Laporan Praktikum II

Pemrograman Database SQL

Muhammad Iqbal/22103001003

[mhmmdiqbal20210@gamil.com](mailto:mhmmdiqbal20210@gamil.com)

Jurusan Teknik Komputer

Institut Teknologi dan Sains Nahdlatul Ulama Kalimantan

***Abstrak***— Basis data adalah sistem penyimpanan data terstruktur yang merepresentasikan realitas. Dalam praktikum ini, kita membahas tentang dasar-dasar pemrograman database SQL, yang merupakan bahasa yang digunakan untuk mengelola basis data.

Basis data memiliki beberapa manfaat, seperti kemudahan pengelolaan, pencarian, dan penyimpanan data. Dengan menggunakan SQL, kita dapat membuat tabel untuk mengorganisasi data, memasukkan data baru, memperbarui data yang ada, menghapus data yang tidak diperlukan, dan melakukan query data untuk mengambil informasi yang diperlukan.

Namun, penggunaan basis data juga memiliki biaya dan memerlukan keahlian khusus untuk digunakan dan dikelola. Dalam laporan praktikum ini, kita belajar tentang dasar-dasar pemrograman database SQL, sehingga kita dapat memahami cara menggunakan SQL untuk mengelola basis data dengan efisien dan efektif.

Praktikum ini menggunakan MySQL sebagai sistem manajemen basis data (DBMS) pilihan.

***Kata kunci : Basis Data, Command Prompt, SQL***

1. **Pendahuluan**
2. **Definisi**

* Basis data: Kumpulan data terorganisir yang mewakili fakta dunia nyata (angka, huruf, simbol, gambar, bunyi).
* Berasal dari kata "basis" (tempat berkumpul) dan "data" (representasi fakta).
* Disimpan dalam media elektronik (file, tabel, arsip).
* Komponen utama: tabel.

1. **Kriteria**

* Berorientasi pada data.
* Digunakan oleh berbagai pengguna/program tanpa perubahan.
* Mudah berkembang (volume/struktur).
* Memenuhi kebutuhan sistem baru.
* Digunakan dengan cara berbeda.
* Kerangkapan data minimal.

1. **Manfaat**

* Pengelolaan data mudah dan cepat.
* Penyimpanan, perubahan, dan pemanggilan data mudah dan cepat.
* Penggunaan ruang penyimpanan efisien.
* Menjamin keakuratan data.
* Keamanan, keselarasan data, dan kebersamaan pemakaian terjamin.
* Standarisasi, pemilahan, dan penyesuaian data sesuai kebutuhan.

1. **Kelebihan:**

* Meningkatkan kemandirian dan konsistensi data.
* Meningkatkan aksesibilitas, keamanan, dan integritas data.
* Memungkinkan penggunaan data bersama-sama dan informasi lebih banyak.

1. **Kekurangan**

* Biaya mahal: pembelian/perawatan hardware dan software, penyimpanan, jaringan, dll.
* Pemahaman mendalam tentang basis data diperlukan untuk menghindari kerugian.
* Konversi ke sistem basis data baru mahal: pelatihan staf, perekrutan DBA.

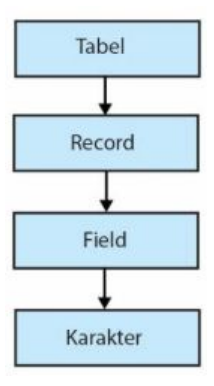
1. **PEMBAHASAN**
2. ***Database Management System* (DBMS)**

Di era digital ini, data menjadi aset penting bagi organisasi dan individu. Untuk mengelola data secara efektif dan efisien, diperlukan sistem yang handal. Di sinilah Sistem Manajemen Basis Data (DBMS) berperan. DBMS adalah perangkat lunak yang dirancang khusus untuk mengelola dan mengatur basis data.

DBMS memungkinkan pengguna untuk membuat, mengakses, memperbarui, dan mengelola basis data dengan mudah dan terstruktur. Pengguna dapat menyimpan data dalam berbagai format, seperti angka, huruf, gambar, dan video. DBMS juga menyediakan berbagai fitur untuk memastikan keamanan, integritas, dan konsistensi data.

1. **Tingkatan Data Dalam Database Relasi**

Dalam suatu sistem database relasi, data yang tersimpan dalam DBMS mempunyai tingkatan-tingkatannya seperti gambar berikut.

****

1. ***Character***

Merupakan bagian terkecil dalam database, dapat berupa karakter numerik (angka 0 s.d 9), huruf (A – Z, a – z) ataupun karakter-karakter khusus, seperti \*, &. %, # dan lain-lain.

1. ***Field (Attribute)***

*Field* merupakan bagian dari *record* yang menunjukkan suatu item data yang sejenis, seperti *field* nama, *field* NIM, dan sebagainya. Setiap *field* memiliki nama dan tipe data tertentu, dan isi dari *field* disebut Data *Value*. Dalam tabel sebuah database, *field* ini juga disebut kolom.

1. ***Record (Tupple)***

*Tupple/Record* adalah kumpulan data *value* dari *attribute* yang berkaitan, yang menjelaskan sebuah entity secara lengkap. Sebagai contoh, *record entity* mahasiswa terdiri dari data *value field* nomor telepon genggam, nama, jurusan, dan alamat per-barisnya. Dalam *database*, *record* juga dikenal sebagai baris.

1. ***Table (Entity)***

*Entity* adalah sesuatu yang dapat diidentifikasi dalam sistem database, seperti objek, orang, tempat, kejadian, atau konsep. Dalam sistem database akademik, entity meliputi mahasiswa, dosen, mata kuliah, dan lainnya. Dalam konteks ini, entity sering disamakan dengan tabel, di mana data diatur dalam bentuk baris dan kolom. Tabel memiliki sifat-sifat seperti tidak ada record yang sama, urutan record tidak penting, setiap field memiliki nama unik, dan setiap field memiliki tipe data tertentu.

1. **Pembuatan Database Baru dan Penggunaan SQL**

Dalam sistem manajemen basis data (DBMS), langkah pertama dalam pengelolaan data adalah membuat *database* baru. Setelah *database* dibuat, SQL (*Structured Query Languag*e) digunakan untuk mendefinisikan struktur, mengelola data, dan mengatur keamanan dalam *database* tersebut. Pada materi ini, kita akan membahas pembuatan *database* baru dan penggunaan SQL DDL, DML, dan DCL untuk mengelola *database*.

1. **Tipe Data**
2. Tipe Data Numerik:
3. INTEGER atau INT: Digunakan untuk menyimpan nilai numerik bulat. Ukuran bervariasi tergantung pada platform, tetapi biasanya 4 byte.

Contoh: ID karyawan (INT(10)), jumlah produk (INT(6)).

1. *DECIMAL* atau *NUMERIC*: Digunakan untuk menyimpan nilai numerik dengan desimal. Ukuran tergantung pada presisi dan skala yang ditentukan. Contoh: Harga produk (*DECIMAL*(10, 2)), persentase diskon (*DECIMAL*(5, 2)).
2. *FLOAT* atau *DOUBLE*: Digunakan untuk menyimpan nilai numerik floating-point. Ukuran bervariasi tergantung pada platform, tetapi biasanya 8 byte. Contoh: Koordinat GPS (*FLOAT*), berat dalam kilogram (*DOUBLE*).
3. Tipe Data Karakter:

* *CHAR*(n): Digunakan untuk menyimpan string karakter dengan panjang tetap. Ukuran tetap sesuai dengan 'n'. Contoh: Kode negara (*CHAR*(2)), kode pos (*CHAR*(5)).
* *VARCHAR*(n): Digunakan untuk menyimpan string karakter dengan panjang variabel. Ukuran maksimum adalah 'n'. Contoh: Nama karyawan (*VARCHAR*(50)), alamat pengiriman (*VARCHAR*(100)).
* *TEXT*: Digunakan untuk menyimpan string karakter dengan panjang yang sangat panjang atau tidak terbatas. Ukuran maksimum bervariasi tergantung pada database. Contoh: Deskripsi produk (*TEXT*), catatan karyawan (*TEXT*).

1. Tipe Data Tanggal dan Waktu:

* *DATE*: Digunakan untuk menyimpan tanggal. Ukuran tetap, biasanya 3 byte. Contoh: Tanggal lahir karyawan (*DATE*), tanggal pembelian produk (*DATE*).
* *TIME*: Digunakan untuk menyimpan waktu. Ukuran tetap, biasanya 3-5 byte. Contoh: Waktu mulai shift kerja (*TIME*), waktu pengiriman pesanan (*TIME*).
* *DATETIME* atau *TIMESTAMP*: Digunakan untuk menyimpan informasi tanggal dan waktu. Ukuran bervariasi, biasanya 8 byte. Contoh: Waktu dan tanggal pembuatan akun (*DATETIME*), tanggal dan waktu terakhir kali data diperbarui (*TIMESTAMP*).

1. Tipe Data Boolean:

* *BOOLEAN*: Digunakan untuk menyimpan nilai kebenaran. Ukuran biasanya 1 byte. Contoh: Status aktif/non-aktif akun (*BOOLEAN*), status pembayaran (*BOOLEAN*).

1. Tipe Data Binary:

* *BINARY*: Digunakan untuk menyimpan data biner dengan panjang tetap. Ukuran tetap sesuai dengan 'n'. Contoh: Gambar profil dalam format biner (*BINARY*(64)).
* *VARBINARY*: Digunakan untuk menyimpan data biner dengan panjang variabel. Ukuran maksimum adalah 'n'. Contoh: Lampiran email (*VARBINARY*(1024)), file PDF (*VARBINARY*(4096)).

1. Tipe Data Lainnya:

* ENUM: Digunakan untuk menyimpan satu nilai dari kumpulan nilai yang telah ditentukan sebelumnya. Ukuran bervariasi tergantung pada jumlah opsi yang didefinisikan. Contoh: Status pesanan ('Menunggu', 'Dalam Pengiriman', 'Selesai') (ENUM ('Menunggu', 'Dalam Pengiriman', 'Selesai')).
* JSON: Digunakan untuk menyimpan data dalam format JSON. Ukuran bervariasi tergantung pada data yang disimpan. Contoh: Data pengguna dalam format JSON ({ "nama": "John", "usia": 30 }) (JSON).

1. **Data *Definition Language* (DDL)**

Di jantung setiap basis data yang kokoh terletak Data Definition Language (DDL). DDL adalah bahasa pemrograman khusus yang digunakan untuk mendefinisikan struktur dan mengelola objek dalam basis data. DDL bagaikan arsitek basis data, merancang dan membangun fondasi yang akan menampung informasi penting.

DDL berfokus pada operasi yang berkaitan dengan struktur data, seperti:

* Membuat tabel: DDL memungkinkan Anda membuat tabel baru untuk menyimpan data yang berbeda, menentukan kolom dan jenis data yang akan disimpan di dalamnya.
* Mengubah tabel: DDL memungkinkan Anda memodifikasi struktur tabel yang ada, seperti menambahkan atau menghapus kolom, mengubah jenis data, atau menetapkan batasan data.
* Menghapus tabel: DDL memungkinkan Anda menghapus tabel beserta datanya jika tidak lagi diperlukan.

DDL juga berperan penting dalam mengelola indeks dan konstrain basis data. Indeks meningkatkan kecepatan pencarian data, sedangkan konstrain memastikan integritas dan validitas data. DDL memungkinkan Anda membuat, memodifikasi, dan menghapus indeks dan konstrain sesuai kebutuhan.

