How to Use VLOOKUP in Google Sheets to Find Values Within a Range

Authored by

stats writer

February 1, 2026

The VLOOKUP function stands as an indispensable tool within Google Sheets, designed specifically for efficient data retrieval across complex tables. Unlike simple search operations, VLOOKUP excels at finding corresponding data points based on a specific criterion. While its primary use is often associated with exact lookups, its true power is unlocked when utilized for range-based searching, allowing users to identify which bracket a given numerical value falls into. This capability is absolutely crucial when managing large datasets, such as those involving tiered pricing, grading scales, or bonus calculations, where the exact search key might not exist in the source table.

Mastering the range lookup aspect of VLOOKUP transforms how analysts handle structured information, enabling automated classification and calculation that would otherwise require tedious manual checking or complex nested IF statements. By providing the function with the necessary parameters--the lookup value, the data range, and the return column index--VLOOKUP meticulously scans the designated table. When configured for an approximate match, it intelligently pinpoints the relevant boundary. This methodology dramatically increases efficiency, minimizing the time and effort spent manually sifting through thousands of rows of data to find the appropriate categorization, thereby enhancing the overall reliability and analytical utility of the spreadsheet.

Google Sheets: Use VLOOKUP to Find Value That Falls Between Range

Configuring VLOOKUP for Approximate Range Matching

To leverage the full potential of VLOOKUP for identifying values that fall within a defined range, the user must explicitly instruct the function to perform an approximate search rather than an exact one. This critical distinction is controlled by the function's fourth argument, the `range_lookup` parameter. By setting this parameter to **TRUE**, we tell VLOOKUP to look for the largest value that is less than or equal to the **lookup_value**, effectively placing the lookup value within a specific bracket defined by the start of the range.

This technique is foundational for lookups involving continuous numerical scales. For instance, if you are looking up a score of 85 on a grading scale where the tiers are 0, 70, 80, and 90, VLOOKUP (set to TRUE) will find 80, as it is the largest value in the range that is still less than or equal to 85. The corresponding result from the adjacent column (e.g., 'B' grade) is then returned. Understanding this mechanism--that the function always seeks the greatest preceding value--is vital for constructing your reference table correctly and predicting the output accurately when dealing with non-exact matches.

It is important to emphasize that while the concept is called an approximate match, its operation is

highly precise and based on specific numerical rules. The system relies on the assumption that your lookup table defines the lower bounds of each range segment. Therefore, **VLOOKUP** is not searching for a close numerical value randomly, but rather it is mapping the input value against a predefined set of ascending thresholds, ensuring that the lookup value is correctly categorized into the appropriate interval defined by the reference data.

Deconstructing the VLOOKUP Syntax and Parameters

The operational framework of the VLOOKUP function relies on four distinct parameters, each playing a vital role in directing the search logic and determining the output. Understanding the proper syntax is paramount to ensure the function executes correctly, especially when transitioning from simple lookups to complex range matching. The standard structure is consistently maintained across various spreadsheet platforms, including Google Sheets, providing a consistent framework for data analysis.

The basic VLOOKUP structure is defined as:

VLOOKUP (lookup_value, table_array, col_index_num,)

Let's break down the role of each argument in the context of range lookups:

lookup_value: This is the specific value you are attempting to locate within the first column of your data range. When performing a range lookup, this is typically a numerical score, sales figure, or quantity that needs categorization.
table_array: This defines the entire range of cells containing the reference data. Crucially, the search value must exist in the **first column** of this array, and the values you wish to return must be contained within the subsequent columns of the same array.
col_index_num: This is a numerical index indicating the column number within the `table_array` that contains the result you want to be returned. For example, if your `table_array` spans columns A through C, and you want a result from column C, the `col_index_num` would be 3.
range_lookup: This Boolean argument determines the search methodology. Setting it to **TRUE** (or omitting it, as TRUE is the default) enables the approximate match required for range searching, while **FALSE** mandates an exact match.

By using a value of **TRUE** for the last argument, you specifically instruct the function to identify the appropriate range boundary, enabling the efficient classification of the **lookup_value** into one of the predefined tiers established in the **table_array**. This setting is what differentiates a simple exact search from a powerful range analysis tool, providing flexibility when dealing with continuous variables.

