ELEX MEDIA KOMPUTINDO

# MySQL untuk Pemula

Referensi kilat tentang MySQL yang paling cocok bagi pemula

Jubilee Enterprise

rumpitekno.com

# MySQL untuk Pemula

Sanksi Pelanggaran Pasal 72
Undang-Undang Nomor 19 Tahun 2002
Tentang HAK CIPTA

1. Barangsiapa dengan sengaja dan tanpa hak melakukan perbuatan sebagaimana dimaksud dalam Pasal 2 Ayat (1) atau Pasal 49 Ayat (1) dan Ayat (2) dipidana dengan pidana penjara masing-masing paling singkat 1 (satu) bulan dan/atau denda paling sedikit Rp1.000.000 (satu juta rupiah), atau pidana penjara paling lama 7 (tujuh) tahun dan/atau denda paling banyak Rp5.000.000.000 (lima miliar rupiah).
2. Barangsiapa dengan sengaja menyiarkan, memamerkan, mengedarkan, atau menjual kepada umum suatu ciptaan atau barang hasil pelanggaran hak cipta atau hak terkait sebagai dimaksud pada Ayat (1) dipidana dengan pidana penjara paling lama 5 (lima) tahun dan/atau denda paling banyak Rp500.000.000 (lima ratus juta rupiah).

# MySQL untuk Pemula

**Jubilee Enterprise**

PENERBIT PT ELEX MEDIA KOMPUTINDO

**MySQL untuk Pemula**

**Jubilee Enterprise**

Diterbitkan pertama kali oleh
Penerbit PT Elex Media Komputindo
Kelompok Gramedia, Anggota IKAPI, Jakarta 2014

nkfadli@elexmedia.co.id

121142516
ISBN: 978-602-02-5390-9



Dicetak oleh Percetakan PT Gramedia, Jakarta
Isi di luar tanggung jawab percetakan

# Kata Pengantar

Mengapa MySQL penting untuk dipelajari? Jika ingin melangkah dari awal untuk masuk ke dalam dunia database, maka MySQL merupakan pijakan yang paling tepat. Saat ini, hampir sebagian besar situs-situs di internet dibuat menggunakan PHP. Dan, PHP plus MySQL merupakan kombinasi yang paling populer, tepat, dan presisi untuk dunia web programming.

Jadi, mempelajari MySQL merupakan bagian dari investasi masa depan karena perangkat lunak database ini sudah menjadi standar dalam pemrograman berbasis database.

Buku ini ditujukan untuk para pemula. Bagi Anda yang ingin mulai belajar MySQL, pasti akan merasa cocok membaca buku ini. Diharapkan, setelah membaca buku ini, Anda akan mengenal MySQL dengan baik.

Yogyakarta, 9 Oktober 2014

Gregorius Agung
**Founder Jubilee Enterprise**
"Information Technology is Our Passion and Book is Our Way"
Do you need top-notch IT Book? Just .... thinkjubilee.com

# Daftar Isi

# BAB 1

# Mengenal Database

## Apa itu Database?

Database adalah suatu aplikasi yang menyimpan sekumpulan data. Setiap database mempunyai API tertentu untuk membuat, mengakses, mengatur, mencari, dan menyalin data yang ada di dalamnya.

Untuk menampung dan mengatur data yang begitu banyak, Anda dapat menggunakan Relational Database Management Systems (RDBMS). Hal ini disebut relational database karena semua data disimpan dalam tabel-tabel yang berbeda dan dihubungkan berdasarkan relasinya dengan menggunakan primary key dan foreign key. Relational Database Management Systems (RDBMS) adalah software yang:

- Memungkinkan Anda untuk mengimplimentasikan sebuah database dengan tabel-tabel, kolom-kolom, dan indeks-indeks.
- Menjamin integritas referensi di antara baris-baris pada berbagai tabel.
- Meng-update indeks-indeks secara otomatis.
- Menginterpretasikan query SQL dan menggabungkan informasi dari berbagai tabel.

## Terminologi RDBMS

Berikut ini adalah istilah-istilah yang digunakan dalam database:

- **Database**: merupakan kumpulan tabel-tabel yang berisi data-data yang saling berkaitan.
- **Tabel**: merupakan matriks berisi data. Tabel dalam database terlihat seperti spreadsheet sederhana.
- **Kolom**: satu kolom (elemen data) mengandung data dengan satu jenis yang sama.
- **Baris**: sebuah baris (masukan atau rekaman data) merupakan sekumpulan data yang berhubungan.
- **Redundancy**: menyimpan data dua kali secara redundant untuk membuat sistem berjalan lebih cepat.
- **Primary Key**: key yang bersifat unik. Sebuah nilai key tidak dapat digunakan dua kali dalam satu tabel.
- **Foreign Key**: merupakan penghubung antara dua tabel.
- **Compound Key**: atau disebut juga composite key merupakan key yang terdiri dari beberapa kolom.
- **Indeks**: merupakan indeks dalam database yang menyerupai indeks pada buku.
- **Integritas referensial**: digunakan untuk memastikan nilai foreign selalu mengacu pada suatu baris yang ada.

## Database MySQL

MySQL adalah RDBMS yang cepat dan mudah digunakan, serta sudah banyak digunakan untuk berbagai kebutuhan. MySQL dikembangkan oleh MySQL AB Swedia.

Berikut ini hal-hal yang menyebabkan MySQL menjadi begitu populer:

- Berlisensi open-source, sehingga Anda dapat menggunakannya secara gratis.
- Merupakan program yang powerful dan menyediakan fitur yang lengkap.
- Menggunakan bentuk standar bahasa data SQL.

- Dapat bekerja dengan banyak sistem operasi dan dengan bahasa-bahasa pemrograman seperti PHP, PERL, C, C++, JAVA, dan lain-lain.
- Bekerja dengan cepat dan baik, bahkan dengan data set yang banyak.
- Sangat mudah digunakan dengan PHP untuk pengembangan aplikasi web.
- Mendukung banyak database, sampai 50 juta baris atau lebih dalam suatu tabel.
- Dapat dikostumisasi sesuai dengan keinginan Anda.

# BAB 2

# Menginstal MySQL

Buku ini akan mengajarkan MySQL dengan kaitannya dengan PHP. Untuk itu Anda terlebih dahulu perlu mempersiapkan server development untuk dapat menggunakan MySQL dan PHP secara bersamaan.

Untuk itu Anda memerlukan server yang akan berperan sebagai web server secara offline. Pada bab ini Anda akan belajar bagaimana cara mempersiapkan server development.

## WAMP, MAMP, dan LAMP

WAMP, MAMP, dan LAMP adalah singkatan dari "Windows, Apache, MySQL, dan PHP", "Mac, Apache, MySQL, dan PHP", dan "Linux, Apache, MySQL, dan PHP". Ketiga server development ini menyediakan pengaturan lengkap yang diperlukan untuk membangun sebuah halaman web yang dinamis.

Dengan server development tersebut, Anda dapat dengan mudah menginstal Apache, MySQL, dan PHP sekaligus dalam satu paket. Anda hanya perlu mendownload dan menginstal satu kali saja untuk mendapatkan fitur lengkap Apache, MySQL, dan PHP.

## Menginstal WAMP di Windows

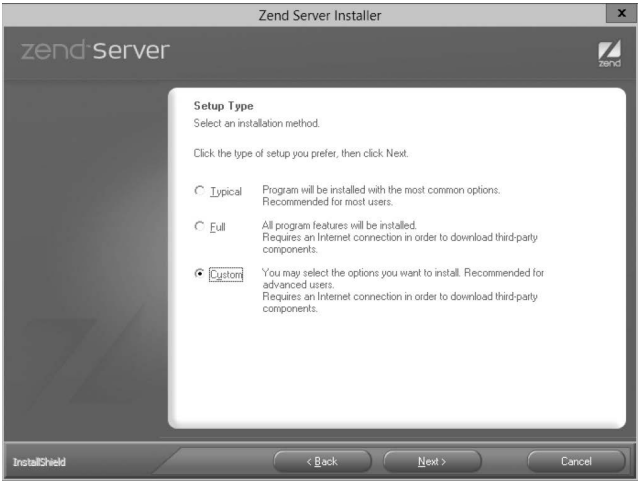
Ada beberapa server WAMP yang tersedia untuk Anda gunakan, dan dalam buku ini Anda akan menggunakan edisi gratis dari Zend Server. Anda dapat men-download-nya melalui situs ***http://tinyurl.com/zendfree*** seperti terlihat pada gambar berikut.


*Situs download Zend Server Free Edition*

Setelah Anda men-download **Zend Server Free Edition**, jalankan file installer untuk mulai menginstal.


*Jendela utama installer Zend Server*

Klik **Next** dan pilih **Accept the license agreement** untuk melanjutkan ke jendela **Setup Type**. Pilih **Custom** sehingga server MySQL dapat diinstal juga.




*Memilih pilihan Custom*

Pada jendela **Custom Setup**, gulir pilihan menu ke bawah dan pastikan bahwa **MySQL Server** sudah dicentang. Setelah itu klik **Next**.


*Centang pilihan MySQL Server*

Pilih **Install an Apache Web Server,** kemudian klik **Next**.


*Menginstal web server Apache*

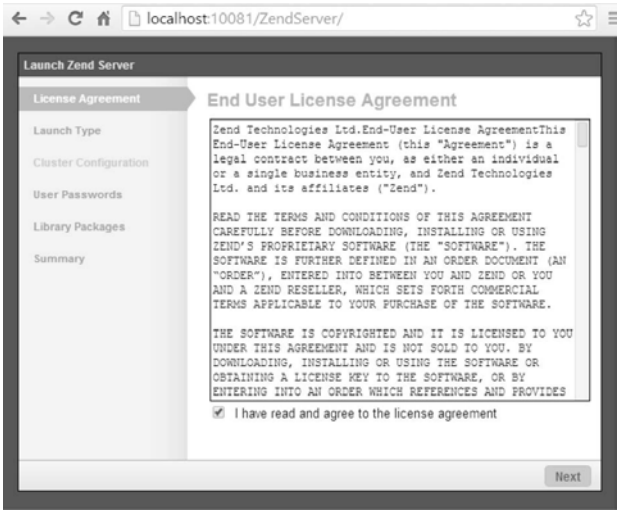
Atur nilai default **Web Server Port** menjadi **80**, dan **Zend Server Interface Port** menjadi **10081**. Kemudian klik **Next**.


*Mengatur port server*

Setelah itu, klik **Install** pada jendela **Installation Settings**.

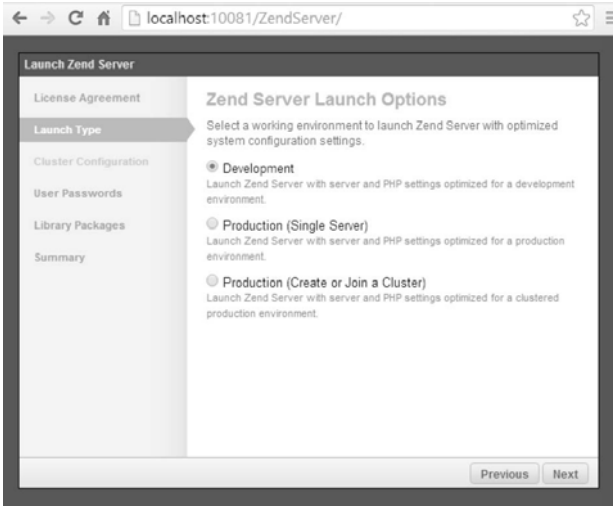

*Jendela Installation Settings*

Setelah instalasi selesai, Anda dapat langsung menggunakan software dengan mengklik **Finish**. Browser default Anda akan terbuka. Pilih kotak **I have read and agree** untuk melanjutkan.



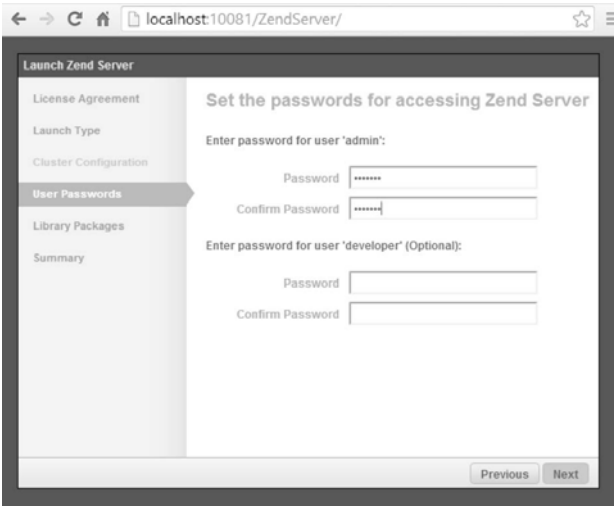

*Pastikan Anda memberi centang pada kotak I have read and agree*

Pada Tab **Launch Type** pilih **Development,** kemudian klik **Next**.



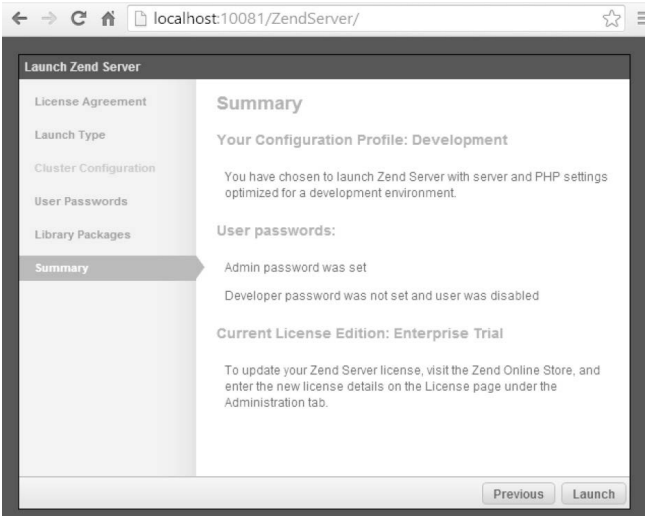

*Memilih pilihan Development*

Atur password untuk admin pengguna. Anda tidak perlu mengatur password untuk developer pengguna.




*Masukkan password untuk admin dua kali*

Klik **Next** pada jendela **Library Packages,** kemudian klik **Launch** untuk menyelesaikan proses instalasi.



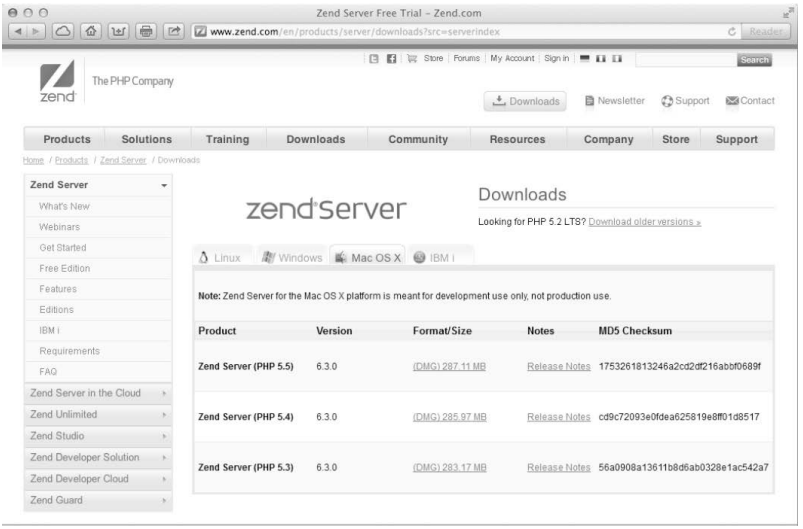

***Klik Submit untuk menyelesaikan pengaturan***

Setelah itu, browser akan menampilkan **Dashboard server**.


***Dashboard Zend Server***

## Menginstal MAMP di Mac OS X

**Zend Server Free Edition** juga dapat digunakan pada OS X. Anda dapat men-download-nya melalui situs ***http://tinyurl.com/zendfree***.




*Mendownload Zend Server untuk Mac OS X*

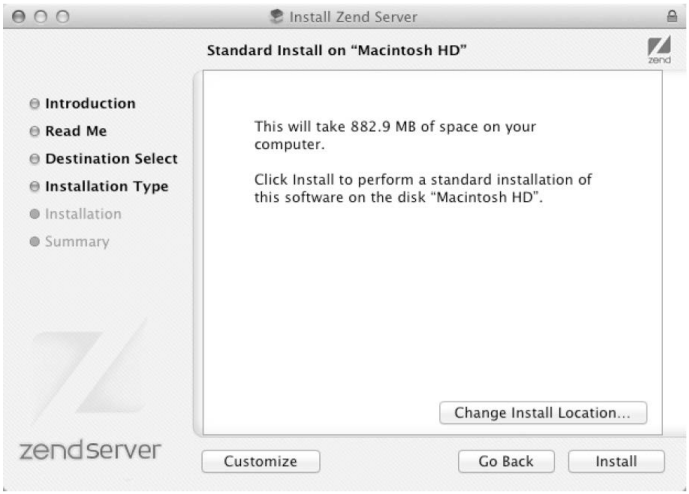
Setelah installer selesai didownload, klik-ganda file **.dmg**.


*Klik-ganda Zend Server untuk menginstal*

Jendela instalasi akan muncul. Klik **Continue**, akan muncul instruksi penginstalan. Klik **Continue** lagi untuk melanjutkan.


*Jendela instalasi*

Pilih lokasi penginstalan, klik **Install** dan masukkan password.




*Mulai menginstal Zend Server*

Setelah selesai menginstal, cari program ZendServer di folder **Applications** dan klik-ganda untuk menyelesaikan pengaturan. Browser akan terbuka dan Anda dapat mengikuti langkah-langkah yang sama dengan instalasi WAMP pada Windows.

## Konfigurasi MySQL

Installer MAMP tidak dilengkapi dengan command untuk memulai dan menutup MySQL server, sehingga Anda harus melakukannya secara manual dengan membuka Terminal dan menuliskan command berikut:

```
sudo nano /usr/local/zend/bin/zendctl.sh
```

Setelah memasukkan password, editor teks **Nano** akan terbuka. Pindah-kan kursor ke bawah sampai Anda menemukan baris MySQL_EN="false". Ubah nilainya menjadi true.

Setelah itu, terus gulir ke bawah sampai Anda menemukan baris berikut:

```
case $1 in
            "start")
                    ….
                    $0 start-apache %
```

Tepat di bawah baris tersebut, tambahkan baris berikut:

```
                    $0 start-MySQL %
```

Kemudian terus gulir ke bawah sampai Anda menemukan bagian berikut:

```
"stop")
                    ….
                    $0 stop-apache %
```

Tepat di bawah baris tersebut, tambahkan baris berikut:

```
                    $0 stop-MySQL %
```

Setelah itu tekan **Ctrl-X** untuk keluar dari mode edit, tekan **Y**, dan kemudian tekan **Return** untuk menyimpan file.

# BAB 3

# Mengatur Administrasi MySQL

Setelah selesai menginstal, Anda dapat menggunakan MySQL melalui jendela command prompt. Perintah-perintah MySQL dapat langsung Anda masukkan dalam command prompt.

## Menjalankan dan Menghentikan Server MySQL

Pertama-tama, Anda perlu memeriksa apakah server MySQL sudah dijalankan atau belum. Anda dapat memeriksanya dengan menggunakan perintah berikut:

```
ps -ef | grep mysqld
```

Jika MySQL sudah dijalankan, maka Anda akan melihat proses mysqld sudah terdaftar. Jika server belum dijalankan, maka Anda dapat menjalankannya dengan menggunakan perintah berikut:

```
root@host# cd /usr/bin ./safe_mysqld &
```

Untuk menghentikan server MySQL, gunakan peritah berikut:

```
root@host# cd /usr/bin
./mysqladmin -u root -p shutdown
Enter password: ******
```

## Menyiapkan Akun Pengguna MySQL

Untuk menambahkan pengguna baru dalam MySQL, Anda hanya perlu menambahkan masukan baru dalam tabel user pada database MySQL.

Berikut ini contoh query untuk menambahkan pengguna baru (‘tamu’) dengan password ‘tamu123’:

```
root@host# mysql -u root -p
Enter password:*******
mysql> use mysql;
Database changed

mysql> INSERT INTO user
  (host, user, password, select_priv, insert_priv,
  update_priv) VALUES ('localhost', 'tamu',
  PASSWORD('tamu123'), 'Y', 'Y', 'Y');
```

Anda kemudian dapat menampilkan akun baru dengan query berikut:

```
mysql> SELECT host, user, password FROM user
  WHERE user = 'tamu';
+-----------+---------+------------------+
| host | user  | password   |
+-----------+---------+------------------+
| localhost | tamu  | 6f8c114b58f2ce9e |
+-----------+---------+------------------+
```

Cara lain untuk menambahkan akun pengguna adalah dengan menggunakan perintah SQL GRANT. Contoh berikut menambahkan pengguna ‘zara’ dengan password zara123 dalam database TUTORIAL:

```
root@host# mysql -u root -p password;
Enter password:*******
mysql> use mysql;
Database changed

mysql> GRANT SELECT,INSERT,UPDATE,DELETE,CREATE,DROP
  -> ON TUTORIALS.*
  -> TO 'zara'@'localhost'
  -> IDENTIFIED BY 'zara123';
```

Perintah ini juga akan menambahkan masukkan pada tabel user dalam database mysql.

## Perintah Administratif MySQL

Berikut ini perintah-perintah MySQL penting yang akan sering Anda gunakan:

- **USE** Namadatabase: ini digunakan untuk memilih database tertentu dalam area kerja MySQL.
- **SHOW DATABASES**: menampilkan daftar database yang dapat diakses oleh DBMS MySQL.
- **SHOW TABLES**: menampilkan tabel-tabel dalam database setelah sebuah database dipilih.
- **SHOW COLUMNS FROM** namatabel: menampilkan atribut, tipe atribut, informasi key, NULL dimungkinkan, nilai default, dan informasi tabel lainnya.
- **SHOW INDEX FROM** namatabel: menampilkan informasi rinci untuk setiap indeks dalam tabel, termasuk PRIMARY KEY.
- **SHOW TABLE STATUS LIKE** namatabel\G: menampilkan informasi rinci mengenai statistik dan performa DBMS MySQL.

