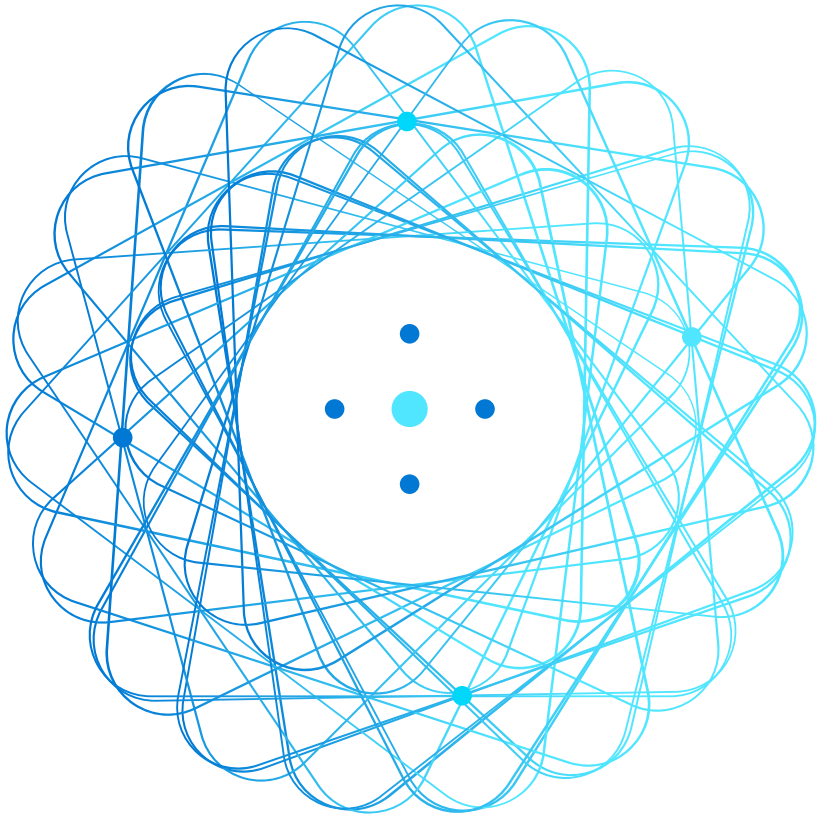
Microsoft Azure

Module 5: GROUP BY and Window functions



Module Agenda

Aggregate functions and GROUP BY

OVER Clause and Window functions

Lesson 1: Aggregate functions and GROUP BY

Aggregate functions

An aggregate function performs a calculation on a set of values, and returns a single value. Except for COUNT, aggregate functions ignore null values. Aggregate functions are often used with the GROUP BY clause of the SELECT statement

- **MIN() Syntax**

```
-- Returns the smallest value of the selected column  --
SELECT MIN(column_name)
FROM table_name
```

- **MAX() Syntax**

```
-- Returns the largest value of the selected column
SELECT MAX(column_name)
FROM table_name
```

- **Try:**

```
SELECT MIN(SalesAmount) as Lowest_sales,  MAX(SalesAmount) as highest_sales
FROM FactInternetSales
```

Aggregate functions

- **COUNT() Syntax**

```sql
-- Returns the number of rows that matches a specified criteria
SELECT COUNT([DISTINCT] columns_name)
FROM table_name
```

- **AVG() Syntax**

```sql
-- Returns the average value of a numeric column
SELECT AVG(column_name)
FROM table_name
```

- **SUM() Syntax**

```sql
-- Returns the total sum of a numeric column
SELECT SUM(column_name)
FROM table_name
```

Aggregate functions – Practice

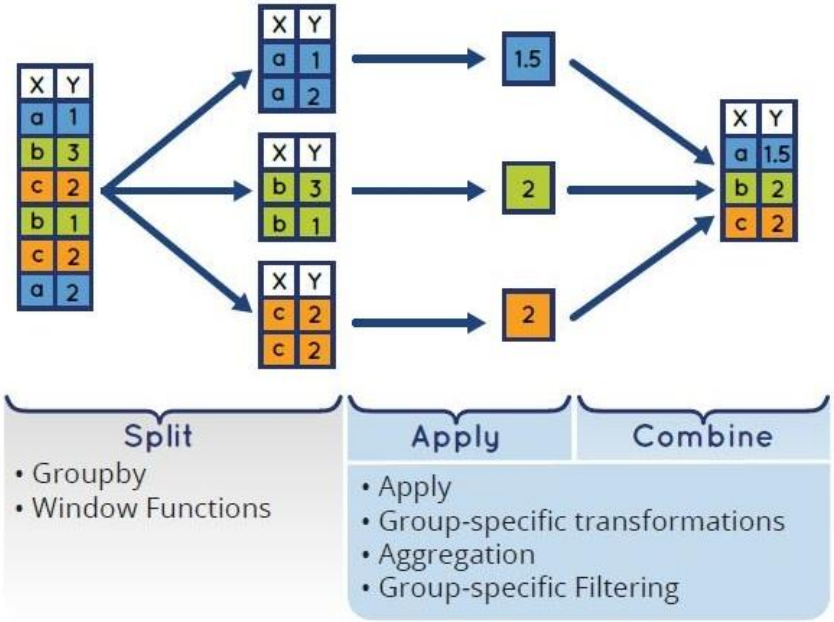
Exercise 1: Write a query to determine the number of products in the DimProduct table.

Exercise 2: Write a query using the DimProduct table that displays the minimum, maximum, and average ListPrice of all Product

Exercise 3: Write a query to determine the number of products in the FactInternetSales table of all time.

Grouping with GROUP BY

- GROUP BY creates groups for output rows, according
  to unique combination of values specified in the GROUP BY clause
- GROUP BY calculates a summary value for aggregate functions in subsequent phases
- Detail rows are not available after GROUP BY clause is processed




```
SELECT OrderDate
, MAX(SalesAmount) as highest_sales
, SUM(SalesAmount) as total_sales
FROM [FactInternetSales]
WHERE OrderDate >= '2011-01-01'
GROUP BY OrderDate
```

GROUP BY – Practice

Exercise 1: Write a query that displays the count of orders placed by each year for each customer using the FactInternetsales table

Exercise 2: Write a query using DimProduct and DimProductSubcategory tables to display number of product in each SubcategoryName

Filtering Groups with HAVING

HAVING clause provides a search condition that each group must satisfy

WHERE clause is processed before GROUP BY, HAVING clause is processed after GROUP BY

```
SELECT column1, Aggregate Functions(column2)
FROM table_name
WHERE condition
GROUP BY column1
HAVING Aggregate Functions(column2) condition
```

```
SELECT MAX(SalesAmount) as highest_sales
, SUM(SalesAmount) as total_sales
, OrderDate
FROM [FactInternetSales]
WHERE OrderDate >= '2011-01-01'
GROUP BY OrderDate
HAVING SUM(SalesAmount) > 10000
```

HAVING – Practice

Exercise : The company is about to run a loyalty scheme to retain customers having total value of orders greater than 5000 USD per year. From FactInternetSales table, retrieve the list of qualified customers and the corresponding year.

Lesson 2: Window Functions

OVER Clause and Window Functions

- **OVER Clause determines the partitioning and ordering of a rowset before the associated window function is applied**
- **Window Functions calculate an aggregate value based on a group of rows and return multiple rows for each group.**
- **ROW_NUMBER and RANK are similar. ROW_NUMBER numbers all rows sequentially (for example 1, 2, 3, 4, 5). RANK provides the same numeric value for ties (for example 1, 2, 2, 4, 5).**
- **Syntax:**

```
ROW_NUMBER ()
    OVER ( [ PARTITION BY value_expression , ... [ n ] ] order_by_clause )
```

```
RANK() OVER (
    [PARTITION BY partition_expression, ... ]
    ORDER BY sort_expression [ASC | DESC], ...)
```

- **First, the PARTITION BY clause divides the rows of the result set partitions to which the function is applied.**
- **Second, the ORDER BY clause specifies the logical sort order of the rows in each a partition to which the function is applied**