POWERBI COMPLETE DAX GUIDE

Data Analysis Expressions (DAX) is a library of functions and operators that can be combined to build formulas and expressions in Power BI, Analysis Services, and Power Pivot in Excel data models.

Aggregation functions overview

AVERAGE

```
AVERAGE(<column>)
If the column contains text, no aggregation can be performed, and
the functions returns blanks.
If the column contains logical values or empty cells, those values
are ignored.
Cells with the value zero are included.

This function is not supported for use in DirectQuery mode when us
ed in calculated columns or row-level security (RLS) rules.
```

AVERAGEA

```
AVERAGEA(<column>)



also handles non-numeric data types according to the following rul
es:

Values that evaluates to TRUE count as 1.
Values that evaluate to FALSE count as 0 (zero).
Values that contain non-numeric text count as 0 (zero).
Empty text ("") counts as 0 (zero).



Transaction ID  Amount  Result
0000123          1       Counts as 1
0000124         20           Counts as 20
0000125         n/a          Counts as 0
0000126                      Counts as 0
0000126         TRUE         Counts as 1
```

AVERAGEX

AVERAGEX follows the same rules as AVERAGE. You cannot include non-numeric or null cells. Both the table and expression arguments are required.

COUNT

The COUNT function counts rows that contain the following kinds of values:

Numbers
Dates
Strings

When the function finds no rows to count, it returns a blank.
Blank values are skipped. TRUE/FALSE values are not supported.
If you want to evaluate a column of TRUE/FALSE values, use the COUNTA function.

COUNTA

Unlike COUNT, COUNTA supports Boolean data type.

COUNTX

The COUNTX function counts only values, dates, or strings. If the function finds no rows to count, it returns a blank.
If you want to count logical values, use the COUNTAX function.

```
= COUNTX(FILTER(Product,RELATED(ProductSubcategory[EnglishProductSubcategoryName])="Caps"), Product[ListPrice])
```

COUNTAX

The COUNTAX function counts non-blank results when evaluating the result of an expression over a table.
COUNTAX(<table>,<expression>)

= COUNTAX(FILTER('Reseller',[Status]="Active"),[Phone])

COUNTBLANK

Counts the number of blank cells in a column.
COUNTBLANK(<column>)

COUNTROWS

This function can be used to count the number of rows in a base ta ble
COUNTROWS([<table>])

DISTINCTCOUNT

Counts the number of distinct values in a column.
DISTINCTCOUNT(<column>)
DISTINCTCOUNT function ALSO counts the BLANK valu

DISTINCTCOUNTNOBLANK

Counts the number of distinct values in a column.
does not count the BLANK value.
= DISTINCTCOUNT(ResellerSales_USD[SalesOrderNumber])

MAX

Returns the largest value in a column, or between two scalar expressions.
MAX(<column>)
MAX(<expression1>, <expression2>)

When comparing two expressions, blank is treated as 0 when comparing.
That is, Max(1, Blank()) returns 1, and Max(-1, Blank()) returns 0.
If both arguments are blank, MAX returns a blank.
If either expression returns a value which is not allowed, MAX returns an error.

TRUE/FALSE values are not supported. If you want to evaluate a column of TRUE/FALSE values, use the MAXA function.

= Max([TotalSales], [TotalPurchases])

MAXA

Returns the largest value in a column.

MAXA(<column>)
looks for the largest value among the following types of values:
Numbers
Dates

Logical values, such as TRUE and FALSE. Rows that evaluate to TRUE count as 1;
rows that evaluate to FALSE count as 0 (zero).

Empty cells are ignored. If the column contains no values that can be used, MAXA returns 0 (zero).
If you want to compare text values, use the MAX function.

= MAXA([ResellerMargin])
= MAXA([TransactionDate])

MAXX

Returns the highest value that results from evaluating an expressi
on for each row of a table.
MAXX(<table>,<expression>,[<variant>])

variant: (Optional) If TRUE, and if there are variant or mixed val
ue types,
the highest value based on ORDER BY DESC is returned.

Of the values to evaluate, only the following are counted:
Numbers
Texts
Dates

Blank values are skipped. TRUE/FALSE values are not supported.

