

Sum Product

NEWSLETTER #145 - December 2024

www.sumproduct.com | www.sumproduct.com/thought



They say it “dozen” matter – but we’ve turned 12!! All aboard as we embark on the terrible teens journey! We start off with our new lease of life considering how a certain accounting standard may change financial modelling and reporting for many of us.

There is also a shout out to the Australasian Reporting Awards (seriously consider it if you are in the area!), plus there is all the usual killer no filler. We’ve got our latest Beat the Boredom Challenge, Charts & Dashboards tips, Excel for Mac, Visual Basics, Power Pivot Principles, Power Query Pointers, Power BI Updates, the latest Excel Updates, plus the **ODD** A to Z of Excel functions and Keyboard Shortcuts continue to keep your fingers busy.

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

Liam Bastick, Managing Director, SumProduct



IFRS 18 and its Impact on Financial Modelling



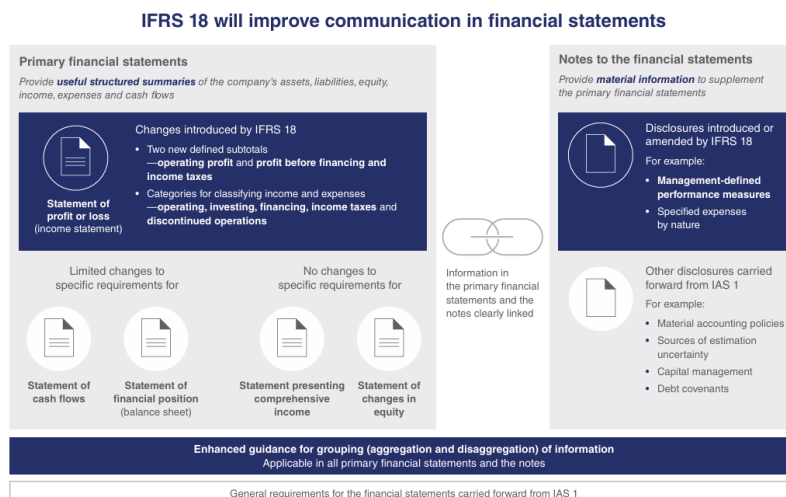
Apologies to our US readers who diligently follow the United States Generally Accepted Accounting Principles (US GAAP), but we need to talk about the International Financial Reporting Standards (IFRS) much of the rest of the world follows. These are a set of accounting standards developed by the International Accounting Standards Board (IASB) to provide a global framework for financial reporting. These standards aim to bring transparency, accountability and efficiency to financial markets around the world.

Among these standards, IFRS 18 is a new standard first issued on 9 April 2024 which governs how companies communicate in their financial statements, with a focus on information about financial performance in

the statement of profit or loss. This will play a crucial role in shaping financial reporting and therefore impact financial modelling in the near future. It must be adopted by 1 January 2027, but can be implemented earlier. As we hit December, perhaps now is a good time to think of its effect in the upcoming years.

IFRS 18 requires entities to classify income and expenses into five [5] categories, three [3] of which are new, operating, investing and financing, as well as the income tax and discontinued operation categories. The new standard sets out detailed requirements for classifying income and expenses into each category.

In summary, there is a helpful graphic:



Impact of IFRS 18 on Financial Modelling

The introduction of IFRS 18 has several significant implications for financial modelling and forecasting. This standard sets out the essential requirements for presenting and disclosing information in financial statements. It emphasises the importance of providing relevant, reliable

and comparable information that users may adopt and use to make informed economic decisions. The standard covers various aspects, including the structure and content of financial statements, notes, and supplementary disclosures.

Models will need to estimate, forecast and interpret these disclosures. In particular, analysts, accountants and modellers will need to:

- ensure consistency and comparability in financial reporting across different entities and jurisdictions
- enhance transparency by requiring entities to disclose pertinent information that reflects their financial position, performance and cash flows
- provide a framework for presenting financial statements that aids users in understanding the financial health and operations of an entity.

The aim is to address the shortcomings of the old standard (IAS 1) regarding presentation of the financial statements. Many complained that there was insufficient analysis of:

- the classification of income and expenses in the P&L
- the presentation of subtotals in the P&L
- the aggregation and disaggregation of information in general.

Using IFRS 18, a financial modeller forecasting the financial performance of two companies in the same industry can better rely on the standardised presentation and disclosure requirements of IFRS 18. This ensures that key financial metrics, such as revenue, expenses and profits, are reported consistently, facilitating a more accurate comparative analysis, especially when broken down into:

- operating activities
- investing activities
- financial activities
- income taxes
- discontinued operations
- required subtotals of 'Operating profit or loss', 'Profit or loss before financing and income taxes' and 'Profit or loss'.

It's clear modellers will need to understand the new categories for the Income Statement to better align it with the Cash Flow Statement. IFRS 18 provides general guidance for entities to classify the items among these categories, but perhaps not in great detail.

Income taxes and discontinued operations should be reasonably straightforward to determine, but the remaining three categories arguably differ slightly from their Cash Flow Statement counterparts:

Operating	Investing	Financing
<p>This is not actually defined by IFRS 18 (!) and represents the "residual" (dump?) category for income and expenses that are not classified in other categories. This will typically include the entity's results from its main business activities.</p> <p>"Default" category.</p>	<p>This category will typically include:</p> <ul style="list-style-type: none"> • results of associated undertakings and joint ventures • results of cash and cash equivalents • assets that generate a return individually and largely independently of other resources. 	<p>This category will include:</p> <ul style="list-style-type: none"> • all income and expenses from liabilities that involve only the raising of finance (e.g. bank borrowings) • interest expense and the effects of changes in interest rates from other liabilities (such as unwinding of the discount on a pension liability).

This will not be entirely straightforward as categorisations will differ between industries. Modelling categories correctly will become crucial. For instance:

	Insurance company	Investment property company	Manufacturer	Retail bank
Gains / losses on investment property	Investing	Operating	Investing	Investing
Interest expense on borrowings	Operating	Financing	Financing	Operating
Interest income on cash and cash equivalents	Operating	Investing	Investing	Operating
Interest income on loans to customers	Investing	Investing	Investing	Operating

Modelling is likely to be needed to be detailed in greater granularity going forward. SumProduct's approach of using control accounts to populate these new charts of accounts will likely be an asset. For example, the Financing category will help enable investors to better analyse companies' performance before the effects of Financing.

A good financial model – both in design and calculation – will greatly assist. Profits may need to be visualised by nature, function or in combination too, *e.g.*

By Function	By Nature	Mixed
Revenue	Revenue	Revenue
Costs of Goods Sold	Purchase of materials	Costs of Goods Sold
Gross Profit	Transport Costs	Gross Profit
R&D Expenses	Depreciation	Impairment Expenses
Administrative Expenses	Employee Costs	Administrative Expenses
Operating Profit	Operating Profit	Operating Profit

Models must be sufficiently flexible going forward to not only undertake scenario and sensitivity analyses, they will also need to consider alternative presentations of key outputs. Furthermore, the standard also discusses Management-defined Performance Measures (MPMs). These may include:

- subtotals of income and expenses not required or specifically exempted by IFRS
- public communications outside financial statements
- measures that communicate management's view of the company's financial performance.

IFRS 18 requires disclosures such as:

- reconciliation back to the IFRS-defined subtotals
- explanation of why and how the MPM is reported
- explanation of any changes made to the MPM.

Modelling will be needed for the quantitative disclosures too:

Explanation	Description	Reconciliation
That MPMs provide management's view of an aspect of the financial performance of the entity as a whole	The aspect of financial performance that is communicated by MPM	The MPM and the comparable subtotals presented using IFRS requirements
That MPMs are not necessarily comparable with other entities' measure that have similar labels	Why MPM provides useful information about the entity's performance	For each reconciling item disclose the amounts related to each line item in the income statement
	How MPM is calculated including whether and how the measure differs from accounting policies in IFRS Accounting Standards	For each reconciling item disclose how the reconciling item is calculated and how it provides useful information

If you haven't already considered the new IFRS 18 approach, check out our Financial Modelling courses or consider our book, *Introduction to Financial Modelling*, for tips for Best Practice tips that will assist you with all of the above. This is available from all good bookshops (so if you can't find it at a bookshop, it's not a good one!).

SumProduct on Financial Modeler's Corner



SumProduct's very own **Liam Bastick** was recently invited onto the most recent podcast episode of Financial Modeler's Corner by "The FP&A Guy" AKA Paul Barnhurst. Liam and Paul waxed lyrical on a variety of topics including:

the importance of developing problem-solving skills

- how Power BI opens up new dimensions – quite literally! – for financial modelling
- the future of Excel
- why cash is as important as ever it was
- the need to embrace AI.

You can check out the show here:

- Spotify Link: https://open.spotify.com/episode/2JsaFquSUojXFah4SZnGTr?si=z_9yiQr3QGymgX8LGu_QXg
- Apple Link: <https://podcasts.apple.com/us/podcast/excel-problem-solving-strategies-for-financial-modelers/id1693354484?i=1000677461323>
- YouTube Link: <https://youtu.be/zKs1MMS0eyY?si=0tEd1wsB9ipNFiyZ>

IFRS 18 and its Impact on Financial Modelling

For those of you in the Australasian region, you might want to act quickly.



In 1950, the Australasian Reporting Awards ("ARA Awards") were introduced as the Annual Report Awards by a group of business professionals concerned that the annual reports at that time were not adequate for informing stakeholders about the purpose, activities and performance of organisations. Well, it's taken off a little since then.

Celebrating its diamond (75th) jubilee next year, the ARA Awards continue to go from strength to strength as companies strive to benchmark their

annual report against world best practice criteria, be recognised for the quality of their reporting, and also have the opportunity to access expert advice to help improve their reporting.

Now in our second year of sponsorship, SumProduct is a proud and ardent Supporter Partner for these awards: they are open to all private sector, public sector and not-for-profit organisations in Australia, New Zealand and countries in the Asia Pacific region.

Those responsible for preparing the annual report may:

- get great personal satisfaction from achieving an Award
- have a feedback session with a senior ARA adjudicator to find out how to improve their reporting
- attend the gala 75th Year ARA Anniversary Awards Presentation Dinner to be held in Melbourne on Thursday, 29 May next year to receive an Award.

Just be mindful of the closing date: entries close on Friday 6 December 2024, although there is an extended closing date of Friday 31 January 2025 if you meet certain criteria. If the release of your organisation's report may be delayed beyond the closing date, please contact ARA at ara@arawards.com.au to obtain an extension.

Full details may be found here: <https://www.arawards.com.au/>

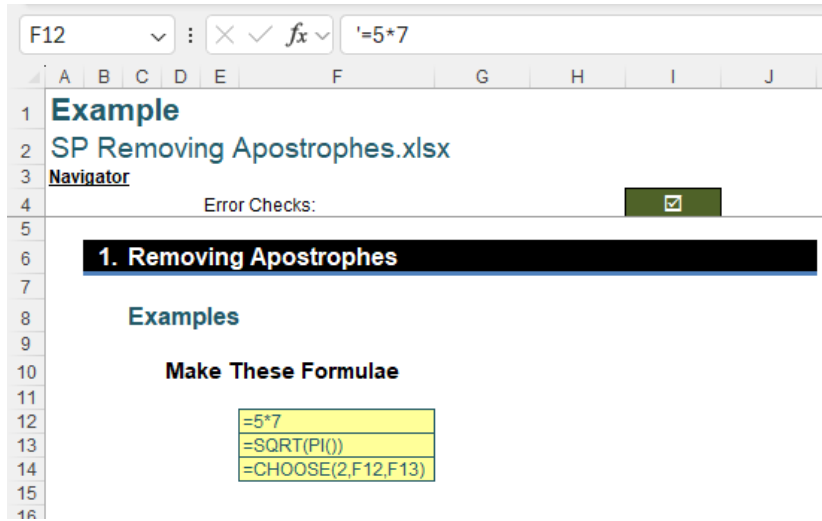
Good luck!

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...

Consider the following example:



In cell **F12** (pictured) we have the “pseudo-formula”:

`'=5*7`

For the purposes of this challenge, the cells **F12:F14** are not formatted as Text. They do not calculate as they are deemed text since they all begin with an apostrophe ('). This month’s challenge is again simple but is a necessary skill for modelling. What is the quickest way to remove all the apostrophes **at the same time** (without using Power Query, VBA or other similar code)?

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

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?*

Here is a contentious one: what constitutes "Best Practice in financial modelling"? Let's see what R2D2 has to say on the matter. Beep! beep! (oh wait a minute, that's Roadrunner...)

This Month's Prompt: What constitutes "Best Practice" in financial modelling and why?

Financial modelling is a critical tool used by businesses, financial analysts and investors to forecast a company's financial performance, evaluate investment opportunities and make strategic decisions. To ensure that financial models are reliable, accurate and useful, it is important to adhere to best practices. These best practices encompass the principles of design, structure, documentation and validation, which together enhance the clarity, efficiency and robustness of financial models.

The design principles now follow.

CONSISTENCY

Maintaining consistency in a financial model is crucial. This includes consistent use of formulae, formats and structures throughout the model. Consistent design makes the model easier to understand and reduces the likelihood of errors.

SIMPLICITY

A financial model should be as simple as possible while still capturing the necessary complexity of the underlying financial situation. Avoid overcomplicating the model with unnecessary details or overly complex formulae. Simplicity enhances both understanding and usability.

TRANSPARENCY

Transparency in a financial model means that all assumptions, inputs and calculations are clearly documented and easy to follow. This allows users to understand how outputs are derived and makes it easier to identify and correct errors.

FLEXIBILITY

A good financial model should be flexible enough to accommodate changes in assumptions, inputs and scenarios without requiring significant rework. This can be achieved by using input cells and avoiding hard-coded numbers within formulae.

Structural best practices should also be considered.

MODULAR DESIGN

Breaking the model into modules or sections, such as inputs, calculations and outputs, can greatly enhance clarity and organisation. Each module should have a specific purpose and be interconnected in a logical manner.

CLEAR LABELLING

All rows and columns should be clearly labelled, making it easy to understand what each number represents. Use of descriptive headings and sub-headings can significantly improve the readability of the model.

COLOUR CODING

Implementing a colour-coding scheme for different types of cells (*e.g.* inputs, calculations, outputs) can help users quickly identify different components of the model. Common conventions include using blue for inputs, black for calculations and green for outputs.

SEPARATION OF INPUTS AND CALCULATIONS

Inputs should be separated from calculations to avoid confusion and reduce the risk of accidental changes to critical formulae. Ideally, all inputs should be placed in a designated input section at the beginning of the model.

There are documentation best practices too.

ASSUMPTIONS

All assumptions used in the model should be clearly documented. This includes explaining the rationale behind key assumptions and providing references or sources where applicable.

VERSION CONTROL

Maintaining a version control system for the financial model can help track changes over time and ensure that the most recent version is always accessible. This can be as simple as including a version number and date in the model's title or file name.

COMMENTS AND ANNOTATIONS

Using comments and annotations within the model can provide additional context and explanations for complex calculations or assumptions. This can be particularly useful for users who are not familiar with the model's intricacies.

Validation best practices are important too.

ERROR CHECKING

Implementing error checks within the model can help identify potential issues. This can include checks for negative values where they are not expected, ensuring that totals add up correctly and using Excel's built-in error checking tools.

SENSITIVITY ANALYSIS

Conducting sensitivity analysis allows users to understand how changes in key assumptions impact the model's outputs. This involves systematically varying assumptions and observing the resulting changes in outputs.

PEER REVIEW

Having a second set of eyes review the model can catch errors that the original creator might have missed. Peer review can be an effective way to ensure the model's accuracy and reliability.

CONCLUSION

Adhering to best practices in financial modelling is essential for creating models that are accurate, efficient and easy to understand. By following principles of design, structure, documentation and validation, financial analysts can build robust models that facilitate better decision-making and provide valuable insights. These best practices not only enhance the quality of the model but also ensure that it can be effectively used and understood by a wide range of stakeholders.

Incorporating these guidelines into your financial modelling process will help you create models that are not only technically sound but also user-

friendly and adaptable to changing circumstances. Whether you are building a model for internal use, presenting it to investors, or using it to guide strategic decisions, following best practices will ensure that your model serves its purpose effectively.

Well it nearly got **CRaFT** right! Send us your comments and ideas for questions to 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'll show you how to protect yourself from losing work in case Excel crashes or is forced to quit without saving your work.

AutoRecover is not a new concept in Excel. It's been an important feature for years. Excel saves a copy of your file automatically as a backup in case Excel freezes or crashes. When you re-launch Excel, you'll get the opportunity to open the backup, known as a recovery file.

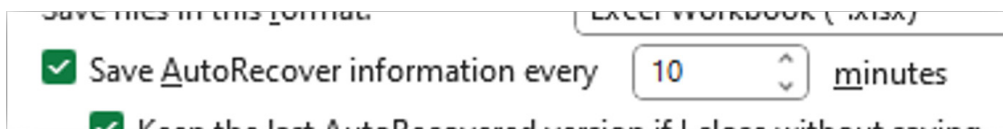
We're Not Talking About AutoSave

To be clear, AutoRecover and AutoSave are not the same, even though they sound similar. AutoSave is where Excel automatically saves your file. AutoRecover is where Excel saves a temporary backup copy of your file. AutoRecover has no impact on the file that you're working on. AutoSave has a direct impact. That said, if your file is being AutoSaved, then it makes AutoRecover a bit less important. Even so, we think it's good to know about how it works.

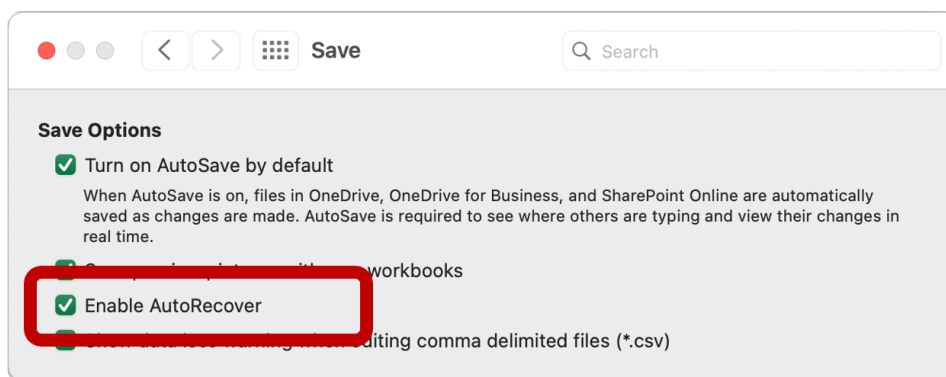
It's Better on Mac

Sometime in 2021, the Excel preference for AutoRecover was changed. Previously, it was just like we see on Windows. You enable AutoRecover to save your information each time a number of minutes goes by. You can set the number of minutes, but the default is 10 minutes. Since a lot of work can be done in 10 minutes, you might prefer setting it to one [1].

This is fine unless the document you're working on takes a long time to save. While Excel is saving the copy, it can seem unresponsive. If Excel becomes unresponsive, even for a few seconds every minute, that can be disruptive.



The AutoRecover preference in Mac no longer has a number of minutes as part of the setting. It's simply enabled or disabled.



Instead of saving at a specific number of minutes, Excel saves the recovery copy of your work as often as possible without causing any disruption. It does this by automatically adjusting the frequency of the AutoRecover save depending on how much time it takes to save the file.

If you're working on a small file that only takes a few milliseconds to save, then AutoRecover will save its copy of the file very frequently. If the file takes several seconds to save, then AutoRecover will save its copy less frequently, so as not to interfere with your work.

In general, AutoRecover seems better on Mac for several reasons:

- there is no guesswork about how many minutes to choose, and there's no chance to leave it at 10 minutes, in which time you can do a lot of work and potentially lose a lot of work if there's a problem
- it is optimised for each file and adjusts as you work. This is better than having one setting that affects all documents the same.

Will This Come to Windows?

We don't know, but we'd be happy if it does. Based upon how it works, it seems likely to reduce the amount of lost work, and it also seems likely that Microsoft is keeping track of how successful it is on Mac. They've been very focused on data in recent years, so if it's shown to reduce lost work, hopefully they will make it the same on both platforms.

We'll continue next month...

Visual Basics

*We thought we'd run an elementary series going through the rudiments of Visual Basic for Applications (VBA) as a springboard for newer users. This month, we consider ways to make the **Case** statement more flexible.*

In last month's newsletter, we introduced the case statement. We noted that it is like a **VLOOKUP** – it searches through a list to find the value that matches, then it reports back on it.

Numeric conditions

If you want the value to be, say, greater than **X**, you can adapt the **Case** expression to provide a test for it to be correct. In addition to checking if the value is exactly equal to the expression, there are two additional options:

1. **Case X To Y**
2. **Case Is >= X.**

In the case of **X To Y**, it will search for values that will fall between (and including) the values **X** and **Y**. In the case of **Is >=X**, other comparison operators (e.g. =, <>, <, <=, > and >=) can be used to provide that restriction.

Therefore, we may use an example such as the following:

```
Select Case margin  
    Case Is > 0.7  
        MsgBox ("Great result!")  
    Case 0.3 To 0.7  
        MsgBox ("Not a bad result")  
    Case Is < 0.3  
        MsgBox ("Need to do better")  
End Select
```

No statements match

Last time, our leap year example had a single case to check if the value was zero [0], rather than 365 or 366 days. However, if a value of 200 was entered, it would not provide a correct answer. This is where we need to use **Case Else**.

Case Else allows us to specify what happens if none of the other cases are true. Going back to the **VLOOKUP** analogy, this would normally result in an #N/A error. However, with **Select Case**, we can provide a default result if nothing else matches. Thus, we could amend the code as follows:

```
Select Case daysinyear  
    Case 365  
        MsgBox ("This is a normal year")  
    Case 366  
        MsgBox ("This is a leap year")  
    Case Else  
        MsgBox ("Please enter a valid number")  
End Select
```

This means that anything that doesn't match 365 or 366 will prompt us with a message box telling us that our value is invalid.

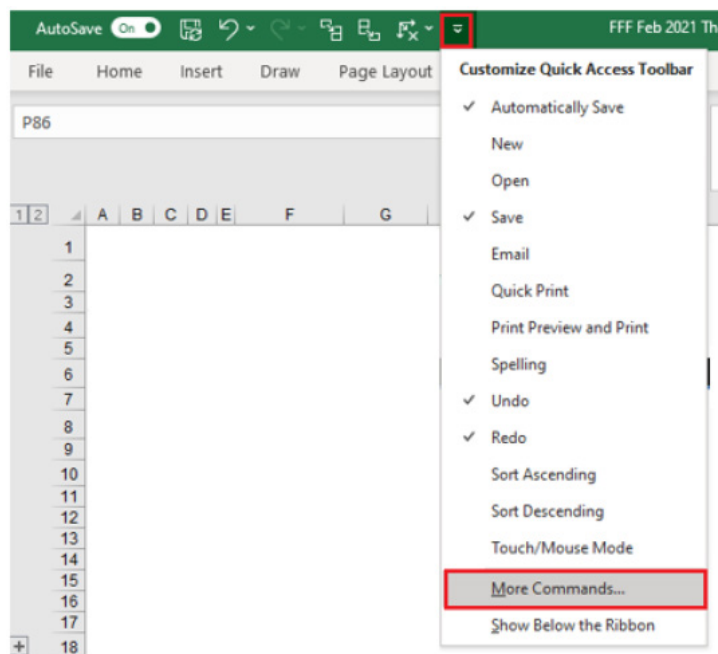
More next time.

