3.2. Student Handout

Introduction to DAX (Data Analysis Expressions) - Student Handout

---

What is DAX?

DAX (Data Analysis Expressions) is a formula language used in Power BI, Excel, and other Microsoft tools to perform calculations on data. It consists of functions, operators, and constants that help calculate and return values.

Why Use DAX?

1. **Handling Large Datasets**: Efficiently processes millions of rows of data.
2. **Relational Data**: Works seamlessly with multiple related tables.
3. **Advanced Calculations**: Supports complex calculations like running totals and custom aggregations.

DAX vs Excel Formulas

Feature	Excel Formulas	DAX
Scope	Individual cells or ranges	Entire tables or columns
Data Size	Limited by Excel’s row/column limits	Handles millions of rows in Power BI
Relational Data	Not designed for relational data	Designed for multiple related tables
Performance	Slower with large datasets	Optimized for large datasets
Complexity	Basic calculations	Advanced calculations and filters

---

Introduction to Calculated Columns, Measures, and Tables

1. **Calculated Columns**: New columns added to your data model, calculated row by row.

- Example 1: `Profit = Sales[Amount] - Sales[Cost]`
- Example 2: `Discounted Price = Sales[Amount] * (1 - Sales[Discount])`
- Example 3: `Full Name = Customers[FirstName] & " " & Customers[LastName]`

2. **Measures**: Calculations performed dynamically based on filters.

- Example 1: `Total Sales = SUM(Sales[Amount])`
- Example 2: `Average Profit = AVERAGE(Sales[Profit])`
- Example 3: `Max Order Value = MAX(Sales[Amount])`

3. **Calculated Tables**: New tables created using DAX formulas.

- Example 1: `Filtered Sales = FILTER(Sales, Sales[Amount] > 1000)`
- Example 2: `Top Customers = TOPN(10, Customers, Customers[TotalSales])`
- Example 3: `Sales Summary = SUMMARIZE(Sales, Sales[Region], "Total", SUM(Sales[Amount]))`

Basic DAX Syntax and Functions

- **Functions**: Predefined formulas for calculations.
- **Columns**: Columns in your data model referenced in formulas.
- **Operators**: Symbols like `+`, `-`, `*`, and `/` for arithmetic operations.

Common DAX Functions

1. **SUM**: Adds up values in a column.

- Example 1: `Total Sales = SUM(Sales[Amount])`
- Example 2: `Total Cost = SUM(Sales[Cost])`
- Example 3: `Total Quantity = SUM(Sales[Quantity])`

2. **AVERAGE**: Calculates the average of values in a column.

- Example 1: `Average Sales = AVERAGE(Sales[Amount])`

- Example 2: `Average Discount = AVERAGE(Sales[Discount])`
- Example 3: `Average Quantity = AVERAGE(Sales[Quantity])`

3. **COUNT**: Counts the number of rows in a column.

- Example 1: `Total Orders = COUNT(Sales[OrderID])`
- Example 2: `Total Products = COUNT(Products[ProductID])`
- Example 3: `Total Customers = COUNT(Customers[CustomerID])`

4. **DISTINCTCOUNT**: Counts the number of unique values in a column.

- Example 1: `Unique Customers = DISTINCTCOUNT(Sales[CustomerID])`
- Example 2: `Unique Products = DISTINCTCOUNT(Sales[ProductID])`
- Example 3: `Unique Regions = DISTINCTCOUNT(Sales[Region])`

Introduction to Row Context and Filter Context in DAX

- **Row Context**: Refers to the current row in a table. Used in calculated columns.
- Example 1: `Profit = Sales[Amount] - Sales[Cost]`
- Example 2: `Net Price = Sales[Amount] - Sales[Discount]`
- Example 3: `Full Name = Customers[FirstName] & " " & Customers[LastName]`
- **Filter Context**: Refers to filters applied to data. Used in measures.
- Example 1: `Total Sales = SUM(Sales[Amount])`
- Example 2: `Filtered Sales = CALCULATE(SUM(Sales[Amount]), Sales[Region] = "West")`
- Example 3: `Yearly Sales = CALCULATE(SUM(Sales[Amount]), YEAR(Sales[Date]) = 2023)`

Hands-On: Writing Simple DAX Expressions for Data Calculations

1. **Total Sales**:

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

2. **Average Sales**:

```
Average Sales = AVERAGE(Sales[Amount])
```

3. **Total Orders**:

```
Total Orders = COUNT(Sales[OrderID])
```

4. **Unique Customers**:

```
Unique Customers = DISTINCTCOUNT(Sales[CustomerID])
```

Conclusion

DAX is a powerful language for performing complex calculations in Power BI. It is designed to handle large datasets and relational data, offering advanced calculation capabilities. By mastering DAX syntax, calculated columns, measures, and common functions, you can create dynamic and insightful reports.

Next Steps

In the next session, we will explore advanced DAX concepts, including time intelligence functions, iterators, and advanced filtering. Practice writing simple DAX expressions to enhance your understanding.

Thank you for participating in this session! If you have any questions, feel free to ask.