

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, the problem statement is displayed, including the task description, input format, and output format. The main editor shows the SQL query:

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

 The 'Test & Results' section is active, showing the 'Custom Input' field and the 'Output' section. The output displays the schema for both tables in a tabular format.

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	MUL	NULL	

TEST CASE 1:

The screenshot shows the ByteXL IDE interface with the same SQL query as before. The 'Test & Results' section is active, and the 'Test Cases' tab is selected. It shows a table with the 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 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 IDE interface. On the left, the problem statement is displayed, including the input and output formats. The input format specifies the Authors table structure: author_id, name, country. The output format shows the Authors table with three rows: (1, 'Ashish', 'India'), (2, 'Smaran', 'USA'), and (3, 'Vaibhav', 'UK'). The Books table structure is also shown: book_id, title, author_id. The main editor displays the SQL query:

```
1 INSERT INTO Authors VALUES (1, 'Ashish', 'India'), (2, 'Smaran', 'USA'), (3, 'Vaibhav', 'UK');
2 INSERT INTO Books VALUES (101, 'Data Science Basics', 1), (102, 'AI in Education', 2), (103, 'SQL Simplified', 1);
```

 The output section shows the results of the query, displaying the Authors and Books tables. The Authors table has three rows: (1, 'Ashish', 'India'), (2, 'Smaran', 'USA'), and (3, 'Vaibhav', 'UK'). The Books table has three rows: (101, 'Data Science Basics', 1), (102, 'AI in Education', 2), and (103, 'SQL Simplified', 1). The output is formatted as a table with columns: author_id, name, country, book_id, title, and author_id.

TEST CASE 2:

The screenshot shows the ByteXL IDE interface with the test case results. The problem statement is displayed on the left. The main editor shows the SQL query:

```
1 INSERT INTO Authors VALUES (1, 'Ashish', 'India'), (2, 'Smaran', 'USA'), (3, 'Vaibhav', 'UK');
2 INSERT INTO Books VALUES (101, 'Data Science Basics', 1), (102, 'AI in Education', 2), (103, 'SQL Simplified', 1);
3 SELECT * FROM Authors;
4 SELECT * FROM Books;
```

 The test case results are shown in the output section. The test case is named 'Test Case 1' and has a status of 'Passed'. The output is formatted as a table with columns: Test Case, Status, and Test Case Info.

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 interface for Problem Statement 3. The problem statement asks to retrieve book titles along with author information using an INNER JOIN. The input format specifies two tables: Authors (author_id, name, country) and Books (book_id, title, author_id). The output format requires a list of books with their title, author name, and country. The SQL query is:

```
1 SELECT Books.title, Authors.name, Authors.country
2 FROM Books INNER JOIN Authors ON Books.author_id = Authors.author_id;
3
```

 The output shows a table with 3 rows: Data Science Basics (Ashish, India), AI in Education (Aarav, USA), and SQL Simplified (Ashish, India). The execution time is 201 ms.

title	name	country
Data Science Basics	Ashish	India
AI in Education	Aarav	USA
SQL Simplified	Ashish	India

TEST CASE 3:

The screenshot shows the ByteXL interface for Problem Statement 3, displaying the test case results. The test case is named 'Test Case 1' and has a status of 'Passed'. The output format requires a list of books with their title, author name, and country. The test case results show that the query successfully retrieved the data.

Test Case	Status	Test Case Info
Test Case 1	Passed	

