Reports:

Objectives

The objectives of the system are to:

1. Design and maintain a library database
2. Include library branches, books, users, and book lending records
3. Allow users to place holds on unavailable books and manage the hold queue

Part 1: Designing the Database

1. **Mission Statement**

Mission Statements: to create a database for E-Library System

---

- The database should manage multiple libraries
- Each library can host its book collections and quantities
- Each book should have information about title, author, and categories
- Its user can borrow books, at most 2 at a time
- If the book is currently unavailable, then the user can hold the book (request for when available)
- If the book became available, the user in front of the queue will be given chance in the duration of a week to borrow the book
- If the user doesn't borrow the book, the chance is given to the next in queue
- User can hold at most 2 books

---

2. **Creating Table Structures**
   By the statement above, we need table to hold information for library, book, user, and book circulations.

We will create tables

a. Libraries
b. Books
c. Users

Then, to supplement book information, we create

d. Authors
e. Categories

To store library book collection, we need a junction table between library and books, we call this table as

f. LibraryBooks

Then to store book circulations and to comply with business rules, we add

g. Loans
h. Holds

A book's category, author, and a user's favorite genres (categories) can have multiple values, creating a many-to-many relationship. For simplicity, this report does not create a junction table (cross-reference) for these relationships. Instead, we handle them as one-to-many relationships.

---

3. Determine Table Relationships

Shown in the ERD's

---

4. Determine Business Rules

Some of the business rules will be implemented in the backends.
Notable rules that implemented in the database are :

a. Library name, book title, and user name should be not null.
b. Quantity of book collections should be nonnegative
c. Gender will be stored in boolean. 0 for female, and 1 for male.
d. Due date for borrowing a book is calculated automatically, 2 weeks after the loan date

---

5. Implementing The Design

The result of this Database Design is an Entity Relationship Diagram (ERD). We create the ERD as shown in Figure 1 Entity Relationship Diagram. There are 8 tables and its relations.

We create a SQL script (a DDL) to create the necessary table in the database. The script is stored in https://github.com/Arasy/E-Library-System/blob/main/scripts/table_create.sql

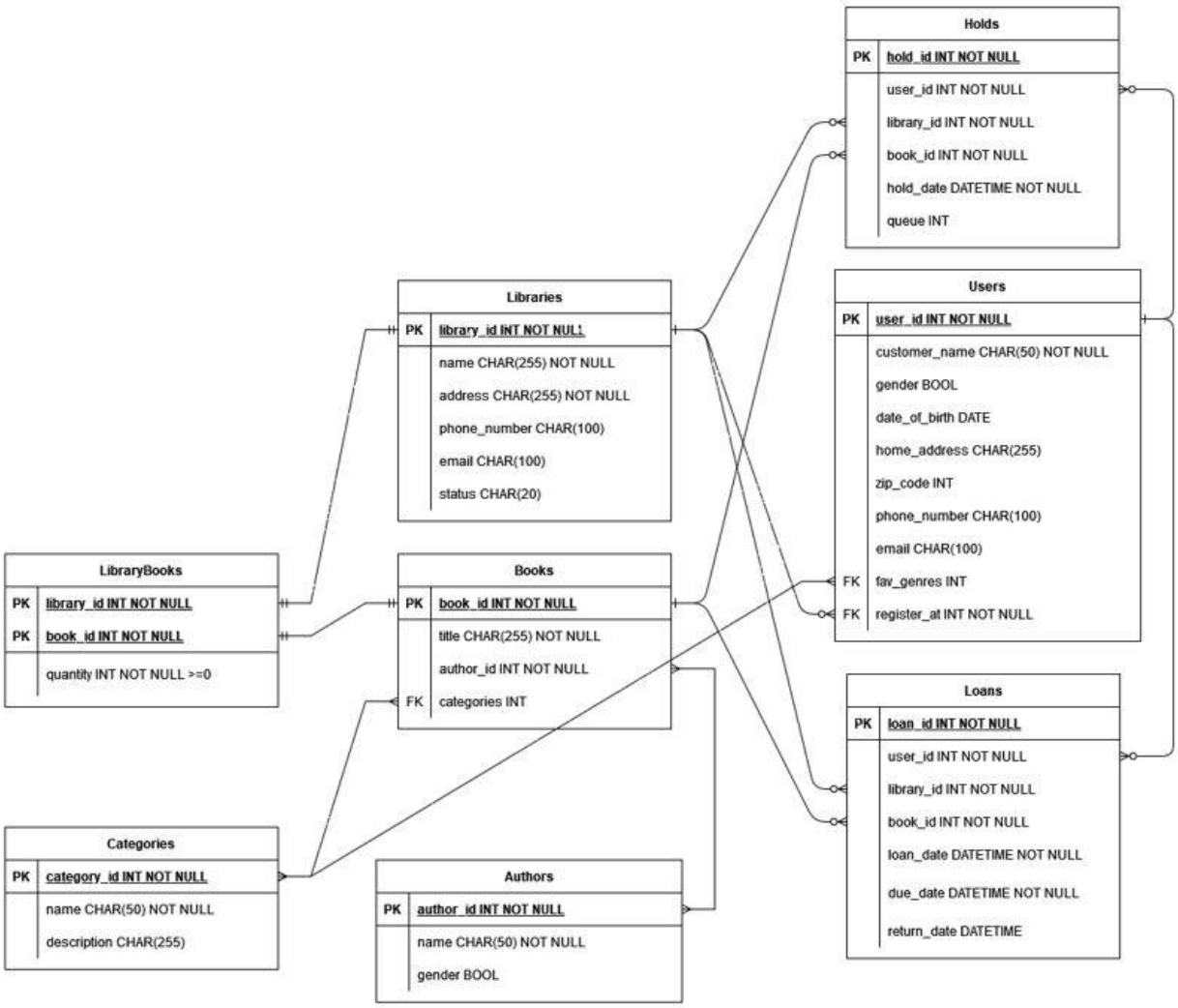

Figure 1 Entity Relationship Diagram

Part 2: Populating Dummy Dataset

After implementing the database design, we create a dummy dataset for each table. The process is documented in the Python notebook file

https://github.com/Arasy/E-Library-System/blob/main/scripts/data_dummy.ipynb .

1. **Create Dummy Dataset**
   Dummy datasets are created using Python and the Faker library, based on the design shown in the ERD (Entity-Relationship Diagram). The created datasets can be exported to JSON or CSV (optional). We include both formats in this report, and they are stored at https://github.com/Arasy/E-Library-System/tree/main/dummy_data

2. **Input Dummy Dataset into the Database:**
   The datasets were inserted into the database using DBeaver as the interface. Some issues related to constraints on the due date column of the loans table occurred but have since been resolved. Notes on how we resolve the problem are recorded in the Python notebook file.

Part 3: Questions

As we already have a library system and its (dummy) data in place, we are intrigued to answer several questions:

1. Which book is the most popular in circulation?
2. Which library lends books the most frequently?
3. Who are the top 5 borrowers?
4. Which genre has the highest circulation rate?
5. What is the percentage of borrowed books returned on time?

Those question will show us the big picture of our library. It can show us measures of effective library policies, book circulations, and user engagements. With proper follow up, it can contribute to effective resource management and service provision.

The query used to answer those questions can be accessed from https://github.com/Arasy/E-Library-System/blob/main/scripts/questions.sql . Below is a snippet of the code and the results:

1.

```
-- 1.   Which book is the most popular in circulation?
select l.book_id, b.title, count(*) as circulation_count
from loans l join books b on l.book_id = b.book_id
group by l.book_id, b.title
order by circulation_count desc
limit 5;
```

loans(+) 1

select l.book_id, b.title, count(*) as circulation_c | Enter a SQL expression to filter results (use Ctrl+Space)

	book_id	title	circulation_count
1	181	Distributed context-sensitive secured line	7
2	104	Function-based explicit Graphical User Interface	7
3	106	Sharable intangible initiative	6
4	32	Self-enabling 5thgeneration parallelism	6
5	62	Secured modular process improvement	5

The most popular book is book id 181, titled “Distributed context-sensitive secured line”, having been borrowed a total of 7 times.

2.

```
--2.    Which library lends books the most frequently?
select l.library_id, lib.name, count(*) as lending_count
from loans l join libraries lib on l.library_id = lib.library_id
group by l.library_id, lib.name
order by lending_count desc
limit 5;
```

loans(+) 1

select l.library_id, lib.name, count(*) as lending_ | Enter a SQL expression to filter results (use Ctrl+

	library_id	name	lending_count
1	0	Sheppard-Wright	107
2	4	Hicks, Stewart and Anderson	100
3	2	Williams, Jackson and Lowe	99
4	1	Alvarado-Willis	99
5	3	Wilson, Golden and Molina	95

Sheppard-Wright Library is the most frequently lending books. With lending count 107 times.

3.

```
--3.    Who are the top 5 borrowers?
select l.user_id, u.customer_name, count(*) as borrowed_count
from loans l join users u on l.user_id = u.user_id
group by l.user_id, u.customer_name
order by borrowed_count desc
limit 5;
```

loans(+) 1

select l.user_id, u.customer_name, count(*) as b | Enter a SQL expression to filter results (use Ctrl+S

	user_id	customer_name	borrowed_count
1	13	Daniel Martin	29
2	6	Daniel Jenkins	26
3	25	Lisa Fernandez	25
4	15	Regina Edwards	23
5	9	Jill Gardner	22

Daniel Martin (user_id 13), Daniel Jenkins (user_id 6), Lisa Fernandez (user_id 25), Regina Edwards (user_id 15), and Jill Gardner (user_id 9) are the top borrowers.

4.

```sql
--4.    Which genre has the highest circulation rate?
select c.category_id, c.name, count(*) as circulation_count
from books b
join loans l on b.book_id = l.book_id
join categories c on b.categories = c.category_id
group by c.category_id
order by circulation_count desc
limit 1;
```

categories 1

```
select c.category_id, c.name, count(*) as circula
```

Enter a SQL expression to filter results (use Ctrl+Sp

	category_id	name	circulation_count
1	5	Mystery-Horror	46

Mystery-Horror is the most popular genre (category). Having been borrowed 46 times.

5.

```sql
--5.    What is the percentage of borrowed books returned on time?
select
    count(case when return_date < due_date then 1 end) as on_time_returns,
    count(*) as total_borrowed,
    (count(case when return_date < due_date then 1 end) * 100.0) / count(*) as percentage
from loans;
```

Results 1

```
select count(case when return_date < due_date
```

Enter a SQL expression to filter results (use Ctrl+Space)

	on_time_returns	total_borrowed	percentage
1	452	500	90.4

The percentage of book returned on time (return date < due date) is 90.4%. The formula is using exclusive symbol because the book is automatically returned after 2 weeks (equal condition), so it considered as not on time returns.

Future Works

1. Connecting python script for generating dummy data directly into the database using sqlalchemy
2. Implementing cross-reference tables for many-to-many relationships that occurred
3. Creating view tables for simplified data analysis

4. Creating backend and user interface, using Django or similar framework

References

https://www.drawio.com/doc/faq/add-rows

https://stackoverflow.com/questions/18992653/entity-relationship-diagram-how-does-the-is-a-relationship-translate-into-table

https://www.postgresql.org/docs/current/sql-createtable.html

https://stackoverflow.com/questions/21503658/prevent-less-than-zero-values-in-postgresql

https://stackoverflow.com/questions/9789736/how-to-implement-a-many-to-many-relationship-in-postgresql

https://faker.readthedocs.io/en/master/

https://www.geeksforgeeks.org/python-faker-library/

https://docs.sqlalchemy.org/en/20/orm/quickstart.html

https://stackoverflow.com/questions/3996904/generate-random-integers-between-0-and-9

https://stackoverflow.com/questions/55198378/faker-python-set-custom-word-list-for-faker-word

https://stackoverflow.com/questions/11875770/how-can-i-overcome-datetime-datetime-not-json-serializable

https://stackoverflow.com/questions/553303/generate-a-random-date-between-two-other-dates

https://dbeaver.com/docs/dbeaver/Data-transfer/#importing-data-from-csv-file

https://www.geeksforgeeks.org/connecting-postgresql-with-sqlalchemy-in-python/

https://stackoverflow.com/questions/24929735/how-to-calculate-date-difference-in-postgresql