

Sum Product

NEWSLETTER #140 - July 2024

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



Half time!

It's a game of two halves and we enter the second half of 2024 with new features and functions for Power BI and Excel. We have the **INFO** below...

If cash is your thing (and you're too rich if it's not), we consider how much we need for a rainy day. Can you make the right calculations? We highlight common mistakes in this month's newsletter.

We have the usual Beat the Boredom Challenge, Charts & Dashboards tips, Excel for Mac, Visual Basics, Power Pivot Principles, Power Query Pointers, Power BI Updates, Excel Updates, A to Z of Excel Functions and Keyboard Shortcuts too.

But that's not all: we decided it's time to add *another* regular article to the fold, **Over to AI**, where we figure if you can't beat it, join it and allow AI to wreak havoc by writing its own article this month. You can be the judge of what you think of it. One thing's for sure: we aren't terminate it...

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

Liam Bastick, Managing Director, SumProduct



Keeping up with Cash

Times remain tough presently and you have to keep a firm handle on your business. There is a wise old saying that reminds you to "control your cash before it controls you" – and these are sage words indeed in the current climate, as we have seen with several of our clients recently.

Therefore, this month, allow us to help you analyse your future cashflow requirements as follows.

Imagine you run the following (simple) business. As cashflow reduces, the need for more accurate and granular forecasts becomes greater. Therefore, you have forecast the following daily cash flow profile (in extract):

| Date | 1 Jan 25 | 2 Jan 25 | 3 Jan 25 | 4 Jan 25 | 5 Jan 25 | 6 Jan 25 | 7 Jan 25 | 8 Jan 25 | 9 Jan 25 | 10 Jan 25 |
|----------------------------------|----------|----------|----------|----------|----------|----------|----------|----------|----------|-----------|
| 1. Daily Cash Flows | | | | | | | | | | |
| Assumptions | | | | | | | | | | |
| Forecast Data | | | | | | | | | | |
| Incoming / (Outgoing) Cash Flows | \$ 807 | (\$807) | 700 | (\$543) | 242 | (\$360) | (\$522) | (\$903) | 427 | 57 |

Here, these cash flows have been predicted for 366 days (*say*).

Presently, you know how much cash you have in reserve. The question is, how much do you need to get you through these tough times? The sooner you identify the amount and the shortfall (if any), the sooner you can make contingency plans, such as reducing costs, organising loans, etc.

For any particular start date, the answer appears simple. All you need to do is keep a running total of the cashflows for each period, and calculate the minimum value. For example:

| | A | B | C | D | E | F | G | H | I | J | K | L | M | N | NJ | NK | NL | NM | NN | | | | | | | | | | | |
|----|--|----------|----------|----------|----------|----------|-----------|----------|---------------------|---|---|---|---|---|----|----|----|----|----|-----------------------|--|--|--|--|--|--|--|--|--|--|
| 1 | Maximum Cash Requirement | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 2 | SP - Maximum Cash Required.xlsm | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 3 | Navigator | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 4 | Error Checks: <input type="checkbox"/> | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 5 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 6 | Date | 1 Jan 25 | 2 Jan 25 | 3 Jan 25 | 4 Jan 25 | 5 Jan 25 | 31 Dec 25 | 1 Jan 26 | | | | | | | | | | | | | | | | | | | | | | |
| 8 | 1. Daily Cash Flows | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 9 | Assumptions | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 10 | Forecast Data | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 11 | Incoming / (Outgoing) Cash Flows | \$ 807 | (\$807) | 700 | (\$543) | 242 | (\$552) | 294 | | | | | | | | | | | | | | | | | | | | | | |
| 12 | Running Total | \$ 807 | - | 700 | 157 | 399 | (\$707) | (\$413) | =SUM(\$J\$15:J\$15) | | | | | | | | | | | | | | | | | | | | | |
| 13 | Max Requirement | \$ 5,499 | | | | | | | | | | | | | | | | | | =MIN(\$J\$17:SNK\$17) | | | | | | | | | | |
| 14 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 15 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 16 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 17 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 18 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 19 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 20 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |



Here, I have created a running total in row 17 using the simple cumulative summation

=SUM(\$J\$15:J\$15)

Then, I simply identify the minimum value in cell I19:

==MIN(\$J\$17:\$NK\$17)

(N.B. The minus sign in the formula is used to avoid confusion: negative shortfalls would confuse many.)

This shows the largest negative value will be \$5,499 – and this is the amount that I need (now) to ensure there is never a shortfall.

Wrong.

This is a common mistake made by businesses time and time again. It is commonly held that replacing the first period's value of \$807 with \$5,499 will prevent shortfalls. This is incorrect:

| Date | 1 Jan 25 | 2 Jan 25 | 3 Jan 25 | 4 Jan 25 | 5 Jan 25 | 31 Dec 25 | 1 Jan 26 |
|----------------------------------|----------|----------|----------|----------|----------|-----------|----------|
| Incoming / (Outgoing) Cash Flows | \$ 5,499 | (807) | 700 | (543) | 242 | (552) | 294 |
| Running Total | \$ 5,499 | 4,692 | 5,392 | 4,849 | 5,091 | 3,985 | 4,279 |
| Max Requirement | \$ 807 | | | | | | |

With \$5,499 used in the first period, there is still a requirement to fund \$807. \$807? Where have I seen that before?

| Date | 1 Jan 25 | 2 J |
|----------------------------------|----------|-----|
| Incoming / (Outgoing) Cash Flows | \$ 807 | |
| Running Total | \$ 807 | |

This amount was the amount originally in the first period. That makes sense. The shortfall already included the amount in the first period. Our calculation simply identifies the *additional* amount required. This would be \$5,499 + \$807 = \$6,306.

This can be calculated as follows:

| Date | 1 Jan 21 | 2 Jan 21 | 3 Jan 21 | 4 Jan 21 | 5 Jan 21 | 31 Dec 21 | 1 Jan 22 |
|----------------------------------|----------|----------|----------|----------|----------|-----------|----------|
| Incoming / (Outgoing) Cash Flows | \$ 807 | (807) | 700 | (543) | 242 | (552) | 294 |
| Running Total | \$ (807) | (107) | (650) | (408) | (1,514) | (1,220) | |
| Max Requirement | \$ 6,306 | | | | | | |

==MIN(\$K\$17:\$NK\$17)

We *almost* had the calculation correct. The running total needs to exclude the current period and start from the next one and then be computed for all periods until the end of the forecast period.

This is all we need to do if we simply want to calculate the requirement for the first period. But what if we want to “roll” our forecast and identify what we need on the 17 August (say) instead? You could repeat this exercise and simply have a similar calculation on each row for each period, similar in appearance to a depreciation schedule. That would

require 366 rows and at least 67,161 formulae!

Surely there is a better way? Yes there is – and stop calling me Shirley.

Let me add another complication too: it might be you only want to see what the maximum cash is needed for a certain number of days (e.g. circumstances may change, you may have other alternatives). I will add in a selector, although I will set it originally to 366, so that we may compare with the original illustration (above).

Then, consider the following calculations:

| | A | B | C | D | E | F | G | H | I | J | K | L | M | N | NJ | NK | NL | NM | NN | NO | NP | NQ | NR | NS | NT |
|----|--|----------|--------------------------|--------------------------|--------------------------|--------------------------|--------------------------|--------------------------|--------------------------|--|---|-----------|---|----------|----|----|----|----|----|----|----|----|----|----|----|
| 1 | Maximum Cash Requirement | | | | | | | | | | | | | | | | | | | | | | | | |
| 2 | SP Maximum Cash Required.xlsm | | | | | | | | | | | | | | | | | | | | | | | | |
| 3 | Worksheet | | | | | | | | | | | | | | | | | | | | | | | | |
| 4 | Error Checks: <input type="checkbox"/> | | | | | | | | | | | | | | | | | | | | | | | | |
| 5 | | | | | | | | | | | | | | | | | | | | | | | | | |
| 6 | Date | 1 Jan 25 | | 2 Jan 25 | | 3 Jan 25 | | 4 Jan 25 | | 5 Jan 25 | | 31 Dec 25 | | 1 Jan 26 | | | | | | | | | | | |
| 7 | Counter | 1 | | 2 | | 3 | | 4 | | 5 | | 365 | | 366 | | | | | | | | | | | |
| 8 | | | | | | | | | | | | | | | | | | | | | | | | | |
| 9 | 1. Daily Cash Flows | | | | | | | | | | | | | | | | | | | | | | | | |
| 10 | Assumptions | | | | | | | | | | | | | | | | | | | | | | | | |
| 11 | Forecast Data | | | | | | | | | | | | | | | | | | | | | | | | |
| 12 | | | | | | | | | | | | | | | | | | | | | | | | | |
| 13 | Incoming / (Outgoing) Cash Flows | | | | | | | | | | | | | | | | | | | | | | | | |
| 14 | | \$ | 807 | (807) | 700 | (543) | 242 | (556) | 284 | | | | | | | | | | | | | | | | |
| 15 | | | | | | | | | | | | | | | | | | | | | | | | | |
| 16 | | | | | | | | | | | | | | | | | | | | | | | | | |
| 17 | | | | | | | | | | | | | | | | | | | | | | | | | |
| 18 | 2. Calculations | | | | | | | | | | | | | | | | | | | | | | | | |
| 19 | Calculations | | | | | | | | | | | | | | | | | | | | | | | | |
| 20 | Volatile Solution | | | | | | | | | | | | | | | | | | | | | | | | |
| 21 | | | | | | | | | | | | | | | | | | | | | | | | | |
| 22 | Number of Look Forward Periods | # | 366 | | | | | | | | | | | | | | | | | | | | | | |
| 23 | Number of Periods | # | 366 | | | | | | | | | | | | | | | | | | | | | | |
| 24 | Reverse Counter | # | 366 | 365 | 364 | 363 | 362 | 2 | 1 | =MAX(\$I7:\$I) | | | | | | | | | | | | | | | |
| 25 | Cumulative Deficit / (Surplus) | \$ | (807) | - | (700) | (157) | (399) | 707 | 413 | =SUM(\$I5:\$I5) | | | | | | | | | | | | | | | |
| 26 | No. of Periods for Maximum Cashflow | # | 141 | 140 | 139 | 138 | 137 | - | - | =MAX(MAXIFS(OFFSET(\$J\$7,,,J\$28),OFFSET(\$J\$30,,,J\$28),MAX(OFFSET(\$J\$30,,,J\$28)))-J\$7,0) | | | | | | | | | | | | | | | |
| 27 | Min Amount Required This Period | \$ | 6,306 | 5,499 | 6,199 | 5,656 | 5,898 | - | - | =SUM(FLU\$32:OFFSETS(\$I5,,,J\$32)) | | | | | | | | | | | | | | | |
| 28 | CHECK: Confirms this would be zero | \$ | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | =SUM(FLU\$32:OFFSETS(\$I5,,,J\$32))-J\$34 | | | | | | | | | | | | | | | |
| 29 | | | | | | | | | | | | | | | | | | | | | | | | | |
| 30 | | | | | | | | | | | | | | | | | | | | | | | | | |
| 31 | | | | | | | | | | | | | | | | | | | | | | | | | |
| 32 | | | | | | | | | | | | | | | | | | | | | | | | | |
| 33 | | | | | | | | | | | | | | | | | | | | | | | | | |
| 34 | | | | | | | | | | | | | | | | | | | | | | | | | |
| 35 | | | | | | | | | | | | | | | | | | | | | | | | | |
| 36 | | | | | | | | | | | | | | | | | | | | | | | | | |
| 37 | | | | | | | | | | | | | | | | | | | | | | | | | |

I can calculate the maximum cash required for any period using just five formulae (and even then, arguably you could condense some of these).

The first formula is situated in cell **I26** (above). This calculates how many periods have been forecast:

$$=MAX($7:$7)$$

In general, you should never create a formula that refers to an entire row or column as they can use up resources, leading to lack of memory issues and poor calculation times. However, this is a simple formula, and allows for additional periods to be added without issue. It is only calculated once. This is good practice: try never to calculate resource hungry expressions more than once.

The formula in row 26 then provides the reverse counter (like a countdown, but considering the stipulated look forward period). For example, in cell **J28**, this is given by the formula

$$=MIN($I24,$I26-J$7+1)$$

For a 366-period model with a 366-period look forward, this provides a countdown from 366 (first period) to one (1) (final period).

The third formula is the cumulative sum, e.g. in cell **J30**, it is given by

$$=-SUM(J15:J15)$$

This formula is negated so that deficits are positive and surpluses are negative. I find it easier for people to understand calculations where the results are positive.

What we want to do now if from a given start position in row 28, we want to find the largest value, as this will represent the relative lower bound of the cashflow. It is relative because this cumulative figure may include values that are past (sunk): it doesn't matter, as the lowest value remaining will still be the lower bound, once adjusted.

This leads to an horrific formula, e.g. in cell **J32**, it is

$$=MAX(MAXIFS(OFFSET(J7,,,J$28),OFFSET($J$30,,,J$28),MAX(OFFSET(J30,,,J$28)))-J$7,0)$$

Any questions? Oh wait, I see there are.

The **MAX** function is fairly straightforward (simply taking the maximum of its arguments), but the other two functions may require further clarification:

- the **OFFSET(reference, rows, columns, [height], [width])** function has the **height** and **width** arguments in square brackets, as they may be omitted from the formula, as usually **OFFSET** deals with moving so many **rows** down and so many **columns** to the right of the **reference**.

However, in this instance I am trying to define a range that, although only one row deep, is so many columns wide. Therefore, aside from the **reference** and the **width** all of the other arguments may take their default values (zero [0], zero [0] and one [1] respectively).

Therefore, the expression **OFFSET(\$J\$7,,,J\$28)** means take a range of cells in row 7, starting at **J7** with a width of **J28** columns. Since **J28** is 366 given the restrictions on look forward and the model duration, this means take the range of 366 cells in row 7, starting in column **J**, i.e. **J7:NK7**. Similarly, **OFFSET(\$J\$30,,,J\$28)** defines the range **J30:NK30**. The beauty about the **OFFSET** function is that if the number of periods varies, this range will update automatically.

This slowly contracting range is used rather than a fixed width that changes references (e.g. **J30:NK30** in the first cell, **K30:NL30** in the second, ...) so that calculations are not any larger than they need to be. An **OFFSET** formula is required as the **width** may vary based upon the Number of Look Forward Periods (cell **I24**) selected

- MAXIFS(max_range, criterion_range1, criterion1, [criterion_range2, criterion2], ...)** returns the maximum value among cells specified by a given set of conditions or criteria, where:
 - max_range** is the actual range of cells in which the maximum is to be determined
 - criterion_range1** is the set of cells to evaluate with the criterion specified
 - criterion1** is the criterion in the form of a number, expression or text that defines which cells will be evaluated as a maximum
 - criterion_range2** (onwards) and **criterion2** (onwards) are the additional ranges and their associated criteria. 126 range / criterion pairs may be specified. All ranges must have the same dimensions otherwise the function returns an **#VALUE!** error.

The previous formula reduces to

$$=MAX(MAXIFS(J\$7:NK\$7,J\$30:NK\$30,MAX(J\$30:NK\$30))-J\$7,0)$$

MAXIFS(J\$7:NK\$7,J\$30:NK\$30,MAX(J\$30:NK\$30)) returns the counter in row 7 that identifies the largest cumulative deficit in row 30. This occurs in period 142 for cell **J32**. This means the formula is now

$$=MAX(142-J\$7,0)$$

Since **J\$7** represents the first period (1), the resulting value is 141. This means that the largest deficit is recorded when you sum the first 142 values in row 15 – but we subtract off the first period, so we do not repeat our previous mistake.

The formula in **J34** then evaluates this summation:

$$=-SUM(IF(J$32,OFFSET(K$15,,,,J$32),))$$

(again, the negative sign to prevent confusion). This formula is the same as

$$=IF((J$32=0,0,SUM(K$15:EU$15))$$

where **K\$15:EU\$15** is the range of 141 cells across starting in cell **K15**. This gives the same value we calculated earlier, *i.e.* \$6,306. However, unlike the previous method, the value in cell **K34** (\$5,499) shows me the value that should be in Period 2 to ensure no future shortfalls, and similarly the value in cell **FU34** (period 168) advises me that on 17 June, I will require \$3,946 in this period to avoid a cashflow shortfall based upon future projections from this point, with a minimum value of zero (nil) occurring 54 days later on 10 August (period 222).

| | A | B | C | D | E | F | G | H | I | FT | FU | FV | FW | FX | FY | FZ | | |
|----|---|---|---|---|---|---|---|---|---|-----------|-----------|-----------|-----------|-----------|-----------|-----------|--|--|
| 1 | Maximum Cash Requirement | | | | | | | | | | | | | | | | | |
| 2 | SP Maximum Cash Required.xlsm | | | | | | | | | | | | | | | | | |
| 3 | Navigator | | | | | | | | | | | | | | | | | |
| 4 | Error Checks: <input checked="" type="checkbox"/> | | | | | | | | | | | | | | | | | |
| 5 | | | | | | | | | | | | | | | | | | |
| 6 | Date | | | | | | | | | 16 Jun 25 | 17 Jun 25 | 18 Jun 25 | 19 Jun 25 | 20 Jun 25 | 21 Jun 25 | 22 Jun 25 | | |
| 7 | Counter | | | | | | | | | 167 | 168 | 169 | 170 | 171 | 172 | 173 | | |
| 19 | Calculations | | | | | | | | | | | | | | | | | |
| 20 | Volatile Solution | | | | | | | | | | | | | | | | | |
| 21 | | | | | | | | | | | | | | | | | | |
| 22 | | | | | | | | | | | | | | | | | | |
| 23 | | | | | | | | | | | | | | | | | | |
| 24 | | | | | | | | | | | | | | | | | | |
| 25 | | | | | | | | | | | | | | | | | | |
| 26 | | | | | | | | | | | | | | | | | | |
| 27 | | | | | | | | | | | | | | | | | | |
| 28 | | | | | | | | | | | | | | | | | | |
| 29 | | | | | | | | | | | | | | | | | | |
| 30 | | | | | | | | | | | | | | | | | | |
| 31 | | | | | | | | | | | | | | | | | | |
| 32 | | | | | | | | | | | | | | | | | | |
| 33 | | | | | | | | | | | | | | | | | | |
| 34 | | | | | | | | | | | | | | | | | | |
| 35 | | | | | | | | | | | | | | | | | | |
| 36 | | | | | | | | | | | | | | | | | | |
| 37 | | | | | | | | | | | | | | | | | | |

The formula in row 36 (e.g. cell **J36**)

$$=SUM(IF(J$32,OFFSET(K$15,,,,J$32),),J$34)$$

simply confirms that the value cited in row 34 is correct.

Regular readers that the **OFFSET** function both fools Excel's auditing tools (e.g. Trace Dependents) and is volatile (*i.e.* a calculation that causes unnecessary recalculations of the formula in the cell where it resides every time Excel recalculates). If calculation is speed / memory usage is important to you (e.g. where this calculation may form part of a very large Excel model), then you may wish to replace **OFFSET** with its non-volatile counterpart, **INDEX**:

| | A | B | C | D | E | F | G | H | I | J | K | L | M | N | NU | NK | NL | NM | NN | NO | NP | NQ | NR | NS | NT | NJ | NV | NW | NX | NY | |
|----|---|---|---|---|---|---|---|---|---|---|---|---|---|---|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|--|
| 1 | Maximum Cash Requirement | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 2 | SP Maximum Cash Required.xlsm | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 3 | Navigator | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 4 | Error Checks: <input checked="" type="checkbox"/> | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 5 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 6 | Date | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 7 | Counter | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 9 | 1. Daily Cash Flows | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 10 | Assumptions | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 11 | Forecast Data | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 12 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 13 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 14 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 15 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 16 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 17 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 18 | 2. Calculations | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 19 | Calculations | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 20 | Non-Volatile Solution | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 21 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 22 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 23 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 24 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 25 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 26 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 27 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 28 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 29 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 30 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 31 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 32 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 33 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 34 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

I don't explain this here, but hopefully, you get the idea.

Word to the Wise

Some readers may view the above as a little basic. It's true; it is. You can add greater complexity by:

- changing the number of days looking forward considering the time value of money (present values)
- allowing for surpluses to earn interest and deficits to incur interest payments and other fees
- including probability weightings to approximate the likelihood of the cash flows occurring
- incorporating direct and indirect tax cash flow consequences.

Whilst it is admittedly simplistic, it is greater analysis than many businesses do. It is a start that you can build upon. Hopefully, you find this useful and may put it to good use.

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...*

Sometimes, you may need to deal with numbers in text format. As you may know, there are various ways to convert them into a numerical data type. Believe us, this challenge is a bit trickier than the usual problem, but you may come across it some day when using Excel.

One of our clients presented us with a problem similar to the following. They provided us with a list of numbers copied from their management information system to Excel. It was in a very strange text format that needed to be converted into numbers.

We have made up a similar list for this challenge that you can download here:

<https://view.officeapps.live.com/op/view.aspx?src=https%3A%2F%2Fwww.sumproduct.com%2Fassets%2Fblog-pictures%2F2021%2Ffff-mmm%2Foct%2Ffff%2Fsp-number-conversion---question.xlsx&wdOrigin=BROWSELINK>

This month's challenge is to write a **formula** that may be dragged down to convert the list of numbers in the associated file into a list of "real" numbers, similar to the ones on the right in the image below:

| Number | Solution |
|--------|----------|
| 0009 | 9 |
| 0011 | 11 |
| 0012 | 12 |
| 0013 | 13 |
| 0017 | 17 |
| 0023 | 23 |
| 0025 | 25 |
| 0027 | 27 |
| 0028 | 28 |
| 0032 | 32 |
| 0039 | 39 |
| 0045 | 45 |
| 0047 | 47 |

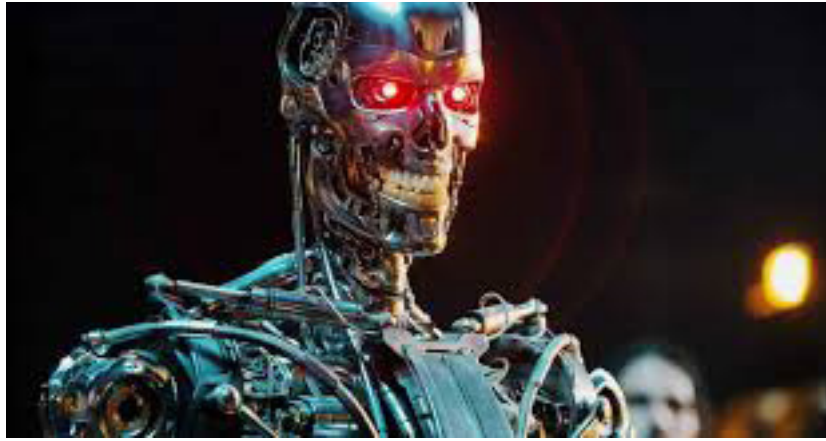
As always, there are some requirements:

- there should be no "helper" column
- this is a formula challenge: no Power Query / Get & Transform, Text to Columns or Flash Fill (although you may wish to test these options)
- the formula needs to be dynamic enough so that if we change from four digits to another length, the formula must still work.

We are sure that it may take you a little bit of time (*i.e.* at least more than 3 minutes) to figure out a way to clean the list!

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

Over to AI



My nose is a little out of joint this month as on two occasions, different editors asked if Artificial Intelligence had “assisted” me in writing various articles. But it got me thinking: if you can’t beat them (apparently, you’re not allowed to hit editors), join them. So, I applied this to AI. Therefore, dear reader, allow me to introduce a new section to this newsletter

[that’s all we need – more articles to edit – Ed.]: “Over to AI”.

Here, each month, I shall ask AI a question and see what it comes back with. Other than correct for UK English, I plan to leave its response intact. See what you think!

Provide a Brief Overview of Some of the Most Popular and Useful Features in Microsoft Excel

Microsoft Excel is a powerful and versatile spreadsheet software that can be used for various purposes, such as data analysis, reporting, budgeting, forecasting and more. Excel has many features and functions that can

help users perform complex calculations, manipulate data, create charts and graphs and automate tasks. Here, we will explore some of the hot topics in Excel that are widely used and appreciated by Excel users.

