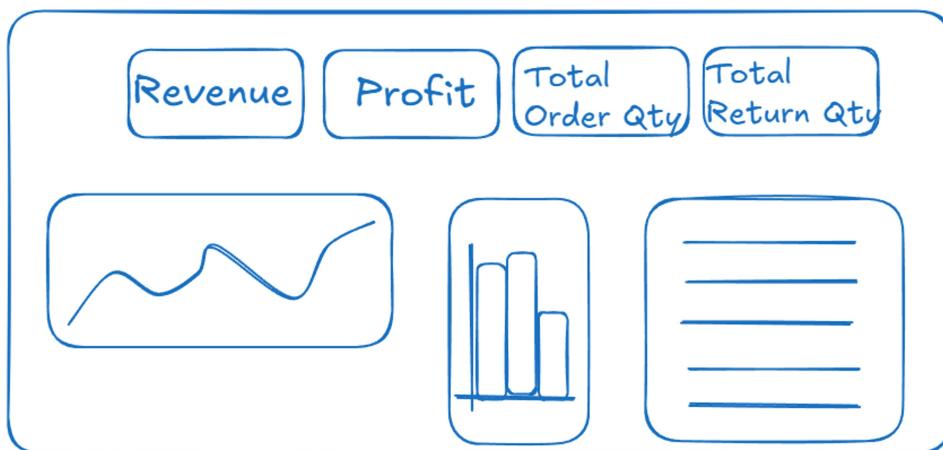
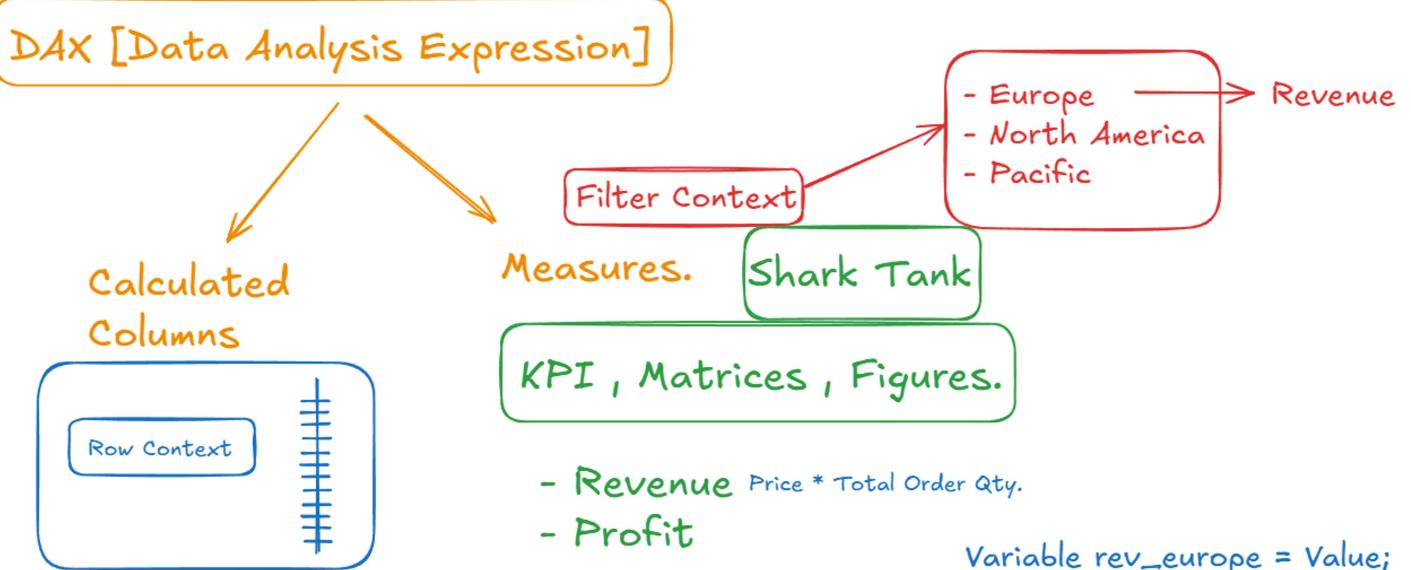


## Power BI - DAX - p1

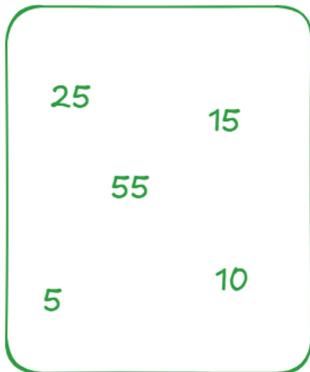


A table shows the relationship between **Total No. of Children** and **Is Married/Not** on one side, and **Is Parent?** on the other. The **Is Parent?** column has two rows: **True** and **False**.

