How to Use the IF Function in Google Sheets by Month

Authored by

stats writer

January 15, 2026

Introduction: Leveraging Date Intelligence in Spreadsheets

The question of whether the IF function in Google Sheets can be utilized based on specific monthly criteria is met with a definitive "Yes." This capability unlocks sophisticated methods for data analysis and reporting, moving beyond simple arithmetic to incorporate powerful conditional logic based on temporal parameters. By integrating the core conditional power of the IF function with date-specific functions, users can create dynamic spreadsheets that react intelligently to the date contained within a cell.

This integration allows financial analysts, project managers, and data entry specialists to automate decision-making processes within their spreadsheets. For instance, imagine a scenario where performance metrics need to be evaluated differently during peak seasons, or budgets must be automatically adjusted based on the current quarter. By isolating the month from a full date timestamp, the IF function gains the precision necessary to execute these nuanced rules. This not only streamlines recurring analytical tasks but also significantly enhances the accuracy and efficiency of monthly summary report generation.

The inherent flexibility derived from comparing the extracted month (represented numerically, e.g., 1 for January, 12 for December) against a specified target month allows for numerous advanced applications. Users can swiftly determine whether a specific sales target was achieved during, for example, the crucial month of December, returning an automated "Target Met" or "Underperformed" response. Furthermore, this technique is essential for categorizing vast amounts of transactional data, enabling the rapid creation of robust monthly summaries, accrual schedules, or fiscal planning reports. Understanding this fundamental combination--the IF function and the MONTH function--is a cornerstone of efficient spreadsheet mastery.

Understanding the IF Function and Its Syntax

The IF function is the backbone of conditional execution in spreadsheets, designed to test a condition and return one value if the condition is deemed true, and another value if it is false. Its syntax is deceptively simple yet immensely powerful: `=IF(logical_expression, value_if_true, value_if_false)`. The `logical_expression` is the core component; this must be a statement that evaluates to either **TRUE** or **FALSE**. When incorporating time-based criteria, the challenge lies in constructing a logical expression that successfully evaluates the date information present in a referenced cell.

In standard applications, the logical expression might involve direct comparisons, such as checking if one cell's value is greater than another (e.g., `A1 > 100`). However, when dealing with dates, direct comparison often proves ineffective because a date is stored as a sequential serial number in Google Sheets, and simply comparing the full date serial numbers rarely achieves the desired monthly isolation. To overcome this, we must introduce a specialized function capable of parsing

the date and returning only the required component--in this case, the month number. This separation process is critical for focusing the conditional test solely on the temporal segment of interest.

By using the MONTH function inside the logical expression, we effectively transform a complex date serial number into a simple integer (1 through 12). This integer is then easily compared against a target number, making the entire logical expression clean and actionable. If the month number matches the target, the condition is met, and the `value_if_true` is executed. If it does not match, the `value_if_false` is returned, often represented by a zero, an alternative calculation, or a text string. This layered approach ensures that the conditional action is triggered only when the month criterion is precisely satisfied.

Isolating Time: The Role of the MONTH Function

To successfully apply monthly conditions, we must first master the MONTH function. This utility is specifically designed to extract the month component from a given date value. When provided with a cell reference containing a valid date (e.g., `A2`, where A2 contains `10/15/2024`), the MONTH function processes the date serial number and returns an integer between **1** (January) and **12** (December). The syntax is straightforward: `=MONTH(date)`. This numerical output is the critical bridge that connects date information to numerical conditional logic.

The importance of this extraction cannot be overstated, particularly when dealing with large datasets where filtering or aggregating based on specific time periods is necessary. Without the MONTH function, comparing dates based only on their month would be computationally complex, potentially requiring elaborate string manipulation or array formulas. The simplicity of `MONTH()` provides an elegant and highly efficient way to parse temporal data, making it readily comparable within any spreadsheet function that requires a numerical argument.

For example, if cell **A2** contains the date `03/05/2023`, executing `=MONTH(A2)` will return the value **3**. If **A2** contains `12/31/2025`, the result is **12**. This output is then seamlessly integrated into the `logical_expression` of the IF function. We can then set up a comparison such as `MONTH(A2) = 10`. This expression reads, "Is the month extracted from cell A2 equal to 10 (October)?" This clear structure transforms abstract date manipulation into concrete, testable arithmetic, forming the foundation for our monthly conditional tests.

Constructing the Basic IF-MONTH Formula

The most fundamental application of monthly conditions involves checking if a date falls within a single, specified month. The standard formula structure combines the `MONTH` function within the logical test of the `IF` function. This arrangement ensures that the condition is only satisfied when

the month component of the date matches the desired numerical indicator.

