

SumProduct

NEWSLETTER #99 - February 2021

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

98..

99

..100

Last time in double digits

for the newsletter – a big landmark is looming ever closer into view! 2021 is definitely up and running now, and we keep chugging on, whether working from the office or home! We hope you are trying to make the most of this brave new world too.

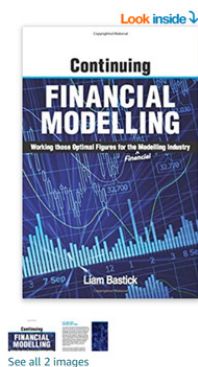
There is no big Power BI update this month (Microsoft decided they would have some time off), but it doesn't mean this month's newsletter is flimsy. We have more news on the second financial modelling book, details of an Excel roadmap update and more on **LAMBDA**. Then there are the regulars: there is another Beat the Boredom Challenge, Visual Basics, Power Pivot Principles, Power Query Pointers, Keyboard Shortcuts and some rather "fishy" A to Z of Excel Functions.

Happy reading and remember: stay safe, stay happy, stay healthy.

Liam Bastick, Managing Director, SumProduct



Continuing Financial Modelling Book Update



Continuing Financial Modelling: Working Those Optimal Figures For the (Financial) Modelling Industry Paperback – 16 January 2021

by Liam Bastick (Author)

★★★★★ 2 ratings

See all formats and editions

Kindle

\$14.28

Read with Our Free App

Paperback

\$53.36

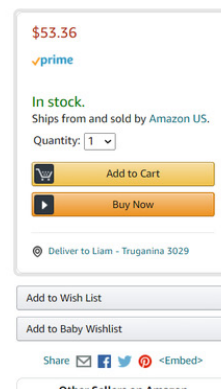
4 New from \$53.36

FREE delivery: 28 Jan - 1 Feb

Fastest delivery: 22 - 26 Jan

US imports may differ from local products. Additional terms apply. [Learn More.](#)

This book is aimed at those who wish to advance their knowledge and expertise in financial modelling by addressing common problems that occur day to day in the world of business/decision analyses, forecasting, and valuations. Building on the sister book, *An Introduction to Financial Modelling*, this book begins where the other ends considering typical issues and traps in cashflow forecasting, inventory modelling, depreciation calculations, debt sculpting, rolling budgets and charts, and valuation construction, to name just a few relevant topics.



At last! Plans have been scuppered with what has been going on around the world, but here at SumProduct, we are finally delighted to advise both our readers that Continuing Financial Modelling (released to Amazon Kindle back in September last year) has been given a physical release on Amazon (one step at a time...).

It has been frustrating getting this book released, but we believe everyone understands there are bigger issues at play in the world at the moment! For all those who have asked, when is the book coming out in paperback? Finally! We are able to deliver an answer.

Happy reading.

Excel Roadmap Updates



"So where did you say Flight Simulator was again..?"

Microsoft is planning to add support for co-authoring protected Excel files. The new feature will allow users to protect Excel files using passwords yet still be able to share them with others who may edit these files without needing to remove the protection.

This feature was recently added to Microsoft Roadmap and will be available for both Microsoft Excel online app and Excel desktop app (the announcement is silent on whether Microsoft plans to add the feature to Android and iOS apps as well). Currently, Excel users can protect files, but they will need to remove the protection if they want to add a co-author. With the new feature, users will be able to add new co-authors without removing protection making it easier and safer to share confidential or sensitive information. This capability is expected to roll out by the end of June 2021.

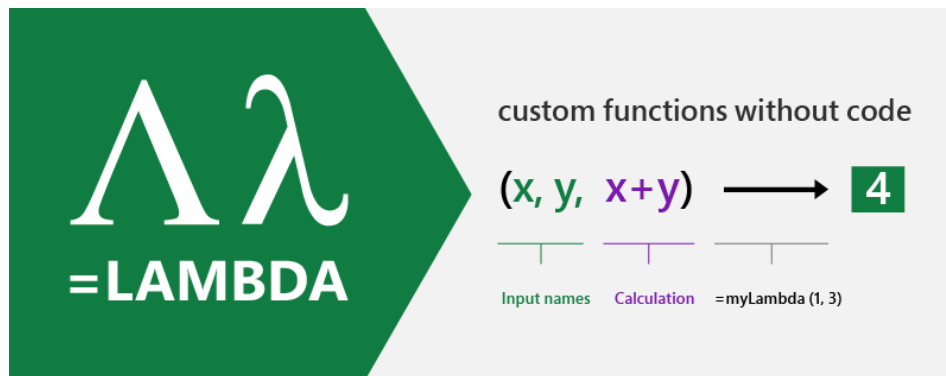
Along with it, Microsoft is also planning to add support for Rich Text Formatting (RTF) inside cells in Excel.



The feature should help users format Excel files better and is scheduled for release by the end of January (i.e. it should be out by the time you read this!).

More on LAMBDA – it's all about ME!!

As mentioned in last month's newsletter, **LET** was recently made Generally Available, followed by its sister function, **LAMBDA**, which has now been released into the Office 365 Beta world. This function "completes" Excel and provides you with the ability to create your own reusable formulae.



As explained previously, in Office 365 Beta, **LAMBDA** allows you to define a custom function in Excel's very own formula language. Moreover, one prescribed function may call another. If the function calls itself, that's an example of recursion, which is a way of a function calling itself, called recursion, which is a way... *[get on with it – Ed.]*

As a reminder, there are three key pieces of **LAMBDA** to understand:

1. **LAMBDA** function components
2. Naming a lambda
3. Calling a lambda function.

1. LAMBDA function components

Let's take a simple example. Consider the following formula:

=LAMBDA(x, x+1)

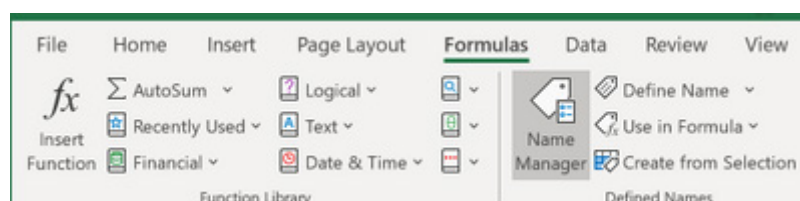
where we have **x** as the argument, which you may pass in when calling the **LAMBDA**, and **x+1** is the logic / operation to be performed. For example, if you were to call this lambda function and define **x** as equal to five (5), then Excel would calculate

$$5 + 1 = 6$$

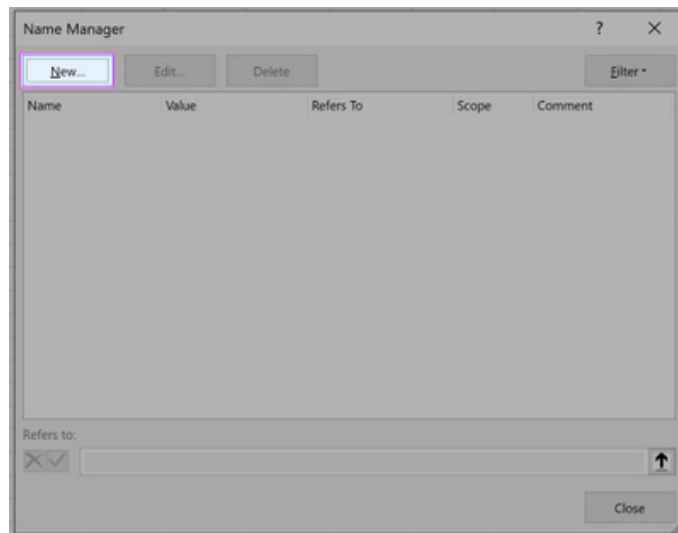
Except it wouldn't. If you tried this you would get **#CALC!** Oops. That's because it's not *quite* as simple as that. You need to name your **LAMBDA**.

2. Naming a Lambda

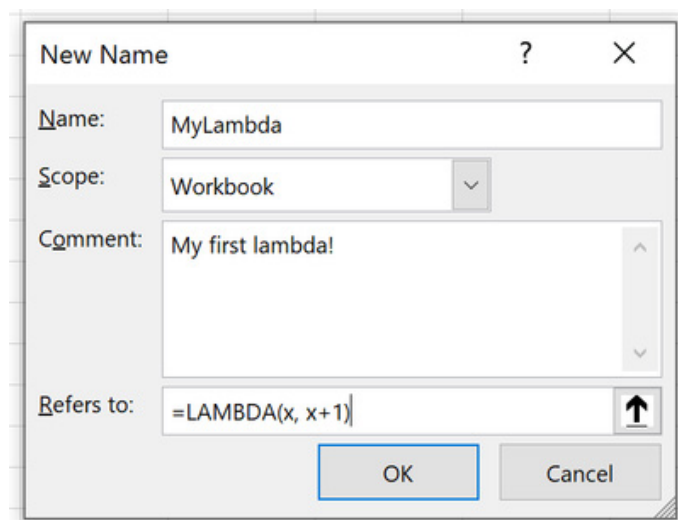
To give your **LAMBDA** a name so it can be re-used, you have to use the Name Manager (**CTRL + F3** / go to the Ribbon and then go to **Formulas** -> **Name Manager**):



Once you open the Name Manager you will see the following dialog:



You then click on 'New' and fill out the related fields, viz.



It's no harder than clicking 'OK' at this point.

3. Calling LAMBDA

Now that you have done this, your first new lambda function may be called in just the same way as every other Excel function is cited, *e.g.*

=MYLAMBDA(5)

which would equal six (6) and not *#CALC!* as before.

You DON'T have to do it this way though if you don't want to. You may call a lambda without naming it. If we hadn't named this marvellous calculation, and simply authored it in the grid as we had first attempted, we could call it by simply typing:

=LAMBDA(x, x+1)(5)

B2				
=LAMBDA(x,x+1)(5)				
	A	B	C	D
1				
2		6		
3				

This trick won't always work though – and our topic for this month is on precisely this issue caused by recursion...

LAMBDA's Aversion to Recursion

To be clear, **LAMBDA** has no aversion; in fact, it enables it. But it makes for a great sub-heading. The “aversion” is that when you use recursion in a **LAMBDA**,

LAMBDA(appropriate syntax)(variable or reference)

may not give the expected result. Instead, it will most likely generate an error, making you think you have written your expression incorrectly. Let me illustrate with a simple example. Consider the following:

	B	C	D	E	F	G	H	I	J
7									
8									
9									
10									
11									
12						9			
13									
14									
15									
16						45			
17									
18						1			
19						2			
20						3			
21						4			
22						5			
23						6			
24						7			
25						8			
26						9			
27									

In cell **G12**, I have typed the number nine (9), which generates the list of numbers one (1) through nine (9) from cell **G18** downwards, using the **SEQUENCE** function, viz.

=SEQUENCE(G12)

For those who recognise this is a dynamic array function and that you require Office 365 at this juncture, you are quite right – but I need Office 365 **Beta** version for the **LAMBDA** functions, so it's OK. I apologise if you don't have Office 365, but again, you might wish to reconsider which version of Excel and Office you are running, in that instance.

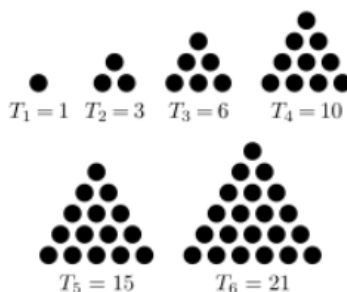
Cell **G16** simply sums this list:

=SUM(G18#)

Therefore, inputting the numbers one (1) through (9) in cell **G12** would generate the results

1, 3, 6, 10, 15, 21, 28, 36, 45, ...

In cell **G16**. These numbers are known as the **triangle numbers**



in mathematics, for very obvious reasons.

I want to show you how you can generate this sequence using a **LAMBDA**. Now, yes, I know the formula above works and you can even use the algebraic solution

=G12*(G12+1)/2

too, but the point is here, I want to show you how to create a recursive **LAMBDA** function that is simple to follow.

On the Ribbon, go to the Formulas tab and click on 'Name Manager' (as above) (**CTRL + F3**) and click on the 'New...' button.

The 'Edit Name' dialog box shows the following details:

- Name:** Triangle
- Scope:** Workbook
- Comment:** (Empty text area)
- Refers to:** =LAMBDA(x,IF(x<2,1,x+Triangle(x-1)))

Buttons: OK, Cancel

I have named my lambda function **Triangle**, and the reference ('Refers to:') is given by

=LAMBDA(x, IF(x<2, 1, x + Triangle(x-1)))

Here, the lambda function takes a parameter **x** and defines it as one (1) if it is less than two (2), else it takes the value **x** and adds the lambda value for **x-1** – hence the recursion. This lambda function has been named **Triangle** (in the 'Name:' box) and is referred to in the formula too.

Therefore, for **x** equals nine (9):

```

Triangle(9) = 9 + Triangle(8)
            = 9 + 8 + Triangle (7)
            = 9 + 8 + 7 + Triangle (6)
            = 9 + 8 + 7 + 6 + Triangle(5)
            = 9 + 8 + 7 + 6 + 5 + Triangle(4)
            = 9 + 8 + 7 + 6 + 5 + 4 + Triangle(3)
            = 9 + 8 + 7 + 6 + 5 + 4 + 3 + Triangle(2)
            = 9 + 8 + 7 + 6 + 5 + 4 + 3 + 2 + Triangle(1)
            = 9 + 8 + 7 + 6 + 5 + 4 + 3 + 2 + 1
            = 45
  
```

Drawn out long hand, the recursion is clear.

This presupposes that **x** is an integer, and I have ensured this by incorporating data validation (**Data -> Data Validation** or **ALT + D + L**) into my input cell (keep it simple!):

Triangle Numbers

Values

The 'Numerical List' section shows an input field for 'Number' containing the value 9. A data validation error message is displayed: "Positive Integer Required. Please enter a whole number greater than zero (0)." The error message box is yellow with a red border. The input field is highlighted in yellow.

Now that the lambda function **Triangle** has been defined, the formula in Excel is easy:

45 =Triangle(G12)

I have purposely created a simple recursive formula to demonstrate how such a calculation might work. In reality, recursive formulae are likely to be much more sophisticated and / or complex.

Let's assume that wasn't the case though, and you wanted to check whether your formula was working. Not everyone (anyone?) can type a formula and have it work first time, every time. Many of us would want to try the formula in a cell first:

#CALC! =LAMBDA(x,IF(x<2,1,x+Triangle1(x-1)))

Oh dear. That didn't work. Of course it doesn't. As explained above, it needs the parameter (x) to be defined:

#NAME? =LAMBDA(x,IF(x<2,1,x+Triangle1(x-1)))(SG\$12)

Rats. This is the problem I alluded to earlier. My definition of the function refers to itself (*i.e.* recursion is exhibited). The name has not yet been defined. Now watch out here. Note the function here is **Triangle1**, not **Triangle**. There is a very important distinction. If I use **Triangle**, a similar formula *will* work:

45 =LAMBDA(x,IF(x<2,1,x+Triangle(x-1)))(SG\$12)

This is very easy to explain. I have already defined **Triangle** in the Name Manager! This is why I am using **Triangle1** to avoid this classic *gotcha*.

