# Advanced Excel

Session 1: Working with Formulas & Functions

• Working with general & advance function (SUM, AVERAGE, MAX, MIN, LARGE &

• SMALL, SUMIF, SUMIFS, AVERAGEIF, AVERAGEIFS, COUNTIF, COUNTIFS)

• Writing conditional expressions (using IF)

• Using logical functions (AND, OR, NOT) inside IF, IFS

• Text Functions (LEFT, RIGHT, MID, LOWER, UPPER, PROPER, TEXT, CONCAT, TEXTJOIN)

Session 2: Working with VLookup, Xlookup & Index and Match

• Finding approximate or exact matches

• Using Match function in Vlookup

• Limitation of Vlookup

• Reverse Vlookup & Nested Vlookup

• Vlookup & Iferror function

• Joining two Vlookup using Concatenate function

• Working with Index & Match function

• Understanding Hlookup function

• Working with Xlookup function

Session 3: Data Validations

• Specifying a valid range of values for a cell

• Specifying a list of valid values for a cell

• Specifying custom validations based on formula for a cell

• Using dynamic range name in validation

Session 4: Protecting Worksheet Styles, Contents and Elements

• Protect & Unprotect The Worksheet

• Hide a Cell Formula

• Protect And Unprotect The Workbook Structure

• Set a Password To Open A Workbook

Session 5: Working with Pivot Tables

• Introducing Pivot Tables

• Creating a Pivot Table

• Modifying a Pivot Table

• Working with Pivot Table Reports

• Working with Page Fields

• Refreshing a PivotTable

Session 6: Automating Tasks Using Macro

• What Is a Macro?

• Record a Macro

• Work With Macros

• Provide Shortcut to a macro

• Assign Macro to toolbar or a button

• Edit A Macro

• Write a macro from scratch

# Study Material by Monica Gupta:

# Advanced Excel

## Working with advanced functions:

### Math & Statistical Functions (SUMIF, SUMIFS, AVERAGEIF, AVERAGEIFS, COUNTIF, COUNTIFS, SMALL, LARGE)

**1. SUMIF**

**Purpose:** Adds values in a range that meet a single condition.

* **Syntax:** =SUMIF(range, criteria, [sum\_range])
* **Example:**
  + =SUMIF(A1:A10, ">10", B1:B10)
  + Adds values in the range B1

where corresponding values in A1

are greater than 10.

**2. SUMIFS**

**Purpose:** Adds values in a range that meet multiple conditions.

* **Syntax:** =SUMIFS(sum\_range, criteria\_range1, criteria1, [criteria\_range2, criteria2], ...)
* **Example:**
  + =SUMIFS(B1:B10, A1:A10, ">10", C1:C10, "<5")
  + Adds values in B1

where values in A1

are greater than 10 and C1

are less than 5.

**3. AVERAGEIF**

**Purpose:** Averages values in a range that meet a single condition.

* **Syntax:** =AVERAGEIF(range, criteria, [average\_range])
* **Example:**
  + =AVERAGEIF(A1:A10, ">10", B1:B10)
  + Averages values in B1

where corresponding values in A1

are greater than 10.

**4. AVERAGEIFS**

**Purpose:** Averages values in a range that meet multiple conditions.

* **Syntax:** =AVERAGEIFS(average\_range, criteria\_range1, criteria1, [criteria\_range2, criteria2], ...)
* **Example:**
  + =AVERAGEIFS(B1:B10, A1:A10, ">10", C1:C10, "<5")
  + Averages values in B1

where values in A1

are greater than 10 and C1

are less than 5.

**5. COUNTIF**

**Purpose:** Counts the number of cells that meet a single condition.

* **Syntax:** =COUNTIF(range, criteria)
* **Example:**
  + =COUNTIF(A1:A10, ">10")
  + Counts the number of cells in A1

with values greater than 10.

**6. COUNTIFS**

**Purpose:** Counts the number of cells that meet multiple conditions.

* **Syntax:** =COUNTIFS(criteria\_range1, criteria1, [criteria\_range2, criteria2], ...)
* **Example:**
  + =COUNTIFS(A1:A10, ">10", B1:B10, "<5")
  + Counts the number of cells where A1

is greater than 10 and B1

is less than 5.

**7. SMALL**

**Purpose:** Returns the nth smallest value in a dataset.

* **Syntax:** =SMALL(array, n)
* **Example:**
  + =SMALL(A1:A10, 2)
  + Returns the second smallest value from A1

.

**8. LARGE**

**Purpose:** Returns the nth largest value in a dataset.

* **Syntax:** =LARGE(array, n)
* **Example:**
  + =LARGE(A1:A10, 3)
  + Returns the third largest value from A1

.

These functions are very useful for performing conditional and statistical calculations in Excel, especially for data analysis.

### Math & Statistical Functions (MINIFS, MAXIFS) (New)

**1. MINIFS**

**Purpose:** Returns the smallest value from a range that meets one or more conditions.

* **Syntax:**  
  =MINIFS(min\_range, criteria\_range1, criteria1, [criteria\_range2, criteria2], ...)
* **Example:**  
  =MINIFS(B1:B10, A1:A10, ">10", C1:C10, "<5")  
  This returns the smallest value from the range B1

, where the corresponding values in A1

are greater than 10, and the corresponding values in C1

are less than 5.

**2. MAXIFS**

**Purpose:** Returns the largest value from a range that meets one or more conditions.

* **Syntax:**  
  =MAXIFS(max\_range, criteria\_range1, criteria1, [criteria\_range2, criteria2], ...)
* **Example:**  
  =MAXIFS(B1:B10, A1:A10, ">10", C1:C10, "<5")  
  This returns the largest value from the range B1

, where the corresponding values in A1

are greater than 10, and the corresponding values in C1

are less than 5.

