Collapse All Code: All

DAX Syntax Specification for PowerPivot

See Also Send Feedback

Data Analysis Expressions (DAX) is a library of functions, operators, and constants that can be combined to build formulas and expressions in PowerPivot for Excel. This section provides details about the syntax and requirements of the DAX language.

For examples of the kinds of formulas that you can build, and how you can use expressions to filter tables and change context, see Data Analysis Expressions (DAX) Overview. This topic contains the following sections:

Syntax Requirements

Naming Requirements

Functions

Operators and Constants

Data Types

Syntax Requirements

DAX formulas are very similar to the formulas you type in Excel tables, but there are some key differences.

- In Microsoft Excel you can reference individual cells or arrays; in PowerPivot you can reference only complete tables or columns of data. However, If you need to work with only part of a column, or with unique values from a column, you can achieve similar behavior by using DAX functions that filter the column or return unique values.
- DAX formulas do not support exactly the same data types as Microsoft Excel. In general, DAX provides more data types than Excel does, and DAX performs implicit type conversions on some data when importing. For more information, see **Data Types in DAX**.

A DAX formula always starts with an equal sign (=). After the equals sign, you can provide any expression that evaluates to a scalar, or an expression that can be converted to a scalar. These include the following:

- A scalar constant, or expression that uses a scalar operator (+,-,*,/,>=,...,&&, ...)
- References to columns or tables. The DAX language always uses tables and columns as inputs to functions, never an array or arbitrary set of values.
- Operators, constants, and values provided as part of an expression.
- The result of a function and its required arguments. Some DAX functions return a table instead of a scalar, and must be wrapped in a function that evaluates the table and returns a scalar; unless the table is a single column, single row table, then it is treated as a scalar value.
  Most PowerPivot functions require one or more arguments, which can include tables, columns, expressions, and values. However, some functions, such as PI, do not require any arguments, but always require parentheses to indicate the null argument. For example, you must always type PI(), not PI. You can also nest functions within other functions.
- Expressions. An expression can contain any or all of the following: operators, constants, or references to columns.

For example, the following are all valid formulas.

Formula	Result
=3	3
="**Sales**"	**Sales**
='Sales'[Amount]	If you use this formula within the Sales table, you will get the value of the column Amount in the Sales table for the current row.
=(0.03 * [Amount]) =0.03 * [Amount]	Three percent of the value in the Amount column of the current table. Although this formula can be used to calculate a percentage, the result is not shown as a percentage unless you apply formatting in the table.
=PI()	The value of the constant pi.

Note

Formulas can behave differently depending on whether they are used in a calculated column, or in a measure within a PivotTable. You must always be aware of the context and how the data that you use in the formula is related to other data that might be used in the calculation. For more information, see Context in DAX Formulas.

Naming Requirements

A PowerPivot window can contain multiple tables, each on its own tab. Together the tables and their columns comprise a database stored in the PowerPivot VertiPaq engine. Within that database, all tables must have unique names. The names of columns must also be unique within each table. All object names are case-insensitive; for example, the names **SALES** and **Sales** would represent the same table.

Each column and measure that you add to an existing PowerPivot database must belong to a specific table. You specify the table that contains the column either implicitly, when you create a calculated column within a table, or explicitly, when you create a measure and specify the name of the table where the measure definition should be stored.

When you use a table or column as an input to a function, you must generally *qualify* the column name. The *fully qualified* name of a column is the table name, followed by the column name in square brackets: for examples, 'U.S. Sales'[Products]. A fully qualified name is always required when you reference a column in the following contexts:

- As an argument to the function, VALUES
- As an argument to the functions, ALL or ALLEXCEPT
- In a filter argument for the functions, CALCULATE or CALCULATETABLE
- As an argument to the function, RELATEDTABLE
- As an argument to any time intelligence function

An *unqualified* column name is just the name of the column, enclosed in brackets: for example, [Sales Amount]. For example, when you are referencing a scalar value from the same row of the current table, you can use the unqualified column name.

If the name of a table contains spaces, reserved keywords, or disallowed characters, you must enclose the table name in single quotation marks. You must also enclose table names in quotation marks if the name contains any characters outside the ANSI alphanumeric character range, regardless of whether your locale supports the character set or not. For example, if you open a workbook that contains table names written in Cyrillic characters, such as 'Таблица', the table name must be enclosed in quotation marks, even though it does not contain spaces.

Note

To make it easier to enter the fully qualified names of columns, we recommend that you use the formula AutoComplete feature in the client.

Tables

- Table names are required whenever the column is from a different table than the current table. Table names must be unique within the database.
- Table names must be enclosed in single quotation marks if they contain spaces, other special characters or any non-English alphanumeric characters.

Measures

- Measure names must always be in brackets.
- Measure names can contain spaces.
- Each measure name must be unique within a database. Therefore, the table name is optional in front of a measure name when referencing an existing measure. However, when you create a measure you must always specify a table where the measure definition will be stored.

Columns

Column names must be unique in the context of a table; however, multiple tables can have columns with the same names (disambiguation comes with the table name).

