**LAPORAN RESMI**

**MODUL I**

**DATA DEFINITION LANGUAGE (DDL) DAN**

**DATA MANIPULATION LANGUAGE (DML)**



**NAMA : ANISYAFAAH**

**N.R.P : 220441100105**

**DOSEN : FITRI DAMAYANTI, S.Kom., M.Kom.**

**ASISTEN : AFFAN MAULANA ZULKARNAIN**

**TGL PRAKTIKUM : 22 MARET 2024**

**Disetujui : 26 Maret 2024**

**Asisten**

**AFFAN MAULANA ZULKARNAIN**

**20.04.411.00052**

**LABORATORIUM BISNIS INTELIJEN SISTEM**

**PRODI SISTEM INFORMASI**

**JURUSAN TEKNIK INFORMATIKA**

**FAKULTAS TEKNIK**

**UNIVERSITAS TRUNOJOYO MADURA**

# **BAB I PENDAHULUAN**

## **Latar Belakang**

Dalam era digital yang semakin maju, kebutuhan akan pengelolaan data yang efisien semakin penting. Database menjadi fondasi utama dalam menyimpan, mengelola, dan mengakses informasi secara terstruktur. Tanpa database, pengelolaan data akan menjadi tidak teratur dan sulit diakses, terutama saat data tersebut memiliki volume yang besar. Dengan menggunakan database, organisasi dapat menyimpan beragam jenis data mulai dari informasi pelanggan, transaksi bisnis, hingga data kesehatan dengan efisien dan aman.

Data Definition Language (DDL) merupakan bagian penting dalam pengelolaan database yang berkaitan dengan definisi struktur database. Melalui DDL, pengguna dapat mendefinisikan tabel, kolom, indeks, dan konstrain dalam database. Ini memungkinkan pengguna untuk merancang struktur database yang sesuai dengan kebutuhan bisnis atau aplikasi yang dijalankan. DDL juga memungkinkan pengguna untuk memodifikasi struktur database, seperti menambah atau menghapus tabel, memodifikasi tipe data, atau menambahkan indeks untuk meningkatkan performa kueri.

Di sisi lain, Data Manipulation Language (DML) berkaitan dengan manipulasi data yang ada dalam database. DML memungkinkan pengguna untuk menyisipkan, mengubah, menghapus, dan memperoleh data dari tabel dalam database. Dengan DML, pengguna dapat melakukan berbagai operasi terhadap data, seperti menambahkan entri baru ke dalam database, memperbarui nilai yang ada, atau menghapus data yang tidak diperlukan lagi.

## **Tujuan**

* Mampu memahami perintah-perintah untuk menjelaskan objek dari database dan mendefinisikan atribut-atribut database, table dan batasan-batasan terhadap suatu atribut serta hubungan antar table
* Mampu memahami dan mengisi table dalam database dan memanipulasi data dalam database

# **BAB II DASAR TEORI**

## **Data Definition Language**

DDL SQL memungkinkan dilakukannya spesifikasi tidak hanya pada himpunan relasi tetapi juga informasi untuk setiap relasi, yang meliputi:

* + - Skema setiap relasi
    - Domain nilai setiap atribut relasi
    - Konstrain integritas
    - Himpunan indeks untuk setiap relasi
    - Sekuriti dan autorisasi setiap relasi
    - Struktur penyimpanan secara fisik untuk setiap relasi dalam disk.

DDL (Data Definition Languange) merupakan sekumpulan set perintah yang bertujuan untuk mendefinisikan atribut – atribut database, tabel, atribut kolom (field), maupun batasan – batasan terhadap suatu atribut dan relasi/hubungan antar tabel. Yang termasuk dalam kelompok perintah DDL adalah: CREATE, ALTER, dan DROP.

**CREATE** merupakan perintah DDL yang digunakan untuk membuat database maupun tabel. Nama database maupun tabel tidak boleh mengandung spasi (space). Nama database tidak boleh sama antar database.

**ALTER** merupakan perintah DDL yang digunakan untuk mengubah nama/struktur tabel.

**DROP** merupakan perintah DDL yang digunakan untuk menghapus database ataupun tabel.

### **DATABASE**

Database dapat dibayangkan sebagai sebuah lemari arsip, dimana arsip yang disimpan didalam lemari tersebut akan disusun dan dikelompokkan berdasarkan urutan tertentu seperti urutan abjad atau urutan kronologis. Hal ini dilakukan sebagai upaya untuk mempermudah pencarian arsip, karena arsip yang tersusun dengan rapi maka proses pencarian arsip dapat lebih cepat.

Syntax untuk membuat database sebagai berikut:

CREATE DATABASE namadatabase;

Sedangkan syntax tambahan untuk menampilkan daftar nama database yang terdapat dalam database server pada MySQL menggunakan perintah:

SHOW DATABASES;

Sebelum kita membuat suatu tabel yang digunakan untuk menyimpan data, terlebih dahulu harus memilih/mengaktifkan salah satu database sebagai database aktif yang akan digunakan untuk menyimpan beberapa tabel yang akan kita buat.

Untuk memilih/mengaktifkan salah satu database menggunakan syntax:

USE namadatabase;

Contohnya kita akan mengaktifkan database db\_universitas: USE db\_universitas;

Selain perintah - perintah di atas, terdapat perintah yang berfungsi untuk menghapus database maupun tabel. Perintah tersebut adalah sebagai berikut:

DROP DATABASE namadatabase;

### **TABEL**

* + Membuat Tabel

Dalam membuat sebuah tabel, nama tabel tidak boleh mengandung spasi (space). Ketika membuat tabel ada beberapa yang harus dideklarasikan dalam pembuatannya, yaitu antara lain meliputi:

* + - Nama tabel,
    - Nama Field (Kolom),
    - Type data dari field dan
    - Panjang data.

