Database and

Management System

Lab

Lab Experiment – 10

Name: Shubhojit Mitra

Roll No: R2142230793

Sap_ID: 500120225

B.Tech CSE, SEM-III, B-1

Title: Create the following views in SQL on the COMPANY database schema presented in Experiment 2

1. A view that has the department name, manager name, and manager salary for every department.

```
CREATE VIEW dept_manager_view AS
SELECT D.Dname AS department_name, E.Fname AS manager_name, E.Salary AS
manager_salary
FROM department D
JOIN employee E ON D.Mgr_ssn = E.Ssn;
```

```
+-----+
| department_name | manager_name | manager_salary|
+-----+
| Research | Franklin | 40000.00 |
| Administration | Jennifer | 43000.00 |
+-----+
```

2. A view that has the employee name, supervisor name, and employee salary for each employee who works in the 'Research' department

```
CREATE VIEW research_employee_view AS

SELECT E1.Fname AS employee_name, E2.Fname AS supervisor_name, E1.Salary AS

employee_salary

FROM employee E1

JOIN employee E2 ON E1.Super_ssn = E2.Ssn

JOIN department D ON E1.Dno = D.Dnumber

WHERE D.Dname = 'Research';
```

3. A view that has the project name, controlling department name, number of employees, and total hours worked per week on the project for each project.

```
CREATE VIEW project_summary_view AS

SELECT P.Pname AS project_name, D.Dname AS controlling_department,

COUNT(DISTINCT W.Essn) AS num_employees, SUM(W.Hours) AS total_hours

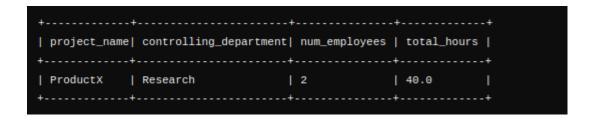
FROM project P

JOIN department D ON P.Dnum = D.Dnumber

JOIN works_on W ON P.Pnumber = W.Pno

GROUP BY P.Pname, D.Dname;
```

4. A view that has the project name, controlling department name, number of employees, and total hours worked per week on the project for each project with more than one employee working on it.



Database and

Management System

Lab

Lab Experiment – 11

Name: Shubhojit Mitra Roll No: R2142230793 Sap_ID: 500120225 Title: To understand the concepts of Index.

Objective: Students will be able to implement the concept of index

1. Create EMPLOYEES Table and Insert Sample Data

```
CREATE TABLE EMPLOYEES (
    Employee_id CHAR(10) PRIMARY KEY,
    First_Name VARCHAR(30) NOT NULL,
    Last_Name VARCHAR(30) NOT NULL,
    DOB DATE,
    Salary DECIMAL(10, 2) NOT NULL,
    Department_id CHAR(10)
);

INSERT INTO EMPLOYEES (Employee_id, First_Name, Last_Name, DOB, Salary, Department_id) VALUES ('E001', 'John', 'Smith', '1980-01-10', 50000, 'D1'),
    ('E002', 'Jane', 'Doe', '1985-02-15', 60000, 'D2'),
    ('E003', 'Alice', 'Johnson', '1978-03-20', 55000, 'D1'),
    ('E004', 'Bob', 'Brown', '1990-04-25', 62000, 'D2'),
    ('E005', 'Charlie', 'Davis', '1982-05-30', 58000, 'D3'),
    ('E006', 'Eve', 'Martinez', '1995-06-10', 54000, 'D3');
```

2. Create an Index on Last_Name and Department_id

```
CREATE INDEX employee_idx ON EMPLOYEES (Last_Name, Department_id);
```

3. Find the ROWID and Create a Unique Index on Employee_id

```
SELECT Employee_id, ROWID FROM EMPLOYEES;
CREATE UNIQUE INDEX employee_unique_idx ON EMPLOYEES (Employee_id);
```

4. Create a Reverse Index on Employee_id

```
CREATE INDEX employee_reverse_idx ON EMPLOYEES (Employee_id) USING BTREE;
```

5. Create a Unique Composite Index on Employee_id to Check Duplicates

```
CREATE UNIQUE INDEX employee_composite_idx ON EMPLOYEES (Employee_id, Last_Name);
```

Create Function-Based Index for Case-Insensitive Search on Last_Name

```
CREATE INDEX last_name_upper_idx ON EMPLOYEES (UPPER(Last_Name));
```

7. Drop the Function-Based Index on Last_Name

DROP INDEX last_name_upper_idx ON EMPLOYEES;