

Experiment-1

1) Aim->

To design and implement a basic book management system using SQL by:

1. Creating two related tables — Authors and Books — with appropriate constraints.
2. Inserting valid sample records into both tables while maintaining referential integrity.
3. Querying the data using an INNER JOIN to retrieve book titles along with corresponding author names and countries.

2) Objective->

A **relational database** is a structured collection of data organized into tables. Each table stores information about a specific entity, and relationships between entities are established through keys.

1. **Authors Table** – stores information about authors such as their ID, name, and country.
2. **Books Table** – stores details about books such as book ID, title, and the author who wrote it.

The relationship between **Authors** and **Books** is **one-to-many**, meaning one author can write multiple books, but each book is written by only one author. This is implemented using a **foreign key**:

- The author_id in the **Books** table refers to the author_id in the **Authors** table.

To maintain **data integrity**, we use:

- **Primary Keys** to uniquely identify each record in a table.
- **Foreign Keys** to ensure that a book can only be linked to an existing author.

3) Procedure/Algorithm->

Step-1: Design of database Schema:

1.1. Create a table named Authors with the following columns:

- author_id (Primary Key, INT)
- name (VARCHAR(50))
- country (VARCHAR(50))

1.2. Create a table named Books with the following columns:

- book_id (Primary Key, INT)
- title (VARCHAR(100))
- author_id (Foreign Key, INT) referencing Authors(author_id)

Step-2: Insert Sample Data:

2.1. Insert at least **three records** into the Authors table with meaningful names and countries.

2.2. Insert at least **three records** into the Books table, making sure that each author_id used exists in the Authors table.

Step -3: Perform Data Retrieval Using JOIN:

3.1. Write an SQL INNER JOIN query to retrieve the following combined information:

- Book title
- Author name
- Author country

3.2. Execute the query to display the output that links each book to its respective author.

Step 4: Verify Integrity and Output

4.1. Ensure that all foreign key constraints are satisfied (i.e., no orphaned books).

4.2. Confirm that the final result shows only valid and related records from both tables.

5)Problem Statement->

Problem statement-1:

Design a basic Book Management System by creating two relational tables: Authors and Books. The system must represent a one-to-many relationship, where one author can write multiple books, but each book is associated with only one author. Use appropriate primary key and foreign key constraints to maintain referential integrity between the tables.

Query-1:

```
CREATE TABLE Authors(  
  
author_id INT PRIMARY KEY,  
  
name VARCHAR(50),  
  
country VARCHAR(50) );  
  
CREATE TABLE Books(  
  
book_id INT PRIMARY KEY,  
  
title VARCHAR(100),  
  
author_id INT,  
  
FOREIGN KEY (author_id) REFERENCES Authors(author_id) );  
  
DESC Authors;  
  
DESC Books;
```

Output-1:

The screenshot displays a web application interface for a SQL lab. The interface is divided into three main sections:

- Left Sidebar:** Contains navigation links for Home, Dashboard, Feedback Requests, Reports, Student Reports, Learning, AI Mentor (Beta), Courses (highlighted), Classes, Editor, Lab, Assessment, Nimbus, and Nimbus Submissions.
- Main Content Area:**
 - Header:** "Create Author and Book Tables using DDL Commands" with a score of 5 and difficulty of easy.
 - Problem Statement:** "You are tasked with designing a basic book management system. Create two tables — **Authors** and **Books** — to represent a one-to-many relationship (one author can write multiple books). Use proper **primary and foreign key constraints** while designing the schema."
 - Input Format:**
 - Table Authors with columns:**
 - author_id (INT, Primary Key)
 - name (VARCHAR(50))
 - country (VARCHAR(50))
 - Table Books with columns:**
 - book_id (INT, Primary Key)
 - title (VARCHAR(100))
 - author_id (INT, Foreign Key referencing Authors)
- Right Panel:**
 - SQL Editor:** Contains the following SQL code:

```
1 create table authors_mitakshu(  
2   author_id INT PRIMARY KEY,  
3   name VARCHAR(50),  
4   country VARCHAR(50)  
5 );  
6  
7  
8 create table books_mitakshu(  
9   book_id INT PRIMARY KEY,  
10  title VARCHAR(100),  
11  author_id INT,  
12  FOREIGN key (author_id) REFERENCES authors_mitakshu(author_id)  
13 );  
14  
15 desc authors_mitakshu;  
16 desc books_mitakshu;
```
 - Test & Results:** A table showing test case results.