# BAB 4

# Syntax PHP

MySQL dapat bekerja dengan baik dalam kombinasi dengan berbagai bahasa pemrograman seperti PERL, C, C++, JAVA, dan PHP. Di antara bahasa-bahasa tersebut, PHP adalah yang paling populer karena memiliki kemampuan untuk mengembangkan aplikasi berbasis web.

PHP menyediakan berbagai fungsi untuk mengakses database MySQL dan memanipulasi rekaman data di dalamnya. Anda dapat memanggil fungsi-fungsi ini dengan cara yang sama seperti memanggil fungsi PHP lainnya.

Fungsi PHP untuk MySQL memiliki format sebagai berikut:

```
mysql_fungsi(nilai,nilai,...);
```

Bagian kedua dari nama fungsi biasanya mendeskripsikan kegunaan dari fungsi tersebut:

```
mysqli_connect($koneksikan);
mysqli_query($koneksikan,"Statemen SQL");
```

Berikut ini contoh syntax generik PHP untuk memanggil fungsi-fungsi MySQL:

```
<html>

 <head>
  <title>PHP dengan MySQL</title>
 </head>

 <body>

  <?php
 $nilaibalik = mysql_function(nilai, [nilai,...]);

 if( !$nilaibalik )
 {
  die ( "Error: pesan error yang sesuai" );
 }
  ?>

 </body>

</html>
```

# BAB 5

# Membuat Koneksi MySQL

## Koneksi MySQL Menggunakan Binary MySQL

Anda dapat mengoneksikan database MySQL menggunakan binary mysql pada command prompt.

Berikut ini contoh sederhana untuk mengoneksikan server MySQL dari command prompt:

```
[root@host]# mysql -u root -p
Enter password:******
```

Perintah tersebut akan menghasilkan command prompt mysql> di mana Anda dapat mengeksekusi perintah-perintah SQL. Berikut ini hasil dari perintah di atas:

```
Welcome to the MySQL monitor. Commands end with ; or \g.
Your MySQL connection id is 2854760 to server version: 5.0.9

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.
```

Pada contoh di atas, root telah digunakan sebagai pengguna, tetapi Anda juga dapat menggunakan pengguna lainnya. Setiap pengguna dapat menjalankan semua operasi SQL sesuai dengan yang telah ditetapkan.

Anda kemudian dapat mengakhiri koneksi MySQL dengan menggunakan command exit pada prompt mysql>.

```
mysql> exit
Bye
```

## Koneksi MySQL Menggunakan Script PHP

PHP menyediakan fungsi mysql_connect() untuk membuka koneksi database. Fungsi ini menggunakan lima parameter (argumen) dan mengembalikan penanda link MySQL jika berhasil, dan nilai FALSE jika gagal. Berikut ini adalah syntaxnya:

```
koneksi
mysql_connect(server,pengguna,password,link_baru,flag_klien);
```

| Parameter | Deskripsi |
|---|---|
| Server | Opsional - nama host yang menjalankan server database. Jika tidak dispesifikasikan, maka nilai default adalah localhost:3036. |
| Pengguna | Opsional - nama pengguna yang mengakses database. Jika tidak dispesifikasikan, maka nilai default adalah nama pengguna pemilik proses server. |
| Password | Opsional - password pengguna yang mengakses database. Jika tidak dispesifikasikan, maka nilai default adalah password kosong. |
| link_baru | Opsional - jika panggilan kedua dibuat untuk mysql_connect() dengan argumen yang sama, tidak ada koneksi baru yang akan dibuat, melainkan penanda koneksi yang sudah dibuka yang akan dikembalikan. |
| client_flags | Opsional - kombinasi dari konstanta-konstanta berikut:<br>MYSQL_CLIENT_SSL - Menggunakan enkripsi SSL.<br>MYSQL_CLIENT_COMPRESS - Menggunakan protocol kompresi.<br>MYSQL_CLIENT_IGNORE_SPACE - |

| | |
|---|---|
| | Memungkinkan spasi setelah nama fungsi.<br><br>MYSQL_CLIENT_INTERACTIVE - Memungkinkan waktu nonaktif interaktif sebelum menutup koneksi. |

*Parameter dalam fungsi mysql_connect()*

Untuk menutup koneksi database Anda dapat menggunakan fungsi PHP mysql_close(). Fungsi ini menggunakan satu parameter yang merupakan koneksi yang dikembalikan oleh fungsi mysql_connect().

Berikut syntaxnya:

```
bool mysql_close ( resource $link_identifier );
```

Berikut ini contoh membuat koneksi dengan server MySQL dengan menggunakan PHP:

```
<html>

 <head>
  <title>Connecting MySQL Server</title>
 </head>

 <body>

  <?php
 $dbhost = 'localhost:3036';
 $dbpengguna = 'guest';
 $dbpass = 'guest123';
 $kon = mysql_connect($dbhost, $dbpengguna, $dbpass);

 if(! $kon )
 {
  die('Tidak dapat terkoneksi: ' . mysql_error());
 }

 echo 'Koneksi berhasil';
 mysql_close($kon);
  ?>

 </body>

</html>
```

# BAB 6

# Membuat Database

## Membuat Database Menggunakan mysqladmin

Anda memerlukan perizinan khusus untuk membuat atau menghapus sebuah database MySQL. Jika Anda memiliki akses pengguna root, Anda dapat membuat database dengan menggunakan binary mysql mysqladmin.

Berikut ini contoh membuat database dengan nama TUTORIAL:

```
[root@host]# mysqladmin -u root -p create TUTORIAL
Enter password:******
```

## Membuat Database Menggunakan Script PHP

PHP menggunakan fungsi mysql_query untuk membuat atau menghapus database MySQL. Fungsi ini menggunakan dua parameter dan mengembalikan nilai TRUE jika berhasil atau nilai FALSE jika gagal.

Berikut syntaxnya:

```
bool mysql_query( sql, koneksi );
```

| Parameter | Deskripsi |
|---|---|
| sql | Harus disertakan - query SQL untuk membuat atau menghapus database MySQL. |
| koneksi | Opsional - jika tidak dispesifikasikan, koneksi terakhir yang dibuka oleh mysql_connect yang akan digunakan. |

*Parameter dalam fungsi mysql_query()*

Berikut ini contoh membuat database dengan menggunakan script PHP:

```
<html>

 <head>
  <title>Membuat Database MySQL</title>
 </head>

 <body>

  <?php
 $dbhost = 'localhost:3036';
 $dbpengguna = 'root';
 $dbpass = 'rootpassword';
 $kon = mysql_connect($dbhost, $dbpengguna, $dbpass);

 if(! $kon )
 {
  die('Tidak dapat terkoneksi: ' . mysql_error());
 }

 echo 'Koneksi berhasil<br />';

 $sql = 'CREATE DATABASE TUTORIAL';
 $nilaibalik = mysql_query( $sql, $kon );

 if(! $nilaibalik )
 {
  die('Tidak dapat membuat database: ' . mysql_error());
 }

 echo "Database TUTORIAL berhasil dibuat\n";
 mysql_close($kon);
  ?>

 </body>

</html>
```

# BAB 7

# Menghapus Database

## Menghapus Database Menggunakan mysqladmin

Anda memerlukan perizinan khusus untuk membuat atau menghapus sebuah database MySQL. Jika Anda memiliki akses pengguna root, Anda dapat menghapus database menggunakan binary mysql mysqladmin.

Berikut ini contoh menghapus database TUTORIAL:

```
[root@host]# mysqladmin -u root -p drop TUTORIAL
Enter password:******
```

Perintah ini akan menghasilkan peringatan konfirmasi apakah Anda benar-benar ingin menghapus database atau tidak. Untuk menghapus database tekan y.

```
Dropping the database is potentially a very bad thing to do.
Any data stored in the database will be destroyed.

Do you really want to drop the 'TUTORIAL' database [y/N] y
Database "TUTORIAL" dropped
```

## Menghapus Database Menggunakan Script PHP

PHP menggunakan fungsi mysql_query untuk membuat atau menghapus database MySQL. Fungsi ini menggunakan dua parameter dan mengembalikan nilai TRUE jika berhasil atau nilai FALSE jika gagal.

Berikut syntaxnya:

```
bool mysql_query( sql, koneksi );
```

| Parameter | Deskripsi |
|---|---|
| sql | Harus disertakan - query SQL untuk membuat atau menghapus database MySQL. |
| koneksi | Opsional - jika tidak dispesifikasikan, koneksi terakhir yang dibuka oleh mysql_connect yang akan digunakan. |

*Parameter dalam fungsi mysql_query()*

Berikut contoh menghapus database dengan menggunakan script PHP:

```
<html>

 <head>
  <title>Menghapus Database MySQL</title>
 </head>

 <body>

  <?php
 $dbhost = 'localhost:3036';
 $dbpengguna = 'root';
 $dbpass = 'rootpassword';
 $kon = mysql_connect($dbhost, $dbpengguna, $dbpass);

 if(! $kon )
 {
  die('Tidak dapat terkoneksi: ' . mysql_error());
 }

 echo 'Koneksi berhasil<br />';

 $sql = 'DROP DATABASE TUTORIAL';
 $nilaibalik = mysql_query( $sql, $kon );

 if(! $nilaibalik )
 {
  die('Tidak dapat menghapus database: ' .
    mysql_error());
 }

 echo "Database TUTORIAL berhasil dihapus\n";
 mysql_close($kon);
  ?>

 </body>

</html>
```

# BAB 8

# Memilih Database

Setelah Anda terkoneksi dengan server MySQL, Anda perlu memilih database tertentu untuk digunakan. Hal ini dikarenakan adanya banyak database yang nantinya akan tersedia pada server MySQL.

## Memilih Database melalui Command Prompt

Untuk memilih suatu database dari prompt mysql>, Anda dapat menggunakan perintah use.

Berikut ini contoh memilih database bernama TUTORIAL:

```
[root@host]# mysql -u root -p
Enter password:******
mysql> use TUTORIAL;
Database changed
mysql>
```

Setelah Anda memilih database TUTORIAL, semua operasi yang dilaksanakan kemudian akan dijalankan pada database tersebut.

**Catatan:**

*Semua nama database, nama tabel, dan nama field tabel bersifat case sensitive. Untuk itu Anda perlu menggunakan nama yang sesuai saat menjalankan perintah SQL.*

## Memilih Database MySQL Menggunakan Script PHP

PHP menyediakan fungsi mysql_select_db() untuk memilih database. Fungsi ini mengembalikan nilai TRUE jika berhasil atau FALSE jika gagal.

Berikut syntaxnya:

```
bool mysql_select_db( nama_db, koneksi );
```

| Parameter | Deskripsi |
|---|---|
| nama_db | Harus disertakan - nama Database MySQL yang ingin dipilih. |
| koneksi | Opsional - jika tidak dispesifikasikan, maka koneksi terakhir yang dibuka oleh mysql_connect yang akan digunakan. |

*Parameter pada fungsi mysql_select_db()*

Berikut ini contoh memilih database dengan script PHP:

```
<html>

 <head>
  <title>Memilih Database MySQL</title>
 </head>

 <body>

  <?php
 $dbhost = 'localhost:3036';
 $dbpengguna = 'tamu';
 $dbpass = 'tamu123';
 $kon = mysql_connect($dbhost, $dbpengguna, $dbpass);

 if(! $kon )
 {
  die('Tidak dapat terkoneksi: ' . mysql_error());
 }

 echo 'Koneksi berhasil';

 mysql_select_db( 'TUTORIAL' );

 mysql_close($kon);
  ?>

 </body>

</html>
```

# BAB 9

# Tipe Data MySQL

Mengatur field-field pada tabel penting dilakukan untuk mengoptimalkan database. Anda harus menggunakan tipe dan ukuran field yang sesuai dengan kebutuhan.

Tipe field (atau kolom) juga direferensikan sebagai tipe data yang digunakan untuk menampung nilai-nilai dalam field.

MySQL menggunakan berbagai tipe data yang dibagi menjadi tiga kategori: numerik, tanggal dan waktu, dan tipe string.

## Tipe Data Numerik

Berikut ini daftar tipe data numerik yang digunakan dalam MySQL:

- **INT** - Nilai integer dengan ukuran normal yang dapat berupa signed atau unsigned. Jika berupa signed, range yang dapat digunakan berkisar dari -2147483648 sampai 2147483647. Jika berupa unsigned, range yang dapat digunakan berkisar dari 0 sampai 4294967295.
- **TINYINT** - Nilai integer yang sangat kecil yang dapat berupa signed atau unsigned. Jika berupa signed, range yang dapat digunakan berkisar dari -128 sampai 127. Jika berupa unsigned, range yang dapat digunakan berkisar dari 0 sampai 255.

- **SMALLINT** - Nilai integer yang kecil yang dapat berupa signed atau unsigned. Jika berupa signed, range yang dapat digunakan berkisar dari -32768 sampai 32767. Jika berupa unsigned, range yang dapat digunakan berkisar dari 0 sampai 65535.
- **MEDIUMINT** - Nilai integer medium yang dapat berupa signed atau unsigned. Jika berupa signed, range yang dapat digunakan berkisar dari -8388608 sampai 8388607. Jika berupa unsigned, range yang dapat digunakan berkisar dari 0 sampai 16777215.
- **BIGINT** - Nilai integer besar yang dapat berupa signed atau unsigned. Jika berupa signed, range yang dapat digunakan berkisar dari -9223372036854775808 sampai 9223372036854775807. Jika berupa unsigned, range yang dapat digunakan berkisar dari 0 sampai 18446744073709551615.
- **FLOAT(M,D)** - Angka desimal yang hanya dapat berupa signed. Anda dapat menentukan panjang nilai yang ditampilkan (M) dan jumlah angka belakang koma (D). Hal ini secara default berisi (10,2), di mana terdapat 2 angka di belakang koma dan terdiri dari 10 digit angka.
- **DOUBLE(M,D)** - Angka desimal presisi ganda yang hanya dapat berupa signed. Anda dapat menentukan panjang nilai yang ditampilkan (M) dan jumlah angka belakang koma (D). Hal ini secara default berisi (16,4), di mana terdapat 4 angka di belakang koma dan terdiri dari 16 digit angka. Tipe data DOUBLE sama dengan REAL.
- **DECIMAL(M,D)** - Angka desimal yang tidak dibungkus dan hanya dapat berupa signed. Pada desimal yang tidak dibungkus, setiap desimal memiliki korespondensi dengan satu byte. Untuk tipe data ini Anda harus menentukan panjang (M) dan jumlah angka belakang koma (D). tipe data DECIMAL sama dengan NUMERIC.

## Tipe Data Tanggal dan Waktu

Berikut ini daftar tipe data tanggal dan waktu dalam MySQL:

- **DATE** - Tanggal dengan format YYYY-MM-DD, di antara 1000-01-01 sampai 9999-12-31. Sebagai contoh, tanggal 30 Desember 1973 akan disimpan sebagai 1973-12-30.

- **DATETIME** - Kombinasi tanggal dan waktu dengan format YYYY-MM-DD HH:MM:SS, di antara 1000-01-01 00:00:00 sampai 9999-12-31 23:59:59. Sebagai contoh, pukul 15:30 sore pada tanggal 30 Desember 1973 akan disimpan sebagai 1973-12-30 15:30:00.
- **TIMESTAMP** - Tanda waktu di antara tengah malam pada 1 Januari 1970 sampai suatu waktu pada tahun 2037. Tipe ini terlihat seperti format DATETIME, hanya saja tidak menggunakan spasi dan tanda pemisah - atau : . Sebagai contoh, pukul 15:30 sore pada tanggal 30 Desember 1973 akan disimpan sebagai 19731230153000.
- **TIME** - Menyimpan nilai waktu dengan format HH:MM:SS. Contohnya 15:30:00 seperti pada contoh sebelumnya.
- **YEAR(M)** - Menyimpan nilai tahun dengan format 2 digit atau 4 digit. YEAR(2) memiliki rentang nilai dari tahun 1970 sampai 2069 (70 sampai 69). Sedangkan YEAR(4) memiliki rentang nilai dari tahun 1901 sampai 2155.

## Tipe Data String

Berikut ini daftar tipe data string dalam MySQL:

- **CHAR(M)** - String dengan panjang tetap antara 1 sampai 255 karakter.
- **VARCHAR(M)** - String dengan panjang bervariasi antara 1 sampai 255 karakter.
- **BLOB** atau **TEXT** - Sebuah field dengan panjang maksimum 65535 karakter. BLOB (Binary Large Objects) digunakan untuk menyimpan data binary yang besar, seperti gambar atau tipe file lainnya. Field yang didefinisikan sebagai TEXT juga menyimpan jumlah data yang besar; perbedaan di antara keduanya adalah BLOB bersifat case sensitive sedangkan TEXT tidak.
- **TINYBLOB** atau **TINYTEXT** - Sebuah kolom BLOB atau TEXT dengan panjang maksimum 255 karakter.
- **MEDIUMBLOB** atau **MEDIUMTEXT** - Sebuah kolom BLOB atau TEXT dengan panjang maksimum 16777215 karakter.
- **LONGBLOB** atau **LONGTEXT** - Sebuah kolom BLOB atau TEXT dengan panjang maksimum 4294967295 karakter.

- **ENUM** - Sebuah enumerasi atau daftar. Saat mendefinisikan sebuah ENUM, Anda membuat sebuah daftar item-item yang mana nilainya harus dipilih. Sebagai contoh, jika Anda ingin field mengandung nilai “A” atau “B” atau “C”, Anda akan mendefinisikannya ENUM (‘A’, ‘B’, ‘C’) dan hanya salah satu nilai-nilai tersebut (atau NULL) yang dapat digunakan dalam field.

# BAB 10

# Membuat Tabel

Untuk membuat tabel, Anda perlu menuliskan tiga hal berikut dalam perintah: nama tabel, nama-nama field, dan definisi untuk setiap field.

Berikut ini syntax SQL generik untuk membuat sebuah tabel MySQL;

```
CREATE TABLE nama_tabel (nama_kolom tipe_kolom);
```

Berikut ini contoh perintah untuk membuat tabel di dalam database TUTORIAL:

```
tabel_tutorial(
  id INT NOT NULL AUTO_INCREMENT,
  judul VARCHAR(100) NOT NULL,
  penulis VARCHAR(40) NOT NULL,
  terbitan DATE, PRIMARY KEY ( id )
);
```

Berikut ini penjelasannya:

- Atribut field NOT NULL digunakan karena field tersebut harus memiliki nilai dan tidak boleh kosong atau NULL.
- Atribut field AUTO_INCREMENT memberitahu MySQL untuk secara otomatis menambahkan angka selanjutnya pada field id.
- Kata kunci PRIMARY KEY digunakan untuk mendefinisikan sebuah kolom sebagai primary key.

## Membuat Tabel dari Command Prompt

Untuk membuat tabel MySQL melalui prompt mysql> Anda perlu menggunakan perintah CREATE TABLE.

Berikut ini contoh membuat tabel dengan nama tabel_tutorial:

```
root@host# mysql -u root -p
Enter password:*******

mysql> use TUTORIAL;
Database changed

mysql> CREATE TABLE tabel_tutorial(
   -> id INT NOT NULL AUTO_INCREMENT,
   -> judul VARCHAR(100) NOT NULL,
   -> penulis VARCHAR(40) NOT NULL,
   -> terbitan DATE,
   -> PRIMARY KEY ( id )
   -> );
Query OK, 0 rows affected (0.16 sec)

mysql>
```

## Membuat Tabel Menggunakan Script PHP

Untuk membuat tabel baru dalam sebuah database, Anda dapat menggunakan fungsi PHP mysql_query(). Anda perlu memberikan argumen kedua yang berupa perintah SQL untuk membuat sebuah tabel.

Berikut ini contoh membuat tabel menggunakan script PHP:

```
<html>

 <head>
  <title>Membuat Tabel MySQL</title>
 </head>

 <body>

  <?php
 $dbhost = 'localhost:3036';
 $dbpengguna = 'root';
 $dbpass = 'rootpassword';
 $kon = mysql_connect($dbhost, $dbpengguna, $dbpass);

 if(! $kon )
 {
  die('Tidak dapat terkoneksi: ' . mysql_error());
 }

 echo 'Koneksi berhasil<br />';

 $sql = "CREATE TABLE tabel_tutorial ( ".
   "id INT NOT NULL AUTO_INCREMENT, ".
```

```
    "judul VARCHAR(100) NOT NULL, ".
    "penulis VARCHAR(40) NOT NULL, ".
    "terbitan DATE, ".
    "PRIMARY KEY ( id )); ";

  mysql_select_db( 'TUTORIAL' );

  $nilaibalik = mysql_query( $sql, $kon );

  if(! $nilaibalik )
  {
   die('Tidak dapat membuat tabel: ' . mysql_error());
  }

  echo "Tabel berhasil dibuat\n";
  mysql_close($kon);
   ?>

  </body>

</html>
```

# BAB 11

# Menghapus Tabel

Anda dapat dengan mudah menghapus sebuah tabel dalam database MySQL. Sebelum menghapus sebuah tabel, pastikan data-data yang ada di dalam tabel sudah benar-benar tidak diperlukan lagi.

Berikut ini syntax SQL generik untuk menghapus sebuah tabel MySQL;

```
DROP TABLE nama_tabel;
```

## Menghapus Tabel dari Command Prompt

Untuk menghapus tabel MySQL melalui prompt mysql> Anda perlu menggunakan perintah DROP TABLE.

Berikut ini contoh menghapus tabel tabel_tutorial:

```
root@host# mysql -u root -p
Enter password:*******

mysql> use TUTORIAL;
Database changed

mysql> DROP TABLE tabel_tutorial
Query OK, 0 rows affected (0.8 sec)

mysql>
```

## Menghapus Tabel Menggunakan Script PHP

Untuk menghapus tabel dalam sebuah database, Anda dapat menggunakan fungsi PHP mysql_query(). Anda perlu memberikan argumen kedua yang berupa perintah SQL untuk menghapus tabel.

Berikut ini contoh menghapus tabel menggunakan script PHP:

```
<html>

 <head>
  <title>Menghapus Tabel MySQL</title>
 </head>

 <body>

  <?php
 $dbhost = 'localhost:3036';
 $dbpengguna = 'root';
 $dbpass = 'rootpassword';
 $kon = mysql_connect($dbhost, $dbpengguna, $dbpass);

 if(! $kon )
 {
  die('Tidak dapat terkoneksi: ' . mysql_error());
 }

 echo 'Koneksi berhasil<br />';

 $sql = "DROP TABLE tabel_tutorial”;

 mysql_select_db( ‘TUTORIAL’ );

 $nilaibalik = mysql_query( $sql, $kon );

 if(! $nilaibalik )
 {
  die('Tidak dapat menghapus tabel: ' . mysql_error());
 }

 echo "Tabel berhasil dibuat\n";
 mysql_close($kon);
  ?>

 </body>

</html>
```

# BAB 12

# Memasukkan Data

Untuk memasukkan data ke dalam tabel MySQL, Anda dapat menggunakan perintah SQL INSERT INTO. Anda dapat memasukkan data ke dalam tabel MySQL dengan menggunakan prompt mysql> atau dengan script seperti PHP.

Berikut ini syntax SQL generik untuk perintah INSERT INTO:

```
INSERT INTO nama_tabel ( field1, field2,...fieldN )
 VALUES
 ( nilai1, nilai2,...nilaiN );
```

## Memasukkan Data Melalui Command Prompt

Berikut ini contoh memasukkan data ke dalam tabel tabel_tutorial:

```
root@host# mysql -u root -p
Enter password:*******

mysql> use TUTORIAL;
Database changed

mysql> INSERT INTO tabel_tutorial
  ->(judul, penulis, terbitan)
  ->VALUES
  ->("Learn PHP", "John Poul", NOW());
Query OK, 1 row affected (0.01 sec)
```

```
mysql> INSERT INTO tabel_tutorial
  ->(judul, penulis, terbitan)
  ->VALUES
  ->("Learn MySQL", "Abdul S", NOW());
Query OK, 1 row affected (0.01 sec)

mysql> INSERT INTO tabel_tutorial
  ->(judul, penulis, terbitan)
  ->VALUES
  ->("JAVA Tutorial", "Sanjay", '2007-05-06');
Query OK, 1 row affected (0.01 sec)

mysql>
```

Pada contoh di atas, NOW() adalah fungsi MySQL yang mengembalikan nilai tanggal dan waktu pada saat perintah dijalankan.

## Memasukkan Data Menggunakan Script PHP

Anda dapat menggunakan perintah SQL INSERT INTO yang sama ke dalam fungsi PHP mysql_query() untuk memasukkan data ke dalam tabel MySQL.

Contoh berikut menggunakan tiga parameter dari pengguna dan akan memasukkannya ke dalam tabel MySQL:

```
<html>

 <head>
  <title>Menambahkan data baru ke dalam Database
  MySQL</title>
 </head>

 <body>

  <?php
  if (isset($_POST[‘tambahkan’]))
  {
 $dbhost = 'localhost:3036';
 $dbpengguna = 'root';
 $dbpass = 'rootpassword';
 $kon = mysql_connect($dbhost, $dbpengguna, $dbpass);

 if(! $kon )
 {
  die('Tidak dapat terkoneksi: ' . mysql_error());
 }

 if(! get_magic_quotes_gpc() )
 {
  $judul = addslashes
  ($_POST['judul']);
  $penulis = addslashes
  ($_POST['penulis']);
```

```
}
else
{
 $judul = $_POST['judul'];
 $penulis = $_POST['penulis'];
}
$terbitan = $_POST['terbitan'];

$sql = "INSERT INTO tabel_tutorial ".
  "(judul,penulis,terbitan) ".
  "VALUES ".
  "('$judul', '$penulis','$terbitan')";

mysql_select_db('TUTORIAL');

$nilaibalik = mysql_query( $sql, $kon );

if(! $nilaibalik )
{
 die('Tidak dapat memasukkan data: ' . mysql_error());
}

echo "Berhasil memasukkan data\n";

mysql_close($kon);
 }
 else
 {
?>
<form method="post" action="<?php $_PHP_SELF ?>">
<table width="600" border="0" cellspacing="1"
  cellpadding="2">
<tr>
 <td width="250">Judul Tutorial</td>
 <td>
  <input name="judul" type="text"
 id="judul"> </td>
</tr>
<tr>
 <td width="250">Penulis Tutorial</td>
 <td>
  <input name="penulis" type="text"
 id="penulis">
 </td>
</tr>
<tr>
 <td width="250">Tanggal terbitan [ yyyy-mm-dd ]</td>
 <td>
  <input name="terbitan" type="text"
 id="terbitan">
 </td>
</tr>
<tr>
 <td width="250"> </td>
 <td> </td>
</tr>
<tr>
 <td width="250"> </td>
 <td>
  <input name="tambahkan" type="submit" id="tambahkan"
  value="Tambahkan Tutorial"> </td>
</tr>
```

```
    </table>
    </form>
    <?php
     }
    ?>

   </body>

</html>
```

Pada contoh tersebut, fungsi get_magic_quotes_gpc() digunakan untuk memeriksa apakah konfigurasi magic quote sudah diatur atau belum. Jika nilai hasil dari fungsi ini adalah false, maka fungsi addslashes() akan dijalankan untuk menambahkan tanda - sebelum quote.

# BAB 13

# Mengambil dan Menampilkan Data

Untuk mengambil dan menampilkan data dari database MySQL, Anda dapat menggunakan perintah SQL SELECT. Anda dapat menggunakan perintah ini pada prompt mysql> atau dalam script seperti PHP.

Berikut ini syntax SQL generik untuk perintah SELECT untuk mengambil dan menampilkan data dari tabel MySQL:

```
SELECT field1, field2,...fieldN nama_tabel1, nama_tabel2...
[WHERE Clause]
[OFFSET M ][LIMIT N]
```

- Anda dapat menggunakan satu tabel atau lebih yang dipisahkan dengan tanda koma untuk menyertakan berbagai kondisi dengan menggunakan clause WHERE. Pada perintah SELECT, clause WHERE bersifat opsional.
- Anda dapat mengambil dan menampilkan satu field atau lebih dalam suatu perintah SELECT.
- Anda dapat menggunakan tanda bintang (*) sebagai pengganti field. Dengan tanda ini SELECT akan menampilkan semua field yang ada.

- Anda dapat menspesifikasikan kondisi dengan menggunakan clause WHERE.
- Anda dapat menspesifikasikan offset dengan menggunakan OFFSET.
- Anda dapat membatasi jumlah data yang ditampilkan dengan menggunakan atribut LIMIT.

## Mengambil dan Menampilkan Data Menggunakan Command Prompt

Berikut ini contoh perintah SELECT untuk mengambil dan menampilkan data dari tabel tabel_tutorial:

```
root@host# mysql -u root -p password;
Enter password:*******

mysql> use TUTORIAL;
Database changed

mysql> SELECT * from tabel_tutorial

+--------+--------------+--------------+--------------+
| id | judul   | penulis  | terbitan |
+--------+--------------+--------------+--------------+
| 1 | Learn PHP | John Poul | 2007-05-21  |
| 2 | Learn MySQL  | Abdul S  | 2007-05-21  |
| 3 | JAVA Tutorial | Sanjay  | 2007-05-21  |
+--------+--------------+--------------+--------------+
3 rows in set (0.01 sec)

mysql>
```

## Mengambil dan Menampilkan Data Menggunakan Script PHP

Anda dapat menggunakan perintah SELECT dalam fungsi mysql_query() pada PHP. Fungsi ini digunakan untuk mengeksekusi perintah SQL dan setelah itu fungsi mysql_fetch_array() dapat digunakan untuk mengambil semua data yang dipilih.

Fungsi ini mengembalikan baris data sebagai array asosiatif, array numerik, atau keduanya. Fungsi ini mengembalikan nilai FALSE jika tidak ada baris data lagi.

Berikut ini contoh untuk mengambil dan menampilkan data dari tabel tabel_tutorial:

```
<?php
 $dbhost = 'localhost:3036';
 $dbpengguna = 'root';
 $dbpass = 'rootpassword';
 $kon = mysql_connect($dbhost, $dbpengguna, $dbpass);

 if(! $kon )
 {
  die('Tidak dapat terkoneksi: ' . mysql_error());
 }

 $sql = 'SELECT id, judul, penulis, terbitan
   FROM tabel_tutorial';

 mysql_select_db('TUTORIAL');

 $nilaibalik = mysql_query( $sql, $kon );

 if(! $nilaibalik )
 {
  die('Tidak dapat mengambil data: ' . mysql_error());
 }

 while($baris = mysql_fetch_array($nilaibalik, MYSQL_ASSOC))
 {
  echo "ID :{$baris['id']} <br> ".
    "Judul: {$baris['judul']} <br> ".
    "Penulis: {$baris['penulis']} <br> ".
    "Tanggal penerbitan : {$baris['terbitan']} <br> ".
    "--------------------------------<br>";
 }

 echo "Data berhasil diambil\n";

 mysql_close($kon);
?>
```

Pada contoh di atas, konstanta MYSQL_ASSOC digunakan sebagai argumen kedua pada fungsi mysql_fetch_array(), sehingga fungsi mengembalikan nilai baris sebagai sebuah array asosiatif. Dengan array asosiatif, Anda dapat mengakses field dengan menggunakan nama.

PHP juga menyediakan fungsi lain yaitu fungsi mysql_fetch_assoc(), yang mengembalikan nilai baris sebagai sebuah array asosiatif.

Berikut ini contoh untuk menampilkan semua data dari tabel tabel_tutorial menggunakan fungsi mysql_fetch_assoc():

```
<?php
 $dbhost = 'localhost:3036';
 $dbpengguna = 'root';
 $dbpass = 'rootpassword';
 $kon = mysql_connect($dbhost, $dbpengguna, $dbpass);

 if(! $kon )
 {
  die('Tidak dapat terkoneksi: ' . mysql_error());
 }
```

```
 $sql = 'SELECT id, judul, penulis, terbitan
   FROM tabel_tutorial';

 mysql_select_db('TUTORIAL');

 $nilaibalik = mysql_query( $sql, $kon );

 if(! $nilaibalik )
 {
  die('Tidak dapat mengambil data: ' . mysql_error());
 }

 while($baris = mysql_fetch_assoc($nilaibalik)
 {
  echo "ID :{$baris['id']} <br> ".
    "Judul: {$baris['judul']} <br> ".
    "Penulis: {$baris['penulis']} <br> ".
    "Tanggal penerbitan : {$baris['terbitan']} <br> ".
    "--------------------------------<br>";
 }

 echo "Data berhasil diambil\n";

 mysql_close($kon);
?>
```

Anda juga dapat menggunakan konstanta MYSQL_NUM sebagai argumen kedua pada fungsi mysql_fetch_array(). Ini akan menyebabkan fungsi mengembalikan sebuah array dengan indeks numerik.

Berikut ini contoh penggunaannya:

```
<?php
 $dbhost = 'localhost:3036';
 $dbpengguna = 'root';
 $dbpass = 'rootpassword';
 $kon = mysql_connect($dbhost, $dbpengguna, $dbpass);

 if(! $kon )
 {
  die('Tidak dapat terkoneksi: ' . mysql_error());
 }

 $sql = 'SELECT id, judul, penulis, terbitan
   FROM tabel_tutorial';

 mysql_select_db('TUTORIAL');

 $nilaibalik = mysql_query( $sql, $kon );

 if(! $nilaibalik )
 {
  die('Tidak dapat mengambil data: ' . mysql_error());
 }

 while($baris = mysql_fetch_array($nilaibalik, MYSQL_NUM)
 {
  echo "ID :{$baris[0]} <br> ".
```

```
    "Judul: {$baris[1]} <br> ".
    "Penulis: {$baris[2]} <br> ".
    "Tanggal penerbitan : {$baris[3]} <br> ".
    "---------------------------------<br>";
 }

 echo "Data berhasil diambil\n";

 mysql_close($kon);
?>
```

## Melepaskan Memory

Anda dapat melepaskan cursor memory pada bagian akhir dari setiap statemen SELECT. Hal ini dapat dilakukan dengan menggunakan fungsi PHP mysql_free_result().

Berikut ini contoh penggunaannya:

```
<?php
 $dbhost = 'localhost:3036';
 $dbpengguna = 'root';
 $dbpass = 'rootpassword';
 $kon = mysql_connect($dbhost, $dbpengguna, $dbpass);

 if(! $kon )
 {
  die('Tidak dapat terkoneksi: ' . mysql_error());
 }

 $sql = 'SELECT id, judul, penulis, terbitan
   FROM tabel_tutorial';

 mysql_select_db('TUTORIAL');

 $nilaibalik = mysql_query( $sql, $kon );

 if(! $nilaibalik )
 {
  die('Tidak dapat mengambil data: ' . mysql_error());
 }

 while($baris = mysql_fetch_array($nilaibalik, MYSQL_NUM)
 {
  echo "ID :{$baris[0]} <br> ".
    "Judul: {$baris[1]} <br> ".
    "Penulis: {$baris[2]} <br> ".
    "Tanggal penerbitan : {$baris[3]} <br> ".
    "---------------------------------<br>";
 }

 mysql_free_result($nilaibalik);

 echo "Data berhasil diambil\n";

 mysql_close($kon);
?>
```

# BAB 14

# Clause WHERE

Anda dapat menggunakan clause WHERE untuk menyaring data-data apa saja yang akan ditampilkan. Dengan clause WHERE, data-data yang ditampilkan hanya data-data yang sesuai dengan kriteria tertentu.

Berikut ini syntax SQL generik dari perintah SELECT yang disertai dengan WHERE:

```
SELECT field1, field2,...fieldN nama_tabel1, nama_tabel2...
[WHERE kondisi1 [AND [OR]] kondisi2.....
```

- Anda dapat menggunakan satu tabel atau lebih yang dipisahkan dengan tanda koma untuk menyertakan berbagai kondisi dengan menggunakan clause WHERE.
- Anda dapat menspesifikasikan kondisi apa saja dengan menggunakan clause WHERE.
- Anda dapat menspesifikasikan lebih dari satu kondisi dengan menggunakan operator AND atau OR.
- Clause WHERE dapat juga digunakan dengan perintah SQL DELETE atau UPDATE.

Clause WHERE digunakan untuk membandingkan nilai yang diberikan dengan nilai field yang ada dalam tabel MySQL. Jika nilai yang diberikan sesuai dengan nilai field yang ada dalam tabel MySQL, maka baris tersebut akan diambil dan ditampilkan.

Berikut ini daftar operator yang dapat digunakan dengan clause WHERE.

Diasumsikan bahwa field A berisi nilai 10 dan field B berisi nilai 20, maka:

| Operator | Deskripsi | Contoh |
|---|---|---|
| = | Memeriksa apakah nilai kedua operan sama atau tidak, jika sama maka kondisi bernilai benar. | (A = B) bernilai tidak benar. |
| != | Memeriksa apakah nilai kedua operan sama atau tidak, jika tidak sama maka kondisi bernilai benar. | (A != B) bernilai benar. |
| > | Memeriksa apakah nilai operan di sebelah kiri lebih dari nilai operan di sebelah kanan, jika ya maka kondisi bernilai benar. | (A > B) bernilai tidak benar. |
| < | Memeriksa apakah nilai operan di sebelah kiri kurang dari nilai operan di sebelah kanan, jika ya maka kondisi bernilai benar. | (A < B) bernilai benar. |
| >= | Memeriksa apakah nilai operan di sebelah kiri lebih dari atau sama dengan nilai operan di sebelah kanan, jika ya maka kondisi bernilai benar. | (A >= B) bernilai tidak benar. |
| <= | Memeriksa apakah nilai operan di sebelah kiri kurang dari atau sama dengan nilai operan di sebelah kanan, jika ya maka kondisi bernilai benar. | (A <= B) bernilai benar. |

***Operator pada clause WHERE***

## Mengambil Data Melalui Command Prompt

Berikut ini contoh penggunaan perintah SELECT dengan clause WHERE dari tabel tabel_tutorial. Data yang dikembalikan adalah data dengan penulis Sanjay:

```
root@host# mysql -u root -p password;
Enter password:*******

mysql> use TUTORIAL;
Database changed

mysql> SELECT * from tabel_tutorial WHERE penulis='Sanjay';

+-------+-----------------+-------------+----------------+
| id  | judul  | penulis | terbitan  |
+-------+-----------------+-------------+----------------+
| 3 | JAVA Tutorial  | Sanjay | 2007-05-21 |
+-------+-----------------+-------------+----------------+
1 rows in set (0.01 sec)

mysql>
```

Pada contoh tersebut perbandingan tidak bersifat case sensitive. Anda dapat membuat pencarian yang bersifat case sensitive dengan menggunakan kata kunci BINARY:

```
root@host# mysql -u root -p password;
Enter password:*******

mysql> use TUTORIAL;
Database changed

mysql> SELECT * from tabel_tutorial \
   WHERE BINARY penulis='sanjay';
Empty set (0.02 sec)

mysql>
```

## Mengambil Data Menggunakan Script PHP

Anda dapat menggunakan perintah SQL SELECT dengan clause WHERE dalam fungsi PHP mysql_query(). Setelah itu, fungsi mysql_fetch_array() dapat digunakan untuk mengambil semua data yang terpilih. Fungsi ini mengembalikan nilai baris sebagai sebuah array asosiatif, array numerik, atau keduanya.

Berikut adalah contoh yang akan mengambil dan menampilkan semua data pada tabel tabel_tutorial yang field penulisnya adalah Sanjay:

```
<?php
 $dbhost = 'localhost:3036';
 $dbpengguna = 'root';
 $dbpass = 'rootpassword';
 $kon = mysql_connect($dbhost, $dbpengguna, $dbpass);

 if(! $kon )
 {
  die('Tidak dapat terkoneksi: ' . mysql_error());
 }

 $sql = 'SELECT id, judul, penulis, terbitan
   FROM tabel_tutorial'
   WHERE penulis="Sanjay"';

 mysql_select_db('TUTORIAL');

 $nilaibalik = mysql_query( $sql, $kon );

 if(! $nilaibalik )
 {
  die('Tidak dapat mengambil data: ' . mysql_error());
 }

 while($baris = mysql_fetch_array($nilaibalik, MYSQL_ASSOC))
 {
  echo "ID :{$baris['id']} <br> ".
    "Judul: {$baris['judul']} <br> ".
    "Penulis: {$baris['penulis']} <br> ".
    "Tanggal penerbitan : {$baris['terbitan']} <br> ".
    "--------------------------------<br>";
 }

 echo "Data berhasil diambil\n";

 mysql_close($kon);
?>
```

# BAB 15

# Meng-update Data

Untuk memodifikasi data yang ada dalam tabel MySQL, Anda dapat menggunakan perintah SQL UPDATE. Perintah ini akan memodifikasi nilai field yang ada dalam tabel.

Berikut ini syntax SQL generik dari perintah UPDATE untuk memodifikasi data dalam tabel:

```
UPDATE nama_tabel SET field1=nilai-baru1, field2=nilai-baru2
[WHERE Clause]
```

- Anda dapat meng-update satu field atau lebih secara bersamaan.
- Anda dapat menspesifikasikan kondisi dengan menggunakan clause WHERE.
- Anda dapat memodifikasi nilai-nilai dalam satu tabel pada saat bersamaan.

## Meng-update Data Melalui Command Prompt

Berikut ini contoh penggunaan perintah UPDATE dengan clause WHERE, untuk mengupdate data yang dipilih pada tabel tabel_tutorial:

```
root@host# mysql -u root -p password;
Enter password:*******

mysql> use TUTORIAL;
Database changed

mysql> UPDATE tabel_tutorial
  -> SET judul='Learning JAVA'
  -> WHERE id=3;
Query OK, 1 row affected (0.04 sec)
Rows matched: 1 Changed: 1 Warnings: 0

mysql>
```

## Meng-update Data Menggunakan Script PHP

Anda dapat menggunakan perintah SQL UPDATE dengan clause WHERE dalam fungsi PHP mysql_query().

```
<?php
 $dbhost = 'localhost:3036';
 $dbpengguna = 'root';
 $dbpass = 'rootpassword';
 $kon = mysql_connect($dbhost, $dbpengguna, $dbpass);

 if(! $kon )
 {
  die('Tidak dapat terkoneksi: ' . mysql_error());
 }

 $sql = 'UPDATE tabel_tutorial
   SET judul="Learning JAVA"
   WHERE id=3';

 mysql_select_db('TUTORIAL');

 $nilaibalik = mysql_query( $sql, $kon );

 if(! $nilaibalik )
 {
  die('Tidak dapat mengupdate data: ' . mysql_error());
 }

 echo "Data berhasil diupdate\n";

 mysql_close($kon);
?>
```

# BAB 16

# Menghapus Data

Untuk menghapus data yang ada dalam tabel MySQL, Anda dapat menggunakan perintah SQL DELETE FOM.

Berikut ini syntax SQL generik dari perintah DELETE untuk menghapus data dalam tabel:

```
DELETE FROM nama_tabel [WHERE Clause]
```

- Jika clause WHERE tidak dispesifikasikan, maka semua data akan dihapus dari tabel yang dipilih.
- Anda dapat menspesifikasikan kondisi dengan menggunakan clause WHERE.
- Anda dapat menghapus data-data dalam satu tabel pada saat bersamaan.

## Menghapus Data Melalui Command Prompt

Berikut ini contoh penggunaan perintah DELETE dengan clause WHERE untuk menghapus data yang dipilih pada tabel tabel_tutorial:

```
root@host# mysql -u root -p password;
Enter password:*******

mysql> use TUTORIAL;
Database changed

mysql> DELETE FROM tabel_tutorial WHERE id=3;
Query OK, 1 row affected (0.23 sec)

mysql>
```

## Menghapus Data Menggunakan Script PHP

Anda dapat menggunakan perintah SQL DELETE dengan clause WHERE dalam fungsi PHP mysql_query().

```
<?php
 $dbhost = 'localhost:3036';
 $dbpengguna = 'root';
 $dbpass = 'rootpassword';
 $kon = mysql_connect($dbhost, $dbpengguna, $dbpass);

 if(! $kon )
 {
  die('Tidak dapat terkoneksi: ' . mysql_error());
 }

 $sql = 'DELETE FROM tabel_tutorial
   WHERE id=3';

 mysql_select_db('TUTORIAL');

 $nilaibalik = mysql_query( $sql, $kon );

 if(! $nilaibalik )
 {
  die('Tidak dapat menghapus data: ' . mysql_error());
 }

 echo "Data berhasil dihapus\n";

 mysql_close($kon);
?>
```

# BAB 17

# Clause Like

Anda telah menggunakan perintah SQL SELECT untuk mengambil data dari tabel MySQL dan menambahkan clause WHERE untuk memilih data mana saja yang akan diambil.

Clause WHERE dengan tanda = dapat digunakan untuk mencari nilai yang sama. Tapi, ada kalanya Anda ingin mengambil data yang memiliki sedikit kesamaan saja (tidak sama persis).

Untuk mengambil data yang memiliki sedikit kesamaan, Anda dapat menggunakan clause SQL LIKE. Jika clause LIKE digunakan dengan karakter %, maka tanda itu akan bekerja seperti karakter meta yang dapat berisi apa saja.

Berikut ini syntax SQL generik untuk perintah SELECT dengan menggunakan clause LIKE:

```
SELECT field1, field2,...fieldN nama_tabel1, nama_tabel2...
WHERE field1 LIKE kondisi1 [AND [OR]] field2 = 'nilaitertentu'
```

- Anda dapat menspesifikasikan kondisi menggunakan clause WHERE.
- Anda dapat menggunakan clause LIKE bersama dengan clause WHERE.
- Anda dapat menggunakan clause LIKE sebagai pengganti tanda =.
- Saat LIKE digunakan bersama dengan tanda %, tanda tersebut akan bekerja seperti karakter meta yang dapat berisi apa saja.
- Anda dapat menspesifikasikan lebih dari satu kondisi dengan menggunakan operator AND atau OR.
- Clause WHERE....LIKE dapat juga digunakan bersama dengan perintah DELETE atau UPDATE.

## Menggunakan Clause LIKE pada Command Prompt

Berikut ini contoh penggunaan clause WHERE....LIKE untuk mengambil data yang dipilih dari tabel tabel_tutorial.

Contoh ini akan menampilkan semua data dari tabel_tutorial yang mana nama penulisnya berakhiran jay:

```
root@host# mysql -u root -p password;
Enter password:*******

mysql> use TUTORIAL;
Database changed

mysql> SELECT * from tabel_tutorial
   -> WHERE penulis LIKE '%jay';

+-------+---------------+---------------+----------------+
| id  | judul   | penulis  | terbitan  |
+-------+---------------+---------------+----------------+
| 3 | JAVA Tutorial | Sanjay   | 2007-05-21 |
+-------+---------------+---------------+----------------+
1 rows in set (0.01 sec)

mysql>
```

## Menggunakan Clause LIKE pada Script PHP

Anda dapat menggunakan syntax clause WHERE....LIKE pada fungsi PHP mysql_query(). Setelah itu, Anda dapat menggunakan fungsi mysql_fetch_array() untuk mengambil semua data jika clause ini digunakan dengan perintah SELECT.

Berikut ini contoh penggunaannya untuk mengambil semua data dari tabel_tutorial yang mana nama penulisnya mengandung jay:

```
<?php
 $dbhost = 'localhost:3036';
 $dbpengguna = 'root';
 $dbpass = 'rootpassword';
 $kon = mysql_connect($dbhost, $dbpengguna, $dbpass);

 if(! $kon )
 {
  die('Tidak dapat terkoneksi: ' . mysql_error());
 }

 $sql = 'SELECT id, judul, penulis, terbitan
   FROM tabel_tutorial
   WHERE penulis LIKE "%jay%"';

 mysql_select_db('TUTORIAL');

 $nilaibalik = mysql_query( $sql, $kon );

 if(! $nilaibalik )
 {
  die('Tidak dapat mengambil data: ' . mysql_error());
 }

 while($baris = mysql_fetch_array($nilaibalik, MYSQL_ASSOC))
 {
  echo "ID :{$baris['id']} <br> ".
   "Judul: {$baris['judul']} <br> ".
   "Penulis: {$baris['penulis']} <br> ".
   "Tanggal penerbitan : {$baris['terbitan']} <br> ".
   "--------------------------------<br>";
 }

 echo "Data berhasil diambil\n";

 mysql_close($kon);
?>
```

# BAB 18

# Mengurutkan Data Hasil

Saat Anda menggunakan perintah SELECT, baris data yang ditampilkan akan diurutkan secara bebas oleh server MySQL. Untuk mengurutkan baris data hasil sesuai dengan kolom-kolom tertentu, Anda dapat menggunakan clause ORDER BY.

Berikut ini syntax SQL generik untuk perintah SELECT dengan menggunakan clause ORDER BY:

```
SELECT field1, field2,...fieldN nama_tabel1, nama_tabel2...
ORDER BY field1, [field2...] [ASC [DESC]]
```

- Anda dapat mengurutkan hasil pada field apa saja yang didaftarkan.
- Anda dapat mengurutkan hasil pada lebih dari satu field.
- Anda dapat menggunakan kata kunci ASC atau DESC untuk mengurutkan nilai naik atau nilai menurun. Nilai default adalah nilai naik (dari kecil ke besar).
- Anda dapat menggunakan clause WHERE....LIKE seperti biasa untuk menambahkan kondisi.

## Menggunakan Clause ORDER BY pada Command Prompt

Berikut ini contoh perintah SELECT dengan menggunakan clause ORDER BY untuk mengambil dan menampilkan data dari tabel tabel_tutorial:

```
root@host# mysql -u root -p password;
Enter password:*******

mysql> use TUTORIAL;
Database changed

mysql> SELECT * from tabel_tutorial ORDER BY penulis ASC

+-------+---------------+---------------+----------------+
| id  | judul   | penulis  | terbitan  |
+-------+---------------+---------------+----------------+
| 2 | Learn MySQL  | Abdul S  | 2007-05-24 |
| 1 | Learn PHP | John Poul | 2007-05-24 |
| 3 | JAVA Tutorial | Sanjay   | 2007-05-06 |
+-------+---------------+---------------+----------------+
3 rows in set (0.42 sec)

mysql>
```

Contoh tersebut mengurutkan hasil berdasarkan nama penulis secara ASC (naik dari A-Z).

## Menggunakan Clause ORDER BY pada Script PHP

Anda dapat menggunakan clause ORDER BY dalam fungsi mysql_query() pada PHP. Fungsi ini digunakan untuk mengeksekusi perintah SQL dan setelah itu fungsi mysql_fetch_array() dapat digunakan untuk mengambil semua data yang dipilih.

Berikut ini contoh untuk mengurutkan data hasil dengan urutan menurun berdasarkan nama penulis (Z-A):

```
<?php
 $dbhost = 'localhost:3036';
 $dbpengguna = 'root';
 $dbpass = 'rootpassword';
 $kon = mysql_connect($dbhost, $dbpengguna, $dbpass);

 if(! $kon )
 {
  die('Tidak dapat terkoneksi: ' . mysql_error());
 }

 $sql = 'SELECT id, judul, penulis, terbitan
```

```
  FROM tabel_tutorial
  ORDER BY penulis DESC';

mysql_select_db('TUTORIAL');

$nilaibalik = mysql_query( $sql, $kon );

if(! $nilaibalik )
{
 die('Tidak dapat mengambil data: ' . mysql_error());
}

while($baris = mysql_fetch_array($nilaibalik, MYSQL_ASSOC))
{
 echo "ID :{$baris['id']} <br> ".
  "Judul: {$baris['judul']} <br> ".
  "Penulis: {$baris['penulis']} <br> ".
  "Tanggal penerbitan : {$baris['terbitan']} <br> ".
  "--------------------------------<br>";
}

echo "Data berhasil diambil\n";

mysql_close($kon);
?>
```

# BAB 19

# Menggunakan Join

Dalam menggunakan database MySQL, Anda seringkali perlu mengambil data dari beberapa tabel dalam satu query. Untuk melakukan ini, Anda dapat menggunakan JOIN untuk menggabungkan dua tabel atau lebih menjadi satu tabel.

Anda dapat menggunakan JOIN dalam statemen SELECT, UPDATE, dan DELETE untuk menggabungkan tabel MySQL.

## Menggunakan Join pada Command Prompt

Misalnya Anda memiliki dua tabel, tabel_jumlah dan tabel_tutorial dalam database TUTORIAL sebagai berikut:

```
root@host# mysql -u root -p password;
Enter password:*******

mysql> use TUTORIAL;
Database changed

mysql> SELECT * FROM tabel_jumlah;
```

```
+-----------------+----------------+
| penulis   | jumlah   |
+-----------------+----------------+
| mahran   | 20   |
| mahnaz   | NULL  |
| Jen   | NULL  |
| Gill  | 20   |
| John Poul  | 1   |
| Sanjay   | 1   |
+-----------------+----------------+
6 rows in set (0.01 sec)

mysql> SELECT * from tabel_tutorial;

+-------+--------------+-----------------+--------------+
| id  | judul   | penulis   | terbitan |
+-------+--------------+-----------------+--------------+
| 1 | Learn PHP | John Poul  | 2007-05-24  |
| 2 | Learn MySQL  | Abdul S   | 2007-05-24  |
| 3 | JAVA Tutorial | Sanjay   | 2007-05-06  |
+-------+--------------+-----------------+--------------+
3 rows in set (0.00 sec)

mysql>
```

Sekarang Anda dapat menulis query SQL untuk menggabungkan dua tabel tersebut. Query ini akan memilih semua penulis dari tabel_tutorial dan digabungkan dengan jumlah yang sesuai dari tabel_jumlah.

```
mysql> SELECT a.id, a.penulis, b.jumlah
  -> FROM tabel_tutorial a, tabel_jumlah b
  -> WHERE a.penulis = b.penulis;

+-------------+-----------------+----------------+
| id   | penulis   | jumlah   |
+-------------+-----------------+----------------+
| 1  | John Poul  | 1   |
| 3  | Sanjay   | 1   |
+-------------+-----------------+----------------+
2 rows in set (0.01 sec)

mysql>
```

## Menggunakan Join pada Script PHP

Anda dapat menggunakan query SQL yang disebutkan di atas dalam script PHP. Anda hanya perlu menambahkan query tersebut ke dalam fungsi mysql_query().

```
<?php
 $dbhost = 'localhost:3036';
 $dbpengguna = 'root';
 $dbpass = 'rootpassword';
 $kon = mysql_connect($dbhost, $dbpengguna, $dbpass);
```

```
if(! $kon )
{
 die('Tidak dapat terkoneksi: ' . mysql_error());
}

$sql = 'SELECT a.id, a.penulis, b.jumlah
  FROM tabel_tutorial a, tabel_jumlah b
  WHERE a.penulis = b.penulis';

mysql_select_db('TUTORIAL');

$nilaibalik = mysql_query( $sql, $kon );

if(! $nilaibalik )
{
 die('Tidak dapat mengambil data: ' . mysql_error());
}

while($baris = mysql_fetch_array($nilaibalik, MYSQL_ASSOC))
{
 echo
  "Penulis: {$baris['penulis']} <br> ".
  "Jumlah: {$baris['jumlah']} <br> ".
  "ID :{$baris['id']} <br> ".
  "--------------------------------<br>";
}

echo "Data berhasil diambil\n";

mysql_close($kon);
?>
```

## LEFT JOIN MySQL

Left join berbeda dengan join biasa. LEFT JOIN MySQL memberikan pertimbangan ekstra yang ada di sebelah kiri.

Jika Anda menggunakan LEFT JOIN, Anda akan mendapatkan data yang sesuai dan juga mendapatkan data ekstra dari setiap data yang tidak sesuai di tabel sebelah kiri join.

Pada contoh berikut, penggunaan LEFT JOIN menyebabkan semua penulis ditampilkan:

```
root@host# mysql -u root -p password;
Enter password:*******

mysql> use TUTORIAL;
Database changed

mysql> SELECT a.id, a.penulis, b.jumlah
  -> FROM tabel_tutorial a LEFT JOIN tabel_jumlah b
  -> ON a.penulis = b.penulis;
```

```
+-------------+-----------------+----------------+
| id   | penulis   | jumlah   |
+-------------+-----------------+----------------+
| 1  | John Poul  | 1   |
| 2  | Abdul S   | NULL  |
| 3  | Sanjay   | 1   |
+-------------+-----------------+----------------+
3 rows in set (0.02 sec)
```

# BAB 20

# Nilai NULL

Anda dapat menggunakan perintah SQL SELECT bersama clause WHERE untuk mengambil data dari tabel MySQL, tapi saat Anda mencoba menggunakan kondisi yang membandingkan field atau kolom dengan NULL, perintah ini tidak akan berjalan dengan baik.

Untuk mengatasi hal ini, MySQL menyediakan tiga operator:

- IS NULL: operator mengembalikan nilai benar jika nilai kolom adalah NULL.
- IS NOT NULL: operator mengembalikan nilai benar jika nilai kolom bukan NULL.
- <=>; operator membandingkan nilai-nilai, yang mana akan bernilai benar bahkan untuk dua nilai NULL.

## Menggunakan Nilai NULL pada Command Prompt

Misalnya, sebuah tabel tabel_jumlah dalam database TUTORIAL mengandung dua kolom penulis dan jumlah, di mana jumlah yang berisi NULL mengindikasikan bahwa nilainya tidak diketahui:

```
root@host# mysql -u root -p password;
Enter password:*******

mysql> use TUTORIAL;
Database changed

mysql> create table tabel_jumlah
  -> (
  -> penulis varchar(40) NOT NULL,
  -> jumlah INT
  -> );
Query OK, 0 rows affected (0.05 sec)

mysql> INSERT INTO tabel_jumlah
  -> (penulis, jumlah) values ('mahran', 20);

mysql> INSERT INTO tabel_jumlah
  -> (penulis, jumlah) values ('mahnaz', NULL);

mysql> INSERT INTO tabel_jumlah
  -> (penulis, jumlah) values ('Jen', NULL);

mysql> INSERT INTO tabel_jumlah
  -> (penulis, jumlah) values ('Gill', 20);

mysql> SELECT * from tcount_tbl;

+-----------------+----------------+
| penulis   | jumlah   |
+-----------------+----------------+
| mahran   | 20   |
| mahnaz   | NULL  |
| Jen   | NULL  |
| Gill  | 20   |
+-----------------+----------------+
4 rows in set (0.00 sec)

mysql>
```

Jika Anda menggunakan operator = dan != pada nilai NULL, pencarian Anda tidak akan berjalan sebagaimana mestinya:

```
mysql> SELECT * FROM tabel_jumlah
  -> WHERE jumlah = NULL;
Empty set (0.00 sec)

mysql> SELECT * FROM tabel_jumlah
  -> WHERE jumlah != NULL;
Empty set (0.01 sec)
```

Untuk menampilkan data di mana kolom jumlah adalah NULL atau bukan NULL, Anda dapat menggunakan query berikut:

```
mysql> SELECT * FROM tabel_jumlah
  -> WHERE jumlah IS NULL;
```

```
+-----------------+----------------+
| penulis   | jumlah   |
+-----------------+----------------+
| mahnaz   | NULL  |
| Jen   | NULL  |
+-----------------+----------------+
2 rows in set (0.00 sec)

mysql> SELECT * from tabel_jumlah
  -> WHERE jumlah IS NOT NULL;

+-----------------+----------------+
| penulis   | jumlah   |
+-----------------+----------------+
| mahran   | 20   |
| Gill  | 20   |
+-----------------+----------------+
2 rows in set (0.00 sec)
```

## Menggunakan Nilai NULL pada Script PHP

Anda dapat menggunakan kondisi if...else untuk menyiapkan query berdasarkan nilai NULL.

Contoh berikut mengambil jumlah dari luar dan kemudian membandingkannya dengan nilai yang ada dalam tabel.

```
<?php
 $dbhost = 'localhost:3036';
 $dbpengguna = 'root';
 $dbpass = 'rootpassword';
 $kon = mysql_connect($dbhost, $dbpengguna, $dbpass);

 if(! $kon )
 {
  die('Tidak dapat terkoneksi: ' . mysql_error());
 }

 if( isset($jumlah ))
 {
  $sql = 'SELECT penulis, jumlah
  FROM tabel_jumlah
  WHERE jumlah = $jumlah';
 }
 else
 {
  $sql = 'SELECT penulis, jumlah
  FROM tabel_jumlah
  WHERE jumlah IS $jumlah';
 }

 mysql_select_db('TUTORIAL');

 $nilaibalik = mysql_query( $sql, $kon );

 if(! $nilaibalik )
 {
  die('Tidak dapat mengambil data: ' . mysql_error());
```

```
}

while($baris = mysql_fetch_array($nilaibalik, MYSQL_ASSOC))
{
 echo "Penulis: {$baris['penulis']} <br> ".
   "Jumlah : {$baris['jumlah']} <br> ".
   "--------------------------------<br>";
}

echo "Data berhasil diambil\n";

mysql_close($kon);
?>
```

# BAB 21

# Regexp (Regular Expression)

MySQL mendukung tipe lain untuk mencocokkan pola selain dengan menggunakan LIKE …%. Operator ini adalah REGEXP yang mencocokkan pola berdasarkan ekspresi reguler.

Berikut ini tabel yang berisi daftar pola yang digunakan dengan operator REGEXP.

| **Pola** | **Nilai yang cocok dengan pola** |
|---|---|
| ^ | Awal dari sebuah string |
| $ | Akhir dari sebuah string |
| . | Satu karakter apa saja |
| […] | Karakter apa saja yang terdaftar dalam tanda kurung siku |
| [^…] | Karakter apa saja yang tidak terdaftar dalam tanda kurung siku |
| p1\|p2\|p3 | Alternation; mencocokkan salah satu pola p1, p2, atau p3 |

| | |
|---|---|
| * | Nol instance atau lebih dari elemen yang mengawali pola |
| + | Satu instance atau lebih dari elemen yang mengawali pola |
| {n} | n instance dari elemen yang mengawali pola |
| {m,n} | m sampai n instance dari elemen yang mengawali pola |

*Tabel pola dalam operator REGEXP*

Berikut ini contoh penggunaannya. Misalkan Anda memiliki tabel dengan nama tabel_orang dan memiliki field nama:

Query untuk menemukan semua nama yang dimulai dengan 'st':

```
mysql> SELECT nama FROM tabel_orang
  WHERE nama REGEXP '^st';
```

Query untuk menemukan semua nama yang berakhiran 'ok':

```
mysql> SELECT nama FROM tabel_orang
  WHERE nama REGEXP 'ok$';
```

Query untuk menemukan semua nama yang mengandung 'mar':

```
mysql> SELECT nama FROM tabel_orang
  WHERE nama REGEXP 'mar';
```

Query untuk menemukan semua nama yang diawali huruf vokal dan diakhiri dengan 'ok':

```
mysql> SELECT nama FROM tabel_orang
  WHERE nama REGEXP '^[aiueo]|ok$';
```

# BAB 22

# Perintah Alter

Perintah MySQL ALTER sangat berguna ketika Anda ingin mengubah nama tabel, field, atau ketika Anda ingin menambahkan atau menghapus kolom yang ada dalam tabel.

Pertama-tama membuat tabel bernama tesalter.

```
root@host# mysql -u root -p password;
Enter password:*******

mysql> use TUTORIAL;
Database changed

mysql> create table tesalter
  -> (
  -> i INT,
  -> c CHAR(1)
  -> );
Query OK, 0 rows affected (0.05 sec)

mysql> SHOW COLUMNS FROM tesalter;

+-------+---------+------+-----+---------+-------+
| Field | Type  | Null | Key | Default | Extra |
+-------+---------+------+-----+---------+-------+
| i | int(11) | YES | | NULL  |  |
| c | char(1) | YES | | NULL  |  |
+-------+---------+------+-----+---------+-------+
2 rows in set (0.00 sec)
```

## Menghapus, Menambahkan, atau Mengubah Posisi Kolom

Misalkan, Anda ingin menghapus kolom i dari tabel MySQL yang ada. Untuk itu, Anda dapat menggunakan clause DROP bersama dengan perintah ALTER sebagai berikut:

```
mysql> ALTER TABLE tesalter DROP i;
```

Untuk menambahkan kolom, gunakan ADD dan spesifikasikan definisi kolom tersebut. Statemen berikut menambahkan kemali kolom i ke dalam tabel tesalter:

```
mysql> ALTER TABLE tesalter ADD i INT;
```

Setelah statemen ini dijalankan, tabel tesalter akan kembali memiliki dua kolom yaitu c dan i.

```
mysql> SHOW COLUMNS FROM tesalter;
+-------+---------+------+-----+---------+-------+
| Field | Type  | Null | Key | Default | Extra |
+-------+---------+------+-----+---------+-------+
| c | char(1) | YES | | NULL  |  |
| i | int(11) | YES | | NULL  |  |
+-------+---------+------+-----+---------+-------+
2 rows in set (0.00 sec)
```

Untuk memposisikan kolom baru yang akan ditambahkan, Anda dapat menggunakan FIRST untuk memposisikannya sebagai kolom pertama atau AFTER nama_kolom untuk memposisikannya setelah kolom tertentu.

Berikut ini contoh penggunaannya:

```
ALTER TABLE tesalter DROP i;
ALTER TABLE tesalter ADD i INT FIRST;
ALTER TABLE tesalter DROP i;
ALTER TABLE tesalter ADD i INT AFTER c;
```

FIRST dan AFTER hanya dapat digunakan dengan clause ADD. Untuk memposisikan sebuah kolom dalam tabel, Anda pertama-tama harus DROP kolom dan kemudian menggunakan ADD untuk mengatur posisinya.

## Mengubah Definisi atau Nama Kolom

Untuk mengubah definisi sebuah kolom, gunakan clause MODIFY atau CHANGE bersama dengan perintah ALTER. Contoh berikut mengubah definisi kolom c dari CHAR(1) menjadi CHAR(10):

```
mysql> ALTER TABLE tesalter MODIFY c CHAR(10);
```

Jika Anda menggunakan CHANGE, syntax-nya sedikit berbeda. Setelah kata kunci CHANGE, kolom yang ingin diubah, dan spesifikasi definisi baru termasuk nama baru. Berikut ini contoh penggunaannya:

```
mysql> ALTER TABLE tesalter CHANGE i j BIGINT;
```

Jika Anda menggunakan CHANGE untuk mengubah j dari BIGINT kembali menjadi INT tanpa mengganti nama kolom, statemen yang digunakan adalah sebagai berikut:

```
mysql> ALTER TABLE tesalter CHANGE j j INT;
```

## Efek ALTER TABLE pada Nilai Null dan Atribut Default

Saat Anda menggunakan MODIFY atau CHANGE, Anda juga dapat menspesifikasikan apakah kolom dapat bernilai NULL dan menentukan nilai default-nya.

Berikut ini contoh di mana kolom NOT NULL memiliki nilai default 100.

```
mysql> ALTER TABLE tesalter
  -> MODIFY j BIGINT NOT NULL DEFAULT 100;
```

## Mengubah Nilai Default Kolom

Anda dapat mengubah nilai default sebuah kolom dengan menggunakan perintah ALTER seperti pada contoh berikut:

```
mysql> ALTER TABLE tesalter ALTER i SET DEFAULT 1000;

mysql> SHOW COLUMNS FROM tesalter;


+-------+---------+------+-----+---------+-------+
| Field | Type  | Null | Key | Default | Extra |
+-------+---------+------+-----+---------+-------+
| c | char(1) | YES | | NULL  |  |
| i | int(11) | YES | | 1000  |  |
+-------+---------+------+-----+---------+-------+
2 rows in set (0.00 sec)
```

Anda dapat menghapus spesifikasi default dengan menggunakan clause DROP bersama dengan perintah ALTER.

```
mysql> ALTER TABLE tesalter ALTER i DROP DEFAULT;
mysql> SHOW COLUMNS FROM tesalter;

+-------+---------+------+-----+---------+-------+
| Field | Type  | Null | Key | Default | Extra |
+-------+---------+------+-----+---------+-------+
| c | char(1) | YES | | NULL  |  |
| i | int(11) | YES | | NULL  |  |
+-------+---------+------+-----+---------+-------+
2 rows in set (0.00 sec)
```

## Mengubah Tipe Tabel

Anda dapat menggunakan clause TYPE bersama dengan perintah ALTER untuk mengubah tipe tabel.

Berikut ini contoh penggunaannya:

```
mysql> ALTER TABLE tesalter TYPE = MYISAM;
mysql> SHOW TABLE STATUS LIKE 'tesalter'\G
*************************** 1. row ***************
  Name: tesalter
  Type: MyISAM
 Row_format: Fixed
  Rows: 0
 Avg_row_length: 0
 Data_length: 0
 Max_data_length: 25769803775
  Index_length: 1024
  Data_free: 0
 Auto_increment: NULL
 Create_time: 2007-06-03 08:04:36
 Update_time: 2007-06-03 08:04:36
 Check_time: NULL
 Create_options:
   Comment:
1 row in set (0.00 sec)
```

## Mengubah Nama Tabel

Untuk mengubah nama tabel, gunakan pilihan RENAME dari statemen ALTER TABLE.

Berikut ini contoh untuk mengubah nama tabel tesalter menjadi tabel_alter:

```
mysql> ALTER TABLE tesalter RENAME TO tabel_alter;
```

# BAB 23

# Penggunaan Indeks

Indeks database adalah struktur yang meningkatkan kecepatan operasi dalam sebuah tabel. Indeks dapat dibuat menggunakan satu kolom atau lebih, yang menyediakan pencarian cepat dan pengurutan data yang efisien.

## Indeks Sederhana dan Unik

Anda dapat membuat indeks unik pada tabel. Indeks unik berarti bahwa dua baris tidak dapat memiliki nilai indeks yang sama.

Berikut ini syntax untuk membuat Indeks pada tabel:

```
CREATE UNIQUE INDEX nama_indeks
ON nama_tabel ( kolom1, kolom2,...);
```

Anda dapat menggunakan satu kolom atau lebih untuk membuat indeks. Sebagai contoh, Anda dapat membuat indeks dalam tabel_tutorial dengan menggunakan penulis.

```
CREATE UNIQUE INDEX INDEKS_PENULIS
ON tabel_tutorial (penulis)
```

Untuk membuat indeks sederhana pada tabel, Anda tidak perlu menyertakan kata kunci UNIQUE saat membuat indeks. Indeks sederhana memungkinkan nilai-nilai duplikat dalam sebuah tabel.

Jika Anda ingin melakukan indeks nilai pada kolom dengan urutan menurun (dari atas ke bawah), Anda dapat menambahkan DESC setelah nama kolom.

```
mysql> CREATE UNIQUE INDEX INDEKS_PENULIS
  ON tabel_tutorial (penulis DESC)
```

## Perintah ALTER untuk Menambahkan dan Menghapus INDEKS

Terdapat empat tipe statemen untuk menambahkan indeks ke dalam tabel:

- ALTER TABLE nama_tabel ADD PRIMARY KEY (daftar_kolom): Statemen ini menambahkan PRIMARY KEY, yang berarti bahwa nilai indeks harus unik dan tidak boleh bernilai NULL.
- ALTER TABLE nama_tabel ADD UNIQUE nama_indeks (daftar_kolom): Statemen ini membuat sebuah indeks yang nilainya harus unik (terkecuali nilai NULL dapat muncul beberapa kali).
- ALTER TABLE nama_tabel ADD INDEX nama_indeks (daftar_kolom): Statemen ini menambahkan indeks biasa yang nilainya dapat muncul lebih dari satu kali.
- ALTER TABLE nama_tabel ADD FULLTEXT nama_indeks (daftar_kolom): Statemen ini membuat indeks FULLTEXT spesial yang digunakan untuk tujuan pencarian teks.

Berikut ini contoh untuk menambahkan indeks ke dalam tabel.

```
mysql> ALTER TABLE tesalter ADD INDEX (c);
```

Anda dapat menghapus indeks dengan menggunakan clause DROP bersama dengan perintah ALTER. Berikut ini contoh penggunaannya:

```
mysql> ALTER TABLE tesalter DROP INDEX (c);
```

## Perintah ALTER untuk Menambahkan dan Menghapus PRIMARY KEY

Anda dapat menambahkan primary key dengan cara yang sama. Tetapi Primary Key hanya bekerja pada kolom-kolom yang NOT NULL.

Berikut ini contoh untuk menambahkan primary key. Contoh ini akan membuat kolom menjadi NOT NULL dan kemudian menambahkannya sebagai primary key:

```
mysql> ALTER TABLE tesalter MODIFY i INT NOT NULL;
mysql> ALTER TABLE tesalter ADD PRIMARY KEY (i);
```

Anda dapat menggunakan perintah ALTER untuk menghapus primary key seperti pada contoh berikut:

```
mysql> ALTER TABLE tesalter DROP PRIMARY KEY;
```

## Menampilkan Informasi Indeks

Anda dapat menggunakan perintah SHOW INDEX untuk menampilkan daftar indeks yang diasosiasikan dengan sebuah tabel.

Berikut contoh penggunaannya:

```
mysql> SHOW INDEX FROM nama_tabel \G
  ........
```

# BAB 24

# Tabel Sementara (Temporary)

Tabel sementara dapat digunakan untuk tujuan menyimpan data sementara. Tabel sementara ini akan dihapus secara otomatis saat sesi klien berakhir atau saat script selesai dieksekusi jika Anda menggunakan PHP.

Berikut ini contoh penggunaan tabel sementara:

```
mysql> CREATE TEMPORARY TABLE Penjualan (
  -> produk VARCHAR(50) NOT NULL
  -> , total_penjualan DECIMAL(12,2) NOT NULL DEFAULT 0.00
  -> , harga DECIMAL(7,2) NOT NULL DEFAULT 0.00
  -> , unit_terjual INT UNSIGNED NOT NULL DEFAULT 0
);
Query OK, 0 rows affected (0.00 sec)

mysql> INSERT INTO Penjualan
  -> (produk, total_penjualan, harga, unit_terjual)
  -> VALUES
  -> ('cucumber', 100.25, 90, 2);

mysql> SELECT * FROM Penjualan;

+--------------+-----------------+------------+--------------+
| produk  | total_penjualan | harga | unit_terjual |
+--------------+-----------------+------------+--------------+
| cucumber | 100.25   | 90.00 | 2  |
+--------------+-----------------+------------+--------------+
1 row in set (0.00 sec)
```

## Menghapus Tabel Sementara

Jika Anda ingin menghapus tabel sementara sebelum sesi klien berakhir atau sebelum script PHP selesai dieksekusi, Anda dapat menggunakan perintah DROP TABLE.

```
mysql> CREATE TEMPORARY TABLE Penjualan (
  -> produk VARCHAR(50) NOT NULL
  -> , total_penjualan DECIMAL(12,2) NOT NULL DEFAULT 0.00
  -> , harga DECIMAL(7,2) NOT NULL DEFAULT 0.00
  -> , unit_terjual INT UNSIGNED NOT NULL DEFAULT 0
);
Query OK, 0 rows affected (0.00 sec)

mysql> INSERT INTO Penjualan
  -> (produk, total_penjualan, harga, unit_terjual)
  -> VALUES
  -> ('cucumber', 100.25, 90, 2);

mysql> SELECT * FROM Penjualan;

+-------------+-----------------+------------+-------------+
| produk  | total_penjualan | harga | unit_terjual |
+-------------+-----------------+------------+-------------+
| cucumber | 100.25   | 90.00 | 2  |
+-------------+-----------------+------------+-------------+
1 row in set (0.00 sec)

mysql> DROP TABLE Penjualan;

mysql> SELECT * FROM Penjualan;
ERROR 1146: Table 'TUTORIAL.Penjualan' doesn't exist
```

# BAB 25

# Tabel Duplikat (Clone)

Ada kalanya, Anda memerlukan tabel duplikat yang memiliki indeks yang sama, nilai default yang sama, dan seterusnya.

Anda dapat mengatasi situasi ini dengan langkah-langkah berikut:

- Menggunakan SHOW CREATE TABLE untuk mendapatkan statemen CREATE TABLE yang menspesifikasikan struktur tabel sumber, indeks-indeks, dan lain-lain.
- Modifikasi statemen untuk mengubah nama tabel duplikat dan mengeksekusi statemen. Dengan begitu, Anda akan memiliki tabel duplikat yang sama dengan tabel sumber.
- Secara opsional, jika Anda ingin konten tabel ikut disalin, gunakan juga statemen INSERT INTO ... SELECT.

Berikut ini contoh untuk membuat tabel duplikat untuk tabel_tutorial.

**Langkah 1: Mendapatkan struktur tabel lengkap.**

```
mysql> SHOW CREATE TABLE tabel_tutorial \G;
*************************** 1. row ***************************
Table: tabel_tutorial
Create Table: CREATE TABLE `tabel_tutorial ` (
 `id` int(11) NOT NULL auto_increment,
 `judul` varchar(100) NOT NULL default '',
 `penulis` varchar(40) NOT NULL default '',
 `terbitan` date default NULL,
 PRIMARY KEY (`id`),
 UNIQUE KEY `INDEKS_PENULIS` (`penulis`)
) TYPE=MyISAM
1 row in set (0.00 sec)

ERROR:
No query specified
```

**Langkah 2: Mengubah nama tabel tersebut dan membuat tabel lain.**

```
mysql> CREATE TABLE `duplikat` (
-> `id` int(11) NOT NULL auto_increment,
-> `judul` varchar(100) NOT NULL default '',
-> `penulis` varchar(40) NOT NULL default '',
-> `terbitan` date default NULL,
-> PRIMARY KEY (`id`),
-> UNIQUE KEY `INDEKS_PENULIS` (`penulis`)
-> ) TYPE=MyISAM;
Query OK, 0 rows affected (1.80 sec)
```

**Langkah 3: Jika Anda ingin menyalin data dari tabel sumber, gunakan statemen INSERT INTO ... SELECT.**

```
mysql> INSERT INTO duplikat (id, judul,
->   penulis,
->   terbitan)
-> SELECT id,judul, penulis, terbitan,
-> FROM tabel_tutorial;
Query OK, 3 rows affected (0.07 sec)
Records: 3 Duplicates: 0 Warnings: 0
```

# BAB 26

# Penggunaan Sequence

Sequence adalah sebuah set angka integer 1, 2, 3, ... yang dibuat secara berurutan sesuai dengan kebutuhan. Sequence digunakan untuk memberikan nilai unik bagi masing-masing baris dalam tabel.

## Menggunakan Kolom AUTO_INCREMENT

Cara paling sederhana untuk menggunakan Sequence adalah dengan cara mendefinisikan kolom sebagai AUTO_INCREMENT dan MySQL akan memberikan nilai unik secara otomatis pada setiap baris.

Berikut ini contoh untuk membuat tabel dan kemudian memasukkan beberapa baris data ke dalamnya.

```
mysql> CREATE TABLE serangga
   -> (
   -> id INT UNSIGNED NOT NULL AUTO_INCREMENT,
   -> PRIMARY KEY (id),
   -> nama VARCHAR(30) NOT NULL, # jenis serangga
   -> tanggal DATE NOT NULL, # tanggal penemuan
   -> asal VARCHAR(30) NOT NULL # tempat penemuan
);
Query OK, 0 rows affected (0.02 sec)
```

```
mysql> INSERT INTO insect (id,nama,tanggal,asal) VALUES
  -> (NULL,'housefly','2001-09-10','dapur'),
  -> (NULL,'millipede','2001-09-10','jalan'),
  -> (NULL,'grasshopper','2001-09-10','halaman');
Query OK, 3 rows affected (0.02 sec)
Records: 3 Duplicates: 0 Warnings: 0

mysql> SELECT * FROM serangga ORDER BY id;

+----+-------------+------------+------------+
| id | nama  | tanggal  | asal  |
+----+-------------+------------+------------+
| 1 | housefly  | 2001-09-10 | dapur |
| 2 | millipede  | 2001-09-10 | jalan |
| 3 | grasshopper | 2001-09-10 | halaman  |
+----+-------------+------------+------------+
3 rows in set (0.00 sec)
```

## Mendapatkan Nilai AUTO_INCREMENT

LAST_INSERT_ID() adalah fungsi SQL yang dapat digunakan untuk mendapatkan nilai Sequence yang terakhir digunakan. Selain itu, script PERL dan PHP juga menyediakan fungsi khusus untuk mendapatkan nilai sequence terakhir.

## Contoh pada PERL

Anda dapat menggunakan atribut mysql_insertid untuk mendapatkan nilai AUTO_INCREMENT yang dibuat dengan query.

Berikut ini contoh penggunaannya:

```
$dbh->do ("INSERT INTO serangga (nama,tangal,asal)
VALUES('moth','2001-09-14','jendela')");
my $seq = $dbh->{mysql_insertid};
```

## Contoh pada PHP

Setelah membuat query untuk membuat nilai AUTO_INCREMENT, Anda dapat memperoleh nilai terakhir dengan menggunakan fungsi mysql_insert_id():

```
mysql_query ("INSERT INTO serangga (nama,tanggal,asal)
VALUES('moth','2001-09-14','jendela')", $kon_id);
$seq = mysql_insert_id ($kon_id);
```

## Mengulang Sequence yang Ada

Untuk mengulang Sequence dari awal, pertama-tama Anda perlu menghapus kolom dan kemudian menambahkannya lagi.

Berikut ini contoh penggunaannya:

```
mysql> ALTER TABLE serangga DROP id;

mysql> ALTER TABLE serangga
  -> ADD id INT UNSIGNED NOT NULL AUTO_INCREMENT FIRST,
  -> ADD PRIMARY KEY (id);
```

## Mengulang Sequence pada Nilai Tertentu

Secara default, MySQL akan memulai sequence dari 1, tetapi Anda dapat menentukan sendiri angka lain untuk memulai sequence.

Contoh berikut memulai sequence dari 100:

```
mysql> CREATE TABLE serangga
  -> (
  -> id INT UNSIGNED NOT NULL AUTO_INCREMENT = 100,
  -> PRIMARY KEY (id),
  -> nama VARCHAR(30) NOT NULL, # jenis serangga
  -> tanggal DATE NOT NULL, # tanggal penemuan
  -> asal VARCHAR(30) NOT NULL # tempat penemuan
);
```

# BAB 27

# Mengatasi Duplikasi

Tabel-tabel atau sekelompok hasil kadang-kadang memiliki data hasil duplikasi. Anda dapat menemukan dan menghapusnya kemudian. Bab ini mendeskripsikan cara mencegah terjadinya duplikasi data dalam tabel dan cara menghapus data duplikasi yang ada.

## Mencegah Terjadinya Duplikasi Data dalam Tabel

Anda dapat menggunakan PRIMARY KEY atau Indeks UNIQUE pada field yang sesuai untuk mencegah terjadinya duplikasi data.

Berikut ini contoh tabel sederhana yang memungkinkan terjadinya duplikasi:

```
CREATE TABLE orang
(
 nama_depan CHAR(20),
 nama_belakang CHAR(20),
 jenis_kelamin CHAR(10)
);
```

Untuk mencegah terjadinya duplikasi data nama awal dan akhir pada tabel, tambahkan PRIMARY KEY pada definisinya. Anda juga perlu mendeklarasikan kolom tersebut sebagai NOT NULL.

```
CREATE TABLE orang
(
 nama_depan CHAR(20),
 nama_belakang CHAR(20),
 jenis_kelamin CHAR(10)
 PRIMARY KEY (nama_depan, nama_belakang)
);
```

Adanya indeks yang unik dalam tabel biasanya menyebabkan error jika Anda memasukkan data dalam tabel yang menduplikasi data yang ada dalam kolom yang mendefinisikan indeks. Untuk mengatasi hal ini, Anda dapat menggunakan INSERT IGNORE.

Berikut contoh penggunaannya:

```
mysql> INSERT IGNORE INTO orang (nama_depan, nama_belakang)
  -> VALUES( 'Jay', 'Thomas');
Query OK, 1 row affected (0.00 sec)

mysql> INSERT IGNORE INTO orang (nama_depan, nama_belakang)
  -> VALUES( 'Jay', 'Thomas');
Query OK, 0 rows affected (0.00 sec)
```

Gunakan REPLACE jika Anda ingin data duplikasi untuk menggantikan data yang ada.

```
mysql> REPLACE INTO orang (nama_depan, nama_belakang)
  -> VALUES( 'Ajay', 'Kumar');
Query OK, 1 row affected (0.00 sec)

mysql> REPLACE INTO orang (nama_depan, nama_belakang)
  -> VALUES( 'Ajay', 'Kumar');
Query OK, 2 rows affected (0.00 sec)
```

Cara lainnya untuk menambahkan keunikan adalah dengan menambahkan indeks UNIQUE dalam tabel.

```
CREATE TABLE orang
(
 nama_depan CHAR(20) NOT NULL,
 nama_belakang CHAR(20) NOT NULL,
 jenis_kelamin CHAR(10)
 UNIQUE (nama_depan, nama_belakang)
);
```

## Menghitung dan Mengidentifikasi Duplikasi

Berikut ini query untuk menghitung data duplikat nama_depan dan nama_belakang dalam tabel.

```
mysql> SELECT COUNT(*) as pengulangan, nama_belakang,
 nama_depan
  -> FROM orang
  -> GROUP BY nama_belakang, nama_depan
  -> HAVING pengulangan > 1;
```

Query ini akan mengembalikan daftar semua data duplikat dalam tabel orang. Secara umum, Anda dapat melakukan hal-hal berikut untuk mengidentifikasi data-data yang diduplikasi:

- Menentukan kolom mana saja yang berisi data yang mungkin diduplikasi.
- Menampilkan daftar kolom-kolom tersebut dalam daftar seleksi kolom, bersama dengan COUNT(*).
- Menampilkan daftar kolom-kolom dalam clause GROUP BY.
- Menambahkan clause HAVING yang mengeliminasi nilai unik dengan mengatur perhitungan grup menjadi lebih dari satu.

## Mengeliminasi Duplikat dari Hasil Query

Anda dapat menggunakan DISTINCT bersama dengan statemen SELECT untuk menemukan data unik yang ada dalam tabel.

```
mysql> SELECT DISTINCT nama_belakang, nama_depan
  -> FROM orang
  -> ORDER BY nama_belakang;
```

Sebagai cara alternatif pengganti DISTINCT, Anda juga dapat menggunakan clause GROUP BY.

```
mysql> SELECT nama_belakang, nama_depan
  -> FROM orang
  -> GROUP BY (nama_belakang, nama_depan);
```

## Menghapus Duplikat Menggunakan Penggantian Tabel

Berikut ini prosedur untuk menghapus semua data duplikat yang ada dalam tabel:

```
mysql> CREATE TABLE temp SELECT nama_belakang, nama_depan,
   jenis_kelamin
  -> FROM orang;
  -> GROUP BY (nama_belakang, nama_depan);

mysql> DROP TABLE orang;

mysql> ALTER TABLE temp RENAME TO orang;
```

Cara mudah untuk menghapus data duplikat dalam sebuah tabel adalah dengan menambahkan INDEX atau PRIMARY KEY.

```
mysql> ALTER IGNORE TABLE orang
  -> ADD PRIMARY KEY (nama_belakang, nama_depan);
```

# BAB 28

# Injeksi SQL dalam MySQL

Jika Anda mengambil masukan pengguna dalam halaman web dan memasukkannya ke dalam database MySQL, ada kemungkinan bahwa Anda dapat terkena masalah keamanan yang dikenal dengan Injeksi SQL. Bab ini akan mengajarkan Anda bagaimana cara mencegah hal ini dan membantu mengamankan script dan statemen MySQL.

Anda perlu memvalidasi setiap masukan dari pengguna sebelum data tersebut diproses. Hal ini dapat dilakukan dengan cara mencocokkan pola. Pada contoh berikut, nama dibatasi hanya dapat berisi karakter alpha numerik dan tanda underscore dan dengan panjang 8 sampai 20 karakter.

```
if (preg_match("/^\w{8,20}$/", $_GET['nama'], $cocok))
{
 $hasil = mysql_query("SELECT * FROM pengguna
 WHERE nama=$cocok[0]");
}
else
{
 echo "nama tidak dapat digunakan";
}
```

Untuk mendemonstrasikan masalah yang dapat terjadi, perhatikan contoh berikut:

```
// masukan seharusnya
$nama = "Qadir'; DELETE FROM pengguna;";
mysql_query("SELECT * FROM pengguna WHERE namea='{$nama}'");
```

Pemanggilan fungsi seharusnya mendapatkan data dari tabel pengguna, di mana kolom nama cocok dengan nama yang dimasukkan oleh pengguna. Pada contoh di atas, dengan menambahkan query baru ke dalam $nama, panggilan ke database dapat menyebabkan query DELETE yang diinjeksi menghapus semua data dalam tabel pengguna.

Untungnya, jika Anda menggunakan MySQL, fungsi mysql_query() tidak mengizinkan pengeksekusian beberapa query dalam satu panggilan fungsi.

