

# SumProduct

NEWSLETTER #143 - October 2024

[www.sumproduct.com](http://www.sumproduct.com) | [www.sumproduct.com/thought](http://www.sumproduct.com/thought)



## We embrace innovation this month

SumProduct is proud to announce it is one of six Victorian state finalists for the Telstra Best of Business Awards 2025. We told you we keep looking to the future! We are proud of the recognition: yes, we'd like to win, but even if we don't, it's a fantastic achievement getting down to the last six.

October sees **Excel Virtually Global** hit your screens once more. Make sure you attend virtually. The event is for charity and yet again, we will see many experts from all over the world providing tips, tricks and ideas on how to make Excel and Power BI go just that little bit further. Check out the details below.

There is more though: we have another new function for you to peruse this month, **TRIMRANGE**, and a new feature, 'Focus Cell' to, er, focus on. We let 'Over to AI' loose on the divisive question **XLOOKUP** vs. **VLOOKUP** too. Do you agree? Does it make sense? Is it made up? Have we run out of ideas? Find out inside!

And then there is everything else. 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, Excel Updates, plus the A to Z of Excel functions and Keyboard Shortcuts continue to inform.

And STOP PRESS!! **GROUPBY** and **PIVOTBY** are now Generally Available!

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

*Liam Bastick*, Managing Director, SumProduct



## Telstra Best of Business Awards: SumProduct is a State Finalist!

Victorian  
Embracing Innovation  
Finalist



#TelstraBestofBusinessAwards

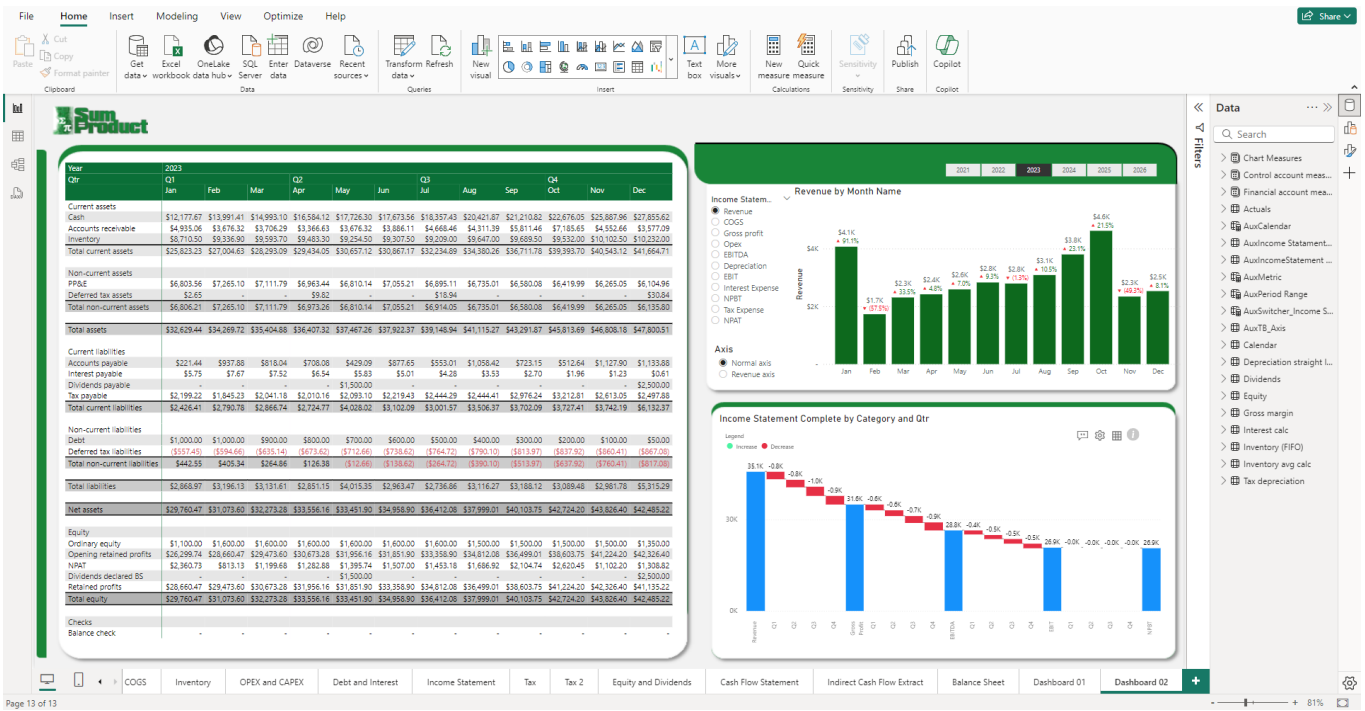
The Telstra Best of Business Awards were created to celebrate Australia's best small and medium-sized businesses. There are various categories, including "Embracing Innovation", which "...recognises businesses innovating with technology to develop solutions for challenges faced by modern Australia".

SumProduct has just been announced as a State Finalist for Victoria.

That's fantastic news.

As Managing Director, I am so proud of this achievement for the team. We champion every day new, innovative solutions not just for Australia, but the world (big ambitions here, lol). We were one of the first companies (not just here in Oz) to use Power Pivot, Power Query, Power BI, Solver and Power Automate.

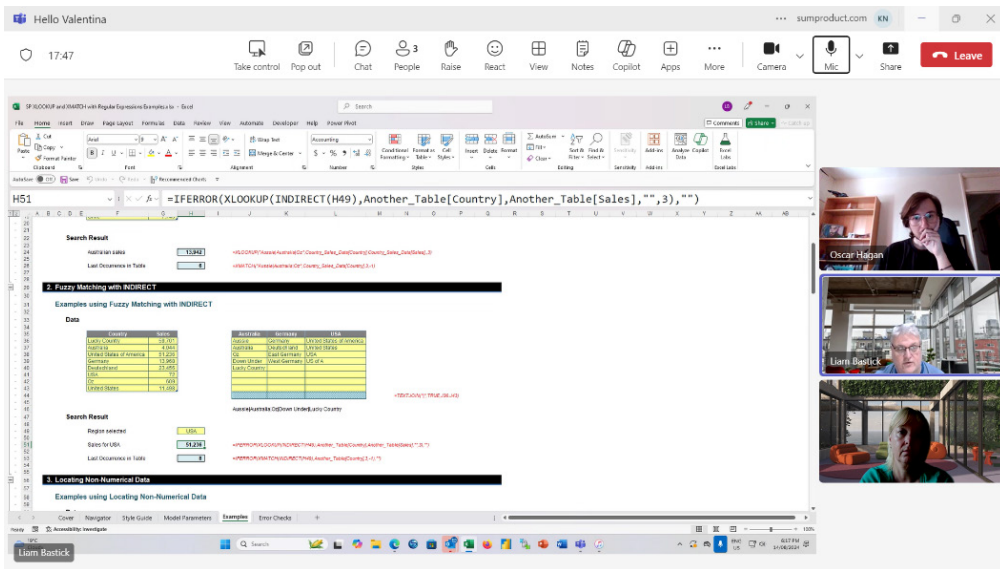
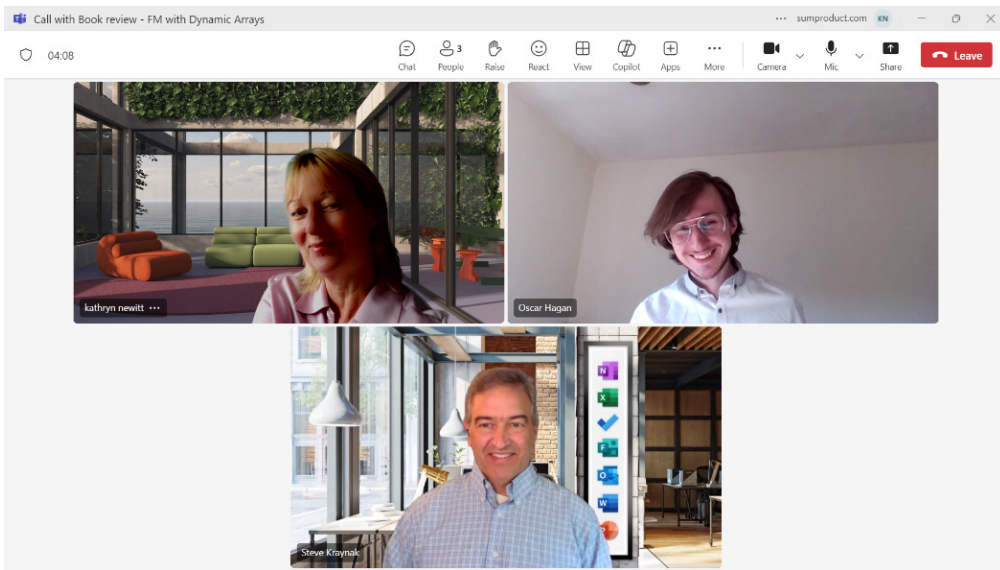




We have launched Financial Modelling in Power BI, Financial Modelling using Dynamic Arrays and are currently initiating successful strategies for Financial Modelling, Strategy and Decision Analysis with AI. We have saved our clients thousands of needless repetitive drudgery work and showed them how to analyse / make decisions more powerfully, intelligently and timely. And we'll keep seeking the next Big Thing too.



Congratulations to the current proactive, thought-provoking (challenging?) colleagues including Bernadette, Henrique, Kathryn, Kris, Myles, Oscar, Sam, Steve, Talia, Tim and Valentina, plus all our casual team members, alumni and circle of clients read friends / extended family. Apologies to those I have forgotten!



Whether we achieve the award or not, the team has been recognised for their diligent, innovative efforts: that's fantastic! (I'd like the team to win it though!!)

We strive to continue to change the world ONE analyst / finance person / accountant / individual at the time.

Looks like we're gaining momentum!

Check out SumProduct here if you are unfamiliar with us: [www.sumproduct.com](http://www.sumproduct.com). But if you are, how have you received this newsletter! 😊



# EXCEL VIRTUALLY GLOBAL 2024

Not long to go now... Just a reminder, this 40+ hour virtual conference, for charity, presents many Excel, Data Platform and PowerPoint MVPs, together with other acknowledged experts from Microsoft and / or around the globe to present, answer questions and demonstrate the future of Excel and how it will make your life easier, personally and professionally.

Topics will include all things Excel, financial modelling, dynamic arrays, Python in Excel, Fabric, Copilot, ChatGPT, Power Pivot, Power Query and Power BI. It's not all in English either!

Each session (including Q&A) will last no more than an hour and topics will cover all expertise levels, from novice to expert. Most presenters are well known in their spheres, and have written blogs, books and articles and / or present video sessions.

Some sessions will be live and some will be recorded so you may watch them later with downloads aplenty – there are no medals for staying up to watch the entire event live! That's just as well, as it will run from midnight GMT / UTC on Tuesday 8 October into some time on Thursday 10 October.

From your own favourite chair, bring a laptop, an inquisitive mind and your sense of humour. Be prepared to learn heaps. And remember, it's for charity – this year, there is no ticketing process: all we ask is an "honesty box" where you donate to your favourite charity.

For more details (e.g. times, speakers, sessions), please go to [www.excelvirtuallyglobal.com](http://www.excelvirtuallyglobal.com). The links will go live nearer the time, so keep checking back as the program may change too.

Hopefully, we'll see you there!

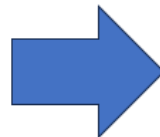
## Beat the Boredom Challenge

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...

Given a table of sales data, how do you find out the favourite store location of each customer? You might wish to filter, then count and then sort. You may also want to open Power Query to perform some advanced manipulations. This month, we challenge you to do that in one [1] formula!

Using the same data, a second part of the challenge is to find out each customer's last visited store location, again with only one [1] Excel formula. The table **Data** contains customer visits data (pictured), and the desired outputs should look like the following upon completion:

Date	Customer	Store
1 Feb 22	Talia Cao	Central
15 Feb 22	Tim Heng	Town Hall
1 Mar 22	Talia Cao	Redfern
15 Mar 22	Sam Ngo	Circular Quay
29 Mar 22	Talia Cao	Central
12 Apr 22	Talia Cao	Town Hall
26 Apr 22	Sam Ngo	Central
10 May 22	Liam Bastick	Wynyard
24 May 22	Sam Ngo	Redfern
7 Jun 22	Liam Bastick	Central
21 Jun 22	Talia Cao	Circular Quay
5 Jul 22	Tim Heng	Redfern
19 Jul 22	Liam Bastick	Wynyard
2 Aug 22	Sam Ngo	Central
5 Jan 23	Talia Cao	Central
5 Jan 23	Tim Heng	Circular Quay
5 Jan 23	Sam Ngo	Town Hall



### Most frequently visited store

Customer	Store
Talia Cao	Central
Tim Heng	Town Hall
Sam Ngo	Central
Liam Bastick	Wynyard

### Most recently visited store

Customer	Store
Talia Cao	Central
Tim Heng	Circular Quay
Sam Ngo	Town Hall
Liam Bastick	Wynyard

As always, there are some requirements:

- each formula needs to be within one cell
- this is a formula challenge; no Power Query / Get & Transform or VBA.

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

## Keeping in TRIM

Microsoft recently announced a new function, **TRIMRANGE**, and an accompanying set of new reference operators. Albeit in Preview only (and presently only to a select few which seems to be slightly less than the number of lottery winners on the moon), **TRIMRANGE** scans in from the edges of a range or array until it finds the first non-blank cell (or value). It then excludes those blank rows or columns.

It should be noted that **TRIMRANGE** and the associated new reference operators are not text functions, so they are not going to be useful for cell contents, such as line breaks or carriage returns.

The syntax is as follows:

**TRIMRANGE(range, [trim\_rows], [trim\_columns])**

It has three [3] arguments:

- **range:** this argument is required and represents the range (or array) to be trimmed
- **trim\_rows:** this argument is optional and determines which rows should be trimmed by selecting one of four [4] values:
  - **0:** none
  - **1:** trims leading blank rows
  - **2:** trims trailing blank rows
  - **3 (default if omitted):** trims both leading and trailing blank rows
- **trim\_columns:** this argument is optional and determines which columns should be trimmed by selecting one of four [4] values:
  - **0:** none
  - **1:** trims leading blank columns
  - **2:** trims trailing blank columns
  - **3 (default if omitted):** trims both leading and trailing blank columns.

In essence, the **TRIMRANGE** function removes empty rows and / or columns from the edges of a range. This can be particularly useful when writing dynamic array formulae or optimising aggregation, array or lambda functions for performance.

We think **TRIMRANGE** may have missed a trick here in its initial Preview guise as we think of the **TRIM** function which removes excess space, not

just at the beginning and end of a text string but throughout. We can't help feeling options to remove blank rows / columns throughout the range might be welcomed by many and feel there are many areas where this would be beneficial (e.g. charting, dashboards, summary outputs).

In the example below, **TRIMRANGE** has been used to calculate the length of any text entered into column **A**:

	A	B	C
1	Eggs		4
2	Milk		4
3	Butter		6
4	Bananas		7
5	Cookies		7
6			

The formula here is given by

**=LEN(A1:A5)**

However, we might wish to extend the range to add more words in column **A**, and this can lead to redundant calculations and possibly slower performance as the spreadsheet becomes more complex.

	A	B	C	D
1	Eggs		4	
2	Milk		4	
3	Butter		6	
4	Bananas		7	
5	Cookies		7	
6			0	
7			0	
8			0	
9			0	
10			0	
11			0	

This is where the formula

**=LEN(TRIMRANGE(A:A))**

can come to the rescue and remove unused calculations (but reinstate them later if more text is added in column A):

	A	B	C	D
1	Eggs		=LEN(TRIMRANGE(A:A))	
2	Milk		4	
3	Butter		6	
4	Bananas		7	
5	Cookies		7	
6			0	
7			0	
8			0	
9			0	
10			0	
11			0	

Easy!

	A	B	C
1	Eggs		4
2	Milk		4
3	Butter		6
4	Bananas		7
5	Cookies		7
6			

Without the use of **TRIMRANGE**,

**=LEN(A:A)**

would calculate for every cell in column C, returning over a million unnecessary results. Besides being inefficient, trailing undesirable zeroes are returned to the grid. This can be especially problematic if you then try and operate on the spill using **=C1#** notation.

**TRIMRANGE** is also a useful tool for optimising the performance of lambda functions that operate on ranges. No doubt Microsoft has some particular scenarios in mind here (more new dynamic array functions soon possibly?). It allows lambda authors to more tightly scope ranges, which can reduce the number of required computations.

We don't have this function ourselves yet, but anecdotal evidence suggests this function will not work with three-dimensional references, but it does seem to work with arrays as well as ranges. We shall do more testing as and when we can!

This new function has provided Microsoft with the opportunity to introduce **Trim References**, also known as **Trim Refs**. These may be used to achieve the same functionality as **TRIMRANGE** more succinctly by replacing the range's colon ":" with one of the three Trim Ref types described below:

Type	Example	Equivalent TRIMRANGE	Description
Trim All (..)	A1:..E10	TRIMRANGE(A1:E10,3,3)	Trim leading and trailing blanks
Trim Trailing (:.)	A1:..E10	TRIMRANGE(A1:E10,2,2)	Trim trailing blanks
Trim Leading (.:)	A1:..E10	TRIMRANGE(A1:E10,1,1)	Trim leading blanks

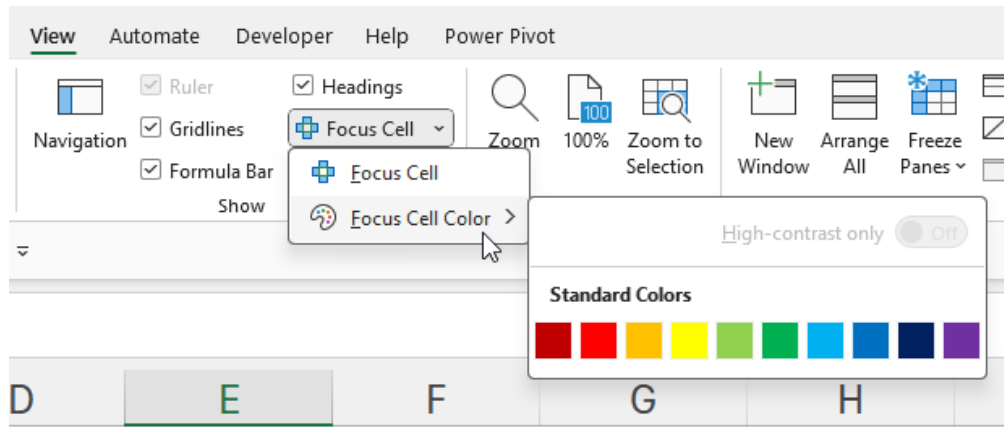
Personally, I'd like a different syntax such as <:, <: and :> to make it easier to read for accessibility purposes, but hey, beggars cannot be choosers. Remember, all of the above is in Preview: your feedback is welcomed and there is still time to make a difference before they land in production / become Generally Available.

Full-column references are often avoided because they can have poor performance with some functions. However, with Trim Refs, they are

much more performant as the full-column reference can be constrained to just the portion with values.

Before you get too excited, this new function and these new references are currently available to Beta Channel users running **Version 2409 (Build 18020.2000)** or later, and even then, it's not everyone – yet! Pray Santa thinks you've been good!

# Focus Cell

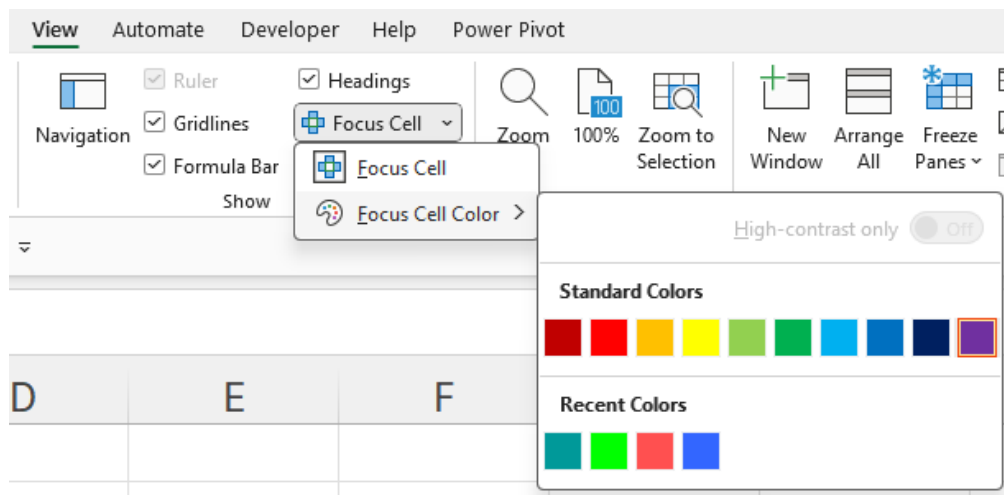


Sometimes, some new functions and features enter the Excel arena in “stealth mode”, *i.e.* with no fanfare and scant documentation. ‘Focus Cell’ is one such new feature.

Arriving with no proclamations, SumProduct discovered this feature hiding in Microsoft 365 Apps for Enterprise edition Version 2410 (Build 18112.20000) Beta Channel. We are unsure whether it is proportionally flighted or not (*i.e.* it is only available to a percentage of all users with this version of Excel).

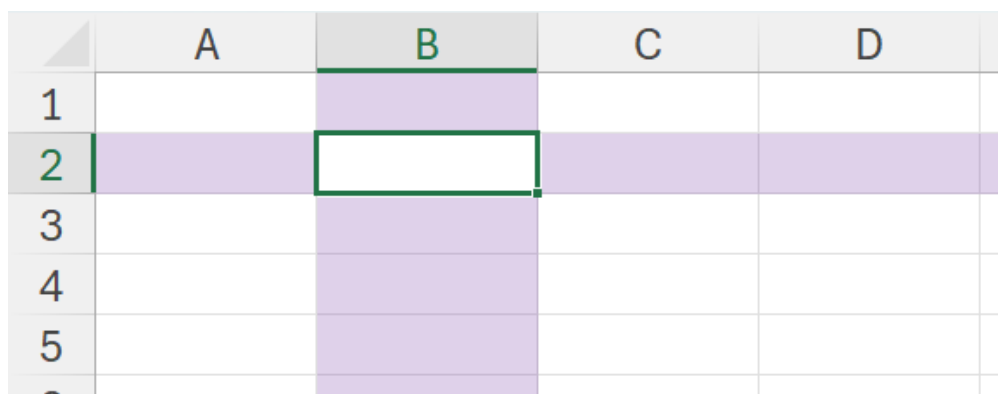
Found on the View tab in the Ribbon, ‘Focus Cell’ provides a small drop-down menu:

- **Focus Cell:** this toggles the feature on or off
- **Focus Cell Color:** this allows you to select from a wide array of, er, 10 colours to use for highlighting. Actually, that’s not true as ‘Recent Colors’ (*sic*) will show other colours that you may employ:



At this stage, we are not sure what the ‘High-contrast only’ toggle switch does as it appears to be permanently disabled.

So what does it do? It appears to enhance accessibility for the visually impaired as it highlights the active row and column of the cell selected, *viz.*



Here, cell B2 imitates Whoopi Goldberg’s 1985 hit ‘The Color Purple’, although clearly the colour displayed is not quite the same as the colours depicted in the ‘Focus Cell Color’ dialog.

Clearly, this feature helps users to zoom in on the cell selected. Other colours may be used, *e.g.*

	A	B	C	D
1				
2				
3				
4				
5				

Here, cell **A1** sees green. Envious, methinks.

If multiple cells are selected, the active cell remains highlighted:

	A	B	C	D	E
1					
2					
3					
4					
5					
6					
7					
8					

or

	A	B	C	D	E
1					
2					
3					
4					
5					
6					
7					
8					
9					

It does seem to act a little unexpectedly though if entire rows or columns are chosen:

	A	B	C	D
1				
2				
3				
4				
5				



	A	B	C	D
1				
2				
3				
4				
5				
6				

Presently, this feature does not appear to work in some pre-existing files or those with 'Freeze Pane' or 'Split' employed. We look forward to the documentation in due course – but this is hot off the press in the meantime!

## Over to AI



Here at SumProduct we write many articles for many magazines and our authoring contracts have been revised recently to ensure Artificial Intelligence hasn't taken over our duties. We can assure everyone they have not.

But it did get us thinking: *what if we did let AI loose and write a regular article for the newsletter?*

So that's what we've done. We thought AI could have a go at that evergreen chestnut (*when are chestnuts green? – Ed.*): **XLOOKUP** versus **VLOOKUP**. See what you think...

***This Month's Prompt: Discuss the Advantages and Disadvantages of XLOOKUP versus VLOOKUP in Excel***

### INTRODUCTION

Excel, the ubiquitous spreadsheet application developed by Microsoft, provides various functions to handle data efficiently. Among these functions, **VLOOKUP** and **XLOOKUP** are two of the most commonly used for performing lookups. While **VLOOKUP** has been a staple for many

years, **XLOOKUP** is a relatively new addition, designed to address some of the limitations of **VLOOKUP**. This article aims to explore the advantages and disadvantages of **XLOOKUP** compared to **VLOOKUP**.

### VLOOKUP: An Overview

**VLOOKUP**, short for "Vertical Lookup," is a function that searches for a value in the first column of a table array and returns a value in the same row from a specified column. The syntax for **VLOOKUP** is:

**=VLOOKUP(lookup\_value, table\_array, col\_index\_num, [range\_lookup])**

### Advantages of VLOOKUP

- **Simplicity:** VLOOKUP is straightforward and easy to understand, making it accessible for beginners
- **Wide Usage:** as a longstanding function, VLOOKUP has extensive documentation and community support
- **Compatibility:** VLOOKUP is available in all versions of Excel, ensuring backward compatibility for spreadsheets.

### Disadvantages of VLOOKUP

- **Column Limitation:** VLOOKUP can only search for data in the first column of the table array and can only return data from columns to the right of the lookup column
- **Performance Issues:** in large datasets, VLOOKUP can be slow, especially if the range is not sorted and the function has to perform an exact match
- **Static Column Index:** the column index number is hard-coded, making updates cumbersome if the table structure changes
- **Case Insensitivity:** VLOOKUP does not differentiate between uppercase and lowercase text, which can be a limitation in some scenarios.

### XLOOKUP: An Overview

XLOOKUP is a more flexible and powerful function introduced in Excel 2019 and Microsoft 365. It replaces the older functions like VLOOKUP, HLOOKUP and LOOKUP with a more versatile lookup capability. The syntax for XLOOKUP is:

**=XLOOKUP(lookup\_value, lookup\_array, return\_array, [if\_not\_found], [match\_mode], [search\_mode])**

### Advantages of XLOOKUP

- **Bidirectional Lookup:** XLOOKUP allows for both vertical and horizontal lookups, searching either to the left or right of the lookup column
- **Dynamic Ranges:** the function references entire arrays rather than specific columns, making it more adaptable to changes in table structure
- **Error Handling:** XLOOKUP includes an **if\_not\_found** argument, which provides a custom message if the lookup value is not found, improving error handling
- **Performance:** XLOOKUP is generally faster and more efficient, particularly with large datasets, as it performs an exact match by default
- **Enhanced Match Modes:** it supports exact match, approximate match, wildcard match and binary search modes, offering greater flexibility
- **Case Sensitivity:** while not inherently case-sensitive, XLOOKUP can be combined with other functions to perform case-sensitive searches.

### Disadvantages of XLOOKUP

- **Availability:** XLOOKUP is only available in Excel 2019 and Microsoft 365, limiting its use for those with earlier versions
- **Complexity:** the added functionality and options can make XLOOKUP more complex for beginners
- **Learning Curve:** users familiar with VLOOKUP may need time to learn and adapt to the new syntax and capabilities of XLOOKUP.

### Practical Example using VLOOKUP

Consider a table with employee ID numbers in column A and their names in column B. To find the name of the employee with ID 102, you would use:

**=VLOOKUP(102, A2:B10, 2, FALSE)**

This formula searches for the ID 102 in column A and returns the corresponding name from column B.

### Practical Example using XLOOKUP

Using the same table, to find the name of the employee with ID 102, the XLOOKUP formula would be:

**=XLOOKUP(102, A2:A10, B2:B10, "Not Found")**

This searches for the ID 102 in column A and returns the corresponding name from column B. If the ID is not found, it returns "Not Found."

### Conclusion

Both VLOOKUP and XLOOKUP serve the purpose of data lookup in Excel, but XLOOKUP offers enhanced flexibility, performance and functionality. While VLOOKUP remains an excellent tool for simpler tasks and ensures compatibility with older versions of Excel, XLOOKUP is the go-to function for more complex and dynamic datasets. Understanding the strengths and limitations of each function will allow users to choose the most appropriate tool for their specific needs.

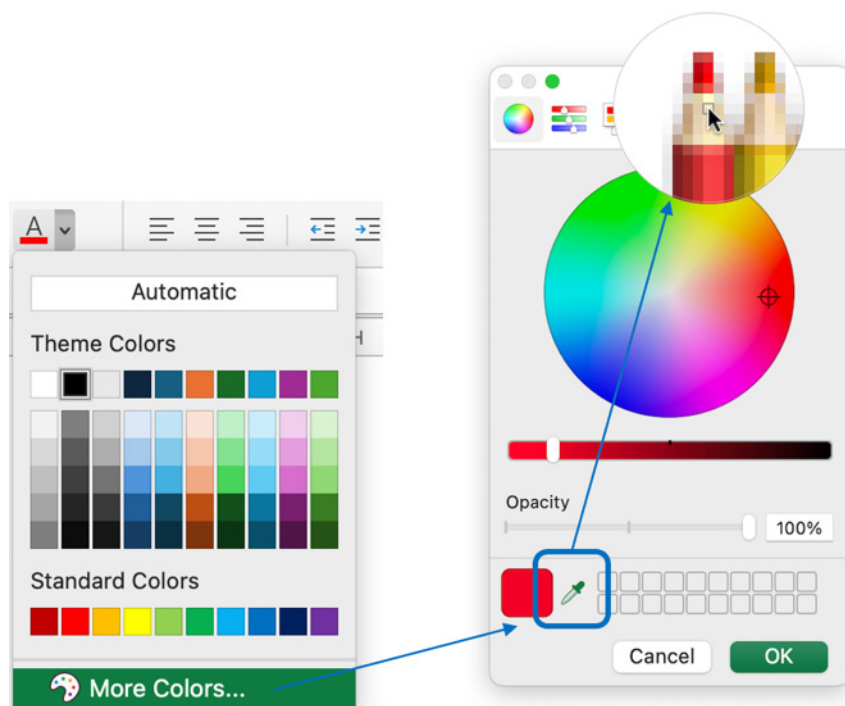
In summary, VLOOKUP is easier to use and widely supported but has significant limitations in flexibility and efficiency. On the other hand, XLOOKUP, despite its complexity and limited availability in older Excel versions, stands out as a powerful and versatile function for modern spreadsheet tasks.

