Experiment 1

Student Name: Punit Kumar **UID: 23BCS14148**
Branch: CSE **Section/Group: 23BCS_KRG-1/A**
Semester: 5 **Date of Performance:24/07/25**
Subject Name: Advanced Database and Management System **Subject Code: 23CSP-333**

1. Aim:

[EASY] Author-Book Relationships using Joins and basic SQL Operations.

1. Design Two tables – one for storing author details and the other for book details.
2. Ensure Foreign Key relationship from 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.

[MEDIUM] 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 courses to specific user.

2. Tools Used: SQL Server Management Studio

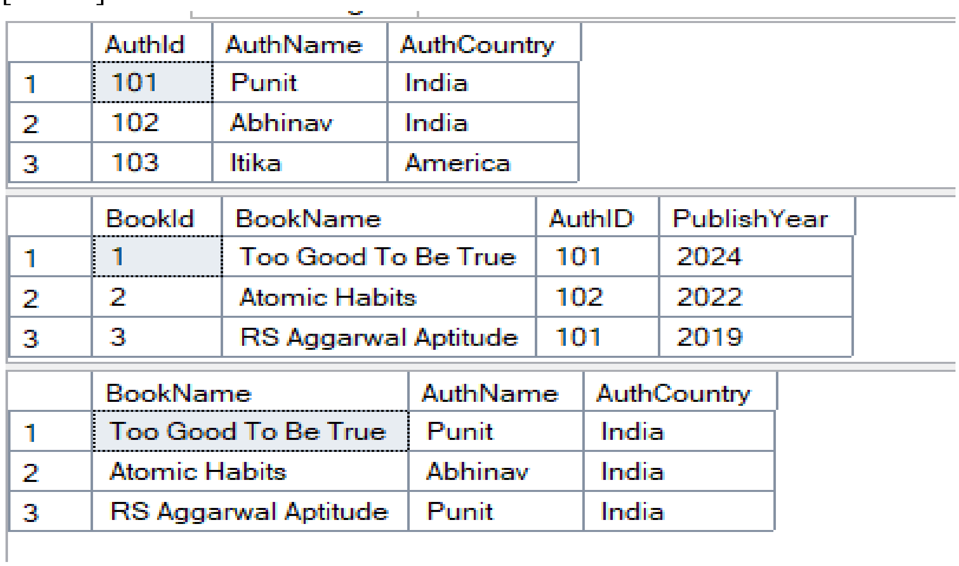
3. Code:

```
-- Q1 -Author-Book Relationship Using Joins and Basic SQL
Operations
CREATE DATABASE EXP1;
USE EXP1;

CREATE TABLE Author_Tbl(AuthId INT PRIMARY KEY, AuthName
VARCHAR(30), AuthCountry VARCHAR(20));
```

```sql
CREATE TABLE Books_Tbl(BookId INT PRIMARY KEY, BookName
VARCHAR(30), AuthID int, PublishYear int);

ALTER TABLE Books_Tbl ADD CONSTRAINT Fk_Books_Author
FOREIGN KEY (AuthID) references Author_Tbl(AuthId);

INSERT INTO Author_Tbl (AuthId, AuthName, AuthCountry)
values (101, 'Punit', 'India'), (102, 'Abhinav', 'India'),
(103, 'Itika', 'America');

SELECT * FROM Author_Tbl;

INSERT INTO Books_Tbl(BookId, BookName, AuthID,
PublishYear)
values
(1, 'Too Good To Be True', 101, 2024),
(2, 'Atomic Habits', 102, 2022),
(3, 'RS Aggarwal Aptitude', 101, 2019);


SELECT * FROM Books_Tbl;


SELECT B.BookName, A.AuthName, A.AuthCountry
FROM Author_Tbl A
INNER JOIN
Books_Tbl B
ON (A.AuthId = B.AuthID);



--Q2 Department-Course Subquery and Access Control

CREATE TABLE Department_Tbl(DeptId int PRIMARY KEY,
DeptName varchar(30));

CREATE TABLE Courses(CourseId int Primary key, CourseName
varchar(30), DeptId int, FOREIGN KEY(DeptId) REFERENCES
Department_Tbl(DeptId));

INSERT INTO Department_Tbl(DeptId, DeptName) VALUES
(1, 'COMPUTER SCIENCE'),
(2, 'MECHANICAL ENGINEERING'),
(3, 'PHYSICS'),
(4, 'MATHEMATICS');

INSERT INTO Courses (CourseId, CourseName, DeptId) VALUES
(101, 'DATA STRUCTURES', 1),
(102, 'OPERATING SYSTEMS', 1),
(103, 'DATABASE SYSTEMS', 1),
(201, 'THERMODYNAMICS', 2),
(202, 'FLUID MECHANICS', 2),
(203, 'MACHINE DESIGN', 2),
(301, 'CLASSICAL MECHANICS', 3),
(401, 'CIRCUIT THEORY', 4),
(402, 'DIGITAL LOGIC DESIGN', 4);


SELECT D.DeptName, COUNT(C.DeptId) AS [NO OF COURSES]
FROM Department_Tbl D
INNER JOIN
Courses C
ON D.DeptId = C.DeptId
GROUP BY D.DeptName;


SELECT D.DeptName, COUNT(C.DeptId) AS [NO OF COURSES]
FROM Department_Tbl D
INNER JOIN
Courses C
ON D.DeptId = C.DeptId
GROUP BY D.DeptName
HAVING COUNT(C.DeptId) > 2;

create login PUNIT with password = 'PUNIT1234';

CREATE USER PUNIT;

GRANT SELECT ON COURSES TO PUNIT;
```

