SumProduct

NEWSLETTER #139 - June 2024
www.sumproduct.com | www.sumproduct.com/thought

NEW REGEX FUNCTIONS

New functions and new arguments abound in this month's newsletter as we explain what regular expressions are and how three **REGEX** functions have been added into the Excel formula fold. **PIVOTBY** gets a makeover too, with a new argument out now: **relative_to**. Not sure what happened to ***relative_one*** to be honest!

And then there is the standard fare. We have the usual Beat the Boredom Challenge, Charts & Dashboards tips, Excel for Mac, Visual Basics, Power Pivot Principles, Power Query Pointers, Power BI Updates and Excel Updates too. Our Keyboard Shortcuts resort to Functions (well, Function Keys!) We're in a "days" (**NETWORKDAYS**) by the time we hit our A to Z of Excel functions, that's for sure!

As always, happy reading and remember: stay safe, stay happy, stay healthy.

Liam Bastick, Managing Director, SumProduct

Regex Comes to Excel

Reg Ex was who my wife referred to as a past acquaintance; regex, on the other hand, is a language used for pattern-matching text content. The term "regex" is an abbreviation of "regular expressions" and is frequently implemented in various programming languages such as C, C++, Java, Python, VBScript – and now, *Excel*.

Microsoft has stated that the version of Regex coming to Excel uses a "flavor" *(sic)* called **PCRE2 (PHP>=7.3)** for those that need to know the underlying technical stuff. Great name for a baby, methinks.

So how is it coming to Excel? Today sees the release of three [3] new Excel functions to the Beta versions of Excel:

1. **REGEXEXTRACT(text, pattern, [return_mode], [ignore_case])**
2. **REGEXREPLACE(text, pattern, replacement, [occurrence], [ignore_case])**
3. **REGEXTEST(text, pattern, [ignore_case])**.

Clearly, we need to learn a little about "regular expressions" before continuing. Alternatively referred to "rational expressions" upon occasion, a regular expression is a sequence of characters that specifies what is known as a "match pattern" in text. You have most likely used this functionality in Excel already, with features such as "Find and Relace" or by using the **FIND** or **SEARCH** functions in Excel. The purpose of these three [3] new functions (presumably, this is just a start!) is to help you match, locate and manage text (strings) in Excel.

The text is obvious but understanding patterns requires you to learn the syntax for regular expressions. Here is a crash course table, which summarises some – but not all – of the main elements, usually referred to as "tokens".

Token	Meaning
\	This converts special characters (metacharacters) to literal characters, and also allows the literal matching of the regex delimiter in use, *e.g.* '/'
.	Matches any character other than newline
^	Matches the start of string without consuming any characters. If multiline mode is used, this will also match immediately after a newline character
$	Matches the end of string without consuming any characters. If multiline mode is used, this will also match immediately before a newline character
a?	Matches zero [0] or one [1] of a. This matches an 'a' character or nothing
a*	Matches zero [0] or more of a. This matches zero or consecutive 'a' characters
a+	Matches one [1] or more of a. This matches consecutive 'a' characters
a{4}	Matches exactly four [4] instances of 'a'
a{4,}	Matches four [4] or more instances of 'a'
a{4,6}	Matches between four [4] and six [6] instances of 'a'
\A	Matches the start of a string only. Unlike ^, this is not affected by multiline mode
\Z	Matches the end of a string only. Unlike $, this is not affected by multiline mode
\z	Matches the absolute end of a string only. Unlike $, this is not affected by multiline mode and in contrast to \Z, this will not match before a trailing newline at the end of a string
\b	Matches a word boundary. It matches without consuming any characters, immediately between a character matched by \w and a character not matched by \w. It cannot be used to separate non-words from words
\B	Matches a non-word boundary. It matches without consuming any characters , at the position between two characters matched by \w or \W
i	A case insensitive match is performed
x	Ignore whitespace / verbose. This flag instructs the engine to ignore all whitespace and allow for comments in the regex, also known as verbose. Comments are indicated by starting with the # character and then escaping with \
xx	Ignore all whitespace / verbose. Similar to x, but whitespace is also ignored inside of character classes
s	Known as single line, this enables the dot (.) metacharacter to also match newlines, thus treating the whole string as a single line input
\n	Matches a newline character
\N	Matches anything other than a newline character
\r	Matches a carriage return, Unicode character U+2185
\R	Careful! Matches any Unicode newline sequence
\t	Matches a tab character (typically, tab stops happen every eight [8] characters)
\0 [zero]	Matches a *null* character, Unicode character U+2400
\d	Matches any decimal / digit. Equivalent to [0-9]
\D	Matches anything other than a decimal / digit

Token	Meaning
\s	Matches any whitespace character (space, tab or newline)
\S	Matches any non-whitespace character (anything other space, tab or newline)
\w	Matches any word character (any letter, digit or underscore). Equivalent to [a-zA-Z0-9_]
\W	Matches any non-word character (anything other than a letter, digit or underscore). Equivalent to [^a-zA-Z0-9_]
[abc]	Matches an 'a', 'b' or 'c' character
[^abc]	Matches any character except 'a', 'b' or 'c'
a|b	Alternate match: matches what is before or after |, in this case 'a' or 'b'
[a-z]	Matches any characters between a and z inclusive
[^a-z]	Matches any characters, except those in the range a to z inclusive
[a-zA-Z]	Matches any characters between a to z or A to Z inclusive
[[:alnum:]]	Double square brackets are required here. Matches letters and digits. This is equivalent to [A-Za-z0-9]
[[:alpha:]]	Matches letters. Equivalent to [a-zA-Z]
[[:ascii:]]	Matches any character in the valid ASCII range (any basic Latin character). ASCII codes 0 to 127 inclusive
[[:blank:]]	Matches spaces and tabs (but not newlines). Equivalent to [\t]
[[:cntrl:]]	Matches characters that are often used to control text presentation, including newlines, *null* characters, tabs and the escape character
[[:digit:]]	Matches decimal / digits. Equivalent to [0-9] or \d
[[:graph:]]	Matches visible characters (not space: printable, non-whitespace, non-control characters only)
[[:lower:]]	Matches lowercase letters. Equivalent to [a-z]
[[:print:]]	Matches printable characters, part of the basic Latin set, such as letters and spaces, but not including control characters
[[:punct:]]	Matches visible punctuation characters that are not whitespace, letters or numbers
[[:space:]]	Matches whitespace characters. Equivalent to \s
[[:upper:]]	Matches uppercase letters. Equivalent to [A-Z]
[[:word:]]	Matches word characters (letters, numbers and underscores). Equivalent to \w or [a-zA-Z0-9_]
[[:<:]]	Matches the start of word
[[:>:]]	Matches the end of word
(?:…)	Match everything enclosed. For example, repeating 1-3 digits and a period 3 times can be identified as follows: /(?:\d{1,3}\.){3}\d{1,3}/
(…}	Capture everything enclosed

Now we are all experts in regex, let's go through the three new functions.

REGEXEXTRACT

This function is used extract one or more strings that match a specified pattern from the text being analysed. You may extract the first match, all matches or capturing groups from the first match. Its syntax is as follows:

REGEXEXTRACT(text, pattern, [return_mode], [ignore_case])

It has the following three arguments:

- **text:** this is required, and represents the **text** you are searching within
- **pattern:** this is also required. This is the regular expression to be applied
- **return_mode:** the first of two optional arguments, this specifies which matches to return. It has three alternatives:

Return Mode	Description
0	First match (default)
1	All matches
2	Capture groups of first match

Capturing groups are part of a regular expression ("regex") pattern surrounded by parentheses "(…)". They allow you to return separate parts of a single match individually

- **ignore_case:** the final (optional) argument. This determines whether the match should be case sensitive. It has the following two [2] options:

Ignore Case	Description
0	Case sensitive match (default)
1	Case insensitive match

This function always returns text values. You may convert these results back to numerical values using the **VALUE** function.

Consider the following examples:

	A	B	C
1	Name		
2	Liam Bastick		
3			
4			
5	Formula	Description	Result
6	=REGEXEXTRACT(A2,"[a-zA-Z]+")	Extract first name based upon letters with pattern "[a-zA-Z]+"	Liam
7	=REGEXEXTRACT(A2,"[a-zA-Z]+",1)	Extract all names based upon letters with pattern "[a-zA-Z]+"	Liam
8	Spilled from above		Bastick
9			

	A	B	C
1	Data		
2	Harry Potter (378) 555-4195		
3	Snooker Potter (878) 555-8622		
4	Beatrix Potter (437) 555-8987		
5	Beer Tricks Potter (619) 555-4212		
6	Trixie Potter (579) 555-5658		
7	Bumble Dore (346) 555-0925		
8	Dumble Bore (405) 555-0887		
9	Bumble Bee (666) 555-1872		
10			
11			
12	Formula	Description	Result
13	=REGEXEXTRACT(A2:A9,"[0-9()]+ [0-9-]+",1)		(378) 555-4195
14	Spilled from above		(878) 555-8622
15	Spilled from above		(437) 555-8987
16	Spilled from above	Extracts phone numbers based upon their pattern	(619) 555-4212
17	Spilled from above		(579) 555-5658
18	Spilled from above		(346) 555-0925
19	Spilled from above		(405) 555-0887
20	Spilled from above		(666) 555-1872
21			

REGEXREPLACE

The **REGEXREPLACE** function replaces strings within the provided text that matches the pattern with replacement. The syntax of the **REGEXREPLACE** function is:

REGEXREPLACE(text, pattern, replacement, [occurrence], [case_sensitivity])

where:

- **text:** this is required, and represents the **text** or the reference to a cell containing the **text** you wish to replace strings within
- **pattern:** this is also required. This is the regular expression ("regex") that describes the pattern you wish to replace
- **replacement:** another required argument, this is the text you wish to replace instances of **pattern**
- **occurrence:** the first of two optional arguments, this specifies which instance of the pattern you wish to replace. By default, **occurrence** is zero [0], which will replace all instances. It should be noted that a negative number replaces that instance, searching from the end instead
- **case_sensitivity:** the final (optional) argument. This determines whether the match should be case sensitive. It has the following two [2] options:

Case Sensitivity	Description
0	Case sensitive match (default)
1	Case insensitive match

This function always returns text values. You may convert these results back to numerical values using the **VALUE** function.

Consider the following examples:

	A	B	C
1	Name		
2	LiamBastick		
3			
4			
5	Formula	Description	Result
6	=REGEXREPLACE(A2,"([A-Z][a-z]+)([A-Z][a-z]+)","$2, $1")	Capturing groups to separate and reorder first and last names	Bastick, Liam

	A	B	C
1	Data		
2	Harry Potter (378) 555-4195		
3	Snooker Potter (878) 555-8622		
4	Beatrix Potter (437) 555-8987		
5	Beer Tricks Potter (619) 555-4212		
6	Trixie Potter (579) 555-5658		
7	Bumble Dore (346) 555-0925		
8	Dumble Bore (405) 555-0887		
9	Bumble Bee (666) 555-1872		
10			
11			
12	Formula	Description	Result
13	=REGEXREPLACE(A2:A9,"[0-9]+-","***-")		Harry Potter (378) ***-4195
14	Spilled from above		Snooker Potter (878) ***-8622
15	Spilled from above		Beatrix Potter (437) ***-8987
16	Spilled from above	Used to anonymise phone numbers by replacing the first three digits of main number with ***	Beer Tricks Potter (619) ***-4212
17	Spilled from above		Trixie Potter (579) ***-5658
18	Spilled from above		Bumble Dore (346) ***-0925
19	Spilled from above		Dumble Bore (405) ***-0887
20	Spilled from above		Bumble Bee (666) ***-1872
21			

REGEXTEST

The **REGEXTEST** function allows you to check whether any part of supplied text matches a regular expression ("regex"). It will return TRUE if there is a match and FALSE otherwise. The syntax of the **REGEXTEST** function is:

REGEXTEST(text, pattern, [case_sensitivity])

where:

- **text:** this is required, and represents the **text** or the reference to a cell containing the **text** you wish to match against
- **pattern:** this is also required. This is the regular expression ("regex") that you wish to match
- **case_sensitivity:** the final (optional) argument. This determines whether the match should be case sensitive. It has the following two [2] options:

Case Sensitivity	Description
0	Case sensitive match (default)
1	Case insensitive match

This function always returns text values. You may convert these results back to numerical values using the **VALUE** function.

Consider the following examples:

	A	B	C
1	Data		
2	Liam.Bastick@sumproduct.com		
3			
4			
5	Formula	Description	Result
6	=REGEXTEST(A2,"@([a-zA-Z0-9_.+-]+)\.[a-zA-Z0-9_.+-]")	Is this an email address?	TRUE

	A	B	C
1	Data		
2	alfalfa		
3			
4			
5	Formula	Description	Result
6	=REGEXTEST(A2,"a")	Does the string contain the letter 'a'?	TRUE
7	=REGEXTEST(A2,"[a-z]")	Are there any lower case letters?	TRUE
8	=REGEXTEST(A2,"[A-Z]")	Are there any upper case letters?	FALSE
9	=REGEXTEST(A2,"[aeiou]")	Are there any vowels?	TRUE
10	=REGEXTEST(A2,"[0-9]")	Does the string contain digits?	FALSE

Word to the Wise

In announcing these three new functions, Microsoft has also stated that they will be shortly introducing a way to use regular expressions within **XLOOKUP** and **XMATCH**, via a new, revised option for the **match_mode** argument. The regex pattern will be supplied as the **lookup_value** – that's coming to Beta very soon!!

In the meantime, feel free to play with these new functions (including why not ask Copilot for regex patterns) which are being rolled out to the Beta channel. You will need both patience and:

- Windows: Version 2406 (Build 17715.20000) or later
- Mac: Version 16.86 (Build 24051422) or later.

Revision to PIVOTBY Function

Back in November last year, Microsoft announced several new functions, including **PIVOTBY**, and eta lambdas such as **PERCENTOF**. On the quiet, they have added a new argument which makes **PIVOTBY** just that little bit more powerful.

At the time of writing, they are rolling out to users enrolled in the beta channel for Windows Excel and Mac Excel. Don't be upset if you don't get this new update straight away. But first, let's have a little refresher…

eta Lambdas

These "eta reduced lambda" functions may sound scary, but they make the world of dynamic arrays more accessible to the inexperienced. They help make the other three functions simpler to use. Dynamic array calculations using basic aggregation functions often require syntax such as

LAMBDA(x, SUM(x))

LAMBDA(y, AVERAGE(y))

etc.

However, given **x** and **y** *(above)* are merely dummy variables, an "eta lambda" function simply replaces the need for this structure with the so-easy-anyone-can-understand-it syntax of

SUM

AVERAGE

etc.

Even I can do it. For example, consider the following formula in cell **G17** below:

Using SUM with BYCOL

	Q1	Q2	Q3	Q4
North	8	8	3	10
South	9	4	9	9
East	8	4	5	8
West	1	10	2	6
Total	**26**	**26**	**19**	**33**

=BYCOL(G13:J16,LAMBDA(x,SUM(x)))

=BYCOL(G13:J16,LAMBDA(x,SUM(x)))

This sums the range **G13:J16** by column using that **LAMBDA(x, SUM(x))** trick. But there is no need for this anymore, *viz.*

	Q1	Q2	Q3	Q4
North	8	8	3	10
South	9	4	9	9
East	8	4	5	8
West	1	10	2	6
Total	**26**	**26**	**19**	**33**

=BYCOL(G21:J24,SUM)

=BYCOL(G21:J24,SUM)

That's much simpler and many one argument functions may now be turned into eta lambdas (and one or two other functions too).

PERCENTOF

This function can be used in conjunction with **PIVOTBY** *(below)* or on its own. This is used to return the percentage that a subset makes up of a given dataset. It is logically equivalent to

SUM(subset) / SUM(everything)

It sums the values in the subset of the dataset and divides it by the sum of all the values. It has the following syntax:

=PERCENTOF(data_subset, data_all)

The arguments are as follows;

- **data_subset:** this is required, and represents the values that are in the data subset
- **data_all:** this too is required and denotes the values that make up the entire set.

You can use it, for example, with **GROUPBY**:

=GROUPBY(tbl[Category],tbl[Sales],PERCENTOF)

Description	Percentage
Accessories	24.47%
Bikes	24.99%
Clothing	25.69%
Components	24.86%
Total	**100.00%**

=GROUPBY(tbl[Category],tbl[Sales],PERCENTOF)

Alternatively, it may be used on its own:

Category	Sales
Accessories	24.47%
Bikes	24.99%
Clothing	25.69%
Components	24.86%
Total	**100.00%**

Accessories and Bikes 49.46%

=PERCENTOF(G13:G14,G13:G16)

PIVOTBY

The reason for this article is that **PIVOTBY** has changed. It has added a new, final argument: **relative_to** – but let's back up first.

The **PIVOTBY** function allows you to create a summary of your data via a formula too, akin to a formulaic PivotTable. It supports grouping along two axes and aggregating the associated values. For instance, if you had a table of sales data, you might generate a summary of sales by state and year.

It should be noted that **PIVOTBY** is a function that returns an array of values that can spill to the grid. Furthermore, at this stage, not all features of a PivotTable appear to be replicable by this function.

The syntax of the **PIVOTBY** function is:

PIVOTBY(row_fields, col_fields, values, function, [field_headers], [row_total_depth], [row_sort_order], [col_total_depth], [col_sort_order], [filter_array], [relative_to])

It has the following arguments:

- **row_fields:** this is required, and represents a column-oriented array or range that contains the values which are used to group rows and generate row headers. The array or range may contain multiple columns. If so, the output will have multiple row group levels
- **col_fields:** also required, and represents a column-oriented array or range that contains the values which are used to group columns and generate column headers. The array or range may contain multiple columns. If so, the output will have multiple column group levels
- **values:** this is also required, and denotes a column-oriented array or range of the data to aggregate. The array or range may contain multiple columns. If so, the output will have multiple aggregations
- **function:** also required, this is an explicit or eta reduced lambda (*e.g.* **SUM**, **PERCENTOF**, **AVERAGE**, **COUNT**) that is used to aggregate **values**. A vector of lambdas may be provided. If so, the output will have multiple aggregations. The orientation of the vector will determine whether they are laid out row- or column-wise
- **field_headers:** this and the remaining arguments are all optional. This represents a number that specifies whether the **row_fields**, **col_fields** and **values** have headers and whether field headers should be returned in the results. The possible values are:
    - **Missing:** Automatic
    - **0:** No
    - **1:** Yes, and don't show
    - **2:** No, but generate
    - **3:** Yes, and show

  It should be noted that "Automatic" assumes the data contains headers based upon the **values** argument. If the first value is text and the second value is a number, then the data is assumed to have headers. Fields headers are shown if there are multiple row or column group levels
- **row_total_depth:** this optional argument determines whether the row headers should contain totals. The possible values are:
    - **Missing**: Automatic, with grand totals and, where possible, subtotals
    - **0:** No Totals
    - **1:** Grand Totals
    - **2:** Grand and Subtotals
    - **-1:** Grand Totals at Top
    - **-2:** Grand and Subtotals at Top

  It should be noted that for subtotals, **row_fields** must have at least two [2] columns. Numbers greater than two [2] are supported provided **row_field** has sufficient columns
- **row_sort_order:** again optional, this argument denotes a number indicating how rows should be sorted. Numbers correspond with the columns in **row_fields** followed by the columns in **values**. If the number is negative, the rows are sorted in descending / reverse order. A vector of numbers may be provided when sorting based upon only **row_fields**
- **col_total_depth:** this optional argument determines whether the column headers should contain totals. The possible values are:
    - **Missing**: Automatic, with grand totals and, where possible, subtotals
    - **0:** No Totals
    - **1:** Grand Totals
    - **2:** Grand and Subtotals
    - **-1:** Grand Totals at Top
    - **-2:** Grand and Subtotals at Top

  It should be noted that for subtotals, **col_fields** must have at least two [2] columns. Numbers greater than two [2] are supported provided **col_field** has sufficient columns

- **col_sort_order:** again optional, this argument denotes a number indicating how they should be sorted. Numbers correspond with the columns in **col_fields** followed by the columns in **values**. If the number is negative, these are sorted in descending / reverse order. A vector of numbers may be provided when sorting based upon only **col_fields**
- **filter_array:** this now penultimate optional argument, this represents a column-oriented one-dimensional array of Boolean values [1, 0] that indicate whether the corresponding row of data should be considered. It should be noted that the length of the array must match the length of **row_fields** and **col_fields**
- **relative_to:** this new, final argument allows you to summarise functions relative to row and column totals or the grand total. Five alternatives are possible:
    - **0:** Column Totals (default) *(Screentip: Calculation performed relative to all values in column)*
    - **1:** Row Totals *(Calculation performed relative to all values in row)*
    - **2:** Grand Total *(Calculation performed relative to all values)*
    - **3:** Parent Column Total *(Calculation performed relative to all values in column parent)*
    - **4:** Parent Row Total *(Calculation performed relative to all values in row parent)*.

Let's look at **PIVOTBY** using **PERCENTOF**, highlighting this new **relative_to** final argument. Consider the following Table (**CTRL + T**) called **Data** *(truncated)*:

Source Table

Year	Quarter	Category	Item	Sales	Rating
2022	Q1	Components	Wheels	$ 4,000	10%
2024	Q1	Components	Pedals	$ 3,200	50%
2022	Q4	Components	Brakes	$ 3,300	45%
2022	Q4	Clothing	Jerseys	$ 1,100	10%
2022	Q2	Components	Saddles	$ 500	85%
2022	Q1	Clothing	Jerseys	$ 1,500	30%
2023	Q2	Accessories	Bike Racks	$ 2,600	85%
2022	Q1	Bikes	Touring Bikes	$ 1,100	30%
2024	Q4	Clothing	Tights	$ 800	65%
2023	Q1	Clothing	Bib-Shorts	$ 1,000	45%
2023	Q1	Accessories	Helmets	$ 2,700	45%
2022	Q4	Clothing	Gloves	$ 800	20%
2024	Q3	Clothing	Vests	$ 1,100	30%
2023	Q4	Components	Brakes	$ 1,100	100%
2024	Q2	Components	Handlebars	$ 3,200	25%
2024	Q3	Accessories	Locks	$ 400	55%
2023	Q3	Accessories	Tyres and Tubes	$ 500	70%
2022	Q4	Components	Pedals	$ 1,000	45%
2023	Q4	Accessories	Helmets	$ 3,600	60%
2022	Q4	Bikes	Touring Bikes	$ 200	55%
2023	Q4	Clothing	Gloves	$ 4,000	100%
2022	Q4	Accessories	Locks	$ 1,500	75%
2024	Q1	Bikes	Road Bikes	$ 600	75%
2024	Q3	Clothing	Gloves	$ 900	65%
2024	Q1	Components	Chains	$ 100	10%
2024	Q1	Components	Chains	$ 1,600	45%
2023	Q4	Bikes	Touring Bikes	$ 2,400	70%

Here, we have two parent / child relationships:

1. **Year** and **Quarter**
2. **Category** and **Item**.

Previously, with **PIVOTBY**, we can create a formulaic alternative to a PivotTable (with crafty formatting) using the following formula:

=PIVOTBY(Data[[Category]:[Item]],Data[[Year]:[Quarter]],Data[Sales],PERCENTOF)

1. Default Presentation

Sales as a Percentage, Displayed by Category and Item vs. Year and Quarter

Default

		2022 Q1	2022 Q2	2022 Q3	2022 Q4	2023 Q1	2023 Q2	2023 Q3	2023 Q4	2024 Q1	2024 Q2	2024 Q3	2024 Q4	Total
Accessories	Bike Racks	10.7%	4.8%	2.8%	0.3%	1.6%	3.1%		5.2%	1.2%	5.0%	2.0%	11.2%	**4.2%**
Accessories	Helmets	4.4%	2.0%	4.4%	11.1%	2.8%	10.3%	10.9%	4.0%	1.8%	4.5%	4.7%	10.2%	**5.8%**
Accessories	Lights	6.4%	2.7%	6.4%	3.3%	5.1%	1.1%	4.2%	1.9%	2.1%	3.9%	1.4%	0.7%	**3.3%**
Accessories	Locks	5.1%	4.0%	2.9%	12.3%	3.1%	3.9%	1.0%	0.8%	3.2%	3.6%	1.5%	1.1%	**3.6%**
Accessories	Pumps	1.7%	3.7%	9.3%	0.8%	11.8%	0.1%		0.6%	7.0%	3.6%	2.2%	3.9%	**3.7%**
Accessories	Tyres and Tubes	5.4%	1.8%	3.9%	1.2%	5.0%	2.7%	7.5%	7.1%	3.3%	3.2%	5.9%	0.4%	**3.9%**
Bikes	Cargo Bikes	5.6%	2.5%	8.7%	6.4%	9.2%	6.3%	5.8%	10.4%	4.6%	8.7%	14.2%	9.0%	**7.5%**
Bikes	Mountain Bikes	3.4%	6.5%	3.7%	7.2%	5.6%	6.6%	4.6%	9.1%	7.3%	5.3%	8.7%	5.8%	**6.2%**
Bikes	Road Bikes	3.3%	2.9%	3.7%	10.7%	0.7%	8.7%	6.9%	3.7%	8.0%	4.3%	5.6%	7.1%	**5.4%**
Bikes	Touring Bikes	4.5%	3.7%	5.4%	5.3%	5.6%	5.4%	6.9%	7.5%	6.9%	6.2%	8.3%	4.9%	**5.8%**
Clothing	Bib-Shorts	2.0%	2.0%	4.6%	5.2%	2.2%	1.6%	0.4%	1.9%	1.8%	2.7%	4.9%	2.2%	**2.6%**
Clothing	Caps	2.1%	5.2%	3.1%	1.1%	4.4%	4.9%		0.7%	1.3%	6.3%	2.5%	0.5%	**2.7%**
Clothing	Gloves	4.3%	10.7%	3.9%	2.6%		2.6%	1.4%	6.0%	2.9%		2.9%	4.8%	**3.6%**
Clothing	Jerseys	0.8%	2.6%		7.5%	6.3%	6.8%	6.4%	4.1%	5.9%	0.1%	3.5%	4.6%	**4.0%**
Clothing	Shorts	3.1%	6.2%	0.7%	0.7%	2.0%	4.7%	3.7%	8.7%	2.3%	3.7%	1.2%	2.6%	**3.4%**
Clothing	Socks	4.6%	2.2%	6.7%	3.1%	1.7%	1.6%	1.8%	2.8%	7.0%	2.6%	0.3%	0.8%	**3.0%**
Clothing	Tights	1.1%		7.8%	2.6%	3.8%	4.5%	1.9%	6.2%	5.9%	1.8%	2.3%	2.2%	**3.3%**
Clothing	Vests	3.3%	3.0%	0.3%	2.1%	7.2%	2.8%	4.8%	2.2%	0.5%	2.7%	3.9%	5.4%	**3.1%**
Components	Bottom Brackets	6.1%	2.9%	3.0%	3.0%	2.5%		3.9%	4.0%		3.6%	2.5%	4.2%	**3.0%**
Components	Brakes	2.6%	3.3%		1.9%	7.5%		7.0%	0.6%	2.7%	0.8%	3.5%	2.9%	**2.7%**
Components	Chains	1.8%	3.8%	0.4%	3.1%	1.3%	2.0%	5.0%	3.5%	5.6%	3.3%	5.2%	4.1%	**3.3%**
Components	Handlebars	6.8%	8.5%	5.2%	2.5%	2.8%	7.8%	5.8%	1.8%	1.4%	2.7%	4.2%	2.4%	**4.3%**
Components	Pedals	0.1%	5.5%	2.2%	4.8%	4.7%	3.9%	1.8%	2.1%	7.8%	11.1%	2.9%	5.8%	**4.4%**
Components	Saddles	2.7%	5.8%	2.9%	0.9%	2.8%	3.0%	2.7%	3.2%	5.0%	6.5%		2.9%	**3.3%**
Components	Wheels	8.1%	3.7%	8.1%		0.1%	5.4%	5.6%	1.9%	4.5%	3.8%	5.6%	0.3%	**3.9%**
Total		**100.0%**	**100.0%**	**100.0%**	**100.0%**	**100.0%**	**100.0%**	**100.0%**	**100.0%**	**100.0%**	**100.0%**	**100.0%**	**100.0%**	**100.0%**

Note that each column of sales is represented as a percentage of that column (including the Total column). Whilst it was a great start, Microsoft received feedback that end users wanted to see percentages summarised in alternative ways – and that is what has been addressed here.