These functions are useful when you need to find the minimum or maximum values based on multiple conditions in a dataset.

Logical Functions (IF, Nested IF, IFERROR, AND, OR)

Here is an overview of **Logical Functions** in Excel:

**1. IF**

**Purpose:** Returns one value if a condition is TRUE and another value if it is FALSE.

* **Syntax:**  
  =IF(logical\_test, value\_if\_true, value\_if\_false)
* **Example:**  
  =IF(A1 > 10, "Yes", "No")  
  This checks if the value in A1 is greater than 10. If TRUE, it returns "Yes"; otherwise, it returns "No."

**2. Nested IF**

**Purpose:** Allows multiple conditions to be tested in a single formula, where the value returned depends on several IF conditions.

* **Syntax:**  
  =IF(condition1, value\_if\_true1, IF(condition2, value\_if\_true2, value\_if\_false))
* **Example:**  
  =IF(A1 > 90, "A", IF(A1 > 80, "B", "C"))  
  This checks if the value in A1 is greater than 90. If TRUE, it returns "A." If FALSE, it checks if the value is greater than 80. If TRUE, it returns "B." Otherwise, it returns "C."

**3. IFERROR**

**Purpose:** Returns a custom result if a formula generates an error; otherwise, it returns the result of the formula.

* **Syntax:**  
  =IFERROR(value, value\_if\_error)
* **Example:**  
  =IFERROR(A1/B1, "Division by zero error")  
  This divides A1 by B1. If there’s an error (e.g., B1 is zero), it returns "Division by zero error."

**4. AND**

**Purpose:** Returns TRUE if all the conditions are TRUE; otherwise, returns FALSE.

* **Syntax:**  
  =AND(logical1, [logical2], ...)
* **Example:**  
  =AND(A1 > 10, B1 < 5)  
  This returns TRUE if the value in A1 is greater than 10 **and** the value in B1 is less than 5. Otherwise, it returns FALSE.

**5. OR**

**Purpose:** Returns TRUE if any of the conditions are TRUE; otherwise, returns FALSE.

* **Syntax:**  
  =OR(logical1, [logical2], ...)
* **Example:**  
  =OR(A1 > 10, B1 < 5)  
  This returns TRUE if either the value in A1 is greater than 10 **or** the value in B1 is less than 5.

**Combining Logical Functions:**

You can combine these logical functions to create more complex conditions.

* **Example of IF with AND/OR:**  
  =IF(AND(A1 > 10, B1 < 5), "Pass", "Fail")  
  This returns "Pass" if both conditions are TRUE (A1 > 10 **and** B1 < 5). Otherwise, it returns "Fail."

These logical functions help in making decisions and performing conditional calculations in Excel based on multiple criteria.

### Logical Function IFS (New)

The **IFS** function is a more flexible and concise alternative to using multiple **IF** functions (nested IFs). It allows you to test multiple conditions and return a value corresponding to the first condition that is TRUE.

**IFS**

**Purpose:** Tests multiple conditions and returns a value corresponding to the first TRUE condition.

* **Syntax:**  
  =IFS(logical\_test1, value\_if\_true1, [logical\_test2, value\_if\_true2], ...)
* **Explanation:**
  + logical\_test1: The first condition to test.
  + value\_if\_true1: The value to return if the first condition is TRUE.
  + You can add more conditions as needed (logical\_test2, value\_if\_true2, etc.).
  + If no conditions are TRUE, the function returns a #N/A error unless a catch-all condition is provided (e.g., TRUE as the final test).

**Example 1:**

=IFS(A1 > 90, "A", A1 > 80, "B", A1 > 70, "C", TRUE, "F")  
This checks:

* If the value in A1 is greater than 90, return "A."
* If A1 is greater than 80, return "B."
* If A1 is greater than 70, return "C."
* If none of the above conditions are met, return "F" (the catch-all condition using TRUE).

**Example 2:**

=IFS(B1="Apples", 5, B1="Bananas", 10, B1="Cherries", 15)  
This returns 5 if B1 contains "Apples," 10 if B1 contains "Bananas," and 15 if B1 contains "Cherries."

**Advantages of IFS over Nested IF:**

* Easier to read and write when there are multiple conditions.
* Avoids deeply nested parentheses, making complex formulas more straightforward.

However, **IFS** doesn't have a built-in else condition (like in traditional IF statements), so you should explicitly handle scenarios where none of the conditions are TRUE (e.g., using TRUE as the last logical\_test).

### Text Functions (TEXT, SUBSTITUTE, CODE, CHAR, CLEAN, TRIM, LEFT, RIGHT, MID, LOWER, UPPER, PROPER, TEXT, CONCATENATE, SEARCH, VALUE)

Here is an overview of common **Text Functions** in Excel:

**1. TEXT**

**Purpose:** Converts a number to text in a specified format.

* **Syntax:**  
  =TEXT(value, format\_text)
* **Example:**  
  =TEXT(A1, "mm/dd/yyyy")  
  This converts the date in A1 into the format "MM/DD/YYYY."

**2. SUBSTITUTE**

**Purpose:** Replaces existing text with new text in a string.

* **Syntax:**  
  =SUBSTITUTE(text, old\_text, new\_text, [instance\_num])
* **Example:**  
  =SUBSTITUTE(A1, "Excel", "Sheets")  
  This replaces the word "Excel" with "Sheets" in the text of A1.

