DAX for Power BI

Sample manual - first two chapters

Wise Owl Training

TABLE OF CONTENTS (1 of 5)

1	INTRODUCTION TO DAX IN POWER BI	Page
1.1	DAX in Power BI	7
	How DAX is Used 1 – Calculated Columns	7
	How DAX is Used 2 – Measures	7
	How DAX is Used 3 – Calculated Tables	8
	Choosing the Type of Calculation	8
	Where Else is DAX Used?	8
1.2	The Construct-a-Creature Database	9
	The Tables and Relationships	9
1.3	Getting Help with DAX	10
	Getting Help within Power BI	10
	Other Sources of Help	10

2	BASIC CALCULATED COLUMNS	Page
2.1	Calculated Columns	11
	Key Features of Calculated Columns	11
	Creating a Calculated Column	11
	Entering a Formula	12
	Calculated Column Properties	13
	Using Calculated Columns in Visuals	13
2.2	Referencing Columns and Tables	14
	Qualifying Column Names	14
	Table Names	14
	Referencing Columns in Other Tables	15
	The Related Function	15
2.3	Editing DAX Code	16
	Multiple Lines and Indenting	16
	Comments	17
	Keyboard Shortcuts	17

3	WORKING WITH DATA TYPES	Page
3.1	DAX Data Types	18
	Viewing a Column's Data Type	18
3.2	Working with Numbers	19
	Basic Arithmetic	19
	Controlling the Calculation Order	19
	Safely Dividing Numbers	20
	Numeric Functions	21
3.3	Working with Text	22
	Writing Text in Calculations	22
	Concatenating Text	22
3.4	Text Functions	23
	Finding and Extracting Text	23
	Replacing Text	23
	Generating Text	24
	Converting and Formatting Text	24
3.5	Working with Dates	25
	Entering Date and Time Values	25
	Returning the Current Date and Time	25
	Calculating Date and Time Values	26
	Calculating the Difference Between Dates	26
	Extracting Date Parts	27
	Formatting Dates	27

4	CONDITIONAL FUNCTIONS	Page
4.1	The IF Function	28
	Testing a Single Condition	28
	Comparison Operators	28
	The IN Operator	28
	Nesting IF Functions	29
	Combining Logical Tests	29
	The NOT Operator	29
4.2	Working with Blanks	30
	Producing a Blank	30
	Blank Arithmetic	30
	Testing for Blanks	30
	The COALESCE Function	31
4.3	Testing for Errors	32
	The ISERROR and IFERROR Functions	32
	Avoiding Error Functions	32
4.4	The SWITCH Function	33
	A Simple SWITCH Function	33
	Logical Tests in a SWITCH Function	33

© Copyright 2025

TABLE OF CONTENTS (2 of 5)

5	BASIC MEASURES	Page
5.1	Introduction to Measures	34
	Measures vs. Calculated Columns	34
	Implicit Measures	34
5.2	Creating a Measure	35
	Adding a Measure to a Table	35
	Formatting Measures	36
	Displaying a Measure in a Visual	36
	Referencing Measures	37
5.3	Filter Context	38
	What is Filter Context?	38
	How DAX Applies Filter Context	39
5.4	Measures Tables	40
	Creating a Separate Measures Table	40
	Moving Measures	41
5.5	Quick Measures	42
	Creating a Quick Measure	42
	Editing a Quick Measure	43
	Using a Quick Measure	43

6	AGGREGATION FUNCTIONS	Page
6.1	Aggregating Column Values	44
	Basic Aggregation Functions	44
	Functions for Counting	45
	Dealing with Boolean Values	45
6.2	Aggregating Expressions	46
	The AggregateX Functions	46
6.3	Iterators and Row Context	47
	A Reminder of Filter Context	47
	Row Context in Iterator Functions	48
	The Final Result	48
	How to Spot Iterators	48

7	THE CALCULATE FUNCTION	Page
7.1	Introducing the CALCULATE Function	49
	Expressions in the CALCULATE Function	49
7.2	Adding New Filters	50
	Basic Filter Expressions	50
	Adding Multiple Filters	51
	Filter Arguments and Filter Context	51
	Multiple Columns in Filter Arguments	52
7.3	Replacing Filters	53
	Replacing an Existing Filter	53
	Comparing Differently Filtered Measures	54
	Dealing with Blank Values	54
7.4	Keeping Filters	55
	The Problem with the Default Behaviour	55
	The KEEPFILTERS Function	56
	Using the VALUES Function	56
7.5	Removing Filters	57
	Removing Every Filter	57
	Using the ALL Function	58
	Comparing Filtered and Unfiltered Values	58
	Removing Filters from Specific Fields	59
	An Issue with Sort-By Fields	60
	Removing Filters from a Specific Table	60
7.6	Special Filter Removal Functions	61
	The ALLEXCEPT Function	61
	The ALLSELECTED Function	62

