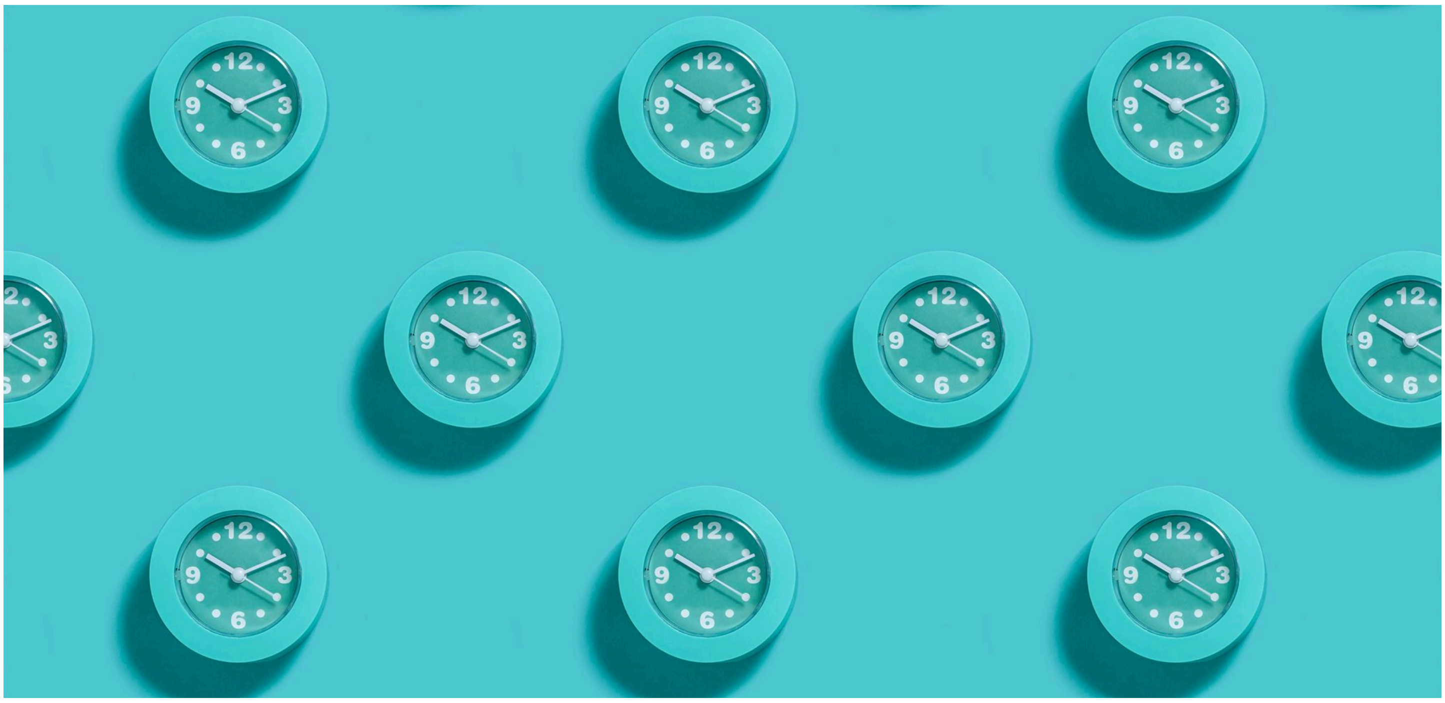

Add icon

Day 2

Topics : INDEX, MATCH, Logical functions (IF, AND, OR, NOT) , Text functions LEFT, RIGHT, MID, CONCATENATE) , Pivot, Slicers and Timelines, VLOOKUP, HLOOKUP

Table of contents

VLOOKUP (Vertical Lookup)
Example 1: Basic VLOOKUP (Exact Match)
Example 2: Approximate Match
Common Errors:
    Example Scenario:
    Practical Use:
HLOOKUP (Horizontal Lookup)
Arguments:
Example 1: Basic HLOOKUP (Exact Match)
Example 2: Approximate Match
Common Errors:
    Example Scenario:
INDEX
MATCH
Advantages of INDEX and MATCH over VLOOKUP/HLOOKUP
Logical Functions (IF, AND, OR, NOT)
Text Functions (LEFT, RIGHT, MID, CONCATENATE)
Pivot Tables
    Steps to Create a Pivot Table
    Example: Basic Pivot Table
    Advanced Features
    When to Use Pivot Tables
    Limitations of Pivot Tables
    Slicers
    Timelines
    Data Summarization Techniques
    Goal Seek
    Data Tables
    Scenario Manager

VLOOKUP (Vertical Lookup)

- **What it is:** Searches for a value in the first column of a range and returns a value in the same row from another column.
- **Importance:** Useful for retrieving related data quickly, like finding a passenger's "Fare" by their "PassengerID."

