How to Write an IF Function with 4 Conditions

Authored by

stats writer

November 20, 2025

The challenge of handling multiple criteria in data analysis frequently requires the use of complex conditional formulae. Implementing the IF Function with four distinct conditions is a common requirement in data processing tools like Excel. A fundamental approach involves nesting the functions, creating a structure that evaluates conditions sequentially. This nested approach can be represented conceptually as: IF(condition1, value1, IF(condition2, value2, IF(condition3, value3, IF(condition4, value4, value_if_false)))). This structure is essential because it allows the formula to test each condition in the specified order, immediately returning the corresponding value upon the first successful evaluation, otherwise proceeding to the next test until all possibilities are exhausted or the final default value is returned.

Mastering the IF Function with Multiple Conditions

Working with vast datasets often necessitates classifying information based on complex, layered rules. When dealing with four or more specific criteria, the traditional simple IF Function falls short. Therefore, data analysts must rely on advanced techniques that combine multiple logical checks. Excel offers several powerful methods to handle these scenarios, primarily involving nesting the IF structure itself or integrating powerful logical operators like AND and OR directly within the IF statement.

The choice of method--whether sequential nesting or combined logical testing--depends entirely on the desired outcome and the relationship between the conditions. If the conditions are mutually exclusive (e.g., scoring tiers), nesting is appropriate. If all conditions must be simultaneously true, the AND function is required. Conversely, if meeting any one of the conditions is sufficient to trigger the result, the OR function provides the necessary flexibility. Understanding the nuances of these approaches is critical for achieving accurate and efficient data classification.

Below, we delve into the three primary methodologies employed within Excel to construct an IF Function capable of evaluating four separate conditions. We will provide detailed explanations and practical examples using a common dataset to illustrate the practical application of each formula structure.

Method 1: The Architecture of Nested IF Statements

The Nested IF Statements approach is the traditional method for handling multiple outcomes based on sequential criteria. In this structure, the "value if false" argument of one IF function becomes the entire subsequent IF function. This creates a chain reaction: if the first condition is not met, the formula proceeds to evaluate the second IF; if the second is not met, it moves to the third, and so on. Since we are addressing four conditions, this requires nesting three IF statements inside the outermost IF function, plus a final default value if none of the four criteria are satisfied.

While effective, nesting four or more IF functions requires careful attention to parentheses placement and logical ordering. The order of evaluation is paramount; since the formula stops at the first true condition, conditions must be arranged from the most specific to the least specific, or in a clear numerical sequence (e.g., ascending or descending order). Misordering the conditions can lead to logical errors where an evaluation that should result in a higher tier is mistakenly caught by a lower tier simply because it evaluates first.

This method is ideally suited for creating rating systems, grading scales, or tier-based categorization where only one outcome is possible for any given input value. It transforms raw data points into actionable categorical labels, providing immediate qualitative assessment of numerical metrics.

Here is the fundamental syntax for implementing this technique:

=IF(C2<15,"Bad",IF(C2<20,"OK",IF(C2<25,"Good",IF(C2<30,"Great","Awesome"))))

Practical Demonstration: Categorization Using Nested IF

To illustrate the power and application of Nested IF Statements, we will utilize a sample dataset tracking player performance metrics. Our goal is to assign a performance status (Bad, OK, Good, Great, or Awesome) based on the numerical value found in the **Points** column (Column C). Notice that with four conditions, we can define five distinct output categories, the fifth being the final default value.

We begin by typing the derived formula directly into cell **E2**. This formula systematically tests the value in cell C2 against thresholds of 15, 20, 25, and 30. Because we are testing in ascending order (less than 15, then less than 20, etc.), the structure ensures that once a value meets a criterion, it immediately exits the function, preventing it from being evaluated by subsequent, broader criteria.