This newly introduced final argument, **relative_to**, behaves the same in scenario **0**: Column Totals. This is the default view:

=PIVOTBY(Data[[Category]:[Item]],Data[[Year]:[Quarter]],Data[Sales],PERCENTOF,,,,,,0)

2. Column Totals (Relative Value 0)

Sales as a Percentage, Displayed by Category and Item vs. Year and Quarter

Column Totals (Relative Value 0)

		2022 Q1	2022 Q2	2022 Q3	2022 Q4	2023 Q1	2023 Q2	2023 Q3	2023 Q4	2024 Q1	2024 Q2	2024 Q3	2024 Q4	Total
Accessories	Bike Racks	10.7%	4.8%	2.8%	0.3%	1.6%	3.1%		5.2%	1.2%	5.0%	2.0%	11.2%	**4.2%**
Accessories	Helmets	4.4%	2.0%	4.4%	11.1%	2.8%	10.3%	10.9%	4.0%	1.8%	4.5%	4.7%	10.2%	**5.8%**
Accessories	Lights	6.4%	2.7%	6.4%	3.3%	5.1%	1.1%	4.2%	1.9%	2.1%	3.9%	1.4%	0.7%	**3.3%**
Accessories	Locks	5.1%	4.0%	2.9%	12.3%	3.1%	3.9%	1.0%	0.8%	3.2%	3.6%	1.5%	1.1%	**3.6%**
Accessories	Pumps	1.7%	3.7%	9.3%	0.8%	11.8%	0.1%		0.6%	7.0%	3.6%	2.2%	3.9%	**3.7%**
Accessories	Tyres and Tubes	5.4%	1.8%	3.9%	1.2%	5.0%	2.7%	7.5%	7.1%	3.3%	3.2%	5.9%	0.4%	**3.9%**
Bikes	Cargo Bikes	5.6%	2.5%	8.7%	6.4%	9.2%	6.3%	5.8%	10.4%	4.6%	8.7%	14.2%	9.0%	**7.5%**
Bikes	Mountain Bikes	3.4%	6.5%	3.7%	7.2%	5.6%	6.6%	4.6%	9.1%	7.3%	5.3%	8.7%	5.8%	**6.2%**
Bikes	Road Bikes	3.3%	2.9%	3.7%	10.7%	0.7%	8.7%	6.9%	3.7%	8.0%	4.3%	5.6%	7.1%	**5.4%**
Bikes	Touring Bikes	4.5%	3.7%	5.4%	5.3%	5.6%	5.4%	6.9%	7.5%	6.9%	6.2%	8.3%	4.9%	**5.8%**
Clothing	Bib-Shorts	2.0%	2.0%	4.6%	5.2%	2.2%	1.6%	0.4%	1.9%	1.8%	2.7%	4.9%	2.2%	**2.6%**
Clothing	Caps	2.1%	5.2%	3.1%	1.1%	4.4%	4.9%		0.7%	1.3%	6.3%	2.5%	0.5%	**2.7%**
Clothing	Gloves	4.3%	10.7%	3.9%	2.6%		2.6%	1.4%	6.0%	2.9%		2.9%	4.8%	**3.6%**
Clothing	Jerseys	0.8%	2.6%		7.5%	6.3%	6.8%	6.4%	4.1%	5.9%	0.1%	3.5%	4.6%	**4.0%**
Clothing	Shorts	3.1%	6.2%	0.7%	0.7%	2.0%	4.7%	3.7%	8.7%	2.3%	3.7%	1.2%	2.6%	**3.4%**
Clothing	Socks	4.6%	2.2%	6.7%	3.1%	1.7%	1.6%	1.8%	2.8%	7.0%	2.6%	0.3%	0.8%	**3.0%**
Clothing	Tights	1.1%		7.8%	2.6%	3.8%	4.5%	1.9%	6.2%	5.9%	1.8%	2.3%	2.2%	**3.3%**
Clothing	Vests	3.3%	3.0%	0.3%	2.1%	7.2%	2.8%	4.8%	2.2%	0.5%	2.7%	3.9%	5.4%	**3.1%**
Components	Bottom Brackets	6.1%	2.9%	3.0%	3.0%	2.5%		3.9%	4.0%		3.6%	2.5%	4.2%	**3.0%**
Components	Brakes	2.6%	3.3%		1.9%	7.5%		7.0%	0.6%	2.7%	0.8%	3.5%	2.9%	**2.7%**
Components	Chains	1.8%	3.8%	0.4%	3.1%	1.3%	2.0%	5.0%	3.5%	5.6%	3.3%	5.2%	4.1%	**3.3%**
Components	Handlebars	6.8%	8.5%	5.2%	2.5%	2.8%	7.8%	5.8%	1.8%	1.4%	2.7%	4.2%	2.4%	**4.3%**
Components	Pedals	0.1%	5.5%	2.2%	4.8%	4.7%	3.9%	1.8%	2.1%	7.8%	11.1%	2.9%	5.8%	**4.4%**
Components	Saddles	2.7%	5.8%	2.9%	0.9%	2.8%	3.0%	2.7%	3.2%	5.0%	6.5%		2.9%	**3.3%**
Components	Wheels	8.1%	3.7%	8.1%		0.1%	5.4%	5.6%	1.9%	4.5%	3.8%	5.6%	0.3%	**3.9%**
Total		**100.0%**	**100.0%**	**100.0%**	**100.0%**	**100.0%**	**100.0%**	**100.0%**	**100.0%**	**100.0%**	**100.0%**	**100.0%**	**100.0%**	**100.0%**

It is clear to see this is identical to the first output. But let's see what happens when we start playing with the final argument. Let's change this value to **1:** Row Totals.

=PIVOTBY(Data[[Category]:[Item]],Data[[Year]:[Quarter]],Data[Sales],PERCENTOF,,,,,,1)

Now, each row of sales is represented as a percentage of that row (including the Total row), *viz.*

3. Row Totals (Relative Value 1)

Sales as a Percentage, Displayed by Category and Item vs. Year and Quarter

Row Totals (Relative Value 1)

		2022 Q1	2022 Q2	2022 Q3	2022 Q4	2023 Q1	2023 Q2	2023 Q3	2023 Q4	2024 Q1	2024 Q2	2024 Q3	2024 Q4	Total
Accessories	Bike Racks	24.1%	10.9%	5.1%	0.7%	2.9%	5.6%		11.2%	2.5%	10.3%	3.7%	23.0%	**100.0%**
Accessories	Helmets	7.1%	3.3%	5.7%	16.9%	3.7%	13.1%	13.6%	6.1%	2.6%	6.7%	6.2%	15.0%	**100.0%**
Accessories	Lights	18.5%	7.9%	14.6%	8.8%	12.0%	2.5%	9.4%	5.4%	5.4%	10.3%	3.4%	1.8%	**100.0%**
Accessories	Locks	13.2%	10.3%	6.0%	29.9%	6.5%	8.1%	2.1%	2.1%	7.6%	8.6%	3.2%	2.5%	**100.0%**
Accessories	Pumps	4.4%	9.5%	19.1%	1.9%	24.6%	0.3%		1.5%	16.5%	8.5%	4.7%	9.2%	**100.0%**
Accessories	Tyres and Tubes	12.9%	4.3%	7.6%	2.7%	9.8%	5.2%	14.0%	16.4%	7.4%	7.1%	11.6%	0.9%	**100.0%**
Bikes	Cargo Bikes	7.0%	3.1%	8.7%	7.5%	9.3%	6.2%	5.6%	12.5%	5.2%	9.9%	14.6%	10.2%	**100.0%**
Bikes	Mountain Bikes	5.2%	9.8%	4.5%	10.3%	6.9%	7.9%	5.4%	13.4%	10.2%	7.4%	10.9%	8.0%	**100.0%**
Bikes	Road Bikes	5.6%	5.0%	5.2%	17.4%	1.0%	11.9%	9.3%	6.1%	12.7%	6.8%	8.0%	11.1%	**100.0%**
Bikes	Touring Bikes	7.2%	5.9%	6.9%	8.0%	7.3%	6.8%	8.6%	11.6%	10.2%	9.1%	11.0%	7.2%	**100.0%**
Clothing	Bib-Shorts	7.1%	7.3%	13.0%	17.4%	6.5%	4.6%	1.1%	6.7%	5.9%	8.8%	14.5%	7.1%	**100.0%**
Clothing	Caps	7.5%	18.2%	8.6%	3.6%	12.6%	13.7%		2.4%	4.3%	20.1%	7.3%	1.7%	**100.0%**
Clothing	Gloves	11.0%	27.3%	8.0%	6.4%		5.4%	2.8%	14.8%	6.8%		6.2%	11.3%	**100.0%**
Clothing	Jerseys	1.9%	6.2%		16.6%	12.0%	12.8%	11.6%	9.4%	12.7%	0.1%	6.8%	9.9%	**100.0%**
Clothing	Shorts	8.7%	17.2%	1.6%	1.8%	4.5%	10.3%	8.1%	23.3%	5.8%	9.4%	2.7%	6.7%	**100.0%**
Clothing	Socks	14.5%	6.8%	17.0%	9.4%	4.4%	3.9%	4.4%	8.7%	20.3%	7.7%	0.7%	2.2%	**100.0%**
Clothing	Tights	3.1%		17.9%	7.1%	9.0%	10.2%	4.2%	17.2%	15.5%	4.6%	5.4%	5.9%	**100.0%**
Clothing	Vests	9.8%	9.0%	0.6%	5.9%	17.5%	6.6%	11.1%	6.4%	1.4%	7.4%	9.6%	14.6%	**100.0%**
Components	Bottom Brackets	18.8%	8.8%	7.5%	8.8%	6.3%		9.5%	11.8%		10.2%	6.3%	11.8%	**100.0%**
Components	Brakes	9.2%	11.5%		6.2%	21.3%		19.0%	2.1%	8.7%	2.4%	10.2%	9.4%	**100.0%**
Components	Chains	5.2%	10.8%	0.9%	8.5%	2.9%	4.6%	11.1%	9.7%	14.8%	8.6%	12.3%	10.6%	**100.0%**
Components	Handlebars	14.7%	18.5%	9.1%	5.1%	5.0%	13.4%	9.8%	3.7%	2.8%	5.4%	7.6%	4.8%	**100.0%**
Components	Pedals	0.2%	11.6%	3.8%	9.6%	8.1%	6.6%	3.0%	4.3%	15.1%	21.5%	5.1%	11.1%	**100.0%**
Components	Saddles	7.9%	16.7%	6.6%	2.5%	6.6%	7.0%	6.0%	8.8%	13.1%	17.2%		7.6%	**100.0%**
Components	Wheels	19.5%	8.8%	15.7%		0.3%	10.4%	10.5%	4.6%	10.0%	8.3%	11.2%	0.7%	**100.0%**
Total		**9.4%**	**9.4%**	**7.5%**	**8.8%**	**7.6%**	**7.4%**	**7.3%**	**9.1%**	**8.6%**	**8.6%**	**7.7%**	**8.6%**	**100.0%**

If you wish, you can show the sales as a percentage of the Grand Total, using **2:** Grand Total:

=PIVOTBY(Data[[Category]:[Item]],Data[[Year]:[Quarter]],Data[Sales],PERCENTOF,,,,,,2)

4. Grand Total (Relative Value 2)

Sales as a Percentage, Displayed by Category and Item vs. Year and Quarter

Grand Total (Relative Value 2)

		2022 Q1	2022 Q2	2022 Q3	2022 Q4	2023 Q1	2023 Q2	2023 Q3	2023 Q4	2024 Q1	2024 Q2	2024 Q3	2024 Q4	Total
Accessories	Bike Racks	1.0%	0.5%	0.2%	0.0%	0.1%	0.2%		0.5%	0.1%	0.4%	0.2%	1.0%	**4.2%**
Accessories	Helmets	0.4%	0.2%	0.3%	1.0%	0.2%	0.8%	0.8%	0.4%	0.2%	0.4%	0.4%	0.9%	**5.8%**
Accessories	Lights	0.6%	0.3%	0.5%	0.3%	0.4%	0.1%	0.3%	0.2%	0.2%	0.3%	0.1%	0.1%	**3.3%**
Accessories	Locks	0.5%	0.4%	0.2%	1.1%	0.2%	0.3%	0.1%	0.1%	0.3%	0.3%	0.1%	0.1%	**3.6%**
Accessories	Pumps	0.2%	0.3%	0.7%	0.1%	0.9%	0.0%		0.1%	0.6%	0.3%	0.2%	0.3%	**3.7%**
Accessories	Tyres and Tubes	0.5%	0.2%	0.3%	0.1%	0.4%	0.2%	0.5%	0.6%	0.3%	0.3%	0.5%	0.0%	**3.9%**
Bikes	Cargo Bikes	0.5%	0.2%	0.7%	0.6%	0.7%	0.5%	0.4%	0.9%	0.4%	0.7%	1.1%	0.8%	**7.5%**
Bikes	Mountain Bikes	0.3%	0.6%	0.3%	0.6%	0.4%	0.5%	0.3%	0.8%	0.6%	0.5%	0.7%	0.5%	**6.2%**
Bikes	Road Bikes	0.3%	0.3%	0.3%	0.9%	0.1%	0.7%	0.5%	0.3%	0.7%	0.4%	0.4%	0.6%	**5.4%**
Bikes	Touring Bikes	0.4%	0.3%	0.4%	0.5%	0.4%	0.4%	0.5%	0.7%	0.6%	0.5%	0.6%	0.4%	**5.8%**
Clothing	Bib-Shorts	0.2%	0.2%	0.3%	0.5%	0.2%	0.1%	0.0%	0.2%	0.2%	0.2%	0.4%	0.2%	**2.6%**
Clothing	Caps	0.2%	0.5%	0.2%	0.1%	0.3%	0.4%		0.1%	0.1%	0.5%	0.2%	0.0%	**2.7%**
Clothing	Gloves	0.4%	1.0%	0.3%	0.2%		0.2%	0.1%	0.5%	0.2%		0.2%	0.4%	**3.6%**
Clothing	Jerseys	0.1%	0.2%		0.7%	0.5%	0.5%	0.5%	0.4%	0.5%	0.0%	0.3%	0.4%	**4.0%**
Clothing	Shorts	0.3%	0.6%	0.1%	0.1%	0.2%	0.3%	0.3%	0.8%	0.2%	0.3%	0.1%	0.2%	**3.4%**
Clothing	Socks	0.4%	0.2%	0.5%	0.3%	0.1%	0.1%	0.1%	0.3%	0.6%	0.2%	0.0%	0.1%	**3.0%**
Clothing	Tights	0.1%		0.6%	0.2%	0.3%	0.3%	0.1%	0.6%	0.5%	0.2%	0.2%	0.2%	**3.3%**
Clothing	Vests	0.3%	0.3%	0.0%	0.2%	0.5%	0.2%	0.3%	0.2%	0.0%	0.2%	0.3%	0.5%	**3.1%**
Components	Bottom Brackets	0.6%	0.3%	0.2%	0.3%	0.2%		0.3%	0.4%		0.3%	0.2%	0.4%	**3.0%**
Components	Brakes	0.2%	0.3%		0.2%	0.6%		0.5%	0.1%	0.2%	0.1%	0.3%	0.3%	**2.7%**
Components	Chains	0.2%	0.4%	0.0%	0.3%	0.1%	0.2%	0.4%	0.3%	0.5%	0.3%	0.4%	0.3%	**3.3%**
Components	Handlebars	0.6%	0.8%	0.4%	0.2%	0.2%	0.6%	0.4%	0.2%	0.1%	0.2%	0.3%	0.2%	**4.3%**
Components	Pedals	0.0%	0.5%	0.2%	0.4%	0.4%	0.3%	0.1%	0.2%	0.7%	1.0%	0.2%	0.5%	**4.4%**
Components	Saddles	0.3%	0.5%	0.2%	0.1%	0.2%	0.2%	0.2%	0.3%	0.4%	0.6%		0.2%	**3.3%**
Components	Wheels	0.8%	0.3%	0.6%		0.0%	0.4%	0.4%	0.2%	0.4%	0.3%	0.4%	0.0%	**3.9%**
Total		**9.4%**	**9.4%**	**7.5%**	**8.8%**	**7.6%**	**7.4%**	**7.3%**	**9.1%**	**8.6%**	**8.6%**	**7.7%**	**8.6%**	**100.0%**

There are still two further scenarios – and this is why our example contained two parent / child relationships. The first is **3:** Parent Column Total:

=PIVOTBY(Data[[Category]:[Item]],Data[[Year]:[Quarter]],Data[Sales],PERCENTOF,,,,,,3)

5. Parent Column Totals (Relative Value 3)

Sales as a Percentage, Displayed by Category and Item vs. Year and Quarter

Parent Column Totals (Relative Value 3)

		2022 Q1	2022 Q2	2022 Q3	2022 Q4	2023 Q1	2023 Q2	2023 Q3	2023 Q4	2024 Q1	2024 Q2	2024 Q3	2024 Q4	Total
Accessories	Bike Racks	59.1%	26.7%	12.5%	1.8%	14.7%	28.2%		57.1%	6.4%	26.0%	9.5%	58.1%	100.0%
Accessories	Helmets	21.5%	10.0%	17.3%	51.2%	10.2%	35.9%	37.1%	16.8%	8.5%	21.8%	20.4%	49.3%	100.0%
Accessories	Lights	37.2%	15.8%	29.4%	17.6%	41.1%	8.4%	32.1%	18.4%	25.7%	49.3%	16.2%	8.8%	100.0%
Accessories	Locks	22.2%	17.3%	10.1%	50.4%	34.8%	43.0%	11.1%	11.1%	34.8%	39.2%	14.6%	11.4%	100.0%
Accessories	Pumps	12.6%	27.2%	54.7%	5.5%	93.2%	1.0%		5.7%	42.4%	21.9%	12.0%	23.7%	100.0%
Accessories	Tyres and Tubes	46.9%	15.5%	27.7%	9.9%	21.7%	11.4%	30.8%	36.2%	27.3%	26.3%	43.1%	3.3%	100.0%
Bikes	Cargo Bikes	26.6%	11.9%	33.0%	28.4%	27.7%	18.5%	16.7%	37.1%	13.1%	24.8%	36.5%	25.6%	100.0%
Bikes	Mountain Bikes	17.5%	32.9%	15.1%	34.5%	20.6%	23.5%	16.0%	39.8%	27.9%	20.3%	29.9%	21.9%	100.0%
Bikes	Road Bikes	17.0%	15.0%	15.6%	52.4%	3.6%	42.2%	32.7%	21.6%	32.9%	17.5%	20.7%	28.8%	100.0%
Bikes	Touring Bikes	25.8%	20.9%	24.6%	28.6%	21.4%	19.8%	25.1%	33.7%	27.1%	24.3%	29.4%	19.3%	100.0%
Clothing	Bib-Shorts	15.8%	16.2%	29.1%	38.9%	34.3%	24.2%	6.1%	35.4%	16.3%	24.2%	40.0%	19.5%	100.0%
Clothing	Caps	19.8%	48.0%	22.8%	9.4%	43.8%	47.7%		8.5%	12.9%	60.1%	21.9%	5.1%	100.0%
Clothing	Gloves	20.9%	51.8%	15.2%	12.0%		23.5%	12.0%	64.5%	27.8%		25.6%	46.6%	100.0%
Clothing	Jerseys	7.7%	25.1%		67.2%	26.2%	27.9%	25.4%	20.4%	42.9%	0.4%	23.2%	33.5%	100.0%
Clothing	Shorts	29.6%	58.7%	5.6%	6.1%	9.7%	22.3%	17.5%	50.5%	23.6%	38.2%	10.9%	27.3%	100.0%
Clothing	Socks	30.4%	14.3%	35.7%	19.6%	20.6%	18.3%	20.6%	40.5%	65.7%	24.9%	2.2%	7.2%	100.0%
Clothing	Tights	11.0%		63.7%	25.3%	22.1%	25.2%	10.3%	42.4%	49.3%	14.8%	17.2%	18.7%	100.0%
Clothing	Vests	38.6%	35.4%	2.5%	23.4%	42.1%	15.8%	26.6%	15.4%	4.4%	22.3%	29.1%	44.2%	100.0%
Components	Bottom Brackets	42.8%	20.1%	17.0%	20.1%	22.9%		34.3%	42.8%		35.9%	22.4%	41.8%	100.0%
Components	Brakes	34.3%	42.7%		23.1%	50.2%		44.9%	4.9%	28.2%	8.0%	33.1%	30.7%	100.0%
Components	Chains	20.6%	42.4%	3.6%	33.3%	10.3%	16.3%	39.1%	34.2%	31.9%	18.6%	26.6%	22.9%	100.0%
Components	Handlebars	31.0%	38.9%	19.2%	10.8%	15.7%	42.0%	30.7%	11.7%	13.6%	26.1%	36.9%	23.3%	100.0%
Components	Pedals	0.9%	46.2%	14.9%	38.0%	36.8%	30.1%	13.5%	19.7%	28.6%	40.6%	9.7%	21.1%	100.0%
Components	Saddles	23.4%	49.5%	19.7%	7.3%	23.4%	24.5%	21.2%	31.0%	34.7%	45.3%		20.0%	100.0%
Components	Wheels	44.2%	20.1%	35.7%		1.0%	40.4%	40.9%	17.7%	33.2%	27.6%	37.1%	2.2%	100.0%
Total		26.8%	26.7%	21.5%	25.1%	24.3%	23.7%	23.2%	28.8%	25.7%	25.6%	23.1%	25.6%	100.0%

Here, the Total column is 100% throughout. It is a little confusing as, if anything, it looks a little like Scenario **1:** Row Totals. This is because the column here refers to the headings in each column, *i.e.* **Year** and **Quarter**. You can see that for any row the sum of the four quarters for any given year totals 100% (including the Total row).

Finally, Scenario **4:** Parent Row Total considers the other parent / child relationship:

=PIVOTBY(Data[[Category]:[Item]],Data[[Year]:[Quarter]],Data[Sales],PERCENTOF,,,,,,4)

6. Parent Row Totals (Relative Value 4)

Sales as a Percentage, Displayed by Category and Item vs. Year and Quarter

Parent Row Totals (Relative Value 4)

		2022 Q1	2022 Q2	2022 Q3	2022 Q4	2023 Q1	2023 Q2	2023 Q3	2023 Q4	2024 Q1	2024 Q2	2024 Q3	2024 Q4	Total
Accessories	Bike Racks	31.7%	25.4%	9.5%	1.2%	5.4%	14.6%		26.4%	6.6%	20.8%	11.4%	40.6%	17.0%
Accessories	Helmets	13.1%	10.7%	14.9%	38.4%	9.6%	48.4%	46.0%	20.2%	9.4%	18.9%	26.5%	37.2%	23.8%
Accessories	Lights	19.1%	14.4%	21.4%	11.2%	17.4%	5.1%	17.9%	9.9%	11.0%	16.4%	8.1%	2.6%	13.4%
Accessories	Locks	15.1%	20.8%	9.7%	42.3%	10.5%	18.5%	4.4%	4.3%	17.3%	15.2%	8.5%	3.8%	14.8%
Accessories	Pumps	5.1%	19.4%	31.3%	2.8%	40.0%	0.6%		3.1%	37.7%	15.2%	12.5%	14.3%	15.0%
Accessories	Tyres and Tubes	15.9%	9.3%	13.3%	4.1%	17.0%	12.7%	31.7%	36.1%	17.9%	13.5%	33.1%	1.5%	15.9%
Bikes	Cargo Bikes	33.4%	16.3%	40.5%	21.6%	43.4%	23.4%	24.0%	33.8%	17.0%	35.4%	38.5%	33.6%	30.1%
Bikes	Mountain Bikes	20.4%	41.5%	17.1%	24.3%	26.6%	24.4%	18.9%	29.8%	27.3%	21.8%	23.7%	21.5%	24.7%
Bikes	Road Bikes	19.4%	18.7%	17.4%	36.2%	3.4%	32.4%	28.6%	12.0%	29.9%	17.5%	15.2%	26.4%	21.8%
Bikes	Touring Bikes	26.8%	23.5%	24.9%	17.9%	26.6%	19.8%	28.6%	24.4%	25.8%	25.4%	22.6%	18.5%	23.4%
Clothing	Bib-Shorts	9.3%	6.4%	16.9%	20.8%	8.1%	5.5%	2.0%	6.0%	6.6%	13.6%	23.0%	9.4%	10.3%
Clothing	Caps	10.1%	16.4%	11.4%	4.3%	16.0%	16.7%		2.2%	4.9%	31.7%	11.8%	2.3%	10.5%
Clothing	Gloves	20.2%	33.4%	14.4%	10.5%		8.9%	6.8%	18.2%	10.4%		13.6%	20.9%	14.2%
Clothing	Jerseys	3.8%	8.3%		30.0%	22.7%	23.2%	31.3%	12.6%	21.3%	0.3%	16.3%	19.8%	15.5%
Clothing	Shorts	14.6%	19.4%	2.7%	2.7%	7.2%	15.8%	18.4%	26.6%	8.3%	18.6%	5.4%	11.5%	13.1%
Clothing	Socks	21.5%	6.7%	24.8%	12.6%	6.2%	5.3%	8.8%	8.7%	25.3%	13.3%	1.2%	3.3%	11.5%
Clothing	Tights	5.1%		28.8%	10.5%	13.8%	15.1%	9.2%	18.9%	21.3%	8.9%	10.6%	9.7%	12.7%
Clothing	Vests	15.4%	9.4%	1.0%	8.5%	26.0%	9.4%	23.5%	6.8%	1.9%	13.6%	18.1%	23.2%	12.2%
Components	Bottom Brackets	21.5%	8.5%	13.8%	18.6%	11.6%		12.4%	23.1%		11.3%	10.3%	18.5%	12.2%
Components	Brakes	9.3%	9.8%		11.6%	34.3%		22.0%	3.6%	10.0%	2.4%	14.7%	13.1%	10.8%
Components	Chains	6.5%	11.3%	1.8%	19.3%	5.8%	9.1%	15.7%	20.5%	20.8%	10.4%	21.7%	18.0%	13.2%
Components	Handlebars	24.0%	25.5%	23.9%	15.4%	13.1%	35.1%	18.3%	10.4%	5.2%	8.5%	17.7%	10.7%	17.4%
Components	Pedals	0.4%	16.5%	10.1%	29.5%	21.6%	17.7%	5.7%	12.4%	28.9%	35.0%	12.2%	25.6%	17.8%
Components	Saddles	9.7%	17.4%	13.2%	5.6%	13.1%	13.7%	8.5%	18.6%	18.4%	20.6%		12.8%	13.1%
Components	Wheels	28.6%	11.0%	37.1%		0.6%	24.4%	17.6%	11.4%	16.7%	11.9%	23.4%	1.3%	15.6%
Total		100.0%	100.0%	100.0%	100.0%	100.0%	100.0%	100.0%	100.0%	100.0%	100.0%	100.0%	100.0%	100.0%

In this final illustration, the Total row is 100% throughout. This looks similar to the default Scenario **0:** Column Totals. This is because the row here refers to the headings in each row, *i.e.* **Category** and **Item**. You can see that for any row the sum of any category for any given **Quarter** and **Year** totals 100% (including the Total column).

Word to the Wise

Starting with **RANDARRAY**, Microsoft continues to venture into new territory by tinkering with new functions and features whilst they remain in beta. Previously, revising a function's signature / syntax was unheard of. Here at SumProduct, we're not complaining. The software giant has been collating formula usage and explicit feedback to determine what is missing / needs revising – and then done something about it.

If only they had done that with **MATCH** many years ago!!

Beat the Boredom Challenge

With many of us currently "working from home" / quarantined, there are only so Zoom / Teams calls and virtual parties you can make before you reach your (data) limit. Perhaps they should measure data allowance in blood pressure millimetres of mercury (mmHg). To try and keep our readers engaged, we will continue to reproduce some of our popular ***Final Friday Fix*** *challenges from yesteryear in this and upcoming newsletters. One suggested solution may be found later in this newsletter. Here's this month's…*

This one is getting more and more relevant every day. Imagine you receive several Excel files containing macros from your friend, your colleague or from the internet. You need all these Excel files to have macros enabled and standing in your way is the security. Unblocking all the security from the 'Properties' window of all the files you have is a time-consuming task. Hence, we have devised a challenge for you to find a simple and efficient way to unblock all the macro-enabled files.

This month's challenge is to remove this message from **all** your Excel files as simply as possible that contain macros *(below)*.

As always, there are some requirements:

- the solution should be simple
- obviously, no coding is allowed!

Sounds easy? Try it. One solution *just might* be found later in this newsletter – but no reading ahead!

Excel for Mac

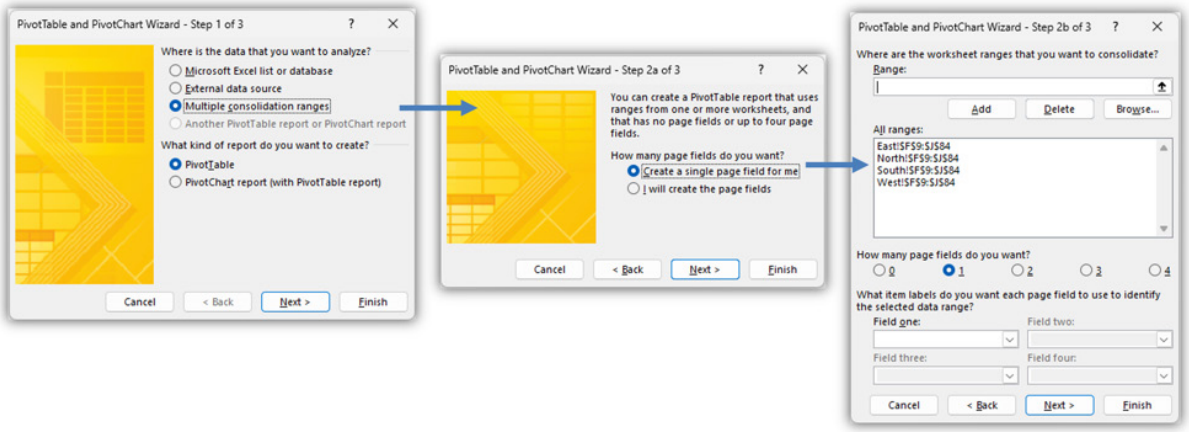
With Steve Kraynak joining the team, we thought we would exploit his knowledge and recant all about Microsoft Excel for Mac. Each month, we'll cover a different topic to help you understand how Excel for Mac is different than Excel for Windows. This month, we show you a better way to create a PivotTable with data from several ranges, sometimes called a Consolidated Range PivotTable.

People using Excel for Mac have requested that Microsoft adds the PivotTable and PivotChart Wizard to Mac so they may create PivotTables that combine, or consolidate, ranges from multiple sheets into a single PivotTable. The wizard exists on Windows, although it's somewhat hidden. It seems that Microsoft doesn't want PivotTables to be created this way and that's fine now that Power Query is available on Mac.

Consolidating Ranges for a PivotTable

Years ago, the way to create a PivotTable based upon multiple ranges from more than one sheet was to use a tool called the 'PivotTable and PivotChart Wizard'. This article doesn't go into the details of how to use it, but it allows you to specify ranges from multiple sheets so the data can be combined into a single PivotTable. This wizard isn't available in Excel for Mac. Even if you're using Windows, it's difficult to find (**ALT + D + P**).

The screen shots below show the wizard on Windows. The last step allows multiple ranges to be combined for use as the data source.



As a tip, in Windows, having pressed **ALT + D + P** to launch the wizard you can then add it to your QAT (Quick Access Toolbar).

