DEPARTMENT OF
COMPUTER SCIENCE & ENGINEERING

CU CHANDIGARH UNIVERSITY Discover. Learn. Empower.

# Experiment 1.1

**Student Name:** Armaan Atri

**UID:** 23BAI70302

**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:

```
CREATE DATABASE AIT_1A

USE AIT_1A

-- Easy Level Problem Solution: Author-Book Relationship Using Joins and Basic
SQL Operations

CREATE TABLE TBL_AUTHOR
(
     AUTHOR_ID INT PRIMARY KEY,
     AUTHOR_NAME VARCHAR(MAX),
     COUNTRY VARCHAR(MAX)
)

CREATE TABLE TBL_BOOKS
(
     BOOK_ID INT PRIMARY KEY,
     BOOK_TITLE VARCHAR(MAX),
     AUTHORID INT
     FOREIGN KEY (AUTHORID) REFERENCES TBL_AUTHOR(AUTHOR_ID)
)
```

```sql
INSERT INTO TBL_AUTHOR (AUTHOR_ID, AUTHOR_NAME, COUNTRY) VALUES
(1, 'J.K. Rowling', 'United Kingdom'),
(2, 'George R.R. Martin', 'United States'),
(3, 'Haruki Murakami', 'Japan'),
(4, 'Isabel Allende', 'Chile'),
(5, 'Chinua Achebe', 'Nigeria'),
(6, 'Gabriel Garcia Marquez', 'Colombia'),
(7, 'Toni Morrison', 'United States'),
(8, 'Leo Tolstoy', 'Russia'),
(9, 'Jane Austen', 'United Kingdom'),
(10, 'Mark Twain', 'United States');


INSERT INTO TBL_BOOKS (BOOK_ID, BOOK_TITLE, AUTHORID) VALUES
(1, 'Harry Potter and the Sorcerer''s Stone', 1),
(2, 'A Game of Thrones', 2),
(3, 'Norwegian Wood', 3),
(4, 'The House of the Spirits', 4),
(5, 'Things Fall Apart', 5),
(6, 'One Hundred Years of Solitude', 6),
(7, 'Beloved', 7),
(8, 'War and Peace', 8),
(9, 'Pride and Prejudice', 9),
(10, 'Adventures of Huckleberry Finn', 10);


CREATE TABLE



SELECT B.BOOK_TITLE AS [Book Title], A.AUTHOR_NAME AS [Author Name], A.COUNTRY
AS [Country]
FROM TBL_BOOKS AS B
INNER JOIN
TBL_AUTHOR AS A
ON
B.AUTHORID = A.AUTHOR_ID


-- Medium Level Problem Solution: Department-Course Subquery and Access Control

CREATE TABLE TBL_DEPARTMENT
(
     DEPARTMENT_ID INT PRIMARY KEY,
     DEPARTMENT_NAME VARCHAR(100) NOT NULL
)

CREATE TABLE TBL_COURSE
(
     COURSE_ID INT PRIMARY KEY,
     COURSE_NAME VARCHAR(100) NOT NULL,
     DEPARTMENT_ID INT,
     FOREIGN KEY (DEPARTMENT_ID) REFERENCES TBL_DEPARTMENT(DEPARTMENT_ID)
)

INSERT INTO TBL_DEPARTMENT (DEPARTMENT_ID, DEPARTMENT_NAME) VALUES
(1, 'Computer Science'),
(2, 'Mathematics'),
(3, 'Physics'),
(4, 'Chemistry'),
(5, 'English Literature');
```

```sql
INSERT INTO TBL_COURSE (COURSE_ID, COURSE_NAME, DEPARTMENT_ID) VALUES
(1, 'Data Structures', 1),
(2, 'Operating Systems', 1),
(3, 'Algorithms', 1),
(4, 'Calculus', 2),
(5, 'Linear Algebra', 2),
(6, 'Quantum Mechanics', 3),
(7, 'Electromagnetism', 3),
(8, 'Organic Chemistry', 4),
(9, 'Physical Chemistry', 4),
(10, 'Shakespearean Literature', 5),
(11, 'Modern Poetry', 5);

-- Query Used to find the Count of total courses in each department

SELECT COUNT(COURSE_NAME) AS Total, DEPARTMENT_NAME AS [Department Name]
FROM TBL_COURSE
INNER JOIN TBL_DEPARTMENT ON
TBL_COURSE.DEPARTMENT_ID = TBL_DEPARTMENT.DEPARTMENT_ID
GROUP BY TBL_DEPARTMENT.DEPARTMENT_NAME

--Subquery used to filter and retrieve only those departments that offer more
than two courses.

SELECT DEPARTMENT_NAME
FROM TBL_DEPARTMENT
WHERE DEPARTMENT_ID IN
(
    SELECT DEPARTMENT_ID
    FROM TBL_COURSE
    GROUP BY DEPARTMENT_ID
    HAVING COUNT(*) > 2
)

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

CREATE LOGIN TEST_LOGIN_ARMAAN
WITH PASSWORD = 'TESTLOGIN@123ARMAAN';

CREATE USER TEST_LOGIN_ARMAAN
FOR LOGIN TEST_LOGIN_ARMAAN


EXECUTE AS USER = 'TEST_USER_ARMAAN'

GRANT SELECT ON TBL_COURSE TO TEST_LOGIN_ARMAAN
```

## 6. Output:

Results Messages

| | Book Title | Author Name | Country |
|---|---|---|---|
| 1 | Harry Potter and the Sorcerer's Stone | J.K. Rowling | United Kingdom |
| 2 | A Game of Thrones | George R.R. Martin | United States |
| 3 | Norwegian Wood | Haruki Murakami | Japan |
| 4 | The House of the Spirits | Isabel Allende | Chile |
| 5 | Things Fall Apart | Chinua Achebe | Nigeria |
| 6 | One Hundred Years of Solitude | Gabriel Garcia Marquez | Colombia |
| 7 | Beloved | Toni Morrison | United States |
| 8 | War and Peace | Leo Tolstoy | Russia |
| 9 | Pride and Prejudice | Jane Austen | United Kingdom |
| 10 | Adventures of Huckleberry Finn | Mark Twain | United States |

Results Messages

| | Total | Department Name |
|---|---|---|
| 1 | 2 | Chemistry |
| 2 | 3 | Computer Science |
| 3 | 2 | English Literature |
| 4 | 2 | Mathematics |
| 5 | 2 | Physics |

Results Messages

| | DEPARTMENT_NAME |
|---|---|
| 1 | Computer Science |

## 7. 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.