1st question: Perform Simple Queries with Where Operators

-- Step 1: Create Students table

CREATE TABLE Students (

StudentID INT PRIMARY KEY,

RollNo INT UNIQUE,

FirstName VARCHAR(50),

LastName VARCHAR(50),

DateOfBirth DATE

);

-- Step 2: Insert sample data

INSERT INTO Students VALUES

(1, 101, 'Rahul', 'Sharma', '2002-05-15'),

(2, 102, 'Priya', 'Kumar', '2001-09-20'),

(3, 103, 'Amit', 'Singh', '2003-12-11'),

(4, 104, 'Neha', 'Verma', '2000-08-25'),

(5, 105, 'Ravi', 'Patel', '2002-01-30');

-- Step 3: Simple Queries using WHERE

-- 1. Find student with RollNo = 102

SELECT \* FROM Students WHERE RollNo = 102;

-- 2. Find students whose FirstName = 'Rahul'

SELECT \* FROM Students WHERE FirstName = 'Rahul';

-- 3. Find students born after 2002

SELECT \* FROM Students WHERE DateOfBirth > '2002-01-01';

-- 4. Find students whose LastName is 'Patel' or 'Verma'

SELECT \* FROM Students WHERE LastName IN ('Patel', 'Verma');

-- 5. Find students whose RollNo is between 102 and 104

SELECT \* FROM Students WHERE RollNo BETWEEN 102 AND 104;

-- 6. Find students whose FirstName starts with 'R'

SELECT \* FROM Students WHERE FirstName LIKE 'R%';

2nd question: Perform Queries involving String Functions

-- Step 1: Create Students table

CREATE TABLE Students (

StudentID INT PRIMARY KEY,

RollNo INT UNIQUE,

FirstName VARCHAR(50),

LastName VARCHAR(50),

DateOfBirth DATE

);

-- Step 2: Insert sample data

INSERT INTO Students VALUES

(1, 101, 'Rahul', 'Sharma', '2002-05-15'),

(2, 102, 'Priya', 'Kumar', '2001-09-20'),

(3, 103, 'Amit', 'Singh', '2003-12-11'),

(4, 104, 'Neha', 'Verma', '2000-08-25'),

(5, 105, 'Ravi', 'Patel', '2002-01-30');

-- Step 3: Queries involving String Functions

-- 1. Convert FirstName to UPPERCASE

SELECT FirstName, UPPER(FirstName) AS UpperName FROM Students;

-- 2. Convert LastName to lowercase

SELECT LastName, LOWER(LastName) AS LowerName FROM Students;

-- 3. Concatenate FirstName and LastName

SELECT CONCAT(FirstName, ' ', LastName) AS FullName FROM Students;

-- 4. Find length of FirstName

SELECT FirstName, LENGTH(FirstName) AS NameLength FROM Students;

-- 5. Extract first 3 letters of LastName

SELECT LastName, SUBSTRING(LastName, 1, 3) AS ShortForm FROM Students;

-- 6. Replace 'a' with '@' in FirstName

SELECT FirstName, REPLACE(FirstName, 'a', '@') AS ModifiedName FROM Students;

3rd question: Create a Students Table with Columns: StudentID (Primary Key), RollNo (Unique), FirstName, LastName, DateOfBirth also insert 3 records in the Table.

CREATE TABLE Students (

StudentID INT PRIMARY KEY,

RollNo INT UNIQUE,

FirstName VARCHAR(50),

LastName VARCHAR(50),

DateOfBirth DATE

);

INSERT INTO Students VALUES

(1, 101, 'Rahul', 'Sharma', '2002-05-15'),

(2, 102, 'Priya', 'Kumar', '2001-09-20'),

(3, 103, 'Amit', 'Singh', '2003-12-11');

4th question: Perform Queries involving Math Functions

-- Step 1: Create a Numbers table (for math operations)

CREATE TABLE Numbers (

NumID INT PRIMARY KEY,

Value INT

);

-- Step 2: Insert sample data

INSERT INTO Numbers VALUES

(1, -25),

(2, 45),

(3, 78),

(4, 120),

(5, -90);

