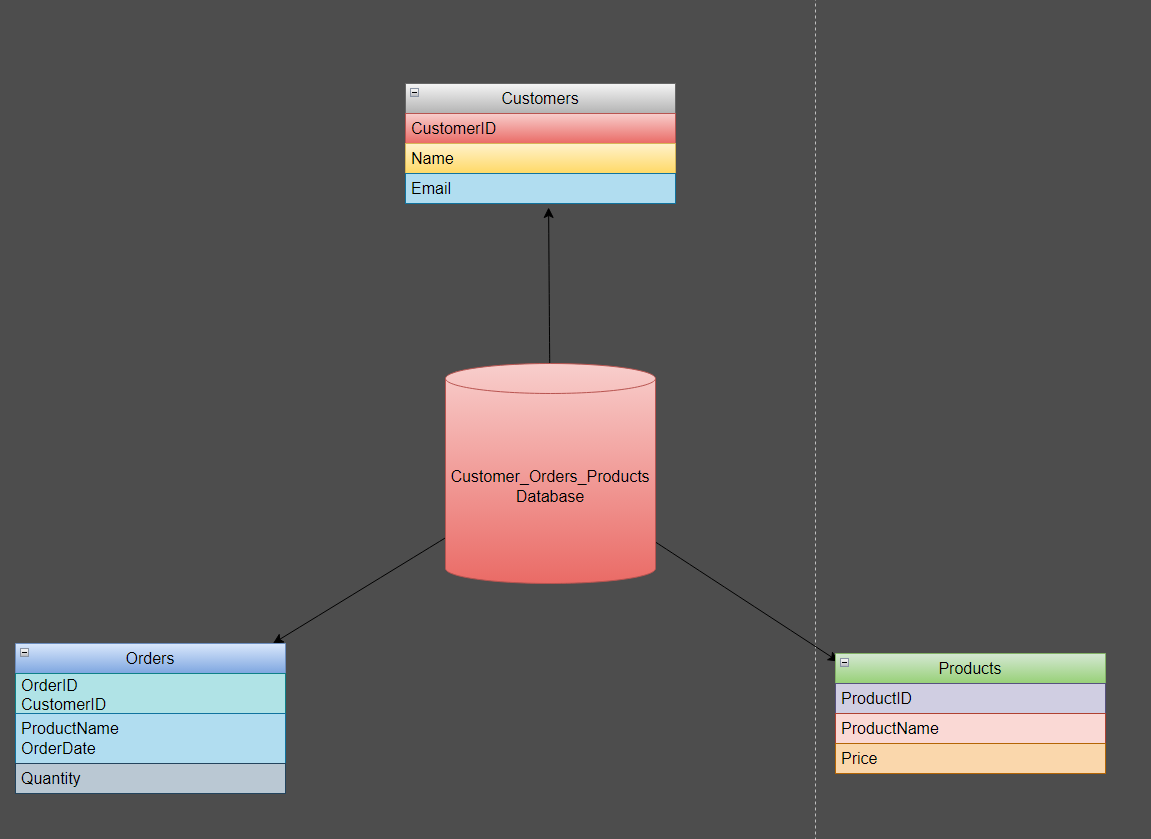


**Capstone Project** **(Customer\_Orders\_Products Database)**



Let’s Create One Database name as Customers\_Orders\_Products

Create three tables called as

Customers

Orders

Products

Insert atleast 10 Records in it

Records are

CREATE TABLE Customers (

CustomerID INT PRIMARY KEY,

Name VARCHAR(50),

Email VARCHAR(100)

);

INSERT INTO Customers (CustomerID, Name, Email)

