DAX for
Power-BI

MASTERING DAX

YOUR COMPLETE GUIDE TO FUNCTIONS & SYNTAX

[FUNCTIONS & SYNTAX]

Aggregation Functions:

Functions	Description
AVERAGE	Calculates the average of a set of values.
COUNT	Counts the number of rows in a table, or the number of distinct values in a column.
COUNTA	Counts the number of non-blank values in a column.
COUNTAX	Counts the number of values in a table expression.
COUNTBLANK	Counts the number of blank values in a column.
COUNTROWS	Counts the number of rows in a table or table expression.
DISTINCTCOUNT	Counts the number of distinct values in a column.
MAX	Returns the largest value in a set of values.
MAXA	Returns the largest value in a column, including text and logical values.
MAXX	Calculates the maximum value of an expression evaluated for each row in a table or filter context.
MEDIAN	Calculates the median of a set of values.
MIN	Returns the smallest value in a set of values.
MINA	Returns the smallest value in a column, including text and logical values.
MINX	Calculates the minimum value of an expression evaluated for each row in a table or filter context.
PRODUCT	Multiplies all values in a column together.
SUM	Calculates the sum of a set of values.
SUMX	Calculates the sum of an expression evaluated for each row in a table or filter context.
VAR	Calculates the variance of a set of values.
VARP	Calculates the variance of a set of values, including text and logical values.

Logical Functions:

Functions	Description
AND	Returns TRUE if all arguments are TRUE, and FALSE otherwise.
IF	Returns one value if a logical expression is TRUE and another value if it is FALSE.
IFERROR	Returns a value if a formula results in an error, and another value if it does not.
NOT	Returns the opposite of a logical expression.
OR	Returns TRUE if any argument is TRUE, and FALSE otherwise.
SWITCH	Evaluates a list of expressions and returns a result based on the first expression that is TRUE

Aggregate Functions for Date/Time Intelligence:

Functions	Description
CLOSINGBALANCEMONTH	Calculates the closing balance for a month in a time period.
CLOSINGBALANCEQUARTER	Calculates the closing balance for a quarter in a time period.
CLOSINGBALANCEYEAR	Calculates the closing balance for a year in a time period.
TOTALYTD	Calculates a total for a given year-to-date period.
SAMEPERIODLASTYEAR	Returns a table that contains a parallel period in the previous year.
DATESYTD	Returns a table of dates for a given year-to-date period.
TOTALMTD	Calculates a total for a given month-to-date period.
TOTALQTD	Calculates a total for a given quarter-to-date period.
TOTALYTD	Calculates a total for a given year-to-date period.

Text Functions:

Functions	Description
CONCATENATE	Joins two or more text strings into one text string.
CONCATENATEX	Joins two or more expressions into one text string.
FORMAT	Formats a value with a specific format string.
LEFT	Returns the leftmost characters of a text string.
LEN	Returns the number of characters in a text string.
LOWER	Converts a text string to lowercase.
MID	Returns a specific number of characters from a text string, starting at a specified position.
REPLACE	Replaces characters within a text string.
REPT	Repeats a text string a specified number of times.
RIGHT	Returns the rightmost characters of a text string.
SUBSTITUTE	Replaces one text string with another.
TRIM	Removes leading and trailing spaces from a text string.
UNICHAR	Returns the Unicode character that is represented by a number.
UNICODE	Returns the Unicode number that corresponds to the first character in a text string.
UPPER	Converts a text string to uppercase.

Date/Time Functions:

Functions	Description
DATE	Creates a date from year, month, and day values.
DATEADD	Returns a date that is a specified number of intervals (such as days or months) before or after a specified date.
DATEDIFF	Calculates the difference between two dates, in days, months, quarters, or years.
DATEVALUE	Converts a text string to a date value.
DAY	Returns the day of the month for a date.
EOMONTH	Returns the last day of the month that is the indicated number of months before or after a specified date.
HOUR	Returns the hour of the day for a time value.
MINUTE	Returns the minute of the hour for a time value.
MONTH	Returns the month of the year for a date.
NOW	Returns the current date and time.
SECOND	Returns the second of the minute for a time value.
TIME	Creates a time value from hour, minute, and second values.
TIMEVALUE	Converts a text string to a time value.
TODAY	Returns the current date.
YEAR	Returns the year for a date.

Time Intelligence Functions:

Functions	Description
CLOSINGBALANCEMONTH	Calculates the balance of an account for the last day of the specified month.
CLOSINGBALANCEQUARTER	Calculates the balance of an account for the last day of the specified quarter.
CLOSINGBALANCEYEAR	Calculates the balance of an account for the last day of the specified year.
DATESBETWEEN	Returns a table of dates between two specified dates.
FIRSTDATE	Returns the first date in a column or table.
LASTDATE	Returns the last date in a column or table.
NEXTDAY	Returns the next day after a specified date.
PARALLELPERIOD	Returns a date in a previous or future period.
PREVIOUSDAY	Returns the previous day before a specified date.
SAMEPERIODLASTYEAR	Returns a date in the same period in the previous year.
TOTALYTD	Calculates a total for the year to date.
YEAR	Returns the year for a date.

Math Functions:

Functions	Description
ABS	Returns the absolute value of a number.
ACOS	Returns the arccosine of a number.
ASIN	Returns the arcsine of a number.
ATAN	Returns the arctangent of a number.
CEILING	Rounds a number up to the nearest integer or multiple of a specified value.
COS	Returns the cosine of an angle.
COT	Returns the cotangent of an angle.
EXP	Returns e raised to a power.
FLOOR	Rounds a number down to the nearest integer or multiple of a specified value.
LN	Returns the natural logarithm of a number.
LOG	Returns the logarithm of a number with a specified base.
LOG10	Returns the base-10 logarithm of a number.
MOD	Returns the remainder after division of two numbers.
PI	Returns the value of pi.
POWER	Raises a number to a power.
RADIANS	Converts degrees to radians.
RAND	Returns a random number between 0 and 1.
ROUND	Rounds a number to a specified number of decimal places.
SIGN	Returns the sign of a number.
SIN	Returns the sine of an angle.
SQRT	Returns the square root of a number.
TAN	Returns the tangent of an angle.
TRUNC	Truncates a number to a specified number of decimal places.