**3. CODE**

**Purpose:** Returns the numeric code for the first character in a text string.

* **Syntax:**  
  =CODE(text)
* **Example:**  
  =CODE(A1)  
  This returns the numeric code of the first character in A1 (e.g., "A" would return 65).

**4. CHAR**

**Purpose:** Returns the character corresponding to a numeric code.

* **Syntax:**  
  =CHAR(number)
* **Example:**  
  =CHAR(65)  
  This returns the character associated with the code 65, which is "A."

**5. CLEAN**

**Purpose:** Removes all non-printable characters from a text string.

* **Syntax:**  
  =CLEAN(text)
* **Example:**  
  =CLEAN(A1)  
  This cleans the text in A1 by removing non-printable characters.

**6. TRIM**

**Purpose:** Removes extra spaces from a text string, leaving only single spaces between words.

* **Syntax:**  
  =TRIM(text)
* **Example:**  
  =TRIM(A1)  
  This removes any extra spaces from the text in A1.

**7. LEFT**

**Purpose:** Extracts a specified number of characters from the beginning (left side) of a text string.

* **Syntax:**  
  =LEFT(text, [num\_chars])
* **Example:**  
  =LEFT(A1, 5)  
  This extracts the first 5 characters from the text in A1.

**8. RIGHT**

**Purpose:** Extracts a specified number of characters from the end (right side) of a text string.

* **Syntax:**  
  =RIGHT(text, [num\_chars])
* **Example:**  
  =RIGHT(A1, 3)  
  This extracts the last 3 characters from the text in A1.

**9. MID**

**Purpose:** Extracts a specified number of characters from the middle of a text string, starting at a given position.

* **Syntax:**  
  =MID(text, start\_num, num\_chars)
* **Example:**  
  =MID(A1, 3, 4)  
  This extracts 4 characters starting from the 3rd character in A1.

**10. LOWER**

**Purpose:** Converts all characters in a text string to lowercase.

* **Syntax:**  
  =LOWER(text)
* **Example:**  
  =LOWER(A1)  
  This converts all the text in A1 to lowercase.

**11. UPPER**

**Purpose:** Converts all characters in a text string to uppercase.

* **Syntax:**  
  =UPPER(text)
* **Example:**  
  =UPPER(A1)  
  This converts all the text in A1 to uppercase.

**12. PROPER**

**Purpose:** Capitalizes the first letter of each word in a text string.

* **Syntax:**  
  =PROPER(text)
* **Example:**  
  =PROPER(A1)  
  This capitalizes the first letter of each word in A1.

**13. TEXT**

**Purpose:** Formats a number as text using a specific format.

* **Syntax:**  
  =TEXT(value, format\_text)
* **Example:**  
  =TEXT(A1, "$#,##0.00")  
  This formats the number in A1 as currency with two decimal places.

**14. CONCATENATE (or CONCAT)**

**Purpose:** Combines (joins) two or more text strings into one.

* **Syntax:**  
  =CONCATENATE(text1, text2, ...)  
  (In newer versions of Excel, use CONCAT(text1, text2, ...))
* **Example:**  
  =CONCATENATE(A1, " ", B1)  
  This joins the text in A1 and B1 with a space between them.

**15. SEARCH**

**Purpose:** Returns the position of a substring within a text string (case-insensitive).

* **Syntax:**  
  =SEARCH(find\_text, within\_text, [start\_num])
* **Example:**  
  =SEARCH("Excel", A1)  
  This returns the position of the substring "Excel" in the text of A1.

**16. VALUE**

**Purpose:** Converts a text string that represents a number into a numeric value.

* **Syntax:**  
  =VALUE(text)
* **Example:**  
  =VALUE(A1)  
  This converts the text in A1 to a numeric value.

These **Text Functions** are helpful for manipulating, cleaning, and formatting text data in Excel, making it easier to work with text strings and convert them into usable forms.

### Text Functions TEXTJOIN, CONCAT (New)

Here’s an overview of the **TEXTJOIN** and **CONCAT** functions, which are commonly used to combine text strings in Excel:

**1. TEXTJOIN**

**Purpose:** Combines (joins) multiple text strings with a specified delimiter (separator) and allows you to ignore empty cells.

* **Syntax:**  
  =TEXTJOIN(delimiter, ignore\_empty, text1, [text2], ...)
* **Explanation:**
  + delimiter: The character(s) to insert between each text string (e.g., space, comma, etc.).
  + ignore\_empty: TRUE to ignore empty cells, FALSE to include them.
  + text1, [text2], ...: The text items to join (you can specify individual cells or ranges).
* **Example 1:**  
  =TEXTJOIN(", ", TRUE, A1:A5)  
  This joins the text in the range A1

, using a comma and space as the delimiter, and ignores empty cells.

* **Example 2:**  
  =TEXTJOIN(" - ", FALSE, "Apple", "Banana", "Cherry")  
  This joins the words "Apple," "Banana," and "Cherry," with a hyphen between them, without ignoring any empty strings.

**2. CONCAT**

**Purpose:** Combines (concatenates) multiple text strings into one. It's an improvement over the old **CONCATENATE** function and can handle ranges.

* **Syntax:**  
  =CONCAT(text1, [text2], ...)
* **Explanation:**
  + text1, [text2], ...: The text strings or ranges to combine.
* **Example 1:**  
  =CONCAT(A1, B1)  
  This combines the text from cells A1 and B1.
* **Example 2:**  
  =CONCAT(A1:A3)  
  This combines the text from the range A1

into a single string.

