Lecture Notes – Case_When-Select

Section 1: Lecture Summary

The lecture introduces the **Case When** syntax in SQL for creating conditional expressions similar to if-then-else statements within queries. It explains the basic structure starting with CASE, followed by multiple WHEN conditions with results, an optional ELSE for default results, and ending with END. The focus is on using **Case When** in the SELECT clause to generate computed columns that classify data based on conditions, with examples demonstrating classification of employees by salary ranges and products by stock levels. It notes that **Case When** can also be used in WHERE, HAVING, ORDER BY clauses, and DML commands like UPDATE and DELETE for filtering, grouping, custom sorting, and conditional operations.

Section 2: Key Concepts and Explanations

Case When functions as a conditional block inside SQL queries, evaluating conditions sequentially and returning the first matching result or the ELSE value if no conditions match. The syntax is: CASE WHEN condition1 THEN result1 WHEN condition2 THEN result2 ... ELSE default END, treated as a single column when used in SELECT, requiring a comma before it like other columns. In SELECT, it creates **computed columns** for meaningful categorization. Usage extends to WHERE for complex row filtering, HAVING for group filtering with aggregates, ORDER BY for **custom sorting logic**, and DML for conditional updates or deletes. An alias can be assigned to the CASE block for clarity, and commas are essential as it acts as an additional column.

Section 3: Example Code and Use Cases

Adapted to eCommerceDB schema:

Classify customers by join year as **new** (2025+), **recent** (2023-2024), or **established**:

```
SELECT CustomerID, FirstName, LastName, JoinDate,
CASE
  WHEN YEAR(JoinDate) >= 2025 THEN 'new'
  WHEN YEAR(JoinDate) >= 2023 THEN 'recent'
  ELSE 'established'
END AS customer_category
FROM Customers;
```

Classify products by **Stock** as **out of stock** (0), **low stock** (<=30), **medium stock** (<=100), or **high stock**:

```
SELECT ProductID, ProductName, Stock,
CASE
  WHEN Stock = 0 THEN 'out of stock'
  WHEN Stock <= 30 THEN 'low stock'
  WHEN Stock <= 100 THEN 'medium stock'
  ELSE 'high stock'
END AS stock_level
FROM Products;
```

Classify orders by **TotalAmount** as **high value** (>=1000), **medium value** (>=500), or **low value**:

```
SELECT OrderID, CustomerID, TotalAmount,
CASE
  WHEN TotalAmount >= 1000 THEN 'high value'
  WHEN TotalAmount >= 500 THEN 'medium value'
  ELSE 'low value'
END AS order_category
FROM Orders;
```

Section 4: Key Takeaways

Case When enables conditional logic in SQL for computed columns, filtering, sorting, and DML operations. Use it in SELECT with comma separation and optional alias for categorization. Conditions evaluate top-to-bottom, returning the first match or ELSE default. Commas are critical as it functions as a column. Practice across SELECT, WHERE, HAVING, ORDER BY for complex scenarios.