📌 VLOOKUP (lookup_value, table_array, col_index_num, [range_lookup])

- `lookup_value` :
  - The value you want to search for.
  - Can be a number, text, or cell reference.
- `table_array` :
  - The range of cells where the search happens.
  - The first column of this range is where `lookup_value` will be searched.
- `col_index_num` :
  - The column number (relative to the `table_array`) from which the corresponding value will be returned.
  - Example: If `col_index_num` is 2, VLOOKUP will return a value from the second column of the `table_array` .
- `[range_lookup]` *(optional)*:
  - **TRUE (1)** (default): Approximate match. The data in the first column of the `table_array` must be sorted in ascending order.
  - **FALSE (0)**: Exact match. The function returns an exact match or an error if no match is found.

Example 1: Basic VLOOKUP (Exact Match)

Imagine you have a table like this:

	Product	Price
1	Apple	30
2	Banana	10
3	Cherry	50

+ New

To find the price of a "Banana":

`=VLOOKUP("Banana", A2:B4, 2, FALSE)`

- `lookup_value` : "Banana"
- `table_array` : A2:B4
- `col_index_num` : 2 (Price column)
- `range_lookup` : FALSE (Exact match)

Result: `10`

Example 2: Approximate Match

If you have the following range for tax brackets:

	Income ($)	Tax Rate
1	0	10%
2	10000	15%
3	20000	20%

+ New

To find the tax rate for an income of $15,000:

`=VLOOKUP(15000, A2:B4, 2, TRUE)`

- **Result**: `15%` (closest lower value without exceeding 15000).

Common Errors:

1. **`#N/A`** :
   - Occurs if no match is found for the `lookup_value`.
   - Check spelling or use `IFERROR` to handle it gracefully.
2. **`#REF!`** :
   - Happens if `col_index_num` exceeds the number of columns in `table_array`.
3. **Incorrect Results with `TRUE`** :
   - If the first column of `table_array` is not sorted in ascending order, approximate matches may fail.

Question:

How can you find the "Fare" of a passenger with a specific "PassengerID" using VLOOKUP functions?

Formula: =VLOOKUP(104, A2:F891, 6, FALSE)

Explanation: Searches for PassengerID `104` in column `A` and retrieves the "Fare" from the 6th column.

Example Scenario:

Imagine you have a dataset with the following columns:

- Column A: Passenger ID
- Column B: Name
- Column C: Age
- Column D: Ticket Number
- Column E: Fare
- Column F: Survival Status

If you want to find the survival status (column F) of the passenger with ID `104`, this formula will search for `104` in column A and return the corresponding value from column F.

Practical Use:

If you have a table like this:

	Passenger ID	Name	Age	Ticket Number	Fare	Survival Status
1	101	John Doe	30	A/5 21171	7.25	0
2	102	Jane Doe	25	PC 17599	71.28	1
3	103	Jim Beam	35	STON/O2. 310	8.05	0
4	104	Jack Daniels	40	113803	53.10	1

+ New

Using the formula `=VLOOKUP(104, A2:F891, 6, FALSE)`, it will return `1`, indicating that Jack Daniels survived.

HLOOKUP (Horizontal Lookup)

- **What it is:** Searches for a value in the first row of a range and returns a value in the same column from another row.
- **Importance:** Used when data is organized horizontally rather than vertically.

NOTE : for approximate case, ensure data is aligned in ascending order.

