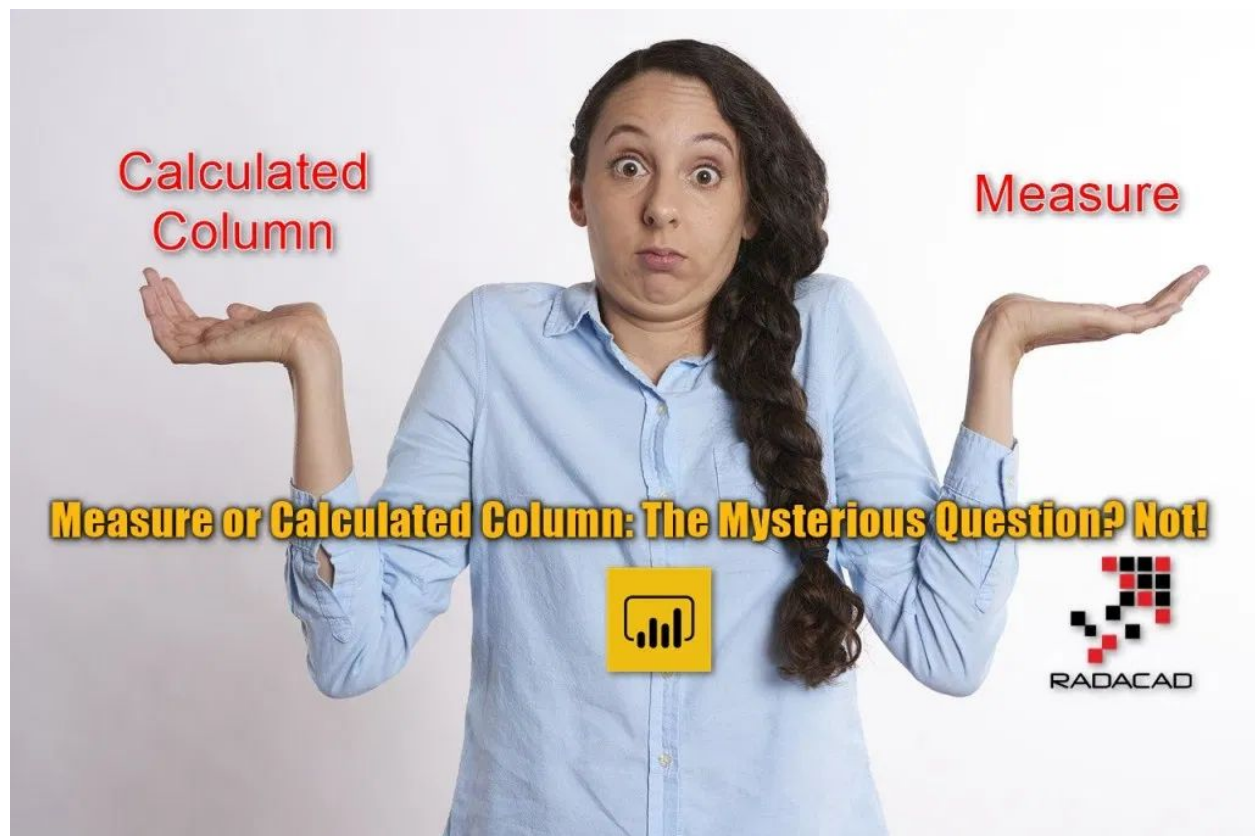


Measure vs Calculated Column: The Mysterious Question? Not!



what is the difference between Measure and Calculated Column? What situation should we use each of these? and on the other hand, what is the difference of creating column here, or in Power Query? I have written previously about “M or DAX, that is the question” which explains situations that you need to use Power Query or DAX. In this post, I’m going to explain what is the difference between DAX Calculated Column and Measure.

Read this If You have any of Questions Below

This blog post is written for you if you want to understand the difference between Calculated Column and Measure in DAX, and have any of below questions;

- What is Calculated Column?
- What is Measure?
- When should I write a calculated column or measure?

- What is their difference in Performance?
- What are operations that I cannot do with these?
- and many other questions about the difference between these two types of calculations in DAX.

What is a Calculated Column?

To understand the difference between these two types of calculation, it is necessary to understand how these are working one by one. **Calculated Column is a column like any other columns, created in the table.** However, the result of a calculated column is coming from calculating an expression (DAX). **usually, calculated column leverages a DAX expression that applies to every row in the dataset, and the result of that will be stored in the new column.**

Example: Profit as a calculated column

Consider a table that we have sales and costs information in it. Calculating Profit in such table would be simply deducting costs from sales for every row. So this basically would be a calculated column.

Profit = FactInternetSales[SalesAmount] - FactInternetSales[TotalProductCost]

OrderDate	ProductStandardCost	TotalProductCost	SalesAmount	TaxAmt	Freight	CarrierTrackingNumber	CustomerPONumber	OrderDate	DueDate	ShipDate	Profit
1/6/2007 12:00:00 AM	1.8663	1.8663	4.99	0.3992	0.1248			1/6/2007 12:00:00 AM	13/6/2007 12:00:00 AM	8/6/2007 12:00:00 AM	3.1237
2/6/2007 12:00:00 AM	1.8663	1.8663	4.99	0.3992	0.1248			2/6/2007 12:00:00 AM	14/6/2007 12:00:00 AM	9/6/2007 12:00:00 AM	3.1237
3/6/2007 12:00:00 AM	1.8663	1.8663	4.99	0.3992	0.1248			3/6/2007 12:00:00 AM	15/6/2007 12:00:00 AM	10/6/2007 12:00:00 AM	3.1237
4/6/2007 12:00:00 AM	1.8663	1.8663	4.99	0.3992	0.1248			4/6/2007 12:00:00 AM	16/6/2007 12:00:00 AM	11/6/2007 12:00:00 AM	3.1237
5/6/2007 12:00:00 AM	1.8663	1.8663	4.99	0.3992	0.1248			5/6/2007 12:00:00 AM	17/6/2007 12:00:00 AM	12/6/2007 12:00:00 AM	3.1237
6/6/2007 12:00:00 AM	1.8663	1.8663	4.99	0.3992	0.1248			6/6/2007 12:00:00 AM	18/6/2007 12:00:00 AM	13/6/2007 12:00:00 AM	3.1237
7/6/2007 12:00:00 AM	1.8663	1.8663	4.99	0.3992	0.1248			7/6/2007 12:00:00 AM	19/6/2007 12:00:00 AM	14/6/2007 12:00:00 AM	3.1237
8/6/2007 12:00:00 AM	1.8663	1.8663	4.99	0.3992	0.1248			8/6/2007 12:00:00 AM	20/6/2007 12:00:00 AM	15/6/2007 12:00:00 AM	3.1237
9/6/2007 12:00:00 AM	1.8663	1.8663	4.99	0.3992	0.1248			9/6/2007 12:00:00 AM	21/6/2007 12:00:00 AM	16/6/2007 12:00:00 AM	3.1237
10/6/2007 12:00:00 AM	1.8663	1.8663	4.99	0.3992	0.1248			10/6/2007 12:00:00 AM	22/6/2007 12:00:00 AM	17/6/2007 12:00:00 AM	3.1237
11/6/2007 12:00:00 AM	1.8663	1.8663	4.99	0.3992	0.1248			11/6/2007 12:00:00 AM	23/6/2007 12:00:00 AM	18/6/2007 12:00:00 AM	3.1237
12/6/2007 12:00:00 AM	1.8663	1.8663	4.99	0.3992	0.1248			12/6/2007 12:00:00 AM	24/6/2007 12:00:00 AM	19/6/2007 12:00:00 AM	3.1237
13/6/2007 12:00:00 AM	1.8663	1.8663	4.99	0.3992	0.1248			13/6/2007 12:00:00 AM	25/6/2007 12:00:00 AM	20/6/2007 12:00:00 AM	3.1237
14/6/2007 12:00:00 AM	1.8663	1.8663	4.99	0.3992	0.1248			14/6/2007 12:00:00 AM	26/6/2007 12:00:00 AM	21/6/2007 12:00:00 AM	3.1237
15/6/2007 12:00:00 AM	1.8663	1.8663	4.99	0.3992	0.1248			15/6/2007 12:00:00 AM	27/6/2007 12:00:00 AM	22/6/2007 12:00:00 AM	3.1237
16/6/2007 12:00:00 AM	1.8663	1.8663	4.99	0.3992	0.1248			16/6/2007 12:00:00 AM	28/6/2007 12:00:00 AM	23/6/2007 12:00:00 AM	3.1237
17/6/2007 12:00:00 AM	1.8663	1.8663	4.99	0.3992	0.1248			17/6/2007 12:00:00 AM	29/6/2007 12:00:00 AM	24/6/2007 12:00:00 AM	3.1237
18/6/2007 12:00:00 AM	1.8663	1.8663	4.99	0.3992	0.1248			18/6/2007 12:00:00 AM	30/6/2007 12:00:00 AM	25/6/2007 12:00:00 AM	3.1237
19/6/2007 12:00:00 AM	1.8663	1.8663	4.99	0.3992	0.1248			19/6/2007 12:00:00 AM	1/7/2007 12:00:00 AM	26/6/2007 12:00:00 AM	3.1237
20/6/2007 12:00:00 AM	1.8663	1.8663	4.99	0.3992	0.1248			20/6/2007 12:00:00 AM	2/7/2007 12:00:00 AM	27/6/2007 12:00:00 AM	3.1237
21/6/2007 12:00:00 AM	1.8663	1.8663	4.99	0.3992	0.1248			21/6/2007 12:00:00 AM	3/7/2007 12:00:00 AM	28/6/2007 12:00:00 AM	3.1237