Adapun syntax yang digunakan untuk membuat tabel secara umum adalah sebagai berikut:

CREATE TABLE namatabel (Field1, TypeData1, Field2, TypeData2);

Contoh berikut syntax untuk membuat Tabel mahasiswa:

CREATE TABLE mahasiswa (nim CHAR (20), nama\_mhs CHAR (50), login CHAR(20), pass CHAR(20), umur INT, ipk real, PRIMARY KEY(nim));

* + Menampilkan Tabel

Untuk menampilkan daftar nama tabel yang terdapat dalam database yang sedang aktif/digunakan menggunakan perintah:

SHOW TABLES;

* + Menampilkan deskripsi atribut table

Untuk menampilkan deskripsi atribut – atribut yang terdapat pada suatu tabel dengan menggunakan perintah:

DESC namaTabel;

* + Menghapus table

Untuk menghapus Tabel perintahnya sama dengan untuk menghapus database yaitu dengan menggunakan perintah DROP. Syntax yang digunakan adalah:

DROP TABLE namaTable;

Tabel yang akan dihapus harus sesuai dengan nama tabel. Misal kita akan menghapus tabel mahasiswa, maka syntax nya adalah: DROP TABLE mahasiswa;

* + Mendefinisikan Null/Not Null

Null ataupun Not Null merupakan pernyataan yang digunakan untuk membuat kolom yang kita buat boleh kosong (Null) atau tidak boleh kosong (Not Null). Ketika pada kolom tabel tidak diset, maka secara default akan bernilai Null (boleh kosong).

Untuk mendefinisikannya maka perintah yang digunakan adalah:

CREATE TABLE mahasiswa (nim CHAR (20) NOT NULL, nama\_mhs CHAR (50) NOT NULL, login CHAR(20) NOT NULL, pass CHAR(20) NOT NULL, umur INT, ipk real, PRIMARY KEY(nim));

* + Mendefinisikan Nilai Default

Nilai default merupakan nilai yang diberikan secara otomatis oleh sistem untuk suatu kolom ketika terjadi penambahan baris baru, sementara nilai pada kolom tersebut tidak diisi oleh pengguna. Contohnya adalah :

CREATE TABLE mahasiswa (nim CHAR (20), nama\_mhs CHAR (50), login CHAR(20), pass CHAR(20), umur INT DEFAULT 0, ipk real, PRIMARY KEY(nim));

* + Mendefinisikan PRIMARY KEY pada Tabel

Suatu keharusan dalam suatu tabel adalah harus memiliki satu kolom yang dijadikan sebagai perwakilan dari tabel tersebut. Pembuatan perwakilan tabel ini berfungsi untuk melakukan hubungan/relasional dengan tabel lain. Bentuk perwakilan ini dalam database disebut sebagai PRIMARY KEY yang aturan pembuatannya adalah sebagai berikut:

* Satu tabel hanya diperbolehkan memiliki satu kolom kunci.
* Nama kolom kunci tidak digunakan pada kolom lain dalam satu tabel
* Nama kolom kunci tidak boleh sama dengan kolom kunci yang ada pada tabel lain
* Bentuk kolom kunci harus diset NOT NULL.

Terdapat tiga cara untuk mendefinisikan primary key. Berikut ini syntax yang digunakan:

CREATE TABLE mahasiswa (nim CHAR (20), nama\_mhs CHAR (50), login CHAR(20), pass CHAR(20), umur INT, ipk real, PRIMARY KEY(nim));

Atau

CREATE TABLE mahasiswa (nim CHAR (20) NOT NULL PRIMARY KEY, nama\_mhs CHAR (50), login CHAR(20), pass CHAR(20), umur INT, ipk real);

Atau

ALTER TABLE mahasiswa ADD CONSTRAINT namaconstraint PRIMARY KEY(namakolom);

* + Menghapus PRIMARY KEY pada Tabel

Cara 1: Jika primary key dibuat menggunakan alter table:

ALTER TABLE namatabel DROP CONSTRAINT namaconstraint;

Cara 2: jika primary key dibuat melalui create table:

ALTER TABLE namatabel DROP PRIMARY KEY;

* + Menambah kolom baru pada tabel

Pada saat kita membuat tabel terkadang kita ingin menambahkan kolom lagi pada tabel yang sudah kita buat. Dalam database, hal ini dapat dilakukan dengan menggunakan perintah sebagai berikut:

ALTER TABLE namatabel ADD fieldbaru typedata(lebar);

Namatabel merupakan nama tabel yang akan ditambahkan kolomnya.

Field baru merupakan nama kolom yang akan ditambahkan, typedata(lebar) merupakan type data dan lebar data yang akan ditambahkan.

Misal kita akan menambahkan kolom telepon pada tabel mahasiswa setelah kolom umur:

ALTER TABLE mahasiswa ADD COLUMN telepon CHAR(15) AFTER umur;

* + Mengubah Tipe Data atau Lebar Kolom pada Tabel

Perintah yang digunakan adalah:

ALTER TABLE namatabel MODIFY COLUMN field

Contoh:

ALTER TABLE mahasiswa MODIFY COLUMN CHANGE COLUMN telepon(12);

ALTER TABLE namatabel namakolomlama namakolombaru typedatabaru(lebarbaru);

* Mengubah Nama Kolom (Field) Perintah yang digunakan adalah:

Contoh:

ALTER TABLE mahasiswa CHANGE COLUMN telepon phone CHAR(25);

* + Menghapus Kolom pada Tabel

Perintah yang digunakan:

ALTER TABLE namatabel DROP COLUMN namakolom;

Contoh:

ALTER TABLE mahasiswa DROP COLUMN phone;

* + Membuat dan Menghapus Index

Index berfungsi untuk mempercepat proses pencarian data dalam suatu tabel. Dengan adanya index pada suatu field tabel menyebabkan proses pencarian otomatis akan dilakukan terlebih dahulu ke dalam index, apabila ditemukan baru akan diambilkan data sesungguhnya dari tabel, apabila tidak ditemukan dalam index, sudah dapat dipastikan bahwa data tersebut tidak ada dalam tabel. Terdapat perintah untuk membuat dan menghapus index, tapi tidak ada perintah untuk merubah index. Perhatikan contoh berikut:

CREATE INDEX IDXNIM ON mahasiswa(nim);

Atau

ALTER TABLE mahasiswa ADD INDEX IDXNIM(nim);

Sedangkan untuk menghapus:

DROP INDEX IDXNIM ON mahasiswa(nim);

Atau

ALTER TABLE mahasiswa DROP INDEX IDXNIM(nim);



