**Retails Store Database**

1. Create table customers

CREATE TABLE Customers ( customer\_id INT PRIMARY KEY, first\_name VARCHAR(100), last\_name VARCHAR(100), email VARCHAR(100), phone VARCHAR(20), address TEXT, join\_date DATE );

**🡺INSERT DATA**

INSERT INTO Customers VALUES

(1, 'Rahul', 'Sharma', 'rahul.sharma@example.com', '9876543210', 'Mumbai, Maharashtra', '2023-01-15'),

(2, 'Priya', 'Verma', 'priya.verma@example.com', '9876543211', 'Delhi, Delhi', '2023-02-10'),

(3, 'Amit', 'Kumar', 'amit.kumar@example.com', '9876543212', 'Bangalore, Karnataka', '2023-03-05'),

(4, 'Anjali', 'Reddy', 'anjali.reddy@example.com', '9876543213', 'Hyderabad, Telangana', '2023-04-20');

2. Create product table

CREATE TABLE Products ( product\_id INT PRIMARY KEY, product\_name VARCHAR(100), category VARCHAR(50), price DECIMAL(10, 2), stock\_quantity INT );

🡺**INSERT DATA**

INSERT INTO Products VALUES (1, 'Mobile Phone', 'Electronics', 15000.00, 50), (2, 'Laptop', 'Electronics', 50000.00, 30), (3, 'Headphones', 'Accessories', 2000.00, 100), (4, 'Shoes', 'Fashion', 3000.00, 80);

3. Create table orders

CREATE TABLE Orders ( order\_id INT PRIMARY KEY, customer\_id INT, order\_date DATE, total\_amount DECIMAL(10, 2), order\_status VARCHAR(20), FOREIGN KEY (customer\_id) REFERENCES Customers(customer\_id) );

🡺**INSERT DATA**

INSERT INTO Orders VALUES (1, 1, '2023-12-01', 17000.00, 'Shipped'), (2, 2, '2023-12-10', 52000.00, 'Pending'), (3, 3, '2023-12-15', 4000.00, 'Shipped');

4. Create table order details

CREATE TABLE OrderDetails ( order\_detail\_id INT PRIMARY KEY, order\_id INT, product\_id INT, quantity INT, unit\_price DECIMAL(10, 2), FOREIGN KEY (order\_id) REFERENCES Orders(order\_id), FOREIGN KEY (product\_id) REFERENCES Products(product\_id) );

🡺**INSERT DATA**

INSERT INTO OrderDetails VALUES

(1, 1, 1, 1, 15000.00),

(2, 1, 3, 1, 2000.00),

(3, 2, 2, 1, 50000.00),

(4, 2, 3, 1, 2000.00),

(5, 3, 4, 1, 3000.00),

(6, 3, 3, 1, 1000.00);

5. Create payment table

CREATE TABLE Payments ( payment\_id INT PRIMARY KEY, order\_id INT, payment\_date DATE, payment\_amount DECIMAL(10, 2), payment\_method VARCHAR(20), FOREIGN KEY (order\_id) REFERENCES Orders(order\_id) );

🡺**INSERT DATA**

INSERT INTO Payments VALUES

(1, 1, '2023-12-02', 17000.00, 'Credit Card'),

(2, 3, '2023-12-16', 4000.00, 'UPI');

**Quires**

1. **Find the Total Number of Orders for Each Customer**

**🡺** SELECT customers.first\_name,COUNT(orders.order\_id) as total\_number\_order\_by\_cutomer

from orders right JOIN customers on customers.customer\_id = orders.order\_id

GROUP by customers.customer\_id,customers.first\_name

1. **Find the Total Sales Amount for Each Product (Revenue per Product)**

**🡺** /\*Find the Total Sales Amount for Each Product (Revenue per Product)\*/

SELECT products.product\_id,products.product\_name,

SUM(orderdetails.quantity \* orderdetails.unit\_price )AS TOTAL\_SALE FROM products

JOIN orderdetails ON orderdetails.product\_id = products.product\_id

GROUP BY products.product\_id

1. **Find the Most Expensive Product Sold**

**🡺** SELECT products.product\_name ,products.category ,MAX(orderdetails.unit\_price)AS MAXPRICE\_ITEM FROM products JOIN orderdetails ON products.product\_id = orderdetails.product\_id GROUP BY products.product\_id , products.product\_name ORDER BY MAXPRICE\_ITEM DESC

1. **Get the List of Customers Who Have Placed Orders in the Last 30 Days**

**🡺** SELECT customers.first\_name FROM customers JOIN orders on orders.customer\_id = customers.customer\_id

WHERE orders.order\_date >= CURDATE() - INTERVAL 30 DAY;

1. **Calculate the Total Amount Paid by Each Customer**

🡺 SELECT customers.customer\_id, customers.first\_name,customers.last\_name ,SUM(payments.payment\_amount)as total\_poid FROM customers JOIN

orders on customers.customer\_id = orders.customer\_id

JOIN payments on payments.order\_id = orders.order\_id

GROUP by customers.customer\_id

1. **Get the Number of Products Sold by Category**

**🡺** SELECT products.category,SUM(orderdetails.quantity) FROM products JOIN orderdetails ON orderdetails.product\_id = products.product\_id GROUP BY products.category

1. **List All Orders That Are Pending (i.e., Orders that haven't been shipped yet)**

**🡺**

SELECT customers.first\_name,customers.last\_name,orders.order\_date,orders.order\_status FROM orders

JOIN customers on orders.customer\_id = customers.customer\_id

WHERE orders.order\_status = 'pending'

1. **Find the Average Order Value (Total Order Amount / Number of Orders)**

**🡺** SELECT COUNT(orders.order\_id)as number\_of\_orders,AVG(orders.total\_amount) as avg\_orderAmount FROM orders

1. **List the Top 5 Customers Who Have Spent the Most Money**

**🡺** SELECT customers.first\_name,customers.last\_name,max(orders.total\_amount)as max\_amount\_spent FROM orders JOIN customers on orders.customer\_id = customers.customer\_id GROUP by customers.customer\_id

order by max\_amount\_spent DESC limit 5

1. **Find the Products That Have Never Been Sold**

**🡺**

SELECT products.product\_name ,products.category,orderdetails.quantity FROM products LEFT JOIN orderdetails on orderdetails.product\_id = products.product\_id

WHERE orderdetails.product\_id IS NULL