Charts and Dashboards

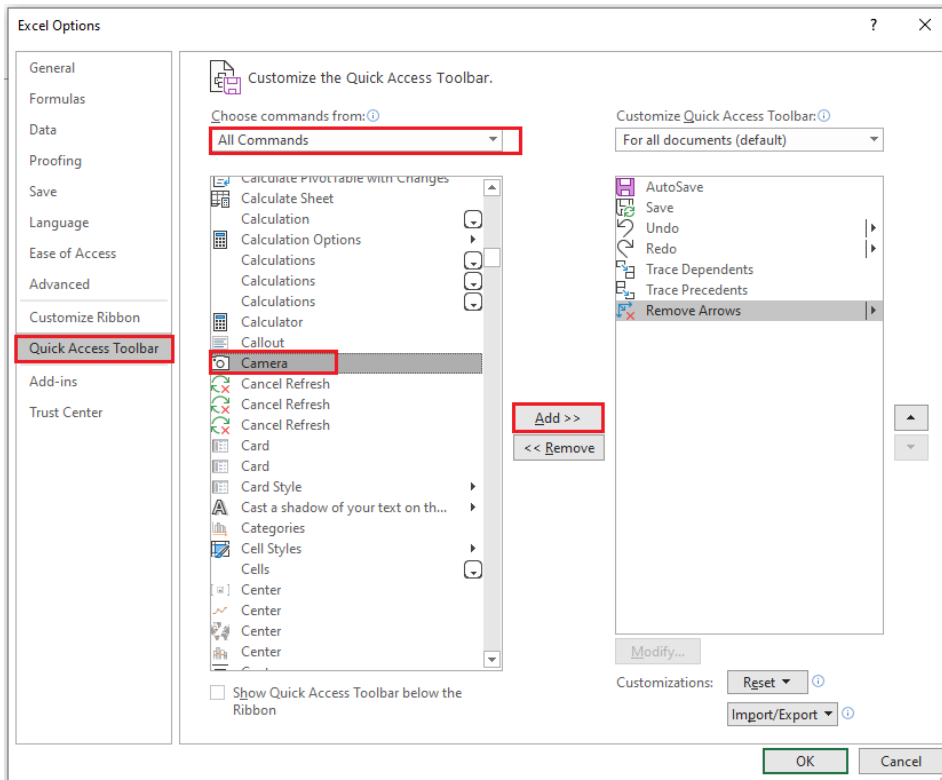
It's time to chart our progress with an introductory series into the world of creating charts and dashboards in Excel. This month, we consider a handy tool when creating dashboards – namely, Excel's Camera tool.

The Camera tool allows us to take a dynamic picture of a cell or range of cells *e.g.* data, table or graph. What we mean by 'dynamic' here is that the snapshot is pasted as a linked picture, so that when the source range of cells changes in value or formatting the copied image will also be updated automatically. These dynamic images may also be copied and pasted into Word and PowerPoint documents too.

To get the Camera tool, in the Quick Access Toolbar drop-down list, choose 'More Commands...'.
(Note: The 'More Commands...' option is highlighted in red in the screenshot below.)

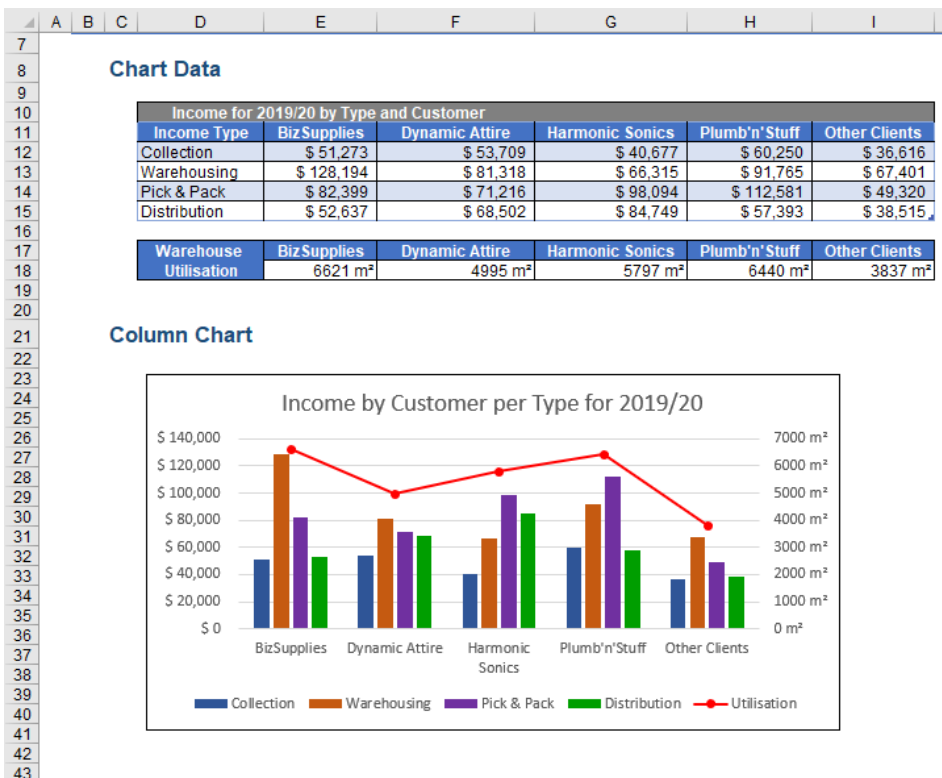


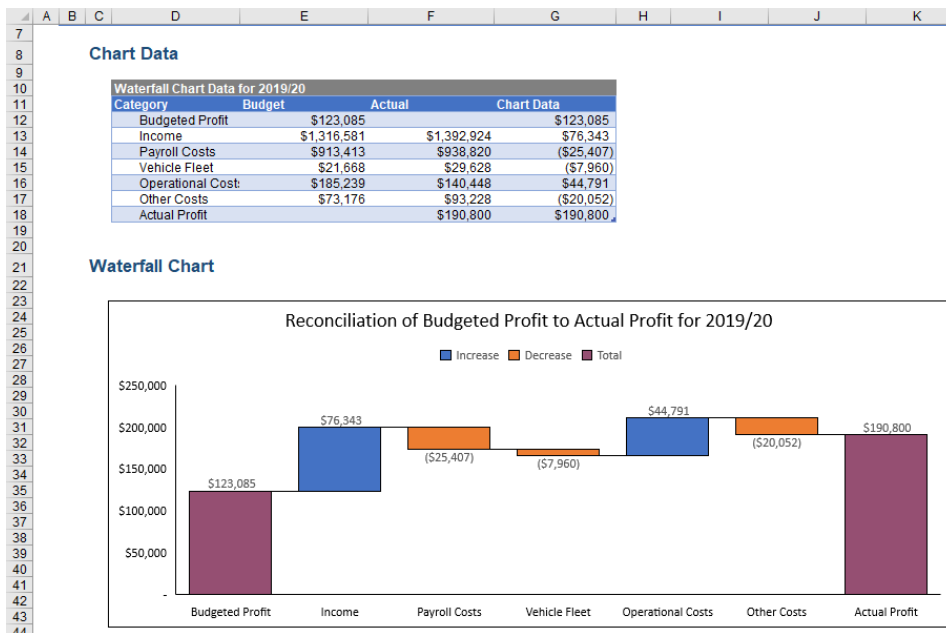
An 'Excel Options' dialog will appear. Under the 'All Commands' list, for easiest access, add Camera to the Quick Access Toolbar and click OK.



When we are creating a dashboard in Excel, sometimes we have the difficulty of fitting different charts and tables together. The Camera tool comes in handy here as we can easily get a snapshot of the chart or the data table and place it on the dashboard in the interim.

For example, here, there are two sets of data tables and charts on separate sheets that we want to bring together into one single dashboard:

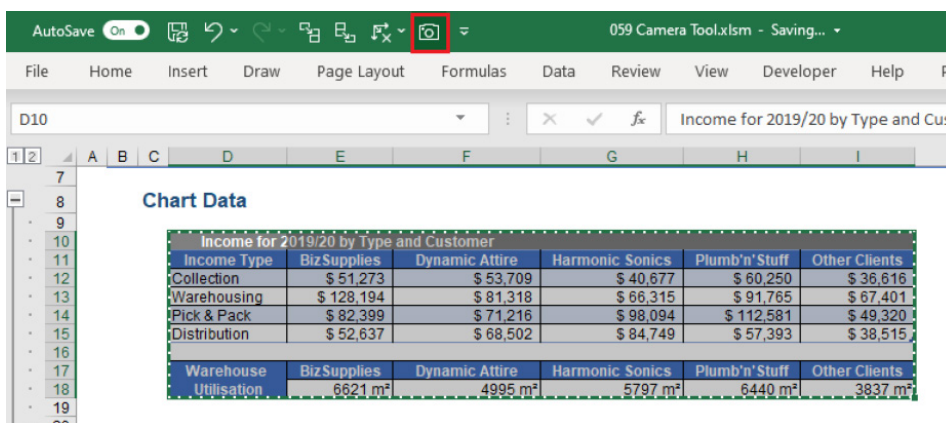




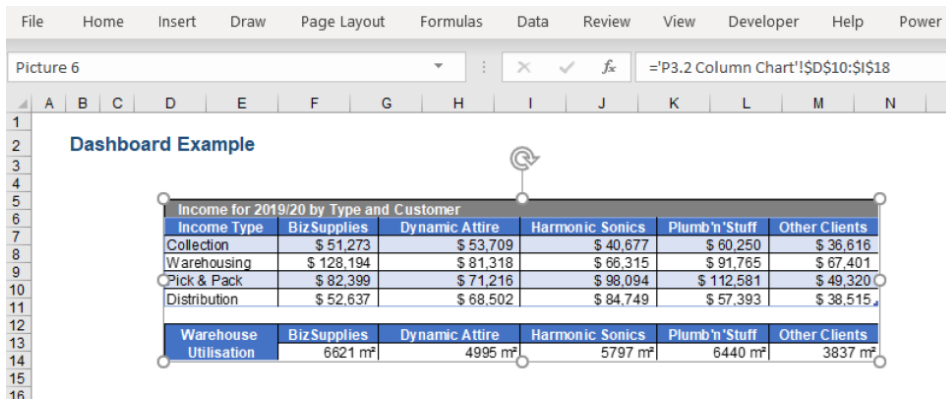
There may be some reason not to recreate the tables on the dashboard (e.g. you may not want end users to change the data, but you still wish them to view it). Therefore, in this case, we can use the Camera tool to take the snapshots of the data tables and charts to paste and rearrange

them on the dashboard. This way, when the data in the source tables change, the dashboard is automatically updated.

To take a snapshot of the data table using the Camera tool, select the range **D10:18** and click on the Camera icon on the Quick Access Toolbar:



then go to the sheet where we are setting up the dashboard and click somewhere on the sheet to get the image. We may notice from the Formula bar that this is a linked image to a range of cells.



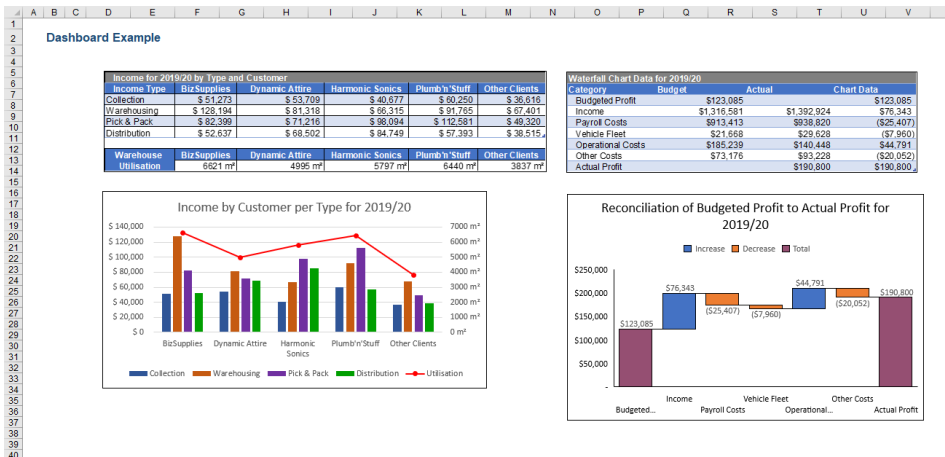
Notice that if the data in the source chart data changes,

Income for 2019/20 by Type and Customer						
Income Type	BizSupplies	Dynamic Attire	Harmonic Sonics	Plumb'n'Stuff	Other Clients	
Collection	\$ 51,373	\$ 53,709	\$ 40,677	\$ 60,250	\$ 36,616	
Warehousing	\$ 128,294	\$ 81,318	\$ 66,315	\$ 91,765	\$ 67,401	
Pick & Pack	\$ 82,499	\$ 71,216	\$ 98,094	\$ 112,581	\$ 49,320	
Distribution	\$ 52,737	\$ 68,502	\$ 84,749	\$ 57,393	\$ 38,515	
Warehouse Utilisation	6721 m ²	4995 m ²	5797 m ²	6440 m ²	3837 m ²	

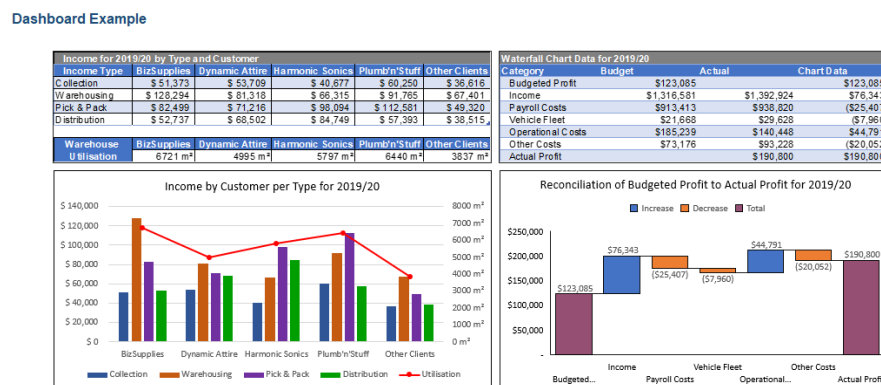
the details in the image also change because it is linked together:

Income for 2019/20 by Type and Customer						
Income Type	BizSupplies	Dynamic Attire	Harmonic Sonics	Plumb'n'Stuff	Other Clients	
Collection	\$ 51,373	\$ 53,709	\$ 40,677	\$ 60,250	\$ 36,616	
Warehousing	\$ 128,294	\$ 81,318	\$ 66,315	\$ 91,765	\$ 67,401	
Pick & Pack	\$ 82,499	\$ 71,216	\$ 98,094	\$ 112,581	\$ 49,320	
Distribution	\$ 52,737	\$ 68,502	\$ 84,749	\$ 57,393	\$ 38,515	
Warehouse Utilisation	6721 m ²	4995 m ²	5797 m ²	6440 m ²	3837 m ²	

Next, we will repeat the process to get the other three data table and chart items into the dashboard page. Alternatively, for the charts, we can just copy and paste them to the dashboard. We will have an initial dashboard like the one below.



We will follow the best practices for the dashboard layout to get the complete dashboard by resizing and aligning the tables and charts.

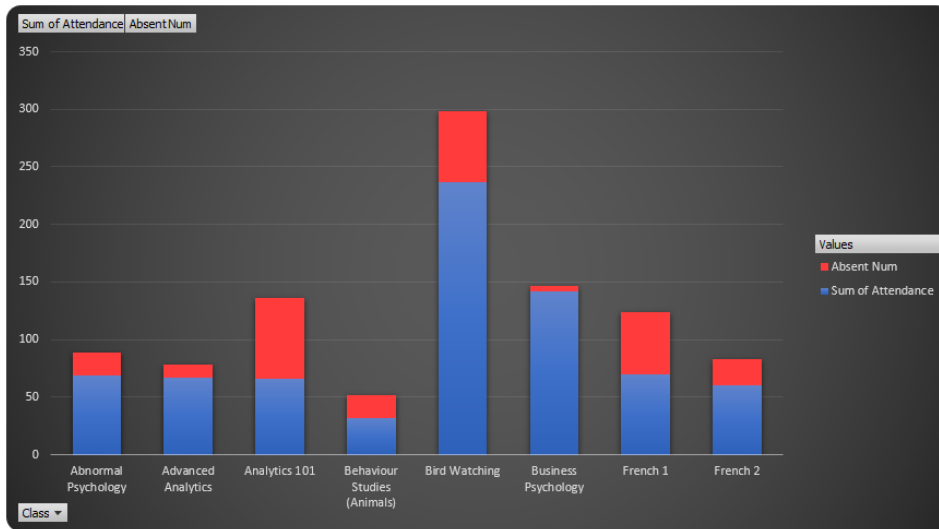


More next time.

Charts and Dashboards

We continue our series on the Excel COM add-in, Power Pivot. This month, we look at how to create a PivotChart that highlights the number of absent students in university classes (say).

Here is the scenario: we are a teaching assistant to a professor and the professor has tasked us with creating a Stacked Bar chart that will highlight the number of students that are absent from several classes:



In this stacked bar chart, the red segments of the chart illustrate the total number of absent students in each class and the blue segments illustrate the number of students present. The red and blue segments combined add up to the total number of students enrolled in each class.

The professor has given us two data Tables, one has the total number of students enrolled in each class:

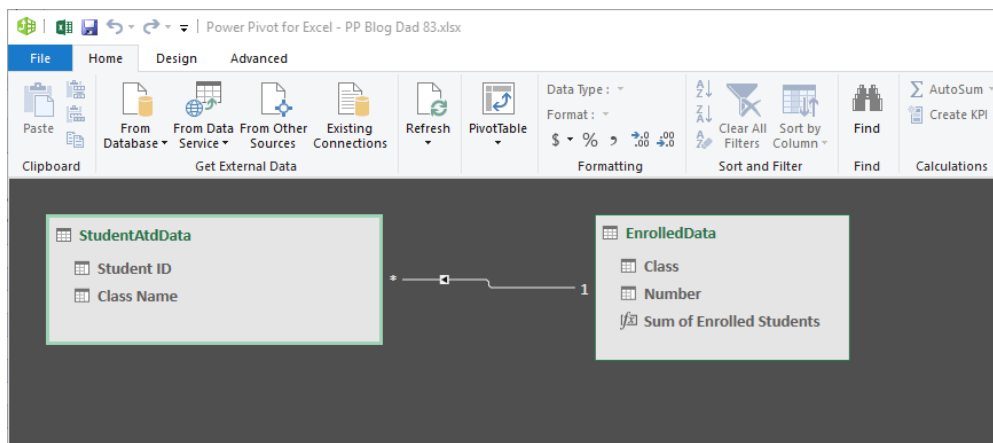
Class	Number
French 1	124
French 2	83
Advanced Analytics	78
Analytics 101	136
Abnormal Psychology	89
Business Psychology	147
Behaviour Studies (Animals)	52
Bird Watching	298

The other data table contains 700+ rows of attendance data. It is organised in such a way that for each student only appears on this list if they had attended the class (i.e. the screenshot is not exhaustive):

Student ID	Class Name
10022	Business Psychology
10023	Behaviour Studies (Animals)
10024	Analytics 101
10025	Business Psychology
10026	Behaviour Studies (Animals)
10027	Bird Watching
10028	Analytics 101
10029	Bird Watching
10030	Bird Watching
10031	Abnormal Psychology
10032	French 1
10033	Abnormal Psychology
10034	Business Psychology
10035	Bird Watching
10036	Behaviour Studies (Animals)
10037	Abnormal Psychology
10038	Bird Watching
10039	Abnormal Psychology
10040	Abnormal Psychology
10041	French 1
10042	Bird Watching
10043	French 1
10044	Abnormal Psychology
10045	French 1
10046	French 1
10047	Business Psychology
10048	Behaviour Studies (Animals)
10049	Behaviour Studies (Animals)
10050	Business Psychology

You must be thinking, “surely we can use **SUMIF** function in Excel to compile the numbers then compare them in a chart” – well, yes... but let’s do it in **DAX**.

The first step is to add both data tables into our data model. We have to create a relationship between the two Tables using the **Class** and ‘**Class Name**’ columns as the common keys between both tables:



The issue now is, how do we illustrate the number of students that are skipping class? There are two [2] methods:

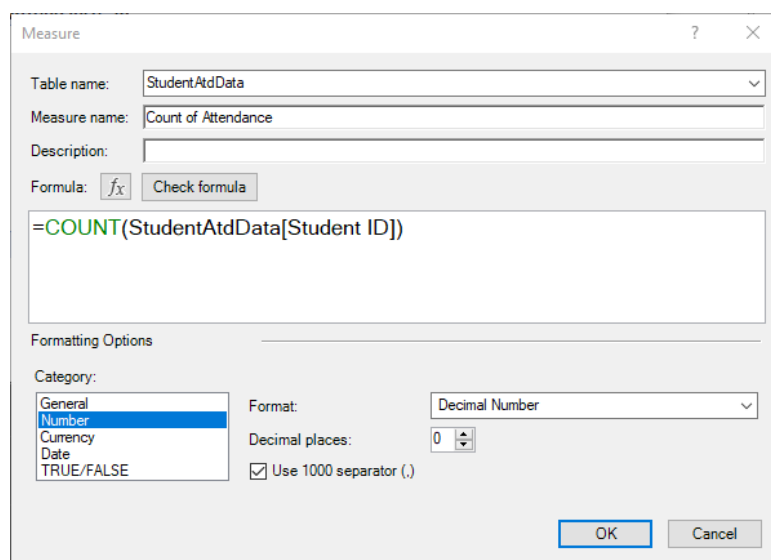
1. We can add a custom column into the table called **Attendance** and populate the entire column with ‘1’ values

	[Attendance]		f_x	=1
	Student ID	Class N...		Attendance
1	10022	Business Psyc...		1
2	10023	Behaviour Stu...		1
3	10024	Analytics 101		1
4	10025	Business Psyc...		1
5	10026	Behaviour Stu...		1
6	10027	Bird Watching		1
7	10028	Analytics 101		1
8	10029	Bird Watching		1

From there we can sum the attendances.

2. If, for whatever reason, you are unable to do that (e.g. not enough memory), we can create a measure to count the number of students going to each class with the **COUNT** function:

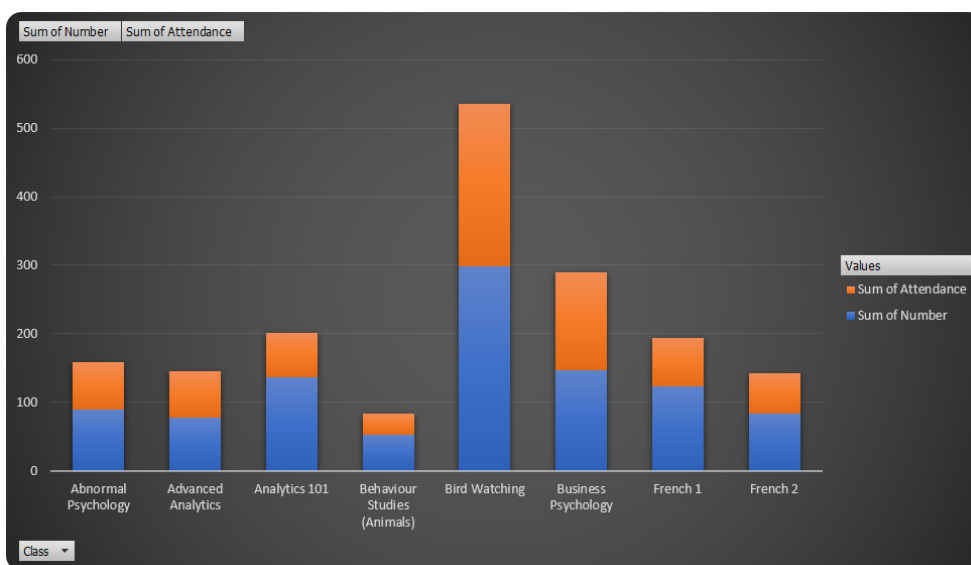
=COUNT(StudentAttdData[Student ID])



Both methods yield the same result:

	A	B	C	D	E
1					
2		Row Labels	Sum of Enrolled Students	Count of Attendance	Sum of Attendance
3		Abnormal Psychology	89	69	69
4		Advanced Analytics	78	67	67
5		Analytics 101	136	66	66
6		Behaviour Studies (Animals)	52	32	32
7		Bird Watching	298	237	237
8		Business Psychology	147	142	142
9		French 1	124	70	70
10		French 2	83	60	60
11		Grand Total	1,007	743	743
12					

Adding the attendance field to our PivotChart yields:



As it stands, the Stacked Bar chart is displaying the attendance of students in orange and the number of enrolled students in blue. We want the orange segment of the bars to illustrate the number of students missing from each class. To do that, we're going to have to create some measures:

=SUM(EnrolledData[Number])

Measure ? X

Table name:

Measure name:

Description:

Formula:

Formatting Options

Category:

Format:

Decimal places:

Use 1000 separator (,)

Once we have the sum of enrolled students, we can create the number of absent students in each class:

$$=[\text{Sum of Enrolled Students}]-\text{SUM}([\text{Attendance}])$$

Measure dialog box configuration:

- Table name: StudentAttdData
- Measure name: Absent Num
- Formula: $=[\text{Sum of Enrolled Students}]-\text{SUM}([\text{Attendance}])$
- Format: Decimal Number
- Decimal places: 0
- Use 1000 separator (.):

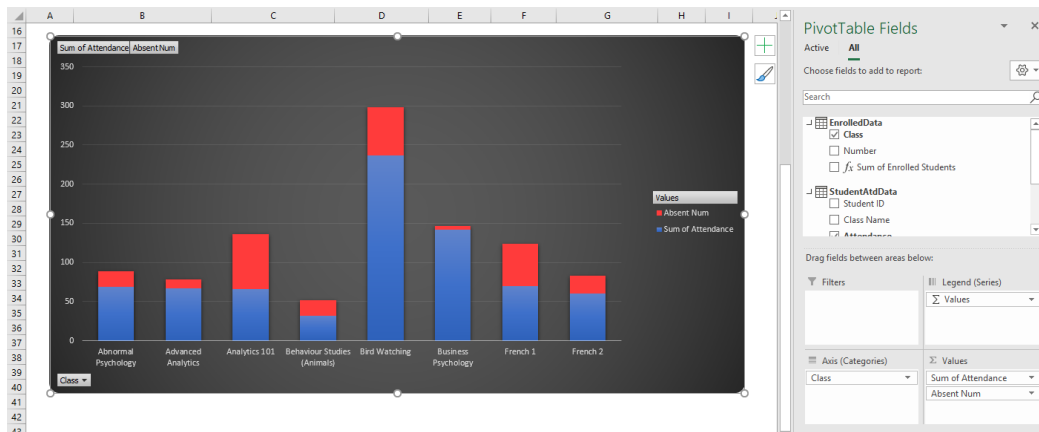
or (if you used the **Count of Attendance** measure):

$$=[\text{Sum of Enrolled Students}]-[\text{Count of Attendance}]$$

Measure dialog box configuration:

- Table name: StudentAttdData
- Measure name: Absent Num 2
- Formula: $=[\text{Sum of Enrolled Students}]-[\text{Count of Attendance}]$
- Format: Whole Number
- Use 1000 separator (.):

With the **Absent Num 2** measure calculating the number of students that were absent in each class, we can create the following Stacked Bar chart (we have also decided to use red instead of orange, because it is more incriminating):

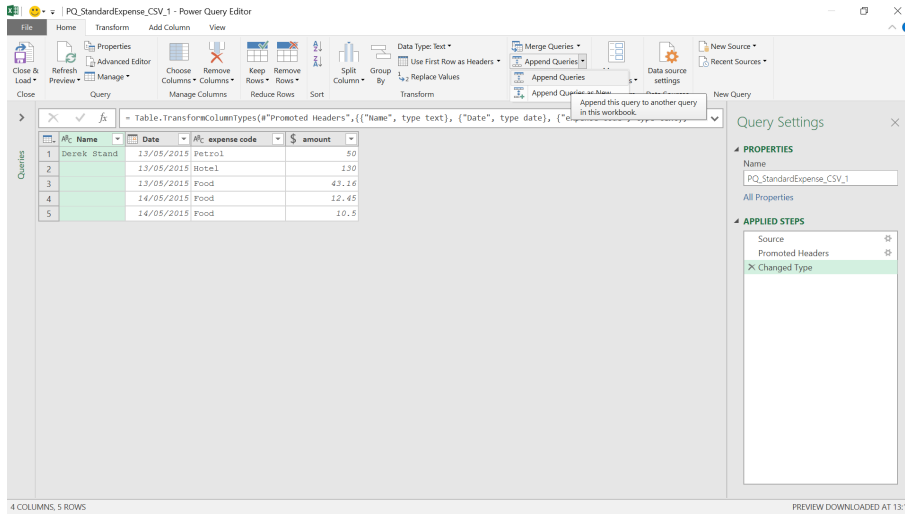


That's it for this month; more next time.

