Excel: Find First Occurrence Based on Multiple Criteria

Authored by

Mohammed looti

November 10, 2025

Navigating complex datasets in Excel requires sophisticated analytical tools that go far beyond simple VLOOKUP or XLOOKUP functions. While standard lookup functions are highly efficient for single-criteria searches, accurately identifying the first data entry that satisfies multiple criteria simultaneously necessitates the deployment of a specialized array formula. This robust technique cleverly integrates the power of the INDEX function with the positional finding capabilities of the MATCH function, creating a dependable solution for complex conditional data extraction.

The standard structure for generating a formula that returns the first occurrence based on multiple conditions is presented below. A key advantage of this specific construction is its use of implicit array handling. This means that while the formula executes powerful array calculations, it generally does not require the traditional Ctrl+Shift+Enter confirmation sequence in modern versions of Excel, simplifying its deployment considerably.

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

At its core, this formula is engineered to scan defined data ranges and return the corresponding value from the designated **Result Range** (`C2:C13`). This retrieval is conditional and highly specific: it only occurs if the element in **Criteria Range 1** (`A2:A13`) matches the value specified in **Criteria Cell 1** (`F1`) **AND** the element in **Criteria Range 2** (`B2:B13`) matches the value in **Criteria Cell 2** (`F2`). This sophisticated arrangement is indispensable for executing precise, targeted lookups within extensive datasets.

The Core Challenge: Translating Multiple Conditions

The fundamental challenge when executing a multi-criteria lookup is transforming several concurrent conditions into a single, usable numerical index that the `MATCH` function can correctly process. Because standard lookup functions are limited to a single input condition, we must employ advanced array formula techniques that utilize Boolean logic operations. When we compare an entire range against a criterion (e.g., `A2:A13=F1`), Excel evaluates this comparison for every cell, dynamically generating an array composed solely of `TRUE` and `FALSE` values.

To enforce the logical "AND" requirement--where all criteria must be satisfied for a result to be valid--we multiply these Boolean arrays together. In the context of arithmetic operations within Excel, the value `TRUE` is implicitly converted to 1, and `FALSE` is converted to 0. Consequently, the multiplication step, represented as `(Criteria 1 Array) * (Criteria 2 Array)`, will yield a result of 1 (1 * 1) only if **both** criteria are met for a specific row. If either criterion is `FALSE` (0 * 1 or 0 * 0), the result is 0. This multiplicative process efficiently condenses the multi-criteria check into a single array of 1s and 0s, where 1 denotes a perfect match across all specified conditions, and 0 signifies a mismatch.

The subsequent task of finding the **first** occurrence is resolved by the structure of the `MATCH` function. Since the resulting array contains only 1s (matches) and 0s (non-matches), searching for the value 1 with an exact match type (`FALSE`) will pinpoint the exact relative position of the very first row where all criteria were simultaneously satisfied. This derived position number is then fed to the outer `INDEX` function, which retrieves the corresponding data point from the designated return column. This ingenious combination of nested functions and Boolean logic provides a powerful and stable method for performing complex data extraction.

Deconstructing the INDEX/MATCH Array Formula

Achieving mastery of this technique requires a clear understanding of how each component contributes to the overall function within the nested structure. This specific formula construction is often favored in modern data analysis because the strategic inclusion of the inner `INDEX` function enables the entire array calculation to execute natively, thereby eliminating the necessity for the user to confirm it as a traditional array formula using the Ctrl+Shift+Enter key combination.

The Criteria Array Core: `(A2:A13=F1)*(B2:B13=F2)`. This segment operates as the calculation engine. It conducts a logical test across every row within the defined criteria ranges. The crucial multiplication operator ensures that the result is a 1 only when **both** comparisons return `TRUE`, effectively generating the critical array of 1s and 0s that maps the match status row by row.

