

MICROSOFT EXCEL:

INTRO TO POWER QUERY, POWER PIVOT & DAX

★★★★★ *With Best-Selling Excel instructor Chris Dutton*



GETTING STARTED

COURSE STRUCTURE

- ★ **100+ Downloadable PDF Slides** to serve as helpful reference guides when you're offline or on the go (or just need a refresher!)
- ★ **Interactive, hands-on demos** to keep you engaged, with **downloadable project files** that you can use to explore and learn at your own pace
- ★ **Course Quizzes** and **Homework Exercises** to test and reinforce key concepts throughout the course

COURSE OUTLINE

1

The “Power” Excel Landscape

- *Power Query/Power Pivot workflow and key benefits vs. “traditional” Excel*

2

Power Query

- *Types of data connectors, query editing tools, loading options, etc.*

3

Data Modeling 101

- *Excel Data Model interface, normalization, table relationships, hierarchies, etc.*

4

Power Pivot & DAX

- *Power Pivot vs. “normal” pivots, calculated columns vs. measures, row & filter context, etc.*

5

Common DAX Functions

- *Basic syntax, math & stats functions, filter functions, time intelligence tools, etc.*

6

Final Project

- *VanArsdel sales data (2000-2010)*

VERSIONS & COMPATIBILITY



IMPORTANT NOTE: Power Pivot is currently ***not available for Mac***, and is ***only available in certain versions of Excel for Windows/PC***

For a full, current list of compatible versions, visit **support.office.com** (or Google “Where is Power Pivot?”):
<https://support.office.com/en-us/article/Where-is-Power-Pivot-aa64e217-4b6e-410b-8337-20b87e1c2a4b> (or use: bit.ly/2yd80rd)

Other considerations:

- Power Pivot works best with **64-bit** Excel, which can access more processing power and memory (*not critical*)
 - *Note: make sure you're running a 64-bit operating system and that you've updated Office to the 64-bit version*
- Power Pivot menus, features and tools have evolved over time; **what you see on your screen may differ from what you see on mine, but the fundamental skills and concepts covered are universally applicable**
- Even if you have a compatible version of Excel, you may need to **enable the Power Pivot or Power Query plug-ins** to access the tools in this course (**File > Options > Add-Ins > Manage: COM Add-Ins**)

GETTING TO KNOW THE FOODMART DATABASE

- Throughout the course, we'll be using sample data from a fictitious super market chain called "FoodMart"^{*}
- In addition to daily transactional records from 1997-1998, our data set includes information about **products**, **customers**, **stores**, and **regions**
- All files are available for download in the **course resources** section of your course dashboard (***Course Dashboard > Course Content > All Resources***)



*This data is provided by Microsoft for informational purposes only as an aid to illustrate a concept. These samples are provided "as is" without warranty of any kind. The example companies, organizations, products, domain names, e-mail addresses, people, places, and events depicted herein are fictitious, and no association with any real company, organization, product, domain name, e-mail address, person, place, or event is intended or should be inferred.

SETTING EXPECTATIONS

1

I'm using Excel 2016 for PC (365 ProPlus, 64-bit)

- *Power Pivot is currently **not available** for Mac*
- *What you see on your screen **will not always match** what you see on mine (especially for Excel 2010 or 2013)*

2

This course is designed to get you **up & running** with Excel's BI tools

- *The goal is to provide a solid **foundational understanding** of Power Query, Power Pivot and DAX; we may simplify some concepts to make them easier to grasp, and will not cover some of the more advanced tools*

3

These tools are incredibly powerful, but still a little "**buggy**"

- *Power Pivot uses a lot of processing power, so it helps to **close other workbooks and applications***
- ***Save new versions** early and often; if you do crash, make sure you have a recent version to work from!*

4

When things get challenging, remember that I'm **here to help**

- *If you feel stuck, remember that you can pause the videos and rewatch them as many times as you'd like!*
- ***Still need support?** Post to the course Q&A section or message me directly and I'd be happy to lend a hand*

COURSE RATINGS & REVIEWS

Ratings and reviews help courses succeed, and provide valuable feedback that I can use to **make the course even better!**

- If you find yourself enjoying the course, or if you have feedback that might improve your experience, please take **15 seconds** to leave a rating or review (*when you're ready – no rush!*)

udemy Categories Search for Courses My Courses

Become an Instructor

My Courses

All Courses Collections Wishlist Archived

Microsoft Excel - Advanced Excel Formulas & Functions
Chris Dutton, Best-selling Udemy Instructor & Founder, Excel...
1% Complete ★★★★★ Your Rating

Microsoft Excel - Data Analysis with Excel Pivot Tables
Chris Dutton, Best-selling Udemy Instructor & Founder, Excel...
2% Complete ★★★★★ Your Rating

Microsoft Excel: Data Visualization w/ Excel Charts...
Chris Dutton, Best-selling Udemy Instructor & Founder, Excel...
2% Complete ★★★★★ Your Rating

STEP 1: Click on “My Courses”

STEP 2: Click on the stars under the course thumbnail

STEP 3: Dance

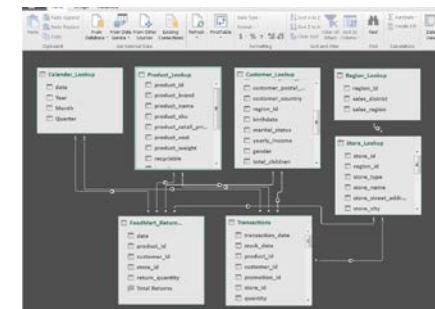
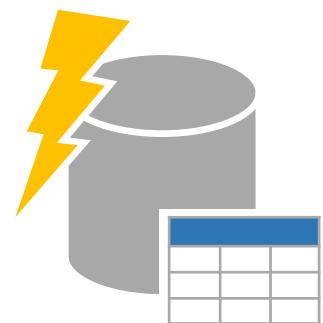
LET'S DO THIS.



INTRO TO “POWER EXCEL”

THE “POWER EXCEL” WORKFLOW

These are Excel’s **Business Intelligence** tools, all of which are available directly in Excel (*provided you have a compatible version*); **no additional software is required!**



RAW DATA

Flat files (csv, txt), Excel tables, databases (SQL, Azure), folders, streaming sources, web data, etc.

POWER QUERY

(aka “*Get & Transform*”)

Connect to sources, import data, and apply shaping and transformation tools (ETL)

DATA MODEL

Create table relationships, add calculated columns, define hierarchies and perspectives, etc.

POWER PIVOT & DAX

Explore and analyze the entire data model, and create powerful measures using Data Analysis Expressions (DAX)

“THE BEST THING TO HAPPEN TO EXCEL IN 20 YEARS”

- **Import and analyze MILLIONS of rows of data in Excel**
 - *Access data from virtually anywhere (database tables, flat files, cloud services, folders, etc.)*
- **Quickly build models to blend and analyze data across sources**
 - *Instantly connect sources and analyze holistic performance across your entire data model*
- **Create fully automated data shaping and loading procedures**
 - *Connect to databases and watch data flow through your model with the click of a button*
- **Define calculated measures using Data Analysis Expressions (DAX)**
 - *No more redundant A1-style “grid” formulas; DAX expressions are flexible, powerful and portable*

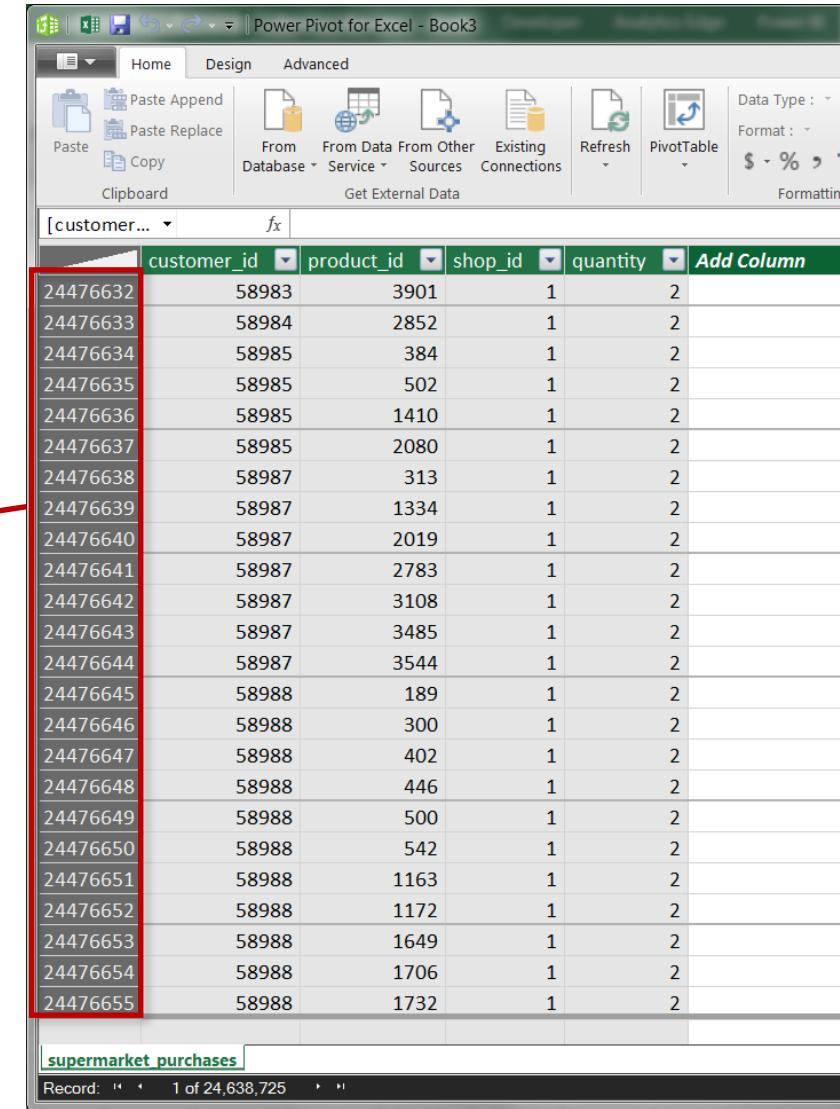


#1: IMPORT & ANALYZE MILLIONS OF ROWS



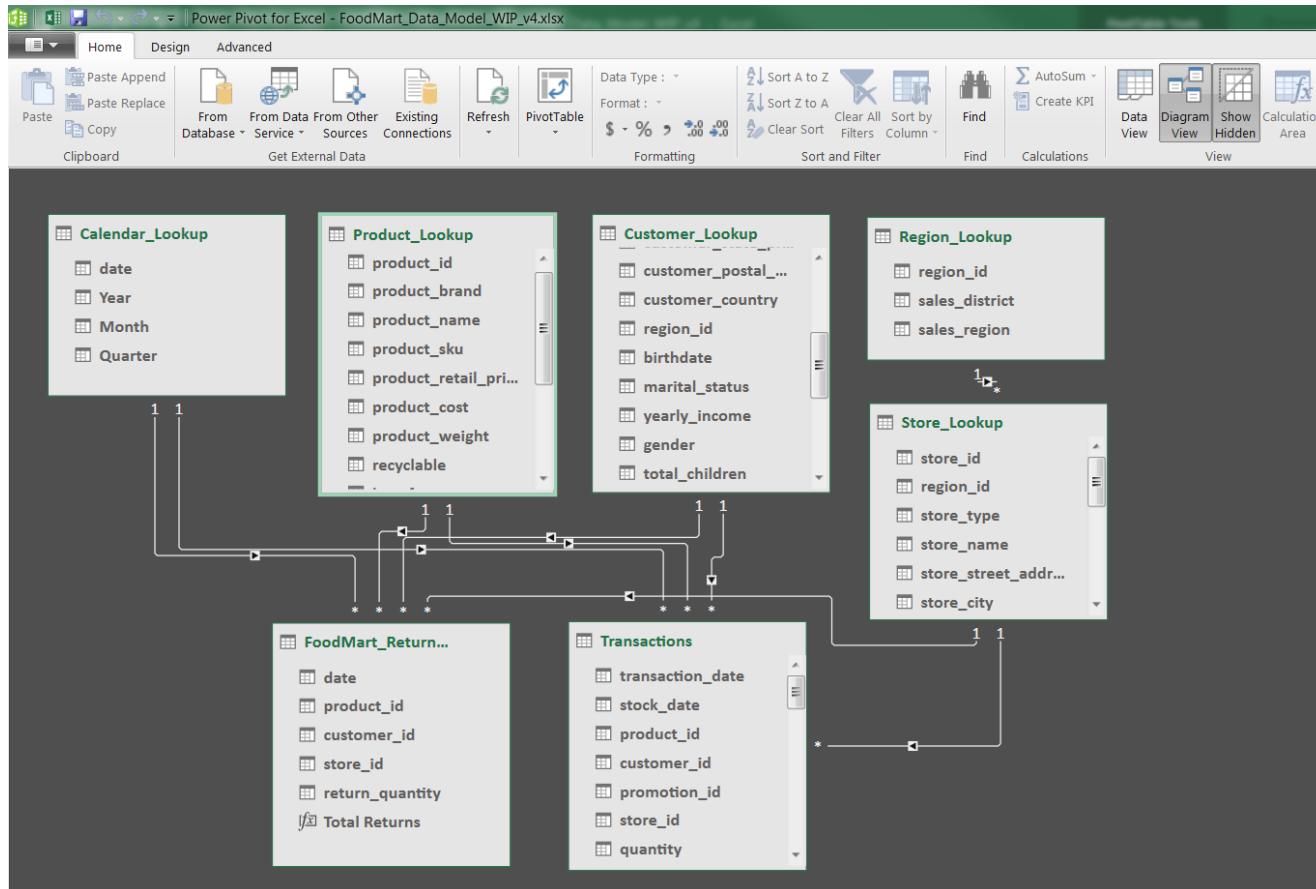
When was the last time you loaded
25,000,000 rows of data into Excel?

When you connect to data with **Power Query** and load it to Excel's **Data Model**, the data is compressed and stored in memory, NOT in worksheets (*no more 1,048,576 row limit!*)



[customer...]	customer_id	product_id	shop_id	quantity	Add Column
24476632	58983	3901	1	2	
24476633	58984	2852	1	2	
24476634	58985	384	1	2	
24476635	58985	502	1	2	
24476636	58985	1410	1	2	
24476637	58985	2080	1	2	
24476638	58987	313	1	2	
24476639	58987	1334	1	2	
24476640	58987	2019	1	2	
24476641	58987	2783	1	2	
24476642	58987	3108	1	2	
24476643	58987	3485	1	2	
24476644	58987	3544	1	2	
24476645	58988	189	1	2	
24476646	58988	300	1	2	
24476647	58988	402	1	2	
24476648	58988	446	1	2	
24476649	58988	500	1	2	
24476650	58988	542	1	2	
24476651	58988	1163	1	2	
24476652	58988	1172	1	2	
24476653	58988	1649	1	2	
24476654	58988	1706	1	2	
24476655	58988	1732	1	2	

#2: BUILD DATA MODELS TO BLEND SOURCES



This is an example of a Data Model in “**Diagram View**”, which allows you to create connections between tables

Instead of manually stitching tables together with cell formulas, you create ***relationships*** to blend data based on common fields

#3: AUTOMATE YOUR DATA PROCESSING

Supermarket_Purchase_Data - Query Editor

Queries

customer_id product_id quantity

Low Value Product

Properties

Name: Supermarket_Purchase_Data

Applied Steps

- Source
- Applied Headers
- Changed Column Type
- Removed Columns
- Filtered Rows
- Added Conditional Column
- Renamed Columns
- Removed Blank Rows

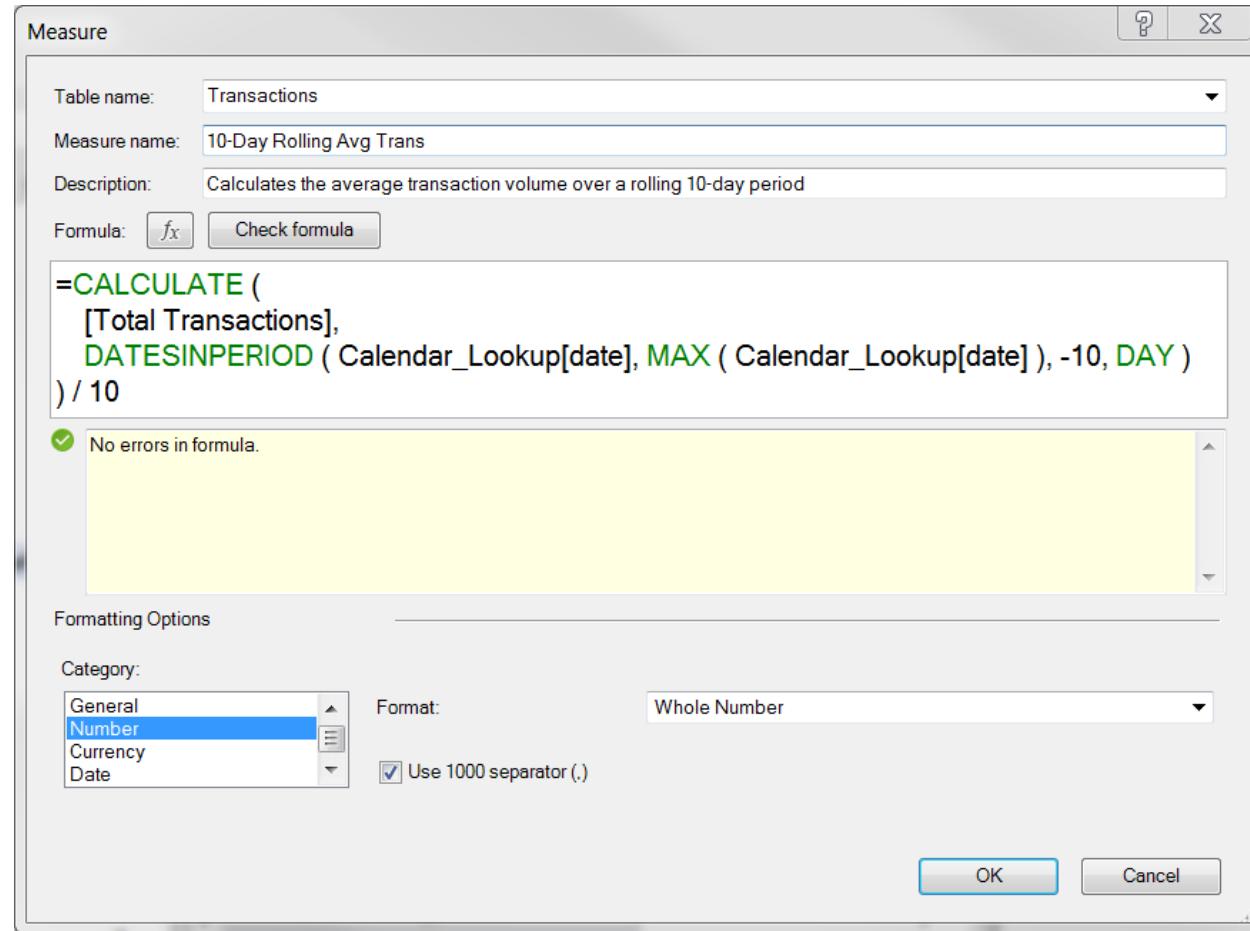
4 COLUMNS, 999+ ROWS

PREVIEW DOWNLOADED AT 5:59 PM

With Power Query, you can filter, shape and transform your raw data before loading it into the data model

Each step is automatically recorded and saved with the query, and applied whenever the source data is refreshed – like a macro!

#4: CREATE POWERFUL MEASURES WITH DAX



Measures are flexible and powerful calculations defined using **Data Analysis Expressions (DAX)**

In this case we're using a DAX time intelligence formula to calculate a **10-day rolling average**

WHEN TO USE POWER QUERY & POWER PIVOT

Use Power Query and Power Pivot when you want to...

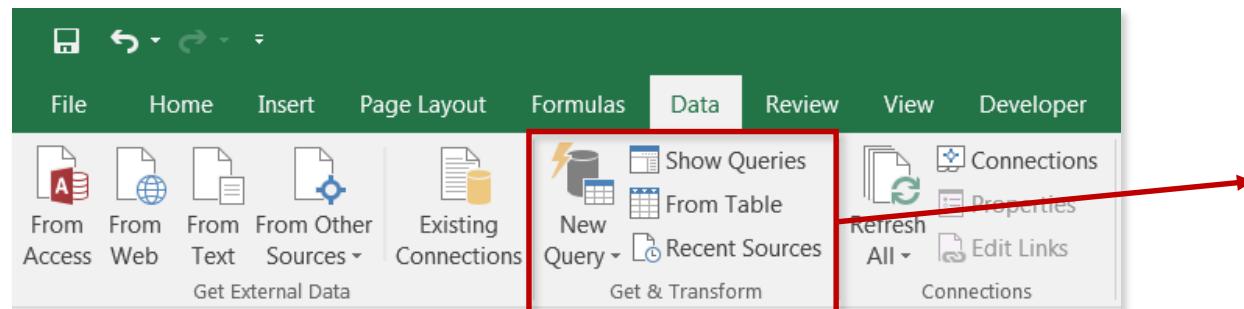
-  Analyze more data than can fit into a worksheet
-  Create connections to databases or external sources
-  Blend data across multiple large tables
-  Automate the process of loading and shaping your data
-  Unleash the **full business intelligence capabilities** of Excel

POWER QUERY

MEET POWER QUERY

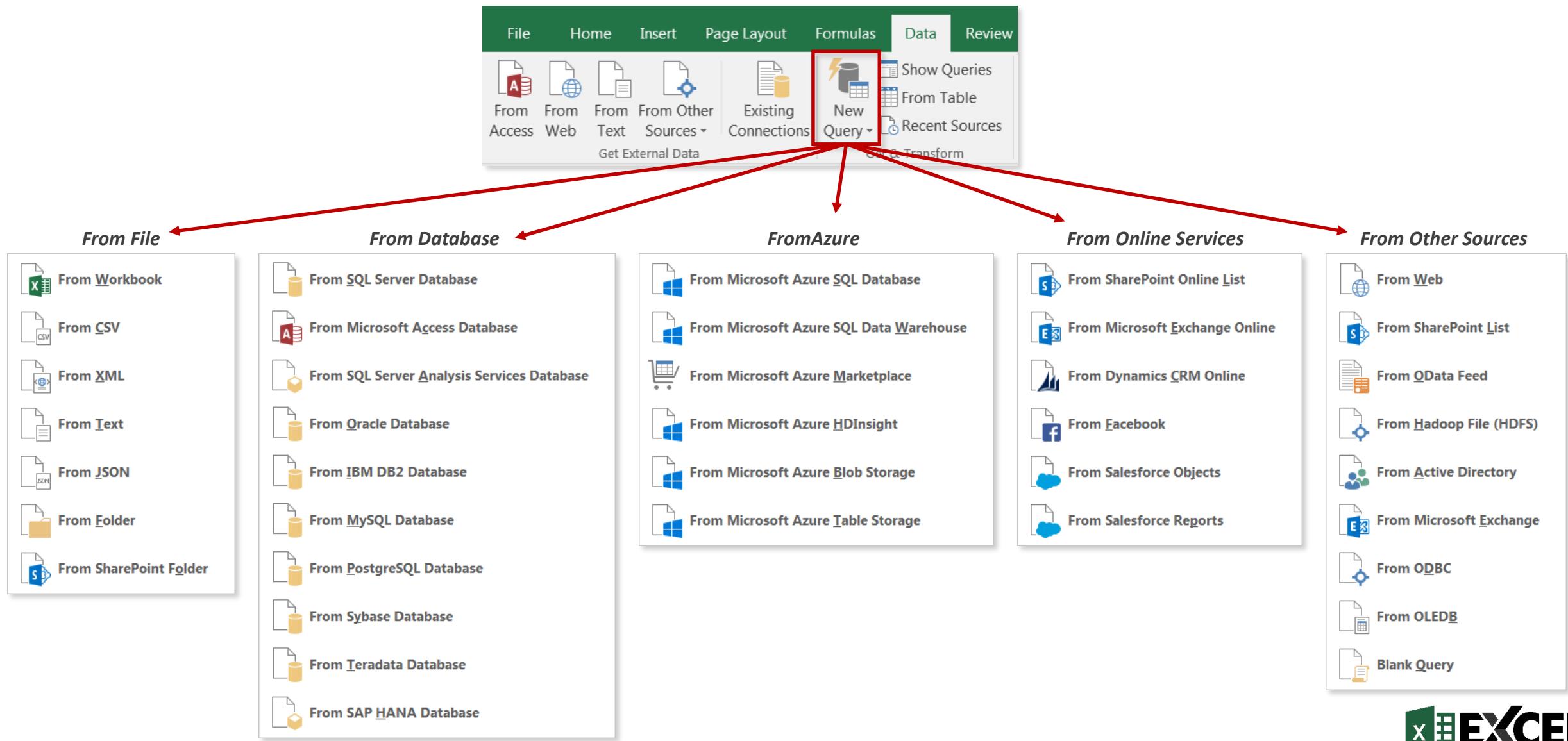
Power Query (aka “Get & Transform”) allows you to:

- Connect to data across a wide range of sources
- Filter, shape, append and transform raw data for further analysis and modeling
- Create stored procedures to automate your data prep (*like a macro!*)



The Power Query tools live in the **Data** tab, under the “**Get & Transform**” section (Excel 2016)

TYPES OF DATA CONNECTIONS



THE QUERY EDITOR

**Query
Editing
Tools**

The screenshot shows the Microsoft Power Query Editor interface. The main area displays a table of data with columns: date, product_id, customer_id, promotion_id, store_id, and quantity. The 'date' column contains dates from 1/1/1997. The 'product_id', 'customer_id', 'promotion_id', and 'store_id' columns contain numerical values. The 'quantity' column contains values like 6, 3, 4, etc. Red arrows point to various parts of the interface:

- A red box highlights the top ribbon bar, labeled "Query Editing Tools".
- A red box highlights the "Formula Bar" (top right) containing M code: `= Table.TransformColumnTypes(#"Promoted Headers",{{"date", type date}, {"product_id",`
- A red box highlights the "Data Preview" pane on the left, showing the first 14 rows of the table.
- A red box highlights the "Name your table!" section in the "Query Settings" pane, where the table is named "FoodMart_Transactions_1997".
- A red box highlights the "Applied Steps" section in the "Query Settings" pane, showing the steps: "Source" (Promoted Headers), "Changed Type" (highlighted in green).

Formula Bar
(this is "M" code)

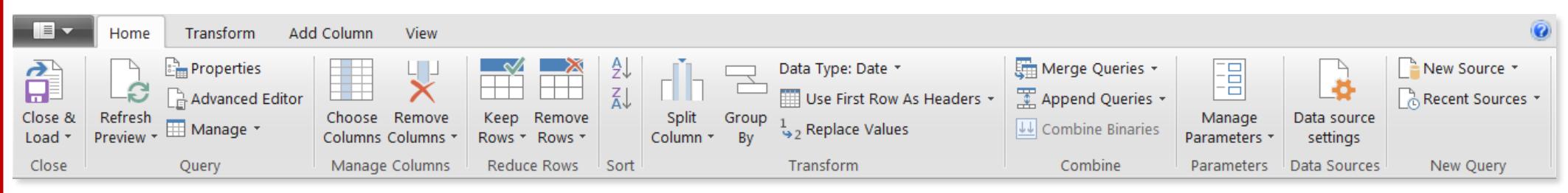
**Name your
table!**

**Applied
Steps**

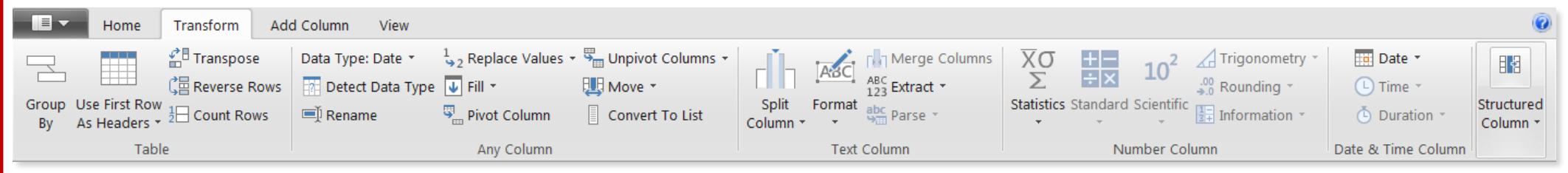
Access the **Query Editor** by creating a new query and choosing the “Edit” option, or by launching the Workbook Queries pane (**Data > Show Queries**) and right-clicking an existing query to edit

QUERY EDITOR TOOLS

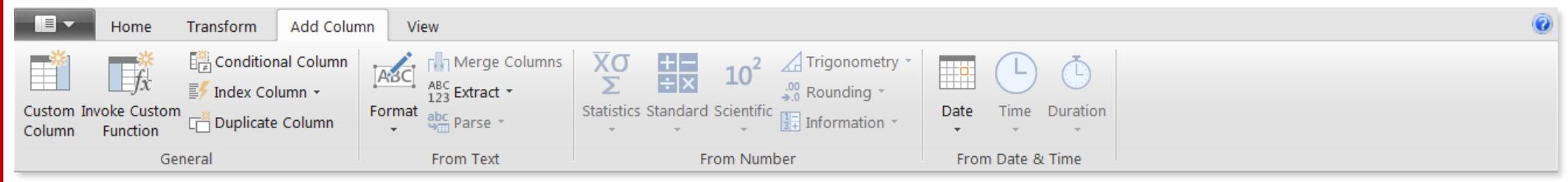
The **HOME** tab includes **general settings** and **common table transformation tools**



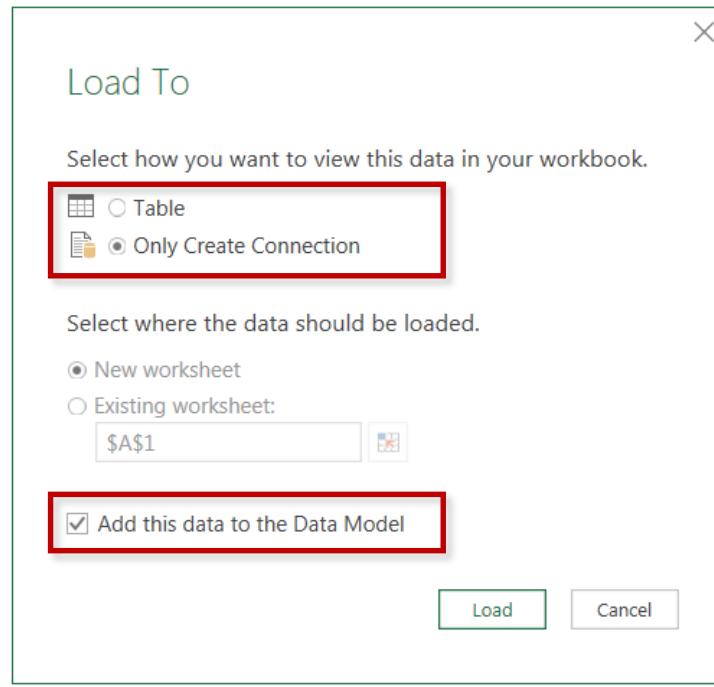
The **TRANSFORM** tab includes tools to **modify existing columns** (splitting/grouping, transposing, extracting text, etc.)



The **ADD COLUMN** tools **create new columns** based on conditional rules, text operations, calculations, dates, etc.



DATA LOADING OPTIONS



When you load data from Power Query, you have several options:

- **Table**
 - *Stores the data in a new or existing worksheet*
 - *Requires relatively small data sets (<1mm rows)*
- **Connection Only**
 - *Saves the data connection settings and applied steps*
 - *Data does not load to a worksheet*
- **Add to Data Model**
 - *Compresses and loads data to Excel's Data Model*
 - *Makes data accessible to Power Pivot for further analysis*

BASIC TABLE TRANSFORMATIONS

The screenshot shows the Power BI ribbon with the 'Transform' tab selected. The ribbon includes icons for Home, Refresh Preview, Close & Load, and Close. Under the Transform tab, there are sections for Add Column, View, Properties, Advanced Editor, Manage, and Query. The 'Manage Columns' section is highlighted with a red box. Below it, a dropdown menu shows options like Remove Columns and Remove Other Columns. The 'Sort values' section is also highlighted with a red box, showing icons for Keep Rows, Remove Rows, and Sort. A dropdown menu below it lists Remove Top Rows, Remove Bottom Rows, Remove Alternate Rows, Remove Duplicates, Remove Blank Rows, and Remove Errors. The 'Change data types' section is highlighted with a red box, showing a dropdown for Data Type (Date) and another for Use First Row As Headers. The 'Promote header row' section is highlighted with a red box, showing a dropdown menu with various options.