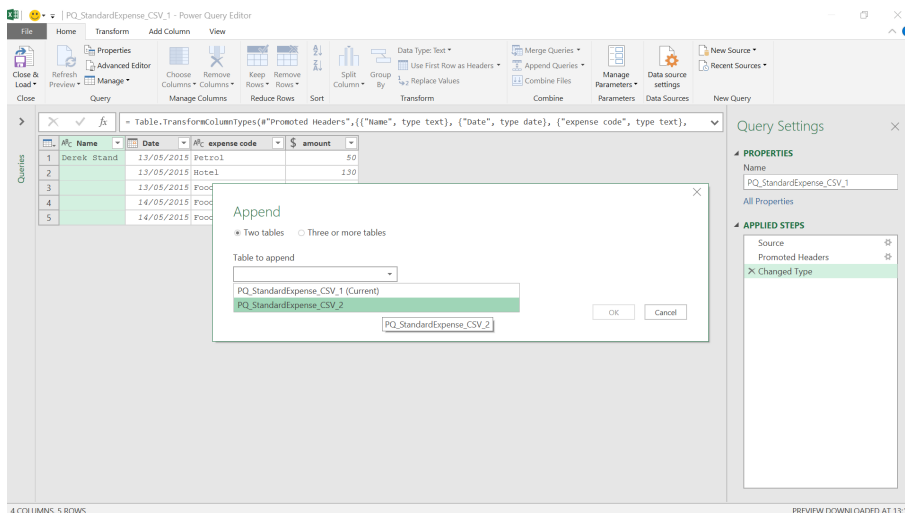
Power Query Pointers

Each month we'll reproduce one of our articles on Power Query (Excel 2010 and 2013) / Get & Transform (Office 365, Excel 2016 and 2019) from www.sumproduct.com/blog. If you wish to read more in the meantime, simply check out our Blog section each Wednesday. This month, we review when to merge or append when combining data.

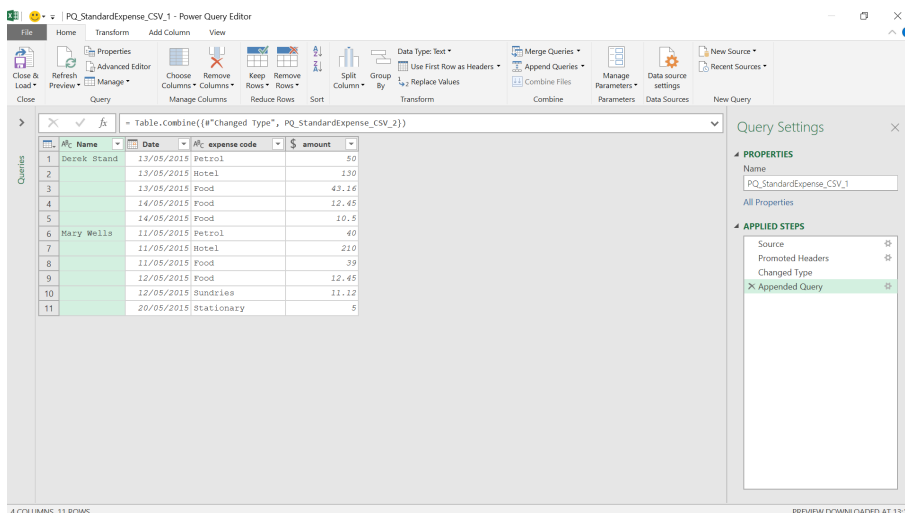
When we have separate queries that are very similar or related, we usually don't want to waste time cleaning them up individually when we may join them together and deal with a big query in one go.



We have a standard expense sheet from one of our imaginary salespeople, Derek, which we plan to clean up ready for reports to be extracted. However, we have a very similar sheet from Mary, so it makes sense to append her sheet and clean them up together.

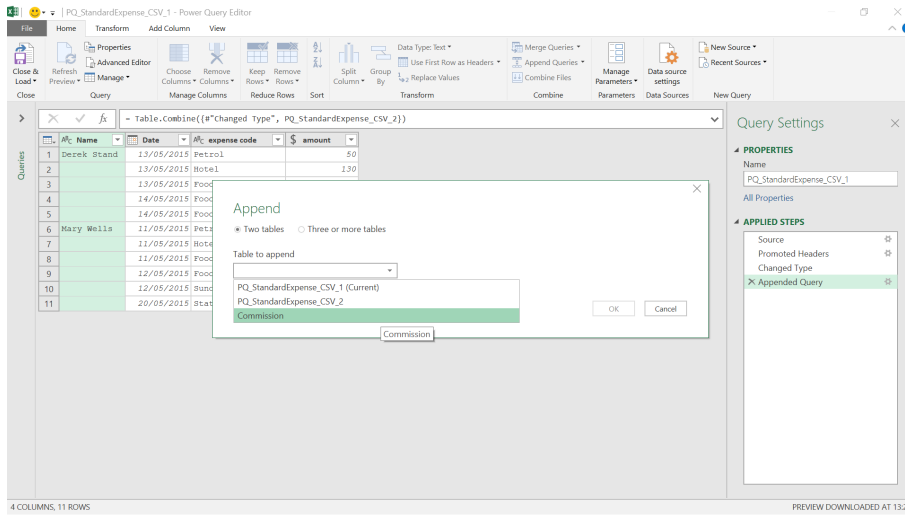


Let's append the query containing Mary's data.

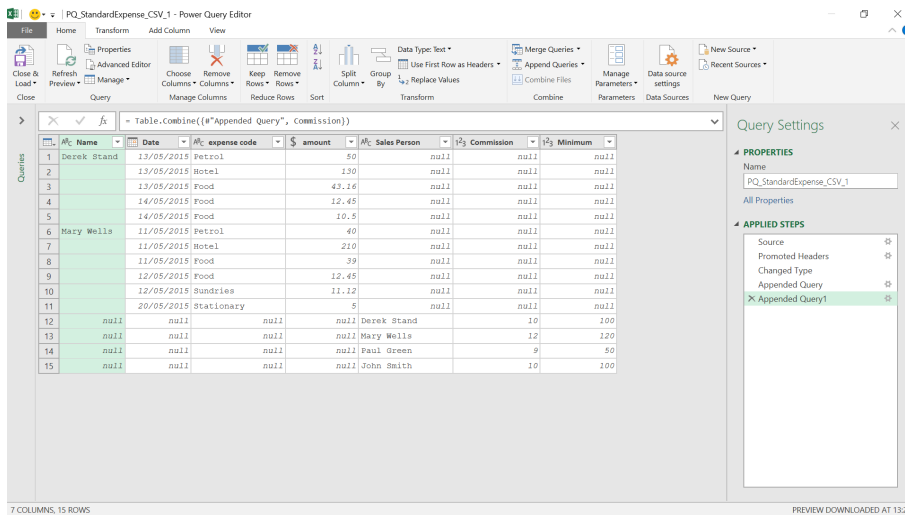


We may repeat this (or append more than one table at a time), as long as my data is similar. In fact, if I have a lot of similar tables in one folder, we can append them all. Appending will just keep adding rows. There are no checks for duplication. Therefore, if we accidentally add Derek's sheet twice, then the rows for Derek will simply appear twice.

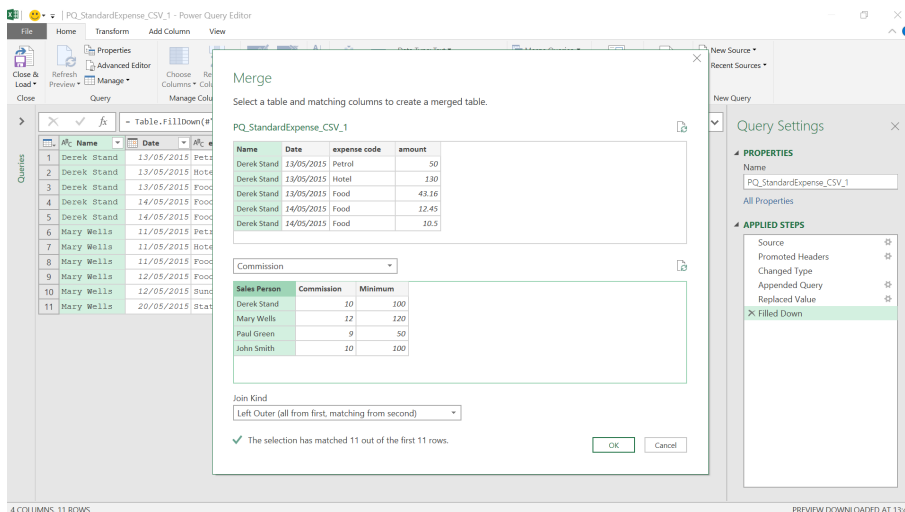
If we wish to add more data that is related, but not similar data to this query – for example, the commission information for my salespeople – then append will not help us.



We can select the 'Commission' query, but appending it produces unsatisfactory results:

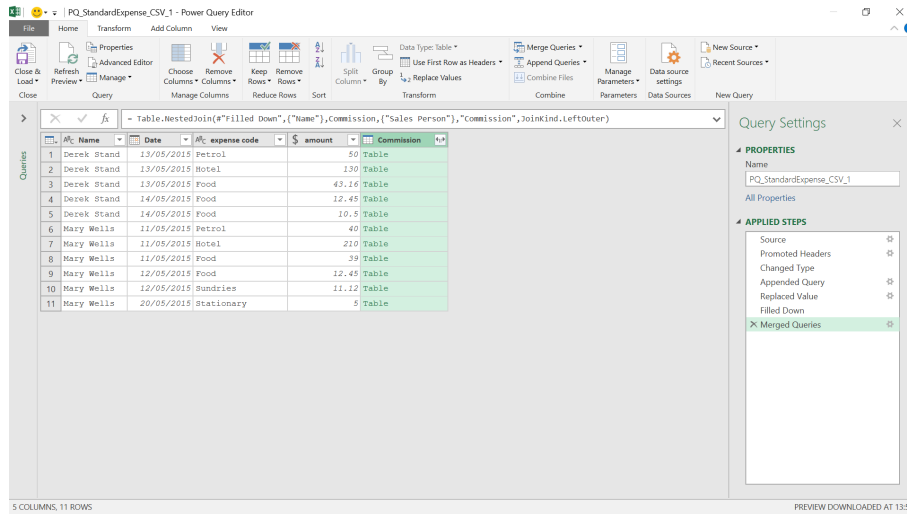


That's not what we were looking for. When appending, Power Query will only add data to the same column if the column names are exactly the same. In this case, not only are the column names different, but so is the data. To combine data that is related, but not similar, we need to merge our queries.

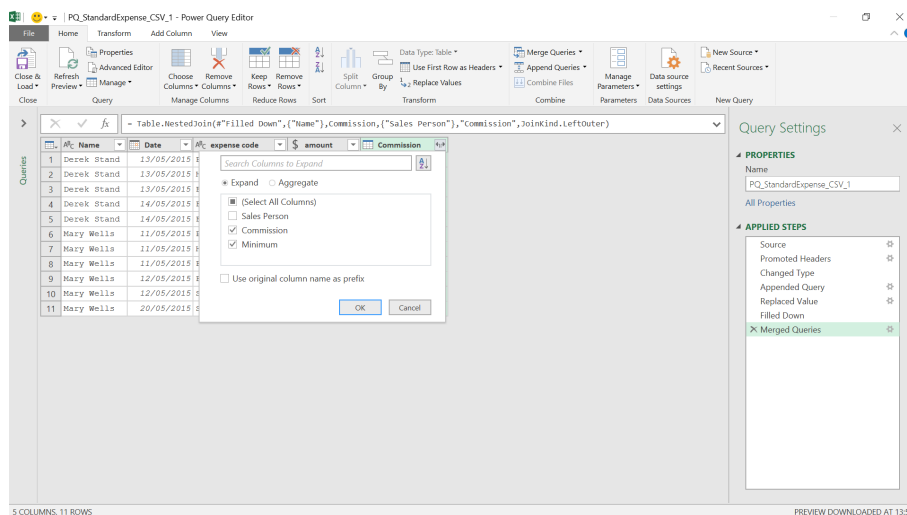


In order to merge, Power Query needs to know how to join my data together, *i.e.* the common link or key. In this case, it is the name of the salesperson. This is why we have cleaned up the data in our query to populate each row in the **Name** column.

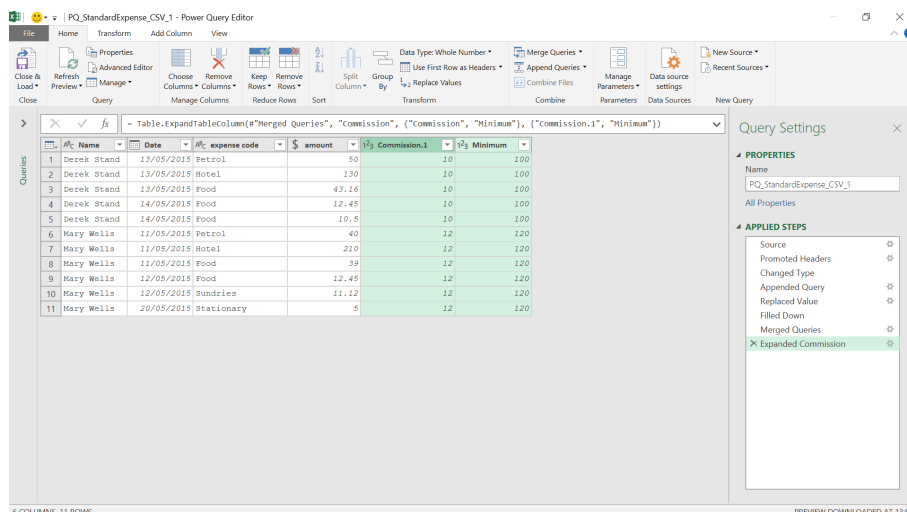
If we take the default join, the left join, then we will obtain everything from the first query: the expenses and the matching rows from the second query, which should give us the commission for Derek and Mary, but not the other salespeople. This is correct, as I don't need to know the commission for the other salespeople in this particular query.



The data from the 'Commissions' query has been added as a table, so we need to expand it using the icon next to the title of the column **Commissions**:



I only need the **Commission** and the **Minimum** fields (and we definitely don't need the 'original name as a prefix'!).



We now have all the data together, allowing us to go on to calculate monthly commission amounts.

In summary:

- if we want to combine queries that contain similar data, we may use Append, but we must make sure that the column names match
- if we want to add related data, where one or more of the columns matches, and some of the data in those columns can be used as a key to join the data together, then we may use Merge and pick the best join for our purposes.

In any case, if we find that we have picked append when we shouldn't, or we chose the wrong join, we may simply delete that step and try again! The data that is extracted to Power Query is still safe and unchanged in its original location.

Until next month.

Power BI Updates

These latest updates see the introduction of Copilot in Power BI Mobile applications, allowing you to use AI to explore data on the go. There is also a new Path layer for the Azure Map visual, making geospatial analysis more powerful by helping you easily map connections between multiple

locations. Microsoft also announced the OneLake catalog (*sic*), the latest evolution of the OneLake data hub, which offers a unified platform for data engineers, data scientists, analysts and decision-makers to browse, manage, and govern data all from one intuitive interface.

The full list of updates is as follows:

Copilot and AI

- Copilot in Power BI mobile apps (Preview)
- Copilot summaries in subscriptions (Preview)

Reporting

- Path Layer for the Azure Map visual
- Visual calculations (Preview)
- Small Multiples for the new card visual (Preview)
- New visual: text slicer (Preview)

Modelling

- Define new measures in DAX query view quick queries
- Metric sets: a new era of metric management in Fabric (Preview)
- Announcing performance improvements for models with calculation groups and format strings in Excel
- DLP policies restrict access action for semantic models (Preview)
- Semantic modelling in Visual Studio Code with the new TMDL extension (Preview)

Data Connectivity

- Snowflake connector updates

Developers and APIs:

- Fabric Git: TMDL format for semantic model export
- Semantic model client library updates

Visualisations

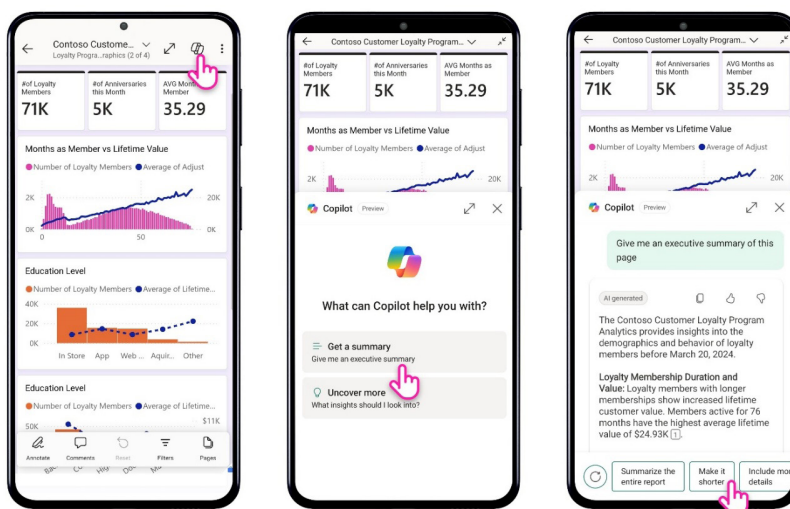
- KPI by Powerviz
- Zebra BI Tables 7.3
- Waterfall PRO by ZoomCharts: interactive waterfall visual for financial data
- Lollipop Bar chart by Nova Silva
- Sales Velocity chart
- New book: *Data Visualization with Microsoft Power BI*

Other

- Support for Power BI language settings when a paginated report is viewed on the Power BI Service.

Let's look at each in turn.

Copilot in Power BI mobile apps (Preview)



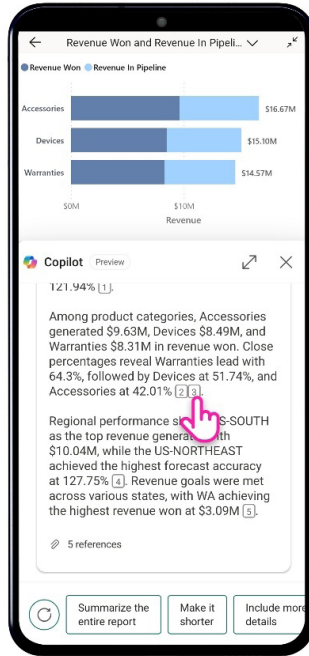
This new feature allows AI to offer a quick and simple way to dive into your data on the go.

With Copilot in Power BI mobile applications, you no longer need to analyse data in silo. Copilot provides report summaries and insights, allowing you to make informed decisions anytime and anywhere.

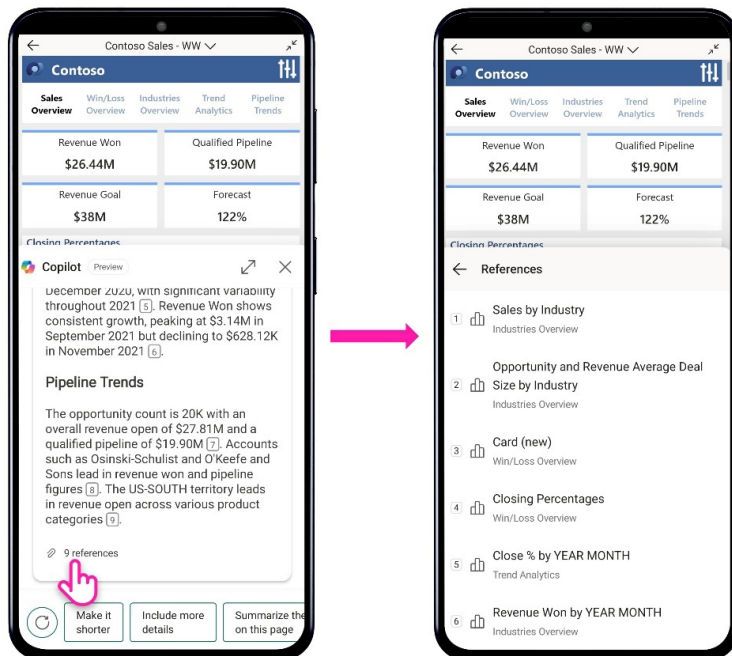
To start using Copilot on your mobile app, simply tap the Copilot button located in the report header (that meets Copilot requirements in Power BI) and choose whether you want to get a summary or to look into

insights. Copilot will deliver a response based on your request. You can then copy and share the response or keep interacting with Copilot by choosing from the suggestions at the bottom. These suggestions can help you tweak the response or create new requests.

The summary is also a useful starting point for exploring your data. Once you receive the summary, tapping on any data reference in the summary will get you a closer look at the relevant visual. The visuals are interactive, and you can dive deeper by long-tapping for tooltips, drilling down or up, or exploring more of your data as needed.

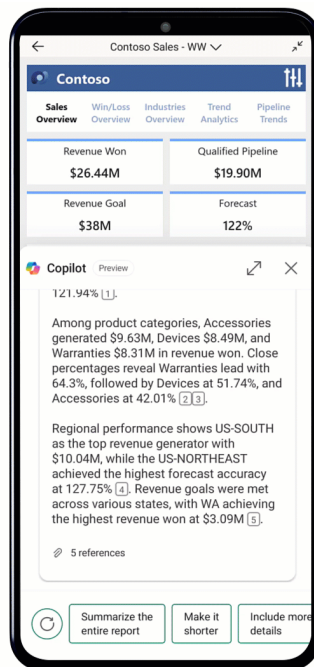


Tapping on references icon will open the reference list, where all the referenced visuals are displayed with their relevant names and pages. This is very useful for easily going over all the visuals selected by Copilot from one place.

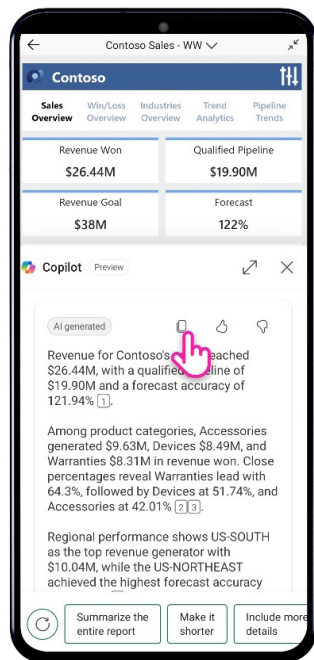


When you're ready for more exploration with Copilot, you can select a suggested prompt from the bottom of the screen. These suggestions can include additional prompts like "Find insights" and "Summarise entire report" or summary adjustments like "Make it shorter" or "Make

it formal". You can view more suggestions by tapping on the refresh button. Each prompt you select should provide an answer tailored to your context and needs.



Once you have the summary you need, you can use the Copy option to copy and paste it into an email, Teams chat or any other platform. Along with the summary, a link to the report is also provided, enabling the recipient to access further details by opening the report directly from that link.



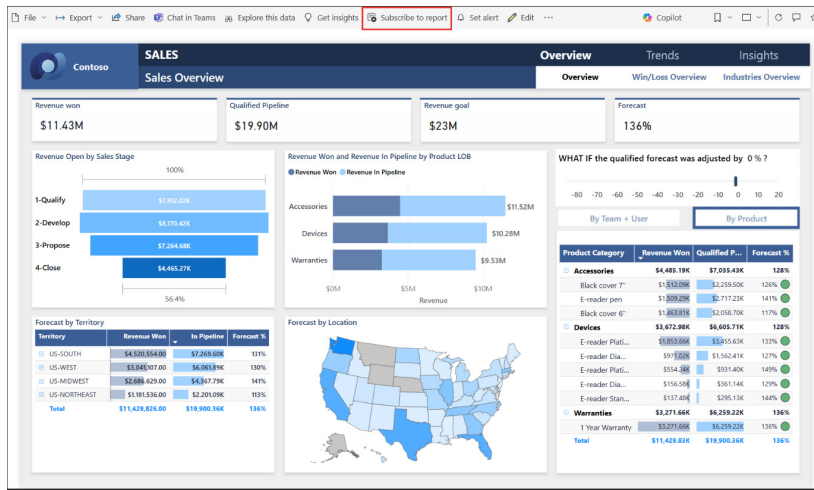
To view the report again, you can close the Copilot chat. The chat history will stay intact when you close the chat and navigate to other pages, as long as you don't exit or refresh the report.

Copilot summaries in subscriptions (Preview)

If you need to extract insights from Power BI report images in your email or quickly digest a summary of your Power BI report, you might wish to subscribe to Copilot summaries for Power BI reports. This feature is available with Standard subscriptions and for reports in a Copilot-eligible capacity.

You may set up the Copilot summaries for Power BI reports that you wish to subscribe to as follows:

1. Select the 'Subscribe' option from the Ribbon for the Power BI report that is of interest



2. Select 'Standard Subscription'

← ×

New subscription

Keep track of your data by subscribing to this report.

🔍 The sensitivity label on this report does not appear in emails.

[Manage all](#) 📄

Choose a subscription type

Standard

Generate and deliver one report. 📧

Dynamic per recipient

Connect to a dataset and send a unique view of the report to each recipient. 👥

3. Subscribe to the report

Attach full report

None ▼

None

📄 PDF (.pdf)

📄 Microsoft PowerPoint (.pptx)

Repeat *

- Add a Copilot summary to the email delivered by the subscription. If you are eligible, your subscription will receive the Copilot summary by default

Subscriptions



Keep track of your data by subscribing to this report.

The sensitivity label on this report does not appear in emails.

Manage all

3 | 10 | 100

Time zone *

(UTC-08:00) Pacific Time (US and Canac

Emails will be sent daily at 03:15 PM (UTC-08:00) Pacific Time (US and Canada) starting Thursday, October 24, 2024.

Summary by Copilot (preview)

Include an AI-generated summary, which will update according to the latest data. Preview a sample of the summary to see what it may look like.

Copilot uses AI. Mistakes are possible. [Read terms](#)

Preview summary



- You can 'Preview summary' to view a sample of the suggested summary for review

The screenshot shows the 'Subscriptions' configuration interface. The main dialog is for 'Sales Overview' and includes fields for 'Subscriber name', 'Recipients', 'Attach full report', 'Send to' (Email), 'Scheduled date and time' (Start date: 10/20/2024, End date: Select a date), 'Repeat' (Daily), and 'Scheduled time' (03:15 PM). The 'Summary by Copilot (preview)' option is checked, and the 'Preview summary' button is highlighted with a red box. A 'Preview a sample summary by Copilot' window is open, showing a sample summary with sections: 'Win/Loss Ratio Overview' (Top Performers: Dan Jump with 82.81% Close Rate), 'Trend Analytics' (Fluctuations in Revenue: Highest single day revenue won: \$161.37K on 2021-07-27), and 'Days to Close Insights' (Top Performers by Days to Close: Jeff Hay with the shortest average Days to Close of 85.69 days).

- Test your subscription by selecting 'Send Now' after you Save the subscription.

Sales Overview - Copilot

(Preview) Summary from Copilot (generated on October 25, 2024 18:20 UTC)

Regional Sales Sample Report: Key Insights

**Sales Overview:

- Qualified Revenue in Pipeline stands at \$19.90 million .
- Revenue Won is \$11.43 million .
- The Forecast Percentage is notably high at 136.22% .
- The Revenue Goal is set at \$23 million .

Revenue Open by Sales Stage:

- Highest open revenue is in the Develop stage with \$8.17 million .
- The Qualify stage has \$7.91 million while Close has the lowest at \$4.47 million .

Product Line Overview:

- Accessories lead with \$4.49 million won and \$7.04 million in the pipeline .
- Devices show \$3.67 million won and \$6.61 million in the pipeline .
- Warranties display \$3.27 million won and \$6.26 million in the pipeline .

Revenue and Forecast by Product:

- Accessories: \$4.49M won, \$7.04M pipeline, 128.01% forecast .
- Devices: \$3.67M won, \$6.61M pipeline, 128.48% forecast .
- Warranties: \$3.27M won, \$6.26M pipeline, 136.16% forecast .

Territory Insights:

- US-SOUTH: \$4.52M won, \$7.27M pipeline, 131% forecast .
- US-WEST: \$3.04M won, \$6.06M pipeline, 130.04% forecast .
- US-MIDWEST: \$2.69M won, \$4.37M pipeline, 141.09% forecast .

State-Specific Revenues:

- WA: \$1.22M won, \$2.49M pipeline .
- TX: \$1.18M won, \$1.71M pipeline .
- CA: \$930.87K won, \$1.65M pipeline .

Calculated Insights:

- Accessories had the highest revenue won at \$4.49M .
- Warranties had the lowest revenue won at \$3.27M .
- There is a positive correlation between Revenue Won and Revenue In Pipeline .
- Revenue Divergence for Warranties: \$2.99M more in pipeline than won .

Created by AI. Mistakes are possible. [Read terms](#)

[Open report in Power BI >](#)



You're receiving this email because you subscribed to the 'Sales Overview' page of the 'Regional Sales Sample' report. The image above was generated on October 25, 2024 18:20 UTC.

[Manage subscription >](#)

Did you find this email helpful? [Yes](#) [No](#)

[Privacy Statement](#)

Microsoft Corporation, One Microsoft Way, Redmond, WA 98052



This feature will roll out gradually over the next few weeks and is not available in Gov clouds.

Path Layer for the Azure Map visual

This month introduces a new feature to the Azure Map visual that takes geospatial analytics to the next level, namely the **Path Layer**. This provides users with the ability to visualise geographic connections between multiple points. Whether you're managing logistics, analysing

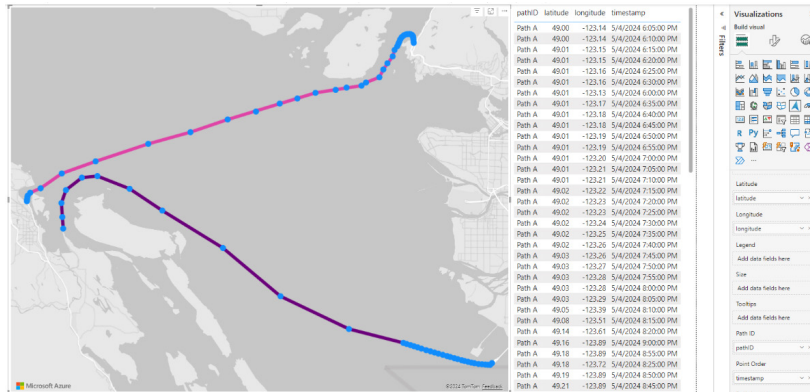
network traffic or tracking asset shipment across the globe, this feature allows you to view your connections between multiple geographic points in an intuitive and interactive way.

The Path Layer is useful for several key scenarios such as:

- **Network Analysis:** for industries like telecommunications, the Path Layer enables you to map intricate network connections. It helps identify inefficiencies, monitor data flow(s) and strengthen critical infrastructure
- **Flight Path Analysis:** airlines can leverage the Path Layer to visualise and analyse flight routes, improving air traffic management. It helps identify new route opportunities and enhances the overall passenger experience by optimising existing routes.

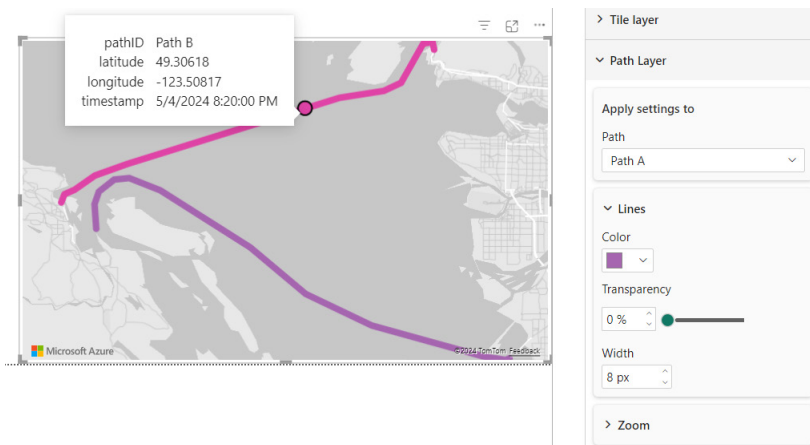
To get started, add the location for each point using either a geocoded location field, such as city names, or latitude and longitude. Then, differentiate between the paths by adding a field to the 'Path ID' field well and indicate the order of connection through the Point Order field well.

For example, you could create a map showing the path of two ships with the latitude and longitude of their positions for each point, a path field with a unique identifier for each ship, and a timestamp for each location to make sure the points are connected in the correct order.



You can also format the visual by controlling the colour, transparency and width of the lines, and even turning off the bubble markers for each point. If you turn off the bubble layer, you'll only see a bubble on

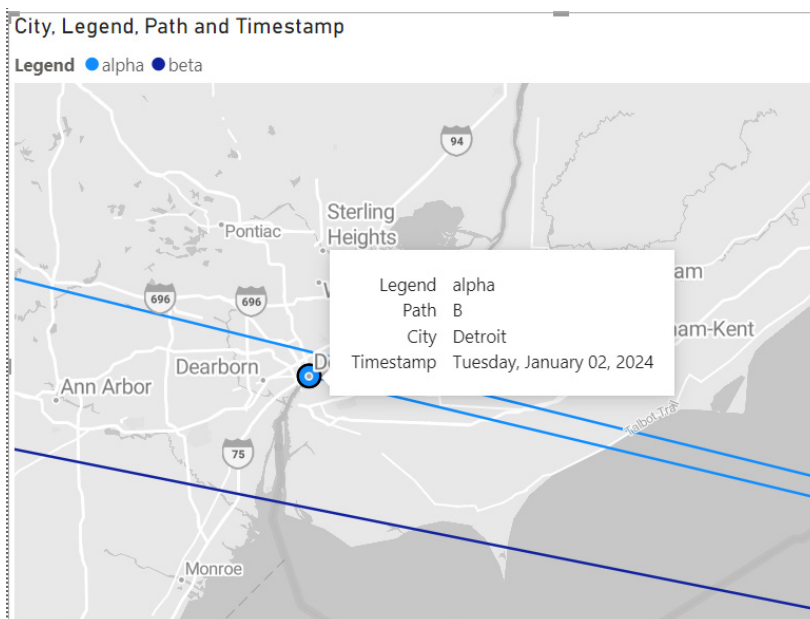
hover showing you the closest point to your pointer location. Paths are interactive as well, so you'll get ToolTips on hover and be able to cross-highlight other visuals by clicking on points of the lines.



There are a couple of interesting behaviours to be aware of with this new layer. First, when using a drill hierarchy with the path layer, the visual will automatically drill down to the lowest level and will not allow you to drill up, as points in the path would be aggregated at higher drill levels.

that location show up on top of each other. If you want to click on the bubbles underneath, just hover on the line associated with its path and it will float to the top and be selectable. Lastly, you can further break down the paths by adding a legend, which will create unique lines for each legend value of a given path ID.

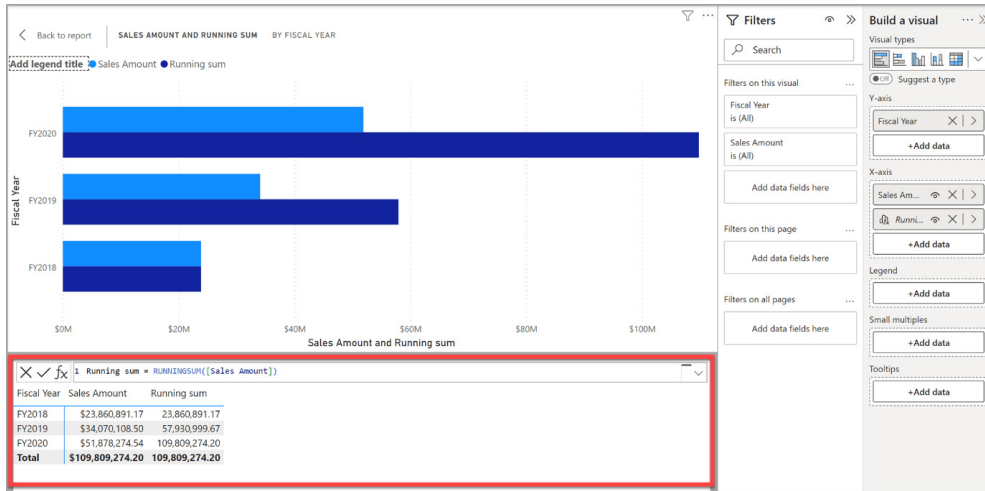
Next, if you have a location that's part of multiple paths, the bubbles for



Currently, the Path Layer operates mainly in conjunction with the bubble layer. Once you add a path to your map, you'll see that the filled, cluster bubbles, heat map and 3D column layers are all disabled. Additionally, while you can use the Path Layer in conjunction with reference layers, the reference layer will be static. It's currently unsupported to mix data bound reference layers with the Path Layer.

The Path Layer is still actively rolling out to all regions. Depending upon what region your tenant is in, you might not see the path layer in the Power BI Service immediately. Be sure to check the report after publishing, and if you don't see the layer, it should be accessible very shortly.

Visual calculations (Preview)



The work on visual calculations continues as usual and this month's updates present support for exporting. You can now export data from visuals that contain one or more visual calculations or hidden fields.

If you export data, hidden fields on a visual are not included in the export, unless you export the underlying data. The results of visual calculations are always included in the export, except when exporting underlying data, since visual calculations are not part of the underlying data as they only exist on the visual.

Small Multiples for the new card visual (Preview)

With this month's update, Microsoft has enhanced the Card visual with a new version that retains all familiar features and updates, whilst adding advanced functionality and an improved user experience with Small Multiples. This new feature is currently in Preview with the new Card visual, offering an excellent opportunity to experience the capabilities of the feature.

Small Multiples are a series of similar card tiles displayed together in a grid format, each representing a different category or dimension of

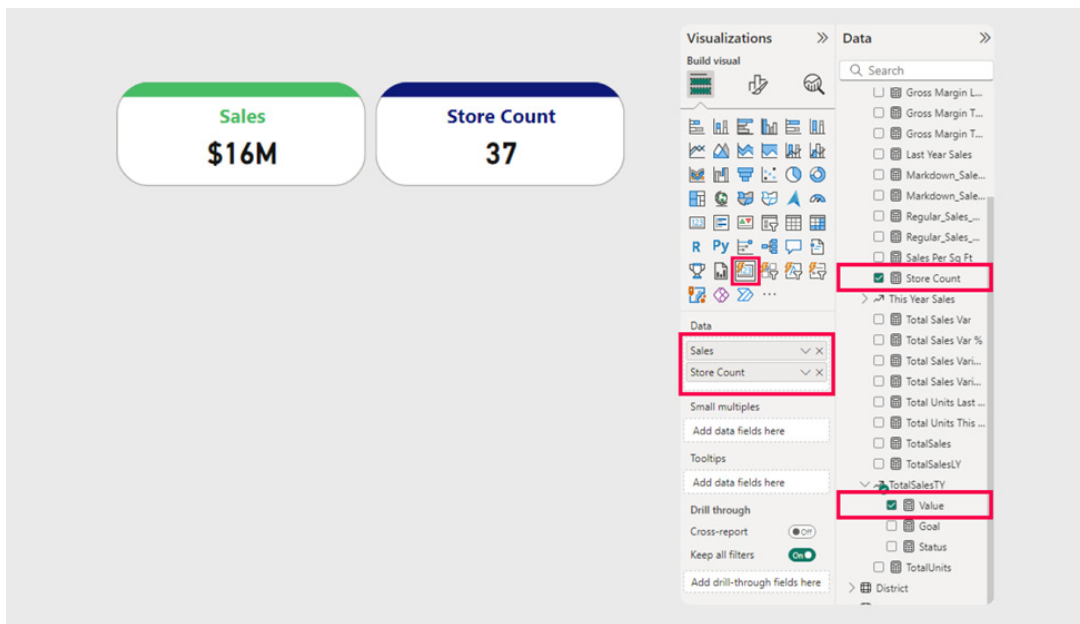
data, allowing for easy Objectives and Key Results (OKR) comparison and analysis across multiple fields.

This newly added feature enhances data organisation, visual clarity and performance, making it easier to analyse and present data effectively. To try it, navigate to **Options and settings -> Options -> Preview features -> New card visual**, and make sure it's enabled.

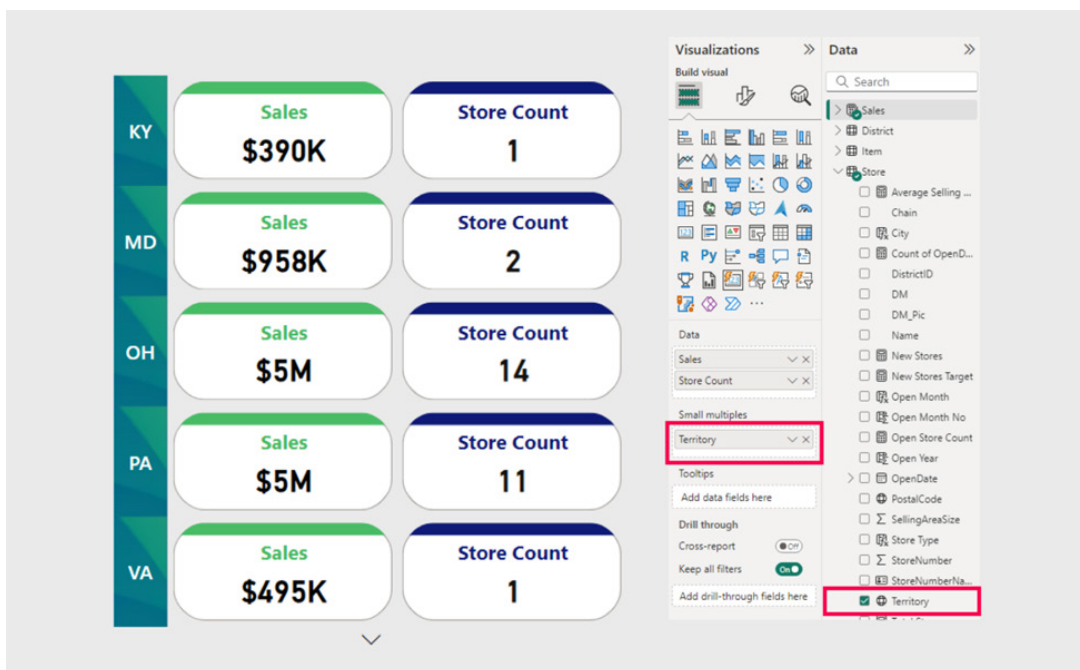
Another advantage of the new Small Multiples feature is the extensive customisation it offers. This includes:

- **Small Multiples layout:** choose from single column, single row or grid, and customise the number of small multiples, rows or columns displayed
- **advanced formatting options:** enhanced features such as font styles, colour-coding and conditional formatting
- **border and gridlines:** when enabled, individual controls for borders and gridlines permit the customisation of style, width, colour and transparency
- **overflow style:** options include continuous scroll or paginated, to smoothly navigate through multiple cards without overwhelming visual space
- **headers:** choose from horizontal or vertical orientation, top or left position, customisable alignment, font, colour, transparency, padding, plus background colour or image.

To create a card visual with Small Multiples, first select the 'Card (new)' icon from the visual gallery on the Build visual tab in the Visualizations pane, then select some data fields from the data model to add them to the data field well.



To categorise your cards using Small Multiples, choose a data field from the data model and add it to the Small Multiples data field well.



This new feature provides extensive customisation options, such as layout, advanced formatting options, conditional formatting, borders and gridlines, overflow style and customisable headers. Small Multiples for the Card visual in Power BI offer another great enhancement that significantly improves data organisation, visual clarity and performance.

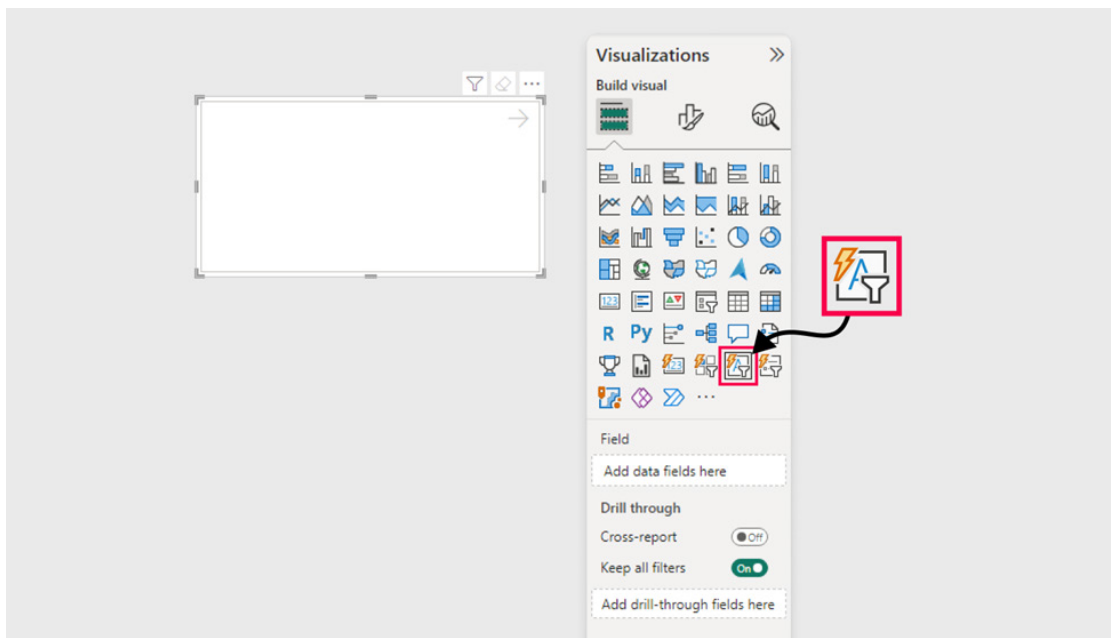
New visual: text slicer (Preview)

There is now a new text slicer, available in the core visuals gallery. You can enable the new text slicer by navigating to **Options and settings** -> **Options** -> **Preview features** -> **text slicer visual** to ensure its selected, and then restart Power BI.

