Experiment 3

Student Name: Dushyant
UID: 23BCS11986
Branch: CSE
Section/Group: 23BCS_KRG-3/B
Semester: 5
Date of Performance:22/08/25
Subject Name: Advanced Database and Management System
Subject Code: 23CSP-333

1. Aim:

[EASY] Generate an Employee relation with only one attribute i.e, Emp_ID. Then, find the maximum Emp_ID, but excluding the duplicates.

[MEDIUM] Create Two Tables. Department (ID, name) and Employees (ID, name, Salary, deptID). Then output the highest earners from each department.

[HARD] Create two tables A and B with the attributes (EmpID, EmpName, Salary) and output the lowest salary of each employee across the two tables.

2. Tools Used: SQL Server Management Studio

3. Code:

```
-- EASY

CREATE TABLE TBL_EMflLOYEE (
    EMfl_ID INT
);

INSERT INTO TBL_EMflLOYEE
VALUES
(2), (4), (4), (6), (6), (7), (8), (8);


SELECT * FROM TBL_EMflLOYEE;

SELECT MAX(EMfl_ID)
FROM TBL_EMflLOYEE
WHERE EMfl_ID N
```

```
(SELECT  EMfl_ID  FROM TBL_EMflLOYEE GROUfl BY  EMfl_ID  HAVING
COUNT(EMfl_ID) = 1);

---------- MEDIUM ----------

CREATE TABLE department (
    id INT flRIMARY KEY,
    dept_name  VARCHAR(50)
);

-- Create Employee Table
CREATE TABLE employee (
    id INT,
    name VARCHAR(50),
    salary INT,
    department_id INT,
    FOREIGN KEY (department_id) REFERENCES department(id)
);


-- Insert into Department Table
XNSERT INTO department (id, dept_name) VALUES
(1, 'IT'),
(2, 'SALES');

-- Insert into Employee Table
INSERT INTO employee (id, name, salary, department_id)
VALUES
(1, "JOE", 70000, 1),
(2, "JIM", 90000, 1),
(3, 'HENRY', 80000, 2),
(4, "SAM", 60000, 2),
(5, "MAX", 90000, 1);


Select d.dept_name, e.name, e.salary
from
employee as e
inner join
department d
on e.department_id = d.id
where e.salary in
(select max(salary) from
employee group by department_id);
```

```
-- or


Select e.name, d.dept_name, e.salary
from
employee as e
inner join
department d
on e.department_id = d.id
where e.salary in
(Select max(salary) from
employee where department_id = e.department_id)
order by d.dept_name;


-- HARD: GIVEN TWO TABLES OUTPUT THEM AS FOLLOWS:

CREATE TABLE TBL_A (
    EMPL_ID INT PRIMARY KEY,
    E_NAME VARCHAR(20),
    SALARY INT
);

CREATE TABLE TBL_B (
    EMPL_ID INT PRIMARY KEY,
    E_NAME VARCHAR(20),
    SALARY INT
);

INSERT INTO TBL_A
VALUES
    (1, 'AA', 1000),
    (2, 'BB', 300);

INSERT INTO TBL_B
VALUES
    (2, 'BB', 400),
    (3, 'CC', 100);

/*
OUTPUT-
EMPL_ID E_NAME SALARY
```

```
1          AA          1000
2          BB          300 (MIN)
3          CC          100
*/

SELECT EMfl_ID, E_NAME, MIN(SALARY)
FROM(
SELECT * FROM TBL_A AS A
UNION ALL
SELECT * FROM TBL_B AS B
) AS RES
GROUfl BY EMfl_ID, E_NAME;
```

4. **Output:**

[EASY]

	EMP_ID
1	2
2	4
3	4
4	6
5	6
6	7
7	8
8	8

	(No column name)
1	7

[MEDIUM]

	dept_name	name	salary
1	IT	JIM	90000
2	IT	MAX	90000
3	SALES	HENRY	80000

[HARD]

	EMP_ID	E_NAME	(No column name)
1	1	AA	1000
2	2	BB	300
3	3	CC	100

5. Learning Outcomes:

- Understand the role of subqueries in simplifying complex SQL operations.
- Apply sub-queries in SELECT, WHERE, FROM clauses to retrieve specific data.
- Utilize sub-queries for filtering, aggregation, and conditional logic.
- Analyze query performance implications when using sub-queries versus joins.