**OUBT Week-1 – Day 4: SQL & Data Modeling basics**

**NAME: LIKHITH SASANK UPPALAPATI VENKATA**

**Overview:**  
Day 4 focused on understanding database fundamentals and data modeling concepts. I learned about SQL basics, relational vs. non-relational databases, ACID properties, normalization, and schema evolution. I also studied how to design a proper data model using entities, attributes, and relationships, and explored star schema and Slowly Changing Dimensions (SCD Types 1 & 2). Along with this, I practiced SQL operations like joins, aggregations, and window functions on sample data to strengthen query skills before working with AWS RDS.

**Hands-On Practice:**

* Created sample tables and inserted data for customers, products, and orders.
* Practiced SQL operations such as:
  + Basic filtering using WHERE.
  + Aggregations with COUNT(), SUM(), and AVG().
  + Joins between multiple tables to fetch related information.
  + Grouping and ordering results.
  + Used the RANK() window function to rank customers by total spending.
* Designed the schema in a **star-schema pattern** with Orders and OrderDetails as fact tables and Customers, Products as dimensions.

A screenshot of a computer

AI-generated content may be incorrect.

A computer screen shot of a computer code

AI-generated content may be incorrect.

**SQL – QUERIES:**

SELECT \* FROM Customers WHERE Country = 'USA';

**-- List all products under $500**

SELECT ProductName, Price FROM Products WHERE Price < 500;

**--  Count total number of orders**

SELECT COUNT(\*) AS TotalOrders FROM Orders;

**--  Get all orders for customer 'Likhith'**

SELECT o.OrderID, o.OrderDate, o.TotalAmount FROM Orders o JOIN Customers c ON o.CustomerID = c.CustomerID

WHERE c.Name = 'Likhith';

**--  Total spent by each customer**

SELECT c.Name, SUM(o.TotalAmount) AS TotalSpent

FROM Customers c

JOIN Orders o ON c.CustomerID = o.CustomerID

GROUP BY c.Name;

**--  List all orders with product names**

SELECT o.OrderID, c.Name AS CustomerName, p.ProductName, od.Quantity

FROM Orders o

JOIN Customers c ON o.CustomerID = c.CustomerID

JOIN OrderDetails od ON o.OrderID = od.OrderID

JOIN Products p ON od.ProductID = p.ProductID;

**--  Find top 2 customers by spending**

SELECT c.Name, SUM(o.TotalAmount) AS TotalSpent

FROM Customers c

JOIN Orders o ON c.CustomerID = o.CustomerID

GROUP BY c.Name

ORDER BY TotalSpent DESC

LIMIT 2;

**--  Find products never ordered**

SELECT p.ProductName

FROM Products p

LEFT JOIN OrderDetails od ON p.ProductID = od.ProductID

WHERE od.ProductID IS NULL;

**--  Average order amount per country**

SELECT c.Country, AVG(o.TotalAmount) AS AvgOrderAmount

FROM Customers c

JOIN Orders o ON c.CustomerID = o.CustomerID

GROUP BY c.Country;

**SQL Query results Screenshots:**

**Query 1: Fetch all customers from the USA**

**A screenshot of a computer

AI-generated content may be incorrect.**

**Query 2: List all products priced under $500  
A screenshot of a computer

AI-generated content may be incorrect.**

**Query 3: Count total number of orders**

**A screenshot of a computer

AI-generated content may be incorrect.**

**Query 4: Retrieve all orders made by customer “Likhith”**

**A screenshot of a computer

AI-generated content may be incorrect.**

**Query 5: Calculate total amount spent by each customer**

**A screenshot of a computer

AI-generated content may be incorrect.**

**Query 6: Display all orders with product names and quantities  
A screenshot of a computer

AI-generated content may be incorrect.**

**Query 7: Find top 2 customers based on total spending**

**A screenshot of a computer

AI-generated content may be incorrect.**

**Query 8: Identify products that were never ordered  
A screenshot of a computer

AI-generated content may be incorrect.**

**Query 9: Find average order amount by country**

**A screen shot of a computer

AI-generated content may be incorrect.**

**A screenshot of a computer

AI-generated content may be incorrect.**

**Key Learnings:**

* Learned how relational databases store and manage structured data.
* Understood **ACID properties**, normalization, and schema design concepts.
* Practiced real SQL operations – joins, aggregations, and ranking queries.
* Learned how **star schema modeling** supports analytical queries.
* Improved understanding of how to represent one-to-many and many-to-many relationships.

**Summary:**

Overall, Day 4 helped me connect theoretical database concepts with real SQL practice. Building the e-commerce schema gave me a clear idea of how relational data is structured, queried, and optimized. This day built a strong base for moving into AWS RDS and advanced data modeling in the next steps.