Lesson 10

Topic: Advanced Filtering in DAX
Prerequisites: Download Lesson 10.xlsx file.

1. **What does FILTER(Sales, Sales[Amount] > 1000) return?**

   The DAX expression:

   DAX

   FILTER(Sales, Sales[Amount] > 1000)

   Returns:

   A table (not a single value) that contains only the rows from the Sales table where the [Amount] column is greater than 1000.

   More specifically:

- FILTER(...) is a table function in DAX.
- It does not evaluate to a number or total by itself.
- It creates a new table that keeps the structure of the Sales table but includes only the rows that meet the condition: Sales[Amount] > 1000.

2. **Write a measure High Sales that sums Amount where Amount > 1000 using FILTER.**

   Sure! Here's how you can write a DAX measure called High Sales that sums the Amount column only where Amount > 1000 using the FILTER function:

   Measure: High Sales

   DAX

```
High Sales :=
CALCULATE(
    SUM(Sales[Amount]),
    FILTER(Sales, Sales[Amount] > 1000)
)
```

Explanation:

- CALCULATE(...) changes the context of the calculation.
- SUM(Sales[Amount]) is the expression being evaluated.
- FILTER(Sales, Sales[Amount] > 1000) creates a new row context where only rows with Amount > 1000 are included.

3. How does ALLEXCEPT(Sales, Sales[Region]) differ from ALL(Sales)?

ALL(Sales)

DAX

ALL(Sales)

🔸 Removes all filters from the entire Sales table.

Use case: If you're calculating a total that ignores all filters (region, product, date, etc.)

Example:

DAX

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

🔹 This will return the total of Sales[Amount] across all data, ignoring any slicers or filters in your report.

🔹 ALLEXCEPT(Sales, Sales[Region])

DAX

ALLEXCEPT(Sales, Sales[Region])

🔸 Removes all filters from the Sales table except for the Region column.

Use case: If you want to keep the Region filter, but ignore everything else (like ProductID, SaleDate, etc.)

Example:

DAX

```
Total Sales by Region :=
CALCULATE(
    SUM(Sales[Amount]),
    ALLEXCEPT(Sales, Sales[Region])
)
```

- This will return total sales per Region, ignoring other filters (like product, date), but preserving the Region grouping.

4. **Use SWITCH to categorize Amount:**

   "Medium" if 500–1000

   "High" if > 1000 ```

   Sure! Let's use the SWITCH function in DAX to categorize Amount values into custom ranges.

   Goal: Categorize Sales[Amount] into labels like:

- "Low" for Amount ≤ 500
- "Medium" for Amount between 501 and 1000
- "High" for Amount > 1000

Measure (or Calculated Column):

DAX

```
Amount Category :=
SWITCH(
    TRUE(),
    Sales[Amount] <= 500, "Low",
```

```
    Sales[Amount] <= 1000, "Medium",
    Sales[Amount] > 1000, "High",
    "Unknown"
)
```

Explanation:

- SWITCH(TRUE(), ...) lets us test multiple conditions like an IF-ELSE ladder.
- It checks the conditions in order:
    1. If Amount <= 500 → "Low"
    2. Else if Amount <= 1000 → "Medium"
    3. Else if Amount > 1000 → "High"
- If none match, it returns "Unknown" (default/fallback).

5. **What is the purpose of ALLSELECTED?**

   ALLSELECTED() is a DAX function that returns all the values that are currently selected in the report, including filters applied by slicers, visuals, or cross-filters — but ignoring row context (like inside a table cell).

   Purpose of ALLSELECTED:

   To preserve slicer or visual-level filters while removing row-level filters in calculations like percent of total, running totals, etc.

   Example: % of Total (Respecting Slicers)

   Suppose you have a table showing sales by region, and you use this measure:

   DAX

```
% of Visible Total :=
DIVIDE(
```

```
    SUM(Sales[Amount]),
    CALCULATE(SUM(Sales[Amount]), ALLSELECTED(Sales))
)
```

🔹 This shows the percent of the total visible in the report, based on slicer selections.

6. **Write a measure Regional Sales % showing each sale's contribution to its region's total (use ALLEXCEPT).**

   Great! Here's how to write a DAX measure called Regional Sales % that shows each sale's contribution to its region's total, using ALLEXCEPT.

   Measure: Regional Sales %

   DAX

```
Regional Sales % :=
DIVIDE(
    SUM(Sales[Amount]),
    CALCULATE(
        SUM(Sales[Amount]),
        ALLEXCEPT(Sales, Sales[Region])
    )
)
```

   Explanation:

- SUM(Sales[Amount]): current sale amount (per row or context).
- ALLEXCEPT(Sales, Sales[Region]): removes all filters except Region, so the total is calculated for the entire Region.
- DIVIDE(...): safely divides sale amount by the regional total.