## **Data Manipulation Language**

Merupakan bentuk bahasa basis data yang berguna untuk melakukan manipulasi data dan pengambilan data pada suatu basis data. Manipulasi data dapat berupa:

1. penyisipan/penambahan data baru ke suatu basis data
2. penghapusan data dari suatu basis data
3. pengubahan data di suatu basis data

Pada level fisik, kita harus mendefinisikan algoritma yang memungkinkan pengaksesan yang efisien terhadap data. Pada level yang lebih tinggi, yang dipentingkan bukan hanya efisiensi akses, tetapi juga efisiensi interaksi manusia (pemakai) dengan sistem (kemudahan permintaan akses). *Data Manipulation Language* (DML) merupakan bahasa yang bertujuan memudahkan pemakai untuk mengakses data sebagaimana direpresentasikan oleh model data. Ada 2 jenis DML, yaitu:

* Prosedural, yang mensyaratkan agar pemakai menentukan, data apa yang diinginkan serta bagaimana cara mendapatkannya.
* Nonprosedural, yang membuat pemakai dapat menentukan data apa yang diinginkan tanpa menyebutkan bagaimana cara mendapatkannya.

Perintah yang termasuk dalam DML adalah: INSERT, DELETE, UPDATE, dan SELECT

### **INSERT**

Perintah INSERT bertujuan untuk menambahkan record data pada suatu tabel. Terdapat beberapa cara untuk menambahkan record, yaitu:

Cara 1: Menambahkan record dengan mengisi data pada setiap kolom:

INSERT INTO namatabel VALUES (nilai1, nilai2, nilai-

Cara 2: menambahkan baris dengan hanya mengisi pada kolom tertentu:

INSERT INTO namatabel (field1, field2, field-n) VALUES (nilai1, nilai2, nilai-n);

Ket: Jika data bertipe string, date, atau time (contoh: didi, basis data, 1984-03-18) maka pemberian nilainya diapit menggunakan tanda petik tunggal ('Didi') atau petik ganda (“Basis Data”). Jika data bertipe numerik (29, 4) maka pemberian nilainya tidak diapit tanda petik tunggal maupun ganda.

### **DELETE**

Perintah DELETE digunakan untuk menghapus satu baris, baris dengan kondisi tertentu maupun seluruh baris. Syntax yang digunakan:

DELETE FROM namatabel WHERE [kondisi];

Perintah dalam tanda [] bersifat pilihan/opsional untuk menghapus suatu baris dengan kondisi tertentu yang dipersyaratkan. Contoh perintah untuk menghapus suatu baris dalam tabel dengan kondisi persyaratan tertentu:

### **UPDATE**

Perintah UPDATE digunakan untuk mengubah isi data pada satu atau beberapa kolom pada suatu tabel. Syntax yang digunakan secara umum adalah sebagai berikut:

UPDATE namatabel SET field1=nilai1, field2=nilai2 [WHERE kondisi];

Perintah dalam tanda [] bersifat pilihan/opsional untuk mengubah suatu baris dengan kondisi tertentu yang dipersyaratkan.

### **SELECT**

Perintah SELECT digunakan untuk menampilkan isi dari suatu tabel yang dapat dihubungkan dengan beberapa tabel lainnya.

Menampilkan data semua kolom dengan menggunakan asterisk (\*):

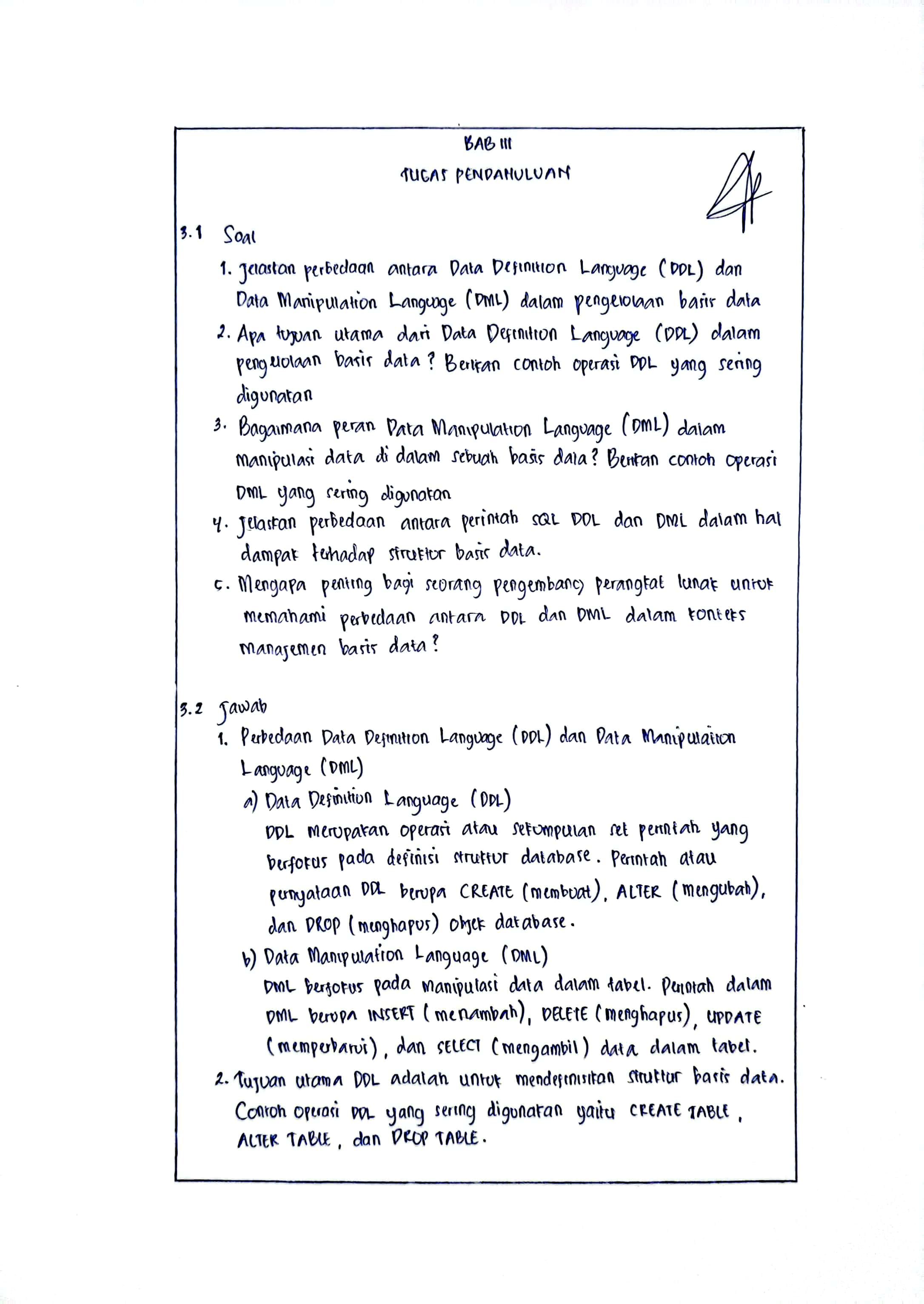
SELECT \* FROM namatabel;

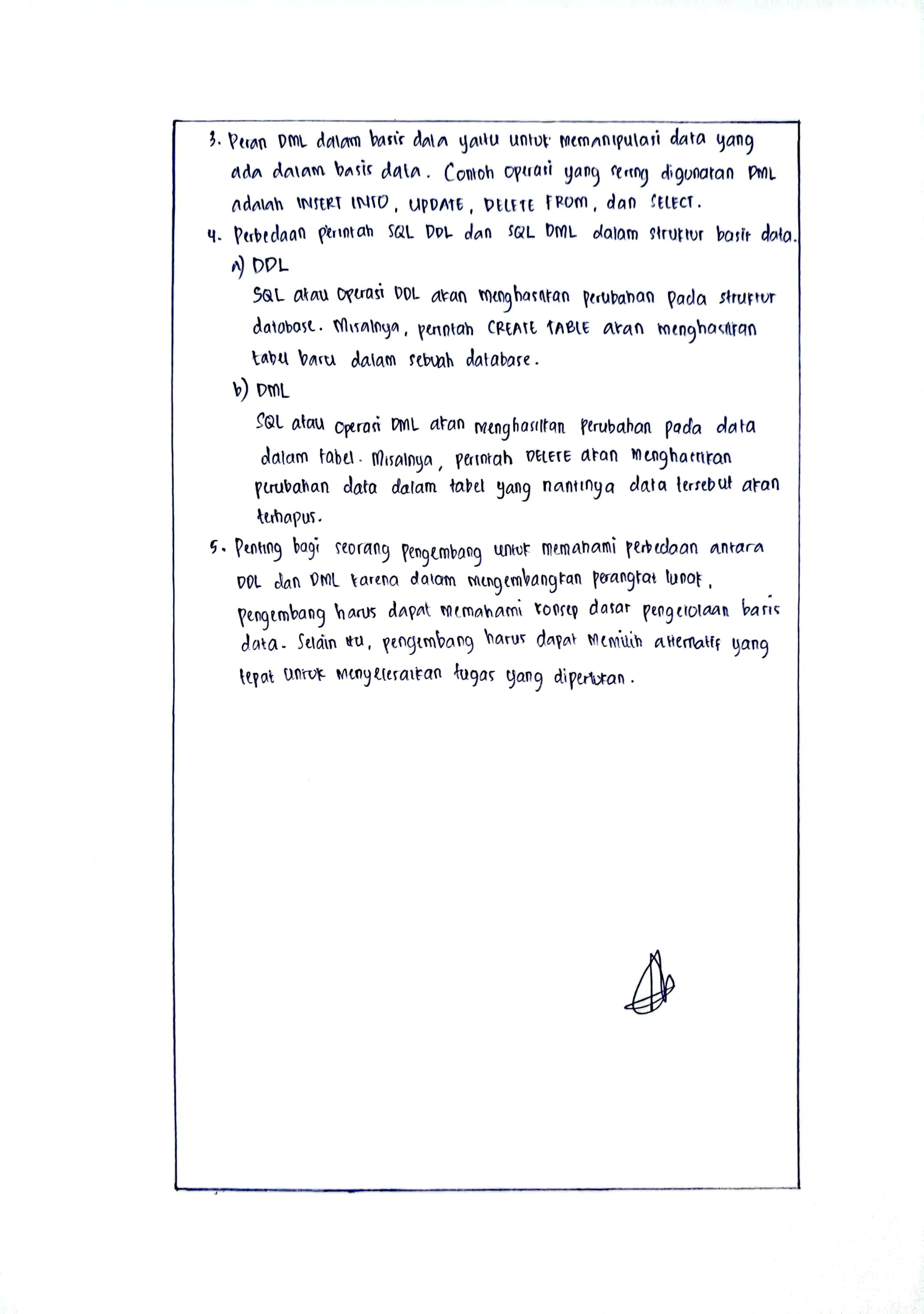
Menampilkan data untuk field/kolom tertentu:

SELECT field1, field2, field-n FROM namatabel;

Menampilkan data dengan kondisi tertentu menggunakan klausa WHERE:

SELECT \* FROM namatabel WHERE kondisi;

****

****

# **BAB IV IMPLEMENTASI**

## **Source Code**

### **Tabel Mahasiswa**

1. **Source Code**

|  |
| --- |
| CREATE DATABASE db\_utm;  USE db\_utm;  CREATE TABLE mahasiswa (nim INT (11) NOT NULL, nama VARCHAR (50), fakultas VARCHAR (50), program\_studi VARCHAR (50), statusMhs VARCHAR (20), PRIMARY KEY (nim));  INSERT INTO mahasiswa VALUES (220441101, "Anisyafaah", "Teknik", "Sistem Informasi", "Belum Lulus");  INSERT INTO mahasiswa VALUES (220441102, "Anisah Nuril Izzati", "Teknik", "Sistem Informasi", "Belum Lulus");  INSERT INTO mahasiswa VALUES (220441103, "Herdiyanti Fifin Purwaningrum", "Teknik", "Sistem Informasi", "Belum Lulus");  INSERT INTO mahasiswa VALUES (220441104, "Firdausi Putri Cahyani", "Teknik", "Sistem Informasi", "Lulus");  INSERT INTO mahasiswa VALUES (220441105, "Adhelia Kusumawati", "Teknik", "Sistem Informasi", "Lulus");  INSERT INTO mahasiswa VALUES (220441106, "Tasya Dwiyanti", "Teknik", "Sistem Informasi", "Lulus");  INSERT INTO mahasiswa VALUES (220441107, "Syahrul Ramadhani", "Teknik", "Sistem Informasi", "Berhenti");  INSERT INTO mahasiswa VALUES (220441108, "Birrur Rijaal", "Teknik", "Sistem Informasi", "Berhenti");  INSERT INTO mahasiswa VALUES (220441109, "Rayhanza Nadhif Athala", "Teknik", "Sistem Informasi", "Berhenti");  INSERT INTO mahasiswa VALUES (220441110, "Ade Verdaus Saputra", "Teknik", "Sistem Informasi", "Berhenti"); |

1. **Penjelasan**

Kode di atas digunakan untuk membuat database, tabel, dan mengisi data sebanyak 10 data. Untuk membuat database, maka perlu perintah CREATE DATABASE namadatabase. Kemudian gunakan database dengan perintah USE, dan dilanjutkan dengan membuat tabel mahasiswa. Tabel di atas memiliki kolom yang terdiri dari nim sebagai primary key, nama, fakultas, program studi dan status mahasiswa. Selanjutnya untuk mengisi data pada setiap kolom menggunakan perintah INSERT INTO.

### **Tabel Dosen**

1. **Source Code**

|  |
| --- |
| CREATE TABLE dosen (nip INT (11) NOT NULL, nama VARCHAR (50),  jurusan VARCHAR (50), jabatan VARCHAR (50), PRIMARY KEY (nip));    INSERT INTO dosen VALUES (19900801, "Dr.BUDI DWI SATOTO, ST., M.Kom.", "Sistem Informasi", "Koordiator Prodi");  INSERT INTO dosen VALUES (19900802, "FIRLI IRHAMNI, ST., M.Kom.", "Sistem Informasi", "Dosen pengajar");  INSERT INTO dosen VALUES (19900803, "MUHAMMAD ALI SYAKUR, S.Si., M.T.", "Sistem Informasi", "Dosen pengajar");  INSERT INTO dosen VALUES (19900804, "NOVI PRASTITI, S.Kom, M.Kom", "Sistem Informasi", "Dosen pengajar");  INSERT INTO dosen VALUES (19900805, "Achmad Zain Nur, S.Kom.,M.T.", "Sistem Informasi", "Dosen pengajar");  INSERT INTO dosen VALUES (19900806, "ROSIDA VIVIN NAHARI, S.Kom,.MT", "Sistem Informasi", "Dosen pengajar");  INSERT INTO dosen VALUES (19900807, "YENI KUSTIYAHNINGSIH, S.Kom., M.Kom.", "Sistem Informasi", "Dosen pengajar");  INSERT INTO dosen VALUES (19900808, "YUDHA DWI PUTRA NEGARA, S.Kom., M.Kom.", "Sistem Informasi", "Dosen pengajar");  INSERT INTO dosen VALUES (19900809, "FITRI DAMAYANTI, S.Kom., M.Kom.", "Sistem Informasi", "Dosen pengajar");  INSERT INTO dosen VALUES (19900810, "ACHMAD YASID, S.Kom. M.Kom ", "Sistem Informasi", "Dosen pengajar"); |

1. **Penjelasan**

Kode di atas digunakan untuk membuat tabel dan mengisi data sebanyak 10 data. Tidak jauh berbeda dengan sebelumnya, untuk membuat tabel dosen menggunakan perintah CREATE TABLE. Tabel di atas memiliki kolom yang terdiri dari nip sebagai primary key, nama, jurusan, dan jabatan. Selanjutnya untuk mengisi data pada setiap kolom menggunakan perintah INSERT INTO.

### **Tabel Mata Kuliah**

1. **Source Code**

|  |
| --- |
| CREATE TABLE matakuliah (kode\_matkul VARCHAR (50) NOT NULL, nip INT (11), matakuliah VARCHAR (50), sks INT (10), semester INT (10), prodi VARCHAR (50), PRIMARY KEY (kode\_matkul), FOREIGN KEY (nip) REFERENCES dosen (nip));    INSERT INTO matakuliah VALUES ("SI1", 19900801, "Sistem Manajemen Basis Data", 4, 4, "Sistem Informasi");  INSERT INTO matakuliah VALUES ("SI2", 19900802, "Pemrograman Bergerak", 4, 4, "Sistem Informasi");  INSERT INTO matakuliah VALUES ("SI3", 19900803, "Sistem Pendukung Keputusan", 3, 4, "Sistem Informasi");  INSERT INTO matakuliah VALUES ("SI4", 19900804, "Perancangan Sumber Daya Perusahaan", 3, 4, "Sistem Informasi");  INSERT INTO matakuliah VALUES ("SI5", 19900805, "Manajemen Proyek IT", 3, 4, "Sistem Informasi");  INSERT INTO matakuliah VALUES ("SI6", 19900806, "Implementasi Perancangan Perangkat Lunak", 3, 4, "Sistem Informasi");  INSERT INTO matakuliah VALUES ("SI7", 19900807, "Data Mining", 3, 4, "Sistem Informasi");  INSERT INTO matakuliah VALUES ("SI8", 19900808, "Pemrograman Visual", 4 , 3, "Sistem Informasi");  INSERT INTO matakuliah VALUES ("SI9", 19900809, "Arsitektur SI/TI", 3, 3, "Sistem Informasi");  INSERT INTO matakuliah VALUES ("SI10", 19900810, "Analisa Desain SI", 3, 3, "Sistem Informasi"); |

1. **Penjelasan**

Kode di atas digunakan untuk membuat tabel matakuliah dan mengisi data sebanyak 10 data. Untuk membuat tabel matakuliah menggunakan perintah CREATE TABLE dosen (namaKolom1, namaKolom2, dst.). Tabel di atas memiliki kolom yang terdiri dari kode matakuliah sebagai primary key, nip, matakuliah, sks, semester dan prodi. Selanjutnya untuk mengisi data pada setiap kolom menggunakan perintah INSERT INTO.

