Lesson 8

Topic: Introduction to DAX Basics & Calculated Columns vs. Measures
Prerequisites: Download DAX_Practice_Data.xlsx file

1. What does DAX stand for?
    - DAX stands for Data Analysis Expressions and is the formula language used in Power BI, including the Power BI Service, for creating custom calculations, defining logic, and enriching data models beyond what basic Power BI imports can achieve.
2. Write a DAX formula to sum the Sales column.

    ```
    SalesSum = SUM(ForDax[Sales])
    ```
3. What is the difference between a calculated column and a measure?
    - The main difference is that calculated columns perform row-by-row calculations and are stored in memory, consuming RAM, while measures perform dynamic aggregations that are calculated on-the-fly based on the report's filter context, consuming CPU instead of memory
    - **When to Use**:
    - **calculated column:** Use for row-level operations like adding two existing columns, creating conditional values based on each row, or performing calculations that don't depend on filters applied in the report.
    - **Measure:** Best for aggregations (SUM, AVERAGE, COUNT), up-to-the-minute calculations, creating metrics, or generating key performance indicators (KPIs).
4. Use the DIVIDE function to calculate Profit Margin (Profit/Sales).

    ```
    Profit Margin = DIVIDE(
        [Total Profit],
        [Total Sales],0
    )
    ```
5. What does COUNTROWS() do in DAX?
    - The DAX function COUNTROWS() in Power BI counts the number of rows in a specified table. This table can be a base table within your data model or a table generated by an expression.
6. Create a measure: Total Profit that subtracts total cost from total sales

```
Total Profit = SUM(ForDax[Sales])-SUM(ForDax[Cost])
```

7. Write a measure to calculate Average Sales per Product.

```
Average Product Sale = DIVIDE(
    [Total Sales],
    DISTINCTCOUNT(ForDax[ProductID]),
    0
)
```

8. Use IF() to tag products as "High Profit" if Profit > 1000.

```
= Table.AddColumn(#"Added Custom", "Profit Status", each if
[Sales] > 1000 then "High Profit" else null)
```

9. What is a circular dependency error in a calculated column?
    - A circular dependency error in a Power BI calculated column occurs when two or more columns reference each other in a way that creates an infinite loop, making it impossible for the DAX engine to calculate their values. Common causes include directly referencing each other in DAX formulas, such as Column A using Column B and Column B using Column A, or the use of CALCULATE in a calculated column that indirectly causes dependencies between multiple columns in the same table.
10. Explain row context vs. filter context.
    - Row context evaluates expressions row by row within a table, typically seen in calculated columns and iterator functions like SUMX, while filter context applies a set of filters (from slicers, visuals, or relationships) to the entire data model, determining the subset of data for a measure's calculation. Row context provides the current row's values for calculations, whereas filter context determines which rows are even considered by the calculation.
11. Write a measure to calculate YTD Sales using TOTALYTD().

```
YTD Sales = TOTALYTD(
    [Total Sales],
    ForDax[Date]
)
```

12. Create a dynamic measure that switches between Sales, Profit, and Margin.

```
ToSwitch = {
```

```
    ("Total Sales", NAMEOF('ForDax'[Total Sales]), 0),
    ("Total Profit", NAMEOF('ForDax'[Total Profit]), 1),
    ("Profit Margin", NAMEOF('ForDax'[Profit Margin]), 2)
}
```

13. Optimize a slow DAX measure using variables (VAR).
    - If a complex expression within your measure is used multiple times, assigning its result to a variable using VAR means the expression is evaluated only once. Subsequent references to the variable retrieve the stored result, significantly reducing processing time, especially for computationally intensive parts of the measure.
    - Example:
      Consider a measure calculating Year-over-Year (YoY) Sales Growth:

      ```
      Sales YoY Growth % =
      DIVIDE(
          ([Sales] - CALCULATE([Sales], PARALLELPERIOD('Date'[Date], -12, MONTH))),
          CALCULATE([Sales], PARALLELPERIOD('Date'[Date], -12, MONTH))
      )
      ```
    - We can optimize it as following:

      ```
      Sales YoY Growth % =
      VAR SalesPriorYear = CALCULATE([Sales], PARALLELPERIOD('Date'[Date], -12,
      MONTH))
      RETURN
          DIVIDE(
              ([Sales] - SalesPriorYear),
              SalesPriorYear
          )
      ```
14. Use CALCULATE() to override a filter
    - To use CALCULATE() to override a filter in DAX, you can use functions like ALL, ALLEXCEPT, or REMOVEFILTERS within the CALCULATE function's filter arguments to remove the existing filter context. For example:

      ```
      Total Sales without Filter = CALCULATE(
          SUM(ForDax[Sales]),
          ALL(ForDax[ProductID])
          )
      ```
      This will ignore all filter applied to ForDax[ProductID].

15. Write a measure that returns the highest sales amount

```
Highest Sale = MAX(
    ForDax[Sales])
```