DBMS Practical Questions with SQL Queries

### ****1. Create a Sales Table and Use Aggregate Functions****

-- a) Create Sales table

CREATE TABLE Sales (

SaleID INT PRIMARY KEY,

ProductID INT,

Quantity INT,

SaleAmount DECIMAL(10,2),

SaleDate DATE

);

-- b) Insert sample records

INSERT INTO Sales VALUES

(1, 101, 2, 2000, '2025-02-01'),

(2, 102, 1, 1500, '2025-02-15'),

(3, 103, 3, 3000, '2025-03-01'),

(4, 101, 1, 1000, '2025-03-10'),

(5, 104, 4, 5000, '2025-02-20'),

(6, 105, 2, 2500, '2025-01-25'),

(7, 102, 1, 1500, '2025-01-30'),

(8, 106, 5, 7500, '2025-03-05'),

(9, 107, 3, 3000, '2025-02-10'),

(10, 108, 2, 2200, '2025-02-25');

-- c) Total revenue

SELECT SUM(SaleAmount) AS TotalRevenue FROM Sales;

-- d) Product with highest sale amount

SELECT MAX(SaleAmount) AS HighestSale FROM Sales;

-- e) Average sale amount per transaction

SELECT AVG(SaleAmount) AS AverageSale FROM Sales;

### ****2. Use DDL and DML Commands****

sql

CopyEdit

-- a) Create Products table

CREATE TABLE Products (

ProductID INT PRIMARY KEY,

ProductName VARCHAR(50),

Price DECIMAL(10,2),

StockQuantity INT

);

-- b) Insert 5 records and display

INSERT INTO Products VALUES

(1, 'Laptop', 50000, 10),

(2, 'Mouse', 500, 50),

(3, 'Keyboard', 1000, 30),

(4, 'Monitor', 8000, 20),

(5, 'Printer', 7000, 15);

SELECT \* FROM Products;

-- c) Update price of product with ID = 3

UPDATE Products SET Price = 1200 WHERE ProductID = 3;

SELECT \* FROM Products WHERE ProductID = 3;

-- d) Delete a product

DELETE FROM Products WHERE ProductID = 5;

SELECT \* FROM Products;

-- e) Add new column Discount with default 5%

ALTER TABLE Products ADD Discount INT DEFAULT 5;

### ****3. Create Customer Table with Integrity Constraints****

sql

CopyEdit

-- a) Create table with constraints

CREATE TABLE Customers (

CustomerID INT PRIMARY KEY,

Name VARCHAR(50),

Email VARCHAR(100) UNIQUE,

Age INT CHECK (Age > 18),

Country VARCHAR(50) DEFAULT 'India'

);

-- b) Insert valid record

INSERT INTO Customers (CustomerID, Name, Email, Age) VALUES

(1, 'Rahul', 'rahul@example.com', 25);

-- c) Age check (will cause error)

-- INSERT INTO Customers VALUES (2, 'Amit', 'amit@example.com', 16, 'India');

-- d) Email uniqueness check (will cause error)

-- INSERT INTO Customers VALUES (3, 'Sunil', 'rahul@example.com', 22, 'USA');

-- e) Retrieve customers older than 25 and not from India

SELECT \* FROM Customers WHERE Age > 25 AND Country <> 'India';

### ****4. Table with Constraints (Foreign Key)****

sql

CopyEdit

-- a) Create Department and EmployeeDetails tables

CREATE TABLE Department (

DepartmentID INT PRIMARY KEY,

DeptName VARCHAR(50)

);

CREATE TABLE EmployeeDetails (

EmployeeID INT PRIMARY KEY,

Name VARCHAR(50),

DepartmentID INT,

Salary DECIMAL(10,2),

FOREIGN KEY (DepartmentID) REFERENCES Department(DepartmentID)

);

-- b) Insert valid and invalid records

INSERT INTO Department VALUES (1, 'HR'), (2, 'IT');

INSERT INTO EmployeeDetails VALUES (101, 'Anil', 1, 30000);

-- Invalid: DepartmentID doesn't exist

-- INSERT INTO EmployeeDetails VALUES (102, 'Sunita', 5, 28000);

-- c) Duplicate primary key test (will cause error)

-- INSERT INTO EmployeeDetails VALUES (101, 'Ravi', 2, 35000);

-- d) Modify Salary column to UNIQUE

ALTER TABLE EmployeeDetails ADD CONSTRAINT salary\_unique UNIQUE (Salary);

-- Try inserting duplicate salary

-- INSERT INTO EmployeeDetails VALUES (103, 'Priya', 2, 30000); -- will cause error

-- e) Delete employee

DELETE FROM EmployeeDetails WHERE EmployeeID = 101;

### ****5. Employee Table with Various Columns****

sql

CopyEdit

-- a) Create Employee table

CREATE TABLE Employee (

EmployeeID INT PRIMARY KEY,

Name VARCHAR(50),

Salary DECIMAL(10,2),

JoiningDate DATE,

ActiveStatus BOOLEAN

);

-- b) Insert 5 records

INSERT INTO Employee VALUES

(1, 'Amit Sharma', 40000, '2022-12-15', TRUE),

(2, 'Nisha Mehta', 35000, '2023-02-10', TRUE),

(3, 'Ravi Kumar', 30000, '2023-05-05', FALSE),

(4, 'Sneha Jain', 45000, '2021-07-25', TRUE),

(5, 'Arjun Singh', 38000, '2024-01-15', TRUE);

-- c) Employees joined before Jan 1, 2023

SELECT \* FROM Employee WHERE JoiningDate < '2023-01-01';

-- d) Update Amit Sharma's salary by 10%

UPDATE Employee SET Salary = Salary \* 1.10 WHERE Name = 'Amit Sharma';

SELECT \* FROM Employee WHERE Name = 'Amit Sharma';

-- e) Active employees

SELECT \* FROM Employee WHERE ActiveStatus = TRUE;

# 6. Create a Student Table and Perform Queries

a) Create Student table  
```sql  
CREATE TABLE Student (  
 RollNo INT PRIMARY KEY,  
 Name VARCHAR(50),  
 Marks INT,  
 Grade CHAR(1)  
);  
```  
  
b) Insert 5 records  
```sql  
INSERT INTO Student VALUES  
(1, 'Aman', 85, 'A'),  
(2, 'Priya', 76, 'B'),  
(3, 'Rohan', 92, 'A'),  
(4, 'Neha', 68, 'C'),  
(5, 'Kiran', 59, 'D');  
```  
  
c) Students with marks more than 75  
```sql  
SELECT \* FROM Student WHERE Marks > 75;  
```  
  
d) Update grade of a student  
```sql  
UPDATE Student SET Grade = 'B' WHERE RollNo = 5;  
SELECT \* FROM Student WHERE RollNo = 5;  
```  
  
e) Delete student with RollNo = 4  
```sql  
DELETE FROM Student WHERE RollNo = 4;  
```

# 7. Create an Order Table and Use BETWEEN, IN, LIKE, IS NULL

a) Create Order table  
```sql  
CREATE TABLE Orders (  
 OrderID INT PRIMARY KEY,  
 CustomerName VARCHAR(50),  
 OrderAmount DECIMAL(10,2),  
 OrderDate DATE  
);  
```  
  
b) Insert 5 records  
```sql  
INSERT INTO Orders VALUES  
(1, 'Rakesh', 2000, '2025-02-01'),  
(2, 'Suresh', 5000, '2025-02-05'),  
(3, 'Nilesh', 10000, '2025-02-10'),  
(4, 'Amit', NULL, '2025-02-15'),  
(5, 'Sneha', 7000, '2025-02-20');  
```  
  
c) Orders between 2000 and 8000  
```sql  
SELECT \* FROM Orders WHERE OrderAmount BETWEEN 2000 AND 8000;  
```  
  
d) Orders by customers 'Rakesh', 'Amit'  
```sql  
SELECT \* FROM Orders WHERE CustomerName IN ('Rakesh', 'Amit');  
```  
  
e) Orders by customers whose name starts with 'S'  
```sql  
SELECT \* FROM Orders WHERE CustomerName LIKE 'S%';  
```  
  
f) Orders where amount is NULL  
```sql  
SELECT \* FROM Orders WHERE OrderAmount IS NULL;  
```