Several new ways

The good news is that even if the wizard were available on Mac, we would still recommend using Power Query or some of the new Excel functions for this scenario.

Power Query is great for combining ranges from multiple sheets, workbook, and other sources. Its flexibility makes it a great choice, and we'll show an example of how to do this.

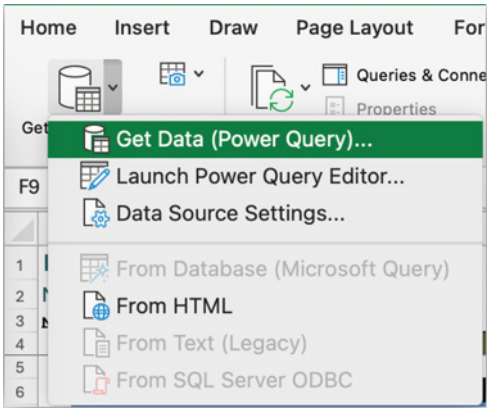
Some of the dynamic array functions are also great for combining data. Specifically, you can use **VSTACK** and **HSTACK** to append arrays (ranges) together into a single array. Then you can use the new array as the source data for a PivotTable. We'll show a brief example of this.

Combine ranges with Power Query

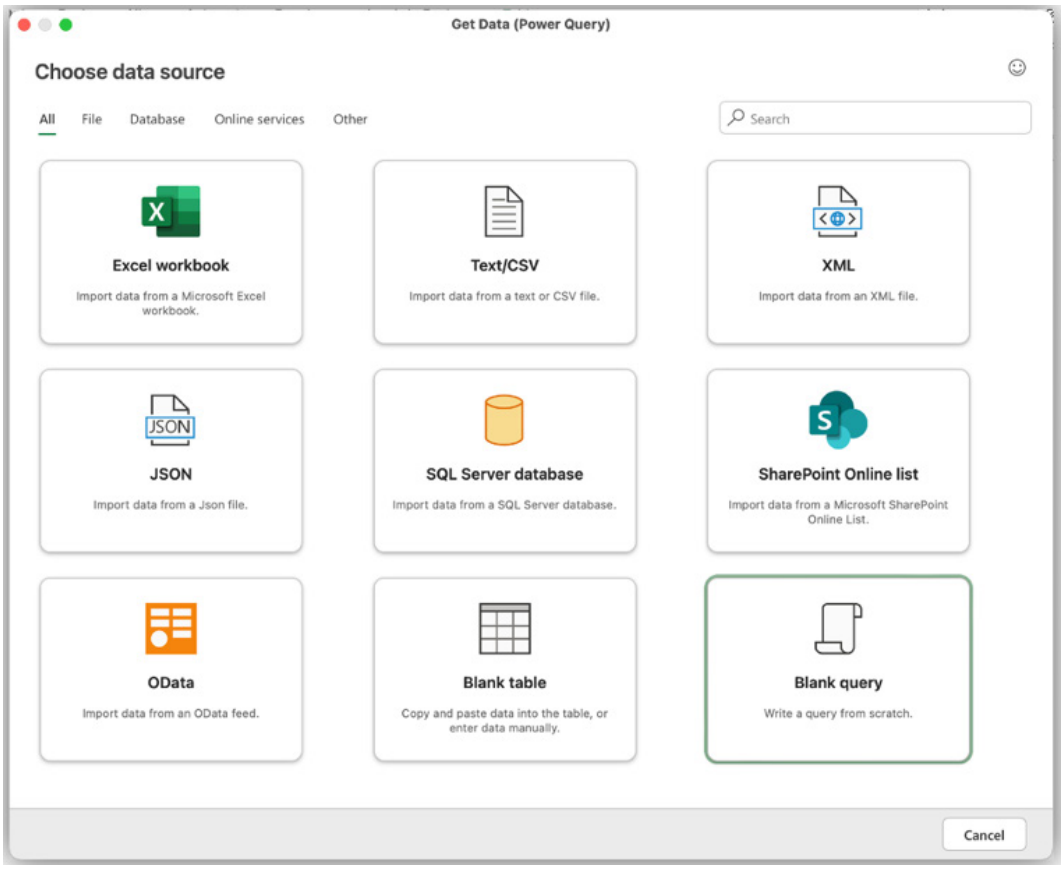
Suppose you have sales data on four [4] sheets representing four regions. The steps here can be used to combine them in Power Query so you can use the combined data in a PivotTable. Hopefully, the format of the data on your sheets is consistent, but if it's not, then Power Query should be able to transform it as needed to make it usable. The same can't be said for the wizard – your data ranges needed to be in a consistent format.

You would just follow these steps:

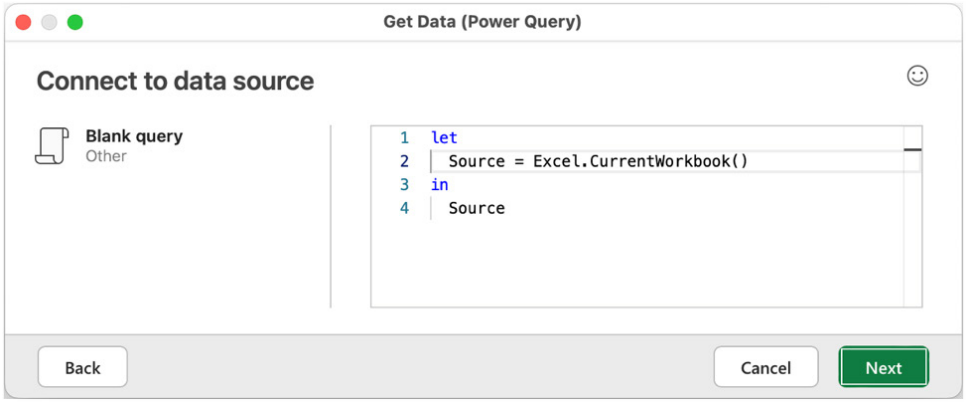
- Create a query for each range: **Choose Data -> Get Data (Power Query)**



- Choose 'Blank Query' from the available data sources



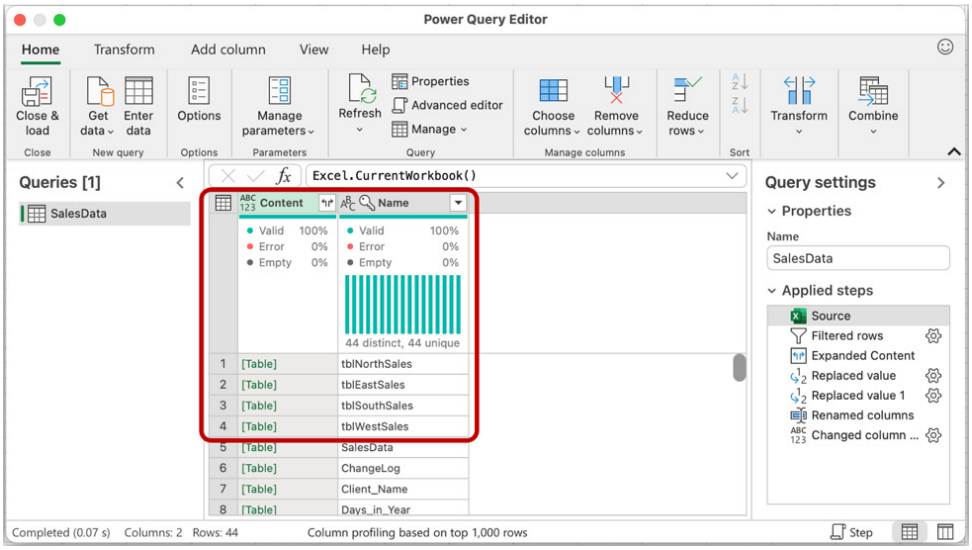
- A new dialog will appear. In the line that says **Source = ""**, replace the double quotes with **Excel.CurrentWorkbook()** as shown in the screen shot below.



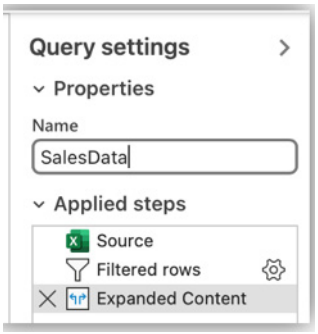
- The code should be exactly as below (it's case-sensitive):

```
let
  Source = Excel.CurrentWorkbook()
in
  Source
```

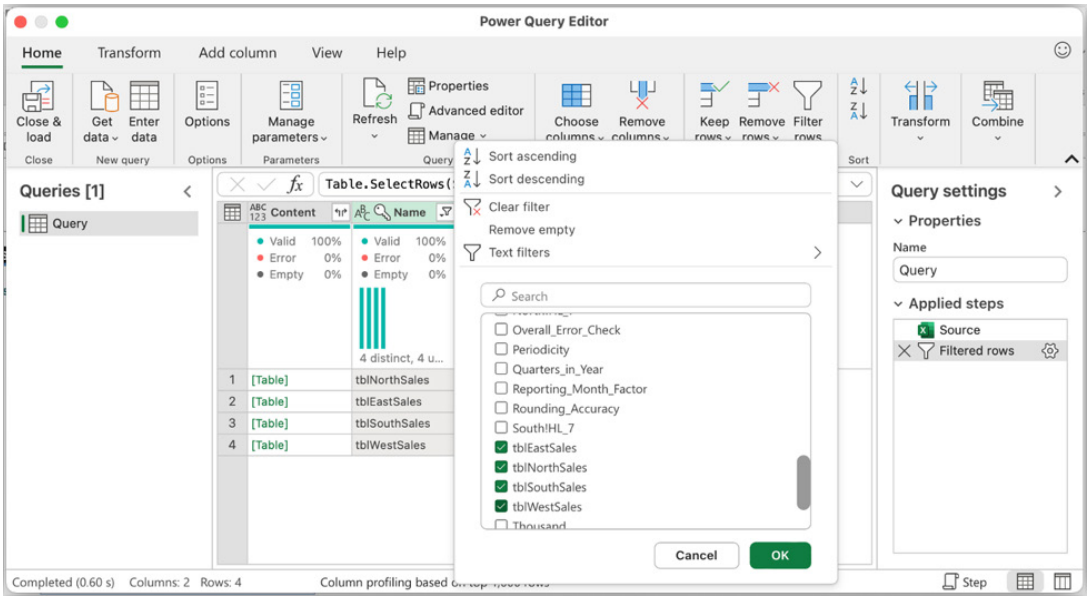
- Press Next. The Power Query Editor window will appear. There will be two [2] columns in the table preview, **Content** and **Name**. The **Content** column will have [Table], which means that each row contains a table or range of data. The **Name** column shows the name of each table or range



- Rename your query to a meaningful name rather than the default name provided. Do this by typing in the Name field in the Query settings pane on the right side. This step is optional and won't affect your data, but the name will be visible when you create your PivotTable



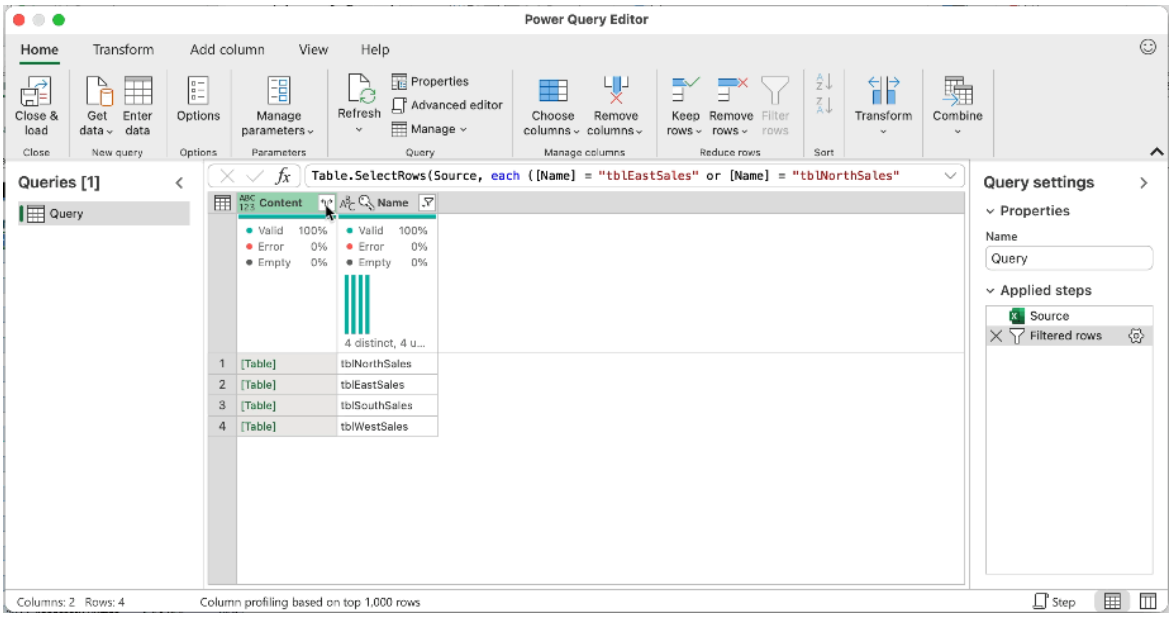
- Click the Filter button in the header of the **Name** column. In our example, we have tables called **tblEastSales**, **tblNorthSales**, **tblSouthSales** and **tblWestSales**. We only want those tables to be included in our data set, so we de-select all the other tables. The fastest way is to click the 'Select all' item, which removes all the tick marks. Then tick the tables that you want. Press OK

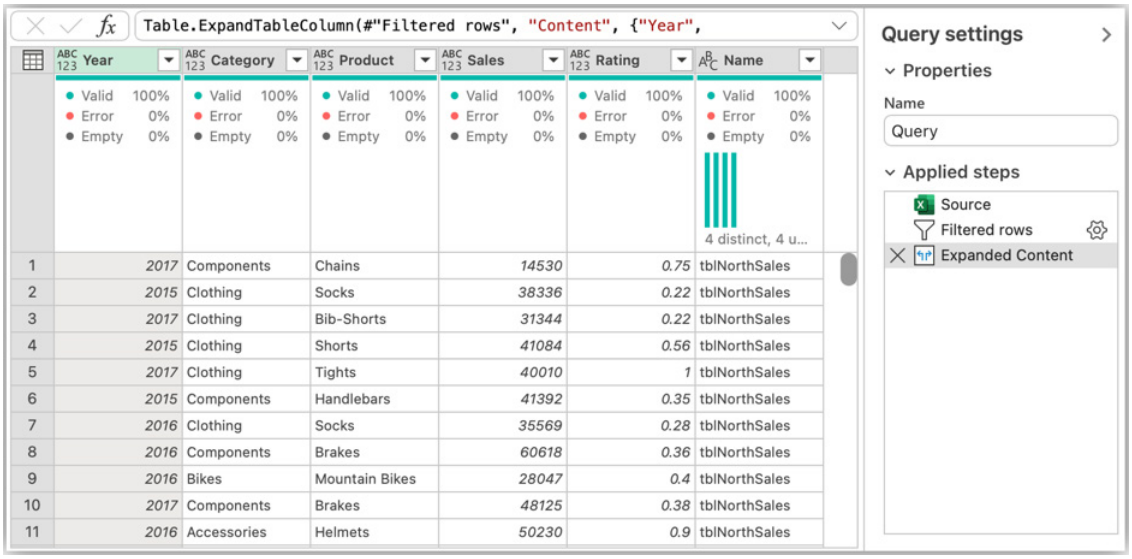


- Expand the **Content** column. Do this by pressing the button on the right side of the column header, then press OK.

The screen shot below shows the Expand button being clicked in the header of the **Content** column. The dialog shows the available columns that will be included in the expanded data. If any of the columns aren't needed for your PivotTable, you can de-select them.

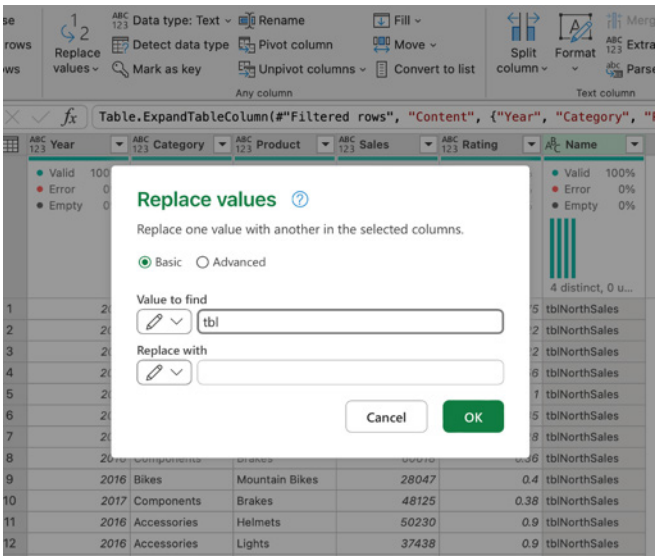
Notice that the **Name** column shows the table name that was the source range for each value



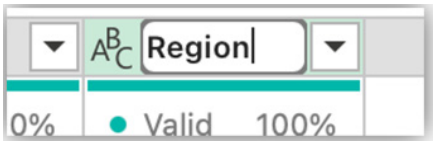


	Year	Category	Product	Sales	Rating	Name
1	2017	Components	Chains	14530	0.75	tblNorthSales
2	2015	Clothing	Socks	38336	0.22	tblNorthSales
3	2017	Clothing	Bib-Shorts	31344	0.22	tblNorthSales
4	2015	Clothing	Shorts	41084	0.56	tblNorthSales
5	2017	Clothing	Tights	40010	1	tblNorthSales
6	2015	Components	Handlebars	41392	0.35	tblNorthSales
7	2016	Clothing	Socks	35569	0.28	tblNorthSales
8	2016	Components	Brakes	60618	0.36	tblNorthSales
9	2016	Bikes	Mountain Bikes	28047	0.4	tblNorthSales
10	2017	Components	Brakes	48125	0.38	tblNorthSales
11	2016	Accessories	Helmets	50230	0.9	tblNorthSales

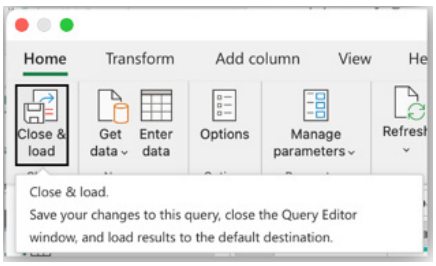
- If needed, you can use all the other features of Power Query to transform your data. You can't do any of that with the wizard, which is why Power Query is a great alternative
- In our example, we want to simplify the values in the **Name** column (region). Since the table names are like **tblNorthSales**, we want to get rid of "**tbl**" and "**sales**". Press the 'Replace Values' button, then type "**tbl**" in the 'Value to find' field and leave the 'Replace with' field empty. Remove the word "**Sales**" in a similar way to be left with just the region. Power Query applies these steps to all the data, so we only need to do it once



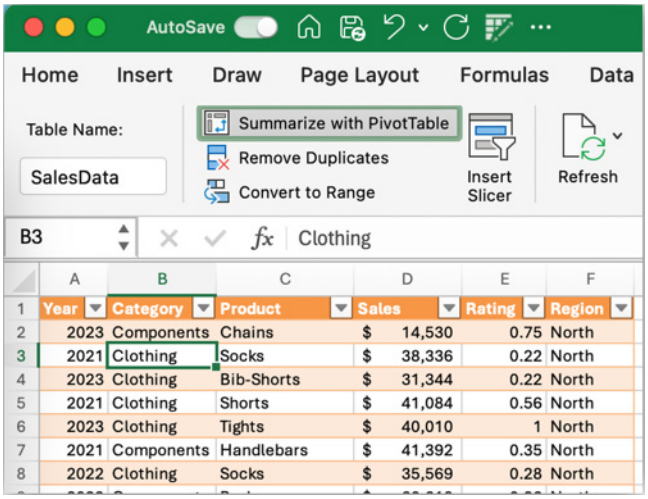
- You may want to rename a column if it makes sense. In our example, we want to rename the **Name** column to **Region**, since that's what we'll see in our PivotTable. Just double-click on the column header and type the new name



- After you've added any other steps you want, press the 'Close & load' button on the Ribbon

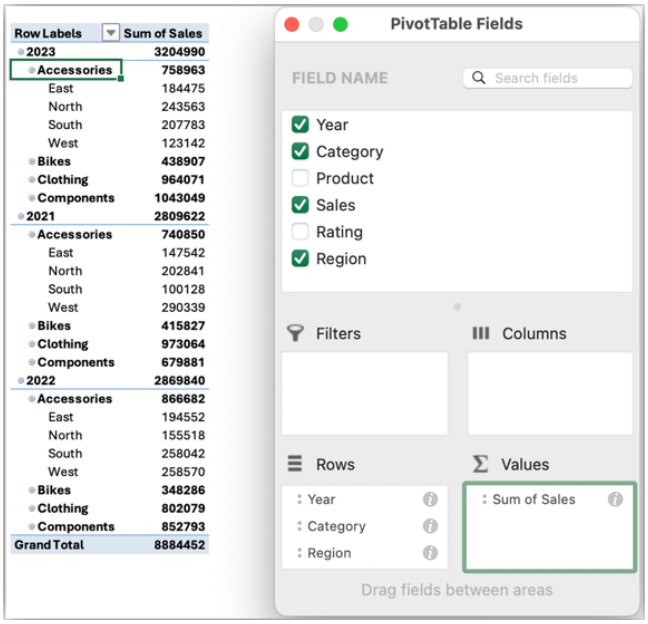


- Your data will be loaded to a new worksheet or the 'Load Data' dialog may appear (as of this writing, the dialog is not available on Mac)
- Now you can create your PivotTable with the combined data from all the ranges. The **Region** column allows us to use the **Region** in our PivotTable, even though it's not in the source tables that we combined



	A	B	C	D	E	F
1	Year	Category	Product	Sales	Rating	Region
2	2023	Components	Chains	$ 14,530	0.75	North
3	2021	Clothing	Socks	$ 38,336	0.22	North
4	2023	Clothing	Bib-Shorts	$ 31,344	0.22	North
5	2021	Clothing	Shorts	$ 41,084	0.56	North
6	2023	Clothing	Tights	$ 40,010	1	North
7	2021	Components	Handlebars	$ 41,392	0.35	North
8	2022	Clothing	Socks	$ 35,569	0.28	North

Here, you can see our PivotTable after combining our ranges and adjusting the region field to be more readable.



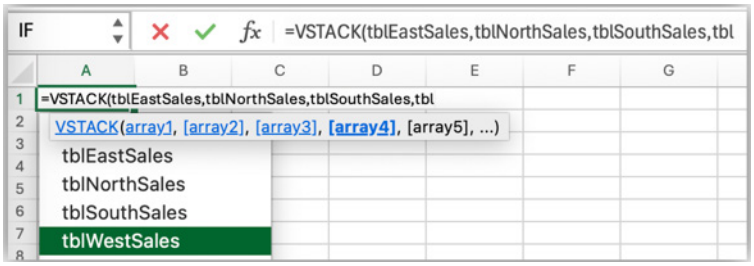
Row Labels	Sum of Sales
2023	3204990
Accessories	758963
East	184475
North	243563
South	207783
West	123142
Bikes	438907
Clothing	964071
Components	1043049
2021	2809622
Accessories	740850
East	147542
North	202841
South	100128
West	290339
Bikes	415827
Clothing	973064
Components	679881
2022	2869840
Accessories	866682
East	194552
North	155518
South	258042
West	258570
Bikes	348286
Clothing	802079
Components	852793
Grand Total	8884452

Combine Ranges with Array formulae

Another way to combine your ranges is by using the **VSTACK** function. In our example, it's very simple to combine the ranges by using the following formula:

=VSTACK(tblEastSales,tblNorthSales,tblSouthSales,tblWestSales)

Since our sales tables have names like **tblEastSales**, it's easy to just list the tables that we want, and they all get combined into a large data set.



If you use this method, you'll likely want to add some data, such as column headers, and a column to each table to show the region.

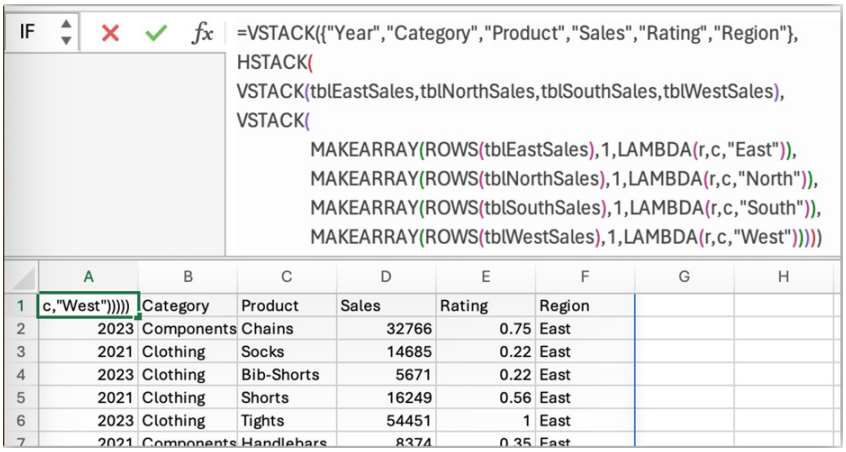
There are a few ways you can add the region, such as:

- add a region column to your sales tables and enter the region for all the rows in each table
- use the **HSTACK**, **MAKEARRAY** and **ROWS** functions to add the regions to the combined data. The **ROWS** function gives the count of rows in the specified range. The formula for the combined data will be:

```
=HSTACK(VSTACK(tblEastSales,tblNorthSales,tblSouthSales,tblWestSales),
VSTACK(
        MAKEARRAY(ROWS(tblEastSales),1,LAMBDA(r,c,"East")),
        MAKEARRAY(ROWS(tblNorthSales),1,LAMBDA(r,c,"North")),
        MAKEARRAY(ROWS(tblSouthSales),1,LAMBDA(r,c,"South")),
        MAKEARRAY(ROWS(tblWestSales),1,LAMBDA(r,c,"West"))))
```

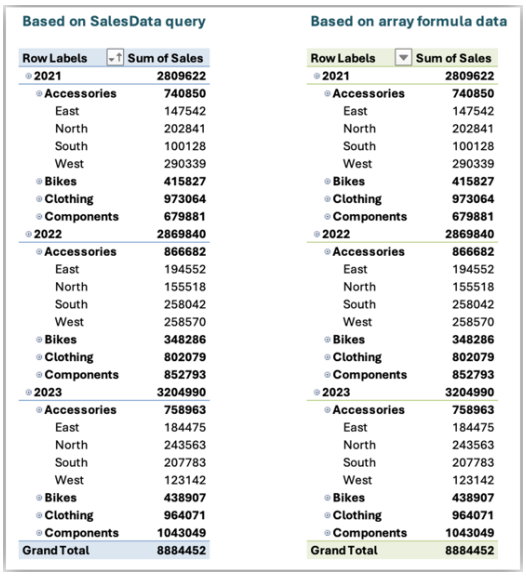
To add the column headers, you could either just type the headers in the row above the formula, or you can add to the formula, such as:

```
=VSTACK({"Year","Category","Product","Sales","Rating","Region"},
HSTACK(VSTACK(tblEastSales,tblNorthSales,tblSouthSales,tblWestSales),
        VSTACK(
                MAKEARRAY(ROWS(tblEastSales),1,LAMBDA(r,c,"East")),
                MAKEARRAY(ROWS(tblNorthSales),1,LAMBDA(r,c,"North")),
                MAKEARRAY(ROWS(tblSouthSales),1,LAMBDA(r,c,"South")),
                MAKEARRAY(ROWS(tblWestSales),1,LAMBDA(r,c,"West")))))
```



	A	B	C	D	E	F	G	H
1	c,"West")))))	Category	Product	Sales	Rating	Region		
2	2023	Components	Chains	32766	0.75	East		
3	2021	Clothing	Socks	14685	0.22	East		
4	2023	Clothing	Bib-Shorts	5671	0.22	East		
5	2021	Clothing	Shorts	16249	0.56	East		
6	2023	Clothing	Tights	54451	1	East		
7	2021	Components	Handlebars	8374	0.35	East		

Here, our screen shot shows two [2] PivotTables that have the same values. The data source for one is the **SalesData** query that we created in Power Query, and the data source for the other is the array formula described above.



Based on SalesData query

Row Labels	Sum of Sales
2021	2809622
Accessories	740850
East	147542
North	202841
South	100128
West	290339
Bikes	415827
Clothing	973064
Components	679881
2022	2869840
Accessories	866682
East	194552
North	155518
South	258042
West	258570
Bikes	348286
Clothing	802079
Components	852793
2023	3204990
Accessories	758963
East	184475
North	243563
South	207783
West	123142
Bikes	438907
Clothing	964071
Components	1043049
Grand Total	8884452

Based on array formula data

Row Labels	Sum of Sales
2021	2809622
Accessories	740850
East	147542
North	202841
South	100128
West	290339
Bikes	415827
Clothing	973064
Components	679881
2022	2869840
Accessories	866682
East	194552
North	155518
South	258042
West	258570
Bikes	348286
Clothing	802079
Components	852793
2023	3204990
Accessories	758963
East	184475
North	243563
South	207783
West	123142
Bikes	438907
Clothing	964071
Components	1043049
Grand Total	8884452

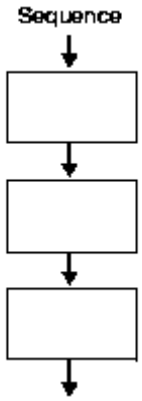
We'll continue next month...

Visual Basics

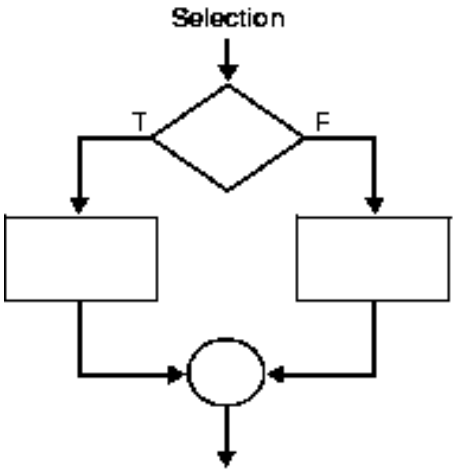
We thought we'd run an elementary series going through the rudiments of Visual Basic for Applications (VBA) as a springboard for newer users. This month, we take a peek at control structures.

