

# How to Easily Filter Data Using the FILTER Function with OR Criteria

Authored by  
**stats writer**

November 21, 2025

## RECOMMENDED CITATION

stats writer (2025). *How to Easily Filter Data Using the FILTER Function with OR Criteria*. PSYCHOLOGICAL SCALES. Retrieved from <https://scales.arabpsychology.com/?p=98717>

The [FILTER function](#) is one of the most powerful tools available within [Google Sheets](#) for data manipulation and extraction. It is specifically designed to extract records from a range of data based on criteria you define. When combined with [OR logic](#), this function becomes exceptionally versatile, allowing you to return rows that match any one of several specified conditions, rather than requiring all conditions to be met (which is the default behavior when using multiple arguments without explicit OR conversion).

Understanding how to implement [OR logic](#) within the [FILTER function](#) is essential for advanced spreadsheet users. While many functions utilize a dedicated `OR()` function, the [FILTER function](#) uses a unique arithmetic approach involving the plus sign (+) to achieve the desired disjunctive criteria. This article provides a comprehensive guide to mastering this technique, ensuring you can efficiently extract complex subsets from your data.

## Understanding the Mechanics of FILTER with OR Logic

When employing the [FILTER function](#), the core requirement is to provide a range to filter and one or more conditions (criteria) that result in an array of `TRUE` or `FALSE` values. When defining multiple criteria, the [FILTER function](#) typically treats them as implicit `AND` statements; that is, a row must satisfy `condition1 AND condition2 AND so on`, to be included in the output. To implement [OR logic](#), where a row is returned if it satisfies `condition1 OR condition2`, we must manipulate these arrays of `TRUE/FALSE` values.

The standard [syntax](#) of the [FILTER function](#) is `FILTER(range_to_filter, condition1, , ...)`. However, when we want to introduce OR conditions, we wrap all criteria within parentheses and use the arithmetic plus sign (+) to join them. This process is crucial because arithmetic operations on Boolean arrays automatically coerce the `TRUE` and `FALSE` values into their numerical equivalents: `TRUE` becomes 1, and `FALSE` becomes 0. By adding these numerical arrays together, we create a single resulting array where any value greater than zero (i.e., 1 or more) signifies that at least one of the criteria was met, thus satisfying the OR requirement.

You can use the following basic [syntax](#) in [Google Sheets](#) to use the **FILTER** function with **OR** logic, where the plus sign acts as the logical OR operator:

```
=FILTER(A1:C10, (A1:A10="A")+(C1:C10<20))
```

This formulation instructs the spreadsheet to return the rows within the range A1:C10 where the value in the array A1:A10 is equal to "A" **or** the corresponding value in C1:C10 is less than 20. The resulting array of 1s and 0s effectively tells the **FILTER** function which rows to keep, based on whether the sum of the conditions for that row is greater than zero.

## The Role of the Plus Sign (+) as the OR Operator

The use of the arithmetic plus sign (+) to represent logical OR is a specific feature of how [Google Sheets](#) and Excel handle array operations. In standard programming contexts, the OR operator might be represented by `||` or the function `OR()`. However, within the criteria argument of the **FILTER** function, we are dealing with arrays of results, not single Boolean values, necessitating this arithmetic coercion.

Consider two conditions, Condition X and Condition Y. When these are evaluated, they produce two distinct arrays of `TRUE/FALSE` values corresponding to each row in the source data. When we use `(Condition X) + (Condition Y)`, the system performs the following transformation for each row:

If Row 1 satisfies X (`TRUE=1`) and satisfies Y (`TRUE=1`), the result is  $1 + 1 = 2$ .

If Row 2 satisfies X (`TRUE=1`) but fails Y (`FALSE=0`), the result is  $1 + 0 = 1$ .

If Row 3 fails X (`FALSE=0`) and fails Y (`FALSE=0`), the result is  $0 + 0 = 0$ .

Any row resulting in a value greater than zero (1 or 2 in this case) indicates that at least one criterion was met, satisfying the definition of **OR logic**. The **FILTER** function then uses this final numerical array to select the corresponding rows from the original range. This method is highly efficient and scalable, making it the preferred technique for complex OR filtering in [Google Sheets](#).

**Note:** The plus (+) sign is used as an OR operator in [Google Sheets](#) when combining Boolean arrays within functions like **FILTER** or **ARRAYFORMULA**. Conversely, using the multiplication sign (\*) would perform implicit AND logic, as only  $1 * 1 = 1$  yields a TRUE result, while any multiplication involving a 0 (FALSE) results in 0.

## Practical Example: Setting Up the Dataset

To demonstrate the efficacy of the **FILTER** function combined with **OR logic**, we will use a hypothetical dataset containing information about basketball players. This example will clearly illustrate how rows meeting either of the criteria are successfully extracted, providing a concise subset of the original data.

Suppose we have the following dataset in [Google Sheets](#) spanning the range A1:C10, which contains information about various basketball players, including their Team affiliation, Position, and Points scored. This structure allows us to easily define criteria against different columns simultaneously.

	A	B	C	D	E
1	<b>Team</b>	<b>Position</b>	<b>Points</b>		
2	A	Guard	15		
3	A	Guard	19		
4	A	Forward	22		
5	A	Forward	29		
6	A	Forward	25		
7	B	Guard	21		
8	B	Guard	30		
9	B	Forward	14		
10	B	Forward	12		
11					
12					
13					
14					
15					
16					
17					
18					

Our goal is to filter this data to identify all players who belong to Team A **or** who scored fewer than 20 points. This requires the use of two distinct conditions applied to two separate columns (A and C), linked by the OR operator (the + sign).

### Implementing the Basic FILTER + OR Formula

When filtering this dataset, we must specify the full range (A1:C10) as the data source, followed by the combined criteria. Our first criterion is checking if the Team column (A1:A10) equals "A". Our second criterion is checking if the Points column (C1:C10) is less than 20. By wrapping these criteria in parentheses and linking them with a plus sign, we enforce the OR condition.

We can use the following formula to filter for all rows where the team is equal to "A" **or** the points scored is less than 20. This formula is entered into a single cell where you want the filtered output to begin (e.g., cell E1):

**=FILTER(A1:C10, (A1:A10="A")+(C1:C10<20))**

The successful execution of this formula relies on array alignment. It is crucial that the ranges used for the criteria (A1:A10 and C1:C10) correspond exactly to the vertical dimensions of the range being filtered (A1:C10). Any mismatch in the number of rows between the data range and the

criteria ranges will result in a #VALUE! error, halting the operation. Furthermore, remember that text comparisons in Google Sheets are often case-sensitive, so ensuring that the criterion "A" matches the casing in the source data is important for accuracy.

## Visualizing the Results and Interpretation

Upon entering the formula, the FILTER function immediately processes the conditions and presents the resulting subset of data. The output table will only contain rows that satisfy at least one of the two defined conditions. This visual confirmation is the best way to verify that the OR logic was correctly applied.

The following screenshot shows how the formula executes in practice, demonstrating the filtered output:

A12    fx    =FILTER(A1:C10, (A1:A10="A")+(C1:C10<20))

	A	B	C	D
1	<b>Team</b>	<b>Position</b>	<b>Points</b>	
2	A	Guard	15	
3	A	Guard	19	
4	A	Forward	22	
5	A	Forward	29	
6	A	Forward	25	
7	B	Guard	21	
8	B	Guard	30	
9	B	Forward	14	
10	B	Forward	12	
11				
12	A	Guard	15	
13	A	Guard	19	
14	A	Forward	22	
15	A	Forward	29	
16	A	Forward	25	
17	B	Forward	14	
18	B	Forward	12	
19				
20				
21				

If you examine the returned rows, you will notice that they include players who scored 30 points (which is NOT less than 20) but belong to Team A (meeting the first criterion), as well as players

who scored 15 points (meeting the second criterion) but belong to Team B. The only rows returned are the ones where the team is equal to "A" **or** the points is less than 20. This clearly demonstrates the inclusive nature of the **OR logic** applied via the plus operator, contrasting sharply with an AND operation which would only return rows where both conditions were simultaneously true.

## Advanced Application: Combining Multiple OR Conditions

The beauty of using the arithmetic method for **OR logic** is its simplicity in scaling. If you need to check for three, four, or even more alternative criteria, you simply continue to add more parenthetical condition arrays using the plus sign (+). This allows for highly complex and nuanced filtering operations without significantly complicating the syntax.

We can also use more plus signs (+) in the **FILTER** function to apply even more **OR** operators. This approach ensures that if any one of the numerous conditions evaluates to **TRUE** (or 1), the corresponding row will be included in the final results, as the sum of the conditions for that row will be greater than zero.

For example, we can use the following syntax to filter for rows where the team is equal to "A" **or** the position is "Guard" **or** points is greater than 15. This introduces a third column (Position, B1:B10) into the criteria evaluation:

**=FILTER(A1:C10, (A1:A10="A")+(B1:B10="Guard")+(C1:C10>15))**

This formulation provides maximum flexibility for extracting players based on very broad criteria. A player only needs to be on Team A, or be categorized as a Guard, or have scored more than 15 points to be included. If a player satisfies all three, the result of the arithmetic sum will be 3, which is still interpreted by **FILTER** as a positive inclusion.

The following screenshot shows how to use this more advanced formula in practice, yielding a larger, more inclusive dataset:

	A	B	C	D	E
A12	=FILTER(A1:C10, (A1:A10="A")+(B1:B10="Guard")+(C1:C10>15))				
1	<b>Team</b>	<b>Position</b>	<b>Points</b>		
2	A	Guard	15		
3	A	Guard	19		
4	A	Forward	22		
5	A	Forward	29		
6	A	Forward	25		
7	B	Guard	21		
8	B	Guard	30		
9	B	Forward	14		
10	B	Forward	12		
11					
12	<b>Team</b>	<b>Position</b>	<b>Points</b>		
13	A	Guard	15		
14	A	Guard	19		
15	A	Forward	22		
16	A	Forward	29		
17	A	Forward	25		
18	B	Guard	21		
19	B	Guard	30		
20					
21					
22					

This **FILTER** function successfully returns the rows in the original dataset where any of the three conditions are met. Mastering this technique is essential for anyone who frequently analyzes large spreadsheets and requires dynamic, multi-criteria data extraction.

## Common Pitfalls and Troubleshooting

While the **FILTER** function with **OR logic** is powerful, it is susceptible to specific errors, primarily related to range dimensions and data types. Ensuring accuracy requires meticulous attention to the structure of the formula:

**Range Mismatch:** The most frequent error occurs when the rows in the criteria arrays (e.g., A1:A10) do not match the number of rows in the data range (e.g., A1:C12). All ranges must align perfectly vertically.

**Missing Parentheses:** Each individual condition within the OR statement must be enclosed in its own set of parentheses (e.g., (A1:A10="A")). Omitting these grouping symbols can cause

misinterpretation by the spreadsheet engine, leading to unexpected arithmetic results or errors.

**Case Sensitivity:** Remember that text comparisons like `= "A"` are often case-sensitive. If you are uncertain about the case in your source data, consider wrapping the comparison range in a function like `LOWER()` or `UPPER()` to standardize the text before comparison, though this adds complexity to the formula.

## Alternative Methods for OR Filtering

While the `FILTER + (+)` method is efficient, advanced users often look for alternatives, especially when dealing with conditions that combine both AND and OR logic simultaneously. The primary alternative in [Google Sheets](#) is the **QUERY** function.

The **QUERY** function utilizes the SQL-like query language, which allows for explicit use of the `OR` keyword within its `WHERE` clause. For example, the same operation as our first example could be written using **QUERY** as: `=QUERY(A1:C10, "SELECT * WHERE A = 'A' OR C < 20", 1)`. While **QUERY** offers clearer syntax for complex Boolean logic, the **FILTER** function often calculates faster on very large datasets and is generally easier for beginners to grasp the fundamental mechanics of array filtering.