# 8. Create Library Table and Perform Date-Based Queries

a) Create Library table  
```sql  
CREATE TABLE Library (  
 BookID INT PRIMARY KEY,  
 Title VARCHAR(100),  
 Author VARCHAR(50),  
 IssueDate DATE,  
 ReturnDate DATE  
);  
```  
  
b) Insert 5 records  
```sql  
INSERT INTO Library VALUES  
(1, 'C Programming', 'E. Balagurusamy', '2025-01-01', '2025-01-15'),  
(2, 'DBMS Concepts', 'Korth', '2025-02-01', '2025-02-10'),  
(3, 'Java Complete Ref.', 'Herbert Schildt', '2025-02-20', '2025-03-01'),  
(4, 'Python Basics', 'Guido', '2025-03-01', '2025-03-15'),  
(5, 'AI Made Easy', 'Peter Norvig', '2025-03-05', '2025-03-20');  
```  
  
c) Books issued in February 2025  
```sql  
SELECT \* FROM Library  
WHERE MONTH(IssueDate) = 2 AND YEAR(IssueDate) = 2025;  
```  
  
d) Books returned after 10 days of issue  
```sql  
SELECT \* FROM Library  
WHERE DATEDIFF(ReturnDate, IssueDate) > 10;  
```  
  
e) Change return date of BookID = 2  
```sql  
UPDATE Library SET ReturnDate = '2025-02-12' WHERE BookID = 2;  
```

# 9. Create Marks Table and Use Aggregate Functions

a) Create Marks table  
```sql  
CREATE TABLE Marks (  
 StudentID INT PRIMARY KEY,  
 Name VARCHAR(50),  
 Subject1 INT,  
 Subject2 INT,  
 Subject3 INT  
);  
```  
  
b) Insert 5 records  
```sql  
INSERT INTO Marks VALUES  
(1, 'Amit', 80, 85, 90),  
(2, 'Ravi', 70, 75, 65),  
(3, 'Priya', 90, 95, 100),  
(4, 'Sneha', 60, 65, 70),  
(5, 'Kiran', 85, 80, 75);  
```  
  
c) Calculate average marks for each student  
```sql  
SELECT Name,  
 (Subject1 + Subject2 + Subject3) / 3 AS AverageMarks  
FROM Marks;  
```  
  
d) Find highest total marks  
```sql  
SELECT Name,  
 (Subject1 + Subject2 + Subject3) AS TotalMarks  
FROM Marks  
ORDER BY TotalMarks DESC  
LIMIT 1;  
```  
  
e) Students with Subject1 marks greater than 75  
```sql  
SELECT \* FROM Marks WHERE Subject1 > 75;  
```

# 10. Create Inventory Table and Perform Updates, Deletes

a) Create Inventory table  
```sql  
CREATE TABLE Inventory (  
 ItemID INT PRIMARY KEY,  
 ItemName VARCHAR(50),  
 Quantity INT,  
 Price DECIMAL(10,2)  
);  
```  
  
b) Insert 5 records  
```sql  
INSERT INTO Inventory VALUES  
(1, 'Pen', 100, 10),  
(2, 'Notebook', 50, 30),  
(3, 'Eraser', 200, 5),  
(4, 'Marker', 80, 20),  
(5, 'File', 40, 50);  
```  
  
c) Increase quantity of 'Notebook' by 20  
```sql  
UPDATE Inventory SET Quantity = Quantity + 20 WHERE ItemName = 'Notebook';  
```  
  
d) Delete item with Quantity less than 50  
```sql  
DELETE FROM Inventory WHERE Quantity < 50;  
```  
  
e) Total inventory value (Quantity × Price)  
```sql  
SELECT SUM(Quantity \* Price) AS TotalInventoryValue FROM Inventory;  
```