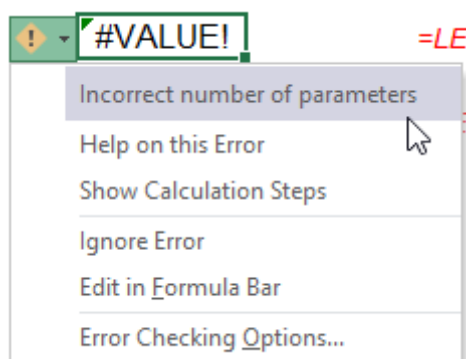
OK, let's define **Triangle1** then. That will mean wrapping the expression in a **LET** function. This allows you to stop writing the same expressions time and time again in a formula, or, as Microsoft puts it, it's "...names on a formula level".

#NAME? =LET(Triangle1,LAMBDA(x,IF(x<2,1,x+Triangle1(x-1))),Triangle1(SG\$12))

This has not worked either, even though **Triangle1** has supposedly been cited by **LET**. The problem is, **Triangle1** has not been defined when we are creating the lambda. This where our trick comes in – which also explains the title for this article: it's all about **ME**. Let's add a parameter (**ME**) to **Triangle1**, and replace the recursive call to **Triangle1** with **ME** (ensuring you pass in **ME** as the first parameter):

#VALUE! =LET(Triangle1,LAMBDA(ME,x,IF(x<2,1,x+ME(ME,x-1))),Triangle1(SG\$12))

Believe it or not, we are getting somewhere now. Even though **#NAME?** has been replaced with **#VALUE!**, the formula has evolved. Assuming you have background error checking enabled (**File -> Options -> Formulas -> Enable background error checking**), you can click on the error to see the issue:



We have an incorrect number of parameters. We have added a parameter to **Triangle1** (namely, **ME**), so the final argument of **LET** (**Triangle1(SG\$12)**) should also have two parameters. We do this by getting **Triangle1** to refer to itself, *viz.*

45 =LET(Triangle1,LAMBDA(ME,x,IF(x<2,1,x+ME(ME,x-1))),Triangle1(Triangle1,SG\$12))

It works! We have checked / debugged our **LAMBDA** in Excel. We started by calling **Triangle1(Triangle1,SG\$12)**, and when we evaluate that lambda, we end up calling **Triangle1(Triangle1,SG\$12-1)**, *etc.*

This technique has employed the **ME** parameter: by passing **Triangle1** as a parameter to itself, it can then use that parameter to call itself.

Once you've got your head around this, proved for yourself Einstein's Theory of General Relativity and demonstrated Schwarzschild's solution to Einstein's field equations using tensor analysis, you have probably got your lambda working properly! Now, simply remove all references to **ME**. Exclude **ME** as the first parameter and replace the **ME** calls with the defined name you want to use (here, **Triangle1**). Therefore, in this instance,

=LET(Triangle1, LAMBDA(ME, x, IF(x<2, 1, x + ME(ME, x - 1))), Triangle1(Triangle1, SG\$12))

becomes

=LAMBDA(x, IF(x<2, 1, x + Triangle1(x - 1)))

i.e. we have proved our expression is correct. We can now add this to the Name Manager.

Sorry to bring back childhood recollections of mathematics, but triangle (or triangular) numbers feature in Pascal's Triangle:

[illegible]

This is an almost “ultimate” recursion, as you can trace any number on any row back through its constituent elements higher up the Triangle. Note that this beast is useful in mathematics as this array displays the binomial coefficients of $(x + 1)^n$. For example:

- $(x + 1)^0 = 1$ – which is the coefficient on the zeroth (0^{th}) row
- $(x + 1)^1 = x + 1 = 1x + 1$ – the coefficients **1** and **1** are on row **1** of Pascal's Triangle
- $(x + 1)^2 = x^2 + 2x + 1 = 1x^2 + 2x + 1$ – the coefficients **1**, **2** and **1** are on row **2** of Pascal's Triangle
- $(x + 1)^3 = x^3 + 3x^2 + 3x + 1 = 1x^3 + 3x^2 + 3x + 1$ – the coefficients **1**, **3**, **3** and **1** are on row **3** of Pascal's Triangle, etc.

number of items (i.e. the number of distinct subsets of items where order is unimportant). You should use **COMBIN** to determine the total possible number of groups for a given number of items.

COMBIN(number, number chosen)

- **number:** this is required and represents the number of items
- **number chosen:** this is also required. This denotes the number of items in each combination.

COMBIN(row number, column number)

```
=LET(Pascal1, LAMBDA(ME, row_num, col_num, IF(OR(row_num < 0, col_num < 0), 0, IF(row_num = 0, IF(col_num = 0, 1, 0), ME(ME, row_num - 1, col_num - 1) + ME(ME, row_num - 1, col_num)))), Pascal1(Pascal1, $I$12, $I$13))
```

Alternative Approaches to Calculating Pascal's Triangle Element		
Assumptions		
Row and Column Numbers		
Row Number	30	
Column Number	8	
Alternative Calculations		
Excel Functions		
Output Value using INDEX	5,852,925	=IFERROR(INDEX("Pascal's Triangle"\$F\$12:\$J,\$I\$2+\$J,\$I\$2+1,\$I\$3+1),)
Output Value using COMBIN	5,852,925	=IFERROR(COMBIN(\$I\$2,\$I\$13),)
LAMBDA Approach		
Using LAMBDA from Name Manager	5,852,925	=Pascal(112,113)
Equivalent formula in Name Manager	#CALC!	=LAMBDA(row_num,col_num,IF(OR(row_num<0,col_num<0),0,IF(row_num=0,IF(col_num=0,1,0),Pascal(row_num-1,col_num-1)+Pascal(row_num-1,col_num))))(112,113)
Equivalent formula with reference	#NAME?	=LAMBDA(row_num,col_num,IF(OR(row_num<0,col_num<0),0,IF(row_num=0,IF(col_num=0,1,0),Pascal(row_num-1,col_num-1)+Pascal(row_num-1,col_num))))(112,113)
Using LET	#NAME?	=LET(Pascal,LAMBDA(row_num,col_num,IF(OR(row_num<0,col_num<0),0,IF(row_num=0,IF(col_num=0,1,0),Pascal(row_num-1,col_num-1)+Pascal(row_num-1,col_num))))),Pascal(\$I\$20,\$I\$21))
Adding ME	#VALUE!	=LET(Pascal,LAMBDA(ME,row_num,col_num,IF(OR(row_num<0,col_num<0),0,IF(row_num=0,IF(col_num=0,1,0),ME(ME,row_num-1,col_num-1)+ME(ME,row_num-1,col_num))))),Pascal(\$I\$20,\$I\$21))
Corrected and checked	5,852,925	=LET(Pascal,LAMBDA(ME,row_num,col_num,IF(OR(row_num<0,col_num<0),0,IF(row_num=0,IF(col_num=0,1,0),ME(ME,row_num-1,col_num-1)+ME(ME,row_num-1,col_num))))),Pascal(Pascal,\$I\$12,\$I\$13))
	✓	Values are equal

This **Pascal** lambda function is a more complex example of recursion that highlights an important design consideration. On my reasonably powerful computer, when I tried to select an element from row 30 of Pascal's Triangle, Excel took c.55 seconds to calculate, due to all the interim recursive calculations.

Be careful. Lambda calculations in Excel are very fast, but if you adopt an excessive recursive approach as I have done here, any powerful PC will be slowed down to a "crawling speed". The problem here is that the calculation speed in this instance is proportional to $4^n/\sqrt{n}$, where n is the row number.

A better lambda might be **Pascal_Row**, where the time taken is proportional to n^2 :

```
=LAMBDA(a, b, LET(Pascal_Row,
  LAMBDA(ME, x,
    IF(x=0, 1, IF(x=1, {1, 1},
      LET(seq, SEQUENCE(x + 1), previous_row, ME(ME, x - 1), shifted, IF(seq=1, 0, INDEX(previous_row, 1, seq - 1)),
        IFERROR(previous_row + shifted, 1)
      )))
    INDEX(Pascal_Row(Pascal_Row, a), 1, b + 1)))
```

This is achieved by calculating on a row, rather than an element, basis.

But even this may be improved upon. Consider **Pascal_Fast**:

```
=LAMBDA(n, k,
  IF(k * 2 > n, Pascal_Fast(n, n - k),
  IF(k = 0, 1,
  LET(gcd_cutoff, 2 ^ 53 / n,
    previous, Pascal_Fast(n, k - 1) * (n + 1 - k) / k,
    IF(n <= gcd_cutoff, previous,
      LET(gcd, GCD(previous, k),
        remainder, k / gcd,
        previous / gcd * ((n + 1 - k) / remainder)))
    )))
```

This is very quick compared to both of the previous alternatives, as the time here is proportional to just n , the row number. The problem as you may have noticed here, is that as you maximise the iterative approach,

the transparency of the calculation – for the average user – tends to become more and more opaque. PhDs in Computer Science are available upon request.

As a compromise, perhaps

```
=LAMBDA(n, k, IF(k=0, 1, Pascal_Almost_As_Fast(n, k - 1) * (n + 1 - k) / k))
```

(**Pascal_Almost_As_Fast**) might work best, which takes approximately twice as long as the above monster (*i.e.* "almost" instantly).

Care with IFS and SWITCH

Given how calculations may "blow out" quickly when creating iterative functions, it's best to avoid another classic gotcha. Let me illustrate with one of the best-known iterative sequences, the Fibonacci sequence:

1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233, 377, ...

This sequence is calculated as $F_n = F_{n-1} + F_{n-2}$.

You might choose to define a lambda function using **IFS** as follows:

```
=LAMBDA(x, IFS(x=1, 1, x = 2, 1, TRUE, Fibonacci(x - 1) + Fibonacci(x - 2)))
```

If you are starting to follow lambdas, you should see this appears to make sense (the final argument of **IFS**, given the criterion is simply **TRUE**, resorts in advising what to do when x is neither equal to one (1) nor two (2)). There is a problem though. This function will run forever, since each parameter in this **IFS** statement must be evaluated before **IFS** gets evaluated. Therefore, when evaluating **Fibonacci(1)**, it will try to evaluate **Fibonacci(0) + Fibonacci(-1)**, and so on. Not good!

Therefore, you should use **IF** instead:

```
=LAMBDA(x, IF(x = 1, 1, IF(x = 2, 1, Fibonacci(x - 1) + Fibonacci(x - 2))))
```

as **IF** does not evaluate a parameter it does not return.

Be advised, **SWITCH** exhibits a similar such property, and you should choose to use **CHOOSE** instead.

Fibonacci Sequence

Values

Number

Numerical List

=OFFSET(\$G\$17,\$G\$12,)

1
1
2
3
5
8
13
21
34
55
89
144
233
377
610
987
1597
2584
4181
6765

Algebraic Alternative

=1/SQRT(5)*((1+SQRT(5))/2)^\$G\$12-((1-SQRT(5))/2)^\$G\$12)

LAMBDA Approach

=Fibonacci(\$G\$12)

#CALC!

=LAMBDA(x,IF(x=1,1,IF(x=2,1,Fibonacci(x-1)+Fibonacci(x-2))))

#NAME?

=LAMBDA(x,IF(x=1,1,IF(x=2,1,Fibonacci(x-1)+Fibonacci(x-2))))(\$G\$12)

#NAME?

=LET(Fibonacci,LAMBDA(x,IF(x=1,1,IF(x=2,1,Fibonacci(x-1)+Fibonacci(x-2))))),Fibonacci(\$G\$12))

#VALUE!

=LET(Fibonacci,LAMBDA(x,IF(x=1,1,IF(x=2,1,ME(ME(x-1)+ME(ME(x-2))))),Fibonacci(\$G\$12))

=LET(Fibonacci,LAMBDA(x,IF(x=1,1,IF(x=2,1,ME(ME(x-1)+ME(ME(x-2))))),Fibonacci(Fibonacci(\$G\$12))

Faster approach

☐

Values are equal

Just to highlight the point earlier about optimisation, the Fibonacci lambda function may be improved upon too – again, at the risk of mathematical / formulaic comprehension:

=LAMBDA(n, LET(Fibonacci_Faster, LAMBDA(ME, x, F_1, F, current, IF(x <= 2, 1, IF(x = current, F, ME(ME, x, F, F_1 + F, current + 1)))), Fibonacci_Faster(Fibonacci_Faster, n, 1, 1, 2)))

Now you have something to study the next time you are stranded on a desert island.

Word to the Wise

Playing with new functions is highly addictive and can lead you to using them when they are **not** needed. No matter how optimised recursive calculations are, I strongly advise that if you don't need them, don't use them. Formulaic alternatives, such as **COMBIN** for Pascal's Triangle, are instant and simple to the naked eye.

Furthermore, do note that the current operand stack limit in Excel is 1,024. This should be borne in mind together with calculation times, as the current recursion limit is set as 1,024 divided by (number of lambda parameters + 1).

Finally, some common problems faced in financial modelling are **not** recursive in nature, so try not to confuse the issue unnecessarily. For example, the most common financial modelling illustration is the calculation of interest on an average cash balance. This is actually an instance of solving two simultaneous equations. Applying recursive logic would be inappropriate.

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...

Here's a common problem that many accountants face regularly. You set a budget, maybe in cash terms, full time equivalents or other resources. And then something happens to mess it all up. It could be the planning horizon extends or contracts and / or the project is brought forward or delayed.

Wouldn't it be good to be able to simply change your start and end dates and have the budgeted numbers reforecast automatically? Well, that's this month's challenge...

Imagine you had just finalised the budget for a project and (say) it started in Period 3 and ended in Period 8 as pictured:

Original Forecast

Assumptions

Period #
Forecast \$

Total	1	2	3	4	5	6	7	8
45			5	6	7	8	9	10

Suddenly, your boss tells you the amounts need to be reallocated on a "similar basis" but for Periods 4 to 15. That's pretty easy, as this duration is double the original project length, so you would just attribute half of each period's amount to the new periods, viz.

Total	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
45	-	-	-	2.50	2.50	3.00	3.00	3.50	3.50	4.00	4.00	4.50	4.50	5.00	5.00

But then your boss challenges you further. They want to be able to flex both the start and end periods and be able to see the result instantly. Could you do it? Could you create a spreadsheet that would reallocate the amounts quickly and accurately?

The gauntlet has been thrown down...

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

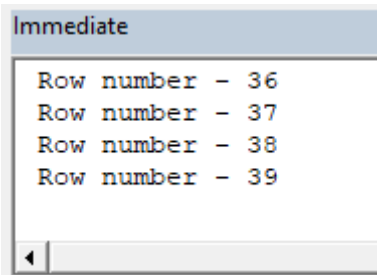
Visual Basics

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 thought we would look at "do-ing" some events...

It's a great feeling when you write an involved piece of code with grand loops that apply over thousands of rows. There's a sweet sensation when you save the file, hit run, and watch your handiwork unfold in front of you... until the screen stops updating, Excel goes to "Not Responding" and you're left wondering: "Has it crashed? Is it still running? What do I do now?"

