How to add blank columns in Google Sheets Query

Authored by

stats writer

November 22, 2025

ARABPSYCHOLOGY.COM

RECOMMENDED CITATION

stats writer (2025). *How to add blank columns in Google Sheets Query*. PSYCHOLOGICAL SCALES. Retrieved from https://scales.arabpsychology.com/?p=100098

The ability to manipulate and structure data dynamically is a cornerstone of effective spreadsheet management. When utilizing the powerful Google Sheets Query function, you often need to insert blank spaces or empty columns into your results set, not for data input, but for improved presentation or future calculations. While standard spreadsheet operations allow easy insertion of columns, performing this action directly within the query output requires a specific, sophisticated syntax. This technique allows you to inject visual breaks between columns of existing data, enhancing Data Visualization and making complex reports significantly more readable.

This approach is particularly valuable when preparing a final report or dashboard view where data separation is crucial. By mastering the use of constant string literals combined with the powerful LABEL clause, you gain granular control over the output structure, ensuring that your data is not only accurate but also organized in a professional and intuitive manner. We will explore the precise syntax needed to achieve this formatting goal, starting with the fundamentals of the QUERY function and progressing through examples demonstrating single and multiple blank column insertion.

The Necessity of Blank Columns in Data Organization

While raw data is often stored contiguously (A, B, C, D...), presenting this data to end-users sometimes requires spacing for better comprehension. Inserting a blank column serves several critical purposes in data analysis and reporting. Foremost among these is visual separation. When dealing with metrics that belong to distinct categories--for example, demographic data followed by financial figures--a clear break helps the reader instantly distinguish between data groups without confusion.

Furthermore, blank columns can be utilized as placeholders for future analysis. Although the QUERY function itself is primarily for selection and filtering, the resulting output range might be the starting point for manual entry or subsequent external formulas (like ArrayFormula) applied outside the initial query. Having a predefined empty column ensures that these calculated or manually entered fields align perfectly with the structured output of the query, maintaining the integrity of the Structured Data organization. This method prevents the need to manually insert columns into the destination sheet after every query execution, which is highly inefficient when dealing with dynamic or refreshing data sources.

In essence, incorporating blank columns directly into the query result streamlines the reporting workflow. It transforms a purely technical data extraction process into a user-friendly formatting operation, ensuring that the exported data is ready for immediate consumption or further high-level analysis without intermediate steps. This attention to output structure is a hallmark of expert-level spreadsheet proficiency.

ARABPSYCHOLOGY.COM

Understanding the Google Sheets QUERY Function

The Google Sheets QUERY function is arguably the most powerful tool within the Google ecosystem, allowing users to apply database-like operations--including selecting, filtering, grouping, and aggregating data--using the Google Visualization API Query Language (GQL). The fundamental structure of the function requires two primary arguments: the data range to be queried (e.g., `A1:C12`) and the query string itself (e.g., `"SELECT A, B WHERE C > 10"`).

When constructing the query string, every item listed after `SELECT` must correspond either to an existing column identifier (like `A` or `B`) or to a calculated field or constant value. It is this capability to select a constant value that we exploit to generate a blank column. By selecting a literal string, such as a space enclosed in single quotes (`' '`), the query engine generates a column where every cell in the resulting row contains that constant value. Since a single space is virtually invisible, this effectively creates a blank column.

However, simply selecting `' '` presents an aesthetic challenge: the query output automatically assigns a header to this new column, typically reflecting the literal value or a generic column name (e.g., `' '` or `Col1`). To achieve a truly "blank" look suitable for professional presentation, we must integrate the LABEL clause, which allows us to rename the header of the constant column to an empty string. This combination is the key to mastering blank column insertion.

Core Syntax: Inserting a Single Blank Column

To insert a single blank column between two existing data columns (say, Column A and Column B), we instruct the query to select the first column, then the constant space, and finally the second column. This process is highly precise, requiring meticulous attention to quoting conventions within the query string. The query string is constructed by concatenating the desired column selectors and the constant literal, ensuring the required separation is maintained.

The fundamental syntax for this operation involves three core components within the `SELECT` statement: the first data column, the constant string literal, and the second data column. The literal string, written as `' '` (single quotes enclosing a space), acts as our blank column placeholder. This tells the query engine: "For every row selected, output the value from column A, then output a space, then output the value from column B."

However, as noted, the system will apply a default label. To manage this, the second crucial component, the LABEL clause, must be appended to the query. The LABEL clause takes the constant value or calculated column name (in this case, `' '`) and assigns it a new label. To make the label disappear, we assign it an empty string (`''`).

