How to Use the IF Function in Google Sheets to Change Values Based on the Month

Authored by

stats writer

January 31, 2026

Google Sheets: Leveraging the IF Function for Month-Based Decision Making

The IF function in Google Sheets stands as an exceptionally powerful instrument, enabling users to implement sophisticated decision-making processes directly within their spreadsheets based on predefined criteria. Fundamentally, this function constructs a logical statement that meticulously evaluates whether a specific condition holds true or false.

When combined with date and time functions, particularly the MONTH function, the IF function acquires unique capabilities. This integration allows for dynamic evaluations and actions contingent upon the month currently represented by a date in a designated cell. Such temporal decision-making is indispensable for organizing, automating, and analyzing complex financial or tracking datasets. By mastering the synergy between these two functions, users can dramatically enhance the efficiency and accuracy of their data management within Google Sheets.

Core Syntax and Temporal Criteria Setup

To harness the power of conditional logic based on a specific month, we utilize the following standard formula structure. This structure is foundational for nearly all month-based filtering and calculation tasks you might encounter in sophisticated reporting or auditing.

=IF(MONTH(A2)=10, B2, 0)

This specific formula initiates a meticulous check: it determines if the numerical representation of the month extracted from the date stored in cell **A2** is precisely equal to **10**. The number 10, in this context, unequivocally corresponds to October, the tenth month of the Gregorian calendar year.

If the logical statement evaluates to **TRUE**--meaning the date in A2 is indeed in October--then the function executes its primary directive and returns the exact value found in cell **B2**. Conversely, if the condition is determined to be **FALSE**--the month is any month other than October--the formula executes the alternative result and simply returns a numerical value of **0**. This simple structure forms the cornerstone of our month-specific data manipulation strategies.

Understanding the MONTH Function Utility

Before diving into complex examples, it is critical to fully grasp the role of the MONTH function. The MONTH function is designed purely to extract the month component from a valid date serial number or date string and return it as an integer between 1 (January) and 12 (December). This numerical output is essential because the IF function relies on clean numerical comparisons to

execute its logic.

If the date in cell A2 is formatted as "2024-03-15," the function `MONTH(A2)` will return the number 3. If A2 contains "12/25/2023," the function will return 12. Using this numerical output ensures that the comparison operator (the equals sign, =) within the logical statement is robust and unambiguous, regardless of how the date itself is displayed or formatted in the spreadsheet.

It is important to ensure that the source cell (A2 in our example) contains a valid date recognizable by Google Sheets. If the cell contains plain text that Sheets cannot interpret as a date, the MONTH function will typically return an error value, preventing the overall IF function from executing correctly. Careful validation of date formats is a prerequisite for reliable month-based logic.

Example: Applying IF Logic to Sales Data

To demonstrate this functionality in a real-world scenario, consider the following practical example. We possess a dataset within Google Sheets which meticulously records the total sales figures achieved on various distinct dates throughout the year at a specific retail establishment. The objective is to isolate and analyze sales that occurred exclusively during a target month, such as October.

	A	B	C	D
1	**Date**	**Sales**		
2	10/12/2023	5		
3	10/15/2023	6		
4	10/19/2023	19		
5	11/5/2023	14		
6	11/30/2023	18		
7	12/7/2023	13		
8	12/22/2023	10		
9	12/23/2023	12		
10	1/4/2024	12		
11	1/20/2024	7		
12				
13				
14				
15				

Our specific task is to implement a conditional function that performs the following two actions: first, it must return the actual sales value if the corresponding date in Column A falls within October.

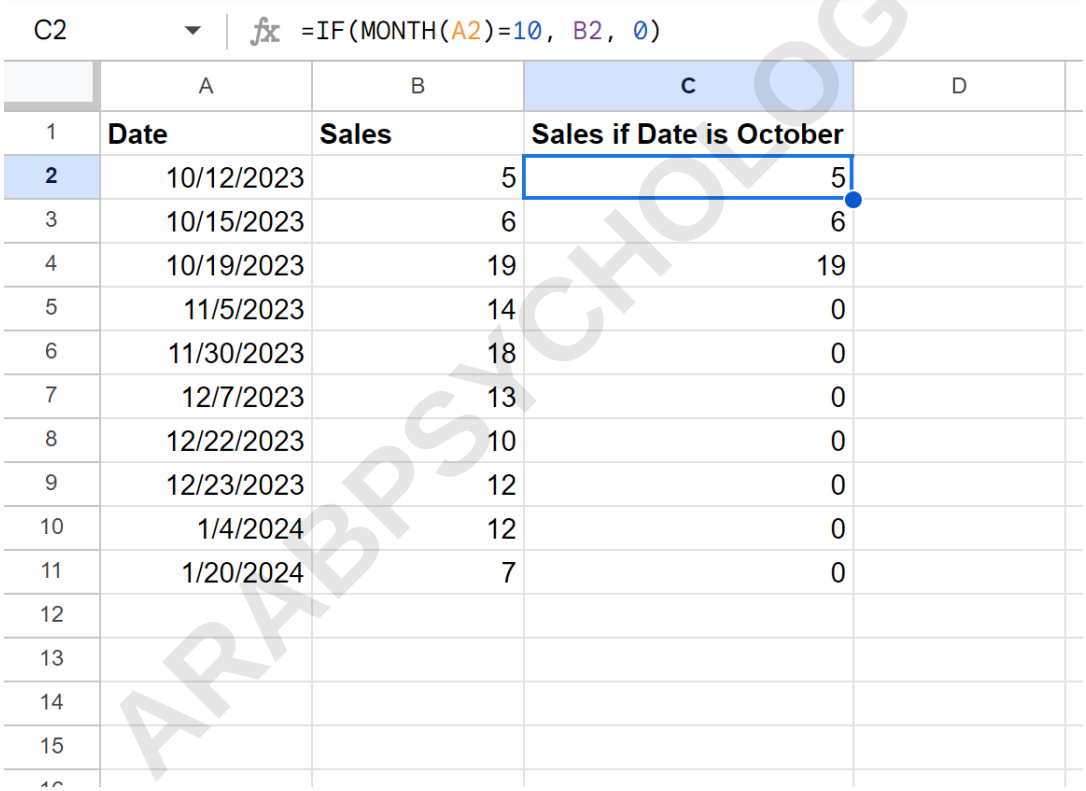
Second, if the date does not fall within October, the function must return a placeholder value of zero, effectively filtering out irrelevant data points for our current analysis.

To achieve this precise filtering and extraction, we enter the following formula directly into cell **C2**, and subsequently drag it down the column to apply the logic to every row in the dataset:

=IF(MONTH(A2)=10, B2, 0)

Execution and Interpretation of the Basic Formula

The screenshot below clearly illustrates the outcome of applying the IF function combined with the MONTH function across the sales data. Observe how Column C, labeled "October Sales," accurately reflects the conditional results based on the date provided in Column A.



	A	B	C	D
1	**Date**	**Sales**	**Sales if Date is October**	
2	10/12/2023	5	5	
3	10/15/2023	6	6	
4	10/19/2023	19	19	
5	11/5/2023	14	0	
6	11/30/2023	18	0	
7	12/7/2023	13	0	
8	12/22/2023	10	0	
9	12/23/2023	12	0	
10	1/4/2024	12	0	
11	1/20/2024	7	0	
12				
13				
14				
15				

When the date encountered in Column A corresponds to a date in October (i.e., the MONTH function returns 10), the formula successfully extracts and returns the corresponding sales figure from Column B. For instance, the sales recorded for 10/10/2023 (row 3) are accurately pulled into Column C.

Conversely, for any date outside of the target month, such as 09/15/2023 (row 2) or 11/01/2023 (row 6), the condition `MONTH(A2)=10` evaluates to false. In these instances, the formula executes

the false value argument, resulting in a return value of zero. This method provides an exceptionally clean and efficient way to segregate data for monthly reporting or budget reconciliation.

Expanding Logic with Multiple Conditions (The OR Function)

While the basic IF function is highly effective for targeting a single month, real-world data analysis frequently requires evaluating multiple months simultaneously. For example, a budget might lump Q4 sales together, requiring filtering for October, November, and December. Google Sheets allows us to embed the OR function directly within the logical test argument of the IF statement.

The OR function enables us to test several individual conditions; if even one of those conditions proves true, the entire logical statement returns TRUE. This dramatically expands the flexibility of our month-based filtering. To filter for sales that occurred in either October (10) or November (11), we modify the formula as follows:

=IF(OR(MONTH(A2)=10, MONTH(A2)=11), B2, 0)

This restructured formula first utilizes the MONTH function twice within the OR wrapper, checking if the month is 10 OR if the month is 11. If either check passes, the OR function returns TRUE, triggering the IF function to return the value in B2. This technique is invaluable when analyzing quarterly results or specific seasonal sales windows.

Visualizing the OR Logic Implementation

The subsequent screenshot demonstrates the application of the formula incorporating the OR function. Notice how the resulting column now captures sales data from both October and November, while still excluding dates from September or December (if present in the full dataset).

C2 =IF(OR(MONTH(A2)=10, MONTH(A2)=11), B2, 0)

	A	B	C
1	**Date**	**Sales**	**Sales if Date is October or November**
2	10/12/2023	5	5
3	10/15/2023	6	6
4	10/19/2023	19	19
5	11/5/2023	14	14
6	11/30/2023	18	18
7	12/7/2023	13	0
8	12/22/2023	10	0
9	12/23/2023	12	0
10	1/4/2024	12	0
11	1/20/2024	7	0
12			
13			
14			
15			
16			

In this enhanced scenario, if the date in column A corresponds to October or November, the formula successfully executes the TRUE result, returning the associated sales figure from Column B. This provides targeted extraction suitable for cumulative period reporting, allowing users to isolate specific periods within a large dataset without resorting to manual sorting or complex pivot tables.