Prerequisites for Effective VLOOKUP Range Matching: The Sorting Requirement

A fundamental requirement, often overlooked by novice users, must be strictly met for the VLOOKUP function to return accurate results when the `range_lookup` argument is set to **TRUE**. The values in the first column of the `table_array` (the lookup column) must be sorted in **ascending order** (from least to greatest). Failure to adhere to this sorting prerequisite will lead to highly unpredictable and frequently incorrect results, completely undermining the accuracy of the range lookup mechanism.

The reason for this strict sorting rule lies in how VLOOKUP performs the approximate search. The function uses a highly optimized binary search algorithm when `range_lookup` is TRUE. This algorithm works by iteratively eliminating half of the remaining data set based on comparisons. It reads down the sorted list until it encounters a value that is greater than the **lookup_value**. Once that larger value is found, the function steps back to the previous row, identifying that row as the corresponding lower bound of the range. If the data is not sorted, this sequential comparison fails, and the function cannot reliably establish the correct lower threshold.

Therefore, before executing any range-based VLOOKUP, analysts must first verify the integrity and order of their reference data. If the data is improperly ordered, the spreadsheet application must be used to sort the entire reference table based on the key column (the first column of the `table_array`). Maintaining this sorting integrity ensures that the search logic functions as intended, identifying the next largest value that is less than the lookup value, and consequently returning the precise corresponding data for the defined range segment.

Practical Example: Calculating Employee Bonuses Based on Sales Tiers

To illustrate the application of VLOOKUP for range matching, consider a common business scenario: determining employee bonuses based on tiered sales performance. This setup requires correlating a specific sales figure (the lookup value) with a defined bonus amount (the return value) based on the bracket the sales figure falls into. The structure of the reference table is critical, as the sales column must define the minimum required sales to achieve the corresponding bonus level.

Suppose we have the following dataset in Google Sheets that shows the minimum sales needed by an employee at a company to earn a certain bonus amount:

	A	B	C	D
1	**Sales**	**Bonus**		
2	0	0		
3	500	25		
4	1000	50		
5	2000	75		
6	5000	100		
7	10000	150		
8				
9				
10				
11				
12				
13				
14				
15				

This reference table establishes clear, ascending thresholds:

If an employee achieves sales starting from **0** (up to 499), they earn **$0** as a bonus.
If an employee achieves sales starting from **500** (up to 999), they earn **$25** as a bonus.
If an employee achieves sales starting from **1,000** (up to 1,999), they earn **$50** as a bonus.
If an employee achieves sales starting from **2,000** (up to 4,999), they earn **$75** as a bonus.
If an employee achieves sales starting from **5,000** (up to 9,999), they earn **$100** as a bonus.
If an employee achieves sales of **10,000** or more, they earn **$150** as a bonus.

Notice that the "Sales" column (Column A) is perfectly sorted in ascending numerical order (0, 500, 1000, 2000, etc.). This adherence to the sorting rule makes the table ready for accurate range analysis using **VLOOKUP**, allowing us to accurately categorize any sales figure against these defined tiers without needing to specify the upper limit of each bracket in the table itself.