Expression:

Profit = FactInternetSales[SalesAmount] - FactInternetSales[TotalProductCost]

Row by Row Calculation: Row Context

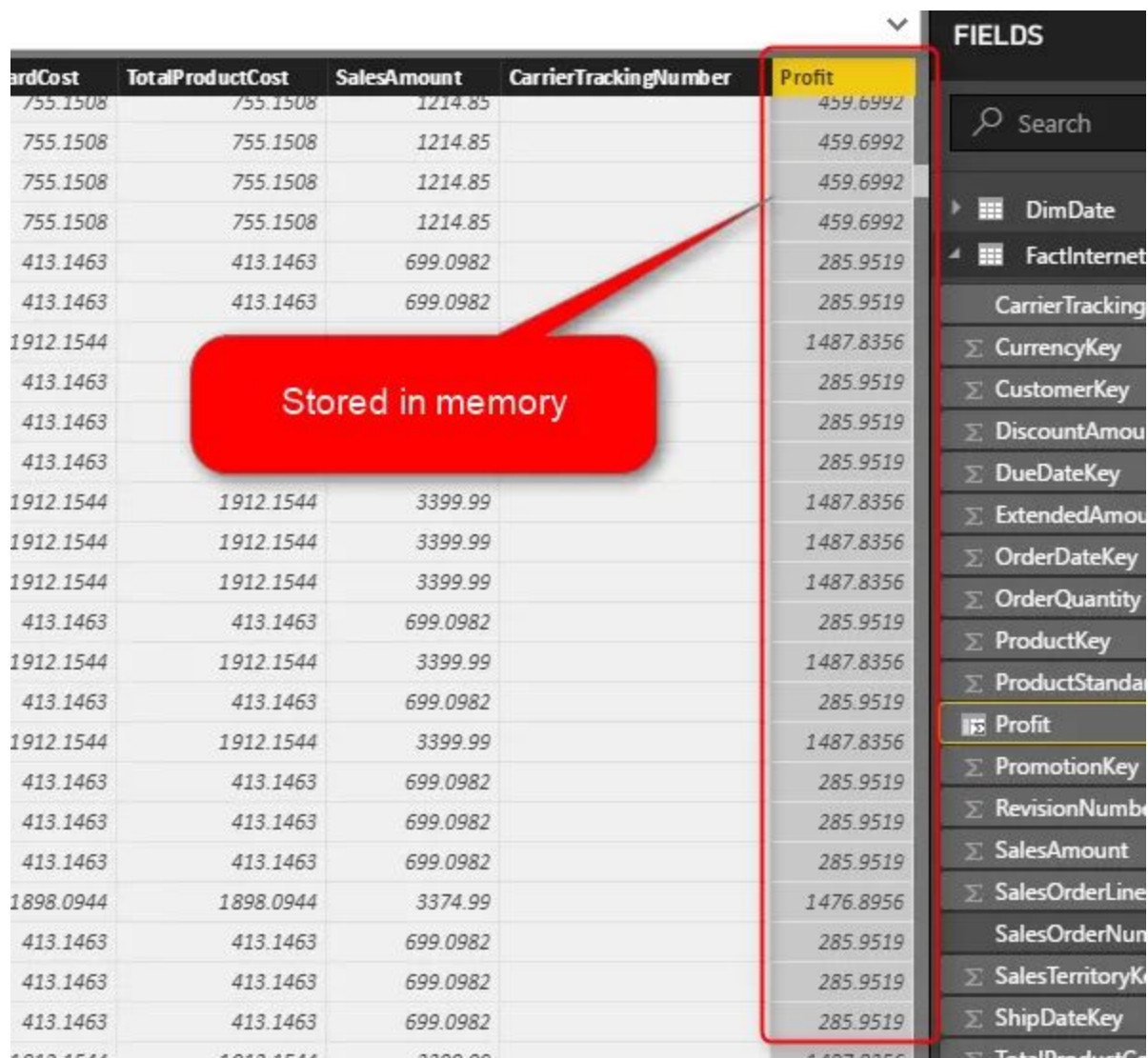
One of the very important concepts about the calculation that you apply in Calculated Column (In the majority of the cases, not always); is that the calculation in one row at a time, or in other words; row by row calculation. In below table; you can see the calculation result for every row stored into the new column;

	TotalProductCost	SalesAmount	CarrierTrackingNumber	Profit
608	755.1508	1214.85		459.6992
608	755.1508	1214.85		459.6992
608	755.1508	1214.85		459.6992
608	755.1508	1214.85		459.6992
663	413.1463	699.0982		285.9519
663	413.1463	699.0982		285.9519
644	1912.1544	3399.99		1487.8356
663	413.1463	699.0982		285.9519
663	413.1463	699.0982		285.9519
663	413.1463	699.0982		285.9519
644	1912.1544	3399.99		1487.8356
644	1912.1544	3399.99		1487.8356
644	1912.1544	3399.99		1487.8356
663	413.1463	699.0982		285.9519
644	1912.1544	3399.99		1487.8356
663	413.1463	699.0982		285.9519
644	1912.1544	3399.99		1487.8356
663	413.1463	699.0982		285.9519
663	413.1463	699.0982		285.9519
663	413.1463	699.0982		285.9519

Row by row calculation called Row Context in DAX terminologies.

