Introduction To Databases DPP Assignment

1. Introduction to SQL and Basic Queries

Objective: Create a database, table, insert data, and retrieve records.

Task 1: Create database `company_db`.

```
CREATE DATABASE company_db;
USE company_db; -- Select the database
```

Task 2: Create `employees` table.

```
CREATE TABLE employees (
    id INT PRIMARY KEY,
    first_name VARCHAR(50),
    last_name VARCHAR(50),
    department VARCHAR(50),
    salary INT
);
```

Task 3: Insert 5 employee records.

```
INSERT INTO employees VALUES
(1, 'John', 'Doe', 'Sales', 45000),
(2, 'Jane', 'Smith', 'HR', 52000),
(3, 'Amit', 'Sharma', 'IT', 65000),
(4, 'Neha', 'Verma', 'Sales', 58000),
(5, 'Rahul', 'Mehta', 'Finance', 60000);
```

Task 4: Retrieve all employee records.

```
SELECT * FROM employees; -- Fetch all rows
```

2. Filtering Data Using WHERE Clause

Objective: Apply filtering using conditions.

Task 1: Employees from Sales department.

```
SELECT * FROM employees
WHERE department = 'Sales';
```

Task 2: Employees with salary greater than 50000.

```
SELECT * FROM employees
WHERE salary > 50000;
```

Task 3: Sales employees with salary greater than 50000.

```
SELECT * FROM employees
WHERE department = 'Sales' AND salary > 50000;
```

Task 4: Unique departments.

```
SELECT DISTINCT department FROM employees;
```

3. Modifying Data (INSERT, UPDATE, DELETE)

Objective: Modify records using DML operations.

Task 1: Insert 3 more employees.

```
INSERT INTO employees VALUES
(6, 'Karan', 'Patel', 'IT', 70000),
(7, 'Sneha', 'Iyer', 'HR', 48000),
(8, 'Vikas', 'Singh', 'Sales', 55000);
```

Task 2: Update salary of employee with id = 2.

```
UPDATE employees
SET salary = 60000
WHERE id = 2;
```

Task 3: Delete employee with id = 1.

```
DELETE FROM employees
WHERE id = 1;
```

Task 4: Verify changes.

```
SELECT * FROM employees;
```

4. Using Constraints

Objective: Ensure data integrity using constraints.

Task 1: Create employees_v2 with constraints.

```
CREATE TABLE employees_v2 (
    id INT PRIMARY KEY,
    name VARCHAR(50) NOT NULL,
    email VARCHAR(100) UNIQUE,
    department VARCHAR(50) NOT NULL,
    salary INT CHECK (salary > 0)
);
```

Task 2: Insert data and test UNIQUE constraint.

```
INSERT INTO employees_v2 VALUES
(1, 'Rohit', 'rohit@gmail.com', 'IT', 60000),
(2, 'Rohit2', 'rohit@gmail.com', 'HR', 55000); -- This will fail due to
duplicate email
```

Expected Output Tables

Expected output after executing `SELECT * FROM employees;`

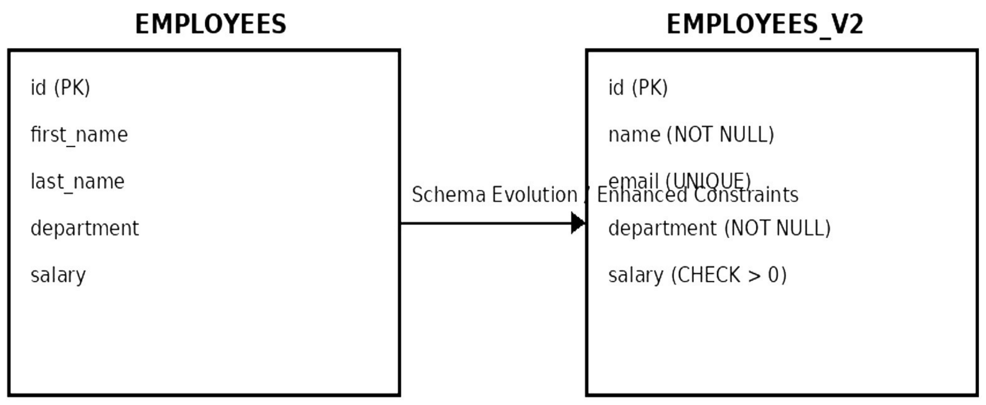
id	first_name	last_name	department	salary
2	Jane	Smith	HR	60000
3	Amit	Sharma	IT	65000
4	Neha	Verma	Sales	58000
5	Rahul	Mehta	Finance	60000
6	Karan	Patel	IT	70000
7	Sneha	Iyer	HR	48000
8	Vikas	Singh	Sales	55000

Expected output for unique departments query:

department
HR
IT
Sales
Finance

ER Diagram

The following ER diagram represents the structure of the `employees` table.



Schema Explanation

• employees: Stores employee master data.

- id: Primary Key, uniquely identifies each employee.
- first_name, last_name: Store employee names.
- department: Represents the department to which an employee belongs.
- salary: Stores employee salary and is used for filtering and constraints.

• employees_v2: Enhanced version of employees table with constraints to ensure data integrity.

- NOT NULL ensures mandatory fields.
- UNIQUE prevents duplicate email entries.
- CHECK enforces valid salary values.