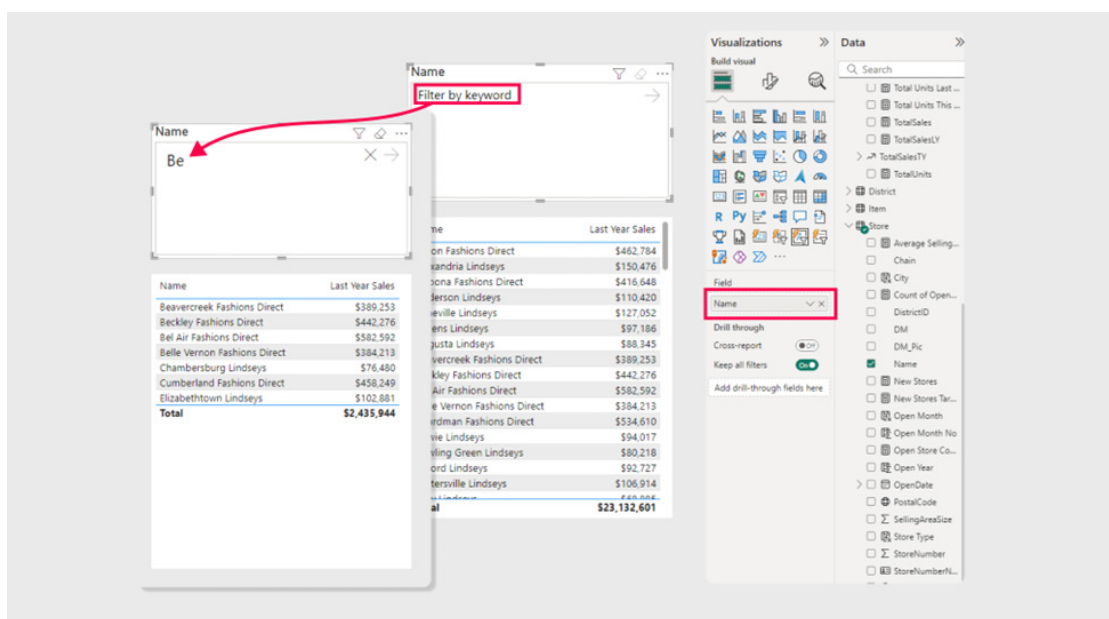
The text slicer works by allowing users to input specific text that acts as a filter, targeting a designated data field. By entering the desired text in the slicer's input box, the slicer effectively narrows down the dataset

to display only the relevant information that contains the entered text. This functionality is particularly useful for handling large datasets, where quick and precise filtering is essential for efficient data analysis and presentation.

To create a text slicer visual, select the text slicer icon from the visual gallery on the Build visual tab in the Visualizations pane. This adds a visual placeholder to the report canvas.



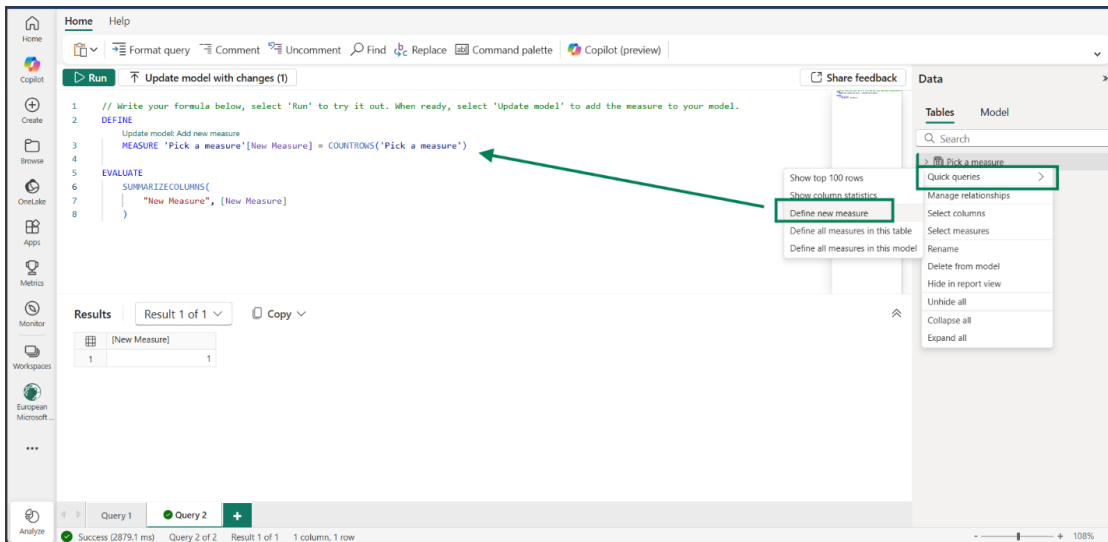
To filter a dataset, add a text field from the data model to the 'Field' well to establish the text slicer's functionality, allowing it to filter the dataset based on user input. Simply add text to the slicer's input box, select the apply icon, or press **ENTER** and the slicer immediately filters the dataset, displaying results on the visual.



As shown here, the new text slicer is a powerful and customisable filtering tool in Power BI.

Define new measures in DAX query view quick queries

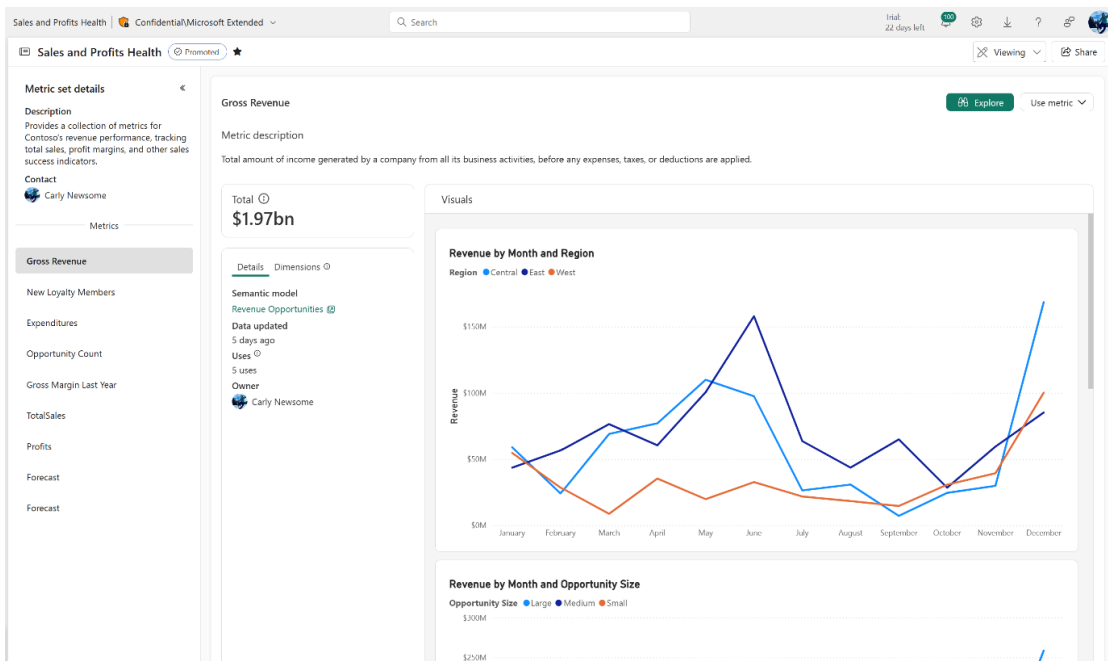
Creating measures in DAX query view has become a little simpler. The 'Quick queries' option available from the context menu of tables, columns or other items in the Data pane, now includes 'Define new measure', viz.



This will create a new query table with the syntax started for you to create a query-scoped measure **DAX** formula ready for you to add your own **DAX** formula and then run when ready.

Metric sets: a new era of metric management in Fabric (Preview)

The public Preview of metric sets is now officially available for both Power BI Service and Desktop. This is a transformative new feature designed to redefine how organisations manage and consume metrics.

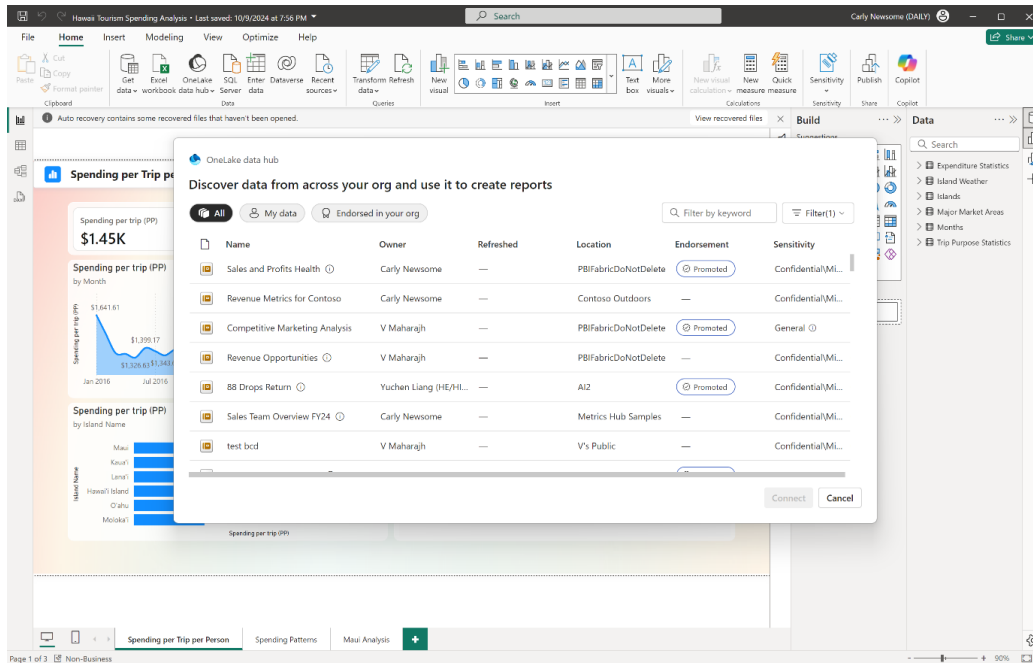


The Fabric Metric Layer's home base is the Metrics Hub in Power BI and brings powerful capabilities to streamline metric management, ensure consistency, and foster trust in data across the business.

Metric sets will be available for both consumers to browse and creators to use in reporting. A Service experience consisting of visualised metrics and data exploration will allow end users to answer their data questions. The Desktop experience will allow creators to connect to the most authoritative metrics to visualise in reports. Key features will 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 may 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. Metric sets may be promoted, endorsed and 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:** regarding metrics in Desktop, in the November release of Power BI Desktop, metric sets will be available to connect to and use in desktop reporting. You can access the metric you want to include in your model via OneLake datahub / data catalog and connect there. This ensures your reports use the most up to date and authoritative measures available.



Announcing performance improvements for models with calculation groups and format strings in Excel

Microsoft has announced “significant performance improvements” for **MDX** queries on models with Calculation Groups and Format Strings. The latest changes should greatly improve the performance and reliability of operations in ‘Analyze in Excel’ on models that include one or both of:

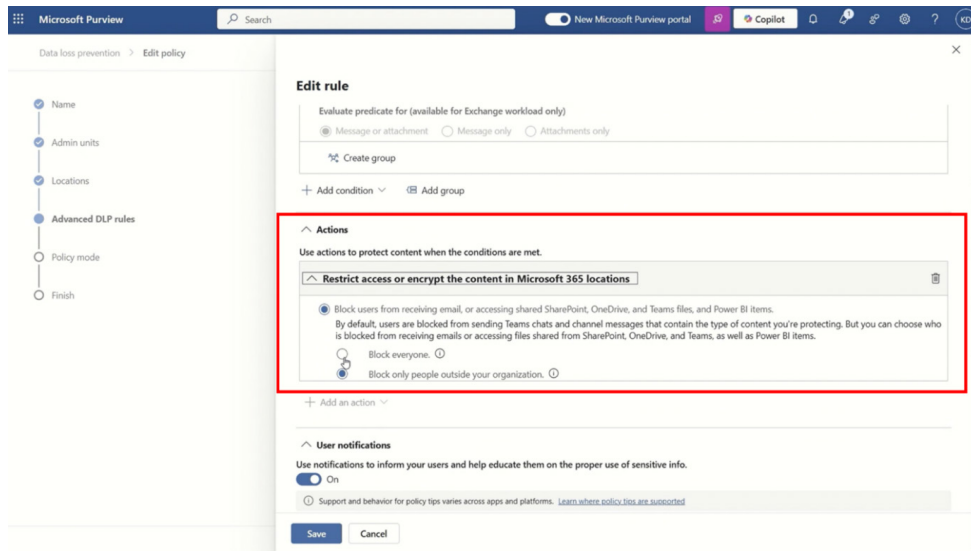
- Dynamic Format Strings for Measures
- Calculated Items with Format Strings.

This extends to other **MDX** scenarios as well, so all client applications that use **MDX** to query semantic models with the above will experience the same performance benefits.

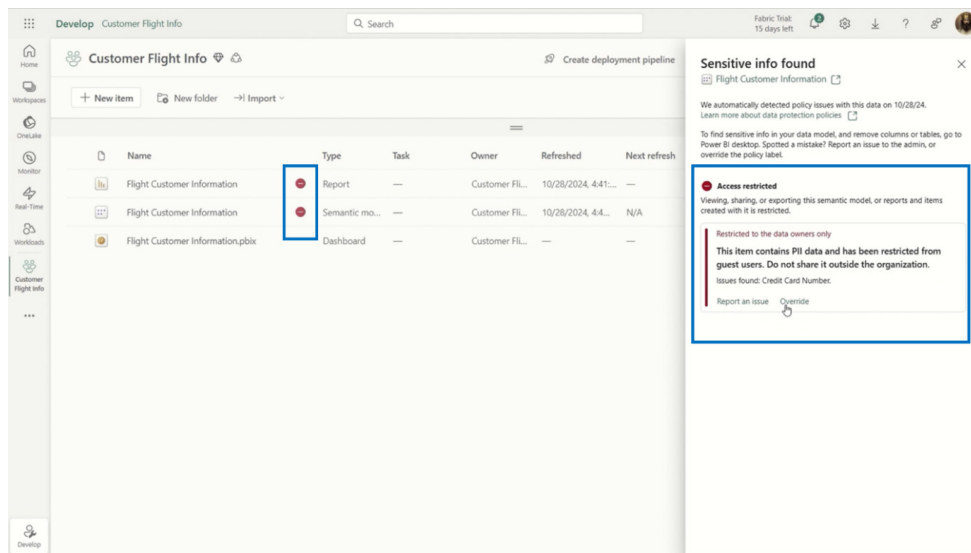
DLP policies restrict access action for semantic models (Preview)

Purview data loss prevention policies for Fabric now enable administrators to restrict access based upon the sensitive information detected within their semantic models’ data.

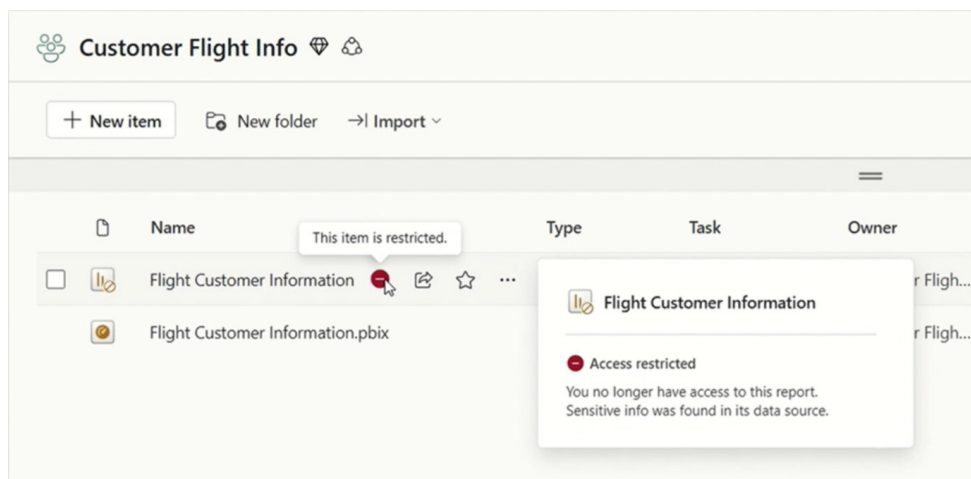
When Purview compliance administrators configure DLP policies for Fabric, they now have the option to decide if upon detecting sensitive information they would like to block access to the data. They have the option to prevent guest users from accessing the data or to restrict access for all users except the data administrator.



In Fabric, data administrators will see an indication that their data is restricted, and can act, such as reporting an issue to the compliance administrator or override the policy rule.



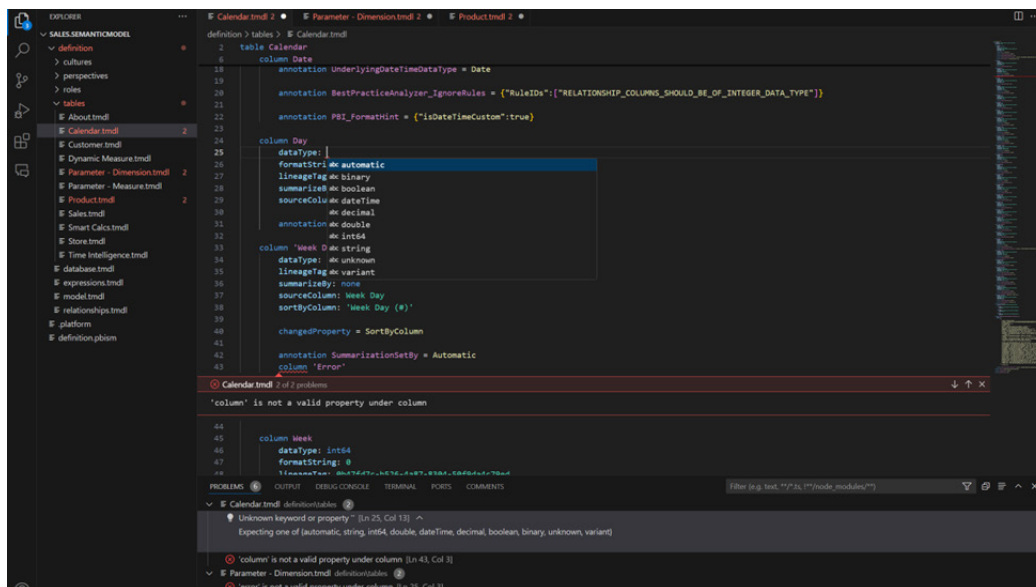
Consumers, such as guest users who have been now restricted from seeing this information, also see an indication letting them know that an organisational policy revoked their access, and if they attempt to see its content, they will not be able to do so.



With restrict access action, compliance administrators get further control and enforcement when uncovering sensitive data in their Fabric tenant.

Semantic modelling in Visual Studio Code with the new TMDL extension (Preview)

The new **TMDL** Extension for Visual Studio Code in public Preview enhances the Tabular Model Definition Language (**TMDL**) editing experience, boosting semantic model development for more advanced programmers.



TMDL is designed to make model representations readable, editable, collaborative and reusable. The **TMDL** Extension builds upon these strengths of **TMDL** with several key features that create a rich development experience, including:

- **semantic highlighting:** improves readability by applying different colours to parts of your code based upon meaning, making it easier to understand the structure and functionality of your **TMDL** briefly
- **error diagnostics:** helps you identify and fix issues in your code by clearly highlighting errors and providing you with detailed messages that guide you on how to resolve them
- **AutoComplete:** offers intelligent suggestions while you type to speed up your workflow, reduce the chance of errors and help you understand your code options.

By working in Visual Studio Code, you can also take advantage of other tools on the platform such as:

- **Source Control:** seamless integration with Git, allowing you to track changes, collaborate with team members and ensure version control of your semantic models
- **GitHub Copilot:** an AI coding assistant that will help you write code faster, generate TMDL from natural language and quickly apply advanced bulk edits to your models.

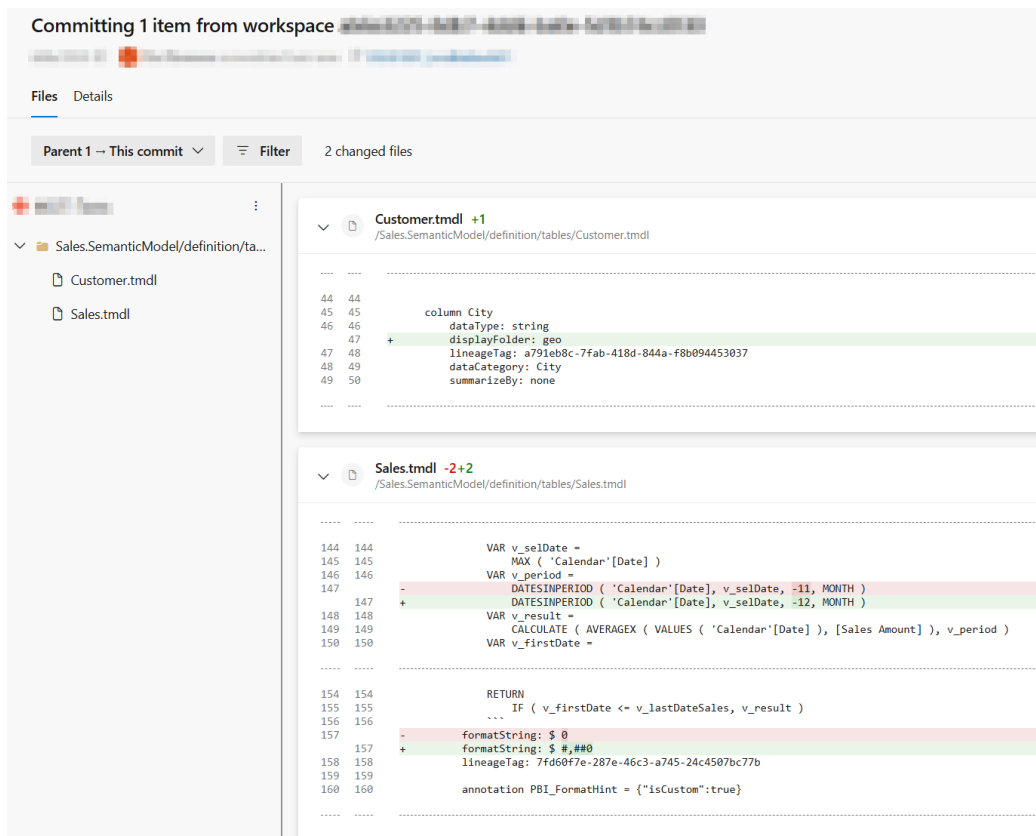
Snowflake connector updates

Queries sent to Snowflake by Power BI will now include a query tag with additional information about the context, such as whether or not it comes from Direct Query. This feature will be rolled out for the Power BI Service from November onwards, and will reach Power BI Desktop and the On-Premises Gateway early in 2025.

Fabric Git: TMDL format for semantic model export

As part of Microsoft's commitment to providing a developer-friendly experience that enhances team collaboration, Fabric Git integration will begin exporting semantic model definitions as Tabular Model Definition Language (**TMDL**) shortly. This change will replace the use of a single **JSON** file (**model.bim**) with Tabular Model Scripting Language (**TMSL**).

Due to its folder representation and readable format, **TMDL** offers a significantly improved source control experience. This enhancement facilitates tracking commit history and simplifies the resolution of merge conflicts, particularly when compared with **TMSL**.



Semantic model client library updates

Client applications, such as Excel or Power BI Desktop, connecting to Power BI semantic models now benefit from better performance due to an automatic conversion of legacy connection strings (e.g. pbiazure://*) to the XMLA endpoint. Requests are routed directly through the XMLA

endpoint, reducing intermediary steps, speeding up request processing, and decreasing the likelihood of errors. You may need to update your firewall rules.

KPI by Powerviz

KPI by Powerviz (Power-BI Certified) is a 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 the visual, with an option to create your own templates.

Design:

- 16 layers and 4vtions and colour styles
- rich customisation, formatting options and colour styles
- create KPI objects in layers, combining charts, metrics and icons.

Analytical:

- data visualisation types:
 - **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 values against targets
- formatting features: 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 schemes
- Small Multiples: support for all chart types, such as Fixed / Fluid with a change chart feature.

Other features include multi-categories comparison, highlight values, layer flexibility, and more.

Business use cases include sales performance, financial health and customer satisfaction analyses.

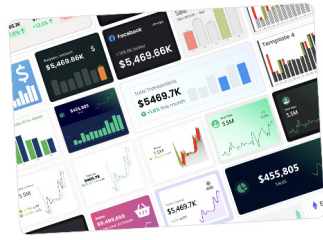
INTRODUCING KPI BY POWERVIZ

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

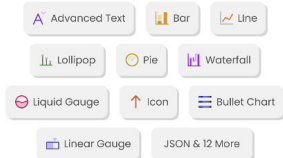
Don't settle for normal looking KPI Cards. Make them look stunning.