* **Key Difference from TEXTJOIN:**  
  **CONCAT** doesn’t allow you to specify a delimiter or ignore empty cells, making it less flexible than **TEXTJOIN** for certain tasks.

**Comparison:**

* **TEXTJOIN**: Offers more flexibility by allowing a delimiter and ignoring empty cells.
* **CONCAT**: A simplified version that just combines text without delimiters or ignoring empty values.

These functions are useful when you need to merge or combine text from different cells or ranges efficiently.

# Working with VLookup, Xlookup & Index and Match

• Finding approximate or exact matches

• Using Match function in Vlookup

• Limitation of Vlookup

• Reverse Vlookup & Nested Vlookup

• Vlookup & Iferror function

• Joining two Vlookup using Concatenate function

• Working with Index & Match function

• Understanding Hlookup function

• Working with Xlookup function

### Lookup/Ref Functions (VLOOKUP, HLOOKUP, INDEXMATCH MATCH, FORMULATEXT)

Here's a detailed overview of commonly used **Lookup/Reference Functions** in Excel, including **VLOOKUP**, **HLOOKUP**, **INDEXMATCH MATCH**, and **FORMULATEXT**:

**1. VLOOKUP**

**Purpose:** Searches for a value in the first column of a range and returns a value in the same row from another column.

* **Syntax:**  
  =VLOOKUP(lookup\_value, table\_array, col\_index\_num, [range\_lookup])
* **Explanation:**
  + lookup\_value: The value to search for.
  + table\_array: The range of cells that contains the data (lookup table).
  + col\_index\_num: The column number (in the table) from which to retrieve the value.
  + range\_lookup: TRUE for an approximate match or FALSE for an exact match.
* **Example:**  
  =VLOOKUP(A1, B2:E10, 3, FALSE)  
  This searches for the value in A1 in the first column of the range B2

and returns the value from the third column of that range.

**2. HLOOKUP**

**Purpose:** Searches for a value in the first row of a range and returns a value from the same column in another row.

* **Syntax:**  
  =HLOOKUP(lookup\_value, table\_array, row\_index\_num, [range\_lookup])
* **Explanation:**
  + lookup\_value: The value to search for.
  + table\_array: The range of cells that contains the data (lookup table).
  + row\_index\_num: The row number (in the table) from which to retrieve the value.
  + range\_lookup: TRUE for an approximate match or FALSE for an exact match.
* **Example:**  
  =HLOOKUP(A1, B1:E10, 3, FALSE)  
  This searches for the value in A1 in the first row of the range B1

and returns the value from the third row of that range.

**3. INDEXMATCH MATCH**

**INDEXMATCH MATCH** is a more flexible alternative to VLOOKUP and HLOOKUP, combining the **INDEX** and **MATCH** functions to perform powerful lookups.

**INDEX**

**Purpose:** Returns the value of a cell in a specified row and column.

* **Syntax:**  
  =INDEX(array, row\_num, [column\_num])
* **Explanation:**
  + array: The range of cells to search.
  + row\_num: The row number of the value to return.
  + column\_num: The column number (optional for 1-column ranges).
* **Example:**  
  =INDEX(A1:C10, 2, 3)  
  This returns the value from the second row and third column of the range A1

.

**MATCH**

**Purpose:** Returns the relative position of an item in a range.

* **Syntax:**  
  =MATCH(lookup\_value, lookup\_array, [match\_type])
* **Explanation:**
  + lookup\_value: The value to search for.
  + lookup\_array: The range to search.
  + match\_type: 0 for an exact match, 1 for less than, -1 for greater than.
* **Example:**  
  =MATCH(A1, B2:B10, 0)  
  This searches for the value in A1 in the range B2

and returns the relative position of the first exact match.

**Combining INDEX and MATCH**

By combining these two functions, you can perform more complex lookups, avoiding some of the limitations of VLOOKUP (e.g., column restrictions).

* **Example (INDEXMATCH):**  
  =INDEX(C2:C10, MATCH(A1, B2:B10, 0))  
  This looks for the value in A1 in the range B2

and returns the corresponding value from C2

.

**Using Double MATCH for Row and Column Lookup**

**Purpose:** This approach allows for a dynamic lookup across both rows and columns.

* **Syntax:**  
  =INDEX(array, MATCH(row\_value, row\_array, 0), MATCH(column\_value, column\_array, 0))
* **Example:**  
  =INDEX(B2:E10, MATCH(A1, A2:A10, 0), MATCH(B1, B2:E2, 0))  
  This performs a lookup based on matching values in both rows and columns, similar to a two-dimensional lookup.

**4. FORMULATEXT**

**Purpose:** Returns the formula in a referenced cell as text.

* **Syntax:**  
  =FORMULATEXT(reference)
* **Explanation:**
  + reference: The cell or range containing the formula to display as text.
* **Example:**  
  =FORMULATEXT(A1)  
  This returns the actual formula used in cell A1 as a text string (e.g., "=SUM(B1

)").

**Key Differences:**

* **VLOOKUP/HLOOKUP**: Simple vertical or horizontal lookups but limited by column position and requiring static ranges.
* **INDEXMATCH MATCH**: A more flexible and dynamic approach, allowing for horizontal, vertical, and two-dimensional lookups, with no column order limitation.
* **FORMULATEXT**: Helps document or display formulas in other cells.

These functions are essential for efficient data retrieval and formula management in Excel.

**1. XLOOKUP**

**Purpose:** Performs a lookup and returns a value from a range (either horizontally or vertically). It's more flexible and powerful than **VLOOKUP** and **HLOOKUP**.