If the date in column A falls outside the specified months of October or November, the entire logical statement nested within the OR function evaluates to FALSE. Consequently, the formula defaults to the specified FALSE result, which is the numerical value zero. This systematic application of conditional logic ensures accurate and customizable data segregation based on temporal parameters.

Advanced Conditional Filtering: Using Nested IF or IFS

While the combination of IF and MONTH is potent, situations may arise where more than two outcomes are required based on different months. For instance, you might want to categorize sales as "Q1 Sales" if the month is 1, 2, or 3, "Q2 Sales" if 4, 5, or 6, and so forth. In such instances, you have two primary options: using nested IF statements or the more modern IFS function (introduced in Google Sheets specifically to simplify multiple IF conditions).

A nested IF function would look like this (simplified for Q1 vs Q4): `=IF(MONTH(A2) <= 3, "Q1",`

`IF(MONTH(A2) >= 10, "Q4", "Mid-Year"))`. This structure chains multiple conditional checks, but it can become cumbersome and difficult to audit if many conditions are added. Each subsequent IF is the FALSE result of the previous one, demanding careful placement of parentheses.

The IFS function provides a cleaner solution for handling sequential month-based checks. It evaluates multiple conditions simultaneously and returns the value corresponding to the first true condition. For example: `=IFS(MONTH(A2)<=3, "Q1 Bonus", MONTH(A2)<=6, "Q2 Review", MONTH(A2)<=9, "Q3 Review", TRUE, "Q4 Bonus")`. By utilizing the greater clarity of the IFS function, analysts can manage complex temporal categorizations with significantly reduced risk of formula errors.

Practical Applications for Financial Reporting and Budgeting

The ability to conditionally select or manipulate data based on the month is fundamental to high-level financial analysis and efficient budgeting processes. This technique allows organizations to shift their focus from static, manual filtering to dynamic, formula-driven reporting that updates automatically as new data is entered into the Google Sheets document.

Specific practical applications include generating automatic monthly expense reports, where the formula only tallies costs incurred in the current fiscal month, or creating dynamic budget comparisons where actual versus budgeted figures are filtered monthly. Furthermore, these month-based filters are essential for year-over-year comparative analysis, allowing an analyst to easily isolate all occurrences of April data across five different years within a single, unified dataset. This targeted extraction capability significantly reduces the time spent on data preparation and increases the reliability of the derived insights.

Another powerful use case involves inventory management. By tracking inventory levels and movement dates, the IF(MONTH) combination can automatically flag items that were stagnant during a specific peak sales month, indicating potential stocking issues or misallocation of resources. The automation inherent in this conditional logic ensures that operational decisions are informed by the most current and relevant monthly data available.

Troubleshooting and Common Errors in Date Functions

While robust, the combination of IF and MONTH is highly sensitive to input data quality, particularly regarding date formatting. One of the most common errors encountered is the `#VALUE!` error, which typically occurs when the MONTH function attempts to process a cell that does not contain a recognizable date format.

To prevent this, users must ensure that all cells referenced by the MONTH function (e.g., cell A2)

are correctly formatted as dates. If the date has been imported as text, functions like `DATEVALUE` might be necessary to convert the string into a valid date serial number before the MONTH function can successfully extract the month integer. Additionally, ensure that the numerical comparison within the logical statement (e.g., `=10`) is accurate, as confusing the month number (10 for October) with the day or year can lead to erroneous results without generating a formal error message.

Conclusion and Further Learning Opportunities

The strategic combination of the IF function and the MONTH function provides Google Sheets users with an essential toolkit for managing time-series data. By allowing the spreadsheet to make conditional decisions based on the month, complex datasets can be analyzed, filtered, and automated with precision and efficiency. Whether you are generating simple single-month reports or implementing complex multi-conditional financial models using OR or IFS, mastering this foundational technique significantly elevates your data analysis capabilities.

We encourage readers to explore related functions that further enhance date-based logic, such as the `YEAR` and `DAY` functions, which can be combined with `IF` to create even more granular temporal conditions. Furthermore, integrating these conditional formulas with array functions (like `ARRAYFORMULA`) allows for efficient application of these month-based rules across massive ranges without manually dragging the formula, pushing the boundaries of automated data processing within Google Sheets.

Related Tutorials for Enhanced Google Sheets Proficiency

The following instructional guides delve into complementary techniques necessary for high-level data manipulation and efficient spreadsheet management in Google Sheets:

How to Use the QUERY Function for Dynamic Reporting
Implementing Advanced Filtering Using FILTER and UNIQUE
Best Practices for Handling Date and Time Formatting Errors