How to Ignore Filters for Measures in Power BI

Authored by

mohammed looti

January 11, 2026

Welcome to this detailed guide on managing data granularity in Power BI. One of the most common challenges faced by analysts is controlling how visual filters interact with calculated values, specifically measures. When a report user applies a slicer or selects a segment, all visuals usually update immediately. However, there are critical scenarios--such as calculating market share, percentage of total, or a grand average--where a measure must remain constant regardless of the surrounding visual context.

The core concept behind this behavior is the Filter Context. Every visual in Power BI operates within this context, defined by the rows, columns, and slicers applied. To achieve complex analytical comparisons, we often need to intentionally break or modify this context for a specific calculation. This article explores two primary, powerful methods to achieve this objective, focusing heavily on the precision offered by Data Analysis Expressions (DAX).

Method 1: Controlling Interactions Through Visual Settings

In certain lightweight scenarios, you can ignore a filter for a measure by utilizing the built-in "Edit Interactions" feature available within the Power BI desktop environment. This feature provides a quick, visual way to specify how filters from one visual (like a slicer or chart) affect another visual containing your measure.

By accessing the "Format" ribbon when a visual is selected and choosing "Edit Interactions," you can define the relationship between the filtering visual and the target visual (often a Card or Table displaying the measure). For interactions where you want the measure to remain unaffected by the filter, you can simply select the "None" interaction setting. This approach is highly useful for specific visual overrides but lacks the universal control and flexibility necessary for reusable corporate measures that must always ignore certain dimensions, regardless of where they are placed on the report page.

Method 2: Mastering DAX for Universal Filter Removal

For robust, enterprise-level calculations that must consistently ignore filters across an entire report, mastering Data Analysis Expressions (DAX) is essential. The most powerful way to override the standard Filter Context is by using the fundamental filter manipulation functions within DAX, primarily the **ALL function** in conjunction with the indispensable **CALCULATE function**.

The **ALL function** is designed to return all rows in a table, or all values in specific columns, effectively ignoring any external filters that might otherwise be applied. When this function is used as a filter argument within the **CALCULATE function**, it allows you to evaluate an expression (like SUM or MAX) across the entire dataset or across a specific, unfiltered dimension, thereby bypassing the current visual context.

For example, if you wish to define a measure named **Max Points** that calculates the maximum value in the **Points** column of a table, but you want this result to remain unaffected by any filtering applied specifically to the **Team** column, you would construct the following syntax in DAX. Notice how the ALL function is used to explicitly remove filters from the 'my_data' column before the aggregation takes place:

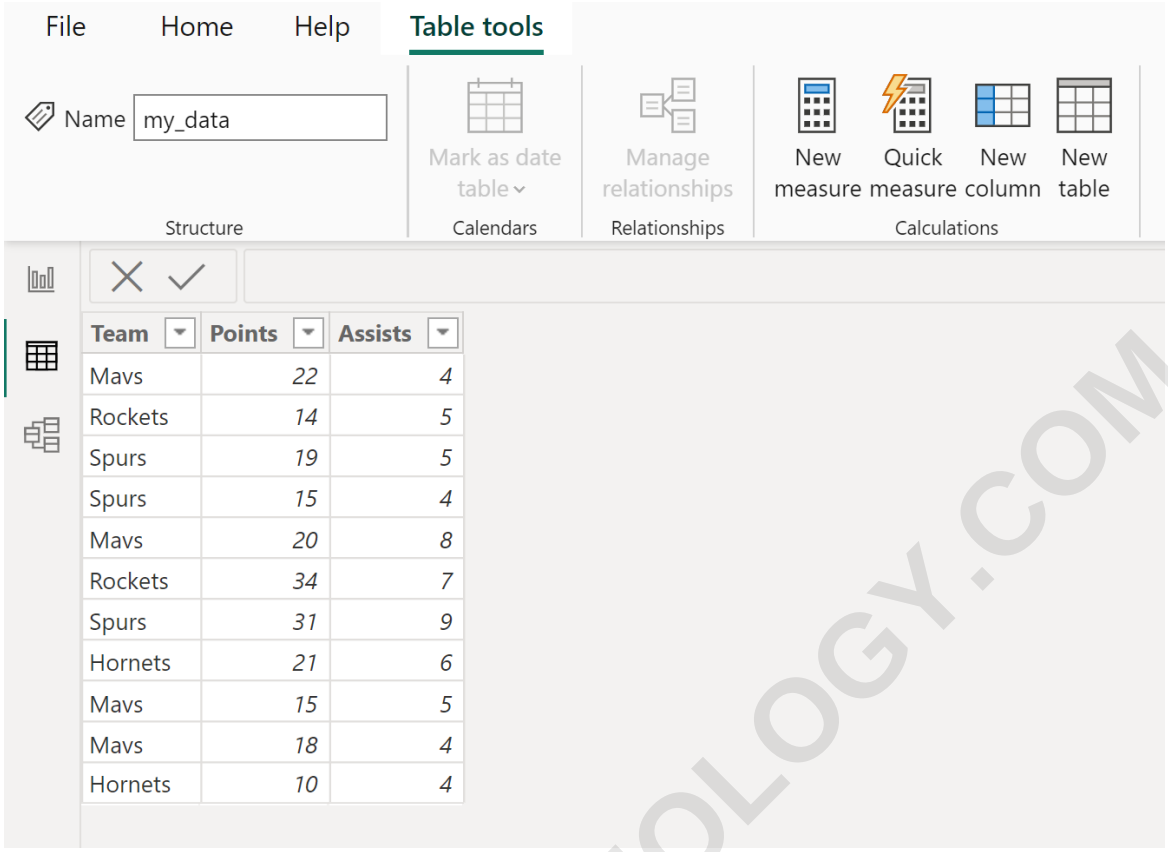
Max Points = CALCULATE(MAX('my_data'),ALL('my_data'))

This powerful combination of functions ensures that the measure always references the overall maximum points scored across all teams, regardless of which individual team is currently selected in a slicer or row context. The following practical demonstration walks through this process step-by-step, illustrating the critical difference between a standard measure and one engineered to ignore filters.

Practical Example: How to Ignore Filter for a Measure in Power BI

To fully understand the mechanics of filter ignoring, let us work through a detailed example using a simple dataset. Suppose we are tracking the performance of basketball players and their scores. We have the following sample data loaded into Power BI, contained within a table named **my_data**. This table includes information such as the player's team and the points they scored in a particular game.

This scenario requires us to calculate two versions of the maximum points: first, the naive maximum that respects filters, and second, a global maximum that ignores filters on the Team column, allowing us to compare individual team performance against the overall highest score recorded in the dataset.



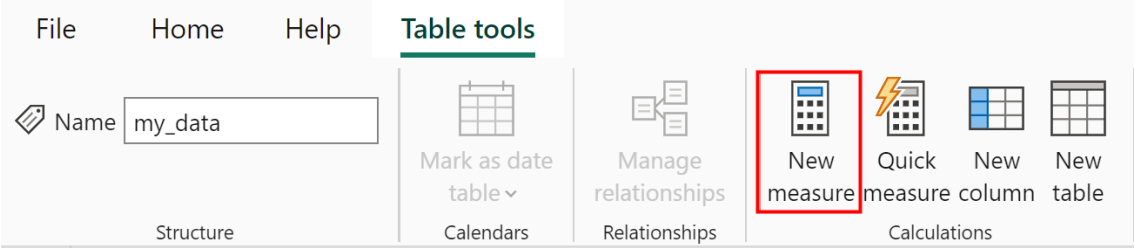
Team	Points	Assists
Mavs	22	4
Rockets	14	5
Spurs	19	5
Spurs	15	4
Mavs	20	8
Rockets	34	7
Spurs	31	9
Hornets	21	6
Mavs	15	5
Mavs	18	4
Hornets	10	4

Our primary objective in this exercise is straightforward: we want to calculate the absolute maximum value present in the **Points** column. Crucially, we want to ensure that this maximum value is always visible, even when the user filters down to a single team.