PIVOTTABLES AND PIVOTCHARTS

Microsoft Excel is a powerful and versatile spreadsheet software that can be used for various purposes, such as data analysis, reporting, budgeting, forecasting and more. Excel has many features and functions that can

help users perform complex calculations, manipulate data, create charts and graphs and automate tasks. Here, we will explore some of the hot topics in Excel that are widely used and appreciated by Excel users.

VLOOKUP AND HLOOKUP

VLOOKUP and **HLOOKUP** are functions that can help users find and retrieve data from a table or a range based on a given value. **VLOOKUP** stands for vertical lookup, and it can search for a value in the first column of a table or a range and return a corresponding value from another column. **HLOOKUP** stands for horizontal lookup, and it can search for

a value in the first row of a table or a range and return a corresponding value from another row. **VLOOKUP** and **HLOOKUP** can be useful for finding information, such as prices, names, or codes from a large database or a list.

CONDITIONAL FORMATTING

Conditional formatting is a feature that can help users highlight or format cells, rows or columns based upon certain rules or criteria. Conditional formatting can apply different colours, fonts, icons or data bars to

cells that meet specific conditions, such as values, formulae or dates. Conditional formatting can help users visualise data, identify trends, spot errors or emphasise important information.

MACROS AND VBA

Macros and VBA are tools that can help users automate tasks and create custom functions in Excel. Macros are sequences of commands or actions that can be recorded and run by a single click or a keyboard shortcut. VBA stands for Visual Basic for Applications, and it is a programming language

that can be used to write macros or create user-defined functions, forms or dialogs. Macros and VBA can help users save time, reduce error, and enhance functionality in Excel.

What do you think? Do you agree? (You know my thoughts on **VLOOKUP**...) Send us your comments and ideas for questions to contact@sumproduct.com.

Excel for Mac

With Steve Kraynak joining the team, we thought we would exploit his knowledge and recant all about Microsoft Excel for Mac. Each month, we'll cover a different topic to help you understand how Excel for Mac

is different than Excel for Windows. This month, we summarise a few worksheet functions that either aren't supported on Mac or work slightly differently from their counterparts on a PC.

Through the years, a few functions have been created for Excel on Windows that make use of special features of the Windows operating system. These, of course, don't work when those special features aren't available, as when running on macOS.

These are the functions:

- **ENCODEURL:** this function is used to replace certain non-alphanumeric characters in a string of text with the percentage symbol (%) and a hexadecimal number. The example below shows how this works for the string <https://www.sumproduct.com/>, which gets converted to "https%3A%2F%2Fwww.sumproduct.com%2F":

| | A | B | C |
|---|---|-----------------------------------|-------------------------------------|
| 1 | Website | | |
| 2 | https://www.sumproduct.com/ | | |
| 3 | | | |
| 4 | | | |
| 5 | Formula | Description | Result |
| 6 | =ENCODEURL(A2) | Converts to a URL-encoded string. | https%3A%2F%2Fwww.sumproduct.com%2F |
| 7 | | | |

The screen shot below shows that the formula =ENCODEURL(A2) returns an #NAME? error, indicating that the function is not supported on Mac

| | A | B | C |
|---|--|-----------------------------------|--------|
| 1 | Website | | |
| 2 | https://www.sumproduct.com | | |
| 3 | | | |
| 4 | Formula | Description | Result |
| 5 | =ENCODEURL(A2) | Converts to a URL-encoded string. | #NAME? |
| 6 | | | |

- **FILTERXML:** this returns specific data from XML content by using the specified xpath. This function relies on XML processing capability of the Windows OS, so it will also return the error #NAME? error on Mac
- **WEBSERVICE:** this returns data from a web service on the Internet or Intranet. It's also not supported on Mac and will result with another #NAME? error. The data returned would typically be further processed using the FILTERXML function, which as above, is not supported on Mac.

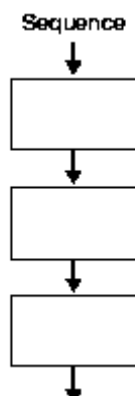
We'll continue next month...

Visual Basics

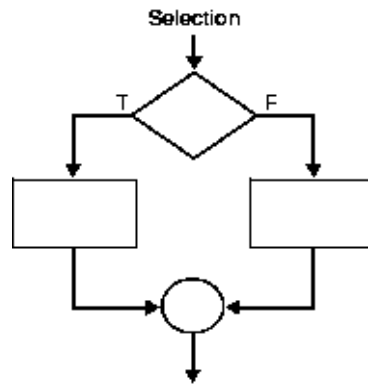
We thought we'd run an elementary series going through the rudiments of Visual Basic for Applications (VBA) as a springboard for newer users. This month, we take continue our discussion on control structures.

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

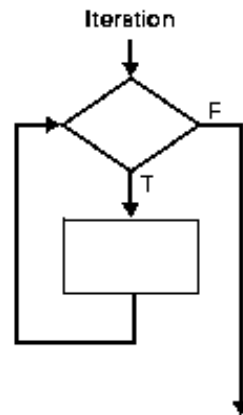
1. **Sequential:** Sequential execution is where each statement in the source code will be executed one by one in a sequential order. This is the default mode of execution



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



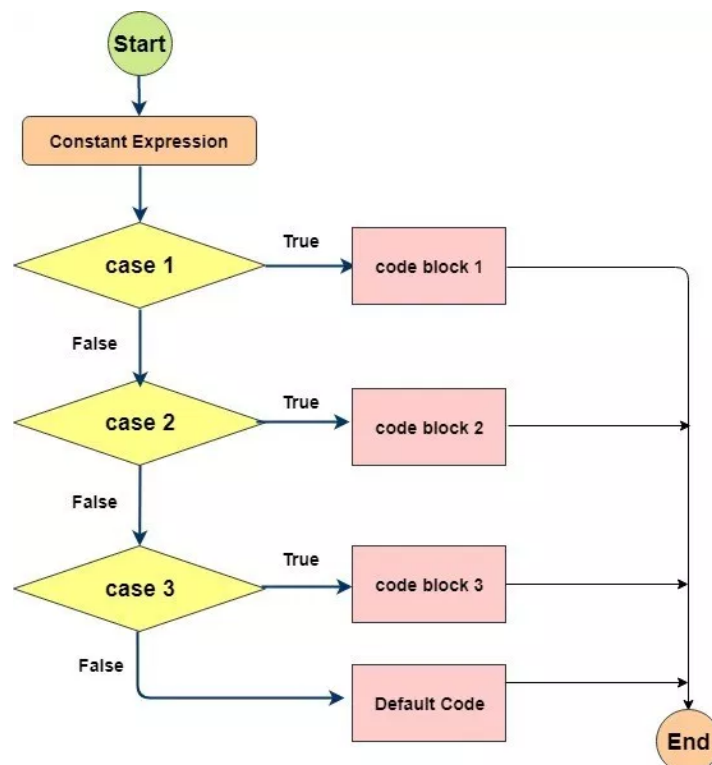
3. **Iteration:** The iterative control structures are used for repetitively executing a block of code multiple times



This month we will look at selection control structures. Here are several examples.

SELECT CASE

SELECT CASE is the VBA equivalent to a switch control structure. A switch statement allows a variable to be tested for equality against a list of values. Each value is called a case, and the variable being “switched” on is checked for each case:



The **SWITCH** statement is used as follows:

```
Select [ Case ] testexpression
    [ Case expressionlist
        [ statements ] ]
    [ Case Else
        [ elsestatements ] ]
End Select
```

Let's look at a coding example:

```
Sub SWITCHStatement()
    Dim myNumber As Integer
    myNumber = 1
    Select Case myNumber
        Case 0
            Debug.Print "Zero the hero!"
        Case 1
            Debug.Print "Number one!"
        Case 2
            Debug.Print "Number two!"
        Case 3
            Debug.Print "Number three!"
        Case Else
            Debug.Print "Not a podium finish"
    End Select
End Sub
```

IS

SELECT CASE allows the use of the **IS** keyword to compare values. If the variable can use comparison operators, then **IS** is used before the operator. For example:

```
Sub SwitchISStatement()
    Dim myNumber As Integer
    myNumber = 1
    Select Case myNumber
        Case Is < 0
            Debug.Print "Negative"
        Case Is < 10
            Debug.Print "Single Digit Positive Integer"
        Case Is < 100
            Debug.Print "Double Digit Positive Integer"
        Case Else
            Debug.Print "Large number!"
    End Select
End Sub
```

Note: upon typing an operator (>, < or =) after the keyword **CASE**, the VBA editor will automatically correct the statement and place the **IS** keyword after **CASE**.

TO

If the variable assessed can lie within a range, the **TO** keyword is called upon to denote the range. For example:

```
Sub SwitchTOSTatement()  
    Dim myNumber As Integer  
    myNumber = 1  
    Select Case myNumber  
        Case Is < 0  
            Debug.Print "Negative"  
        Case 0 To 9  
            Debug.Print "Single Digit Positive Integer"  
        Case 10 To 99  
            Debug.Print "Double Digit Positive Integer"  
        Case Else  
            Debug.Print "Large number!"  
    End Select  
End Sub
```

Combination

Multiple expressions can be used in a single **CASE** statement for optimum efficiency. Another example:

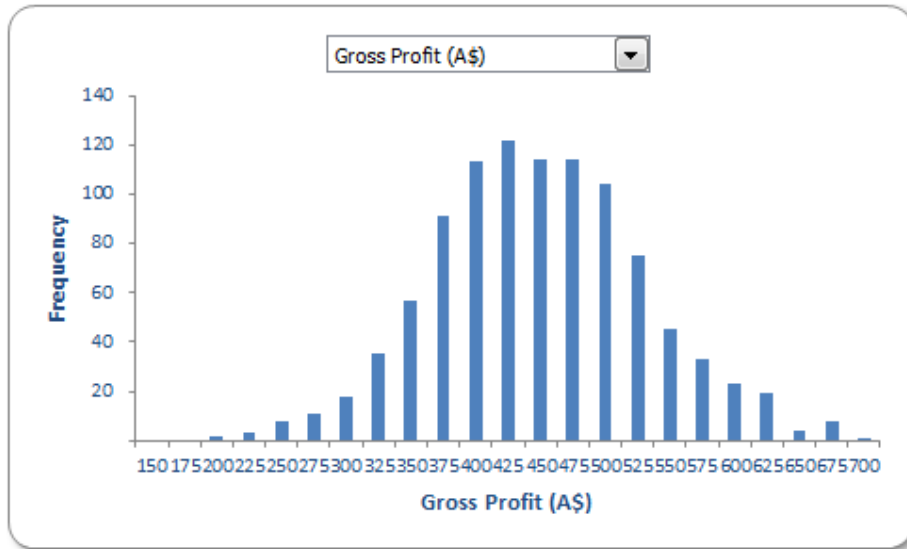
```
Sub SwitchCombinationStatement()  
    Dim myNumber As Integer  
    myNumber = 1  
    Select Case myNumber  
        Case Is < 0  
            Debug.Print "Negative"  
        Case 1, 2, 3, 4, 5 To 9  
            Debug.Print "Single Digit Positive Integer"  
        Case 10 To 50, 51, 52, 53, 54, 55 To 99  
            Debug.Print "Double Digit Positive Integer"  
        Case Else  
            Debug.Print "Large number!"  
    End Select  
End Sub
```

More next time.

Charts and Dashboards

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

Quite a while ago now, on SumProduct's Thought (<https://www.sumproduct.com/thought>) page, we discussed simulations (<https://www.sumproduct.com/thought/simulation-stimulation>), where we applied simulation analysis in Excel and created several charts, e.g.

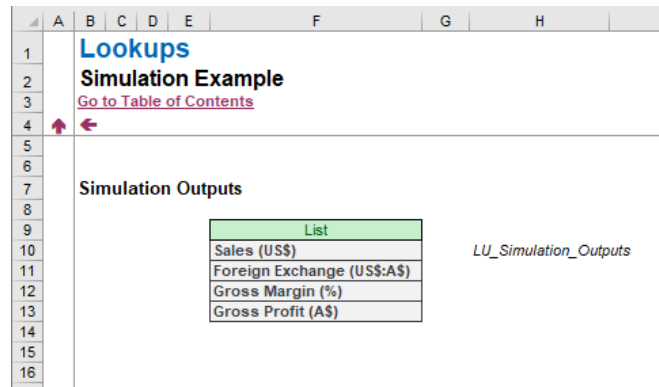


This time, we will go through the steps of creating one. The following Excel file can be used to follow along:

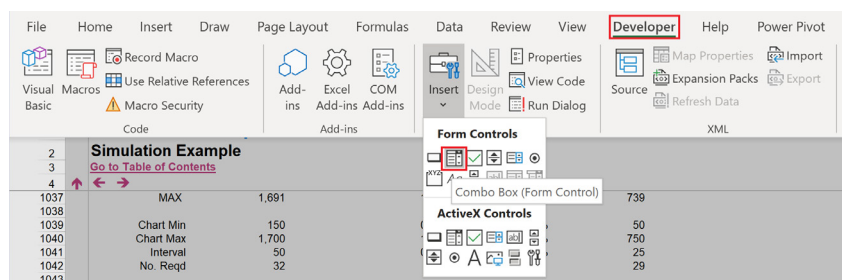
<https://view.officeapps.live.com/op/view.aspx?src=https%3A%2F%2Fwww.sumproduct.com%2Fassets%2Fblog-pictures%2F2020%2Fcharts-and-dashboards%2F048%2Fsimulation-example.xlsm&wdOrigin=BROWSELINK>

The above column chart has dynamic series and axes defined as 'Chart_Stratification' and 'Chart_Count', which change by the choice from the drop-down list. It only requires a quick collation step to summarise the outputs graphically.

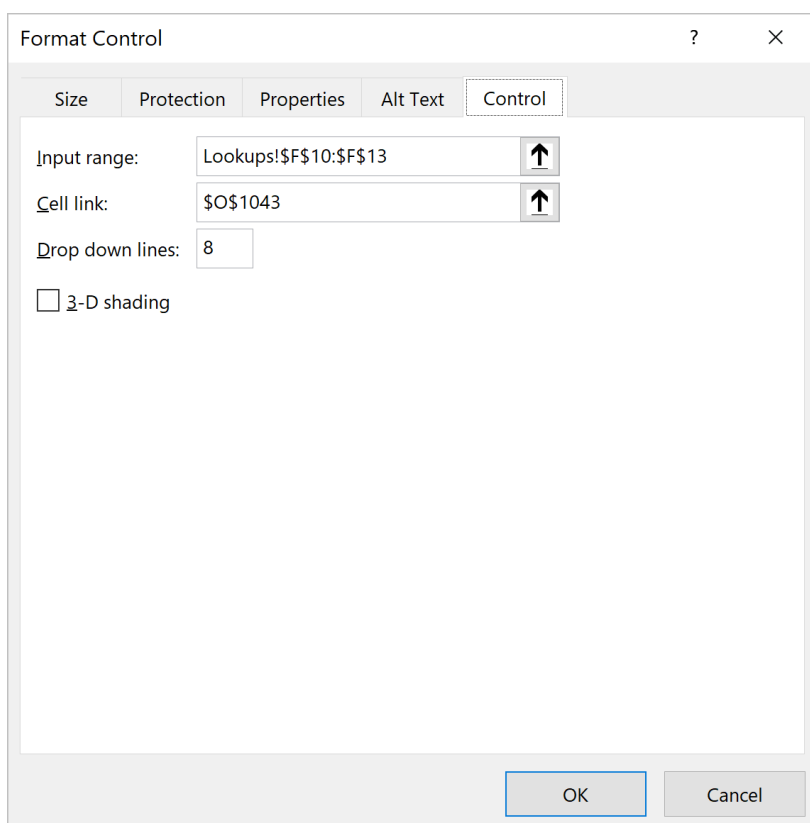
First, in the **Lookups** sheet, define the 'LU_Simulation_Outputs' range (cell F10:F13).



Now, to create the drop-down list, navigate to the Developer tab on the Ribbon, from Insert, choose 'Combo Box (Form Control)' and draw a drop-down list box.



Then, right-click on the drop-down list box and choose 'Format Control'. Link the 'Input range' and 'Cell link'. The number of 'Drop down lines' should be eight (8).



The cell **O1043** is named 'DD_Simulations_Outputs', which is the indication of the drop-down choice.

The formula in cell **O1045** is

=INDEX(LU_Simulation_Outputs,DD_Simulations_Outputs)

The **OFFSET** function is used to get the **Stratification** and **Count**, respectively:

=OFFSET(D1047,,2*DD_Simulations_Outputs)

and

=OFFSET(E1047,,2*DD_Simulations_Outputs)

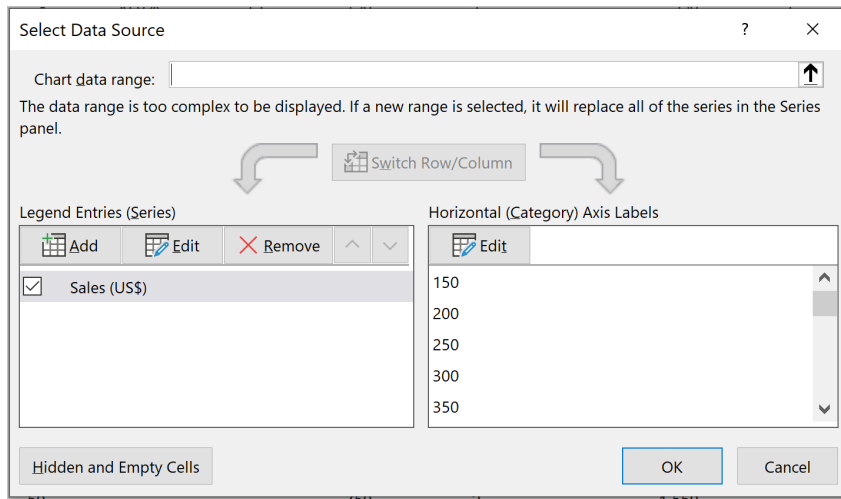
| | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P |
|------|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | Simulation Example | | | | | | | | | | | | | | | |
| 2 | Simulation Example | | | | | | | | | | | | | | | |
| 3 | Go to Table of Contents | | | | | | | | | | | | | | | |
| 4 | ← → | | | | | | | | | | | | | | | |
| 1043 | | | | | | | | | | | | | | | | 1 |
| 1044 | | | | | | | | | | | | | | | | |
| 1045 | | | | | | | | | | | | | | | | |
| 1046 | | | | | | | | | | | | | | | | |
| 1047 | | | | | | | | | | | | | | | | |
| 1048 | | | | | | | | | | | | | | | | |
| 1049 | | | | | | | | | | | | | | | | |
| 1050 | | | | | | | | | | | | | | | | |
| 1051 | | | | | | | | | | | | | | | | |
| 1052 | | | | | | | | | | | | | | | | |
| 1053 | | | | | | | | | | | | | | | | |
| 1054 | | | | | | | | | | | | | | | | |
| 1055 | | | | | | | | | | | | | | | | |
| 1056 | | | | | | | | | | | | | | | | |
| 1057 | | | | | | | | | | | | | | | | |
| 1058 | | | | | | | | | | | | | | | | |
| 1059 | | | | | | | | | | | | | | | | |
| 1060 | | | | | | | | | | | | | | | | |
| 1061 | | | | | | | | | | | | | | | | |
| 1062 | | | | | | | | | | | | | | | | |
| 1063 | | | | | | | | | | | | | | | | |
| 1064 | | | | | | | | | | | | | | | | |
| 1065 | | | | | | | | | | | | | | | | |
| 1066 | | | | | | | | | | | | | | | | |
| 1067 | | | | | | | | | | | | | | | | |
| 1068 | | | | | | | | | | | | | | | | |
| 1069 | | | | | | | | | | | | | | | | |
| 1070 | | | | | | | | | | | | | | | | |
| 1071 | | | | | | | | | | | | | | | | |
| 1072 | | | | | | | | | | | | | | | | |

Next, define the dynamic range name (using the **OFFSET** function):

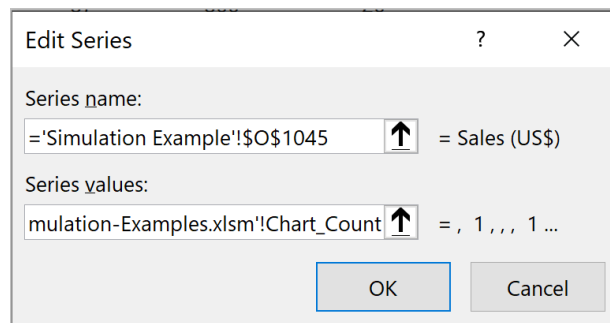
Chart_Stratification = OFFSET('Simulation Example'!\$O\$1047,,,OFFSET('Simulation Example'!\$E\$1042,'Simulation Example'!\$O\$1043),1)

Chart_Count = OFFSET('Simulation Example'!\$P\$1047,,,OFFSET('Simulation Example'!\$E\$1042,'Simulation Example'!\$O\$1043),1)

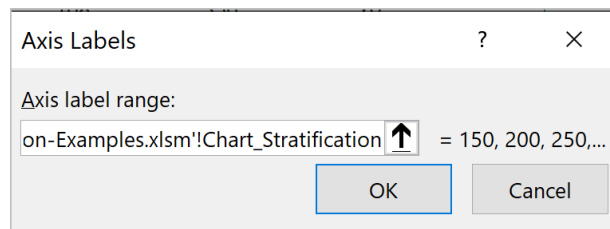
Select the value in the **Stratification** and **Count** columns to create a Column chart, then right-click on the chart and choose 'Select Data...'.



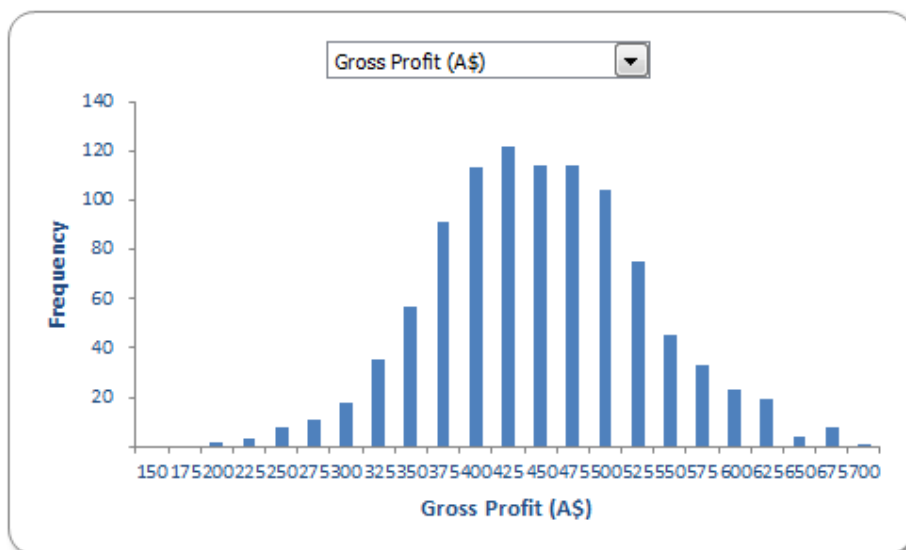
Let the 'Series values' take the dynamic 'Chart_Count' range



and 'Axis label range' be the 'Chart_Stratification' range. Please note that the sheet name should also be included.



Last but not least, make the axis label dynamic by point it to the cell 'Simulation Example!\$O\$1045'.



More next time.

Power Pivot Principles

We continue our series on the Excel COM add-in, Power Pivot. This month, we show you how to create a dynamic ranking measure that changes based upon selection filters.

This month, we are going to create a measure that will dynamically rank product prices based upon a selection filter. We are going to use the **ALLSELECTED** and **CALCULATE** functions. You can read about the **ALLSELECTED** function [here](#), and the **CALCULATE** function [here](#).

