SumProduct

NEWSLETTER #122 - January 2023

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

Happy New Year! Our lead article might seem a little dated this month, but don't let it deter you from digging into this month's newsletter. The new year starts off with all the regulars that drew the curtains on 2022.

You may recall those regulars did not include Power BI Updates, which obviously came back a little worse from the Christmas break. It looks a little green to me, but more on that later...

The others celebrated our 10th birthday happily though and start as they mean to go on for '23. There is our usual Beat the Boredom Challenge, plus Charts & Dashboards, Visual Basics, Power Pivot Principles, Power Query Pointers and Excel Updates, which are all present and correct, except for any typos. We do so much on the **IS** functions this month in the A to Z of Excel functions series you could argue they should be renamed **ARE** and we finish off with a myriad of **CTRL + ALT** keyboard shortcuts.

Liam Bastick, Managing Director, SumProduct

New Year's Resolutions Getting Dated..?

There comes a time in most people's lives that they realise it's time to confront the world head-on, go out there and ask for a date. And what better time to resolve such a course of action than at the turn of the New Year! You can imagine asking a model for a date can be even more stressful as these jokes get older and, er, more dated *(please stop – Ed.)*. Well puns now completed – yup, I've stopped them – period!), it's not that difficult to master dates, but it does deserve a discussion.

Most forecast models project key outputs over multiple periods. Typically, these periods are not headed "Time 1", "Time 2", *etc.* like in our usual generic examples, but display end dates to assist users to understand payback periods, seasonality, trends and so on.

An example time series could contain some or all of the following:

Month Ending	Jan-20	Feb-20	Mar-20	Apr-20	May-20	Jun-20
Month	M12	M1	M2	M3	M4	M5
Period End Year	2020	2020	2020	2020	2020	2020
Financial Year	2020	2021	2021	2021	2021	2021
Days in Period End Year	366	366	366	366	366	366
Days in Financial Year	365	366	366	366	366	366
Financial Year Period	M12	M1	M2	M3	M4	M5
Start Date	1/1/20	1/2/20	1/3/20	1/4/20	1/5/20	1/6/20
End Date	31/1/20	29/2/20	31/3/20	30/4/20	31/5/20	30/6/20
Months in Month	1	1	1	1	1	1
Days in Period	31	29	31	30	31	30
Days in Month	31	29	31	30	31	30
Counter	1	2	3	4	5	6

The above is an example of time series data in a real-life financial model. My question is this though: are all of these rows absolutely necessary?

I would suggest not.

Essentially, three lines are necessarily needed when modelling (the rest may be derived as necessary):

- **Start date:** This will allow for models where the first period is not a "full" period (often called a 'stub' period), *e.g.* a business may wish to project its profits from now until the end of the calendar year for the first year;
- **End date:** This will define the end of the period and will often coincide with reporting dates, *e.g.* end of financial year or quarter ends. By having both the start date and end date defined, a modeller can determine the number of days / weeks / months in the period, which financial year the period pertains too and so forth;
- **Counter:** Start and end dates are insufficient. Constructing calculations based on consideration of a date is fraught with potential issues in Excel. This is because dates are really serial numbers in Excel which may differ depending upon the underlying operating system (*e.g.* Day 1 for Microsoft Excel for Windows is 1 January 1900, whilst Day 1 is 1 January 1904 for earlier versions of Microsoft Excel for the Macintosh). Further, if you are building a monthly model you may wish to divide an annual figure evenly instead of based on the number of days. This is also the easiest way to identify the first and last periods in a robust manner.

So, bearing this in mind, how do you build up the necessary formulae for these three line items allowing for the more common eventualities? Well, to begin with, there's only really one troublesome formula. This is because:

- The counter is simply the last period's number plus one. I tend to use the formula **=N(Previous_Cell)+1**, where the **N()** function takes the numerical value in the previous cell, and more importantly, text is ignored so that *#VALUE!* errors will not arise;
- The start date is simply the **Model Start Date** for the first period and the day following the last period's end date otherwise. This can simply be written as

  =IF(Counter=1,Model_Start_Date,Previous_Period_End_Date+1).

Therefore, we need only consider the formula for the **Period End Date**. Consider the following simple example:

1. Timing Assumptions

Data (do not change once modelling has commenced)

Model Start Date	17 Jan 20	
Number of Months in a Full Period	3	
Example Reporting Month	12	*e.g. 31-Dec-20*
Reporting Month Factor	3	
Months per Year	12	

I have selected an arbitrary start date (**Model_Start_Date**) of 17 January 2020, and assumed that the number of months in a full period (**Periodicity**) is 3. The third line item is a little more subtle: this specifies which periods are period ends by specifying one month that will be a period end month. For example, tax may be paid quarterly in the months of January, April, July and October. By entering a **Periodicity** of 3 and specifying an **Example_Reporting_Month** of any of 1, 4, 7 or 10, this will provide sufficient information to work out the quarter ends, i.e. 31-Jan, 30-Apr, 31-Jul and 31-Oct. The **Reporting_Month_Factor** is simply the minimum of these acceptable alternative values and is calculated automatically here. The approach I will use here requires that the periodicity is a divisor of the number of months in a year (**Months_in_Year**) – which is why my example only allows the **Periodicity** to be 1, 2, 3, 4, 6 or 12.

In the example above, we are building a quarterly model where December is one of the quarter ends. Therefore, the possible quarter end months are:

End Date Month
3
6
9
12

This example table allows for up to 12 month ends (*i.e.* for a monthly model).

So how do we derive the necessary formula? I will give you some insight into my simplistic view of the world. First, I would construct the following table:

Model Start Date Month	End Date Month	Addition Reqd
1	3	2
2	3	1
3	3	0
4	6	2
5	6	1
6	6	0
7	9	2
8	9	1
9	9	0
10	12	2
11	12	1
12	12	0

This simple table considers all 12 months of the year for the **Model_Start_Date** (first column). The middle column displays which month would be the first quarter, given the assumption regarding the month of the **Model_Start_Date**. Therefore, a start date in January, February or March will give rise to a March quarter end, *etc.*

I can use an array formula to calculate this month number dynamically. This is not necessary, and the values could just be typed in – remember, this table is simply a tool to ascertain how to construct the formula required.

The final column is then the difference between the end date month and the **Model_Start_Date** month. It is slightly more complicated than this as we need to consider what happens if the **Model_Start_Date** month exceeds the final end date month. For example, in my tax example above, tax arising in November (month 11) is after the final payment period of the year (month 10). This would be paid in month 1 of the following year instead.

The point is, the final column highlights the pattern of how many months after the **Model_Start_Date** the first reporting period will occur. We can now use two functions in tandem to derive this first period end date:

- **EOMONTH(Date,Months)** returns the last day of the month so many months from the date specified. For example, **=EOMONTH(11 Dec-20,14)** would be 28-Feb-22, *i.e.* the end date 14 months after the end of December 2020.
- **MOD(Number,Divisor)** returns the remainder when **Number** is divided by the **Divisor**. For example, **=MOD(17,6)** is 5, since 17/6 = 2 remainder 5.

With trial and error, the number of months we need to add on can be calculated as follows:

=MOD(Perodicity+Reporting_Month_Factor-MONTH(Model_Start_Date),Perodicity)

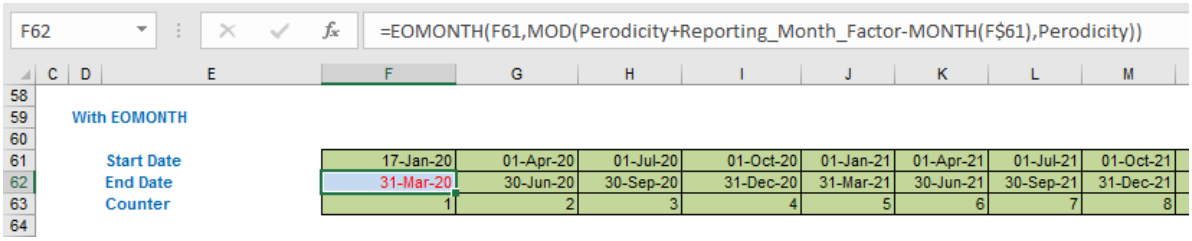
and therefore, if we call this equation our **Additive_Factor**, then the reporting end date will be:

=EOMONTH(Model_Start_Date,Additive_Factor)

In the example file, I have checked my workings, *viz*.

Model Start Date Month	End Date Month	Addition Reqd	Dates Calc	Verification
1	3	2	2	3
2	3	1	1	3
3	3	0	0	3
4	6	2	2	6
5	6	1	1	6
6	6	0	0	6
7	9	2	2	9
8	9	1	1	9
9	9	0	0	9
10	12	2	2	12
11	12	1	1	12
12	12	0	0	12

Furthermore, a robust yet flexible time series can be constructed:



Word to the Wise

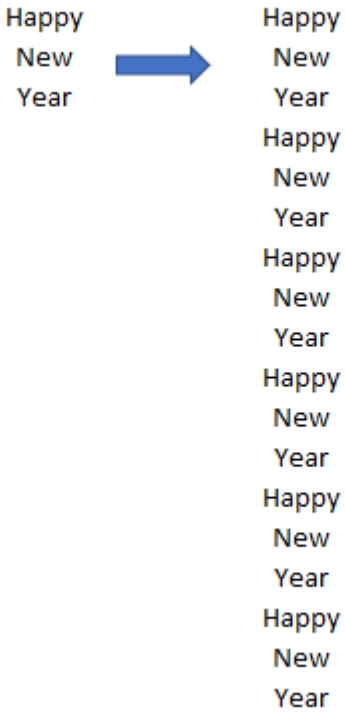
Even allowing for flexible start dates and "Reporting Month Factors", the above will not work in all circumstances. Other periodicities may be sought, whilst some businesses require weekly reporting, "last Thursday of the month" or 5-4-4 week period regimes. Nonetheless, the above approach can be modified and extrapolated to consider most complications.

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…*

Time for a seasonal challenge as we welcome in 2023. The challenge here is to spill a range of cells – "Happy New Year" – multiple times using Excel formulae…

Now when we say "spill", we are referring to an Excel formula that returns a range of cells according to user's needs. Data often needs to be organised and arranged in different ways. Now, the challenge for this is to spill "Happy", "New" and "Year" six (6) times each as shown in the example below:



This challenge is designed to get you to think outside the box and beyond your basic functions of copy and paste. It's not the hardest challenge ever this month, and you never know, "spill" may "spill the beans"…

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

Charts and Dashboards

It's time to chart our progress with an introductory series into the world of creating charts and dashboards in Excel. This month, we continue considering how to hide data.

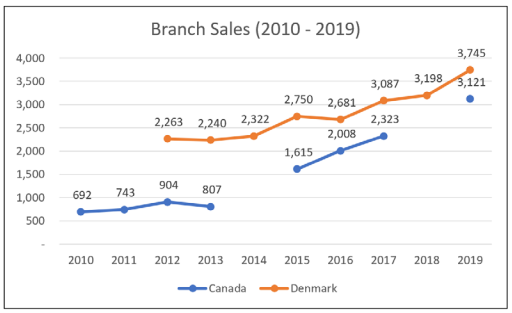
Last month, we talked about chart data with missing values and different ways to hide these 'blanks' on charts. This time, we'll discuss some tips and tricks related to hiding data *in general*. For example, imagine we have sales data as below, with some missing data, and we present those missing points in the chart as gaps, *viz.*

Chart Data

Branch Sales (2010-2019)

	2010	2011	2012	2013	2014	2015	2016	2017	2018	2019
Canada	692	743	904	807		1,615	2,008	2,323		3,121
Denmark			2,263	2,240	2,322	2,750	2,681	3,087	3,198	3,745

Chart



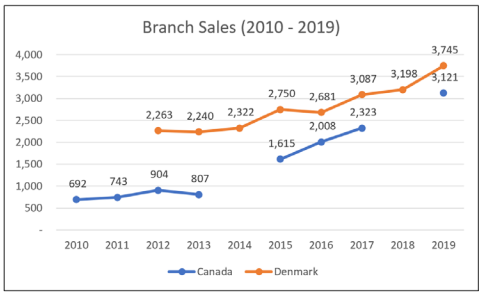
To better lay out our dashboard, we wish to group our data and chart in separate sections, by highlighting the rows we want to group, navigating to the Data tab on the Ribbon and clicking Group, or else **ALT + A + G + G**:

Chart Data

Branch Sales (2010-2019)

	2010	2011	2012	2013	2014	2015	2016	2017	2018	2019
Canada	692	743	904	807		1,615	2,008	2,323		3,121
Denmark			2,263	2,240	2,322	2,750	2,681	3,087	3,198	3,745

Chart



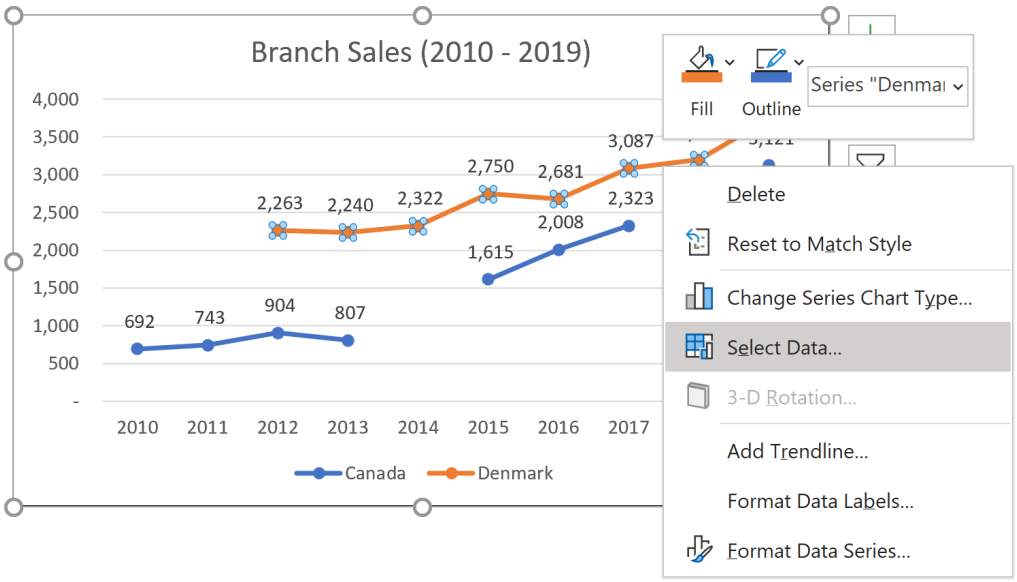
When we hide our data in order to show just the chart … oops … where is the chart!?

Chart Data

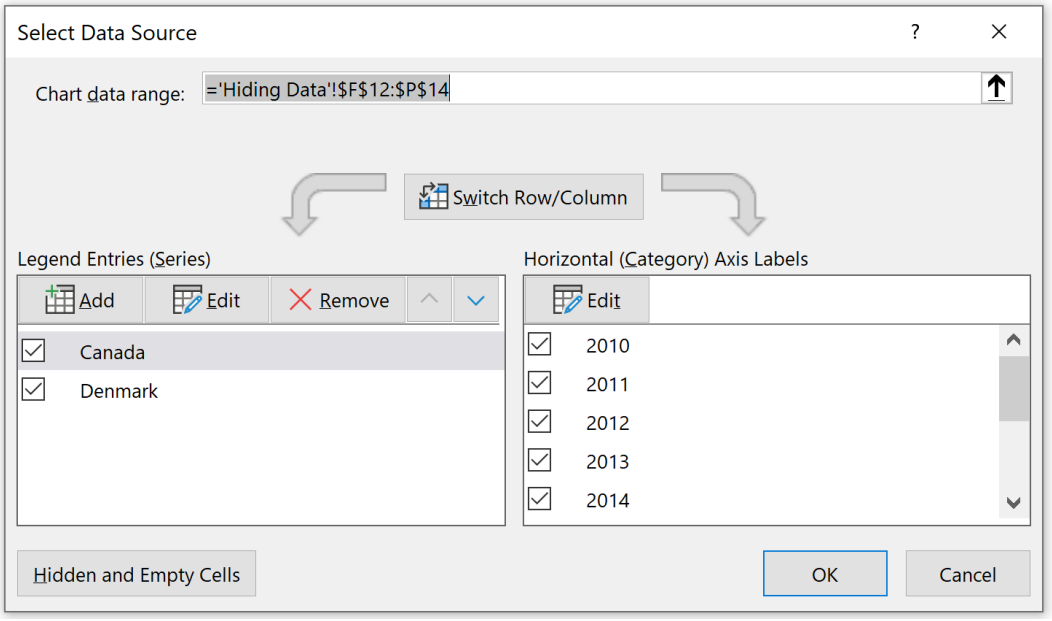
Branch Sales (2010-2019)

Chart

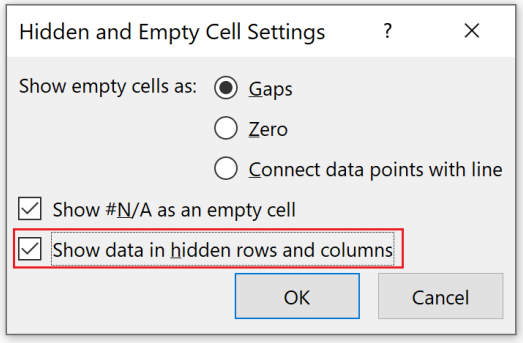
…The chart is hidden as we have hidden the grouped rows relating to the chart data! To fix this, open the grouped rows (so that we may see the chart), then right-click on the data series and choose 'Select Data…':



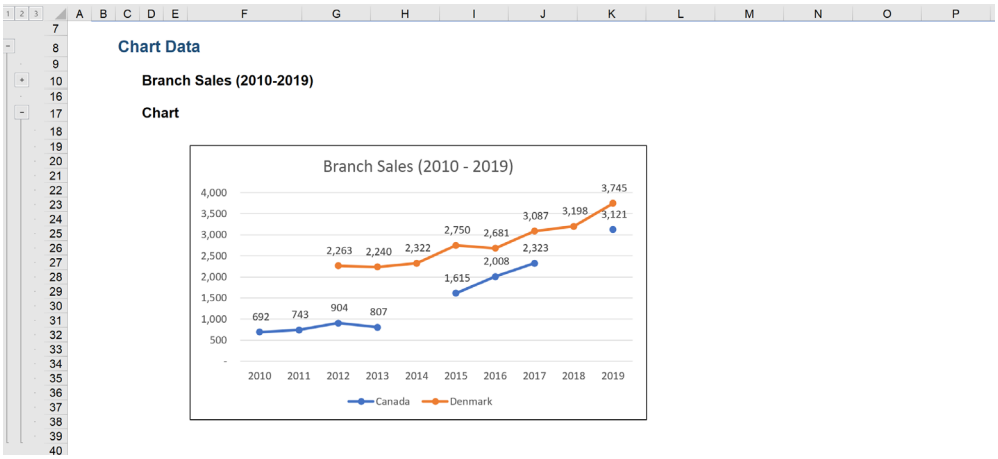
The 'Select Data Source' dialog will appear. In here, go to the 'Hidden and Empty Cells' in the left-hand bottom corner:



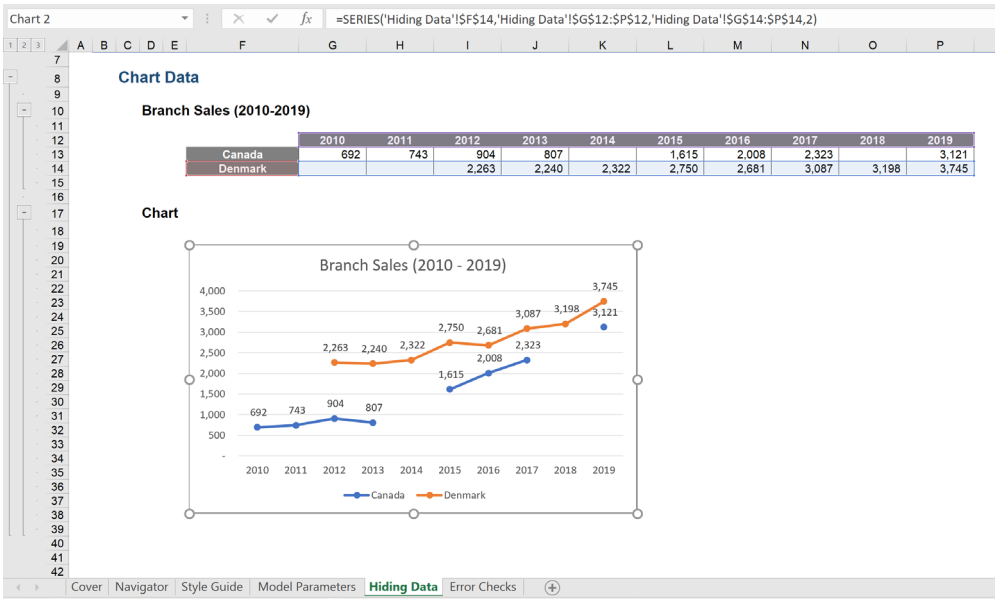
In the 'Hidden and Empty Cell Settings' pop-up dialog, make sure the 'Show data in hidden rows and columns' box is ticked:



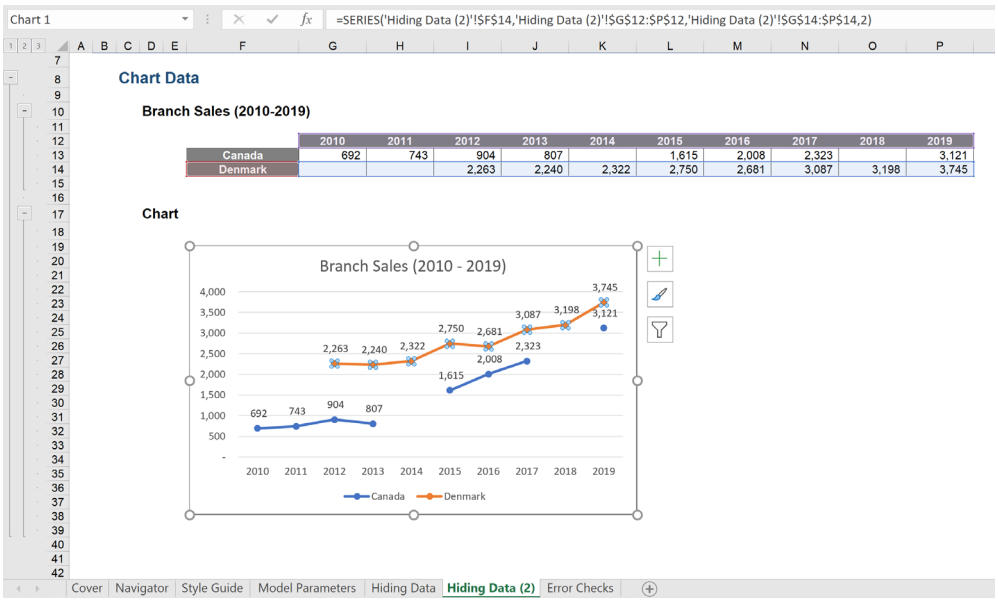
Now, if we hide the group rows, the chart is still displayed nicely:



