Rakamin Academy

investree

Virtual Internship Experience

# MySQL: Automation

1. Store Procedure

Stored Procedure adalah sebuah fungsi berisi kode SQL yang dapat digunakan kembali.

Dalam Stored Procedure juga dapat dimasukkan parameter sehingga fungsi dapat digunakan lebih dinamis berdasarkan parameter tersebut. Berikut adalah contoh cara penulisan procedure

```
DELIMITER $$

CREATE PROCEDURE nama_procedure()
BEGIN
    kode sql
END$$

DELIMITER ;
```

Selanjutnya untuk memanggil prosedur nya seperti ini

```
CALL nama_procedure();
```

Kita akan menerapkan contoh procedure dengan table mahasiswa. Pertama kita membuat dulu table mahasiswa untuk contoh querynya seperti di bawah:

```
CREATE TABLE mahasiswa
(
    nim INT(10),
    nama VARCHAR(100),
    alamat VARCHAR(100),
    PRIMARY KEY(nim)
);
```

Dan jangan lupa untuk insert beberapa data,

```
INSERT INTO mahasiswa
VALUES
(21400200,"faqih","bandung"),
(21400201,"ina","jakarta"),
(21400202,"anto","semarang"),
(21400203,"dani","padang"),
(21400204,"jaka","bandung"),
(21400205,"nara","bandung"),
(21400206,"senta","semarang");
```

Kemudian setelah kita berhasil untuk insert data ke table mahasiswa maka langkah selanjutnya adalah membuat procedure yang akan kita panggil nantinya contohnya seperti di bawah ini:

```
DELIMITER $$

CREATE PROCEDURE selectMahasiswa()
BEGIN
    SELECT nim, nama FROM mahasiswa;
END$$

DELIMITER ;
```

Kita membuat procedure yang bernamakan selectMahasiswa() procedure ini akan digunakan untuk melaksanakan query “select nim, nama from

mahasiswa” jadi untuk memanggil procedure yang sudah kita buat menggunakan contoh di bawah ini:

```
CALL selectMahasiswa()
```

Maka akan tampil hasil seperti di bawah ini:

```
> CALL selectMahasiswa();
+----------+-------+
| nim      | nama  |
+----------+-------+
| 21400200 | faqih |
| 21400201 | ina   |
| 21400202 | anto  |
| 21400203 | dani  |
| 21400204 | jaka  |
| 21400205 | nara  |
| 21400206 | senta |
+----------+-------+
7 rows in set (0.00 sec)
```

2. Trigger

TRIGGER adalah kumpulan kode SQL yang berjalan secara otomatis untuk mengeksekusi perintah INSERT, UPDATE, DELETE.

Biasanya TRIGGER akan dijalankan sebelum atau sesudah proses INSERT, UPDATE, DELETE (Perintah DML). Berikut adalah contoh penulisan Trigger :

```
DELIMITER $$
CREATE TRIGGER nama_trigger
{BEFORE | AFTER} {INSERT | UPDATE| DELETE }
    ON nama_table
    FOR EACH ROW
BEGIN
    KODE SQL
END$$
DELIMITER ;
```

Beberapa command trigger ada penjelasannya detail seperti di bawah ini

| Jenis Trigger | Keterangan Trigger |
|---|---|
| BEFORE INSERT | TRIGGER dijalankan sebelum record dimasukkan ke database |
| AFTER INSERT | TRIGGER dijalankan sesudah record dimasukkan ke database |
| BEFORE UPDATE | TRIGGER dijalankan sebelum record dirubah di database |
| AFTER UPDATE | TRIGGER dijalankan sesudah record dirubah database |
| BEFORE DELETE | TRIGGER dijalankan sebelum record dihapus di database |
| AFTER DELETE | TRIGGER dijalankan sesudah record dihapus di database |

Selanjutnya kita akan mencoba membuat contoh trigger case-nya adalah ketika ada yang mengubah alamat mahasiswa kita akan menyimpan log perubahan semua mahasiswa tersebut. Pertama kita buat table mahasiswa berikut contoh querynya

```
CREATE TABLE mahasiswa
(
    nim INT(10),
    nama VARCHAR(100),
    alamat VARCHAR(100),
    PRIMARY KEY(nim)
);
```

Jangan lupa untuk menambah data ke table mahasiswa yang nantinya akan kita update datanya:

```
INSERT INTO mahasiswa
VALUES
(21400200,"faqih","bandung"),
(21400201,"ina","jakarta"),
(21400202,"anto","semarang"),
(21400203,"dani","padang"),
(21400204,"jaka","bandung"),
(21400205,"nara","bandung"),
(21400206,"senta","semarang");
```

Selanjutnya kita akan membuat table log_mahasiswa, berikut contohnya

```
CREATE TABLE log_mahasiswa
(
    id_log INT(10) AUTO_INCREMENT,
    nim INT(10),
    alamat_lama VARCHAR(100),
    alamat_baru VARCHAR(100),
    waktu DATE,
    PRIMARY KEY(id_log)
);
```

Nah setelah kita sudah membuat dua table yang saling berhubungan, selanjutnya kita akan membuat query trigger gimana ketika ada yang update alamat di table mahasiswa akan langsung insert ke table log_mahasiswa

```
DELIMITER $$
CREATE TRIGGER update_alamat_mahasiswa
    BEFORE UPDATE
    ON mahasiswa
    FOR EACH ROW
BEGIN
    INSERT INTO log_mahasiswa
    set nim = OLD.nim,
    alamat_lama=old.alamat,
    alamat_baru=new.alamat,
    waktu = NOW();
END$$
DELIMITER ;
```

Keyword **OLD** digunakan untuk mengambil data kolom di table yang lama sedangkan keyword **NEW** digunakan untuk mengambil data kolom di table yang baru

Sekarang kita akan coba update alamat mahasiswa dengan NIM 21400200. Sebelum diupdate alamat mahasiswa dengan NIM 21400200 adalah “bandung”

Kita ganti alamat “bandung” menjadi “surabaya”, berikut contoh querynya

```
UPDATE mahasiswa
SET alamat = 'surabaya'
WHERE nim = 21400200;
```

Sekarang coba lakukan perintah SELECT untuk melihat **table log_mahasiswa**

```
> SELECT * FROM log_mahasiswa;
+--------+----------+-------------+-------------+------------+
| id_log | nim      | alamat_lama | alamat_baru | waktu      |
+--------+----------+-------------+-------------+------------+
|      1 | 21400200 | bandung     | surabaya    | 2019-11-02 |
+--------+----------+-------------+-------------+------------+
1 row in set (0.00 sec)
```

Oke, record baru secara otomatis telah ditambahkan ke **table log_mahasiswa** untuk mahasiswa dengan NIM 21400200 yang telah diubah alamat awal “bandung” menjadi “surabaya”

Sedangkan pada table mahasiswa alamat yang tercantum adalah alamat yang baru

```
> SELECT * FROM mahasiswa WHERE nim=21400200;
+----------+-------+----------+
| nim      | nama  | alamat   |
+----------+-------+----------+
| 21400200 | faqih | surabaya |
+----------+-------+----------+
1 row in set (0.00 sec)
```

Daftar Pustaka

https://ngodingdata.com/cara-membuat-stored-procedure-di-mysql/
https://ngodingdata.com/cara-membuat-trigger-mysql/