4.2. Student Handout

Intermediate DAX Functions Training - Student Handout

Welcome to the Intermediate DAX Functions Training! This handout will guide you through the key concepts and functions covered in the session. By the end, you'll be equipped to create dynamic measures, work with time intelligence, and optimize calculations using variables in Power BI.

What is DAX?

DAX (Data Analysis Expressions) is a formula language used in Power BI, Excel Power Pivot, and SQL Server Analysis Services (SSAS) for custom calculations on data. It enables advanced data manipulation and analysis beyond simple operations.

Intermediate DAX Functions Overview

1. Logical Functions: IF, SWITCH

IF Function

The IF function checks a condition and returns one value if true and another if false.

Syntax:

```
IF(Condition, Result if True, Result if False)
```

Examples:

1. Categorize sales as "High" if greater than 100,000, otherwise "Low":

```
Sales Category = IF(Sales[Amount] > 100000, "High", "Low")
```

2. Determine if a product is "In Stock" or "Out of Stock":

```
Stock Status = IF(Inventory[Quantity] > 0, "In Stock", "Out of Stock")
```

3. Check if a customer is "VIP" based on purchase history:

```
VIP Status = IF(Customer[TotalPurchases] > 5000, "VIP", "Regular")
```

SWITCH Function

The SWITCH function evaluates multiple conditions without nesting multiple IF statements.

Syntax:

```
SWITCH(Expression, Value1, Result1, Value2, Result2, ..., ElseResult)
```

Examples:

1. Categorize sales into "Low", "Medium", and "High":

```
Sales Category = SWITCH(
TRUE(),
Sales[Amount] < 50000, "Low",
Sales[Amount] >= 50000 && Sales[Amount] < 100000, "Medium",
Sales[Amount] >= 100000, "High"
```

```
)
```

2. Assign grades based on scores:

```
Grade = SWITCH(
TRUE(),
Scores[Value] >= 90, "A",
Scores[Value] >= 80, "B",
Scores[Value] >= 70, "C",
"F"
)
```

3. Determine shipping method based on order size:

```
Shipping Method = SWITCH(
TRUE(),
Orders[Size] < 10, "Standard",
Orders[Size] >= 10 && Orders[Size] < 50, "Express",
"Freight"
)
```

---

2. Time Intelligence Functions: TOTALYTD, SAMEPERIODLASTYEAR

TOTALYTD Function

Calculates the year-to-date total for a given measure.

Syntax:

```
TOTALYTD(Expression, Dates)
```

Examples:

1. Calculate year-to-date sales:

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

2. Year-to-date profit calculation:

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

3. Year-to-date expenses:

```
YTD Expenses = TOTALYTD(SUM(Expenses[Amount]), Dates[Date])
```

SAMEPERIODLASTYEAR Function

Returns the same period from the previous year for year-over-year comparisons.

Syntax:

```
SAMEPERIODLASTYEAR(Dates)
```

Examples:

1. Compare this year's sales with last year's:

```
Sales Last Year = CALCULATE(SUM(Sales[Amount]),
SAMEPERIODLASTYEAR(Dates[Date]))
```

2. Last year's profit:

```
Profit Last Year = CALCULATE(SUM(Sales[Profit]),
SAMEPERIODLASTYEAR(Dates[Date]))
```

3. Last year's expenses:

```
Expenses Last Year = CALCULATE(SUM(Expenses[Amount]),
SAMEPERIODLASTYEAR(Dates[Date]))
```

---

3. Aggregation and Filtering Functions: CALCULATE, FILTER, ALL

CALCULATE Function

Changes the context of a calculation by applying filters.

Syntax:

```
CALCULATE(Expression, Filter1, Filter2, ...)
```

Examples:

1. Total sales for a specific region:

```
Sales in North = CALCULATE(SUM(Sales[Amount]), Sales[Region] = "North")
```

2. Total profit for a specific product category:

```
Profit for Electronics = CALCULATE(SUM(Sales[Profit]), Products[Category] = "Electronics")
```

3. Total expenses for a specific department:

```
Expenses for HR = CALCULATE(SUM(Expenses[Amount]), Departments[Name] = "HR")
```

FILTER Function

Returns a table filtered based on a condition.

Syntax:

```
FILTER(Table, Condition)
```

Examples:

1. Filter sales greater than 100,000:

```
High Sales = FILTER(Sales, Sales[Amount] > 100000)
```

2. Filter products with low inventory:

```
Low Inventory = FILTER(Inventory, Inventory[Quantity] < 10)
```

3. Filter customers with high purchase history:

```
High Value Customers = FILTER(Customers, Customers[TotalPurchases] > 5000)
```

ALL Function

Removes all filters from a table or column.