For this example, we are going to use the following dataset:

| ID | Product Type | Category | Price |
|----|---|--------------------|--------|
| 1 | Steel Baking Tray | Baking | 20.00 |
| 2 | Premium 40cm Pan | Pots & Pans | 120.00 |
| 3 | Dinner Set | Cutlery | 35.00 |
| 4 | Waffle Maker | Kitchen Appliances | 125.00 |
| 5 | Rice Cooker Premium | Kitchen Appliances | 160.00 |
| 6 | 5,000 Piece Napkin | Expendable | 20.00 |
| 7 | 10,000 Piece Napkins | Expendable | 30.00 |
| 8 | Rice Cooker | Kitchen Appliances | 85.00 |
| 9 | Dish Washing Capsules 100 pack | Expendable | 35.00 |
| 10 | Slow Cooker | Kitchen Appliances | 140.00 |
| 11 | 6 Slice Toaster (Imported) | Kitchen Appliances | 120.00 |
| 12 | Water Filter | Kitchen Appliances | 50.00 |
| 13 | Small Baking Dish | Cutlery | 12.00 |
| 14 | Filter Water Bottle | Kitchen Appliances | 15.00 |
| 15 | Chef Knife | Cutlery | 90.00 |
| 16 | Enviro-friendly Dishwashing Capsules 200 Pack | Expendable | 20.00 |
| 17 | Can Opener | Cutlery | 3.00 |
| 18 | Fine China Dining Set | Cutlery | 80.00 |
| 19 | Mortar & Pestle | Pots & Pans | 20.00 |
| 20 | Stainless Steel Pot | Pots & Pans | 60.00 |
| 21 | Kitchen Knife Set | Cutlery | 80.00 |

You might recognise this Table, as it is the same dataset used in last month's newsletter, with a new column 'Category' added. Let's imagine that we want to create a PivotTable that will display the products and their prices based on the category.

The screenshot shows an Excel spreadsheet with columns A through H. Column B contains 'Product Type' and column C contains 'Price'. The data is as follows:

| Product Type | Price |
|-----------------------|---------|
| Can Opener | \$ 3.00 |
| Chef Knife | \$90.00 |
| Dinner Set | \$35.00 |
| Fine China Dining Set | \$80.00 |
| Kitchen Knife Set | \$80.00 |
| Small Baking Dish | \$12.00 |

The PivotTable Fields task pane is open on the right. It shows the data source 'ProductListKitchen' with the following fields:

- ID
- Product Type
- Category
- Price
- Price Rank

The 'Product Type' field is placed in the Rows area and 'Price' is placed in the Values area.

Now, we wish to have another column that will rank the products based on their prices. We have created a ranking column previously; let's see if this will work here.

As a recap, we created a ranking column in PowerPivot with the following **DAX** code here:

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

| ID | Product Type | Price | Category | Price Rank |
|----|--------------------------------|-------|---------------|------------|
| 1 | Steel Baking Tray | 20 | Baking | 4 |
| 2 | Premium 40cm Pan | 120 | Pots & Pans | 17 |
| 3 | Dinner Set | 35 | Cutlery | 9 |
| 4 | Waffle Maker | 125 | Kitchen Ap... | 19 |
| 5 | Rice Cooker Premium | 160 | Kitchen Ap... | 21 |
| 6 | 5,000 Piece Napkin | 20 | Expendable | 4 |
| 7 | 10,000 Piece Napkins | 30 | Expendable | 8 |
| 8 | Rice Cooker | 85 | Kitchen Ap... | 15 |
| 9 | Dish Washing Capsules 100 p... | 35 | Expendable | 9 |
| 10 | Slow Cooker | 140 | Kitchen Ap... | 20 |

Let's see if bringing this column into our PivotTable will work:

| Product Type | Price | Sum of Price Rank |
|-----------------------|---------|-------------------|
| Can Opener | \$ 3.00 | 1 |
| Chef Knife | \$90.00 | 16 |
| Dinner Set | \$35.00 | 9 |
| Fine China Dining Set | \$80.00 | 13 |
| Kitchen Knife Set | \$80.00 | 13 |
| Small Baking Dish | \$12.00 | 2 |

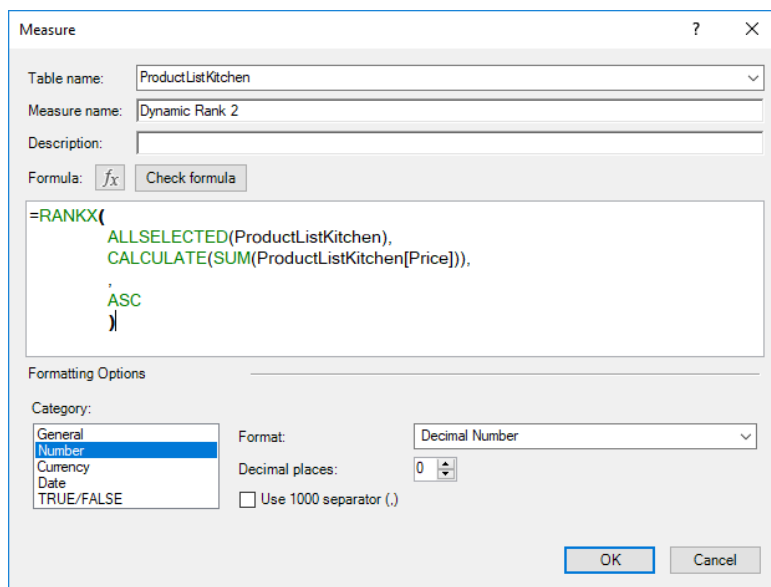
The Rank column does not seem to be dynamic. Let's try creating a measure instead. We are going to use the following DAX code:

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

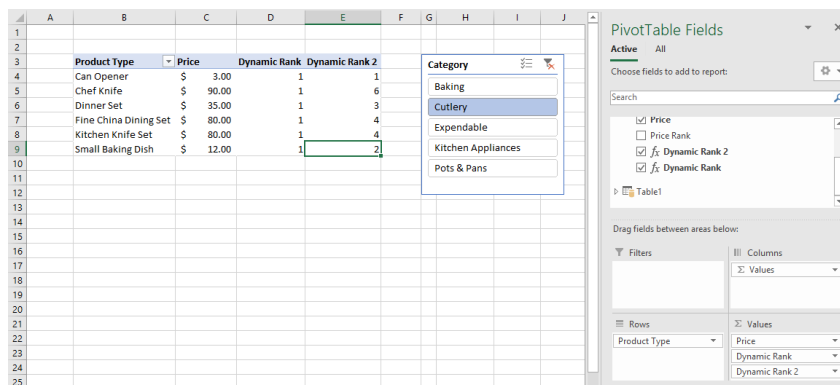
| Product Type | Price | Sum of Price Rank | Dynamic Rank |
|-----------------------|---------|-------------------|--------------|
| Can Opener | \$ 3.00 | 1 | 1 |
| Chef Knife | \$90.00 | 16 | 1 |
| Dinner Set | \$35.00 | 9 | 1 |
| Fine China Dining Set | \$80.00 | 13 | 1 |
| Kitchen Knife Set | \$80.00 | 13 | 1 |
| Small Baking Dish | \$12.00 | 2 | 1 |

All of the ranks appear as '1'. This is because the **RANKX** function is going through each row individually and ranking that row's **ProductListKitchen[Price]** with itself on the same row. Therefore, it will always be one [1]. We need to wrap the **CALCULATE** function around the current expression. This will bypass the current row by same row comparison, and compare the current row's price with the rest of the table, viz.

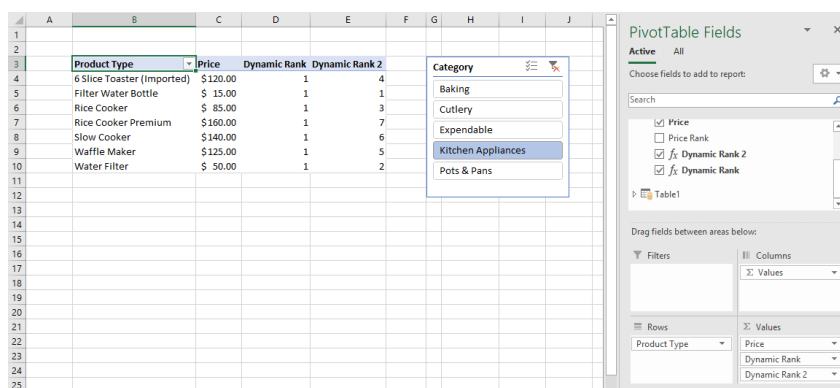
```
=RANKX(
  ALLSELECTED(ProductListKitchen),
  CALCULATE(SUM(ProductListKitchen[Price])),
  ,
  ASC
)
```



After wrapping the **CALCULATE** function, our PivotTable looks like this:



We can change our slicer selection and the dynamic rank will be applied.



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

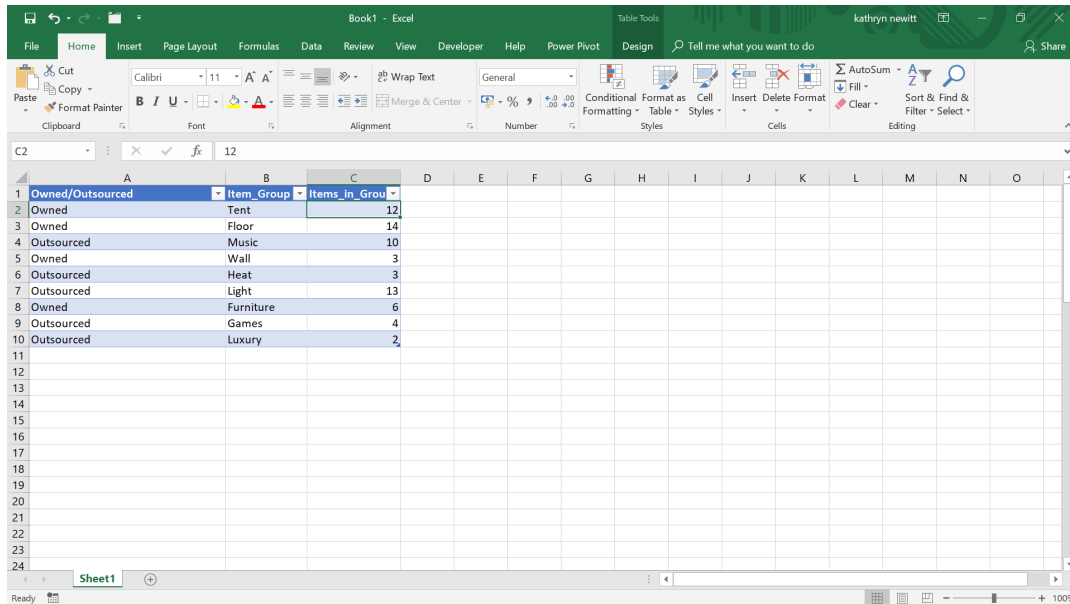
That's it for this month; more next time.

Power Query Pointers

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

This time, let's look at how to achieve a group share calculation in Power Query, in the same way that we may perform one in Excel. This is an exercise in how to manipulate data in Power Query; it is not how I would actually attempt to do this calculation, since it is much easier to do this in Excel or PowerPivot.

Let's imagine we have the following data. We're interested in finding out which groups make up most owned items. We wish to see the percentage of owned or outsourced items for each group.

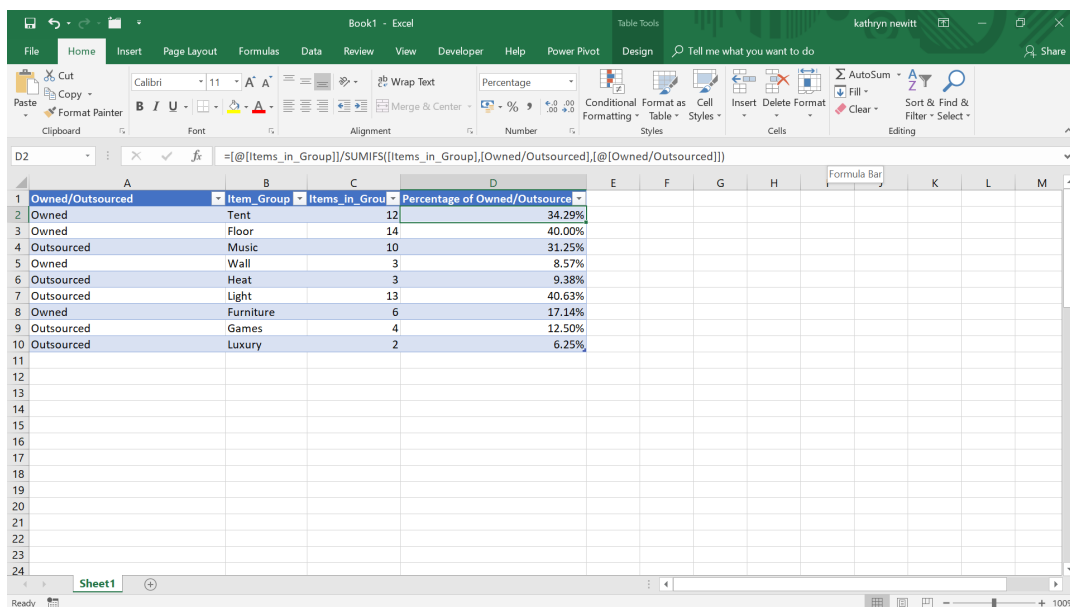


| Owned/Outsourced | Item_Group | Items_in_Group |
|------------------|------------|----------------|
| Owned | Tent | 12 |
| Owned | Floor | 14 |
| Outsourced | Music | 10 |
| Owned | Wall | 3 |
| Outsourced | Heat | 3 |
| Outsourced | Light | 13 |
| Owned | Furniture | 6 |
| Outsourced | Games | 4 |
| Outsourced | Luxury | 2 |

In Excel, we can create an additional column and enter the formula

`=[@[Items_in_Group]]/SUMIFS([Items_in_Group],[Owned/Outsourced],[@[Owned/Outsourced]])`

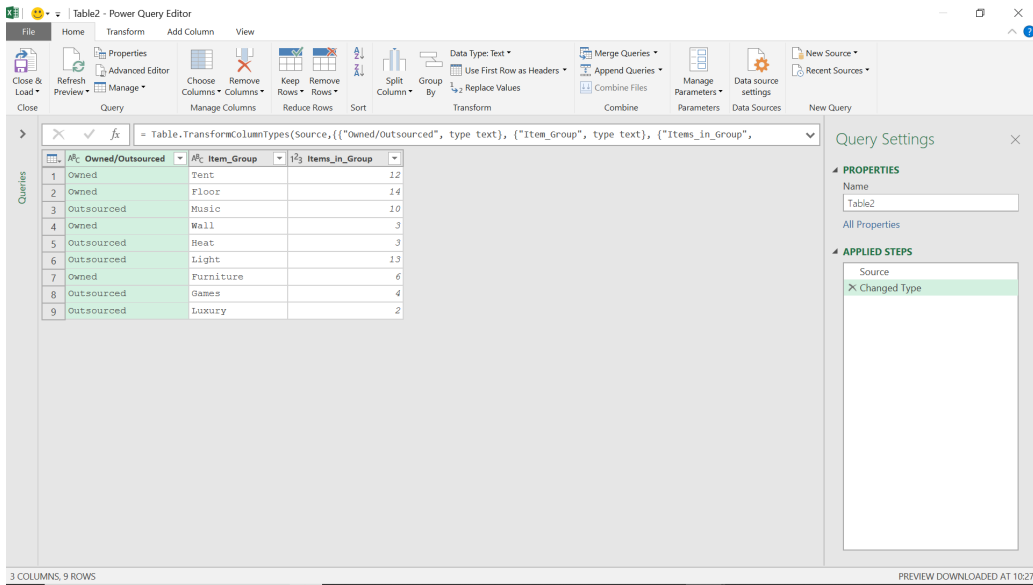
which calculates the percentage of outsourced or owned items that are in the current group.



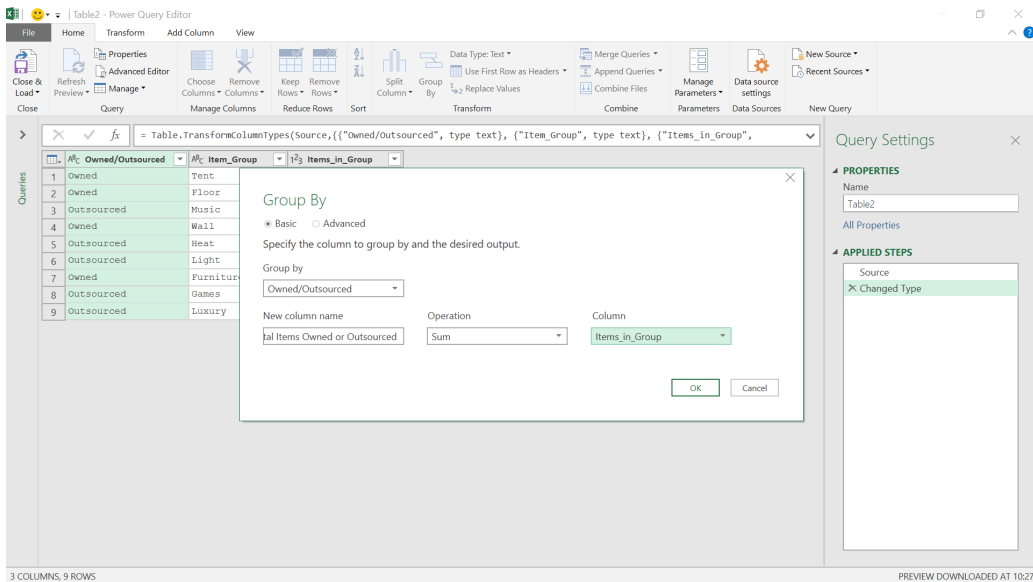
| Owned/Outsourced | Item_Group | Items_in_Group | Percentage of Owned/Outsource |
|------------------|------------|----------------|-------------------------------|
| Owned | Tent | 12 | 34.29% |
| Owned | Floor | 14 | 40.00% |
| Outsourced | Music | 10 | 31.25% |
| Owned | Wall | 3 | 8.57% |
| Outsourced | Heat | 3 | 9.38% |
| Outsourced | Light | 13 | 40.63% |
| Owned | Furniture | 6 | 17.14% |
| Outsourced | Games | 4 | 12.50% |
| Outsourced | Luxury | 2 | 6.25% |

We can see that 'Floor' items make up 40% of the owned items.

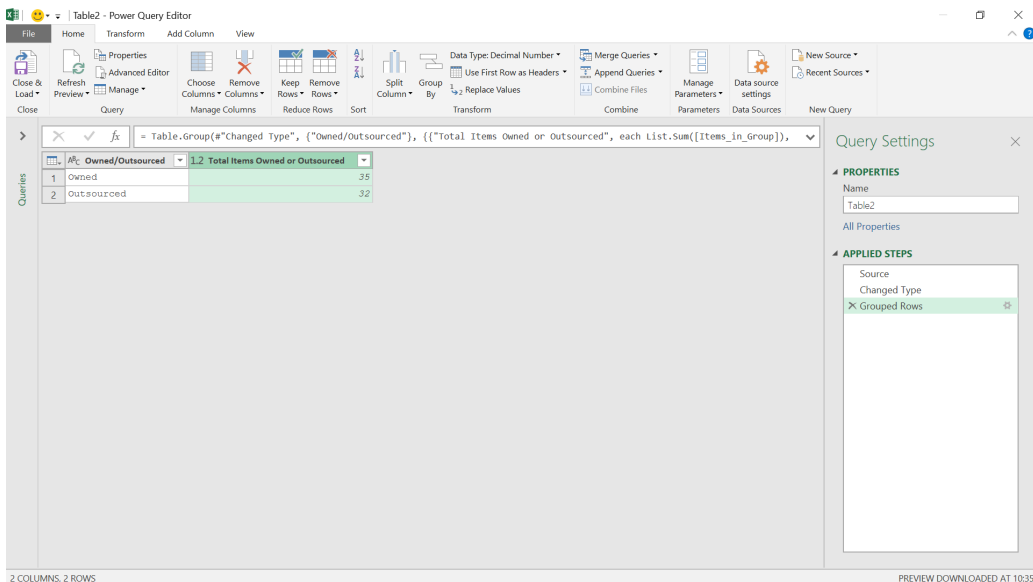
We want to perform the same calculation in Power Query, so we will create a new query 'From Table' from the 'Get & Transform' section on the 'Data' tab.



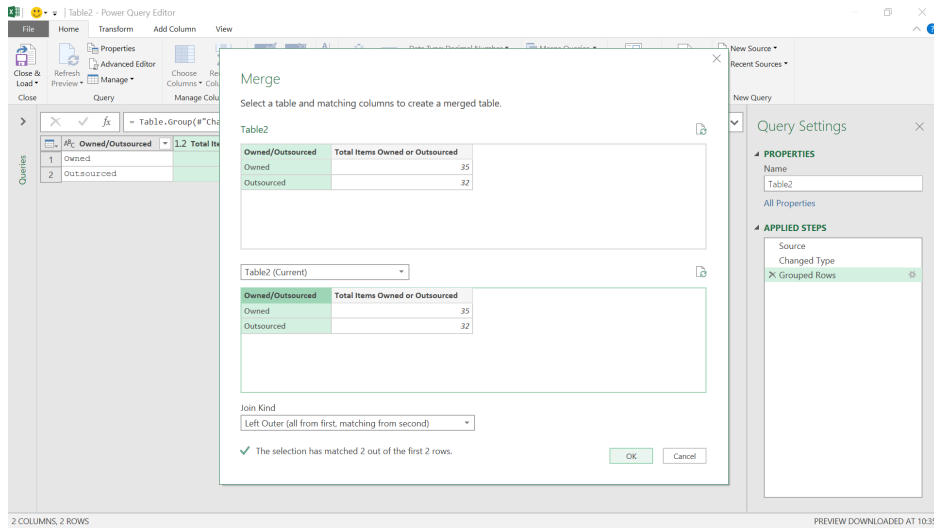
We start by grouping by our **Owned/Outsourced** column. This will allow us to take our first step - to calculate the total number of items in each group:



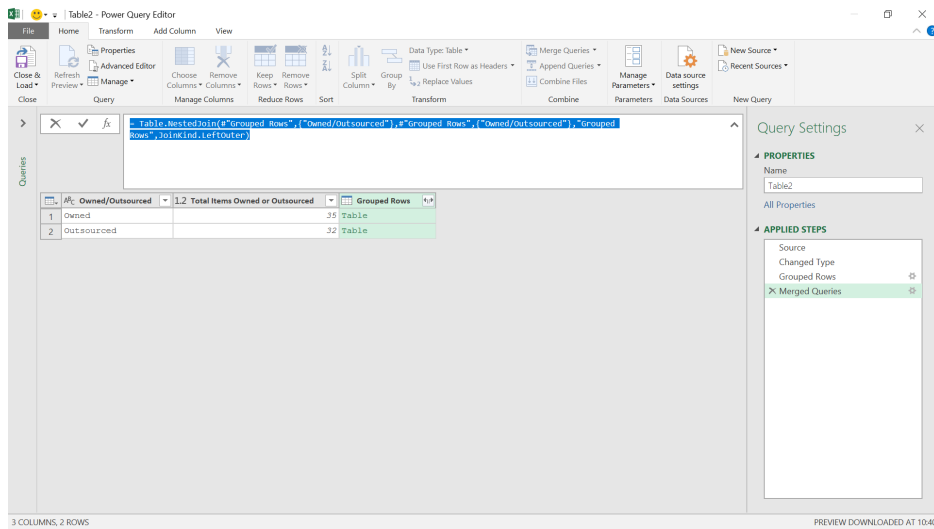
We use a basic group by statement, where we sum **Items_In_Group**. This will simplify our data so that we may see the number of items Owned and Outsourced.



We know that step 'Changed Type' shows us all the details, and 'Grouped Rows' shows us the total items owned and outsourced. One method we can use to get to our final calculation is to merge these steps by merging 'Table2' with itself (we will show another method later).



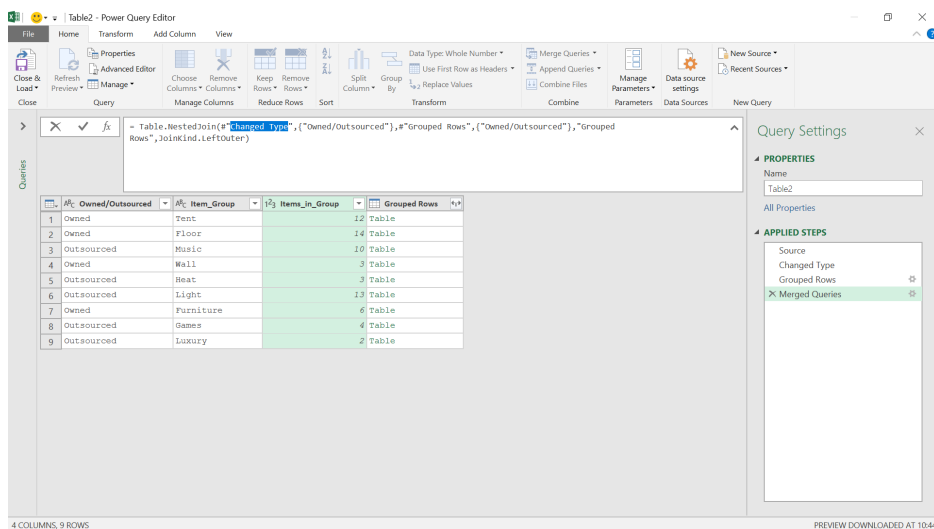
We choose the first column each time and select the left outer join.



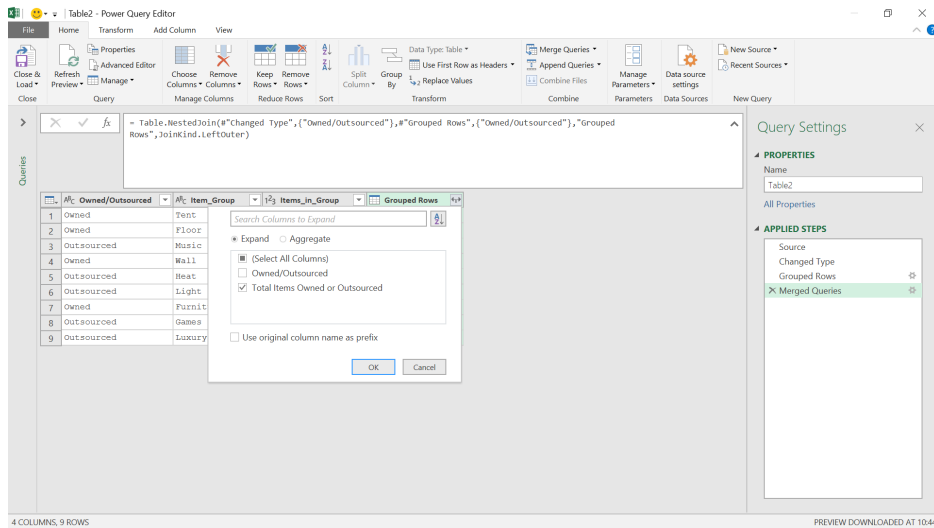
From the M code generated for this step, we can see that the 'Grouped Rows' step has been merged with itself.

= Table.NestedJoin(#"Grouped Rows",{"Owned/Outsourced"},#"Grouped Rows",{"Owned/Outsourced"},"Grouped Rows",JoinKind.LeftOuter)

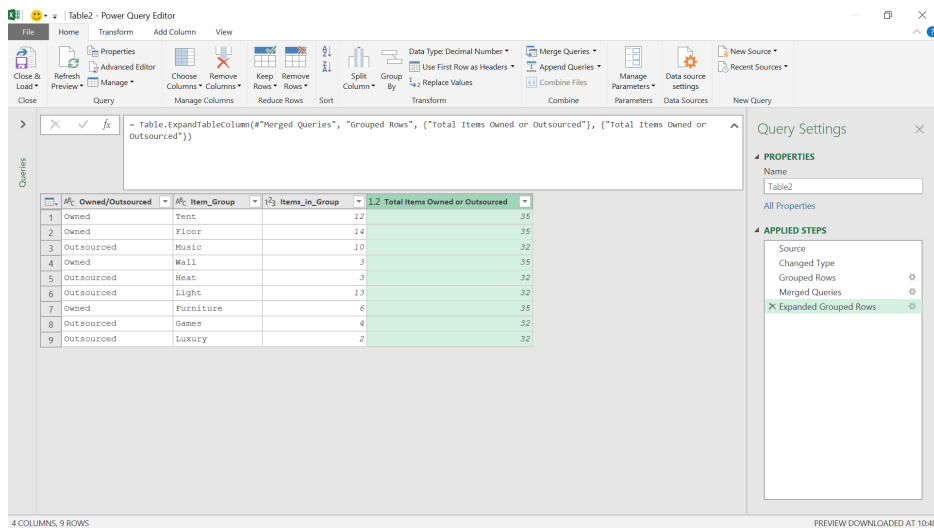
We can edit this step so that we will merge the 'Changed Type' step instead.



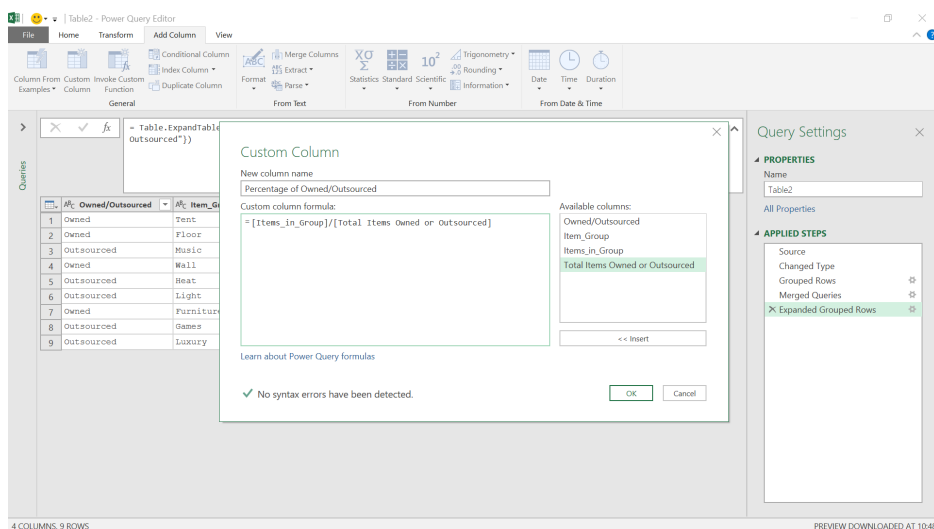
We have now taken our 'Changed Type' step with all the details and merged the 'Grouped Rows' step. We only want **Total Items Owned or Outsourced** from the 'Grouped Rows' step, so we will expand the table and pick this column.



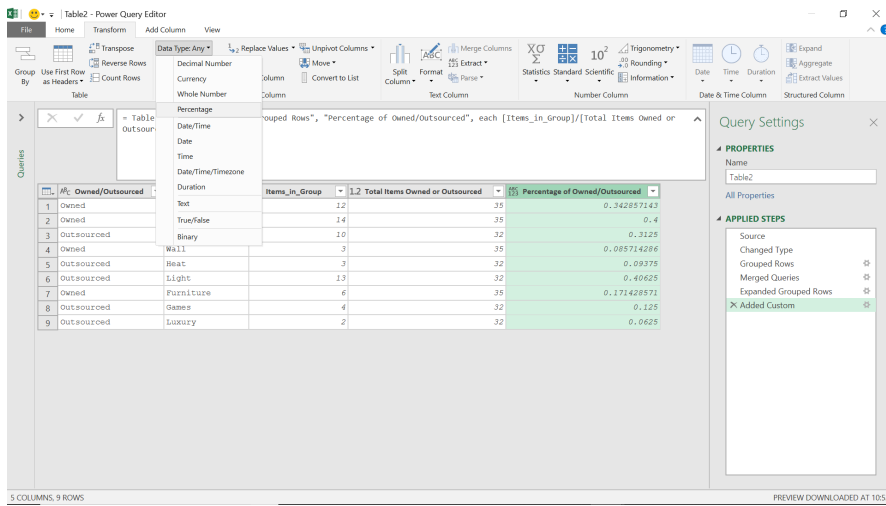
As usual, we do not want to 'Use original column name as prefix'.



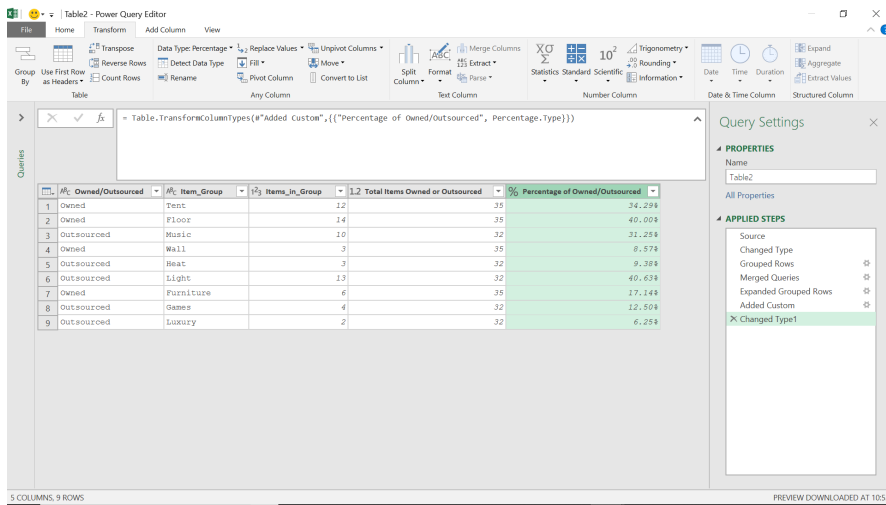
Now we have the two key values for our percentage calculation, we may create a custom column from the 'Add Column' tab.



We divide the number of items in the group by the items owned or outsourced to see which group is mostly made up of owned and outsourced items respectively. We could also do this using the 'Divide' option from the 'Standard' part of the 'From Number' section.

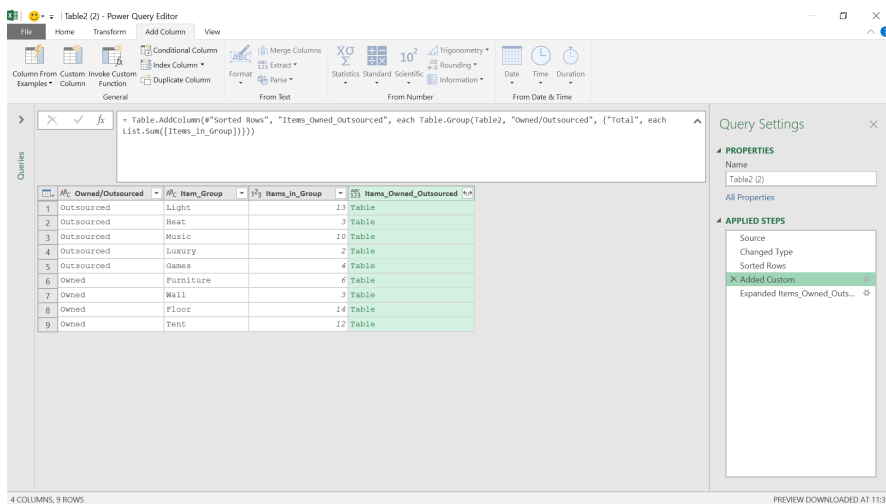


In the same way we did for the Excel method, we convert the new column to type 'Percentage'.



Our calculations are consistent with the Excel method.

We may achieve the same result in another way. We can use M code if we don't want to merge the queries. Instead of using the 'Grouped Rows' step above, we can create a custom column which will carry out the grouping for us.

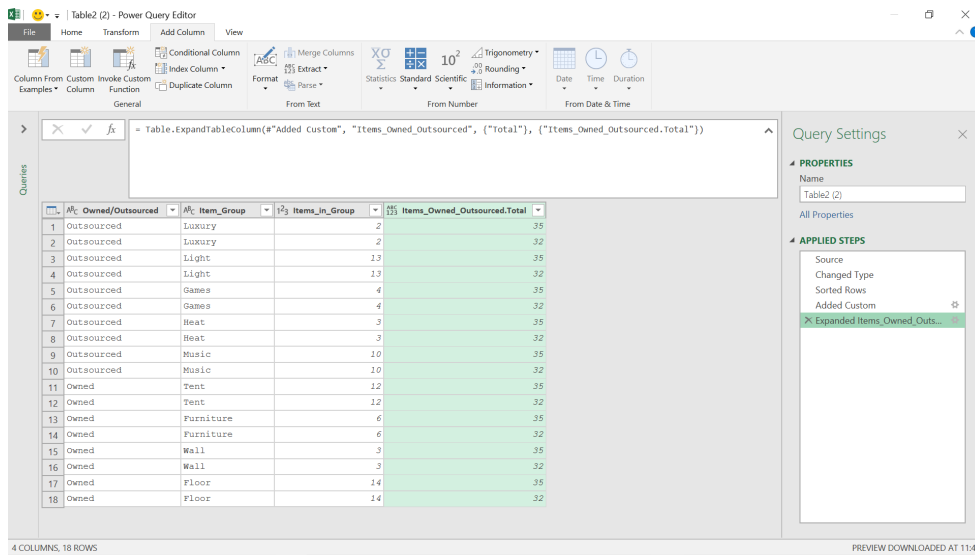


The M code used is:

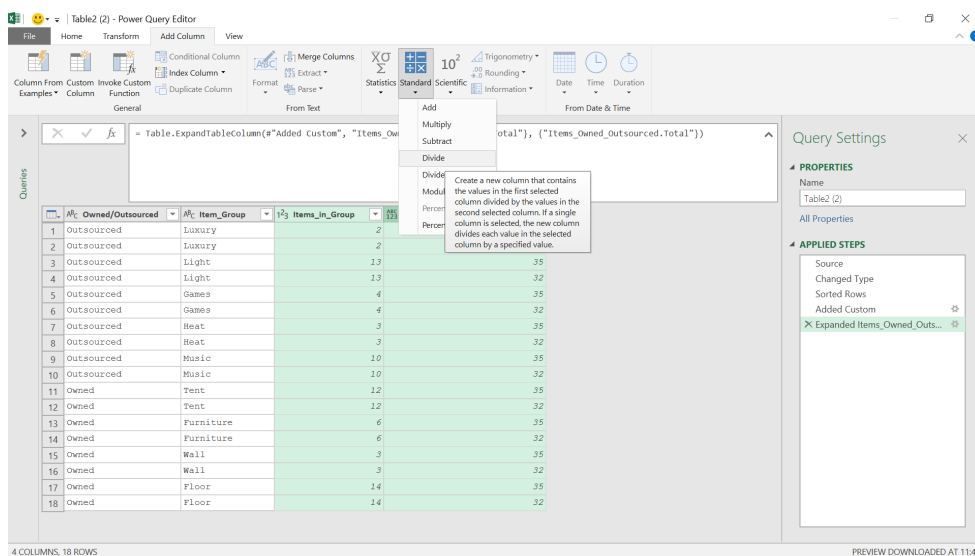
```
= Table.AddColumn("#Sorted Rows", "Items_Owned_Outsourced", each Table.Group(Table2, "Owned/Outsourced", {"Total", each List.Sum([Items_in_Group])}))
```

The part of this which creates my grouped total is `Table.Group(Table2, "Owned/Outsourced", {"Total", each List.Sum([Items_in_Group])})`.

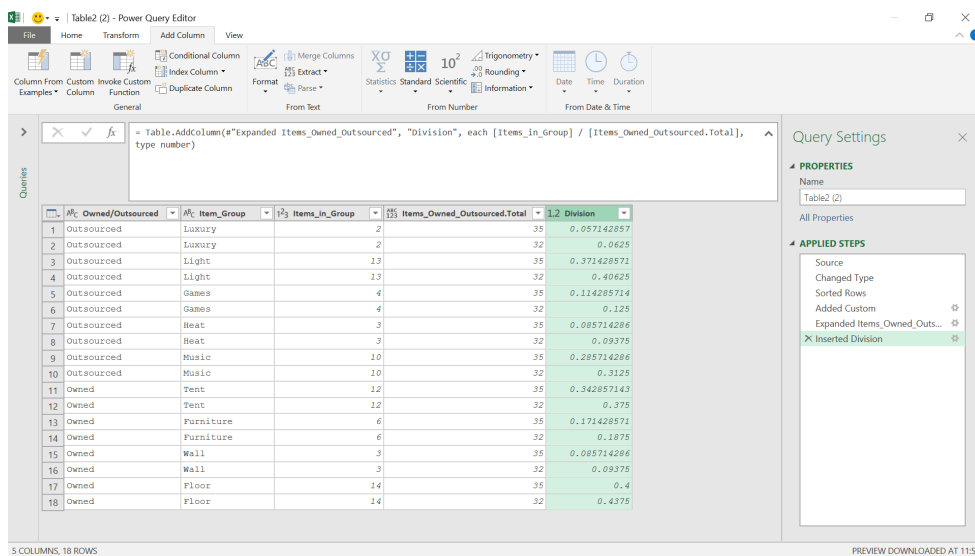
We have grouped by **Owned/Outsourced** and summed **Items_in_Group** within this. This gives us a table very similar to the one created by the merging of the query with itself, which we may expand.



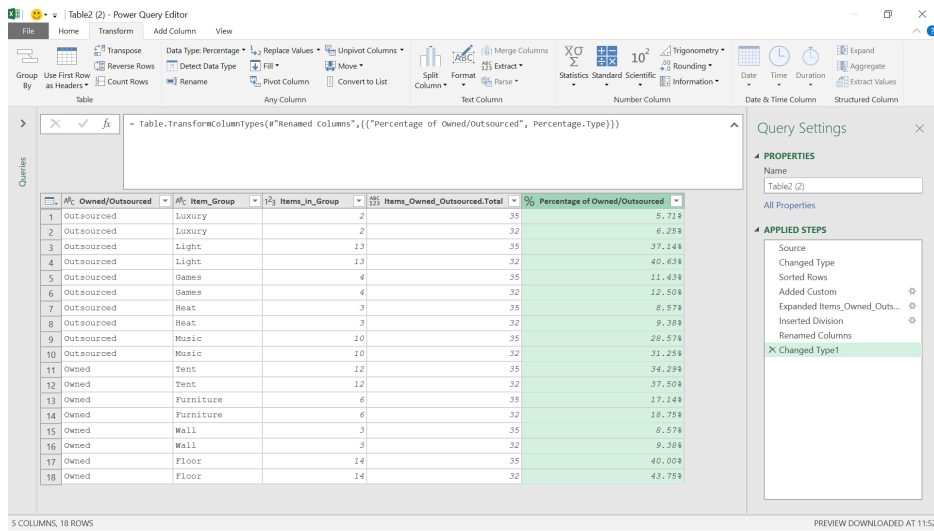
As before, we can create a custom (or divide) column and convert it to a percentage. This time, we will use the divide functionality.



This gives me a new Division column which we can rename.



Having converted this to the right data type, our data is consistent with the other results.



Until next month.

Power Query Pointers

Time for the latest updates. Amongst other announcements, you can now download large semantic models, there is a Power BI enhanced report format (PBIR) and there is also now Subfolder support in Power BI Report Builder.

The full list of updates is as follows:

Reporting

- Visual calculations update in Preview
- Power BI Home in Desktop is now Generally Available

Modelling

- Download large semantic models
- New **INFO** functions

Mobile

- Show Visuals as Tables in Preview

Developers

- Power BI enhanced report format (PBIR) in Preview

Visualisations

- New visuals in AppSource
- Word Cloud by Powerviz
- Drill Down Timeline PRO by ZoomCharts
- Attribute Control Chart by Nova Silva
- Download Button by JTA 18
- New Updates for accoPLANNING (Release 69)

Paginated Reports

- Subfolder support in Power BI Report Builder.

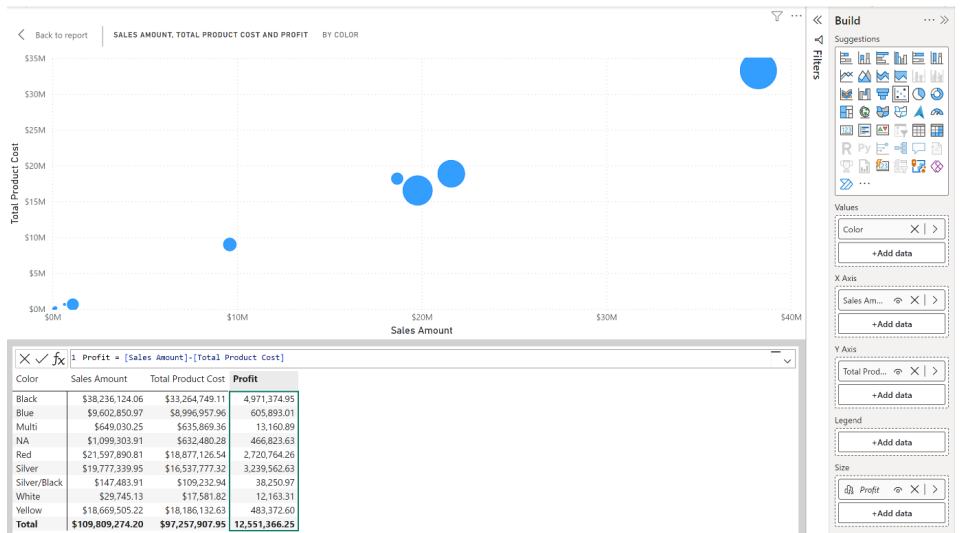
Let's look at each in turn.

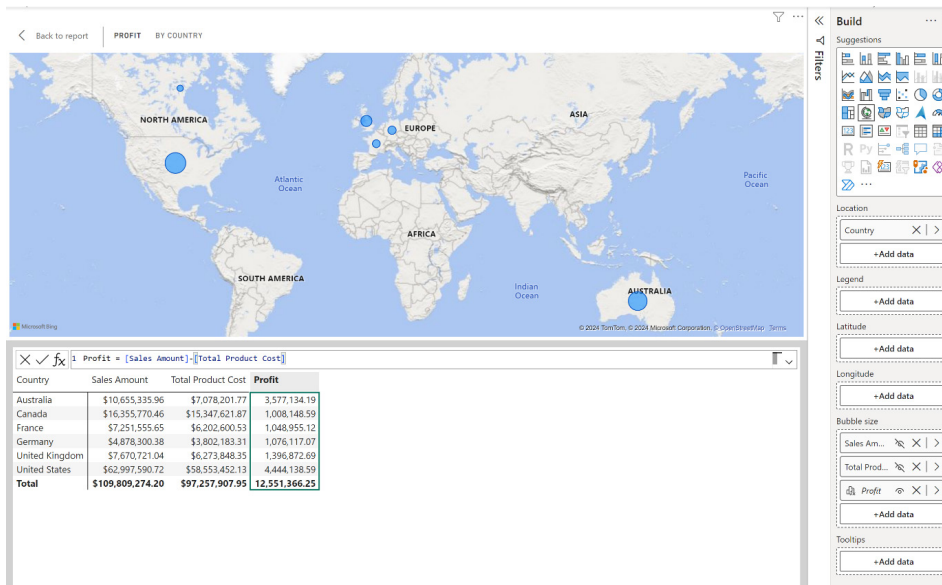
Visual calculations update in Preview

There are two updates to report on here.

SUPPORT FOR SCATTER PLOT AND MAPS

Scatter plots and Maps are now supported with visual calculations. This means you can now add visual calculations to these visual types. Please note that the play axis is not yet supported.



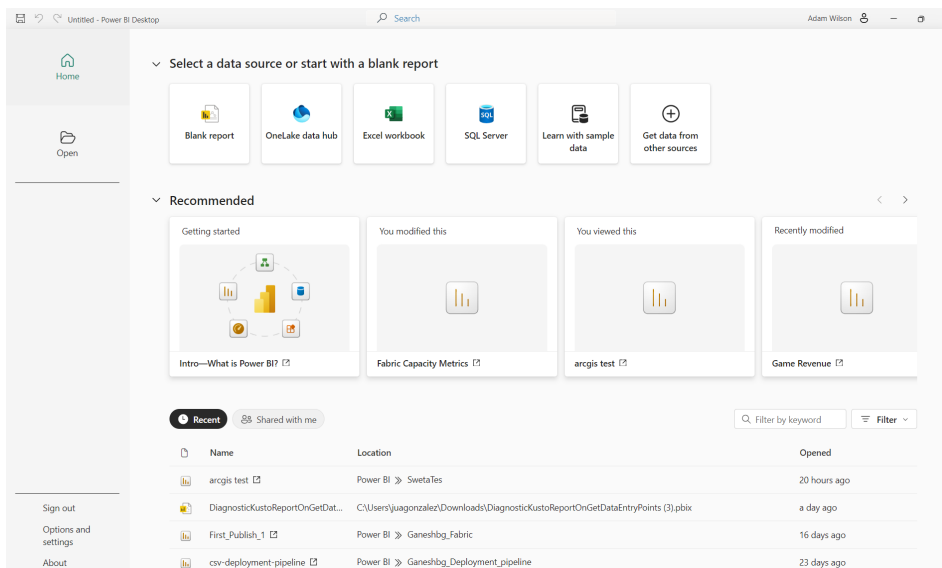


HIGHLIGHTS IN VISUAL MATRIX

If you look closely at the previous screenshots, you might notice Power BI has added a highlight to the visual matrix to indicate which visual calculation you are editing. Apparently, Microsoft plans to rely on the visual matrix to make working with visual calculations easier in the future. There is more to follow...

