How to Find First Occurrence Based on Multiple Criteria in Excel?

Authored by

stats writer

November 18, 2025

Mastering Multi-Criteria Lookups in Excel

The ability to perform complex lookups is essential for advanced data analysis in Excel. While basic lookups, often handled by functions like VLOOKUP or XLOOKUP, are straightforward, locating a specific item based on two or more simultaneous conditions requires a more sophisticated technique. When multiple rows meet the criteria, the challenge is often to find the very first instance—a task that standard lookup functions cannot handle natively without special manipulation. This comprehensive guide introduces a robust and universally compatible method using a combination of the INDEX function and the MATCH function, integrated with powerful Boolean logic.

This powerful technique allows analysts and data professionals to precisely pinpoint specific data records based on conditions applied across different columns. The core philosophy of this solution revolves around creating a virtual lookup column where all specified criteria are evaluated simultaneously for every row. By enforcing these multiple conditions through mathematical operations, we generate an array of TRUE and FALSE values, which are subsequently converted into numerical 1s and 0s. The location of the first '1' in this generated array corresponds exactly to the row number where all stipulated criteria are satisfied, ensuring we retrieve the correct first occurrence.

To effectively find the first occurrence of a value in a specified return column in your spreadsheet based on multiple criteria, you can utilize the following complex but highly effective formula structure. We will dedicate significant detail to breaking down each component to ensure a complete and thorough understanding of how this array mechanism operates within the calculation engine of Excel. This method stands out as an exemplary demonstration of advanced spreadsheet proficiency.

Deconstructing the Core Formula Structure

The specific formula we employ harnesses the combined power of nested functions to simulate an advanced, conditional lookup behavior that standard functions cannot replicate alone. This structured approach is invaluable for users who frequently manage large, detailed datasets where accuracy and the correct identification of the first qualifying record are absolutely paramount for reporting integrity.

The final formula structure is presented below. It is critical to identify and understand the roles of the distinct components: the final return range (C2:C13), the criteria ranges used for evaluation (A2:A13 and B2:B13), and the cells containing the specific lookup values (F1 and F2).

=INDEX(C2:C13,MATCH(1,INDEX((A2:A13=F1)*(B2:B13=F2),),FALSE))

Fundamentally, this formula is designed to return the value located at the first matching position within the result range, **C2:C13**. This retrieval is strictly conditional: the corresponding entry in range **A2:A13** must be equal to the value found in cell **F1**, and simultaneously, the corresponding entry in range **B2:B13** must be equal to the value found in cell **F2**. The use of the multiplication operator (`*`) between the two conditional checks is the single most critical element, as it enables the multi-criteria matching by applying Boolean logic.

Implementing Boolean Logic for Condition Evaluation

The success of this method hinges on understanding how Boolean logic is mathematically processed within Excel. When Excel evaluates an array comparison operation (such as A2:A13=F1), it produces a corresponding array consisting entirely of TRUE and FALSE values, indicating whether each cell in the range meets the condition. For any subsequent mathematical calculation, Excel automatically coerces TRUE to the numerical value 1 and FALSE to the numerical value 0.

The centerpiece of the multi-criteria check is the expression: **((A2:A13=F1)*(B2:B13=F2))**. This expression performs two separate Boolean array comparisons and then multiplies the resulting arrays element-by-element across the rows. Because multiplication is used, the only way a row can result in a value of 1 (representing a successful match) is if both individual conditions return TRUE (1 * 1 = 1). If either condition is FALSE (0), or if both are FALSE, the resulting product will be 0. This multiplication effectively creates a single array of 1s and 0s, where 1 precisely identifies a row that satisfies every specified criterion.

The nested INDEX function wrapper used here, **INDEX(...,)**, serves a historical and technical role. While modern Excel versions often automatically handle the implicit conversion of the Boolean array, using this construction guarantees that the array of 1s and 0s is properly formed and passed as a single column array argument to the subsequent MATCH function, avoiding potential errors related to required Ctrl+Shift+Enter (CSE) array formula entry in older versions.

Pinpointing the Row Position using MATCH

Once the numerical array of 1s and 0s is successfully generated by the Boolean multiplication, the MATCH function takes center stage. Its crucial purpose is to find the position (or relative row index) of the very first '1' within that calculated array, as this '1' marks the first occurrence of the data that satisfies both criteria. The syntax employed is **MATCH(1, , FALSE)**.

The lookup value is explicitly set to **1**: This is because we are searching for the exact position where the combination of Boolean conditions resulted in a TRUE outcome (1).

The lookup array is the stream of 1s and 0s generated by the inner array operation.

The match type is set to **FALSE** (or 0): This mandates an exact match. By searching for the exact value 1, the MATCH function correctly identifies the row number relative to the start of the defined range where the first successful match occurs.

The output of the MATCH function is a single integer. This integer represents the position of the first successful match relative to the start of the defined data ranges (e.g., if the match is found on the fifth row of the data, MATCH returns 5). This numerical index is the final piece of data required by the outermost INDEX function to complete the retrieval.

Finalizing Data Retrieval with the INDEX Function

The outermost function in our structure is the INDEX function. The INDEX function serves as the final delivery mechanism, requiring two primary inputs: the array from which the desired value should be retrieved (the result range) and the exact row number within that array.

Array: This is defined as C2:C13. This column holds the values we ultimately wish to return (in our example, the points scored).

Row Number: This is the crucial integer output provided directly by the nested MATCH function.

By seamlessly integrating these elements, the INDEX function efficiently retrieves the value from the designated row in the result column. This retrieved value corresponds exactly to the first row identified by the MATCH function as meeting all the defined criteria. This highly robust and reliable structure is why the INDEX/MATCH combination remains one of the most powerful foundational skills for any advanced spreadsheet user seeking precise conditional lookups.

Practical Example: Applying Criteria to a Dataset

To solidify the theoretical explanation, let's observe this formula in action using a practical dataset of basketball player statistics. This example clearly demonstrates how the formula isolates and locates the first record that satisfies two distinct conditions simultaneously: the player's team affiliation and their playing position.

Consider the following dataset, which includes Player Name, Team, Position, and Points Scored over a specific season. The data spans from row 2 to row 13, covering columns A through C:

	A	B	C	D	E	F
1	**Team**	**Position**	**Points**			
2	Mavs	Guard	22			
3	Mavs	Forward	14			
4	Mavs	Forward	17			
5	Mavs	Guard	19			
6	Spurs	Guard	30			
7	Spurs	Forward	31			
8	Spurs	Forward	37			
9	Spurs	Guard	20			
10	Rockets	Forward	28			
11	Rockets	Guard	12			
12	Rockets	Guard	16			
13	Rockets	Guard	19			
14						
15						
16						
17						
18						
19						

Our immediate objective is to perform a targeted query: we need to return the **Points** value for the very first player who meets the dual criteria of being affiliated with the **Spurs** team and plays the **Forward** position. It is essential that we identify the first instance, as the dataset may contain multiple players who match this criteria, but our requirement is strictly the initial entry found when reading the data from top to bottom.

This specific requirement underscores why using simple filters or basic lookups like VLOOKUP is insufficient. We require a function that systematically traverses the data, identifies the specific row where both Column A (Team) equals "Spurs" AND Column B (Position) equals "Forward," and then returns the corresponding numerical value from Column C (Points).

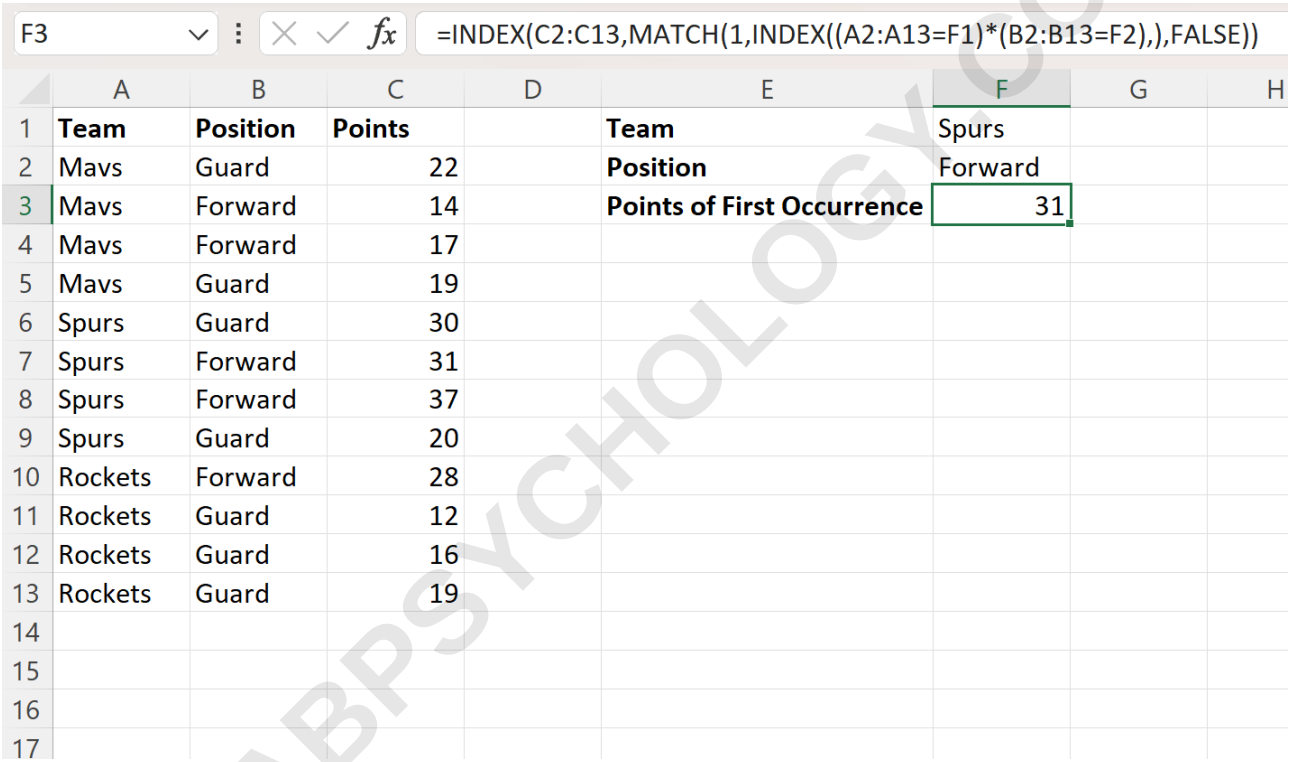
Executing the Formula and Interpreting the Result

For maximum flexibility, we define our search parameters in separate input cells. We will use cell **F1** to hold the primary criteria (Team: "Spurs") and cell **F2** to hold the secondary criteria (Position: "Forward"). This allows the formula to be dynamic, enabling instantaneous calculation updates upon criteria changes without requiring any formula editing.

We then enter the complete array formula into cell **F3**. The formula instructs Excel to search for matches within the ranges **A2:A13** (Team) and **B2:B13** (Position) and return the corresponding value from **C2:C13** (Points).

=INDEX(C2:C13,MATCH(1,INDEX((A2:A13=F1)*(B2:B13=F2),),FALSE))

Upon processing, the internal Boolean evaluation generates the array, and the MATCH function pinpoints the exact location of the first '1'. This position is then fed to the outer INDEX function. The following visual result clearly shows the successful execution:



	A	B	C	D	E	F	G	H
1	**Team**	**Position**	**Points**		**Team**	Spurs		
2	Mavs	Guard	22		**Position**	Forward		
3	Mavs	Forward	14		**Points of First Occurrence**	31		
4	Mavs	Forward	17					
5	Mavs	Guard	19					
6	Spurs	Guard	30					
7	Spurs	Forward	31					
8	Spurs	Forward	37					
9	Spurs	Guard	20					
10	Rockets	Forward	28					
11	Rockets	Guard	12					
12	Rockets	Guard	16					
13	Rockets	Guard	19					
14								
15								
16								
17								

The formula successfully returns a points value of **31**. A quick manual inspection of the source data reveals that the player on row 4 (which is the third row relative to the start of the data range A2:A13) is the first entry where the Team column specifies "Spurs" and the Position column specifies "Forward." This precise retrieval validates the accuracy and power of the advanced lookup structure.

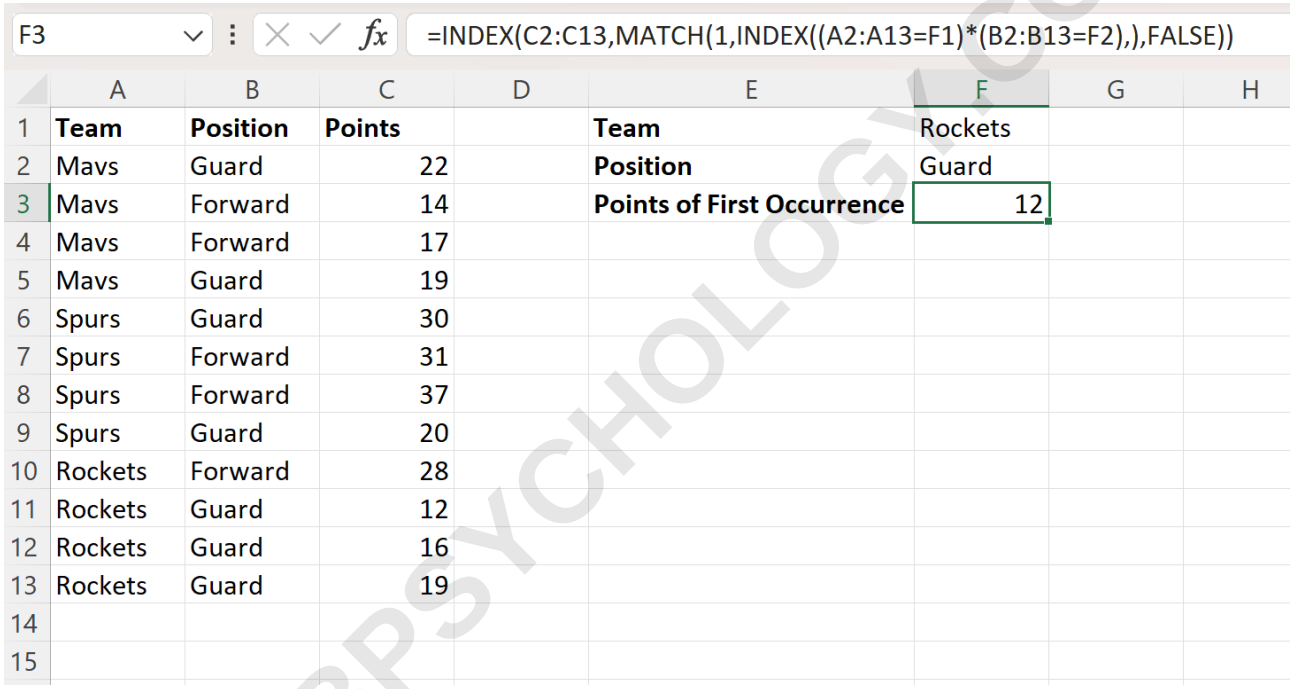
Demonstrating Dynamic Functionality with New Criteria

A significant benefit of linking criteria to external input cells (F1 and F2) is the inherent dynamic recalculation capability. Unlike fixed or hardcoded formulas, changing the text in the criteria cells instantly forces the formula to re-evaluate the entire dataset, finding the first match for the new

conditions without requiring the user to edit the complex formula itself. This makes the method exceptionally useful for interactive reporting and analytical tools.

Let us test this dynamic nature by changing the search parameters to look for a different team and position combination. Suppose we modify the criteria in cell **F1** to **Rockets** and the criteria in cell **F2** to **Guard**.

The core formula in cell F3 remains unchanged, yet it automatically processes the new input values. The Boolean logic array is recalculated internally, identifying the new position of the first '1' corresponding to the "Rockets/Guard" combination within the dataset.



F3 =INDEX(C2:C13,MATCH(1,INDEX((A2:A13=F1)*(B2:B13=F2),),FALSE))

	A	B	C	D	E	F	G	H
1	**Team**	**Position**	**Points**		**Team**	Rockets		
2	Mavs	Guard	22		**Position**	Guard		
3	Mavs	Forward	14		**Points of First Occurrence**	12		
4	Mavs	Forward	17					
5	Mavs	Guard	19					
6	Spurs	Guard	30					
7	Spurs	Forward	31					
8	Spurs	Forward	37					
9	Spurs	Guard	20					
10	Rockets	Forward	28					
11	Rockets	Guard	12					
12	Rockets	Guard	16					
13	Rockets	Guard	19					
14								
15								

As evidenced by the updated screenshot, the formula correctly returns a value of **12**. This result is verified by checking the dataset: 12 points is the value that corresponds to the first player to be affiliated with the **Rockets** team and hold the **Guard** position. This clear demonstration confirms the adaptability and accuracy of this multi-criteria array formula structure, making it a reliable tool for analysts facing diverse and evolving lookup challenges.