BAB II
Dasar-Dasar MySQL

Standar Kompetensi :

1. Mahasiswa dapat mengetahui, memahami, menguasai dan mampu mengimplementasi teori, konsep dan prinsip pemrograman database MySQL dengan logika pemrograman yang benar, ringkas, dan tepat dalam penerapannya di bidang teknologi informasi

Kompetensi Dasar	Indikator
1.4 Mengetahui Dasar-Dasar MySQL	▪ Mengetahui tipe data dalam MySQL ▪ Mengenal perintah dasar MySQL dan mengetahui fungsinya ▪ Menggunakan MySQL untuk mengetahui tanggal dan waktu ▪ Menggunakan MySQL sebagai kalkulator
1.5 Membuat Database	▪ Mengetahui cara membuat database, menggunakan database dan menghapus database
1.6 Membuat Tabel	▪ Mengetahui cara membuat tabel, menentukan field beserta tipe datanya ▪ Mengetahui cara menghapus tabel
1.7 Mengenal Constraint	▪ Mengetahui beberapa tipe constraint ▪ Mengetahui fungsi msing-masing tipe constraint

Materi

2.1 Dasar-Dasar MySQL

Dalam bahasa SQL pada umumnya informasi tersimpan dalam tabel-tabel yang secara logik merupakan struktur dua dimensi terdiri dari baris (*row* atau *record*) dan kolom (*column* atau *field*). Sedangkan dalam sebuah *database* dapat terdiri dari beberapa *table*.

Prompt **mysql>_** menunjukkan bahwa database mysql telah aktif. Jika prompt ini telah aktif dapat langsung mengetikkan perintah-perintah dilingkungan MySQL.

Perintah-perintah MySQL antara lain :

Tabel 2.2 Perintah-perintah MySQL

Perintah	Perintah Singkat	Kegunaan
Help	\h	Menampilkan daftar perintah
Clear	\c	Menghapus (clear)
Connect	\r	Menghubungkan kembali database MySQL
Exit	\q	Keluar dari MySQL
Go	\g	Mengirimkan perintah kepada MySQL
Ego	\G	Mengirimkan perintah kepada MySQL dan menampilkan hasilnya secara vertical
Print	\p	Mencetak perintah saat ini
Use	\u	Membuat/mengganti koneksi kepada database

Ketentuan Memberikan Perintah

- Perintah dalam MySQL mengenal case insensitive, perintah dapat ditulis dengan huruf besar (uppercase), ataupun dengan huruf kecil (lowercase).
- Setiap perintah diakhiri dengan ; (tanda titik koma) atau dengan memberikan perintah \g diakhir perintah
- Perintah dapat berupa perintah SQL atau perintah khusus MySQL
- Jika Prompt mysql> berganti dengan -> berarti prompt tersebut menunggu kelengkapan perintah dari baris sebelumnya atau menunggu diberikan tanda ; atau \g.

  Contoh : Perhatikan perintah dibawah ini ditulis **tanpa tanda titik-koma ";".**

```
mysql> create database latihan1
    ->
```

  Sistem MySQL akan menampilkan tanda panah '**->**' yang menyatakan bahwa perintah MySQL tersebut dianggap belum selesai (karena belum diakhiri dengan tanda titik-koma ';').

  Sekarang kita lengkapi perintah sebelumnya dengan tanda titik-koma ';'

```
mysql> create database latihan1
-> ;
Query OK, 1 row affected (0.02 sec)
```

2.2 Tipe Data pada MySQL

Pemilihan tipe data merupakan suatu hal yang cukup penting dalam mengelola server. Salah satu sebabnya adalah berkaitan dengan ruang di harddisk dan memori yang akan "digunakan" oleh data-data tersebut.

Berikut ini akan diberikan tipe-tipe data yang didukung oleh MySQL yang terambil dari dokumentasi MySQL. Tipe - tipe data ini diberikan dalam bentuk yang siap dituliskan pada sintaks-sintaks MySQL, misalnya *Create Table*. Pada tipe-tipe data tersebut terdapat beberapa atribut yang memiliki arti sebagai berikut:

- M, menunjukkan lebar karakter maksimum. Nilai M maksimum adalah 255.
- D, menunjukkan jumlah angka di belakang koma. Nilai maksimum D adalah 30 tetapi dibatasi oleh nilai M, yaitu tidak boleh lebih besar daripada M-2.
- Atribut yang diberi tanda [dan] berarti pemakaiannya adalah optional.
- Jika atribut ZEROFILL disertakan, MySQL akan otomatis menambahkan atribut UNSIGNED.
- UNSIGNED adalah bilangan tanpa tanda di depannya (misalnya tanda negatif).

