**DAX**

**Here is a list of some of the most commonly used DAX functions in Power BI, along with their explanation and examples:**

1. SUM: Calculates the sum of a column of numbers.

Syntax: SUM(column\_name)

Example: SUM(Amount)

1. COUNT: Counts the number of rows in a table or column.

Syntax: COUNT(table\_name) or COUNT(column\_name)

Example: COUNT(Sales) or COUNT(Product)

1. AVERAGE: Calculates the average of a column of numbers.

Syntax: AVERAGE(column\_name)

Example: AVERAGE(Amount)

1. MIN: Returns the minimum value in a column.

Syntax: MIN(column\_name)

Example: MIN(Amount)

1. MAX: Returns the maximum value in a column.

Syntax: MAX(column\_name)

Example: MAX(Amount)

1. IF: Evaluates an expression and returns one of two values based on the result.

Syntax: IF(condition, value\_if\_true, value\_if\_false)

Example: IF(Amount > 1000, "High", "Low")

1. VLOOKUP: Searches for a value in the first column of a table and returns the corresponding value from another column in the same row.

Syntax: VLOOKUP(lookup\_value, table\_name, column\_index, [exact\_match])

Example: VLOOKUP(Product, Sales, 2, TRUE)

1. SUMX: Calculates the sum of an expression for each row in a table based on a specific filter condition.

Syntax: SUMX(table, expression)

Example: SUMX(Sales, Amount \* (Product = "Bicycle"))

1. CALCULATE: Modifies the filter context of a measure or table by applying one or more filters.

Syntax: CALCULATE(expression, filter1, [filter2, ...])

Example: CALCULATE(SUM(Amount), Sales[Product] = "Bicycle")

1. ADDCOLUMNS: Adds a new column to a table, with the values in the new column calculated based on a DAX expression.

Syntax: ADDCOLUMNS(table, "column\_name", expression)

Example: ADDCOLUMNS(Sales, "Sales Tax", Amount \* 0.1)

These are just a few examples of the many DAX functions available in Power BI. By using these functions, you can perform a wide range of calculations and data manipulations, and gain insights into your data that can inform your business decisions.

**Here is a list of some of the most complex DAX functions in Power BI, along with their explanation and examples:**

1. RANKX: Returns the rank of a value within a column, based on a specified order.

Syntax: RANKX(table, expression, [value], [order])

Example: RANKX(Sales, SUM(Amount),, DESC)

1. RANK.EQ: Returns the rank of a value within a column, based on a specified order, and breaking ties with the same rank.

Syntax: RANK.EQ(column, [value], [order])

Example: RANK.EQ(Amount, 5000, 0)

1. PERCENTILE.EXC: Returns the nth percentile of a column of numbers, excluding zeros.

Syntax: PERCENTILE.EXC(column, k)

Example: PERCENTILE.EXC(Amount, 0.9)

1. WINDOW\_MAX: Returns the maximum value in a specified range of rows.

Syntax: WINDOW\_MAX(column, [offset], [default])

Example: WINDOW\_MAX(Amount, 0, BLANK())

1. WINDOW\_MIN: Returns the minimum value in a specified range of rows.

Syntax: WINDOW\_MIN(column, [offset], [default])

Example: WINDOW\_MIN(Amount, 0, BLANK())

1. RANK.AVG: Returns the rank of a value within a column, based on a specified order, breaking ties by averaging the ranks.

Syntax: RANK.AVG(column, [value], [order])

Example: RANK.AVG(Amount, 5000, 0)

1. DIVIDE: Returns the result of dividing one number by another.

Syntax: DIVIDE(numerator, denominator, [default\_result])

Example: DIVIDE(SUM(Amount), COUNT(Product), 0)

1. TOPN: Returns the top N items from a column based on a specified order.

Syntax: TOPN(count, table, expression, [order])

Example: TOPN(5, Sales, SUM(Amount), DESC)

1. RANK.DENSE: Returns the dense rank of a value within a column, based on a specified order.

Syntax: RANK.DENSE(column, [value], [order])

Example: RANK.DENSE(Amount, 5000, 0)

1. EARLIER: Returns the value of an expression evaluated at an earlier row in the same context.

Syntax: EARLIER(expression)

Example: EARLIER(Amount) + SUM(Amount)

These are some of the more complex DAX functions in Power BI, and they can be used to perform advanced data manipulations and calculations. To use these functions effectively, it is important to have a good understanding of DAX syntax and concepts, as well as a solid grasp of the data being analyzed.