

How to Easily Rank Items by Multiple Columns in Google Sheets

Authored by
stats writer

December 1, 2025

RECOMMENDED CITATION

stats writer (2025). *How to Easily Rank Items by Multiple Columns in Google Sheets*. PSYCHOLOGICAL SCALES. Retrieved from <https://scales.arabpsychology.com/?p=102975>

The process of ranking items within a spreadsheet environment like [Google Sheets](#) often requires more complexity than a simple numerical sequence. While the standard [RANK function](#) excels at ordering data based on a single criterion, real-world data analysis frequently demands sophisticated tie-breaking mechanisms involving secondary or tertiary columns. Mastering multi-column ranking is essential for producing accurate, insightful reports, especially when dealing with metrics that might frequently result in identical values.

To effectively address this need, we must move beyond basic functions and integrate advanced techniques, primarily utilizing a combination of the [RANK function](#) and the powerful [SUMPRODUCT function](#). This approach allows us to establish a primary ranking column (e.g., total score) and then systematically apply a decisive secondary criterion (e.g., time taken or assists) to resolve any ties, ensuring a unique rank for every item in the list. Furthermore, once ranking is complete, you could optionally employ the [FILTER function](#) to isolate specific subsets of data, such as only the top ten ranked entries.

The Advanced Formula for Multi-Column Ranking

Achieving a unique, precise ranking across multiple columns requires a specialized formula structure that mathematically incorporates both the primary rank and the tie-breaking criteria. This is achieved by taking the result of the standard primary ranking and incrementing it based on how many competing entries share the same primary value but have a better score in the secondary column. This ensures that tied entries are correctly sequentialized.

You can use the following comprehensive syntax to rank items by multiple columns in [Google Sheets](#), guaranteeing that ties are resolved efficiently based on the secondary column's values:

```
=RANK(B2,$B$2:$B$11)+SUMPRODUCT(--($B$2:$B$11=$B2),--(C2<$C$2:$C$11))
```

This particular iteration of the formula is designed to produce a [descending order](#) ranking. It first sorts the items primarily by the values found in column B. If two or more entries share the identical value in column B, the formula then utilizes the values in column C to break the tie. In this context, a higher value in column C will result in a better rank (a lower number) among the tied entries.

Deconstructing the Formula Components: RANK and SUMPRODUCT

To fully appreciate the mechanism of this multi-column ranking solution, we must examine the roles of its two main components: the [RANK function](#) and the [SUMPRODUCT function](#). The [RANK function](#) establishes the foundational numerical position, essentially treating all ties equally. For instance, if three players tie for the 5th best score, they would all initially receive a rank of 5.

The critical role of the second part, the SUMPRODUCT function, is to calculate the necessary offset or adjustment needed for the tiebreakers. It accomplishes this by evaluating two separate logical conditions simultaneously. The first part, `--(B2:B11=$B2)`, checks which entries in the primary range (`B2:B11`) are equal to the primary value of the current row (`$B2`). The second part, `--(C2:C11>$C2)` (or `C2<$C$2:$C$11` in the provided example, achieving the same comparison), checks which entries in the secondary range (`C2:C11`) have a better secondary score than the current row's secondary score (`$C2`).

By multiplying the results of these two conditions (which return 1 for true and 0 for false, achieved via the double negative `--` coercion), the SUMPRODUCT function counts precisely how many rows share the same primary rank AND possess a superior secondary rank. This count is then added to the base rank provided by the initial RANK function, effectively incrementing the rank of the current item by the number of tied items that outperformed it on the secondary criterion, thus resolving the tie.

Example: Rank Items by Multiple Columns in Google Sheets

To illustrate the practical implementation of this advanced ranking technique, consider a scenario involving athletic performance data. Suppose we maintain a dataset in Google Sheets detailing the performance statistics for basketball players. We intend to rank these players based primarily on Points, and secondarily on Assists, using the latter metric only to resolve ties in the Points column.

Suppose we have the following dataset in Google Sheets that shows the points and assists for 10 different basketball players:

	A	B	C	D
1	Player	Points	Assists	
2	A	25	8	
3	B	25	5	
4	C	20	4	
5	D	20	7	
6	E	18	9	
7	F	18	9	
8	G	16	4	
9	H	14	3	
10	I	11	4	
11	J	11	3	
12				
13				
14				
15				
16				
17				
18				

Our objective is clear: we want to assign a rank of "1" to the absolute best player and "10" to the absolute worst player. The hierarchy for determining "best" is strictly based on the value in the **Points** column. Crucially, if two players register the same number of points, the tie must be broken by referring to the value in the **Assists** column, where a higher assist count indicates a better rank.

Step-by-Step Implementation of Descending Ranking

To begin the process of ranking, we assume the player data (Points and Assists) resides in columns B and C, respectively, starting from row 2. We will apply the calculated ranking in a new column, typically column D. The formula must be entered into cell D2 and then efficiently copied down through D11 to cover all players in the dataset.

