Data Manipulation Language (DML)

Oleh : Andri Heryandi, MT, (andri@unikom.ac.id) Teknik Informatika – Universitas Komputer Indonesia (UNIKOM) - 2010

DML

Perintah SQL yang termasuk dalam DML adalah :

- INSERT
- SELECT
- UPDATE
- REPLACE
- DELETE
- TRUNCATE

INSERT

- INSERT digunakan untuk menambahkan data ke suatu tabel.
- Bentuk umum pemanggilan insert adalah :

```
INSERT [INTO] nama_tabel[(nama_kolom1[, nama_kolom2, ...])]
  {VALUE | VALUES} (isi_kolom1[, isi_kolom2, ...])
```

- Keterangan :
  - INTO sifatnya opsional. Boleh ditulis, boleh tidak.
  - Daftar nama kolom boleh ditulis, boleh tidak. Jika tidak ditulis, maka dianggap sesuai urutan nama kolom dalam pendefinisian tabel (lihat dengan perintah DESC namatabel. Gunakan koma sebagai pemisah.
  - {VALUE | VALUES}, wajib ditulis. Boleh VALUE atau VALUES.
  - Isi kolom harus sesuai dengan urutan daftar nama kolom. Boleh diisi ekspresi atau DEFAULT. Gunakan koma sebagai pemisah.

INSERT

- Contoh :

```
create table member(
no int primary key auto_increment,
nama varchar(50) not null,
tgllhr date,
berat decimal(6,2),
status enum('A','T') default 'A'
);
```

INSERT

Contoh :

```
insert into member value(null,'Adi','1980-06-10',65.5,'A');
```

- Keterangan :
  - Menggunakan **INTO**
  - Daftar isi kolom tidak disebutkan, sehingga semua kolom harus disebut.
  - Menggunakan **VALUE**.
  - Kolom yang **AUTO_INCREMENT** diisi dengan nilai **NULL**.
  - Pengisian data untuk setiap kolom ditulis sesuai urutan kolom dalam pendefinisian tabel.

INSERT

Contoh :

```
insert member values(null,'Budi','1985-12-10',50,'T');
```

- Keterangan :
  - Tidak menggunakan **INTO**
  - Daftar isi kolom tidak disebutkan, sehingga semua kolom harus disebut.
  - Menggunakan **VALUES**.
  - Kolom yang **AUTO_INCREMENT** diisi dengan nilai **NULL**.
  - Pengisian data untuk setiap kolom ditulis sesuai urutan kolom dalam pendefinisian tabel.

INSERT

Contoh :

```
insert into member(nama,status) values('Eni',DEFAULT);
```

- Keterangan :
  - Daftar isi kolom disebutkan (nama, status), sehingga hanya kolom tertentu yang akan disebut isinya.
  - Kolom status diisi dengan DEFAULT. Ini menyatakan bahwa kolom tersebut akan diisi dengan nilai DEFAULT kolom. Dalam hal ini kolom status akan berisi ‘A’.

INSERT

Contoh :

```
insert into member values(null,'Febri','1990-10-
12',60,'T');
select last_insert_id();
```

- Keterangan :
  - Pengisian data dengan nama Febri.
  - Untuk melihat nilai hasil AUTO_INCREMENT, gunakan function LAST_INSERT_ID().

INSERT

Contoh :

```
insert into member (nama,tgllhr) values('Gungun',now());
```

- Keterangan :
  - Pengisian data untuk kolom menggunakan function now() yang berguna untuk mengisi data tanggal sekarang di komputer.

INSERT

Contoh :

```
insert into member (no,nama) values(10,'Hendri');
insert into member (no,nama) values(null,'Ina');
```

- Keterangan :
  - Pada insert yang pertama, nilai auto_increment diisi secara manual dengan nilai 10.
  - Insert ke-dua akan menghasilkan nilai auto_increment 11 karena nilai auto_increment sebelumnya telah diisi dengan nilai 10.