📌 =HLOOKUP (lookup_value, table_array, row_index_num, [range_lookup])

Arguments:

1. `lookup_value`:
   - The value you want to find in the first row of the table.
   - Can be a number, text, or cell reference.
2. `table_array`:
   - The range of cells where the search happens.
   - The first row of this range is where the `lookup_value` will be searched.
3. `row_index_num`:
   - The row number (relative to the `table_array`) from which the corresponding value will be returned.
   - Example: If `row_index_num` is 2, HLOOKUP will return a value from the second row of the `table_array`.
4. `[range_lookup]` *(optional)*:
   - **TRUE (1)** (default): Approximate match. The data in the first row of the `table_array` must be sorted in ascending order.
   - **FALSE (0)**: Exact match. The function returns an exact match or an error if no match is found.

Example 1: Basic HLOOKUP (Exact Match)

Imagine you have a table like this:

		A	B	C
1	Month	Jan	Feb	Mar
2	Sales	5000	7000	6000

+ New

To find the sales for "Feb":

```
=HLOOKUP("Feb", A1:C2, 2, FALSE)
```

- `lookup_value`: "Feb"
- `table_array`: A1:C2
- `row_index_num`: 2 (Sales row)
- `range_lookup`: FALSE (Exact match)

Result: `7000`

Example 2: Approximate Match

If you have the following grading scale:

		A	B	C	D
1	Score	90	80	70	60
2	Grade	A	B	C	D

+ New

To find the grade for a score of 75: `=HLOOKUP(75, A1:D2, 2, TRUE)`

- `lookup_value`: 75
- `table_array`: A1:D2
- `row_index_num`: 2 (Grade row)
- `range_lookup`: TRUE (Approximate match)

Result: `C` (closest lower value without exceeding 75 is 70).

Common Errors:

1. `#N/A`:

- Occurs if no match is found for the `lookup_value`.
- Use `IFERROR` to handle it.

2. **`#REF!`**:
   - Happens if `row_index_num` exceeds the number of rows in `table_array`.
3. **Incorrect Results with `TRUE`**:
   - Ensure the first row of the `table_array` is sorted in ascending order when using `TRUE`.

Question:

How can you find the "Fare" of a passenger with a specific "PassengerID" using HLOOKUP functions?

Formula: `=HLOOKUP("Fare", A1:Z5, 5, FALSE)`

Explanation: Looks for "Fare" in row 1 and retrieves the value from the 5th row of the range.

Example Scenario:

Assume you have the following data in your range `A1:Z5`:

	A1	B1	C1	D1	...	Z1
1						

+ New

	Name	Age	Fare	Ticket	...	
1	John	30	7.25	A/5 21171	...	
2	Jane	25	71.28	PC 17599	...	
3	Jim	35	8.05	STON/O2. 310	...	

+ New

	Jack	40	53.10	113803	...	
1	...	...	...	...	...	...

+ New

In this case, the formula `=HLOOKUP("Fare", A1:Z5, 5, FALSE)` will return `53.10`, which is the value in the 5th row under the "Fare" column.

INDEX

- **What it is:** Returns the value of a cell based on its row and column number in a specified range.
- **Importance:** Provides precise cell reference retrieval, especially when combined with `MATCH` for flexible lookups.

📌 =INDEX (array, row_num, [column_num])

Arguments:

1. **`array`**:
   - The range of cells to look in.
2. **`row_num`**:
   - The row number in the `array` to retrieve data from.

3. `column_num` *(optional)*:
    - The column number in the `array` to retrieve data from.
    - If omitted, the function assumes it's a single column.

Example 1: Basic INDEX

If the table looks like this:

	A	B	C
1	Apple	Banana	Cherry
2	3	1	5

+ New

To find the value in the second row and third column (Cherry's price): `=INDEX(A1:C2, 2, 3)`

Result: `5`

MATCH

- **What it is** Returns the position of a value within a row or column.
- **Importance:** Often paired with `INDEX` for dynamic lookups, offering more flexibility than `VLOOKUP` or `HLOOKUP`.

📌 =MATCH(lookup_value, lookup_array, [match_type])

Arguments:

1. `lookup_value`:
    - The value you want to find.
2. `lookup_array`:
    - The range of cells to search in (can be vertical or horizontal).
3. `match_type` *(optional)*:
    - **1**: Finds the largest value less than or equal to `lookup_value` (data must be sorted in ascending order).
    - **0**: Finds an exact match.
    - **-1**: Finds the smallest value greater than or equal to `lookup_value` (data must be sorted in descending order).