What do you think? Do you agree? (I am still trying to find those agreed best practices and standards for financial modelling...) Send us your comments and ideas for questions to [contact@sumproduct.com](mailto:contact@sumproduct.com).

## 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 handy feature that is only available in Excel if you're using a Mac. There's a great colour picker that lets you easily choose any colour that you can see on your screen.

Choosing colours for things like the cell fill or the font in Excel is a common task, and sometimes you just want to pick a colour that you see on screen. If you use Excel for Mac, you can do just that, and we'll show you how to use this feature that doesn't exist in Excel for Windows (as at the time of writing).

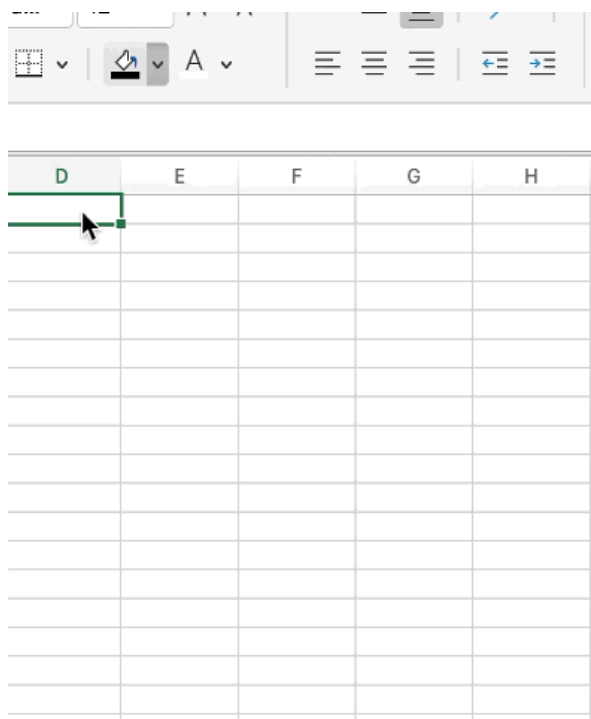


### How to Pick Any Colour

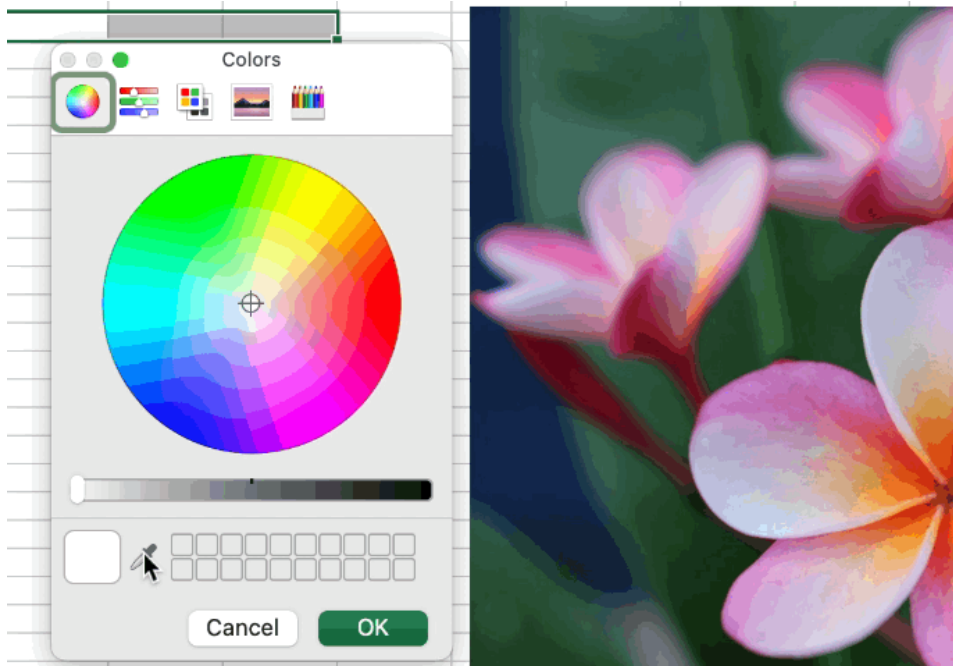
Wherever you need to choose a colour in Excel, such as the fill or font colour, cell borders, shapes, conditional formats, you can grab any colour from your screen. Just follow the simple steps below.

As our example, we'll use the task of setting the fill colour for a cell, but these steps apply to any task where you need to pick a colour.

- Select a cell and expand the 'Fill Color' menu on the Home tab of the Ribbon



- Select a cell and expand the 'Fill Color' menu on the Home tab of the Ribbon
- Choose "More Colors...". A new dialog will appear where you have numerous options for picking colours
- Select the icon that has an eyedropper
- A magnifying circle will appear. You can then move the circle anywhere on the screen to get a closer view of the colour on that part of the screen. When you find a colour you like, click on it



- The colour will appear in the large square in the Colors dialog box, indicating the colour that will be applied to the cell
- Press OK.

You're done. You can pick a colour this way anywhere in Excel. In fact, you can pick a colour this way in most apps on Mac, because the Colors dialog and the eyedropper tool are part of the Mac operating system.

Furthermore, if you want to know the RGB (red, blue, green) values as you're picking a colour, just press **SPACE** on your keyboard. You'll see the RGB values appear in the magnifying circle, as shown in the screen shot below:

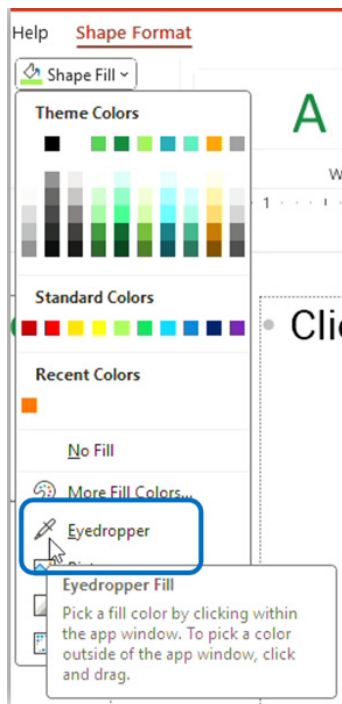


## How to Pick a Colour on Windows

At the time of writing, there is no similar colour picking tool in Excel for Windows, but you can do it in PowerPoint for Windows. It works very much the same, but it's limited to picking a colour from the PowerPoint presentation. You can't pick a colour from any part of the screen outside of the presentation, but it still makes it easy to find the colour you want.

If you've picked a colour in PowerPoint, you can copy it to Excel by copying the HEX code which you can find in the colour picker's RGB panel.

If the colour you want is outside the PowerPoint window, you can press **SHIFT + Windows Key + S** to take a screen clipping of it. Just paste that onto your slide and then use the eyedropper to pick the colour.



We hope you find this topic helpful. Check back for more details about Excel for Mac and how it's different to Excel for Windows.

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, just when you thought AutoSave couldn't get any worse...*

As you might be aware, we're not particularly fond of the AutoSave feature here at SumProduct. From our perspective, AutoSave is only good for overwriting files when you open them up just to do some quick test calculations, intending on hitting 'No' when asked if you want to save the file, only to find that AutoSave has ruined your perfectly balanced balance sheets and models.

Well, a recently raised issue between the AutoSave feature and the long-standing VBA command **ActiveWorkbook.Saved = True** has given us more ammunition in the battle against AutoSave.

In a nutshell, **ActiveWorkbook.Saved** is the property in Excel that helps to determine whether changes have been made in a file, and therefore, whether you are prompted to save the file before closing. A common use for this bit of code is to 'trick' Excel into thinking that

the file has been saved to avoid popping up further dialog boxes. We use this when we have **BeforeSave** macros that change the sheet presentation when the file is saved. Because these changes are generally reverted once the file is saved, setting **ActiveWorkbook.Saved = True** will stop Excel from prompting you to save the file again when you try to close it.

Unfortunately, this is intrinsically linked to the way that AutoSave looks at changes in your file and determines whether to resync and save it back to the server. The technical basis for this is a bit long to go into here, but the Microsoft team just released a good primer on what is causing the issue, and some suggested workarounds: <https://docs.microsoft.com/en-us/office/vba/library-reference/concepts/how-autosave-impacts-addins-and-macros>.

```
VB Copy  
  
Sub UseAutoSaveOn()  
    ActiveWorkbook.AutoSaveOn = False  
    MsgBox "This workbook is being saved automatically: " & ActiveWorkbook.AutoSaveOn  
End Sub
```

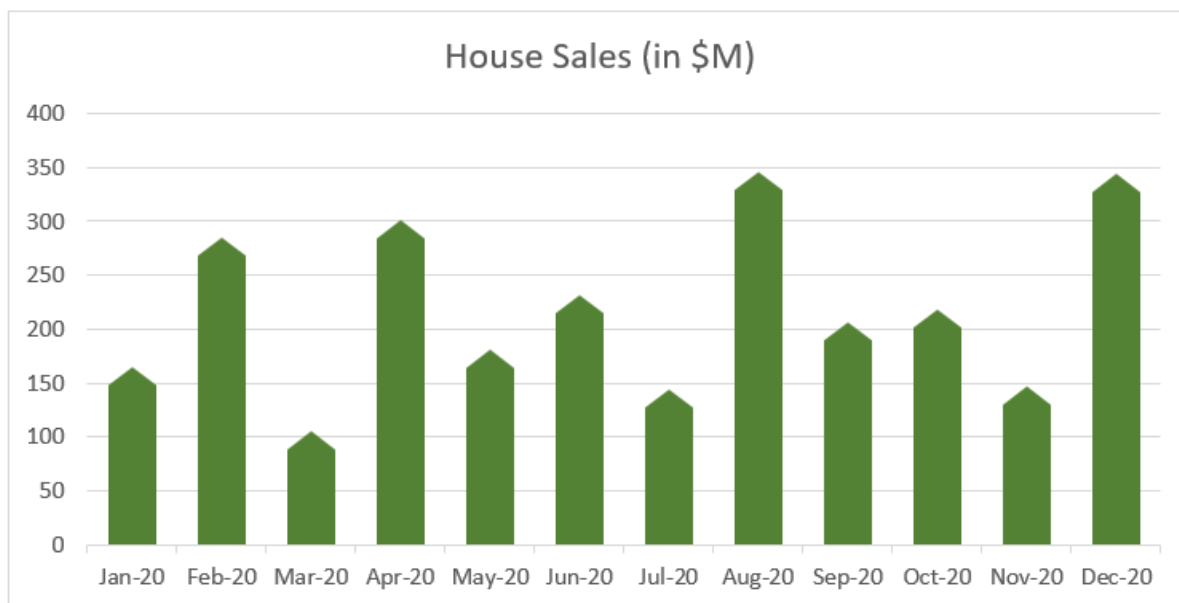
This is Microsoft's suggested script to get rid of AutoSave for files where you might be using **ActiveWorkbook.Saved = True**. Now, if only we could make this default in every file...

More next time.

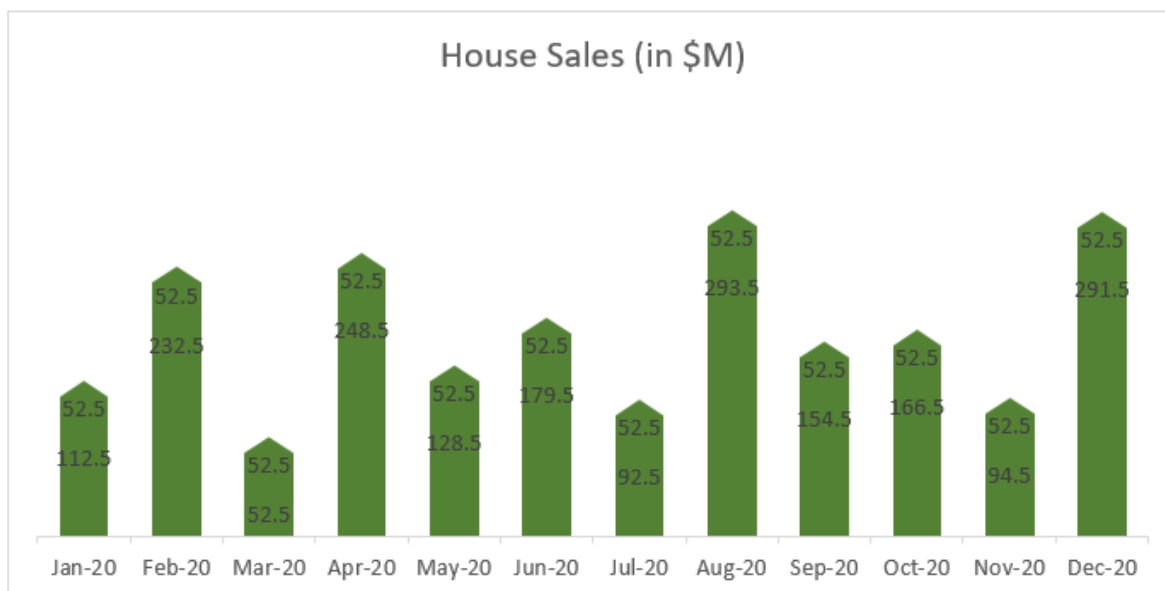
## Charts and Dashboards

*It's time to chart our progress with an introductory series into the world of creating charts and dashboards in Excel. This month, we continue to discuss customising chart shapes.*

In the last newsletter, we have talked about customising chart shapes, whereby we created a chart for house sales with the house-shaped columns like the one below:



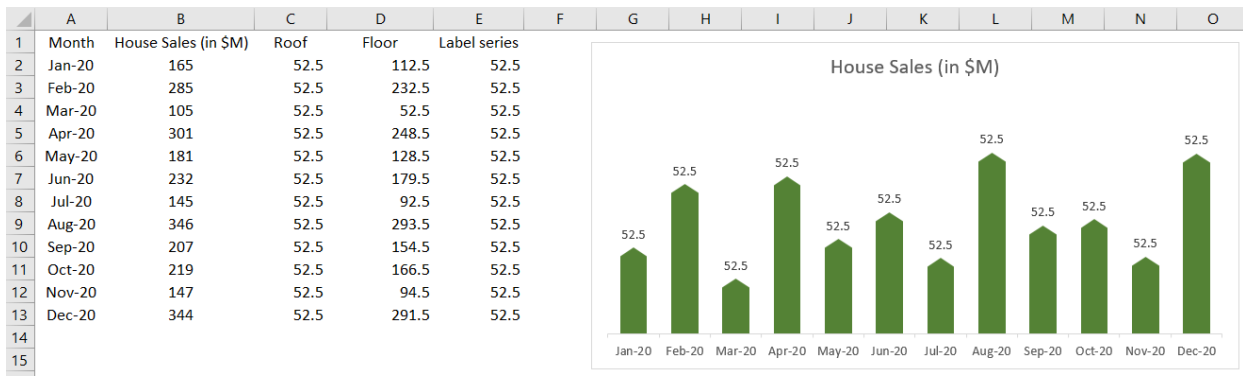
The chart is actually a Stacked Column chart and we left the vertical axis in place. Now, we want the data labels representing sales to be displayed on top of each column. However, when we remove the vertical axis and gridlines and add data labels to the chart, since it is a Stacked Column chart with two data series, we get two series of data labels.



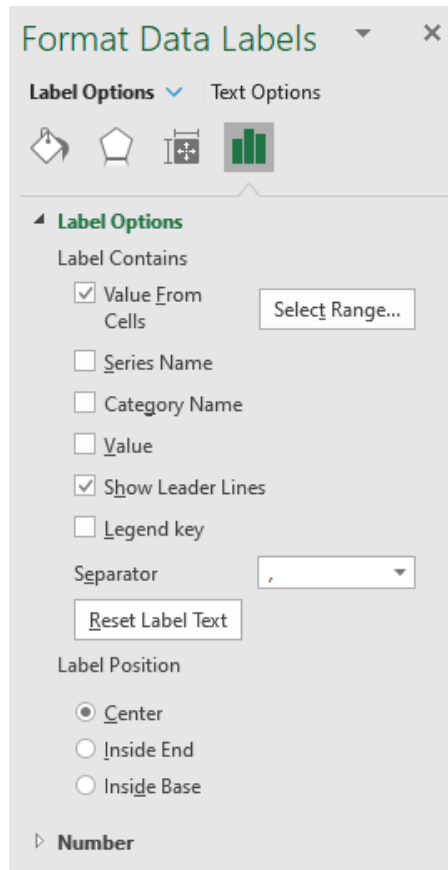
Unlike the Column chart options where we can add data labels at the outside end of the column, in the Stacked Column chart, we need a few tweaks to get this done:

- we will need a helper data series to top the Stack chart, say 'Label series', whose values are equal to the **Roof** series
- right-click on the chart and choose 'Select data' and add this series to the chart
- right-click on the top series and choose 'Format Data Series' in the 'Series Options'
- let the Fill be 'No fill' and Border be 'No line' to make this series invisible
- to delete unnecessary series of data labels, just click on a label of the series and hit Delete.

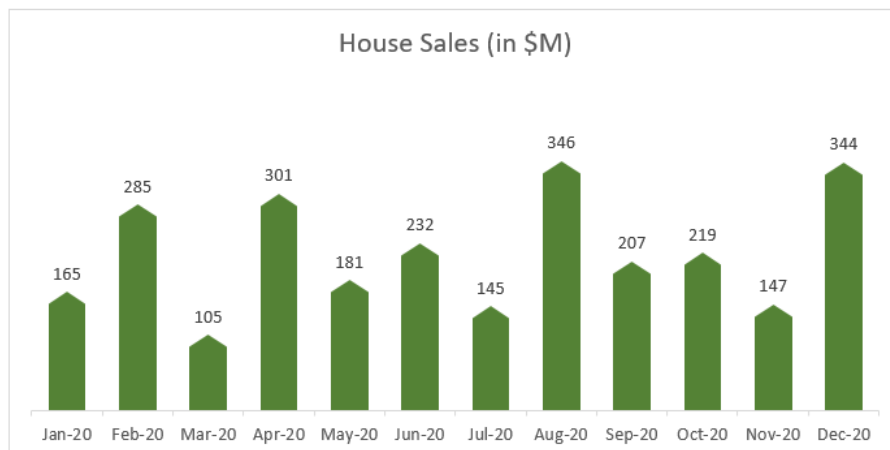
Now we have brought the data labels to the top of the columns.



To get the data labels to point to the house sales instead of the helper stack data, we will use dynamic chart labels. By clicking on one of the labels, on the 'Format Data Labels' panel, under the 'Label Options', uncheck 'Value' and check 'Value From Cells' instead.



In the 'Data Label Range' dialog that occurs, point the range to the house sales, which is the range **B2:B13**. The chart is now complete.



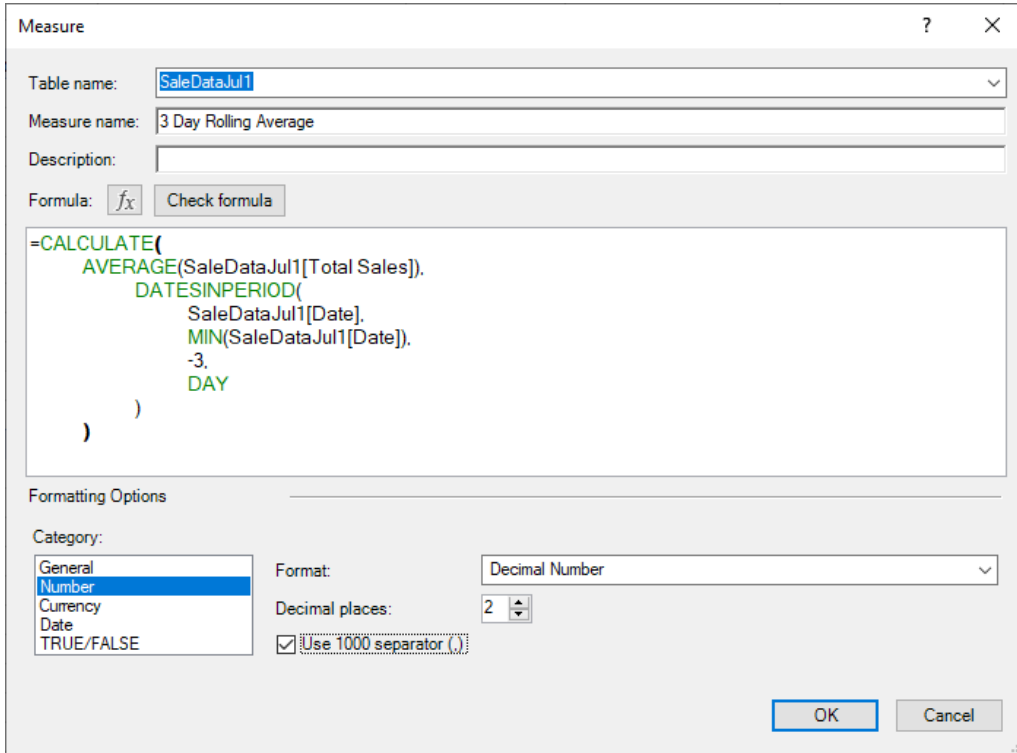
More next time.

# Power Pivot Principles

We continue our series on the Excel COM add-in, Power Pivot. This month, we expand on the rolling average concept and show you how to include a dynamic selection range with a rolling average measure.

We have created a rolling average measure previously. As a recap, we used the following measure to calculate the three-day rolling average:

```
=CALCULATE(  
    AVERAGE(SaleDataJul1[Total Sales]),  
    DATESINPERIOD(  
        SaleDataJul1[Date],  
        MIN(SaleDataJul1[Date]),  
        -3,  
        DAY  
    )  
)
```



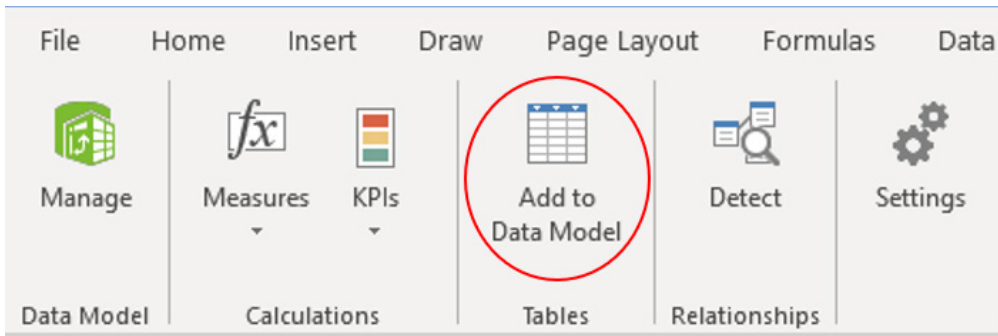
This month, we are going to build in the ability to select the number of days we wish to include in the rolling average. The first step we must take is to create a disconnected table.

In this example we are going to use this disconnected table, namely the **Days\_Average** table:

	A	B	C
1			
2		Days to Average	
3			1
4			2
5			3
6			4
7			5
8			6
9			7
10			8
11			9
12			10
13			



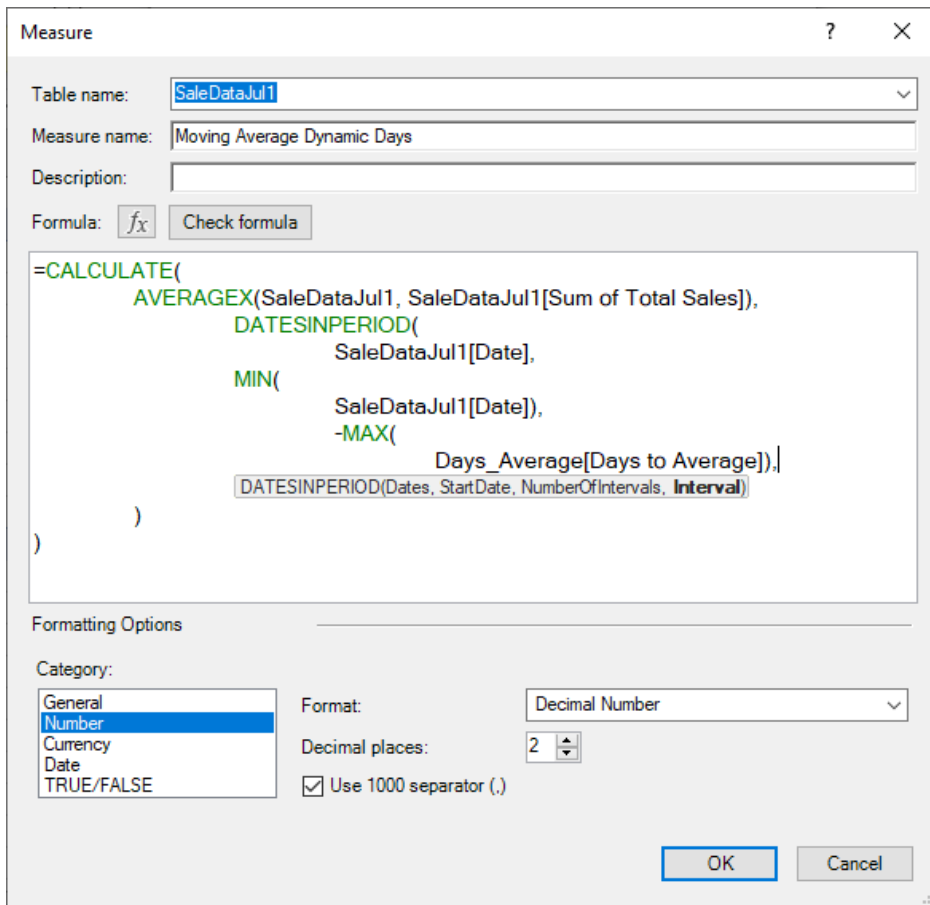
It has a range of one to 10 days. We then add this table to the data model:



We do not create a relationship between this table and our sales data table (called **SaleDataJul1**); we want these two tables to remain disconnected. Otherwise, this trick won't work.

The next step is to modify our previous measure:

```
=CALCULATE(
    AVERAGEX(SaleDataJul1, SaleDataJul1[Sum of Total Sales]),
    DATESINPERIOD(
        SaleDataJul1[Date],
        MIN(
            SaleDataJul1[Date]),
        -MAX(
            Days_Average[Days to Average]),
        DAY
    )
)
```



Instead of defining the days in the **DATESINPERIOD** function as '3' we use the **MAX** function to determine the maximum number in the **Days\_Average** table. Before we add this measure to our PivotTable, we need to create a slicer, so that our users may select the number of days to be included in our rolling average.

After creating the slicer and including our measure in our PivotTable, we have something similar to the following:

	A	B	C	D	E	F
1						
2		Date	Sum of Total Sales	Moving Average Dynamic Days	Days to Average	
3		1/07/2018	353	353.00	1	
4		2/07/2018	446	399.50	2	
5		3/07/2018	383	394.00	3	
6		4/07/2018	283	366.25	4	
7		5/07/2018	285	350.00	5	
8		6/07/2018	446	366.00	6	
9		7/07/2018	323	359.86	7	
10		8/07/2018	299	352.25	8	
11		9/07/2018	356	352.67		
12		10/07/2018	435	360.90		
13		11/07/2018	318	357.40		
14		12/07/2018	358	348.60		
15		13/07/2018	413	351.60		
16		14/07/2018	346	357.90		
17		15/07/2018	449	374.30		
18		16/07/2018	489	378.60		
19		17/07/2018	277	374.00		
20		18/07/2018	440	388.10		
21		19/07/2018	324	384.90		
22		20/07/2018	381	379.50		
23		21/07/2018	301	377.80		

**PivotTable Fields**

Active All

Choose fields to add to report:

Search

Rolling Average All  
 Max Date  
 Moving Average Dynamic Days

Table1

Drag fields between areas below:

Filters	Columns
	Σ Values
Rows	Σ Values
Date	Sum of Total Sales
	Moving Average Dyna...

The moving average range seems to be set at 10 days. That is because we have not selected a number in the slicer. Therefore, the measure is picking up the maximum value in the **Days\_Average** table, which is 10.

After selecting five (5) days, we the following result:

	A	B	C	D	E	F
1						
2		Date	Sum of Total Sales	Moving Average Dynamic Days	Days to Average	
3		1/07/2018	353	353.00	1	
4		2/07/2018	446	399.50	2	
5		3/07/2018	383	394.00	3	
6		4/07/2018	283	366.25	4	
7		5/07/2018	285	350.00	5	
8		6/07/2018	446	368.60	6	
9		7/07/2018	323	344.00	7	
10		8/07/2018	299	327.20	8	
11		9/07/2018	356	341.80		
12		10/07/2018	435	371.80		
13		11/07/2018	318	346.20		
14		12/07/2018	358	353.20		
15		13/07/2018	413	376.00		
16		14/07/2018	346	374.00		
17		15/07/2018	449	376.80		
18		16/07/2018	489	411.00		
19		17/07/2018	277	394.80		
20		18/07/2018	440	400.20		
21		19/07/2018	324	395.80		
22		20/07/2018	381	382.20		
23		21/07/2018	301	344.60		

**PivotTable Fields**

Active All

Choose fields to add to report:

Search

Rolling Average All  
 Max Date  
 Moving Average Dynamic Days

Table1

Drag fields between areas below:

Filters	Columns
	Σ Values
Rows	Σ Values
Date	Sum of Total Sales
	Moving Average Dyna...