100+ Prebuilt Templates

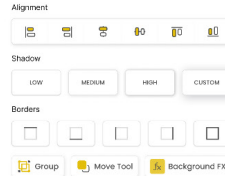
KPI offers 100+ pre-built template within the visual or create your own templates.



Build Any KPI You Can Imagine With KPI By Powerviz



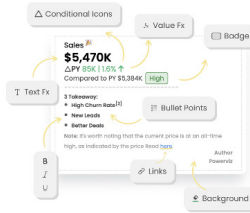
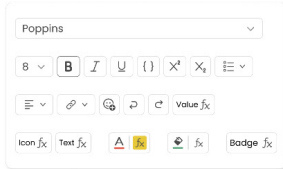
Upto 16 Layers And 40+ Chart Variations



Customization Options

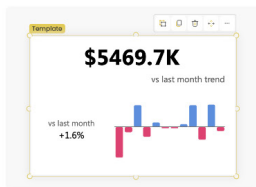
Advanced Text Editor

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

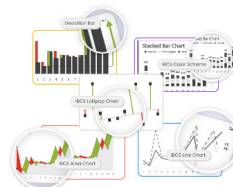


INTRODUCING KPI BY POWERVIZ

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



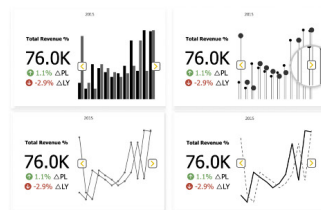
On-Object Interaction



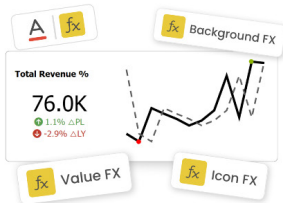
IBCS Support



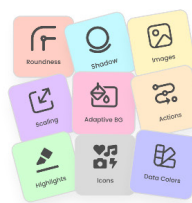
Small Multiples - Fixed And Fluid



Small Multiples - Change Chart Type



Conditional Formatting



Other Features

Unlock A World With Endless Possibilities With The KPI By Powerviz

Zebra BI Tables 7.3

With Zebra BI Tables 7.3, users may harness the power of a text editor to create and update visual comments with efficiency. This feature allows you to style and format your text, add bullet points and insert hyperlinks, making your report a one-stop shop for the entire team by just leaving links to reports and documentation different stakeholders might have an interest in. Well-structured comments can streamline communication within your reports, enabling readers to quickly grasp essential insights.

By emphasising what's important and explaining why it matters, you guide your audience towards critical information and promote clarity and understanding. This clarity is crucial in any business environment,

where time is often limited, and strategic decisions must be made swiftly. Effective comments reduce the time and effort required to generate actionable insights, which ultimately improves report quality and effectiveness.

Incorporating thoughtful commentary can transform a standard report into a powerful tool for decision-making. With Zebra BI Tables, enhancing your reports with meaningful comments makes it easier: all so you can communicate your message more effectively and engage with your audience better.



Waterfall PRO by ZoomCharts: interactive waterfall visual for financial data

Waterfall PRO by ZoomCharts is a user-friendly and insightful way to visualise financial data, combining incredible user experience with customisability and various features. It also seamlessly cross-filters data across multiple visuals, allowing you to create truly interactive Power BI reports.

Main features:

- **custom sequence:** have full control over the column order with the Sequence field
- **drill down:** use multiple categories to enable drill down directly on the Waterfall chart
- **automatic subtotal calculation:** display subtotals even if you don't have them in your data
- **rich customisation:** customise x and y axes, legends, ToolTip content and adjust the appearance settings for positive, negative and total columns separately
- **thresholds:** display up to four [4] constant or dynamic thresholds as lines or areas
- **cross-chart filtering:** dynamically filter data across multiple visuals.

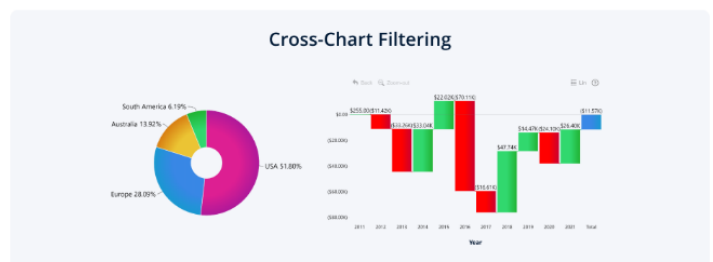
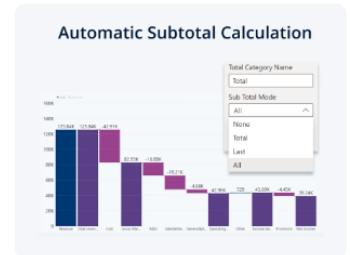
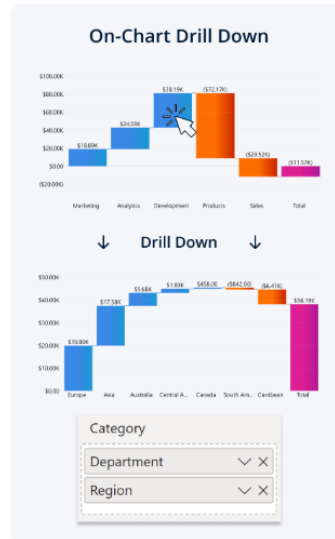


Drill Down Waterfall PRO



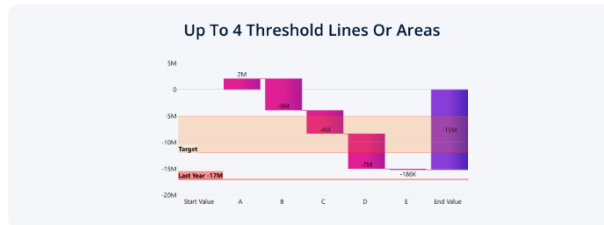
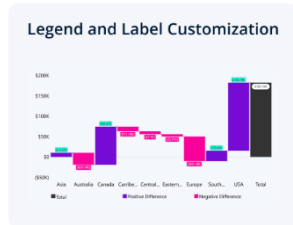
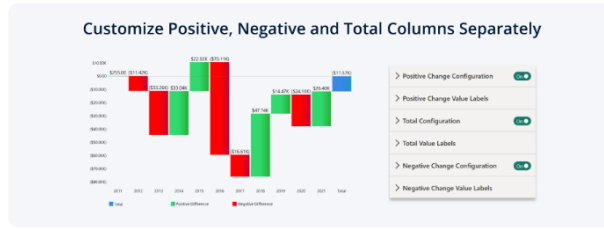
INTERACTIVE VISUAL FOR FINANCIAL DATA

Make your reports come alive with the most interactive and configurable waterfall chart visual for Power BI.



VISUALIZE DATA YOUR WAY

Explore more than 270 customization settings to create the perfect waterfall chart for your report.

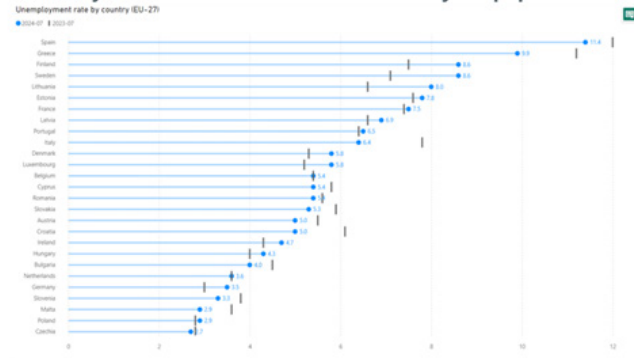


Lollipop Bar chart by Nova Silva

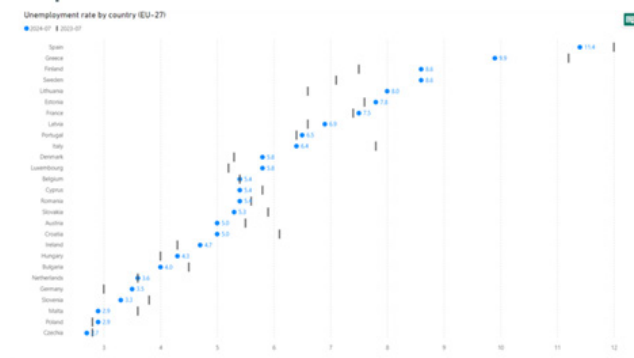
In this latest Lollipop Bar Chart release for Power BI, Microsoft added a much-requested feature: secondary markers. This allows you to display not only the primary value but also add context by including a secondary value marker.

Lollipop Bar Chart Nova Silva

Possibility to add an additional marker to every lollipop



Also possible to use this as a Dot Plot



This new feature integrates seamlessly with all other Lollipop Bar Chart functionalities, such as transforming the Lollipop Bar chart into a dot plot by removing the connecting bars, as shown in the second image (above). This also removes the requirement to start your numeric scale at zero [0], allowing you to have a closer look at the values and their differences.

While standard Bar charts are great for comparing a single measure across categories, they can become cluttered with larger datasets

(i.e. more than 10 categories). The coloured bars may fill too much of the chart space. To address this, the Lollipop Bar Chart offers a cleaner, more efficient alternative, minimising clutter without sacrificing clarity.

You can try the Lollipop Bar chart for free now on your own data by downloading it from the AppSource.

Sales Velocity chart

The Sales Velocity chart is a tool for analysing product sales and profitability in specific countries. It uses a combination of pie charts, needles and colour coding to visually represent key metrics.

Key features include:

- **needles:** length indicates sales percentage; width reflects profit margin
- **Pie and Circle size:** reflects overall current sales in a country
- **colour coding:** green (high profit), yellow (moderate), red (low profit)
- **sales trend dot:** grey (no data), red (decreasing sales), green (increasing sales).

Benefits include:

- **visual clarity:** easy to understand data representation
- **dynamic and scalable:** handles large datasets and adapts to screen size
- **interactive features:** ToolTip displays details, premium options offer filtering and logo removal.

Use cases include:

- businesses may identify top sales regions and areas needing improvement
- financial analysts can pinpoint high and low profit contributors.

Salient Features

- Visual Clarity: Offers a clear and organized graphical representation of sales and profit data.
- Resizable & Dynamic: Chart adjusts dynamically based on device size and user preferences.
- Interactive Elements: Tooltips, dropdowns, and legends provide a user-friendly interface for deeper insights.
- Color-Coded Representations: Uses colors (Green, Yellow, Red) to indicate profit levels, making comparisons easier.
- Animation Support: Includes smooth animations for a visually appealing presentation of data.

Circle Size Reflects Sales 1

The radius and size of the circle represent the current sales in a region, with larger circles indicating higher sales figures.

Region Boundary 2

The boundary of each region highlights the number of products sold in that specific area, visually distinguishing sales distribution across regions.

Pre-Sales Data Indicator 3

The center circle turns green to highlight strong pre-sales performance, while a green middle dot shows that current sales exceed previous sales, signaling positive growth.

Sales Decline Indicator 4

The center circle turns red to show low total sales or pre-sales data, and a red middle dot appears if previous sales surpass current sales, highlighting a drop in performance.

Needle Width 5

The width of the needle in the chart reflects the profit margin, with wider needles indicating higher profits for each product or region.

Needle Length 6

The length of the needle represents the sales percentage of a product in a specific region, with longer needles indicating higher sales contributions.

Why the Sales Velocity Chart is Used?

- Sales & Profit Analysis: Helps identify regions with the highest and lowest sales and profit.
- Trend Comparison: Tracks and compares current and previous year sales performances.
- Efficient Data Handling: Manages large datasets effectively for quick insights.
- Product Performance Forecasting: Analyzes pre-sales data to predict future sales trends.
- Customizable Views: Allows users to filter and view data for top products or regions using dropdown settings.

New book: Data Visualization with Microsoft Power BI

Microsoft has suggested the new book *Data Visualization with Microsoft Power BI* by Alex Kolokolov and Maxim Zelensky, which suggests DataViz best practices for Power BI. We haven't read it ourselves, but apparently, there are 25 chapters about different chart types, 40 visuals and the book has 400 colour pages too.



The book is separated into three [3] parts:

- **classic visuals:** authors explain how to choose charts for basic types of analysis and avoid common mistakes. It also discusses how to set up interactions and put visuals together on a dashboard
- **trusted advanced visuals:** different options and data requirements for Waterfall and Bullet charts, Gantt, Tornado, Funnel, Sankey, etc.
- **risky advanced visuals:** charts that may confuse the average user. Simpler alternatives are suggested.

Support for Power BI language settings when a paginated report is viewed on the Power BI Service

When a localised paginated report is published to the Power BI Service, the viewer of the report will now see the report in the preferred language that they have selected in the Power BI / Fabric Settings page. Previously, the rendering of the report was determined by the server settings.

More next month.

New Features for Excel

This month's updates see the announcement of Clean Data with Copilot in Excel now available in Excel for the web. Also, **GROUPBY** and **PIVOTBY** functions are now Generally Available in Excel for Windows, Mac and the web. Furthermore, 'Focus Cell' is now available to Insider users on Excel for Windows.

The full list is as follows:

Excel for the web

- 'Clean Data' with Copilot in Excel

Excel for Windows, Mac and web

- **GROUPBY** and **PIVOTBY** functions

Excel for Windows

- Focus Cell (Insiders)
- 'Color Picker'
- New templates in Excel
- 'Draw and Hold' gesture (Insiders)

Excel for Mac and iOS

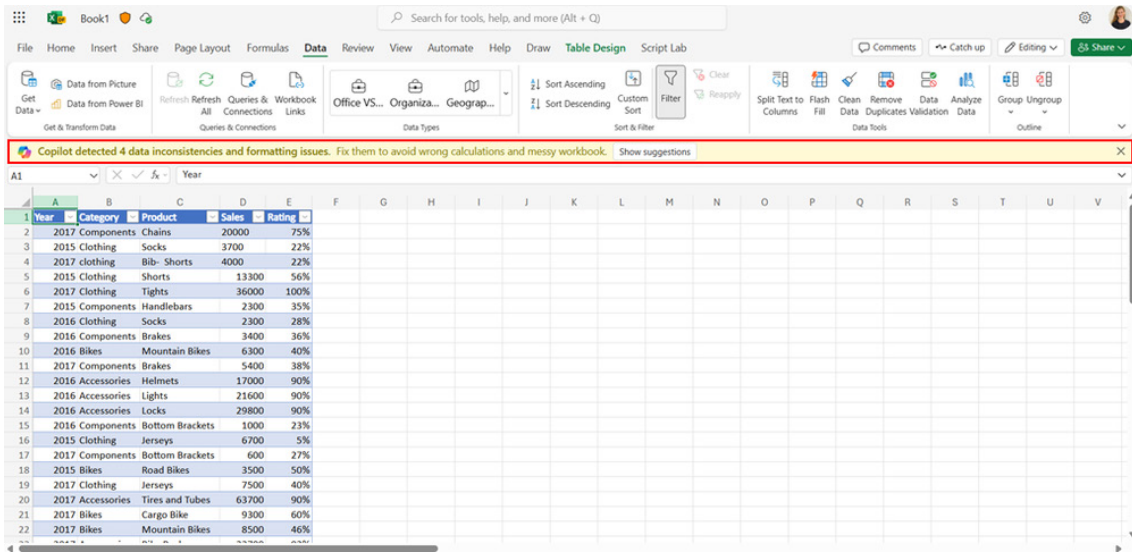
- Switch between Apple devices with Handoff support (Insiders).

Let's get started.

'Clean Data' with Copilot in Excel

There is a new AI-powered feature in Excel for the web: 'Clean Data'. With Copilot in Excel, you can clean data with just one click. 'Clean Data' detects and offers solutions for text inconsistencies, number format issues and superfluous spaces. As mentioned above, this feature is now available to Excel for the web users and will be rolling out to Windows desktop users shortly.

It's easy to use. Simply select the 'Clean Data' option in the Data tab of the Ribbon or interact with the pop-up banner to get started, viz.

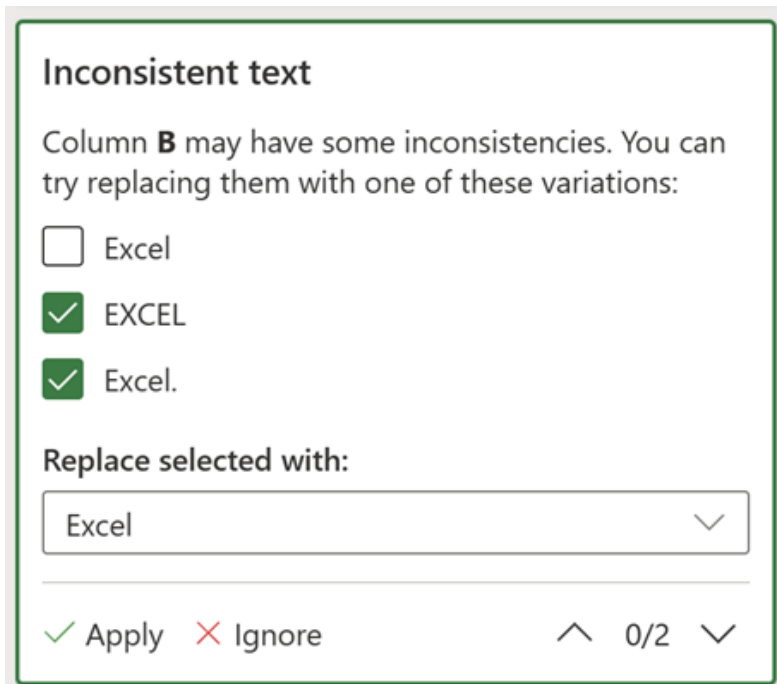


Here, you can see the Excel window showing what is known as the Business bar with the text: "Copilot detected 4 data inconsistencies and formatting issues. Fix them to avoid wrong calculations and messy workbook".

There are various issues this feature may detect.

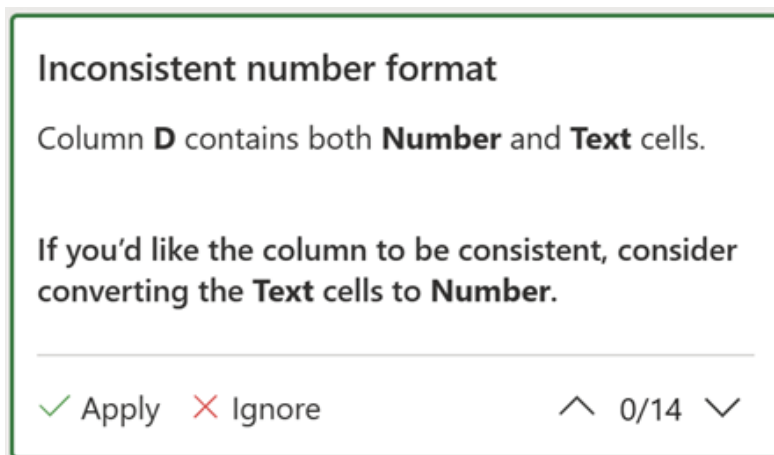
TEXT INCONSISTENCIES IN CELLS OF THE SAME VALUE

This feature seeks to identify unnoticed variations across similar values that could cause subsequent problems. For example, a column with a mix of uppercase and lowercase text in the cells, or cells in a column containing differences in punctuation or diacritics. 'Clean Data' will locate these inconsistencies and will help you to easily correct them.



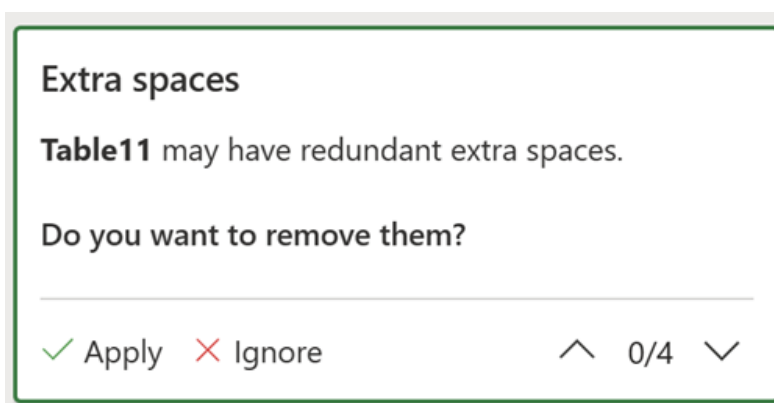
NUMBER FORMAT INCONSISTENCY IN A COLUMN

This feature seeks to identify columns with a mix of both text and number formatted cells that could result in incorrect calculations. 'Clean Data' will locate these inconsistencies and will help you convert the text formatted cells to a number format.



EXTRA SPACE REMOVAL SUGGESTIONS

'Clean Data' will also detect and suggest the removal of unnecessary spaces in the highlighted table, such as extra leading spaces, trailing spaces and between-value spaces.



Microsoft offers the following helpful pointers for getting the most out of 'Clean Data':

- 'Clean Data' works best with tabular data of up to 100 columns and 50,000 rows
- 'Clean Data' works best with data that is formatted as an Excel table. To create an Excel table, click anywhere in your data and then select **Home- > Tables -> Format as Table**
- While not limited to English, 'Clean Data' will initially perform best in English.

GROUPBY and PIVOTBY functions

GROUPBY and PIVOTBY – and their associated functions and features – are finally in Excel for Windows, Mac and the web. 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 LAMBDA

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												
11												
12												
13												
14												
15												
16												
17												
18												

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

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

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

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

	E	F	G	H	I	J	K	L	M
19									
20									
21									
22									
23									
24									
25									

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

=BYCOL(G21:J24,SUM)

=BYCOL(G21:J24,SUM)

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

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
Total	1,984,200

=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
Total		1,984,200

=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
Total		1,984,200

=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
Total		1,984,200

=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
Total		1,984,200

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
Total		1,984,200

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
Total		1,984,200

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%
Total		1,990	51%

`=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
Total	1,984,200

`=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

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

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%
Total	100.00%

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

Alternatively, it may be used on its own:

	E	F	G	H	I	J	K	L
11								
12			Category	Sales				
13			Accessories	24.47%				
14			Bikes	24.99%				
15			Clothing	25.69%				
16			Components	24.86%				
17			Total	100.00%				
18								
19			Accessories and Bikes	49.46%				
20								

=PERCENTOF(G13:G14,G13:G16)

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*).

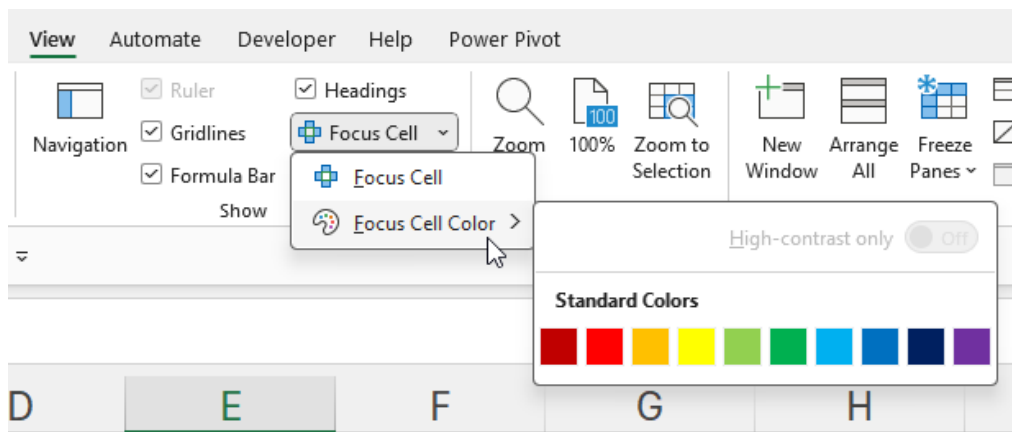
Let's look at **PIVOTBY** using **PERCENTOF**. Consider the following Table (**CTRL + T**) called **Data** (*truncated*):

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%

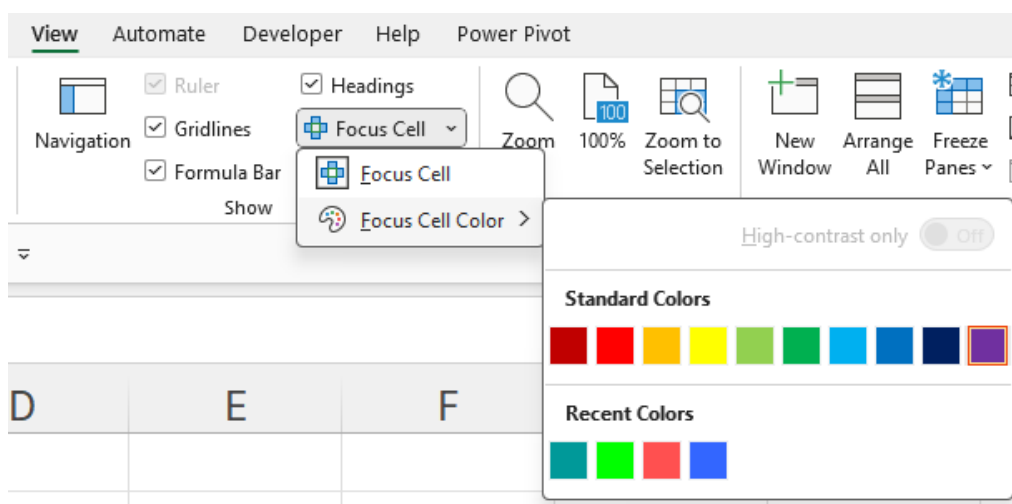
In this final illustration, the Total row is 100% throughout. This looks similar to the default Scenario 0: Column Totals. This is because the row here refers to the headings in each row, *i.e.* **Category** and **Item**. You can see that for any row the sum of any category for any given **Quarter**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

Focus Cell (Insiders)



Found on the View tab in the Ribbon, 'Focus Cell' is new to Excel for Windows and has now been "officially announced" (even though we detailed it two months ago!). It 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.*

	A	B	C	D
1				
2				
3				
4				
5				

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.