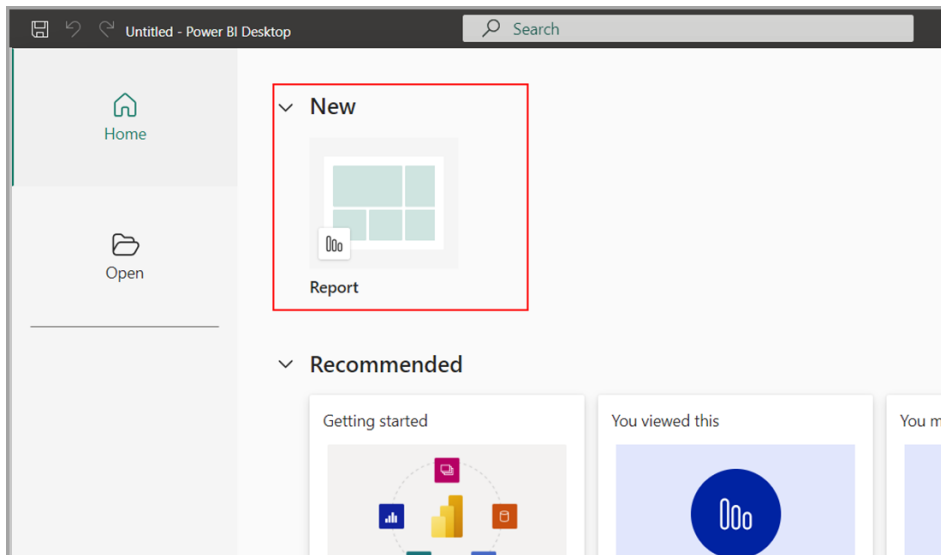
Power BI Home in Desktop is now Generally Available

The new Power BI Home in Desktop is now Generally Available. Initially introduced last February, this update brings new functionalities that make it easier for you to create reports. With data sources directly accessible from the Home screen and enhanced discoverability features like the recommendation section and the quick access list, getting started has become simpler.

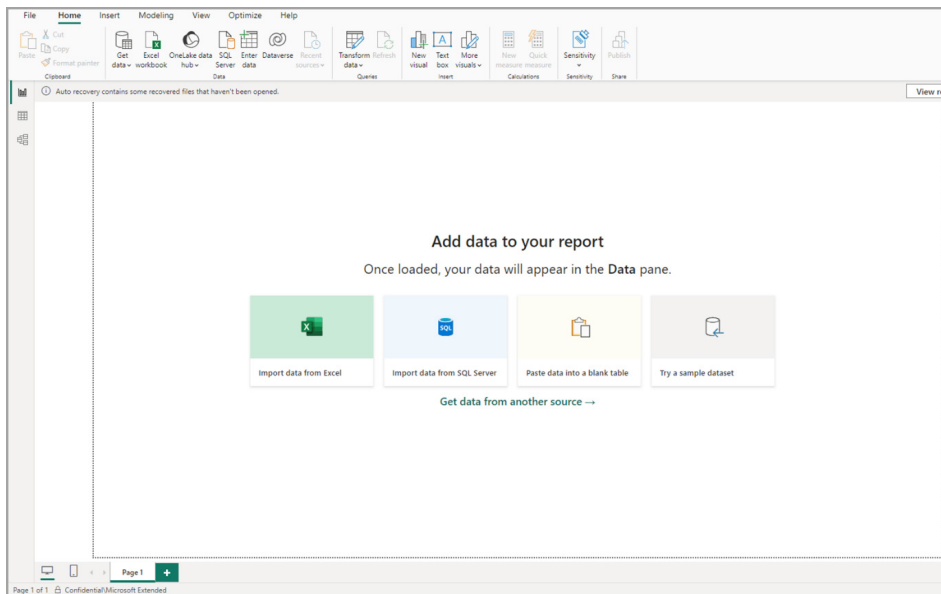


Power BI Home is a central location for your Power BI content and is designed to consolidate your Power BI tasks into a single location, eliminating the need to navigate through multiple menus.

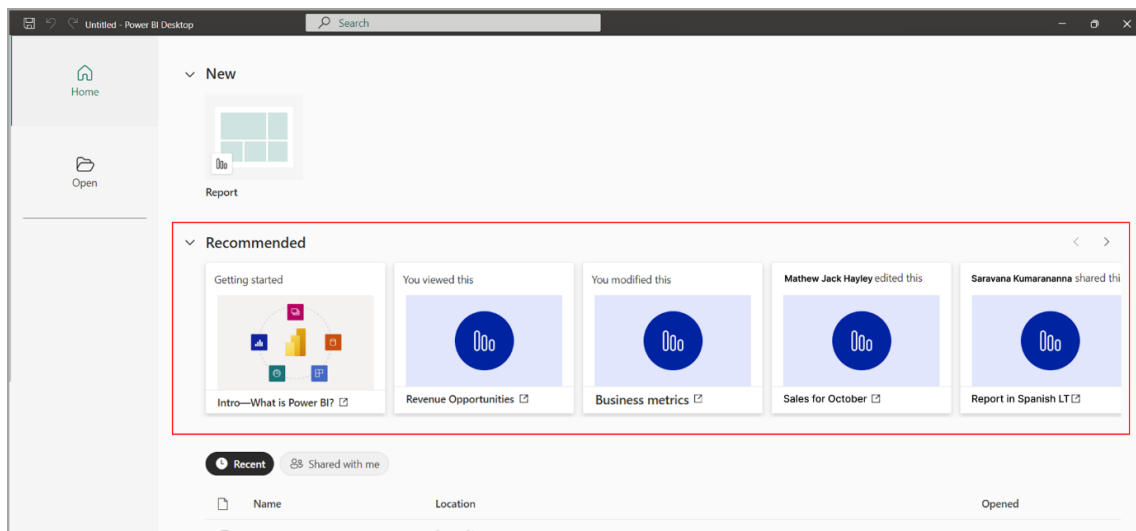
You can open a new report by expanding the 'New' section in Power BI Desktop Home, and selecting Report, viz.



Selecting the 'Report' button creates a new report and opens a blank canvas on which you can add data and begin building your report.



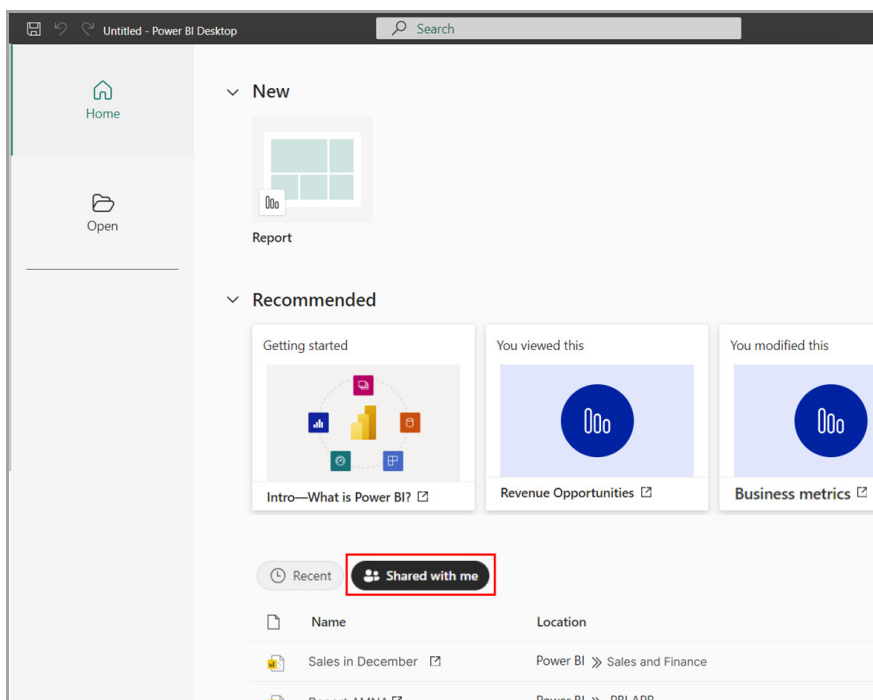
You can open existing reports from the 'Recommended' section of Power BI Home. Files are recommended if they've been recently viewed, edited, edited by someone else or else shared with you.



You can collapse the 'New' and 'Recommended' sections by selecting the small arrow next to their titles.

Furthermore, in Power BI Home, a list of reports you've recently accessed is displayed below the 'Recent' section. Each item shows the file name, its location and the last time you opened the file.

When you select the 'Shared with Me' button at the top of the quick access list, this list displays files that are shared by people in the same organisation, *i.e.* this provides a central location for shared files, making it easier to work together in Power BI.



To view a report in Power BI Desktop Home, you must meet the following requirements:

- have access to the file being displayed
- have a Power BI Pro, Power BI Premium Per User or E5 license.

You should keep the following considerations in mind when using Power BI Desktop Home:

- B2B users and anonymous users can't search files in Power BI Desktop
- If substrate capability is turned off for your organisation, Power BI Home doesn't display Recommended files, you'll only see local files
- Power BI Home features don't work with personal OneDrive accounts
- Power BI Home features aren't available in a sovereign cloud tenant
- Reports with row-level security aren't currently displayed in the Recommended list. You must download such files to see or use them in Power BI Desktop.

Download large semantic models

It has now been announced that you may download your large semantic models to Power BI Desktop as a .pbix file. Previously, you could only edit reports connected to these models in live connected report mode. Now, you can open and edit your reports and the large model itself as a .pbix file all within Desktop. This unlocks the full end to end Desktop authoring experience for your large semantic models.

New INFO functions

New **INFO** functions are now available. All the **INFO** functions are **DAX** function versions of existing Analysis

| DAX Function | DMV |
|---------------------|--------------------------|
| INFO.CALCDEPENDENCY | DISCOVER_CALC_DEPENDENCY |
| INFO.CATALOGS | DBSCHEMA_CATALOGS |
| INFO.PROPERTIES | MDSHEMA_PROPERTIES |

All **INFO** functions may now take optional input parameters. As DMVs these optional input parameters are called **restrictions**. For all the **INFO** functions, this includes their columns, but some have additional restrictions or optional input parameters, allowing you to analyse the items in the semantic model needed for that **DAX** query. This is helpful

when analysing what is being used by the semantic model in a **DAX** query from a visual in a report.

For example, these are valid uses of **INFO** functions with and without the input parameters:

```

1  EVALUATE
2  |   INFO.CALCDEPENDENCY()
3
4  // Show dependent items when querying the 'Date' table
5  EVALUATE
6  |   INFO.CALCDEPENDENCY("Query", "EVALUATE 'Date'")
7
8  // Show dependent items of the measure [Total Gross Sales]
9  EVALUATE
10 |   INFO.CALCDEPENDENCY("Query", "EVALUATE {[Total Gross Sales]}")
11
12

```

| [DA...] | [OBJECT_TYPE] | [TABLE] | [OBJECT] | [EXPRESSION] | [REFERENCED_I] | |
|---------|---------------|---------------------|--------------|------------------|-----------------------------|---------------|
| 13 | 4cc... | CALC_TABLE | Aggregate by | Aggregate by | { ("Sales", NAMEOF('Fin... | MEASURE |
| 14 | 4cc... | CALC_TABLE | Aggregate by | Aggregate by | { ("Sales", NAMEOF('Fin... | MEASURE |
| 15 | 4cc... | CALC_TABLE | Aggregate by | Aggregate by | { ("Sales", NAMEOF('Fin... | MEASURE |
| 16 | 4cc... | CALC_TABLE | Aggregate by | Aggregate by | { ("Sales", NAMEOF('Fin... | MEASURE |
| 17 | 4cc... | CALC_TABLE | Aggregate by | Aggregate by | { ("Sales", NAMEOF('Fin... | MEASURE |
| 18 | 4cc... | ATTRIBUTE_HIERARCHY | Aggregate by | Parameter | | ATTRIBUTE_HIE |
| 19 | 4cc... | ATTRIBUTE_HIERARCHY | Aggregate by | Parameter Fields | | ATTRIBUTE_HIE |
| 20 | 4cc... | MEASURE | Financials | Total Units Sold | SUM('Financials'[Units S... | TABLE |
| 21 | 4cc... | MEASURE | Financials | Total Units Sold | SUM('Financials'[Units S... | COLUMN |
| 22 | 4cc... | MEASURE | Financials | Total Sales | SUM('Financials'[Sales]) | TABLE |
| 23 | 4cc... | MEASURE | Financials | Total Sales | SUM('Financials'[Sales]) | COLUMN |

Additionally, a bug with the copy functionality of the results grid has been fixed. Previously, if the results included a blank cell the copy did not work. Now, copy works with blank cells, and Microsoft has included

a right-click copy as well as the option to choose if you want to copy the entire table or just the selected cells from the copy button.

```

1  EVALUATE
2  |   DATATABLE(
3  |       "Num", INTEGER,
4  |       "Val", STRING,
5  |       {
6  |           {1, blank()},
7  |           {2, "hello"}
8  |       }
9  |   )
10
11

```

| [Num] | [Val] |
|-------|-------|
| 1 | |
| 2 | hello |

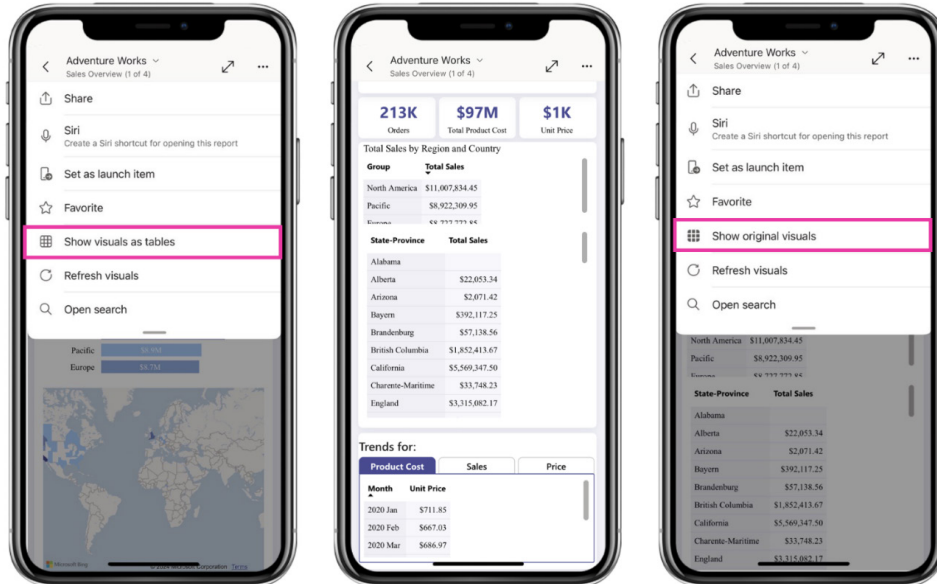
Copy -> **Entire table** will include the headers. **Copy** -> **Selected cells** by selecting all the cells, including if you click the table icon in the top left of the results, will not include the headers.

Show Visuals as Tables in Preview

Microsoft claims they are striving to make the Power BI mobile app as accessible as possible for everyone. This is apparently why they have added the 'Show Visuals as Tables' view mode to all reports. You may use this feature to instantly view all the visuals (excluding slicers, cards and non-data-driven visuals) in your report as Table visuals. This feature makes it possible for users who rely on accessibility screen readers to read the underlying data presented in each visual. It also benefits users

who prefer to see their data in tabular form, which is more like viewing it in Excel.

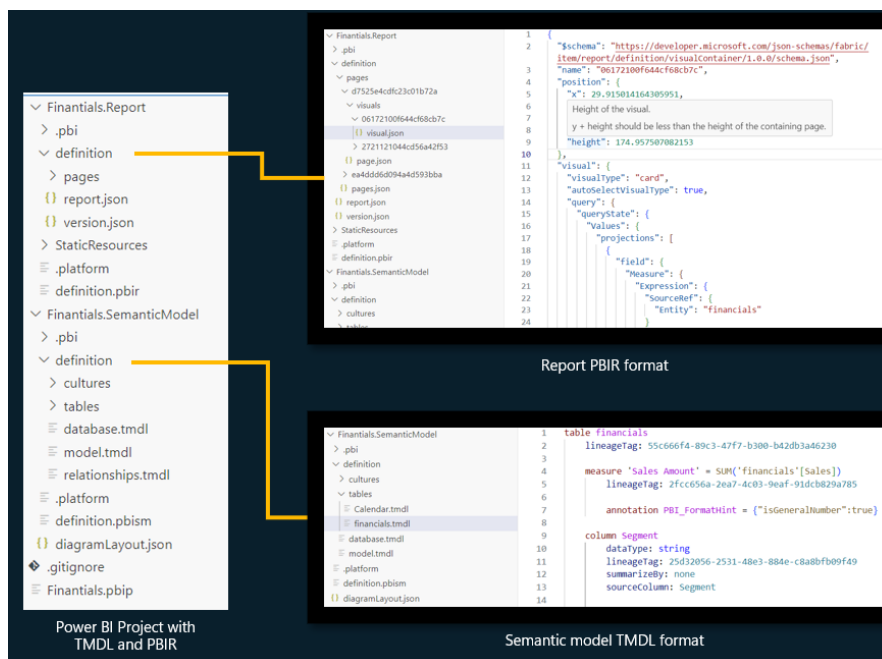
To turn on the 'Show Visuals as Tables' view mode, tap the new button that has been added to the options menu (...) in the report header. The view mode is applied to all pages in the report. To go back to the regular view, tap 'Show original visuals' in the same menu.



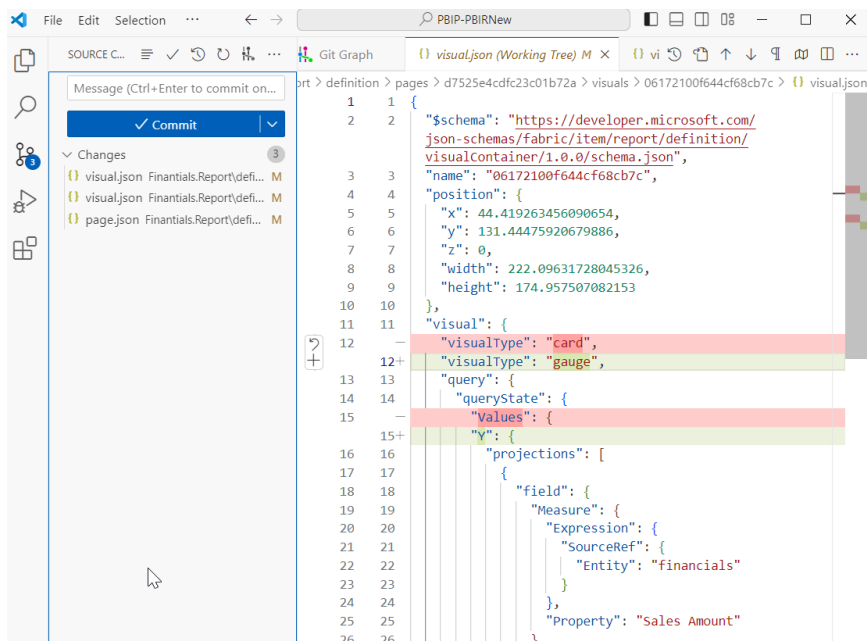
Power BI enhanced report format (PBIR) in Preview

Microsoft has also announced the **Power BI enhanced report format (PBIR)** for Power BI Project files (PBIP). This marks a significant milestone in achieving the primary goal of Power BI Desktop developer mode: to provide source control friendly file formats that unblock co-development and enhance development efficiency.

Power BI Projects (PBIP) now support saving the report and semantic model into a folder using source-control friendly formats: **PBIR** for the report and **TMDL** for the semantic model.



The PBIR file format greatly simplifies the tracking of changes and resolution of merge conflicts by using properly formatted JSON and organizing each visual, page, bookmark, etc., in separate individual files within a folder structure.



You can also greatly enhance your report development efficiency, either by simply copying and pasting or pasting visuals / pages / bookmarks/... files between reports or apply manual / programmatic batch changes to the PBIR files.

Unlike PBIR-Legacy (report.json), PBIR is a publicly documented format and allows modifications from non-Power BI applications. Each file has a public JSON schema, which documents each property and lets code editors like Visual Studio Code perform syntax validation while editing. On open, Power BI Desktop will validate the changed PBIR files to guarantee successful loading.

To open it, do remember PBIR is currently in Preview, and you can only create or convert existing Power BI project files to PBIR using Power BI Desktop. You must first enable the feature in Power BI Desktop preview features: go to **File -> Options and settings -> Options -> Preview**

features and check the box next to 'Store reports using enhanced metadata format (PBIR)'.

During Preview, Fabric Git Integration and Fabric REST Apis will continue to use PBIR-Legacy (report.json) when exporting the report definitions. However, if the report is imported into Fabric using PBIR format, then both features will start exporting the report definition using PBIR format. At General Availability, PBIR will become the default report format.

Initially, the PBIR format will have some service restrictions, such as:

- unable to publish the report in Power BI App
- unable to use subscriptions
- unable to download PBIX.

These restrictions will be removed in the following months.

New visuals in AppSource

This month see the following new visuals:

- Aimplan Status Visual
- hi-chart Reporting Studio
- Matrix Planner
- vuurmans_custom_polar_area_chart
- Water Cup.

Word Cloud by Powerviz

Powerviz's Word Cloud is a visual representation of text, with word size indicating frequency or importance in the given content. It offers a quick overview of key themes and is commonly used in presentations and data analysis to highlight patterns and key terms.

Key features include:

- **word options:** customise text styles and appearance
- **direction:** control word orientation with various styles
- **colours:** choose from 30+ colour schemes
- **shapes:** create unique word clouds with icons and images
- **ranking:** filter out Top / Bottom N Words
- **exclude:** easily remove unwanted words, symbols from the text to create a clean and focused word cloud
- **conditional formatting:** easily spot words with dynamic rules
- **Lasso / Reverse Lasso:** select and deselect multiple words together.

Business use cases include:

- **Marketing:** analyse feedback, SEO keywords and sentiment
- **Education:** improve writing skills and engage students with word clouds
- **Market Research:** quickly analyse survey responses and opinions
- **Presentations:** capture attention and summarise information visually
- **Data Analysis:** explore textual data for insights and trends.

Word Cloud

CREATE EYE-CATCHING & STUNNING WORD CLOUDS
CREATE A POWERFUL AND FEATURE-RICH WORD CLOUD WITH SHAPES, COLORS AND ADVANCED CUSTOMIZATIONS

Examples of word clouds in various shapes and colors:

- Apple shape (red)
- Magazine shape (yellow)
- Twitter bird shape (blue)
- Dog shape (black)
- Tree shape (green)
- Leaf shape (green)

Word Cloud

CREATE EYE-CATCHING & STUNNING WORD CLOUDS
WORD OPTION, DIRECTION, VARIOUS SHAPES, EXCLUDE, CONDITIONAL FORMATTING, RANKING, AND MORE

Exclude Unwanted Elements Just in One Click.

Before After

Options shown:

- Exclude Unwanted Elements: Numbers, Emojis, Words, Conjunctions, Punctuations, Special Characters
- Colors: Color palette and gradient options
- Direction: Orientation and layout settings
- Conditional Formatting: Rules for word colors and sizes
- Word Options: Font, size, and character count settings

Drill Down Timeline PRO by ZoomCharts

With Drill Down Timeline PRO, you can create intuitive timeline charts with up to 25 series. Users may drill down by simply clicking data directly on the chart. For example, the initial view may show aggregated monthly totals, and the user can click on a specific month to quickly access daily values.

When paired with other visuals that support cross-filtering, Timeline PRO can become a part of a dynamic and engaging report that provides insights to the user. For example, when you select a specific time range on Timeline PRO, other visuals will display data that's relevant only to that time period. Conversely, other visuals or slicers can dynamically filter data on your timeline chart as you explore the report.

