

How to Count Distinct Values with Filters in Power BI

Authored by
stats writer

January 27, 2026

RECOMMENDED CITATION

stats writer (2026). *How to Count Distinct Values with Filters in Power BI*.

PSYCHOLOGICAL SCALES. Retrieved from <https://scales.arabpsychology.com/?p=128063>

Power BI is recognized globally as an extremely powerful tool for data analysis and data visualization, enabling users to seamlessly process and manipulate massive datasets. A crucial capability within the platform is the ability to accurately count distinct values while simultaneously applying specific filters. This function empowers analysts to define precise criteria to segment their data before calculating the unique entries within that filtered subset.

The ability to perform a count of distinct values under filtering conditions is invaluable for identifying subtle patterns, verifying data integrity, and establishing trends that might be obscured in the overall dataset. By leveraging specialized functions within DAX (Data Analysis Expressions), users can ensure efficient and accurate analysis, leading directly to more informed and data-driven decisions. This guide will meticulously detail the syntax and practical application required to execute a distinct count with a filter in Power BI.

Before diving into the implementation, it is essential to appreciate the importance of calculated measures in Power BI. Unlike calculated columns, measures perform calculations on the fly, often aggregating data across multiple rows based on the current context--or in our case, based on an explicit filtering instruction embedded within the calculation itself. Mastering this technique is fundamental to advanced data modeling in Power BI.

Power BI: Count Distinct Values with Filter

Understanding the Core DAX Syntax

To perform a filtered distinct count in DAX, we must utilize a combination of functions that control the evaluation context. The cornerstone of this operation is the CALCULATE function, which is arguably the most powerful function in DAX, allowing the modification of the filter context under which an expression is evaluated.

The following syntax provides a clean and effective way to count the number of distinct values in a column of a table after applying a specific filter condition. This structure ensures that the aggregation only operates on the subset of data meeting the specified criteria.

Distinct Points =

```
CALCULATE (  
    DISTINCTCOUNT ( 'my_data' ),  
    FILTER ( 'my_data', 'my_data' = "C" )  
)
```

In this particular expression, a new measure named **Distinct Points** is being created. This measure is designed to first define a filter context using the FILTER function, restricting the scope

to rows where the **Team** column in the 'my_data' table equals "C". Once this subset is defined, the DISTINCTCOUNT function is applied to count the unique entries within the **Points** column, limited only to those filtered rows. This layered approach guarantees precision in complex analytical tasks.

Deconstructing the Key DAX Functions

A deep understanding of the three primary functions involved--CALCULATE, DISTINCTCOUNT, and FILTER--is essential for writing robust and accurate DAX code. These functions work together sequentially to achieve the desired result of conditional aggregation.

First, the CALCULATE function takes the expression (the DISTINCTCOUNT) and applies the subsequent filter arguments. It temporarily changes the evaluation context for the contained expression. Secondly, the DISTINCTCOUNT function performs the actual calculation, counting all non-blank unique values in the specified column. Finally, the FILTER function is an iterator that evaluates a Boolean condition for every row of the specified table (in this case, 'my_data'). It returns a virtual table containing only the rows where the condition ('my_data' = "C") is true, which then acts as the new filter context for CALCULATE.

By combining these elements, we achieve an operation that is far more flexible than a simple pivot table aggregation. We are effectively overriding the default row context or report-level filters to enforce a specific, hard-coded filter ('Team' equals "C") solely for the purpose of calculating this specific measure, allowing for highly targeted metrics within a broader report. This powerful capability sets DAX apart for professional business intelligence reporting.

Step-by-Step Example Setup: Defining the Dataset

To illustrate the practical use of this formula, let us consider a sample dataset. Suppose we are working with a table in Power BI named **my_data**. This table contains performance statistics, specifically tracking the points scored by individual basketball players distributed across various teams.

This sample dataset is structured typically, including columns like Player ID, Team affiliation, and Points scored. The objective in this scenario is not just to count all unique scores but specifically to determine how many unique scoring levels exist only within a specific team, identified as "Team C".

The structure of our hypothetical data table, **my_data**, is displayed below. Note the redundancy in the 'Points' column for Team C; this is precisely why a distinct count is necessary--to count the variety of scores rather than the total number of scoring instances.

The screenshot shows the Power BI Desktop interface. The 'Table tools' ribbon is active, displaying options like 'Mark as date table', 'Manage relationships', and 'New measure'. Below the ribbon, a data table is visible with the following columns: Team, Position, and Points. The table contains 15 rows of data. A large watermark 'ARABPSYCHOLOGY.COM' is overlaid diagonally across the table.

