E-LIBRARY APPLICATION

Project Objective

Tujuan utama dari proyek ini adalah untuk merancang dan mengimplementasikan sistem manajemen e-library yang efisien, yang mampu menangani berbagai perpustakaan, koleksi buku, dan transaksi peminjaman serta pemesanan. Sistem ini diharapkan dapat menyediakan platform yang memudahkan pengguna dalam mencari, meminjam, dan memesan buku yang tersedia di beberapa perpustakaan. Selain itu, sistem ini juga dirancang untuk memberikan informasi yang akurat tentang ketersediaan buku, sehingga perpustakaan dapat mengelola dan mendistribusikan koleksinya secara lebih efisien berdasarkan kebutuhan dan permintaan pengguna.

Melalui penerapan query data dan laporan, sistem ini bertujuan untuk memberikan wawasan yang lebih dalam mengenai kebiasaan pengguna dalam meminjam buku, kategori buku yang paling diminati, dan efisiensi pengelolaan transaksi perpustakaan. Pengelola perpustakaan dapat menggunakan informasi ini untuk memperbaiki layanan mereka dan memastikan bahwa pengguna mendapatkan pengalaman terbaik. Selain itu, proyek ini juga berfokus pada penegakan aturan peminjaman dan pemesanan secara otomatis, seperti batas waktu peminjaman dan pembatasan jumlah buku yang dapat dipinjam atau dipesan, sehingga dapat berjalan sesuai dengan kebijakan yang telah ditentukan.

Sistem ini juga dirancang untuk memastikan bahwa setiap perpustakaan yang tergabung dapat diakses secara mudah oleh pengguna, memberikan lebih banyak opsi dan fleksibilitas dalam pencarian dan peminjaman buku. Aspek penting lainnya adalah menjaga validasi dan integritas data antar tabel melalui penerapan foreign key dan constraints untuk memastikan keakuratan data transaksi. Pada akhirnya, proyek ini bertujuan untuk menciptakan sistem yang skalabel dan fleksibel, yang mampu berkembang sesuai dengan peningkatan jumlah pengguna dan perpustakaan di masa mendatang, serta memberikan pengalaman pengguna yang optimal dalam pencarian dan akses koleksi buku berdasarkan kategori, judul, atau penulis.

Mission Statement

Misi dari aplikasi e-library ini adalah menyediakan platform yang efisien dan mudah digunakan yang mengelola banyak perpustakaan dengan koleksi buku yang beragam. Sistem ini akan memungkinkan pengguna yang terdaftar untuk meminjam, memesan, dan mengelola interaksi mereka dengan buku-buku dari berbagai perpustakaan. Database akan mendukung fungsi e-library dengan mengorganisir informasi buku, mengelola akun pengguna, menangani transaksi peminjaman dan reservasi, serta memastikan integritas data. Tujuan utamanya adalah untuk mempermudah proses peminjaman buku sambil memastikan akses yang adil terhadap sumber daya yang terbatas, mengoptimalkan pencarian dan

pengambilan informasi buku, serta mendukung pengalaman pengguna yang mulus di seluruh perpustakaan yang terlibat.

Tujuan utama untuk sistem Database ini meliputi:

1. Pengelolaan Efisien Berbagai Perpustakaan: Sistem harus mampu menangani banyak perpustakaan, masing-masing dengan koleksi buku yang beragam, termasuk kategori, penulis, dan jumlah yang tersedia.
2. Memfasilitasi Interaksi Pengguna: Memungkinkan pengguna untuk mendaftar, menelusuri katalog, meminjam, memesan buku yang tidak tersedia, dan mengelola aktivitas pinjaman mereka.
3. Transaksi Peminjaman dan Pemesanan: Melacak periode pinjaman, tanggal pengembalian, serta mengatur antrian pemesanan untuk mengelola permintaan buku.
4. Pencarian dan Kategorisasi: Menyediakan fungsi pencarian dan pemfilteran yang kuat dengan mengkategorikan buku ke dalam genre atau tema (misalnya, pengembangan diri, biografi, fiksi, dan lain-lain).
5. Kepatuhan Terhadap Aturan Peminjaman: Menerapkan batas peminjaman, tenggat waktu pengembalian, dan batasan pemesanan untuk memastikan penggunaan sumber daya perpustakaan yang adil.

Database ini dirancang untuk mendukung fitur-fitur tersebut sambil menawarkan skalabilitas, integritas data, dan kemudahan penggunaan, memastikan baik pengguna maupun administrator perpustakaan dapat berinteraksi dengan sistem secara efisien dan efektif.

Table Structure

Object

Beberapa objek utama yang diperlukan adalah sebagai berikut:

- Users (Pengguna): Informasi tentang pengguna yang terdaftar di platform.
- Libraries (Perpustakaan): Informasi tentang perpustakaan yang tergabung dalam aplikasi.
- Books (Buku): Informasi tentang buku-buku yang tersedia di berbagai perpustakaan.
- Loans (Peminjaman): Data tentang transaksi peminjaman buku oleh pengguna.
- Holds (Pemesanan): Data tentang pemesanan buku yang saat ini tidak tersedia.
- Categories (Kategori): Kategori buku (misalnya: fiksi, biografi, pengembangan diri, dll).

Deskripsi Tabel dan Field

a. Table Users

users		
user_id: int name: varchar(100) email: varchar(100) password: varchar(100) registration_date: int		PK CK

b. Table Libraries

libraries		
library_id: int library_name: varchar(100) location: varchar(100)		PK

c. Table Books

books		
book_id: int title: varchar(100) author: varchar(100) quantity: int library_id: int category_ID: int		PK

d. Table Loans

loans		
loan_id: int user_id: int book_id: int loan_date: int return_date: int		PK

e. Table Holds

holds		
hold_id: int		PK

user_id: int book_id: int hold_date: int queue_position: int		

f. Table Categories

categories		
category_id: int category_name: varchar(100)		PK

Table Relationships

Libraries (One) → Books (Many)

- Foreign Key: library_id di tabel Books.

Categories (One) → Books (Many)

- Foreign Key: category_id di tabel Books.

Users (One) → Loans (Many)

- Foreign Key: user_id di tabel Loans.

Users (One) → Holds (Many)

- Foreign Key: user_id di tabel Holds.

Books (One) → Loans (Many)

- Foreign Key: book_id di tabel Loans.

Books (One) → Holds (Many)

- Foreign Key: book_id di tabel Holds.

Business Rules

1. Tabel Users

- **Fields dan Business Rules:**

- user_id: **Primary Key**, harus unik.
- name: **NOT NULL**, nama pengguna tidak boleh kosong.
- email: **NOT NULL**, **UNIQUE**, email harus unik dan tidak boleh kosong.
- password: **NOT NULL**, kata sandi harus ada.
- registration_date: **NOT NULL**, tanggal pendaftaran harus ada dan diisi dengan format tanggal.

2. Tabel Libraries

- **Fields dan Business Rules:**
  - library_id: **Primary Key**, harus unik.
  - library_name: **NOT NULL**, nama perpustakaan tidak boleh kosong.
  - location: **NOT NULL**, lokasi perpustakaan harus diisi.

3. Tabel Books

- **Fields dan Business Rules:**
  - book_id: **Primary Key**, harus unik.
  - title: **NOT NULL**, judul buku harus diisi.
  - author: **NOT NULL**, nama penulis buku harus diisi.
  - quantity: **NOT NULL**, **CHECK(quantity >= 0)**, jumlah buku tidak boleh kosong dan harus lebih dari atau sama dengan 0.
  - library_id: **Foreign Key**, mengacu ke library_id di tabel Libraries, **NOT NULL**.
  - category_id: **Foreign Key**, mengacu ke category_id di tabel Categories, **NOT NULL**.

4. Tabel Categories

- **Fields dan Business Rules:**
  - category_id: **Primary Key**, harus unik.
  - category_name: **NOT NULL**, nama kategori tidak boleh kosong.

