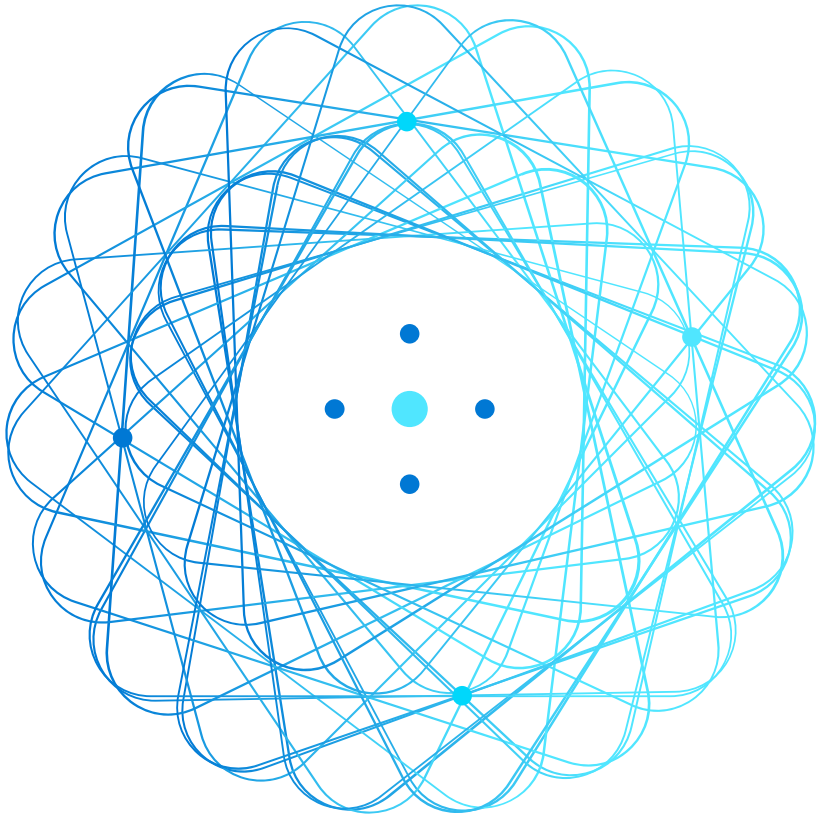
Microsoft Azure



Module 3:
Using Joins and Unions

Module Agenda

Using Logical Statement

Using Joins, Unions

Lesson 1: Using Logical Statement

Logical Statement

- The CASE function goes through conditions, and once a condition is True, this function will stop and return the result (and not go through the rest).
- If none of the specified conditions evaluates to true, this function returns the value in the ELSE clause
- If there is no ELSE clause and no conditions evaluates to true, this function returns NULL

```
CASE [expression]
    WHEN condition_1 THEN value_1
    WHEN condition_2 THEN value_2
    ...
    WHEN condition_N THEN value_N
    ELSE value_0
END
```

Another method: Using logical function

```
IIF( boolean_expression, true_value, false_value )
```

Logical Statement

Example. From DimProduct, retrieve EnglishProductName and generate a new column named EnglishProduct_type based on the following rules: if EnglishProductName contains “Mountain” then assign value “Mountain”, if EnglishProductName contains “Road” then assign value “Road”, else assign value “Other”

```
SELECT EnglishProductName
,CASE
WHEN EnglishProductName like N'%Mountain%' THEN 'Mountain'
WHEN EnglishProductName like N'%Road%' THEN 'Road'
ELSE 'OTHER'
END  as EnglishProduct_type
FROM DimProduct
```

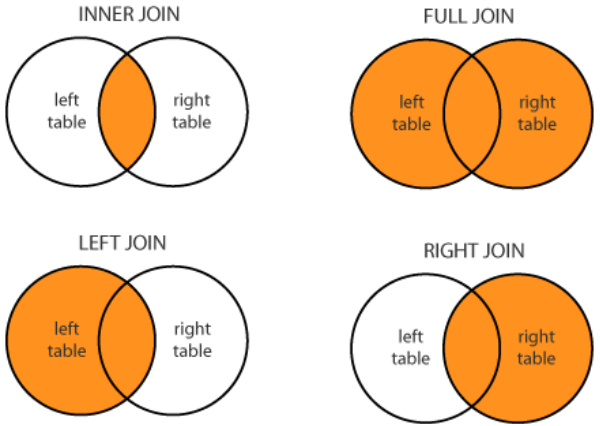
Lesson 2: Using Joins, Unions

Join Concepts

Combine rows from multiple tables by specifying matching criteria

- Usually based on primary key – foreign key relationships
- For example, return rows that combine data from the **Employee** and **SalesOrder** tables by matching the **Employee.EmployeeID** primary key to the **SalesOrder.EmployeeID** foreign key
- Joins type:
  - **(INNER) JOIN:** Returns records that have matching values in both tables
  - **LEFT (OUTER) JOIN:** Return all records from the left table, and the matched records from the right table
  - **RIGHT (OUTER) JOIN:** Return all records from the right table, and the matched records from the left table
  - **FULL (OUTER) JOIN: Return all records when there is a match in either left or right table**

