**Experiment 1.1**

**Student Name: Umang Kaushik UID: 23BCS10712**

**Branch: BE CSE Section/Group: KRG-1-B**

**Semester: 5th Date of Performance:29th July**

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

1. **Problem Title:** Author-Book Relationship Using Joins and Basic SQL Operations
2. **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.**

1. **Code:**

-- Create the author table

CREATE TABLE TBL\_AUTHOR

(

AUTHOR\_ID INT PRIMARY KEY,

AUTHOR\_NAME VARCHAR(20),

COUNTRY VARCHAR(20)

);

GO

-- Create the book table with a foreign key

CREATE TABLE TBL\_BOOK

(

BOOK\_ID INT PRIMARY KEY,

BOOK\_TITLE VARCHAR(20),

AUHTORID INT,

FOREIGN KEY (AUHTORID) REFERENCES TBL\_AUTHOR(AUTHOR\_ID) -- Used to maintain referential integrity

);

GO

-- Insert values into TBL\_AUTHOR

INSERT INTO TBL\_AUTHOR (AUTHOR\_ID, AUTHOR\_NAME, COUNTRY) VALUES

(1, 'J.K. Rowling', 'UK'),

(2, 'R.K. Narayan', 'India'),

(3, 'Chetan Bhagat', 'India');

GO

-- Insert values into TBL\_BOOK

INSERT INTO TBL\_BOOK (BOOK\_ID, BOOK\_TITLE, AUHTORID) VALUES

(101, 'Harry Potter', 1),

(102, 'Malgudi Days', 2),

(103, 'Two States', 3),

(104, 'The Guide', 2);

GO

-- Select and join data from both tables

SELECT B.BOOK\_TITLE, A.AUTHOR\_NAME, A.COUNTRY

FROM TBL\_BOOK AS B

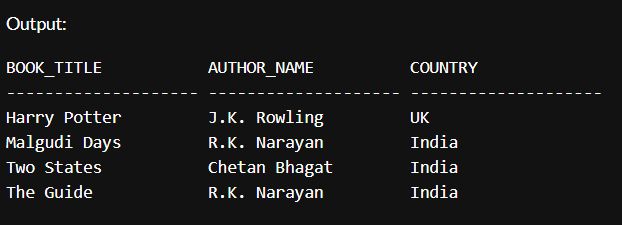
INNER JOIN

TBL\_AUTHOR AS A

ON

B.AUHTORID = A.AUTHOR\_ID;

GO



**Medium-Level Problem**

**Problem Title:** Department-Course Subquery and Access Control

**Procedure (Step-by-Step):**

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. - GROUP BY
4. **Code:**

DROP TABLE IF EXISTS COURSES;

DROP TABLE IF EXISTS DEPARTMENTS;

GO

CREATE TABLE DEPARTMENTS (

DEPTID INT PRIMARY KEY,

DEPARTMENT\_NAME VARCHAR(100) NOT NULL

);

GO

CREATE TABLE COURSES (

COURSEID INT PRIMARY KEY,

COURSE\_NAME VARCHAR(100) NOT NULL,

DEPTID INT,

FOREIGN KEY (DEPTID) REFERENCES DEPARTMENTS(DEPTID)

);

GO

INSERT INTO DEPARTMENTS (DEPTID, DEPARTMENT\_NAME) VALUES

(1, 'Computer Science'),

(2, 'Mathematics'),

(3, 'Physics'),

(4, 'History'),

(5, 'Art');

GO

INSERT INTO COURSES (COURSEID, COURSE\_NAME, DEPTID) VALUES

(101, 'Introduction to Programming', 1),

(102, 'Data Structures', 1),

(103, 'Database Systems', 1),

(104, 'Operating Systems', 1),

(201, 'Calculus I', 2),

(202, 'Linear Algebra', 2),

(301, 'Classical Mechanics', 3),

(302, 'Electromagnetism', 3),

(303, 'Quantum Physics', 3),

(401, 'World History', 4),

(501, 'Drawing Fundamentals', 5);

GO

SELECT DEPARTMENT\_NAME

FROM DEPARTMENTS

WHERE DEPTID IN (

SELECT DEPTID

FROM COURSES

GROUP BY DEPTID

HAVING COUNT(\*) > 2

);

GO

BEGIN

CREATE USER UMANG WITHOUT LOGIN;

END

GO

GRANT SELECT ON COURSES TO UMANG;

GO

DROP TABLE IF EXISTS EMPLOYEE;

GO

CREATE TABLE EMPLOYEE (

EMPID INT,

EMPNAME VARCHAR(50),

SALARY INT

);

GO

INSERT INTO EMPLOYEE (EMPID, EMPNAME, SALARY) VALUES

(1, 'Alice', 70000),

(2, 'Bob', 25000),

(3, 'Charlie', 65000),

(4, 'David', 32000),

(5, 'Eve', 70000);

GO

SELECT MAX(SALARY) AS SecondHighestSalary

FROM EMPLOYEE

WHERE SALARY < (SELECT MAX(SALARY) FROM EMPLOYEE);

GO

SELECT SALARY AS SecondHighestSalary

FROM EMPLOYEE

ORDER BY SALARY DESC

OFFSET 1 ROWS

FETCH NEXT 1 ROWS ONLY;

GO

