The NERD'S GUIDE TO DAX

PRAGMATIC WORKS

TABLE OF CONTENTS

DAX
FUNDAMENTALS

DAX, OR DATA ANALYSIS EXPRESSIONS, is an elegant and easy to learn formula language used in Power BI, Power Pivot and SSAS Tabular to add analytical value to your data model. DAX improves data models by allowing users to extract more value from their data and make more informed business decisions.

Microsoft specifically developed DAX to support a large user base. DAX is much easier to learn than traditional technical languages, making it an ideal language for users who don't come from a technical background but want to do their own Self-Service Business Intelligence. DAX is often compared to an advanced version of Excel, having high-end capability of managing and manipulating data. Many DAX functions are similar to functions in Excel, which means new users can leverage their existing knowledge to make an easy transition to writing and authoring DAX formulas. While there are similarities between DAX and Excel, the two languages are not interchangeable.

DAX is a functional language, which means formulas are constructed by applying and composing various functions. The functions can contain other nested functions, conditional statements, and value references. Execution starts from the innermost function, or parameter, and works outward, making formatting important.

DAX works with data that is stored in a tabular data model. These models are comprised of one or more tables, each table containing columns and rows of data. There are two primary data types: Numeric and non-numeric or other. Numeric includes integers, decimals, dates, and currency, while other includes strings and binary objects. If a function works on a number, it works for any numeric data type.

While you can create Power BI reports that show valuable insights without using DAX formulas, creating effective DAX formulas will help you get the most out of your data, providing insight and solutions that might be missed with typical analysis.

DAX CALCULATION TYPES

The two most common uses for DAX are calculated columns and calculated measures.

Calculated Columns

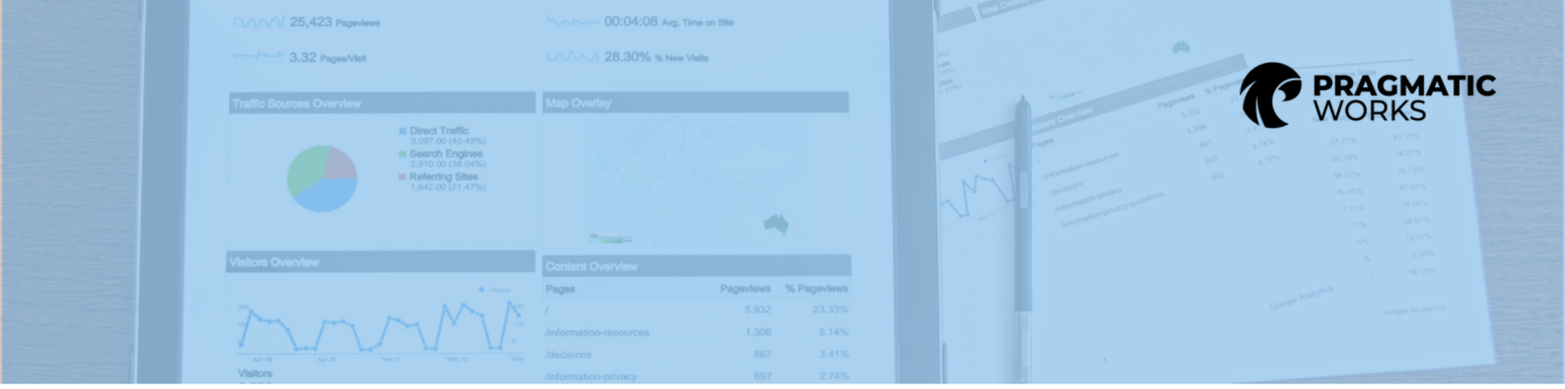
CALCULATED COLUMNS ARE PRIMARILY used to add new columns to a table providing more ways to describe and break down the data. For example, you may add an age column to a customer table so that sales and profit margin can be analyzed and broken down by age demographic. Another common use case for creating calculated columns is to create a unique key on a table, which may be necessary to define a relationship between two tables.

Calculated Measures

CALCULATED MEASURES ARE DYNAMIC calculations that recalculate depending on how a report is viewed or filtered. For example, if a user changed a time-range slider on a report, the measures on that report would be recalculated to reflect the time-range selected. Unlike calculated columns which are calculated during processing of the data model, measures are calculated at runtime when a report is opened or when a user interacts with the filters on a report. Therefore, the results of a measure are always changing and are not stored in your database. Calculated Measures are different from columns in quite a few ways.