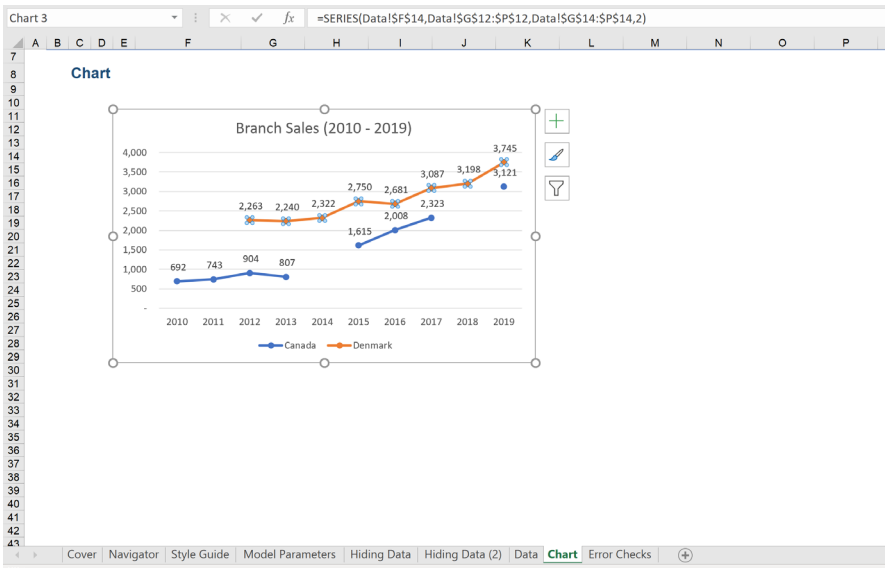
One more tip that I'd like to mention here is the reason why we should put our chart in the same sheet as its linked data. If we click on the chart series, we'll see where the data series is:



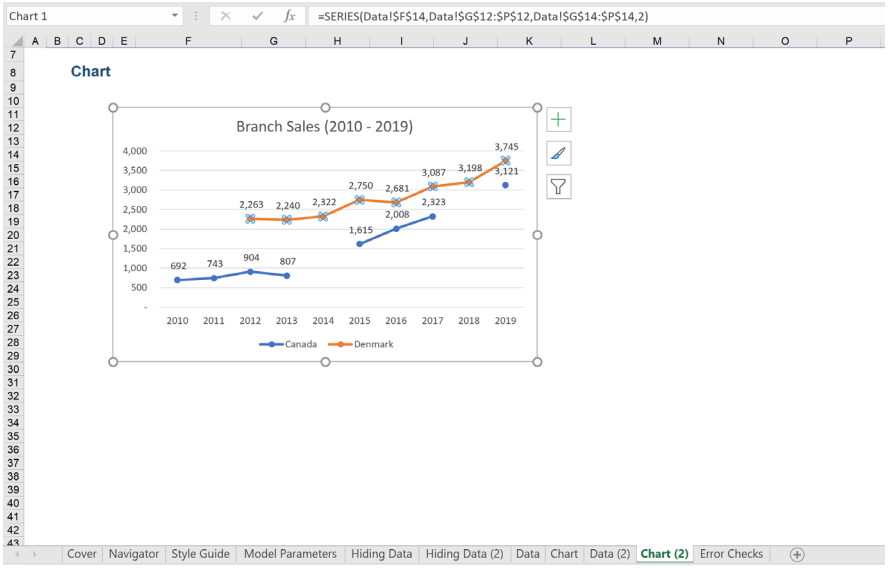
If we copy the whole worksheet, the chart in the new worksheet, '**Hiding Data (2)**', has its data series map to the data in the same sheet, which is clear:



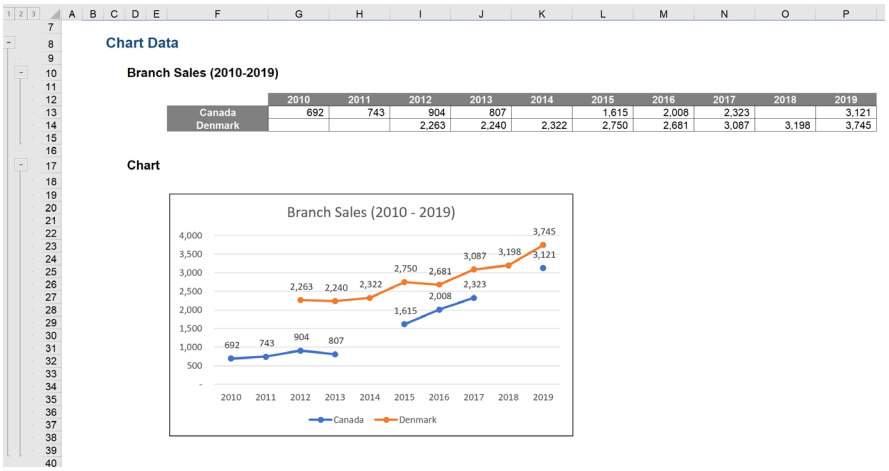
Now, we create a **Data** sheet with only the sales data, and create a chart if I create a **Data** sheet separate from the **Chart** sheet, the series will definitely point to the data on the **Data** sheet:



However, if we copy the **Data** sheet to **Data (2)** sheet, and **Chart** sheet to **Chart (2)** sheet, let's see what happens. The chart series in the copied **Chart (2)** sheet still links to the data in the original **Data** sheet (it should have linked to the copied **Data (2)** sheet instead):



That is the reason why it is a better idea for the chart to be placed in the same sheet as its linked data set, plus it may then be formatted to optimise the layout of the whole worksheet:



	2010	2011	2012	2013	2014	2015	2016	2017	2018	2019
Canada	692	743	904	807		1,615	2,008	2,323		3,121
Denmark			2,263	2,240	2,322	2,750	2,681	3,087	3,198	3,745

More 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 see what we can dig up with VBA…

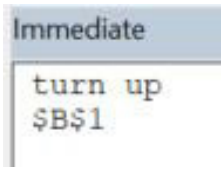
Last time, we ran into the **SearchOrder** parameter of the **Find** function. We knew that we could go right across the row or down the column. But what if we wanted to manipulate the direction of which we searched? Instead of going down the columns using **xlByColumns**, what if we wanted to go up? We can change that direction by using the **SearchDirection** parameter.

The **SearchDirection** parameter accepts the values of **xlNext** which means it's looking forwards (*i.e.* down columns and to the right of rows) or its opposite **xlPrevious**.

Let's see how it works in our amended subroutine from last month.

```
Sub FindAfterByGoingUpColumns()
    Dim searchRange As Range
    Set searchRange = Range("A1:E10")
    Dim foundrange As Range
    Set foundrange = searchRange.Find("up", After:=Range("C5"), SearchOrder:=xlByColumns,
SearchDirection:=xlPrevious)
    If foundrange Is Nothing Then
        Debug.Print "not found!"
    Else
        Debug.Print foundrange
        Debug.Print foundrange.Address
    End If
End Sub
```

This results in the following:



Changing the **SearchOrder** to **xlByRows** will force the **Find** method to go left across the rows first before going up. It's simple to go after what you are looking for in any direction.

Next newsletter, we're going to be experimenting with more parameters of the **Find** function.

Until then.

Power Pivot Principles

*We continue our series on the Excel COM add-in, Power Pivot. This month, we discuss the **CALCULATETABLE** function.*

The **CALCULATETABLE** function is somewhat similar to the **CALCULATE** function in Power Pivot, in that it applies filters to the data returned with a calculated value. The main difference between the two is that the **CALCULATETABLE** function returns with a table whereas the **CALCULATE** function returns with a scalar value.

The **CALCULATETABLE** requires the following syntax to operate:

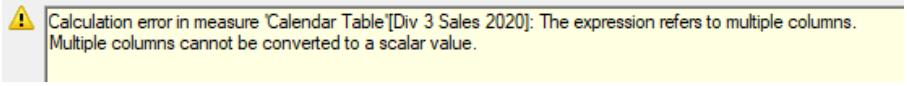
CALCULATETABLE(**expression**, **filter1**, **filter2**, …)

The **expression** is the table to be evaluated, and the **filter** is a column that can be located in any table imported to Power Pivot. In order for the filters to work they have to be from tables that have proper connections established in the data model.

For example, let's create a measure that calculates the sales for Division 3 in the year 2020:

```
=CALCULATETABLE(
        'Sales Table','Calendar Table'[Year]=2020
        )
```

Using the **CALCULATETABLE** formula alone will not work, as Power Pivot will return with the error:



As it stands above, the current **CALCULATETABLE** measure returns with a table looking like this:

	A	B	C	D	E	F
1						
2		**Row Labels**	**Sales**	**Sum of Division 1**	**Sum of Division 2**	**Sum of Divisi**
3		2020	$21,489,980	$18,919,908	$783,733	$1,786
4		**Grand Total**	**$21,489,980**	**$18,919,908**	**$783,733**	**$1,786**

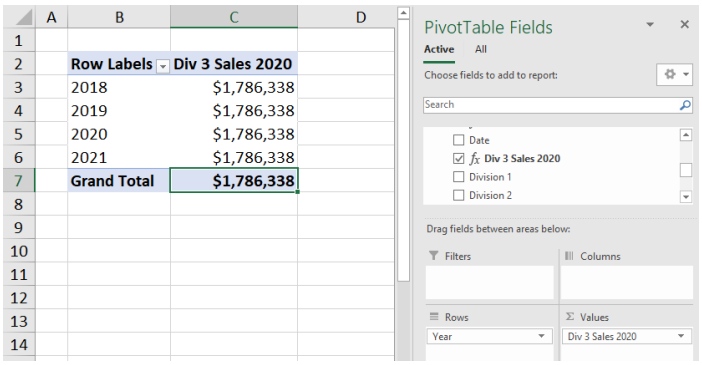
Power Pivot can't aggregate an entire table into a single cell. We have to specify which expression we want to be evaluated. We should use the **SUMX** function. The **SUMX** function "returns the sum of an expression evaluated for each row in a table" and requires the following syntax to operate:

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

With that in mind, we can adjust our previous measure:

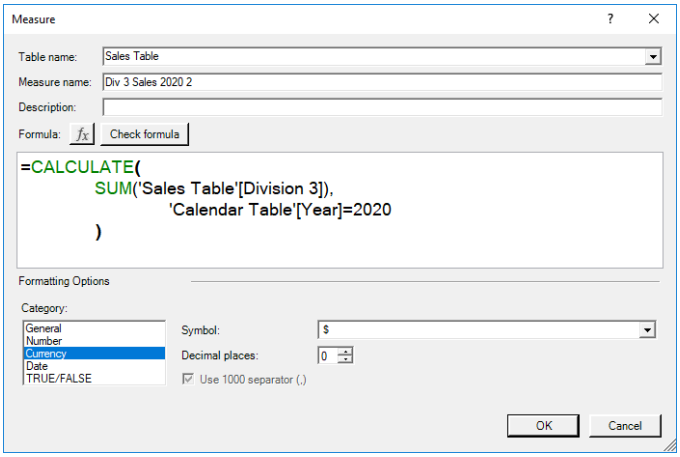
```
=SUMX(
        CALCULATETABLE('Sales Table','Calendar Table'[Year]=2020)
                , [Division 3]
        )
```

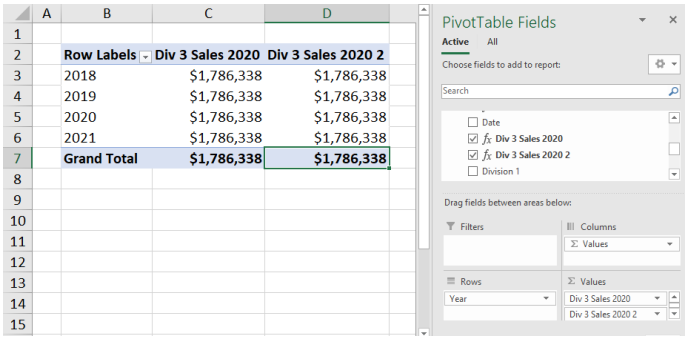
The resulting PivotTable is as follows:



	A	B	C	D
1				
2		**Row Labels**	**Div 3 Sales 2020**	
3		2018	$1,786,338	
4		2019	$1,786,338	
5		2020	$1,786,338	
6		2021	$1,786,338	
7		**Grand Total**	**$1,786,338**	
8				
9				
10				
11				
12				
13				
14				

Of course, this result may be achieved using the **CALCULATE** function:





	A	B	C	D
1				
2		**Row Labels**	**Div 3 Sales 2020**	**Div 3 Sales 2020 2**
3		2018	$1,786,338	$1,786,338
4		2019	$1,786,338	$1,786,338
5		2020	$1,786,338	$1,786,338
6		2021	$1,786,338	$1,786,338
7		**Grand Total**	**$1,786,338**	**$1,786,338**
8				
9				
10				
11				
12				
13				
14				
15				

As it stands, the **CALCULATETABLE** and the **CALCULATE** functions are quite similar. The key difference is their outputs. We would use the **CALCULATETABLE** function when we need to use other functions that expect a table as an expression and **CALCULATE** when we use functions that expect a single value.

More *Power Pivot Principles* next month.

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 look at some useful Boolean **Text** functions in **M**.*

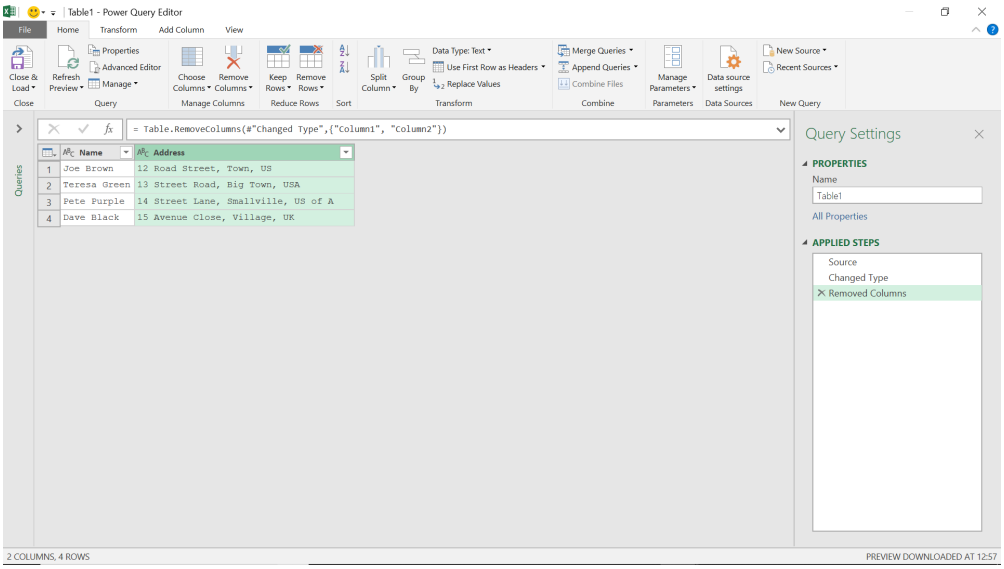
Cleaning data often involves transforming text strings to ensure they contain consistently formatted information. There are a few **Text()** functions in **M** which are particularly useful. For this newsletter, we'll begin with some true / false functions, and we'll provide an example for each one.

Text.Contains

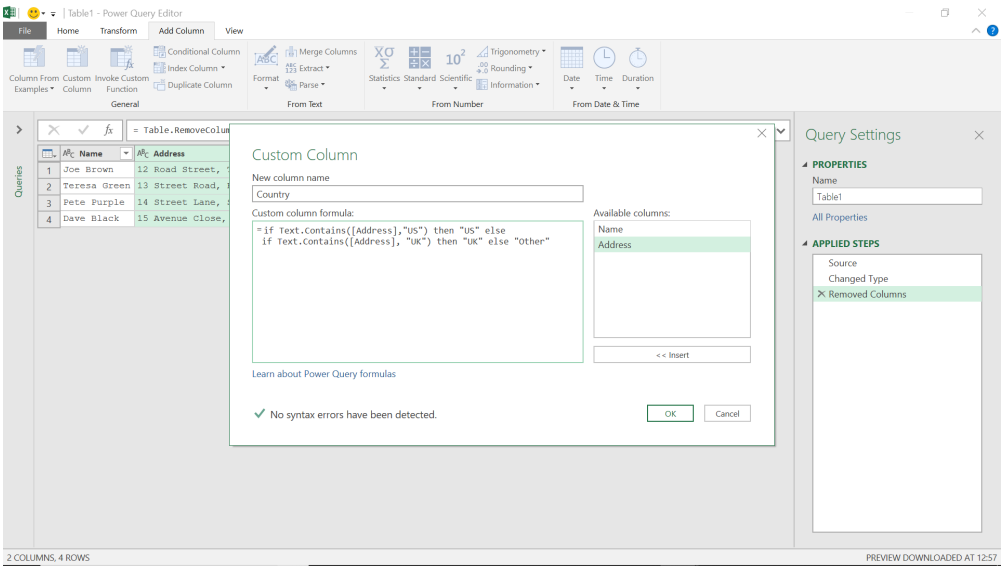
Text.Contains(string as nullable text**, substring** as text**, optional comparer** as nullable function**)** as nullable logical

This returns true if a text value **substring** was found within a text value **string**; otherwise, false.

This function is either true or false, so if a particular word or series of letters exists in a column, the function will be true, otherwise it will be false. This function can be useful in conditional statements where you need to decide how to populate a new column. Here is a simple example *(below)*, where we wish to create a column containing the country:



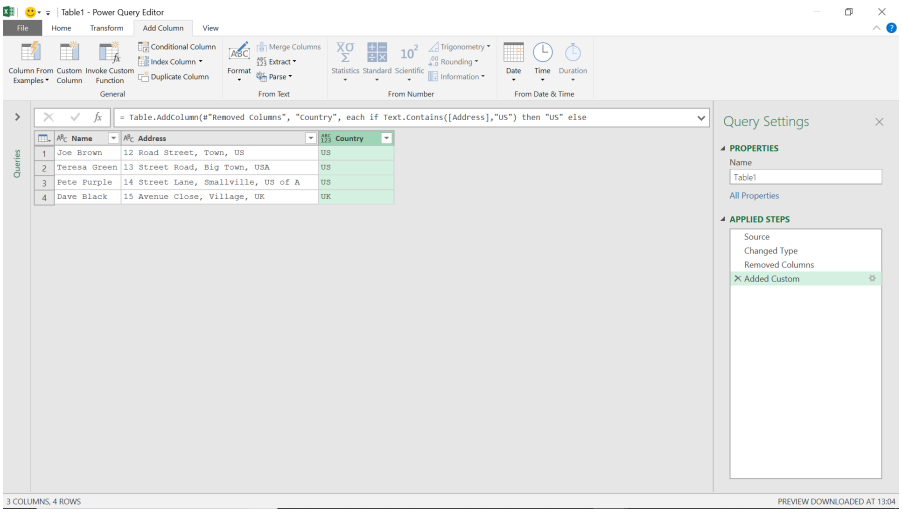
Let's choose to create a new custom column so that we have complete control over the formula that populates our column. We'll choose 'Custom Column' from the 'Add Column' tab (which we will use for all of the following examples).



The formula used is:

= if Text.Contains([Address],"US") then "US" else if Text.Contains([Address], "UK") then "UK" else "Other"

The resulting column extracts the country to make our data more useful and easier to combine with other tables.



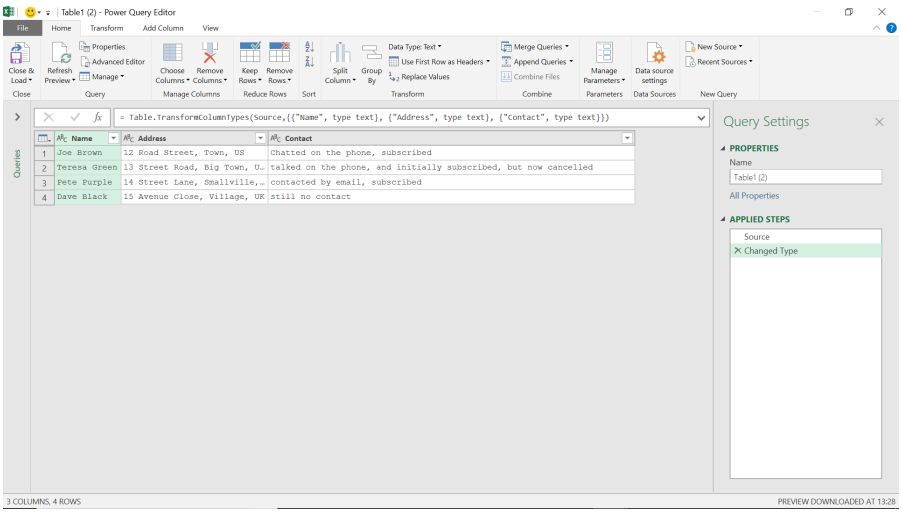
Text.EndsWith

This is also a Boolean (true or false) function.

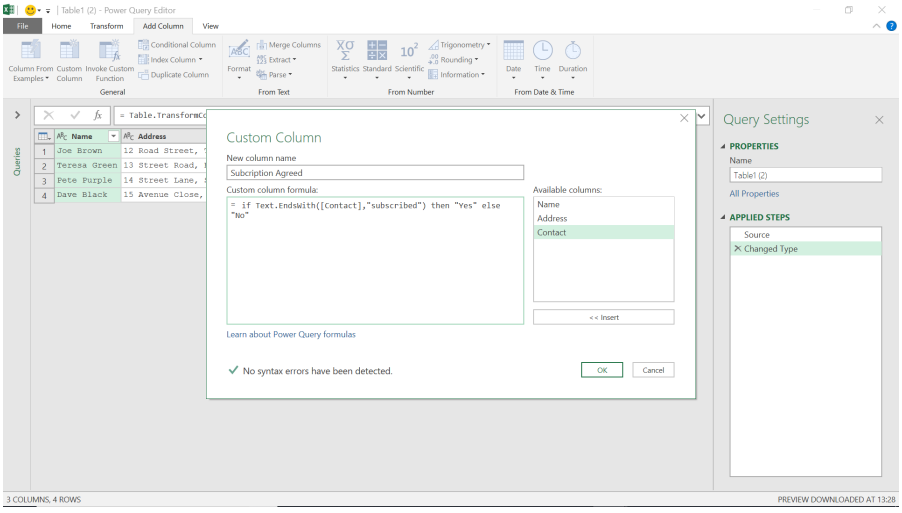
Text.EndsWith(string as nullable text**, substring** as text**, optional comparer** as nullable function**)** as nullable logical

This returns a logical value indicating whether a text value **substring** was found at the end of a **string**.

We have added a new column to our data and we wish to know which client subscribed to our marketing communication. Note that we can't just look for the substring 'subscribed' as one of our targets changed their mind!

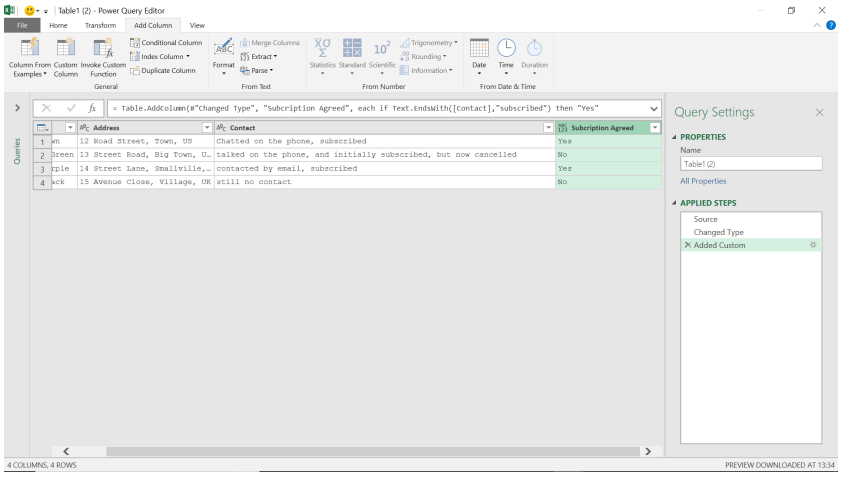


We add a new custom column from the 'Add Column' tab.



This time, the formula is:

= if Text.EndsWith([Contact], "subscribed") then "Yes" else "No"



The two clients who are currently subscribing are identified by the new column.

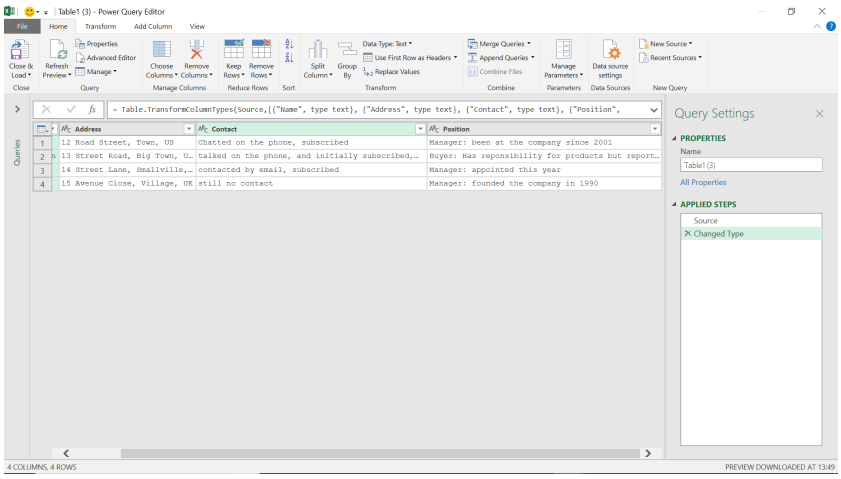
Text.StartsWith

This is clearly related to the previous function, but this time, we're looking for a substring at the beginning of the column.

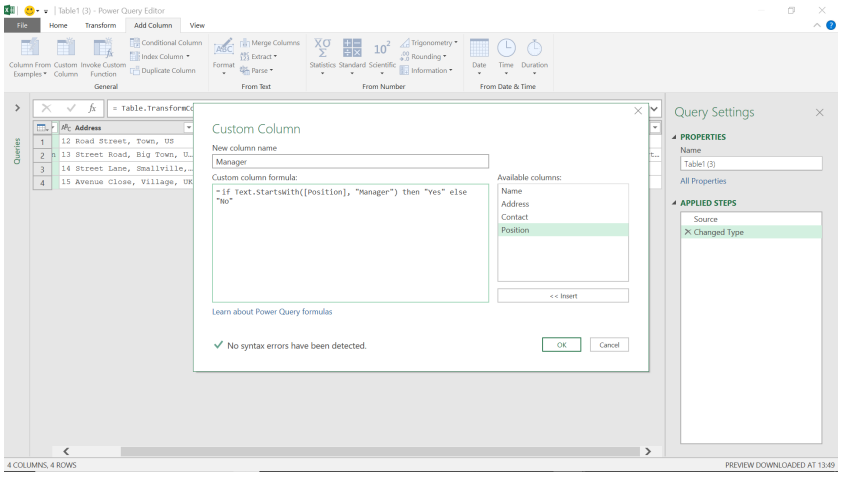
Text.StartsWith(string as nullable text**, substring** as text**, optional comparer** as nullable function**)** as nullable logical

This returns a logical value indicating whether a text value **substring** was found at the beginning of a **string**.

On the query below, we want a column to indicate whether the contact is the manager at their company.

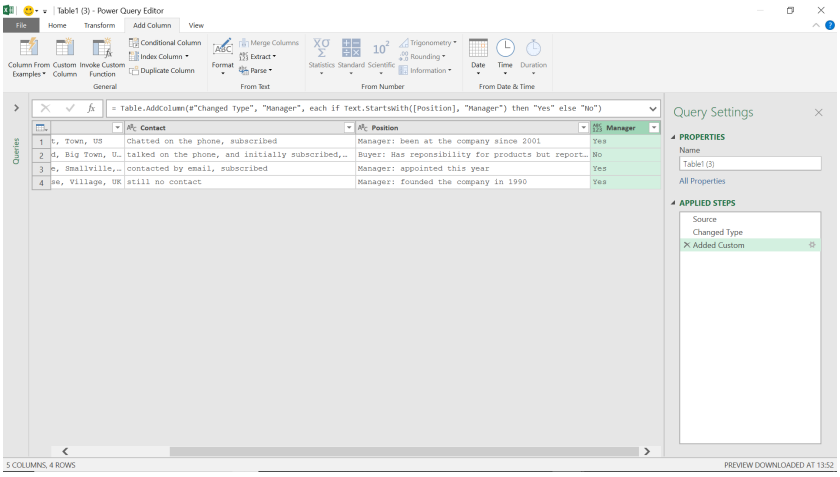


Again, we'll create a custom column from the 'Add Column' tab.



The formula we have used here is:

= if Text.StartsWith([Position], "Manager") then "Yes" else "No"



We can now see which managers have been contacted.

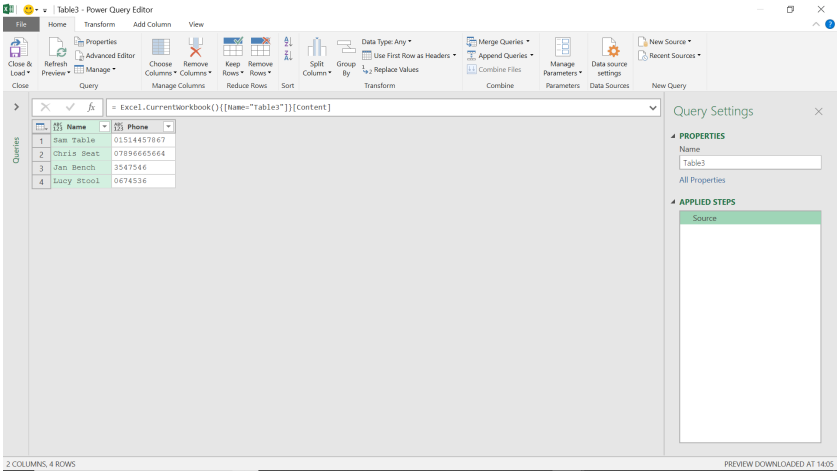
Text.Length

This is not actually a Boolean function, since it returns a length, but it is often used within a Boolean statement, as shown in the following example.

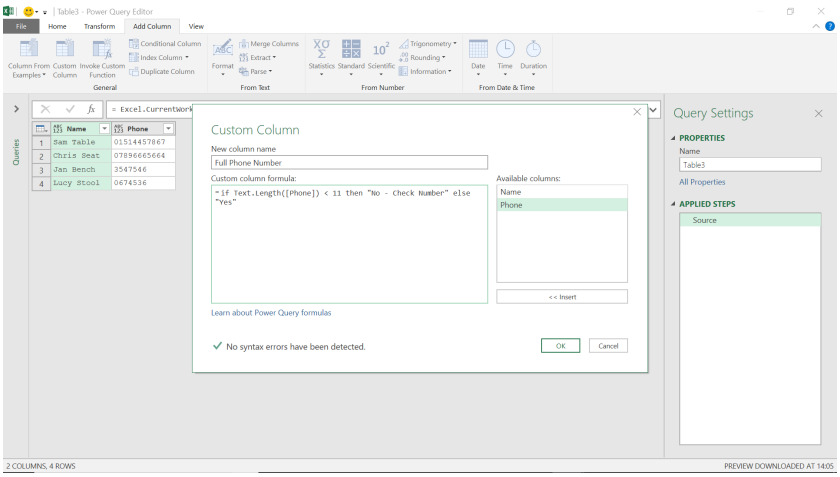
Text.Length(text as nullable text**)** as nullable number

This returns the number of characters in a text value.

On the next screen, we have some (UK) phone numbers, but not all of them include the location code(s). The easiest way to tell whether the full number has not been given is to look at the length of the number:

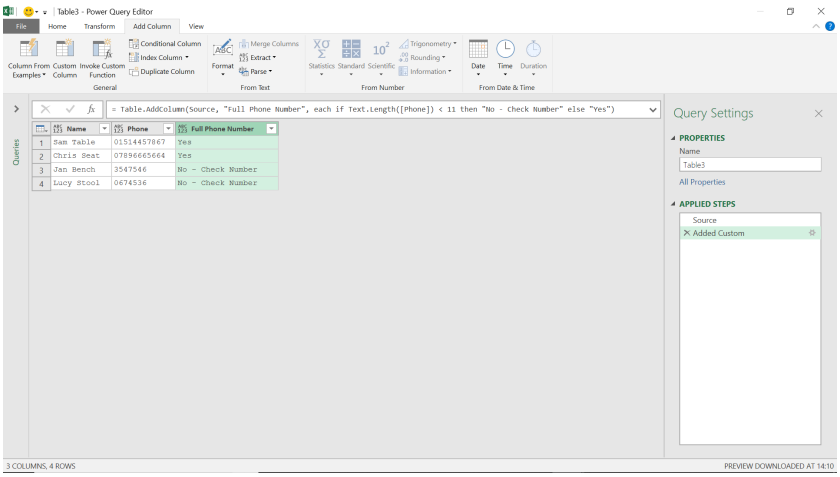


Let's create a custom column from the 'Add Column' tab:



The formula used here is:

= if Text.Length([Phone]) < 11 then "No - Check Number" else "Yes"



We can now see the numbers that may need further investigation.

Next month, we'll look at some useful **M Text()** functions that can be used to edit the text in a column.

Power BI Updates

Due to Microsoft running late and our own publishing deadlines, we had no updates to bring to you for our 10-year anniversary newsletter. We put that right now. This month, the updates see the introduction of an Optimize Ribbon which makes your report creation experience more streamlined, especially if you are working in DirectQuery mode. Also new this time around are Linked Metrics and a new **EvaluateAndLog** DAX function that enable you to log intermediate values of your DAX expressions.

The full list is as follows:

Change of Image

- Announcing New Power BI Colour Accent

Reporting

- Unshared and unsynchronised axes for small multiples charts
- Controlling and customising labels on Azure Maps
- Create dynamic slicers using field parameters in Preview
- Composite models over Power BI datasets and Analysis Services in Preview

Modelling

- Streamline your report authoring experience with the Optimize Ribbon
- New DAX functions: **EVALUATEANDLOG**, **TOCSV** and **TOJSON**

Data connectivity and preparation

- Azure Databricks, Databricks
- Dremio Cloud
- Kognitwin
- Spigit, Projectplace, Planview Enterprise One – PRM, Planview Enterprise One – CTM
- SumTotal

Service

- New way to upload Power BI and Excel files
- Subscribe to a report with filters applied
- Linked metrics
- Information protection update

Paginated Reports

- Formatted Table authoring experience

Visualisations

- New visuals in AppSource
  accoPLANNING by Accobat
- ADWISE RoadMap / Gantt v2.5
- Control Chart XmR by Nova Silva
- Zebra BI Cards 1.3
- ZoomCharts Drill Down Graph PRO

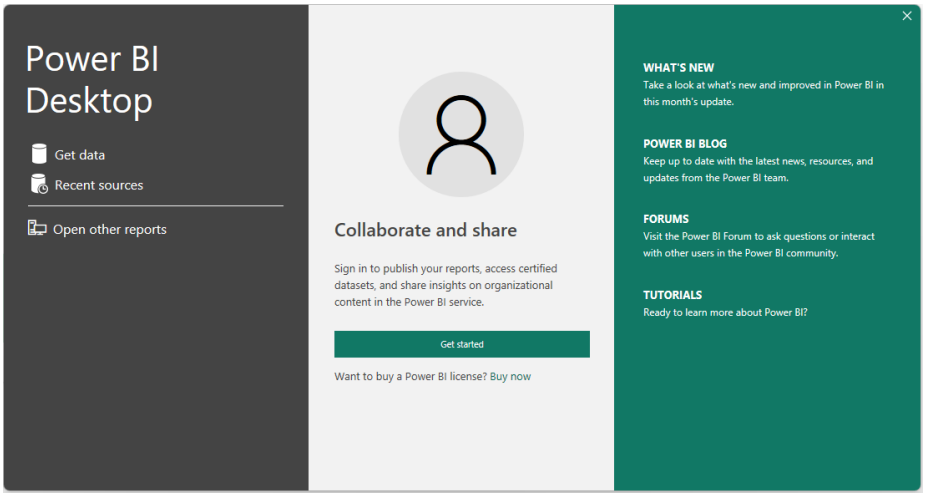
Other

Power BI Desktop infrastructure update (WebView2).

Let's look at each in turn.

Announcing New Power BI Colour Accent

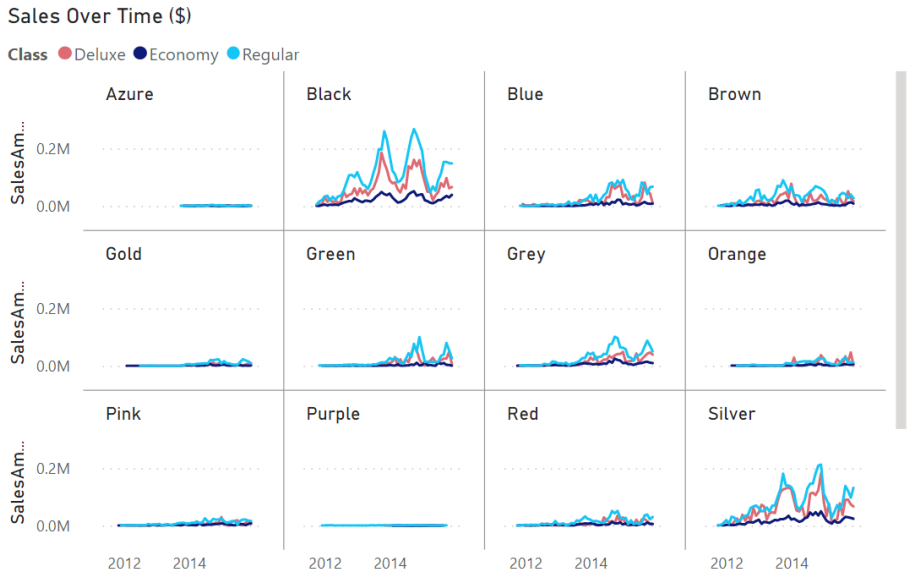
In this release, Power BI is updating its accent colour to teal. This change was made to ensure Power BI is more accessible for users with disabilities. The new colour supposedly improves contrast and increases visibility of the user interface in Power BI (we really aren't so sure), making the experience easier to use and more inclusive.



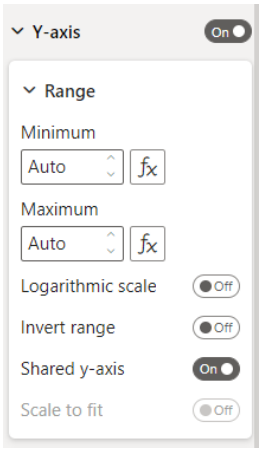
The Power BI brand colour and icon logo will remain yellow.

Unshared and unsynchronised axes for small multiples charts

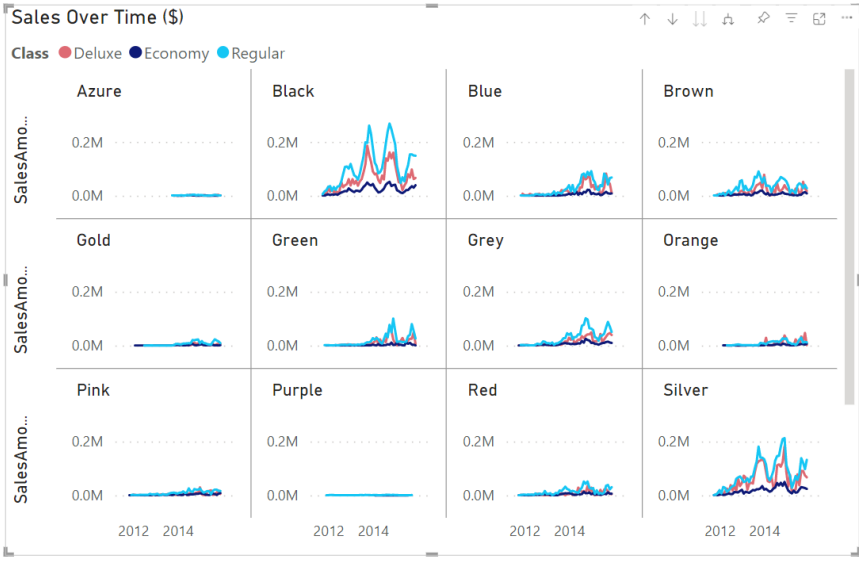
Sometimes, when reading a small multiples visual, a user might not be too concerned in comparing the absolute values of the numbers displayed against each other, instead only interested in comparing the trends of each category across time. However, when the ranges of the data for each small multiples category vary wildly, the charts with low maxima get pushed down compared to charts with high maxima. It becomes difficult to evaluate the sales trend for lower charts because it looks close to a horizontal line.



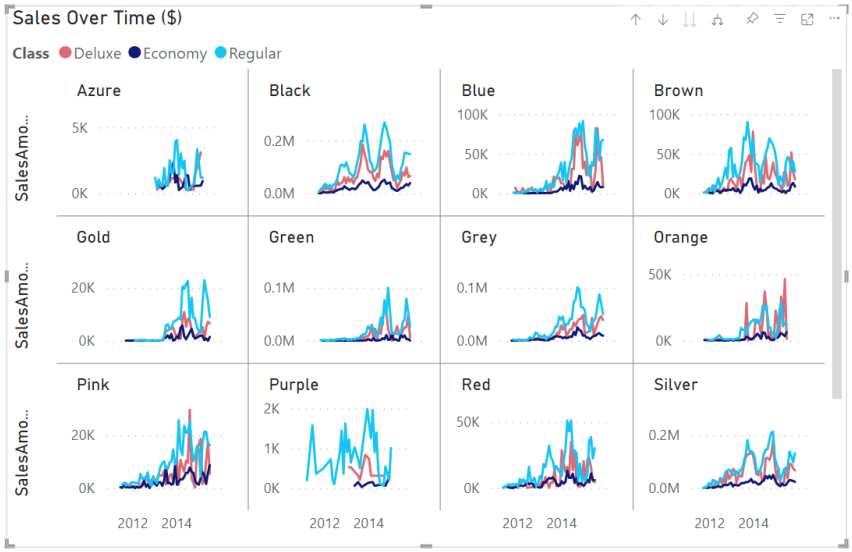
This update introduces a couple of new options to help you accommodate for those cases, allowing you to plot each small multiples chart against separate **y**-axes and then change the automatic scaling of each axis. You'll find the new options, 'Shared **y**-axis' and 'Scale to fit', in the **Y**-axis card of the Formatting pane.



Turning off the 'shared **y**-axis' option will give each of your small multiples charts their own **y**-axis:



Turning on the 'scale to fit' option will change their automatic scaling:



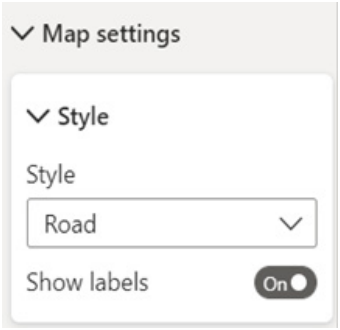
This new functionality will affect the auto minimum and maximum values. If the user has defined a minimum or maximum for the small multiples, the toggle won't override that boundary. Any secondary **y**-axes will follow the settings from the main **y**-axis.