Sort values
(A-Z, Low-High, etc.)

Change data types
(date, \$, %, text, etc.)

Promote header row

Keep or remove columns

Tip: use the “Remove Other Columns” option if you always want a specific set

Keep or remove rows

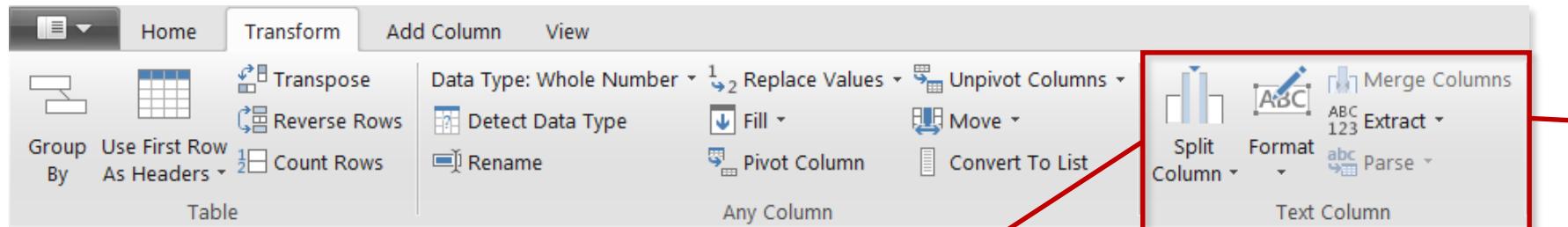
Tip: use the “Remove Duplicates” option to create a new lookup table from scratch

Duplicate, move & rename columns

Tip: Right-click the column header to access common tools

transaction_date	1/1/1997	12/29/1996
1	1/1/1997	12/29/1996
2	1/1/1997	12/27/1996
3	1/1/1997	12/31/1996
4	1/1/1997	12/26/1996
5		
6		
7		
8		
9		
10		
11		
12		
13		
14		
15		
16		
17		
18		
19		
20	1/1/1997	12/29/1996
21	1/1/1997	12/27/1996
22	1/1/1997	12/31/1996
23	1/1/1997	12/26/1996

TEXT-SPECIFIC TOOLS



Split a text column based on either a specific delimiter or a number of characters

By Delimiter
By Number of Characters

Length
First Characters
Last Characters
Range

Extract characters from a text column using a fixed length, first or last, or a defined range

Tip: Select two or more columns to merge or concatenate fields

lowercase
UPPERCASE
Capitalize Each Word
Trim
Clean
Add Prefix
Add Suffix

Format a text column to upper, lower or proper case, or add a prefix or suffix

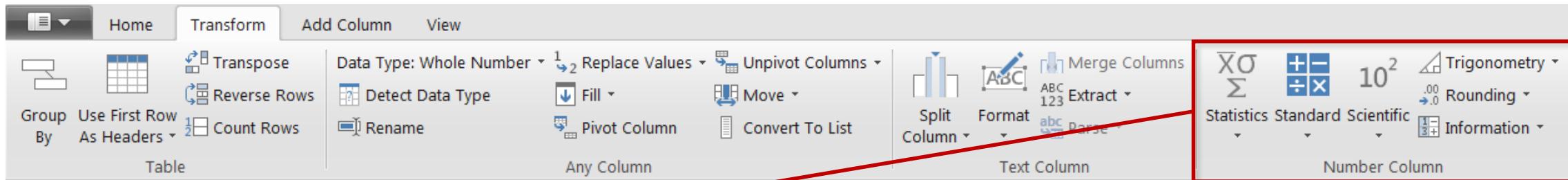
Tip: Use "Trim" to eliminate leading & trailing spaces, or "Clean" to remove non-printable characters



HEY THIS IS IMPORTANT!

You can access many of these tools in both the “Transform” and “Add Column” menus -- the difference is whether you want to **add a new column** or **modify an existing one**

NUMBER-SPECIFIC TOOLS



- Sum
- Minimum
- Maximum
- Median
- Average
- Standard Deviation
- Count Values
- Count Distinct Values

Statistics functions allow you to evaluate basic stats for the selected column (sum, min/max, average, count, countdistinct, etc)

Note: These tools return a **SINGLE** value, and are commonly used to explore a table rather than prepare it for loading

- Add
- Multiply
- Subtract
- Divide
- Integer-Divide
- Modulo
- Percentage
- Percent Of

- Absolute Value
- Power
- Square Root
- Exponent
- Logarithm
- Factorial

- Sine
- Cosine
- Tangent
- Arcsine
- Arccosine
- Arctangent

Standard

Scientific

Trigonometry

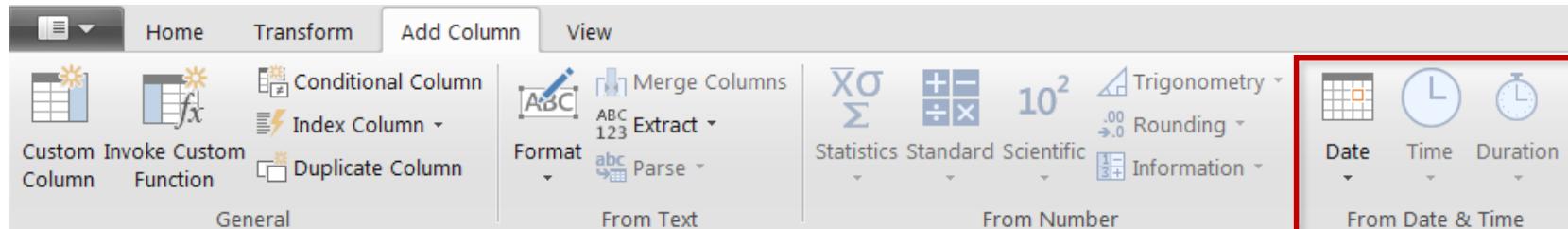
Standard, Scientific and **Trigonometry** tools allow you to apply standard operations (addition, multiplication, division, etc.) or more advanced calculations (power, logarithm, sine, tangent, etc) to each value in a column

Note: Unlike the Statistics options, these tools are applied to each individual row in the table

Information tools allow you to define binary flags (**TRUE/FALSE** or **1/0**) to mark each row in a column as even, odd, positive or negative

- Is Even
- Is Odd
- Sign

DATE-SPECIFIC TOOLS



The screenshot shows the Power BI ribbon with the "Transform" tab selected. The "Date & Time" tools group is highlighted with a red box. A red arrow points from this group to a detailed dropdown menu on the right.

Date & Time tools are relatively straight-forward, and include the following options:

- **Age:** Difference between the current time and the date in each row
- **Date Only:** Removes the time component of a date/time field
- **Year/Month/Quarter/Week/Day:** Extracts individual components from a date field
(Time-specific options include Hour, Minute, Second, etc.)
- **Earliest/Latest:** Evaluates the earliest or latest date from a column as a single value (can only be accessed from the “Transform” menu)

Note: You will almost always want to perform these operations from the “Add Column” menu to build out new fields, rather than transforming an individual date/time column

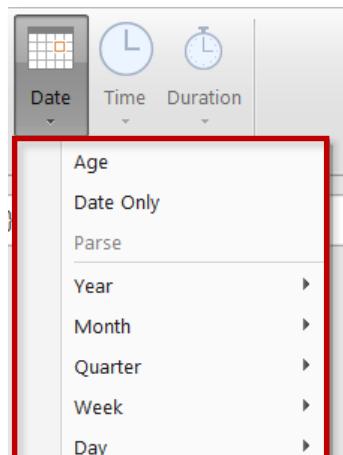


PRO TIP:

Load up a table containing a **single date column** and use Date tools to build out an **entire calendar table**

CREATING A BASIC CALENDAR TABLE

	date
1	1/1/1997
2	1/2/1997
3	1/3/1997
4	1/4/1997
5	1/5/1997
6	1/6/1997
7	1/7/1997
8	1/8/1997
9	1/9/1997
10	1/10/1997
11	1/11/1997
12	1/12/1997
13	1/13/1997
14	1/14/1997
15	1/15/1997
16	1/16/1997
17	1/17/1997
18	1/18/1997
19	1/19/1997
20	1/20/1997
21	1/21/1997
22	1/22/1997
23	1/23/1997



Use pre-defined **Date** options in the “**Add Column**” menu to quickly build out a calendar table from a list of dates

	date	Year	Month	Quarter	WeekOfYear	Day Name
1	1/1/1997	1997	1	1	1	Wednesday
2	1/2/1997	1997	1	1	1	Thursday
3	1/3/1997	1997	1	1	1	Friday
4	1/4/1997	1997	1	1	1	Saturday
5	1/5/1997	1997	1	1	2	Sunday
6	1/6/1997	1997	1	1	2	Monday
7	1/7/1997	1997	1	1	2	Tuesday
8	1/8/1997	1997	1	1	2	Wednesday
9	1/9/1997	1997	1	1	2	Thursday
10	1/10/1997	1997	1	1	2	Friday
11	1/11/1997	1997	1	1	2	Saturday
12	1/12/1997	1997	1	1	3	Sunday
13	1/13/1997	1997	1	1	3	Monday
14	1/14/1997	1997	1	1	3	Tuesday
15	1/15/1997	1997	1	1	3	Wednesday
16	1/16/1997	1997	1	1	3	Thursday
17	1/17/1997	1997	1	1	3	Friday
18	1/18/1997	1997	1	1	3	Saturday
19	1/19/1997	1997	1	1	4	Sunday
20	1/20/1997	1997	1	1	4	Monday
21	1/21/1997	1997	1	1	4	Tuesday
22	1/22/1997	1997	1	1	4	Wednesday
23	1/23/1997	1997	1	1	4	Thursday

PRO TIP: CREATING A ROLLING CALENDAR

- 1) Create a new, blank query (**Data > New Query > From Other Sources > Blank Query**)
- 2) In the formula bar, generate a starting date by entering a “literal” (1/1/2013 shown below):

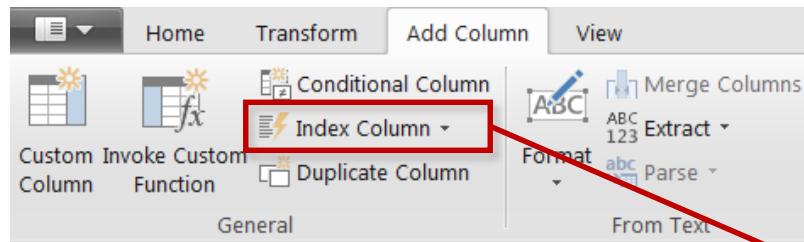


- 3) Click the fx icon to add a new custom step, and enter the following formula *exactly* as shown:

A screenshot of a software's formula editor. At the top, there are icons for clear, checkmark, and fx. The fx icon is highlighted with a red box. Below the editor is a preview area showing a list of four dates: 1/1/2013, 1/2/2013, 1/3/2013, and 1/4/2013. The formula bar contains the formula = List.Dates(Source, Number.From(DateTime.LocalNow())- Number.From(Source) ,#duration(1,0,0,0)). This formula uses the List.Dates function to generate a list of dates from the current date minus the date from the source column, for a duration of one day.

- 4) Convert the resulting list into a Table (**List Tools > To Table**) and format the column as a Date
- 5) Add calculated Date columns (Year, Month, Week, etc.) as necessary using the **Add Column** tools

ADDING AN INDEX COLUMN

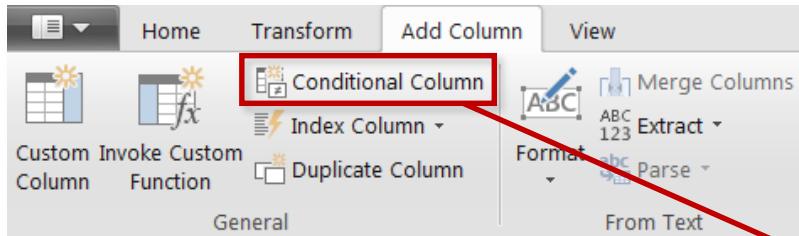


Index Columns contain a list of sequential values that can be used to identify each unique row in a table (*typically starting from 0 or 1*)

These columns are often used to create **unique IDs** that can be used to form relationships between tables (*more on that later!*)

	Index	transaction_date	stock_date	product_id	customer_id	promotion_id	sl
1	0	1/1/1997	12/28/1996	761	6613	0	
2	1	1/1/1997	12/30/1996	1435	8830	0	
3	2	1/1/1997	12/29/1996	1175	8830	0	
4	3	1/1/1997	12/30/1996	1152	8830	0	
5	4	1/1/1997	12/31/1996	1245	5005	0	
6	5	1/1/1997	12/27/1996	209	5005	0	
7	6	1/1/1997	12/28/1996	1345	5005	0	
8	7	1/1/1997	12/28/1996	1468	5005	0	
9	8	1/1/1997	12/26/1996	84	7962	0	
10	9	1/1/1997	12/30/1996	966	7962	0	
11	10	1/1/1997	12/27/1996	1022	7962	0	
12	11	1/1/1997	12/29/1996	440	7962	0	
13	12	1/4/1997	12/28/1996	151	2274	1054	
14	13	1/4/1997	12/28/1996	1287	8648	1054	
15	14	1/4/1997	12/30/1996	1264	8648	1054	
16	15	1/4/1997	12/31/1996	188	8648	1054	
17	16	1/4/1997	1/1/1997	1526	8648	1054	
18	17	1/4/1997	12/29/1996	518	8762	1054	
19	18	1/5/1997	12/31/1996	963	4018	0	
20	19	1/5/1997	12/29/1996	154	1418	0	
21							

ADDING A CONDITIONAL COLUMN



Conditional Columns allow you to define new fields based on logical rules and conditions (*IF/THEN statements*)

In this case we're creating a new conditional column called "**Order Size**", which depends on the values in the "quantity" column, as follows:

- If quantity >5, Order Size = "Large"
- If quantity is from 2-5, Order Size = "Medium"
- If quantity =1, Order Size = "Small"
- Otherwise Order Size = "Other"

Add Conditional Column

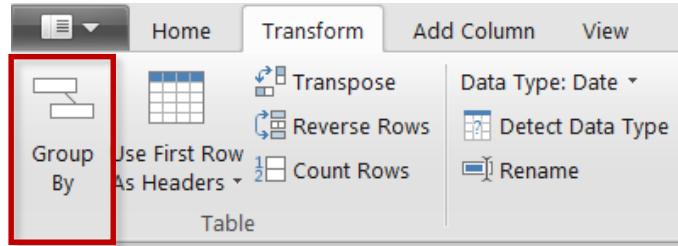
Add a conditional column that is computed from the other columns or values.

New column name

Column Name Operator Value Output

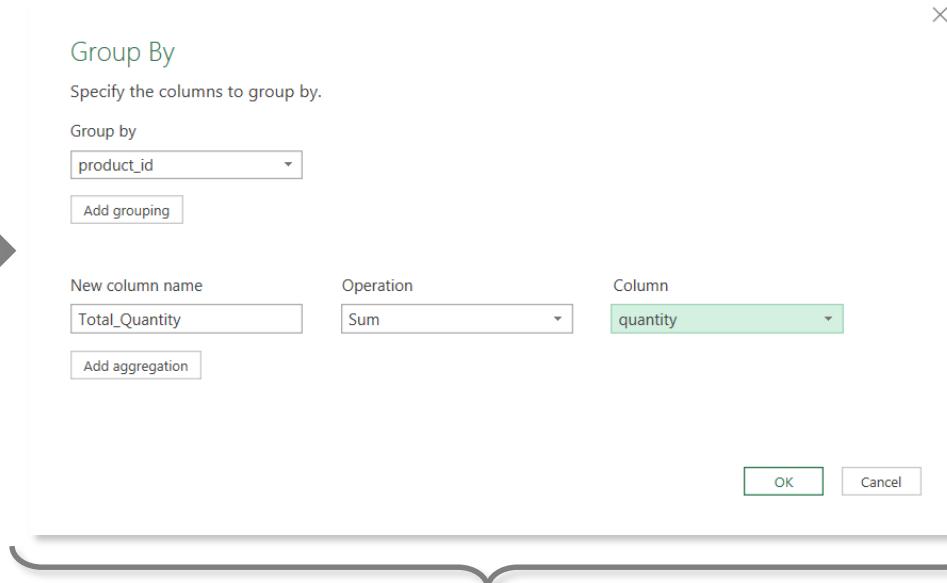
If quantity	is greater than	ABC 123	5	Then ABC 123	Large
Else If quantity	is greater than or...	ABC 123	2	Then ABC 123	Medium
Else If quantity	equals	ABC 123	1	Then ABC 123	Small
... <input type="button" value="Add Rule"/>					
Otherwise <input type="text" value="ABC 123"/> <input type="text" value="Other"/>					

GROUPING & AGGREGATING DATA



Group By allows you to aggregate your data at a different level
(i.e. transform daily data into monthly, roll up transaction-level data by store, etc.)

	transaction_date	stock_date	product_id	customer_id	store_id	quantity
1	8/12/1997	8/7/1997	1	3441	3	3
2	6/17/1997	6/12/1997	1	456	15	4
3	9/20/1997	9/15/1997	1	10140	17	3
4	1/3/1997	1/29/1996	1	4728	7	4
5	7/29/1997	7/24/1997	1	7704	3	2
6	11/28/1997	11/23/1997	1	2270	11	3
7	5/3/1997	4/28/1997	1	1312	3	3
8	9/19/1997	9/14/1997	1	9652	14	2
9	2/17/1997	2/12/1997	1	6666	17	3
10	11/11/1997	11/6/1997	1	3065	3	2
11	12/22/1997	12/17/1997	1	4707	11	3
12	8/16/1997	8/11/1997	1	6248	24	4
13	9/7/1997	9/2/1997	1	1565	24	3
14	12/20/1997	12/15/1997	1	157	24	3
15	6/12/1997	6/7/1997	1	5607	6	4
16	4/7/1997	4/2/1997	1	916	7	4
17	1/11/1997	1/6/1997	1	9788	13	3
18	12/27/1997	12/22/1997	1	8202	3	3
19	7/23/1997	7/18/1997	1	923	15	3
20	5/14/1997	5/9/1997	1	9169	23	4
21	10/6/1997	10/1/1997	1	3528	17	3
22	8/18/1997	8/13/1997	1	5929	15	5
23	4/18/1997	4/13/1997	1	4461	11	3



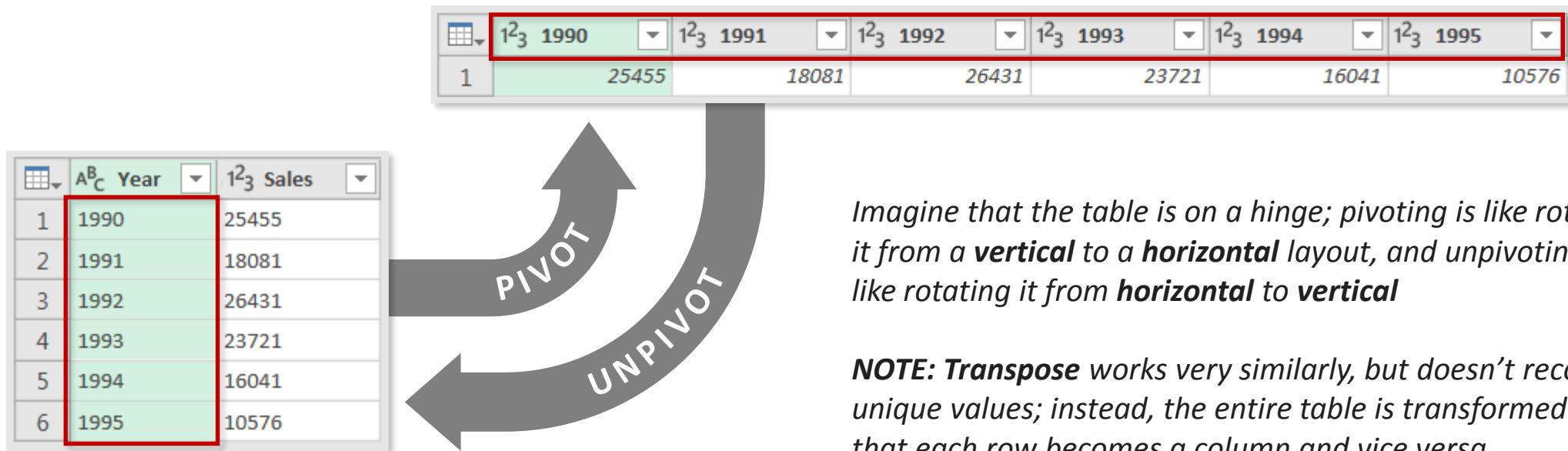
	product_id	Total_Quantity
1	4	304
2	11	322
3	12	372
4	14	325
5	16	319
6	23	368
7	46	355
8	50	313
9	56	318
10	59	336
11	61	314
12	75	321
13	89	321
14	90	323
15	112	357
16	115	356
17	119	329
18	120	325
19	126	352
20	127	353
21	130	384
22	139	332
23	159	394

In this case we're transforming a daily, transaction-level table into a summary of "quantity" by "product_id"

Note that we lose any field not specified in the Group By settings

PIVOTING & UNPIVOTING

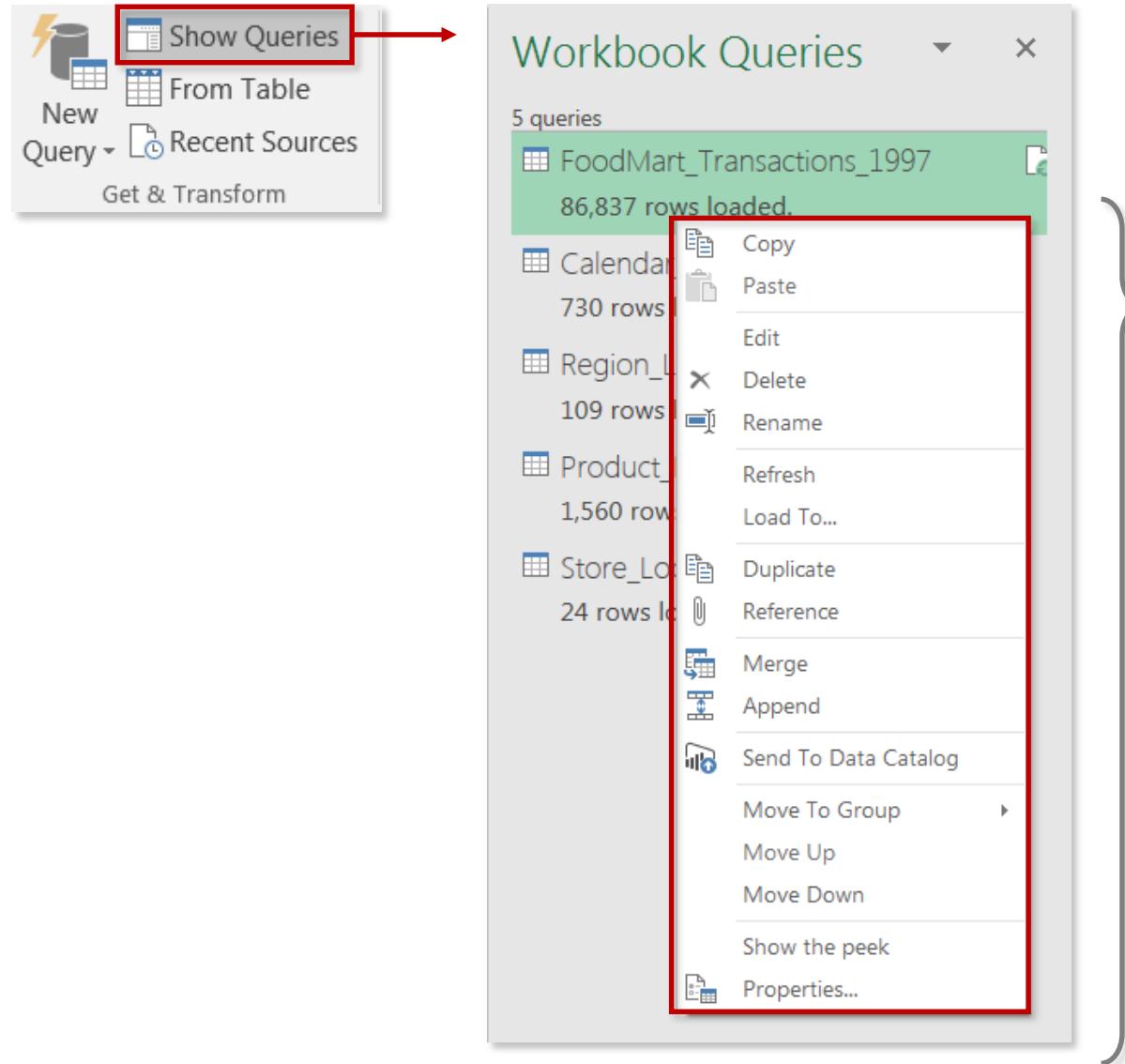
“Pivoting” is a fancy way to describe the process of turning **distinct row values** into **columns** (“*pivoting*”) or turning **columns** into **rows** (“*unpivoting*”)



Imagine that the table is on a hinge; pivoting is like rotating it from a **vertical** to a **horizontal** layout, and unpivoting is like rotating it from **horizontal** to **vertical**

NOTE: *Transpose* works very similarly, but doesn't recognize unique values; instead, the entire table is transformed so that each row becomes a column and vice versa

MODIFYING WORKBOOK QUERIES



Click on **Show Queries** to launch the **Workbook Queries** pane

Right-click any individual query to access common options and tools:

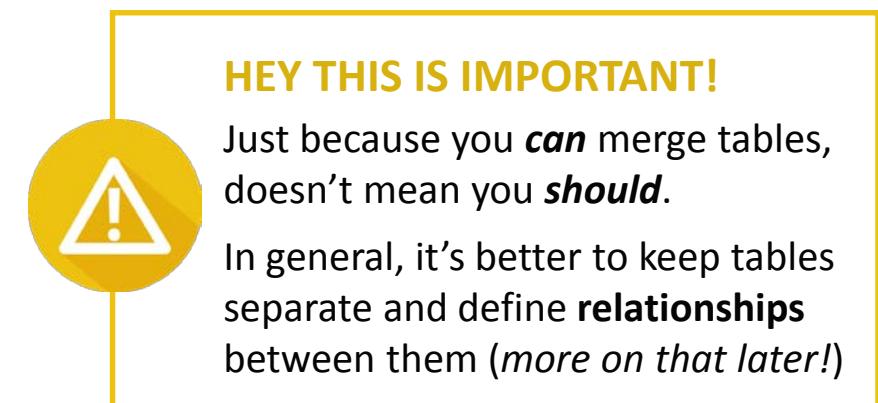
- **Edit** (launches the Query Editor)
- **Delete**
- **Rename**
- **Refresh**
- **Duplicate**
- **Merge**
- **Append**

MERGING QUERIES

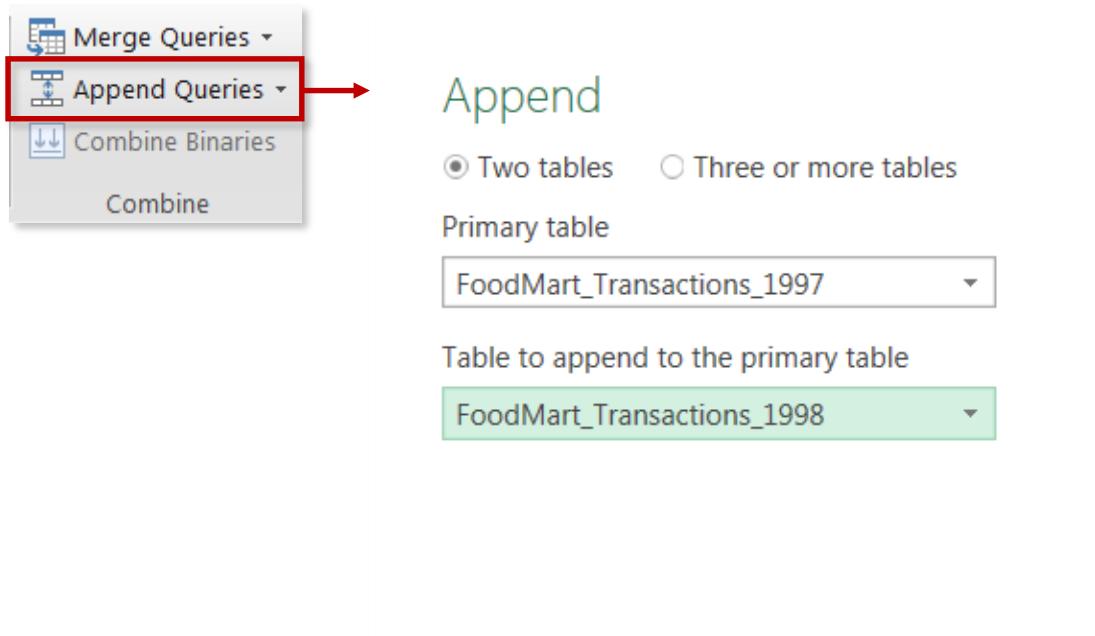
The screenshot shows the 'Merge' dialog box in Power Query. On the left, a vertical menu lists 'Merge Queries', 'Append Queries', 'Combine Binaries', and 'Combine'. An arrow points from 'Merge Queries' to the main area. The main area has two tables: 'FoodMart_Transactions_1997' at the top and 'Product_Lookup' below it. Both tables have their first rows selected. Below the tables, a 'Join Kind' dropdown is set to 'Left Outer (all from first, matching from second)'. A status message at the bottom says 'The selection has matched 86837 out of the first 86837 rows.' There are 'OK' and 'Cancel' buttons at the bottom right.

- Merging queries allows you to **join tables** based on a common column (like VLOOKUP)
- In this case we're merging the **FoodMart_Transactions_1997** table with the **Product_Lookup** table, which share a "*product_id*" column

TIP: Merging adds columns to an existing table



APPENDING QUERIES



The screenshot shows the Power Query ribbon with the 'Append Queries' option highlighted by a red box and an arrow pointing to the 'Append' dialog box. The 'Append' dialog box contains the following settings:

- Append**: The primary table is set to 'FoodMart_Transactions_1997'.
- Table to append to the primary table**: The target table is set to 'FoodMart_Transactions_1998'.
- Two tables** is selected as the number of tables to append.

- Appending queries allows you to **combine** (or **stack**) tables that share a common structure and set of columns
- In this case we're appending the **FoodMart_Transactions_1998** table to the **FoodMart_Transactions_1997** table, since they contain the same set of columns and data types

TIP: Appending **adds rows** to an existing table



PRO TIP:

Use the “**From Folder**” query option to automatically append all files from within the same folder

POWER QUERY BEST PRACTICES



Give your queries clear and intuitive names, *before* loading the data

- *Define names immediately; updating query & table names later can be a headache, especially if you've already referenced them in calculated measures*
- *Don't use spaces in table names (otherwise you have surround them with single quotes)*



Do as much shaping as possible at the source of the data

- *Shaping data at the source (i.e. SQL, Access) minimizes the need for complex procedures in Power Query, and allows you to create new models without replicating the same process*



When working with large tables, only load the data you need

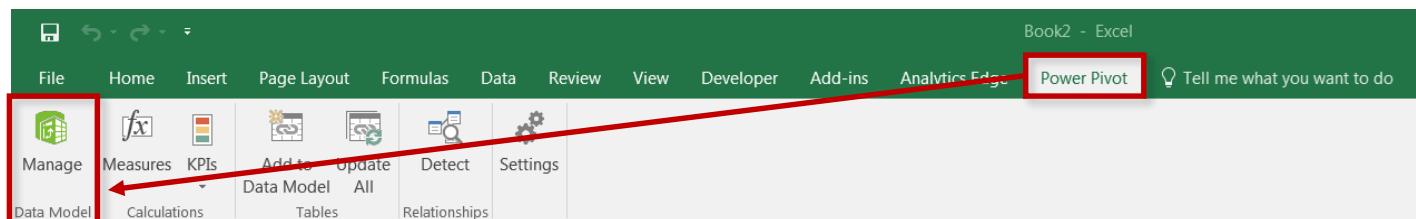
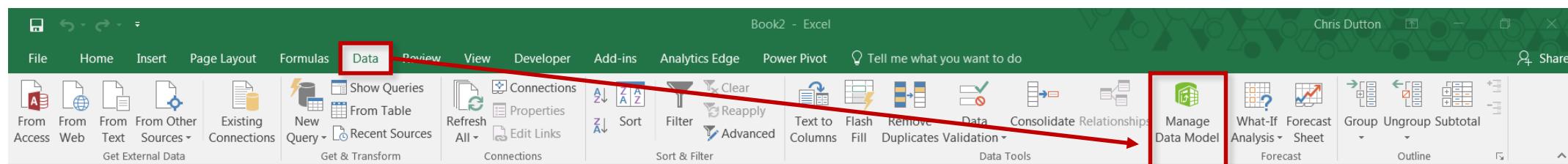
- *Don't include hourly data when you only need daily, or product-level transactions when you only care about store-level performance; extra data will only slow you down*

DATA MODELING 101

MEET EXCEL'S DATA MODEL

The **Data Model** provides simple and intuitive tools for building relational databases directly in Excel. With the data model you can:

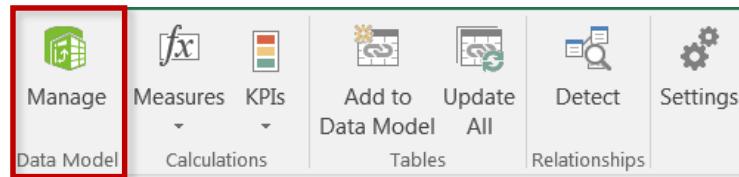
- Manage massive datasets that can't fit into worksheets
- Create table relationships to blend data across multiple sources
- Define custom hierarchies and perspectives



Access the **Data Model** through the **Power Pivot** tab or the **Data** tab

(*Note: you may need to enable the Power Pivot tab via File > Options > Add-Ins > Manage COM Add-Ins*)

THE DATA MODEL WINDOW



The **Data Model** opens in a separate Excel window, where you can view your data tables, calculate new measures, and define table relationships

Note: Closing the Data Model window does *NOT* close your Excel workbook

A screenshot of the Power Pivot Data Model window titled 'Power Pivot for Excel - FoodMart_Data_Model_WIP.xlsx'. The window shows a table of transaction data with columns: date, product_id, customer_id, promotion_id, store_id, quantity. The first 10 rows of data are displayed. The bottom of the window shows tabs for 'FoodMart Transactions', 'Store_Lookup', 'Product_Lookup', 'Customer_Lookup', 'Promotion_Lookup', and 'Calendar_Lookup'. A status bar at the bottom indicates 'Record: 1 of 86,837'.

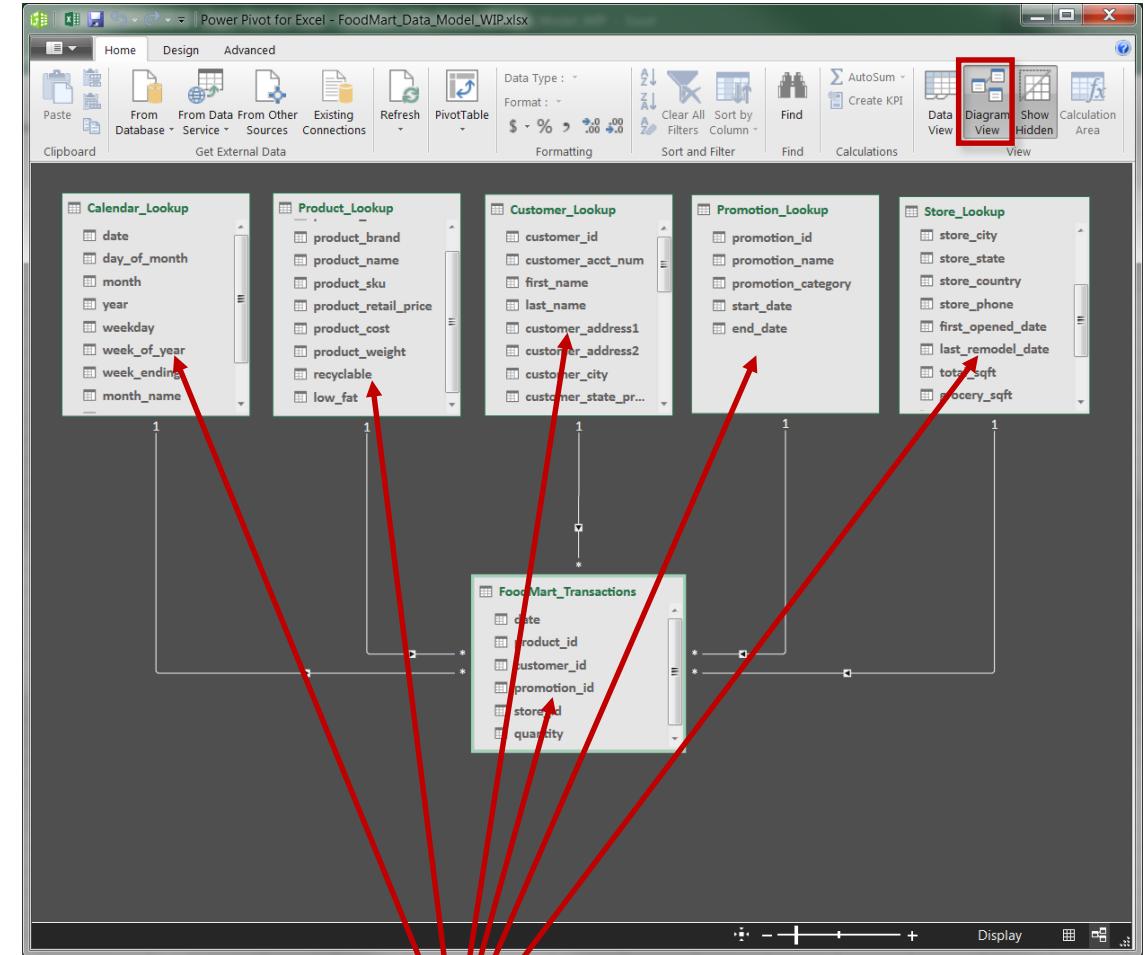
DATA VIEW VS. DIAGRAM VIEW

DATA VIEW

A screenshot of the Power Pivot Data View interface. The main area displays a table of transaction data with columns: date, product_id, customer_id, promotion_id, store_id, and quantity. The 'date' column shows dates from July 27, 1997, to August 2, 1997. The 'quantity' column contains values like 0, 13, and 3. Below the table, tabs for 'FoodMart_Transactions', 'Store_Lookup', 'Product_Lookup', 'Customer_Lookup', 'Promotion_Lookup', and 'Calendar_Lookup' are visible. The ribbon at the top includes tabs for Home, Design, Advanced, and various data-related buttons. The 'Data View' button in the ribbon is highlighted with a red box.

Tables organized in tabs

DIAGRAM VIEW



Tables organized as objects

DATABASE NORMALIZATION

Normalization is the process of organizing the tables and columns in a relational database to reduce redundancy and preserve data integrity. It is commonly used to:

- **Eliminate redundant data** to decrease table sizes and improve processing speed & efficiency
- **Minimize errors and anomalies** from data modifications (inserting, updating or deleting records)
- **Simplify queries** and structure the database for meaningful analysis

In a normalized database, each table should serve a ***distinct*** and ***specific*** purpose (*i.e. product information, calendar fields, transaction records, customer attributes, etc.*)

date	product_id	quantity	product_brand	product_name	product_sku	product_weight
1/1/1997	869	5	Nationeel	Nationeel Grape Fruit Roll	52382137179	17
1/7/1997	869	2	Nationeel	Nationeel Grape Fruit Roll	52382137179	17
1/3/1997	1	4	Washington	Washington Berry Juice	90748583674	8.39
1/1/1997	1472	3	Fort West	Fort West Fudge Cookies	37276054024	8.28
1/6/1997	1472	2	Fort West	Fort West Fudge Cookies	37276054024	8.28
1/5/1997	2	4	Washington	Washington Mango Drink	96516502499	7.42
1/1/1997	76	4	Red Spade	Red Spade Sliced Chicken	62054644227	18.1
1/1/1997	76	2	Red Spade	Red Spade Sliced Chicken	62054644227	18.1
1/5/1997	3	2	Washington	Washington Strawberry Drink	58427771925	13.1
1/7/1997	3	2	Washington	Washington Strawberry Drink	58427771925	13.1
1/1/1997	320	3	Excellent	Excellent Cranberry Juice	36570182442	16.4

When you **don't** normalize, you end up with tables like this; all of the duplicate product records could be eliminated with a lookup table based on **product_id**

This may not seem critical now, but minor inefficiencies can become major problems as databases scale in size

DATA TABLES VS. LOOKUP TABLES

Models generally contain two types of tables: **data** (or “*fact*”) tables, and **lookup** (or “*dimension*”) tables

- **Data tables** contain numbers or values, typically at the most granular level possible, with ID or “key” columns that can be used to connect to each lookup table
- **Lookup tables** provide descriptive, often text-based attributes about each dimension in a table

date	product_id	quantity
1/1/1997	869	5
1/1/1997	1472	3
1/1/1997	76	4
1/1/1997	320	3
1/1/1997	4	4
1/1/1997	952	4
1/1/1997	1222	4
1/1/1997	517	4
1/1/1997	1359	4
1/1/1997	357	4
1/1/1997	1426	5
1/1/1997	190	4
1/1/1997	367	4
1/1/1997	250	5
1/1/1997	600	4
1/1/1997	702	5

date	day_of_month	month	year	weekday	week_of_year	week_ending	month_name	quarter
1/1/1997	1	1	1997	Wednesday	1	1/5/1997	January	Q1
1/2/1997	2	1	1997	Thursday	1	1/5/1997	January	Q1
1/3/1997	3	1	1997	Friday	1	1/5/1997	January	Q1
1/4/1997	4	1	1997	Saturday	1	1/5/1997	January	Q1
1/5/1997	5	1	1997	Sunday	2	1/5/1997	January	Q1
1/6/1997	6	1	1997	Monday	2	1/12/1997	January	Q1

This **Calendar Lookup** table provides additional attributes about each **date** (month, year, weekday, quarter, etc.)

product_id	product_brand	product_name	product_sku	product_retail_price	product_cost	product_weight
1	Washington	Washington Berry Juice	90748583674	2.85	0.94	8.39
2	Washington	Washington Mango Drink	96516502499	0.74	0.26	7.42
3	Washington	Washington Strawberry Drink	58427771925	0.83	0.4	13.1
4	Washington	Washington Cream Soda	64412155747	3.64	1.64	10.6
5	Washington	Washington Diet Soda	85561191439	2.19	0.77	6.66
6	Washington	Washington Cola	29804642796	1.15	0.37	15.8
7	Washington	Washington Diet Cola	20191444754	2.61	0.91	18
8	Washington	Washington Orange Juice	89770532250	2.59	0.8	8.97

This **Product Lookup** table provides additional attributes about each **product** (brand, product name, sku, price, etc.)

This **Data Table** contains “*quantity*” values, and connects to lookup tables via the “*date*” and “*product_id*” columns

PRIMARY & FOREIGN KEYS

date	product_id	quantity
1/1/1997	869	5
1/1/1997	1472	3
1/1/1997	76	4
1/1/1997	320	3
1/1/1997	4	4
1/1/1997	952	4
1/1/1997	1222	4
1/1/1997	517	4
1/1/1997	1359	4
1/1/1997	357	4
1/1/1997	1426	5
1/1/1997	190	4
1/1/1997	367	4
1/1/1997	250	5
1/1/1997	600	4
1/1/1997	702	5

These columns are **foreign keys**; they contain *multiple* instances of each value, and are used to match the **primary keys** in related lookup tables

date	day_of_month	month	year	weekday	week_of_year	week_ending	month_name	quarter
1/1/1997	1	1	1997	Wednesday	1	1/5/1997	January	Q1
1/2/1997	2	1	1997	Thursday	1	1/5/1997	January	Q1
1/3/1997	3	1	1997	Friday	1	1/5/1997	January	Q1
1/4/1997	4	1	1997	Saturday	1	1/5/1997	January	Q1
1/5/1997	5	1	1997	Sunday	2	1/5/1997	January	Q1
1/6/1997	6	1	1997	Monday	2	1/12/1997	January	Q1

product_id	product_brand	product_name	product_sku	product_retail_price	product_cost	product_weight
1	Washington	Washington Berry Juice	90748583674	2.85	0.94	8.39
2	Washington	Washington Mango Drink	96516502499	0.74	0.26	7.42
3	Washington	Washington Strawberry Drink	58427771925	0.83	0.4	13.1
4	Washington	Washington Cream Soda	64412155747	3.64	1.64	10.6
5	Washington	Washington Diet Soda	85561191439	2.19	0.77	6.66
6	Washington	Washington Cola	29804642796	1.15	0.37	15.8
7	Washington	Washington Diet Cola	20191444754	2.61	0.91	18
8	Washington	Washington Orange Juice	89770532250	2.59	0.8	8.97

These columns are **primary keys**; they *uniquely* identify each row of a table, and match the **foreign keys** in related data tables

RELATIONSHIPS VS. MERGED TABLES



*Can't I just **merge queries** or use **LOOKUP** or **RELATED** functions to pull those attributes into the fact table itself, so that I have everything in one place??*

-Anonymous confused man

Original **Fact Table** fields

Attributes from **Calendar Lookup** table

Attributes from **Product Lookup** table

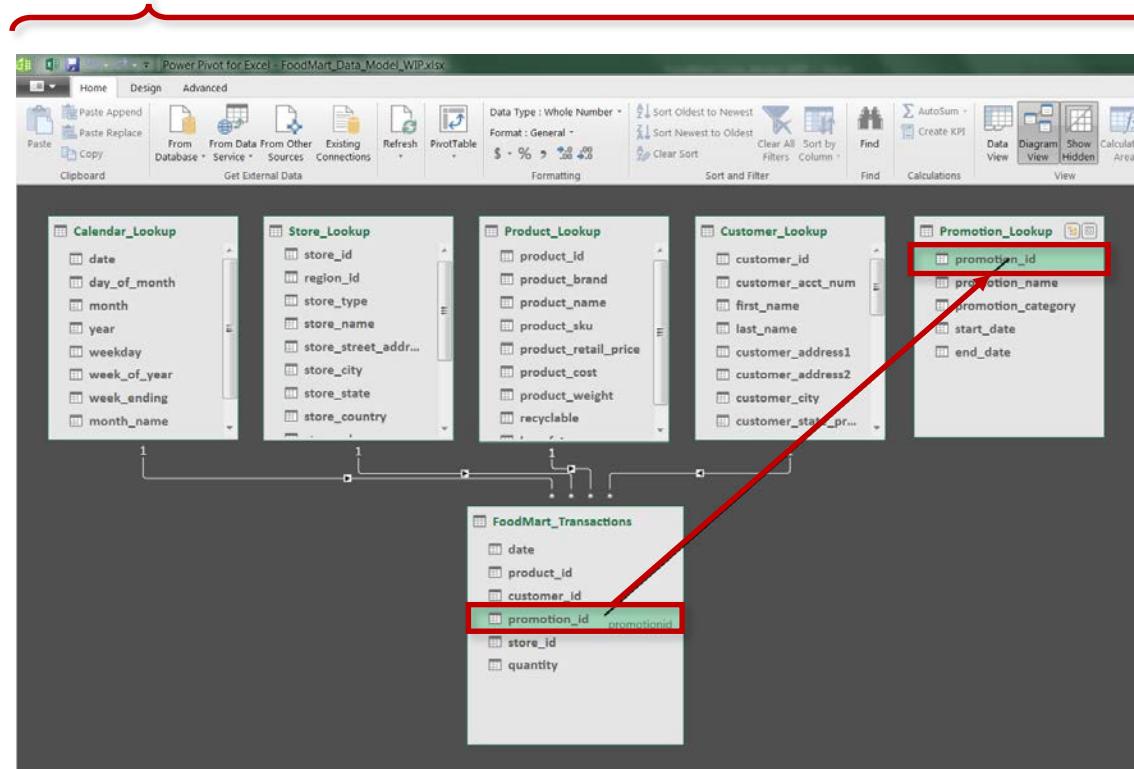
date	product_id	quantity	day_of_month	month	year	weekday	month_name	quarter	product_brand	product_name	product_sku	product_weight
1/1/1997	869	5	1	1	1997	Wednesday	January	Q1	Nationeel	Nationeel Grape Fruit Roll	52382137179	17
1/7/1997	869	2	7	1	1997	Tuesday	January	Q1	Nationeel	Nationeel Grape Fruit Roll	52382137179	17
1/3/1997	1	4	3	1	1997	Friday	January	Q1	Washington	Washington Berry Juice	90748583674	8.39
1/1/1997	1472	3	1	1	1997	Wednesday	January	Q1	Fort West	Fort West Fudge Cookies	37276054024	8.28
1/6/1997	1472	2	6	1	1997	Monday	January	Q1	Fort West	Fort West Fudge Cookies	37276054024	8.28
1/5/1997	2	4	5	1	1997	Sunday	January	Q1	Washington	Washington Mango Drink	96516502499	7.42
1/1/1997	76	4	1	1	1997	Wednesday	January	Q1	Red Spade	Red Spade Sliced Chicken	62054644227	18.1
1/1/1997	76	2	1	1	1997	Wednesday	January	Q1	Red Spade	Red Spade Sliced Chicken	62054644227	18.1
1/5/1997	3	2	5	1	1997	Sunday	January	Q1	Washington	Washington Strawberry Drink	58427771925	13.1
1/7/1997	3	2	7	1	1997	Tuesday	January	Q1	Washington	Washington Strawberry Drink	58427771925	13.1
1/1/1997	320	3	1	1	1997	Wednesday	January	Q1	Excellent	Excellent Cranberry Juice	36570182442	16.4

Sure, but it's extremely inefficient.

- Merging data in this way creates **redundant data** and utilizes **significantly more memory and processing power** than creating relationships between multiple small tables

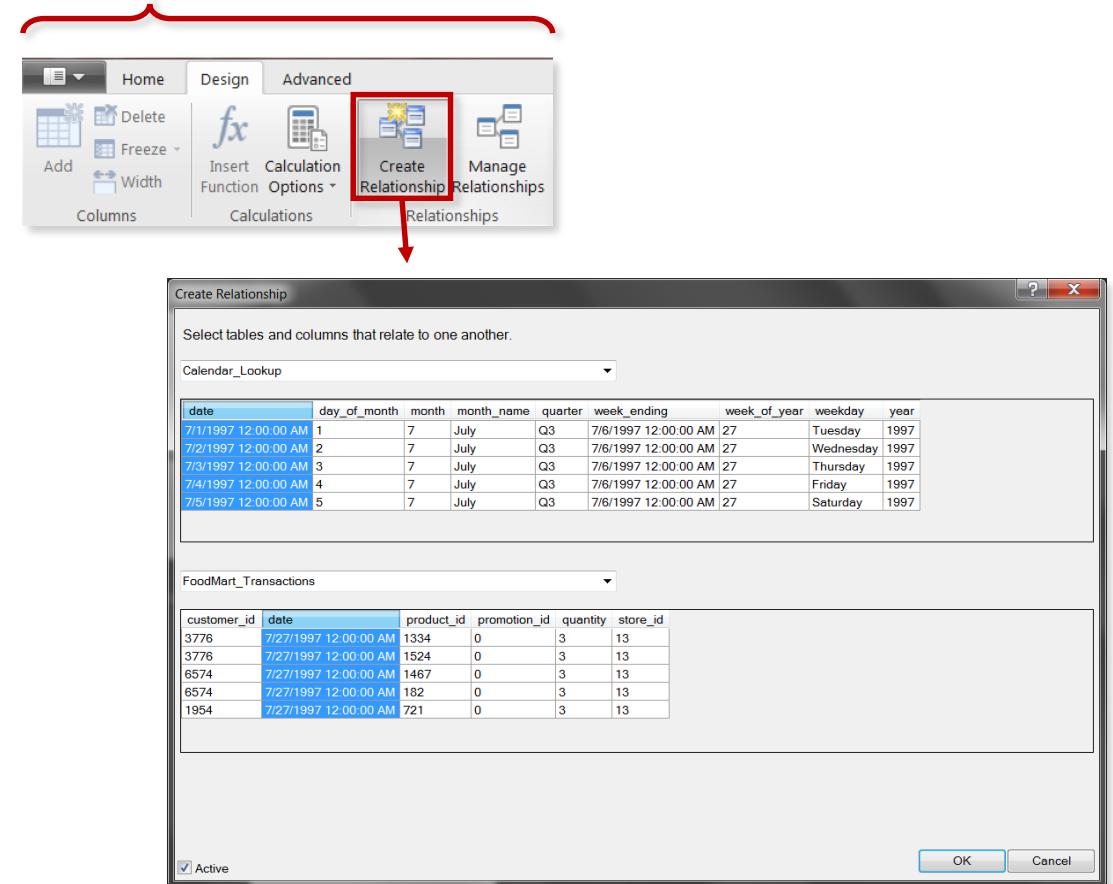
CREATING TABLE RELATIONSHIPS

Option 1: Click and drag relationships in Diagram View



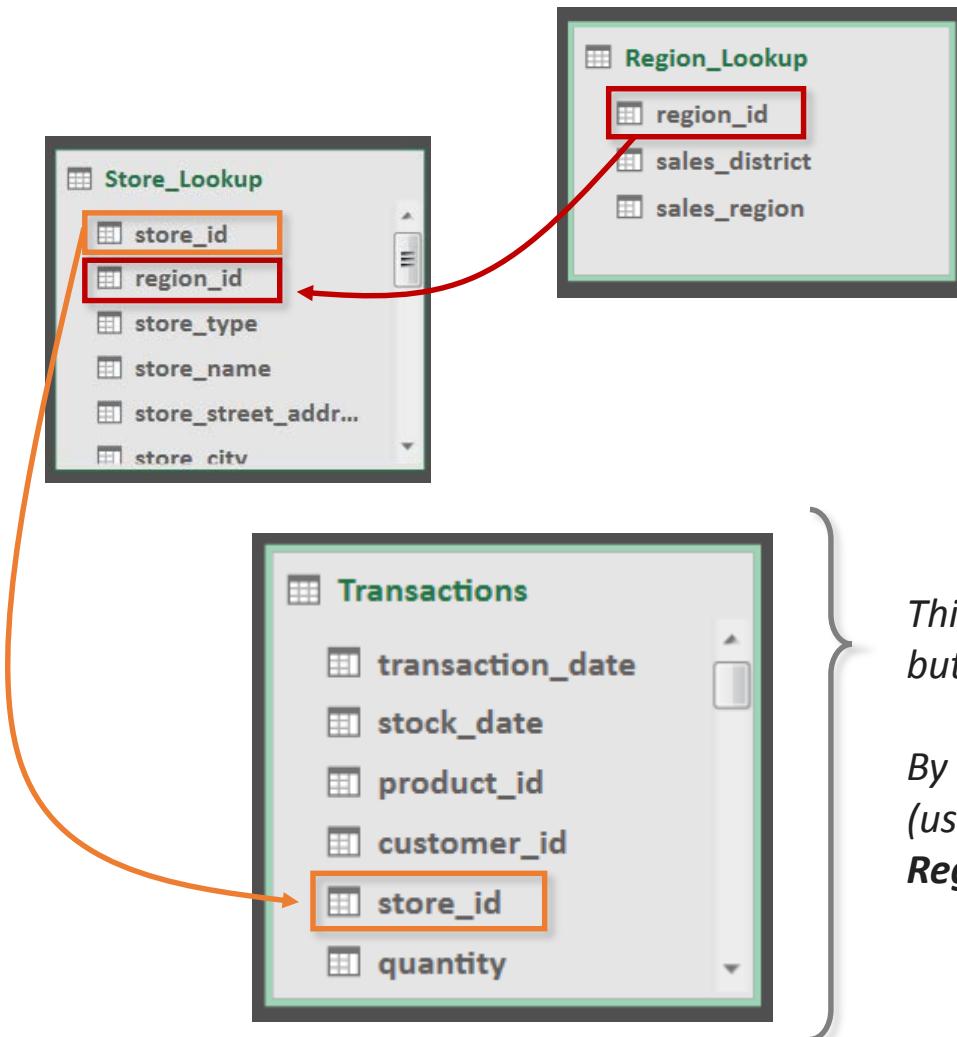
Tip: Always drag relationships from the **Data** table to the **Lookup** tables

Option 2: Use “Create Relationship” in the Design tab



*Note: In Excel 2010/2013 the diagram view looks a bit different, and arrows point in the opposite direction by default

CONNECTING LOOKUPS TO LOOKUPS



PRO TIP:

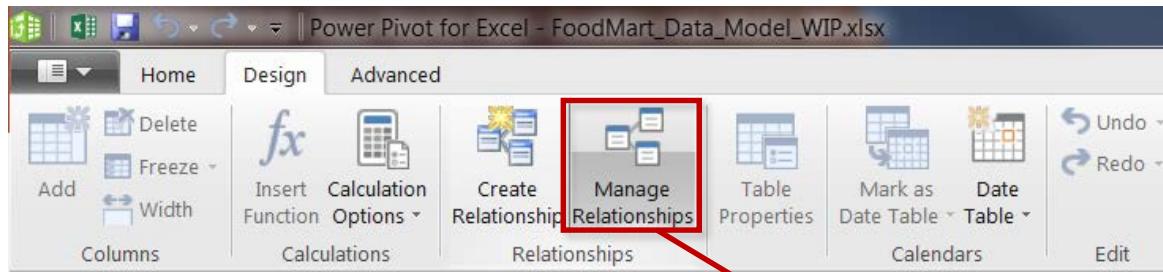
Models with multiple related lookup tables are called “**snowflake**” schemas

Models with a single table for each lookup or dimension are called “**star**” schemas

This **Transactions** data table can connect to **Store_Lookup** using **store_id**, but does not contain a **region_id** to connect to the **Region_Lookup** table

By creating a relationship between **Store_Lookup** and **Region_Lookup** (using **region_id**), we have essentially connected **Transactions** with **Region_Lookup**; filter context will now flow all the way down the chain

MODIFYING TABLE RELATIONSHIPS



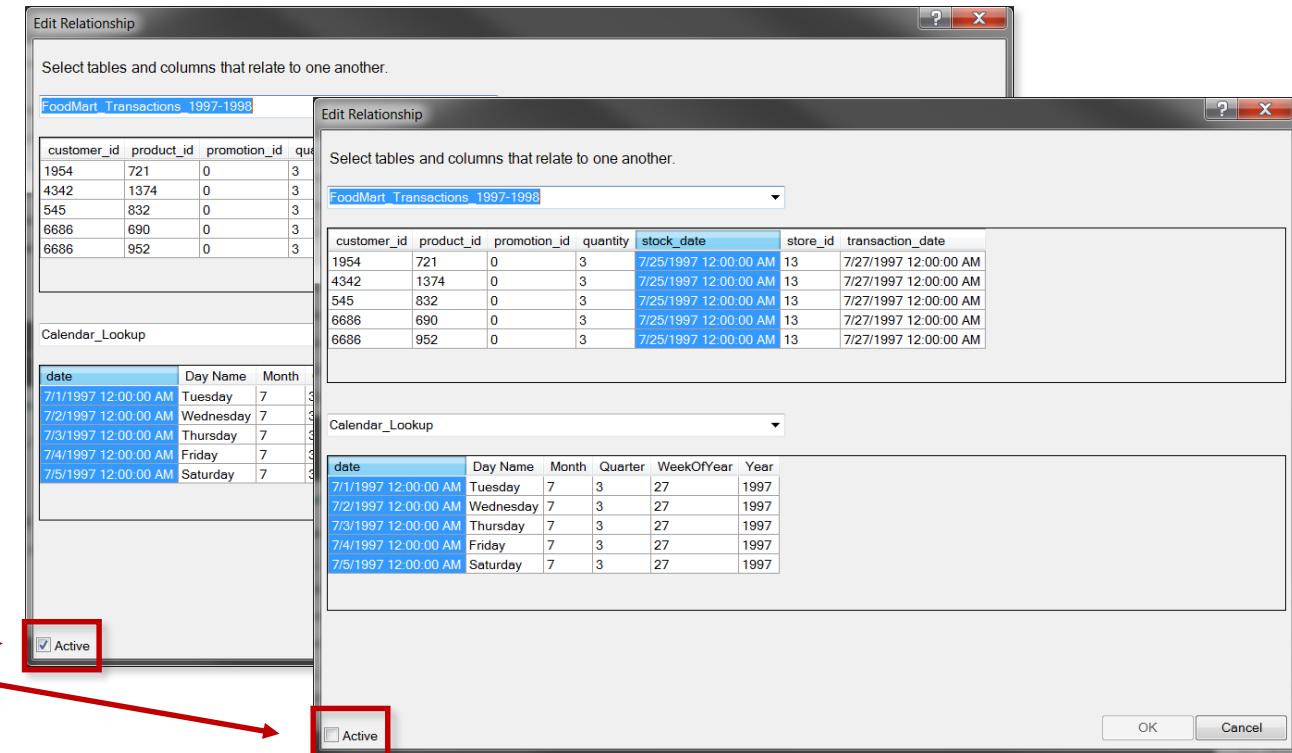
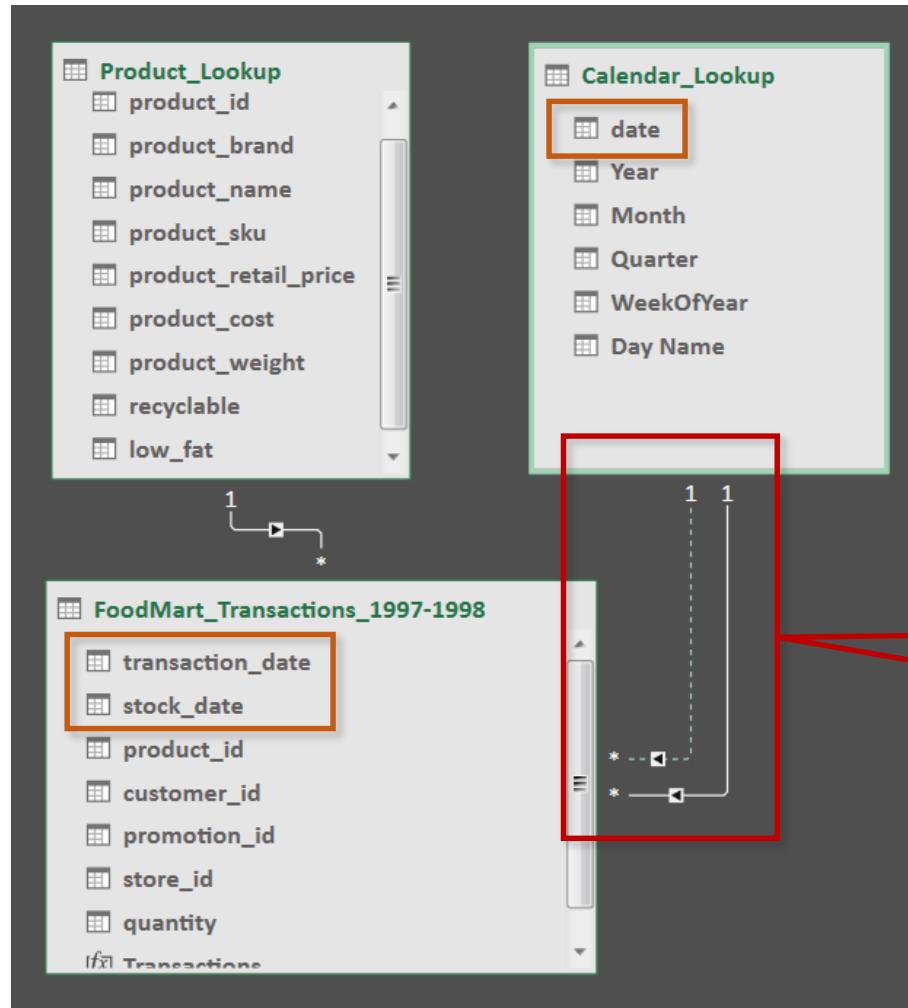
The **Manage Relationships** window allows you to create, edit or delete any connection in the data model

- Use this to see all table relationships, as well as table names, cardinality and filter direction
- **Note:** double-click a single connection in diagram view to edit an individual relationship

The 'Manage Relationships' dialog box is open, showing a list of existing connections. The table has columns for Active, Table 1, Cardinality, Filter Direction, and Table 2. The 'Create' button is highlighted.

Active	Table 1	Cardinality	Filter Direction	Table 2
Yes	FoodMart_Returns_1997-1998 [customer_id]	Many to One (*:1)	<< To FoodMart_Returns_1997-1998	Customer_Lookup [customer_id]
Yes	FoodMart_Returns_1997-1998 [date]	Many to One (*:1)	<< To FoodMart_Returns_1997-1998	Calendar_Lookup [date]
Yes	FoodMart_Returns_1997-1998 [product_id]	Many to One (*:1)	<< To FoodMart_Returns_1997-1998	Product_Lookup [product_id]
Yes	FoodMart_Returns_1997-1998 [store_id]	Many to One (*:1)	<< To FoodMart_Returns_1997-1998	Store_Lookup [store_id]
Yes	FoodMart_Transactions_1997-1998 [customer_id]	Many to One (*:1)	<< To FoodMart_Transactions_1997-1998	Customer_Lookup [customer_id]
Yes	FoodMart_Transactions_1997-1998 [product_id]	Many to One (*:1)	<< To FoodMart_Transactions_1997-1998	Product_Lookup [product_id]
Yes	FoodMart_Transactions_1997-1998 [promotion_id]	Many to One (*:1)	<< To FoodMart_Transactions_1997-1998	Promotion_Lookup [promotion_id]
No	FoodMart_Transactions_1997-1998 [stock_date]	Many to One (*:1)	<< To FoodMart_Transactions_1997-1998	Calendar_Lookup [date]
Yes	FoodMart_Transactions_1997-1998 [store_id]	Many to One (*:1)	<< To FoodMart_Transactions_1997-1998	Store_Lookup [store_id]
Yes	FoodMart_Transactions_1997-1998 [transaction_date]	Many to One (*:1)	<< To FoodMart_Transactions_1997-1998	Calendar_Lookup [date]

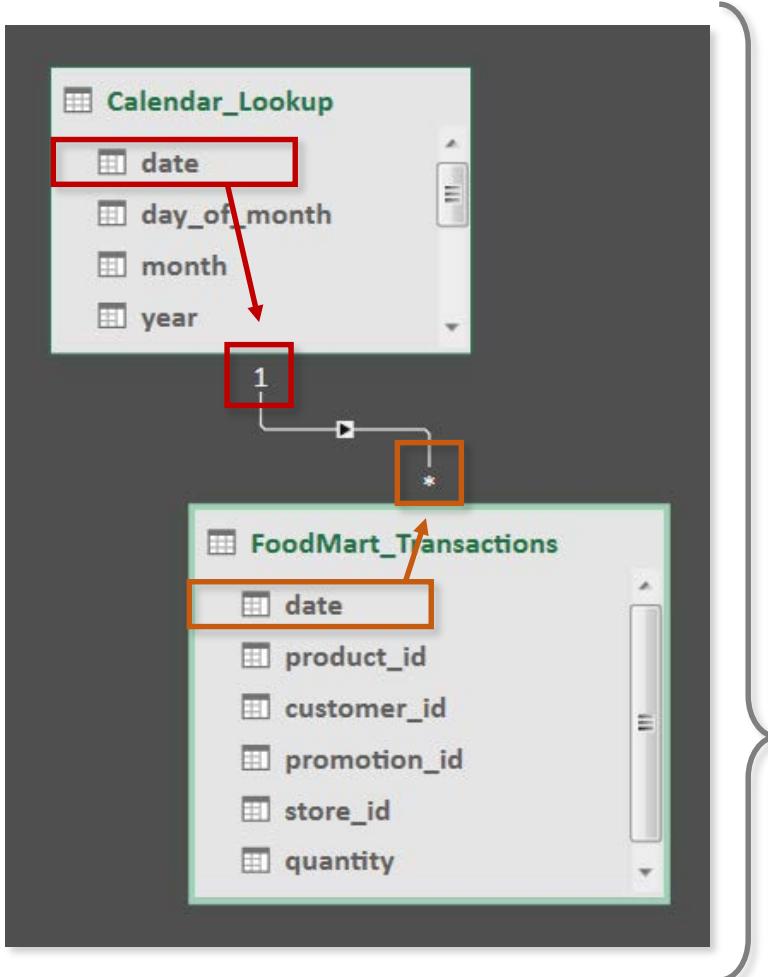
ACTIVE VS. INACTIVE RELATIONSHIPS



We can connect the **Calendar_Lookup** and **FoodMart_Transactions** tables on both **transaction_date** and **stock_date**; however, only one can be active at a time

To make a connection active or inactive, double-click the connection and check the box, or right-click the relationship line itself (**Note:** must deactivate one before activating another!)

RELATIONSHIP CARDINALITY



Cardinality refers to the uniqueness of values in a column

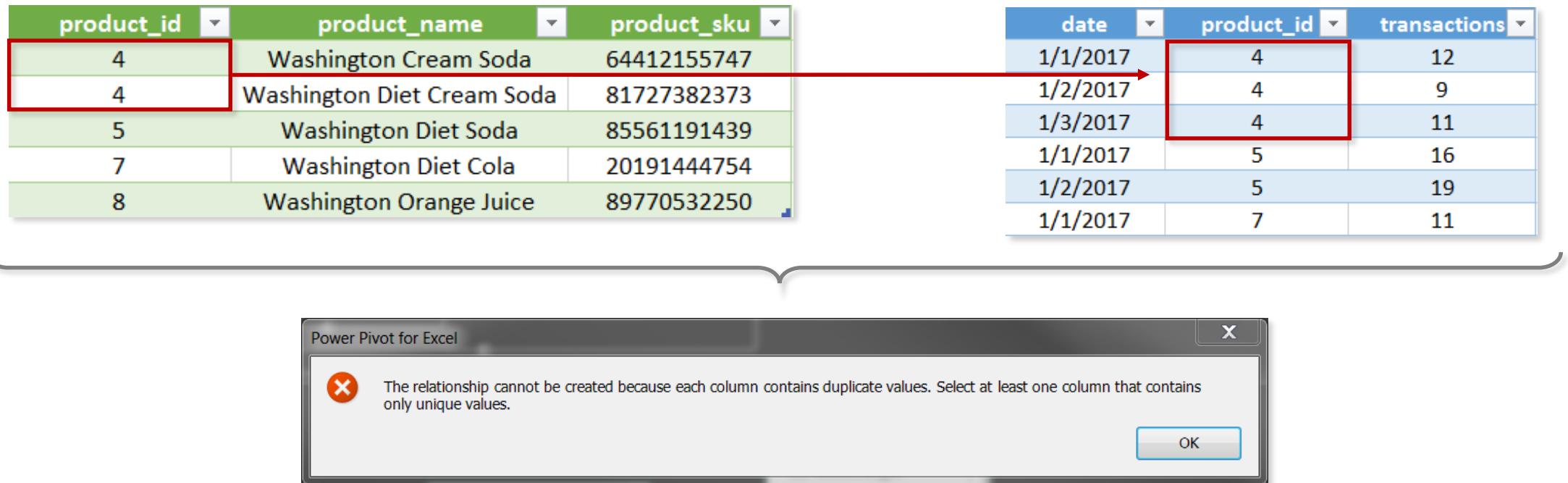
In Power Pivot, all relationships in a data model should follow a “**one-to-many**” cardinality

- Each column (or “key”) used to join tables can only have **one instance** of each unique value in the lookup table (these are the *primary keys*), but may have **many instances** of each unique value in the data table (these are the *foreign keys*)

*In this case we’re joining the **Calendar_Lookup** table to the **FoodMart_Transactions** data table using the **date** column as our key*

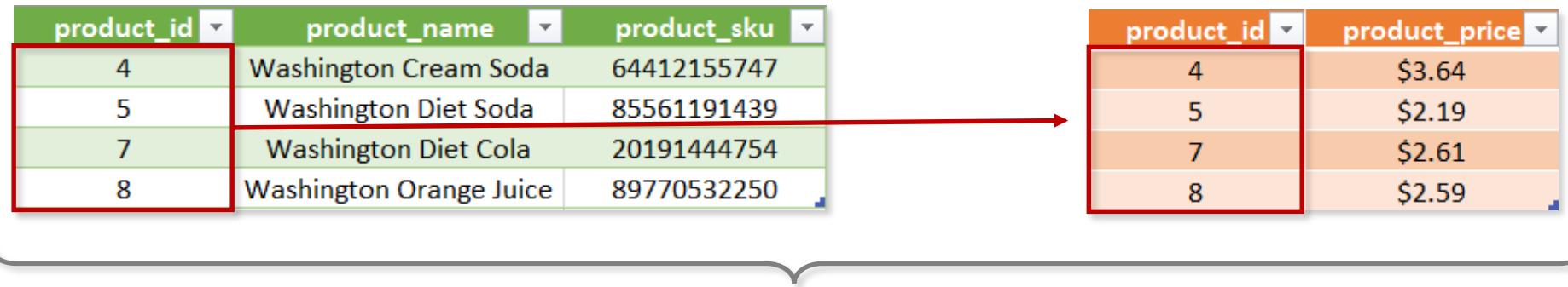
*There is only **one** instance of each date in the lookup table (noted by the “**1**”), but **many** instances of each date in the data table (noted by the asterisk “*****”), since multiple transactions occur each day*

BAD CARDINALITY: MANY-TO-MANY



- If we try to connect these tables using the **product_id** field, we'll have a **many-to-many** relationship since there are multiple instances of each ID in both tables
- Even if we *could* create this relationship in Power Pivot, how would you know which product was actually sold on each date – *Cream Soda* or *Diet Cream Soda*?

BAD CARDINALITY: ONE-TO-ONE



- In this case, connecting the tables above using the **product_id** field creates a **one-to-one** relationship, since each ID only appears once in each table
- Unlike many-to-many, there is nothing *illegal* about this relationship; it's just **inefficient**

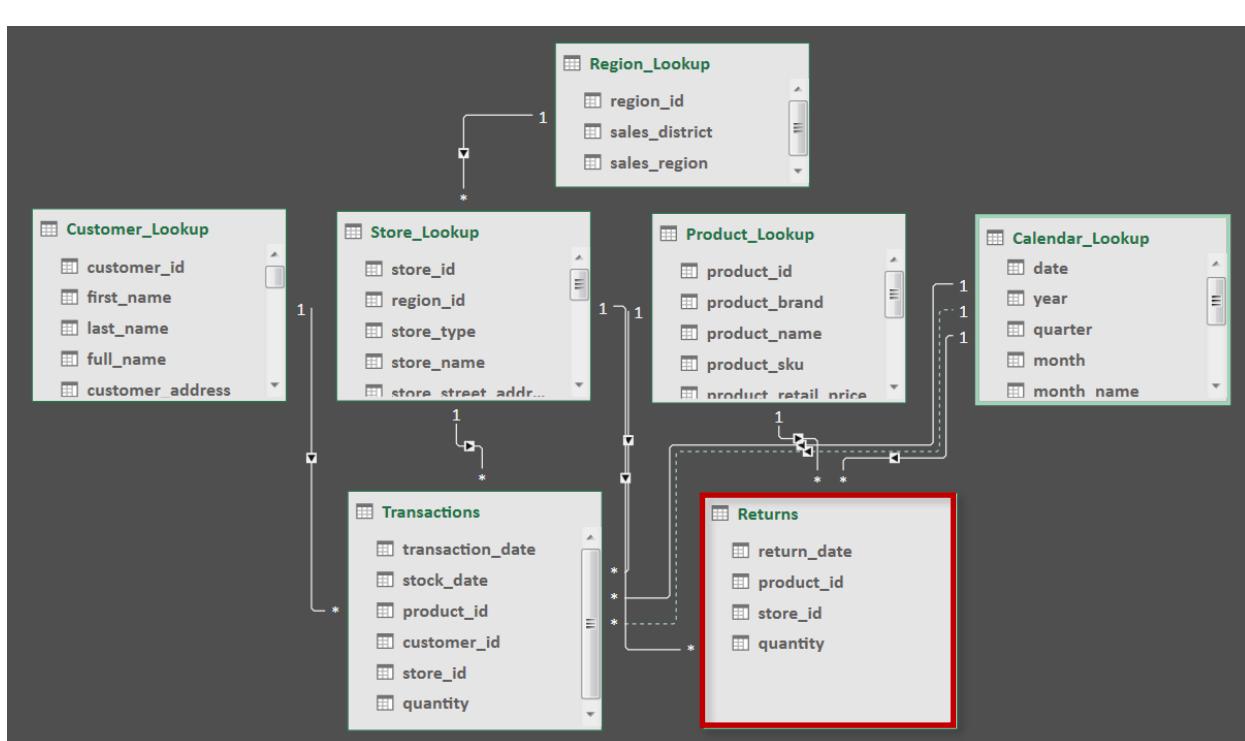
To eliminate the inefficiency, you could simply **merge the two tables** into a single, valid lookup

Note: this still respects the laws of normalization, since all rows are unique and directly related to the primary key

A curly brace on the left groups the two tables from the previous diagram, indicating they are being merged. To the right, a single table shows the result of this merge. It contains four rows with product IDs 4, 5, 7, and 8. Each row contains four columns: product_id, product_name, product_sku, and product_price. The rows are color-coded to match the original tables: the first row (ID 4) is light green, the second (ID 5) is light orange, the third (ID 7) is light green, and the fourth (ID 8) is light orange.

product_id	product_name	product_sku	product_price
4	Washington Cream Soda	64412155747	\$3.64
5	Washington Diet Soda	85561191439	\$2.19
7	Washington Diet Cola	20191444754	\$2.61
8	Washington Orange Juice	89770532250	\$2.59

CONNECTING MULTIPLE DATA TABLES



Here we've loaded a second data table named **Returns**, containing records of returns by date, product and store

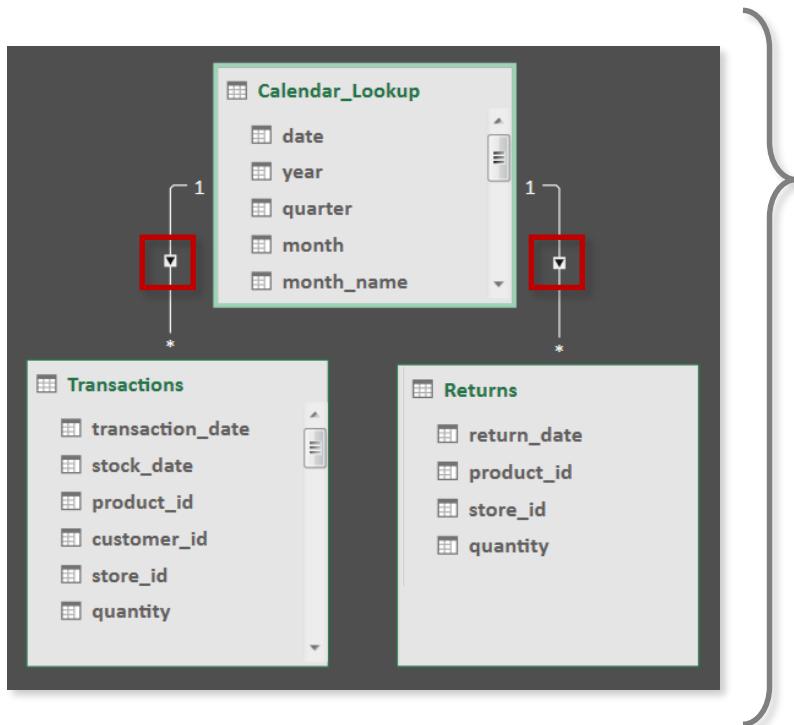
- This table connects to each lookup exactly like the **Transactions** table did, except that there is no way to connect the **Returns** table to **Customer_Lookup**
- This allows us to analyze data across both tables in the same pivot, **as long as we only filter or segment the data using lookups that are common to both**
 - In other words, we know which *product* was returned, which *store* it was returned to, and which *date* the return occurred, but NOT which *customer* was responsible

HEY THIS IS IMPORTANT!

NEVER try to connect data tables directly to each other;
ALWAYS connect them indirectly via shared lookup tables!



FILTER DIRECTION IS IMPORTANT



This model includes two data tables (**Transactions** and **Returns**), both connected to the **Calendar_Lookup**

Note the filter directions (shown as arrows) in each relationship; **in Power Pivot (2016) these will always point from the “one” side of the relationship (lookups) to the “many” side (data tables)***

- Filtering a table will impact any tables “downstream” of it, as defined by the filter relationship (i.e the direction of the arrow)
- Let’s say we’re analyzing both Transactions and Returns in the same PivotTable; filtering by the **Calendar_Lookup** date field will return correctly filtered data from both data tables, but filtering by the **Transactions** date field will yield *unfiltered* Returns values

PRO TIP:



Arrange your lookup tables **above** your data tables in diagram view to remind you that filters always flow “downstream”

FILTER DIRECTION IS IMPORTANT (CONT.)

PivotTable Fields ▾ ×

Active All

Choose fields to add to report:

Search 🔎

Calendar_Lookup
date
Year
Month
Quarter
WeekOfYear
Day Name

FoodMart_Returns

FoodMart_Transactions

Drag fields between areas below:

Filters
Rows
date

Columns
Values

Rows
Transactions
Returns

Values
Transactions
Returns

Defer Layout Update Update

	A	B	C
1	Row Labels	Transactions	Returns
2	1/1/1997	348	3
3	1/2/1997	635	6
4	1/3/1997	589	7
5	1/4/1997	20	
6	1/5/1997	966	10
7	1/6/1997	993	11
8	1/7/1997	1,265	8
9	1/8/1997	35	
10	1/9/1997	525	9
11	1/10/1997	460	5



Calendar_Lookup filters flow “down” to both the **Transactions** and **Returns** tables, so we can filter or segment those metrics using any field from the Calendar table

PivotTable Fields ▾ ×

Active All

Choose fields to add to report:

Relationships between tables may be needed.

Auto-Detect... CREATE...

Search 🔎

Calendar_Lookup

FoodMart_Returns

FoodMart_Transactions
transaction_date
stock_date
product_id

Drag fields between areas below:

Filters
Rows
transaction_d...

Columns
Values

Rows
Transactions
Returns

Values
Transactions
Returns

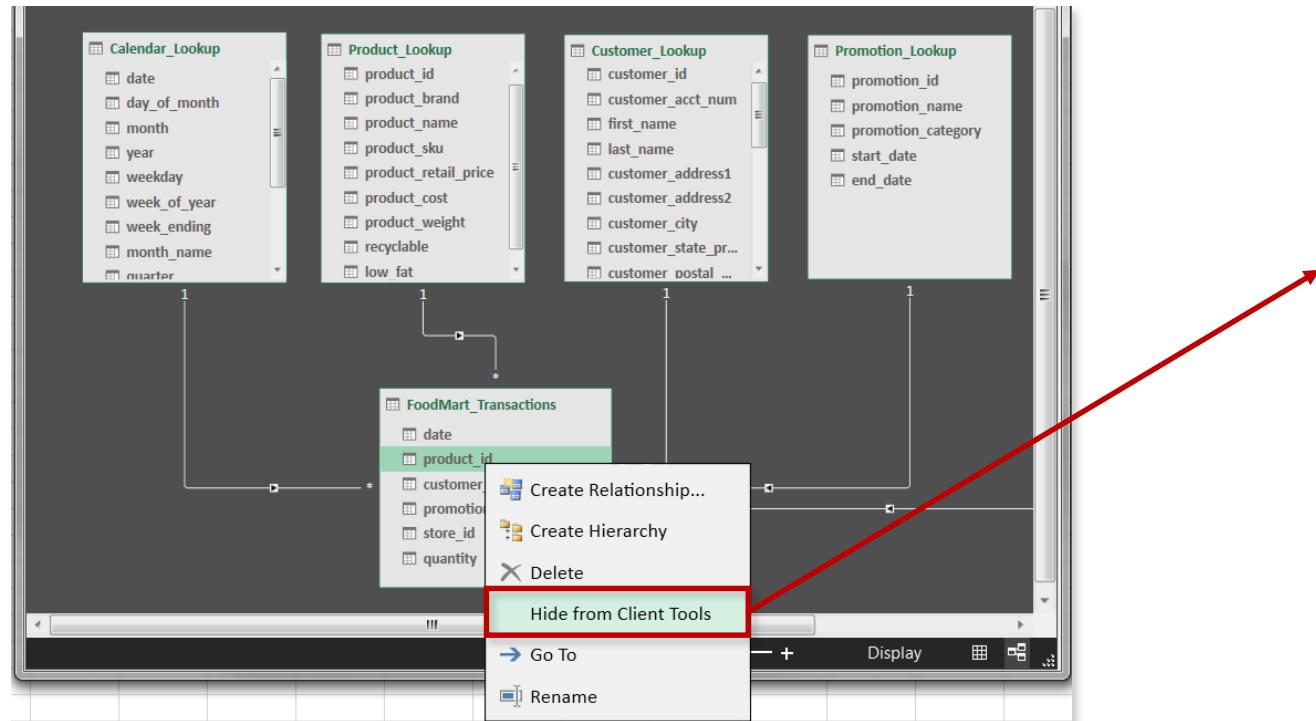
Defer Layout Update Update

	A	B	C
1	Row Labels	Transactions	Returns
2	1/1/1997	348	8,289
3	1/2/1997	635	8,289
4	1/3/1997	589	8,289
5	1/4/1997	20	8,289
6	1/5/1997	966	8,289
7	1/6/1997	993	8,289
8	1/7/1997	1,265	8,289
9	1/8/1997	35	8,289
10	1/9/1997	525	8,289
11	1/10/1997	460	8,289



Filtering by date in the **Transactions** table yields incorrect, unfiltered values from the **Returns** table, since filter context cannot flow “upstream” to the **Calendar** table

HIDING FIELDS FROM CLIENT TOOLS



When you **hide a field from Client Tools**, you make it invisible to tools outside of the data model (i.e. Power Pivot)

This can be used to prevent users from filtering or segmenting on invalid fields, or to hide irrelevant metrics from view



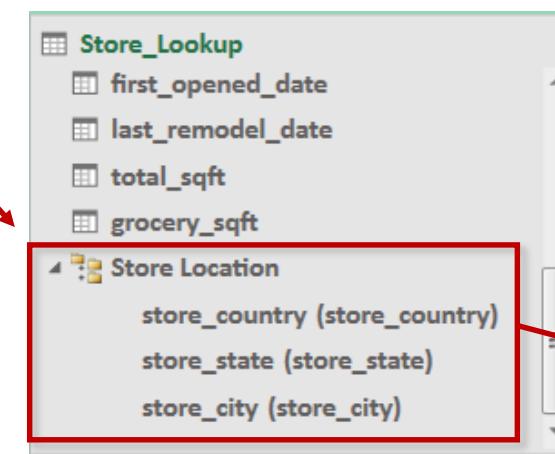
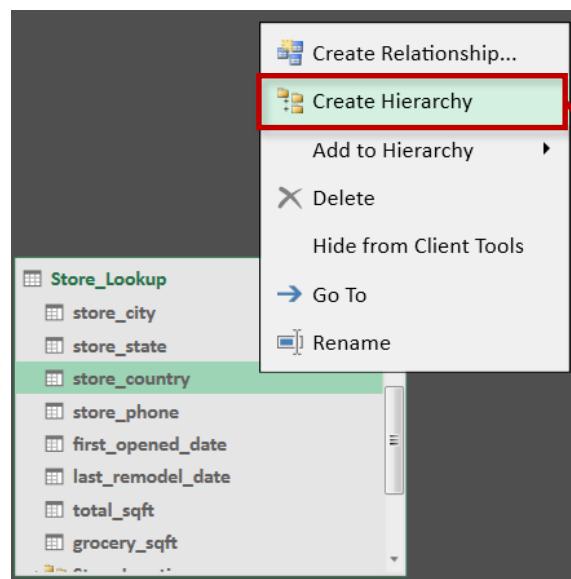
PRO TIP:

Always hide the **foreign key columns** in your data tables to prevent users from accidentally filtering on them!

DEFINING HIERARCHIES

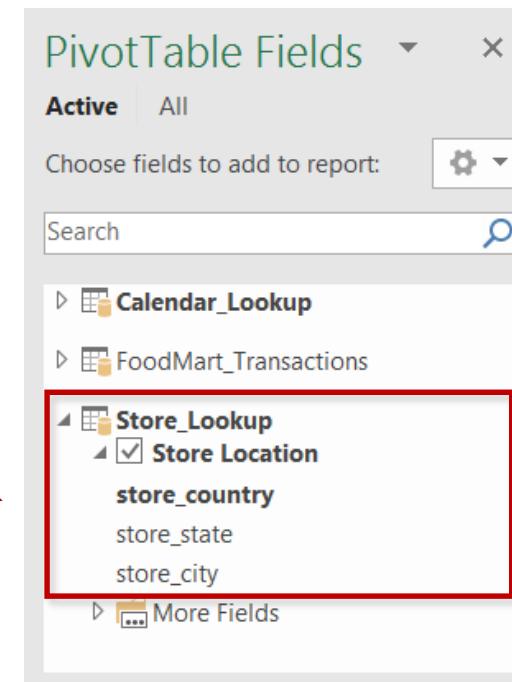
Hierarchies are groups of nested columns that reflect multiple levels of granularity

- For example, a “**Geography**” hierarchy might include **Country**, **State**, and **City** columns
- Each hierarchy is treated as a **single item** in PivotTables and PivotCharts, allowing users to “drill up” and “drill down” through different levels of the hierarchy in a meaningful way



Right-click a field to see the hierarchy options

Drag fields to create a hierarchy



Hierarchies appear in Power Pivot

DATA MODEL BEST PRACTICES



Normalize your data model before you do anything else

- *Make sure that each table in your model serves a single, distinct purpose*
- *Use relationships vs. merged tables; long & narrow tables are better than short & wide*



Organize lookup tables *above* data tables in the diagram view

- *This serves as a visual reminder that filters always flow “downstream”*



Hide fields from client tools to prevent invalid filter context

- *All foreign key columns should be hidden from data tables, so that users are only able to use valid fields for filtering and segmentation*

POWER PIVOT & DAX 101

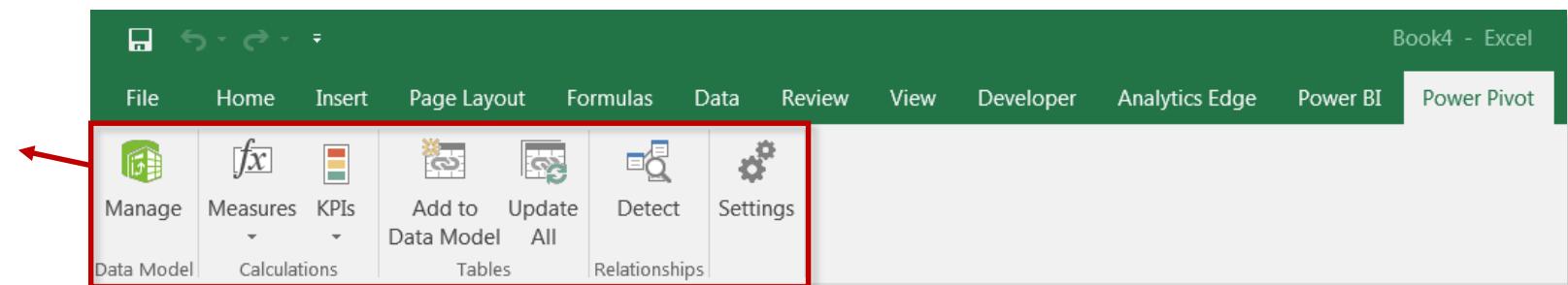
MEET POWER PIVOT

A “**Power**” Pivot is just like a normal PivotTable, except it sits on top of an *entire data model* rather than a single table or range. This allows you to:

- Explore massive datasets consisting of multiple sources and tables, using familiar, user-friendly PivotTable tools and options
- Create powerful and flexible calculations using Data Analysis Expressions (DAX)

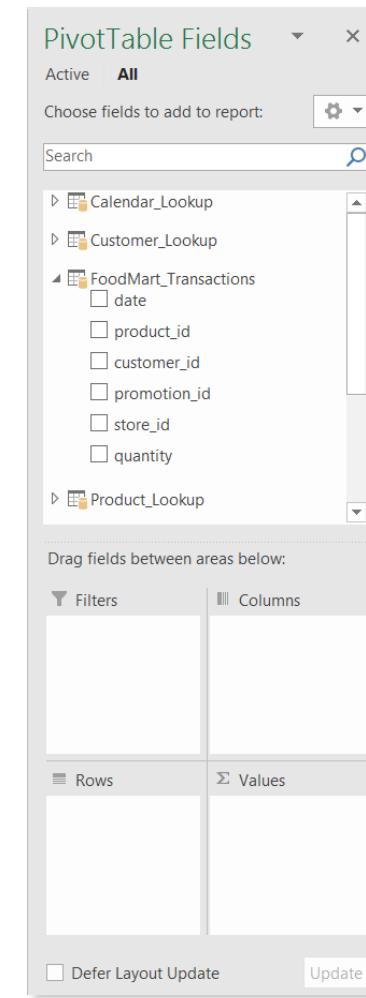
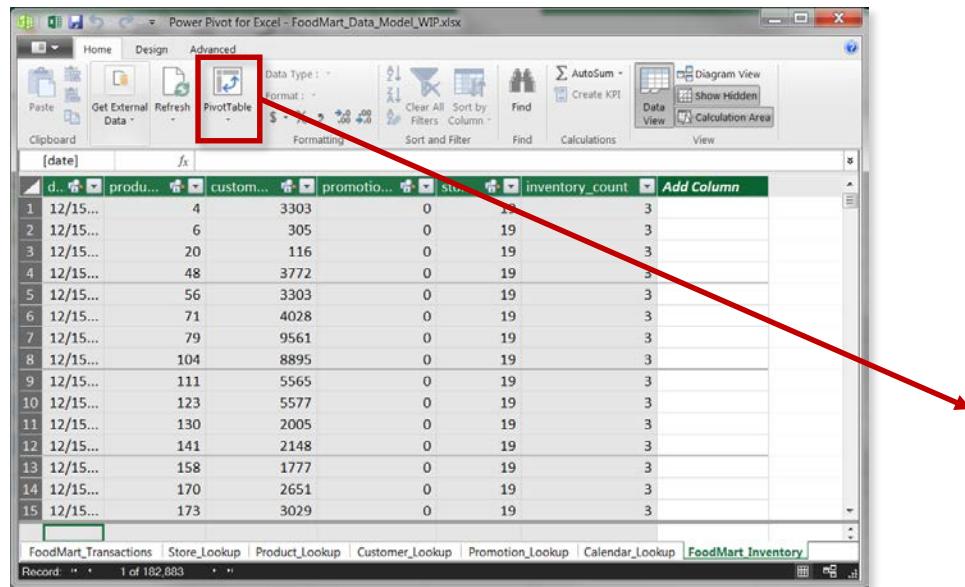
The **Power Pivot** tab includes tools to manage the data model and define new measures

(Note: you may need to enable this tab by selecting **File > Options > Add-Ins > Manage COM Add-Ins**)

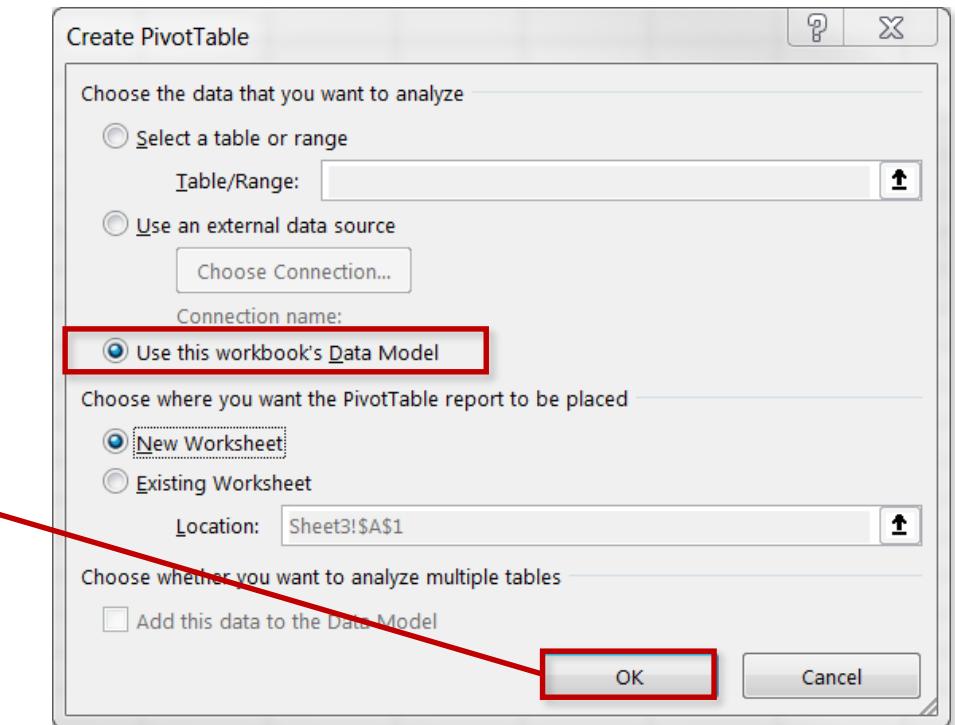


CREATING A “POWER” PIVOT TABLE

Option #1: From the Data Model



Option #2: From the *Insert > PivotTable* dialog box



“NORMAL” PIVOTS VS. “POWER” PIVOTS



NORMAL PIVOT

- Can analyze data from **one table at a time**; multiple tables must be flattened or “stitched” together with cell functions
- Restricted to the data capacity of a **single Excel worksheet** (1,048,576 rows)
- Limited to relatively **basic calculated fields**, using a sub-set of Excel functions



POWER PIVOT

- Can analyze an **entire data model**, consisting of multiple tables connected via relationships rather than cell functions
- Virtually **unlimited data capacity** as tables are compressed outside of normal worksheets
- Performs **complex calculations** using Data Analysis Expressions (DAX)

NOTE: It's not the *PivotTable* itself that's different; it's the *data behind it*

“NORMAL” PIVOTS VS. “POWER” PIVOTS

Normal Pivot

A screenshot of a regular PivotTable in Excel. The data is in rows 6 to 20, with columns A, B, and C. Row 6 contains the header "transaction_date" and "Sum of quantity". The "Sum of quantity" cell (B7) is highlighted. The PivotTable Fields pane shows fields like transaction_date, product_id, customer_id, promotion_id, store_id, and quantity. The "More Tables..." button is visible. The status bar at the bottom says "Regular Pivot".

