What is the difference between the Excel functions SEARCH and FIND?

Authored by

stats writer

November 18, 2025

RECOMMENDED CITATION

stats writer (2025). *What is the difference between the Excel functions SEARCH and FIND?*. PSYCHOLOGICAL SCALES. Retrieved from https://scales.arabpsychology.com/?p=95325

ARABPSYCHOLOGY.COM

Introduction: Locating Substrings in Excel

When working with large datasets in Excel, the need often arises to precisely locate a specific sequence of characters, or a substring, within a larger text string. Microsoft Excel provides two primary functions for this task: the **SEARCH function** and the **FIND function**. While both are designed to return the starting position of a substring within a designated text field, the subtle yet crucial differences in their operational behavior dictate which one should be employed for specific data manipulation tasks.

Understanding the distinctions between these two powerful text-locating tools is fundamental for advanced spreadsheet analysis. Their output provides the numeric position of the first character of the search string. If the substring is not found, both functions will return the **#VALUE!** error. However, before deploying either function, it is imperative to consider two primary characteristics that separate them, relating to their handling of character case and their ability to interpret special symbols for pattern matching.

The following discussion will thoroughly detail the operational variances, syntax requirements, and practical applications of these functions, utilizing real-world examples to illustrate how differing data requirements necessitate the selection of one function over the other.

Core Distinctions: SEARCH vs. FIND

The operational differences between **SEARCH** and **FIND** are centered around two key areas: character sensitivity and pattern recognition capability. These characteristics allow users to select the appropriate tool based on whether they need an exact, literal match or a more flexible, generalized search. Grasping these differences is essential for writing robust and efficient Excel formulas.

We can summarize the fundamental functional contrasts between the two string manipulation tools as follows:

The **SEARCH function** is inherently **not case-sensitive**. This means it treats uppercase and lowercase letters as identical. When searching for a character, it will find the first occurrence regardless of capitalization, making it ideal for general text parsing.
Conversely, the **FIND function** is strictly **case-sensitive**. When searching for a specific character, it will only match that character if its capitalization exactly matches the search string. This ensures high precision for data validation tasks.
The **SEARCH function** supports the use of **wildcard characters**, specifically the asterisk (*****) and the question mark (**?**), allowing for flexible pattern matching within text strings.
The **FIND function** does not recognize or interpret **wildcard characters**. If these symbols are used in the search string for **FIND**, they are treated as literal characters that must be located within

ARABPSYCHOLOGY.COM

the target text, making pattern matching impossible.

Syntax and Arguments Explained

Although their underlying mechanisms differ significantly, **SEARCH** and **FIND** share the exact same syntax structure, utilizing three required arguments, two of which are mandatory and one is optional. Understanding these arguments is crucial for correct implementation. The generic syntax is `=FUNCTION(find_text, within_text, )`.

The first argument, **`find_text`**, specifies the substring or character sequence that the function is attempting to locate. This argument must be supplied as a text string, typically enclosed in quotation marks, or as a reference to a cell containing the desired text. Crucially, if the **SEARCH function** is used, this argument is the location for including any desired **wildcard characters**; if **FIND** is used, this must represent the exact literal string sought, including precise capitalization.

The second mandatory argument, **`within_text`**, dictates the cell or text string in which the search operation will be conducted. This is the larger string that will be scanned from left to right. The position returned by the function is always the numeric index of the first character of the located `find_text` relative to the start of the `within_text` string, where the first character holds position 1.

The final, optional argument, , allows the user to specify the character position from which the search should commence. If this argument is omitted, the search automatically starts at the beginning (position 1). Utilizing `start_num` is a powerful technique, often employed in conjunction with other text functions or nested within iterative formulas, to locate subsequent occurrences of a substring after the first instance has been found and recorded.

Practical Demonstration: The Case Sensitivity Difference

The most fundamental practical distinction lies in the functions' approach to character capitalization. To effectively demonstrate this, consider the following column of basketball team names, which includes text strings with varying use of uppercase and lowercase letters. We will attempt to locate the starting position of the lowercase letter "s" within each text string.

ARABPSYCHOLOGY.COM

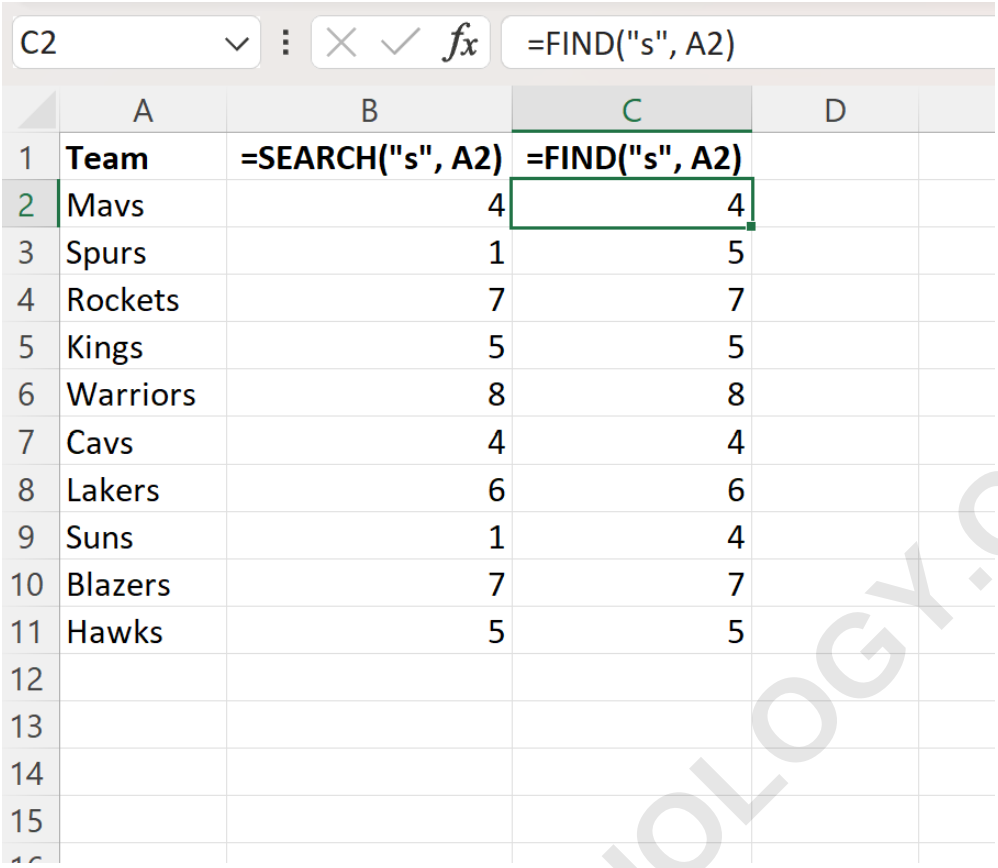
	A	B	C	D	E
1	**Team**				
2	Mavs				
3	Spurs				
4	Rockets				
5	Kings				
6	Warriors				
7	Cavs				
8	Lakers				
9	Suns				
10	Blazers				
11	Hawks				
12					
13					
14					
15					
16					
17					

ARABPSYCHOLOGY.COM

To execute this comparison, we will input the respective formulas into cells **B2** (for **SEARCH**) and **C2** (for **FIND**), both targeting the text in cell **A2**. The critical point here is that the search string specified is strictly lowercase: "s".

B2 (Case-Insensitive): **=SEARCH("s", A2)**
C2 (Case-Sensitive): **=FIND("s", A2)**

By clicking and dragging these formulas down to apply them to all rows in columns B and C, we can observe the outputs below, highlighting how the two functions handle the capitalization of the initial character in the string "Spurs" versus the internal characters in other strings.



	A	B	C	D
1	**Team**	**=SEARCH("s", A2)**	**=FIND("s", A2)**	
2	Mavs	4	4	
3	Spurs	1	5	
4	Rockets	7	7	
5	Kings	5	5	
6	Warriors	8	8	
7	Cavs	4	4	
8	Lakers	6	6	
9	Suns	1	4	
10	Blazers	7	7	
11	Hawks	5	5	
12				
13				
14				
15				

ARABPSYCHOLOGY.COM

Analyzing the SEARCH Function Results

In Column B, the results generated by the **SEARCH function** confirm its case-insensitive operation. When instructed to locate "s", **SEARCH** treats 's' and 'S' as identical. It scans the string and returns the position of the very first character that is alphabetically an 'S', regardless of its literal capitalization.

For the entry **Spurs** in cell A2, the **SEARCH** formula returns **1**. This outcome is direct evidence of its case-insensitivity: the function immediately finds the uppercase 'S' at the first position, considers it a match for the lowercase search string "s", and terminates the search, returning the earliest possible position.