-- Step 3: Queries using Math Functions

-- 1. Absolute value of numbers

SELECT Value, ABS(Value) AS AbsoluteValue FROM Numbers;

-- 2. Square root of values

SELECT Value, SQRT(ABS(Value)) AS SquareRoot FROM Numbers;

-- 3. Power function (Value^2)

SELECT Value, POWER(Value, 2) AS SquareValue FROM Numbers;

-- 4. Round function

SELECT Value, ROUND(Value / 7.0, 2) AS RoundedValue FROM Numbers;

-- 5. Modulus (remainder)

SELECT Value, MOD(Value, 7) AS ModulusValue FROM Numbers;

5th question: Create a Employees Table with Columns: EmployeeID (Primary Key), FirstName, LastName, DateOfBirth, Email (Unique), PhoneNumber also insert 3 records in the Table

-- Step 1: Create Employees Table

CREATE TABLE Employees (

EmployeeID INT PRIMARY KEY,

FirstName VARCHAR(50),

LastName VARCHAR(50),

DateOfBirth DATE,

Email VARCHAR(100) UNIQUE,

PhoneNumber VARCHAR(15)

);

-- Step 2: Insert sample data

INSERT INTO Employees VALUES

(1, 'John', 'Doe', '1990-03-15', 'john.doe@email.com', '9876543210'),

(2, 'Sara', 'Khan', '1992-07-25', 'sara.khan@email.com', '9876543211'),

(3, 'David', 'Smith', '1988-11-05', 'david.smith@email.com', '9876543212');

6th question: Perform Queries involving Date Functions

-- Step 1: Create Employees table

CREATE TABLE Employees (

EmployeeID INT PRIMARY KEY,

FirstName VARCHAR(50),

LastName VARCHAR(50),

DateOfBirth DATE,

Email VARCHAR(100) UNIQUE,

PhoneNumber VARCHAR(15)

);

-- Step 2: Insert sample data

INSERT INTO Employees VALUES

(1, 'John', 'Doe', '1990-03-15', 'john.doe@email.com', '9876543210'),

(2, 'Sara', 'Khan', '1992-07-25', 'sara.khan@email.com', '9876543211'),

(3, 'David', 'Smith', '1988-11-05', 'david.smith@email.com', '9876543212');

-- Step 3: Queries using Date Functions

-- 1. Extract year of birth

SELECT FirstName, YEAR(DateOfBirth) AS BirthYear FROM Employees;

-- 2. Extract month of birth

SELECT FirstName, MONTH(DateOfBirth) AS BirthMonth FROM Employees;

-- 3. Find day of month

SELECT FirstName, DAY(DateOfBirth) AS BirthDay FROM Employees;

-- 4. Calculate Age (approx in years)

SELECT FirstName, TIMESTAMPDIFF(YEAR, DateOfBirth, CURDATE()) AS Age FROM Employees;

-- 5. Show employees born before year 1991

SELECT \* FROM Employees WHERE YEAR(DateOfBirth) < 1991;

7th question: Create a Table and Perform the Altering Operation

-- Step 1: Create Departments table

CREATE TABLE Departments (

DeptID INT PRIMARY KEY,

DeptName VARCHAR(50)

);

-- Step 2: Perform Altering Operations

-- Add a new column

ALTER TABLE Departments ADD Location VARCHAR(50);

-- Modify an existing column

ALTER TABLE Departments MODIFY DeptName VARCHAR(100);

-- Drop a column

ALTER TABLE Departments DROP COLUMN Location;

8th question: Perform the Simple Queries with Aggregate functions

-- Step 1: Create Marks table

CREATE TABLE Marks (

StudentID INT PRIMARY KEY,

StudentName VARCHAR(50),

Score INT

);

-- Step 2: Insert sample data

INSERT INTO Marks VALUES

(1, 'Rahul', 85),

(2, 'Priya', 92),

(3, 'Amit', 76),

(4, 'Neha', 88),

(5, 'Ravi', 95);

-- Step 3: Aggregate Function Queries

-- 1. Count number of students

SELECT COUNT(\*) AS TotalStudents FROM Marks;

-- 2. Average score

SELECT AVG(Score) AS AverageScore FROM Marks;

-- 3. Highest and lowest score

SELECT MAX(Score) AS HighestScore, MIN(Score) AS LowestScore FROM Marks;

-- 4. Total of all scores

SELECT SUM(Score) AS TotalMarks FROM Marks;

9th question: Create a Products Table with Columns: ProductID (Primary Key), ProductName, Category, Price also insert 3 records in the Table

-- Step 1: Create Products table

CREATE TABLE Products (

ProductID INT PRIMARY KEY,

ProductName VARCHAR(50),

Category VARCHAR(50),

Price DECIMAL(10,2)

);

-- Step 2: Insert 3 records

INSERT INTO Products VALUES

(1, 'Laptop', 'Electronics', 55000.00),

(2, 'Chair', 'Furniture', 3500.00),

(3, 'Book', 'Stationery', 500.00);

10th question: Perform the Queries with Aggregate functions: Group by and Having clause

-- Step 1: Create Sales table

CREATE TABLE Sales (

SaleID INT PRIMARY KEY,

ProductName VARCHAR(50),

Category VARCHAR(50),

Quantity INT,

Price DECIMAL(10,2)

);

-- Step 2: Insert sample records

INSERT INTO Sales VALUES

(1, 'Laptop', 'Electronics', 2, 55000.00),

(2, 'Mobile', 'Electronics', 5, 20000.00),

(3, 'Chair', 'Furniture', 10, 3500.00),

(4, 'Table', 'Furniture', 4, 8000.00),

(5, 'Book', 'Stationery', 20, 500.00);

-- 1. Total sales quantity by category

SELECT Category, SUM(Quantity) AS TotalQuantity

FROM Sales

GROUP BY Category;

-- 2. Average price by category

SELECT Category, AVG(Price) AS AveragePrice

FROM Sales

GROUP BY Category

-- 3. Show only categories where total quantity > 5

SELECT Category, SUM(Quantity) AS TotalQuantity

FROM Sales

GROUP BY Category

HAVING SUM(Quantity) > 5;

11th question : Create a Customers Table with Columns: CustomerID (Primary Key), CustomerCode (Unique), FirstName, LastName, Email (Unique), PhoneNumber also insert 3 records in the Table

-- Step 1: Create Customers table

CREATE TABLE Customers (

CustomerID INT PRIMARY KEY,

CustomerCode VARCHAR(20) UNIQUE,

FirstName VARCHAR(50),

LastName VARCHAR(50),

Email VARCHAR(100) UNIQUE,

PhoneNumber VARCHAR(15)

);

-- Step 2: Insert 3 records

INSERT INTO Customers VALUES

(1, 'C001', 'Arun', 'Verma', 'arun@email.com', '987650001'),

(2, 'C002', 'Neha', 'Patel', 'neha@email.com', '987650002'),

(3, 'C003', 'Ravi', 'Gupta', 'ravi@email.com', '987650003');

12th question: Retrieving Data from a Table using Inner Join and Left Outer Join

-- Step 1: Create Orders table

CREATE TABLE Orders (

OrderID INT PRIMARY KEY,

OrderNumber VARCHAR(20) UNIQUE,

CustomerID INT,

OrderDate DATE,

TotalAmount DECIMAL(10,2),

FOREIGN KEY (CustomerID) REFERENCES Customers(CustomerID)

);

-- Step 2: Insert records into Orders

INSERT INTO Orders VALUES

(1, 'O001', 1, '2023-07-10', 2500.00),

(2, 'O002', 2, '2023-07-12', 5000.00),

(3, 'O003', 1, '2023-07-15', 1500.00);

-- Step 3: INNER JOIN (customers who placed orders)

SELECT Customers.FirstName, Customers.LastName, Orders.OrderNumber, Orders.TotalAmount

FROM Customers

INNER JOIN Orders ON Customers.CustomerID = Orders.CustomerID;

-- Step 4: LEFT JOIN (all customers, even those without orders)

SELECT Customers.FirstName, Customers.LastName, Orders.OrderNumber, Orders.TotalAmount

FROM Customers

LEFT JOIN Orders ON Customers.CustomerID = Orders.CustomerID;