In programming, a control structure determines the order in which statements are executed. Control structures can be grouped into three main categories:

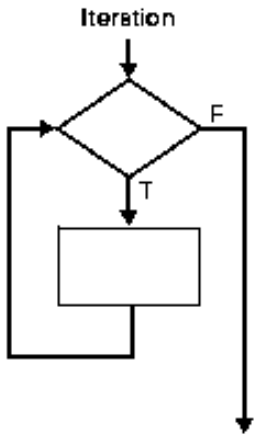
1. **Sequential:** Sequential execution is where each statement in the source code will be executed one by one in a sequential order. This is the default mode of execution



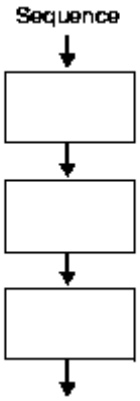
2. **Selection:** The selection control structure is used for making decisions and branching statements



3. **Iteration:** The iterative control structures are used for repetitively executing a block of code multiple times



This month we will look at sequential execution. This is where each statement in the source code will be executed one by one in a sequential order. This is the default mode of execution.



Here are a couple of prime examples.

IF…THEN…ELSE

This is the most basic sequential control structure. Identical to the **IF** function in Excel, this calls on the most basic decision path – execute steps if a condition is met, otherwise execute other steps.

```
If expression Then
 <statements>
Else
 <statements>
End If
```

Here's an example:

```
Option Explicit
Sub IFStatement()
    Dim myNumber As Integer
    myNumber = 3
    If myNumber > 0 Then
        Debug.Print "Number is Positive!"
    Else
        Debug.Print "Number is Negative!"
    End If
End Sub
```

Note however that the **ELSE** statement is not mandatory, which is the same as the **IF** function in Excel. If no statements need to be executed should the condition prove false, then the **ELSE** section can be omitted entirely.

ELSE IF

Sometimes users are required to test for more than one condition. Quite often in Excel, one comes across multiple **IF** statements chained together in one formula.

```
IF( condition1, value_if_true1, IF( condition2, value_if_true2, value_if_false2 ))
```

Often this is referred to as "nested **IF**s". The idea of nesting comes from embedding or "nesting" one **IF** function inside another. In VBA, this can be similarly achieved by using the **ELSEIF** statement.

The above example would be written in VBA as:

```
IF condition1 THEN
   value_if_true1
ELSEIF condition2 THEN
   value_if_true2
ELSE
   value_if_false2
END IF
```

Here's a working VBA code sample:

```
Option Explicit
Sub IFStatement()
    Dim myNumber As Integer
    myNumber = 3
    If myNumber < 10 Then
        Debug.Print "Number is single digit!"
    Else If myNumber < 100
        Debug.Print "Number is double digit!"
    Else
        Debug.Print "Number is very large!"
    End If
End Sub
```

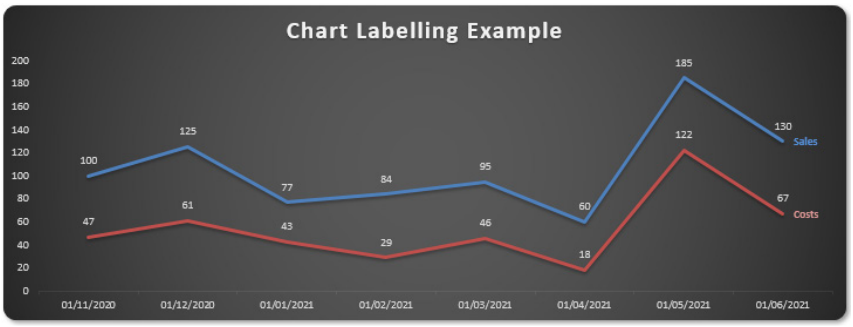
More next time.

Charts and Dashboards

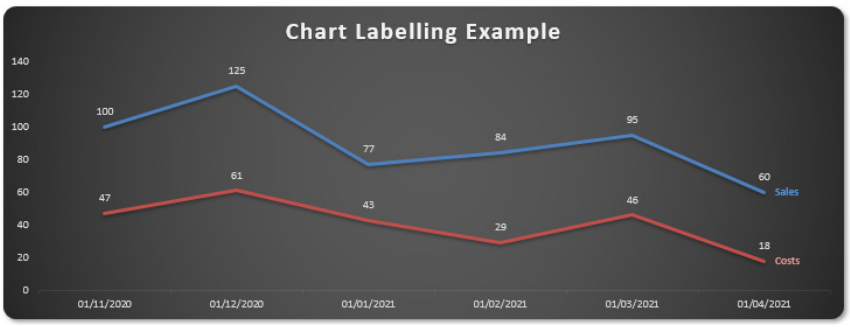
It's time to chart our progress with an introductory series into the world of creating charts and dashboards in Excel. This month, we show you how to create dynamic chart labels.

Here, we have created a straightforward Line chart in Excel and want to show both the data values and label the chart so that the label attaches itself to the position of the final data point rather than in a legend somewhere.

In this example, the aim is to attach the label for the chart series to the series itself, *e.g.*



As you can see from the above figure, the Line chart's label appears on the right-hand side next to the final data point in the series. If the values were to change, the label would move too:



You may note that the final period is earlier and that the values for the final point differ – yet the labels hang on steadfastly.

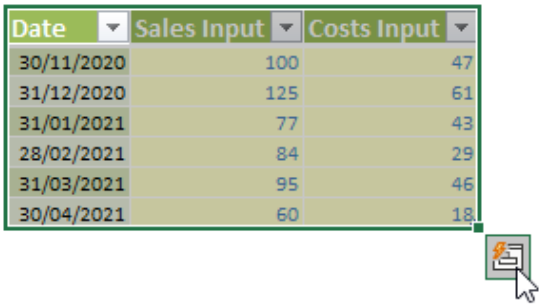
So how do we do this?

First of all, make sure your chart data is collated in a Table.

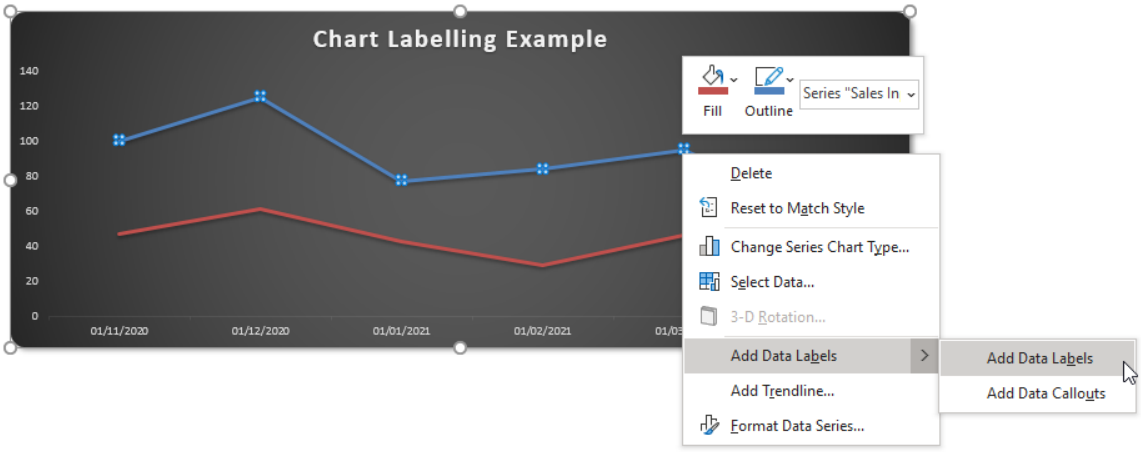
Date	Sales Input	Costs Input
30/11/2020	100	47
31/12/2020	125	61
31/01/2021	77	43
28/02/2021	84	29
31/03/2021	95	46
30/04/2021	60	18

The benefit is that if charts are linked to a Table range and the range is contracted or extended, the dependent chart will update automatically without having to use the 'Select Data…' functionality.

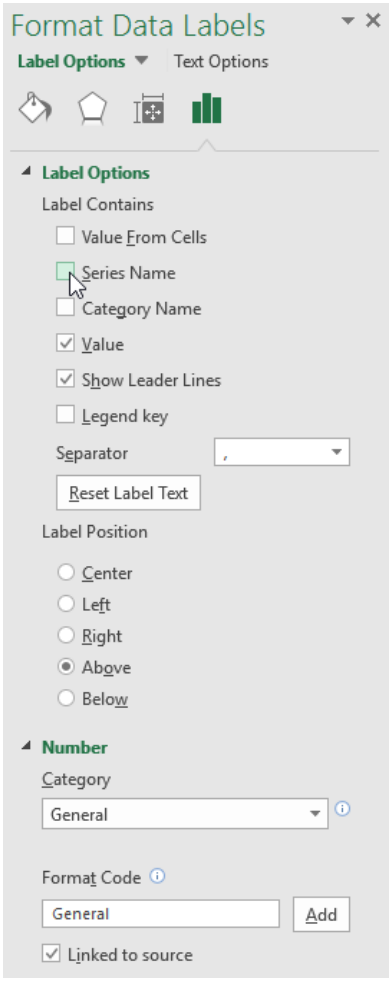
Creating a Line chart from this Table is trivial. Simply highlight the Table and click on the 'Quick Analysis' tool in the bottom right-hand corner (**CTRL + Q**):



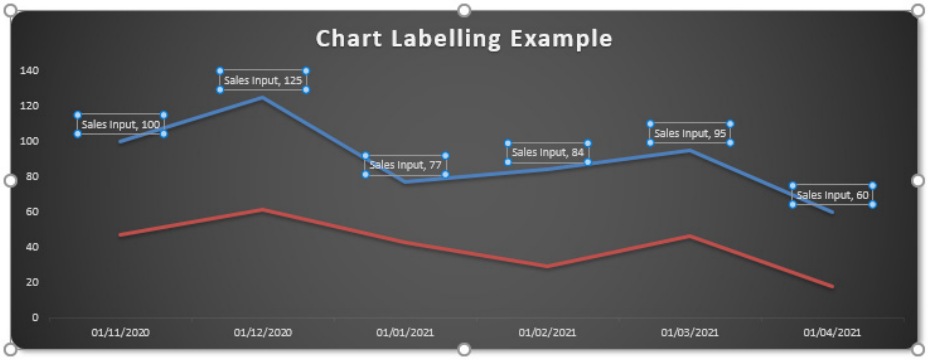
Following the resultant prompts leads to a Line chart in all of about two seconds. Right-clicking on one of the data series then allows you to add data labels:



By default, Excel adds the values. Right-clicking on these values and selecting 'Format Data Labels…' from the pop-up context menu triggers the 'Format Data Labels' task pane and allows you to choose **what** should be displayed (*e.g.* value, series name, category name), **where** it should be displayed (*e.g.* left of the data point, above it, below it) and **how** it should be displayed (format to use):

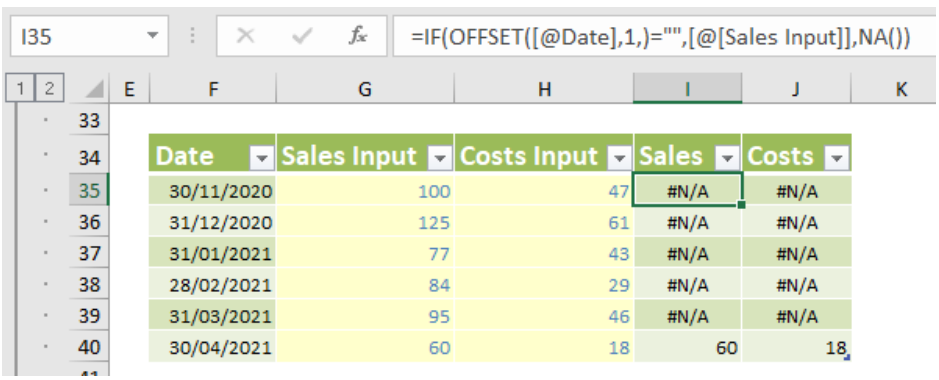


Selecting 'Series Name' in 'Label Options' *almost* provides what is required:



The problem here is that we only require the series name against the final data point. If we need this to be flexible, manual adjustment is insufficient: we need to cheat.

'Cheating' requires us to add two more columns to the underlying data Table:



	Date	Sales Input	Costs Input	Sales	Costs
35	30/11/2020	100	47	#N/A	#N/A
36	31/12/2020	125	61	#N/A	#N/A
37	31/01/2021	77	43	#N/A	#N/A
38	28/02/2021	84	29	#N/A	#N/A
39	31/03/2021	95	46	#N/A	#N/A
40	30/04/2021	60	18	60	18

Assuming the first three columns have been labelled **Date**, **Sales Input** and **Costs Input** as in the example illustration above, we add two more columns, **Sales** and **Costs** (*N.B.* in a Table, the same column heading may not be used more than once). The formulae in the two columns would be as follows:

=IF(OFFSET([@Date],1,)="",[@[Sales Input]],NA()) *and*

=IF(OFFSET([@Date],1,)="",[@[Costs Input]],NA())

The **@** symbol for Tables in Excel 2010 onwards signifies that the formula is referring to the data point for that field in that row, *e.g.* the formula highlighted in cell **I35** in the image is essentially

=IF(OFFSET(F35,1,)="",G35,NA())

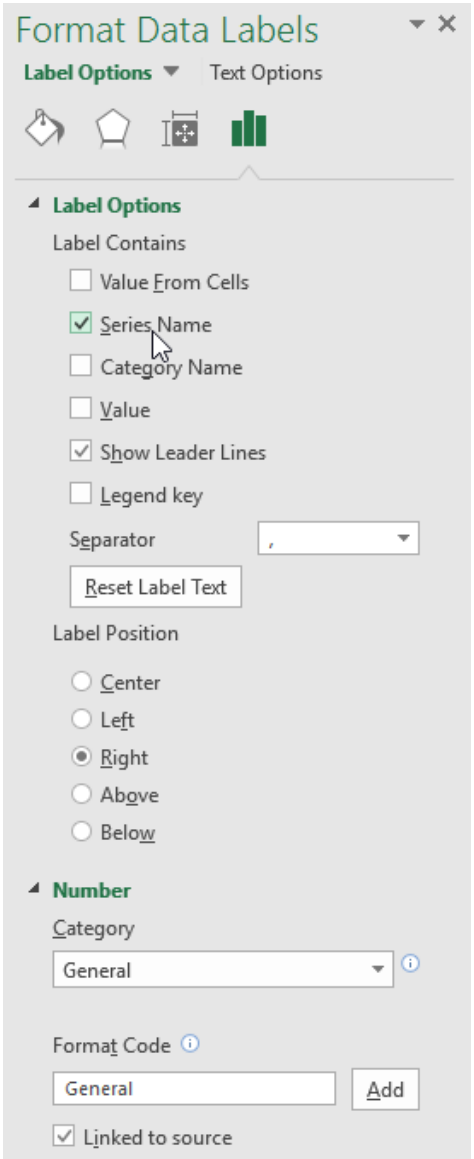
which is effectively

=IF(F36="",G35,NA())

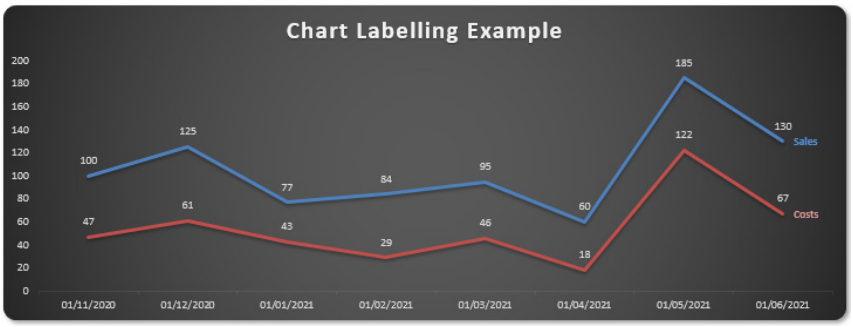
This is necessary as Table formulae do not like calculations linking to other rows.

These formulae cause the corresponding **Sales** and **Costs** values only to appear in the final row of the table, with *#N/A* elsewhere. Normally, we would strongly recommend against having *prima facie* errors in an Excel file, but here they are useful – this is the syntax required for these data points to be ignored by the chart engine.

You are almost done. All that is left is to highlight the revised Table and recreate the line chart from earlier. After selecting the Data Labels for the **Sales Input** and **Costs Input** series (just as previously), you simply add Data Labels for the two new "series" **Sales** and **Costs** even though they are singleton points:



Et voila! You have created the chart required:



More next time.

Power Pivot Principles

We continue our series on the Excel COM add-in, Power Pivot. This month, we show you how to set up slicers to dynamically segregate text data based upon text inputs.

The **RANKX** function returns with the ranking of the number values in a column in a table. It uses the following syntax:

RANKX(table, expression [, value, order, ties,])

The **RANKX** requires a **table** and an **expression** input. The **value**, **order**, and **ties** inputs are optional.

For this example, we are going to look at the following table, note that this table (an Excel Table) is given the name '**ProductListKitchen**' in Excel:

Product Type	Price
Steel Baking Tray	20.00
Premium 40cm Pan	120.00
Dinner Set	35.00
Waffle Maker	125.00
Rice Cooker Premium	160.00
5,000 Piece Napkin	20.00
10,000 Piece Napkins	30.00
Rice Cooker	85.00
Dish Washing Capsules 100 pack	35.00
Slow Cooker	140.00
6 Slice Toaster (Imported)	120.00
Water Filter	50.00
Small Baking Dish	12.00
Filter Water Bottle	15.00
Chef Knife	90.00
Enviro-friendly Dishwashing Capsules 200 Pack	20.00
Can Opener	3.00
Fine China Dining Set	80.00
Mortar & Pestle	20.00
Stainless Steel Pot	60.00
Kitchen Knife Set	80.00

Imagine that we wanted to rank the products in this Excel Table from the highest to lowest price in Power Pivot. Here we may use the **RANKX** function.

After adding the Table to the data model, we can create a calculated column with the following **DAX** code:

```
=RANKX(
        ProductListKitchen,
        [Price]
        )
```

The **table** input in this code is our table '**ProductListKitchen**', because this is the Table that we wish to evaluate. The **expression** is the column that contains the values that we wish to be ranked, and that would be the **Price** column.

The resulting calculated column will then look like this:

[Price Rank] f_x =RANKX(
ProductListKitchen,
[Price]
)

	Product Type	Price	Price Rank
1	Steel Baking Tray	20	15
2	Premium 40cm Pan	120	4
3	Dinner Set	35	12
4	Waffle Maker	125	3
5	Rice Cooker Premium	160	1
6	5,000 Piece Napkin	20	15
7	10,000 Piece Napkins	30	14
8	Rice Cooker	85	7

We can then sort the items:

[Price Rank] f_x =RANKX(
ProductListKitchen,
[Price]
)

	Product Type	Price	Price Rank
1	Rice Cooker Premium	160	1
2	Slow Cooker	140	2
3	Waffle Maker	125	3
4	Premium 40cm Pan	120	4
5	6 Slice Toaster (Imported)	120	4
6	Chef Knife	90	6
7	Rice Cooker	85	7
8	Fine China Dining Set	80	8

The default ranking is applied in descending order where **RANKX** ranks the product type with the highest price first. To swap the ranking order around we have to add 'ASC', which is short for ascending, as the **order** input:

```
=RANKX(
        ProductListKitchen,
        [Price],
        ,
        ASC
        )
```

The resulting table will now look like this:

[Price Rank] =RANKX(
ProductListKitchen,
[Price],
,
ASC
)

	Product Type	Price	Price Rank
1	Can Opener	3	1
2	Small Baking Dish	12	2
3	Filter Water Bottle	15	3
4	Steel Baking Tray	20	4
5	5,000 Piece Napkin	20	4
6	Enviro-friendly Dishwas...	20	4
7	Mortar & Pestle	20	4
8	10,000 Piece Napkins	30	8

There we have it: we have ranked our products by ascending price in our data table.

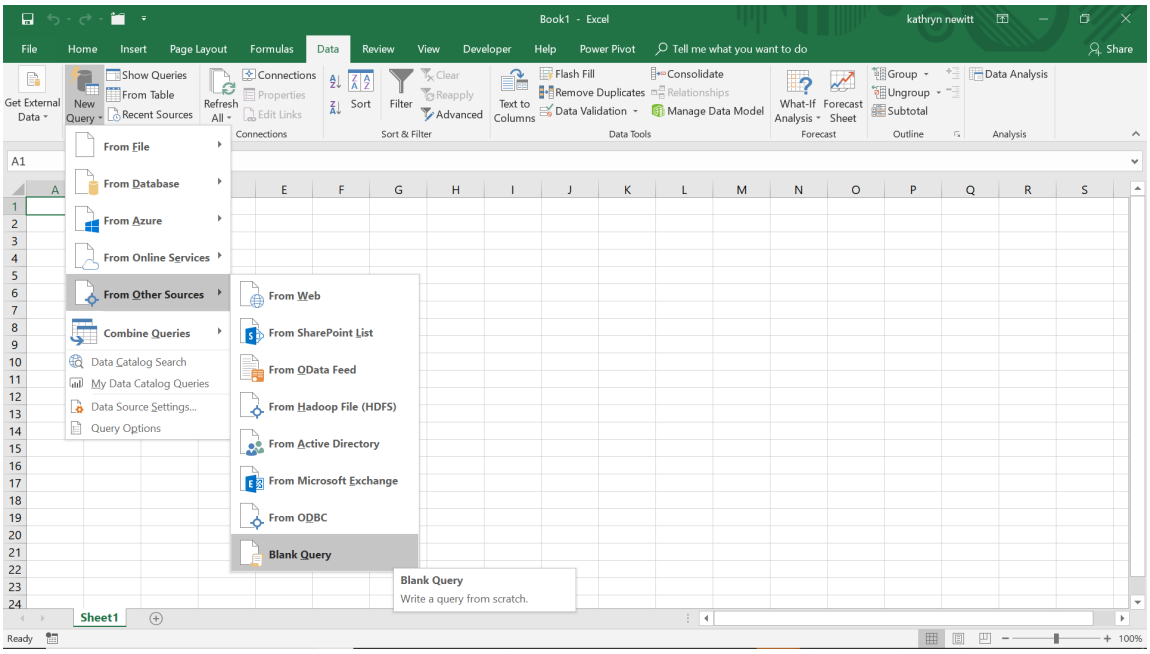
That's it for this month; more next time.

Power Query Pointers

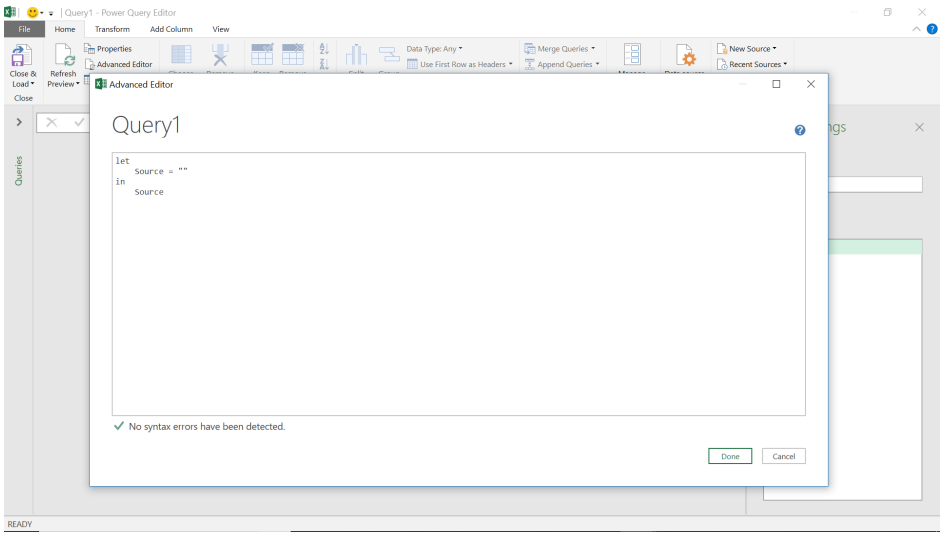
*Each month we'll reproduce one of our articles on Power Query (Excel 2010 and 2013) / Get & Transform (Office 365, Excel 2016 and 2019) from www.sumproduct.com/blog. If you wish to read more in the meantime, simply check out our Blog section each Wednesday. This month, we consider some of the (most) simple **M** statements.*

With IntelliSense borne in mind, this month we're looking at how to get to grips with **M**. Here, let's consider just how basic a piece of **M** code can be.

We'll start by creating a new Blank Query from the 'New Query' option on the 'Get & Transform' section of the 'Data' tab. In the 'From Other Sources' dropdown, we may choose to create a new 'Blank Query', *viz*.

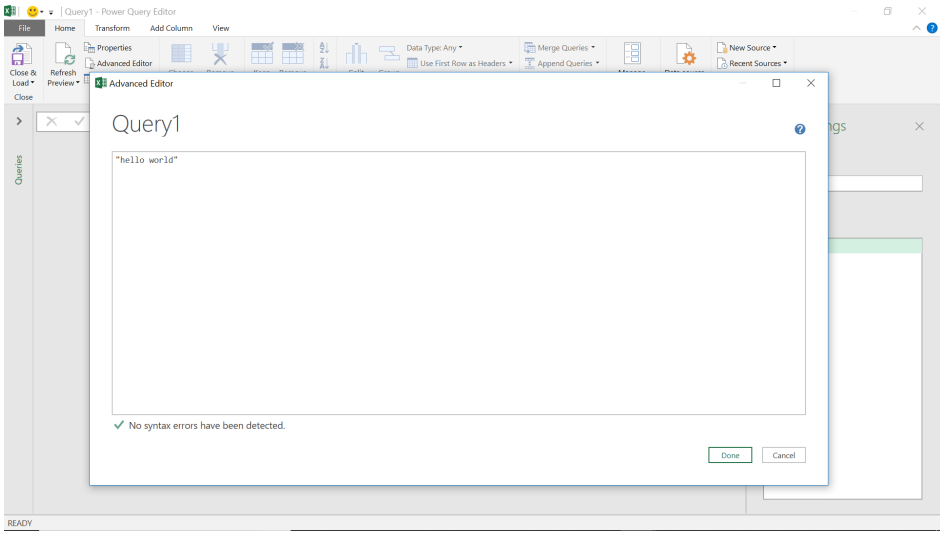


This gives rise to the following screenshot:

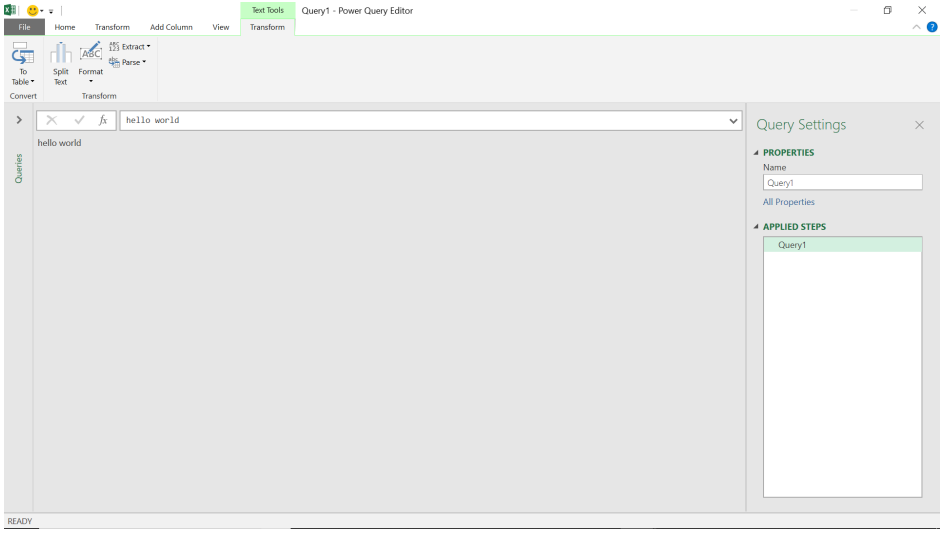


In the Advanced Editor, Power Query has automatically tried to define a source. This is because Power Query is most often used to clean up existing data from other sources. However, the source definition, and even a 'let' and 'in' statements, are not actually required.

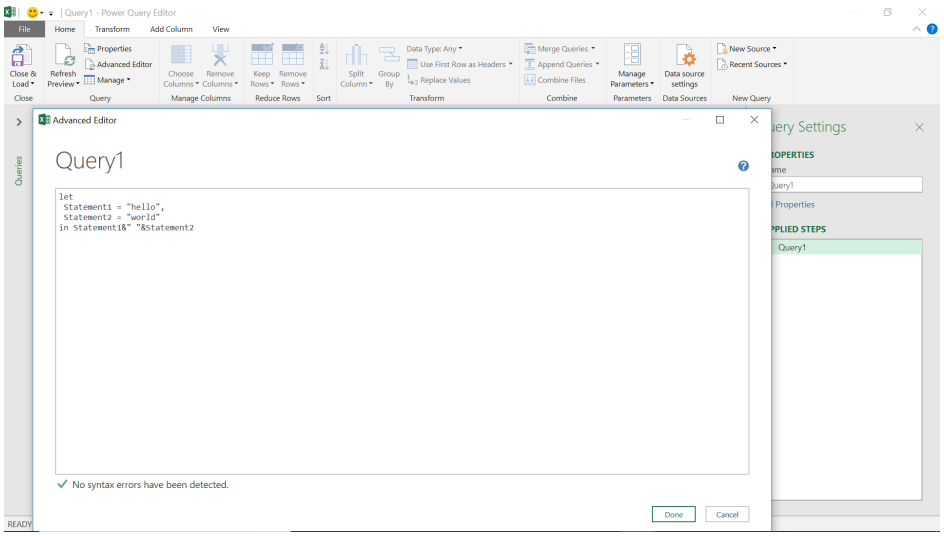
So, what shall we do instead?