TIP!

It is generally considered best practice to create new columns in your data model in the original source or in the Power Query editor, before the data is loaded in Power BI Desktop. This gives the user the best possible compression for the data.



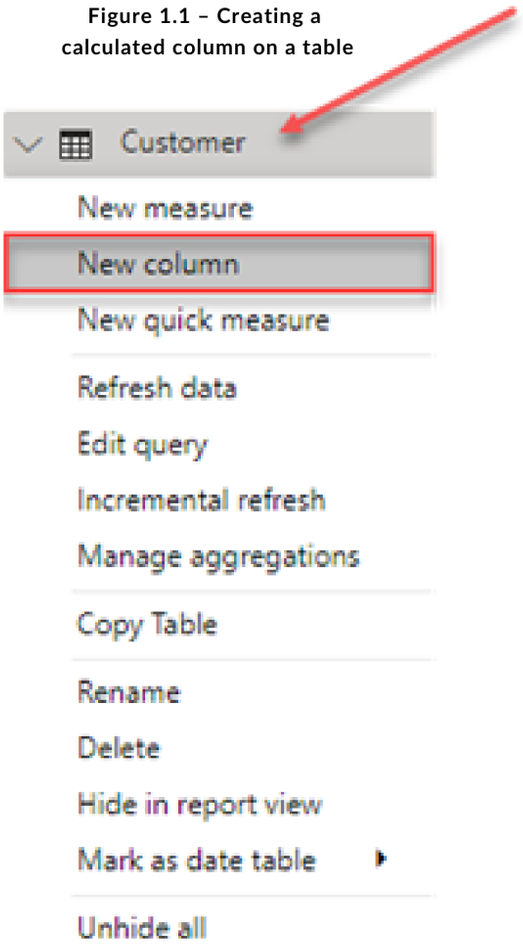
COLUMNS VS MEASURES

The results of calculated columns are immediately viewable in the table.	The results of calculated measures are not stored in the table.
Calculated columns can be used in filters and slicers.	Calculated measures can not be used in filters and slicers.
Calculated columns are updated when the data model is refreshed.	Calculated measures are dynamic and calculated as filters are applied.
Calculated columns take up more space in the data model.	Calculated measures do not take up space in the model.
Calculated columns are not dynamic.	Calculated measures are dynamic and always changing based on the filters applied.

HOW TO CREATE CALCULATED COLUMNS

CREATING A NEW CALCULATED COLUMN IS DONE BY ADDING A NEW COLUMN TO AN existing table and then completing the desired expression in the formula bar. Once again, this experience is very similar to users familiar to with Excel. The example presented here is specific to Power BI Desktop but also applies to Power Pivot and SSAS tabular. To create a new calculated column on a table, right click on the table name and select New column. See Figure 1.1.

Figure 1.1 – Creating a calculated column on a table



Once New column is selected, the power user can now write the DAX expression in the active formula bar. See Figure 1.2.

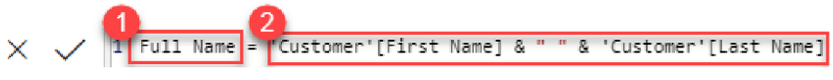


Figure 1.2 – DAX Expression for new calculated column

The expression in Figure 1.2 creates a new Full Name column on the customer table by combining the first and last name separated by a space. Let's take a closer look at the steps:

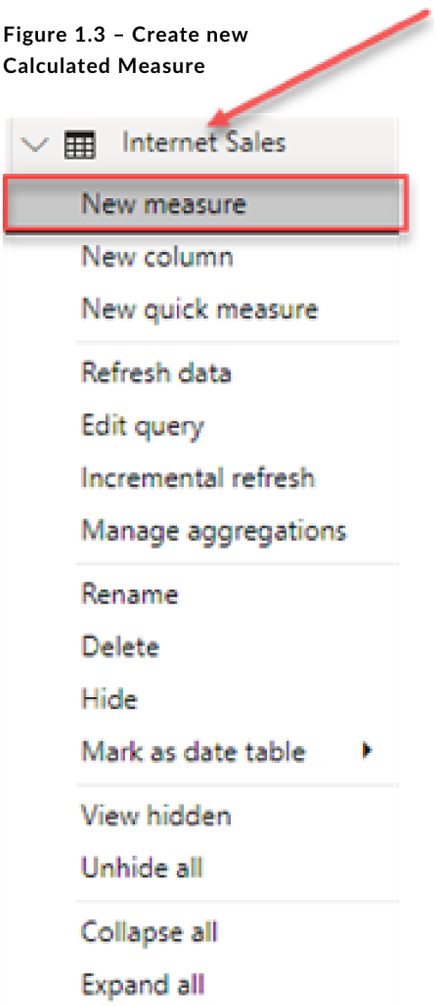
1) The text that appears before the = is the name of the column.
2) The text that appears after the = is the DAX expression.