We'll use the following complex formula to rank the players, ensuring the descending sorting order (higher scores are better):

=RANK(B2,\$B\$2:\$B\$11)+SUMPRODUCT(--(\$B\$2:\$B\$11=\$B2),--(C2<\$C\$2:\$C\$11))

It is vital that the ranges used in the formula, specifically `B2:B11` and `C2:C11`, utilize absolute referencing (the dollar signs) so that when the formula is dragged down to subsequent

rows, these reference ranges remain static and correctly encompass the entire evaluation population. Only the single-cell references, such as **B2** and **C2**, should be relative, allowing them to shift to **B3** and **C3**, and so on, for each player evaluated.

The following screenshot demonstrates the practical result of implementing this formula across the entire dataset:

D2 fx `=RANK(B2,B2:B11)+SUMPRODUCT(--(B2:B11=$B2),--(C2<$C$2:$C$11))`

	A	B	C	D	E	F
1	Player	Points	Assists	Ranking		
2	A	25	8	1		
3	B	25	5	2		
4	C	20	4	4		
5	D	20	7	3		
6	E	18	9	5		
7	F	18	9	5		
8	G	16	4	7		
9	H	14	3	8		
10	I	11	4	9		
11	J	11	3	10		
12						
13						
14						
15						
16						
17						
18						

Upon review of the resulting ranks, the player assigned a ranking of 1 is definitively considered the best performing, while the player with a ranking of 10 is considered the worst, according to the dual-criteria ranking logic we established. This unique ranking system provides much greater fidelity than a simple single-column sort.

Analyzing the Results: Handling Tiebreakers

The power of the SUMPRODUCT adjustment becomes evident when analyzing players who shared the same score in the primary column (Points). Without the tie-breaking logic, multiple players would have been assigned the same rank, complicating subsequent analysis or filtering operations.

Here's a detailed breakdown of how our sophisticated formula successfully handled the required

tie-breaking logic:

The primary objective was to rank all players from best to worst, determined by the numerical value in the **Points** column.

The secondary objective, invoked only when the primary values were equal, was to refer to the value in the **Assists** column, assigning a better rank to the player with more assists.

Consider the instance where Player A and Player B registered identical scores in the Points column, initially resulting in a tie for the top ranks. However, since Player A had a higher count of Assists than Player B, the SUMPRODUCT function correctly calculated the necessary adjustment. Consequently, Player A received the highest ranking of **1**, while Player B, despite having the same primary score, was accurately assigned the ranking of **2** due to the lower secondary criterion score. This demonstrates the formula's ability to maintain a truly unique ranking sequence.

Adapting the Formula for Ascending Sorting Order

The formula provided thus far executes a descending order ranking, where higher values are considered superior (1=best). If the required business logic dictates an **ascending** order ranking, meaning that the lowest value is considered the best (e.g., in a race where time is tracked, 1=fastest/lowest time), the formula requires specific modification.

To rank the players in **ascending** order (where 1=worst, and 10=best in this context, assuming higher points are still generally good but we want the ranking sequence inverted), we need to make two key changes. First, the standard RANK function must be modified by including the third optional argument, **1**, to signify ascending order. Second, the entire formula needs an adjustment factor of **+1** at the end to correct for the inherent zero-based counting mechanism introduced by the RANK function in ascending mode when combined with SUMPRODUCT.

The revised formula for ascending ranking (1=worst, high number=best) is as follows:

```
=RANK(B2,$B$2:$B$11, 1)+SUMPRODUCT(--($B$2:$B$11=$B2),--(C2<$C$2:$C$11))+1
```

This modified formula ensures that the primary sorting logic is inverted. Crucially, the tie-breaking logic ($C2 < \$C\$2 : \$C\11) remains unchanged if the goal is still for a higher score in column C to be considered 'better' among tied entries, regardless of the overall sorting order of the primary column B. If the secondary tie-breaker also needs to be inverted (e.g., lower C score is better), the comparison operator would need to be changed accordingly (e.g., $C2 > \$C\$2 : \$C\11).

The following screenshot shows the practical results of using this modified formula, demonstrating the inverted ranking sequence:

D2 *fx* =RANK(B2,\$B\$2:\$B\$11, 1)-SUMPRODUCT(--(\$B\$2:\$B\$11=\$B2),--(C2<\$C\$2:\$C\$11))+1

	A	B	C	D	E	F
1	Player	Points	Assists	Ranking		
2	A	25	8	10		
3	B	25	5	9		
4	C	20	4	7		
5	D	20	7	8		
6	E	18	9	6		
7	F	18	9	6		
8	G	16	4	5		
9	H	14	3	4		
10	I	11	4	2		
11	J	11	3	1		
12						
13						
14						
15						
16						
17						
18						
19						

With this ascending configuration, the "best" player (highest overall performance based on primary and secondary criteria) now correctly receives the highest ranking number, **10**, while the "worst" player is designated with the ranking of **1**. This flexibility ensures that the ranking methodology can be accurately tailored to meet diverse analytical requirements within Google Sheets.