MODUL PRAKTIKUM BASIS DATA

Pertemuan 4: Data Definition Language (DDL) pada PostgreSQL

1. Capaian Pembelajaran Mata Kuliah (CPMK)

Mahasiswa mampu memahami konsep dasar Data Definition Language (DDL) dan dapat mengimplementasikannya untuk merancang, membuat, serta memodifikasi struktur skema basis data relasional menggunakan PostgreSQL dengan menerapkan *constraint* (aturan) data yang tepat.

2. Tujuan Praktikum

Setelah menyelesaikan praktikum ini, mahasiswa diharapkan:

- Memahami dan mampu menggunakan perintah dasar DDL (CREATE, ALTER, DROP, TRUNCATE).
- Mampu membuat *database* dan tabel baru dengan pemilihan tipe data yang sesuai di PostgreSQL.
- Mampu mendefinisikan *constraint* seperti PRIMARY KEY, FOREIGN KEY, NOT NULL, dan UNIQUE.
- Mampu memodifikasi struktur tabel yang sudah ada (menambah kolom, menghapus kolom, mengubah tipe data).

3. Dasar Teori

Dalam sistem basis data, Structured Query Language (SQL) dibagi ke dalam beberapa kategori perintah yang dirancang untuk mendefinisikan struktur, mengelola data, mengatur hak akses, dan mengendalikan transaksi. Klasifikasi ini membantu memastikan pengelolaan database berlangsung secara terstruktur, efisien, dan konsisten.

Bagian Utama Perintah SQL

Perintah SQL dikelompokkan menjadi beberapa bagian sesuai fungsinya:

1. **DDL (Data Definition Language)**
   Digunakan untuk mengatur struktur database. Contoh: CREATE, ALTER, DROP, TRUNCATE.
2. **DML (Data Manipulation Language)**

Digunakan untuk mengolah data di dalam tabel. Contoh: SELECT, INSERT, UPDATE, DELETE.

3. **DCL (Data Control Language)**
   Digunakan untuk mengatur hak akses pengguna. Contoh: GRANT, REVOKE.
4. **4. TCL (Transaction Control Language)**
   Digunakan untuk mengelola transaksi data. Contoh: COMMIT, ROLLBACK, SAVEPOINT.

3.1. Data Definition Language (DDL)

adalah kumpulan perintah SQL yang digunakan untuk mendefinisikan, mengubah, serta menghapus struktur dari objek-objek di dalam basis data (seperti *database*, tabel, *view*, *index*). DDL tidak berurusan dengan manipulasi isi/data, melainkan hanya dengan kerangka penyimpanannya.

Perintah utama dalam DDL meliputi:

- **CREATE**: Membuat objek baru (*database* atau tabel).

  Membuat Database

  ```
  CREATE DATABASE nama_database;
  ```

  Membuat Tabel

  ```
  CREATE TABLE nama_tabel (
      nama_kolom_1 tipe_data_1 [constraint_1],
      nama_kolom_2 tipe_data_2 [constraint_2],
      ...
      [table_constraint]
  );
  ```

- **ALTER**: Memodifikasi struktur objek yang sudah ada.

  Menambah Kolom Baru:

  ```
  ALTER TABLE nama_tabel
  ADD COLUMN nama_kolom_baru tipe_data;
  ```

  Menghapus Kolom:

  ```
  ALTER TABLE nama_tabel
  DROP COLUMN nama_kolom;
  ```

  Mengubah Tipe Data Kolom:

```
ALTER TABLE nama_tabel
ALTER COLUMN nama_kolom TYPE tipe_data_baru;
```

Mengganti Nama Kolom:

```
ALTER TABLE nama_tabel
RENAME COLUMN nama_kolom_lama TO nama_kolom_baru;
```

- **DROP**: Menghapus objek beserta seluruh strukturnya dari sistem.

  Menghapus Tabel:

  ```
  DROP TABLE nama_tabel;
  ```

  Menghapus Database:

  ```
  DROP DATABASE nama_database;
  ```

- **TRUNCATE**: Menghapus semua baris data di dalam tabel dengan cepat, namun tetap mempertahankan struktur tabelnya.

  Mengosongkan Satu Tabel:

  ```
  TRUNCATE TABLE nama_tabel;
  ```

  Mengosongkan Tabel dan Mengulang Auto-Increment (SERIAL):

  ```
  TRUNCATE TABLE nama_tabel RESTART IDENTITY;
  ```

Tipe Data Umum di PostgreSQL:

- **Numerik**: INTEGER, BIGINT, NUMERIC, SERIAL (untuk *auto-increment*).
- **Karakter/Teks**: VARCHAR(n) (teks dengan batas panjang *n*), CHAR(n), TEXT (teks panjang tanpa batas).
- **Tanggal/Waktu**: DATE, TIME, TIMESTAMP.
- **Boolean**: BOOLEAN (menyimpan nilai *true* atau *false*).

4. Alat dan Bahan

- PC atau Laptop.
- Sistem Operasi Windows, Linux, atau macOS.

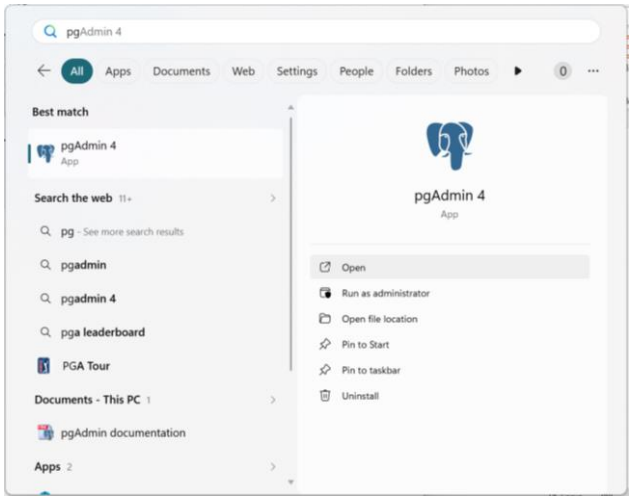
- Aplikasi *Database Server*: PostgreSQL (versi 10 ke atas direkomendasikan).
- *Database Client/GUI*: pgAdmin 4 atau Command Line Interface (psql).

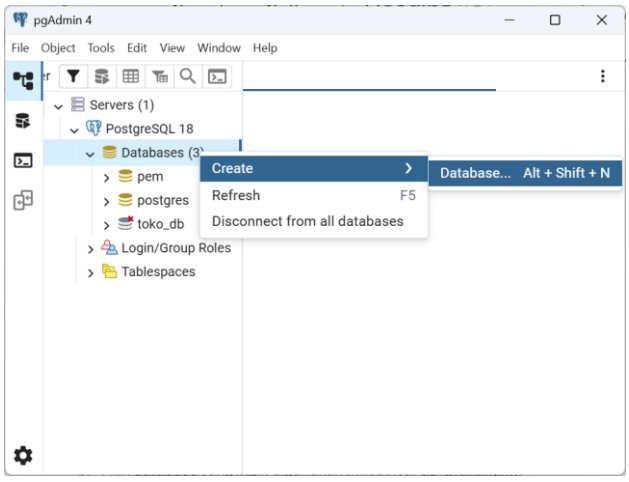
5. Percobaan (Langkah Praktikum)

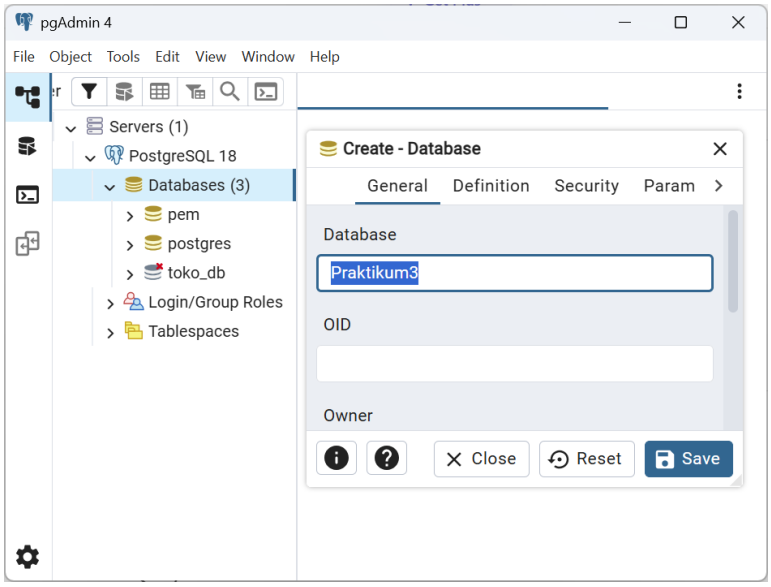
Pada percobaan ini, kita akan membuat sebuah sistem basis data Akademik sederhana yang terdiri dari tabel Fakultas dan tabel Mahasiswa. Buka *Query Tool* di pgAdmin atau terminal psql Anda, lalu jalankan perintah berikut secara berurutan:

A. pgAdmin

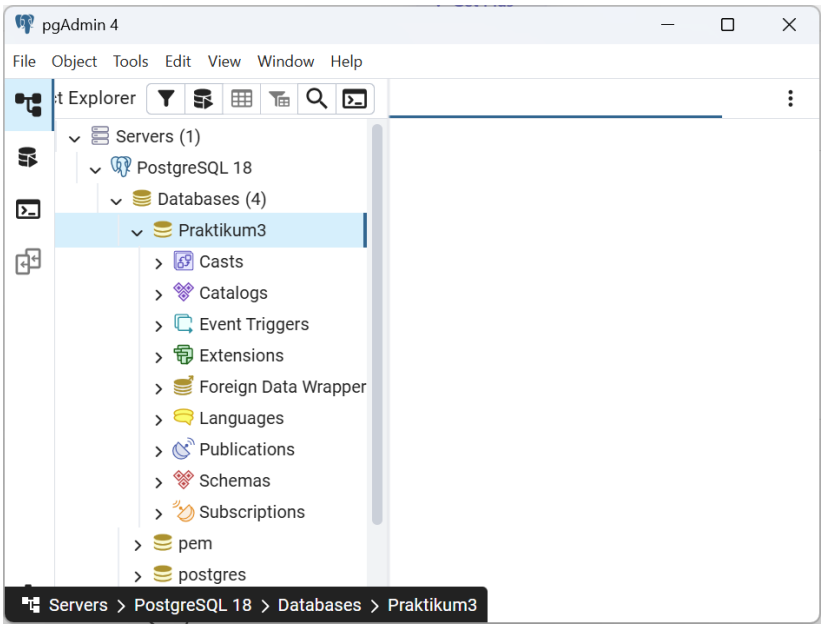
1. Buka pgAdmin
   a. Jalankan aplikasi pgAdmin.
   b. Login menggunakan password PostgreSQL jika diminta.