* **Syntax:**  
  =XLOOKUP(lookup\_value, lookup\_array, return\_array, [if\_not\_found], [match\_mode], [search\_mode])
* **Explanation:**
  + lookup\_value: The value to search for.
  + lookup\_array: The range to search in.
  + return\_array: The range from which to return a result.
  + if\_not\_found: The value to return if no match is found (optional).
  + match\_mode: Specify exact or approximate match (optional, default is exact).
  + search\_mode: Specify search direction (optional, default is search from top).
* **Example 1:**  
  =XLOOKUP(A1, B2:B10, C2:C10)  
  This looks for the value in A1 in the range B2

and returns the corresponding value from C2

.

* **Example 2:**  
  =XLOOKUP(A1, B2:B10, C2:C10, "Not Found")  
  This returns "Not Found" if the lookup value in A1 isn't found in B2

**1. Finding Approximate or Exact Matches with VLOOKUP**

=VLOOKUP(lookup\_value, table\_array, col\_index\_num, [range\_lookup])

* TRUE → Approximate Match (sorted list)
* FALSE → Exact Match

Example (Exact):

=VLOOKUP("John", A2:C10, 2, FALSE)

**2. Using MATCH Function in VLOOKUP**

Dynamic column index using MATCH:

=VLOOKUP("John", A1:D10, MATCH("Age", A1:D1, 0), FALSE)

This gets the column number of "Age" dynamically.

**3. Limitations of VLOOKUP**

* Only searches **left to right**
* Breaks if columns are **inserted**
* Slower with large datasets
* Doesn’t support multiple criteria directly

**4. Reverse VLOOKUP (Lookup Left)**

Use INDEX + MATCH:

=INDEX(A2:A10, MATCH("John", B2:B10, 0))

Looks up "John" in column B and returns value from column A.

**5. Nested VLOOKUP**

One VLOOKUP inside another:

=VLOOKUP(VLOOKUP(102, A2:B4, 2, FALSE), D2:E4, 2, FALSE)

Looks up name from Emp ID, then salary from name.

**6. VLOOKUP + IFERROR**

Handles errors gracefully:

=IFERROR(VLOOKUP("David", A2:B10, 2, FALSE), "Not Found")

**7. Joining Two VLOOKUPs Using CONCATENATE**

Get first and last name using two VLOOKUPs:

=VLOOKUP(101, A2:C10, 2, FALSE) & " " & VLOOKUP(101, A2:C10, 3, FALSE)

**8. Working with INDEX & MATCH**

More flexible than VLOOKUP:

=INDEX(C2:C10, MATCH("John", A2:A10, 0))

* Can lookup left
* Doesn't break with column insertion
* Supports multi-criteria (with array formulas)

**9. Understanding HLOOKUP**

Looks **horizontally** (row-wise):

=HLOOKUP("Math", A1:G3, 2, FALSE)

Finds "Math" in row 1 and returns value from row 2.

**10. Working with XLOOKUP (Excel 365 / 2019+)**

Syntax:

=XLOOKUP(lookup\_value, lookup\_array, return\_array, [if\_not\_found], [match\_mode], [search\_mode])

Example:

=XLOOKUP("John", A2:A10, C2:C10, "Not Found")

* Replaces VLOOKUP, HLOOKUP, INDEX-MATCH
* Can lookup **left or right**
* Handles **exact and approximate** matches
* Can return multiple values

# Data Validations

# • Specifying a valid range of values for a cell

# • Specifying a list of valid values for a cell

# • Specifying custom validations based on formula for a cell

# • Using dynamic range name in validation

# Data Validations

Data Validation ensures that only specific types of data are entered into a cell, helping maintain data accuracy.

**a. Setting Up Data Validation**

1. Select the cell or range you want to validate.
2. Go to the **Data** tab → **Data Validation**.
3. In the **Settings** tab, choose the type of validation (e.g., Whole Number, Decimal, List, Date, etc.).
4. Define the criteria for validation and click **OK**.

**b. Using a Drop-Down List**

You can create a drop-down list using data validation to limit the entries to a predefined list of values.

**Example:**

* To create a drop-down list of departments (HR, IT, Finance):
  + Select the cells where you want the list.
  + Go to **Data** → **Data Validation** → **List**.
  + In the **Source** box, type: HR, IT, Finance.
  + Now, when you click on a cell, you'll see a drop-down list with these options.

**c. Error Messages for Invalid Data**

You can also set up error messages to inform users when they enter invalid data.

* In the **Data Validation** dialog, go to the **Error Alert** tab.
* Enter a **Title** and **Error Message**.

**Example:**

* If you want only numbers between 1 and 100, set up a validation rule for whole numbers and display an error message if the user enters something outside this range.

**Specifying a Valid Range of Values for a Cell**

**Use Case**: Only allow numbers between 10 and 100 in a cell.

**Steps:**

1. Select the cell(s)
2. Go to **Data > Data Validation**
3. Under **Allow**, choose **Whole number** (or Decimal)
4. Set **Minimum**: 10, **Maximum**: 100
5. Click OK

Now only values between 10–100 are accepted.

**Specifying a List of Valid Values for a Cell**

**Use Case**: Drop-down with fixed items (e.g., Apple, Banana, Orange)

**Steps:**

1. Select the cell(s)
2. Go to **Data > Data Validation**
3. Under **Allow**, choose **List**
4. In **Source**:

Apple, Banana, Orange

1. Click OK

Users will now see a drop-down with the list.