You can use the following syntax in a Google Sheets query to insert a blank column in the output

ARABPSYCHOLOGY.COM

of the query:

=QUERY(A1:C12, "SELECT A, ' ', B LABEL ' ' "")

This particular query will select column A from the range **A1:C12**, then insert a blank column represented by the literal `' '`, and subsequently select column B from the specified range **A1:C12**. Notice how the entire query string is enclosed in double quotes, while the literal string and the LABEL assignment use single quotes, a convention critical for the Google Visualization API Query Language parser.

Implementing the LABEL Clause for Clean Headers

The proper use of the LABEL clause is indispensable when inserting blank columns. Without it, the output header row would display the constant literal string `' '`, which defeats the purpose of creating a visually clean, separated structure. The LABEL clause syntax requires identifying the column whose label you wish to change, followed by the new label you wish to apply.

In our case, the column we want to relabel is the constant literal we introduced: `' '`. We want to rename its header to an empty string, represented by two single quotes with no space in between: `''`. Therefore, the phrase `LABEL ' ' ''` explicitly instructs the query engine: "Take the column derived from the constant space (`' '`) and rename its header to nothing (`''`)."

This step is critical for maintaining professional output. If you omit the LABEL clause, the output is structurally correct but visually confusing. By combining the constant string insertion in the `SELECT` statement and the corresponding empty label assignment, we achieve a result that mimics a manually inserted column, but with the efficiency and automation provided by the QUERY function.

Note: The **LABEL** clause tells the query to use an empty header for the new blank column. This ensures the output is clean and lacks visible headers above the separator space, preserving the visual intent of the data structure.

The following example shows how to use this syntax in practice.

Example: Inserting a Blank Column in Google Sheets Query

Suppose we have the following dataset that contains information about various basketball players. We want to display the Player Name (Column A) and the Points Scored (Column B), but we need a clear visual break between these two key metrics.

ARABPSYCHOLOGY.COM

	A	B	C	D
1	**Team**	**Points**	**Assists**	
2	Mavericks	22	4	
3	Hawks	24	10	
4	Kings	30	12	
5	Hawks	18	8	
6	Lakers	14	8	
7	Lakers	15	7	
8	Spurs	22	12	
9	Hawks	20	7	
10	Warriors	15	5	
11	Spurs	17	10	
12	Cavs	24	11	
13				
14				
15				
16				
17				

ARABPSYCHOLOGY.COM

The dataset spans the range **A1:C12**, where Column A is the Player, Column B is Points, and Column C is Team. Our objective is to extract A and B, separated by an empty column. We can use the following formula to select column A from the range **A1:C12**, then insert an empty column using the constant literal ' ', and finally select column B. The crucial step is the labeling component to hide the header.

=QUERY(A1:C12, "SELECT A, ' ', B LABEL ' ' """)

The following screenshot demonstrates the practical application of this query formula within a Google Sheet:

A15 =QUERY(A1:C12, "SELECT A, ' ', B LABEL ' ' ''")

	A	B	C	D	E
1	**Team**	**Points**	**Assists**		
2	Mavericks	22	4		
3	Hawks	24	10		
4	Kings	30	12		
5	Hawks	18	8		
6	Lakers	14	8		
7	Lakers	15	7		
8	Spurs	22	12		
9	Hawks	20	7		
10	Warriors	15	5		
11	Spurs	17	10		
12	Cavs	24	11		
13					
14					
15	Team		Points		
16	Mavericks		22		
17	Hawks		24		
18	Kings		30		
19	Hawks		18		
20	Lakers		14		
21	Lakers		15		
22	Spurs		22		
23	Hawks		20		
24	Warriors		15		
25	Spurs		17		
26	Cavs		24		
27					
28					

ARABPSYCHOLOGY.COM

As shown in the output, the query successfully returns the values from column A (Player Name) in the original dataset, followed by a completely empty column, and then the values from column B (Points Scored). Both the data cells and the header cell for the inserted column appear blank, achieving the desired clean separation for enhanced data readability.

Handling Multiple Blank Columns

There may be instances where a single blank column is insufficient for the desired visual separation. For example, you might need two or three empty columns to create a wider buffer

between different sections of your report, perhaps to delineate a significant functional break or to prepare space for future, complex calculations requiring multiple helper columns. Inserting multiple blank columns requires a slight but important modification to the constant string literal technique.

If you simply repeat the literal `' '` (space in single quotes) multiple times in the `SELECT` statement (e.g., `SELECT A, ' ', ' ', B`), the QUERY function engine often optimizes or collapses identical constant columns into a single output column. This behavior is designed to prevent redundant data, but in this specific case, redundancy is exactly what we need. To trick the engine into recognizing each blank column as distinct, we must use unique constant strings.