8	VARIABLES	Page
8.1	Introduction to Variables	63
	Using Variables in Measures	63
8.2	How Variables are Evaluated	64
	Lazy Evaluation	64
	DAX Variables are Constants	64
8.3	Debugging with Variables	65
	Returning Different Variables	65
8.4	Nesting Variables	66
	Variables in Functions	66
	Variable Scope	67

© Copyright 2025

TABLE OF CONTENTS (3 of 5)

9	THE FILTER FUNCTION	Page
9.1	Introduction to the FILTER Function	68
	A Basic FILTER Example	68
	Using the CALCULATE Function	68
	How CALCULATE and FILTER are Related	69
	Using Multiple Filters	69
	Using Variables	69
9.2	FILTER vs. CALCULATE	70
	Referencing Multiple Fields	70
	Using Fields from Different Tables	70
	Referring to Measures	71
	Replacing Filters	72
9.3	The CALCULATETABLE Function	73
	Using CALCULATETABLE	73

10	FILTERS AND RELATIONSHIPS	Page
10.1	Relationships and Filter Direction	74
	The Problem with Relationships	74
	Changing the Cross Filter Direction	75
	Solving the Problem using Filters	76
10.2	Cross Filter Direction in Measures	77
	The CROSSFILTER Function	77
	Using Single and Both Filter Directions Simultaneously	78
	Multiple CROSSFILTER Functions	78

11	CONTEXT TRANSITION	Page
11.1	What is Context Transition?	79
	Row and Filter Context	79
11.2	Context Transition in Calculated Columns	80
	Row Context in Calculated Columns	80
	Performing Context Transition	80
	Implicit Context Transition	81
	The RELATEDTABLE Function	81
11.3	Context Transition in Measures	82
	Row Context in Measures	82
	Context Transition in Measures	83
	The Effect of Filter Context	83
	Removing Filters	84
11.4	Ranking Values	85
	The RANKX Function	85
	Ranking in Calculated Columns	85
	Context Transition in Calculated Columns	86
	Ranking in Measures	86

12	TIME INTELLIGENCE	Page
12.1	Introduction to Time Intelligence	87
	Calendar Tables	87
	The Date Column	88
	Referring to Calendar Tables	88
12.2	Time Intelligence Functions	89
	General Time Intelligence Functions	89
	Using the DATEADD Function	89
	How DATEADD Works	90
	Using the DATESINPERIOD Function	90
	Using the Current Date	91
	Using Specific Dates	91
12.3	To Date Calculations	92
	Returning Date Ranges	92
	Calculating Running Totals	92
	Total To Date Functions	93
	Easier Running Total Calculations	93
	Specifying Year End Dates	94
	Calculating Life to Date Values	94
12.4	Next and Previous Periods	95
	Next and Previous Period Functions	95
	Comparing Entire Previous Years	96
	Comparing Parts of Previous Years	97
12.5	Period Start and End Dates	98
	Period Start and End Functions	98
	Start and End Dates	99
	Opening and Closing Balances	100
	First and Last Non-Blank Dates	100
	First and Last Non-Blank Values	101
	Non-Blank Opening Balances	101
12.6	Moving Averages	102
	Calculating a Moving Average	102

© Copyright 2025

TABLE OF CONTENTS (4 of 5)

13	CUSTOM CALENDARS	Page
13.1	Why use Custom Calendars?	103
	Disabling Automatic Calendars	103
13.2	Creating a Custom Calendar	104
	The CALENDARAUTO and CALENDAR Functions	104
	Adding Extra Columns	105
	Financial Years	106
13.3	Finishing the Calendar	107
	Marking as a Date Table	107
	Changing Default Aggregations	107
	Setting Sort-By Columns	108
	Creating Hierarchies	108
	Hiding Fields	109
	Creating a Relationship	109
13.4	Using a Custom Calendar	110
	Creating Visuals	110
	Time Intelligence Functions	110
13.5	Multiple Date Fields	111
	Using Multiple Calendars	111
	Using a Single Calendar	112
	Changing the Active Relationship	112
	The USERELATIONSHIP Function	113
13.6	Special Dates	114

14	DYNAMIC MEASURES	Page
14.1	Dynamic Labels	115
	Why use Dynamic Labels?	115
14.2	Returning a Single Value	116
	The VALUES Function	116
	Testing for a Single Value	117
	The SELECTEDVALUE Function	117
14.3	Concatenating Values	118
	The CONCATENATEX Function	118
	More Complex Expressions	118
14.4	Filtered Values	119
	Testing for Filtered Values	119
	Testing for Cross Filtered Values	119
	Selecting the Top N Rows	120
14.5	Disconnected Slicers	121
	Creating a Disconnected Table	121
	Creating a Disconnected Slicer	122
	Referencing the Selected Value	122
14.6	Formatting with Measures	123
	Calculating Colours with Measures	123
	Using Measures in Conditional Formatting	124
	Choosing Colours with Slicers	124

© Copyright 2025

TABLE OF CONTENTS (5 of 5)

