How to Calculate Conditional Averages with AVERAGEIF in Power BI

Authored by

mohammed looti

January 12, 2026

The concept of **AVERAGE IF** is fundamental in data analysis, allowing users to calculate the mean of a dataset based on specific, predefined criteria. While spreadsheet applications like Excel offer a dedicated `AVERAGEIF` function, Power BI achieves this powerful conditional aggregation using a combination of functions within the DAX (Data Analysis Expressions) language. This capability is exceptionally useful for business intelligence professionals seeking to drill down into complex datasets, isolating specific segments for targeted metric calculation.

1. Introduction: Conditional Aggregation in Power BI

A classic application involves scenario analysis, such as when a sales manager needs to calculate the average monthly sales solely for a specific product line or within a particular geographic region. By implementing conditional averaging, analysts can quickly identify performance disparities, understand which segments are contributing most effectively, and pinpoint areas that require strategic improvement. Furthermore, this conditional logic seamlessly integrates with other analytical operations, allowing for robust and flexible data manipulation within the Power BI environment.

To effectively implement the logic of **AVERAGE IF** in Power BI, we rely primarily on two critical DAX functions: CALCULATE, which is essential for context modification, and FILTER, which defines the criteria under which the average is computed. Understanding how these functions interact is paramount to mastering complex metric generation in business intelligence reporting.

2. The Core DAX Components for Conditional Averaging

To translate the simple idea of "AVERAGE IF" into functional DAX code, we must understand the role of the evaluation context. CALCULATE is arguably the most powerful function in DAX because it allows us to override the default filter context that Power BI establishes. When calculating a column or measure, CALCULATE takes an expression (in this case, the AVERAGE calculation) and then applies one or more filter arguments to redefine the scope of that calculation.

The FILTER function is used as one of these arguments within CALCULATE. FILTER iterates over a specified table and returns a subset of rows that meet a defined logical condition. This is precisely how we implement the "IF" condition. By nesting the AVERAGE function within CALCULATE, and then applying a FILTER argument, we instruct Power BI to calculate the average, but only for the rows that satisfy our specific criteria.

3. Defining the AVERAGE IF Syntax in DAX

The standard methodology for writing an **AVERAGE IF** calculation in DAX involves defining the measure or calculated column using the structure outlined below. This syntax is highly flexible and

can be adapted for virtually any conditional aggregation requirement across different fields and tables within your data model.

The following syntax demonstrates the standard structure used in DAX to achieve the functionality of an **AVERAGE IF** function in Power BI:

```
Avg Points =
CALCULATE (
AVERAGE ( 'my_data' ),
FILTER ( 'my_data', 'my_data' = EARLIER ( 'my_data' ) )
)
```

This particular formula creates a new calculated column named **Avg Points** that contains the average value derived from the **Points** column, specifically calculated for each unique value present in the **Team** column within the table designated as **my_data**. This conditional calculation is performed row by row, ensuring that the average is contextualized by the team associated with that specific row.

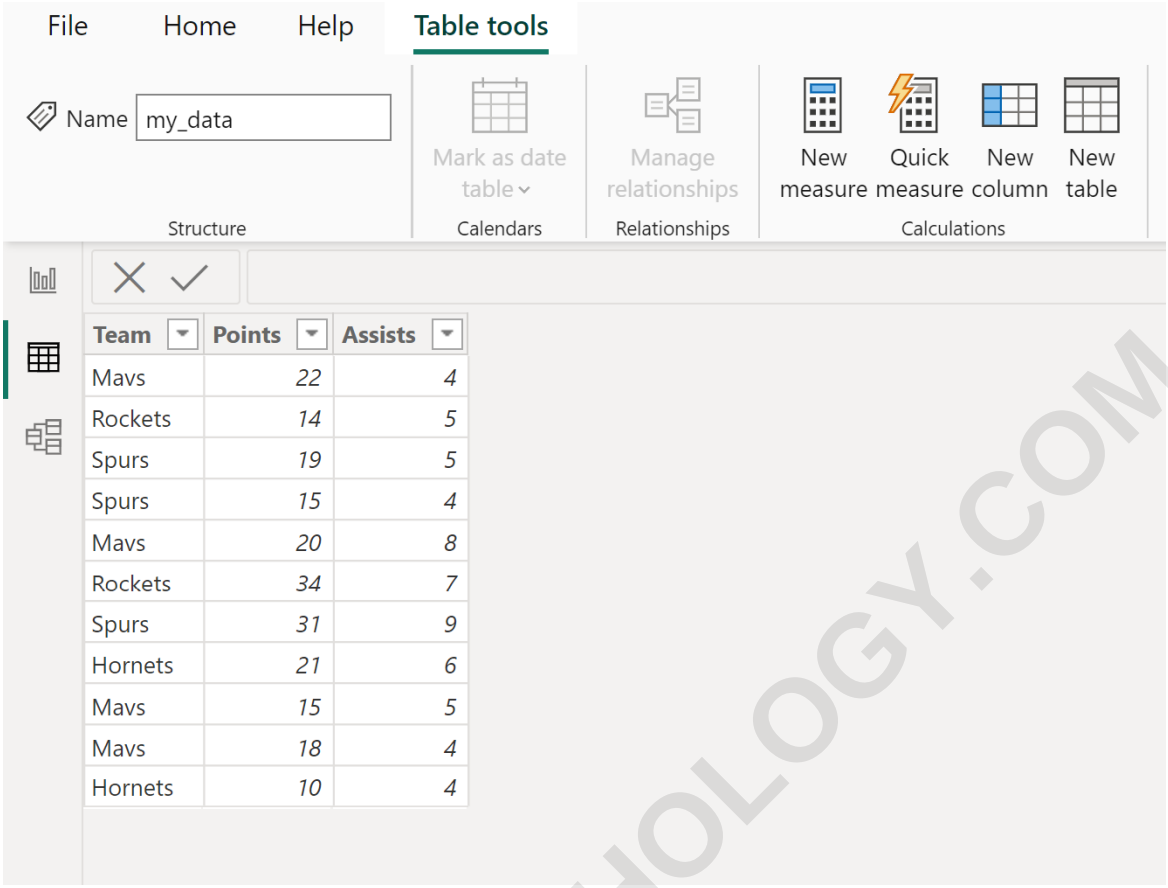
The following example shows how to use this formula in practice.