Information Functions:

Functions	Description
ISBLANK	Returns TRUE if a value is blank, and FALSE otherwise.
ISERROR	Returns TRUE if a value is any error value, and FALSE otherwise.
ISLOGICAL	Returns TRUE if a value is a logical value (TRUE or FALSE), and FALSE otherwise.
ISNONTEXT	Returns TRUE if a value is not text, and FALSE otherwise.
ISNUMBER	Returns TRUE if a value is a number, and FALSE otherwise.
ISTEXT	Returns TRUE if a value is text, and FALSE otherwise.

Statistical Functions:

Functions	Description
NORM.DIST	Returns the cumulative distribution function of a standard normal distribution.
NORM.INV	Returns the inverse of the cumulative distribution function of a standard normal distribution.
NORM.S.DIST	Returns the cumulative distribution function of a normal distribution with a specified mean and standard deviation.
NORM.S.INV	Returns the inverse of the cumulative distribution function of a normal distribution with a specified mean and standard deviation.
STDEV.P	Calculates the standard deviation of a population.
STDEV.S	Calculates the standard deviation of a sample.
VAR.P	Calculates the variance of a population.
VAR.S	Calculates the variance of a sample.

Filter Functions:

Functions	Description
ALL	Removes all filters from the data, returning the entire table.
ALLEXCEPT	Removes all filters except for specified columns.
ALLSELECTED	Returns all values currently selected, including those affected by slicers.
HASONEVALUE	Checks if there is only one value in a column.
HASONEFILTER	Checks if there is only one filter applied to a column.
CALCULATE	Evaluates an expression in a modified filter context.
CALCULATETABLE	Evaluates a table expression in a modified filter context.
FILTER	Returns a table that represents a subset of another table based on a condition.
KEEPFILTERS	Preserves the filters applied to the data model.
REMOVEFILTERS	Clears filters from the specified tables or columns.

Note that this list is not exhaustive and new functions may be added to DAX in the future. It's important to consult the official Microsoft documentation for the latest information and **Syntax** on DAX functions.

[EXPLANATION OF EACH FUNCTIONS WITH AN EXAMPLE]

Aggregation Functions:

AVERAGE: Calculates the average of a set of values.

Syntax: AVERAGE(<column>)

Example: To calculate the average of sales amounts in a table, use the following formula:

AVERAGE('Sales'[SalesAmount])

COUNT: Counts the number of rows in a table, or the number of distinct values in a column.

Syntax: COUNT(<column>)

Example: To count the number of rows in a table, use the following formula:

COUNT('Sales')

COUNTA: Counts the number of non-blank values in a column.

Syntax: COUNTA(<column>)

Example: To count the number of non-blank values in a column, use the following formula:

COUNTA('Sales'[Product])

COUNTAX: Counts the number of values in a table expression.

Syntax: COUNTAX(<table>, <expression>)

Example: To count the number of products with sales greater than $1000 in a table, use the following formula:

COUNTAX('Sales', IF('Sales'[SalesAmount] > 1000, 1, BLANK()))

COUNTBLANK: Counts the number of blank values in a column.

Syntax: COUNTBLANK(<column>)

Example: To count the number of blank values in a column, use the following formula:

COUNTBLANK('Sales'[Product])

COUNTROWS: Counts the number of rows in a table or table expression.

Syntax: COUNTROWS(<table>)

Example: To count the number of rows in a table, use the following formula:

COUNTROWS('Sales')

DISTINCTCOUNT: Counts the number of distinct values in a column.

Syntax: DISTINCTCOUNT(<column>)

Example: To count the number of distinct products in a table, use the following formula:

DISTINCTCOUNT('Sales'[Product])

MAX: Returns the largest value in a set of values.

Syntax: MAX(<column>)

Example: To find the maximum sales amount in a table, use the following formula:

MAX('Sales'[SalesAmount])

MAXA: Returns the largest value in a column, including text and logical values.

Syntax: MAXA(<column>)

Example: To find the maximum value in a column containing both numeric and text values, use the following formula:

MAXA('Sales'[Product])

MAXX: Calculates the maximum value of an expression evaluated for each row in a table or filter context.

Syntax: MAXX(<table>, <expression>)

Example: To find the maximum sales amount for each product in a table, use the following formula:

```
MAXX(GROUPBY('Sales', 'Sales'[Product], "TotalSales", SUMX(CURRENTGROUP(),
'Sales'[SalesAmount])), [TotalSales])
```

MEDIAN: Calculates the median of a set of values.

Syntax: MEDIAN(<column>)

Example: To calculate the median sales amount in a table, use the following formula:

```
MEDIAN('Sales'[SalesAmount])
```

MIN: Returns the smallest value in a set of values.

Syntax: MIN(<column>)

Example: To find the minimum sales amount in a table, use the following formula:

```
MIN('Sales'[SalesAmount])
```

MINA: Returns the smallest value in a column, including text and logical values.

Syntax: MINA(<column>)

Example: To find the minimum value in a column containing both numeric and text values, use the following formula:

```
MINA('Sales'[Product])
```

MINX: Calculates the minimum value of an expression evaluated for each row in a table or filter context.

Syntax: MINX(<table>, <expression>)

Example: To find the minimum sales amount for each product in a table, use the following formula:

```
MINX(GROUPBY('Sales', 'Sales'[Product], "TotalSales", SUMX(CURRENTGROUP(),
'Sales'[SalesAmount])), [TotalSales])
```

PRODUCT: Multiplies all values in a column together.

Syntax: PRODUCT(<column>)

Example: To calculate the product of sales amounts in a table, use the following formula:

PRODUCT('Sales'[SalesAmount])

SUM: Calculates the sum of a set of values.

Syntax: SUM(<column>)

Example: To find the total sales amount in a table, use the following formula:

SUM('Sales'[SalesAmount])

SUMX: Calculates the sum of an expression evaluated for each row in a table or filter context.

Syntax: SUMX(<table>, <expression>)

Example: To calculate the total sales amount for each product in a table, use the following formula:

SUMX(GROUPBY('Sales', 'Sales'[Product], "TotalSales", SUMX(CURRENTGROUP(), 'Sales'[SalesAmount])), [TotalSales])

VAR: Calculates the variance of a set of values.