Stored in Memory

Calculated Column stores values in the memory, like any other columns. the calculation happens at Refresh time, and the result will be stored in the memory.



The screenshot shows a table with the following columns: HardCost, TotalProductCost, SalesAmount, CarrierTrackingNumber, and Profit. The Profit column is highlighted in yellow. A red callout bubble points to the Profit column with the text "Stored in memory".

HardCost	TotalProductCost	SalesAmount	CarrierTrackingNumber	Profit
755.1508	755.1508	1214.85		459.6992
755.1508	755.1508	1214.85		459.6992
755.1508	755.1508	1214.85		459.6992
755.1508	755.1508	1214.85		459.6992
413.1463	413.1463	699.0982		285.9519
413.1463	413.1463	699.0982		285.9519
1912.1544				1487.8356
413.1463				285.9519
413.1463				285.9519
413.1463				285.9519
1912.1544	1912.1544	3399.99		1487.8356
1912.1544	1912.1544	3399.99		1487.8356
1912.1544	1912.1544	3399.99		1487.8356
413.1463	413.1463	699.0982		285.9519
1912.1544	1912.1544	3399.99		1487.8356
413.1463	413.1463	699.0982		285.9519
1912.1544	1912.1544	3399.99		1487.8356
413.1463	413.1463	699.0982		285.9519
413.1463	413.1463	699.0982		285.9519
413.1463	413.1463	699.0982		285.9519
1898.0944	1898.0944	3374.99		1476.8956
413.1463	413.1463	699.0982		285.9519
413.1463	413.1463	699.0982		285.9519
413.1463	413.1463	699.0982		285.9519
413.1463	413.1463	699.0982		285.9519

On the right side, the "FIELDS" pane is visible, showing a list of columns. The "Profit" column is highlighted in yellow.

This means that more calculated memory you have, more memory consumption you will end up with, and your refresh time will be longer as well. However, many calculations are very simple, so your refresh time might not be affected too much.

Calculated Column highlights

Based on above explanations, here are highlights of a calculated column;

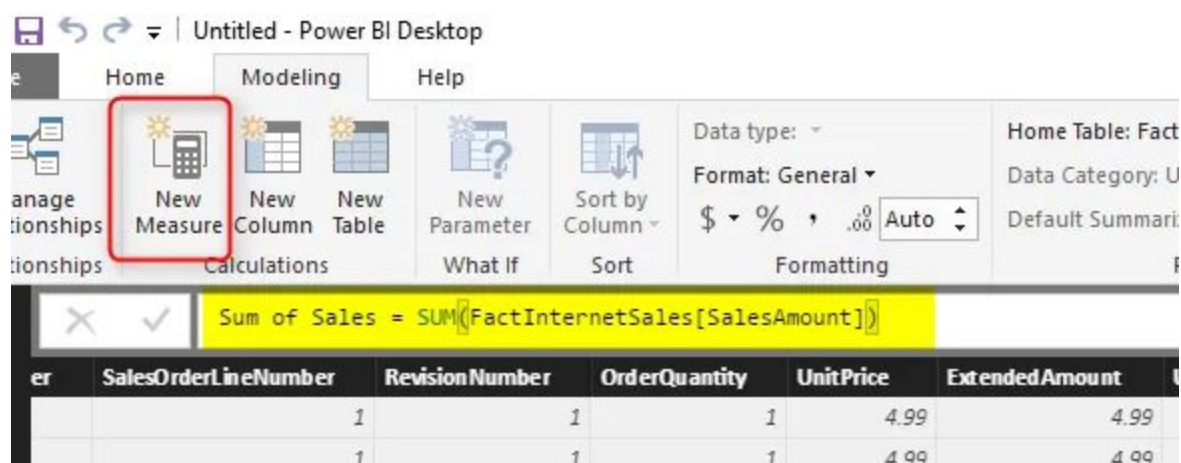
- Row by row calculation: Row Context (usually, not always)
- Stored in the memory (consumes RAM)
- calculated at the time of refreshing the report (either scheduled basis, or manual)

What is Measure?

A measure is usually a calculation that works on an aggregated level basis. This aggregation can be as simple as a sum of sales or can be a little bit more complex, such as calculating monthly average sales in a rolling 12 months period. Measures have dynamic nature, they affect on a subset of data from one or more table. Hence, the subset of data can be changed through the filters applied in the Power BI Report, then the calculation will have to be evaluated dynamically. So Measures are not pre-calculated, they will be calculated on the fly when adding it in the report.