INSERT

Contoh salah:

```
insert into member (no,nama) values(10,'Hendra');
```

- Keterangan :
  - Akan menghasilkan error "**ERROR 1062 (23000): Duplicate entry '10' for key 'PRIMARY'**", karena nomor dengan nilai 10 sudah ada di tabel.

INSERT

Contoh salah:

```
insert into member values(null,'Rendi','19900601',67);
```

- Keterangan :
  - Akan menghasilkan error "**ERROR 1136 (21S01): Column count doesn't match value count at row 1**", karena banyaknya kolom di daftar isian kolom (4 kolom) tidak sesuai dengan banyaknya kolom dalam pendefinisian kolom (5 kolom).

INSERT

Contoh salah:

```
insert into member(nama,tgllhr)
values('Jefry','19900601',67);
```

- Keterangan :
  - Akan menghasilkan error "**ERROR 1136 (21S01): Column count doesn't match value count at row 1**", karena banyaknya kolom di daftar isian kolom (3 kolom) tidak sesuai dengan banyaknya kolom yang akan diisi (2 kolom).

INSERT

Contoh salah:

```
insert into member(nama,berat) values('Jefry', 'a67');
```

- Keterangan :
  - Akan menghasilkan error "**ERROR 1366 (HY000): Incorrect decimal value: 'a67' for column 'berat' at row 1**", karena adanya kesalahan dalam pengisian data. Seharusnya diisi dengan angka.

INSERT

Contoh salah:

```
insert into member(nama,berat) values('Jefry', 10000);
```

- Keterangan :
  - Akan menghasilkan error "**ERROR 1264 (22003): Out of range value for column 'berat' at row 1**", karena adanya kesalahan dalam pengisian data. Karena menggunakan decimal(6,2) maka isi kolom berat tidak boleh lebih dari 9999.99.

INSERT

Contoh salah:

```
insert into member(tgllhr,berat) values(1995,55);
```

- Keterangan :
  - Akan menghasilkan error "**ERROR 1364 (HY000): Field 'nama' doesn't have a default value**", karena adanya kesalahan yang diakibatkan oleh karena kolom nama tidak diisi dan tidak mempunyai nilai value. Hal ini menjadi error karena nama didefinisikan sebagai NOT NULL,

INSERT ... SELECT

- INSERT … SELECT digunakan untuk melakukan pengisian data yang datanya diambil dari tabel lain.
- Bentuk umum pemanggilan insert adalah :

```
INSERT [INTO] nama_tabel[(nama_kolom1[, nama_kolom2, ...])]
  SELECT {*|daftar_kolom} FROM nama_tabel_sumber;
```

- Keterangan :
  - Daftar kolom INSERT harus sesuai dengan daftar_kolom dalam SELECT.
  - Tidak ada VALUE/VALUES. Bagian VALUE/VALUES diganti dengan SELECT.

INSERT ... SELECT

Contoh salah:

```
create table negara(
    kode varchar(3) primary key,
    nama varchar(52)
);
insert into negara select code,name from world.country;
```

- Keterangan :
  - SQL pertama akan membuat tabel negara.
  - SQL kedua akan melakukan insert ke tabel negara dengan isi diambil dari SQL select dari tabel country yang ada di skema world.

INSERT DELAYED

- INSERT DELAYED digunakan untuk melakukan pengisian data dimana server akan menempatkan baris yang akan diinsertkan ke buffer. Proses insert akan dilakukan ketika tabel sedang tidak digunakan oleh proses lain. Dengan cara ini, penyisipan akan dilakukan tanpa harus menunggu status insertnya selesai.

```
INSERT  DELAYED
  [INTO] nama_tabel[(nama_kolom1[, nama_kolom2, ...])]
  {VALUE | VALUES} (isi_kolom1[, isi_kolom2, ...])
```

- Keterangan :
  - DELAYED ditempatkan setelah perintah INSERT.
  - Hanya berlaku untuk tabel berengine MyISAM, MEMORY, ARCHIVE.

