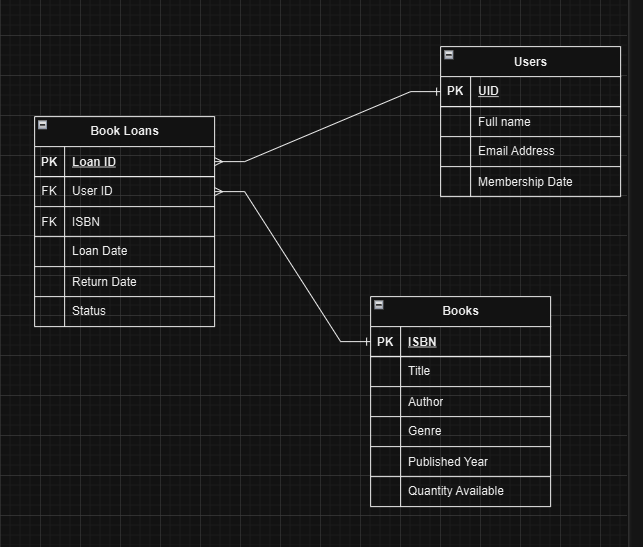
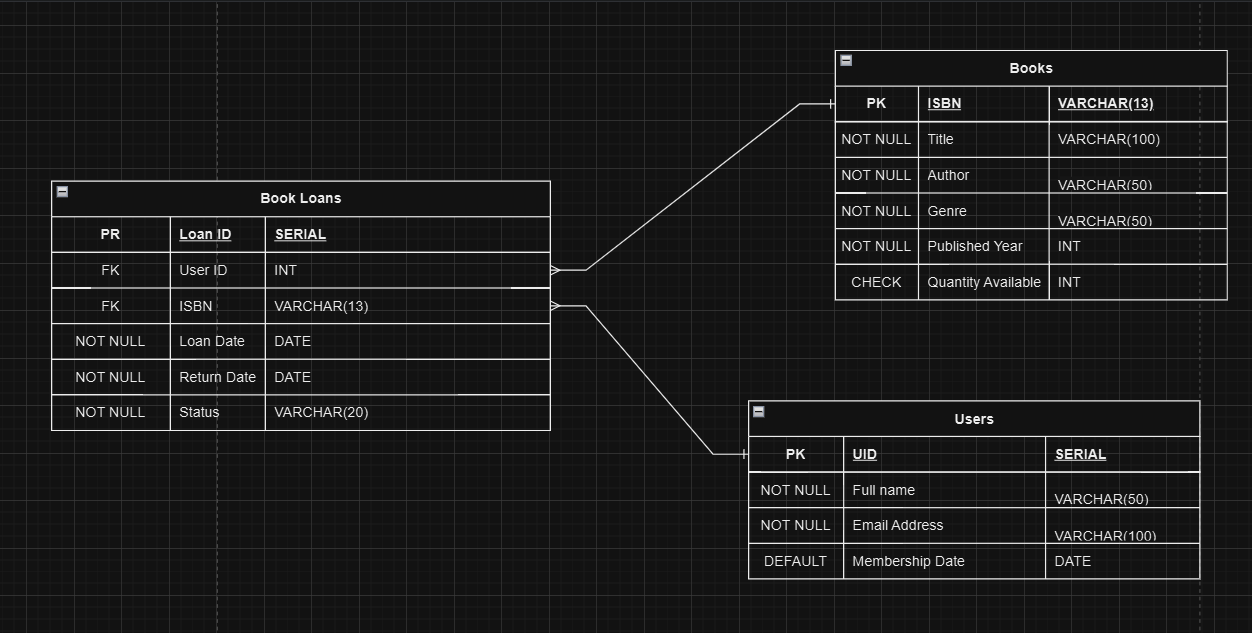
## Part 1: Conceptual Design - 25pts



