Lecture Note #5: Functions

BUSI 201: Business Data Analysis

Fall 2023

Topic 1. Functions: Embedding IF within IF

Sometimes, the condition we want to check may be too complicated to express in a single argument. One way users can check for multiple conditions is to embed another IF function within the main IF function. The second-order IF function can either be placed as the second or third argument of the main IF function. If the embedded IF function takes the place of the third argument, the form can be expressed as:

$$= \texttt{IF(C1, OUTPUT\_C1TRUE, IF(C2, OUTPUT\_C2TRUE, OUTPUT\_C1C2FALSE))}$$

Please note that there are three possible outcomes (errors notwithstanding) for this two-tiered IF function. Excel first checks if the first condition, C1, is met. If the conditions are deemed to be true, Excel will output OUTPUT_C1TRUE, and if false, it will move on to the embedded IF function to check the second condition. Please refer to the flowchart in Figure 1.
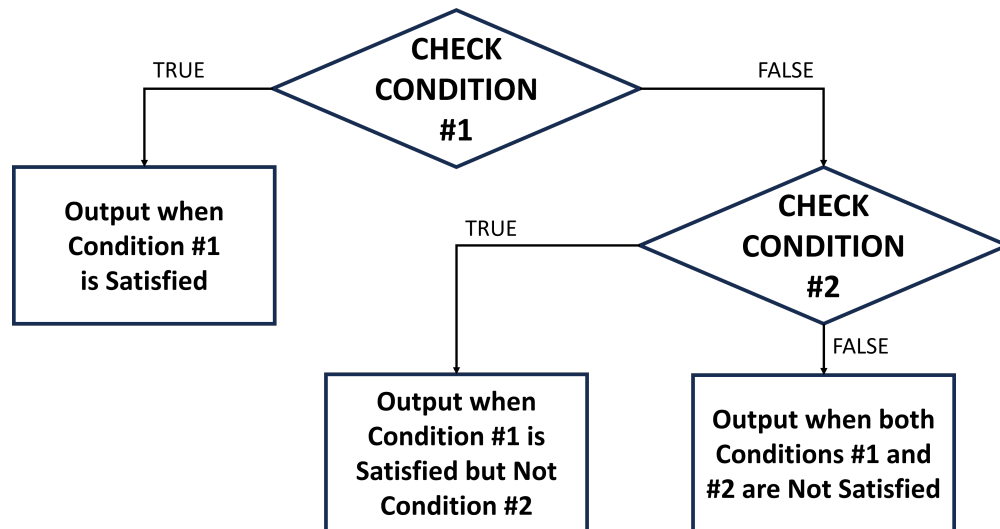

Figure 1: Flowchart of Embedded IF Functions

Navigate to the EMBED sheet of the workbook BUSI201-LEC05-Workbook.xlsx for a hands-on exercise. The worksheet contains a hypothetical list of 100 courses. Some of these courses are taught in-person, while others offer hybrid or online instruction.
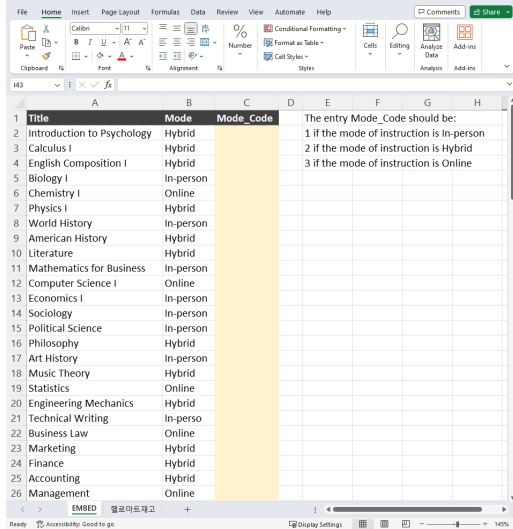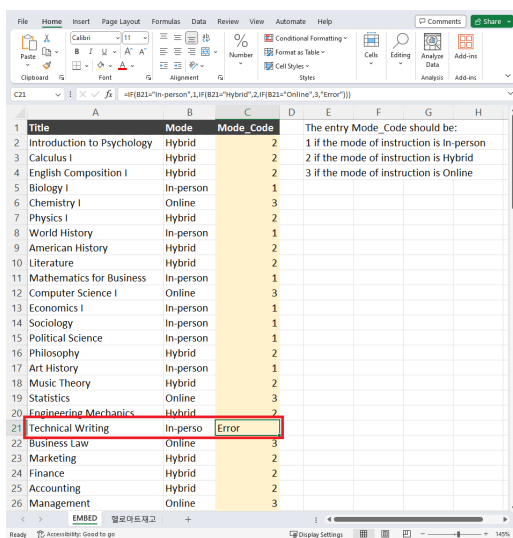
We want the empty Mode_Code column to be filled up with numerical values that represent the mode of instruction. Since there are three modes of instruction, we need to check for at least two conditions using the IF function. In this specific case, we can use the following formula for cell C2:

$$= IF(B2=\text{"In-person"}, 1, IF(B2=\text{"Hybrid"}, 2, 3))$$

Please take some time to try out this approach and check if the results you get from the embedded IF function are as intended. You may notice that the result in cell C21 is incorrect. Let's examine this mistake a bit closely to learn if there are some pitfalls from using the IF function.

Figure 2: Embedded IF Example

The primary reason that the result is incorrect is due to a typo in the original data. Instead of In-person, the last letter, n, was lost, and the input in cell B21 was In-perso. Therefore, the first condition of B21="In-person" was not met, and the second condition of B21="Hybrid" was also not met. Following this process, Excel labels the Mode_Code as 3.

This is one of the issues we may encounter when applying the IF function to perform logical tests. The embedded IF function, as it appears above, has a critical blind spot, where it assigns 3 to values that fail the first two logic tests without actually checking if the adjacent cell spells out Online. One way to fix this issue is to embed another IF function. Returning to cell B2:

$$= IF(B2=\text{"In-person"}, 1, IF(B2=\text{"Hybrid"}, 2, IF(B2=\text{"Online"}, 3, \text{"Error"})))$$

Applying the updated IF function to the example, we find that row 21 returns a customized error message instead of the wrong classification of 3. This should be considered best practice when dealing with vast datasets, where it is not feasible to manually check for any errors in the raw data. Applying safeguards such as the custom Error message, we can ensure that there are no misclassified items along with checking for potential errors in the data.

Figure 3: Embedded IF Example

Topic 2. Some Extensions of the `IF` Function

In the previous example, we learned how to make use of the `IF` function. Somewhat like an *extension* of the `IF` function, there are functions that perform calculations based on certain conditions set by the user. We will briefly cover three of these functions here: `SUMIFS`, `AVERAGEIFS`, `COUNTIFS`. These three functions will calculate the sum, average, and count of the cells that satisfy one or more conditions that the user can set. The syntax of these functions `SUMIFS` and `AVERAGEIFS` are identical:

$$= \texttt{SUMIFS(SUM\_ARRAY, COND\#1\_ARRAY, COND\#1, [COND\#2\_ARRAY, COND\#2], \cdots)}$$

The function reads as: "Find the sum of the cells in `SUM_ARRAY` which satisfy the condition `COND#1` in the region `COND#1_ARRAY`, and the condition `COND#2` in the region `COND#2_ARRAY`, and so forth." There are similar functions `SUMIF`, `AVERAGEIF`, and `COUNTIF` that can only handle one condition, instead of the many conditions that are allowed in the `SUMIFS`, `AVERAGEIFS`, and `COUNTIFS` functions. Therefore it may be redundant to use the more simple functions.

The syntax for the `COUNTIFS` would be:

$$= \texttt{COUNTIFS(COND\#1\_ARRAY, COND\#1, [COND\#2\_ARRAY, COND\#2], \cdots)}$$

We turn to another exercise to see how these functions work. Please open the sheet `IFS` in the workbook `BUSI201-LEC05-Workbook.xlsx`. The content of the worksheet will be a synthetic list of goods sold in a department store arranged by the date of the sale, the department which made the sale, the price of the item, and some information regarding the item itself.

Figure 4: Worksheet IFS

We will first turn our attention to the table with the red headings. This table is asking for us to find the number of sales made by each department, the total value of the sales by department, and the average price of each sale. These each require the `COUNTIFS`, `SUMIFS`, and `AVERAGEIFS` functions. Before turning to the next page for the solutions, please try to fill out each table on your own.

