**Internship Task - RDBMS and SQL Task #3**

**The case will be based on a hypothetical business scenario involving a retail store's database. The database contains tables for Customers, Orders, Products, OrderDetails, and Payments.**

create database retail\_store;

CREATE TABLE Customers (

customer\_id INT PRIMARY KEY,

first\_name VARCHAR(100) NOT NULL,

last\_name VARCHAR(100) NOT NULL,

email VARCHAR(100),

phone VARCHAR(20),

address TEXT,

join\_date DATE

);

INSERT INTO Customers (customer\_id, first\_name, last\_name, email, phone, address, join\_date) VALUES

(1, 'Ravi', 'Sharma', 'ravi.sharma@example.com', '9876543210', 'Mumbai, Maharashtra', '2023-01-10'),

(2, 'Priya', 'Verma', 'priya.verma@example.com', '9867543210', 'Delhi, Delhi', '2023-02-15'),

(3, 'Amit', 'Kumar', 'amit.kumar@example.com', '8765432109', 'Bangalore, Karnataka', '2023-03-10'),

(4, 'Sneha', 'Patel', 'sneha.patel@example.com', '7896541230', 'Ahmedabad, Gujarat', '2023-04-05'),

(5, 'Rahul', 'Joshi', 'rahul.joshi@example.com', '9812345678', 'Jaipur, Rajasthan', '2023-05-20'),

(6, 'Anjali', 'Singh', 'anjali.singh@example.com', '8765123456', 'Lucknow, Uttar Pradesh', '2023-06-18'),

(7, 'Kunal', 'Mehta', 'kunal.mehta@example.com', '7894561237', 'Chennai, Tamil Nadu', '2023-07-02'),

(8, 'Divya', 'Reddy', 'divya.reddy@example.com', '7891234568', 'Hyderabad, Telangana', '2023-08-11'),

(9, 'Rajesh', 'Desai', 'rajesh.desai@example.com', '8123456789', 'Pune, Maharashtra', '2023-09-01'),

(10, 'Nisha', 'Yadav', 'nisha.yadav@example.com', '8996541237', 'Kolkata, West Bengal', '2023-10-12'),

(11, 'Arjun', 'Thakur', 'arjun.thakur@example.com', '7654321098', 'Chandigarh, Punjab', '2023-11-03'),

(12, 'Meera', 'Kapoor', 'meera.kapoor@example.com', '7987654321', 'Bhopal, Madhya Pradesh', '2023-12-22'),

(13, 'Varun', 'Gupta', 'varun.gupta@example.com', '8901234567', 'Surat, Gujarat', '2024-01-05'),

(14, 'Pooja', 'Bhatia', 'pooja.bhatia@example.com', '9878901234', 'Nagpur, Maharashtra', '2024-02-14'),

(15, 'Neeraj', 'Chauhan', 'neeraj.chauhan@example.com', '8009876543', 'Cochin, Kerala', '2024-03-10');

CREATE TABLE Products (

product\_id INT PRIMARY KEY,

product\_name VARCHAR(100) NOT NULL,

category VARCHAR(50),

price DECIMAL(10, 2) NOT NULL,

stock\_quantity INT NOT NULL

);

INSERT INTO Products (product\_id, product\_name, category, price, stock\_quantity) VALUES

(1, 'Tata Tea', 'Groceries', 250.00, 50),

(2, 'Parle-G Biscuits', 'Snacks', 50.00, 100),

(3, 'Samsung Galaxy M14', 'Electronics', 14000.00, 20),

(4, 'Lenovo Laptop', 'Electronics', 50000.00, 10),

(5, 'Nike Running Shoes', 'Footwear', 4500.00, 30),

(6, 'Adidas T-shirt', 'Apparel', 1200.00, 25),

(7, 'Prestige Cooker', 'Home Appliances', 2800.00, 15),

(8, 'Dettol Soap', 'Personal Care', 40.00, 200),

(9, 'Colgate Toothpaste', 'Personal Care', 90.00, 150),

(10, 'Classmate Notebook', 'Stationery', 50.00, 500),

(11, 'Amul Butter', 'Dairy', 200.00, 80),

(12, 'Fortune Oil', 'Groceries', 1200.00, 40),

(13, 'Sony Headphones', 'Electronics', 3500.00, 25),

(14, 'Puma Cap', 'Apparel', 600.00, 40),

(15, 'LG Refrigerator', 'Home Appliances', 30000.00, 5);

CREATE TABLE Orders (

order\_id INT PRIMARY KEY,

customer\_id INT,

order\_date DATE NOT NULL,

total\_amount DECIMAL(10, 2),

order\_status VARCHAR(20),

FOREIGN KEY (customer\_id) REFERENCES Customers(customer\_id)

);

INSERT INTO Orders (order\_id, customer\_id, order\_date, total\_amount, order\_status) VALUES

(1, 1, '2023-07-20', 1450.00, 'Shipped'),

(2, 2, '2023-08-10', 4500.00, 'Pending'),

(3, 3, '2023-09-05', 6000.00, 'Cancelled'),

(4, 4, '2023-09-15', 2500.00, 'Shipped'),

(5, 5, '2023-10-01', 8000.00, 'Pending'),

(6, 6, '2023-10-12', 3000.00, 'Shipped'),

(7, 7, '2023-11-03', 5000.00, 'Shipped'),

(8, 8, '2023-11-15', 1500.00, 'Cancelled'),

(9, 9, '2023-12-05', 3500.00, 'Shipped'),

(10, 10, '2024-01-05', 4500.00, 'Shipped'),

(11, 11, '2024-01-10', 3000.00, 'Pending'),

(12, 12, '2024-01-15', 1000.00, 'Cancelled'),

(13, 13, '2024-02-01', 7000.00, 'Shipped'),

(14, 14, '2024-02-10', 2500.00, 'Pending'),

(15, 15, '2024-02-20', 4500.00, 'Shipped');

CREATE TABLE OrderDetails (

order\_detail\_id INT PRIMARY KEY,

order\_id INT,

product\_id INT,

quantity INT NOT NULL,

unit\_price DECIMAL(10, 2) NOT NULL,

FOREIGN KEY (order\_id) REFERENCES Orders(order\_id),

FOREIGN KEY (product\_id) REFERENCES Products(product\_id)

);

INSERT INTO OrderDetails (order\_detail\_id, order\_id, product\_id, quantity, unit\_price) VALUES

(1, 1, 1, 5, 250.00),

(2, 2, 3, 1, 14000.00),

(3, 3, 7, 2, 2800.00),

(4, 4, 8, 10, 40.00),

(5, 5, 5, 2, 4500.00),

(6, 6, 2, 20, 50.00),

(7, 7, 13, 2, 3500.00),

(8, 8, 9, 15, 90.00),

(9, 9, 12, 1, 1200.00),

(10, 10, 14, 1, 600.00),

(11, 11, 11, 10, 200.00),

(12, 12, 4, 1, 50000.00),

(13, 13, 6, 3, 1200.00),

(14, 14, 10, 50, 50.00),

(15, 15, 15, 1, 30000.00);

CREATE TABLE Payments (

payment\_id INT PRIMARY KEY,

order\_id INT,

payment\_date DATE NOT NULL,

payment\_amount DECIMAL(10, 2) NOT NULL,

payment\_method VARCHAR(20),

FOREIGN KEY (order\_id) REFERENCES Orders(order\_id)

);

INSERT INTO Payments (payment\_id, order\_id, payment\_date, payment\_amount, payment\_method) VALUES

(1, 1, '2023-07-21', 1450.00, 'Credit Card'),

(2, 2, '2023-08-11', 4500.00, 'UPI'),

(3, 3, '2023-09-06', 6000.00, 'PayPal'),

(4, 4, '2023-09-16', 2500.00, 'Net Banking'),

(5, 5, '2023-10-02', 8000.00, 'Debit Card'),

(6, 6, '2023-10-13', 3000.00, 'Credit Card'),

(7, 7, '2023-11-04', 5000.00, 'UPI'),

(8, 8, '2023-11-16', 1500.00, 'Cash'),

(9, 9, '2023-12-06', 3500.00, 'UPI'),

(10, 10, '2024-01-06', 4500.00, 'Credit Card'),

(11, 11, '2024-01-11', 3000.00, 'PayPal'),

(12, 12, '2024-01-16', 1000.00, 'Net Banking'),

(13, 13, '2024-02-02', 7000.00, 'UPI'),

(14, 14, '2024-02-11', 2500.00, 'Cash'),

(15, 15, '2024-02-21', 4500.00, 'Credit Card');

-----------------------------------------------------------------------------------------------------------------

**#SQL Queries for the Case Study#**

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

Ans=

select c.customer\_id, c.first\_name, c.last\_name, count(order\_id) as taotal\_orders FROM customers as c left join orders as o on c.customer\_id=o.customer\_id GROUP by c.customer\_id, c.first\_name, c.last\_name

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

Ans=

select p.product\_id, p.product\_name, sum(o.quantity\*o.unit\_price)as total\_revenu from products as p left JOIN orderdetails as o on p.product\_id=o.product\_id GROUP by p.product\_id, p.product\_name

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

Ans=

SELECT product\_name, max(price)as most\_expensive FROM products as p LEFT join OrderDetails as o on p.product\_id=o.product\_id where order\_id in(SELECT order\_id FROM orders) GROUP by product\_name order by most\_expensive desc limit 1;

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

Ans=

SELECT c.customer\_id ,c.first\_name ,c.last\_name FROM customers as c LEFT join orders as o on c.customer\_id=o.customer\_id WHERE o.order\_date >= '2024-02-01'

GROUP by c.customer\_id

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

Ans=

select c.customer\_id, c.first\_name, c.last\_name,sum(payment\_amount) as total\_amount FROM customers as c LEFT join orders as o on c.customer\_id=o.customer\_id LEFT join payments as p on o.order\_id=p.order\_id GROUP by c.customer\_id ,c.first\_name, c.last\_name;

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

Ans=

select category, count(product\_id) as numOfProducts FROM products where product\_id in(select product\_id from OrderDetails) GROUP by category

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

Ans=

select order\_id from orders WHERE order\_status='Pending' GROUP by order\_id

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

Ans=

SELECT (sum(total\_amount) /count(order\_id))as Average\_Order\_Value FROM orders

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

Ans=

select c.customer\_id, max(p.payment\_amount)as Most\_money FROM customers as c left join orders as o on c.customer\_id=o.customer\_id left join payments as p on o.order\_id=p.order\_id GROUP by c.customer\_id order by Most\_money desc LIMIT 5

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

Ans=

select p.product\_id FROM products as p LEFT join orderdetails as o on p.product\_id=o.product\_id WHERE order\_id is null GROUP by p.product\_id