Excel: Use SUBSTITUTE Function with Wildcards

Authored by

Mohammed looti

November 14, 2025

The Limitations of the Native SUBSTITUTE Function in Excel

The **SUBSTITUTE function** in **Excel** is a fundamental and highly effective tool for manipulating text strings. Its primary purpose is to locate every instance of a designated old text within a cell and replace it with a specified new text. This function is instrumental for standardizing data and performing precise text cleaning operations, as it relies on an exact match mechanism. For example, if you need to ensure consistency across a large dataset, using `SUBSTITUTE` to correct a misspelled product name or category is both quick and reliable.

However, the requirement for an **exact match** presents a significant limitation when dealing with dynamic text segments. While advanced users often rely on **wildcard characters**--such as the asterisk (*) to represent any sequence of characters or the question mark (?) for any single character--in conjunction with functions like `VLOOKUP` or `COUNTIF`, the native **SUBSTITUTE function** does not inherently recognize or process these symbols in its arguments. Consequently, if the segment of text you wish to replace varies from row to row, a simple application of `SUBSTITUTE` will prove inadequate for the task.

Overcoming this constraint requires a shift from a single-function approach to a complex, combined formula that simulates the behavior of **wildcard characters**. This advanced method allows **Excel** users to perform dynamic text replacement, targeting variable content situated between known boundary markers or delimiters. The key is to leverage multiple text manipulation functions in concert to precisely isolate the unwanted section and stitch the remaining, desired text back together with the new phrase. This approach unlocks advanced data processing capabilities far exceeding basic text substitution.

Architecting a Wildcard-Mimicking Substitution Formula

Given the inability of the **SUBSTITUTE function** to natively support **wildcard characters**, we must construct a formula that achieves the same result: replacing a variable segment of text defined by surrounding known strings. This technique is particularly valuable for data standardization, where you need to update commentary or descriptors that follow a consistent structure but contain non-standardized middle content. Our strategy involves breaking the original string into three parts--the prefix, the new text, and the suffix--and then concatenating them.

To illustrate this powerful concept, consider a scenario where we need to replace all text found between the anchor strings "Mavs" and "team" with the standardized phrase "are an awesome." The following formula encapsulates the necessary logic, utilizing core **Excel** functions to perform this complex, boundary-defined substitution:

```
=IFERROR(LEFT(A2,FIND("Mavs",A2)+LEN("Mavs"))&"are an awesome "&RIGHT(A2,LEN(A2)-FIND("team",A2)+LEN("team")-3), A2)
```

This robust expression relies on the combination of **LEFT**, **RIGHT**, **FIND**, and **LEN functions**. The entire formula is wrapped in the **IFERROR function** to ensure that if the anchor text is missing, the formula gracefully handles the error and returns the original cell content (e.g., cell **A2**). Understanding how each component precisely calculates the start and end points of the desired text segments is key to mastering this dynamic manipulation technique.

Detailed Breakdown of Core Functionality

To effectively apply this technique, a deep understanding of how each function contributes to the overall text extraction and recombination process is essential. The formula works by determining where the replacement should start and end, extracting the text before and after those points, and discarding the variable content in between.

The **FIND function** is the locator. It searches for a specified substring within a given text string and returns the starting numerical position of that substring. For instance, `FIND("Mavs", A2)` provides the starting position of "Mavs" in cell **A2**. This function is case-sensitive, which means "Mavs" is treated differently from "mavs"--an important detail for data integrity. We use `FIND` to establish the exact boundaries for our replacement.

The **LEN function** calculates the total number of characters in a text string. We apply it to our anchor strings, such as `LEN("Mavs")`, to determine their length. This length is vital because it allows us to calculate not just the start of an anchor but the character position immediately following it, ensuring we extract the text segment up to and including the first anchor word ("Mavs").

The **LEFT function** handles the extraction of the prefix. The expression `LEFT(A2, FIND("Mavs", A2) + LEN("Mavs"))` is designed to extract characters from the left side of cell **A2**, precisely up to the last character of the first anchor string ("Mavs"). This segment forms the beginning of our new, modified string.