[Module1 (Code)] (Not Responding)

This has happened to us recently, running a VBA script on a database of 50,000 names, and supposedly ceasing to respond a mere 39 rows in.



This happens essentially because Windows thinks that Excel isn't responding (since Windows displays what's at the top of your screen), and it thinks that because Excel is devoting all of its resources to actually running your macro. It also happens when you're updating the status bar or immediate window, and you can actually see the updates stop and pause.

An easy way around this is to include the code "DoEvents" into your loop somewhere. DoEvents essentially passes control back to Windows, essentially pausing your code and allowing Windows to send all of the keystrokes, commands and any other events through to Excel.

```
Debug.Print "Row number - " & i
DoEvents

Next i
```

Therefore, if you're wondering why Excel sometimes doesn't respond to **ESC** or **CTRL + Break**, incorporating DoEvents into your code will help make it more responsive and ensure that it continues to update you while it's running your macros. It's the virtual equivalent of having

Excel take its headphones off and look up, to see what the rest of the operating system is doing. More next time.

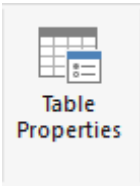
Power Pivot Principles

We continue our series on the Excel COM add-in, Power Pivot. This month, we discuss potential steps we can take when we receive 'messy' data.

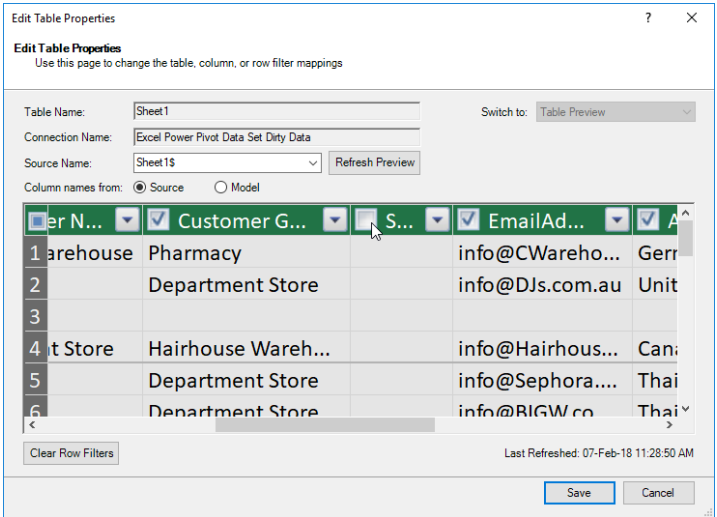
Consider the following data, which, after importing it into Power Pivot, looks something like this:

	CustomerKey	SalesTerritoryKey	Customer Name	Customer Group	Suffix	EmailAddress	AddressLine1	AddressLine2
1	101	8	Chemist Warehouse	Pharmacy		info@CWareho...	Germany	
2	102	10	DJs	Department Store		info@DJs.com.au	United Kingdom	
3								
4	104	6	Department Store	Hairhouse Wareh...		info@Hairhous...	Canada	
5	105	4	Sephora	Department Store		info@Sephora...	Thailand	
6	106	4	BIGW	Department Store		info@BIGW.co...	Thailand	
7	107	10	Target	info@Target.com.au		Department St...	United Kingdom	
8	108	3	Woolworths	Supermarket		info@Woolwor...	New Zealand	
9	109	2	National Pharmaces	Pharmacy		info@National ...	Italy	
10	110	1	Terry White	Pharmacy		info@Terry Whi...	United States	
11								
12	112	7	Wholesale	Tanning Shop		info@Tanning S...	France	
13	113	8	BigBeauty	Wholesale		info@BigBeaut...	Germany	
14	114	10	Marble Shop	Wholesale		United Kingdom	info@Marble S...	
15	115	9	Distrobeauty	Wholesale		info@Distrobe...	Australia	
16	116	3	Health Spa	Wholesale		info@Health Sp...	New Zealand	
17	117	1	Golden Tan	Wholesale		info@Golden T...	United States	

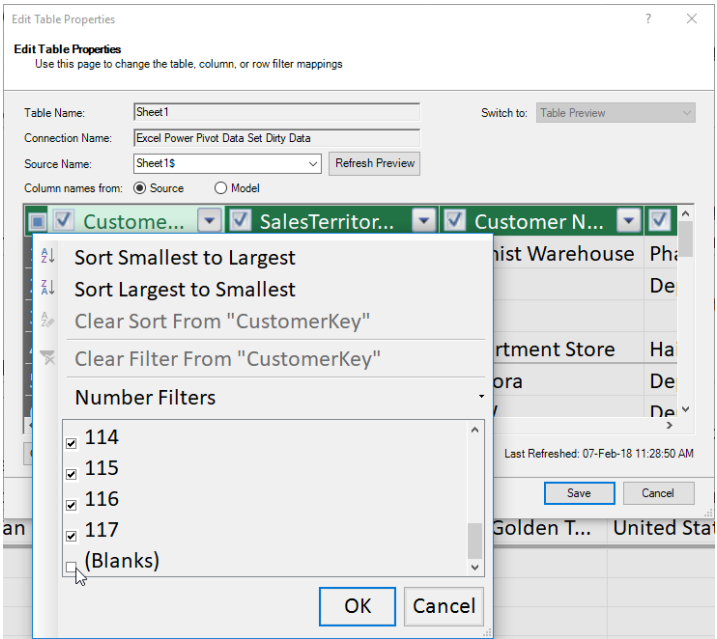
There are columns that are entirely blank, such as the **Suffix** and **AddressLine2** columns. Looking at the rows, some of them are completely blank too. To eliminate blank rows and columns, we can filter the data before importing it to Power Pivot. If you forgot how to do this, head to the design tab on the Ribbon of the Power Pivot window, and select 'Table Properties'.



We can then filter out the blank columns by choosing to not have them imported into the dataset:



To filter out the blank rows, we can choose a drop-down menu of one of the columns 'Customer Name' and unselect the 'Blanks' values.



Our dataset is now looking a lot better:

	CustomerKey	SalesTerritoryKey	Customer Name	Customer Group	EmailAddress	AddressLine1	Phone	DateFirstPurchase
1	101	8	Chemist Warehouse	Pharmacy	info@CWareho...	Germany	1 (11) 50...	418
2	102	10	DJs	Department Store	info@DJs.com.au	United Kingdom	1 (11) 50...	418
3	104	6	Department Store	Hairhouse Wareh...	info@Hairhous...	Canada	1 (11) 50...	418
4	105	4	Sephora	Department Store	info@Sephora....	Thailand	1 (11) 50...	418
5	106	4	BIGW	Department Store	info@BIGW.co...	Thailand	1 (11) 50...	418
6	107	10	Target	Department St...	info@Target.com.au	United Kingdom	1 (11) 50...	418
7	108	3	Woolworths	Supermarket	info@Woolwor...	New Zealand	1 (11) 50...	418
8	109	2	National Pharmacies	Pharmacy	info@National ...	Italy	1 (11) 50...	418
9	110	1	Terry White	Pharmacy	info@Terry Whi...	United States	1 (11) 50...	418
10	112	7	Wholesale	Tanning Shop	info@Tanning S...	France	1 (11) 50...	418
11	113	8	BigBeauty	Wholesale	info@BigBeaut...	Germany	717-555...	426
12	114	10	Marble Shop	Wholesale	United Kingdom	info@Marble S...	817-555...	426
13	115	9	Distrobeauty	Wholesale	info@Distrobe...	Australia	431-555...	426
14	116	3	Health Spa	Wholesale	info@Health Sp...	New Zealand	208-555...	425
15	117	1	Golden Tan	Wholesale	info@Golden T...	United States	135-555...	425

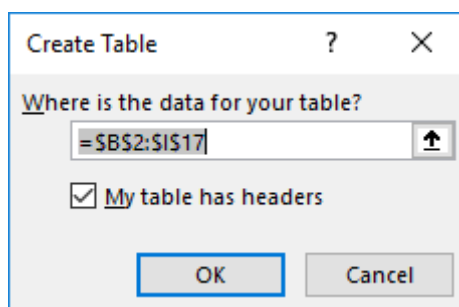
Looking closer, some entries in the **Customer Name** column have been entered incorrectly into the **Customer Group** column, together with a couple of errors in the **Customer Group** and **Email Address** columns.

We can fix these issues by ‘cleaning’ the data in Excel. Let’s utilise Power Pivot’s clipboard function. Select all (**CTRL + A**) and copy (**CTRL + C**) the table from Power Pivot and paste it (**CTRL + V**) into a sheet in Excel. Let’s perform a few more adjustments to the data, *et voilà!*

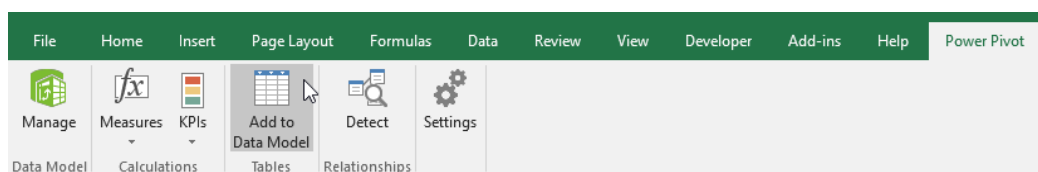
CustomerKey	SalesTerritoryKey	Customer Name	Customer Group	EmailAddress	AddressLine1	Phone	DateFirstPurchase
101	8	Chemist Warehouse	Pharmacy	info@CWarehouse.co	Germany	1 (11) 500 555-0162	41842
102	10	DJs	Department Store	info@DJs.com.au	United Kingdom	1 (11) 500 555-0110	41838
104	6	Hairhouse Warehouse	Department Store	info@Hairhouse Warehouse.com.au	Canada	1 (11) 500 555-0162	41821
105	4	Sephora	Department Store	info@Sephora.com.au	Thailand	1 (11) 500 555-0131	41846
106	4	BIGW	Department Store	info@BIGW.com.au	Thailand	1 (11) 500 555-0151	41822
107	10	Target	Department Store	info@Target.com.au	United Kingdom	1 (11) 500 555-0184	41847
108	3	Woolworths	Supermarket	info@Woolworths.com.au	New Zealand	1 (11) 500 555-0126	41832
109	2	National Pharmacies	Pharmacy	info@National Pharmacies.com.au	Italy	1 (11) 500 555-0164	41848
110	1	Terry White	Pharmacy	info@Terry White.com.au	United States	1 (11) 500 555-0110	41850
112	7	Tanning Shop	Wholesale	info@Tanning Shop.com.au	France	1 (11) 500 555-0117	41822
113	8	BigBeauty	Wholesale	info@BigBeauty.com.au	Germany	717-555-0164	42630
114	10	Marble Shop	Wholesale	info@Marble Shop.com.au	United Kingdom	817-555-0185	42658
115	9	Distrobeauty	Wholesale	info@Distrobeauty.com.au	Australia	431-555-0156	42637
116	3	Health Spa	Wholesale	info@Health Spa.com.au	New Zealand	208-555-0142	42573
117	1	Golden Tan	Wholesale	info@Golden Tan.com.au	United States	135-555-0171	42595

Obviously, it would make sense to perform the adjustments in Power Query or Power Pivot, but that’s not the point here...

Now how do we get this back into Power Pivot? First, convert the range into a Table. Highlight the range and press (**CTRL + T**), to convert it into a Table. Be sure to tick ‘My table has headers’.



Give the Table a name, then head up to the ‘Power Pivot’ tab on the Ribbon and select the ‘Add to Data Model’ option.



And there you have it, our data all cleaned up, back in Power Pivot.

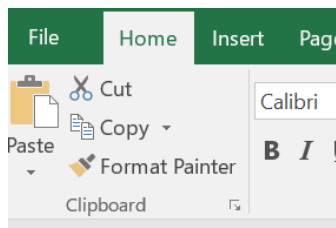
	CustomerKey	SalesTerritoryKey	Customer Name	Customer Group	EmailAddress	AddressLine1	Phone	DateFirstPurchase
1	101	8	Chemist Warehouse	Pharmacy	info@CWareho...	Germany	1 (11) 50...	418
2	102	10	DJs	Department Store	info@DJs.com.au	United Kingdom	1 (11) 50...	418
3	104	6	Hairhouse Wareh...	Department Store	info@Hairhous...	Canada	1 (11) 50...	418
4	105	4	Sephora	Department Store	info@Sephora....	Thailand	1 (11) 50...	418
5	106	4	BIGW	Department Store	info@BIGW.co...	Thailand	1 (11) 50...	418
6	107	10	Target	Department Store	info@Target.co...	United Kingdom	1 (11) 50...	418
7	108	3	Woolworths	Supermarket	info@Woolwor...	New Zealand	1 (11) 50...	418
8	109	2	National Pharmacies	Pharmacy	info@National ...	Italy	1 (11) 50...	418
9	110	1	Terry White	Pharmacy	info@Terry Whi...	United States	1 (11) 50...	418
10	112	7	Tanning Shop	Wholesale	info@Tanning S...	France	1 (11) 50...	418
11	113	8	BigBeauty	Wholesale	info@BigBeaut...	Germany	717-555...	426
12	114	10	Marble Shop	Wholesale	info@Marble S...	United Kingdom	817-555...	426
13	115	9	Distrobeauty	Wholesale	info@Distrobe...	Australia	431-555...	426
14	116	3	Health Spa	Wholesale	info@Health Sp...	New Zealand	208-555...	425
15	117	1	Golden Tan	Wholesale	info@Golden T...	United States	135-555...	425

More *Power Pivot Principles* next month.

Power Query Pointers

Each month we'll reproduce one of our articles on Power Query (Excel 2010 and 2013) / Get & Transform (Office 365, Excel 2016 and 2019) from www.sumproduct.com/blog. If you wish to read more in the meantime, simply check out our Blog section each Wednesday. This month, we look at combining several Power Query functions in order to standardise some incoming data (almost a continuation of the Power Pivot article!).

For today's example, we will use our ever-reliable fictional salespeople – but as usual, they have thrown some problems into my data...