Total No. of Children	Is Married/Not	Is Parent?
2	True	True
1	True	True
0	False	False
1	True	
2	True	
3	True	
0	False	

An orange box labeled **Row Context** points to a red box containing the following DAX functions:

- **IFS()**
- **SUMIFS()**
- **COUNTIF()**
- **AVERAGEIF()**
- **MAXIF()**



110 - SUM  
 $110/5 = 22$  - Average  
 55 - MAX  
 5 - MIN



Aggregation : Can't be perform without Grouping the data.

TableName[ColumnName]

Total Orders = **DISTINCTCOUNT**(Sales\_Data[OrderNumber])

urns Data

Total Revenue = **SUMX**(Sales\_Data, Sales\_Data TableName[ColumnName]  
 [OrderQuantity] \* RELATED(Product\_Lookup[ProductPrice]))

Quantity Ordered = **SUM**(Sales\_Data[OrderQuantity])

~~Clothing~~

~~Bike Returns~~ =

**CALCULATE**(

[Total Returns], Measure → Calling the existing measure

'Product Categories Lookup'[Category Name] = "Bikes" Filter

)

Category Name	Total Returns	Bike Returns	Clothing Returns
Accessories	1,115		
Bikes	427	427	
Clothing	267		
<b>Total</b>	<b>1,809</b>	<b>427</b>	<b>267</b>

## CALCULATED FIELDS WITH DAX

In this section we'll use Data Analysis Expressions (DAX) to add calculated columns & measures to our model, and introduce topics like row & filter context, iterators and more

### TOPICS WE'LL COVER:

- 1. DAX
- 2. Row & Filter Context
- 3. Common Functions
- 4. Iterators
- 5. Columns & Measures
- 6. DAX Syntax
- 7. Calculate
- 8. Time Intelligence

## GOALS FOR THIS SECTION:

- Introduce DAX fundamentals and learn when to use calculated columns and measures.
- Understand the difference between row context and filter context, and how they impact DAX calculations.
- Learn DAX formula syntax, basic operators and common function categories (math, logical, text, date/time, filter, etc.).
- Explore nested functions, and more complex topics like iterators and time intelligence patterns.

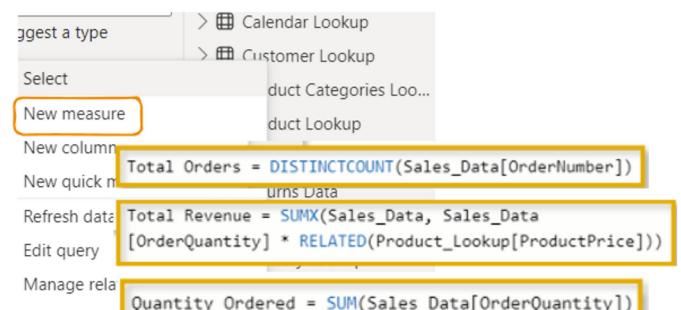
## MEET DAX

Data Analysis Expressions (commonly known as DAX) is the formula language that drives the Power BI front-end. With DAX, you can:

- Go beyond the capabilities of traditional spreadsheet formulas, with powerful and flexible functions built specifically to work with relational data models.
- Add calculated columns (for filtering) and measures (for aggregation) to enhance data models.



A screenshot of the Power BI Data View interface. A table is displayed with columns: Marital Status, Email Address, Annual Income, Total Children, Education Level, and Parent. The 'Parent' column contains the formula: `Parent = IF('Customer Lookup'[Total Children]>0,"Yes","No")`. The 'Email Address' column shows email addresses for various customers, and the 'Annual Income' column shows their annual income levels.



# M VS. DAX

M and DAX are two distinct functional languages used within Power BI Desktop:

- M is used in the Power Query editor, and is designed specifically for extracting, transforming and loading data.
- DAX is used in the Power BI front-end, and is designed specifically for analyzing relational data models

## M CODE

Query Editor:

```

Properties
Name: Territory Lookup
All Properties

Applied Steps
Source
Promoted Headers
Changed Type (highlighted)

#"Changed Type" = Table.TransformColumnTypes(
#"Promoted Headers",
{
    {"SalesTerritoryKey", Int64.Type},
    {"Region", type text},
    {"Country", type text},
    {"Continent", type text}
})

```

## DAX

Report View:

Category Name	Total Returns	Bike Returns
Accessories	1,115	
Bikes	427	427
Clothing	267	
<b>Total</b>	<b>1,809</b>	<b>427</b>

Explicit Measures

```

1 Bike Returns =
2 CALCULATE(
3     [Total Returns],
4     'Product Categories Lookup'[Category Name] = "Bikes"
5 )

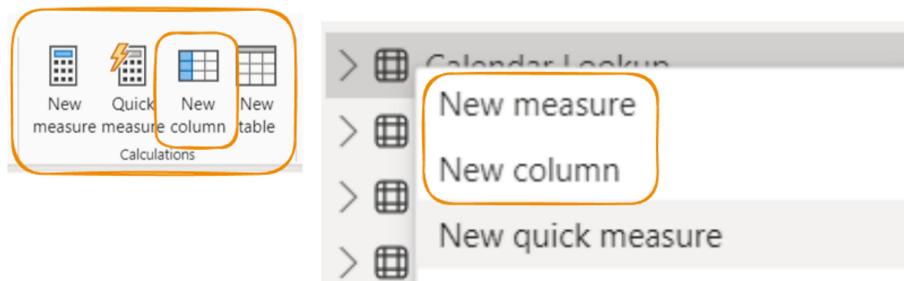
```

Existing Measure

```

// Counting total returns
// filtered for bikes only

```



File Home Help Table tools

Name: Calendar Lookup

Structure

Mark as date table Calendars

Manage relationships Relationships

Calculations

Date	Day Name	Start of Week	Start of Month	Month Name	Start of Year
01-01-2020 00:00:00	Wednesday	29-12-2019	01-01-2020	January	01 January 2020
02-01-2020 00:00:00	Thursday	29-12-2019	01-01-2020	January	01 January 2020
03-01-2020 00:00:00	Friday	29-12-2019	01-01-2020	January	01 January 2020
04-01-2020 00:00:00	Saturday	29-12-2019	01-01-2020	January	01 January 2020

IF(**LogicalTest**, **ResultIfTrue**, [**ResultIfFalse**])  
Checks whether a condition is met, and returns one value if TRUE, and another value if FALSE.

Parent = IF('Customer Lookup'[TotalChildren] > 0,"Yes","No")

 Parent

## Relative Reference

```
TotalQuantity = SUM('Sales Record 2020-2022'[OrderQuantity])
```

The screenshot shows the 'Add Column' dialog box from Microsoft Power BI. On the left, there's a vertical toolbar with icons for different data types like Date, Number, and Text. Below it is a 'Suggest a type' button. The main area is titled 'Columns' and contains two rows of columns. The first row has one column labeled 'CategoryName'. The second row has one column labeled 'Total Returns', which is highlighted with an orange border. At the bottom of the list is a button labeled '+Add data'. To the right of the list, a vertical scroll bar is visible. A large orange box highlights the 'Sum' option in the list of aggregation functions.

Don't summarize

Sum

Average

Minimum

Maximum

Count (Distinct)

Count

Standard deviation

Variance

Median

Sum

CategoryName	Total Returns
Accessories	1130
Bikes	429
Clothing	269
<b>Total</b>	<b>1828</b>

## Implicit Measure -Report View

Rows  
Columns  
Value  
Filter

The screenshot shows the 'Suggest a type' dialog box from Power BI. It has a title bar with 'Suggest a type' and a close button. Below the title bar, there are two sections: 'Columns' and 'Data'. The 'Columns' section contains a table with three rows: 'CategoryName' (with a dropdown arrow), 'Total Returns' (with a dropdown arrow), and '+ Add data'. The 'Data' section contains a table with two rows: 'Field' (containing 'Sum of ReturnQuantity') and 'Aggregation' (containing 'Sum'). A red arrow points from the 'off' radio button in the top-left corner to the 'Aggregation' row in the 'Data' section.

values

## CALCULATED COLUMNS

---

Calculated columns allow you to add new, formula-based columns to tables in a model

- Calculated columns refer to entire tables or columns (no A1-style cell references).
- Calculated columns generate values for each row, which are visible within tables in the Data view.
- Calculated columns understand row context; they're great for defining properties based on information in each row, but generally useless for aggregation (sum, count, etc.)

