Experiment No: 1.1

Student Name: Mohammad Kaif **UID: 23BCS12884**
Branch: B.E./C.S.E. **Section/Group: KRG_1-B**
Semester: 5th
Subject Name: ADBMS
Subject Code: 23CSP-333

Question 1 :Easy Level Problem

Problem Title: Author-Book Relationship Using Joins and Basic SQL Operations
Procedure (Step-by-Step):

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

Sample Output Description: When the join is performed, we get a list where each book title is shown along with its author's name and their country question

Solution:

```
CREATE TABLE TBL_AUTHOR
(
	AUTHOR_ID INT PRIMARY KEY,
	AUTHOR_NAME VARCHAR(50),
	COUNTRY VARCHAR(50)
);

CREATE TABLE TBL_BOOK
(
	BOOK_ID INT PRIMARY KEY,
	BOOK_TITLE VARCHAR(50),
	AUTHOR_ID INT,
```

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

INSERT INTO TBL_AUTHOR VALUES (1, 'Alice', 'USA');
INSERT INTO TBL_AUTHOR VALUES (2, 'Bob', 'UK');
INSERT INTO TBL_AUTHOR VALUES (3, 'Cathy', 'India');

INSERT INTO TBL_BOOK VALUES (101, 'Book A', 1);
INSERT INTO TBL_BOOK VALUES (102, 'Book B', 2);
INSERT INTO TBL_BOOK VALUES (103, 'Book C', 3);

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

Output

Question 2 :Medium Level Problem

Problem Title: Transaction Management and Savepoint Simulation in Student Enrollments
Procedure (Step-by-Step):

1. Create three normalized tables — one each for students, courses, and enrollments.
2. Insert sample data for students and courses, then begin a transaction.
3. Add one enrollment successfully, then create a SAVEPOINT.
4. Attempt to insert a faulty or invalid enrollment to simulate an error.
5. Roll back only to the SAVEPOINT (not the entire transaction), then commit the valid data
6. Finally, join all three tables to display the student's name, the course title they enrolled in, and the grade they received.

Sample Output Description: After performing the join, we get a list of students with the courses they are enrolled in, along with their grades.

Solution:

```
CREATE TABLE STUDENT (
   STUDENT_ID INT PRIMARY KEY,
   STUDENT_NAME VARCHAR(50)
);

CREATE TABLE COURSE (
   COURSE_ID INT PRIMARY KEY,
   COURSE_NAME VARCHAR(50)
);

CREATE TABLE ENROLLMENT (
```

```
    ENROLLMENT_ID INT PRIMARY KEY,
    STUDENT_ID INT FOREIGN KEY REFERENCES STUDENT(STUDENT_ID),
    COURSE_ID INT FOREIGN KEY REFERENCES COURSE(COURSE_ID),
    GRADE CHAR(1)
);

INSERT INTO STUDENT VALUES (1, 'Alice');
INSERT INTO STUDENT VALUES (2, 'Bob');
INSERT INTO STUDENT VALUES (3, 'Charlie');

INSERT INTO COURSE VALUES (101, 'Data Structures');
INSERT INTO COURSE VALUES (102, 'Algorithms');
INSERT INTO COURSE VALUES (103, 'DBMS');
INSERT INTO COURSE VALUES (104, 'Operating Systems');
INSERT INTO COURSE VALUES (105, 'Networking');
INSERT INTO COURSE VALUES (106, 'Web Development');
INSERT INTO COURSE VALUES (107, 'Software Engineering');
INSERT INTO COURSE VALUES (108, 'Machine Learning');
INSERT INTO COURSE VALUES (109, 'Computer Graphics');
INSERT INTO COURSE VALUES (110, 'Cyber Security');

BEGIN TRANSACTION;
INSERT INTO ENROLLMENT VALUES (1, 1, 101, 'A');
SAVE TRANSACTION sp1;

BEGIN TRY
    INSERT INTO ENROLLMENT VALUES (2, 99, 102, 'B');
END TRY
BEGIN CATCH
```

```
    ROLLBACK TRANSACTION sp1;
END CATCH;

INSERT INTO ENROLLMENT VALUES (3, 2, 102, 'B');
INSERT INTO ENROLLMENT VALUES (4, 3, 103, 'C');
INSERT INTO ENROLLMENT VALUES (5, 2, 104, 'A');
INSERT INTO ENROLLMENT VALUES (6, 3, 105, 'B');
COMMIT TRANSACTION;
SELECT
    S.STUDENT_NAME,
    C.COURSE_NAME,
    E.GRADE
FROM
    ENROLLMENT E
JOIN STUDENT S ON E.STUDENT_ID = S.STUDENT_ID
JOIN COURSE C ON E.COURSE_ID = C.COURSE_ID;
```

Output :