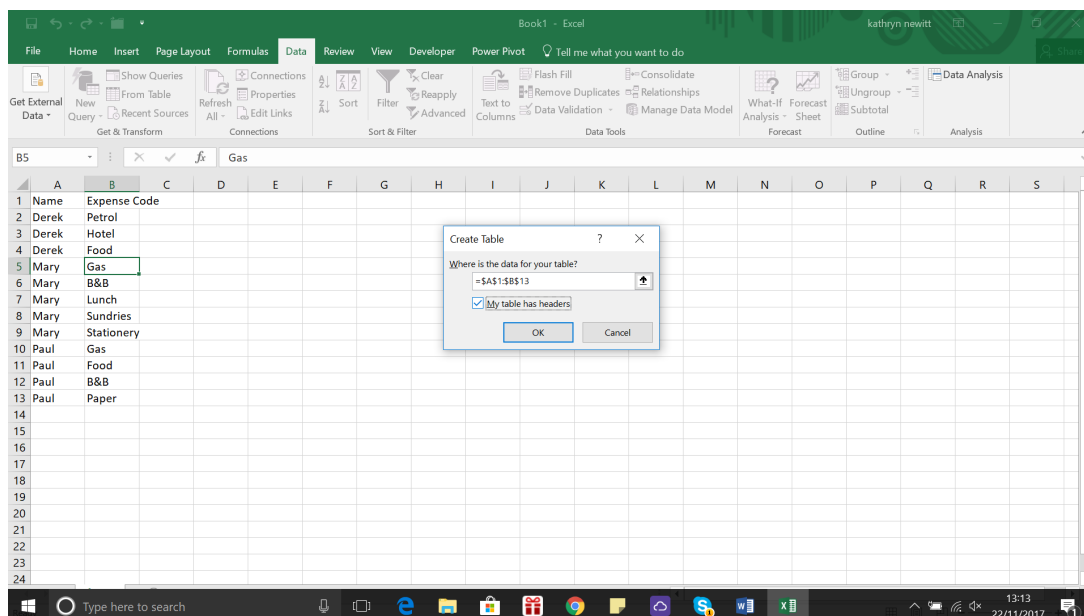


	A	B	C
1	Name	Expense Code	
2	Derek	Petrol	
3	Derek	Hotel	
4	Derek	Food	
5	Mary	Gas	
6	Mary	B&B	
7	Mary	Lunch	
8	Mary	Sundries	
9	Mary	Stationery	
10	Paul	Gas	
11	Paul	Food	
12	Paul	B&B	
13	Paul	Paper	
14			

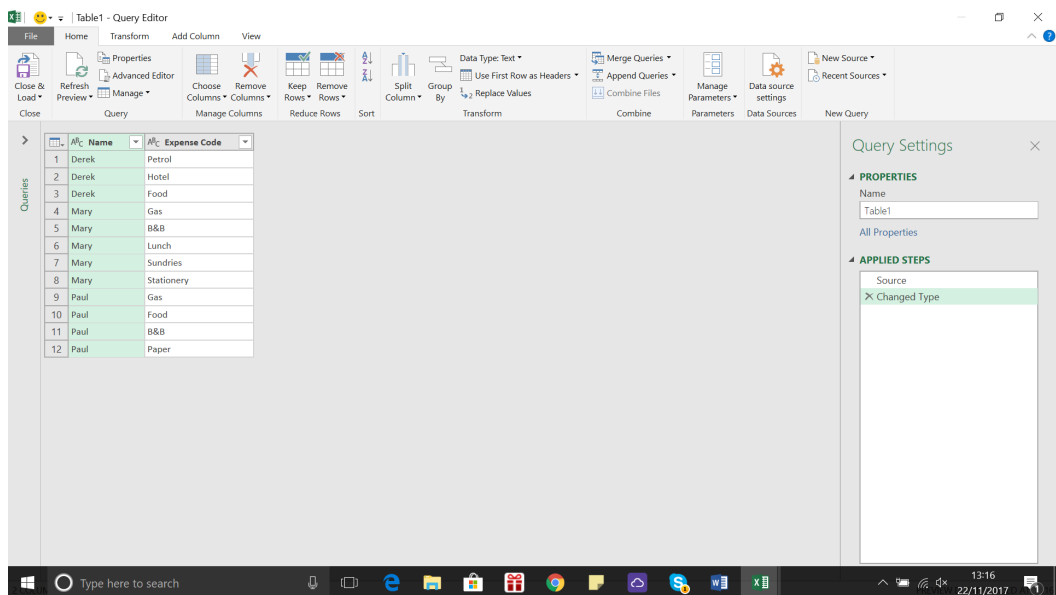
Derek has helpfully entered the expense codes as we would like to describe them, but Mary and Paul have been less consistent. We would like to change the expense codes to the standard names so that we can

calculate totals, but without having to go through and change each entry manually – in a large dataset many people may have used 'Gas' instead of 'Petrol'.

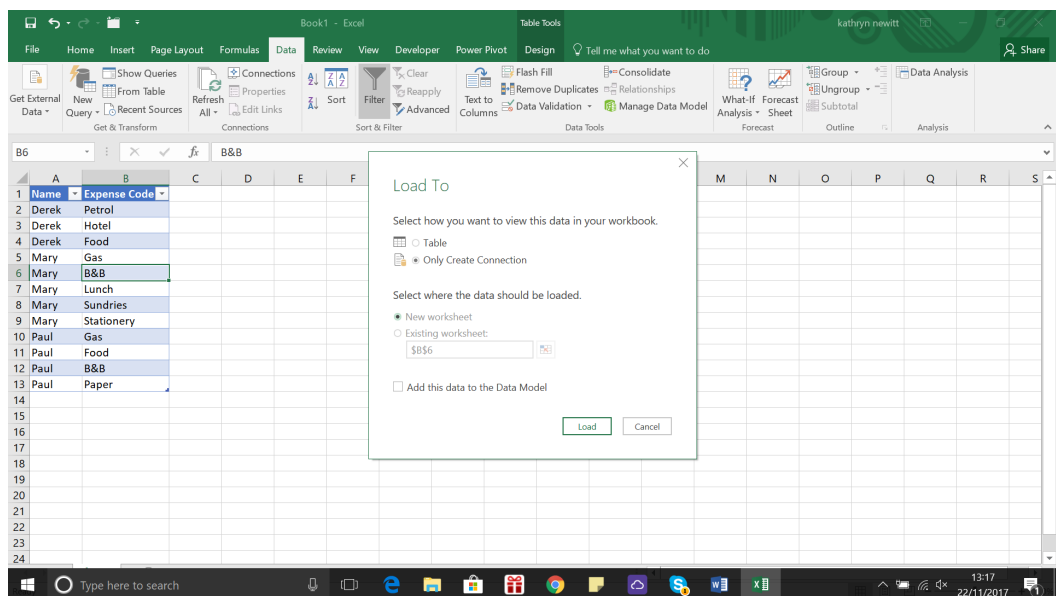
First, we need to create a query for my data, which we do on the Data tab in the 'Get and Transform' section. We'll opt to create a query 'From Table':



Our data is not yet in a Table, so this will be done as part of the query creation process; we're prompted for the boundaries and whether we have titles (headers). Then, we may create our query.

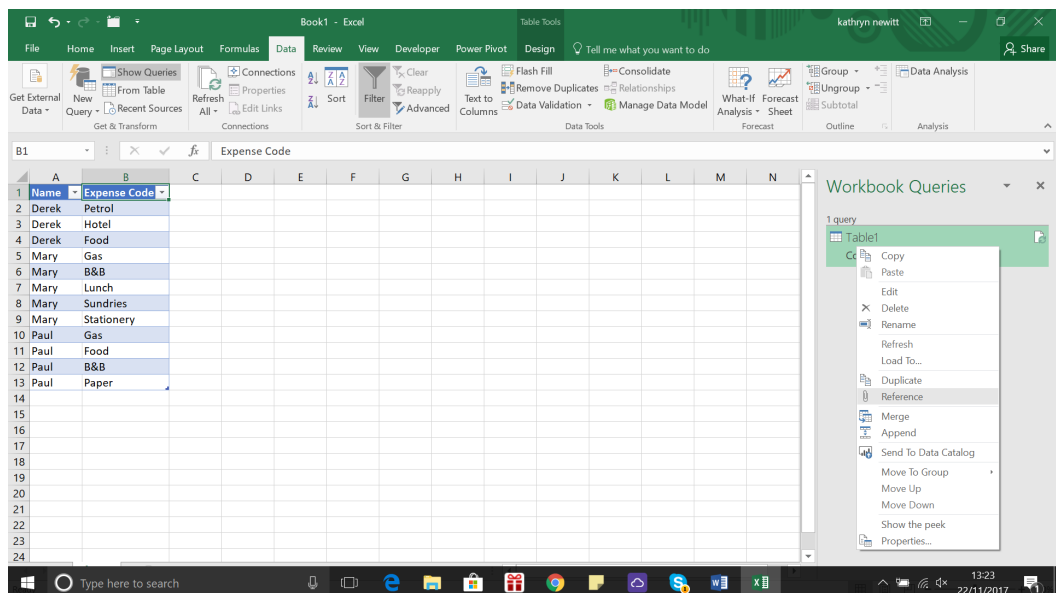


From the File or Home tab, we'll choose to 'Close and Load To', in order to create a connection.

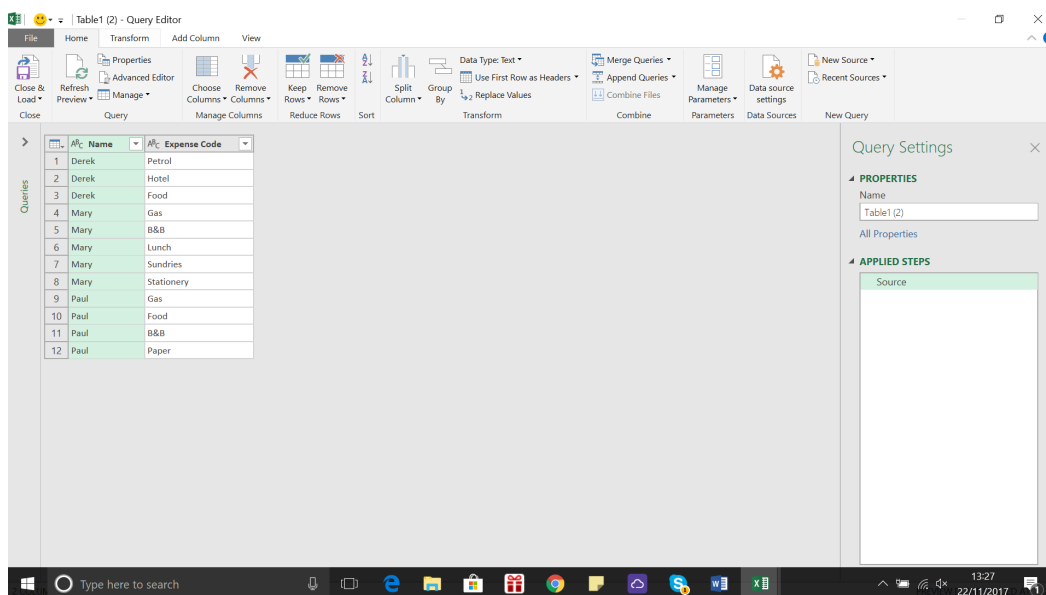


We don't want to load data to my workbook at this point, as nothing has yet changed.

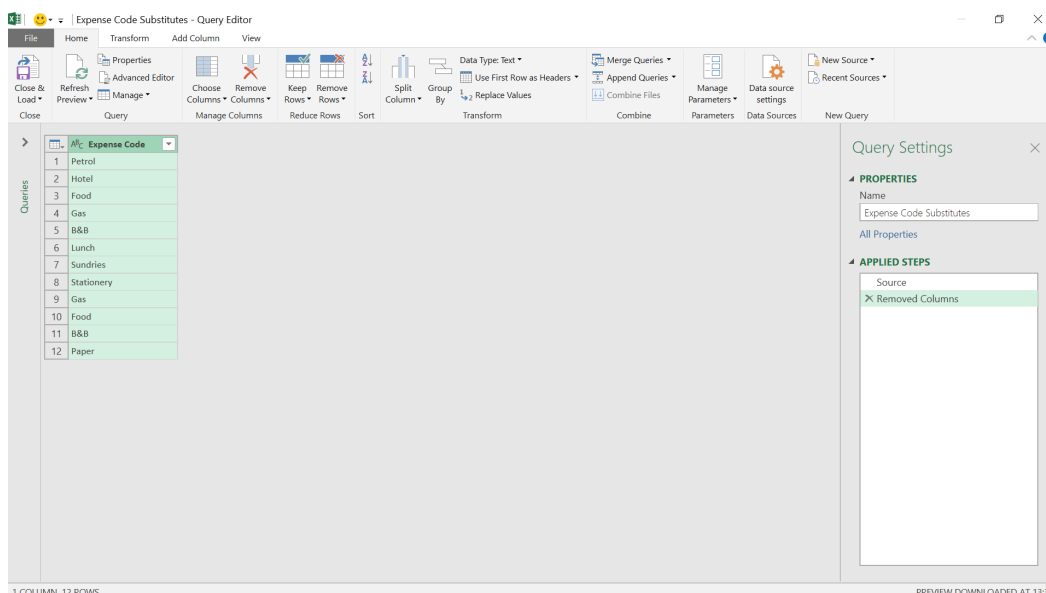
To replace the non-standard entries in **Expense Code**, we need a reference table. This avoids the need to 'hard code' anything (if you don't know what this means ask a programmer and watch them grimace). Therefore, we right-click on our query and look at the options:



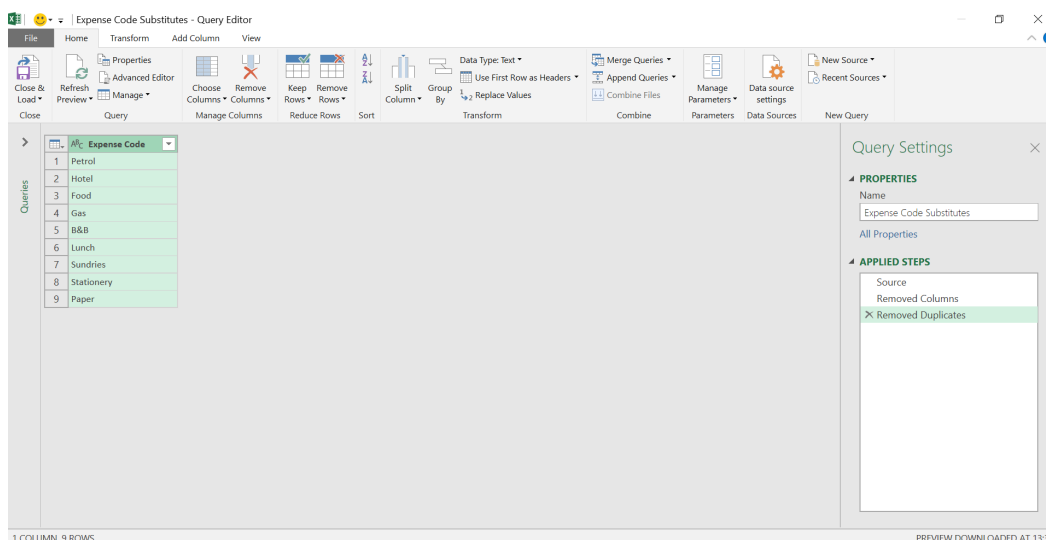
We have a 'Reference' option, so let's choose this. The difference between 'Duplicate' and 'Reference' is subtle: if we choose 'Reference', the new query is based upon the results of the first table, rather than replicating it entirely.