If the expression has variant or mixed value types such as text an
d number,
then by default MAXX considers only numbers. If <variant> = TRUE,
the maximum value is returned.

= MAXX(FILTER(InternetSales,[SalesTerritoryCode]="5"), InternetSal
es[TaxAmt]+ InternetSales[Freight])

MIN

Returns the smallest value in a column, or between two scalar expr
essions.
MIN(<column>)
MIN(<expression1>, <expression2>)

The following types of values in the columns are counted:
Numbers
Texts
Dates
Blanks

When comparing expressions, blank is treated as 0 when comparing.
That is, Min(1,Blank()) returns 0, and Min(-1, Blank()) returns
-1.
If both arguments are blank, MIN returns a blank.
If either expression returns a value which is not allowed, MIN ret
urns an error.

TRUE/FALSE values are not supported. If you want to evaluate a col
umn of TRUE/FALSE values, use the MINA function.

MINA

MINA(<column>)
Returns the smallest value in a column.
If the column contains no values, MINA returns 0 (zero).
Rows in the column that evaluates to logical values,
such as TRUE and FALSE are treated as 1 if TRUE and 0 (zero) if FA
LSE.Empty cells are ignored.

MINX

Returns the lowest value that results from evaluating an expressio
n for each row of a table.
MINX(<table>, < expression>,[<variant>])

variant: (Optional) If TRUE, and if there are variant or mi
xed value types,
the lowest value based on ORDER BY ASC is returned.

Blank values are skipped. TRUE/FALSE values are not supported.

If the expression has variant or mixed value types such as text an
d number,
then by default MINX considers only numbers. If <variant> = TRUE,
the minimum value is returned.

= MINX(FILTER(InternetSales, [SalesTerritoryKey] = 5),[Freight])

= MINX(FILTER(InternetSales, InternetSales[SalesTerritoryKey] =
5), InternetSales[Freight] + InternetSales[TaxAmt])

PRODUCT

Returns the product of the numbers in a column.
PRODUCT(<column>)
Return value: A decimal number.

Only the numbers in the column are counted. Blanks, logical value
s, and text are ignored. For example,
PRODUCT(Table[Column]) is equivalent to PRODUCTX(Table, Table[C
olumn]).

PRODUCTX

Returns the product of an expression evaluated for each row in a t
able.
PRODUCTX(<table>, <expression>)
Only the numbers in the column are counted. Blanks, logical value
s, and text are ignored.

= [PresentValue] * PRODUCTX(AnnuityPeriods, 1+[FixedInterestRate]
)

SUM

Adds all the numbers in a column.
SUM(<column>)

SUMX

Returns the sum of an expression evaluated for each row in a tabl
e.
SUMX(<table>, <expression>)

Only the numbers in the column are counted. Blanks, logical value
s, and text are ignored.
= SUMX(FILTER(InternetSales, InternetSales[SalesTerritoryID]=5),[F
reight])

In [ ]:
```
1
```

Date and time functions overview

These functions help you create calculations based on dates and time. Many of the functions in DAX are similar to the Excel date and time functions. However, DAX functions use a datetime data type, and can take values from a column as an argument.

CALENDAR

Returns a table with a single column named "Date" that contains a
contiguous set of dates.
CALENDAR(<start_date>, <end_date>)

An error is returned if start_date is greater than end_date.
= CALENDAR (DATE (2015, 1, 1), DATE (2021, 12, 31))
= CALENDAR (MINX (Sales, [Date]), MAXX (Forecast, [Date]))

CALENDARAUTO

Returns a table with a single column named "Date" that contains a contiguous set of dates.
The range of dates is calculated automatically based on data in th e model.

CALENDARAUTO([fiscal_year_end_month])

The earliest date in the model which is not in a calculated column or calculated table is taken as the MinDate.
The latest date in the model which is not in a calculated column o r calculated table is taken as the MaxDate.
The date range returned is dates between the beginning of the fisc al year associated with MinDate and the end of the fiscal year ass ociated with MaxDate.

An error is returned if the model does not contain any datetime va lues which are not in calculated columns or calculated tables.

In this example, the MinDate and MaxDate in the data model are Jul y 1, 2010 and June 30, 2011.