**VI. Hasil Dan Pembahasan**

Pada praktikum kali ini, peserta menggunakan aplikasi *Command Prompt* pada Windows dengan melakukan beberapa perintah yaitu :

Sebuah gambar berisi teks, cuplikan layar, Font

Deskripsi dibuat secara otomatis

Perintah ini terdiri dari beberapa bagian, yaitu :

* mysql: Nama program mysql.
* -u root: Opsi untuk menentukan nama pengguna. Dalam kasus ini, nama pengguna adalah root.
* -p: Opsi untuk meminta kata sandi.
* Enter password: Prompt untuk memasukkan kata sandi pengguna MySQL.
* Welcome to the MariaDB monitor. Commands end with ; or \g.: Pesan selamat datang dari monitor MariaDB. Pesan ini menunjukkan bahwa Anda telah berhasil terhubung ke server MySQL.
* Your MariaDB connection id is 18: Informasi ID koneksi Anda ke server MySQL.
* Server version: 10.4.27-MariaDB
* mariadb.org binary distribution: Informasi versi server MySQL yang Anda gunakan.
* Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others. : Informasi hak cipta server MySQL.
* Type 'help;' or '\h' for help. Type '\c' to clear the current input statement. : Petunjuk untuk mendapatkan bantuan dan pembersihan input saat ini.

1. Buat database baru bernama 'perkuliahan' Buat beberapa tabel berikut pada database tersebut:

* mahasiswa,
* dosen, dan
* matakuliah

MariaDB [(none)]> CREATE DATABASE perkuliahan1; Perintah ini akan membuat database baru dengan nama "perkuliahan1". Database ini berfungsi sebagai wadah untuk menyimpan berbagai tabel yang terkait dengan perkuliahan.

MariaDB [(none)]> use perkuliahan1; Perintah ini memilih database "perkuliahan1" sebagai database yang aktif. Artinya, semua perintah SQL selanjutnya akan dijalankan terhadap database ini.

MariaDB [perkuliahan1]> CREATE TABLE perkuliahan1 ( mahasiswa varchar(50) PRIMARY key, dosen varchar(50), matakuliah varchar(50) ); Perintah ini membuat tabel baru dengan nama "perkuliahan1" di dalam database "perkuliahan1". Tabel ini memiliki tiga kolom:

* **mahasiswa**: Kolom ini menyimpan nama mahasiswa dengan tipe data varchar(50), yang berarti dapat menampung hingga 50 karakter. Kolom ini juga ditetapkan sebagai PRIMARY KEY, yang berarti setiap nilai di kolom ini harus unik dan tidak boleh duplikat.
* **dosen**: Kolom ini menyimpan nama dosen dengan tipe data varchar(50), yang berarti dapat menampung hingga 50 karakter.
* **matakuliah**: Kolom ini menyimpan nama mata kuliah dengan tipe data varchar(50), yang berarti dapat menampung hingga 50 karakter.

Tabel ini akan digunakan untuk menyimpan informasi tentang mahasiswa, dosen, dan mata kuliah yang terkait dengan perkuliahan.

2. Tabel mahasiswa beriksikan beberapa field/kolom sebagai berikut:

* nim int(20),
* nama\_mahasiswa varchar(100),
* jurusan varchar(50)

MariaDB [perkuliahan1]> CREATE DATABASE mahasiswa; Perintah ini akan membuat database baru dengan nama "mahasiswa". Database ini berfungsi sebagai wadah untuk menyimpan berbagai tabel yang terkait dengan mahasiswa.

MariaDB [perkuliahan1]> use mahasiswa; Perintah ini memilih database "mahasiswa" sebagai database yang aktif. Artinya, semua perintah SQL selanjutnya akan dijalankan terhadap database ini.

MariaDB [mahasiswa]> CREATE TABLE mahasiswa ( nim int(20) PRIMARY key, nama\_mahasiswa varchar(100), jurusan varchar(50)); Perintah ini membuat tabel baru dengan nama "mahasiswa" di dalam database "mahasiswa". Tabel ini memiliki tiga kolom:

* **nim**: Kolom ini menyimpan nomor induk mahasiswa (NIM) dengan tipe data int(20), yang berarti dapat menampung hingga 20 digit angka. Kolom ini juga ditetapkan sebagai PRIMARY KEY, yang berarti setiap nilai di kolom ini harus unik dan tidak boleh duplikat.
* **nama\_mahasiswa**: Kolom ini menyimpan nama lengkap mahasiswa dengan tipe data varchar(100), yang berarti dapat menampung hingga 100 karakter.
* **jurusan**: Kolom ini menyimpan jurusan mahasiswa dengan tipe data varchar(50), yang berarti dapat menampung hingga 50 karakter.

Tabel ini akan digunakan untuk menyimpan informasi tentang NIM, nama lengkap, dan jurusan mahasiswa.

3. Tabel dosen beriksikan beberapa field/kolom sebagai berikut:

* nidn int(20), -
* nama\_dosen varchar(50),
* Jurusan varchar(50)

MariaDB [mahasiswa]> CREATE DATABASE dosen; Perintah ini akan membuat database baru dengan nama "dosen". Database ini berfungsi sebagai wadah untuk menyimpan berbagai tabel yang terkait dengan dosen.

MariaDB [mahasiswa]> use dosen; Perintah ini memilih database "dosen" sebagai database yang aktif. Artinya, semua perintah SQL selanjutnya akan dijalankan terhadap database ini.

MariaDB [dosen]> CREATE TABLE dosen ( nidn int(20) PRIMARY key, nama\_dosen varchar(50), jurusan varchar(50));. Perintah ini membuat tabel baru dengan nama "dosen" di dalam database "dosen". Tabel ini memiliki tiga kolom:

* **nidn**: Kolom ini menyimpan nomor induk dosen nasional (NIDN) dengan tipe data int(20), yang berarti dapat menampung hingga 20 digit angka. Kolom ini juga ditetapkan sebagai PRIMARY KEY, yang berarti setiap nilai di kolom ini harus unik dan tidak boleh duplikat.
* **nama\_dosen**: Kolom ini menyimpan nama lengkap dosen dengan tipe data varchar(50), yang berarti dapat menampung hingga 50 karakter.
* **jurusan**: Kolom ini menyimpan jurusan dosen dengan tipe data varchar(50), yang berarti dapat menampung hingga 50 karakter.

Tabel ini akan digunakan untuk menyimpan informasi tentang NIDN, nama lengkap, dan jurusan dosen.

4. Tabel matakuliah beriksikan beberapa field/kolom sebagai berikut:

* kode\_mk varchar(10),
* nama\_mk varchar(50),
* sks int (nilai yang dimasukkan di sks dari 1-3)

MariaDB [dosen]> CREATE DATABASE matakuliah;. Perintah ini akan membuat database baru dengan nama "matakuliah". Database ini berfungsi sebagai wadah untuk menyimpan berbagai tabel yang terkait dengan mata kuliah.

MariaDB [dosen]> use matakuliah;. Perintah ini memilih database "matakuliah" sebagai database yang aktif. Artinya, semua perintah SQL selanjutnya akan dijalankan terhadap database ini.

MariaDB [matakuliah]> CREATE TABLE matakuliah ( kode\_mk varchar(10) PRIMARY key, nama\_mk varchar(50), sks int(1) );. Perintah ini membuat tabel baru dengan nama "matakuliah" di dalam database "matakuliah". Tabel ini memiliki tiga kolom:

* **kode\_mk**: Kolom ini menyimpan kode mata kuliah dengan tipe data varchar(10), yang berarti dapat menampung hingga 10 karakter. Kolom ini juga ditetapkan sebagai PRIMARY KEY, yang berarti setiap nilai di kolom ini harus unik dan tidak boleh duplikat.
* **nama\_mk**: Kolom ini menyimpan nama lengkap mata kuliah dengan tipe data varchar(50), yang berarti dapat menampung hingga 50 karakter.
* **sks**: Kolom ini menyimpan jumlah Satuan Kredit Semester (SKS) mata kuliah dengan tipe data int(1), yang berarti hanya dapat menampung nilai 1 sampai 9.

Tabel ini akan digunakan untuk menyimpan informasi tentang kode, nama, dan SKS mata kuliah.

5. Isikan masing-masing tabel minimal 3 data awal menggunakan perintah insert

* untuk penambahan data tabel Mahasiswa.
* MariaDB [matakuliah]> use mahasiswa;. Perintah ini memilih database "mahasiswa" sebagai database yang aktif. Artinya, semua perintah SQL selanjutnya akan dijalankan terhadap database ini.
* MariaDB [mahasiswa]> insert into mahasiswa (nim, nama\_mahasiswa, jurusan) values (22103001003, 'MUHAMMAD IQBAL', 'Teknik kompute');
* MariaDB [mahasiswa]> insert into mahasiswa (nim, nama\_mahasiswa, jurusan) values (03, 'MUHAMMAD FARHAN', 'Teknik kompute');
* MariaDB [mahasiswa]> insert into mahasiswa (nim, nama\_mahasiswa, jurusan) values (04, 'MUHAMMAD AGUS', 'Teknik kompute');

Perintah ini memasukkan data ke dalam tabel "mahasiswa". Setiap baris perintah INSERT INTO mewakili satu baris data yang akan dimasukkan. Kolom-kolom yang disebutkan dalam perintah harus sesuai dengan kolom yang ada di tabel. Nilai-nilai data yang dimasukkan harus diketik di dalam tanda kurung.

**Penjelasan Kolom:**

* **nim**: Kolom ini menyimpan nomor induk mahasiswa (NIM) dengan tipe data int(20). Dalam perintah SQL, Anda memasukkan nilai NIM 22103001003, 03, dan 04.
* **nama\_mahasiswa**: Kolom ini menyimpan nama lengkap mahasiswa dengan tipe data varchar(100). Dalam perintah SQL, Anda memasukkan nama "MUHAMMAD IQBAL", "MUHAMMAD FARHAN", dan "MUHAMMAD AGUS".
* **jurusan**: Kolom ini menyimpan jurusan mahasiswa dengan tipe data varchar(50). Dalam perintah SQL, Anda memasukkan jurusan "Teknik kompute" untuk semua mahasiswa.
* Untuk penambahan data tabel dosen.
* MariaDB [mahasiswa]> USE dosen;. Perintah ini memilih database "dosen" sebagai database yang aktif. Artinya, semua perintah SQL selanjutnya akan dijalankan terhadap database ini.
* MariaDB [dosen]> insert into dosen (nidn, nama\_dosen, jurusan) values (01, 'MUHAMMAD BAGAS', 'Teknik kompute');
* MariaDB [dosen]> insert into dosen (nidn nama\_dosen, jurusan) values (02, 'PANJUL', 'Teknik kompute');
* MariaDB [dosen]> insert into dosen (nidn, nama\_dosen, jurusan) values (03, 'JALI', 'Teknik kompute');

Perintah ini memasukkan data ke dalam tabel "dosen". Setiap baris perintah INSERT INTO mewakili satu baris data yang akan dimasukkan. Kolom-kolom yang disebutkan dalam perintah harus sesuai dengan kolom yang ada di tabel. Nilai-nilai data yang dimasukkan harus diketik di dalam tanda kurung.

**Penjelasan Kolom:**

* **nidn**: Kolom ini menyimpan nomor induk dosen nasional (NIDN) dengan tipe data int(20). Dalam perintah SQL, Anda memasukkan nilai NIDN 01, 02, dan 03.
* **nama\_dosen**: Kolom ini menyimpan nama lengkap dosen dengan tipe data varchar(50). Dalam perintah SQL, Anda memasukkan nama "MUHAMMAD BAGAS", "PANJUL", dan "JALI".
* **jurusan**: Kolom ini menyimpan jurusan dosen dengan tipe data varchar(50). Dalam perintah SQL, Anda memasukkan jurusan "Teknik kompute" untuk semua dosen.
* Untuk penambahan data tabel matakuliah.
* MariaDB [dosen]> USE matakuliah;. Perintah ini memilih database "matakuliah" sebagai database yang aktif. Artinya, semua perintah SQL selanjutnya akan dijalankan terhadap database ini.
* MariaDB [matakuliah]> insert into matakuliah (kode\_mk, nama\_mk, sks) values ('MT', 'jaringan komputer', 3);
* MariaDB [matakuliah]> insert into matakuliah (kode\_mk, nama\_mk, sks) values ('AM', 'pemograman', 3);
* MariaDB [matakuliah]> insert into matakuliah (kode\_mk, nama\_mk, sks) values ('AB', 'fisika', 2);