Controlling and customising labels on Azure Maps

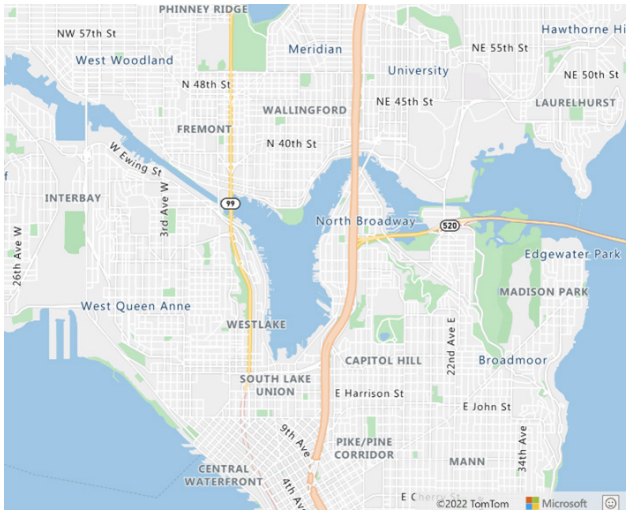
With this release, Power BI is introducing two key features to help you customise your Azure Maps visualisation to better suit your needs.

CONTROLLING DATA LABELS IN THE BASE MAP

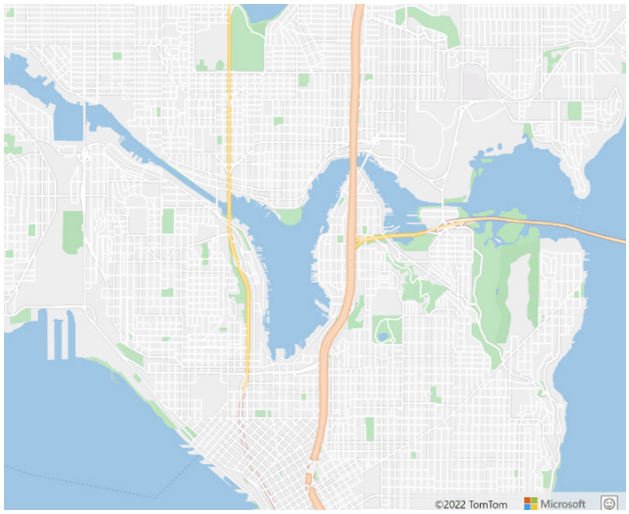
You can now decide if you want to show or hide labels on your map. You'll find the new 'Show labels' toggle in the 'Map settings' card of the Formatting pane.



If you toggle the labels off, the map will not show any labels on the base map to give you a clean map. This removes distractions for situations where the labels are not required to interpret the data shown. For example, here is what the map of part of Seatle looks like with labels on:



and here is the same map with the labels excluded:



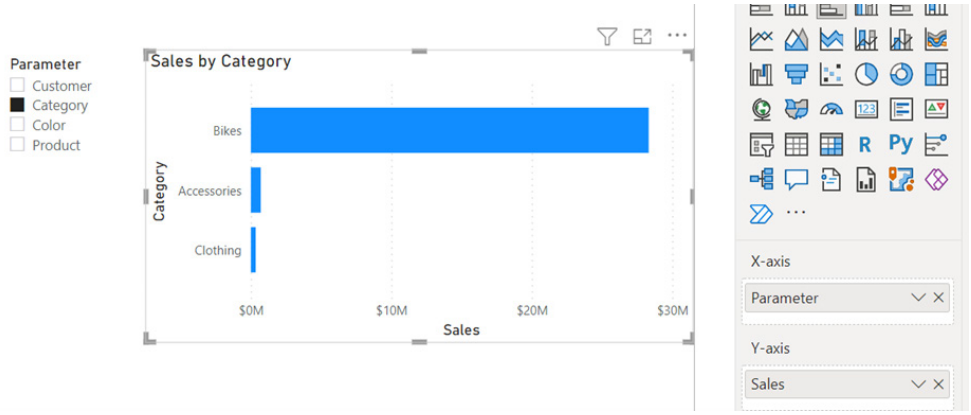
CUSTOMISABLE CATEGORY LABELS

In addition to turning off labels on the base map, you can now customise Category labels on your map as well. You'll find these options in the 'Category labels' card in the Format pane.

You can not only toggle category labels on or off, but you can also customise the font and colours used.

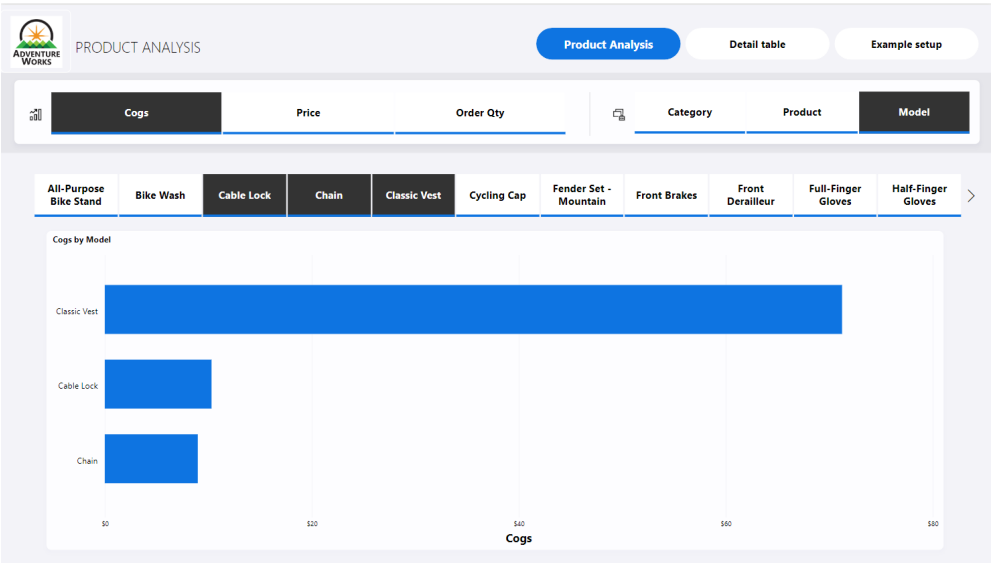
Create dynamic slicers using field parameters in Preview

Mid 2022, Microsoft released a public Preview of field parameters, which allowed end-users to dynamically change the measures and dimensions being analysed within a report. The initial release supported scenarios where Slicers or Filter cards could control which fields would be used in different visuals:

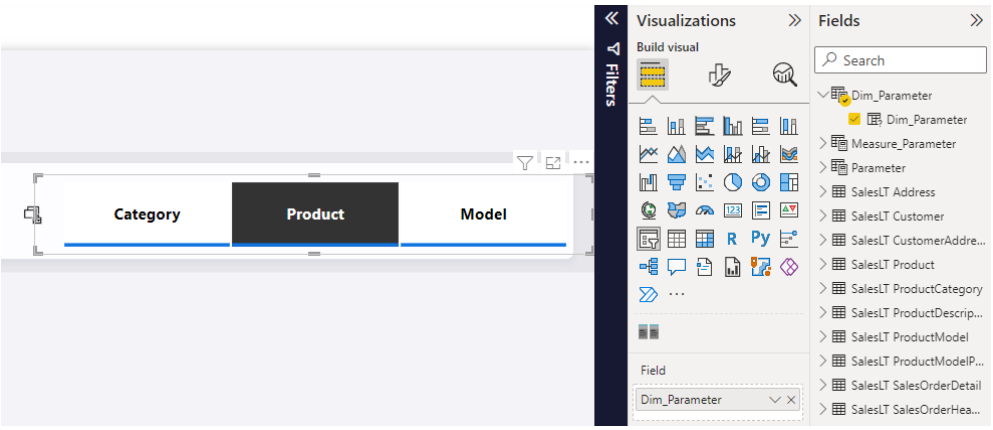


However, it did not support the ability for slicers to control the fields used in other slicers, also referred to as dynamic slicers. Microsoft has now removed this limitation, and you can now parameterise your slicers to support your dynamic filtering scenarios.

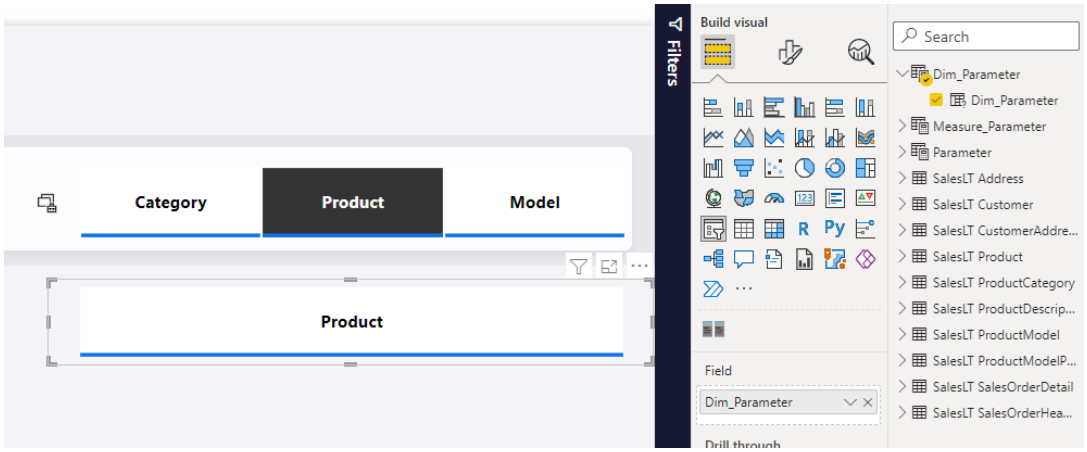
For example, in the report below we have three slicers: a measure picker, a dimension picker, and a dynamic slicer for filtering values. When you change the selected field for the dimension picker, not only will the main visual update but also the dynamic slicer will update as well to show the values of the selected dimension, which may be used for filtering.



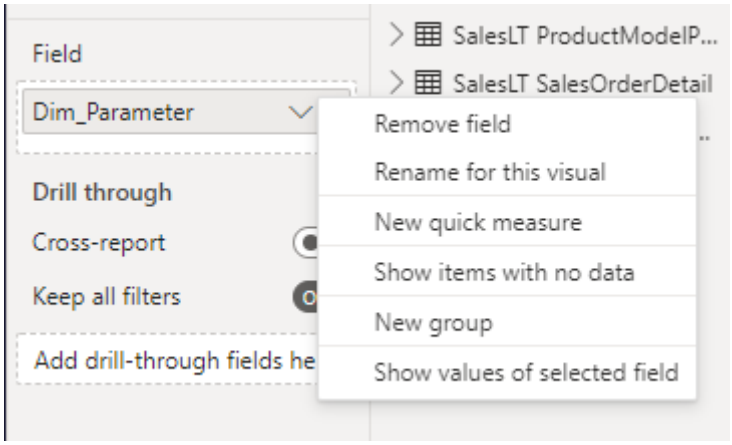
With this release, Microsoft has also enabled the field parameters Preview feature by default, so you do not have to manually turn on the Preview feature in the options menu. To create a dynamic slicer using field parameter, you will need to start by adding a field parameter to a slicer. This is the slicer that we will use as the field / dimension picker for the dynamic slicer:



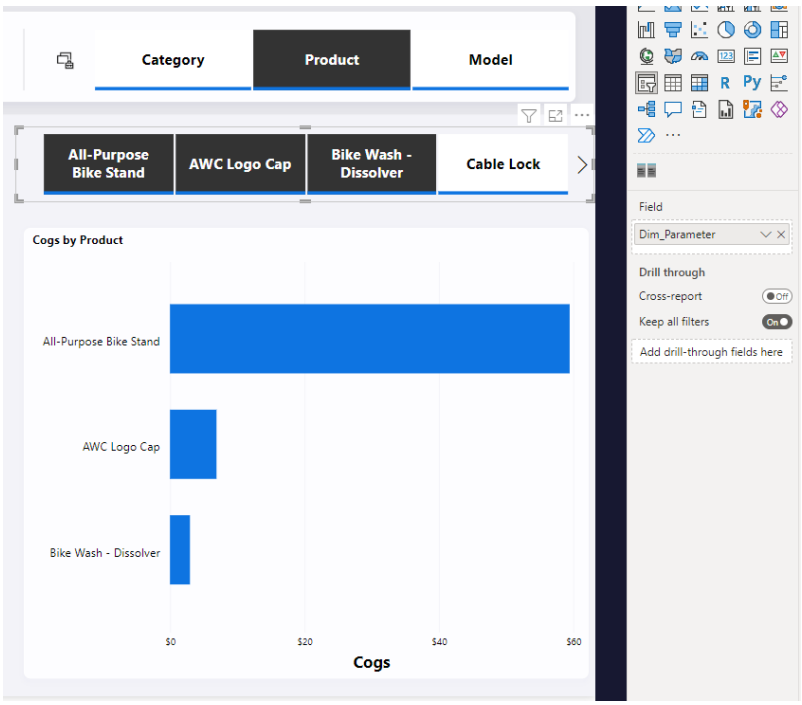
Next, you might want to duplicate the slicer using copy and paste. This slicer will become the dynamic slicer used for filtering values:



With the second slicer selected, you may launch the field input context menu (right-click menu) and select ‘Show values of the selected field’:



Now you have a dynamic slicer!



Composite models over Power BI datasets and Analysis Services in Preview

This update makes it possible for a single table to simultaneously filter more than one table in a remote source group. Previously this was not possible: the single table could filter more than one table in a remote source group but not at the same time. For example, a local Date table could filter either the remote Sales or the remote Inventory table in the same source group, but not at the same time. This restriction has now been lifted.

Streamline your report authoring experience with the Optimize Ribbon

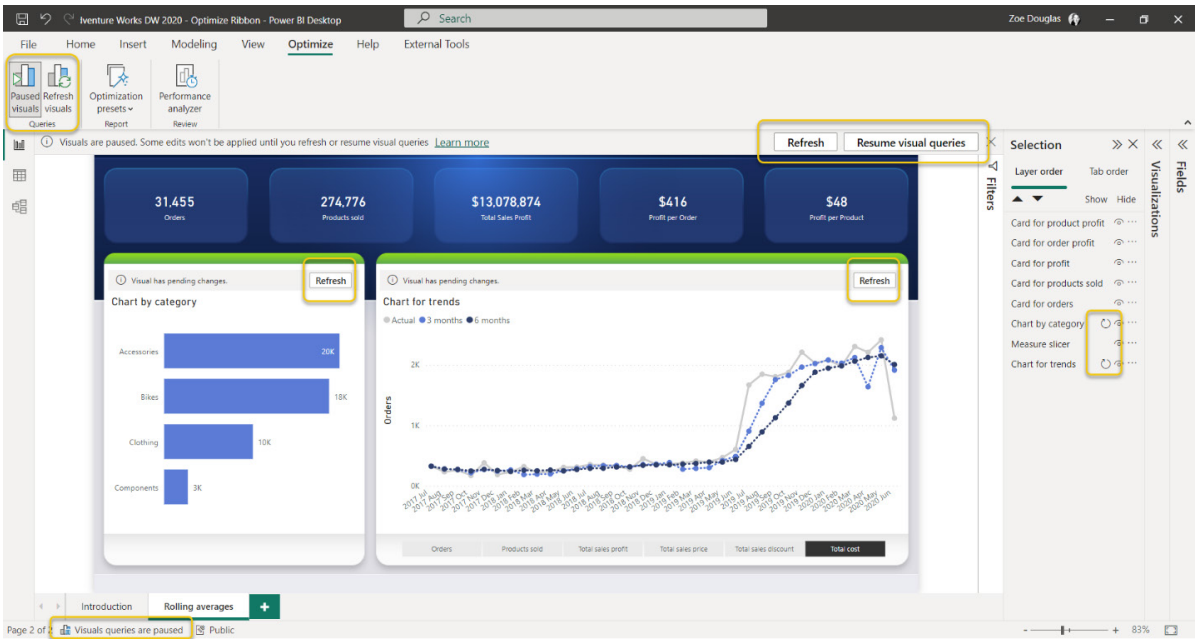
This update sees the public Preview of the Optimize Ribbon in Power BI Desktop. This is an entire Ribbon devoted to making an end user's experience authoring reports streamlined, especially in DirectQuery mode.

If you have ever wanted to stop visuals from loading data while you are still making changes, or wanted to edit multiple DAX measures without updating the report canvas in-between those edits, or wanted to take full control over when visuals refresh, then 'Pause visuals' is for you, for example.

'Pause visuals' provides you with control over when visuals send out DAX queries. Now you may make your edits undisturbed. Simply update your visual when you are ready. Moreover, the new Optimize Ribbon makes it easy to find the right report settings you need for DirectQuery reports. Now you can find and apply Optimization presets with only a few mouse clicks. The Optimize Ribbon is a significant step towards streamlining the DirectQuery report authoring experience.

In summary, the authoring tools on the Optimize Ribbon provide you with the following capabilities:

- **fully control when visuals refresh**. Switch to the Optimize Ribbon, click on 'Pause visuals', as in the below screenshot, and from now on visuals will no longer refresh automatically. Click on 'Refresh visuals' when you are ready for visuals to run their queries and update their data
- **quickly choose and apply predefined combinations of settings tailored to your reporting needs**. No longer are you limited to default settings in Power BI Desktop, as you may quickly choose and apply settings tailored to your reporting needs with Optimization presets. Click the drop down on 'Optimize presets' to choose between full interactivity, full query reduction or customise to find the perfect balance for your report
- **conveniently launch 'Performance Analyzer' to analyse the queries your report visuals generate**. Easy access to important performance optimisation tools goes a long way to boost report author productivity. The Optimize Ribbon is now your one-stop shop to the tools to create optimized experiences for your report users.



The above screenshot shows you the Optimize Ribbon in action. It is now available in Power BI Desktop. Just keep in mind that you need to enable the Optimize Ribbon explicitly. As a Preview feature, go to **Files -> Options and Settings**, then click on **Options**, select **Preview features**, and then select the 'Optimize Ribbon' checkbox.

New DAX functions: EVALUATEANDLOG, TOCSV and TOJSON

EVALUATEANDLOG

If you have ever wanted to see the intermediate results of a **DAX** expression or to debug your **DAX** by printing out values, then consider the new **DAX** function **EVALUATEANDLOG**. It takes any **DAX** expression, evaluates it and returns the result. That's not all though: it will also log the result (hence the name **EVALUATEANDLOG**) to the **DAX** evaluation log that you may read out using tools like SQL Profiler. It's very much like a "print debugging statement".

For example, consider the following measure that calculates the sales growth year over year, using a **Sales Amount** measure and the **SAMEPERIODLASTYEAR** function:

YoYGrowth := [Sales Amount] – CALCULATE([Sales Amount], SAMEPERIODLASTYEAR('Date'[Date]))

To make sure this measure does what you need it to do you can now wrap the parts of the measure in **EVALUATEANDLOG** to see the intermediate result, as in the example below. All the while, **Profit** will still return the same result as before:

YoYGrowth := EVALUATEANDLOG([Sales Amount]) – EVALUATEANDLOG(CALCULATE([Sales Amount], SAMEPERIODLASTYEAR('Date'[Date])))

However, additionally, the **DAX** evaluation log will now contain the result of **Sales Amount** and **CALCULATE([Sales Amount], SAMEPERIODLASTYEAR('Date'[Date]))**.

You can see the output using SQL Profiler by connecting to Desktop, starting a trace and subscribing to the 'DAX Evaluation Log' event. Alternatively, you can use the open source DAXDebugOutput tool. Please note that the DAXDebugOutput tool is not an official Microsoft tool and hence is neither signed nor supported.

TOCSV and TOJSON

These two very similar **DAX** functions are closely related to **EVALUATEANDLOG** but may also be separately. These functions convert the input Table to CSV or JSON, respectively. For example:

MyCSV = TOCSV(DimProduct)

returns something like this (screenshot truncated):

'DimProduct'[ProductKey],'DimProduct'[ProductAlternateKey],'DimProduct'[ProductSubcategoryKey],'DimProduct'[WeightUnitMeasureCode],'DimProduct'[SizeUnitMeasureCode],'DimProduct'[EnglishProductName],'DimProduct'[SpanishProductName],'DimProduct'[FrenchProductName],'DimProduct'[StandardCost],'DimProduct'[FinishedGoodsFlag],'DimProduct'[Color],'DimProduct'[SafetyStockLevel],'DimProduct'[ReorderPoint],'DimProduct'[ListPrice],'DimProduct'[Size],'DimProduct'[SizeRange],'DimProduct'[Weight],'DimProduct'[DaysToManufacture],'DimProduct'[ProductLine],'DimProduct'[DealerPrice],'DimProduct'[Class],'DimProduct'[Style],'DimProduct'[ModelName],'DimProduct'[EnglishDescription],'DimProduct'[FrenchDescription],'DimProduct'[ChineseDescription],'DimProduct'[ArabicDescription],'DimProduct'[HebrewDescription],'DimProduct'[ThaiDescription],'DimProduct'[GermanDescription],'DimProduct'[JapaneseDescription],'DimProduct'[TurkishDescription],'DimProduct'[StartDate],'DimProduct'[EndDate],'DimProduct'[Status]
520,SE-R995,15,,,HL Road Seat/Saddle,Sillín/asiento de carretera GA,Selle de vélo de route HL,23.3722,TRUE,NA,500,375,52.64,,NA,,1,R,31.584,H,,HL Road Seat/Saddle 2,Lightweight kevlar racing saddle. Leather.,Selle de course légère en Kevlar. Cuir.,含轻型凯夫拉纤维的比赛用车座。皮革材质。,مقعد سباق خفيف الوزن.

You may change the delimiter (applicable to **TOCSV** only) and whether you want the headers to be included. Furthermore, you may also specify the maximum number of rows to be returned too.

Azure Databricks, Databricks

The Databricks connectors now support gracefully cancelling queries when consecutive queries are sent.

Dremio Cloud

The Dremio Cloud connector has been updated to now provide customers with the ability to manually enter in their server name to provide more flexibility on connecting to different servers.

Kognitwin

The Kognitwin connector has been updated with the following changes:

- add support for time series to enable monitoring of continuous sensor data
- resolve external data flag in API
- resolve missing data rows bug.

Spigit, Projectplace, Planview Enterprise One – PRM, Planview Enterprise One – CTM

The Planview connectors have been updated with new names:

- **Spigit** is now **Planview Ideaplace**
- **Projectplace** is now **Planview Projectplace**
- **Planview Enterprise One – PRM** is now **Planview Portfolios**
- **Planview Enterprise One – CTM** is now **Planview Enterprise Architecture**.