Main features include:

- on-chart drill down
- up to 25 series (columns, lines and areas)
- 'Legend' field support
- custom date / time hierarchy (from milliseconds to decades)
- **DAX** measure support
- customisation: defaults or individual series settings
- up to four [4] static or dynamic threshold lines / areas
- conditional formatting
- touch support.

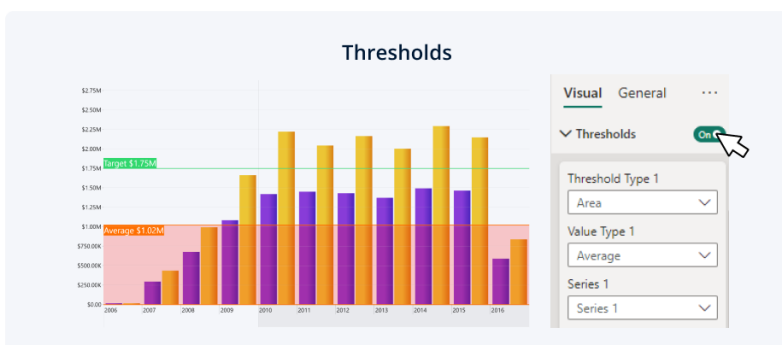


Drill Down Timeline PRO



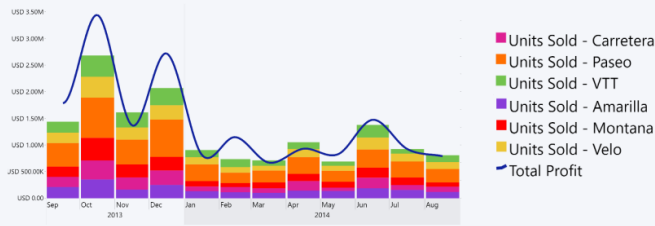
SIMPLE AND POWERFUL

Create beautiful and interactive charts for your date/time data that reveal valuable insights with just a click.





'Legend' Field Support

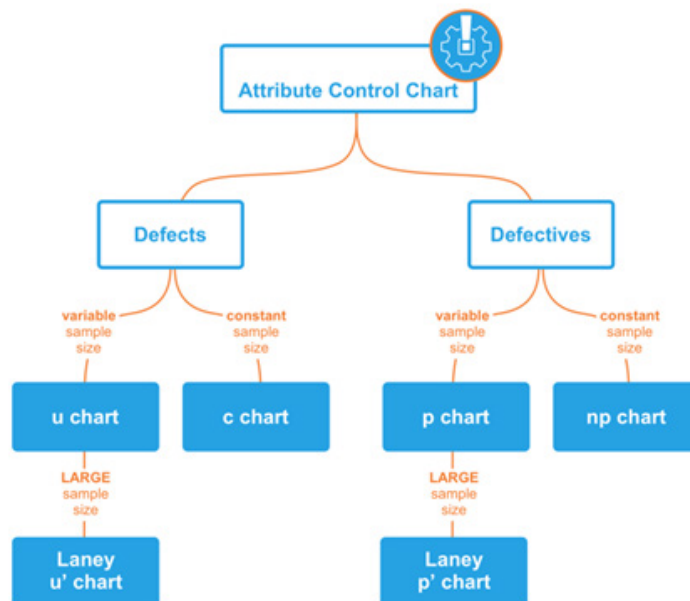


Series Customization



Attribute Control Chart by Nova Silva

The new Attribute Control chart for Power BI is a robust mechanism for monitoring and controlling categorical data variations. The Attribute Control Chart offers a comprehensive suite of six [6] attribute charts to cater to diverse analytical needs:



- **c chart:** monitors number of defects per sample (sample size is constant)
- **u chart:** tracks the number of defects per unit (sample size varies)
- **Laney u' chart:** a modified version of the u chart to adjust for overdispersion or under dispersion in your data
- **p chart:** observes the proportion of defective items per sample (sample size varies)
- **Laney p' chart:** a modified version of the p chart to adjust for overdispersion or under dispersion in your data
- **np chart:** displays the count of defective items per sample (sample size is constant).

Attribute Control Chart: monitor process quality



Power BI users can delve deeper into their data, facilitating timely decision-making and continuous improvement. From manufacturing to healthcare, the Attribute Control Chart equips organisations across industries with the insights needed to uphold quality standards and drive operational excellence.

Download Button by JTA 18

The 'Download Button by JTA' is a custom visual designed specifically for Power BI users, empowering them to export large datasets effortlessly. Its key features are as follows:

- **enhanced data export:** download up to 300,000 rows and 15 columns of data in CSV format
- **customisable interface:** tailor the visual interface to match the reporting style.

The custom visual leverages the browser's cache memory. It works by fetching data in "chunks" of 30k rows, storing each chunk in the

browser's cache until the entire dataset is compiled for download. This incremental approach requires the use of browser memory to ensure seamless processing. Unfortunately, this means that the visual does not operate as intended in Power BI Desktop due to technical constraints and security restrictions within that environment and you must use the Power BI Service.

For the very same reason, there's a possibility that it may encounter limitations or restrictions in certain browsers. Each browser has its own set of security measures and permissions that can affect how the visual operates.

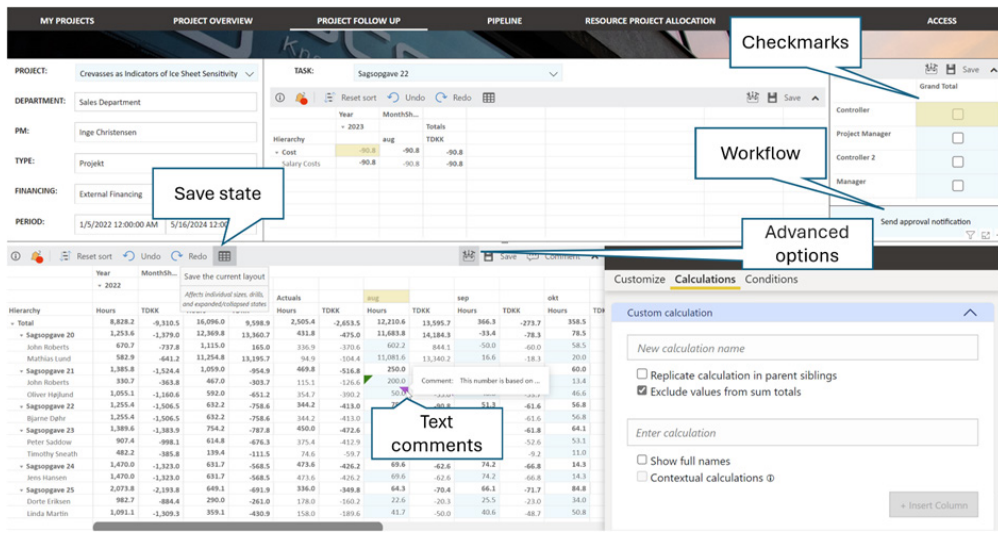
New Updates for accoPLANNING (Release 69)

accoPLANNING for Power BI enhances your business capabilities with advanced planning, forecasting, budgeting, project management and analysis solutions, all whilst enabling writeback capabilities.

The new features for this release include:

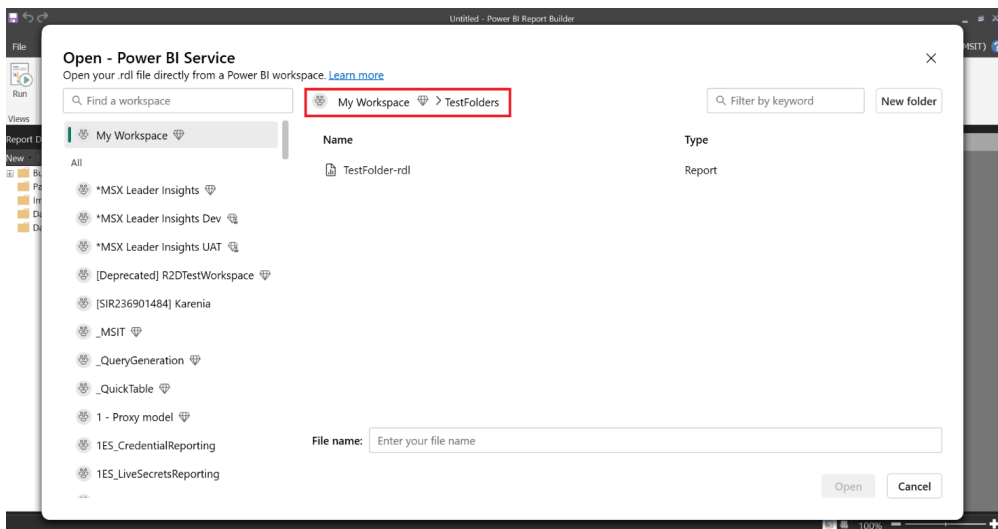
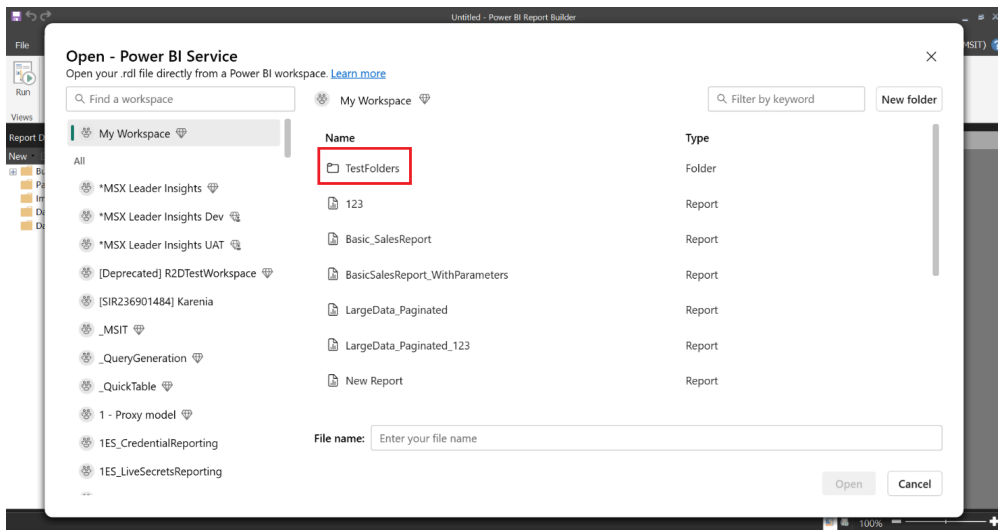
- **advanced options and save state:** customise column and row colours, adjust text formats and create custom calculated rows and columns
- **improved settings interface:** the chart has completely revamped the formatting areas and relocated some settings to make them more intuitive
- **writeback table scripts:** easily create writeback tables in your database for accoPLANNING visuals with this new feature
- **dropdown list:** limit input to predefined values from a dropdown list consistent with data validation
- **new aggregation options:** Median, Product, Min and Max have now been added, amongst other useful aggregations for *ad hoc* scenarios
- **enhanced copy-paste functionality:** improved copy-paste capabilities between Excel and the visual
- **themes:** quickly create visually appealing reports with new theme options
- **self-hosted API:** now available with enhanced data security features, including additional encryption
- **Fabric ready:** the visual now supports writeback directly to Fabric Data Warehouse.

By integrating accoPLANNING with Power BI, you can streamline your planning and reporting processes.



Subfolder support in Power BI Report Builder

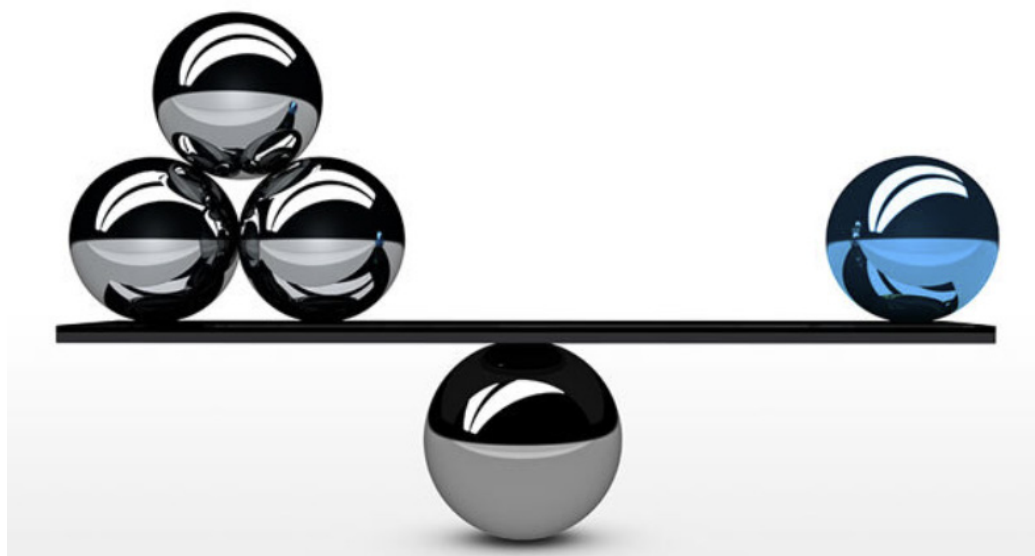
You can now open reports that are in folders and subfolders from Power BI Report Builder. These reports may reside in folders and / or subfolders in workspaces that the user has access to.



You can also publish paginated reports to folders / subfolders that you have access to in the Power BI Service.

More next month.

New Features for Excel



It's a little quieter on the Excel front this month, but I suppose it's more about quality than quantity. This month sees two key announcements: filtering for comments is rolling out to Excel for the web and the new regular expression ("regex") functions are rolling out to Insider Beta for Excel for Windows & Mac.

It seems a little weird explicitly listing them this time out:

Excel for the web

- Filtering for comments

Excel for Windows

- New regular expression ("regex") functions (Insiders)

Excel for Mac

- New regular expression ("regex") functions (Insiders).

Let's get started with, er, "all" of them.

Filtering for comments

You can now filter comments that are active, resolved or ones that **@mention** you so that you can find the comments you're looking for. Filtering to a certain set of comments will update which comment indicators you see on the worksheet so you can focus on which comments are important with the context of your work. This feature is already supported on Mac and is currently rolling out to Web users.

New regular expressions ("regex") functions

Yes, we know we covered it last month, but our CDO (like Obsessive Compulsive Disorder but with a need to have the initials in alphabetical order) requires us to reiterate in this section.

Regular expressions is a language used for pattern-matching text content. The term "regex" is an abbreviation of "regular expressions" and is frequently implemented in various programming languages such as C, C++, Java, Python, VBScript – and now, Excel for Windows and Mac, albeit the Insiders editions.

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

So how is it coming to Excel? There are three [3] new Excel functions coming to the Beta versions of Excel:

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

Possibly, we should re-explain "regular expressions" before continuing (but feel free to skip if you checked this all out in last month's newsletter). Alternatively referred to "rational expressions" upon occasion, a regular expression is a sequence of characters that specifies what is known as a "match pattern" in text. You have most likely used this functionality in Excel already, with features such as "Find and Relace" or by using the **FIND** or **SEARCH** functions in Excel. The purpose of these three [3] new functions (presumably, this is just a start!) is to help you match, locate and manage text (strings) in Excel.

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

| Token | Meaning |
|-----------|---|
| \ | This converts special characters (metacharacters) to literal characters, and also allows the literal matching of the regex delimiter in use, e.g. '/' |
| . | Matches any character other than newline |
| ^ | Matches the start of string without consuming any characters. If multiline mode is used, this will also match immediately after a newline character |
| \$ | Matches the end of string without consuming any characters. If multiline mode is used, this will also match immediately before a newline character |
| a? | Matches zero [0] or one [1] of a. This matches an 'a' character or nothing |
| a* | Matches zero [0] or more of a. This matches zero or consecutive 'a' characters |
| a+ | Matches one [1] or more of a. This matches consecutive 'a' characters |
| a{4} | Matches exactly four [4] instances of 'a' |
| a{4,} | Matches four [4] or more instances of 'a' |
| a{4,6} | Matches between four [4] and six [6] instances of 'a' |
| \A | Matches the start of a string only. Unlike ^, this is not affected by multiline mode |
| \Z | Matches the end of a string only. Unlike \$, this is not affected by multiline mode |
| \z | Matches the absolute end of a string only. Unlike \$, this is not affected by multiline mode and in contrast to \Z, this will not match before a trailing newline at the end of a string |
| \b | Matches a word boundary. It matches without consuming any characters, immediately between a character matched by \w and a character not matched by \w. It cannot be used to separate non-words from words |
| \B | Matches a non-word boundary. It matches without consuming any characters, at the position between two characters matched by \w or \W |
| i | A case insensitive match is performed |
| x | Ignore whitespace / verbose. This flag instructs the engine to ignore all whitespace and allow for comments in the regex, also known as verbose. Comments are indicated by starting with the # character and then escaping with \ |
| xx | Ignore all whitespace / verbose. Similar to x, but whitespace is also ignored inside of character classes |
| s | Known as single line, this enables the dot (.) metacharacter to also match newlines, thus treating the whole string as a single line input |
| \n | Matches a newline character |
| \N | Matches anything other than a newline character |
| \r | Matches a carriage return, Unicode character U+2185 |
| \R | Careful! Matches any Unicode newline sequence |
| \t | Matches a tab character (typically, tab stops happen every eight [8] characters) |
| \0 [zero] | Matches a <i>null</i> character, Unicode character U+2400 |
| \d | Matches any decimal / digit. Equivalent to [0-9] |
| \D | Matches anything other than a decimal / digit |
| \s | Matches any whitespace character (space, tab or newline) |
| \S | Matches any non-whitespace character (anything other space, tab or newline) |
| \w | Matches any word character (any letter, digit or underscore). Equivalent to [a-zA-Z0-9_] |
| \W | Matches any non-word character (anything other than a letter, digit or underscore). Equivalent to [^a-zA-Z0-9_] |
| [abc] | Matches an 'a', 'b' or 'c' character |
| [^abc] | Matches any character except 'a', 'b' or 'c' |
| a b | Alternate match: matches what is before or after , in this case 'a' or 'b' |

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

Let's go through the three new functions.

REGEXEXTRACT

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

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

It has the following three arguments:

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

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

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

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

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

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

Consider the following examples:

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

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

REGEXREPLACE

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

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

where:

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

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

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

Consider the following examples:

| A | B | C |
|---|---|----------------------|
| 1 | Name | |
| 2 | LiamBastick | |
| 3 | | |
| 4 | | |
| Formula | Description | Result |
| =REGEXREPLACE(A2,"([A-Z][a-z]+)([A-Z][a-z]+)","\$2,\$1") | Capturing groups to separate and reorder first and last names | Bastick, Liam |
| 6 | | |
| 7 | | |

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

REGEXTEST

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

REGEXTEST(text, pattern, [case_sensitivity])

where:

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

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

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

Consider the following examples:

| | A | B | C |
|---|---|---------------------------|--------|
| 1 | Data | | |
| 2 | Liam.Bastick@sumproduct.com | | |
| 3 | | | |
| 4 | | | |
| 5 | Formula | Description | Result |
| 6 | =REGEXTEST(A2,"@[([a-zA-Z0-9_+])\.([a-zA-Z0-9_+])"]") | Is this an email address? | TRUE |
| 7 | | | |

| | A | B | C |
|----|--------------------------|---|--------|
| 1 | Data | | |
| 2 | alfalfa | | |
| 3 | | | |
| 4 | | | |
| 5 | Formula | Description | Result |
| 6 | =REGEXTEST(A2,"a") | Does the string contain the letter 'a'? | TRUE |
| 7 | =REGEXTEST(A2,"[a-z]") | Are there any lower case letters? | TRUE |
| 8 | =REGEXTEST(A2,"[A-Z]") | Are there any upper case letters? | FALSE |
| 9 | =REGEXTEST(A2,"[aeiou]") | Are there any vowels? | TRUE |
| 10 | =REGEXTEST(A2,"[0-9]") | Does the string contain digits? | FALSE |
| 11 | | | |

WORD TO THE WISE

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

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

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

Until next month.

The A to Z of Excel Functions: NOMINAL

This function returns the nominal annual interest rate, given the effective annual interest rate and the number of compounding periods per year.




Nominal Interest Rate Formula = $n \times \left[(1 + i)^{\frac{1}{n}} - 1 \right]$



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

NOMINAL(effective_rate, npery)

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

- **effective_rate**: this is required and represents the effective interest rate
- **npery**: this is also required. This is the number of compounding periods per year.

It should be further noted that:

- **npery** is truncated to an integer
- if either argument is nonnumeric, **NOMINAL** returns the #VALUE! error value
- if **effective_rate** ≤ 0 or if **npery** < 1, **NOMINAL** returns the #NUM! error value.

NOMINAL(effective_rate, npery) is related to the **EFFECT (nominal_rate, npery)** function through the identity

$$EFFECT = \left(1 + \frac{Nominal_rate}{Npery}\right)^{Npery} - 1$$

Please see our example below:

| | A | B | C |
|---|-----------------|--|--------|
| 1 | Data | Description | |
| 2 | 5.35% | Effective interest rate | |
| 3 | 4 | Number of compounding periods per year | |
| 4 | | | |
| 5 | | | |
| 6 | Formula | Description | Result |
| 7 | =NOMINAL(A2,A3) | Nominal interest rate, with the terms as given | 5.250% |

These types of calculations give rise to common errors in financial spreadsheets. One common issue is the inability of many analysts to convert an annual interest rate into a monthly or quarterly rate correctly (or vice versa). Sometimes the error is immaterial; other times, it can cause major issues (e.g. for bank forecasts). This isn't so much an Excel issue as a mathematical problem, but it is still relevant to financial modellers and more often than not, it's calculated incorrectly.

The key thing any modeller charged with this exercise must do is read the debt term sheet or deposit account prospectus to see what nominal rate is. Accountants talk about nominal interest rates and such like, but the effective annual rate is the amount of interest expressed as a percentage of the opening debt or cash balance if interest were to be paid and calculated as per the terms of the underlying document.

It's easier said than done. And it doesn't necessarily need either function cited.

Let me demonstrate with an example: consider a loan of \$100 where interest is calculated in arrears on a monthly compounding basis paid quarterly at the end of each quarter. The effective rate is determined to be 12%. The question is, what is the appropriate monthly rate for cashflow calculations?

Who said 1%? That's a common answer, derived as 12% divided by the number of months in a year (12). It sounds good, but if I crunch the numbers I will get the following result:

1. Simple Monthly Rate

Demonstration

Effective Interest Rate % p.a. 12.00%

Rate Used % p.m. 1.00000%

Opening Balance Assumed \$ 100

| | Month 1 | Month 2 | Month 3 | Month 4 | Month 5 | Month 6 | Month 7 | Month 8 | Month 9 | Month 10 | Month 11 | Month 12 |
|------------------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|
| Opening Balance | \$ 100.00 | \$ 101.00 | \$ 102.01 | \$ 100.00 | \$ 101.00 | \$ 102.01 | \$ 100.00 | \$ 101.00 | \$ 102.01 | \$ 100.00 | \$ 101.00 | \$ 102.01 |
| Interest Accrued | \$ 1.00 | \$ 1.01 | \$ 1.02 | \$ 1.00 | \$ 1.01 | \$ 1.02 | \$ 1.00 | \$ 1.01 | \$ 1.02 | \$ 1.00 | \$ 1.01 | \$ 1.02 |
| Interest Paid | \$ - | \$ - | \$ (3.03) | \$ - | \$ - | \$ (3.03) | \$ - | \$ - | \$ (3.03) | \$ - | \$ - | \$ (3.03) |
| Closing Balance | \$ 101.00 | \$ 102.01 | \$ 100.00 | \$ 101.00 | \$ 102.01 | \$ 100.00 | \$ 101.00 | \$ 102.01 | \$ 100.00 | \$ 101.00 | \$ 102.01 | \$ 100.00 |

Total Interest For Year \$ 12.12

Opening Balance Assumed \$ 100

Effective Interest Rate % p.a. 12.12%

On a \$100 balance with interest of 1% monthly paid quarterly, the total interest for the year will be \$12.12 – that’s an effective annual rate of 12.12% rather than 12% (12.00%). This is not correct, as interest is being rolled up at the end of each of the first two months in the quarter and is

attracting further interest, *i.e.* interest is compounding and has not been taken into account by simply dividing the annual rate by the number of months in a year.