Step 1: Creating the Default Filtered Measure

Before implementing the filter-ignoring logic, it is beneficial to first create a standard measure that inherently respects the Filter Context. This baseline measure will serve as a crucial comparison point, demonstrating what happens when filters are applied without explicit DAX control.

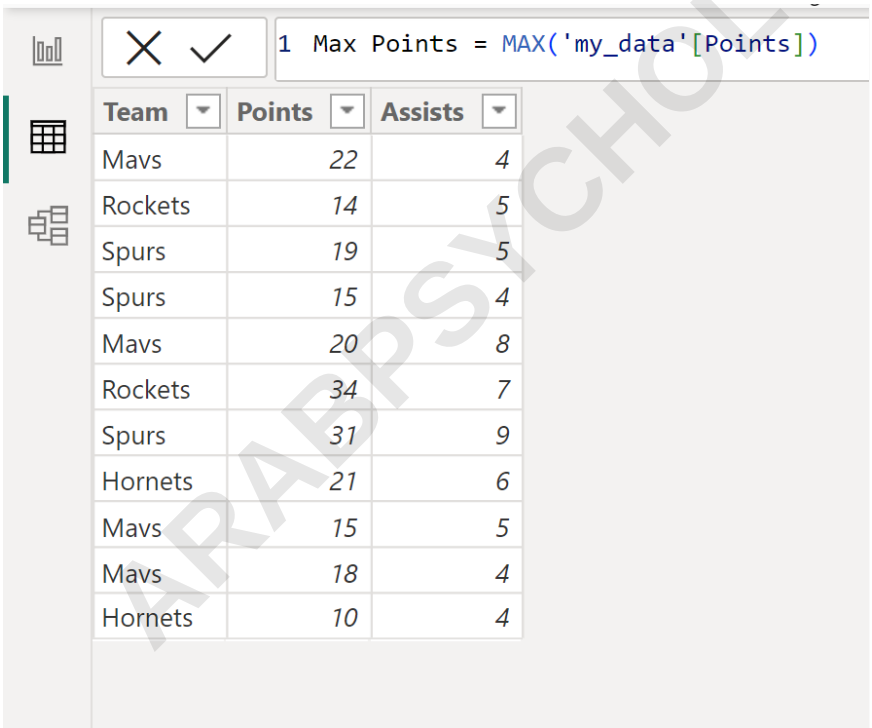
To begin, navigate to the **Table tools** tab located along the top ribbon in Power BI Desktop, and then click the **New measure** icon. This initiates the process of writing our first calculation. We will define a simple measure that uses the standard **MAX** aggregation function.



In the formula bar, we define the basic maximum points calculation. This definition simply finds the maximum value within the 'my_data' column within whatever context the visual currently dictates:

Max Points = MAX('my_data')

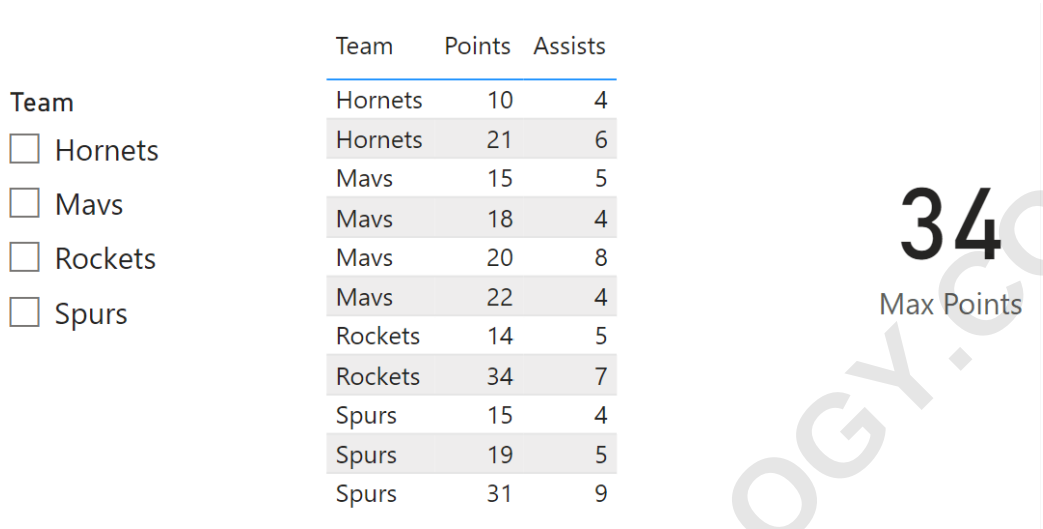
Executing this formula creates a new measure named **Max Points**. When displayed in a visualization, it correctly shows the overall maximum points available in the entire table, which we observe to be **34**.



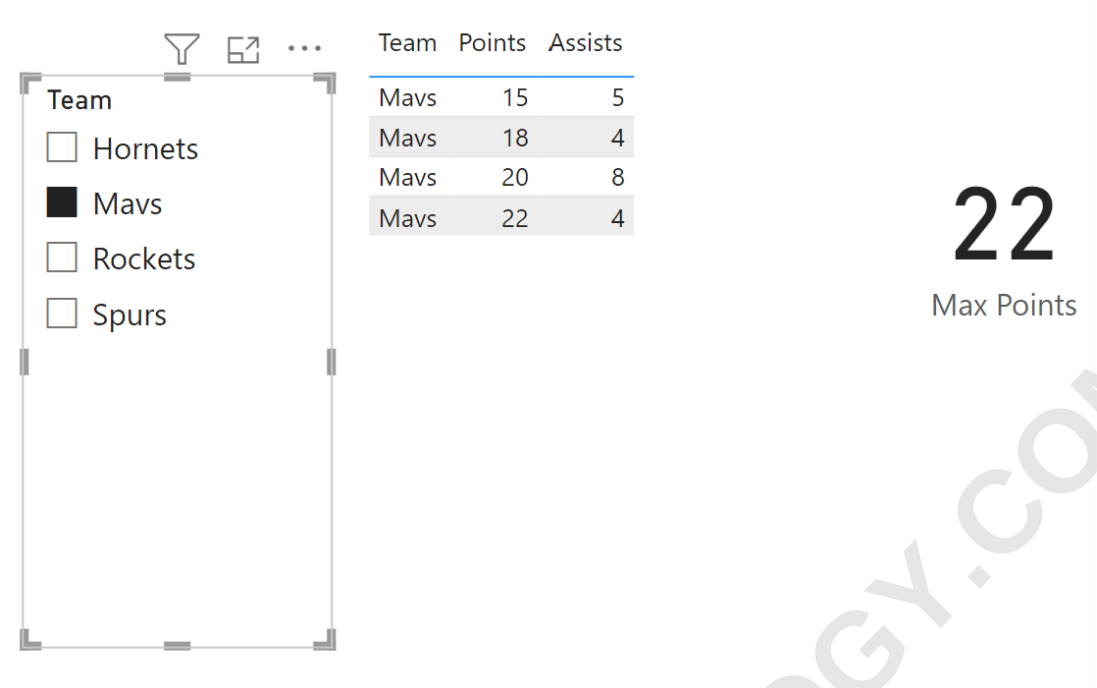
Step 2: Observing Standard Filter Behavior

Next, we switch to the report view and incorporate this newly created measure into a visualization, such as a Card visual, to observe its initial behavior. We also add a Slicer visual utilizing the 'my_data' column, which will allow us to manipulate the filter context dynamically.

Initially, with no team selected in the slicer, the Card correctly displays the absolute maximum value found across the entire dataset, confirming that the maximum value in the Points column is indeed **34**. This state represents the global context where no filters are actively reducing the scope of the calculation.



Now, consider what happens when a filter is applied. If a user selects a specific team--for instance, the "Lakers"--in the slicer, the standard **Max Points** measure immediately changes. It updates to display the maximum points achieved **only** by players on the Lakers team. This demonstrates how the report filter is flowing through to the measure's calculation environment, constraining the result set based on the user selection.



Step 3: Implementing the Filter-Ignoring Measure using CALCULATE and ALL

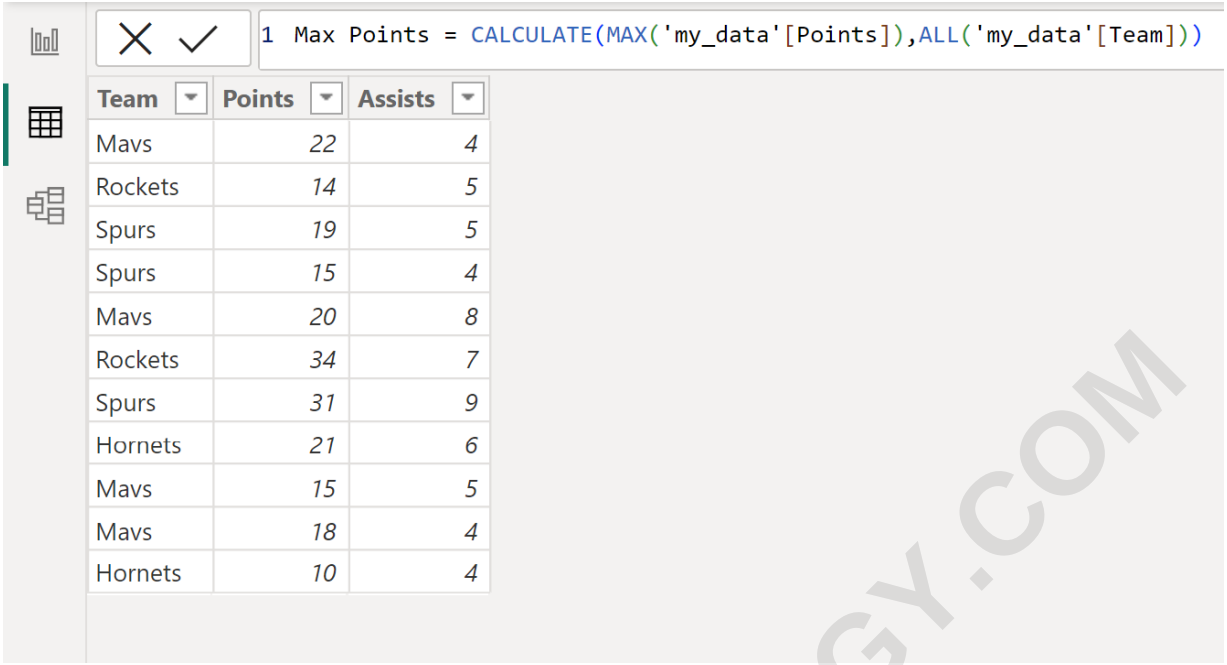
The goal is to redefine the measure so that, even if the user filters the visual down to a single team, the measure still displays the overall maximum score of 34. To achieve this crucial separation from the visual context, we must rename our original measure (or create a new one, perhaps called "Max Points Global") and introduce the **CALCULATE function** coupled with the **ALL function**.

The **CALCULATE function** is the single most important function in DAX, as it allows you to modify the Filter Context under which an expression is evaluated. By passing the expression `MAX('my_data')` and the filter argument `ALL('my_data')`, we instruct Power BI to perform the maximum calculation only after removing any filters currently applied to the 'my_data' column.

We use the following syntax instead for our redefined measure:

Max Points Global = CALCULATE(MAX('my_data'),ALL('my_data'))

This definition creates a robust new measure, **Max Points Global**, which is programmed to calculate the absolute maximum points across the entire underlying table, systematically overriding any filtering based on the **Team** column within the visual's local context. This measure is now ready to be deployed for comparisons.

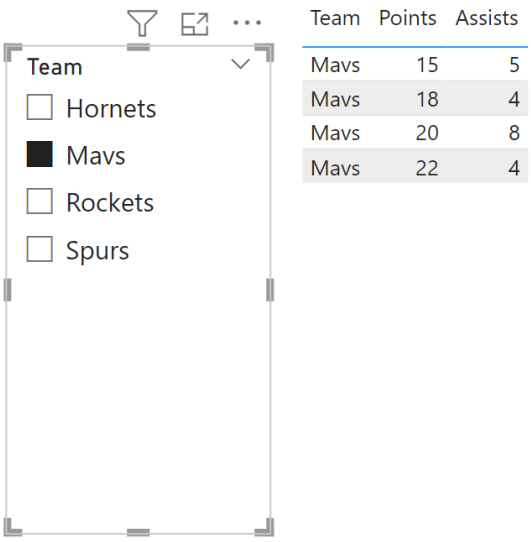


Team	Points	Assists
Mavs	22	4
Rockets	14	5
Spurs	19	5
Spurs	15	4
Mavs	20	8
Rockets	34	7
Spurs	31	9
Hornets	21	6
Mavs	15	5
Mavs	18	4
Hornets	10	4

Step 4: Verifying the Filter Independence

Finally, we replace the original measure with the new **Max Points Global** measure in our Card visualization. We then re-apply the filter, this time selecting the "Mavs" team in the slicer to test its effectiveness.

Despite the report view being filtered exclusively to the Mavs team, the value shown in the card visualization remains **34**. This is the desired outcome. The measure successfully ignored the filter applied to the 'my_data' column, continuing to report the maximum score from the entire dataset. This filter independence is precisely what is needed for calculating ratios or fixed benchmarks.



Summary of Filter Context Manipulation in DAX

The core takeaway from this example is understanding the distinction between the standard evaluation context and the modified context enforced by the CALCULATE function. When you use `ALL('my_data')`, you are telling DAX to treat the column specified within `ALL` as if no filters were applied to it, even if a slicer is actively filtering that column.

This method is powerful because it offers granular control. You can use `ALL(Table)` to remove all filters from an entire table (e.g., when calculating a grand total), or you can use `ALL(Table)`, as demonstrated here, to selectively remove filters from just a single column while allowing other related filters (e.g., date or region) to still affect the measure.

It is important to note the versatility of the **CALCULATE function**. It is the engine that drives virtually all advanced DAX calculations requiring context modification. Familiarity with its syntax and its interaction with functions like `ALL`, `ALLEXCEPT`, and `REMOVEFILTERS` is mandatory for mastering analytical reporting in Power BI.

For more detailed technical specifications and advanced uses, you can find the complete documentation for the **CALCULATE** function in DAX online documentation. Understanding this function is the key to unlocking highly flexible and context-aware reporting capabilities.

Further Learning Resources

The concepts of Filter Context and row context are foundational to writing efficient and accurate DAX. To deepen your expertise, consider exploring tutorials on related concepts:

Understanding the difference between `ALL()`, `ALLEXCEPT()`, and `REMOVEFILTERS()`.
How to use `CALCULATE` for time intelligence functions (e.g., Year-over-Year calculations).
Developing measures for calculating Percent of Grand Total using the pattern demonstrated here.

The following list provides links to other helpful tutorials explaining how to perform other common tasks in Power BI: