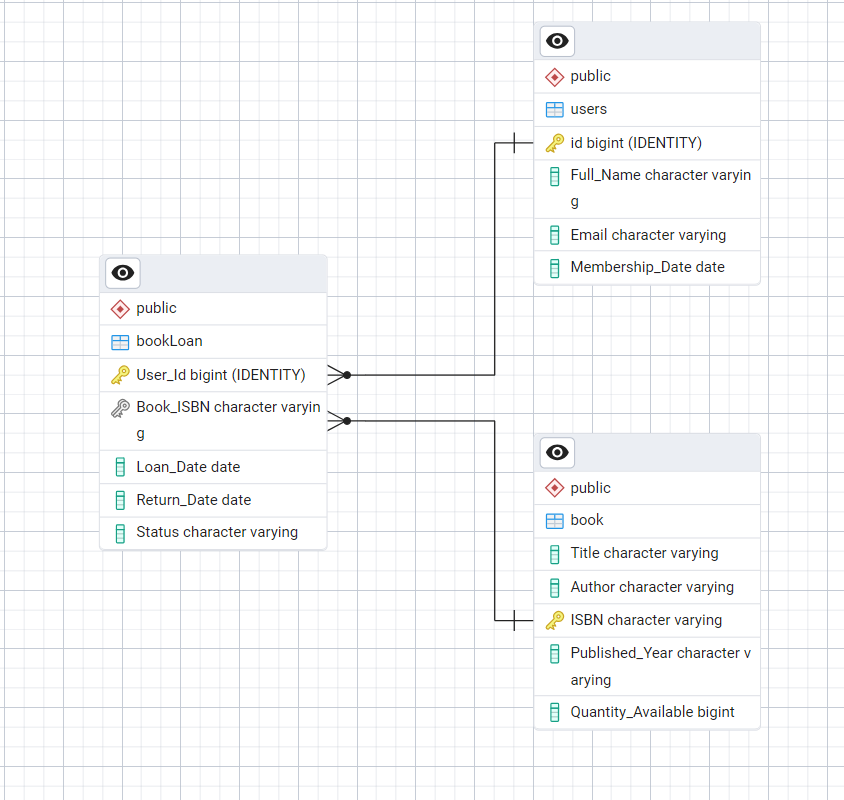
**LABORAT0RY 4 - DATABASE**

DECEMBER 11, 2024

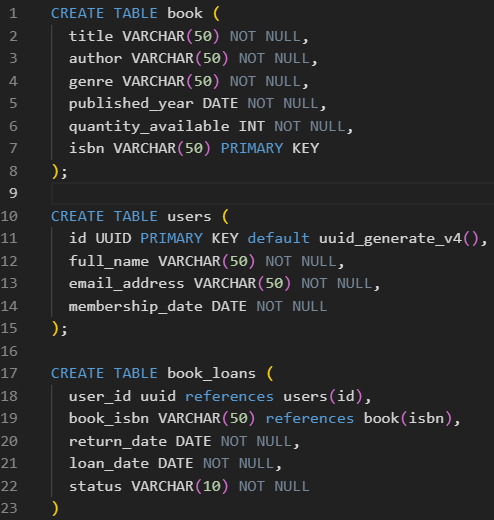
Submitted by:

