ALL, ALLEXCEPT, ALLSELECTED, REMOVEFILTERS, KEEPFILTERS for DAX Power BI!

R. Ganesh · Follow

4 min read · Jul 30, 2024

1

`ALLEXCEPT`, `ALLSELECTED`, `REMOVEFILTERS`, and `KEEPFILTERS` to control how filters are applied in your calculations. Here's a detailed explanation and example for each...

Date	Produc	Amoun	Region
01-07-2024	A	10,000	North
01-07-2024	B	15,000	North
01-07-2024	A	5,000	South
01-07-2024	B	20,000	South
02-07-2024	A	12,000	North
02-07-2024	B	10,000	North
02-07-2024	A	7,000	South
02-07-2024	B	25,000	South

1. ALL

Function: Removes all filters from a table or column, effectively returning the entire table or column regardless of any applied filters.

Syntax:

```
ALL(TableName)
ALL(TableName[ColumnName])
```

Example: Suppose you have a table called `Sales` with columns `Region` and `SalesAmount`.

To calculate the total sales amount without any filters applied, you can use:

```
Total Sales All = CALCULATE(SUM(Sales[SalesAmount]), ALL(Sales))
```

This formula ignores any filters on the `Sales` table and calculates the total sales amount for the entire table.

```
Total Sales All = SUMX(ALL(Sales), Sales[Amount])
```

Scenario:

- No filters applied.

Output:

- **Total Sales:** $10,000 + $15,000 + $5,000 + $20,000 + $12,000 + $10,000 + $7,000 + $25,000 = **$84,000**

2. ALLEXCEPT

Function: Removes all filters from a table or column except for the specified columns. Useful when you want to preserve certain filters while ignoring others.

Syntax:

```
ALLEXCEPT(TableName, TableName[ColumnName1], TableName[ColumnName2], ...)
```

Example: To calculate total sales while keeping the filter on `Region`, but ignoring filters on other columns, you can use:

```
Total Sales By Region = CALCULATE(SUM(Sales[SalesAmount]),
ALLEXCEPT(Sales, Sales[Region]))
```

This formula calculates the total sales amount while keeping the filter on the `Region` column but removing filters on other columns.

Scenario:

- **Filter:** `Product = "A"`

Output:

- **Total Sales (Product A only):** $34,000 (summing up sales for Product A across all regions and dates).

3. ALLSELECTED

Function: Returns all the values in a column or table that are currently selected or visible based on the user's selections in visuals, but not affected by filters outside the current context.

Syntax:

```
ALLSELECTED(TableName[ColumnName])
```

Example: If you want to calculate the total sales amount based on the selected regions in a slicer, you can use:

```
Total Sales Selected Regions = CALCULATE(SUM(Sales[SalesAmount]),
ALLSELECTED(Sales[Region]))
```

This formula calculates the total sales for the regions selected in the slicer, considering any filters applied in the visual.

```
Total Sales Selected = SUMX(ALLSELECTED(Sales), Sales[Amount])
```

Scenario:

- **Slicer Filter:** `Date = 2024-07-01` (through a slicer)
- **Visual Filter:** `Region = "North"`

Output:

- **Total Sales (Date = 2024–07–01, Region = "North"):** $25,000 (considering slicer selections and ignoring internal visual filters).

4. REMOVEFILTERS

Function: Removes filters from a table or column. It is similar to `ALL` but is often used for readability and explicit filter removal.

Syntax:

```
REMOVEFILTERS(TableName)
REMOVEFILTERS(TableName[ColumnName])
```

Example: To calculate the total sales amount without any filters on the `Sales` table but preserving slicers on other visuals, you can use:

```
Total Sales Remove Filters = CALCULATE(SUM(Sales[SalesAmount]),
REMOVEFILTERS(Sales))
```

This formula calculates the total sales amount for the entire `Sales` table, disregarding any filters applied to the `Sales` table.

```
Total Sales Remove Region Filter = SUMX(REMOVEFILTERS(Sales[Region]),
Sales[Amount])
```

Scenario:

- **Filter:** `Region = "North"`

Output:

- **Total Sales (Ignoring Region Filter):** $104,000 (total sales across all regions).

5. `KEEPFILTERS`

Purpose: Keeps filters on a specific column or table even if they are overridden by other filter functions.

