

DAX Queries Cheatsheet with Syntax

Introduction

DAX (Data Analysis Expressions) is a formula language used in Power BI, Analysis Services, and Power Pivot. This comprehensive cheatsheet covers essential syntax, functions, and common patterns for data analysis.

1. Basic Syntax and Structure

Formula Structure

= FunctionName(Parameter1, Parameter2, ...)

Common Operators

Operator	Description	Example
+	Addition	Sales[Amount] + Tax[Amount]
-	Subtraction	Total - Discount
*	Multiplication	Price * Quantity
/	Division	Total / Count
^	Exponentiation	Value ^ 2
=	Equals (comparison)	Region = "West"
<>	Not equal	Status <> "Inactive"
<	Less than	Age < 18
>	Greater than	Sales > 1000
<=	Less or equal	Age <= 65
>=	Greater or equal	Score >= 90
&&	AND logical operator	Region = "West" && Year = 2024
\\	OR logical operator	Status = "Active" \\ Status = "Pending"
IN {}	OR for multiple values	Category IN {"Electronics", "Furniture"}

2. Aggregation Functions

Aggregation functions return a single scalar value by aggregating data from a column or table.

Basic Aggregations

Function	Syntax	Description	Example
SUM	SUM(column)	Sum of all values	SUM(Sales[Amount])
AVERAGE	AVERAGE(column)	Average of values	AVERAGE(Products[Price])
COUNT	COUNT(column)	Count of non-empty cells	COUNT(Orders[OrderID])
COUNTA	COUNTA(column)	Count of non-blank cells	COUNTA(Customers[Name])
COUNTBLANK	COUNTBLANK(column)	Count of blank cells	COUNTBLANK(Data[Value])
COUNTROWS	COUNTROWS(table)	Count rows in a table	COUNTROWS(Sales)
DISTINCTCOUNT	DISTINCTCOUNT(column)	Count unique values	DISTINCTCOUNT(Customers[CustomerID])
MIN	MIN(column)	Minimum value	MIN(Products[Price])
MAX	MAX(column)	Maximum value	MAX(Sales[Amount])
MAXX	MAXX(table, expression)	Max across iteration	MAXX(Sales, Sales[Amount] * Sales[Quantity])
MINX	MINX(table, expression)	Min across iteration	MINX(Products, Products[Price])

Iterator Functions (X Functions)

Iterator functions evaluate an expression for each row in a table and aggregate results.

Function	Syntax	Description	Example
SUMX	SUMX(table, expression)	Sum of expression across rows	SUMX(Sales, Sales[Qty] * Sales[Price])
AVERAGEX	AVERAGEX(table, expression)	Average of expression	AVERAGEX(Orders, Orders[Total])
COUNTX	COUNTX(table, expression)	Count non-blank results	COUNTX(Sales, Sales[Amount])
PRODUCTX	PRODUCTX(table, expression)	Product of expression values	PRODUCTX(Table, Table[Value])

3. Filtering Functions

Filtering functions manipulate the data context to create dynamic calculations.

CALCULATE (Most Important)

Syntax:

CALCULATE(expression, filter1, filter2, ...)

Description: Evaluates an expression in a modified filter context.

Examples:

-- Sum for specific region

```
CALCULATE(SUM(Sales[Amount]), Sales[Region] = "West")
```

-- Multiple filters

```
CALCULATE(
SUM(Sales[Amount]),
Sales[Region] = "North",
Sales[Year] = 2024
)
```

-- Using column references

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

ALL Functions

Function	Syntax	Description	Example
ALL	ALL(table or column)	Remove all filters	CALCULATE(SUM(Sales[Amount]), ALL(Sales))
ALLEXCEPT	ALLEXCEPT(table, col1, col2, ...)	Remove filters except specific	CALCULATE(SUM(Sales[Amount]), ALLEXCEPT(Sales, Sales[Region]))
ALLSELECTED	ALLSELECTED(column)	Clear filters except selections	CALCULATE(SUM(Sales[Amount]), ALLSELECTED(Sales[Region]))

FILTER Functions

Function	Syntax	Description	Example
FILTER	FILTER(table, condition)	Filter table by condition	FILTER(Products, Products[Price] > 100)
VALUES	VALUES(column)	Unique values in column	VALUES(Customers[City])
DISTINCT	DISTINCT(column)	Distinct values (like VALUES)	DISTINCT(Products[Category])

4. Text Functions

Text functions manipulate and analyze string values.

