**Available Tables to Perform Below Queries :**

1. **Employees Table**

| EmployeeID | Name | Department | JobTitle | Age | Salary | HireDate |
| --- | --- | --- | --- | --- | --- | --- |
| 1 | Amit Sharma | HR | Manager | 35 | 70000.00 | 2012-05-20 |
| 2 | Neha Patil | Finance | Analyst | 28 | 60000.00 | 2018-07-15 |
| 3 | Rohan Kumar | IT | Developer | 30 | 55000.00 | 2020-02-10 |
| 4 | Priya Singh | HR | Executive | 25 | 45000.00 | 2022-08-01 |
| 5 | Vikas Kapoor | Finance | Manager | 38 | 80000.00 | 2010-12-05 |

1. **Orders Table**

| OrderID | CustomerID | ProductID | Amount | OrderDate |
| --- | --- | --- | --- | --- |
| 1 | 1 | 101 | 1500.00 | 2025-03-10 |
| 2 | 2 | 102 | 2000.00 | 2025-03-12 |
| 3 | 1 | 103 | 750.00 | 2025-03-15 |
| 4 | 3 | 101 | 3000.00 | 2025-03-18 |
| 5 | 2 | 104 | 500.00 | 2025-03-20 |

1. **Products Table**

| ProductID | ProductName | Category | Price |
| --- | --- | --- | --- |
| 101 | Laptop | Electronics | 50000.00 |
| 102 | Mobile | Electronics | 20000.00 |
| 103 | Washing Machine | Home Appliances | 15000.00 |
| 104 | Air Conditioner | Home Appliances | 30000.00 |
| 105 | Headphones | Accessories | 1500.00 |

1. **Customers Table**

| CustomerID | Name | Age | City | TotalSpent |
| --- | --- | --- | --- | --- |
| 1 | Rahul Verma | 32 | Mumbai | 8000.00 |
| 2 | Pooja Mehta | 29 | Delhi | 4500.00 |
| 3 | Kunal Shah | 35 | Pune | 3000.00 |
| 4 | Anita Nair | 40 | Bangalore | 6000.00 |
| 5 | Sameer Joshi | 28 | Hyderabad | 2500.00 |

1. **Library Table**

| BookID | Title | Author | CopiesAvailable |
| --- | --- | --- | --- |
| 201 | The Alchemist | Paulo Coelho | 8 |
| 202 | Atomic Habits | James Clear | 5 |
| 203 | Rich Dad Poor Dad | Robert Kiyosaki | 12 |
| 204 | Think Like a Monk | Jay Shetty | 4 |
| 205 | Ikigai | Francesc Miralles | 10 |

1. **Borrowings Table**

| BorrowingID | MemberID | BookID | BorrowDate | ReturnDate |
| --- | --- | --- | --- | --- |
| 1 | 1 | 201 | 2025-02-15 | 2025-03-01 |
| 2 | 2 | 202 | 2025-03-05 | 2025-03-15 |
| 3 | 1 | 203 | 2025-02-28 | 2025-03-12 |
| 4 | 3 | 204 | 2025-03-08 | 2025-03-20 |
| 5 | 2 | 205 | 2025-03-10 | 2025-03-22 |

**🔹 Basic Aggregate Functions**

1. Write a query to find the total number of employees in the Employees table.

SELECT COUNT(\*) AS EmployeeCount FROM Employees;

| **EmployeeCount** |
| --- |
| 5 |

2. Retrieve the sum of all salaries from the Employees table.

SELECT SUM(Salary) AS TotalSalary FROM Employees;

| **TotalSalary** |
| --- |
| 310000.00 |

3. Find the average price of all products in the Products table.

SELECT AVG(Price) AS AveragePrice FROM Products;

| **AveragePrice** |
| --- |
| 23300.00 |

4. Write a query to find the maximum age of employees.

SELECT MAX(Age) AS MaxAge FROM Employees;

| **MaxAge** |
| --- |
| 38 |

5. Write a query to count the number of orders placed in the Orders table.

SELECT COUNT(\*) AS OrderCount FROM Orders;

| **OrderCount** |
| --- |
| 5 |

**🔹 GROUP BY Queries**

6. Find the total number of employees in each department.

SELECT Department, COUNT(\*) AS EmployeeCount FROM Employees GROUP BY Department;

| **Department** | **EmployeeCount** |
| --- | --- |
| HR | 2 |
| Finance | 2 |
| IT | 1 |

7. Find the average salary per department.

SELECT Department, AVG(Salary) AS AvgSalary FROM Employees GROUP BY Department;

| **Department** | **AvgSalary** |
| --- | --- |
| HR | 57500.00 |
| Finance | 70000.00 |
| IT | 55000.00 |

8. Retrieve the number of products in each category.

SELECT Category, COUNT(\*) AS ProductCount FROM Products GROUP BY Category;

| **Category** | **ProductCount** |
| --- | --- |
| Electronics | 2 |
| Home Appliances | 2 |
| Accessories | 1 |

9. Find the total revenue generated by each customer from the Orders table.

SELECT CustomerID, SUM(Amount) AS TotalRevenue FROM Orders GROUP BY CustomerID;

| **CustomerID** | **TotalRevenue** |
| --- | --- |
| 1 | 2250.00 |
| 2 | 2500.00 |
| 3 | 3000.00 |

10. Count the number of employees in each job title

SELECT JobTitle, COUNT(\*) AS EmployeeCount FROM Employees GROUP BY JobTitle;

| **JobTitle** | **EmployeeCount** |
| --- | --- |
| Manager | 2 |
| Analyst | 1 |
| Developer | 1 |
| Executive | 1 |

**🔹 HAVING Queries**

11. Retrieve departments with more than 5 employees.

SELECT Department, COUNT(\*) AS EmployeeCount FROM Employees GROUP BY Department HAVING COUNT(\*) > 5;

| **Department** | **EmployeeCount** |
| --- | --- |
| HR | 2 |
| Finance | 2 |

12. Find categories where the total product price exceeds $10,000.

SELECT Category, SUM(Price) AS TotalPrice FROM Products GROUP BY Category HAVING SUM(Price) > 10000;

| **Category** | **TotalPrice** |
| --- | --- |
| Electronics | 70000.00 |
| Home Appliances | 45000.00 |

13. Display job titles where the average salary is greater than $60,000.

SELECT JobTitle, AVG(Salary) AS AvgSalary FROM Employees GROUP BY JobTitle HAVING AVG(Salary) > 60000;

| **JobTitle** | **AvgSalary** |
| --- | --- |
| Manager | 75000.00 |

14. Show customers who have spent more than $5000.

SELECT CustomerID, SUM(Amount) AS TotalSpent FROM Orders GROUP BY CustomerID HAVING SUM(Amount) > 5000;

| **CustomerID** | **TotalSpent** |
| --- | --- |
| 3 | 3000.00 |

15. Retrieve product categories with more than 10 products.

SELECT Category, COUNT(\*) AS ProductCount FROM Products GROUP BY Category HAVING COUNT(\*) > 10;

| **Category** | **ProductCount** |
| --- | --- |
| Electronics | 2 |
| Home Appliances | 2 |

**🔹 Intermediate Queries**

16. Retrieve the top 3 highest product prices using MAX() and LIMIT.

SELECT Price FROM Products ORDER BY Price DESC LIMIT 3;

| **Price** |
| --- |
| 50000.00 |
| 30000.00 |
| 20000.00 |

17. Find the minimum, maximum, and average salary of employees in the Employees table.

SELECT MIN(Salary) AS MinSalary, MAX(Salary) AS MaxSalary, AVG(Salary) AS AvgSalary FROM Employees;

| **MinSalary** | **MaxSalary** | **AvgSalary** |
| --- | --- | --- |
| 45000.00 | 80000.00 | 62000.00 |

