Pertemuan 12

Pengenalan *Structured Query Language*

- Apa Itu SQL ?
- Membuat, Menampilkan, Membuka dan Menghapus Database
- Membuat, Mengubah dan Menghapus *Table*
- Menambah Record dengan INSERT
- Mengedit Record dengan UPDATE
- Menghapus Record dengan DELETE
- Menampilkan Record dengan SELECT

Apa Itu SQL ?

SQL merupakan singkatan dari *Structured Query Language*. SQL atau juga sering disebut sebagai query merupakan suatu bahasa (*language*) yang digunakan untuk mengakses database. SQL dikenalkan pertama kali dalam IBM pada tahun 1970 dan sebuah standar ISO dan ANSII ditetapkan untuk SQL. Standar ini tidak tergantung pada mesin yang digunakan (IBM, Microsoft atau Oracle). Hampir semua software database mengenal atau mengerti SQL. Jadi, perintah SQL pada semua software database hampir sama.

Terdapat 2 (dua) jenis perintah SQL, yaitu :

1. DDL atau Data Definition Language
   DDL merupakan perintah SQL yang berhubungan dengan pendefinisian suatu struktur database, dalam hal ini *database* dan *table*. Beberapa perintah dasar yang termasuk DDL ini antara lain :
   - CREATE
   - ALTER
   - RENAME
   - DROP
2. DML atau Data Manipulation Language
   DML merupakan perintah SQL yang berhubungan dengan manipulasi atau pengolahan data atau *record* dalam table. Perintah SQL yang termasuk dalam DML antara lain :
   - SELECT
   - INSERT
   - UPDATE
   - DELETE

Membuat, Menampilkan, Membuka dan Menghapus Database

Membuat Database

Sintaks umum SQL untuk membuat suatu database adalah sebagai berikut :

```
CREATE DATABASE [IF NOT EXISTS] nama_database;
```

Bentuk perintah di atas akan membuat sebuah database baru dengan nama `nama_database`. Aturan penamaan sebuah database sama seperti aturan penamaan sebuah variabel, dimana secara umum nama database boleh terdiri dari huruf, angka dan *under-score* (_). Jika database yang akan dibuat sudah ada, maka akan muncul pesan error. Namun jika ingin otomatis menghapus database yang lama jika sudah ada, aktifkan option `IF NOT EXISTS`. Setiap kita membuat database baru, maka sebenarnya MySQL akan membuat suatu folder (direktori) sesuai dengan nama databasenya yang ditempatkan secara *default* di C:\mysql\data. Di dalam folder tersebut nantinya akan terdapat file-file yang berhubungan dengan tabel dalam database.

Berikut ini contoh perintah untuk membuat database baru dengan nama "**mahasiswa**" :

```
CREATE DATABASE mahasiswa;
```

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

```
Query OK, 1 row affected (0.02 sec)
```

Membuat Database

Untuk melihat database yang baru saja dibuat atau yang sudah ada, dapat menggunakan perintah sebagai berikut :

```
SHOW DATABASES;
```

Hasil dari perintah di atas akan menampilkan semua database yang sudah ada di MySQL. Berikut ini contoh hasil dari query di atas :

```
+--------------+
| Database     |
+--------------+
| mahasiswa    |
| mysql        |
| test         |
+--------------+
3 rows in set (0.02 sec)
```

Membuka Database

Sebelum melakukan manipulasi tabel dan record yang berada di dalamnya, kita harus membuka atau mengaktifkan databasenya terlebih dahulu. Untuk membuka database "**mahasiswa**", berikut ini querynya :

```
USE mahasiswa;
```

Jika perintah atau query di atas berhasil, maka akan ditampilkan pesan sebagai berikut :

```
Database changed
```

Menghapus Database

Untuk menghapus suatu database, sintaks umumnya adalah sbb :

```
DROP DATABASE [IF EXISTS] nama_database;
```

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

Berikut ini contoh perintah untuk menghapus database dengan nama "**mahasiswa**" :

```
DROP DATABASE mahasiswa;
```

Membuat, Mengubah dan Menghapus *Table*

Membuat *Table*

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

```
CREATE TABLE nama_tabel (
field1 tipe(panjang),
field2 tipe(panjang),
...
fieldn tipe(panjang),
PRIMARY KEY (field_key)
);
```

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

MySQL menyediakan berbagai tipe data dengan spesifikasi dan panjang masing-masing. Tipe data untuk field dalam MySQL diantaranya ditampilkan pada tabel berikut ini :

JENIS TIPE	TIPE	KETERANGAN
NUMERIK	TINYINT	-128 s/d 127 SIGNED 0 s/d 255 UNSIGNED
	SMALLINT	-32768 s/d 32767 SIGNED 0 s/d 65535 UNSIGNED.
	MEDIUMINT	-8388608 s/d 8388607 SIGNED 0 s/d 16777215 UNSIGNED
	INT	-2147483648 s/d 2147483647 SIGNED 0 s/d 4294967295 UNSIGNED.
	BIGINT	-9223372036854775808 s/d 9223372036854775807 SIGNED 0 s/d 18446744073709551615 UNSIGNED.
	FLOAT	Bilangan pecahan presisi tunggal
	DOUBLE	Bilangan pecahan presisi ganda
	DECIMAL	Bilangan dengan desimal
DATE/TIME	DATE	Tanggal dengan format YYYY-MM-DD
	DATETIME	Tanggal dan waktu dengan format : YYYY-MM-DD HH:MM:SS
	TIMESTAMP	Tanggal dan waktu dengan format : YYYYMMDDHHMMSS
	TIME	Waktu dengan format HH:MM:SS
	YEAR	Tahun dengan format YYYY
STRING	CHAR	0 – 255 karakter
	VARCHAR	0 – 255 karakter
	TINYTEXT	String dengan panjang maksimum 255 karakter
	TEXT	String dengan panjang maksimum 65535 karakter
	BLOB	String dengan panjang maksimum 65535

		karakter
	MEDIUMTEXT	String dengan panjang maksimum 16777215 karakter
	MEDIUMBLOB	String dengan panjang maksimum 16777215 karakter
	LONGTEXT	String dengan panjang maksimum 4294967295 karakter
	LONGBLOB	String dengan panjang maksimum 4294967295 karakter
KHUSUS	ENUM	Tipe data dengan isi tertentu
	SET	Tipe data dengan isi tertentu

Sebagai contoh, kita akan membuat tabel baru dengan struktur sebagai berikut :

Nama tabel : **mhs**

No	Nama Field	Tipe	Panjang
1	**nim**	Varchar	10
2	nama	Varchar	30
3	tgllahir	Date	-
4	alamat	Text	-

Untuk membuat tabel tersebut di atas, query atau perintah SQL-nya adalah sebagai berikut :

```
CREATE TABLE mhs (
 nim varchar(10) NOT NULL,
 nama varchar(30) NOT NULL,
 tgllahir date,
 alamat text,
 PRIMARY KEY(nim)
);
```

Jika query untuk membuat tabel di atas berhasil dijalankan, maka akan ditampilkan pesan sebagai berikut :

```
Query OK, 0 rows affected (0.16 sec)
```

Pada perintah di atas, beberapa hal yang perlu diperhatikan :

- `CREATE TABLE` merupakan perintah dasar dari pembuatan table.
- `mhs` merupakan nama tabel yang akan dibuat.
- `Nim, nama, tgllahir` dan alamat merupakan nama field
- `Varchar, date` dan `text` merupakan tipe data dari field
- `NOT NULL` merupakan option untuk menyatakan bahwa suatu field tidak boleh kosong.
- `PRIMARY KEY` merupakan perintah untuk menentukan field mana yang akan dijadikan primary key pada tabel.
- `10` dan `30` di belakang tipe data merupakan panjang maksimal dari suatu field
- Untuk tipe data date dan text (dan beberapa tipe data lainnya) panjang karakter maksimalnya tidak perlu ditentukan.
- Jangan lupa akhiri perintah dengan titik-koma (;)