	A	B	C	D
1	product_id	(All)		
2	customer_id	(All)		
3	promotion_id	(All)		
4	store_id	(All)		
5				
6	transaction_date	Sum of quantity		
7	1/1/1997	348		
8	1/2/1997	635		
9	1/3/1997	589		
10	1/4/1997	20		
11	1/5/1997	966		
12	1/6/1997	993		
13	1/7/1997	1,265		
14	1/8/1997	35		
15	1/9/1997	525		
16	1/10/1997	460		
17	1/11/1997	2,760		
18	1/12/1997	355		
19	1/13/1997	1,629		
20	1/14/1997	261		
21	1/15/1997	512		
22	1/16/1997	585		
23	1/17/1997	1,146		
24	1/18/1997	420		
25	1/19/1997	688		
26	1/20/1997	2,112		
27	1/21/1997	1,198		
28	1/23/1997	798		
29	1/24/1997	28		
30	1/25/1997	1,172		

PivotTable Fields

Choose fields to add to report:

transaction_date
product_id
customer_id
promotion_id
store_id
quantity

More Tables...

Drag fields between areas below:

Filters

product_id
customer_id
promotion_id

Columns

Rows

transaction_date

Values

Sum of quantity

Defer Layout Update

Update

Regular Pivot

Power Pivot

A screenshot of a Power PivotTable in Excel. The data is in rows 6 to 20, with columns A, B, and C. Row 6 contains the header "date" and "Total Quantity". The "Total Quantity" cell (B7) is highlighted. The PivotTable Fields pane shows fields like date and Total Quantity. The "More Tables..." button is visible, and a red box highlights the "More Tables!" section which lists Calendar_Lookup, Customer_Lookup, FoodMart_Transactions_1997, Product_Lookup, Promotion_Lookup, and Store_Lookup. The status bar at the bottom says "Power Pivot".

	A	B	C	D
1	product_id	All		
2	customer_id	All		
3	promotion_id	All		
4	store_id	All		
5				
6	date	Total Quantity		
7	1/1/1997	348		
8	1/2/1997	635		
9	1/3/1997	589		
10	1/4/1997	20		
11	1/5/1997	966		
12	1/6/1997	993		
13	1/7/1997	1,265		
14	1/8/1997	35		
15	1/9/1997	525		
16	1/10/1997	460		
17	1/11/1997	2,760		
18	1/12/1997	355		
19	1/13/1997	1,629		
20	1/14/1997	261		
21	1/15/1997	512		
22	1/16/1997	585		
23	1/17/1997	1,146		
24	1/18/1997	420		
25	1/19/1997	688		
26	1/20/1997	2,112		
27	1/21/1997	1,198		
28	1/23/1997	798		
29	1/24/1997	28		
30	1/25/1997	1,172		

PivotTable Fields

Active All

More Tables!

Choose fields to add to report:

Search

Calendar_Lookup
Customer_Lookup
FoodMart_Transactions_1997
Product_Lookup
Promotion_Lookup
Store_Lookup

Drag fields between areas below:

Filters

product_id
customer_id
promotion_id

Columns

date

Values

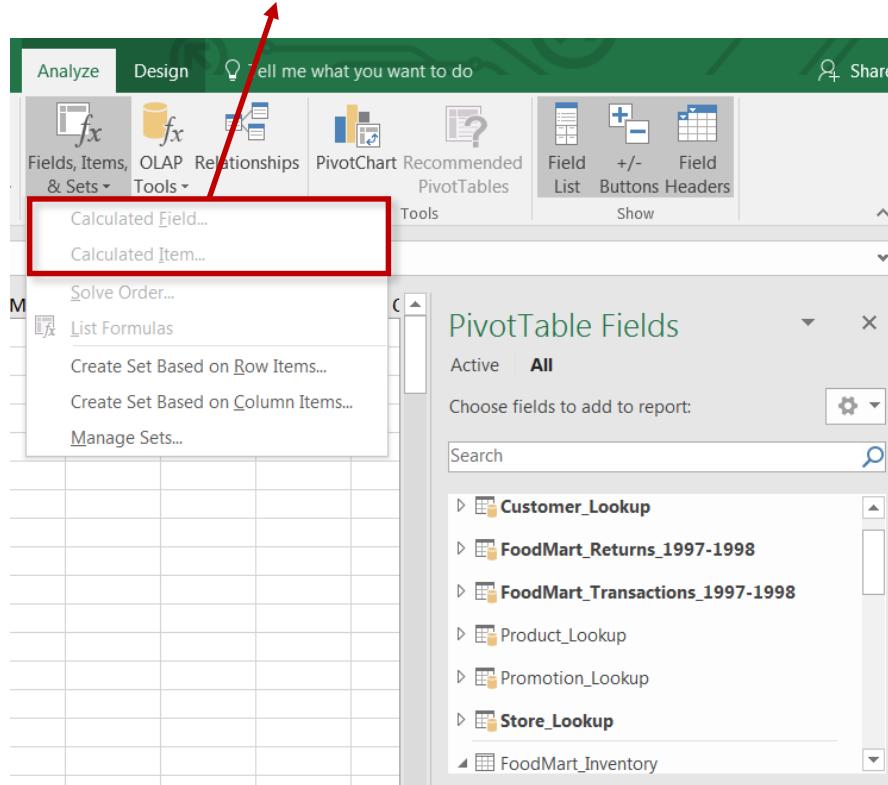
Total Quantity

Defer Layout Update

Update

NO MORE “CALCULATED FIELDS”

*Oh rats, where are my **calculated fields**??*



One of the key Power Pivot features is the ability to create *much* more robust calculated fields, known as **measures***

Because these measures interact directly with the data model (including tables stored in memory), traditional cell formulas won't do the trick

- Instead, we'll use a new (but familiar) formula language called **Data Analysis Expressions (DAX)**

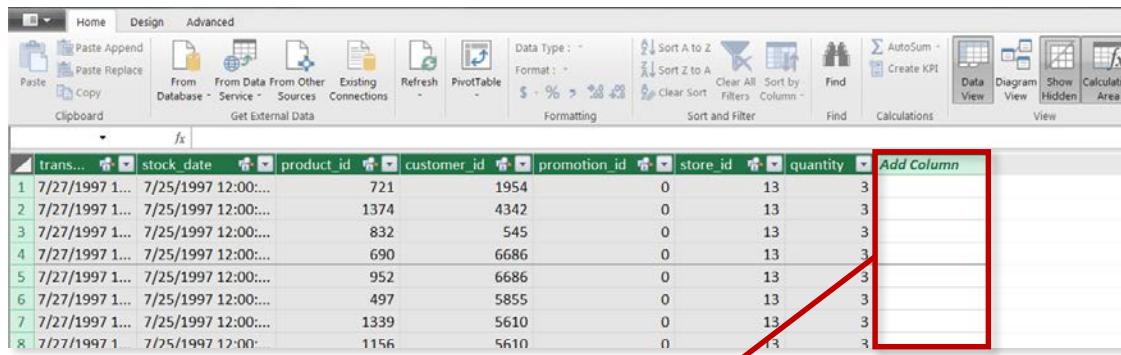
*Note: Depending on the version of Excel you're using, you might see these referred to as either “**Measures**” (Excel 2010, 2016) or “**Calculated Fields**” (Excel 2013)

DATA ANALYSIS EXPRESSIONS (DAX)

Data Analysis Expressions, commonly known as **DAX**, is the formula language that drives Power Pivot. With DAX, you can:

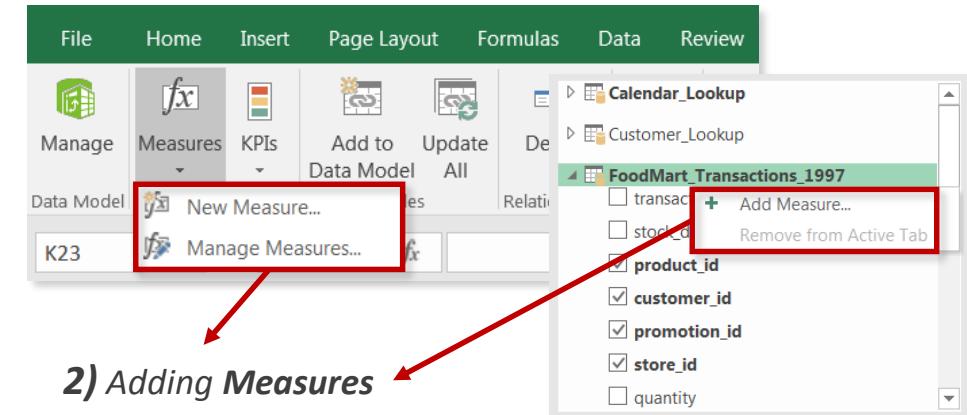
- Add **calculated columns** and **measures** to your model, using intuitive syntax
- Go beyond the capabilities of traditional “grid-style” formulas, with powerful functions built specifically to work with relational data

Two places to use DAX:



A screenshot of Microsoft Excel showing a data grid. The grid contains several columns with numerical and date values. At the bottom right of the grid, there is a button labeled "Add Column". A red box highlights this button.

1) Adding *Calculated Columns*



A screenshot of Microsoft Power Pivot. The ribbon at the top shows tabs like File, Home, Insert, etc. Below the ribbon, there's a toolbar with icons for Manage, Measures, KPIs, and Data Model. The "Measures" icon is highlighted. A context menu is open over a data table, with the "New Measure..." option highlighted by a red box. A red arrow points from the "Add Column" button in the Excel screenshot to this "New Measure..." option. Another red box highlights the "FoodMart_Transactions_1997" table in the data model list on the right. A second red arrow points from the "Measures" tab in the Power Pivot ribbon to the "FoodMart_Transactions_1997" table.

2) Adding **Measures**

CALCULATED COLUMNS

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

- No “A1-style” references; calculated columns refer to **entire tables or columns**
- Calculated columns are computed at the row-level, and **values are stored with the table (this eats up memory)**
- Calculated columns understand **row context**; they’re great for defining new properties based on information in each row, but generally useless for aggregation (*SUM, AVERAGE, COUNT, etc.*)



HEY THIS IS IMPORTANT!

As a rule of thumb, **ONLY** use calculated columns if you want to “stamp” static, fixed values to each row in a table (or use Power Query!)

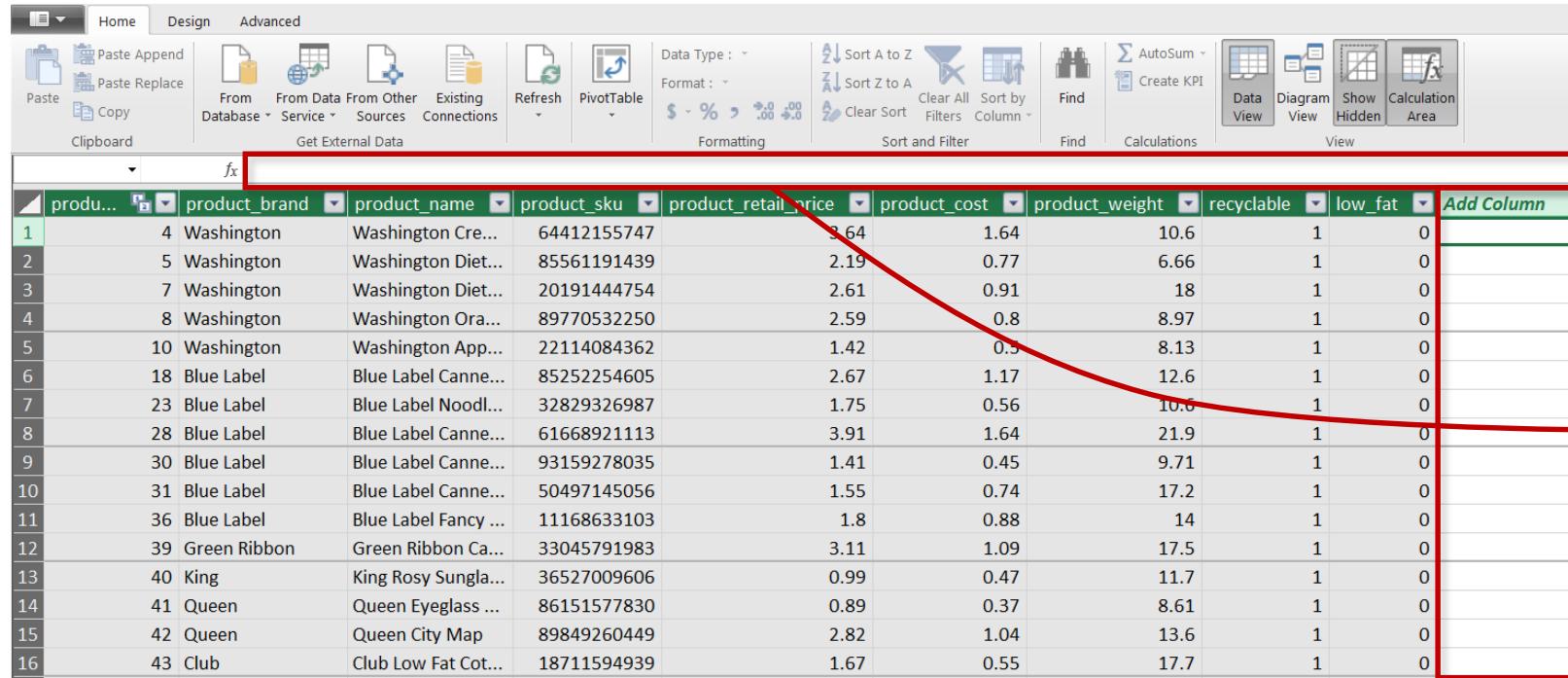
DO NOT use calculated columns for aggregation formulas, or to calculate fields for the “Values” area of a pivot (use **measures** instead)



PRO TIP:

*Calculated columns are typically placed in the **Filters, Slicers, Rows or Columns** areas of a pivot*

CREATING CALCULATED COLUMNS



The screenshot shows the Microsoft Power BI Data View ribbon. The 'Home' tab is selected. In the 'View' section, the 'Data View' button is highlighted. A red box highlights the 'Add Column' button in the bottom right corner of the table area. The table contains 16 rows of product data with columns for product ID, brand, name, SKU, retail price, cost, weight, recyclability, low fat status, and a new column labeled 'Add Column'.

product_id	product_brand	product_name	product_sku	product_retail_price	product_cost	product_weight	recyclable	low_fat	Add Column
1	4	Washington	Washington Cre...	64412155747	3.64	1.64	10.6	1	0
2	5	Washington	Washington Diet...	85561191439	2.19	0.77	6.66	1	0
3	7	Washington	Washington Diet...	20191444754	2.61	0.91	18	1	0
4	8	Washington	Washington Ora...	89770532250	2.59	0.8	8.97	1	0
5	10	Washington	Washington App...	22114084362	1.42	0.5	8.13	1	0
6	18	Blue Label	Blue Label Canne...	85252254605	2.67	1.17	12.6	1	0
7	23	Blue Label	Blue Label Noodl...	32829326987	1.75	0.56	10.6	1	0
8	28	Blue Label	Blue Label Canne...	61668921113	3.91	1.64	21.9	1	0
9	30	Blue Label	Blue Label Canne...	93159278035	1.41	0.45	9.71	1	0
10	31	Blue Label	Blue Label Canne...	50497145056	1.55	0.74	17.2	1	0
11	36	Blue Label	Blue Label Fancy ...	11168633103	1.8	0.88	14	1	0
12	39	Green Ribbon	Green Ribbon Ca...	33045791983	3.11	1.09	17.5	1	0
13	40	King	King Rosy Sungla...	36527009606	0.99	0.47	11.7	1	0
14	41	Queen	Queen Eyeglass ...	86151577830	0.89	0.37	8.61	1	0
15	42	Queen	Queen City Map	89849260449	2.82	1.04	13.6	1	0
16	43	Club	Club Low Fat Cot...	18711594939	1.67	0.55	17.7	1	0

Step 1: In the data model “Data View”, choose a table and then select any cell in the “Add Column” section

Step 2: Enter a DAX function in the formula bar (*we'll cover specific functions in the next section*)

Step 3: Press “Enter”, and all cells in the column will update

CALCULATED COLUMNS: GOOD & BAD

A screenshot of the Power BI Data View interface. A red box highlights the formula bar with the expression `=IF(Product_Lookup[product_retail_price]>2,"High","Low")`. Another red box highlights the 'price_category' column in the table, which contains values like 'High' and 'Low'. A green thumbs-up icon is overlaid on the table area.

product_id	product_brand	product_name	product_sku	product_retail_price	product_cost	product_weight	recyclable	low_fat	price_category
1	4 Washington	Washington Cre...	64412155747	3.64	1.64	10.6	1	0	High
2	5 Washington	Washington Diet...	85561191439	2.19	0.77	6.66	1	0	High
3	7 Washington	Washington Diet...	20191444754	2.61	0.91	18	1	0	High
4	8 Washington	Washington Ora...	89770532250	2.59	0.8	8.97	1	0	High
5	10 Washington	Washington App...	22114084362	1.42	0.5	8.13	1	0	Low
6	18 Blue Label	Blue Label Canne...	85252254605	2.67	1.17	12.6	1	0	High
7	23 Blue Label	Blue Label Noodl...	32829326987	1.75	0.56	10.6	1	0	Low
8	28 Blue Label	Blue Label Canne...	61668921113	3.91	1.64	21.9	1	0	High
9	30 Blue Label	Blue Label Canne...	93159278035	1.41	0.45	9.71	1	0	Low
10	31 Blue Label	Blue Label Canne...	50497145056	1.55	0.74	17.2	1	0	Low
11	36 Blue Label	Blue Label Fancy ...	11168633103	1.8	0.88	14	1	0	Low
12	39 Green Ribbon	Green Ribbon Ca...	33045791983	3.11	1.09	17.5	1	0	High
13	40 King	King Rosy Sungla...	36527009606	0.99	0.47	11.7	1	0	Low
14	41 Queen	Queen Eyeglass ...	86151577830	0.89	0.37	8.61	1	0	Low
15	42 Queen	Queen City Map	89849260449	2.82	1.04	13.6	1	0	High
16	43 Club	Club Low Fat Cot...	18711594939	1.67	0.55	17.7	1	0	Low

In this case we've added a **calculated column** called **price_category**, which equals "**High**" if the retail price is >\$2, and "**Low**" otherwise (*just like you would write in Excel!*)

- Since calculated columns understand **row context**, a new value is calculated in each row based on that row's price
- This is a **valid use** of calculated columns; it creates a new row "property" that we can now 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_revenue**

- Since calculated columns do not understand **filter context**, the same grand total is returned in *every single 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, subdivided, etc.

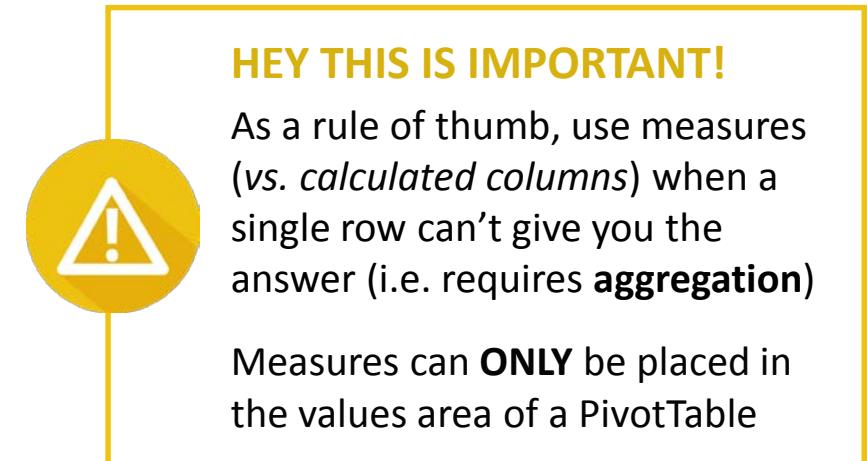
A screenshot of the Power BI Data View interface. A red box highlights the formula bar with the expression `=SUM(Transactions[revenue])`. Another red box highlights the 'total_revenue' column in the table, which contains values like '\$1,199,308.31'. A red arrow points from the formula bar to the 'total_revenue' column.

transaction_id	stock_date	product_id	customer_id	promotion_id	store_id	quantity	retail_price	revenue	total_revenue	
1	12/15/1998 12:00:00	12/8/1998 1...	4	3303	0	19	3	3.64	10.92	\$1,199,308.31
2	12/15/1998 12:00:00	12/10/1998 ...	6	305	0	19	3	1.15	3.45	\$1,199,308.31
3	12/15/1998 12:00:00	12/13/1998 ...	20	116	0	19	3	2.78	8.34	\$1,199,308.31
4	12/15/1998 12:00:00	12/8/1998 1...	48	3772	0	19	3	1.88	5.64	\$1,199,308.31
5	12/15/1998 12:00:00	12/13/1998 ...	56	3303	0	19	3	0.71	2.13	\$1,199,308.31
6	12/15/1998 12:00:00	12/12/1998 ...	71	4028	0	19	3	2.76	8.28	\$1,199,308.31
7	12/15/1998 12:00:00	12/11/1998 ...	79	9561	0	19	3	1.32	3.96	\$1,199,308.31
8	12/15/1998 12:00:00	12/12/1998 ...	104	8895	0	19	3	3.89	11.67	\$1,199,308.31
9	12/15/1998 12:00:00	12/12/1998 ...	111	5565	0	19	3	3.48	10.44	\$1,199,308.31
10	12/15/1998 12:00:00	12/12/1998 ...	123	5577	0	19	3	3.84	11.52	\$1,199,308.31

DAX MEASURES

Measures are DAX formulas used to generate dynamic values within a PivotTable

- Like calculated columns, measures reference **entire tables** or **columns** (*no A1-style or “grid” references*)
- Unlike calculated columns, measures don’t actually *live* in the table; they get placed in the **values** area of a PivotTable and dynamically calculated in each individual cell
- Measures are evaluated based on the **filter context** of each cell, which is determined by the PivotTable layout (filters, slicers, rows and columns)



PRO TIP:

Use measures to create values that users can explore with a pivot (Power Pivot version of a “Calculated Field”)

CREATING IMPLICIT MEASURES

The screenshot shows the 'PivotTable Fields' ribbon. In the 'Values' section, the 'Sum of quantity' field is selected. A red arrow points from the text 'STEP 1' to the checked 'quantity' field in the list above. Another red arrow points from the text 'STEP 2' to the 'Sum of quantity' field in the 'Values' section.

STEP 1: Check the box next to a value field in a data table, or manually drag it into the “Values” box

STEP 2: Pat yourself on the back, you just created a measure!

HEY THIS IS IMPORTANT!

Before you pop the champagne, there's a catch. When you drag a raw data field into the values section of a pivot, you create what's called an **implicit measure**. While there's nothing *wrong* with implicit measures, they are extremely limited.

Explicit measures (defined using DAX) will give us *much* more flexibility, as well as the ability to reuse measures in multiple places (measure trees!)

FROM NOW ON, JUST SAY “NO” TO IMPLICIT MEASURES

CREATING EXPLICIT MEASURES (AUTOSUM)

The screenshot shows the Power BI Data View interface. At the top, there's a ribbon with tabs like Home, Design, Advanced, and various data import options. Below the ribbon is a toolbar with Paste, Refresh, PivotTable, and other data manipulation tools. A dropdown menu labeled "AutoSum" is open, showing options: Sum, Average, Count, Distinct Count, Max, and Min. The "Sum" option is highlighted. To the right of the toolbar, there are buttons for Diagram View, Show Hidden, and Calculation Area. The main area contains a table with columns for transaction_id, stock_date, product_id, customer_id, promotion_id, store_id, quantity, product_name, and revenue_cc. The "revenue_cc" column has some calculated values. At the bottom, there's a "Measures Pane" section with a single entry: "Revenue_Measure:...".

AutoSum is a shortcut for creating simple DAX formulas (Sum, Average, Count, Distinct Count, Max and Min)

To use AutoSum:

- Click on a cell in the Measures Pane (see below), within the column you want to evaluate
- Select the AutoSum menu and choose an option from the list

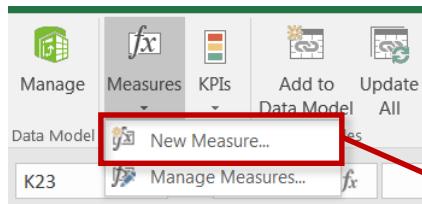
The **Measures Pane** sits beneath the data in the “Data View” of the model



PRO TIP:

AutoSum is a nice way to get comfortable with basic DAX and quickly add measures; just don't rely on them when things start to get more complicated!

CREATING EXPLICIT MEASURES (POWER PIVOT)

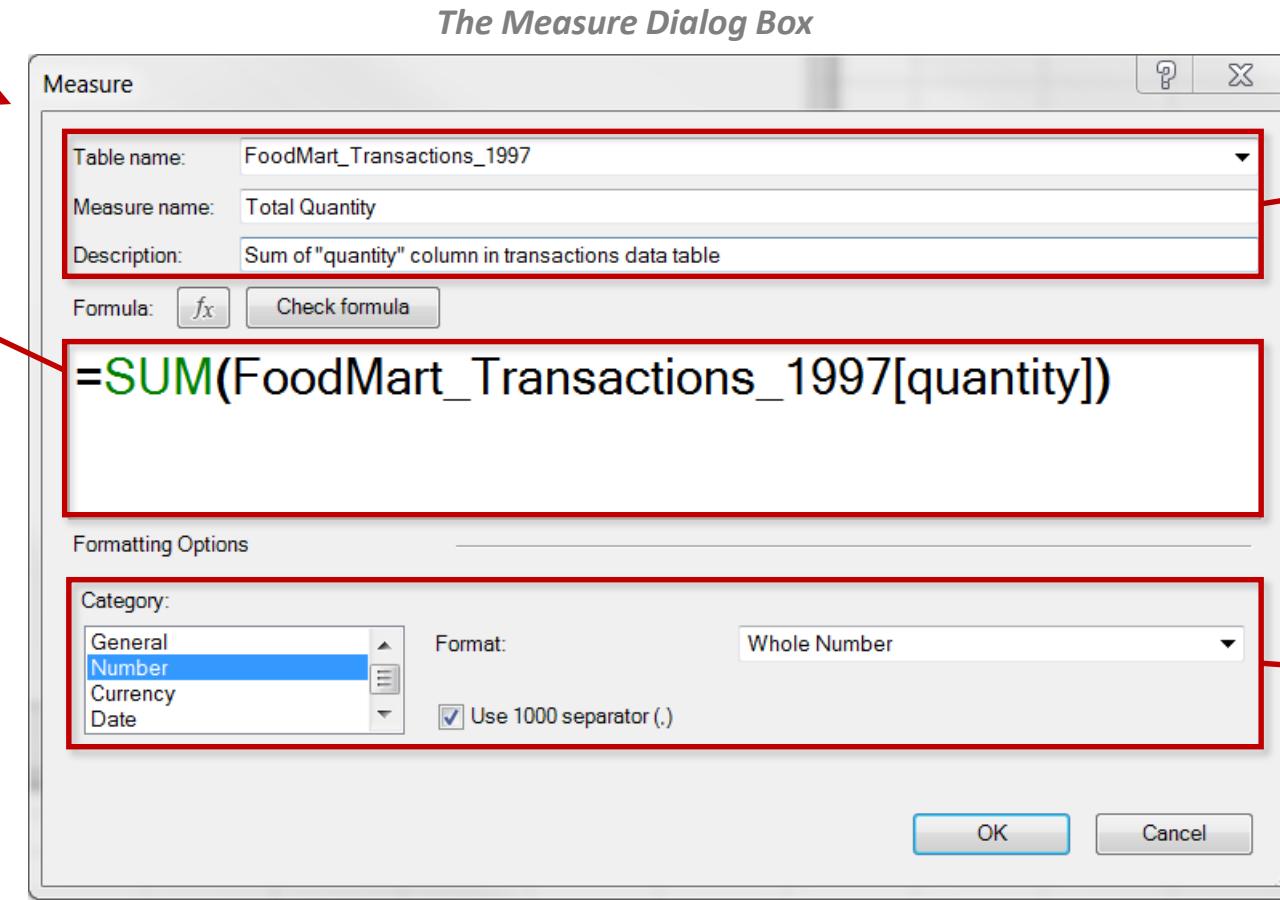


The **Formula** pane contains the actual DAX code, as well as options to browse the formula library or check syntax

Note: just start typing, and "Intellisense" will kick in to help you auto-populate formula names and tables

PRO TIP:

Ctrl+scroll adjusts formula text size



Each measure is assigned to a table and given a **measure name** (as well as an optional description)

Use the **Formatting Options** to specify a format for each measure

UNDERSTANDING FILTER CONTEXT

Measures are calculated based on **filter context**, which is the set of filters (or “coordinates”) determined by the PivotTable layout (filters, slicers, row labels and column labels)



HEY THIS IS IMPORTANT!

