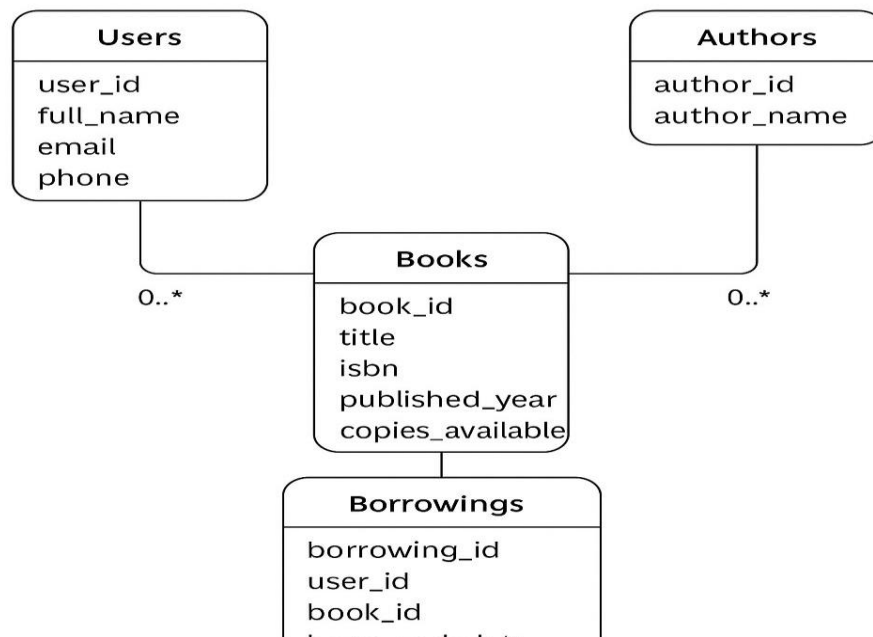


# Library Management System – Database Design Report

## Introduction

The purpose of this project is to design and implement a relational database for a Library Management System. The database allows the library to store information about users, books, authors, and borrowings. It enforces data integrity using constraints and relationships between entities.

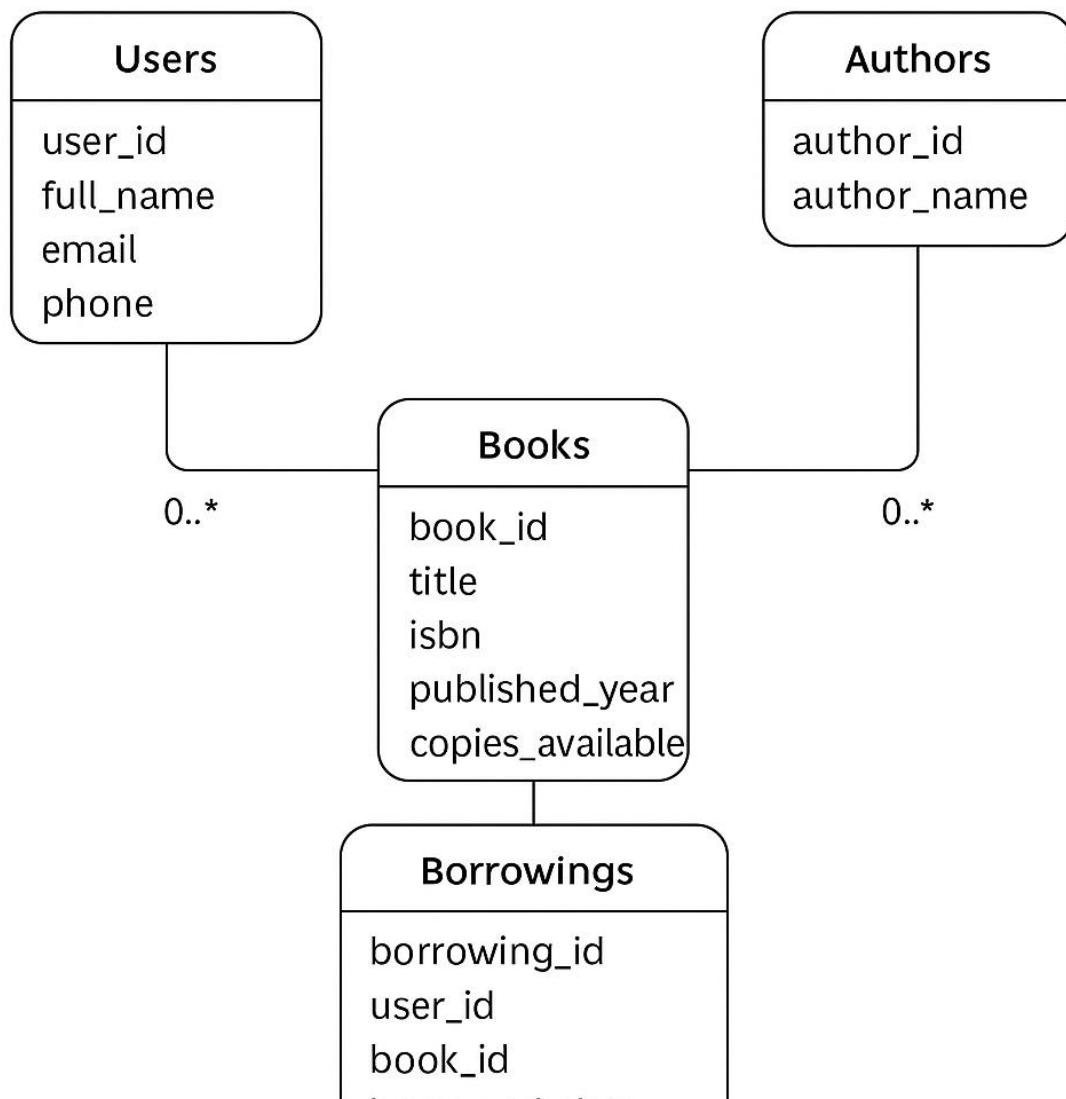


## Entity–Relationship Diagram (ERD)

The ERD below shows the main entities (Users, Books, Authors, Borrowings) and their relationships.

## Database Schema

The database is implemented in MySQL as library\_db. Main Entities: 1. Users – Stores library user details such as name, email, and phone number. 2. Authors – Stores author information. 3. Books – Stores details of all books in the library including ISBN, year, and available copies. 4. BookAuthors – A junction table to represent the many-to-many relationship between books and authors. 5. Borrowings – Records when users borrow and return books. Constraints: - Primary Keys: Each table has a unique identifier (user\_id, book\_id, author\_id, borrowing\_id). - Foreign Keys: Ensure referential integrity between related tables (e.g., Borrowings.user\_id references Users.user\_id). Unique: Emails and ISBNs must be unique. - Not Null: Essential fields cannot be empty. - Check Constraint: Ensures number of available book copies is not negative.



## Relationships

- One-to-Many: A user can borrow many books, but each borrowing record belongs to one user. Many-to-Many: A book can have multiple authors, and an author can write multiple books (resolved via BookAuthors). - One-to-One: Each borrowing record links one user to one book.

## Normalization

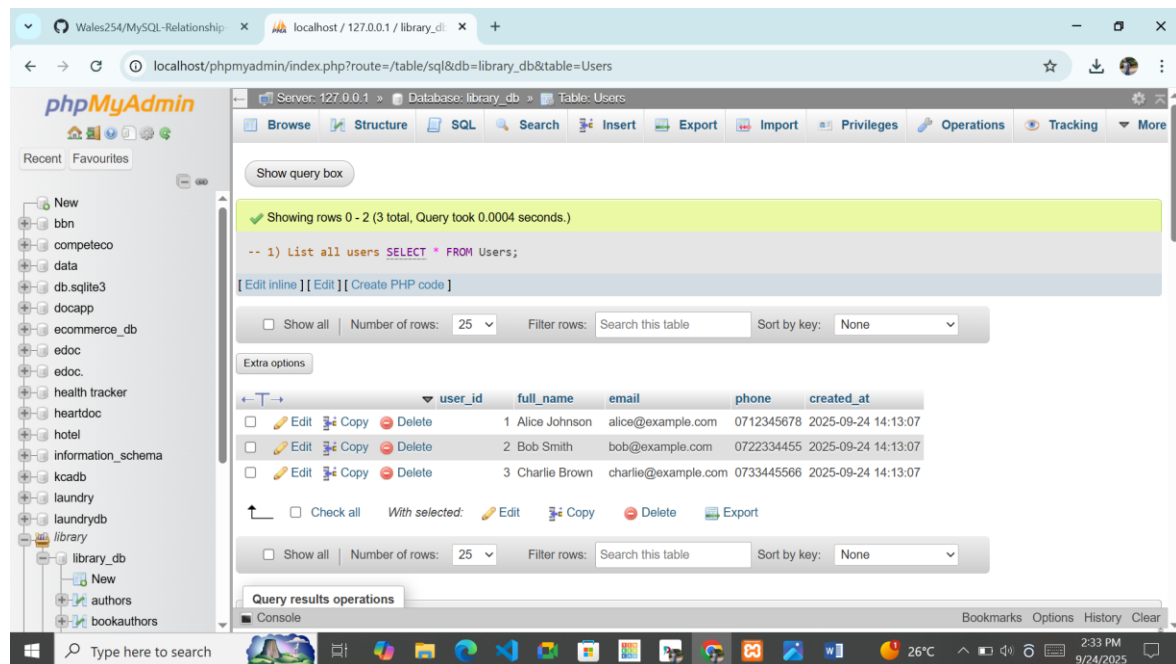
The schema is normalized up to Third Normal Form (3NF): - No repeating groups (1NF). - All non-key attributes depend on the whole primary key (2NF). - No transitive dependencies (3NF). This ensures data integrity and avoids redundancy (e.g., author names stored only once).

## Sample SQL Queries

a) List all books with their authors: `SELECT b.title, a.author_name FROM Books b JOIN`

BookAuthors ba ON b.book\_id = ba.book\_id JOIN Authors a ON ba.author\_id = a.author\_id; b) Show which user borrowed which book: SELECT u.full\_name, b.title, br.borrowed\_date, br.return\_date FROM Borrowings br JOIN Users u ON br.user\_id = u.user\_id JOIN Books b ON br.book\_id = b.book\_id; c) Count how many books each user has borrowed: SELECT u.full\_name, COUNT(\*) AS total\_borrowed FROM Borrowings br JOIN Users u ON br.user\_id = u.user\_id GROUP BY u.full\_name;

Sample query result.



The screenshot shows the phpMyAdmin web interface. The left sidebar displays a list of databases, with 'library\_db' selected. The main panel shows the 'Users' table structure and data. The table has columns: user\_id, full\_name, email, phone, and created\_at. The data is as follows:

user_id	full_name	email	phone	created_at
1	Alice Johnson	alice@example.com	0712345678	2025-09-24 14:13:07
2	Bob Smith	bob@example.com	0722334455	2025-09-24 14:13:07
3	Charlie Brown	charlie@example.com	0733445566	2025-09-24 14:13:07

## Conclusion

This Library Management System database provides a solid foundation for managing users, books, authors, and borrowings. With proper constraints and relationships, it ensures data integrity and supports efficient library operations. Future improvements could include features like tracking overdue books, fines, or integrating with a web-based library portal.