Lecture Notes – EXCEPT

Section 1: Lecture Summary

The lecture covers the **except** set operation in SQL, which performs a difference operation by subtracting common rows from the first query's result using the second query's result. Two examples from the eCommerceDB demonstrate its use: finding orders with missing payments by subtracting paid orders from all orders, and identifying cities with customers but no delivered orders by subtracting cities with delivered orders from all customer cities.

Section 2: Key Concepts and Explanations

Except syntax involves a first SELECT query followed by EXCEPT and a second SELECT query with matching column count and compatible data types. It returns rows from the first query that do not appear in the second query. For meaningful results, select relevant columns like OrderID, CustomerID, and TotalAmount. Use joins in the second query to align columns, such as natural join between Orders and Payments on OrderID. Apply **DISTINCT** when needed to handle duplicates, as in city lists. In eCommerceDB, Orders has 20 rows and Payments has 19, revealing one missing payment via except. For cities, join Customers and Orders on CustomerID, filter OrderStatus = 'DELIVERED', and subtract from all customer cities to find Bengaluru with no delivered orders.

Section 3: Example Code and Use Cases

Query for orders with missing payments:

```
SELECT OrderID, CustomerID, TotalAmount
FROM Orders
EXCEPT
SELECT o.OrderID, o.CustomerID, o.TotalAmount
FROM Orders o
NATURAL JOIN Payments;
```

Result: OrderID 111006, CustomerID 6, TotalAmount 1600 (canceled order with no payment).

Query for cities with customers but no delivered orders:

```
SELECT DISTINCT City
FROM Customers
```

```
EXCEPT
SELECT DISTINCT c.City
FROM Customers c
NATURAL JOIN Orders o
WHERE o.OrderStatus = 'DELIVERED';
```

Result: Bengaluru (customer 4 has shipped/pending orders, none delivered).

Section 4: Key Takeaways

Except subtracts matching rows from the first query using the second, requiring identical column structures. Join tables in the second query to match columns like CustomerID or TotalAmount. Use **DISTINCT** for unique values such as cities. Understand data patterns, like 20 orders vs 19 payments, to verify results. Practice on eCommerceDB to confirm outputs like missing payment for order 111006 and undelivered city Bengaluru.