| 1 | 1. Create a Sales Table and Use Aggregate Functions 2. Create a Sales table with columns: SaleID, ProductID, Quantity, SaleAmount, and SaleDate.   CREATE TABLE Sales (  SaleID INT PRIMARY KEY,  ProductID INT,  Quantity INT,  SaleAmount DECIMAL(10,2),  SaleDate DATE  );   1. Insert at least 10 sales records with different products and quantities.   INSERT INTO Sales (SaleID, ProductID, Quantity, SaleAmount, SaleDate) VALUES  (1, 101, 2, 50.00, '2025-04-01'),  (2, 102, 1, 30.00, '2025-04-02'),  (3, 103, 5, 150.00, '2025-04-03'),  (4, 104, 3, 90.00, '2025-04-04'),  (5, 101, 4, 100.00, '2025-04-05'),  (6, 105, 2, 80.00, '2025-04-06'),  (7, 106, 6, 210.00, '2025-04-07'),  (8, 102, 3, 90.00, '2025-04-08'),  (9, 107, 1, 25.00, '2025-04-09'),  (10, 108, 2, 70.00, '2025-04-10');   1. Write a query to calculate the total revenue generated using the SUM function.   SELECT SUM(SaleAmount) AS TotalRevenue  FROM Sales;   1. Find the product with the highest sale amount using the MAX function.   SELECT ProductID, SaleAmount  FROM Sales  WHERE SaleAmount = (SELECT MAX(SaleAmount) FROM Sales);   1. Retrieve the average sale amount per transaction using the AVG function.   SELECT AVG(SaleAmount) AS AverageSaleAmount  FROM Sales;  Which recent tool or technology have you studied for database management, and can you briefly explain its key features and why it is used in the industry? |
| --- | --- |
| 2 | 2) Use DDL and DML Commands  a) Create a Products table with columns for ProductID, ProductName, Price, and StockQuantity using DDL commands.  CREATE TABLE Products (  ProductID INT PRIMARY KEY,  ProductName VARCHAR(100),  Price DECIMAL(10,2),  StockQuantity INT  );  b) Insert five product records and display all products using a SELECT query.  -- Insert products  INSERT INTO Products (ProductID, ProductName, Price, StockQuantity) VALUES  (1, 'Laptop', 1000.00, 50),  (2, 'Smartphone', 600.00, 150),  (3, 'Headphones', 150.00, 200),  (4, 'Monitor', 300.00, 80),  (5, 'Keyboard', 40.00, 300);  c) Update the price of a product with ProductID = 3 and check the changes using a SELECT statement.  -- Update price  UPDATE Products  SET Price = 175.00  WHERE ProductID = 3;  -- Verify the change  SELECT \* FROM Products  WHERE ProductID = 3;  d) Delete a product from the table and verify whether the changes are reflected.  -- Delete product with ProductID = 5  DELETE FROM Products  WHERE ProductID = 5;  -- Verify deletion  SELECT \* FROM Products;   1. Alter the table to add a new column Discount and set a default value of 5%.   -- Alter table to add a new column  ALTER TABLE Products  ADD Discount DECIMAL(5,2) DEFAULT 5.00;  Which recent tool or technology have you studied for database management, and can you briefly explain its key features and why it is used in the industry? |
| 3 | 3) Create a Customer Table with Integrity Constraints  a) Create a Customers table with constraints: CustomerID (PRIMARY KEY), Email (UNIQUE), Age (CHECK Age > 18).  CREATE TABLE Customers (  CustomerID INT PRIMARY KEY,  CustomerName VARCHAR(100),  Email VARCHAR(100) UNIQUE,  Age INT CHECK (Age > 18),  Country VARCHAR(50) DEFAULT 'India'  );  b) Insert a valid customer record and verify that the default country is assigned if not explicitly provided.  -- Insert a valid customer (without specifying Country)  INSERT INTO Customers (CustomerID, CustomerName, Email, Age)  VALUES (1, 'John Doe', 'john.doe@example.com', 28);  -- Verify insertion  SELECT \* FROM Customers;  c) Attempt to insert a customer with an age of 16 and observe the CHECK constraint violation.  -- This will fail  INSERT INTO Customers (CustomerID, CustomerName, Email, Age)  VALUES (2, 'Jane Doe', 'jane.doe@example.com', 16);  d) Try inserting two customers with the same email ID and observe the UNIQUE constraint violation.  -- First valid insert  INSERT INTO Customers (CustomerID, CustomerName, Email, Age)  VALUES (2, 'Alice Smith', 'alice.smith@example.com', 30);  -- Second insert with duplicate email - will fail  INSERT INTO Customers (CustomerID, CustomerName, Email, Age)  VALUES (3, 'Bob Brown', 'alice.smith@example.com', 35);  e) Retrieve all customers who are older than 25 and belong to a country other than 'India'.  SELECT \*  FROM Customers  WHERE Age > 25  AND Country != 'India';  Which recent tool or technology have you studied for database management, and can you briefly explain its key features and why it is used in the industry? |
| 4 | 4) Create a Table with Constraints  a) Create an EmployeeDetails table with EmployeeID as the PRIMARY KEY and DepartmentID as a FOREIGN KEY referencing a Department table.  -- Create Department table  CREATE TABLE Department (  DepartmentID INT PRIMARY KEY,  DepartmentName VARCHAR(100)  );  -- Create EmployeeDetails table  CREATE TABLE EmployeeDetails (  EmployeeID INT PRIMARY KEY,  EmployeeName VARCHAR(100),  DepartmentID INT,  Salary DECIMAL(10,2),  FOREIGN KEY (DepartmentID) REFERENCES Department(DepartmentID)  );  b) Insert a valid employee record with an existing DepartmentID, then attempt to insert an employee with a non-existent DepartmentID and observe the constraint violation.  -- Insert a valid department first  INSERT INTO Department (DepartmentID, DepartmentName) VALUES  (1, 'HR'),  (2, 'Finance');  -- Insert a valid employee (DepartmentID exists)  INSERT INTO EmployeeDetails (EmployeeID, EmployeeName, DepartmentID, Salary)  VALUES (101, 'Alice Johnson', 1, 50000.00);  -- Attempt to insert employee with non-existent DepartmentID (will fail)  INSERT INTO EmployeeDetails (EmployeeID, EmployeeName, DepartmentID, Salary)  VALUES (102, 'Bob Smith', 5, 55000.00);  c) Insert an employee with a duplicate EmployeeID and check how the primary key constraint prevents duplicate entries.  -- Attempt to insert duplicate EmployeeID (will fail)  INSERT INTO EmployeeDetails (EmployeeID, EmployeeName, DepartmentID, Salary)  VALUES (101, 'Charlie Brown', 2, 60000.00);  d) Modify the Salary column to have a UNIQUE constraint and attempt to insert two employees with the same salary to test the constraint.  -- Add UNIQUE constraint to Salary  ALTER TABLE EmployeeDetails  ADD CONSTRAINT Unique\_Salary UNIQUE (Salary);  -- Insert employee with a unique salary (will succeed)  INSERT INTO EmployeeDetails (EmployeeID, EmployeeName, DepartmentID, Salary)  VALUES (103, 'David Lee', 2, 60000.00);  -- Try inserting employee with duplicate salary (will fail)  INSERT INTO EmployeeDetails (EmployeeID, EmployeeName, DepartmentID, Salary)  VALUES (104, 'Emma Watson', 1, 60000.00);  e) Write a query to delete an employee from EmployeeDetails and ensure that the deletion does not violate any referential integrity constraints.  -- Delete an employee  DELETE FROM EmployeeDetails  WHERE EmployeeID = 103;  Which recent tool or technology have you studied for database management, and can you briefly explain its key features and why it is used in the industry? |
| 5 | 5) Create an Employee Table with Various Columns  a) Create a table Employee with attributes: EmployeeID (INT, PRIMARY KEY), Name (VARCHAR), Salary (DECIMAL), JoiningDate (DATE), and ActiveStatus (BOOLEAN).  CREATE TABLE Employee (  EmployeeID INT PRIMARY KEY,  Name VARCHAR(100),  Salary DECIMAL(10,2),  JoiningDate DATE,  ActiveStatus BOOLEAN  );  b) Insert five sample employee records and ensure each employee has a unique EmployeeID.  INSERT INTO Employee (EmployeeID, Name, Salary, JoiningDate, ActiveStatus) VALUES  (1, 'Amit Sharma', 60000.00, '2022-05-15', TRUE),  (2, 'Neha Verma', 55000.00, '2023-03-01', TRUE),  (3, 'Rohan Mehta', 70000.00, '2021-11-20', TRUE),  (4, 'Sneha Kapoor', 50000.00, '2024-01-10', FALSE),  (5, 'Karan Malhotra', 65000.00, '2022-08-05', TRUE);  c) Write a query to find all employees who joined before January 1, 2023.  SELECT \*  FROM Employee  WHERE JoiningDate < '2023-01-01';  d) Update the salary of an employee named ‘Amit Sharma’ by 10% and display the updated record.  -- Update salary by 10%  UPDATE Employee  SET Salary = Salary \* 1.10  WHERE Name = 'Amit Sharma';  -- Display updated record  SELECT \*  FROM Employee  WHERE Name = 'Amit Sharma';  e) Retrieve all employees who are currently active (ActiveStatus = TRUE).  SELECT \*  FROM Employee  WHERE ActiveStatus = TRUE;  Which recent tool or technology have you studied for database management, and can you briefly explain its key features and why it is used in the industry? |
| 6 | 6) Aggregate Functions (on a single table: Create a Sales table with columns: SaleID, ProductID, ProductName, Quantity, Discount, SaleAmount, and SaleDate.)  CREATE TABLE Sales (  SaleID INT PRIMARY KEY,  ProductID INT,  ProductName VARCHAR(100),  Quantity INT,  Discount DECIMAL(5,2),  SaleAmount DECIMAL(10,2),  SaleDate DATE  );  INSERT INTO Sales (SaleID, ProductID, ProductName, Quantity, Discount, SaleAmount, SaleDate) VALUES  (1, 101, 'Laptop', 2, 10.00, 1800.00, '2025-02-05'),  (2, 102, 'Smartphone', 1, 5.00, 550.00, '2025-02-10'),  (3, 103, 'Monitor', 3, 7.50, 750.00, '2025-01-15'),  (4, 101, 'Laptop', 1, 8.00, 950.00, '2025-02-20'),  (5, 104, 'Headphones', 4, 12.00, 400.00, '2025-03-02');   1. From the Sales table, calculate the total sales amount (SUM) generated in the month of February 2025.   SELECT SUM(SaleAmount) AS TotalSalesFebruary2025  FROM Sales  WHERE MONTH(SaleDate) = 2 AND YEAR(SaleDate) = 2025;   1. Find the average (AVG) billing amount from the Sales table to assess customer spending behavior.   SELECT AVG(SaleAmount) AS AverageBillingAmount  FROM Sales;   1. Identify the minimum (MIN) quantity of products sold in any transaction using the Sales table.   SELECT MIN(Quantity) AS MinimumQuantitySold  FROM Sales;   1. Determine the highest (MAX) discount applied on any sale using the Sales table.   SELECT MAX(Discount) AS MaximumDiscountApplied  FROM Sales;   1. Use the COUNT function to find how many transactions were recorded in the Sales table for the product “Laptop”.   SELECT COUNT(\*) AS LaptopTransactionCount  FROM Sales  WHERE ProductName = 'Laptop';  Which recent tool or technology have you studied for database management, and can you briefly explain its key features and why it is used in the industry? |
| 7 | 7) Constraints (on a single table: Employees)  a) Create the Employees table with EmployeeID as PRIMARY KEY, Email as UNIQUE, and Salary with a CHECK (Salary > 10000) constraint.  CREATE TABLE Employees (  EmployeeID INT PRIMARY KEY,  Name VARCHAR(100) NOT NULL,  Email VARCHAR(100) UNIQUE,  Salary DECIMAL(10,2) CHECK (Salary > 10000),  Status VARCHAR(20) DEFAULT 'Active'  );  b) Add a NOT NULL constraint on the Name column in the Employees table and try inserting a record without the name.  -- This will fail because Name cannot be NULL  INSERT INTO Employees (EmployeeID, Email, Salary)  VALUES (1, 'john.doe@example.com', 12000.00);  c) Add a DEFAULT value ‘Active’ to the Status column in Employees, and insert a record without specifying the status to verify the default.  -- Status should default to 'Active'  INSERT INTO Employees (EmployeeID, Name, Email, Salary)  VALUES (2, 'Neha Sharma', 'neha.sharma@example.com', 15000.00);  -- Verify  SELECT \* FROM Employees;  d) Insert a record into Employees where Salary is less than 10000 to test the CHECK constraint.  -- This will fail because salary is less than 10000  INSERT INTO Employees (EmployeeID, Name, Email, Salary)  VALUES (3, 'Amit Verma', 'amit.verma@example.com', 9000.00);   1. Try inserting two employees with the same Email ID to verify the enforcement of the UNIQUE constraint.   -- First insert (valid)  INSERT INTO Employees (EmployeeID, Name, Email, Salary)  VALUES (4, 'Rohit Singh', 'rohit.singh@example.com', 16000.00);  -- Second insert with duplicate email (will fail)  INSERT INTO Employees (EmployeeID, Name, Email, Salary)  VALUES (5, 'Sneha Gupta', 'rohit.singh@example.com', 17000.00);  Which recent tool or technology have you studied for database management, and can you briefly explain its key features and why it is used in the industry? |
| 8 | 8) DDL and DML Commands  a) Use DDL commands to create a Library database and define a Books table with fields: BookID, Title, Author, Genre, and Price.  -- Create Library database  CREATE DATABASE Library;  -- Use Library database  USE Library;  -- Create Books table  CREATE TABLE Books (  BookID INT PRIMARY KEY,  Title VARCHAR(255),  Author VARCHAR(255),  Genre VARCHAR(100),  Price DECIMAL(10,2)  );  b) Insert at least five sample records into the Books table using INSERT (DML) and verify them using a SELECT query.  -- Insert sample records  INSERT INTO Books (BookID, Title, Author, Genre, Price) VALUES  (1, 'Deep Learning', 'Ian Goodfellow', 'Technology', 500.00),  (2, 'To Kill a Mockingbird', 'Harper Lee', 'Fiction', 300.00),  (3, 'Clean Code', 'Robert C. Martin', 'Programming', 450.00),  (4, 'Ancient Computing', 'John Doe', 'Outdated Technology', 100.00),  (5, 'The Great Gatsby', 'F. Scott Fitzgerald', 'Fiction', 250.00);  -- Verify inserted records  SELECT \* FROM Books;  c) A new column PublicationYear needs to be added. Use ALTER TABLE to modify the existing table structure.  ALTER TABLE Books  ADD PublicationYear INT;  d) Update the price of all books published before 2020 by increasing 10% using the UPDATE statement.  -- First, update some sample years  UPDATE Books SET PublicationYear = 2018 WHERE BookID = 1;  UPDATE Books SET PublicationYear = 1960 WHERE BookID = 2;  UPDATE Books SET PublicationYear = 2008 WHERE BookID = 3;  UPDATE Books SET PublicationYear = 1995 WHERE BookID = 4;  UPDATE Books SET PublicationYear = 1925 WHERE BookID = 5;  -- Now update price where PublicationYear < 2020  UPDATE Books  SET Price = Price \* 1.10  WHERE PublicationYear < 2020;  -- Verify  SELECT \* FROM Books;  e) Use DELETE to remove all books where the genre is ‘Outdated Technology’ and validate the change with a SELECT query.  DELETE FROM Books  WHERE Genre = 'Outdated Technology';  -- Verify remaining records  SELECT \* FROM Books;  Which recent tool or technology have you studied for database management, and can you briefly explain its key features and why it is used in the industry? |
| 9 | 9) DDL and DML Commands (on a single table: Books)  a) Create a table Books using DDL with fields: BookID, Title, Author, Price, and StockAvailable.  CREATE TABLE Books (  BookID INT PRIMARY KEY,  Title VARCHAR(255),  Author VARCHAR(255),  Price DECIMAL(10,2),  StockAvailable INT  );  b) Insert 5 book records into the Books table using the INSERT command.  INSERT INTO Books (BookID, Title, Author, Price, StockAvailable) VALUES  (1, 'Atomic Habits', 'James Clear', 450.00, 10),  (2, 'The Alchemist', 'Paulo Coelho', 350.00, 0),  (3, 'Sapiens', 'Yuval Noah Harari', 600.00, 5),  (4, '1984', 'George Orwell', 400.00, 3),  (5, 'Thinking, Fast and Slow', 'Daniel Kahneman', 550.00, 0);  -- Verify inserted data  SELECT \* FROM Books;  c) Modify the structure of Books table by adding a new column Genre using the ALTER TABLE command.  ALTER TABLE Books  ADD Genre VARCHAR(100);  d) Use the UPDATE command to increase the price of all books by RS 50 in the Books table.  UPDATE Books  SET Price = Price + 50;  -- Verify the updated prices  SELECT \* FROM Books;  e) Delete all records from the Books table where StockAvailable is 0 using the DELETE command.  DELETE FROM Books  WHERE StockAvailable = 0;  -- Verify remaining records  SELECT \* FROM Books;  Which recent tool or technology have you studied for database management, and can you briefly explain its key features and why it is used in the industry? |
| 10 | 10) Analyze Sales Performance Using Aggregate Functions Create a Sales table with columns: SaleID, ProductID, ProductName, Quantity, Discount, SaleAmount, and SalesPerson.)  CREATE TABLE Sales (  SaleID INT PRIMARY KEY,  ProductID INT,  ProductName VARCHAR(255),  Quantity INT,  Discount DECIMAL(5,2),  SaleAmount DECIMAL(10,2),  SaleDate DATE,  SalesPerson VARCHAR(100)  );  INSERT INTO Sales (SaleID, ProductID, ProductName, Quantity, Discount, SaleAmount, SaleDate, SalesPerson) VALUES  (1, 101, 'Laptop', 2, 5.00, 1500.00, '2025-02-10', 'Anil Kumar'),  (2, 102, 'Smartphone', 5, 10.00, 2500.00, '2025-03-05', 'Pooja Singh'),  (3, 103, 'Tablet', 3, 7.50, 1200.00, '2025-03-12', 'Anil Kumar'),  (4, 104, 'Smartwatch', 1, 15.00, 300.00, '2025-02-18', 'Rohit Mehra'),  (5, 105, 'Earbuds', 8, 3.00, 800.00, '2025-03-22', 'Pooja Singh');   1. Calculate the total quantity of products sold across all transactions in the Sales table.   SELECT SUM(Quantity) AS TotalQuantitySold  FROM Sales;   1. Find the average sale amount for transactions made in March 2025.   SELECT AVG(SaleAmount) AS AvgSaleAmountMarch  FROM Sales  WHERE MONTH(SaleDate) = 3 AND YEAR(SaleDate) = 2025;   1. Identify the product with the minimum sale quantity from the Sales table.   SELECT ProductName, Quantity  FROM Sales  WHERE Quantity = (SELECT MIN(Quantity) FROM Sales);   1. Determine the maximum discount offered in February 2025.   SELECT MAX(Discount) AS MaxDiscountFebruary  FROM Sales  WHERE MONTH(SaleDate) = 2 AND YEAR(SaleDate) = 2025;   1. Count how many sales were made by each salesperson using GROUP BY SalesPerson.   SELECT SalesPerson, COUNT(\*) AS NumberOfSales  FROM Sales  GROUP BY SalesPerson;  Which recent tool or technology have you studied for database management, and can you briefly explain its key features and why it is used in the industry? |