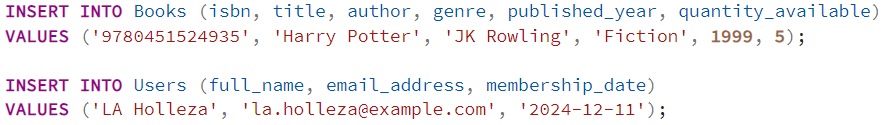
## Part 2: Logical Design - 25pts



## Part 3: SQL Queries

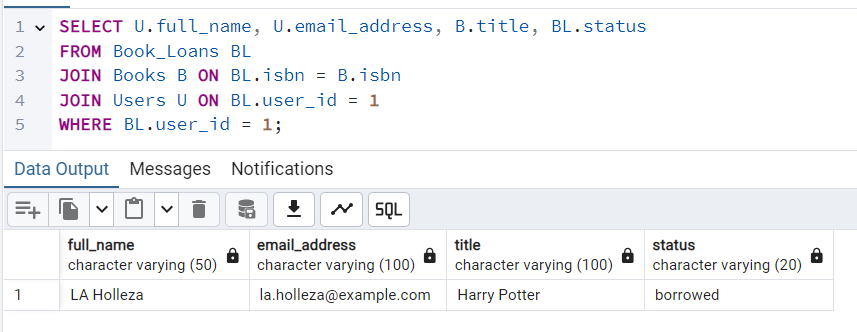
a. Insert a new book into the library with a quantity of 5.

b. Add a new user to the system.

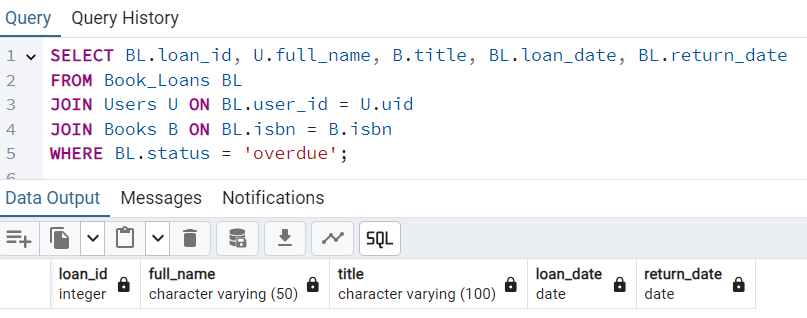


c. Record a book loan for a user.

d. Find all books borrowed by a specific user.



e. List all overdue loans.



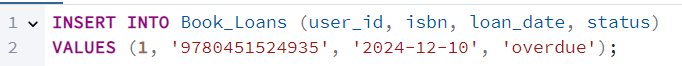
## Part 4: Data Integrity and Optimization

4. Explain how you would ensure:

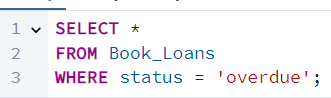
○ The prevention of borrowing books when no copies are available. (15 pts)

I would ensure that the database prevents borrowing of unavailable books by implementing a function that checks if there are copies of books in stock before populating the table.

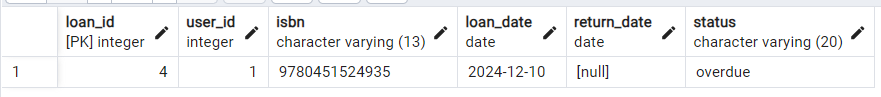
○ Fast retrieval of overdue loans. (20 pts - with CODE and actual screenshot of performance)

In order to show this in action, first I will populate the database with mock data.

Then, what I would do to retrieve overdue loans is to write a query that shows me all the book loan rows that contain the status “overdue”



This is the result:



## Part 5: Reflection (25 pts)

5. What challenges might arise when scaling this database to handle millions of users and books? Suggest one solution for each challenge.

1. Slower Speed - As the database grow, the queries will take longer to respond. If that were to happen then I would use indexing on frequently queried data like the “status” and “user\_id” on book\_loans.

2. Storage - Storage would become a problem when the data is scaled to millions. If so, I would archive data that are not used for a year since most likely individuals never touch the book again once they are finished, especially outdated ones

3. Maintenance – Maintenance would be time-consuming if I were to singlehandedly manage a huge database. If so, I would utilize monitoring tools to provide support on finding problems in the database.