### HEY THIS IS IMPORTANT!

As a rule of thumb, use calculated columns to "stamp" static, fixed values to each row in a table (or go upstream and use the Query Editor!)  
DO NOT use calculated columns for aggregation – this is what measures are for!

### PRO TIP:

Calculated columns are typically used for filtering & grouping data, rather than creating aggregate numerical values

## EXAMPLE: CALCULATED COLUMNS

A screenshot of the Power BI Data View interface. A calculated column named "Parent" is being defined with the formula: `1 Parent = IF('Customer Lookup'[Total Children]>0,"Yes","No")`. The formula bar shows the formula with a green checkmark. The table contains columns: Email Address, Annual Income, Total Children, Education Level, and the newly created Parent column. The Parent column values are all "Yes". The Fields pane on the right shows the Customer Lookup table with fields like Annual Income, Birth Date, Customer Key, Education Level, Email Address, and First Name.

In this case we've added a calculated column named Parent, which equals "Yes" if the [Total Children] field is greater than 0, and "No" otherwise

- Since calculated columns understand row context, a new value is calculated in each row based on the value in the [Total Children] column.
- This is a valid use of calculated columns; it creates a new row "property" that we can use to filter or segment any related data within the model.

Here we're using an aggregation function (SUM) to calculate a new column named Total Quantity

A screenshot of the Power BI Data View interface. A calculated column named "TotalQuantity" is being defined with the formula: `1 TotalQuantity = SUM('Sales Data'[Order Quantity])`. The formula bar shows the formula with a green checkmark. The table contains columns: Order Date, Order Number, Product Key, Customer Key, Territory Key, Order Line Item, Order Quantity, Index, and the newly created TotalQuantity column. The TotalQuantity column values are all 84174. The Fields pane on the right shows the Sales Data table with fields like Order Date, Order Number, Product Key, Customer Key, Territory Key, Order Line Item, Order Quantity, Index, and TotalQuantity.

- Since this is an aggregation function, the same grand total is returned in every row of the table
- This is not a valid use of calculated columns; these values are statically "stamped" onto the table and can't be filtered, sliced, etc.

## DAX MEASURES



Measures are DAX formulas used to generate new calculated values

- Like calculated columns, measures reference entire tables or columns (no A1-style cell references).
- Unlike calculated columns, measures aren't visible within tables; they can only be "seen" within a visualization like a chart or matrix (similar to a calculated field in a PivotTable).
- Measures evaluate based on filter context, which means they recalculate when the fields or filters around them change.

### HEY THIS IS IMPORTANT!

As a rule of thumb, use measures when a single row can't give you the answer, or when you need to aggregate values across multiple rows in a table

### PRO TIP:

Use measures to create numerical, calculated values that can be analyzed in the "values" field of a report visual

## IMPLICIT VS. EXPLICIT MEASURES

The screenshot shows the 'Build a visual' interface in Power BI. On the left, under 'Y-axis', there is a box labeled 'Sum of Order ...' with a yellow arrow pointing to it. To its right is a 'Select data' pane with a search bar and a list of fields. One item, 'Σ Order Quantity', is highlighted with a yellow box and has a yellow arrow pointing to it from the Y-axis box.

Example of an implicit measure

Implicit measures are created when you drag raw numerical fields into a report visual and manually select an aggregation mode (Sum, Average, Min, Max, Count, etc.)

Explicit measures are created when you actually write a DAX formula and define a new measure that can be used within the model

### HEY THIS IS IMPORTANT!

Implicit measures are only accessible within the specific visualization in which they were created, and cannot be referenced elsewhere.

Explicit measures can be used anywhere in the report, and referenced by other DAX calculations to create "measure trees".

The screenshot shows the 'Build a visual' interface. In the 'Values' section, there are two boxes: 'Sum of OrderQuantity' and 'Sum of ReturnQuantity'. To the right of these boxes is a dropdown menu with various aggregation options. The 'Sum' option is highlighted with a blue box and has a blue arrow pointing to it from the 'Values' boxes.

Implicit Measures.