It's good to stick with the classics! This is enough to be a complete piece of **M** code. If we click 'Done', the code will be executed:



So, the scarily complex **M** code can actually be very simple. We can make it slightly more complex…



I have used the following **M** code

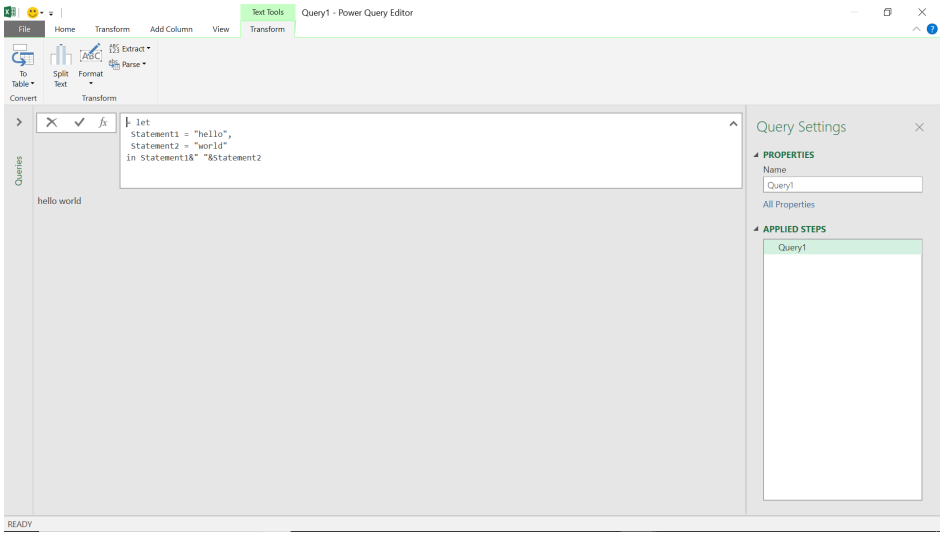
let

Statement1 = “hello”,

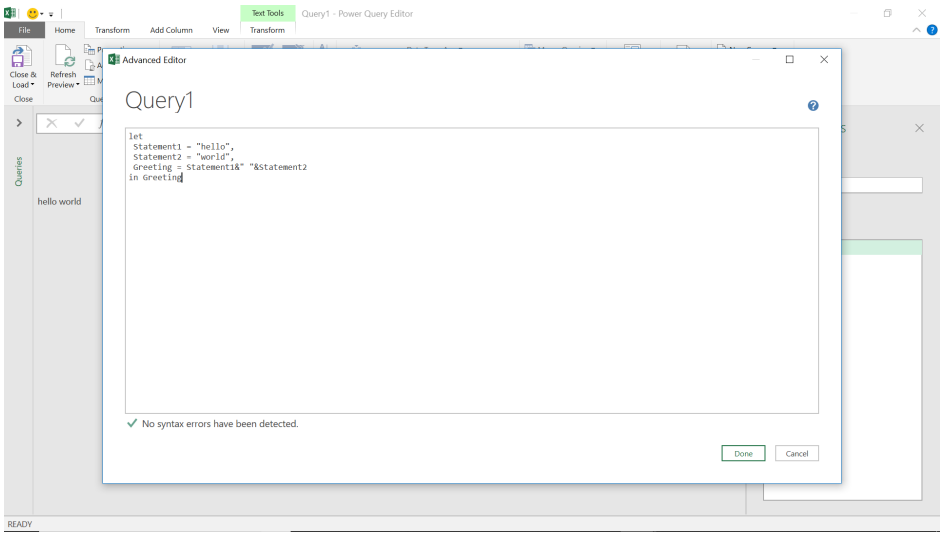
Statement2 = “world”

in Statement1&” “&Statement2

This creates variables, *i.e.* items which may have different values assigned to them. We can create a variable called **Statement1**, which we will give the value “hello” and then we may create a variable **Statement2**, which we can give the value “world’ to. The final ‘in’ statement contains the function, where I combine the variables, inserting a blank space between them. When we click ‘Done’ the result looks familiar:



We have created the same result using variables. However, in typical generated code, the ‘in’ statement doesn’t usually contain a function. So, let’s make amends.



let

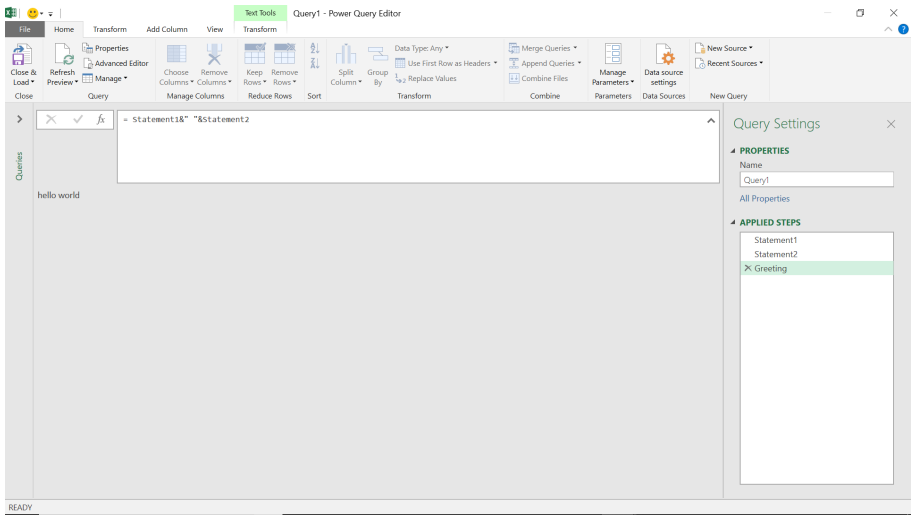
Statement1 = “hello”,

Statement2 = “world”

Greeting = Statement1&” “&Statement2

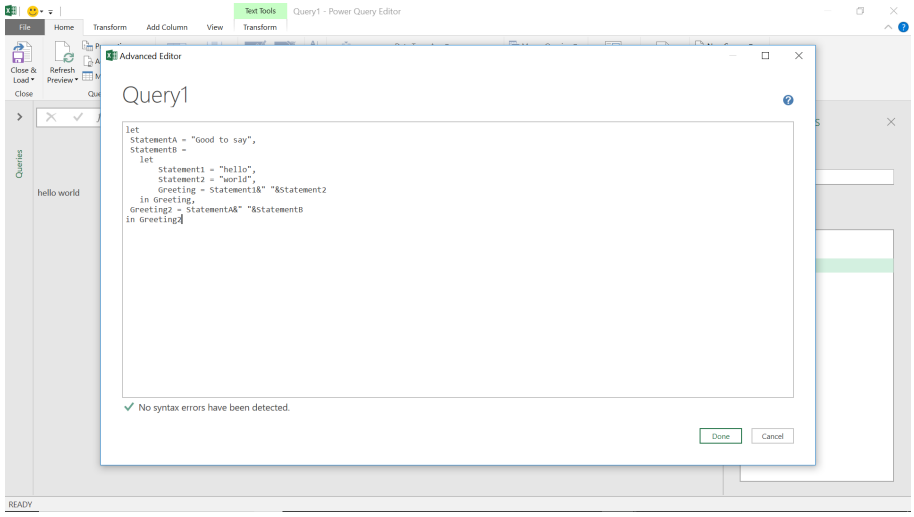
in Greeting

This format looks more like the Power Query generated code, and should give the same result when we click ‘Done’:



It is the same result on the main screen, but in the applied steps, there are now three – we define **Statement1**, then we define **Statement2**, then we put them together in **Greeting**. This makes it much easier to edit the steps without going to the main editor, and the sequence of the logic is clear(er) to other users. These steps are created by Power Query if the ‘in’ step has a single function associated with it. In order to get the most out of Power Query’s built-in features, it’s best to mimic the way that Power Query creates **M** code.

Variable creation can be nested – this is like having boxes within boxes. We can take my ‘hello world’ box and put in in another variable:



let

StatementA = “Good to say”

StatementB =

let

Statement1 = “hello”,

Statement2 = “world”

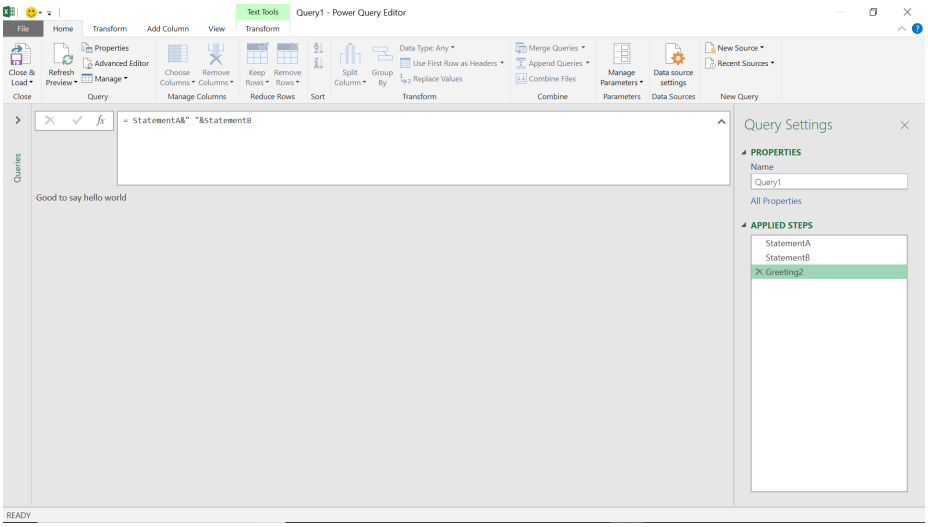
Greeting = Statement1&” “&Statement2

in Greeting,

Greeting2 = StatementA&” “&StatementB

in Greeting2

We have nested my ‘hello world’ box in a bigger statement.



Our longer phrase appears. Nesting removes the steps in the nested box from the 'Applied Steps' list. This can make queries appear simpler as well as organising the **M** code in the editor – it really depends upon what you are trying to achieve.

Until next month.

Power BI Updates

This month's updates see new On-Object Interaction updates, DAX query view now Generally Available, and you can find out how to view reports in OneDrive and SharePoint with live connected semantic models.

The full list of updates is as follows:

Reporting

- Modern ToolTip now on by default
- Matrix layouts
- Line updates
- On-Object Interaction updates
- Publish to folders in Preview
- Ask Copilot questions about data in your model (Preview)

Modelling

- Announcing General Availability of DAX query view
- Copilot to write and explain **DAX** queries in DAX query view updates in Preview
- New 'Manage relationships' dialog
- Refreshing calculated columns and calculated tables referencing DirectQuery sources with Single Sign-On
- Announcing General Availability of Model Explorer and authoring calculation groups in Power BI Desktop

Data Connectivity

- Microsoft Entra ID SSO support for Oracle database
- Certified connector updates
- View reports in OneDrive and SharePoint with live connected semantic models

Service

- Storytelling in PowerPoint: Image mode in the Power BI add-in for PowerPoint
- Storytelling in PowerPoint: data updated notification

Developers

- Git Integration support for Direct Lake semantic models

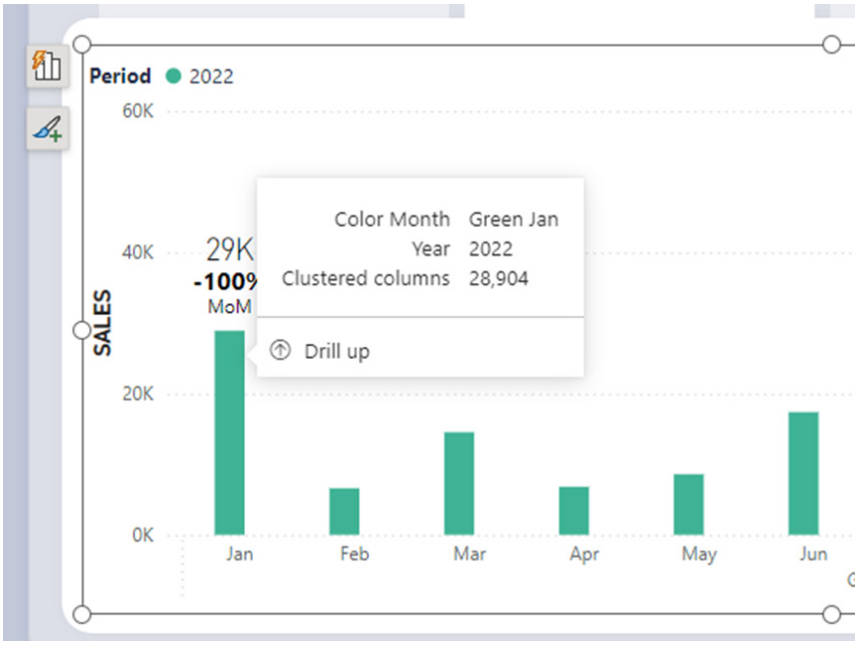
Visualisations

- New visuals in AppSource
- Financial Reporting Matrix by Profitbase
- Horizon chart by Powerviz
- Milestone Trend Analysis chart by Nova Silva
- Sunburst chart by Powerviz
- Stacked Bar chart with Line by JTA
- Drill Down Combo PRO: now with Legend field.

Let's look at each in turn.

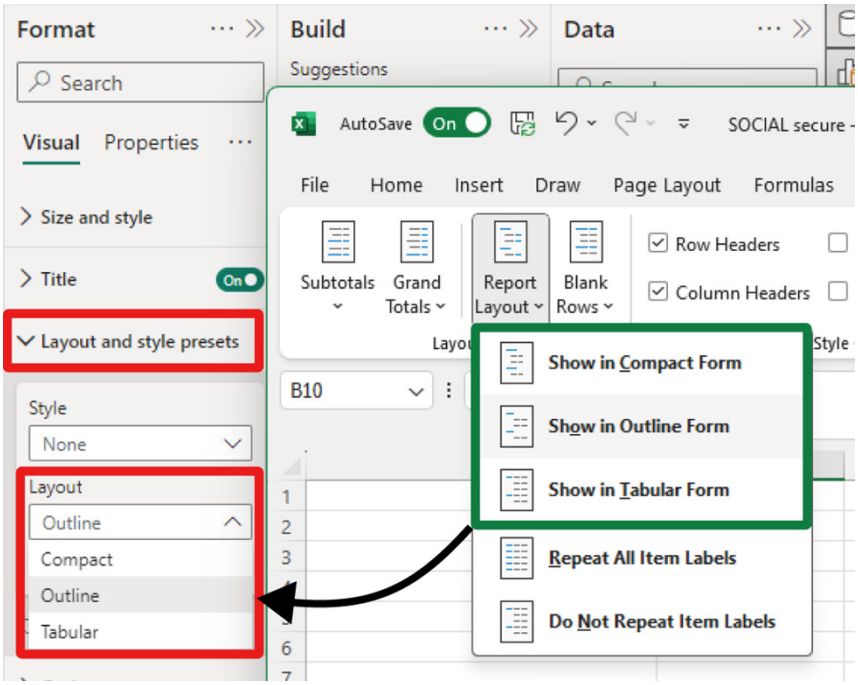
Modern ToolTip now on by default

Power BI ToolTips are being improved to enhance their functionality. To lay the groundwork, Microsoft is introducing the modern ToolTip as the new default, a feature that many users may already recognise from its previous Preview status. This change is more than just an upgrade; it's the first step in a series of improvements. These future developments intend to improve ToolTip management and customisation.

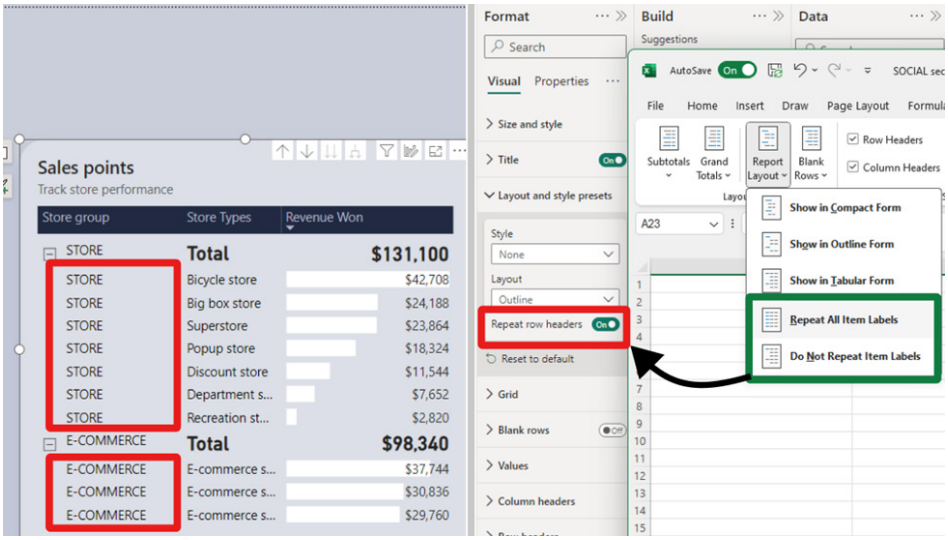


Matrix layouts

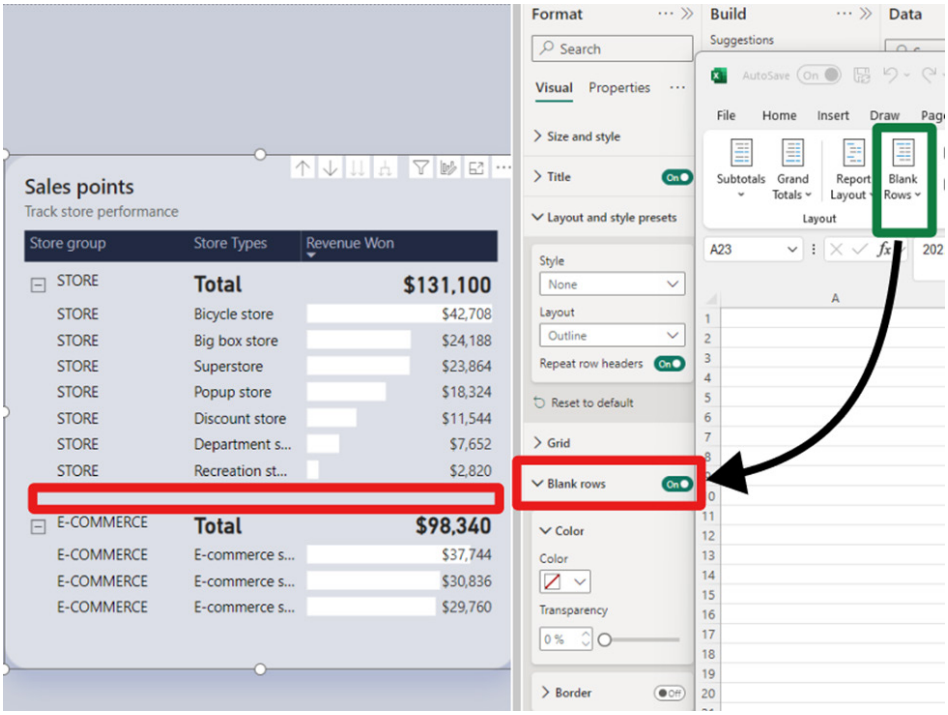
Drawing inspiration from the familiar PivotTable in Excel, this update aims to build new features and capabilities upon a stronger foundation. Here, Microsoft has introduced 'Layouts for Matrix'. Now, you can select from compact, outline or tabular layouts to alter the arrangement of components in a manner akin to Excel.



As an extension of the new layout options, report creators may now craft custom layout patterns by repeating row headers. This control, inspired by Excel's PivotTable layout, enables the creation of a matrix that closely resembles the look and feel of a table. This enhancement not only provides greater flexibility but also brings a touch of Excel's intuitive design to Power BI's Matrix visuals. This is only available for Outline and Tabular layouts.



To further align with Excel's functionality, report creators now have the option to insert blank rows within the matrix. This feature allows for the separation of higher-level row header categories, significantly enhancing the readability of the report. It's a useful addition that brings a new level of clarity and organisation to Power BI's Matrix visuals and opens a path for future enhancements for totals / subtotals and rows / column headers.



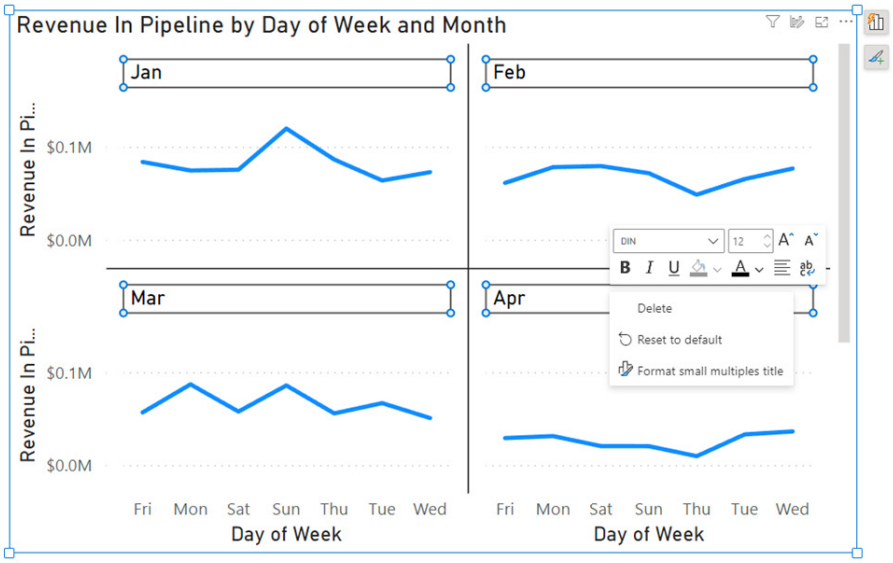
Line updates

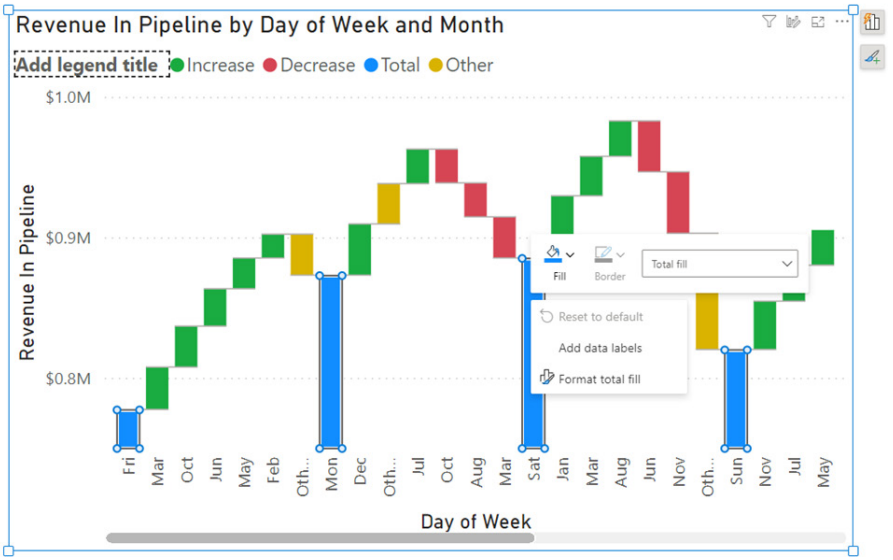
Following last month's introduction of the initial Line enhancements, this month sees further improvements:

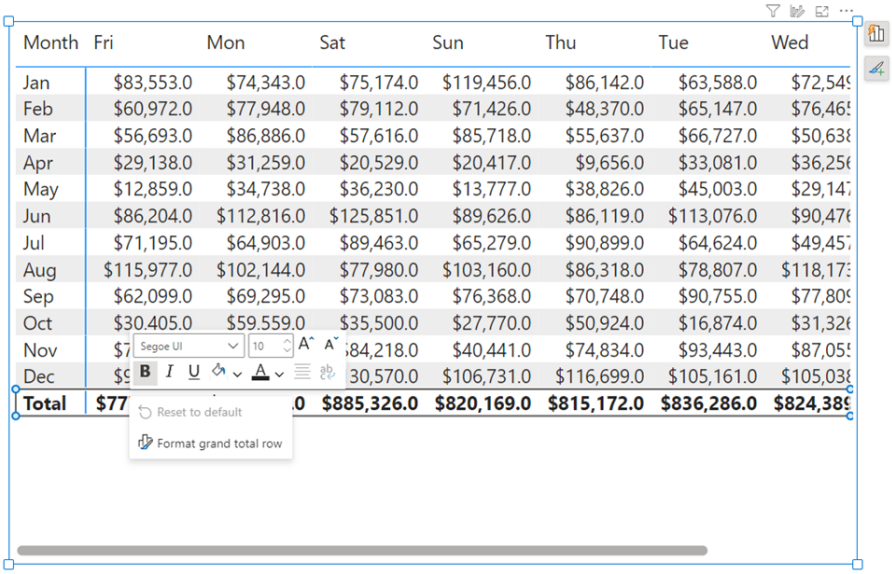
- **Hide / show lines:** gain control over the visibility of your lines for a more focused report
- **Customised line pattern:** tailors the pattern of your lines to match the style and context of your data
- **Auto-scaled line pattern:** ensures your line patterns scale with your data, maintaining consistency and clarity
- **Line dash cap:** customises the end caps of your customised dashed lines for a professional look
- **Line upgrades across other line types:** improvements have been made to reference lines, forecast lines, leader lines, small multiple gridlines and the new card's divider line.

On-Object Interaction updates

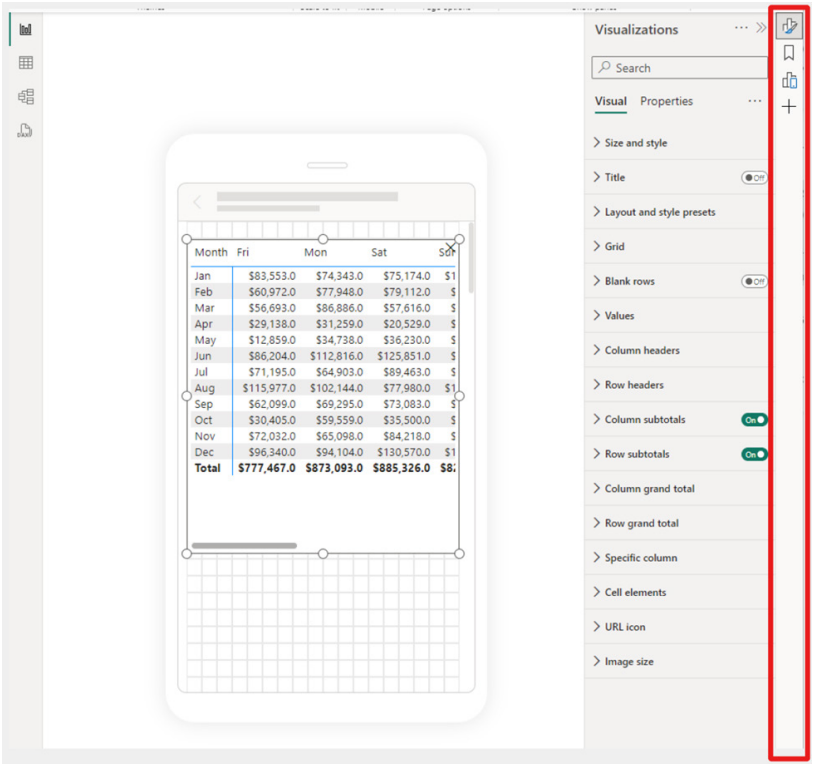
This release introduces On-Object formatting support for Small multiples, Waterfall and Matrix visuals. This new feature allows users to interact directly with these visuals for a more intuitive and efficient formatting experience. By double-clicking on any of these visuals, users may now right-click on the specific visual component they wish to format, bringing up a convenient mini-toolbar. This streamlined approach not only saves time but also enhances the user's ability to customise and refine their reports.







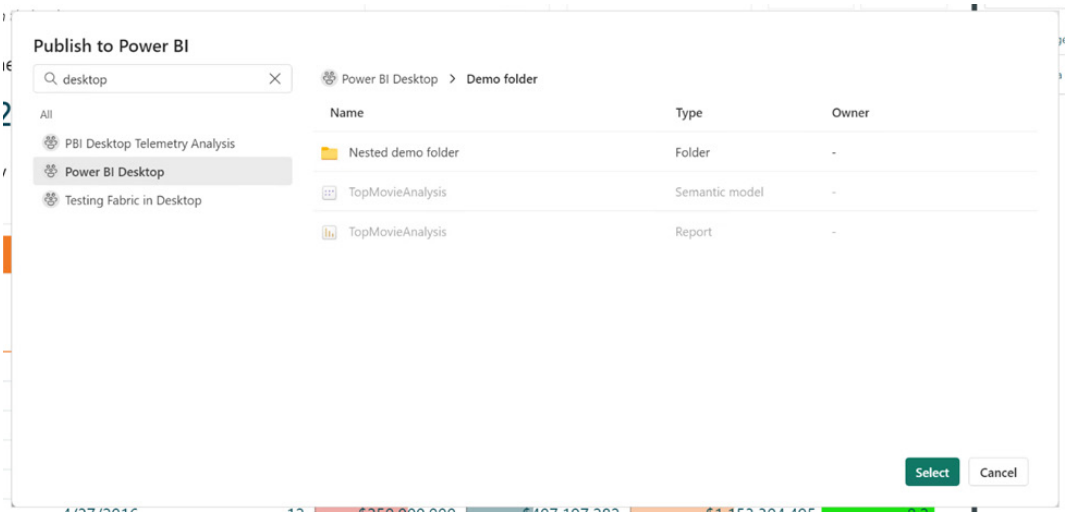
Microsoft has also announced a significant enhancement to the Mobile reporting experience with the introduction of the pane manager for the Mobile layout view. This feature empowers users to open and close panels via a dedicated menu, streamlining the design process of Mobile reports.



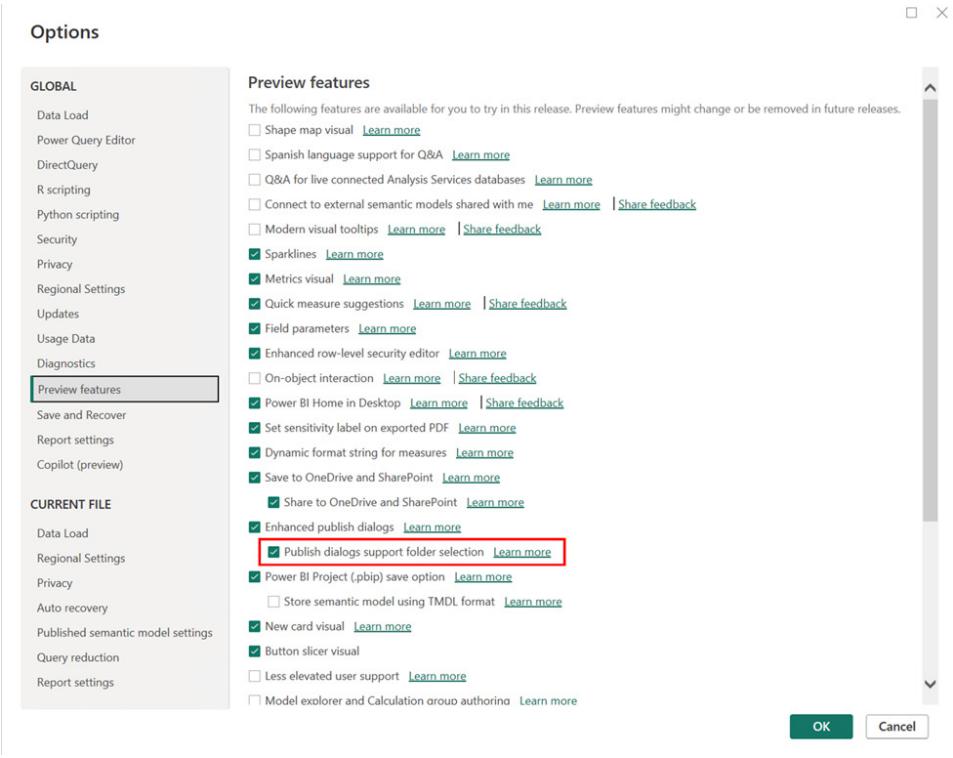
Publish to folders in Preview

Microsoft recently announced a public Preview for folders in workspaces, allowing you to create a hierarchical structure for organising and managing your items. In the latest Power BI Desktop release, you can now publish your reports to specific folders in your workspace.

When you publish a report, you can choose the specific workspace and folder for your report. The interface is simplistic and easy to understand, making organising your Power BI content more straightforward.

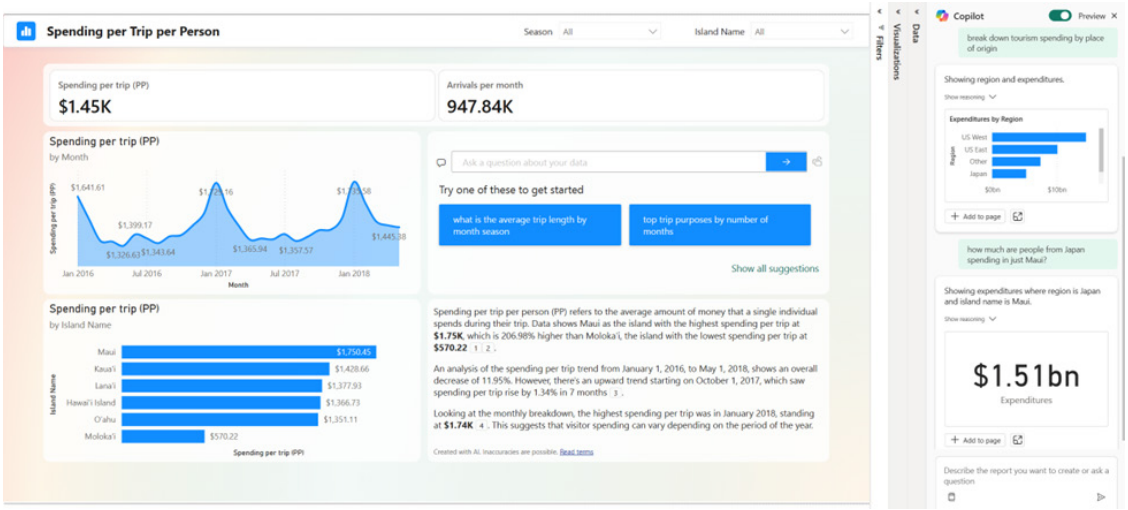


To publish reports to specific folders in Power BI Service, make sure the 'Publish dialogs support folder selection' setting is enabled in the Preview features tab in the Options menu.



Ask Copilot questions about data in your model (Preview)

In Preview, there is now a new capability for Power BI Copilot allowing you to ask questions about the data in your model. Whilst you could already ask questions about the data present in the visuals on your report pages, you can now go deeper by getting answers directly from the underlying model. Just ask questions about your data, and if the answer isn't already on your report, Copilot will then query your model for the data instead and return the answer to your question in the form of a visual.



Microsoft is initiating this capability in both Edit and View modes in Power BI Service. Since this is a Preview feature, you'll need to enable it via the Preview toggle in the Copilot pane.

Announcing General Availability of DAX query view

DAX query view has become Generally Available. DAX query view is the fourth view in Power BI Desktop to run **DAX** queries on your semantic model. It comes with several ways to help you become as productive as possible with DAX queries:

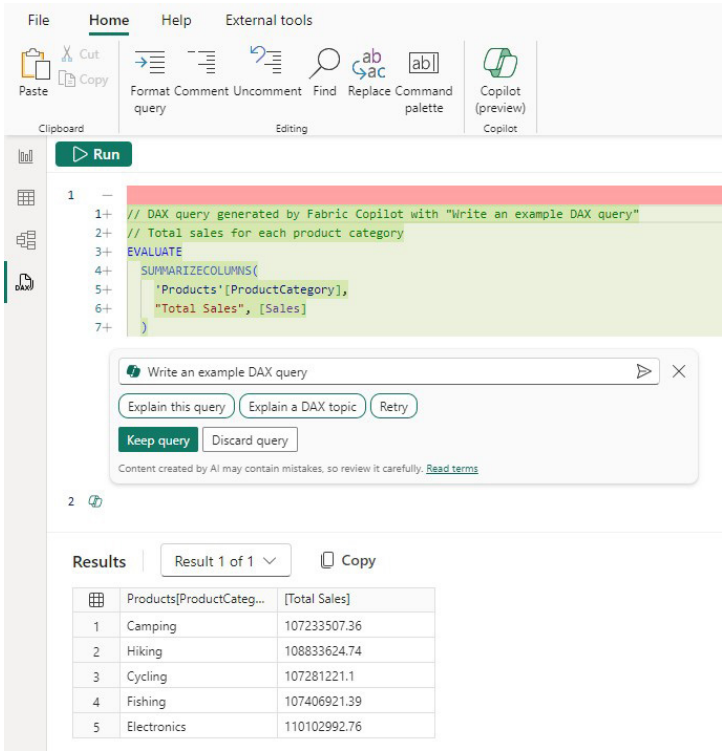
- **Quick queries:** you can have the **DAX** query written for you from the context menu of tables, columns or measures in the Data pane of DAX query view. You can obtain the top 100 rows of a table, statistics of a column or **DAX** formula of a measure to edit and validate in just a couple of clicks
- **DirectQuery model authors can also use DAX query view:** you can view the data in your tables whenever you wish
- **Create and edit measures:** edit one or multiple measures at once. Make changes and see the change in action in a **DAX** query. Then, update the model when you are ready from DAX query view

- **See the DAX query of visuals:** investigate the visuals **DAX** query in DAX query view. Go to the 'Performance Analyzer' pane and choose 'Run in DAX query view'
- **Write DAX queries:** you may now create **DAX** queries with IntelliSense, formatting, commenting / uncommenting and syntax highlighting. There are also additional professional code editing experiences such as 'Change all occurrences' and block folding to expand and collapse sections. Ad flavour of the month regex is also possible with 'Find and Replace' options.

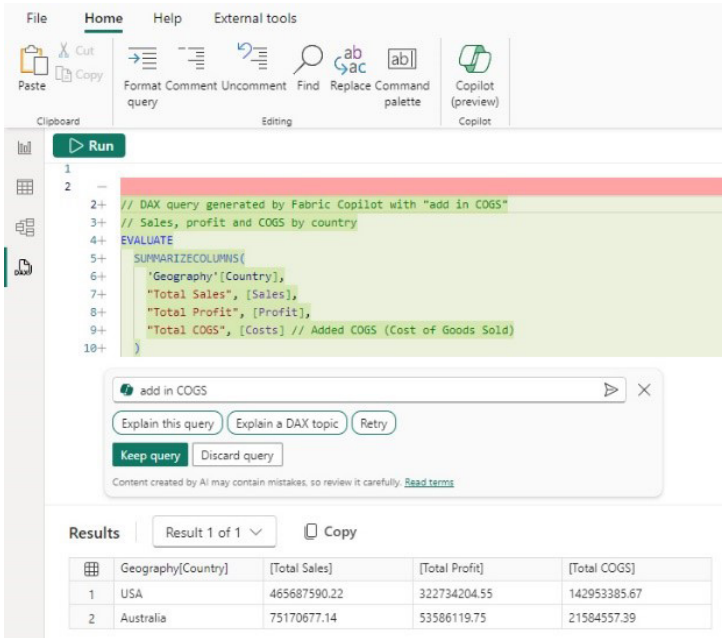
Copilot to write and explain DAX queries in DAX query view updates in Preview

DAX query view includes an inline Fabric Copilot to write and explain **DAX** queries, which remains in public Preview. This month, Power BI has received the following updates:

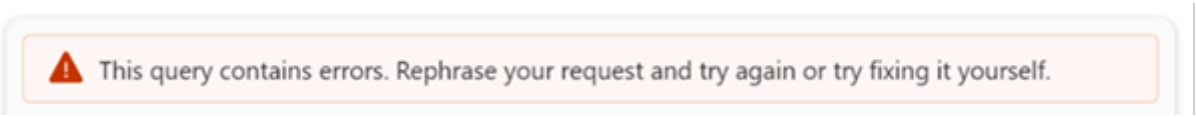
1. **Run the DAX query before you keep it.** Previously, the Run button was disabled until the generated **DAX** query was accepted or Copilot was closed. Now, you can Run the **DAX** query then decide to Keep or Discard the **DAX** query



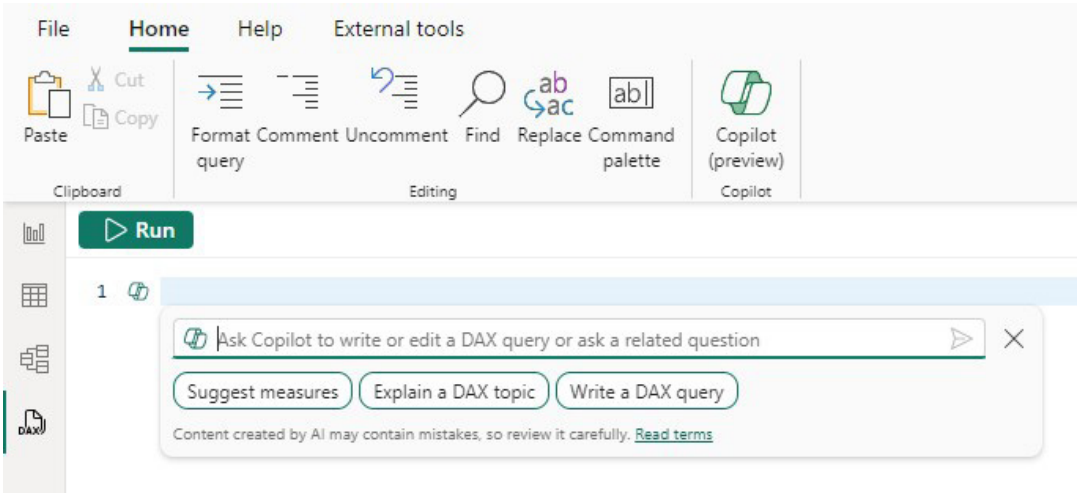
2. **Conversationally build the DAX query.** Previously, the **DAX** query generated was not considered if you typed additional prompts. You had to keep the **DAX** query, select it again, then use Copilot again to adjust. Now, you may simply adjust by typing in additional user prompts



3. **Syntax checks on the generated DAX query.** Previously, there was no syntax check before the generated **DAX** query was returned. Now, the syntax is checked and the prompt is automatically retried once. If the retry is also invalid, the generated **DAX** query is returned with a note that there is an issue, giving you the option to rephrase your request or fix the generated **DAX** query

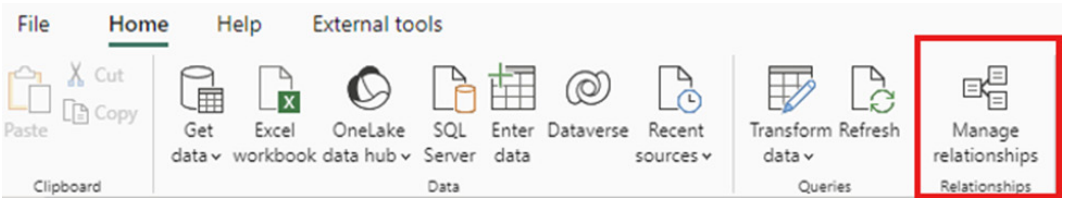


4. **Inspire buttons to get you started with Copilot.** Previously, nothing happened until a prompt was entered. Now, click any of these buttons to quickly see what you can do with Copilot.

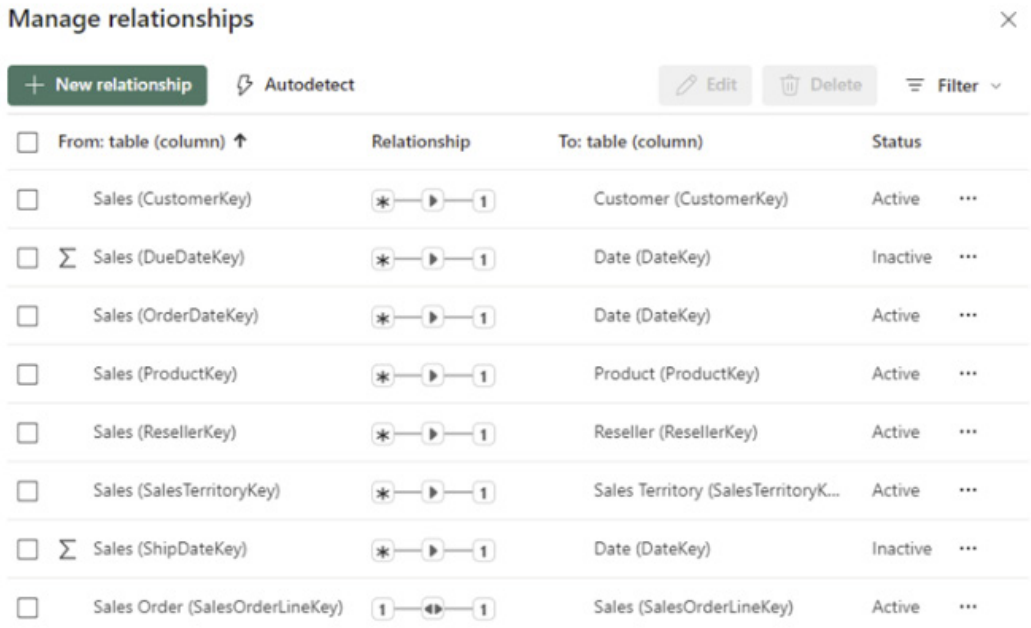


New 'Manage relationships' dialog

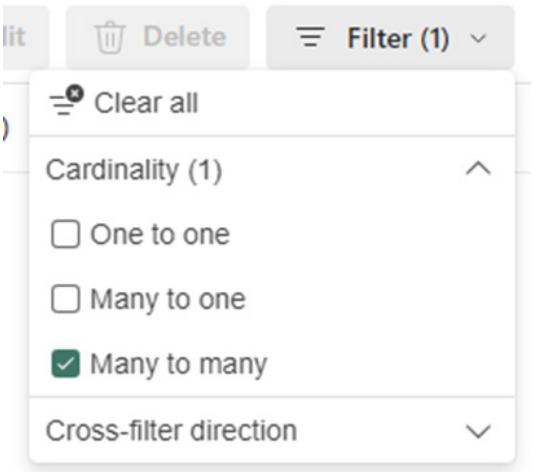
This month's updates witness the redesigned 'Manage relationships' dialog in Power BI Desktop. To open this dialog, simply select the 'Manage relationships' button in the Home tab on the Ribbon.



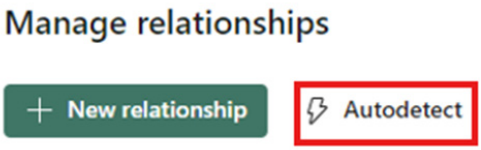
Once opened, you'll find a comprehensive view of all your relationships, along with their key properties, all in one convenient location. From here you can create new relationships or edit an existing one, *viz.*



Additionally, you have the option to filter and focus on specific relationships in your model based upon cardinality and cross filter direction.



To make things even easier, Power BI can find and create relationships for you by selecting the 'Autodetect' button.



Refreshing calculated columns and calculated tables referencing DirectQuery sources with Single Sign-On

Ever since composite models were released on Power BI semantic models and Analysis Services, end users have been asking Microsoft to support the refresh of calculated columns and tables in Power BI Service. This month, refresh of calculated columns and tables has been enabled in Power BI Service for any DirectQuery source that uses Single Sign-On authentication. This includes the sources you use when working with composite models on Power BI semantic models and Analysis Services.

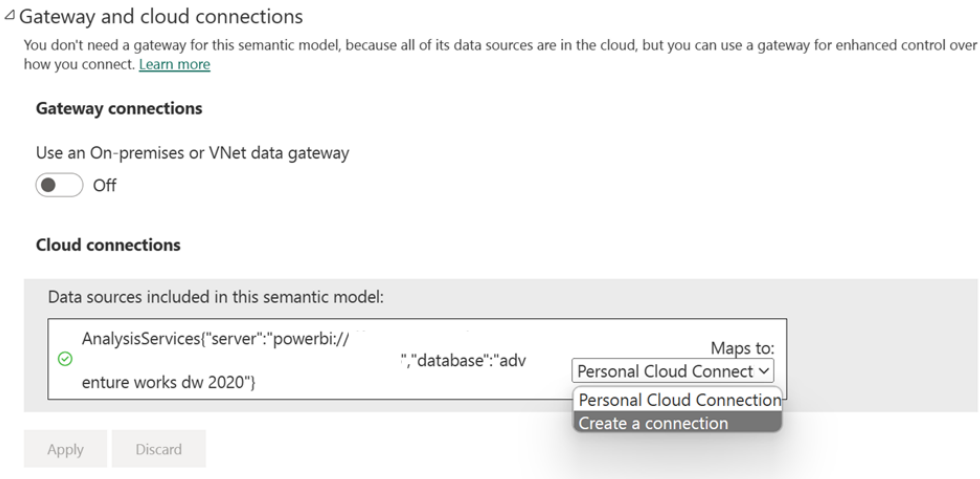
Previously, the refresh of a semantic model that uses a DirectQuery source with single-sign-on authentication failed with one of the following error messages: "Refresh is not supported for datasets with a calculated table or calculated column that depends on a table which references Analysis Services using DirectQuery" or "Refresh over a dataset with a calculated table or a calculated column which references a Direct Query data source is not supported".

Commencing with this update, you can now successfully refresh the calculated table and calculated columns in a semantic model in the Service using specific credentials as long as:

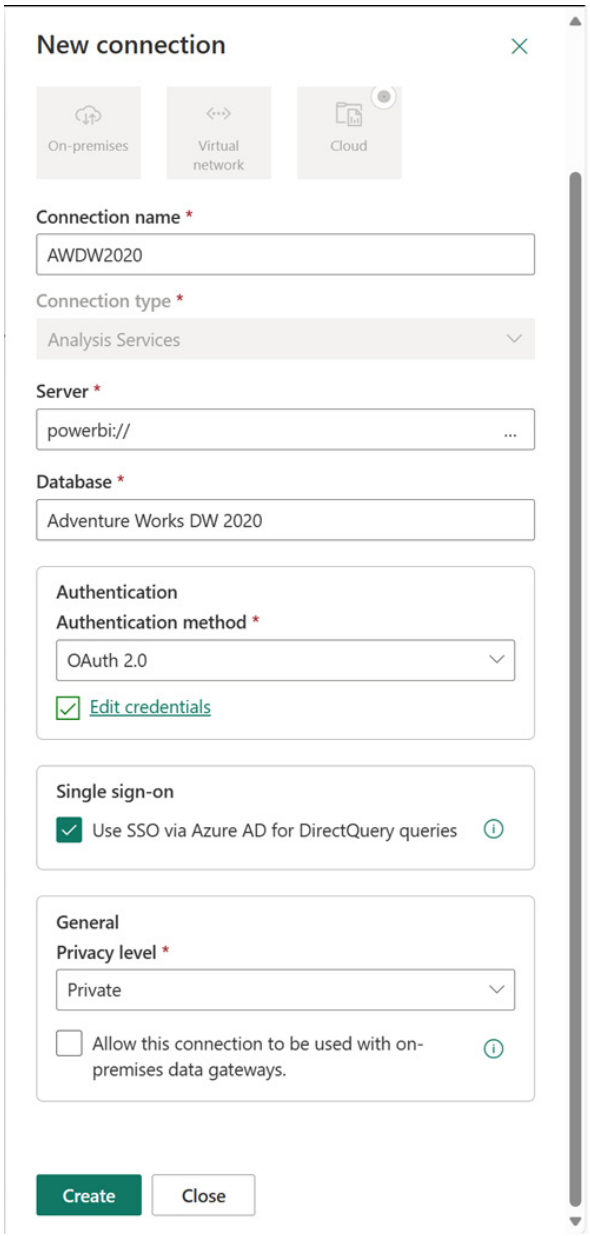
- you used a shareable cloud connection and assigned it *and / or*
- you enabled granular access control for all data connection types.

It works as follows:

- Create and publish your semantic model that uses a Single Sign-On DirectQuery source. This can be a composite model but doesn't have to be
- In the semantic model settings, under Gateway and cloud connections, map each Single Sign-On DirectQuery connection to a specific connection. If you don't have a specific connection yet, select 'Create a connection' to create it:



- If you are creating a new connection, fill out the connection details and click Create, making sure to select 'Use SSO via Azure AD' for DirectQuery queries:



- Select the connection for each Single Sign-On DirectQuery source and select Apply:

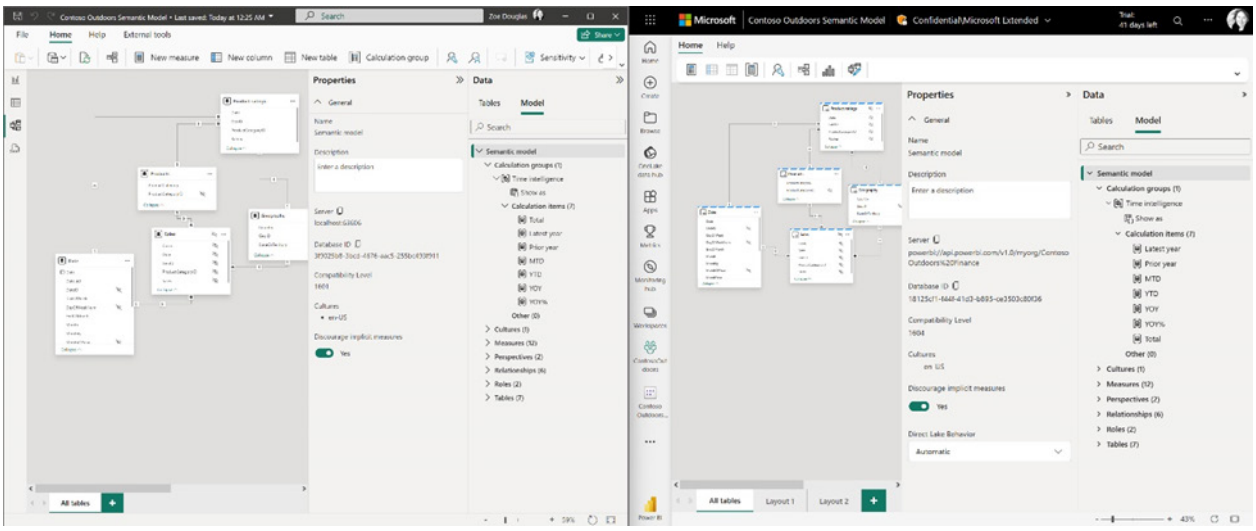
- Either refresh the semantic model manually or plan a scheduled refresh to confirm the refresh now works successfully.

Announcing General Availability of Model Explorer and authoring calculation groups in Power BI Desktop

This month also sees the announcement of the General Availability of Model Explorer in the Model view of Power BI, including the authoring of calculation groups. Semantic modelling is even easier with an at-a-glance tree view with item counts, search and in context paths to edit the semantic model items with Model Explorer. Top level semantic model properties are also available as well as the option to quickly create relationships in the Properties pane. Additionally, the styling for the Data pane is updated to 'Fluent UI', also used in Office and Teams.

A popular community request from the Ideas forum, authoring calculation groups is also included in Model Explorer. Calculation groups significantly reduce the number of redundant measures by allowing you to define **DAX** formulae as calculation items that can be applied to existing measures. For example, define a year over year, prior month, conversion or whatever your report needs in a **DAX** formula once as a calculation item and reuse it with existing measures. This can reduce the number of measures you need to create and make the maintenance of the business logic simpler.

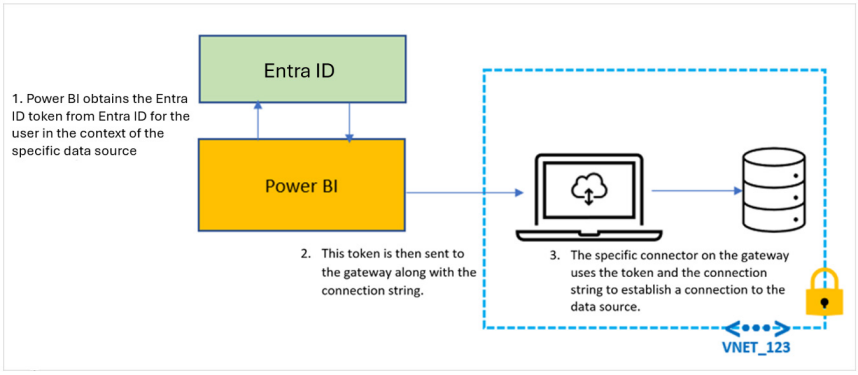
This is available in both Power BI Desktop and when editing a semantic model in the workspace.



Microsoft Entra ID SSO support for Oracle database

The Oracle database connector has been enhanced this month with the addition of Single Sign-On support in the Power BI Service with Microsoft Entra ID authentication.

Microsoft Entra ID SSO enables Single Sign-On to access data sources that rely on Microsoft Entra ID based authentication. When you configure Microsoft Entra SSO for an applicable data source, queries run under the Microsoft Entra identity of the user that interacts with the Power BI report.



Certified connector updates

There are new and updated connectors in this release:

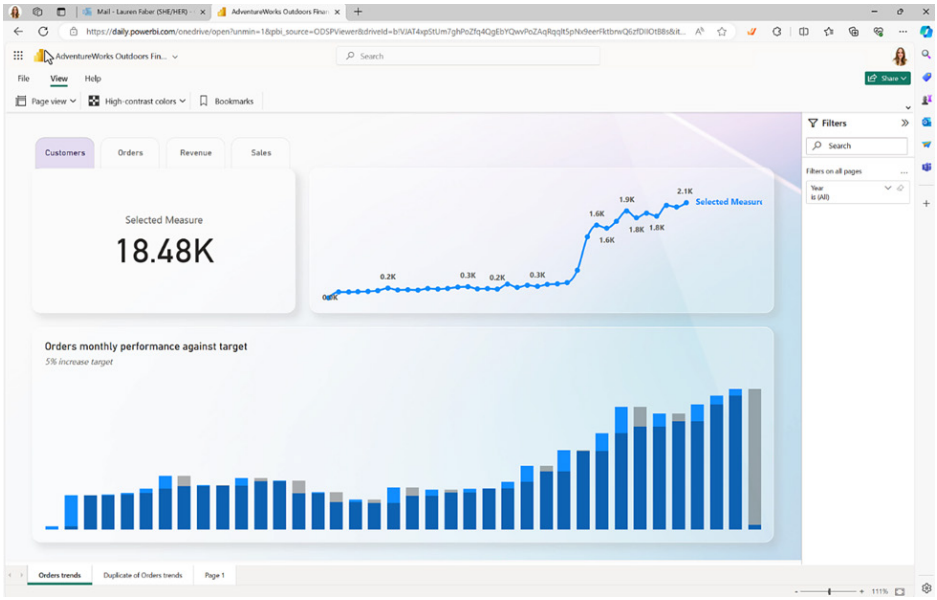
- OneStream (new): the OneStream Power BI Connector enables you to connect Power BI to your OneStream applications by simply logging in with your OneStream credentials. The connector uses your OneStream security, allowing you to access only the data you have based upon your permissions within the OneStream application. Use the connector to pull cube and relational data along with metadata members, including all their properties. Visit OneStream Power BI Connector to learn more. Find this connector in the other category
- Zendesk Data (new): a new connector developed by the Zendesk team that aims to go beyond the functionality of the existing Zendesk legacy connector created by Microsoft. Learn more about what this new connector brings
- CCH Tagetik (new)
- Azure Databricks (update).

View reports in OneDrive and SharePoint with live connected semantic models

Last May, Microsoft announced the integration between Power BI and OneDrive and SharePoint. Previously, this capability was limited to only reports with data in import mode. The team is excited to announce that you can now view Power BI reports with live connected data directly in OneDrive and SharePoint.

When working on Power BI Desktop with a report live connected to a semantic model in the Service, you can easily share a link to collaborate with others on your team and allow them to quickly view the report in their browser. Power BI has made it easier than ever to access the latest data updates without ever leaving your familiar OneDrive and SharePoint environments. This integration streamlines your workflows and allows you to access reports within the platforms you already use. With collaboration at the heart of this improvement, teams can work together more effectively to make informed decisions by leveraging live connected semantic models without being limited to data only in import mode.

Utilising OneDrive and SharePoint allows you to take advantage of built-in version control, always have your files available in the cloud, and implement familiar and simplistic sharing.



Storytelling in PowerPoint: Image mode in the Power BI add-in for PowerPoint

This month sees changes to the 'Public snapshot' mode. To address some of the feedback received from end users. Microsoft has made a couple of changes in this area:

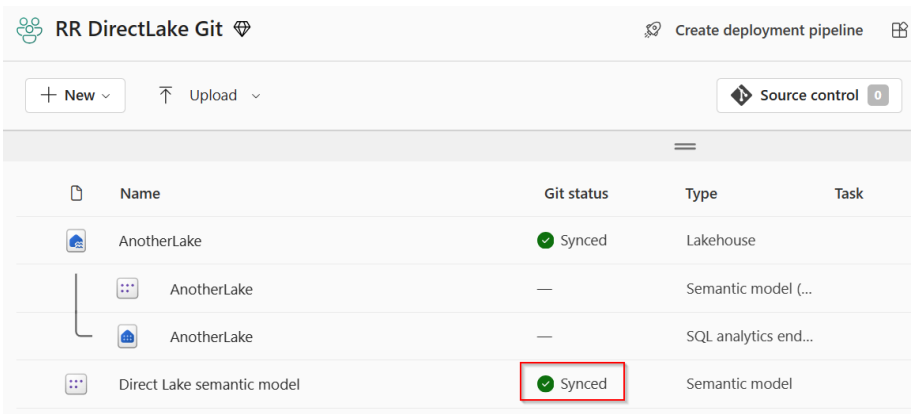
- add-ins that were saved as 'Public snapshot' may be printed and will not require that you go over all the slides and load the add-ins for permission check before the public image is made visible
- you may use the 'Show as saved image' on add-ins that were saved as 'Public snapshot'. This will replace the entire add-in with an image representation of it, so that the load time might be faster when you are presenting.

Storytelling in PowerPoint: data updated notification

Many of us keep presentations open for a long time, which might cause the data in the presentation to become outdated. To make sure you have in your slides the data you need, Power BI has added a new notification that tells you if more up to date data exists in Power BI and offers you the option to refresh and get the latest data from Power BI.

Git Integration support for Direct Lake semantic models

Direct Lake semantic models are now supported in Fabric Git Integration, enabling streamlined version control, enhanced collaboration among developers and the establishment of CI / CD pipelines for your semantic models using Direct Lake.



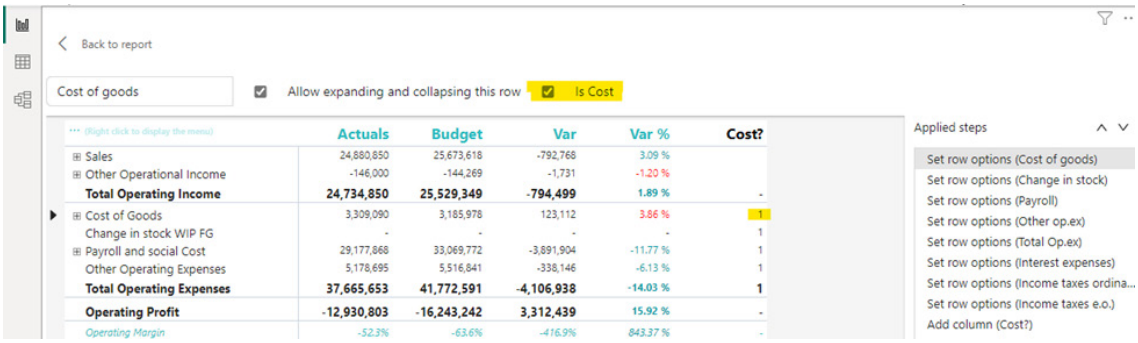
New visuals in AppSource

This month's new visuals include the following:

- Chart Flare
- Counter Calendar
- Donut Chart image
- Gas Detection Chart
- Income Statement Table
- Mapa Polski – Województwa, Powiaty, Gminy
- PictoBar
- PlanIn BI – Data Refresh Service
- ProgBar
- Seasonality Chart
- Workstream.

Financial Reporting Matrix by Profitbase

Making financial statements with a proper layout has just become a little easier with the latest version of the Financial Reporting Matrix. Users are now able to specify which rows should be classified as cost rows, which will make it easier to get the conditional formatting of variances correctly:



	Actuals	Budget	Var	Var %	Cost?
Sales	24,880,850	25,673,618	-792,768	3.09 %	
Other Operational Income	-146,000	-144,269	-1,731	-1.20 %	
Total Operating Income	**24,734,850**	**25,529,349**	**-794,499**	**1.89 %**	-
Cost of Goods	3,309,090	3,185,978	123,112	3.86 %	1
Change in stock WIP FG	-	-	-	-	1
Payroll and social Cost	29,177,868	33,069,772	-3,891,904	-11.77 %	1
Other Operating Expenses	5,178,695	5,516,841	-338,146	-6.13 %	1
Total Operating Expenses	**37,665,653**	**41,772,591**	**-4,106,938**	**-14.03 %**	**1**
Operating Profit	**-12,930,803**	**-16,243,242**	**3,312,439**	**15.92 %**	-
Operating Margin	*-52.3%*	*-63.6%*	*-416.9%*	*843.37 %*	-

Selecting a row, and ticking 'is cost' will tag the row as cost. This can be used in conditional formatting to make sure that positive variances on expenses are bad for the result, whilst a positive variance on an income row is good for the result.

The new version also includes more flexibility in measuring placement and column subtotals. Measures may be placed either:

- Default (below column headers)
- Above column headers
- In rows

Default measure placement

	Company 1			Company 2			Company 3		
	Actuals	Budget	Variance	Actuals	Budget	Variance	Actuals	Budget	Variance
Sales	13,355,932	14,040,248	-684,316	-	58,094	-58,094	18,390,342	18,701,672	-311,330
Other Operational Income	600,914	741,859	-140,945	-	-	-	201,520	209,085	-7,565
Cost of Goods	3,408,758	3,667,743	-258,985	780	722	58	1,582,139	1,566,869	15,270
Payroll and social Cost	-1,979,217	-2,219,364	240,147	5,364,677	4,847,746	516,931	13,479,555	16,746,544	-3,266,989
Other Operating Expenses	1,999,558	2,045,475	-45,917	33	41	-8	6,164,159	6,374,563	-210,404

Measure placement above column headers

	Actuals			Budget			Variance		
	Company 1	Company 2	Company 3	Company 1	Company 2	Company 3	Company 1	Company 2	Company 3
Sales	13,355,932	-	18,390,342	14,040,248	58,094	18,701,672	-684,316	-58,094	-311,330
Other Operational Income	600,914	-	201,520	741,859	-	209,085	-140,945	-	-7,565
Cost of Goods	3,408,758	780	1,582,139	3,667,743	722	1,566,869	-258,985	58	15,270
Payroll and social Cost	-1,979,217	5,364,677	13,479,555	-2,219,364	4,847,746	16,746,544	240,147	516,931	-3,266,989
Other Operating Expenses	1,999,558	33	6,164,159	2,045,475	41	6,374,563	-45,917	-8	-210,404

Measure placement in rows

	Company 1	Company 2	Company 3
Sales			
Actuals	13,355,932	-	18,390,342
Budget	14,040,248	58,094	18,701,672
Variance	-684,316	-58,094	-311,330
Other Operational Income			
Actuals	600,914	-	201,520
Budget	741,859	-	209,085
Variance	-140,945	-	-7,565

- Conditionally hide columns.

Highlighted new features include:

- Measure placement – in rows
- Select Column Subtotals
- New 'Format Pane' design
- Row options.

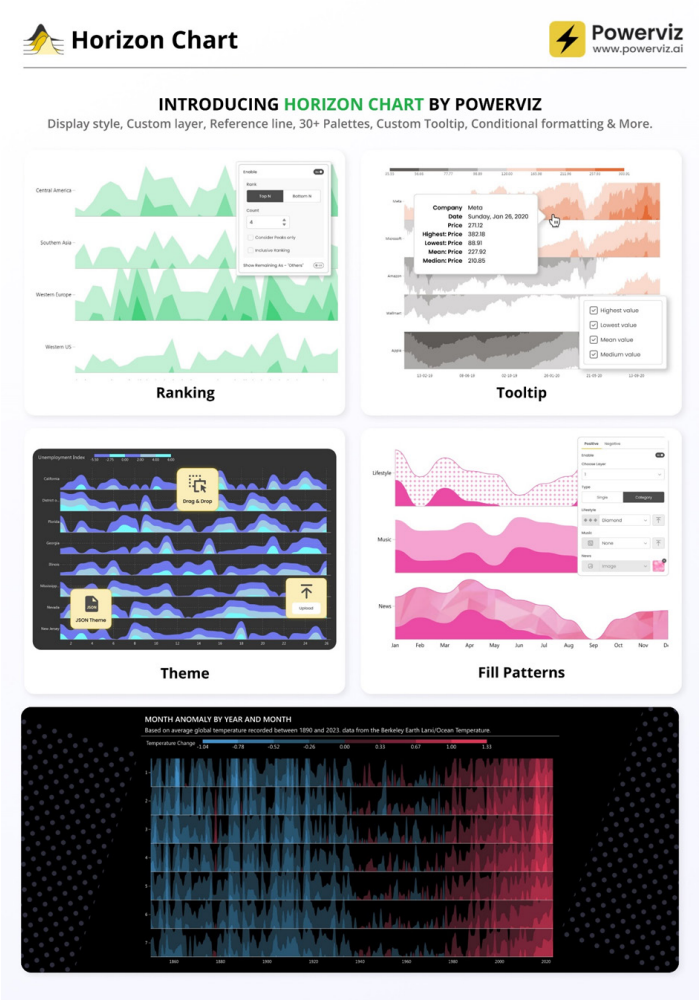
Horizon chart by Powerviz

An Horizon chart is an advanced visual for time-series data, revealing trends and anomalies. It displays stacked data layers, allowing users to compare multiple categories while maintaining data clarity. Horizon Charts are particularly useful to monitor and analyse complex data over time, making this a valuable visual for data analysis and decision-making.

Key features include:

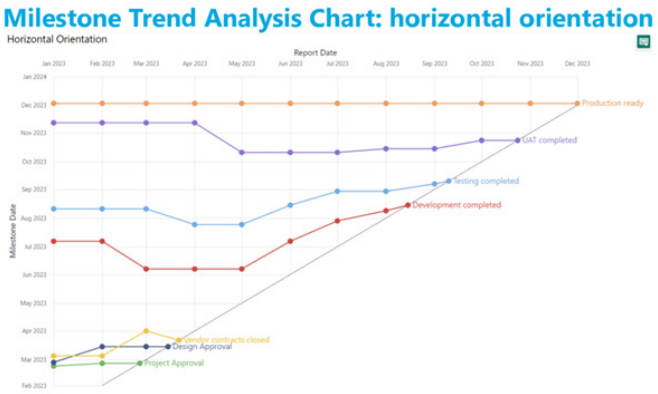
- **Horizon styles:** you may choose Natural, Linear or Step with adjustable scaling
- **Layer:** layer data by range or custom criteria. Display positive and negative values together or separately on top
- **Reference Line:** highlight patterns with **x**-axis lines and labels.
- **Colours:** apply 30+ colour palettes and use FX rules for dynamic colouring
- **Ranking:** filter Top / Bottom **N** values, with "Others"
- **Gridlines:** add gridlines to the **x** and **y** axis.
- **Custom ToolTip:** add highest, lowest, mean and median points without additional **DAX**
- **Themes:** save designs and share seamlessly with **JSON** files
- **Other features** included are ranking, annotation, grid view, show condition and accessibility support.

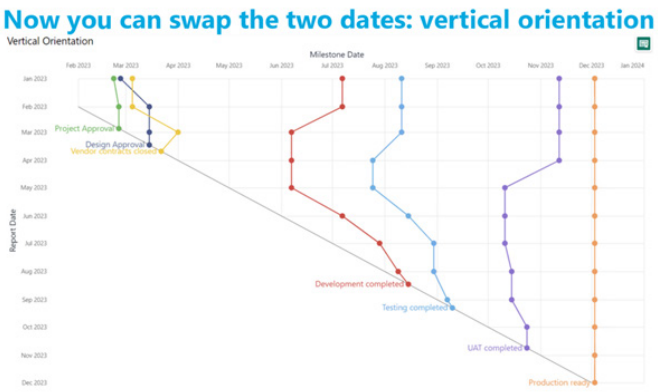
Business use cases include Time-Series Data Comparison, Environmental Monitoring and Anomaly Detection.



Milestone Trend Analysis chart by Nova Silva

You may now switch between horizontal and vertical orientations, catering to your preferred visualisation style. This means that the Milestone Trend Analysis (MTA) Chart remains simple and useful for swiftly identifying deadline trends, empowering you to take timely corrective actions. With this update, the chart aims to enhance deadline awareness among project participants and stakeholders alike.





In this latest version, it's easy to navigate between horizontal and vertical views within the familiar Power BI interface. It also benefits from supported features like themes, interactive selection and ToolTips.

Sunburst chart by Powerviz

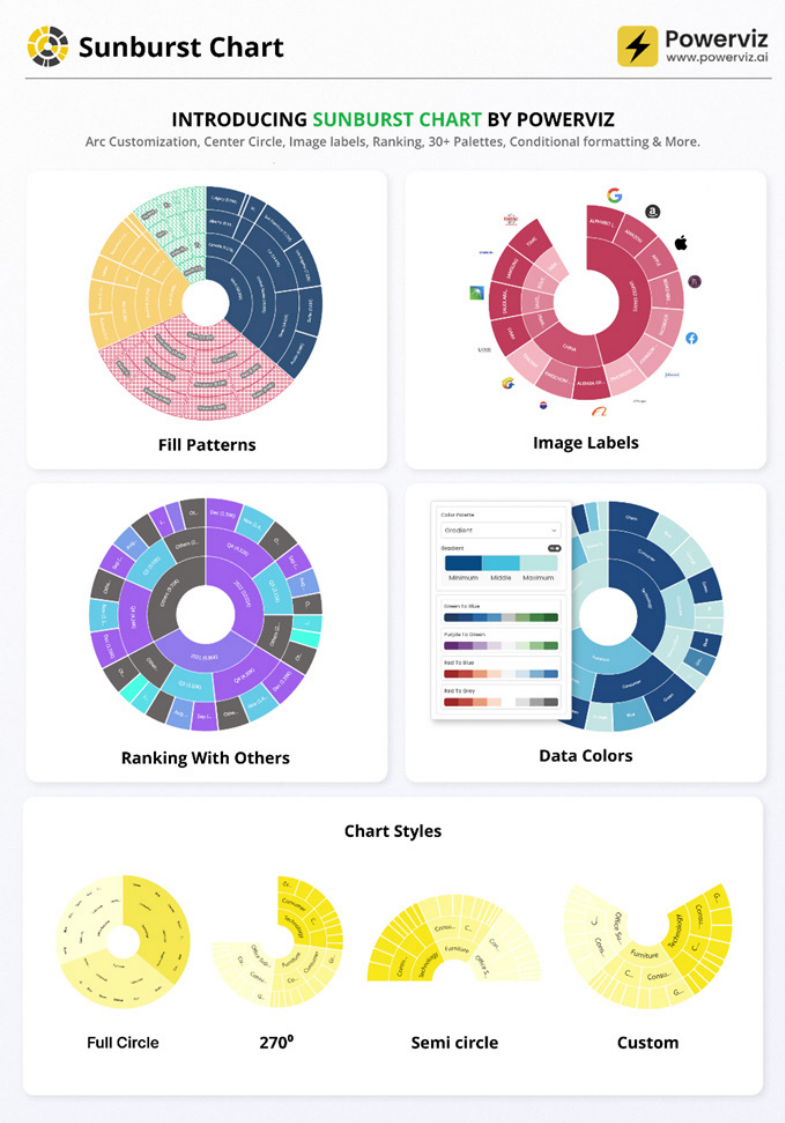
Powerviz's Sunburst chart is an interactive tool for hierarchical data visualisation. With this chart, you can easily visualise multiple columns in a hierarchy and uncover valuable insights. The concentric circle design helps in displaying part-to-whole relationships.

Key features include:

- **Arc customisation:** customise shapes and patterns
- **Colour scheme:** accessible palettes with 30+ options
- **Centre circle:** design an inner circle with layers. You can add text, measure, icons and images
- **Ranking:** filter Top / Bottom **N** values, with "Others"
- **Conditional formatting:** identify outliers based upon measure or category rules
- **Labels:** smart data labels for readability
- **Image labels:** add an image as an outer label
- **Interactivity:** zoom, drill down, cross-filtering and ToolTip features
- **Other features** included are annotation, grid view, show condition and accessibility support.

Business use cases include:

- **Sales and Marketing:** market share analysis and customer segmentation
- **Finance:** department budgets and expenditures distribution
- **Operations:** supply chain management
- **Education:** course structure, curriculum creation
- **Human Resources:** organisation structure, employee demographics.



Stacked Bar chart with Line by JTA

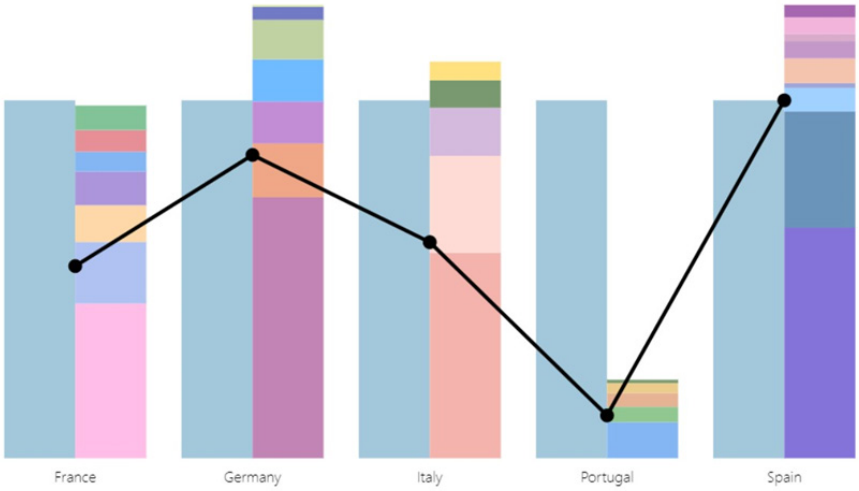
The Stacked Bar chart with Line by JTA seamlessly merges the simplicity of a traditional Bar chart with the versatility of a Stacked Bar, changing the way you showcase multiple datasets in a single, cohesive display.

Unlocking a new dimension of insight, this visual features a dynamic line that provides a snapshot of data trends at a glance. You may navigate through your data with multiple configurations, gaining a swift and comprehensive understanding of your information.

You can tailor your visual experience with an array of functionalities and customisation options, enabling you to effortlessly compare a primary metric with the performance of an entire set. The flexibility to customise the visual according to your unique preferences empowers you to harness the full potential of your data.

Features of Stacked Bar chart with Line include:

- Stack the second bar
- Format the axis and gridlines
- Add a legend
- Format the colours and text
- Add a Line chart
- Format the line
- Add marks to the line
- Format the labels for bars and line.



Drill Down Combo PRO: now with Legend field

A new feature has been added to these Combo PRO, Combo Bar PRO and Timeline PRO visuals – the Legend field support. The Legend field makes it easy to visually split series values into smaller segments without the need to use measures or create separate series. You simply add a column with category names that are adjacent to the series values, and the visual will do the following:

- display separate segments as a stack or cluster, showing how each segment contributed to the total Series value
- create legend items for each segment to quickly show / hide them without filtering
- apply custom fill colours to each segment
- show each segment value in the ToolTip.

You can choose between three chart types: Columns, Lines or Areas, and feature up to 25 different series in the same visual and configure each series independently.

More next month.

New Features for Excel

Generating multiple formula columns, creating complex formula columns that span across multiple tables and new ways to engage with Copilot in Excel are now all available to Excel users on web and Windows (subject to roll out). Copying and pasting improvements and sharing links to sheet views are also available in Excel for the web, and the 'Ink to Text Pen' is rolling out to Insiders running Excel for Windows.

In summary, here are the new features for this month:

Excel for the web

- Generating multiple formula columns with Copilot in Excel
- Creating complex formula columns that span multiple tables with Copilot in Excel
- Copying and pasting improvements
- Sharing link to sheet views

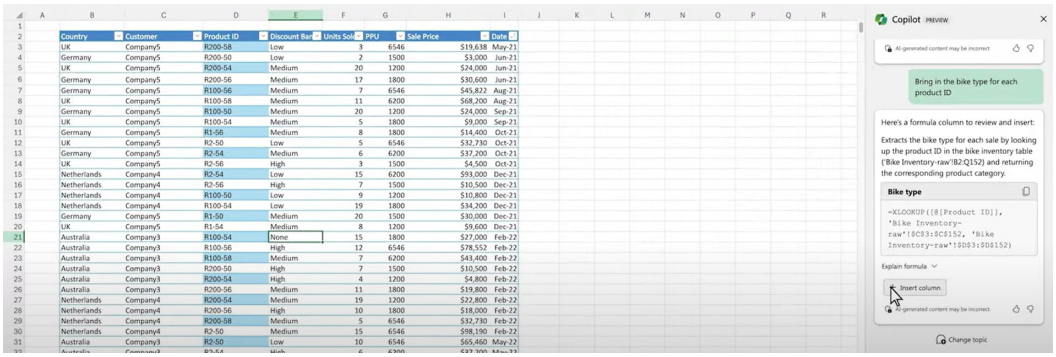
Excel for Windows

- Generating multiple formula columns with Copilot in Excel
- Creating complex formula columns that span multiple tables with Copilot in Excel
- New ways to engage with Copilot in Excel
- Ink to text pen (Insiders).

Let's get started.

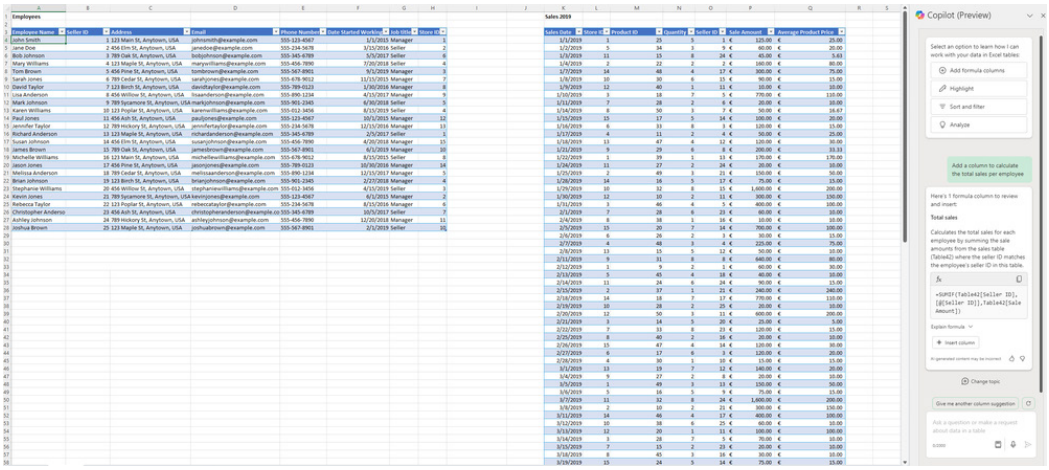
Generating multiple formula columns with Copilot in Excel

For both Excel for the web and in Windows, Copilot now supports generating multiple formula columns from a single prompt. You may now ask one question and Copilot can return two formula columns simultaneously. For instance, you can extract both the first name and last name from a single prompt, neatly separating the information into distinct columns.



Creating complex formula columns that span multiple tables with Copilot in Excel

Also in Excel for the web and Windows, Copilot can now create complex formula columns that span across multiple tables, using functions like **XLOOKUP** and **SUMIF**. This capability streamlines data processing and empowers you to handle more complex data analysis tasks efficiently, even when using multiple tables.

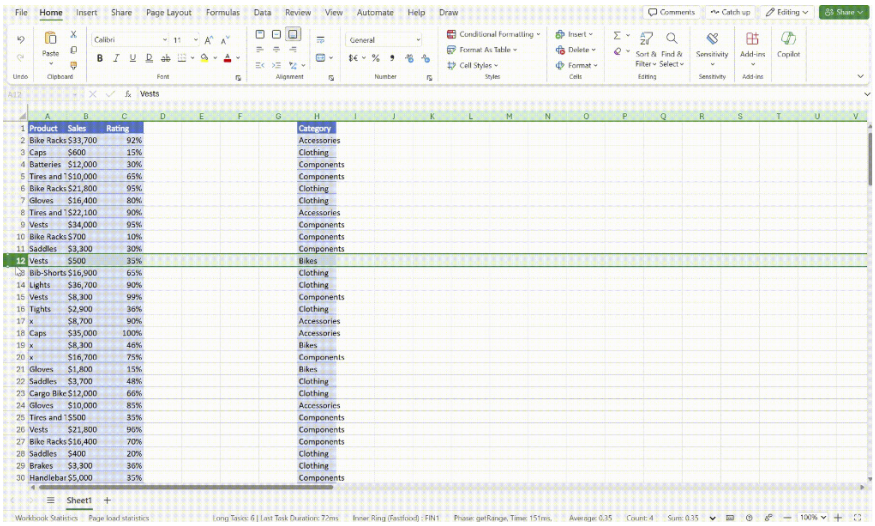


Copying and pasting improvements

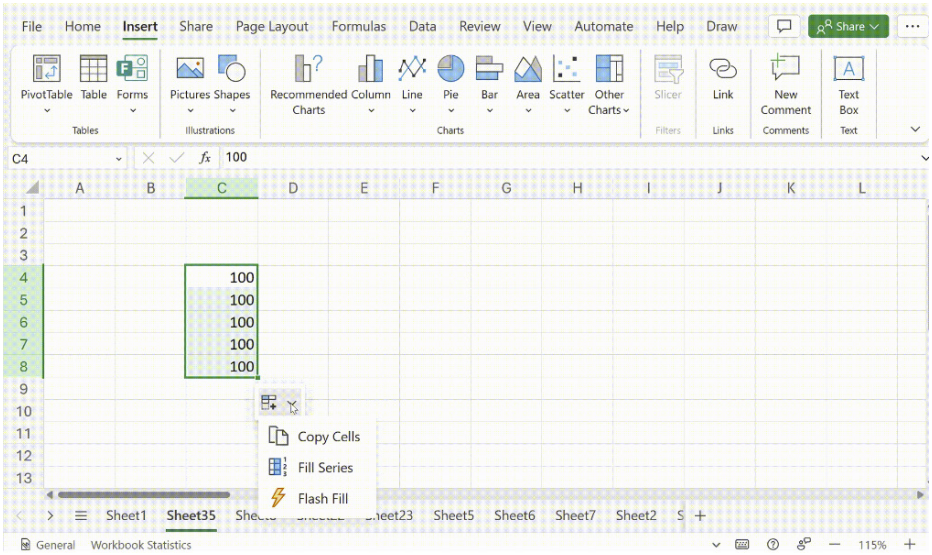
The following improvements have been made to Excel for the web to the copying and pasting experience:

- **Drag and drop** enhancements provide a more seamless and intuitive way to manipulate data
- **Autofill** enables you to drag data to fill series, copy cells, and split data
- **Paste options** improvements allow you to choose the right paste option for your needs by right-clicking or accessing the auto-recovery paste options.

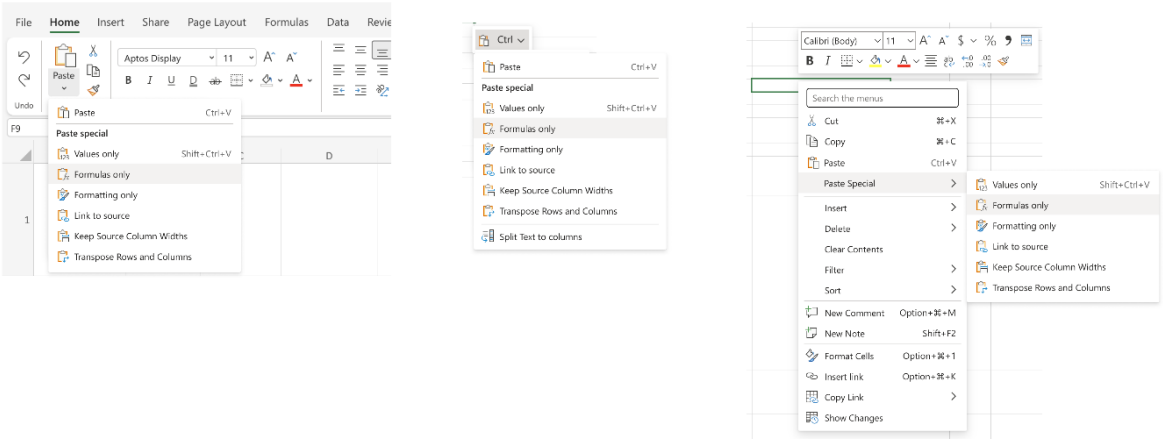
To try the drag and drop feature, click any row or column, hold and drag when the cursor shows the hand icon, and then drop in any other row or column.



Regarding the autofill feature, add data in a cell, select the cell and then drag using the plus sign in the bottom right corner of the selected cell.