5. Tabel Loans

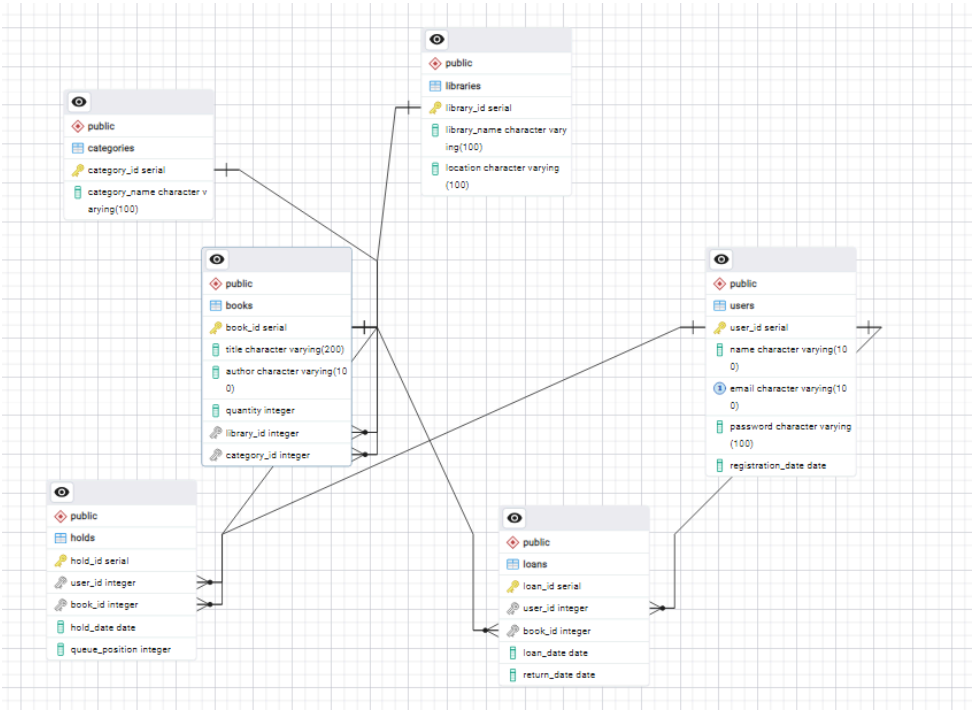
- **Fields dan Business Rules:**
  - loan_id: **Primary Key**, harus unik.
  - user_id: **Foreign Key**, mengacu ke user_id di tabel Users, **NOT NULL**.
  - book_id: **Foreign Key**, mengacu ke book_id di tabel Books, **NOT NULL**.
  - loan_date: **NOT NULL**, tanggal peminjaman harus diisi.
  - return_date: **NOT NULL**, tanggal pengembalian harus diisi dan tidak boleh melebihi 2 minggu dari loan_date.
    - **CHECK(return_date <= loan_date + INTERVAL '14' DAY)**, untuk memastikan bahwa tanggal pengembalian tidak lebih dari 14 hari dari tanggal peminjaman.
  - **Constraint tambahan:** Setiap pengguna hanya dapat meminjam maksimal 2 buku pada waktu yang sama.

        - **CHECK((SELECT COUNT(*) FROM Loans WHERE user_id = current_user_id AND return_date IS NULL) <= 2)**.

6. Tabel Holds

- **Fields dan Business Rules:**
    - hold_id: **Primary Key**, harus unik.
    - user_id: **Foreign Key**, mengacu ke user_id di tabel Users, **NOT NULL**.
    - book_id: **Foreign Key**, mengacu ke book_id di tabel Books, **NOT NULL**.
    - hold_date: **NOT NULL**, tanggal pemesanan harus diisi.
    - queue_position: **NOT NULL**, posisi antrian harus diisi dan bernilai lebih dari 0.
        - **CHECK(queue_position > 0)**.
    - **Constraint tambahan:** Setiap pengguna hanya dapat memesan maksimal 2 buku pada waktu yang sama.
        - **CHECK((SELECT COUNT(*) FROM Holds WHERE user_id = current_user_id AND queue_position IS NOT NULL) <= 2)**.

ERD



Implementing The Design in PostgreSQL

Pengimplementasian dalam design atau Entitiy Relationship Diagram (ERD) dalam databse dengan PostgeSQL mempunyai beberapa langkah:

1. Membuat Database baru dengan nama elibrary_db

   SQL:
   CREATE DATABASE elibrary_db;

   Database ini akan menjadi wadah tempat semua tabel dan data dari sistem e-library akan disimpan.

2. Menyusun Tabel Berdasarkan ERD

   Setelah database dibuat, langkah berikutnya adalah menerjemahkan setiap entitas dari ERD menjadi tabel di PostgreSQL. Proses ini dilakukan dengan menggunakan perintah SQL CREATE TABLE, di mana setiap tabel didefinisikan dengan kolom, tipe data, dan constraint seperti PRIMARY KEY dan FOREIGN KEY.

   SQL :

```
CREATE TABLE Users (
    user_id SERIAL PRIMARY KEY,
    name VARCHAR(100) NOT NULL,
    email VARCHAR(100) UNIQUE NOT NULL,
    password VARCHAR(100) NOT NULL,
    registration_date DATE NOT NULL
);

CREATE TABLE Libraries (
    library_id SERIAL PRIMARY KEY,
    library_name VARCHAR(100) NOT NULL,
    location VARCHAR(100) NOT NULL
);


CREATE TABLE Categories (
    category_id SERIAL PRIMARY KEY,
    category_name VARCHAR(100) NOT NULL
);


CREATE TABLE Books (
    book_id SERIAL PRIMARY KEY,
    title VARCHAR(200) NOT NULL,
```

```
    author VARCHAR(100) NOT NULL,
    quantity INT NOT NULL,
    library_id INT NOT NULL,
    category_id INT NOT NULL,
    FOREIGN KEY (library_id) REFERENCES Libraries(library_id),
    FOREIGN KEY (category_id) REFERENCES Categories(category_id)
);


CREATE TABLE Loans (
    loan_id SERIAL PRIMARY KEY,
    user_id INT NOT NULL,
    book_id INT NOT NULL,
    loan_date DATE NOT NULL,
    return_date DATE NOT NULL,
    FOREIGN KEY (user_id) REFERENCES Users(user_id),
    FOREIGN KEY (book_id) REFERENCES Books(book_id)
);


CREATE TABLE Holds (
    hold_id SERIAL PRIMARY KEY,
    user_id INT NOT NULL,
    book_id INT NOT NULL,
    hold_date DATE NOT NULL,
    queue_position INT NOT NULL,
    FOREIGN KEY (user_id) REFERENCES Users(user_id),
    FOREIGN KEY (book_id) REFERENCES Books(book_id)
);
```

Tabel-tabel ini dibentuk berdasarkan entitas di ERD, dan relasi antar tabel dibuat dengan FOREIGN KEY untuk menjaga integritas data antar tabel. Misalnya, pada tabel Books, library_id dan category_id adalah foreign key yang menghubungkan tabel Books ke Libraries dan Categories.

3. Menyusun Relasi Antar Tabel
   Setiap relasi yang telah digambarkan dalam ERD perlu diwujudkan di basis data melalui foreign key constraints. Ini memastikan bahwa data dalam tabel saling terhubung dengan benar. Sebagai contoh, foreign key user_id pada tabel Loans menghubungkan ke tabel Users, dan foreign key book_id menghubungkan ke tabel Books.

Proses ini mengharuskan tabel-tabel dibuat dalam urutan yang tepat. Sebagai contoh, tabel Books harus dibuat setelah tabel Libraries dan Categories karena tabel Books membutuhkan foreign key yang merujuk pada tabel Libraries dan Categories.

4. Menguji Struktur Basis Data
   Disini, saya menjalankan beberapa perintah SELECT, INSERT, dan UPDATE untuk menguji bahwa constraint (seperti foreign key dan primary key) bekerja dengan benar.

5. Memasukan Dummy Data dengan menggunakan code Python
   Data dummy diimpor ke dalam tabel menggunakan perintah SQL COPY atau fitur import dari pgAdmin. Proses ini memastikan bahwa tabel-tabel terisi dengan data yang sesuai untuk digunakan dalam pengujian lebih lanjut atau analisis.

   SQL :

```
COPY public.users (user_id, name, email, password, registration_date)
FROM 'E:\Downloads\data\users_dummy_data.csv'
DELIMITER ','
CSV HEADER;


COPY public.libraries (library_id, library_name, location)
FROM 'E:\Downloads\data\libraries_dummy_data.csv'
DELIMITER ','
CSV HEADER;


COPY public.categories (category_id, category_name)
FROM 'E:\Downloads\data\categories_dummy_data.csv'
DELIMITER ','
CSV HEADER;


COPY public.books (book_id, title, author, quantity, library_id, category_id)
FROM 'E:\Downloads\data\books_dummy_data.csv'
DELIMITER ','
CSV HEADER;


COPY public.loans (loan_id, user_id, book_id, loan_date, return_date)
FROM 'E:\Downloads\data\loans_dummy_data.csv'
DELIMITER ','
```

```
CSV HEADER;

COPY public.holds (hold_id, user_id, book_id, hold_date, queue_position)
FROM 'E:\Downloads\data\holds_dummy_data.csv'
DELIMITER ','
CSV HEADER;
```

