How to Extract Text Between Commas in Excel Using TEXTBEFORE and TEXTAFTER Functions

Authored by

Mohammed looti

November 9, 2025

In the landscape of modern data processing and analysis, the ability to accurately parse and isolate specific components from complex text strings is paramount. When utilizing Microsoft Excel, particularly with structured data that relies on consistent punctuation for value separation, extracting text situated between two identical characters--such as two commas--is a frequent requirement. This guide focuses on harnessing the efficiency of the dedicated text manipulation functions, **TEXTBEFORE** and **TEXTAFTER**, offering a vastly simplified alternative to the traditional, error-prone nesting of **MID**, **FIND**, and **LEN** functions.

These powerful, dedicated functions are now standard in modern versions of Excel (specifically Microsoft 365 and Excel 2021+), allowing users to cleanly and rapidly extract any substring located between a specified pair of delimiters. This capability significantly streamlines the data cleansing process, enabling analysts to focus quickly on the relevant data component.

The foundational syntax required to execute this targeted extraction utilizes a concise nested structure, combining the two functions to define both the start and end points of the desired text:

=TEXTBEFORE(TEXTAFTER(A2, ","), ",")

This construction is specifically engineered to target and extract the substring positioned between the first and second occurrences of the comma within the specified cell reference, here exemplified by cell **A2**. The result is a refined data point, immediately suitable for subsequent calculations, reporting, or database integration. The following sections provide a detailed walkthrough of this formula's application using a typical data normalization scenario.

Practical Application: Isolating Substrings Using Sequential Delimiters

Raw data frequently arrives consolidated, where multiple attributes are combined into a single cell, separated by a consistent character known as a delimiter. A common example involves geographical data, where the city, state, and country are concatenated and separated by commas. Our objective in this scenario is to demonstrate how to efficiently extract only the middle component--the state name--from these combined strings.

Consider the following representative dataset presented in Column A of an Excel sheet:

	A	B	C	D
1	**Location**			
2	Toledo, Ohio, United States			
3	Miami, Florida, United States			
4	Augusta, Maine, United States			
5	Seattle, Washington, United States			
6	Oakland, California, United States			
7	Phoenix, Arizona, United States			
8	Portland, Oregon, United States			
9				
10				
11				
12				
13				
14				
15				
16				
17				

Our primary task is to dynamically isolate the state name, which is consistently situated between the first comma (separating the city) and the second comma (separating the country). Crucially, the formula must be robust enough to handle the varying lengths of the preceding city names, ensuring accurate extraction regardless of the string's initial complexity.

Specifically, our desired extraction results for the initial rows are:

The state name **Ohio** must be extracted from the first text string.
The state name **Florida** must be extracted from the second text string.
The state name **Maine** must be extracted from the third text string.

