Mentoring Week 6 - SQL Query

SQL and Relational Database - Job Preparation Program - Pacmann AI

Author: Aditya Haryanto

Kelas: AI & ML Engineering

Task Description

Your assignment is to design a database system for an e-library application. The application oversees multiple libraries, each hosting a diverse collection of books with varying quantities available for borrowing. Users can borrow or place holds on books (when the book is not immediately available for borrowing).
Below are the key points and requirements for the e-library database system:

- Manages multiple libraries
  The application manages multiple libraries, each housing a diverse collection of books with varying quantities available for borrowing.
- Book Collection
  - The database needs to store information about the diverse collection of books, including titles, authors, and available quantities.
  - To make searching easier for users, books are also divided into categories such as: self-improvement, biography, Fantasy, Romance, Science Fiction, etc.
- User Registration
  Users can register on the e-library platform. Registered users can interact with the platform by borrowing books, placing holds, and managing their account.
- Loan and Hold System
  - Users can borrow books from any library in this application if the book is available.
  - The loan period is 2 weeks. Users can return books earlier than the due date
  - Books will be automatically returned when they exceed the due date
  - Users can only borrow 2 books at a time
  - The platform keeps track of loan transactions, including loan dates, due dates, and return dates.
  - Users can place holds on books that are currently unavailable.
  - The library maintains a hold queue, and when a book becomes available, it can be borrowed by the customer at the front of the queue. Additionally, if a customer doesn't borrow a held book within one week, the book is released for other users to borrow.
  - Users can only hold 2 books at the same time

Your task involves designing the database schema, ensuring that the relationships between tables are well-defined, and foreign keys are used appropriately. Include any additional attributes or tables necessary to support the described functionalities.

Your design should reflect a comprehensive understanding of the e-library's requirements and provide an effective solution to manage books, holds, and loans within a multi-library environment.

The Tools

- **PostgreSQL** is our primary tool during the class and mentoring session.
- You can use **draw.io** to illustrate the database

Your Tasks

Part 1: Designing The Database

Ikuti tahap-tahap perancangan database yang telah dipelajari

1. **Mission Statement**

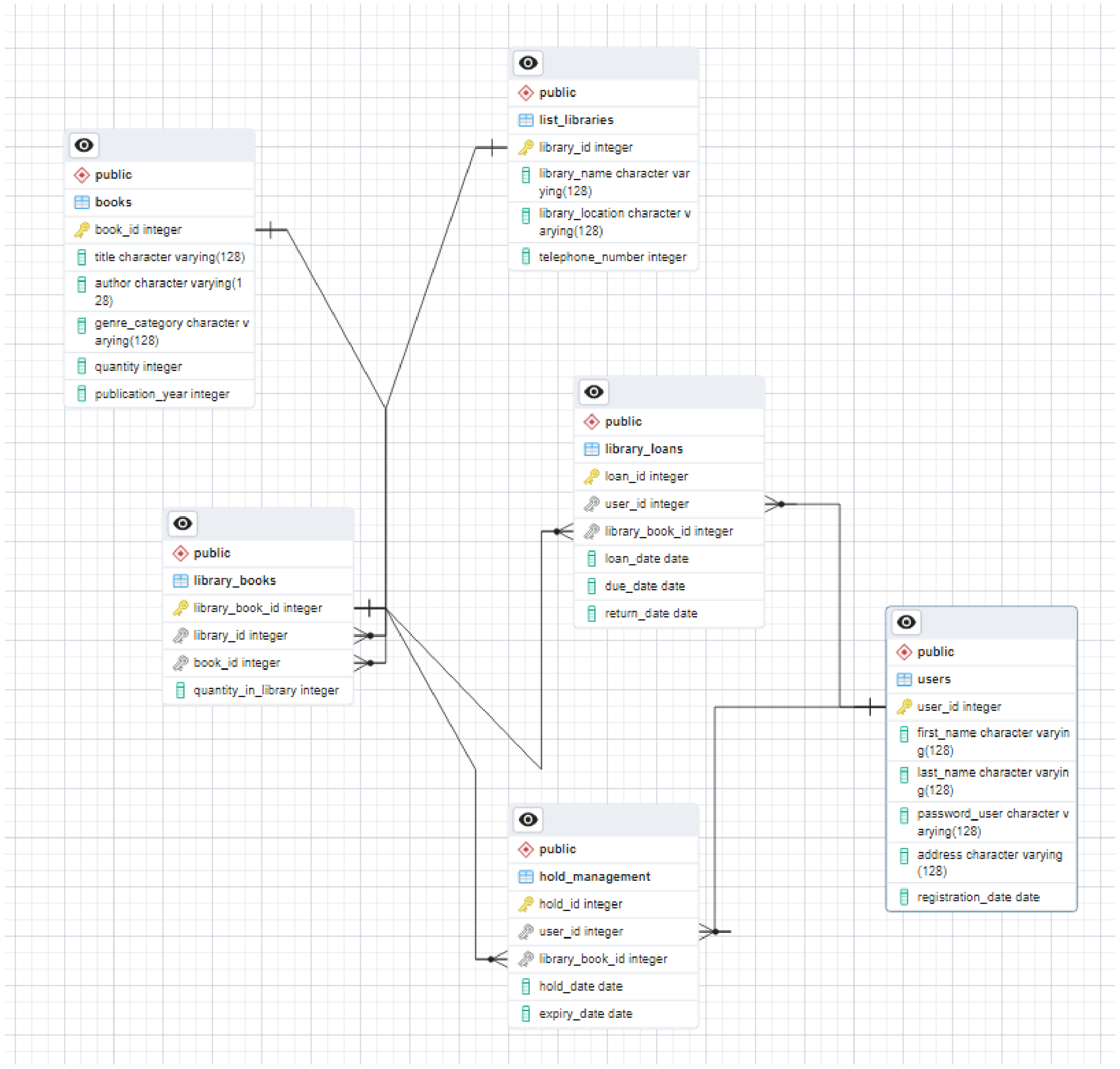
   "E-Library Database for overseeing day-to-day activities of several local libraries in the greater Jakarta area and manage books and users loan- time"