INSERT DELAYED

Contoh:

```
create table demodelayed(angka int) Engine=MyISAM;
insert delayed into demodelayed values(10);
```

- Keterangan :
  - SQL pertama akan membuat tabel negara.
  - SQL kedua akan melakukan insert ke tabel negara dengan isi diambil dari SQL select dari tabel country yang ada di skema world.

INSERT ... ON DUPLICATE KEY UPDATE

- Perintah INSERT … ON DUPLICATE KEY UPDATE digunakan untuk menyisipkan data (INSERT). Jika data yang dimasukkan ternyata mengakibatkan DUPLICATE KEY, maka operasi yang akan dilakukan bukanlah INSERT tapi UPDATE.
- Perintah ini akan menghidari terjadinya error DUPLICATE KEY ketika ada pengisian data baru.

```
INSERT [INTO] nama_tabel[(nama_kolom1[, nama_kolom2, ...])]
  {VALUE | VALUES} (isi_kolom1[, isi_kolom2, ...])
  ON DUPLICATE KEY UPDATE
     kolom_update1=isi1[,kolom_update2=isi2];
```

INSERT ... ON DUPLICATE KEY UPDATE

Contoh:

```
create table demodupkey(angka int primary key, angka2 int);
insert into demodupkey values(1,2);
insert into demodupkey values (2,100);
insert into demodupkey values (1,5); -- Akan ERROR
insert into demodupkey values (1,5)
   on duplicate key update angka2=values(angka2);
```

- Keterangan :
  - Pada SQL ke-4, akan terjadi error jika menginsert data dengan angka 1 (karena sudah ada).
  - SQL-5 tidak akan menghasilkan error, karena menggunakan ON DUPLICATE KEY UPDATE. Ketika terjadi DUPLICATE UPDATE, maka akan mengupdate kolom angka2 dengan nilai baru (values) yang diinsertkan. Value digunakan untuk mengambil nilai dari baris yang diinsertkan.

MULTIROW INSERT

- Multirow insert digunakan untuk melakukan penambahan lebih dari 1 baris baru dalam sekali INSERT.
- Bentuk umum pemanggilan multirow insert adalah :

```
INSERT [INTO] nama_tabel[(nama_kolom1[, nama_kolom2, ...])]
  {VALUE | VALUES} (isi_kolom1[, isi_kolom2, ...])
                   [, (isi_kolom1[, isi_kolom2, ...])]
```

- Keterangan :
  - Pada bagian setelah VALUES, daftar isi kolom bisa dilakukan lebih dari 1 kali.

MULTIROW INSERT

Contoh:

```
create table demomultirow(angka int);

insert into demomultirow values(5),(7),(4),(3);
```

- Keterangan :
  - SQL insert akan melakukan pengisian data sebanyak 4 baris dalam satu kali eksekusi query INSERT.
  - Setiap daftar isi kolom dipisahkan dengan koma (,).

SELECT

- SELECT digunakan untuk mengambil data dari database.
- Bentuk umum pemanggilan SELECT adalah :

```
SELECT {*|daftar_kolom} FROM nama_tabel
[WHERE kondisi_where]
```

- Keterangan :
  - * digunakan untuk menampilkan semua kolom yang ada dalam tabel
  - Daftar_kolom digunakan untuk mengampikan kolom tertentu saja
  - WHERE digunakan jika ingin membatasi data yang ditampilkan

- Perintah SELECT lebih lanjut akan diterangkan pada bab-bab berikutnya.

SELECT

Contoh:

```
Select * from member;
```

Menghasilkan :

```
+----+--------+------------+-------+--------+
| no | nama   | tgllhr     | berat | status |
+----+--------+------------+-------+--------+
|  1 | Adi    | 1980-06-10 | 65.50 | A      |
|  2 | Budi   | 1985-12-10 | 50.00 | T      |
|  3 | Catur  | NULL       |  NULL | A      |
|  4 | Eni    | NULL       |  NULL | A      |
|  5 | Febri  | 1990-10-12 | 60.00 | T      |
|  6 | Gungun | 2010-10-25 |  NULL | A      |
| 10 | Hendri | NULL       |  NULL | A      |
| 11 | Ina    | NULL       |  NULL | A      |
+----+--------+------------+-------+--------+
8 rows in set (0.00 sec)
```

SELECT

Contoh:

```
select no,nama,tgllhr from member;
```

Menghasilkan :

```
+----+--------+------------+
| no | nama   | tgllhr     |
+----+--------+------------+
|  1 | Adi    | 1980-06-10 |
|  2 | Budi   | 1985-12-10 |
|  3 | Catur  | NULL       |
|  4 | Eni    | NULL       |
|  5 | Febri  | 1990-10-12 |
|  6 | Gungun | 2010-10-25 |
| 10 | Hendri | NULL       |
| 11 | Ina    | NULL       |
+----+--------+------------+
8 rows in set (0.00 sec)
```

SELECT

Contoh:

```
select no,nama,tgllhr from member where no>5;
```

Menghasilkan :

```
+----+--------+------------+
| no | nama   | tgllhr     |
+----+--------+------------+
|  6 | Gungun | 2010-10-25 |
| 10 | Hendri | NULL       |
| 11 | Ina    | NULL       |
+----+--------+------------+
3 rows in set (0.00 sec)
```

UPDATE

- UPDATE digunakan untuk mengganti isi data.
- Bentuk umum penggunaan update adalah :

```
UPDATE nama_tabel
SET kolom1={isi|DEFAULT} [, kolom2={expr2|DEFAULT}] ...]
[WHERE kondisi_where]
[ORDER BY nama_kolom_pengurutan]
[LIMIT banyak_baris]
```

- Keterangan :
  - WHERE digunakan untuk membatasi banyaknya baris yang diupdate
  - ORDER BY digunakan untuk melakukan pengurutan data yang akan diupdate.
  - LIMIT digunakan untuk membatasi baris yang akan diupdate (setelah data dibatasi dengan WHERE).

UPDATE

Contoh data :

```
select * from member;
```

Menghasilkan :

```
+----+--------+------------+-------+--------+
| no | nama   | tgllhr     | berat | status |
+----+--------+------------+-------+--------+
|  1 | Adi    | 1980-06-10 | 65.50 | A      |
|  2 | Budi   | 1985-12-10 | 50.00 | T      |
|  3 | Catur  | NULL       |  NULL | A      |
|  4 | Eni    | NULL       |  NULL | A      |
|  5 | Febri  | 1990-10-12 | 60.00 | T      |
|  6 | Gungun | 2010-10-25 |  NULL | A      |
| 10 | Hendri | NULL       |  NULL | A      |
| 11 | Ina    | NULL       |  NULL | A      |
+----+--------+------------+-------+--------+
8 rows in set (0.00 sec)
```

UPDATE

Contoh :

```
update member set nama=upper(nama);
```

Keterangan :

Update seluruh (tanpa where) baris dengan mengisi (set) kolom nama dengan upper(nama).

```
+----+--------+------------+-------+--------+
| no | nama   | tgllhr     | berat | status |
+----+--------+------------+-------+--------+
|  1 | ADI    | 1980-06-10 | 65.50 | A      |
|  2 | BUDI   | 1985-12-10 | 50.00 | T      |
|  3 | CATUR  | NULL       |  NULL | A      |
|  4 | ENI    | NULL       |  NULL | A      |
|  5 | FEBRI  | 1990-10-12 | 60.00 | T      |
|  6 | GUNGUN | 2010-10-25 |  NULL | A      |
| 10 | HENDRI | NULL       |  NULL | A      |
| 11 | INA    | NULL       |  NULL | A      |
+----+--------+------------+-------+--------+
8 rows in set (0.00 sec)
```

UPDATE

Contoh :

```
update member set tgllhr=19000101 where tgllhr is null;
```

Keterangan :

Update baris yang tgllahir-nya null dengan mengisi (set) kolom tgllhr dengan tanggal 1-Januari-1900.

```
+----+--------+------------+-------+--------+
| no | nama   | tgllhr     | berat | status |
+----+--------+------------+-------+--------+
|  1 | ADI    | 1980-06-10 | 65.50 | A      |
|  2 | BUDI   | 1985-12-10 | 50.00 | T      |
|  3 | CATUR  | 1900-01-01 |  NULL | A      |
|  4 | ENI    | 1900-01-01 |  NULL | A      |
|  5 | FEBRI  | 1990-10-12 | 60.00 | T      |
|  6 | GUNGUN | 2010-10-25 |  NULL | A      |
| 10 | HENDRI | 1900-01-01 |  NULL | A      |
| 11 | INA    | 1900-01-01 |  NULL | A      |
+----+--------+------------+-------+--------+
8 rows in set (0.00 sec)
```

UPDATE

Contoh :

```
update member set berat=berat+1 order by berat desc limit 2;
```

Keterangan : Update 2 baris pertama yang mempunyai berat paling besar(order by berat desc) dengan menambahkan 1 pada kolom berat.

```
+----+--------+------------+-------+--------+
| no | nama   | tgllhr     | berat | status |
+----+--------+------------+-------+--------+
|  1 | ADI    | 1980-06-10 | 66.50 | A      | <- Berubah
|  2 | BUDI   | 1985-12-10 | 50.00 | T      |
|  3 | CATUR  | 1900-01-01 |  NULL | A      |
|  4 | ENI    | 1900-01-01 |  NULL | A      |
|  5 | FEBRI  | 1990-10-12 | 61.00 | T      | <- Berubah
|  6 | GUNGUN | 2010-10-25 |  NULL | A      |
| 10 | HENDRI | 1900-01-01 |  NULL | A      |
| 11 | INA    | 1900-01-01 |  NULL | A      |
+----+--------+------------+-------+--------+
8 rows in set (0.00 sec)
```

REPLACE

- REPLACE berkerja seperti INSERT. Perbedaannya adalah jika terjadi error karena duplicate key, maka data lama akan dihapus dan data baru diinsertkan.
- Bentuk umum penggunaan replaceadalah :

```
REPLACE
  [INTO] nama_tabel[(nama_kolom1[, nama_kolom2, ...])]
  {VALUE | VALUES} (isi_kolom1[, isi_kolom2, ...])
```

- Keterangan :
  - Pemanggilannya mirip dengan insert, begitu juga aturan pengisian datanya.

REPLACE

Contoh :

```
create table demoreplace(id int primary key, nama char(10));
insert into demoreplace values(1,'AA'),(2,'BB'),
                              (3,'CC'),(4,'DD');
select * from demoreplace;
```

```
+----+------+
| id | nama |
+----+------+
|  1 | AA   |
|  2 | BB   |
|  3 | CC   |
|  4 | DD   |
+----+------+
```

REPLACE

Contoh :

```
insert into demoreplace values(1,'Ade'); -- Error Duplicate

replace into demoreplace values(1, 'Ade');
```

Keterangan :

- SQL pertama akan error : “ERROR 1062 (23000): Duplicate entry '1' for key 'PRIMARY'” karena duplikat key.
- SQL kedua akan berhasil dan mengupdate data

```
+----+------+
| id | nama |
+----+------+
|  1 | Ade  |
|  2 | BB   |
|  3 | CC   |
|  4 | DD   |
+----+------+
```

DELETE

Contoh data :

```
select * from member;
```

Menghasilkan :

```
+----+--------+------------+-------+--------+
| no | nama   | tgllhr     | berat | status |
+----+--------+------------+-------+--------+
|  1 | ADI    | 1980-06-10 | 66.50 | A      |
|  2 | BUDI   | 1985-12-10 | 50.00 | T      |
|  3 | CATUR  | 1900-01-01 |  NULL | A      |
|  4 | ENI    | 1900-01-01 |  NULL | A      |
|  5 | FEBRI  | 1990-10-12 | 61.00 | T      |
|  6 | GUNGUN | 2010-10-25 |  NULL | A      |
| 10 | HENDRI | 1900-01-01 |  NULL | A      |
| 11 | INA    | 1900-01-01 |  NULL | A      |
+----+--------+------------+-------+--------+
8 rows in set (0.00 sec)
```

