EXPERIMENT-01

Student Name: DIVYANSH **UID: 23BCS11778**

Branch: BE-CSE **Section/Group: KRG 1(A)**

Semester: 05 **Date of Performance: 17/07/25**

Subject Name: ADBMS **Subject Code: 23CSP-333**

Easy-Level Problem

1. Aim:

Author - Book Relationship Using Joins and Basic SQL Operations.

- 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. Objective:

- Understand basic table creation with primary and foreign keys.
- Practice inserting data into relational tables.
- Implement JOIN operations to fetch combined data from multiple tables.

3. Code:

```
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 (AUTHOR_ID, AUTHOR_NAME) VALUES
(1, 'C.J. Date'),
(2, 'Silberschatz'),
(3, 'A. Tanenbaum');

INSERT INTO TBL_BOOK (BOOK_ID, BOOK_TITLE, AUTHOR_ID) VALUES
(101, 'Database Systems', 1),
(102, 'Operating Systems', 2),
(103, 'Computer Networks', 3),
(104, 'Advanced Databases', 1),
(105, 'Modern OS', 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:

Results Messages

	BOOK_ID	BOOK_TITLE	AUTHOR_ID
1	101	Database Systems	1
2	102	Operating Systems	2
3	103	Computer Networks	3
4	104	Advanced Databases	1
5	105	Modern OS	2

Results Messages

	AUTHOR_ID	AUTHOR_NAME
1	1	C.J. Date
2	2	Silberschatz
3	3	A. Tanenbaum

Results Messages

	BOOK_TITLE	AUTHOR_NAME
1	Database Systems	C.J. Date
2	Operating Systems	Silberschatz
3	Computer Networks	A. Tanenbaum
4	Advanced Databases	C.J. Date
5	Modern OS	Silberschatz

Medium-Level Problem

1. Aim:

Department- Course Subquery and Access Control.

1. Design normalized tables for departments and the courses they offer, maintaining a foreign key relationship.
2. Insert five departments and at least ten courses across those departments.
3. Use a subquery to count the number of courses under each department.
4. Filter and retrieve only those departments that offer more than two courses.
5. Grant SELECT-only access on the courses table to a specific user.

2. Objective:

- Create normalized tables with proper foreign key relationships.
- Use subqueries to count and filter relational data.

3. Code:

```
CREATE TABLE TBL_DEPARTMENT (
DEPT_ID INT PRIMARY KEY,
DEPT_NAME VARCHAR(30));

CREATE TABLE TBL_COURSE(
COURSE_ID INT PRIMARY KEY,
COURSE_NAME VARCHAR(30),
DEPT_ID INT,
FOREIGN KEY (DEPT_ID) REFERENCES TBL_DEPARTMENT(DEPT_ID));

INSERT INTO TBL_DEPARTMENT (DEPT_ID, DEPT_NAME) VALUES
(1, 'Computer Science'),
(2, 'Electrical Engineering'),
(3, 'Mechanical Engineering'),
(4, 'Civil Engineering'),
(5, 'Mathematics');

INSERT INTO TBL_COURSE (COURSE_ID, COURSE_NAME, DEPT_ID) VALUES
(101, 'Data Structures', 1),
(102, 'Algorithms', 1),
```

```
    (103, 'Operating Systems', 1),
    (104, 'Circuits', 2),
    (105, 'Digital Logic', 2),
    (106, 'Thermodynamics', 3),
    (107, 'Fluid Mechanics', 3),
    (108, 'Surveying', 4),
    (109, 'Calculus', 5),
    (110, 'Linear Algebra', 5),
    (111, 'Discrete Math', 5);

SELECT DEPT_NAME
FROM TBL_DEPARTMENT
WHERE DEPT_ID IN (
    SELECT DEPT_ID
    FROM TBL_COURSE
    GROUP BY DEPT_ID
    HAVING COUNT(*) > 2
);
```

4. Output:

Results | Messages

	DEPT_NAME
1	Computer Science
2	Mathematics