EXCEL WORKSHOP II

ADVANCED FORMULAS AND FUNCTIONS IN EXCEL
Table of Contents

- **Text Functions**: Text to Columns, LEFT, RIGHT, MID, FIND, LEN
- **Lookup Functions**: VLOOKUP, INDEX and MATCH
- **PivotTable Features**: Slicers and Timeline
- **Table Creation / Manipulation**: OFFSET
- **Optimization Functions**: Goal Seek, Solver
Text to Columns

- **What it does:** Separates data into columns either by fixed width or by a delimiter.
- **Where to find it:** Data → Text to Columns
- Very useful if we’re presented with raw data.
Other Text Functions

- **LEFT**(text, [num_chars]) returns the first character(s) in a string, based on the number of characters you specify.
- **RIGHT**(text,[num_chars]) returns the last character(s) in a string, based on the number of characters you specify.
- **MID**(text, start_num, num_chars) returns a specific number of characters from a text string, starting at the position you specify, based on the number of characters you specify.
  - **Note**: the final argument is the length, not the end index!
Other Text Functions

- **LEN(text)** returns the number of characters in a text string.
- **FIND(find_text, within_text, [start_num])** locates a substring within a string, and returns the index of the starting position of the substring from the first character of the string.
VLOOKUP

• Recall: VLOOKUP(lookup_value, table_array, col_index_num, [range_lookup])
  • lookup_value: the value you want to look up
  • table_array: the range of cells in which VLOOKUP will search for lookup_value and the return value
  • col_index_num: the column number that contains the return value
  • range_lookup (optional): TRUE (default) for approximate match (assumes the first column is sorted), FALSE for exact match

• HLOOKUP(lookup_value, table_array, row_index_num, [range_lookup]) does the same thing, but horizontally
Why learn INDEX MATCH?

• VLOOKUP requires us to count how many columns over the return value is found
• VLOOKUP can only search for lookup_value in the first column of table_array
• If we insert a column, VLOOKUP will either return the wrong column’s value or break completely
• INDEX MATCH allows us to look up across both rows and columns, whereas VLOOKUP and HLOOKUP only allow one dimension
INDEX

- **What it does**: Returns a value from within a table or range with given row and column number.
- **Syntax**: `INDEX(array, row_num, [column_num])`
- **Arguments**:
  - `array`: a range of cells
  - `row_num`: the number of the row relative to the array
  - `column_num` (optional): the number of the column relative to the array
MATCH

- **What it does**: Returns the position, relative to a selected range, of a specified item.
- **Syntax**: MATCH(lookup_value, lookup_array, [match_type])
- **Arguments**:
  - **lookup_value**: the value being looked up in `lookup_array`
  - **lookup_array**: the range of cells being searched
  - **match_type** (optional):
    - 1 (default): finds the largest value less than or equal to `lookup_value` (*lookup_array* **MUST** be in ascending order)
    - 0: finds the first value exactly equal to `lookup_value`
    - -1: finds the smallest value greater than or equal to `lookup_value` (*lookup_array* **MUST** be in descending order)
Combining INDEX and MATCH

- INDEX(array, row_num, [column_num])
- Instead of manually choosing a row_num and a column_num, we can use MATCH to fill this information in.
Example: VLOOKUP vs. INDEX MATCH

• VLOOKUP is essentially a special case of INDEX MATCH:
  • VLOOKUP(lookup_value, table_array, col_index_num, FALSE)
    is the same as
  • INDEX(table_array, MATCH(lookup_value, lookup_array, 0), col_index_num)

• If we wanted INDEX MATCH to do the same thing as VLOOKUP, lookup_array would simply be the first column of table_array.

• We can see INDEX MATCH gives much more freedom:
  • We can select any lookup_array we want
  • We can even replace col_index_num with another MATCH statement!
PivotTable Slicers and Timeline

- **What it does:** Allows you to filter your PivotTable data with more flexibility and ease of access than through the Filter option.
- **Where to find them:** Click inside your PivotTable, then Analyze → Insert Slicer/Timeline.
OFFSET

- **What it does**: Returns a reference to a range a specified number of rows and columns away from an “anchor” cell.

- **Syntax**: `OFFSET(reference, rows, cols, [height], [width])`

- **Arguments**:
  - `reference`: the cell about which the offset is based.
  - `rows`: the number of rows up (pos.) or down (neg.) you want the upper-left cell to refer to
  - `cols`: the number of columns right (pos.) or left (neg.) you want the upper-left cell to refer to
  - `height` (opt.): the number of rows in the returned reference (default: same as reference)
  - `width` (opt.): the number of columns in the returned reference (default: same as reference)
OFFSET: Returning a range of cells

• When we return a range of cells, we cannot visualize the range in a single cell.
• However, we can use formulas that take a range as input, including SUM, PRODUCT, AVERAGE, COUNT, etc.
Goal Seek

• **When to use it:** You know the result that you want from a formula, but are not sure what input value the formula needs to get that result

• Kind of like an automated “guess-and-check”

• **Where to find it:** Data → What-If Analysis → Goal Seek

• **How to use it:**
  - **Set cell:** The cell containing the formula you’re using (this formula should depend on your input for “by changing cell”)
  - **To value:** The value to which you want the formula in “set cell” to evaluate
  - **By changing cell:** The cell whose value you want to change
Shortcomings of Goal Seek

• If there are multiple input values that allow us to reach our target, Goal Seek only returns one (and we don’t know which one)
  • For example, we don’t want negative interest rates or loss ratios not between 0% and 100%
• Goal Seek only allows us to change one cell to reach our target
Solver

- **When to use it:** You want to optimize (maximize/minimize) or set the value of a formula to some result by changing one or more variable cells subject to one or more constraints
- Solver does everything Goal Seek can with a lot more power
- Solver is an **add-in**, and hence must be loaded into Excel