DELETE

Contoh :

```
delete from member where no=6;
```

Penghapusan dilakukan pada data dengan no=6, sehingga data menjadi :

```
+----+--------+------------+-------+--------+
| no | nama   | tgllhr     | berat | status |
+----+--------+------------+-------+--------+
|  1 | ADI    | 1980-06-10 | 66.50 | A      |
|  2 | BUDI   | 1985-12-10 | 50.00 | T      |
|  3 | CATUR  | 1900-01-01 |  NULL | A      |
|  4 | ENI    | 1900-01-01 |  NULL | A      |
|  5 | FEBRI  | 1990-10-12 | 61.00 | T      |
| 10 | HENDRI | 1900-01-01 |  NULL | A      |
| 11 | INA    | 1900-01-01 |  NULL | A      |
+----+--------+------------+-------+--------+
7 rows in set (0.00 sec)
```

DELETE

Contoh :

```
delete from member order by berat desc limit 2;
```

Penghapusan 2 baris (limit 2) yang mempunyai berat paling besar (order by berat desc). Data menjadi :

```
+----+--------+------------+-------+--------+
| no | nama   | tgllhr     | berat | status |
+----+--------+------------+-------+--------+
|  2 | BUDI   | 1985-12-10 | 50.00 | T      |
|  3 | CATUR  | 1900-01-01 |  NULL | A      |
|  4 | ENI    | 1900-01-01 |  NULL | A      |
| 10 | HENDRI | 1900-01-01 |  NULL | A      |
| 11 | INA    | 1900-01-01 |  NULL | A      |
+----+--------+------------+-------+--------+
5 rows in set (0.00 sec)
```

DELETE

Contoh :

```
delete from member;
```

Penghapusan semua baris (tanpa where, tanpa limit). Data menjadi “empty set”.

TRUNCATE

- TRUNCATE berguna untuk melakukan penghapusan semua baris (mirip dengan DELETE tanpa WHERE dan LIMIT).
- Format pemanggilan TRUNCATE adalah :

```
TRUNCATE [TABLE] tbl_name
```

- Perbedaan antara TRUNCATE dan DELETE
  - Jika penghapusan menggunakan DELETE, maka nilai auto_increment tetap berlaku dan akan digunakan untuk insert berikutnya. Tetapi jika menggunakan truncate, maka nilai auto_increment akan kembali ke nilai awal.
  - Penghapusan dengan DELETE dilakukan dengan melakukan penghapusan per baris, tetapi TRUNCATE akan melakukan penghapusan dengan DROP tabel dan kemudian CREATE tabel baru. Hal ini akan menyebabkan eksekusi TRUNCATE akan lebih cepat dibandingkan dengan DELETE.

TRUNCATE

Contoh (penghapusan data dengan delete):

```
Create table demodelete(
       id int primary key auto_increment, nama varchar(5));
Insert into demodelete(nama) values('aa'),('bb'),('cc');
Delete from demodelete;
Insert into demodelete(nama) values('dd');
```

Keterangan :

- SQL kedua akan mengisi 3 data. Nilai auto_increment terakhir adalah 3.
- SQL ketiga akan menghapus semua data.
- SQL keempat akan menambah data dengan nilai auto_increment adalah 4

TRUNCATE

Contoh (penghapusan data dengan truncate):

```
Create table demotruncate(
       id int primary key auto_increment, nama varchar(5));
Insert into demotruncate(nama) values('aa'),('bb'),('cc');
Truncate table demotruncate;
Insert into demotruncate (nama) values('dd');
```

Keterangan :

- SQL kedua akan mengisi 3 data. Nilai auto_increment terakhir adalah 3.
- SQL ketiga akan menghapus semua data.
- SQL keempat akan menambah data dengan nilai auto_increment kembali ke awal yaitu 1.