Syntax:

```
KEEPFILTERS(<table_or_column>)
```

Example: To calculate the total sales amount while explicitly keeping filters applied to the `Category` column:

```
Total Sales Keep Product Filter = CALCULATE(SUM(Sales[Amount]),
KEEPFILTERS(Sales[Product]))
```

This measure ensures that filters on the `Category` column are respected even if other filter functions are applied.

```
Total Sales Keep Category Filter = SUMX(KEEPFILTERS(Sales[Category]),
Sales[Amount])
```

Scenario:

- **Filter:** `Product = "B"`

Output:

- **Total Sales (Product B only):** $70,000 (respecting the filter on Product B).

Summary of Use Cases

- **`ALL`:** Removes all filters from a table or column.
- **`ALLEXCEPT`:** Removes all filters except for specified columns.
- **`ALLSELECTED`:** Returns values based on the user's current selections in visuals.

- **`REMOVEFILTERS`**: Removes filters from a table or column, similar to `ALL`, but can be used for clarity.
- **`KEEPFILTERS`**: Use to explicitly keep filters on certain columns even if other filters might override them.

Measure	Description	Output
Total Sales All	Total sales ignoring all filters	$84,000
Total Sales Except Product	Total sales while keeping filter on Product	$34,000
Total Sales Selected	Total sales considering slicers and visual filters	$25,000
Total Sales Remove Region Filter	Total sales removing filter on Region	$104,000
Total Sales Keep Product Filter	Total sales while keeping filter on Product	$70,000

By understanding and using these DAX functions effectively, you can control filter contexts and perform complex calculations in Power BI.

. . .

Happy Learning !!!

Regards!
Together, we can grow and learn.
Please share this again with your network.

About me

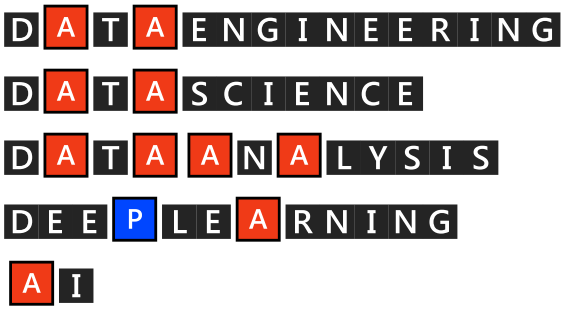
Please do follow my LinkedIn profile link: R GANESH for more update…

Please check out my GitHub project link: https://lnkd.in/gxjKWsXj

Please check out my blogs Medium account link: https://lnkd.in/gDhRarfE

Please check out my latest Post Instagram account for link: https://lnkd.in/gizfkVcy

Please check out any career assistance book slot topmate.io link: https://lnkd.in/gJaF9Dx5



Written by R. Ganesh

Follow

189 Followers

Senior Data Engineer https://www.linkedin.com/in/r-ganesh-a86418155/ For any assistance book slot https://topmate.io/ganesh_r0203.

More from R. Ganesh

R. Ganesh

Different methods to add column in spark DataFrame...

R. Ganesh

Flatten JSON format different methods using Python!

In Apache Spark, there are several methods to add a new column to a DataFrame. Here...

Feb 7 3

Flattening a JSON object can be useful for various data processing tasks, such as...

Jul 1

R. Ganesh

Most important PYSPARK concepts as a Data Engineer mus...

As a Data Engineer working with PySpark, it's crucial to master a variety of concepts to...

Jun 24 121

R. Ganesh

50 recently asked Pyspark Interview Questions Big Data!

Here are 50 PySpark interview questions that are commonly asked in big data roles:

Aug 1

See all from R. Ganesh

Recommended from Medium

Vishal Barvaliya

IN vs EXISTS in SQL

When you're working with SQL, you'll often find yourself needing to filter data based on...

Aug 14 71 1

Attraversiamo.srr

Azure Data Factory (ADF)

Introduction

Jun 14 19

Lists

Staff Picks

717 stories · 1241 saves

Stories to Help You Level-Up at Work

19 stories · 757 saves

Self-Improvement 101

20 stories · 2602 saves

Productivity 101

20 stories · 2239 saves

Feruz Urazaliev

Advanced PySpark Techniques for Data Engineers

This article explores advanced PySpark techniques that are essential for data...

Aug 14 7

Jagadesh Jamjala in Dev Genius

4 Ways in SQL to find Duplicate rows

Duplicate data in databases is a common issue that can have significant impacts on...

Jul 2 172 4

Konstantinos Patronas

SQL: How to find the diff between two record sets

In this article, I will show you how to subtract one result set from another. The database I...

Jun 28 59 2

Shashanka Shekhar

2 easy steps to create Customer Order Dates Difference Table in...

Power BI is a platform that allows you to connect to various data sources, transform...

Aug 11

See more recommendations