Syntax: VAR(<column>)

Example: To calculate the variance of sales amounts in a table, use the following formula:

VAR('Sales'[SalesAmount])

VARP: Calculates the variance of a set of values, including text and logical values.

Syntax: VARP(<column>)

Example: To calculate the variance of a column containing both numeric and text values, use the following formula:

VARP('Sales'[Product])

Logical Functions:

Logical functions are used to evaluate logical expressions and produce true/false results or make decisions based on those results. Here are the definitions and examples for the logical functions:

AND: The AND function returns TRUE if all of the arguments are TRUE, and FALSE otherwise.

Syntax: =AND(logical1, [logical2], ...)

Example: =AND(1=1, "Hello"="Hello", 5>3) returns TRUE because all three arguments are TRUE.

IF: The IF function returns one value if a logical expression is TRUE and another value if it is FALSE.

Syntax: =IF(logical_test, [value_if_true], [value_if_false])

Example: =IF(5>3, "Yes", "No") returns "Yes" because 5 is greater than 3.

IFERROR: The IFERROR function returns a value if a formula results in an error, and another value if it does not.

Syntax: =IFERROR(value, value_if_error)

Example: =IFERROR(1/0, "Error") returns "Error" because the formula 1/0 results in an error.

NOT: The NOT function returns the opposite of a logical expression.

Syntax: =NOT(logical)

Example: =NOT(5>3) returns FALSE because 5 is greater than 3, but NOT negates the result.

OR: The OR function returns TRUE if any of the arguments are TRUE, and FALSE otherwise.

Syntax: =OR(logical1, [logical2], ...)

Example: =OR(1=2, "Hello"="World", 5>3) returns TRUE because the last argument is TRUE.

SWITCH: The SWITCH function evaluates a list of expressions and returns a result based on the first expression that is TRUE.

Syntax: =SWITCH(expression, value1, result1, [value2, result2], …, [default])

Example: =SWITCH(3, 1, "One", 2, "Two", 3, "Three") returns "Three" because the expression is 3, which matches the third value in the list.

Aggregate Functions for Date/Time Intelligence:

CLOSINGBALANCEMONTH: Calculates the closing balance for a month in a time period.

Syntax: CLOSINGBALANCEMONTH(<expression>, <dates>)

Example: To calculate the closing balance for each month in a table, use the following formula:

CLOSINGBALANCEMONTH(SUM('Sales'[SalesAmount]), 'Date'[Date])

CLOSINGBALANCEQUARTER: Calculates the closing balance for a quarter in a time period.

Syntax: CLOSINGBALANCEQUARTER(<expression>, <dates>)

Example: To calculate the closing balance for each quarter in a table, use the following formula:

CLOSINGBALANCEQUARTER(SUM('Sales'[SalesAmount]), 'Date'[Date])

CLOSINGBALANCEYEAR: Calculates the closing balance for a year in a time period.

Syntax: CLOSINGBALANCEYEAR(<expression>, <dates>)

Example: To calculate the closing balance for each year in a table, use the following formula:

CLOSINGBALANCEYEAR(SUM('Sales'[SalesAmount]), 'Date'[Date])

TOTALYTD: Calculates a total for a given year-to-date period.

Syntax: TOTALYTD(<expression>, <dates>)

Example: To calculate the year-to-date sales amount for each product in a table, use the following formula:

TOTALYTD(SUM('Sales'[SalesAmount]), 'Date'[Date])

SAMEPERIODLASTYEAR: Returns a table that contains a parallel period in the previous year.

Syntax: SAMEPERIODLASTYEAR(<dates>)

Example: To calculate the sales amount for the same period in the previous year, use the following formula:

CALCULATE(SUM('Sales'[SalesAmount]), SAMEPERIODLASTYEAR('Date'[Date]))

DATESYTD: Returns a table of dates for a given year-to-date period.

Syntax: DATESYTD(<dates>)

Example: To return a table of dates for the year-to-date period in a table, use the following formula:

DATESYTD('Date'[Date])

TOTALMTD: Calculates a total for a given month-to-date period.

Syntax: TOTALMTD(<expression>, <dates>)

Example: To calculate the month-to-date sales amount for each product in a table, use the following formula:

TOTALMTD(SUM('Sales'[SalesAmount]), 'Date'[Date])

TOTALQTD: Calculates a total for a given quarter-to-date period.

Syntax: TOTALQTD(<expression>, <dates>)

Example: To calculate the quarter-to-date sales amount for each product in a table, use the following formula:

TOTALQTD(SUM('Sales'[SalesAmount]), 'Date'[Date])

TOTALYTD: Calculates a total for a given year-to-date period.

Syntax: TOTALYTD(<expression>, <dates>)

Example: To calculate the year-to-date sales amount for each product in a table, use the following formula:

TOTALYTD(SUM('Sales'[SalesAmount]), 'Date'[Date])

SAMEPERIODLASTYEAR: Returns a table that contains a parallel period in the previous year.

Syntax: SAMEPERIODLASTYEAR(<Dates>)

Example: Assuming a table named Sales with columns Date and Amount, the following DAX formula returns the total sales amount for the same period in the previous year:

CALCULATE(SUM(Sales[Amount]), SAMEPERIODLASTYEAR(Sales[Date]))

DATESYTD: Returns a table of dates for a given year-to-date period.

Syntax: DATESYTD(<Dates>[, <YearEndDate>][, <YearEndDateColumnName>])

Example: Assuming a table named Sales with columns Date and Amount, the following DAX formula returns the total sales amount for the current year to date:

```
CALCULATE(SUM(Sales[Amount]), DATESYTD(Sales[Date]))
```

TOTALMTD: Calculates a total for a given month-to-date period.

Syntax: TOTALMTD(<Expression>, <Dates>[, <Filter>])

Example: Assuming a table named Sales with columns Date and Amount, the following DAX formula returns the total sales amount for the current month to date:

```
CALCULATE(SUM(Sales[Amount]), TOTALMTD(Sales[Date]))
```

TOTALQTD: Calculates a total for a given quarter-to-date period.

