MODUL II

Data Definision Language (DDL)

Tujuan :

Setelah menyelesaikan modul ini, anda diharapkan dapat :

1. Membuat database dan tabel dengan data definition language
2. Mampu memodifikasi tabel

Dasar Teori

DDL (Data Definition Language), DDL merupakan kelompok perintah yang berfungsi untuk mendefinisikan atribut-atribut basis data, tabel, field(kolom), batasan-batasan terhadap suatu field, serta hubungan antar tabel. Yang termasuk dalam kelompok DDL ini adalah CREATE, ALTER, dan DROP. DDL adalah sebuah metode Query SQL yang berguna untuk mendefinisikan data pada sebuah Database.

1.1 Database

Database adalah sebuah media utama yang harus dibuat dalam membangun sebuah basis data agar nantinya dapat kita letakkan beberapa tabel dengan field-fieldnya.

a. Perintah untuk membuat database adalah

```
CREATE DATABASE namadatabase;
```

namadatabase tidak boleh mengandung spasi dan tidak boleh memiliki nama yang sama antar database. Berikut ini perintah untuk membuat database dengan nama **pethouse**.

```
CREATE DATABASE pethouse;
```

Hasil perintah untuk membuat database pethouse adalah sebagai berikut:

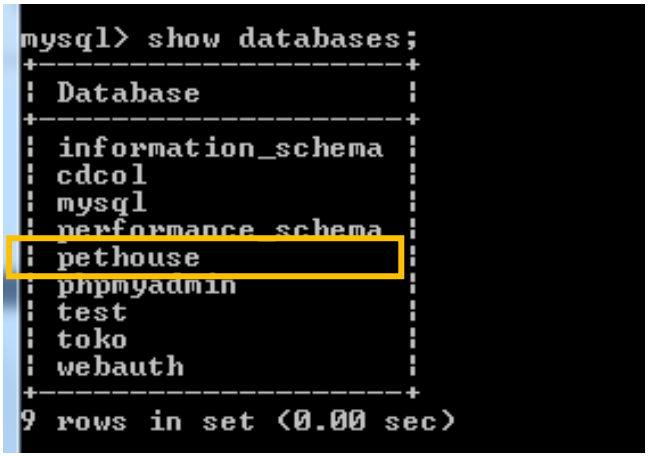
```
mysql> create database pethouse;
Query OK, 1 row affected (0.00 sec)
```

b. Melihat database

Setelah perintah membuat database berhasil, maka kita dapat melihat database yang telah dibuat, dengan menggunakan perintah sebagai berikut:

```
SHOW databases;
```

Hasilnya adalah sebagai berikut:



Pada tampilan di atas, dapat dilihat semua database yang ada, antara lain adalah database **pethouse** yang telah dibuat sebelumnya.

c. Memilih database

Untuk memilih database mana yang akan digunakan, kita dapat memilih database dengan menggunakan perintah USE. *Syntax* penggunaan perintah USE adalah sebagai berikut :

```
USE namadatabase;
```

Misalkan kita ingin menggunakan database **pethouse** yang telah kita buat sebelumnya, maka kita dapat menggunakan perintah sebagai berikut :

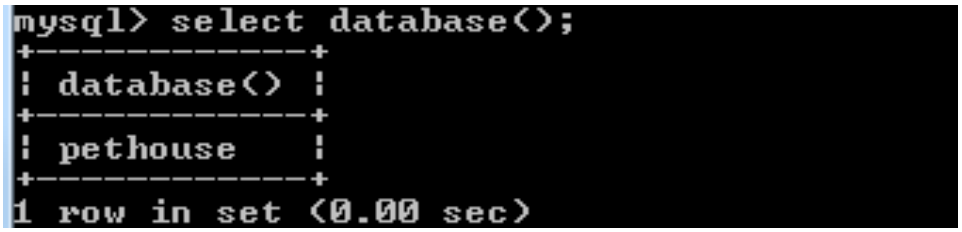
Maka database yang digunakan saat ini adalah ‘pethouse’.

d. Cek Database

Untuk melihat nama database yang digunakan saat ini, kita dapat melakukan pengecekan dengan menggunakan perintah :

```
select database();
```

Hasilnya adalah sebagai berikut:



Database yang digunakan saat ini adalah ‘pethouse’.

e. Menghapus Database

Untuk menghapus database, kita dapat menggunakan perintah sebagai berikut :

```
DROP DATABASE namadatabase;
```

Namadatabase disesuaiakan dengan database yang akan dihapus. Misalkan kita akan menghapus dapatabase ‘pethouse’, maka dilakukan dengan menggunakan perintah sebagai berikut :

```
mysql> DROP DATABASE pethouse;
Query OK, 0 rows affected (0.12 sec)
```

Untuk memastikan bahwa database ‘pethouse’ sudah terhapus, maka kita dapat melakukan pengecekan dengan menampilkan semua database yang ada, dengan perintah :

```
mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| cdcol              |
| mysql              |
| performance_schema |
| phpmyadmin         |
| test               |
| toko               |
| webauth            |
+--------------------+
8 rows in set (0.00 sec)
```

Dari daftar semua database yang ada, tidak ditemukan database ‘pethouse’, karena sudah dilakukan penghapusan database ‘pethouse’.

1.2 Tabel

a. Membuat tabel

Sebelum kita membuat tabel, maka kita buat terlebih dahulu database yang akan digunakan, dengan langkah-langkah sebagai berikut :

1. Buat database ‘pethouse’.
2. Gunakan database ‘pethouse’
3. Cek tabel yang terdapat pada database ‘pethouse’, karena belum pernah dilakukan pembuatan tabel pada database ‘pethouse’, maka muncul komentar ‘empty set’.
4. Buat tabel ‘pet’, dengan nama-nama *field, type*, dan *length* sebagai berikut :

Filed	***Type***	***Length***
name	VARCHAR	20

owner	VARCHAR	20
species	VARCHAR	20
sex	CHAR	1
birth	DATE	-
death	DATE	-

Beberapa tipe data pada mysql adalah sebagai berikut :

Type Data	Keterangan
TINYINT	Ukuran 1 byte. Bilangan bulat terkecil, dengan jangkauan untuk bilangan bertanda: -128 sampai dengan 127 dan untuk yang tidak bertanda : 0 s/d 255. Bilangan tak bertandai dengan kata UNSIGNED
SMALLINT	Ukuran 2 Byte. Bilangan bulat dengan jangkauan untuk bilangan bertanda : -32768 s/d 32767 dan untuk yang tidak bertanda : 0 s/d 65535
MEDIUMINT	Ukuran 3 byte. Bilangan bulat dengan jangkauan untuk bilangan bertanda : -8388608 s/d 8388607 dan untuk yang tidak bertanda : 0 s/d 16777215
INT	Ukuran 4 byte. Bilangan bulat dengan jangkauan untuk bilangan bertanda : -2147483648 s/d 2147483647 dan untuk yang tidak bertanda : 0 s/d 4294967295
INTEGER	Ukuran 4 byte. Sinonim dari int
BIGINT	Ukuran 8 byte. Bilangan bulat terbesar dengan jangkauan untuk bilangan bertanda : -9223372036854775808 s/d 9223372036854775807 dan untuk yang tidak bertanda : 0 s/d 1844674473709551615
FLOAT	Ukuran 4 byte. Bilangan pecahan
DOUBLE	Ukuran 8 byte. Bilangan pecahan
DOUBLEPRECISION	Ukuran 8 byte. Bilangan pecahan
REAL	Ukuran 8 byte. Sinonim dari DOUBLE
DECIMAL (M,D)	Ukuran M byte. Bilangan pecahan, misalnya DECIMAL(5,2 dapat digunakan untuk menyimpan bilangan -99,99 s/d 99,99
NUMERIC (M,D)	Ukuran M byte. Sinonim dari DECIMAL, misalnya NUMERIC(5,2) dapat digunakan untuk menyimpan bilangan -99,99 s/d 99,99

Type Data untuk Bilangan (Number)

Type Data	Keterangan
DATETIME	Ukuran 8 byte. Kombinasi tanggal dan jam, dengan jangkauan dari '1000-01-01 00:00:00' s/d '9999-12-31 23:59:59'
DATE	Ukuran 3 Byte. Tanggal dengan jangkauan dari '1000-01-01' s/d '9999-12-31'
TIMESTAMP	Ukuran 4 byte. Kombinasi tanggal dan jam, dengan jangkauan dari '1970-01-01 00:00:00' s/d '2037'
TIME	Ukuran 3 byte. Waktu dengan jangkauan dari '839:59:59' s/d '838:59:59'
YEAR	Ukuran 1 byte. Data tahun antara 1901 s/d 2155

Type Data untuk Tanggal dan Jam

Type Data	Keterangan
CHAR	Mampu menangani data hingga 255 karakter. Tipe data CHAR mengharuskan untuk memasukkan data yang telah ditentukan oleh kita.
VARCHAR	Mampu menangani data hingga 255 karakter. Tipe data VARCHAR tidak mengharuskan untuk memasukkan data yang telah ditentukan oleh kita.
TINYBLOB, TINYTEXT	Ukuran 255 byte. Mampu menangani data sampai 2^8-1 data.
BLOB, TEXT	Ukuran 65535 byte. Type string yang mampu menangani data hingga 2^16-1 (16M-1) data.
MEDIUMBLOB, MEDIUMTEXT	Ukuran 16777215 byte. Mampu menyimpan data hingga 2^24-1 (16M-1) data.
LONGBLOB, LONGTEXT	Ukuran 4294967295 byte. Mampu menyimpan data hingga berukuran GIGA BYTE. Tipe data ini memiliki batas penyimpanan hingga 2^32-1 (4G-1) data.
ENUM('nilai1','nilai2',...,'nilaiN')	Ukuran 1 atau 2 byte. Tergantung jumlah nilai enumerasinya (maksimum 65535 nilai)
SET('nilai1','nilai2',...,'nilaiN')	1,2,3,4 atau 8 byte, tergantung jumlah anggota himpunan (maksimum 64 anggota)

Type Data untuk Karakter dan Lain-lain

Perintah untuk membuat tabel adalah sebagai berikut :

```
CREATE TABLE namatabel (
Field1 TipeData1(length),
Field2 TipeData2(length)
);
```

Nama tabel tidak boleh mengandung spasi (space). Field1 dan TipeData1 merupakan nama kolom pertama dan tipe data untuk kolom pertama. Jika ingin membuat tabel dengan kolom lebih dari satu, maka setelah pendefinisian tipe data sebelumnya diberikan tanda koma (,).

Berikut ini perintah untuk membuat tabel dengan nama 'pet' :

```
mysql> create table pet (name varchar(20), owner varchar(20),
    -> species varchar(20), sex char(1), birth date, death date);
Query OK, 0 rows affected (0.11 sec)
```

b. Menampilkan Tabel

Untuk menampilkan daftar nama tabel yang ada pada database yang sedang aktif/digunakan (dalam hal ini database pethouse) :

```
SHOW TABLES;
```

Hasilnya sebagai berikut :

```
mysql> show tables;
+--------------------+
| Tables_in_pethouse |
+--------------------+
| pet                |
+--------------------+
1 row in set (0.00 sec)
```

Pada tampilan di atas, dapat dilihat, bahwa tabel yang terdapat pada database 'pethouse' hanya terdapat 1 row yaitu tabel 'pet'.

c. Menampilkan Field & TipeData (Struktur Tabel)

Untuk melihat struktur tabel 'pet' yang telah dibuat sebelumnya, maka kita dapat menggunakan perintah sebagai berikut :

```
DESCRIBE namatabel;
```

Hasilnya adalah sebagai berikut :

```
mysql> DESCRIBE pet;
+---------+-------------+------+-----+---------+-------+
| Field   | Type        | Null | Key | Default | Extra |
+---------+-------------+------+-----+---------+-------+
| name    | varchar(20) | YES  |     | NULL    |       |
| owner   | varchar(20) | YES  |     | NULL    |       |
| species | varchar(20) | YES  |     | NULL    |       |
| sex     | char(1)     | YES  |     | NULL    |       |
| birth   | date        | YES  |     | NULL    |       |
| death   | date        | YES  |     | NULL    |       |
+---------+-------------+------+-----+---------+-------+
6 rows in set (0.02 sec)
```

Pada struktur tabel 'pet' dapat dilihat, bahwa pada tabel tersebut memiliki *field* name. Owner, species, sex, birth dan death.

d. Memasukkan Data

Jika tabel sudah berhasil dibuat, maka langkah selanjutnya kita dapat mengisi tabel 'pet' dengan beberapa data, dengan menggunakan perintah sebagai berikut :

```
INSERT INTO namatabel(field1, field2,..,field-n)
VALUES ('.....', '.....'..., '.... ');
```

Atau dapat menggunakan perintah :

```
INSERT INTO namatabel
VALUES ('.....', '.....'..., '.... ');
```

Hasil penggunaan perintah di atas pada tabel 'pet' adalah sebagai berikut :

```
mysql> insert into pet
    -> values ('Fluffy', 'Harold', 'cat','f','1993-02-04',null);
Query OK, 1 row affected (0.08 sec)
```

Untuk melihat hasil dari proses INSERT data ke tabel 'pet', dapat menggunakan perintah :

```
SELECT * FROM namatabel;
```

Namatabel kita sesuaikan dengan tabel yang akan dilihat isinya. Hasil dari perintah tersebut adalah sebagai berikut :

```
mysql> select * from pet;
+--------+--------+---------+------+------------+-------+
| name   | owner  | species | sex  | birth      | death |
+--------+--------+---------+------+------------+-------+
| Fluffy | Harold | cat     | m    | 1993-02-04 | NULL  |
+--------+--------+---------+------+------------+-------+
1 row in set (0.00 sec)
```

e. Menghapus Tabel

Untuk menghapus tabel, dilakukan dengan menggunakan perintah berikut ini :

```
DROP TABLE  namatabel;
```

Nama tabel disesuaikan dengan tabel yang akan dihapus. Setelah tabel dihapus, maka untuk memastikan tabel tidak ada, kita dapat menampilkan semua tabel dengan perintah **SHOW TABLES;**

Praktikum

1. Buatlah sebuah database dengan nama library.
2. Dengan menggunakan database tersebut, buatlah tabel book, dengan nama *field* (tittle, author, publisher).
3. Tentukan tipe data dan *length* untuk masing-masing *filed.*
4. Tampilkan struktur tabel yang anda buat !
5. Isikan tabel tersebut dengan beberapa data, dengan menggunakan perintah INSERT.
6. Tampilkan isi tabel hasil dari perintah 5.
7. Buat folder atas nama anda masing-masing di drive document, dalam forder tersebut, buat folder dengan nama modul_2, simpan hasil dokumentasi hasil praktikum yang anda dengan nama praktikum!

Latihan

Dengan menggunakan *command line,*

1. Buatlah sebuah tabel bernama ‘event’ dalam database ‘pethouse’ dengan kolol sebagai berikut :

Field	***Type***	***Length***
name	VARCHAR	20
date	DATE	-

type	VARCHAR	15
remark	VARCHAR	255

2. Isikan tabel tersebut dengan menggunakan data di bawah ini, dengan menggunakan perintah INSERT !

name	**date**	**type**	**remark**

Fluffy	1995-05-15	Litter	4 kittens, 3 female, 1 male
Buffy	1993-06-23	Litter	5 puppies, 2 female, 3 male
Buffy	1994-06-19	Litter	3 puppies, 3 female
Chirpy	1999-03-21	Vet	Needed beak straightened

3. Isikan tabel tersebut dengan data di bawah ini, dengan menggunakan Insert !

name	**date**	**type**	**remark**
Slim	1997-08-03	Vet	Broken rib
Bowser	1991-10-12	kennel	-
Fang	1991-10-12	Kennel	-
Fang	1998-08-28	Birthday	Gave him a new chew toy
Claws	1998-03-17	Birthday	Gave him a new flea collar
Whistler	1998-12-09	Birthday	First birthday

4. Tampilkan data yang sudah berhasil dimasukkan, kemudian buat *screenshoot* letakkan pada file tugas.doc pada folder yang sudah anda buat saat praktikum !