1. What does DAX stand for?

DAX = Data Analysis Expressions
It's the formula language in Power BI, Excel Power Pivot, and SSAS Tabular models.

---

2. Write a DAX formula to sum the Sales column.

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

---

3. What is the difference between a calculated column and a measure?

- **Calculated Column**: Stored in the table, calculated row by row, increases data model size.
- **Measure**: Calculated on the fly, depends on filters/context, more efficient.

---

4. Use the DIVIDE function to calculate Profit Margin (Profit/Sales).

```
Profit Margin = DIVIDE(SUM(Retail_Sales_Data[Profit]),
SUM(Retail_Sales_Data[Sales]), 0)
```

(*The 0 is the alternate result if denominator = 0*).

---

5. What does COUNTROWS() do in DAX?

It **returns the number of rows** in a table. Example:

```
Number of Orders = COUNTROWS(Retail_Sales_Data)
```

---

6. Create a measure: Total Profit that subtracts total cost from total sales

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

---

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

```
Average Sales per Product = AVERAGEX(VALUES(Retail_Sales_Data[Product]),
SUM(Retail_Sales_Data[Sales]))
```

---

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

(As a calculated column example:)

```
Profit Tag = IF(Retail_Sales_Data[Profit] > 1000, "High Profit", "Low
Profit")
```

---

9. What is a circular dependency error in a calculated column?

It happens when a calculated column depends on itself (directly or indirectly), creating an endless loop. Example: Column A references Column B, and Column B references Column A.

---

10. Explain row context vs. filter context.

- **Row Context**: When DAX evaluates row by row (e.g., in calculated columns or iterators like SUMX).
- **Filter Context**: Filters applied by slicers, visuals, or CALCULATE that affect what data is visible for calculation.

---

11. Write a measure to calculate YTD Sales using TOTALYTD().

```
YTD Sales = TOTALYTD(SUM(Retail_Sales_Data[Sales]), Retail_Sales_Data[Date])
```

---

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

```
Selected Measure =
SWITCH(
    SELECTEDVALUE(Metrics[Metric]),
    "Sales", SUM(Retail_Sales_Data[Sales]),
    "Profit", SUM(Retail_Sales_Data[Profit]),
    "Margin", DIVIDE(SUM(Retail_Sales_Data[Profit]),
SUM(Retail_Sales_Data[Sales]), 0)
)
```

(*Requires a disconnected table* `Metrics` *with values Sales, Profit, Margin*).

---

13. Optimize a slow DAX measure using variables (VAR).

Instead of repeating SUM calculations:

```
Profit Margin Optimized =
VAR TotalSales = SUM(Retail_Sales_Data[Sales])
VAR TotalProfit = SUM(Retail_Sales_Data[Profit])
RETURN DIVIDE(TotalProfit, TotalSales, 0)
```

---

14. Use CALCULATE() to override a filter

Example: Sales ignoring Product filter:

```
Sales All Products = CALCULATE(SUM(Retail_Sales_Data[Sales]),
ALL(Retail_Sales_Data[Product]))
```

15. Write a measure that returns the highest sales amount

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

Or across aggregated values:

```
Max Sales Amount = MAXX(VALUES(Retail_Sales_Data[Product]),
SUM(Retail_Sales_Data[Sales]))
```