

How to Create Measures and Filter on Multiple Values in Power BI

Authored by
stats writer

January 27, 2026

RECOMMENDED CITATION

stats writer (2026). *How to Create Measures and Filter on Multiple Values in Power BI*. PSYCHOLOGICAL SCALES. Retrieved from <https://scales.arabpsychology.com/?p=128090>

Power BI is recognized globally as a robust platform for business intelligence, offering essential tools for comprehensive data analysis and visualization. One of the most critical aspects of advanced reporting within this environment is the ability to create dynamic calculations, known as Measures, which can be conditionally filtered. Defining a calculation that operates only on specific subsets of data--particularly when that subset involves multiple discrete values--is a common requirement for precise analytical reporting.

This guide details the expert methodology for constructing a custom DAX (Data Analysis Expressions) Measure designed to calculate results based on an internal filter spanning multiple values. This technique leverages the power of the **CALCULATE** function combined with the set-based **IN** operator, ensuring the calculation accurately reflects the desired criteria without relying on external visual filters that affect the entire report context.

The process begins with defining a new Measure in the Power BI Desktop environment. Users navigate to the "Home" tab or "Table tools" ribbon, select "New Measure," and then input the required DAX formula. By embedding the filtering logic directly into the Measure definition, we achieve stable and reusable calculations. This capability is fundamental for efficient data analysis, allowing analysts to create complex, multi-layered calculations quickly and effectively, significantly enhancing the depth of their visualizations and insights.

Power BI: Creating a Conditional Measure with Multi-Value Filtering

Understanding the Role of DAX in Conditional Calculations

DAX serves as the crucial language layer that enables sophisticated computation within Power BI. Unlike simple aggregate functions (like SUM or AVERAGE) that operate over the current filter context, complex calculations often require modifying that context or applying specific conditions directly within the calculation itself. This is where the **CALCULATE function becomes indispensable, acting as the**

foundation for nearly all advanced data analysis expressions.

When we aim to filter a calculation based on multiple criteria--for example, calculating the total sales only for 'Region A' and 'Region C'--we are performing a calculation under a modified filter context. Using the **CALCULATE** function allows us to override or inject new filter conditions into the existing report context. This guarantees that the resulting Measure is context-independent and will return the specified value regardless of the slicers or filters placed on the visual, unless those filters specifically target the Measure itself.

The ability to define precise conditional logic within a Measure is central to effective data modeling. By embedding complex filtering rules using **DAX**, analysts can create metrics that are both robust and highly specific, leading to clearer reporting and better decision-making. The structure of the underlying data model, often resembling a relational database structure, is leveraged by DAX to efficiently process these complex filter requests across large datasets.

Designing Measures with the CALCULATE and IN Functions

To effectively filter a calculation based on a list of discrete values, we combine the power of **CALCULATE** with a set operation. The standard approach involves using the logical OR condition repeatedly, but this can lead to verbose and error-prone code if the list of values is long. A far more elegant and scalable solution utilizes the **IN operator**. The **IN operator** simplifies the process of checking if a column value belongs to a specified list of constants, effectively creating an 'OR' condition across the entire list.

The general syntax for implementing this multi-value filter within a **CALCULATE** statement requires specifying the column to be filtered, followed by the IN operator, and then the list of values enclosed in curly braces {}. Each value in the list must be correctly formatted according to its data type--text values must be enclosed in double quotes, as demonstrated in the practical example below. This approach is highly efficient because it creates a filter table containing only the specified values and uses it to modify the calculation context.

DAX provides the necessary flexibility for this sophisticated filtering. You can use the following syntax to create a measure that filters rows based on multiple values, essentially implementing an AND Condition on the overall calculation, where the calculation is performed only for rows matching one of the specified criteria in the filter argument:

Method 1: Create Measure by Filtering with AND Condition using IN

Sum of Points =

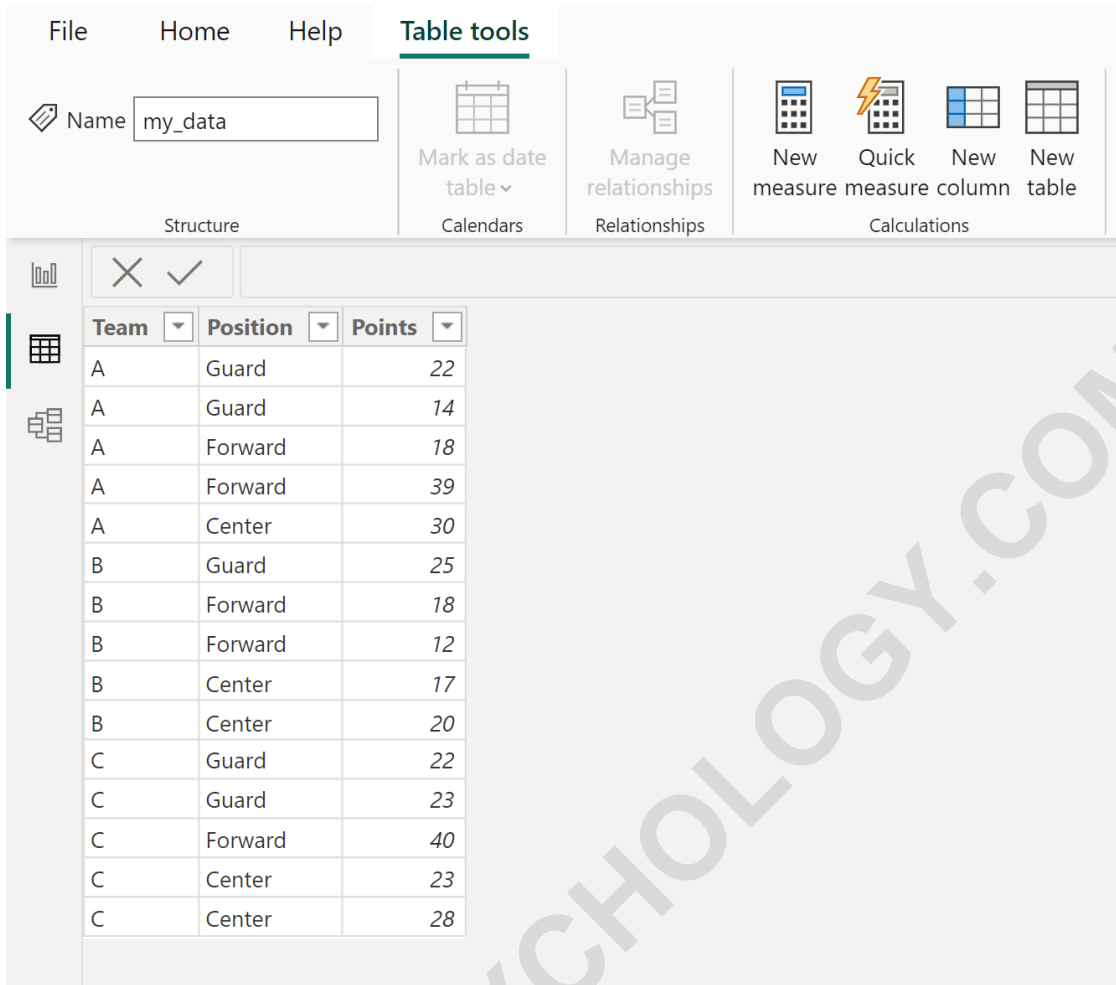
```
CALCULATE ( SUM ( 'my_data' ), 'my_data' IN { "A", "C" } )
```

This particular example defines a new **Measure** named **Sum of Points**. It utilizes the **CALCULATE** function to modify the context under which the **SUM** of the Points column is computed. Crucially, the calculation is restricted only to those rows where the Team column holds a value equal to either "A" or "C". The structure ensures that the filtering logic is applied internally before the aggregation takes place.

Example: Applying Multi-Value Filtering to Sample Data

To illustrate this concept practically, let us consider a dataset within Power BI Desktop. Suppose we have a table named `my_data` that captures performance statistics for various basketball players. This table includes key fields such as Player Name, Team affiliation, and Points scored. Our objective is to calculate the aggregated score only for players belonging to specific teams, simulating a focused performance analysis.

The following structure represents the foundational data table, `my_data`, loaded into the Power BI data model. Note the categorical nature of the Team column, which serves as our filtering dimension, and the numerical Points column, which is the subject of our aggregation.



The screenshot displays the Microsoft Power BI Desktop interface. The 'Table tools' ribbon is active, showing options for 'Name' (my_data), 'Mark as date table', 'Manage relationships', 'New measure', 'Quick measure', 'New column', and 'New table'. Below the ribbon, a table is displayed with the following data:

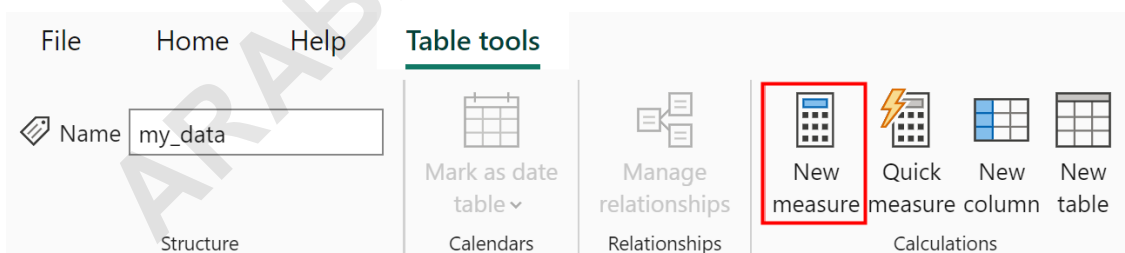
Team	Position	Points
A	Guard	22
A	Guard	14
A	Forward	18
A	Forward	39
A	Center	30
B	Guard	25
B	Forward	18
B	Forward	12
B	Center	17
B	Center	20
C	Guard	22
C	Guard	23
C	Forward	40
C	Center	23
C	Center	28

The specific requirement here is to calculate the total sum of values in the Points column, but we must strictly limit this calculation to include only those rows where the Team column value is either 'A' or 'C'. This operation demonstrates a fundamental necessity in business intelligence: isolating and aggregating data based on specific, non-contiguous categories.

Implementing the Measure in Power BI Desktop

The implementation process is straightforward once the **DAX** formula is defined. Within **Power BI Desktop**, the first step is to initiate the creation of a new **Measure**. This is typically done by navigating to the "Table tools" tab in the ribbon interface. This tab provides all the necessary functionalities for working with data model tables and calculations.

Under the "Table tools" ribbon, locate and click the New measure icon. Clicking this icon opens the formula bar, which is the dedicated space for writing and editing **DAX** expressions. It is crucial to ensure that the Measure is associated with the correct table (although Measures can float, linking them conceptually helps with organization).



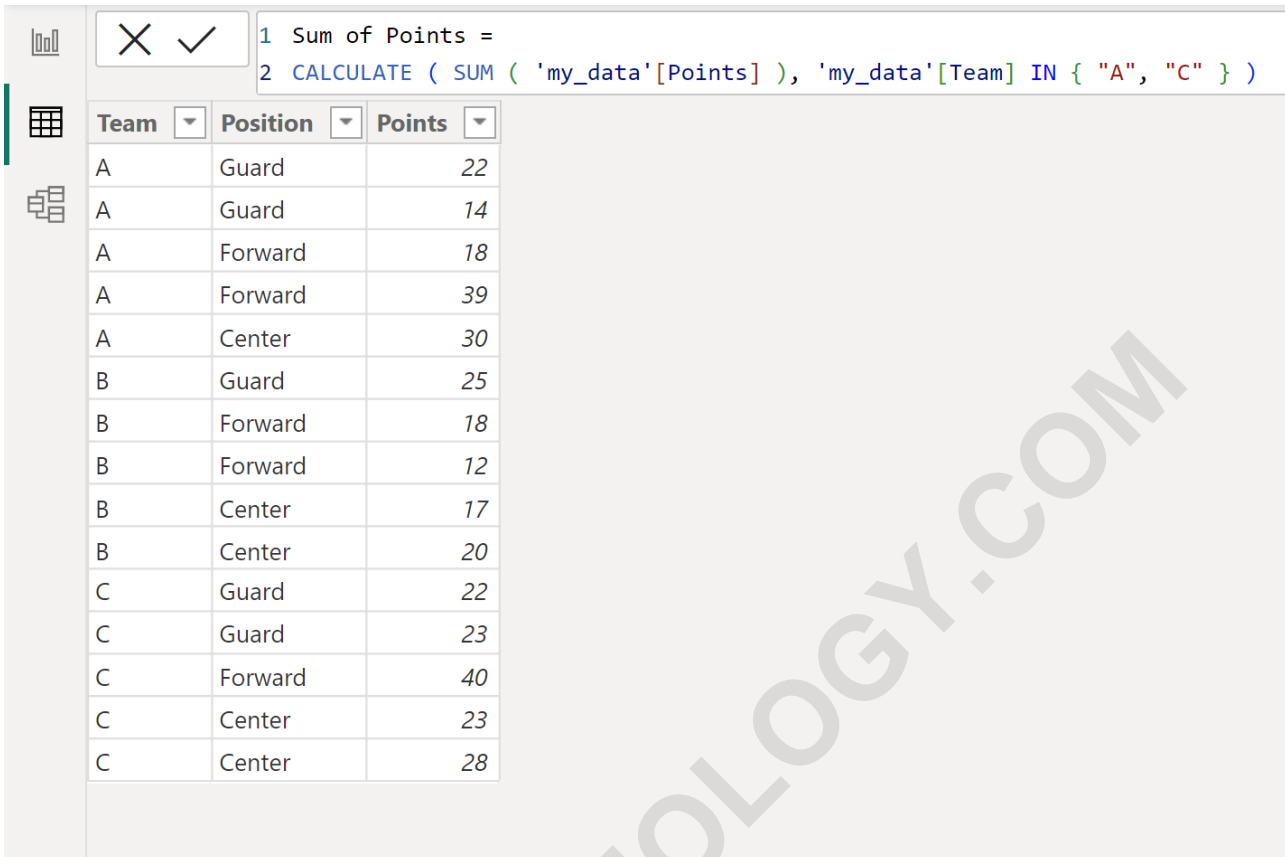
In the newly opened formula bar, the analyst must then type or paste the precise **DAX** formula that incorporates the multi-value filtering logic, ensuring correct spelling of column and table names. The formula used for this

specific requirement, calculating the sum of points for Team A and Team C, is as follows:

Sum of Points =

```
CALCULATE ( SUM ( 'my_data' ), 'my_data' IN { "A", "C" } )
```

After inputting the formula and pressing Enter, the new **Sum of Points Measure** is saved within the data model. This Measure is now ready to be used in any visualization, where it will consistently return the filtered sum as defined by the embedded **CALCULATE** statement, regardless of the visual context.



The screenshot shows the DAX editor in Power BI. The formula bar contains the following DAX code:

```
1 Sum of Points =  
2 CALCULATE ( SUM ( 'my_data'[Points] ), 'my_data'[Team] IN { "A", "C" } )
```

Below the formula bar, a table is displayed with the following data:

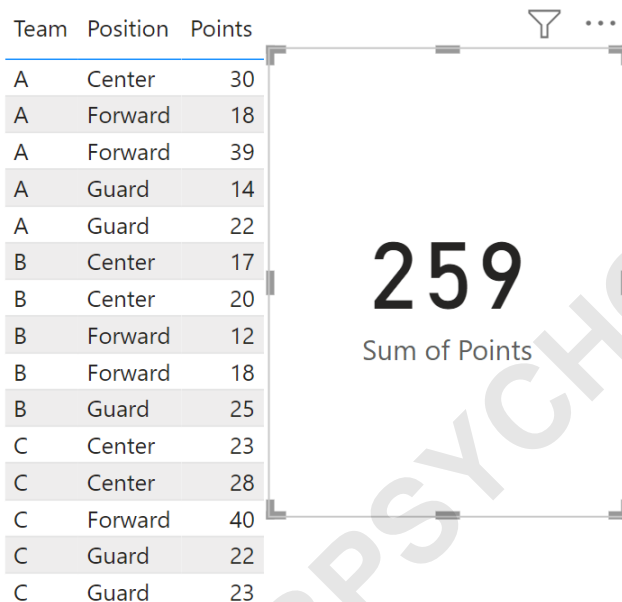
Team	Position	Points
A	Guard	22
A	Guard	14
A	Forward	18
A	Forward	39
A	Center	30
B	Guard	25
B	Forward	18
B	Forward	12
B	Center	17
B	Center	20
C	Guard	22
C	Guard	23
C	Forward	40
C	Center	23
C	Center	28

Analyzing and Interpreting the Results

Once the filtered Measure has been successfully defined, the next step is to visualize the result to confirm its correctness. This typically involves switching from the Data or Model view to the Report View in Power BI Desktop. To showcase a single aggregated value clearly, the Card visualization type is often the most suitable choice.

By dragging the newly created Sum of Points Measure onto the report canvas and selecting the Card visual,

the calculated value is displayed prominently. The value shown reflects the sum of all points for players whose team affiliation matches either "A" or "C", effectively demonstrating the successful application of the multi-value filter defined using the IN operator within the CALCULATE function.



Team	Position	Points
A	Center	30
A	Forward	18
A	Forward	39
A	Guard	14
A	Guard	22
B	Center	17
B	Center	20
B	Forward	12
B	Forward	18
B	Guard	25
C	Center	23
C	Center	28
C	Forward	40
C	Guard	22
C	Guard	23

259
Sum of Points

Upon reviewing the results against the source data (where Team A points are $100 + 56 = 156$, and Team C points are $45 + 58 = 103$), the calculated total for the measure is confirmed to be the sum of these subsets, resulting in $156 + 103 = 259$. This validation step confirms that the DAX formula correctly filtered out

rows associated with Team B and Team D before performing the final aggregation. The isolation of this specific numerical outcome is vital for targeted performance analysis and reporting.

Alternative DAX Filtering Techniques

While the IN operator within CALCULATE provides the cleanest and most efficient way to filter based on a list of discrete values, there are other methods available in DAX that achieve similar results, though often with greater complexity or reduced performance. Understanding these alternatives is essential for advanced data analysts.

One common alternative, especially when dealing with only two or three values, involves using the logical OR function. For instance, the same calculation could be written using nested OR conditions: `CALCULATE(SUM('my_data'), OR('my_data' = "A", 'my_data' = "C"))`. However, as the list of desired teams grows, this syntax becomes long, cumbersome, and prone to typographical errors. The IN operator is specifically designed to handle set membership checks more efficiently than repeated OR statements.

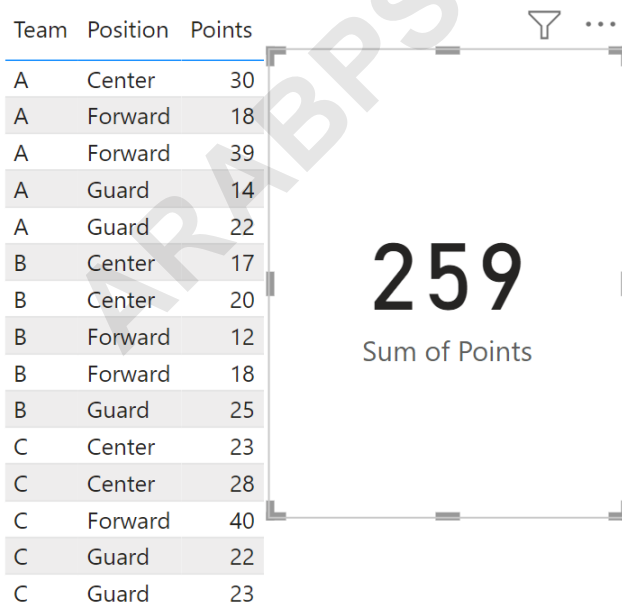
Another powerful, albeit more complex, approach is using the **FILTER** function in conjunction with **CALCULATE**. This method requires explicitly defining a new filter context that applies to the entire table or a subset thereof. While highly flexible, the structure for achieving multi-value filtering is still less direct than the specialized **IN operator**. Ultimately, for simple, constant-based multi-value filtering within a **Measure**, the syntax demonstrated in this article using `'Column' IN { Value1, Value2, ... }` remains the industry standard for clarity and performance.

Conclusion and Further Learning

The ability to create dynamically filtered measures is a cornerstone of advanced **Power BI** report development. By mastering the **CALCULATE** function in conjunction with the **IN operator**, users gain precise control over their aggregation logic, ensuring that complex business rules are accurately reflected in the derived metrics. This methodology allows for the creation of robust, reusable, and self-contained **Measures** that perform critical **data analysis** irrespective of external visual filters.

For those seeking to deepen their expertise in **DAX** and **Power BI**, further exploration of context transition, filter propagation, and advanced filter functions like **FILTER** is highly recommended. These skills are essential for tackling more sophisticated scenarios, such as creating dynamic lists for filtering based on user selection or applying calculations across disconnected tables in a complex **data modeling** environment.

The following resources and tutorials explain how to perform other common tasks in Power BI, offering pathways to continuous skill development and mastery of the platform:



Team	Position	Points
A	Center	30
A	Forward	18
A	Forward	39
A	Guard	14
A	Guard	22
B	Center	17
B	Center	20
B	Forward	12
B	Forward	18
B	Guard	25
C	Center	23
C	Center	28
C	Forward	40
C	Guard	22
C	Guard	23

259
Sum of Points