## Mencegah Injeksi SQL

Anda dapat mengatasi semua escape characters dalam bahasa script seperti PERL dan PHP. Ekstensi MySQL untuk PHP menyediakan fungsi mysql_real_escape_string() untuk membebaskan karakter masukan yang khusus bagi MySQL.

```
if (get_magic_quotes_gpc())
{
 $nama = stripslashes($nama);
}

$nama = mysql_real_escape_string($nama);
mysql_query("SELECT * FROM pengguna WHERE nama='{$nama}'");
```

## Penggunaan LIKE Quandary

Untuk menggunakan LIKE quandary, mekanisme tertentu harus mengubah karakter % dan _ yang dimasukkan pengguna menjadi karakter literal. Anda dapat menggunakan addcslashes().

```
$sub = addcslashes(mysql_real_escape_string("%sesuatu_"), "%_");
// $sub == \%sesuatu\_
mysql_query("SELECT * FROM pesan WHERE subjek LIKE '{$sub}%'");
```

# BAB 29

# Mengekspor Database

Cara paling sederhana untuk mengekspor sebuah tabel data ke dalam file teks adalah dengan menggunakan statemen SELECT....INTO OUTFILE yang mengeksport sebuah hasil query langsung ke dalam sebuah file pada server host.

## Mengekspor Data dengan Statemen SELECT....INTO OUTFILE

Syntax untuk statemen ini menggabungkan SELECT dengan INTO OUTFILE namafile di bagian akhir. Format keluaran default-nya sama dengan LOAD DATA, jadi statemen berikut ini mengekspor tabel tabel_tutorial ke dalam file /temp/tutorial.txt:

```
mysql> SELECT * FROM tabel_tutorial
  -> INTO OUTFILE '/temp/tutorial.txt';
```

Statemen SELECT....INTO OUTFILE memiliki properti berikut:

- File yang dihasilkan dibuat secara langsung oleh server MySQL, jadi nama file harus mengindikasikan di mana Anda ingin menyimpan file dalam server host.

- Anda harus memiliki perizinan MySQL FILE untuk mengeksekusi statemen SELECT….INTO.
- File keluaran tidak boleh ada sebelumnya. Hal ini untuk memastikan bahwa file-file penting yang ada tidak terhapus dan digantikan dengan file yang baru.
- Anda harus mempunyai akun pada server host atau cara tertentu untuk mengakses file dari host.
- Pada UNIX, file yang dibuat dimiliki oleh server MySQL. Hal ini berarti walaupun Anda bisa membaca file tersebut, Anda tidak bisa menghapusnya.

## Mengekspor Tabel-Tabel Sebagai Data Raw

Program mysqldump digunakan untuk menyalin atau membackup tabel-tabel dan database. Program ini dapat membuat tabel keluaran baik sebagai data raw atau sebagai sekumpulan statemen INSERT yang membuat kembali data-data dalam tabel.

Untuk mengeluarkan tabel sebagai datafile, Anda harus menspesifikasikan pilihan --tab yang mengidentifikasikan direktori penulisan file.

Sebagai contoh, untuk mengeluarkan tabel tabel_tutorial dari database TUTORIAL menjadi sebuah file dalam direktori /temp, Anda dapat menggunakan perintah berikut:

```
$ mysqldump -u root -p --no-create-info \
  --tab=/temp TUTORIAL tabel_tutorial
password ******
```

## Mengekspor Konten atau Definisi Tabel dalam Format SQL

Untuk mengekspor sebuah tabel ke dalam format SQL, Anda dapat menggunakan perintah berikut:

```
$ mysqldump -u root -p TUTORIAL tabel_tutorial > dump.txt
password ******
```

Hal ini akan menghasilkan file dengan konten sebagai berikut:

```
-- MySQL dump 8.23
--
-- Host: localhost  Database: TUTORIAL
---------------------------------------------------------
-- Server version   3.23.58

--
-- Table structure for table `tabel_tutorial`
--

CREATE TABLE tabel_tutorial (
 id int(11) NOT NULL auto_increment,
 judul varchar(100) NOT NULL default '',
 penulis varchar(40) NOT NULL default '',
 terbitan date default NULL,
 PRIMARY KEY (id),
 UNIQUE KEY AUTHOR_INDEX (penulis)
) TYPE=MyISAM;

--
-- Dumping data for table `tabel_tutorial `
--

INSERT INTO tabel_tutorial
  VALUES (1,'Learn PHP','John Poul','2007-05-24');
INSERT INTO tabel_tutorial
  VALUES (2,'Learn MySQL','Abdul S','2007-05-24');
INSERT INTO tabel_tutorial
  VALUES (3,'JAVA Tutorial','Sanjay','2007-05-06');
```

Untuk mengeluarkan beberapa tabel, daftarkan nama-nama tabel setelah argumen database. Untuk mengeluarkan database, daftarkan nama database saja:

```
$ mysqldump -u root -p TUTORIAL > database_dump.txt
password ******
```

Untuk membackup semua database yang ada pada host, gunakan perintah berikut:

```
$ mysqldump -u root -p --all-databases > database_dump.txt
password ******
```

## Menyalin Tabel atau Database ke Host Lain

Jika Anda ingin menyalin tabel atau database dari satu server MySQL ke server lainnya, Anda dapat menggunakan mysqldump dengan nama database dan nama tabel.

Jalankan perintah ini pada host sumber untuk mengeluarkan database lengkap ke dalam file dump.txt:

```
$ mysqldump -u root -p database_name table_name > dump.txt
password *****
```

Anda kemudian dapat menyalin database lengkap tanpa menggunakan nama tabel tertentu.

Setelah itu, masukkan file dump.txt ke dalam host lain dan gunakan perintah berikut. Pastikan, sebelumnya Anda telah membuat nama_database pada server tujuan.

```
$ mysql -u root -p nama_database < dump.txt
password *****
```

# BAB 30

# Mengimpor Database

Ada dua cara sederhana dalam MySQL untuk memuat data ke dalam database dari file backup yang dibuat sebelumnya.

## Mengimpor Data dengan LOAD DATA

MySQL menyediakan statemen Load DATA yang bertugas sebagai pemuat data. Berikut ini contoh statemen yang membaca file dump.txt dari direktori lokal dan memuatnya ke dalam tabel tabelku pada database lokal:

```
mysql> LOAD DATA LOCAL INFILE 'dump.txt' INTO TABLE tabelku;
```

- Jika kata kunci LOCAL tidak digunakan, MySQL mencari datafile pada server host dengan pathname absolut sesuai dengan yang dituliskan dalam perintah.
- Secara default, LOAD DATA mengasumsikan bahwa datafile mengandung baris-baris berupa linefeeds (baris baru) dan nilai-nilai data dalam baris dipisahkan dengan tab.
- Untuk menspesifikasikan format file secara langsung, gunakan clause FIELDS untuk mendeskripsikan karakteristik field dalam baris, dan clause LINE untuk menspesifikasikan sequence akhir baris. Statemen berikut menspesifikasikan bahwa datafile mengandung nilai-nilai yang dipisahkan dengan tanda : dan baris-baris diakhiri dengan baris baru:

```
mysql> LOAD DATA LOCAL INFILE 'dump.txt' INTO TABLE tabelku
-> FIELDS TERMINATED BY ':'
-> LINES TERMINATED BY '\r\n';
```

- LOAD DATA mengasumsikan kolom-kolom dalam datafile mempunyai urutan yang sama dengan kolom-kolom pada tabel. Jika tidak, Anda dapat menspesifikasikan daftar untuk mengidentifikasikan kolom pada datafile dengan kolom yang sesuai pada tabel. Misalkan, tabel Anda memiliki kolom a, b, dan c, tetapi kolom-kolom pada datafile memiliki urutan b, c, dan a. Anda dapat memuat file sebagai berikut:

```
mysql> LOAD DATA LOCAL INFILE 'dump.txt'
-> INTO TABLE tabelku (b, c, a);
```

## Mengimpor Data dengan mysqlimport

MySQL juga menyediakan program mysqlimport yang bertugas sebagai pembungkus LOAD DATA, sehingga Anda dapat memuat file secara langsung dari command line.

Untuk memuat data dari dump.txt ke dalam tabelku, gunakan perintah berikut:

```
$ mysqlimport -u root -p --local nama_database dump.txt
password *****
```

Jika Anda menggunakan mysqlimport, pilihan command line menyediakan pengatur format. Berikut ini perintah mysqlimport yang berhubungan dengan dua statemen LOAD DATA sebelumnya:

```
$ mysqlimport -u root -p --local --fields-terminated-by=":" \
  --lines-terminated-by="\r\n" nama_database dump.txt
password *****
```

Statemen mysqlimport menggunakan pilihan --columns untuk menspesifikasikan urutan kolom:

```
$ mysqlimport -u root -p --local --columns=b,c,a \
  nama_database dump.txt
password *****
```

# BAB 31

# Fungsi-Fungsi MySQL

Berikut ini daftar beberapa fungsi MySQL penting yang dapat Anda gunakan.

## Clause Group By

Anda dapat menggunakan GROUP BY untuk mengelompokkan nilai-nilai dari sebuah kolom, dan dapat juga melakukan perhitungan pada kolom tersebut. Anda dapat menggunakan fungsi COUNT, SUM, AVG, dan lain-lain.

Berikut ini contoh clause GROUP BY. Misalnya, ada tabel karyawan yang memiliki data sebagai berikut:

```
mysql> SELECT * FROM karyawan;

+------+------+------------+--------------------+
| id   | nama | tanggal    | jumlah_halaman |
+------+------+------------+--------------------+
| 1    | John | 2007-01-24 | 250 |
| 2    | Ram | 2007-05-27 | 220 |
| 3    | Jack | 2007-05-06 | 170 |
| 3    | Jack | 2007-04-06 | 100 |
| 4    | Jill | 2007-04-06 | 220 |
| 5    | Zara | 2007-06-06 | 300 |
| 5    | Zara | 2007-02-06 | 350 |
+------+------+------------+--------------------+
7 rows in set (0.00 sec)
```

Misalnya berdasarkan tabel tersebut Anda ingin menghitung jumlah hari kerja setiap karyawan. Jika Anda menggunakan query berikut, hasilnya tidak akan akan sesuai:

```
mysql> SELECT COUNT(*) FROM karyawan;

+---------------------------+
| COUNT(*)   |
+---------------------------+
| 7  |
+---------------------------+
```

Untuk itu, Anda perlu menggunakan GROUP BY seperti pada contoh berikut:

```
mysql> SELECT nama, COUNT(*)
  -> FROM karyawan
  -> GROUP BY nama;

+------+----------+
| nama | COUNT(*) |
+------+----------+
| Jack | 2  |
| Jill | 1  |
| John | 1  |
| Ram | 1  |
| Zara | 2  |
+------+----------+
5 rows in set (0.04 sec)
```

## Clause IN

Anda dapat menggunakan clause IN untuk menggantikan kondisi OR.

Misalnya, ada tabel karyawan yang memiliki data sebagai berikut:

```
mysql> SELECT * FROM karyawan;

+------+------+------------+--------------------+
| id  | nama | tanggal  | jumlah_halaman |
+------+------+------------+--------------------+
| 1  | John | 2007-01-24 | 250 |
| 2  | Ram | 2007-05-27 | 220 |
| 3  | Jack | 2007-05-06 | 170 |
| 3  | Jack | 2007-04-06 | 100 |
| 4  | Jill | 2007-04-06 | 220 |
| 5  | Zara | 2007-06-06 | 300 |
| 5  | Zara | 2007-02-06 | 350 |
+------+------+------------+--------------------+
7 rows in set (0.00 sec)
```

Kemudian, Anda ingin menampilkan data dengan jumlah_halaman sama dengan 250 dan 220 dan 170. Hal ini dapat dilakukan dengan menggunakan kondisi OR berikut:

```
mysql>SELECT * FROM karyawan
  ->WHERE jumlah_halaman= 250 OR
  ->jumlah_halaman= 220 OR jumlah_halaman= 170;

+------+------+------------+--------------------+
| id  | nama | tanggal  | jumlah_halaman |
+------+------+------------+--------------------+
| 1  | John | 2007-01-24 | 250 |
| 2  | Ram | 2007-05-27 | 220 |
| 3  | Jack | 2007-05-06 | 170 |
| 4  | Jill | 2007-04-06 | 220 |
+------+------+------------+--------------------+
4 rows in set (0.02 sec)
```

Dengan menggunakan clause IN, Anda akan mendapatkan hasil yang sama juga:

```
mysql> SELECT * FROM karyawan
  -> WHERE jumlah_halaman IN ( 250, 220, 170 );

+------+------+------------+--------------------+
| id  | nama | tanggal  | jumlah_halaman |
+------+------+------------+--------------------+
| 1  | John | 2007-01-24 | 250 |
| 2  | Ram | 2007-05-27 | 220 |
| 3  | Jack | 2007-05-06 | 170 |
| 4  | Jill | 2007-04-06 | 220 |
+------+------+------------+--------------------+
4 rows in set (0.02 sec)
```

## Clause BETWEEN

Anda dapat menggunakan clause BETWEEN untuk menggantikan kombinasi kondisi "lebih dari sama dengan AND kurang dari sama dengan".

Misalnya, ada tabel karyawan yang memiliki data sebagai berikut:

```
mysql> SELECT * FROM karyawan;

+------+------+------------+--------------------+
| id  | nama | tanggal  | jumlah_halaman |
+------+------+------------+--------------------+
| 1  | John | 2007-01-24 | 250 |
| 2  | Ram | 2007-05-27 | 220 |
| 3  | Jack | 2007-05-06 | 170 |
| 3  | Jack | 2007-04-06 | 100 |
| 4  | Jill | 2007-04-06 | 220 |
| 5  | Zara | 2007-06-06 | 300 |
| 5  | Zara | 2007-02-06 | 350 |
+------+------+------------+--------------------+
7 rows in set (0.00 sec)
```

Berikut ini contoh untuk menampilkan data dengan kondisi jumlah_halaman lebih dari sama dengan 170 dan kurang dari sama dengan 300 dengan menggunakan kondisi >= dan <=:

```
mysql>SELECT * FROM karyawan
  ->WHERE jumlah_halaman >= 170 AND
  ->jumlah_halaman <= 300;

+------+------+------------+--------------------+
| id  | nama | tanggal  | jumlah_halaman |
+------+------+------------+--------------------+
| 1  | John | 2007-01-24 | 250 |
| 2  | Ram | 2007-05-27 | 220 |
| 3  | Jack | 2007-05-06 | 170 |
| 4  | Jill | 2007-04-06 | 220 |
| 5  | Zara | 2007-06-06 | 300 |
+------+------+------------+--------------------+
5 rows in set (0.03 sec)
```

Hasil yang sama dapat dicapai dengan menggunakan clause BETWEEN sebagai berikut:

```
mysql> SELECT * FROM karyawan
  -> WHERE jumlah_halaman BETWEEN 170 AND 300;

+------+------+------------+--------------------+
| id  | nama | tanggal  | jumlah_halaman |
+------+------+------------+--------------------+
| 1  | John | 2007-01-24 | 250 |
| 2  | Ram | 2007-05-27 | 220 |
| 3  | Jack | 2007-05-06 | 170 |
| 4  | Jill | 2007-04-06 | 220 |
| 5  | Zara | 2007-06-06 | 300 |
+------+------+------------+--------------------+
5 rows in set (0.03 sec)
```

## Kata Kunci UNION

Anda dapat menggunakan UNIO jika Anda ingin memilih baris-baris dari beberapa tabel atau beberapa kumpulan baris dari suatu tabel sebagai satu hasil.

Misalnya, Anda memiliki tiga tabel yang berisi pelanggan tetap dan pelanggan prospektif, serta daftar vendor tempat Anda membeli produk. Anda dapat membuat satu tabel grup email dengan menggabungkan nama dan alamat dari ketiga tabel tersebut dengan menggunakan UNION.

Misalnya, tabel-tabel itu berisi data sebagai berikut:

```
mysql> SELECT * FROM prospek;

+---------+--------+-----------------------+
| nama_d | nama_b | almt |
+---------+--------+-----------------------+
| Peter  | Jones | 482 Rush St., Apt. 402 |
| Bernice | Smith | 916 Maple Dr.   |
+---------+--------+-----------------------+

mysql> SELECT * FROM pelanggan;

+---------------+------------+---------------------+
| nama_belakang | nama_depan | alamat   |
+---------------+------------+---------------------+
| Peterson | Grace | 16055 Seminole Ave. |
| Smith   | Bernice  | 916 Maple Dr.  |
| Brown   | Walter | 8602 1st St.  |
+---------------+------------+---------------------+

mysql> SELECT * FROM vendor;

+-------------------+---------------------+
| perusahaan  | jalan |
+-------------------+---------------------+
| ReddyParts, Inc. | 38 Industrial Blvd. |
| Parts-to-go, Ltd. | 213B Commerce Park. |
+-------------------+---------------------+
```

Walaupun ketiga tabel memiliki nama kolom yang berbeda, contoh query berikut tetap dapat memilih nama dan alamat dari ketiga tabel sekaligus:

```
mysql> SELECT nama_d, nama_b, almt FROM prospek
  -> UNION
  -> SELECT nama_depan, nama_belakang, alamat FROM pelanggan
  -> UNION
  -> SELECT perusahaan, '', jalan FROM vendor;

+-------------------+----------+-----------------------+
| nama_d  | nama_b  | almt |
+-------------------+----------+-----------------------+
| Peter   | Jones  | 482 Rush St., Apt. 402 |
| Bernice  | Smith  | 916 Maple Dr.   |
| Grace   | Peterson | 16055 Seminole Ave.  |
| Walter  | Brown  | 8602 1st St.  |
| ReddyParts, Inc. |   | 38 Industrial Blvd.  |
| Parts-to-go, Ltd. |   | 213B Commerce Park.  |
+-------------------+----------+-----------------------+
```

## Fungsi COUNT

Fungsi COUNT pada MySQL sangat berguna untuk menghitung jumlah data yang dikembalikan oleh statemen SELECT.

Misalnya, ada tabel karyawan yang memiliki data sebagai berikut:

```
mysql> SELECT * FROM karyawan;

+------+------+------------+--------------------+
| id   | nama | tanggal    | jumlah_halaman |
+------+------+------------+--------------------+
| 1    | John | 2007-01-24 | 250 |
| 2    | Ram | 2007-05-27 | 220 |
| 3    | Jack | 2007-05-06 | 170 |
| 3    | Jack | 2007-04-06 | 100 |
| 4    | Jill | 2007-04-06 | 220 |
| 5    | Zara | 2007-06-06 | 300 |
| 5    | Zara | 2007-02-06 | 350 |
+------+------+------------+--------------------+
7 rows in set (0.00 sec)
```

Jika Anda ingin menghitung jumlah total baris pada tabel ini, Anda dapat menggunakan query berikut:

```
mysql>SELECT COUNT(*) FROM karyawan ;

+----------+
| COUNT(*) |
+----------+
| 7  |
+----------+
1 row in set (0.01 sec)
```

Dengan cara yang sama, jika Anda ingin menghitung jumlah data untuk Zara, Anda dapat menggunakan query berikut:

```
mysql>SELECT COUNT(*) FROM karyawan
  -> WHERE name="Zara";

+----------+
| COUNT(*) |
+----------+
| 2  |
+----------+
1 row in set (0.04 sec)
```

## Fungsi MAX

Fungsi MAX digunakan untuk menemukan data dengan nilai paling besar di antara sekumpulan data.

Misalnya, ada tabel karyawan yang memiliki data sebagai berikut:

```
mysql> SELECT * FROM karyawan;

+------+------+------------+--------------------+
| id   | nama | tanggal    | jumlah_halaman |
+------+------+------------+--------------------+
| 1    | John | 2007-01-24 | 250 |
| 2    | Ram | 2007-05-27 | 220 |
| 3    | Jack | 2007-05-06 | 170 |
| 3    | Jack | 2007-04-06 | 100 |
| 4    | Jill | 2007-04-06 | 220 |
| 5    | Zara | 2007-06-06 | 300 |
| 5    | Zara | 2007-02-06 | 350 |
+------+------+------------+--------------------+
7 rows in set (0.00 sec)
```

Misalnya, Anda ingin menemukan nilai paling besar dari jumlah_halaman, maka Anda dapat menggunakan perintah berikut:

```
mysql> SELECT MAX(jumlah_halaman)
  -> FROM karyawan;

+-------------------------+
| MAX(jumlah_halaman) |
+-------------------------+
| 350  |
+-------------------------+
1 row in set (0.00 sec)
```

Anda juga dapat menemukan semua data dengan data paling besar untuk tiap nama dengan menggunakan clause GROUP BY:

```
mysql> SELECT id, nama, MAX(jumlah_halaman)
  -> FROM karyawan GROUP BY nama;

+------+------+-------------------------+
| id   | nama | MAX(jumlah_halaman) |
+------+------+-------------------------+
| 3    | Jack | 170  |
| 4    | Jill | 220  |
| 1    | John | 250  |
| 2    | Ram | 220  |
| 5    | Zara | 350  |
+------+------+-------------------------+
5 rows in set (0.00 sec)
```

## Fungsi MIN

Fungsi MIN digunakan untuk menemukan data dengan nilai paling kecil di antara sekumpulan data.

Misalnya, ada tabel karyawan yang memiliki data sebagai berikut:

```
mysql> SELECT * FROM karyawan;

+------+------+------------+--------------------+
| id   | nama | tanggal    | jumlah_halaman |
+------+------+------------+--------------------+
| 1    | John | 2007-01-24 | 250 |
| 2    | Ram | 2007-05-27 | 220 |
| 3    | Jack | 2007-05-06 | 170 |
| 3    | Jack | 2007-04-06 | 100 |
| 4    | Jill | 2007-04-06 | 220 |
| 5    | Zara | 2007-06-06 | 300 |
| 5    | Zara | 2007-02-06 | 350 |
+------+------+------------+--------------------+
7 rows in set (0.00 sec)
```

Misalnya, Anda ingin menemukan nilai paling kecil dari jumlah_halaman, maka Anda dapat menggunakan perintah berikut:

```
mysql> SELECT MIN(jumlah_halaman)
   -> FROM karyawan;

+-------------------------+
| MIN(jumlah_halaman) |
+-------------------------+
| 100  |
+-------------------------+
1 row in set (0.00 sec)
```

Anda juga dapat menemukan semua data dengan data paling kecil untuk tiap nama dengan menggunakan clause GROUP BY:

```
mysql> SELECT id, nama, MIN(jumlah_halaman)
   -> FROM karyawan GROUP BY nama;

+------+------+-------------------------+
| id   | nama | MIN(jumlah_halaman) |
+------+------+-------------------------+
| 3    | Jack | 100  |
| 4    | Jill | 220  |
| 1    | John | 250  |
| 2    | Ram | 220  |
| 5    | Zara | 300  |
+------+------+-------------------------+
5 rows in set (0.00 sec)
```

## Fungsi AVG

Fungsi AVG digunakan untuk menemukan nilai rata-rata dari nilai-nilai yang ada dalam field.

Misalnya, ada tabel karyawan yang memiliki data sebagai berikut:

```
mysql> SELECT * FROM karyawan;

+------+------+------------+--------------------+
| id   | nama | tanggal    | jumlah_halaman |
+------+------+------------+--------------------+
| 1    | John | 2007-01-24 | 250 |
| 2    | Ram | 2007-05-27 | 220 |
| 3    | Jack | 2007-05-06 | 170 |
| 3    | Jack | 2007-04-06 | 100 |
| 4    | Jill | 2007-04-06 | 220 |
| 5    | Zara | 2007-06-06 | 300 |
| 5    | Zara | 2007-02-06 | 350 |
+------+------+------------+--------------------+
7 rows in set (0.00 sec)
```

Misalnya, Anda ingin menghitung nilai rata-rata dari field jumlah_halaman pada tabel karyawan, Anda dapat menggunakan perintah berikut:

```
mysql> SELECT AVG(jumlah_halaman)
   -> FROM karyawan;

+-------------------------+
| AVG(jumlah_halaman) |
+-------------------------+
| 230.0000 |
+-------------------------+
1 row in set (0.03 sec)
```

Anda dapat mendapatkan nilai rata-rata dari beberapa kumpulan data dengan menggunakan clause GROUP BY.

```
mysql> SELECT nama, AVG(jumlah_halaman)
   -> FROM karyawan GROUP BY nama;

+------+-------------------------+
| nama | AVG(jumlah_halaman) |
+------+-------------------------+
| Jack | 135.0000 |
| Jill | 220.0000 |
| John | 250.0000 |
| Ram | 220.0000 |
| Zara | 325.0000 |
+------+-------------------------+
5 rows in set (0.20 sec)
```

## Fungsi SUM

Fungsi SUM digunakan untuk mendapatkan jumlah total nilai-nilai dalam suatu field.

Misalnya, ada tabel karyawan yang memiliki data sebagai berikut:

```
mysql> SELECT * FROM karyawan;

+------+------+------------+--------------------+
| id   | nama | tanggal    | jumlah_halaman |
+------+------+------------+--------------------+
| 1    | John | 2007-01-24 | 250 |
| 2    | Ram | 2007-05-27 | 220 |
| 3    | Jack | 2007-05-06 | 170 |
| 3    | Jack | 2007-04-06 | 100 |
| 4    | Jill | 2007-04-06 | 220 |
| 5    | Zara | 2007-06-06 | 300 |
| 5    | Zara | 2007-02-06 | 350 |
+------+------+------------+--------------------+
7 rows in set (0.00 sec)
```

Misalnya, Anda ingin menghitung jumlah total dari jumlah_halaman pada tabel karyawan, Anda dapat menggunakan perintah berikut:

```
mysql> SELECT SUM(jumlah_halaman)
  -> FROM karyawan;

+-------------------------+
| SUM(jumlah_halaman) |
+-------------------------+
| 1610 |
+-------------------------+
1 row in set (0.00 sec)
```

Anda dapat mendapatkan jumlah total nilai dari beberapa kumpulan data dengan menggunakan clause GROUP BY.

```
mysql> SELECT nama, SUM(jumlah_halaman)
  -> FROM karyawan GROUP BY nama;

+------+-------------------------+
| nama | SUM(jumlah_halaman) |
+------+-------------------------+
| Jack | 270  |
| Jill | 220  |
| John | 250  |
| Ram | 220  |
| Zara | 650  |
+------+-------------------------+
5 rows in set (0.17 sec)
```

## Fungsi SQRT

Fungsi SQRT digunakan untuk menemukan akar pangkat dua dari sebuah bilangan.

Anda dapat menggunakan statemen SELECT sebagai berikut:

```
mysql> select SQRT(16);

+----------+
| SQRT(16) |
+----------+
| 4.000000 |
+----------+
1 row in set (0.00 sec)
```

Misalnya, ada tabel karyawan yang memiliki data sebagai berikut:

```
mysql> SELECT * FROM karyawan;

+------+------+------------+--------------------+
| id   | nama | tanggal    | jumlah_halaman |
+------+------+------------+--------------------+
| 1    | John | 2007-01-24 | 250 |
| 2    | Ram | 2007-05-27 | 220 |
| 3    | Jack | 2007-05-06 | 170 |
| 3    | Jack | 2007-04-06 | 100 |
| 4    | Jill | 2007-04-06 | 220 |
| 5    | Zara | 2007-06-06 | 300 |
| 5    | Zara | 2007-02-06 | 350 |
+------+------+------------+--------------------+
7 rows in set (0.00 sec)
```

Anda juga dapat menggunakan fungsi SQRT untuk mendapatkan akar pangkat dua dari beberapa data. Contoh berikut menghitung akar pangkat dua dari semua jumlah_halaman pada tabel karyawan:

```
mysql> SELECT nama, SQRT(jumlah_halaman)
  -> FROM karyawan;


+------+-------------------------+
| nama | SQRT(jumlah_halaman) |
+------+-------------------------+
| John | 15.811388 |
| Ram | 14.832397 |
| Jack | 13.038405 |
| Jack | 10.000000 |
| Jill | 14.832397 |
| Zara | 17.320508 |
| Zara | 18.708287 |
+------+-------------------------+
7 rows in set (0.00 sec)
```

## Fungsi RAND

Fungsi RAND dapat digunakan untuk mendapatkan nilai acak di antara 0 sampai 1.

```
mysql> SELECT RAND( ), RAND( ), RAND( );
```

```
+------------------+-----------------+------------------+
| RAND( )   | RAND( )   | RAND( )   |
+------------------+-----------------+------------------+
| 0.45464584925645 | 0.1824410643265 | 0.54826780459682 |
+------------------+-----------------+------------------+
1 row in set (0.00 sec)
```

Misalnya, ada tabel karyawan yang memiliki data sebagai berikut:

```
mysql> SELECT * FROM karyawan;

+------+------+------------+--------------------+
| id  | nama | tanggal  | jumlah_halaman |
+------+------+------------+--------------------+
| 1  | John | 2007-01-24 | 250 |
| 2  | Ram | 2007-05-27 | 220 |
| 3  | Jack | 2007-05-06 | 170 |
| 3  | Jack | 2007-04-06 | 100 |
| 4  | Jill | 2007-04-06 | 220 |
| 5  | Zara | 2007-06-06 | 300 |
| 5  | Zara | 2007-02-06 | 350 |
+------+------+------------+--------------------+
7 rows in set (0.00 sec)
```

Anda dapat menggunakan ORDER BY RAND() untuk mengacak sekumpulan baris atau nilai seperti pada contoh berikut:

```
mysql> SELECT * FROM karyawan ORDER BY RAND();

+------+------+------------+--------------------+
| id  | nama | tanggal  | jumlah_halaman |
+------+------+------------+--------------------+
| 5  | Zara | 2007-06-06 | 300 |
| 3  | Jack | 2007-04-06 | 100 |
| 3  | Jack | 2007-05-06 | 170 |
| 2  | Ram | 2007-05-27 | 220 |
| 4  | Jill | 2007-04-06 | 220 |
| 5  | Zara | 2007-02-06 | 350 |
| 1  | John | 2007-01-24 | 250 |
+------+------+------------+--------------------+
7 rows in set (0.01 sec)

mysql> SELECT * FROM karyawan ORDER BY RAND();

+------+------+------------+--------------------+
| id  | nama | tanggal  | jumlah_halaman |
+------+------+------------+--------------------+
| 5  | Zara | 2007-02-06 | 350 |
| 2  | Ram | 2007-05-27 | 220 |
| 3  | Jack | 2007-04-06 | 100 |
| 1  | John | 2007-01-24 | 250 |
| 4  | Jill | 2007-04-06 | 220 |
| 3  | Jack | 2007-05-06 | 170 |
| 5  | Zara | 2007-06-06 | 300 |
+------+------+------------+--------------------+
7 rows in set (0.00 sec)
```

## Fungsi CONCAT

Fungsi CONCAT digunakan untuk menyatukan dua string atau lebih menjadi satu string.

```
mysql> SELECT CONCAT('PERTAMA ', 'KEDUA');

+-----------------------------+
| CONCAT('PERTAMA ', 'KEDUA') |
+-----------------------------+
| PERTAMA KEDUA |
+-----------------------------+
1 row in set (0.00 sec)
```

Misalnya, ada tabel karyawan yang memiliki data sebagai berikut:

```
mysql> SELECT * FROM karyawan;

+------+------+------------+--------------------+
| id  | nama | tanggal  | jumlah_halaman |
+------+------+------------+--------------------+
| 1  | John | 2007-01-24 | 250 |
| 2  | Ram | 2007-05-27 | 220 |
| 3  | Jack | 2007-05-06 | 170 |
| 3  | Jack | 2007-04-06 | 100 |
| 4  | Jill | 2007-04-06 | 220 |
| 5  | Zara | 2007-06-06 | 300 |
| 5  | Zara | 2007-02-06 | 350 |
+------+------+------------+--------------------+
7 rows in set (0.00 sec)
```

Misalnya, Anda ingin menyatukan semua id, nama, dan tanggal pada tabel karyawan. Anda dapat melakukannya seperti pada contoh berikut:

```
mysql> SELECT CONCAT(id, nama, tanggal)
  -> FROM karyawan;

+-----------------------------+
| CONCAT(id, nama, tanggal)  |
+-----------------------------+
| 1John2007-01-24   |
| 2Ram2007-05-27   |
| 3Jack2007-05-06   |
| 3Jack2007-04-06   |
| 4Jill2007-04-06   |
| 5Zara2007-06-06   |
| 5Zara2007-02-06   |
+-----------------------------+
7 rows in set (0.00 sec)
```

## Fungsi DATE dan TIME

Berikut ini beberapa fungsi DATE dan TIME penting dalam MySQL.

### ADDDATE(tanggal, INTERVAL unit ekspr), ADDDATE(ekspr, hari)

Berikut ini adalah contoh penggunaannya:

```
mysql> SELECT DATE_ADD('1998-01-02', INTERVAL 31 DAY);

+-------------------------------------------------------+
| DATE_ADD('1998-01-02', INTERVAL 31 DAY)   |
+-------------------------------------------------------+
| 1998-02-02   |
+-------------------------------------------------------+
1 row in set (0.00 sec)

mysql> SELECT ADDDATE('1998-01-02', INTERVAL 31 DAY);

+-------------------------------------------------------+
| ADDDATE('1998-01-02', INTERVAL 31 DAY)   |
+-------------------------------------------------------+
| 1998-02-02   |
+-------------------------------------------------------+
1 row in set (0.00 sec)

mysql> SELECT ADDDATE('1998-01-02', 31);

+-------------------------------------------------------+
| DATE_ADD('1998-01-02', INTERVAL 31 DAY)   |
+-------------------------------------------------------+
| 1998-02-02   |
+-------------------------------------------------------+
1 row in set (0.00 sec)
```

### ADDTIME(ekspr1, ekspr2)

Fungsi ini menambahkan ekspr2 pada ekspr1 dan mengembalikan hasilnya. ekspr1 adalah ekspresi time atau datetime dan ekspr2 adalah ekspresi time.

```
mysql> SELECT ADDTIME('1997-12-31 23:59:59.999999','1
 1:1:1.000002');

+-----------------------------------------------------------+
| DATE_ADD('1997-12-31 23:59:59.999999','1 1:1:1.000002') |
+-----------------------------------------------------------+
| 1998-01-02 01:01:01.000001  |
+-----------------------------------------------------------+
1 row in set (0.00 sec)
```

## CURDATE()

Mengembalikan tanggal saat ini dengan format 'YYYY-MM-DD' atau YYYYMMDD.

```
mysql> SELECT CURDATE();

+------------------------------------------------------+
| CURDATE()  |
+------------------------------------------------------+
| 1997-12-15   |
+------------------------------------------------------+
1 row in set (0.00 sec)

mysql> SELECT CURDATE() + 0;

+------------------------------------------------------+
| CURDATE() + 0  |
+------------------------------------------------------+
| 19971215  |
+------------------------------------------------------+
1 row in set (0.00 sec)
```

## CURTIME()

Mengembalikan waktu saat ini dengan format 'HH:MM:SS' atau HHMMSS.

```
mysql> SELECT CURTIME();

+------------------------------------------------------+
| CURTIME()  |
+------------------------------------------------------+
| 23:50:26  |
+------------------------------------------------------+
1 row in set (0.00 sec)

mysql> SELECT CURTIME() + 0;

+------------------------------------------------------+
| CURTIME() + 0  |
+------------------------------------------------------+
| 235026   |
+------------------------------------------------------+
1 row in set (0.00 sec)
```

### DATE(ekspr)

Mengekstrak bagian tanggal dari ekspresi time atau datetime.

```
mysql> SELECT DATE('2003-12-31 01:02:03');

+-----------------------------------------------------------+
| DATE('2003-12-31 01:02:03')  |
+-----------------------------------------------------------+
| 2003-12-31   |
+-----------------------------------------------------------+
1 row in set (0.00 sec)
```

### DAYNAME(tanggal)

Mengembalikan nama hari pada tanggal tersebut.

```
mysql> SELECT DAYNAME('1998-02-05');

+-----------------------------------------------------------+
| DAYNAME('1998-02-05')   |
+-----------------------------------------------------------+
| Thursday  |
+-----------------------------------------------------------+
1 row in set (0.00 sec)
```

### DAYOFWEEK(tanggal)

Mengembalikan indeks hari 1 sampai 7 dalam satu minggu.

```
mysql> SELECT DAYOFWEEK('1998-02-03');

+-----------------------------------------------------------+
|DAYOFWEEK('1998-02-03')    |
+-----------------------------------------------------------+
| 3    |
+-----------------------------------------------------------+
1 row in set (0.00 sec)
```

### DAYOFYEAR(tanggal)

Mengembalikan indeks hari 1 sampai 366 dalam satu tahun.

```
mysql> SELECT DAYOFYEAR('1998-02-03');

+-----------------------------------------------------------+
| DAYOFYEAR('1998-02-03')    |
+-----------------------------------------------------------+
| 34   |
+-----------------------------------------------------------+
1 row in set (0.00 sec)
```

## MONTHNAME(tanggal)

Mengembalikan nama bulan.

```
mysql> SELECT MONTHNAME('1998-02-05');

+-----------------------------------------------------------+
| MONTHNAME('1998-02-05')    |
+-----------------------------------------------------------+
| February  |
+-----------------------------------------------------------+
1 row in set (0.00 sec)
```

## NOW()

Mengembalikan tanggal dan waktu sekarang dengan format 'YYYY-MM-DD HH:MM:SS' atau YYYYMMDDHHMMSS.

```
mysql> SELECT NOW();

+-----------------------------------------------------------+
| NOW()    |
+-----------------------------------------------------------+
| 1997-12-15 23:50:26    |
+-----------------------------------------------------------+
1 row in set (0.00 sec)
```

## SEC_TO_TIME(detik)

Mengonversi argumen detik menjadi format waktu jam, menit, detik.

```
mysql> SELECT SEC_TO_TIME(2378);

+-----------------------------------------------------------+
| SEC_TO_TIME(2378)  |
+-----------------------------------------------------------+
| 00:39:38  |
+-----------------------------------------------------------+
1 row in set (0.00 sec)
```

## TIME(ekspr)

Mengekstrak bagian waktu dari ekspresi time atau datetime.

```
mysql> SELECT TIME('2003-12-31 01:02:03');

+-----------------------------------------------------------+
| TIME('2003-12-31 01:02:03')  |
+-----------------------------------------------------------+
| 01:02:03  |
+-----------------------------------------------------------+
1 row in set (0.00 sec)
```

### TIMESTAMP(ekspr), TIMESTAMP(ekspr1, ekspr2)

Dengan satu argumen, fungsi ini mengembalikan ekspresi sebagai nilai datetime. Dengan dua argumen, fungsi menambahkan ekspr2 pada ekspr1 dan mengembalikannya sebagai nilai datetime.

```
mysql> SELECT TIMESTAMP('2003-12-31');

+---------------------------------------------------------+
| TIMESTAMP('2003-12-31')    |
+---------------------------------------------------------+
| 2003-12-31 00:00:00    |
+---------------------------------------------------------+
1 row in set (0.00 sec)
```

### TIME_TO_SEC(waktu)

Mengembalikan argumen waktu yang dikonversi menjadi detik.

```
mysql> SELECT TIME_TO_SEC('22:23:00');

+---------------------------------------------------------+
| TIME_TO_SEC('22:23:00')    |
+---------------------------------------------------------+
| 80580    |
+---------------------------------------------------------+
1 row in set (0.00 sec)
```

### WEEKOFYEAR(tanggal)

Mengembalikan urutan minggu 1 sampai 53 dalam satu tahun.

```
mysql> SELECT WEEKOFYEAR('1998-02-20');

+---------------------------------------------------------+
| WEEKOFYEAR('1998-02-20')   |
+---------------------------------------------------------+
| 8    |
+---------------------------------------------------------+
1 row in set (0.00 sec)
```

## Fungsi-fungsi Numerik

Berikut ini beberapa fungsi numerik penting dalam MySQL.

### ABS(X)

Mengembalikan nilai absolut dari X.

```
mysql> SELECT ABS(2);

+-----------------------------------------------------------+
| ABS(2)   |
+-----------------------------------------------------------+
| 2    |
+-----------------------------------------------------------+
1 row in set (0.00 sec)

mysql> SELECT ABS(-2);

+-----------------------------------------------------------+
| ABS(2)   |
+-----------------------------------------------------------+
| 2    |
+-----------------------------------------------------------+
1 row in set (0.00 sec)
```

## CEIL(X), CEILING(X)

Mengembalikan nilai integer terkecil yang tidak lebih kecil dari X.

```
mysql> SELECT CEILING(3.46);

+-----------------------------------------------------------+
| CEILING(3.46)  |
+-----------------------------------------------------------+
| 4    |
+-----------------------------------------------------------+
1 row in set (0.00 sec)

mysql> SELECT CEIL(-6.43);

+-----------------------------------------------------------+
| CEIL(-6.43)   |
+-----------------------------------------------------------+
| -6   |
+-----------------------------------------------------------+
1 row in set (0.00 sec)
```

## COS(X)

Mengembalikan nilai kosinus dari X.

```
mysql>SELECT COS(90);

+-----------------------------------------------------------+
| COS(90)   |
+-----------------------------------------------------------+
| -0.44807361612917  |
+-----------------------------------------------------------+
1 row in set (0.00 sec)
```

## COT(X)

Mengembalikan nilai kotangen dari X.

```
mysql>SELECT COT(1);

+-------------------------------------------------------+
| COT(1)   |
+-------------------------------------------------------+
| 0.64209261593433  |
+-------------------------------------------------------+
1 row in set (0.00 sec)
```

## DEGREES(X)

Mengembalikan nilai X yang dikonversi dari radian menjadi derajat.

```
mysql>SELECT DEGREES(PI());

+-------------------------------------------------------+
| DEGREES(PI())  |
+-------------------------------------------------------+
| 180.000000   |
+-------------------------------------------------------+
1 row in set (0.00 sec)
```

## FLOOR(X)

Mengembalikan nilai integer terbesar yang tidak lebih dari nilai X.

```
mysql>SELECT FLOOR(7.55);

+-------------------------------------------------------+
| FLOOR(7.55)   |
+-------------------------------------------------------+
| 7    |
+-------------------------------------------------------+
1 row in set (0.00 sec)
```

## FORMAT(X, D)

Fungsi ini digunakan untuk memformat nilai X dengan format: ###,###,###.## dibulatkan menjadi D angka belakang koma.

```
mysql>SELECT FORMAT(423423234.65434453,2);

+-------------------------------------------------------+
| FORMAT(423423234.65434453,2)   |
+-------------------------------------------------------+
| 423,423,234.65    |
+-------------------------------------------------------+
1 row in set (0.00 sec)
```

## GREATEST(n1, n2, n3, …….)

Fungsi ini mengembalikan nilai paling besar dari sekumpulan parameter masukan (n1, n2, n3, dst…).

```
mysql>SELECT GREATEST(3,5,1,8,33,99,34,55,67,43);

+-----------------------------------------------------------+
| GREATEST(3,5,1,8,33,99,34,55,67,43)  |
+-----------------------------------------------------------+
| 99   |
+-----------------------------------------------------------+
1 row in set (0.00 sec)
```

## LEAST(n1, n2, n3, …….)

Fungsi ini merupakan kebalikan dari fungsi GREATEST() dan mengembalikan nilai paling kecil dari daftar nilai (n1, n2, n3, dst...).

```
mysql>SELECT LEAST(3,5,1,8,33,99,34,55,67,43);

+-----------------------------------------------------------+
| LEAST(3,5,1,8,33,99,34,55,67,43) |
+-----------------------------------------------------------+
| 1    |
+-----------------------------------------------------------+
1 row in set (0.00 sec)
```

## LOG(X), LOG(B, X)

Dengan satu argumen, fungsi ini mengembalikan nilai logaritma natural dari X. Dengan dua argumen, fungsi ini mengembalikan nilai logaritma X untuk nilai dasar B.

```
mysql>SELECT LOG(45);

+-----------------------------------------------------------+
| LOG(45)   |
+-----------------------------------------------------------+
| 3.806662  |
+-----------------------------------------------------------+
1 row in set (0.00 sec)


mysql>SELECT LOG(2,65536);

+-----------------------------------------------------------+
| LOG(2,65536)  |
+-----------------------------------------------------------+
| 16.000000  |
+-----------------------------------------------------------+
1 row in set (0.00 sec)
```

### LOG10(X)

Mengembalikan logaritma dasar 10 dari X.

```
mysql>SELECT LOG10(100);

+-----------------------------------------------------------+
| LOG10(100)   |
+-----------------------------------------------------------+
| 2.000000  |
+-----------------------------------------------------------+
1 row in set (0.00 sec)
```

### MOD(N,M)

Mengembalikan sisa hasil bagi dari N dibagi M.

```
mysql>SELECT MOD(29,3);

+-----------------------------------------------------------+
| MOD(29,3)  |
+-----------------------------------------------------------+
| 2    |
+-----------------------------------------------------------+
1 row in set (0.00 sec)
```

### PI()

Mengembalikan nilai pi.

```
mysql>SELECT PI();

+-----------------------------------------------------------+
| PI()    |
+-----------------------------------------------------------+
| 3.141593  |
+-----------------------------------------------------------+
1 row in set (0.00 sec)
```

### POW(X, Y), POWER(X, Y)

Mengembalikan nilai X pangkat Y.

```
mysql> SELECT POWER(3,3);

+-----------------------------------------------------------+
| POWER(3,3)   |
+-----------------------------------------------------------+
| 27   |
+-----------------------------------------------------------+
1 row in set (0.00 sec)
```

## RADIANS(X)

Mengembalikan nilai X yang dikonversi dari derajat menjadi radian.

```
mysql>SELECT RADIANS(90);

+-----------------------------------------------------------+
| RADIANS(90)   |
+-----------------------------------------------------------+
|1.570796   |
+-----------------------------------------------------------+
1 row in set (0.00 sec)
```

## ROUND(X), ROUND(X, D)

Mengembalikan X yang dibulatkan menjadi nilai integer terdekat. Jika menggunakan argumen kedua, fungsi akan mengembalikan X yang dibulatkan dengan D angka belakang koma.

```
mysql>SELECT ROUND(5.693893);

+-----------------------------------------------------------+
| ROUND(5.693893)    |
+-----------------------------------------------------------+
| 6    |
+-----------------------------------------------------------+
1 row in set (0.00 sec)

mysql>SELECT ROUND(5.693893,2);

+-----------------------------------------------------------+
| ROUND(5.693893,2)  |
+-----------------------------------------------------------+
| 5.69    |
+-----------------------------------------------------------+
1 row in set (0.00 sec)
```

## SIN(X)

Mengembalikan nilai sinus dari X.

```
mysql>SELECT SIN(90);

+-----------------------------------------------------------+
| SIN(90)   |
+-----------------------------------------------------------+
| 0.893997  |
+-----------------------------------------------------------+
1 row in set (0.00 sec)
```

## SQRT(X)

Mengembalikan akar pangkat dua dari X.

```
mysql>SELECT SQRT(49);

+-----------------------------------------------------------+
| SQRT(49)  |
+-----------------------------------------------------------+
| 7    |
+-----------------------------------------------------------+
1 row in set (0.00 sec)
```

## TAN(X)

Mengembalikan nilai tangen dari X.

```
mysql>SELECT TAN(45);

+-----------------------------------------------------------+
| TAN(45)   |
+-----------------------------------------------------------+
| 1.619775  |
+-----------------------------------------------------------+
1 row in set (0.00 sec)
```

## TRUNCATE(X, D)

Fungsi ini digunakan untuk mengembalikan nilai X yang dipotong menjadi D angka belakang koma.

```
mysql>SELECT TRUNCATE(7.536432,2);

+-----------------------------------------------------------+
| TRUNCATE(7.536432,2)   |
+-----------------------------------------------------------+
| 7.53    |
+-----------------------------------------------------------+
1 row in set (0.00 sec)
```

## Fungsi-Fungsi String

Berikut ini beberapa fungsi String penting dalam MySQL.

### ASCII(str)

Mengembalikan nilai numerik dari karakter paling kiri pada string str.

```
mysql> SELECT ASCII('2');

+-----------------------------------------------------------+
| ASCII('2')   |
+-----------------------------------------------------------+
| 50   |
+-----------------------------------------------------------+
1 row in set (0.00 sec)


mysql> SELECT ASCII('dx');

+-----------------------------------------------------------+
| ASCII('dx')   |
+-----------------------------------------------------------+
| 100   |
+-----------------------------------------------------------+
1 row in set (0.00 sec)
```

### BIN(N)

Mengembalikan representasi string nilai biner dari N, di mana N merupakan angka BIGINT.

```
mysql> SELECT BIN(12);

+-----------------------------------------------------------+
| BIN(12)   |
+-----------------------------------------------------------+
| 1100    |
+-----------------------------------------------------------+
1 row in set (0.00 sec)
```

### BIT_LENGTH(str)

Mengembalikan panjang bit dari string str.

```
mysql> SELECT BIT_LENGTH('teks');

+-----------------------------------------------------------+
| BIT_LENGTH('teks')    |
+-----------------------------------------------------------+
| 32   |
+-----------------------------------------------------------+
1 row in set (0.00 sec)
```

## CHAR(N, ….)

Fungsi ini menginterpretasikan setiap argumen sebagai integer dan mengembalikan string yang terdiri dari karakter yang sesuai dengan kode nilai integer yang ada.

```
mysql> SELECT CHAR(77,121,83,81,'76');

+-----------------------------------------------------------+
| CHAR(77,121,83,81,'76')    |
+-----------------------------------------------------------+
| MySQL    |
+-----------------------------------------------------------+
1 row in set (0.00 sec)
```

## CHAR_LENGTH(str)

Mengembalikan panjang string str, dihitung dari jumlah karakternya.

```
mysql> SELECT CHAR_LENGTH("teks");

+-----------------------------------------------------------+
| CHAR_LENGTH("teks")    |
+-----------------------------------------------------------+
| 4    |
+-----------------------------------------------------------+
1 row in set (0.00 sec)
```

## CONCAT(str1, str2, ….)

Mengembalikan string yang berupa gabungan dari argumen-argumen yang ada.

```
mysql> SELECT CONCAT('My', 'S', 'QL');

+-----------------------------------------------------------+
| CONCAT('My', 'S', 'QL')    |
+-----------------------------------------------------------+
| MySQL    |
+-----------------------------------------------------------+
1 row in set (0.00 sec)
```

## CONCAT_WS(pemisah, str1, str2, ….)

Fungsi ini menggabungkan string pada argumen-argumen yang ada dengan menambahkan tanda pemisah yang didefinisikan pada argumen pertama.

```
mysql> SELECT CONCAT_WS(',','Nama depan','Nama Belakang' );

+-------------------------------------------------------+
| CONCAT_WS(',','Nama depan','Nama Belakang' )  |
+-------------------------------------------------------+
| Nama depan, Nama Belakang   |
+-------------------------------------------------------+
1 row in set (0.00 sec)
```

### ELT(N, str1, str2, str3, ….)

Mengembalikan str1 jika N = 1, str2 jika N = 2, dan seterusnya.

```
mysql> SELECT ELT(1, 'ej', 'Heja', 'hej', 'foo');

+-------------------------------------------------------+
| ELT(1, 'ej', 'Heja', 'hej', 'foo')  |
+-------------------------------------------------------+
| ej   |
+-------------------------------------------------------+
1 row in set (0.00 sec)
```

### FIELD(str, str1, str2, str3, ….)

Mengembalikan indeks (posisi dimulai dari 1) dari str pada daftar str1, str2, str3, ….

```
mysql> SELECT FIELD('ej', 'Hej', 'ej', 'Heja', 'hej', 'foo');

+-------------------------------------------------------+
| FIELD('ej', 'Hej', 'ej', 'Heja', 'hej', 'foo')   |
+-------------------------------------------------------+
| 2    |
+-------------------------------------------------------+
1 row in set (0.00 sec)
```

### FIELD_IN_SET(str, daftarstr)

Mengembalikan nilai dengan rentang 1 sampai N jika string str ada dalam daftar string daftarstr.

```
mysql> SELECT FIND_IN_SET('b','a,b,c,d');

+-------------------------------------------------------+
| SELECT FIND_IN_SET('b','a,b,c,d') |
+-------------------------------------------------------+
| 2    |
+-------------------------------------------------------+
1 row in set (0.00 sec)
```

## HEX(A_atau_S)

Jika A_atau_S adalah angka, fungsi akan mengembalikan representasi string nilai hexadesimal dari N. Jika A_atau_S adalah string, fungsi akan mengembalikan representasi string nilai hexadesimal dari A_atau_S di mana setiap karakter dikonversi ke dalam dua digit hexadesimal.

```
mysql> SELECT HEX(255);

+-----------------------------------------------------------+
| HEX(255)  |
+-----------------------------------------------------------+
| FF   |
+-----------------------------------------------------------+
1 row in set (0.00 sec)


mysql> SELECT 0x616263;

+-----------------------------------------------------------+
| 0x616263  |
+-----------------------------------------------------------+
| abc   |
+-----------------------------------------------------------+
1 row in set (0.00 sec)
```

## INSERT(str, pos, panj, strbaru)

Fungsi ini mengembalikan str, dengan string yang dimulai pada posisi pos digantikan dengan string strbaru dengan panjang panj.

```
mysql> SELECT INSERT('Quadratic', 3, 4, 'What');

+-----------------------------------------------------------+
| INSERT('Quadratic', 3, 4, 'What') |
+-----------------------------------------------------------+
| QuWhattic  |
+-----------------------------------------------------------+
1 row in set (0.00 sec)
```

## INSTR(str, substr)

Mengembalikan indeks posisi di mana terdapat substr dalam str.

```
mysql> SELECT INSTR('foobarbar', 'bar');

+-----------------------------------------------------------+
| INSTR('foobarbar', 'bar')   |
+-----------------------------------------------------------+
| 4    |
+-----------------------------------------------------------+
1 row in set (0.00 sec)
```

## LEFT(str, panj)

Mengembalikan karakter dengan jumlah panj dari sebelah kiri str.

```
mysql> SELECT LEFT('foobarbar', 5);

+----------------------------------------------------------+
| LEFT('foobarbar', 5)   |
+----------------------------------------------------------+
| fooba    |
+----------------------------------------------------------+
1 row in set (0.00 sec)
```

## LENGTH(str)

Mengembalikan panjang string str, dihitung berdasarkan bit.

```
mysql> SELECT LENGTH('teks');

+----------------------------------------------------------+
| LENGTH('teks')    |
+----------------------------------------------------------+
| 4    |
+----------------------------------------------------------+
1 row in set (0.00 sec)
```

## LOAD_FILE(nama_file)

Fungsi ini membaca dan mengembalikan konten file dalam bentuk string. Untuk menggunakan fungsi ini, file harus ada pada server host, Anda harus menspesifikasian full pathname, dan Anda harus memiliki izin FILE.

```
mysql> UPDATE tabel_tes
  -> SET blob_col=LOAD_FILE('/tmp/gambar')
  -> WHERE id=1;

.....................................................
```

## LOCATE(substr, str), LOCATE(substr, str, pos)

Syntax pertama mengembalikan posisi di mana substr muncul di dalam str. Syntax kedua mengembalikan posisi di mana substr muncul di dalam str, dimulai dari posisi pos.

```
mysql> SELECT LOCATE('bar', 'foobarbar');

+-----------------------------------------------------------+
| LOCATE('bar', 'foobarbar')  |
+-----------------------------------------------------------+
| 4    |
+-----------------------------------------------------------+
1 row in set (0.00 sec)
```

## LOWER(str)

Mengembalikan str dengan semua karakter diubah menjadi huruf kecil.

```
mysql> SELECT LOWER('QUADRATICALLY');

+-----------------------------------------------------------+
| LOWER('QUADRATICALLY')    |
+-----------------------------------------------------------+
| quadratically  |
+-----------------------------------------------------------+
1 row in set (0.00 sec)
```

## LTRIM(str)

Mengembalikan string str dengan menghapus karakter spasi di depannya.

```
mysql> SELECT LTRIM(' barbar');

+-----------------------------------------------------------+
| LTRIM(' barbar')  |
+-----------------------------------------------------------+
| barbar   |
+-----------------------------------------------------------+
1 row in set (0.00 sec)
```

## ekspr REGEXP pola

Fungsi ini mencocokkan ekspr dengan pola. Mengembalikan 1 jika cocok, dan 0 jika tidak.

```
mysql> SELECT 'ABCDEF' REGEXP 'A%C%%';

+-----------------------------------------------------------+
| 'ABCDEF' REGEXP 'A%C%%'    |
+-----------------------------------------------------------+
| 0    |
+-----------------------------------------------------------+
1 row in set (0.00 sec)

mysql> SELECT 'ABCDE' REGEXP '.*';
```

```
+-----------------------------------------------------------+
| 'ABCDE' REGEXP '.*'    |
+-----------------------------------------------------------+
| 1    |
+-----------------------------------------------------------+
1 row in set (0.00 sec)

mysql> SELECT 'baris*\n*baru' REGEXP 'baris\\*.\\*baru';

+-----------------------------------------------------------+
| 'baris*\n*baru' REGEXP 'baris\\*.\\*baru' |
+-----------------------------------------------------------+
| 1    |
+-----------------------------------------------------------+
1 row in set (0.00 sec)
```

## REPEAT(str, jumlah)

Mengembalikan string yang terdiri dari string str yang diulang sebanyak jumlah kali.

```
mysql> SELECT REPEAT('MySQL', 3);

+-----------------------------------------------------------+
| REPEAT('MySQL', 3)    |
+-----------------------------------------------------------+
| MySQLMySQLMySQL    |
+-----------------------------------------------------------+
1 row in set (0.00 sec)
```

## REPLACE(str, str_awal, str_hasil)

Mengembalikan string str dengan semua str_awal dalam str diganti dengan str_hasil.

```
mysql> SELECT REPLACE('www.mysql.com', 'w', 'Ww');

+-----------------------------------------------------------+
| REPLACE('www.mysql.com', 'w', 'Ww')  |
+-----------------------------------------------------------+
| WwWwWw.mysql.com  |
+-----------------------------------------------------------+
1 row in set (0.00 sec)
```

## REVERSE(str)

Mengembalikan string str dengan urutan karakter dibalik.

```
mysql> SELECT REVERSE('abcd');

+-----------------------------------------------------------+
| REVERSE('abcd')    |
+-----------------------------------------------------------+
| dcba    |
+-----------------------------------------------------------+
1 row in set (0.00 sec)
```

## RIGHT(str, panj)

Mengembalikan sejumlah panj karakter dari sebelah kanan/akhir string str.

```
mysql> SELECT RIGHT('foobarbar', 4);

+-----------------------------------------------------------+
| RIGHT('foobarbar', 4)   |
+-----------------------------------------------------------+
| rbar    |
+-----------------------------------------------------------+
1 row in set (0.00 sec)
```

## RTRIM(str)

Mengembalikan string str dengan menghapus karakter spasi di belakangnya.

```
mysql> SELECT RTRIM('barbar ');

+-----------------------------------------------------------+
| RTRIM('barbar ')  |
+-----------------------------------------------------------+
| barbar   |
+-----------------------------------------------------------+
1 row in set (0.00 sec)
```

## SPACE(N)

Mengembalikan sejumlah N karakter spasi.

```
mysql> SELECT SPACE(6);

+-----------------------------------------------------------+
| SELECT SPACE(6)    |
+-----------------------------------------------------------+
|  |
+-----------------------------------------------------------+
1 row in set (0.00 sec)
```

## STRCMP(str1, str2)

Membandingkan dua string dan mengembalikan 0 jika kedua string sama, -1 jika str1 lebih kecil dari str2, dan 1 jika sebaliknya.

```
mysql> SELECT STRCMP('MOHD', 'MOHD');

+-----------------------------------------------------------+
| STRCMP('MOHD', 'MOHD')    |
+-----------------------------------------------------------+
| 0    |
+-----------------------------------------------------------+
1 row in set (0.00 sec)

mysql> SELECT STRCMP('AMOHD', 'MOHD');

+-----------------------------------------------------------+
| STRCMP('AMOHD', 'MOHD')    |
+-----------------------------------------------------------+
| -1   |
+-----------------------------------------------------------+
1 row in set (0.00 sec)

mysql> SELECT STRCMP('MOHD', 'AMOHD');

+-----------------------------------------------------------+
| STRCMP('MOHD', 'AMOHD')    |
+-----------------------------------------------------------+
| 1    |
+-----------------------------------------------------------+
1 row in set (0.00 sec)
```

## SUBSTRING(str, pos), SUBSTRING(str FROM pos), SUBSTRING(str, pos, panj), SUBSTRING(str FROM pos FOR panj)

Bentuk fungsi tanpa argumen panj mengembalikan substring dari string str dimulai dari posisi pos.

```
mysql> SELECT SUBSTRING('Quadratically',5);

+-----------------------------------------------------------+
| SSUBSTRING('Quadratically',5)   |
+-----------------------------------------------------------+
| ratically  |
+-----------------------------------------------------------+
1 row in set (0.00 sec)
```

Bentuk fungsi dengan argumen panj mengembalikan sejumlah panj karakter dari string str, dimulai dari posisi pos.

```
mysql> SELECT SUBSTRING('Quadratically',5,6);

+-----------------------------------------------------+
| SUBSTRING('Quadratically',5,6)  |
+-----------------------------------------------------+
| ratica   |
+-----------------------------------------------------+
1 row in set (0.00 sec)
```

Bentuk fungsi yang menggunakan FROM sama dengan yang menggunakan pos. Bentuk ini juga dapat menggunakan nilai negatif untuk memulai pos dari karakter akhir str.

```
mysql> SELECT SUBSTRING('foobarbar' FROM 4);

+-----------------------------------------------------+
| SUBSTRING('foobarbar' FROM 4)   |
+-----------------------------------------------------+
| barbar   |
+-----------------------------------------------------+
1 row in set (0.00 sec)
```

## TRIM( [ {KEDUANYA | AWAL | AKHIR} [remstr] FROM ] str), TRIM( [remstr FROM] str)

Mengembalikan string str dengan menghapus semua awalan atau akhiran remstr.

```
mysql> SELECT TRIM(' bar ');

+-----------------------------------------------------+
| TRIM(' bar ')  |
+-----------------------------------------------------+
| bar   |
+-----------------------------------------------------+
1 row in set (0.00 sec)

mysql> SELECT TRIM(LEADING 'x' FROM 'xxxbarxxx');

+-----------------------------------------------------+
| TRIM(LEADING 'x' FROM 'xxxbarxxx')  |
+-----------------------------------------------------+
| barxxx   |
+-----------------------------------------------------+
1 row in set (0.00 sec)

mysql> SELECT TRIM(BOTH 'x' FROM 'xxxbarxxx');

+-----------------------------------------------------+
| TRIM(BOTH 'x' FROM 'xxxbarxxx')  |
+-----------------------------------------------------+
| bar   |
+-----------------------------------------------------+
1 row in set (0.00 sec)
```

```
mysql> SELECT TRIM(TRAILING 'xyz' FROM 'barxxyz');

+-------------------------------------------------------+
| TRIM(TRAILING 'xyz' FROM 'barxxyz')  |
+-------------------------------------------------------+
| barx    |
+-------------------------------------------------------+
```

## UNHEX(str)

Menjalankan kebalikan dari fungsi HEX(str).

```
mysql> SELECT UNHEX('4D7953514C');

+-------------------------------------------------------+
| UNHEX('4D7953514C')    |
+-------------------------------------------------------+
| MySQL    |
+-------------------------------------------------------+
1 row in set (0.00 sec)
```

## UPPER(str)

Mengembalikan string str dengan mengubah semua karakter menjadi huruf kapital.

```
mysql> SELECT UPPER('Tes-huruf-kapital');

+-------------------------------------------------------+
| UPPER('Tes-huruf-kapital')  |
+-------------------------------------------------------+
| TES-HURUF-KAPITAL  |
+-------------------------------------------------------+
1 row in set (0.00 sec)
```

# Tentang Penulis

## JUBILEE ENTERPRISE

Jubilee Enterprise adalah "a Creative Media Content Provider" dengan misi "Mengeksplorasi Teknologi Informasi tercanggih di dunia dan menyajikannya dalam bentuk media dengan gaya bahasa yang sederhana, mudah dicerna, dan gampang dipraktikkan oleh siapa pun".

Di Jubilee Enterprise, "Information Technology is our passion". Itulah mengapa setiap hari kami mengeksplorasi, meneliti, dan bereksperimen dengan banyak teknologi tercanggih saat ini. Hasil penelitian tersebut kami persembahkan dalam bentuk media cetak (buku) dan elektronik (blog).

Buku-buku kami, yang diterbitkan oleh PT Elex Media Komputindo (Kelompok Kompas Gramedia), telah didistribusikan ke seluruh Indonesia dan Malaysia, membantu dan menginspirasi pembaca-pembaca kami ketika menggunakan program Photoshop, CorelDraw, MS Office, Internet, Gagdet, dan lain sebagainya secara mudah dan praktis.

***Catatan:***

Untuk melakukan pemesanan buku, hubungi
Layanan Langsung PT Elex Media Komputindo:
**Gramedia Direct**
Jl. Palmerah Barat No. 33, Jakarta 10270
Telemarketing/CS: 021-53650110/111 ext: 3901/3902
Email: **endang@gramediapublishers.com**

# MySQL untuk Pemula

Mengapa Anda harus mengenal dan mempelajari MySQL? Jika Anda tertarik dengan dunia web programming, terutama berbasis PHP, maka MySQL merupakan peranti lunak yang wajib dikuasai. MySQL membantu Anda mengelola database dan bisa dimanfaatkan untuk menciptakan website yang interaktif.

Buku ini mengupas MySQL secara khusus. Anda akan mengenal cara instalasi MySQL hingga berbagai metode pengolahan database. Kelebihan buku ini: kupasannya sangat sederhana, langkah demi langkah yang dibagi dalam tiap bab, dan lengkap dengan contoh scripting-nya.

Bagi Anda yang benar-benar ingin memulai dari awal, buku ini paling praktis untuk dipelajari. Diharapkan, Anda mengenal cara penggunaan MySQL untuk pengolahan database. (thinkjubilee.com)

PT ELEX MEDIA KOMPUTINDO
Kompas Gramedia Building
Jl. Palmerah Barat 29-37, Jakarta 10270
Telp. (021) 53650110-53650111, Ext 3214
Webpage: http://www.elexmedia.co.id

gramediana

Kelompok
Pemrograman
Keterampilan
☑ Tingkat Pemula
☑ Tingkat Menengah
☐ Tingkat Mahir
Jenis Buku
☑ Referensi
☑ Tutorial
☐ Latihan

ISBN 978-602-02-5390-9
9 786020 253909
121142516