You can use the following formula to implement the IF function based on a month in Google Sheets:

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

This particular formula meticulously checks if the month extracted from the date stored in cell **A2** is equivalent to the integer **10**. As per the standard date interpretation in spreadsheets, the value **10** corresponds to the tenth month of the year, which is **October**. This test forms the heart of the conditional statement, evaluating to **TRUE** only for dates falling within October, regardless of the year or day.

If the date in **A2** indeed falls within October (i.e., the logical expression returns **TRUE**), the formula is designed to immediately return the corresponding value found in cell **B2**. Cell **B2** typically holds data related to that specific transaction or entry, such as sales figures, inventory counts, or project statuses. This ensures that only data associated with the target month is selected and outputted.

Otherwise, if the month extracted from **A2** is any month other than October (i.e., the logical expression returns **FALSE**), the formula executes the `value_if_false` component, which in this example is simply the numerical value **0**. This use of zero is common for filtering data, effectively nullifying the entry if the date criteria are not met, thereby isolating the desired monthly records for subsequent analysis or reporting. The following example shows how to use this formula in practice.

Practical Example: Filtering Sales Data by Month

To illustrate the practical utility of this conditional structure, let us consider a common business requirement: filtering a large sales dataset to extract only those transactions that occurred during a specific, high-priority month. Suppose we maintain a spreadsheet detailing daily sales figures, where column A contains the transaction date and column B contains the total sales amount for that date. Our objective is to generate a summary column that shows sales figures exclusively for October, returning zero for any other month.

Suppose we have the following initial dataset in Google Sheets that shows the total sales made on various dates at some store. This data spans several months, necessitating a dynamic filter mechanism:

	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				

The goal is precise: we want to use an IF function that evaluates the date in column A. If the date belongs to October (month 10), the corresponding sales value from column B should be returned into the new summary column (Column C). If the date belongs to any other month, the formula must return a value of zero, thereby creating a clean, October-specific sales column ready for summation or further calculation.

Step-by-Step Implementation of the Single-Month Filter

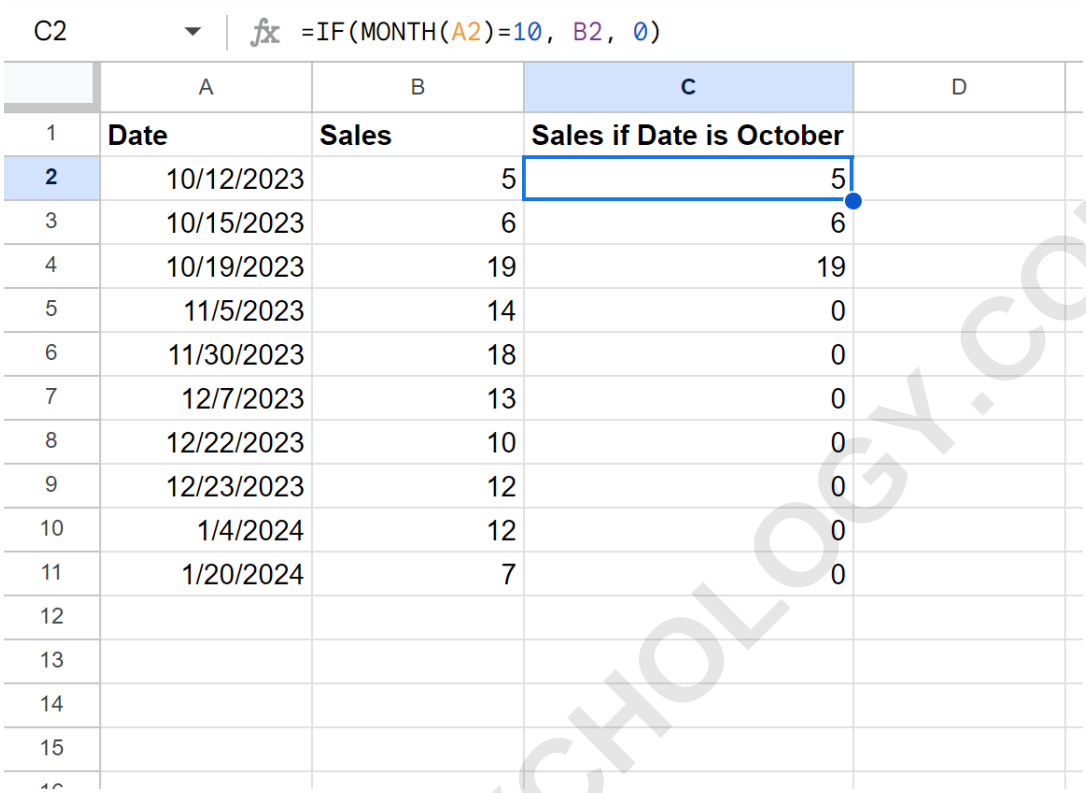
The implementation process is straightforward and relies on correctly placing the combined `IF` and `MONTH` formula in the first row of the results column, and then dragging it down to apply to the entire dataset. Specifically, we will start by inputting the formula into cell **C2**, which is the corresponding row for our first transaction date in **A2**. The formula we deploy is the basic single-condition check:

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

Once entered into cell **C2**, we execute the formula. Google Sheets first evaluates `MONTH(A2)`. If A2 holds a date like `10/01/2024`, the result is 10. Since `10 = 10` is **TRUE**, the formula returns the sales value from **B2**. If A2 holds `09/25/2024`, the result is 9. Since `9 = 10` is **FALSE**, the formula returns **0**.