15	VISUAL CALCULATIONS	Page
15.1	What are Visual Calculations?	125
15.2	Creating Visual Calculations	126
	Creating a Visual	126
	Adding a Visual Calculation	126
	Writing the Formula	127
	Using Pre-Defined Visual Calculations	128
15.3	Editing a Visual Calculation	129
	Formatting the Results	129
	Editing the Formula	129
15.4	Hiding Fields in a Visual	130
15.5	Running Totals	131
	Calculating a Running Total	131
	Running Totals and Hierarchies	131
	Resetting a Running Total	132
	Changing the Direction	133
15.6	Moving Averages	134
	Calculating a Moving Average	134
	Excluding the Current Row	134
15.7	Referring to Other Rows	135
	Next and Previous Rows	135
	Setting the Step Value	136
	First and Last Rows	136
15.8	Referring to Parent Values	137
	Collapsing Filters	137
	Collapsing Filters Relatively	138
	Collapsing All Filters	138
15.9	Referring to Child Rows	139
	Expanding Filters	139
	Expanding Filters Relatively	139
15.10	Controlling Group Headers	140
	Testing for the Presence of a Column	140

© Copyright 2025

CHAPTER 1 - INTRODUCTION TO DAX IN POWER BI

1.1 DAX in Power BI

In Power BI you can use the *DAX* (*Data Analysis eXpressions*) language to create *calculated columns*, *measures* and *tables*. You can see an example of each below.

How DAX is Used 1 – Calculated Columns

A *calculated column* is like a formula in an Excel table. The results of a calculated column are stored in the data model.

```
1 Sale value = Sales[Price] * Sales[Quantity]
```

SaleDate	ProductId	CentreId	Quantity	Price	Sale value
21 July 2023	12	178	2	£6.99	£13.98
21 July 2023	13	207	1	£3.95	£3.95
21 July 2023	13	342	4	£3.95	£15.80

This calculated column gives the value for each row of a sales table by multiplying the price of an item by the quantity sold.

As we will see in this manual, a calculated column is evaluated for each row of a table; DAX uses the *row context* to access the correct values for each separate calculation.

How DAX is Used 2 – Measures

A *measure* is a formula which calculates a value when you place it in a visual. The visual provides a *filter context* which tells the measure which values it can use in the calculation.

```
1 Sum of sale value = SUMX(
2     Sales,
3     Sales[Price] * Sales[Quantity]
4 )
```

RegionName	Air	Land	Water	Total
East Anglia	1,268.71	13,640.28	2,050.91	**16,959.90**
East Midlands	2,609.87	26,289.48	5,033.67	**33,933.02**
London	4,899.45	54,018.62	9,136.50	**68,054.57**
North	2,874.62	25,980.55	4,334.49	**33,189.66**
North West	7,950.17	69,751.18	12,614.71	**90,316.06**
South East	10,697.97	102,171.04	17,507.12	**130,376.13**
South West	2,616.48	23,769.80	4,506.69	**30,892.97**
West Midlands	4,702.33	49,116.82	8,878.73	**62,697.88**
Yorkshire & Humberside	6,231.78	47,396.27	8,509.78	**62,137.83**
Total	**43,851.38**	**412,134.04**	**72,572.60**	**528,558.02**

This measure multiplies the price of an item by the quantity sold for each row of a sales table, then sums the results.

When we place the measure in a visual (a matrix in this case) the measure is calculated for each data point in the visual.

The filter context controls which values the measure can access. This cell gives the total sales value for products in the **Water** environment sold in the **South West** region.

© Copyright 2025

How DAX is Used 3 – Calculated Tables

Although most of the tables in your Power BI data model will be created by importing data, you can also use DAX to calculate tables. It's common to use this technique to create custom calendars.

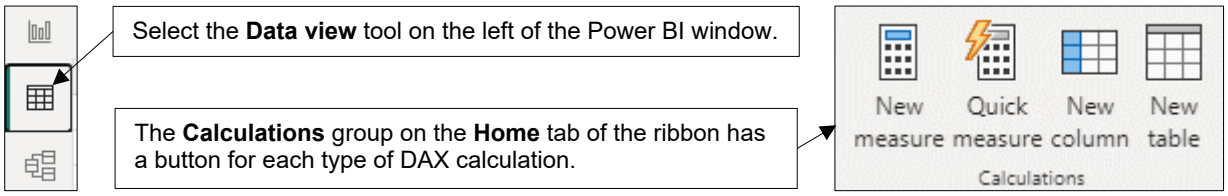
```
1  Sales Calendar = ADDCOLUMNS(
2      CALENDAR(
3          DATE(YEAR(MIN(Sales[SaleDate])), 1, 1),
4          DATE(YEAR(MAX(Sales[SaleDate])), 12, 31)
5      ),
6      "Year", YEAR([Date]),
7      "Month", MONTH([Date]),
8      "Month Name", FORMAT([Date], "MMMM"),
9      "Day", DAY([Date])
10 )
```

This DAX expression creates a calculated table which contains a range of dates related to the sales in our database.