Test Case	Status	Test Case Info
Test Case 1	Passed	

Problem statement-2:

After creating the Authors and Books tables, your next task is to insert sample records into both tables. You must add at least three authors and three books, ensuring that each book correctly references an existing author through the author_id field.

Query-2:

```
insert into Authors(author_id,name,country)
values(1,'Ashish','India'),
(2,'Smaran','USA'),
(3,'Vaibhav','UK');
insert into Books(book_id,title,author_id)
values(101,'Data Science Basics',1),
(102,'AI in Education',2),
(103,'SQL Simplified',1);
select * from Authors;select * from Books;
```

Output-2:

The screenshot shows the ByteXL SQL editor interface. On the left is a sidebar with navigation links: Home, Dashboard, Feedback Requests, Reports, Student Reports, Learning, AI Mentor (Beta), Courses (highlighted), Classes, Editor, Lab, Assessment, Nimbus, and Nimbus Submissions. The main area is divided into three sections: 'Output Format', 'Constraints', and 'Test & Results'.

Output Format:

Authors Table:

author_id	name	country
1	Ashish	India
2	Smaran	USA
3	Vaibhav	UK

Books Table:

book_id	title	author_id
101	Data Science Basics	1
102	AI in Education	2
103	SQL Simplified	1

Constraints:

- Insert meaningful names and countries (e.g., Ashish, Smaran, Vaibhav).
- Insert book titles that are easy to associate with those authors.
- Use valid foreign keys.

Test & Results:

Custom Input

Test Cases

Test Case	Status	Test Case Info
Test Case 1	Passed	

The SQL editor on the right shows the following code:

```

1
2 insert into Authors(author_id,name, country)
3 values(1,'Ashish','India'),(2,'Smaran','USA'),(3,'Vaibhav','UK');
4
5 insert into Books(book_id,title,author_id)
6 values(101,'Data Science Basics',1),(102,'AI in Education',2),(103,'SQL Simplified',1);
7 select * from Authors;
8 select * from Books;

```

A green checkmark and 'Answer submitted!' message are visible at the top of the SQL editor.

Problem statement-3:

Using the existing Authors and Books tables, your task is to retrieve a list of books along with their corresponding author's name and country. This requires performing an INNER JOIN on the author_id field to combine data from both tables based on their relationship.

Query-3:

```
select Books.title,Authors.name,Authors.country from Books
```

```
Inner Join Authors ON Books.author_id=Authors.author_id;
```

Output-3:

The screenshot shows the ByteXL lab interface. The sidebar on the left contains navigation links: Home, Dashboard, Feedback Requests, Reports, Student Reports, Learning, AI Mentor (Beta), Courses (highlighted), Classes, Editor, Lab, Assessment, Nimbus, Nimbus Submissions, and Nimbus Announcements. The main content area displays a problem titled "Retrieve Book Titles Along with Author Information Using INNER JOIN" with a score of 5 and difficulty of easy. The problem statement asks to retrieve book titles along with author names and countries using an INNER JOIN. The input format specifies pre-existing Authors and Books tables. The Authors table has columns: author_id (INT, Primary Key), name (VARCHAR(50)), and country (VARCHAR(50)). The Books table has columns: book_id (INT, Primary Key), title (VARCHAR(100)), and author_id (INT, Foreign Key referencing Authors). The SQL editor shows the following query:

```
1 .title,Authors.name,Authors.country from Books Inner Join Authors ON Books.author_id=Authors.author_id;
```

Below the editor, the "Test & Results" section shows a table with test cases:

Test Case	Status	Test Case Info
Test Case 1	Passed	

6) Learning Outcomes->

- Understand how to model one-to-many relationships using SQL.
- Learn to create database tables with primary and foreign key constraints.
- Gain skills in inserting valid data while maintaining referential integrity.
- Practice writing INNER JOIN queries to combine related data.
- Develop the ability to query and interpret results from relational tables.