Tables Name	**Description**
list_libraries	This tables stores detailed information for all participating libraries including library unique ID and name respectively (*library_id* & *library_name*) with further information to location and their contact number (*library_location* & *telephone_number*)
books	This table stores general information regarding all books available across all library branches. Features included in this tables are *book_id*, *title*, *genre_category*, *quantity*, *publication_year*
library_books	This table stores information regarding all books available in a specific library. Features included in this tables are *library_book_id*, *library_id*, *book_id*, *quantity_in_library*
library_loans	This table records details of book loans. Features included in this table are *loan_id*, *user_id*, *library_book_id, loan_date, due_date, return_date*
hold_management	This table manages hold placed by users on unavailable books in the library book pool. Features included in this table are hold_id, user_id, *library_book_id, hold_date, expiry_date.*
users	This table stores user general and personal information. Features included in this table are *user_id, first_name, last_name, password_user, address, registration_date.*

2. Creating Table Structures & Business Rules

list_libraries	books	library_books
library_id (Primary Key, Serial) library_name (Varchar, 128, Not Null) library_location (Varchar, 128, Not Null) telephone_number (Integer, Not Null)	book_id (Primary Key, Serial) title (Varchar, 128, Not Null) author (Varchar, 128, Not Null) genre_category (Varchar, 128, Not Null) quantity (Integer, Not Null) publication_year (Integer, Not Null)	library_book_id (Primary Key, Serial) library_id (Integer, Not Null, Foreign Key - References list_libraries) book_id (Integer, Not Null, Foreign Key - References books) quantity_in_library (Integer, Not Null)
library_loans	**hold_management**	**users**
loan_id (Primary Key, Serial) user_id (Integer, Not Null, Foreign Key - References users) library_book_id (Integer, Not Null, Foreign Key - References library_books) loan_date (Date, Not Null) due_date (Date, Not Null) return_date (Date)	hold_id (Primary Key, Serial) user_id (Integer, Not Null, Foreign Key - References users) library_book_id (Integer, Not Null, Foreign Key - References library_books) hold_date (Date, Not Null) expiry_date (Date, Not Null)	user_id (Primary Key, Serial) first_name (Varchar, 128, Not Null) last_name (Varchar, 128, Not Null) password_user (Varchar, 128, Not Null) address (Varchar, 128, Not Null) registration_date (Date, Not Null)

3. Determine Table Relationships



4. Implementing The Design

```
-- Dropping existing tables if they exist
DROP TABLE IF EXISTS books;
DROP TABLE IF EXISTS list_libraries;
DROP TABLE IF EXISTS library_books;
DROP TABLE IF EXISTS users;
DROP TABLE IF EXISTS library_loans;
DROP TABLE IF EXISTS hold_management;
```