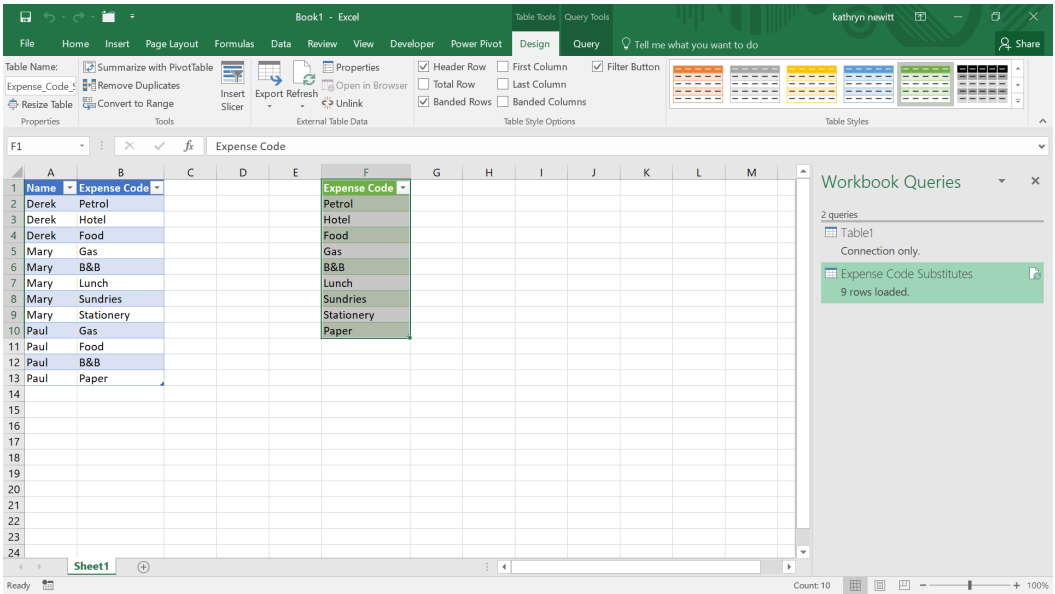
Thus, it looks just like the first table – we will rename this table to 'Expense Codes Substitutes' so that it's easier to understand its purpose. Remove the **Name** column as we are only interested in the expense codes for now.



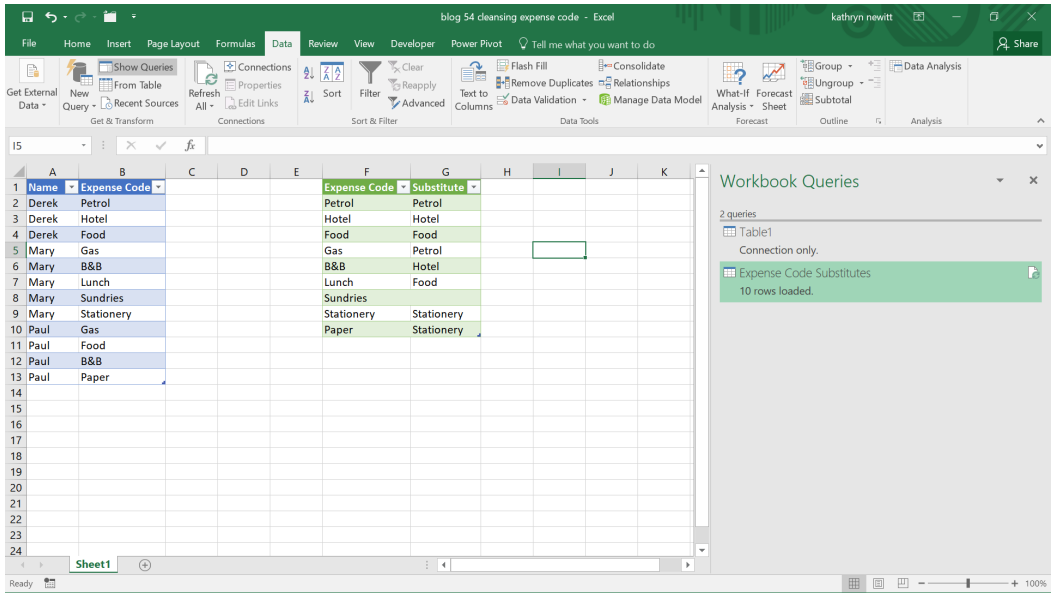
We have too many similar entries, so in the 'Reduce Rows' section, let's choose the 'Remove Rows' dropdown and select 'Remove Duplicate Rows'.



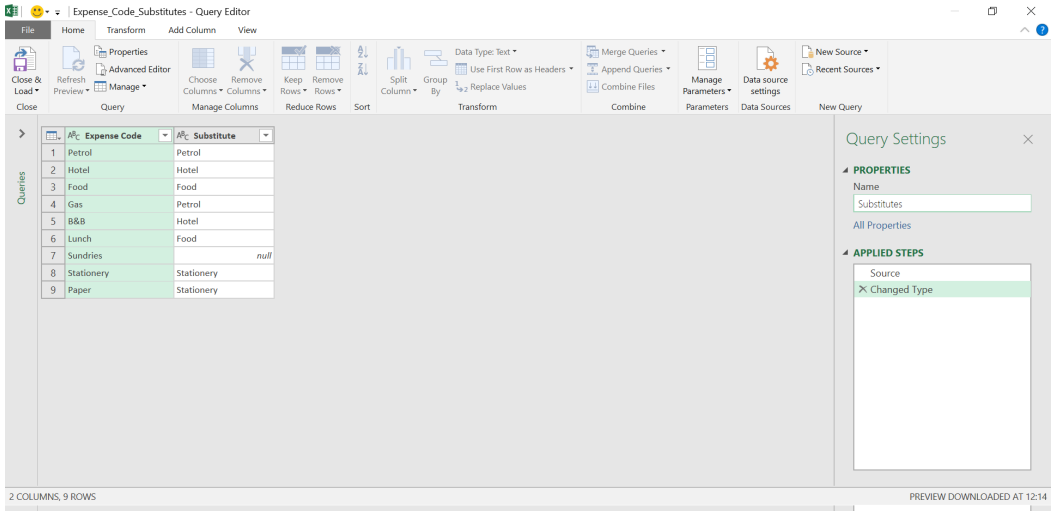
The next step will be to decide which cells are standard entries and which are substitute entries. Now that our initial data has been reduced to unique values, we need to go back to Excel to indicate manually which values need to be substituted with a standard value. We close and load to the existing worksheet.



Next, we need to add a column to the new table called **Substitute** and reorganise our data so that Power Query can read what and when to substitute for each entered expense code.

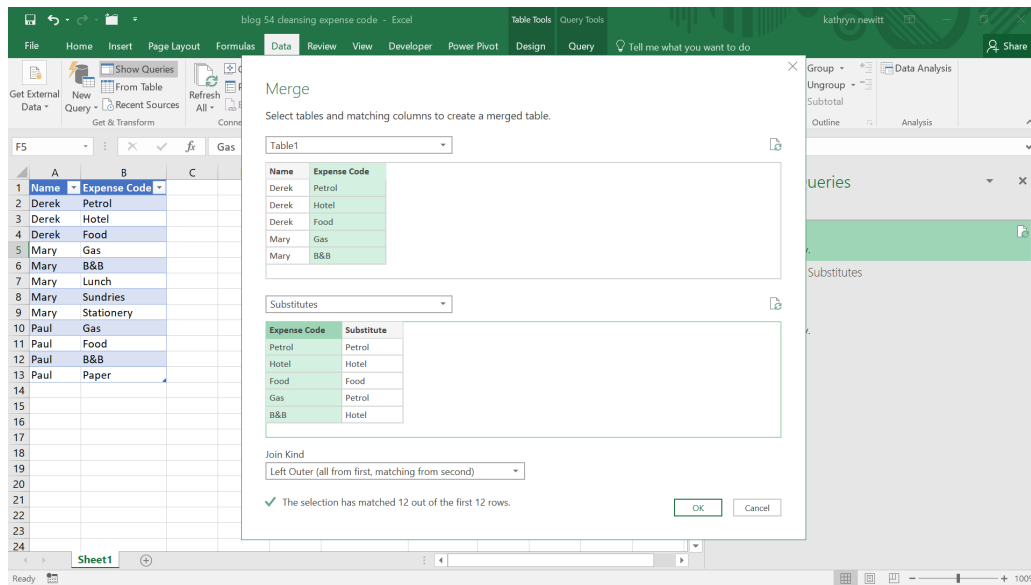


We have indicated substitutes where appropriate (and deliberately missed one!). We now need to create a new query for the edited table:

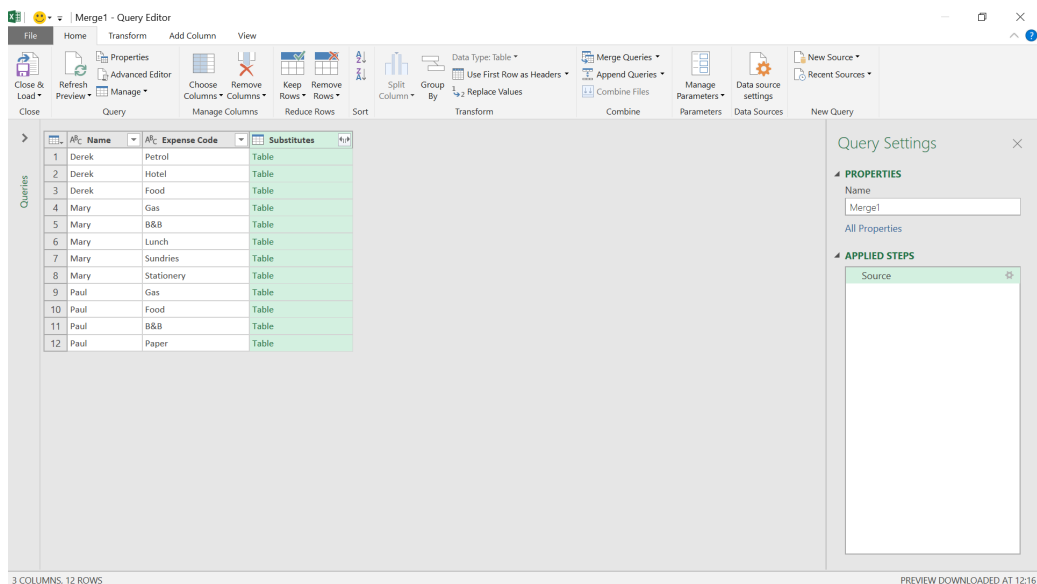


Let's call the new query 'Substitutes' and create it as connection only (since we have not changed it, we don't need to load it).

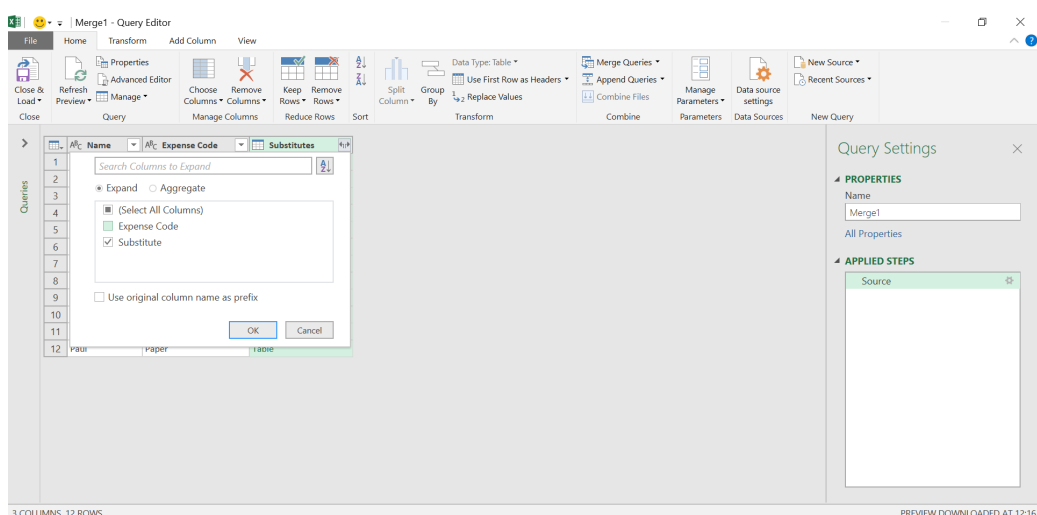
Back in the worksheet, right-click on the original query 'Table1' and join it to 'Substitutes', using the 'Merge' option. Since the data in 'Substitutes' came from 'Table1' this is known as *self-referencing*.



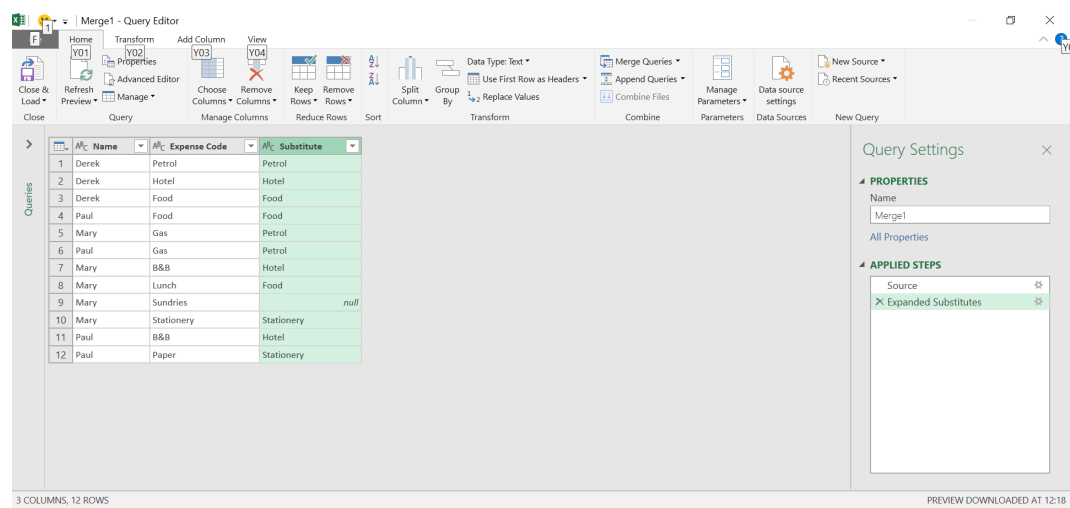
Select **Expense Code** from each table and choose the 'Left Outer' join option and click 'OK'.