Syntax: TOTALQTD(<Expression>, <Dates>[, <Filter>])

Example: Assuming a table named Sales with columns Date and Amount, the following DAX formula returns the total sales amount for the current quarter to date:

```
CALCULATE(SUM(Sales[Amount]), TOTALQTD(Sales[Date]))
```

TOTALYTD: Calculates a total for a given year-to-date period.

Syntax: TOTALYTD(<Expression>, <Dates>[, <Filter>][, <YearEndDate>][, <YearEndDateColumnName>])

Example: Assuming a table named Sales with columns Date and Amount, the following DAX formula returns the total sales amount for the current year to date:

```
CALCULATE(SUM(Sales[Amount]), TOTALYTD(Sales[Date]))
```

Text Functions:

CONCATENATE: The CONCATENATE function combines two or more text strings into a single text string.

Syntax: CONCATENATE(text1, [text2], ...)

Example: =CONCATENATE("Power", " ", "BI")

Output: "Power BI"

CONCATENATEX: The CONCATENATEX function joins the results of an expression evaluated for each row of a table, with a delimiter between each result.

Syntax: CONCATENATEX(table, expression, [delimiter])

Example: =CONCATENATEX(Orders, Orders[Product], ", ")

Output: "Product A, Product B, Product C"

FORMAT: The FORMAT function formats a value with a specific format string.

Syntax: FORMAT(value, format_text)

Example: =FORMAT(1234.567, "0.00")

Output: "1234.57"

LEFT: The LEFT function returns the specified number of characters from the start of a text string.

Syntax: LEFT(text, num_chars)

Example: =LEFT("Power BI", 5)

Output: "Power"

LEN: The LEN function returns the number of characters in a text string.

Syntax: LEN(text)

Example: =LEN("Power BI")

Output: 7

LOWER: The LOWER function converts a text string to lowercase.

Syntax: LOWER(text)

Example: =LOWER("POWER BI")

Output: "power bi"

MID: The MID function returns a specific number of characters from a text string, starting at a specified position.

Syntax: MID(text, start_num, num_chars)

Example: =MID("Power BI", 3, 4)

Output: "wer "

REPLACE: The REPLACE function replaces a specified number of characters within a text string with new text.

Syntax: REPLACE(old_text, start_num, num_chars, new_text)

Example: =REPLACE("Power BI", 6, 2, "Query")

Output: "Power Query"

REPT: The REPT function repeats a text string a specified number of times.

Syntax: REPT(text, num_times)

Example: =REPT("Power BI", 3)

Output: "Power BIPower BIPower BI"

RIGHT: The RIGHT function returns the specified number of characters from the end of a text string.

Syntax: RIGHT(text, num_chars)

Example:=RIGHT("Power BI", 2)

Output: "BI"

SUBSTITUTE: The SUBSTITUTE function replaces one text string with another in a given text string. If the optional instance_num argument is provided, the function will replace only the specified instance of the old_text.

Syntax: SUBSTITUTE(text, old_text, new_text, [instance_num])

Example: =SUBSTITUTE("Power BI", "BI", "Pivot")

Output: "Power Pivot"

TRIM: The TRIM function removes leading and trailing spaces from a text string.

Syntax: TRIM(text)

Example:=TRIM(" Power BI ")

Output: "Power BI"

UNICHAR:The UNICHAR function returns the Unicode character that is represented by a number.

Syntax: UNICHAR(number)

Example: =UNICHAR(65)

Output: "A"

UNICODE: The UNICODE function returns the Unicode number that corresponds to the first character in a text string.

Syntax: UNICODE(text)

Example: =UNICODE("A")

Output: 65

UPPER: Converts a text string to uppercase.

Syntax: UPPER(<text>)

Example: UPPER("Hello World") returns "HELLO WORLD"

Date/Time Functions:

DATE: Creates a date from year, month, and day values.

Syntax: DATE(year, month, day)

Example: DATE(2022, 4, 12) will return the date 4/12/2022.

DATEADD: Returns a date that is a specified number of intervals (such as days or months) before or after a specified date.

Syntax: DATEADD(start_date, number_of_intervals, interval)

Example: DATEADD(Dates[Date], -7, DAY) will return the date that is seven days before the date in the Dates[Date] column.

DATEDIFF: Calculates the difference between two dates, in days, months, quarters, or years.

Syntax: DATEDIFF(start_date, end_date, interval)

Example: DATEDIFF(Dates[Start Date], Dates[End Date], DAY) will return the number of days between the start and end dates in the Dates table.

DATEVALUE: Converts a text string to a date value.

Syntax: DATEVALUE(date_text)

Example: DATEVALUE("4/12/2022") will return the date 4/12/2022.

DAY: Returns the day of the month for a date.

Syntax: DAY(date)

Example: DAY(Dates[Date]) will return the day of the month for the date in the Dates[Date] column.

EOMONTH: Returns the last day of the month that is the indicated number of months before or after a specified date.

Syntax: EOMONTH(start_date, number_of_months)

Example: EOMONTH(Dates[Date], 3) will return the last day of the month that is three months after the date in the Dates[Date] column.

HOUR: Returns the hour of the day for a time value.

Syntax: HOUR(time)

Example: HOUR(Times[Time]) will return the hour of the day for the time in the Times[Time] column.

MINUTE: Returns the minute of the hour for a time value.

Syntax: MINUTE(time)

Example: MINUTE(Times[Time]) will return the minute of the hour for the time in the Times[Time] column.

MONTH: Returns the month of the year for a date.

Syntax: MONTH(date)

Example: MONTH(Dates[Date]) will return the month of the year for the date in the Dates[Date] column.

NOW: Returns the current date and time.

Syntax: NOW()

Example: NOW() will return the current date and time.

SECOND: Returns the second of the minute for a time value.

Syntax: SECOND(time)

Example: SECOND(Times[Time]) will return the second of the minute for the time in the Times[Time] column.

TIME: Creates a time value from hour, minute, and second values.

Syntax: TIME(hour, minute, second)

Example: TIME(9, 30, 0) will return the time 9:30:00 AM.

TIMEVALUE: Converts a text string to a time value.

Syntax: TIMEVALUE(time_text)

Example: TIMEVALUE("9:30 AM") will return the time 9:30:00 AM.

TODAY: Returns the current date.

Syntax: TODAY()

Example: TODAY() will return the current date.

YEAR: Returns the year for a date.