Example: Sum of Sales

Measures are usually aggregations. A very simple aggregation we can use as an example is a sum of sales.

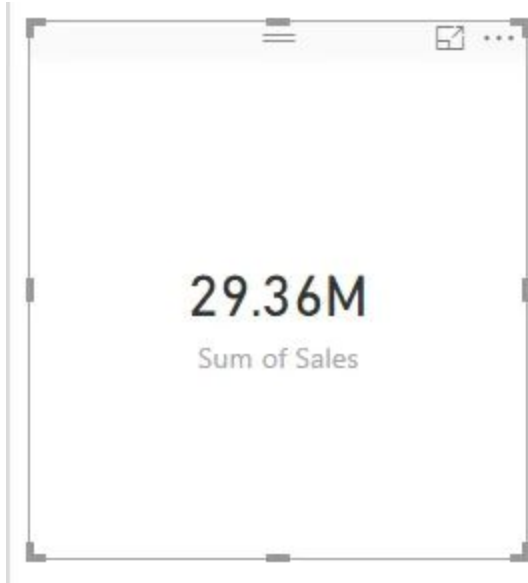


Aggregation can be done with a number of functions in DAX, such as Sum, SumX, Average, Calculate, and heaps of other aggregation functions. Now, let's answer the most important question:

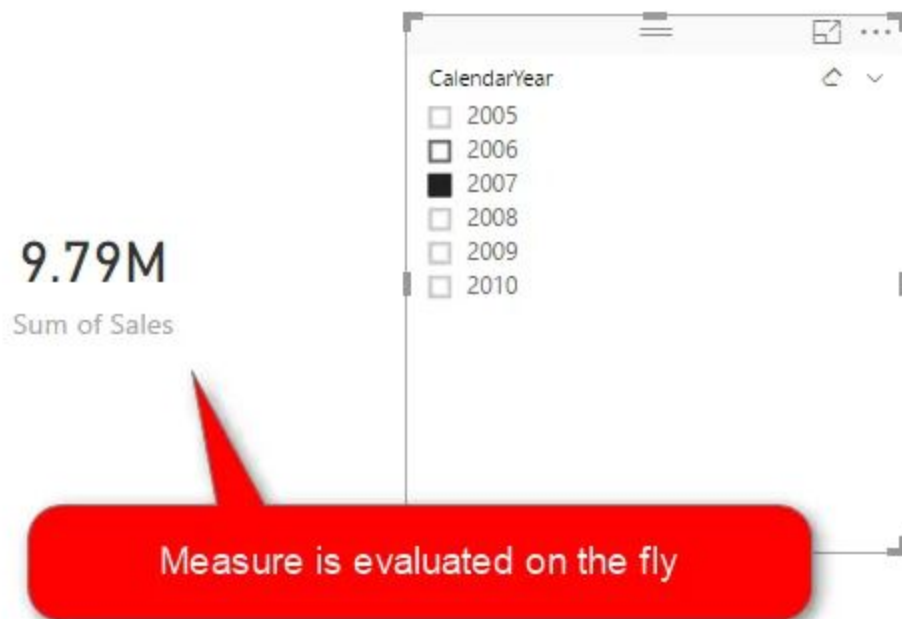
How to see the Value of the Measure?

Measures are calculated on the fly. This is, in fact, one of the most conceptual differences between a measure and calculated column. Okay, measure values are calculated on the fly, so how you can see the value?! The answer is by putting that into a report!

If I drag the measure above in a report as a card visual, then I would get a result;



When there is no filter applied in the report, this would return the grand total of sales, \$29.36M. However, if I add a slicer in the report, and select a value in it, I'll see a different result;

