

# **EXPERIMENT-01**

## **AIM:**

- (i) Create Author and Book Tables using DDL Commands
- (ii) Insert Sample Records into Author and Book Tables
- (iii) Retrieve Book Titles Along with Author Information Using INNER JOIN

## **OBJECTIVE:**

The objective of this experiment is to understand the core components of database schema design, particularly the creation and linking of tables using primary and foreign keys.

It also aims to strengthen the practical knowledge of DDL (Data Definition Language) and DML (Data Manipulation Language) operations, including table creation, data insertion, and joining tables to retrieve meaningful insights.

By performing this experiment on the ByteSQL platform, students will gain hands-on experience in relational database management and writing efficient SQL queries for real-world data modeling scenarios.

## **PROCEDURE:**

- Launch the ByteSQL platform to perform SQL operations in an interactive environment.
- Use CREATE TABLE statements to define the Authors table with the following fields:

i. author\_id (Primary Key)

ii. name (VARCHAR)

iii. country (VARCHAR)

- Define the Books table using CREATE TABLE with the fields:

i. book\_id (Primary Key)

ii. title (VARCHAR)

iii. author\_id (Foreign Key referencing Authors.author\_id)

- Insert sample data into the Authors table using INSERT INTO commands with at least three distinct authors.
- Insert sample data into the Books table using INSERT INTO commands while ensuring each book is linked to a valid author via the author\_id foreign key.
- Use an INNER JOIN SQL query to combine both tables and retrieve the book titles, author names, and author countries, matching records based on the common author\_id.
- Validate the results by ensuring that each book is correctly displayed with its corresponding author's information as per the join condition.

# 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));

DESCRIBE Authors;

DESCRIBE Books;

## OUTPUT 1:

The screenshot shows the ByteXL IDE interface. On the left is a sidebar with navigation links. The main area is titled "Create Author and Book Tables using DDL Commands" with a score of 5/10 (Difficulty: easy). It contains the problem statement, input/output formats, and constraints. The "Test & Results" tab is active, showing the SQL code and the output of the queries.

```
1 create table authors(author_id int primary key not null, name varchar(50), country varchar(50));
2 create table books(book_id int primary key not null, title varchar(100), author_id int, foreign key(author_id) References Authors(author_id));
3 describe authors;
4 describe books;
```

**Output:**

| Field     | Type        | Null | Key | Default | Extra |
|-----------|-------------|------|-----|---------|-------|
| author_id | int         | NO   | PRI | NULL    |       |
| name      | varchar(50) | YES  |     | NULL    |       |
| country   | varchar(50) | YES  |     | NULL    |       |

  

| Field     | Type         | Null | Key | Default | Extra |
|-----------|--------------|------|-----|---------|-------|
| book_id   | int          | NO   | PRI | NULL    |       |
| title     | varchar(100) | YES  |     | NULL    |       |
| author_id | int          | YES  | FK  | NULL    |       |

364 ms

## TEST CASE 1:

The screenshot shows the ByteXL IDE interface with the "Test & Results" tab active. It displays the SQL code and the output of the queries, which matches the output shown in the previous screenshot.

```
1 create table authors(author_id int primary key not null, name varchar(50), country varchar(50));
2 create table books(book_id int primary key not null, title varchar(100), author_id int, foreign key(author_id) References Authors(author_id));
3 describe authors;
4 describe books;
```

**Output:**

| Field     | Type        | Null | Key | Default | Extra |
|-----------|-------------|------|-----|---------|-------|
| author_id | int         | NO   | PRI | NULL    |       |
| name      | varchar(50) | YES  |     | NULL    |       |
| country   | varchar(50) | YES  |     | NULL    |       |

  

| Field     | Type         | Null | Key | Default | Extra |
|-----------|--------------|------|-----|---------|-------|
| book_id   | int          | NO   | PRI | NULL    |       |
| title     | varchar(100) | YES  |     | NULL    |       |
| author_id | int          | YES  | FK  | NULL    |       |

364 ms

**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 VALUES (1, 'Ashish', 'India'), (2, 'Smaran', 'USA'), (3, 'Vaibhav', 'UK');

INSERT INTO Books 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 interface with the problem statement and the SQL query execution results. The problem statement is: "After creating the Authors and Books tables, your next task is to insert sample records. Insert at least 3 authors and 3 books, ensuring books reference valid authors using the foreign key." The input format is: "Pre-existing Authors and Books table structures from Problem 1." The output format is: "Pre-existing Authors and Books table structures from Problem 1." The SQL query is: "INSERT INTO Authors VALUES (1, 'Ashish', 'India'), (2, 'Smaran', 'USA'), (3, 'Vaibhav', 'UK'); INSERT INTO Books VALUES (101, 'Data Science Basics', 1), (102, 'AI in Education', 2), (103, 'SQL Simplified', 1); SELECT \* FROM Authors; SELECT \* FROM Books;". The output shows the Authors table with 3 rows and the Books table with 3 rows.

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

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

## TEST CASE 2:

The screenshot shows the ByteXL interface with the problem statement and the SQL query execution results. The problem statement is: "After creating the Authors and Books tables, your next task is to insert sample records. Insert at least 3 authors and 3 books, ensuring books reference valid authors using the foreign key." The input format is: "Pre-existing Authors and Books table structures from Problem 1." The output format is: "Pre-existing Authors and Books table structures from Problem 1." The SQL query is: "INSERT INTO Authors VALUES (1, 'Ashish', 'India'), (2, 'Smaran', 'USA'), (3, 'Vaibhav', 'UK'); INSERT INTO Books VALUES (101, 'Data Science Basics', 1), (102, 'AI in Education', 2), (103, 'SQL Simplified', 1); SELECT \* FROM Authors; SELECT \* FROM Books;". The output shows the Authors table with 3 rows and the Books table with 3 rows.

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

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

**Problem Statement 3:** Given two tables, Authors and Books, retrieve the titles of all books along with their author's name and country. This involves creating tables, inserting data, and using an INNER JOIN to combine records based on author\_id.

### 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 SQL editor interface. The left sidebar contains a navigation menu with options like Dashboard, Feedback Requests, Reports, Student Reports, Learning, AI Mentor (Beta), Courses, Classes, Editor, Lab, Assessment, Nimbus, Nimbus Submissions, and Nimbus Apps. The main content area displays the problem statement for 'Retrieve Book Titles Along with Author Information Using INNER JOIN'. The problem statement asks to retrieve the titles of all books along with their author's name and country using an INNER JOIN. The input format specifies the structure of the Authors and Books tables. The output format shows a list of books with their title, author's name, and country. The SQL query is entered in the editor: `1 select b.title, a.name, a.country from Authors a inner join Books b on a.author_id = b.author_id;`. The output is displayed as a table with 3 rows and 3 columns: title, name, and country. The output shows books like 'Data Science Basics' by Ashish from India, 'AI in Education' by Snehan from USA, and 'SQL Simplified' by Ashish from India.

**Problem Statement**

Given two tables, Authors and Books, retrieve the titles of all books along with their author's name and country. This involves creating tables, inserting data, and using an INNER JOIN to combine records based on author\_id.

**Input Format:**

- Pre-existing Authors and Books table structures from Problem 1.

**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)

**Output Format:**

- A list of books with their title, name of the author, and country of the author.

**Constraints:**

**SQL Query:**

```
1 select b.title, a.name, a.country from Authors a inner join Books b on a.author_id = b.author_id;
```

**Output:**

| title               | name   | country |
|---------------------|--------|---------|
| Data Science Basics | Ashish | India   |
| AI in Education     | Snehan | USA     |
| SQL Simplified      | Ashish | India   |

### TEST CASE 3:

The screenshot shows the ByteXL SQL editor interface. The left sidebar contains a navigation menu with options like Dashboard, Feedback Requests, Reports, Student Reports, Learning, AI Mentor (Beta), Courses, Classes, Editor, Lab, Assessment, Nimbus, Nimbus Submissions, and Nimbus Apps. The main content area displays the problem statement for 'Retrieve Book Titles Along with Author Information Using INNER JOIN'. The problem statement asks to retrieve the titles of all books along with their author's name and country using an INNER JOIN. The input format specifies the structure of the Authors and Books tables. The output format shows a list of books with their title, author's name, and country. The SQL query is entered in the editor: `1 select b.title, a.name, a.country from Authors a inner join Books b on a.author_id = b.author_id;`. The output is displayed as a table with 3 rows and 3 columns: title, name, and country. The output shows books like 'Data Science Basics' by Ashish from India, 'AI in Education' by Snehan from USA, and 'SQL Simplified' by Ashish from India.

**Problem Statement**

Given two tables, Authors and Books, retrieve the titles of all books along with their author's name and country. This involves creating tables, inserting data, and using an INNER JOIN to combine records based on author\_id.

**Input Format:**

- Pre-existing Authors and Books table structures from Problem 1.

**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)

**Output Format:**

- A list of books with their title, name of the author, and country of the author.

**Constraints:**

**SQL Query:**

```
1 select b.title, a.name, a.country from Authors a inner join Books b on a.author_id = b.author_id;
```

**Output:**

| title               | name   | country |
|---------------------|--------|---------|
| Data Science Basics | Ashish | India   |
| AI in Education     | Snehan | USA     |
| SQL Simplified      | Ashish | India   |