**Specifying Custom Validations Based on Formula**

**Use Case**: Allow only values that are even numbers.

**Steps:**

1. Select the cell
2. Go to **Data > Data Validation**
3. Under **Allow**, choose **Custom**
4. In **Formula**:

=MOD(A1, 2)=0

*(Assuming A1 is the current cell)*

You can use **any formula** that returns TRUE/FALSE:

* Only if today is a Monday: =WEEKDAY(TODAY(),2)=1
* Value must be greater than another cell: =A1>B1

**Using Dynamic Range Name in Validation**

**Use Case**: Drop-down list that updates automatically as new items are added.

**Example List in A2:A10**

(May grow beyond A10)

**Step 1: Define a Dynamic Named Range**

1. Go to **Formulas > Name Manager > New**
2. Name: MyList
3. Refers to:

=OFFSET(Sheet1!$A$2,0,0,COUNTA(Sheet1!$A:$A)-1,1)

**Step 2: Use in Validation**

1. Select the input cell
2. Go to **Data > Data Validation > List**
3. Source:

=MyList

The drop-down will now grow with your list.

**Bonus Tip: Use Excel Table**

Convert the list to a **Table** (Ctrl + T) and name the column.  
Then use:

=INDIRECT("TableName[ColumnName]")

# Protecting Worksheets Styles, Contents and Elements

## Protect & Unprotect the Worksheet

## Unlock Cells for Editing a Protected Sheet

## Hide a Cell Formula

## Set a Password to Edit a Specific Range

## Set a Password to Open a Workbook

## Remove Password from a Workbook

**Protecting Worksheets, Styles, Contents, and Elements**

In Excel, protecting worksheets helps prevent unauthorized changes to the data. You can protect the entire sheet or specific elements within it, such as formatting, content, or formulas. Additionally, you can set passwords to restrict access to certain parts of the workbook.

**1. Protect & Unprotect the Worksheet**

**a. Protecting the Worksheet**

When you protect a worksheet, Excel prevents users from making changes to the content, formatting, or elements of the sheet unless you specifically unlock certain cells.

**Steps to Protect a Worksheet:**

1. Go to the **Review** tab in the Excel ribbon.
2. Click on **Protect Sheet**.
3. In the **Protect Sheet** dialog box, check the options you want to allow users to perform (e.g., Select locked cells, Format columns, etc.).
4. Enter a password (optional) to prevent others from unprotecting the sheet.
5. Click **OK** and re-enter the password if prompted (if you set one).

**Example:**

* You have a worksheet that contains sensitive financial data. By protecting the sheet, you prevent anyone from accidentally modifying the data or formulas.

**b. Unprotecting the Worksheet**

To unprotect a worksheet, you simply need to remove the protection. If a password was set, you’ll need to enter it.

**Steps to Unprotect the Worksheet:**

1. Go to the **Review** tab.
2. Click **Unprotect Sheet**.
3. If prompted, enter the password you previously set.

**Example:**

* After completing data analysis or when you need to make changes, you can unprotect the sheet to allow edits.

**2. Unlock Cells for Editing a Protected Sheet**

By default, when you protect a worksheet, all cells are locked, meaning no one can edit them. However, you can unlock specific cells for editing while keeping others protected.

**Steps to Unlock Cells for Editing:**

1. Select the cells that you want to remain editable (e.g., input fields, data entry sections).
2. Right-click the selected cells and choose **Format Cells**.
3. Go to the **Protection** tab.
4. Uncheck the **Locked** option.
5. Click **OK**.
6. Now, protect the sheet as mentioned earlier. Only the unlocked cells can be edited.

**Example:**

* In a budget tracking sheet, you may want to allow users to enter values in specific cells (e.g., expenses), but protect the rest of the sheet to prevent accidental changes.

**3. Hide a Cell Formula**

Sometimes, you may want to hide the formulas in cells so that users can see the results but not the underlying formula. This is useful for protecting your formulas and preventing unauthorized edits.

**Steps to Hide a Formula:**

1. Select the cell or range of cells that contain the formulas you want to hide.
2. Right-click and select **Format Cells**.
3. In the **Format Cells** dialog box, go to the **Protection** tab.
4. Check the **Hidden** box.
5. Click **OK**.
6. Now, protect the sheet. The formulas will be hidden from view, but users can still see the results.

**Example:**

* In a budget sheet, you may want to hide the formulas used to calculate totals, but let users see the final calculated amounts.

**4. Set a Password to Edit a Specific Range**

You can set a password that allows only specific users to edit a designated range of cells. This is useful when you want to restrict editing to certain sections of the worksheet.

**Steps to Set a Password to Edit a Specific Range:**

1. Select the range of cells you want to allow specific users to edit.
2. Go to the **Review** tab and click on **Allow Users to Edit Ranges**.
3. In the **Allow Users to Edit Ranges** dialog box, click **New**.
4. Define the range of cells and set a password if desired.
5. Click **OK** to save the range and set the password.

**Example:**

* In an expense report, you may want only the finance team to edit the cells containing budget amounts, while the rest of the worksheet remains protected.

**5. Set a Password to Open a Workbook**

If you want to prevent unauthorized access to an entire workbook, you can set a password that must be entered to open the workbook.

**Steps to Set a Password to Open a Workbook:**

1. Go to the **File** tab and click on **Info**.
2. Click on **Protect Workbook** → **Encrypt with Password**.
3. Enter a password in the **Password** box and click **OK**.
4. Re-enter the password to confirm it and click **OK**.

**Example:**

* If you have a financial report that should only be accessible to authorized personnel, you can set a password to prevent unauthorized users from opening the workbook.

