**Additional information: M Code and DAX**

**M Code**

Power Query Formula Language, commonly referred to as "M," is a functional, case-sensitive language used in Power Query for data transformation and manipulation. It's designed for creating complex data mashups and transformation tasks in tools like Microsoft Excel and Power BI.

**Key Features of M Language**

**Functional Language**: M is functional rather than procedural, meaning it focuses on defining functions and applying them to data rather than writing step-by-step instructions.

**Case-Sensitive**: The language distinguishes between uppercase and lowercase letters. For example, **Table** and **table** would be different entities.

**Syntax**: M uses a combination of keywords, symbols, and functions to define transformations. The syntax can be quite different from other programming languages but is designed to be intuitive for data manipulation tasks.

**Basic Syntax and Structure**

***Let Expression***

The **let** expression is commonly used in M to define variables and expressions.

let

    Source = Excel.CurrentWorkbook(){[Name="Table1"]}[Content],

    FilteredRows = Table.SelectRows(Source, each [Column1] > 5),

    Result = Table.Sort(FilteredRows, {"Column2", Order.Ascending})

in

    Result

**In this example:**

**Source**  loads data from an Excel table named "Table1."

**FilteredRows**  filters rows where the value in "Column1" is greater than 5.

**Result**  sorts the filtered rows by "Column2" in ascending order.

The final output of the **let** block is specified after **in**.

**M Code Functions**

<https://learn.microsoft.com/en-us/powerquery-m/power-query-m-function-reference>

**Common Functions**

**Table Functions:** Table.SelectRows, Table.Sort, Table.AddColumn

**List Functions:** List.Select, List.Sum, List.Transform

**Text Functions:** Text.Upper, Text.Lower, Text.Replace

**Date Functions:** Date.From, DateTime.LocalNow, Date.AddDays

**Query M Conclusion**

Power Query M language is a powerful tool for data transformation and is integral to Power BI and Excel's data processing capabilities. Understanding its syntax and functional approach can significantly enhance your ability to clean, transform, and analyze data efficiently.

**DAX code**

DAX (Data Analysis Expressions) is a formula language used in Microsoft Power BI, Power Pivot for Excel, and Analysis Services Tabular models. It's designed for data modeling and analysis, allowing users to create custom calculations and queries.

**Key Concepts of DAX**

**Calculated Columns**

Created in the data model

Computed for each row in a table

Stored in the model, increasing its size

Syntax: NewColumnName = Formula

**Measures**

Created in the data model

Calculated on-the-fly when used in a visual

Don't increase model size

Syntax: MeasureName = Formula

**Functions**

DAX  function types:

Aggregation functions (SUM, AVERAGE, MIN, MAX)

Time intelligence functions (DATEADD, TOTALYTD)

Filter functions (FILTER, ALL, RELATED)

Text functions (CONCATENATE, LEFT, RIGHT)

Logical functions (IF, AND, OR)

**Context**

Row context: For calculated columns

Filter context: For measures affected by report filters and slicers

**DAX Syntax Elements**

Table references: 'Table Name'

Column references: 'Table Name'[Column Name]

Measure references: [Measure Name]

Operators: +, -, \*, /, ==, <>, >=, <=, &&, ||

**Examples of DAX Code**

**Calculated Column:**

**Total Price = 'Sales'[Quantity] \* 'Sales'[Unit Price]**

**Simple Measure:**

**Total Sales = SUM('Sales'[Total Price])**

**More Complex Measure:**

**YoY Growth % =**

**VAR CurrentYearSales = [Total Sales]**

**VAR PreviousYearSales = CALCULATE([Total Sales], DATEADD('Date'[Date], -1, YEAR))**

**RETURN**

**DIVIDE(CurrentYearSales - PreviousYearSales, PreviousYearSales, 0)**

**Conclusion**

DAX is a formula language used within Power BI and other Microsoft platforms to perform complex data analysis directly within the data model, allowing for dynamic calculations and visualizations through a combination of functions, operators, and context-aware operations, enabling users to create sophisticated measures and insights without needing extensive external coding.