

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 SQL editor interface. On the left is a sidebar with navigation links: Home, Dashboard, Feedback Requests, Reports, Student Reports, Learning, AI Mentor (Beta), Courses (selected), Classes, Editor, Lab, Assessment, Nimbus, Nimbus Submissions, Nimbus Apps, Community, Organizations, Branches, Batches, and Classrooms.

The main content area is titled "Create Author and Book Tables using DDL Commands" with a score of 5 and difficulty of easy. It contains a "Problem Statement" section with a description of the task, an "Input Format" section listing the columns for the Authors and Books tables, an "Output Format" section, and a "Constraints" section.

The "Input Format" section lists the columns for the Authors table: author_id (INT, Primary Key), name (VARCHAR(50)), and country (VARCHAR(50)). It also lists the columns for the Books table: book_id (INT, Primary Key), title (VARCHAR(100)), and author_id (INT, Foreign Key referencing Authors).

The "Output Format" section states that the output should be the Authors and Books tables created, with a description of the table.

The "Constraints" section lists the constraints: the author_id in Books must exist in the Authors table, use appropriate data types and constraints, and name and country should allow up to 50 characters.

The "Sample Input" section shows the SQL query to create the tables. The "Sample Output" section shows the output of the DESCRIBE command.

The "Test & Results" section shows the executed SQL query and its results. The query is:

```
1 -- Write your Query here
2 create table authors(author_id int primary key not null, name varchar(50),country varchar(50));
3 create table books(book_id int primary key not null, title varchar(100),author_id int, foreign key(author_id) references authors(author_id));
4 describe authors;
5 describe books;
```

 The results show the structure of the Authors and Books tables.

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

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

TEST CASE 1:

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 for the problem "Insert Sample Records into Author and Book Tables". The problem statement requires inserting at least 3 authors and 3 books, ensuring valid foreign key references. The input format shows pre-existing table structures for Authors and Books. The output format shows the expected data after insertion. The SQL query is executed, and the results are displayed in a table format.

Problem Statement

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.

Input Format:

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

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.

SQL Query:

```
1 -- Write your query here
2 insert into authors values(1,'Ashish','India'),(2,'Smaran','USA'),(3,'Vaibhav','UK');
3 insert into books values(101,'Data Science Basics',1),(102,'AI in Education',2),(103,'SQL Simplified',1);
4 select * from authors;
5 select * from books;
```

Test & Results

Custom input:

Test Cases:

Run Code

Output:

```
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 for the problem "Insert Sample Records into Author and Book Tables". The problem statement requires inserting at least 3 authors and 3 books, ensuring valid foreign key references. The input format shows pre-existing table structures for Authors and Books. The output format shows the expected data after insertion. The SQL query is executed, and the results are displayed in a table format.

Problem Statement

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.

Input Format:

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

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 |

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;
```

Test & Results

Custom input:

Test Cases:

Run Code

Test Case Results:

| Test Case | Status | Test Case Info |
|-------------|--------|----------------|
| Test Case 1 | Pass | |

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 displays the byteXL SQL editor interface. On the left is a sidebar with navigation links: Home, Dashboard, Feedback Requests, Reports, Student Reports, Learning, All Mentor (Beta), Courses (highlighted), Classes, Editor, Lab, Assessment, Nimbus, Nimbus Submissions, Nimbus Apps, Community, Organizations, Branches, Batches, and Classrooms. The main content area is titled "Retrieve Book Titles Along with Author Information Using INNER JOIN" with a score of 5/1 and difficulty of easy. It contains a "Problem Statement" section with the text: "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." Below this is the "Input Format" section, which states: "Pre-existing Authors and Books table structures from Problem 1." It then lists the columns for the Authors table (author_id, name, country) and the Books table (book_id, title, author_id). The "Output Format" section states: "A list of books with their title, name of the author, and country of the author." The "Constraints" section lists: "Each book must be linked to one valid author.", "Each author can be linked to one or more books.", "No NULLs are allowed in the author_id field of the Books table.", and "Use the same data as shown in the sample table." On the right, the SQL editor shows the query: "select b.title , a.name , a.country from Authors a inner join Books b on a.author_id=b.author_id;". The interface also includes a "Test & Results" button and a "Submit" button.

TEST CASE 3:

2.9hr

1

2

3

Retrieve Book Titles Along with Author Information Using INNER JOIN

Score: 5 | Difficulty: easy

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

SQL

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

Test & Results

Custom Input

Test Cases

| Test Case | Status | Test Case Info |
|-------------|--------|----------------|
| Test Case 1 | Passed | |