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

Our goal is precise: we intend to count the number of unique, distinct values present in the **Points** column of the table, but only for those rows where the corresponding value in the **Team** column is equal to "C". This focused analysis isolates the scoring patterns specific to that team.

Implementing the New Measure in Power BI

The implementation begins within the Power BI Desktop environment, specifically within the data or modeling view, where new calculations are defined. Measures are generally preferred over calculated columns for aggregations because they save space and calculate dynamically based on the visual context.

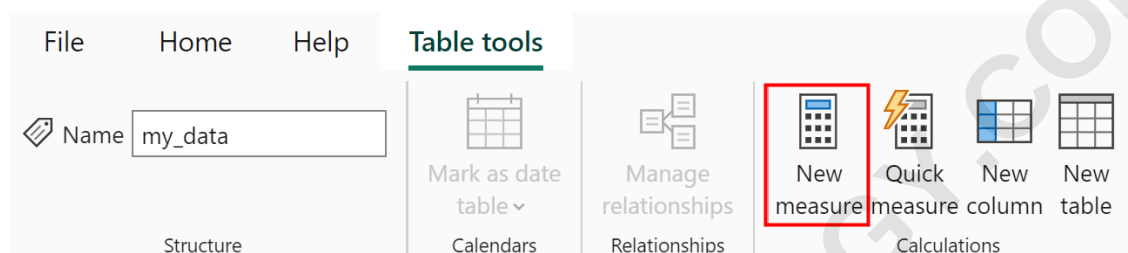
To initiate the creation of our custom analytic measure, the user must navigate to the appropriate menu within the Power BI interface. The standard procedure involves the following steps:

Click on the **Table tools** tab, which is located prominently along the top ribbon interface of Power

BI Desktop.

Locate and click the **New measure** icon within that tab. This action opens the formula bar, allowing the user to define the DAX expression.

This process ensures the new calculation is defined as a measure, which will reside in the data model and can be dragged into any visualization. The measure will recalculate whenever the filter context changes, although in this case, the internal CALCULATE function fixes one aspect of the filtering.



Defining the DAX Formula and Observing the Result

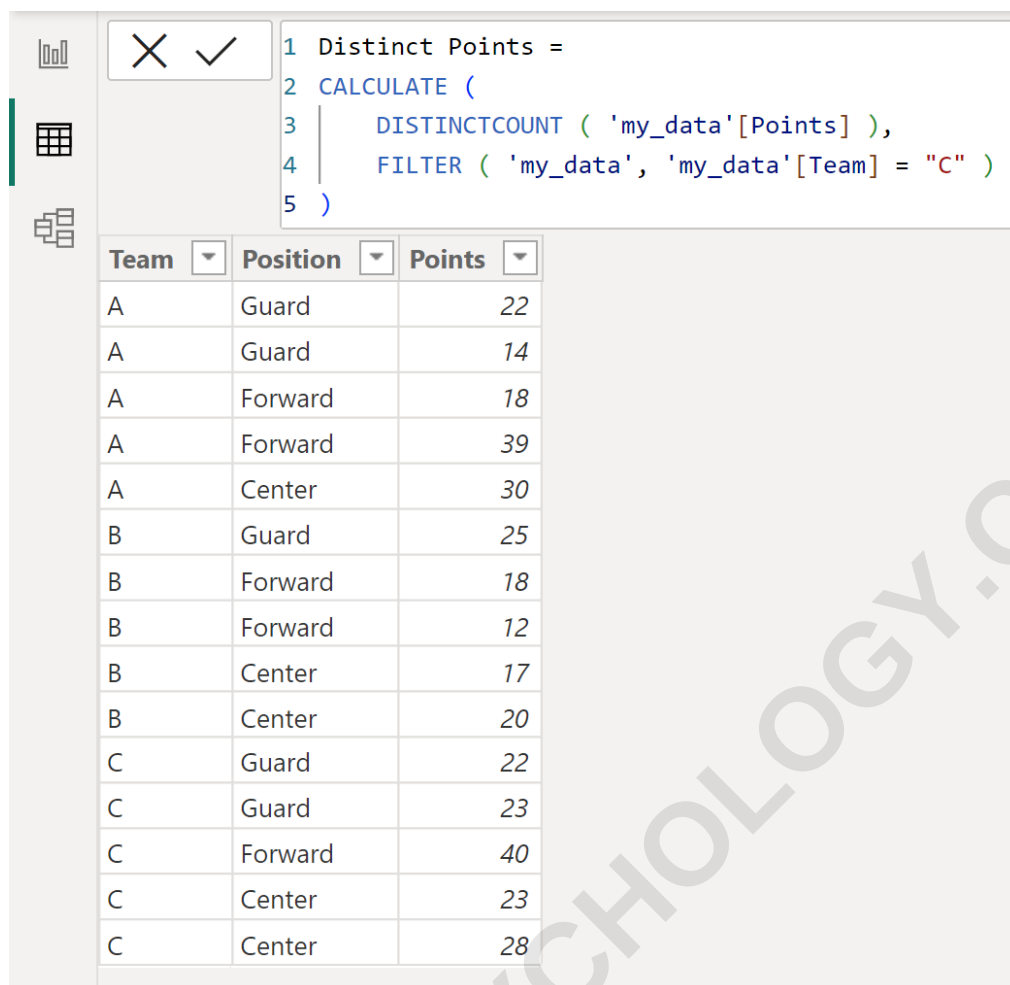
Once the formula bar is active, the previously defined DAX formula must be carefully entered. Precision in capitalization and syntax is paramount for DAX functions to execute correctly.

Type the following formula exactly as shown into the formula bar:

```
Distinct Points =
CALCULATE (
DISTINCTCOUNT ( 'my_data' ),
FILTER ( 'my_data', 'my_data' = "C" )
)
```

Upon committing this formula, the Power BI data model successfully registers a new measure named **Distinct Points**. This measure now encapsulates the logic necessary to count the distinct values in the **Points** column, restricted exclusively to those rows where the **Team** value is equal to "C". The measure will then display the single numerical result when used in a visualization.

The immediate result of creating this measure can often be viewed directly in the data table or measure list, showing the calculated value which is 4 in this specific instance. This outcome signifies that Team C has scored 4 unique point totals across all their recorded entries in the dataset.



The screenshot shows the Power BI Desktop interface. On the left is the Visualizations pane with icons for Bar chart, Table, and Matrix. The main area displays a DAX formula in the formula bar and a data table below it. The formula is: `1 Distinct Points =`
`2 CALCULATE (`
`3 DISTINCTCOUNT ( 'my_data'[Points] ),`
`4 FILTER ( 'my_data', 'my_data'[Team] = "C" )`
`5 )`

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

Visualizing the Calculated Measure

While the measure is correctly calculated within the data model, its utility is realized when it is presented visually on a report page. The Card visual is the most straightforward and effective method for displaying a single, powerful numeric metric like a distinct count.

To display this value within a report, follow these steps in the Power BI **Report View**:

Navigate to the **Report View** tab in Power BI Desktop.

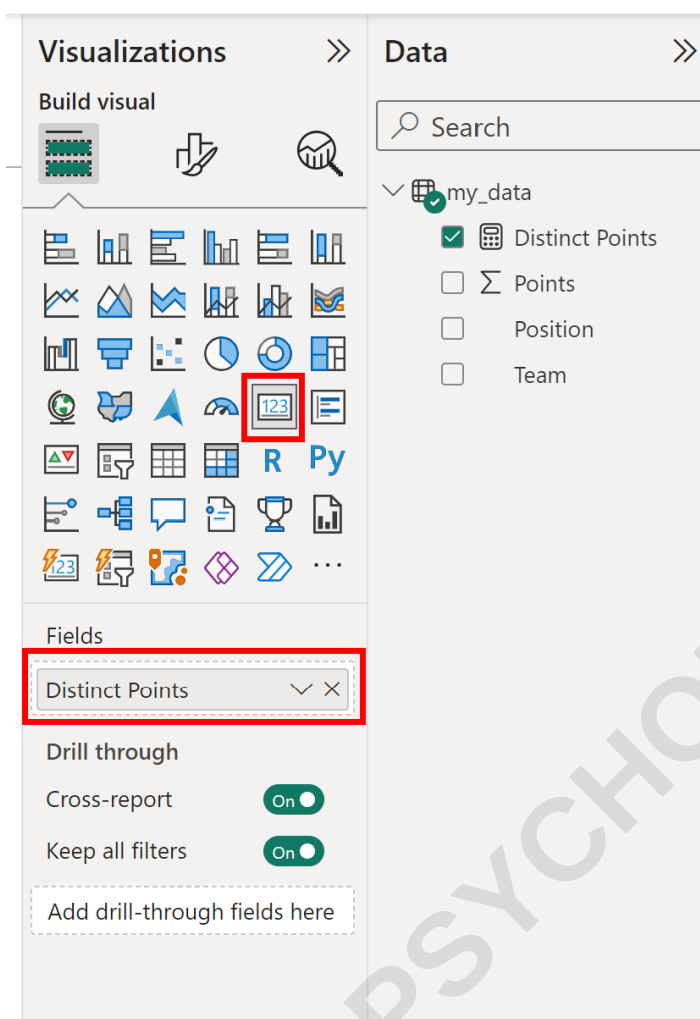
Under the **Visualizations** pane, click the **Card** icon to insert a new Card visual onto the report canvas.

From the Fields list, locate the newly created measure, **Distinct Points**.

Drag this measure into the **Fields** well of the Card visualization.

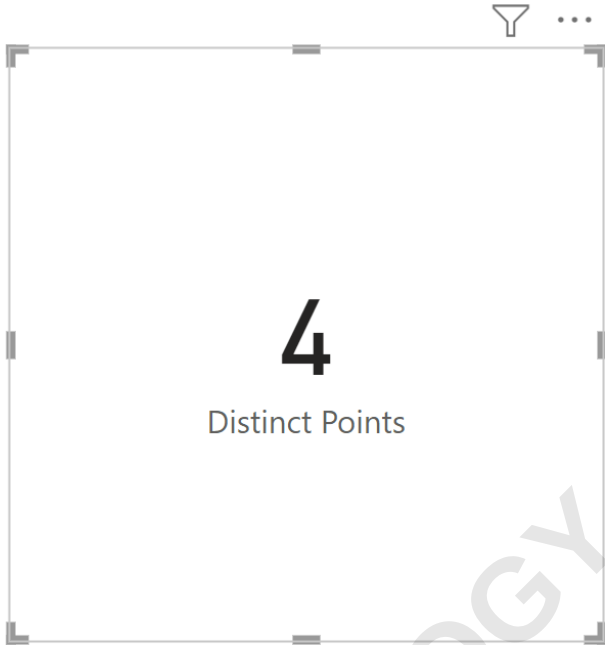
This procedure links the analytical output directly to the report interface, transforming the

underlying DAX logic into actionable insight. This visualization is particularly useful when comparing fixed metrics against dynamic, context-driven metrics within the same dashboard.



The final output is a clean Card visual displaying the result of the filtered distinct count. This visual clearly communicates the derived metric to the end-user, illustrating that the analytical goal--counting the unique point values specifically for team C--has been successfully achieved and validated.

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



As confirmed by the output, the Card visual proudly displays the number **4**. This confirms that there are exactly four unique point totals recorded in the **Points** column specifically for rows associated with Team C, demonstrating the precise control offered by the combined CALCULATE and FILTER functions.

Advanced Considerations and Performance Implications

While the CALCULATE(DISTINCTCOUNT(FILTER())) pattern is highly effective, it is important for expert users to consider alternatives and performance implications, especially when dealing with extremely large datasets. DAX calculations that involve filtering large tables can sometimes impact query performance.

For specific scenarios, particularly where context transition is desired or performance needs optimization, alternative functions might be used. For instance, sometimes the DISTINCT function can be used to generate a virtual table of distinct values before counting them, though for simple measures, the structure provided remains the most readable and standard approach.

It is also worth noting the critical difference between using the FILTER function as a separate argument within CALCULATE, as shown here, versus using a simple Boolean expression directly. When you use a complex filter like FILTER('my_data', condition), it forces an iteration over the entire 'my_data' table, potentially impacting performance. However, for filters based on simple column equality against a constant value (like 'Team' = "C"), Power BI often optimizes this

internally, making the complex syntax acceptable and highly readable.

Note: You can find the complete documentation for the **DISTINCTCOUNT** function in [DAX](#), along with details on its usage, performance characteristics, and limitations, on the official Microsoft Learn documentation platform.

Further Power BI Tutorials and Resources

Mastering the art of context modification using [CALCULATE](#) and filtering functions is a key step toward becoming proficient in [Power BI](#) data modeling. The principles applied here extend to numerous other aggregation types, such as calculating filtered sums (**SUM**) or averages (**AVERAGE**).

The following resources explain how to perform other common and advanced tasks in Power BI, building upon the foundational knowledge of DAX measures and context control demonstrated in this tutorial:

Implementing complex time intelligence functions using DAX.

Creating dynamic segmentation based on user interaction filters.

Understanding the difference between row context and filter context in measure calculation.

Optimizing data models for faster report rendering and calculation speed.

By consistently practicing and applying these DAX principles, users can unlock the full potential of Power BI as a comprehensive business intelligence solution.