Function	Syntax	Description	Example
CONCATENATE	CONCATENATE(text1, text2, ...)	Combine strings	CONCATENATE(First[Name], " ", Last[Name])
&	text1 & text2	String concatenation operator	First[Name] & " " & Last[Name]
UPPER	UPPER(text)	Convert to uppercase	UPPER(Customers[Name])
LOWER	LOWER(text)	Convert to lowercase	LOWER(Products[Category])
TRIM	TRIM(text)	Remove leading/trailing spaces	TRIM(Data[Value])
LEN	LEN(text)	Length of text	LEN(Description[Text])
LEFT	LEFT(text, num_chars)	First N characters	LEFT(Code[Value], 3)
RIGHT	RIGHT(text, num_chars)	Last N characters	RIGHT(Code[Value], 2)
MID	MID(text, start, num_chars)	Extract substring	MID(Text[Value], 1, 10)
SEARCH	SEARCH(find_text, text, [position])	Find text position	SEARCH("Sales", [Text])
FIND	FIND(find_text, text, [position])	Find (case-sensitive)	FIND("John", Customers[Name])
SUBSTITUTE	SUBSTITUTE(text, old, new)	Replace text	SUBSTITUTE(Text[Value], "Old", "New")
VALUE	VALUE(text)	Convert text to number	VALUE("123.45")

Function	Syntax	Description	Example
TEXT	TEXT(value, format)	Format value as text	TEXT(Date[Today], "YYYY-MM-DD")

5. Date and Time Functions

Date functions help create time-based calculations and comparisons.

Basic Date Functions

Function	Syntax	Description	Example
TODAY	TODAY()	Current date	TODAY()
NOW	NOW()	Current date and time	NOW()
DATE	DATE(year, month, day)	Create date	DATE(2024, 12, 25)
YEAR	YEAR(date)	Extract year	YEAR(Orders[OrderDate])
MONTH	MONTH(date)	Extract month	MONTH(Orders[OrderDate])
DAY	DAY(date)	Extract day	DAY(Orders[OrderDate])
QUARTER	QUARTER(date)	Extract quarter (1-4)	QUARTER(Orders[OrderDate])
WEEKDAY	WEEKDAY(date, [mode])	Day of week (1-7)	WEEKDAY(Orders[OrderDate])
WEEKNUM	WEEKNUM(date, [mode])	Week number (1-53)	WEEKNUM(Orders[OrderDate])
EDATE	EDATE(date, months)	Date N months away	EDATE(StartDate, 6)
EOMONTH	EOMONTH(date, months)	End of month	EOMONTH(TODAY(), 0)
DATEDIFF	DATEDIFF(date1, date2, interval)	Difference between dates	DATEDIFF(StartDate, EndDate, DAY)

Time Intelligence Functions

Time intelligence functions enable year-to-date, month-to-date, and other time-based calculations.

Function	Syntax	Description	Example
TOTALYTD	TOTALYTD(expression, date_column, [filter])	Year-to-date total	TOTALYTD(SUM(Sales[Amount]), Dates[Date])
TOTALMTD	TOTALMTD(expression, date_column, [filter])	Month-to-date total	TOTALMTD(SUM(Sales[Amount]), Dates[Date])
TOTALQTD	TOTALQTD(expression, date_column, [filter])	Quarter-to-date total	TOTALQTD(SUM(Sales[Amount]), Dates[Date])
SAMEPERIODLASTYEAR	SAMEPERIODLASTYEAR(dates)	Same period previous year	CALCULATE(SUM(Sales[Amount]), SAMEPERIODLASTYEAR(Dates[Date]))
PARALLELPERIOD	PARALLELPERIOD(column, intervals, [period])	Different time period	PARALLELPERIOD(Dates[Date], -1, YEAR)
DATESYTD	DATESYTD(date_column, [last_date])	All dates year-to-date	CALCULATE(SUM(Sales[Amount]), DATESYTD(Dates[Date]))
DATESMTD	DATESMTD(date_column)	All dates month-to-date	DATESMTD(Dates[Date])

6. Logical Functions

Logical functions evaluate conditions and return TRUE or FALSE.