The **RIGHT function** extracts the suffix. Its calculation, `RIGHT(A2, LEN(A2) - FIND("team", A2) + LEN("team") - 3)`, is more intricate. It calculates the total length of the string, subtracts the position of the second anchor ("team"), and then makes adjustments (like the `-3` in this specific example, which must be tailored based on the length of the replaced text) to ensure the extraction starts exactly where the replacement segment ends, thereby capturing only the desired trailing text.

By using the concatenation operator (`&`), the formula effectively joins the extracted prefix, the new replacement text ("are an awesome "), and the extracted suffix, resulting in a single, dynamically modified string. This sophisticated process simulates the functionality of a **wildcard character** without requiring native support.

Practical Application and Implementation Steps

To solidify the understanding of this powerful technique, let's examine its application within a typical **Excel** data cleaning context. This formula is invaluable when dealing with large datasets where consistency is paramount, but source data contains variability between key identifying markers. Our goal remains the replacement of variable text between "Mavs" and "team" with the standardized phrase "are an awesome."

Consider a column of data where the text segment varies, as shown in the image below. The content between "Mavs" and "team" might be "are an average NBA" or "are a solid playoff," which prevents the use of the simple **SUBSTITUTE function** for a batch operation.

	A	B	C	D
1	**Phrase**			
2	The Mavs are a good team			
3	The Lakers are a great team			
4	The Mavs are a solid team			
5	The Warriors are a decent team			
6	The Blazers are not a good team			
7	The Mavs are a nice team			
8	The Jazz are a bad team			
9	The Mavs are a mediocre team			
10	The Celtics are a consistent team			
11				
12				
13				
14				
15				
16				
17				

The implementation involves entering the composite formula into a new column. Assuming the original text resides in column A, we will input the formula into cell **B2**. This calculation identifies the position of the anchor words, isolates the surrounding text, inserts the new phrase, and produces the transformed string.

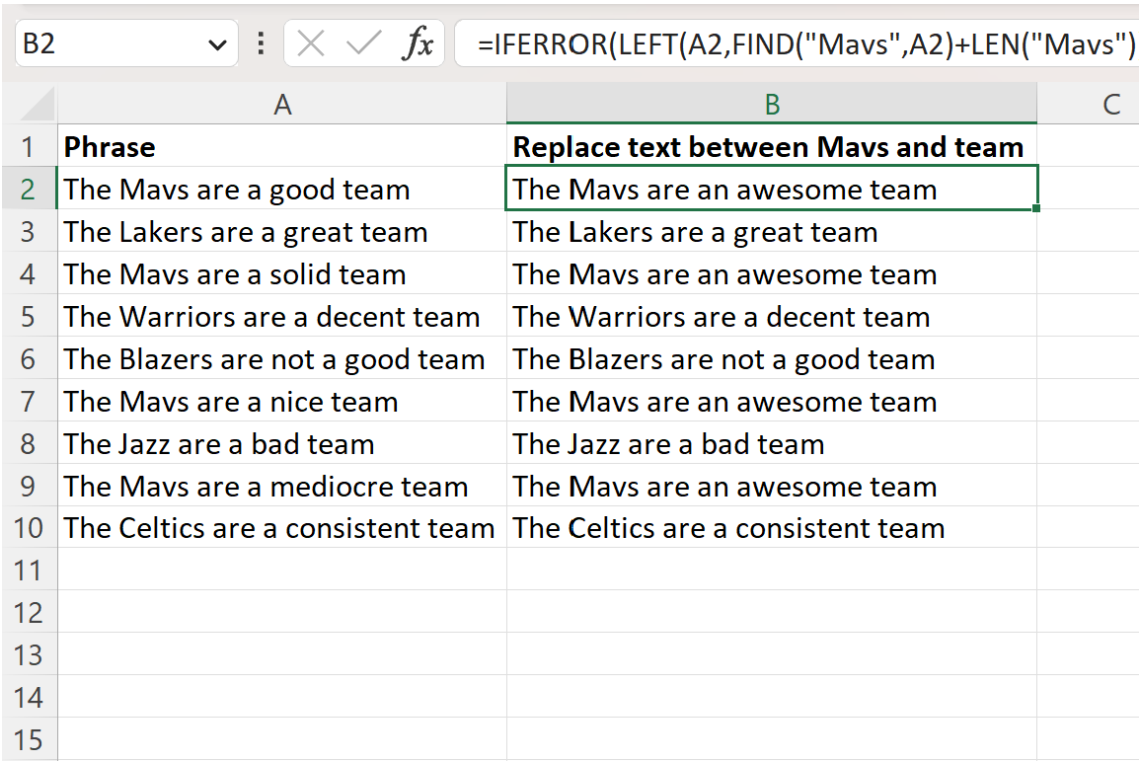
The formula to be entered into cell **B2** is:

=IFERROR(LEFT(A2,FIND("Mavs",A2)+LEN("Mavs"))&"are an awesome "&RIGHT(A2,LEN(A2)-FIND("team",A2)+LEN("team")-3), A2)

After confirming the formula in **B2**, the final step is to apply it across the entire dataset using **Excel's Fill Handle**. By clicking and dragging the small square at the bottom-right corner of cell **B2** down, the formula is copied to the subsequent rows, automatically adjusting the cell references (A2 becomes A3, A4, etc.). This ensures that the dynamic substitution logic is uniformly applied to every phrase in the list, yielding a consistent output.

Verifying Results and Ensuring Formula Robustness

Once the formula has been successfully applied to the entire column, the transformation becomes immediately evident. The output column (B) will contain standardized phrases where the variable text between "Mavs" and "team" has been uniformly replaced with "are an awesome." This visual confirmation validates the effectiveness of the complex formula in achieving dynamic, **wildcard character**-like substitution based on defined text boundaries.



	A	B	C
1	**Phrase**	**Replace text between Mavs and team**	
2	The Mavs are a good team	The Mavs are an awesome team	
3	The Lakers are a great team	The Lakers are a great team	
4	The Mavs are a solid team	The Mavs are an awesome team	
5	The Warriors are a decent team	The Warriors are a decent team	
6	The Blazers are not a good team	The Blazers are not a good team	
7	The Mavs are a nice team	The Mavs are an awesome team	
8	The Jazz are a bad team	The Jazz are a bad team	
9	The Mavs are a mediocre team	The Mavs are an awesome team	
10	The Celtics are a consistent team	The Celtics are a consistent team	
11			
12			
13			
14			
15			

As shown in the image above, the resulting text in column B is entirely consistent, demonstrating the successful manipulation of text segments that were previously non-uniform. This technique is significantly more flexible than relying on the conventional **SUBSTITUTE function**, which would have required multiple, specific formulas for each unique variation of the text to be replaced.

A key element guaranteeing the reliability of this solution is the wrapping **IFERROR function**. The internal logic--relying on **FIND** to locate anchor strings--is inherently fragile if the anchor strings are missing. If "Mavs" or "team" cannot be found in a cell, the `FIND` function returns a **#VALUE!**

error, which would halt the calculation. By using `IFERROR(..., A2)`, we ensure that if any part of the complex substitution fails, the original content of cell **A2** is returned instead. This prevents disruptive error messages from cluttering the output column, making the formula exceptionally robust for real-world datasets that may contain inconsistencies.

Conclusion: Mastering Advanced Excel Text Manipulation

Although **Excel's** **SUBSTITUTE function** is limited to exact-match replacement, the power of combined functions provides a sophisticated workaround to simulate **wildcard character** functionality. By skillfully integrating **LEFT**, **RIGHT**, **FIND**, and **LEN**, and prioritizing data integrity through the use of the **IFERROR function**, users can execute highly precise and dynamic text transformations.

This method empowers you to handle complex data cleaning and standardization tasks where variable text segments need uniform replacement, defined only by their surrounding context. Mastering this technique significantly elevates your ability to manipulate text strings in **Excel**, ensuring efficiency and accuracy across diverse and large-scale data projects.

Additional Resources