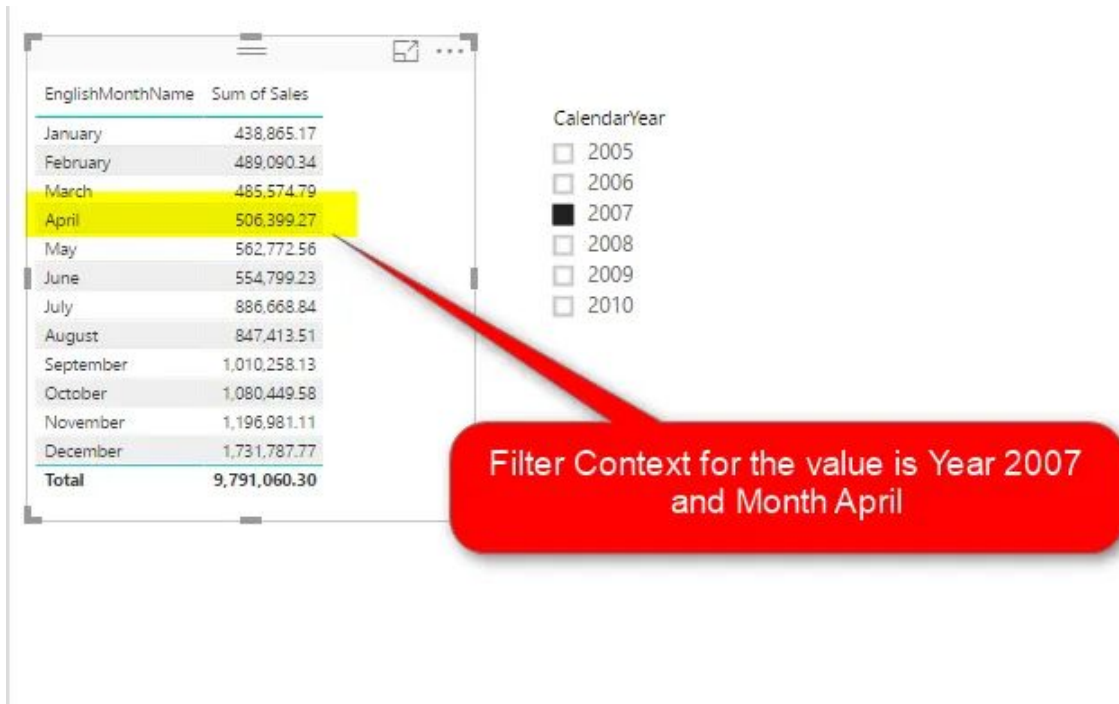


Now the measure calculation only shows me the sum of sales for the year 2007, which is \$9.79M.

Filter Context

Measure evaluates on the fly, if there is a slicer value for 2007, then the calculation will be done on the subset of data which is for 2007. If there is a table in visualization somewhere that slice and dice data by Education category, the result of the measure will take that into account as well. We can then say this;

Measure evaluates the value based on the subset of data selected by filters, slicers, or slicing and dicing components of visuals in the report. This filtered dataset, called Filter Context.



Filter Context basically is a combination of all filters that effect on the calculation of measure. There are much more to talk about when we say filter context. However, this should be enough for understanding rest of this article.

Measures do not consume RAM, they consume CPU

based on what you've learned above; measure calculation is done on the fly. This means to measure value is not stored in the memory. The measure will not consume Memory or RAM at all. On the other hand, Measures consume CPU, because their calculation should be done right at the time of visualizing it. If you change a filter or slicer, the calculation should be done again. Because the response time should be fast, then this calculation happens by CPU.

What is the side effect?

If you have many measures in your report, and their calculation is also complex calculation, then with changing every filter or slicer, you end up with a lot of rounding circles which shows CPU is desperately working hard to calculate all values.

Measures highlights

Based on above explanations, here are highlights of a Measure;

- *Calculated based on all filters: Filter Context (usually, not always)*
- *Is not Stored and is not pre-calculated*
- *Calculated on the Fly when you put it on a report page when you change a slicer, filter, or click on a column chart or any other visual to highlight and it affect this measure's value.*
- *Consumes CPU for calculation.*

When to use Measure, and when to use Calculated Column

Now that you know about these two types of calculation, we come to the critical question: When to use which? Do you need a Measure for your calculation or Calculated Column? The answer to this question is depends on what you want to calculate? This is an important question that you should be asking yourself when you want to create a new calculation:

Is the calculation row by row? or it is an aggregation? Is it going to be affected by filter criteria in the report?