Step-by-Step Implementation of Approximate Matching

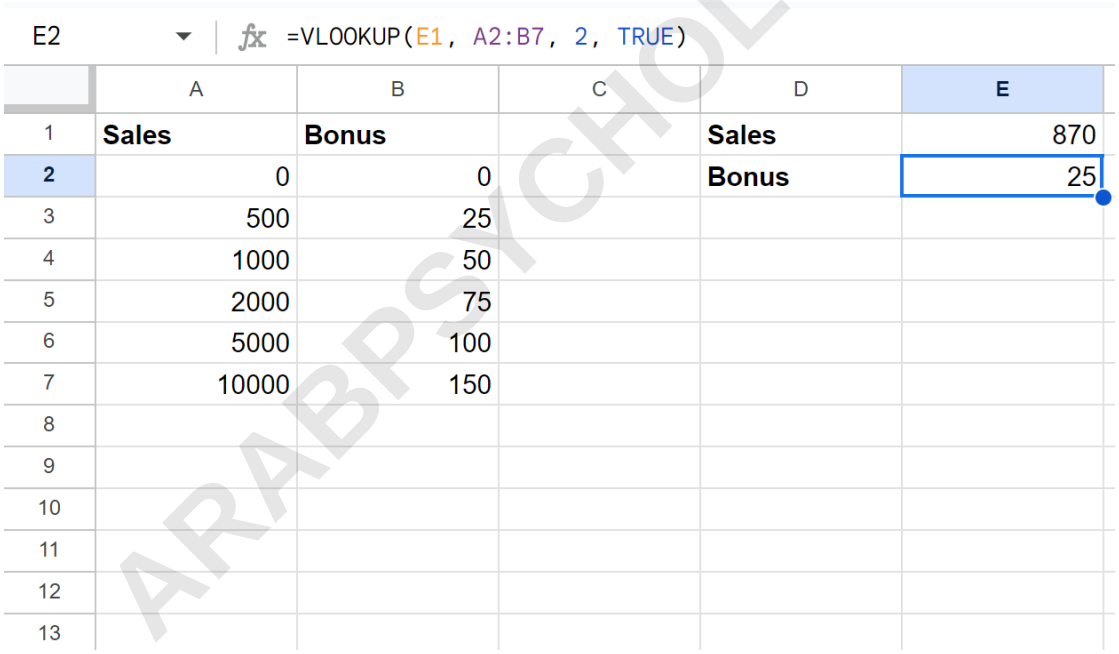
Now, let us proceed with the actual implementation of the VLOOKUP formula to determine the bonus for a specific employee whose sales figure is known. Suppose we want to look up the value of **870** in the Sales column (Column A) and return the corresponding Bonus amount from the reference table (Columns A and B). We define the lookup value in cell **E1** and wish to display the result in cell **E2**.

The core objective is to locate the largest sales threshold that is less than or equal to 870. Given our reference table, VLOOKUP should recognize that 500 is the largest preceding threshold, which corresponds to a bonus of $25. To achieve this result, we must construct the formula using the approximate match setting:

=VLOOKUP(E1, A2:B7, 2, TRUE)

In this formula: **E1** is the `lookup_value` (870); **A2:B7** is the `table_array` covering the Sales and Bonus columns; **2** is the `col_index_num` because Bonus is the second column in the array; and **TRUE** explicitly enables the range lookup feature. This configuration directs the function to perform the necessary approximate search based on the predefined, sorted sales thresholds, ensuring the correct bonus tier is identified automatically.

The following screenshot demonstrates the successful application of this formula within the Google Sheets interface, confirming that the function accurately identifies the appropriate bonus amount based on the employee's sales figure relative to the tiered structure:



Interpreting Results and Avoiding Common Pitfalls

When the formula executes with a `lookup_value` of **870**, the VLOOKUP function begins its search in column A. Since the exact value of **870** does not exist, the function looks for the next largest value in the Sales column that is less than or equal to 870. It successfully identifies the value of **500**. Subsequently, it moves across to the second column of the array (the Bonus column) and returns the corresponding value, which was **25**. This confirms that the employee falls into the $25

bonus bracket (sales between 500 and 999).

It is paramount to understand the implication of the search method: the formula returned the bonus corresponding to the threshold of **500**, not the threshold of 1000. If the data were not sorted, or if the reference table started defining ranges incorrectly (e.g., listing the maximums instead of the minimums), the output would be erroneous. A common pitfall is forgetting the critical sorting prerequisite; if the first column in your lookup range is not sorted from least to greatest, the VLOOKUP function will use faulty logic and could return unexpected results, often without generating an error message, making debugging complex.

Another important consideration for users of approximate matching is defining the starting point. If the `lookup_value` is less than the smallest value in the first column of the `table_array` (e.g., if we looked up -10, but the minimum sales listed is 0), VLOOKUP will typically return an `#N/A` error, as it cannot find any value that is less than or equal to the lookup value. Analysts must always ensure their lookup value falls within the numerical scope defined by the sorted reference table to guarantee a valid result. For complete documentation and further examples regarding the nuances of the VLOOKUP function, refer to the official Google Sheets documentation.

Further Exploration in Google Sheets Data Management

The successful implementation of range lookups using VLOOKUP opens the door to numerous advanced data management techniques. While VLOOKUP is powerful, it is also important to explore alternative or complementary functions, such as **INDEX MATCH** or **XLOOKUP** (if available), which offer greater flexibility, particularly in scenarios where the lookup column is not the leftmost column of the data range, or when searching horizontally instead of vertically.

Understanding these fundamental lookup mechanisms is crucial for efficient spreadsheet operations. The following tutorials explain how to perform other common tasks in Google Sheets: