

# **LAPORAN AKHIR PROYEK SISTEM BASIS DATA**

## **Sistem Perpustakaan Digital**

### **KELOMPOK-07**

**Anggota :**

<b>Nama</b>	<b>NIM</b>
Winda V. Sembiring	11323013
Ricky J. Silaen	11323028
Vinci G. Baringbing	11323051
Febyanti Hutahaeen	11323055

**Prodi: D-III Teknologi Informasi 2023**

**Fakultas Vokasi  
Intitute Teknologi Del  
2024/2025**



**Proyek SBD 2024  
Institut Teknologi Del**

---

*Tanggal : 11-12-2024*

*Jumlah Halaman :*

## DAFTAR ISI

DAFTAR ISI .....	2
1 PENDAHULUAN .....	4
1.1 Latar Belakang .....	4
1.2 Rumusan Masalah .....	4
1.3 Tujuan .....	4
1.4 Manfaat .....	4
2 URAIAN PROYEK .....	5
2.1 Tempat dan Waktu Pelaksanaan.....	5
2.1.1 Tempat.....	5
2.1.2 Waktu Pelaksanaan:.....	5
2.2 Stack Teknologi.....	6
2.3 Design ER-Diagram .....	6
3 IMPLEMENTASI.....	7
3.1 Authorization.....	7
3.1.1 Role Admin.....	7
3.1.2 Role Anggota.....	7
3.2 Create Tabel User.....	8
3.2.1 Admin .....	8
3.2.2 Anggota .....	10
3.3 Create Table .....	12
3.3.1 Tabel Kategori .....	12
3.3.2 Tabel Buku.....	13
3.3.3 Tabel log_peminjaman .....	13
3.3.4 Tabel log_denda.....	14
3.4 Insert Data Menggunakan Crawling.....	15
3.4.1 Import Library .....	15
3.4.2 Koneksi ke database PostgreSQL .....	15
3.4.3 Mengambil data buku dari Google Books API.....	16
3.4.4 Insert Kategori Baru.....	16
3.4.5 Insert data buku.....	17
3.4.6 Eksport data ke file csv .....	17
3.4.7 Mengambil data buku untuk setiap kategori .....	17
3.4.8 Export data kategori dan buku ke file .csv .....	18
3.4.9 Tutup Koneksi .....	18
3.5 Create Trigger Insert Data .....	18
3.5.1 Tabel kategori .....	18
3.5.2 Tabel Anggota .....	20
3.5.3 Tabel Buku.....	23
3.6 Create Proses Peminjaman Buku menggunakan Trigger dan Fungsi .....	26
3.6.1 Create Function.....	26
3.6.2 Create Trigger.....	27
3.6.3 Pengujian .....	27
3.7 Create Proses Pengembalian Buku menggunakan Transaksi Manual .....	29
3.7.1 Create Transaction .....	29
3.7.2 Pengujian .....	30
3.8 Proses Pembayaran Denda menggunakan Transaksi Manual.....	31
3.8.1 Create transaction .....	31
3.8.2 Pengujian .....	32
3.9 Create Tabel Laporan .....	33
3.9.1 Tabel laporan jumlah denda peminjaman setiap anggota .....	33
3.9.2 Tabel laporan buku paling sering dipinjam per-tahun .....	34
3.9.3 Tabel laporan buku paling sering dipinjam per-bulan .....	35
3.9.4 Tabel laporan buku paling sering dipinjam per-minggu .....	36
3.10 Stored Procedure untuk Laporan buku paling sering dibaca .....	36
3.10.1 Create Stored Procedure.....	37
3.10.2 Memanggil laporan buku terpopuler .....	37

3.11	Cursor untuk Laporan buku paling sering dibaca .....	38
3.11.1	Create cursor .....	38
3.11.2	Memanggil laporan buku terpopuler .....	38
3.12	Backup dan Restore .....	39
3.12.1	Backup database.....	39
3.12.2	Restore database.....	39
3.13	Fungsi Mencari Buku .....	40
3.13.1	Create Function .....	40
3.13.2	Pengujian.....	41
4	Kesimpulan dan Saran.....	42
4.1	Kesimpulan .....	42
4.2	Saran.....	42
HASIL PRESENTASI.....		43

# **1 PENDAHULUAN**

## **1.1 Latar Belakang**

Buku dan informasi telah menjadi bagian penting dalam kehidupan manusia, baik sebagai sarana pembelajaran, hiburan, maupun pengembangan wawasan. Di era digital ini, perpustakaan digital menjadi solusi untuk memberikan akses mudah dan cepat ke berbagai koleksi secara online.

Sistem perpustakaan digital sederhana yang menggunakan sintaks SQL dirancang untuk memenuhi kebutuhan dasar pengelolaan koleksi, seperti pencatatan, dan peminjaman buku secara efisien. Dengan pendekatan ini, pengguna akan dapat mengakses dan mengelola data dengan lebih mudah melalui antarmuka sederhana yang berfokus pada fungsionalitas esensial. Sistem ini menjadi langkah awal yang penting dalam mewujudkan perpustakaan digital yang lebih terintegrasi dan canggih di masa depan.

## **1.2 Rumusan Masalah**

Berdasarkan latar belakang yang telah dijelaskan, rumusan masalah yang diangkat dalam pengembangan sistem perpustakaan digital adalah sebagai berikut:

1. Bagaimana merancang sistem perpustakaan digital yang dapat mencatat log peminjaman buku secara efisien ?
2. Bagaimana sistem dapat menghitung dan mencatat log denda secara otomatis berdasarkan tanggal pengembalian yang terlambat?
3. Bagaimana sistem dapat menyimpan dan menampilkan laporan buku pengguna secara terstruktur dan mudah diakses?

## **1.3 Tujuan**

Tujuan utama dari pengembangan sistem perpustakaan digital adalah sebagai berikut:

1. Membangun fitur pencatatan log peminjaman yang dapat mencatat setiap transaksi peminjaman buku dengan akurat dan efisien.
2. Mengimplementasikan fitur untuk menghitung denda otomatis dan mencatat log denda berdasarkan peraturan yang telah ditentukan (misalnya, denda per hari keterlambatan).
3. Menyediakan laporan yang mudah diakses oleh petugas perpustakaan, baik laporan harian, mingguan, maupun tahunan, guna memudahkan pengelolaan dan pemantauan.

## **1.4 Manfaat**

Pengembangan sistem perpustakaan digital ini memberikan berbagai manfaat, baik bagi pengguna maupun pengelola perpustakaan, antara lain:

1. Memudahkan pemantauan dan analisis tren peminjaman melalui laporan otomatis harian, mingguan, atau tahunan.
2. Meningkatkan efisiensi dan akurasi pencatatan peminjaman, pengembalian, serta denda secara otomatis.

## 2 URAIAN PROYEK

### 2.1 Tempat dan Waktu Pelaksanaan

Proyek pengembangan Sistem Perpustakaan Digital ini dilaksanakan di **Institut Teknologi Del** selama periode perkuliahan, dengan rincian sebagai berikut:

#### 2.1.1 Tempat

- Ruang kelas Gedung 5 dan lingkungan kampus Institut Teknologi Del.
- Pengembangan dilakukan menggunakan fasilitas dan perangkat pribadi .

#### 2.1.2 Waktu Pelaksanaan:

Peiode waktu pelaksanaannya dimulai dari Minggu ke-14 perkuliahan (28 November 2024) hingga minggu ke-16 ( 10 Desember 2024) perkuliahan. Untuk melihat lebih jelas tahapan pelaksanaannya silahkan lihat **Tabel 1. Log Aktivitas** berikut.

**Tabel 1. Log Aktivitas**

Minggu	Sesi	Aktivitas
14	1	1. Design ER-Diagram
	2	1. Design ER-Diagram II 2. Create Tabel User
	3	1. Create Tabel Kategori, Buku, Log Peminjaman 2. Create Tabel log_denda 3. Insert Dummy Data
15	1	1. Create Authorization 2. Memberi Hak Akses 3. Create Trigger Insert Data
	2	1. Create Trigger Insert Data 2. Create Trigger dan Fungsi Proses Peminjaman 3. Create Transaksi Manual Proses Pengembalian
	3	1. Create Transaksi Manual Pembayaran Denda 2. Create View anggota_peminjaman_denda 3. Create View Laporan Buku Pertahun 4. Create View Laporan Buku Perbulan
16	1	1. Create Stored Procedure Laporan Bulu Paling Sering dibaca 2. Create Cursor Laporan Buku paling sering dibaca
	2	1. Menambahkan data pada tabel buku dengan crawling 2. Menyusun Laporan Akhir.
	3	1. Lanjutan membuat crawling 2. Lanjutan Menyusun Laporan dan finalisasi laporan 3. Membuat slide presentasi 4. Backup dan Restore 5. Create View Cari Buku

## 2.2 Stack Teknologi

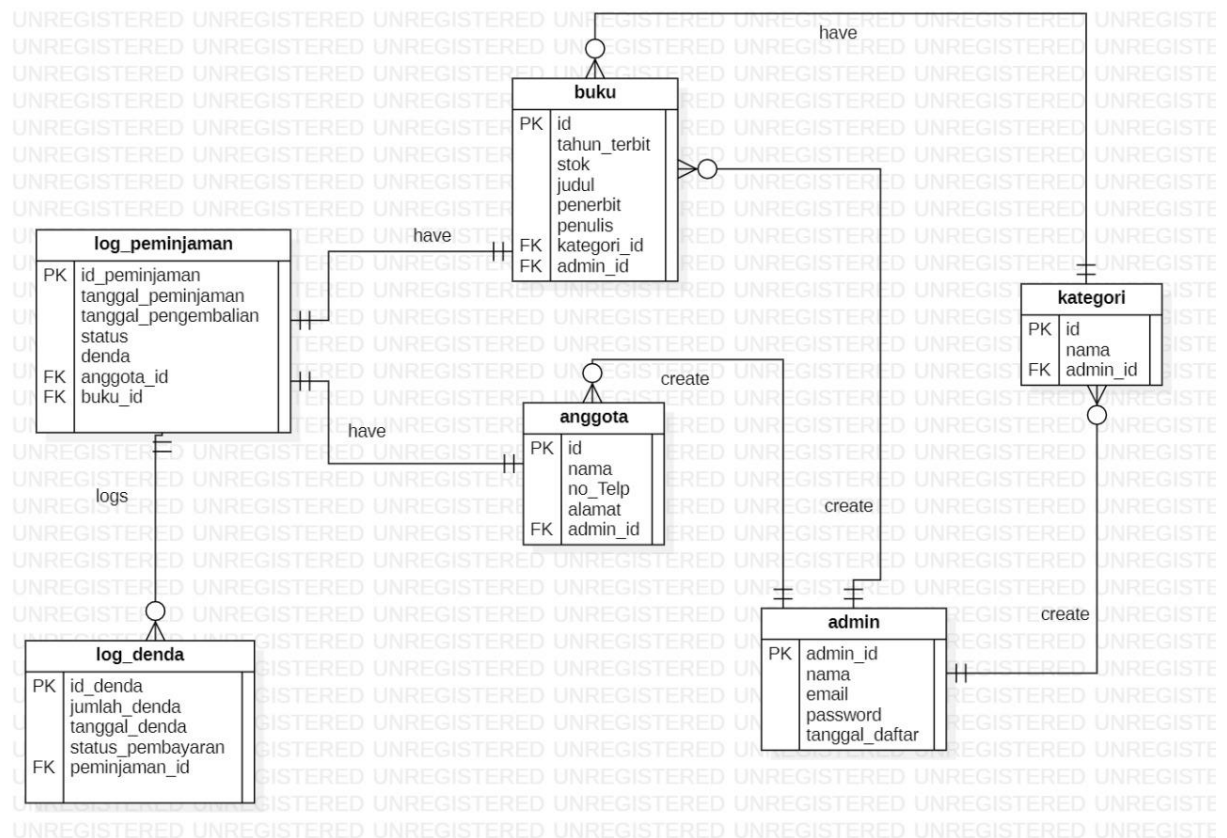
Berikut adalah detail teknologi, versi yang digunakan, jenis teknologi, serta kegunaannya dalam proyek pengembangan Sistem Perpustakaan Digital. Lihat **Tabel 2. Stack Teknologi** berikut.

**Tabel 2. Stack Teknologi**

No	Nama Teknologi	Versi	Jenis	Kegunaan
1	PostgreSQL	16	Database	Sebagai tempat untuk menyimpan data sistem.
2	Canva	N/A	Desain	Membuat desain presentasi atau poster.
3	StarUML	5	UML Modelling	Membuat ER-Diagram dan model UML.
4	Trello	N/A	Manajemen Proyek	Mengatur dan mengelola proyek secara efisien.
5	GitHub	N/A	Version Control	Tempat kolaborasi dan manajemen kode sumber.

## 2.3 Design ER-Diagram

Sistem Perpustakaan Digital\_07 memiliki dua peran utama, yaitu admin dan anggota. Admin memiliki hak penuh untuk melakukan CRUD (Create, Read, Update, Delete) pada tabel anggota, buku, dan kategori. Sementara itu, anggota hanya dapat mencari buku, meminjam buku, mengembalikan buku, serta membayar denda apabila pengembalian buku melewati batas waktu yang telah ditentukan. Sistem ini dirancang untuk mendukung pengelolaan perpustakaan secara efisien dan terstruktur.



### 3 IMPLEMENTASI

#### 3.1 Authorization

Pada authorization bertujuan untuk mengelola hak akses pengguna dalam system

##### 3.1.1 Role Admin

Role admin memberikan hak akses penuh bagi pengguna untuk mengelola serta mengakses semua data yang ada dalam system ,yang dimana admin dapat melakukan CRUD (Create,Update,Delete)

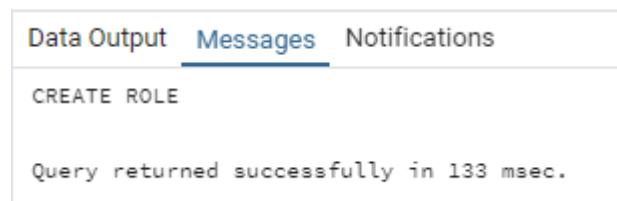
###### 3.1.1.1 Create Role Admin

Langkah pertama untuk membuat role admin dengan cara membuat perintah CREATE ROLE ADMIN

- Berikut merupakan syntaxnya :

```
-- 3.1.1.1 Create Role Admin  
CREATE ROLE admin;
```

- Output yang dihasilkan:



```
Data Output Messages Notifications  
CREATE ROLE  
  
Query returned successfully in 133 msec.
```

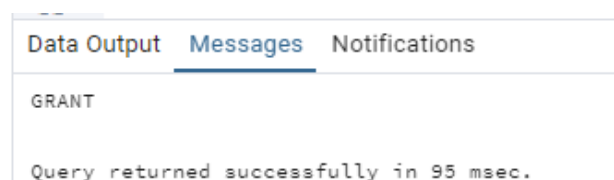
###### 3.1.1.2 Memberikan hak akses untuk admin

Query ini memberikan hak akses penuh kepada role admin sehingga berfungsi untuk mengelola seluruh data seperti mengakses, mengubah, menambah, atau menghapus

- Berikut merupakan syntaxnya :

```
-- 3.1.1.2 Memberikan hak akses untuk admin  
-- Memberikan hak akses ALL PRIVILEGES pada skema public kepada admin  
GRANT ALL PRIVILEGES ON SCHEMA public TO admin;
```

- Output yang dihasilkan:



```
Data Output Messages Notifications  
GRANT  
  
Query returned successfully in 95 msec.
```

##### 3.1.2 Role Anggota

Role anggota memiliki hak akses terbatas dalam system ini karena disini role anggota hanya berfungsi untuk melakukan log peminjaman untuk meminjam buku dan melakukan pembayaran denda apabila terjadi keterlambatan saat pengembalian buku .

###### 3.1.2.1 Create Role Anggota

Untuk membuat Role anggota dapat dilakukan dengan membuat perintah “Create Role Anggota”

- Berikut merupakan syntaxnya:

```
-- 3.1.2 Role Anggota  
-- 3.1.2.1 Create Role Anggota  
CREATE ROLE anggota;
```

- **Output yang dihasilkan:**

```

Data Output Messages Notifications
CREATE ROLE

Query returned successfully in 85 msec.

```

### 3.1.2.2 Memberikan hak akses untuk anggota

Query ini memberikan hak akses kepada role *anggota* untuk melihat, melakukan peminjaman, membayar denda tabel *buku*, *log\_peminjaman*, dan *log\_denda*. Dengan hak akses ini, *anggota* dapat mengakses daftar buku yang tersedia, riwayat peminjaman yang telah dilakukan, serta informasi terkait denda yang dikenakan akibat keterlambatan pengembalian buku. Namun, *anggota* hanya memiliki akses untuk membaca data tanpa kemampuan untuk mengubah, menambah data

- **Berikut merupakan syntaxnya:**

```

-- 3.1.2.2 Memberikan hak akses untuk anggota
GRANT SELECT, INSERT, UPDATE ON TABLE buku, log_peminjaman, log_denda TO anggota;

```

- **Output yang dihasilkan:**

```

Data Output Messages Notifications
GRANT

Query returned successfully in 92 msec.

```

## 3.2 Create Tabel User

Tabel ini berfungsi untuk memisahkan data pengguna sesuai dengan peran serta hak akses yang dimiliki masing-masing.

### 3.2.1 Admin

#### 3.2.1.1 Create Tabel

Query ini berfungsi untuk membuat tabel yang menyimpan data admin dalam system.

- **Berikut merupakan syntaxnya:**

```

-- 3.2.1.1 Create Tabel
CREATE TABLE admin (
    id SERIAL PRIMARY KEY,
    nama VARCHAR(255) NOT NULL,
    email VARCHAR(255) UNIQUE NOT NULL,
    password VARCHAR(255) NOT NULL,
    tanggal_daftar TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);

```

- **Output yang dihasilkan:**

```

Data Output Messages Notifications
CREATE TABLE

Query returned successfully in 64 msec.

```



### 3.2.1.2 Insert Data

Berikut merupakan data yang akan ditambahkan pada table admin . Data tersebut mencakup beberapa informasi penting, seperti nama admin, email, dan password, yang dimasukkan untuk mengelola sistem

- Berikut merupakan syntaxnya:

```
-- 3.2.1.2 Insert Data
INSERT INTO admin (nama, email, password)
VALUES
('FebyantiHutahaeen', 'feby@gmail.com', 'feby1602'),
('VinciG.Baringbing', 'vinci@gmail.com', 'vinci2707'),
('WindaSembiring', 'winda@gmail.com', 'winda1234'),
('RickySilaen', 'ricky@gmail.com', 'ricky5678');
```

- Output yang dihasilkan:

Data Output Messages Notifications

---

INSERT 0 4

Query returned successfully in 57 msec.

Setelah proses penambahan data selesai, dilakukan pemeriksaan untuk memastikan bahwa data telah berhasil disimpan ke dalam tabel.

- Berikut merupakan syntaxnya :

```
-- Cek data dalam tabel
SELECT * FROM admin;
```

- Output yang dihasilkan:

	id [PK] integer	nama character varying (255)	email character varying (255)	password character varying (255)	tanggal_daftar timestamp without time zone
1	1	FebyantiHutahaeen	feby@gmail.com	feby1602	2024-12-10 17:14:48.601182
2	2	VinciG.Baringbing	vinci@gmail.com	vinci2707	2024-12-10 17:14:48.601182
3	3	WindaSembiring	winda@gmail.com	winda1234	2024-12-10 17:14:48.601182
4	4	RickySilaen	ricky@gmail.com	ricky5678	2024-12-10 17:14:48.601182

### 3.2.1.3 Menambahkan admin yang baru diinsert ke dalam role admin

Query ini bertujuan untuk menambahkan admin yang telah diinsert sebelumnya ke dalam role admin, sehingga memberikan mereka hak akses penuh untuk mengelola sistem

- Berikut merupakan syntaxnya:

```
-- ini untuk memberikan hak akses kepada admin
GRANT admin TO "FebyantiHutahaeen",
               "VinciG.Baringbing",
               "WindaSembiring",
               "RickySilaen";
```

- Output yang dihasilkan:

Data Output Messages Notifications

---

GRANT ROLE

Query returned successfully in 77 msec.

## 3.2.2 Anggota

### 3.2.2.1 Create Tipe Data Alamat

Query ini bertujuan untuk menyimpan informasi alamat dengan data yang terstruktur .

- Berikut merupakan syntaxnya:

```
-- 3.2.2.1 Create Tipe Data Alamat
CREATE TYPE alamat_type AS (
    jalan VARCHAR(255),
    kota VARCHAR(100),
    provinsi VARCHAR(100),
    kode_pos VARCHAR(10),
    negara VARCHAR(100)
);
```

- Output yang dihasilkan:

Data Output	Messages	Notifications
CREATE TYPE		
Query returned successfully in 103 msec.		

### 3.2.2.2 Create Tabel

Query ini berfungsi untuk membuat tabel yang menyimpan anggota dalam system.

- Berikut merupakan syntaxnya :

```
-- 3.2.2.2 Create Tabel
CREATE TABLE anggota (
    id SERIAL PRIMARY KEY,
    nama VARCHAR(255) NOT NULL,
    email VARCHAR(255) UNIQUE NOT NULL,
    password VARCHAR(255) NOT NULL,
    alamat alamat_type,
    telepon VARCHAR(15),
    tanggal_daftar TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
```

- Output yang dihasilkan:

Data Output	Messages	Notifications
CREATE TABLE		
Query returned successfully in 134 msec.		

### 3.2.2.3 Insert Data

Berikut merupakan data yang akan ditambahkan pada table anggota . Data tersebut mencakup beberapa informasi penting, seperti nama admin, email, password, alamat ,telepon ,tanggal daftar yang dibuat untuk mencatat anngota.

- Berikut merupakan syntaxnya :

```
-- 3.2.2.3 Insert Data
INSERT INTO anggota (nama, email, password, alamat, telepon, tanggal_daftar)
VALUES
('Febiola Alya Hutagalung', 'febiola@gmail.com', 'febiola123',
('Jalan Raya Toba No. 1', 'Parapat', 'Sumatera Utara', '22381', 'Indonesia'), '081234567890', CURRENT_TIMESTAMP),
('Jessica Anastasya Purba', 'jessica.purba@gmail.com', 'jessica123',
('Jalan Toba Indah No. 5', 'Balige', 'Sumatera Utara', '22382', 'Indonesia'), '081234567891', CURRENT_TIMESTAMP),
('Aan Kristian Sitinjak', 'aan.sitinjak@gmail.com', 'aan123',
('Jalan Sisingamangaraja No. 12', 'Toba Samosir', 'Sumatera Utara', '22383', 'Indonesia'), '081234567892', CURRENT_TIMESTAMP),
('Ferry Bastian Siagian', 'ferry.siaagian@gmail.com', 'ferry123',
('Jalan Batu Gantung No. 3', 'Sibolga', 'Sumatera Utara', '22384', 'Indonesia'), '081234567893', CURRENT_TIMESTAMP),
('Kevin Christian B. Rumapea', 'kevin.rumapea@gmail.com', 'kevin123',
('Jalan Pintu Angin No. 9', 'Samosir', 'Sumatera Utara', '22385', 'Indonesia'), '081234567894', CURRENT_TIMESTAMP),
('Estina Pangaribuan', 'estina.pangaribuan@gmail.com', 'estina123',
('Jalan Raya Samosir No. 2', 'Tuktuk', 'Sumatera Utara', '22386', 'Indonesia'), '081234567895', CURRENT_TIMESTAMP),
('Valencia L Tobing', 'valencia.tobing@gmail.com', 'valencia123',
('Jalan Parapat No. 7', 'Parapat', 'Sumatera Utara', '22387', 'Indonesia'), '081234567896', CURRENT_TIMESTAMP),
('Andri Agung Exaudi Sigiros', 'andri.sigiros@gmail.com', 'andri123',
('Jalan Danau Toba No. 8', 'Balige', 'Sumatera Utara', '22388', 'Indonesia'), '081234567897', CURRENT_TIMESTAMP),
('Ayu Enissa Marettys Sinaga', 'ayu.sinaga@gmail.com', 'ayu123',
('Jalan Raja Sisingamangaraja No. 10', 'Ajibata', 'Sumatera Utara', '22389', 'Indonesia'), '081234567898', CURRENT_TIMESTAMP),
('Indra Aziz Nugraha', 'indra.nugraha@gmail.com', 'indra123',
('Jalan Tuktuk No. 6', 'Samosir', 'Sumatera Utara', '22390', 'Indonesia'), '081234567899', CURRENT_TIMESTAMP),
('Daniel Haganta Ginting', 'daniel.ginting@gmail.com', 'daniel123',
('Jalan Merdeka No. 11', 'Toba Samosir', 'Sumatera Utara', '22391', 'Indonesia'), '081234567900', CURRENT_TIMESTAMP),
('Diva Lorenza Marbun', 'diva.marbun@gmail.com', 'diva123',
('Jalan Raya Samosir No. 15', 'Parapat', 'Sumatera Utara', '22392', 'Indonesia'), '081234567901', CURRENT_TIMESTAMP);
```

- Output yang dihasilkan:

Data Output Messages Notifications

INSERT 0 12

Query returned successfully in 84 msec.

Setelah proses penambahan data selesai, dilakukan pemeriksaan untuk memastikan bahwa data telah berhasil disimpan ke dalam tabel

- Berikut merupakan syntaxnya :

```
-- Cek data dalam tabel
SELECT * FROM anggota;
```

- Output yang dihasilkan:

	id [PK] integer	nama character varying (255)	email character varying (255)	password character varying (255)	alamat alamat_type	telepon character varying (15)	tanggal_daftar timestamp without time zone
1	1	Febiola Alya Hutagalung	febiola@gmail.com	febiola123	('Jalan Raya Toba No. 1', 'Parapat', 'Sumatera Utara', '22381', 'Indonesia')	081234567890	2024-12-10 17:30:38.536591
2	2	Jessica Anastasya Purba	jessica.purba@gmail.com	jessica123	('Jalan Toba Indah No. 5', 'Balige', 'Sumatera Utara', '22382', 'Indonesia')	081234567891	2024-12-10 17:30:38.536591
3	3	Aan Kristian Sitinjak	aan.sitinjak@gmail.com	aan123	('Jalan Sisingamangaraja No. 12', 'Toba Samosir', 'Sumatera Utara', '22383', 'Indonesia')	081234567892	2024-12-10 17:30:38.536591
4	4	Ferry Bastian Siagian	ferry.siaagian@gmail.com	ferry123	('Jalan Batu Gantung No. 3', 'Sibolga', 'Sumatera Utara', '22384', 'Indonesia')	081234567893	2024-12-10 17:30:38.536591
5	5	Kevin Christian B. Rumapea	kevin.rumapea@gmail.com	kevin123	('Jalan Pintu Angin No. 9', 'Samosir', 'Sumatera Utara', '22385', 'Indonesia')	081234567894	2024-12-10 17:30:38.536591
6	6	Estina Pangaribuan	estina.pangaribuan@gmail.com	estina123	('Jalan Raya Samosir No. 2', 'Tuktuk', 'Sumatera Utara', '22386', 'Indonesia')	081234567895	2024-12-10 17:30:38.536591
7	7	Valencia L Tobing	valencia.tobing@gmail.com	valencia123	('Jalan Parapat No. 7', 'Parapat', 'Sumatera Utara', '22387', 'Indonesia')	081234567896	2024-12-10 17:30:38.536591
8	8	Andri Agung Exaudi Sigiros	andri.sigiros@gmail.com	andri123	('Jalan Danau Toba No. 8', 'Balige', 'Sumatera Utara', '22388', 'Indonesia')	081234567897	2024-12-10 17:30:38.536591
9	9	Ayu Enissa Marettys Sinaga	ayu.sinaga@gmail.com	ayu123	('Jalan Raja Sisingamangaraja No. 10', 'Ajibata', 'Sumatera Utara', '22389', 'Indonesia')	081234567898	2024-12-10 17:30:38.536591
10	10	Indra Aziz Nugraha	indra.nugraha@gmail.com	indra123	('Jalan Tuktuk No. 6', 'Samosir', 'Sumatera Utara', '22390', 'Indonesia')	081234567899	2024-12-10 17:30:38.536591
11	11	Daniel Haganta Ginting	daniel.ginting@gmail.com	daniel123	('Jalan Merdeka No. 11', 'Toba Samosir', 'Sumatera Utara', '22391', 'Indonesia')	081234567900	2024-12-10 17:30:38.536591
12	12	Diva Lorenza Marbun	diva.marbun@gmail.com	diva123	('Jalan Raya Samosir No. 15', 'Parapat', 'Sumatera Utara', '22392', 'Indonesia')	081234567901	2024-12-10 17:30:38.536591

### 3.2.2.4 Menambahkan anggota yang baru diinsert ke dalam role anggota

Query ini bertujuan untuk memberikan hak akses kepada anggota yang baru dimasukkan ke dalam table anggota sehingga dengan menambahkan role tersebut anggota memiliki hak akses terbatas yang dimana hanya dapat melakukan log peminjaman untuk meminjam buku dan melakukan pembayaran denda apabila terjadi keterlambatan saat pengembalian buku.

- Berikut merupakan syntaxnya :

```
GRANT anggota TO "Febiola Alya Hutagalung",
                "Jessica Anastasya Purba",
                "Aan Kristian Sitinjak",
                "Ferry Bastian Siagian",
                "Kevin Christian B. Rumapea",
                "Estina Pangaribuan",
                "Valencia L Tobing",
                "Andri Agung Exaudi Sigiroy",
                "Ayu Enissa Maretty Sinaga",
                "Indra Aziz Nugraha",
                "Daniel Haganta Ginting",
                "Diva Lorenza Marbun";
```

- Output yang dihasilkan:

Data Output	Messages	Notifications
GRANT ROLE		
Query returned successfully in 84 msec.		

### 3.3 Create Table

Query ini berfungsi untuk membuat tabel yang menyimpan kategori dalam system , khususnya untuk mengelompokkan buku-buku yang ada di dalam perpustakaan berdasarkan jenis

#### 3.3.1 Tabel Kategori

Pada table kategori fungsinya untuk menyimpan informasi mengenai kategori-kategori buku yang tersedia dalam sistem perpustakaan

- Berikut merupakan syntaxnya :

```
-- 3.3.1 Tabel Kategori
CREATE TABLE kategori (
    id SERIAL PRIMARY KEY,
    nama VARCHAR(255) NOT NULL
);
```

- Output yang dihasilkan:

Data Output	Messages	Notifications
CREATE TABLE		
Query returned successfully in 89 msec.		

Setelah pembuatan table maka dilakukan pengecekan untuk memeriksa data yang ada dalam tabel

- Berikut merupakan syntaxnya :

```
-- Cek data dalam tabel
SELECT * FROM kategori;
```

- Output yang dihasilkan:

id	nama
[PK] integer	character varying (255)

### 3.3.2 Tabel Buku

Query ini digunakan untuk membuat table buku yang menyimpan informasi mengenai buku-buku dalam sistem perpustakaan, termasuk data seperti judul, penulis, penerbit, tahun terbit, stok, dan kategori.

- Berikut merupakan syntaxnya :

```
-- 3.3.2 Tabel buku
CREATE TABLE buku (
  id SERIAL PRIMARY KEY,
  judul VARCHAR(255) UNIQUE NOT NULL DEFAULT 'Unknown',
  penulis VARCHAR(255) NOT NULL DEFAULT 'Unknown',
  penerbit VARCHAR(255) NOT NULL DEFAULT 'Unknown',
  tahunTerbit INTEGER NOT NULL DEFAULT 0,
  stok INT NOT NULL DEFAULT 10,
  kategori_id INT REFERENCES kategori(id) ON DELETE CASCADE
);
```

- Output yang dihasilkan:

Data Output	Messages	Notifications
CREATE TABLE		
Query returned successfully in 92 msec.		

Setelah pembuatan table maka dilakukan pengecekan untuk memeriksa data yang ada dalam tabel

- Berikut merupakan syntaxnya :

```
-- Cek data dalam tabel
SELECT * FROM buku;
```

- Output yang dihasilkan:

id	judul	penulis	penerbit	tahunTerbit	stok	kategori_id
[PK] integer	character varying (255)	character varying (255)	character varying (255)	integer	integer	integer

### 3.3.3 Tabel log\_peminjaman

Query ini digunakan untuk melakukan penyimpanan informasi tentang transaksi peminjaman buku di perpustakaan termasuk informasi tentang anggota, buku yang dipinjam, tanggal peminjaman, tanggal pengembalian, status peminjaman, dan denda jika ada keterlambatan

- Berikut merupakan syntaxnya :

```
-- 3.3.3 Tabel log_peminjaman
CREATE TABLE log_peminjaman (
  id_peminjaman SERIAL PRIMARY KEY,
  anggota_id INT REFERENCES anggota(id) ON DELETE CASCADE,
  buku_id INT REFERENCES buku(id) ON DELETE CASCADE,
  tanggal_peminjaman DATE NOT NULL,
  tanggal_pengembalian DATE,
  status VARCHAR(50) CHECK (status IN ('Kembali', 'Dipinjam', 'Terlambat')),
  denda NUMERIC(10, 2) DEFAULT 0.00
);
```

- **Output yang dihasilkan:**

Data Output Messages Notifications

---

CREATE TABLE

Query returned successfully in 107 msec.

Setelah pembuatan table maka dilakukan pengecekan untuk memeriksa data yang ada dalam tabel

- **Berikut merupakan syntaxnya :**

```
-- Cek data dalam tabel
SELECT * FROM log_peminjaman;
```

- **Output yang dihasilkan:**

id_peminjaman [PK] integer	anggota_id integer	buku_id integer	tanggal_peminjaman date	tanggal_pengembalian date	status character varying (50)	denda numeric (10,2)	

### 3.3.4 Tabel log\_denda

Query ini digunakan untuk membuat table log\_denda yang menyimpan data denda terkait transaksi peminjaman buku, termasuk informasi mengenai peminjaman yang terkait, jumlah denda, tanggal denda, dan status pembayaran denda. Tabel ini menghubungkan data dengan table log\_peminjaman yang mengatur penghapusan data secara otomatis sehingga dapat memeriksa status pembayaran apakah 'Belum Dibayar' atau 'Sudah Dibayar'

- **Berikut merupakan syntaxnya :**

```
-- 3.3.4 Tabel log_denda
CREATE TABLE log_denda (
  id_denda SERIAL PRIMARY KEY,
  peminjaman_id INT REFERENCES log_peminjaman(id_peminjaman) ON DELETE CASCADE,
  jumlah_denda NUMERIC(10, 2) NOT NULL,
  tanggal_denda DATE NOT NULL DEFAULT CURRENT_DATE,
  status_pembayaran VARCHAR(20) DEFAULT 'Belum Dibayar'
  CHECK (status_pembayaran IN ('Belum Dibayar', 'Sudah Dibayar'))
);
```

- **Output yang dihasilkan:**

Data Output Messages Notifications

---

CREATE TABLE

Query returned successfully in 105 msec.

Setelah pembuatan table maka dilakukan pengecekan untuk memeriksa data yang ada dalam tabel

- **Berikut merupakan syntaxnya :**

```
-- Cek data dalam tabel
SELECT * FROM log_denda;
```

- **Output yang dihasilkan:**

SQL					
id_denda [PK] integer	peminjaman_id integer	jumlah_denda numeric (10,2)	tanggal_denda date	status_pembayaran character varying (20)	

### 3.4 Insert Data Menggunakan Crawling

#### 3.4.1 Import Library

```
import csv
import requests
import psycopg2
```

[1] ✓ 1.1s

#### 3.4.2 Koneksi ke database PostgreSQL

```
conn = psycopg2.connect(
    host="localhost",
    database="Perpustakaan_Kel-07_SBD",
    user="postgres",
    password="feby1602",
    port=5432
)

cursor = conn.cursor()
print("Koneksi ke database berhasil!")
```

[2] ✓ 0.1s

... Koneksi ke database berhasil!

### 3.4.3 Mengambil data buku dari Google Books API

```
def fetch_books_from_api(query="", max_results=100):
    url = f"https://www.googleapis.com/books/v1/volumes?q={query}&maxResults=40"
    books = []
    while len(books) < max_results:
        response = requests.get(url)
        if response.status_code == 200:
            data = response.json()
            new_books = data.get("items", [])
            books.extend(new_books)
            next_start_index = len(books)
            if next_start_index < max_results and 'nextPageToken' in data:
                url = f"https://www.googleapis.com/books/v1/volumes?q={query}&maxResults=40&pageToken={data['nextPageToken']}"
            else:
                break
        else:
            print(f"Error fetching data: {response.status_code}")
            break
    result = []
    for book in books[:max_results]:
        volume_info = book.get("volumeInfo", {})
        title = volume_info.get("title", "Unknown")
        authors = ", ".join(volume_info.get("authors", ["Unknown"]))
        publisher = volume_info.get("publisher", "Unknown")
        published_date = volume_info.get("publishedDate", "0")[:4]
        result.append({
            "judul": title,
            "penulis": authors,
            "penerbit": publisher,
            "tahunTerbit": published_date if published_date.isdigit() else 0,
            "stok": 10
        })
    return result
```

### 3.4.4 Insert Kategori Baru

```
def insert_category_if_not_exists(category_name):
    cursor.execute("SELECT id FROM kategori WHERE nama = %s", (category_name,))
    category = cursor.fetchone()
    if not category:
        cursor.execute("INSERT INTO kategori (nama) VALUES (%s) RETURNING id", (category_name,))
        conn.commit()
        category_id = cursor.fetchone()[0]
        print(f"Kategori '{category_name}' berhasil ditambahkan!")
    else:
        category_id = category[0]
        print(f"Kategori '{category_name}' sudah ada.")
    return category_id
```

[4] ✓ 0.0s



### 3.4.5 Insert data buku

```
def insert_books_to_db(books, query):
    try:
        kategori_id = insert_category_if_not_exists(query)
        for book in books:
            try:
                cursor.execute("""
                    INSERT INTO buku (judul, penulis, penerbit, tahunTerbit, stok, kategori_id)
                    VALUES (%s, %s, %s, %s, %s, %s)
                    ON CONFLICT (judul) DO NOTHING;
                """, (
                    book["judul"],
                    book["penulis"],
                    book["penerbit"],
                    int(book["tahunTerbit"]) if book["tahunTerbit"].isdigit() else 0,
                    book["stok"],
                    kategori_id
                ))
            except Exception as e:
                print(f"Error inserting book: {book['judul']}, Error: {e}")
                conn.rollback()
        conn.commit()
        print("Data berhasil dimasukkan ke database!")
    except Exception as e:
        print(f"Error in insert_books_to_db: {e}")
        conn.rollback()
```

[5] ✓ 0.0s

### 3.4.6 Eksport data ke file csv

```
def export_to_csv(filename, query, columns):
    cursor.execute(query)
    rows = cursor.fetchall()
    with open(filename, mode='w', newline='', encoding='utf-8') as file:
        writer = csv.writer(file)
        writer.writerow(columns)
        writer.writerows(rows)
    print(f"Data berhasil disimpan ke file {filename}!")
```

[6] ✓ 0.0s

### 3.4.7 Mengambil data buku untuk setiap kategori

```
queries = [
    "art", "biography", "business", "children", "computers", "cooking", "design",
    "history", "health", "horror", "literature", "music", "mystery", "philosophy",
    "poetry", "psychology", "religion", "science", "self-help", "spirituality",
    "sports", "technology", "travel", "true-crime", "fiction", "non-fiction",
    "romance", "young-adult", "science-fiction", "fantasy"
]

for query in queries:
    books = fetch_books_from_api(query, max_results=100)
    insert_books_to_db(books, query)
```

✓ 33.5s

### 3.4.8 Export data kategori dan buku ke file .csv

```
export_to_csv(  
    filename="07_HasilCrawlingSistemPerpustakaanDigital_kategori.csv",  
    query="SELECT id, nama FROM kategori",  
    columns=["id", "nama"]  
)  
  
export_to_csv(  
    filename="07_HasilCrawlingSistemPerpustakaanDigital_buku.csv",  
    query="SELECT id, judul, penulis, penerbit, tahunTerbit, stok, kategori_id FROM buku",  
    columns=["id", "judul", "penulis", "penerbit", "tahunTerbit", "stok", "kategori_id"]  
)  
[9] ✓ 0.0s  
... Data berhasil disimpan ke file 07_HasilCrawlingSistemPerpustakaanDigital_kategori.csv!  
Data berhasil disimpan ke file 07_HasilCrawlingSistemPerpustakaanDigital_buku.csv!
```

### 3.4.9 Tutup Koneksi

```
cursor.close()  
conn.close()  
print("Koneksi ke database telah ditutup.")  
[10] ✓ 0.0s  
... Koneksi ke database telah ditutup.
```

## 3.5 Create Trigger Insert Data

Membuat trigger untuk memvalidasi data yang akan ditambahkan ke dalam table kategori

### 3.5.1 Tabel kategori

Tabel kategori berfungsi untuk menyimpan informasi mengenai kategori buku yang tersedia di perpustakaan. sehingga memudahkan anngota mengakses buku berdasarkan jenis atau topiknya,

#### 3.5.1.1 Create Function

Query ini berfungsi untuk memvalidasi data yang akan dimasukkan ke dalam table kategori yang dimana fungsi ini akan akan dipanggil oleh trigger sebelum data baru disisipkan ke dalam tabel. dan memeriksa apakah kolom nama pada data yang akan dimasukan (New=nama) kosong maka sistem akan menampilkan pesan pemberitahuan "Nama kategori tidak boleh kosong" dan membatalkan proses insert dengan mengembalikan. Selanjutnya fungsi memeriksa apakah nama kategori yang sama sudah ada di dalam table kategori Jika ditemukan kategori dengan nama yang sama, fungsi akan menampilkan pesan "Kategori dengan nama [nama kategori] sudah ada" dan kembali membatalkan proses insert. Jika kedua kondisi di atas tidak ditemukan, fungsi akan mengembalikan yang berarti data kategori yang baru dapat disimpan ke dalam tabel

- Berikut merupakan syntaxnya :

```
-- 3.5.1.1 Create Function
CREATE OR REPLACE FUNCTION cek_kategori() RETURNS TRIGGER AS $$
BEGIN
    IF NEW.nama = '' THEN
        RAISE NOTICE 'Nama kategori tidak boleh kosong';
        RETURN NULL;
    END IF;

    IF EXISTS (SELECT 1 FROM kategori WHERE nama = NEW.nama) THEN
        RAISE NOTICE 'Kategori dengan nama % sudah ada', NEW.nama;
        RETURN NULL;
    END IF;

    RETURN NEW;
END;
$$ LANGUAGE plpgsql;
```

- Output yang dihasilkan:

Data Output	Messages	Notifications
CREATE FUNCTION		
Query returned successfully in 141 msec.		

### 3.5.1.2 Create Trigger

Query ini digunakan untuk membuat trigger yang akan dijalankan sebelum data di insert ke dalam table kategori , trigger ini akan memanggil fungsi cek kategori untuk memeriksa apakah nama kategori yang akan dimasukkan valid atau tidak. Jika nama kategori kosong atau sudah ada dalam tabel, fungsi akan membatalkan proses insert dan memberikan pemberitahuan.

- Berikut merupakan syntaxnya :

```
-- 3.5.1.2 Create Trigger
CREATE TRIGGER before_insert_kategori
BEFORE INSERT ON kategori
FOR EACH ROW EXECUTE FUNCTION cek_kategori();
```

- Output yang dihasilkan:

Data Output	Messages	Notifications
CREATE TRIGGER		
Query returned successfully in 173 msec.		

### 3.5.1.3 Pengujian

Pengujian I :

Query ini memasukkan data dengan nama kategori kosong ke dalam kategori sehingga dengan adanya fungsi trigger yang akan memeriksa apakah nama kategori kosong, maka query ini akan gagal. Sistem akan memberikan pemberitahuan "Nama kategori tidak boleh kosong" dan data tidak akan disisipkan ke dalam tabel

- Berikut merupakan syntaxnya :

```
-- 3.5.1.3 Pengujian
-- Insert kategori kosong(gagal)
INSERT INTO kategori (nama) VALUES ('');
```

- **Output yang dihasilkan:**

Data Output	Messages	Notifications
NOTICE: Nama kategori tidak boleh kosong		
INSERT 0 0		
Query returned successfully in 126 msec.		

Pengujian II :

Query ini dapat memasukkan kategori dengan nama 'Teknologi', yang diasumsikan sudah ada dalam table kategori Fungsi trigger akan memeriksa apakah kategori dengan nama yang sama sudah ada. Jika ada, maka query ini akan gagal, dan sistem akan memberi pemberitahuan "Kategori dengan nama Teknologi sudah ada". Data tidak akan disisipkan

- **Berikut merupakan syntaxnya :**

```
-- Insert kategori yang sudah ada (gagal)
INSERT INTO kategori (nama) VALUES ('Teknologi');
```

- **Output yang dihasilkan:**

Data Output	Messages	Notifications
NOTICE: Kategori dengan nama Teknologi sudah ada		
INSERT 0 0		
Query returned successfully in 152 msec.		

Pengujian III :

Query ini memasukkan kategori dengan nama 'Komedi'. Karena nama kategori ini belum ada sebelumnya dan tidak kosong, query ini akan berhasil. Data kategori baru akan disisipkan ke dalam tabel tanpa masalah

- **Berikut merupakan syntaxnya :**

```
-- Insert kategori (berhasil)
INSERT INTO kategori (nama) VALUES ('Komedi');
-- Expected output: Data berhasil disisipkan tanpa peringatan
```

- **Output yang dihasilkan:**

Data Output	Messages	Notifications
INSERT 0 1		
Query returned successfully in 111 msec.		

### 3.5.2 Tabel Anggota

Pada anggota memiliki hak akses terbatas dalam system ini karena disini anggota hanya berfungsi untuk melakukan log peminjaman untuk meminjam buku dan melakukan pembayaran denda apabila terjadi keterlambatan saat pengembalian buku

### 3.5.2.1 Create Function

Query ini digunakan untuk memvalidasi data anggota sebelum dimasukkan ke dalam tabel. Fungsi ini pertama-tama memeriksa apakah nomor telepon yang dimasukkan valid, yaitu terdiri dari 10 hingga 15 digit angka. Jika nomor telepon tidak valid, fungsi akan membatalkan penyisipan dan memberikan pemberitahuan. Selain itu, fungsi juga memastikan bahwa tidak ada duplikasi nama anggota dengan memeriksa apakah nama yang sama sudah ada dalam tabel. Jika ada duplikasi, penyisipan juga dibatalkan

- Berikut merupakan syntaxnya :

```
-- 3.5.2.1 Create Function
CREATE OR REPLACE FUNCTION cek_anggota() RETURNS TRIGGER AS $$
BEGIN

    IF NEW.telepon IS NOT NULL AND NEW.telepon !~ '^d{10,15}$' THEN
        RAISE NOTICE 'Nomor telepon % tidak valid', NEW.telepon;
        RETURN NULL;
    END IF;

    IF EXISTS (SELECT 1 FROM anggota WHERE nama = NEW.nama) THEN
        RAISE NOTICE 'Anggota dengan nama % sudah ada', NEW.nama;
        RETURN NULL;
    END IF;

    RETURN NEW;
END;
$$ LANGUAGE plpgsql;
```

- Output yang dihasilkan:

Data Output Messages Notifications

CREATE FUNCTION

Query returned successfully in 66 msec.

### 3.5.2.2 Create Trigger

Query ini dibuat untuk menjalankan fungsi cek\_anggota sebelum data dimasukkan ke table anggota . Fungsi ini akan memeriksa apakah nomor telepon valid dan apakah nama anggota sudah ada di dalam sistem. Jika ada masalah, seperti nomor telepon tidak valid atau nama sudah terdaftar, data tidak akan dimasukkan dan akan muncul pemberitahuan

- Berikut merupakan syntaxnya :

```
-- 3.5.2.2 Create Trigger
CREATE TRIGGER before_insert_anggota
BEFORE INSERT ON anggota
FOR EACH ROW EXECUTE FUNCTION cek_anggota();
```

- Output yang dihasilkan:

Data Output Messages Notifications

CREATE TRIGGER

Query returned successfully in 83 msec.

### 3.5.2.3 Pengujian

Pengujian I :

Pada pengujian ini dapat memasukkan data anggota dengan nomor telepon yang kurang dari 10 digit, yang tidak memenuhi format yang valid sesuai dengan aturan yang ditetapkan dalam fungsi. Karena nomor telepon tersebut tidak valid, trigger akan memblokir proses insert dan menampilkan pesan pemberitahuan bahwa nomor telepon tidak valid. Dengan demikian, data tidak akan dimasukkan ke dalam tabel.

- **Anggota Cek\_anggota.** Berikut merupakan syntaxnya :

```
-- Insert nomor telpon < 10 (Gagal)
INSERT INTO anggota (nama, alamat, telepon) VALUES ('John Doe', ('Jalan Raya Samosir No. 15',
|'Parapat', 'Sumatera Utara', '22392', 'Indonesia'), '12345');
```

- **Output yang dihasilkan:**

Data Output	Messages	Notifications
NOTICE: Nomor telepon 12345 tidak valid		
INSERT 0 0		
Query returned successfully in 66 msec.		

Pengujian II :

Pada pengujian ini, dilakukan upaya untuk menambahkan data anggota dengan nama **Febiola Alya Hutagalung** yang sudah ada dalam table anggota. Fungsi cek\_anggota yang terkait dengan trigger akan memeriksa keberadaan nama tersebut di tabel sebelum proses insert dilakukan. Karena nama tersebut sudah ada, fungsi akan memblokir proses insert dan memunculkan pemberitahuan bahwa anggota dengan nama tersebut sudah ada. Dengan demikian, data baru tidak akan dimasukkan, dan tabel anggota tetap tidak mengalami perubahan.

- **Syntax :**

```
-- Insert nama yang sudah ada (Gagal)
INSERT INTO anggota (nama, email, password, alamat, telepon, tanggal_daftar)
VALUES
('Febiola Alya Hutagalung', 'febiola@gmail.com', 'febiola123',
('Jalan Raya Toba No. 1', 'Parapat', 'Sumatera Utara', '22381', 'Indonesia'), '081234567890', CURRENT_TIMESTAMP);
```

- **Output :**

Data Output	Messages	Notifications
NOTICE: Anggota dengan nama Febiola Alya Hutagalung sudah ada		
INSERT 0 0		
Query returned successfully in 78 msec.		

Pengujian III :

Pada pengujian ini, dilakukan penambahan data anggota baru dengan nama **James Tambunan**. Karena data yang dimasukkan valid—baik nama, email, password, alamat, nomor telepon, maupun tanggal pendaftaran—fungsi cek\_anggota tidak menemukan pelanggaran aturan. Nama anggota tidak duplikat, dan nomor telepon memenuhi format yang ditentukan. Oleh karena itu, proses insert berhasil dilakukan, dan data anggota baru ditambahkan ke table anggota tanpa masalah.

- **Syntax** :

```
-- Insert anggota berhasil
INSERT INTO anggota (nama, email, password, alamat, telepon, tanggal_daftar)
VALUES
('James Tambunan', 'james@gmail.com', 'james123',
('Jalan Bunga No. 1', 'Parapat', 'Sumatera Utara', '22381', 'Indonesia'), '086521541528', CURRENT_TIMESTAMP);
```

- **Output** :

Data Output	Messages	Notifications
INSERT 0 1		
Query returned successfully in 89 msec.		

Lakukan pengecekan data :

- **Syntax** :

```
-- Verifikasi tabel anggota
SELECT * FROM anggota;
```

- **Output** :

Data Output			Messages	Notifications		
<div><div><div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div></div><div>SQL</div></div></div>						
id	nama	email	password	alamat	telepon	tanggal_daftar
[PK] integer	character varying (255)	character varying (255)	character varying (255)	alamat_type	character varying (15)	timestamp without time zone
6	Estina Pangaribuan	estina.pangaribuan@gmail.com	estina123	("Jalan Raya Samsosir No. 2','Tuktuk','Sumatera Utara','22386,Indonesia)	081234567895	2024-12-10 17:30:38.536591
7	Valencia L Tobing	valencia.tobing@gmail.com	valencia123	("Jalan Parapat No. 7','Parapat','Sumatera Utara','22387,Indonesia)	081234567896	2024-12-10 17:30:38.536591
8	Andri Agung Exaudi Sigi	andri.sigi@gmail.com	andri123	("Jalan Danau Toba No. 8','Balige','Sumatera Utara','22388,Indonesia)	081234567897	2024-12-10 17:30:38.536591
9	Ayu Enissa Marety Sinaga	ayu.sinaga@gmail.com	ayu123	("Jalan Raja Sisingamangaraja No. 10','Ajibata','Sumatera Utara','22389,Indonesia)	081234567898	2024-12-10 17:30:38.536591
10	Indra Aziz Nugraha	indra.nugraha@gmail.com	indra123	("Jalan Tuktuk No. 6','Samosir','Sumatera Utara','22390,Indonesia)	081234567899	2024-12-10 17:30:38.536591
11	Daniel Haganta Ginting	daniel.ginting@gmail.com	daniel123	("Jalan Merdeka No. 11','Toba Samsosir','Sumatera Utara','22391,Indonesia)	081234567900	2024-12-10 17:30:38.536591
12	Diva Lorenza Marbun	diva.marbun@gmail.com	diva123	("Jalan Raya Samsosir No. 15','Parapat','Sumatera Utara','22392,Indonesia)	081234567901	2024-12-10 17:30:38.536591
13	James Tambunan	james@gmail.com	james123	("Jalan Bunga No. 1','Parapat','Sumatera Utara','22381,Indonesia)	086521541528	2024-12-10 23:08:13.675883
Total rows: 13 of 13    Query complete 00:00:00.093    Ln 298, Col 5						

### 3.5.3 Tabel Buku

Pada table ini digunakan untuk membuat table buku yang menyimpan informasi mengenai buku-buku dalam sistem perpustakaan, termasuk data seperti judul, penulis, penerbit, tahun terbit, stok, dan kategori.

#### 3.5.3.1 Create Function

Query ini digunakan untuk memvalidasi data sebelum disisipkan ke table buku. Fungsi ini memastikan bahwa data yang dimasukkan memenuhi kriteria tertentu. Pertama, judul, penulis, dan penerbit tidak boleh kosong atau bernilai NULL. Selanjutnya, stok buku harus bernilai positif dan tidak boleh 0, untuk memastikan buku yang tersedia memiliki jumlah yang valid. Fungsi ini juga memeriksa apakah ada buku dengan kombinasi judul dan penulis yang sama untuk mencegah duplikasi data. Jika semua kondisi terpenuhi, data baru akan diterima; jika tidak, fungsi akan memberikan notifikasi kesalahan dan membatalkan proses insert. Fungsi ini menjaga konsistensi dan integritas data dalam sistem perpustakaan.

- **Syntax :**

```
-- 3.5.3.1 Create Function
CREATE OR REPLACE FUNCTION cek_buku() RETURNS TRIGGER AS $$
BEGIN

    IF NEW.judul IS NULL OR NEW.judul = '' THEN
        RAISE NOTICE 'Judul buku tidak boleh kosong';
        RETURN NULL;
    END IF;

    IF NEW.penulis IS NULL OR NEW.penulis = '' THEN
        RAISE NOTICE 'Penulis buku tidak boleh kosong';
        RETURN NULL;
    END IF;

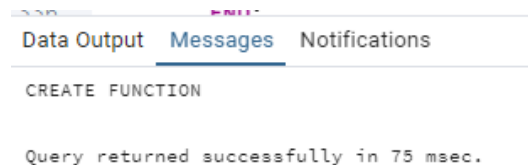
    IF NEW.penerbit IS NULL OR NEW.penerbit = '' THEN
        RAISE NOTICE 'Penerbit buku tidak boleh kosong';
        RETURN NULL;
    END IF;

    IF NEW.stok < 1 THEN
        RAISE NOTICE 'Stok buku harus bernilai positif dan tidak 0!';
        RETURN NULL;
    END IF;

    IF EXISTS (
        SELECT 1
        FROM buku
        WHERE judul = NEW.judul AND penulis = NEW.penulis
    ) THEN
        RAISE NOTICE 'Buku dengan judul % dan penulis % sudah ada', NEW.judul, NEW.penulis;
        RETURN NULL;
    END IF;

    RETURN NEW;
END;
$$ LANGUAGE plpgsql;
```

- **Output :**



The screenshot shows a SQL IDE interface with tabs for 'Data Output', 'Messages', and 'Notifications'. The 'Messages' tab is selected, displaying the text 'CREATE FUNCTION' and 'Query returned successfully in 75 msec.'

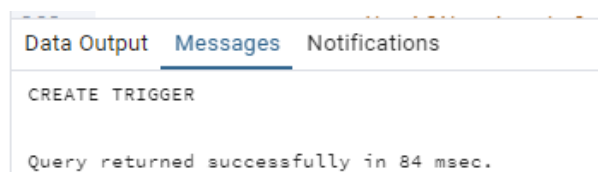
### 3.5.3.2 Create Trigger

Pada query ini digunakan untuk table buku agar validasi data dilakukan secara otomatis setiap kali ada upaya memasukkan data baru. Trigger ini memanfaatkan fungsi cek\_buku() untuk memastikan data memenuhi aturan yang telah ditentukan. Dengan deklarasi BEFORE INSERT, trigger dijalankan sebelum proses INSERT dilakukan. Setiap baris data baru akan diperiksa melalui fungsi tersebut, sehingga hanya data valid yang dapat dimasukkan ke tabel buk

- **Syntax :**

```
-- 3.5.3.2 Create Trigger
CREATE TRIGGER before_insert_buku
BEFORE INSERT ON buku
FOR EACH ROW EXECUTE FUNCTION cek_buku();
```

- **Output :**



The screenshot shows a SQL IDE interface with tabs for 'Data Output', 'Messages', and 'Notifications'. The 'Messages' tab is selected, displaying the text 'CREATE TRIGGER' and 'Query returned successfully in 84 msec.'



### 3.5.3.3 Pengujian

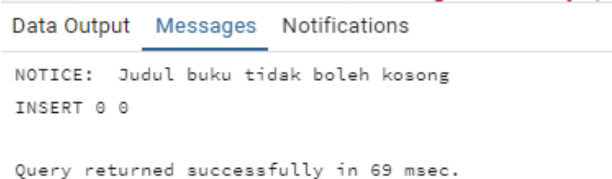
Pengujian I :

Pengujian ini dilakukan dengan mencoba menyisipkan data ke tabel buku menggunakan nilai judul kosong ("). Berdasarkan logika dalam fungsi `cek_buku`, jika kolom judul tidak diisi atau bernilai kosong, maka sistem akan menampilkan pesan pemberitahuan **"Judul buku tidak boleh kosong"** dan proses penyisipan data akan dihentikan. Hal ini menunjukkan bahwa mekanisme validasi untuk mencegah data tidak valid berfungsi dengan baik, sehingga integritas data pada tabel buku tetap terjaga.

- **Syntax** :

```
-- Insert buku dengan judul kosong (Gagal)
INSERT INTO buku (judul, penulis, penerbit, tahunTerbit, stok, kategori_id)
VALUES ('', 'John Doe', 'Publisher A', 2023, 5, 1);
```

- **Output** :



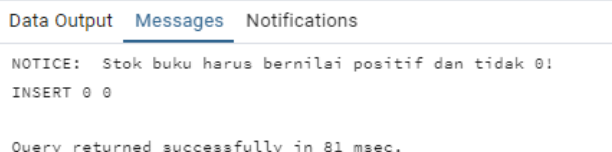
The screenshot shows a SQL client interface with tabs for 'Data Output', 'Messages', and 'Notifications'. The 'Messages' tab is active, displaying a red error message: 'NOTICE: Judul buku tidak boleh kosong'. Below the message, it shows 'INSERT 0 0' and 'Query returned successfully in 69 msec.'

Pengujian II :

- **Syntax** :

```
-- Insert Stok negatif (Gagal)
INSERT INTO buku (judul, penulis, penerbit, tahunTerbit, stok, kategori_id)
VALUES ('Book A', 'John Doe', 'Publisher A', 2023, -3, 1);
```

- **Output** :



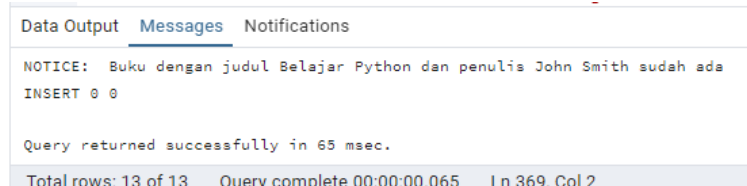
The screenshot shows a SQL client interface with tabs for 'Data Output', 'Messages', and 'Notifications'. The 'Messages' tab is active, displaying a red error message: 'NOTICE: Stok buku harus bernilai positif dan tidak 0!'. Below the message, it shows 'INSERT 0 0' and 'Query returned successfully in 81 msec.'

Pengujian III :

- **Syntax** :

```
-- Insert buku dengan judul dan penulis yang sama
INSERT INTO buku (judul, penulis, penerbit, tahunTerbit, stok, kategori_id)
VALUES ('Belajar Python', 'John Smith', 'Tech Publisher', 2023, 10, 1);
```

- **Output** :



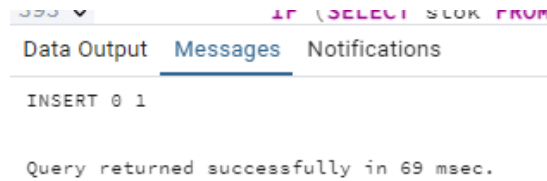
The screenshot shows a SQL client interface with tabs for 'Data Output', 'Messages', and 'Notifications'. The 'Messages' tab is active, displaying a red error message: 'NOTICE: Buku dengan judul Belajar Python dan penulis John Smith sudah ada'. Below the message, it shows 'INSERT 0 0' and 'Query returned successfully in 65 msec.'. At the bottom, a status bar indicates 'Total rows: 13 of 13', 'Query complete 00:00:00.065', and 'Ln 369, Col 2'.

Pengujian IV :

- **Syntax** :

```
-- Insert data berhasil
INSERT INTO buku (judul, penulis, penerbit, tahunTerbit, stok, kategori_id)
VALUES ('Pemrograman Ruby', 'Yukihiro Matsumoto', 'Tech Publisher', 2023, 8, 1);
```

- **Output** :



```
IF (SELECT STOK FROM
Data Output Messages Notifications
INSERT 0 1
Query returned successfully in 69 msec.
```

Lakukan pengecekan data :

- **Syntax** :

```
-- Verifikasi tabel buku
SELECT * FROM buku;
```

- **Output** :

id [PK] integer	judul character varying (255)	penulis character varying (255)	penerbit character varying (255)	tahunTerbit integer	stok integer	kategori_id integer
1055	1193 Fate - The Books of Fantasy	Andarani Putri	KPDF Publisher	0	10	30
1056	1194 The Economics of Fantasy	Sharon Stockton	Ohio State University Press	2006	10	30
1057	1195 My Fantasy Story	Adila Bilqys	SPASI MEDIA	0	10	30
1058	1801 Kecerdasan Emosional	Daniel Goleman	Gramedia Pustaka Utama	2024	10	16
1059	2396 Pemrograman Ruby	Yukihiro Matsumoto	Tech Publisher	2023	8	1

Total rows: 1059 of 1059 Query complete 00:00:00.107 Ln 381, Col 2

## 3.6 Create Proses Peminjaman Buku menggunakan Trigger dan Fungsi

### 3.6.1 Create Function

Proses peminjaman buku dalam sistem perpustakaan ini melibatkan penggunaan fungsi dan trigger untuk memastikan bahwa peminjaman hanya dilakukan jika semua kondisi valid. Fungsi `lakukan_peminjaman()` pertama-tama memvalidasi keberadaan anggota dan buku yang akan dipinjam dengan memeriksa ID anggota dan ID buku di database. Jika ID anggota atau ID buku tidak ditemukan, maka fungsi akan menampilkan pesan kesalahan yang sesuai. Selain itu, fungsi ini juga memeriksa apakah stok buku mencukupi untuk peminjaman. Jika stok buku kurang dari atau sama dengan 0, peminjaman dibatalkan dan muncul pesan bahwa stok tidak mencukupi. Jika semua syarat tersebut terpenuhi, fungsi akan mengurangi stok buku sebanyak satu dan menetapkan status peminjaman buku serta tanggal pengembalian yang ditentukan tujuh hari setelah peminjaman. Trigger `peminjaman_trigger` kemudian dipasang untuk memastikan fungsi ini dijalankan setiap kali ada upaya untuk memasukkan data peminjaman baru ke dalam tabel `log_peminjaman`. Pengujian dilakukan dalam beberapa skenario, antara lain peminjaman dengan ID anggota yang tidak terdaftar, peminjaman dengan buku yang tidak tersedia, serta peminjaman dengan stok buku yang tidak mencukupi. Semua pengujian yang gagal menghasilkan pesan kesalahan yang sesuai, sementara peminjaman yang berhasil tercatat dengan benar dalam `log_peminjaman`. Sistem ini memastikan bahwa proses peminjaman berjalan sesuai dengan aturan dan menjaga akurasi data di dalam sistem perpustakaan.

- **Syntax** :

```
-- 3.6.1 Create Function
CREATE OR REPLACE FUNCTION lakukan_peminjaman() RETURNS TRIGGER AS $$
BEGIN
    IF NOT EXISTS (SELECT 1 FROM anggota WHERE id = NEW.anggota_id) THEN
        RAISE EXCEPTION 'Data anggota dengan ID % tidak ditemukan', NEW.anggota_id;
    END IF;

    IF NOT EXISTS (SELECT 1 FROM buku WHERE id = NEW.buku_id) THEN
        RAISE EXCEPTION 'Data buku dengan ID % tidak ditemukan', NEW.buku_id;
    END IF;

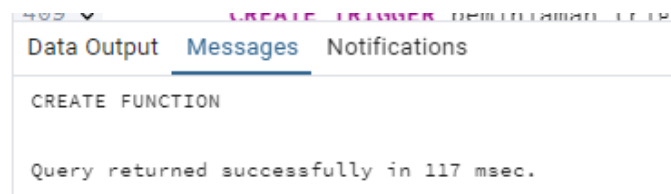
    IF (SELECT stok FROM buku WHERE id = NEW.buku_id) <= 0 THEN
        RAISE EXCEPTION 'Stok buku tidak mencukupi untuk peminjaman';
    END IF;

    UPDATE buku
    SET stok = stok - 1
    WHERE id = NEW.buku_id;

    NEW.status := 'Dipinjam';
    NEW.tanggal_pengembalian := NEW.tanggal_peminjaman + INTERVAL '7 days';

    RETURN NEW;
END;
$$ LANGUAGE plpgsql;
```

- Output :



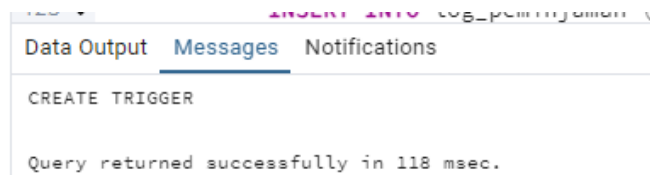
The screenshot shows a database interface with tabs for 'Data Output', 'Messages', and 'Notifications'. The 'Messages' tab is selected, displaying the text 'CREATE FUNCTION' and 'Query returned successfully in 117 msec.'

### 3.6.2 Create Trigger

- Syntax :

```
-- 3.6.2 Create Trigger
CREATE TRIGGER peminjaman_trigger
BEFORE INSERT ON log_peminjaman
FOR EACH ROW
EXECUTE FUNCTION lakukan_peminjaman();
```

- Output :



The screenshot shows a database interface with tabs for 'Data Output', 'Messages', and 'Notifications'. The 'Messages' tab is selected, displaying the text 'CREATE TRIGGER' and 'Query returned successfully in 118 msec.'

### 3.6.3 Pengujian

Pengujian I :

- Syntax :

```
-- Lakukan peminjaman dengan ID anggota yang tidak tersedia (Gagal)
INSERT INTO log_peminjaman (anggota_id, buku_id, tanggal_peminjaman)
VALUES (999, 1, CURRENT_DATE);
```

- Output :

Data Output	Messages	Notifications
ERROR: Data anggota dengan ID 999 tidak ditemukan		
CONTEXT: PL/pgSQL function lakukan_peminjaman() line 4 at RAISE		
SQL state: P0001		

Pengujian II :

- **Syntax** :

```
-- Lakukan peminjaman dengan ID buku yang tidak tersedia (Gagal)
INSERT INTO log_peminjaman (anggota_id, buku_id, tanggal_peminjaman)
VALUES (1, 500000, CURRENT_DATE);
```

- **Output** :

Data Output	Messages	Notifications
ERROR: Data buku dengan ID 500000 tidak ditemukan		
CONTEXT: PL/pgSQL function lakukan_peminjaman() line 8 at RAISE		
SQL state: P0001		

Pengujian III :

- **Syntax** :

```
-- Lakukan peminjaman dengan Stok Buku Tidak Cukup (Gagal)
UPDATE buku SET stok = 0 WHERE id = 1;
INSERT INTO log_peminjaman (anggota_id, buku_id, tanggal_peminjaman)
VALUES (1, 1, CURRENT_DATE);
```

- **Output** :

Data Output	Messages	Notifications
ERROR: Stok buku tidak mencukupi untuk peminjaman		
CONTEXT: PL/pgSQL function lakukan_peminjaman() line 12 at RAISE		
SQL state: P0001		

Pengujian IV :

- **Syntax** :

```
-- Lakukan peminjaman berhasil
INSERT INTO log_peminjaman (anggota_id, buku_id, tanggal_peminjaman)
VALUES (3, 2, CURRENT_DATE);
```

- **Output** :

Data Output	Messages	Notifications
INSERT 0 1		
Query returned successfully in 63 msec.		

Lakukan pengecekan data nya

- **Syntax** :

```
-- Memeriksa data dalam tabel
SELECT * FROM log_peminjaman;
```

- **Output** :

	id_peminjaman [PK] integer	anggota_id integer	buku_id integer	tanggal_peminjaman date	tanggal_pengembalian date	status character varying (50)	denda numeric (10,2)
1	2	1	999	2024-12-10	2024-12-17	Dipinjam	0.00
2	3	1	1061	2024-12-10	2024-12-17	Dipinjam	0.00
3	6	3	2	2024-12-10	2024-12-17	Dipinjam	0.00

## 3.7 Create Proses Pengembalian Buku menggunakan Transaksi Manual

### 3.7.1 Create Transaction

Fungsi `proses_pengembalian_transaksi` bertujuan untuk memproses pengembalian buku yang dipinjam oleh anggota, dengan mempertimbangkan apakah pengembalian dilakukan tepat waktu atau terlambat. Fungsi ini menerima dua parameter: `p_id_peminjaman` (ID peminjaman) dan `p_tanggal_pengembalian` (tanggal pengembalian buku). Pertama, fungsi memeriksa status peminjaman pada tabel `log_peminjaman`. Jika peminjaman sudah dikembalikan atau terlambat, fungsi akan mengeluarkan pemberitahuan dan tidak melanjutkan proses pengembalian. Selanjutnya, fungsi menghitung batas waktu pengembalian (yaitu 7 hari setelah tanggal peminjaman) dan menghitung berapa lama keterlambatan, jika ada. Jika pengembalian dilakukan tepat waktu, status peminjaman diperbarui menjadi "Kembali" dan tanggal pengembalian diset. Jika pengembalian terlambat, status diperbarui menjadi "Terlambat" dan denda dihitung berdasarkan jumlah hari keterlambatan, kemudian dimasukkan ke dalam tabel `log_denda`. Setelah itu, jumlah stok buku pada tabel `buku` akan ditambahkan satu, menandakan bahwa buku telah kembali. Fungsi ini diujikan dengan tiga kondisi: pengembalian lebih awal, tepat waktu, dan terlambat, untuk memeriksa apakah pengembalian diproses dengan benar.

#### - Syntax :

```
-- 3.7.1 Create Transaction
CREATE OR REPLACE FUNCTION proses_pengembalian_transaksi(
    p_id_peminjaman INT,
    p_tanggal_pengembalian DATE
) RETURNS VOID AS $$
DECLARE
    batas_pengembalian DATE;
    lama_terlambat INT;
    jumlah_denda NUMERIC(10, 2);
    current_status TEXT;
BEGIN
    SELECT status INTO current_status
    FROM log_peminjaman
    WHERE id_peminjaman = p_id_peminjaman;

    IF NOT FOUND THEN
        RAISE EXCEPTION 'Peminjaman dengan ID % tidak ditemukan', p_id_peminjaman;
    END IF;

    IF current_status = 'Kembali' OR current_status = 'Terlambat' THEN
        RAISE NOTICE 'Pengembalian tidak dapat diproses. Status saat ini: %.', current_status;
        RETURN;
    END IF;

    SELECT tanggal_peminjaman + INTERVAL '7 days' INTO batas_pengembalian
    FROM log_peminjaman WHERE id_peminjaman = p_id_peminjaman;

    lama_terlambat := GREATEST((p_tanggal_pengembalian - batas_pengembalian)::INTEGER, 0);

    IF p_tanggal_pengembalian <= batas_pengembalian THEN
        UPDATE log_peminjaman
        SET status = 'Kembali', tanggal_pengembalian = p_tanggal_pengembalian
        WHERE id_peminjaman = p_id_peminjaman;
```

**- Output :**

### 3.7.2 Pengujian

- **Syntax** :

- **Output :**

## Pengujian II :

- **Syntax** :

- **Output :**

Data Output

Messages

Notifications

Pengujian III :

- **Syntax** :

```
-- Lakukan pengembalian terlambat
select proses_pengembalian_transaksi(3, '2024-12-18');
```

- **Output** :

Data Output Messages Notifications

NOTICE: Pengembalian berhasil diproses. Status: Terlambat, Denda: 2000.00

Successfully run. Total query runtime: 112 msec.

1 rows affected.

Lakukan pengecekan terhadap data :

- **Syntax** :

```
-- Periksa isi data dalam tabel
SELECT * FROM log_peminjaman;
```

- **Output** :

Data Output Messages Notifications

	id_peminjaman [PK] integer	anggota_id integer	buku_id integer	tanggal_peminjaman date	tanggal_pengembalian date	status character varying (50)	denda numeric (10,2)
1		2	1	999 2024-12-10	2024-12-17	Dipinjam	0.00
2		3	1	1061 2024-12-10	2024-12-17	Dipinjam	0.00
3		6	3	2 2024-12-10	2024-12-13	Kembali	0.00

### 3.8 Proses Pembayaran Denda menggunakan Transaksi Manual

Fungsi proses\_pembayaran\_denda bertujuan untuk memproses pembayaran denda berdasarkan ID yang diberikan. Pertama, fungsi memeriksa apakah denda dengan ID tersebut ada di dalam tabel log\_denda. Jika tidak ditemukan, fungsi akan menampilkan pesan kesalahan. Jika denda sudah dibayar sebelumnya, maka fungsi akan memberikan pemberitahuan bahwa denda tersebut sudah dibayar, dan proses akan dihentikan. Jika denda belum dibayar, status pembayaran akan diperbarui menjadi "Sudah Dibayar", dan pemberitahuan akan ditampilkan bahwa pembayaran denda berhasil. Pengujian dilakukan untuk dua kondisi: pertama dengan ID denda yang tidak ada, yang menyebabkan kegagalan, dan kedua dengan ID denda yang valid, yang berhasil memperbarui status pembayaran.

#### 3.8.1 Create transaction

- **Syntax** :

```
-- 3.8.1 Create transaction --
CREATE OR REPLACE FUNCTION proses_pembayaran_denda(
    p_id_denda INT
) RETURNS VOID AS $$
DECLARE
    current_status_pembayaran VARCHAR(20);
BEGIN
    SELECT status_pembayaran INTO current_status_pembayaran
    FROM log_denda
    WHERE id_denda = p_id_denda;

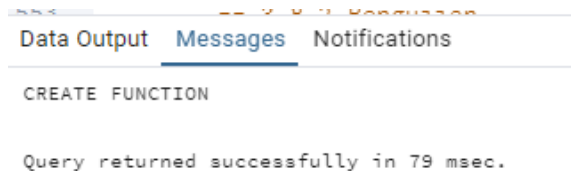
    IF NOT FOUND THEN
        RAISE EXCEPTION 'Denda dengan ID % tidak ditemukan', p_id_denda;
    END IF;

    IF current_status_pembayaran = 'Sudah Dibayar' THEN
        RAISE NOTICE 'Denda dengan ID % sudah dibayar sebelumnya.', p_id_denda;
        RETURN;
    END IF;

    UPDATE log_denda
    SET status_pembayaran = 'Sudah Dibayar'
    WHERE id_denda = p_id_denda;

    RAISE NOTICE 'Pembayaran denda dengan ID % berhasil diproses. Status: Sudah Dibayar', p_id_denda;
END;
$$ LANGUAGE plpgsql;
```

- **Output** :



The screenshot shows a database interface with tabs for 'Data Output', 'Messages', and 'Notifications'. The 'Messages' tab is selected, displaying the text: 'CREATE FUNCTION' and 'Query returned successfully in 79 msec.'

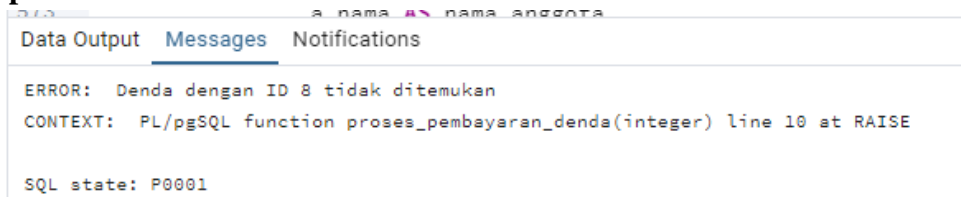
### 3.8.2 Pengujian

Pengujian I :

- **Syntax** :

```
-- Lakukan bayar denda dengan Id tidak ditemukan(Gagal)
SELECT proses_pembayaran_denda(8);
```

- **Output** :



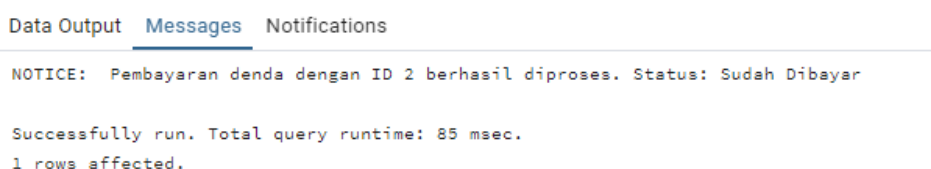
The screenshot shows a database interface with tabs for 'Data Output', 'Messages', and 'Notifications'. The 'Messages' tab is selected, displaying an error message: 'ERROR: Denda dengan ID 8 tidak ditemukan' and 'CONTEXT: PL/pgSQL function proses\_pembayaran\_denda(integer) line 10 at RAISE'. The SQL state is shown as 'P0001'.

Pengujian II :

- **Syntax** :

```
-- Lakukan bayar denda(Berhasil)
SELECT proses_pembayaran_denda(2);
```

- **output** :



The screenshot shows a database interface with tabs for 'Data Output', 'Messages', and 'Notifications'. The 'Messages' tab is selected, displaying a notice: 'NOTICE: Pembayaran denda dengan ID 2 berhasil diproses. Status: Sudah Dibayar'. Below the notice, it says 'Successfully run. Total query runtime: 85 msec.' and '1 rows affected.'



Lakukan pengecekan data :

- **Syntax** :

```
-- Memeriksa isi data dalam tabel
SELECT * FROM log_denda;
```

- **Output** :

	id_denda [PK] integer	peminjaman_id integer	jumlah_denda numeric (10,2)	tanggal_denda date	status_pembayaran character varying (20)
1	1	3	2000.00	2024-12-11	Belum Dibayar
2	2	2	4000.00	2024-12-11	Sudah Dibayar

## 3.9 Create Tabel Laporan

### 3.9.1 Tabel laporan jumlah denda peminjaman setiap anggota

Fungsi anggota\_peminjaman\_denda adalah sebuah **view** yang menampilkan laporan peminjaman dan denda setiap anggota. View ini menggabungkan data dari beberapa tabel: anggota, log\_peminjaman, buku, dan log\_denda. Data yang ditampilkan mencakup ID anggota, nama anggota, judul buku, tanggal peminjaman, tanggal pengembalian, status peminjaman, jumlah denda (jika ada), dan status pembayaran denda. Jika denda tidak ada, maka jumlah denda akan bernilai 0 dan status pembayaran akan bernilai "Belum Dibayar".

#### 3.9.1.1 Create View

- **Syntax** :

```
-- 3.9.1.1 Create View
CREATE VIEW anggota_peminjaman_denda AS
SELECT
    a.id AS anggota_id,
    a.nama AS nama_anggota,
    b.judul AS judul_buku,
    lp.tanggal_peminjaman,
    lp.tanggal_pengembalian,
    lp.status AS status_peminjaman,
    COALESCE(ld.jumlah_denda, 0.00) AS jumlah_denda,
    COALESCE(ld.status_pembayaran, 'Belum Dibayar') AS status_denda
FROM
    anggota a
JOIN
    log_peminjaman lp ON a.id = lp.anggota_id
JOIN
    buku b ON lp.buku_id = b.id
LEFT JOIN
    log_denda ld ON lp.id_peminjaman = ld.peminjaman_id;
```

- **Output** :

Data Output	Messages	Notifications
CREATE VIEW		
Query returned successfully in 94 msec.		

### 3.9.1.2 Menampilkan hasil tabel laporan buku

- Syntax :

```
-- 3.9.1.2 Menampilkan hasil tabel laporan
SELECT * FROM anggota_peminjaman_denda;
```

- Output :

	anggota_id integer	nama_anggota character varying (255)	judul_buku character varying (255)	tanggal_peminjaman date	tanggal_pengembalian date	status_peminjaman character varying (50)	jumlah_denda numeric	status_denda character varying
1	1	Febiola Alya Hutagalung	New Perspectives on Bare Noun Phrases in Romance and Beyo...	2024-12-10	2024-12-18	Terlambat	2000.00	Belum Dibayar
2	1	Febiola Alya Hutagalung	What Is Non-fiction Cinema?	2024-12-10	2024-12-19	Terlambat	4000.00	Sudah Dibayar
3	3	Aan Kristian Sitingak	Untuk apa seni?	2024-12-10	2024-12-13	Kembali	0.00	Belum Dibayar

### 3.9.2 Tabel laporan buku paling sering dipinjam per-tahun

Query ini membuat view bernama laporan\_buku\_per\_tahun, yang menghasilkan laporan mengenai buku yang paling sering dipinjam setiap tahunnya. Data yang ditampilkan meliputi judul, penulis, penerbit, tahun terbit, status peminjaman, dan jumlah peminjaman untuk setiap buku. Query ini menggabungkan data dari tabel buku dan log\_peminjaman dengan melakukan perhitungan jumlah peminjaman per buku, dikelompokkan berdasarkan tahun peminjaman. Hasilnya diurutkan berdasarkan jumlah peminjaman terbanyak, tahun, dan status peminjaman, untuk memudahkan analisis buku yang paling populer tiap tahun.

#### 3.9.2.1 Create View

- Syntax :

```
-- 3.9.2.1 Create View
CREATE VIEW laporan_buku_per_tahun AS
SELECT
    b.judul,
    b.penulis,
    b.penerbit,
    b.tahunTerbit,
    EXTRACT(YEAR FROM lp.tanggal_peminjaman) AS tahun,
    lp.status,
    COUNT(lp.id_peminjaman) AS jumlah_peminjaman
FROM
    buku b
JOIN
    log_peminjaman lp ON b.id = lp.buku_id
GROUP BY
    b.id, b.judul, b.penulis, b.penerbit, b.tahunTerbit, tahun, lp.status
ORDER BY
    jumlah_peminjaman DESC, tahun DESC, lp.status DESC;
```

- Output :

Data Output	Messages	Notifications
CREATE VIEW		
Query returned successfully in 5 secs 697 msec.		

#### 3.9.2.2 Menampilkan hasil tabel laporan

- Syntax :

```
-- 3.9.2.2 Menampilkan hasil tabel laporan
SELECT * FROM laporan_buku_per_tahun;
```

- **Output :**

	judul character varying (255)	penulis character varying (255)	penerbit character varying (255)	tahunTerbit integer	tahun numeric	status character varying (50)	jumlah_peminjaman bigint
1	New Perspectives on Bare Noun Phrases in Romance and Beyo...	Johannes Kabatek, Albert Wall	John Benjamins Publishing	2013	2024	Terlambat	1
2	What Is Non-fiction Cinema?	Trevor Ponech	Routledge	2021	2024	Terlambat	1
3	Untuk apa seni?	I. Bambang Sugiharto	Unknown	2015	2024	Kembali	1

### 3.9.3 Tabel laporan buku paling sering dipinjam per-bulan

Query ini ini membuat view bernama laporan\_buku\_per\_bulan, yang menghasilkan laporan mengenai buku yang paling sering dipinjam setiap bulan. Data yang ditampilkan meliputi judul, penulis, penerbit, tahun terbit, bulan peminjaman, status peminjaman, dan jumlah peminjaman per buku. Query ini menggabungkan data dari tabel buku dan log\_peminjaman dengan menghitung jumlah peminjaman setiap buku, dikelompokkan berdasarkan bulan peminjaman. Hasilnya diurutkan berdasarkan jumlah peminjaman terbanyak, bulan, dan status peminjaman untuk memudahkan analisis buku yang paling populer tiap bulan.

#### 3.9.3.1 Create View

- **Syntax :**

```
-- 3.9.3.1 Create View
CREATE VIEW laporan_buku_per_bulan AS
SELECT
    b.judul,
    b.penulis,
    b.penerbit,
    b.tahunTerbit,
    DATE_TRUNC('month', lp.tanggal_peminjaman) AS bulan,
    lp.status,
    COUNT(lp.id_peminjaman) AS jumlah_peminjaman
FROM
    buku b
JOIN
    log_peminjaman lp ON b.id = lp.buku_id
GROUP BY
    b.id, b.judul, b.penulis, b.penerbit, b.tahunTerbit, bulan, lp.status
ORDER BY
    jumlah_peminjaman DESC, bulan DESC, lp.status DESC;
```

- **Output :**

Data Output	Messages	Notifications
CREATE VIEW		
Query returned successfully in 105 msec.		

#### 3.9.3.2 Menampilkan hasil tabel laporan

- **Syntax :**

```
-- 3.9.3.2 Menampilkan hasil tabel laporan
SELECT * FROM laporan_buku_per_bulan;
```

- **Output :**

	judul character varying (255)	penulis character varying (255)	penerbit character varying (255)	tahunTerbit integer	bulan timestamp with time zone	status character varying (50)	jumlah_peminjaman bigint
1	New Perspectives on Bare Noun Phrases in Romance and Beyo...	Johannes Kabatek, Albert Wall	John Benjamins Publishing	2013	2024-12-01 00:00:00+07	Terlambat	1
2	What Is Non-fiction Cinema?	Trevor Ponech	Routledge	2021	2024-12-01 00:00:00+07	Terlambat	1
3	Untuk apa seni?	I. Bambang Sugiharto	Unknown	2015	2024-12-01 00:00:00+07	Kembali	1

### 3.9.4 Tabel laporan buku paling sering dipinjam per-minggu

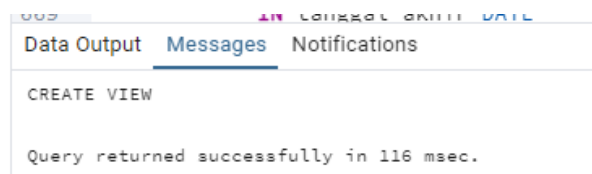
Query ini membuat view bernama laporan\_buku\_per\_minggu, yang menyajikan laporan buku yang paling sering dipinjam per minggu. Laporan ini mencakup informasi judul buku, penulis, penerbit, tahun terbit, minggu peminjaman, status peminjaman, serta jumlah peminjaman setiap buku. Dengan menggunakan fungsi DATE\_TRUNC('week', ...), tanggal peminjaman dipotong menjadi minggu, kemudian hasilnya dikelompokkan berdasarkan buku dan minggu peminjaman. Hasil laporan diurutkan berdasarkan jumlah peminjaman tertinggi, minggu peminjaman, dan status peminjaman, untuk mempermudah identifikasi tren peminjaman buku dalam periode mingguan.

#### 3.9.4.1 Create View

- Syntax :

```
-- 3.9.4.1 Create View
CREATE VIEW laporan_buku_per_minggu AS
SELECT
    b.judul,
    b.penulis,
    b.penerbit,
    b.tahunTerbit,
    DATE_TRUNC('week', lp.tanggal_peminjaman) AS minggu,
    lp.status,
    COUNT(lp.id_peminjaman) AS jumlah_peminjaman
FROM
    buku b
JOIN
    log_peminjaman lp ON b.id = lp.buku_id
GROUP BY
    b.id, b.judul, b.penulis, b.penerbit, b.tahunTerbit, minggu, lp.status
ORDER BY
    jumlah_peminjaman DESC, minggu DESC, lp.status DESC;
```

- Output :



#### 3.9.4.2 Menampilkan hasil tabel laporan buku

- Syntax :

```
-- 3.9.4.2 Menampilkan hasil tabel laporan buku
SELECT * FROM laporan_buku_per_minggu;
```

- Output :

	judul character varying (255)	penulis character varying (255)	penerbit character varying (255)	tahunTerbit integer	minggu timestamp with time zone	status character varying (50)	jumlah_peminjaman bigint
1	What Is Non-fiction Cinema?	Trevor Ponech	Routledge	2021	2024-12-09 00:00:00+07	Terlambat	1
2	New Perspectives on Bare Noun Phrases in Romance and Beyo...	Johannes Kabatek, Albert Wall	John Benjamins Publishing	2013	2024-12-09 00:00:00+07	Terlambat	1
3	Untuk apa seni?	I. Bambang Sugiharto	Unknown	2015	2024-12-09 00:00:00+07	Kembali	1

✓ Successfully run. Total query runtime: 116 msec.

### 3.10 Stored Procedure untuk Laporan buku paling sering dibaca

Stored procedure laporan\_buku\_terpopuler ini dibuat untuk menghasilkan laporan buku yang paling sering dipinjam dalam periode waktu tertentu. Procedure ini menerima dua parameter: tanggal\_mulai dan tanggal\_akhir, yang digunakan untuk menentukan rentang waktu peminjaman. Di dalam procedure, sebuah tabel sementara (temp\_laporan\_buku) dibuat untuk menyimpan hasil laporan, yang mencakup judul buku dan jumlah peminjaman dalam rentang

waktu tersebut, diurutkan berdasarkan jumlah peminjaman terbanyak. Setelah selesai, laporan disimpan di tabel sementara, dan informasi tentang laporan yang berhasil dibuat ditampilkan.

### 3.10.1 Create Stored Procedure

- **Syntax** :

```
-- 3.10.1 Create Stored Procedure
CREATE OR REPLACE PROCEDURE laporan_buku_terpopuler(
    IN tanggal_mulai DATE,
    IN tanggal_akhir DATE
)
LANGUAGE plpgsql
AS $$
BEGIN
    RAISE NOTICE 'Laporan Buku Paling Sering Dipinjam (% - %)', tanggal_mulai, tanggal_akhir;

    -- Masukkan hasil ke tabel sementara
    CREATE TEMP TABLE temp_laporan_buku AS
    SELECT
        b.judul,
        COUNT(lp.id_peminjaman) AS total_peminjaman
    FROM
        buku b
    JOIN
        log_peminjaman lp ON b.id = lp.buku_id
    WHERE
        lp.tanggal_peminjaman BETWEEN tanggal_mulai AND tanggal_akhir
    GROUP BY
        b.judul
    ORDER BY
        total_peminjaman DESC
    LIMIT 10;

    RAISE NOTICE 'Laporan berhasil dibuat. Data tersedia di tabel sementara temp_laporan_buku.';
END;
$$;
```

- **Output** :

Data Output	Messages	Notifications
CREATE PROCEDURE		
Query returned successfully in 76 msec.		

### 3.10.2 Memanggil laporan buku terpopuler

- **Syntax** :

```
-- 3.10.2 Memanggil laporan buku terpopuler
CALL laporan_buku_terpopuler('2024-01-01', '2024-01-31');
```

- **Output** :

Data Output	Messages	Notifications
NOTICE: Laporan Buku Paling Sering Dipinjam (2024-01-01 - 2024-01-31)		
NOTICE: Laporan berhasil dibuat. Data tersedia di tabel sementara temp_laporan_buku.		
CALL		
Query returned successfully in 77 msec.		

### 3.11 Cursor untuk Laporan buku paling sering dibaca

Stored procedure laporan\_buku\_populer menggunakan cursor (cur\_populer) untuk mengambil data buku yang paling sering dipinjam. Pertama, query SELECT mengambil ID buku, judul, dan jumlah peminjaman dari tabel buku dan log\_peminjaman, mengurutkannya berdasarkan jumlah peminjaman terbanyak. Setelah itu, cursor dibuka dan data yang diambil dimasukkan ke dalam variabel (buku\_id, buku\_judul, total\_pinjam) yang kemudian ditampilkan menggunakan RAISE NOTICE. Prosedur ini berjalan dalam loop hingga semua data buku ditampilkan, lalu cursor ditutup.

#### 3.11.1 Create cursor

- Syntax :

```
-- 3.11.1 Create cursor
CREATE OR REPLACE PROCEDURE laporan_buku_populer()
LANGUAGE plpgsql
AS $$
DECLARE
    cur_populer CURSOR FOR
        SELECT
            b.id,
            b.judul,
            COUNT(p.buku_id) AS total_peminjaman
        FROM
            buku b
        LEFT JOIN
            log_peminjaman p ON b.id = p.buku_id
        GROUP BY
            b.id, b.judul
        ORDER BY
            total_peminjaman DESC;

    buku_id INTEGER;
    buku_judul TEXT;
    total_pinjam INTEGER;
BEGIN
    OPEN cur_populer;

    RAISE NOTICE 'ID Buku | Judul Buku | Total Peminjaman';

    LOOP
        FETCH cur_populer INTO buku_id, buku_judul, total_pinjam;
        EXIT WHEN NOT FOUND;
        RAISE NOTICE '% | % | %', buku_id, buku_judul, total_pinjam;
    END LOOP;

    CLOSE cur_populer;
END;
$$;
```

- Output :

Data Output	Messages	Notifications
CREATE PROCEDURE		
Query returned successfully in 74 msec.		

#### 3.11.2 Memanggil laporan buku terpopuler

- Syntax :

```
-- 3.11.2 Memanggil laporan buku terpopuler
CALL laporan_buku_populer();
```

- Output :

Data Output	Messages	Notifications
NOTICE: ID Buku   Judul Buku   Total Peminjaman		
NOTICE: 999   What Is Non-fiction Cinese?   1		
NOTICE: 2   Untuk apa seni?   1		
NOTICE: 1061   New Perspectives on Bare Noun Phrases in Romance and Beyond   1		
NOTICE: 991   Beauty Sleep   0		
NOTICE: 70   Read On...Biography   0		
NOTICE: 639   Science and Technology in World History   0		
NOTICE: 1073   The romance of the English stage   0		
NOTICE: 390   Integrated Food Safety and Veterinary Public Health   0		
NOTICE: 539   A History of Greek Philosophy from the Earliest Period to the Time of Socrates   0		
NOTICE: 758   Spirituality, Values and Mental Health   0		
NOTICE: 1108   Condition of Education 2009   0		
NOTICE: 1128   The Span of Mainstream and Science Fiction   0		
NOTICE: 874   The technology of bacteria investigation; explicit directions for the   0		
NOTICE: 278   Visualizing Medieval Medicine and Natural History, 1200-1550   0		

## 3.12 Backup dan Restore

### 3.12.1 Backup database

Backup adalah proses membuat salinan data dari database yang ada. Backup dilakukan untuk memastikan bahwa data dapat dipulihkan jika terjadi kehilangan data.

- **Syntax :**

```
C:\Program Files\PostgreSQL\16\bin>pg_dump -U postgres -d Perpustakaan_Kel-07_SBD -F c -f "C:\Semester3\SBD\PROYEK\07_Sistem Perpustakaan Digital\07_BackupRestoreSistemPerpustakaanDigital.backup"
```

- **Output :**

Name	Size	Modified	Type
07_Sistem Perpustakaan Digital.backup	11/12/2024 08:37	BACKUP File	0 KB

### 3.12.2 Restore database

Restore adalah proses mengembalikan database dari salinan backup. Proses ini dilakukan ketika database mengalami kerusakan atau jika ada kebutuhan untuk memuat ulang data ke dalam sistem baru.

- **Syntax :**

```
C:\Program Files\PostgreSQL\16\bin>pg_restore -U postgres -d Restore_Perpustakaan_Kel-07_SBD -F c "C:\Semester3\SBD\PROYEK\07_Sistem Perpustakaan Digital\07_BackupRestoreSistemPerpustakaanDigital.backup"
Password:

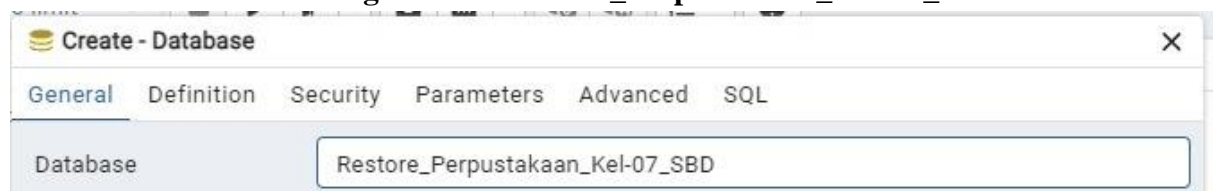
C:\Program Files\PostgreSQL\16\bin>
```

- **Output :**

Local Disk (C:) > Semester3 > SBD > PROYEK > 07_Sistem Perpustakaan Digital			
Sort View ...			
Name	Date modified	Type	Size
07_BackupRestoreSistemPerpustakaanDigital.ba...	11/12/2024 08:41	BACKUP File	72 KB

Lakukan pengecekan pada database yang sudah di create dengan nama '07\_BackupRestoreSistemPerpustakaanDigital.backup' sebagai berikut :

#### 1. Create database baru dengan nama Restore\_Perpustakaan\_Kel-07\_SBD



## 2. Jika sudah, klik Create Script -> lakukan select \* from log\_peminjaman.

help

Dashboard X Properties X SQL X Statistics X Dependencies X Dependents X Processes X Kel-07\_Sistem Pes... X Restore\_Pe

Restore\_Perpustakaan\_Kel-07\_SBD/postgres@PostgreSQL 16

Query Query History

```
1 -- Database: Restore_Perpustakaan_Kel-07_SBD
2
3 -- DROP DATABASE IF EXISTS "Restore_Perpustakaan_Kel-07_SBD";
4
5 CREATE DATABASE "Restore_Perpustakaan_Kel-07_SBD"
6 WITH
7     OWNER = postgres
8     ENCODING = 'UTF8'
9     LC_COLLATE = 'English_Indonesia.1252'
10    LC_CTYPE = 'English_Indonesia.1252'
11    LOCALE_PROVIDER = 'libc'
12    TABLESPACE = pg_default
13    CONNECTION LIMIT = -1
14    IS_TEMPLATE = False;
15
16 SELECT * FROM log_peminjaman;
```

## 3. Nanti akan menghasilkan output :

	id_peminjaman [PK] integer	anggota_id integer	buku_id integer	tanggal_peminjaman date	tanggal_pengembalian date	status character varying (50)	denda numeric (10,2)
1		6	3	2024-12-10	2024-12-13	Kembali	0.00
2		3	1061	2024-12-10	2024-12-18	Terlambat	2000.00
3		2	999	2024-12-10	2024-12-19	Terlambat	4000.00

## 3.13 Fungsi Mencari Buku

### 3.13.1 Create Function

- Syntax :

```
-- 3.12.1 Create View
CREATE OR REPLACE FUNCTION cari_buku(judul_buku VARCHAR)
RETURNS TABLE (
    id INTEGER,
    judul VARCHAR,
    penulis VARCHAR,
    penerbit VARCHAR,
    tahunTerbit INTEGER,
    stok INTEGER,
    kategori_id INTEGER
) AS $$
BEGIN
    RETURN QUERY
    SELECT b.id, b.judul, b.penulis, b.penerbit, b.tahunTerbit, b.stok, b.kategori_id
    FROM buku b
    WHERE b.judul ILIKE '%' || judul_buku || '%';
END;
$$ LANGUAGE plpgsql;
```

- Output :



[Data Output](#)   [Messages](#)   [Notifications](#)

CREATE FUNCTION

Query returned successfully in 211 msec.

### 3.13.2 Pengujian

- **Syntax** :

### -- 3.12.2 Pengujian

```
SELECT * FROM cari_buku('art');
```

- **Output :**

Data Output		Messages	Notifications
			SQL
ID integer	Judul character varying		
1	2 Art in the Hellenistic Age		
2	4 Nail Art		
3	5 American Folk Art for Kids		
4	6 Primitivism and Twentieth-century Art		
5	7 Chinese Art in Detail		
6	8 Art and Collecting Art (English Edition)		
7	9 Lewat Deviant Art		
8	10 What's Wrong with Contemporary Art?		
9	11 The Art of Forgiving 2		
10	12 The Art of Forgiving 1		
11	13 Perawatan Tangan, Kaki, Nail Art, dan Rias Wajah Khusus dan Kreatif SMK/MAK Kelas XII		
12	14 Aplikasi Algorithmic Art Untuk Motif Geometris		

## **4 Kesimpulan dan Saran**

### **4.1 Kesimpulan**

Sistem Perpustakaan Digital yang telah dikembangkan oleh Kelompok-07 berhasil memenuhi kebutuhan utama perpustakaan modern, seperti pengelolaan koleksi digital, peminjaman, pengembalian, dan pembayaran denda secara online. Sistem ini memberikan kemudahan akses bagi pengguna dan mengotomatiskan proses manajemen perpustakaan. Namun, sistem masih memiliki keterbatasan pada jumlah koleksi buku yang hanya sebanyak 1059 item dan tergantung pada koneksi internet untuk menjalankan fitur crawling.

### **4.2 Saran**

#### **1. Penambahan Koleksi Buku**

Sistem dapat ditingkatkan dengan menambahkan lebih banyak koleksi buku untuk memenuhi kebutuhan pengguna yang beragam.

#### **2. Optimalisasi Crawling**

Usahakan untuk mengoptimalkan proses crawling agar tetap efisien, bahkan pada koneksi internet yang kurang stabil.

#### **3. Pengembangan Fitur Tambahan**

Tambahkan fitur seperti rekomendasi buku berbasis preferensi pengguna atau integrasi dengan platform pembelajaran.

#### **4. Pengujian Lebih Lanjut**

Lakukan pengujian sistem secara menyeluruh untuk memastikan performa, keamanan, dan konsistensi data.

#### **5. Pengembangan Sistem Offline**

Pertimbangkan untuk menyediakan beberapa fitur dasar yang dapat diakses secara offline untuk mendukung pengguna dengan kendala internet.

Saran-saran ini diharapkan dapat membantu meningkatkan kualitas dan efisiensi Sistem Perpustakaan Digital.

## HASIL PRESENTASI

**Tabel 3. Jawaban Pertanyaan**

No .	Pertanyaan	11323013	11323028	11323051	11323055
1	Apakah keseluruhan implementasi fitur tercapai dan relevan dengan tujuannya?	Ya, Seluruh fitur kami berjalan dengan sempurna.	Ya, dari semua fitur yang kami tampilkan berjalan dengan baik.	Ya, secara keseluruhan fitur yang tersedia sudah berjalan dengan baik	Ya, secara keseluruhan semua fitur sudah terimplementasi dengan baik.
2	Apa yang lebih dan apa yang kurang untuk diperbaiki (kalau ada)?	Kami menambahkan beberapa fitur tambahan selain fitur utama yang sudah ditentukan, yaitu fitur proses pembayaran denda, laporan peminjaman (tahunan, bulanan, mingguan), laporan semua anggota yang meminjam, serta fitur pencarian.	Selain fitur utama yang telah ditetapkan, kami juga mengembangkan fitur tambahan, seperti proses pembayaran denda, laporan peminjaman (tahunan, bulanan, mingguan), laporan semua anggota peminjam, dan fitur pencarian data.	Fitur utama yang ditentukan telah kami penuhi, namun kami juga menambahkan fitur lainnya, yaitu proses pembayaran denda, laporan peminjaman (berdasarkan periode), laporan semua anggota peminjam, serta fitur pencarian.	Kami melengkapi fitur utama dengan beberapa tambahan, seperti proses bayar denda, laporan peminjaman (tahunan, bulanan, mingguan), laporan peminjaman semua anggota, dan fitur pencarian untuk memudahkan pengguna.
3	Menurutmu apakah kontribusimu sudah maksimal dan kamu paham dengan yang kamu implementasikan?	Kontribusi yang sudah saya lakukan dapat saya pahami dan dapat saya implementasikan, serta pengerjaan yang saya lakukan dapat saya pertanggung jawabkan	Kontribusi yang saya lakukan dapat saya pahami dan dapat implementasikan dengan baik dengan pekerjaan yang saya lakukan.	Menurut saya kontribusi yang sudah saya lakukan sudah cukup baik dalam tim ini, dan setelah saya mengerjakan proyek ini saya juga menjadi lebih paham mengenai pengimplementasian setiap syntax.	Menurut saya, kontribusi saya sudah maksimal karena saya terlibat dalam penyusunan laporan dan pembuatan query bersama kelompok. Saya juga memahami implementasi yang dilakukan, sehingga dapat berkontribusi secara optimal.

**Tabel 4. Penilaian**

No.	Jenis Penilaian	11323013	11323028	11323051	11323055
1	Hasil Kelompok	96	96	97	98
2	Diri Sendiri	95	95	97	97