MODUL 7
TRIGGER

Dasar Teori

Pernyataan CREATE TRIGGER digunakan untuk membuat *trigger*, termasuk aksi apa yang dilakukan saat *trigger* diaktifkan. *Trigger* berisi program yang dihubungkan dengan suatu tabel atau *view* yang secara otomatis melakukan suatu aksi ketika suatu baris di dalam tabel atau *view* dikenai operasi INSERT, UPDATE atau DELETE.

Sintak :

```
CREATE
    [DEFINER = { user | CURRENT_USER }]
    TRIGGER trigger_name trigger_time trigger_event
    ON tbl_name FOR EACH ROW trigger_stmt
```

Keterangan :

- ***[DEFINER = { user | CURRENT_USER }]***: Definisi *user* yang sedang aktif, sifatnya opsional.
- ***trigger_name***: Nama *trigger*.
- ***trigger_time***: waktu menjalankan *trigger*. Ini dapat berupa BEFORE atau AFTER.
  - ✓ BEFORE: Membuat *trigger* diaktifkan sebelum dihubungkan dengan suatu operasi.
  - ✓ AFTER: Membuat *trigger* diaktifkan setelah dihubungkan dengan suatu operasi.
- ***trigger_event***: berupa kejadian yang akan dijalankan *trigger*.
- ***trigger_event*** dapat berupa salah satu dari berikut:
  - ✓ INSERT : *trigger* diaktifkan ketika sebuah *record* baru disisipkan ke dalam tabel. Contoh: statemen INSERT, LOAD DATA, dan REPLACE.
  - ✓ UPDATE : *trigger* diaktifkan ketika sebuah *record* dimodifikasi. Contoh: statemen UPDATE.
  - ✓ DELETE : *trigger* diaktifkan ketika sebuah *record* dihapus. Contoh: statemen DELETE dan REPLACE.

  Catatan : `trigger_event` **tidak** merepresentasikan statemen SQL yang diaktifkan *trigger* sebagai suatu operasi tabel. Sebagai contoh, *trigger* `BEFORE` untuk `INSERT` akan diaktifkan tidak hanya oleh statemen `INSERT` tetapi juga statemen `LOAD DATA`.
- ***tbl_name:*** Nama tabel yang berasosiasi dengan *trigger*.
- ***trigger_stmt*** : Statemen (tunggal atau jamak) yang akan dijalankan ketika *trigger* aktif.

Sekarang kita masuk ke bahasan utama, yaitu implementasi. Untuk menerapkan TRIGGER, PROCEDURE, FUNCTION dan VIEW dibutuhkan suatu relasi, misalkan: *mahasiswa* dan *prodi*, sebagaimana yang diilustrasikan dengan perintah SQL di bawah ini.

B. Kegiatan Praktikum

1. Buatlah database dengan nama akademik.
2. Kemudian buatlah tabel mhs dan prodi dengan struktur sebagai berikut :

```
mysql> desc mhs;
+----------+-------------+------+-----+---------+-------+
| Field    | Type        | Null | Key | Default | Extra |
+----------+-------------+------+-----+---------+-------+
| nim      | char(5)     | NO   | PRI |         |       |
| nama     | varchar(25) | YES  |     | NULL    |       |
| alamat   | varchar(50) | YES  |     | NULL    |       |
| kd_prodi | char(3)     | YES  |     | NULL    |       |
+----------+-------------+------+-----+---------+-------+
4 rows in set (0.02 sec)

mysql> desc prodi;
+-------------+-------------+------+-----+---------+-------+
| Field       | Type        | Null | Key | Default | Extra |
+-------------+-------------+------+-----+---------+-------+
| kd_prodi    | char(3)     | NO   | PRI |         |       |
| nama_prodi  | varchar(25) | YES  |     | NULL    |       |
| jurusan     | varchar(20) | YES  |     | NULL    |       |
+-------------+-------------+------+-----+---------+-------+
3 rows in set (0.00 sec)
```

3. Agar kedua tabel diatas berelasi, buatlah kunci tamu pada tabel mhs yang mereferensi ke tabel prodi (kd_prodi) sebagai berikut :

```
mysql> alter table mhs add foreign key(kd_prodi)
    -> references prodi(kd_prodi);
Query OK, 0 rows affected (0.25 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> desc mhs;
+----------+-------------+------+-----+---------+-------+
| Field    | Type        | Null | Key | Default | Extra |
+----------+-------------+------+-----+---------+-------+
| nim      | char(5)     | NO   | PRI |         |       |
| nama     | varchar(25) | YES  |     | NULL    |       |
| alamat   | varchar(50) | YES  |     | NULL    |       |
| kd_prodi | char(3)     | YES  | MUL | NULL    |       |
+----------+-------------+------+-----+---------+-------+
4 rows in set (0.00 sec)
```