Selanjutnya untuk melihat tabel mhs sudah benar-benar sudah ada atau belum, ketikkan perintah berikut ini :

```
SHOW TABLES;
```

Perintah di atas akan menampilkan seluruh tabel yang sudah ada dalam suatu database. Contoh hasil dari perintah di atas adalah sebagai berikut :

```
+---------------------+
| Tables_in_mahasiswa |
+---------------------+
| mhs                 |
+---------------------+
1 rows in set (0.01 sec)
```

Untuk melihat struktur tabel "**mhs**" secara lebih detail, cobalah perintah atau query sebagai berikut :

```
DESC mhs;
```

DESC merupakan singkatan dari **DESCRIBE** (dalam query bisa ditulis lengkap atau hanya 4 karakter pertama) dan **mhs** adalah nama tabel yang akan dilihat strukturnya. Dari perintah di atas, akan ditampilkan struktur tabel **mhs** sebagai berikut :

```
+----------+-------------+------+-----+---------+-------+
| Field    | Type        | Null | Key | Default | Extra |
+----------+-------------+------+-----+---------+-------+
| nim      | varchar(10) |      | PRI |         |       |
| nama     | varchar(30) |      |     |         |       |
| tgllahir | date        | YES  |     | NULL    |       |
| alamat   | text        | YES  |     | NULL    |       |
+----------+-------------+------+-----+---------+-------+
4 rows in set (0.00 sec)
```

Dari struktur tabel mhs yang ditampilkan di atas, dapat diketahui bahwa :
- Terdapat 4 (empat) field dengan tipe masing-masing.
- Primary Key dari tabel mhs adalah nim. Lihat kolom **Key** pada field nim.
- Untuk field nim dan nama defaultnya tidak boleh kosong. Lihatlah kolom **Null** dan **Default** pada field nim dan nama
- Untuk field tgllahir dan alamat defaultnya boleh kosong. Lihatlah kolom **Null** dan **Default** pada field tgllahir dan alamat.

Mengubah Struktur *Table* dengan ALTER

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

```
ALTER TABLE nama_tabel alter_options;
```

dimana :
- `ALTER TABLE` merupakan perintah dasar untuk mengubah tabel.
- `nama_tabel` merupakan nama tabel yang akan diubah strukturnya.
- `alter_options` merupakan pilihan perubahan tabel. Option yang bisa digunakan, beberapa di antaranya sebagai berikut :
  - » `ADD definisi_field_baru`
    Option ini digunakan untuk menambahkan field baru dengan "**definisi_field_baru**" (nama field, tipe dan option lain).
  - » `ADD INDEX nama_index`

Option ini digunakan untuk menambahkan index dengan nama "**nama_index"** pada tabel.

» `ADD PRIMARY KEY (field_kunci)`
Option untuk menambahkan primary key pada tabel

» `CHANGE field_yang_diubah definisi_field_baru`
Option untuk mengubah field_yang_diubah menjadi definisi_field_baru

» `MODIFY definisi_field`
Option untuk mengubah suatu field menjadi definisi_field

» `DROP nama_field`
Option untuk menghapus field nama_field

» `RENAME TO nama_tabel_baru`
Option untuk mengganti nama tabel

Beberapa contoh variasi perintah ALTER untuk mengubah struktur suatu tabel antara lain :

1. Menambahkan field "**agama**" ke tabel **mhs**

```
ALTER TABLE mhs ADD agama varchar(15) NOT NULL;
```

2. Menambahkan primary key pada suatu tabel

```
ALTER TABLE mhs ADD PRIMARY KEY(nim);
```

3. Mengubah **panjang field agama** menjadi 10 karakter dalam tabel **mhs**

```
ALTER TABLE mhs CHANGE agama agama varchar(10);
```

4. Mengubah **tipe field agama** menjadi char(2) dalam tabel **mhs**

```
ALTER TABLE mhs MODIFY agama char(2) NOT NULL;
```

5. Menghapus field **agama** dari tabel **mhs**

```
ALTER TABLE mhs DROP agama;
```

Mengubah Nama Tabel

Untuk mengubah nama suatu tabel, dapat menggunakan perintah SQL sbb :

```
RENAME TABLE mhs TO mahasiswa;
ALTER TABLE mhs RENAME TO mahasiswa;
```

Perintah di atas akan mengubah tabel **mhs** menjadi **mahasiswa**.

