University Institute of Engineering

Department of Computer Science & Engineering

# Experiment: 1.2

Student Name: Dushyant Singh Chauhan UID: 23BCS11986

Branch: Computer Science & Engineering Section/Group: KRG-3B

Semester: 5th Subject Code:23CSP-339

Subject Name: ADBMS

## 1. Aim of the practical:

Department-Course Subquery and Access Control

1. Design normalized tables for departments and the courses they offer, maintaining a foreign key relationship.
2. Insert five departments and at least ten courses across those departments.
3. Use a subquery to count the number of courses under each department.
4. Filter and retrieve only those departments that offer more than two courses.
5. Grant SELECT-only access on the courses table to a specific user.

Sample Output Description:

The result shows the names of departments which are associated with more than two courses in the system.

## 2. Tool Used: SQL Server Management Studio.

## 3. CODE:

```
CREATE TABLE Departments ( dept_id INT
    PRIMARY KEY, dept_name
    VARCHAR(100) NOT NULL
);


CREATE TABLE Courses ( course_id INT
    PRIMARY KEY, course_name
    VARCHAR(150) NOT NULL, dept_id INT,
    FOREIGN KEY (dept_id) REFERENCES Departments(dept_id)
);

INSERT INTO Departments (dept_id, dept_name) VALUES
(1, 'Computer Science'),
(2, 'Mathematics'),
(3, 'Physics'),
```

```
(4, 'English'),
(5, 'Biology');

INSERT INTO Courses (course_id, course_name, dept_id) VALUES
(101, 'Data Structures', 1),
(102, 'Operating Systems', 1),
(103, 'Algorithms', 1), (104, 'Calculus
I', 2),
(105, 'Linear Algebra', 2),
(106, 'Quantum Mechanics', 3),
(107, 'Classical Mechanics', 3),
(108, 'Modern Poetry', 4),
(109, 'Cell Biology', 5),
(110, 'Genetics', 5);

SELECT dept_name
FROM Departments
WHERE dept_id IN ( SELECT
        dept_id
        FROM Courses
        GROUP BY dept_id
        HAVING COUNT(course_id) > 2
);
```

## 4. LEARNING OUTCOMES:-

- Understand how to design normalized relational tables with foreign key constraints for real-world entities like departments and courses.
- Gain proficiency in inserting multiple records into related tables using the INSERT INTO statement.
- Learn how to use subqueries with GROUP BY and HAVING to aggregate data and apply conditional logic.
- Apply filtering logic to retrieve records from a parent table based on results from a subquery on a related child table.