Q1. consider a table called Students which contains student_id, first_name, last_name, department, and age as Columns. Create a simple select stored procedure that will select and display student records based on a specified department.

```
CREATE TABLE Students (
   student_id INT PRIMARY KEY,
   first_name VARCHAR(50),
   last_name VARCHAR(50),
   department VARCHAR(50),
   age INT
);

INSERT INTO Students (student_id, first_name, last_name, department, age)
VALUES
   (1, 'John', 'Doe', 'Computer Science', 21),
   (2, 'Jane', 'Smith', 'Engineering', 23),
   (3, 'Alice', 'Johnson', 'Mathematics', 20),
   (4, 'Bob', 'Williams', 'Physics', 22),
   (5, 'Emma', 'Brown', 'Biology', 19);

delimiter //
CREATE PROCEDURE GetStudentsByDepartment(
IN p_department VARCHAR(50)
)
BEGIN
SELECT * from students
where department = p_department;
end//
delimiter ;

CALL GetStudentsByDepartment('Computer Science');
```

Q2. Create a database, create a table, demonstrate all DDL commands. Set primary key after creation of table.

```
CREATE DATABASE IF NOT EXISTS my_database;

USE my_database;

CREATE TABLE IF NOT EXISTS students_details (
   student_id INT AUTO_INCREMENT,
   first_name VARCHAR(50),
   last_name VARCHAR(50),
   department VARCHAR(50),
   age INT,
   PRIMARY KEY (student_id) -- Define the primary key after table creation
);

ALTER TABLE students_details
ADD email VARCHAR(100);

ALTER TABLE students_details
MODIFY COLUMN age INT UNSIGNED;

ALTER TABLE students_details
DROP COLUMN email;

ALTER TABLE students_details
RENAME TO student_information;

DROP TABLE IF EXISTS student_information;
```

Q3. Create a table to store employee details. Define input parameters within the CREATE PROCEDURE statement and pass them in the CALL statement.

```
CREATE DATABASE IF NOT EXISTS company_database;

USE company_database;

CREATE TABLE IF NOT EXISTS employee_details (
   employee_id INT AUTO_INCREMENT PRIMARY KEY,
   first_name VARCHAR(50),
   last_name VARCHAR(50),
   department VARCHAR(50),
   age INT
);

DELIMITER //

CREATE PROCEDURE InsertEmployeeDetails(
   IN emp_first_name VARCHAR(50),
   IN emp_last_name VARCHAR(50),
   IN emp_department VARCHAR(50),
   IN emp_age INT
)
BEGIN
   INSERT INTO employee_details (first_name, last_name, department, age)
   VALUES (emp_first_name, emp_last_name, emp_department, emp_age);
END //

DELIMITER ;

CALL InsertEmployeeDetails('John', 'Doe', 'IT', 30);

select * from employee_details;
```

Q4. create a row-level trigger for the STUDENT table that would get executed by the DML statement like UPDATE, INSERT or DELETE on that table. The trigger will compute and show the age difference
between current and previous values.

```
CREATE TABLE IF NOT EXISTS age_difference_log (
   student_id INT,
   old_age INT,
   new_age INT,
   age_difference INT,
   change_date TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);

DELIMITER //

CREATE TRIGGER student_age_difference
BEFORE UPDATE ON students
FOR EACH ROW
BEGIN
   DECLARE prev_age INT;
   DECLARE age_difference INT;

   SELECT age INTO prev_age FROM students WHERE student_id = OLD.student_id;

   SET age_difference = NEW.age - prev_age;

   INSERT INTO age_difference_log (student_id, old_age, new_age, age_difference)
   VALUES (OLD.student_id, prev_age, NEW.age, age_difference);
END //

DELIMITER ;

select * from age_difference_log;
```

Q5.
A) Write a SQL statement to change salary of employee to 8000 whose ID is 105, if the existing salary is less than 5000.
B) change job_title of employee which ID is 118, to SH_CLERK if the employee belongs to department, which ID is 30 and the existing job title does not start with SH.

```
CREATE TABLE Employee (
   E_id INT PRIMARY KEY,
   E_name VARCHAR(50),
   E_dept INT,
   E_salary DECIMAL(10, 2),
   job_title VARCHAR(50)
);

INSERT INTO Employee (E_id, E_name, E_dept, E_salary, job_title)
VALUES
   (105, 'John Doe', 40, 4500.00, 'Manager'),
   (118, 'Jane Smith', 30, 5200.00, 'Clerk'),
   (119, 'Mike Johnson', 110, 6000.00, 'SH_MANAGER'),
   (120, 'Emily Davis', 30, 4800.00, 'Sales Assistant');

select * from Employee;


UPDATE Employee
SET E_salary = 8000
WHERE E_id = 105 AND E_salary < 5000;

select * from Employee;

UPDATE Employee
SET job_title = 'SH_CLERK'
WHERE E_id = 118 AND E_dept = 30 AND NOT job_title LIKE 'SH%';

select * from Employee;
```

Q6. Write a SQL statement to increase the salary of employees under the department 40, 90 and 110 according to the company rules that, salary will be increased by 25% for the department 40, 15% for department 90 and 10% for the department 110 and the rest of the departments will remain same

```
CREATE TABLE Employee (
   E_id INT PRIMARY KEY,
   E_name VARCHAR(50),
   E_dept INT,
   E_salary DECIMAL(10, 2)
);
INSERT INTO Employee (E_id, E_name, E_dept, E_salary)
VALUES
   (1, 'John Doe', 40, 50000.00),
   (2, 'Jane Smith', 90, 60000.00),
   (3, 'Mike Johnson', 110, 55000.00),
   (4, 'Emily Davis', 30, 45000.00),
   (5, 'Chris Brown', 40, 48000.00),
   (6, 'Emma Watson', 90, 62000.00);
select * from Employee;

UPDATE Employee
SET E_salary =
   CASE
      WHEN E_dept = 40 THEN E_salary * 1.25
      WHEN E_dept = 90 THEN E_salary * 1.15
      WHEN E_dept = 110 THEN E_salary * 1.10
      ELSE E_salary
   END
WHERE E_dept IN (40, 90, 110);
select * from Employee;
```

Q.7 Create table EMPLOYEE with attributes E_id, E_name, E_dept, E_salary, E_pno, E_city. Create view having E_id, E_name, E_dept, E_salary. create another table Employee details with some attributes and create another view from both the tables

```
CREATE TABLE EMPLOYEE (
    E_id INT PRIMARY KEY,
    E_name VARCHAR(50),
    E_dept VARCHAR(50),
    E_salary DECIMAL(10, 2),
    E_pno INT,
    E_city VARCHAR(50)
);

INSERT INTO EMPLOYEE (E_id, E_name, E_dept, E_salary, E_pno, E_city)
VALUES
    (1, 'John Doe', 'Engineering', 60000.00, 101, 'New York'),
    (2, 'Jane Smith', 'HR', 50000.00, 102, 'Los Angeles'),
    (3, 'Mike Johnson', 'Marketing', 55000.00, 103, 'Chicago');

CREATE VIEW EmployeeView AS
SELECT E_id, E_name, E_dept, E_salary
FROM EMPLOYEE;

CREATE TABLE EmployeeDetails (
    E_id INT PRIMARY KEY,
    E_age INT,
    E_address VARCHAR(100),
    E_phone VARCHAR(15)
);

INSERT INTO EmployeeDetails (E_id, E_age, E_address, E_phone)
VALUES
    (1, 30, '123 Main St, New York', '555-1234'),
    (2, 35, '456 Elm St, Los Angeles', '555-5678'),
    (3, 28, '789 Oak St, Chicago', '555-9012');

CREATE VIEW CombinedEmployee AS
SELECT E.E_id, E.E_name, E.E_dept, E.E_salary, ED.E_age, ED.E_address, ED.E_phone
FROM EMPLOYEE E
JOIN EmployeeDetails ED ON E.E_id = ED.E_id;

SELECT * FROM EmployeeView;

SELECT * FROM CombinedEmployee;
```

Q8.

Display name, credit_rating, sales_rep_id from S_customer table of those customer who either satisfies the condition that credit_rating is greater than 5 out of 10 and sales_rep_id is equal to 4232. Demonstrate pattern matching and logical operator.

```
CREATE TABLE S_customer ( name VARCHAR(50), credit_rating INT, sales_rep_id INT);

INSERT INTO S_customer VALUES('Arun', 7, 4231), ('Atharva', 6, 4231);

SELECT name, credit_rating, sales_rep_id
FROM S_customer
WHERE (credit_rating > 5 AND sales_rep_id = 4232);
```

Q9. Display the id, name and phone number of the customer

1) Whose id falls in the range 303 to 306

2)Whose id is greater than 300 and customer belongs to Pune

3)display the id, names of employee whose names contains fourth and fifth leters are 'sh' followed by anything and also belogs to pune city.

```
CREATE TABLE customers (
id INT PRIMARY KEY,
name VARCHAR(100),
phone_number VARCHAR(15),
city VARCHAR(50)
);

CREATE TABLE employees (
id INT PRIMARY KEY,
name VARCHAR(100),
city VARCHAR(50)
);



INSERT INTO customers (id, name, phone_number, city)
VALUES
 (301, 'John Doe', '123-456-7890', 'Pune'),
 (302, 'Jane Smith', '987-654-3210', 'Mumbai'),
 (303, 'Alice Wonderland', '555-123-4567', 'Pune'),
 (304, 'Bob Builder', '777-888-9999', 'Pune'),
 (305, 'Charlie Chaplin', '444-555-6666', 'Pune'),
 (306, 'David Beckham', '111-222-3333', 'Delhi');

INSERT INTO employees (id, name, city)
VALUES
 (101, 'Ashley Johnson', 'Pune'),
 (102, 'Michelle Sharma', 'Mumbai'),
 (103, 'Joshua Smith', 'Pune'),

 (104, 'Nisha Shah', 'Pune'),

 (105, 'Rajesh Patel', 'Mumbai'),

 (106, 'Rakesh Kumar', 'Pune');


SELECT * FROM customers

WHERE id BETWEEN 303 AND 306;


SELECT * FROM customers

WHERE id > 300 AND city = 'Pune';


SELECT * FROM employees

WHERE name LIKE '__sh%' AND city = 'Pune';
```

Q10. Create a student table, showing the records of the students having

1) Highest marks in math

2) Lowest attendance

3) Total number of students

4) Average marks of dbms

```
CREATE TABLE student (
    student_id INT PRIMARY KEY,
    student_name VARCHAR(100),
    math_marks INT,
    attendance INT,
    dbms_marks INT
);

INSERT INTO student (student_id, student_name, math_marks, attendance, dbms_marks)
VALUES
    (1, 'John Doe', 95, 90, 85),
    (2, 'Jane Smith', 85, 92, 88),
    (3, 'Mike Johnson', 90, 85, 90),
    (4, 'Emily Davis', 88, 95, 92),
    (5, 'Chris Brown', 92, 80, 87);

SELECT *
FROM student
WHERE math_marks = (SELECT MAX(math_marks) FROM student);


SELECT *
FROM student
WHERE attendance = (SELECT MIN(attendance) FROM student);
SELECT COUNT(*) AS total_students
FROM student;

SELECT AVG(dbms_marks) AS avg_dbms_marks
FROM student;
```

Q11. Consider the table EmployeeDetails with the following attributes:

EmpId	FullName	ManagerId	DateOfJoining	City
121	SnowJohn	321	01/31/2019	Toronto
321	WhiteWalter	986	01/30/2020	California
421	RanaKuldeep	876	27/11/2021	DelhiNew

A) Write an SQL query to fetch the EmpId and FullName of all the employees working under the Manager with id - '986'.

B) Write an SQL query to fetch the FullName of employees where the name starts with "hn" and ends with any sequence of characters.

C) Write an SQL query to fetch the employee's full names and replace the space with '-'.

```
CREATE TABLE EmployeeDetails (
   EmpId INT PRIMARY KEY,
   FullName VARCHAR(100),
   ManagerId INT,
   DateOfJoining DATE,
   City VARCHAR(50)
);

INSERT INTO EmployeeDetails (EmpId, FullName, ManagerId, DateOfJoining, City)
VALUES
   (121, 'Snow John', 321, '2019-01-31', 'Toronto'),
   (321, 'White Walter', 986, '2020-01-30', 'California'),
   (421, 'Rana Kuldeep', 876, '2021-11-27', 'DelhiNew');

SELECT EmpId, FullName
FROM EmployeeDetails
WHERE ManagerId = '986';

SELECT FullName
FROM EmployeeDetails
WHERE FullName LIKE 'hn%';

SELECT REPLACE(FullName, ' ', '_') AS ModifiedFullName
FROM EmployeeDetails;
```