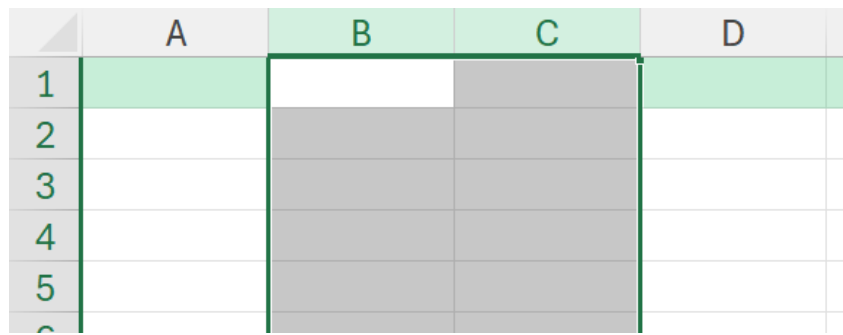
If multiple cells are selected, the active cell remains highlighted:

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

	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 column are chosen:

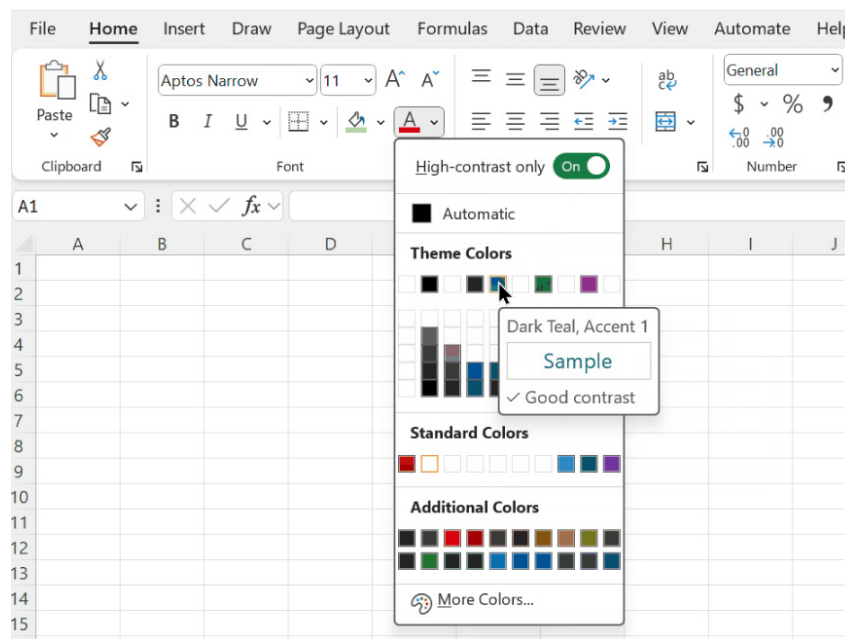
	A	B	C	D
1				
2				
3				
4				
5				



Presently, this feature does not appear to work in some pre-existing files or those with 'Freeze Pane' or 'Split' employed.

'Color Picker'

Also in Excel for Windows, Excel's 'Color picker' (*sic*) includes a feature that shows you if the selected colour contrasts adequately with the background. By hovering over a colour, you will see a ToolTip indicating whether the colour has "Low Contrast" or "Good Contrast" against the backdrop. Additionally, you can turn on the "High-contrast only" toggle to filter and display only colours with sufficient contrast.



New templates in Excel

There are some new templates in Excel for Windows this month. To access them, go to **File -> New**. Then type in "Python" in the Template search bar. These Python in Excel templates are currently only available in the English language and are detailed below.

1. **Python in Excel tutorial:** here, you may learn about Python in Excel and how you can use it to analyse data directly within your Excel spreadsheets

Get started with Python in Excel

Easily combine Python and Excel analytics in one workbook with Python in Excel, no setup needed. Just type Python in a cell, and calculations run securely in the Microsoft Cloud. Results, including plots and visualizations, appear in your worksheet.

Available now on Excel for Windows. Stay informed by joining the waitlist at aka.ms/expywaitlist.



Let's go >

Become familiar with Python (=PY)

1. To write or edit Python code, you need to make sure you're using a Python formula. To do so, go to **Formulas > Python** and select **Insert Python**, or enter **=PY** into a cell and select **PY** from the formula AutoComplete menu. You can also enter **=PY** in a cell to automatically select the PY function.



CHECK THIS OUT

Select the **Insert Python** button in the Formulas tab to view tips and samples to help you get started with Python in Excel.

2. You don't need to install Python on your computer to use this feature. All Python runs securely in the Microsoft Cloud.

3. You can see a change in the cell and formula bar when Python is enabled. This helps distinguish when you're writing or editing a Python formula. To commit Python code, you must first always enable the Python formula.



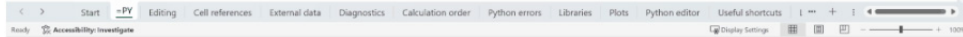
Python in Excel cell badge



Formula bar Python in Excel badge

CHECK THIS OUT

To learn more about =PY, visit [PY function - Microsoft Support](#).



Reference cells to bring data into Python

To protect your security, common functions used to import data in Python (like `pandas.read_csv` and `pandas.read_excel`) aren't compatible with Python in Excel. To import data, use an Excel worksheet or a Power Query data source.

To import data from an Excel worksheet, use the `xlf()` function for Python in Excel.

This function takes two parameters: a reference string pointing to data in Excel, and an optional parameter to adjust how headers are treated. Examples: `xlf("A1")`, `xlf("A1", headers=True)`, and `xlf("SAS1:SDS10", headers=False)`.

1. To start, take a look at DataFrames created from the example data in cells **D4** to **F10**. Cell **H5** is a DataFrame as a Python object and **L5** is a DataFrame as Excel values. Inspect or play around with the reference and headers parameters. Note: Multi-cell references (or ranges) like **A1:C5** are treated by default as pandas DataFrame objects.

2. To create a new Excel reference while in a Python cell, select the data on the grid that you'd like to use. If you prefer to use the keyboard, use **F2** to toggle between Enter mode and Edit mode in Python cells, and then use your arrow keys to select cells or ranges. Don't forget to type **=PY** first!

3. Give it a try! In cells **H15** and **L15**, create new DataFrames using the Income and Spend data. Set **H15** to be a Python object and set **L15** to Excel values. Note: If you reference an Excel table, you need to include the full structured table reference: `=PY(xlf("Table1[#All]"))`.

Example data

Category	Product	Sales
Accessories	Helmets	\$8,300
Accessories	Tires and Tubes	\$8,700
Bikes	Road Bikes	\$3,500
Bikes	Mountain Bikes	\$3,100
Components	Pedals	\$800
Components	Wheels	\$10,000

Example: as Python object

`[x]` DataFrame

Try it out! Range

User ID	Income	Spend
1	233	150
2	250	187
3	204	172
4	236	178
5	354	163
6	192	148
7	294	153
8	263	173
9	199	162
10	168	174

Set to Python object

Try here!

Try it out! Excel table

Set to Python object

Try here!



Understand a few errors

1. **#BUSY!** This displays when a Python formula is calculating.

Formula

Try here!

Expected result

#PYTHON!

2. **#PYTHON!** This error displays when there's an issue with the Python code. By default, the **Diagnostics** pane automatically opens with more details on the error. To see how this works, select cell **D6**, enable Python in the cell, enter `"1/0"`, and commit the formula. You should see a **#PYTHON!** error alongside the diagnostics pane with more information. You can also select the error icon next to the cell and then select **Show Error Message** from the menu.

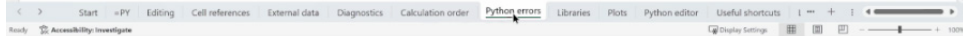
GOOD TO KNOW

You can change your calculation mode through **Formulas > Calculation > Calculation Options**.

Automatic automatically recalculates all dependent formulas every time you make a change in Excel. This is the default setting.

Manual recalculates formulas when you manually trigger the process to do so.

Partial is a hybrid between manual and automatic. This mode recalculates all formulas automatically except those that are typically time consuming, like Python and Data Tables.



Create Python plots

1. You can leverage powerful Python libraries to create data visualizations and plots. Give it a try! Modify the type of plot created in cell **H8** by choosing from the dropdown options in cell **H7**. You'll notice that as you switch the plot type, the code recalculates to create the appropriate plot and the floating image of the plot also updates automatically.

2. By default, you see a Python object returned to the cell with an Image data type. To view the plot, create a linked reference of the image that can be resized. In the **Insert Data** or right-click menu, select **Display Plot over Cells**. Alternatively, use the keyboard shortcut **Ctrl+Alt+Shift+C** when selecting a cell with the Image data type. This has been done already for Plot 1 in cell **H6**, but give it a try when you create the multiplot and histogram in the next step.



Insert Data menu



Right-click menu

3. Try it out! Use the Python code below to experiment with plots. To view the plots, don't forget to display the plot over cells as explained in step 2.

Try a **Multiplot** in cell **H24**.

```
decompose = seasonal_decompose(dflots["Passengers"], model='additive', period=36)
```

Example data

Dates	Passengers
1/1/1949	112
2/1/1949	118
3/1/1949	132
4/1/1949	129
5/1/1949	121
6/1/1949	135
7/1/1949	148
8/1/1949	148
9/1/1949	136
10/1/1949	119
11/1/1949	104
12/1/1949	118
1/1/1950	115
2/1/1950	126
3/1/1950	141
4/1/1950	135
5/1/1950	125
6/1/1950	149
7/1/1950	170
8/1/1950	170

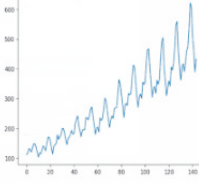
Libraries

Libraries Loaded

DataFrame [x] DataFrame

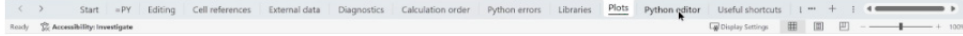
Type of Plot

Plot 1 [x] Image



Multiplot

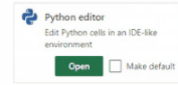
Try here!



Use the Excel Labs Python editor to write or view code

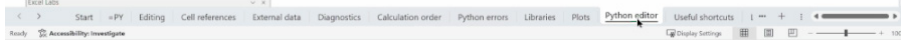
The Python editor is a new addition to Excel Labs that allows you to write and edit Python formulas in Excel, using a dedicated code editor with similar capabilities as those available in Python notebook environments. This is an experimental release to get early feedback, so we can develop Excel to be the most useful for you. The Python editor is designed to complement the Python in Excel experience with an editing surface optimized for more complex and longer Python code snippets.

1. Install **Excel Labs**: Go to **Home > Add-ins** in the ribbon, search for Excel Labs and select **install**. Once installed, the Excel Labs add-in displays in **Home**.
2. Launch Excel Labs and load the **Python editor** experiment. Select **Open** to load the editor experience. You can choose to also make it open by default.



Select Python editor in Excel Labs

3. Give it a try! You can edit any existing formulas or create new Python formulas in the editor.



Use Python libraries

1. Python in Excel uses a secure, open-source library distribution from Anaconda. The following core set of libraries are recommended: **pandas, NumPy, scipy, scikit learn, statsmodels, matplotlib, seaborn, json, and regex**. Note: NumPy and pandas are pre-initialized by default using **import numpy as np** and **import pandas as pd**.

Formula
 Try here!
2. Python in Excel settings and configuration details, like importing open-source Python are done through the **Initialization** pane. View the default configuration by selecting **Formulas > Python (Preview) > Initialization**.

GOOD TO KNOW
 By default, single cell references are converted to their scalar value while multi-cell references are converted to DataFrames.
3. The initialization pane is read-only, so you must import new libraries in a cell. To try it out, enter **=PY** in cell **D4** and add an import statement (such as **import seaborn as sns**). Return a string in your code like **"Python libraries loaded"**. If you don't return anything, the formula returns a Python object of NoneType.

CHECK THIS OUT
 To learn more about open-source libraries to use with Python in Excel, visit [Open-source libraries and Python in Excel - Microsoft Support](#).

```

py import seaborn as sns
import numpy as np
"Python libraries loaded"
    
```

Library import statements in the formula bar.

Next, learn about plots.

2. **QR code generator:** you can now easily generate customised QR codes in Excel to share with others for quick access to links

QR Code Generator

Note: Use of this template requires an active Microsoft 365 subscription.

URL:

Style:

Color Gradient:

Background Color:

Fill Color 1:

Fill Color 2:

3. **Dummy data generator:** you may now generate realistic dummy data for testing or demos.

Random data generator

Please note that using this template requires an active Microsoft 365 subscription.

1. Select column items: name, date of birth, city, country, company, job, phone number, email, address, text

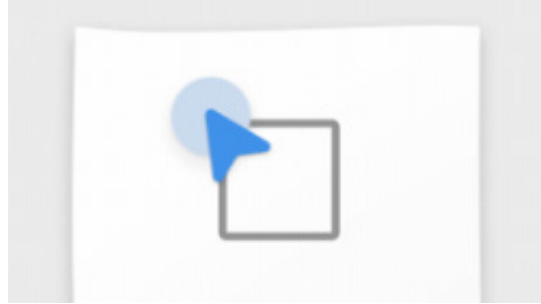
2. Number of rows:

3. Your random data generated as a DataFrame:

name	city	company
@ Kyle Fisher Jr.	@ Williamland	@ Fox-Miller
@ Darlene Thomas	@ New Bruce	@ May-Perry
@ Edgar Boyd	@ Scottfort	@ Garcia PLC
@ Frances Page	@ Combshaven	@ Morales-Davis
@ Trevor Ferguson	@ South James	@ Dodson-Brown
@ Rebecca Ford	@ Lake Matthewberg	@ Burton-Moore
@ Kristin Steele	@ New Austin	@ Lewis, Smith and Owens
@ Charles Baker	@ Amberbury	@ Brewer, Andrews and Velazquez
@ Adam Morrow	@ Clarkport	@ Pierce and Sons
@ Dean Caldwell	@ West Richard	@ Serrano, Cantu and Martinez

'Draw and Hold' gesture (Insiders)

In early 2023, Microsoft introduced the 'Draw and Hold' gesture in OneNote for Windows. Next, it was brought into Word for Windows. Now, Microsoft has extended the availability of this handy tool and enable you to use it in your worksheets and presentations in Excel for Windows (and PowerPoint).



To use it:

- open a new or existing Excel worksheet (or PowerPoint presentation)
- using a single stroke, sketch a line, arrow or shape on the canvas, you should then hold your pen in place for a few milliseconds
- watch as the shape is converted to a neater version of the stroke you drew.

This feature supports shapes such as circle, ellipse, triangle, square, rectangle, rhombus, trapezoid, pentagon and hexagon. If you're satisfied with its appearance and location, you can leave the shape as it is. Otherwise, you may simply drag the object to resize or rotate it.

This feature is available to Beta Channel users running Version 2410 (Build 18108.20000) or later.

Switch between Apple devices with Handoff support (Insiders)

Handoff is an Apple feature that allows you to continue working on content while switching from one Apple device to another. It is now available on Excel for Mac and iOS (Insiders versions).

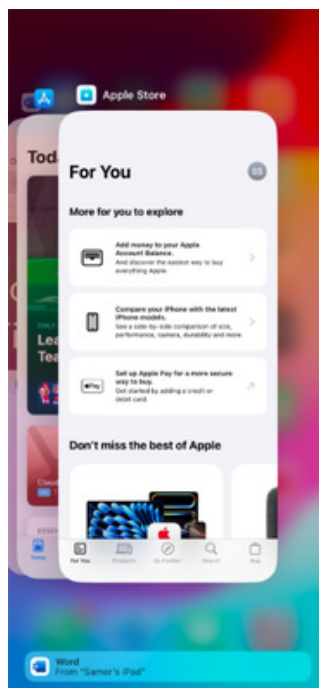
For example, if you're working on a PowerPoint presentation on your iPhone and want to switch to your Mac, Handoff makes it possible with just a couple of clicks.

It should be noted that:

- **you must use supported devices:** Handoff for Word, Excel and PowerPoint are available on iPad, iPhone and Mac. It is not supported on Apple Watch
- **you must use supported apps:** install Word, Excel or PowerPoint from the App Store, sign in with the same Apple ID across your devices and ensure your apps are up to date
- **you must use cloud files:** the document you want to Handoff should be stored in the cloud on OneDrive or SharePoint.

To activate the feature, open an existing document, worksheet or presentation on one of your Apple devices (Mac, iPhone or iPad) and bring it near another Apple device where you want to continue working.

After a few seconds, you will see in the Dock on your Mac or iPad, or the switcher on your iPhone indicating that the file is waiting for you to hand it off.



Tap on that application to open the file on the second device.

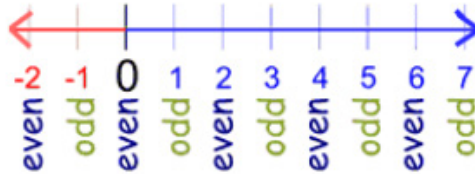
This capability is available to users running Word, Excel and PowerPoint for:

- **iOS:** Version 2.89 (Build 24090210) or later
- **Mac:** Version 16.89 (Build 24091630) or later .

Until next month.

The A to Z of Excel Functions: ODD

This function converts an octal number (base eight) to a binary number (base two).



It might seem odd, but this function returns a **number** rounded up to the nearest ODD integer. You can use this function for ensuring a binary number always ends in one [1].

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

ODD(number)

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

- **number:** this is required and represents the value to round.

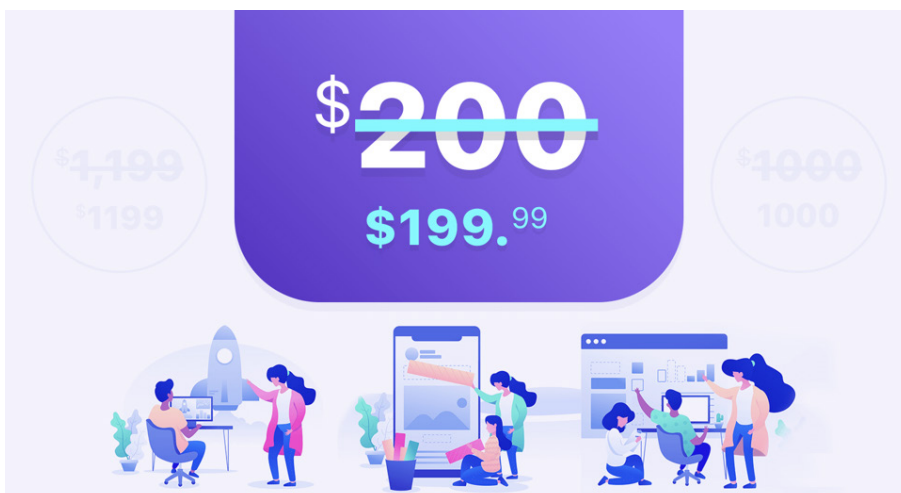
It should be further noted that:

- if **number** is nonnumeric, **ODD** returns the **#VALUE!** error value
- regardless of the sign of **number** (*i.e.* positive, zero or negative), a value is rounded up when adjusted away from zero. If **number** is an ODD integer, no rounding occurs.

Please see our example below:

	A	B	C
1	Formula	Description	Result
2	=ODD(0.4)	Rounds 0.4 (away from zero) to the next odd integer	1
3	=ODD(3)	Rounds 3 (away from zero) to the next odd integer	3
4	=ODD(-4)	Rounds -4 (away from zero) to the next odd integer	-5
5	=ODD(-5)	Rounds -5 (away from zero) to the next odd integer	-5

The A to Z of Excel Functions: ODDFPRICE



A loaf of bread for \$5,000? Yes, inflation is on the up, but that is an **ODDFPRICE!!** Hmm, maybe not – and whilst we’re at it, don’t bother looking for an **EVENFPRICE** either...

The **ODDFPRICE** function is an Excel financial function that calculates the price per \$100 face value of a security or bond with an irregular first period. That’s what is meant by “odd”! The **F** isn’t an expression for the horrific formula upon which it is based, but rather denoting this calculates for the First period.

In financial analysis, some bonds come with irregular first or last periods. Due to the irregular first or last period, the payment doesn’t fit in any of the usual or standard patterns. If you wish to calculate the price of a bond with an “odd” first period (either short or long), we can use the **ODDFPRICE** function.

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

ODDFPRICE(settlement, maturity, issue, first_coupon, rate, yield, redemption, frequency, [basis])

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

- **settlement:** this is required. This is the security's settlement date. The security settlement date is the date after the issue date when the security is traded to the buyer
- **maturity:** this is also required. This represents the security's maturity date. The maturity date is the date when the security expires
- **issue:** again, required. This is the security's issue date
- **first_coupon:** required. This denotes the security's first coupon date
- **rate:** yet again required. This is the security's interest rate
- **yield:** this is also required. This is the security's annual yield
- **redemption:** required. The security's redemption value per \$100 face value
- **frequency:** also required. This considers the number of coupon payments per year. For annual payments, **frequency** = 1; for semiannual, **frequency** = 2; for quarterly, **frequency** = 4
- **basis:** this last argument is the only one that is optional. This is the type of day count **basis** to use. There are five options:

Basis	Day count basis
0 or omitted	US (NASD) 30 / 360
1	Actual / actual
2	Actual / 360
3	Actual / 365
4	European 30 / 360

It should be further noted that:

- Microsoft Excel stores dates as sequential serial numbers so they can be used in calculations. By default, January 1, 1900 is serial number 1, and January 1, 2008 is serial number 39448 because it is 39,448 days after January 1, 1900
- dates should be entered using the **DATE** function, or as results of other formulae or functions. For example, use **=DATE(2020,2,29)** for the 29th of February, 2020. Problems may occur if dates are entered as text
- the **settlement** date is the date a buyer purchases a coupon, such as a bond. The **maturity** date is the date when a coupon expires. For example, suppose a 30-year bond is issued on January 1, 2024, and is purchased by a buyer six months later. The **issue** date would be January 1, 2024, the **settlement** date would be July 1, 2024, and the **maturity** date would be January 1, 2054, which is 30 years after the January 1, 2024, **issue** date
- **settlement, maturity, issue, first_coupon** and **basis** are all truncated to integers
- if **settlement, maturity, issue** or **first_coupon** is not a valid date, **ODDFPRICE** returns the **#VALUE!** error value
- if **rate** < 0 or if **yield** < 0, **ODDFPRICE** returns the **#NUM!** error value
- if **basis** < 0 or if **basis** > 4, **ODDFPRICE** returns the **#NUM!** error value
- the following date condition must be satisfied; otherwise, **ODDFPRICE** returns the **#NUM!** error value:
maturity > **first_coupon** > **settlement** > **issue**
- **ODDFPRICE** is calculated as follows. For an odd, short first coupon:

$$\begin{aligned}
 ODDFPRICE = & \left[\frac{\text{redemption}}{\left(1 + \frac{\text{yld}}{\text{frequency}}\right)^{\left(N + \frac{DSC}{E}\right)}} \right] + \left[\frac{100 \times \frac{\text{rate}}{\text{frequency}} \times \frac{DFC}{E}}{\left(1 + \frac{\text{yld}}{\text{frequency}}\right)^{\frac{DSC}{E}}} \right] \\
 & + \left[\sum_{k=2}^N \frac{100 \times \frac{\text{rate}}{\text{frequency}}}{\left(1 + \frac{\text{yld}}{\text{frequency}}\right)^{\left(k + \frac{DSC}{E}\right)}} \right] \\
 & - \left[100 \times \frac{\text{rate}}{\text{frequency}} \times \frac{A}{E} \right]
 \end{aligned}$$

where:

- **A** = number of days from the beginning of the coupon period to the **settlement** date (accrued days)
 - **DSC** = number of days from the **settlement** to the next coupon date
 - **DFC** = number of days from the beginning of the odd first coupon to the first coupon date
 - **E** = number of days in the coupon period
 - **N** = number of coupons payable between the **settlement** date and the **redemption** date (if this number contains a fraction, it is raised to the next whole number)
- For an odd, long first coupon:

$$\begin{aligned}
 ODDFPRICE = & \left[\frac{\text{redemption}}{\left(1 + \frac{\text{yld}}{\text{frequency}}\right)^{\left(N + N_q \frac{DSC}{E}\right)}} \right] \\
 & + \left[\frac{100 \times \frac{\text{rate}}{\text{frequency}} \times \left[\sum_{i=1}^{NC} \frac{DC_i}{NL_i} \right]}{\left(1 + \frac{\text{yld}}{\text{frequency}}\right)^{\left(N_q \frac{DSC}{E}\right)}} \right] \\
 & + \left[\sum_{A=1}^N \frac{100 \times \frac{\text{rate}}{\text{frequency}}}{\left(1 + \frac{\text{yld}}{\text{frequency}}\right)^{\left(k + N_q \frac{DSC}{E}\right)}} \right] \\
 & - \left[100 \times \frac{\text{rate}}{\text{frequency}} \times \sum_{i=1}^{NC} \frac{A_i}{NL_i} \right]
 \end{aligned}$$

where:

- **A_i** = number of days from the beginning of the *i*th, or last, quasi-coupon period within odd period
- **DC_i** = number of days from dated date (or issue date) to first quasi-coupon (*i* = 1) or number of days in quasi-coupon (*i* = 2, ..., *i* = **NC**)
- **DSC** = number of days from **settlement** to next coupon date
- **E** = number of days in coupon period
- **N** = number of coupons payable between the first real coupon date and **redemption** date (if this number contains a fraction, it is raised to the next whole number)
- **NC** = number of quasi-coupon periods that fit in odd period (if this number contains a fraction, it is raised to the next whole number)
- **NL_i** = normal length in days of the full *i*th, or last, quasi-coupon period within odd period
- **N_q** = number of whole quasi-coupon periods between settlement date and first coupon.

Please see our example below:

	A	B	C
1	Data	Description	
2	11-Nov-24	Settlement date	
3	01-Mar-37	Maturity date	
4	15-Oct-24	Issue date	
5	01-Mar-25	First coupon date	
6	7.85%	Percent coupon	
7	6.25%	Percent yield	
8	100	Redemptive value	
9	2	Frequency is semiannual	
10	1	Actual / actual basis	
11			
12			
13	Formula	Description	Result
14	=ODDFPRICE(A2,A3,A4,A5,A6,A7,A8,A9,A10)	The price per \$100 face value of a security having an odd (short or long) first period, for the bond using the terms in cells A2:A10 as arguments for the function.	\$113.60

The A to Z of Excel Functions: ODDFYIELD



In financial analysis, some bonds come with irregular first or last periods. Due to the irregular first or last period, the payment doesn't fit in any of the usual or standard patterns. If you wish to calculate the yield of a security with an "odd" first period (either short or long), we can use the **ODDFYIELD** function.

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

ODDFYIELD(settlement, maturity, issue, first_coupon, rate, price, redemption, frequency, [basis])

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

- **settlement**: this is required. This is the security's settlement date. The security settlement date is the date after the issue date when the security is traded to the buyer
- **maturity**: this is also required. This represents the security's maturity date. The maturity date is the date when the security expires
- **issue**: again, required. This is the security's issue date
- **first_coupon**: required. This denotes the security's first coupon date
- **rate**: yet again required. This is the security's interest rate
- **price**: this is also required. This is the security's price
- **redemption**: required. The security's redemption value per \$100 face value
- **frequency**: also required. This considers the number of coupon payments per year. For annual payments, **frequency** = 1; for semiannual, **frequency** = 2; for quarterly, **frequency** = 4
- **basis**: this last argument is the only one that is optional. This is the type of day count **basis** to use. There are five options:

Basis	Day count basis
0 or omitted	US (NASD) 30 / 360
1	Actual / actual
2	Actual / 360
3	Actual / 365
4	European 30 / 360

It should be further noted that:

- Microsoft Excel stores dates as sequential serial numbers so they can be used in calculations. By default, January 1, 1900 is serial number 1, and January 1, 2008 is serial number 39448 because it is 39,448 days after January 1, 1900
- dates should be entered using the **DATE** function, or as results of other formulae or functions. For example, use **=DATE(2020,2,29)** for the 29th of February, 2020. Problems may occur if dates are entered as text
- the **settlement** date is the date a buyer purchases a coupon, such as a bond. The **maturity** date is the date when a coupon expires. For example, suppose a 30-year bond is issued on January 1, 2024, and is purchased by a buyer six months later. The **issue** date would be January 1, 2024, the **settlement** date would be July 1, 2024, and the **maturity** date would be January 1, 2054, which is 30 years after the January 1, 2024, **issue** date
- **settlement**, **maturity**, **issue**, **first_coupon** and **basis** are all truncated to integers
- if **settlement**, **maturity**, **issue** or **first_coupon** is not a valid date, **ODDFYIELD** returns the **#VALUE!** error value
- if **rate** < 0 or if **price** ≤ 0, **ODDFYIELD** returns the **#NUM!** error value
- if **basis** < 0 or if **basis** > 4, **ODDFYIELD** returns the **#NUM!** error value
- the following date condition must be satisfied; otherwise, **ODDFYIELD** returns the **#NUM!** error value:
maturity > **first_coupon** > **settlement** > **issue**
- Excel uses an iterative technique to calculate **ODDFYIELD**. This function uses the Newton method based upon the formula used for the function **ODDFPRICE**. The **yield** is changed through 100 iterations until the estimated price with the given **yield** is close to the **price**
- **ODDFYIELD** employs the **ODDFPRICE** formulae in its calculations as follows. For an odd, short first coupon:

$$\begin{aligned}
 \text{ODDFPRICE} = & \left[\frac{\text{redemption}}{\left(1 + \frac{\text{yld}}{\text{frequency}}\right)^{\left(N + \frac{\text{DSC}}{E}\right)}} \right] + \left[\frac{100 \times \frac{\text{rate}}{\text{frequency}} \times \frac{\text{DFC}}{E}}{\left(1 + \frac{\text{yld}}{\text{frequency}}\right)^{\frac{\text{DSC}}{E}}} \right] \\
 & + \left[\sum_{k=2}^N \frac{100 \times \frac{\text{rate}}{\text{frequency}}}{\left(1 + \frac{\text{yld}}{\text{frequency}}\right)^{\left(k + \frac{\text{DSC}}{E}\right)}} \right] \\
 & - \left[100 \times \frac{\text{rate}}{\text{frequency}} \times \frac{A}{E} \right]
 \end{aligned}$$

where:

- **A** = number of days from the beginning of the coupon period to the **settlement** date (accrued days)
- **DSC** = number of days from the **settlement** to the next coupon date
- **DFC** = number of days from the beginning of the odd first coupon to the first coupon date
- **E** = number of days in the coupon period
- **N** = number of coupons payable between the **settlement** date and the **redemption** date (if this number contains a fraction, it is raised to the next whole number)

For an odd, long first coupon:

$$\begin{aligned}
 \text{ODDFPRICE} = & \left[\frac{\text{redemption}}{\left(1 + \frac{\text{yld}}{\text{frequency}}\right)^{\left(N + N_q \frac{\text{DSC}}{E}\right)}} \right] \\
 & + \left[\frac{100 \times \frac{\text{rate}}{\text{frequency}} \times \left[\sum_{j=1}^{NC} \frac{DC_j}{NL_j} \right]}{\left(1 + \frac{\text{yld}}{\text{frequency}}\right)^{\left(N_q \frac{\text{DSC}}{E}\right)}} \right] \\
 & + \left[\sum_{k=1}^N \frac{100 \times \frac{\text{rate}}{\text{frequency}}}{\left(1 + \frac{\text{yld}}{\text{frequency}}\right)^{\left(k - N_q + \frac{\text{DSC}}{E}\right)}} \right] \\
 & - \left[100 \times \frac{\text{rate}}{\text{frequency}} \times \sum_{j=1}^{NC} \frac{A_j}{NL_j} \right]
 \end{aligned}$$

where:

- A_i = number of days from the beginning of the i th, or last, quasi-coupon period within odd period
- DC_i = number of days from dated date (or issue date) to first quasi-coupon ($i = 1$) or number of days in quasi-coupon ($i = 2, \dots, i = NC$)
- **DSC** = number of days from **settlement** to next coupon date
- **E** = number of days in coupon period
- **N** = number of coupons payable between the first real coupon date and **redemption** date (if this number contains a fraction, it is raised to the next whole number)
- **NC** = number of quasi-coupon periods that fit in odd period (if this number contains a fraction, it is raised to the next whole number)
- NL_i = normal length in days of the full i th, or last, quasi-coupon period within odd period
- N_q = number of whole quasi-coupon periods between settlement date and first coupon.

Please see the next example below:

	A	B	C
1	Data	Description	
2	11-Nov-24	Settlement date	
3	01-Mar-37	Maturity date	
4	15-Oct-24	Issue date	
5	01-Mar-25	First coupon date	
6	5.75%	Percent coupon	
7	84.50	Price	
8	100	Redemptive value	
9	2	Frequency is semiannual	
10	1	Actual / actual basis	
11			
12			
13	Formula	Description	Result
14	=ODDFYIELD(A2,A3,A4,A5,A6,A7,A8,A9,A10)	The yield of a security that has an odd (short or long) first period, for the bond using the terms in cells A2:A10 as function arguments. The result is 0.0772 or 7.72%.	7.72%

The A to Z of Excel Functions: ODDLPRICE



This one does an Oddjob [*groan – Ed.*].

The **ODDLPRICE** function is an Excel financial function that calculates the price per \$100 face value of a security or bond with an irregular last period. That's what is meant by "odd"! The L isn't an expression for where you end up if you were to use this function, but rather denoting this calculates for the last period.

In financial analysis, some bonds come with irregular first or last periods. Due to the irregular first or last period, the payment doesn't fit in any of the usual or standard patterns. If you wish to calculate the price of a bond with an "odd" last period (either short or long), we can use the **ODDLPRICE** function.

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

ODDLPRICE(settlement, maturity, last_interest, rate, yield, redemption, frequency, [basis])

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

- **settlement:** this is required. This is the security's settlement date. The security settlement date is the date after the issue date when the security is traded to the buyer
- **maturity:** this is also required. This represents the security's maturity date. The maturity date is the date when the security expires
- **last_interest:** required. This denotes the security's last coupon date
- **rate:** yet again required. This is the security's interest rate
- **yield:** this is also required. This is the security's annual yield
- **redemption:** required. The security's redemption value per \$100 face value
- **frequency:** also required. This considers the number of coupon payments per year. For annual payments, **frequency** = 1; for semiannual, **frequency** = 2; for quarterly, **frequency** = 4
- **basis:** this last argument is the only one that is optional. This is the type of day count **basis** to use. There are five options:

Basis	Day count basis
0 or omitted	US (NASD) 30 / 360
1	Actual / actual
2	Actual / 360
3	Actual / 365
4	European 30 / 360

It should be further noted that:

- Microsoft Excel stores dates as sequential serial numbers so they can be used in calculations. By default, January 1, 1900 is serial number 1, and January 1, 2008 is serial number 39448 because it is 39,448 days after January 1, 1900
- dates should be entered using the **DATE** function, or as results of other formulae or functions. For example, use **=DATE(2020,2,29)** for the 29th of February, 2020. Problems may occur if dates are entered as text
- the **settlement** date is the date a buyer purchases a coupon, such as a bond. The **maturity** date is the date when a coupon expires. For example, suppose a 30-year bond is issued on January 1, 2024, and is purchased by a buyer six months later. The **issue** date would be January 1, 2024, the **settlement** date would be July 1, 2024, and the **maturity** date would be January 1, 2054, which is 30 years after the January 1, 2024, **issue** date
- **settlement**, **maturity**, **last_interest** and **basis** are all truncated to integers
- if **settlement**, **maturity** or **last_interest** is not a valid date, **ODDLPRICE** returns the **#VALUE!** error value
- if **rate** < 0 or if **yield** < 0, **ODDLPRICE** returns the **#NUM!** error value
- if **basis** < 0 or if **basis** > 4, **ODDLPRICE** returns the **#NUM!** error value
- the following date condition must be satisfied; otherwise, **ODDLPRICE** returns the **#NUM!** error value:
maturity > **settlement** > **last_interest**.

Please see our penultimate example below:

	A	B	C
1	Data	Description	
2	07-Feb-24	Settlement date	
3	15-Jun-24	Maturity date	
4	25-Oct-23	Last interest date	
5	3.75%	Percent coupon	
6	4.05%	Percent yield	
7	100	Redemptive value	
8	2	Frequency is semiannual	
9	0	30 / 360 basis	
10			
11			
12	Formula	Description	Result
13	=ODDLPRICE(A2,A3,A4,A5,A6,A7,A8,A9)	The price per \$100 of a security having an odd (short or long) last coupon period, for a bond using the terms in cells A2:A9 as function arguments.	\$99.88

The A to Z of Excel Functions: ODDLYIELD



We always thought **ODDLYIELD** was the perfect collective noun for a group of financial modellers...

In financial analysis, some bonds come with irregular first or last periods. Due to the irregular first or last period, the payment doesn't fit in any of the usual or standard patterns. If you wish to calculate the yield of a security with an "odd" Last period (either short or long), we can use the **ODDLYIELD** function.

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

ODDLYIELD(settlement, maturity, last_interest, rate, price, redemption, frequency, [basis])

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

- **settlement:** this is required. This is the security's settlement date. The security settlement date is the date after the issue date when the security is traded to the buyer
- **maturity:** this is also required. This represents the security's maturity date. The maturity date is the date when the security expires
- **last_interest:** required. This denotes the security's last coupon date
- **rate:** yet again required. This is the security's interest rate
- **price:** this is also required. This is the security's price
- **redemption:** required. The security's redemption value per \$100 face value
- **frequency:** also required. This considers the number of coupon payments per year. For annual payments, **frequency** = 1; for semiannual, **frequency** = 2; for quarterly, **frequency** = 4
- **basis:** this last argument is the only one that is optional. This is the type of day count **basis** to use. There are five options:

Basis	Day count basis
0 or omitted	US (NASD) 30 / 360
1	Actual / actual
2	Actual / 360
3	Actual / 365
4	European 30 / 360

It should be further noted that:

- Microsoft Excel stores dates as sequential serial numbers so they can be used in calculations. By default, January 1, 1900 is serial number 1, and January 1, 2008 is serial number 39448 because it is 39,448 days after January 1, 1900
- dates should be entered using the **DATE** function, or as results of other formulae or functions. For example, use **=DATE(2020,2,29)** for the 29th of February, 2020. Problems may occur if dates are entered as text
- the **settlement** date is the date a buyer purchases a coupon, such as a bond. The **maturity** date is the date when a coupon expires. For example, suppose a 30-year bond is issued on January 1, 2024, and is purchased by a buyer six months later. The **issue** date would be January 1, 2024, the **settlement** date would be July 1, 2024, and the **maturity** date would be January 1, 2054, which is 30 years after the January 1, 2024, **issue** date
- **settlement**, **maturity**, **last_interest** and **basis** are all truncated to integers
- if **settlement**, **maturity** or **last_interest** is not a valid date, **ODDLYIELD** returns the **#VALUE!** error value
- if **rate** < 0 or if **price** ≤ 0, **ODDLYIELD** returns the **#NUM!** error value
- if **basis** < 0 or if **basis** > 4, **ODDLYIELD** returns the **#NUM!** error value
- the following date condition must be satisfied; otherwise, **ODDLYIELD** returns the **#NUM!** error value:
maturity > **settlement** > **last_interest**
- **ODDLYIELD** is calculated as follows:

$$\text{ODDLYIELD} = \left[\frac{\left(\text{redemption} + \left(\sum_{i=1}^{MC} \frac{DC_i}{NL_i} \times \frac{100 \times \text{rate}}{\text{frequency}} \right) \right) - \left(\text{par} + \left(\sum_{i=1}^{MC} \frac{A_i}{NL_i} \times \frac{100 \times \text{rate}}{\text{frequency}} \right) \right)}{\text{par} + \left(\sum_{i=1}^{MC} \frac{A_i}{NL_i} \times \frac{100 \times \text{rate}}{\text{frequency}} \right)} \right] \times \left[\frac{\text{frequency}}{\left(\sum_{i=1}^{MC} \frac{DSC_i}{NL_i} \right)} \right]$$

where:

- A_i = number of accrued days for the i th, or last, quasi-coupon period within odd period counting forward from last interest date before **redemption**
- DC_i = number of days counted in the i th, or last, quasi-coupon period as delimited by the length of the actual coupon period
- NC = number of quasi-coupon periods that fit in the odd period; if this number contains a fraction it will be raised to the next whole number
- NL_i = normal length in days of the i th, or last, quasi-coupon period within the odd coupon period.

Please see our final example for this month below:

	A	B	C
1	Data	Description	
2	20-Apr-24	Settlement date	
3	15-Jun-24	Maturity date	
4	24-Dec-23	Last interest date	
5	3.75%	Percent coupon	
6	99.875	Percent yield	
7	100	Redemptive value	
8	2	Frequency is semiannual	
9	0	30 / 360 basis	
10			
11			
12	Formula	Description	Result
13	<code>=ODDLYIELD(A2,A3,A4,A5,A6,A7,A8,A9)</code>	The yield of a security that has an odd (short or long) last period, for the bond using the terms in cells A2:A9 as function arguments. The result is 0.04519, or 4.52%.	4.52%

More Excel Functions next month.

Beat the Boredom Suggested Solution

Earlier in this newsletter, we raised a challenge with an all too familiar example:

The screenshot shows an Excel spreadsheet with the following content:

- Cell F12: `=5*7`
- Cell F13: `=SQRT(PI())`
- Cell F14: `=CHOOSE(2,F12,F13)`

The spreadsheet also displays the results of these formulas in a table:

Make These Formulae	Result
<code>=5*7</code>	35
<code>=SQRT(PI())</code>	1.772453851
<code>=CHOOSE(2,F12,F13)</code>	1.772453851

Here, cell **F12** (pictured) had the “pseudo-formula”:

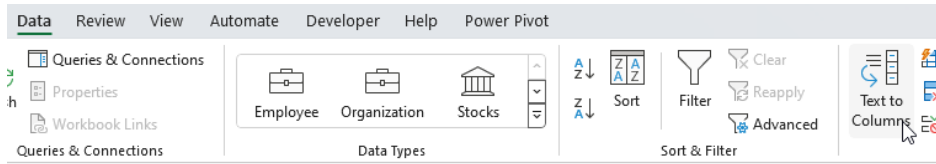
`'=5*7`

It should be made clear that the cells **F12:F14** are not formatted as Text. They do not calculate as they are deemed text since they all begin with an apostrophe ('). This month's challenge was to remove all the apostrophes **at the same time** (without using Power Query, VBA or other similar code). How did you get on?

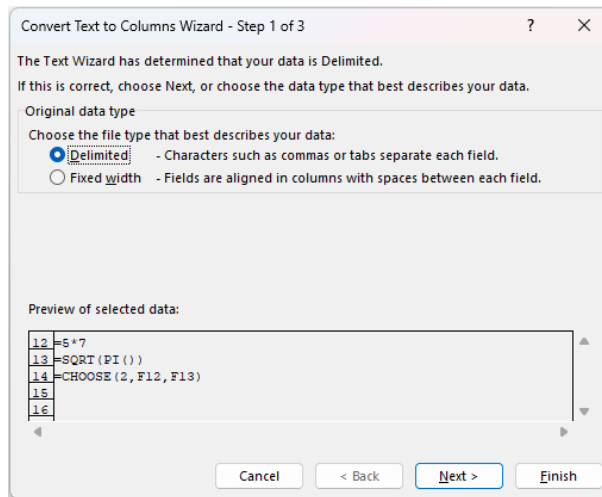
Suggested Solution

Your first thought might be to use 'Replace...' (**CTRL + H**) and replace '= with =. Unfortunately, this does not work in all versions of Excel as 'Replace...' does not seem to recognise apostrophes in certain instances.

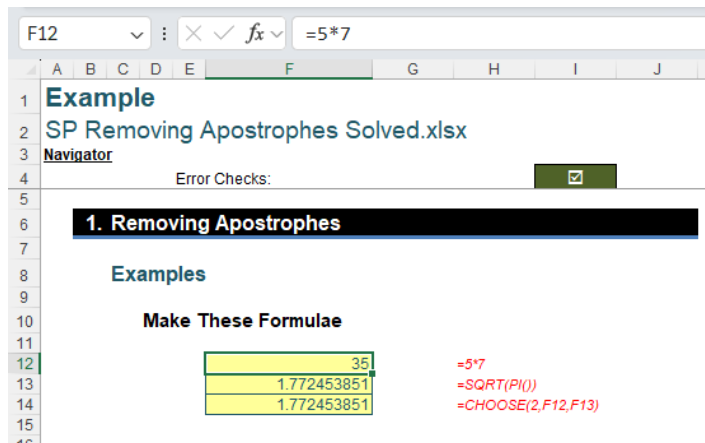
However, there is a very simple trick to circumvent this problem. With this data still selected, click on the 'Text to Columns' button in the 'Data Tools' group of the 'Data' tab on the Ribbon (**ALT + D + E**):



This launches the 'Text to Columns Wizard' dialog box. In the first step, ensure that the '...file type that best describes your data...' is set to 'Delimited':



Then, believe it or not, simply depress the 'Finish' button. The spreadsheet will then reinstate the formulae, viz.



Simple!

Word to the Wise

This challenge was incredibly simple *if you knew what to do*. Again, we offer no apology because this is a trick you should add to your repertoire, if you're not already aware of it.

Don't worry; normal service will be resumed next month.

Upcoming SumProduct Training Courses

Location	Course	Course Date	Local Time	UTC	Duration
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
London UK	Financial Modelling	13 January 2025 - 14 January 2025	09:00 - 17:00 GMT	13 January 2025 09:00 UTC - 14 January 2025 16:00 UTC	2 Days
Melbourne Australia	Power Pivot, Power Query and Power BI	14 January 2025 - 15 January 2025	09:00 - 17:00 AEDT	13 January 2025 22:00 UTC - 15 January 2025 06:00 UTC	2 Days
Melbourne Australia	ChatGPT	15 January 2025 - 16 January 2025	09:00 - 17:00 AEDT	14 January 2025 22:00 UTC - 16 January 2025 06:00 UTC	2 Days
Melbourne Australia	Excel Tips and Tricks	20 January 2025	09:00 - 17:00 AEDT	19 January 2025 22:00 UTC - 20 January 2025 06:00 UTC	1 Day
Sydney Australia	Excel Tips and Tricks	22 January 2025	09:00 - 17:00 AEDT	21 January 2025 22:00 UTC - 22 January 2025 06:00 UTC	1 Day
London UK	ChatGPT	22 January 2025 - 23 January 2025	09:00 - 17:00 GMT	22 January 2025 09:00 UTC - 23 January 2025 17:00 UTC	2 Days
London UK	Financial Modelling	17 February 2025 - 18 February 2025	09:00 - 17:00 BST	17 February 2025 09:00 UTC - 18 February 2025 16:00 UTC	2 Days
Melbourne Australia	Power Pivot, Power Query and Power BI	18 February 2025 - 19 February 2025	09:00 - 17:00 AEDT	17 February 2025 22:00 UTC - 19 February 2025 06:00 UTC	2 Days
Melbourne Australia	ChatGPT	19 February 2025 - 20 February 2025	09:00 - 17:00 AEDT	18 February 2025 22:00 UTC - 20 February 2025 06:00 UTC	2 Days

Key Strokes

Each newsletter, we'd like to introduce you to useful keystrokes you may or may not be aware of. This time, we thought we'd play a full game of 'Twister' on the keyboard, using **CTRL**, **ALT** and **SHIFT** with the function keys:

Keystroke	What it does
CTRL + ALT + SHIFT + F2	Print
CTRL + ALT + SHIFT + F4	Close application
CTRL + ALT + SHIFT + F9	Recalculation: full rebuild

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

Our Services

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

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

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

Link to Others

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

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

Any Questions?

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

Training

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

Check out our more popular courses in our training brochure:



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

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

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