**Instructions:**

* Answer all questions using **MySQL**.
* Use appropriate **subqueries**, **joins**, and **aggregate functions** wherever applicable.
* Make sure to use proper **aliasing**, **GROUP BY**, **HAVING**, **DISTINCT**, etc., as needed.
* Data

-- Customers Table

CREATE TABLE Customers (

CustomerID INT PRIMARY KEY,

Name VARCHAR(100),

City VARCHAR(100)

);

-- Orders Table

CREATE TABLE Orders (

OrderID INT PRIMARY KEY,

CustomerID INT,

OrderDate DATE,

Amount DECIMAL(10,2),

FOREIGN KEY (CustomerID) REFERENCES Customers(CustomerID)

);

-- Products Table

CREATE TABLE Products (

ProductID INT PRIMARY KEY,

ProductName VARCHAR(100),

Price DECIMAL(10,2)

);

-- OrderDetails Table

CREATE TABLE OrderDetails (

OrderDetailID INT PRIMARY KEY,

OrderID INT,

ProductID INT,

Quantity INT,

FOREIGN KEY (OrderID) REFERENCES Orders(OrderID),

FOREIGN KEY (ProductID) REFERENCES Products(ProductID)

);

**Part A – Subqueries (20 marks)**

1. Write a query to find customers who have placed orders in **every month** of the current year.
2. Retrieve the names of products that have been ordered **more than the average quantity** across all products.
3. Find customers who have **never ordered a product** priced above ₹1000.
4. List the **top 3 products by total revenue** using a subquery.
5. Find orders that contain **only one product** using a **correlated subquery**.

**Part B – Correlated & Nested Subqueries (25 marks)**

1. Retrieve the names of customers who placed an order on the **same date as 'John'**.
2. Find the name of the customer who placed the **most recent order**.
3. Write a query to find the product that has the **second lowest price** using a subquery.
4. Display customer names who have spent **more than double the average spending**.
5. List customers whose **total order amount is more than the total order amount of any customer from 'Delhi'**.

**Part C – Join + Subquery Mix (30 marks)**

1. Use a correlated subquery to find customers who have placed **more orders than the average** number of orders placed by all customers.
2. Find all products whose **total sales quantity** is higher than the average total quantity sold per product.
3. Get customers who have ordered at least **one product that no one else has ordered**.
4. Retrieve all orders where the total order amount is equal to the **maximum order amount for that customer**.
5. Write a query to list customers who have **never placed an order with a quantity greater than 5**.

**Part D – Joins & Set Operations (25 marks)**

1. Use a subquery to list the **top 5 customers by total spending**.
2. Find all customers who have only ordered **one unique product** using subqueries.
3. List all orders where the amount is **not in the top 10 highest order amounts**.
4. Retrieve customer names who placed an order in the **last 7 days** but **not** in the **previous 30 days** before that.
5. Write a query to list all products ordered in the **highest number of distinct orders**.

Part – A

Here are the answers to all 20 SQL questions written in **sentence case** as requested, based on your schema:

**Part A – Subqueries (20 marks)**

1. **Find customers who have placed orders in every month of the current year.**

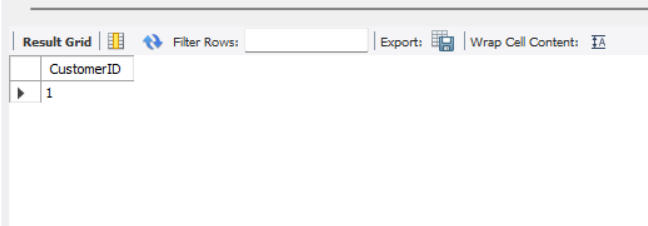
Select CustomerID

From Orders

Where Year(OrderDate) = 2025

Group by CustomerID

Having Count(Distinct Month(OrderDate)) = 12;



1. **Retrieve the names of products that have been ordered more than the average quantity across all products.**

Select ProductName

From Products

Where ProductID In (

Select ProductID

From OrderDetails

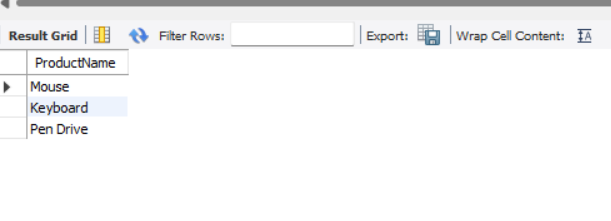
Group by ProductID

Having Avg(Quantity) > (

Select Avg(Quantity) From OrderDetails

)

);



1. **Find customers who have never ordered a product priced above ₹1000.**

Select Name

From Customers

Where CustomerID Not In (

Select Distinct O.CustomerID

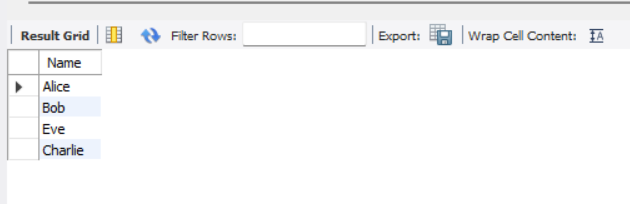
From Orders O

Join OrderDetails OD On O.OrderID = OD.OrderID

Join Products P On OD.ProductID = P.ProductID

Where P.Price > 1000

);



1. **List the top 3 products by total revenue using a subquery.**

Select ProductName

From Products

Where ProductID In (

Select ProductID

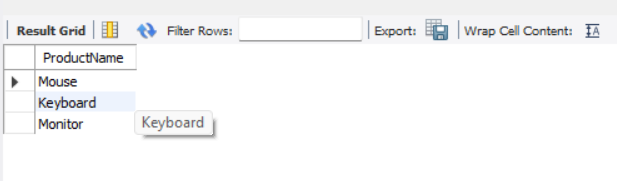
From OrderDetails

Group by ProductID

Order by Sum(Quantity \* (Select Price From Products Where Products.ProductID = OrderDetails.ProductID)) Desc

Limit 3

);



1. **Find orders that contain only one product using a correlated subquery.**

Select OrderID

From OrderDetails OD1

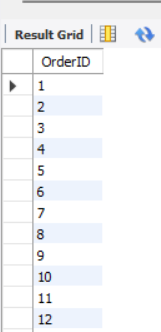
Where 1 = (

Select Count(\*)

From OrderDetails OD2

Where OD1.OrderID = OD2.OrderID

);



**Part B – Correlated & Nested Subqueries (25 marks)**

1. **Retrieve the names of customers who placed an order on the same date as 'John'.**

Select Distinct C.Name

From Customers C

Join Orders O On C.CustomerID = O.CustomerID

Where O.OrderDate In (

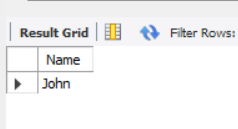
Select OrderDate

From Orders O1

Join Customers C1 On O1.CustomerID = C1.CustomerID

Where C1.Name = 'John'

);



1. **Find the name of the customer who placed the most recent order.**

Select Name

From Customers

Where CustomerID = (

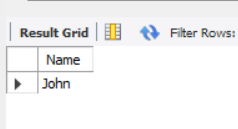
Select CustomerID

From Orders

Order by OrderDate Desc

Limit 1

);



1. **Find the product that has the second lowest price using a subquery.**

Select ProductName

From Products

Where Price = (

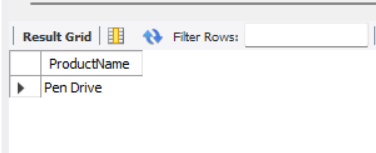
Select Price

From Products

Order by Price Asc

Limit 1 Offset 1

);



1. **Display customer names who have spent more than double the average spending.**

Select Name

From Customers

Where CustomerID In (

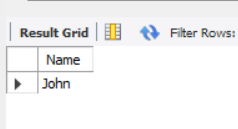
Select CustomerID

From Orders

Group by CustomerID

Having Sum(Amount) > 2 \* (Select Avg(Amount) From Orders)

);



1. **List customers whose total order amount is more than the total order amount of any customer from 'Delhi'.**

Select Name

From Customers

Where CustomerID In (

Select CustomerID

From Orders

Group by CustomerID

Having Sum(Amount) > Any (

Select Sum(Amount)

From Orders O

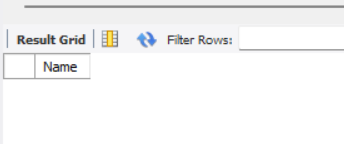
Join Customers C On O.CustomerID = C.CustomerID

Where C.City = 'Delhi'

Group by O.CustomerID

)

);



**Part C – Join + Subquery Mix (30 marks)**

1. **Find customers who have placed more orders than the average number of orders placed by all customers.**

Select Name

From Customers C

Where (

Select Count(\*)

From Orders O

Where O.CustomerID = C.CustomerID

) > (

Select Avg(OrderCount)

From (

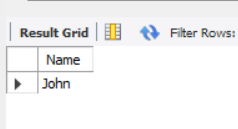
Select Count(\*) as OrderCount

From Orders

Group by CustomerID

) as AvgOrders

);



1. **Find all products whose total sales quantity is higher than the average total quantity sold per product.**

Select ProductName

From Products P

Where (

Select Sum(Quantity)

From OrderDetails OD

Where OD.ProductID = P.ProductID

) > (

Select Avg(TotalQty)

From (

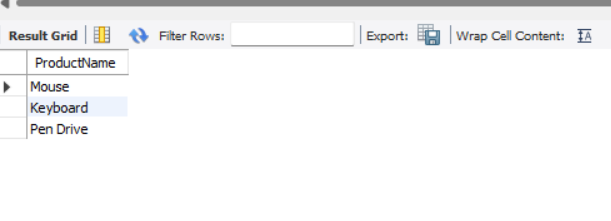
Select Sum(Quantity) as TotalQty

From OrderDetails

Group by ProductID

) as QtyStats

);



1. **Get customers who have ordered at least one product that no one else has ordered.**

Select Distinct Name

From Customers C

Join Orders O On C.CustomerID = O.CustomerID

Join OrderDetails OD On O.OrderID = OD.OrderID

Where OD.ProductID Not In (

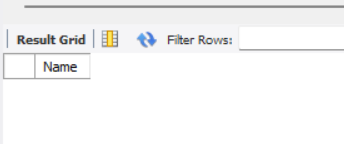
Select ProductID

From OrderDetails

Group by ProductID

Having Count(Distinct OrderID) > 1

);



1. **Retrieve all orders where the total order amount is equal to the maximum order amount for that customer.**

Select OrderID

From Orders O

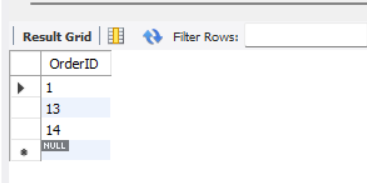
Where Amount = (

Select Max(Amount)

From Orders

Where CustomerID = O.CustomerID

);



1. **List customers who have never placed an order with a quantity greater than 5.**

Select Name

From Customers

Where CustomerID Not In (

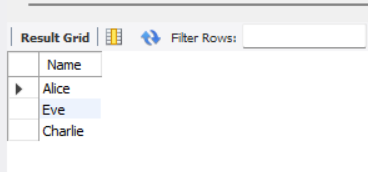
Select Distinct O.CustomerID

From Orders O

Join OrderDetails OD On O.OrderID = OD.OrderID

Where Quantity > 5

);



**Part D – Joins & Set Operations (25 marks)**

1. **List the top 5 customers by total spending using a subquery.**

Select Name

From Customers

Where CustomerID In (

Select CustomerID

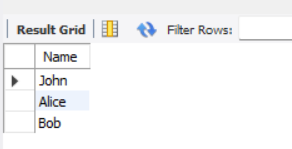
From Orders

Group by CustomerID

Order by Sum(Amount) Desc

Limit 5

);



1. **Find all customers who have only ordered one unique product using subqueries.**

Select Name

From Customers

Where CustomerID In (

Select CustomerID

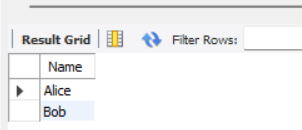
From Orders O

Join OrderDetails OD On O.OrderID = OD.OrderID

Group by O.CustomerID

Having Count(Distinct OD.ProductID) = 1

);



1. **List all orders where the amount is not in the top 10 highest order amounts.**

Select OrderID

From Orders

Where Amount Not In (

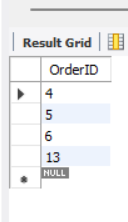
Select Amount

From Orders

Order by Amount Desc

Limit 10

);



1. **Retrieve customer names who placed an order in the last 7 days but not in the previous 30 days before that.**

Select Name

From Customers C

Where C.CustomerID In (

Select CustomerID

From Orders

Where OrderDate Between Date\_Sub(Curdate(), Interval 7 Day) And Curdate()

)

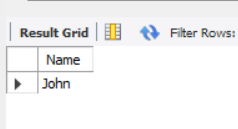
And C.CustomerID Not In (

Select CustomerID

From Orders

Where OrderDate Between Date\_Sub(Curdate(), Interval 37 Day) And Date\_Sub(Curdate(), Interval 8 Day)

);



1. **List all products ordered in the highest number of distinct orders.**

Select ProductName

From Products

Where ProductID In (

Select ProductID

From OrderDetails

Group by ProductID

Having Count(Distinct OrderID) = (

Select Max(OrderCount)

From (

Select Count(Distinct OrderID) as OrderCount

From OrderDetails

Group by ProductID

) as SubCounts

)

);