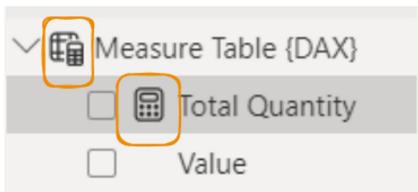
Explicit Measure

Black\_Quantity\_Sold = 10,590

Europe

Pacific  
North America

Europe\_QS-black



Total Quantity = `SUM('Sales Record 2020-2022'[OrderQuantity])`

Total Order Quantity  
**84K**

## FILTER CONTEXT

Measures are evaluated based on filter context, which means that they recalculate whenever the fields or filters around them change

Top 10 Products	Orders	Revenue	Return %
Water Bottle - 30 oz.	3,983	\$39,755	1.95%
Patch Kit/8 Patches	2,952	\$13,506	1.61%
Mountain Tire Tube	<b>2,846</b>	\$28,333	1.64%
Road Tire Tube	2,173	\$17,265	1.55%
Sport-100 Helmet, Red	2,099	\$73,444	3.33%
AWC Logo Cap	2,062	\$35,865	1.11%
Sport-100 Helmet, Blue	1,995	\$67,112	3.31%
Fender Set - Mountain	1,975	\$87,041	1.36%
Sport-100 Helmet, Black	1,940	\$65,262	2.68%
Mountain Bottle Cage	1,896	\$38,062	2.02%
<b>Total</b>	<b>15,587</b>	<b>\$465,644</b>	<b>1.85%</b>

For this value in the matrix (2,846), the Orders measure is calculated based on the following filter context:  
Products[Product Name] = "Mountain Tire Tube"

- This allows the measure to return the total order quantity for each product specifically (or whatever context the row and column labels dictate – years, countries, categories, customer names, etc.)

This total (15,587) does NOT calculate by summing the values above; it evaluates as an independent measure with no filter context applied

- IMPORTANT:** Every measure value in a report evaluates independently (like an island) and calculates based on its own filter context

The screenshot shows a donut chart with the following data:

- High 2.8K
- Average 11.6K
- Low 10.3K

Below the chart, a sidebar lists the filters and slicers affecting the visual:

- Customer Metric Selection**: is Total Customers
- IncomeLevel**: is Average, High, or Low
- Year**: is 2021 or 2022

**PRO TIP:** Clicking the filter icon will show you the filters currently applied to a selected visual

## EXAMPLE: FILTER CONTEXT

**MEASURE: Revenue Per Customer**

**FILTER CONTEXT:**

- *Calendar[Year] = 2021 or 2022*

**MEASURE: Total Customers**

**FILTER CONTEXT:**

- *Calendar[Date] = September 26, 2021*

**Filters**

Filters on this page  
Year is 2021 or 2022

Add data fields here

Filters on all pages

Add data fields here

This is a **page-level filter**, which impacts **ALL** visuals on this report page (more on this later!)

**MEASURE: Total Orders**

**FILTER CONTEXT:**

- *Calendar[Year] = 2021 or 2022*
- *Customers[Occupation] = Skilled Manual*

**MEASURE: Total Revenue**

**FILTER CONTEXT:**

- *Calendar[Year] = 2021 or 2022*
- *Customer[Full Name] = Top 100 by Total Orders*

**MEASURE: Total Revenue**

**FILTER CONTEXT:**

- *Calendar[Year] = 2021 or 2022*
- *Customer[Full Name] = Mr. Maurice Shan*

**COLUMN: Customer Full Name**

**FILTER CONTEXT:**

- *Calendar[Year] = 2021 or 2022*
- *Customer[Full Name] = Top 1 by Total Revenue*

## STEP-BY-STEP MEASURE CALCULATION

Product Color	Quantity Sold
Black	10,590
Red	4,011
Yellow	4,638

How exactly is this measure value calculated?

- NOTE: This all happens instantly behind the scenes, every time the filter context changes

### STEP 1

Filter context is detected & applied



Product Color	Quantity Sold
Black	10,590
Red	4,011
Yellow	4,638

'Product Lookup'[Product Color] = "Black"

