Experiment 1

Student Name: Gaurav Singh **UID: 23BCS12992**
Branch: BE CSE **Section/Group: KRG-1-A**
Semester: 5 **Date of Performance: 25-7-25**
Subject Name: ADBMS **Subject Code: 23CSP-333**

1. Aim:

Author-Book Relationship Using Joins and Basic SQL.

2. Requirements (Hardware/Software):

Microsoft SQL server

3. Procedure:

Q.1. 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.

Code:

```
-- create
CREATE TABLE TBL_AUTHORS (
  AUTH_ID int primary key,
  AUTH_NAME varchar(20)
);
CREATE TABLE TBL_BOOKS (
  BOOK_ID INT primary key,
  BOOK_NAME varchar(20),   AUTH_ID int,   foreign key
(AUTH_ID) references TBL_AUTHORS(AUTH_ID)
)



-- insert
INSERT INTO TBL_AUTHORS VALUES (1, 'Fyodor Dostoevsky');
```

```
INSERT INTO TBL_AUTHORS VALUES (2, 'Franz Kafka');
INSERT INTO TBL_AUTHORS VALUES (3, 'J.K. Rowling');

INSERT INTO TBL_BOOKS VALUES (1, 'Metamorphosis',2);
INSERT INTO TBL_BOOKS VALUES (2, 'Harry Potter',3);
INSERT INTO TBL_BOOKS VALUES (3, 'Crime and Punishment',1);

Select TBL_BOOKS.BOOK_ID, TBL_BOOKS.BOOK_NAME ,TBL_AUTHORS.*
from TBL_AUTHORS
inner Join
TBL_BOOKS
on
TBL_AUTHORS.AUTH_ID=TBL_BOOKS.AUTH_ID;
```

Q.2. 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.

```
CREATE TABLE Department (
   DeptID INT PRIMARY KEY,
   DeptName VARCHAR(100)
);

-- Create Course Table
CREATE TABLE Course (
   CourseID INT PRIMARY KEY,
   CourseName VARCHAR(100),
   DeptID INT,
   FOREIGN KEY (DeptID) REFERENCES Department(DeptID)
);

-- Insert Departments
INSERT INTO Department VALUES
(1, 'Computer Science'),
(2, 'Physics'),
(3, 'Mathematics'),
(4, 'Chemistry'),
(5, 'Biology');




-- Insert Courses
INSERT INTO Course VALUES
```

```
(101, 'Data Structures', 1),
(102, 'Operating Systems', 1),
(103, 'Quantum Mechanics', 2),
(104, 'Electromagnetism', 2),
(105, 'Linear Algebra', 3),
(106, 'Calculus', 3),
(107, 'Organic Chemistry', 3),
(108, 'Physical Chemistry', 4),
(109, 'Genetics', 5),
(110, 'Molecular Biology', 5);

Select DeptID, (select count(*) from course where course.DeptID=Department.DeptID) as
Num_Courses from Department;

SELECT DEPTNAME
FROM DEPARTMENT
WHERE DEPTID IN (
   SELECT DEPTID
   FROM COURSE
   GROUP BY DEPTID
   HAVING COUNT(*) > 2
);
```

4. Output:

Q.1.

```
Output:

BOOK_ID     BOOK_NAME            AUTH_ID     AUTH_NAME
----------- -------------------- ----------- --------------------
          1 Metamorphosis                  2 Franz Kafka
          2 Harry Potter                   3 J.K. Rowling
          3 Crime and Punishment           1 Fyodor Dostoevsky
```

Q.2.

```
Output:

DeptID      Num_Courses
----------- -----------
          1           2
          2           2
          3           3
          4           1
          5           2
DEPTNAME
------------------------------------------------------------
Mathematics
```

5. Learning Outcome:

- **Understand the purpose and use cases of different SQL join types (INNER, LEFT, RIGHT, FULL).**
- **Apply SQL joins to retrieve data from multiple related tables effectively.**
- **Analyze the impact of join conditions on query results and data integrity.**
- **Construct optimized SQL queries using joins for real-world database scenarios.**