Filtering and counting with DAX

DAX IN POWER BI

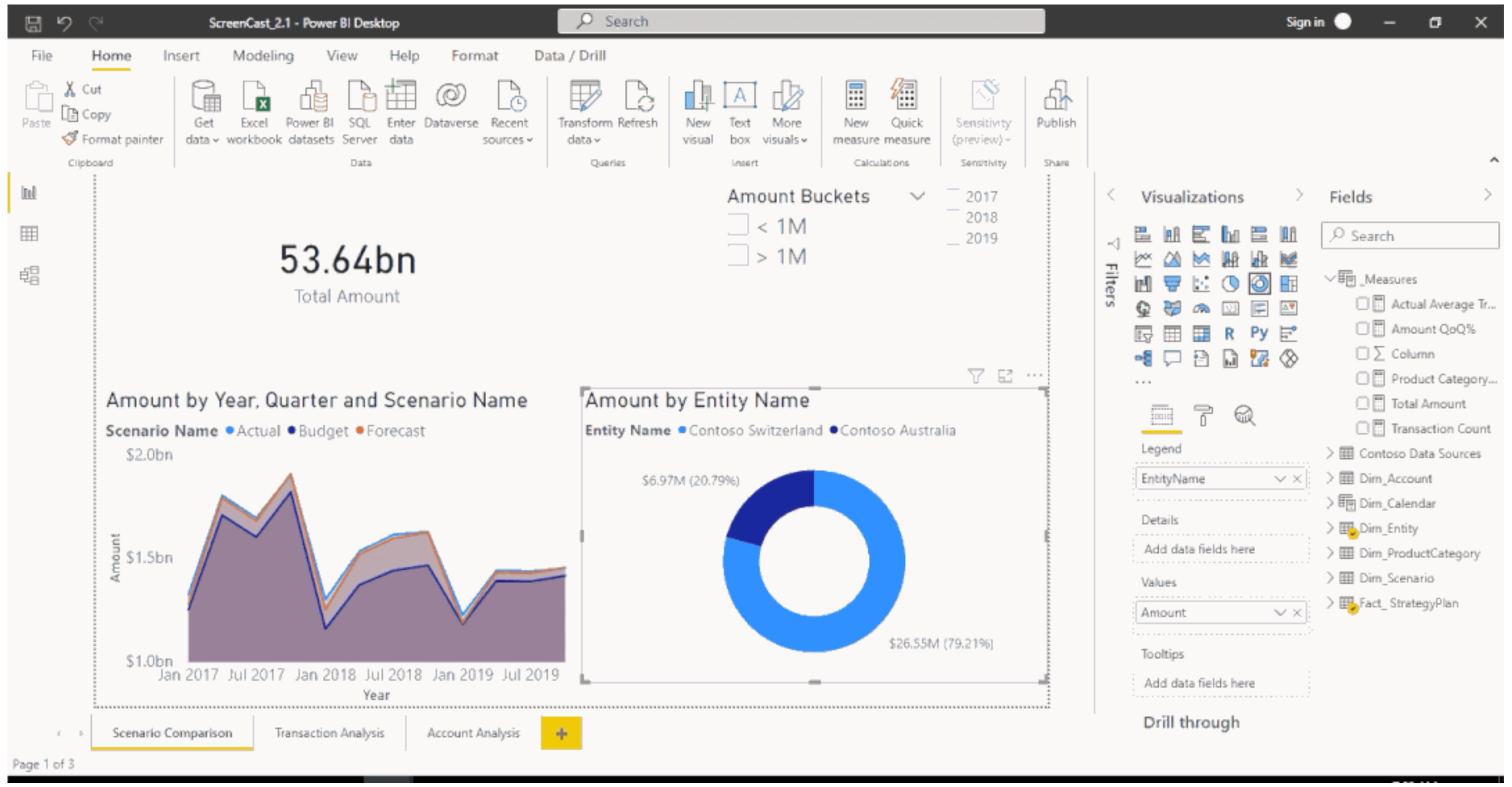
Maarten Van den Broeck
Content Developer at DataCamp

datacamp

Filter functions

- Filters are applied on the filter context
- Filters take precedence over any visual

```
Total Sales = SUM(Orders[Sales])
```



Filter functions

- Filters are applied on the filter context
- Filters take precedence over any visual

```
Total Sales = SUM(Orders[Sales])
```

```
CALCULATE(<expression>,
            <filter1> , [<filter2> [, ...]])
```

- Used with intermediate functions

```
Total Sales ALL = CALCULATE(
                    [Total Sales],
                    ALL(Orders))
```

Region	**Total Sales**
Central	$501,239.89
East	$678,781.24
South	$391,721.91
West	$725,457.82
TOTAL	**$2,297,200.86**

Filter functions

- Filters are applied on the filter context
- Filters take precedence over any visual

```
Total Sales = SUM(Orders[Sales])
```

```
CALCULATE(<expression>,
            <filter1> , [<filter2> [, ...]])
```

- Used with intermediate functions

```
Total Sales ALL = CALCULATE(
                    [Total Sales],
                    ALL(Orders))
```

Region	Total Sales	Total Sales ALL
Central	$501,239.89	$2,297,200.86
East	$678,781.24	$2,297,200.86
South	$391,721.91	$2,297,200.86
West	$725,457.82	$2,297,200.86
TOTAL	**$2,297,200.86**	**$2,297,200.86**

More filter options

- `FILTER(<table>, <filter>)`
    - *Returns a filtered table*

```
Total Sales Chuck =
CALCULATE(
    [Total Sales],
    FILTER(Fact_Orders,
        RELATED(Dim_Sales[Salesperson]) = "Chuck"))
```

More filter options

- `FILTER(<table>, <filter>)`
  - *Returns a filtered table*

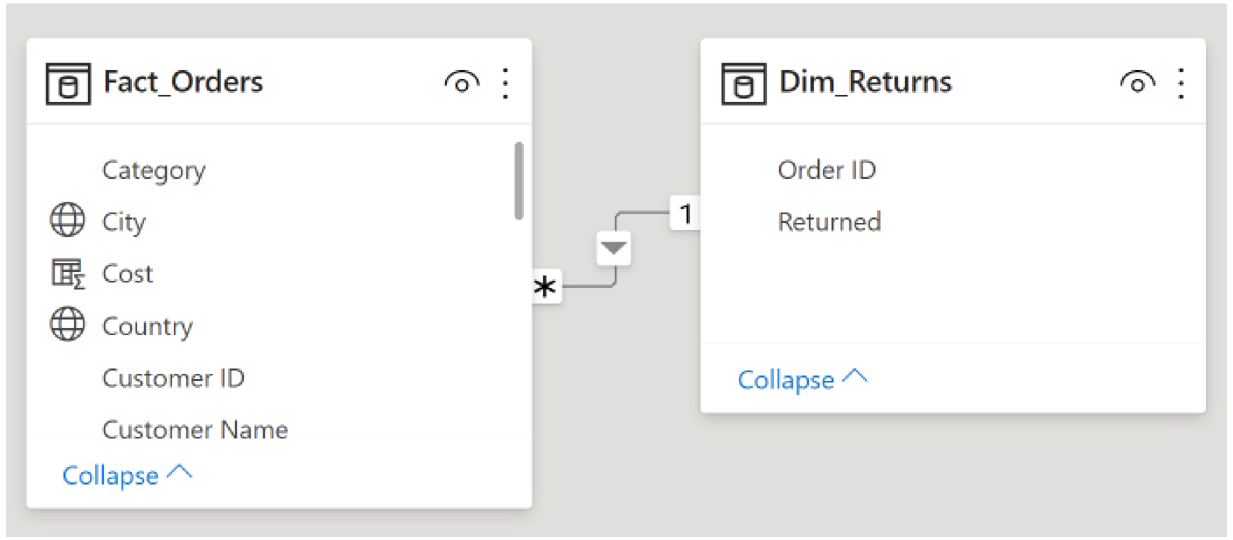
```
Total Sales Chuck =
CALCULATE(
    [Total Sales],
    FILTER(Fact_Orders,
        RELATED(Dim_Sales[Salesperson]) = "Chuck"))
```

Total Sales	Total Sales Chuck
$2,297,200.86	$235,856.05

- `RELATED()` is used to return values from another table

More filter options

- `CROSSFILTER(<col1>, <col2>, <direction>)`
  - *Specifies the cross-filtering direction between two columns*



```
CROSSFILTER(Dim_Returns[Order ID],
            Fact_Orders[Order ID],
            Both)
```

- Overrides relationship direction of data model

The benefits of filtering in DAX

- Improves performance
  - Filter out unnecessary data
  - Define specific relationships between tables
- Reusability
  - Refer to other calculated measures
- More complex computations
  - Concise syntax

Counting

- `COUNT(<column>)`
  - *Returns the amount of rows with numbers, dates, or strings in a column*
- `COUNTA(<column>)`
  - *Returns the amount of rows with numbers, dates, strings, or booleans in a column*
- `COUNTBLANKS(<column>)`
  - *Returns the amount of blank rows*
- `DISTINCTCOUNT(<column>)`
  - *Returns the amount of distinct values in a column*
- `COUNTROWS(<table>)`
  - *Returns the amount of rows with numbers, dates, and strings in a table*

Let's practice!