Example: How to Use AVERAGE IF in Power BI

4. Data Preparation and Scenario Setup

To illustrate the practical application of the conditional average using DAX, we will work with a sample dataset. This data represents key performance indicators for various basketball players, including their associated team and the points they scored in recent games. Our objective is to enrich this data model by adding a column that reveals the average performance of their teammates.

Suppose we have the following raw table loaded into Power BI, named **my_data** that contains information about various basketball players:



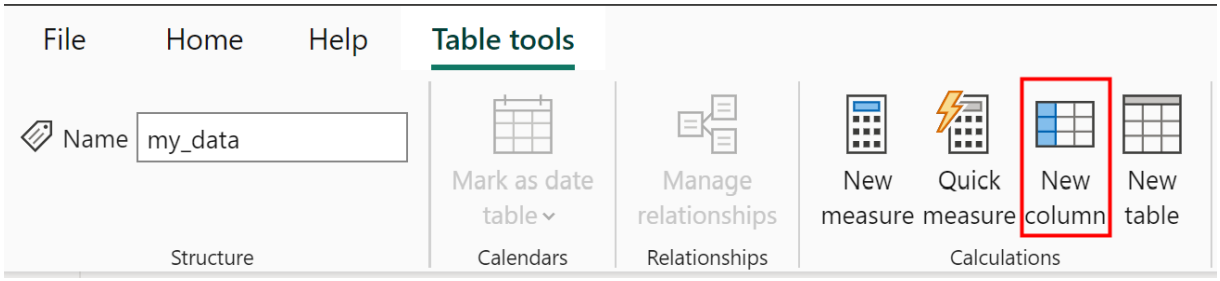
Team	Points	Assists
Mavs	22	4
Rockets	14	5
Spurs	19	5
Spurs	15	4
Mavs	20	8
Rockets	34	7
Spurs	31	9
Hornets	21	6
Mavs	15	5
Mavs	18	4
Hornets	10	4

Our specific goal is to leverage the power of conditional aggregation to create a new column that effectively shows the average points scored by all players within the same team. This metric is valuable for comparative analysis, allowing us to see how an individual player's score relates to the overall performance standard established by their peers within the organization.

5. Implementing the Calculated Column in Power BI

The creation of a calculated column in Power BI is a straightforward process when following standard development protocols. We will initiate this process from the Data view within the Power BI Desktop application.

To initiate the creation of our conditional average column, first click the **Table tools** tab, then click the **New column** icon:



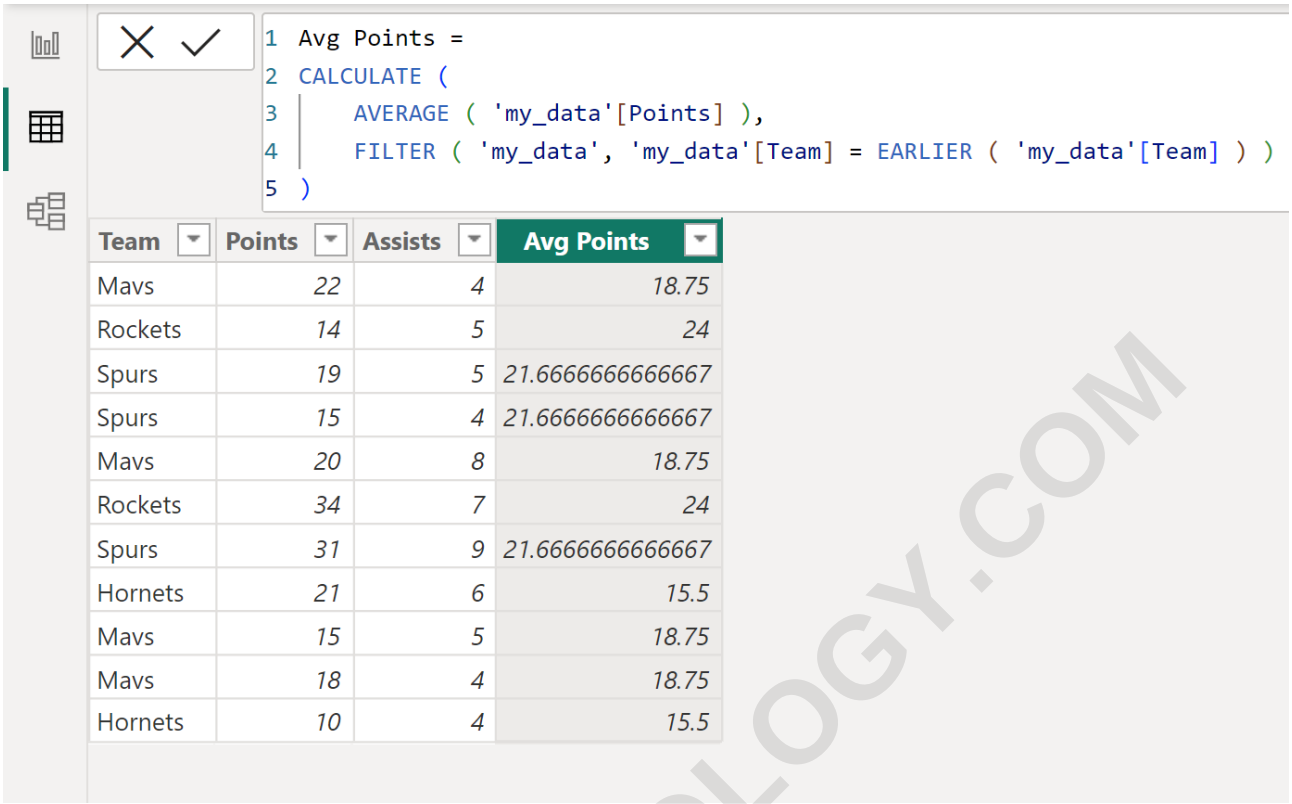
Once the formula bar is active, the next step involves inputting the precise DAX expression. This formula, combining CALCULATE, AVERAGE, and FILTER, executes the conditional average logic previously discussed.

6. Execution and Detailed Breakdown of the DAX Formula

Then type the following comprehensive formula into the formula bar:

```
Avg Points =
CALCULATE (
AVERAGE ( 'my_data' ),
FILTER ( 'my_data', 'my_data' = EARLIER ( 'my_data' ) )
)
```

Upon successful execution, a new column named **Avg Points** will be automatically appended to the **my_data** table. This column dynamically displays the corresponding average points value for every player's team, revealing the calculated average points value for each team:



```
Avg Points =
CALCULATE (
    AVERAGE ( 'my_data'[Points] ),
    FILTER ( 'my_data', 'my_data'[Team] = EARLIER ( 'my_data'[Team] ) )
)
```

Team	Points	Assists	Avg Points
Mavs	22	4	18.75
Rockets	14	5	24
Spurs	19	5	21.6666666666667
Spurs	15	4	21.6666666666667
Mavs	20	8	18.75
Rockets	34	7	24
Spurs	31	9	21.6666666666667
Hornets	21	6	15.5
Mavs	15	5	18.75
Mavs	18	4	18.75
Hornets	10	4	15.5

7. Analyzing and Interpreting the Results

By reviewing the output, we can draw immediate conclusions about the aggregated performance metrics. The averages are calculated based on the underlying data shown in the original table, providing a quick summary statistic for each defined group.

The average points value for players on the **Mavs** team is **18.75**, derived from summing their points and dividing by the count of players (4).
The average points value for players on the **Rockets** team is **24**, based on the total points divided by the team size (3).
The average points value for players on the **Spurs** team is **21.67**, reflecting their collective scoring performance.
The average points value for players on the **Hornets** team is **15.5**, calculated across the two listed team members.

Note: The AVERAGE function in DAX is fundamental to these operations. For complete documentation regarding all facets of the standard AVERAGE function, developers should always refer to the official Microsoft documentation.

8. Further Considerations: AVERAGEX and Context Transition

While the combination of CALCULATE and FILTER successfully implements the **AVERAGE IF** logic as a calculated column, it is important to consider alternative and often more efficient DAX functions, particularly when developing measures. For scenarios where the conditional average needs to be calculated dynamically in the filter context, the `AVERAGEX` function is frequently preferred, as it efficiently handles iteration and expression evaluation.

The following tutorials explain how to perform other common tasks in Power BI: