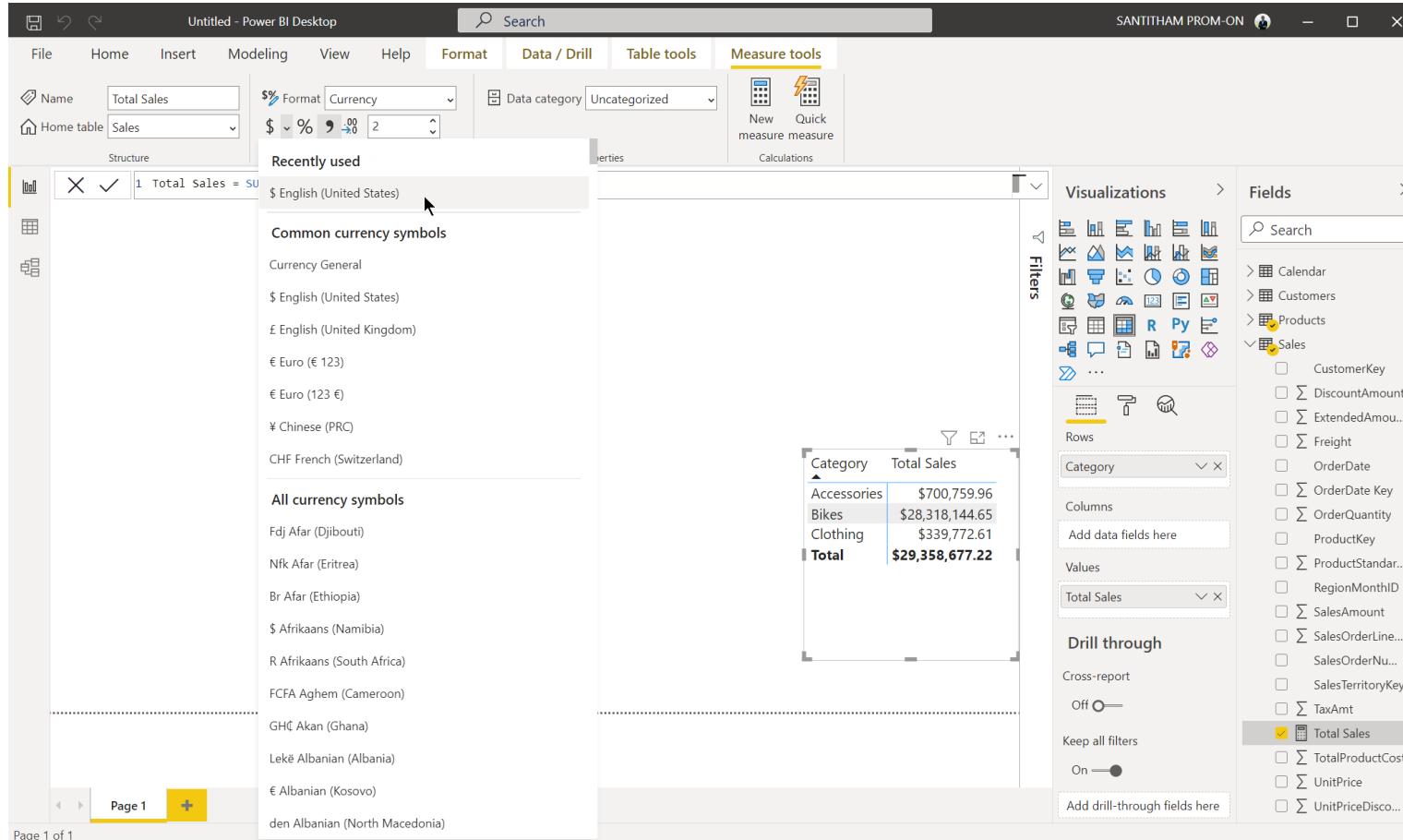
Test your measure with Matrix



The screenshot shows the Power BI Matrix visual interface. On the left, a vertical pane displays a hierarchy under 'Category': Accessories, Bikes, Clothing, and Components. The 'Bikes' item is selected. To the right of this is the main matrix area, which includes a 'Rows' section with 'Category' and a 'Values' section with 'Add data fields here'. Below the matrix are sections for 'Drill through', 'Cross-report', and a toggle switch labeled 'Off'. At the bottom, there is a context menu with options: 'Keep all', 'Name', '[Total Sales]', and a highlighted 'Total Sales' option with a small icon. A red arrow points from this highlighted option towards the list of available measures on the right.

- \sum DiscountAmount
- \sum ExtendedAmou...
- \sum Freight
- OrderDate
- \sum OrderDate Key
- \sum OrderQuantity
- ProductKey
- \sum ProductStandar...
- RegionMonthID
- \sum SalesAmount
- \sum SalesOrderLine...
- SalesOrderNu...
- SalesTerritoryKey
- \sum TaxAmt
- Total Sales

Format your DAX



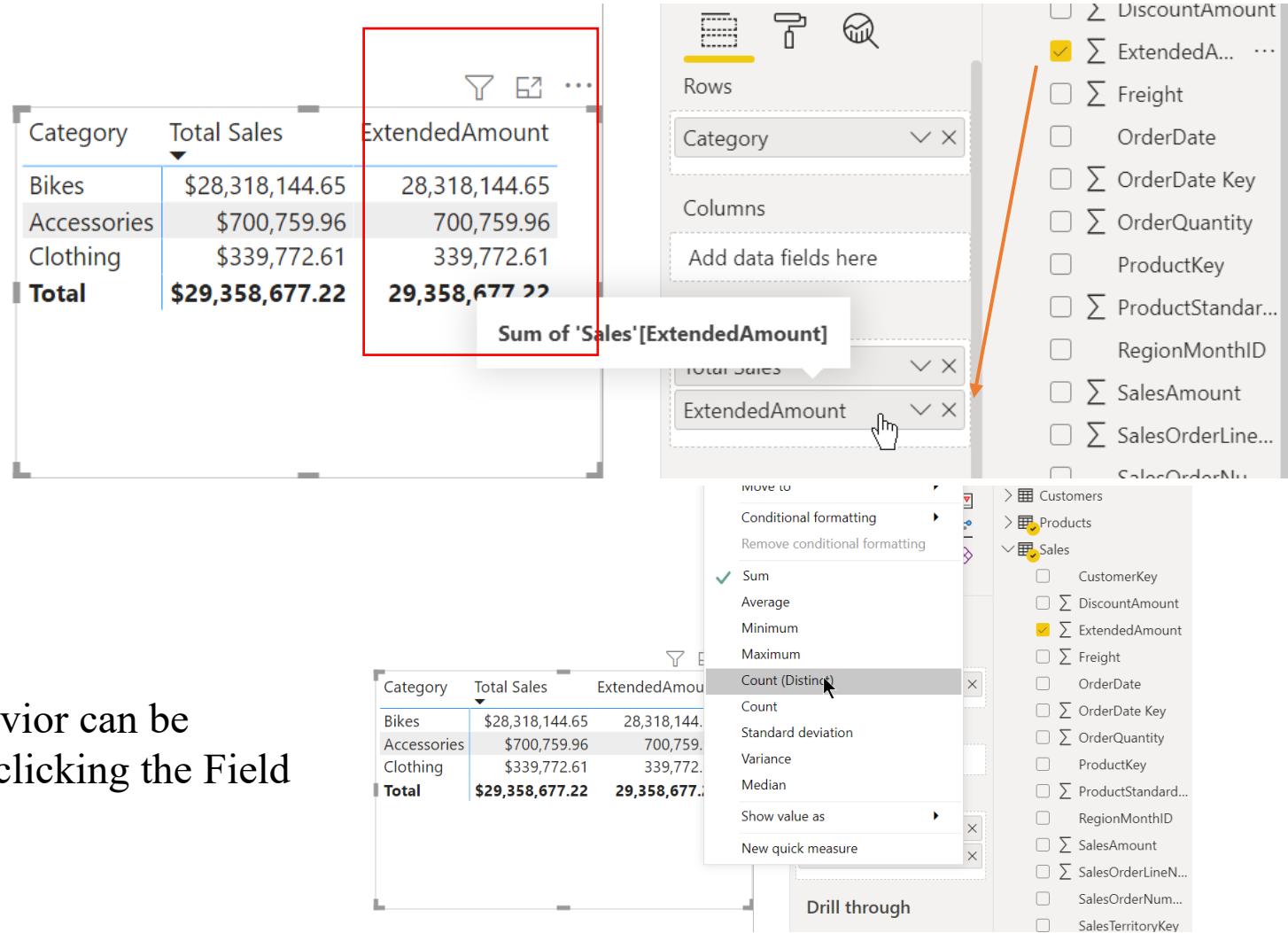
The screenshot shows the Power BI Desktop interface with the 'Format' tab selected in the ribbon. A dropdown menu is open for the measure 'Total Sales', displaying a list of currency symbols from various countries. The 'Currency' option is highlighted. In the main workspace, a table visualization is displayed with the following data:

Category	Total Sales
Accessories	\$700,759.96
Bikes	\$28,318,144.65
Clothing	\$339,772.61
Total	\$29,358,677.22

The 'Fields' pane on the right shows the 'Sales' table with various columns listed. The 'Total Sales' column has a checkmark next to it, indicating it is selected.

Avoid implicit measure

If you directly drag the Field to the Values, it will be automatically aggregated. (SUM as default)



The screenshot shows a Power BI report interface. On the left is a table with three columns: Category, Total Sales, and ExtendedAmount. The ExtendedAmount column has its first three rows highlighted with a red box. The 'Total' row in the ExtendedAmount column also has a red box around it. A tooltip 'Sum of 'Sales'[ExtendedAmount]' appears over the total value. To the right is a 'Rows' pane with 'Category' selected. Below it is a 'Columns' pane with 'Add data fields here'. A context menu is open over the 'ExtendedAmount' field in the Columns pane, with 'Count (Distinct)' highlighted. To the far right is a large list of available measures, with 'Σ ExtendedAmount' checked.

Category	Total Sales	ExtendedAmount
Bikes	\$28,318,144.65	28,318,144.65
Accessories	\$700,759.96	700,759.96
Clothing	\$339,772.61	339,772.61
Total	\$29,358,677.22	29,358,677.22

Sum of 'Sales'[ExtendedAmount]

Rows

Category

Columns

Add data fields here

ExtendedAmount

Move to

- Conditional formatting
- Remove conditional formatting
- Sum** (checked)
- Average
- Minimum
- Maximum
- Count (Distinct) **Selected**
- Count
- Standard deviation
- Variance
- Median
- Show value as
- New quick measure

Drill through

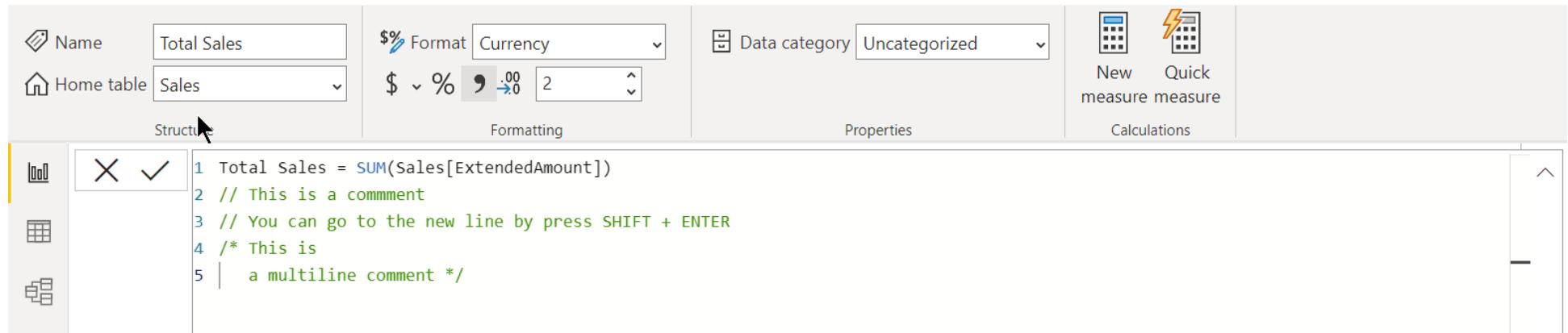
- Σ DiscountAmount
- Σ ExtendedA... ...
- Σ Freight
- OrderDate
- Σ OrderDate Key
- Σ OrderQuantity
- ProductKey
- Σ ProductStandar...
- RegionMonthID
- Σ SalesAmount
- Σ SalesOrderLine...
- SalesOrderN...
- Customers
- Products
- Sales**
- CustomerKey
- Σ DiscountAmount
- Σ ExtendedAmount
- Σ Freight
- OrderDate
- Σ OrderDate Key
- Σ OrderQuantity
- ProductKey
- Σ ProductStandard...
- RegionMonthID
- Σ SalesAmount
- Σ SalesOrderLine...
- SalesOrderNum...
- SalesTerritoryKey

Aggregation behavior can be changed by right clicking the Field

Why implicit measure should be avoided?

- Name is not helpful. Need additional steps in renaming.
- No formatting is applied. While this can be done in the source field, you cannot have different formatting for different visuals.
- You cannot reference implicit measures in other measures.
- You will not learn how to write DAX

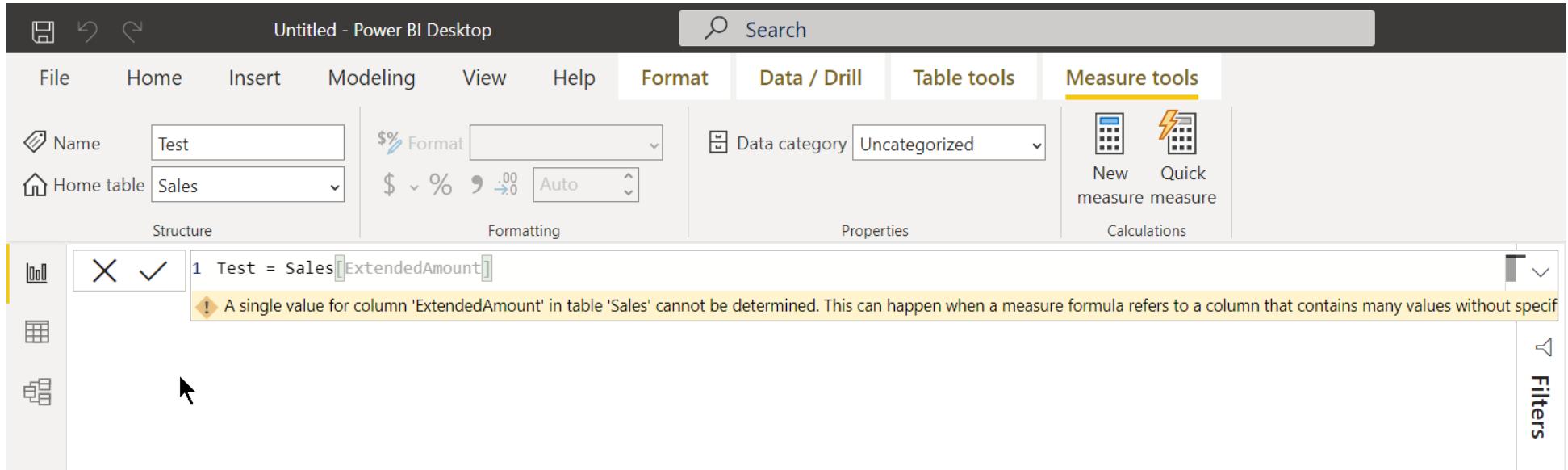
Comments



To write go to the new line
SHIFT + ENTER

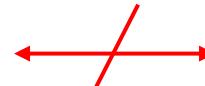
To add comment
Use either // or /* */

Measure with Naked columns do not work



Expect a single value

Return a column



DAX, Aggregation: SUM Practice

- [Total Sales], you can use the ExtendedAmount in the Sales table
- [Total Cost], you can use any product cost in Sales table
- [Total Tax Paid], you can use the tax column in Sales table
- [Total Order Quantity], you can use the quantities in Sales table

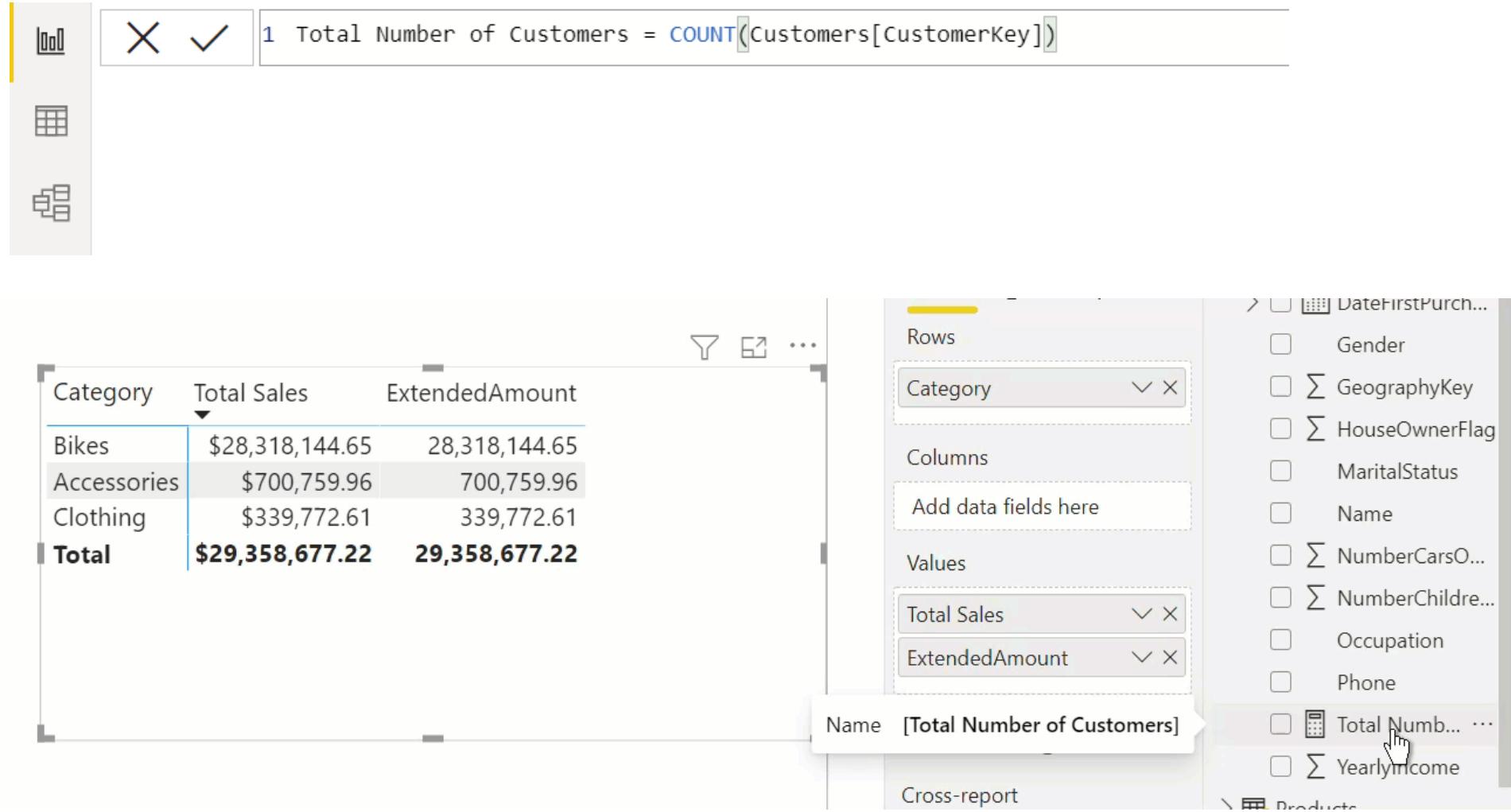
Check Point – Modeler Habits

- Did you create a matrix and put Product[Category] on Rows in your matrix?
- Did you right-click the Sales table in the Fields pane and select New Measure to start the process?
- Did you reference all columns in your measures in the format *TableName[ColumnName]*
- Did you apply formatting to your measure immediately after you write it?
- Did you use the keyboard and look at the IntelliSense and you typed the measures?

DAX, Aggregation: COUNT() Practice

- [Total Number of Products], use the product table to count how many products there are
- [Total Number of Customers], use the customer name instead of customer key

Total Number of Customers



The screenshot shows a data visualization interface with a table and a data model pane.

Table:

Category	Total Sales	ExtendedAmount
Bikes	\$28,318,144.65	28,318,144.65
Accessories	\$700,759.96	700,759.96
Clothing	\$339,772.61	339,772.61
Total	\$29,358,677.22	29,358,677.22

Data Model Pane:

- Rows:** Category
- Columns:** Add data fields here
- Values:** Total Sales, ExtendedAmount
- Name:** [Total Number of Customers]
- Cross-report:** DateFirstPurch...
- Available Fields:** Gender, GeographyKey, HouseOwnerFlag, MaritalStatus, Name, NumberCarsO..., NumberChildre..., Occupation, Phone, Total Numb..., YearlyIncome

Why?

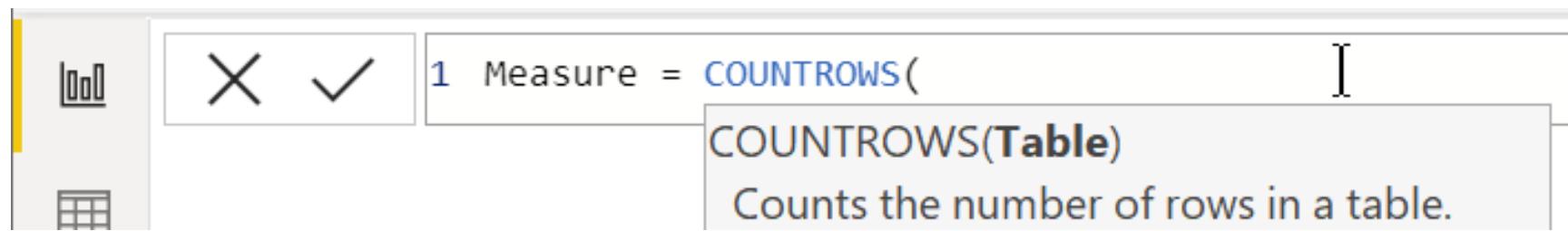
Category	Number of Customers	Total Sales	ExtendedAmount
Bikes	18484	\$28,318,144.65	28,318,144.65
Accessories	18484	\$700,759.96	700,759.96
Clothing	18484	\$339,772.61	339,772.61
Components	18484		
Total	18484	\$29,358,677.22	29,358,677.22

Total number
of customers

Total sales of
each Category

DAX, Aggregation: COUNTROWS()

- COUNTROWS[Table] counts the number of rows in a table

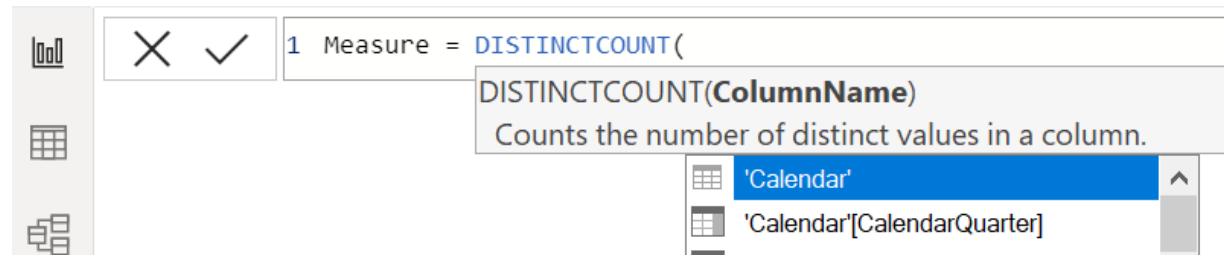


Practice

- [Total Number of Products COUNTROW]
- [Total Number of Customers COUNTROW]

DAX, Aggregation: DISTINCTCOUNT()

- Count the number of distinct values in a column.



Practice

- Use occupation instead of category
- [Total Customers DISTINCTCOUNT], should we use customer key or name?
- [Count of Country], use Territories table and count the number of countries in each Group
- [Total Customers That Have Purchased], use subcategories as Row Header, count the distinct customers purchased products in each subcategories

DAX, Aggregation: MAX(), MIN(), AVERAGE()

Use Product[Category] as Row header

- [Maximum Tax Paid on a Product] use a suitable column from the Sales table along with MAX() function
- [Minimum Price Paid for a Product] use a suitable column from the Sales table along with MIN() function
- [Average Price Paid for a Product] use a suitable column from the Sales table along with AVERAGE()

Expected output

Category	Maximum Tax Paid on a Product	Minimum Price Paid for a Product	Average Price Paid for a Product
Accessories	\$12.72	\$2.29	\$19.42
Bikes	\$286.26	\$539.99	\$1,862.42
Clothing	\$5.60	\$8.99	\$37.33
Total	\$286.26	\$2.29	\$486.09

DAX, Aggregation: COUNTBLANK()

- Count the number of blanks in a column
- Format

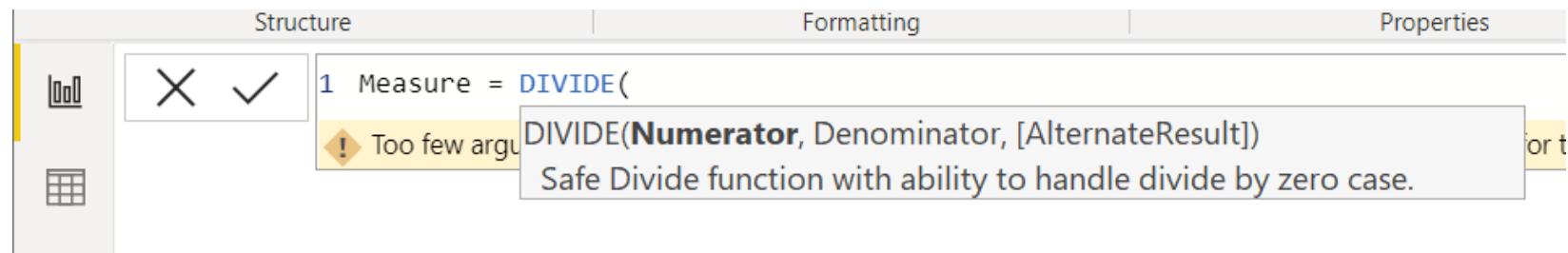
COUNTBLANK(Column Name)

Practice

- [Total Customers Without Address Line 2]
- [Total Products Without Weight Values]

DAX, Operator: DIVIDE()

- Safe divide



Practice

- [Margin %], [Total Margin \$] divided by [Total Sales]
 - Total Margin \$ is the profit
- [Tax %], [Total Tax] divided by [Total Sales]

Query Evaluation

- There are 2 distinct engines that work together to process every DAX query: Formula Engine and Storage Engine

Formula Engine

Receives, interpret and executes all DAX request

Process DAX query and generate **query plan**

Work with **datacache** to evaluate DAX query and return it

Storage Engine

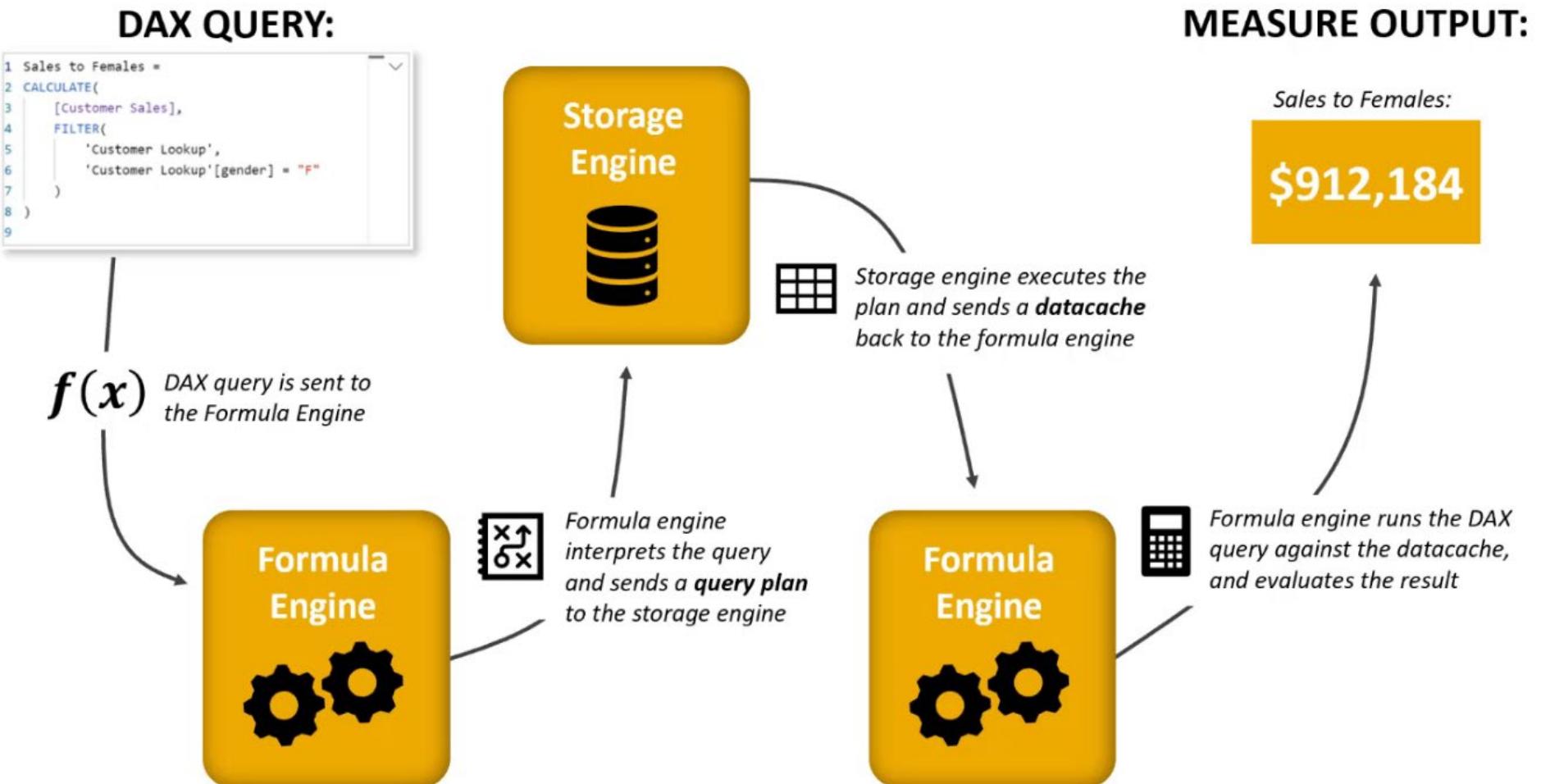
Compresses and encodes raw data and only communicate with Formula engine

Receive a query plan from Formula Engine, executes it and return **datacache**

There are two types of Storage Engine

- **VertiPaq** is used for in-memory data (import mode)
- **DirectQuery** is used for data read directly from the source (e.g. MSSQL)

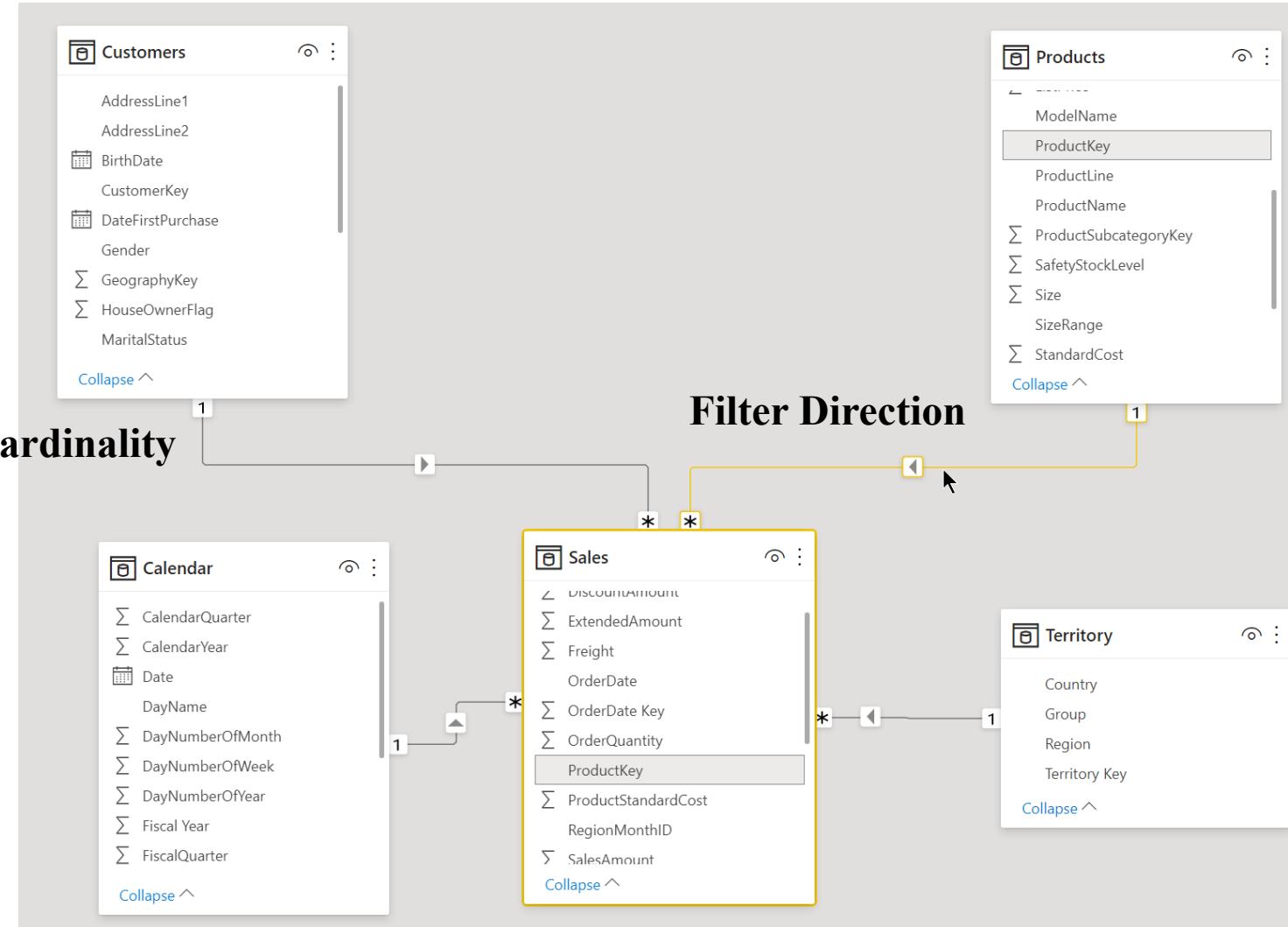
DAX Query Evaluation



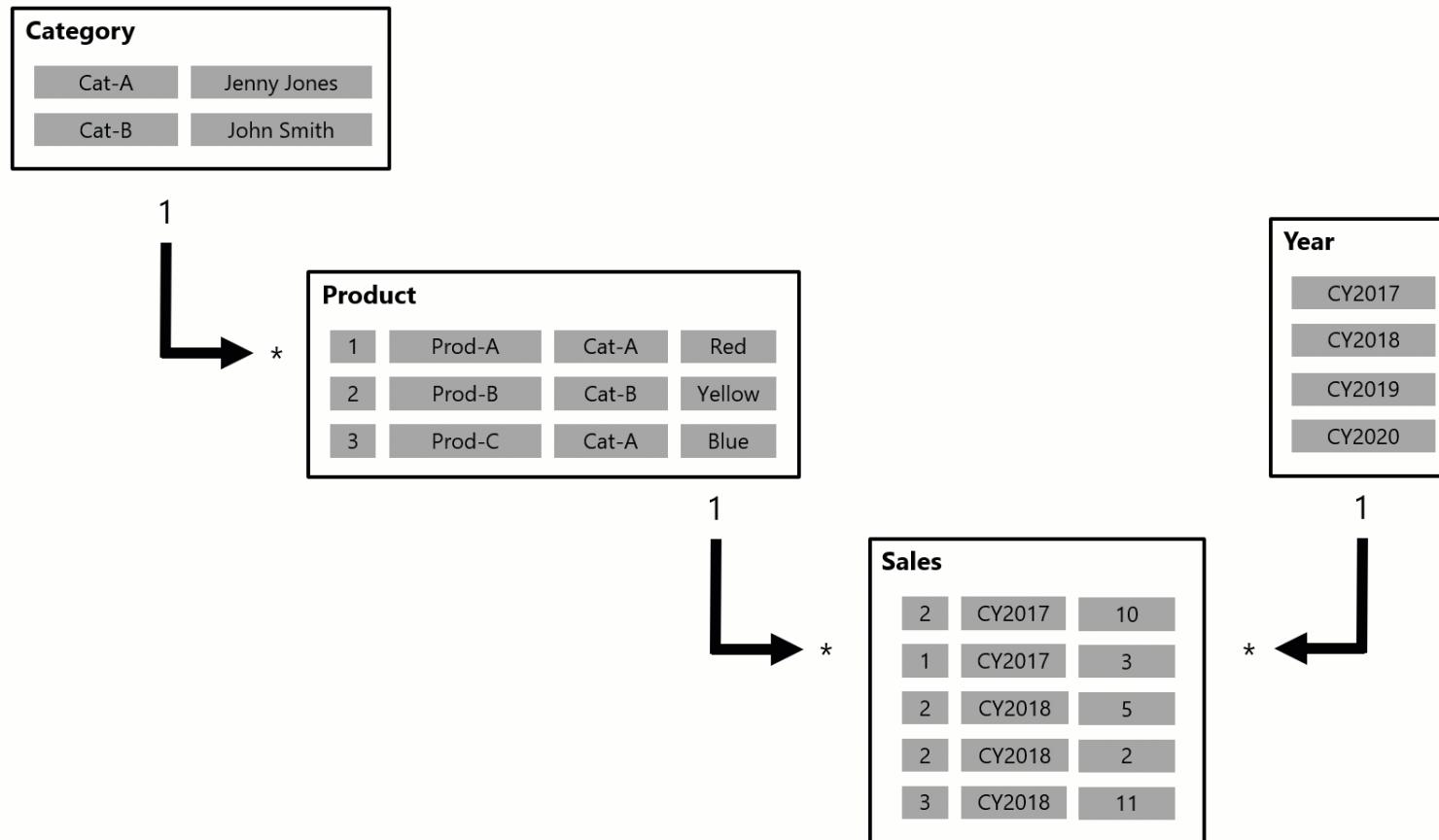
Part 2

- Relationship and cardinality
- Filter propagation
- DAX: Calculated columns
- DAX: RELATED()
- DAX: Iterator
- DAX: Calculate

Model View



Filter propagation



Cross Filter Direction

- One-to-many (1:*)
- Many-to-one (*:1)
- One-to-one (1:1)
- Many-to-many (*:*)

Cardinality type	Cross filter options
One-to-many (or Many-to-one)	Single Both
One-to-one	Both
Many-to-many	Single (Table1 to Table2) Single (Table2 to Table1) Both

- *Single* cross filter direction means "single direction"
- *Both* means "both directions". aka *bi-directional*.
- For One-to-many relationships, the cross filter direction is always from the "one" side, and optionally from the "many" side (bi-directional).
- For One-to-one relationships, the cross filter direction is always from both tables.
- For the Many-to-many relationships, cross filter direction can be from either one of the tables, or from both tables.

Initial Filter

Category	Total Number of Customers	Total Number of Products
Accessories	18,484	35
Bikes	18,484	125
Clothing	18,484	48
Components	18,484	189
Total	18,484	397

The matrix first “filters” the data model (in this case, it is the Products[Category] column), and then Power BI evaluates the measures for each row in the matrix with the filters applied

Let’s call it “Initial Filter”

Initial Filters

Rows (#1)

Slicers (#3)

Columns (#2)

Filters (#4)



The screenshot illustrates the application of four types of filters in a Power BI visualization:

- Rows (#1):** A slicer for 'Size' on the left, with options like (Blank), 38, 40, 42, 44, 46, 48, 50, 52, 54, and 56. The 'NA' option is selected, highlighted with a red circle containing the number 1.
- Columns (#2):** A table showing sales data by color and category. The columns are Color, Accessories, Bikes, Clothing, and Total. The 'Bikes' column is highlighted with a red circle containing the number 2.
- Slicers (#3):** The 'Size' slicer itself, which is a dropdown menu with the selected item 'NA' highlighted.
- Filters (#4):** The 'Filters' pane on the right, which contains four items:
 - Search bar: 'search'
 - Filters on this visual: 'Category is (All)', 'Color is (All)', and 'Total Sales is (All)'
 - Add data fields here
 - Filters on this page

The table data is as follows:

Color	Accessories	Bikes	Clothing	Total
Black	\$72,954	\$8,659,117	\$106,341	\$8,838,412
Blue	\$74,354	\$2,169,056	\$35,687	\$2,279,096
Multi			\$106,471	\$106,471
NA (1)	\$435,117			\$435,117
Red	\$78,028	\$7,646,303		\$7,724,331
Silver	\$40,308	\$5,073,081		\$5,113,389
White			\$5,106	\$5,106
Yellow		\$4,770,588	\$86,168	\$4,856,756
Total	\$700,760	\$28,318,145	\$339,773	\$29,358,677

Let's add [Total Number of Products] to table Product



1 Total Number of Product = `DISTINCTCOUNT(Products[ProductKey])`

Category	Total Number of Customers	Total Number of Product	Total Sales
Accessories	18484	35	\$700,759.96
Bikes	18484	125	\$28,318,144.65
Clothing	18484	48	\$339,772.61
Components	18484	189	
Total	18484	397	\$29,358,677.22

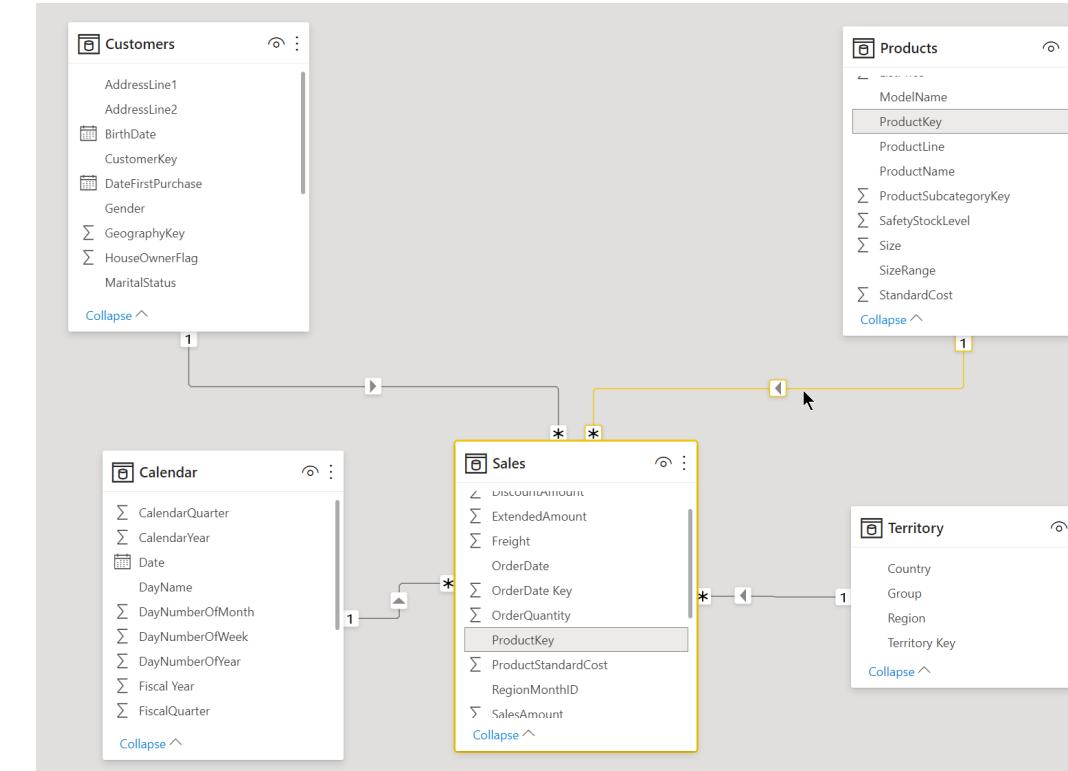
Filter Propagation

Initial Filter

Category	Total Number of Customers	Total Number of Product	Total Sales
Accessories	18484	35	\$700,759.96
Bikes	18484	125	\$28,318,144.65
Clothing	18484	48	\$339,772.61
Components	18484	189	
Total	18484	397	\$29,358,677.22

In the same table
as initial filer

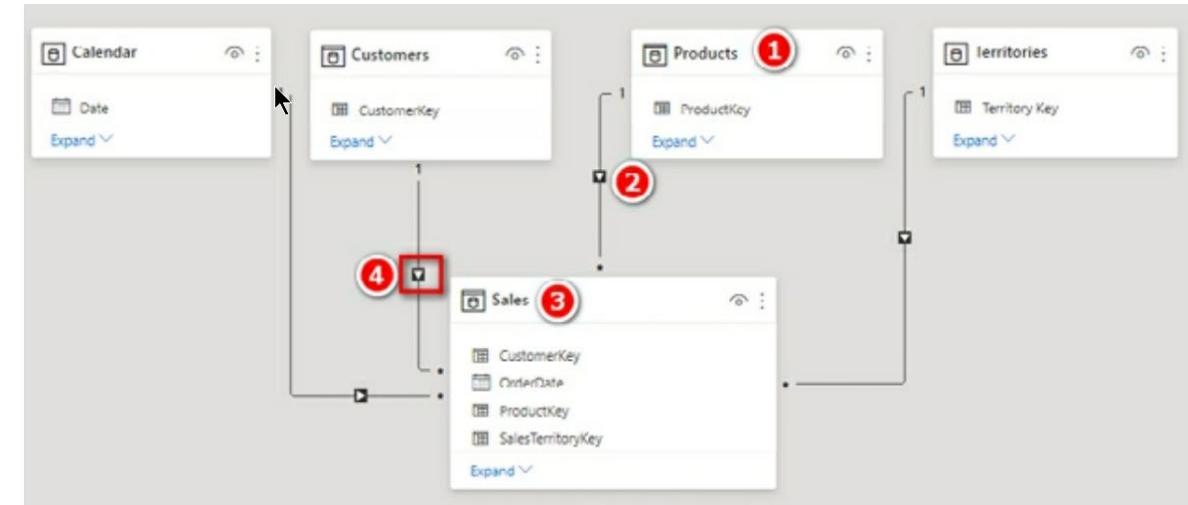
Not related
to initial
filter



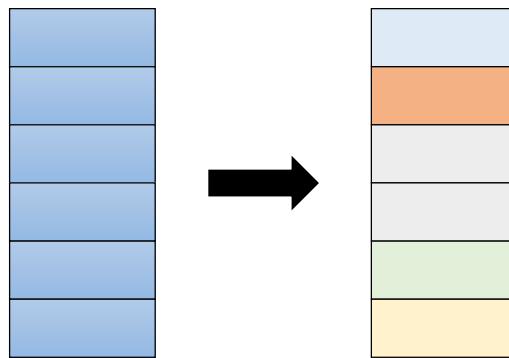
Cross-filter
direction from
Product to Sales

