How to Convert a Table to a List in Google Sheets

Authored by

stats writer

January 16, 2026

The ability to convert two-dimensional tabular data into a single, vertical list is a fundamental requirement in data analysis and preparation. This transformation--often referred to as unpivoting or stacking--is crucial when preparing data for specific reports, certain functions that only accept single-column input, or integration with external tools.

While various spreadsheet programs offer complex tools for this process, Google Sheets provides several elegant solutions. This guide details both the traditional, manual approach and the modern, highly efficient method utilizing the dedicated **FLATTEN** function, ensuring your **data structure** is optimized for subsequent analysis.

Understanding how to efficiently manipulate the presentation of your data, especially moving from a wide format (table) to a long format (list), is a key skill for any power user seeking to streamline their workflow and enhance the integrity of their reports. We will explore how to achieve this transformation with precision and minimal effort.

The Manual Conversion Technique: Filtering and Sorting

Before advanced functions like **FLATTEN** were widely utilized, data analysts relied on a multi-step manual process to transform a data table into a list. This method, while effective for small datasets, is time-consuming and prone to errors when dealing with extensive or frequently updated information. However, it remains a valid method for quick, single-use conversions.

To manually convert a table to a list in **Google Sheets**, the first step involves selecting the entire range of data. Once selected, navigate to the Data tab in the menu ribbon and click on the "Create a filter" option. This action immediately applies filter iconography to each column header within the selected range, granting access to powerful organizational tools.

Next, you must utilize the sorting capabilities provided by the filter. Click on the arrow next to the relevant column header--typically the header containing the key identifier or the first column you wish to stack--and select "Sort A-Z" (or "Sort Z-A"). This sorts the data based on the criteria of that specific column. The essential final step involves carefully copying the resulting sorted columns and pasting them into a new sheet or location. When pasting, the data, previously arranged in a matrix, is often condensed into a sequential list format, depending on the arrangement of the copied columns.

While the manual filter and sort method offers immediate results without requiring complex formulas, it lacks scalability and repeatability. If the source data changes, the entire process must be repeated, making it inefficient for dynamic dashboards or continuous reporting. For robust, automated **data restructuring**, a formula-based approach is vastly superior.

Introducing the Powerful FLATTEN Function

The **FLATTEN** function represents a significant advancement in data manipulation within **Google Sheets**. It is designed explicitly for the purpose of taking multiple ranges or arrays of data--including a typical data table--and stacking them into a single, contiguous vertical list. This function eliminates the need for complex array formulas or iterative copy-pasting, automating the unpivoting process entirely.

When implementing **FLATTEN**, the function reads the input ranges sequentially, moving horizontally across each row of the first specified range before progressing to the next row, and finally moving onto subsequent ranges if multiple are provided. This ensures a logical and predictable sequence in the resulting list, maintaining the original order of the data as it appeared in the source table.

The primary advantage of using the FLATTEN function is its dynamic nature. Since it is a formula, if the original source table is updated, the resulting vertical list automatically recalculates and updates instantly. This live connection ensures that any analysis or downstream calculations based on the stacked list are always working with the most current information, drastically improving data integrity and efficiency.

Syntax and Application of the FLATTEN Formula

The syntax for the **FLATTEN** function is deceptively simple, requiring only the input ranges as arguments. This simplicity contrasts sharply with older methods that required chaining functions like **TRANSPOSE** and **ARRAYFORMULA** to achieve similar results.

For example, if you have a data table spanning from cell `B2` to `E4`, representing four columns across three rows, the formula required to convert this range into a vertical list is straightforward:

=FLATTEN(B2:E4)

This single line of code instructs **Google Sheets** to process the two-dimensional **array** specified by the range `B2:E4` and output all values contained within it into a single column. The function automatically handles the expansion of the results into the required number of rows below the cell where the formula is entered, provided there is enough space.

The following example provides a detailed walkthrough of how to use this powerful function in a common business scenario involving quarterly sales data. This illustrates the elegance and efficiency of **FLATTEN** compared to manual manipulation.

Practical Example: Transforming Sales Data

Consider a scenario where a company tracks its total sales figures across several consecutive years, segmenting the data by quarter. This information is typically stored in a wide-format **table**, where each row represents a year and each column represents a specific quarter (Q1, Q2, Q3, Q4).

Suppose we have the following dataset in **Google Sheets** that displays the total sales made by some company during each quarter of three consecutive years, spanning the range B2:E4:

	A	B	C	D	E
1	**Year**	**Q1**	**Q2**	**Q3**	**Q4**
2	**2021**	8	7	4	13
3	**2022**	11	10	5	16
4	**2023**	14	15	8	14
5					
6					
7					
8					
9					
10					
11					
12					
13					
14					

While this wide format is excellent for high-level comparison between years, it is often necessary to analyze the entire sequence of quarterly sales sequentially, perhaps to calculate a running average or to plot a time series graph across all quarters regardless of the year demarcation. For these purposes, we require a single, consolidated vertical list.

The objective is to convert this four-column by three-row table into a single list that contains all twelve sales values stacked one on top of the other, starting with Q1 of the first year and ending with Q4 of the last year.

Implementing the FLATTEN Formula

To achieve the desired vertical list transformation, we need only select an empty cell--ideally cell `A6`, located immediately below our source data--and input the **FLATTEN** formula referencing the

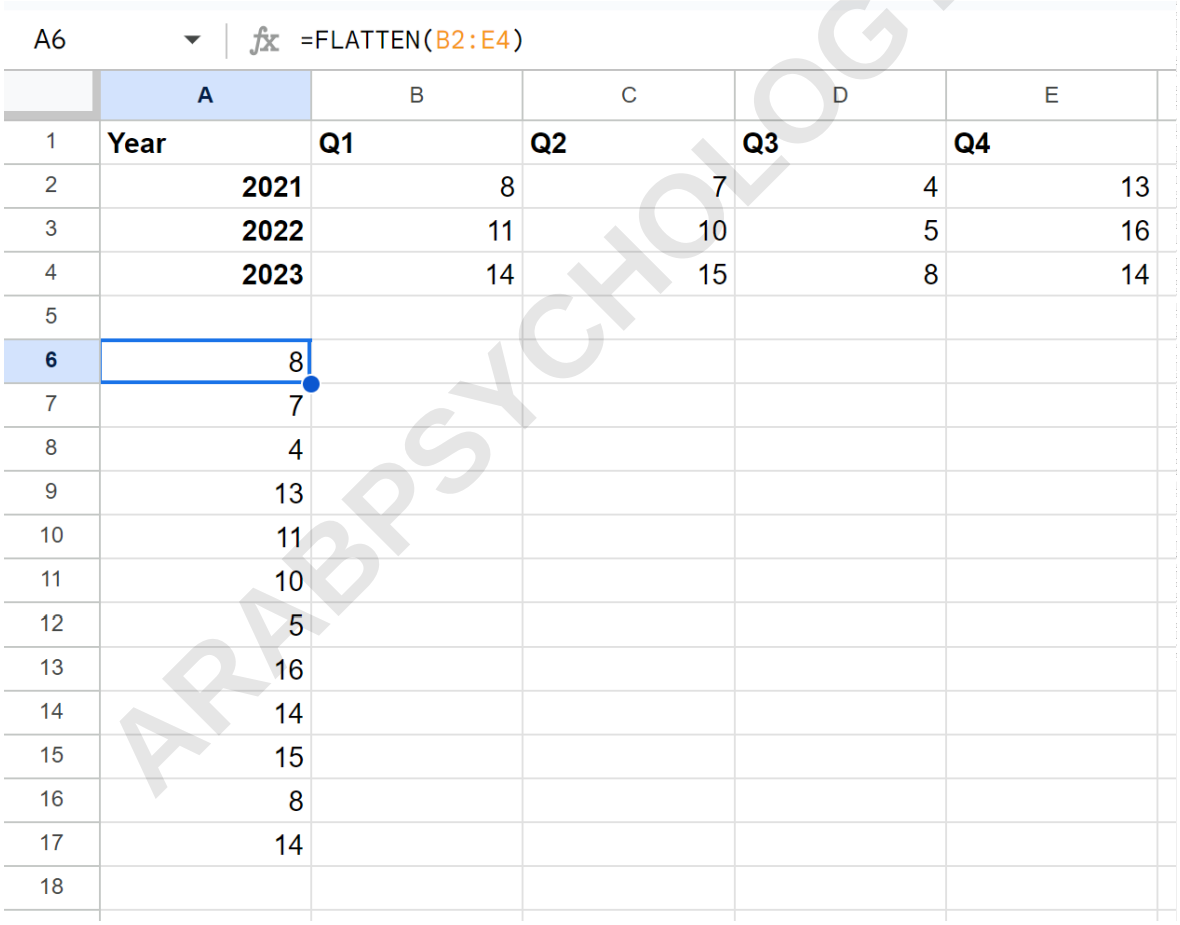
sales data range.

We can type the following formula into cell **A6** to execute the conversion:

=FLATTEN(B2:E4)

Upon execution, the FLATTEN function takes the values from the specified range and structures them into a single column output. The order of output is crucial: the function processes the data row by row, moving from left to right across the columns within that row, before dropping down to the next row and repeating the left-to-right sequence.

The following screenshot shows the result of implementing this formula in cell **A6**, demonstrating the immediate and accurate conversion of the tabular data into the desired list format:



	A	B	C	D	E
1	**Year**	**Q1**	**Q2**	**Q3**	**Q4**
2	**2021**	8	7	4	13
3	**2022**	11	10	5	16
4	**2023**	14	15	8	14
5					
6	8				
7	7				
8	4				
9	13				
10	11				
11	10				
12	5				
13	16				
14	14				
15	15				
16	8				
17	14				
18					

We can see that the formula was able to successfully convert the table of values into a single list that shows the sales values in sequential order across all quarters and years, beginning with the top-left cell of the input range and concluding with the bottom-right cell.

Interpreting the Resulting Vertical List

The resulting vertical list provides the same data as the original table, but restructured to facilitate longitudinal analysis. Understanding the sequence in which the data is stacked is essential for validating the results and ensuring correct interpretation in downstream calculations.

The FLATTEN function prioritizes reading the source **array** row by row. In our example, the function first reads all quarterly sales for the year 2021 (B2 through E2), then proceeds to 2022 (B3 through E3), and finally 2023 (B4 through E4). This row-major order ensures temporal continuity in our sales sequence.

For example, examining the first few values in the resulting list:

The first value in the list shows the sales for **Q1 2021** (from cell B2).
The second value in the list shows the sales for **Q2 2021** (from cell C2).
The third value in the list shows the sales for **Q3 2021** (from cell D2).
The fourth value in the list shows the sales for **Q4 2021** (from cell E2).

The sequence continues in this manner, seamlessly transitioning to the sales for Q1 2022, Q2 2022, and so on, until all twelve values from the original **table** are stacked into a single column. This output format is ideal for pivot tables, charts, or any function requiring sequential input.

Advantages and Limitations of FLATTEN

The introduction of the **FLATTEN** function has revolutionized how users perform data aggregation and **data restructuring** in **Google Sheets**. The advantages over previous methods are numerous, primarily centered around simplicity, speed, and accuracy.

One major advantage is the formula's conciseness. Instead of relying on complex nested functions or iterative scripting, a single, intuitive function call achieves the desired result. This reduces the cognitive load on the user and minimizes the opportunity for syntax errors that often plague lengthy array formulas. Furthermore, because **FLATTEN** is a native function, it processes large datasets significantly faster than manual processes or script-based solutions.

However, it is important to recognize the limitations. **FLATTEN** is designed purely to output values into a single column. It does not automatically generate corresponding labels for the converted data. In our sales example, the resulting list contains sales figures but loses the context of which year and quarter each number belongs to. If you need to retain these identifiers, **FLATTEN** must be used in conjunction with other functions, such as **ARRAYFORMULA**, to generate matching label columns that repeat the Year and Quarter identifiers for every value.

Alternative Techniques for Data Restructuring

While **FLATTEN** is the modern, preferred tool for simple stacking, situations involving complex criteria, conditional stacking, or the need for simultaneous label generation often necessitate alternative techniques utilizing established **Google Sheets** functions. These methods offer greater control but introduce complexity.

One robust alternative involves using the **QUERY** function combined with array literals. This technique allows for highly conditional unpivoting. By manually creating an array literal (using curly braces `{}`) to stack specific columns, and then using **QUERY** to select and arrange them, users can selectively flatten data. For instance, one might want to stack Q1 and Q4 data but exclude Q2 and Q3 data--a task that requires conditional logic beyond the capabilities of a standalone **FLATTEN** function.

Another powerful but more complex method involves creating a custom iterative **ARRAYFORMULA** using **ROW**, **COLUMN**, and **INDEX**. This intricate combination identifies the position of each cell in the source table and calculates its corresponding position in the final list. While offering maximum flexibility, this method is significantly harder to write, debug, and maintain than the simple **FLATTEN** formula, confirming that **FLATTEN** remains the best practice for most standard table-to-list conversions.

Note: You can find detailed documentation explaining the mechanics and variations of the FLATTEN function in Google Sheets online resources.