Experiment 1

Student Name: Shashwat Mishra **UID: 23BCS10866**
Branch: CSE **Section/Group: KRG 3-A**
Semester: 5th **Date of Performance:24/07/2025**
Subject Name: ADBMS **Subject Code: 23CSP-333**

1. **Aim**: University Database System helps in managing student enrolments , course allocations, and professor assignments effectively. The system also demonstrates secure access control and transaction safety. This includes CRUD operations, JOIN queries, and database-level user permission management.
   a. Author-Book Relationship Using Joins and Basic SQL Operations
   b. Department-Course Subquery and Access Control

2. **Objective:**

   The objective of the given SQL code is to demonstrate how to model and query relational data in a university-like database system using foundational and intermediate SQL concepts. It covers the creation of tables for authors, books, departments, and courses, and showcases how to retrieve combined data using JOINs, as well as how to apply subqueries for conditional filtering based on grouped data. The code emphasizes practical understanding of entity relationships and efficient data retrieval strategies.

3. **DBMS script and output:**

   Solution-(a)

```
-- Creating AUTHOR table

CREATE TABLE AUTHOR_INFO (

   ID INT PRIMARY KEY,

   Name VARCHAR(50),

   Nation VARCHAR(50)

);



-- Creating BOOK table

CREATE TABLE BOOK_INFO (
```

```
BookCode INT PRIMARY KEY,
    BookTitle VARCHAR(100),
    WrittenBy INT,
    FOREIGN KEY (WrittenBy) REFERENCES AUTHOR_INFO(ID)
);
-- Inserting authors
INSERT INTO AUTHOR_INFO (ID, Name, Nation) VALUES
(1, 'Shashwat', 'India'),
(2, 'Chetna', 'Nepal'),
(3, 'Akshat', 'Bhutan'),
(4, 'Sweety', 'India'),
(5, 'Sachet', 'Nigeria');
-- Inserting books
INSERT INTO BOOK_INFO (BookCode, BookTitle, WrittenBy) VALUES
(201, 'DBMS Unlocked', 1),
(202, 'DAA Simplified', 2),
(203, 'Code Like a Pro', 3),
(204, 'Easy Algorithms', 4),
(205, 'Crack C++', 5);
-- Query to get book details with author info
SELECT B.BookTitle, A.Name AS AuthorName, A.Nation
FROM BOOK_INFO B
JOIN AUTHOR_INFO A ON B.WrittenBy = A.ID;
```

	BookTitle	AuthorName	Nation
1	DBMS Unlocked	Shashwat	India
2	DAA Simplified	Chetna	Nepal
3	Code Like a Pro	Akshat	Bhutan
4	Easy Algorithms	Sweety	India
5	Crack C++	Sachet	Nigeria

Solution-(b)

```
--Medium-Level Problem

-- Creating DEPARTMENT table
CREATE TABLE DEPT_MASTER (
    DeptCode INT PRIMARY KEY,
    Department VARCHAR(50)
);

-- Creating COURSE table
CREATE TABLE COURSE_LIST (
    CID INT PRIMARY KEY,
    CourseTitle VARCHAR(100),
    DCode INT,
    FOREIGN KEY (DCode) REFERENCES DEPT_MASTER(DeptCode)
);

-- Inserting departments
INSERT INTO DEPT_MASTER (DeptCode, Department) VALUES
(10, 'CSE'),
(20, 'IT'),
(30, 'ECE'),
(40, 'ME'),
(50, 'CE');
```

```
-- Inserting courses
INSERT INTO COURSE_LIST (CID, CourseTitle, DCode) VALUES
(301, 'Intro to DBMS', 10),


(302, 'Design & Analysis', 10),
(303, 'Coding Bootcamp', 10),
(304, 'Frontend Essentials', 20),
(305, 'Security Basics', 20),
(306, 'DS in Depth', 20),
(307, 'Logic Circuits', 30),
(308, 'Microcontrollers', 30),
(309, 'Heat Transfer', 40),
(310, 'Civil Engineering Basics', 50);

-- Query to get departments with more than 2 courses
SELECT Department
FROM DEPT_MASTER
WHERE DeptCode IN (
   SELECT DCode
   FROM COURSE_LIST
   GROUP BY DCode
   HAVING COUNT(*) > 2
);
```

	Department
1	CSE
2	IT

4. Learning Outcomes (What I have Learnt):

- Understand the creation of relational tables using **primary and foreign key constraints**.
- Gain hands-on experience with **INNER JOIN** to retrieve related data across tables.
- Learn to use **subqueries with GROUP BY and HAVING** clauses for data aggregation and filtering.
- Develop the ability to represent **one-to-many relationships** between entities (e.g., Authors → Books, Departments → Courses).
- Enhance skills in performing **basic and intermediate-level SQL queries** involving selection, joins, and subqueries.