

Excel: How to Use MID Function to End of String

Authored by
stats writer

November 22, 2025

RECOMMENDED CITATION

stats writer (2025). *Excel: How to Use MID Function to End of String*. PSYCHOLOGICAL SCALES. Retrieved from <https://scales.arabpsychology.com/?p=100035>

The MID function in Excel is an indispensable tool for data manipulation, allowing users to precisely isolate and extract a specific segment of a text string of text housed within a cell. While its primary design focuses on extracting characters starting from a given position, it becomes exceptionally powerful when combined with other functions to target the remainder of the string--that is, extracting text from a specified start point all the way to the final character. Mastering this technique is crucial for data cleaning and reporting, particularly when dealing with inconsistent data formats where a common prefix needs to be removed while preserving the dynamic suffix.

Unlike functions such as **LEFT** or **RIGHT**, which are limited to the extremities of the text, the MID function offers positional flexibility. However, achieving extraction to the very end of the string requires a dynamic length argument, which Excel cannot determine inherently. To overcome this limitation, we integrate the MID function with the **LEN** function. This powerful combination allows the formula to automatically calculate the total character count available, ensuring that the extraction process captures every single character from the designated starting position until the very end, regardless of how long the source text might be.

Introduction to Text Extraction in Excel

Data preprocessing frequently involves isolating specific components from compound text fields, a task at which Excel excels through its robust suite of text manipulation functions. While functions like **LEFT** and **RIGHT** serve well for boundary extractions, the **MID** function is designed for internal segmentation. It requires three critical arguments: the source text, the character position where extraction should begin (the `start_num`), and the total number of characters to be extracted (the `num_chars`). This inherent structure makes it perfect for scenarios where the desired text starts consistently at a specific point within the string, but its length varies significantly.

The challenge arises when the segment we wish to extract spans from a specific middle character all the way to the end of the input string. If we simply guess the number of characters to extract, any string longer than our guess will be truncated, and any string shorter will result in an error or incorrect behavior. Therefore, a static definition of the length argument is highly impractical in real-world datasets characterized by variability. This necessitates a dynamic solution that can adapt its length parameter based on the characteristics of the individual cell being processed, thus guaranteeing complete capture of the remainder of the string.

Effective data cleaning using this technique is vital when dealing with standardized identifiers or structured naming conventions, such as product codes that always begin with a fixed two-letter prefix followed by a unique, variable-length identifier. By leveraging the combined power of MID and **LEN**, we automate the process of stripping away the unnecessary prefix and retrieving only the unique segment, thereby streamlining subsequent analysis, sorting, or database imports. This automation ensures both speed and accuracy across thousands of data entries.

Understanding the Core **MID** Function Syntax

To properly utilize the **MID** function for advanced extraction tasks, a solid understanding of its syntax and parameters is required. The function takes the form: `=MID(text, start_num, num_chars)`. The first argument, `text`, is straightforward: it is the cell reference or the direct text string from which you intend to extract data. This is typically a reference like **A2**, **B5**, or any cell containing the source data. The function is case-sensitive only in its output, but its positional counting is invariant, treating spaces and special characters as single characters.

The second argument, `start_num`, defines the exact character position where the extraction process should begin. Crucially, counting in Excel text functions is always based on 1-indexing, meaning the first character is position 1, the second is position 2, and so on. If you need to start extraction from the fourth character, `start_num` must be 4. If the specified `start_num` exceeds the total length of the source text, the MID function will return an empty string, preventing calculation errors but resulting in null output, which is an important consideration for debugging complex formulas.

The third argument, `num_chars`, determines how many characters, starting from the `start_num`, should be returned in the result. In standard Excel usage, this is usually a fixed number, such as 5 if you want to extract five characters. However, when we aim to extract text all the way to the end, specifying a fixed number becomes problematic because the required length changes for every row. This is precisely where the innovative combination of functions comes into play, replacing the static `num_chars` with a dynamic calculation that always guarantees sufficiency, effectively telling the function to extract "the rest of the string."

The **LEN** Function: Determining String Length

To successfully calculate the dynamic number of characters remaining from a specific starting point to the end of the string, we must first ascertain the total length of the source text. This is the sole purpose of the **LEN** function in Excel. The syntax is remarkably simple: `=LEN(text)`. It takes a single argument, which is the cell reference containing the text, and returns an integer representing the total number of characters, including spaces, punctuation, and digits. This returned value serves as the crucial input required to dynamically inform the **MID** function.