Calculated columns appear in the Fields list of a table with a special icon indicating that the column was calculated with DAX.

HOW TO CREATE CALCULATED MEASURES

CALCULATED MEASURES ARE CREATED IN MUCH the same way that calculated columns are. First, right click on the table that you want the measure to be assigned to and select New measure from the drop down menu. See Figure 1.3.

Figure 1.3 – Create new Calculated Measure



Once **New measure** is selected, the power user can now write the DAX expression in the active formula bar. See Figure 1.4.

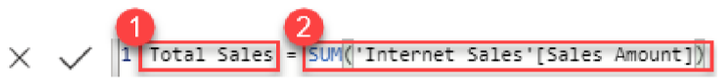


Figure 1.4 – DAX Expression for new calculated measure

The DAX expression in Figure 1.4 creates a new calculated Total Sales measure which is simply the SUM of the sales amount column. The first part of this expression is the name of the calculated measure and the second part is the DAX expression.

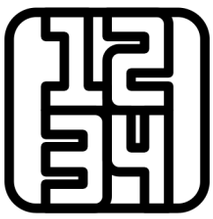
You may already know that in Power BI Desktop, numeric columns can be added to a visualization and be automatically aggregated using a default summarization like sum, max, count, etc. These default aggregations can offer quick insights into your data. However, we recommend you always explicitly create your own calculated measures because there are many advantages to doing so. Let's take a look at some of the advantages of explicitly creating calculated measures:

1. **The formatting can be set on the measure, creating consistency across report visuals.**
2. **The complex business logic is saved in a single location, making logic changes easier.**
3. **Calculated measures can be leveraged in other calculated measures, making the code easier to read and reducing rework.**

PRAGMATIC WORKS

DAX FUNCTIONS

DAX IS A FORMULA LANGUAGE THAT USES A COLLECTION OF FUNCTIONS which can be used to author columns, measures and tables as previously discussed. There are many different function categories in DAX which we will take a look at in this section. Functions have required parameters and optional parameters as inputs. These DAX functions, which perform various tasks from mathematical calculations to time intelligence, are what make DAX such a powerful and simple language to learn. Those familiar with Excel formulas will see similarities in many of these functions, although DAX formulas do differ in important ways. They always reference a complete column or a table and require filters to focus on specific values. There are functions that use the current row value or a related value as a parameter to perform calculations that vary by context to customize calculations on a row-by-row basis. And aside from returning a single value, many DAX functions return a table, which can be used as an input to other functions.



Aggregation Functions

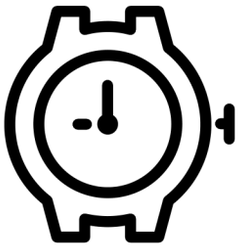
Aggregation functions are probably the most common in DAX. Aggregations group together multiple values to create a single value. DAX has numerous aggregation functions available. Here are some, not all, of the available aggregate functions: SUM, MIN, MAX, AVERAGE, COUNT, etc.

Calculate Function

The CALCULATE function is one of the most powerful and widely used DAX functions as it helps evaluate functions by modifying the filter context of the expression. The CALCULATE function is similar to the Excel SUMIF & COUNTIF functions, however CALCULATE is not limited to a single aggregation.

Date and Time Functions

These functions perform calculations on dates and time values. They are like Excel date and time functions, but DAX functions use a datetime data type and do not necessarily apply to time intelligence. Common examples include TOTALYTD, SAMEPERIODLASTYEAR, and DATESINPERIOD.



Filter Functions

Filter functions return specific data types, look up values in related tables, and filter by related values. They differ greatly from Excel functions. Lookup functions work by using tables and relationships, like a database. Filter functions can manipulate data context to make dynamic calculations. Filter functions are similar to the calculate function except for the fact that they are not mutable and are most commonly used to return a subset of an expression or table.

Financial Functions

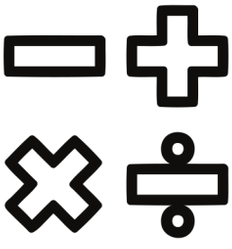
Financial functions are used in formulas that perform calculations such as net present value and rate of return. They're comparable to financial functions in Excel.

Information Functions

These functions look at a data within a table, column, or row provided as an argument to another function and confirm whether the value matches the expected type. For instance, the function ISERROR returns TRUE if the data evaluated contains an error. While these functions can be helpful, it's recommended to know the data type of your columns, rather than depending solely on these functions.

Logical Functions

Logical functions evaluate an expression or argument and return TRUE or FALSE if the condition is met or not. For example, the TRUE function confirms whether an expression being evaluated returns a TRUE value. These functions can be expressed with operators when more than two conditions are needed in a formula, such as AND being replaced with &&, although for readability it is best practice use the function name itself where possible.



Math & Trig Functions

Mathematical and trig functions perform various mathematical functions on referred values, including basic arithmetic, conditional sums & products, exponents & logarithms, and the trigonometric ratios.

DAX FUNCTIONS CONTINUED

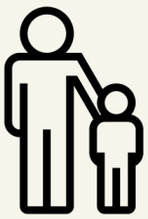
PRAGMATIC WORKS

Other Functions

These functions don't fall into other categories as they perform actions that aren't defined by any of the categories most functions belong to.

Parent and Child Functions

These functions assist users with managing data that's presented as a parent/child hierarchy in data models.



Relationship Management Functions

These functions manage and utilize relationships between tables.

Statistical Functions

These functions perform statistical and aggregation functions on data values, such as finding sums and averages and minimum and maximum values. In DAX, you can filter a column before aggregating or create aggregations based on related tables.

Table Manipulation Functions

The table functions apply operations and conditions on entire tables. The output is then used as inputs for other expressions or arguments in a DAX formula. The results of these functions retain the relationships between columns of that table.

Text Functions

Text functions can return part of a string, search for text within a string, or concatenate string values. They can also control the formats for dates, times, and numbers. These text functions work similarly to Excel functions with the same name.

Time Intelligence Functions

The time-intelligence functions use built-in knowledge about calendars and dates and evaluate data over a fixed period such as days, weeks, months, quarter, or years. They require well-formed tables to compare time periods for sales, inventory, and more.

ROW-LEVEL SECURITY (RLS)

WITH POWER BI

ROW LEVEL SECURITY (RLS) USES GROUP MEMBERSHIP OR EXECUTION CONTEXT TO

control access to rows in a database table. It simplifies coding and design of security and enforces restrictions on data row access.

The access restriction logic is in the database tier, which minimizes the surface area of the security system. The database system applies the access restrictions every time that data access is attempted from any tier.

Row-Level Security can be set up in Power BI itself, or through respecting the data source restrictions from a live connection like to SSAS Tabular.

Dynamic RLS can also be accomplished by storing the definition of security beside the user account information in the data source. For example, when John logs in to the system, based on data tables that define John's access, he should be able to see only the data relevant to him. This is possible using DAX functions like UserName() or UserPrincipalName() when creating Roles in RLS.

DAX TIPS

PRAGMATIC WORKS

1 Use Measures to Carry Out Calculations

Aggregations, ratios, percentages and other calculations should generally be created in calculated measures not calculated columns. Measures are dynamic and responsive to filters whereas columns are static and updated only during the data model refresh.

2 Time Intelligence With a Date Table

Always use a separate and distinct date table in your data model to simplify filtering of data and for development of time intelligence calculations like year-to-date and year-over-year.

3 Format Measures

Always format a measure once it has been developed to create consistency across visualizations and reports. Once the formatting is set it will carry into pivot tables or charts each time that measure is selected.

4 Write DAX Like Code

Calculated measures and columns written with DAX should look more like programming code and not excel functions. Well authored DAX expressions should use variables and comments. Commenting will help with reviewing and understanding the code while variables help with readability and debugging code.

5 Avoid Data Transformation Steps

When possible, data transformation steps should not be carried out in a DAX calculation. Data transformation and cleansing should be carried out with a different tool, such as Power Query.

PRAGMATIC WORKS

pragmaticworks.com

hello@pragmaticworks.com

(904) 413-1911

7175 US HWY 17

Suite 2

Fleming Island, FL 32003

Follow us on social media