For strings like **Clippers** and **Mavericks**, **SEARCH** returns **7** and **10** respectively, corresponding to the position of the lowercase 's'. This behavior confirms that **SEARCH** is concerned primarily with the character's alphabetical value, making it highly valuable for general text analysis where case consistency cannot be guaranteed or is unimportant.

Analyzing the FIND Function Results

Conversely, the **FIND function** in Column C demonstrates strict case sensitivity. When **FIND** is deployed with the search string "s" (lowercase), it will ignore any uppercase 'S' characters it encounters, continuing its scan until an exact, lowercase "s" match is located.

Reviewing the result for **Spurs** (A2), the formula `=FIND("s", A2)` yields the position **5**. The function bypasses the initial uppercase 'S' (position 1). It continues through 'p', 'u', and 'r' until it reaches the fifth character, which is the lowercase 's', thus confirming the match at position 5. Had the search string been "S", **FIND** would have returned 1.

The difference between the **SEARCH** result (1) and the **FIND** result (5) for the text "Spurs" serves as the definitive illustration of their differing case handling methodologies. When processing data where capitalization signifies meaning or requires exact compliance with established standards, **FIND** is the function of choice.

Utilizing Wildcard Characters for Flexible Searching

The second major operational variance is the support for pattern matching using **wildcard characters**. The **SEARCH function** recognizes the question mark (**?**) to represent any single character and the asterisk (*****) to represent any sequence of characters. The **FIND function**, by contrast, possesses no such pattern-matching capability.

Suppose the goal is to find the position of a three-character sequence where the sequence ends in "rs" but the preceding character can be any single character, such as "urs" or "ers". We can achieve this flexible search using the single-character wildcard, resulting in the search string **"?rs"**. We apply the formulas again to our team name data:

B2 (Wildcard Enabled): **=SEARCH("?rs", A2)**
C2 (Wildcard Disabled): **=FIND("?rs", A2)**

The application of these formulas demonstrates a radical difference in output, particularly when the literal search string does not exist but a pattern match does.

ARABPSYCHOLOGY.COM

C2 =FIND("?rs", A2)

	A	B	C	D
1	**Team**	**=SEARCH("?rs", A2)**	**=FIND("?rs", A2)**	
2	Mavs	#VALUE!	#VALUE!	
3	Spurs	3	#VALUE!	
4	Rockets	#VALUE!	#VALUE!	
5	Kings	#VALUE!	#VALUE!	
6	Warriors	6	#VALUE!	
7	Cavs	#VALUE!	#VALUE!	
8	Lakers	4	#VALUE!	
9	Suns	#VALUE!	#VALUE!	
10	Blazers	5	#VALUE!	
11	Hawks	#VALUE!	#VALUE!	
12				
13				
14				
15				
16				

ARABPSYCHOLOGY.COM

In Column B, the **SEARCH function** successfully interprets the wildcard. For **Spurs**, the pattern "?rs" matches "urs". Since the sequence begins at 'u' (position 3), **SEARCH** returns **3**. For **Clippers**, "ers" matches the pattern, starting at 'e' (position 6), returning **6**. This demonstrates the power of **SEARCH** in pattern recognition, making it indispensable for advanced text manipulation tasks requiring flexible criteria.

Conversely, the **FIND function** in Column C treats "?rs" as a literal string consisting of a question mark, followed by an 'r', followed by an 's'. Because none of the team names contain this exact sequence, **FIND** fails to locate a match in any row, returning the error value **#VALUE!** consistently. This confirms that **FIND** is limited exclusively to literal string searches.

Summary of Use Cases and Best Practices

The choice between **SEARCH** and **FIND** is fundamentally a decision between flexibility and precision. When the goal is to perform a general, non-specific inquiry across varying text formats, the **SEARCH** function is the appropriate tool. Its case-insensitivity and support for **wildcard characters** allow users to locate substrings efficiently, regardless of inconsistent capitalization, which is common in user-entered data fields.

Conversely, when data integrity or strict adherence to specific formatting rules is required, the **FIND** function must be used. Its strict case-sensitive nature ensures that operations dependent on

exact text matching, such as locating predefined codes or verifying data against standardized nomenclature, are executed with maximum accuracy. Utilizing the correct function prevents misleading results and enhances the reliability of Excel data processing workflows.

ARABPSYCHOLOGY.COM