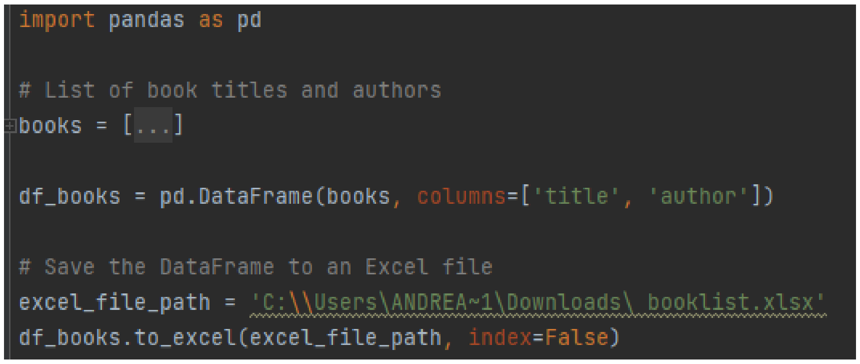
```
-- Creating books table
CREATE TABLE books (
   book_id SERIAL PRIMARY KEY,
   title VARCHAR(128) NOT NULL,
   author VARCHAR(128) NOT NULL,
   genre_category VARCHAR(128) NOT NULL,
   quantity INTEGER NOT NULL,
   publication_year INTEGER NOT NULL
);

-- Creating list_libraries table
CREATE TABLE list_libraries (
   library_id SERIAL PRIMARY KEY,
   library_name VARCHAR(128) NOT NULL,
   library_location VARCHAR(128) NOT NULL,
   telephone_number INTEGER NOT NULL
);

-- Creating library_books table
CREATE TABLE library_books (
   library_book_id SERIAL PRIMARY KEY,
   library_id INTEGER NOT NULL REFERENCES list_libraries,
   book_id INTEGER NOT NULL REFERENCES books,
   quantity_in_library INTEGER NOT NULL
);

-- Creating users table
CREATE TABLE users (
   user_id SERIAL PRIMARY KEY,
   first_name VARCHAR(128) NOT NULL,
   last_name VARCHAR(128) NOT NULL,
   password_user VARCHAR(128) NOT NULL,
   address VARCHAR(128) NOT NULL,
   registration_date DATE NOT NULL
);

-- Creating library_loans table
CREATE TABLE library_loans (
   loan_id SERIAL PRIMARY KEY,
   user_id INTEGER NOT NULL REFERENCES users,
   library_book_id INTEGER NOT NULL REFERENCES library_books,
   loan_date DATE NOT NULL,
   due_date DATE NOT NULL,
   return_date DATE NOT NULL
);

-- Creating hold_management table
CREATE TABLE hold_management (
   hold_id SERIAL PRIMARY KEY,
   user_id INTEGER NOT NULL REFERENCES users,
   library_book_id INTEGER NOT NULL REFERENCES library_books,
   hold_date DATE NOT NULL,
   expiry_date DATE NOT NULL
);
```

Part 2: Populating Dummy Dataset

2.1. Dataset for Table “Books”

Datasets for this project are generated through the use of ChatGPT which is tasked to generate multiple features of fictional books such as: book titles, authors, publication date and so on. The goal was to populate with at least 90 books in total for all libraries. For the first step, the following prompt was used to generate 100 fictional books alongside its Author:

“*hi im creating a dataset for my dummy sql project. This project creates a database for fictional libraries and I want you to create 100 random book names that the library stores. Further, please create an excel file separating the title of the book and the author of the book* ”



```
import pandas as pd

# List of book titles and authors
books = [...]

df_books = pd.DataFrame(books, columns=['title', 'author'])

# Save the DataFrame to an Excel file
excel_file_path = 'C:\\Users\ANDREA~1\Downloads\ booklist.xlsx'
df_books.to_excel(excel_file_path, index=False)
```

ChatGPT then responded by creating a list of books and authors using python script with pandas, however the process to generate all 100 data always failed when data reached around 75 books and the code was never finished. I then continued the result from chat GPT and added manually around 17 rows of random book titles and authors in the dataset. A new excel file is then created with pandas library with a total of 92 books entries (Fig. 1).

Book genres are created manually within the excel file with 6 total categories ranging from educational , Science Fiction, Romance, Self-Help, Biography and Nature. The assignment of book genres to each book is created manually and the amount of books for each category is randomly assigned. The number of book quantities is created using python code to generate 92 random numbers ranging from 2 to 8. Same concept is applied for generating the publication year for all books listed in the dataset, code used for the generation is as follows:

```
import random

# Generating 98 random numbers ranging from 2 to 8
random_numbers = [random.randint(2, 8) for _ in range(98)]
print(random_numbers)
```

```
# Generating 92 random years
random_years = [random.randint(1996, 2022) for _ in range(98)]
print(random_years)
```

All of the results are then copied and formatted in the master excel file for the table “Books” and saved as CSV data. The complete CSV-file is checked again using text editor for unreadable characters and missing data. Finally, the CSV file are imported to the database using import data function in pgAdmin4

2.2 Dataset Generation for Table “list_libraries”

This dataset is generated manually using excel as the main program since content is considered small. Since there are only 5 libraries in the database, each library is assigned an ID from 1 to 5 and their number represents their location such as Jakarta Pusat library has the library_id: 1 ; Jakarta Barat Library has the library_id: 2 and so on. Their telephone numbers are created randomly and manually in the excel file. The complete CSV-file is checked again using text editor for unreadable characters and missing data. Finally, the CSV file are imported to the database using the import data function in pgAdmin4 .

2.3 Dataset Generation for Table “users”

This dataset is generated manually using excel as well as automatically. Features of the table are generated randomly through pandas and random library in Phyton. List of first and last name was generated from ChatGPT and copied to list of first / last name in python, resulting in the following code:

```
import pandas as pd
import random

# Sample first names and last names
first_names = ["James", "Mary", "John", "Patricia", "Robert",
"Jennifer", "Michael", "Linda", "William", "Elizabeth",
              "David", "Barbara", "Richard", "Susan", "Joseph",
"Jessica", "Thomas", "Sarah", "Charles", "Karen",
              "Christopher", "Nancy", "Daniel", "Lisa", "Matthew",
"Margaret", "Anthony", "Betty", "Mark", "Sandra",
              "Donald", "Ashley", "Steven", "Kimberly", "Paul",
"Emily", "Andrew", "Donna", "Joshua", "Michelle"]

last_names = ["Smith", "Johnson", "Williams", "Brown", "Jones",
"Miller", "Davis", "Garcia", "Rodriguez", "Wilson",
             "Martinez", "Anderson", "Taylor", "Thomas",
"Hernandez", "Moore", "Martin", "Jackson", "Thompson", "White",
             "Lopez", "Lee", "Gonzalez", "Harris", "Clark",
"Lewis", "Robinson", "Walker", "Perez", "Hall",
             "Young", "Allen", "Sanchez", "Wright", "King",
"Scott", "Green", "Baker", "Adams", "Nelson"]

# random list of names
random_names = [{"First Name": random.choice(first_names), "Last
Name": random.choice(last_names)} for _ in range(34)]

#main dataframe
names_df = pd.DataFrame(random_names)

# creating excel file
file_path = 'C:\\Users\ANDREA~1\Downloads\ name_list_random.xlsx'
names_df.to_excel(file_path, index=False)

# Verify result
print(names_df.head())
```

User_id are then created using “2015” as the unique identification and followed by adding numbers 1 to (n) according to the number of users registered. Password are created automatically using the following python code:

```
import string

import pandas as pd

import random as random


def generate_simple_password(length=8):

   """

   Create simple random password containing 8 varieties character

   """

   characters = string.ascii_letters + string.digits

   return ''.join(random.choice(characters) for _ in range(length))

# generate password

passwords = [generate_simple_password() for _ in range(34)]

#Dataframe for password

passwords_df = pd.DataFrame(passwords, columns=["Password"])

#Create excel file
```

```
#passwords_file_path        =        'C:\\Users\ANDREA~1\Downloads\
randomPassword.xlsx'

#passwords_df.to_excel(passwords_file_path, index=False)

print(passwords_df.head())
```

User addresses are created manually with 5 main locations that are listed similarly with library locations (Jakarta Pusat, Barat and so on). Registration dates for users are created manually in excel with a random date in August to simulate a grand opening sequence in late July and throughout the month of August. The complete CSV-file is checked again using text editor for unreadable characters and missing data. Finally, the CSV-file are imported to the database using the import data function in pgAdmin4 .

2.5 Dataset Generation for Table "library_books"

The dataset for this table is generated manually using excel for almost all features excluding the book quantity for each library that was created using python. Library_book_id was created by combining the library_id and book_id resulting in unique code for each book in each library. The quantity in the library feature is connected to the "books" tables and contains the same number of books listed there. The complete CSV-file is checked again using text editor for unreadable characters and missing data. Finally, the CSV-file are imported to the database using the import data function in pgAdmin4 .

2.5 Dataset Generation for Table "library_loans"

The dataset for this table is generated manually using excel for all the features included. Loan_id is created by combining the user_id and library_book_id. Dates for start of book loan and return date are generated manually with some assumption that not all users return their loaned book on time. This is also the reason for investigating which users violate the return policy later on in the SQL-Query section. Due date is calculated by adding 14 days on the start of the loan date. The complete CSV-file is checked again using text editor for unreadable characters and missing data. Finally, the CSV-file are imported to the database using the import data function in pgAdmin4 .