Example 2: Basic MATCH

If the table looks like this:

	A	B	C
1	Apple	Banana	Cherry

+ New

To find the position of "Banana": `=MATCH("Banana", A1:C1, 0)`

Result: `2`

INDEX + MATCH:

INDEX and **MATCH** are two Excel functions that, when combined, offer a more flexible and powerful alternative to **VLOOKUP** and **HLOOKUP**. They can look up values in any direction (not just left-to-right or top-to-bottom) and handle dynamic data ranges effectively.

Example: Find Cherry's Price

	A	B	C
1	Item	Apple	Banana
2	Price	3	1

+ New

To find Cherry's price: `=INDEX(A2:C2, MATCH("Cherry", A1:C1, 0))`

1. **`MATCH("Cherry", A1:C1, 0)`** returns `3` (Cherry's position in the first row).
2. **`INDEX(A2:C2, 3)`** retrieves the value from the third column in the second row.

Result: `5`

Advantages of INDEX and MATCH over VLOOKUP/HLOOKUP

1. **No Left-to-Right Restriction**:
   - **VLOOKUP** can only search in the first column and return data from the right. **INDEX-MATCH** can search in any column and return data from any direction.
2. **Dynamic Column/Row References**:
   - You can use dynamic references for row/column numbers in **INDEX**.
3. **No Need for Sorted Data**:
   - With **MATCH** using `0` for an exact match, data does not need to be sorted.
4. **Handles Inserted/Deleted Columns**:
   - **VLOOKUP** relies on fixed column indices, which can break if columns are added or removed. **INDEX-MATCH** is more robust.

Logical Functions (IF, AND, OR, NOT)

- **What they are:**
  - `IF` : Executes a conditional operation and returns one value if true, another if false.
  - `AND` : Checks if all conditions are true.
  - `OR` : Checks if any condition is true.
  - `NOT` : Reverses a logical value.
- **Importance:** Essential for decision-making in formulas, enabling dynamic and conditional outputs.

Python

```python
N = 3
IF N > 0:
  POSITIVE
ELSE:
  NEGATIVE


```

N = 3

IF N > 0: POSITIVE

ELSE : NEGATIVE

Question:

How can you classify passengers into groups based on survival and class?

Solution:

- **IF Function:**

Formula: `=IF(B2=1, "Survived", " Did Not Survive ")`

Explanation: If the passenger survived (`B2=1`), display "Survived"; otherwise, "Did Not Survive".

AND Function:

Formula: `=IF( AND(B2=1, C2=1 ), " First Class Survivor ", "Other")`

Explanation: Checks if the passenger survived and belongs to the first class.

OR Function:

Formula: `=IF(OR(C2=1, D2="Female"), "Priority Passenger", "Regular Passenger")`

Explanation: Classifies as "Priority Passenger" if in first class or female.

NOT Function:

Formula: `=IF(NOT(D2="Male"), "Not Male", "Male")`

Text Functions (LEFT, RIGHT, MID, CONCATENATE)

- **What they are:**
  - `LEFT` : Extracts characters from the start of a string.
  - `RIGHT` : Extracts characters from the end of a string.
  - `MID` : Extracts characters from the middle of a string.
  - `CONCATENATE` (or `TEXTJOIN` / `CONCAT`): Combines multiple text strings into one.
- **Importance:** Useful for manipulating and cleaning text data, such as splitting names or creating combined identifiers.

Question:

How can you extract the first name of passengers and combine it with their survival status?

Solution:

- **LEFT Function:**

Formula: `=LEFT(A2, 5)`

Explanation: Extracts the first 5 characters of column `A` .

RIGHT Function:

Formula: `=RIGHT(Name, LEN(Name)-FIND(",", Name)-1)`

Explanation: Extracts the first name by removing the surname (assuming names are in "Last, First" format).

MID Function:

Formula: `=MID(Name, FIND(",", Name)+2, LEN(Name))`

Explanation: Similar to the RIGHT function, but explicitly extracts the name after the comma.

CONCATENATE:

Formula: `=CONCATENATE(MID(Name, FIND(",", Name)+2, LEN(Name)), " - ", IF(Survived=1, "Survived", "Did Not Survive"))`

Explanation: Combines the first name with survival status.

Pivot Tables

Pivot tables are one of the most powerful tools in Excel, allowing you to analyze, summarize, and present large datasets quickly. They can transform raw data into meaningful insights with just a few clicks.