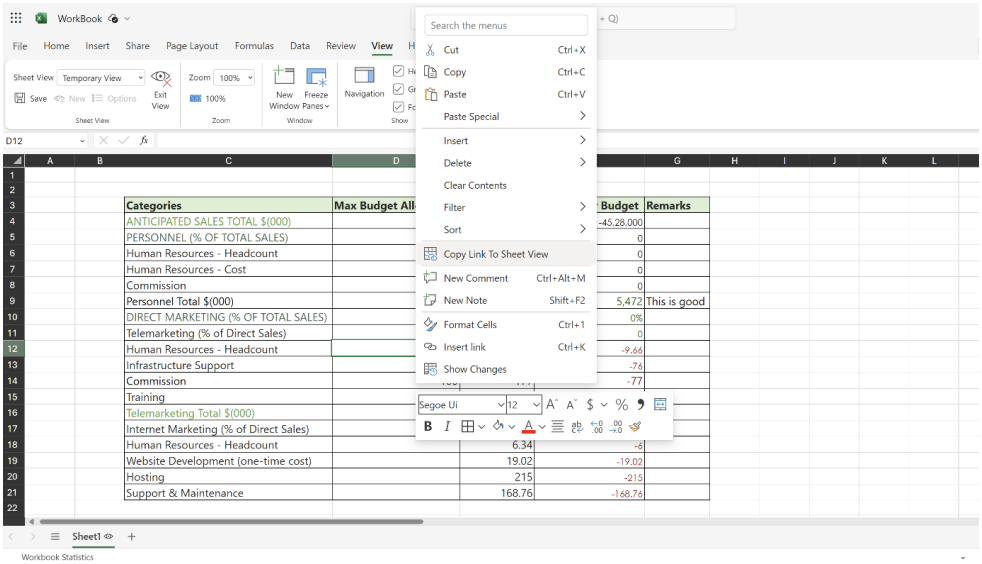
With reference to the paste option improvements, copy some data and either explore the paste options available in the Ribbon, or right-click in the location where you want to paste the copied information and select the paste option you want.



Sharing link to sheet views

You may now share a link to a Sheet view in Excel for the web that captures a subset of a workbook's contents. By doing so, you can more efficiently collaborate on large or complex workbooks and easily gather feedback and input from others.

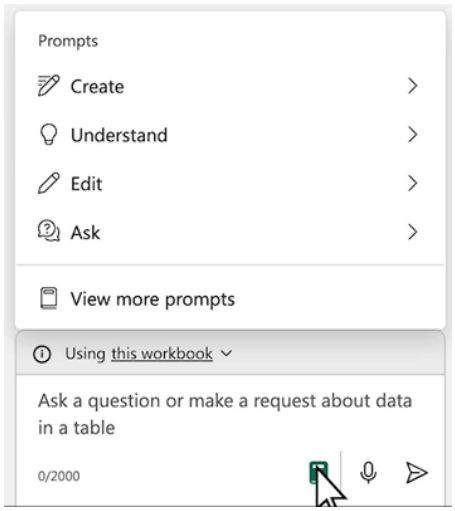
Click the View tab, and then either click the 'Sheet View' arrow and select an existing view in the list or create a new sheet view. In the Sheet view, right-click any cell or the sheet tab that contains the content you want to share, and then select 'Copy Link To Sheet View'.



In the Link to Sheet message that appears, click the Close button. You can share the copied link with others in the way you want (*e.g.* in an email message, in a Teams chat), or keep and use it as reference link.

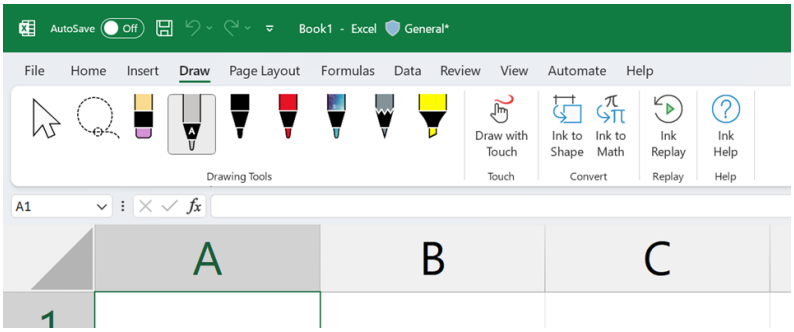
New ways to engage with Copilot in Excel

Microsoft has enhanced Copilot in Excel for Windows to make the experience better. Now, you can use the Microphone feature to prompt Copilot verbally, eliminating the need for manual typing. In addition, for fresh prompt ideas tailored to your specific data, now you can select the View Prompts icon to explore the Prompt Guide in the chat pane. This feature is currently rolling out to Windows users.



Ink to text pen (Insiders)

Also in Excel for Windows, the 'Ink to Text Pen' lets you use your digital pen (or stylus) to automatically convert your handwriting into text as you go, helping you quickly enter content into cells. Additionally, you can use pen gestures to select and delete cell content. It should be noted that the 'Ink to Text Pen' replaces the functionality of the 'Action Pen' in Excel.



Until next month.

The A to Z of Excel Functions: NETWORKDAYS

The **NETWORKDAYS** function returns the number of whole working days between a start date and an end date. This excludes weekends by default and may also except a custom list of dates to account for regional holidays. This can be used to calculate employee benefits that accrue based upon the number of days worked during a specific term.

The **NETWORKDAYS** function employs the following syntax to operate:

NETWORKDAYS(start_date, end_date, holidays)

The **NETWORKDAYS** function has the following arguments:

- **start_date:** this is required and is a date representing the start of the period
- **end_date:** this is also required. This is a date representing the end of the period
- **holidays:** this is optional. This represents an optional range of one or more dates to be excluded from the working calendar. This list can be either a range of cells containing the dates or an array constant of serial numbers representing the dates.

It should be further noted that:

- dates should be entered by using the **DATE** function, or as results of other formulas or functions. For example, use **DATE**(2012,5,23) for the 23rd day of May, 2012. Problems can occur if dates are entered as text
- Excel stores dates as sequential serial numbers so they can be used in calculations. By default, January 1, 1900 is serial number 1, and January 1, 2012 is serial number 40909 because it is 40,908 days after January 1, 1900.
- if any argument is not a valid date, **NETWORKDAYS** returns the *#VALUE!* error value.

Please see our example below:

	A	B	C	D	E	F
1	Description		Value	Holidays		
2	Start date		01-Jul-22	14-Jul-22		
3	End date		31-Jul-22	15-Jul-22		
4				16-Jul-22		
5				18-Jul-22		
6				19-Jul-22		
7						
8						
9	Description				Result	Formula
10	Number of working days in July 2022, not considering any holidays				21	=NETWORKDAYS(C2,C3)
11	Number of working days in July 2022, excluding days identified as holidays above				17	=NETWORKDAYS(C2,C3,D2:D6)

As you can see above, the function may be used to calculate the number of non-weekend days in a period, potentially excluding a custom list of holidays too.

The A to Z of Excel Functions: NETWORKDAYS.INTL

This is a more "global" function than the above **NETWORKDAYS** function. Workdays typically include Monday to Friday, excluding any holidays. However, this isn't the case everywhere in the world.

The **NETWORKDAYS.INTL** function returns the number of whole working days between a start date and an end date, using parameters to indicate which and how many days are considered weekends / not workdays. The specified weekends are excluded as well as a custom list of dates to account for regional holidays. This can be used to calculate employee benefits that accrue based upon the number of days worked during a specific term.

The **NETWORKDAYS.INTL** function employs the following syntax to operate:

NETWORKDAYS.INTL(start_date, end_date, [weekend], [holidays])

The **NETWORKDAYS.INTL** function has the following arguments:

- **start_date:** this is required and is a date representing the start of the period
- **end_date:** this is also required. This is a date representing the end of the period
- **weekend:** this is optional. This is an integer that indicates the days of the week that should be considered as weekends. They can be any value between 1 and 17 and indicate the following weekend days:

Weekend no.	Weekend days
1 or omitted	Saturday, Sunday
2	Sunday, Monday
3	Monday, Tuesday
4	Tuesday, Wednesday
5	Wednesday, Thursday
6	Thursday, Friday
7	Friday, Saturday
11	Sunday only
12	Monday only
13	Tuesday only
14	Wednesday only
15	Thursday only
16	Friday only
17	Saturday only

- **holidays:** this is also optional. This is an optional range of one or more dates to be excluded from the working calendar. This list can be either a range of cells containing the dates or an array constant of serial numbers representing the dates.

It should be further noted that:

- if the **start_date** is later than the **end_date**, the return value will be negative, and the magnitude will be the number of whole workdays
- if the **start_date** is out of range for the current date base value, **NETWORKDAYS.INTL** returns the *#NUM!* error value
- if the **end_date** is out of range for the current date base value, **NETWORKDAYS.INTL** returns the *#NUM!* error value
- if a **weekend** string is of invalid length or contains invalid characters, **NETWORKDAYS.INTL** returns the *#VALUE!* error value
- dates should be entered by using the **DATE** function, or as results of other formulas or functions. For example, use **DATE(2012,5,23)** for the 23rd day of May, 2012. Problems can occur if dates are entered as text
- Excel stores dates as sequential serial numbers so they can be used in calculations. By default, January 1, 1900 is serial number 1, and January 1, 2012 is serial number 40909 because it is 40,908 days after January 1, 1900.

Please see our example below:

	A	B	C	D	E	F
1	Description		Value	Holidays		
2	Start date		01-Jul-22	14-Jul-22		
3	End date		31-Jul-22	15-Jul-22		
4	Weekends		6	16-Jul-22		
5				18-Jul-22		
6				19-Jul-22		
7						
8						
9	Description				Result	Formula
10	Number of working days in July 2022, not considering any holidays				21	=NETWORKDAYS.INTL(C2,C3)
11	Number of working days in July 2022, excluding days identified as holidays above				17	=NETWORKDAYS.INTL(C2,C3,,D2:D6)
12	Number of working days in July 2022, excluding days identified as holidays above, considering Thursdays and Fridays as weekends.				19	=NETWORKDAYS.INTL(C2,C3,C4,D2:D6)
13						

As you can see above, the function can be used to calculate the number of working days in a period with varying weekend consideration, potentially also excluding a custom list of holidays.

More Excel Functions next month.

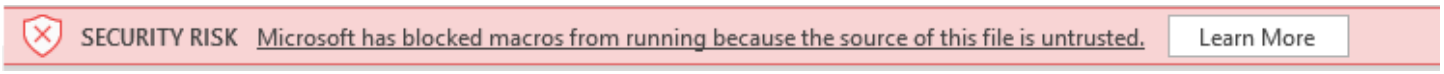
Beat the Boredom Suggested Solution

The challenge this month was to unblock multiple macro-enabled Excel files from an external source.

The Challenge

Imagine you had received several Excel files containing macros from your friend(s), your colleague(s) or from the internet. You needed all these Excel files to have macros enabled and standing in your way was the security. Unblocking all the security from the 'Properties' window of all the files you have is a time-consuming task. Hence, we devised a challenge for you to find a simple and efficient way to unblock all the macro-enabled files.

This month's challenge was to get rid of this message for all of your Excel files that contains macro *(below)*.



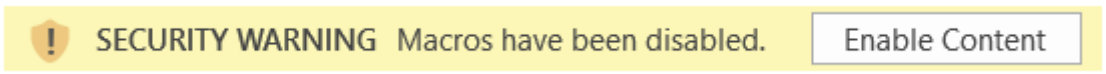
As always, there were some requirements:

- the solution should be simple
- no coding was allowed.

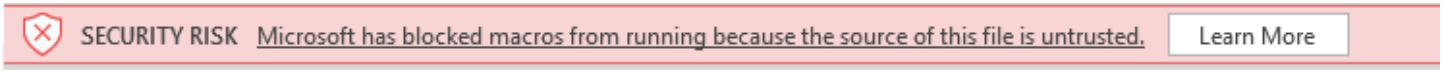
Suggested Solution

Before we begin, let's discuss the relevant Microsoft security features here.

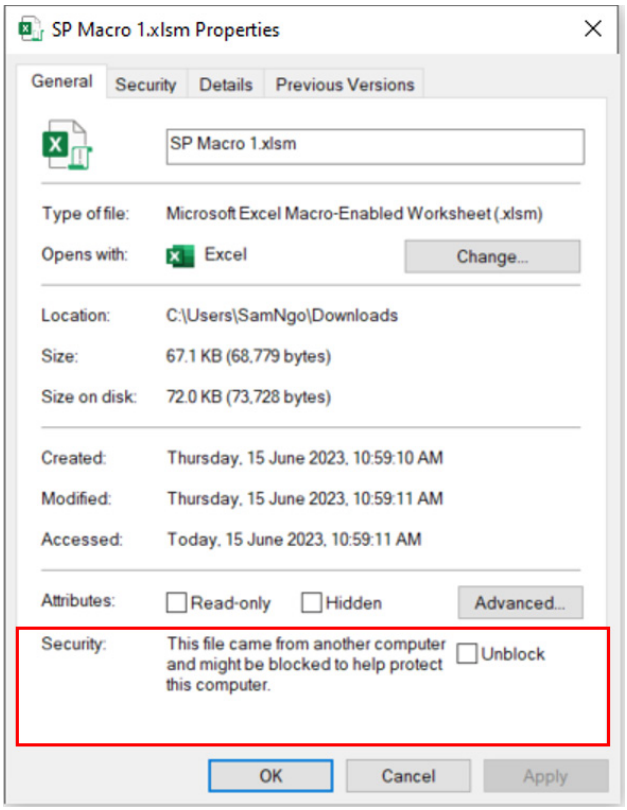
Microsoft's software puts up a message called 'SECURITY WARNING' if we are opening files from the internet. The reason for this 'SECURITY WARNING' is due to the fact that when you open the Excel file that have macros most people ignore this security warning and press the 'Enable Content' button right away (*below*):



This can potentially infect the PC with virus, especially if the Excel macro-enabled files come from an untrusted source. Hence, Microsoft must implement different security checks for every file originating from internet or Restricted zones. Now, you will see a 'SECURITY RISK' message appear after you press 'Enable Content':



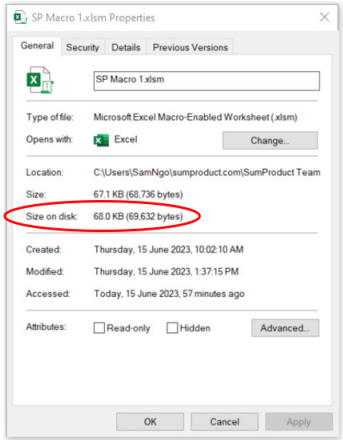
The way Microsoft detects whether a file is from the internet is by a mark. If we download an Excel file from the internet it will have the 'Mark of the Web' (MOTW). We can see the indicator of MOTW via the **Properties -> General -> Security**:

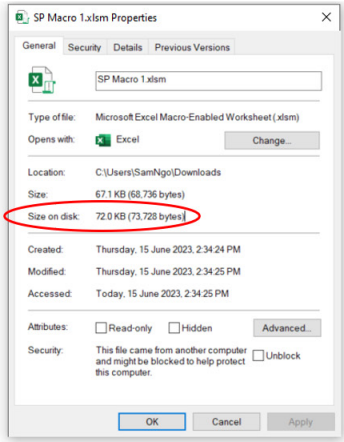


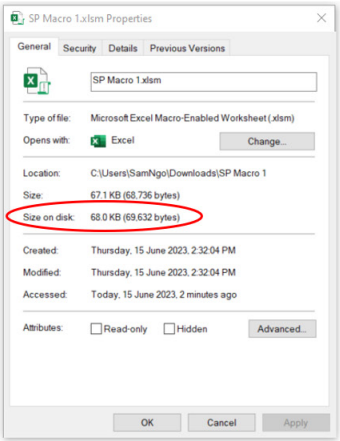
The standard way we remove this MOTW is to tick the unblock box and apply it. It will unblock macros for our workbook. However, if we have hundreds or thousands of Excel files that contains macros, it will take significant time to unblock them all.

Solution 1: Zip and Unzip

To solve this issue, we can utilise the built-in Zip program in Windows. This is a simple solution that involves zipping all the Excel files that contain macros downloaded from the web or have the MOTW (Mark of the Web) attribute. Once the files have been zipped, we can then proceed to unzip the file, which will automatically unblock all the Excel files in the zip. The pictures below will show what happen to the MOTW:





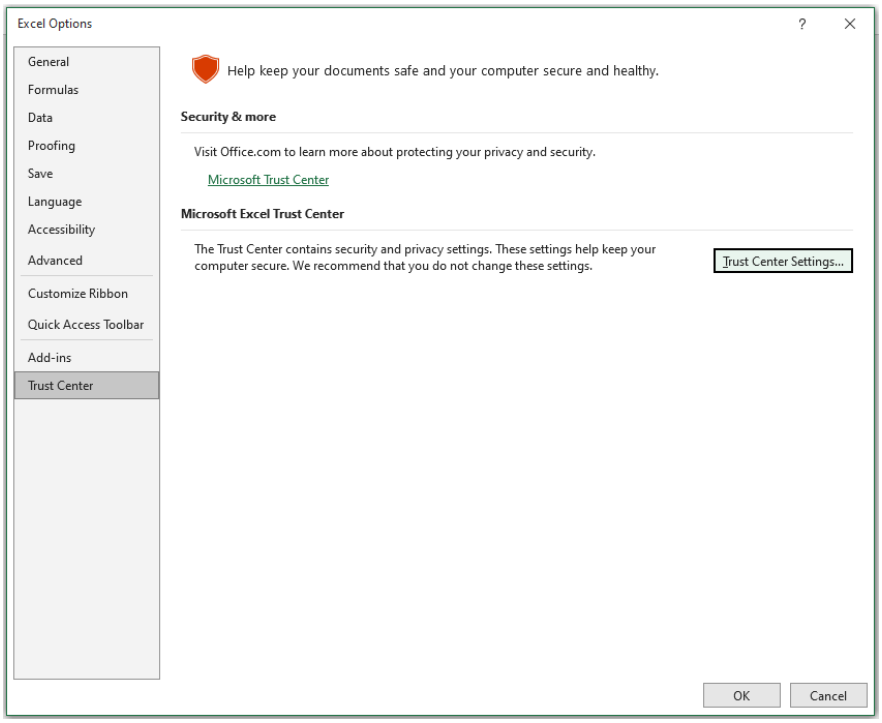


We can see here that our source file (Properties window on the left) which its size on disk is 68 KB jump up to 72 KB and have a MOTW (Properties window in the middle). After we zip and unzip the size on disk of the file (Properties window on the right) is the same as our source file. Thus, this means that files downloaded from the internet will have a MOTW added which increase the file size. By zipping and unzipping the file(s) we will remove the MOTW from our file(s).

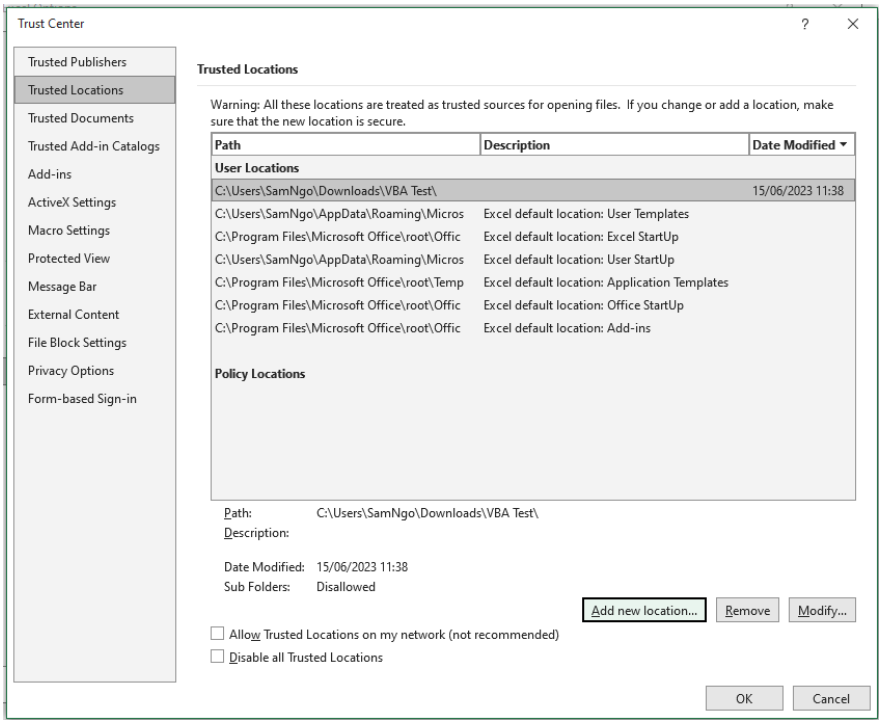
If we are using a different zip program, we might need to unblock the security for the zip file before proceeding with the unzipping process. To do this, we can right-click on the zip file, select **Properties -> General -> Security -> Unblock** then proceed with unzipping the file to achieve the same result.

Solution 2: Trusted Location

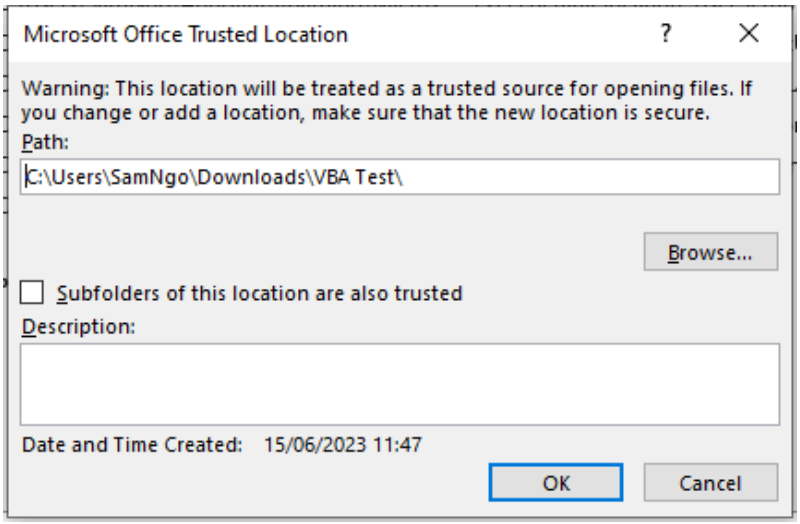
We can also set up a trusted location where we can drop all excel files that contains macro. We can do this by going to **File -> Options -> Trust Center -> Microsoft Excel Trust Center -> Trust Center Settings**.



This will pop up the Trust Center window. From here we go to **Trusted Location -> Add new location**.



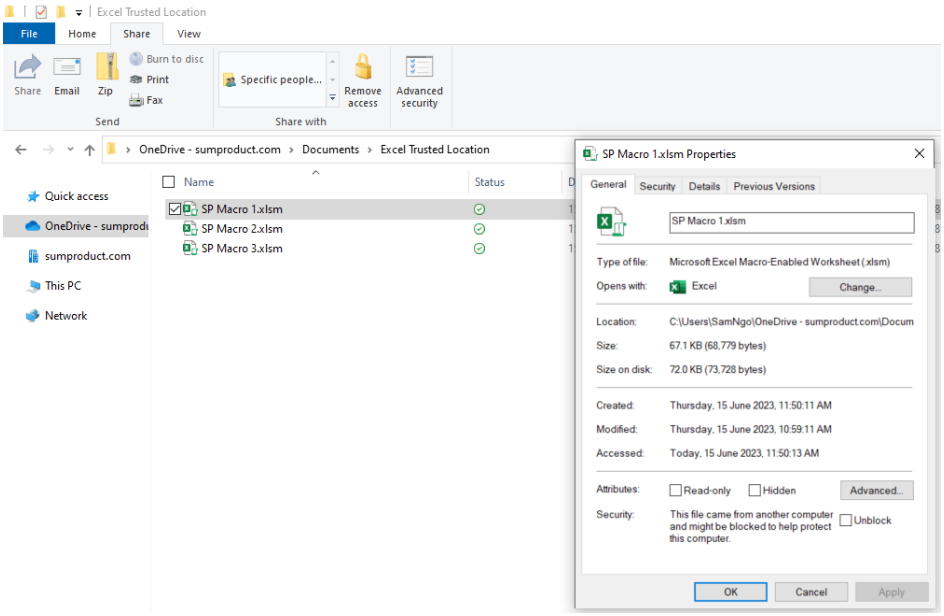
In the Microsoft Office Trusted Location, we can specify the folder path in the Path section or we can browse the folder we want with the Browse button.



In this example, our trusted location is:

C:\Users\SamNgo\Downloads\VBA Test

Although we just move my file into the trusted location, the MOTW is still there:



Well, don't worry about that! If we open our Excel files from here, there is no longer any security risk message, and we may use our macro-enabled workbook(s) freely.

Solution 3: SharePoint

SharePoint is a web based platform for collaboration that seamlessly integrates with Microsoft 365. Similar to designating a Trusted Location, uploading an Excel workbook that contains macros to SharePoint will unblock those with security issues, allowing macros to run freely. This removes a major point of friction for Excel macro workflows by making the macros available right away to all team members with access to that SharePoint document library.

The MOTW warning will remain in the workbook properties to remind users that macros should only be enabled for files stored within the organisation's secure SharePoint site. Macros should function properly once there in SharePoint as it "transforms" into a digital command centre for your most important Excel macro projects.

Word to the Wise

There are some other ways we can do to unblock the macro files like writing an VBA code on your local PC or using PowerShell which can quickly turn off the MOTW attributes. But hey, three solutions is surely enough, yes..?

More next month.

Upcoming SumProduct Training Courses

Location	Course	Course Date	Local TIme	UTC	Duration
Melbourne Australia	Power Pivot, Power Query and Power BI	17 June 2024 - 18 June 2024	09:00 - 17:00 AEDT	16 June 2024 22:00 UTC- 18 June 2024 06:00 UTC	2 Days
Melbourne Australia	Excel Tips and Tricks	19 June 2024	09:00 - 17:00 AEDT	18 June 2024 22:00 UTC- 19 June 2024 06:00 UTC	1 Day
Melbourne Australia	Financial Modelling	20 June 2024 - 21 June 2024	09:00 - 17:00 AEDT	19 June 2024 22:00 UTC- 21 June 2024 06:00 UTC	2 Days
Virtual (Australia)	ChatGPT Part 1	24 June 2024	13:30 - 17:00 AEDT	24 June 2024 02:30 UTC- 24 June 2024 06:00 UTC	1 Day
Virtual (Australia)	ChatGPT Part 2	1 July 2024	13:30 - 17:00 AEDT	1 July 2024 02:30 UTC- 1 July 2024 06:00 UTC	1 Day
Sydney Australia	Power Pivot, Power Query and Power BI	15 July 2024 - 16 July 2024	09:00 - 17:00 AEDT	14 July 2024 22:00 UTC- 16 July 2024 06:00 UTC	2 Days
Sydney Australia	Excel Tips and Tricks	17 July 2024	09:00 - 17:00 AEDT	16 July 2024 22:00 UTC- 17 July 2024 06:00 UTC	1 Day
Sydney Australia	Financial Modelling	18 July 2024 - 19 July 2024	09:00 - 17:00 AEDT	17 July 2024 22:00 UTC- 19 July 2024 06:00 UTC	2 Days

Key Strokes

Each newsletter, we'd like to introduce you to useful keystrokes you may or may not be aware of. We take a look at the function keys this month:

Keystroke	What it does
F1	Help
F2	Toggle Select, Edit and Enter / Point modes
F3	Paste Names
F4	Redo / Edit (**F2**) Mode: Toggle **$** references
F5	Go To / Refresh File List
F6	Next Pane
F7	Check Spelling
F8	Extend Selection Mode
F9	Calculate Now
F10	Activate Menus
F11	Insert Chart on New Sheet
F12	Save As

There are *c.*550 keyboard shortcuts in Excel. For a comprehensive list, please download our Excel file at http://www.sumproduct.com/thought/keyboard-shortcuts. Also, check out our new daily **Excel Tip of the Day** feature on the www.sumproduct.com homepage.

Our Services

We have undertaken a vast array of assignments over the years, including:

- **Business planning**
- **Building three-way integrated financial statement projections**
- **Independent expert reviews**
- **Key driver analysis**
- **Model reviews / audits for internal and external purposes**
- **M&A work**
- **Model scoping**
- **Power BI, Power Query & Power Pivot**
- **Project finance**
- **Real options analysis**
- **Refinancing / restructuring**
- **Strategic modelling**
- **Valuations**
- **Working capital management**

If you require modelling assistance of any kind, please do not hesitate to contact us at contact@sumproduct.com.

Link to Others

These newsletters are not intended to be closely guarded secrets. Please feel free to forward this newsletter to anyone you think might be interested in converting to "the SumProduct way".

If you have received a forwarded newsletter and would like to receive future editions automatically, please subscribe by completing our newsletter registration process found at the foot of any www.sumproduct.com web page.

Any Questions?

If you have any tips, comments or queries for future newsletters, we'd be delighted to hear from you. Please drop us a line at newsletter@sumproduct.com.

Training

SumProduct offers a wide range of training courses, aimed at finance professionals and budding Excel experts. Courses include Excel Tricks & Tips, Financial Modelling 101, Introduction to Forecasting and M&A Modelling.

Check out our more popular courses in our training brochure:

Drop us a line at training@sumproduct.com for a copy of the brochure or download it directly from www.sumproduct.com/training.

Sydney Address: SumProduct Pty Ltd, Suite 803, Level 8, 276 Pitt Street, Sydney NSW 2000
New York Address: SumProduct Pty Ltd, 48 Wall Street, New York, NY, USA 10005
London Address: SumProduct Pty Ltd, Office 7, 3537 Ludgate Hill, London, EC4M 7JN, UK
Melbourne Address: SumProduct Pty Ltd, Ground Floor, 470 St Kilda Road, Melbourne, VIC 3004
Registered Address: SumProduct Pty Ltd, Level 14, 440 Collins Street, Melbourne, VIC 3000

contact@sumproduct.com
www.sumproduct.com
+61 3 9020 2071