**6. Remove Password from a Workbook**

If you no longer want a password on a workbook, you can remove it by following these steps.

**Steps to Remove a Password from a Workbook:**

1. Open the workbook that is password-protected.
2. Go to the **File** tab and click **Info**.
3. Click **Protect Workbook** → **Encrypt with Password**.
4. Delete the existing password in the **Password** box.
5. Click **OK** to remove the password.

**Example:**

* Once a project is completed and you no longer need to restrict access to the workbook, you can remove the password for easier access by all users.

**Summary of Protection Features**

| **Feature** | **Description** | **Steps to Implement** | **Example** |
| --- | --- | --- | --- |
| **Protect Worksheet** | Prevents changes to the entire worksheet | **Review** → **Protect Sheet**, set a password if needed | Protecting a financial sheet to prevent accidental edits |
| **Unlock Cells for Editing** | Allows certain cells to be editable while protecting others | Unlock cells via **Format Cells** → **Protection** | Allowing data entry while protecting formulas |
| **Hide Formulas** | Hides formulas but allows users to see results | **Format Cells** → **Protection** → **Hidden**, protect sheet | Hiding complex formulas in a budget sheet |
| **Set Password to Edit Range** | Restricts editing of a range to authorized users only | **Review** → **Allow Users to Edit Ranges**, set password | Restricting edits to budget cells in a report |
| **Password to Open Workbook** | Prevents unauthorized users from opening the workbook | **File** → **Info** → **Protect Workbook** → **Encrypt** | Setting a password to protect sensitive project files |
| **Remove Password** | Removes password protection from a workbook | **File** → **Info** → **Protect Workbook** → **Encrypt** | Removing password protection after project completion |

By utilizing these protection features in Excel, you can ensure that your data remains secure, while still allowing authorized users to interact with specific parts of the workbook.

## Pivot table

### Creating Pivot tables

### Formatting and customizing Pivot tables

### Using advanced options of Pivot tables

### Creating group in Pivot tables

### Pivot charts

### Slicers

Here's a comprehensive guide on working with **Pivot Tables** in Excel, including creating, formatting, and customizing them, along with using Pivot Charts and Slicers.

**1. Creating Pivot Tables**

**Steps to Create a Pivot Table:**

1. **Select Your Data:**
   * Ensure your data is organized in a tabular format with headers. Select any cell within the data range.
2. **Insert Pivot Table:**
   * Go to the **Insert** tab in the Ribbon.
   * Click on **PivotTable**.
3. **Create PivotTable Dialog Box:**
   * Choose whether to place the Pivot Table in a new worksheet or an existing worksheet.
   * Click **OK**.
4. **Build the Pivot Table:**
   * In the **PivotTable Field List**, drag fields to the Rows, Columns, Values, and Filters areas to arrange the data as desired.

**2. Formatting and Customizing Pivot Tables**

**Basic Formatting:**

1. **Select the Pivot Table:**
   * Click anywhere in the Pivot Table to activate it.
2. **Design Tab:**
   * Go to the **PivotTable Analyze** or **Design** tab in the Ribbon.
3. **Choose a Style:**
   * Click on **PivotTable Styles** to select a predefined style for your table.

**Customizing Pivot Tables:**

1. **Adjust Field Settings:**
   * Right-click on any value in the Pivot Table and select **Value Field Settings** to change how values are summarized (e.g., Sum, Average).
2. **Format Numbers:**
   * Right-click on the values and choose **Number Format** to format them (e.g., currency, percentage).
3. **Add or Remove Subtotals:**
   * Click on a Row or Column field, go to the **Design** tab, and use the **Subtotals** dropdown to adjust subtotal settings.

**3. Using Advanced Options of Pivot Tables**

1. **Refreshing Data:**
   * If your source data changes, click on the Pivot Table and go to **PivotTable Analyze** > **Refresh**.
2. **Grouping Data:**
   * You can group data based on dates or categories.
   * Right-click on a Row or Column field and select **Group** to group items (e.g., by months or years).
3. **Sorting Data:**
   * Click on any Row or Column label, then go to the **Data** tab and choose **Sort Ascending** or **Sort Descending**.
4. **Using Calculated Fields:**
   * In the **PivotTable Analyze** tab, click on **Fields, Items & Sets**, and select **Calculated Field** to create a new field based on a calculation of existing fields.

**4. Creating Groups in Pivot Tables**

**Steps to Group Data:**

1. **Select the Field to Group:**
   * Right-click on the Row or Column field you want to group (e.g., a date field).
2. **Choose Grouping Options:**
   * Select **Group** from the context menu.
   * For date fields, you can specify to group by months, quarters, or years.
3. **Confirm Grouping:**
   * Click **OK** to create the groups in the Pivot Table.

**5. Creating Pivot Charts**

1. **Select Pivot Table:**
   * Click anywhere inside your Pivot Table.
2. **Insert Pivot Chart:**
   * Go to the **PivotTable Analyze** tab and click on **PivotChart**.
3. **Choose Chart Type:**
   * Select the desired chart type and click **OK**.
4. **Customize the Pivot Chart:**
   * Use the **Chart Tools** (Design and Format tabs) to modify the chart appearance and layout.

**6. Using Slicers**

Slicers are a great way to filter Pivot Tables visually.

**Steps to Add Slicers:**

1. **Select the Pivot Table:**
   * Click anywhere inside the Pivot Table.
2. **Insert Slicer:**
   * Go to the **PivotTable Analyze** tab and click on **Insert Slicer**.
3. **Choose Fields:**
   * In the dialog box, select the fields for which you want to add slicers and click **OK**.
4. **Using Slicers:**
   * Click on the buttons in the slicers to filter the data in the Pivot Table dynamically.
5. **Formatting Slicers:**
   * Right-click on a slicer to access formatting options, allowing you to adjust the style and layout of the slicers.

**Conclusion**

Pivot Tables are a powerful tool in Excel for data analysis, enabling you to summarize, analyze, and visualize data efficiently. This guide covers the essentials of creating, formatting, and customizing Pivot Tables, as well as utilizing Pivot Charts and Slicers for enhanced data interaction. By mastering these features, you can significantly improve your data analysis capabilities in Excel.

## Macros

### What Is A Macro

### Displaying the Developer Tab

### Recording and executing macros

### Understanding different types of references in macros

### Saving Workbooks with Macros

### Assigning macros to toolbars or menu items

### Switch Scenarios and Views with Macros

### Use of Worksheet Buttons to Trigger Macros

### Prompting for User Input

### Using the If…Then…Else Statement

Here’s a comprehensive guide on **Macros** in Excel, covering the basics, recording and executing macros, user input, and control structures like If…Then…Else statements.

**1. What Is a Macro?**

A macro in Excel is a sequence of instructions or actions that automate repetitive tasks. By recording a macro, you can execute a set of commands with a single click, improving efficiency and accuracy.

**2. Displaying the Developer Tab**

The Developer tab is necessary for accessing macro features.

**Steps to Enable the Developer Tab:**

1. **Go to Excel Options:**
   * Click on **File** in the Ribbon.
   * Select **Options**.
2. **Customize the Ribbon:**
   * In the Excel Options dialog, select **Customize Ribbon**.
   * In the right pane, check the box for **Developer**.
3. **Click OK:**
   * The Developer tab will now appear in the Ribbon.

**3. Recording and Executing Macros**

**Recording a Macro:**

1. **Open the Developer Tab:**
   * Click on the **Developer** tab in the Ribbon.
2. **Record Macro:**
   * Click on **Record Macro**.
   * Enter a name for the macro, a shortcut key (optional), and choose where to store it (e.g., **This Workbook**).
3. **Perform Actions:**
   * Execute the tasks you want to automate. Excel will record these actions.
4. **Stop Recording:**
   * Click **Stop Recording** in the Developer tab.

**Executing a Macro:**

1. **Use Shortcut Key:**
   * Press the assigned shortcut key if set.
2. **Run from Developer Tab:**
   * Go to **Developer** > **Macros**.
   * Select the macro and click **Run**.

**4. Understanding Different Types of References in Macros**

**Types of References:**

* **Relative References:** Records actions based on the current selection or active cell. Use this when you want the macro to adapt to different locations.
* **Absolute References:** Records actions based on specific cell addresses. Use this when you want the macro to apply to fixed locations.

**How to Set:**

* When recording a macro, you can toggle between relative and absolute references by clicking **Use Relative References** in the Developer tab.

**5. Saving Workbooks with Macros**

**Saving Workbooks:**

1. **Save As:**
   * Go to **File** > **Save As**.
2. **Choose File Type:**
   * Select **Excel Macro-Enabled Workbook (\*.xlsm)** from the file type dropdown.
3. **Save the File:**
   * Enter a file name and click **Save**.

**6. Assigning Macros to Toolbars or Menu Items**

**Steps to Assign Macros:**

1. **Open Customize Ribbon:**
   * Right-click on the Ribbon and select **Customize the Ribbon**.
2. **Add New Tab/Group:**
   * Click on **New Tab** or select an existing tab, then click on **New Group**.
3. **Select Macros:**
   * In the left pane, choose **Macros** from the dropdown.
4. **Add Macros:**
   * Select the macro you want to add and click **Add**.
5. **Click OK:**
   * The macro will now appear in the selected tab or group.

**7. Switching Scenarios and Views with Macros**

You can use macros to switch between different scenarios or views in your workbook.

**Example Macro to Switch Views:**

Sub SwitchView()

ActiveWindow.View = xlNormalView ' or xlPageBreakPreview

End Sub

**8. Use of Worksheet Buttons to Trigger Macros**

**Steps to Add a Button:**

1. **Insert a Button:**
   * Go to the **Developer** tab, click on **Insert**, and choose **Button (Form Control)**.
2. **Draw the Button:**
   * Click and drag to draw the button on the worksheet.
3. **Assign Macro:**
   * The Assign Macro dialog box will open; select the macro you want to run when the button is clicked.
4. **Format the Button:**
   * Right-click the button to change text, size, or format.

**9. Prompting for User Input**

You can prompt users for input using the InputBox function.

**Example:**

Sub PromptForInput()

Dim userInput As String

userInput = InputBox("Please enter your name:", "User Input")

MsgBox "Hello, " & userInput

End Sub

**10. Using the If…Then…Else Statement**

Control flow in macros can be managed with If…Then…Else statements.

**Example:**

Sub CheckValue()

Dim cellValue As Integer

cellValue = Range("A1").Value

If cellValue > 10 Then

MsgBox "Value is greater than 10."

ElseIf cellValue < 10 Then

MsgBox "Value is less than 10."

Else

MsgBox "Value is equal to 10."

End If

End Sub

**Conclusion**

Macros are a powerful tool in Excel for automating repetitive tasks and improving productivity. This guide provides an overview of creating, recording, and executing macros, as well as advanced features like user input and control structures. By mastering these techniques, you can enhance your Excel experience significantly.