Date	Year	Month	Month Name	Day
01/01/2020 00:00:00	2020	1	January	1
02/01/2020 00:00:00	2020	1	January	2
03/01/2020 00:00:00	2020	1	January	3

Choosing the Type of Calculation

Power BI provides several ways to create calculated columns, measures and tables. One way is to click the relevant tool on the **Home** tab of the ribbon while in the **Data** view.



Where Else is DAX Used?

In addition to Power BI, you can write DAX in the following applications:

Application	Description
Power Pivot	Power Pivot is an add-in for Excel which allows you to combine data from multiple sources and present this in a pivot table or chart.
SQL Server Analysis Services (SSAS) Tabular	SSAS Tabular Model allows you to combine data from lots of different data sources, apply security to it to control who sees what and then allow employees of your organisation to share the resulting data model.

© Copyright 2025

1.2 The Construct-a-Creature Database

This courseware uses data from the (fictitious) Wise Owl subsidiary **Construct-a-Creature** (a retail chain loosely modelled on Build-a-Bear, but with a wider range of animals available for purchase).

The Tables and Relationships

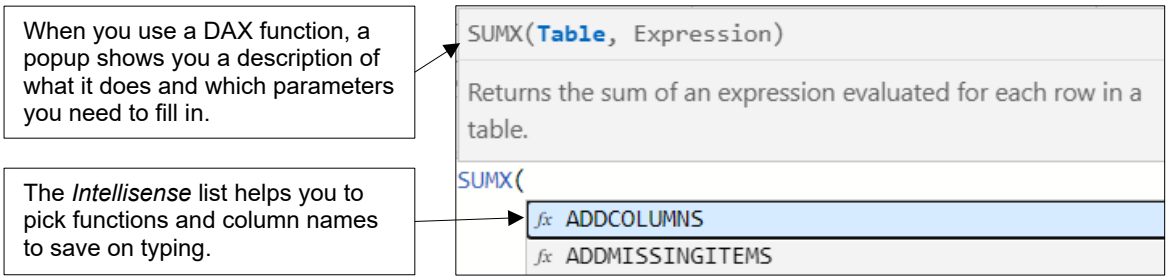
You can see the tables and relationships of the database in the diagram below:

1.3 Getting Help with DAX

The amount of help available for DAX both within Power BI and from third parties has increased dramatically since the product was first released.

Getting Help within Power BI

You'll see various popups appear to help you as you write DAX in Power BI.



Other Sources of Help

You can find more descriptive help for DAX on a range of websites, as shown in the table below:

Website	Description	URL
Microsoft DAX Reference	Microsoft's official documentation for DAX functions. It's somewhat dry but a useful technical reference.	https://learn.microsoft.com/en-us/dax/
Power BI Community	A Microsoft forum in which you can post questions about any aspect of Power BI and rely on other members to provide answers.	https://community.fabric.microsoft.com/t5/Microsoft-Power-BI-Community/ct-p/powerbi
SQLBI	A third-party site maintained by Marco Russo and Alberto Ferrari. The site contains lots of free resources to help you with DAX.	https://www.sqlbi.com/
DAX Guide	A third-party alternative to Microsoft's DAX Reference created by the Italians. This site fleshes out the detail of DAX functions and provides links to helpful articles which describe in more detail how the functions work.	https://dax.guide/
Wise Owl	The Wise Owl website contains lots of free resources including videos, blogs and exercises to help you with learning DAX.	https://www.wiseowl.co.uk/resources/

© Copyright 2025

CHAPTER 2 - BASIC CALCULATED COLUMNS

2.1 Calculated Columns

A *calculated column* is a type of calculation you can create using DAX in Power BI. This chapter shows you the basics of writing DAX using calculated columns.

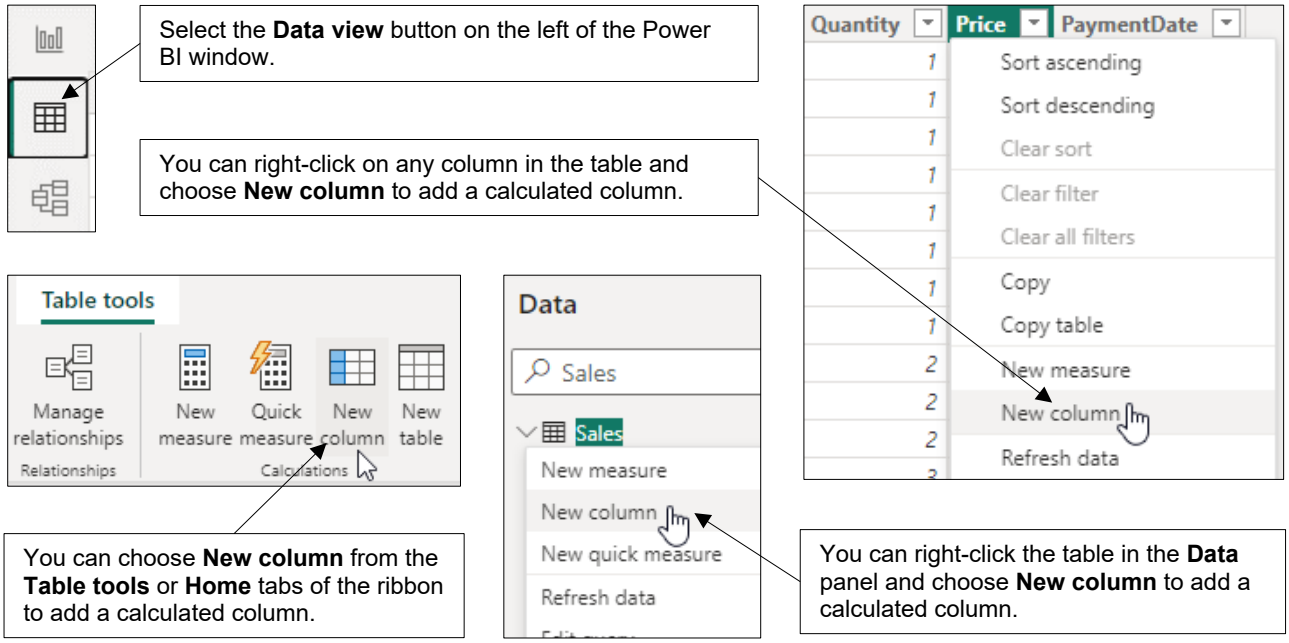
Key Features of Calculated Columns

You can see some of the key features of calculated columns in the table below:

Feature	Description
Created in data tables	You create a calculated column in a table in the data model. The calculated column can refer directly to any column in the same table.
Calculated immediately	A calculated column produces its results as soon as you enter it. The values are updated whenever the data model is refreshed.
Stores data in the model	A calculated column stores its results in the data model. Each calculated column you create increases the storage space required by the model.
Uses row context	The expression in a calculated column is evaluated for each row in the table. The row context provides the expression with access to values on the same row in the table.

Creating a Calculated Column

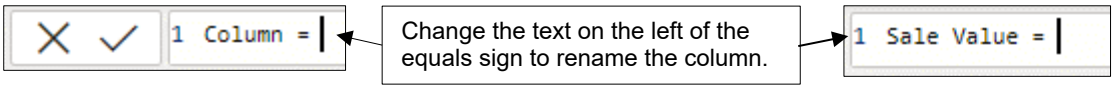
You can create a calculated column in any of the three Power BI views but, if you want to see the results of your calculation, it's best to select the **Data** view.



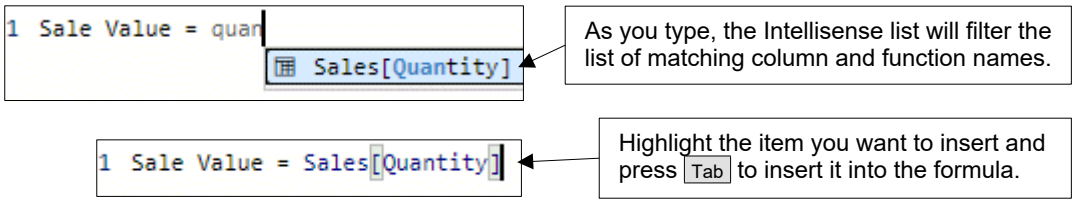
Entering a Formula

After choosing to create a calculated column you can enter your DAX code in the formula bar below the ribbon. The example below multiplies one column by another to create a new value:

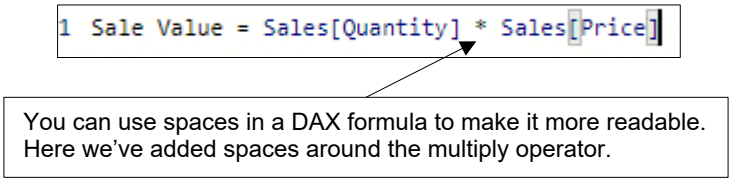
1) Start by giving the column a sensible name.



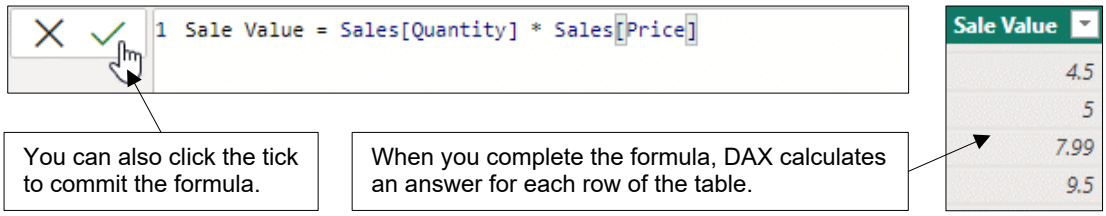
2) To reference a column in the same table, simply begin typing the column name.



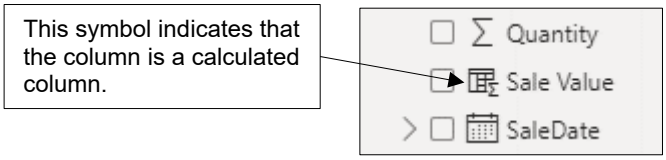
3) Type in an operator then reference the next column.



4) Press Enter to commit the formula.



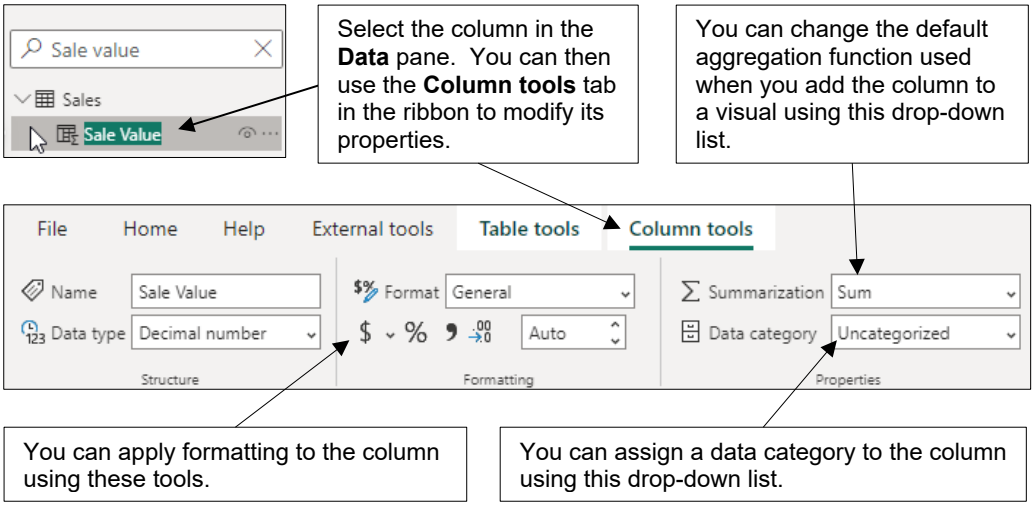
Your calculated column will appear in the table in the **Data** pane with a special symbol to indicate its status.



© Copyright 2025

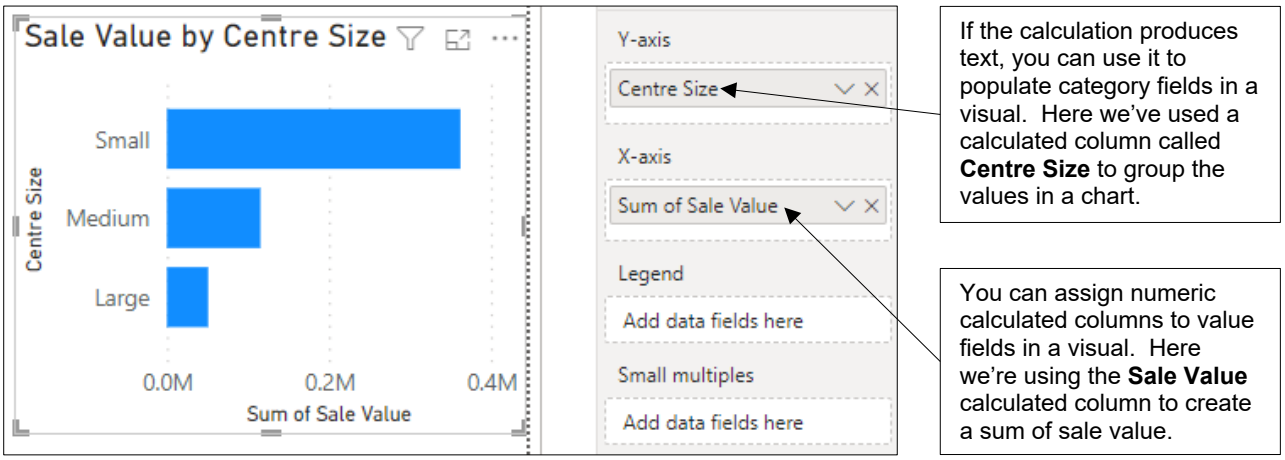
Calculated Column Properties

Once you've created a calculated column, you can modify it in the same way as any other column in your data model, as shown in the diagram below.



Using Calculated Columns in Visuals

You can use a calculated column to populate visuals in your report, just as for any other column in your data model.



© Copyright 2025

2.2 Referencing Columns and Tables

This section shows you various ways to refer to columns and tables in your DAX formulae.

Qualifying Column Names

When you select a column name from the Intellisense list, it will be automatically qualified with the name of the table to which it belongs. You don't have to include the table name, however.

```
1  Average Unit Size = Centre[SquareMetres] / Centre[NumberUnits]
```

```
1  Average Unit Size = [SquareMetres] / [NumberUnits]
```

These two formulae give the same result.

Although you don't always have to include the table name when referencing a column, it makes sense to do so for the following reasons:

- If you always include the table name you don't have to remember when it is or isn't required.
- It allows you to spot when you're referring to a column as opposed to, say, a measure.
- When a column name exists in multiple tables it makes the reference unambiguous.

Table Names

If the name of a table contains spaces, or it conflicts with another DAX keyword, you must enclose the table name in single quotes.

```
1  Sale value = 'Sales Table'[Price] * 'Sales Table'[Quantity]
```

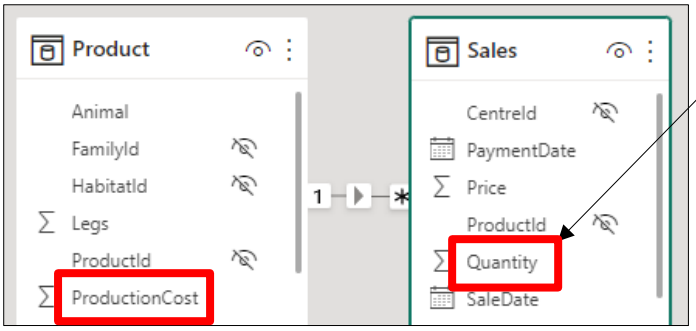
Sales Table must be enclosed in single quotes due to the space in the table name.

You can always enclose a table name in single quotes, even when it isn't required.

© Copyright 2025

Referencing Columns in Other Tables

In a calculated column you can only directly reference other columns which belong to the same table. This is a problem when your calculation needs to refer to columns in other tables!



We'd like to create a calculated column in the **Sales** table which multiplies the **Quantity** by the **ProductionCost** in the **Product** table.

```
1  Sale cost = Sales[Quantity] * Product[ProductionCost]
```

A single value for column 'ProductionCost' in table 'Product' cannot be

When we reference the **ProductionCost** column directly, the formula results in an error.

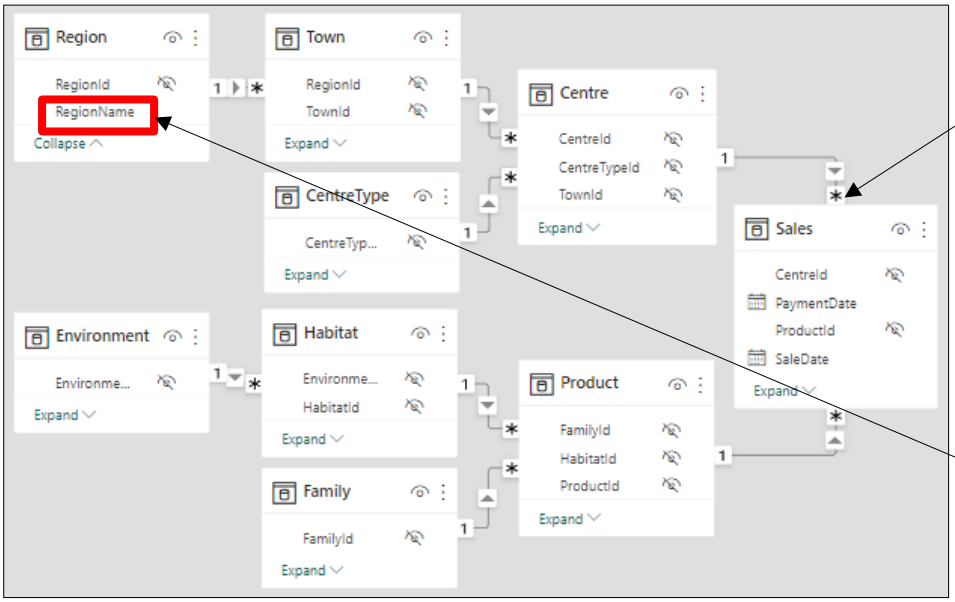
The Related Function

The key to solving the above problem is that the **Product** and **Sales** tables have a relationship. This allows us to use the **RELATED** function to reference a column in a related table.

```
1  Sale cost = Sales[Quantity] * RELATED(Product[ProductionCost])
```

We can successfully reference the **ProductionCost** column by using the **RELATED** function.

You can reference columns from a table at the **1** end of a relationship, regardless of how many steps it takes. In our model, the **Sales** table sits at the many end of every relationship:



A calculated column in the **Sales** table can reference any other column in the data model using the **RELATED** function.

In the **Sales** table you could return the **RegionName** with this formula:

=RELATED(Region[RegionName])

2.3 Editing DAX Code

Although you can write a DAX formula as a continuous stream of code, there are several things you can do to make your code more readable.

```
1  Sale Value % of Total = IF(NOT(ISBLANK(SUMX(Sales,Sales[Price]*Sales
   [Quantity]))),DIVIDE(SUMX(Sales,Sales[Price]*Sales[Quantity]),
   CALCULATE(SUMX(Sales,Sales[Price]*Sales[Quantity]),REMOVEFILTERS())))
```

Laying out a formula like this makes it very difficult to work with!