Figure 5: IFS Sheet Partially Filled

For cell `I3`, we use the `COUNTIFS` function to count the number of sales from the Home department:

$$= \texttt{COUNTIFS(\$F\$3:\$F\$52,H3)}$$

In cell `J3`, we use the `SUMIFS` function, and apply the `AVERAGEIFS` function in cell `K3`:

`= SUMIFS($D$3:$D$52,$F$3:$F$52,H3)`

`= AVERAGEIFS($D$3:$D$52,$F$3:$F$52,H3)`

We are using absolute references for the array of cells to be added and the array that contain information about our "conditions." The criteria itself is left as a relative reference, since we want the criteria to shift downward as we copy and paste the functions from the cells above.

The second table which is meant to be the monthly sales figure of the entire department store is more complicated than the previous table. Partially, it has to do with the difficulty in dealing with dates. Take some time observing the formatting of the months listed in column `H`. The full information stored in each cell is of the form: `YYYY/MM/DD`, but only shows us the month of the year due to its formatting being set to `Custom`→`mmmm`. This is required if we want to use the following approach for cell `I12`:

`= COUNTIFS($E$3:$E$52,">="&DATE(YEAR($H12),MONTH($H12),1),`
`          $E$3:$E$52,"<="&EOMONTH(DATE(YEAR($H12),MONTH($H12),1),0))`

We should break this function down bit by bit to see what is happening. First, the `$E$3:$E$52` portion is simply the array of items which are subject to checking the conditions. Then we move onto the portion that describes the condition to be checked in `">="&DATE(YEAR($H12),MONTH($H12),1)`.

The first part `">="` tells Excel that the condition to be met is that for the cell in `$E$3:$E$52` is to be greater than or equal to something to be described later. `&` exists to "link" the text argument `">="` to the upcoming function. `DATE(·)` is a function that can be used to input date information. The three required arguments of the `DATE` function are a number for the year, a number for the month, and a number for the day, in that order. The arguments in `DATE(·)` in our case rely on functions that extract the year and month information from an existing cell, the `YEAR(·)` and `MONTH(·)` functions, respectively, while the day is manually typed in as 1. To sum up, the first condition (first two arguments) in this `COUNTIFS` function is telling Excel to "The first condition that must be satisfied to be included in the count is that the corresponding date is to be more recent than March 1st of 2023." We must now close this `COUNTIFS` function by giving it a "sale had to be made before or on March 31st of 2023."

`"<="&EOMONTH(DATE(YEAR($H12),MONTH($H12),1),0)` is fulfilling that task. The only new function here is `EOMONTH` which can be interpreted as the "End Of MONTH" function, which requires two arguments. The first is the date to serve as the reference point, and the other is the number of months to be counted from that reference point. Here, we find the end of the month of March 2023.

The sum of all sales by month, and the average price of items sold in each month can be found using the `SUMIFS` and `COUNTIFS` functions:

```
= SUMIFS($D$3:$D$52,$E$3:$E$52,">="&DATE(YEAR($H12),MONTH($H12),1),
        $E$3:$E$52,"<="&EOMONTH(DATE(YEAR($H12),MONTH($H12),1),0))
```

For the final challenge given in the table with the purple header, we want to find the total sales by department-month. This will require us to use three conditions with the `SUMIFS` function. The formula for cell `N3` will be:

```
= SUMIFS($D$3:$D$52,
        $E$3:$E$52,">="&DATE(YEAR(N$2),MONTH(N$2),1),
        $E$3:$E$52,"<="&EOMONTH(DATE(YEAR(N$2),MONTH(N$2),1),0),
        $F$3:$F$52,$M3)
```

The first line simply informs Excel that we are using the `SUMIFS` function and specifies the values to be added up if they meet the conditions we will set up later. The second and third lines represent the date conditions for sales made in the month of March 2023. Please note that there are slight variations in how cells are referenced, as the date expands horizontally. The final line represents the condition that the sales must be from the Home department. This is how we can calculate the sum of sales from the Home department in the month of March 2023.