The role of the **LEN** function in this combined formula is twofold. First, it provides the maximum possible length of the text. Second, and more importantly for our specific use case, it acts as a robust placeholder for the `num_chars` argument in the MID function. When **LEN(A2)** is used as the length parameter, we are essentially telling the MID function: "Start at position X, and extract characters equal to the maximum possible length of the string." Since it is impossible to extract more characters than are present in the string, Excel automatically extracts only the remaining

characters up to the end.

Consider a practical example: if cell A2 contains "EXCEL TRAINING" (14 characters), `LEN(A2)` returns 14. If we want to start extraction at position 7 (the 'T' in 'TRAINING'), and we use 14 (the result of `LEN(A2)`) as the number of characters to extract, **MID** calculates that it needs to extract 14 characters starting from position 7. Since only 8 characters remain ('TRAINING'), Excel automatically limits the output to these 8 characters. This elegant solution bypasses the need for complex arithmetic like calculating `LEN(A2) - start_num + 1`, simplifying the formula significantly.

Combining **MID** and **LEN** for End-of-String Extraction

The core objective of this technique is to create a generalized formula that extracts all characters from a specified starting point right up to the concluding character of the text. This is achieved by embedding the **LEN** function within the `num_chars` argument of the **MID** function. By using the total length of the string as the extraction length, we ensure that the required length is always greater than or equal to the remaining characters, thus forcing Excel to return everything that follows the starting number.

The resulting composite formula takes the standard structure, but replaces the static length with the dynamic calculator: `=MID(Cell_Reference, Starting_Position, LEN(Cell_Reference))`. For instance, if the source data is in cell **A2** and we wish to start extraction at the third character, the formula becomes:

=MID(A2, 3, LEN(A2))

This formulation is highly efficient because it eliminates the need for manual calculation of the remaining characters. If **A2** contains 15 characters, `LEN(A2)` returns 15. The formula then attempts to extract 15 characters starting from the 3rd position. Since only 13 characters exist from the 3rd position onward, Excel correctly returns those 13 characters. If the next cell, **A3**, contains only 8 characters, `LEN(A3)` returns 8, and the formula correctly returns the 6 characters remaining after the 3rd position. This dynamic adaptation makes the formula scalable across entire datasets with varying input lengths.

It is critical to remember the role of the `start_num` parameter in determining which characters are preserved. If the data needs to be extracted based on a delimiter (like a hyphen or a space) rather than a fixed position, the formula must be further nested using functions like **FIND** or **SEARCH** to dynamically determine the `start_num`. However, for cases where the prefix is of a consistent length, such as removing the first two characters of every entry, fixing the `start_num` to 3 (to start after character 2) is the most direct and reliable approach, as demonstrated in the practical

example below.

Practical Example: Extracting Data from Team Names in Excel

To illustrate the effectiveness of combining the **MID** and **LEN** functions, let us apply this methodology to a real-world dataset. Imagine we possess a list of basketball team names where each entry begins with a two-letter abbreviation followed by the full team name, and we only wish to retain the full name starting from the third character. The dataset might look like the following:

	A	B	C	D	E	F
1	Team					
2	Mavericks					
3	Rockets					
4	Hornets					
5	Pacers					
6	Raptors					
7	Thunder					
8	Pelicans					
9	Nuggets					
10	Timberwolves					
11						
12						
13						
14						
15						
16						
17						
18						
19						
20						
21						

Our goal is to create a derived column that isolates the characters starting precisely from the 3rd position to the absolute end of the string for every entry in Column A. This ensures that the two-letter prefix is cleanly removed, regardless of how long the remaining team name is. This task is impossible to achieve efficiently using only the **RIGHT** function, as **RIGHT** requires knowing the total count of characters to keep, which varies dramatically across the list.

We implement the standardized formula detailed previously. Since the data starts in cell **A2**, and we want to begin extraction at the 3rd character, our formula should reference **A2**, set the start position to 3, and use **LEN(A2)** for the length argument. The formula entered into cell **B2** is

therefore:

=MID(A2, 3, LEN(A2))

Upon entering this formula in cell **B2** and dragging it down the column, Excel dynamically applies the logic to each corresponding cell in Column A. The LEN component recalculates the total length for each unique team name, ensuring that the MID function is instructed to extract sufficient characters to reach the end of that specific string. This eliminates manual adjustments and ensures data integrity across the entire dataset.

Step-by-Step Implementation Guide

Implementing the combined MID and **LEN** function formula is straightforward once the logic is understood. Following this guide ensures correct application in your spreadsheet:

Identify the Source Data and Starting Point: Determine the cell containing the text string (e.g., **A2**) and the exact character position where extraction must begin (e.g., position 3).

Select the Destination Cell: Click on the cell where you want the resulting extracted text to appear (e.g., **B2**).

Construct the Formula: Begin by typing `=MID(`.

Input the Text Argument: Reference the source cell (e.g., **A2**). The formula now reads: `=MID(A2,`.

Input the Start Position: Enter the numerical starting position (e.g., 3). The formula now reads: `=MID(A2, 3,`.

Input the Dynamic Length Argument: Incorporate the **LEN** function by typing `LEN(A2)`. This tells Excel to use the maximum length of the string as the number of characters to extract.

Finalize and Execute: Close the parenthesis `)` and press Enter. The complete formula should be `=MID(A2, 3, LEN(A2))`.

The resulting output in the destination cell will display the desired segment. For the team name example, the formula in cell **B2** yields the extracted name, devoid of the two-letter prefix. The process is then completed by utilizing the fill handle--the small square at the bottom right corner of the active cell--to drag the formula down to the subsequent rows. This action automatically adjusts the cell references (e.g., from **A2** to **A3**, **A4**, etc.) ensuring that the dynamic extraction logic is applied consistently across the entire column.

Visually, the result of applying this formula to the basketball team data clearly shows how Column **B** successfully removes the leading abbreviations while preserving the remainder of the text string perfectly, regardless of whether the team name is "Nuggets" (short) or "Timberwolves" (long). The following image provides a clear visualization of the successful extraction:

B2 ✕ ✓ fx =MID(A2, 3, LEN(A2))					
	A	B	C	D	E
1	Team	3rd Character to Last			
2	Mavericks	vericks			
3	Rockets	ckets			
4	Hornets	rnets			
5	Pacers	cers			
6	Raptors	ptors			
7	Thunder	under			
8	Pelicans	licans			
9	Nuggets	ggets			
10	Timberwolves	mberwolves			
11					
12					
13					
14					
15					
16					
17					
18					
19					
20					

Observe that Column B now accurately reflects the text segment ranging from the 3rd character up to the end of the text in Column A. This technique provides a robust, adaptable, and far more reliable method for text manipulation than attempting to calculate the exact remaining length manually.

Benefits and Advanced Considerations

The primary benefit of using the **MID(..., LEN(...))** combination is its unparalleled flexibility and dynamic behavior. It eliminates the need for complex nested IF statements or lengthy arithmetic to determine remaining string length. This combination is particularly beneficial when importing heterogeneous data from external sources where the prefix structure is fixed but the suffix length varies widely, such as file paths, standardized IDs, or customer records.

While the focus here is on extracting from a fixed starting point, it is worth noting that for extractions based on identifying specific delimiters, the `start_num` argument itself must become dynamic. For example, if you wanted to extract text after the first space in a phrase, you would

replace the static '3' with an embedded **FIND** function: `=MID(A2, FIND(" ", A2) + 1, LEN(A2))`. Here, the **FIND** function dynamically calculates the position of the first space and adds one, ensuring that the extraction always starts immediately after the delimiter, providing a truly adaptable solution for complex data parsing tasks.

Finally, while the combination of MID and LEN is the gold standard for fixed-start, variable-end extraction, modern versions of Excel (Microsoft 365) offer newer functions like **TEXTAFTER** and **TEXTBEFORE**. These functions simplify text extraction considerably and may be preferred by users with access to the latest software versions. However, the MID/LEN method remains crucial for compatibility with older software installations and for understanding the foundational logic behind text array manipulation in spreadsheet environments. Understanding this technique solidifies one's expertise in robust data preparation.

Excel: A Formula for MID From Right

Excel: How to Use MID Function for Variable Length Strings