Each measure cell in the pivot **calculates independently**, based on its coordinates (*think of each cell as an island*)

When you change the pivot layout (by updating filters/slicers, row labels or column labels), each measure cell **detects its new coordinates** and then **recalculates its value**

customer_city	Total Quantity
Acapulco	16,428
Camacho	26,024
Hidalgo	52,888
La Cruz	10,251
Merida	40,994
Mexico City	10,666
Orizaba	27,334
San Andres	10,861
Santa Anita	11,834
Santa Fe	4,717
Tixapan	12,440
Guadalajara	2,401
Grand Total	226,838

The coordinate for this measure cell is **Customer_Lookup[customer_city] = “Hidalgo”**

- Given this coordinate, Excel filters down to the “Hidalgo” rows in the **Customer_Lookup** table, filters all *related* tables (based on the relationships in data model), then evaluates the arithmetic in the table defined by the measure (*in this case Total Quantity equals the sum of quantity from the transactions data table*)

This cell does NOT add up the values above it (*it's an island, remember?*)

- Total rows represent a **lack of filters**; since this cell does *not* have a customer_city coordinate, it evaluates the Total Quantity measure across the entire, unfiltered Customer_Lookup table

FILTER CONTEXT EXAMPLES

customer_id	All	<input type="button" value="▼"/>
Year	1997	<input type="button" value="▼"/>
Month	All	<input type="button" value="▼"/>
customer_country	USA	<input type="button" value="▼"/>

customer_city	Total Quantity
Albany	6,806
Altadena	2,574
Anacortes	766

Cell coordinates:

- Calendar_Lookup[Year] = **1997**
- Customer_Lookup[customer_country] = "**USA**"
- Customer_Lookup[customer_city] = "**Altadena**"

customer_country	<input type="button" value="▼"/>	<input type="button" value="X"/>
Canada		
Mexico		
USA		

Year	Quarter	Total Quantity
1997	1	266,773
1997	2	66,291
1997	3	62,610
1997	4	65,848
1997		72,024
1998	1	289,126
1998	2	69,785
1998	3	68,855
1998	4	68,574
1998		81,912
Grand Total		555,899

Cell coordinates:

- Calendar_Lookup[Year] = **1997**
- Customer_Lookup[customer_country] = "**USA**"

Cell coordinates:

- Calendar_Lookup[Year] = **1998**
- Calendar_Lookup[Quarter] = **1**
- Customer_Lookup[customer_country] = "**USA**"

Cell coordinates:

- Customer_Lookup[customer_country] = "**USA**"

store_country	Canada	<input type="button" value="▼"/>
---------------	--------	----------------------------------

Total Quantity	store_city	Vancouver	Victoria	Grand Total
product_brand	<input type="button" value="▼"/>			
ADJ		30	10	40
Akron		50	15	65
American		334	103	487
Amigo		50	31	81

Cell coordinates:

- Store_Lookup[store_country] = "**Canada**"
- Store_Lookup[store_city] = "**Vancouver**"
- Product_Lookup[product_brand] = "**Akron**"

Cell coordinates:

- Store_Lookup[store_country] = "**Canada**"
- Product_Lookup[product_brand] = "**Amigo**"

STEP-BY-STEP MEASURE CALCULATION

Row Labels	Total Quantity
CANADA	50,752
MEXICO	226,838
USA	555,899
Grand Total	833,489

• How *exactly* is this measure calculated?

- **REMEMBER:** This all happens *instantly* behind the scenes, every time a measure cell calculates

STEP 1

Detect pivot coordinates & apply filter context



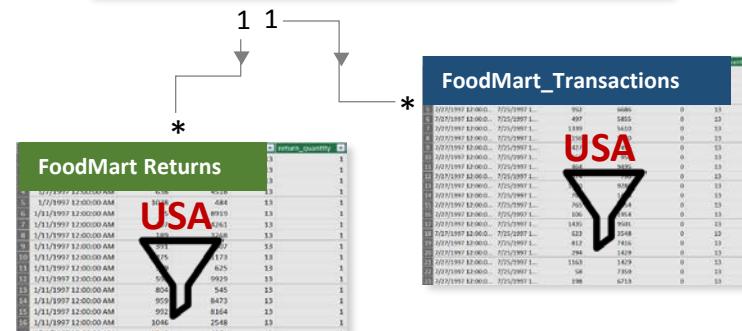
Row Labels	Total Quantity
CANADA	50,752
MEXICO	226,838
USA	555,899
Grand Total	833,489

Store_Lookup[store_country] = "USA"

Store_Lookup Table				Region	Address	City	State	Zip	Country
5	11	21	22	Supermarket	Store 13	5271 Holland Circle	Bellevue	98004	USA
6	12	23	24	Small Super.	Store 14	3219 Valley Dr.	Seattle	98103	USA
7	13	25	26	Small Super.	Store 14	4565 Indigo Ct	Los Angeles	90040	USA
8	14	27	28	Small Grocery	Store 14	5005 Highland Driv	Portland	97203	USA
9	15	29	30	Supermarket	Store 16	9522 La Salle Ct	Spokane	99205	USA
10	16	31	32	Supermarket	Store 16	4700 Rainier Rd	Tacoma	98408	USA
11	17	33	34	Supermarket	Store 17	9600 Julian Loop	Walla Walla	99362	USA
12	18	35	36	Small Super.	Store 17	3907 Main Court	Yakima	98901	USA
13	19	37	38	Small Super.	Store 18	1234 Main St	Spokane	99201	USA
14	20	39	40	Small Super.	Store 18	4567 Main St	Seattle	98101	USA
15	21	41	42	Small Super.	Store 19	5678 Main St	Portland	97201	USA
16	22	43	44	Small Super.	Store 19	9876 Main St	Bellevue	98001	USA
17	23	45	46	Small Super.	Store 20	1234 Main St	Seattle	98101	USA
18	24	47	48	Small Super.	Store 20	4567 Main St	Portland	97201	USA
19	25	49	50	Small Super.	Store 21	5678 Main St	Bellevue	98001	USA
20	26	51	52	Small Super.	Store 21	9876 Main St	Seattle	98101	USA
21	27	53	54	Small Super.	Store 22	1234 Main St	Portland	97201	USA
22	28	55	56	Small Super.	Store 22	4567 Main St	Bellevue	98001	USA
23	29	57	58	Small Super.	Store 23	5678 Main St	Seattle	98101	USA
24	30	59	60	Small Super.	Store 23	9876 Main St	Portland	97201	USA
25	31	61	62	Small Super.	Store 24	1234 Main St	Bellevue	98001	USA
26	32	63	64	Small Super.	Store 24	4567 Main St	Seattle	98101	USA
27	33	65	66	Small Super.	Store 25	5678 Main St	Portland	97201	USA
28	34	67	68	Small Super.	Store 25	9876 Main St	Bellevue	98001	USA
29	35	69	70	Small Super.	Store 26	1234 Main St	Seattle	98101	USA
30	36	71	72	Small Super.	Store 26	4567 Main St	Portland	97201	USA
31	37	73	74	Small Super.	Store 27	5678 Main St	Bellevue	98001	USA
32	38	75	76	Small Super.	Store 27	9876 Main St	Seattle	98101	USA
33	39	77	78	Small Super.	Store 28	1234 Main St	Portland	97201	USA
34	40	79	80	Small Super.	Store 28	4567 Main St	Bellevue	98001	USA
35	41	81	82	Small Super.	Store 29	5678 Main St	Seattle	98101	USA
36	42	83	84	Small Super.	Store 29	9876 Main St	Portland	97201	USA
37	43	85	86	Small Super.	Store 30	1234 Main St	Bellevue	98001	USA
38	44	87	88	Small Super.	Store 30	4567 Main St	Seattle	98101	USA
39	45	89	90	Small Super.	Store 31	5678 Main St	Portland	97201	USA
40	46	91	92	Small Super.	Store 31	9876 Main St	Bellevue	98001	USA
41	47	93	94	Small Super.	Store 32	1234 Main St	Seattle	98101	USA
42	48	95	96	Small Super.	Store 32	4567 Main St	Portland	97201	USA
43	49	97	98	Small Super.	Store 33	5678 Main St	Bellevue	98001	USA
44	50	99	100	Small Super.	Store 33	9876 Main St	Seattle	98101	USA
45	51	101	102	Small Super.	Store 34	1234 Main St	Portland	97201	USA
46	52	103	104	Small Super.	Store 34	4567 Main St	Bellevue	98001	USA
47	53	105	106	Small Super.	Store 35	5678 Main St	Seattle	98101	USA
48	54	107	108	Small Super.	Store 35	9876 Main St	Portland	97201	USA
49	55	109	110	Small Super.	Store 36	1234 Main St	Bellevue	98001	USA
50	56	111	112	Small Super.	Store 36	4567 Main St	Seattle	98101	USA
51	57	113	114	Small Super.	Store 37	5678 Main St	Portland	97201	USA
52	58	115	116	Small Super.	Store 37	9876 Main St	Bellevue	98001	USA
53	59	117	118	Small Super.	Store 38	1234 Main St	Seattle	98101	USA
54	60	119	120	Small Super.	Store 38	4567 Main St	Portland	97201	USA
55	61	121	122	Small Super.	Store 39	5678 Main St	Bellevue	98001	USA
56	62	123	124	Small Super.	Store 39	9876 Main St	Seattle	98101	USA
57	63	125	126	Small Super.	Store 40	1234 Main St	Portland	97201	USA
58	64	127	128	Small Super.	Store 40	4567 Main St	Bellevue	98001	USA
59	65	129	130	Small Super.	Store 41	5678 Main St	Seattle	98101	USA
60	66	131	132	Small Super.	Store 41	9876 Main St	Portland	97201	USA
61	67	133	134	Small Super.	Store 42	1234 Main St	Bellevue	98001	USA
62	68	135	136	Small Super.	Store 42	4567 Main St	Seattle	98101	USA
63	69	137	138	Small Super.	Store 43	5678 Main St	Portland	97201	USA
64	70	139	140	Small Super.	Store 43	9876 Main St	Bellevue	98001	USA
65	71	141	142	Small Super.	Store 44	1234 Main St	Seattle	98101	USA
66	72	143	144	Small Super.	Store 44	4567 Main St	Portland	97201	USA
67	73	145	146	Small Super.	Store 45	5678 Main St	Bellevue	98001	USA
68	74	147	148	Small Super.	Store 45	9876 Main St	Seattle	98101	USA
69	75	149	150	Small Super.	Store 46	1234 Main St	Portland	97201	USA
70	76	151	152	Small Super.	Store 46	4567 Main St	Bellevue	98001	USA
71	77	153	154	Small Super.	Store 47	5678 Main St	Seattle	98101	USA
72	78	155	156	Small Super.	Store 47	9876 Main St	Portland	97201	USA
73	79	157	158	Small Super.	Store 48	1234 Main St	Bellevue	98001	USA
74	80	159	160	Small Super.	Store 48	4567 Main St	Seattle	98101	USA
75	81	161	162	Small Super.	Store 49	5678 Main St	Portland	97201	USA
76	82	163	164	Small Super.	Store 49	9876 Main St	Bellevue	98001	USA
77	83	165	166	Small Super.	Store 50	1234 Main St	Seattle	98101	USA
78	84	167	168	Small Super.	Store 50	4567 Main St	Portland	97201	USA
79	85	169	170	Small Super.	Store 51	5678 Main St	Bellevue	98001	USA
80	86	171	172	Small Super.	Store 51	9876 Main St	Seattle	98101	USA
81	87	173	174	Small Super.	Store 52	1234 Main St	Portland	97201	USA
82	88	175	176	Small Super.	Store 52	4567 Main St	Bellevue	98001	USA
83	89	177	178	Small Super.	Store 53	5678 Main St	Seattle	98101	USA
84	90	179	180	Small Super.	Store 53	9876 Main St	Portland	97201	USA
85	91	181	182	Small Super.	Store 54	1234 Main St	Bellevue	98001	USA
86	92	183	184	Small Super.	Store 54	4567 Main St	Seattle	98101	USA
87	93	185	186	Small Super.	Store 55	5678 Main St	Portland	97201	USA
88	94	187	188	Small Super.	Store 55	9876 Main St	Bellevue	98001	USA
89	95	189	190	Small Super.	Store 56	1234 Main St	Seattle	98101	USA
90	96	191	192	Small Super.	Store 56	4567 Main St	Portland	97201	USA
91	97	193	194	Small Super.	Store 57	5678 Main St	Bellevue	98001	USA
92	98	195	196	Small Super.	Store 57	9876 Main St	Seattle	98101	USA
93	99	197	198	Small Super.	Store 58	1234 Main St	Portland	97201	USA
94	100	199	200	Small Super.	Store 58	4567 Main St	Bellevue	98001	USA
95	101	201	202	Small Super.	Store 59	5678 Main St	Seattle	98101	USA
96	102	203	204	Small Super.	Store 59	9876 Main St	Portland	97201	USA
97	103	205	206	Small Super.	Store 60	1234 Main St	Bellevue	98001	USA
98	104	207	208	Small Super.	Store 60	4567 Main St	Seattle	98101	USA
99	105	209	210	Small Super.	Store 61	5678 Main St	Portland	97201	USA
100	106	211	212	Small Super.	Store 61	9876 Main St	Bellevue	98001	USA
101	107	213	214	Small Super.	Store 62	1234 Main St	Seattle	98101	USA
102	108	215	216	Small Super.	Store 62	4567 Main St	Portland	97201	USA
103	109	217	218	Small Super.	Store 63	5678 Main St	Bellevue	98001	USA
104	110	219	220	Small Super.	Store 63	9876 Main St	Seattle	98101	USA
105	111	221	222	Small Super.	Store 64	1234 Main St	Portland	97201	USA
106	112	223	224	Small Super.	Store 64	4567 Main St	Bellevue	98001	USA
107	113	225	226	Small Super.	Store 65	5678 Main St	Seattle	98101	USA
108	114	227	228	Small Super.	Store 65	9876 Main St	Portland	97201	USA
109	115	229	230	Small Super.	Store 66	1234 Main St	Bellevue	98001	USA
110	116	231	232	Small Super.	Store 66	4567 Main St	Seattle	98101	USA
111	117	233	234	Small Super.	Store 67	5678 Main St	Portland	97201	USA
112	118	235	236	Small Super.	Store 67	9876 Main St	Bellevue	98001	USA
113	119	237	238	Small Super.	Store 68	1234 Main St	Seattle	98101	USA
114	120	239	240	Small Super.	Store 68	4567 Main St	Portland	97201	USA
115	121	241	242	Small Super.	Store 69	5678 Main St	Bellevue	98001	USA
116	122	243	244	Small Super.	Store 69	9876 Main St	Seattle	98101	USA
117	123	245	246	Small Super.	Store 70	1234 Main St	Portland	97201	USA
118	124	247	248	Small Super.	Store 70	4567 Main St	Bellevue	98001	USA
119	125	249	250	Small Super.	Store 71	5678 Main St	Seattle	98101	USA
120	126	251	252	Small Super.	Store 71	9876 Main St	Portland	97201	USA
121	127	253	254	Small Super.	Store 72	1234 Main St	Bellevue	98001	USA
122	128	255	256	Small Super.	Store 72	4567 Main St	Seattle	98101	USA
123	129	257	258	Small Super.	Store 73	5678 Main St	Portland	97201	USA
124	130	259	260	Small Super.	Store 73	9876 Main St	Bellevue	98001	USA
125	131	261	262	Small Super.	Store 74	1234 Main St	Seattle	98101	USA
126	132	263	264	Small Super.	Store 74	4567 Main St	Portland	97201	USA
127	133	265	266	Small Super.	Store 75	5678 Main St	Bellevue	98001	USA
128	134	267	268	Small Super.	Store 75	9876 Main St	Seattle	98101	USA
129	135	269	270	Small Super.	Store 76	1234 Main St	Portland	97201	USA
130	136	271	272	Small Super.	Store 76	4567 Main St	Bellevue	98001	USA
131	137	273	274	Small Super.	Store 77	5678 Main St	Seattle	98101	USA
132	138	275	276	Small Super.	Store 77	9876 Main St	Portland	97201	USA
133	139	277	278	Small Super.	Store 78	1234 Main St	Bellevue	98001	USA
134	140	279	280	Small Super.	Store 78	4567 Main St	Seattle	98101	USA
135	141	281	282	Small Super.	Store 79	5678 Main St	Portland	97201	USA
136	142	283	284	Small Super.	Store 79	9876 Main St	Bellevue	98001	USA
137	143	285	286	Small Super.	Store 80	1234 Main St	Seattle	98101	USA
138	144	287	288	Small Super.	Store 80	4567 Main St	Portland	97201	USA
139	145	289	290	Small Super.	Store 81	5678 Main St	Bellevue	98001	USA
140	146	291	292	Small Super.	Store 81	9876 Main St	Seattle	98101	USA
141	147	293	294	Small Super.	Store 82	1234 Main St	Portland	97201	USA
142	148	295	296	Small Super.	Store 82	4567 Main St	Bellevue	98001	USA
143	149	297	298	Small Super.	Store 83	5678 Main St	Seattle	98101	USA
144	150	299	300	Small Super.	Store 83	9876 Main St	Portland	97201	USA
145	151	301	302	Small Super.	Store 84	1234 Main St	Bellevue	98001	USA
146	152	303	304	Small Super.	Store 84	4567 Main St	Seattle	98101	USA
147	153	305	306	Small Super.	Store 85	5678 Main St	Portland	97201	USA
148	154	307							

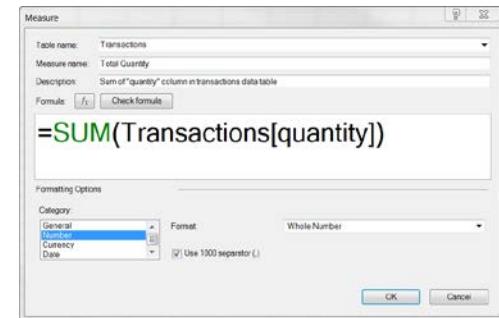
STEP 2

Carry filters “downstream” & apply to all related tables



STEP 3

*Evaluate the measure formula
against the filtered table*



Sum of Transactions[quantity] when store_country = "USA" = **555,899**

RECAP: CALCULATED COLUMNS VS. MEASURES

CALCULATED COLUMNS

- Evaluated in the context of each row of the table to which it belongs (has **row context**)
 - Appends static values to each row in a table and stores them in the model, increasing file size
 - Only recalculated on data source refresh or changes to component columns
 - Primarily used as **rows**, **columns**, **slicers** or **filters**

Calculated columns “live” in tables



MEASURES

- Evaluated in the context of each cell of the PivotTable in which it is displayed (has **filter context**)
 - Does not create new data in the tables themselves, and does not increase file size
 - Recalculated in response to any change in the PivotTable view
 - Can *only* be used as PivotTable **values**

store_country	Canada		
Total Quantity	store_city		
product_brand	Vancouver	Victoria	Grand Total
ADJ	30	10	40
Akron	50	15	65
American	384	103	487
Amigo	50	31	81

*Measures “live” in **PivotTables***



***Note:** Calculated columns CAN be placed in the values area of a pivot, but you can (and should) use a measure instead.

POWER PIVOT BEST PRACTICES



Avoid using implicit measures whenever possible

- *Implicit measures are limited in functionality and restricted to the pivot in which they were created; explicit measures are more **portable** and **powerful***



Don't use a calculated column when a measure will do the trick

- *Only use calculated columns to “stamp” static, fixed values to each row in a table*
- *Use measures when aggregation is necessary, or to create dynamic values in a pivot*



Know your data model inside and out!

- *It's easy to produce incorrect results in Power Pivot if you don't respect the model's table relationships, and errors are often difficult to spot without a thorough QA*

COMMON DAX FUNCTIONS

DAX SYNTAX

MEASURE NAME

- Note: Measures are always surrounded in 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 for each row)
 - In a measure, =Transactions[quantity] will return an error since Excel doesn't know how to evaluate that as a single value in a pivot (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:

For **column** references, use the fully qualified name (i.e. Table[Column])

For **measure** references, just use the measure name (i.e. [Measure])



DAX OPERATORS

Arithmetic Operator	Meaning	Example
+	Addition	$2 + 7$
-	Subtraction	$5 - 3$
*	Multiplication	$2 * 6$
/	Division	$4 / 2$
\wedge	Exponent	$2 \wedge 5$

Hey! Pay attention to these!

Comparison Operator	Meaning	Example
=	Equal to	$[City] = "Boston"$
>	Greater than	$[Quantity] > 10$
<	Less than	$[Quantity] < 10$
\geq	Greater than or equal to	$[Unit_Price] \geq 2.5$
\leq	Less than or equal to	$[Unit_Price] \leq 2.5$
\neq	Not equal to	$[Country] \neq "Mexico"$

Text/Logical Operator	Meaning	Example
&	Concatenates two values to produce one text string	$[City] & " " & [State]$
&&	Create an AND condition between two logical expressions	$([State] = "MA") \&& ([Quantity] > 10)$
 (double pipe)	Create an OR condition between two logical expressions	$([State] = "MA") ([State] = "CT")$
IN	Creates a logical OR condition based on a given list (using curly brackets)	<code>'Store Lookup'[State] IN { "MA", "CT", "NY" }</code>

*Head to www.msdn.microsoft.com for more information about DAX syntax, operators, troubleshooting, etc.

COMMON FUNCTION CATEGORIES

MATH & STATS Functions

*Basic aggregation functions as well as “**iterators**” evaluated at the row-level*

Common Examples:

- SUM
- AVERAGE
- MAX/MIN
- DIVIDE
- COUNT/COUNTA
- COUNTROWS
- DISTINCTCOUNT

Iterator Functions:

- SUMX
- AVERAGEX
- MAXX/MINX
- RANKX
- COUNTX

LOGICAL Functions

*Functions for returning information about values in a given **conditional expression***

Common Examples:

- IF
- IFERROR
- AND
- OR
- NOT
- SWITCH
- TRUE
- FALSE

TEXT Functions

*Functions to manipulate **text strings** or **control formats** for dates, times or numbers*

Common Examples:

- CONCATENATE
- FORMAT
- LEFT/MID/RIGHT
- UPPER/LOWER
- PROPER
- LEN
- SEARCH/FIND
- REPLACE
- REPT
- SUBSTITUTE
- TRIM
- UNICHAR

FILTER Functions

*Lookup functions based on related tables and **filtering** functions for dynamic calculations*

Common Examples:

- CALCULATE
- FILTER
- ALL
- ALLEXCEPT
- RELATED
- RELATEDTABLE
- DISTINCT
- VALUES
- EARLIER/EARLIEST
- HASONEVALUE
- HASONEFILTER
- ISFILTERED
- USERELATIONSHIP

DATE & TIME Functions

*Basic **date and time** functions as well as advanced **time intelligence** operations*

Common Examples:

- DATEDIFF
- YEARFRAC
- YEAR/MONTH/DAY
- HOUR/MINUTE/SECOND
- TODAY/NOW
- WEEKDAY/WEEKNUM

Time Intelligence Functions:

- DATESYTD
- DATESQTD
- DATESMTD
- DATEADD
- DATESINPERIOD

**Note: This is NOT a comprehensive list (does not include trigonometry functions, parent/child functions, information functions, or other less common functions)*

BASIC MATH & STATS FUNCTIONS

SUM()

Evaluates the sum of a column

=**SUM**(<column>)

AVERAGE()

Returns the average (arithmetic mean) of all the numbers in a column

=**AVERAGE**(<column>)

MAX()

Returns the largest value in a column or between two scalar expressions

=**MAX**(<column>) or =**MAX**(<exp1>, <exp2>)

MIN()

Returns the smallest value in a column or between two scalar expressions

=**MIN**(<column>) or =**MIN**(<exp1>, <exp2>)

DIVIDE()

Performs division and returns the alternate result (or blank) if div/0

=**DIVIDE**(<numerator>, <denominator>, <other>)

BASIC MATH & STATS FUNCTIONS (EXAMPLES)

Sum of quantity from the Transactions table

Measure

Table name: Transactions
Measure name: Total Quantity
Description: Sum of "quantity" column in transactions data table
Formula: `fx` `Check formula`

=SUM(Transactions[quantity])

Formatting Options

Category: General
Number
Currency
Date
Format: Whole Number
Symbol: \$
Decimal places: 2
Use 1000 separator (.)

OK Cancel

Average of product_retail_price

Measure

Table name: Product_Lookup
Measure name: Avg Retail Price
Description:
Formula: `fx` `Check formula`

=AVERAGE(Product_Lookup[product_retail_price])

Formatting Options

Category: General
Number
Currency
Date
Format: Whole Number
Symbol: \$
Decimal places: 2
Use 1000 separator (.)

OK Cancel

Quantity Returned divided by Total Quantity

Measure

Table name: Returns
Measure name: Return Rate
Description:
Formula: `fx` `Check formula`

=DIVIDE([Quantity Returned], [Total Quantity])

Formatting Options

Category: General
Number
Currency
Date
Format: Percentage
Decimal places: 1
Use 1000 separator (.)

OK Cancel

PRO TIP:



*Even though it might seem unnecessary, **creating measures for even simple calculations** (like the sum of a column) allows you to use those measures within other calculations, anywhere in the workbook*

COUNT, COUNTA, DISTINCTCOUNT & COUNTROWS

COUNTROWS()

Counts the number of rows in the specified table, or a table defined by an expression

=**COUNTROWS(<table>)**

COUNT()

Counts the number of cells in a column that contain numbers

=**COUNT(<column>)**

COUNTA()

Counts the number of non-empty cells in a column (numerical and non-numerical)

=**COUNTA(<column>)**

DISTINCTCOUNT()

Counts the number of different cells in a column of numbers

=**DISTINCTCOUNT(<column>)**

COUNT FUNCTIONS (EXAMPLES)

Count of **all rows** in the Transactions table

Measure

Table name: Transactions
Measure name: Transactions
Description: Number of rows in Transactions table
Formula: `fx` Check formula

=COUNTRows(Transactions)

Formatting Options

Category: General Number Currency Date Format: Whole Number Use 1000 separator (.)

OK

Count of **unique values** in the product_id column

Measure

Table name: Product_Lookup
Measure name: Unique Products
Description: Distinct count of product IDs
Formula: `fx` Check formula

=DISTINCTCOUNT(Product_Lookup[product_id])

Formatting Options

Category: General Number Currency Date Format: Whole Number Use 1000 separator (.)

OK Cancel

Count of **non-empty cells** in the recyclable column

Measure

Table name: Product_Lookup
Measure name: Recyclable Products
Description: Counts products where "recyclable" field is non-empty
Formula: `fx` Check formula

=COUNTA(Product_Lookup[recyclable])

Formatting Options

Category: General Number Currency Date Format: Whole Number Use 1000 separator (.)

OK Cancel

BASIC LOGICAL FUNCTIONS (IF/AND/OR)

IF()

Checks if a given condition is met, and returns one value if the condition is TRUE, and another if the condition is FALSE

=IF(<logical test>, <value_if_true>, <value_if_false>)

IFERROR()

Evaluates an expression and returns a specified value if the expression returns an error, otherwise returns the expression itself

=IFERROR(value, value_if_error)

AND()

Checks whether both arguments are TRUE, and returns TRUE if both arguments are TRUE, otherwise returns FALSE

=AND(<logical1>, <logical2>)

OR()

Checks whether one of the arguments is TRUE to return TRUE, and returns FALSE if both arguments are FALSE

=OR (<logical1>, <logical2>)

Note: Use the **&&** and **||** operators if you want to include more than two conditions!

BASIC LOGICAL FUNCTIONS (EXAMPLES)

Education level equals "**Grad**" if customer has a bachelors degree or a graduate degree, otherwise "**Non-Grad**"

home	education	acct_open_date	member_card	occupation	homeowner	full_name	birth_year	has_children	customer_age	education_level	customer_priority	Add Column
1	0 High School ...	11/16/1994 12:00... Bronze	Manual	N	Bertha Jam...	1948	Y		69	Non-Grad	Other	
2	0 High School ...	5/5/1992 12:00:0... Bronze	Manual	N	Ole Weldon	1931	Y		86	Non-Grad	Other	
3	0 High School ...	6/26/1994 12:00:... Bronze	Manual	N	Paul Alcorn	1973	Y		44	Non-Grad	Other	
4	0 High School ...	2/9/1990 12:00:0... Bronze	Manual	N	Jared Busta...	1910	Y		107	Non-Grad	Other	
5	0 High School ...	3/15/1992 12:00:... Bronze	Manual	N	Margaret A...	1979	Y		35	Non-Grad	Other	
6	0 High School ...	3/2/1994 12:00:0... Bronze	Manual	N	Vanessa Ten...	1930	Y		87	Non-Grad	Other	
7	0 High School ...	6/4/1993 12:00:0... Bronze	Manual	N	Catherine ...	1966	Y		53	Non-Grad	Other	
8	0 High School ...	3/5/1992 12:00:0... Bronze	Manual	N	Stacey Cere...	1943	Y		74	Non-Grad	Other	
9	0 High School ...	5/17/1992 12:00:... Bronze	Manual	N	Marlin Coriell	1933	Y					
10	0 High School ...	9/8/1992 12:00:0... Bronze	Manual	N	Deanna Sab...	1916	Y					
11	0 High School ...	2/21/1991 12:00:... Bronze	Manual	N	Joseph Tho...	1968	Y					
12	0 High School ...	6/12/1994 12:00:... Bronze	Manual	N	Roberta Stu...	1919	Y					

Supermarket_size equals "**Large**" if sq ft >30,000, otherwise "**Small**"

address	store_city	store_state	full_store_address	store_country	store_phone	area_code	first_opened_date	last_remodel_date	total_sqft	grocery_sqft	supermarket_size	Add Column
1	Acapulco	Guerrero	2853 Bailey Rd, Acap...	MEXICO	262-555-5124	262	1/9/1982 12:00:00...	12/5/1990 12:00:00...	23593	17475	Small	
2	Two Way	Bellingham	5203 Catanzaro Way...	USA	605-555-8203	605	4/2/1970 12:00:00...	6/4/1973 12:00:00...	28206	22271	Small	
3	Circle	Bremerton	1501 Ramsey Circle, ...	USA	509-555-1596	509	6/14/1959 12:00:00...	11/19/1967 12:00:00...	39696	24390	Large	
4	Dr	Camacho	433 St George Dr, Ca...	MEXICO	304-555-1474	304	9/27/1994 12:00:00...	12/1/1995 12:00:00...	23759	16844	Small	
5	drive	Guadalajara	1250 Coggins Drive, ...	MEXICO	801-555-4324	801	9/18/1978 12:00:00...	6/29/1991 12:00:00...	24597	15017	Small	
6	Canyon R...	Beverly Hills	5495 Mitchell Canyon...	USA	958-555-5002	958	1/3/1981 12:00:00...	3/13/1991 12:00:00...	23688	15337	Small	
7	ive	Los Angeles	1077 Wharf Drive, L...	USA	477-555-7967	477	5/21/1971 12:00:00...	10/20/1981 12:00:00...	23598	14210	Small	
8	sta Ave	Merida	3173 Buena Vista Av...	MEXICO	797-555-3417	797	9/23/1958 12:00:00...	11/18/1967 12:00:00...	30797	20141	Large	
9	Road	Mexico City	1872 El Pintado Roa...	MEXICO	439-555-3524	439	3/18/1955 12:00:00...	6/7/1959 12:00:00...	36509	22450	Large	
10	m Dr	Orizaba	7894 Rotherham Dr., MEXICO	MEXICO	212-555-4774	212	4/13/1979 12:00:00...	1/30/1982 12:00:00...	34791	26354	Large	
11	circle	Portland	5371 Holland Circle, ...	USA	685-555-8995	685	9/17/1976 12:00:00...	5/15/1982 12:00:00...	20319	16232	Small	
12	ter Pl	Hidalgo	1120 Westchester Pl...	MEXICO	151-555-1702	151	3/25/1968 12:00:00...	12/18/1993 12:00:00...	30584	21938	Large	
13	e	Salem	5179 Valley Ave, Sal...	USA	977-555-2724	977	4/13/1957 12:00:00...	11/10/1997 12:00:00...	27694	18670	Small	
14	San Francisco	CA	4365 Indigo Ct, San ...	USA	135-555-4888	135	11/24/1957 12:00:00...	1/7/1958 12:00:00...	22478	15321	Small	
15	Drive	Seattle	5006 Highland Drive...	USA	893-555-1024	893	7/24/1963 12:00:00...	10/19/1973 12:00:00...	21215	13305	Small	
16	t	Spokane	5922 La Salle Ct, Spo...	USA	643-555-3645	643	8/23/1974 12:00:00...	7/13/1977 12:00:00...	30268	22063	Large	
17	nd	Tacoma	490 Risdon Road, Ta...	USA	855-555-5581	855	5/30/1970 12:00:00...	6/23/1976 12:00:00...	33858	22123	Large	
18	d	Hidalgo	6764 Glen Road, Hid...	MEXICO	528-555-8317	528	6/28/1969 12:00:00...	8/30/1975 12:00:00...	38382	30351	Large	
19	Drive	Vancouver	6644 Sundance Drive,...	CANADA	862-555-7395	862	3/27/1977 12:00:00...	10/25/1990 12:00:00...	23112	16418	Small	
20	Ln	Victoria	3706 Marquelle Ln, Vi...	CANADA	897-555-1931	897	2/6/1980 12:00:00...	4/9/1987 12:00:00 ...	34452	27463	Large	

SWITCH & SWITCH(TRUE)

SWITCH()

Evaluates an expression against a list of values and returns one of multiple possible result expressions

=SWITCH(<expression>, <value1>, <result1>, <value2>, <result2>, ... <else>)

Any DAX expression that returns a single scalar value, evaluated multiple times (for each row/constant)

Examples:

- Calendar_Lookup[month_num]
- Product_Lookup[product_brand]



PRO TIP:

Use the SWITCH(TRUE()) combo to generate results based on Boolean (True/False) expressions (instead of those pesky nested IF statements!)

List of **values** produced by the expression, each paired with a **result** to return for rows/cases that match

Examples:

```
=SWITCH(Calendar_Lookup[month_num],  
    1, "January",  
    2, "February",  
    etc...
```

```
=SWITCH(TRUE(),  
    [retail_price]<5, "Low Price",  
    AND([retail_price]>=5, [retail_price]<20), "Med Price",  
    AND([retail_price]>=20, [retail_price]<50), "High Price"  
    "Premium Price")
```

SWITCH & SWITCH(TRUE) (EXAMPLES)

Switch quarter 1 with "Q1", quarter 2 with "Q2", quarter 3 = "Q3", else "Q4"

The screenshot shows a Microsoft Excel interface with the 'Home' tab selected. In the formula bar, the formula is displayed as: `=SWITCH(Calendar_Lookup[quarter], 1,"Q1", 2,"Q2", 3,"Q3", "Q4")`. The table below contains data from July 1997, with the 'quarter_name' column highlighted and a red box around the value 'Q3' in the third row.

	date	year	quarter	month	month_name	week_of_year	start_of_week	day	weekday	weekend	quarter_name	Add Column
1	7/1/1997 12:00:00	1997	3	7	July		27	6/30/1997 12:00:00	1	Tuesday	0	Q3
2	7/2/1997 12:00:00	1997	3	7	July		27	6/30/1997 12:00:00	2	Wednesday	0	Q3
3	7/3/1997 12:00:00	1997	3	7	July		27	6/30/1997 12:00:00	3	Thursday	0	Q3
4	7/4/1997 12:00:00	1997	3	7	July		27	6/30/1997 12:00:00	4	Friday	0	Q3
5	7/5/1997 12:00:00	1997	3	7	July		27	6/30/1997 12:00:00	5	Saturday	1	Q3
6	7/6/1997 12:00:00	1997	3	7	July		28	6/30/1997 12:00:00	6	Sunday	1	Q3
7	7/7/1997 12:00:00	1997	3	7	July							
8	7/8/1997 12:00:00	1997	3	7	July							
9	7/9/1997 12:00:00	1997	3	7	July							
10	7/10/1997 12:00:00	1997	3	7	July							
11	7/11/1997 12:00:00	1997	3	7	July							
12	7/12/1997 12:00:00	1997	3	7	July							

Set product_price_category to "High" if retail price > \$3, "Medium" if price is between \$2 and \$3, "Low" if price is <=\$2, else "Other"

The screenshot shows a Microsoft Excel interface with the 'Home' tab selected. In the formula bar, the formula is displayed as: `=SWITCH(TRUE(), Product_Lookup[product_retail_price]>3, "High", Product_Lookup[product_retail_price]>2 && Product_Lookup[product_retail_price] <=3, "Medium", Product_Lookup[product_retail_price]<=2, "Low", "Other")`. The table below contains product data, with the 'product_price_category' column highlighted and a red box around the value 'Medium' in the first row.

	produ...	product_brand	product_name	product_sku	product_retail_price	discount_retail_price	product_cost	product_weight	recyclable	low_fat	product_price_category	Add Column
1	1	Washington	Washington Berr...	90748583674	\$2.85	\$2.28	\$0.94	8.39			Medium	
2	2	Washington	Washington Ma...	96516502499	\$0.74	\$0.59	\$0.26	7.42		1	Low	
3	3	Washington	Washington Stra...	58427771925	\$0.83	\$0.66	\$0.40	13.1	1	1	High	
4	4	Washington	Washington Cre...	64412155747	\$3.64	\$2.91	\$1.64	10.6	1			
5	5	Washington	Washington Diet...	85561191439	\$2.19	\$1.75	\$0.77	6.66	1		Medium	
6	6	Washington	Washington Cola	29804642796	\$1.15	\$0.92	\$0.37	15.8			Low	
7	7	Washington	Washington Diet...	20191444754	\$2.61	\$2.09	\$0.91	18	1		Medium	
8	8	Washington	Washington Ora...	89770532250	\$2.59	\$2.07	\$0.80	8.97	1		Medium	

TEXT FUNCTIONS

LEN()	Returns the number of characters in a string
CONCATENATE()	Joins two text strings into one
LEFT/MID/ RIGHT()	Returns a number of characters from the start/middle/end of a text string
UPPER/LOWER/ PROPER()	Converts letters in a string to upper/lower/proper case
SUBSTITUTE()	Replaces an instance of existing text with new text in a string
SEARCH()	Returns the position where a specified string or character is found, reading left to right

=LEN(<text>)

Note: Use the & operator as a shortcut, or to combine more than two strings!

=CONCATENATE(<text1>, <text2>)

=LEFT/RIGHT(<text>, <num_chars>)

=MID(<text>, <start_num>, <num_chars>)

=UPPER/LOWER/PROPER(<text>)

=SUBSTITUTE(<text>, <old_text>, <new_text>, <instance>)

=SEARCH(<find_text>, <within_text>, <start_num>, <NotFoundValue>)

TEXT FUNCTIONS (EXAMPLES)

Extract the **left 3 characters** from each value in the store_country column

[country_a...]									country_abbreviat...	store_street
1 Alley Rd, Acap...	MEXICO	262-555-5124	262	1/9/1982 12:00:00...	12/5/1990 12:00:00...	23593	17475	Small	MEX	2853
2 Itanzaro Way...	USA	605-555-8203	605	4/2/1970 12:00:00...	6/4/1973 12:00:00...	28206	22271	Medium	USA	5203
3 amsey Circle, ...	USA	509-555-1596	509	6/14/1959 12:00:00...	11/19/1967 12:00:00...	39696	24390	Huge	USA	1501
4 George Dr, Ca...	MEXICO	304-555-1474	304	9/27/1994 12:00:00...	12/1/1995 12:00:00...	23759	16844	Small	MEX	433
5 oggin Drive, ...	MEXICO	801-555-4324	801	9/18/1978 12:00:00...	6/29/1991 12:00:00...	24597	15012	Small	MEX	1250
6 itchell Canyo...	USA	958-555-5002	958	1/3/1981 12:00:00...	3/13/1991 12:00:00...	23688	15337	Small	USA	5495
7 harl Drive, L...	USA	477-555-7967	477	5/21/1971 12:00:00...	10/20/1981 12:00:00...	23598	14210	Small	USA	1077
8 ena Vista Av...	MEXICO	797-555-3417	797	9/23/1958 12:00:00...	11/18/1967 12:00:00...	30797	20141	Large	MEX	3173
9 Pintado Roa...	MEXICO	439-555-3524	439	3/18/1955 12:00:00...	6/7/1959 12:00:00...	36509	22450	Huge	MEX	1872
10 otherham Dr, ...	MEXICO	212-555-4774	212	4/13/1979 12:00:00...	1/30/1982 12:00:00...	34791	26354	Large	MEX	7894
11 lland Circle, ...	USA	685-555-8995	685	9/17/1976 12:00:00...	5/15/1982 12:00:00...	20319	16232	Small	USA	5371
12 estchester Pl...	MEXICO	151-555-1702	151	3/25/1968 12:00:00...	12/18/1993 12:00:00...	30584	21938	Large	MEX	1120
13 alley Ave, Sal...	USA	977-555-2724	977	4/13/1957 12:00:00...	11/10/1997 12:00:00...	27694	18670	Medium	USA	5179
14 digo Ct, San ...	USA	135-555-4888	135	11/24/1957 12:00:00...	1/7/1958 12:00:00...	22478	15321	Small	USA	4365
15 ghland Drive...	USA	893-555-1024	893	7/24/1969 12:00:00...	10/19/1973 12:00:00...	21215	13305	Small	USA	5006
16 Salle Ct, Spo...	USA	643-555-3645	643	8/23/1974 12:00:00...	7/13/1977 12:00:00...	30268	22063	Large	USA	5922
17 don Road, Ta...	USA	855-555-5581	855	5/30/1970 12:00:00...	6/23/1976 12:00:00...	3231				
18 en Road, Hid...	MEXICO	528-555-8317	528	6/28/1969 12:00:00...	8/30/1975 12:00:00...	3231				
19 dence Drive,...	CANADA	862-555-7395	862	3/27/1977 12:00:00...	10/25/1990 12:00:00...	3231				
20 arvelle Ln, Vi...	CANADA	897-555-1931	897	2/6/1980 12:00:00...	4/9/1987 12:00:00...	3231				

Concatenate the values from the year and month columns

[date]	year	quarter	month	month_name	start_of_year	day	weekday	weekend	quarter_name	year_month
1 7/1/1997 12:00:00...	1997	3	3	July	27 6/30/1997 12:00:00...	1	Tuesday	0	Q3	19977
2 7/2/1997 12:00:00...	1997	3	4	July	27 6/30/1997 12:00:00...	2	Wednesday	0	Q3	19977
3 7/3/1997 12:00:00...	1997	3	5	July	27 6/30/1997 12:00:00...	3	Thursday	0	Q3	19977
4 7/4/1997 12:00:00...	1997	3	6	July	27 6/30/1997 12:00:00...	4	Friday	0	Q3	19977
5 7/5/1997 12:00:00...	1997	3	7	July	27 6/30/1997 12:00:00...	5	Saturday	1	Q3	19977
6 7/6/1997 12:00:00...	1997	3	8	July	28 6/30/1997 12:00:00...	6	Sunday	1	Q3	19977
7 7/7/1997 12:00:00...	1997	3	9	July	28 7/7/1997 12:00:00...	7	Monday	0	Q3	19977
8 7/8/1997 12:00:00...	1997	3	10	July	28 7/7/1997 12:00:00...	8	Tuesday	0	Q3	19977

Extract characters from the left of the customer_address column, up to the space

[customer_id]	acct_open_date	member_card	occupation	homeowner	full_name	birth_year	has_children	customer_age	education_level	customer_priority	customer_house_type
1	11/16/1994 12:00:00...	Bronze	Manual	N	Bertha Jam...	1948	Y	69	Non-Grad	Other	3029
2	5/5/1992 12:00:00...	Bronze	Manual	N	Ole Weldon	1931	Y	86	Non-Grad	Other	5754
3	6/26/1994 12:00:00...	Bronze	Manual	N	Paul Alcorn	1973	Y	44	Non-Grad	Other	4822
4	2/9/1990 12:00:00...	Bronze	Manual	N	Jared Busta...	1910	Y	107	Non-Grad	Other	4222
5	3/15/1992 12:00:00...	Bronze	Manual	N	Margaret A...	1979	Y	38	Non-Grad	Other	8452
6	3/2/1994 12:00:00...	Bronze	Manual	N	Vanessa Ten...	1930	Y	87	Non-Grad	Other	6621
7	6/4/1993 12:00:00...	Bronze	Manual	N	Catherine ...	1966	Y	51	Non-Grad	Other	1239
8	3/5/1992 12:00:00...	Bronze	Manual	N	Stacey Cere...	1943	Y	74	Non-Grad	Other	852
9	5/17/1992 12:00:00...	Bronze	Manual	N	Marlin Coriell	1933	Y	84	Non-Grad	Other	4824
10	9/8/1992 12:00:00...	Bronze	Manual	N	Deanna Sab...	1916	Y	101	Non-Grad	Other	8942
11	2/21/1991 12:00:00...	Bronze	Manual	N	Joseph Tho...	1968	Y	49	Non-Grad	Other	2099
12	6/12/1994 12:00:00...	Bronze	Manual	N	Roberta Stu...	1919	Y	98	Non-Grad	Other	2086

CALCULATE

CALCULATE()

Evaluates a given expression or formula under a set of defined filters

=CALCULATE(<expression>, <filter1>, <filter2>,...)



Name of an existing measure or a formula for a valid measure

Examples:

- [Total Transactions]
- SUM(Transactions[quantity])

*List of simple Boolean (True/False) filter expressions
(note: these require simple, fixed values; you cannot create filters based on measures)*

Examples:

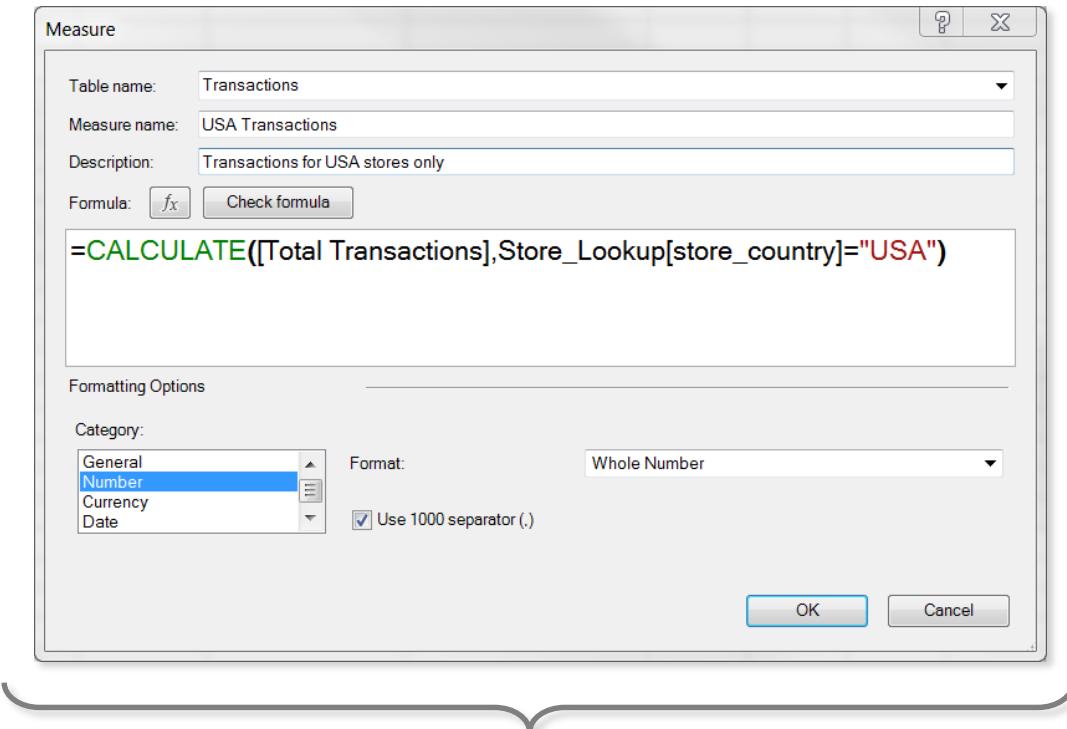
- Store_Lookup[store_country]=“USA”
- Calendar[Year]=1998
- Transactions[quantity]>=5

PRO TIP:



CALCULATE works just like **SUMIF** or **COUNTIF**, except it can evaluate measures based on ANY sort of calculation (not just a sum, count, etc); it may help to think of it like “**CALCULATEIF**”

CALCULATE (EXAMPLE)



In this case we've defined a new measure named "**USA Transactions**", which evaluates the "**Total Transactions**" measure when the store country equals "**USA**"

store_country	Total Transactions	USA Transactions
CANADA	16,091	180,823
MEXICO	72,806	180,823
USA	180,823	180,823
Grand Total	269,720	180,823

Why do we see the same repeating value when we add **store_country** to rows? Shouldn't these cells have filter contexts for Canada and Mexico?

HEY THIS IS IMPORTANT!

The CALCULATE function **modifies filters** and **overrules** any competing ones defined by the PivotTable coordinates!

In this example, the MEXICO cell has a filter context of `store_country= “MEXICO”` (*defined by the row label*) AND `store_country= “USA”` (*defined by the CALCULATE function*)

Both cannot be true at the same time, so the MEXICO filter is overwritten and CALCULATE takes priority

CALCULATE CHANGES THE FILTER CONTEXT

CALCULATE

Modify filters if measure contains CALCULATE

Store Lookup[store country] = "USA"

STEP 1

Detect pivot coordinates & apply filter context

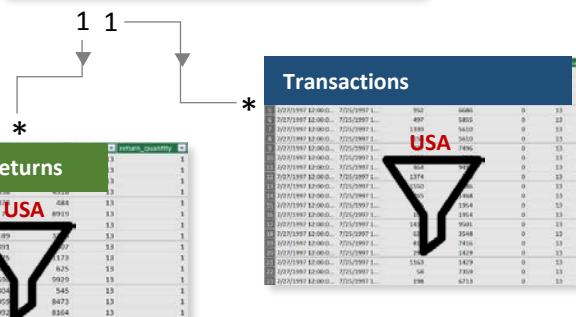
Store_Lookup Table			
StoreID	StoreName	Address	City
6	47 Discount Superstore	Street 1	Beverly Hills
7	Super Discount	Street 2	Los Angeles
20	Deluxe Superstore	Street 3	Merida
9	2 Mid-Size Grocer	Street 4	Mexico City
10	Supermarket	Street 5	D.F.
13	25 Discount Superstore	Street 6	Ventura
12	25 Deluxe Superstore	Street 7	Veracruz
13	25 Deluxe Superstore	Street 8	Winnipeg
14	10 Supermarket	Street 9	Zacatecas
18	Supermarket	Street 10	Salem
24	Supermarket	Street 11	Salinas
26	87 Supermarket	Street 12	San Francisco
17	84 Deluxe Superstore	Street 13	Seattle
21	10 Supermarket	Street 14	Tacoma
25	10 Supermarket	Street 15	Toronto
5	5 Deluxe Superstore	Street 16	Vancouver
6	8 Mid-Size Grocer	Street 17	Victoria
20	Supermarket	Street 18	Victoria
31	10 Deluxe Superstore	Street 19	San Antonio
22	48 Supermarket	Street 20	Santa Barbara
23	89 Mid-Size Grocer	Street 21	2929 North Court
24	89 Mid-Size Grocer	Street 22	Wau
25	89 Mid-Size Grocer	Street 23	Wau

Store_Lookup[store_country] = "MEXICO"

The measure being evaluated contains a **CALCULATE** function, filter context is modified between **Step 1 & Step 2**

STEP 2

Carry the filters across all table relationships



STEP 3

*Evaluate the formula
against the filtered table*

1

The screenshot shows the 'New Measure' dialog box from Microsoft Power BI. The 'Measure' tab is selected. The 'Table name:' dropdown is set to 'Transactions'. The 'Measure name:' dropdown is set to 'USA Transactions'. The 'Description:' field contains the text 'Transactions for USA stores only'. Below these fields is a 'Formulas:' section with a 'Check formula' button. A large text input area displays the DAX formula: `=CALCULATE([Total Transactions], Store_Lookup[store_country]="USA")`. At the bottom of the dialog are 'Formatting Options' and 'Category' dropdowns (set to 'Number'), a 'Format' button, a 'Whole Number' checkbox, and a 'Use 1000 separator' checkbox. The bottom right corner features 'OK' and 'Cancel' buttons.

Total Transactions where
store_country = "USA" = 180,823



FILTER

FILTER()

Returns a table that represents a subset of another table or expression

=FILTER(<table>, <filter expression>)

Table to be filtered

Examples:

- Store_Lookup
- Product_Lookup

A Boolean (True/False) filter expression to be evaluated for each row of the table

Examples:

- Store_Lookup[store_country]=“USA”
- Calendar[Year]=1998
- [retail_price]>AVERAGE[retail_price]



HEY THIS IS IMPORTANT!

FILTER is used to **add filter context** on top of what's already defined by the PivotTable layout.

Since FILTER returns a table (as opposed to a scalar), it's almost always used as an *input* to other functions, like **enabling more complex filtering options within a CALCULATE function** (or passing a filtered table to an iterator like SUMX)



PRO TIP:

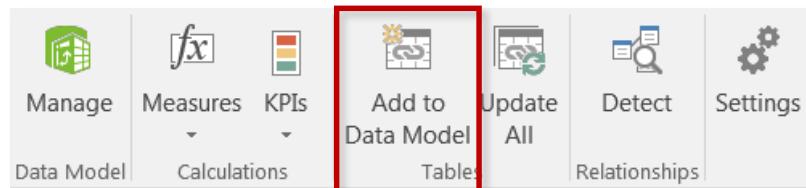
Since FILTER iterates through each row in a table, it can be slow and processor-intensive; **never use FILTER when a normal CALCULATE function will accomplish the same thing!**

PRO TIP: FILTERING WITH DISCONNECTED SLICERS (PART 1)

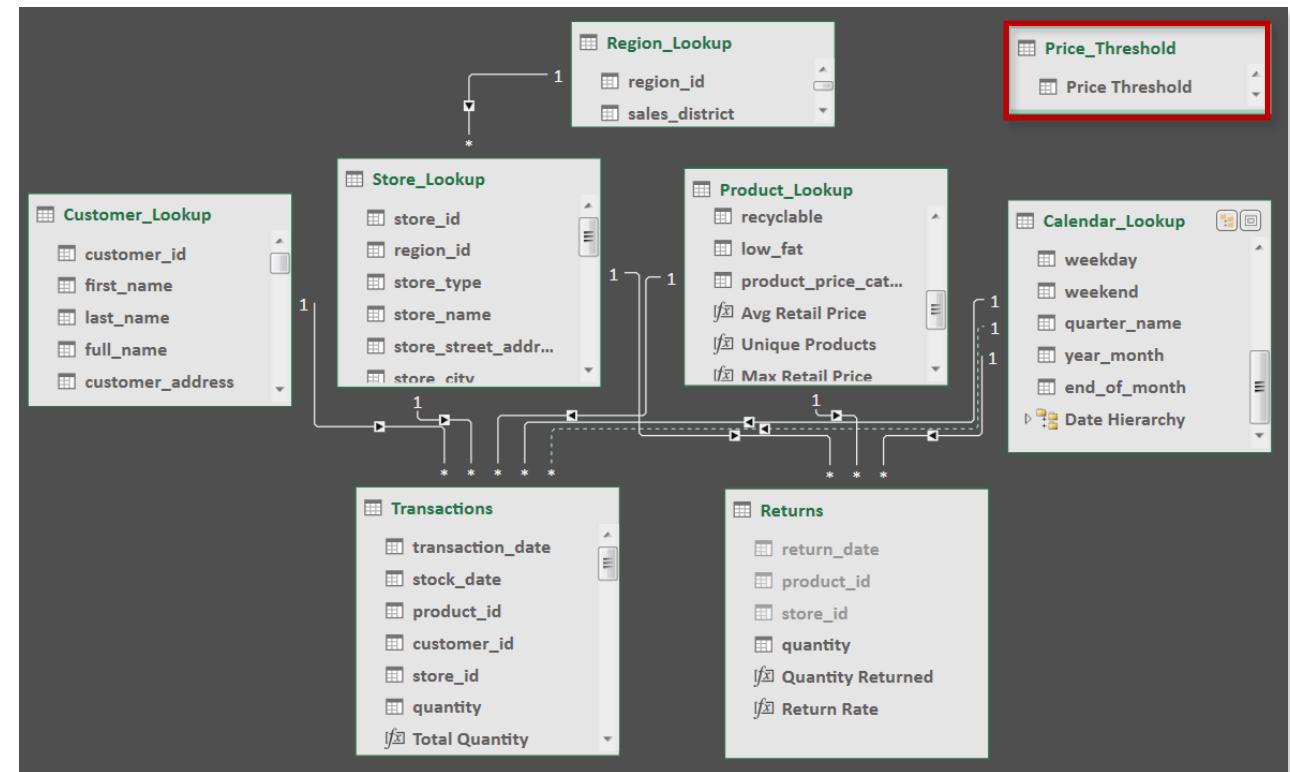
STEP 1: Create an Excel table containing a list of values to use as thresholds or parameters:

A
1
Price Threshold
2
3
4
5

STEP 2: Add the table to the *Data Model* (from **Power Pivot** tab):

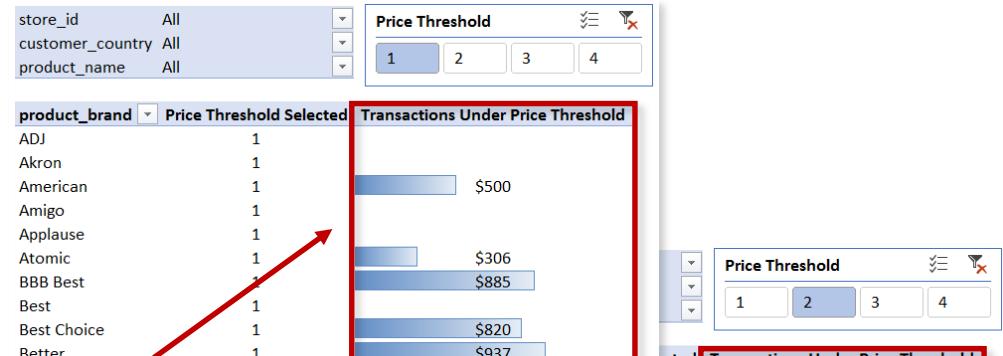


STEP 3: Make sure that your table loaded, and is NOT connected to any other table in the model:

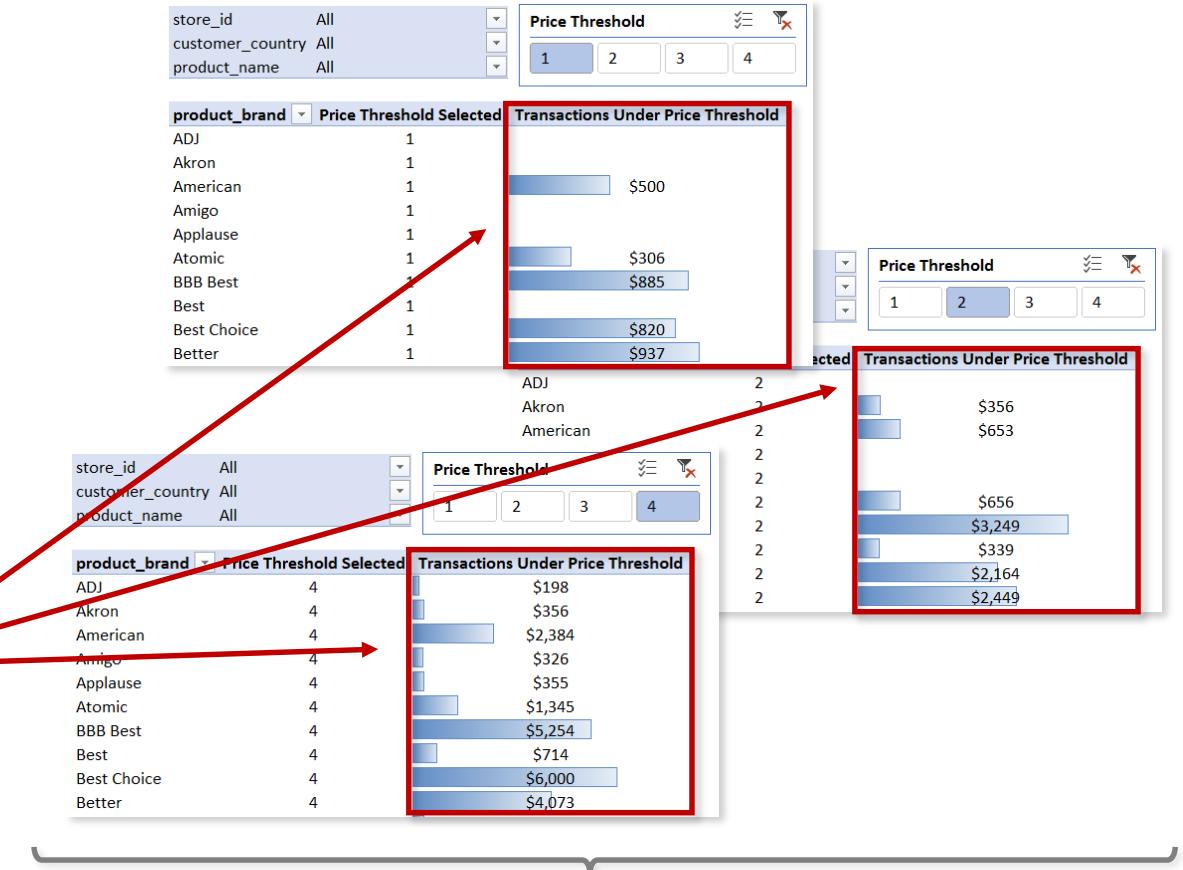
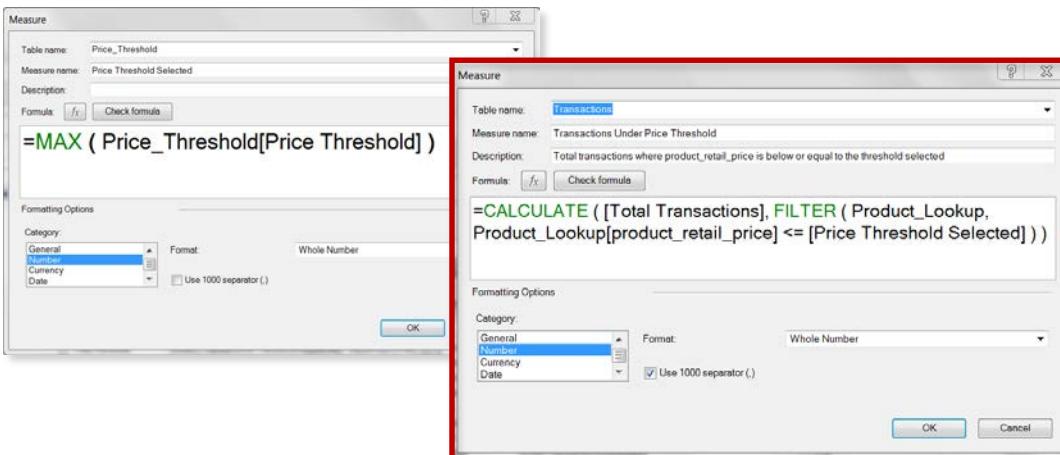


PRO TIP: FILTERING WITH DISCONNECTED SLICERS (PART 2)

STEP 4: Place your new table on the pivot as a slicer:



STEP 5: Create a measure to capture the slicer selection, then reference it in a **FILTER** statement within **CALCULATE**:



The **Transactions Under Price Threshold** measure calculates Total Transactions when the product price is below the selected threshold

FILTER (EXAMPLES)

Calculate Total Transactions **only** for cases where the product price is below a selected threshold

Measure

Table name: **Transactions**

Measure name: **Transactions Under Price Threshold**

Description: Total transactions where product_retail_price is below or equal to the threshold selected

Formula: **=CALCULATE ([Total Transactions], FILTER (Product_Lookup,
Product_Lookup[product_retail_price] <= [Price Threshold Selected]))**

Formatting Options

Category: **Number** Format: Whole Number

Use 1000 separator (.)

OK Cancel

Calculate Total Revenue, **but only** for USA stores

Measure

Table name: **Transactions**

Measure name: **USA Revenue**

Description:

Formula: **=SUMX (**
FILTER (Store_Lookup, Store_Lookup[store_country] = "USA"), [Total Revenue])

Formatting Options

Category: **Currency** Symbol: **\$** Decimal places: **0**

Use 1000 separator (.)

OK Cancel

ALL

ALL()

Returns all rows in a table, or all values in a column, ignoring any filters that have been applied

=ALL(<table> or <column>, [column1], [column2], ...)

The table or column that you want to clear filters on

Examples:

- Transactions
- Product_Lookup[product_brand]

List of columns that you want to clear filters on (optional)

Notes:

- If your first parameter is a table, you can't specify additional columns
- All columns must include the table name, and come from the same table

Examples:

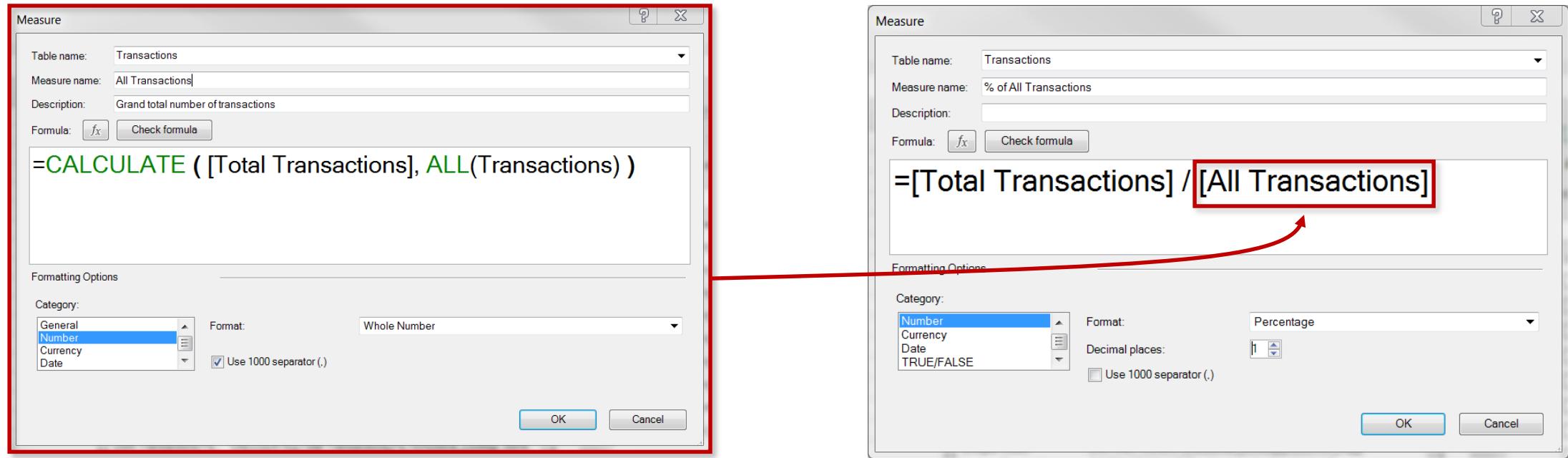
- Customer_Lookup[customer_city], Customer_Lookup[customer_country]
- Product_Lookup[product_name]

PRO TIP:



ALL is like the opposite of FILTER; instead of adding filter context, ALL **removes filter context**. This is often used when you need unfiltered values that won't be skewed by the PivotTable layout (i.e. Category sales as % of Total)

ALL (EXAMPLE)



- In this example, we use **ALL** to calculate total transactions across *all rows* in the Transactions table, **ignoring any filter context from the PivotTable**
 - By dividing the original **[Total Transaction]** measure (which responds to PivotTable filter context as expected) by the new **[All Transactions]** measure, we can correctly calculate the percentage of the total no matter how the PivotTable is filtered

RELATED

RELATED()

Returns related values in each row of a table using relationships with other tables

=RELATED(<column>)



The column that contains the values you want to retrieve

Examples:

- Product_Lookup[product_brand]
- Store_Lookup[store_country]



HEY THIS IS IMPORTANT!

RELATED works almost *exactly* like a **VLOOKUP** function – it uses the relationship between tables (*defined by primary and foreign keys*) to pull values from one table into a new column of another.

Since this function requires row context, it can only be used as a **calculated column** or as part of an **iterator function** that cycles through all rows in a table (FILTER, SUMX, MAXX, etc.)

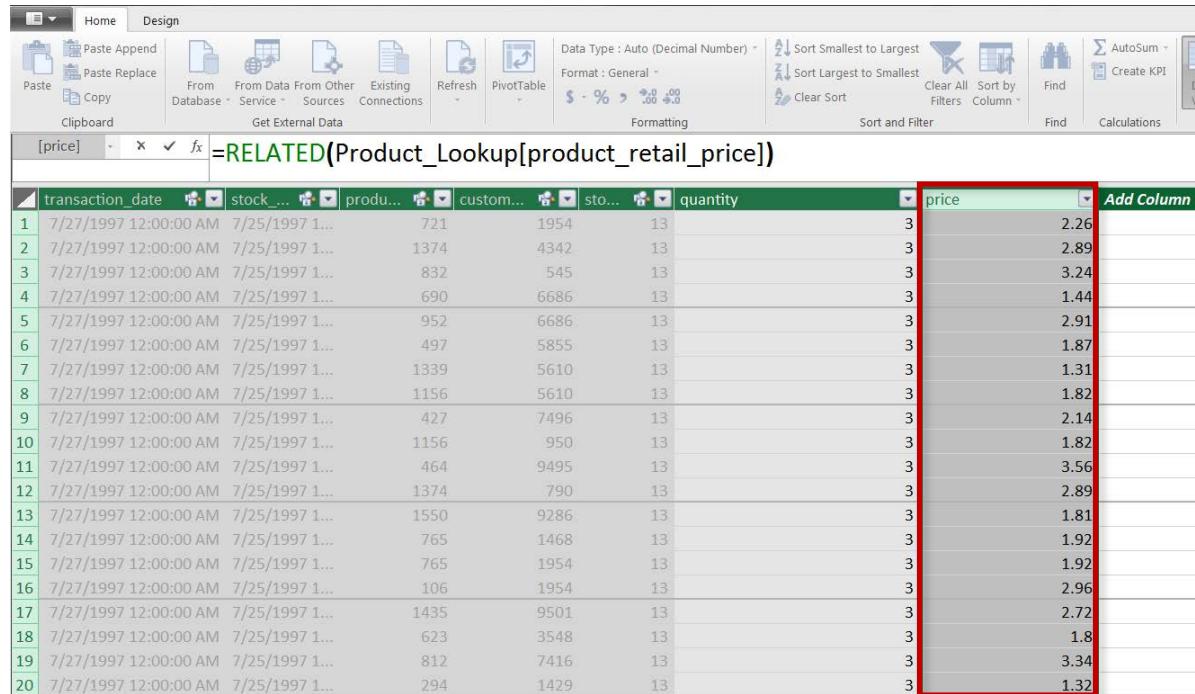
PRO TIP:

Avoid using RELATED to create redundant calculated columns unless you absolutely need them, since those extra columns increase file size; instead, use RELATED within a measure like FILTER or SUMX



RELATED (EXAMPLES)

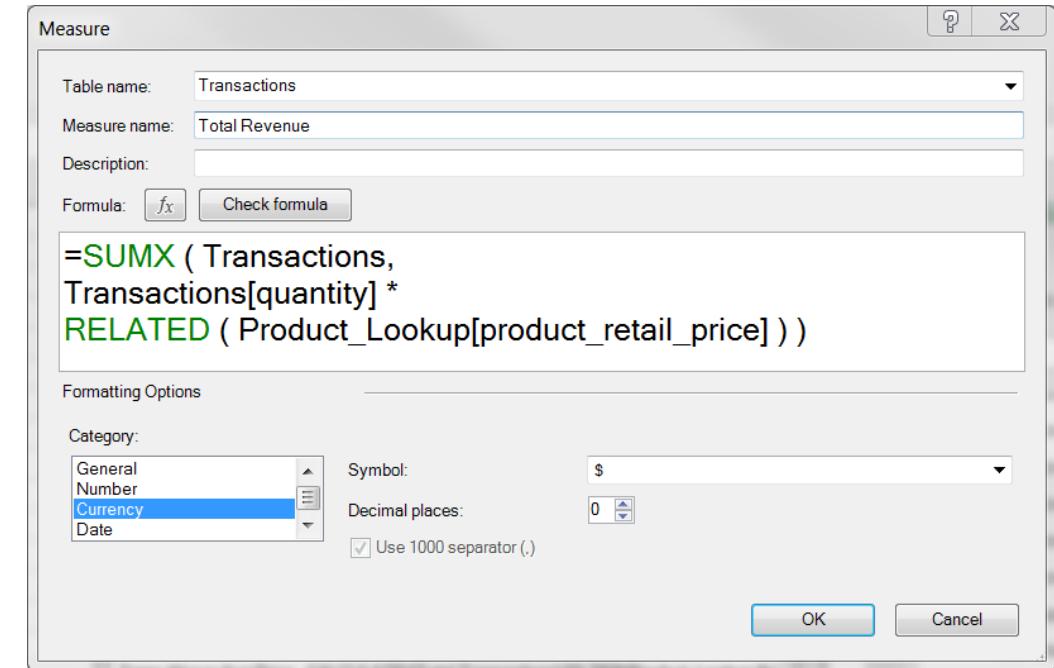
Retrieve the **retail price** from the *Product_Lookup* table and append it to the *Transactions* table



The screenshot shows a Microsoft Excel spreadsheet with a table of transaction data. The table has columns for transaction date, stock ID, product ID, customer ID, store ID, quantity, and price. The 'price' column is highlighted with a red border. The formula bar at the top shows the formula =RELATED(Product_Lookup[product_retail_price]). The ribbon menu is visible at the top.

	transaction_date	stock_id	product_id	customer_id	store_id	quantity	price	Add Column
1	7/27/1997 12:00:00 AM	7/25/1997 1...	721	1954	13	3	2.26	
2	7/27/1997 12:00:00 AM	7/25/1997 1...	1374	4342	13	3	2.89	
3	7/27/1997 12:00:00 AM	7/25/1997 1...	832	545	13	3	3.24	
4	7/27/1997 12:00:00 AM	7/25/1997 1...	690	6686	13	3	1.44	
5	7/27/1997 12:00:00 AM	7/25/1997 1...	952	6686	13	3	2.91	
6	7/27/1997 12:00:00 AM	7/25/1997 1...	497	5855	13	3	1.87	
7	7/27/1997 12:00:00 AM	7/25/1997 1...	1339	5610	13	3	1.31	
8	7/27/1997 12:00:00 AM	7/25/1997 1...	1156	5610	13	3	1.82	
9	7/27/1997 12:00:00 AM	7/25/1997 1...	427	7496	13	3	2.14	
10	7/27/1997 12:00:00 AM	7/25/1997 1...	1156	950	13	3	1.82	
11	7/27/1997 12:00:00 AM	7/25/1997 1...	464	9495	13	3	3.56	
12	7/27/1997 12:00:00 AM	7/25/1997 1...	1374	790	13	3	2.89	
13	7/27/1997 12:00:00 AM	7/25/1997 1...	1550	9286	13	3	1.81	
14	7/27/1997 12:00:00 AM	7/25/1997 1...	765	1468	13	3	1.92	
15	7/27/1997 12:00:00 AM	7/25/1997 1...	765	1954	13	3	1.92	
16	7/27/1997 12:00:00 AM	7/25/1997 1...	106	1954	13	3	2.96	
17	7/27/1997 12:00:00 AM	7/25/1997 1...	1435	9501	13	3	2.72	
18	7/27/1997 12:00:00 AM	7/25/1997 1...	623	3548	13	3	1.8	
19	7/27/1997 12:00:00 AM	7/25/1997 1...	812	7416	13	3	3.34	
20	7/27/1997 12:00:00 AM	7/25/1997 1...	294	1429	13	3	1.32	

Multiply the **quantity** in each row of the *Transactions* table with the related **retail price** from the *Product_Lookup* table, and sum the results



The screenshot shows the 'Measure' dialog box in Power BI. The formula entered is =SUMX (Transactions, Transactions[quantity] * RELATED (Product_Lookup[product_retail_price])). The dialog box includes fields for Table name (Transactions), Measure name (Total Revenue), Description, Formula, and various formatting options like Category (set to Currency) and Decimal places (0).

ITERATOR (“X”) FUNCTIONS

Iterator (or “X”) functions allow you to loop through the same calculation or expression on *each row of a table*, and then apply some sort of aggregation to the results (SUM, MAX, etc.)

=**SUMX**(<table>, <expression>)

Aggregation to apply
to calculated rows*

Examples:

- SUMX
- COUNTX
- AVERAGEX
- RANKX
- MAXX/MINX

Table in which the
expression will be evaluated

Examples:

- Transactions
- FILTER(Transactions,
RELATED(Store_Lookup[country])="USA")

Expression to be evaluated for
each row of the given table

Examples:

- [Total Transactions]
- Transactions[price] * Transactions[quantity]



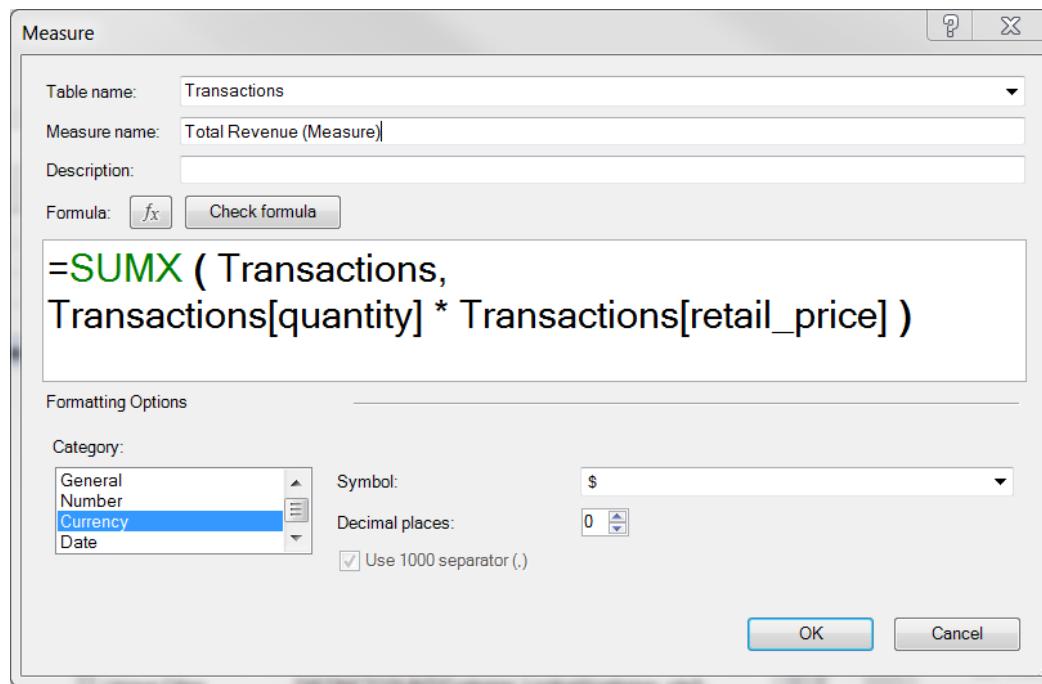
PRO TIP:

Imagine the function **adding a temporary new column** to the table, calculating the value in each row (based on the expression) and then applying the aggregation to that new column (like SUMPRODUCT)

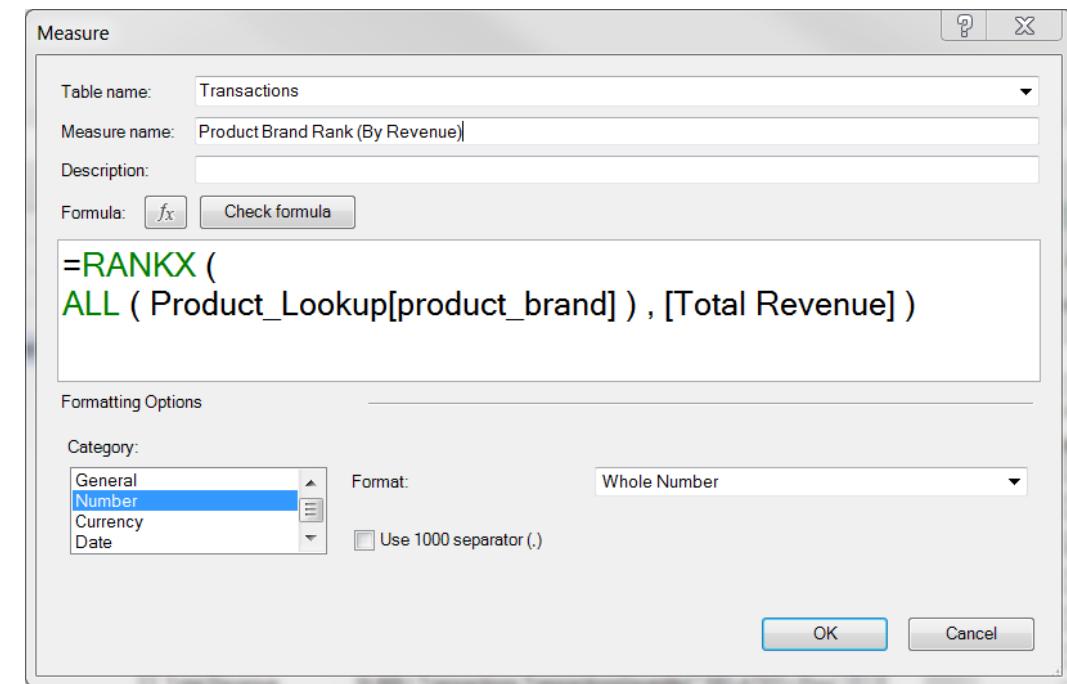
*In this example we're looking at **SUMX**, but all “X” functions follow a similar syntax

ITERATOR ("X") FUNCTIONS (EXAMPLES)

Multiply **quantity** and **retail price** for each row in the Transactions table, and sum the results



Calculate the **rank** of each product brand, based on total revenue



BASIC DATE & TIME FUNCTIONS

**DAY/MONTH/
YEAR()**

Returns the day of the month (1-31), month of the year (1-12), or year of a given date

=**DAY/MONTH/YEAR(<date>)**

**HOUR/MINUTE/
SECOND()**

Returns the hour (0-23), minute (0-59), or second (0-59) of a given datetime value

=**HOUR/MINUTE/SECOND(<datetime>)**

TODAY/NOW()

Returns the current date or exact time

=**TODAY/NOW()**

**WEEKDAY/
WEEKNUM()**

Returns a weekday number from 1 (Sunday) to 7 (Sunday), or the week # of the year

=**WEEKDAY/WEEKNUM(<date>, <type>)**

EOMONTH()

Returns the date of the last day of the month, +/- a specified number of months

=**EOMONTH(<start_date>, <months>)**

DATEDIFF()

Returns the difference between two dates, based on a selected interval

=**DATEDIFF(<start_date>, <end_date>, <interval>)**

BASIC DATE & TIME FUNCTIONS (EXAMPLES)

Calculate the time difference between the customer birthdate and current date, in years

	acct_open_date	member_card	occupation	homeowner	full_name	birth_year	has_children	customer_age	education_level
1	11/16/1994 12:00:00	Bronze	Manual	N	Bertha Jam...	1948	Y	69	Non-Grad
2	5/5/1992 12:00:00	Bronze	Manual	N	Ole Weldon	1931	Y	86	Non-Grad
3	6/26/1994 12:00:00	Bronze	Manual	N	Paul Alcorn	1973	Y	44	Non-Grad
4	2/9/1990 12:00:00	Bronze	Manual	N	Jared Busta...	1910	Y	107	Non-Grad
5	3/15/1992 12:00:00	Bronze	Manual	N	Margaret A...	1979	Y	38	Non-Grad
6	3/2/1994 12:00:00	Bronze	Manual	N	Vanessa Ten...	1930	Y	87	Non-Grad
7	6/4/1993 12:00:00	Bronze	Manual	N	Catherine ...	1966	Y	51	Non-Grad
8	3/5/1992 12:00:00	Bronze	Manual	N	Sherry G...	1912	Y	74	Non-Grad
9	5/17/1992 12:00:00	Bronze	Manual	N					
10	9/8/1992 12:00:00	Bronze	Manual	N					
11	2/21/1991 12:00:00	Bronze	Manual	N					
12	6/12/1994 12:00:00	Bronze	Manual	N					
13	4/24/1992 12:00:00	Bronze	Manual	N					
14	11/8/1991 12:00:00	Bronze	Manual	N					
15	6/16/1992 12:00:00	Bronze	Manual	N					
16	3/8/1993 12:00:00	Bronze	Manual	N					
17	6/19/1994 12:00:00	Bronze	Manual	N					
18	8/27/1993 12:00:00	Bronze	Manual	N					
19	3/26/1991 12:00:00	Bronze	Manual	N					
20	3/28/1994 12:00:00	Bronze	Manual	N					

Calculate the end date of the month, for each row in the Calendar_Lookup table

	year	quarter	month	month_name	week_of_year	start_of_week	day	weekday	weekend	quarter_name	year_month	end_of_month	Add Column
1	1997	3	7	July	27	6/30/1997 12:00:00	1	Tuesday	0	Q3	19977	7/31/1997 12:00:00 AM	
2	1997	3	7	July	27	6/30/1997 12:00:00	2	Wednesday	0	Q3	19977	7/31/1997 12:00:00 AM	
3	1997	3	7	July	27	6/30/1997 12:00:00	3	Thursday	0	Q3	19977	7/31/1997 12:00:00 AM	
4	1997	3	7	July	27	6/30/1997 12:00:00	4	Friday	0	Q3	19977	7/31/1997 12:00:00 AM	
5	1997	3	7	July	27	6/30/1997 12:00:00	5	Saturday	1	Q3	19977	7/31/1997 12:00:00 AM	
6	1997	3	7	July	28	6/30/1997 12:00:00	6	Sunday	1	Q3	19977	7/31/1997 12:00:00 AM	
7	1997	3	7	July	28	7/7/1997 12:00:00	7	Monday	0	Q3	19977	7/31/1997 12:00:00 AM	
8	1997	3	7	July	28	7/7/1997 12:00:00	8	Tuesday	0	Q3	19977	7/31/1997 12:00:00 AM	
9	1997	3	7	July	28	7/7/1997 12:00:00	9	Wednesday	0	Q3	19977	7/31/1997 12:00:00 AM	
10	1997	3	7	July	28	7/7/1997 12:00:00	10	Thursday	0	Q3	19977	7/31/1997 12:00:00 AM	
11	1997	3	7	July	28	7/7/1997 12:00:00	11	Friday	0	Q3	19977	7/31/1997 12:00:00 AM	
12	1997	3	7	July	28	7/7/1997 12:00:00	12	Saturday	1	Q3	19977	7/31/1997 12:00:00 AM	

TIME INTELLIGENCE FORMULAS

Time Intelligence functions allow you to easily calculate common time comparisons:

Performance
To-Date

=CALCULATE(<measure>, DATESYTD(Calendar[Date]))

Use DATESQTD for Quarters or DATESMTD for Months

Previous
Period

=CALCULATE(<measure>, DATEADD(Calendar[Date], -1, MONTH))

Running
Total

=CALCULATE(<measure>,

DATESINPERIOD(Calendar[Date], MAX(Calendar[Date]), -10, DAY))

*Select an interval (DAY, MONTH, QUARTER, or YEAR) and the
of intervals to compare (i.e. previous month, rolling 10-day)*



PRO TIP:

To calculate a **moving average**, use the running total calculation above and divide by the # of intervals!

SPEED & PERFORMANCE CONSIDERATIONS

★ Avoid using unnecessary slicers, or consider disabling cross-filtering

- When you use multiple slicers, they “cross-filter” by default; in other words, options in *Slicer B* are automatically grayed out if they aren’t relevant given a selected value in *Slicer A*
- To disable, select *Slicer Tools > Slicer Settings* and uncheck “Visually indicate items with no data”

★ Eliminate redundant columns; keep data tables narrow

- Data tables should ideally only contain quantitative values and foreign keys; any extra descriptive columns should live in a related lookup table

★ Imported columns are better than calculated columns

- When possible, create calculated columns at the source (i.e. in your raw database) or using Power Query; this is more efficient than processing those calculations in the Data Model/Power Pivot

★ Minimize iterator functions (FILTER, SUMX, etc.)

- Functions that cycle through each row in a table are “expensive”, meaning that they take time and consume processing power

DAX BEST PRACTICES

★ Write measures for even the simplest calculations (*i.e. Sum of Sales*)

- *Once you create a measure it can be used anywhere in the workbook and as an input to other, more complex calculations*

★ Break measures down into simple, component parts

- *DAX is a difficult language to master; focus on practicing and understanding simple components at first, then assemble them into more advanced formulas*

★ Reference columns with the table name, and measures alone

- *Using “fully qualified” column references (preceded by the table name) helps make formulas more readable and intuitive, and differentiates them from measure references*

WRAPPING UP

DATA VISUALIZATION OPTIONS

There are several options for building **visuals** and **reports** from a data model:

- 1 **PivotCharts & Conditional Formatting**
 - Check out my *Data Analysis with Excel PivotTables* course for a deep dive
 - 2 **Spreadsheet-based dashboards built with CUBE functions**
 - Use CUBE functions to pull values from the data model for custom Excel reports (no pivots)
 - 3 **Power View, Power Map, etc.**
 - Excel plug-in with Power Pivot and other BI tools; recommend PowerBI as a better option
 - 4 **Microsoft PowerBI**
 - Brand new (*free!*) self-service BI product for **loading, shaping, modeling, and visualizing data**
- Available within Excel*
- Standalone product (desktop + online)*

SNEAK PEEK: POWERBI



PowerBI is a standalone Microsoft business intelligence product, which includes both desktop and web-based applications for loading, modeling, and visualizing data

For info about plans & pricing: powerbi.microsoft.com

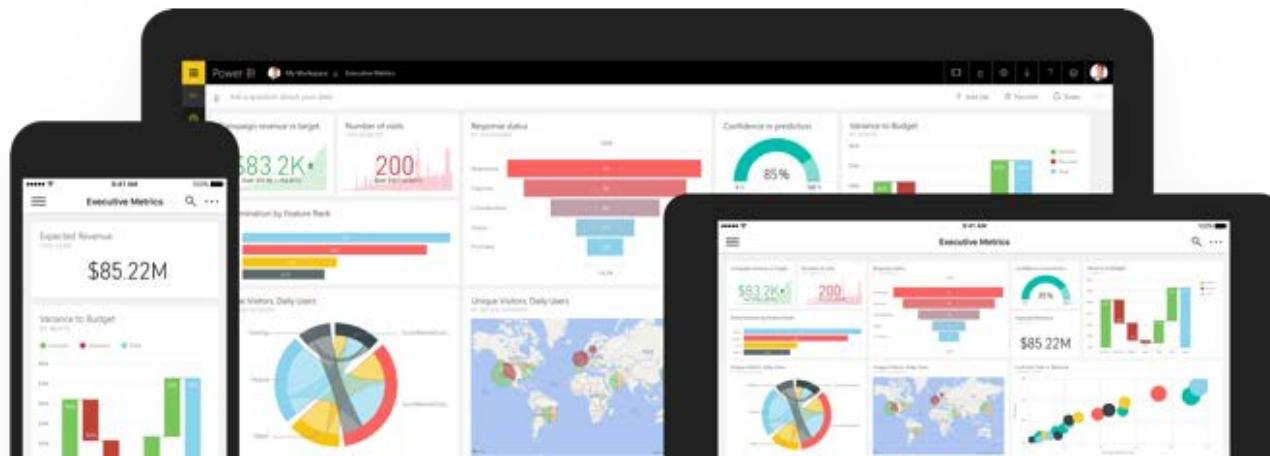
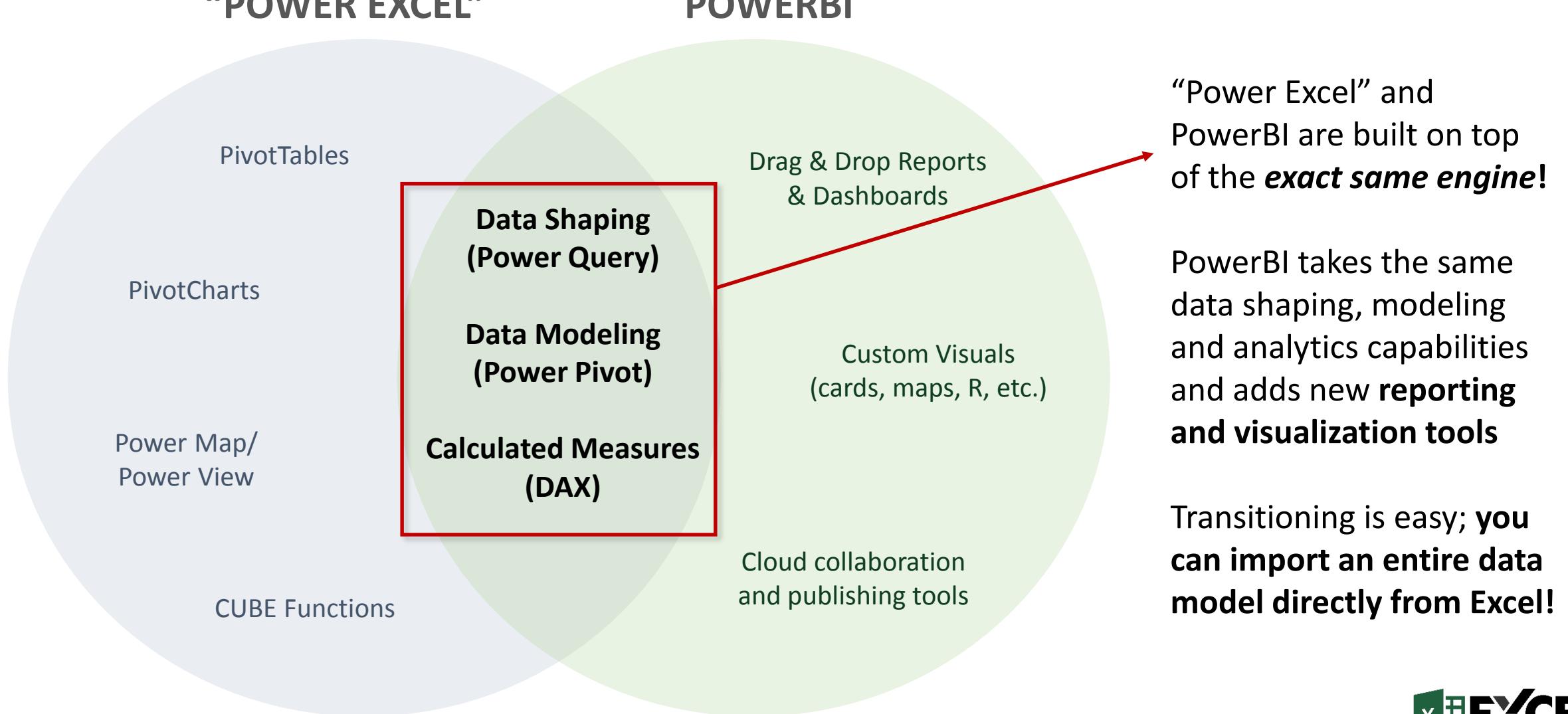


Figure 1. Magic Quadrant for Business Intelligence and Analytics Platforms



“POWER EXCEL” VS. POWERBI



RESOURCES & NEXT STEPS



Looking to become an absolute Excel **ROCK STAR**? Try the full stack:

- *Microsoft Excel – Data Analysis with Excel PivotTables*
- *Microsoft Excel – Advanced Excel Formulas & Functions*
- *Microsoft Excel – Data Viz with Excel Charts & Graphs*
- *Microsoft PowerBI Essentials* (**COMING SOON!**)



Check out these **awesome resources** for additional support:

- *msdn.microsoft.com* for DAX documentation and support
- *powerpivotpro.com* for blogs, articles, and additional Power Pivot resources
- *Power Query & Power BI* by Rob Collie (paperback, available on Amazon)



Ratings and reviews mean the world to me, so **please** share feedback!

- *Feel free to post to the Q&A section or message me directly if you need any support, or if there's anything I can do to improve your course experience!*



THANK YOU!