1. It’s the **formula language** used in:

- **Power BI**
- **Excel Power Pivot**
- **SQL Server Analysis Services (SSAS) Tabular**

🔑 Asosiy vazifasi:

- Calculated **columns** va **measures** yozish
- Aggregations, filtering, time intelligence (YoY growth, moving average, cumulative total) kabi hisob-kitoblarni qilish.

👉 Uni **Excel formulasiga o‘xshash**, lekin **data model** (relationships, filters, context) bilan ishlaydigan, ko‘proq OLAP uchun mo‘ljallangan kuchli til deb tushunish mumkin.

Would you like me to give you a **real-life simple example** of a DAX formula vs. an Excel

3. 🔹 **Calculated Column**

- **Row by row**: Evaluated for each row in a table.
- **Stored in the model**: Once created, it becomes part of the table and consumes memory.
- **Static**: Value doesn’t change with slicers or filters; it’s fixed per row.
- **Use case**: When you need a new field for filtering, grouping, or creating relationships.

✅ Example:

Profit = Sales[SalesAmount] - Sales[Cost]

👉 Creates a new column in the Sales table with profit for every transaction row.

🔹 **Measure**

- **Context dependent**: Calculated at query time, based on filters/slicers/visuals.
- **Not stored**: It’s calculated on the fly, so it doesn’t bloat the model.
- **Dynamic**: Value changes depending on the filter context.
- **Use case**: KPIs, aggregations, dashboards (like totals, averages, ratios).

✅ Example:

Total Profit = SUM(Sales[SalesAmount]) - SUM(Sales[Cost])

👉 This will give total profit for whatever filters (time period, region, product) are active in the visual.

📊 Key Difference Summary:

Feature	Calculated Column	Measure
Evaluation	Row by row	Based on filter context
Storage	Stored in table	Not stored (calculated on demand)
Performance impact	Uses memory	Uses CPU when queried
Best for	Filtering, relationships	Aggregations, KPIs

👉 Easy way to remember:

- **Calculated column** = new data field for each row.
- **Measure** = dynamic number that changes with your report filters.

Do you want me to show you a **Power BI example with visuals** where the same formula behaves differently as a column vs as a measure?

9. In **Power BI (DAX)** a **circular dependency error** happens when a **calculated column or measure depends on itself directly or indirectly**.

🔄 **Meaning:**
If Column A's formula references Column B, and Column B's formula also references Column A (directly or through another column/measure), Power BI cannot determine which one should be calculated first → this creates a *circular dependency*.

🔧 Example of Circular Dependency

Column A = Sales[Column B] + 1

Column B = Sales[Column A] + 1

Here, Column A depends on Column B and Column B depends on Column A → infinite loop → ❌ circular dependency error.

⚠️ Common Causes

1. Trying to create a calculated column that references itself.

2. Using a column in a calculation that indirectly depends on the result of the same calculation.
3. Complex relationships between tables where calculated columns rely on each other.

✅ How to Fix It

- Break the dependency by rethinking the logic.
- Move the calculation into a **measure** (since measures calculate in query context, not stored row by row).
- Use variables (VAR) inside the formula to store intermediate values.
- Restructure your data model (sometimes extra calculated columns aren't needed).

10. 🔹 **Row Context**

- **Definition:** Row context exists when DAX evaluates an expression **row by row**.
- It's like saying: *"I'm currently looking at this row, so I can access its column values directly."*
- **Where it happens:**
    - In **calculated columns** (because they are evaluated for each row).
    - In functions like SUMX, FILTER, ADDCOLUMNS — they create a row context.

✅ Example:

Profit = Sales[SalesAmount] - Sales[Cost]

- If this is a calculated column, DAX goes row by row, subtracting Cost from SalesAmount.
- That's **row context** in action.

🔹 Filter Context

- **Definition:** Filter context is the set of filters applied to evaluate a DAX expression.
- It comes from:
    - Slicers in reports
    - Filters on visuals

    - Relationships between tables
    - CALCULATE() function, which **modifies filter context**

✅ **Example:**

Total Sales = SUM(Sales[SalesAmount])

- If you drop this measure in a table by Product:
    - Filter context = "only rows where Product = A" → sum only those SalesAmounts.

🔹 **Difference**

- **Row Context** = operates *inside a single row*.
- **Filter Context** = defines *which rows are visible* before evaluation.

🔹 **Key Interaction: Context Transition**

When you use CALCULATE(), it **converts row context into filter context**. That's why measures inside iterators like SUMX work correctly.

✅ **Example:**

Total Sales per Row = CALCULATE(SUM(Sales[SalesAmount]))

- Here, CALCULATE turns the row context into a filter so the measure knows which rows to sum.

👉 Think of it like this:

- **Row context** = "I'm looking at *this row's* data."
- **Filter context** = "I'm looking at *these rows* because of filters/slicers/relationships."