18. Write a query to list customers who placed more than 3 orders.

SELECT CustomerID, COUNT(\*) AS OrderCount FROM Orders GROUP BY CustomerID HAVING COUNT(\*) > 3;

| **CustomerID** | **OrderCount** |
| --- | --- |
| 1 | 2 |
| 2 | 2 |

19. Retrieve the total number of books borrowed by each member from a Borrowings table.

SELECT MemberID, COUNT(BookID) AS TotalBorrowed FROM Borrowings GROUP BY MemberID;

| **MemberID** | **TotalBorrowed** |
| --- | --- |
| 1 | 2 |
| 2 | 2 |
| 3 | 1 |

20. Find the oldest and youngest customer from the Customers table.

SELECT MIN(Age) AS Youngest, MAX(Age) AS Oldest FROM Customers;

| **Youngest** | **Oldest** |
| --- | --- |
| 28 | 40 |

**🔹 Advanced Queries**

21. Find the most expensive product in each category.

SELECT Category, MAX(Price) AS MostExpensive FROM Products GROUP BY Category;

| **Category** | **MostExpensive** |
| --- | --- |
| Electronics | 50000.00 |
| Home Appliances | 30000.00 |
| Accessories | 1500.00 |

22. Find employees whose salary is greater than the average salary.

SELECT EmployeeID, Name, Salary FROM Employees WHERE Salary > (SELECT AVG(Salary) FROM Employees);

| **EmployeeID** | **Name** | **Salary** |
| --- | --- | --- |
| 1 | Amit Sharma | 70000.00 |
| 5 | Vikas Kapoor | 80000.00 |

23. Retrieve customers who made at least one purchase in every product category.

SELECT O.CustomerID, C.Name FROM Orders O

JOIN Customers C ON O.CustomerID = C.CustomerID

JOIN Products P ON O.ProductID = P.ProductID

GROUP BY O.CustomerID, C.Name

HAVING COUNT(DISTINCT P.Category) =

( SELECT COUNT(DISTINCT Category) FROM Products);

| **CustomerID** | **Name** |
| --- | --- |
| 1 | Rahul Verma |

24. Find the department with the highest total salary.

SELECT Department, SUM(Salary) AS TotalSalary FROM Employees GROUP BY Department ORDER BY TotalSalary DESC LIMIT 1;

| **Department** | **TotalSalary** |
| --- | --- |
| Finance | 140000.00 |

25. Write a query to find employees who earn above the department’s average salary.

SELECT E.EmployeeID, E.Name, E.Salary, E.Department FROM Employees E WHERE E.Salary > (SELECT AVG(Salary) FROM Employees WHERE Department = E.Department );

| **EmployeeID** | **Name** | **Salary** | **Department** |
| --- | --- | --- | --- |
| 1 | Amit Sharma | 70000.00 | HR |
| 5 | Vikas Kapoor | 80000.00 | Finance |

26. Find products where the total revenue exceeds $50,000.

SELECT ProductID, SUM(Amount) AS TotalRevenue FROM Orders GROUP BY ProductID HAVING SUM(Amount) > 50000;

|  |
| --- |
| Empty Set |

27. Retrieve employees who have been in the company for more than 10 years.

SELECT EmployeeID, Name, HireDate FROM Employees WHERE DATEDIFF(CURDATE(), HireDate) > 3650;

| **EmployeeID** | **Name** | **HireDate** |
| --- | --- | --- |
| 1 | Amit Sharma | 2012-05-20 |
| 5 | Vikas Kapoor | 2010-12-05 |

28. Write a query to find books with more than 5 copies available in a Library table.

SELECT BookID, Title, CopiesAvailable FROM Library WHERE CopiesAvailable > 5;

| **BookID** | **Title** | **CopiesAvailable** |
| --- | --- | --- |
| 201 | The Alchemist | 8 |
| 203 | Rich Dad Poor Dad | 12 |
| 205 | Ikigai | 10 |

29. Find the customer who placed the most orders.

SELECT CustomerID, COUNT(\*) AS OrderCount FROM Orders GROUP BY CustomerID ORDER BY OrderCount DESC LIMIT 1;

| **CustomerID** | **OrderCount** |
| --- | --- |
| 1 | 2 |

30. Retrieve departments where the total number of employees is below the company’s average

SELECT Department, COUNT(\*) AS EmployeeCount FROM Employees GROUP BY Department HAVING COUNT(\*) < (

SELECT AVG(DepartmentCount) FROM ( SELECT COUNT(\*) AS DepartmentCount FROM Employees GROUP BY Department) AS AvgDept);

| **Department** | **EmployeeCount** |
| --- | --- |
| IT | 1 |