DEPARTMENT OF
COMPUTER SCIENCE & ENGINEERING
CU CHANDIGARH UNIVERSITY
Discover. Learn. Empower.

# Experiment 1.1

**Student Name:** Divyansh Manhas

**UID:** 23BAI70239

**Branch:** BE-AIT-CSE

**Section/Group:** 23AIT_KRG-G1_A

**Semester:** 5th

**Date of Performance:** 23 July, 2025

**Subject Name:** ADBMS

**Subject Code:** 23CSP-333

## 1. Aim:

**EASY LEVEL PROBLEM:**
To create author and book tables linked by a foreign key, insert sample data, and use an INNER JOIN to display each book's title with its author's name and country, demonstrating basic SQL joins and relational design.

**MEDIUM LEVEL PROBLEM:**
Create normalized tables for departments and courses linked by a foreign key, insert sample data, use a subquery to count and filter departments offering more than two courses, and grant SELECT-only access to a specific user on the courses table, demonstrating subqueries, filtering, and access control in SQL.

## 2. Objective:

Design related tables for authors-books and departments-courses with foreign keys; insert sample data; use INNER JOIN to link books with authors, subqueries to find departments with over two courses, and grant SELECT-only access to a user—demonstrating core SQL concepts of relational design, data retrieval, and access control.

## 3. Theory:

This exercise involves foundational concepts of relational databases and SQL operations. Relational databases organize data into tables (called relations), where each table consists of rows (records or tuples) and columns (attributes or fields).

Tables are linked by keys: a primary key uniquely identifies each record in a table, and a foreign key establishes a relationship between tables by referring to a primary key in another table. This structure supports efficient data storage, retrieval, and ensures data integrity through referential constraints.

SQL (Structured Query Language) is the standard language used to create, manipulate, and query relational databases. Key SQL operations used here include:

- **Table creation and data insertion:** Defining tables with appropriate columns and constraints (like foreign keys), then inserting sample data into them.
- **INNER JOIN:** A fundamental join operation that links rows between two tables based on a matching key, allowing combined information to be retrieved (e.g., books linked with their authors).
- **Subqueries with aggregation:** Using nested queries to compute summary data (such as counting courses per department) and filtering results based on aggregate conditions.
- **Access control:** Managing database security by granting specific privileges (e.g., SELECT-only permission) to users, limiting their ability to modify data.

Together, these concepts demonstrate relational database design principles, how to model relationships using keys, retrieve meaningful combined data across tables, analyze data subsets through subqueries, and implement basic security measures in an SQL environment. This approach enables organized, consistent, and secure management of interconnected data.

This theory provides the conceptual foundation behind creating author-book and department-course database schemas, performing joins and subqueries for data retrieval, and applying access restrictions in practice.

## 4. Procedure:

### 1. Design Tables:

- Create an Author table to store author details (e.g., `AUTHOR_ID`, `AUTHOR_NAME`, Country).
- Create a Book table to store book details (e.g., `BOOK_ID`, Title, `AUTHOR_ID`), with a foreign key `AUTHOR_ID` referencing the Author table
- Create a department table to store department details (e.g., `DEPARTMENT_ID`, `DEPARTMENT_ID`).
- Create a Course table to store course details (e.g., `COURSE_ID`, `COURSE_NAME`, `DEPARTMENT_ID`), with a foreign key `DEPARTMENT_ID` referencing the Department table.

## 2. Insert Sample Data:

- Insert at least three records into the Author table.
- Insert at least three records into the Book table linking to authors through `AUTHOR_ID`.
- Insert five departments into the Department table.
- Insert at least ten courses into the Course table, distributed among the departments via `DEPARTMENT_ID`.

## 3. Perform SQL Operations:

- Use an INNER JOIN query to retrieve and display each book's title, corresponding author's name, and author's country by joining the Book and Author tables on `AUTHOR_ID`.
- Use a subquery with aggregation (COUNT) on the Course table grouped by `DEPARTMENT_ID` to find the number of courses per department.
- Filter the departments to retrieve only those having more than two courses based on the subquery result.

## 4. Apply Access Control:

- Grant SELECT permission on the Course table to a specific user to restrict data access to read-only.

## 5. Code Easy:

```
--Easy Level
-- Author-Book Relationship Using Joins and Basic SQL Operations

-- Creating / Using database

use Exp1;

-- Creating Tables


CREATE TABLE Tbl_Author(
Author_ID int PRIMARY KEY,
Author_Name varchar(30) NOT NULL DEFAULT('Unnammed'),
Author_Publication varchar(30));

CREATE TABLE Tbl_Books(
Book_ID int PRIMARY KEY,
Author_Id int NOT NULL,
Book_Name varchar(30) NOT NULL,
Book_Publication varchar(30),
Book_price int NOT NULL,
Units int DEFAULT(10)
FOREIGN KEY (Author_Id) REFERENCES Tbl_Author(Author_ID));
```

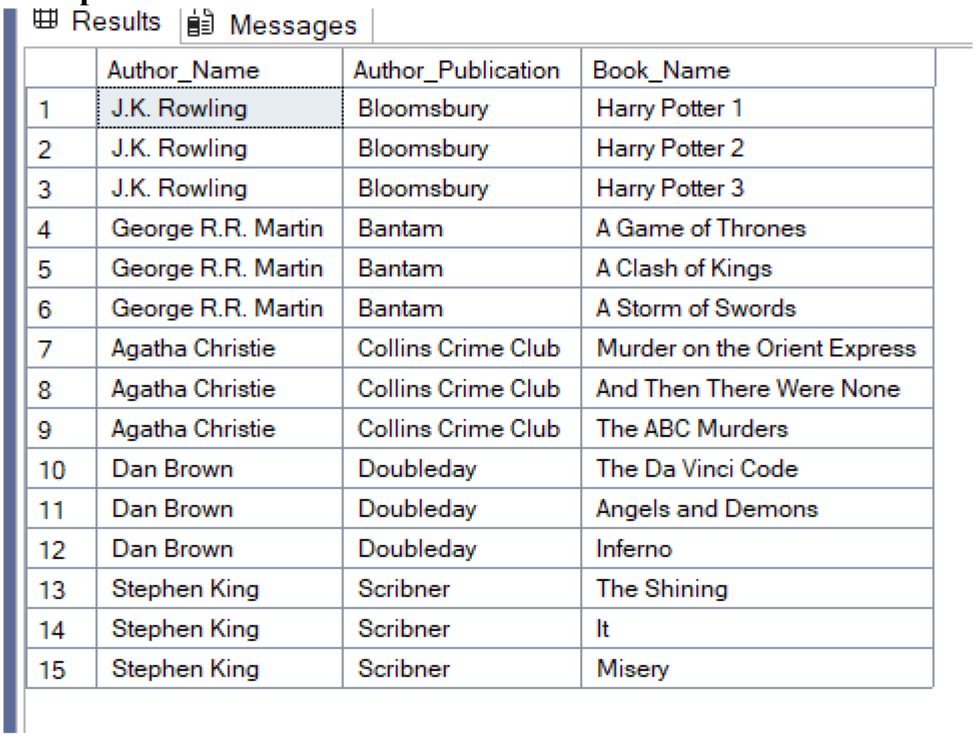
```sql
-- adding values

INSERT INTO Tbl_Author (Author_ID, Author_Name, Author_Publication) VALUES
(1, 'J.K. Rowling', 'Bloomsbury'),
(2, 'George R.R. Martin', 'Bantam'),
(3, 'Agatha Christie', 'Collins Crime Club'),
(4, 'Dan Brown', 'Doubleday'),
(5, 'Stephen King', 'Scribner');

INSERT INTO Tbl_Books (Book_ID, Author_Id, Book_Name, Book_Publication,
Book_price, Units) VALUES
(1, 1, 'Harry Potter 1', 'Bloomsbury', 450, 20),
(2, 1, 'Harry Potter 2', 'Bloomsbury', 470, 15),
(3, 1, 'Harry Potter 3', 'Bloomsbury', 500, 12),

(4, 2, 'A Game of Thrones', 'Bantam', 550, 10),
(5, 2, 'A Clash of Kings', 'Bantam', 580, 9),
(6, 2, 'A Storm of Swords', 'Bantam', 600, 11),

(7, 3, 'Murder on the Orient Express', 'Collins', 400, 14),
(8, 3, 'And Then There Were None', 'Collins', 420, 10),
(9, 3, 'The ABC Murders', 'Collins', 390, 13),

(10, 4, 'The Da Vinci Code', 'Doubleday', 500, 25),
(11, 4, 'Angels and Demons', 'Doubleday', 480, 22),
(12, 4, 'Inferno', 'Doubleday', 510, 19),

(13, 5, 'The Shining', 'Scribner', 530, 10),
(14, 5, 'It', 'Scribner', 600, 8),
(15, 5, 'Misery', 'Scribner', 490, 9);


--INNER JOINS

SELECT a.Author_Name, a.Author_Publication, b.Book_Name
FROM Tbl_Author as a
INNER JOIN Tbl_Books as b
ON a.Author_ID = b.Author_Id;
```

## 6. Code Medium:

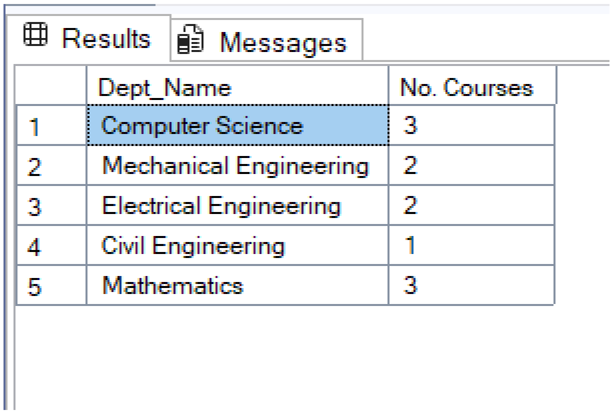
```sql
--Medium Level

--Department-Course Subquery and Access Control


CREATE TABLE Tbl_MDepartment (
    Dept_ID INT PRIMARY KEY,
    Dept_Name VARCHAR(100) NOT NULL UNIQUE,
    Dept_Head varchar(30)
);

CREATE TABLE Tbl_Course (
    Course_ID INT PRIMARY KEY,
    Course_Name VARCHAR(100) NOT NULL,
    Credits INT NOT NULL,
    Dept_ID INT NOT NULL,
    FOREIGN KEY (Dept_ID) REFERENCES Tbl_MDepartment(Dept_ID)
);

-- Adding values

INSERT INTO Tbl_MDepartment (Dept_ID, Dept_Name) VALUES
```

```sql
(1, 'Computer Science'),
(2, 'Mechanical Engineering'),
(3, 'Electrical Engineering'),
(4, 'Civil Engineering'),
(5, 'Mathematics');


INSERT INTO Tbl_Course (Course_ID, Course_Name, Credits, Dept_ID) VALUES
(101, 'Data Structures', 4, 1),
(102, 'Algorithms', 4, 1),
(103, 'Operating Systems', 4, 1),
(104, 'Thermodynamics', 3, 2),
(105, 'Fluid Mechanics', 3, 2),
(106, 'Circuits', 3, 3),
(107, 'Power Systems', 4, 3),
(108, 'Concrete Structures', 3, 4),
(109, 'Linear Algebra', 3, 5),
(110, 'Calculus', 4, 5),
(111, 'Probability & Statistics', 3, 5);


-- Sub Query

SELECT d.Dept_Name,
    (SELECT COUNT(*)
    FROM Tbl_Course as c
    WHERE c.Dept_ID = d.Dept_ID) as 'No. Courses'
FROM Tbl_MDepartment as d;

-- Filter

SELECT d.Dept_Name
FROM Tbl_MDepartment as d
WHERE (SELECT COUNT(*)
    FROM Tbl_Course as c
    WHERE c.Dept_ID = d.Dept_ID) > 2;

--Granted SELECT-only access on the courses table to a specific user.

CREATE LOGIN TEST_LOGIN_DIVYANSH
WITH PASSWORD = 'TESTLOGIN@70239';

CREATE USER TEST_LOGIN_DIVYANSH
FOR LOGIN TEST_LOGIN_DIVYANSH



EXECUTE AS USER = 'TEST_USER_DIVYANSH'

GRANT SELECT ON TBL_COURSE TO TEST_LOGIN_DIVYANSH
```

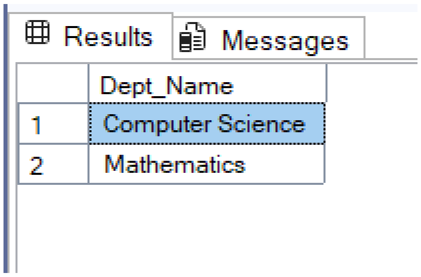
**7. Output:**



| | Author_Name | Author_Publication | Book_Name |
|---|---|---|---|
| 1 | J.K. Rowling | Bloomsbury | Harry Potter 1 |
| 2 | J.K. Rowling | Bloomsbury | Harry Potter 2 |
| 3 | J.K. Rowling | Bloomsbury | Harry Potter 3 |
| 4 | George R.R. Martin | Bantam | A Game of Thrones |
| 5 | George R.R. Martin | Bantam | A Clash of Kings |
| 6 | George R.R. Martin | Bantam | A Storm of Swords |
| 7 | Agatha Christie | Collins Crime Club | Murder on the Orient Express |
| 8 | Agatha Christie | Collins Crime Club | And Then There Were None |
| 9 | Agatha Christie | Collins Crime Club | The ABC Murders |
| 10 | Dan Brown | Doubleday | The Da Vinci Code |
| 11 | Dan Brown | Doubleday | Angels and Demons |
| 12 | Dan Brown | Doubleday | Inferno |
| 13 | Stephen King | Scribner | The Shining |
| 14 | Stephen King | Scribner | It |
| 15 | Stephen King | Scribner | Misery |



| | Dept_Name | No. Courses |
|---|---|---|
| 1 | Computer Science | 3 |
| 2 | Mechanical Engineering | 2 |
| 3 | Electrical Engineering | 2 |
| 4 | Civil Engineering | 1 |
| 5 | Mathematics | 3 |



| | Dept_Name |
|---|---|
| 1 | Computer Science |
| 2 | Mathematics |

## 8. Learning Outcomes:

1. Understand how to **design a relational schema** for a real-world system.
2. Practice **creating and linking tables** using SQL.
3. Use **JOINs to query multi-table data** meaningfully.
4. Implement **data access control** using GRANT/REVOKE.