2.6 Dataset Generation for Table "hold_management"

The dataset for this table is generated manually using excel for all the features included. Hold_id is created by combining the user_id and library_book_id. Dates for start of loan hold and expiry date are generated manually with some assumption that not all users are granted access due to unavailability of book.Expiry date is calculated by adding 7 days on the start of the hold period. The complete CSV-file is checked again using text editor for unreadable characters and missing data. Finally, the CSV-file are imported to the database using the import data function in pgAdmin4 .

Part 3: 5 objectives/questions

After populating the dataset with complete data, 5 basic questions are then formulated to investigate the library user activities from August 2022.

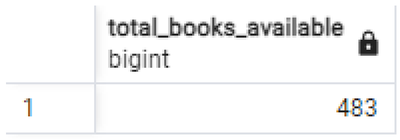
First we want to find out information regarding the book available in the library. Therefore for the first question there are several question needs to be point out:

1.a. How many books are available across all libraries and which library has the most?

First Query:

```
SELECT SUM(quantity_in_library) AS total_books_available
FROM library_books;
```

Result:

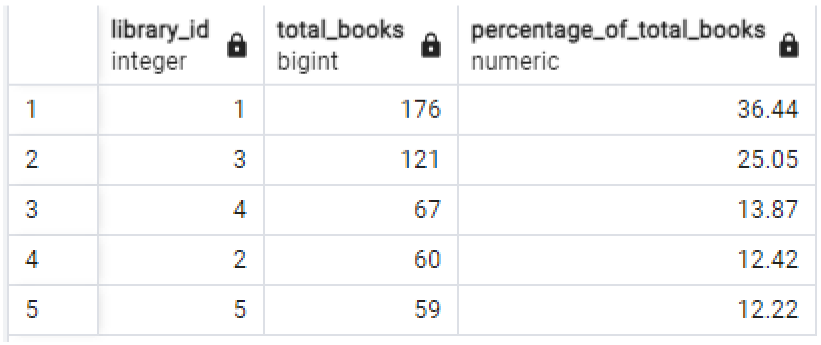


```
   total_books_available
   bigint
1                    483
```

Second Query:

```
WITH TotalBooks AS (
   SELECT SUM(quantity_in_library) AS total_books_available
   FROM public.library_books
)

SELECT
   lb.library_id,
   SUM(lb.quantity_in_library) AS total_books,
   ROUND((SUM(lb.quantity_in_library) * 100.0) / tb.total_books_available, 2) AS
percentage_of_total_books
FROM
   public.library_books lb,
   TotalBooks tb
GROUP BY
   lb.library_id,
   tb.total_books_available
ORDER BY
   total_books DESC;
```

Result:



	library_id integer	total_books bigint	percentage_of_total_books numeric
1	1	176	36.44
2	3	121	25.05
3	4	67	13.87
4	2	60	12.42
5	5	59	12.22

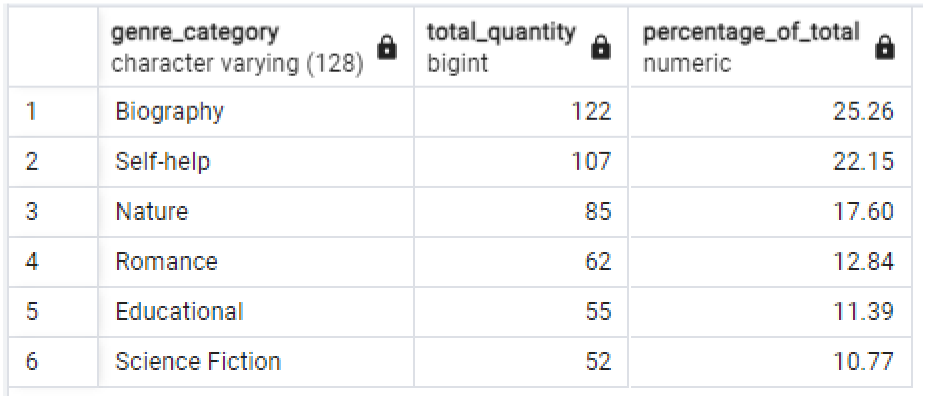
Insights & recommendation:

The availability of books significantly varies across the top three libraries. The Jakarta Pusat branch, which leads the list, holds 176 books, while the Jakarta Selatan Branch possesses less than half of this number. Closely following Jakarta Pusat in terms of book collection is the Jakarta Timur Branch who own approximately a quarter of total book collection. However, disparities in book availability are evident in other branches, such as Jakarta Barat Jakarta Utara and Jakarta Selatan, with the three of these combined having approximately 40% of the total book count and each of these branches only having about 12,8% of book share. The aim moving forward is to standardize book availability across all branches, beginning with adding more books to these three libraries.

