Laporan Praktikum I

Pemrograman Database SQL

Muhammad Yusron Nashrulloh/22103001005

[myusron434@gamil.com](mailto:myusron434@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 dasar-dasar pemrograman database SQL. Basis data memiliki manfaat seperti kemudahan pengelolaan, pencarian, dan penyimpanan data. Dengan SQL, kita dapat membuat tabel, memasukkan, memperbarui, dan menghapus data, serta melakukan query untuk mengambil informasi yang diperlukan. Penggunaan basis data memerlukan biaya dan keahlian khusus. Praktikum ini membantu kita memahami cara menggunakan SQL secara efisien dan efektif dalam mengelola basis data.

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

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

1. **Pendahuluan**

Basis data adalah kumpulan data terorganisir yang merepresentasikan fakta dunia nyata. Dalam basis data, data disimpan dalam media elektronik seperti file dan tabel. Basis data memiliki manfaat seperti pengelolaan data yang mudah dan cepat, penyimpanan yang efisien, dan keamanan data terjamin. Namun, penggunaan basis data juga memiliki kekurangan, seperti biaya mahal dan kebutuhan pemahaman mendalam tentang basis data.

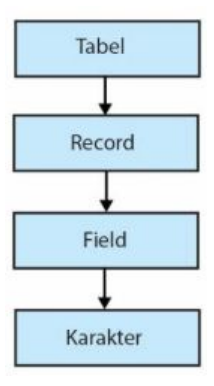
1. **PEMBAHASAN**
2. ***Database Management System* (DBMS)**

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

Dengan DBMS, pengguna dapat dengan mudah membuat, mengakses, memperbarui, dan mengelola basis data. Data dapat disimpan dalam berbagai format, seperti angka, huruf, gambar, dan video. DBMS juga menyediakan fitur-fitur penting seperti keamanan, integritas, dan konsistensi data. Dengan demikian, DBMS memainkan peran kunci dalam mengelola data secara terstruktur dan terorganisir.

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)**

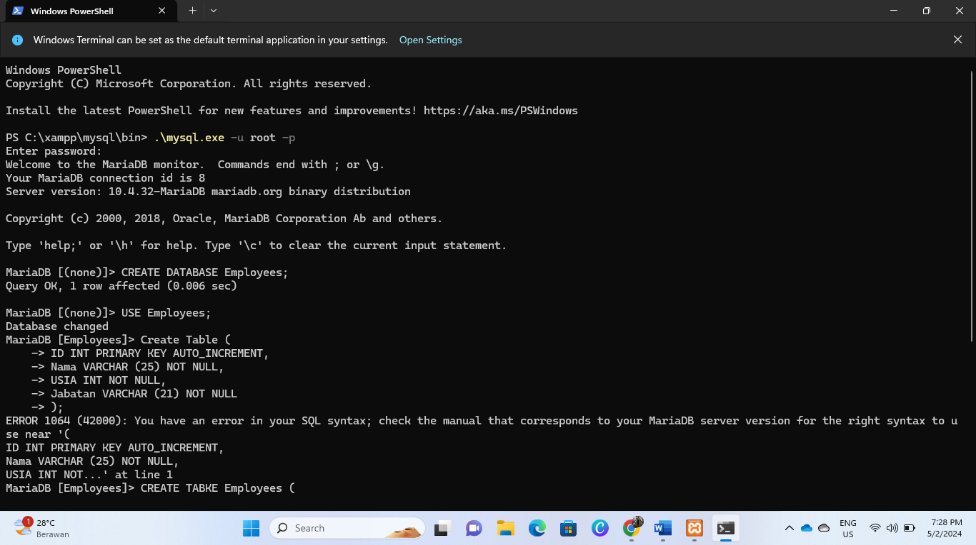
DDL (Data Definition Language) adalah bahasa pemrograman khusus yang digunakan untuk mendefinisikan struktur dan mengelola objek dalam basis data. DDL berperan sebagai arsitek basis data, merancang dan membangun fondasi yang menampung informasi penting. DDL memungkinkan pembuatan tabel baru, modifikasi struktur tabel, penghapusan tabel, serta pengelolaan indeks dan konstrain basis data. Dengan DDL, pengelolaan struktur dan objek dalam basis data menjadi lebih mudah dan sesuai kebutuhan.

