EXPERIMENT- 01

Student Name: Devjot Singh **UID: 23BCS10864**

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

Semester: 05 **Date of Performance: 22/07/25**

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

Easy-Level Problem

1. Aim:

i. Design two tables — one for storing author details and the other for book details.

ii. Insert at least three records in each table.

iii. Perform an INNER JOIN to link each book with its author using the common author ID and Select the book title, author name, and author's country.

2. Objective:

- Understand basic table creation with primary and foreign keys.
- Practice inserting data into relational tables.
- Implement JOIN operations to fetch combined data from multiple tables.

3. DBMS script and output:

Step 1: Create 'authors' table

```
CREATE TABLE authors (
   author_id INT PRIMARY KEY,
   author_name VARCHAR(100),
   country VARCHAR(50)
);
```

Step 2: Create 'books' table with foreign key to 'authors'

```
CREATE TABLE books (
```

```
    book_id INT PRIMARY KEY,
    title VARCHAR(100),
    author_id INT,
    FOREIGN KEY (author_id) REFERENCES authors(author_id)
);
```

Step 3: Insert sample data into 'authors'

```
INSERT INTO authors (author_id, author_name, country) VALUES
(1, 'George Orwell', 'United Kingdom'),
(2, 'Haruki Murakami', 'Japan'),
(3, 'Chinua Achebe', 'Nigeria');
```

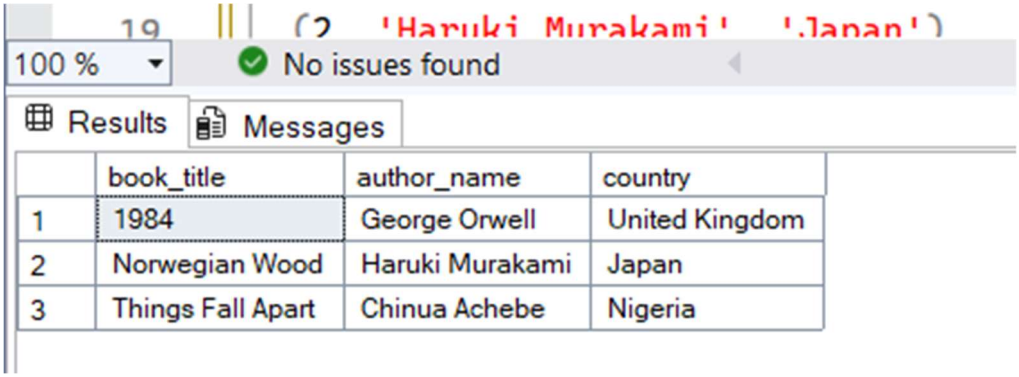
Step 4: Insert sample data into 'books'

```
INSERT INTO books (book_id, title, author_id) VALUES
(101, '1984', 1),
(102, 'Norwegian Wood', 2),
(103, 'Things Fall Apart', 3);
```

Step 5: Perform INNER JOIN to get desired result

```
SELECT
    b.title AS book_title,
    a.author_name,
    a.country
FROM
    books b
INNER JOIN
    authors a ON b.author_id = a.author_id;
```

4. Output:



	book_title	author_name	country
1	1984	George Orwell	United Kingdom
2	Norwegian Wood	Haruki Murakami	Japan
3	Things Fall Apart	Chinua Achebe	Nigeria

Medium-Level Problem

1. Aim:

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 the courses table to a specific user.

2. Objective:

- Create normalized tables with proper foreign key relationships.
- Use subqueries to count and filter relational data.
- Implement user-level access control using GRANT statement.

2. DBMS script and output:

STEP 1: Create 'departments' table

```
CREATE TABLE departments (
    department_id INT PRIMARY KEY,
    department_name VARCHAR(100)
);
```

STEP 2: Create 'courses' table with foreign key

```
CREATE TABLE courses (
    course_id INT PRIMARY KEY,
    course_name VARCHAR(100),
    department_id INT,
    FOREIGN KEY (department_id) REFERENCES departments(department_id)
);
```

STEP 3: Insert sample data into 'departments'

```
INSERT INTO departments (department_id, department_name) VALUES
(1, 'Computer Science'),
(2, 'Mathematics'),
(3, 'Physics'),
(4, 'Biology'),
(5, 'Chemistry');
```

STEP 4: Insert sample data into 'courses'

```
INSERT INTO courses (course_id, course_name, department_id) VALUES
(101, 'Data Structures', 1),
(102, 'Algorithms', 1),
(103, 'Databases', 1),
(201, 'Linear Algebra', 2),
(202, 'Calculus', 2),
(301, 'Quantum Mechanics', 3),
(302, 'Electrodynamics', 3),
(303, 'Thermodynamics', 3),
(401, 'Genetics', 4),
(501, 'Organic Chemistry', 5);
```

STEP 5: Use subquery to find departments with more than 2 courses

```
SELECT department_name
FROM departments
WHERE department_id IN (
   SELECT department_id
   FROM courses
   GROUP BY department_id
   HAVING COUNT(*) > 2
);
```

STEP 6: Create SQL Server login (for entire server)

```
IF NOT EXISTS (SELECT * FROM sys.server_principals WHERE name =
'student_user')
BEGIN
   CREATE LOGIN student_user WITH PASSWORD = 'StrongPass123!';
END;
```

STEP 7: Create database user for current database

```
IF NOT EXISTS (SELECT * FROM sys.database_principals WHERE name =
'student_user')
BEGIN
   CREATE USER student_user FOR LOGIN student_user;
END;
```

STEP 8: Grant SELECT access on 'courses' table to a specific user

```
GRANT SELECT ON courses TO student_user;
```

5. Output: