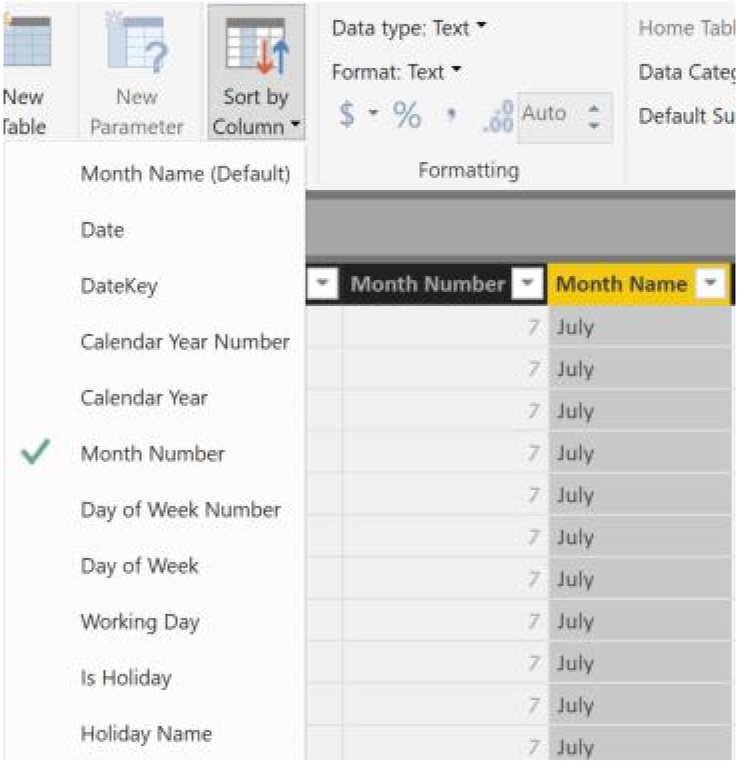
Side effects of the Sort By Column setting in DAX

The Sort By Column feature in Power BI causes side effects that are important to know when writing a DAX formula. This article explains these side effects and how to write correct DAX code to avoid getting incorrect results.

The Sort By Column feature sorts the values of a column using the values from another column. For example, a developer might want to display months by their name, in a sequence like January, February and March. To this end, it is necessary to apply the Sort By Column property to the Month Name column, indicating that the Month Number column must be used to define the sort order of month names. This avoids an alphabetical sort of the Month Name values.



Data type: Text

Format: Text

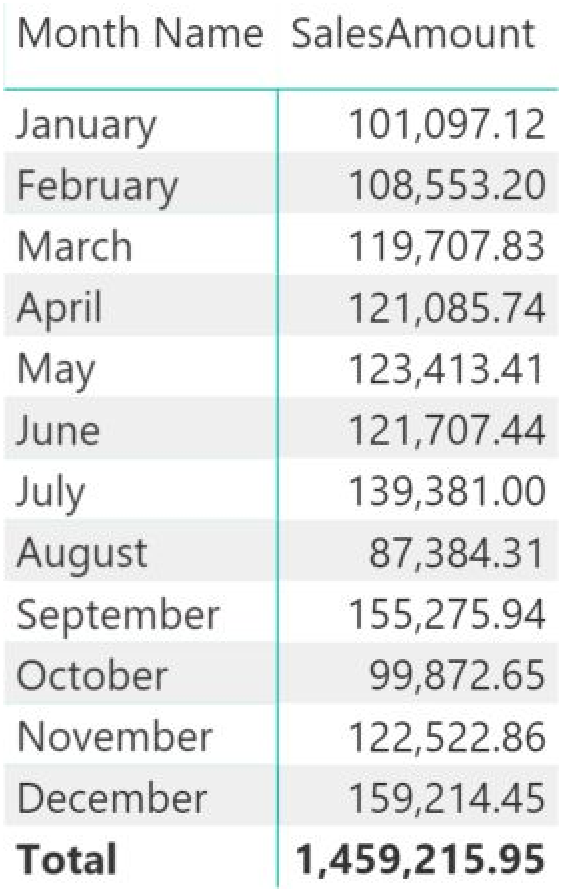
Month Number	Month Name
7	July
7	July
7	July
7	July
7	July
7	July
7	July
7	July
7	July
7	July
7	July

This setting causes different side effects depending on the client used to query the model. For example, Power BI generates queries in DAX to populate a report, whereas an Excel PivotTable generates queries in MDX. The different languages used by the client tools might have an impact on the result of a measure defined in the model. The same measure might show different results in similar reports created using different tools. In order to understand why this happens and how to avoid these different results, it is necessary to know how MDX and DAX work.

Column sort order in DAX and MDX

A query in MDX automatically inherits the correct column sort order from the data model; the result of an MDX query is always sorted according to the Sort By Column setting. However, DAX does not have any implicit sort order for the columns other than the natural sort order of the underlying data type. For this reason, a DAX query must always specify the sort order in an ORDER BY condition – similarly to a query in SQL. Because DAX requires for a column used in ORDER BY to be part of the query result, a Power BI visual that sorts a column always generates a query that includes at least two columns: the column requested in the report and the underlying column used in the Sort By Column setting. In other words, a Power BI visual showing data by Month must generate a query that contains both Month Name and Month Number, whereas MDX only requires the Month Name column.

For example, consider the following report in Power BI.



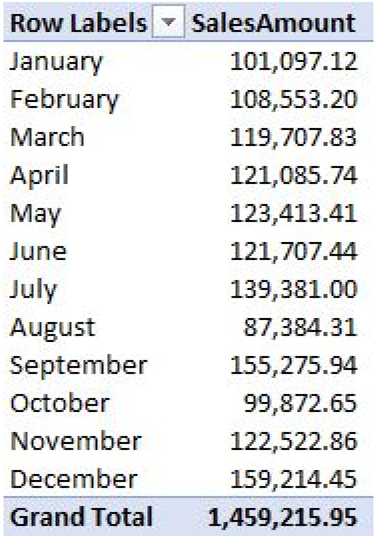
Month Name	SalesAmount
January	101,097.12
February	108,553.20
March	119,707.83
April	121,085.74
May	123,413.41
June	121,707.44
July	139,381.00
August	87,384.31
September	155,275.94
October	99,872.65
November	122,522.86
December	159,214.45
Total	**1,459,215.95**

The query generated by DAX is the following:

```
EVALUATE
TOPN (
    502,
    SUMMARIZECOLUMNS (
        ROLLUPADDISSUBTOTAL (
            ROLLUPGROUP (
                'Date'[Month Name],
                'Date'[Month Number]
            ), "IsGrandTotalRowTotal"
        ),
        "SalesAmount", 'Sales'[SalesAmount]
    ),
    [IsGrandTotalRowTotal], 0,
    'Date'[Month Number], 1,
    'Date'[Month Name], 1
)
ORDER BY
    [IsGrandTotalRowTotal] DESC,
    'Date'[Month Number],
    'Date'[Month Name]
```

Though the report only shows the Month Name, the DAX query includes both Month Name and Month Number. The result is correctly sorted by Month Number, indeed.

Now, consider the equivalent PivotTable in Excel connected to the same data model, for example using the Analyze In Excel feature at powerbi.com.



Row Labels ▾	SalesAmount
January	101,097.12
February	108,553.20
March	119,707.83
April	121,085.74
May	123,413.41
June	121,707.44
July	139,381.00
August	87,384.31
September	155,275.94
October	99,872.65
November	122,522.86
December	159,214.45
Grand Total	**1,459,215.95**

The query generated by MDX is the following:

```
SELECT
NON EMPTY
    Hierarchize(
        { DrilldownLevel(
            { [Date].[Month Name].[All] },
            ,,
            INCLUDE_CALC_MEMBERS
        ) }
    )
    DIMENSION PROPERTIES
        PARENT_UNIQUE_NAME,
        HIERARCHY_UNIQUE_NAME
    ON COLUMNS
FROM [Model]
```

The MDX query only includes the Month Name. It is not necessary to specify the Month Number, because MDX guarantees that the Month Name attribute is already sorted by Month Number in the result.

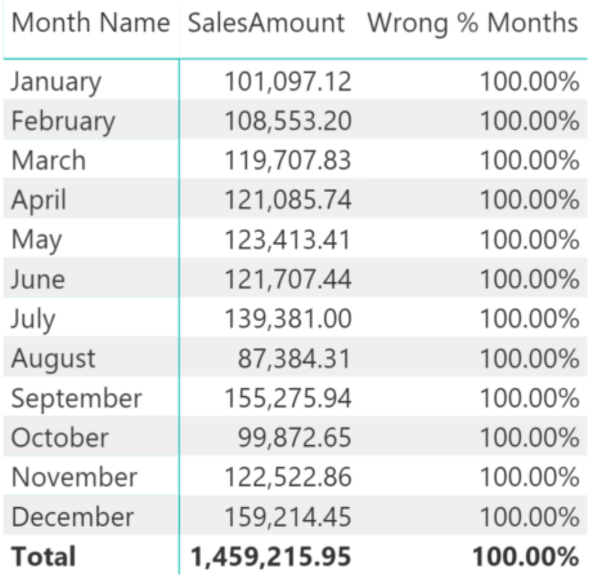
Side effects of Sort By Column in DAX queries

The presence of the two columns also causes side effects on the filter context. For example, consider the following measures:

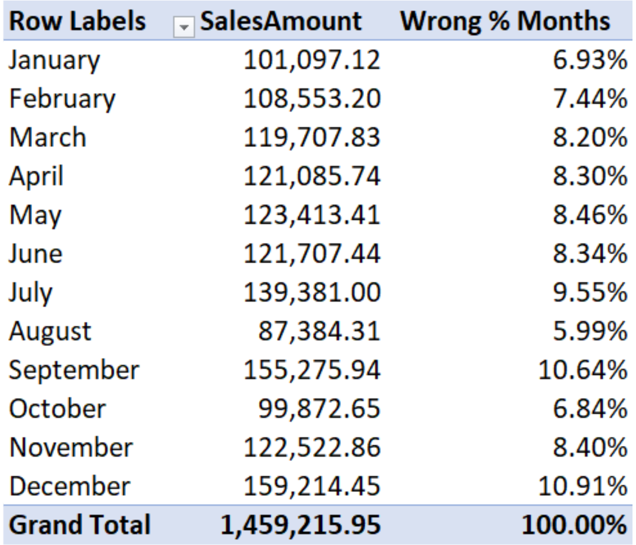
```
FilterMonthName :=
ISFILTERED ( 'Date'[Month Name] )

FilterMonthNumber :=
ISFILTERED ( 'Date'[Month Number] )
```

```
Wrong % Months :=
DIVIDE (
    [SalesAmount],
    CALCULATE (
        [SalesAmount],
        ALL ( 'Date'[Month Name] )
    )
)
```

By comparing the results of similar reports in both Power BI and Excel it is possible to notice several differences.

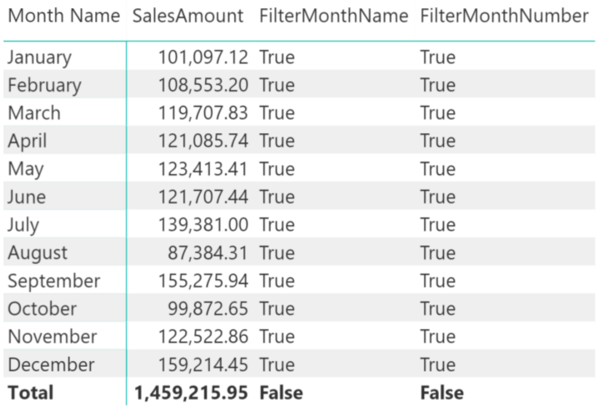


Month Name	SalesAmount	Wrong % Months
January	101,097.12	100.00%
February	108,553.20	100.00%
March	119,707.83	100.00%
April	121,085.74	100.00%
May	123,413.41	100.00%
June	121,707.44	100.00%
July	139,381.00	100.00%
August	87,384.31	100.00%
September	155,275.94	100.00%
October	99,872.65	100.00%
November	122,522.86	100.00%
December	159,214.45	100.00%
Total	**1,459,215.95**	**100.00%**

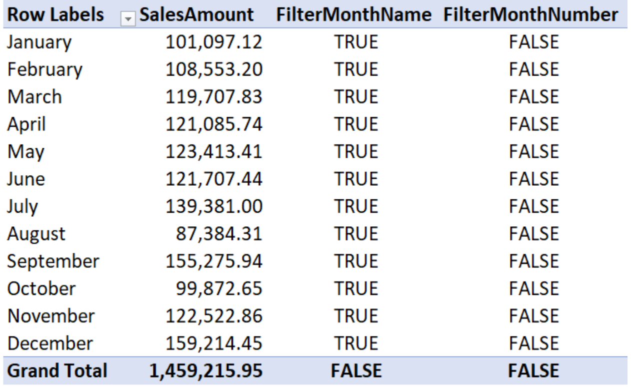


Row Labels	SalesAmount	Wrong % Months
January	101,097.12	6.93%
February	108,553.20	7.44%
March	119,707.83	8.20%
April	121,085.74	8.30%
May	123,413.41	8.46%
June	121,707.44	8.34%
July	139,381.00	9.55%
August	87,384.31	5.99%
September	155,275.94	10.64%
October	99,872.65	6.84%
November	122,522.86	8.40%
December	159,214.45	10.91%
Grand Total	**1,459,215.95**	**100.00%**

The Wrong % Months measure does not work in Power BI (left side) as it always returns 100%, whereas it works correctly in Excel (right side).



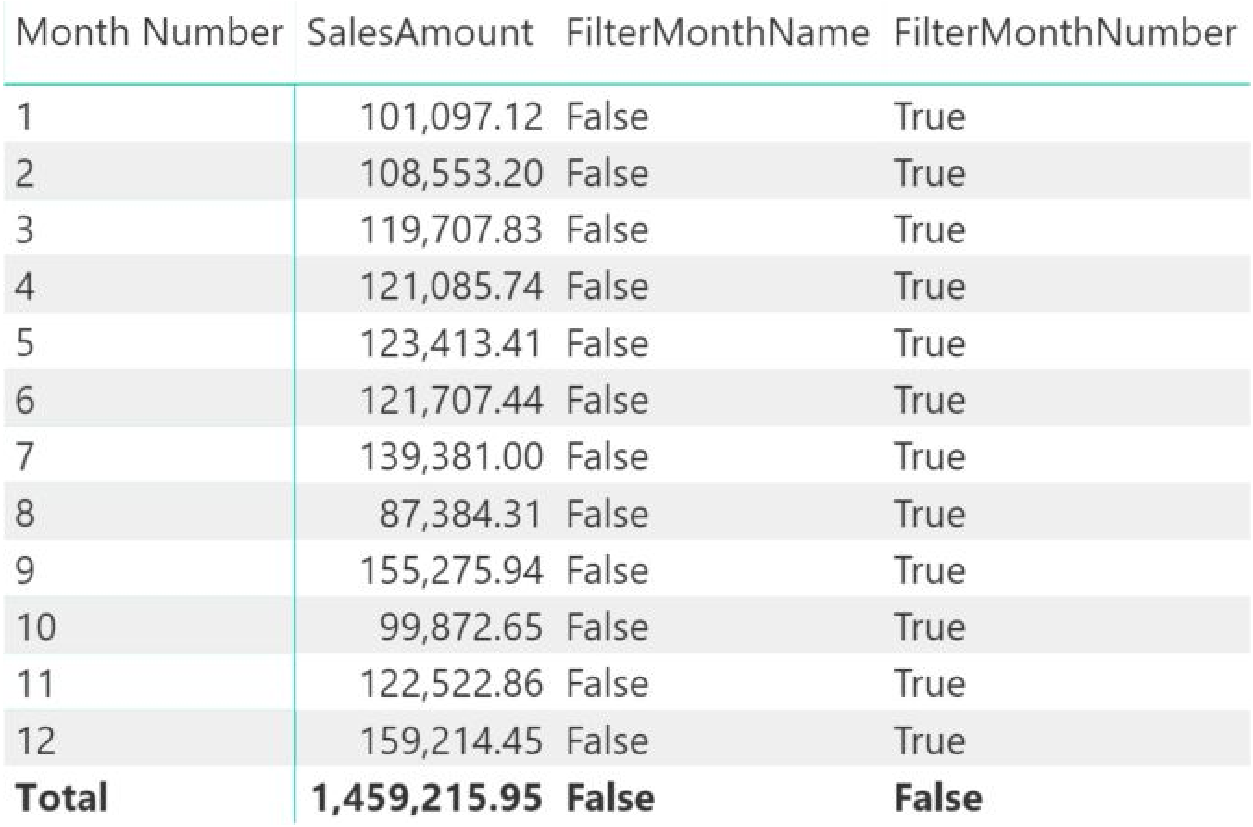
Month Name	SalesAmount	FilterMonthName	FilterMonthNumber
January	101,097.12	True	True
February	108,553.20	True	True
March	119,707.83	True	True
April	121,085.74	True	True
May	123,413.41	True	True
June	121,707.44	True	True
July	139,381.00	True	True
August	87,384.31	True	True
September	155,275.94	True	True
October	99,872.65	True	True
November	122,522.86	True	True
December	159,214.45	True	True
Total	**1,459,215.95**	**False**	**False**



Row Labels	SalesAmount	FilterMonthName	FilterMonthNumber
January	101,097.12	TRUE	FALSE
February	108,553.20	TRUE	FALSE
March	119,707.83	TRUE	FALSE
April	121,085.74	TRUE	FALSE
May	123,413.41	TRUE	FALSE
June	121,707.44	TRUE	FALSE
July	139,381.00	TRUE	FALSE
August	87,384.31	TRUE	FALSE
September	155,275.94	TRUE	FALSE
October	99,872.65	TRUE	FALSE
November	122,522.86	TRUE	FALSE
December	159,214.45	TRUE	FALSE
Grand Total	**1,459,215.95**	**FALSE**	**FALSE**

Moreover, the ISFILTERED function provides different results for the FilterMonthNumber column in the two clients: Power BI always returns TRUE when a Month Name is selected (left side), whereas Excel returns FALSE in a similar report. The reason is that DAX includes both columns in the query – this generates a different filter context to evaluate the measures compared to the one generated by MDX.

Finally, when a report uses the Month Number column without Month Name, the presence of Sort By Column on Month Name does not cause any side effects. The following Power BI report shows that only the FilterMonthNumber measure returns TRUE when the Month Number is used in the rows of a matrix, without affecting the FilterMonthName measure.



Month Number	SalesAmount	FilterMonthName	FilterMonthNumber
1	101,097.12	False	True
2	108,553.20	False	True
3	119,707.83	False	True
4	121,085.74	False	True
5	123,413.41	False	True
6	121,707.44	False	True
7	139,381.00	False	True
8	87,384.31	False	True
9	155,275.94	False	True
10	99,872.65	False	True
11	122,522.86	False	True
12	159,214.45	False	True
Total	**1,459,215.95**	**False**	**False**

Golden rules for measures using Sort By Column

In order for a measure that manipulates a column with a Sort By Column setting to act predictably, it is better to always include the two columns involved in the Sort By Column setting whenever a filter is removed or manipulated.

Thus, instead of writing a filter test over a single column:

```
ISFILTERED ( 'Date'[Month] )
ISFILTERED ( 'Date'[Month Number] )
```

It is better to write a single expression including the two columns, such as:

```
ISFILTERED ( 'Date'[Month] ) || ISFILTERED ( 'Date'[Month Number] )
```

Whenever there is a filter removal over a single column:

```
ALL ( 'Date'[Month] )
ALL ( 'Date'[Month Number] )
```

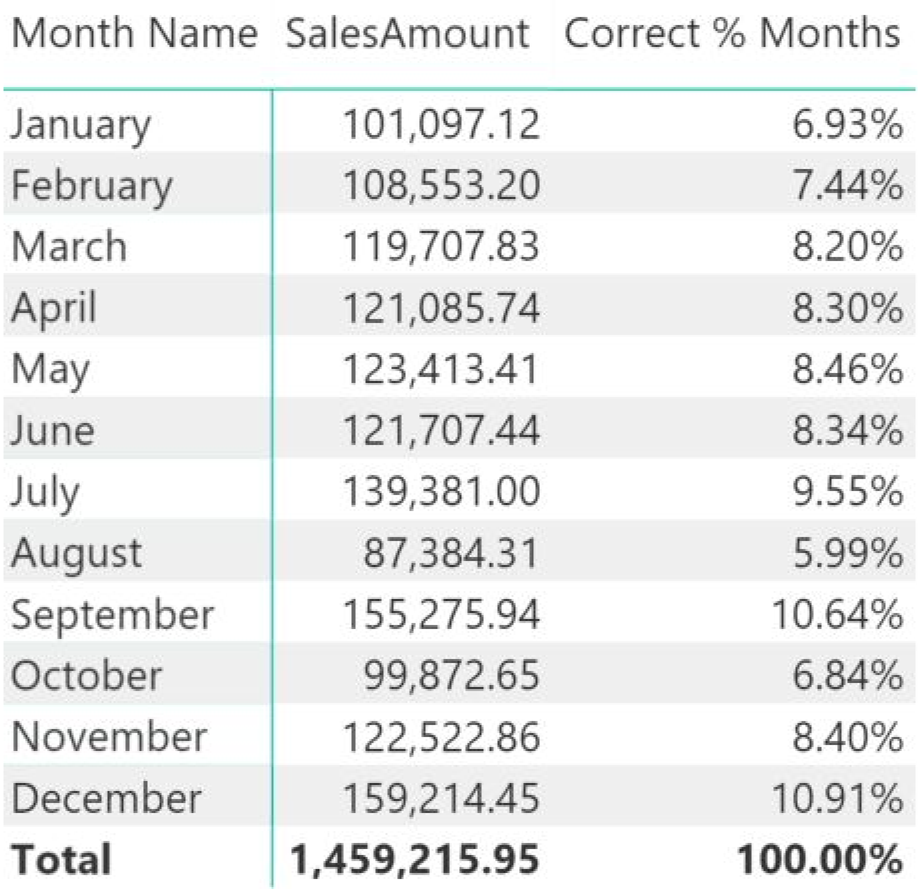
It is better to write a single filter including both columns:

```
ALL ( 'Date'[Month], 'Date'[Month Number] )
```

Rewritten as per these best practices, the previous percentage measure appears as follows:

```
Correct % Months :=
DIVIDE (
    [SalesAmount],
    CALCULATE (
        [SalesAmount],
        ALL ( 'Date'[Month Name], 'Date'[Month Number] )
    )
)
```

The Power BI report using this last measure shows the correct percentage by month when Month Name is used in the matrix rows.



Month Name	SalesAmount	Correct % Months
January	101,097.12	6.93%
February	108,553.20	7.44%
March	119,707.83	8.20%
April	121,085.74	8.30%
May	123,413.41	8.46%
June	121,707.44	8.34%
July	139,381.00	9.55%
August	87,384.31	5.99%
September	155,275.94	10.64%
October	99,872.65	6.84%
November	122,522.86	8.40%
December	159,214.45	10.91%
Total	**1,459,215.95**	**100.00%**

Conclusions

When a DAX expression removes a filter from a column that has the Sort By Column setting active, it is a best practice to always include both columns in the ALL function. In order to test whether a column is filtered, it is a best practice to include two ISFILTERED functions in an OR condition, testing both columns involved in the Sort By Column setting.