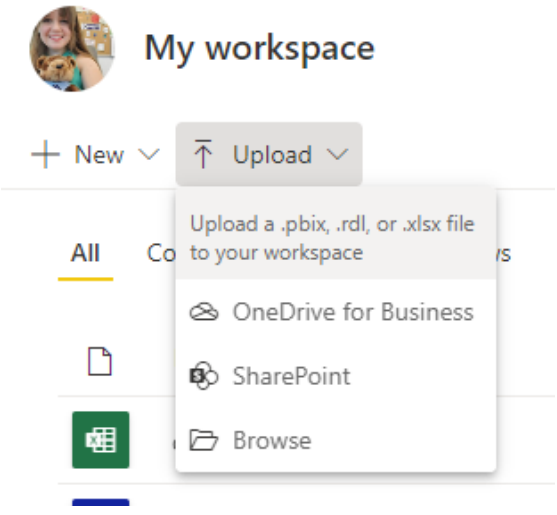
SumTotal

The SumTotal connector has been updated with the following changes:

- added support for filtering entity results by rowVersionId and rowVersionDecimal
- added support for filtering entity results by isActive, IsDeleted flags.

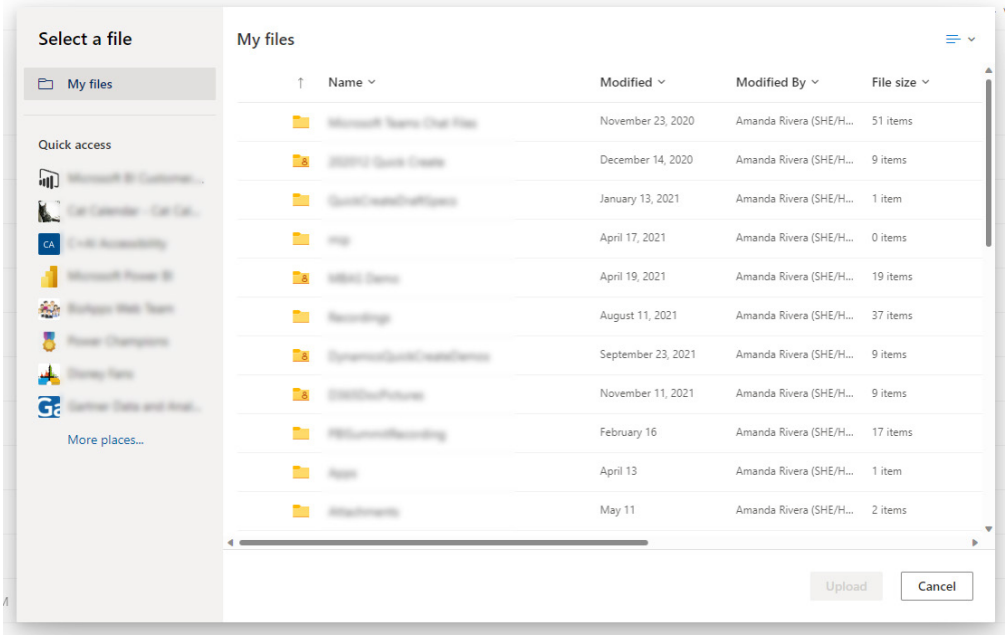
New way to upload Power BI and Excel files

Microsoft has now introduced a new, streamlined experience for uploading files to the Power BI Service. In the workspace you want to add files to, you'll see an Upload option next to the New button. You'll be able to use this dropdown to connect to files stored in OneDrive for Business or any SharePoint site you have access to or upload them from your computer through the Browse option.

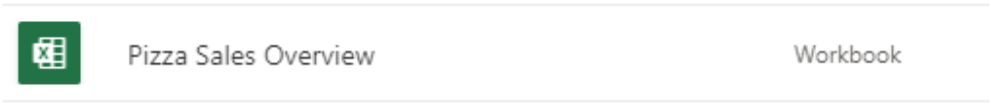


If you choose to upload a local file, a copy of that file is added to the workspace. If you use the OneDrive or SharePoint option, Power BI creates a connection to the file, and as you make changes to the file in SharePoint, Power BI can automatically synchronise those changes approximately every hour.

One of the benefits of this new way to upload files, in addition to being centrally located, is that the OneDrive and SharePoint options use the same file picker you're used to seeing in other Microsoft products.

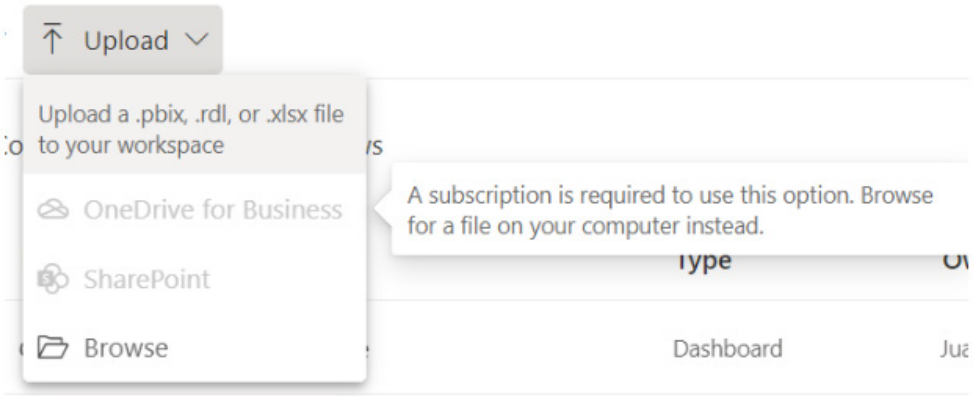


Instead of having to paste a direct URL to a given SharePoint site, you can now just select one of your sites through the 'Quick access' section or the 'More places' link. When uploading an Excel file through this experience, your workbook appears in Power BI just like it would in Excel Online:



For now, if you want to actually import the data from the Excel file into a Power BI dataset, you can continue to use the older 'Get Data' flow. Very shortly (if not already completed by the time this is published), Microsoft will also introduce a new way to create datasets from CSV and Excel files.

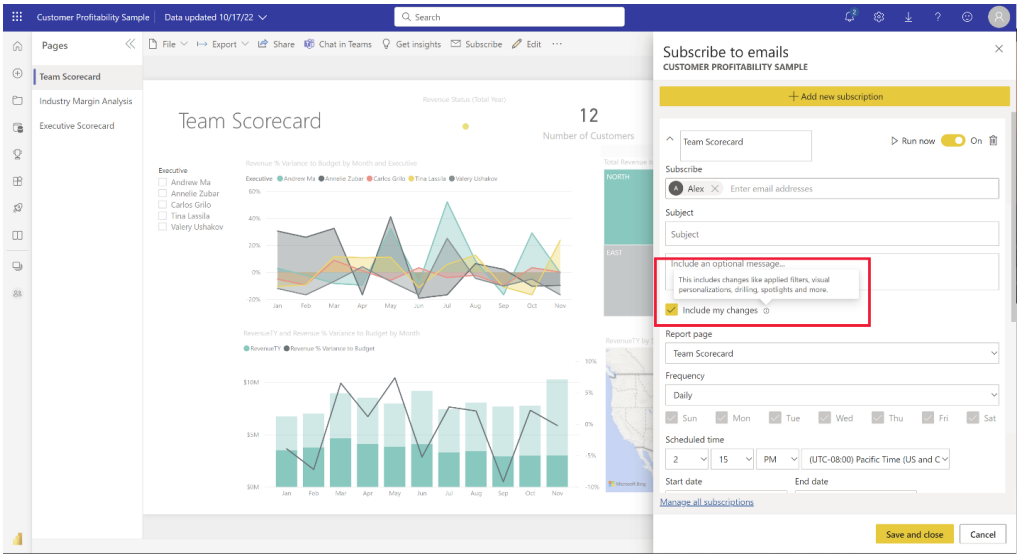
The OneDrive for Business and SharePoint options will be greyed out if you don't have a subscription for these products. Of course, you'll still be able to browse for local files on your computer.



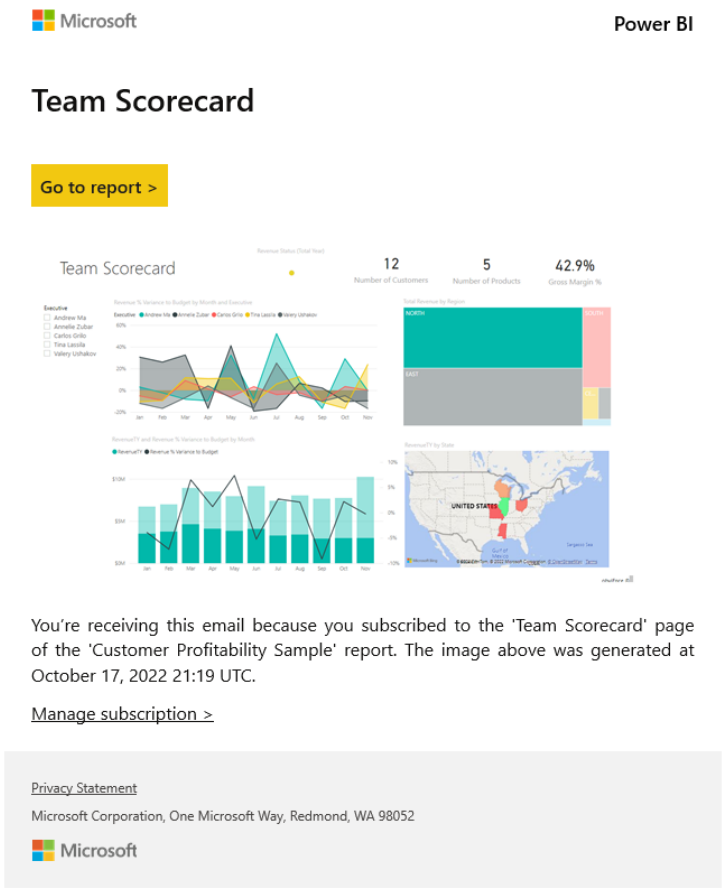
It should also be noted that presently Power BI does not have an option to upload files from personal OneDrive accounts. For now, you may continue using the older 'Get Data' flow to access files in your personal account. However, very shortly, Microsoft does plan on removing the older 'Get Data' experience along with the option to upload files from personal OneDrive accounts. At this point, you'll still be able to upload local files if you do not have access to a business OneDrive account.

Subscribe to a report with filters applied

You can now create subscriptions to a view of a Power BI report uniquely relevant to you. When creating a new subscription, just select the 'Include my changes' option to subscribe to the view of the report as you currently see it, including any changes you have applied. By selecting the 'Include my changes' option, the view of the report sent to your email will include any of the following changes: filters, slicers, personalize visuals, cross-filtering or cross-highlighting, drill down or drill up and spotlight.



When selecting 'Go to report' from the email subscription body, you will be navigated to the view of the report in the Power BI Service with the same set of changes (*e.g.* filters, slicers) that were applied when you created the subscription. This new 'Include my changes' option is available to any user who can create email subscriptions.

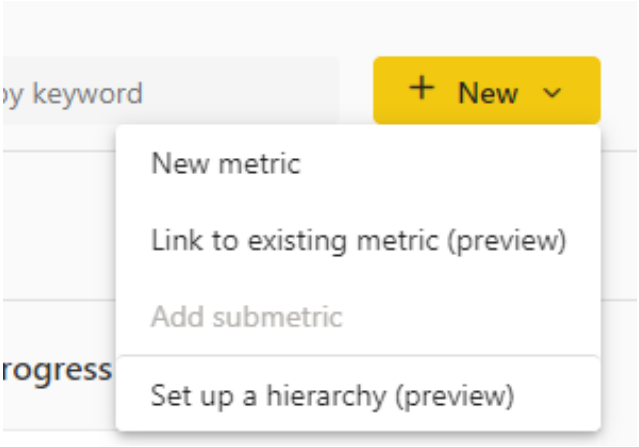


Linked metrics

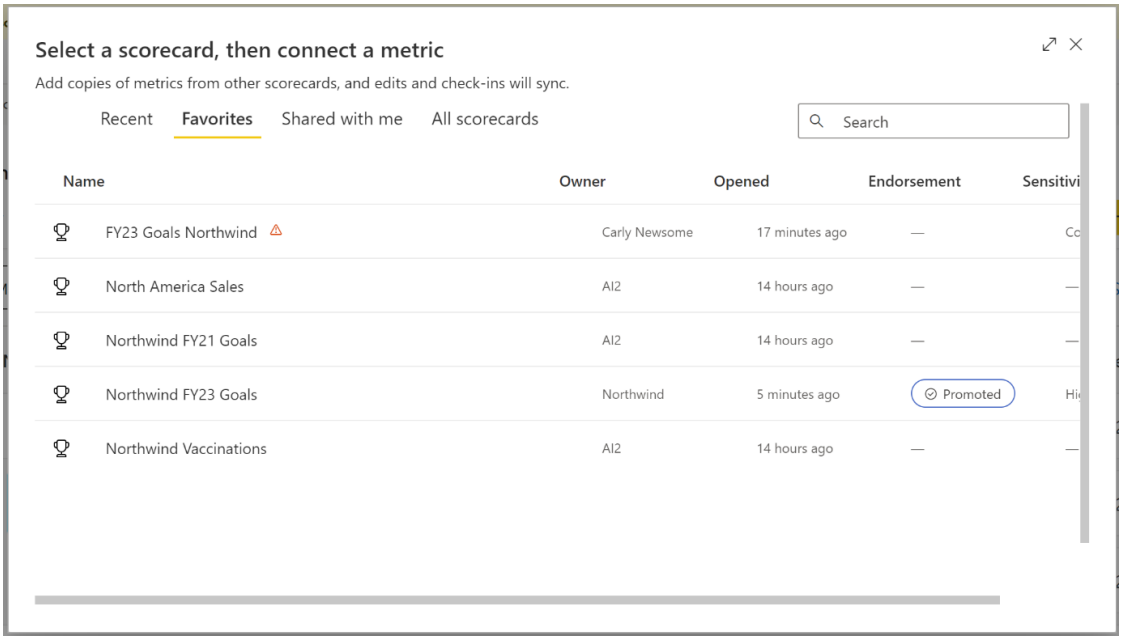
With the release of linked metrics, you now can show the same metric on multiple scorecards, across multiple workspaces. All check-ins, edits and updates will be reflected in all the metric locations, making it easier to not create duplicate metrics tracking the same thing.

For example, in many organisations, the leadership team has a scorecard, and each department has its own scorecard with metrics from the former along with others that are relevant to the specific department. Using this feature, you can now link such metrics to any number of scorecards and get them to be in sync automatically.

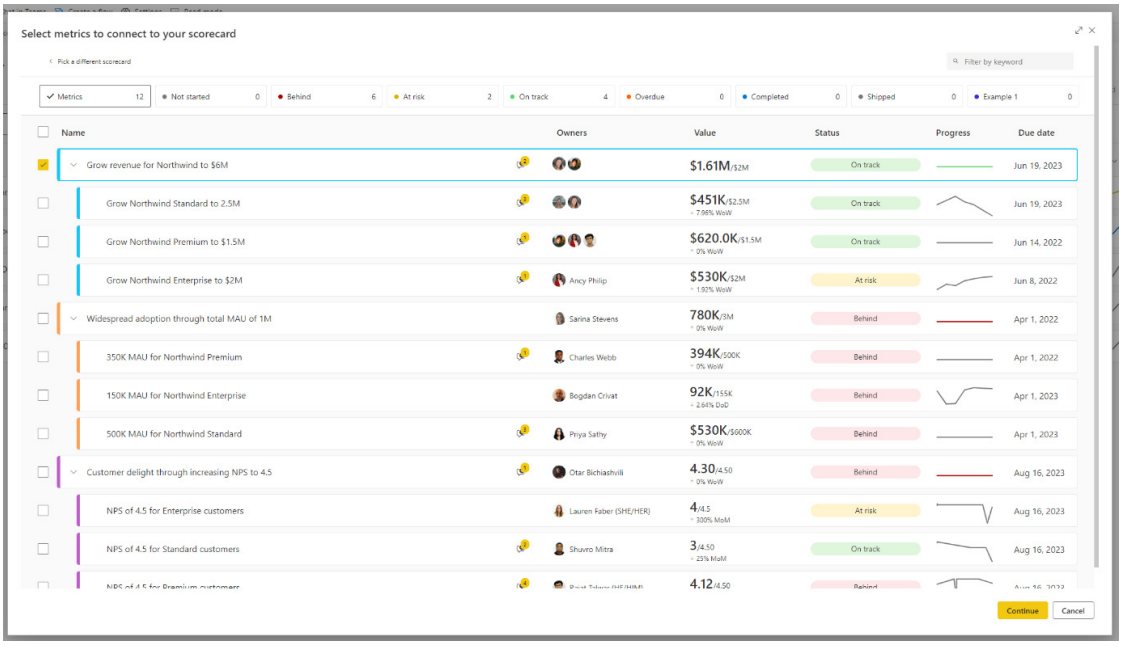
You can link metrics from a scorecard you have build access to (*i.e.* permissions to build content with the data associated with the scorecard), to any scorecards to which you have edit access. To link a metric, go to edit mode and select 'Link to existing metric' from the New button.



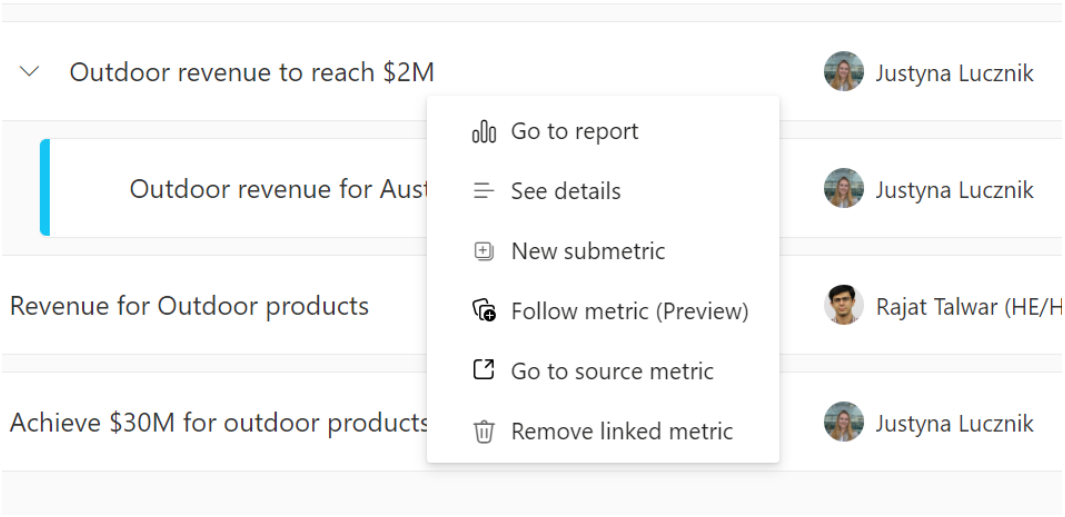
Pick the scorecard that contains the metric you'd like to link.



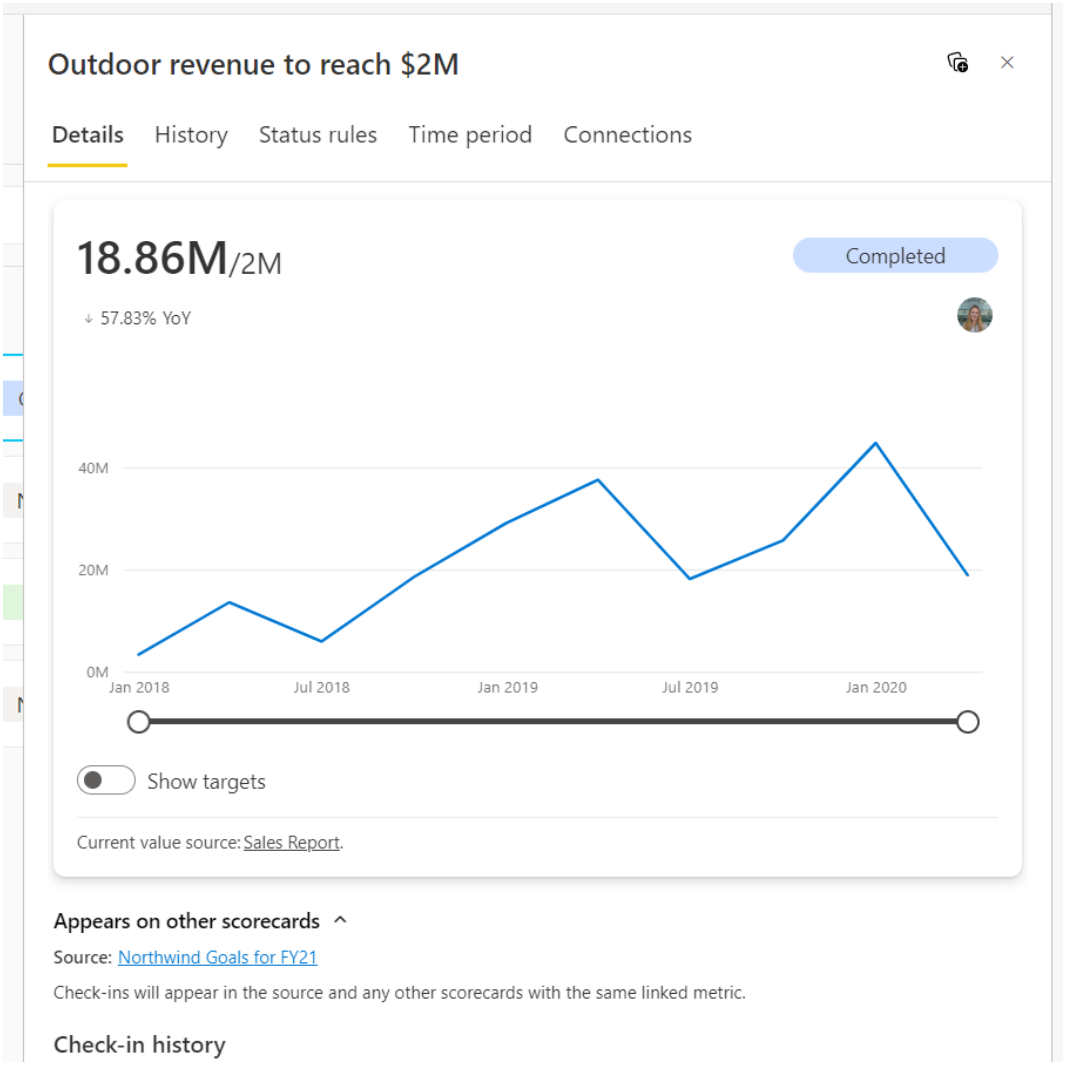
Now select the metric(s) you'd like to include in this scorecard.

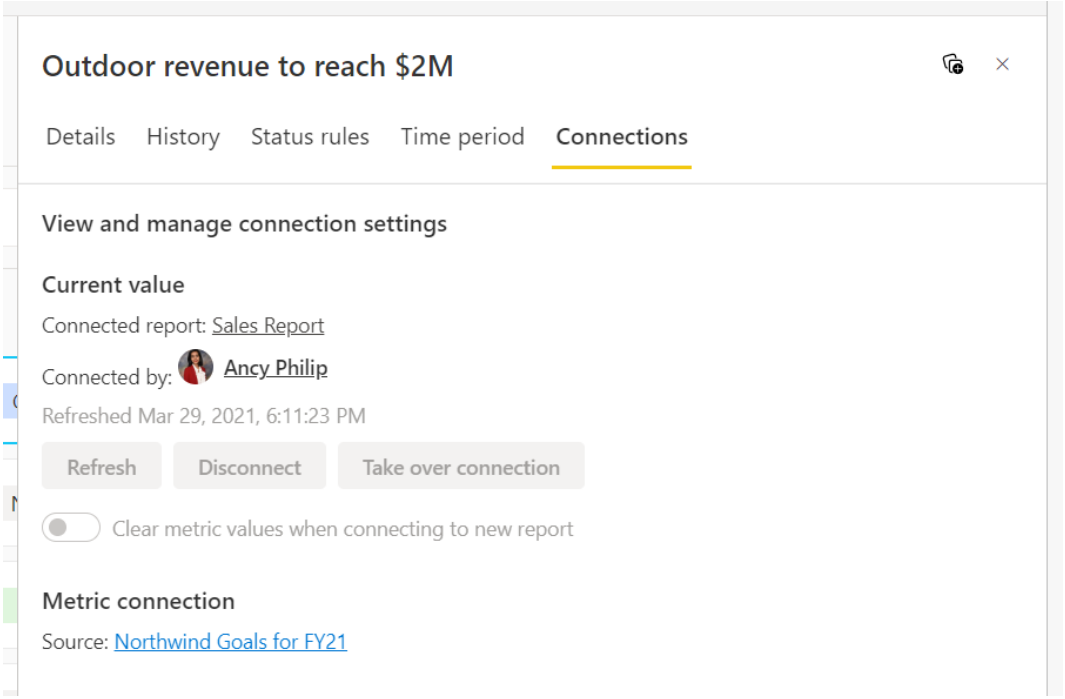


Once you hit 'Continue' the linked metric(s) will appear on the scorecard. You should note that you can add check-ins from the linked metric, but you can make edits only in the source metric. You can navigate from the linked metric to the source metric by selecting 'Go to source metric' from the overflow menu of the metric.



You can see the name and link to the source scorecard in the Details pane and in the Connections pane.





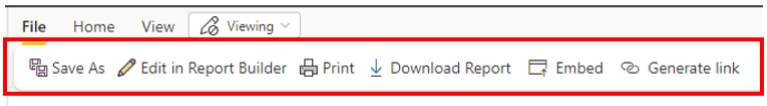
Information protection update

Power BI enables you to import files from OneDrive or SharePoint Online into the Power BI Service, and ensures your work in Power BI Desktop stays coordinated with the Power BI Service. Now you may also import files from OneDrive or SharePoint Online when they have sensitivity labels applied (with some exceptions).

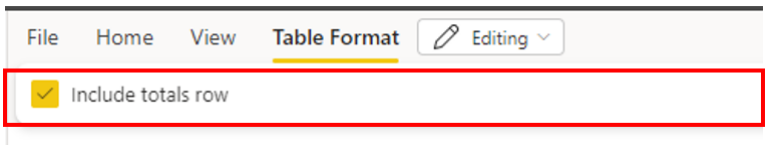
Formatted Table authoring experience

In the last update, Power BI introduced 'show query' and 'column resizing' for data preview. Since then, Microsoft has added some additional capabilities to formatted tables as well.

The Ribbon toolbar contains a series of controls organised into groups. The Ribbon tab provides flexibility to represent functionality visually. For instance, contextual tabs only appear when an object is selected. The Ribbon tab commands are presented horizontally from left to right. For accessibility, Press **CTRL + F6** to navigate to the Ribbon section. Once there, you may use **TAB** to move between the top and bottom bars and use arrow keys to move between elements.

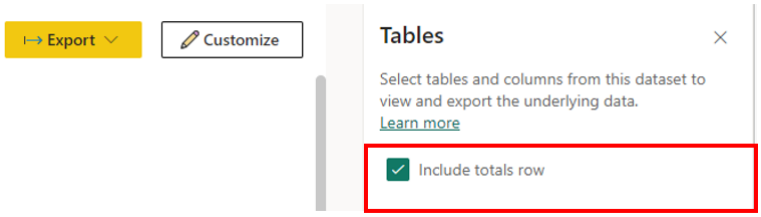


Furthermore, 'Grand Totals for Table Format' allows you to format a selected table, with the ability to include or exclude grand totals.



This feature is especially useful to show totals on explicit measures in formatted tables. Grand totals are also useful for summary reports of data, although it should be noted that the totals row will only be selectable if you include an explicit or implicit measure in your table.

But what if you are making some discovery of your data and you decide to pull data fields from a dataset in 'Table preview'? Grand totals are available as a toggle in the Data Preview as well.



Right-clicking the mouse button will allow you to cut, copy, paste and remove objects on the canvas. Users can resize the width of columns through a drag handle. This functionality makes the formatted table more readable, especially for long column input values. A selected table can be repositioned on the canvas. This will become handy when you want to move visuals around and prepare your report for print layouts. The great thing about these options is when you save, export, or print your report, the formatting is preserved in your newly created report.

New visuals in AppSource

These are the new visuals this month:

- Traqplan Timeline and Roadmaps
- Supermetrics Charts – Tile grid map
- CP Radar Chart
- Sankey Diagram for Power BI by ChartExpo
- Gantt by Lingaro.

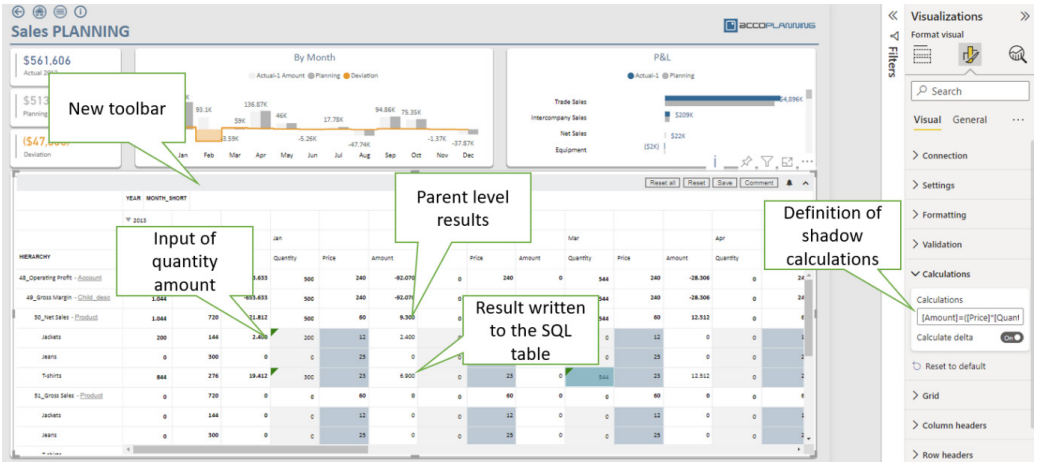
accoPLANNING by Accobat

This enables writeback for any Power BI model. You can use Power BI for planning and forecast.

With this visual you can now writeback numbers, text and dates on any existing or new Power BI model. There are new advanced features like leaf level shadow calculation, enhanced copy and paste functionality. and the direct purchase and maintenance of licensing through AppSource and Office 365.

Key new functionalities:

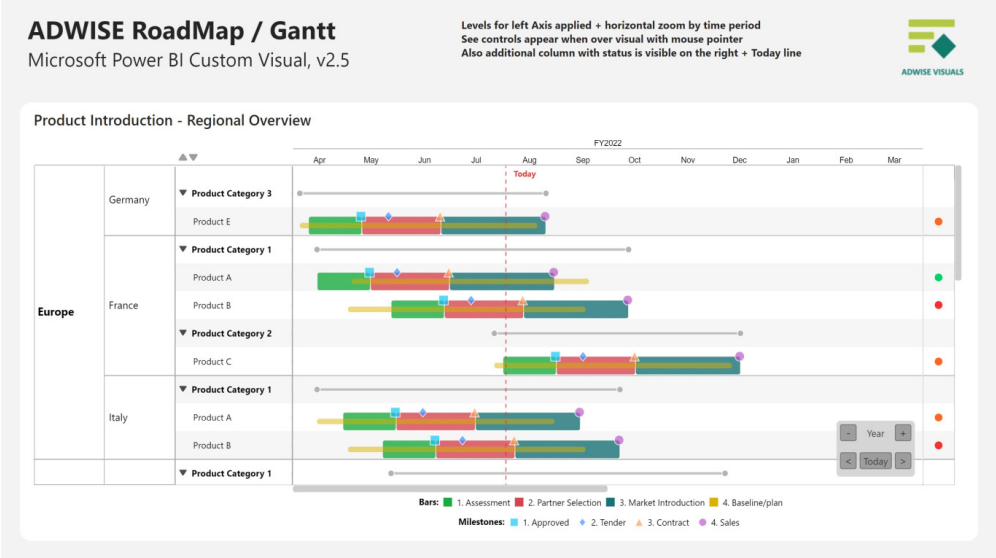
1. hadow calculations
    a. enables you to avoid performance overhead on your leaf calculations like **Price * Quantity**
    b. you will benefit from having the result of your calculations stored in the writeback table for use in other models or systems
    c. realtime calculations while you are typing
2. General enhancements
    a. optimised and extended functionality for copying and pasting of data from Excel, providing you with a fast and easy way to add data to your model
    b. better handling of typing long comments and text with cell scrollbar and mouse over preview of text
3. AppSource license handling
    a. you can now purchase license directly from the AppSource
    b. easy and flexible license allocation to relevant users though the Office 365 Admin portal.



With the accoPLANNING visual, you combine the planning and reporting process in Power BI.

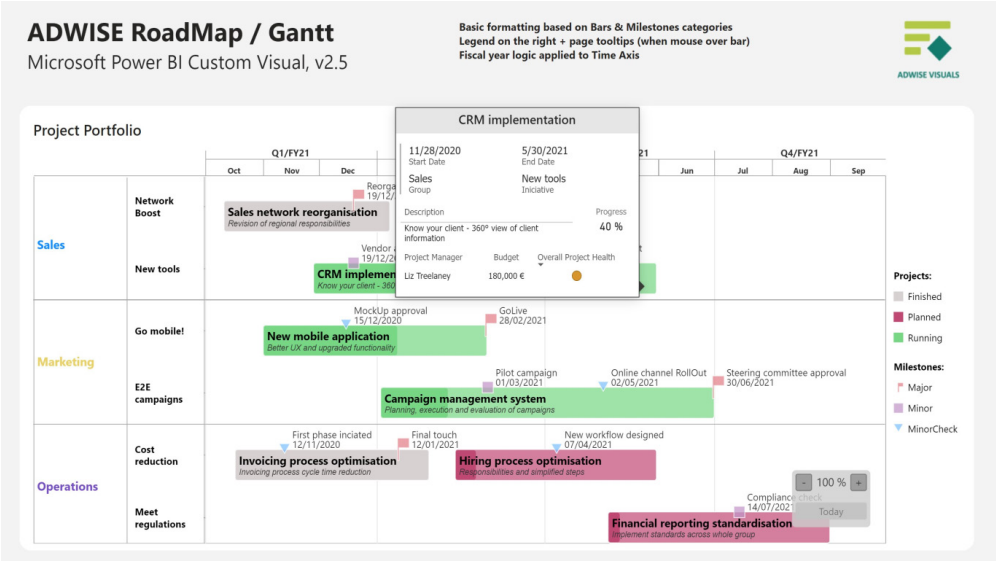
ADWISE RoadMap / Gantt v2.5

This is a flexible and comprehensible visual for reporting plans, schedules and roadmaps. It intends to clearly present your project / portfolio, product(s) or campaign plans, or plans of whatever else that can be time-scheduled and present your phases as well as key milestones, all in a one-page or scrollable view.



There are significant updates in this version 2.5:

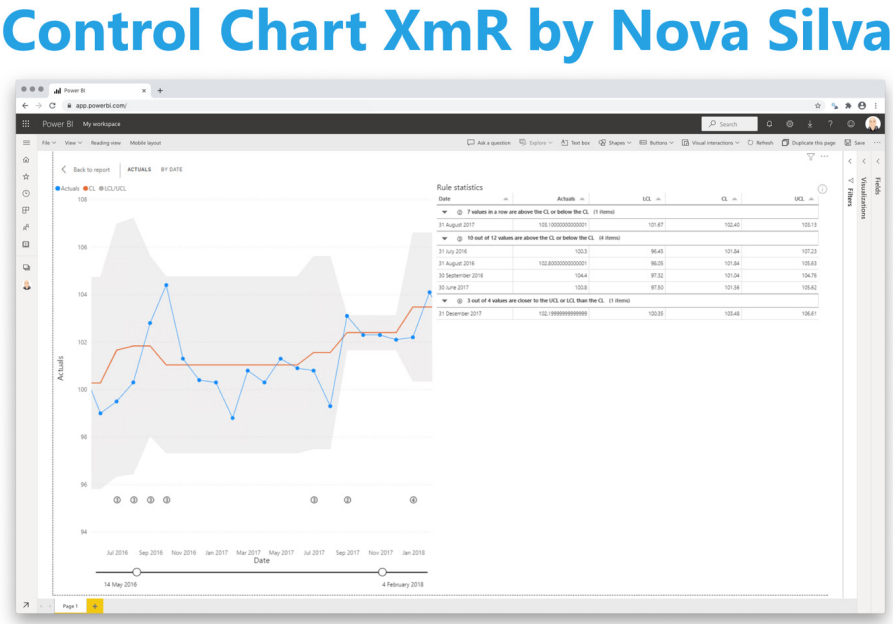
- **Tree hierarchy:** expandable/collapsible tree levels within Left Axis with separate formatting and indentation
- **Groups and SubGroups:** one more hierarchy level (column) to show in the visual
- **Actual vs. Plan in Bar:** used for plan or baseline visualisation
- **Additional Info column:** space for any additional information you want to display
- **Legend:** automatically generated from Category field for Bars and / or Milestones
- **Fiscal year logic:** set the month for start of fiscal year
- **Time interval / period zoom:** you can zoom by day / week / month / quarter / year
- **Go To Today:** centre visual around 'Today' date
- **Multiple new formatting options:** row and group background, milestones positioning, gridline colours, *etc.*



This visual offers in-app purchases and comes with 30 days free trial version. You have full experience of our visual during trial period, with all features accessible.

Control Chart XmR by Nova Silva

In the last release of the Control Chart XmR, several new features were added to make it easier to find real signals and ignore random noise in your data. First, ZoomSlider support was added to the Control Chart XmR to make it easier to navigate through time:



Second, the visualisation also allows for multiple targets, so you may define more than just one target.

Finally, you may now review all the signals in a table when you view the Control Chart XmR in "focus mode".

The new Control Chart XmR may be downloaded from the AppSource. All features are available for free to evaluate this visual within Power BI Desktop.

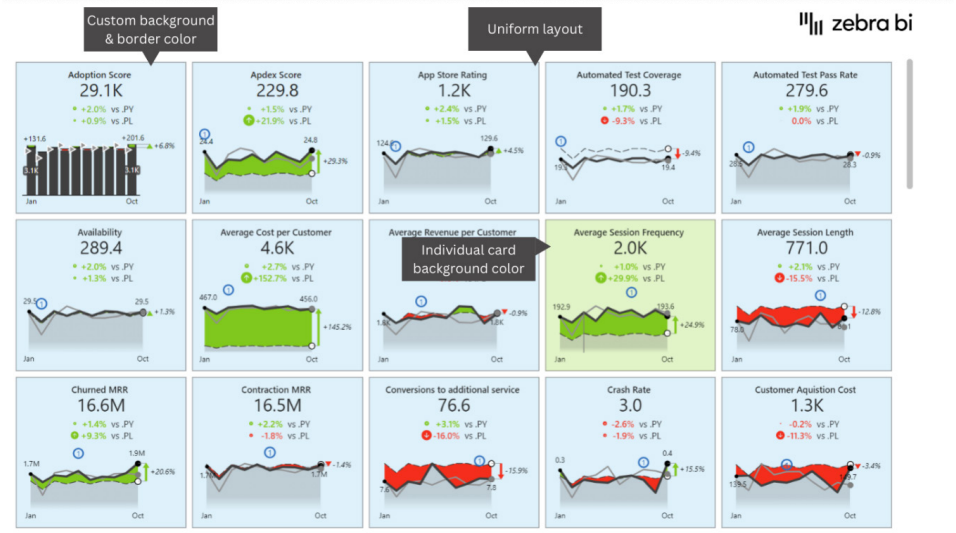
Zebra BI Cards 1.3

Zebra BI Cards visual is bringing a new layout possibility which makes it the only custom visual with three different layouts for displaying KPIs. Furthermore, reports with Zebra BI Cards have become more accessible with custom background colours.

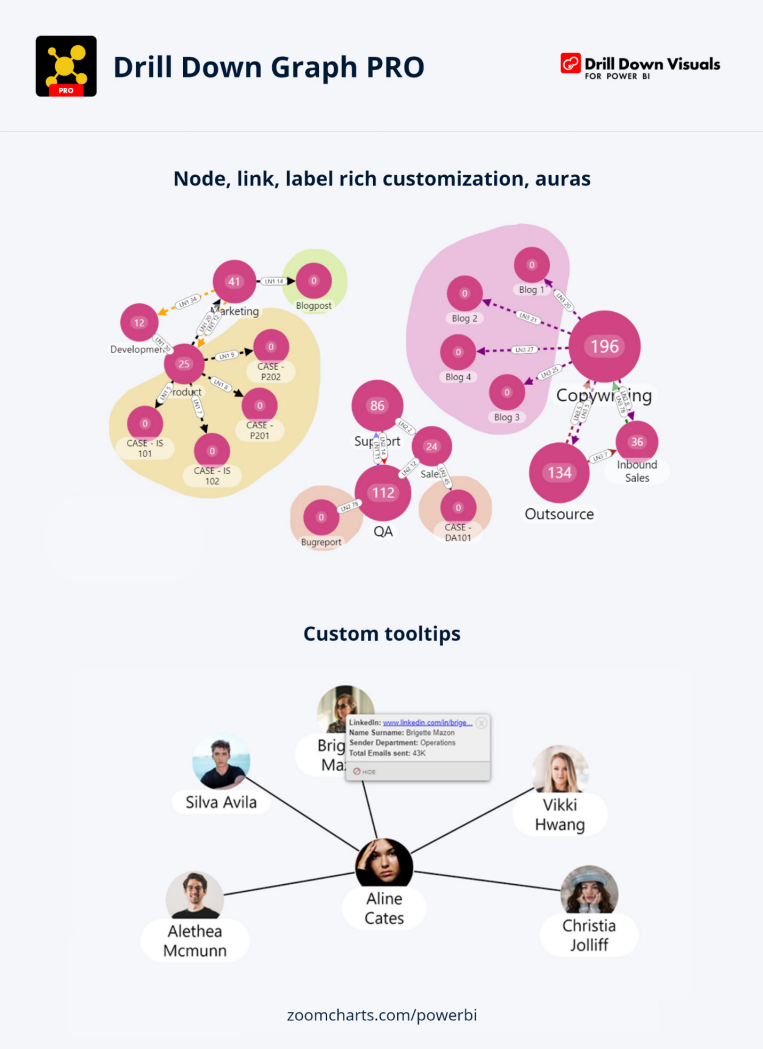
Users can now display the KPIs in three different layouts: custom, rows, or uniform. While the first one gives you the freedom to adjust each card's size separately, the latter enables you to show all the cards in the same size. There is also a ToolTip indicating the size of the cards, which means you may display as many KPIs as you require inside just one visual and adjust the same size for all with a simple drag.

Also, Zebra BI Cards visual lets you adjust the background and border colour of all cards globally or just of specific KPIs. Users may choose any colour from the colour-picker in the global toolbar. This gives you the ability to:

- match your reports with company branding
- highlight only specific KPIs
- create dark-themed reports and still clearly show the KPIs.



ZoomCharts Drill Down Graph PRO



Notable features include:

- multiple graph layout options: dynamic, hierarchical, radial
- on-chart interactions
- focus mode support
- customisation options: choose colour, shape, images and labels
- bidirectional links
- cross-chart filtering.

Popular use cases:

- **Information Technologies:** asset management, IT infrastructure, IoT monitoring
- **Logistics and Transportation:** fleet management, stock management, parcel tracking
- **Sales and Marketing:** community detection, account management, web analytics, *etc*

ZoomCharts Drill Down Visuals are known for interactive drilldowns, smooth animations and numerous customisation options. They support interactions, selections, custom and native ToolTips, filtering, bookmarks and context menus.

Power BI Desktop infrastructure update (WebView2).

The move to WebView2 as part of the infrastructure update continues. As Microsoft gets closer to Generally Available, they wish to ensure they have solved as many issues as possible. Starting with this release, Power BI Desktop will check if you don't have WebView2 installed yet, and if you don't, it will install it for you. If that installation fails, you will see the following warning every time you launch desktop if you don't have WebView2 installed yet.

This is one of the final steps to be taken before making WebView2 mandatory for using Power BI Desktop, so you should install WebView2 and enable it now if you haven't done so already.

That's it for this month: more soon no doubt!