Figure 6: Worksheet IFS Last Table

Topic 3. Detour: Wildcard Characters

One way to broaden or refine search criteria is by using one of the three wildcard characters: *, ?, and ~. Suppose we have a dataset consisting of the names of US states. Each of these wildcard characters allows you to perform the following tasks:

Wildcard	Function
*	Replaces an arbitrary number of characters in the search. (i.e. North* will return North Carolina and North Dakota)
?	Replaces one character in the search. (i.e. N? returns NE, NV, NH, NJ, NM, NY, NC, ND)
~	Used to override other wildcards. (i.e. N~* returns N*)

These wildcard characters can be valuable when working with certain conditional functions. Please navigate to the `WILDCARD` sheet of `BUSI201-LEC05-Workbook.xlsx`. You should find a list of Accounting, Business, and Economics courses offered by a hypothetical college. Suppose you want to determine the number of courses offered by each department.
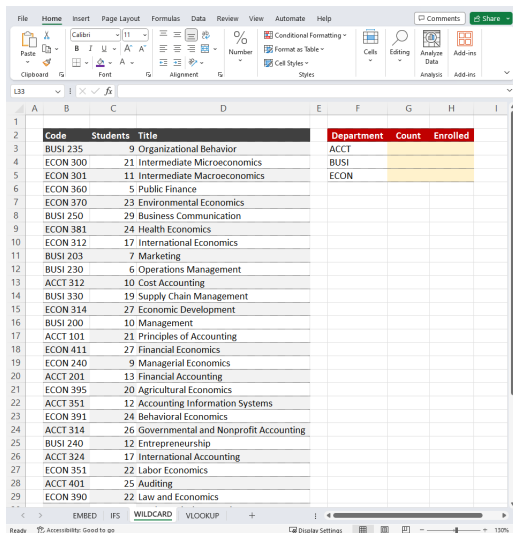

Figure 7: Wildcard Example

While there are other ways of achieving this goal, we will apply the `COUNTIFS` function with some wildcard characters included in the conditions. To find the number of courses offered by the Accounting department, we use the following function in cell G3:

$$= \text{COUNTIFS}(\$B\$3:\$B\$40, F3\&"*")$$

In order to find the total enrollment in each department, we can use the `SUMIFS` function as follows:

$$= \text{SUMIFS}(\$C\$3:\$C\$40, \$B\$3:\$B\$40, F3\&"*")$$

Topic 4. The VLOOKUP Function

Of all functions that we covered up to this point, the VLOOKUP function is one of the most useful. VLOOKUP is a function that searches a table for some specified item, and then returns the items in the same row as the item that was found. The basic syntax should be:

```
= VLOOKUP(LOOKUP VALUE,
          RANGE WHERE LOOKUP VALUE IS LOCATED,
          COLUMN NUMBER IN THE RANGE CONTAINING VALUE TO RETURN,
          [EXACT/APPROXIMATE MATCH])
```

It is sometime mildly confusing to simply read the syntax of a function, especially when there are multiple arguments. Please navigate to sheet VLOOKUP in the workbook BUSI201-LEC05-Workbook.xlsx. This worksheet contains a list of items sold iin a large retail store, with some internal item codes, make, item name, quantity in stock, and the price. Suppose that you want to look up information about a product based on the internal item codes.

Figure 8: VLOOKUP References

Here we demonstrate how to set up a VLOOKUP function that returns the market price of a product based on the internal item code, where the data is organized "vertically." Cell L3 should have an entry that resembles the following:

$$= \text{VLOOKUP}(\$H3, \$B\$3:\$F\$52, 5, 0)$$

Note that the first argument uses a mixed reference where the column is kept constant at H, as we may be copying and pasting the cell L3 to the left. Meanwhile, the second argument, which is the range of lookup, remains constant across all cases and uses absolute references. If all goes well, the function should return the price of the Samsung Smart LED TV, which is $599.99.

The final option of whether you would like to enforce an exact match or not comes with some conditions of its own. Users may choose to allow for an approximate match by specifying the last argument as 1. However, the exact method how VLOOKUP function handles these approximate matches is to return the value from the "previous" value. For instance, if individuals' ID numbers are sorted, and we are looking for individual 75 where individual 75 does not exist, VLOOKUP will return the values assigned to individual 74. This function is rarely used, but it is useful to know how Excel deals with some of these issues.