1. The initial filter is applied to the table(s). In this example, Products[Category] = "Clothing".
2. This filter automatically propagates through the relationships that exist between the tables, flowing in the direction of the arrow to the connected tables.
3. The connected table(s) (Sales in this example) is then also filtered so that the same products in the Products table will remain in the Sales table.
4. The filter applied to the Sales table does not automatically flow back uphill to the Customers table

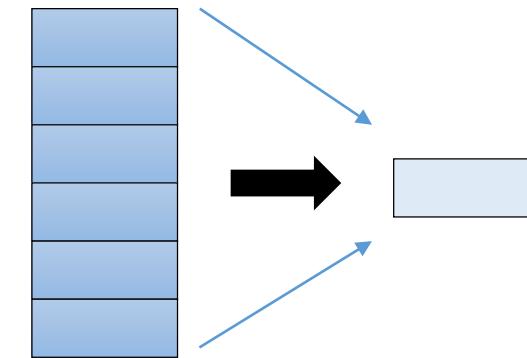
Category	Total Number of Customers	Total Number of Products
Accessories	18,484	35
Bikes	18,484	125
Clothing	18,484	48
Components	18,484	189
Total	18,484	397



New Column



New Measure



Calculated columns

- Row-by-row evaluation
- Add column to data model (consume more memory)

Basic (but wrong) calculated columns

The screenshot illustrates a common mistake in creating calculated columns in a data visualization tool like Power BI.

Top Table: A list of Sales Order Lines. The columns are: SalesOrderNumber, SalesOrderLineNumber, OrderQuantity, UnitPrice, ExtendedAmount (highlighted with a red box), TaxAmt, and Freight.

Middle Table: A calculated column named "Total Sales Plus Tax Column" is created by summing ExtendedAmount and TaxAmt. The resulting values are highlighted with a red border.

Bottom Table: A summary table grouped by Category. The "Total Sales Plus Tax Wrong" column shows incorrect totals for each category and a wrong total for the entire dataset.

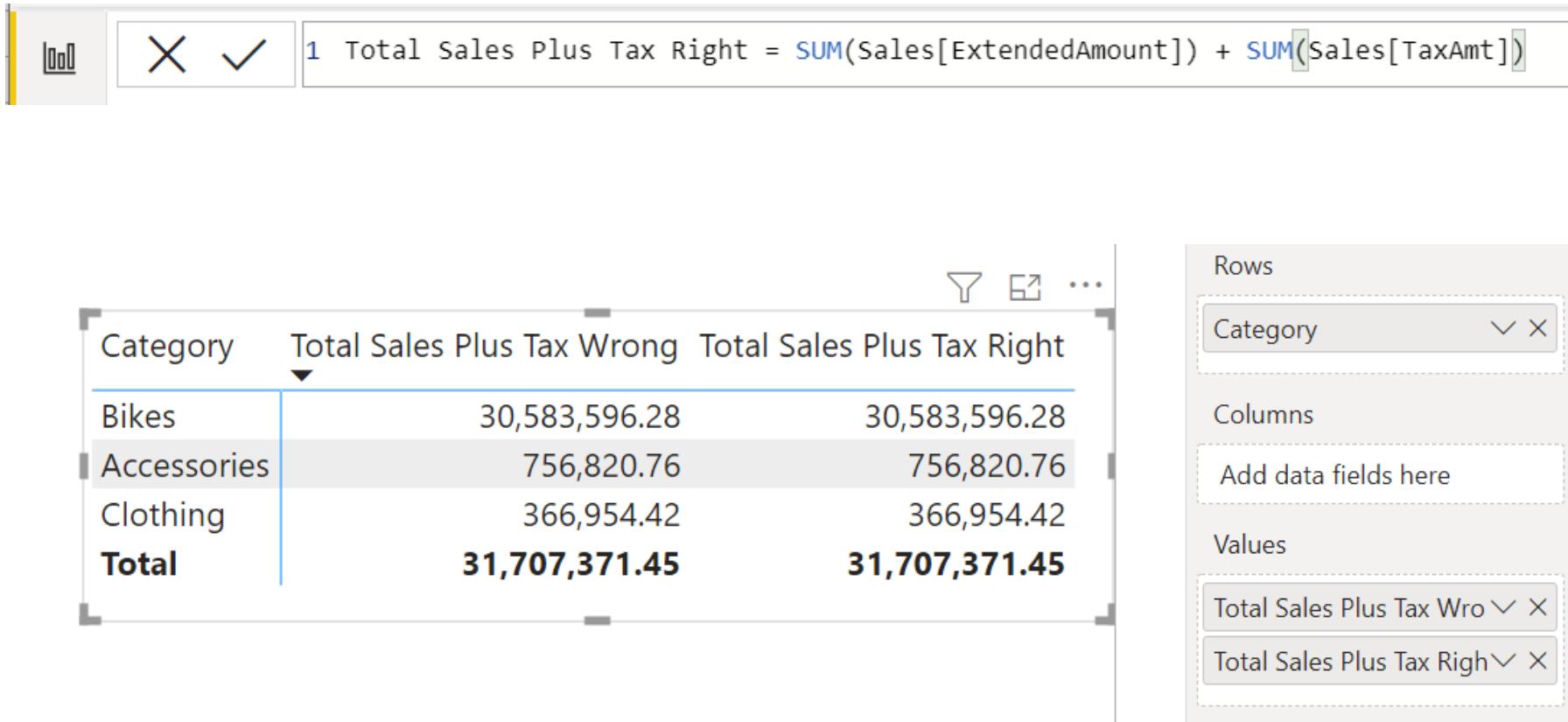
Toolbars and Context: The interface includes various toolbar icons (grid, filter, search) and a context pane on the right showing the current rows, columns, and values selected.

SalesOrderNumber	SalesOrderLineNumber	OrderQuantity	UnitPrice	ExtendedAmount	TaxAmt	Freight
072061		1	564.99	564.99	45.1992	14.1248
072062		1	564.99	564.99	45.1992	14.1248
072066		1	24.49	24.49	1.9592	0.6123
072138		1	8.99	8.99	0.7192	0.2248
072185		1	8.99	8.99	0.7192	0.2248
072271		1	564.99	564.99	45.1992	14.1248
072284		1	62.00	62.00	4.9102	1.8402

Total Sales Plus Tax Column		
539.99	43.1992	\$583.19
159	12.72	\$171.72
120	9.6	\$129.60
1	742.35	\$801.74
769.49	61.5592	\$831.05
564.99	45.1992	\$610.19

Category	Total Sales Plus Tax Wrong
Bikes	30,583,596.28
Accessories	756,820.76
Clothing	366,954.42
Total	31,707,371.45

Use New Measure Instead



The screenshot shows a Power BI interface with a table comparing two measures across different product categories. The table has three columns: Category, Total Sales Plus Tax Wrong, and Total Sales Plus Tax Right. The rows include Bikes, Accessories, Clothing, and a total row labeled 'Total'. The 'Total Sales Plus Tax Right' column contains identical values to the 'Total Sales Plus Tax Wrong' column, except for the total row which is bolded.

Category	Total Sales Plus Tax Wrong	Total Sales Plus Tax Right
Bikes	30,583,596.28	30,583,596.28
Accessories	756,820.76	756,820.76
Clothing	366,954.42	366,954.42
Total	31,707,371.45	31,707,371.45

Rows

Category

Columns

Add data fields here

Values

Total Sales Plus Tax Wro

Total Sales Plus Tax Righ

Big Problem with Calculated Columns

- Calculated columns make your tables and workbooks bigger.
- Mostly (though not always) you can get the same outcome by using a measure, which is a better way.

You should not use calculated columns if you can

- Use a measure instead
- Bring the data into a table directly from the source data
- Create the column during data load by using Power Query

When to use calculated columns

- You need to filter/slice a visual based on the results of a column (i.e., you want to use the column on Filters, Slicer, Rows, or Columns). Measures don't work in this case.
- You can't bring the column of data you need in from your source data or by using Power Query (for whatever reason).

Example: Day Type Calculated Column (1)

Screenshot of Power BI Desktop showing the creation of a calculated column.

The ribbon is set to **Table tools**.

The table is named **Calendar**.

The **Fields** pane shows various columns and their properties. An orange arrow points from the search bar in the Fields pane to the context menu, which is open over the **New column** option.

The context menu options include:

- New measure
- New column (highlighted)
- New quick measure
- Refresh data
- Edit query
- Manage relationships
- Incremental refresh
- Manage aggregations
- Copy Table
- Rename
- Delete from model
- Hide in report view
- Mark as date table
- Unhide all
- Collapse all
- Expand all

The table structure is as follows:

ID	Date	DayNumberOfWeek	DayName	MonthName	MonthNumberOfYear	DayNumberOfYear	WeekNumberOfYear	CalendarQuarter	CalendarYear	Fiscal Year	FiscalQuarter
1	Friday, 1 July 2016	6	Friday	July		7	183	27	3	2016	2016
2	Saturday, 2 July 2016	7	Saturday	July		7	184	27	3	2016	2016
3	Sunday, 3 July 2016	1	Sunday	July		7	185	27	3	2016	2016
4	Monday, 4 July 2016	2	Monday	July		7	186	28	3	2016	2016
5	Tuesday, 5 July 2016	3	Tuesday	July		7	187	28	3	2016	2016
6	Wednesday, 6 July 2016	4	Wednesday	July		7	188	28	3	2016	2016
7	Thursday, 7 July 2016	5	Thursday	July		7	189	28	3	2016	2016
8	Friday, 8 July 2016	6	Friday	July		7	190	28	3	2016	2016
9	Saturday, 9 July 2016	7	Saturday	July		7	191	28	3	2016	2016
10	Sunday, 10 July 2016	1	Sunday	July		7	192	28	3	2016	2016
11	Monday, 11 July 2016	2	Monday	July		7	193	29	3	2016	2016
12	Tuesday, 12 July 2016	3	Tuesday	July		7	194	29	3	2016	2016
13	Wednesday, 13 July 2016	4	Wednesday	July		7	195	29	3	2016	2016
14	Thursday, 14 July 2016	5	Thursday	July		7	196	29	3	2016	2016
15	Friday, 15 July 2016	6	Friday	July		7	197	29	3	2016	2016
16	Saturday, 16 July 2016	7	Saturday	July		7	198	29	3	2016	2016
17	Sunday, 17 July 2016	1	Sunday	July		7	199	29	3	2016	2016
18	Monday, 18 July 2016	2	Monday	July		7	200	30	3	2016	2016
19	Tuesday, 19 July 2016	3	Tuesday	July		7	201	30	3	2016	2016
20	Wednesday, 20 July 2016	4	Wednesday	July		7	202	30	3	2016	2016
21	Thursday, 21 July 2016	5	Thursday	July		7	203	30	3	2016	2016
22	Friday, 22 July 2016	6	Friday	July		7	204	30	3	2016	2016
23	Saturday, 23 July 2016	7	Saturday	July		7	205	30	3	2016	2016
24	Sunday, 24 July 2016	1	Sunday	July		7	206	30	3	2016	2016



Example: Day Type Calculated Column (1)

```
Day Type = IF( 'Calendar'[DayNumberOfWeek] = 1 ||  
'Calendar'[DayNumberOfWeek]=7, "Weekend", "Weekday" )
```

ID	Date	DayNumberOfWeek	DayName
1	Friday, 1 July 2016	6	Friday
2	Saturday, 2 July 2016	7	Saturday
3	Sunday, 3 July 2016	1	Sunday
4	Monday, 4 July 2016	2	Monday
5	Tuesday, 5 July 2016	3	Tuesday
6	Wednesday, 6 July 2016	4	Wednesday
7	Thursday, 7 July 2016	5	Thursday
8	Friday, 8 July 2016	6	Friday
9	Saturday, 9 July 2016	7	Saturday
10	Sunday, 10 July 2016	1	Sunday
11	Monday, 11 July 2016	2	Monday
12	Tuesday, 12 July 2016	3	Tuesday
13	Wednesday, 13 July 2016	4	Wednesday
14	Thursday, 14 July 2016	5	Thursday
15	Friday, 15 July 2016	6	Friday
16	Saturday, 16 July 2016	7	Saturday
17	Sunday, 17 July 2016	1	Sunday
18	Monday, 18 July 2016	2	Monday
19	Tuesday, 19 July 2016	3	Tuesday
20	Wednesday, 20 July 2016	4	Wednesday
21	Thursday, 21 July 2016	5	Thursday
22	Friday, 22 July 2016	6	Friday
23	Saturday, 23 July 2016	7	Saturday
24	Sunday, 24 July 2016	1	Sunday
25	Monday, 25 July 2016	2	Monday

Different forms of logical expression

- Day Type = IF(
'Calendar'[DayNumberOfWeek] = 1 ||
'Calendar'[DayNumberOfWeek] = 7,
"Weekend", "Weekday")
- IF(OR('Calendar'[DayNumberOfWeek] = 1,
'Calendar'[DayNumberOfWeek] = 7)
"Weekend", "Weekday")
- IF('Calendar'[DayNumberOfWeek] IN {1, 7},
"Weekend", "Weekday")

Power BI arithmetic operator

Arithmetic operator	Meaning	Example
+ (plus sign)	Addition	3+3
- (minus sign)	Subtraction or sign	3-1-1
* (asterisk)	Multiplication	3*3
/ (forward slash)	Division	3/3
^ (caret)	Exponentiation	16^4

Power BI comparison operators

Comparison operator	Meaning	Example
=	Equal to	[Region] = "USA"
==	Strict equal to	[Region] == "USA"
>	Greater than	[Sales Date] > "Jan 2009"
<	Less than	[Sales Date] < "Jan 1 2009"
>=	Greater than or equal to	[Amount] >= 20000
<=	Less than or equal to	[Amount] <= 100
<>	Not equal to	[Region] <> "USA"

Power BI logical operators

Text operator	Meaning	Examples
&& (double ampersand)	Creates an AND condition between two expressions that each have a Boolean result. If both expressions return TRUE, the combination of the expressions also returns TRUE; otherwise the combination returns FALSE.	([Region] = "France") && ([BikeBuyer] = "yes")
(double pipe symbol)	Creates an OR condition between two logical expressions. If either expression returns TRUE, the result is TRUE; only when both expressions are FALSE is the result FALSE.	(([Region] = "France") ([BikeBuyer] = "yes"))
IN	Creates a logical OR condition between each row being compared to a table. Note: the table constructor syntax uses curly braces.	'Product'[Color] IN { "Red", "Blue", "Black" }

Operator Precedence

Higher Precedence
Execute First



Operator	Description
$^$	Exponentiation
$-$	Sign (as in -1)
$*$ and $/$	Multiplication and division
!	NOT (unary operator)
$+$ and $-$	Addition and subtraction
$\&$	Connects two strings of text (concatenation)
$=, ==, <, >, <=, >=, <>$	Comparison

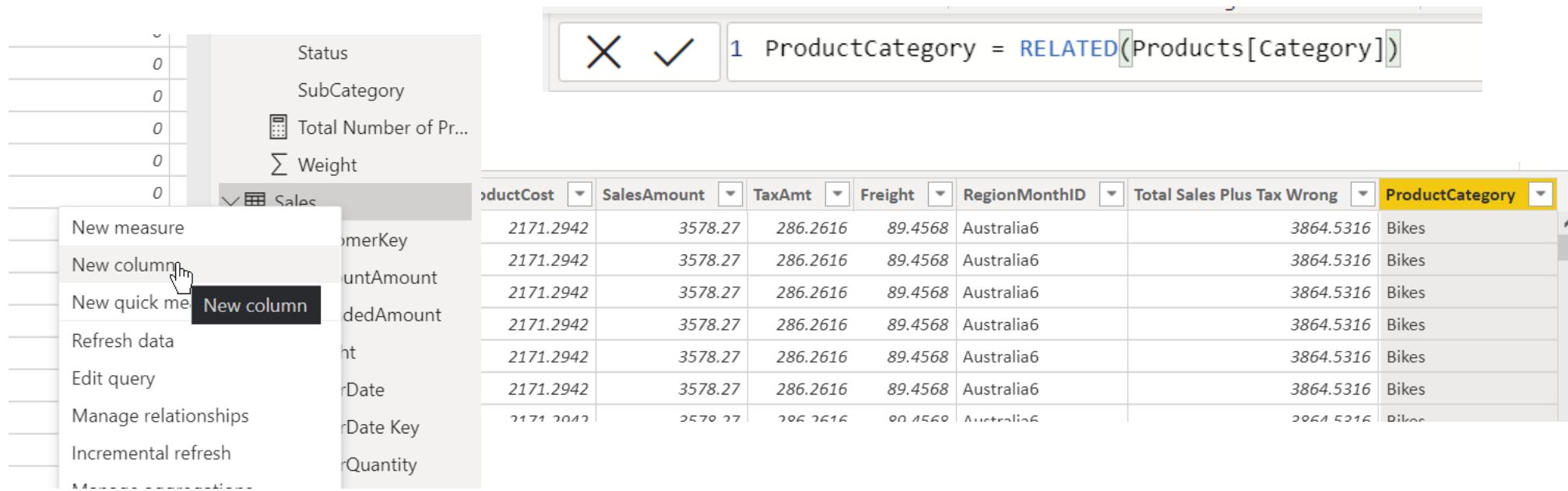


DAX, Calculated Columns Practices

- Creating a Half-Year column
 - In the Calendar table, write a calculated column that returns the value H1 for the first half of each year (January through June) and returns H2 for the second half of each year (July through December). Hint: You might want to use an IF statement to do this.

DAX, RELATED

- The functions RELATED() is typically used in calculated columns to reference relevant records in other tables



A screenshot of the Power BI desktop interface. On the left, there's a data view pane showing columns like Status, SubCategory, Total Number of Pr..., Weight, and Sales. A context menu is open over the Sales column, with the 'New column' option highlighted. To the right, a DAX formula bar shows the formula: `ProductCategory = RELATED(Products[Category])`. Below the formula bar is a table with several rows of data. The table has columns: ProductCost, SalesAmount, TaxAmt, Freight, RegionMonthID, Total Sales Plus Tax Wrong, and ProductCategory. All rows show the same value for ProductCategory: "Bikes".

ProductCost	SalesAmount	TaxAmt	Freight	RegionMonthID	Total Sales Plus Tax Wrong	ProductCategory
2171.2942	3578.27	286.2616	89.4568	Australia6	3864.5316	Bikes
2171.2942	3578.27	286.2616	89.4568	Australia6	3864.5316	Bikes
2171.2942	3578.27	286.2616	89.4568	Australia6	3864.5316	Bikes
2171.2942	3578.27	286.2616	89.4568	Australia6	3864.5316	Bikes
2171.2942	3578.27	286.2616	89.4568	Australia6	3864.5316	Bikes
2171.2942	3578.27	286.2616	89.4568	Australia6	3864.5316	Bikes
2171.2942	3578.27	286.2616	89.4568	Australia6	3864.5316	Bikes
2171.2942	3578.27	286.2616	89.4568	Australia6	3864.5316	Bikes

DAX, Iterators

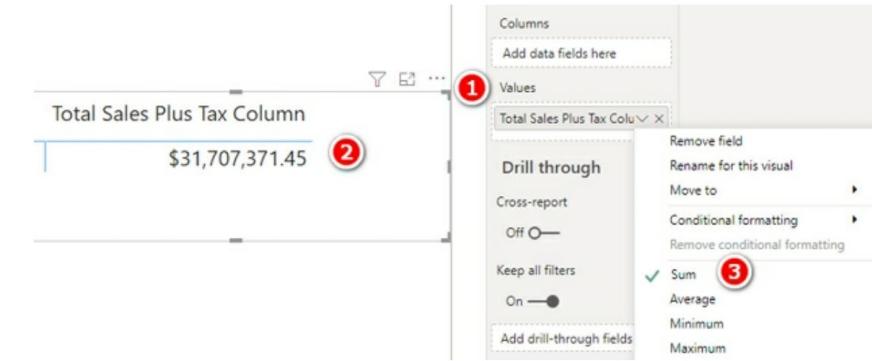
- SUM(), COUNT(), etc. are called “aggregation functions”
- There is another class of functions that perform aggregation after expression. These X-functions are part of the iterator family.
- These iterators are measures.

Iterating function vs calculated columns

Total Sales Plus Tax Column =
 $Sales[ExtendedAmount] + Sales[TaxAmt]$

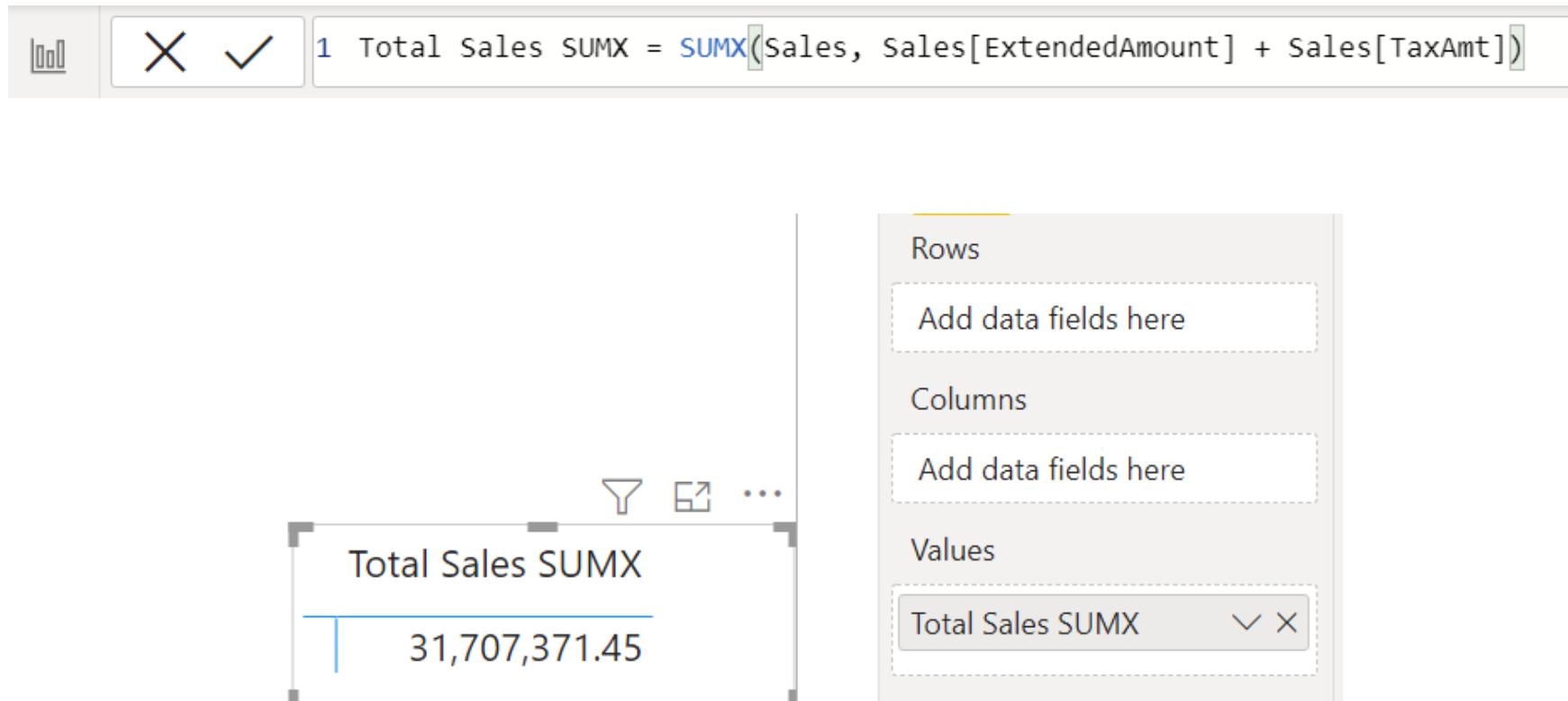
VS

1 | Total Sales Including Tax SUMX Version =
2 | $SUMX (Sales, Sales[ExtendedAmount] + Sales[TaxAmt])$



The big difference between iterating functions like `SUMX()` and calculated columns is that a calculated column permanently stores the results in the new column.

DAX, iterator: SUMX()



The screenshot shows a Power BI interface with a DAX formula editor at the top and a visual representation below.

DAX Formula:

```
1 Total Sales SUMX = SUMX(Sales, Sales[ExtendedAmount] + Sales[TaxAmt])
```

Visual Result:

A card visual displays the calculated total sales value.

Total Sales SUMX
31,707,371.45

Data Source (Rows, Columns, Values):

- Rows: Add data fields here
- Columns: Add data fields here
- Values: Total Sales SUMX

DAX, iterator: SUMX() Practices

- [Total Sales SUMX version]
 - Multiply quantity by unit price
- [Total Sales Including Freight]
 - Add the ExtendedAmount column to the freight cost

DAX, iterator: AVERAGEX Practices

- [Average Selling Price Including Tax Per Item]
 - Selling price + Tax
- [Average profit per item sold]
 - Profit is selling price – cost price

Other iterators

- MAXX()
- MINX()
- COUNTAX()
- COUNTX()

CALCULATE()

- CALCULATE() is the most important and powerful function in the DAX language.
- It is important because it is the only function that has the ability to modify the natural filtering behavior of visuals.
- CALCULATE() alters the natural filtering behaviour of visuals.
- It modifies an expression (which can be a measure or another DAX formula) by applying, removing, or modifying filters.
- The syntax of CALCULATE() is:

 $= \text{CALCULATE}(\text{expression}, \text{filter 1}, \text{filter 2}, \dots, \text{filter n})$

Simple filter

- Suppose we want only the total sales of blue product

Total Sales of Blue Products = CALCULATE([Total Sales], Products[Color]="Blue")

Category	Total Sales
Accessories	\$700,760
Bikes	\$28,318,145
Clothing	\$339,773
Total	\$29,358,677

Category	Total Sales	Total Sales of Blue Products
Accessories	\$700,760	\$74,354
Bikes	\$28,318,145	\$2,169,056
Clothing	\$339,773	\$35,687
Total	\$29,358,677	\$2,279,096

DAX: CALCULATE() Practices, one table

- [Total Male Customers], for each occupation
- [Total Customers Born Before 1970], use DATE() function for creating date reference, for each occupation
- [Customers earning at least \$100,000 per year], for each occupation



DAX: CALCULATE() Practices, multiple table

- Use Region as Row Header
- [Total Sales] as a reference, not use CALCULATE
- [Total Sales of Clothing], for Region
- [Sales to Female Customers]



Thank you

Question?