The following screenshot demonstrates the application and results of this formula, clearly showing

how only the October sales figures are preserved, while all other monthly entries are suppressed to zero. This filtered column (Sales in October) provides an immediate visual and functional separation of the data:



	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				

As you can observe, whenever the date in column A falls within October, the formula returns the corresponding value in the sales column. For all other instances, such as dates in September or November, the formula returns the mandated value of zero. This technique is exceptionally useful for creating dynamic summaries; for example, one could easily calculate the total October sales by simply summing column C, knowing that all non-October data has been neutralized. Note that you can also use multiple conditions within the IF function.

Expanding Conditions: Utilizing OR Logic for Multiple Months

While checking for a single month is useful, often analytical requirements dictate that criteria must be met across a range of months, such as an entire financial quarter (e.g., Q4 includes October, November, and December). To handle multiple conditions simultaneously within a single **IF** function, we must incorporate Boolean operators like `OR` or `AND` into the logical test. The `OR` function is particularly well-suited for this task, as it allows the overall condition to evaluate to **TRUE** if **any** of the nested conditions are **TRUE**.

To filter sales data for both October (10) and November (11), we embed two separate `MONTH`

comparisons within the `OR` function, which in turn acts as the single logical test for the primary `IF` statement. This creates a powerful, nested formula that checks for multiple temporal matches. The structure requires that each month check be a complete logical expression separated by a comma within the `OR` function: `OR(MONTH(A2)=10, MONTH(A2)=11)`.

The comprehensive formula for checking multiple months is structured as follows:

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

This formulation provides a robust solution for quarter-end reporting or seasonal analysis. If the date in column A yields a month value of 10 **OR** a month value of 11, the entire `OR` statement evaluates to **TRUE**, and the formula proceeds to return the sales value from column B. If the date is neither October nor November, the `OR` statement evaluates to **FALSE**, and the formula returns zero. This demonstrates the enhanced capability of combining conditional operators with date extraction for highly flexible data manipulation.

Visualizing Multi-Month Filtering Results

Applying the multi-month formula across the dataset demonstrates how efficiently Google Sheets can handle complex temporal filtering. By entering the formula into cell C2 and copying it down the column, we immediately generate a column that is exclusive to the October-November period, excluding transactions from all other months, such as September or December.

The following screenshot visually represents the output when the `OR` condition is successfully implemented. Notice how the resulting column, now representing "Sales in Oct/Nov," captures the sales figures for dates falling in both month 10 and month 11, while consistently zeroing out data from month 9:

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			

If the date in column A is identified as either **October** or **November**, then the formula returns the corresponding value in the sales column, successfully passing the expanded conditional logic test. Conversely, if the date falls outside this specified range, the formula executes the `value_if_false` clause, ensuring a clear return of zero. This technique is invaluable for segmenting data based on multi-month criteria, enabling precise calculations for quarterly performance reviews, seasonal inventory adjustments, or multi-period financial comparisons.

Conclusion and Further Conditional Possibilities

In summary, the utilization of the IF function in conjunction with the MONTH function provides spreadsheet users with an essential tool for implementing highly specific, time-based conditional logic. This method transcends static analysis, allowing dynamic evaluation of data based purely on the month of the entry. Whether filtering for a single month or creating complex criteria covering multiple periods using operators like `OR` and `AND`, this capability is foundational for robust temporal data management in Google Sheets.

Beyond simple filtering, this methodology can be extended to perform complex nested operations. For instance, one could use a nested `IF` structure to apply different commission rates based on which month a sale occurred, or employ the results within array formulas to dynamically summarize monthly expenditures across an entire fiscal year. Mastering the combined power of `IF`

and `MONTH` is a key step toward automating intricate reporting processes and achieving greater efficiency in data analysis.

The following tutorials explain how to perform other common and valuable data manipulation tasks in Google Sheets, building upon the principles of conditional execution and data extraction:

How to combine the **IF** function with the `DAY` or `YEAR` function for deeper temporal analysis.
Techniques for using `ARRAYFORMULA` to apply these monthly conditions to entire columns without dragging the formula.
Advanced methods for creating dynamic reporting dashboards based on extracted date components.