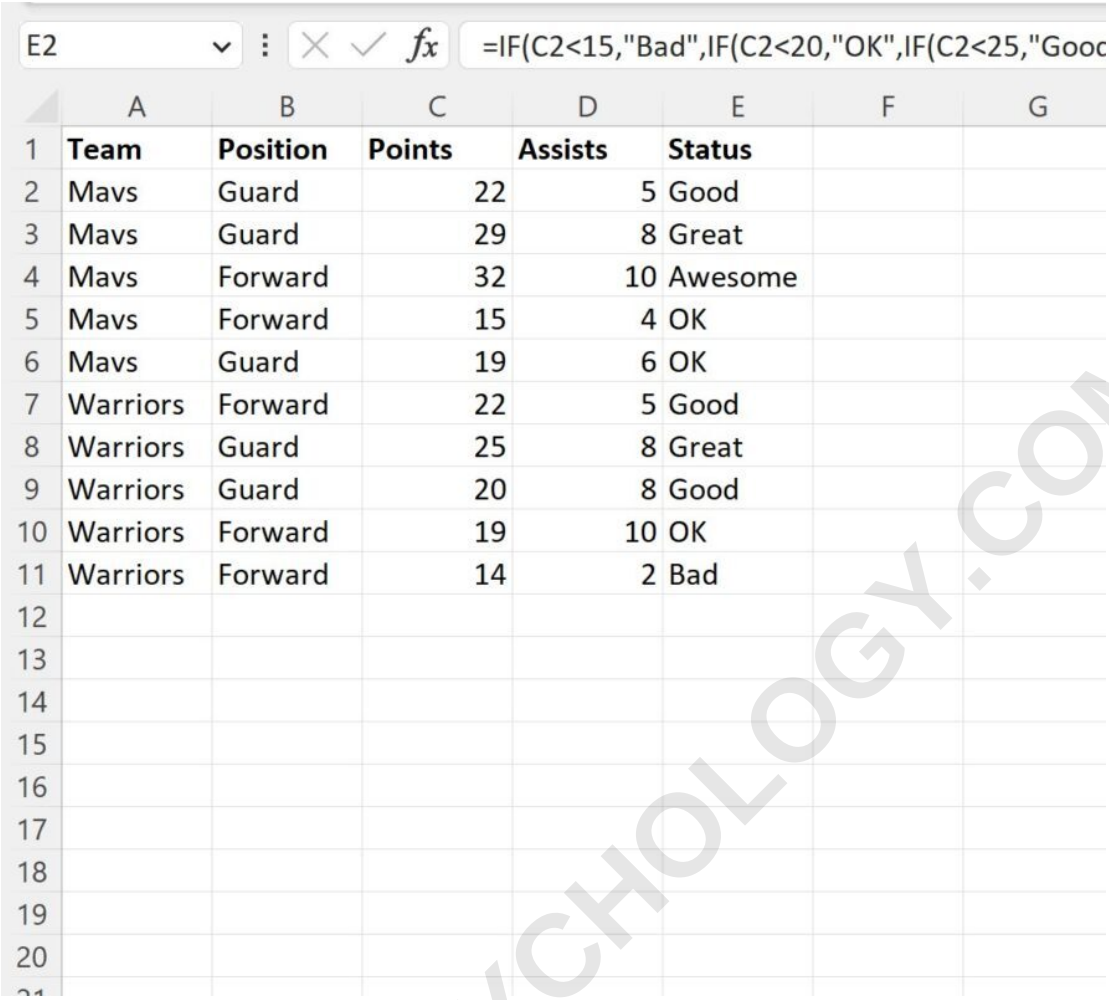
Following the successful input into E2, the efficiency of Excel allows us to quickly apply this logic to the entire dataset. By dragging the fill handle down from E2 to the remaining cells in column E, the relative cell references (C2, C3, C4, etc.) automatically adjust, instantly calculating the performance status for every player based on their respective point totals. This technique highlights how complex conditional logic can be scaled across hundreds or thousands of rows with minimal effort. The following examples show how to use each formula in practice with the following dataset in Excel:

	A	B	C	D	E	F
1	**Team**	**Position**	**Points**	**Assists**		
2	Mavs	Guard	22	5		
3	Mavs	Guard	29	8		
4	Mavs	Forward	32	10		
5	Mavs	Forward	15	4		
6	Mavs	Guard	19	6		
7	Warriors	Forward	22	5		
8	Warriors	Guard	25	8		
9	Warriors	Guard	20	8		
10	Warriors	Forward	19	10		
11	Warriors	Forward	14	2		
12						
13						
14						
15						
16						
17						
18						
19						
20						
21						

We can type the following formula into cell **E2** to return a specific value based on the value for each player in the **Points** column:

=IF(C2<15,"Bad",IF(C2<20,"OK",IF(C2<25,"Good",IF(C2<30,"Great","Awesome"))))

We can then drag and fill this formula down to each remaining cell in column E:



The logic within the formula operates as follows, defining five outcome tiers:

If the value in the Points column is less than 15, return **Bad**.
Else, if the value in the Points column is less than 20, return **OK**.
Else, if the value in the Points column is less than 25, return **Good**.
Else, if the value in the Points column is less than 30, return **Great**.
Else (meaning the value is 30 or greater), return **Awesome**.

Method 2: Integrating IF with the AND Logical Operator

While Nested IF Statements are excellent for sequential classification, there are scenarios where a single outcome depends on the simultaneous truth of multiple, unrelated conditions. This is where the AND function becomes indispensable. The AND function is a form of Boolean Logic that returns a TRUE value only if all of its supplied logical arguments evaluate to TRUE. If even one condition is FALSE, the entire AND statement returns FALSE.

When used inside an IF Function, the AND function replaces the single logical test. This allows us

to collapse four separate conditions into one comprehensive test. If the AND returns TRUE (meaning all four criteria are met), the IF function returns its designated "value if true." If the AND returns FALSE, the IF function returns its "value if false." This structure provides a clean, single-cell formula to enforce strict, multi-faceted requirements.

For instance, if we are evaluating an athlete and require that they must play for a specific team, hold a specific position, exceed a minimum point threshold, and exceed an assist threshold, we need the conjunction provided by the AND operator. This setup contrasts sharply with nesting, where the criteria are evaluated one at a time. The AND structure demands absolute compliance across all specified logical tests.

The structure for combining IF and AND is concise and powerful, clearly detailing the four required conditions:

=IF(AND(A2="Mavs", B2="Guard", C2>20, D2>4), "Yes", "No")

Executing Strict Criteria Checks with IF and AND

Using the same player dataset, we can now assess which players meet a highly selective set of criteria using the combined IF and AND function. We want to identify players who are exceptional performers for a specific subset. Our four non-negotiable conditions are: Team must be "Mavs", Position must be "Guard", Points must be greater than 20, and Assists must be greater than 4. The formula is entered into cell E2 and then dragged down column E.

The resulting column, displayed in the image below, provides a simple binary outcome (Yes/No) that immediately flags records meeting all four stringent requirements. This type of conditional formulae is essential for compliance checks, filtering specific demographics, or pinpointing records that satisfy every element of a complex requirement profile.

It is important to emphasize the principle of Boolean Logic here: the output is "Yes" only if the AND statement returns a TRUE value, meaning every single check within the parentheses yielded TRUE. If any player fails to meet even one of the four requirements--for instance, scoring 25 points but only having 3 assists--the AND statement fails, and the IF Function returns "No."

The visual representation below illustrates the application of this strict filtering logic across the dataset in Excel:

=IF(AND(A2="Mavs", B2="Guard", C2>20, D2>4), "Yes", "No")

We can then drag and fill this formula down to each remaining cell in column E:

E2 =IF(AND(A2="Mavs", B2="Guard", C2>20, D2>4), "Yes", "No")

	A	B	C	D	E	F
1	**Team**	**Position**	**Points**	**Assists**	**Mavs & Guard & Points>20 & Assists>4?**	
2	Mavs	Guard	22	5	Yes	
3	Mavs	Guard	29	8	Yes	
4	Mavs	Forward	32	10	No	
5	Mavs	Forward	15	4	No	
6	Mavs	Guard	19	6	No	
7	Warriors	Forward	22	5	No	
8	Warriors	Guard	25	8	No	
9	Warriors	Guard	20	8	No	
10	Warriors	Forward	19	10	No	
11	Warriors	Forward	14	2	No	
12						
13						
14						
15						
16						
17						
18						
19						
20						

The logic executed by the formula is defined by the absolute necessity of all four conditions:

If the value in the Team column was "Mavs" and the value in the Position column was "Guard" and the value in the Points column was greater than 20 and the value in the Assists column was greater than 4, return **Yes**.

Else, if at least one condition is not met then return **No**.

Method 3: Utilizing IF with OR Logical Operator

In contrast to the strict requirements imposed by the AND function, the OR function provides flexibility, returning TRUE if any one of its conditions evaluates to TRUE. This operator is highly useful when we are looking for records that satisfy at least one of several potential criteria, broadening the scope of the assessment significantly.

Similar to the AND method, the OR function replaces the main logical test within the IF Function. If OR returns TRUE (because one, two, three, or all four conditions were met), the IF function executes the "value if true." The only scenario in which the OR function returns FALSE is if every

single nested condition fails. This makes it ideal for inclusion criteria or identifying records that fall into any specified group.

For example, if we are searching for players who are either members of the "Mavs" team, regardless of position, OR any player who scores highly, OR any player who handles the ball frequently (high assists), we use the OR operator. This ensures that a player who is not a "Mavs" guard but has stellar points and assists will still be included in the positive result set. This logic fundamentally changes the way the four conditions interact, moving from joint necessity to individual sufficiency.

The structure of the formula is nearly identical to the AND method, substituting only the logical operator:

=IF(OR(A2="Mavs", B2="Guard", C2>20, D2>4), "Yes", "No")

Applying Broad Inclusion Rules with IF and OR

To demonstrate the inclusionary nature of the OR function, we apply the formula to cell **E2**, seeking players who meet at least one of the four specified conditions: Team equals "Mavs", Position equals "Guard", Points are greater than 20, or Assists are greater than 4. We then propagate this formulae down column E.

As evident in the resulting table, the OR logic yields far more "Yes" results compared to the AND logic. A player might not be a "Mavs" guard, but if they score 21 points (meeting the C2>20 criterion), they instantly qualify for a "Yes" designation. This methodology is indispensable for creating expansive filters or identifying all potential candidates that satisfy any minimum bar set by the criteria.

Understanding the distinction between AND and OR is crucial for accurate analysis using Boolean Logic within conditional formulas. While AND narrows the focus to only those records that are perfect matches across all criteria, OR widens the focus to capture all records that possess at least one desired attribute.

We can type the following formula into cell **E2** to return "Yes" if one of four conditions are met for a specific player or "No" if none of the conditions are met:

=IF(OR(A2="Mavs", B2="Guard", C2>20, D2>4), "Yes", "No")

We can then drag and fill this formula down to each remaining cell in column E:

	A	B	C	D	E
1	**Team**	**Position**	**Points**	**Assists**	**Mavs or Guard or Points>20 or Assists>4?**
2	Mavs	Guard	22	5	Yes
3	Mavs	Guard	29	8	Yes
4	Mavs	Forward	32	10	Yes
5	Mavs	Forward	15	4	Yes
6	Mavs	Guard	19	6	Yes
7	Warriors	Forward	22	5	Yes
8	Warriors	Guard	25	8	Yes
9	Warriors	Guard	20	8	Yes
10	Warriors	Forward	19	10	Yes
11	Warriors	Forward	14	2	No
12					
13					
14					
15					
16					
17					
18					
19					
20					

The mechanism of this formula is based on sufficiency rather than necessity:

If the value in the Team column was "Mavs" or the value in the Position column was "Guard" or the value in the Points column was greater than 20 or the value in the Assists column was greater than 4, return **Yes**.

Else, if none of the conditions are met then return **No**.

Summary of Conditional Approaches and Best Practices

When faced with four or more conditions, choosing the appropriate structure is paramount for achieving the correct analytical result. The three methods--Nested IF Statements, IF with AND, and IF with OR--serve fundamentally different purposes in data segmentation.

The Nested IF approach is primarily used for sequential decision-making that leads to categorical outputs. They are best deployed when you need to assign a single grade or status based on where a single input value falls within a range of possibilities. The flow is exclusive: if condition A is true,

conditions B, C, and D are ignored. Conversely, the logical operators are used when you need to evaluate multiple independent columns or criteria simultaneously to determine a single outcome. The AND function establishes a high barrier for entry, requiring perfection across all four measures, ideal for filtering highly specific subsets. The OR function establishes a low barrier, acting as an inclusionary filter for any record meeting at least one key metric.

For modern Excel users (Excel 2019 and later), an alternative method for multi-condition classification that often replaces complex nesting is the IFS function. This dedicated function streamlines the process by removing the need for repeated closing parentheses and clearly separating the condition-value pairs. While not covered in detail here, mastering the fundamental nested approach remains essential for backward compatibility and deeper logical comprehension. When constructing multi-condition formulae, ensure you prioritize the logical order of conditions in Nested IF Statements and utilize named ranges or cell references for criteria thresholds whenever possible to improve maintainability.