Syntax: YEAR(date)

Example: YEAR(Dates[Date]) will return the year for the date in the Dates[Date] column.

Time Intelligence Functions:

Time Intelligence functions in Power BI are specifically designed to perform calculations and analysis based on dates and time periods. Here are the explanations and examples of each Time Intelligence function:

CLOSINGBALANCEMONTH: Calculates the balance of an account for the last day of the specified month.

Syntax: CLOSINGBALANCEMONTH(expression, dates)

Example: CLOSINGBALANCEMONTH(SUM(Sales[Amount]), 'Date'[Date])

CLOSINGBALANCEQUARTER: Calculates the balance of an account for the last day of the specified quarter.

Syntax: CLOSINGBALANCEQUARTER(expression, dates)

Example: CLOSINGBALANCEQUARTER(SUM(Sales[Amount]), 'Date'[Date])

CLOSINGBALANCEYEAR: Calculates the balance of an account for the last day of the specified year.

Syntax: CLOSINGBALANCEYEAR(expression, dates)

Example: CLOSINGBALANCEYEAR(SUM(Sales[Amount]), 'Date'[Date])

DATESBETWEEN: Returns a table of dates between two specified dates.

Syntax: DATESBETWEEN(start_date, end_date)

Example: DATESBETWEEN(DATE(2022, 1, 1), DATE(2022, 12, 31))

FIRSTDATE: Returns the first date in a column or table.

Syntax: FIRSTDATE(dates)

Example: FIRSTDATE('Date'[Date])

LASTDATE: Returns the last date in a column or table.

Syntax: LASTDATE(dates)

Example: LASTDATE('Date'[Date])

NEXTDAY: Returns the next day after a specified date.

Syntax: NEXTDAY(date)

Example: NEXTDAY('Date'[Date])

PARALLELPERIOD: Returns a date in a previous or future period.

Syntax: PARALLELPERIOD(dates, n, interval)

Example: PARALLELPERIOD('Date'[Date], -1, YEAR)

PREVIOUSDAY: Returns the previous day before a specified date.

Syntax: PREVIOUSDAY(date)

Example: PREVIOUSDAY('Date'[Date])

SAMEPERIODLASTYEAR: Returns a date in the same period in the previous year.

Syntax: SAMEPERIODLASTYEAR(date)

Example: SAMEPERIODLASTYEAR('Date'[Date])

TOTALYTD: Calculates a total for the year to date.

Syntax: TOTALYTD(expression, dates, filter)

Example: TOTALYTD(SUM(Sales[Amount]), 'Date'[Date])

YEAR: Returns the year for a date.

Syntax: YEAR(date)

Example: YEAR('Date'[Date])

These Time Intelligence functions are useful for performing calculations based on specific time periods, comparing data across different periods, and analyzing trends and patterns over time.

Math Functions:

ABS: ABS function returns the absolute value of a number. Absolute value refers to the magnitude of a number without taking into account its sign. If the number is negative, ABS function returns its positive value.

Syntax: ABS(number)
Example: ABS(-5) returns 5

ACOS: ACOS function returns the arccosine of a number, which is the angle whose cosine is the given number.

Syntax: ACOS(number)
Example: ACOS(0.5) returns 1.047197551

ASIN: ASIN function returns the arcsine of a number, which is the angle whose sine is the given number.

Syntax: ASIN(number)
Example: ASIN(0.5) returns 0.523598776

ATAN: ATAN function returns the arctangent of a number, which is the angle whose tangent is the given number.

Syntax: ATAN(number)
Example: ATAN(1) returns 0.785398163

CEILING: CEILING function rounds a number up to the nearest integer or multiple of a specified value.

Syntax: CEILING(number, [significance])
Example: CEILING(4.3) returns 5, CEILING(4.3, 0.5) returns 4.5

COS: COS function returns the cosine of an angle in radians.

Syntax: COS(number)
Example: COS(0) returns 1

COT: COT function returns the cotangent of an angle in radians.

Syntax: COT(number)
Example: COT(0.785) returns 0.618

EXP: EXP function returns the value of e raised to a power.

Syntax: EXP(number)
Example: EXP(1) returns 2.718

FLOOR: FLOOR function rounds a number down to the nearest integer or multiple of a specified value.

Syntax: FLOOR(number, [significance])
Example: FLOOR(4.7) returns 4, FLOOR(4.7, 0.5) returns 4.5

LN: LN function returns the natural logarithm of a number, which is the logarithm to the base e.

Syntax: LN(number)
Example: LN(2.718) returns 1

LOG: LOG function returns the logarithm of a number with a specified base.

Syntax: LOG(number, [base])
Example: LOG(8, 2) returns 3

LOG10: LOG10 function returns the base-10 logarithm of a number.

Syntax: LOG10(number)
Example: LOG10(100) returns 2

MOD: MOD function returns the remainder after division of two numbers.

Syntax: MOD(number, divisor)
Example: MOD(10, 3) returns 1

PI: PI function returns the value of pi.

Syntax: PI()
Example: PI() returns 3.141592654

POWER: POWER function raises a number to a power.

Syntax: POWER(number, power)
Example: POWER(2, 3) returns 8

RADIANS: RADIANS function converts degrees to radians.

Syntax: RADIANS(number)
Example: RADIANS(180) returns 3.141592654

RAND: Returns a random number between 0 and 1. The function takes no arguments.

Syntax: =RAND()
Example: =RAND() will return a random number between 0 and 1 each time the worksheet is calculated.

ROUND: Rounds a number to a specified number of decimal places.

Syntax: =ROUND(number, num_digits)
Example: =ROUND(3.14159, 2) will return 3.14, rounding the number to 2 decimal places.

SIGN: Returns the sign of a number. Returns 1 if the number is positive, -1 if it is negative, and 0 if it is zero.

Syntax: =SIGN(number)
Example: =SIGN(-10) will return -1, indicating that the number is negative.

SIN: Returns the sine of an angle.

Syntax: =SIN(angle)
Example: =SIN(45) will return 0.707106781186548, which is the sine of 45 degrees.

SQRT: Returns the square root of a number.

Syntax: =SQRT(number)

Example: =SQRT(16) will return 4, which is the square root of 16.

TAN: Returns the tangent of an angle.