The Array Wrapper (Inner INDEX): `INDEX(...,)`. The inner INDEX function serves as a wrapper for the criteria array result. By intentionally leaving the row number argument blank (indicated by the trailing comma), it forces Excel to pass the entire result of the array calculation (the 1s and 0s) to the subsequent function. This mechanism is what successfully bypasses the need for explicit CSE entry, adhering to contemporary efficiency standards.

The Position Finder: `MATCH(1, ..., FALSE)`. The MATCH function then searches the array of 1s and 0s for the first instance of the number 1. Because the match type is set to `FALSE` (exact match), it guarantees the return of the position of the first 1 found--this position directly corresponds to the first row where all conditions were met. This derived row index is essential for the final lookup step.

The Value Retriever (Outer INDEX): `INDEX(C2:C13, ...)`. Finally, the outer INDEX function receives the relative row index from the `MATCH` function. It looks into the specified **Return Range** (`C2:C13`) and accurately extracts the value housed at that specific relative row position. This result constitutes the final output of the multi-criteria search.

Practical Implementation: A Step-by-Step Example

To demonstrate the practical utility of this powerful formula, let us consider a typical dataset

containing statistics for basketball players, including their associated Team, their specific Position, and their Points Scored. Our objective is to create a dynamic lookup that can identify the points scored by the very first player who matches a set of specific team and position criteria.

Assume we are working with the following sample dataset structure:

	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						

For this example, we aim to retrieve the points value for the first player who belongs to the **Spurs** team **AND** plays the position of **Forward**. Satisfying these two distinct criteria simultaneously requires the use of our specialized array formula. To ensure flexibility, we establish the search criteria in dedicated, easily modifiable cells.

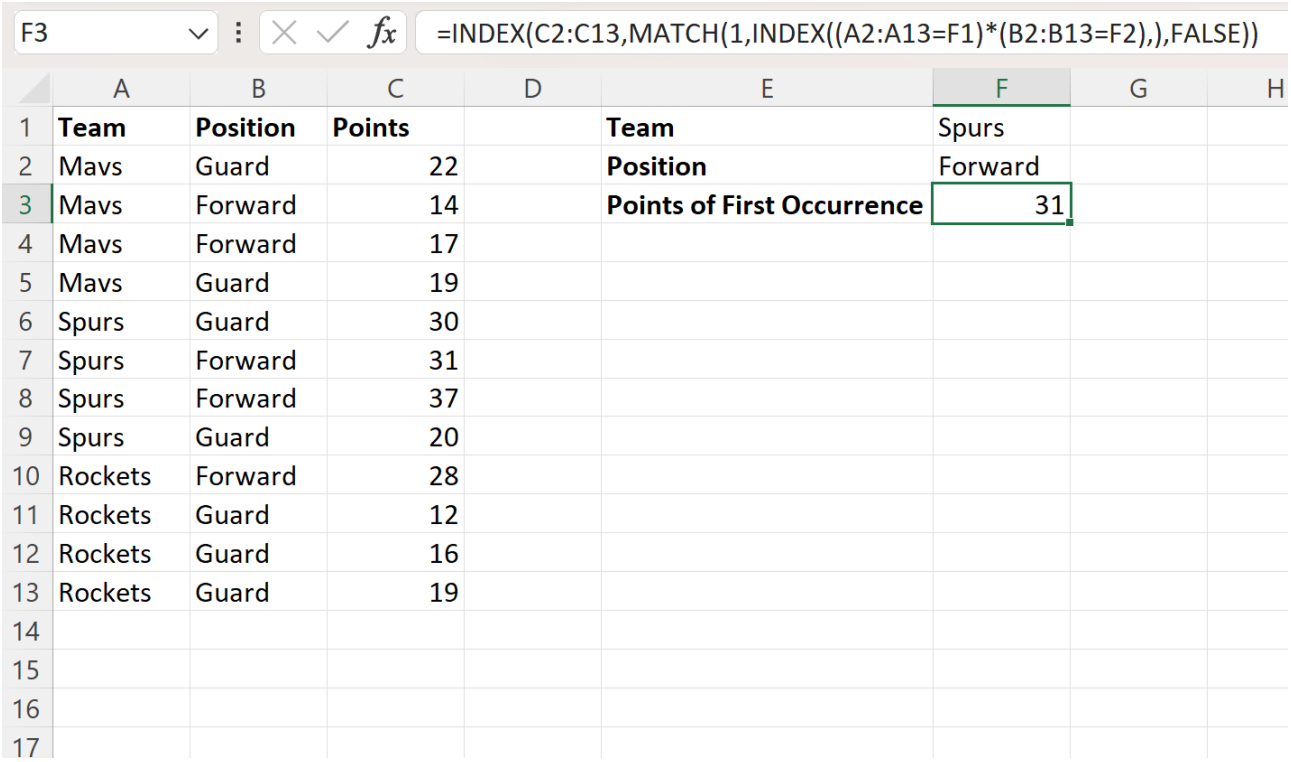
We will place the Team criterion ("Spurs") in cell **F1** and the Position criterion ("Forward") in cell **F2**. The lookup formula itself will be entered into cell **F3**. The formula is scoped to search across rows 2 through 13, where Column A contains the Team, Column B contains the Position, and Column C contains the Points (our desired return value). The specific formula entered into cell **F3** remains:

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

Upon execution, the criteria array calculates the match status for every player in the list. The formula scans the table sequentially until it identifies the first row where the Team equals the value in F1 ("Spurs") and the Position equals the value in F2 ("Forward"). The resulting row number is captured by the `MATCH` function, and the final points value is subsequently retrieved by the outer `INDEX` function.

Dynamic Analysis and Criteria Modification

When the formula is applied using the initial criteria--**Team: Spurs** and **Position: Forward**--it successfully generates a specific numerical result. The following visual confirms the immediate execution within the spreadsheet environment:



	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 correctly returns a points value of **31**. This outcome is achieved because, scanning the dataset from top to bottom, the first player entry that satisfies both criteria (Spurs and Forward) is found, and their corresponding points tally is 31. This result validates that the combined use of `INDEX`, `MATCH`, and Boolean logic accurately isolates the first match based on the specified complex conditions.

One of the most valuable aspects of placing the criteria in external cells (F1 and F2) is the formula's inherent flexibility and dynamism. Should the lookup requirements evolve, the user does not need to manually edit the formula structure; simply updating the values in the criteria cells will automatically trigger a recalculation and return a new result based on the revised conditions.

For example, if we modify the requirements to search for the first player on the **Rockets** team who plays the position of **Guard**, we merely update cell **F1** to "Rockets" and cell **F2** to "Guard."

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								

With the updated criteria, the Excel formula immediately recalculates the criteria array. It now searches for the first instance of '1' where the Team is Rockets AND the Position is Guard. The formula accurately returns a new value of **12**, corresponding to the points scored by the first player entry that satisfies this new set of conditions. This powerfully demonstrates the robustness of the `INDEX/MATCH` multi-criteria method as an indispensable tool for dynamic data analysis and complex lookup operations within large datasets.

Summary and Advanced Applications

The strategic use of the `INDEX` and `MATCH` functions, combined with Boolean logic through array multiplication, offers a versatile and highly efficient solution for performing multi-criteria lookups and reliably identifying the first matching occurrence. This technique successfully overcomes the limitations inherent in single-criteria functions and maintains exceptional performance even within extensive workbooks, solidifying its position as a cornerstone of advanced data mastery. By grasping the principles of the underlying array mechanics, users can easily adapt this formula structure to accommodate three, four, or even more criteria by simply appending additional parenthetical comparisons multiplied into the core logical array.

For professionals seeking to further expand their capabilities in complex data manipulation, exploring advanced topics can enhance efficiency and analytical depth. The following resources

provide avenues for continuous learning regarding common and advanced operations in spreadsheet management:

Techniques for managing and utilizing relative versus absolute cell references.

Methods for performing inexact (approximate) data lookups efficiently.

Strategies for handling common formula errors (such as #N/A) gracefully using functions like `IFERROR`.