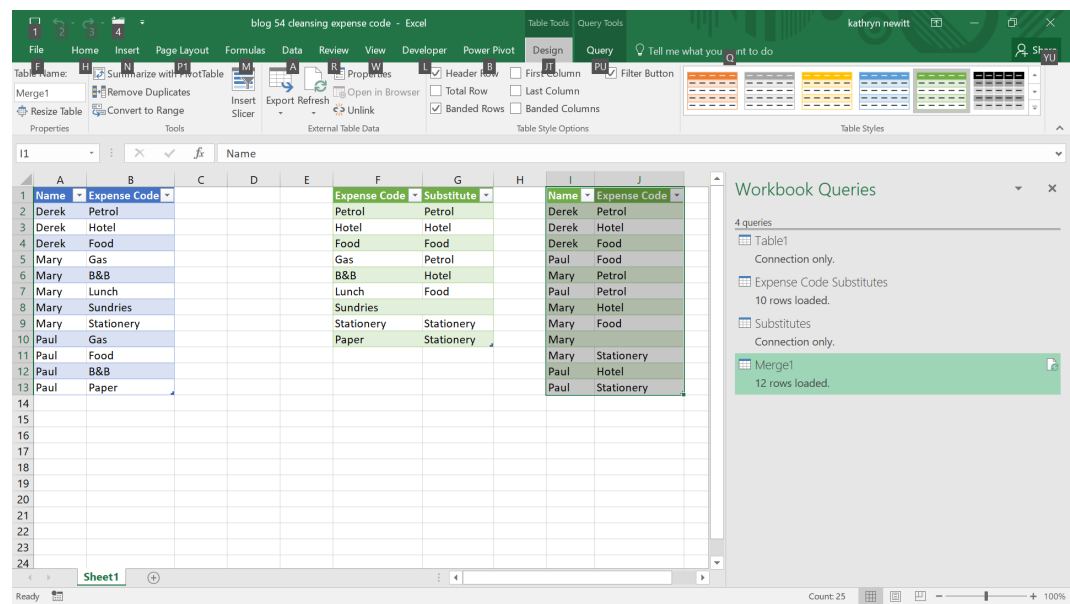
We can now expand the **Substitutes** column:



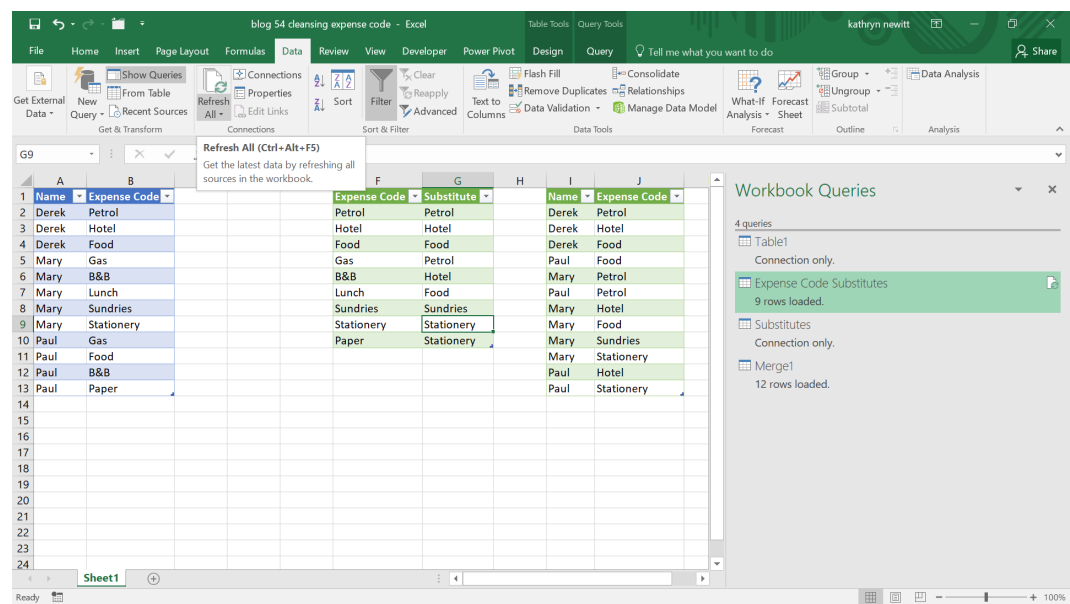
We only want the **Substitute** column from the ‘Substitutes’ table:



We may now delete the original **Expense Code** field and rename **Substitute** to be the new **Expense Code...** but wait, Mary has no substitute set up for ‘Sundries’ (see, we missed one on purpose!). To show that everything may be refreshed, we continue with the deletion and renaming here, and load our query to the same worksheet.



We manually edit the ‘Substitutes’ table to make sure all the substitutes are populated (including ‘Sundries’), and refresh the data using the ‘Refresh All’ option on the Data tab.



Our data is now updated with standard expense code names. We will still need to maintain the substitute table, but we have no need to trawl through our data to find any non-standard entries that don't already

appear on the substitute table, as an update will add any unrecognised entries to the substitute table. To prove this, let's add an 'A4 folder' to Paul's expenses ready to add it to our table:

Obviously, the update is not quite sure how to treat the extra information in my manual table, but a line has appeared and we need to adjust the data. A new row has appeared for Paul in the final table. We simply update our manual table and refresh again.

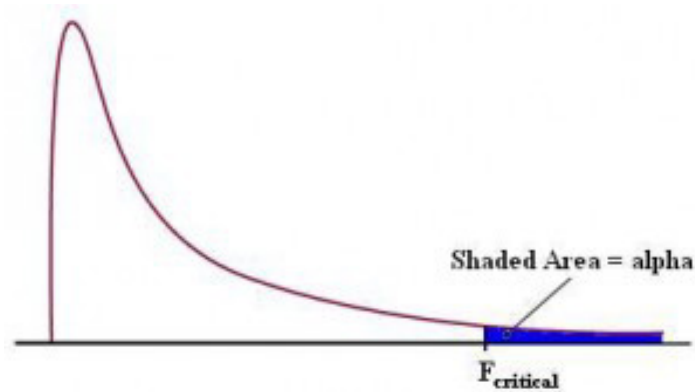
All the data is now showing the correct expense code, as required.
More next month!



Power BI Updates

All quiet on the Western front this month.
More next month we're sure!

The A to Z of Excel Functions: FINV



The F-distribution is defined as a continuous distribution obtained from the ratio of two chi-square distributions and used in particular to test the equality of the variances of two normally distributed variances a continuous probability distribution, which means that it is defined for an infinite number of different values. This analysis of variance is often referred to as “ANOVA”, which I still think is a cheap car.

In slightly simpler terms, the F-distribution can be used for several types of applications, including testing hypotheses about the equality of two population variances and testing the validity of what is known as a multiple regression equation, *i.e.* a formulaic attempt to model the relationship between two or more explanatory variables and a response variable by fitting a linear equation to observed data.

The F-distribution shares one important property with the Student’s t-distribution: probabilities are determined by a concept known as **degrees of freedom**. Unlike the Student’s t-distribution, the F-distribution is characterized by two different types of degrees of freedom: **numerator** and **denominator** degrees of freedom.

To calculate the F ratio, two estimates of the variance are made:

1. **Variance between samples:** An estimate of σ^2 that is the variance of the sample means multiplied by n (when the sample sizes are the same). If the samples are different sizes, the variance between samples is weighted to account for the different sample sizes. The variance is also called variation due to treatment or explained variation
2. **Variance within samples:** An estimate of σ^2 that is the average of the sample variances (also known as a pooled variance). When the sample sizes are different, the variance within samples is weighted. The variance is also called the variation due to error or unexplained variation:
 - SS_{between} = the sum of squares that represents the variation among the different samples
 - SS_{within} = the sum of squares that represents the variation within samples that is due to chance.

To find a “sum of squares” means to add together squared quantities that, in some cases, may be weighted. **MS** means “mean square”. MS_{between} is the variance between groups, and MS_{within} is the variance within groups.

To calculate the sum of squares and the mean square, let:

- k = the number of different groups
- n_j = the size of the j th group
- s_j = the sum of the values in the j th group
- n = total number of all the values combined (total sample size: $\sum n_j$)
- x = one value: $\sum x = \sum s_j$

The sum of squares of all values from every group combined: $\sum x^2$

The between group variability is therefore

$$SS_{\text{total}} = \sum x^2 - \frac{\sum x^2}{n}$$

and the total sum of squares is given by

$$\sum x^2 - \frac{(\sum x)^2}{n}$$

The F-distribution has two important properties:

1. It’s defined only for positive values.
2. It’s not symmetrical about its mean; instead, it’s positively skewed.

A distribution is positively skewed if the mean is greater than the median (the mean is the average value of a distribution, and the median is the midpoint; half of the values in the distribution are below the median and half are above). This function returns the (right-tailed) F probability distribution (degree of diversity) for two datasets.

This distribution is a relatively new one. The associated F statistic is a ratio (a fraction). There are two sets of degrees of freedom; one for the numerator and one for the denominator. As mentioned above, the F distribution is derived from the Student’s t-distribution. The values of the F distribution are squares of the corresponding values of the t-distribution.

The explained variation is therefore the sum of squares representing variation among the different samples:

$$SS_{\text{between}} = \sum \left[\frac{(s_j)^2}{n_j} \right] - \frac{(\sum s_j)^2}{n}$$

This should be compared with the unexplained variation, namely the sum of squares representing variation within samples due to chance:

$$SS_{\text{within}} = SS_{\text{total}} - SS_{\text{between}}$$

The degrees of freedom for different groups (*i.e.* the degrees of freedom for the numerator) are calculated as $df = k - 1$, whereas the equation for errors within samples (degrees of freedom for the denominator) is calculated as $df_{\text{within}} = n - k$.

Further, the mean square (variance estimate) explained by the different groups is

$$MS_{\text{between}} = \frac{SS_{\text{between}}}{df_{\text{between}}}$$

whereas the mean square (variance estimate) that is due to chance (unexplained) is

$$MS_{\text{within}} = \frac{SS_{\text{within}}}{df_{\text{within}}}$$

MS_{between} and MS_{within} can be written as follows:

$$MS_{\text{between}} = \frac{SS_{\text{between}}}{df_{\text{between}}} = \frac{SS_{\text{between}}}{k - 1}$$

$$MS_{\text{within}} = \frac{SS_{\text{within}}}{df_{\text{within}}} = \frac{SS_{\text{within}}}{n - k}$$

The one-way ANOVA test depends on the fact that MS_{between} can be influenced by population differences among means of the several groups. Since MS_{within} compares values of each group to its own group mean, the fact that group means might be different does not affect MS_{within} .

The null hypothesis says that all groups are samples from populations having the same normal distribution. The alternate hypothesis says that at least two of the sample groups come from populations with different normal distributions. If the null hypothesis (H_0) is TRUE, MS_{between} and MS_{within} should both estimate the same value.

To be clear, the null hypothesis says that all the group population means are equal. The hypothesis of equal means implies that the populations have the same normal distribution, because it is assumed that the populations are normal and that they have equal variances.

The foregoing calculations were done with groups of different sizes. If the groups are the same size, the calculations simplify somewhat, and the F-ratio can be written as:

$$F = \frac{n \cdot s_x^2}{s_{\text{pooled}}^2}$$

where

- n = the sample size
- $df_{\text{numerator}} = k - 1$
- $df_{\text{denominator}} = n - k$
- s_{pooled}^2 = the mean of the sample variances (pooled variance)
- s_x^2 = the variance of the sample means.

Now that you have had the statistics lecture, you may use the worksheet function **FINV** to return a value for the F-ratio when you supply an area under the curve (*i.e.* the associated probability), plus the number of degrees of freedom for both the numerator and denominator alike, that define the distribution.

So, all of this brings us to the F-ratio or F-statistic, defined as

$$F = \frac{MS_{\text{between}}}{MS_{\text{within}}}$$

If MS_{between} and MS_{within} estimate the same value (following the belief that H_0 is TRUE), then the F-ratio should be approximately equal to one (1). Mostly, just sampling errors would contribute to variations away from one. As it turns out, MS_{between} consists of the population variance plus a variance produced from the differences between the samples. MS_{within} is an estimate of the population variance. Since variances are always positive, if the null hypothesis is FALSE, MS_{between} will generally be larger than MS_{within} . Then the F-ratio will be larger than one. However, if the population effect is small, it is not unlikely that MS_{within} will be larger in a given sample.

The traditional approach has been to obtain a critical F-value early on in an experiment. The researcher would know how many groups would be involved, and would have some knowledge of how many individual observations would be available at the conclusion of the experiment. The variable **alpha**, the level of significance, would be chosen before any outcome data is available.

For example, let's consider a researcher who was testing four groups consisting of 10 people each, and that alpha was set to a level of 0.05 (5%). Then, an F-value could be looked up in a statistics text or else you could use the following formula to determine the critical F-value:

=FINV(0.05,3,36).

Important: This **FINV** has been replaced by **F.INV.RT** so that it may provide improved accuracy and whose names better reflect their usage. Although the previous function is still available for backward compatibility, you should consider using this newer function from now on, because the older variant may not be available in future versions of Excel, according to Microsoft.

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

FINV(probability,deg_freedom1,deg_freedom2)

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

- **probability:** this is required and represents the probability associated with the F-cumulative distribution
- **deg_freedom1:** this is also required and denotes the numerator degrees of freedom
- **deg_freedom2:** this is again required and represents the denominator degrees of freedom.

It should be further noted that:

- if any argument is non-numeric, **FINV** returns the #VALUE! error value
- if **probability** < 0 or **probability** > 1, **FINV** returns the #NUM! error value
- if **deg_freedom1** or **deg_freedom2** is not an integer, it is truncated
- If either **deg_freedom1** or **deg_freedom2** is strictly less than 1, or **deg_freedom1** or **deg_freedom2** is greater than or equal to 10^10, **FINV** returns the #NUM! error value.

FINV can be used to return critical values from the F distribution. For example, the output of an ANOVA calculation often includes data for the F-statistic, F probability, and F-critical value at the 0.05 significance level. To return the critical value of F, use the significance level as the probability argument to **FINV**.

Given a value for probability, **FINV** seeks that value x such that **FDIST(x,**

Further, from Excel 2010, you should use the following formula instead:

=F.INV(0.95,3,36).

These two formulae will return the same value, 2.866. The older function, **FINV**, returns the F-value that cuts off the rightmost 5% of the distribution; the newer **F.INV** function returns the F-value that cuts off the leftmost 95% of the distribution. Be careful!!

Once this has been determined (using either of these Excel functions), the ANOVA test may now be completed, with the calculated F compared to the critical F, and if the former is greater than the latter, the null hypothesis (H_0) will be rejected.

In summary, the **FINV** function returns the inverse of the (right-tailed) F probability distribution. If **p = FDIST(x,...)**, then **FINV(p,...) = x**. The F distribution can be used in an F-test that compares the degree of variability in two data sets.

deg_freedom1, deg_freedom2) = probability. Thus, precision of **FINV** depends on precision of **FDIST**. **FINV** uses an iterative search technique. If the search has not converged after 100 iterations, the function returns the #N/A error value.

Please see our example below:

	A	B	C
1	Data	Description	
2	0.050	Probability associated with the F-cumulative distribution	
3	3.000	Numerator degrees of freedom	
4	36.000	Denominator degree of freedom	
5			
6			
7	Formula	Description	Result
8	=FINV(A2,A3,A4)	Inverse of the F probability distribution for the terms above.	2.8663

The A to Z of Excel Functions: FISHER

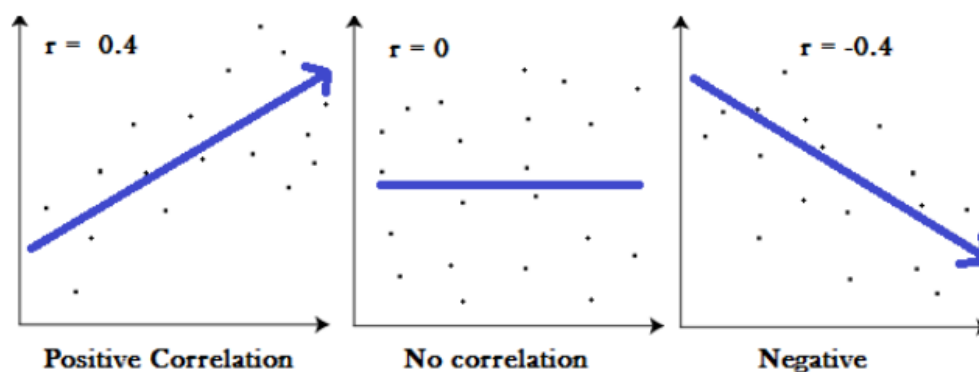
Correlation coefficients are used in statistics to measure how strong a relationship is between two variables. There are several types of correlation coefficient: Pearson's correlation (also called Pearson's **R**) is a correlation coefficient commonly used in linear regression. If you're starting out in statistics, you'll probably learn about Pearson's **R** first. In fact, when anyone refers to the correlation coefficient, they are usually talking about Pearson's **R**.