Syntax: =TAN(angle)
Example: =TAN(45) will return 1, which is the tangent of 45 degrees.

TRUNC: Truncates a number to a specified number of decimal places.

Syntax: =TRUNC(number, num_digits)
Example: =TRUNC(3.14159, 2) will return 3.14, truncating the number to 2 decimal places.

Information Functions:

These functions look at a table or column provided as an argument to another function and returns whether the value matches the expected type. For example, the ISERROR function returns TRUE if the value you reference contains an error.

ISBLANK: Returns TRUE if a value is blank, and FALSE otherwise.

Syntax: =ISBLANK(value)

Example: If cell A1 is blank, the formula =ISBLANK(A1) will return TRUE.

ISERROR: Returns TRUE if a value is any error value, and FALSE otherwise.

Syntax: =ISERROR(value)

Example: If cell A1 contains an error value, such as #VALUE! or #DIV/0!, the formula =ISERROR(A1) will return TRUE.

ISLOGICAL: Returns TRUE if a value is a logical value (TRUE or FALSE), and FALSE otherwise.

Syntax: =ISLOGICAL(value)

Example: If cell A1 contains the logical value TRUE, the formula =ISLOGICAL(A1) will return TRUE.

ISNONTEXT: Returns TRUE if a value is not text, and FALSE otherwise.

Syntax: =ISNONTEXT(value)

Example: If cell A1 contains a non-text value, such as a number or a date, the formula =ISNONTEXT(A1) will return TRUE.

ISNUMBER: Returns TRUE if a value is a number, and FALSE otherwise.

Syntax: =ISNUMBER(value)

Example: If cell A1 contains a numeric value, the formula =ISNUMBER(A1) will return TRUE.

ISTEXT: Returns TRUE if a value is text, and FALSE otherwise.

Syntax: =ISTEXT(value)

Example: If cell A1 contains a text value, the formula =ISTEXT(A1) will return TRUE.

Statistical Functions:

Statistical functions in Power BI are used to analyze a set of data and calculate various statistical measures to understand the underlying trends and patterns. Here are the explanations and examples of some commonly used statistical functions in Excel:

NORM.DIST: Returns the cumulative distribution function of a standard normal distribution.

Syntax: =NORM.DIST(x,mean,standard_dev,cumulative)

Example: If you have a data set with mean value 50 and standard deviation 5, and you want to find the probability of getting a value less than or equal to 55, you can use the NORM.DIST function as follows:

=NORM.DIST(55,50,5,TRUE)

Result: 0.841344746

NORM.INV: Returns the inverse of the cumulative distribution function of a standard normal distribution.

Syntax: =NORM.INV(probability,mean,standard_dev)

Example: If you have a data set with mean value 50 and standard deviation 5, and you want to find the value that corresponds to a probability of 0.8413, you can use the NORM.INV function as follows:

=NORM.INV(0.8413,50,5)

Result: 55.0084

NORM.S.DIST: Returns the cumulative distribution function of a normal distribution with a specified mean and standard deviation.

Syntax: =NORM.S.DIST(x,mean,standard_dev,cumulative)

Example: If you have a data set with mean value 50 and standard deviation 5, and you want to find the probability of getting a value less than or equal to 55, you can use the NORM.S.DIST function as follows:

=NORM.S.DIST(55,50,5,TRUE)

Result: 0.841344746

NORM.S.INV: Returns the inverse of the cumulative distribution function of a normal distribution with a specified mean and standard deviation.

Syntax: =NORM.S.INV(probability,mean,standard_dev)

Example: If you have a data set with mean value 50 and standard deviation 5, and you want to find the value that corresponds to a probability of 0.8413, you can use the NORM.S.INV function as follows:

=NORM.S.INV(0.8413,50,5)

Result: 55.0084

STDEV.P: Calculates the standard deviation of a population.

Syntax: =STDEV.P(number1,[number2],...)

Example: If you have a data set with values {1, 2, 3, 4, 5}, you can use the STDEV.P function to find the standard deviation of the population as follows:

=STDEV.P(1,2,3,4,5)

Result: 1.5811

STDEV.S: Calculates the standard deviation of a sample.

Syntax: =STDEV.S(number1,[number2],...)

Example: If you have a data set with values {1, 2, 3, 4, 5}, you can use the STDEV.S function to find the standard deviation of the sample as follows:

=STDEV.S(1,2,3,4,5)

Result: 1.8708

VAR.P: Calculates the variance of a population.

Syntax: =VAR.P(number1,[number2],...)

Example: If you have a data set with values {1, 2, 3, 4, 5}, you can use the VAR.P function to find the variance of the population as follows:

=VAR.P(1,2,3,4,5)

Result: 2.5

VAR.S: VAR.S is an Excel statistical function that calculates the variance of a sample. Variance is a measure of how spread out a dataset is. The variance is calculated as the average of the squared differences from the mean of the data points.

Syntax:=VAR.S(number1,[number2],...)

Arguments:
number1 (required): The first number or range of numbers in the sample.
number2 (optional): The second number or range of numbers in the sample. You can include up to 254 additional number arguments.

Example:
Suppose you have a sample of test scores for a class of 20 students. The scores are in cells A2:A21. You can use VAR.S to calculate the sample variance of the test scores as follows:
=VAR.S(A2:A21)
This will return the variance of the sample of test scores.

Filter Functions:

SALES TABLE			
Product	**Category**	**Amount**	**Year**
A	Electronics	1000	2020
B	Electronics	2000	2020
C	Clothing	1500	2021
D	Clothing	500	2021
E	Electronics	1200	2021

PRODUCT TABLE	
Product	**Category**
A	Electronics
B	Electronics
C	Clothing
D	Clothing
E	Electronics

ALL: Returns all the rows in a table, or all the values in a column, ignoring any filters that might have been applied.

Syntax: = ALL([<tableName> | <columnName> [, <columnName> [, ...]]])

Example: How can you create a measure that calculates the total sales, ignoring any filters applied to the Product column?

Total Sales (All Products) = CALCULATE(SUM(Sales[Amount]), ALL(Product))

Result: Total Sales (All Products) = 1000 + 2000 + 1500 + 500 + 1200 = 6200

The ALL function removes any filters applied to the Product column, allowing the calculation to include all rows in the Sales table.