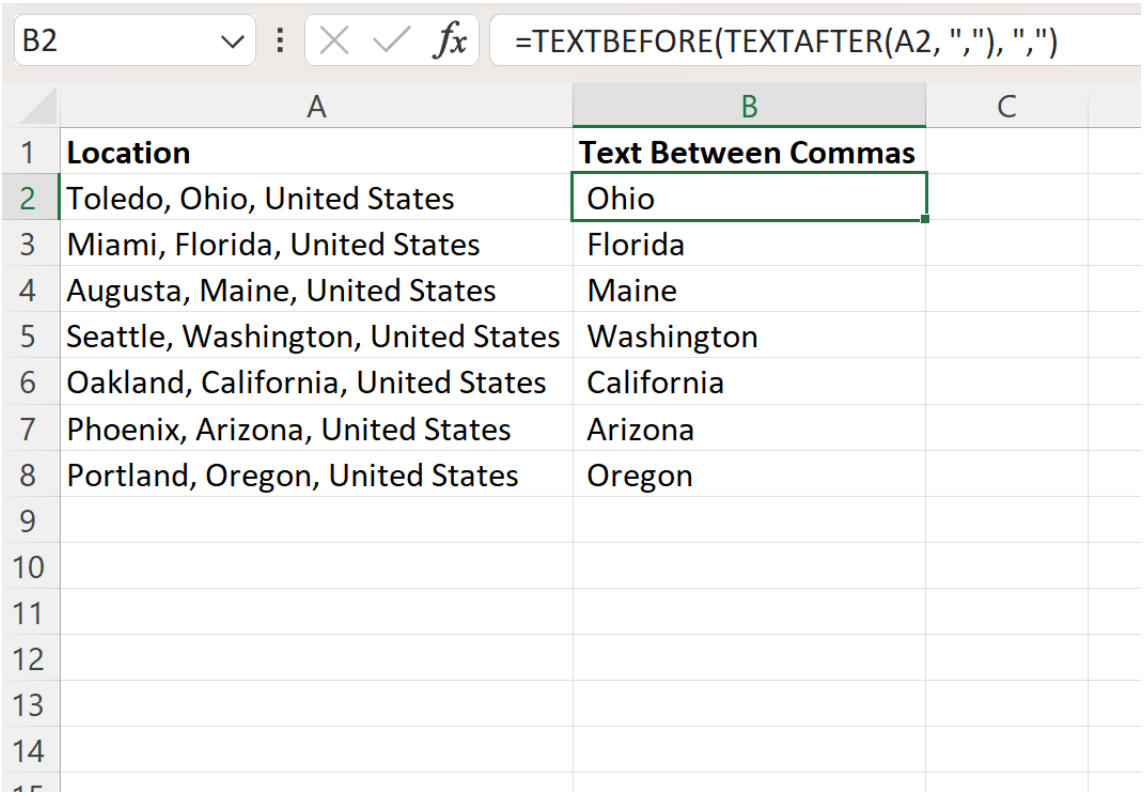
Implementing the Nested TEXTBEFORE and TEXTAFTER Formula

To initiate the precise extraction process, the core nested formula must be entered into cell **B2**, targeting the source data located in cell **A2**. This nested structure is engineered to perform two critical text manipulation steps in sequence, ensuring that the output from the first step feeds directly into the second, achieving the precise isolation required.

=TEXTBEFORE(TEXTAFTER(A2, ","), ",")

Once the formula is correctly established in **B2**, it can be rapidly deployed across the entire dataset. This is accomplished using the Excel fill handle, where the formula is dragged down Column B. This action automatically adjusts the relative cell references (A2 becomes A3, A4, etc.), ensuring the extraction logic is applied consistently and correctly to every subsequent row in the dataset.

The result of deploying this formula demonstrates the exceptional efficiency of the combined **TEXTBEFORE** and **TEXTAFTER** functions when managing consistently delimited data:



As clearly illustrated above, Column B now precisely contains the state names, having successfully isolated the text segment located between the first and second commas in each corresponding location string within Column A. This technique provides a dynamic and non-destructive method for data separation.

Deconstructing the Logic of the Nested Extraction Formula

A thorough understanding of how this nested function operates is vital for adapting it to various text manipulation challenges. The formula executes strictly from the inside out, utilizing the result of the inner function as the primary text input for the outer function. Let us analyze the execution steps for the source data in cell **A2**, which contains "Cleveland, Ohio, United States".

The core efficiency of this method stems from the complementary roles of the two functions. The inner TEXTAFTER function effectively discards the initial component (the city name) and retains everything following the first delimiter. Conversely, the outer TEXTBEFORE function then operates on this shortened string, isolating and returning only the segment that precedes the next comma.

Below is a detailed, sequential breakdown of the formula's execution flow:

=TEXTBEFORE(TEXTAFTER(A2, ","), ",")

Inner Function Execution (TEXTAFTER): The function `TEXTAFTER(A2, ",")` is evaluated first. For the text in **A2** ("Cleveland, Ohio, United States"), this command instructs Excel to return all text that appears subsequent to the first comma.
Intermediate Result: The immediate output generated by the inner function is the remaining text string: **" Ohio, United States"**. It is important to note that any spaces immediately following the initial comma are included in this intermediate result.
Outer Function Execution (TEXTBEFORE): This intermediate string is now passed as the input to the outer function: `TEXTBEFORE(" Ohio, United States", ",")`. This command instructs Excel to take this new, shorter text and extract the segment that precedes the next occurrence of the comma.
Final Output: The calculated result is **" Ohio"**, which successfully isolates the desired state name, although it retains the leading whitespace.

Enhancing Reliability: Incorporating the TRIM Function for Data Hygiene

In text extraction operations, especially when integrating data from disparate sources, managing extraneous whitespace is a critical step. As demonstrated in the previous section, the output from text functions frequently includes leading or trailing spaces if the original data format included a space immediately after the delimiter (e.g., ", Ohio"). These unwanted spaces--even if subtle--can severely compromise subsequent data operations, leading to failed lookups, incorrect comparisons, or skewed sorting results.

To guarantee that the extracted data is perfectly standardized and ready for immediate analytical use, it is highly recommended practice to enclose the entire nested structure within the robust TRIM function. The **TRIM** function is specifically engineered to eliminate all excess spaces from a text string, including any leading or trailing characters, while strictly preserving only single spaces located between actual words.

The optimized, production-ready formula, incorporating **TRIM** to ensure optimal data hygiene, is presented as follows:

=TRIM(TEXTBEFORE(TEXTAFTER(A2, ","), ","))

By implementing this essential modification, the resulting output for the cell A2 example will be precisely **"Ohio"**, completely devoid of any potentially disruptive whitespace. This step is not merely optional; it is fundamental for maintaining data integrity and ensuring consistent, reliable results across complex spreadsheet environments.

Compatibility Considerations and Historical Alternatives

While the combination of **TEXTBEFORE** and **TEXTAFTER** provides the most elegant and user-friendly solution for modern Excel users, it is crucial to recognize that these functions are recent additions to the application's library. Users relying on older versions of Excel (prior to 2021) or those without an active Microsoft 365 subscription will not have access to these features and must employ alternative, more complex methodologies to achieve identical results.

Historically, the extraction of text segments based on delimiters necessitated the arduous nesting of the **MID** function alongside multiple instances of the **FIND** function. This traditional approach required calculating the exact starting position and the precise length of the desired substring, making the resulting formulas significantly more cumbersome, difficult to debug, and inherently susceptible to errors if the data structure or number of delimiters varied. For illustrative purposes, the complex legacy formula required for this exact task would look similar to: `=MID(A2, FIND(",", A2) + 1, FIND(",", A2, FIND(",", A2) + 1) - (FIND(",", A2) + 1))`. This stark contrast highlights the significant productivity leap offered by the specialized text functions.

For users constrained by non-compatible versions, or when processing only a few isolated data points, alternative non-formulaic techniques remain viable. These include the automated pattern recognition of **Flash Fill** or the structured data separation capabilities of the traditional **Text to Columns** feature, typically utilizing the "Delimited" option. However, these methods lack the dynamic, automated updating capability inherent in a dedicated cell formula. The introduction of **TEXTBEFORE** and **TEXTAFTER** truly marks a revolutionary shift in how Excel handles text strings, greatly simplifying advanced data preparation tasks.

Resources for Advanced Text Manipulation and Data Preparation

Mastering the art of text extraction and manipulation is an indispensable skill for any power user of Excel. The efficient methods discussed in this guide, centered around sequential delimiters, can be readily adapted to parse virtually any structured dataset by simply adjusting the specified delimiter (e.g., using pipes, semicolons, or dashes instead of commas).

To further enhance your data cleaning and processing capabilities, the following related tutorials and documentation resources cover other essential text manipulation techniques:

Detailed tutorial on maximizing the effectiveness of the TRIM function for general data cleaning.

A comprehensive guide to extracting text based on fixed character counts using the **LEFT** and **RIGHT** functions.
Advanced techniques focusing on conditional extraction and dynamic position finding using the **FIND** and **SEARCH** functions.