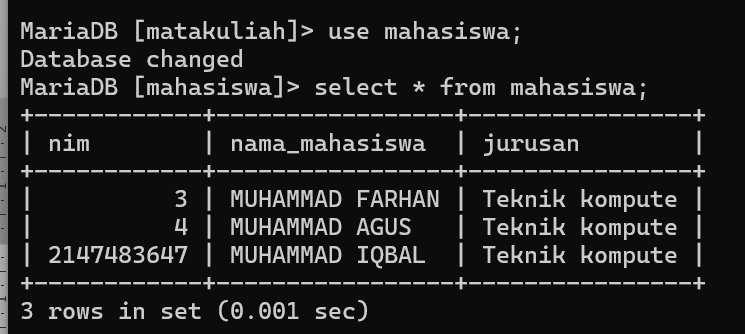
Perintah ini memasukkan data ke dalam tabel "matakuliah". Setiap baris perintah INSERT INTO mewakili satu baris data yang akan dimasukkan. Kolom-kolom yang disebutkan dalam perintah harus sesuai dengan kolom yang ada di tabel. Nilai-nilai data yang dimasukkan harus diketik di dalam tanda kurung.

**Penjelasan Kolom:**

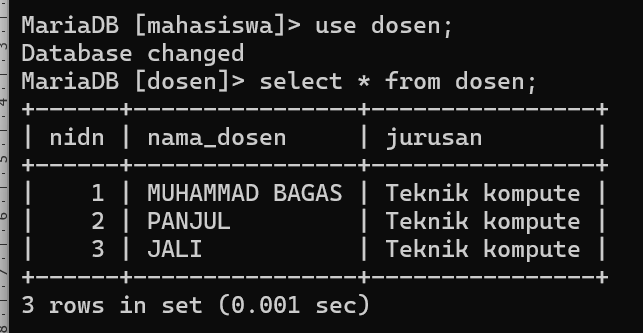
* **kode\_mk**: Kolom ini menyimpan kode mata kuliah dengan tipe data varchar(10). Dalam perintah SQL, Anda memasukkan kode "MT", "AM", dan "AB".
* **nama\_mk**: Kolom ini menyimpan nama lengkap mata kuliah dengan tipe data varchar(50). Dalam perintah SQL, Anda memasukkan nama "jaringan komputer", "pemograman", dan "fisika".
* **sks**: Kolom ini menyimpan jumlah Satuan Kredit Semester (SKS) mata kuliah dengan tipe data int(1). Dalam perintah SQL, Anda memasukkan SKS 3 untuk "jaringan komputer" dan "pemograman", dan SKS 2 untuk "fisika".

6. Panggil tabel mahasiswa, dosen, dan matakuliah menggunakan perintah select

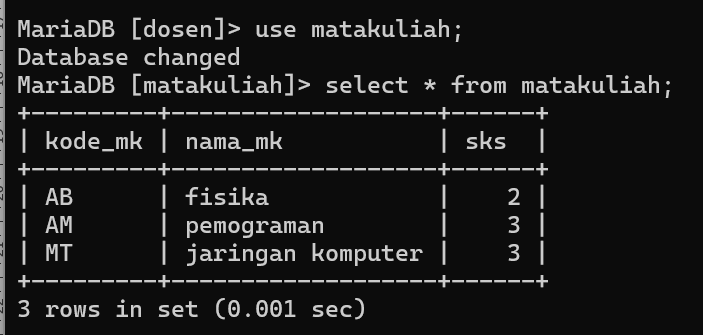
* tabel mahasiswa



* tabel dosen

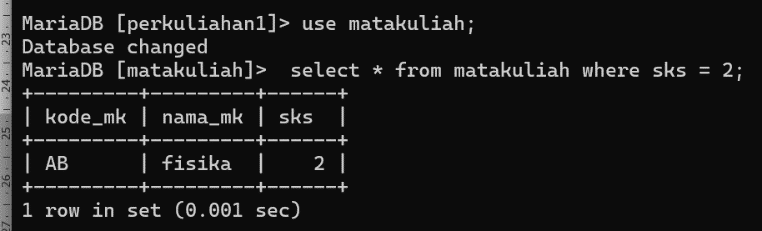


* tabel matakuliah



7. Panggil tabel matakuliah dengan kondisi sks tertentu.

* Cara memanggil tabel matakuliah dengan kondisi sks tertentu

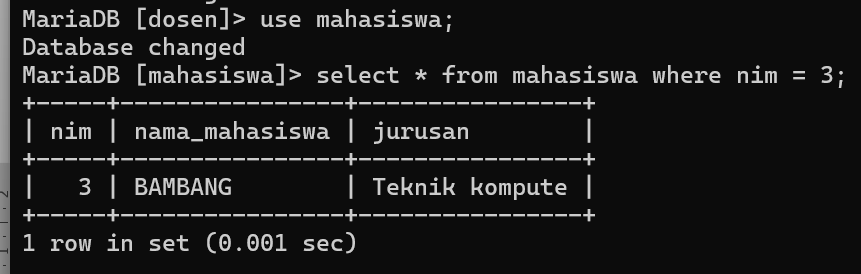


8. Rubah salah satu data nama pada tabel mahasiswa dengan nama baru dengan menggunakan perintah update, kemudian tampilkan hanya data yang telah dirubah tersebut

MariaDB [matakuliah]> use mahasiswa;. Perintah ini memilih database "mahasiswa" sebagai database yang aktif. Artinya, semua perintah SQL selanjutnya akan dijalankan terhadap database ini.

MariaDB [mahasiswa]> update mahasiswa set nama\_mahasiswa = 'BAMBANG' WHERE nim = 3;. Perintah ini memperbarui data nama mahasiswa di tabel "mahasiswa". Perintah ini memiliki dua bagian:

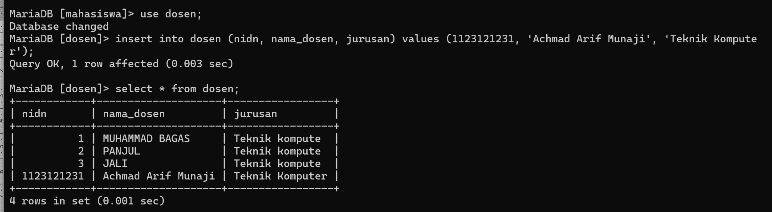
* **UPDATE mahasiswa**: Bagian ini menentukan tabel mana yang akan diperbarui. Dalam hal ini, tabel yang diperbarui adalah "mahasiswa".
* **SET nama\_mahasiswa = 'BAMBANG' WHERE nim = 3**: Bagian ini menentukan data mana yang akan diperbarui dan kondisi yang harus dipenuhi untuk memperbarui data tersebut. Dalam hal ini:
  + **SET nama\_mahasiswa = 'BAMBANG'**: Bagian ini menetapkan nilai baru untuk kolom "nama\_mahasiswa", yaitu "BAMBANG".
  + **WHERE nim = 3**: Bagian ini menentukan kondisi yang harus dipenuhi untuk memperbarui data. Dalam hal ini, data hanya akan diperbarui untuk mahasiswa dengan NIM 3.



9. Tambahkan 1 data baru pada tabel dosen berikut:

* nidn: 1123121231
* nama\_dosen: Achmad Arif Munaji
* Jurusan: Teknik Komputer

Menambahkan 1 data baru pada tabel dosen



**V. Kesimpulan**

Praktikum ini telah memberikan pengenalan dasar tentang pemrograman database SQL. Praktikum ini telah mengajarkan cara membuat tabel, memasukkan data, memperbarui data, menghapus data, dan melakukan query data. Peserta praktikum diharapkan dapat menggunakan SQL untuk mengelola data dalam database MySQL.