4. Isikan data ke tabel mhs dan prodi sebagai berikut :

```
mysql> select * from prodi;
+----------+--------------------+------------+
| kd_prodi | nama_prodi         | jurusan    |
+----------+--------------------+------------+
| P01      | Eks Ilmu Komputer  | Matematika |
| P02      | Ilmu Komputer      | Matematika |
| P03      | D3 Komsi           | Matematika |
| P04      | D3 Rekmed          | Matematika |
| P05      | D3 Ellins          | Fisika     |
+----------+--------------------+------------+
5 rows in set (0.00 sec)

mysql> select * from mhs;
+-------+---------------+--------------------+----------+
| nim   | nama          | alamat             | kd_prodi |
+-------+---------------+--------------------+----------+
| 00543 | Muhammad      | Karangmalang A-50  | P01      |
| 10041 | Sugiharta     | Karangmalang A-23  | P02      |
| 10043 | Ahmad Sholihun | Karangmalang D-17 | P02      |
+-------+---------------+--------------------+----------+
3 rows in set (0.00 sec)
```

5. Untuk pembuatan TRIGGER, contoh yang akan dibahas adalah mencatat kejadian-kejadian yang terjadi beserta waktunya pada tabel mhs, dan catatan-catatan tadi disimpan dalam tabel yang lain, misal **log_mhs**. Misalkan struktur tabel **log_mhs** adalah sebagai berikut:

```
mysql> create table log_mhs
    -> (kejadian varchar(25),
    -> waktu datetime);
Query OK, 0 rows affected (0.06 sec)

mysql> desc log_mhs;
+----------+-------------+------+-----+---------+-------+
| Field    | Type        | Null | Key | Default | Extra |
+----------+-------------+------+-----+---------+-------+
| kejadian | varchar(25) | YES  |     | NULL    |       |
| waktu    | datetime    | YES  |     | NULL    |       |
+----------+-------------+------+-----+---------+-------+
2 rows in set (0.00 sec)
```

```
mysql> create table log_mhs
    -> (kejadian varchar(25),
    -> waktu datetime);
Query OK, 0 rows affected (0.06 sec)

mysql> desc log_mhs;
+----------+-------------+------+-----+---------+-------+
| Field    | Type        | Null | Key | Default | Extra |
+----------+-------------+------+-----+---------+-------+
| kejadian | varchar(25) | YES  |     | NULL    |       |
| waktu    | datetime    | YES  |     | NULL    |       |
+----------+-------------+------+-----+---------+-------+
2 rows in set (0.00 sec)
```

6. Buatlah TRIGGER untuk mencatat kejadian setelah dilakukan perintah INSERT pada tabel mhs dan disimpan ke dalam tabel log_mhs sebagai berikut:

```
mysql> create trigger ins_mhs after insert on mhs
    -> for each row insert into log_mhs values ('Tambah data',now());
Query OK, 0 rows affected (0.11 sec)

mysql> insert into mhs values
    -> ('00631','Hanif','Kalasan','P01');
Query OK, 1 row affected (0.07 sec)

mysql> select * from mhs;
+-------+----------------+-------------------+----------+
| nim   | nama           | alamat            | kd_prodi |
+-------+----------------+-------------------+----------+
| 00543 | Muhammad       | Karangmalang A-50 | P01      |
| 00631 | Hanif          | Kalasan           | P01      |
| 10041 | Sugiharta      | Karangmalang A-23 | P02      |
| 10043 | Ahmad Sholihun | Karangmalang D-17 | P02      |
+-------+----------------+-------------------+----------+
4 rows in set (0.00 sec)

mysql> select * from log_mhs;
+-------------+---------------------+
| kejadian    | waktu               |
+-------------+---------------------+
| Tambah data | 2016-03-11 09:33:42 |
+-------------+---------------------+
1 row in set (0.00 sec)
```

Dari contoh diatas dapat dilihat bahwa ketika satu record pada tabel ***mhs*** *disisipkan (insert), maka secara otomatis tabel* ***log_mhs*** *akan disisipkan satu record, yaitu kejadian* `'Tambah data'` *dan waktu saat record pada tabel* ***mhs*** *disisipkan*

7. Selanjutnya buatlah TRIGGER untuk mencatat perubahan data yang dilakukan setelah mendapatkan perintah UPDATE pada tabel mhs:

```
mysql> create trigger updt_mhs after update on mhs
    -> for each row insert into log_mhs values ('Ubah Data',now());
Query OK, 0 rows affected (0.07 sec)

mysql> update mhs
    -> set nama='Moh.Riyan' where nim='00543';
Query OK, 1 row affected (0.06 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> select * from mhs;
+-------+---------------+-------------------+----------+
| nim   | nama          | alamat            | kd_prodi |
+-------+---------------+-------------------+----------+
| 00543 | Moh.Riyan     | Karangmalang A-50 | P01      |
| 00631 | Hanif         | Kalasan           | P01      |
| 10041 | Sugiharta     | Karangmalang A-23 | P02      |
| 10043 | Ahmad Sholihun | Karangmalang D-17 | P02      |
+-------+---------------+-------------------+----------+
4 rows in set (0.00 sec)

mysql> select * from log_mhs;
+------------+---------------------+
| kejadian   | waktu               |
+------------+---------------------+
| Tambah data | 2016-03-11 09:33:42 |
| Ubah Data  | 2016-03-11 09:37:31 |
+------------+---------------------+
2 rows in set (0.00 sec)
```

8. Kemudian TRIGGER untuk perintah DML DELETE pada tabel mhs sebagai berikut, dimana proses kejadiannya direkam pada tabel log_mhs

```
mysql> select * from mhs;
+-------+---------------+-------------------+----------+
| nim   | nama          | alamat            | kd_prodi |
+-------+---------------+-------------------+----------+
| 00543 | Moh.Riyan     | Karangmalang A-50 | P01      |
| 10041 | Sugiharta     | Karangmalang A-23 | P02      |
| 10043 | Ahmad Sholihun | Karangmalang D-17 | P02      |
+-------+---------------+-------------------+----------+
3 rows in set (0.00 sec)

mysql> select * from log_mhs;
+------------+---------------------+
| kejadian   | waktu               |
+------------+---------------------+
| Tambah data | 2016-03-11 09:33:42 |
| Ubah Data  | 2016-03-11 09:37:31 |
| Hapus Data | 2016-03-11 09:39:42 |
+------------+---------------------+
3 rows in set (0.00 sec)
```

*Dari contoh diatas dapat dilihat bahwa ketika satu record pada tabel **mhs** dihapus (delete), maka secara otomatis tabel **log_mhs** akan disisipkan satu record, yaitu kejadian 'Hapus data' dan waktu saat record pada tabel **mhs** dihapus.*

9. Tampilkanlah semua TRIGGER yang telah dibuat

```
mysql> show triggers;
+---------+--------+----------+-----------+--------+---------+----------+
| Trigger | Event  | Table    | Statement | Timing | Created | sql_mode |
+---------+--------+----------+-----------+--------+---------+----------+
| ins_mhs | INSERT | mahasiswa |     S1    | AFTER  | NULL    |   SQL1   |
| updt_mhs | UPDATE | mahasiswa |     S2    | AFTER  | NULL    |   SQL2   |
| del_mhs | DELETE | mahasiswa |     S3    | AFTER  | NULL    |   SQL3   |
+---------+--------+----------+-----------+--------+---------+----------+
```

Keterangan (*record* pada kolom "statement" dan "sql_mode"):

```
S1 : insert into log_mhs values('Tambah data',now())
S2 : insert into log_mhs values('Ubah data',now())
S3 : insert into log_mhs values('Hapus data',now())
SQL1 : STRICT_TRANS_TABLES,NO_AUTO_CREATE_USER,NO_ENGINE_SUBSTITUTION
SQL2 : STRICT_TRANS_TABLES,NO_AUTO_CREATE_USER,NO_ENGINE_SUBSTITUTION
SQL3 : STRICT_TRANS_TABLES,NO_AUTO_CREATE_USER,NO_ENGINE_SUBSTITUTION
```

Karena hasil eksekusi perintah "show triggers" sangat panjang, tampilan di atas sengaja diedit dengan tujuan agar mudah dipahami

C. Tugas Praktikum

Implementasikan hasil praktikum di atas terhadap tabel kelompok, kemudian buat laporannya

MODUL 8
BACKUP DAN RESTORE

A. Dasar Teori

Sangat penting untuk mengenal perintah mysqldump, Syntax yang digunakan adalah sebagai berikut.

```
mysqldump -u [username] -p [password] [nama_database] > [nama_file_backup.sql]
```

Kode diatas adalah format penulisan mysqldump, lalu bagaimana contoh penulisannya, pada kesempatan kali ini jurnalweb.com akan mendemokan beberapa jenis cara backup mysql dengan berbagai tujuan.

Contoh cara backup

```
mysqldump -u root -p [password] nama_database> jurnalwebcom.sql.
```

B. Kegiatan Praktikum

Lakukan backup database pada server kalian dengan CLI dan GUI