**ADBMS LAB EXPERIMENT - 02**

**Name:** Ambika Kashyap

* **UID:** 23BCC70026
* **Subject:** ADBMS
* **Experiment Number:** 02

**AIM:**

To design a normalized academic schema (up to 3NF) for managing departments and their courses, populate it with meaningful sample data, query departments offering more than two courses using a subquery, and implement access control using Data Control Language (DCL).

**THEORY:**

**Normalization (3NF):**

Third Normal Form (3NF) eliminates transitive dependencies.  
A relation is in 3NF if:

* It is in 2NF, and
* No transitive functional dependency exists between non-prime attributes.

**Relational Model Design:**

* **Departments Table**: Holds unique department data.
* **Courses Table**: Associates each course with exactly one department using a foreign key.

**Subqueries:**

A subquery is a query nested inside another SQL query. It helps filter, transform, or summarize data based on conditions.

**Access Control using DCL:**

DCL (Data Control Language) statements like GRANT manage user privileges.  
For example, granting SELECT access ensures a user can view but not modify data.

CODE:

-- Drop existing tables if they exist (clean slate)

DROP TABLE IF EXISTS Courses;

DROP TABLE IF EXISTS Departments;

-- Step 1: Create Departments table

CREATE TABLE Departments (

dept\_id INT PRIMARY KEY,

dept\_name VARCHAR(50) NOT NULL UNIQUE

);

-- Step 2: Create Courses table with foreign key to Departments

CREATE TABLE Courses (

course\_id INT PRIMARY KEY,

course\_name VARCHAR(100) NOT NULL,

dept\_id INT NOT NULL,

FOREIGN KEY (dept\_id) REFERENCES Departments(dept\_id) ON DELETE CASCADE

);

-- Step 3: Insert data into Departments

INSERT INTO Departments (dept\_id, dept\_name) VALUES

(1, 'Computer Science'),

(2, 'Electrical'),

(3, 'Mechanical'),

(4, 'Civil'),

(5, 'Electronics');

-- Step 4: Insert data into Courses

INSERT INTO Courses (course\_id, course\_name, dept\_id) VALUES

(101, 'DBMS', 1),

(102, 'Operating Systems', 1),

(103, 'Power Systems', 2),

(104, 'Digital Circuits', 2),

(105, 'Thermodynamics', 3),

(106, 'Fluid Mechanics', 3),

(107, 'Structural Engineering', 4),

(108, 'Surveying', 4),

(109, 'Embedded Systems', 5),

(110, 'VLSI Design', 5),

(111, 'Compiler Design', 1); -- extra course to make sure CS has >2

-- Step 5: Query departments offering more than 2 courses

SELECT dept\_name

FROM Departments

WHERE dept\_id IN (

SELECT dept\_id

FROM Courses

GROUP BY dept\_id

HAVING COUNT(\*) >= 2

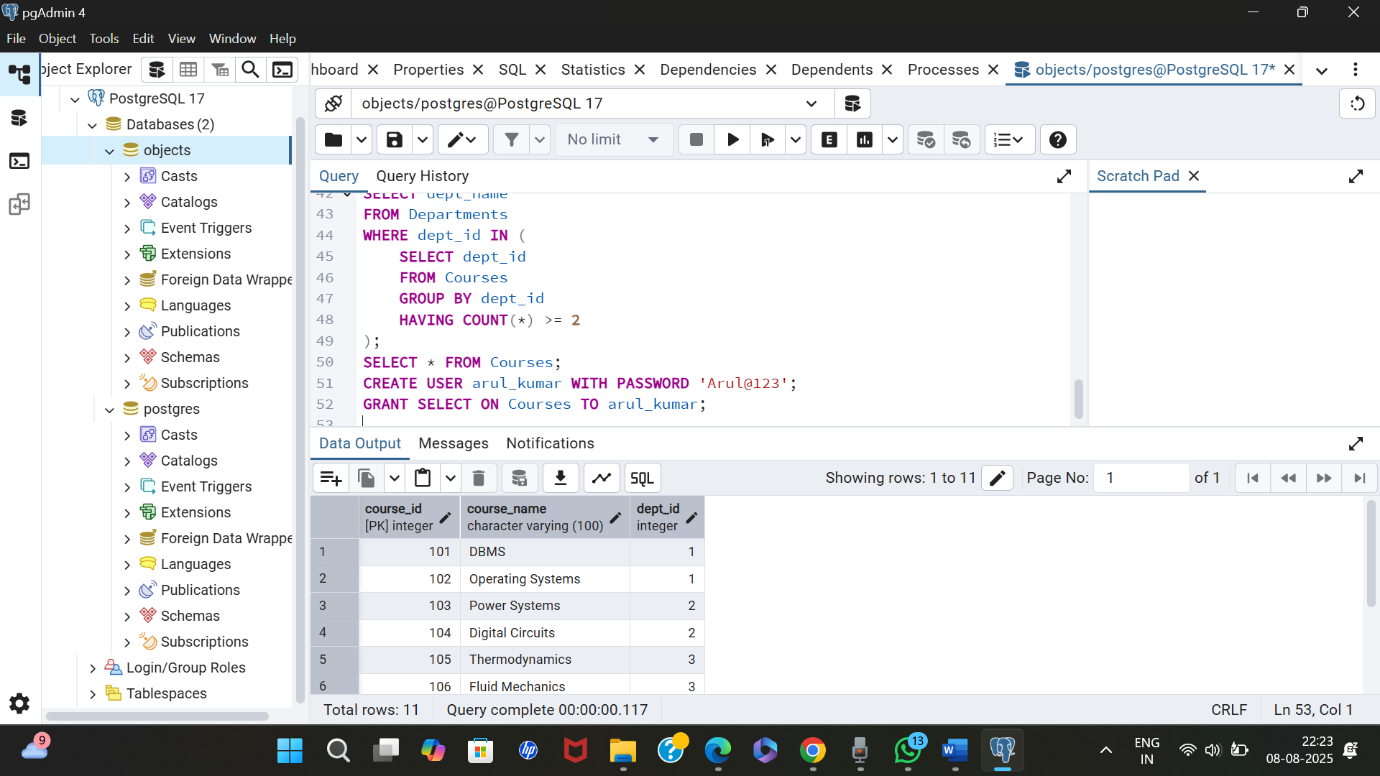
);

SELECT \* FROM Courses;

CREATE USER arul\_kumar WITH PASSWORD ‘Arul@123';

GRANT SELECT ON Courses TO arul\_kumar;

OUTPUT:



**LEARNING OUTCOMES**

✔ Understand and apply **3NF normalization** in relational database design.  
✔ Use **foreign key constraints** to maintain referential integrity.  
✔ Write **subqueries with GROUP BY and HAVING** to analyze relationships.  
✔ Implement **access control** using GRANT statements in PostgreSQL.  
✔ Efficiently handle **real-world schema modeling and data organization** tasks.