Product Lookup Table	
Product ID	Product Name
P00000001	Product Name 01
P00000002	Product Name 02
P00000003	Product Name 03
P00000004	Product Name 04
P00000005	Product Name 05
P00000006	Product Name 06
P00000007	Product Name 07
P00000008	Product Name 08
P00000009	Product Name 09
P00000010	Product Name 10
P00000011	Product Name 11
P00000012	Product Name 12
P00000013	Product Name 13
P00000014	Product Name 14
P00000015	Product Name 15
P00000016	Product Name 16
P00000017	Product Name 17
P00000018	Product Name 18
P00000019	Product Name 19
P00000020	Product Name 20
P00000021	Product Name 21
P00000022	Product Name 22
P00000023	Product Name 23
P00000024	Product Name 24
P00000025	Product Name 25
P00000026	Product Name 26
P00000027	Product Name 27
P00000028	Product Name 28
P00000029	Product Name 29
P00000030	Product Name 30
P00000031	Product Name 31
P00000032	Product Name 32
P00000033	Product Name 33
P00000034	Product Name 34
P00000035	Product Name 35
P00000036	Product Name 36
P00000037	Product Name 37
P00000038	Product Name 38
P00000039	Product Name 39
P00000040	Product Name 40
P00000041	Product Name 41
P00000042	Product Name 42
P00000043	Product Name 43
P00000044	Product Name 44
P00000045	Product Name 45
P00000046	Product Name 46
P00000047	Product Name 47
P00000048	Product Name 48
P00000049	Product Name 49
P00000050	Product Name 50
P00000051	Product Name 51
P00000052	Product Name 52
P00000053	Product Name 53
P00000054	Product Name 54
P00000055	Product Name 55
P00000056	Product Name 56
P00000057	Product Name 57
P00000058	Product Name 58
P00000059	Product Name 59
P00000060	Product Name 60
P00000061	Product Name 61
P00000062	Product Name 62
P00000063	Product Name 63
P00000064	Product Name 64
P00000065	Product Name 65
P00000066	Product Name 66
P00000067	Product Name 67
P00000068	Product Name 68
P00000069	Product Name 69
P00000070	Product Name 70
P00000071	Product Name 71
P00000072	Product Name 72
P00000073	Product Name 73
P00000074	Product Name 74
P00000075	Product Name 75
P00000076	Product Name 76
P00000077	Product Name 77
P00000078	Product Name 78
P00000079	Product Name 79
P00000080	Product Name 80
P00000081	Product Name 81
P00000082	Product Name 82
P00000083	Product Name 83
P00000084	Product Name 84
P00000085	Product Name 85
P00000086	Product Name 86
P00000087	Product Name 87
P00000088	Product Name 88
P00000089	Product Name 89
P00000090	Product Name 90
P00000091	Product Name 91
P00000092	Product Name 92
P00000093	Product Name 93
P00000094	Product Name 94
P00000095	Product Name 95
P00000096	Product Name 96
P00000097	Product Name 97
P00000098	Product Name 98
P00000099	Product Name 99
P00000100	Product Name 100

The screenshot shows the Product Lookup Table with a context filter applied to the 'Color' column, where only rows for 'Black' are visible. A context menu is open on the right side of the screen.

## STEP 2

Filters flow “downstream” to related tables



The Sales Data table shows sales records for various dates. A context filter for 'Black' is applied to the 'Product Key' column. Below the tables is a diagram illustrating the data model's relationships:

```

    graph LR
      SD1[Sales Data] -- "1" --> RD1[Returns Data]
      SD1 -- "*" --> SD2[Sales Data]
  
```

The diagram shows two Sales Data tables connected by a many-to-many relationship (\*). The first Sales Data table is connected to the Returns Data table via a one-to-many relationship (1).

## STEP 3

Measure evaluates against the filtered table



```

1 Quantity Sold =
2 SUM(
3   |   'Sales Data'[Order Quantity]
4 )
  
```

- Sum of values in the **Order Quantity** column of the **Sales Data** table, filtered to rows where the product color is “**Black**”

= 10,590

# DAX SYNTAX

## MEASURE NAME

- Measures are always surrounded by brackets (i.e. [Total Quantity]) when referenced in formulas, so spaces are OK

Total Quantity: = SUM(Transactions[quantity])

## FUNCTION NAME

- Calculated columns don't always use functions, but measures do:
  - In a Calculated Column, =Transactions[quantity] returns the value from the quantity column in each row (since it evaluates one row at a time).
  - In a Measure, = Transactions[quantity] will return an error since Power BI doesn't know how to translate that as a single value – you need some sort of aggregation

## Referenced TABLE NAME

## Referenced COLUMN NAME

This is a "fully qualified" column, since it's preceded by the table name.

NOTE: Table names with spaces must be surrounded by single quotes:

- Without a space: Transactions[quantity]
- With a space: 'Transactions Table'[quantity]

## PRO TIP:

Column references use fully qualified names (i.e. 'Table'[Column])

Measure references just use the measure name (i.e. [Measure]) and can be called by typing an open square bracket "[ "