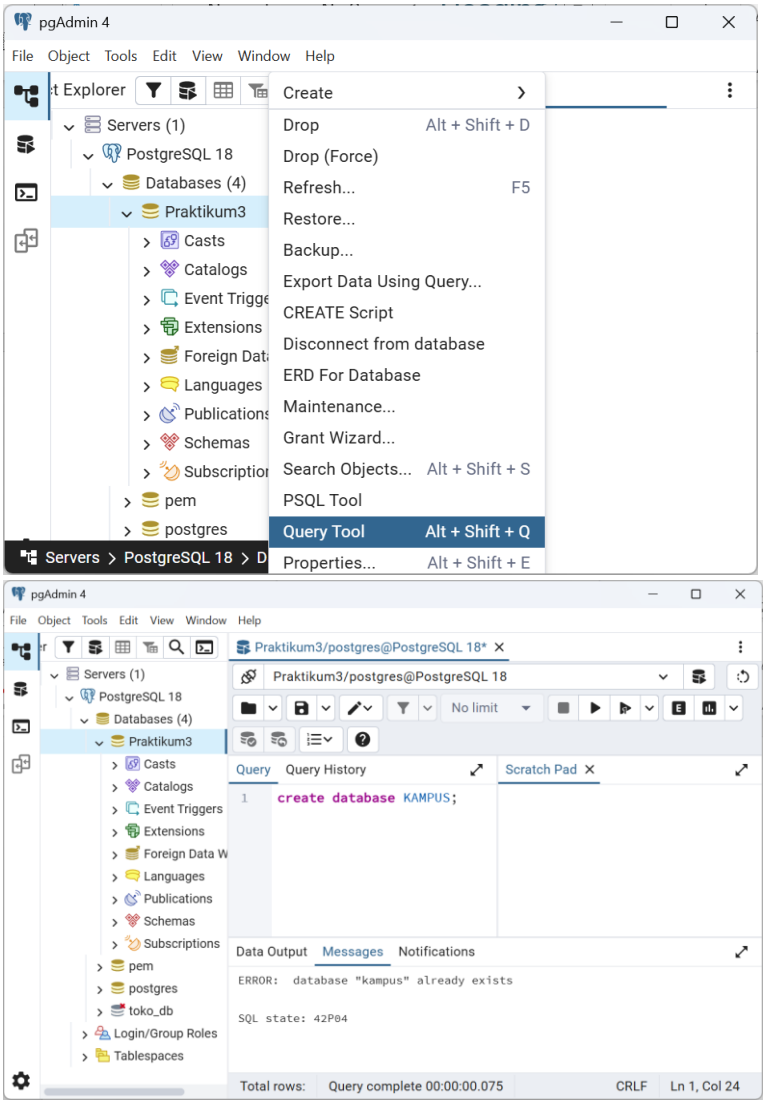


2. Pilih Database
   a. Di panel kiri (Browser), buka:
      - Servers
      - pilih server PostgreSQL Anda
      - klik Databases
   b. Pilih database yang ingin digunakan (misalnya: praktikum3).



3. Buka Query Tool

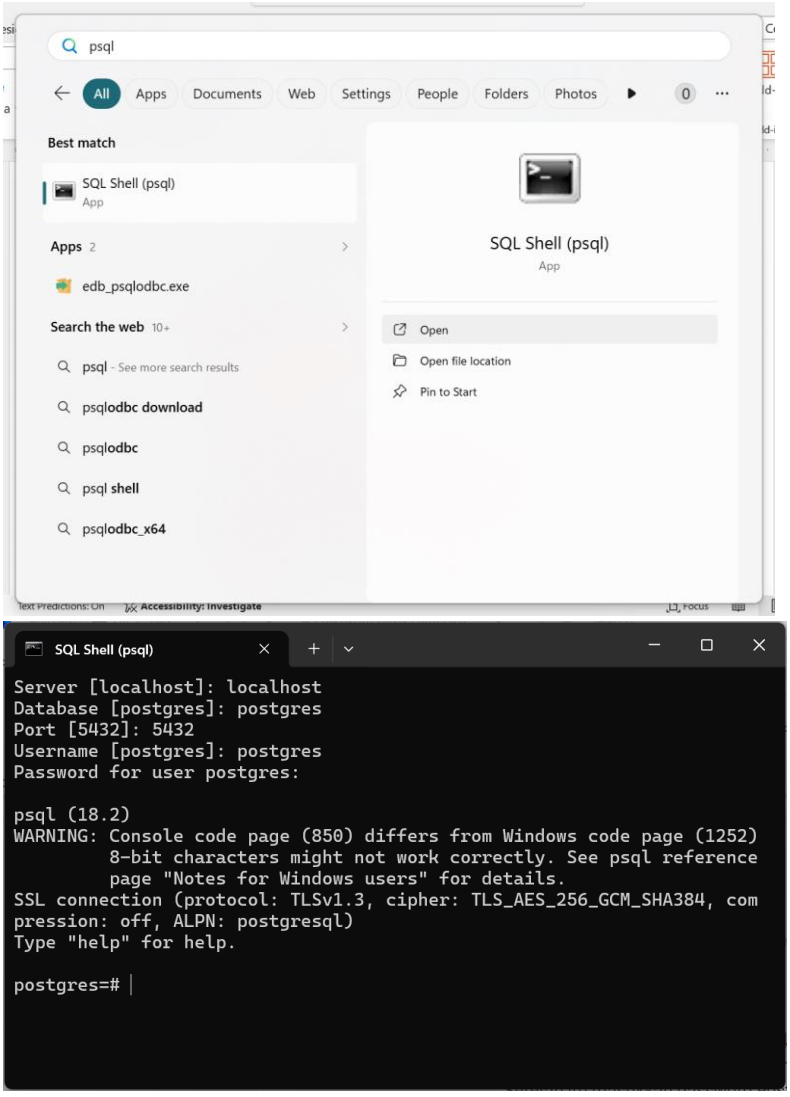
   Klik kanan pada database → pilih Query Tool.
   Atau melalui menu:
   Tools → Query Tool
   Akan muncul SQL Editor tempat menulis perintah SQL.



B. Psql terminal

1. Membuka psql

   Buka Terminal psql, kemudian jalankan perintah:

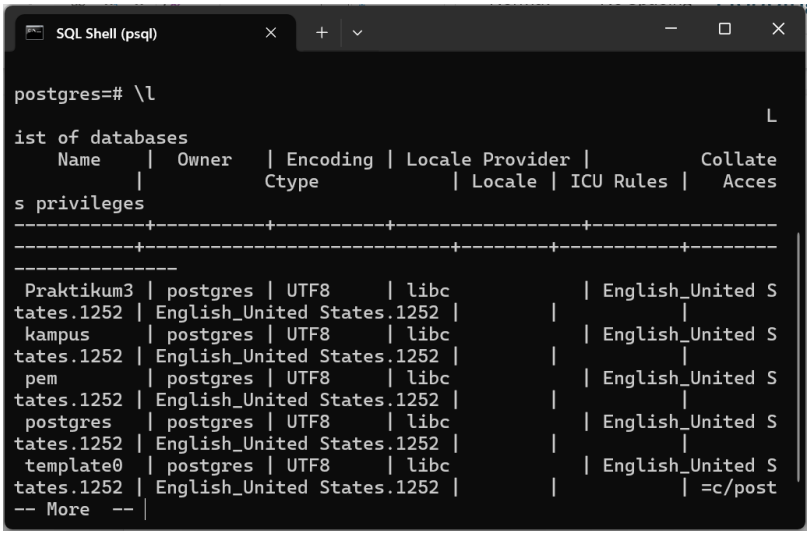


Penjelasan:

- psql → program client PostgreSQL
- -U postgres → login menggunakan user postgres

Setelah itu masukkan password PostgreSQL.
Jika berhasil, tampilan akan berubah seperti:
postgres=#
Artinya Anda sudah masuk ke psql shell.

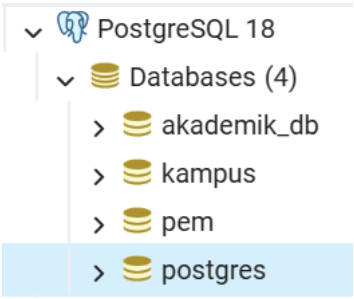
2. Memilih atau Membuat Database
Melihat daftar database
\l



A. Membuat Database

Pertama, buat sebuah *database* baru.

```
postgres=# CREATE DATABASE akademik_db;
CREATE DATABASE
postgres=#
```



B. Membuat Tabel dengan Constraint (CREATE)

Kita akan membuat tabel Fakultas dan Mahasiswa. Tabel Mahasiswa akan memiliki relasi ke tabel Fakultas.

-- Membuat tabel Fakultas

```
postgres=# CREATE TABLE Fakultas (
postgres(#     id_fakultas SERIAL PRIMARY KEY,
postgres(#     nama_fakultas VARCHAR(100) NOT NULL UNIQUE
postgres(# );
CREATE TABLE
```

-- Membuat tabel Mahasiswa

Query Query History

```
1  CREATE TABLE Mahasiswa (
2      nim VARCHAR(15) PRIMARY KEY,
3      nama VARCHAR(150) NOT NULL,
4      tanggal_lahir DATE,
5      id_fakultas INT,
6      CONSTRAINT fk_fakultas
7          FOREIGN KEY (id_fakultas)
8          REFERENCES Fakultas(id_fakultas)
9          ON DELETE SET NULL
10 );
11
```

Data Output Messages Notifications

```
CREATE TABLE

Query returned successfully in 117 msec.
```

C. Memodifikasi Struktur Tabel (ALTER)

Misalnya, kita lupa menambahkan kolom alamat dan jenis kelamin pada tabel Mahasiswa, serta ingin mengubah batasan panjang karakter pada kolom nama.

-- Menambahkan kolom alamat

Query Query History

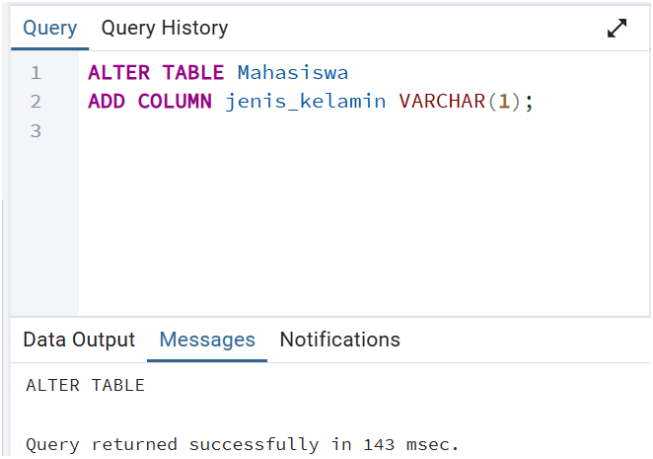
```
1  ALTER TABLE Mahasiswa
2  ADD COLUMN alamat TEXT;
3
```

Data Output Messages Notifications

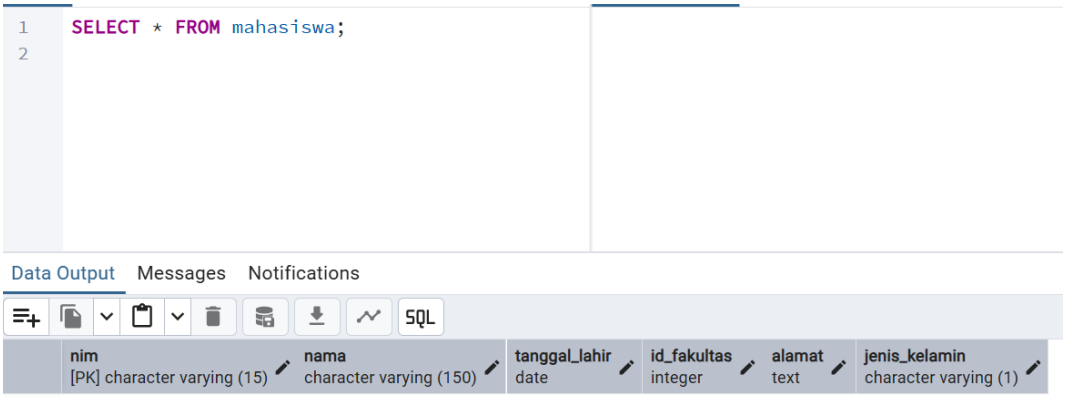
```
ALTER TABLE

Query returned successfully in 109 msec.
```

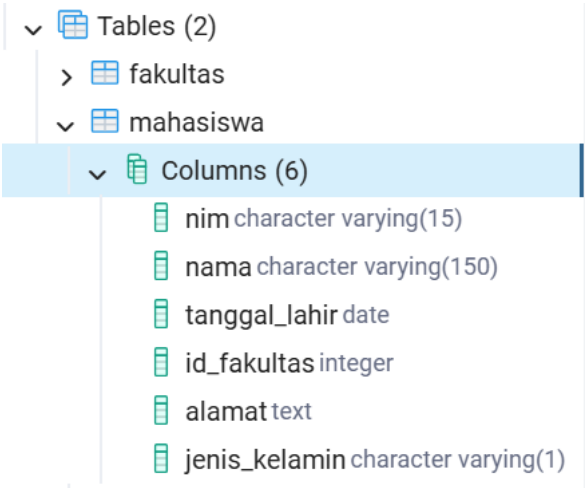
-- Menambahkan kolom jenis kelamin



-- Melihat isi tabel



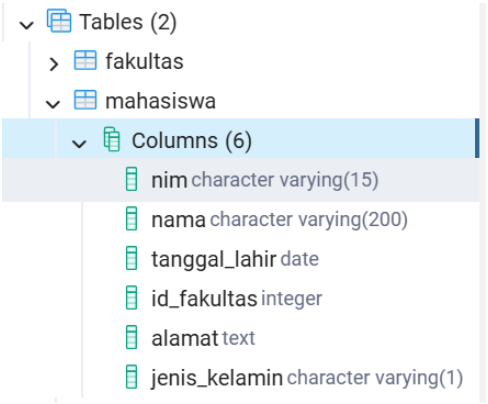
-- Mengubah tipe data kolom nama menjadi VARCHAR(200)



```
ALTER TABLE Mahasiswa
ALTER COLUMN nama TYPE VARCHAR(200);
```

```
ALTER TABLE

Query returned successfully in 108 msec.
```

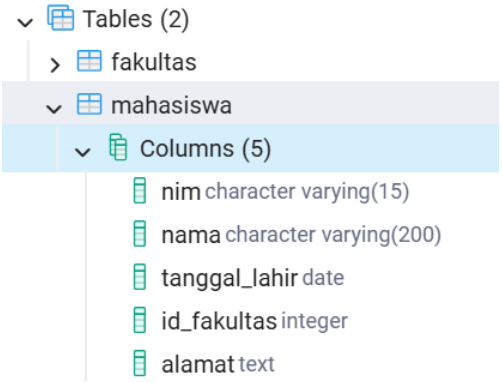


-- Menghapus kolom jenis_kelamin

```
ALTER TABLE Mahasiswa
DROP COLUMN jenis_kelamin;
```

```
ALTER TABLE

Query returned successfully in 187 msec.
```

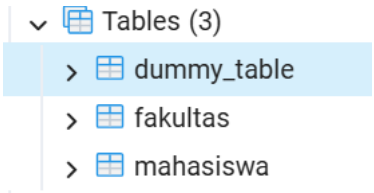


D. Menghapus Data dan Objek (TRUNCATE & DROP)

Untuk keperluan uji coba, mari buat sebuah tabel *dummy*, lalu kita kosongkan dan hapus.

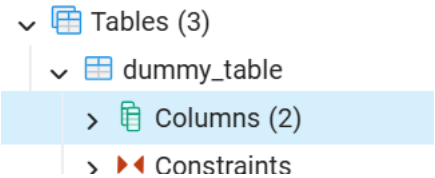
-- Membuat tabel dummy

```
CREATE TABLE Dummy_Table (
    id INT PRIMARY KEY,
    catatan TEXT
);
```



-- Menghapus semua isi tabel (struktur tetap ada)

```
TRUNCATE TABLE Dummy_Table;
```



-- Menghapus tabel secara permanen dari database

```
DROP TABLE Dummy_Table;
```

6. Latihan

Untuk menguji pemahaman, selesaikan tugas berikut secara mandiri menggunakan PostgreSQL!

Studi Kasus: Sistem Rumah Sakit

1. Buatlah sebuah *database* baru bernama rumahsakit_xx.(**xx adalah No. NRP akhir dari masing-masing mahasiswa**)
2. Buatlah tabel Dokter dengan ketentuan kolom berikut:
    - id_dokter (integer, *auto-increment*/SERIAL, Primary Key)
    - nama_dokter (teks, panjang maksimal 100, tidak boleh kosong)
    - spesialisasi (teks, panjang maksimal 50)
3. Buatlah tabel Pasien dengan ketentuan kolom berikut:
    - id_pasien (integer, *auto-increment*/SERIAL, Primary Key)
    - nama_pasien (teks, panjang maksimal 100, tidak boleh kosong)
    - tanggal_daftar (tipe data tanggal)
4. Gunakan perintah ALTER untuk:
    - Menambahkan kolom no_telepon (VARCHAR maksimal 15 karakter) pada tabel Pasien.
    - Mengubah tipe data kolom spesialisasi di tabel Dokter menjadi VARCHAR maksimal 100 karakter.
5. *(Tantangan)* Buatlah tabel Rekam_Medis yang menghubungkan Dokter dan Pasien (menggunakan *Foreign Key* ke id_dokter dan id_pasien), ditambah kolom keluhan (TEXT) dan tanggal_periksa (DATE).