It's not important to understand what the code in this section does – instead, focus on the techniques used to make it more readable.

Multiple Lines and Indenting

You can break a formula onto multiple lines and add tab spaces to make it more readable. You can use the following keys to add new lines and indenting to a formula:

Key	What it does
Shift + Enter or Alt + Enter	Adds a new line, and a tab level if appropriate.
Tab	Indents the highlighted lines one tab space.
Shift + Tab	Outdents the highlighted lines one tab space.

You can see an example of a formula with new lines and indenting in the diagram below:

When you use a function (in this case **DIVIDE**), open the round brackets then start a new line and add a tab space.

Close the brackets for a function at the same indent level as the line on which the brackets were opened. Use the vertical grey lines to help you put the brackets in the right place.

```
1  Sale Value % of Total = DIVIDE(
2      SUMX(
3          Sales,
4          Sales[Price]*Sales[Quantity]
5      ),
6      CALCULATE(
7          SUMX(
8              Sales,
9              Sales[Price]*Sales[Quantity]
10         ),
11         REMOVEFILTERS()
12     )
13 )
```

Each new function that you use should have its arguments indented one extra tab level.

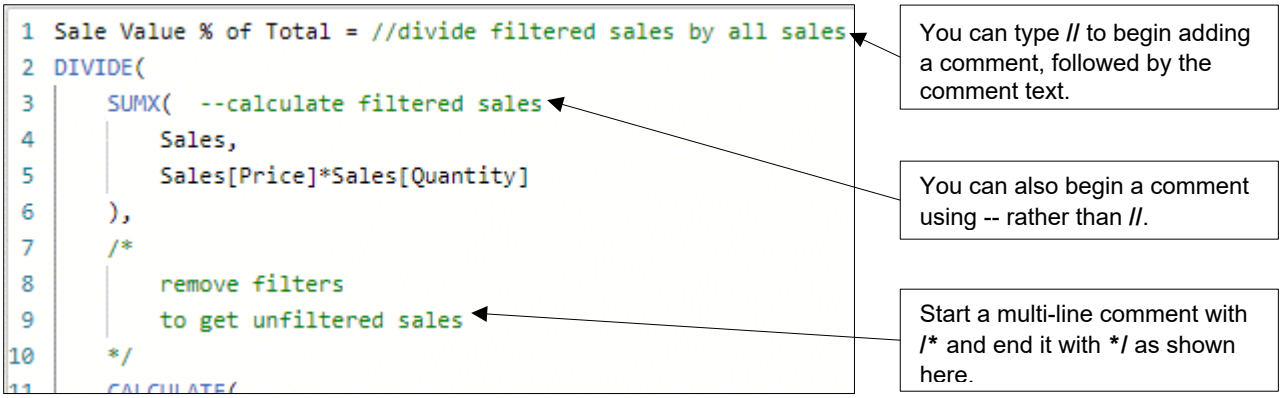
All of the arguments for a single function should be at the same tab level.

Use a comma at the end of a line to separate one argument from the next.

© Copyright 2025

Comments

You can use *comments* to annotate your code. You can add a comment at the end of any line after the = sign in a formula.



Keyboard Shortcuts

You can use a range of keyboard shortcuts to help you edit your DAX code. You can see some of these in the table below:

Key	What it does
Ctrl + G	Goto the specified line number.
Alt + ↑ / Alt + ↓	Move the line of code up / down.
Shift + Alt + ↑ / Shift + Alt + ↓	Copy the line of code up / down.
Ctrl + Shift + \	Jump to the paired bracket.
Alt + Left mouse click	Add a text cursor at the clicked position.
Ctrl + L	Select the line of code.
Ctrl + Shift + L	Select all occurrences of the current selection.
Ctrl + F2	Select all occurrences of the current word.
Ctrl + /	Comment/uncomment the line of code.
Ctrl + = / Ctrl + -	Zoom in / zoom out.
Ctrl + Space Bar	Show the Intellisense list.
Ctrl + I	Show and hide tooltips.
Ctrl + J	Expand and collapse the formula bar.

© Copyright 2025

Wise Owl
Training

Wise Owl
Training

Wise Owl
Training

Wise Owl
Training

Wise Owl
Training

Wise Owl
Training

Wise Owl
Training

Wise Owl
Training

Wise Owl
Training

Wise Owl
Training

What we do!

		Basic training	Advanced training	Systems / consultancy
Office	Microsoft Excel			
	VBA macros			
	Office Scripts			
	Microsoft Access			
Power BI, etc	Power BI and DAX			
	Power Apps			
	Power Automate (both)			
SQL Server	SQL			
	Reporting Services			
	Report Builder			
	Integration Services			
	Analysis Services			
Coding	Visual C#			
	VB programming			
	MySQL			
	Python			

Wise Owl
Training