It can help to think of the tables as sets in a Venn diagram



Join Concepts - INNER JOIN

INNER JOIN selects records that have matching values in both tables

```
SELECT column1, column2, column3
FROM table1
[INNER] JOIN table2 ON table1.key_column = table2.key_column
```

Try

```
SELECT Sales.ProductKey, Product.ProductKey, SalesAmount, OrderDate
FROM dbo.FactInternetSales as Sales
JOIN dbo.DimProduct  as Product
on Product.ProductKey = Sales.ProductKey
```

Join Concepts - LEFT JOIN

LEFT JOIN returns all records from the left table (table1), and the matched records from the right table (table2). The result is NULL from the right side, if there is no match.

```
SELECT column1, column2, column3
FROM table1
LEFT JOIN table2 ON table1.key_column = table2.key_column
```

Try

```
SELECT Sales.ProductKey,
Product.ProductKey,
SalesAmount,
OrderDate
FROM dbo.FactInternetSales as Sales
LEFT JOIN dbo.DimProduct  as Product
on Product.ProductKey = Sales.ProductKey
Order By OrderDate, Sales.ProductKey
```

Join Concepts - RIGHT JOIN

RIGHT JOIN returns all records from the right table (table 2), and the matched records from the left table (table 1). The result is NULL from the left side, if there is no match.

```
SELECT column1, column2, column3
FROM table1
RIGHT JOIN table2 ON table1.key_column = table2.key_column
```

Try

```
SELECT Sales.ProductKey,
Product.ProductKey,
SalesAmount,
OrderDate
FROM dbo.FactInternetSales as Sales
RIGHT JOIN dbo.DimProduct  as Product
on Product.ProductKey = Sales.ProductKey
Order By OrderDate, Sales.ProductKey
```

Join Concepts – FULL OUTER JOIN

FULL OUTER JOIN returns all records when there is a match in either left (table1) or right (table2) table records.

```
SELECT column1, column2, column3
FROM table1
FULL [OUTER] JOIN table2 ON table1.key_column = table2.key_column
```

Try

```
SELECT Sales.ProductKey, Product.ProductKey, SalesAmount, OrderDate
FROM dbo.FactInternetSales as Sales
FULL OUTER JOIN dbo.DimProduct as Product on Product.ProductKey = Sales.ProductKey
Order By OrderDate, Sales.ProductKey
```

Practise

Ex1 (CASE WHEN)
From DimProduct, retrieve ProductKey, ListPrice and generate a new column named ProductSegmentation based on the following rules: if ListPrice is greater than 2000 then assign value “Premium”, ListPrice from 1000 to 2000 then assign value “Normal”, ListPrice is lower than 1000 then assign value “Cheap”, in case ListPrice is NULL, assign value “Undefined”

Ex2 (JOIN)
From DimCustomer, DimGeography
Retrieve CustomerKey, CustomerFullName (based on FirstName, MiddleName, LastName) and their EnglishCountryRegionName, StateProvinceName

Union operator

- **UNION** operator is used to combine the result-set of two or more SELECT statements and returns only distinct values by default
  - Each SELECT statement within UNION must have the same number of columns
  - The columns must also have similar data types
  - The columns in each SELECT statement must also be in the same order

```
SELECT column1, column2, column3 FROM table1
UNION
SELECT column1, column2, column3 FROM table2
```

- Try

```
SELECT [OrderDate], [SalesAmount]
FROM [FactInternetSales]
UNION
SELECT [OrderDate], [SalesAmount]
FROM [FactResellerSales]
```

! Pay attention to the number of records.

Union operator

- **UNION ALL** operator is used to combine the result-set of two or more SELECT statements and allows duplicate values.
  - Each SELECT statement within UNION ALL must have the same number of columns
  - The columns must also have similar data types
  - The columns in each SELECT statement must also be in the same order

```
SELECT column1, column2, column3 FROM table1
UNION ALL
SELECT column1, column2, column3 FROM table2
```

- Try:

```
SELECT [OrderDate], [SalesAmount]
FROM [FactInternetSales]
UNION ALL
SELECT [OrderDate], [SalesAmount]
FROM [FactResellerSales]
```

! Pay attention to the number of records.