Menghapus Tabel

Untuk menghapus sebuah tabel, bentuk umum dari perintah SQL adalah sebagai berikut :

```
DROP TABLE nama_tabel;
```

Contohnya kita akan menghapus tabel dengan nama "**mahasiswa**" maka perintah SQL-nya adalah :

```
DROP TABLE mahasiswa;
```

Menambah Record dengan INSERT

Bentuk umum perintah SQL untuk menambahkan *record* atau data ke dalam suatu tabel adalah sebagai berikut :

```
INSERT INTO nama_tabel VALUES ('nilai1','nilai2',...);
```

atau dapat dengan bentuk sebagai berikut :

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

atau dapat juga dengan bentuk sebagai berikut :

```
INSERT INTO nama_tabel
SET field1='nilai1', field2='nilai2',...;
```

Sebagai contoh, kita akan menambahkan sebuah record ke dalam tabel **mhs** yang telah kita buat sebelumnya. Berikut ini perintah SQL untuk menambahkan sebuah record ke dalam tabel **mhs** :

```
INSERT INTO mhs VALUES ('0411500121','Achmad Solichin',
'1982-06-05','Jakarta Selatan');
```

Jika perintah SQL di atas berhasil dieksekusi maka akan ditampilkan pesan sebagai berikut :

```
Query OK, 1 row affected (0.00 sec)
```

Setelah perintah SQL di atas berhasil dieksekusi, maka record atau data dalam tabel mhs akan bertambah. Jalankan perintah berikut ini untuk melihat isi tabel mhs !

```
SELECT * FROM mhs;
```

Dan berikut ini hasil dari perintah SQL di atas :

```
+------------+-----------------+------------+-----------------+
| nim        | nama            | tgllahir   | alamat          |
+------------+-----------------+------------+-----------------+
| 0411500121 | Achmad Solichin | 1982-06-05 | Jakarta Selatan |
+------------+-----------------+------------+-----------------+
1 row in set (0.19 sec)
```

Latihan

Tambahkan 10 data (record) baru ke tabel **mhs** sehingga isi tabel mhs menjadi sebagai berikut !

```
+------------+-----------------+------------+-----------------+
| nim        | nama            | tgllahir   | alamat          |
+------------+-----------------+------------+-----------------+
| 0411500121 | Achmad Solichin | 1982-06-05 | Jakarta Selatan |
| 0411500123 | Chotimatul M    | 1983-03-12 | Jakarta Selatan |
| 0422500111 | Bajuri          | 1983-03-25 | Tangerang       |
| 0444500011 | Oneng           | 1980-05-22 | Jakarta Utara   |
| 0433500115 | Unyil           | 1980-08-29 | Tangerang       |
```

```
| 0411500116 | Ujang           | 1984-10-06 | Jakarta Barat   |
| 0422500316 | Jebleh          | 1984-10-06 | Cengkareng      |
| 0433500333 | Dono            | 1984-10-06 | Jakarta Selatan |
| 0422500433 | Dini            | 1986-12-10 | Jakarta Selatan |
| 0411500331 | Dana            | 1986-07-11 | Jakarta Selatan |
| 0444500315 | Dani            | 1985-01-01 | Jakarta Barat   |
+------------+-----------------+------------+-----------------+
```

Mengedit Record dengan UPDATE

Proses update bisa sewaktu-waktu dilakukan jika terdapat data atau record dalam suatu tabel yang perlu diperbaiki. Proses update ini tidak menambahkan data (record) baru, tetapi memperbaiki data yang lama. Perubahan yang terjadi dalam proses update bersifat permanen, artinya setelah perintah dijalankan tidak dapat di-*cancel* (*undo*).

Bentuk umum perintah SQL untuk mengedit suatu *record* atau data dari suatu tabel adalah sebagai berikut :

```
UPDATE nama_tabel SET field1='nilaibaru'
[WHERE kondisi];
```

Pada perintah untuk update di atas :

- `UPDATE` merupakan perintah dasar untuk mengubah *record* tabel.
- `nama_tabel` merupakan nama tabel yang akan diubah *recordnya*.
- Perintah `SET` diikuti dengan *field-field* yang akan diubah yang mana diikuti juga dengan perubahan isi dari masing-masing *field*. Untuk mengubah nilai dari beberapa *field* sekaligus, gunakan koma (,) untuk memisahkan masing-masing *field*.
- Perintah `WHERE` diikuti oleh kondisi tertentu yang menentukan record mana yang akan diedit (diubah). Perintah `WHERE` ini boleh ada boleh juga tidak. Jika `WHERE` tidak ditambahkan pada perintah update maka semua *record* dalam tabel bersangkutan akan berubah.

Perhatikan beberapa contoh perintah UPDATE tabel **mhs** berikut ini !

1. Mengubah alamat menjadi "Tangerang" untuk mahasiswa yang mempunyai nim 0411500121

```
UPDATE mhs SET alamat='Tangerang' WHERE
nim='0411500121';
```

Dan jika query di atas berhasil dieksekusi maka akan ditampilkan hasil sebagai berikut :

```
Query OK, 1 row affected (0.27 sec)
Rows matched: 1  Changed: 1  Warnings: 0
```

2. Mengubah tanggal lahir menjadi "12 Maret 1983" dan alamat menjadi "Bandung" untuk mahasiswa yang mempunyai nim 0422500316

```
UPDATE mhs SET tgllahir='1982-06-05', alamat='Jakarta
Selatan' WHERE nim='0422500316';
```

Menghapus Record dengan DELETE

Proses delete dilakukan jika terdapat data atau record dalam suatu tabel yang perlu dihapus atau dihilangkan. Perubahan yang terjadi dalam proses *delete* bersifat permanen, artinya setelah perintah dijalankan tidak dapat di-*cancel* (*undo*). Jadi berhati-hatilah dengan perintah *delete* !

Bentuk umum perintah SQL untuk menghapus suatu *record* atau data dari tabel adalah sebagai berikut :

```
DELETE FROM nama_tabel [WHERE kondisi];
```

Pada perintah untuk *delete* di atas :

- `DELETE FROM` merupakan perintah dasar untuk menghapus suatu *record* dari tabel.
- `nama_tabel` merupakan nama tabel yang akan dihapus *recordnya*.
- Perintah `WHERE` diikuti oleh kondisi tertentu yang menentukan record mana yang akan dihapus (didelete). Perintah `WHERE` ini boleh ada boleh juga tidak. Namun demikian, jika `WHERE` tidak ditambahkan pada perintah delete maka semua *record* dalam tabel bersangkutan akan **terhapus**. Jadi jangan lupa menambahkan WHERE jika kita tidak bermaksud mengosongkan tabel

Perhatikan beberapa contoh perintah DELETE dari tabel **mhs** berikut ini !

1. Menghapus data mahasiswa yang mempunyai nim 0411500331

```
DELETE FROM mhs WHERE nim='0411500331';
```

Dan jika query di atas berhasil dieksekusi maka akan ditampilkan hasil sebagai berikut :

```
Query OK, 1 row affected (0.11 sec)
```

2. Menghapus semua mahasiswa yang beralamat di "Bandung"

```
DELETE FROM mhs WHERE alamat='Bandung';
```

Menampilkan Record dengan SELECT

Perintah SELECT digunakan untuk menampilkan sesuatu. Sesuatu di sini bisa berupa sejumlah data dari tabel dan bisa juga berupa suatu ekspresi. Dengan SELECT kita bisa mengatur tampilan atau keluaran sesuai tampilan yang diinginkan.

Bentuk dasar perintah SELECT data dari tabel adalah sebagai berikut :

```
SELECT [field | *] FROM nama_tabel [WHERE kondisi];
```

Perhatikan beberapa contoh perintah SELECT dari tabel **mhs** berikut ini !

1. Menampilkan seluruh data atau record (*) dari tabel **mhs**

```
SELECT * FROM mhs;
```

Dan jika query di atas berhasil dieksekusi maka akan ditampilkan hasil sebagai berikut :

```
+------------+-----------------+------------+-----------------+
| nim        | nama            | tgllahir   | alamat          |
+------------+-----------------+------------+-----------------+
| 0411500121 | Achmad Solichin | 1982-06-05 | Jakarta Selatan |
| 0411500123 | Chotimatul M    | 1983-03-12 | Jakarta Selatan |
| 0422500111 | Bajuri          | 1983-03-25 | Tangerang       |
| 0444500011 | Oneng           | 1980-05-22 | Jakarta Utara   |
| 0433500115 | Unyil           | 1980-08-29 | Tangerang       |
| 0411500116 | Ujang           | 1984-10-06 | Jakarta Barat   |
| 0422500316 | Jebleh          | 1982-06-05 | Jakarta Selatan |
| 0433500333 | Dono            | 1984-10-06 | Jakarta Selatan |
| 0422500433 | Dini            | 1986-12-10 | Jakarta Selatan |
| 0444500315 | Dani            | 1985-01-01 | Jakarta Barat   |
+------------+-----------------+------------+-----------------+
10 rows in set (0.25 sec)
```

2. Menampilkan *field* **nim** dan **nama** dari seluruh mahasiswa dalam tabel **mhs**

```
SELECT nim, nama FROM mhs;
```

Jika query di atas berhasil dieksekusi maka akan ditampilkan hasil sebagai berikut :

```
+------------+-----------------+
| nim        | nama            |
+------------+-----------------+
| 0411500121 | Achmad Solichin |
| 0411500123 | Chotimatul M    |
| 0422500111 | Bajuri          |
| 0444500011 | Oneng           |
| 0433500115 | Unyil           |
| 0411500116 | Ujang           |
| 0422500316 | Jebleh          |
| 0433500333 | Dono            |
| 0422500433 | Dini            |
| 0444500315 | Dani            |
+------------+-----------------+
10 rows in set (0.11 sec)
```

3. Menampilkan data mahasiswa yang mempunyai nim **0411500123**

```
SELECT * FROM mhs WHERE nim = '0411500123';
```

Hasil query di atas adalah sbb :

```
+------------+--------------+------------+-----------------+
| nim        | nama         | tgllahir   | alamat          |
+------------+--------------+------------+-----------------+
| 0411500123 | Chotimatul M | 1983-03-12 | Jakarta Selatan |
+------------+--------------+------------+-----------------+
1 row in set (0.06 sec)
```

4. Menampilkan data semua mahasiswa yang beralamat di luar **Jakarta Selatan**

```
SELECT * FROM mhs WHERE alamat != 'Jakarta Selatan';
```

Hasil query di atas adalah sbb :

```
+------------+--------+------------+---------------+
| nim        | nama   | tgllahir   | alamat        |
+------------+--------+------------+---------------+
| 0422500111 | Bajuri | 1983-03-25 | Tangerang     |
| 0444500011 | Oneng  | 1980-05-22 | Jakarta Utara |
| 0433500115 | Unyil  | 1980-08-29 | Tangerang     |
| 0411500116 | Ujang  | 1984-10-06 | Jakarta Barat |
| 0444500315 | Dani   | 1985-01-01 | Jakarta Barat |
+------------+--------+------------+---------------+
5 rows in set (0.02 sec)
```

Berikut ini operator **perbandingan** yang dapat digunakan untuk membandingkan dua buah nilai dalam MySQL :

- **Operator =**, akan bernilai TRUE jika nilai yang dibandingkan sama.
- **Operator !=** atau **<>**, akan bernilai TRUE jika nilai yang dibandingkan TIDAK SAMA (berbeda).
- **Operator >**, akan bernilai TRUE jika nilai yang pertama lebih besar dari nilai kedua.
- **Operator >=**, akan bernilai TRUE jika nilai yang pertama lebih besar atau sama dengan nilai kedua.
- **Operator <**, akan bernilai TRUE jika nilai yang pertama lebih kecil dari nilai kedua.
- **Operator <=**, akan bernilai TRUE jika nilai yang pertama lebih kecil atau sama dengan nilai kedua.

