DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

Assignment -1.2

Name: Shubh Rai
UID: 23BCS12916
Subject: DBMS
Semester: 5th
Section: KRG - 3B
Date of performance: 21th July 2025
Branch: BE - CSE

1.1 Medium problem:

1: AIM:

-- Medium-Level Problem
-- Problem Title: Department-Course Subquery and Access Control
-- Procedure (Step-by-Step):

-- Design normalized tables for departments and the courses they offer, maintaining a foreign key relationship.

-- Insert five departments and at least ten courses across those departments.

-- Use a subquery to count the number of courses under each department.

-- Filter and retrieve only those departments that offer more than two courses.

-- Grant SELECT-only access on the courses table to a specific user.

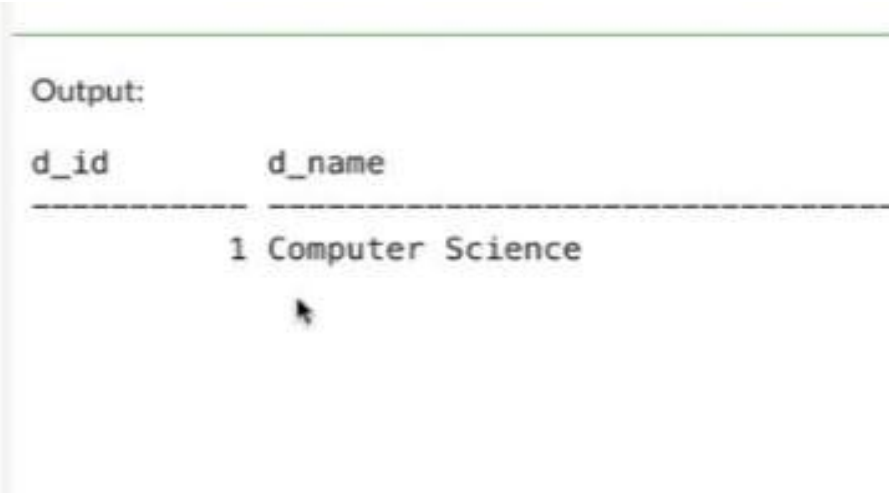
2:CODE

```
create table department (
    d_id int primary key,
    d_name varchar(MAX)
);

create table course(
    c_id int primary key,
    c_name varchar (MAX),
    d_id int ,
    foreign key (d_id) references department (d_id)
);

INSERT INTO department (d_id, d_name) VALUES
(1, 'Computer Science'),
(2, 'Mathematics'),
(3, 'Physics'),
(4, 'Biology'),
(5, 'History');

INSERT INTO course (c_id, c_name, d_id) VALUES
(101, 'Data Structures', 1),
(102, 'Operating Systems', 1),
(103, 'Database Systems', 1),
(104, 'Calculus', 2),
(105, 'Linear Algebra', 2),
(106, 'Classical Mechanics', 3),
(107, 'Quantum Physics', 3),
(108, 'Genetics', 4),
(109, 'Molecular Biology', 4),
(110, 'World War II', 5);


-- Use a subquery to count the number of courses under each
department.

select * from department
where d_id in
(
    select d_id from course
    group by d_id
```

```
    having count(c_id)>2
)
```

grant select on course to readOnly_User;

3: OUTPUT:



4: LEARNING OUTCOME:

By working through this SQL task, you should be able to:

1. **Understand and apply relational schema design:**
    - Define tables using `CREATE TABLE` with primary and foreign keys.
    - Establish **one-to-many relationships** (e.g., a department can offer many courses).
2. **Insert and manage data:**
    - Use `INSERT INTO` to populate tables with meaningful data.
3. **Use subqueries and aggregation:**
    - Write subqueries using `GROUP BY` and `HAVING` to filter

based on aggregate conditions.

- Understand how to retrieve departments based on course counts (e.g., departments with more than 2 courses).

4. **Implement data access control:**
    - Use the `GRANT` statement to give specific users permissions (e.g., read-only access).
5. **Interpret SQL results:**
    - Predict the result of a query based on data and understand the relational structure behind the results.