Lesson 9

Topic: Understanding Context in DAX & CALCULATE and Basic Filters, Variables
Prerequisites: Download DAX_Context_Practice.xlsx file.

1. What is row context? Give an example in a calculated column.
    - Row context evaluates an expression for a single, current row of a table, as seen in calculated columns or iterators like SUMX.

    Example:
    In a calculated column, [Quantity] * [Price] multiplies the quantity and price for that specific row.
2. Write a measure that finds total sales.

    ```
    Total Sales = SUM(DAX_Practice_Data[Sales])
    ```

3. Use RELATED to fetch the Name from the Customers table into the Sales table.

    ```
    Customer Name = RELATED('Customer Lookup'[LastName])&" "&RELATED('Customer Lookup'[FirstName])
    ```
4. What does CALCULATE(SUM(Sales[Quantity]), Sales[Category] = "Electronics") return?
    - The total quantity of items sold, but only for products in the "Electronics" category, by evaluating the SUM of the Quantity column within a filter context that includes only "Electronics" sales.
5. Explain the difference between VAR and RETURN in DAX.

    VAR (Variable Declaration):

    VAR is used to declare a variable and assign the result of an expression to it.

    The expression assigned to a VAR is evaluated once when the variable is defined, and its value is stored. This stored value can then be reused multiple times within the subsequent RETURN statement or other VAR declarations.

    Variables declared with VAR improve readability by breaking down complex calculations into smaller, named parts.

    RETURN (Result Expression):

RETURN signifies the end of the variable declarations and introduces the final expression that defines the result of the measure or query.

The expression after RETURN can use any of the variables previously defined with VAR.

This is the part of the DAX code that actually produces the final output.

6. Create a calculated column in Sales called TotalPrice using row context (Quantity * UnitPrice).

```
= Table.AddColumn(#"Added Conditional Column", "TotalPrice",
each [quantity]*[Cost])
```

7. Write a measure Electronics Sales using CALCULATE to sum sales only for the "Electronics" category.

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

8. Use ALL(Sales[Category]) in a measure to show total sales ignoring category filters.

```
All Sales without Filter = CALCULATE(
    SUM(Sales[Amount]),
    All(Products[Category]))
```

9. Fix this error: A calculated column in Sales uses RELATED(Customers[Region]) but returns blanks.
    - RELATED(Customers[Region]), you must verify a many-to-one relationship exists between the Sales and Customers tables, ensure the relationship is active, and check for a referential integrity mismatch or extra spaces in the connecting columns. If these issues are resolved and blanks persist, use an IF or COALESCE function to handle the blanks, or consider if the calculation belongs in a measure instead.
10. Why does CALCULATE override existing filters?
    - The CALCULATE function overrides existing filters on the same columns mentioned in its filter arguments, effectively replacing the current filter

context for those specific columns. This behavior is automatic and a fundamental aspect of how CALCULATE modifies filter contexts, providing a new, independent context for its expression. Filters on other columns, not involved in the CALCULATE's filter arguments, remain in effect.

11. Write a measure that returns average unitprice of products

```
Average UnitPrice = DIVIDE(
    SUM(DAX_Practice_Data[Sales]),
    SUM(DAX_Practice_Data[quantity]),
    0
)
```

12. Use VAR to store a temporary table of high-quantity sales (Quantity > 2), then count rows.

```
HighQuantitySalesCount =
    VAR HighQuantitySale =
    FILTER(DAX_Practice_Data,
        DAX_Practice_Data[quantity]>2)
    RETURN
        COUNTROWS(HighQuantitySale)
```

13. Write a measure `% of Category Sales` that shows each sale's contribution to its category total.

```
% of Category Sales = DIVIDE(
    SUM(Sales[Amount]),
    CALCULATE(
        SUM(Sales[Amount]),
        ALL(Products[Category])
    )
)
```

14. Simulate a "remove filters" button using ALL in a measure.

    Create a "Clear Filters" Bookmark:

    Apply any desired filters to your report, then clear them manually (e.g., by clicking "Clear filters" on slicers or removing filters from the Filters pane).

Go to the View tab in Power BI Desktop and select Bookmarks.

Click Add to create a new bookmark.

Rename the bookmark to something descriptive, like "Clear Filters."

Ensure that the "Data" option is checked for the bookmark, and uncheck "Display," "Current Page," and "Selected Visuals" if you only want to clear filters.

Create a Button and Assign the Bookmark:

Go to the Insert tab and select Buttons > Blank.

Place the button on your report page.

With the button selected, go to the Format pane and turn on the Action property.

Set the Type to "Bookmark" and select the "Clear Filters" bookmark you created.

Add text to the button (e.g., "Clear Filters") and format it as desired.

Another option with dax:

```
Total Sales Unfiltered = CALCULATE(SUM(Sales[Amount]),
ALL(Sales))
```

15. Troubleshoot: A CALCULATE measure ignores a slicer. What's the likely cause?
    - A CALCULATE measure in Power BI likely ignores a slicer because the DAX expression inside CALCULATE is missing a filter context, or it explicitly uses `ALL()` or a similar ALL() family function to remove filters from the relevant table. To fix this, you need to ensure that the measure's definition includes the appropriate context by using ALL() or ALLSELECTED() on specific tables or columns to remove unwanted filters, or use ALLEXCEPT() to preserve filters from other tables.