6. Menjalankan Query Analisis

   Contoh:
   ```
   SELECT u.name, COUNT(l.loan_id) AS total_loans
   FROM loans l
   JOIN users u ON l.user_id = u.user_id
   GROUP BY u.name
   ORDER BY total_loans DESC
   LIMIT 1;
   ```

Populating the Database

Proses populating the database melibatkan dua langkah utama: membangun dataset dummy dan memasukkan data tersebut ke dalam basis data PostgreSQL. Berikut adalah deskripsi mengenai kedua langkah tersebut:

1. Generating a Dummy Dataset
   Disini saya menggunakan Python dengan menggunakan Library Faker. lalu, Menyimpan Dataset dalam Format CSV. Setelah data dummy berhasil dibuat, langkah selanjutnya adalah menyimpan dataset tersebut dalam format yang mudah diimpor, seperti CSV.

2. Input Data ke dalam Database PostgreSQL
   Disini saya menggunakan perintah SQL Copy untuk menginput data ke dalam PostgreSQLnya.

Retrieve Data: Questions and Analysis

Pada bagian ini, kita akan membahas lima pertanyaan yang sebelumnya telah dirumuskan untuk mengekstraksi data dari basis data e-library. Saya akan menjelaskan setiap pertanyaan, alasan di balik pertanyaan tersebut, menjalankan query SQL untuk mengambil data, dan memberikan analisis dari hasil yang diperoleh.

1. Berapa banyak buku yang tersedia di setiap perpustakaan?

   SQL :
   ```
   SELECT l.library_name, SUM(b.quantity) AS total_books
   ```

```
FROM books b
JOIN libraries l ON b.library_id = l.library_id
GROUP BY l.library_name;
```

Output :

	library_name character varying (100)	total_books bigint
1	Rodriguez Ltd	33
2	Valenzuela, Cobb and Wright	15
3	Hester, Rojas and Jones	39

Disini, kita dapat melihat bahwa query ini menampilkan nama perpustakaan beserta total jumlah buku yang tersedia di setiap perpustakaan.

2. Siapa pengguna yang paling sering meminjam buku?

SQL:

```
SELECT u.name, COUNT(l.loan_id) AS total_loans
FROM loans l
JOIN users u ON l.user_id = u.user_id
GROUP BY u.name
ORDER BY total_loans DESC
LIMIT 1;
```

Output:

	name character varying (100)	total_loans bigint
1	Adam Smith	3

Kita dapat lihat disini bahwa Query ini menampilkan pengguna yang paling sering meminjam buku, berdasarkan jumlah transaksi peminjaman yang dilakukan, yaitu Adam Smith dengan total loans atau peminjamannya adalah 3.

3. Buku kategori apa yang paling sering dipinjam?

SQL:

```
SELECT c.category_name, COUNT(l.loan_id) AS total_loans
FROM loans l
JOIN books b ON l.book_id = b.book_id
JOIN categories c ON b.category_id = c.category_id
```

GROUP BY c.category_name
ORDER BY total_loans DESC
LIMIT 1;

Output:

	category_name character varying (100)	total_loans bigint
1	Science Fiction	6

Hasil dari query ini akan menunjukkan kategori buku yang paling sering dipinjam, seperti "Fiksi" atau "Biografi".

4. Berapa jumlah buku yang saat ini sedang dipesan (on-hold) oleh pengguna?
   SQL :
   SELECT COUNT(hold_id) AS total_on_hold
   FROM holds;

   Output:

	total_on_hold bigint
1	5

   Hasil dari query ini menunjukkan jumlah total buku yang sedang dalam status pemesanan.

5. Berapa lama rata-rata waktu peminjaman buku sebelum dikembalikan?
   SELECT AVG(return_date - loan_date) AS average_loan_duration
   FROM loans;

   Output:

	average_loan_duration numeric
1	2.1000000000000000

   Hasil query ini menunjukkan rata-rata durasi peminjaman buku (dalam hari) antara tanggal peminjaman dan tanggal pengembalian.