PAUL ANDREI M. ARIDENTE BSSE-2

**Part 1: Conceptual Design**

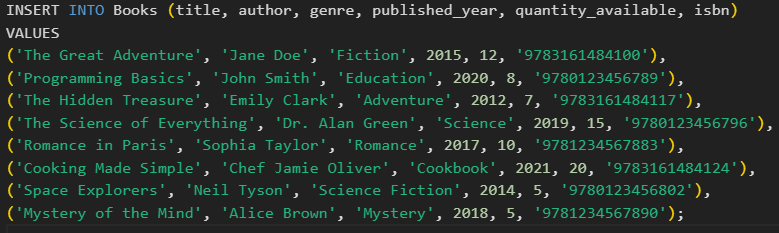


**Part 2: Logical Design**

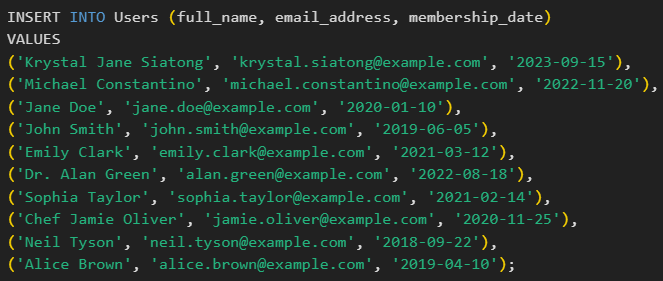


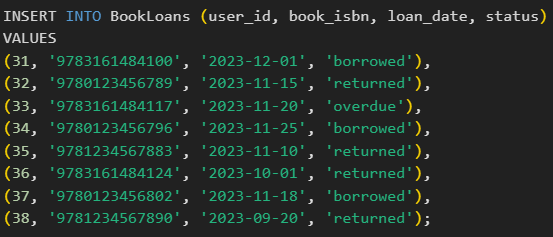
**Part 3: SQL Queries**

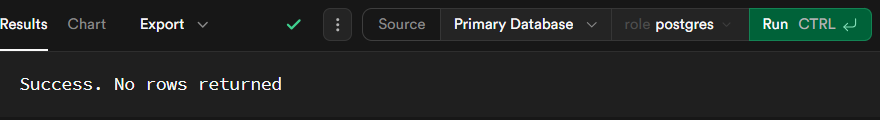
**BOOKS**

****

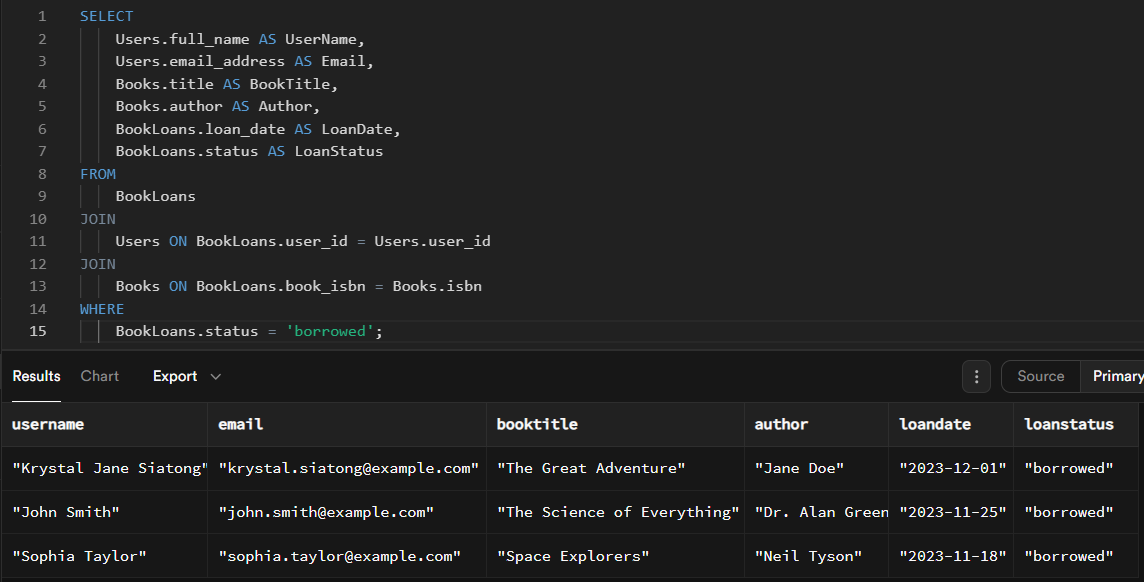
**USERS**

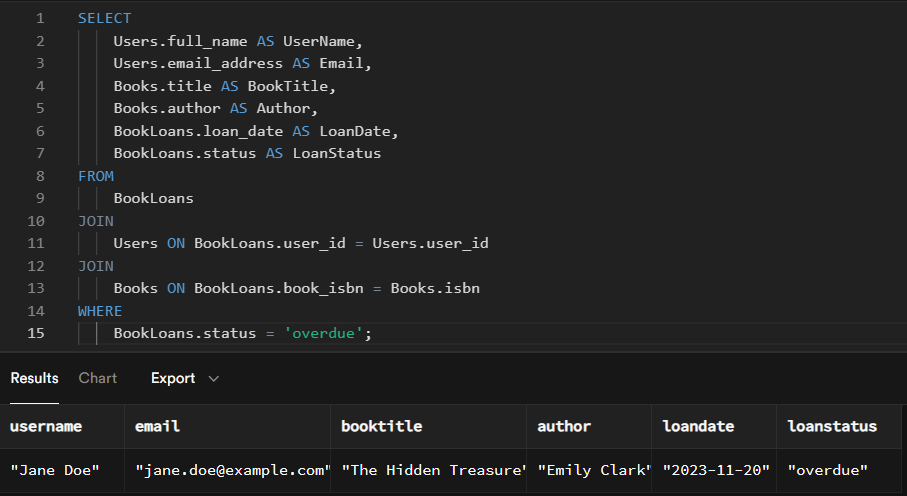
****

** BOOK LOANS**



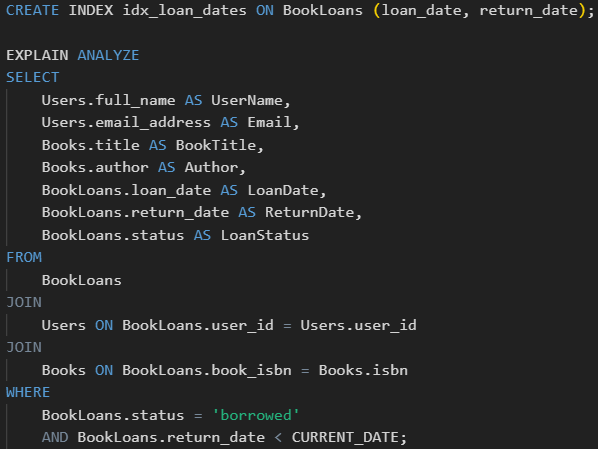
**Find all books borrowed by a specific user**

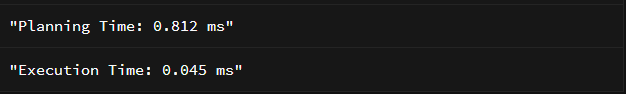
****

**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.**
* *To ensure that users cannot borrow books when no copies are available, I would implement a trigger in the database. The trigger would check the quantity\_available of the book before a new loan is inserted into the BookLoans table. If the quantity is 0 or less, an exception would be raised, and the loan would not be processed. This way, users cannot borrow a book if there are no available copies.*
* **Fast retrieval of overdue loans.**
* *To quickly retrieve overdue loans, I would index the loan\_date and return\_date columns in the BookLoans table. This would optimize the performance of queries that filter on those columns. Additionally, I would write a query to retrieve overdue loans, where the return\_date is in the past and the status is 'borrowed'.*

****

**Part 5: Reflection**

What challenges might arise when scaling this database to handle millions of users and books?

* One of the many challenges that might arise when scaling this database to handle millions of users and books is the Data corruption, as the database grows, ensuring data consistency between related tables (BookLoans, Books, Users) becomes more complex. A single inconsistency could cause significant issues, such as a book being incorrectly marked as available or a user borrowing the same book multiple times simultaneously. A solution to this would be to use foreign key constraints and check constraints to enforce data integrity and implementing atomic transactions this checks if updates to multiple tables are fully committed or fully rolled back.
* Another would be the storing the Database, storing millions of records, especially with rich information like loan histories and large books catalogs, will consume a significant amount of storage. As the database grows, managing the disk space becomes more challenging. A solution to this would be to implement data archiving strategies to offload old or inactive records to cheaper, slower storage or separate data stories.