Experiment-1

Student Name: Manjot Singh
UID: 23BCS12549
Branch: CSE
Section/Group: KRG-2B
Semester: 5th
Date of Performance: 28/07/25
Subject Name: ADBMS
Subject Code: 23CSP-333

1. Aim:

This exercise focuses on managing departmental and course data within a database. You will learn to structure data efficiently, use advanced querying techniques like subqueries for aggregation, filter results based on specific criteria, and implement basic access control.

- 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.

Departments Table

department_id	department_name	location
1	Computer Science	Building A
2	Electrical Eng.	Building B
3	Mechanical Eng.	Building C
4	Civil Eng.	Building D
5	Humanities	Building E

Courses Table

course_id	course_name	credits	department_id
101	Data Structures	3	1
102	Algorithms	4	1
103	Operating Systems	3	1
104	Digital Logic Design	3	2
105	Microprocessors	4	2
106	Thermodynamics	3	3
107	Fluid Mechanics	3	3
108	Structural Analysis	4	4
109	Engineering Ethics	2	5
110	Technical Writing	3	5
111	Database Management	4	1
112	Communication Systems	3	2

2. Objective:

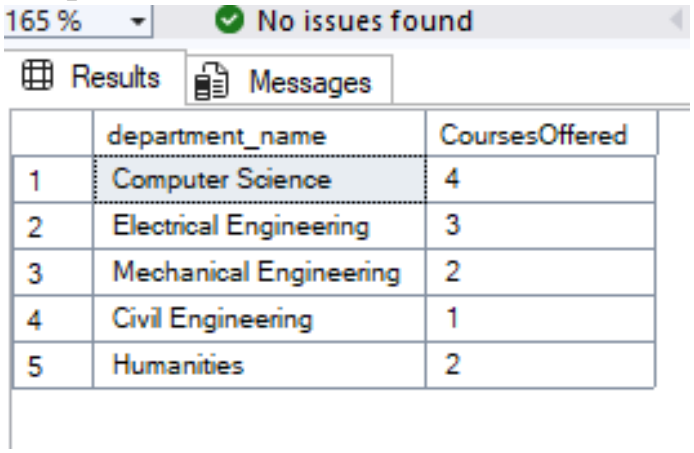
- Learn to normalize data using parent-child tables.
- Practice inserting multiple records into tables.
- Understand subquery usage for data aggregation.
- Filter results based on aggregated data.
- Implement basic user permissions in SQL.

3. DBMS Script :

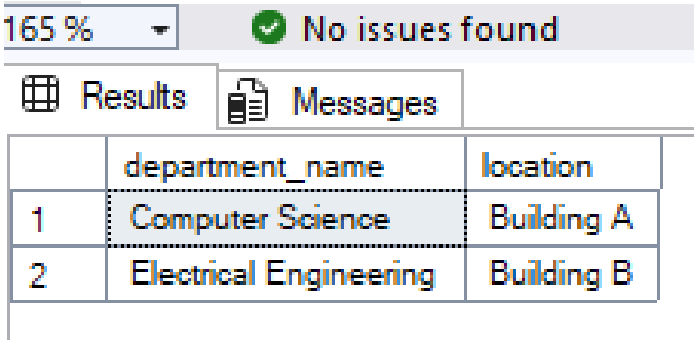
```
USE KRG_2B;

CREATE TABLE Departments (
    department_id INT PRIMARY KEY,
    department_name VARCHAR(100),
    location VARCHAR(100)
);

CREATE TABLE Courses (
    course_id INT PRIMARY KEY,
    course_name VARCHAR(200),
    credits INT,
    department_id INT,
    FOREIGN KEY (department_id) REFERENCES Departments(department_id)
);

INSERT INTO Departments (department_id, department_name, location) VALUES
(1, 'Computer Science', 'Building A'),
(2, 'Electrical Engineering', 'Building B'),
(3, 'Mechanical Engineering', 'Building C'),
(4, 'Civil Engineering', 'Building D'),
(5, 'Humanities', 'Building E');

INSERT INTO Courses (course_id, course_name, credits, department_id) VALUES
(101, 'Data Structures', 3, 1),
(102, 'Algorithms', 4, 1),
(103, 'Operating Systems', 3, 1),
(104, 'Digital Logic Design', 3, 2),
(105, 'Microprocessors', 4, 2),
(106, 'Thermodynamics', 3, 3),
(107, 'Fluid Mechanics', 3, 3),
(108, 'Structural Analysis', 4, 4),
(109, 'Engineering Ethics', 2, 5),
```

```
(110, 'Technical Writing', 3, 5),
(111, 'Database Management', 4, 1),
(112, 'Communication Systems', 3, 2);

-- Use a subquery to count the number of courses under each department.
SELECT d.department_name,(SELECT COUNT(c.course_id)
   FROM Courses c
   WHERE c.department_id = d.department_id) AS CoursesOffered
FROM Departments d;

-- Filter and retrieve only those departments that offer more than two courses.
SELECT d.department_name, d.location
FROM Departments d
WHERE (SELECT COUNT(c.course_id)
   FROM Courses c
   WHERE c.department_id = d.department_id) > 2;

-- Grant SELECT-only access on the courses table to a specific user
GRANT SELECT ON Courses TO student_user;
```

4. Output:



	department_name	CoursesOffered
1	Computer Science	4
2	Electrical Engineering	3
3	Mechanical Engineering	2
4	Civil Engineering	1
5	Humanities	2



	department_name	location
1	Computer Science	Building A
2	Electrical Engineering	Building B

5. Learning Outcomes:

- Successfully designed and populated relational tables.
- Implemented foreign key constraints for data integrity.
- Utilized subqueries for complex data analysis.
- Filtered results based on aggregated subquery values.
- Managed database access permissions efficiently.