Inilah tipe-tipe data tersebut:

1. **TINYINT[(M)] [UNSIGNED] [ZEROFILL]**

   Integer yang sangat kecil jangkauan nilainya, yaitu -128 hingga 127. Jangkauan unsigned adalah 0 hingga 255.

2. **SMALLINT[(M)] [UNSIGNED] [ZEROFILL]**

   Integer yang kecil jangkauan nilainya, yaitu -32768 hingga 32767. Jangkauan unsigned adalah 0 hinga 65535.

3. **MEDIUMINT[(M)] [UNSIGNED] [ZEROFILL]**

   Integer tingkat menengah. Jangkauan nilainya adalah -8388608 hingga 8388607. Jangkauan unsigned adalah 0 hingga 16777215.

4. **INT[(M)] [UNSIGNED] [ZEROFILL]**

   Integer yang berukuran normal. Jangkauan nilainya adalah -2147483648 hingga 2147483647. Jangkauan unsigned adalah 0 hingga 4294967295.

5. **INTEGER[(M)] [UNSIGNED] [ZEROFILL]**

   Sama dengan INT.

6. **BIGINT[(M)] [UNSIGNED] [ZEROFILL]**

   Integer berukuran besar. Jangkauan nilainya adalah -9223372036854775808 hingga 9223372036854775807. Jangkauan unsigned adalah 0 hingga 18446744073709551615.

7. **FLOAT(precision) [ZEROFILL]**

   Bilangan floating-point. Tidak dapat bersifat unsigned. Nilai atribut precision adalah <=24 untuk bilangan floating-point presisi tunggal dan di antara 25 dan 53 untuk bilangan floating-point presisi ganda.

8. **FLOAT[(M,D)] [ZEROFILL]**

   Bilangan floating-point presisi tunggal. Tidak dapat bersifat unsigned. Nilai yang diijinkan adalah -3.402823466E+38 hingga -1.175494351E-38 untuk nilai negatif, 0, and 1.175494351E-38 hingga 3.402823466E+38 untuk nilai positif.

9. **DOUBLE[(M,D)] [ZEROFILL]**

   Bilangan floating-point presisi ganda. Tidak dapat bersifat unsigned. Nilai yang diijinkan adalah -1.7976931348623157E+308 hingga -2.2250738585072014E-308 untuk nilai negatif, 0, dan 2.2250738585072014E-308 hingga 1.7976931348623157E+308 untuk nilai positif.

10. **DOUBLE PRECISION[(M,D)] [ZEROFILL] dan REAL[(M,D)] [ZEROFILL]**

    Keduanya sama dengan DOUBLE.

11. **DECIMAL[(M[,D])] [ZEROFILL]**

    Bilangan floating-point yang “unpacked”. Tidak dapat bersifat unsigned. Memiliki sifat mirit dengan CHAR. Kata “unpacked'' berarti bilangan disimpan sebagai string, menggunakan satu karakter untuk setiap digitnya. Jangkauan nilai dari DECIMAL sama dengan DOUBLE, tetapi juga tergantung dai nilai atribut M dan D yang disertakan. Jika D tidak diisi akan dianggap 0. Jika M tidak diisi maka akan

dianggap 10. Sejak MySQL 3.22 nilai M harus termasuk ruang yang ditempati oleh angka di belakang koma dan tanda + atau -.

12. **NUMERIC(M,D) [ZEROFILL]**

    Sama dengan DECIMAL.

13. **DATE**

    Sebuah tanggal. MySQL menampilkan tanggal dalam format 'YYYY-MM-DD'. Jangkauan nilainya adalah '1000-01-01' hingga '9999-12-31'.

14. **DATETIME**

    Sebuah kombinasi dari waktu (jam) dan tanggal. MySQL menampilkan waktu dan tanggal dalam format 'YYYY-MM-DD HH:MM:SS'. Jangkauan nilainya adalah '1000-01-01 00:00:00' hingga '9999-12-31 23:59:59'.

15. **TIMESTAMP[(M)]**

    Sebuah timestamp. Jangkauannya adalah dari '1970-01-01 00:00:00' hingga suatu waktu di tahun 2037. MySQL menampilkan tipe data TIMESTAMP dalam format YYYYMMDDHHMMSS, YYMMDDHHMMSS, YYYYMMDD, atau YYMMDD, tergantung dari nilai M, apakah 14 (atau tidak ditulis), 12, 8, atau 6.

16. **TIME**

    Tipe data waktu. Jangkauannya adalah '-838:59:59' hingga '838:59:59'. MySQL menampilkan TIME dalam format 'HH:MM:SS'.

17. **YEAR[(2|4)]**

    Angka tahun, dalam format 2- atau 4-digit (default adalah 4-digit). Nilai yang mungkin adalah 1901 hingga 2155, 0000 pada format 4-digit, dan 1970-2069 pada format 2-digit (70-69).

18. **CHAR(M) [BINARY]**

    String yang memiliki lebar tetap. Nilai M adalah dari 1 hingga 255 karakter. Jika ada sisa, maka sisa tersebut diisi dengan spasi (misalnya nilai M adalah 10, tapi data yang disimpan hanya memiliki 7 karakter, maka 3 karakter sisanya diisi dengan spasi). Spasi ini akan dihilangkan apabila data dipanggil. Nilai dari CHAR akan

disortir dan diperbandingkan secara case-insensitive menurut default character set yang tersedia, kecuali bila atribut BINARY disertakan.

19. **VARCHAR(M) [BINARY]**

    String dengan lebar bervariasi. Nilai M adalah dari 1 hingga 255 karakter. Jika nilai M adalah 10 sedangkan data yang disimpan hanya terdiri dari 5 karakter, maka lebar data tersebut hanya 5 karakter saja, tidak ada tambahan spasi.

20. **TINYBLOB dan TINYTEXT**

    Sebuah BLOB (semacam catatan) atau TEXT dengan lebar maksimum 255 (2^8 - 1) karakter.

21. **BLOB dan TEXT**

    Sebuah BLOB atau TEXT dengan lebar maksimum 65535 (2^16 - 1) karakter.

22. **MEDIUMBLOB dan MEDIUMTEXT**

    Sebuah BLOB atau TEXT dengan lebar maksimum 16777215 (2^24 - 1) karakter.

23. **LONGBLOB dan LONGTEXT**

    Sebuah BLOB atau TEXT dengan lebar maksimum 4294967295 (2^32 - 1) karakter.

24. **ENUM('value1','value2',...)**

    Sebuah enumerasi, yaitu objek string yang hanya dapat memiliki sebuah nilai, dipilih dari daftar nilai 'value1', 'value2', ..., NULL atau nilai special "" error. Sebuah ENUM maksimum dapat memiliki 65535 jenis nilai.

25. **SET('value1','value2',...)**

    Sebuah set, yaitu objek string yang dapat memiliki 0 nilai atau lebih, yang harus dipilih dari daftar nilai 'value1', 'value2', ... Sebuah SET maksimum dapat memiliki 64 anggota.

2.2 Melihat User dan Versi MySQL

Untuk melihat user dan versi MySQL Anda, cukup dengan menggunakan rumus :

Select User (),Version ();

```
mysql> select user(),version();
+----------------+-----------------+
| user()         | version()       |
+----------------+-----------------+
| root@localhost | 5.1.34-community |
+----------------+-----------------+
1 row in set (0.05 sec)
```

2.3 Melihat Tanggal dan Waktu

Untuk melihat tanggal didalam MySQL anda dapat melakukan dengan rumus **curdate();**

```
Mysql>select curdate();
```

Sedangkan untuk melihat waktu dengan rumus **curtime();**

```
Mysql>select curtime();
```

Untuk melihat waktu dan sekaligus tanggal, maka tumus yang dituliskan adalah **now();**

```
Mysql>select now();
```

2.4 MySQL Sebagai Kalkulator

Dengan MySQL, kita tidak usah bingung ketika suatu saat kita harus menggunakan alat bantu kalkulator., karena hal ini dapat ditangani langsung oleh MySQL tanpa harus membuat program terlebih dahulu. Rumus yang dituliskan adalah **select rumus_perhitungan;**

```
mysql> select 5+5;
+-----+
| 5+5 |
+-----+
|  10 |
+-----+
1 row in set (0.03 sec)
```

Operator penghitungan meliputi : penjumlahan (+), pengurangan (-), perkalian (*), pembagian (/). Silakan Anda coba melakukan penghitungan bilangan dengan menggunakan operator tersebut.

2.5 Membuat Database Baru

Sudah ada 3 buah database di dalam sistem MySQL. Sekarang kita akan membuat sebuah database untuk latihan kita. Gunakan perintah "**CREATE DATABASE**" untuk membuat sebuah database.

```
mysql> create database latihan2 ;
Query OK, 1 row affected (0.02 sec)
```

Kita periksa hasil dari perintah di atas dengan "**SHOW DATABASE**".

```
mysql> show databases ;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| latihan1           |
| latihan2           |
| mysql              |
| test               |
+--------------------+
5 rows in set (0.00 sec)
```

2.6 Menghapus Database

Jika kita tidak memerlukan database *latihan2*, maka kita dapat menghapusnya dengan perintah **DROP DATABASE**.

*Hati-hati dalam menggunakan perintah **DROP DATABASE** ini, karena database beserta seluruh isinya akan lenyap dari muka bumi tanpa bisa kita kembalikan lagi! Parahnya lagi, sistem MySQL tidak memberikan pertanyaan konfirmasi kepada Anda sebelum melakukan proses penghapusan database ini!*

```
mysql> drop database latihan2 ;
Query OK, 0 row affected (0.02 sec)
```

Anda bisa memeriksanya lagi hasil dari perintah di atas dengan "**SHOW DATABASE**".

```
mysql> show databases ;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| latihan1           |
| mysql              |
| test               |
+--------------------+
4 rows in set (0.00 sec)
```

Anda perhatikan, database *latihan2* sudah menghilang. Sekali lagi, hati-hati dalam menggunakan perintah DROP DATABASE !

2.7 Memilih dan Membuka Sebuah Database

Sekarang kita pilih database "**latihan1**" dan kita buka dengan perintah "**USE**"

```
mysql> use latihan1 ;
Database change
```

2.8 Melihat Isi Sebuah Database

Untuk melihat apa isi dari sebuah database, kita gunakan perintah "**SHOW TABLES**". Mari kita coba.

```
mysql> show tables ;
Empty set (0.00 sec)
```

Hasil dari perintah SHOW TABLES diatas adalah "**Empty Set**", yang berarti belum ada tabel apapun di dalam database *latihan1*.

2.9 Membuat Tabel Baru

Kita akan membuat sebuah tabel baru dengan menggunakan perintah "**CREATE TABLE**". Contohnya sebagai berikut..

```
mysql> create table karyawan ;
ERROR 1113 (42000): A table must have at least 1 column
```

Ternyata ada kesalahan yang terjadi. Untuk membuat sebuah tabel di MySQL, kita harus menentukan minimal satu buah field/kolom di dalamnya. Sekarang kita ubah perintah di atas menjadi sebagai berikut:

```
mysql> create table karyawan
    -> (nopeg INT UNSIGNED AUTO_INCREMENT PRIMARY KEY,
    -> nama VARCHAR(50) NOT NULL)
    -> ;
    Query OK, 0 rows affected (0.14 sec)
```

Cukup panjang ya perubahan perintahnya. Mungkin sintaks perintahnya agak membingungkan pada awalnya. Tidak apa-apa, nanti akan kita bahas artinya. Secara umum, kita akan membuat sebuah **tabel Karyawan** dengan 2 buah kolom/field.

- Kolom pertama adalah **NOPEG** dengan jenis data bilangan bulat (**INTeger**), tanpa tanda negatif (**UNSIGNED**), yang akan bertambah nilainya secara otomatis (**AUTO_INCREMENT**), kolom NOPEG adalah kolom utama (**PRIMARY KEY**).
- Pada kolom kedua, **NAMA** akan menampung nama karyawan, dengan jenis data **VARiabel CHARacter**, lebar datanya dapat menampung maksimal 50 karakter, dan tidak boleh dikosongkan (**NOT NULL**). Kurang lebih seperti itulah ceritanya.. :)

Kita lihat kembali apa isi dari database *latihan1*:

```
mysql> show tables ;
+--------------------+
| Tables_in_latihan1 |
+--------------------+
| karyawan           |
+--------------------+
1 row in set (0.00 sec)
```

Dari hasil perintah di atas, kita lihat bahwa database *latihan1* telah memiliki sebuah tabel yang bernama *karyawan*. Selanjutnya kita akan lihat apa struktur dari tabel *karyawan* tersebut.

2.10 Melihat Struktur Tabel

Untuk melihat struktur sebuah tabel dapat menggunakan perintah "**DESCRIBE**" atau bisa juga menggunakan perintah "**SHOW COLUMNS FROM**". Contohnya berikut ini:

```
mysql> describe karyawan ;
+------+----------------+-----+----+-------+--------------+
|Field |Type            | Null|Key |Default|Extra         |
+------+----------------+-----+----+-------+--------------+
|nopeg |int(10) unsigned| NO  |PRI |NULL   |auto_increment|
|nama  |varchar(50)     | NO  |    |       |              |
+------+----------------+-----+----+-------+--------------+
2 rows in set (0.02 sec)
```

Atau menggunakan perintah "SHOW COLUMNS FROM..."

```
mysql> show columns from karyawan ;
+-----+---------------+----+---+-------+--------------+
|Field|Type           |Null|Key|Default|Extra         |
+-----+---------------+----+---+-------+--------------+
|nopeg|int(10) unsigned|NO  |PRI|NULL   |auto_increment|
|nama |varchar(50)    |NO  |   |       |              |
+-----+---------------+----+---+-------+--------------+
2 rows in set (0.00 sec)
```

Tidak ada perbedaan hasil dari dua perintah di atas, bukan? Sekarang kita buat sebuah tabel baru lagi, kita namakan saja tabel *contoh1*.

```
mysql> create table contoh1
    -> (noid INT)
    -> ;
Query OK, 0 rows affected (0.13 sec)
```

Sekarang kita lihat berapa tabel yang ada di dalam database *latihan1*:

```
mysql> show tables ;
+--------------------+
| Tables_in_latihan1 |
+--------------------+
| contoh1            |
| karyawan           |
+--------------------+
2 rows in set (0.00 sec)
```

2.11 Menghapus Tabel

Tabel *contoh1* yang baru saja kita buat ini akan kita hapus kembali. Perintah untuk menghapus sebuah tabel dalam MySQL adalah "**DROP TABLE**". Cukup mirip dengan perintah menghapus database, bukan? Kita harus menggunakan perintah "DROP" ini dengan **kehati-hatian yang tinggi**. Sistem MySQL tidak akan memberikan peringatan awal atau konfirmasi untuk proses penghapusan tabel. Dan bila sudah dihapus, maka tabel tersebut tidak bisa lagi kita kembalikan. **Maka, berhati-hatilah!**

```
mysql> drop table contoh1 ;
Query OK, 0 rows affected (0.03 sec)
```

Kita lihat lagi tabel yang ada di dalam database *latihan1*:

```
mysql> show tables ;
+--------------------+
| Tables_in_latihan1 | +--------------------
+
| karyawan           | +--------------------
+
1 rows in set (0.00 sec)
```

2.12 Constraint

Constraint adalah batasan atau aturan yang ada pada tabel.

MySQL menyediakan beberapa tipe constraint berikut :

- NOT NULL

  Suatu kolom yang didefinisikan dengan constraint NOT NULL tidak boleh berisi nilai NULL. Kolom yang befungsi sebagai kunci primer (primary key) otomatis tidak boleh NULL.

- UNIQUE

  Mendefinisikan suatu kolom menjadi bersifat unik, artinya antara satu data dengan data lainnya namanya tidak boleh sama, misal alamat email.

- PRIMARY KEY

  Constraint PRIMARY KEY membentuk key yang unik untuk suatu tabel.

- FOREIGN KEY

  FOREIGN KEY constraint didefinisikan pada suatu kolom yang ada pada suatu table, dimana kolom tersebut juga dimiliki oleh table yang lain sebagai suatu PRIMARY KEY, biasa dipakai untuk menghubungkan antara 2 tabel.

Soal Latihan

Buat Database dengan nama dbKursus. Pilih dan buka database tersebut. Buat tabel dengan nama peserta untuk menyimpan data peserta meliputi : nomor, nama, email, alamat, kota.

Sedangkan strukturnya seperti tabel dibawah ini :

Kolom (*Field*)	Tipe Data (*Data Type*)
nomor	Char (8) Not Null Primary Key
nama	VarChar (20) Not Null
email	VarChar (30) Null
alamat	VarChar (20) Not Null
kota	VarChar (10) Not Null

Daftar Pustaka

http://dev.mysql.com

http://www.rohmat-mimi.com/download/MODUL PRAKTIKUM MY SQL-BASIS DATA

http://www.mysql.com

http://www.arbiedesign.com/index.php

Tim Training SMK-TI.Modul MySQL