Func tion	Syntax	Descripti on	Example
IF	IF(condition, value_if_true, value_if_false)	Condition al statement	IF(Sales[Amount] > 1000, "High", "Low")
SWI TCH	SWITCH(expression , value1, result1, [default])	Multiple condition s	SWITCH(Status[Type], "A", "Active", "I", "Inactive", "Other")
AND	AND(logical1, logical2, ...)	All condition s true	AND(Age > 18, Status = "Active")
OR	OR(logical1, logical2, ...)	Any condition true	OR(Region = "North", Region = "South")
NOT	NOT(logical)	Negate condition	NOT(Status = "Inactive")
TRU E	TRUE()	Boolean TRUE	TRUE()
FALS E	FALSE()	Boolean FALSE	FALSE()
IFER ROR	IFERROR(value, value_if_error)	Handle errors	IFERROR(SUM(Sales[Am ount]), 0)

7. Information Functions

Information functions check data types and properties.

Function	Syntax	Description	Example
ISBLANK	ISBLANK(value)	Check if blank	ISBLANK(Data[Value])
ISTEXT	ISTEXT(value)	Check if text	ISTEXT(Column[Value])
ISNUMBER	ISNUMBER(value)	Check if number	ISNUMBER(Amount[Value])
ISERROR	ISERROR(value)	Check if error	ISERROR(Formula)
HASONEVALUE	HASONEVALUE(column)	Check if single value selected	HASONEVALUE(Region[Region])

8. Common Advanced Patterns

Year-to-Date (YTD) Calculation

Sales YTD = TOTALYTD(SUM(Sales[Amount]), Dates[Date])

Month-over-Month Growth

MoM Growth =

VAR CurrentMonth = CALCULATE(SUM(Sales[Amount]), DATESMTD(Dates[Date]))

VAR PreviousMonth = CALCULATE(
SUM(Sales[Amount]),
DATEADD(DATESMTD(Dates[Date]), -1, MONTH)
)

RETURN DIVIDE(CurrentMonth - PreviousMonth, PreviousMonth)

Running Total

Running Total =

CALCULATE(
SUM(Sales[Amount]),
FILTER(ALL(Dates[Date]), Dates[Date] <= MAX(Dates[Date]))
)

Percentage of Total

% of Total =

DIVIDE(
SUM(Sales[Amount]),
CALCULATE(SUM(Sales[Amount]), ALL(Sales[Region]))
)

Conditional Sum

High Value Sales =
SUMIF(Sales[Amount], ">1000")

Top 10 Products

Top Products =
CALCULATE(
SUM(Sales[Amount]),
FILTER(Products, RANK(Products[Revenue], DESCENDING) <= 10)
)

9. Quick Reference by Use Case

Sales Analysis

- **Total Sales:** SUM(Sales[Amount])
- **Average Order Value:** AVERAGE(Orders[Total])
- **Unique Customers:** DISTINCTCOUNT(Customers[CustomerID])
- **Sales by Region:** CALCULATE(SUM(Sales[Amount]), Sales[Region] = "North")

Performance Metrics

- **Growth Rate:** DIVIDE(Current - Previous, Previous)
- **Variance:** Actual - Budget
- **Index:** DIVIDE(Value, Benchmark) * 100
- **Rank:** RANK(Sales, DESCENDING)

Time Analysis

- **Current Year Sales:** TOTALYTD(SUM(Sales[Amount]), Dates[Date])
 - **Previous Year Comparison:** SAMEPERIODLASTYEAR(Dates[Date])
 - **Days since Date:** DATEDIFF(StartDate, TODAY(), DAY)
-

10. Best Practices

✓ DO:

- Use CALCULATE for context filtering
- Leverage X functions for row-by-row calculations
- Create measures instead of calculated columns when possible
- Use variables (VAR) for complex formulas
- Name measures descriptively

✗ DON'T:

- Use nested IF statements excessively (use SWITCH instead)
- Filter entire tables in CALCULATE without specific columns
- Avoid circular dependencies
- Don't hardcode values; use parameters
- Avoid heavy calculations in row context

References

- [1] Microsoft Learn. (2025). DAX function reference. <https://learn.microsoft.com/en-us/dax/dax-function-reference>
- [2] DAX Guide. (2025). DAX Functions Reference. <https://dax.guide>
- [3] Microsoft Learn. (2025). Data Analysis Expressions (DAX) Reference. <https://learn.microsoft.com/en-us/dax/>