**VI. Hasil Dan Pembahasan**

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

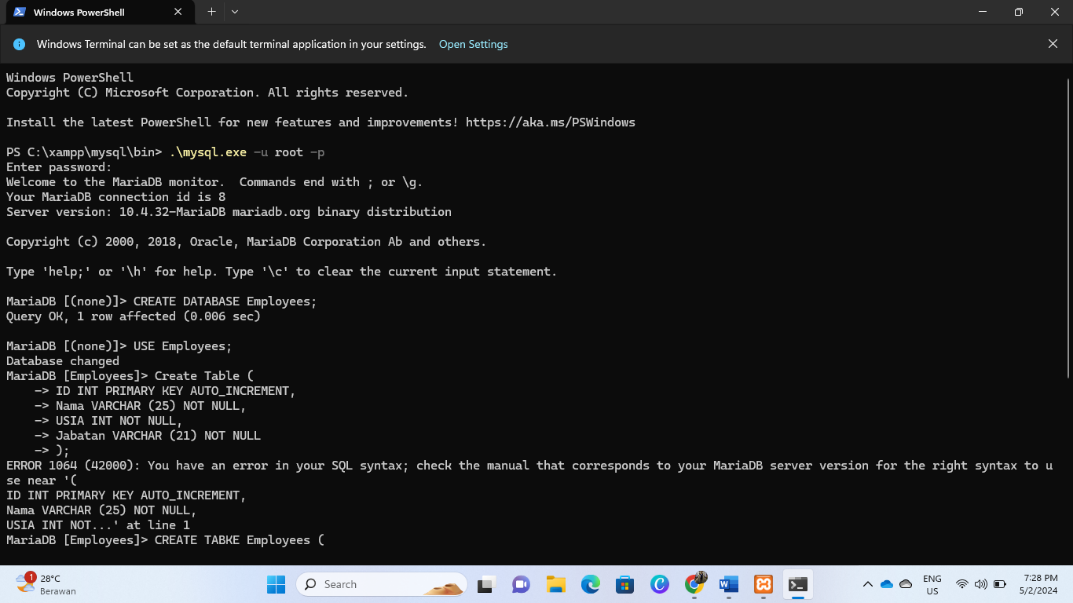
1. Buatlah sebuah tabel dengan nama 'Employees' yang memiliki kolom-kolom berikut:

* ID (integer)
* Nama (varchar)
* Usia (integer)
* Jabatan (varchar)



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.



Kueri ini digunakan untuk membuat tabel baru dengan nama Employees dalam database MySQL.Tabel ini memiliki empat kolom:

* ID: Kolom bilangan bulat yang merupakan kunci utama tabel dan bertambah secara otomatis.
* Nama: Kolom VARCHAR dengan panjang maksimal 25 karakter yang menyimpan nama karyawan.
* Usia: Kolom bilangan bulat yang menyimpan usia karyawan.
* Jabatan: Kolom VARCHAR dengan panjang maksimum 21 karakter yang menyimpan jabatan karyawan.

Berikut rincian kuerinya:

* CREATE TABLE:Kata kunci ini memberitahu MySQL bahwa Anda ingin membuat tabel baru.
* Employeess: Ini adalah nama tabel yang Anda buat.
* ( : Karakter ini menandai awal dari definisi kolom.
* ID INT PRIMARY KEY: Ini mendefinisikan ID kolom. Ini adalah kolom bilangan bulat yang merupakan kunci utama tabel dan bertambah secara otomatis.
* , : Karakter ini memisahkan definisi kolom.
* Nama VARCHAR (25) NOT NULL : Ini mendefinisikan Nama kolom. Ini adalah kolom VARCHAR dengan panjang maksimum 25 karakter dan tidak boleh nol.
* Usia INT NOT NULL : Ini mendefinisikan Usia kolom. Ini adalah kolom bilangan bulat yang tidak boleh nol.
* Jabatan VARCHAR(21) NOT NULL : Ini mendefinisikan Jabatan kolom. Ini adalah kolom VARCHAR dengan panjang maksimum 21 karakter dan tidak boleh nol.
* ) : Karakter ini menandai akhir dari definisi kolom.
* ) ; : Karakter ini mengakhiri kueri.

1. Tambahkan kolom baru dengan nama 'Gaji' ke dalam tabel 'Employees' dengan tipe data 'decimal' (uang).

Sebuah gambar berisi teks, cuplikan layar, software, Software multimedia

Deskripsi dibuat secara otomatis

Perintah SQL ALTER TABLE digunakan untuk mengubah struktur tabel yang sudah ada. Dalam perintah ini, parameter Employees adalah nama tabel yang ingin diubah, dan ADD Gaji DECIMAL(10,2) adalah perintah untuk menambahkan kolom baru bernama Gaji dengan tipe data DECIMAL(10,2). Tipe data DECIMAL(10,2) berarti kolom ini dapat menyimpan nilai desimal dengan panjang maksimum 10 digit, termasuk 2 digit di belakang koma.

1. Hapus kolom 'Usia' dari tabel 'Employees'.

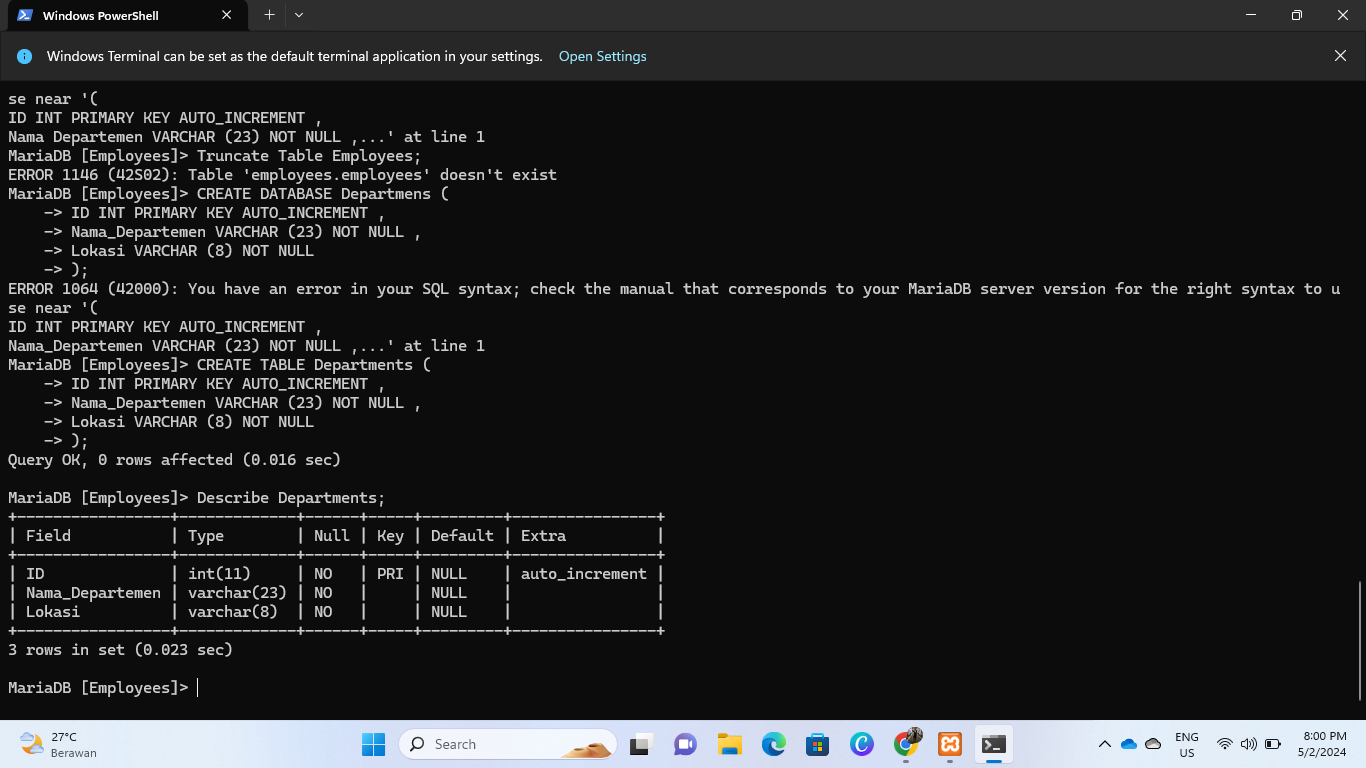
Sebuah gambar berisi teks, cuplikan layar, Font

Deskripsi dibuat secara otomatis

Perintah SQL ALTER TABLE digunakan untuk mengubah struktur tabel yang sudah ada. Dalam perintah ini, parameter Employees adalah nama tabel yang ingin diubah, dan DROP COLUMN Usia adalah perintah untuk menghapus kolom Usia.

1. Buatlah tabel baru dengan nama 'Department' yang memiliki kolom-kolom berikut:

* ID (integer)
* Nama\_Departemen (varchar)
* Lokasi (varchar)



Perintah SQL CREATE TABLE digunakan untuk membuat tabel baru dalam database. Perintah ini memiliki beberapa parameter, di antaranya:

* TABLE\_NAME : Nama tabel yang ingin dibuat.
* COLUMN\_DEFINITION: Definisi kolom-kolom dalam tabel.
* Dalam praktikum ini, parameter COLUMN\_DEFINITIONberisi definisi tiga kolom:
* ID: Kolom ID dengan tipe data integer dan merupakan kunci primer dengan kenaikan otomatis.
* Nama\_Departemen: Kolom Nama\_Departemen dengan tipe data varchar dengan panjang maksimum 23 karakter dan not null.
* Lokasi: Kolom Lokasi dengan tipe data varchar dengan panjang maksimal 8 karakter dan not null.

**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.