CALENDARAUTO() will return all dates between January 1, 2010 and D ecember 31, 2011.

CALENDARAUTO(3) will return all dates between April 1, 2010 and Ma rch 31, 2012.

DATE

Returns the specified date in datetime format.
DATE(<year>, <month>, <day>)

The value of the year argument can include one to four digits. Dates beginning with March 1, 1900 are supported.

For values greater than 9999 or less than zero (negative values), the function returns a #VALUE! error.

DATEDIFF

Returns the number of interval boundaries between two dates.
DATEDIFF(<Date1>, <Date2>, <Interval>)
Interval The interval to use when comparing dates. The valu e can be one of the following:

- SECOND
- MINUTE
- HOUR
- DAY
- WEEK
- MONTH
- QUARTER
- YEAR

```
    EVALUATE
  VAR StartDate =  DATE ( 2019, 07, 01 )
  VAR EndDate =    DATE ( 2021, 12, 31 )
  RETURN
      {
          ( "Year",     DATEDIFF ( StartDate, EndDate, YEAR )
),
          ( "Quarter",  DATEDIFF ( StartDate, EndDate, QUARTER
) ),
          ( "Month",    DATEDIFF ( StartDate, EndDate, MONTH )
),
          ( "Week",     DATEDIFF ( StartDate, EndDate, WEEK )
),
          ( "Day",      DATEDIFF ( StartDate, EndDate, DAY ) )
      }
```

DATEVALUE

Converts a date in text format to a date in datetime format.
DATEVALUE(date_text)

DAY

Returns the day of the month, a number from 1 to 31.
= DAY("3-4-1007")
= DAY("March 4 2007")
= IF(DAY([SalesDate])=10,"promotion","")

EDATE

Returns the date that is the indicated number of months before or after the start date. Use EDATE to calculate maturity dates or due dates that fall on the same day of the month as the date of issue.

EDATE(<start_date>, <months>)

MONTHS: An integer that represents the number of months be fore or after start_date.
= EDATE([TransactionDate],3)

EOMONTH

Returns the date in datetime format of the last day of the month, before or after a specified number of months. Use EOMONTH to calcu late maturity dates or due dates that fall on the last day of the month

EOMONTH(<start_date>, <months>)
The following expression returns May 31, 2008, because the months argument is rounded to 2.
= EOMONTH("March 3, 2008",1.5)

HOUR

Returns the hour as a number from 0 (12:00 A.M.) to 23 (11:00 P. M.).
HOUR(<datetime>)

Return value: An integer number from 0 to 23.
= HOUR("March 3, 2008 3:00 PM")

MINUTE

Returns the minute as a number from 0 to 59, given a date and time value.
MINUTE(<datetime>)
= MINUTE("March 23, 2008 1:45 PM")

MONTH

Returns the month as a number from 1 (January) to 12 (December).
MONTH(<datetime>)
= MONTH("March 3, 2008 3:45 PM")

NETWORKDAYS

Returns the number of whole workdays between two dates (inclusiv

NOW

The NOW function is useful when you need to display the current da
te and time on a worksheet
or calculate a value based on the current date and time,
and have that value updated each time you open the worksheet.

```
NOW()
```

In the Power BI Service, the result of the NOW function is always
in the UTC timezone. Universal Time Coordinated (UTC)
The following example returns the current date and time plus 3.5 d
ays:

```
= NOW()+3.5
```

QUARTER

Returns the quarter as a number from 1 (January – March) to 4 (Oct
ober – December).
QUARTER(<date>)
If the input value is BLANK, the output value is also BLANK.

```
EVALUATE { QUARTER(DATE(2019, 2, 1)), QUARTER(DATE(2018, 12, 31))
}
```

```
EVALUATE
ADDCOLUMNS(
    FILTER(
        VALUES(
            FactInternetSales[OrderDate]),
            [OrderDate] >= DATE(2008, 3, 31) && [OrderDate] <= DAT
E(2008, 4, 1)
        ),
    "Quarter", QUARTER([OrderDate])
)
```

SECOND

Returns the seconds of a time value, as a number from 0 to 59.
SECOND(<time>)

```
= SECOND("March 3, 2008 12:00:03")
```

TIME

Converts hours, minutes, and seconds given as numbers to a time in datetime format.
TIME(hour, minute, second)

Term	Definition
hour	**Import mode:** A number from 0 to 32767 representing the hour. Any value greater than 23 will be divided by 24 and the remainder will be treated as the hour value, represented as a fraction of a day. For example, TIME(27,0,0) = TIME(3,0,0) = 3:00:00 AM **DirectQuery mode:** A number from 0 to 23 representing the hour.
minute	**Import mode:** A number from 0 to 32767 representing the minute. Any value greater than 59 minutes will be converted to hours and minutes. Any value greater than 1440 (24 hours) does not alter the date portion - instead, it will be divided by 1440 and the remainder will be treated as the minute value, represented as a fraction of a day. For example, TIME(0,2190,0) = TIME(0,750,0) = TIME(12,30,0) = 12:30:00 PM **DirectQuery mode:** A number from 0 to 59 representing the minute.
second	**Import mode:** A number from 0 to 32767 representing the second. Any value greater than 59 will be converted to hours, minutes, and seconds. For example, TIME(0,0,2000) = TIME(0,33,20) = 12:33:20 AM **DirectQuery mode:** A number from 0 to 59 representing the second.

= TIME(27,0,0)
= TIME(12,30,0)

TIMEVALUE

Converts a time in text format to a time in datetime format.
TIMEVALUE(time_text)
= TIMEVALUE("20:45:30")

TODAY

Returns the current date.
TODAY()

= YEAR(TODAY())-1963

UTCNOW

Returns the current UTC date and time.
UTCNOW()

The result of the UTCNOW function changes only when the formula is

UTCTODAY

Returns the current UTC date.
UTCTODAY()

UTCTODAY returns the time value 12:00:00 PM for all dates.
The UTCNOW function is similar but returns the exact time and dat
e.

EVALUATE { FORMAT(UTCTODAY(), "General Date") }

WEEKDAY

Returns a number from 1 to 7 identifying the day of the week of a
date.
By default the day ranges from 1 (Sunday) to 7 (Saturday).

WEEKDAY(<date>, <return_type>)

Term	Definition
date	A date in **datetime** format. Dates should be entered by using the DATE function, by using expressions that result in a date, or as the result of other formulas.
return_type	A number that determines the Return value: Return type: **1**, week begins on Sunday (1) and ends on Saturday (7). numbered 1 through 7. Return type: **2**, week begins on Monday (1) and ends on Sunday (7). Return type: **3**, week begins on Monday (0) and ends on Sunday (6).numbered 1 through 7.

= WEEKDAY([HireDate]+1)

WEEKNUM

Returns the week number for the given date according to the return_type value. The week number indicates where the week falls numerically within a year.

WEEKNUM(<date>[, <return type>])

YEAR

Returns the year of a date as a four digit integer in the range 1900-9999.
YEAR(<date>)

Return value: An integer in the range 1900-9999.
= YEAR("March 2007")

= YEAR(TODAY())

YEARFRAC

Calculates the fraction of the year represented by the number of whole days between two dates.
YEARFRAC(<start_date>, <end_date>, <basis>)

Term	Definition
start_date	The start date in **datetime** format.
end_date	The end date in **datetime** format.
basis	(Optional) The type of day count basis to use. All arguments are truncated to integers. Basis - Description 0 - US (NASD) 30/360 (Default value) 1 - Actual/actual 2 - Actual/360 3 - Actual/365 4 - European 30/360

If start_date or end_date are not valid dates, YEARFRAC returns an error.
If basis < 0 or if basis > 4, YEARFRAC returns an error.

= YEARFRAC(Orders[TransactionDate],Orders[ShippingDate])

= YEARFRAC("Jan 1 2007","Mar 1 2007")

In [ ]:
```
1
```

Filter functions overview

The filter and value functions in DAX are some of the most complex and powerful,and differ greatly from Excelfunctions.
The lookup functions work by using tables and relationships, like a database. The filtering functions let you manipulate data contex t to create dynamic calculations.

In [ ]:
```
1
```