7. **Create a dynamic measure using SWITCH to toggle between SUM, AVERAGE, and COUNT of Amount.**

Excellent! Here's how to create a dynamic measure using the SWITCH function to toggle between SUM, AVERAGE, and COUNT of the Amount column based on a slicer selection.

Goal:

Let the user choose between:

- "SUM"
- "AVERAGE"
- "COUNT"

Using a slicer (from a helper table), the measure will dynamically respond.

Step-by-Step Instructions

- Step 1: Create a helper table (manually or with DAX)

You can create a table called CalculationType with these values:

CalculationType

SUM

AVERAGE

COUNT

If creating with DAX:

DAX

```
CalculationType = DATATABLE(
    "CalculationType", STRING,
    {
        {"SUM"},
        {"AVERAGE"},
        {"COUNT"}
    }
)
```

Add this table to your report, then create a slicer on CalculationType[CalculationType].

- Step 2: Create the dynamic measure

DAX

```
Dynamic Measure :=
SWITCH(
    SELECTEDVALUE(CalculationType[CalculationType]),
    "SUM", SUM(Sales[Amount]),
    "AVERAGE", AVERAGE(Sales[Amount]),
    "COUNT", COUNT(Sales[Amount]),
    BLANK()
)
```

Explanation:

- SELECTEDVALUE() gets the selected calculation type from the slicer.
- SWITCH(...) picks the right aggregation based on the selected type.
- BLANK() is returned if no valid option is selected.

Example Usage in Power BI:

1. Add the CalculationType[CalculationType] field to a slicer.
2. Place Dynamic Measure in a card or table.
3. The measure will update based on your slicer choice: SUM, AVERAGE, or COUNT.

8. Use FILTER inside CALCULATE to exclude "Furniture" sales (Products[Category] = "Furniture").

Measure: Sales Without Furniture

DAX

```
Sales Without Furniture :=
CALCULATE(
```

```
    SUM(Sales[Amount]),
    FILTER(
        Products,
        Products[Category] <> "Furniture"
    )
)
```

Explanation:

- CALCULATE(...): Changes the context of the calculation.
- SUM(Sales[Amount]): Adds up the sales amount.
- FILTER(Products, Products[Category] <> "Furniture"): Keeps only non-Furniture products.

9. Why might ALLSELECTED behave unexpectedly in a pivot table?

ALLSELECTED can behave unexpectedly in pivot tables because it depends on user-selected filters (like slicers or visuals) — and pivot rows/columns are not treated the same as slicers.

10. Write a measure that calculates total sales and ignores filters from region

Measure: Total Sales (Ignore Region)

DAX

```
Total Sales Ignore Region :=
CALCULATE(
    SUM(Sales[Amount]),
    REMOVEFILTERS(Sales[Region])
)
```

Explanation:

- SUM(Sales[Amount]): Calculates the total sales amount.
- REMOVEFILTERS(Sales[Region]): Removes any filters (from slicers, visuals, etc.) that are applied to the Region column.

11. **Optimize this measure:**
High Sales = CALCULATE(SUM(Sales\[Amount]), FILTER(Sales, Sales\[Amount] > 1000)) (Hint: Replace FILTER with a Boolean filter inside CALCULATE.)

Original Measure (Less Efficient):

DAX

```
High Sales :=
CALCULATE(
    SUM(Sales[Amount]),
    FILTER(Sales, Sales[Amount] > 1000)
)
```

Optimized Measure (More Efficient):

DAX

```
High Sales :=
CALCULATE(
    SUM(Sales[Amount]),
    Sales[Amount] > 1000
)
```

Why it's better:

Aspect	Explanation
Faster	DAX engine doesn't have to iterate row by row (like in FILTER)
Cleaner syntax	Easier to read and write
Same result	As long as the column is in the same table you're filtering

12. **Write a measure Top 2 Products using TOPN and FILTER to show the highest-grossing products.**

Show only the top 2 products by total sales amount (SUM(Sales[Amount])).
Assumptions:

- You have a Sales table with ProductID and Amount.
- You have a Products table with ProductID, ProductName.
- A relationship exists between Sales[ProductID] and Products[ProductID].

Step-by-Step DAX Measure

- 1. Create a measure to show only Top 2 Products:

DAX

```
Top 2 Product Sales :=
VAR Top2Products =
    TOPN(
        2,
        ADDCOLUMNS(
            VALUES(Products[ProductID]),
            "TotalSales", CALCULATE(SUM(Sales[Amount]))
        ),
        [TotalSales],
        DESC
    )
RETURN
CALCULATE(
    SUM(Sales[Amount]),
    FILTER(
        Products,
        Products[ProductID] IN
```

```
        SELECTCOLUMNS(Top2Products, "ProductID",
Products[ProductID])
    )
)
```

- OR a simpler approach in visuals:

If you want to limit visuals (like a table or bar chart) to top 2 products:

Create a ranking measure:

DAX

```
Product Rank :=
RANKX(
    ALL(Products),
    CALCULATE(SUM(Sales[Amount])),
    ,
    DESC
)
```

Then apply a visual-level filter:
Product Rank <= 2

This is more flexible and recommended in Power BI dashboards.

13. Use ALLSELECTED with no parameters to respect slicers but ignore visual-level filters.

Use ALLSELECTED() to:

- Respect slicers (like Product slicers, Date ranges, etc.)
- Ignore visual-level filters (like those directly applied to a chart or matrix)

Example Measure: % of Total (Respect Slicers)

DAX

```
% of Total (Slicers Only) :=
```

```
DIVIDE(
    SUM(Sales[Amount]),
    CALCULATE(
        SUM(Sales[Amount]),
        ALLSELECTED()
    )
)
```

How it works:

Part	What it does
SUM(Sales[Amount])	Current value in context (row)
ALLSELECTED()	Removes only visual filters, but keeps slicer filters
DIVIDE(..., ...)	Safe division (avoids divide-by-zero)

Scenario:

- You apply a Product slicer: ✅ this measure respects it.
- You add a Region filter directly on the visual: ❌ it ignores that.
- So it shows each value as a % of the slicer-selected total, not just the visible chart rows.

14. Debug: A SWITCH measure returns incorrect values when fields are added to a matrix visual.

DEBUG: SWITCH Measure Returns Incorrect Values in Matrix Visual

Problem:

You create a SWITCH() measure to toggle between different calculations:

DAX

```
Selected Measure :=
```

```
SWITCH(
    SELECTEDVALUE(MeasureTable[MeasureName]),
    "Total Sales", SUM(Sales[Amount]),
    "Total Quantity", SUM(Sales[Quantity]),
    "Average Sales", AVERAGE(Sales[Amount])
)
```

It works in a card or single-value visual, but shows wrong or blank values in a matrix when you add fields like Region or Product.

Why It Happens:

- SWITCH() depends on SELECTEDVALUE(...), often coming from a disconnected slicer table.
- In a matrix, multiple rows are evaluated at once, and SELECTEDVALUE(...) might return blank or unexpected value because:
    - There's no single selected value for the slicer at that level.
    - You added matrix rows that break context.

Fix: Create an Independent Calculation Table

Step 1: Create a Measure Selection Table (manually or in Power BI):

MeasureName

Total Sales

Total Quantity

Average Sales

Load it as a disconnected table (don't link to anything in the model).

Step 2: Create a Proper Measure Using ISFILTERED() or HASONEVALUE() check:

DAX

```
Selected Measure :=
VAR Selected = SELECTEDVALUE(MeasureTable[MeasureName])
RETURN
SWITCH(
    TRUE(),
    Selected = "Total Sales", SUM(Sales[Amount]),
    Selected = "Total Quantity", SUM(Sales[Quantity]),
    Selected = "Average Sales", AVERAGE(Sales[Amount])
)
```

Use TRUE() pattern instead of nested SWITCH()s — it avoids blanks due to context confusion.

15.Simulate a "reset filters" button using ALL in a measure.

You want to show total values ignoring slicer selections — as if filters were reset.

Step-by-step Example: Sample Measure:

DAX

```
Reset Sales :=
CALCULATE(
   SUM(Sales[Amount]),
   ALL(Sales)
)
```

What does this do?

- Ignores ALL filters from:
    - Slicers
    - Visual filters
    - Page filters
- Returns the grand total of Sales[Amount], as if nothing was selected.

If you want to ignore only a specific slicer (e.g. Region):

DAX

CopyEdit

```
Reset Sales (Ignore Region) :=
CALCULATE(
    SUM(Sales[Amount]),
    ALL(Sales[Region])
)
```

This ignores just the Region slicer but respects other slicers (like Product or Date).

How to Simulate the "Reset" Button Visually

1. Create a card visual or measure with the Reset Sales measure.
2. Label it something like "Total Sales (All)".
3. Optionally, create a button with a bookmark that clears filters visually — but DAX will handle this internally.