DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

Experiment 3

Student Name: Harsh Kumar
UID: 23BCS11889
Branch: B.E. C.S.E.
Section/Group: KRG-2A
Semester: 5th
Date of Performance: 21st August 2025
Subject Name: ADBMS
Subject Code: 23CSP-333

1. Aim:

To understand and implement sub-queries (scalar, multi-valued, and correlated) in SQL for retrieving complex results from multiple conditions.

2. Objective:

- To create and insert records into the MyEmployees table.
- To practice scalar sub-queries (returning a single value) for comparisons.
- To practice multi-valued sub-queries using operators like IN.
- To apply sub-queries with aggregate functions (MAX, COUNT, etc.).
- To understand the use of sub-queries inside WHERE and HAVING clauses.

3. DBMS Script:

```
-- Create table MyEmployees
CREATE TABLE MyEmployees (
    EmpId INT PRIMARY KEY IDENTITY(1,1),
    EmpName VARCHAR(50),
    Gender VARCHAR(10),
    Salary INT,
    City VARCHAR(50),
    Dept_id INT
);
```

```
-- Insert records
INSERT INTO MyEmployees (EmpName, Gender, Salary, City, Dept_id)
VALUES
('Amit', 'Male', 50000, 'Delhi', 2),
('Priya', 'Female', 60000, 'Mumbai', 1),
('Rajesh', 'Male', 45000, 'Agra', 3),
('Sneha', 'Female', 55000, 'Delhi', 4),
('Anil', 'Male', 52000, 'Agra', 2),
('Sunita', 'Female', 48000, 'Mumbai', 1),
('Vijay', 'Male', 47000, 'Agra', 3),
('Ritu', 'Female', 62000, 'Mumbai', 2),
('Alok', 'Male', 51000, 'Delhi', 1),
('Neha', 'Female', 53000, 'Agra', 4),
('Simran', 'Female', 33000, 'Agra', 3);

-- Second highest salary
SELECT MAX(Salary) AS SecondHighestSalary
FROM MyEmployees
WHERE Salary < (SELECT MAX(Salary) FROM MyEmployees);

-- Scalar subquery example
-- (Assumes you already have a 'dept' table created with columns id, dept_name)
SELECT * FROM MyEmployees
WHERE Dept_id <> (SELECT id FROM dept WHERE dept_name = 'Accounts');

-- Multi-valued subquery
SELECT * FROM MyEmployees
```

```
WHERE EmpName IN (
  SELECT EmpName FROM MyEmployees WHERE Gender = 'Female'
);

-- Employee table example
CREATE TABLE employee (
  id INT
);

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

-- Largest unique employee ID (SQL Server style, no LIMIT)
SELECT TOP 1 id
FROM employee
GROUP BY id
HAVING COUNT(id) = 1
ORDER BY id DESC;
```

4. Output:

```
60000
Parse error near line 230: no such table: dept
2|Priya|Female|60000|Mumbai|1
4|Sneha|Female|55000|Delhi|4
6|Sunita|Female|48000|Mumbai|1
8|Ritu|Female|62000|Mumbai|2
10|Neha|Female|53000|Agra|4
11|Simran|Female|33000|Agra|3
7
Press any key to continue . . .
```

5. Learning Outcomes:

- Differentiate between **scalar sub-queries** and **multi-valued sub-queries**.
- Use **sub-queries with aggregate functions** like `MAX`, `COUNT`.
- Implement **nested queries** in `WHERE` and `HAVING` clauses.
- Retrieve results that are not directly available from a single query.
- Understand how **sub-queries help break down complex problems** into manageable steps.