So, what about using the compounding formula instead? Interest can then be calculated as

$$=(1+12\%)^{1/12} - 1$$

This equates to 0.9489% per month. This will take into account compounding – well, sort of:

| 1. Compounding Monthly Rate | | | | | | | | | | | | | |
|-----------------------------|--------|---------|---------|---------|---------|---------|---------|---------|---------|---------|----------|----------|----------|
| Demonstration | | | | | | | | | | | | | |
| Effective Interest Rate | % p.a. | 12.00% | | | | | | | | | | | |
| Rate Used | % p.m. | 0.9489% | | | | | | | | | | | |
| Opening Balance Assumed | \$ | 100 | | | | | | | | | | | |
| | | Month 1 | Month 2 | Month 3 | Month 4 | Month 5 | Month 6 | Month 7 | Month 8 | Month 9 | Month 10 | Month 11 | Month 12 |
| Opening Balance | \$ | 100.00 | 100.95 | 101.91 | 100.00 | 100.95 | 101.91 | 100.00 | 100.95 | 101.91 | 100.00 | 100.95 | 101.91 |
| Interest Accrued | \$ | 0.95 | 0.96 | 0.97 | 0.95 | 0.96 | 0.97 | 0.95 | 0.96 | 0.97 | 0.95 | 0.96 | 0.97 |
| Interest Paid | \$ | - | - | (2.87) | - | - | (2.87) | - | - | (2.87) | - | - | (2.87) |
| Closing Balance | \$ | 100.95 | 101.91 | 100.00 | 100.95 | 101.91 | 100.00 | 100.95 | 101.91 | 100.00 | 100.95 | 101.91 | 100.00 |
| Total Interest For Year | \$ | 11.49 | | | | | | | | | | | |
| Opening Balance Assumed | \$ | 100 | | | | | | | | | | | |
| Effective Interest Rate | % p.a. | 11.49% | | | | | | | | | | | |

This hasn’t worked either: interest has been under-accrued (that’s a good word I’ve just made up) as only \$11.49 has been computed for the year – an effective rate of 11.49%.

Here, the calculation is again too simplistic as it does not take into account that the interest is paid quarterly. This means that compounding throughout the year does not occur, hence the shortfall of 0.51%.

The correct formula is:

$$=(1+(\text{Effective Annual Interest Rate} \times \text{Payment Frequency} / \text{Months in Year}))^{1/\text{Payment Frequency}} - 1$$

If payments are made once every three months then there are four payments (equal to **Months in Year / Payment Frequency** or 12 / 3) each year. At these points, compounding stops. Therefore, the interest rate of 12% per annum is effectively 3% per quarter. The compounding formula can then be applied to the quarterly rate to get the monthly rate accordingly:

$$=(1+3\%)^{1/3} - 1$$

This equates to 0.9902%. This is correct, viz.

| 1. Correct Interest Rate | | | | | | | | | | | | | |
|--------------------------|--------|---------|---------|---------|---------|---------|---------|---------|---------|---------|----------|----------|----------|
| Demonstration | | | | | | | | | | | | | |
| Effective Interest Rate | % p.a. | 12.00% | | | | | | | | | | | |
| Rate Used | % p.m. | 0.9902% | | | | | | | | | | | |
| Opening Balance Assumed | \$ | 100 | | | | | | | | | | | |
| | | Month 1 | Month 2 | Month 3 | Month 4 | Month 5 | Month 6 | Month 7 | Month 8 | Month 9 | Month 10 | Month 11 | Month 12 |
| Opening Balance | \$ | 100.00 | 100.99 | 101.99 | 100.00 | 100.99 | 101.99 | 100.00 | 100.99 | 101.99 | 100.00 | 100.99 | 101.99 |
| Interest Accrued | \$ | 0.99 | 1.00 | 1.01 | 0.99 | 1.00 | 1.01 | 0.99 | 1.00 | 1.01 | 0.99 | 1.00 | 1.01 |
| Interest Paid | \$ | - | - | (3.00) | - | - | (3.00) | - | - | (3.00) | - | - | (3.00) |
| Closing Balance | \$ | 100.99 | 101.99 | 100.00 | 100.99 | 101.99 | 100.00 | 100.99 | 101.99 | 100.00 | 100.99 | 101.99 | 100.00 |
| Total Interest For Year | \$ | 12.00 | | | | | | | | | | | |
| Opening Balance Assumed | \$ | 100.00 | | | | | | | | | | | |
| Effective Interest Rate | % p.a. | 12.00% | | | | | | | | | | | |

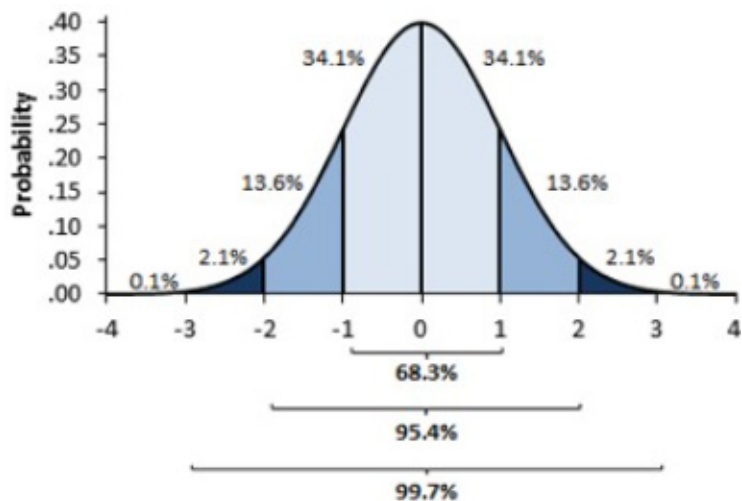
Success!

This is the formula that should be applied to calculate the appropriate monthly interest rate from a quoted effective annual one.

It should be noted that if interest is paid monthly, the formula above reduces to the simple interest rate method (*i.e.* simply divide the rate

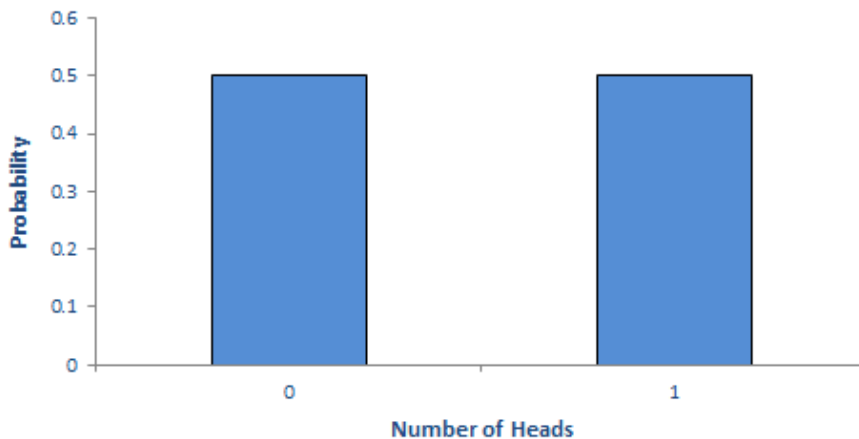
by 12). Similarly, if interest is only paid at the end of the year, the compounding formula is correct too. Essentially, the formula explained above is the “halfway house” between the two.

The A to Z of Excel Functions: NORM.DIST



Imagine I toss an unbiased coin; half of the time it will come down heads, half tails:

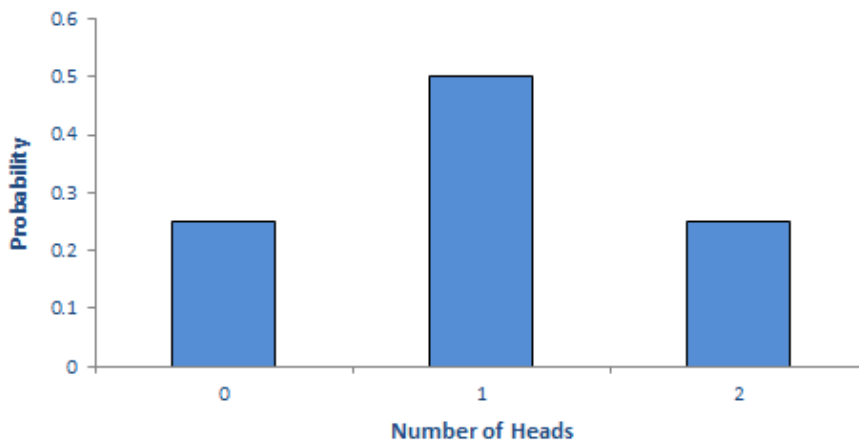
Probability of Heads from One Coin Toss



It is not the most exciting chart I have ever constructed, but it's a start.

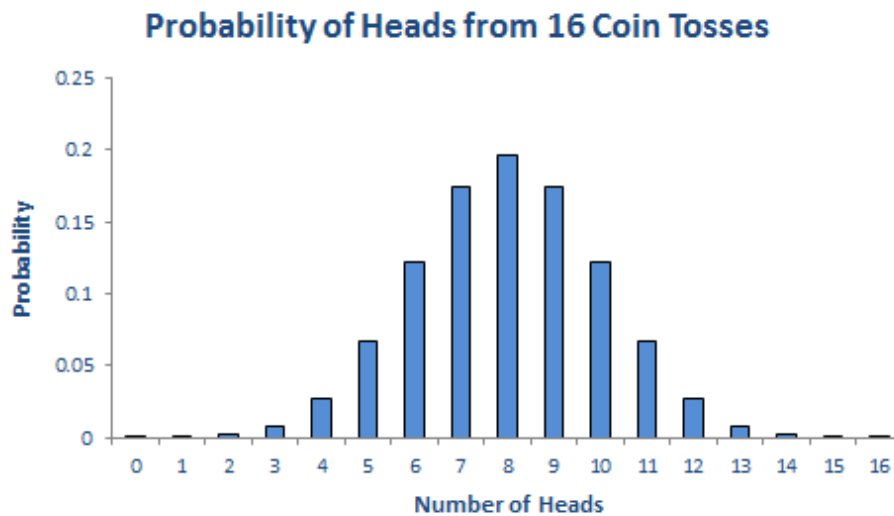
If I toss two coins, I get four possibilities: two Heads, a Head and a Tail, a Tail and a Head, and two Tails.

Probability of Heads from Two Coin Tosses



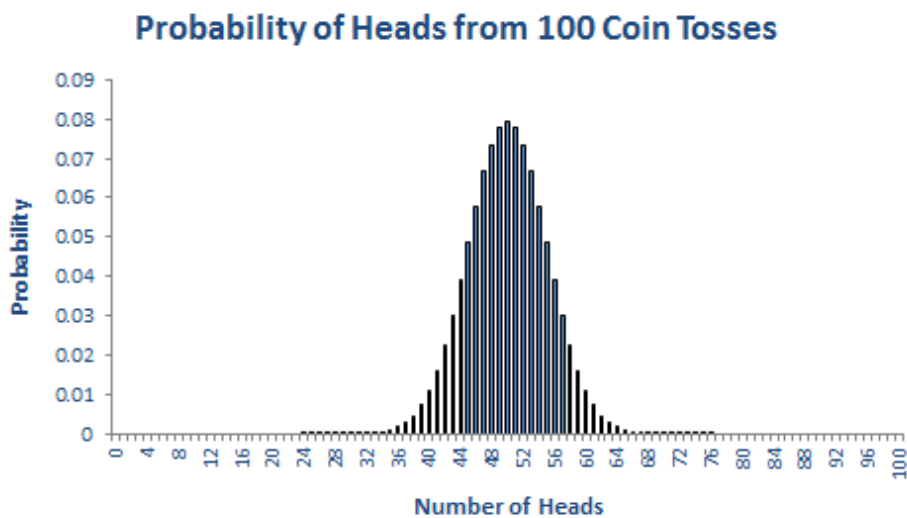
In summary, I should get two heads a quarter of the time, one head half of the time and no heads a quarter of the time. Note that $(1/4) + (1/2) + (1/4) = 1$. These fractions are the probabilities of the events occurring and the sum of all possible outcomes must always add up to 1.

The story is similar if we consider 16 coin tosses say:



Again, if you were to add up all of the individual probabilities, they would total to 1. Notice that in symmetrical distributions (such as this one) it is common for the most likely event (here, eight heads) to be the event at the midpoint.

Of course, why should we stop at 16 coin tosses?



All of these charts represent **probability distributions**, *i.e.* it displays how the probabilities of certain events occurring are distributed. If we can formulate a probability distribution, we can estimate the likelihood of a particular event occurring (*e.g.* probability of precisely 47 heads from 100 coin tosses is 0.0666, probability of less than or equal to 25 heads occurring in 100 coin tosses is 2.82×10^{-7}).

Now, I would like to ask the reader to verify this last chart. Assuming you can toss 100 coins, count the number of heads and record the outcome at one coin toss per second, it shouldn't take you more than 4.0×10^{22} **centuries** to generate every permutation. Even if we were to simulate this experiment using a computer programme capable of generating

many calculations a second it would not be possible. For example, in February 2012, the *Japan Times* announced a new computer that could compute 10,000,000,000,000 calculations per second. If we could use this computer, it would only take us a mere 401,969 years to perform this computation. Sorry, but I can't afford the electricity bill.

Let's put this all into perspective. All I am talking about here is considering 100 coin tosses. If only business were that *simple*. Potential outcomes for a business would be much more complex. Clearly, if we want to consider all possible outcomes, we can only do this using some sampling technique based on understanding the underlying probability distributions.

Probability Distributions

If I plotted charts for 1,000 or 10,000 coin tosses similar to the above, I would generate similarly shaped distributions. This classic distribution which only allows for two outcomes is known as the **Binomial distribution** and is regularly used in probabilistic analysis.

The 100 coin toss chart shows that the average (or 'expected' or 'mean') number of heads here is 50. This can be calculated using a weighted

average in the usual way. The 'spread' of heads is clearly quite narrow (tapering off very sharply at less than 40 heads or greater than 60). This spread is measured by statisticians using a measure called **standard deviation** which is defined as the square root of the average value of the square of the difference between each possible outcome and the mean, *i.e.*

$$\sigma = \sqrt{\frac{1}{N} \sum_{i=1}^N (x_i - \mu)^2}$$

where: σ = standard deviation

N = total number of possible outcomes

Σ = summation

x_i = each outcome event (from first x_1 to last x_n)

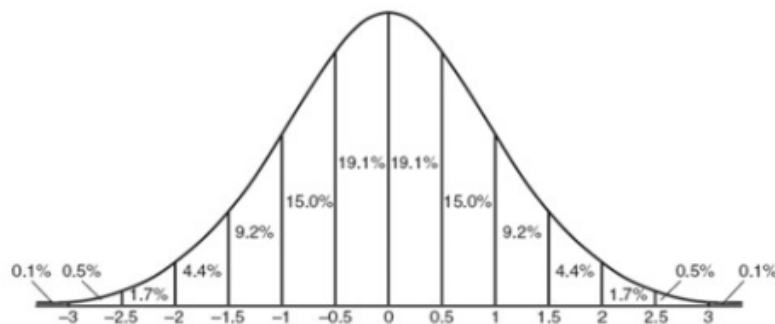
μ = mean or average

The Binomial distribution is not the most common distribution used in probability analysis: that honour belongs to the **Gaussian or Normal distribution**:



The Central Limit Theorem states that the sampling distribution of the sample means approaches a normal distribution as the sample size gets larger — no matter what the shape of the population distribution. This is why this distribution is so important in probability and statistics.

Generated by a complex mathematical formula, this distribution is defined by specifying the **mean** and **standard deviation** (see above). The Normal distribution's population is spread as follow:



i.e. 68% of the population is within one standard deviation of the mean, 95% within two standard deviations and 99.7% within three standard deviations.

The formula for the Normal distribution is given by

$$p(x) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(x - \mu)^2}{2\sigma^2}\right)$$

and Excel has the function **NORM.DIST** to calculate it. Its syntax is:

NORM.DIST(x, mean, standard_deviation, cumulative)

NORM.DIST has the following arguments:

- **x**: this is required and represents the value for which you want the distribution
- **mean**: this is also required. This is the arithmetic mean of the distribution
- **standard_deviation**: this is required. This denotes the standard deviation of the distribution
- **cumulative**: the final argument is required too. This denotes a logical value that determines the form of the function. If **cumulative** is TRUE, **NORM.DIST** returns the cumulative distribution function; if FALSE, it returns the probability density function.

It should be further noted that:

- since Excel 2007, Microsoft has updated many of its statistical functions. To provide backward compatibility, they changed the names of their updated functions by adding periods within the name. This function supersedes **NORMDIST**. If you are new to both of these functions, I would suggest using this variant as Microsoft has advised it may not support the former function in future versions of Excel
- if **mean** or **standard_deviation** is nonnumeric, **NORM.DIST** returns the #VALUE! error value
- if **standard_deviation** ≤ 0, **NORM.DIST** returns the #NUM! error value
- if **mean** = 0, **standard_deviation** = 1, and **cumulative** = TRUE, **NORM.DIST** returns the standard normal distribution, **NORM.S.DIST**
- the equation for the normal density function (**cumulative** = FALSE) is:

$$f(x; \mu, \sigma) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\left(\frac{(x-\mu)^2}{2\sigma^2}\right)}$$

- when **cumulative** = TRUE, the formula is the integral from negative infinity to **x** of the given formula (*above*).

Please see our example below:

| | A | B | C |
|----|----------------------------|--|-------------|
| 1 | Data | Description | |
| 2 | 20 | Value for which you want the distribution | |
| 3 | 15 | Arithmetic mean of the distribution | |
| 4 | 5 | Standard deviation of the distribution | |
| 5 | | | |
| 6 | | | |
| 7 | Formula | Description | Result |
| 8 | =NORM.DIST(A2,A3,A4,TRUE) | Cumulative distribution function for the terms above | 0.841344746 |
| 9 | =NORM.DIST(A2,A3,A4,FALSE) | Probability mass function for the terms above | 0.048394145 |
| 10 | | | |

The A to Z of Excel Functions: NORM.INV



From above, the formula for the Normal distribution is given by

$$p(x) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(x - \mu)^2}{2\sigma^2}\right)$$

However, sometimes you may wish to determine **x** for a given probability, known as the inverse function. Excel has the function **NORM.INV** to calculate the inverse of the normal cumulative distribution for the specified mean and standard deviation.

Its syntax is:

NORM.INV(probability, mean, standard_deviation)

NORM.INV has the following arguments:

- **probability**: this is required and represents a probability corresponding to the normal distribution
- **mean**: this is required and denotes the arithmetic mean of the distribution
- **standard_deviation**: this last argument is also required. This is the standard deviation of the distribution.

It should be further noted that:

- since Excel 2007, Microsoft has updated many of its statistical functions. This function supersedes **NORMINV**. If you are new to both of these functions, I would suggest using **NORM.INV** as Microsoft has advised it may not support the original function in future versions of Excel
- if any argument is nonnumeric, **NORM.INV** returns the #VALUE! error value
- if **probability** ≤ 0 or if ≥ 1, **NORM.INV** returns the #NUM! error value
- if **standard_deviation** ≤ 0, **NORM.INV** returns the #NUM! error value
- if **mean** = 0 and **standard_deviation** = 1, **NORM.INV** uses the standard normal distribution, **NORM.S.INV**
- given a value for **probability**, **NORM.INV** seeks that value **x** such that **NORM.DIST(x, mean, standard_deviation, TRUE) = probability**. Thus, precision of **NORM.INV** depends upon the precision of **NORM.DIST**.

Please see our example below:

| | A | B | C |
|---|---------------------|--|--------|
| 1 | Data | Description | |
| 2 | 0.841344746 | Probability corresponding to the normal distribution | |
| 3 | 15 | Arithmetic mean of the distribution | |
| 4 | 5 | Standard deviation of the distribution | |
| 5 | | | |
| 6 | | | |
| 7 | Formula | Description | Result |
| 8 | =NORM.INV(A2,A3,A4) | Cumulative distribution function for the terms above | 20 |

Extending the Idea to Simulations Analysis

For any given distribution, we cannot model every possible combination / permutation of outcomes. The aim is to analyse a representative sample based on a known, or assumed, probability distribution.

There are various ways to sample, with the most popular approach being the "Monte Carlo" method, which involves picking data randomly (i.e. using no stratification or bias). Excel's **RAND()** function picks a number between 0 and 1 randomly, which is very useful as cumulative probabilities can only range between 0 and 1.

NORM.INV(probability, mean, standard_deviation) returns the value **x**

such that the cumulative probability specified (**probability**) represents the observed value of a Normal random variable with specified **mean** and **standard_deviation** is less than or equal to **x**. In essence, this is the inverse function of **NORM.DIST(x, mean, standard_deviation, TRUE)**.

Therefore, we can get Excel to pick a random number between zero (0) and one (1), and for a given mean and standard deviation, generate a particular outcome appropriate to the probability distribution specified, which can then be used in the model as in the following illustration:

| | Mean | Standard Deviation | Value Used |
|-----------------------------|-------|--------------------|------------|
| Sales (US\$) | 1,000 | 175 | 709 |
| Foreign Exchange (US\$:A\$) | 0.92 | 0.05 | 0.99 |
| Gross Margin (%) | 40.0% | 2.0% | 36.8% |

=NORM.INV(RAND(),Mean_Sales,SD_Sales)

The mean and standard deviation are easy to calculate – simply list all of your historical data and use the Excel functions **AVERAGE** for mean and **STDEV.S** for the standard deviation.

Here, three variables, **Sales**, **Foreign Exchange** and **Gross Margin** all employ the **NORM.INV** function to generate the assumptions needed to calculate the Gross Profit. We can run the simulation a given number of times by running a simple one-dimensional Data Table.

The actual approach is a little crafty though:

| I31 | | | | | | fx {=TABLE(,E30)} | | | | | |
|--------------|----|---|---|---|-----------------------------|-----------------------------|------------------|--------------------|-------------------------------------|--|--|
| A | B | C | D | E | F | G | H | I | J | | |
| Calculations | | | | | | | | | | | |
| 18 | | | | | Sales (US\$) | | | | 1096 | | |
| 19 | | | | | Foreign Exchange (US\$:A\$) | | | | 0.86 | | |
| 20 | | | | | Sales (A\$) | | | | 1270 | | |
| 21 | | | | | Gross Margin (%) | | | | 38.0% | | |
| 23 | | | | | Gross Profit (A\$) | | | | 482 | | |
| Data Table | | | | | | | | | | | |
| 28 | | | | | Data Table Switch | | | | <input checked="" type="checkbox"/> | | |
| 30 | | | | | Sales (US\$) | Foreign Exchange (US\$:A\$) | Gross Margin (%) | Gross Profit (A\$) | | | |
| 31 | 1 | | | | 922 | 0.87 | 40.0% | 425 | | | |
| 32 | 2 | | | | 849 | 0.93 | 41.0% | 373 | | | |
| 33 | 3 | | | | 928 | 0.87 | 41.1% | 438 | | | |
| 34 | 4 | | | | 1,037 | 0.84 | 43.3% | 531 | | | |
| 35 | 5 | | | | 1,136 | 1.00 | 42.7% | 485 | | | |
| 36 | 6 | | | | 886 | 0.95 | 39.4% | 367 | | | |
| 37 | 7 | | | | 1,254 | 0.94 | 43.1% | 575 | | | |
| 38 | 8 | | | | 1,039 | 0.84 | 39.9% | 491 | | | |
| 39 | 9 | | | | 1,103 | 0.88 | 39.0% | 492 | | | |
| 40 | 10 | | | | 1,112 | 0.95 | 43.0% | 502 | | | |
| 41 | 11 | | | | 799 | 0.98 | 42.1% | 343 | | | |

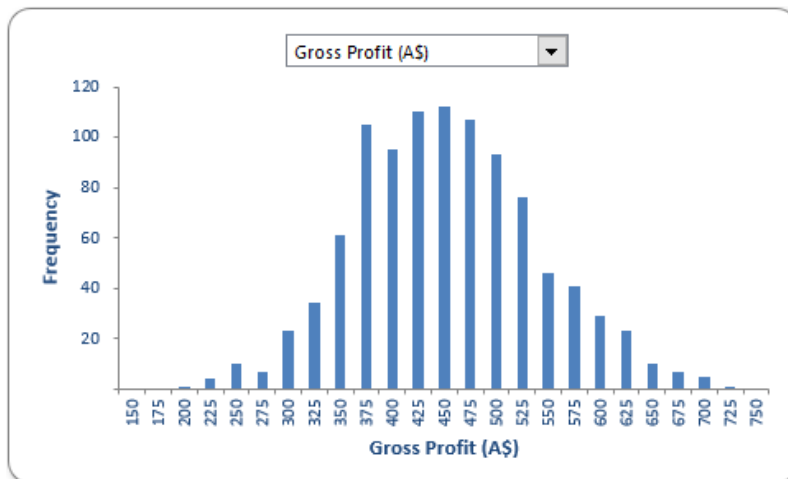
Since the variables use the **RAND** function to generate random numbers, each time the end user presses **ENTER** or **F9**, the variables will recalculate (this quality is known as 'volatility'). I have created a Data Table (**ALT + D + T**) to create multiple trials (the headings are actually the outputs required using clever number formatting to disguise what they are). Once dynamic arrays become Generally Available, this technique will become even simpler.

