

How to Use VLOOKUP to Find and Return the Value in the Next Cell in Excel

Authored by
stats writer

February 2, 2026

RECOMMENDED CITATION

stats writer (2026). *How to Use VLOOKUP to Find and Return the Value in the Next Cell in Excel*. PSYCHOLOGICAL SCALES. Retrieved from <https://scales.arabpsychology.com/?p=129113>

The VLOOKUP function in Excel is a powerful utility designed to search for a specific value within a table and return a corresponding value from a designated column. When utilized with the range lookup argument set to FALSE, it guarantees the retrieval of an **exact match** for the lookup criteria. While highly efficient for standard lookups, achieving the goal of returning the value in the adjacent cell (the cell immediately following the match) requires integrating VLOOKUP's searching capability with other dynamic positional functions. This advanced technique is crucial for optimizing data organization and analysis, eliminating the reliance on tedious manual data manipulation. Furthermore, leveraging complementary functions such as INDEX or IFERROR allows for robust handling and manipulation of the returned values, ensuring data integrity and flexibility.

Excel: Use VLOOKUP to Return Value in Next Cell

The VLOOKUP Misconception and the Need for Positional Control

Often, users wish to employ a lookup function like VLOOKUP in Excel to locate a specific value and retrieve the corresponding data not from the same row, but from the cell located immediately below it in a different column. Although VLOOKUP is excellent for horizontal column offsets, it fundamentally lacks the inherent capability to dynamically offset its return position vertically relative to the identified match row. This limitation makes it unsuitable for complex positional requirements.

The standard structure of VLOOKUP requires a fixed column index number, meaning it always returns data from the same relative column for every match it finds. Since VLOOKUP cannot calculate a return row index

based on the match position plus or minus an offset, a more sophisticated approach is necessary to achieve this vertical shift. This requires migrating from **VLOOKUP** to a combination of functions that offers granular control over both the search index and the final retrieval position.

The easiest and most robust way to achieve this dynamic vertical offsetting is by utilizing the powerful pairing of the **INDEX** and **MATCH** functions. This methodology allows for the precise calculation of the desired row index, enabling the retrieval of the value located in the cell immediately *next* (i.e., one row below) the position of the exact match.

Introducing the Power Combination: **INDEX** and **MATCH**

The **INDEX** and **MATCH** combination offers unparalleled flexibility over traditional lookup methods. While **VLOOKUP** searches and retrieves simultaneously based on fixed relative positions, **INDEX-MATCH** decouples the process: **MATCH** finds the position, and **INDEX** retrieves the value at that exact calculated position. This separation is key to implementing dynamic offsets.

The **MATCH** function is used first to determine the exact row number where the lookup value resides within a specific column range. For instance, if the lookup value "Lakers" is found on the fourth row of the defined range, **MATCH** returns the number 4. This numerical output becomes the foundation for our positional adjustment.

The critical step is applying the arithmetic offset. To target the cell immediately below the match, we simply add 1 to the result generated by **MATCH**. If **MATCH** returned 4, the new index becomes 5. This calculated index is then passed to the **INDEX** function, instructing it to retrieve the value from the fifth position in the designated return column. This technique provides superior performance and flexibility over traditional lookup methods, especially when dealing with large or evolving **datasets**.

Formula Breakdown: Retrieving the Next Cell Value

The comprehensive formula structure for returning the value in the next cell involves carefully nesting the functions to achieve sequential execution. This formula is written as:

=INDEX(B2:B11,MATCH("Lakers",A2:A11,0)+1)

This particular formula is designed to perform a specific lookup: it searches for the text string "Lakers" within the range A2:A11 (the lookup column) and subsequently returns the corresponding value from the range B2:B11 (the return column) that is situated exactly one cell below the row where the match was identified. The parentheses and arguments must be handled precisely to ensure correct data flow.

The `MATCH("Lakers", A2:A11, 0)` segment is executed internally first. The `0` signifies that an exact match is required. If, for example, the team "Lakers" is found in the third row of the A2:A11 range, the MATCH function outputs the row index 3. This index indicates the relative position of the match within the range A2:A11.

Following the match calculation, the offset `+1` is applied to the result. Using the example above, $3 + 1$ equals 4. This value, 4, is the calculated index that precisely targets the next cell in the sequence. Finally, the outer function, `INDEX(B2:B11, 4)`, takes this index and retrieves the value found in the fourth row of the return range

B2:B11, thereby successfully isolating the value in the next cell relative to the lookup.

Detailed Example: Using INDEX and MATCH in Practice

Suppose we are working with a sports dataset in Excel, containing performance statistics like the points scored by various basketball teams. This data is organized with 'Team' names in column A and 'Points' scored in column B, spanning rows 2 through 11. We aim to locate the entry for "Lakers" in the Team column and retrieve the score from the Points column corresponding to the team listed immediately after the Lakers.

The initial structure of our dataset is shown below, illustrating the relationship between the lookup column (Team) and the return column (Points):

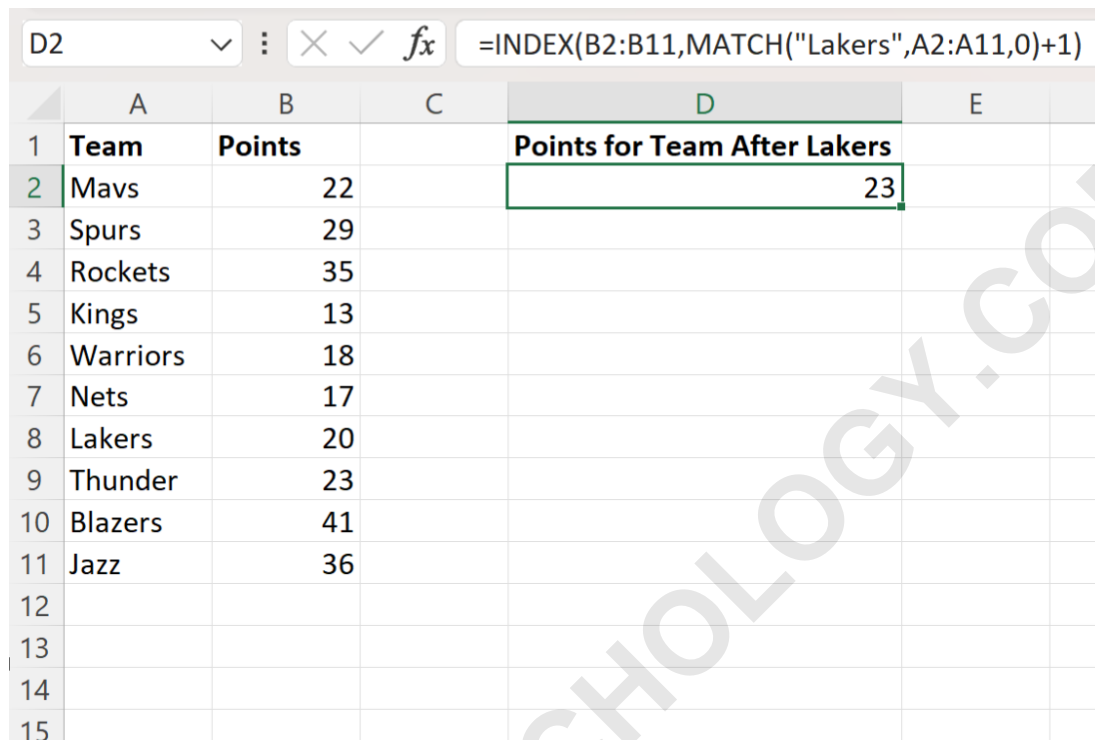
	A	B	C	D	E
1	Team	Points			
2	Mavs	22			
3	Spurs	29			
4	Rockets	35			
5	Kings	13			
6	Warriors	18			
7	Nets	17			
8	Lakers	20			
9	Thunder	23			
10	Blazers	41			
11	Jazz	36			
12					
13					
14					
15					

To execute the dynamic lookup and return the subsequent value, we insert the calculated formula into a designated output cell, such as cell D2. The formula specifies the return range B2:B11, the lookup target "Lakers", the lookup range A2:A11, and the crucial row offset of $+1$:

=INDEX(B2:B11,MATCH("Lakers",A2:A11,0)+1)

Upon entering the formula, Excel processes the request. The **MATCH** function identifies the row index for "Lakers", adds one to this index, and the **INDEX** function retrieves the value at the newly calculated

offset position. The result of this operation is visually confirmed in the following screenshot:



	A	B	C	D	E
1	Team	Points		Points for Team After Lakers	
2	Mavs	22		23	
3	Spurs	29			
4	Rockets	35			
5	Kings	13			
6	Warriors	18			
7	Nets	17			
8	Lakers	20			
9	Thunder	23			
10	Blazers	41			
11	Jazz	36			
12					
13					
14					
15					

The formula successfully returns a value of 23. This output confirms that the formula correctly located the entry for the Lakers, calculated the index of the cell immediately following its corresponding points value, and retrieved the contents of that next cell in the Points column. This demonstrates a reliable method for extracting sequential data points relative to a key identifier within a structured dataset.

Advanced Modification: Retrieving the Previous Cell Value

The adaptability of the **INDEX** and **MATCH** combination allows for easy adjustments to retrieve data in any relative position, including the cell located one row *above* the match. This is achieved by simply altering the arithmetic operator applied to the row index generated by the **MATCH** function. Instead of adding 1, we subtract 1.

To return the value from the cell that is one row above the points value corresponding to the Lakers, the formula is modified as follows:

```
=INDEX(B2:B11,MATCH("Lakers",A2:A11,0)-1)
```

This subtraction operation calculates the row index that precedes the identified match. If the **MATCH** function returns an index of 4 (for the Lakers position), the formula now calculates 4 minus 1, resulting in an index of 3. The **INDEX** function then retrieves the value from the third row within the specified return range B2:B11. It is vital to remember that this approach will yield an error if the lookup value happens to be the very first item in the lookup range (e.g., in cell A2), as subtracting one would result in an invalid row index of zero.

The following screenshot demonstrates the practical result of implementing this formula modification, confirming the successful retrieval of the preceding data point:

The screenshot shows an Excel spreadsheet with the following data:

	A	B	C	D	E
1	Team	Points		Points for Team Before Lakers	
2	Mavs	22		17	
3	Spurs	29			
4	Rockets	35			
5	Kings	13			
6	Warriors	18			
7	Nets	17			
8	Lakers	20			
9	Thunder	23			
10	Blazers	41			
11	Jazz	36			
12					
13					
14					

The formula bar for cell D2 shows: `=INDEX(B2:B11,MATCH("Lakers",A2:A11,0)-1)`

The formula returns a value of 17, which is the exact score located in the Points column one cell above the points value for the Lakers. This reinforces the utility of the INDEX-MATCH structure in providing backward-looking retrieval capabilities, offering complete directional control for advanced data analysis in Excel.

Handling Boundary Conditions and Error Prevention

While the INDEX-MATCH structure is highly powerful for dynamic offsetting, users must account for boundary conditions to maintain data integrity and prevent errors. The two most common errors relate to the lookup value not being found and the offset calculation pushing the row index beyond the limits of the defined data range.

If the lookup value specified in the **MATCH** function is absent from the lookup range, the function returns the standard `#N/A` error. To manage this cleanly, the entire formula should be enveloped in the `IFERROR` function. This ensures that a custom message, such as "Team Not Found" or a blank cell, is returned instead of a confusing error code, thereby improving the readability and robustness of the report or dashboard.

Furthermore, boundary errors often occur when the offset calculation (`+1` or `-1`) targets a row outside the defined return range. For instance, using `+1` when the lookup value is the very last item in the list will attempt to retrieve data from a non-existent row, resulting in a `#REF!` error. Similarly, using `-1` when the lookup value is the first item will result in a row index of 0, also

producing a `#REF!` error. To mitigate these risks, advanced users can incorporate conditional logic using the `IF` function nested with `MATCH` to check the position before allowing the **INDEX** retrieval to proceed. This preemptive check ensures the formula only executes when the offset position is valid.

Conclusion: Mastering Advanced Lookup Techniques

The aspiration to retrieve the value in the next cell relative to a match point is a common challenge that highlights the limitations of standard **VLOOKUP**. By embracing the flexibility offered by the combined **INDEX** and **MATCH** functions, analysts gain the ability to perform highly dynamic, positional lookups in **Excel**. This technique allows for precise vertical offsetting, whether looking forward (using `+N`) or backward (using `-N`), relative to the found value.

The ability to calculate and retrieve data based on relative position is fundamental for automating complex sequential analysis and data reporting tasks. By understanding how **MATCH** determines the index and how simple arithmetic controls the offset, users can unlock the full potential of Excel for handling intricate

datasets. This expertise transforms manual data handling into efficient, scalable formula-driven workflows.

Further Excel Resources

The following tutorials explain how to perform other common operations in Excel:

ARABPSYCHOLOGY.COM