EXPERIMENT - 1

Student Name: Aditya Bhatia **UID: 23BCS10267**
Branch: BE-CSE **Section/Group: KRG_1-B**
Semester: 5th **Date of Performance: 22/07/2025**
Subject Name: ADBMS **Subject Code: 23CSP-333**

1. AIM: Ques 1 :- Author-Book Relationship Using Joins and Basic SQL Operations. **(EASY LEVEL)**

- Design two tables — one for storing author details and the other for book details.
- Ensure a foreign key relationship from the book to its respective author.
- Insert at least three records in each table.
- Perform an INNER JOIN to link each book with its author using the common author ID.
- Select the book title, author name, and author's country.

2. TOOLS USED:- MS SSMS & Microsoft SQL Server

3. SQL CODE:

```
use KRG1B;


CREATE TABLE TBL_AUTHOR(

AUTHOR_ID INT PRIMARY KEY,

AUTHOR_NAME VARCHAR(30));


CREATE TABLE TBL_BOOK(

BOOK_ID INT PRIMARY KEY,

BOOK_TITLE VARCHAR(30),

AUTHOR_ID INT,

FOREIGN KEY (AUTHOR_ID) REFERENCES TBL_AUTHOR(AUTHOR_ID));
```

```
INSERT INTO TBL_AUTHOR VALUES
(1, 'A'),
(2, 'B'),
(3, 'C');

INSERT INTO TBL_BOOK VALUES
(101, 'Maze Runners', 1),
(102, 'Hunger Games', 2),
(103, 'Harappa', 3),
(104, 'The Darkest Minds', 1),
(105, 'Harry Potter', 2);

SELECT * FROM TBL_BOOK;
SELECT * FROM TBL_AUTHOR;

SELECT B.BOOK_TITLE , A.AUTHOR_NAME
FROM TBL_BOOK AS B
INNER JOIN
TBL_AUTHOR AS A
ON
B.AUTHOR_ID = A.AUTHOR_ID;
```

4. OUTPUT:

	BOOK_TITLE	AUTHOR_NAME
1	Maze Runners	A
2	Hunger Games	B
3	Harappa	C
4	The Darkest Minds	A
5	Harry Potter	B

5. Ques 2: -Department-Course Subquery and Access Control. **(MEDIUM LEVEL)**

- 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.

6. SQL CODE:

```
-- Step 1: Create Tables
CREATE TABLE Departments (
department_id INT PRIMARY KEY,
department_name VARCHAR(100) NOT NULL
);

CREATE TABLE Courses (
course_id INT PRIMARY KEY,
course_name VARCHAR(100) NOT NULL,
department_id INT,
FOREIGN KEY (department_id) REFERENCES Departments(department_id)
);

-- Step 2: Insert Data into Departments
INSERT INTO Departments (department_id, department_name) VALUES
(1, 'Computer Science'),
(2, 'Mechanical Engineering'),
(3, 'Electrical Engineering'),
(4, 'Civil Engineering'),
(5, 'Mathematics');
```

```
-- Step 3: Insert Data into Courses
INSERT INTO Courses (course_id, course_name, department_id) VALUES
(101, 'Data Structures', 1),
(102, 'Operating Systems', 1),
(103, 'Machine Learning', 1),
(104, 'Thermodynamics', 2),

(105, 'Fluid Mechanics', 2),
(106, 'Circuits and Systems', 3),
(107, 'Control Systems', 3),
(108, 'Structural Analysis', 4),
(109, 'Linear Algebra', 5),
(110, 'Calculus', 5),
(111, 'Probability Theory', 5);

-- Step 4: Count Number of Courses per Department
SELECT
   department_name,
   (SELECT COUNT(*)
    FROM Courses c
    WHERE c.department_id = d.department_id) AS course_count
FROM Departments d;

-- Step 5: Filter Departments Offering More Than 2 Courses
SELECT
   department_name,
   (SELECT COUNT(*)
    FROM Courses c
    WHERE c.department_id = d.department_id) AS course_count
FROM Departments d
WHERE (SELECT COUNT(*)
      FROM Courses c
      WHERE c.department_id = d.department_id) > 2;
```

7. OUTPUT

	department_name	course_count
1	Computer Science	3
2	Mathematics	3