Since each Data Table entry causes the model to recalculate for each column input, the values will change automatically. On this basis, note

that the column input cell in the example above refers to **E30** (the cell highlighted in yellow) which is unused by any formula on the spreadsheet.

The example here has 1,000 rows (*i.e.* 1,000 simulations) and you can refer back to this month's Charts and Dashboards article. Since the variables have been generated randomly, this is a simple Monte Carlo simulation – no fancy software or macros required!

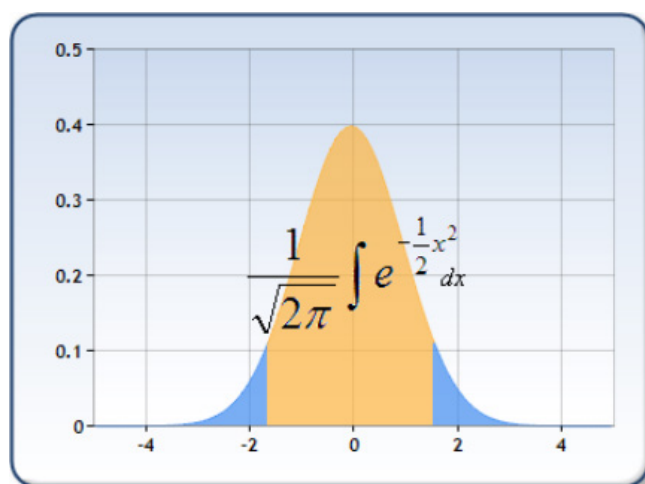
It only requires a quick collation step to summarise the outputs graphically:



Please see our example below:

| | A | B | C |
|---|--------------------------------|---|--------------------|
| 1 | Formula | Description | Result |
| 2 | =NORM.S.DIST(1.5,TRUE) | Normal cumulative distribution function at 1.5 | 0.933192799 |
| 3 | =NORM.S.DIST(1.5,FALSE) | Normal probability distribution function at 1.5 | 0.129517596 |

The A to Z of Excel Functions: NORM.S.INV



$$p(x) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$$

As earlier, sometimes you may wish to determine **x** for a given probability, known as the inverse function. And sometimes you want it specifically to be the inverse of the standard normal cumulative distribution (*i.e.* with a mean of zero [0] and a standard deviation of one [1]). Excel has the function **NORM.S.INV** to calculate the inverse of this standard normal cumulative.

Its syntax is:

NORM.S.INV(probability)

NORM.S.INV has the following arguments:

- **probability**: this is required and represents a probability corresponding to the normal distribution.

It should be further noted that:

- since Excel 2007, Microsoft has updated many of its statistical functions. This function supersedes **NORMSINV**. If you are new to both of these functions, I would suggest using **NORM.S.INV** as Microsoft has advised it may not support the original function in future versions of Excel
- if **probability** is nonnumeric, **NORM.S.INV** returns the **#VALUE!** error value
- if **probability** ≤ 0 or if ≥ 1, **NORM.S.INV** returns the **#NUM!** error value.

Please see our example below:

| | A | B | C |
|---|-----------------------------|--|---------------|
| 1 | Formula | Description | Result |
| 2 | =NORM.S.INV(0.77337) | Inverse of the standard normal cumulative distribution, with a probability of 0.77337. | 0.7500 |

The A to Z of Excel Functions: NORMDIST



It's getting boring:

$$p(x) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$$

Excel has the function **NORMDIST** to calculate it. Its syntax is:

NORMDIST(x, mean, standard_deviation, cumulative)

NORMDIST has the following arguments:

- **x**: this is required and represents the value for which you want the distribution
- **mean**: this is also required. This is the arithmetic mean of the distribution
- **standard_deviation**: this is required. This denotes the standard deviation of the distribution
- **cumulative**: the final argument is required too. This denotes a logical value that determines the form of the function. If **cumulative** is TRUE, **NORMDIST** returns the cumulative distribution function; if FALSE, it returns the probability density function.

It should be further noted that:

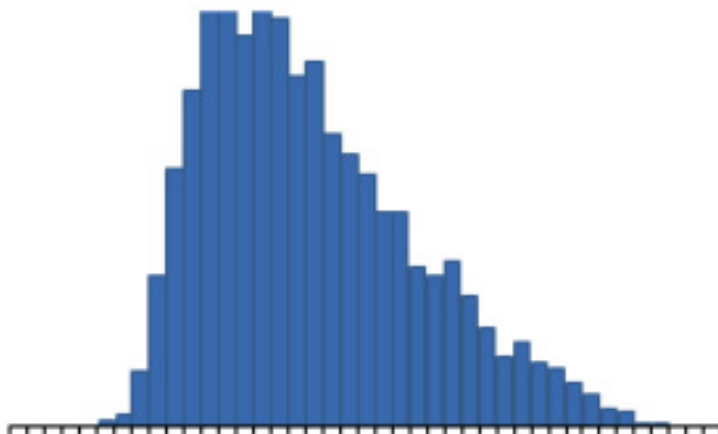
- since Excel 2007, Microsoft has updated many of its statistical functions. This is an older function which has been superseded by **NORM.DIST**. If you are new to both of these functions, I would suggest using **NORM.DIST** as Microsoft has advised it may not support the original function in future versions of Excel
- if **mean** or **standard_deviation** is nonnumeric, **NORMDIST** returns the #VALUE! error value
- if **standard_deviation** ≤ 0, **NORMDIST** returns the #NUM! error value
- if **mean** = 0, **standard_deviation** = 1, and **cumulative** = TRUE, **NORMDIST** returns the standard normal distribution, **NORMSDIST** the equation for the normal density function (**cumulative** = FALSE) is:

$$f(x; \mu, \sigma) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\left(\frac{(x-\mu)^2}{2\sigma^2}\right)}$$

- when **cumulative** = TRUE, the formula is the integral from negative infinity to **x** of the given formula (*above*).

| | A | B | C |
|---|---------------------------|--|-------------|
| 1 | Data | Description | |
| 2 | 20 | Value for which you want the distribution | |
| 3 | 15 | Arithmetic mean of the distribution | |
| 4 | 5 | Standard deviation of the distribution | |
| 5 | | | |
| 6 | | | |
| 7 | Formula | Description | Result |
| 8 | =NORMDIST(A2,A3,A4,TRUE) | Cumulative distribution function for the terms above | 0.841344746 |
| 9 | =NORMDIST(A2,A3,A4,FALSE) | Probability mass function for the terms above | 0.048394145 |

The A to Z of Excel Functions: NORMINV



Blah blah blah blah blah:

$$p(x) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(x - \mu)^2}{2\sigma^2}\right)$$

Again, sometimes you may wish to determine **x** for a given probability, known as the inverse function. Excel has the function **NORMINV** to calculate the inverse of the normal cumulative distribution for the specified mean and standard deviation.

Its syntax is:

NORMINV(probability, mean, standard_deviation)

NORMINV has the following arguments:

- **probability**: this is required and represents a probability corresponding to the normal distribution
- **mean**: this is required and denotes the arithmetic mean of the distribution
- **standard_deviation**: this last argument is also required. This is the standard deviation of the distribution.

It should be further noted that:

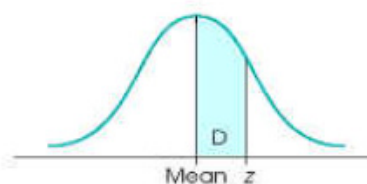
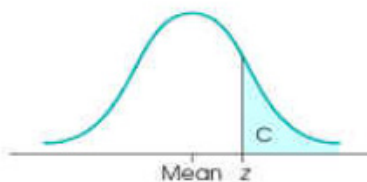
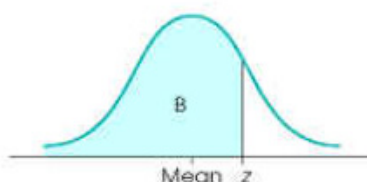
- since Excel 2007, Microsoft has updated many of its statistical functions. This is an older function which has been superseded by **NORM.INV**. If you are new to both of these functions, I would suggest using **NORM.INV** as Microsoft has advised it may not support the original function in future versions of Excel
- if any argument is nonnumeric, **NORMINV** returns the **#VALUE!** error value
- if **probability** ≤ 0 or if ≥ 1, **NORMINV** returns the **#NUM!** error value
- if **standard_deviation** ≤ 0, **NORMINV** returns the **#NUM!** error value
- if **mean** = 0 and **standard_deviation** = 1, **NORMINV** uses the standard normal distribution, **NORMSINV**
- given a value for **probability**, **NORMINV** seeks that value **x** such that **NORMDIST(x, mean, standard_deviation, TRUE) = probability**. Thus,
- precision of **NORMINV** depends upon the precision of **NORMDIST**.

Please see our example below:

| | A | B | C |
|---|--------------------|--|--------|
| 1 | Data | Description | |
| 2 | 0.841344746 | Probability corresponding to the normal distribution | |
| 3 | 15 | Arithmetic mean of the distribution | |
| 4 | 5 | Standard deviation of the distribution | |
| 5 | | | |
| 6 | | | |
| 7 | Formula | Description | Result |
| 8 | =NORMINV(A2,A3,A4) | Cumulative distribution function for the terms above | 20 |

The A to Z of Excel Functions: NORMSDIST

| (A) z | (B) Proportion in Body | (C) Proportion in Tail | (D) Proportion Between Mean and z |
|----------|------------------------------|------------------------------|--|
| 0.00 | .5000 | .5000 | .0000 |
| 0.01 | .5040 | .4960 | .0040 |
| 0.02 | .5080 | .4920 | .0080 |
| 0.03 | .5120 | .4880 | .0120 |
| ... | | | |
| 0.21 | .5832 | .4168 | .0832 |
| 0.22 | .5871 | .4129 | .0871 |
| 0.23 | .5910 | .4090 | .0910 |
| 0.24 | .5948 | .4052 | .0948 |
| 0.25 | .5987 | .4013 | .0987 |
| 0.26 | .6026 | .3974 | .1026 |
| 0.27 | .6064 | .3936 | .1064 |
| 0.28 | .6103 | .3897 | .1103 |
| 0.29 | .6141 | .3859 | .1141 |
| 0.30 | .6179 | .3821 | .1179 |
| 0.31 | .6217 | .3783 | .1217 |
| 0.32 | .6255 | .3745 | .1255 |
| 0.33 | .6293 | .3707 | .1293 |
| 0.34 | .6331 | .3669 | .1331 |



This may be simplified in the case of what is known as the **standard Normal distribution** which has a mean of zero [0] and a standard deviation of one [1]. Excel has the function **NORMSDIST** to calculate it. You should use this function in place of a table of standard normal curve areas.

Its syntax is:

NORMSDIST(z)

NORMSDIST has the following argument:

- **z**: this is required and represents the value for which you want the distribution.

It should be further noted that:

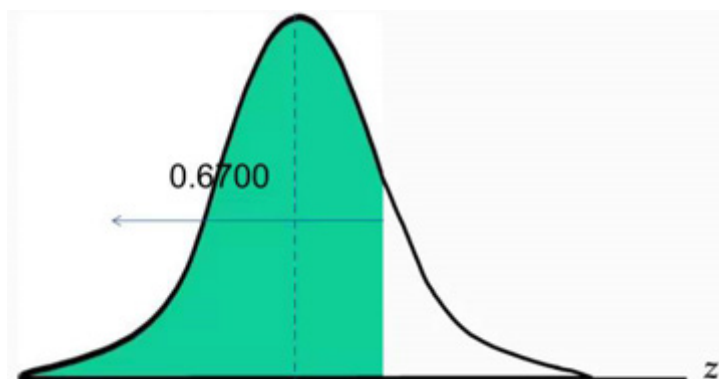
- since Excel 2007, Microsoft has updated many of its statistical functions. This is an older function which has been superseded by **NORM.S.DIST**. If you are new to both of these functions, I would suggest using **NORM.S.DIST** as Microsoft has advised it may not support the original function in future versions of Excel
- if **z** is nonnumeric, **NORMSDIST** returns the **#VALUE!** error value
- the equation for the standard normal density function is:

$$f(z) = \frac{1}{\sqrt{2\pi}} e^{-\frac{z^2}{2}}$$

Please see our penultimate example below:

| | A | B | C |
|---|------------------------|--|--------------------|
| 1 | Formula | Description | Result |
| 2 | =NORMSDIST(1.5) | Normal cumulative distribution function at 1.5 | 0.933192799 |

The A to Z of Excel Functions: NORMSINV



The formula for the Normal distribution is given by

$$p(x) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$$

However, sometimes you may wish to determine x for a given probability, known as the inverse function. And sometimes you want it specifically to be the inverse of the standard normal cumulative distribution (*i.e.* with a mean of zero [0] and a standard deviation of one [1]). Excel has the function **NORMSINV** to calculate the inverse of this standard normal cumulative.

Its syntax is:

NORMSINV(probability)

NORMSINV has the following arguments:

- **probability**: this is required and represents a probability corresponding to the normal distribution.

It should be further noted that:

- since Excel 2007, Microsoft has updated many of its statistical functions. This is an older function which has been superseded by **NORM.S.INV**. If you are new to both of these functions, I would suggest using **NORM.S.INV** as Microsoft has advised it may not support the original function in future versions of Excel
- if **probability** is nonnumeric, **NORMSINV** returns the #VALUE! error value
- if **probability** ≤ 0 or if ≥ 1 , **NORMSINV** returns the #NUM! error value.

Please see our final example for this month below:

| | A | B | C |
|---|---------------------------|--|---------------|
| 1 | Formula | Description | Result |
| 2 | =NORMSINV(0.77337) | Inverse of the standard normal cumulative distribution, with a probability of 0.77337. | 0.7500 |

More Excel Functions next month.

Beat the Boredom Suggested Solution

For this newsletter's challenge, we just wanted you to convert numbers stored as Text into Numbers. Easy, yes?

The Challenge

Sometimes, you may need to deal with numbers in the text format. As you may know, there are various ways to convert them into "real" numbers. This challenge was a little trickier, but you may come across it some day when using Excel.

One of our clients recently came up with this challenge. They provided us with a list of numbers copied from their management information system into Excel. It was in a text format that needed to be converted into numbers.

We have made up a similar list for this challenge that you can download here: <https://view.officeapps.live.com/op/view.aspx?src=https%3A%2F%2Fwww.sumproduct.com%2Fassets%2Fblog-pictures%2F2021%2Ffff-mmm%2Foct%2Ffff%2Fsp-number-conversion--question.xlsm&wdOrigin=BROWSELINK>

The aim was to write a **formula** that could be copied down to convert the list of numbers from text format into a numerical data type, like the ones on the right that you can see below:

| Number | Solution |
|--------|----------|
| 0009 | 9 |
| 0011 | 11 |
| 0012 | 12 |
| 0013 | 13 |
| 0017 | 17 |
| 0023 | 23 |
| 0025 | 25 |
| 0027 | 27 |
| 0028 | 28 |
| 0032 | 32 |
| 0039 | 39 |
| 0045 | 45 |
| 0047 | 47 |

There were some constraints:

- there should be no “helper” column
- this was a formula challenge: no Power Query / Get & Transform, Text to Columns or Flash Fill
- the formula needed to be dynamic enough so that if we change from four digits to a different length, the formula must still work.

Suggested Solution

As you may know, to clean and convert Text to Number, there are several common ways to apply such as:

- functions (e.g. **TRIM**, **CLEAN**, **N** and **VALUE**)
- coercing techniques (e.g. using *1, +0 or -- in your formula)
- Paste Special, multiplying by one [1] or adding zero [0]
- Text to Columns
- Flash Fill
- etc.

However, after we had tried all the options above, we realised that they could not convert properly and / or did not meet the requirements of the question.

Before explaining our solutions, we will clarify how we came up with them first.

Brainstorming

After using some common ways as mentioned above, we tried to find out what kind of characters in the strings that did not allow us to convert Text to Numbers properly.

Firstly, we used **LEN(text)** function which returns number of characters in a text string. According to the result below, all strings have more than four characters. Hence, there must be other characters beside the four digit numbers that we could not see.

| Number | LEN |
|--------|-----|
| 0009 | 5 |
| 0011 | 6 |
| 0012 | 6 |
| 0013 | 6 |
| 0017 | 6 |
| 0023 | 6 |
| 0025 | 6 |
| 0027 | 6 |
| 0028 | 6 |
| 0032 | 6 |
| 0039 | 6 |
| 0045 | 6 |
| 0047 | 6 |

Secondly, we wanted to identify the positions of those unwanted characters. Therefore, we split each character of the text strings into different cells by using a combination of **MID**, **SEQUENCE** and **LEN** functions.

While you may be familiar with **MID** and **LEN** functions, **SEQUENCE** is one of the Dynamic Arrays that are only available in Excel for Microsoft 365 / online versions. The formula is as follows:

=MID(Numbers[@Number],SEQUENCE(1,LEN(Numbers[@Number])),1)

In the formula above:

- **Numbers** is the name of the table in the question. Therefore, **Numbers[@Number]** specifies one row of column **Number**, where the above formula is located
- **SEQUENCE(1,LEN(Numbers[@Number]))** will help us create a horizontal list of consecutive numbers from 1 to the last number which is equal to the length of the string. For example:

| | A | B | C | D | E |
|---|---|---|---|---|---|
| 1 | 1 | 2 | 3 | 4 | 5 |

- The **MID** function will then extract each character of a string with the starting point one by one, equal to the number list created by **SEQUENCE** above.

After you copy down the formula, the result should look like this:

| | | | | | |
|--|---|---|---|---|--|
| | 0 | 0 | 0 | 9 | |
| | 0 | 0 | 1 | 1 | |
| | 0 | 0 | 1 | 2 | |
| | 0 | 0 | 1 | 3 | |
| | 0 | 0 | 1 | 7 | |
| | 0 | 0 | 2 | 3 | |
| | 0 | 0 | 2 | 5 | |
| | 0 | 0 | 2 | 7 | |
| | 0 | 0 | 2 | 8 | |
| | 0 | 0 | 3 | 2 | |
| | 0 | 0 | 3 | 9 | |
| | 0 | 0 | 4 | 5 | |
| | 0 | 0 | 4 | 7 | |

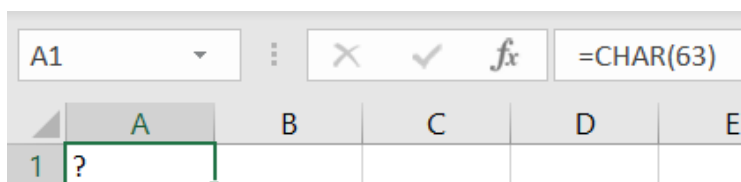
We can see that there are invisible characters at the beginning and the end of the strings.

Thirdly, we needed to identify what those invisible characters were. We tried applying **CODE** and **CHAR** functions to our dynamic arrays above first. **CODE** allows us to return a numeric code for the first character in a text string. **CHAR** translates that code back into a character.

The result is as follows: the code of these “invisible” characters is apparently 63, viz.

| | | | | | |
|----|----|----|----|----|----|
| 63 | 48 | 48 | 48 | 57 | |
| 63 | 48 | 48 | 49 | 49 | 63 |
| 63 | 48 | 48 | 49 | 50 | 63 |
| 63 | 48 | 48 | 49 | 51 | 63 |
| 63 | 48 | 48 | 49 | 55 | 63 |
| 63 | 48 | 48 | 50 | 51 | 63 |
| 63 | 48 | 48 | 50 | 53 | 63 |
| 63 | 48 | 48 | 50 | 55 | 63 |
| 63 | 48 | 48 | 50 | 56 | 63 |
| 63 | 48 | 48 | 51 | 50 | 63 |
| 63 | 48 | 48 | 51 | 57 | 63 |
| 63 | 48 | 48 | 52 | 53 | 63 |
| 63 | 48 | 48 | 52 | 55 | 63 |

However, when we used **CHAR** function to return a character, it returned a question mark (?). It seemed to be incorrect as “?” is still visible. Something was not right!



We realised that **CODE** and **CHAR** can only work with characters with number codes between 1 and 255. Any codes above 255 cannot be identified so Excel returns a question mark (?). Great. Sorry, I meant “grate”.

Then, we used **UNICODE** and **UNICHAR** instead which have similar usage but support higher number of characters. The result is as follows:

| | | | | | |
|------|----|----|----|----|------|
| 8237 | 48 | 48 | 48 | 57 | |
| 8237 | 48 | 48 | 49 | 49 | 8236 |
| 8237 | 48 | 48 | 49 | 50 | 8236 |
| 8237 | 48 | 48 | 49 | 51 | 8236 |
| 8237 | 48 | 48 | 49 | 55 | 8236 |
| 8237 | 48 | 48 | 50 | 51 | 8236 |
| 8237 | 48 | 48 | 50 | 53 | 8236 |
| 8237 | 48 | 48 | 50 | 55 | 8236 |
| 8237 | 48 | 48 | 50 | 56 | 8236 |
| 8237 | 48 | 48 | 51 | 50 | 8236 |
| 8237 | 48 | 48 | 51 | 57 | 8236 |
| 8237 | 48 | 48 | 52 | 53 | 8236 |
| 8237 | 48 | 48 | 52 | 55 | 8236 |

As you can see above, characters 63 have turned into “Left-To-Right Override” (8237) and “Pop Directional Formatting” (8236) which are both non-printable characters. The rest is still the same.

The mystery was SOLVED!

In summary, we need to eliminate characters 8237 and 8236 before converting the text strings to numbers.

Returning to the Suggested Solution

The main function in our solutions is **SUBSTITUTE**. It will be used to remove characters 8237 and 8236 from the text strings. Thus, in order to specify the two characters 8237 and 8236 in the formula, we have two options:

Option 1: If you use Excel 2013 onwards, you can use the **UNICHAR** function to return the characters from their codes (i.e. 8237 and 8236). The formula is as below:

=SUBSTITUTE(SUBSTITUTE(Numbers[@Number],UNICHAR(8237),""),UNICHAR(8236),""))+0

Therefore, we use two **SUBSTITUTE** functions and two **UNICHAR** functions to replace those unwanted characters with blank, "". Then, we add zero [+0] to force the remaining strings to become numbers.

Option 2: This option is for all current Excel versions. Please note that as characters 8237 and 8236 are non-printable characters, you will not see them in the formula below. Please copy the formula (not typing it again).

=SUBSTITUTE(SUBSTITUTE(Numbers[@Number],"",""),"", "")+0

More next month.

Upcoming SumProduct Training Courses

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

Key Strokes

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

| Keystroke | What it does |
|------------|--|
| CTRL + F1 | Excel 2007: Show / Hide Ribbon; Excel 2003: Show / Hide Task Pane |
| CTRL + F2 | Excel 2007: Print; Excel 2003: SHOW.INFO() |
| CTRL + F3 | Excel 2007: Open Name Manager; Excel 2003: Open Define Names Dialog Box |
| CTRL + F4 | Close Window |
| CTRL + F5 | Restore Window |
| CTRL + F6 | Next Window / Workbook |
| CTRL + F7 | Move Window |
| CTRL + F8 | Size Window |
| CTRL + F9 | Minimise Window |
| CTRL + F10 | Toggle Maximised / Restored |
| CTRL + F11 | Insert New Macro Sheet |
| CTRL + F12 | Open |

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

Our Services

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

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

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

Link to Others

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

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

Any Questions?

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

Training

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

Check out our more popular courses in our training brochure:



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

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

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