ALLEXCEPT: Removes all filters in a table except the filters on the specified columns.

Syntax: = ALLEXCEPT(<tableName>, <columnName> [, <columnName> [, ...]])

Example: How can you calculate the total sales but keep the filter on the Product[Category] column?

Total Sales (By Category) = CALCULATE(SUM(Sales[Amount]), ALLEXCEPT(Product, Product[Category]))

Result:
For Electronics: 1000 + 2000 + 1200 = 4200
For Clothing: 1500 + 500 = 2000

The ALLEXCEPT function removes all filters except those on the Product[Category] column, retaining category-specific calculations.

ALLSELECTED: Removes filters from all columns except the filters that are applied directly to the specified columns.

Syntax: = ALLSELECTED([<tableName> | <columnName> [, <columnName> [, ...]]])

Example: How can you calculate the total sales for the selected products, excluding the filters applied by the visualizations?

Total Sales (Selected) = CALCULATE(SUM(Sales[Amount]), ALLSELECTED(Product))

Result: This function depends on the context of the visualizations in Power BI. If all products are selected, the total is 6200.

The ALLSELECTED function retains the filters directly applied by the user selection, ignoring the visualization filters.

HASONEVALUE: Returns TRUE if there is only one filter applied to the column.

Syntax: = HASONEFILTER(<column>)

Example: How can you check if there is only one filter applied on the Product column?

Has One Product = HASONEVALUE(Product[ProductName])

Result: Depends on the selection context (TRUE if one product, FALSE otherwise).

The HASONEVALUE function checks if only one unique value is in the context.

HASONEFILTER: Returns TRUE if there is only one filter applied to the column.

Syntax: = HASONEFILTER(<column>)

Example: How can you check if there is only one filter applied on the Product column?

Has One Product Filter = HASONEFILTER(Product[ProductName])

Result: Depends on the filter context (TRUE if one filter, FALSE otherwise).

The HASONEFILTER function returns TRUE if the column has exactly one filter applied.

CALCULATE: Modifies the context in which the data is evaluated.

Syntax: = CALCULATE(<expression>, <filter1>, <filter2>, ...)

Example: How can you calculate the total sales for the year 2020?

Total Sales (2020) = CALCULATE(SUM(Sales[Amount]), Sales[Year] = 2020)

Result: 3000

The CALCULATE function evaluates the sum of sales within the context of the year 2020 filter.

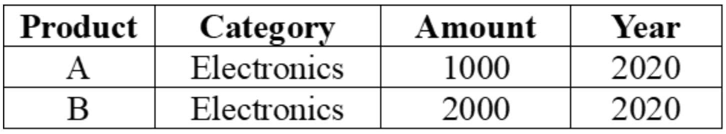
CALCULATETABLE: Evaluates a table expression in a modified filter context.

Syntax: = CALCULATETABLE(<table>, <filter1>, <filter2>, ...)

Example: How can you create a table containing only sales from 2020?

Sales 2020 Table = CALCULATETABLE(Sales, Sales[Year] = 2020)

Result:



Product	Category	Amount	Year
A	Electronics	1000	2020
B	Electronics	2000	2020

The CALCULATETABLE function creates a new table with rows where the year is 2020.

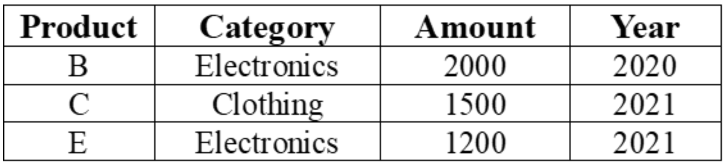
FILTER: Returns a table that represents a subset of another table or expression.

Syntax: = FILTER(<table>, <expression>)

Example: How can you create a table with only the rows where sales are greater than $1000?

High Sales = FILTER(Sales, Sales[Amount] > 1000)

Result:



Product	Category	Amount	Year
B	Electronics	2000	2020
C	Clothing	1500	2021
E	Electronics	1200	2021

The FILTER function returns a table with rows that meet the condition Sales[Amount] > 1000.

KEEPFILTERS: Modifies how filters are applied while retaining the initial filter context.

Syntax: = KEEPFILTERS(<filter>)

Example: How can you calculate the total sales, retaining filters applied on the Product[Category] column?

Total Sales (Keep Filters) = CALCULATE(SUM(Sales[Amount]),KEEPFILTERS(Product[Category] = "Electronics"))

Result: Total Sales (Electronics) = 1000 + 2000 + 1200 = 4200

The KEEPFILTERS function ensures that the original filters on Product[Category] are retained during calculation.

REMOVEFILTERS: Removes all filters from the specified columns or the entire table, effectively resetting the context for those columns.

Syntax: = REMOVEFILTERS([<tableName> | <columnName> [, <columnName> [, ...]]])

Example: How can you calculate the total sales for all products, ignoring any filters applied to the Product table?

Total Sales (Ignore Filters) = CALCULATE(SUM(Sales[Amount]), REMOVEFILTERS(Product))

Result: 6200

The REMOVEFILTERS function removes any filters applied to the Product table, allowing the calculation to include all rows in the Sales table, similar to the ALL function but more explicitly used for clearing filters.

Calculate Function:

The CALCULATE function in Power BI is a powerful and versatile function used to modify the context in which an expression or calculation is evaluated. It allows you to apply filters and manipulate the context of calculations, enabling more advanced analysis and calculations in your Power BI reports. Here's a detailed explanation of the CALCULATE function, including its syntax, examples, and use cases:

Or

The CALCULATE function in Power BI allows you to perform calculations on your data while considering specific filters or conditions. It helps you modify the context in which your calculations are done.

Think of it like applying a "special lens" to your data. You can use the CALCULATE function to zoom in on specific parts of your data by applying filters. For example, you can calculate the total sales for a particular product category or within a specific date range.

Syntax: CALCULATE(<expression>, <filter1>, <filter2>, ...)

The CALCULATE function takes an expression as its first argument, which can be a measure or a calculated column. The subsequent arguments are optional filters that define the context in which the expression is evaluated.

Examples:

Calculate Total Sales for a Specific Product Category:

```
CALCULATE(SUM(Sales[Amount]), Sales[Category] = "Electronics")
```

In this example, the CALCULATE function calculates the sum of the Sales[Amount] column but only considers rows where the Sales[Category] is "Electronics". The filter Sales[Category] = "Electronics" modifies the context of the calculation.

Calculate Sales for a Specific Date Range:

```
CALCULATE(SUM(Sales[Amount]), Sales[Date] >= DATE(2022, 1, 1), Sales[Date] <= DATE(2022, 12, 31))
```

This example calculates the sum of the Sales[Amount] column but only includes rows where the Sales[Date] falls within the specified date range. The filters Sales[Date] >= DATE(2022, 1, 1) and Sales[Date] <= DATE(2022, 12, 31) define the context for the calculation.

Calculate Sales Excluding a Specific Region:

CALCULATE(SUM(Sales[Amount]), NOT Sales[Region] = "North")

Here, the CALCULATE function calculates the sum of the Sales[Amount] column but excludes rows where the Sales[Region] is "North". The filter NOT Sales[Region] = "North" modifies the context of the calculation.

Use Cases and When to Use:

The CALCULATE function is especially useful in the following scenarios:

Applying Filters: You can use CALCULATE to apply specific filters to your calculations, such as filtering data based on a specific category, date range, or any other condition.

Modifying Context: CALCULATE allows you to change the context in which an expression is evaluated. It enables you to override the default row context and apply filters to specific calculations.

Creating Complex Measures: You can use CALCULATE to combine multiple measures or perform calculations that involve multiple levels of filtering or manipulation.

How to Use:

To use the CALCULATE function, follow these steps:

Specify the expression you want to calculate as the first argument of the CALCULATE function.
Add additional arguments to the CALCULATE function to define filters or conditions that modify the context of the calculation.

Use comparison operators (=, <>, >, <, >=, <=) to specify filter conditions.

Combine multiple filters using logical operators (AND, OR, NOT) to create complex conditions.
By using the CALCULATE function effectively, you can perform advanced calculations, apply dynamic filters, and analyze your data from different perspectives in Power BI.

Complex Examples That Demonstrate The Versatility Of The CALCULATE Function:

Example 1: Calculate Sales Amount for a Specific Product Category and Date Range, Excluding Returns

```
CALCULATE( SUM(Sales[Amount]), Sales[Category] = "Electronics", Sales[Date] >= DATE(2022, 1, 1), Sales[Date] <= DATE(2022, 12, 31), NOT Sales[Type] = "Return")
```

In this example, the CALCULATE function calculates the sum of the Sales[Amount] column but only includes rows where the Sales[Category] is "Electronics", the Sales[Date] falls within the specified date range, and the Sales[Type] is not "Return". This complex filter combination allows you to analyze the total sales amount for electronics products within a specific date range, excluding any returns.

Example 2: Calculate Average Sales per Customer, Considering Only High-Value Customers

```
CALCULATE( AVERAGE(Sales[Amount]), FILTER(Customers,
CALCULATE(SUM(Sales[Amount]),Sales[CustomerID] = Customers[CustomerID]) > 10000))
```

In this example, the CALCULATE function calculates the average of the Sales[Amount] column but only considers rows where the associated customer's total sales amount is greater than $10,000. The FILTER function is used to filter the Customers table based on the total sales amount calculated within the CALCULATE function. This allows you to analyze the average sales per customer, focusing only on high-value customers.

Example 3: Calculate Total Sales Amount, Ignoring Filters on Product Color

```
CALCULATE( SUM(Sales[Amount]), ALL(Sales[Color]))
```

In this example, the CALCULATE function calculates the sum of the Sales[Amount] column, ignoring any filters applied to the Sales[Color] column. The ALL function removes the filter context from the Sales[Color] column, ensuring that all colors are included in the calculation. This is useful when you want to analyze the total sales amount without considering the filter on a specific attribute.

These examples showcase the flexibility of the CALCULATE function in applying complex filters and modifying the context of calculations in Power BI. By combining different filters and expressions within the CALCULATE function, you can perform advanced analysis and gain deeper insights from your data.

When to use New Measure & Quick Measure?

In Power BI, a measure is a calculation performed on a dataset. A measure is used to aggregate or summarize the data in a table or a visualization. There are two ways to create measures in Power BI: New Measure and Quick Measure.

New Measure allows you to create a custom calculation using DAX **Syntax**. This is useful when you need to create a more complex calculation that is not available as a pre-built function or when you want to create a measure that can be reused across multiple visuals.

Quick Measure is a pre-built function that allows you to quickly create common calculations, such as sum, count, average, min, and max. Quick Measure uses a point-and-click interface, making it easy to create simple calculations without writing any DAX code.

Both New Measure and Quick Measure can be used to create measures in Power BI. The choice between the two depends on the complexity of the calculation you want to perform. If the calculation is simple and can be performed using a pre-built function, Quick Measure is a good choice. If the calculation is more complex or requires custom logic, New Measure is the way to go.

Difference Between New Measure & Quick Measure

In Power BI, a measure is a calculation based on the data in your model. Measures allow you to perform calculations and aggregations on your data to get more meaningful insights. There are two ways to create measures in Power BI: new measure and quick measure.

New measure: This is the traditional way of creating a measure in Power BI. You can create a new measure by going to the "Modeling" tab and clicking on "New Measure". This will open the formula bar where you can write your own DAX formula to create the measure.

Quick measure: Quick measures are pre-built DAX formulas that can be easily created using the "Quick measures" feature. This feature allows you to select the type of calculation you want to perform and the fields you want to use, and it will generate the DAX formula for you automatically.

The main difference between new measure and quick measure is the level of control you have over the calculation. With a new measure, you have complete control over the DAX formula and can customize it to meet your specific needs. With a quick measure, you are limited to the pre-built formulas that are available, but it is a faster and easier way to create common calculations. Additionally, quick measures are designed to be dynamic and adjust to changes in your data, while new measures require manual updates to the DAX formula if your data changes.

For a comprehensive understanding of **DAX functions**, we encourage you to explore the official Microsoft Learn resource available at Microsoft Learn: DAX Function Reference. This guide provides in-depth details and examples to help you master DAX functions effectively.

If you would like to learn more &
stay updated, please follow me on
LinkedIn