New Features for Excel

The latest updates see the Navigation pane is now Generally Available for Excel for Windows, and for Windows and Mac Insiders, Automate Tasks with Office Scripts enables you to automate repetitive tasks in your Excel work. There are other additions / improvements too, with the full list as follows:

Excel for the web

- Chart Data Foils

Excel for Windows

- Show Changes
- Data from Picture

Excel for Mac

- Show Changes
- Data from Picture

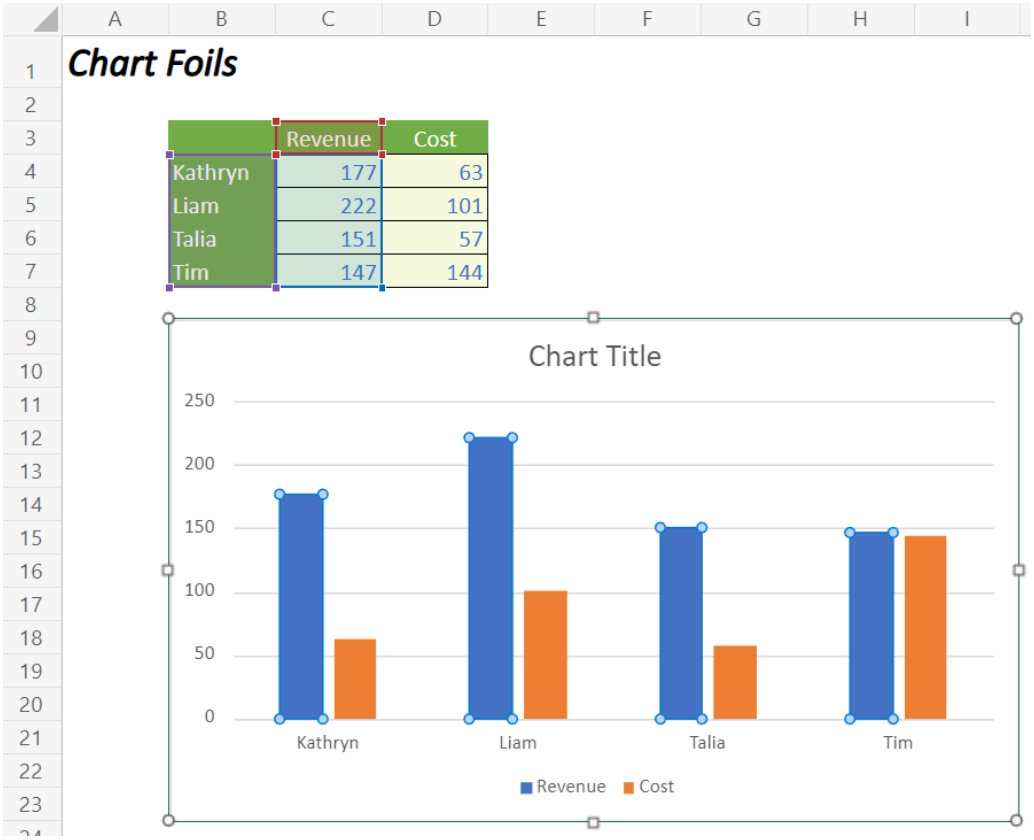
iOS

- Get Fast and Fluent for Office on iPhone (Insiders).

Let's plough through.

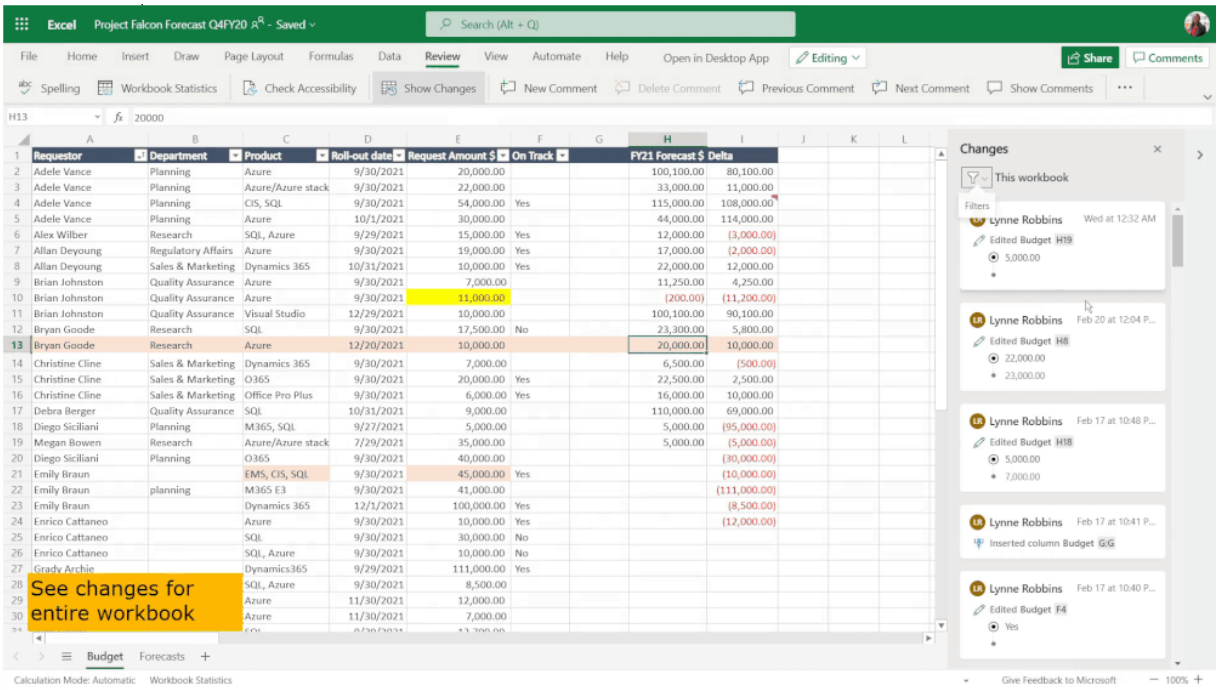
Chart Data Foils

Chart Data Foils provides a visual indicator directly on the spreadsheet highlighting the source range of cells for charts. Users can easily modify that range with the click and drag of a mouse. After a user has created a chart, they can now leverage Data Foils to modify which rows / columns of data are used for their chart including its series titles, series values and category values.



Show Changes

'Show Changes' in Excel is now available in both Excel for Windows and Excel for Mac. It lets you see exactly what edits were made to your workbooks, so you can confidently allow others to collaborate on your work. You can see details of who changed what, where, and when, along with the previous value of the cell for quick reversion. You can narrow down the list of changes by selecting any sheet, range or individual cell, to see all changes that were made, including bulk edits. You can see past changes for up to 60 days.



You can view changes for the entire workbook as follows:

- in the Review tab, select 'Show Changes'
- changes are shown in the pane with the most recent changes on top, in the order the changes were made
- you can see who made edits, exactly where in the workbook, when, and what they changed
- you can also see changes made at once by clicking on 'See changes' in a bulk card.

It is possible to filter and see changes for a subset or range instead:

- select any sheet, range or single cell
- right-click to open the context menu and select 'Show Changes'.

If you or others edit the workbook while this pane is open, select the 'See New Changes' button to update the changes. The notification lets you stay in control when you want to refresh the list and not be distracted when reviewing changes.

You may also filter changes from the Changes pane. At the top of the pane, select the Filter icon and then choose the Range or Sheet to filter the changes shown. Then, do the following:

- to show a range or cell's changes, select Range and enter the range or cell in the text box
- select the arrow icon next to the text box to confirm.

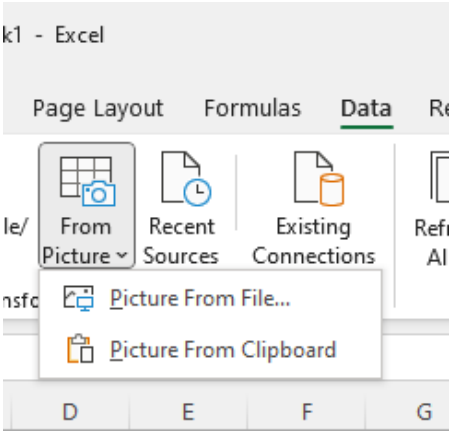
Data from Picture

'Data from Picture' is also now available in both Excel for Windows and Excel for Mac. With the new 'Data from Picture', you can insert data from a picture on your clipboard or an image file from your computer. There's no need to type all the data: Excel can do it for you.

If you have ever had to type data from a printed page into Excel, wouldn't it be a lot easier if you could take a picture, and then have Excel do the tedious work of inputting the data for you? Well, with this new 'Data from Picture' feature in Excel, you can insert data from a picture on your clipboard or an image file from your computer.

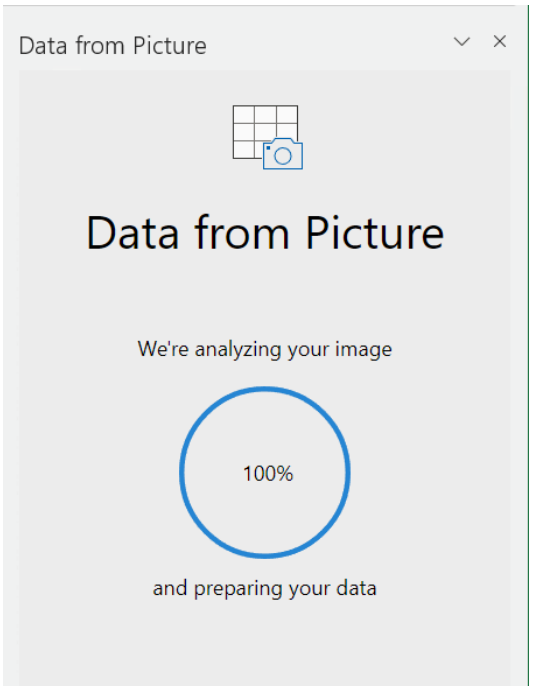
To get it to work, use one of the options below to capture the content you want to digitise:

- Select **Data -> From Picture -> Picture From File**

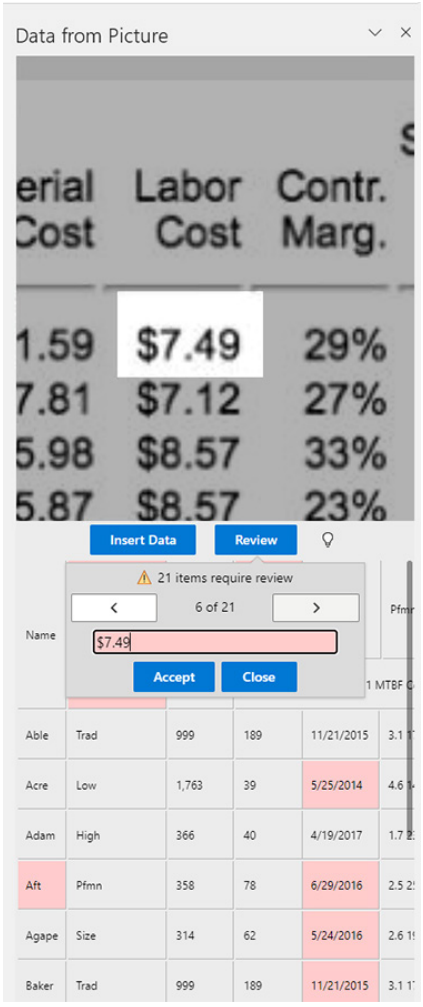


- Copy an image of a table to your clipboard. For example, take a screenshot of a table by pressing **Windows Key + SHIFT + S**. Then, select **Data -> From Picture -> Picture From Clipboard**.

After you have captured the picture of the content you want to bring into Excel, the 'Data from Picture' pane appears. This shows you the progress as the image is being analysed.



You may review the results and make any corrections necessary, and then select 'Insert Data': the data is now in your Excel worksheet.



You can consider the following scenarios:

- **Screen capture a table from a website.** If you've ever tried to copy and paste data from a website, you've likely noticed that it often results in formatting discrepancies. Instead, capture an image of the table (by pressing **Windows Key + SHIFT + S**), then select **Data -> From Picture -> Picture From Clipboard**. Follow the instructions on the screen: you should get just what you're looking for
- **Take a picture of some printed data.** Maybe you'd like to get data from your previous tax returns into Excel and you only have printed copies. This is straightforward. Simply take a picture of each one and transfer the pictures to your Windows computer (OneDrive may be used here). Then, select **Data -> From Picture -> Picture From File**. Again, follow the on-screen instructions to convert the picture to data.

This feature supports the following languages (and recognises these character sets): English, Bosnian, Croatian, Czech, Danish, Dutch, Finnish, French, German, Hungarian, Italian, Norwegian, Polish, Portuguese, Romanian, Serbian, Slovak, Slovenian, Spanish, Swedish and Turkish.

Get Fast and Fluent for Office on iPhone

These updates bring a new look and feel to the contextual command bar and Ribbon palette when working with documents, presentations and worksheets on an iPhone. The new cleaner and simplified experience include:

- a visual update of both the contextual command bar and Ribbon palette to align with Microsoft's Fluent design, implemented using our open-source components
- more performant menus on the contextual command bar that open faster and are resizable.

The updated version of the grid with all the new features is fast becoming too complicated to show clearly here. Nonetheless, you can find the interactive links at aka.ms/ExcelFeaturesFlyer.

Excel Features Availability

	Insider		Production				
Feature	Windows Find the latest Excel version for this platform	Mac Find the latest Excel version for this platform	Windows/CC Find the latest Excel version for this platform	Windows/MEC Find the latest Excel version for this platform	Windows/SA Find the latest Excel version for this platform	Mac Find the latest Excel version for this platform	Web
Chart Data Foils							November 2022
Show Changes			Version 2209 (Build 15703.10000) or later			Version 16.66 (Build 22092500) or later	Already Supported
Data from picture			Version 2210 (Build 15723) or later			Version 16.38 or later	
New Paste Options	Version 2210 (Build 15726.20000) or later						
Quickly Find the Command you need			Version 2206 (Build 15331.20010) or later				October 2022
New DAX Functions	Version 2208 (Build 15504.10000) or later						
Navigation Pane			Version 2209 (Build 15629.10000) or later				
Smooth Scrolling			Version 2205 (Build 15225.20092) or later	Version 2208 (Build 15601.20230)		Already Supported	Already Supported
Check Performance							September 2022
Share Section of Excel Workbook							September 2022
Dynamic Array Support in Charts	Version 2209 (Build 15617.10000) or later						September 2022
Modern Comments			Version 2209 (Build 15427.20000) or later				
Manage Your Storage Accounts from Mac		Version 16.64 (Build 22082100) or later					
New Excel functions			Version 2208 (Build 15427.20194) or later			Version 16.64 (Build 22081401) or later	August 2022

Excel Features Availability

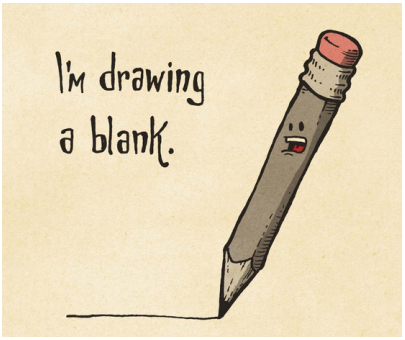
	Insider		Production				
Feature	Windows Find the latest Excel version for this platform	Mac Find the latest Excel version for this platform	Windows/CC Find the latest Excel version for this platform	Windows/MEC Find the latest Excel version for this platform	Windows/SA Find the latest Excel version for this platform	Mac Find the latest Excel version for this platform	Web
Power Query Group operations							August 2022
Improvements to the connected Power BI experience	Version 2208 (Build 15601.20028) or later						August 2022
Add and edit rich text formatting			Already Supported	Already Supported	Already Supported	Already Supported	August 2022
Sort by color or icon from auto filter menu			Already Supported	Already Supported	Already Supported	Already Supported	August 2022
Edit files with legacy data connections			Already Supported	Already Supported	Already Supported	Already Supported	August 2022
Edit files with legacy Shared Workbook feature			Already Supported	Already Supported	Already Supported	Already Supported	August 2022
Delete chart elements							August 2022
Multiline formula bar							August 2022
IMAGE function	Version 2209 (Build 15608.10000) or later	Version 16.65 (Build 22080701) or later					
Search within PivotTable Field List			Already Supported	Already Supported	Already Supported	Already Supported	July 2022
Set automatic data conversions	Version 2207 (Build 15427.20000) or later						
Natural Language Query Improvements			Version 2206 (Build 15330.20230) or later	Version 2205 (Build 15225.20356) or later		Version 16.63 (Build 22070801) or later	
Resize Conditional Formatting dialog box		Version 16.64 (Build 22070600) or later					
Sheet protection			Already Supported	Already Supported	Already Supported	Already Supported	June 2022

Excel Features Availability

	Insider		Production				
Feature	Windows Find the latest Excel version for this platform	Mac Find the latest Excel version for this platform	Windows/CC Find the latest Excel version for this platform	Windows/MEC Find the latest Excel version for this platform	Windows/SA Find the latest Excel version for this platform	Mac Find the latest Excel version for this platform	Web
Semi-select for links creation			Already Supported	Already Supported	Already Supported	Already Supported	June 2022
Add "PivotTable Connections to Slicer settings pane			Already Supported	Already Supported	Already Supported	Already Supported	June 2022
Import from local text, CSV, and XLSX files						Version 16.57 (22011100) or later	
Provide automatic alt-text suggestions on charts and PivotCharts			Version 2205 (Build 15225.20288) or later	Version 2204 (Build 15128.20280) or later		Version 16.62 (22061100) or later	
Power Query refresh for selected data sources			Already Supported	Already Supported	Already Supported	Already Supported	May 2022
Changing source file for workbook links			Already Supported	Already Supported	Already Supported	Already Supported	May 2022
Improved Recommended PivotTable experience	Version 2204 (Build 15128.10000) or later						
Faster recalc on resource constrained devices		Version 16.62 (Build 22050804) or later	Version 2204 (Build 15128.20248) or later	Version 2204 (Build 15128.20280) or later			
Faster AutoFilter				Version 2204 (Build 15128.20248) or later		Version 16.61 (22050700) or later	
Dataflow connector				Version 2203 (Build 15028.20248) or later			
Dataverse connector			Version 2204 (Build 15128.20178) or later				
Shaping data with Power Query Editor		Version 16.64 (Build 22072501) or later					
Improved Find dialog and Find All						Version 16.60 (220410) or later	

More next month, we're sure.

The A to Z of Excel Functions: ISBLANK



The latest updates see the Navigation pane is now Generally Available for Excel for Windows, and for Windows and Mac Insiders, Automate Tasks with Office Scripts enables you to automate repetitive tasks in your Excel work. There are other additions / improvements too, with the full list as follows:

1. **ISBLANK(reference):** checks whether the **reference** is to an empty cell
2. **ISERR(value):** checks whether the **value** is an error (*e.g. #REF!, #DIV/0!, #NULL!*). This check specifically excludes *#N/A*
3. **ISERROR(value):** checks whether the **value** is an error (*e.g. #REF!, #DIV/0!, #NULL!*). This is probably the most commonly used of these functions in financial modelling
4. **ISEVEN(number):** checks to see if the **number** is even
5. **ISFORMULA(reference):** checks to see whether the **reference** is to a cell containing a formula
6. **ISLOGICAL(value):** checks to see whether the **value** is a logical (TRUE or FALSE) value
7. **ISNA(value):** checks to see whether the **value** is *#N/A*. This gives us the rather crude identity **ISERR + ISNA = ISERROR**
8. **ISNONTEXT(value):** checks whether the **value** is not text (*N.B.* blank cells are not text)
9. **ISNUMBER(value):** checks whether the **value** is a number
10. **ISODD(number):** checks to see if the **number** is odd. Personally, I find the number 46 very odd, but Excel doesn't
11. **ISREF(value):** checks whether the **value** is a reference
12. **ISTEXT(value):** checks whether the **value** is text.

As stated above, the **ISBLANK** function checks whether the reference is to an empty cell and returns either TRUE or FALSE accordingly. It has the following syntax:

ISVALUE(reference)

The **ISBLANK** function has the following argument:

- **reference:** this is required and represents the **reference** for which you wish to determine whether it is completely empty (blank).

It should be further noted that:

- cells with formulas that return "" (empty text) are not counted. This is different behaviour to the **ISBLANK** function which deem empty text to be blank, *i.e.* **COUNTA(range) + COUNTBLANK(range)** does not necessarily equal the number of cells in the **range**, which many modellers assume to be an identity
- cells with spaces, zeroes or error values are not considered blank.

Please see our example below:

	A	B	C
1	Data	Empty Text	
2	07/01/2020		=""
3	8		
4	0		
5			
6	TRUE		
7	#DIV/0!		
8			
9			

10	Formula	Description	Result
11	=ISBLANK(A2)	Ascertains whether cell A2 is blank.	FALSE
12	=ISBLANK(A3)	Ascertains whether cell A3 is blank.	FALSE
13	=ISBLANK(A4)	Ascertains whether cell A4 is blank.	FALSE
14	=ISBLANK(A5)	Ascertains whether cell A5 is blank.	TRUE
15	=ISBLANK(A6)	Ascertains whether cell A6 is blank.	FALSE
16	=ISBLANK(A7)	Ascertains whether cell A7 is blank.	FALSE
17	=ISBLANK(B2)	Ascertains whether cell B2 is blank.	FALSE
18	=COUNTBLANK(B2)	Counts whether cell B2 is blank.	1

The A to Z of Excel Functions: ISERR

As stated in discussing the previous function, the **ISERR** function checks whether the value is an error and returns either TRUE or FALSE accordingly. This specifically checks for all types of *prima facie* errors (*e.g. #VALUE!*, *#REF!*, *#NAME?* and the relatively new ones, *#CALC!*, *#SPILL!* and *#FIELD!*) except for *#N/A.* It has the following syntax:

ISERR(value)

The **ISERR** function has the following argument:

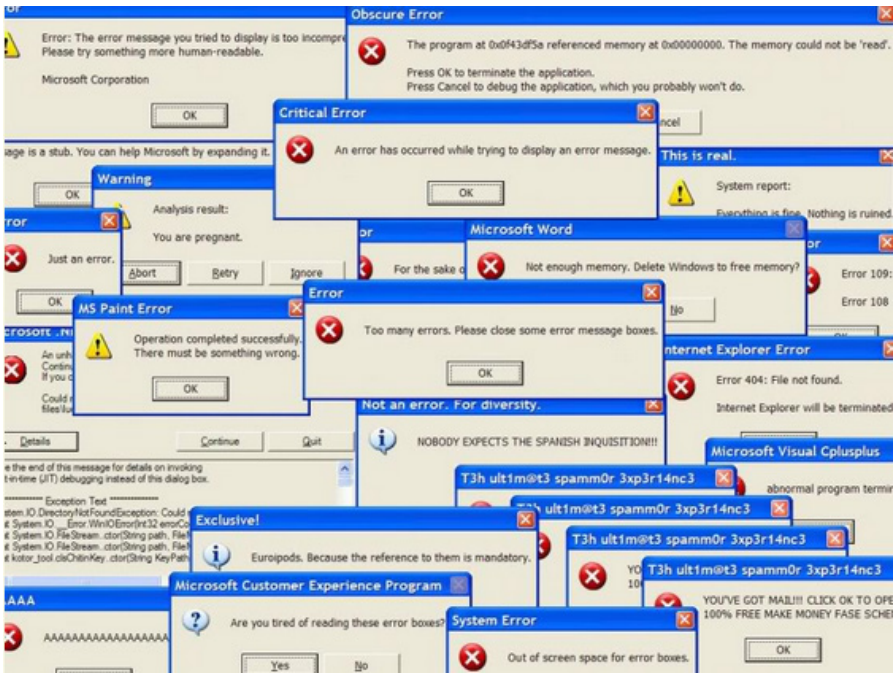
- **value:** this is required and represents the **value** for which you wish to determine whether it is an error (excluding *#N/A*).