In general, columns can be referenced without referencing the base table that they belong to, except when there might be a name conflict to resolve or with certain functions that require column names to be fully qualified.

Reserved Keywords

If the name that you use for a table is the same as an Analysis Services reserved keyword, an error is raised, and you must rename the table. However, you can use keywords in object names if the object name is enclosed in brackets (for columns) or quotation marks (for tables).

Note

Note that quotation marks can be represented by several different characters, depending on the application. If you paste formulas from an external document or Web page, make sure to check the ASCII code of the character that is used for opening and closing quotes, to ensure that they are the same. Otherwise DAX may be unable to recognize the symbols as quotation marks, making the reference invalid.

Special Characters

The following characters and character types are not valid in the names of tables, columns, or measures:

- Leading or trailing spaces; unless the spaces are enclosed by name delimiters, brackets, or single apostrophes.
- Control characters
- The following characters that are not valid in the names of PowerPivot objects: .,;':/*|?&%$!+=()[]{}<>

Examples of Object Names

The following table shows examples of some object names:

Object Types	Examples	Comment
Table name	**Sales**	If the table name does not contain spaces or other special characters, the name does not need to be enclosed in quotation marks.
Table name	**'Canada Sales'**	If the name contains spaces, tabs or other special characters, enclose the name in single quotation marks.
Fully qualified column name	**Sales [Amount]**	The table name precedes the column name, and the column name is enclosed in brackets.
Fully qualified measure name	**Sales[Profit]**	The table name precedes the measure name, and the measure name is enclosed in brackets. In certain contexts, a fully qualified name is always required.
Unqualified column name	**[Amount]**	The unqualified name is just the column name, in brackets. Contexts where you can use the unqualified name include formulas in a calculated column within the same table, or in an aggregation function that is scanning over the same table.
Fully qualified column in table with spaces	**'Canada Sales'[Qty]**	The table name contains spaces, so it must be surrounded by single quotes.

Note

To make it easier to enter the fully qualified names of columns, we recommend that you use the AutoComplete feature when building formulas. For more information, see Build Formulas for Calculations.

Miscellaneous Restrictions

The syntax required for each function, and the type of operation it can perform, varies greatly depending on the function. In general, however, the following rules apply to all formulas and expressions:

- DAX formulas and expressions cannot modify or insert individual values in tables.
- You cannot create calculated rows by using DAX. You can create only calculated columns and measures.
- When defining calculated columns, you can nest functions to any level.
- DAX has several functions that return a table. Typically, you use the values returned by these functions as input to other functions, which require a table as input.

Functions in DAX

DAX provides the following types of functions.

- Date and Time Functions (DAX)
- Filter Functions (DAX)
- Information Functions (DAX)
- Logical Functions (DAX)
- Math and Trigonometric Functions (DAX)
- Statistical Functions (DAX)
- Text Functions (DAX)

DAX Operators and Constants

The following table lists the operators that are supported by DAX. In general, operators in DAX behave the same way that they do in Microsoft Excel, with some minor exceptions. For more information about the syntax of individual operators, see DAX Operator Reference for PowerPivot.

Operator Type	Symbol and Use
Parenthesis operator	() precedence order and grouping of arguments
Arithmetic operators	+ (addition) - (subtraction/sign) * (multiplication) / (division) ^ (exponentiation)
Comparison operators	= (equal to) > (greater than) < (less than) >= (greater than or equal to) <= (less than or equal to) <> (not equal to)
Text concatenation operator	& (concatenation)
Logic operators	&& (and) || (or)

Data Types in DAX

You do not need to cast, convert, or otherwise specify the data type of a column or value that you use in a DAX formula. When you use data in a DAX formula, DAX automatically identifies the data types in referenced columns and of the values that you type in, and performs implicit conversions where necessary to complete the specified operation.

For example, if you try to add a number to a date value, PowerPivot will interpret the operation in the context of the function, like Excel does, and convert the numbers to a common data type, and then present the result in the intended format, a date.

However, there are some limitations on the values that can be successfully converted. If a value or a column has a data type that is incompatible with the current operation, DAX returns an error. Also, DAX does not provide functions that let you explicitly change, convert, or cast the data type of existing data that you have imported into a PowerPivot workbook.

Important

PowerPivot does not support use of the variant data type used in Excel. Therefore, when you load or import data, it is expected that the data in each column is generally of a consistent data type.

Some functions return scalar values, including strings, whereas other functions work with numbers, both integers and real numbers, or dates and times. The data type required for each function is described in the section, DAX Function Reference for PowerPivot.

Tables are a new data type in PowerPivot. You can use tables containing multiple columns and multiple rows of data as the argument to a function. Some functions also return tables, which are stored in memory and can be used as arguments to other functions.

For more information about the different numeric and date/time data types, and for details on the handling of nulls and empty strings, see **Data Types Supported in PowerPivot Workbooks**.

See Also

Concepts

Build Formulas for Calculations

Stay Up to Date with PowerPivot for Excel
For the latest information about PowerPivot for Excel, visit the PowerPivot for Excel Online Help