VALUES

(1, 'John Doe', 'johndoe@example.com'),

(2, 'Jane Smith', 'janesmith@example.com'),

(3, 'Robert Johnson', 'robertjohnson@example.com'),

(4, 'Emily Brown', 'emilybrown@example.com'),

(5, 'Michael Davis', 'michaeldavis@example.com'),

(6, 'Sarah Wilson', 'sarahwilson@example.com'),

(7, 'David Thompson', 'davidthompson@example.com'),

(8, 'Jessica Lee', 'jessicalee@example.com'),

(9, 'William Turner', 'williamturner@example.com'),

(10, 'Olivia Martinez', 'oliviamartinez@example.com'),

(11, 'Alex Brown', 'alexbrown@example.com'),

(12, 'Sophie Turner', 'sophieturner@example.com'),

(13, 'Charlie Miller', 'charliemiller@example.com'),

(14, 'Ava Wilson', 'avawilson@example.com'),

(15, 'Daniel Harris', 'danielharris@example.com'),

(16, 'Emma Davis', 'emmadavis@example.com'),

(17, 'James White', 'jameswhite@example.com'),

(18, 'Lily Martin', 'lilymartin@example.com'),

(19, 'Benjamin Johnson', 'benjaminjohnson@example.com'),

(20, 'Zoe Anderson', 'zoeanderson@example.com'),

(21, 'Jackson Moore', 'jacksonmoore@example.com'),

(22, 'Victoria Lewis', 'victorialewis@example.com'),

(23, 'Ethan Garcia', 'ethangarcia@example.com'),

(24, 'Grace Taylor', 'gracetaylor@example.com'),

(25, 'Christopher Hall', 'christopherhall@example.com');

CREATE TABLE Orders (

OrderID INT PRIMARY KEY,

CustomerID INT,

ProductName VARCHAR(50),

OrderDate DATE,

Quantity INT

);

INSERT INTO Orders (OrderID, CustomerID, ProductName, OrderDate, Quantity)

VALUES

(1, 1, 'Product A', '2023-07-01', 5),

(2, 2, 'Product B', '2023-07-02', 3),

(3, 3, 'Product C', '2023-07-03', 2),

(4, 4, 'Product A', '2023-07-04', 1),

(5, 5, 'Product B', '2023-07-05', 4),

(6, 6, 'Product C', '2023-07-06', 2),

(7, 7, 'Product A', '2023-07-07', 3),

(8, 8, 'Product B', '2023-07-08', 2),

(9, 9, 'Product C', '2023-07-09', 5),

(10, 10, 'Product A', '2023-07-10', 1),

(11, 11, 'Product D', '2023-07-11', 2),

(12, 12, 'Product E', '2023-07-12', 3),

(13, 13, 'Product F', '2023-07-13', 4),

(14, 14, 'Product G', '2023-07-14', 1),

(15, 15, 'Product H', '2023-07-15', 5),

(16, 16, 'Product I', '2023-07-16', 2),

(17, 17, 'Product J', '2023-07-17', 3),

(18, 18, 'Product A', '2023-07-18', 4),

(19, 19, 'Product B', '2023-07-19', 2),

(20, 20, 'Product C', '2023-07-20', 1),

(21, 21, 'Product D', '2023-07-21', 3),

(22, 22, 'Product E', '2023-07-22', 5),

(23, 23, 'Product F', '2023-07-23', 2),

(24, 24, 'Product G', '2023-07-24', 1),

(25, 25, 'Product H', '2023-07-25', 4);

CREATE TABLE Products (

ProductID INT PRIMARY KEY,

ProductName VARCHAR(50),

Price DECIMAL(10, 2)

);

INSERT INTO Products (ProductID, ProductName, Price)

VALUES

(1, 'Product A', 10.99),

(2, 'Product B', 8.99),

(3, 'Product C', 5.99),

(4, 'Product D', 12.99),

(5, 'Product E', 7.99),

(6, 'Product F', 6.99),

(7, 'Product G', 9.99),

(8, 'Product H', 11.99),

(9, 'Product I', 14.99),

(10, 'Product J', 4.99),

(11, 'Product K', 15.99),

(12, 'Product L', 9.99),

(13, 'Product M', 6.49),

(14, 'Product N', 11.49),

(15, 'Product O', 8.99),

(16, 'Product P', 7.49),

(17, 'Product Q', 10.99),

(18, 'Product R', 13.99),

(19, 'Product S', 16.49),

(20, 'Product T', 5.49),

(21, 'Product U', 14.99),

(22, 'Product V', 8.49),

(23, 'Product W', 9.99),

(24, 'Product X', 12.49),

(25, 'Product Y', 4.99);

After Creating tables Solve Following tasks:

**Task 1 :-**

1. Write a query to retrieve all records from the Customers table..

Answer- Select\* from Customers

1. Write a query to retrieve the names and email addresses of customers whose names start with 'J'.

Answer- SELECT Name, Email

FROM Customers

WHERE Name LIKE 'J%'

1. Write a query to retrieve the order details (OrderID, ProductName, Quantity) for all orders..

Answer- SELECT

o.OrderID,

o.ProductName,

o.Quantity

FROM

Orders o

JOIN

Products p ON o.ProductName = p.ProductName

1. Write a query to calculate the total quantity of products ordered.

Answer- SELECT SUM(Quantity) AS TotalQuantity

FROM Orders

1. Write a query to retrieve the names of customers who have placed an order.

Answer- SELECT DISTINCT c.Name

FROM Customers c

JOIN Orders o ON c.CustomerID = o.CustomerID

1. Write a query to retrieve the products with a price greater than $10.00.

Answer- SELECT \*

FROM Products

WHERE Price > 10.00

1. Write a query to retrieve the customer name and order date for all orders placed on or after '2023-07-05'.

Answer- SELECT

c.Name AS CustomerName,

o.OrderDate

FROM

Customers c

JOIN

Orders o ON c.CustomerID = o.CustomerID

WHERE

o.OrderDate >= '2023-07-05'

1. Write a query to calculate the average price of all products.

Answer- SELECT AVG(Price) AS AveragePrice

FROM Products

1. Write a query to retrieve the customer names along with the total quantity of products they have ordered.

Answer- SELECT

c.Name AS CustomerName,

SUM(o.Quantity) AS TotalQuantity

FROM

Customers

JOIN

Orders o ON c.CustomerID = o.CustomerID

GROUP BY

c.Name

1. Write a query to retrieve the products that have not been ordered.

SELECT

Products.\*

FROM

Products p

LEFT JOIN

Orders ON Products.ProductName = Orders.ProductName

WHERE

Orders.OrderID IS NULL

Task 2 :-

1. Write a query to retrieve the top 5 customers who have placed the highest total quantity of orders.

Answer- SELECT

Customers.Name AS CustomerName,

SUM(o.Quantity) AS TotalQuantity

FROM

Customers

JOIN

Orders ON Customers .CustomerID = Orders .CustomerID

GROUP BY

Customers .CustomerID, Customer.Name

ORDER BY

TotalQuantity DESC

LIMIT 5

1. Write a query to calculate the average price of products for each product category.

Answer- SELECT

c.CategoryName,

AVG(p.Price) AS AveragePrice

FROM

Products p

JOIN

Categories ON p.CategoryID = c.CategoryID

GROUP BY

c.CategoryName

1. Write a query to retrieve the customers who have not placed any orders.

Answer- SELECT

customers.\*

FROM

Customers

LEFT JOIN

Orders ON c.CustomerID = o.CustomerID

WHERE

o.OrderID IS NULL

1. Write a query to retrieve the order details (OrderID, ProductName, Quantity) for orders placed by customers whose names start with 'M'.