Correlation coefficient formulas are used to find how strong a relationship is between data. The formulas return a value between -1 and 1, where:

- 1 indicates a strong positive relationship (as one variable increases, so does the other generally)
- -1 indicates a strong negative relationship (as one variable increases, the other decreases generally)
- a result of zero (0) indicates no relationship at all.



Examples of correlation coefficients are as follows:



To be clear:

- a correlation coefficient of 1 means that for every positive increase in one variable, there is a positive increase of a fixed proportion in the other. For example, shoe sizes go up in (almost) perfect correlation with foot length
- a correlation coefficient of -1 means that for every positive increase in one variable, there is a negative decrease of a fixed proportion in the other. For example, the amount of gas in a tank decreases in (almost) perfect correlation with distance travelled since last filling the tank
- zero means that for every increase, there isn't a positive or negative increase; the two variables just aren't related.

The absolute value of the correlation coefficient gives us the relationship strength. The larger the number, the stronger the relationship. For example, $|-0.75| = 0.75$, which has a stronger relationship than 0.65 even though this is positive.

There are several types of correlation coefficient formulas. One of the most commonly used formulas in stats is Pearson's correlation coefficient formula. Given a set of n bivariate sample pairs (x_i, y_i) , $i = 1, \dots, n$, the sample correlation coefficient r is given by

$$r = \frac{n(\sum xy) - (\sum x)(\sum y)}{\sqrt{[n\sum x^2 - (\sum x)^2][n\sum y^2 - (\sum y)^2]}}$$

The correlation between sets of data is a measure of how well they are related. The most common measure of correlation in stats is this Pearson Correlation. The full name is the Pearson Product Moment Correlation (PPMC). It shows the linear relationship between two sets of data. In simple terms, it answers the question, can I draw a line graph to represent the data? Two letters are used to represent the Pearson correlation: Greek letter rho (ρ) for a population and the letter " r " for a sample.

To understand how this fits in with the **FISHER** function, let's consider a sampling distribution. This is a graph of a statistic for your sample data. Common examples include:

- mean
- mean absolute value of the deviation from the mean
- range
- standard deviation of the sample
- unbiased estimate of variance
- variance of the sample.

Most commonly when using charts, we plot numbers. For example, you might have graphed a data set and found it follows the shape of a normal distribution with a mean score of 100. Where probability distributions differ is that you aren't working with a single set of numbers; you're dealing with multiple statistics for multiple sets of numbers. If you find that concept hard to grasp you aren't alone.

While most people can imagine what the graph of a set of numbers looks like, it's much more difficult to imagine what stacks of, say, averages look like.

Let's start with a mean, like heights of financial modellers. Like many things in nature, heights follow a bell curve shape. Let's say the average

height was 5'9" (sorry, I am a Brit and I still think in feet and inches). If you had 10 modellers, you might get 5'9", 5'8", 5'10", 5'9", 5'7", 5'9", 5'9", 5'10", 5'7", and 5'9". The distribution of small samples may vary significantly (e.g. uniform or "scattered"), but for larger datasets, there is a mathematical trick we may employ.

In statistics, the Central Limit Theorem tells us that if you have lots of data, if you plot the average of these averages it will eventually approximate to a good, old-fashioned bell curve. That's the basis behind a sampling distribution: you take your average (or another statistic, like the variance) and you plot those statistics on a graph.

The **Fisher Z-Transformation** is simply a way to transform the sampling distribution of Pearson's r (i.e. the correlation coefficient) so that it becomes normally distributed. The " z " in Fisher Z stands for a **z-score**, and the formula to transform r to a z-score is given by

$$z' = .5 [\ln(1 + r) - \ln(1 - r)].$$

This is what the **FISHER** function does.

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

FISHER(r)

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

- **r**: this is required and represents the numeric value for which you want the transformation.

It should be noted that:

- if r is nonnumeric, **FISHER** returns the #VALUE! error value
- if $r \leq -1$ or if $r \geq 1$, **FISHER** returns the #NUM! error value.

Please see our example below:

	A	B	C
1	Data	Description	
2	0.500	Value used for transformation (r).	
3			
4			
5	Formula	Description	Result
6	=FISHER(A2)	Fisher transformation at 0.50.	0.5493061

The A to Z of Excel Functions: FISHERINV

As stated above, the **Fisher Z-Transformation** is simply a way to transform the sampling distribution of Pearson’s r (i.e. the correlation coefficient) so that it becomes normally distributed. The “z” in Fisher Z stands for a z-score, and the formula to transform r to a z-score is given by

$$z' = .5 [\ln(1 + r) - \ln(1 - r)].$$

That’s what the **FISHER** function does. The **FISHERINV** function is simply the inverse of this transformation, i.e. if **y = FISHER(x)**, then **x = FISHERINV(y)**. The formula is given by

$$x = \frac{e^{2y} - 1}{e^{2y} + 1}$$

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

FISHERINV(y)

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

- **y**: this is required and represents the numeric value for which you want to perform the inverse of the Fisher transformation.

It should be noted that:

- if **y** is nonnumeric, **FISHERINV** returns the #VALUE! error value.

Please see this month’s final example below:

	A	B	C
1	Data	Description	
2	0.549306	Fisher transformation value.	
3			
4			
5	Formula	Description	Result
6	=FISHERINV(A2)	Inverse of the Fisher transformation (r) at 0.549306.	0.5000000

More Excel Functions next month...

Beat the Boredom Suggested Solution

Earlier in this month’s newsletter, we set up the scenario that you had just finalised the budget for a project and (say) it started in Period 3 and ended in Period 8 as pictured:

Original Forecast

Assumptions

Period	#
Forecast	\$

Total	1	2	3	4	5	6	7	8
45			5	6	7	8	9	10

Suddenly, your boss told you the amounts needed to be reallocated on a “similar basis” but for Periods 4 to 15. As we recognised before the weekend that’s pretty easy, as this duration is double the original project length, so you would just attribute half of each period’s amount to the new periods, viz.

Total	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
45	-	-	-	2.50	2.50	3.00	3.00	3.50	3.50	4.00	4.00	4.50	4.50	5.00	5.00

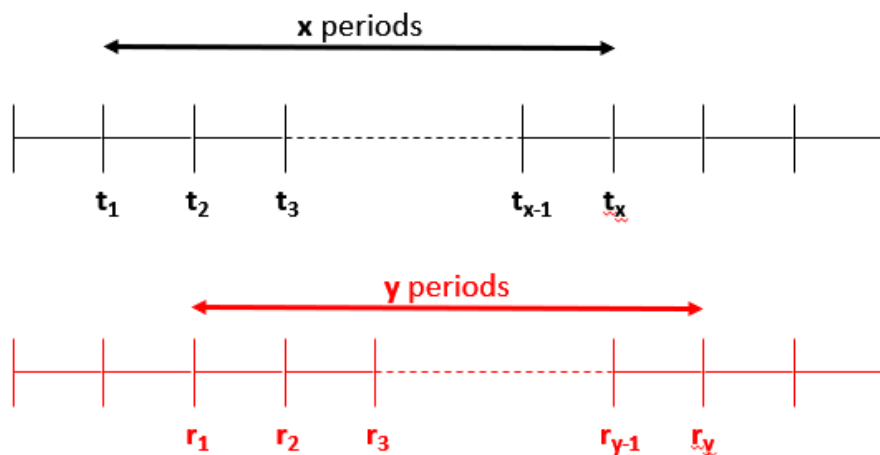
But that wasn’t the challenge. Instead, it was to create a spreadsheet that could cope with project advancements or delays and changes in duration at the same time. It sounds pretty horrible, but truth be told, finance staff face these types of challenge day-in, day-out.

Below, here’s how we went about it...

Suggested Solution

Assuming no inflationary factors to consider (*e.g.* time value of money), the problem boils down to pro-rating the original numbers across the new number of periods. The revised start and end dates tell you when the calculations begin, but in essence it is the number of periods in the revised forecast that drives the calculations.

Sorry for the algebra, but sometimes that’s what’s needed in a financial model! Let’s assume our original forecast has x periods going from start period t_1 to end period t_x , and the revised forecast has y periods going from revised start period r_1 to revised end period r_y .



In this illustration, r_1 occurs after t_1 , but this does not have to be true necessarily.

Regardless of start and finish dates (which simply governs when the calculations are made), there are basically three scenarios:

1. $x > y$, *i.e.* the revised forecast duration is shorter than the original one
2. $x < y$, *i.e.* the revised forecast duration is longer than the original one
3. $x = y$, *i.e.* the durations of both forecast periods are equal (this effectively simply moves the forecast period).

Let’s focus on the first scenario for a moment as it brings into focus how we could go about calculating the revised forecast. If the original duration was longer, then the revised forecast will consider the effects of more than one original period in each period, *e.g.*



In this graphic, the red boxes / yellow shading represent original periods and the blue boxes / borders denote a revised period. If $x > y$, then the blue box must straddle at least two red boxes. It could be more though, which is what is depicted here, where we have:

- a **start period**, where this is the proportion of the earliest original period considered
- **middle [or full] period(s)**, which (when $x > y$) are original periods that must be fully included. There could be more than one. If $x < y$, middle (full) period is not defined
- an **end period**, which is the proportion of the final original period considered.

Sounds confusing? Let's explain with an example:

1. Original Forecast

Original Forecast

Assumptions

Period #
Forecast \$

Start Period #
End Period #

Period
1
8

Total	1	2	3	4	5	6	7	8
36	1	2	3	4	5	6	7	8

$\{=MIN(IF(LU_Original_Forecast_Data<>0,LU_Periods))\}$
 $\{=MAX(IF(LU_Original_Forecast_Data<>0,LU_Periods))\}$

2. Revised Forecast

Revised Forecast

Assumptions

Start Period #
End Period #

Period	Check
4	
6	

Revised Forecast

Period #
Forecast \$

Total	1	2	3	4	5	6	7	8
36				5	12	19		

In the original forecasts, the cashflows of \$1 to \$8 (big spenders here!) were allocated across the first eight periods for a total of a rather exorbitant \$36. However, the revised forecast wanted the same profile over just periods 4 to 6 (three periods). That is, the start date t_1 is period 1, x is 8 and the final period t_x (t_8) is period 8.

The start and end dates (r_1 and r_3 , periods 4 and 6 respectively) for the revised forecast just denote when the forecast starts and stops. The key information is that there are only 3 (y) periods. This means that each period in the revised forecast includes $8/3$ (known as the **Period Factor** in the range name used) which equals two and two-thirds (2.67) periods of the old forecast data viz.

- Revised Period 4 = Old Period 1 + Old Period 2 + $2/3$ of Old Period 3 = $1 + 2 + (2 \times 3)/3 = 5$
- Revised Period 5 = $1/3$ of Old Period 3 + Old Period 4 + Old Period 5 + $1/3$ of Old Period 6 = $(1 \times 3)/3 + 4 + 5 + (1 \times 6)/3 = 12$
- Revised Period 6 = $2/3$ of Old Period 6 + Old Period 7 + Old Period 8 = $(2 \times 6)/3 + 7 + 8 = 19$.

You may identify which original periods are used in each revised period,

Forecast Allocation

Period #
Revised Flag [1,0]
Start #
End #

	1	2	3	4	5	6	7	8
				-	2.67	5.33		
				2.67	5.33	8.00		

what the start, middle / full and end periods are,

Forecast Allocation

Period #
Start Part Period #
Start Full Period #
End Full Period #
End Part Period #

	1	2	3	4	5	6	7	8
	-	-	-	-	3.00	6.00	-	-
	-	-	-	-	4.00	7.00	-	-
	-	-	-	2.00	5.00	8.00	-	-
	-	-	-	3.00	6.00	-	-	-

and what proportions to use of each:

Forecast Allocation

Period #
Start Part %
Full Part %
End Part %

	1	2	3	4	5	6	7	8
	-	-	-	-	33%	67%	-	-
	-	-	-	-	200%	200%	-	-
	-	-	-	67%	33%	-	-	-

These then cross-multiply the original forecast numbers for the appropriate periods using the **SUMIF** and **SUMIFS** functions to get the values explained above.

When the revised forecast period is longer than the original one, the problem is slightly simpler as there are no middle / full periods

(i.e. no period of original data is ever in just one revised period). Otherwise, the logic remains the same.

For those who are interested or are insomniacs, the detail is discussed below...

Devil's in the Detail

Let's use the example to talk through the formulae we used.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
5																	
6																	
7																	
8																	
9																	
10																	
11																	
12																	
13																	
14																	
15																	
16																	
17																	
18																	

The first section captures the original forecast (inputs) in cells **J13:Q13** and automatically computes the start and end period using the array formulae for **MIN(IF)** and **MAX(IF)** (cells **G16** and **G17** respectively). These must be entered using **CTRL + SHIFT + ENTER** as **IF** will not work across a range (an array) of cells otherwise.

The next section is the Revised Forecast assumptions:

	A	B	C	D	E	F	G	H
19								
20								
21								
22								
23								
24								
25								
26								
27								
28								
29								

This collects the required start and end periods in cells **G27** and **G28**, together with an error check in cell **H28** to ensure that the end period is not before the start period.

The first part of the next section simply collates all of the date to be used:

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
36																	
37																	
38																	
39																	
40																	
41																	
42																	
43																	
44																	
45																	
46																	
47																	
48																	
49																	
50																	
51																	
52																	
53																	
54																	
55																	
56																	

The key calculation here is the Period Factor (cell **H55**) which divides the original forecast duration by the revised forecast duration. This represents the number of original periods in each revised period and this is pivotal to all of the calculations.

The next part of this section works out how the original periods are reallocated to the revised periods:

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	Y
38																		
39																		
40																		
41																		
42																		
43																		
44																		
45																		
46																		
47																		
48																		
49																		
50																		
51																		
52																		
53																		
54																		
55																		
56																		

The Revised Flag (row 63) use the formula

$$=AND(J\$62>=\$H\$50,J\$62<=\$H\$51)*1$$

to check that the period counters in row 62 are greater than or equal to the revised start period (**H50**) and less than or equal to the revised end period (**H51**). The value is 1 if these assumptions are true and zero (0) otherwise.

The formula for the Start (row 64),

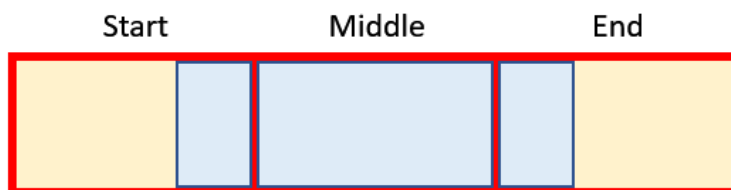
$$=IF(J\$63,I65+(\$G\$50-1)*(J\$62=\$H\$50),)$$

is a simple formula that takes the previous period's closing balance (as long as the flag is active) but also accounts for the fact that the original forecast may not have occurred in Period 1 (*i.e.* it sets the first period t_1 of the original forecast period).

The final formula for the End (row 65),

$$=IF(J\$63,J64+Period_Factor,)$$

simply adds the Period Factor to the Start period as long as the flag is active. This gives us ultimately the beginning and the end of the blue section in our graphic from before:



The next section starts working out which original periods need to be considered for the start, middle (full) and end:

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	Y
36																		
37																		
38																		
58																		
59																		
60																		
61																		
62																		
67																		
68																		
69																		
70																		
74																		

The Start Part Period uses the formula

$$=IF(ROUNDUP(J\$64,0)-J64,ROUNDUP(J\$64,0),)$$

Essentially, if Start (row 64) is an integer it uses that period number otherwise it uses the next period (**ROUNDUP(z,0)** rounds **z** up to zero decimal places, *i.e.* the next whole number).

Rows 68 and 69 establish the beginning and the end of the middle (full) period – sort of. Row 69, the calculation for the Start Part Period,

$$=IF(J\$64,ROUNDUP(J\$64,0)+1,)$$

adds one to the Start Part Period (row 67) (as long this is not zero) to avoid any double count. Row 69's formula for the End Full Period,

$$=ROUNDDOWN(J\$65,0)$$

takes the “beginning of the end”, that is, up to but not including the End period. Therefore, the way these two dates are calculated it is possible that the Start Full Period could be a period prior to the End Full Period. That is actually pictured in our example (*above*) and is acceptable – it simply means there is no full / middle period in that instance.

The final formula here (row 70) for End Part Period,

$$=IF(ROUNDUP(J\$65,0)-J65,ROUNDUP(J\$65,0),)$$

uses the same logic as per the Start Part Period. This means we now have the relevant original periods identified!

Next, we need to know what percentages should be used for Start and End Part Periods.

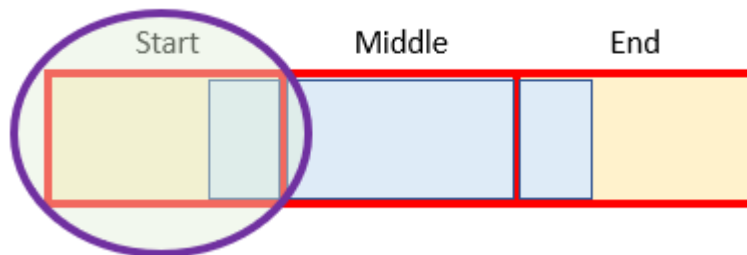
	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	Y	Z
36																			
37																			
38																			
58																			
59																			
60																			
61																			
62																			
72																			
73																			
74																			
76																			

The Full Part % is also calculated as it ensures the End Part % is not overstated.

The formula in row 72 for the Start Part %,

=MIN(MOD(ROUND(J67-J64,Rounding_Accuracy),1),Period_Factor)*J\$63

looks horrible but isn't as bad as it seems (honest)! **J67-J64** calculates the proportion Start Part Period less Start (*i.e.* this formula computes the proportion of the first red box that is blue).



ROUND is used to prevent rounding errors and **MOD** is incorporated to ensure this proportion is less than 100%.

The second formula is not pleasant either. The Full Part % (row 73) is given by

=MIN(IF(AND(J\$69>=J\$68,J\$68*J\$69<>0),MIN(Period_Factor,1)*(J\$69-J\$68+1),),Period_Factor-J\$72)

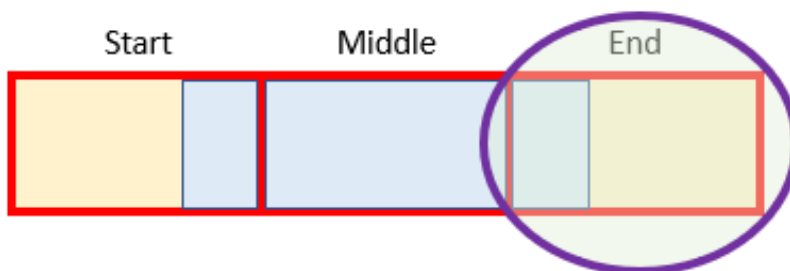
Erm, lovely... Again, once you get your head wrapped around it, it's not so bad. The two **IF** conditions required (inside the **AND** expression) check that the periods are not zero and that the end is not before the beginning (as discussed above). If this test is passed, it takes the **MIN(Period_Factor,1)** (you cannot count more than the forecast amount

in an original period) and multiplies this by the number of full original periods in the revised period. This is then restricted so that the sum of the Start Part % and the Full Part % cannot exceed the Period Factor. This number is calculated only to keep the End Part % honest. Talking of which...

The End Part % (row 74),

=MOD((Period_Factor-SUM(J72:J73))*J\$63,1)

just mops up the rest of the Period Factor where the flag is active. This is equal to the section highlighted:



This concludes the percentages needed. We now have identified which periods are the Start Middle and End and what proportions we require for the Start and End. "All" we have to do is multiply it out:

	A	B	C	D	E	F	J	K	L	M	N	O	P	Q	Y
36															
37															
38															
58															
59															
60															
61															
76															
77															
78															

3. Calculations									
Calculations									
Forecast Allocation									
	1	2	3	4	5	6	7	8	
Revised Forecast	0.53	0.60	1.07	1.20	1.60	1.80	2.13	2.40	=(SUMIF(LU_Periods,J\$67,LU_Original_Forecast_Data)*J\$72) +(SUMIFS(LU_Original_Forecast_Data,LU_Periods,">="&J\$68,LU_Periods,"<="&J\$69)*MIN(Period_Factor,1)) *(SUMIF(LU_Periods,J\$70,LU_Original_Forecast_Data)*J\$74)

I say "all" because we've left the best to last...

=(SUMIF(LU_Periods,J\$67,LU_Original_Forecast_Data)*J\$72)
+(SUMIFS(LU_Original_Forecast_Data,LU_Periods,">="&J\$68,LU_Periods,"<="&J\$69)*MIN(Period_Factor,1))
+(SUMIF(LU_Periods,J\$70,LU_Original_Forecast_Data)*J\$74)

Again, it's really not that bad! There are three calculations here = one each for the start, middle and end. The first one

=SUMIF(LU_Periods,J\$67,LU_Original_Forecast_Data)*J\$72

locates the original period to be used for Start and multiplies it by the appropriate proportion (**SUMIF** only sums the range **LU_Original_Forecast_Data** where the counter in **LU_Periods** is equal to the value in cell **J67**, *i.e.* the correct original period to be used).

The last formula,

=SUMIF(LU_Periods,J\$70,LU_Original_Forecast_Data)*J\$74

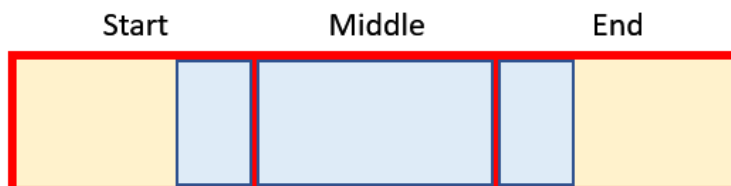
performs a similar operation for the End period. This just leaves

=SUMIFS(LU_Original_Forecast_Data,LU_Periods,">="&J\$68,LU_Periods,"<="&J\$69)*MIN(Period_Factor,1)

SUMIFS is used here as we need to sum based on two conditions, not one (that the full periods meet the conditions for the Start and End Full Periods). Here, you can clearly see if the End Full Period precedes the Start Full Period, no amounts will be summed. The factor **MIN(Period_**

Factor,1) is required when the number of revised periods is greater than the original number of forecast periods (so only the correct proportion) is used and to ensure the amount in a full period is never multiplied by a factor greater than 1 also.

These three added together give us our total:



Anyway, this was our method of solving the challenge. Did you come up with a simpler approach? If so, we'd love to hear about it so that we can steal it forever more and pretend we knew it all along...

Until next time.

Upcoming SumProduct Training Courses - COVID-19 update

Due to the COVID-19 pandemic that is currently spreading around the globe, we are suspending our in-person courses until further notice. However, to accommodate the new working-from-home dynamic, we are switching our public and in-house courses to an online delivery stream, presented via Microsoft Teams, with a live presenter running through the same course material, downloadable workbooks to complete the hands-on exercises during the training session, and a recording of the sessions for

your use within 1 month for you to refer back to in the event of technical difficulties. To assist with the pacing and flow of the course, we will also have a moderator who will help answer questions during the course.

If you're still not sure how this will work, please contact us at training@sumproduct.com and we'll be happy to walk you through the process.

Location	Course	Date	Date	Duration	Duration
Online (Australia)	Power Pivot, Power Query and Power BI	15 - 17 Feb 2021	09:00-17:00 AEDT	(-1 day) 22:00-07:00 GMT	3 Days
Online (Australia)	Excel Tips and Tricks	22 Feb 2021	09:00-17:00 AEDT	(-1 day) 22:00-07:00 GMT	1 Day
Online (Australia)	Financial Modelling	23 - 24 Feb 2021	09:00-17:00 AEDT	(-1 day) 22:00-07:00 GMT	2 Days
Online (Australia)	Power Pivot, Power Query and Power BI	7 - 9 Apr 2021	09:00-17:00 AEDT	(-1 day) 22:00-07:00 GMT	3 Days
Online (Australia)	Excel Tips and Tricks	14 Apr 2021	09:00-17:00 AEDT	(-1 day) 22:00-07:00 GMT	1 Day
Online (Australia)	Financial Modelling	15 - 16 Apr 2021	09:00-17:00 AEDT	(-1 day) 22:00-07:00 GMT	2 Days
Online (Australia)	Power Pivot, Power Query and Power BI	10 - 12 May 2021	09:00-17:00 AEDT	(-1 day) 22:00-07:00 GMT	3 Days
Online (Australia)	Excel Tips and Tricks	17 May 2021	09:00-17:00 AEDT	(-1 day) 22:00-07:00 GMT	1 Day
Online (Australia)	Financial Modelling	18 - 19 May 2021	09:00-17:00 AEDT	(-1 day) 22:00-07:00 GMT	2 Days
Online (Australia)	Power Pivot, Power Query and Power BI	15 - 17 Jun 2021	09:00-17:00 AEDT	(-1 day) 22:00-07:00 GMT	3 Days
Online (Australia)	Excel Tips and Tricks	22 Jun 2021	09:00-17:00 AEDT	(-1 day) 22:00-07:00 GMT	1 Day
Online (Australia)	Financial Modelling	23 - 24 Jun 2021	09:00-17:00 AEDT	(-1 day) 22:00-07:00 GMT	2 Days
Online (Australia)	Power Pivot, Power Query and Power BI	19 - 21 Jul 2021	09:00-17:00 AEDT	(-1 day) 22:00-07:00 GMT	3 Days
Online (Australia)	Excel Tips and Tricks	26 Jul 2021	09:00-17:00 AEDT	(-1 day) 22:00-07:00 GMT	1 Day
Online (Australia)	Financial Modelling	27 - 28 Jul 2021	09:00-17:00 AEDT	(-1 day) 22:00-07:00 GMT	2 Days
Online (Australia)	Power Pivot, Power Query and Power BI	23 - 25 Aug 2021	09:00-17:00 AEDT	(-1 day) 22:00-07:00 GMT	3 Days

Location	Course	Date	Date	Duration	Duration
Online (Australia)	Excel Tips and Tricks	30 Aug 2021	09:00-17:00 AEDT	(-1 day) 22:00-07:00 GMT	1 Day
Online (Australia)	Financial Modelling	31 Aug - 1 Sep 2021	09:00-17:00 AEDT	(-1 day) 22:00-07:00 GMT	2 Days
Online (Australia)	Power Pivot, Power Query and Power BI	29 Sep - 1 Oct 2021	09:00-17:00 AEDT	(-1 day) 22:00-07:00 GMT	3 Days
Online (Australia)	Excel Tips and Tricks	6 Oct 2021	09:00-17:00 AEDT	(-1 day) 22:00-07:00 GMT	1 Day
Online (Australia)	Financial Modelling	7 - 8 Oct 2021	09:00-17:00 AEDT	(-1 day) 22:00-07:00 GMT	2 Days
Online (Australia)	Power Pivot, Power Query and Power BI	3 - 5 Nov 2021	09:00-17:00 AEDT	(-1 day) 22:00-07:00 GMT	3 Days
Online (Australia)	Excel Tips and Tricks	10 Nov 2021	09:00-17:00 AEDT	(-1 day) 22:00-07:00 GMT	1 Day
Online (Australia)	Financial Modelling	11 - 12 Nov 2021	09:00-17:00 AEDT	(-1 day) 22:00-07:00 GMT	2 Days
Online (Australia)	Power Pivot, Power Query and Power BI	8 - 10 Dec 2021	09:00-17:00 AEDT	(-1 day) 22:00-07:00 GMT	3 Days
Online (Australia)	Excel Tips and Tricks	15 Dec 2021	09:00-17:00 AEDT	(-1 day) 22:00-07:00 GMT	1 Day
Online (Australia)	Financial Modelling	16 - 17 Dec 2021	09:00-17:00 AEDT	(-1 day) 22:00-07:00 GMT	2 Days

Key Strokes

Each newsletter, we'd like to introduce you to useful keystrokes you may or may not be aware of. This year, we thought we'd revisit each function key in depth (there are 12 – one for each month of the year!). Given it's February, let's look at the **F2** tips:

Keystroke	What it does
F2	Toggle Select, Edit and Enter / Point modes
ALT + F2	Save as
CTRL + F2	Print (Excel 2007 onwards)
SHIFT + F2	Insert / edit comment
ALT + SHIFT + F2	Save
CTRL + ALT + F2	Open
CTRL + ALT + SHIFT + F2	Print

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

Our Services

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

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

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

Link to Others

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

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

Any Questions?

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

Training

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

Check out our more popular courses in our training brochure:



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

Sydney Address: SumProduct Pty Ltd, Suite 803, Level 8, 276 Pitt Street, Sydney NSW 2000
New York Address: SumProduct Pty Ltd, 48 Wall Street, New York, NY, USA 10005
London Address: SumProduct Pty Ltd, Office 7, 3537 Ludgate Hill, London, EC4M 7JN, UK
Melbourne Address: SumProduct Pty Ltd, Ground Floor, 470 St Kilda Road, Melbourne, VIC 3004
Registered Address: SumProduct Pty Ltd, Level 6, 468 St Kilda Road, Melbourne, VIC 3004

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