1.b. What is the most popular genre across all books?
First Query:

```
WITH TotalBooks AS (
    SELECT SUM(quantity_in_library) AS total_quantity
    FROM library_books
)

SELECT b.genre_category, SUM(lb.quantity_in_library) AS total_quantity,
    ROUND((SUM(lb.quantity_in_library) * 100.0) / tb.total_quantity, 2) AS percentage_of_total
FROM public.books b
JOIN
    public.library_books lb ON b.book_id = lb.book_id,
    TotalBooks tb
GROUP BY
    b.genre_category,
    tb.total_quantity
ORDER BY
    total_quantity DESC;
```

Result:



	genre_category character varying (128)	total_quantity bigint	percentage_of_total numeric
1	Biography	122	25.26
2	Self-help	107	22.15
3	Nature	85	17.60
4	Romance	62	12.84
5	Educational	55	11.39
6	Science Fiction	52	10.77

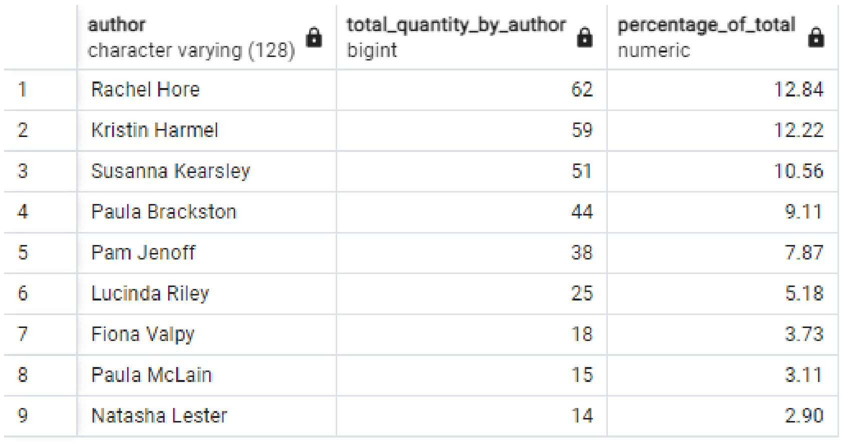
Insights & recommendation:

This question gives a deeper dive into the content of all books across all libraries. The query showed a fairly healthy distribution across genres available to access. However our libraries have only 6 different types of genre which left a lot to be desired. Popular genres such as Mystery/Detective, Thriller, Dystopian and Humor are not included in the current book selection [1]. Therefore future goals should focus on adding book which are categorized as popular genre for all people.

1.c. Top 9 Authors with the most books available across all libraries?

```
WITH TotalBooks AS (
    SELECT SUM(quantity_in_library) AS total_quantity
    FROM public.library_books
)

SELECT
    b.author,
    SUM(lb.quantity_in_library) AS total_quantity_by_author,
    ROUND((SUM(lb.quantity_in_library) * 100.0) / tb.total_quantity, 2) AS percentage_of_total
FROM
    public.books b
JOIN
    public.library_books lb ON b.book_id = lb.book_id,
    TotalBooks tb
GROUP BY
    b.author,
    tb.total_quantity
ORDER BY
    total_quantity_by_author DESC
LIMIT 9;
```

Result:



	author character varying (128)	total_quantity_by_author bigint	percentage_of_total numeric
1	Rachel Hore	62	12.84
2	Kristin Harmel	59	12.22
3	Susanna Kearsley	51	10.56
4	Paula Brackston	44	9.11
5	Pam Jenoff	38	7.87
6	Lucinda Riley	25	5.18
7	Fiona Valpy	18	3.73
8	Paula McLain	15	3.11
9	Natasha Lester	14	2.90

Insights & recommendation:

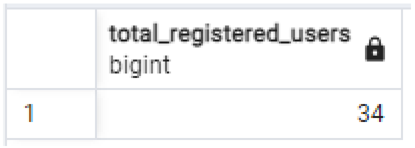
There are 35 different authors in total for the current book selection and the top 9 authors are listed on the result. For a standpoint of small libraries, the current selection of authors is good enough in terms of quantity. While the quantity of authors is satisfactory, a deeper analysis of the quality of their books is necessary. This includes determining whether the selection represents the authors' best-selling works. Therefore, a more thorough investigation into the book selection is recommended to ensure it meets both quantity and quality standards

Since the library's grand opening at the end of July and big promotion of user registration in August, we want to find out about the user information. Therefore, some question for users are:

2.a. How many users have already registered in the library ?

```
SELECT COUNT(*) AS total_registered_users
FROM users;
```

Result:

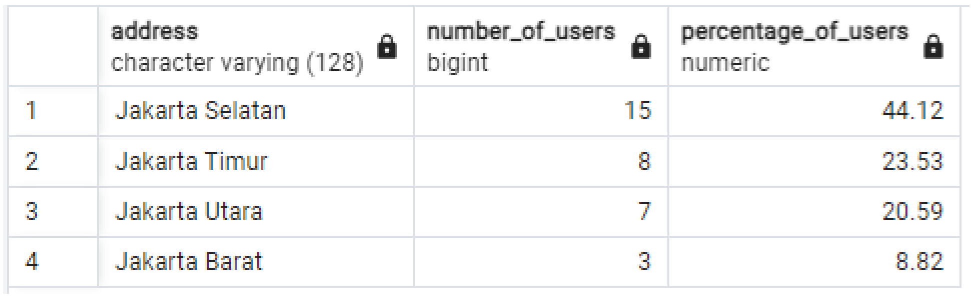


	total_registered_users bigint
1	34

2.b. Most common administrative cities the users are coming from?

```
WITH TotalUsers AS (
   SELECT COUNT(*) AS total_users
   FROM public.users
)

SELECT
    u.address,
    COUNT(*) AS number_of_users,
    ROUND((COUNT(*) * 100.0) / tu.total_users, 2) AS percentage_of_users
FROM
    public.users u,
    TotalUsers tu
WHERE
    u.address IN ('Jakarta Utara', 'Jakarta Barat', 'Jakarta Timur', 'Jakarta Selatan')
GROUP BY
    u.address,
    tu.total_users
ORDER BY
    number_of_users DESC;
```

Result:



	address character varying (128)	number_of_users bigint	percentage_of_users numeric
1	Jakarta Selatan	15	44.12
2	Jakarta Timur	8	23.53
3	Jakarta Utara	7	20.59
4	Jakarta Barat	3	8.82

Insights & recommendation:

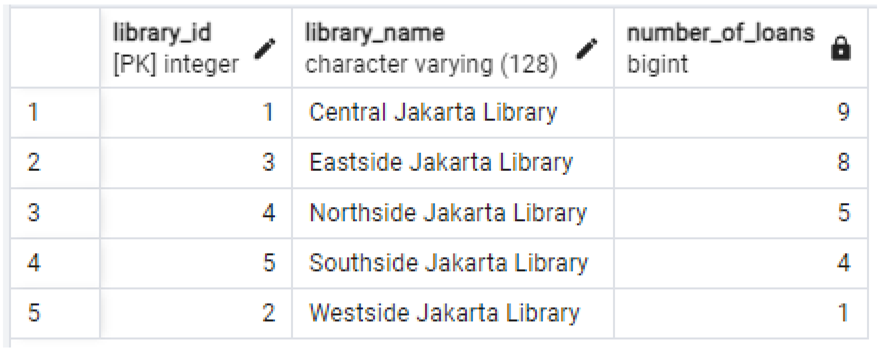
After the grand opening of all libraries in July and August, there are only 34 total user registration. This number is considerably low for the investment committed to create the grand opening. Another problem was found when looking at the place of user registration and number of books available in each library. Jakarta Selatan, which only has approximately 12.22% of all book collections, recorded the highest number of user registration. Further, there is no user registration for Jakarta Pusat branch which has the largest number of books available. This can be problematic if users with restricted means of mobility must borrow a book from another library due to the book being unavailable in their local library. This discovery underlines the importance of equal distribution of book availability across all library branches.

To track activities happening during the opening month, we also want to find out about the book loans. The following questions are then formulated:

3.a. Number of loans for each library?

```
SELECT
   l.library_id,
   l.library_name AS library_name,
   COUNT(ll.loan_id) AS number_of_loans
FROM
   public.library_books lb
JOIN
   public.library_loans ll ON lb.library_book_id = ll.library_book_id
JOIN
   public.list_libraries l ON lb.library_id = l.library_id
GROUP BY
   l.library_id, l.library_name
ORDER BY
   number_of_loans DESC;
```

Result:



	library_id [PK] integer	library_name character varying (128)	number_of_loans bigint
1	1	Central Jakarta Library	9
2	3	Eastside Jakarta Library	8
3	4	Northside Jakarta Library	5
4	5	Southside Jakarta Library	4
5	2	Westside Jakarta Library	1

Insights & recommendation:

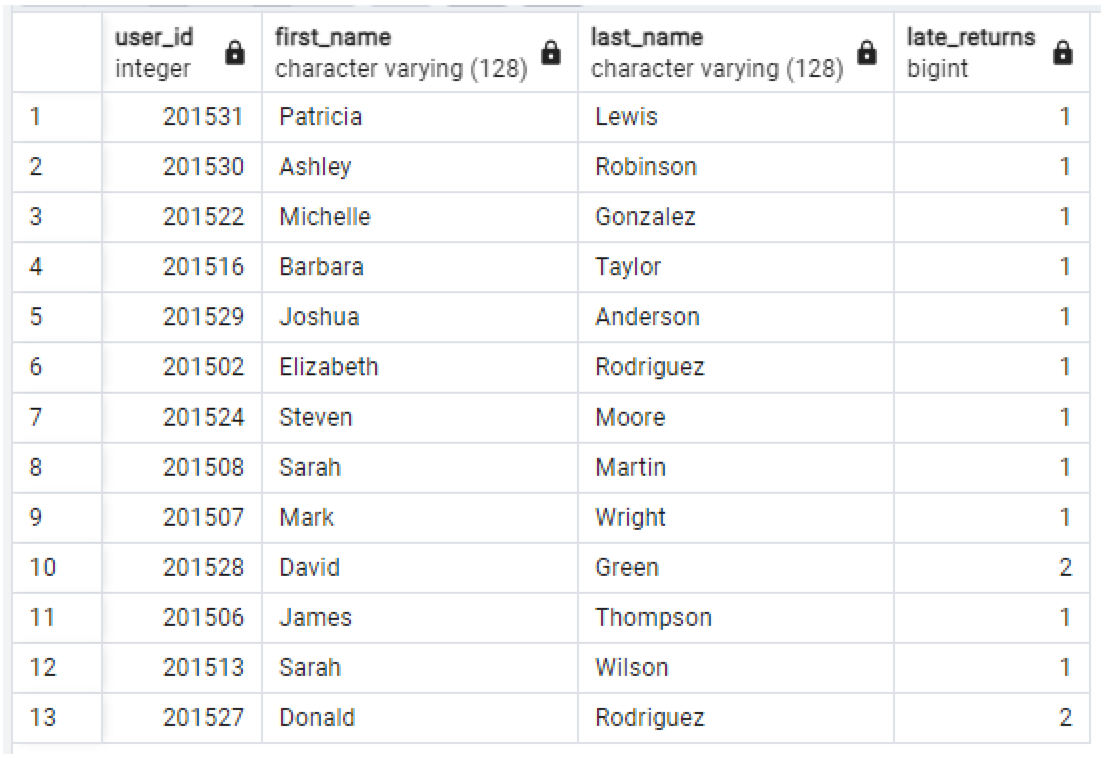
In total, there were 28 registered book loans during the month of August which resulted in 0.79 loan to user ratio (28 loans / 34 users). This indicates a high number of loan activity among users and shows that the current selection meets the demand of the registered users. The Jakarta Pusat Branch recorded the highest number of loans, followed closely by the Jakarta Timur branch. Generally, all library branches have had a book loan. However, due to the small number of loan data, analysis on user behavior for book loans are not yet appropriate since data can significantly fluctuate. Therefore, user analysis should be delayed once the loan data reached at least 100 - 120 registered loans

3.b. Are there any loans that missed the return date? If yes how many times and show the user_id to issue a fine.

```
SELECT
```

```
    ll.user_id,
    u.first_name,
    u.last_name,
    COUNT(ll.loan_id) AS late_returns
FROM
    public.library_loans ll
JOIN
    public.users u ON ll.user_id = u.user_id
WHERE
    ll.return_date > ll.due_date
GROUP BY
    ll.user_id, u.first_name, u.last_name
HAVING
    COUNT(ll.loan_id) > 0;
```

Result:



	user_id integer	first_name character varying (128)	last_name character varying (128)	late_returns bigint
1	201531	Patricia	Lewis	1
2	201530	Ashley	Robinson	1
3	201522	Michelle	Gonzalez	1
4	201516	Barbara	Taylor	1
5	201529	Joshua	Anderson	1
6	201502	Elizabeth	Rodriguez	1
7	201524	Steven	Moore	1
8	201508	Sarah	Martin	1
9	201507	Mark	Wright	1
10	201528	David	Green	2
11	201506	James	Thompson	1
12	201513	Sarah	Wilson	1
13	201527	Donald	Rodriguez	2

Insights & recommendation:

Of our users, 13 have violated their loan agreements, with some repeatedly overextending their borrowing period. Considering that over a third of total users have missed their return dates, it's imperative to implement stricter, enforceable loan policies. Our first strategy involves introducing a reminder system and increasing fines for late returns. These measures aim to significantly reduce the frequency of loan violations among users

References:

[1] Abi Jhonson. (2023). *23 Most Popular Book Genres for 2024 - Bona Fide BookWorm.* Online: https://bonafidebookworm.com/most-popular-book-genres/