### **Tabel KRS**

1. **Source Code**

|  |
| --- |
| CREATE TABLE krs (kode\_matkul VARCHAR (10) NOT NULL, nim INT (11), nama VARCHAR (50), mataKuliah VARCHAR (50), semester INT (5), tahun\_ajaran VARCHAR (20), PRIMARY KEY(kode\_matkul), FOREIGN KEY (nim) REFERENCES mahasiswa (nim), FOREIGN KEY (kode\_matkul) REFERENCES matakuliah (kode\_matkul));  INSERT INTO krs VALUES ("SI1", 220441101, "Anisyafaah", "Sistem Manajemen Basis Data", 4, "2022-2023");  INSERT INTO krs VALUES ("SI2", 220441102, "Anisah Nuril Izzati", "Pemrograman Bergerak", 4, "2022-2023");  INSERT INTO krs VALUES ("SI3", 220441103, "Herdiyanti Fifin Purwaningrum", "Sistem Pendukung Keputusan", 4, "2022-2023");  INSERT INTO krs VALUES ("SI4", 220441104, "Firdausi Putri Cahyani", "Perancangan Sumber Daya Perusahaan", 4, "2020-2021");  INSERT INTO krs VALUES ("SI5", 220441105, "Adhelia Kusumawati", "Manajemen Proyek IT", 4, "2020-2021");  INSERT INTO krs VALUES ("SI6", 220441106, "Tasya Dwiyanti", "Implementasi Perancangan Perangkat Lunak", 4, "2020-2021");  INSERT INTO krs VALUES ("SI7", 220441107, "Syahrul Ramadhani", "Data Mining", 4, "2022-2023");  INSERT INTO krs VALUES ("SI8", 220441108, "Birrur Rijaal", "Pemrograman Visual", 4, "2022-2023");  INSERT INTO krs VALUES ("SI9", 220441109, "Rayhanza Nadhif Athala", "Arsitektur SI/TI", 4, "2022-2023");  INSERT INTO krs VALUES ("SI10", 220441110, "Ade Verdaus Saputra", "Analisa Desain SI", 4, "2022-2023"); |

1. **Penjelasan**

Kode di atas digunakan untuk membuat tabel KRS atau pengambilan matakuliah dan mengisi data sebanyak 10 data. Untuk membuat tabel KRS, maka menggunakan perintah CREATE TABLE KRS (namaKolom1, namaKolom2, dst.). Tabel di atas memiliki kolom yang terdiri dari kode matakuliah sebagai primary key, nip, matakuliah, sks, semester dan prodi. Selanjutnya untuk mengisi data pada setiap kolom menggunakan perintah INSERT INTO.

### **Tabel Pembayaran UKT**

1. **Source Code**

|  |
| --- |
| CREATE TABLE pembayaranUKT (id\_pembayaran INT (10) NOT NULL, NIM INT(11), nama VARCHAR(50), semester INT (10), jml\_ukt VARCHAR (50), tgl\_bayar DATE, PRIMARY KEY (id\_pembayaran), FOREIGN KEY (nim) REFERENCES mahasiswa (nim));  INSERT INTO pembayaranUKT VALUES (1, 220441101, "Anisyafaah", 4, "Rp 3.000.000", "2024-02-16");  INSERT INTO pembayaranUKT VALUES (2, 220441102, "Anisah Nuril Izzati", 4, "Rp 3.000.000", "2024-02-17");  INSERT INTO pembayaranUKT VALUES (3, 220441103, "Herdiyanti Fifin Purwaningrum", 4, "Rp 3.000.000", "2024-02-18");  INSERT INTO pembayaranUKT VALUES (4, 220441104, "Firdausi Putri Cahyani", 4, "Rp 3.000.000", "2024-02-19");  INSERT INTO pembayaranUKT VALUES (5, 220441105, "Adhelia Kusumawati", 4, "Rp 3.000.000", "2024-02-20");  INSERT INTO pembayaranUKT VALUES (6, 220441106, "Tasya Dwiyanti", 4, "Rp 3.000.000", "2024-02-21");  INSERT INTO pembayaranUKT VALUES (7, 220441107, "Syahrul Ramadhani", 4, "Rp 3.000.000", "2024-02-22");  INSERT INTO pembayaranUKT VALUES (8, 220441108, "Birrur Rijaal", 4, "Rp 3.000.000", "2024-02-23");  INSERT INTO pembayaranUKT VALUES (9, 220441109, "Rayhanza Nadhif Athala", 4, "Rp 3.000.000", "2024-02-24");  INSERT INTO pembayaranUKT VALUES (10, 220441110, "Ade Verdaus Saputra", 4, "Rp 3.000.000", "2024-02-25"); |

1. **Penjelasan**

Kode di atas digunakan untuk membuat tabel pembayaran UKT dan mengisi data sebanyak 10 data. Untuk membuat tabel pembayaran UKT menggunakan perintah CREATE TABLE pembayaranUKT (namaKolom1, namaKolom2, dst.). Tabel di atas memiliki kolom yang terdiri dari id\_pembayaran sebagai primary key, nim, nama, semester, jumlah UKT dan tanggal bayar. Selanjutnya untuk mengisi data pada setiap kolom menggunakan perintah INSERT INTO.

### **Tabel Gaji Dosen**

1. **Source Code**

|  |
| --- |
| CREATE TABLE gaji\_dosen (id\_gaji INT (10) NOT NULL, nip INT(11), nama VARCHAR(50), jmlGaji VARCHAR (20), tgl\_dibayar DATE, PRIMARY KEY (id\_gaji), FOREIGN KEY (nip) REFERENCES dosen (nip));  INSERT INTO gaji\_dosen VALUES (1, 19900801, "Dr.BUDI DWI SATOTO, ST., M.Kom.", "Rp 5.000.000", "2024-07-01");  INSERT INTO gaji\_dosen VALUES (2, 19900802, "FIRLI IRHAMNI, ST., M.Kom.", "Rp 5.000.000", "2024-07-02");  INSERT INTO gaji\_dosen VALUES (3, 19900803, "MUHAMMAD ALI SYAKUR, S.Si., M.T.", "Rp 10.000.000", "2024-07-01");  INSERT INTO gaji\_dosen VALUES (4, 19900804, "NOVI PRASTITI, S.Kom, M.Kom", "Rp 10.000.000", "2024-07-01");  INSERT INTO gaji\_dosen VALUES (5, 19900805, "Achmad Zain Nur, S.Kom.,M.T.", "Rp 5.000.000", "2024-07-02");  INSERT INTO gaji\_dosen VALUES (6, 19900806, "ROSIDA VIVIN NAHARI, S.Kom,.MT", "Rp 10.000.000", "2024-07-01");  INSERT INTO gaji\_dosen VALUES (7, 19900807, "YENI KUSTIYAHNINGSIH, S.Kom., M.Kom.", "Rp 5.000.000", "2024-07-02");  INSERT INTO gaji\_dosen VALUES (8, 19900808, "YUDHA DWI PUTRA NEGARA, S.Kom., M.Kom.", "Rp 10.000.000", "2024-07-02");  INSERT INTO gaji\_dosen VALUES (9, 19900809, "FITRI DAMAYANTI, S.Kom., M.Kom.", "Rp 10.000.000", "2024-07-01");  INSERT INTO gaji\_dosen VALUES (10, 19900810, "ACHMAD YASID, S.Kom. M.Kom ", "Rp 5.000.000", "2024-07-02"); |

1. **Penjelasan**

Kode di atas digunakan untuk membuat tabel gaji dosen dan mengisi data sebanyak 10 data. Untuk membuat tabel gaji dosen menggunakan perintah CREATE TABLE gaji\_dosen (namaKolom1, namaKolom2, dst.). Tabel di atas memiliki kolom yang terdiri dari id\_gaji sebagai primary key, nip, nama, jumlah gaji, dan tanggal dibayar. Selanjutnya untuk mengisi data pada setiap kolom menggunakan perintah INSERT INTO.

### **Tabel Pembelian Peralatan**

1. **Source Code**

|  |
| --- |
| CREATE TABLE pembelianPeralatan (id\_pembelian INT (10) NOT NULL, peralatan VARCHAR(50), banyakPeralatan INT (10), tgl\_beli DATE, totalHarga VARCHAR (50), PRIMARY KEY (id\_pembelian));  INSERT INTO pembelianPeralatan VALUES (1, "Papan Tulis", 3, "2024-01-07", "Rp 140.000");  INSERT INTO pembelianPeralatan VALUES (2, "Proyektor", 2, "2024-01-09", "Rp 2.000.000");  INSERT INTO pembelianPeralatan VALUES (3, "Printer", 2, "2024-01-10", "Rp 4.000.000");  INSERT INTO pembelianPeralatan VALUES (4, "Penghapus", 5, "2024-01-07", "Rp 50.000");  INSERT INTO pembelianPeralatan VALUES (5, "Spidol", 4, "2024-01-10", "Rp 36.000");  INSERT INTO pembelianPeralatan VALUES (6, "WebCam", 2, "2024-01-15", "Rp 1.000.000");  INSERT INTO pembelianPeralatan VALUES (7, "Mouse", 3, "2024-01-08", "Rp 600.000");  INSERT INTO pembelianPeralatan VALUES (8, "Keyboard", 3, "2024-01-07", "Rp 320.000");  INSERT INTO pembelianPeralatan VALUES (9, "Monitor", 1, "2024-01-07", "Rp 300.000");  INSERT INTO pembelianPeralatan VALUES (10, "HDMI", 2, "2024-01-07", "Rp 60.000"); |

1. **Penjelasan**

Kode di atas digunakan untuk membuat tabel pembelian peralatan dan bahan ajar serta mengisi data sebanyak 10 data. Untuk membuat tabel pembelian peralatan menggunakan perintah CREATE TABLE pembelianPeralatan (namaKolom1, namaKolom2, dst.). Tabel di atas memiliki kolom yang terdiri dari id\_pembelian sebagai primary key, peralatan, banyak peralatan, tanggal beli, dan total harga. Selanjutnya untuk mengisi data pada setiap kolom menggunakan perintah INSERT INTO.

### **Tabel Pembayaran Tagihan**

1. **Source Code**

|  |
| --- |
| CREATE TABLE pembayaranTagihan (id\_pembayaran INT (10), jenisTagihan VARCHAR(70), tgl\_bayar DATE, totalTagihan VARCHAR (50), PRIMARY KEY (id\_pembayaran));    INSERT INTO pembayaranTagihan VALUES (1, "Biaya Listrik", "2024-01-01", "Rp 700.000");  INSERT INTO pembayaranTagihan VALUES (2, "Biaya Air", "2024-01-02", "Rp 500.000");  INSERT INTO pembayaranTagihan VALUES (3, "Biaya Layanan Pendukung Mahasiswa", "2024-01-02", "Rp 900.000");  INSERT INTO pembayaranTagihan VALUES (4, "Biaya Pajak dan Regulasi", "2024-01-04", "Rp 1.200.000");  INSERT INTO pembayaranTagihan VALUES (5, "Biaya Administrasi", "2024-01-05", "Rp 300.000");  INSERT INTO pembayaranTagihan VALUES (6, "Biaya Pembelian Peralatan Akademik", "2024-01-06", "Rp 800.000");  INSERT INTO pembayaranTagihan VALUES (7, "Biaya Gaji Pengajar", "2024-01-07", "Rp 20.000.000");  INSERT INTO pembayaranTagihan VALUES (8, "Biaya Seminar & Konferensi", "2024-01-08", "Rp 500.000");  INSERT INTO pembayaranTagihan VALUES (9, "Biaya Pengadaan Buku", "2024-01-09", "Rp 700.000");  INSERT INTO pembayaranTagihan VALUES (10, "Biaya Teknologi & Akses Internet", "2024-01-10", "Rp 500.000"); |

1. **Penjelasan**

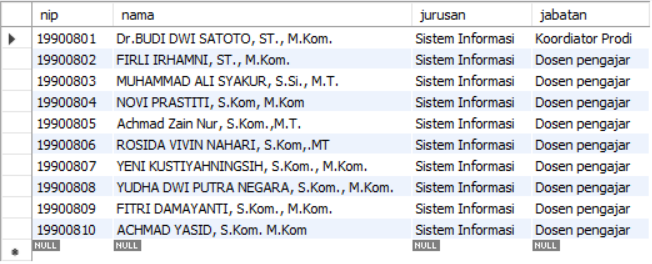
Kode di atas digunakan untuk membuat tabel pembayaran tagihan dan mengisi data sebanyak 10 data. Untuk membuat tabel pembayaran tagihan menggunakan perintah CREATE TABLE pembayaranTagihan (namaKolom1, namaKolom2, dst.). Tabel di atas memiliki kolom yang terdiri dari id\_pembayaran sebagai primary key, jenis tagihan, tanggal tagihan, dan total tagihan. Selanjutnya untuk mengisi data pada setiap kolom menggunakan perintah INSERT INTO.

## **Hasil**

### **Tabel Mahasiswa**



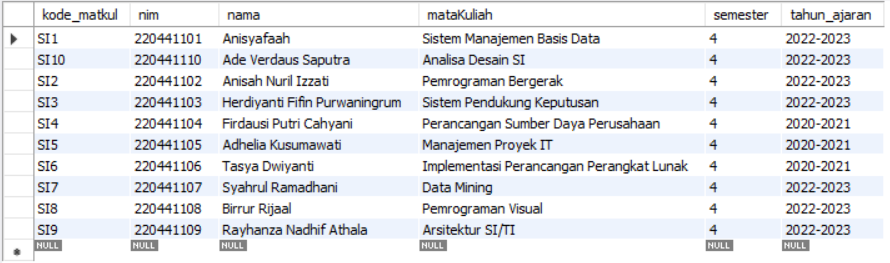
### **Tabel Dosen**



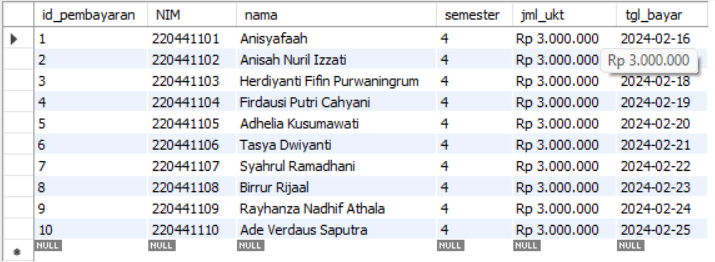
### **Tabel Mata Kuliah**



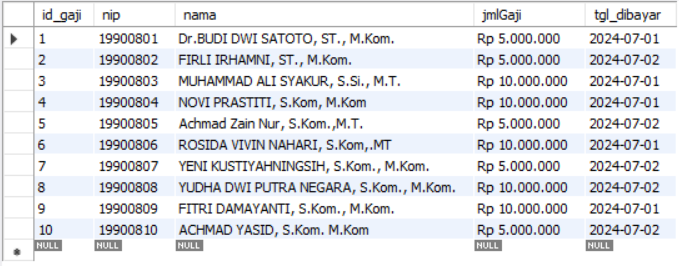
### **Tabel KRS**



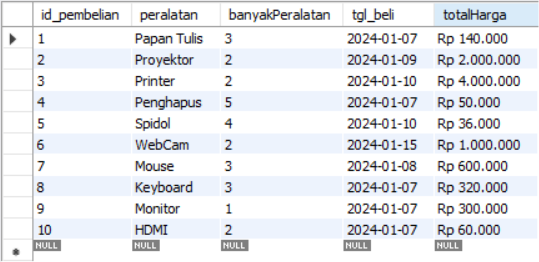
### **Tabel Pembayaran UKT**



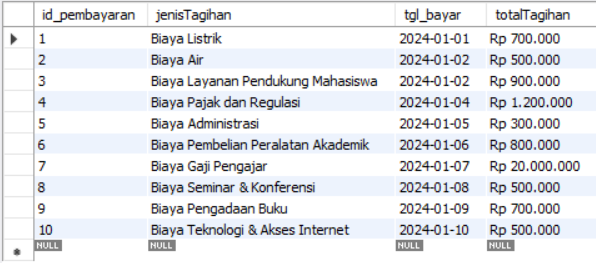
### **Tabel Gaji Dosen**



### **Tabel Pembelian Peralatan**



### **Tabel Pembayaran Tagihan**



# **BAB V PENUTUP**

## **Analisa**

Dari hasil praktikum, praktikan menganalisa bahwa Data Definition Language (DDL) berfungsi untuk mendefinisikan struktur database, termasuk tabel, kolom, indeks, dan constraint. Dengan DDL, seorang pengembang dapat membuat tabel dengan menentukan tipe data yang tepat untuk setiap kolom, serta menambahkan batasan ke tabel untuk menjaga integritas data. Misalnya, dengan menetapkan kunci primer dan kunci asing, pengembang dapat memastikan bahwa data yang dimasukkan ke dalam tabel sesuai dengan aturan yang telah ditetapkan. Selain itu,

Sementara itu, Data Manipulation Language (DML) memungkinkan pengembang untuk memanipulasi data yang disimpan dalam database. Melalui DML, pengembang dapat melakukan operasi seperti insert, update, delete, dan select data. DML memungkinkan pengguna untuk memperbarui data yang ada, menghapus data yang tidak diperlukan, serta mengekstrak informasi yang relevan dari database. Dengan menggunakan perintah SELECT, pengembang dapat mengambil data dari satu atau beberapa tabel berdasarkan kriteria tertentu, yang memungkinkan untuk menganalisis dan memvisualisasikan data sesuai kebutuhan. Selain itu, terdapat beberapa perintah lainnya seperti INSERT untuk menambah record, DELETE untuk menghapus, dan UPDATE untuk memperbarui record.

## **Kesimpulan**

1. Data Definition Language (DDL) merupakan bagian penting dalam pengelolaan database yang berkaitan dengan definisi struktur database.
2. Data Manipulation Language (DML) berkaitan dengan manipulasi data yang memungkinkan pengguna untuk menyisipkan, mengubah, menghapus, dan memperoleh data dari tabel dalam database.
3. Tugas praktikum dalam modul ini yaitu membuat database, tabel, beserta isi datanya agar dapat lebih memahami apa itu DDL dan DML.