## Named ranges and tables

How can we fill this in?

Graphical user interface, application, table

Description automatically generated

* Option A: One-at-a-time add the formula
* Option B: Used absolute references and drag the formulas over
* Option C: Give everything a name and operate on those names
  + Use either Formulas > Name Manager or the upper-left hand cell reference
  + Start typing the formula and hit tab to complete the desired object name
  + We will now have fixed\_cost + (variable\_cost \* units\_sold) and complete the entire analysis in one swoop:

Table

Description automatically generated with medium confidence

### Excel tables

What are some reasons to love Excel tables?

1. You **must** have column headers
   1. Ctrl + T to insert a table
   2. Table does NOT have headers
   3. You can dynamically refer to the headers. For example, let’s see how we could automatically convert these all to upper-case:

=UPPER(Table1[#Headers])

* 1. Or maybe we want to know the column position of the “island” variable:

=MATCH("island", Table1[#Headers], 0)

1. You must have a table name
   1. Go to the name manager and rename to penguins
   2. And now these references will point to penguins not Table1
2. They look good
   1. Table Design > Table Styles
   2. Also Table Style options, we will work on Totals later
   3. Everything is formatted for you – and it stays that way!
3. You can add and delete data easily
   1. Go ahead and add and delete some rows and columns, look how everything is expanded
      1. Do a RANDBETWEEN() to see how easy that was for a new column
      2. Manually insert a row and see how that is updated instantly
4. No rewriting formulas
   1. Take the sum of one of the columns and see how it doesn’t move! Even when we add new rows
5. We can point to data by name, not location
   1. Divide one of the columns by the other – how does this work?
6. We can easily work with totals
   1. Play around with the totals and see how easy it is
   2. Also show how this is NOT going to mess up other formulas
   3. Change different summary methods at the bottom
7. Portal into PivotTables & Power Query
   1. Let’s quickly demonstrate how to get in there

### IF() statements

Let’s practice on the commissions worksheet

* Column D: We can find which rep has >50 clients with a simple IF() statement: =IF(C2>50,"Yes","No")
* Column F: We want to basically “back out” the region from the District. If region is 1, 3 or 5, it’s District A, otherwise B.
  + This is like an IF() statement but with one more layer:
  + =IF(OR(E2=1, E2 = 3, E2 = 5),"A", "B")
* We could also combine this with a NOT() statement to find which stores are new, i.e. NOT old:
  + =IF(NOT(OR(E2=1,E2=2)),"X","")
* Finally, we could use an AND() statement to meet multiple criteria:
  + Column I: =IF(AND(B2>75000,H2>4.75),"X","")
  + Let’s practice on Column J: How can we flag which reps have fewer than 40 clients and a satisfaction score of <4?
    - =IF(AND(C2<40,H2<4),"X","")
* Return to slides to cover IFS()
* Let’s calculate the commission using the commission schedule over to the side. Based on a certain satisfaction score the rep gets a certain commission
  + We can do this gracefully with IFS():

=IFS(H2>4.5,0.1,

H2>4.25,0.05,

H2>4,0.03,

TRUE,0)

### IFS() and conditional aggregation

Let’s use conditional aggregation to answer these questions about the data:

* =SUMIF(F2:F16,"A",B2:B16)
* =COUNTIF(C2:C16,">50")
* =AVERAGEIF(H2:H16,"> 4.7",L2:L16)
* =AVERAGEIF(H2:H16,">= 4.7",L2:L16)
* =COUNTIFS(L2:L16,">5000",H2:H16,">4.5",C2:C16,">50")
* =SUMIFS(L2:L16,E2:E16,4,B2:B16,">50000")
* =AVERAGEIFS(L2:L16,C2:C16,"<50",H2:H16,">4.0")

## Dynamic arrays

UNIQUE() – This should probably really say “distinct,” but it will remove duplicates in the data.

* UNIQUE(computers\_clean) will give the number of unique rows
* Do ROWS(UNIQUE(computers\_clean)) = ROWS(computers\_clean)? Why?

FILTER() – We can filter to get resulting records.