DAX IN POWER BI

Using different filters with DAX

DAX IN POWER BI

Maarten Van den Broeck
Content Developer at DataCamp

Let's practice!

DAX IN POWER BI

Iterating functions

DAX IN POWER BI

Maarten Van den Broeck
Content Developer at DataCamp

Iterating functions

- Iterate over each row of a given table to perform an expression

`SUMX(<table>, <expression>)` `AVERAGEX(<table>, <expression>)`

- X stands for eXpression
- Allow for advanced calculations specified at each row

Iterating functions: SUMX()

Calculated column example

```
Cost = Fact_Orders[Sales] - Fact_Orders[Profit]
```

```
Total Costs = SUM(Fact_Orders[Cost])
```

Sales	Profit	Cost
$77.88	$3.89	$73.99
$22.72	$10.22	$12.50
...	...	...

Total Costs
$2,569

Iterating functions: SUMX()

Calculated column example

```
Cost = Fact_Orders[Sales] - Fact_Orders[Profit]
```

```
Total Costs = SUM(Fact_Orders[Cost])
```

Sales	Profit	Cost
$77.88	$3.89	$73.99
$22.72	$10.22	$12.50
...	...	...

Total Costs
$2,569

Iterating function example

```
Total Costs SUMX =
SUMX(Fact_Orders,
     Fact_Orders[Sales] - Fact_Orders[Profit])
```

Total Costs SUMX
$2,569

Filtering iterating functions

- Use filter functions, such as FILTER(), to return a filtered table

```
SUMX(
    FILTER(
        <table>,
        <filter>),
    <expression>)
```

```
Total Costs East SUMX =
SUMX(
    FILTER(
        Fact_Orders,
        Fact_Orders[Region] = "East"),
    Fact_Orders[Sales] - Fact_Orders[Profit])
```

Filtering iterating functions

- Use filter functions, such as FILTER(), to return a filtered table

```
SUMX(
    FILTER(
        <table>,
        <filter>),
    <expression>)
```

```
Total Costs East SUMX =
SUMX(
    FILTER(
        Fact_Orders,
        Fact_Orders[Region] = "East"),
    Fact_Orders[Sales] - Fact_Orders[Profit])
```

Region	Total Costs	Total Costs East SUMX
Central	$501,239.89	
East	$678,781.24	$678,781.24
South	$391,721.91	
West	$725,457.82	
TOTAL	**$2,297,200.86**	**$678,781.24**

Iterating functions: RANKX()

```
RANKX(
    <table>,
    <expression>)
```

- Rank regions by total costs

```
Total Costs RANKX =
RANKX(
    ALL(Dim_Sales[Region]),
    [Total Costs])
```

- Use `ALL()` to evaluate all rows from the dimension table

Iterating functions: RANKX()

```
RANKX(
    <table>,
    <expression>)
```

- Rank regions by total costs

```
Total Costs RANKX =
RANKX(
    ALL(Dim_Sales[Region]),
    [Total Costs])
```

- Use `ALL()` to evaluate all rows from the dimension table

Region	Total Costs	Total Costs RANKX
Central	$725,457.82	1
East	$678,781.24	2
South	$501,239.89	3
West	$391,721.91	4

Operators in DAX

COMPARISON OPERATORS

Operator	Meaning
=	Equal to
==	Strict equal to
>	Greater than
<	Smaller than
>=	Greater than or equal to
<=	Smaller than or equal to
<>	Not equal to

Operators in DAX

COMPARISON OPERATORS

Operator	Meaning
=	Equal to
==	Strict equal to
>	Greater than
<	Smaller than
>=	Greater than or equal to
<=	Smaller than or equal to
<>	Not equal to

TEXT OPERATOR

Operator	Meaning	Example
&	Concatenates text values	[City]&", "& [State]

Operators in DAX

COMPARISON OPERATORS

Operator	Meaning
=	Equal to
==	Strict equal to
>	Greater than
<	Smaller than
>=	Greater than or equal to
<=	Smaller than or equal to
<>	Not equal to

TEXT OPERATOR

Operator	Meaning	Example
&	Concatenates text values	[City]&", "& [State]

LOGICAL OPERATORS

Operator	Meaning	Example
&&	`AND` condition	([City] = "Bru") && ([Return] = "Yes"))
||	`OR` condition	([City] = "Bru") || ([Return] = "Yes"))
IN { }	`OR` condition for each row	Product[Color] IN {"Red", "Blue", "Gold"}

Lesson[Knowledge] IN {"Poor", "Great", "Awesome!"}

DAX IN POWER BI

Iterating functions in Power BI

DAX IN POWER BI

Full Name
Instructor

datacamp

Let's practice!

DAX IN POWER BI