Answer- SELECT

o.OrderID,

o.ProductName,

o.Quantity

FROM

Orders o

JOIN

Customers c ON o.CustomerID = c.CustomerID

WHERE

c.Name LIKE 'M%'

1. Write a query to calculate the total revenue generated from all orders.

Answer- SELECT SUM(o.Quantity \* p.Price) AS TotalRevenue

FROM Orders o

JOIN Products p ON o.ProductName = p.ProductName

1. Write a query to retrieve the customer names along with the total revenue generated from their orders.

Answer- SELECT

c.Name AS CustomerName,

SUM(o.Quantity \* p.Price) AS TotalRevenue

FROM

Customers c

JOIN

Orders o ON c.CustomerID = o.CustomerID

JOIN

Products p ON o.ProductName = p.ProductName

GROUP BY

c.Name

1. Write a query to retrieve the customers who have placed at least one order for each product category.

Answer- SELECT

c.Name AS CustomerName

FROM

Customers c

JOIN

Orders o ON c.CustomerID = o.CustomerID

JOIN

Products p ON o.ProductName = p.ProductName

GROUP BY

c.CustomerID, c.Name

HAVING

COUNT(DISTINCT p.CategoryID) = (SELECT COUNT(DISTINCT CategoryID) FROM Products)

1. Write a query to retrieve the customers who have placed orders on consecutive days.

Answer- SELECT

c.Name AS CustomerName

FROM

Customers c

JOIN

Orders o1 ON c.CustomerID = o1.CustomerID

JOIN

Orders o2 ON c.CustomerID = o2.CustomerID

AND o1.OrderDate = DATE\_ADD(o2.OrderDate, INTERVAL -1 DAY)

GROUP BY

c.CustomerID, c.Name

1. Write a query to retrieve the top 3 products with the highest average quantity ordered.

Answer- SELECT

ProductName,

AVG(Quantity) AS AverageQuantity

FROM

Orders

GROUP BY

ProductName

ORDER BY

AverageQuantity DESC

LIMIT 3

1. Write a query to calculate the percentage of orders that have a quantity greater than the average quantity.

Answer- SELECT

(COUNT(CASE WHEN Quantity > avg\_quantity THEN 1 END) / COUNT(\*)) \* 100 AS Percentage

FROM

Orders,

(SELECT AVG(Quantity) AS avg\_quantity FROM Orders) AS subquery

**Task 3:-**

1. Write a query to retrieve the customers who have placed orders for all products.

Answer- SELECT

c.Name AS CustomerName

FROM

Customers c

JOIN

Orders o ON c.CustomerID = o.CustomerID

GROUP BY

c.CustomerID, c.Name

HAVING

COUNT(DISTINCT o.ProductName) = (SELECT COUNT(DISTINCT ProductName) FROM Products)

1. Write a query to retrieve the products that have been ordered by all customers.

Answer- SELECT

p.ProductName

FROM

Products p

JOIN

Orders o ON p.ProductName = o.ProductName

GROUP BY

p.ProductName

HAVING

COUNT(DISTINCT o.CustomerID) = (SELECT COUNT(DISTINCT CustomerID) FROM Customers)

1. Write a query to calculate the total revenue generated from orders placed in each month.

Answer- SELECT

DATE\_FORMAT(OrderDate, '%Y-%m') AS Month,

SUM(Quantity \* Price) AS TotalRevenue

FROM

Orders

JOIN

Products ON Orders.ProductName = Products.ProductName

GROUP BY

DATE\_FORMAT(OrderDate, '%Y-%m')

1. Write a query to retrieve the products that have been ordered by more than 50% of the customers.

Answer- SELECT

ProductName

FROM

Orders

GROUP BY

ProductName

HAVING

COUNT(DISTINCT CustomerID) > (SELECT COUNT(DISTINCT CustomerID) \* 0.5 FROM Customers)

1. Write a query to retrieve the top 5 customers who have spent the highest amount of money on orders.

Answer- SELECT

c.Name AS CustomerName,

SUM(o.Quantity \* p.Price) AS TotalSpent

FROM

Customers c

JOIN

Orders o ON c.CustomerID = o.CustomerID

JOIN

Products p ON o.ProductName = p.ProductName

GROUP BY

c.CustomerID, c.Name

ORDER BY

TotalSpent DESC

LIMIT 5

1. **Write a query to calculate the running total of order quantities for each customer.**

**Answer-** **SELECT**

**CustomerID,**

**OrderID,**

**OrderDate,**

**Quantity,**

**SUM(Quantity) OVER (PARTITION BY CustomerID ORDER BY OrderDate) AS RunningTotal**

**FROM**

**Orders**

**ORDER BY**

**CustomerID, OrderDate**

1. Write a query to retrieve the top 3 most recent orders for each customer.

Answer- SELECT

CustomerID,

OrderID,

ProductName,

OrderDate

FROM (

SELECT

CustomerID,

OrderID,

ProductName,

OrderDate,

ROW\_NUMBER() OVER (PARTITION BY CustomerID ORDER BY OrderDate DESC) AS row\_num

FROM

Orders

) AS ranked\_orders

WHERE

row\_num <= 3

1. Write a query to calculate the total revenue generated by each customer in the last 30 days.

Answer- SELECT

c.CustomerID,

c.Name AS CustomerName,

SUM(o.Quantity \* p.Price) AS TotalRevenue

FROM

Customers c

JOIN

Orders o ON c.CustomerID = o.CustomerID

JOIN

Products p ON o.ProductName = p.ProductName

WHERE

o.OrderDate >= DATE\_SUB(CURDATE(), INTERVAL 30 DAY)

GROUP BY

c.CustomerID, c.Name

1. Write a query to retrieve the customers who have placed orders for at least two different product categories.

Answer- SELECT

c.CustomerID,

c.Name AS CustomerName

FROM

Customers c

JOIN

Orders o ON c.CustomerID = o.CustomerID

JOIN

Products p ON o.ProductName = p.ProductName

GROUP BY

c.CustomerID, c.Name

HAVING

COUNT(DISTINCT p.CategoryID) >= 2

1. **Write a query to calculate the average revenue per order for each customer.**

**Answer-** **SELECT**

**c.CustomerID,**

**c.Name AS CustomerName,**

**SUM(o.Quantity \* p.Price) / COUNT(o.OrderID) AS AverageRevenuePerOrder**

**FROM**

**Customers c**

**JOIN**

**Orders o ON c.CustomerID = o.CustomerID**

**JOIN**

**Products p ON o.ProductName = p.ProductName**

**GROUP BY**

**c.CustomerID, c.Name**

1. Write a query to retrieve the customers who have placed orders for every month of a specific year.

Answer- SELECT

c.CustomerID,

c.Name AS CustomerName

FROM

Customers c

JOIN

Orders o ON c.CustomerID = o.CustomerID

WHERE

YEAR(o.OrderDate) = 2023 -- Specify the specific year here

GROUP BY

c.CustomerID, c.Name

HAVING

COUNT(DISTINCT MONTH(o.OrderDate)) = 12; -- Assuming orders for all 12 months are require

1. Write a query to retrieve the customers who have placed orders for a specific product in consecutive months.

Anwwer- WITH OrderedOrders AS (

SELECT

\*,

LAG(OrderDate) OVER (PARTITION BY CustomerID, ProductName ORDER BY OrderDate) AS PreviousOrderDate

FROM

Orders

)

SELECT DISTINCT

c.CustomerID,

c.Name AS CustomerName

FROM

Customers c

JOIN

OrderedOrders oo ON c.CustomerID = oo.CustomerID

WHERE

MONTH(oo.OrderDate) - MONTH(oo.PreviousOrderDate) = 1

AND oo.ProductName = 'Specific Product'

1. Write a query to retrieve the products that have been ordered by a specific customer at least twice.

SELECT

o.ProductName

FROM

Orders o

WHERE

o.CustomerID = 1

GROUP BY

o.ProductName

HAVING

COUNT(o.ProductName) >= 2