

# **PRAKTIKUM BASIS DATA**

## **PERTEMUAN 11**

“Disusun untuk memenuhi tugas mata kuliah Praktikum Basis Data”

Dosen Pengampu: Ridwan Setiawan, S.T. M.Kom .



Disusun Oleh:

Ilpi Ameliansyah Sugandar (2206053)

**TEKNIK INFORMATIKA**

**INSTITUT TEKNOLOGI GARUT**

**2024**

## GROUPING DATA

### A. Grouping Data

Digunakan bersama-sama dengan fungsi Aggregate untuk menyatukan dua atau lebih grup data kedalam suatu fungsi data tunggal. Fungsi Aggregate digunakan untuk mengambil data tunggal hasil dari perhitungan data yang tersimpan dalam suatu kolom.

- Create Table

Buat table dengan nama table transaksi terlebih dahulu

```
MariaDB [(none)]> use online_shop;
Database changed
MariaDB [online_shop]> create table transaksi(
    -> id_trans varchar(3) not null primary key,
    -> nama_pelanggan varchar(15),
    -> total_harga int(11));
Query OK, 0 rows affected (0.026 sec)
```

- Show Table

Setelah membuat table, tampilkan table transaksi kemudian akan muncul dengan nama transaksi

```
MariaDB [online_shop]> show tables;
+-----+
| Tables_in_online_shop |
+-----+
| angka                  |
| barang                 |
| detail_pesanan         |
| kategori               |
| pelanggan              |
| pesan                  |
| provinsi               |
| test_date              |
| transaksi               |
+-----+
9 rows in set (0.002 sec)

MariaDB [online_shop]> desc transaksi;
+-----+-----+-----+-----+-----+-----+
| Field      | Type      | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| id_trans   | varchar(3) | NO   | PRI | NULL    |       |
| nama_pelanggan | varchar(15) | YES  |     | NULL    |       |
| total_harga | int(11)    | YES  |     | NULL    |       |
+-----+-----+-----+-----+-----+-----+
3 rows in set (0.035 sec)
```

- Tambahkan data pada table transaksi dengan memasukkan id\_transaksi, nama\_pelanggan, dan total-harga.

```
MariaDB [online_shop]> insert into transaksi(id_trans,nama_pelanggan,total_harga) values
-> ('001','Budi','20000'),
-> ('002','Jajang','5000'),
-> ('003','Jajang','10000'),
-> ('004','Udin','25000'),
-> ('005','Budi','10000'),
-> ('006','Budi','14000'),
-> ('007','Asep','30000');
Query OK, 7 rows affected (0.162 sec)
Records: 7 Duplicates: 0 Warnings: 0
```

```
MariaDB [online_shop]> select * from transaksi;
+-----+-----+-----+
| id_trans | nama_pelanggan | total_harga |
+-----+-----+-----+
| 001      | Budi           | 20000      |
| 002      | Jajang         | 5000       |
| 003      | Jajang         | 10000      |
| 004      | Udin           | 25000      |
| 005      | Budi           | 10000      |
| 006      | Budi           | 14000      |
| 007      | Asep           | 30000      |
+-----+-----+-----+
7 rows in set (0.001 sec)
```

## 1. Fungsi Aggregate

hasilnya merupakan data tunggal yang menunjukkan jumlah baris (atau jumlah transaksi) dalam tabel transaksi.

```
MariaDB [online_shop]> select count(nama_pelanggan) as jumlah_baris from transaksi;
+-----+
| jumlah_baris |
+-----+
| 7            |
+-----+
1 row in set (0.050 sec)
```

## 2. GROUP BY

Menampilkan data pelanggan dan berapa banyak transaksi pembelian barang yang telah dilakukannya.

Perintah GROUP BY dalam SQL digunakan untuk mengelompokkan baris yang memiliki nilai yang sama dalam kolom tertentu. Penggunaan GROUP BY biasanya dipasangkan dengan fungsi agregat (seperti COUNT, SUM, AVG, MAX, MIN) untuk melakukan operasi pada setiap grup.

```
MariaDB [online_shop]> select
-> nama_pelanggan,
-> count(nama_pelanggan) 'banyak_pembelian'
-> from transaksi group by nama_pelanggan;
```

nama_pelanggan	banyak_pembelian
Asep	1
Budi	3
Jajang	2
Udin	1

4 rows in set (0.056 sec)

### 3. GROUP BY WITH ORDER BY

GROUP BY dapat ditambah dengan ORDER BY untuk mengurutkan hasil Query dengan syarat :

- ORDER BY tidak dapat digunakan pada Query yang hanya mengandung fungsi Aggregate, yaitu tanpa ada GROUP BY.
- GROUP BY harus ditulis sebelum ORDER BY.

```
MariaDB [online_shop]> select
-> nama_pelanggan,
-> count(nama_pelanggan) 'banyak_pembelian'
-> from transaksi
-> group by nama_pelanggan
-> order by banyak_pembelian desc;
```

nama_pelanggan	banyak_pembelian
Budi	3
Jajang	2
Asep	1
Udin	1

4 rows in set (0.039 sec)

### 4. HAVING

Digunakan dengan GROUP BY untuk mengkondisikan suatu group data hasil perhitungan dari fungsi Aggregate. HAVING mempunyai fungsi dan sintak yang sama dengan WHERE.

```

MariaDB [online_shop]> select
  -> nama_pelanggan,
  -> count(nama_pelanggan) 'banyak_pembelian'
  -> from transaksi
  -> group by nama_pelanggan
  -> having banyak_pembelian >=2
  -> order by banyak_pembelian desc;
+-----+-----+
| nama_pelanggan | banyak_pembelian |
+-----+-----+
| Budi          | 3                |
| Jajang        | 2                |
+-----+-----+
2 rows in set (0.042 sec)

```

## 5. Trigger

Trigger dalam database adalah kode prosedural yang secara otomatis dijalankan untuk menanggapi perubahan tertentu pada table tertentu atau tampilan dalam database. Trigger dapat didefinisikan untuk menjalankan perintah sebelum atau setelah eksekusi DML (Data Manipulation Language) seperti INSERT, UPDATE, dan DELETE. Trigger banyak digunakan untuk menjaga integritas informasi pada database.

### Contoh Penerapan Trigger

Kita akan membuat fitur yang mencatat log perubahan harga barang pada sebuah database penjualan, dimana terdapat tabel produk sebagai tabel untuk menyimpan informasi produk yang memiliki field kode\_produk, nama\_produk dan harga.

```

MariaDB [online_shop]> create table produk (
  -> kode_produk varchar(6) not null primary key,
  -> nama_produk varchar(100) not null,
  -> harga int(11) not null);
Query OK, 0 rows affected (0.082 sec)

```

```

MariaDB [online_shop]> describe produk;
+-----+-----+-----+-----+-----+-----+
| Field      | Type          | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| kode_produk | varchar(6)    | NO   | PRI | NULL    |       |
| nama_produk | varchar(100)  | NO   |     | NULL    |       |
| harga       | int(11)       | NO   |     | NULL    |       |
+-----+-----+-----+-----+-----+-----+
3 rows in set (0.164 sec)

```

lalu kita akan membuat sebuah tabel log\_harga\_produk untuk menyimpan informasi perubahan harga produk, informasi yang akan kita simpan adalah kode\_produk, harga\_lama, harga\_baru dan waktu\_perubahan.

```
MariaDB [online_shop]> create table log_harga_produk (
-> log_id int(11) not null primary key auto_increment,
-> kode_produk varchar(8) not null,
-> harga_lama int(11) not null,
-> harga_baru int(11) not null,
-> waktu_perubahan datetime not null);
Query OK, 0 rows affected (0.053 sec)
```

```
MariaDB [online_shop]> describe log_harga_produk;
+-----+-----+-----+-----+-----+-----+
| Field          | Type          | Null | Key | Default | Extra          |
+-----+-----+-----+-----+-----+-----+
| log_id         | int(11)       | NO   | PRI | NULL    | auto_increment |
| kode_produk    | varchar(8)    | NO   |     | NULL    |                |
| harga_lama     | int(11)       | NO   |     | NULL    |                |
| harga_baru     | int(11)       | NO   |     | NULL    |                |
| waktu_perubahan | datetime      | NO   |     | NULL    |                |
+-----+-----+-----+-----+-----+-----+
5 rows in set (0.081 sec)
```

selanjutnya kita akan membuat sebuah trigger untuk mencatat perubahan harga produk ketika sebuah record produk di update,

```
MariaDB [online_shop]> delimiter $$
MariaDB [online_shop]> create trigger before_produk_update
-> before update on produk
-> for each row
-> begin
-> insert into log_harga_produk
-> set kode_produk = old.kode_produk,
-> harga_baru=new.harga,
-> harga_lama=old.harga,
-> waktu_perubahan=now();
-> end$$
Query OK, 0 rows affected (0.062 sec)
```

Penjelasan Script:

- Line 2 – Kita membuat sebuah trigger baru dengan nama before\_produk\_update
- Line 3 – Pada Trigger ini kita menggunakan event BEFOREUPDATE
- Line 6 – Query SQL untuk melakukan insert data ke table log\_harga\_produk

Setelah semua nya selesai kita akan melakukan uji coba terhadap trigger yang kita buat, kita akan melakukan insert beberapa data yang kita gunakan :

```
MariaDB [online_shop]> insert into produk values ('BR001','SEMIN
GGU JAGO CODEIGNITER','120000');
Query OK, 1 row affected (0.070 sec)

MariaDB [online_shop]> insert into produk values ('BR002','SEMIN
GGU JAGO PHP MYSQL','80000');
Query OK, 1 row affected (0.044 sec)
```

```
MariaDB [online_shop]> select * from produk;
```

kode_produk	nama_produk	harga
BR001	SEMINGGU JAGO CODEIGNITER	120000
BR002	SEMINGGU JAGO PHP MYSQL	80000

```
2 rows in set (0.046 sec)
```

langkah selanjutnya adalah kita akan melakukan update data produk :

```
MariaDB [online_shop]> update produk set harga=9000 where kode_produk='BR001';
Query OK, 1 row affected (0.047 sec)
Rows matched: 1 Changed: 1 Warnings: 0
```

Setelah kita melakukan proses update data lalu muncul sebuah record baru pada tabel log\_harga\_produk tentang informasi perubahan data produk yang sudah di update.

```
MariaDB [online_shop]> select * from produk;
```

kode_produk	nama_produk	harga
BR001	SEMINGGU JAGO CODEIGNITER	9000
BR002	SEMINGGU JAGO PHP MYSQL	80000

```
2 rows in set (0.001 sec)
```

```
MariaDB [online_shop]> select * from log_harga_produk;
```

log_id	kode_produk	harga_lama	harga_baru	waktu_perubahan
1	BR001	120000	9000	2024-05-28 16:05:06

```
1 row in set (0.001 sec)
```

## Show trigger

untuk menampilkan list trigger pada sebuah database bisa menggunakan perintah show triggers;

```
MariaDB [online_shop]> show triggers;
```

Trigger	Event	Table	Statement	Timing	Created	sql_mode	Definer	character_set_client	collation_connection	Database Collation
before_produk_update	UPDATE	produk	begin insert into log_harga_produk set kode_produk = old.kode_produk, harga_baru=new.harga, harga_lama=old.harga, waktu_perubahan=now(); end	BEFORE	2024-05-28 15:59:27.69	NO_ZERO_IN_DATE,NO_ZERO_DATE,NO_ENGINE_SUBSTITUTION	root@localhost	utf8mb4	utf8mb4_general_ci	utf8mb4_general_ci

```
1 row in set (0.053 sec)
MariaDB [online_shop]>
```

## Menghapus Trigger

Untuk menghapus trigger gunakan perintah drop trigger nama\_trigger;

Contoh:

```
MariaDB [online_shop]> drop trigger before_produk_update;
Query OK, 0 rows affected (0.007 sec)
```