

# Judul Artikel

**Untung Subagyo**

*uunboy@gmail.com*

*http://masuun.web.id*

## ***Lisensi Dokumen:***

*Copyright © 2003-2007 IlmuKomputer.Com*

*Seluruh dokumen di IlmuKomputer.Com dapat digunakan, dimodifikasi dan disebarkan secara bebas untuk tujuan bukan komersial (nonprofit), dengan syarat tidak menghapus atau merubah atribut penulis dan pernyataan copyright yang disertakan dalam setiap dokumen. Tidak diperbolehkan melakukan penulisan ulang, kecuali mendapatkan ijin terlebih dahulu dari IlmuKomputer.Com.*

## **Mengelola Database**

Sebelum kita membuat suatu tabel yang digunakan untuk menyimpan data, maka harus terlebih dahulu dibuat database yang merupakan kumpulan atau berisi tabel-tabel yang saling berhubungan dengan menggunakan kunci-kunci yang ditentukan. Tapi bagaimana caranya memerintahkan mysql untuk membuat database, tabel, dan lainnya yang kita perlukan.

Untuk itu kita perlu untuk mempelajari bahasa gaul yang akan digunakan untuk berkomunikasi dengan MySQL. Bahasa ini disebut dengan SQL (Structured Query Language) dan orang sering mengakronimnya dengan "sequel". SQL merupakan bahasa standar untuk pengolahan database. Ini berarti bahwa DBMS yang lain juga mengenal bahasa gaul ini. Walau ada beberapa istilah yang di salah satu DBMS tidak dikenal, tapi dikenal di DBMS yang lain.

Di dalam bahasa SQL, perintah dibedakan menjadi 3 sub bahasa:

- a. DDL (Data Definition Language)
- b. DML (Data Manipulation Language)
- c. DCL (Data Control Language)

## **Data Definition Language**

Kelompok perintah ini bisa digunakan untuk melakukan pendefinisian database dan pendefinisian tabel. Sehingga dengan menggunakan perintah-perintah ini, kita bisa memerintahkan untuk membuat database, membuat tabel, mengubah strukturnya, menghapus tabel, membuat index tabel dan lain-lain yang berhubungan dengan pendefinisian database dan tabel.

## **Membuat Database**

Untuk mengetahui atau melihat database yang sudah ada, bisa digunakan perintah:

```
mysql> show databases;  
+-----+  
| Database |  
+-----+  
| mysql    |  
| test     |  
+-----+
```

Dalam pembuatan database perlu perhatikan penulisan nama database tidak boleh menggunakan spasi dan karakter non standar. Bentuk penulisan perintah untuk membuat database baru adalah **create database <nama\_database>;**

```
mysql> create database perpustakaan;
```

Untuk memastikan bahwa database yang kita buat sudah jadi, perintahkan **show databases**, sehingga semua database yang ada di server dan bisa diakses oleh user akan ditampilkan.

```
mysql> show databases;
+-----+
| Database |
+-----+
| mysql    |
| perpustakaan |
| test     |
+-----+
```

### Membuat Tabel

Selanjutnya untuk memulai membuat tabel di dalam database perpustakaan, maka kita harus mengaktifkan terlebih dulu database perpustakaan dengan menggunakan perintah **use <nama\_database>**.

```
mysql> use perpustakaan;
```

Setelah database aktif, kita baru bisa memulai untuk membuat tabel yang kita perlukan. Pada database perpustakaan, akan disimpan data-data tentang buku, judul buku, kelompok buku, pengarang, dan penerbitnya. Selain itu juga untuk menyimpan data anggota beserta transaksi yang dilakukan.

Pertama kali, kita akan membuat tabel buku yang fungsinya untuk menyimpan data tentang buku.

```
mysql> create table buku (
-> id_buku bigint(5) NOT NULL auto_increment primary key,
-> index_buku varchar(15) NOT NULL default '',
-> no_urut tinyint(3) default 0,
-> referensi tinyint(1)) type=myISAM;
```

Untuk membuktikan bahwa kita berhasil, maka tabel yang ada database bisa kita tampilkan dengan cara:

```
mysql> show tables;
+-----+
| Tables_in_perpustakaan |
+-----+
| buku                    |
+-----+
```

Pada hasil perintah diatas ditampilkan, bahwa tabel yang sudah ada di database perpustakaan adalah buku. Untuk menampilkan struktur dari tabel buku, perintahkan:

```
mysql> desc buku;
+-----+-----+-----+-----+-----+-----+
| Field      | Type          | Null | Key | Default | Extra          |
+-----+-----+-----+-----+-----+-----+
| id_buku    | bigint(5)     | NO   | PRI | NULL    | auto_increment |
| id_judul   | varchar(15)   | NO   |     |          |                 |
| no_urut     | smallint(3)   | NO   |     | 0        |                 |
| referensi  | tinyint(1)    | NO   |     | 0        |                 |
| status     | tinyint(1)    | NO   |     | 0        |                 |
| no_inventory | varchar(15)   | NO   | UNI |          |                 |
+-----+-----+-----+-----+-----+-----+
```

**id\_buku** : menyimpan no\_id tidak buku walaupun judulnya sama, sifatnya unik. Dan semua field

yang lain bergantung hanya kepada id\_buku. Karena itu field ini dijadikan primary key.

**index\_buku** : digunakan untuk menyimpan indeks setiap judul buku yang tiap judul bukunya tersimpan di dalam tabel judul\_buku yang akan kita buat berikutnya.

**no\_urut** : digunakan untuk menyimpan no\_urut buku no\_urut buku untuk judul yang sama.

**referensi** : digunakan untuk mengetahui apakah buku boleh dipinjam/dibawa keluar atau tidak, jika boleh maka nilainya 0 jika tidak maka nilainya 1. Seperti ini apabila tiap judul buku ada yang boleh dibawa keluar ada yang tidak. Seandainya setiap judul buku yang sama memiliki kondisi referensi yang sama, maka field ini dimasukkan kedalam tabel judul\_buku.

Berikutnya kita akan membuat tabel judul\_buku.

```
mysql> create table judul_buku (
-> index_buku varchar(10) NOT NULL primary key,
-> judul varchar(90),
-> kd_pengarang varchar(5),
-> kd_kelompok smallint(3),
-> edisi tinyint(2),
-> thn_terbit int(4))type=myISAM;
```

**index\_buku** : buku dengan judul, pengarang, penerbit, dan edisi yang sama akan memiliki index yang sama.

**judul** : judul buku.

**kd\_pengarang** : kode pengarang yang nama pengarangnya ada di tabel pengarang.

**kd\_kelompok** : kode kelompok isi buku sesuai ilmu, yang domainnya dari tabel kelompok.

**edisi** : cetakan ke berapa untuk judul, penerbit, dan pengarang yang sama.

**thn\_terbit** : tahun terbit buku.

untuk tabel-tabel yang selanjutnya, coba buatlah sendiri dengan struktur sebagai berikut:

**bonus**: untuk mempermudah pengeditan perintah sql, kita bisa menuliskannya pada salah satu text editor, misalnya kwrite. Setelah disimpan, kemudian kita bisa menjalankannya dengan perintah **source <[/nama\_dir/]nama\_file>**

Tabel Kelompok

```
mysql> desc kelompok;
+-----+-----+-----+-----+-----+-----+
| Field | Type | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| kd_kelompok | smallint(3) | NO | PRI | 0 | |
| kelompok | varchar(20) | YES | | NULL | |
+-----+-----+-----+-----+-----+-----+
2 rows in set (0.06 sec)
```

Tabel Penerbit

```
mysql> desc penerbit;
+-----+-----+-----+-----+-----+-----+
| Field | Type | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| id_penerbit | varchar(5) | NO | PRI | | |
| penerbit | varchar(20) | YES | | NULL | |
| kota | varchar(20) | YES | | NULL | |
+-----+-----+-----+-----+-----+-----+
3 rows in set (0.00 sec)
```

Tabel Pengarang

```
mysql> desc pengarang;
+-----+-----+-----+-----+-----+-----+
| Field | Type | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| id_pengarang | varchar(5) | NO | PRI | | |
| nama | varchar(30) | NO | | | |
+-----+-----+-----+-----+-----+-----+
2 rows in set (0.02 sec)
```

Tabel Anggota

Field	Type	Null	Key	Default	Extra
kd_anggota	varchar(7)	NO	PRI		
nama_anggota	varchar(50)	NO			
tgllahir	date	YES		NULL	
jkln	tinyint(1)	YES		NULL	
alamat	varchar(50)	YES		NULL	
no_telp	varchar(15)	YES		NULL	
tmp_lahir	varchar(20)	YES		NULL	
sekolah	varchar(30)	YES		NULL	

Tabel Peminjam

Field	Type	Null	Key	Default	Extra
id_anggota	varchar(7)	NO	PRI		
id_buku	varchar(5)	YES		NULL	
tgl_pinjam	date	YES		NULL	
tgl_kembali	date	YES		NULL	

### Merubah dan Memodifikasi Tabel

Terkadang ketika kita, sedang membuat tabel, ternyata ada nama field yang kurang, atau nama\_fieldnya susah dipahami, atau bisa juga type data serta ukurannya tidak/ kurang sesuai. Sekarang coba kita perhatikan tabel yang sudah kita buat sebelumnya.

Pada tabel buku terdapat field id\_buku dengan type bigint ukurannya 5, sedangkan pada tabel peminjam id\_buku typenya varchar ukurannya 5. Ini tidak sama, bisa-bisa timbul kekacauan ini (*emangnya apaan, kok sampai terjadi kekacauan segala*) Maksudnya ini harus disamakan supaya kedua tabel bisa dihubungkan. Sekarang, id\_buku pada tabel peminjam diubah menjadi bertype bigint(5), caranya:

```
mysql> alter table peminjam modify id_buku bigint(5);
Query OK, 0 rows affected (0.00 sec)
Records: 0 Duplicates: 0 Warnings: 0
```

Coba sekarang ditampilkan lagi struktur dari tabel peminjam, adakah perubahan? (*harus dong... kita harus selalu berubah untuk menuju yang lebih baik...*)

Field	Type	Null	Key	Default	Extra
id_anggota	varchar(7)	NO	PRI		
id_buku	bigint(5)	YES		NULL	
tgl_pinjam	date	YES		NULL	
tgl_kembali	date	YES		NULL	

Sekarang kita perhatikan field index\_buku pada tabel buku dan pada tabel judul\_buku, ukurannya juga berbeda, nah sekarang tugas Anda/Antum semuanya adalah menyamakan dengan ukuran 15 pada tabel judul\_buku. Sudah tahu caranya? (*masa gitu aja gak tahu, sich*)

### Merubah Nama Field

Pada tabel buku terdapat field id\_buku dan index\_buku, kalau melihat namanya, maka seolah-olah fungsinya sama, kenapa ada dua field id(index) buku pada satu tabel. Padahal fungsi kedua field tersebut berbeda. Field index\_buku fungsinya untuk menyimpan indeks judul buku sedangkan field id\_buku untuk menyimpan indeks untuk setiap buku yang domainnya ada di tabel judul\_buku. Judul buku yang sama memiliki index\_buku yang sama tapi id\_bukunya berbeda. Sebetulnya namanya tetap seperti itu juga tidak ada masalah, tapi akan membingungkan. (*bagi orang-orang yang bingung... :)*)

Untuk lebih mempermudah pemahaman hanya dengan melihat namanya saja, maka sebaiknya namanya diganti dengan id\_judul. Caranya?

```
mysql> alter table buku change index_buku id_judul varchar(15);  
Query OK, 0 rows affected (0.00 sec)  
Records: 0 Duplicates: 0 Warnings: 0
```

Sudah tahu tugas berikutnya? Nah... Ubahlah field index\_buku yang ada di tabel judul\_buku dengan id\_judul. Ubah juga field yang berawalan dengan kd dengan id, misalnya kd\_penerbit menjadi id\_penerbit. Ini hanya untuk melatih cara merubah nama field serta untuk menyeragamkan penamaan saja. Selain itu, ubah juga field sekolah\_universitas menjadi sekolah saja. *Untuk apa panjang-panjang, universitas juga sekolah kan? Apakah SD itu universitas? Kenapa nggak sekolah\_sma, atau sekolah\_lpk, atau sekolah\_stmik saja, kenapa sekolah\_universitas? he... he... he... :)*

### Menambah Field

Kalau kita perhatikan tabel penerbit, dengan tabel manakah tabel ini berhubungan? Kalau tabel pengarang berhubungan dengan tabel judul\_buku menggunakan field kd\_pengarang / id\_pengarang (jika sudah diganti). Sehingga untuk mengetahui nama pengarang dari suatu judul buku tinggal menghubungkan tabel judul\_buku dengan pengarang menggunakan field kunci id\_pengarang. Harusnya tabel penerbit diperlukan juga oleh tabel judul\_buku untuk mengetahui nama penerbit dari suatu judul buku. Tapi jangan khawatir, SQL punya juga bahasa yang digunakan untuk menambah/menyisipkan field kedalam suatu tabel.

```
mysql> alter table judul_buku add id_penerbit varchar(5) after judul;
```

Perintah di atas digunakan untuk menyisipkan field id\_penerbit ke dalam tabel judul\_buku diletakkan setelah field judul. Untuk membuktikan, tampilkan saja struktur tabel judul\_buku.

### Menghapus Field

Di dalam tabel anggota, kita bisa melihat di sana terdapat field no\_hp dan no\_telp. Perlukah kedua-duanya disimpan? Tidak semua anggota punya no. hp, dan tidak pula semuanya memiliki no. telpon. *(macam mana pula ini, kok disimpan semuanya? ini pemborosan namanya. instruksi presiden kan disuruh hemat energi. Lho... kok sampai bbm?)* Untuk menghapus field no\_hp gunakan perintah berikut:

```
mysql> alter table anggota drop no_hp;  
Query OK, 0 rows affected (0.00 sec)  
Records: 0 Duplicates: 0 Warnings: 0
```

### Merubah Nama Tabel

Tabel peminjam berfungsi untuk menyimpan data anggota yang melakukan peminjaman dan pengembalian buku. Mungkin akan lebih jelasnya kalau nama peminjam diganti dengan transaksi. Untuk melakukannya gunakan perintah :

```
mysql> alter table peminjam rename transaksi;
```

### Menghapus Table dan Database

Terkadang kita sudah membuat suatu tabel atau database, tapi ternyata tabel tersebut sebetulnya tidak diperlukan. Mungkin karena pembatasan masalah, ataupun mungkin karena diketahui setelahnya bahwa tabel tersebut tidaklah diperlukan, dan hanya memboroskan tempat penyimpanan saja, atau dengan tabel tersebut menjadikan tidak efektif dan efisien. Atau bisa saja tabel tersebut diciptakan hanya untuk keperluan sementara saja(temporari), maka kita harus atau perlu untuk menghapus tabel tersebut. Begitu juga dengan database yang sudah kita buat ,ternyata database tersebut sudah kita perlukan lagi. Perintah yang digunakan untuk menghapus adalah **DROP**.

### Menghapus Tabel

struktur perintah:

**drop table** <nama\_table>

contoh:

```
mysql> drop table pengarang;
```

### Menghapus Database

struktur perintah:

**drop database** <nama\_table>

contoh:

```
mysql> drop database pengarang;
```

Tampilkan semua tabel yang ada di dalam database, atau tampilkan semua database yang ada di dalam server, masih adakah tabel atau database yang diberlakukan padanya perintah drop? Jika perintah yang dilakukan sesuai dengan aturannya (*syariat*) tanpa mengurangi ataupun menambahnya (*dengan kebid'ahan-kebid'ahan*), pasti akan mendapatkan hasilnya sesuai yang sudah disebutkan (*dijanjikan, kecuali jika ada kesalahan penulisan maka diluar tanggung jawab percetakan. Iho...?*).

### Biografi Penulis



**Untung Subagyo.** Menyelesaikan D3 di universitas Gadjah Mada Progam Studi Komputer dan Sistem Informasi tahun 2004. Sebelumnya belajar dulu di LPK El Rahma selama 1 tahun. Setelah lulus dari D3 kembali mengabdikan di STMIK El Rahma Yogyakarta (<http://stmikelrahma.ac.id>), sebagai mahasiswa sekaligus sebagai asisten dosen. Selain itu juga merangkap sebagai Programmer agak ecek2 di beberapa tempat terpencil di sudut kota-kota di Jawa Tengah..