5. Menampilkan data semua mahasiswa yang beralamat di **Jakarta Selatan** dan lahir pada tahun **1982**.

```
SELECT * FROM mhs WHERE alamat = 'Jakarta Selatan' &&
YEAR(tgllahir) = '1982';
```

Hasil query di atas adalah sbb :

```
+------------+-----------------+------------+-----------------+
| nim        | nama            | tgllahir   | alamat          |
+------------+-----------------+------------+-----------------+
| 0411500121 | Achmad Solichin | 1982-06-05 | Jakarta Selatan |
| 0422500316 | Jebleh          | 1982-06-05 | Jakarta Selatan |
+------------+-----------------+------------+-----------------+
2 rows in set (0.00 sec)
```

Berikut ini operator **penghubung** yang dapat digunakan untuk menghubungkan antara dua kondisi dalam MySQL :

- **Operator &&** atau **AND**, akan menghubungkan dua kondisi dimana akan bernilai TRUE jika kedua kondisi bernilai TRUE.
- **Operator ||** atau **OR**, akan menghubungkan dua kondisi dimana akan bernilai TRUE jika salah satu atau kedua kondisi bernilai TRUE.
- **Operator !**, akan me-*reverse* nilai suatu kondisi logika.

Keterangan

Fungsi **YEAR** pada query di atas akan menghasilkan nilai TAHUN dari suatu tanggal. Selain fungsi YEAR, juga terdapat fungsi MONTH yang akan menghasilkan nama BULAN dari tanggal, fungsi DAY yang akan menghasilkan

hari dari suatu tanggal, dan masih banyak fungsi lain yang berhubungan dengan tanggal.

6. Menampilkan **nim, nama** dan **umur** dari semua mahasiswa.

```
SELECT nim, nama, YEAR(now())-YEAR(tgllahir) AS umur
 FROM mhs;
```

Hasil query di atas adalah sbb :

```
+------------+-----------------+------+
| nim        | nama            | umur |
+------------+-----------------+------+
| 0411500121 | Achmad Solichin |   24 |
| 0411500123 | Chotimatul M    |   23 |
| 0422500111 | Bajuri          |   23 |
| 0444500011 | Oneng           |   26 |
| 0433500115 | Unyil           |   26 |
| 0411500116 | Ujang           |   22 |
| 0422500316 | Jebleh          |   24 |
| 0433500333 | Dono            |   22 |
| 0422500433 | Dini            |   20 |
| 0444500315 | Dani            |   21 |
+------------+-----------------+------+
10 rows in set (0.05 sec)
```

Keterangan
Pada query di atas terdapat fungsi **YEAR** yang akan mengambil tahun dari suatu tanggal. Selanjutnya fungsi **now**() akan me-*return* tanggal dan waktu sistem saat query dieksekusi. Proses perhitungan umur dialiaskan dengan nama '**umur'**. Untuk mengaliaskan gunakan perintah **AS** yang diikuti nama alias.

7. Menampilkan semua mahasiswa **jurusan TI**

```
SELECT * FROM mhs WHERE SUBSTRING(nim,3,2)='11';
```

Hasil query di atas adalah sbb :

```
+------------+-----------------+------------+-----------------+
| nim        | nama            | tgllahir   | alamat          |
+------------+-----------------+------------+-----------------+
| 0411500121 | Achmad Solichin | 1982-06-05 | Jakarta Selatan |
| 0411500123 | Chotimatul M    | 1983-03-12 | Jakarta Selatan |
| 0411500116 | Ujang           | 1984-10-06 | Jakarta Barat   |
+------------+-----------------+------------+-----------------+
3 rows in set (0.19 sec)
```

Keterangan
Pada query di atas terdapat fungsi **SUBSTRING** yang berguna untuk memotong suatu string. Format fungsi SUBSTRING adalah sebagai berikut :

```
SUBSTRING(field, awal, panjang)
```

8. Menampilkan semua data mahasiswa secara urut berdasarkan **nama** dengan perintah **ORDER BY**

```
SELECT * FROM mhs ORDER BY nama;
```

Hasil query di atas adalah sbb :

```
+------------+-----------------+------------+-----------------+
| nim        | nama            | tgllahir   | alamat          |
+------------+-----------------+------------+-----------------+
| 0411500121 | Achmad Solichin | 1982-06-05 | Jakarta Selatan |
| 0422500111 | Bajuri          | 1983-03-25 | Tangerang       |
| 0411500123 | Chotimatul M    | 1983-03-12 | Jakarta Selatan |
| 0444500315 | Dani            | 1985-01-01 | Jakarta Barat   |
| 0422500433 | Dini            | 1986-12-10 | Jakarta Selatan |
| 0433500333 | Dono            | 1984-10-06 | Jakarta Selatan |
| 0422500316 | Jebleh          | 1982-06-05 | Jakarta Selatan |
| 0444500011 | Oneng           | 1980-05-22 | Jakarta Utara   |
| 0411500116 | Ujang           | 1984-10-06 | Jakarta Barat   |
| 0433500115 | Unyil           | 1980-08-29 | Tangerang       |
+------------+-----------------+------------+-----------------+
10 rows in set (0.01 sec)
```

9. Menampilkan semua data mahasiswa secara urut berdasarkan **nim** secara **DESCENDING**

```
SELECT * FROM mhs ORDER BY nim DESC;
```

Hasil query di atas adalah sbb :

```
+------------+-----------------+------------+-----------------+
| nim        | nama            | tgllahir   | alamat          |
+------------+-----------------+------------+-----------------+
| 0444500315 | Dani            | 1985-01-01 | Jakarta Barat   |
| 0444500011 | Oneng           | 1980-05-22 | Jakarta Utara   |
| 0433500333 | Dono            | 1984-10-06 | Jakarta Selatan |
| 0433500115 | Unyil           | 1980-08-29 | Tangerang       |
| 0422500433 | Dini            | 1986-12-10 | Jakarta Selatan |
| 0422500316 | Jebleh          | 1982-06-05 | Jakarta Selatan |
| 0422500111 | Bajuri          | 1983-03-25 | Tangerang       |
| 0411500123 | Chotimatul M    | 1983-03-12 | Jakarta Selatan |
| 0411500121 | Achmad Solichin | 1982-06-05 | Jakarta Selatan |
| 0411500116 | Ujang           | 1984-10-06 | Jakarta Barat   |
+------------+-----------------+------------+-----------------+
10 rows in set (0.00 sec)
```

10. Menampilkan 5 record (data) pertama dari tabel **mhs** secara urut berdasarkan **nim** dengan **LIMIT**

```
SELECT * FROM mhs ORDER BY nim LIMIT 0,5;
```

Hasil query di atas adalah sbb :

```
+------------+-----------------+------------+-----------------+
| nim        | nama            | tgllahir   | alamat          |
+------------+-----------------+------------+-----------------+
| 0411500116 | Ujang           | 1984-10-06 | Jakarta Barat   |
| 0411500121 | Achmad Solichin | 1982-06-05 | Jakarta Selatan |
| 0411500123 | Chotimatul M    | 1983-03-12 | Jakarta Selatan |
| 0422500111 | Bajuri          | 1983-03-25 | Tangerang       |
| 0422500316 | Jebleh          | 1982-06-05 | Jakarta Selatan |
+------------+-----------------+------------+-----------------+
5 rows in set (0.13 sec)
```

Keterangan

Pada query di atas bentuk **LIMIT** digunakan untuk membatasi hasil tampilan. LIMIT banyak digunakan untuk menampilkan data yang relatif banyak. Format fungsi LIMIT adalah sebagai berikut :

```
LIMIT awal, jumlah_record
```