It should be further noted that:

- technically, *#N/A* is not an error: it is a special value that you may manually enter into a cell to indicate that the necessary value is not available yet
- *#####* is also not technically an error: this denotes that the column is not wide enough to display all the characters required in the current cell width
- the *#CALC!* and *#SPILL!* errors are only recognised in versions of Excel that support dynamic arrays; otherwise, these are treated as text strings and therefore are not considered errors.

Please see examples below:

	A	B	C	D
1	Data	Formula	Description	Result
2	################	=ISERR(A2)	Ascertains whether cell A2 is an applicable error.	FALSE
3	#CALC!	=ISERR(A3)	Ascertains whether cell A3 is an applicable error.	TRUE
4	#DIV/0!	=ISERR(A4)	Ascertains whether cell A4 is an applicable error.	TRUE
5	#FIELD!	=ISERR(A5)	Ascertains whether cell A5 is an applicable error.	TRUE
6	#GETTING_DATA	=ISERR(A6)	Ascertains whether cell A6 is an applicable error.	TRUE
7	#N/A	=ISERR(A7)	Ascertains whether cell A7 is an applicable error.	FALSE
8	#NAME?	=ISERR(A8)	Ascertains whether cell A8 is an applicable error.	TRUE
9	#NULL!	=ISERR(A9)	Ascertains whether cell A9 is an applicable error.	TRUE
10	#NUM!	=ISERR(A10)	Ascertains whether cell A10 is an applicable error.	TRUE
11	#REF!	=ISERR(A11)	Ascertains whether cell A11 is an applicable error.	TRUE
12	#SPILL!	=ISERR(A12)	Ascertains whether cell A12 is an applicable error.	TRUE
13	#VALUE!	=ISERR(A13)	Ascertains whether cell A13 is an applicable error.	TRUE
14				



The **ISERROR** function checks whether the value is an error and returns either TRUE or FALSE accordingly. This specifically checks for all types of *prima facie* errors (*e.g. #VALUE!*, *#REF!*, *#NAME?* and the relatively new ones, *#CALC!*, *#SPILL!* and *#FIELD!*) including *#N/A*. It has the following syntax:

ISERROR(value)

The **ISERROR** function has the following argument:

- **value:** this is required and represents the **value** for which you wish to determine whether it is an error.

It should be further noted that:

- technically, *#N/A* is not an error: it is a special value that you may manually enter into a cell to indicate that the necessary value is not available yet. If you wish to exclude this, use **ISERR** instead
- *#####* is also not technically an error: this denotes that the column is not wide enough to display all the characters required in the current cell width
- the *#CALC!* and *#SPILL!* errors are only recognised in versions of Excel that support dynamic arrays; otherwise, these are treated as text strings and therefore are not considered errors
- **IFERROR** and **IF(ISERROR)** are different; **IFERROR** calculates the expression if there is no error; you may choose to calculate a different expression if there is no error with **IF(ISERROR)**, *e.g.* **IFERROR(5/2,0)** = 2.5 but **IF(ISERROR(5/2),0,1)** = 1.

Please see more examples below:

	A	B	C	D
1	**Data**	**Formula**	**Description**	**Result**
2	################	***=ISERROR(A2)***	Ascertains whether cell **A2** is an applicable error.	**FALSE**
3	#CALC!	***=ISERROR(A3)***	Ascertains whether cell **A3** is an applicable error.	**TRUE**
4	#DIV/0!	***=ISERROR(A4)***	Ascertains whether cell **A4** is an applicable error.	**TRUE**
5	#FIELD!	***=ISERROR(A5)***	Ascertains whether cell **A5** is an applicable error.	**TRUE**
6	#GETTING_DATA	***=ISERROR(A6)***	Ascertains whether cell **A6** is an applicable error.	**TRUE**
7	#N/A	***=ISERROR(A7)***	Ascertains whether cell **A7** is an applicable error.	**TRUE**
8	#NAME?	***=ISERROR(A8)***	Ascertains whether cell **A8** is an applicable error.	**TRUE**
9	#NULL!	***=ISERROR(A9)***	Ascertains whether cell **A9** is an applicable error.	**TRUE**
10	#NUM!	***=ISERROR(A10)***	Ascertains whether cell **A10** is an applicable error.	**TRUE**
11	#REF!	***=ISERROR(A11)***	Ascertains whether cell **A11** is an applicable error.	**TRUE**
12	#SPILL!	***=ISERROR(A12)***	Ascertains whether cell **A12** is an applicable error.	**TRUE**
13	#VALUE!	***=ISERROR(A13)***	Ascertains whether cell **A13** is an applicable error.	**TRUE**

The A to Z of Excel Functions: ISEVEN

The **ISEVEN** function checks whether a number is even. It has the following syntax:

ISEVEN(number)

The **ISEVEN** function has the following argument:

- **number:** this is required and represents the **number** for which you wish to determine whether it is even. If **number** is not an integer, it is truncated (*i.e.* not rounded, simply ended).

It should be further noted that:

- if **number** is nonnumeric, **ISEVEN** returns the *#VALUE!* error value.

Please see our examples below:

	A	B	C	D
1	Data	Formula	Description	Result
2	-4.1	=ISEVEN(A2)	Ascertains whether cell A2 is even.	TRUE
3	-3.9	=ISEVEN(A3)	Ascertains whether cell A3 is even.	FALSE
4	0	=ISEVEN(A4)	Ascertains whether cell A4 is even.	TRUE
5	1.99	=ISEVEN(A5)	Ascertains whether cell A5 is even.	FALSE
6	2	=ISEVEN(A6)	Ascertains whether cell A6 is even.	TRUE
7	2.01	=ISEVEN(A7)	Ascertains whether cell A7 is even.	TRUE
8	01-Jan-20	=ISEVEN(A8)	Ascertains whether cell A8 is even (1 Jan 2020 is the serial number 43831).	FALSE
9	dog	=ISEVEN(A9)	Ascertains whether cell A9 is even (it is text).	#VALUE!
10	2	=ISEVEN(A10)	Ascertains whether cell A10 is even (it is text).	TRUE

The A to Z of Excel Functions: ISFORMULA

$$\nabla^2 c = \kappa^2 c,$$
$$\partial c_\mathrm{a}/\partial t = [J_\mathrm{a}^1 \alpha(c, c_\mathrm{a}) + J_\mathrm{a}(c_\mathrm{a})\beta(c, c_\mathrm{a})]R$$
$$+ D_\mathrm{a}\nabla^2 c_\mathrm{a} - k c_\mathrm{i} c_\mathrm{a}$$
$$\partial c_i/\partial t = D_\mathrm{i}\nabla^2 c_\mathrm{i} - k_\mathrm{a} c_\mathrm{i} c_\mathrm{a} + J_\mathrm{i}(c, c_\mathrm{a})\beta(c, c_\mathrm{a})R$$
$$\partial R/\partial t = [D_\mathrm{cell} - (\lambda + \lambda_2 \gamma(c, c_\mathrm{a}))R]\nabla^2 R$$
$$- \lambda_2 \partial\gamma/\partial c_\mathrm{a} R^2 \nabla^2 c_\mathrm{a} - \lambda_2 \partial\gamma/\partial c R^2 \nabla^2 c$$
$$+ rR(R_\mathrm{eq} - R) - k_{23}\gamma(c, c_\mathrm{a})R.$$

The **ISFORMULA** function checks whether the reference is to a cell that contains a formula. It has the following syntax:

ISFORMULA(reference)

The **ISFORMULA** function checks whether the reference is to a cell that contains a formula. It has the following syntax:

ISFORMULA(reference)

The **ISFORMULA** function has the following argument:

- **reference:** this is required and represents the **reference** for which you wish to determine whether it contains a formula. This can be to one or more cells, a reference or a name that refers to one or more cells.

It should be further noted that:

- if **reference** refers to more than one cell, as long as one cell contains a formula the result will be TRUE (FALSE otherwise)
- if **reference** is not a valid data type, such as a defined name that is not a reference, **ISFORMULA** returns the *#VALUE!* error value.

Please see more examples below:

	A	B	C	D	E
1	Data	Formula	Description	Result	Formula
2	-4.1	*-4.1*	-4.1 is not a formula	FALSE	*=ISFORMULA(A2)*
3	10/01/2020	*=TODAY()*	Cell A3 contains a formula	TRUE	*=ISFORMULA(A3)*
4	SumProduct	*SumProduct*	This is text	FALSE	*=ISFORMULA(A4)*
5	#DIV/0!	*=1/0*	Cell A5 contains a formula, regardless of the fact it evaluates as an error	TRUE	*=ISFORMULA(A5)*
6					

The A to Z of Excel Functions: ISLOGICAL

The **ISLOGICAL** function checks whether the value is logical (*i.e.* TRUE or FALSE). It has the following syntax:

ISLOGICAL(value)

The **ISLOGICAL** function has the following argument:

- **value:** this is required and represents the **value** for which you wish to determine whether it contains a logical (TRUE or FALSE) reference.

It should be further noted that:

- if **value** refers to more than one cell, the formula the result will be FALSE
- if **value** is not a valid data type, such as a defined name that is not a reference, **ISLOGICAL** returns the *#VALUE!* error value.

Please see our examples below:

	A	B	C	D
1	Data	Formula	Description	Result
2	FALSE	*=ISLOGICAL(A2)*	FALSE is logical	TRUE
3	TRUE	*=ISLOGICAL(A3)*	TRUE is logical	TRUE
4		*=ISLOGICAL(A2:A3)*	References to more than one value are not logical	FALSE
5	1	*=ISLOGICAL(A5)*	All numbers (including 1 and 0) are not logical	FALSE
6	#DIV/0!	*=ISLOGICAL(A6)*	Errors are not logical - rather than giving rise to an error	FALSE
7				

The A to Z of Excel Functions: ISNA

The **ISNA** function checks to see whether the value is *#N/A* (value is not available). This gives us the rather crude identity **ISERR + ISNA = ISERROR**. It has the following syntax:

ISNA(value)

The **ISNA** function has the following argument:

- **value:** this is required and represents the **value** for which you wish to determine whether it contains an *#N/A* message.

It should be further noted that:

- technically, *#N/A* is not an error: it is a special value that you may manually enter into a cell to indicate that the necessary value is not available yet
- if **value** is not a valid data type, such as a defined name that is not a reference, **ISNA** returns the *#VALUE!* error value.

Please see the examples below:

	A	B	C	D
1	Data	Formula	Description	Result
2	5	=ISNA(A2)	Cell A2 does not contain #N/A	FALSE
3	#N/A	=ISNA(A3)	Cell A3 contains #N/A	TRUE

The A to Z of Excel Functions: ISNONTEXT

The **ISNONTEXT** function checks to see whether the value is not text. It has the following syntax:

ISNONTEXT(value)

The **ISNONTEXT** function has the following argument:

- **value:** this is required and represents the **value** for which you wish to determine whether it does not contain text.

It should be further noted that:

- if **ISNONTEXT** is FALSE, this does not mean **ISTEXT** is TRUE. For example, *#DIV/0!* Is neither text nor non-text; it is an error
- if **value** is not a valid data type, such as a defined name that is not a reference, **ISNONTEXT** returns the *#VALUE!* error value.

Please see our penultimate examples for this month below:

	A	B	C	D
1	Data	Formula	Description	Result
2	3	=ISNONTEXT(A2)	Cell A2 is not text	TRUE
3	abc	=ISNONTEXT(A3)	Cell A3 is text	FALSE
4	#REF!	=ISNONTEXT(A4)	Cell A4 is neither text nor non-text	TRUE
5				

The A to Z of Excel Functions: ISNUMBER

The **ISNUMBER** function checks to see whether the value is a number. It has the following syntax:

ISNUMBER(value)

The **ISNUMBER** function has the following argument:

- **value:** this is required and represents the **value** for which you wish to determine whether it is a number.

It should be further noted that:

- if **value** is not a valid data type, such as a defined name that is not a reference, **ISNUMBER** returns the *#VALUE!* error value.

Please see our final examples for this month below:

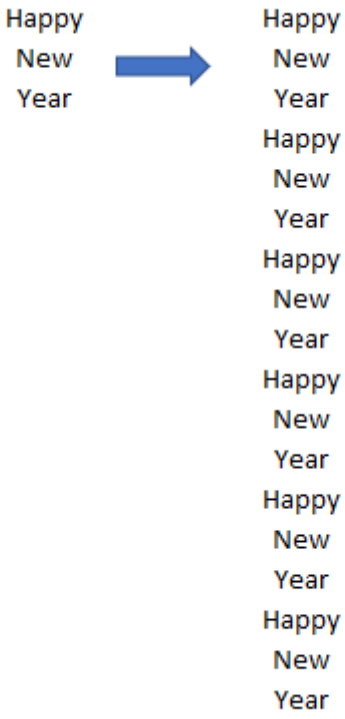
A	B	C	D
Data	Formula	Description	Result
3	=ISNUMBER(A2)	Cell A2 is a number	TRUE
abc	=ISNUMBER(A3)	Cell A3 is not a number	FALSE
#REF!	=ISNUMBER(A4)	Cell A4 is an error and not a number	FALSE

The next function is not part of the "**IS**" family, so we'll take a break here. More Excel Functions next month.

Beat the Boredom Suggested Solution

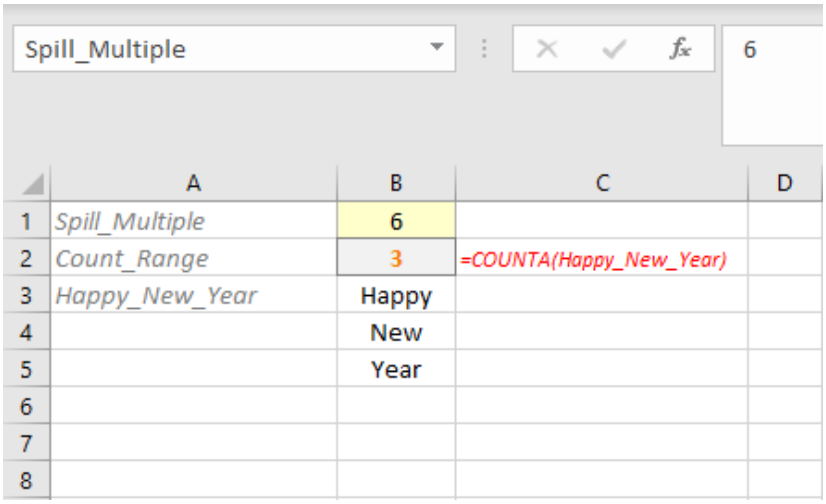
Our first challenge of the new year was to spill a range of cells – "Happy New Year" multiple times using Excel formulae.

Now when we said "spill", we were referring to an Excel formula that returns a range of cells according to user's needs. Data often needs to be organised and arranged in different ways. Now the challenge for this was to spill "Happy", "New" and "Year" six times each as shown in the example below:



This challenge is designed to get you to think outside the box and beyond your basic functions of copy and paste.

First of all, let's create a Dynamic Name Range as follows.



Let's enter a spill multiple (6) in cell **B1** and a count range (*i.e.* how many items, so "Happy", "New" and "Year" are three items) for cell **B2**. Note you have to use the **COUNTA** function, not **COUNT**, as the former counts non-blank cells, whereas the latter only counts numbers. We use cell **B1** to input the number of times that the **Happy_New_Year** range needs to be spilled and name it as **Spill_Multiple**, and cell **B2** to calculate the range count, named as **Count_Range**.

For the stipulated in cell **B3**, we use the **OFFSET** (https://www.sumproduct.com/thought/onset-of-offset) function. To create this, navigate to the Formulas tab on the Ribbon and choose 'Define Name', where we will define the name as follows:

Happy_New_Year = OFFSET($B$3,0,0,COUNTA($B$3:$B$100))

Now, let's go through how to spill the range.

First, we use the **SEQUENCE** (https://www.sumproduct.com/thought/getting-arrays-spilling-the-beans-on-seven-new-functions) function to calculate the number of rows this will need to spill over (*e.g.* if it is six times for an expression three lines deep, we will need a total of 3 x 6 = 18 lines. **SEQUENCE** will set up a counter from 1 to 18 for us down 18 rows).

The formula in cell **E3** is given by

=SEQUENCE(Spill_Multiple*Count_Range,1)

Excel will forgive you if you choose not to employ the second argument (1). This is the assumed default value in any case.

	A	B	C	D	E
1	*Spill_Multiple*	6			**Step 1:**
2	*Count_Range*	3	*=COUNTA(Happy_New_Year)*		*=SEQUENCE(Spill_Multiple*Count_Range,1)*
3	*Happy_New_Year*	Happy			1
4		New			2
5		Year			3
6					4
7					5
8					6
9					7
10					8
11					9
12					10
13					11
14					12
15					13
16					14
17					15
18					16
19					17
20					18
21					

Next, find the index order of the sequencing of **Happy_New_Year** (*i.e.* first word, second word, third word, then repeat a further five times). Here, we may use the **MOD** (https://www.sumproduct.com/thought/a-modicum-of-mod) function. This function is used to return the remainder after the **number** (the first argument) is divided by the **divisor** (the second argument). The formula in cell **F3** is:

=MOD(E3#-1,Count_Range)+1

	A	B	C	D	E	F
1	*Spill_Multiple*	6			**Step 1:**	**Step 2:**
2	*Count_Range*	3	*=COUNTA(Happy_New_Year)*		*=SEQUENCE(Spill_Multiple*Count_Range,1)*	*=MOD(E3#-1,Count_Range)+1*
3	*Happy_New_Year*	Happy			1	1
4		New			2	2
5		Year			3	3
6					4	1
7					5	2
8					6	3
9					7	1
10					8	2
11					9	3
12					10	1
13					11	2
14					12	3
15					13	1
16					14	2
17					15	3
18					16	1
19					17	2
20					18	3

Once we have the indexing, the next step is to simply to "index" the numbers to the text – **Happy_New_Year**. The **INDEX** (https://www.sumproduct.com/thought/index-match) function is used to return a value or the reference to the value from within a table or range (list). In this case, it will match the index order in column **F** with the **Happy_New_Year** range that needs to be spilled. The formula in cell **G3** is:

=INDEX(Happy_New_Year,F3#)

	A	B	C	D	E	F	G
1	*Spill_Multiple*	6			**Step 1:**	**Step 2:**	**Step 3:**
2	*Count_Range*	3	*=COUNTA(Happy_New_Year)*		*=SEQUENCE(Spill_Multiple*Count_Range,1)*	*=MOD(E3#-1,Count_Range)+1*	*=INDEX(Happy_New_Year,F3#)*
3	*Happy_New_Year*	Happy			1	1	Happy
4		New			2	2	New
5		Year			3	3	Year
6					4	1	Happy
7					5	2	New
8					6	3	Year
9					7	1	Happy
10					8	2	New
11					9	3	Year
12					10	1	Happy
13					11	2	New
14					12	3	Year
15					13	1	Happy
16					14	2	New
17					15	3	Year
18					16	1	Happy
19					17	2	New
20					18	3	Year
21							

Even better, if you are an Excel expert you may have combined all three steps into one as shown below. The formula in cell **I4** is:

=INDEX(Happy_New_Year,MOD(SEQUENCE(Spill_Multiple*Count_Range,1)-1,Count_Range)+1)

	A	B	C	D	I	J	K	L
1	*Spill_Multiple*	6			**Combined formula:**			
2	*Count_Range*	3	*=COUNTA(Happy_New_Year)*		*=INDEX(Happy_New_Year,MOD(SEQUENCE(Spill_Multiple*Count_Range,1)-1,Count_Range)+1)*			
3	*Happy_New_Year*	Happy			Happy			
4		New			New			
5		Year			Year			
6					Happy			
7					New			
8					Year			
9					Happy			
10					New			
11					Year			
12					Happy			
13					New			
14					Year			
15					Happy			
16					New			
17					Year			
18					Happy			
19					New			
20					Year			
21								

We're good to go! Happy New Year 2023!

Until next time.

Key Strokes

Each newsletter, we'd like to introduce you to useful keystrokes you may or may not be aware of. This month, we look again at the **CTRL** and **SHIFT** keys, but this time combined with various special characters that Excel uses:

Keystroke	What it does
CTRL + ALT + F1	New Macro sheet
CTRL + ALT + F2	Open
CTRL + ALT + F3	Excel 2007 onwards: new Name
CTRL + ALT + F4	Close application
CTRL + ALT + F5	Refresh All
CTRL + ALT + F9	Full recalculation
CTRL + ALT + F12	Thai dictionary (!)
CTRL + ALT + Down Arrow	Intel chipset: invert screen (turn 180 degrees)
CTRL + ALT + Left Arrow	Intel chipset: turn screen +90 degrees, else move active cell to previous non-adjacent area within selection
CTRL + ALT + Right Arrow	Intel chipset: turn screen -90 degrees, else move active cell to next non-adjacent area within selection
CTRL + ALT + TAB	Indent
CTRL + ALT + Up Arrow	Intel chipset: display screen normally (zero [0] degrees)

There are *c.*550 keyboard shortcuts in Excel. For a comprehensive list, please download our Excel file at 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