Syntax:

```
ALL(Table)
```

Examples:

1. Calculate percentage of total sales for each region:

```
% of Total Sales = DIVIDE(SUM(Sales[Amount]), CALCULATE(SUM(Sales[Amount]),
ALL(Sales[Region])))
```

2. Calculate percentage of total profit for each product:

```
% of Total Profit = DIVIDE(SUM(Sales[Profit]), CALCULATE(SUM(Sales[Profit]),
ALL(Products[Name])))
```

3. Calculate percentage of total expenses for each department:

```
% of Total Expenses = DIVIDE(SUM(Expenses[Amount]),
CALCULATE(SUM(Expenses[Amount]), ALL(Departments[Name])))
```

---

4. Working with Hierarchies and Parent-Child Relationships

PATH Function

Returns the path from a child to a parent in a parent-child hierarchy.

Syntax:

```
PATH(ChildColumn, ParentColumn)
```

Examples:

1. Create a path from employee to top-level manager:

```
Employee Path = PATH(Employee[EmployeeID], Employee[ManagerID])
```

2. Create a path for product categories:

```
Category Path = PATH(Products[CategoryID], Products[ParentCategoryID])
```

3. Create a path for organizational units:

```
Org Unit Path = PATH(OrgUnits[UnitID], OrgUnits[ParentUnitID])
```

---

5. Creating Dynamic Measures

Dynamic measures change based on user selections or filters.

Examples:

1. Dynamic measure for total sales or profit:

```
Dynamic Measure = SWITCH(

SELECTEDVALUE(MeasureSelection[Measure]),

"Sales", SUM(Sales[Amount]),
```

```
"Profit", SUM(Sales[Profit])
)
```

2. Dynamic measure for revenue or cost:

```
Dynamic RevenueCost = SWITCH(
SELECTEDVALUE(MeasureSelection[Measure]),
"Revenue", SUM(Revenue[Amount]),
"Cost", SUM(Cost[Amount])
)
```

3. Dynamic measure for quantity or discount:

```
Dynamic QuantityDiscount = SWITCH(
SELECTEDVALUE(MeasureSelection[Measure]),
"Quantity", SUM(Sales[Quantity]),
"Discount", SUM(Sales[Discount])
)
```

---

6. Using Variables in DAX Expressions

Variables store calculation results for reuse, improving readability and performance.

Syntax:

```
VAR VariableName = Expression

RETURN Result
```

Examples:

1. Calculate profit margin using variables:

```
Profit Margin =

VAR TotalSales = SUM(Sales[Amount])

VAR TotalProfit = SUM(Sales[Profit])

RETURN DIVIDE(TotalProfit, TotalSales)
```

2. Calculate average order value using variables:

```
Average Order Value =

VAR TotalOrders = COUNT(Orders[OrderID])

VAR TotalSales = SUM(Sales[Amount])

RETURN DIVIDE(TotalSales, TotalOrders)
```

3. Calculate cost per unit using variables:

```
Cost Per Unit =

VAR TotalCost = SUM(Cost[Amount])

VAR TotalUnits = SUM(Sales[Quantity])
```

```
RETURN DIVIDE(TotalCost, TotalUnits)
```

---

7. Combining Multiple DAX Functions for Complex Calculations

Combine functions for advanced calculations.

Examples:

1. Calculate year-over-year growth for sales in regions with sales > 100,000:

```
YoY Growth =

VAR SalesLastYear = CALCULATE(SUM(Sales[Amount]),
SAMEPERIODLASTYEAR(Dates[Date]))

VAR SalesThisYear = SUM(Sales[Amount])

RETURN IF(SalesThisYear > 100000, DIVIDE(SalesThisYear - SalesLastYear,
SalesLastYear), BLANK())
```

2. Calculate profit growth for products with profit > 50,000:

```
Profit Growth =

VAR ProfitLastYear = CALCULATE(SUM(Sales[Profit]),
SAMEPERIODLASTYEAR(Dates[Date]))

VAR ProfitThisYear = SUM(Sales[Profit])

RETURN IF(ProfitThisYear > 50000, DIVIDE(ProfitThisYear - ProfitLastYear,
ProfitLastYear), BLANK())
```

3. Calculate expense reduction for departments with expenses > 20,000:

```
Expense Reduction =
```

```
VAR ExpensesLastYear = CALCULATE(SUM(Expenses[Amount]),
SAMEPERIODLASTYEAR(Dates[Date]))

VAR ExpensesThisYear = SUM(Expenses[Amount])

RETURN IF(ExpensesThisYear > 20000, DIVIDE(ExpensesLastYear -
ExpensesThisYear, ExpensesLastYear), BLANK())
```

Hands-On: Creating Advanced KPIs and Measures

Create KPIs showing year-to-date sales, sales from the same period last year, and percentage growth.

Example:

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


Sales Last Year = CALCULATE(SUM(Sales[Amount]),
SAMEPERIODLASTYEAR(Dates[Date]))


YoY Growth = DIVIDE([YTD Sales] - [Sales Last Year], [Sales Last Year])
```

Conclusion

You've completed the Intermediate DAX Functions Training. Practice creating your own measures and KPIs using these functions to enhance your Power BI reports. Happy analyzing!