We can switch between any number in our Days to Average slicer. The Moving Average Dynamic Days measure will update accordingly:

	A	B	C	D	E	F
1						
2		Date	Sum of Total Sales	Moving Average Dynamic Days	Days to Average	
3		1/07/2018	353	353.00	1	
4		2/07/2018	446	399.50	2	
5		3/07/2018	383	394.00	3	
6		4/07/2018	283	366.25	4	
7		5/07/2018	285	349.25	5	
8		6/07/2018	446	349.25	6	
9		7/07/2018	323	334.25	7	
10		8/07/2018	299	338.25	8	
11		9/07/2018	356	356.00		
12		10/07/2018	435	353.25		
13		11/07/2018	318	352.00		
14		12/07/2018	358	366.75		
15		13/07/2018	413	381.00		
16		14/07/2018	346	358.75		
17		15/07/2018	449	391.50		
18		16/07/2018	489	424.25		
19		17/07/2018	277	390.25		
20		18/07/2018	440	413.75		
21		19/07/2018	324	382.50		
22		20/07/2018	381	355.50		
23		21/07/2018	301	361.50		

**PivotTable Fields**

Active All

Choose fields to add to report:

Search

Rolling Average All  
 Max Date  
 Moving Average Dynamic Days

Table1

Drag fields between areas below:

Filters	Columns
	Σ Values
Rows	Σ Values
Date	Sum of Total Sales
	Moving Average Dyna...

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](http://www.sumproduct.com/blog). If you wish to read more in the meantime, simply check out our Blog section each Wednesday. This month, we look at how **M** and culture can go together.

Previously, some of the **M** functions we have used have included a culture parameter. For example, I have mentioned the following function in the past:

**Number.From(value as any, optional culture as nullable text)** as nullable number

At that time, we were converting a date to its numerical equivalent in order to create a list. We took the default local culture, but now we'd like to know the other options available. We are going to try this with another culture to see what happens. We have a large choice of cultures. The following information comes from the Microsoft help pages:

Culture	Language
af	Afrikaans
am	Amharic
ar-sa	Arabic (Saudi Arabia)
as	Assamese
az-Latn	Azerbaijani (Latin)
be	Belarusian
bg	Bulgarian
bn-BD	Bangla (Bangladesh)
bn-IN	Bangla (India)
bs	Bosnian (Latin)
ca	Catalan Spanish
ca-ES-valencia	Valencian
cs	Czech
cy	Welsh
da	Danish
de	German (Germany)
de-de	German (Germany)
el	Greek
en-GB	English (United Kingdom)
en-US	English (United States)
es	Spanish (Spain)
es-ES	Spanish (Spain)
es-US	Spanish (United States)
es-MX	Spanish (Mexico)
et	Estonian
eu	Basque
fa	Persian
fi	Finnish
fil-Latn	Filipino
fr	French (France)
fr-FR	French (France)
fr-CA	French (Canada)
ga	Irish
gd-Latn	Scottish Gaelic
gl	Galician
gu	Gujarati
ha-Latn	Hausa (Latin)
he	Hebrew

Culture	Language
hi	Hindi
hr	Croatian
hu	Hungarian
hy	Armenian
id	Indonesian
ig-Latn	Igbo
is	Icelandic
it	Italian (Italy)
it-it	Italian (Italy)
ja	Japanese
ka	Georgian
kk	Kazakh
km	Khmer
kn	Kannada
ko	Korean
kok	Konkani
ku-Arab	Central Kurdish
ky-Cyrl	Kyrgyz
lb	Luxembourgish
lt	Lithuanian
lv	Latvian
mi-Latn	Maori
mk	Macedonian
ml	Malayalam
mn-Cyrl	Mongolian (Cyrillic)
mr	Marathi
ms	Malay (Malaysia)
mt	Maltese
nb	Norwegian (Bokmål)
ne	Nepali (Nepal)
nl	Dutch (Netherlands)
nl-BE	Dutch (Netherlands)
nn	Norwegian (Nynorsk)
nso	Sesotho sa Leboa
or	Odia
pa	Punjabi (Gurmukhi)
pa-Arab	Punjabi (Arabic)
pl	Polish

Culture	Language
prs-Arab	Dari
pt-BR	Portuguese (Brazil)
pt-PT	Portuguese (Portugal)
qut-Latn	K'iche'
quz	Quechua (Peru)
ro	Romanian (Romania)
ru	Russian
rw	Kinyarwanda
sd-Arab	Sindhi (Arabic)
si	Sinhala
sk	Slovak
sl	Slovenian
sq	Albanian
sr-Cyrl-BA	Serbian (Cyrillic, Bosnia and Herzegovina)
sr-Cyrl-RS	Serbian (Cyrillic, Serbia)
sr-Latn-RS	Serbian (Latin, Serbia)
sv	Swedish (Sweden)
sw	Kiswahili
ta	Tamil

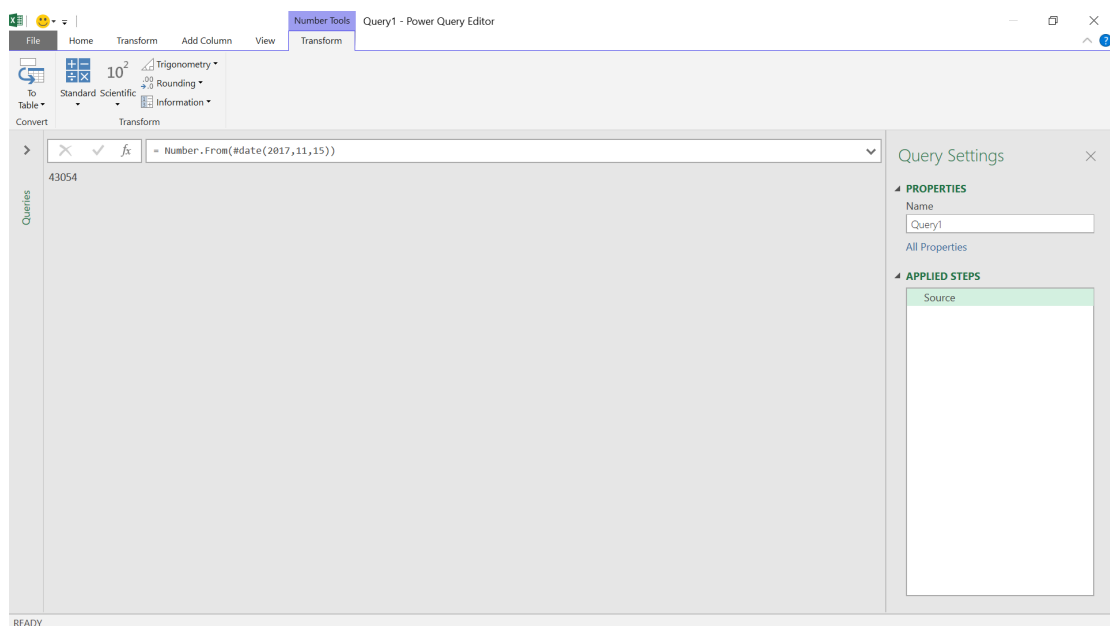
Culture	Language
te	Telugu
tg-Cyrl	Tajik (Cyrillic)
th	Thai
ti	Tigrinya
tk-Latn	Turkmen (Latin)
tn	Setswana
tr	Turkish
tt-Cyrl	Tatar (Cyrillic)
ug-Arab	Uyghur
uk	Ukrainian
ur	Urdu
uz-Latn	Uzbek (Latin)
vi	Vietnamese
wo	Wolof
xh	isiXhosa
yo-Latn	Yoruba
zh-Hans	Chinese (Simplified)
zh-Hant	Chinese (Traditional)
zu	isiZulu

<sup>1</sup> When you specify es-MX or es-US in the request, the culture is converted to es-ES.

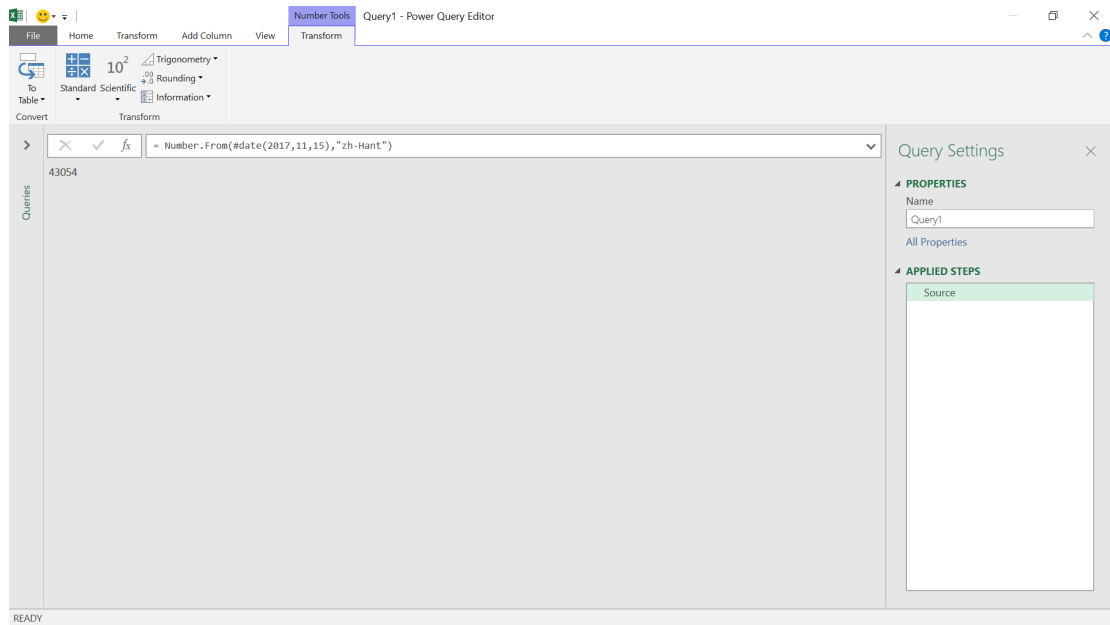
<sup>2</sup> When you specify fr-CA in the request, the culture is converted to fr-FR.

Previously, we wanted to convert a date to a number:

**= Number.From(#date(2017,11,15))**



Let's try with another culture.



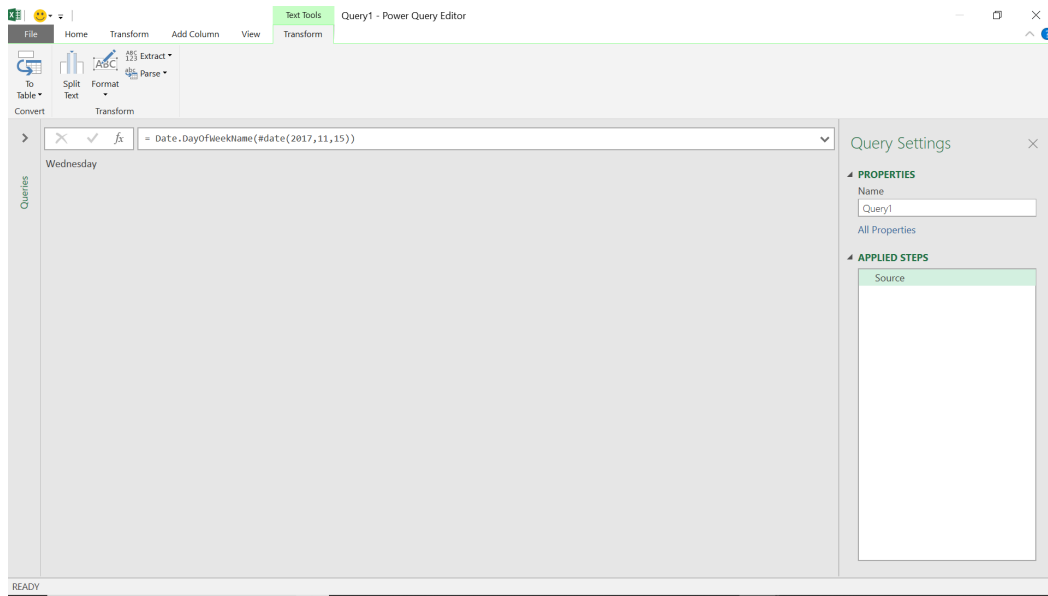
In this case, we get the same answer. However, there are many **M** functions that can use culture, as the following list shows:

- |                              |                                   |
|------------------------------|-----------------------------------|
| <b>Byte.From</b>             | <b>Int64.From</b>                 |
| <b>Comparer.FromCulture</b>  | <b>Int8.From</b>                  |
| <b>Currency.From</b>         | <b>Number.From</b>                |
| <b>Date.DayOfWeekName</b>    | <b>Number.FromText</b>            |
| <b>Date.From</b>             | <b>Number.ToText</b>              |
| <b>Date.FromText</b>         | <b>Percentage.From</b>            |
| <b>Date.MonthName</b>        | <b>Single.From</b>                |
| <b>Date.ToText</b>           | <b>Table.TransformColumnTypes</b> |
| <b>DateTime.From</b>         | <b>Text.Format</b>                |
| <b>DateTime.FromText</b>     | <b>Text.From</b>                  |
| <b>DateTime.ToText</b>       | <b>Text.Lower</b>                 |
| <b>DateTimeZone.From</b>     | <b>Text.Proper</b>                |
| <b>DateTimeZone.FromText</b> | <b>Text.Upper</b>                 |
| <b>DateTimeZone.ToText</b>   | <b>Time.From</b>                  |
| <b>Decimal.From</b>          | <b>Time.FromText</b>              |
| <b>Double.From</b>           | <b>Time.ToText</b>                |
| <b>Int16.From</b>            | <b>Value.FromText</b>             |
| <b>Int32.From</b>            |                                   |

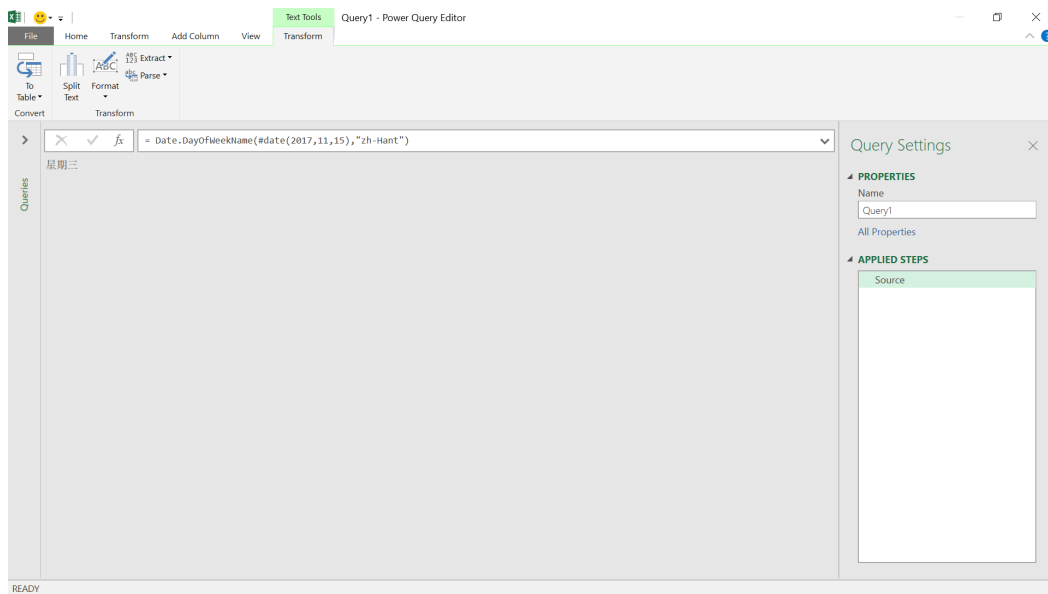
So, we'll try looking at the function **Date.DayOfWeekName**:

**Date.DayOfWeekName**(date as any, optional culture as nullable text)

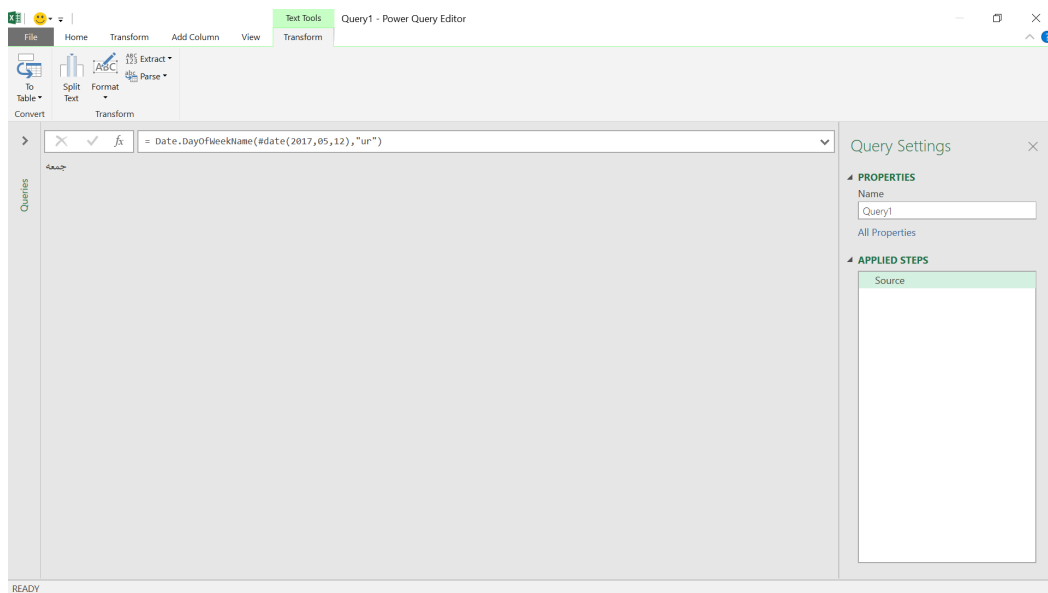
Firstly, with our local culture:



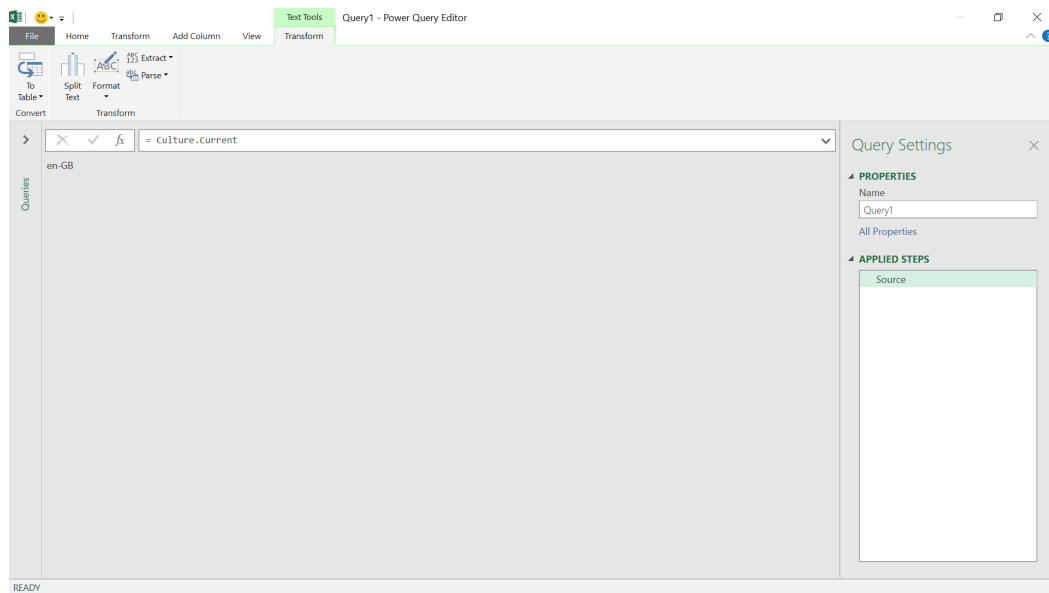
So far, so good. If we change the culture to Chinese, we get



and in Urdu...



Much more satisfying results! Of course, this does leave the question, how do we know the local culture that Power Query is using?



The **M** function

**Culture.Current**

will tell me the current setting. Simple!

Until next month.

## Power BI Updates

Some big news for those that like to “go dark”.



As of these latest updates, you can now choose from a variety of themes for Power BI Desktop, including the oft-requested dark mode. You may now personalise it to match your preferences and working environment. In addition, Microsoft has now consolidated similar options in the menu bar and streamlined the button text for better readability and more responsive screen sizing.

The Copilot chat pane will now automatically provide text-based answers and summaries across all pages in a report. Previously, users had to specifically request cross-page summaries or click a 'base summary on

the entire report' button. With this update, cross-page summaries and answers are now the default setting.

There is also a transformative new feature designed to redefine how you might manage and consume metrics, featuring visuals and Copilot insights Called Metrics Hub. This is an innovative metric layer within Fabric, aimed at helping you define, discover and reuse trusted metrics in a straightforward fashion. This feature allows trusted creators within an organisation to develop standardised metrics that incorporate essential business logic, ensuring consistency.

The full list of updates is as follows:

### General

- Dark mode is now available in Power BI Desktop

### Copilot and AI

- Default full report summaries and answers in the Copilot pane

### Reporting

- Updated menu bar in Power BI Service view mode
- Visual calculations update (Preview)
- Visual level format strings (Preview)
- Power BI in Teams: now supporting Multi Factor Authentication
- Multiple organisation applications per workspace with custom colours available (Preview)
- Subscribe to reports on the last day of the month

### Modelling

- Introducing Fabric Metrics Layer: metric management in Fabric (Preview)
- Live edit of semantic models in Direct Lake mode with Power BI Desktop
- Q&A now supports semantic models with OLS

### Data Connectivity

- Connected tables in Excel are now available in Semi-Annual channel

### Mobile

- NFC tag support in Power BI Mobile

### Visualisations

- New in AppSource:
  - KPI by Powerviz
  - Drill Down Combo Bar PRO by ZoomCharts
  - BI Pixie by DataChant: measure the engagement of your BI audience
  - Decomposition Tree by JTA

### Other

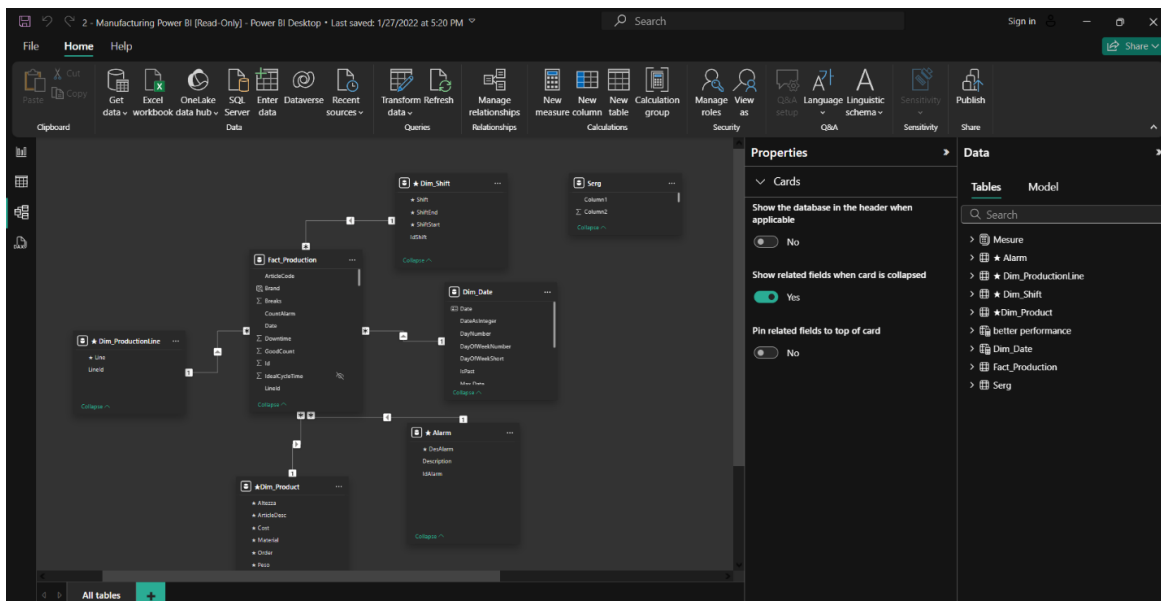
- Announcing the General Availability of the 'Get data' experience in Power BI Report Builder.

Let's look at each in turn.

### Dark mode is now available in Power BI Desktop

Perhaps more eagerly awaited than a Taylor Swift show or an investigation into Oasis ticket pricing, Microsoft has finally announced a new feature in Power BI Desktop that allows you to choose from a variety of themes, including the most requested dark mode, which replaces all the chrome around the design surface with white-on-black text instead of black-on-white. Thus, you may now customise your data visualisation experience to match your preferences and working environment.

Whether you're working late into the night or simply prefer a darker interface (all you Jack Bauer wannabes out there), dark mode provides an alternative, modern look that reduces eye strain and enhances focus. You can enable any theme by going to **Options and settings -> Global -> Report settings -> Personalization**.



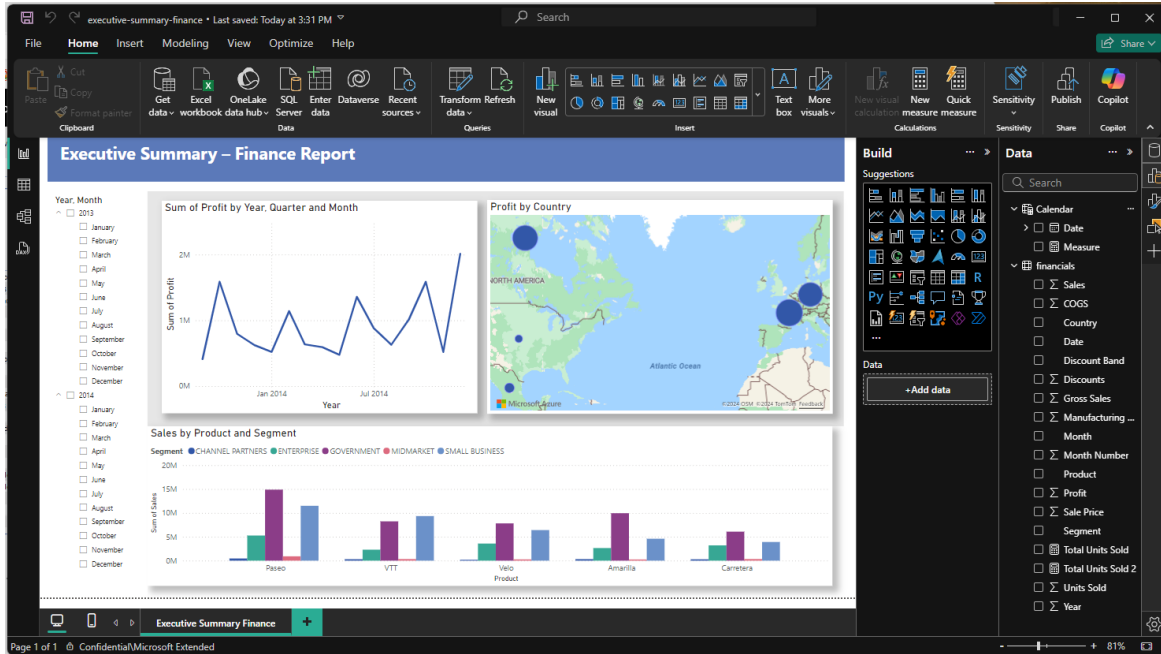


In the aforementioned **Personalization** section, you may select the theme you want:

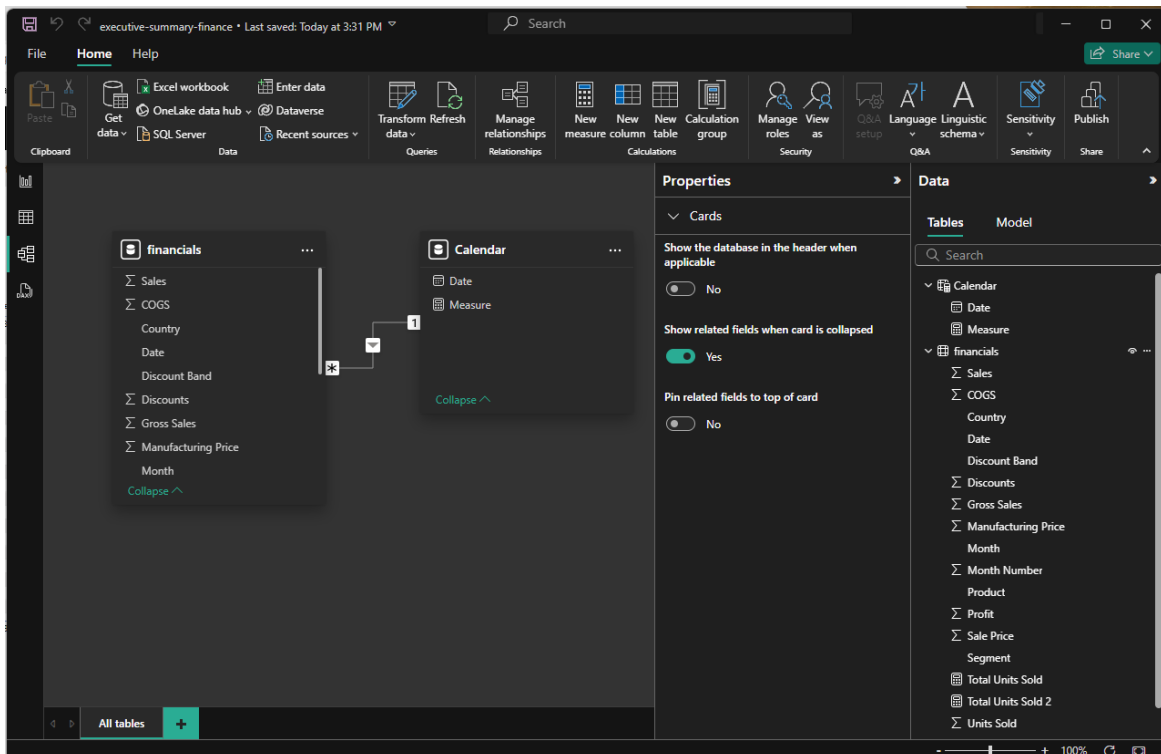
- **Default mode:** the standard white theme of the desktop application
- **Dark mode:** a dark theme that transforms the desktop application into a darker interface
- **Light mode:** a white theme with a more stylish look than the default theme
- **User system mode:** adapts the theme based on your Windows system settings.

There's a mode for every window in Power BI Desktop. Using dark mode as an example:

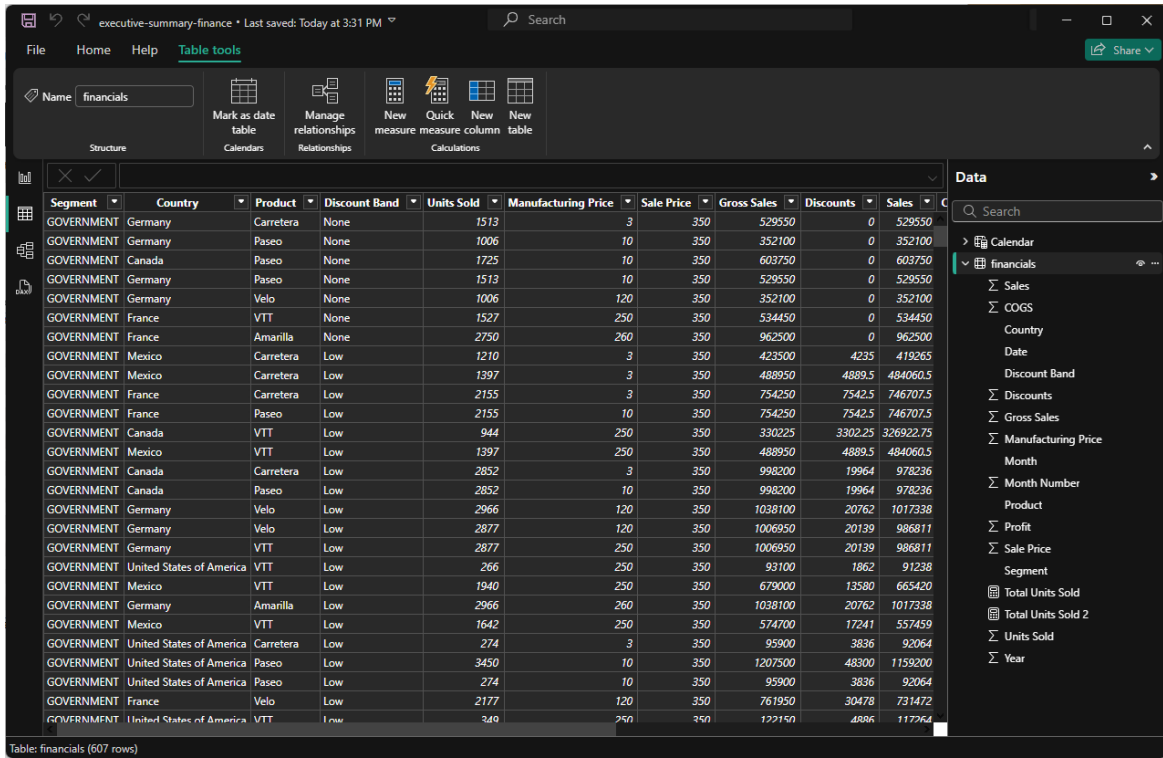
- Report view



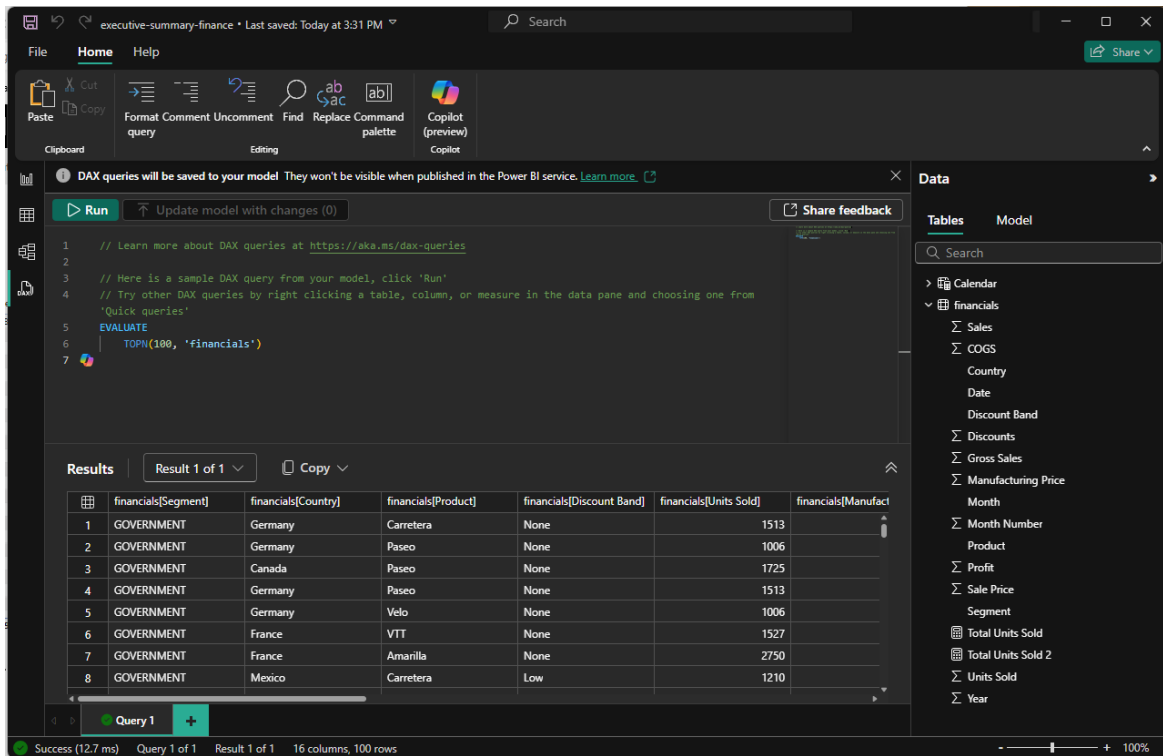
- Model view



- Table view



- DAX query view.



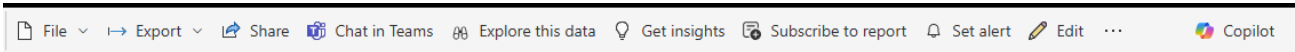
### Default full report summaries and answers in the Copilot pane

The Copilot chat pane will now automatically provide text-based answers and summaries across all pages in a report. Previously, you had to specifically request cross-page summaries or click a 'base summary on the entire report' button. With this update, cross-page summaries and answers are now the default setting, streamlining the exploration process. This enhancement ensures that users can efficiently navigate and understand their data across all pages in applications and reports.

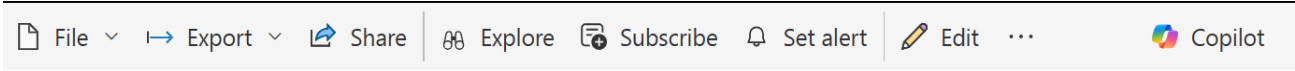
### Updated menu bar in Power BI Service view mode

Over the last few years, Microsoft has been adding many new features to Power BI that have been showcased in the menu bar. With these new feature additions, the menu had become cluttered and hard to navigate. Power BI has now consolidated similar options and streamlined the button text for better readability and responsive screen sizing. This change is only applicable to the view mode of reports and applications.

Before:



After:



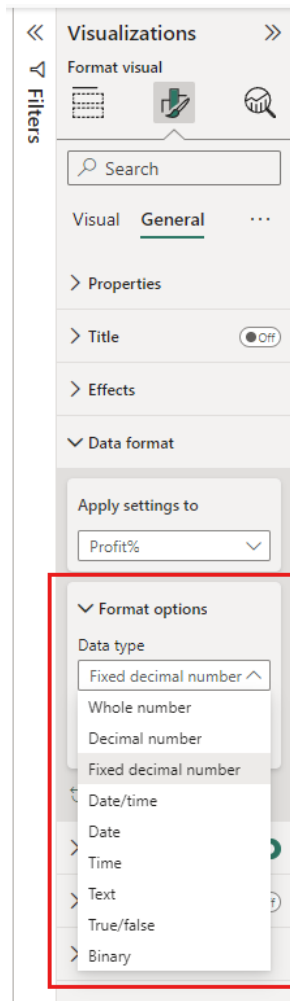
It should be noted that:

- 'Chat in Teams' is now available through the Share option
- 'Get insights' was moved to the "..." menu.

### Visual calculations update (Preview)

Visual calculations are now on by default so you can start using them right away without having to enable the Preview feature. Of course, you can still turn them off if you wish by disabling the visual calculations preview in **Options and Settings -> Options -> Preview features**.

Apart from adding more formatting options as part of the visual level format strings (*see below*), this month's updates also enables data types for visual calculations, which allows you to now set the data type for your visual calculations in the Visual format pane under **General -> Data format**:



Selecting the correct data type is important as it not only influences how your visual calculation can be used in your charts and in your calculations, but also determines which formatting options are available for further customisation. You should bear in mind that if the data type and data do not match, the data type is ignored. For example, if you have data rows that contain 'ABC', and set the data type to 'Decimal number' then

the data cannot be formatted as decimal number, and the data type will not be applied.

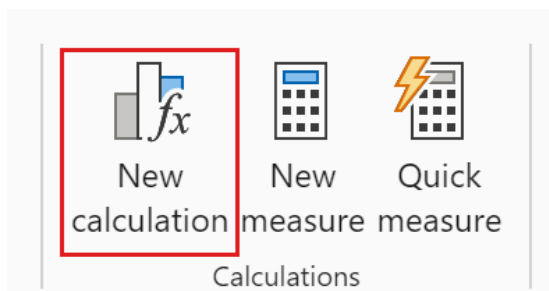
It should further be noted that the data type setting is only available for visual calculations, not for fields or measures as their data type can be set in the model only.

As a reminder, you can now add calculations directly onto your visual using visual calculations, which are **DAX** calculations that are defined and executed directly on a visual. A calculation can refer to any data in the visual, including columns, measures or other visual calculations. This approach removes the complexity of the semantic model and simplifies the process of writing **DAX**. You may use visual calculations to complete common business calculations such as running sums or moving averages. Visual calculations make it easy to create calculations

that were previously very hard or even almost impossible to construct.

To use visual calculations whilst it remains in Preview, you need to enable it in **Options and Settings -> Options -> Preview features**. Select visual calculations and click OK. Visual calculations will then be enabled after Power BI Desktop is restarted.

To add a visual calculation, you first need to select a visual. Then, select the 'New calculation' button in the Ribbon:



To add a visual calculation, type the expression in the Formula bar in the 'Visual Calculations edit mode' that opens. For example, in a visual that contains **Sales Amount** and **Total Product Cost by Fiscal Year**, you can add a visual calculation that calculates the profit for each year by simply typing:

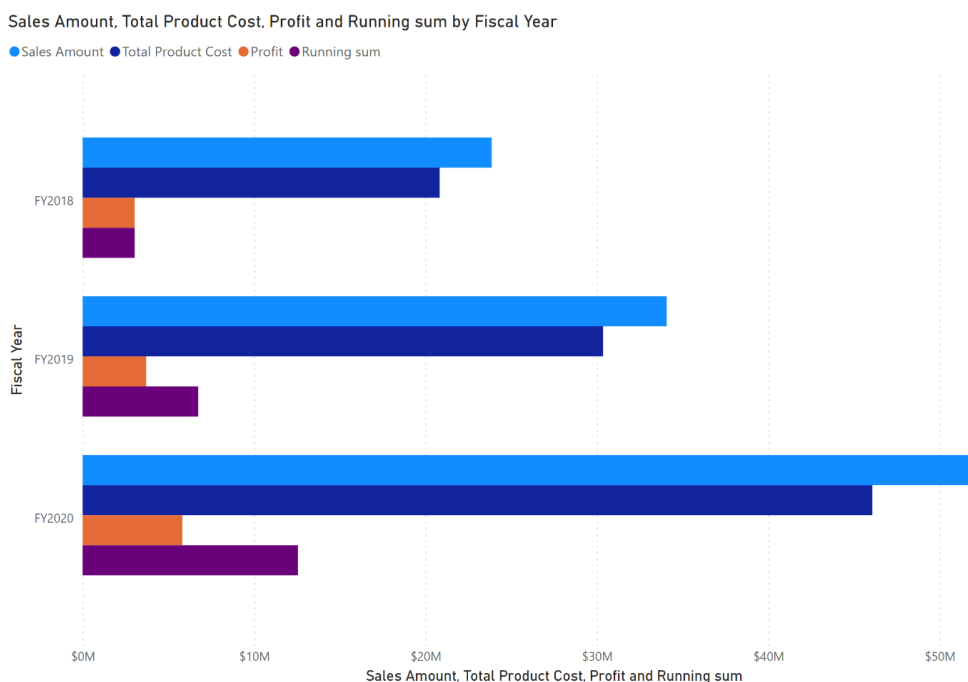
$$\text{Profit} = [\text{Sales Amount}] - [\text{Total Product Cost}]$$

Fiscal Year	Sales Amount	Total Product Cost	Profit
FY2018	\$23,860,891.17	\$20,824,957.60	3,035,933.57
FY2019	\$34,070,108.50	\$30,362,108.34	3,708,000.16
FY2020	\$51,878,274.54	\$46,070,842.02	5,807,432.52
<b>Total</b>	<b>\$109,809,274.20</b>	<b>\$97,257,907.95</b>	<b>12,551,366.25</b>

The visual matrix is updated as you add visual calculations in the Formula bar: new visual calculations are added as columns to the visual matrix. Additionally, you can easily add a running sum of profit by writing:

$$\text{Running sum profit} = \text{RUNNINGSUM}([\text{Profit}])$$

Here is a visual with the two visual calculations we have just created:

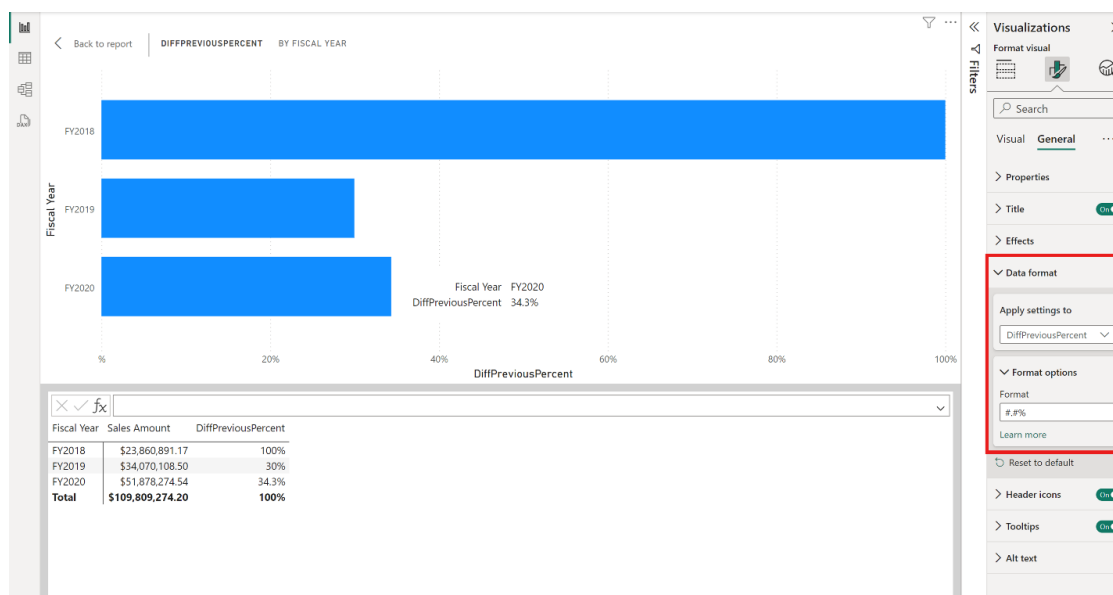


You can use many existing **DAX** functions in visual calculations. Functions specific to visual calculations are also available, such as **RUNNINGSUM**, **PREVIOUS** and **MOVINGAVERAGE**. Using these and other functions, visual calculations are much easier to read, write and maintain than the current **DAX** required. As author of the recent *Financial Modelling in Power BI*, I wholeheartedly agree!

### Visual level format strings (Preview)

Last month, Microsoft introduced the first version of visual level format strings. As a quick reminder, visual level format strings allow you to format fields and measures on a visual, overriding any format string already set in the model. This gives you more flexibility and a quicker way to get the formatting you wanted.

Visual level format strings provide you with more options to configure formatting in Power BI. Originally built for visual calculations, the core ability that visual-level format strings provide is the ability to format visual calculations. Since visual calculations are not in the model, you could not format them, unless you were using them in data labels or in specific parts of the new card and new slicer visuals. With visual level format strings, you can.



However, visual level format strings are useful even without using visual calculations. With the introduction of visual-level format strings, Power BI now has three [3] levels for format strings:

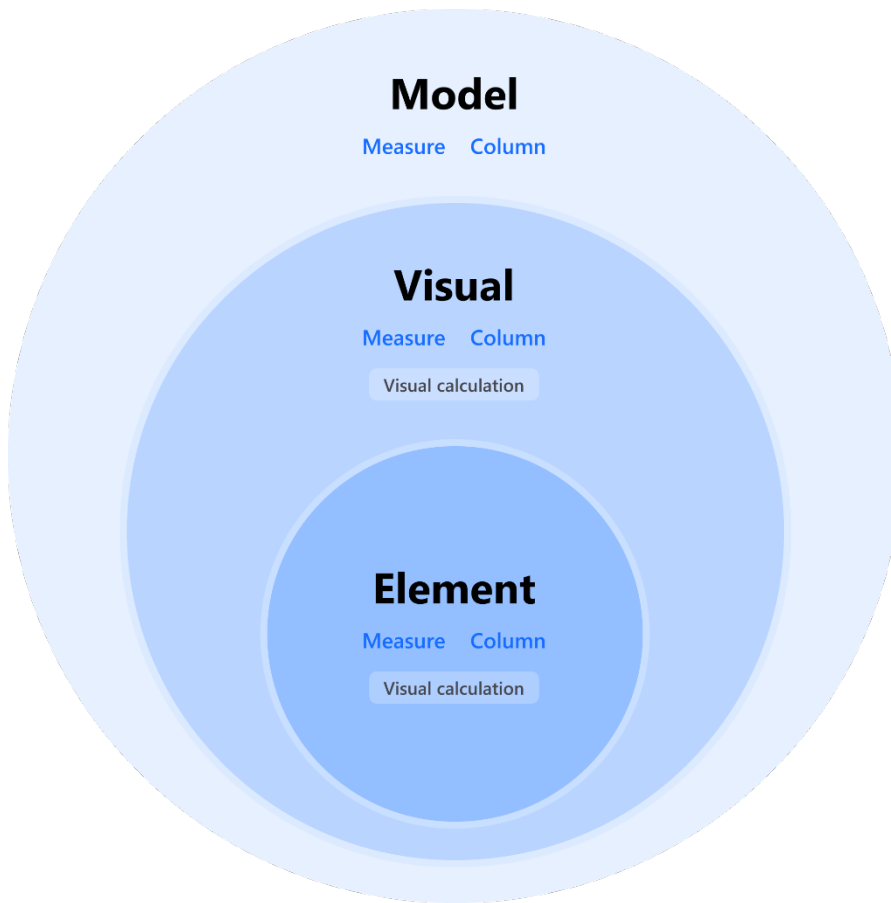
1. **Model.** You can set a format string for columns and measures in the model. Anywhere you use that column or measure the format string will be applied, unless it's overridden by a visual or element level format string
2. **Visual.** This is the update. You may set format strings on any column, measure or visual calculation that is on your visual, even if they already had a format string. In that case, the model level format string will be overridden, and the visual level format string is used
3. **Element.** You can set a format string for data labels and for specific elements of the new card and the new slicer visuals. This level will be expanded to include much more in the future. Any format string you set here will override the format string set on the visual and model level.

These levels are hierarchical, with the model level being the lowest level and the element level the highest. A format string defined on a column, measure or visual calculation on a higher-level override what was defined on a lower level.

Since visual calculations are not in the model, they cannot have a format string set on the model level but can on the visual or element level. Measures and columns can have format strings on all three levels:

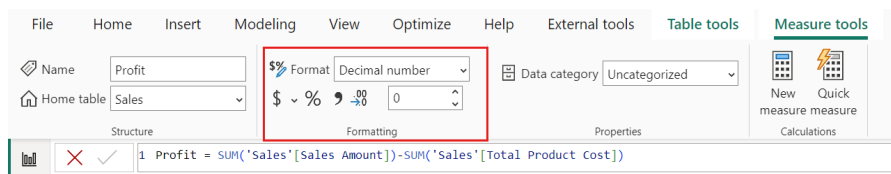
Level	Impacts	Available for	
		Measures / columns	Visual calculations
Element	Selected element of the selected visual	X	X
Visual	Selected visual	X	X
Model	All visuals / pages / reports on same model	X	

The image below summarizes this and shows that higher level format strings override lower-level format strings:

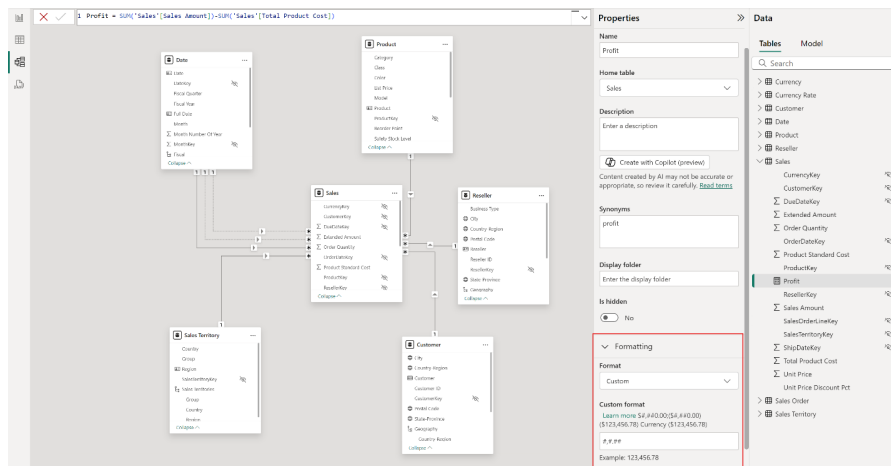


Let's look at an example using a measure.

Assume we have a **Profit** measure in our model, which is set to a decimal number format. To do this, you might have set the formatting for this measure using the Ribbon:



Alternatively, you could have made the same selections in the Properties pane for the measure in the Model view or entered the following custom formatting code:



If you put this measure on a visual it now returns a decimal number, as expected:

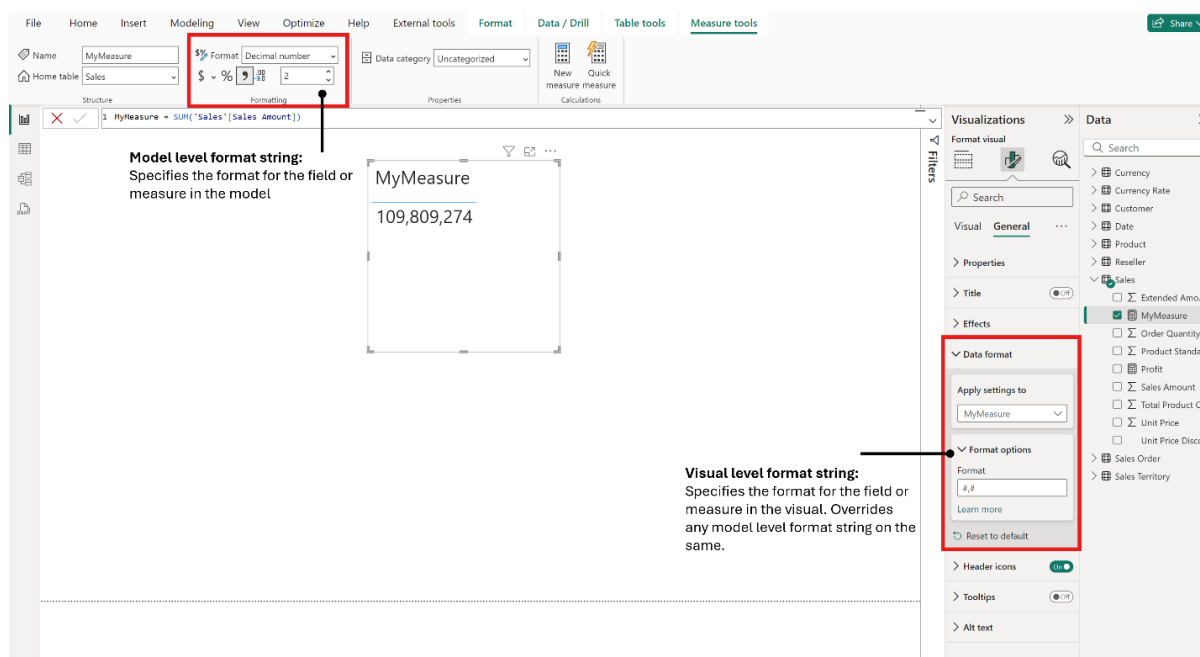
Total is formatted as decimal number in the model

Total

---

109,809,274.2

However, on a particular visual you want that measure to be formatted as a whole number. You can now do that by setting the format code on the visual level by opening the Format pane for that visual and the Data format options found there under General:



Now that same measure shows as a whole number, but just on that visual:

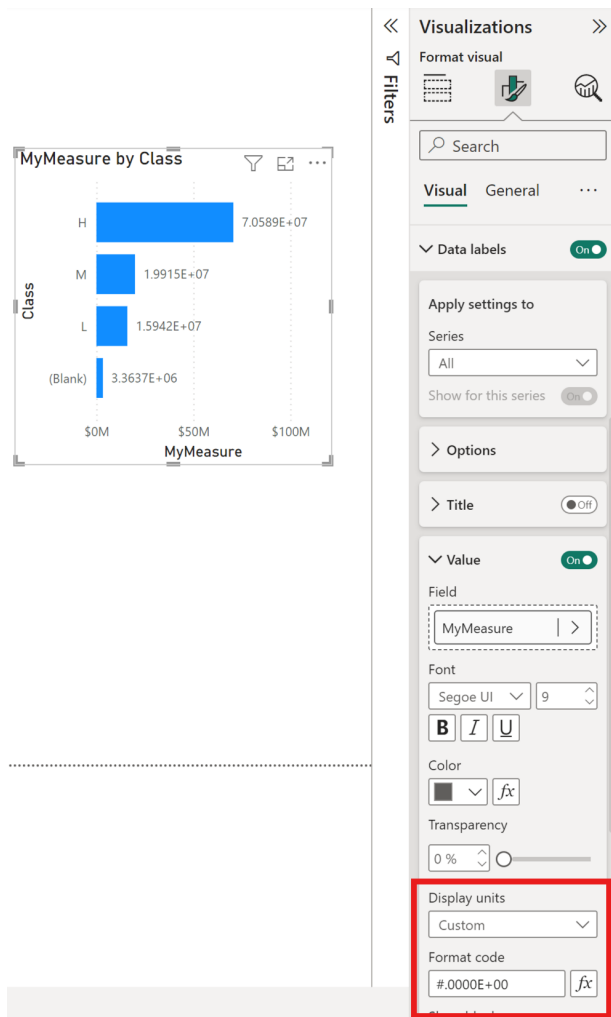
Total is formatted as whole number on this visual

Total

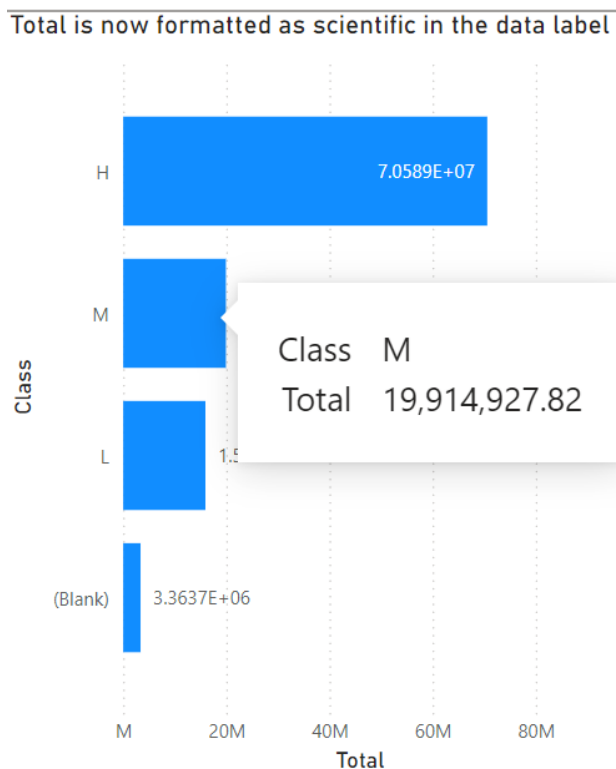
---

109,809,274

Further, you might want to use a scientific notation for that measure but only in the data label on a particular visual. No problem; you set the format code on the data label for that measure:

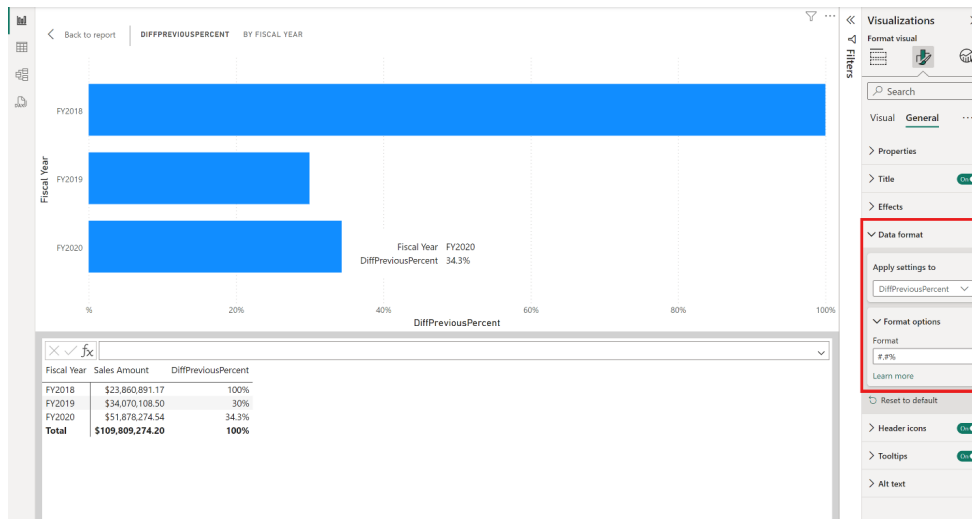


Now, the total shows in scientific notation, but only in the data label and not in other places (such as the ToolTip as shown below). Notice how the element level format is used in the data label but the visual or model level format string is still used for the other elements in the same visual.





For visual calculations the same principle applies but of course without the model level. For example, if you have a visual calculation that returns a percentage, you can now format it as such using the 'Data Format' options in the General on the visual in the Format pane:



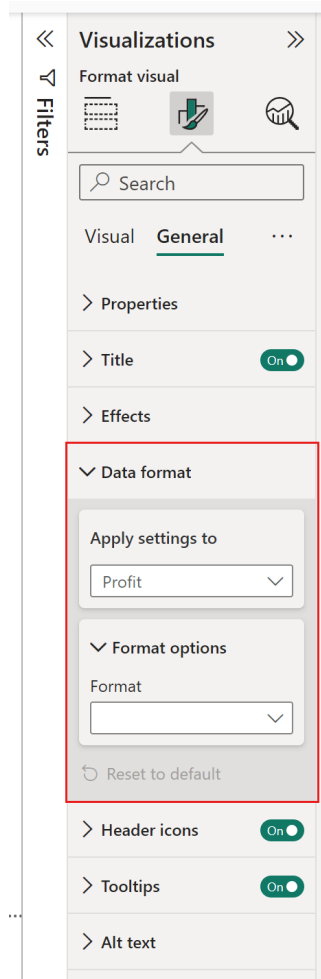
The ability to set visual level format strings makes it much easier to get the exact formatting you need for your visualisations. However, this is only the first iteration of the visual level format strings. Microsoft is planning to add the settings you're used to for the model level format strings to the visual level soon.

Since visual level format strings are introduced as part of the visual calculations Preview, you will need to turn on the visual calculations Preview to use them. To do that, go to **Options and Settings -> Options -> Preview features**. Select 'Visual calculations' and then OK. Visual calculations and visual level format strings are enabled after Power BI Desktop is restarted.

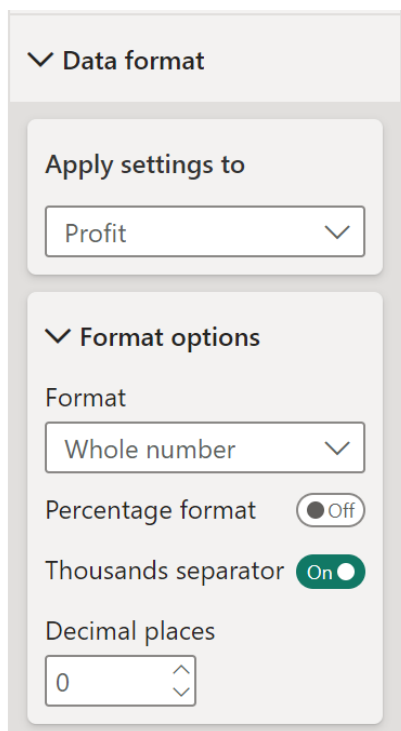
This month, Microsoft is now turning these visual level format strings on by default and expanding them beyond just custom format strings.

Now that visual calculations are on by default visual level format strings are too. If you want, you can turn them off by disabling the **visual calculations Preview in Options and Settings -> Options -> Preview features**.

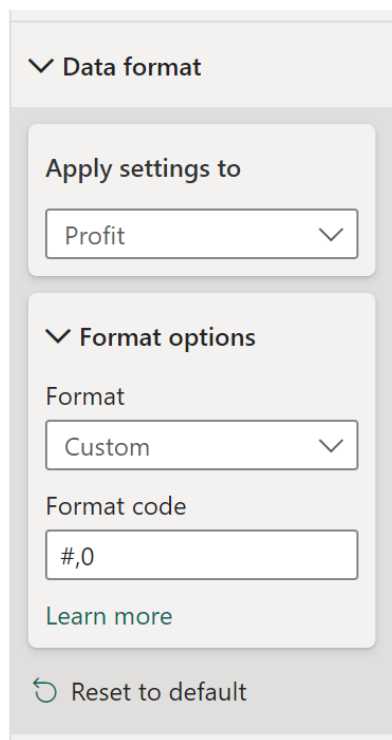
You find visual level format strings in the Format pane for a visual under **General -> Data format**:



Last month, you could only enter a custom format string here. This month, Power BI is aligning the formatting options in the Ribbon, so your visual level formatting is even more powerful and easier to perform. This includes setting the format, such as Text, Currency and Whole number and, for numerical formats, options like thousands separator and number of decimals:



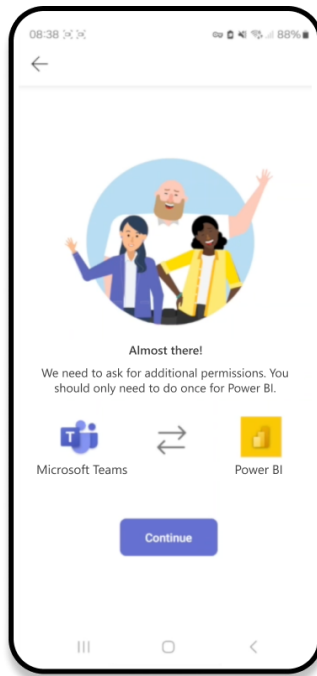
In the Properties pane in the Model view, if you want to enter a custom format string, you can do so by selecting 'Custom':



### **Power BI in Teams: now supporting Multi Factor Authentication**

Organisations usually configure Multi Factor Authentication (MFA) to protect digital assets by forcing users to provide two or more verification factors to gain access to a website or application. This method enhances the protection of the organisation's data by preventing unauthorised access that could occur if someone were to obtain a single password.

Many of us use Teams in our daily work. Therefore, Microsoft has improved how the authentication in Power BI in Teams is handled where MFA is configured. From now on, if the account is configured with Multi Factor Authentication (MFA) the user will need to authenticate before launching the Power BI app from Teams, using a new authentication screen.



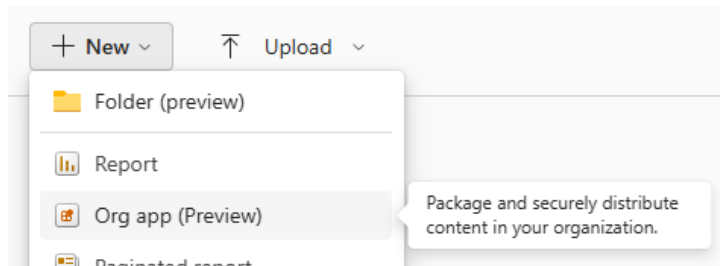
Great. Even more ways not to be able to access a Teams meeting now. 😊

**Multiple organisation applications per workspace with custom colours available (Preview)**

When it's time to distribute reporting to a breadth of teams, you need a secure and easy to use experience that's uniquely suited to each team. That's now possible with Power BI workspace applications revised for Fabric as a new item type: organisation (org) apps: the public Preview of org apps as items are coming soon. Org apps as items can be created in

workspaces with Fabric capacity or trial (additional license and capacity will be supported in the future).

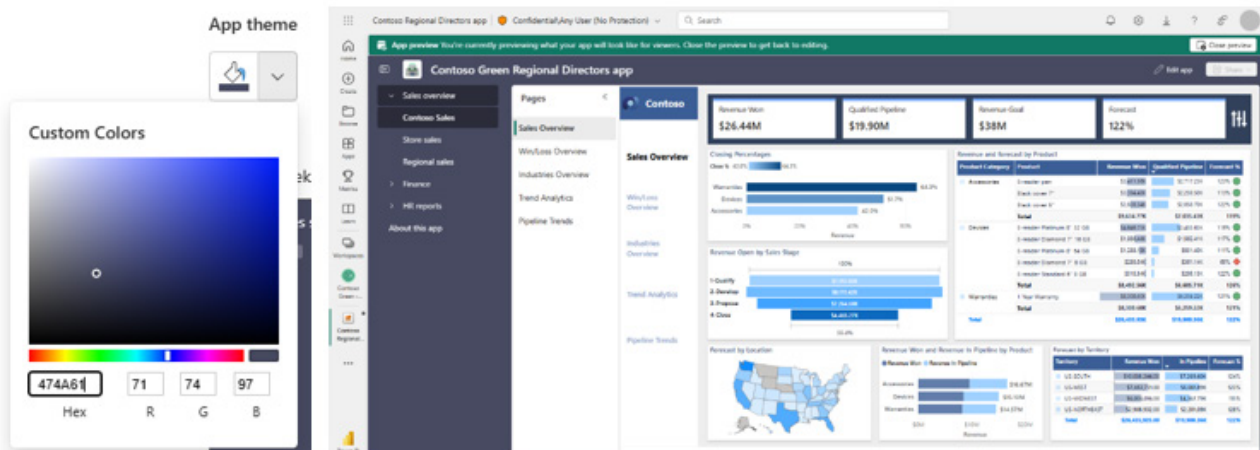
You may create an org app like you would any other item:



With org apps as items, in your workspace you can create multiple distinct org apps tailored to each unique audience you serve. Each org app will have its own unique set of content, navigation, custom theme colour and name. You may manage org apps the way you would any other item type (from creating a new org app, to managing access or

sharing an org app): what you find familiar about managing other items, like reports, will be familiar with org apps as items.

You may select any colour for your org app theme and preview your org app before saving changes:



### Subscribe to reports on the last day of the month

With this enhancement, you can subscribe to standard and dynamic per recipient reports on the last day of the month. You no longer need to manage leap years or figure out how to make things work on a 30-day

and / or 31-day basis. When you set up the monthly subscription, select the 'last day of month' option.

### Subscriptions

Keep track of your data by subscribing to this report.

[Manage all](#)

---

Executive Overview ✎ 🗑️ 🔴

⚠️ Unsaved changes

Subscription name \*

Recipients \*

Attach full report

Send to \*

Scheduled date and time

Start date \*  End date

Repeat \*

Every month on day(s) \*

Last day of month

Scheduled time \*

Time zone \*

Emails will be sent on the last day of every month, starting Tuesday, July 23, 2024 at 09:00 AM (UTC-08:00) Pacific Time (US and Canada)

For the dynamic per recipient subscription, choose the 'last day of month' option as part of the Schedule setup.

### Introducing Fabric Metrics Layer: metric management in Fabric (Preview)

Metrics sets is now in public Preview. It is a transformative new feature designed to redefine how organisations manage and consume metrics. The Fabric Metric Layer's home base in Power BI brings powerful capabilities to streamline metric management, ensure consistency and foster trust in data across your organisation.

The Fabric Metrics Layer is aimed at helping organisations define, discover and reuse trusted metrics simply. Trusted creators within an organisation can develop standardised metrics that incorporate essential business

logic, ensuring consistency across the organisation. Creators may arrange these metrics into collections, promote and certify them, and make them easily discoverable for end users and other creators. These endorsed and promoted metrics can then be used to build reports, improving data quality across the organisation, and can also be reused in other Fabric solutions, such as Notebooks. Consumers may adopt these metrics to gain insights into their business questions from reports or directly from the Metrics hub, which features visuals and Copilot insights.

The Power BI Metrics sets in the Fabric Metrics Layer address three [3] main problems:

1. **Data quality across an organisation** by establishing metrics (rooted in measures) that can be reused eliminating the need for authors to rebuild duplicative measures in reports
2. **Enabling self-service** by establishing "mini" models in the metric set details page that allow business users to Explore the data they are interested in by opening the metric and its associated, curated dimensions to add filters, slicers and answer their own questions
3. **Eliminating siloed data** by allowing other Fabric artifacts to leverage metrics defined in the Fabric metrics layer, for example Notebooks. Data science notebooks can connect to a metric defined in a PBI semantic model and use the data outside of Power BI itself. More Fabric artifacts will be available as Microsoft continues to roll out features.

Metrics sets provide a solution by enabling users to manage and reuse key metrics across the organisation, ensuring a single source of truth and improving trust in the metrics being used. Key features include:

- **Curated collection of metrics:** metric sets will serve as a collection of measure pointers to source semantic models and include key dimensions so end users and authors alike can unambiguously understand how a metric should be grouped or used
- **Rich consumption experiences:** users can explore and consume metrics from the metric set itself, allowing for deep insights and understanding. Copilot summaries and multiple visuals will be available for users to scroll through and go from data to insights in seconds
- **Efficiency:** consumers no longer need to rely on report creators to answer questions or build custom reports for specific needs. Consumers can leverage the Explore dialog to dig deeper into a given metric in an environment where everything in the data pane “just works” because the dimensions have been curated specifically for the metric
- **Discoverability and Reuse:**
  - **Consumers:** metrics are discoverable via search, and metric sets can be promoted, endorsed, certified just like any artifact so that users trust it. Consumers can also leverage the Explore dialog to dig deeper into a given metric in a safe environment where everything in the data pane “just works” because the dimensions have been curated specifically for the metric
  - **Authors:** metrics can be discovered and reused during report creation, using the source semantic model upon saving an exploration or report
- **Integration with Copilot experiences:** integration with Copilot will enhance the user experience by providing AI-driven insights and recommendations.

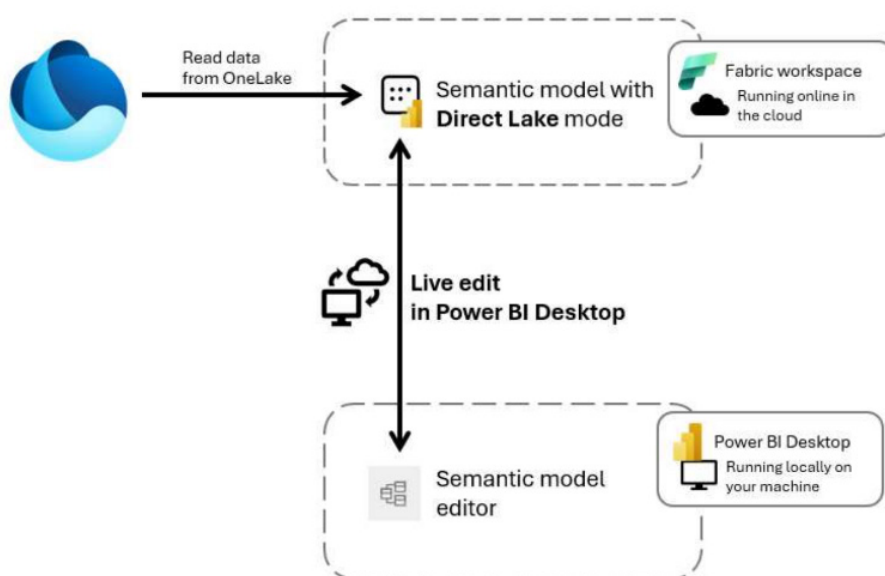
Furthermore, you may be familiar with scorecard metrics: well, these are not going anywhere. Scorecard metrics will be renamed to their original name of Goals, and the functionality will remain. Scorecards are a great downstream use for the new Metrics Layer – simply connect a scorecard goal to a metric that is visualised in a report, and that metric not only standardised across an organisation, but viewable in one pane

of glass for leadership to track with a target, due date and to get progress updates on along with their other goals.

Metrics sets mark a significant step forward in metric management within Fabric, addressing long-standing challenges and introducing powerful new capabilities. Watch this space.

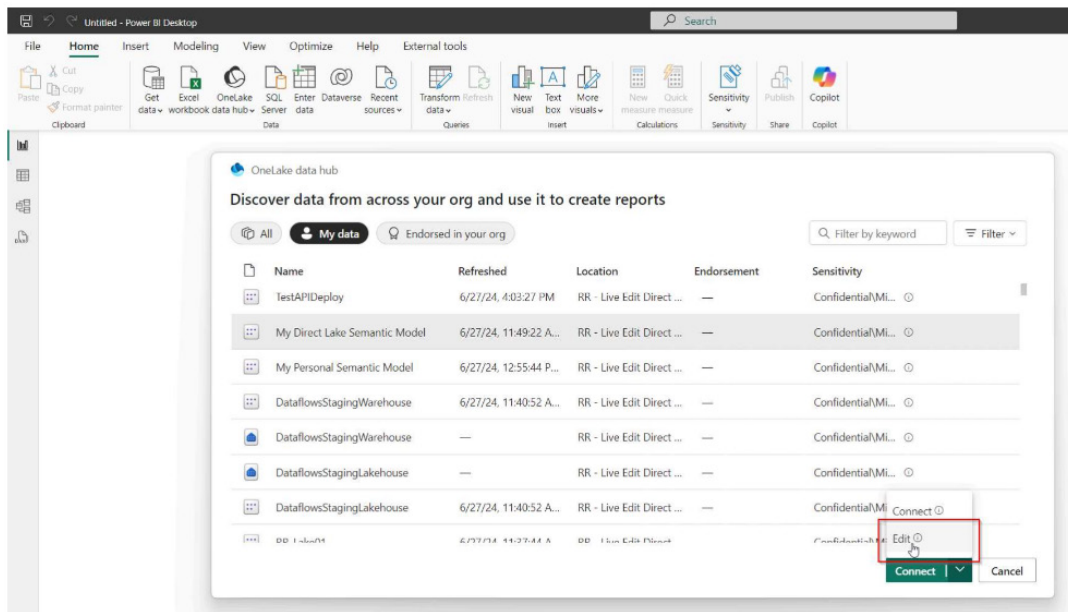
#### **Live edit of semantic models in Direct Lake mode with Power BI Desktop**

Live editing of a semantic model in Direct Lake mode is now available in Power BI Desktop. This capability will allow you to live edit semantic models in Direct Lake mode with Power BI Desktop, directly accessing the OneLake data, and leveraging the Power BI Analysis Services engine in a Fabric workspace instead of your local machine.



With ‘live edit’, you can now make changes to your semantic models in Direct Lake mode with Power BI Desktop. Every modification is applied to the semantic model in the workspace, ensuring a seamless and efficient workflow. This feature is designed to enhance your data modelling experience by providing familiar experience of Power BI Desktop with the option to export to Power BI Project (PBIP) to support professional enterprise development workflows.

To get started, enable the ‘live edit’ feature in the Power BI Desktop preview options menu, select your semantic model in Direct Lake mode from the OneLake data hub, and choose Edit from the Connect button drop-down. Whether you’re creating new measures, adding calculation groups, creating relationships between tables or running DAX queries, Direct Lake mode streamlines the process as data is accessed directly from the OneLake.



### Q&A now supports semantic models with OLS

The Q&A visuals allow users to ask natural language questions about their semantic model. Previously, models that had OLS applied were not supported for use with Q&A. Microsoft has now removed this limitation; Q&A now supports semantic models with OLS.

### Connected tables in Excel are now available in Semi-Annual channel

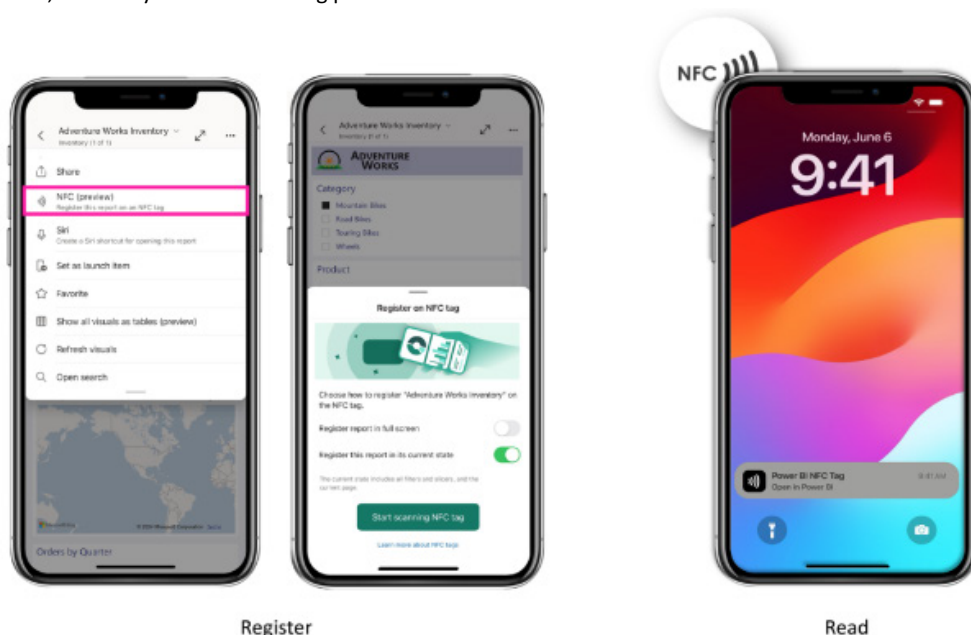
The connected table feature in the Power BI Datasets Add-in in Excel is now available on the Semi-Annual channel. With this capability, you can easily add Power BI data to your Excel workbooks by creating a connected table directly within Excel. This streamlines the workflow of adding data and is friendlier to more Excel users.

### NFC tag support in Power BI Mobile

Imagine being able to access your important data just by tapping your device on a small tag. That's the power of NFC (Near Field Communication) tags. Now, Microsoft has introduced NFC Tag Support in the Power BI Mobile app. This feature allows you to register and read Power BI items such as reports, scorecards, dashboards or even a set of items like an app or workspace on NFC tags, directly from the app. It creates a seamless connection between your data and the physical world.

For example, a retail manager could quickly access inventory data by tapping their phone on an NFC tag placed on a storage shelf. This feature is especially useful for frontline workers who need quick access to data while managing retail floors, inventory or manufacturing processes.

Getting started is easy: open the Power BI Mobile app, navigate to the item you want to register, press the Register to NFC button in the three dots menu, and follow the prompts to link it to an NFC tag. Once you've registered your desired item to the tag, anyone with the Power BI Mobile app can get to the item simply by tapping the tag with their device, whether the app is open or not. If the user doesn't have permission to access the item, they'll be taken to the request access flow. NFC tags are flexible, durable and reliable, and can easily be reused.



Register



Read

## KPI by Powerviz

KPI by Powerviz is custom visual for Power BI that allows users to visualise and create advanced Key Performance Indicators (KPIs). Key features include:

- 100+ Prebuilt KPI templates within visual and option to create your own templates
- 16 layers and 40+ chart variations to create infographic designs Rich customisation, formatting options and colour styles
- Ability to create KPI objects in layers, combining charts, metrics and icons
- Categorical: compare values across categories
- Comparison: analyse differences between values
- Composition: show parts of a whole
- Progression: display trends over time
- Actual vs Target: compare actual against targets
- Formatting: configure the Ranking, Sorting, Axis, Number-Formatting, ToolTip, Gridlines, Data Labels and Series Labels for visuals
- IBCS Theme Support: includes deviation bars, series labels and consistent colour scheme(s)
- Small Multiples: support for all chart types – Fixed / Fluid with change chart feature
- Other features include multi-categories comparison, Highlight values and Layer Flexibility.

Business Use Cases include assessing sales performance, financial health and customer satisfaction.



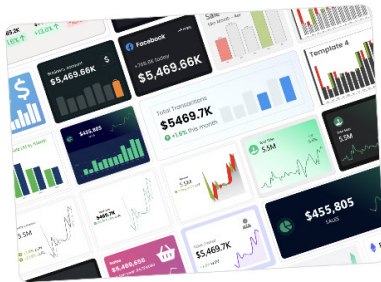
### INTRODUCING KPI BY POWERVIZ

Most advanced KPI Visual with 16 Chart Layers, IBCS Theme Support, Conditional Formatting & On-object Interaction.

Don't settle for boring looking KPI Cards. Make them look stunning.

**100+ Prebuilt Templates**

KPI offers 100+ pre-built template within the visual or create your own templates.



**Build Any KPI You Can Imagine With KPI By Powerviz**

Advanced Text | Bar | Line

Lollipop | Pie | Waterfall

Liquid Gauge | Icon | Bullet Chart

Linear Gauge | JSON & 12 More

Alignment

Shadow: LOW, MEDIUM, HIGH, CUSTOM

Borders

Group | Move Tool | Background FX

**Upto 16 Layers And 40+ Chart Variations**      **Customization Options**

#### Text Editor

The advanced text editor lets you add Value FX, Text FX, Icon FX, and more —without any hassle.

Poppins

B I U { } X<sub>2</sub>

Value fx

Icon fx | Text fx | A fx | fx | Badge fx

Conditional Icons

Value fx

Badge

T Text fx

B I U

Links

Background

Bullet Points

3 Takeaway:  
• High Churn Rate  
• New Leads  
• Better Deals

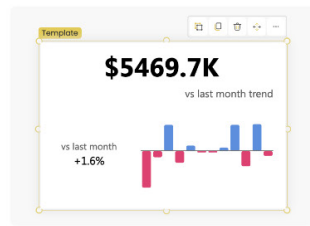
Note: It's worth noting that the current price is at an all-time High, as indicated by the price floor [here](#).

Author: Powerviz

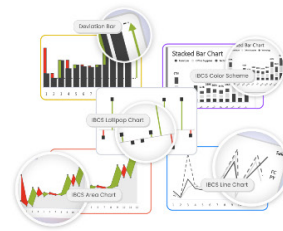
contact@sumproduct.com | www.sumproduct.com | +61 3 9020 2071

### INTRODUCING KPI BY POWERVIZ

Advanced Text Editor, Highlighting & Styling Options, User Friendly UI, Multi Categories/Measures & More.



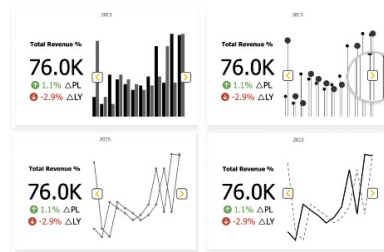
On-Object Interaction



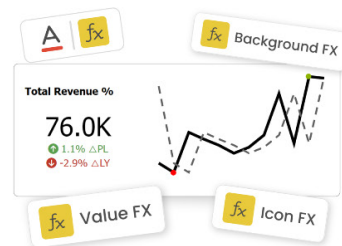
IBCS Support



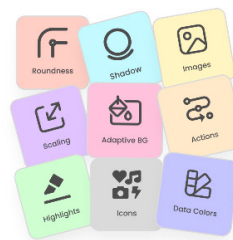
Small Multiples - Fixed And Fluid



Small Multiples - Change Chart Type



Conditional Formatting



Other Features

Unlock A World With Endless Possibilities With The **KPI By Powerviz**

#### Drill Down Combo Bar PRO by ZoomCharts

With Drill Down Combo Bar PRO, you can mix and match Bar, Area and Line charts in one visual. Whether you want to show just one series or 25, Combo Bar PRO will do it all in a way that combines user-friendliness with data density. You can then use multiple category columns to create a drill down hierarchy, allowing the user to drill down with just a click.

Main features include:

- Visualise up to 25 series (as Bars, Lines or Areas)
- Drill down and cross-filtering
- Intuitive on-chart interactions (select, drill down, pan, invert selection, and more)
- 'Legend' field support
- Stacks and clusters
- Static and dynamic threshold lines or areas
- Rich customization (axes, series, legends, labels, ToolTips and more)
- Conditional formatting.

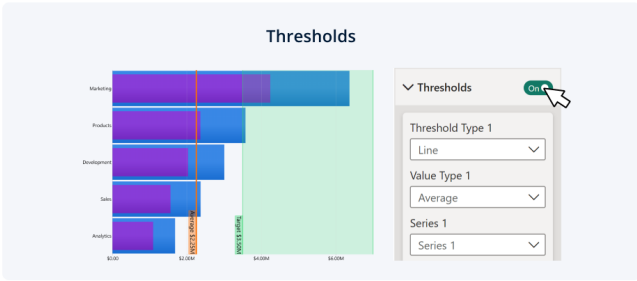
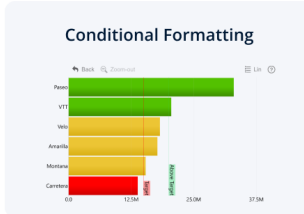
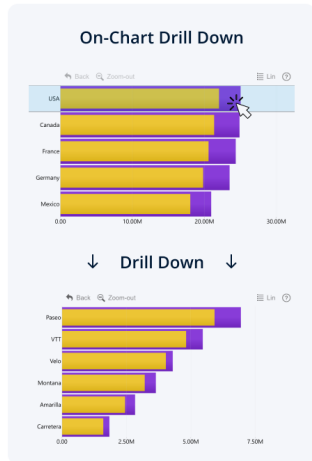
With Combo Bar PRO, you can make your reports faster, more intuitive and more insightful. The visual will seamlessly integrate with other visuals and cross-filter data across the entire report, providing focused insights from multiple dimensions.





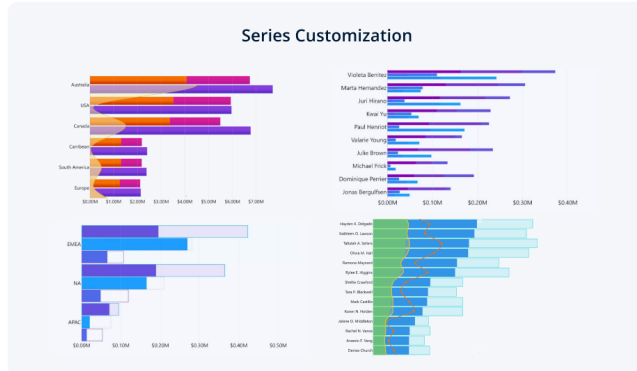
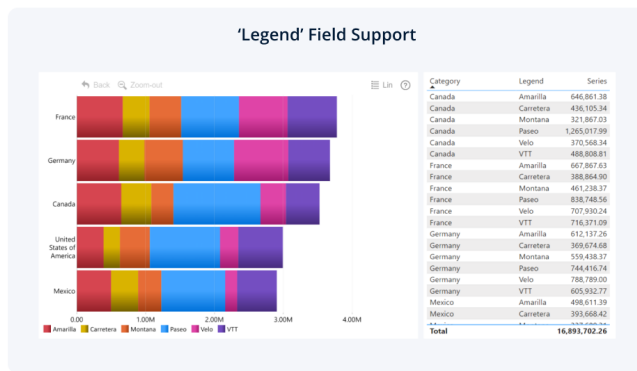
ONE VISUAL. ENDLESS POSSIBILITIES.

Create stunning charts that are easy to explore and make your entire report more interactive.



RICH CUSTOMIZATION

Explore more than 300 customization settings and create the perfect chart for your report.



**BI Pixie by DataChant: measure the engagement of your BI audience**

BI Pixie is an innovative solution that tracks the usage and engagement of your Power BI audience. With BI Pixie, you can gain insight into user adoption, attrition and time-intelligent engagement metrics that will help you understand the effectiveness of your BI portfolio, critical reports and measure the return on your Power BI investments.



BI Pixie captures detailed user actions, providing a comprehensive view of engagement. It goes beyond basic page views and can help you differentiate between passive and active users by tracking these activities:

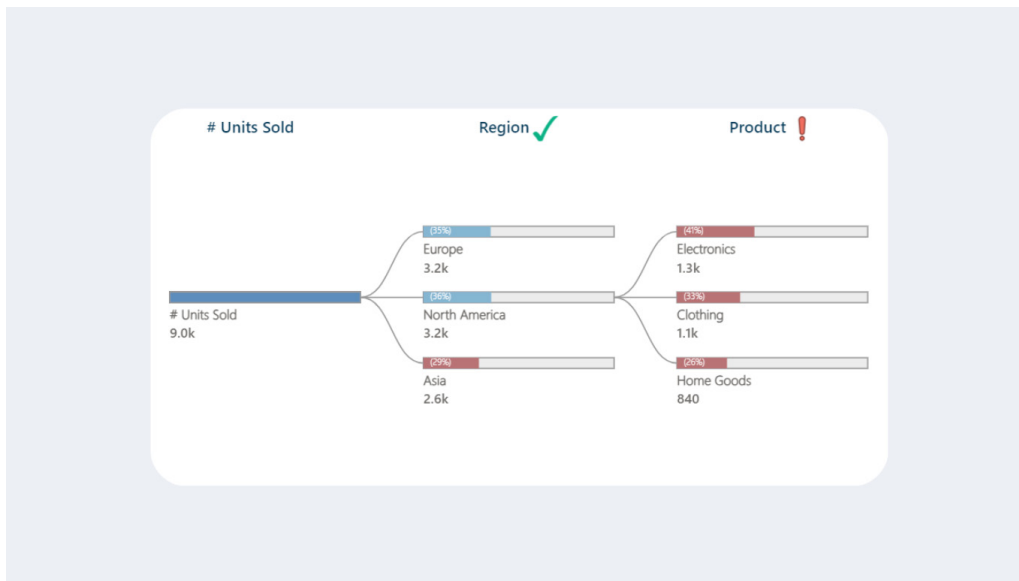
- Interactions in visuals
- Slicing and filtering
- Clicks on bookmarks
- Clicks on links in tables, buttons, shapes and images
- ToolTip views
- Drill-throughs.

It's designed to ensure your data stays in your organisation and scale to tens of thousands of reports and users, whereby BI Pixie can be deployed in your Azure tenant.

**Decomposition Tree by JTA**

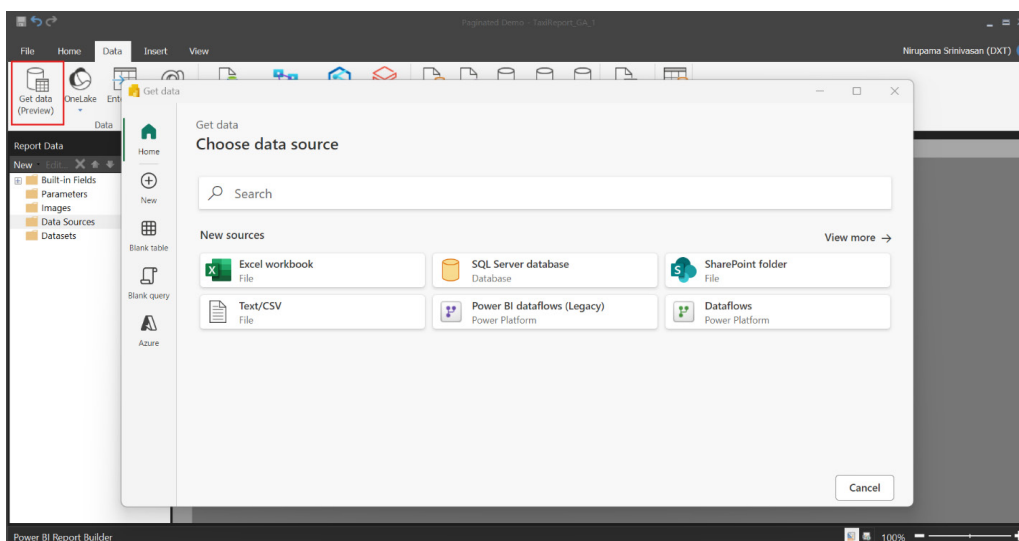
The Decomposition Tree by JTA is a versatile tool for powerful data visualisation. This horizontal decomposition tree allows you to personalise multiple aspects to fit your data needs. Key features include:

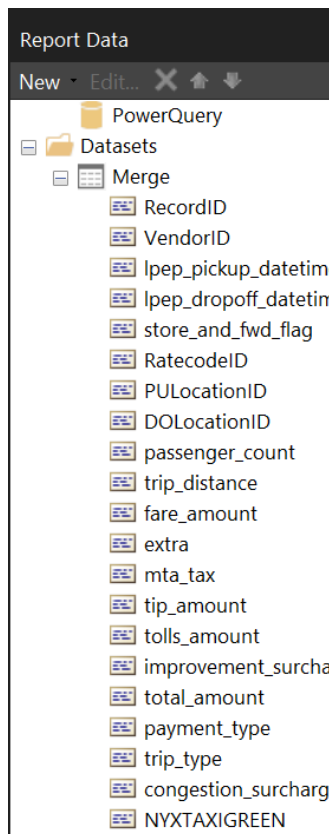
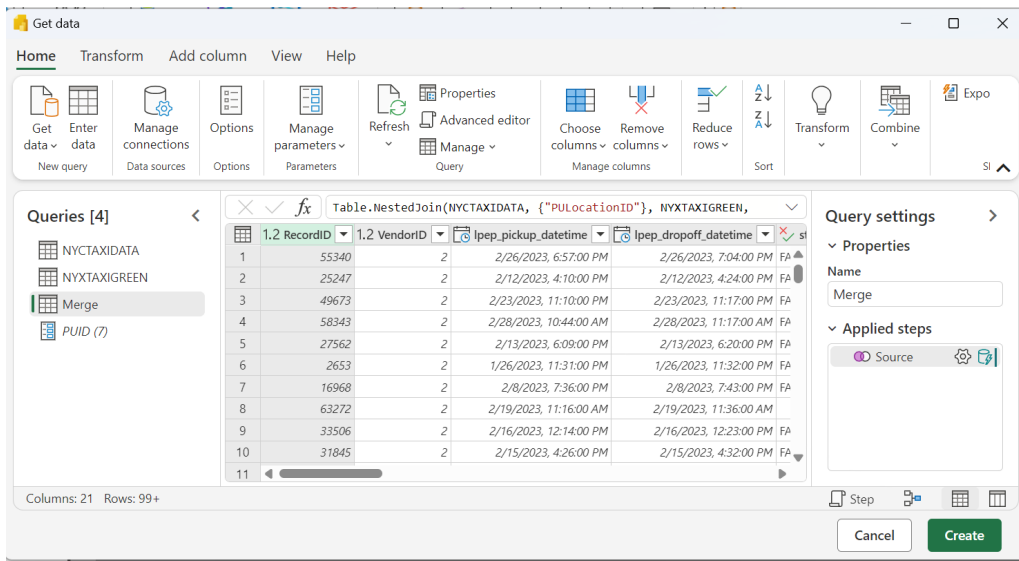
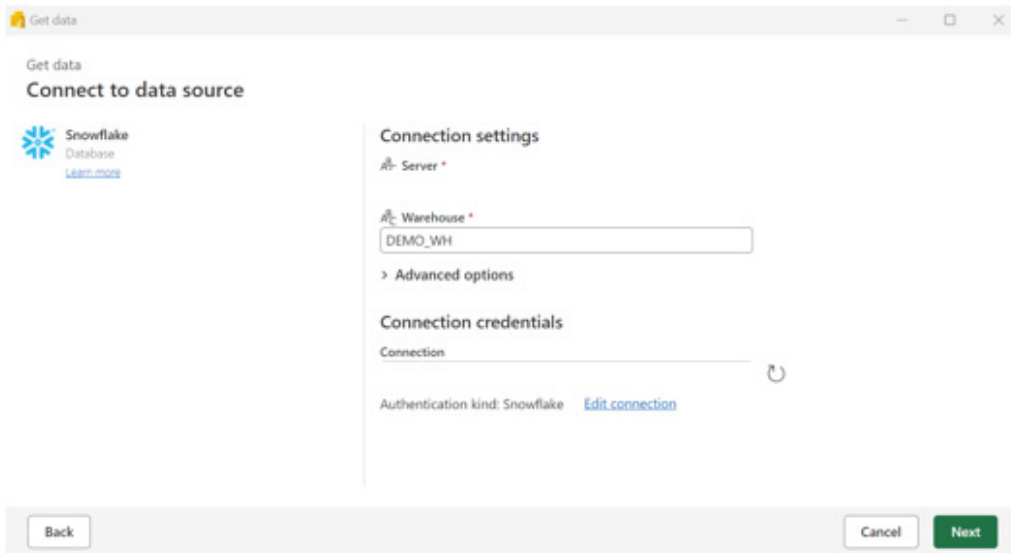
- **Personalise Bars:** adjust the width, height, margins, borders and colours to represent your data
- **Customise text:** modify titles, legends, values and percentages with customisable margins, colours, fonts and alignments
- **Percentage display:** choose to display percentages as a grand total or by category, with the option to show them on the Bars
- **Tree appearance:** tailor your tree's look by adjusting path colours, level distances and display options, including hiding blanks and setting default trees
- **Icon personalisation:** set unique icons for percentages below, above or within specific thresholds to enhance data interpretation
- **Conditional formatting:** easily apply colour-coding to highlight key values
- **Animation control:** enable or disable smooth transitions to fit your visual style
- **ToolTip customisation:** personalise ToolTip titles and content for detailed data insights.

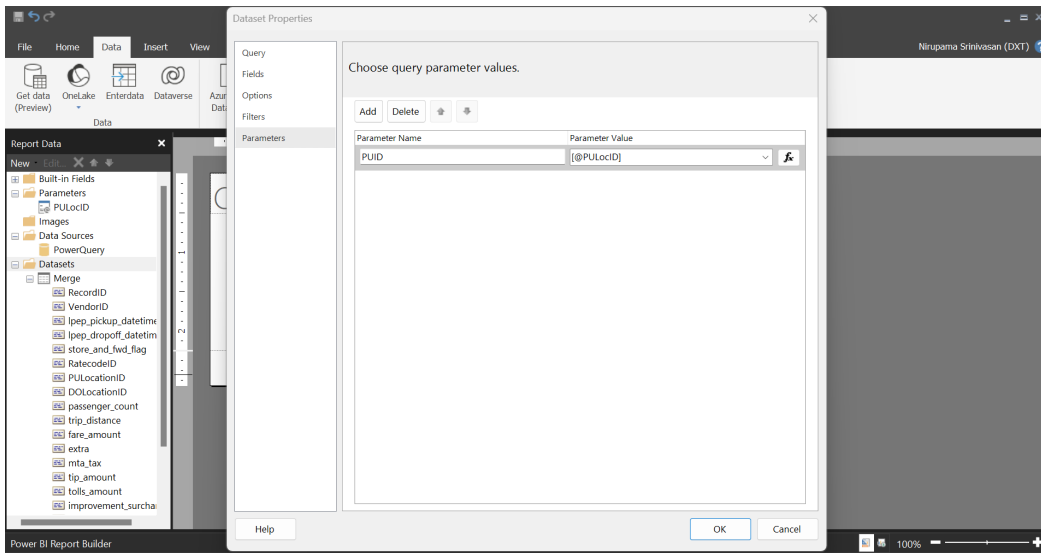


**Announcing the General Availability of the 'Get data' experience in Power BI Report Builder**

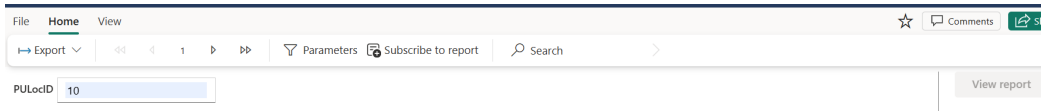
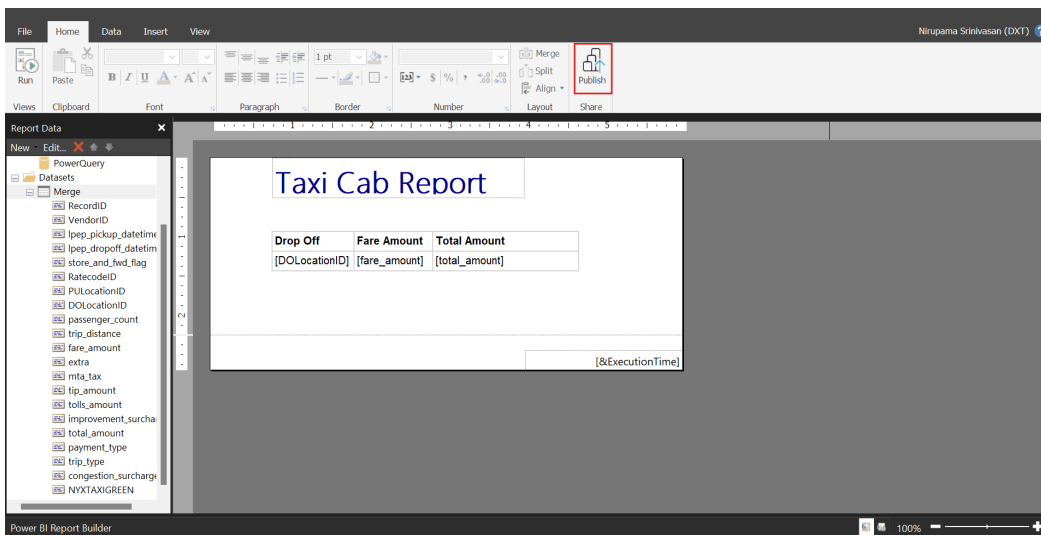
The 'Get data' experience for Power BI Report Builder is now Generally Available, enabling paginated report creators to build great content on top of 100+ data sources. When connecting via Power Query, creators can build performant reports against massive datasets by leveraging parameters to pull in just the data necessary. This feature is now available to Sovereign clouds.







Further, Microsoft has improved the sharing experiences by eliminating the need to separately share the sharable cloud connection (SCC).



## Taxi Cab Report

Drop Off Location ID	Fare Amount	Total Amount
7	7.90	11.28
168	13.50	15.00
41	9.30	14.16
48	28.20	38.10
238	12.80	24.44
92	100.00	121.20
74	9.30	15.96
173	16.64	18.14
134	12.10	16.32
41	7.90	11.90
196	10.70	12.20
41	14.30	16.30

More next month.

# New Features for Excel

This month's updates see Excel reach a new watermark – quite literally! Dynamic watermarking, further improvements to Copilot in Excel and the new regular expression modes for **XLOOKUP** and **XMATCH** allow you to continue to increasingly get more and more out of Excel.

The full list is as follows:

## Excel for Windows, Mac and the web

- Copilot in Excel: create custom charts and PivotTables
- Copilot in Excel: text insights
- Dynamic watermarking (Insiders)

## Excel for Windows and Mac

- New regular expression (regex) modes for **XLOOKUP** and **XMATCH** (Insiders)

## Excel for Windows

- Filtering for comments.

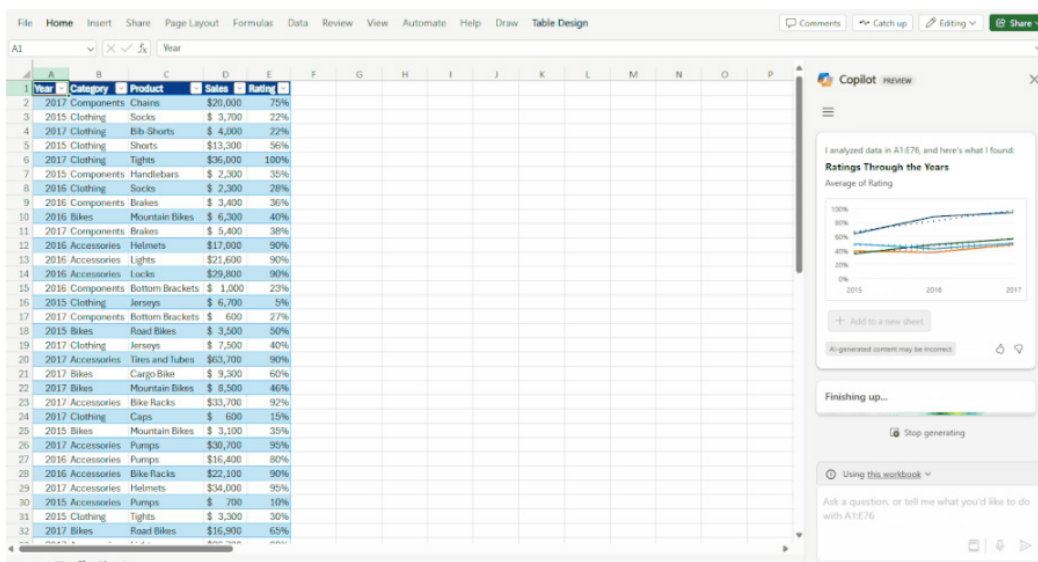
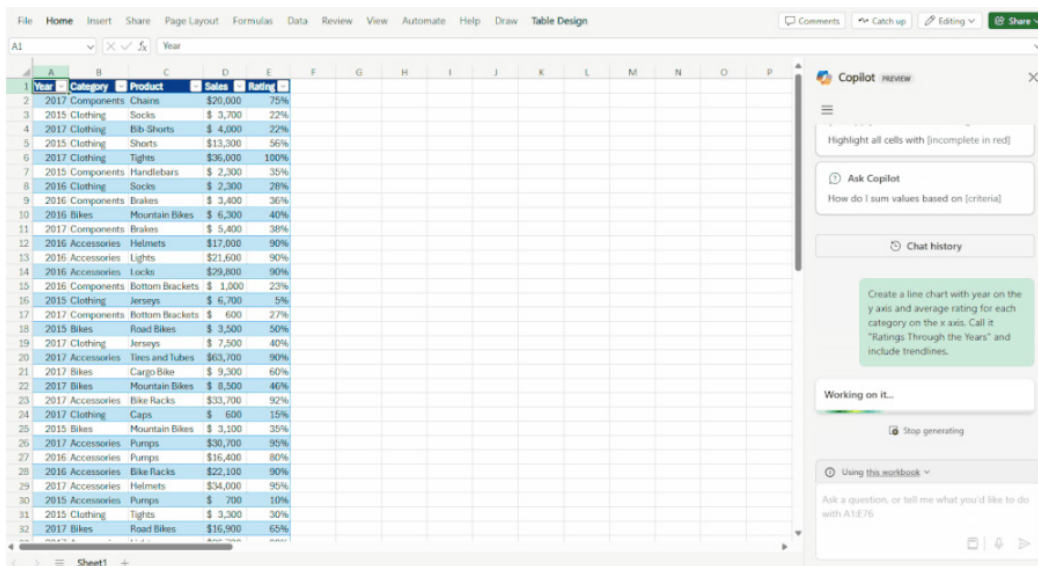
Let's get started.

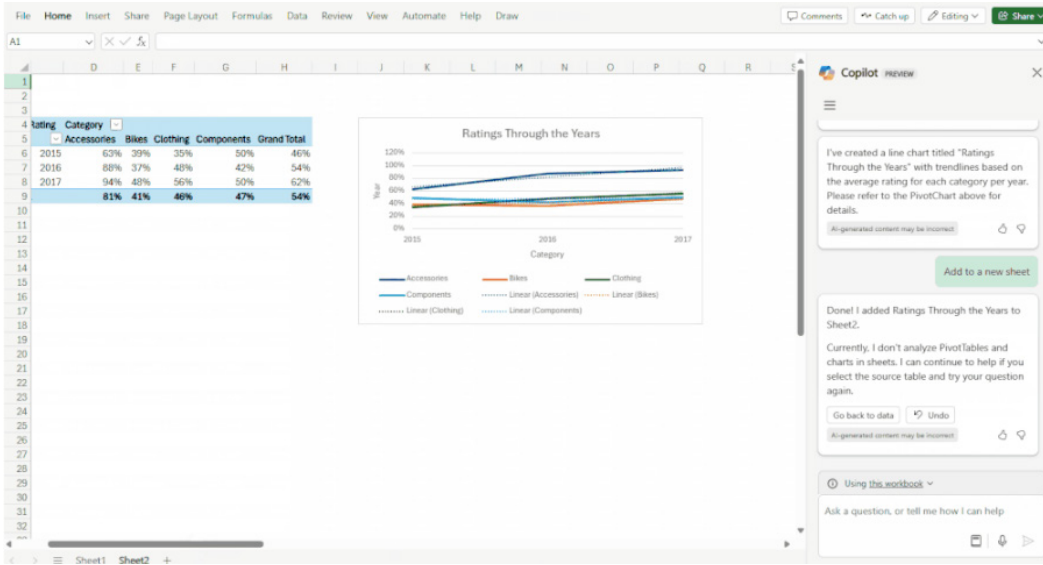
## Copilot in Excel: create custom charts and PivotTables

Now in Excel for Windows, Mac and the web, Copilot can now create PivotCharts and PivotTables according to your specification. Simply prompt Copilot and tell it what kind of chart you want and configure the axes, title, labels, and more. You may also specify the configuration for a PivotTable that you would like to create.

Current chart types that are supported are as follows: Line, Pie, Clustered Bar, Stacked Bar, Hundred Percent Stacked bar, Clustered Column, Stacked Column, Hundred Percent Stacked Column, Donut, Area, Hundred Percent Stacked Area, Histograms and Scatter.

You may create a custom PivotChart as follows, for instance:

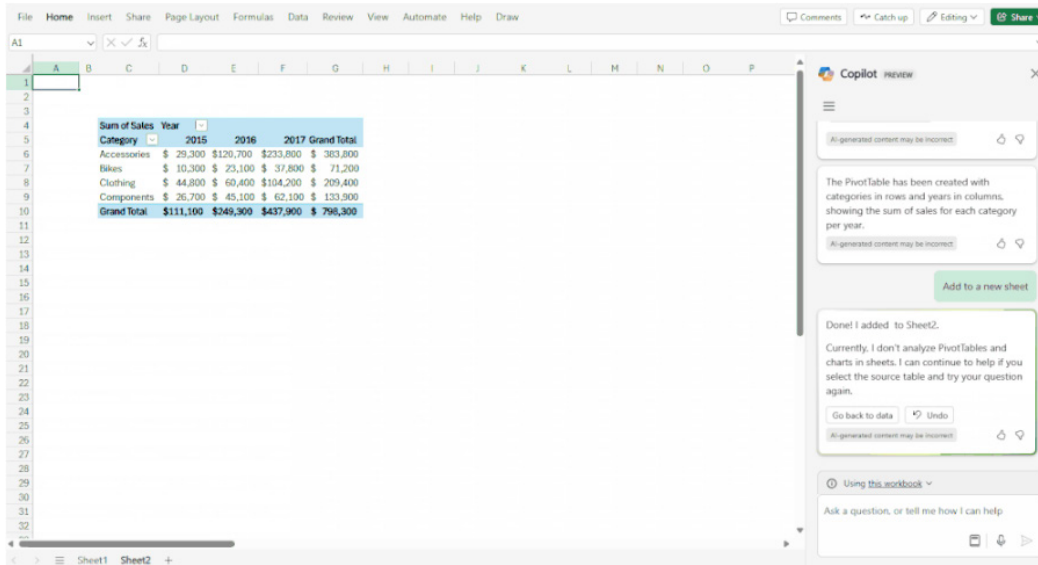




Here is a PivotTable example:

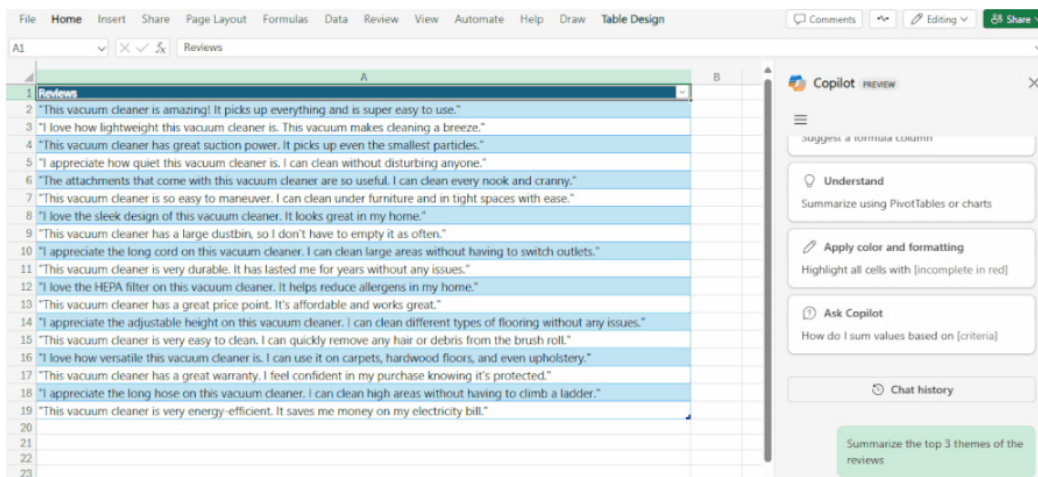
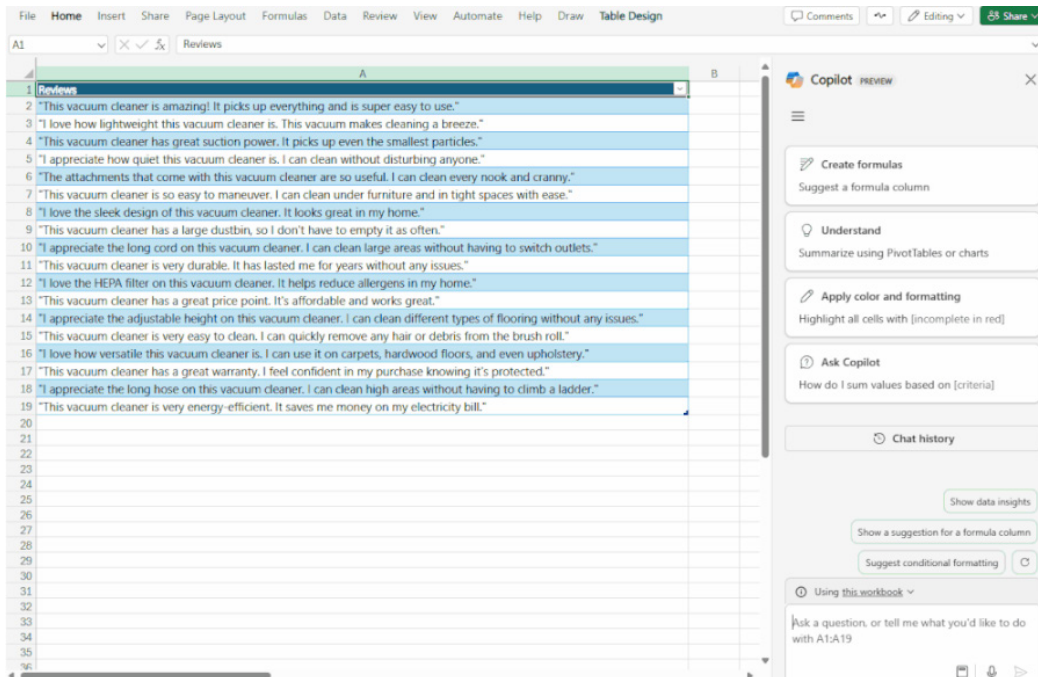
Year	Category	Product	Sales	Rating
2017	Components	Chairs	\$20,000	75%
2015	Clothing	Socks	\$ 3,700	22%
2017	Clothing	Bib-Shorts	\$ 4,000	22%
2015	Clothing	Shorts	\$13,300	56%
2017	Clothing	Tights	\$36,000	100%
2015	Components	Handlebars	\$ 2,300	35%
2016	Clothing	Socks	\$ 2,300	28%
2016	Components	Brakes	\$ 3,400	36%
2016	Bikes	Mountain Bikes	\$ 6,300	40%
2017	Components	Brakes	\$ 5,400	38%
2016	Accessories	Helmets	\$17,000	90%
2016	Accessories	Lights	\$21,600	90%
2016	Accessories	Locks	\$29,800	90%
2016	Components	Bottom Brackets	\$ 1,000	23%
2015	Clothing	Jerseys	\$ 6,700	5%
2017	Components	Bottom Brackets	\$ 600	27%
2015	Bikes	Road Bikes	\$ 3,500	50%
2017	Clothing	Jerseys	\$ 7,500	40%
2017	Accessories	Tires and Tubes	\$63,700	90%
2017	Bikes	Cargo Bikes	\$ 9,300	60%
2017	Bikes	Mountain Bikes	\$ 8,500	46%
2017	Accessories	Bike Racks	\$33,700	92%
2017	Clothing	Caps	\$ 600	15%
2015	Bikes	Mountain Bikes	\$ 3,100	35%
2017	Accessories	Pumps	\$30,700	95%
2016	Accessories	Pumps	\$16,400	80%
2016	Accessories	Bike Racks	\$22,100	90%
2017	Accessories	Helmets	\$34,000	95%
2015	Accessories	Pumps	\$ 700	10%
2015	Clothing	Tights	\$ 3,300	30%
2017	Bikes	Road Bikes	\$16,900	65%

Year	Category	Product	Sales	Rating
2017	Components	Chairs	\$20,000	75%
2015	Clothing	Socks	\$ 3,700	22%
2017	Clothing	Bib-Shorts	\$ 4,000	22%
2015	Clothing	Shorts	\$13,300	56%
2017	Clothing	Tights	\$36,000	100%
2015	Components	Handlebars	\$ 2,300	35%
2016	Clothing	Socks	\$ 2,300	28%
2016	Components	Brakes	\$ 3,400	36%
2016	Bikes	Mountain Bikes	\$ 6,300	40%
2017	Components	Brakes	\$ 5,400	38%
2016	Accessories	Helmets	\$17,000	90%
2016	Accessories	Lights	\$21,600	90%
2016	Accessories	Locks	\$29,800	90%
2016	Components	Bottom Brackets	\$ 1,000	23%
2015	Clothing	Jerseys	\$ 6,700	5%
2017	Components	Bottom Brackets	\$ 600	27%
2015	Bikes	Road Bikes	\$ 3,500	50%
2017	Clothing	Jerseys	\$ 7,500	40%
2017	Accessories	Tires and Tubes	\$63,700	90%
2017	Bikes	Cargo Bikes	\$ 9,300	60%
2017	Bikes	Mountain Bikes	\$ 8,500	46%
2017	Accessories	Bike Racks	\$33,700	92%
2017	Clothing	Caps	\$ 600	15%
2015	Bikes	Mountain Bikes	\$ 3,100	35%
2017	Accessories	Pumps	\$30,700	95%
2016	Accessories	Pumps	\$16,400	80%
2016	Accessories	Bike Racks	\$22,100	90%
2017	Accessories	Helmets	\$34,000	95%
2015	Accessories	Pumps	\$ 700	10%
2015	Clothing	Tights	\$ 3,300	30%
2017	Bikes	Road Bikes	\$16,900	65%

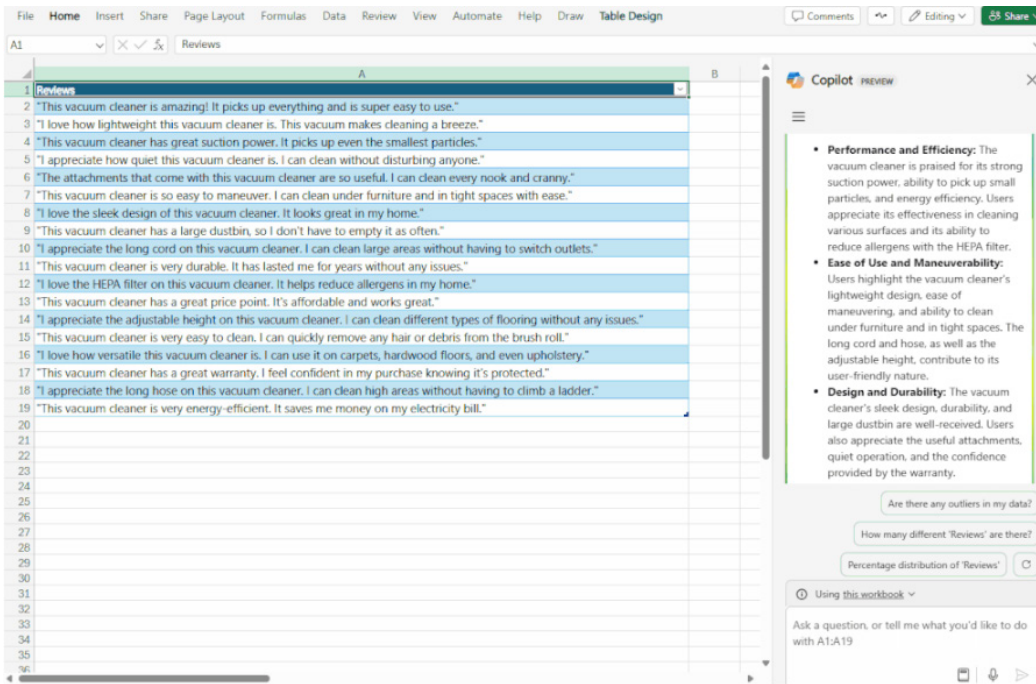


### Copilot in Excel: text insights

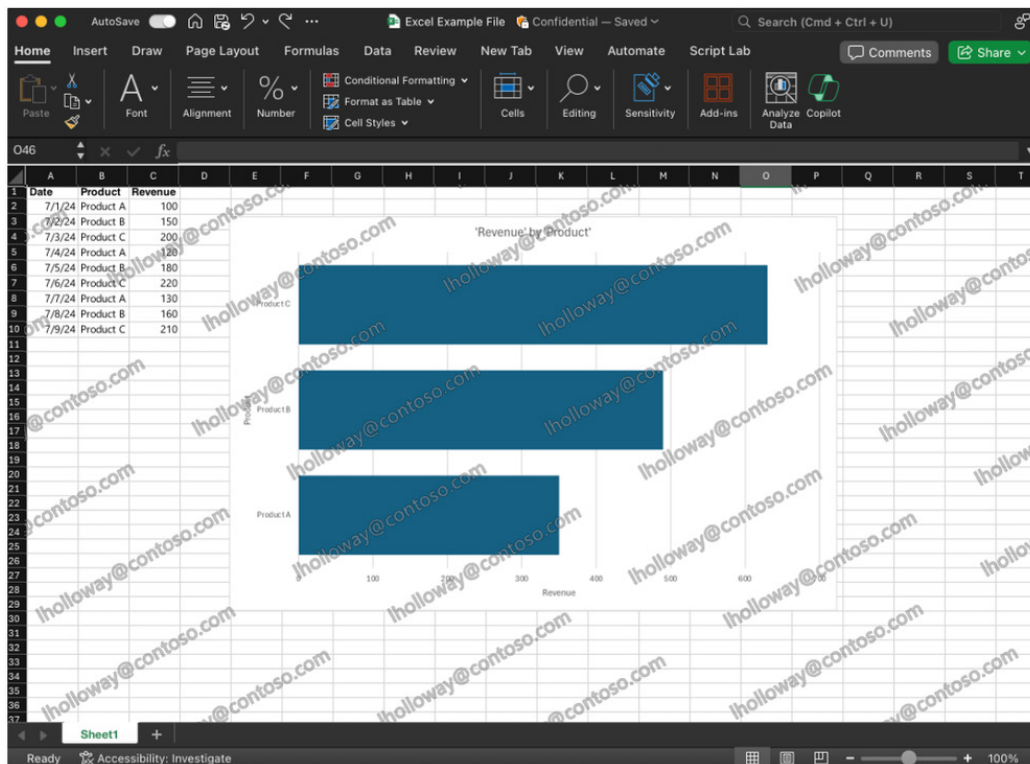
Also now available in Excel for Windows, Mac and the web, you may now summarise textual data like survey results, reviews, comments, and more with Copilot in Excel. Tone, length and content of the summary are all adjustable through your prompt.







### Dynamic watermarking (Insiders)

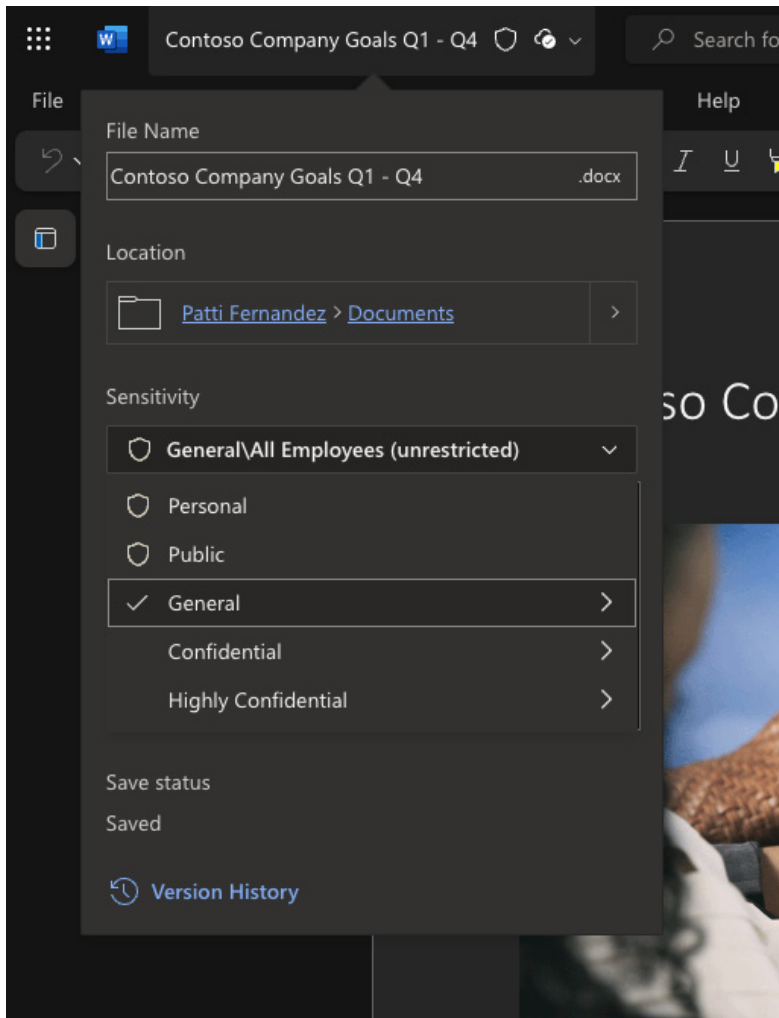


That's not all for Excel for Windows, Mac and the web. If you work with sensitive or confidential documents, you know how vital it is to prevent any leaks of information from these documents. Sensitivity labels from Microsoft Purview Information Protection offer a highly effective way to limit access to sensitive files and prevent people from taking inappropriate actions with them, such as printing a document, while still allowing for full collaboration.

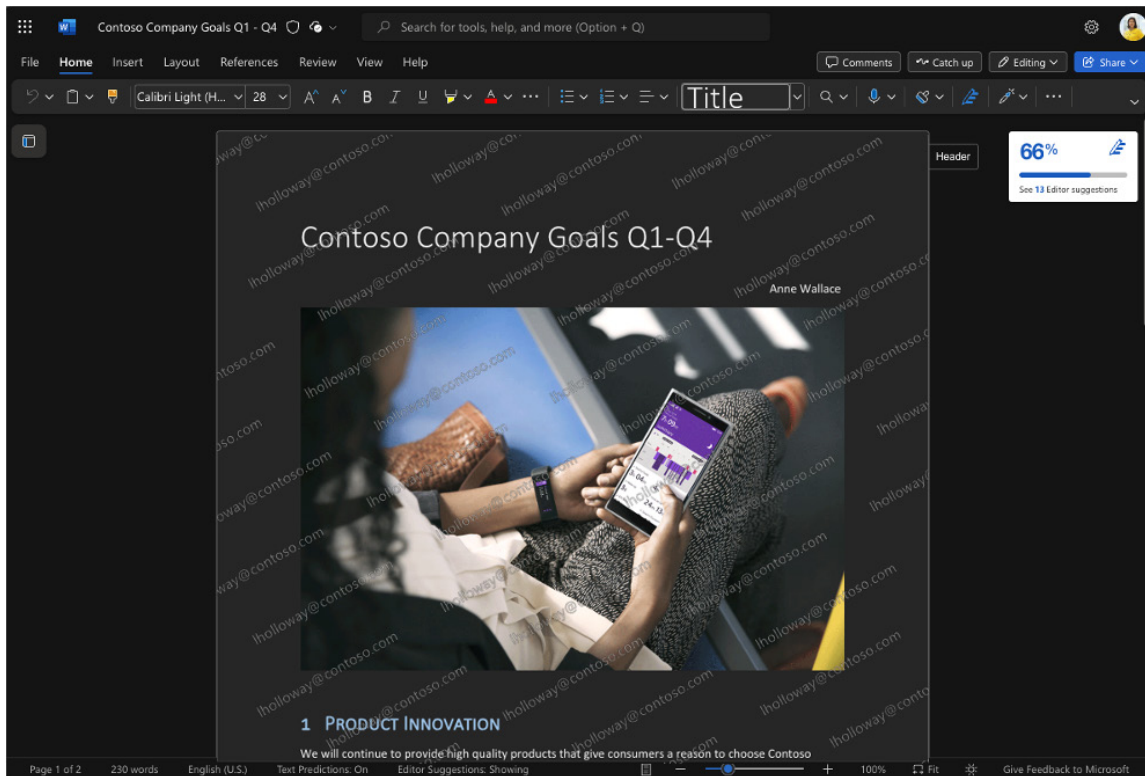
However, it's still possible for someone to take a picture of a sensitive file on their screen or of a presentation being shared either online or

in-person (some forms of screenshots cannot be blocked with existing technology). This loophole presents a simple way to bypass the protection that sensitivity labels place on a document. Dynamic watermarking can be a potent weapon in combatting these kinds of leaks.

To apply a sensitivity label with dynamic watermarking, select the file name in the title bar or click the Sensitivity button on the Ribbon, and then select a label from the dropdown list.



When the label is applied, the watermarks will appear on the screen shortly afterwards.



When you open a file protected with a sensitivity label that has the dynamic watermarking setting enabled, you will see tiled watermarks superimposed across the file content. These watermarks will contain the email address for the account you're using to view the file.

You'll be able to work with your file like normal (viewing, editing, and collaborating) with the watermarks always visible on top of your file content. You won't be able to remove these watermarks unless you have IRM usage rights that allow you to change the file's label to one that doesn't have dynamic watermarking.

If you attempt to open a file labelled with dynamic watermarks in a version of Microsoft 365 that does not support the feature, you will get an "access denied" message (unless you are the owner of the file). This is to prevent files that should be shown with dynamic watermarks from being viewed on Microsoft 365 clients that cannot show the watermarks.

If you need to open a file with dynamic watermarking but your client does not support it yet, open the file in Microsoft 365 for the web instead. You'll be able to view, edit and collaborate on your document there with full support for dynamic watermarking.

Watermarks are included when you print a file but are not included if you export it to another file format. Remember this when deciding whether to export a file.

File views outside of Word, Excel and PowerPoint do not support dynamic watermarking. This means that certain experiences, such as PPT Live, will not render watermarks.

To access this feature, you must be in an organisation that is licensed for sensitivity labels in Microsoft 365 and that has enabled dynamic watermarking on a sensitivity label that you can access. Furthermore, you may only apply a label with dynamic watermarks if your administrator has enabled the setting on a sensitivity label available to you.

This feature is currently rolling out to users on the web and to Current Channel Preview users running:

- Windows: Version 2407 (Build 17830.20000) or later
- Mac: Version 16.87 (Build 24070110) or later.

### ***New regular expression (regex) modes for XLOOKUP and XMATCH (Insiders)***

Next up is a new addition to Excel for Windows and Mac – and something we wrote about last month. Nonetheless, we reproduce the article again here for those that might be new to the newsletter or read these things like my mother-in-law drives.

As we have explained in recent newsletters, the term "regex" is an abbreviation of "regular expressions" and is a language used for pattern-matching text content. It is frequently implemented in various programming languages such as C, C++, Java, Python, VBScript – and of course, that latest and greatest software, 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.

To use this fully, you need to understand 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".

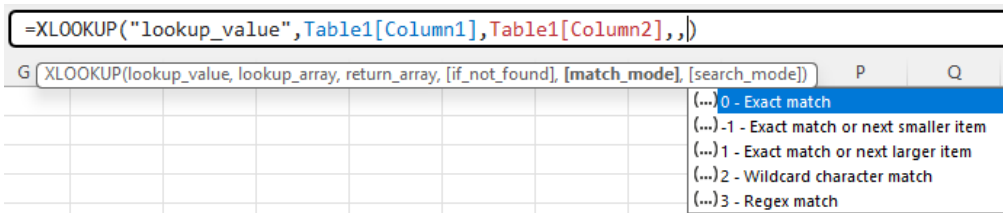
Token	Meaning
\	This converts special characters (metacharacters) to literal characters, and also allows the literal matching of the regex delimiter in use, <i>e.g.</i> '/'
.	Matches any character other than newline
^	Matches the start of string without consuming any characters. If multiline mode is 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 \

Token	Meaning
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 <i>null</i> character, Unicode character U+2400
\d	Matches any decimal / digit. Equivalent to [0-9]
\D	Matches anything other than a decimal / digit
\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, <i>null</i> 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 that I have provided a refresher, regular expressions are starting to infiltrate Excel. It began with three new functions:

1. **REGEXEXTRACT(text, pattern, [return\_mode], [ignore\_case])**
2. **REGEXREPLACE(text, pattern, replacement, [occurrence], [ignore\_case])**
3. **REGEXTTEST(text, pattern, [ignore\_case]).**

However, now you may take further advantage of regex within the existing **XLOOKUP** and **XMATCH** functions, by using the new **match\_mode = 3** and a regex pattern as the **lookup\_value**.



This will allow **XLOOKUP** and **XMATCH** to match against parts of text in a cell, or by any other pattern of text that can be described with regex.

Here is one example we've all been desperate for: **fuzzy matching**, i.e. a search technique used to identify similar text strings, such as looking for a reference to Australia, such as "Aussie", "Australia" or "Oz".

Consider the following Table of data called **Country\_Sales\_Data**:

Examples using Fuzzy Matching										
Data										
Country	Sales									
Aussie	13,942									
Australia	7,119									
Luxembourg	61,844									
Vatican City	307									
Netherlands	119									
Oz	20,438									
Chile	7,746									

Search Result		
Australian sales	13,942	=XLOOKUP("Aussie Australia Oz",Country_Sales_Data[Country],Country_Sales_Data[Sales],3)
Last Occurrence in Table	6	=XMATCH("Aussie Australia Oz",Country_Sales_Data[Country],3,-1)

Here, I have created a formula in cell **H24** which has determined the sales for the first record that contains an aforementioned reference to Australia:

**=XLOOKUP("Aussie | Australia | Oz",Country\_Sales\_Data[Country],Country\_Sales\_Data[Sales],,3)**

Note that the fifth argument (**match\_mode**) is three [3], which is the new Regex match.

"**Aussie | Australia | Oz**" is the regular expression (in quotation marks) that provides alternate matches (see above *Regex tokens table*). It does not matter which order these three alternatives are cited: **XLOOKUP** will seek out the first match, which in this case is the very first record.

**XMATCH** works similarly in cell **H26**, viz.

**=XMATCH("Aussie | Australia | Oz",Country\_Sales\_Data[Country],3,-1)**

Again, note that the third argument (**match\_mode**) is three [3], which is the new Regex match. The fourth argument (**search\_mode**) is -1 here, so that **XMATCH** searches last to first.

This is great, but I am not keen on hard code, so it got me thinking: how about I create a table of date for all my fuzzy match acceptable alternatives? Well, that's precisely what I did:

Examples using Fuzzy Matching with INDIRECT										
Data										
Country	Sales	Australia	Germany	USA						
Lucky Country	59,701	Aussie	Germany	United States of America						
Australia	4,044	Australia	Deutschland	United States						
United States of America	51,236	Oz	East Germany	USA						
Germany	13,968	Down Under	West Germany	US of A						
Deutschland	23,456	Lucky Country								
USA	72									
Oz	609									
United States	11,498									

Search Result		
Region selected	Australia	
Sales for Australia	59,701	=IFERROR(XLOOKUP(INDIRECT(H49),Another_Table[Country],Another_Table[Sales],,3),")
Last Occurrence in Table	7	=IFERROR(XMATCH(INDIRECT(H49),Another_Table[Country],3,-1),")

There is a new Table here called **Another\_Table** (I'm nothing if not unimaginative). Then, in cells **J35:L43**, I have provided an input data table for alternative names for the countries Australia, Germany and USA. In the shaded cells **J44:L44**, I have created three hidden formulae. For example, the formula in cell **J44** is

**=TEXTJOIN("|",TRUE,J36:J43)**

This uses the **TEXTJOIN** function to create a text string of all non-blank values in the range **J36:J43** separating them with **|** and ignoring blanks. This provides the albeit hidden result:

Aussie|Australia|Oz|Down Under|Lucky Country

Each of these three cells has been given a range name: **Australia** (cell **J44**), **Germany** (cell **K44**) and **USA** (cell **L44**). I have then created a drop-down data validation list (**ALT + D + L** or **Data -> Data Validation** from the Ribbon):



The **INDIRECT** function is then employed in the corresponding **XLOOKUP** and **XMATCH** functions. **INDIRECT** allows the creation of a formula by referring to the contents of a cell, rather than the cell reference itself.

The sales figure in cell **H51** is calculated as

**=IFERROR(XLOOKUP(INDIRECT(H49),Another\_Table[Country],Another\_Table[Sales],"",3),"")**

Similarly, the final occurrence formula in cell **H53** is given by

**=IFERROR(XMATCH(INDIRECT(H49),Another\_Table[Country],3,-1),"")**

This means that I can switch the country without having to revise the Regex code:

**Examples using Fuzzy Matching with INDIRECT**

Country	Sales
Lucky Country	59,701
Australia	4,044
United States of America	51,236
Germany	13,968
Deutschland	23,456
USA	72
Oz	609
United States	11,498

Australia	Germany	USA
Aussie	Germany	United States of America
Australia	Deutschland	United States
Oz	East Germany	USA
Down Under	West Germany	US of A
Lucky Country		

**Search Result**

Region selected: **USA**

Sales for USA: **51,236**

Last Occurrence in Table: **8**

**=TEXTJOIN("|",TRUE,J36:J43)**

**Aussie|Australia|Oz|Down Under|Lucky Country**

**=IFERROR(XLOOKUP(INDIRECT(H49),Another\_Table[Country],Another\_Table[Sales],"",3),"")**

**=IFERROR(XMATCH(INDIRECT(H49),Another\_Table[Country],3,-1),"")**

Some people aren't keen on **INDIRECT** as it is both volatile (*i.e.* it recalculates whenever something changes in the file) and is non-auditable ("fools" Excel's built-in audit tools). Therefore, an alternative (using **Yet\_Another\_Table** as the Table data source) would be the following:

**Examples using Fuzzy Matching with Nested XLOOKUP**

Country	Sales
Lucky Country	59,701
Australia	4,044
United States of America	51,236
Germany	13,968
Deutschland	23,456
USA	72
Oz	609
United States	11,498

Australia	Germany	USA
Aussie	Germany	United States of America
Australia	Deutschland	United States
Oz	East Germany	USA
Down Under	West Germany	US of A
Lucky Country		

**Search Result**

Region selected: **USA**

Sales for USA: **51,236**

Last Occurrence in Table: **8**

**=TEXTJOIN("|",TRUE,J63:J70)**

**=IFERROR(XLOOKUP(XLOOKUP(H76,J62:L62,J71:L71),Yet\_Another\_Table[Country],Yet\_Another\_Table[Sales],"",3),"")**

**=IFERROR(XMATCH(XLOOKUP(H76,J62:L62,J71:L71),Yet\_Another\_Table[Country],3,-1),"")**

Here, we swap out the **lookup\_value** in both the **XLOOKUP** and **XMATCH** functions which uses **INDIRECT** for a “nested” **XLOOKUP** expression instead. Thus, cell **H78** contains the formula

```
=IFERROR(XLOOKUP(XLOOKUP(H76,J62:L62,J71:L71),
Yet_Another_Table[Country],Yet_Another_Table[Sales],"",3), "")
```

and cell **H80** contains the revised formula

```
=IFERROR(XMATCH(XLOOKUP(H76,J62:L62,J71:L71),Yet_Another_Table[Country],3,-1), "")
```

Of course, fuzzy matching is just one use of the new features in **XLOOKUP** and **XMATCH**. I can go hunting for text strings that include inadvertent non-numerical values. For example, consider the data Table called **Data**:

	B	C	D	E	F	G	H	I	J	K	L	M	N
85	<b>Examples using Locating Non-Numerical Data</b>												
86													
87					<b>Data</b>								
88													
89							<b>Part Number</b>						
90							0017598						
91							23445743646413						
92							657541346734						
93							245754454541575						
94							1545455454250557454						
95							215454214						
96							234577527						
97							2347452757545						
98							4575457545						
99													
100													
101													
102													
103							First non-Numeric Value						
104													
105							Corresponding value						
106													

Here, the formula in cell **H103** identifies the first record that contains a non-numerical value:

```
=IFERROR(XMATCH("[^0-9]",Data[Part Number],3),"All numerical.")
```

**[^0-9]** simply means find something that is not the numbers zero [0] to nine [9]. Once this has been located, I can then use the **INDEX** function to identify it in cell **H105**:

```
=IFERROR(INDEX(Data[Part Number],H103),"n/a")
```

I have made the non-numerical value deliberately difficult to spot:

1545455454250557454

That’s right: that is the capital letter **O**, not a zero [0]! It didn’t fool Excel.

Of course, you can get more complex:

	B	C	D	E	F	G	H	I	J	K	L	M
110	<b>Examples using Locating Adjacent Repeated Letters (Case Sensitive)</b>											
111												
112					<b>Data</b>							
113												
114							<b>Text</b>					
115							Liam					
116							doesn't					
117							know					
118							Regex					
119							apPle					
120							Cheese					
121							The The					
122							I love spaces					
123							Not blank					
124												
125												
126												
127												
128							First occurrence of repeat					
129												

Here, for the Table **Text\_Data**, I have used the formula

```
=IFERROR(XMATCH("(\\w*([a-zA-Z])\\2\\w*)",Text_Data[Text],3),"No instances.")
```

in cell **H128**. Take my word for it, but the expression

```
(\\w*([a-zA-Z])\\2\\w*)
```

seeks out any text string that contains adjacent repeated letters that are either both lower case or both upper case (hence “apple” is not recognised but “Cheese” is). Try it for yourself; I am sure you can construct even more complex monstrosities!

Of course, you will all whinge at me when you discover you don't have this feature – yet. These new function modes are in Preview only presently. Their results may change substantially before being widely released, based upon Insider Beta users' feedback. Thus, I do not recommend using these functions in important workbooks until they become Generally Available.

### Filtering for comments

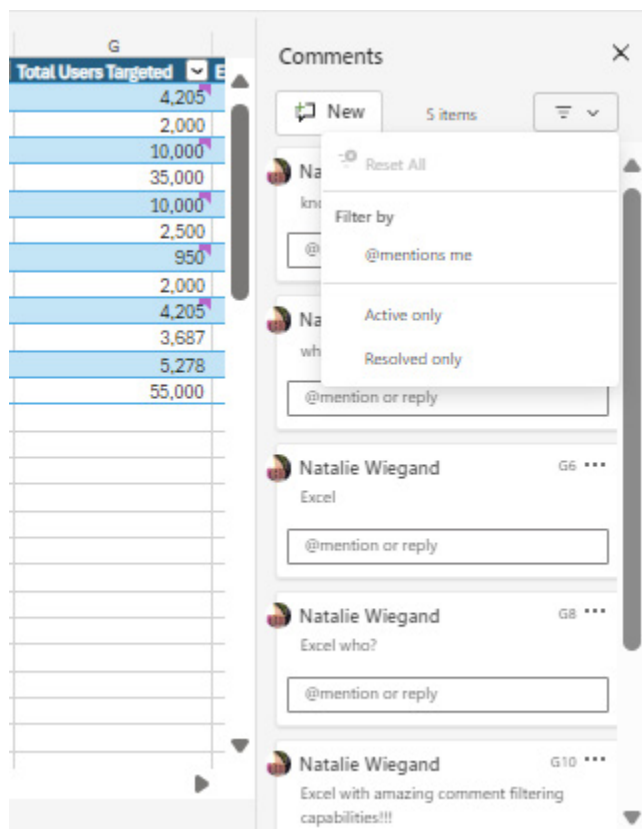
Finally, just for Excel for Windows, you may now filter comments that are active, resolved or ones that @mention you so that you can find the comments you're looking for. Filtering to a certain set of comments will update which comment indicators you see on the worksheet so you can

Presently, these functions are rolling out to Beta Channel users running:

- Windows: Version 2408 (Build 17931.20000)
- Mac: Version 16.89 (Build 24080715).

Don't let it deter you though!

focus on which comments are important with the context of your work. This feature is already supported on Mac and web and is currently rolling out to all Windows users.



Until next month.

## The A to Z of Excel Functions: NUMBERSTRING





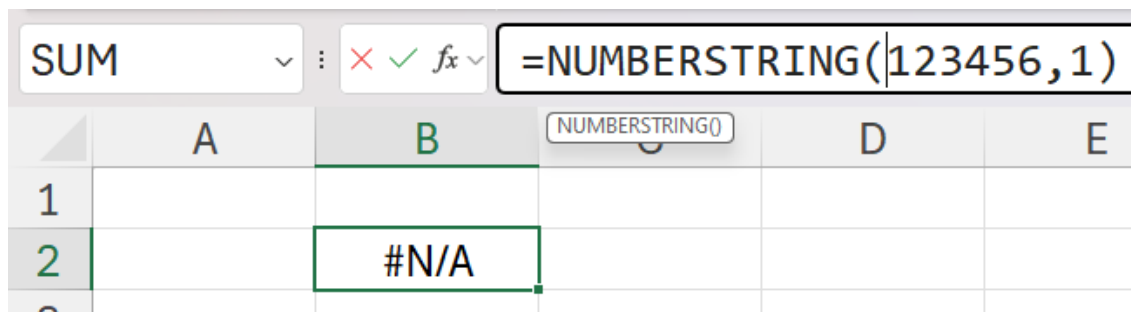
Like **DATESTRING**, **NUMBERSTRING** seems to have been a function created for backward compatibility with Lotus 1-2-3 and other similar programs. It is thought the syntax of the function is of the form

**NUMBERSTRING(number, format)**

where:

- **number** is the **number** to be converted
- **format** is the argument to specify the conversion, e.g. to the Chinese number format.

It appears to be no longer operational but is still recognised in Excel, viz.



This function is presented for completeness only.

## The A to Z of Excel Functions: **NUMBERVALUE**



The **NUMBERVALUE** function converts text into a number, in a locale-independent way. This may also be used to convert local-specific values into locale-independent values.

The **NUMBERVALUE** syntax is as follows:

**NUMBERVALUE(text, [decimal\_separator], [group\_separator])**

The **NUMBERVALUE** function syntax has the following arguments:

- **text**: this is required. This is the **text** to convert into a number
- **decimal\_separator**: this argument is optional. This is the character used to separate the integer and fractional part of the result
- **group\_separator**: this argument is also optional. This represents the character used to separate groupings of numbers, such as thousands from hundreds and millions from thousands.

It should be noted that:

- if the **decimal\_separator** and **group\_separator** arguments are not specified, separators from the current locale are used
- if multiple characters are used in the **decimal\_separator** or **group\_separator** arguments, only the first character is used
- if an empty string ("") is specified as the **text** argument, the result is zero [0]
- empty spaces in the **text** argument are ignored, even in the middle of the argument. For example, " 3 000 " is returned as 3,000
- if a **decimal\_separator** is used more than once in the **text** argument, **NUMBERVALUE** returns the #VALUE! error value
- if the **group\_separator** occurs before the **decimal\_separator** in the **text** argument, the **group\_separator** is ignored
- if the **group\_separator** occurs after the **decimal\_separator** in the **text** argument, **NUMBERVALUE** returns the #VALUE! error value
- if any of the arguments are not valid, **NUMBERVALUE** returns the #VALUE! error value
- if the **text** argument ends in one or more percent signs (%), they are used in the calculation of the result. Multiple percent signs are additive if they are used in the **text** argument just as they are if they are used in a formula. For example, =**NUMBERVALUE**("9%%") returns the same result (0.0009) as the formula =9%%.

Please see my example below:

	A	B	C
1	Text		
2	2.500,27		
3	3.5%		
4			
5			
6	Formula	Description	Result
7	= <b>NUMBERVALUE</b> (A2,"",".")	The decimal separator of the text argument in the example is specified in the second argument as a comma, and the group separator is specified in the third argument as a period.	2,500.27
8	= <b>NUMBERVALUE</b> (A3)	Because no optional arguments are specified, the decimal and group separators of the current locale are used. The % symbol is not shown, although the percentage is calculated.	0.035
9	= <b>NUMBERVALUE</b> ("1.234,5","",".")	The decimal separator of the text argument in the example is again specified in the second argument as a comma, and the group separator is specified in the third argument as a period.	1,234.5

More Excel Functions next month.

## Beat the Boredom Suggested Solution

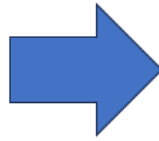
The challenge this month was to find favourite stores and recently visited stores by customer from sales data, using only one [1] formula for each task.

### The Challenge

Earlier in this newsletter, we challenged you to find out the favourite store of each customer using only one [1] Excel formula, and to find out each customer's lastly visited store location with, again, "just" one [1] formula.

The table **Data** contained customer visits data, and the desired outputs were to look like the following upon completion:

Date	Customer	Store
1 Feb 22	Talia Cao	Central
15 Feb 22	Tim Heng	Town Hall
1 Mar 22	Talia Cao	Redfern
15 Mar 22	Sam Ngo	Circular Quay
29 Mar 22	Talia Cao	Central
12 Apr 22	Talia Cao	Town Hall
26 Apr 22	Sam Ngo	Central
10 May 22	Liam Bastick	Wynyard
24 May 22	Sam Ngo	Redfern
7 Jun 22	Liam Bastick	Central
21 Jun 22	Talia Cao	Circular Quay
5 Jul 22	Tim Heng	Redfern
19 Jul 22	Liam Bastick	Wynyard
2 Aug 22	Sam Ngo	Central
5 Jan 23	Talia Cao	Central
5 Jan 23	Tim Heng	Circular Quay
5 Jan 23	Sam Ngo	Town Hall



#### Most frequently visited store

Customer	Store
Talia Cao	Central
Tim Heng	Town Hall
Sam Ngo	Central
Liam Bastick	Wynyard

#### Most recently visited store

Customer	Store
Talia Cao	Central
Tim Heng	Circular Quay
Sam Ngo	Town Hall
Liam Bastick	Wynyard

There were some requirements:

- each formula needed to be within one cell
- this was a formula challenge; no Power Query / Get & Transform or VBA.

### Suggested Solution for Task 1: Most Frequently Visited Store

The steps are detailed below.

#### COUNT THE VISITS

We can first use **UNIQUE** and **TRANSPOSE** to get the lists of customers and stores from **Data**, and place them as row and column headers for our array of counts. The formula

**=UNIQUE(Data[Customer])**

provides an unsorted list of the customers down a column. The formula

**=TRANSPOSE(UNIQUE(Data[Store]))**

provides a list of the stores across a row (hence the need for the **TRANSPOSE** function, which transforms a column vector to a row vector).

The number of visits by customer to a store can be found using **COUNTIFS** with **Data[Store]** and **Data[Customer]** as the criteria ranges. Also, be careful with table references and remember to use **CTRL + D** and **CTRL + R** to copy the formula down and to the right. Then **COUNTIFS** produces the following matrix of counts:

	Central	Town Hall	Redfern	Circular Quay	Wynyard
Talia Cao	3	1	1	1	-
Tim Heng	-	1	1	1	-
Sam Ngo	2	1	1	1	-
Liam Bastick	1	-	-	-	2

Moreover, if we use the column headers (stores) in the first criteria argument of **COUNTIFS** instead of single stores, and the row headers (customers) in the second criteria argument, we can obtain the whole matrix as a spilled array using only one formula:

**=COUNTIFS(Data[Store], F10#, Data[Customer], E11#)**

This is the Dynamic Array equivalent of a PivotTable.

## FIND THE FAVOURITE STORES

Given the matrix of visit counts, we can look for the most frequently visited store by each customer and output the corresponding store name. An **XLOOKUP** formula for the maximum of each row with store names being the return array can achieve that:

**=XLOOKUP(MAX(\$F11:\$J11), \$F11:\$J11, F\$10#)**

Here it's necessary to anchor the column titles **F10#** at the row, to fill down for all customers. The output would look like the following:

	Central	Town Hall	Redfern	Circular Quay	Wynyard	Favourite Stores
Talia Cao	3	1	1	1	-	Central
Tim Heng	-	1	1	1	-	Town Hall
Sam Ngo	2	1	1	1	-	Central
Liam Bastick	1	-	-	-	2	Wynyard

So how do we look up for all customers in one [1] formula? We can use the **BYROW** function. The **BYROW** function has two [2] arguments: an array and a **LAMBDA** function:

**=BYROW(array, LAMBDA(parameter, calculation))**

The **LAMBDA** function takes each row of the array as its parameter and performs the specified calculation on it. Given the above matrix **F11#** of visit counts, we can use a **BYROW** function on the whole matrix, and write the **XLOOKUP** as the **LAMBDA** function inside:

**=BYROW(F11#, LAMBDA(r, XLOOKUP(MAX(r), r, F10#)))**

Then, the **XLOOKUP** will be performed row-by-row (hence the name **BYROW**). For interested readers we have written blogs with more details on **BYROW** and **LAMBDA**.

## COMBINE THE FORMULAE

In the previous two [2] steps, we first obtained the matrix **F11#** of visit counts using **COUNTIFS** with two-dimensional conditions, and then we used **BYROW** on **F11#** to perform an **XLOOKUP** for the most frequently visited store by each customer. The two [2] steps can be combined by writing the first formula in place of **F11#** inside the second formula:

**=BYROW(COUNTIFS(Data[Store], F10#, Data[Customer], E11#),  
LAMBDA(r, XLOOKUP(MAX(r), r, F10#)))**

The **COUNTIFS** matrix is only a helper, so instead of referencing the column headers (**F10#**) and row headers (**E11#**), we can substitute with the original **UNIQUE** and **TRANSPOSE** formulas:

**=BYROW(COUNTIFS(Data[Store], TRANSPOSE(UNIQUE(Data[Store])),  
Data[Customer], UNIQUE(Data[Customer])),  
LAMBDA(r, XLOOKUP(MAX(r), r, TRANSPOSE(UNIQUE(Data[Store])))))**

For better readability, we can also use a **LET** function to name some variables for intermediate steps:

**=LET(Store, TRANSPOSE(UNIQUE(Data[Store])),  
Customer, UNIQUE(Data[Customer]),  
Matrix, COUNTIFS(Data[Store], Store, Data[Customer], Customer),  
BYROW(Matrix, LAMBDA(r, XLOOKUP(MAX(r), r, Store))))**

Here we name the intermediate matrix **Matrix**, and the final calculation of the **LET** is a **BYROW** on **Matrix** to look-up the **MAX** from each row.

## Suggested Solution for Task 2: Most Recently Visited Store

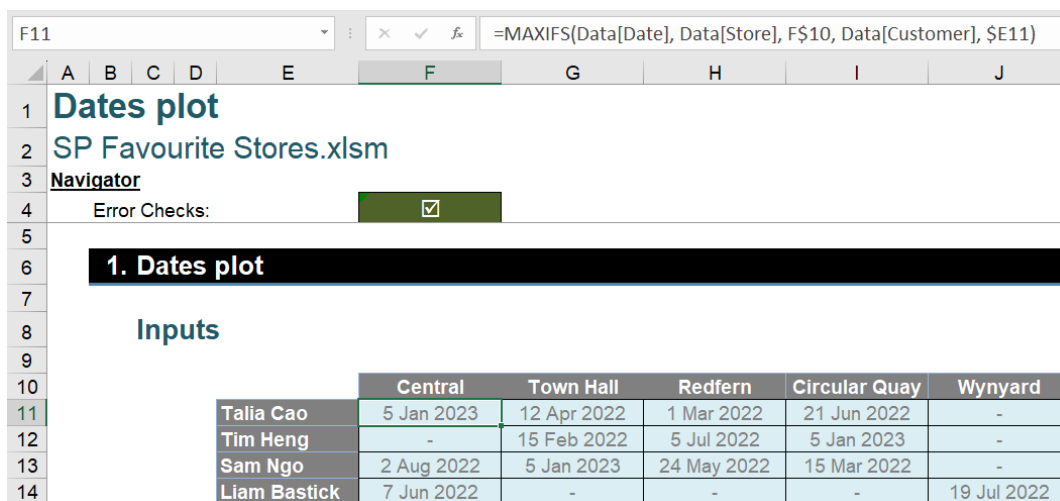
Let's continue.

## STARTING SIMILARLY...

To find the lastly visited store by each customer, we can use an approach very similar to the previous solution, by sorting visit dates of each customer to each store with a **MAXIFS**:

**=MAXIFS(Data[Date], Data[Store], F\$10, Data[Customer], \$E11)**

Again, use **CTRL + D** and **CTRL + R** to fill the formula down and to the right. Then the output will look like the following:



	Central	Town Hall	Redfern	Circular Quay	Wynyard
Talia Cao	5 Jan 2023	12 Apr 2022	1 Mar 2022	21 Jun 2022	-
Tim Heng	-	15 Feb 2022	5 Jul 2022	5 Jan 2023	-
Sam Ngo	2 Aug 2022	5 Jan 2023	24 May 2022	15 Mar 2022	-
Liam Bastick	7 Jun 2022	-	-	-	19 Jul 2022

We can look up the latest visit of each customer to return the corresponding store. Furthermore, following the same process we can build up a final formula:

**=LET(Store, TRANSPOSE(UNIQUE(Data[Store])),  
Customer, UNIQUE(Data[Customer]),  
Matrix, MAXIFS(Data[Date], Data[Store], Store, Data[Customer], Customer),  
BYROW(Matrix, LAMBDA(r, XLOOKUP(MAX(r), r, Store))))**

## A DIFFERENT APPROACH

The above approach is arguably not optimal for the second task, in the sense that it is sorting dates twice. The **MAXIFS** first sorts visit dates for each customer to each store, and then the **XLOOKUP** sorts the latest visits to each store by each customer. Conceptually, the optimal algorithm would be sorting all visits of each customer by dates, and then returning the corresponding stores of those latest visits. That is, instead of a two-dimensional problem, this task is a two-layer problem.

We can use the combination of **FILTER**, **SORT** and **INDEX**. For example, for the customer 'Sam Ngo', we can get his most recently visited store with the following formula:

**=INDEX(SORT(FILTER(Data, Data[Customer] = "Sam Ngo"), 1, -1), 1, 3)**

Here **FILTER** first filters for all store visits of this customer. Then **SORT** sorts the first column, **Date**, in descending order, so the first row of the spilled array will contain the date, customer name and the store location of the latest store visit of 'Sam Ngo'. Lastly **INDEX** outputs the cell in the first row and the third column (**Store**) of the table.

To cover all customers in one formula, we can again use **BYROW**:

**=BYROW(UNIQUE(Data[Customer]), LAMBDA(name, INDEX(SORT(FILTER(Data, Data[Customer]=name), 1, -1), 1, 3)))**

## COMPARING THE APPROACHES

The advantage of the above approach is that **FILTER** preserves rows of the table **Data**. Consider instead if we were to use **MAXIFS** instead of **FILTER** and **SORT**, to replicate the same idea of sorting **Date** for each **Customer** first and then return the corresponding **Store**. For each customer we would start with:

**=MAXIFS(Data[Date], Data[Customer], \$F11)**

Then the outcome would be a single date value, which is not necessarily unique:

	A	B	C	D	E	F	G	H
1	<b>Discussion plot</b>							
2	SP Favourite Stores.xlsm							
3	Navigator							
4	Error Checks:					<input checked="" type="checkbox"/>		
5								
6	<b>1. Discussion plot</b>							
7								
8	<b>Inputs</b>							
9								
10					<b>Last visit</b>			
11	Talia Cao				5 Jan 2023			
12	Tim Heng				5 Jan 2023			
13	Sam Ngo				5 Jan 2023			
14	Liam Bastick				19 Jul 2022			

That means we can't use the sorted-out dates alone to look up the corresponding **Store**. We have no choice but to combine it with the list of customers. One way is to use the ampersand (&) to concatenate a unique **Customer-Date** label, to look up from the table **Data**:

=INDEX(Data[Store], MATCH(\$F11&" - "&\$G11, Data[Customer]  
&" - "&Data[Date], 0))

In comparison, **FILTER** preserves the rows, and hence preserves the **Date-Customer-Store** connection. We just need to specify the index to sort and to output.

More next month.

## STOP PRESS: GROUPBY and PIVOTBY are now Generally Available

**GROUPBY** and **PIVOTBY** – and their associated functions and features – have finally gone into Production. These functions may now be accessed by the Office 365 masses. As a reminder, just under a year ago, Microsoft announced several new functions, including **PIVOTBY**, and eta lambdas such as **PERCENTOF**.

### 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:

	C	D	E	F	G	H	I	J	K	L	M	N
9												
10	<b>Using SUM with BYCOL</b>											
11												
12		Q1	Q2	Q3	Q4							
13	North	8	8	3	10							
14	South	9	4	9	9							
15	East	8	4	5	8							
16	West	1	10	2	6							
17	<b>Total</b>	<b>26</b>	<b>26</b>	<b>19</b>	<b>33</b>	=BYCOL(G13:J16,LAMBDA(x,SUM(x)))						
18												

=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.

	E	F	G	H	I	J	K	L	M
19									
20			Q1	Q2	Q3	Q4			
21		North	8	8	3	10			
22		South	9	4	9	9			
23		East	8	4	5	8			
24		West	1	10	2	6			
25		<b>Total</b>	<b>26</b>	<b>26</b>	<b>19</b>	<b>33</b>			=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).

### GROUPBY

The new **GROUPBY** function allows you to create a summary of your data formulaically. It supports grouping along one axis and aggregating the associated values. For instance, if you had a table of sales data, you might generate a summary of sales by year, or by salesperson, or by category, or by...

In essence, it allows you to group, aggregate, sort and filter data based upon the fields you specify.

The syntax of the **GROUPBY** function is given by:

**GROUPBY(row\_fields, values, function, [field\_headers], [total\_depth], [sort\_order], [filter\_array], [field\_relationship])**

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
- **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** 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

- **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, fields must have at least two [2] columns. Numbers greater than two [2] are supported provided there are sufficient columns

- **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**

- **filter\_array**: the 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**
- **field\_relationship**: the final optional argument, this specifies the relationship fields when multiple columns are provided to **row\_fields**. The possible values are:
  - **0**: Hierarchy (default)
  - **1**: Table
- With a Hierarchy field relationship [0], sorting of later field columns takes into account the hierarchy of earlier columns; with a Table field relationship [1], sorting of each field column is done independently. Subtotals are not supported as they rely on the data having a hierarchy.

To show how **GROUPBY** works, we took inspiration from Microsoft's data table:

## Example Data

**Table Used for Formulae**

Year	Category	Item	Sales	Rating
2020	Components	Wheels	4,000	10%
2022	Components	Pedals	3,200	50%
2020	Components	Brakes	3,300	45%
2020	Clothing	Jerseys	1,100	10%
2020	Components	Saddles	500	85%
2020	Clothing	Jerseys	1,500	30%
2021	Accessories	Bike Racks	2,600	85%
2020	Bikes	Touring Bikes	1,100	30%
2022	Clothing	Tights	800	65%
2021	Clothing	Bib-Shorts	1,000	45%
2021	Accessories	Helmets	2,700	45%
2020	Clothing	Gloves	800	20%
2022	Clothing	Vests	1,100	30%
2021	Components	Brakes	1,100	100%
2022	Components	Handlebars	3,200	25%
2022	Accessories	Locks	400	55%
2021	Accessories	Tyres and Tubes	500	70%
2020	Components	Pedals	1,000	45%
2021	Accessories	Helmets	3,600	60%
2020	Bikes	Touring Bikes	200	55%
2021	Clothing	Gloves	4,000	100%
2020	Accessories	Locks	1,500	75%
2022	Bikes	Road Bikes	600	75%
2022	Clothing	Gloves	900	65%
2022	Components	Chains	100	10%
2022	Components	Chains	1,600	45%
2021	Bikes	Touring Bikes	2 400	70%

I have converted this data table into an Excel Table by selecting all the data and using **Insert -> Table (CTRL + T)** and calling the resultant Table **tbl**. Look, it's late as I write this and I have no imagination, OK!?

I can summarise my Table very simply using the formula

**=GROUPBY(tbl[Category],tbl[Sales],SUM)**

Description	Amount
Accessories	485,500
Bikes	495,800
Clothing	509,700
Components	493,200
<b>Total</b>	<b>1,984,200</b>

**=GROUPBY(tbl[Category],tbl[Sales],SUM)**



How easy is that!? Essentially, I am summing the sales (using the eta lambda **SUM**) by the **Category** field.

If you want to aggregate by more than one **row\_field**, as stated above, this is possible. One way is to use **HSTACK**:

**=GROUPBY(HSTACK(tbl[Year],tbl[Category]),tbl[Sales],SUM)**

Year	Category	Sales
2020	Accessories	193,500
2020	Bikes	144,300
2020	Clothing	182,900
2020	Components	175,600
2021	Accessories	145,400
2021	Bikes	161,800
2021	Clothing	173,600
2021	Components	142,400
2022	Accessories	146,600
2022	Bikes	189,700
2022	Clothing	153,200
2022	Components	175,200
<b>Total</b>		<b>1,984,200</b>

*=GROUPBY(HSTACK(tbl[Year],tbl[Category]),tbl[Sales],SUM)*

This simply combines the **Year** and **Category** fields in the **tbl** Table, and then sums **Sales** across them. However, I think I prefer the **CHOOSECOLS** approach:

**=GROUPBY(CHOOSECOLS(tbl,1,2),tbl[Sales],SUM)**

Year	Category	Sales
2020	Accessories	193,500
2020	Bikes	144,300
2020	Clothing	182,900
2020	Components	175,600
2021	Accessories	145,400
2021	Bikes	161,800
2021	Clothing	173,600
2021	Components	142,400
2022	Accessories	146,600
2022	Bikes	189,700
2022	Clothing	153,200
2022	Components	175,200
<b>Total</b>		<b>1,984,200</b>

*=GROUPBY(CHOOSECOLS(tbl,1,2),tbl[Sales],SUM)*

Here, the idea is that I shall **SUM Sales** by columns 1 (**Year**) and 2 (**Category**) of the **tbl** Table. This might not seem as clear as the **HSTACK** alternative at first glance as you have to refer to the Table to identify what the columns are. However, stick with me. Let me make the formula more complex:

**=GROUPBY(CHOOSECOLS(tbl,MATCH(F\$12,tbl[#Headers],0),MATCH(G\$12,tbl[#Headers],0)),tbl[Sales],SUM)**

Year	Category	Sales
2020	Accessories	193,500
2020	Bikes	144,300
2020	Clothing	182,900
2020	Components	175,600
2021	Accessories	145,400
2021	Bikes	161,800
2021	Clothing	173,600
2021	Components	142,400
2022	Accessories	146,600
2022	Bikes	189,700
2022	Clothing	153,200
2022	Components	175,200
<b>Total</b>		<b>1,984,200</b>

*=GROUPBY(CHOOSECOLS(tbl,MATCH(F\$12,tbl[#Headers],0),MATCH(G\$12,tbl[#Headers],0)),tbl[Sales],SUM)*

Looks horrible, yes? I have replaced the values 1 and 2 in the previous formula with

**MATCH(F\$12,tbl[#Headers],0)**

and

**MATCH(G\$12,tbl[#Headers],0)**

which return the positions in the **Headers** row of the Table **tbl**. Now, this may seem overkill but consider the following image:

Year	Category	Sales
2020	Accessories	193,500
2020	Bikes	144,300
2020	Clothing	182,900
2020	Components	175,600
2021	Accessories	145,400
2021	Bikes	161,800
2021	Clothing	173,600
2021	Components	142,400
2022	Accessories	146,600
2022	Bikes	189,700
2022	Clothing	153,200
2022	Components	175,200
<b>Total</b>		<b>1,984,200</b>

Brilliant. I have changed the background colour of the first two headers to yellow. Well no, it's a little more than that. I have used data validation dropdown lists (**ALT + D + L**) to create input headers!!

Year	Category	Sales
Year	Accessories	193,500
Category	Bikes	144,300
Item	Clothing	182,900
Sales	Components	175,600
Rating	Accessories	145,400
2021	Bikes	161,800
2021	Clothing	173,600
2021	Components	142,400
2022	Accessories	146,600
2022	Bikes	189,700
2022	Clothing	153,200
2022	Components	175,200
<b>Total</b>		<b>1,984,200</b>

Thus, if I change the selections, I have dynamic summarisations, such as

Category	Item	Sales
Accessories	Bike Racks	82,700
Accessories	Helmets	115,700
Accessories	Lights	64,900
Accessories	Locks	72,000
Accessories	Pumps	72,900
Accessories	Tyres and Tube	77,300
Bikes	Cargo Bikes	149,300
Bikes	Mountain Bikes	122,500
Bikes	Road Bikes	108,100
Bikes	Touring Bikes	115,900
Clothing	Bib-Shorts	52,300
Clothing	Caps	53,300
Clothing	Gloves	72,400
Clothing	Jerseys	79,000
Clothing	Shorts	67,000
Clothing	Socks	58,700
Clothing	Tights	64,700
Clothing	Vests	62,300
Components	Bottom Bracket	60,000
Components	Brakes	53,100
Components	Chains	65,000
Components	Handlebars	85,600
Components	Pedals	87,900
Components	Saddles	64,700
Components	Wheels	76,900
<b>Total</b>		<b>1,984,200</b>

or

Rating	Category	Sales
0.05	Accessories	20,800
0.05	Bikes	32,200
0.05	Clothing	25,800
0.05	Components	24,800
0.1	Accessories	25,800
0.1	Bikes	30,800
0.1	Clothing	28,200
0.1	Components	32,000
0.15	Accessories	32,000
0.15	Bikes	15,600
0.15	Clothing	26,500
0.15	Components	18,400
0.2	Accessories	26,600
0.2	Bikes	19,800
0.2	Clothing	17,900
0.2	Components	27,600
0.25	Accessories	22,100
0.25	Bikes	31,000
0.25	Clothing	19,500
0.25	Components	23,300
0.3	Accessories	25,000
0.3	Bikes	36,000
0.3	Clothing	24,700
0.3	Components	30,300
0.35	Accessories	26,800
0.35	Bikes	18,600
0.35	Clothing	16,200

Having multiple summary statistics may be created similarly, or else you can simply connect them if the reporting fields are contiguous, e.g.

**=GROUPBY(CHOOSECOLS(tbl,1,2),tbl[[Sales]:[Rating]],AVERAGE)**

Year	Category	Sales	Rating
2020	Accessories	2,059	48%
2020	Bikes	1,659	51%
2020	Clothing	2,204	49%
2020	Components	2,116	52%
2021	Accessories	1,795	48%
2021	Bikes	1,926	49%
2021	Clothing	2,019	51%
2021	Components	2,064	54%
2022	Accessories	2,065	53%
2022	Bikes	2,062	54%
2022	Clothing	1,990	55%
2022	Components	1,947	50%
<b>Total</b>		<b>1,990</b>	<b>51%</b>

**=GROUPBY(CHOOSECOLS(tbl,1,2),tbl[[Sales]:[Rating]],AVERAGE)**

Here, **tbl[[Sales]:[Rating]]** may be used to specify the **values** as they are side by side.

Obviously, there are many more arguments to play with, but hopefully, you get the general idea, such as ranking the **Item** field in descending order by **Sales** using the formula

**=GROUPBY(tbl[Item],tbl[Sales],SUM,,,-2)**

Item	Sales
Cargo Bikes	149,300
Mountain Bikes	122,500
Touring Bikes	115,900
Helmets	115,700
Road Bikes	108,100
Pedals	87,900
Handlebars	85,600
Bike Racks	82,700
Jerseys	79,000
Tyres and Tubes	77,300
Wheels	76,900
Pumps	72,900
Gloves	72,400
Locks	72,000
Shorts	67,000
Chains	65,000
Lights	64,900
Tights	64,700
Saddles	64,700
Vests	62,300
Bottom Brackets	60,000
Socks	58,700
Caps	53,300
Brakes	53,100
Bib-Shorts	52,300
<b>Total</b>	<b>1,984,200</b>

`=GROUPBY(tbl[Item],tbl[Sales],SUM,,-2)`

Indeed, the outputs summarised don't have to be numerical. A more comprehensive example summarising the **Items** field might look like this:

`=GROUPBY(tbl[Category],tbl[Item],LAMBDA(x,ARRAYTOTEXT(SORT(UNIQUE(x))))))`

Accessories	Bike Racks, Helmets, Lights, Locks, Pumps, Tyres and Tubes
Bikes	Cargo Bikes, Mountain Bikes, Road Bikes, Touring Bikes
Clothing	Bib-Shorts, Caps, Gloves, Jerseys, Shorts, Socks, Tights, Vests
Components	Bottom Brackets, Brakes, Chains, Handlebars, Pedals, Saddles, Wheels
Total	Bib-Shorts, Bike Racks, Bottom Brackets, Brakes, Caps, Cargo Bikes, Chains, Gloves, Handlebars, Helmets, Jerseys, Lights, Locks, Mountain Bikes, Pedals, Pumps, Road Bikes, Saddles, Shorts, Socks, Tights, Touring Bikes, Tyres and Tubes, Vests, Wheels

`=GROUPBY(tbl[Category],tbl[Item],LAMBDA(x,ARRAYTOTEXT(SORT(UNIQUE(x))))))`

### 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

$\text{SUM}(\text{subset}) / \text{SUM}(\text{everything})$

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%
<b>Total</b>	<b>100.00%</b>

`=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%
<b>Total</b>	<b>100.00%</b>

Accessories and Bikes	49.46%	<code>=PERCENTOF(G13:G14,G13:G16)</code>
-----------------------	--------	--

## PIVOTBY

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**. 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.**

You 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	2022	2022	2022	2023	2023	2023	2023	2024	2024	2024	2024	Total
		Q1	Q2	Q3	Q4	Q1	Q2	Q3	Q4	Q1	Q2	Q3	Q4	
Accessories	Bike Racks	10.7%	4.8%	2.8%	0.3%	1.6%	3.1%		5.2%	1.2%	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%	4.3%	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%			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%	5.9%	4.4%
Components	Saddles	2.7%	5.6%	2.9%	0.9%	2.8%	3.0%	2.7%	3.2%	5.0%	6.5%	2.9%	3.3%	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%
<b>Total</b>		<b>100.0%</b>	<b>100.0%</b>	<b>100.0%</b>	<b>100.0%</b>	<b>100.0%</b>	<b>100.0%</b>	<b>100.0%</b>	<b>100.0%</b>	<b>100.0%</b>	<b>100.0%</b>	<b>100.0%</b>	<b>100.0%</b>	<b>100.0%</b>



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				2023				2024				Total
		Q1	Q2	Q3	Q4	Q1	Q2	Q3	Q4	Q1	Q2	Q3	Q4	
Accessories	Bike Racks	59.1%	26.7%	12.5%	1.8%	14.7%	28.2%		57.1%	5.4%	26.6%	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%	21.6%	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	Tires 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%	15.5%	52.4%	3.6%	42.2%	32.7%	21.5%	32.9%	17.5%	20.7%	28.8%	100.0%
Bikes	Touring Bikes	25.9%	20.9%	24.6%	28.9%	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%	25.5%	45.6%	100.0%
Clothing	Jerseys	7.7%	26.1%		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%	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%	49.5%	14.8%	17.2%	16.7%	100.0%
Clothing	Vests	38.6%	35.4%	2.5%	23.4%	42.1%	15.6%	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%	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%	38.8%	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%		1.9%	40.1%	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				2023				2024				Total
		Q1	Q2	Q3	Q4	Q1	Q2	Q3	Q4	Q1	Q2	Q3	Q4	
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	Tires 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%	18.3%	19.8%	15.5%
Clothing	Shorts	14.5%	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%	9.6%	26.9%	9.4%	23.5%	6.9%	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	28.5%	11.0%	37.1%		0.6%	24.4%	17.6%	11.4%	16.7%	11.9%	23.4%	1.3%	15.8%
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).

Time to play!!



## Upcoming SumProduct Training Courses

Location	Course	Course Date	Local Time	UTC	Duration
Sydney Australia	Excel Tips and Tricks	1 October 2024	09:00 - 17:00 AEST	30 Sep 2024 23:00 UTC - 01 Oct 2024 07:00 UTC	1 Day
Melbourne Australia	Financial Modelling	14 October 2024 - 15 October 2024	09:00 - 17:00 AEDT	13 Oct 2024 22:00 UTC - 15 Oct 2024 06:00 UTC	2 Days
Philadelphia, USA	Excel Tips and Tricks	15 October 2024	09:00 - 17:00 EDT	15 Oct 2024 13:00 UTC - 15 Oct 2024 21:00 UTC	1 Day
New York, USA	Excel Tips and Tricks	24 October 2024	09:00 - 17:00 EDT	24 Oct 2024 13:00 UTC - 24 Oct 2024 21:00 UTC	1 Day
London UK	ChatGPT	29 October 2024 - 30 October 2024	09:00 - 17:00 GMT	29 Oct 2024 09:00 UTC - 30 Oct 2024 17:00 UTC	2 Days
Melbourne Australia	Power Pivot, Power Query and Power BI	5 November 2024 - 6 November 2024	09:00 - 17:00 AEDT	4 Nov 2024 22:00 UTC - 6 Nov 2024 06:00 UTC	2 Days
Philadelphia USA	Excel Tips and Tricks	11 December 2024	14:00 - 17:00 EDT	11 Dec 2024 19:00 UTC - 11 Dec 2024 22:00 UTC	3 Hours

## Key Strokes

Each newsletter, we'd like to introduce you to useful keystrokes you may or may not be aware of. This time, we start to play we put in an **AL**ternative **SHIFT** with the function keys this month:

Keystroke	What it does
ALT + SHIFT + F1	Insert new sheet
ALT + SHIFT + F2	Save
ALT + SHIFT + F4	Close application
ALT + SHIFT + F10	Show On-Object User Interface (OOUI)
ALT + SHIFT + F11	Show Script Editor

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](http://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](mailto: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](http://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](mailto: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](mailto:training@sumproduct.com) for a copy of the brochure or download it directly from [www.sumproduct.com/training](http://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](mailto:contact@sumproduct.com)  
[www.sumproduct.com](http://www.sumproduct.com)  
 +61 3 9020 2071