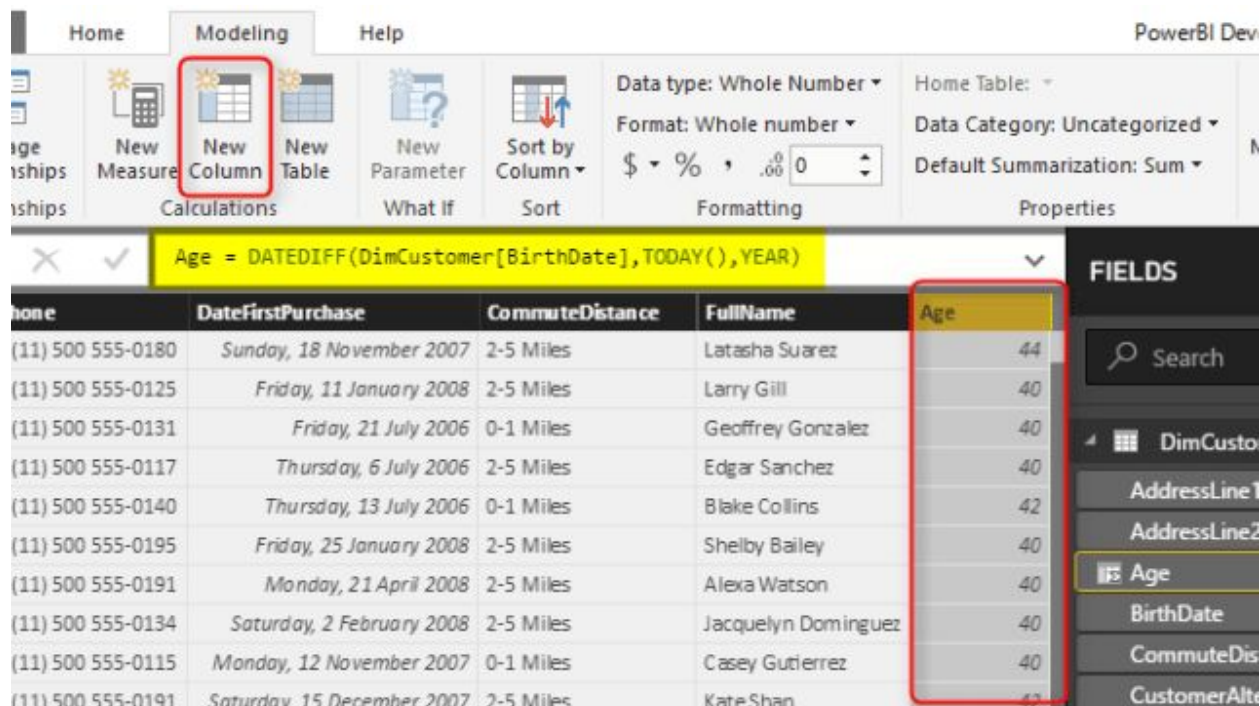
If the calculation is row by row (example: Profit = Sales – Cost, or Full name = First Name & " " & Last Name), then Calculated Column is what you need.

If the calculation is an aggregation or it is going to be affected by filter criteria in the report (example: Sum of Sales = Sum(Sales), or Sales Year to Date = TotalYTD(...)), then Measure is your friend.

Let's go through some examples;

Example 1: Calculating the age of customers

Age of customers does not change based on filters! It is only dependent on one thing; birthdate of the customer. and in the customer table, usually, you have the birthdate as a field. So this calculation can be simply a calculated column, which evaluates row by row for every customer.

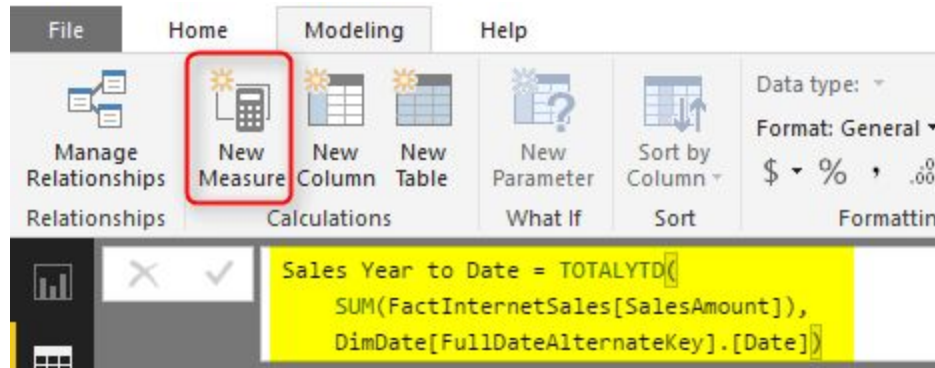


The screenshot shows the PowerBI Desktop interface. The 'Modeling' tab is active, and the 'New Column' button is highlighted with a red box. Below the ribbon, the formula bar displays the DAX formula: `Age = DATEDIFF(DimCustomer[BirthDate], TODAY(), YEAR)`. The table below shows the resulting data with an 'Age' column highlighted by a red box.

ID	Phone	DateFirstPurchase	CommuteDistance	FullName	Age
(11) 500 555-0180		Sunday, 18 November 2007	2-5 Miles	Latasha Suarez	44
(11) 500 555-0125		Friday, 11 January 2008	2-5 Miles	Larry Gill	40
(11) 500 555-0131		Friday, 21 July 2006	0-1 Miles	Geoffrey Gonzalez	40
(11) 500 555-0117		Thursday, 6 July 2006	2-5 Miles	Edgar Sanchez	40
(11) 500 555-0140		Thursday, 13 July 2006	0-1 Miles	Blake Collins	42
(11) 500 555-0195		Friday, 25 January 2008	2-5 Miles	Shelby Bailey	40
(11) 500 555-0191		Monday, 21 April 2008	2-5 Miles	Alexa Watson	40
(11) 500 555-0134		Saturday, 2 February 2008	2-5 Miles	Jacquelyn Dominguez	40
(11) 500 555-0115		Monday, 12 November 2007	0-1 Miles	Casey Gutierrez	40
(11) 500 555-0191		Saturday, 15 December 2007	2-5 Miles	Kate Shan	42

