Introduction

Statistical functions are integral to understanding and interpreting an organization's data landscape. They provide meaningful insights into datasets that help businesses to make informed decisions. Statistical functions in Microsoft's Power BI are used to create compelling visualizations, perform real-time analytics, and drive business intelligence.

In this reading, you'll explore six important statistical functions that you'll make frequent use of in DAX expressions. These are **AVERAGE**, **MEDIAN**, **COUNT**, **DISTINCTCOUNT**, **MIN**, and **MAX**. You'll also learn how they are used through practical examples.

The AVERAGE function

The AVERAGE function, also known as the mean, sums up all the numbers in a dataset and divides the result by the total count of numbers. This function is frequently used to identify a central tendency in a dataset. It is beneficial when you need to find the middle ground or commonality within data.

Example: You want to analyze the average quantities of a product sold over a specified period to determine the typical sales volume for Adventure Works.

```
Avg. Of Quantities Sold = AVERAGE (Sales [Quantity])
```

In this example, **Sales** is the table name, and **Quantity** is the column name that contains the numbers for which you want the average.

Related Functions:

There are several functions related to AVERAGE.

AVERAGEX

This function calculates the average (arithmetic mean) of a set of expressions evaluated over a table.

```
= AVERAGEX (Sales, Sales [Quantity] * Sales [Price])
```

In the above example, the **AVERAGEX** function is applied to the **Sales** table. The expression within the function multiplies the quantity sold by the price for each row in the **Sales** table, giving the sales amount for each product. Then the function calculates the average of these sales amounts.

AVERAGEA

This function returns the values' average (arithmetic mean) in a column. It handles text and non-numeric values. Whenever there are no rows to aggregate, the function returns a blank.

```
= AVERAGEA([Amount])
```

The above example returns the average of non-blank cells in the referenced column.

The MEDIAN function

The MEDIAN function calculates the middle value in a set of numbers. It sorts the numbers in ascending order and then selects the middle number. The median is the average of the two middle numbers for datasets with an even number of observations.

Unlike the average, the median is less affected by outliers and extreme values. This means that it's useful for datasets with skewed distributions. Only numeric data types are supported in this function. Dates, logical values, and text columns are not supported.

Example: You are analyzing customer data at Adventure Works. You can easily analyze customers' locations from the dimension attributes. However, you also need to identify the median age of customers. You can perform this calculation by using the DAX median function as follows:

```
Median Customer Age = MEDIAN(Customers [Age])
```

Customers is the table name, and the **Age** column contains the numbers you require for the median **customer age** column.

The COUNT function

The COUNT function counts the number of rows in a column or a table. It is often used to measure the size of a dataset. You can use it to count all or only rows meeting specific criteria. The only argument in the function is the column. When the function finds no rows to count, it returns a blank.

Example: You need to identify the total number of customers in a particular country or city. In this case, you can use the **COUNT** function.

```
Number of Customers = COUNT([CustomerID])
```

CustomerID is the column name that contains the values to be counted.

Related functions:

The **COUNT** function has several related functions.

COUNTA

The **COUNTA** functions count the number of cells in a column that are not empty. It counts rows containing numeric and non-blank values, including text, dates, and logical values.

```
= COUNTA ('Reseller' [Phone])
```

The above function returns all rows in the reseller table with any value in the column that stores phone numbers.

COUNTAX

The **COUNTAX** function counts non-blank results when evaluating the result of an expression over a table. It works just like the **COUNTA** function but is used to iterate through the rows in a table and count rows where the specified expressions result in a non-blank result.

```
COUNTAX(<table>, <expression>)= COUNTAX (FILTER (Reseller, [Status] = "Active"), [Phone])
```

In the above example, the function counts the number of non-blank rows in the **phone** column while using the table that results from filtering the reseller table on Status = active.

COUNTBLANK

This function counts the number of blank cells in a column. If no rows are found that meet the condition, blanks are returned.

```
= COUNTBLANK (Reseller [BankName])
```

The above function returns the blank values for the **BankName** column in the **Reseller** table.

COUNTROWS

The **COUNTROWS** function counts the number of rows in the specified table or a table defined by an expression.

```
= COUNTROWS(['Orders'])
```

The above example shows how to count the number of rows in the **Orders** table, while the following example counts rows from the **resellersales** table.

```
= COUNTROWS(RELATEDTABLE(ResellerSales))
```

The DISTINCTCOUNT function

The **DISTINCTCOUNT** function counts the number of distinct values in a dataset. This function is helpful when you need to understand the count of unique values or categories.

The only argument allowed for this function is a column. You can use columns containing any type of data. When the function finds no rows to count, it returns a **BLANK**. Otherwise, it returns the count of distinct values.

Example: You have a sales dataset that records **customerID**, **products**, and **sales amount**. The **COUNT** function gives you the total counts, but **DISTINCTCOUNT** provides information about the customers' distinct counts and associated purchases.

```
Number of distinct Customers = DISTINCTCOUNT([CustomerID])
```

In this code, **CustomerID** is the column name that contains values to be counted.

The MIN function

The MIN function is used to identify the smallest values in a column or between two scalar expressions. These values provide an overview of the range of your data.

Example: Adventure Works records data on resellers. The company's resellers operate on specific margin values. You must calculate and generate insights on the reseller margin values as a calculated column. You can use the **MIN** function to generate data on the minimum reseller margin within your data.

```
Minimum Reseller Margin =MIN([Reseller Margin])
```

In this example, **Reseller Margin** is the column name that contains values to be evaluated.

Related functions:

There are several other functions related to the **MIN** function:

MINA

This function returns the smallest value in a column. It does not ignore logical values and text. When **MINA** operates with Boolean data types, it considers **TRUE** as **1** and **FALSE** as **0**.

```
= MINA (Sales [Freight])
```

The above expression returns the minimum freight charge from the **Sales** table.

MINX

This function returns the smallest value that results from evaluating an expression for each table row. The function takes a table or an expression that returns a table as its first argument. The second argument contains the expression evaluated for each row of the table.

```
= MINX (FILTER (Sales, [SalesTerritoryKey] = 5), [Freight])
```

The above example filters the **Sales** table and returns only rows for a specific sales territory. The formula then finds the minimum value in the **Freight** column.

The MAX function

The **MAX** function is used to identify the largest value in a column or the larger value between two scalar expressions. The **MIN** and **MAX** functions can provide an overview of the range of your data.

Example: The Adventure Works **Sales** dataset contains a **Sales Amount** column. This column represents sales for different transactions. You can use the **MAX** function to find the maximum sales amounts.

```
Max Sales Amount = MAX (Sales [Sales Amount])
```

Sales is the name of the table, and **Sales Amount** is the column name that contains the values to be evaluated.

Related functions:

There are several other functions related to the **MAX** function:

MAXA

This function returns the largest value in a column and does not ignore logical values and text. Therefore, this function can also be used in the date column.

```
= MAXA([Margin]
```

The following example returns the greatest value from a calculated column named **Margin**.

MAXX

This function evaluates an expression for each table row and returns the largest value. The table argument to the **MAXX** function can be a table name or an expression that evaluates a table. The second argument indicates the expression to be evaluated for each table row.

```
1    = MAXX (Sales, Sales [TaxAmt]+ Sales [Freight])
```

The following formula uses an expression as the second argument to calculate the total taxes and shipping amount for each order in the **Sales** table.