**MODUL II**

**DDL (*Data Definition Language)***

# TUJUAN

* 1. Memahami konsep dasar Data Definition Language (DDL)
  2. Memahami implementasi beberapa perintah Create dari DDL
  3. Memahami implementasi beberapa perintah Alter dari DDL

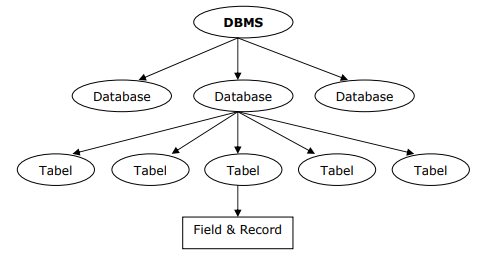
# DASAR TEORI

## Pengertian DDL

*Data Definition Language* (DDL) adalah kumpulan perintah SQL yang berkaitan dengan pembuatan, perubahan, dan penghapusan *database* maupun objek-objek yang terdapat di dalam database. Salah satu bentuk bahasa basis data yaitu *Data Definition Language* (DDL) yang digunakan untuk membuat, mengubah, serta menghapus basis data dan objek-objek yang diperlukan dalam basis data. Metode yang digunakan untuk menerjemahkan kalimat perintah dalam bahasa Indonesia ke *query* DDL adalah metode *Rule-Based.* Proses pada sistem yang dibangun memiliki dua tahap utama yaitu Preprocessing dan *Translasi.* Tahap Preprocessing terdiri dari *case folding, filtering, tokenizing* kata*, stemming*, dan *removing stopword*. Tahap *Translasi* terdiri dari deteksi kata kunci, tokenizing perintah, identifikasi perintah DDL, indentifikasi konten, dan penyusunan *query*

## Pengertian SQL

SQL atau *Structured Query Language* merupakan suatu bahasa (*language*) yang digunakan untuk mengakses *database*. SQL sering disebut juga sebagai *query*. Bahasa ini secara *de facto* merupakan bahasa standar yang digunakan dalam manajemen basis data relasional. Saat ini hampir semua server basis data yang ada mendukung bahasa ini untuk melakukan manajemen datanya. Sejarah SQL dimulai dari artikel seorang peneliti dari IBM bernama EF Codd yang membahas tentang ide pembuatan basis data relasional pada bulan Juni 1 1970. Artikel ini juga membahas kemungkinan pembuatan bahasa standar untuk mengakses data dalam basis data tersebut. Bahasa tersebut kemudian diberi nama SEQUEL (*Structured English Query Language*). Setelah terbitnya artikel tersebut, IBM mengadakan proyek pembuatan basis data relasional berbasis bahasa SEQUEL. Akan tetapi, karena permasalahan hukum mengenai penamaan SEQUEL, IBM pun mengubahnya menjadi SQL. Implementasi basis data relasional dikenal dengan *System/R*. Di akhir tahun 1970-an, muncul perusahaan bernama *Oracle* yang membuat server basis data populer yang bernama sama dengan nama perusahaannya. Dengan naiknya kepopuleran *Oracle*, maka SQL juga ikut populer sehingga saat ini menjadi standar *de facto* bahasa dalam manajemen basis data.



Gambar 2.1 Hierarki Database

## Membuat Data Base dan menghapus data base

1. Membuat Data Base

Sintaks umum SQL untuk membuat suatu database adalah sebagai berikut :

|  |
| --- |
| CREATE DATABASE [IF NOT EXISTS]nama\_database; |

Bentuk perintah di atas akan membuat sebuah database baru dengan nama nama\_database. Aturan penamaan sebuah database sama seperti aturan penamaan sebuah variabel, dimana secara umum nama database boleh terdiri dari huruf, angka dan under-score (\_). Jika database yang akan dibuat sudah ada, maka akan muncul pesan error. Namun jika ingin otomatis menghapus database yang lama jika sudah ada, aktifkan option IF NOT EXISTS.

Berikut ini contoh perintah untuk membuat database baru dengan nama “penjualan” :

|  |
| --- |
| CREATE DATABASE penjualan; |

Jika query di atas berhasil dieksekusi dan database berhasil dibuat, maka akan ditampilkan pesan kurang lebih sebagai berikut :

|  |
| --- |
| Query OK, 1 row affected (0.02 sec) |

1. Menghapus Data Base

Untuk menghapus suatu database, sintaks umumnya adalah sbb :

|  |
| --- |
| DROP DATABASE [IF EXISTS] nama\_database; |

Bentuk perintah di atas akan menghapus database dengan nama nama\_database. Jika databasenya ada maka database dan juga seluruh tabel di dalamnya akan dihapus. Jadi berhati-hatilah dengan perintah ini! Jika nama database yang akan dihapus tidak ditemukan, maka akan ditampilkan pesan error. Aktifkan option IF EXISTS untuk memastikan bahwa suatu database benar-benar ada

## *Stored Procedure*

*Stored Procedure* merupakan suatu kumpulan perintah atau statement yang disimpan dan dieksekusi di server database MySQL. Dengan SP (*Stored Procedure*), kita dapat menyusun program sederhana berbasis sintaks SQL untuk menjalankan fungsi tertentu. Hal ini menjadikan aplikasi yang kita buat lebih efektif dan efisien.

*Stored Procedure* dari segi bentuk dan sifatnya terbagi menjadi 2 (dua), yaitu FUNCTION dan PROCEDURE. Perbedaan utama antara function dan Procedure adalah terletak pada nilai yang dikembalikannya (di-return). Function memiliki suatu nilai yang dikembalikan (di-return), sedangkan procedure tidak. Umumnya suatu procedure hanya berisi suatu kumpulan proses yang tidak menghasilnya value, biasanya hanya menampilkan saja.

## Membuat dan Menghapus Tabel

1. Membuat Tabel

Bentuk umum SQL untuk membuat suatu table secara sederhana sebagai berikut :

|  |
| --- |
| CREATE TABLE nama\_tabel (  field1 tipe(panjang),  field2 tipe(panjang),  fieldn tipe(panjang),  PRIMARY KEY (field\_key)  ); |

Bentuk umum di atas merupakan bentuk umum pembuatan tabel yang sudah disederhanakan. Penamaan tabel dan field memiliki aturan yang sama dengan penamaan database.

1. Menghapus table

Untu`k menghapus sebuah tabel, bentuk umum dari perintah SQL adalah sebagai berikut :

|  |
| --- |
| DROP TABLE nama\_tabel; |

* + 1. **Merubah Struktur Tabel**

Untuk mengubah struktur suatu tabel, bentuk umum perintah SQL-nya sebagai berikut :

|  |
| --- |
| ALTER TABLE nama\_tabel alter\_options; |

dimana :

* ALTER TABLE merupakan perintah dasar untuk mengubah tabel.
* nama\_tabel merupakan nama tabel yang akan diubah strukturnya.
* alter\_options merupakan pilihan perubahan tabel. Option yang bisa digunakan, beberapa di antaranya sebagai berikut :
* ADD definisi\_field\_baru

Option ini digunakan untuk menambahkan field baru dengan “definisi\_field\_baru” (nama field, tipe dan option lain).

* ADD INDEX nama\_index

Option ini digunakan untuk menambahkan index dengan nama “nama\_index” pada tabel.

* ADD PRIMARY KEY (field\_kunci)

Option untuk menambahkan primary key pada tabel

* CHANGE field\_yang\_diubah definisi\_field\_baru Option untuk mengubah field\_yang\_diubah menjadi definisi\_field\_baru
* MODIFY definisi\_field

Option untuk mengubah suatu field menjadi definisi\_field

* DROP nama\_field

Option untuk menghapus field nama\_field

* RENAME TO nama\_tabel\_baru

Option untuk mengganti nama tabel

# DATA HASIL

## Entity Relationship Diagram

## Jobsheet

|  |  |  |  |
| --- | --- | --- | --- |
| No | Latihan |  | Query dan Hasil |
| 1 | Database Baru | Membuat Database baru | create database dataGudang; |
|  |
| Melihat database | show databases; |
|  |
| Menggunakan database yang telah dibuat | use dataGudang; |
|  |
| 2 | Membuat tabel | Membuat table | create table Supplier( idSupplier int, namaSupplier varchar(50), alamatSupplier varchar(50), noHP varchar(13), primary key (idSupplier)); |
|  |
| Membuat tabel | create table Barang( kodeBarang int, namaBarang varchar(50), jumlahBarang int, hargaBarang int, noTataLetak int, tanggalMasuk int, tanggal keluar int, primary key (kodeBarang)); |
|  |
|  | create table Checker( idChecker int, namaChecker varchar(50), primary key (idChecker)); |
| Membuat tabel |  |
|  | Membuat Index | Membuat Index | create index `namaBarang` on `barang`(`namaBarang`); |
|  |  |
| 4 | Membuat stored procude | Membuat | delimiter //  create procedure lihat()  begin  select\*from barang;  end // |
|  |
| delimiter //  create procedure lihat  (IN kodeInput int)  begin  select\*from barang where kodeBarang=kodeInput;  end // |
|  |
| delimiter //  create procedure lihat  (IN inputKode int, IN inputNama varchar(50), IN inputMerek varchar(20), IN inputJumlah int, IN inputHarga int, IN inputLetak int, IN inputMasuk int, IN inputKeluar int)  begin  insert into Barang values (inputKode, inputNama, inputMerek, inputJumlah, inputHarga, inputLetak, inputMasuk, inputKeluar);  end //  delimiter ; |
|  |
| 5 | Melihat tabel | Melihat table yang sudah di buat | show tables; |
|  |
| 6 | Melihat deskripsi tabel | Melihat deskripsi dari tabel | select\*from Supplier; |
|  |
| Melihat deskripsi dari tabel | select\*from Barang; |
|  |
|  | select\*from Checker; |
| Melihat deskripsi dari tabel |  |
| 7 | Merubah tabel | Merubah nama tabel | alter table Supplier  rename to Penyuplai; |
|  |
| Membuat kolom baru | alter table Penyuplai  add namaSupplier varchar(100); |
|  |
| Mengahpus kolom pada tabel | alter table Penyuplai  drop namaSupplier; |
|  |

# ANALISA DATA HASIL

|  |  |  |  |
| --- | --- | --- | --- |
| **No.** | **Latihan** | **Query dan Hasil** | **ANALISA** |
| **1** | Membuat database | create database dataGudang; | Untuk membuat database menggunakan sintax create database nama\_database; dan pada gambar di samping nama databasenya adalah dataGudang; |
|  |
| **2** | Melihat database | show databases; | Sintax untuk melihat database pada MySQL adalah show databases; pada gambar disamping terlihat beberapa database salah satunya database yang baru dibuat adalah dataGudang. |
|  |
| **3** | Menggunakan database | Use dataGudang; | Setelah membuat database selanjutnya adalah menggunakan database dengan sintax use nama\_database, dan pada gambar disamping nama databasenya adalah dataGudang sehingga penulisan sintaxnya use dataGudang |
|  |
| **4** | Membuat tabel | create table Supplier( idSupplier int, namaSupplier varchar(50), alamatSupplier varchar(50), noHP varchar(13), primary key (idSupplier)); | Pada query dan gambar disamping digunakan untuk mebuat tabel diaman nama tabelnya adalah Supplier dan memiliki kolom idSupplier dengan tipe data int, selanjutnya kolom namaSupplier dengan tipe data varchar(50), dan kolom-kolom selajutnya lalu tutup kurung dan diakhiri titik koma. |
|  |
| **5** | Membuat index | create index `namaBarang` on `barang`(`namaBarang`); | Pada query dan gambar disamping digunakan untuk membuat index dengan sintax create index `nama\_index` on `nama\_tabel`(`nama\_kolom`); pada kasus disamping nama index yang dibuat adalah namaBarang pada tabel Barangdibaris namaBarang. Pada gambar disamping terlihat kolom null yang fungsinya untuk mengetahui bahwa tabel tersebut terisi atau tidak dengan yes berarti berisi dan sebaliknya, kemudian pada kolom key kodeBarang sebagai primary key dan namaBarang sebagai index. Pada kolom default fungsinya untuk menginformasikan bahwa kolom dalam keadaan awal yaitu null atau belum terisi suatu data. |
|  |
| **6** | Membuat stored procedure | delimiter //  create procedure lihat  (IN inputKode int, IN inputNama varchar(50), IN inputMerek varchar(20), IN inputJumlah int, IN inputHarga int, IN inputLetak int, IN inputMasuk int, IN inputKeluar int)  begin  insert into Barang values (inputKode, inputNama, inputMerek, inputJumlah, inputHarga, inputLetak, inputMasuk, inputKeluar);  end //  delimiter ; | Pada query di samping digunakan untuk membuat stored procedure dengan sintax create procedure nama\_prosedure(value) begin perintah\_yang\_akan\_dilakukan end // pada kasus query disamping digunakan untuk membuat stored procedure menginputkan isi kolom. |
|  |
| **7** | Melihat tabel | Show tables; | Pada query di samping digunakan untuk melihat tabel dengan sintax Show tables; maka akan terlihat tabel yang telah dibuat pada database. |
|  |
| **8** | Melihat deskripsi tabel | select\*from supplier; | Pada query disamping digunakn untuk melihat isi dari tabel secara keseluruhan dengan sintax select\*from nama\_tabel; dan pada gambar di samping terlihat nama tabel yang akan dilihat adalah supplier sehingga penulisan querynya adalah select\*from supplier;. |
|  |
| **9** | Merubah nama tabel | alter table Supplier  rename to Penyuplai; | Untuk merubah nama tabel adalah dengan sinta alter table nama\_tabel rename to nama\_tabel\_baru; pada query disamping nama tabel yang akan diubah adalah Supplier menjadi Penyuplai. |
|  |
| **10** | Membuat kolom baru | alter table Penyuplai  add namaSupplier varchar(100); | Untuk membuat kolom baru menggunakan sintax alter table nama\_tabel add nama\_kolom type\_data; pada query disamping nama kolom yang akan di tambahkan adalah namaSupplier dengan tipe data varchar (100) pada tabel Supplier. |
|  |
| **11** | Menghapus kolom | alter table Penyuplai  drop namaSupplier; | Untuk menghapus kolom adalah dengan sintax alter table nama\_tabel drop nama\_kolom; pada query disamping nama kolom yang akan dihapus adalah namaSupplier pada tabel Penyuplai. |
|  |

# KESIMPULAN

# 1. *Data Definition Language* (DDL) adalah kumpulan perintah SQL yang berkaitan dengan pembuatan, perubahan, dan penghapusan *database* maupun objek-objek yang terdapat di dalam database. Salah satu bentuk bahasa basis data yaitu *Data Definition Language* (DDL) yang digunakan untuk membuat, mengubah, serta menghapus basis data dan objek-objek yang diperlukan dalam basis data.

# 2. Perintah-peritah yang terdapat di dalam DDL yang dapat membuat, menghapus, dan merubah *database* dan tabel yang ada di dalam *database* yaitu CREATE, USE, ALTER, DROP dan CREATE INDEX.

**3.** CREATE adalah perintah yang dapat membuat database dan tabel, USE adalah perintah yang digunakan jika database yang sudah di buat (CREATE) ingin digunakan. ALTER adalah sebuah perintah yang dapat di gunakan jika ingin mengubah sebuah tabel yang sudah di buat di dalam database. DROP merupakan perintah yang dapat menghapus sebuah databse dan tabel yang tidak di inginkan. CREATE INDEX adalah perintah untuk membuat sebuah index.

# DAFTAR PUSTAKA

Anwar, Purnamasari. *PENERJEMAHAN TEKS BAHASA INDONESIAMENJADI DATA DEFINITION LANGUAGE (DDL) DENGAN PENANGANAN KALIMAT MAJEMUK.* Teknik Informatika Universitas Komputer Indonesia.

Achmad Solichin. 2010 . *MySQL 5 Dari Pemula hingga Mahir.* Universitas Budi Luhur , Jakarta

Muhammad Denny Prayoga. *Pengertian dan Komponen SQL.* Fakultas Komputer