* We only want the premium computers?
  + FILTER(computers\_clean, computers\_clean[premium] = "yes")
* Add the headers to the results: computers\_clean[#Headers]
* AND criteria with \*
  + FILTER(computers\_clean,(computers\_clean[premium]="yes")\*computers\_clean[speed]>30)
* OR criteria with +
  + FILTER(computers\_clean,(computers\_clean[premium]="yes+computers\_clean[speed]>30

SORTBY() – This function lets you sort by multiple columns and you can return whichever arrays you want. Operates a lot like SUMIFS().

* Sort by price, descending: =SORTBY(computers\_clean, computers\_clean[price], -1)
* Sort by speed descending and premium ascending: =SORTBY(computers\_clean, computers\_clean[speed], -1, computers\_clean[premium], 1)
* Sort by price, descending but *only* return the index column: =SORTBY(computers\_clean[Index], computers\_clean[price], -1)

### XLOOKUP()

* Let’s get the name of the location added to the employee table
  + This would be difficult with VLOOKUP() because of “looking to the left”
  + We can easily do it with XLOOKUP: =XLOOKUP([@locno], locations[locno], locations[cty])
* Let’s get the name of the department added to the employee table
  + With VLOOKUP() we would get a matching error and have to do something like IFERROR()
  + With XLOOKUP() we simply add another argument to the function: =XLOOKUP([@deptno],departments[deptno], departments[dept\_title], "OTHER")

### Dynamic visualizations with dynamic arrays

Worksheet: iris

* Dynamic arrays can be a bunch of different sizes, so it’s very possible their results will “spill” into a pre-populated area.
* For example, let’s try SEQUENCE(5) but put some text directly beneath it. What happens? We get a spill error
* If we want to refer to the dynamic array range *as a whole* we can do so with the spill operator, #
  + For example, we can reprint the array in A1 with =A1# or we could even operate on it as a whole with =SUM(A1#)
* Let’s create a dynamic scatterplot that you can filter and see the data for each group of iris.
* First thing we are going to do is create a dropdown that lists the three species:
  + Let’s create a helper worksheet iris\_helper and use the UNIQUE() function to list the unique species: =UNIQUE(iris[species])
* We can dynamically tie the dropdown to this array result with the spill operator:

Graphical user interface, text, application

Description automatically generated

* Now we can use the FILTER() function to get the resulting sepal length and widths for each:

Table

Description automatically generated

* Now, select the resulting sepal length vs width dataset and insert a scatterplot
* Go to Formulas > Name Manager > New
  + We will create a sepal\_length named range using the spill operator
* Right-click the resulting scatterplot > Edit > Select Data
* Specify the series X and Y values so that they refer to dynamic arrays with the spill operator # :  
  Graphical user interface, application

  Description automatically generated
* Right-click on the current scatterplot > Select Data > Edit
  + Keep the iris! worksheet preface in there but change the numbered ranges to your NAMED ranges   
    Graphical user interface, text, application

    Description automatically generated
  + You should now have a fully dynamic, interactive visualization right from Excel!

### LET(), LAMBDA() and the future of Excel

Let’s write the quadratic formula using 3 options in Excel.

1. Using formulas alone. The {-1, 1} is used to multiply by both -1 and 1 at the same time:

= (-$B$2 - {-1,1} \* SQRT($B$2^2 - 4 \* $B$1\* $B$3)) / (2 \*$B$1)

Not every elegant, right?! Let’s try NOT repeating ourselves with the LET() function:

=LET(a,B1,

b,B2,

c,B3,

(-b - {-1,1} \* (SQRT(b^2- 4\*a\*c)))/(2\*a))

Definitely getting better! Also we should definitely be including whitespace like this eh? 😊

Let’s see if we can do one better and actually create a new FUNCTION that we can reuse and get this stuff.

=LAMBDA(a,b,c,

(-b+{-1,1}\*SQRT(b^2-4\*a\*c))/

(2\*a))

We can evaluate it on our data like so:

=LAMBDA(a,b,c,

(-b+{-1,1}\*SQRT(b^2-4\*a\*c))/

(2\*a))(B1,B2,B3)

It’s a little counterproductive to write this function and then have to keep copy-pasting it every time we want to use it, so to get around this we can actually save our LAMBDA() function in the Name Manager!