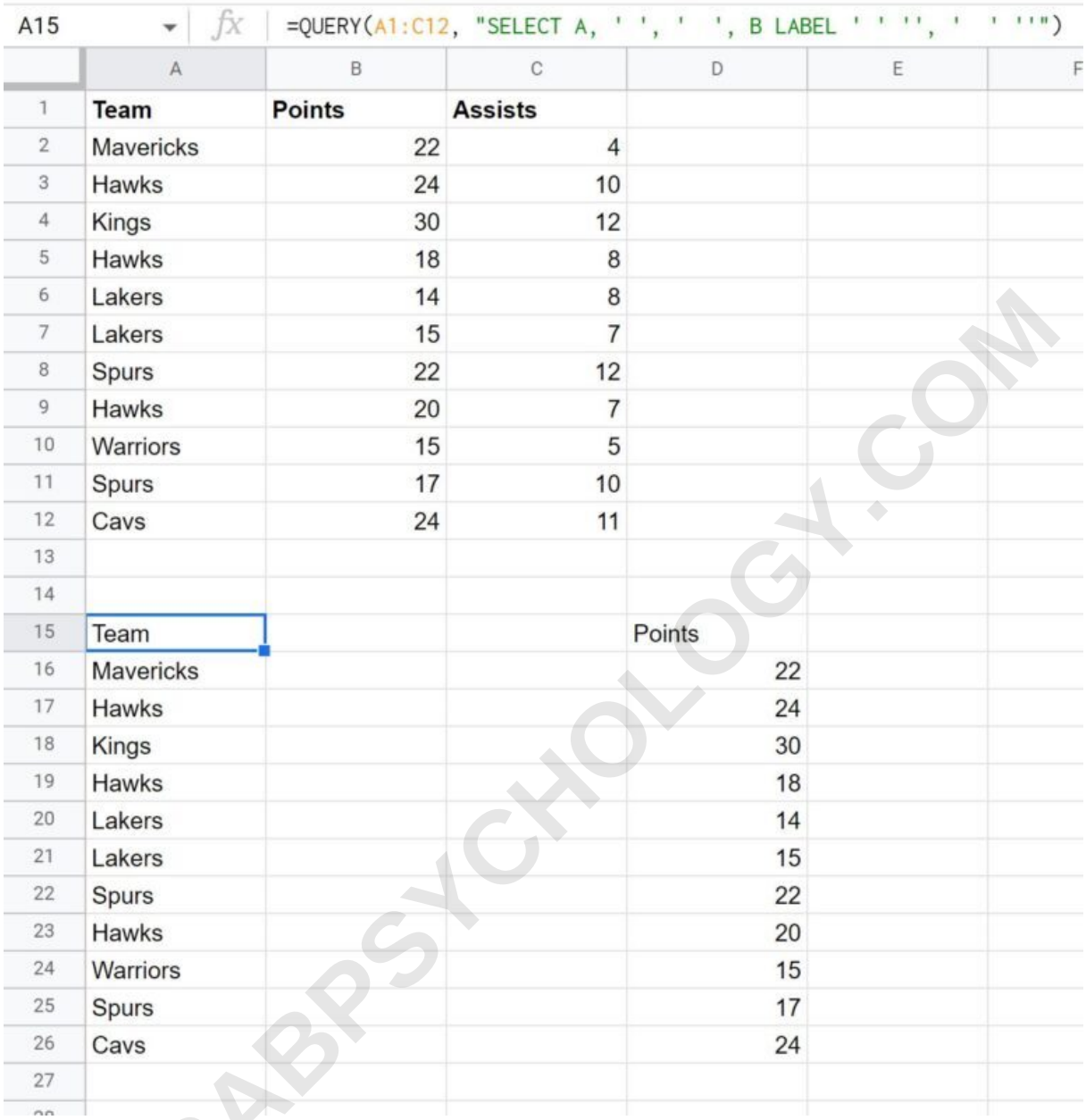
The most effective way to ensure uniqueness is to vary the number of spaces or use different non-visible characters within the single quotes. For instance, the first blank column can be represented by `' '` (one space), the second by `' '` (two spaces), and the third by `' '` (three spaces). Although the visual difference in the cell content is negligible, the query engine treats these as three separate, unique constant columns.

If you'd like to insert multiple blank columns, you must adjust both the `SELECT` clause to include unique constants and the `LABEL` clause to apply an empty label to each unique constant used. The following syntax demonstrates how to insert two distinct blank columns between Column A and Column B:

=QUERY(A1:C12, "SELECT A, ' ', ' ', B LABEL ' ' '', ' ' ''")

The subsequent screenshot shows the result of using this query in practice, successfully creating two separate, side-by-side empty columns for maximum visual separation:

ARABPSYCHOLOGY.COM



	A	B	C	D	E	F
1	**Team**	**Points**	**Assists**			
2	Mavericks	22	4			
3	Hawks	24	10			
4	Kings	30	12			
5	Hawks	18	8			
6	Lakers	14	8			
7	Lakers	15	7			
8	Spurs	22	12			
9	Hawks	20	7			
10	Warriors	15	5			
11	Spurs	17	10			
12	Cavs	24	11			
13						
14						
15	Team			Points		
16	Mavericks			22		
17	Hawks			24		
18	Kings			30		
19	Hawks			18		
20	Lakers			14		
21	Lakers			15		
22	Spurs			22		
23	Hawks			20		
24	Warriors			15		
25	Spurs			17		
26	Cavs			24		
27						

ARABPSYCHOLOGY.COM

Also note that we had to use the **LABEL** clause to specify two separate empty labels for the two new blank columns: `' ' ''` for the column containing one space, and `'  ' ''` for the column containing two spaces. Failing to label both constants will result in one or both new columns displaying unwanted headers.

Advanced Use Cases and Alternatives

While the constant literal method is the most robust way to guarantee blank columns within a query result, it is important to consider scenarios where alternative data manipulation techniques might be more appropriate. For example, if the goal is not just visual separation but the insertion of complex formulas based on the query output, you might leverage the native structure of the

spreadsheet itself.

One alternative technique involves using the ArrayFormula function in conjunction with the QUERY function. If you run your query in one location (e.g., cell E1), you could then use a separate array formula starting in the adjacent column (e.g., cell G1) to pull in calculated fields, leaving Column F blank. However, this approach divorces the structural output from the data extraction, making the sheet less self-contained. The advantage of the `' '` method is that the entire output, including the blank separator, is produced by a single, self-contained formula.

The blank column insertion technique is particularly useful when creating aggregated reports. Imagine a summary query that calculates total sales per region. You might want to display Region, a blank separator, Total Sales, another blank separator, and then the Average Sale Value. By carefully placing the constant literals, you turn a single query operation into a highly formatted, presentation-ready report, suitable for sharing or embedding in external documentation. This flexibility underscores the power of integrating formatting needs directly into the GQL syntax.

Troubleshooting Common QUERY Issues

When implementing advanced QUERY function syntax, users often encounter errors related to quoting and column references. The most frequent issues when inserting blank columns stem from the strict parsing rules of the GQL.

First, ensure that the query string itself is enclosed in **double quotes** (`" "`), while all internal string literals--including the blank column placeholder (`' '`) and the new header (`''`)--are enclosed in **single quotes**. Mixing these up or omitting quotes entirely is the most common cause of the `#ERROR!` return value, often accompanied by a "Formula parse error" message.

Second, when inserting multiple blank columns, failing to use unique constant strings (e.g., repeating `' '` three times) will cause the query engine to collapse them. Always use slight variations like `' '`, `' '`, and `' '`, and ensure that every unique constant used in the `SELECT` statement is explicitly mapped to an empty label in the `LABEL` clause. If you use `' '` and `' '` but only include `LABEL ' ' ''`, the column represented by `' '` will still display its default header.

Finally, always verify that your data range (e.g., `A1:C12`) is correctly specified and that the column identifiers (A, B, C) match the columns within that range. While blank column insertion is independent of the data type, any subsequent filtering or aggregation in the query must adhere to the strict rules of homogeneous data types within columns, a foundational principle of the QUERY function.

ARABPSYCHOLOGY.COM

Summary and Best Practices for Structured Data Output

Adding blank columns into a Google Sheets query output is an elegant solution for improving the structure and readability of dynamically generated reports. By leveraging the power of constant string literals and the precise control offered by the LABEL clause, analysts can customize the visual layout without resorting to manual post-query formatting.

The best practice for this technique involves the following steps:

Identify Separation Points: Determine exactly where in the output stream the visual break is required.

Insert Constant Literals: Use `' '` within the `SELECT` clause to represent the blank column. For multiple blanks, ensure uniqueness by varying the number of internal spaces (e.g., `'  '`, `'   '`).

Mask Headers with LABEL: Immediately follow the `SELECT` statement with the LABEL clause, mapping each constant literal to an empty string (`''`).

Verify Quoting: Ensure all components adhere to the GQL standard: double quotes for the overall query string, and single quotes for internal strings and column literals.

Mastering this technique is essential for anyone generating complex, highly formatted reports using the Google Sheets Query function. It allows for the creation of clean, visually appealing, and highly functional outputs that are optimized for both immediate human consumption and subsequent formulaic analysis.

ARABPSYCHOLOGY.COM