13 question: Create a Orders Table with Columns: OrderID (Primary Key), OrderNumber (Unique), CustomerID, OrderDate, TotalAmount also insert 3 records in the Table

-- Step 1: Create Orders table

CREATE TABLE Orders (

OrderID INT PRIMARY KEY,

OrderNumber VARCHAR(20) UNIQUE,

CustomerID INT,

OrderDate DATE,

TotalAmount DECIMAL(10,2)

);

-- Step 2: Insert 3 records

INSERT INTO Orders VALUES

(1, 'O001', 1, '2023-07-10', 2500.00),

(2, 'O002', 2, '2023-07-12', 5000.00),

(3, 'O003', 3, '2023-07-15', 1500.00);

14 question: Create a Table and perform Following Functions: Renaming, Truncating, Dropping Tables

-- Step 1: Create TempTable

CREATE TABLE TempTable (

ID INT PRIMARY KEY,

Name VARCHAR(50)

);

-- Step 2: Rename Table

ALTER TABLE TempTable RENAME TO NewTable;

-- Step 3: Truncate Table (remove all records)

TRUNCATE TABLE NewTable;

-- Step 4: Drop Table

DROP TABLE NewTable;

15 question: Create a Books Table with Columns: BookID (Primary Key), Title, Author, PublishedYear also insert 3 records in the Table

-- Step 1: Create Books table

CREATE TABLE Books (

BookID INT PRIMARY KEY,

Title VARCHAR(100),

Author VARCHAR(50),

PublishedYear INT

);

-- Step 2: Insert 3 records

INSERT INTO Books VALUES

(1, 'Introduction to SQL', 'James', 2015),

(2, 'Database Systems', 'Korth', 2019),

(3, 'Learning MySQL', 'Allen', 2021);

16 question: Retrieving Data from a Table using Right Outer Join and Full Outer Join

-- Step 1: Create Customers Table

CREATE TABLE Customers (

CustomerID INT PRIMARY KEY,

CustomerCode VARCHAR(20) UNIQUE,

FirstName VARCHAR(50),

LastName VARCHAR(50),

Email VARCHAR(100) UNIQUE,

PhoneNumber VARCHAR(15)

);

-- Step 2: Insert sample data into Customers

INSERT INTO Customers VALUES

(1, 'C001', 'Arun', 'Verma', 'arun@email.com', '987650001'),

(2, 'C002', 'Neha', 'Patel', 'neha@email.com', '987650002'),

(3, 'C003', 'Ravi', 'Gupta', 'ravi@email.com', '987650003'),

(4, 'C004', 'Sneha', 'Shah', 'sneha@email.com', '987650004'); -- Customer with no orders

-- Step 3: Create Orders Table

CREATE TABLE Orders (

OrderID INT PRIMARY KEY,

OrderNumber VARCHAR(20) UNIQUE,

CustomerID INT,

OrderDate DATE,

TotalAmount DECIMAL(10,2),

FOREIGN KEY (CustomerID) REFERENCES Customers(CustomerID)

);

-- Step 4: Insert sample data into Orders

INSERT INTO Orders VALUES

(1, 'O001', 1, '2023-07-10', 2500.00),

(2, 'O002', 2, '2023-07-12', 5000.00),

(3, 'O003', 1, '2023-07-15', 1500.00),

(4, 'O004', 5, '2023-07-20', 2000.00); -- Order with no matching customer

-- RIGHT OUTER JOIN: show all orders, even if customer does not exist

SELECT Customers.FirstName, Customers.LastName, Orders.OrderNumber, Orders.TotalAmount

FROM Customers

RIGHT JOIN Orders ON Customers.CustomerID = Orders.CustomerID;

-- FULL OUTER JOIN simulation using UNION

SELECT Customers.FirstName, Customers.LastName, Orders.OrderNumber, Orders.TotalAmount

FROM Customers

LEFT JOIN Orders ON Customers.CustomerID = Orders.CustomerID

UNION

SELECT Customers.FirstName, Customers.LastName, Orders.OrderNumber, Orders.TotalAmount

FROM Customers

RIGHT JOIN Orders ON Customers.CustomerID = Orders.CustomerID;