Example 2: Calculating Sales Year to Date

Year to date calculation depends on the filter criteria in the report, and also it is an aggregation. It becomes very complicated to calculation year to date for all variations of fields (per day, per month, per customer, per product, and etc). So this needs to be a Measure.



every time you put this measure into a report it calculates based on the filter criteria of the report;

Year	Quarter	Month	Sales Year to Date
2005	Qtr 3	July	473,388.16
2005	Qtr 3	August	979,579.85
2005	Qtr 3	September	1,453,522.89
2005	Qtr 4	October	1,966,852.36
2005	Qtr 4	November	2,510,845.77
2005	Qtr 4	December	3,266,373.66
2006	Qtr 1	January	596,746.56
2006	Qtr 1	February	1,147,563.25
2006	Qtr 1	March	1,791,698.45
2006	Qtr 2	April	2,455,390.74
2006	Qtr 2	May	3,128,946.94
2006	Qtr 2	June	3,805,710.59
2006	Qtr 3	July	4,306,075.74
2006	Qtr 3	August	4,852,077.21
2006	Qtr 3	September	5,202,544.20
2006	Qtr 4	October	5,617,934.44

Calculated Column or Power Query?

When it comes to calculating row by row, then Power Query is a better option in the majority of the cases. I have explained before what is M or DAX, and what are scenarios that you need to use each. Calculated Columns (in the majority of the cases, not always), can be implemented by Power Query. Read my other post about [M or DAX; That is the question](#) to learn more about this.

Measure: The Hidden Gem of DAX

You can do a Calculated column in the majority of the cases in Power Query as well, and in fact, it is much better to do that in Power Query in those cases. This means the hidden gem of DAX is Measure. Measure calculation is dynamic, on the fly, and based on filters applied in the report. The dynamic nature of measure calculation, make it the invincible feature of DAX or Power BI. You have seen in above calculation that Year to Date value is showed by month. If you simply bring Day value in the table, then this calculation will evaluate on a daily basis and works still perfectly fine;

Year	Quarter	Month	Day	Sales Year to Date
2005	Qtr 3	July	1	14,477.34
2005	Qtr 3	July	2	28,408.86
2005	Qtr 3	July	3	43,421.04
2005	Qtr 3	July	4	50,577.58
2005	Qtr 3	July	5	65,589.75
2005	Qtr 3	July	6	79,902.83
2005	Qtr 3	July	7	87,758.47
2005	Qtr 3	July	8	95,614.11
2005	Qtr 3	July	9	116,523.89
2005	Qtr 3	July	10	127,080.42
2005	Qtr 3	July	11	141,393.50
2005	Qtr 3	July	12	155,528.30
2005	Qtr 3	July	13	162,684.84
2005	Qtr 3	July	14	187,732.73
2005	Qtr 3	July	15	198,963.36
2005	Qtr 3	July	16	213,276.44
2005	Qtr 3	July	17	227,411.24
2005	Qtr 3	July	18	234,364.50
2005	Qtr 3	July	19	259,933.21
2005	Qtr 3	July	20	271,188.84
2005	Qtr 3	July	21	285,501.92
2005	Qtr 3	July	22	293,743.31

If you do it on a quarter level, the year to date calculation evaluates on the quarter level;

Year	Quarter	Sales Year to Date
2005	Qtr 3	1,453,522.89
2005	Qtr 4	3,266,373.66
2006	Qtr 1	1,791,698.45
2006	Qtr 2	3,805,710.59
2006	Qtr 3	5,202,544.20
2006	Qtr 4	6,530,343.53
2007	Qtr 1	1,413,530.30
2007	Qtr 2	3,037,501.36
2007	Qtr 3	5,781,841.84
2007	Qtr 4	9,791,060.30
2008	Qtr 1	4,283,629.96
2008	Qtr 2	9,720,059.11
2008	Qtr 3	9,770,899.74
2008	Qtr 4	9,770,899.74

This Dynamic nature of Measure calculation in DAX is something that you cannot find in many tools. That is why Measures are so commonly used in DAX. In fact 70% of your time when you write DAX is used for writing measures, if not more!

Summary: Calculated Column vs Measure in nutshell

Let's wrap up it all and go through comparison of these two types of calculations in DAX;

Measure	Calculated Column
Is not stored	Stored in Memory
Calculated on the fly	Calculates at the time of Refresh Report
Consumes CPU	Consumes Memory
Usually is a result of an aggregation	Row by row calculation usually
Value can be seen when adding in the report	Value can be seen in the column in Data tab
DAX usually is the best place for this calculation	In the majority of the cases can be done in Power Query
Example: Sales Year to Date	Example: Profit
Sales YTD = TotalYTD (Sum(Sales), DateField.[Date])	Profit = Sales - Cost

Hopefully, this post, helped you to understand the difference between these two types of calculation. If you have any questions, please don't hesitate to post your questions below.