4. Output:

[EASY]

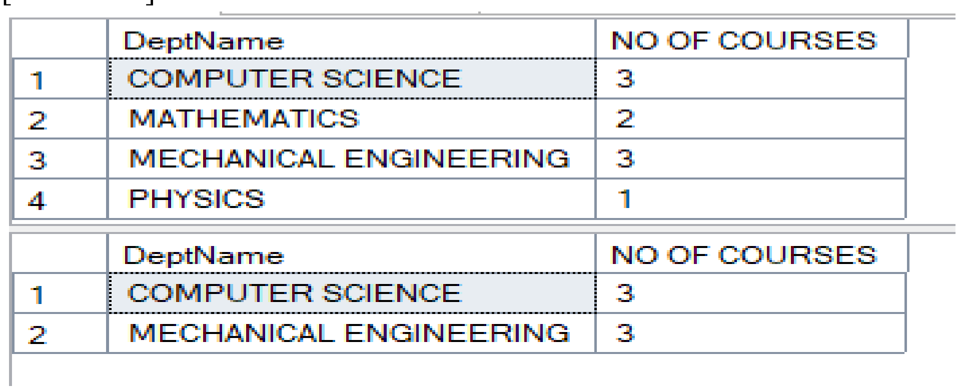


	AuthId	AuthName	AuthCountry
1	101	Punit	India
2	102	Abhinav	India
3	103	Itika	America

	BookId	BookName	AuthID	PublishYear
1	1	Too Good To Be True	101	2024
2	2	Atomic Habits	102	2022
3	3	RS Aggarwal Aptitude	101	2019

	BookName	AuthName	AuthCountry
1	Too Good To Be True	Punit	India
2	Atomic Habits	Abhinav	India
3	RS Aggarwal Aptitude	Punit	India

[MEDIUM]



	DeptName	NO OF COURSES
1	COMPUTER SCIENCE	3
2	MATHEMATICS	2
3	MECHANICAL ENGINEERING	3
4	PHYSICS	1

	DeptName	NO OF COURSES
1	COMPUTER SCIENCE	3
2	MECHANICAL ENGINEERING	3

5. Learning Outcomes:

- Learnt how to define and create relational databases tables using CREATE TABLE syntax. Understand the use of data types like INT and VARCHAR.
- Gain practical knowledge of establishing a primary key for uniquely identifying records.
- Understand how to create and enforce foreign key relationships to maintain data integrity between related tables (Books - Authors).
- Develop the ability to use INNER JOIN to combine data from multiple tables based on a common key (e.g author_id)

- Understand how to design normalized tables with foreign key constraints for real-world entities like departments and courses.
- Gain proficiency in inserting multiple records into relational tables using INSERT INTO statement.
- Learnt how to use sub-queries with GROUP BY and HAVING to aggregate data and apply conditional logic.
- Apply filtering logic to retrieve records from a parent table based on the results from a subquery on a related child table.