Google Sheets: Use an IF Function with 3 Conditions

Authored by

Mohammed looti

October 27, 2025

Introduction to Advanced Conditional Logic in Google Sheets

In the highly dynamic world of data analysis and effective management, the capacity to implement sophisticated conditional logic is absolutely paramount. This capability enables users to automate complex decision-making processes directly within their datasets, transforming simple, raw information into valuable, actionable business insights. Google Sheets, recognized as a highly powerful, cloud-based spreadsheet application, provides a robust suite of tools designed for this purpose, with the foundational IF function residing at its core.

While the basic structure of the IF function is undeniably effective for evaluating single conditions, real-world data scenarios invariably demand far more intricate evaluations involving multiple criteria. This comprehensive guide delves deeply into three primary, high-utility methods for successfully implementing the IF function when faced with three or more conditions in Google Sheets. By mastering these techniques, you will significantly enhance your ability to categorize, filter, and analyze vast quantities of data with extreme precision and efficiency. We will meticulously explore the construction of formulas capable of handling numerous conditions, providing detailed, practical examples for each unique approach.

Developing proficiency in these advanced methods will dramatically boost your overall productivity and elevate your analytical capabilities within Google Sheets. Whether your task involves assigning hierarchical ratings based on specific numerical ranges, identifying records that must strictly meet all specified criteria, or flagging entries that satisfy at least one flexible criterion, these three methods offer versatile and powerful solutions to the most common data manipulation challenges.

Understanding the Foundational IF Function

Before transitioning into complex multi-conditional scenarios, it is absolutely essential to firmly grasp the fundamental mechanics and structure of the core IF function. At its essence, this function performs a simple evaluation of a logical expression and subsequently returns one predefined value if that expression is determined to be **TRUE**, and a different predefined value if the expression is determined to be **FALSE**. Its standard syntax is structured as follows: `=IF(logical_expression, value_if_true, value_if_false)`.

The required **logical_expression** is any statement that can ultimately be assessed as having only two possible outcomes: **TRUE** or **FALSE**. This might involve a simple comparison operator, such as `C2<15`, a text comparison like `A2="Mavs"`, or a greater-than check such as `C2>25`. The **value_if_true** is the output Google Sheets will display if the logical expression holds true, while the **value_if_false** is displayed when the expression is false. These resulting values can be various data types, including text strings (which must be enclosed in double quotes), numerical values,

references to other formulas, or even strategically left as empty strings.

The true power of the IF function lies not only in its inherent simplicity but also in its crucial capacity to serve as a foundational building block for constructing far more sophisticated conditional logic. When analysts need to handle more than just two potential outcomes or when multiple distinct criteria must be rigorously evaluated, we must employ advanced techniques. These include strategically nesting IF statements or combining the core IF function with specialized logical functions like the AND function and the OR function.

Method 1: Implementing a Nested IF Function for Multiple Outcomes

One of the most widely used and intuitive techniques for managing more than two distinct conditions using the IF function is through the application of nested IF functions. This powerful approach requires placing one complete IF function inside another, specifically situating the subsequent IF function within the current IF function's **value_if_false** argument. This configuration establishes a cascading chain of conditions, ensuring that each subsequent IF statement is only evaluated if all preceding conditions have failed (evaluated to **FALSE**). This methodology is exceptionally valuable when the requirement is to assign varying labels or numerical values based upon a series of ordered, mutually exclusive thresholds or established categories.

Although highly effective and versatile, excessively deep nested IF functions can quickly become complex, difficult to read, and challenging to debug effectively. However, for a moderate number of conditions, typically three or four, they provide a straightforward, sequential pathway to achieving multiple distinct and conditional outcomes. Let us examine a practical application using a sample dataset within Google Sheets. Imagine a scenario where the objective is to assign performance ratings (such as Bad, OK, Good, or Great) based on an athlete's total recorded points.

The following formula demonstrates the structure for a nested IF function that systematically categorizes player performance based on the value found in the Points column:

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

This formula, when correctly entered into the starting cell D2, begins by evaluating the numerical value present in cell C2 (the Points column). If C2 contains a value less than 15, the formula immediately returns the rating "Bad". If that condition is not met, the formula proceeds to the next nested IF statement. If C2 is found to be less than 20 (but implicitly 15 or greater, since the first condition failed), it returns "OK". This precise pattern of sequential evaluation continues until a condition is successfully met or, finally, the default value "Great" is returned if all preceding conditions in the chain are proven false.

The subsequent visual example clearly illustrates how this expertly constructed formula is applied

across a typical dataset. The standard procedure is to enter this formula into the first relevant data row (e.g., D2) and then utilize the spreadsheet's convenient "drag and fill" functionality to apply it automatically to all subsequent rows, ensuring that the cell reference (C2) adjusts dynamically for each row.

	A	B	C	D	E
1	**Team**	**Position**	**Points**		
2	Mavs	Guard	22		
3	Mavs	Guard	30		
4	Mavs	Forward	19		
5	Warriors	Guard	14		
6	Warriors	Forward	18		
7	Warriors	Forward	12		
8	Heat	Guard	29		
9	Heat	Guard	31		
10	Heat	Forward	23		
11					
12					
13					
14					
15					
16					
17					
18					
19					
20					

After successfully dragging the formula down through column D, you will observe the precise, calculated performance ratings generated for every player in the dataset.

D2 =IF(C2<15, "Bad", IF(C2<20, "OK", IF(C2<25, "Good", "Great")))

	A	B	C	D	E	F
1	**Team**	**Position**	**Points**	**Status**		
2	Mavs	Guard	22	Good		
3	Mavs	Guard	30	Great		
4	Mavs	Forward	19	OK		
5	Warriors	Guard	14	Bad		
6	Warriors	Forward	18	OK		
7	Warriors	Forward	12	Bad		
8	Heat	Guard	29	Great		
9	Heat	Guard	31	Great		
10	Heat	Forward	23	Good		
11						
12						
13						
14						
15						
16						
17						
18						

Let's systematically break down the logical flow inherent in this specific formula:

First Check: If the value housed in the Points column (C2) is strictly less than 15, the formula immediately returns the rating **"Bad"**.

Second Check: Else (meaning C2 is 15 or greater), the formula then checks if the value in the Points column is less than 20. If this is true, it returns **"OK"**.

Third Check: Else (meaning C2 is 20 or greater), the formula subsequently checks if the value in the Points column is less than 25. If this condition is true, it returns **"Good"**.

Final Default: Else (meaning C2 is 25 or greater), it signifies that none of the preceding conditions were met, so the formula executes the final default action, returning **"Great"**.

Method 2: Combining IF with AND Logic for Strict Conditions

When the objective is to rigorously verify if **all** of several distinct conditions are true simultaneously, the dedicated AND function becomes an indispensable analytical tool. The AND function accepts multiple logical expressions as its arguments and is designed to return **TRUE** only if every single one of those expressions evaluates to true; if even one expression is false, it returns **FALSE**. This

mechanism makes it the perfect choice for scenarios that demand strict, simultaneous adherence to multiple criteria.

By strategically embedding the AND function directly within the IF function's primary **logical_expression** argument, users can construct highly focused and powerful conditional statements. The rigorous Boolean logic governing the AND operation ensures that the **value_if_true** for the overarching IF function is executed only when every specified condition is successfully met.

Let us consider a challenging situation where you need to identify players who fulfill three specific requirements: they must belong to the exact team named "Mavs", they must play the specific position "Guard", and they must have recorded a score above a defined point threshold (e.g., more than 25 points). Here is the precise formula structure required to achieve this strict evaluation:

=IF(AND(A2="Mavs", B2="Guard", C2>25), "Yes", "No")

This formula simultaneously checks three distinct conditions for the data contained in row 2: Does the team equal "Mavs"? Does the position equal "Guard"? Are the points greater than 25? Crucially, if all three statements are true, the formula returns "Yes"; otherwise, if even one condition fails, it returns "No".

In a similar manner to the nested IF example, you should enter this formula into the starting cell D2 and then drag it down the column to automatically populate the results for the entire dataset.

D2 =IF(AND(A2="Mavs", B2="Guard", C2>25), "Yes", "No")

	A	B	C	D
1	**Team**	**Position**	**Points**	**Mavs and Guard and Points > 25?**
2	Mavs	Guard	22	No
3	Mavs	Guard	30	Yes
4	Mavs	Forward	19	No
5	Warriors	Guard	14	No
6	Warriors	Forward	18	No
7	Warriors	Forward	12	No
8	Heat	Guard	29	No
9	Heat	Guard	31	No
10	Heat	Forward	23	No
11				
12				
13				
14				
15				
16				
17				
18				
19				
20				
21				
22				

Upon a careful review of the calculated results, the following strict logic is clearly demonstrated:

Strict Match: 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 25, the formula returned **"Yes"**.

Failure: Else, if at least one single condition among the three was not successfully met, the formula automatically returned **"No"**.

It is important to notice that in this specific visual example, only a single row successfully returned "Yes". This vividly underscores the fundamental characteristic of the AND function: for the entire expression to be true, every single specified condition must be fully satisfied without exception.

Method 3: Utilizing IF with OR Logic for Flexible Conditions

Standing in distinct contrast to the strict, simultaneous requirements imposed by the AND function, the specialized OR function introduces significant flexibility by verifying if **any** of the specified

conditions are met. The OR function accepts multiple logical expressions and returns **TRUE** if a minimum of one of those expressions evaluates to true; conversely, it returns **FALSE** only if all expressions are false. This flexibility is invaluable when the analytical need is to broadly identify records that satisfy any one criterion from a defined set.

By integrating the OR function within the IF function's central **logical_expression**, users can construct conditions that are met when a much broader set of criteria applies. The inclusive nature of Boolean logic's OR operation guarantees that the **value_if_true** for the IF function is triggered as long as even one condition successfully holds true.

Let's utilize a scenario identical to the previous example, but with a critical modification: this time, our goal is to identify players who are either members of the "Mavs" team, OR play the "Guard" position, OR have scored more than 25 points. This means a player only needs to satisfy one of these three conditions to be successfully flagged by the formula.

You should enter the following formula into cell D2 to return "Yes" if at least one of the three conditions is met for a particular player, or "No" only if none of the conditions are met:

=IF(OR(A2="Mavs", B2="Guard", C2>25), "Yes", "No")

This formula checks the identical three conditions as the AND example, but the difference is profound: if *any* of the conditions (Team is "Mavs", Position is "Guard", or Points are greater than 25) are evaluated as true, it returns "Yes". Only if *all three* conditions are found to be false will the formula return the value "No".

We can subsequently drag and fill this formula down to each remaining cell in column D to process the entire dataset:

D2 =IF(OR(A2="Mavs", B2="Guard", C2>25), "Yes", "No")

	A	B	C	D
1	**Team**	**Position**	**Points**	**Mavs or Guard or Points > 25?**
2	Mavs	Guard	22	Yes
3	Mavs	Guard	30	Yes
4	Mavs	Forward	19	Yes
5	Warriors	Guard	14	Yes
6	Warriors	Forward	18	No
7	Warriors	Forward	12	No
8	Heat	Guard	29	Yes
9	Heat	Guard	31	Yes
10	Heat	Forward	23	No
11				
12				
13				
14				
15				
16				
17				
18				

Here is a summary of the inclusive logic that this formula successfully accomplished:

Inclusive Match: 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 25, the formula returned **"Yes"**.

Failure: Else, if absolutely none of the conditions were met, the formula returned **"No"**.

It is important to note that six total rows returned "Yes" in this example, which is a significantly greater number than achieved with the strict AND function. This is solely because each of those rows successfully met at least one of the three criteria, clearly demonstrating the highly inclusive nature of the OR function.

Choosing the Right Conditional Approach for Your Data

The decision of whether to employ Nested IF functions, IF(AND(...)), or IF(OR(...)) should always hinge directly on the specific logical requirements your data analysis demands. Each of these three methods serves a profoundly distinct purpose within Google Sheets, and a thorough understanding of their core differences is critical to writing formulas that are both efficient and accurately reflect the intended business logic.

You should utilize a **Nested IF function** when the requirement is to evaluate a sequential series of conditions, where the evaluation of each subsequent condition is explicitly dependent on the failure of the one preceding it. This is the ideal structure for assigning ordered categories based on numerical ranges (e.g., academic grading systems or performance tiers) where only one definitive outcome is logically possible for any given input value. Analysts should, however, remain mindful that excessive levels of nesting can quickly lead to formulas that are difficult to read, challenging to audit, and strenuous to maintain over time.

You must opt for **IF combined with AND logic** when your true outcome mandates that all specified conditions must be met simultaneously and without exception. This approach effectively enforces the strictest possible criteria, making it essential for tasks such as identifying a precise demographic that must fit multiple characteristics (e.g., matching a specific age, geographical location, and income bracket). If even one condition within the argument is false, the entirety of the AND statement will fail, returning the false value.

Finally, choose **IF combined with OR logic** when your true outcome is satisfied if at least one condition from a group is met. This provides the highest degree of flexibility, making it invaluable for flagging records that meet any one of a set of acceptable criteria (e.g., identifying users who have engaged with page A, page B, or page C). The OR statement is true as long as one condition successfully holds.

Conclusion and Best Practices for Complex Formulas

The developed proficiency in effectively utilizing the IF function alongside multiple conditions represents a fundamental cornerstone of advanced Google Sheets expertise. By skillfully leveraging nested IF functions, or by combining IF with the powerful operators AND and OR, users gain immense analytical power to automate highly complex decision-making processes directly within their spreadsheets. These versatile tools empower you to systematically categorize data, rigorously filter records, and derive exceptionally specific insights based on sophisticated conditional logic.

To guarantee that your constructed formulas remain robust, highly reliable, and easily maintainable, consider adhering to these essential best practices:

Start Simple and Outline: Always begin the process by clearly outlining the desired logical flow in simple, plain language before attempting to translate that logic directly into a complex formula structure.

Use Parentheses Wisely: The correct and judicious use of parentheses is absolutely crucial for defining the precise order of operations, especially in complex nested IF statements and combined AND/OR statements, where misplacement can lead to incorrect results.

Test Iteratively and Thoroughly: Always test your newly developed formula on a small, representative subset of your data first to confirm unequivocally that it behaves exactly as expected before deploying it across your entire, valuable dataset.

Document Your Logic: For formulas that are particularly complex or dense, make it a practice to add inline comments or maintain separate, detailed notes explaining the purpose and function of each condition. This documentation will greatly assist future users (and your future self) in quickly understanding and efficiently modifying the formula when necessary.

Mastering these advanced conditional techniques will not only significantly streamline your data processing workflows in Google Sheets but will also profoundly enhance your overall analytical thinking, enabling you to approach complex data challenges with assured confidence and high precision.

Additional Resources

To further expand your overall proficiency and maximize your potential within Google Sheets, we encourage you to explore the following related tutorials. These resources explain how to successfully perform other common data tasks and unlock a broader array of its powerful features.