- **What they are:** A tool for summarizing, analyzing, and presenting data by grouping and aggregating information.
- **Importance:** Allows quick exploration of trends, patterns, and comparisons in large datasets without manual calculations.

Steps to Create a Pivot Table

1. Prepare Your Data

Ensure your data is organized in a tabular format:

- Each column should have a **header** (e.g., Date, Product, Sales).
- No blank rows or columns in the data.

- Avoid merged cells.

2. Select Your Data

- Highlight the dataset (or place your cursor inside the data table).
- Go to the **Insert** tab on the Ribbon and select **PivotTable**.

3. Insert the Pivot Table

- In the dialog box:
    - **Select Data Range**: The highlighted range appears by default.
    - **Choose Destination**: Decide whether to place the Pivot Table in a new worksheet or the current one.
    - Click **OK**.

4. Build Your Pivot Table

The **PivotTable Fields Pane** will appear, containing:

- **Field List**: Columns from your dataset.
- **Drag-and-Drop Sections**:
    - **Rows**: Categories for rows (e.g., Product names).
    - **Columns**: Categories for columns (e.g., Months).
    - **Values**: Data to summarize (e.g., Sales totals).
    - **Filters**: Filters to refine the view (e.g., Region).

5. Customize Your Table

- Drag fields to the appropriate areas in the PivotTable Fields Pane.
- Change summary calculations (e.g., from Sum to Count) by clicking the dropdown next to a field in the Values area and selecting **Value Field Settings**.
- Apply filters or sort the data for better clarity.

Example: Basic Pivot Table

Dataset:

	Date	Product	Region	Sales
1	2024-01-01	Apple	East	500
2	2024-01-01	Banana	West	300
3	2024-01-02	Apple	East	600
4	2024-01-02	Banana	West	400

+ New

Pivot Table:

To summarize **Sales by Product and Region**:

1. Drag **Product** to Rows.
2. Drag **Region** to Columns.
3. Drag **Sales** to Values.

Result:

	Product	East	West	Grand Total
1	Apple	1100		1100
2	Banana		700	700

3	Grand Total	1100	700	1800

+ New

Advanced Features

1. Sorting and Filtering

- Sort data (e.g., by highest sales) by clicking on column headers in the Pivot Table.
- Use filters or slicers for interactive data segmentation.

2. Grouping Data

- Group numeric values (e.g., by ranges like 0-500, 501-1000).
- Group dates (e.g., by months, quarters, years).

3. Adding Calculated Fields

- Go to the PivotTable Analyze tab → **Fields, Items & Sets** → **Calculated Field**.
- Create custom calculations (e.g., profit = sales - cost).

4. Refreshing Data

- When the source data changes, refresh the Pivot Table by right-clicking it and selecting **Refresh**.

5. Conditional Formatting

- Apply color scales or data bars to make trends more visible.

When to Use Pivot Tables

- Analyzing sales data by region or product.
- Comparing year-over-year performance.
- Summarizing customer or employee data.
- Creating interactive dashboards.

Limitations of Pivot Tables

- Cannot modify individual cells directly in the Pivot Table.
- Requires properly formatted source data.
- May slow down with extremely large datasets.

Slicers

- **What they are:** Visual filters for pivot tables or pivot charts, providing an interactive way to filter data.
- **Importance:** Enhances usability and enables dynamic data exploration, particularly in dashboards.

Timelines

- **What they are:** Specialized slicers designed for filtering date-based data.
- **Importance:** Makes filtering date ranges intuitive and efficient in pivot tables

---

Data Summarization Techniques

- **What they are:** Methods like sorting, filtering, grouping, and using aggregation functions (SUM, AVERAGE, COUNT).
- **Importance:** Helps distill large datasets into meaningful insights.

Goal Seek

- **What it is:** A tool for finding the input value required to achieve a specific result in a formula.
- **Importance:** Useful for reverse calculations, like determining the fare required to reach a specific revenue.

Data Tables

- **What they are:** Tools for performing sensitivity analysis by creating tables that display multiple scenarios of data based on variable inputs.
- **Importance:** Useful for exploring outcomes and performing what-if analysis.

Scenario Manager

- **What it is:** A tool for saving and comparing different sets of input values to see how they affect results.
- **Importance:** Enables strategic planning and decision-making by comparing scenarios side-by-side.