BAHASA QUERY

MODUL PRATIKUM

Dedy Arisandi, ST, M.Kom

Indra Aulia

FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
PROGRAM STUDI TEKNOLOGI INFORMASI
UNIVERSITAS SUMATERA UTARA

DAFTAR ISI

BAB 1
DASAR-DASAR MySQL

1.1 Pendahuluan

Bab ini membahas operasi dasar dengan MySQL dimulai dari cara membuat database hingga menghapus table.

SQL (*Structured Query Language*) pertama kali difenisikan oleh ISO (*International Standards Organization*) dan ANSI (*the American National Standards Institute*) yang dikenal sebagai SQL86. MySQL sebagai database server juga mendukung perintah SQL. Perintah yang dapat dipahami oleh database server MySQL disebut dengan pernyataan. Pernyataan adalah sebuah perintah yang dapat dijalankan oleh MySQL dengan ciri selalu diakhiri dengan tanda titik-koma (;). Prinsip kerja yang terjadi adalah ketika pernyataan diakhiri dengan tanda titik-koma kemudian tekan enter, maka program klien MySQL akan segera mengirimkannya ke database server MySQL dan MySQL akan memberikan respon atasnya.

1.2 Mengaktifkan MySQL

Sebelum mengeksekusi perintah SQL maka kita perlu mengaktifkan MySQL engine dan MySQL dengan tahap-tahap berikut ini :

- Mengaktifkan MySQL Engine
  - ∞ Buka sistem operasi dos dengan klik start, run, ketikkan cmd lalu ok
  - ∞ Setelah masuk pada command prompt ketikkan perintah untuk membuka direktori xampp

    cd c:\xampp ↵

    c:\xampp>xampp_start.exe ↵

    Starting "xampp"...

    Jika berhasil mengaktifkan MySQL Engine akan tampil pesan sebagai berikut :

    "xampp" is started.
- Mengaktifkan MySQL
  - ∞ Setelah masuk ke direktori xampp\mysql\bin
  - ∞ Aktifkan MySQL dengan mengetik perintah di bawah ini pada command prompt

    Mysql –u root –p ↵

    Enter Password : ↵

    Artinya masuk ke MySQL dengan –u (username) root dan –p (password) jika ada. Proses akan berhasil jika telah tampil pesan di layar sebagai berikut :

    Welcome to the mysql monitor. Command end with ; or \g

    Your mysql connection id is 1

```
Server version : 5.1.41 source distribution
Type 'help;' or '\h' for help. Type '\c' to clear current input statement.
Mysql>
```

1.3 Menonaktifkan MySQL

Untuk menonaktifkan MySQL maka terlebih dahulu kita harus keluar dari `mysql>` dengan mengetikkan perintah :

```
exit ↵
bye
```

Setelah itu kemudian masuklah ke direktori xampp dengan mengetikkan perintah :

```
cd c:\xampp
c:\xampp>xampp_stop.exe
Stopping "xampp"...
```

Jika berhasil menonaktifkan MySQL maka akan tampil pesan sebagai berikut :

```
"xampp" is stopped.
```

BAB 2
PERINTAH DASAR

2.1 Membuat Database

Untuk membuat database, pergunakan kata-kunci CREATE DATABASE. Penulisannya adalah sebagai berikut :

```
CREATE DATABASE nama_database ;
```

Contoh penggunaan ini misalnya kita ingin membuat database dengan nama DBPenjualan, maka penulisannya adalah sebagai berikut :

```
mysql> CREATE DATABASE DBPenjualan;
Query OK, 1 row affected (0.03 sec)
```

Untuk melihat database yang baru anda buat pergunakan perintah sebagai berikut :

```
mysql> SHOW DATABASES;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| dbpenjualan        |
| dbtoko             |
| mahasiswa          |
| my_db              |
| phpmyadmin         |
| sql                |
+--------------------+
7 rows in set (0.22 sec)
```

2.2 Membatalkan Perintah

Ketik "\c" (singkatan dari cancel) diakhir perintah. Contoh penggunaan :

```
mysql> SHOW
    -> DATABASES
    -> \c
```

2.3 Memilih Database

Untuk menggunakan database yang telah dibuat, maka terlebih dahulu harus memilih database. Memilih database dapat menggunakan sintaks USE dengan kaidah umum sebagai berikut :

```
USE nama_database;
```

Contoh penggunaan tersebut seperti berikut :

```
mysql> USE dbpenjualan
Database changed
```

2.4 Menciptakan Table

Setelah kita membuat database dan memilih database tersebut, langkah selanjutnya adalah pembuatan table. Dalam hal ini pernyataan yang dipergunakan adalah CREATE TABLE dengan kaidah penulisannya adalah :

```
CREATE TABLE nama_table
(
nama_field_1 tipe_data(ukuran),
nama_field_2 tipe_data(ukuran),
…,
nama_field_n tipe_data(ukuran)
);
```

Contoh penggunaan

```
mysql> CREATE TABLE Barang
    -> (
    -> kodebrg varchar(10) PRIMARY KEY,
    -> namabrg varchar(35),
    -> satuan varchar(30),
    -> harga integer
    -> );
Query OK, 0 rows affected (0.09 sec)
```

2.5 Menampilkan Tabel-Tabel

Untuk menampilkan table-tabel yang ada pada database, dipergunakan sintaks SHOW TABLES. Contohnya :

```
mysql> SHOW TABLES;
+------------------------+
| Tables_in_dbpenjualan  |
+------------------------+
| barang                 |
+------------------------+
1 row in set (0.06 sec)
```

2.6 Menampilkan Struktur Tabel

Menampilkan struktur pada sebuah table memiliki kaidah penulisan sebagai berikut :

```
SHOW COLUMNS FROM nama_tabel;
```

atau

```
DESC nama_tabel;
```

Contoh :

```
mysql> DESC barang;
+----------+-------------+------+-----+---------+-------+
| Field    | Type        | Null | Key | Default | Extra |
+----------+-------------+------+-----+---------+-------+
| kodebrg  | varchar(10) | NO   | PRI | NULL    |       |
| namabrg  | varchar(35) | YES  |     | NULL    |       |
| satuan   | varchar(30) | YES  |     | NULL    |       |
| harga    | int(11)     | YES  |     | NULL    |       |
+----------+-------------+------+-----+---------+-------+
4 rows in set (0.02 sec)

mysql> SHOW COLUMNS FROM barang;
+----------+-------------+------+-----+---------+-------+
| Field    | Type        | Null | Key | Default | Extra |
+----------+-------------+------+-----+---------+-------+
| kodebrg  | varchar(10) | NO   | PRI | NULL    |       |
| namabrg  | varchar(35) | YES  |     | NULL    |       |
| satuan   | varchar(30) | YES  |     | NULL    |       |
| harga    | int(11)     | YES  |     | NULL    |       |
+----------+-------------+------+-----+---------+-------+
4 rows in set (0.01 sec)
```

2.7 Mengisi Data Pada Tabel

Untuk mengisi data dalam table yang sudah dibuat, pergunakanlah pernyataan INSERT. Ada beberapa cara untuk mengisi data dalam table sebagai berikut :

a. Mengisi data satu baris

Untuk memasukkan data satu baris dapat ditulis dengan kaidah sebagai berikut :

```
INSERT INTO nama_table VALUES (nilai_1, nilai_2, nilai_3,…,nilai_n) ;
```

Atau dengan menuliskan urutan nama_kolom setelah nama table seperti berikut ini :

```
INSERT INTO nama_table (kolom_1, kolom_2, kolom3,…,kolom_n) VALUES
(nilai_1, nilai_2, nilai_3,…,nilai_n) ;
```

Contoh :

```
mysql> INSERT INTO Barang
    -> VALUES ('BR001','Sabun LUX','Buah',3000);
Query OK, 1 row affected (0.03 sec)
```

Atau :

```
mysql> INSERT INTO Barang (kodebrg,namabrg,satuan,harga)
    -> VALUES ('BR001','Sabun LUX','Buah',3000);
Query OK, 1 row affected (0.03 sec)
```

b. Mengisi data sejumlah baris

Adakalanya kita memasukkan data sejumlah baris dalam satu waktu. Maka kaidah penulisan INSERT yang dapat ditulis adalah sebagai berikut :

INSERT INTO *nama_table*

VALUES (*nilai_1, nilai_2, nilai_3,…,nilai_n*),

(*nilai_1, nilai_2, nilai_3,…,nilai_n*),

(*nilai_1, nilai_2, nilai_3,…,nilai_n*),

(*nilai_1, nilai_2, nilai_3,…,nilai_n*) ;

Contoh :

```
mysql> INSERT INTO Barang (kodebrg,namabrg,satuan,harga)
    -> VALUES ('BR001','Sabun LUX','Buah',3000),
    -> ('BR002','Pepsodent','Buah',3000),
    -> ('BR003','Sabun Cuci','Plastik',3000000);
Query OK, 3 row affected (0.04 sec)
```

c. Mengisi data bernilai NULL

Secara eksplisit nilai NULL dapat diberikan kepada suatu kolom dalam table dengan menggunakan pernyataan INSERT. Sebagaimana diketahui nilai NULL menyatakan bahwa kolom tersebut belum diisikan. Kaidah penulisannya adalah sebagai berikut :

INSERT INTO *nama_table* **VALUES** (*nilai_1, NULL, nilai_3,…,nilai_n*);

Pada contoh kaidah di atas, nilai_2 diisikan dengan NULL.

Contoh :

```
mysql> INSERT INTO Barang (kodebrg,namabrg,satuan,harga)
    -> VALUES ('BR001','Sabun LUX','NULL',3000);
Query OK, 1 row affected (0.03 sec)
```

d. Mengisi Data Secara Masal Pada Tabel Lain

Anda dapat melakukan pengisian data pada tabel secara masal. Ketik perintah berikut ini untuk membuat tabel baru :

```
mysql> Create Table CopyBarang
    -> (Kode char(5) Primary Key,
    -> Namabrg varchar(20) not null,
    -> Satuan varchar(10),
    -> Harga int(5));
Query OK, 0 rows affected (0.00 sec)
```

Ketik perintah berikut ini untuk mengcopy semua data pada tabel barang ke tabel copybarang :

```
mysql> Insert into CopyBarang
    -> select * FROM barang;
Query OK, 13 rows affected (0.00 sec)
Records: 13  Duplicates: 0  Warnings: 0
```

Ketik Perintah berikut ini untuk menampilkan hasilnya :

```
mysql> select * FROM copybarang;
+-------+-------------------+---------+-------+
| Kode  | Namabrg           | Satuan  | Harga |
+-------+-------------------+---------+-------+
| BR001 | Sabun LUX         | Buah    |  3000 |
| BR002 | Pepsodent 25 g    | Buah    |  1000 |
| BR003 | Sabun Cuci        | Plastik |  3000 |
| BR004 | Pengharum Ruangan | Kaleng  | 10000 |
| BR005 | Obat Nyamuk       | Bungkus |  3000 |
| BR006 | Lilin             | Bungkus |  4000 |
| BR007 | Korek Api         | Bungkus |   500 |
| BR008 | Penyedap Rasa     | Bungkus |  1000 |
| BR009 | Ikan Kaleng       | Kaleng  |  4500 |
| BR010 | Coca Cola         | Botol   |  2500 |
| BR011 | Sprite            | Botol   |  2000 |
| BR012 | Fanta             | Botol   |  NULL |
| BR013 | Teh Sosro         | Botol   |  1500 |
+-------+-------------------+---------+-------+
13 rows in set (0.00 sec)
```

2.8 Menampilkan Data dari Tabel

Untuk menampilkan data dalam suatu table, pernyataan yang dapat kita pergunakan adalah **SELECT**. Ada beberapa cara untuk menampilkan data yang ada pada suatu table yakni :

a. Menampilkan Semua Kolom dan Semua Baris

Untuk menampilkan semua data yang ada pada table maka kaidah penulisan sebagai berikut :

```
SELECT * FROM nama_table ;
```

Contoh :

```
mysql> select * FROM barang;
```

```
+---------+-------------------+---------+-------+
| Kodebrg | Nama_Barang       | Satuan  | Harga |
+---------+-------------------+---------+-------+
| BR001   | Sabun LUX         | Buah    |  3000 |
| BR002   | Pepsodent 25 g    | Buah    |  1000 |
| BR003   | Sabun Cuci        | Plastik |  3000 |
| BR004   | Pengharum Ruangan | Kaleng  | 10000 |
| BR005   | Obat Nyamuk       | Bungkus |  3000 |
| BR006   | Lilin             | Bungkus |  4000 |
| BR007   | Korek Api         | Bungkus |   500 |
| BR008   | Penyedap Rasa     | Bungkus |  1000 |
| BR009   | Ikan Kaleng       | Kaleng  |  4500 |
| BR010   | Coca Cola         | Botol   |  2500 |
| BR011   | Sprite            | Botol   |  2000 |
| BR012   | Fanta             | Botol   |  NULL |
| BR013   | Teh Sosro         | Botol   |  1500 |
+---------+-------------------+---------+-------+
13 rows in set (0.00 sec)
```

Contoh di atas identik dengan jika kita menuliskan pernyataannya menjadi sebagai berikut :

```
mysql> select kodebrg, namabrg, satuan, harga FROM barang;
```

Namun, tentu saja penggunaan tanda * lebih praktis daripada kalau menyebutkan semua nama kolom karena tanda * sudah menyatakan semua kolom dan semua baris.

b. Menampilkan Kolom tertentu

Adakalanya kita menghendaki untuk menampilkan hanya kolom-kolom tertentu. Untuk keperluan ini, sebutkan nama-nama kolom sesudah kata SELECT. Contoh :

```
mysql> select kodebrg, namabrg FROM barang;
```

```
+---------+-------------------+
| Kodebrg | Nama_Barang       |
+---------+-------------------+
| BR001   | Sabun LUX         |
| BR002   | Pepsodent 25 g    |
| BR003   | Sabun Cuci        |
| BR004   | Pengharum Ruangan |
| BR005   | Obat Nyamuk       |
| BR006   | Lilin             |
| BR007   | Korek Api         |
| BR008   | Penyedap Rasa     |
| BR009   | Ikan Kaleng       |
| BR010   | Coca Cola         |
| BR011   | Sprite            |
| BR012   | Fanta             |
| BR013   | Teh Sosro         |
+---------+-------------------+
13 rows in set (0.00 sec)
```

BAB 3

MANIPULASI DATA

5.1. Mengubah Struktur Tabel

Adakalanya suatu ketika diperlukan untuk mengubah struktur table. Untuk keperluan ini, maka pergunakan pernyataan ALTER TABLE. Pernyataan ini digunakan untuk menambah, menghapus, atau memodifikasi kolom yang ada pada suatu table.

Sintak : **ALTER TABLE** nama_tabel **JENIS_PENGUBAHAN**;

Jenis Pengubahan yang dapat dilakukan yaitu :

- ADD berfungsi untuk menambah field
- CHANGE berfungsi untuk mengubah struktur field
- DROP berfungsi untuk menghapus field

Contoh :

- Perintah berikut ini berfungsi untuk **menambah** sebuah field baru ke dalam tabel Barang.

```
mysql> ALTER TABLE Barang ADD Stok_Awal Integer(5);
Query OK, 10 rows affected (0.06 sec)
Records: 10  Duplicates: 0  Warnings: 0
```

Ketik perintah berikut ini untuk melihat struktur tabel saat ini.

```
mysql> DESC Barang;
+-----------+-------------+------+-----+---------+-------+
| Field     | Type        | Null | Key | Default | Extra |
+-----------+-------------+------+-----+---------+-------+
| Kodebrg   | varchar(5)  |      | PRI |         |       |
| Namabrg   | varchar(20) |      |     |         |       |
| Satuan    | varchar(10) | YES  |     | NULL    |       |
| Harga     | int(5)      | YES  |     | NULL    |       |
| Stok_Awal | int(5)      | YES  |     | NULL    |       |
+-----------+-------------+------+-----+---------+-------+
5 rows in set (0.02 sec)
```

Ketik perintah berikut ini untuk melihat isi tabel saat ini.

```
mysql> Select * FROM Barang;
+---------+------------------+---------+-------+-----------+
| Kodebrg | Namabrg          | Satuan  | Harga | Stok_Awal |
+---------+------------------+---------+-------+-----------+
| BR001   | Sabun LUX        | Buah    |  3000 |      NULL |
| BR002   | Pasta Gigi       | Buah    |  5000 |      NULL |
| BR003   | Sabun Cuci       | Bungkus |  3000 |      NULL |
| BR004   | Pengharum Ruangan| Kaleng  | 10000 |      NULL |
| BR005   | Obat Nyamuk      | Bungkus |  3000 |      NULL |
| BR006   | Lilin            | Bungkus |  4000 |      NULL |
| BR007   | Korek Api        | Bungkus |   500 |      NULL |
| BR008   | Penyedap Rasa    | Bungkus |  1000 |      NULL |
| BR009   | Ikan Kaleng      | Kaleng  |  4500 |      NULL |
| BR010   | Minuman Botol    | Botol   |  2500 |      NULL |
+---------+------------------+---------+-------+-----------+
10 rows in set (0.00 sec)
```

- Perintah berikut ini berfungsi untuk **mengubah** struktur field Stok_Awal dalam tabel Barang menjadi Stok.

```
mysql> ALTER TABLE Barang CHANGE Stok_awal Stok INTEGER(5);
Query OK, 10 rows affected (0.03 sec)
Records: 10  Duplicates: 0  Warnings: 0
```

Ketik perintah berikut ini untuk melihat struktur tabel saat ini.

```
mysql> DESC Barang;
+---------+-------------+------+-----+---------+-------+
| Field   | Type        | Null | Key | Default | Extra |
+---------+-------------+------+-----+---------+-------+
| Kodebrg | varchar(5)  |      | PRI |         |       |
| Namabrg | varchar(20) |      |     |         |       |
| Satuan  | varchar(10) | YES  |     | NULL    |       |
| Harga   | int(5)      | YES  |     | NULL    |       |
| Stok    | int(5)      | YES  |     | NULL    |       |
+---------+-------------+------+-----+---------+-------+
5 rows in set (0.00 sec)
```

- Perintah berikut ini berfungsi untuk **menghapus** field Stok dalam tabel Barang..

```
mysql> ALTER TABLE Barang DROP STOK;
Query OK, 10 rows affected (0.03 sec)
Records: 10  Duplicates: 0  Warnings: 0
```

Ketik perintah berikut ini untuk melihat struktur tabel saat ini.

```
mysql> DESC Barang;
+---------+-------------+------+-----+---------+-------+
| Field   | Type        | Null | Key | Default | Extra |
+---------+-------------+------+-----+---------+-------+
| Kodebrg | varchar(5)  |      | PRI |         |       |
| Namabrg | varchar(20) |      |     |         |       |
| Satuan  | varchar(10) | YES  |     | NULL    |       |
| Harga   | int(5)      | YES  |     | NULL    |       |
+---------+-------------+------+-----+---------+-------+
4 rows in set (0.01 sec)
```

5.2. Menghapus Tabel

Apabila kita ingin menghapus suatu table karena table tersebut sudah tidak diperlukan lagi, kita dapat menggunakan pernyataan DROP TABLE. Kaidah Penulisan adalah sebagai berikut :

Sintak : **DROP TABLE** nama_tabel;

Sebelum perintah menghapus tabel ini kita coba terlebih dahulu buatlah sebuah tabel dengan nama “coba” seperti perintah berikut ini :

```
mysql> CREATE TABLE Coba
    -> (Field1 CHAR(5),
```

```
    -> Field2 CHAR(10),
    -> Field3 INTEGER(6));
Query OK, 0 rows affected (0.02 sec)
```

Ketik perintah berikut ini untuk melihat daftar tabel pada database DBPenjualan saat ini.

```
mysql> SHOW TABLES;
+----------------------+
| Tables_in_DBPenjualan |
+----------------------+
| barang               |
| coba                 |
+----------------------+
2 rows in set (0.02 sec)
```

Untuk menghapus tabel Coba ketik perintah berikut ini :

```
mysql> DROP TABLE Coba;
Query OK, 0 rows affected (0.00 sec)
```

Ketik perintah berikut ini untuk melihat daftar tabel pada database DBPenjualan saat ini setelah terjadi penghapusan.

```
mysql> SHOW TABLES;
+----------------------+
| Tables_in_DBPenjualan |
+----------------------+
| barang               |
+----------------------+
```

5.3. Mengubah Data Dalam Tabel

Seandainya data yang tersimpan dalam table terdapat yang salah, data tersebut dapat dibenarkan dengan menggunakan pernyataan UPDATE. Penulisan kaidah UPDATE dapat ditulis dengan cara :

Sintak : UPDATE *nama_table*

SET *kolom_1 = nilai_baru_1,*

Kolom_2 = nilai_baru_2,

…,

Kolom_n = nilai_baru_n

[**WHERE** *kondisi*]

Contoh :

Tempilkan terlebih dahulu seluruh data pada tabel Barang dengan perintah :

```
mysql> SELECT * FROM BARANG;
```

```
+---------+-------------------+---------+-------+
| Kodebrg | Namabrg           | Satuan  | Harga |
+---------+-------------------+---------+-------+
| BR001   | Sabun LUX         | Buah    |  3000 |
| BR002   | Pasta Gigi        | Buah    |  5000 |
| BR003   | Sabun Cuci        | Bungkus |  3000 |
| BR004   | Pengharum Ruangan | Kaleng  | 10000 |
| BR005   | Obat Nyamuk       | Bungkus |  3000 |
| BR006   | Lilin             | Bungkus |  4000 |
| BR007   | Korek Api         | Bungkus |   500 |
| BR008   | Penyedap Rasa     | Bungkus |  1000 |
| BR009   | Ikan Kaleng       | Kaleng  |  4500 |
| BR010   | Minuman Botol     | Botol   |  2500 |
+---------+-------------------+---------+-------+
10 rows in set (0.01 sec)
```

Ketik perintah berikut ini untuk mengubah Nama barang “Minuman Botol” menjadi “Coca Cola”.

```
mysql> UPDATE Barang SET Namabrg='Coca Cola'
    -> WHERE Kodebrg='BR010'
    -> ;
Query OK, 1 row affected (0.02 sec)
Rows matched: 1  Changed: 1  Warnings: 0
```

Lihat hasil perubahan dengan perintah :

```
mysql> SELECT * FROM BARANG;
```

```
+---------+-------------------+---------+-------+
| Kodebrg | Namabrg           | Satuan  | Harga |
+---------+-------------------+---------+-------+
| BR001   | Sabun LUX         | Buah    |  3000 |
| BR002   | Pasta Gigi        | Buah    |  5000 |
| BR003   | Sabun Cuci        | Bungkus |  3000 |
| BR004   | Pengharum Ruangan | Kaleng  | 10000 |
| BR005   | Obat Nyamuk       | Bungkus |  3000 |
| BR006   | Lilin             | Bungkus |  4000 |
| BR007   | Korek Api         | Bungkus |   500 |
| BR008   | Penyedap Rasa     | Bungkus |  1000 |
| BR009   | Ikan Kaleng       | Kaleng  |  4500 |
| BR010   | Coca Cola         | Botol   |  2500 |
+---------+-------------------+---------+-------+
```

5.4. Menghapus Data

Seandainya suatu data yang ada pada table tidak dipergunakan lagi, kita bisa menghapusnya dengan menggunakan pernyataan DELETE. Kaidah penulisan ini adalah :

Sintak : DELETE FROM NamaTabel

[WHERE Kondisi**];**

Ketiklah perintah-perintah berikut untuk melihat hasil dari penggunaan perintah Delete :

```
mysql> Delete FROM CopyBarang
    -> WHERE Kode='BR010';
Query OK, 1 row affected (0.00 sec)
```

Perintah di atas berfungsi untuk menghapus baris dengan kode BR010, lihat hasilnya dengan mengetikkan perintah berikut:

```
mysql> select * FROM copybarang;
```

Kode	Namabrg	Satuan	Harga
BR001	Sabun LUX	Buah	3000
BR002	Pepsodent 25 g	Buah	1000
BR003	Sabun Cuci	Plastik	3000
BR004	Pengharum Ruangan	Kaleng	10000
BR005	Obat Nyamuk	Bungkus	3000
BR006	Lilin	Bungkus	4000
BR007	Korek Api	Bungkus	500
BR008	Penyedap Rasa	Bungkus	1000
BR009	Ikan Kaleng	Kaleng	4500
BR011	Sprite	Botol	2000
BR012	Fanta Botol Kecil	Botol	2500
BR013	Teh Sosro	Botol	1500

12 rows in set (0.00 sec)

Jika anda ingin menghapus data copybarang dengan satuan botol maka anda dapat mengetikkan perinth berikut ini :

```
mysql> Delete FROM CopyBarang
    -> WHERE Satuan='Botol';
Query OK, 3 rows affected (0.02 sec)
```

```
mysql> select * FROM copybarang;
```

Kode	Namabrg	Satuan	Harga
BR001	Sabun LUX	Buah	3000
BR002	Pepsodent 25 g	Buah	1000
BR003	Sabun Cuci	Plastik	3000
BR004	Pengharum Ruangan	Kaleng	10000
BR005	Obat Nyamuk	Bungkus	3000
BR006	Lilin	Bungkus	4000
BR007	Korek Api	Bungkus	500
BR008	Penyedap Rasa	Bungkus	1000
BR009	Ikan Kaleng	Kaleng	4500

9 rows in set (0.00 sec)

Jika anda bermaksud menghapus semua data pada tabel, anda dapat mengetik perintah berikut ini:

```
mysql> Delete FROM CopyBarang;
```

Hati-hati menggunakan perintah Delete. Jika anda silap sedikit saja maka seluruh data pada tabel akan terhapus dan tidak bisa dikembalikan lagi.

BAB 4
PENGGUNAAN OPERATOR

5.5. Operator Relasi

Digunakan untuk membandingkan suatu nilai dengan nilai lainnya. Operator relasi yang dapat digunakan yaitu:

Operator	Keterangan
=	Sama dengan
>	Lebih besar
<	Lebih kecil
>=	Lebih besar atau sama dengan
<=	Lebih kecil atau sama dengan
<>	Tidak sama dengan

Ketiklah contoh penggunaan masing-masing opertor relasi berikut ini dan lihat serta pahami hasilnya:

```
mysql> select * FROM barang WHERE kodebrg='BR005';
+---------+-------------+---------+-------+
| Kodebrg | Namabrg     | Satuan  | Harga |
+---------+-------------+---------+-------+
| BR005   | Obat Nyamuk | Bungkus |  3000 |
+---------+-------------+---------+-------+
1 row in set (0.02 sec)
```

```
mysql> select * FROM barang WHERE harga > 4000;
+---------+------------------+--------+-------+
| Kodebrg | Namabrg          | Satuan | Harga |
+---------+------------------+--------+-------+
| BR002   | Pasta Gigi       | Buah   |  5000 |
| BR004   | Pengharum Ruangan | Kaleng | 10000 |
| BR009   | Ikan Kaleng      | Kaleng |  4500 |
+---------+------------------+--------+-------+
3 rows in set (0.00 sec)
```

```
mysql> select * FROM barang WHERE harga < 4000;
+---------+--------------+---------+-------+
| Kodebrg | Namabrg      | Satuan  | Harga |
+---------+--------------+---------+-------+
| BR001   | Sabun LUX    | Buah    |  3000 |
| BR003   | Sabun Cuci   | Bungkus |  3000 |
| BR005   | Obat Nyamuk  | Bungkus |  3000 |
| BR007   | Korek Api    | Bungkus |   500 |
| BR008   | Penyedap Rasa | Bungkus |  1000 |
| BR010   | Coca Cola    | Botol   |  2500 |
+---------+--------------+---------+-------+
6 rows in set (0.00 sec)
```

```
mysql> select * FROM barang WHERE harga >= 4000;
+---------+-------------------+---------+-------+
| Kodebrg | Namabrg           | Satuan  | Harga |
+---------+-------------------+---------+-------+
| BR002   | Pasta Gigi        | Buah    |  5000 |
| BR004   | Pengharum Ruangan | Kaleng  | 10000 |
| BR006   | Lilin             | Bungkus |  4000 |
| BR009   | Ikan Kaleng       | Kaleng  |  4500 |
+---------+-------------------+---------+-------+
4 rows in set (0.00 sec)
```

```
mysql> select * FROM barang WHERE harga <= 4000;
+---------+---------------+---------+-------+
| Kodebrg | Namabrg       | Satuan  | Harga |
+---------+---------------+---------+-------+
| BR001   | Sabun LUX     | Buah    |  3000 |
| BR003   | Sabun Cuci    | Bungkus |  3000 |
| BR005   | Obat Nyamuk   | Bungkus |  3000 |
| BR006   | Lilin         | Bungkus |  4000 |
| BR007   | Korek Api     | Bungkus |   500 |
| BR008   | Penyedap Rasa | Bungkus |  1000 |
| BR010   | Coca Cola     | Botol   |  2500 |
+---------+---------------+---------+-------+
7 rows in set (0.00 sec)
```

```
mysql> select * FROM barang WHERE satuan <> 'Bungkus';
+---------+-------------------+--------+-------+
| Kodebrg | Namabrg           | Satuan | Harga |
+---------+-------------------+--------+-------+
| BR001   | Sabun LUX         | Buah   |  3000 |
| BR002   | Pasta Gigi        | Buah   |  5000 |
| BR004   | Pengharum Ruangan | Kaleng | 10000 |
| BR009   | Ikan Kaleng       | Kaleng |  4500 |
| BR010   | Coca Cola         | Botol  |  2500 |
+---------+-------------------+--------+-------+
5 rows in set (0.00 sec)
```

5.6. Operator Logika

Digunakan untuk operasi logika yaitu :

Operator	Fungsi	Keterangan
NOT atau !	Sebagai negasi atau kebalikan	NOT nikah atau !nikah (artinya adalah menikah)
OR atau ||	Atau	Unit OR Bungkus Medan || Padang
AND atau &&	Dan	Buah AND 5000 Sabun && Botol

Ketiklah contoh penggunaan masing-masing opertor logika berikut ini dan lihat serta pahami hasilnya:

```
mysql> select * FROM barang WHERE satuan='buah' or harga=3000;
+---------+----------------+---------+-------+
| Kodebrg | Nama_Barang    | Satuan  | Harga |
+---------+----------------+---------+-------+
| BR001   | Sabun LUX      | Buah    |  3000 |
| BR002   | Pepsodent 25 g | Buah    |  1000 |
| BR003   | Sabun Cuci     | Plastik |  3000 |
| BR005   | Obat Nyamuk    | Bungkus |  3000 |
+---------+----------------+---------+-------+
4 rows in set (0.03 sec)
```

```
mysql> select * FROM barang WHERE satuan='buah' and harga=3000;
+---------+-------------+--------+-------+
| Kodebrg | Nama_Barang | Satuan | Harga |
+---------+-------------+--------+-------+
| BR001   | Sabun LUX   | Buah   |  3000 |
+---------+-------------+--------+-------+
1 row in set (0.00 sec)
```

```
mysql> select * FROM barang WHERE satuan != 'buah';
+---------+------------------+---------+-------+
| Kodebrg | Nama_Barang      | Satuan  | Harga |
+---------+------------------+---------+-------+
| BR003   | Sabun Cuci       | Plastik |  3000 |
| BR004   | Pengharum Ruangan | Kaleng  | 10000 |
| BR005   | Obat Nyamuk      | Bungkus |  3000 |
| BR006   | Lilin            | Bungkus |  4000 |
| BR007   | Korek Api        | Bungkus |   500 |
| BR008   | Penyedap Rasa    | Bungkus |  1000 |
| BR009   | Ikan Kaleng      | Kaleng  |  4500 |
| BR010   | Coca Cola        | Botol   |  2500 |
| BR011   | Sprite           | Botol   |  2000 |
| BR012   | Fanta            | Botol   |  NULL |
| BR013   | Teh Sosro        | Botol   |  1500 |
+---------+------------------+---------+-------+
11 rows in set (0.00 sec)
```

5.7. Operator Pembanding

Operator	Keterangan
BETWEEN	Apakah suatu nilai diantara dua batasan nilai?
IN	Apakah suatu nilai berada di dalam pilihan yang ada?
NOT IN	Apakah suatu nilai tidak berada di dalam pilihan yang ada?
IS NULL	Apakah sebuah nilai adalah NULL?
IS NOT NULL	Apakah sebuah nilai adalah tidak NULL?
LIKE	Apakah suatu nilai sesuai dengan kriteria tertentu?
NOT LIKE	Apakah suatu nilai tidak sesuai dengan kriteria tertentu?

Ketiklah contoh penggunaan masing-masing opertor pembanding berikut ini dan lihat serta pahami hasilnya:

Menampilkan data barang dimana harga barang diantara 2000 sampai 6000 :

mysql> select * FROM barang WHERE harga between 2000 and 6000;

```
+---------+-------------+---------+-------+
| Kodebrg | Nama_Barang | Satuan  | Harga |
+---------+-------------+---------+-------+
| BR001   | Sabun LUX   | Buah    |  3000 |
| BR003   | Sabun Cuci  | Plastik |  3000 |
| BR005   | Obat Nyamuk | Bungkus |  3000 |
| BR006   | Lilin       | Bungkus |  4000 |
| BR009   | Ikan Kaleng | Kaleng  |  4500 |
| BR010   | Coca Cola   | Botol   |  2500 |
| BR011   | Sprite      | Botol   |  2000 |
+---------+-------------+---------+-------+
7 rows in set (0.00 sec)
```

Menampilkan data barang dimana harga barang tidak diantara 2000 sampai 6000 :

mysql> select * FROM barang WHERE harga not between 2000 and 6000;

```
+---------+------------------+---------+-------+
| Kodebrg | Nama_Barang      | Satuan  | Harga |
+---------+------------------+---------+-------+
| BR002   | Pepsodent 25 g   | Buah    |  1000 |
| BR004   | Pengharum Ruangan | Kaleng  | 10000 |
| BR007   | Korek Api        | Bungkus |   500 |
| BR008   | Penyedap Rasa    | Bungkus |  1000 |
| BR013   | Teh Sosro        | Botol   |  1500 |
+---------+------------------+---------+-------+
5 rows in set (0.00 sec)
```

Menampilkan nama barang dan satuan pada data barang dimana satuan = ‘bungkus’ :

mysql> select nama_barang, satuan FROM barang WHERE satuan in ('bungkus');

```
+---------------+---------+
| nama_barang   | satuan  |
+---------------+---------+
| Obat Nyamuk   | Bungkus |
| Lilin         | Bungkus |
| Korek Api     | Bungkus |
| Penyedap Rasa | Bungkus |
+---------------+---------+
4 rows in set (0.00 sec)
```

Menampilkan nama barang dan satuan pada data barang dimana satuan selain ‘bungkus’ :

mysql> select nama_barang, satuan FROM barang WHERE satuan not in ('bungkus');

```
+------------------+---------+
| nama_barang      | satuan  |
+------------------+---------+
| Sabun LUX        | Buah    |
| Pepsodent 25 g   | Buah    |
| Sabun Cuci       | Plastik |
| Pengharum Ruangan | Kaleng  |
| Ikan Kaleng      | Kaleng  |
| Coca Cola        | Botol   |
```

```
| Sprite            | Botol   |
| Fanta             | Botol   |
| Teh Sosro         | Botol   |
+-------------------+---------+
9 rows in set (0.00 sec)
```

Menampilkan data barang yang belum ada harganya :

```
mysql> select * FROM barang WHERE harga is null;
+---------+-------------+--------+-------+
| Kodebrg | Nama_Barang | Satuan | Harga |
+---------+-------------+--------+-------+
| BR012   | Fanta       | Botol  |  NULL |
+---------+-------------+--------+-------+
1 row in set (0.00 sec)
```

Jika IS NULL diganti menjadi = NULL maka hasilnya tidak akan terbaca :

```
mysql> select * FROM barang WHERE harga = null;
Empty set (0.00 sec)
```

Menampilkan data barang yang sudah ada harganya :

```
mysql> select * FROM barang WHERE harga is not null;
+---------+------------------+---------+-------+
| Kodebrg | Nama_Barang      | Satuan  | Harga |
+---------+------------------+---------+-------+
| BR001   | Sabun LUX        | Buah    |  3000 |
| BR002   | Pepsodent 25 g   | Buah    |  1000 |
| BR003   | Sabun Cuci       | Plastik |  3000 |
| BR004   | Pengharum Ruangan | Kaleng  | 10000 |
| BR005   | Obat Nyamuk      | Bungkus |  3000 |
| BR006   | Lilin            | Bungkus |  4000 |
| BR007   | Korek Api        | Bungkus |   500 |
| BR008   | Penyedap Rasa    | Bungkus |  1000 |
| BR009   | Ikan Kaleng      | Kaleng  |  4500 |
| BR010   | Coca Cola        | Botol   |  2500 |
| BR011   | Sprite           | Botol   |  2000 |
| BR013   | Teh Sosro        | Botol   |  1500 |
+---------+------------------+---------+-------+
12 rows in set (0.00 sec)
```

Operator LIKE sangat bermanfaat untuk mencari data semacam siapa saja pegawai yang namanya mengandung huruf a atau siapa saja yang mengandung huruf ‘ana’. Dalam melakukan pencarian dengan operator ini perlu dipergunakannya tanda *wildcard* berupa garis bawah(_) atau persen (%).

- Tanda garis bawah (_) berarti sebuah karakter apa saja

  Contoh :

  B_ _i cocok dengan budi, bedi, tetapi tidaklah cocok dengan budri, badri

- Tanda persen (%) berarti cocok dengan karakter apa saja dan berapapun panjangnya (termasuk cocok dengan nol karakter)

  Menampilkan data barang dengan nama barang didahului oleh kata sabun :

```
mysql> select * FROM barang WHERE nama_barang like 'sabun%';
+---------+-------------+---------+-------+
| Kodebrg | Nama_Barang | Satuan  | Harga |
+---------+-------------+---------+-------+
| BR001   | Sabun LUX   | Buah    |  3000 |
| BR003   | Sabun Cuci  | Plastik |  3000 |
+---------+-------------+---------+-------+
2 rows in set (0.00 sec)
```

Menampilkan nama barang yang didalamnya terdapat huruf i :

```
mysql> select nama_barang FROM barang WHERE nama_barang like '%i%';
+-------------+
| nama_barang |
+-------------+
| Sabun Cuci  |
| Lilin       |
| Korek Api   |
| Ikan Kaleng |
| Sprite      |
+-------------+
5 rows in set (0.00 sec)
```

Menampilkan nama barang yang didalamnya tidak terdapat huruf i :

```
mysql> select nama_barang FROM barang WHERE nama_barang not like '%i%';
+--------------------+
| nama_barang        |
+--------------------+
| Sabun LUX          |
| Pepsodent 25 g     |
| Pengharum Ruangan  |
| Obat Nyamuk        |
| Penyedap Rasa      |
| Coca Cola          |
| Fanta              |
| Teh Sosro          |
+--------------------+
8 rows in set (0.00 sec)
```

Menampilkan data barang dimana terdapat huruf a pada nama barang dan kata bungkus pada satuan :

```
mysql> select * FROM barang WHERE nama_barang like '%a%' and
    -> satuan like 'bungkus';
+---------+---------------+---------+-------+
| Kodebrg | Nama_Barang   | Satuan  | Harga |
+---------+---------------+---------+-------+
| BR005   | Obat Nyamuk   | Bungkus |  3000 |
| BR007   | Korek Api     | Bungkus |   500 |
| BR008   | Penyedap Rasa | Bungkus |  1000 |
+---------+---------------+---------+-------+
3 rows in set (0.00 sec)
```

BAB 5

FUNGSI-FUNGSI STANDAR SQL

5.1. AVG (ekspresi)

Berfungsi mencari nilai rata-rata pada suatu field bertipe numerik atau integer.

Contoh :

Lihat tampilan seluruh data barang :

```
mysql> select * FROM barang;
```

```
+---------+-------------------+---------+-------+
| Kodebrg | Nama_Barang       | Satuan  | Harga |
+---------+-------------------+---------+-------+
| BR001   | Sabun LUX         | Buah    |  3000 |
| BR002   | Pepsodent 25 g    | Buah    |  1000 |
| BR003   | Sabun Cuci        | Plastik |  3000 |
| BR004   | Pengharum Ruangan | Kaleng  | 10000 |
| BR005   | Obat Nyamuk       | Bungkus |  3000 |
| BR006   | Lilin             | Bungkus |  4000 |
| BR007   | Korek Api         | Bungkus |   500 |
| BR008   | Penyedap Rasa     | Bungkus |  1000 |
| BR009   | Ikan Kaleng       | Kaleng  |  4500 |
| BR010   | Coca Cola         | Botol   |  2500 |
| BR011   | Sprite            | Botol   |  2000 |
| BR012   | Fanta             | Botol   |  NULL |
| BR013   | Teh Sosro         | Botol   |  1500 |
+---------+-------------------+---------+-------+
13 rows in set (0.00 sec)
```

```
mysql> select avg(harga) as 'Harga Rata-Rata' FROM barang;
```

```
+-----------------+
| Harga Rata-Rata |
+-----------------+
|       3000.0000 |
+-----------------+
1 row in set (0.00 sec)
```

5.2. COUNT (x)

Berfungsi menghitung jumlah record dari suatu field atau tabel.

Contoh :

```
mysql> select count(kodebrg) as 'Jumlah Barang' FROM barang;
```

```
+---------------+
| Jumlah Barang |
+---------------+
|            13 |
+---------------+
1 row in set (0.00 sec)
```

```
mysql> select satuan, count(satuan) as 'Jumlah Barang' FROM barang
    -> group by satuan;
```

```
+---------+---------------+
| satuan  | Jumlah Barang |
+---------+---------------+
| Botol   |             4 |
| Buah    |             2 |
| Bungkus |             4 |
| Kaleng  |             2 |
| Plastik |             1 |
+---------+---------------+
5 rows in set (0.05 sec)
```

```
mysql> select count(distinct satuan) FROM barang;
+------------------------+
| count(distinct satuan) |
+------------------------+
|                      5 |
+------------------------+
1 row in set (0.06 sec)
```

5.3. MAX (ekspresi)

Berfungsi mencari nilai terbesar dari suatu field..

Contoh :

```
mysql> select max(harga) as 'Harga Barang Tertinggi' FROM barang;
+------------------------+
| Harga Barang Tertinggi |
+------------------------+
|                  10000 |
+------------------------+
1 row in set (0.00 sec)
```

5.4. MIN (ekspresi)

Berfungsi mencari nilai terkecil dari suatu field..

Contoh :

```
mysql> select min(harga) as 'Harga Barang Terendah' FROM barang;
+-----------------------+
| Harga Barang Terendah |
+-----------------------+
|                   500 |
+-----------------------+
1 row in set (0.00 sec)
```

5.5. SUM (ekspresi)

Berfungsi mendapatkan nilai total dari suatu field.

Contoh :

```
mysql> select sum(harga) as 'Total Harga' FROM barang;
+-------------+
| Total Harga |
+-------------+
|       36000 |
+-------------+
1 row in set (0.00 sec)
```

```
mysql> select satuan, sum(harga) as 'Total Harga' FROM barang
    -> group by satuan;
+---------+-------------+
| satuan  | Total Harga |
+---------+-------------+
| Botol   |        6000 |
| Buah    |        4000 |
| Bungkus |        8500 |
| Kaleng  |       14500 |
| Plastik |        3000 |
+---------+-------------+
5 rows in set (0.00 sec)
```

5.6. CEILING (x)

Berfungsi menghasilkan angka bulat terbesar yang lebih besar dari x.

Contoh :

```
mysql> select ceiling(7.83) as 'Pembulatan Keatas';
+-------------------+
| Pembulatan Keatas |
+-------------------+
|                 8 |
+-------------------+
1 row in set (0.00 sec)
```

5.7. FLOOR (x)

Berfungsi membulatkan angka ke integer terdekat yang lebih kecil dari nilai x.

Contoh :

```
mysql> select floor(7.897) as 'Pembulatan Kebawah';
+--------------------+
| Pembulatan Kebawah |
+--------------------+
|                  7 |
+--------------------+
1 row in set (0.00 sec)
```

5.8. ROUND (x)

Berfungsi membulatkan angka x menjadi angka bulat integer.

Contoh :

```
mysql> select round(7.897) as 'Pembulatan Terdekat';
+---------------------+
| Pembulatan Terdekat |
+---------------------+
|                   8 |
+---------------------+
1 row in set (0.00 sec)
```

```
mysql> select round(7.497) as 'Pembulatan Terdekat';
+--------------------+
| Pembulatan Terdekat |
+--------------------+
|                  7 |
+--------------------+
1 row in set (0.00 sec)
```

5.9. ROUND (x,y)

Berfungsi membulatkan nilai x sampai dengan posisi desimal y tertentu, jika y didefinisikan.

Contoh :

```
mysql> select round(7.345765,3);
+-------------------+
| round(7.345765,3) |
+-------------------+
|             7.346 |
+-------------------+
1 row in set (0.00 sec)
```

5.10. TRUNCATE (x,y)

Berfungsi melakukan pemenggalan suatu angka desimal dari nilai x, sampai sejumlah angka dibilangkan koma y.

Contoh :

```
mysql> select truncate(7.345765,3);
+----------------------+
| truncate(7.345765,3) |
+----------------------+
|                7.345 |
+----------------------+
1 row in set (0.00 sec)
```

5.11. FORMAT (num, dec)

Berfungsi memformat angka ke dalam format desimal dengan susunan ‘9,999,999,99’ dengan pembulatan sejumlah angka yang didefinisikan dengan dec

Contoh :

```
mysql> select format(750000,2);
+------------------+
| format(750000,2) |
+------------------+
| 750,000.00       |
+------------------+
1 row in set (0.00 sec)
```

```
mysql> select nama_barang, format(harga,0) as 'Harga Barang' FROM barang;
+-------------------+--------------+
| nama_barang       | Harga Barang |
+-------------------+--------------+
| Sabun LUX         | 3,000        |
| Pepsodent 25 g    | 1,000        |
```

```
| Sabun Cuci        | 3,000        |
| Pengharum Ruangan | 10,000       |
| Obat Nyamuk       | 3,000        |
| Lilin             | 4,000        |
| Korek Api         | 500          |
| Penyedap Rasa     | 1,000        |
| Ikan Kaleng       | 4,500        |
| Coca Cola         | 2,500        |
| Sprite            | 2,000        |
| Fanta             | NULL         |
| Teh Sosro         | 1,500        |
+-------------------+--------------+
13 rows in set (0.00 sec)
```

5.12. POW (x,y) atau POWER(x,y)

Berfungsi menghitung nilai x pangkat.

Contoh :

```
mysql> select pow(2,8) as '2 Pangkat 8';
+-------------+
| 2 Pangkat 8 |
+-------------+
|  256.000000 |
+-------------+
1 row in set (0.03 sec)
```

```
mysql> select ceiling(pow(2,8)) as '2 Pangkat 8';
+-------------+
| 2 Pangkat 8 |
+-------------+
|         256 |
+-------------+
1 row in set (0.00 sec)
```

```
mysql> select round(pow(2,8)) as '2 Pangkat 8';
+-------------+
| 2 Pangkat 8 |
+-------------+
|         256 |
+-------------+
1 row in set (0.02 sec)
```

5.13. SQRT (x)

Berfungsi mencari akar kuadrat angka x.

Contoh :

```
mysql> select sqrt(81) as 'Akar 81';
+----------+
| Akar 81  |
+----------+
| 9.000000 |
+----------+
1 row in set (0.00 sec)
```

```
mysql> select round(sqrt(81)) as 'Akar 81';
+---------+
| Akar 81 |
+---------+
|       9 |
+---------+
1 row in set (0.00 sec)
```

5.14. CONCAT (x,y,z,...)

Berfungsi menggabungkan beberapa nilai string menjadi satu nilai. Namun bila ada nilai yang mengandung NULL, maka hasilnya tetap NULL.

Contoh :

```
mysql> select concat('Selamat ', 'Belajar', ' MySQL');
+----------------------------------------+
| concat('Selamat ', 'Belajar', ' MySQL') |
+----------------------------------------+
| Selamat Belajar MySQL                  |
+----------------------------------------+
1 row in set (0.00 sec)
```

```
mysql> select concat('Selamat ', 'Belajar', Null);
+-------------------------------------+
| concat('Selamat ', 'Belajar', Null) |
+-------------------------------------+
| NULL                                |
+-------------------------------------+
1 row in set (0.00 sec)
```

5.15. LCASE (x) atau LOWER(x)

Berfungsi mengkonversi semua karakter dari nilai x ke huruf kecil semua.

Contoh :

```
mysql> select lcase('SELAMAT BELAJAR MYSQL');
+---------------------------------+
| lcase('SELAMAT BELAJAR MYSQL') |
+---------------------------------+
| selamat belajar mysql          |
+---------------------------------+
1 row in set (0.06 sec)
```

```
mysql> select lower(nama_barang) FROM barang limit 5;
+--------------------+
| lcase(nama_barang) |
+--------------------+
| sabun lux          |
| pepsodent 25 g     |
| sabun cuci         |
| pengharum ruangan  |
| obat nyamuk        |
+--------------------+
5 rows in set (0.00 sec)
```

5.16. UCASE (x) atau UPPER(x)

Berfungsi mengkonversi semua karakter dari nilai x ke huruf besar semua.

Contoh :

```
mysql> select ucase(nama_barang) FROM barang limit 4;
+--------------------+
| ucase(nama_barang) |
+--------------------+
| SABUN LUX          |
| PEPSODENT 25 G     |
| SABUN CUCI         |
| PENGHARUM RUANGAN  |
+--------------------+
4 rows in set (0.00 sec)
```

5.17. LEFT (x,y)

Berfungsi mengambil sejumlah y karakter dari string x, dimulai posisi pertama.

Contoh :

```
mysql> select left(nama_barang,7) FROM barang limit 3;
+---------------------+
| left(nama_barang,7) |
+---------------------+
| Sabun L             |
| Pepsode             |
| Sabun C             |
+---------------------+
3 rows in set (0.00 sec)
```

5.18. Right (x,y)

Berfungsi mengambil sejumlah y karakter dari string x, dimulai dari posisi paling terakhir (paling kanan).

Contoh :

```
mysql> select right(nama_barang,7) FROM barang limit 4;
+----------------------+
| right(nama_barang,7) |
+----------------------+
| bun LUX              |
| nt 25 g              |
| un Cuci              |
| Ruangan              |
+----------------------+
4 rows in set (0.00 sec)
```

5.19. MID (x,y,z)

Berfungsi mengambil sejumlah karakter dari string x sejumlah z karakter mulai dari posisi ke y.

Contoh :

```
mysql> select mid(nama_barang,3,4) FROM barang limit 4;
```

```
+----------------------+
| mid(nama_barang,3,4) |
+----------------------+
| bun                  |
| psod                 |
| bun                  |
| ngha                 |
+----------------------+
4 rows in set (0.00 sec)
```

5.20. LENGTH (x)

Berfungsi mendapatkan panjang suatu string.

Contoh :

```
mysql> select nama_barang, length(nama_barang) FROM barang
    -> limit 4;
```

```
+-------------------+----------------------+
| nama_barang       | length(nama_barang) |
+-------------------+----------------------+
| Sabun LUX         |                    9 |
| Pepsodent 25 g    |                   14 |
| Sabun Cuci        |                   10 |
| Pengharum Ruangan |                   17 |
+-------------------+----------------------+
4 rows in set (0.00 sec)
```

5.21. LTRIM (x)

Berfungsi membuang spasi di sebelah kiri string x, apabila x diawali dengan spasi.

Contoh :

```
mysql> select ltrim('   Selamat Belajar');
```

```
+-----------------------------+
| ltrim('   Selamat Belajar') |
+-----------------------------+
| Selamat Belajar             |
+-----------------------------+
1 row in set (0.00 sec)
```

5.22. RTRIM (x)

Berfungsi membuang spasi di sebelah kanan string x, apabila x diakhiri dengan spasi.

Contoh :

```
mysql> select rtrim('Selamat Belajar               ');
```

```
+---------------------------------------+
| rtrim('Selamat Belajar             ') |
+---------------------------------------+
| Selamat Belajar                       |
+---------------------------------------+
1 row in set (0.00 sec)
```

5.23. PASSWORD (password)

Berfungsi menghasilkan string password yang disandikan(dienkripsi).

Contoh :

```
mysql> select password('DEDY') as PasswordDedy;
+------------------+
| PasswordDedy     |
+------------------+
| 4ef4828c00c90998 |
+------------------+
1 row in set (0.00 sec)
```

5.24. REPEAT (x,y)

Berfungsi menghasilkan string x yang diulang sebanyak y kali.

Contoh :

```
mysql> select repeat('Dedy',6) as StringBerulang;
+--------------------------+
| StringBerulang           |
+--------------------------+
| DedyDedyDedyDedyDedyDedy |
+--------------------------+
1 row in set (0.00 sec)
```

5.25. REPLACE (x,y,z)

Berfungsi mengganti semua string y di dalam string x yang ditemukan dengan string z.

Contoh :

```
mysql> select replace('DEDI ARISANDI','I','Y');
+----------------------------------+
| replace('DEDI ARISANDI','I','Y') |
+----------------------------------+
| DEDY ARYSANDY                    |
+----------------------------------+
1 row in set (0.00 sec)
```

5.26. REVERSE (x)

Berfungsi menghasilkan string yang membalik urutan penulisan dari string x.

Contoh :

```
mysql> select reverse('DEDY ARISANDI');
+--------------------------+
| reverse('DEDY ARISANDI') |
+--------------------------+
| IDNASIRA YDED            |
+--------------------------+
1 row in set (0.00 sec)
```

5.27. INSERT(str, i, j, strpengganti)

Berfungsi menghasilkan string yang berasal dari str, dengan karakter mulai dari i dan sebanyak j karakter diganti dengan strpengganti.

Contoh :

```
mysql> SELECT INSERT('abcde',2,2,'SWER');
```

```
+----------------------------+
| INSERT('abcde',2,2,'SWER') |
+----------------------------+
| aSWERde                    |
+----------------------------+
1 row in set (0.00 sec)
```

5.28. LPAD (str, n, pengisi) atau RPAD (str, n, pengisi)

Menghasilkan string yang berupa str diatur agar memiliki panjang n karakter. Dalam hal ini, pengisi digunakan sebagai penambah agar string hasil memiliki panjang n karakter. Penambahan dilakukan di bagian kiri atau kanan.

Contoh :

```
mysql> SELECT name, LPAD(price,5,char(32))
    -> FROM food;
+----------------+------------------------+
| name           | LPAD(price,5,char(32)) |
+----------------+------------------------+
| Nasi Goreng    | 10000                  |
| Nasi Uduk      | 15000                  |
| Mie Aceh       | 11000                  |
| Soto Makasar   | 13500                  |
| Bakso          |  9000                  |
| Mie Ayam       |  9000                  |
| Sate Madura    | 12500                  |
| Empek-Empek    | 11500                  |
| Gado-Gado      |  8500                  |
| Pecel          |  7500                  |
| Tahu Gimbal    | 10000                  |
| Rujak Cingur   |  9000                  |
| Gudeg          |  7000                  |
| Nasi Rawon     |  7500                  |
| Sroto Sokaraja |  9500                  |
| Soto Medan     |  9000                  |
| Sate Padang    | 10000                  |
| Ayam Tangkap   | 15000                  |
| Ikan Bakar     | 17000                  |
| Ayam Goreng    | 13000                  |
+----------------+------------------------+
20 rows in set (0.00 sec)
```

5.29. ADDDATE(tgl, INTERVAL tipe_ekspr)

Menghasilkan tanggal yang merupakan penjumlahan antara tanggal dan nilai INTERVAl tipe_ekspr.

Tipe_ekspr dapat diisi sesuai dengan table dibawah ini :

Nama Interval	Keterangan
MICROSECOND	Mikrodetik
SECOND	Detik
MINUTE	Menit
HOUR	Jam
DAY	Hari
WEEK	Minggu
MONTH	Bulan
QUARTER	Perempattahun
YEAR	Tahun
SECOND_MICROSECOND	Detik dan mikrodetik, dengan format 'detik.mikrodetik'
MINUTE_MICROSECOND	Menit dan mikrodetik, dengan format 'menit.mikrodetik'
MINUTE_SECOND	Menit dan detik, dengan format 'menit: detik'
HOUR_MICROSECOND	Jam dan mikorodetik, dengan format 'jam.mikrodetik'
HOUR_SECOND	Jam, menit, dan detik dengan format 'Jam:Menit:Detik'
HOUR_MINUTE	Jam dan menit, dengan format 'Jam:menit'
DAY_MICROSECOND	Jam dan mikorodetik, dengan format 'jam.mikrodetik'
DAY_SECOND	Hari, jam, menit dan detik, dengan format 'hari jam:menit: detik'
DAY_MINUTE	Hari, jam, dan menit, dengan format 'hari jam:menit'
DAY_HOUR	Hari dan jam dengan format 'hari jam'
YEAR_MONTH	Tahun dan bulan, dengan format 'tahun-bulan'

Contoh :

```
mysql> SELECT ADDDATE('2011/11/17',INTERVAL 5 DAY);
+--------------------------------------+
| ADDDATE('2011/11/17',INTERVAL 5 DAY) |
+--------------------------------------+
| 2011-11-22                           |
+--------------------------------------+
1 row in set (0.00 sec)

mysql> SELECT ADDDATE('2011/11/17',INTERVAL 5 MONTH);
+---------------------------------------+
| ADDDATE('2011/11/17',INTERVAL 5 MONTH) |
+---------------------------------------+
| 2012-04-17                            |
+---------------------------------------+
1 row in set (0.00 sec)
```

5.30. ADDTIME (ekspr1, ekspr2)

Menghasilkan waktu yang merupakan penjumlahan dari ekspr1 dan ekspr2.

Contoh :

```
mysql> SELECT ADDDATE('2011/11/17 10:0:0','1 10:45:20');
+-------------------------------------------+
| ADDDATE('2011/11/17 10:0:0','1 10:45:20') |
+-------------------------------------------+
| 2011-11-18 10:00:00                       |
+-------------------------------------------+
1 row in set, 1 warning (0.00 sec)
```

5.31. CURDATE()

Menghasilkan tanggal sekarang.

Contoh :

```
mysql> SELECT CURDATE();
```

```
+------------+
| CURDATE()  |
+------------+
| 2011-11-17 |
+------------+
1 row in set (0.00 sec)
```

5.32. CURTIME()

Menghasilkan jam sekarang dengan format JJ:MM:DD. Contoh :

```
mysql> SELECT CURTIME();
```

```
+-----------+
| CURTIME() |
+-----------+
| 07:11:20  |
+-----------+
1 row in set (0.00 sec)
```

5.33. NOW()

Menghasilkan tanggal dan jam sekarang. Contoh :

```
mysql> SELECT NOW();
```

```
+---------------------+
| NOW()               |
+---------------------+
| 2011-11-17 07:32:41 |
+---------------------+
1 row in set (0.00 sec)
```

5.34. DATE (ekspr)

Menghasilkan bagian tanggal dari suatu ekspresi yang mengandung tanggal dan jam. Contoh :

```
mysql> SELECT DATE('1992/12/12 11:20:11');
```

```
+-----------------------------+
| DATE('1992/12/12 11:20:11') |
+-----------------------------+
| 1992-12-12                  |
+-----------------------------+
1 row in set (0.00 sec)
```

5.35. DATEDIFF (ekspr1, ekspr2)

Menghasilkan selisih hari dari kedua argument. Contoh :

```
mysql> SELECT DATEDIFF('1999/12/20', '1999/12/29');
```

```
+-------------------------------------+
| DATEDIFF('1999/12/20', '1999/12/29') |
+-------------------------------------+
|                                  -9 |
+-------------------------------------+
1 row in set (0.06 sec)
```

```
mysql> SELECT DATEDIFF('1999/12/29', '1999/12/20');
+--------------------------------------+
| DATEDIFF('1999/12/29', '1999/12/20') |
+--------------------------------------+
|                                    9 |
+--------------------------------------+
1 row in set (0.02 sec)
```

5.36. DAY (tgl) atau DAYOFMONTH (tgl)

Menghasilkan bagian tanggal (1 s/d 31) dari suatu tanggal. Contoh :

```
mysql> SELECT DAY('1990/05/30');
+-------------------+
| DAY('1990/05/30') |
+-------------------+
|                30 |
+-------------------+
1 row in set (0.00 sec)
```

```
mysql> SELECT DAYOFMONTH('1990/05/30');
+---------------------------+
| DAYOFMONTH('1990/05/30') |
+---------------------------+
|                        30 |
+---------------------------+
1 row in set (0.00 sec)
```

5.37. DAYNAME (tgl)

Menghasilkan nama hari. Contoh :

```
mysql> SELECT DAYNAME('1990/05/30');
+-----------------------+
| DAYNAME('1990/05/30') |
+-----------------------+
| Wednesday             |
+-----------------------+
1 row in set (0.00 sec)
```

5.38. DAYOFWEEK (tgl)

Menghasilkan kode hari (1 = minggu, 2 = senin, 3 = selasa, 4 = rabu, 5 = kamis, 6 = jumat dan 7 = sabtu). Contoh :

```
mysql> SELECT DAYOFWEEK('1990/05/30');
+-------------------------+
| DAYOFWEEK('1990/05/30') |
+-------------------------+
|                       4 |
+-------------------------+
1 row in set (0.00 sec)
```

5.39. DAYOFYEAR (tgl)

Menghasilkan posisi hari dalam satu tahun. Contoh :

```
mysql> SELECT DAYOFYEAR('1990/05/30');
```

```
+-------------------------+
| DAYOFYEAR('1990/05/30') |
+-------------------------+
|                     150 |
+-------------------------+
1 row in set (0.00 sec)
```

5.40. HOUR (time)

Menghasilkan bagian jam dari suatu waktu. Contoh :

```
mysql> SELECT HOUR('12:11:11');
```

```
+------------------+
| HOUR('12:11:11') |
+------------------+
|               12 |
+------------------+
1 row in set (0.00 sec)
```

5.41. LAST_DAY (date)

Menghasilkan tanggal terakhir dari bulan yang tertera pada tanggal. Contoh :

```
mysql> SELECT LAST_DAY('1990/05/30');
```

```
+------------------------+
| LAST_DAY('1990/05/30') |
+------------------------+
| 1990-05-31             |
+------------------------+
1 row in set (0.00 sec)
```

5.42. MINUTE (time)

Menghasilkan bagian menit dari suatu waktu. Contoh :

```
mysql> SELECT MINUTE('12:11:11');
```

```
+--------------------+
| MINUTE('12:11:11') |
+--------------------+
|                 11 |
+--------------------+
1 row in set (0.00 sec)
```

5.43. MONTH (date)

Menghasilkan nilai bulan (1 s/d 12) dari suatu tanggal. Contoh :

5.44. MONTHNAME (date)

Menghasilkan nama bulan. Contoh :

```
mysql> SELECT MONTHNAME('1990/05/30');
```

```
+-------------------------+
| MONTHNAME('1990/05/30') |
+-------------------------+
| May                     |
+-------------------------+
1 row in set (0.00 sec)
```

5.45. SECOND (time)

Menghasilkan bagian detik dari suatu waktu. Contoh :

```
mysql> SELECT SECOND('12:11:11');
+--------------------+
| SECOND('12:11:11') |
+--------------------+
|                 11 |
+--------------------+
1 row in set (0.00 sec)
```

5.46. SYSDATE ()

Menghasilkan tanggal dan jam sekarang. Contoh :

```
mysql> SELECT SYSDATE();
+---------------------+
| SYSDATE()           |
+---------------------+
| 2011-11-17 07:37:59 |
+---------------------+
1 row in set (0.00 sec)
```

5.47. WEEKDAY(date)

Menghasilkan kode hari (0 = senin, 1 = selasa, 2 = rabu, 3 = kamis, 4 = jumat, 5 = sabtu dan 6 = minggu). Contoh :

```
mysql> SELECT WEEKDAY('1990/05/30');
+-----------------------+
| WEEKDAY('1990/05/30') |
+-----------------------+
|                     2 |
+-----------------------+
1 row in set (0.00 sec)
```

5.48. YEAR (tanggal)

Menghasilkan nilai bulan (1 s/d 12) dari suatu tanggal. Contoh :

```
mysql> SELECT YEAR('1990/05/30');
+--------------------+
| YEAR('1990/05/30') |
+--------------------+
|               1990 |
+--------------------+
1 row in set (0.00 sec)
```

BAB 6

MENDALAMI PERINTAH SELECT

6.1. Mengurutkan Data

Untuk mengurutkan data pada perintah SELECT dapat menggunakan klausa ORDER BY. Data dapat diurutkan secara ascending (ASC) atau DESCending (DESC).

Ketiklah perintah SQL berikut ini untuk mengetahui cara pengurutan data :

```
mysql> select * FROM barang order by nama_barang;
+---------+-------------------+---------+-------+
| Kodebrg | Nama_Barang       | Satuan  | Harga |
+---------+-------------------+---------+-------+
| BR010   | Coca Cola         | Botol   |  2500 |
| BR012   | Fanta             | Botol   |  NULL |
| BR009   | Ikan Kaleng       | Kaleng  |  4500 |
| BR007   | Korek Api         | Bungkus |   500 |
| BR006   | Lilin             | Bungkus |  4000 |
| BR005   | Obat Nyamuk       | Bungkus |  3000 |
| BR004   | Pengharum Ruangan | Kaleng  | 10000 |
| BR008   | Penyedap Rasa     | Bungkus |  1000 |
| BR002   | Pepsodent 25 g    | Buah    |  1000 |
| BR003   | Sabun Cuci        | Plastik |  3000 |
| BR001   | Sabun LUX         | Buah    |  3000 |
| BR011   | Sprite            | Botol   |  2000 |
| BR013   | Teh Sosro         | Botol   |  1500 |
+---------+-------------------+---------+-------+
13 rows in set (0.00 sec)

mysql> select * FROM barang order by nama_barang DESC;
+---------+-------------------+---------+-------+
| Kodebrg | Nama_Barang       | Satuan  | Harga |
+---------+-------------------+---------+-------+
| BR013   | Teh Sosro         | Botol   |  1500 |
| BR011   | Sprite            | Botol   |  2000 |
| BR001   | Sabun LUX         | Buah    |  3000 |
| BR003   | Sabun Cuci        | Plastik |  3000 |
| BR002   | Pepsodent 25 g    | Buah    |  1000 |
| BR008   | Penyedap Rasa     | Bungkus |  1000 |
| BR004   | Pengharum Ruangan | Kaleng  | 10000 |
| BR005   | Obat Nyamuk       | Bungkus |  3000 |
| BR006   | Lilin             | Bungkus |  4000 |
| BR007   | Korek Api         | Bungkus |   500 |
| BR009   | Ikan Kaleng       | Kaleng  |  4500 |
| BR012   | Fanta             | Botol   |  NULL |
| BR010   | Coca Cola         | Botol   |  2500 |
+---------+-------------------+---------+-------+
13 rows in set (0.00 sec)

mysql> select satuan, nama_barang, kodebrg FROM barang
    -> order by satuan asc, nama_barang DESC;
+---------+-------------------+---------+
| satuan  | nama_barang       | kodebrg |
+---------+-------------------+---------+
| Botol   | Teh Sosro         | BR013   |
```

```
| Botol   | Sprite            | BR011   |
| Botol   | Fanta             | BR012   |
| Botol   | Coca Cola         | BR010   |
| Buah    | Sabun LUX         | BR001   |
| Buah    | Pepsodent 25 g    | BR002   |
| Bungkus | Penyedap Rasa     | BR008   |
| Bungkus | Obat Nyamuk       | BR005   |
| Bungkus | Lilin             | BR006   |
| Bungkus | Korek Api         | BR007   |
| Kaleng  | Pengharum Ruangan | BR004   |
| Kaleng  | Ikan Kaleng       | BR009   |
| Plastik | Sabun Cuci        | BR003   |
+---------+-------------------+---------+
13 rows in set (0.00 sec)
```

6.2. Mengelompokkan Data

Untuk mengelompokkan data pada perintah SELECT dapat menggunakan klausa GROUP BY. Misalkan satuan pada data barang akan dikelompokkan maka dapat menggunakan perintah berikut :

```
mysql> select satuan FROM barang group by satuan;
```

```
+---------+
| satuan  |
+---------+
| Botol   |
| Buah    |
| Bungkus |
| Kaleng  |
| Plastik |
+---------+
5 rows in set (0.03 sec)
```

```
mysql> select * FROM barang group by harga, satuan;
```

```
+---------+-------------------+---------+-------+
| Kodebrg | Nama_Barang       | Satuan  | Harga |
+---------+-------------------+---------+-------+
| BR012   | Fanta             | Botol   |  NULL |
| BR007   | Korek Api         | Bungkus |   500 |
| BR002   | Pepsodent 25 g    | Buah    |  1000 |
| BR008   | Penyedap Rasa     | Bungkus |  1000 |
| BR013   | Teh Sosro         | Botol   |  1500 |
| BR011   | Sprite            | Botol   |  2000 |
| BR010   | Coca Cola         | Botol   |  2500 |
| BR001   | Sabun LUX         | Buah    |  3000 |
| BR005   | Obat Nyamuk       | Bungkus |  3000 |
| BR003   | Sabun Cuci        | Plastik |  3000 |
| BR006   | Lilin             | Bungkus |  4000 |
| BR009   | Ikan Kaleng       | Kaleng  |  4500 |
| BR004   | Pengharum Ruangan | Kaleng  | 10000 |
+---------+-------------------+---------+-------+
13 rows in set (0.02 sec)
```

6.3. Mengelompokkan Data Dengan Kondisi

Untuk mengelompokkan data dengan kondisi pada perintah SELECT dapat menggunakan klausa HAVING. Ketiklah perintah-perintah berikut ini :

Perintah untuk menampilkan data barang yang dikelompokkan berdasarkan harga dan satuan dimana harga tidak sama dengan null :

```
mysql> select * FROM barang group by harga, satuan
    -> having harga is not null;
+---------+------------------+---------+-------+
| Kodebrg | Nama_Barang      | Satuan  | Harga |
+---------+------------------+---------+-------+
| BR007   | Korek Api        | Bungkus |   500 |
| BR002   | Pepsodent 25 g   | Buah    |  1000 |
| BR008   | Penyedap Rasa    | Bungkus |  1000 |
| BR013   | Teh Sosro        | Botol   |  1500 |
| BR011   | Sprite           | Botol   |  2000 |
| BR010   | Coca Cola        | Botol   |  2500 |
| BR001   | Sabun LUX        | Buah    |  3000 |
| BR005   | Obat Nyamuk      | Bungkus |  3000 |
| BR003   | Sabun Cuci       | Plastik |  3000 |
| BR006   | Lilin            | Bungkus |  4000 |
| BR009   | Ikan Kaleng      | Kaleng  |  4500 |
| BR004   | Pengharum Ruangan | Kaleng  | 10000 |
+---------+------------------+---------+-------+
12 rows in set (0.00 sec)
```

Perintah untuk menampilkan data barang yang dikelompokkan berdasarkan harga dan satuan dimana harga tidak sama dengan null dan satuan tidak sama dengan 'botol':

```
mysql> select * FROM barang group by harga, satuan
    -> having harga is not null and satuan <> 'botol';
+---------+------------------+---------+-------+
| Kodebrg | Nama_Barang      | Satuan  | Harga |
+---------+------------------+---------+-------+
| BR007   | Korek Api        | Bungkus |   500 |
| BR002   | Pepsodent 25 g   | Buah    |  1000 |
| BR008   | Penyedap Rasa    | Bungkus |  1000 |
| BR001   | Sabun LUX        | Buah    |  3000 |
| BR005   | Obat Nyamuk      | Bungkus |  3000 |
| BR003   | Sabun Cuci       | Plastik |  3000 |
| BR006   | Lilin            | Bungkus |  4000 |
| BR009   | Ikan Kaleng      | Kaleng  |  4500 |
| BR004   | Pengharum Ruangan | Kaleng  | 10000 |
+---------+------------------+---------+-------+
9 rows in set (0.00 sec)
```

6.4. Membatasi Penampilan Data

Untuk membatasi penampilan data pada perintah SELECT dapat menggunakan klausa LIMIT. Ketiklah perintah-perintah berikut ini :

Perintah untuk menampilkan 4 baris data saja pada tabel barang :

```
mysql> select * FROM barang limit 4;
+---------+------------------+---------+-------+
| Kodebrg | Nama_Barang      | Satuan  | Harga |
+---------+------------------+---------+-------+
| BR001   | Sabun LUX        | Buah    |  3000 |
| BR002   | Pepsodent 25 g   | Buah    |  1000 |
| BR003   | Sabun Cuci       | Plastik |  3000 |
| BR004   | Pengharum Ruangan | Kaleng  | 10000 |
+---------+------------------+---------+-------+
4 rows in set (0.00 sec)
```

Perintah untuk menampilkan 3 baris data dimulai pada baris ke 5 pada tabel barang :

```
mysql> select * FROM barang limit 4,3;
```

```
+---------+-------------+---------+-------+
| Kodebrg | Nama_Barang | Satuan  | Harga |
+---------+-------------+---------+-------+
| BR005   | Obat Nyamuk | Bungkus |  3000 |
| BR006   | Lilin       | Bungkus |  4000 |
| BR007   | Korek Api   | Bungkus |   500 |
+---------+-------------+---------+-------+
3 rows in set (0.00 sec)
```

Perintah untuk menampilkan 3 baris data dimulai pada baris pertama pada tabel barang :

```
mysql> select * FROM barang limit 0,3;
```

```
+---------+---------------+---------+-------+
| Kodebrg | Nama_Barang   | Satuan  | Harga |
+---------+---------------+---------+-------+
| BR001   | Sabun LUX     | Buah    |  3000 |
| BR002   | Pepsodent 25 g | Buah    |  1000 |
| BR003   | Sabun Cuci    | Plastik |  3000 |
+---------+---------------+---------+-------+
3 rows in set (0.00 sec
```

BAB 7

EKSPRESI DALAM QUERY

7.1. Ekspresi Berkondisi dengan CASE

Bentuk sederhana dari ekspresi berkondisi dengan CASE ini adalah sebagai berikut :

CASE nilai_ekspresi

WHEN nilai_ekspresi

THEN nilai_ekspresi

[**WHEN** nilai_ekspresi

THEN nilai_ekspresi]

[**ELSE** nilai_ekspresi]

END

Yang ada dalam tanda [] bersifat optional. Bentuk tersebut dapat dimanfaatkan untuk mengubah output yang berbeda dengan data asalnya. Contoh berikut menunjukkan cara mengubah isi jenisbrg "bungkus" menjadi "bngks".

```
mysql> SELECT namabrg,
    -> CASE jenisbrg
    -> WHEN 'Bungkus' THEN 'Bngks'
    -> END AS 'Jenis_Barang'
    -> FROM barang;
+-------------------+--------------+
| namabrg           | Jenis_Barang |
+-------------------+--------------+
| Sabun LUX         | NULL         |
| Pasta Gigi        | NULL         |
| Sabun cuci        | Bngks        |
| Pengharum Ruangan | NULL         |
| Obat Nyamuk       | Bngks        |
| Lilin             | Bngks        |
| Korek Api         | Bngks        |
| Penyedap Rasa     | Bngks        |
| Ikan Kaleng       | NULL         |
| Minuman Botol     | NULL         |
+-------------------+--------------+
10 rows in set (0.00 sec)
```

Contoh menunjukkan penggunaan sejumlah WHEN dalam CASE

```
mysql> SELECT namabrg,
    -> CASE jenisbrg
    -> WHEN 'Bungkus' THEN 'Bngks'
    -> WHEN 'Buah' THEN 'Bh'
    -> WHEN 'Kaleng' THEN 'Klng'
    -> WHEN 'Botol' THEN 'Btl'
```

```
    -> END AS 'Jenis_Barang'
    -> FROM barang;
+-------------------+--------------+
| namabrg           | Jenis_Barang |
+-------------------+--------------+
| Sabun LUX         | Bh           |
| Pasta Gigi        | Bh           |
| Sabun cuci        | Bngks        |
| Pengharum Ruangan | Klng         |
| Obat Nyamuk       | Bngks        |
| Lilin             | Bngks        |
| Korek Api         | Bngks        |
| Penyedap Rasa     | Bngks        |
| Ikan Kaleng       | Klng         |
| Minuman Botol     | Btl          |
+-------------------+--------------+
10 rows in set (0.00 sec)
```

Contoh kompleks

```
mysql> SELECT namabrg,
    -> CASE
    -> WHEN harga > 5000 THEN 'Harga di atas lima ribu rupiah'
    -> WHEN harga < 5000 THEN 'Harga di bawah lima ribu rupiah'
    -> ELSE 'Harga sama dengan lima ribu rupiah'
    -> END AS 'Info Harga'
    -> FROM barang;
+-------------------+-----------------------------------+
| namabrg           | Info Harga                        |
+-------------------+-----------------------------------+
| Sabun LUX         | Harga di bawah lima ribu rupiah   |
| Pasta Gigi        | Harga sama dengan lima ribu rupiah |
| Sabun cuci        | Harga di bawah lima ribu rupiah   |
| Pengharum Ruangan | Harga di atas lima ribu rupiah    |
| Obat Nyamuk       | Harga di bawah lima ribu rupiah   |
| Lilin             | Harga di bawah lima ribu rupiah   |
| Korek Api         | Harga di bawah lima ribu rupiah   |
| Penyedap Rasa     | Harga di bawah lima ribu rupiah   |
| Ikan Kaleng       | Harga di bawah lima ribu rupiah   |
| Minuman Botol     | Harga di bawah lima ribu rupiah   |
+-------------------+-----------------------------------+
10 rows in set (0.00 sec)
```

7.2. Ekspresi Berkondisi dengan IF

Selain menggunakan CASE, Anda bisa menggunakan fungsi IF() untuk menangani kondisi output. Bentuk umum adalah :

```
IF(ekspresi1,ekspresi2,ekspresi3)
```

Contoh penggunaan

```
mysql> SELECT id_person,
    -> IF(gender='M','Pria','Wanita')
    -> FROM person;
```

```
+-----------+--------------------------------+
| id_person | IF(gender='M','Pria','Wanita') |
+-----------+--------------------------------+
|         1 | Pria                           |
|         2 | Wanita                         |
|         3 | Pria                           |
|         4 | Pria                           |
|         5 | Wanita                         |
|         6 | Pria                           |
|         7 | Wanita                         |
|         8 | Wanita                         |
|         9 | Wanita                         |
|        10 | Pria                           |
+-----------+--------------------------------+
10 rows in set (0.00 sec)
```

7.3. Kondisi dengan NULL

Terkait dengan nilai NULL, terdapat fungsi berkondisi berupa fungsi IFNULL. Bentuk umum adalah :

IFNULL(ekspresi1,ekspresi2)

Fungsi ini menghasilkan nilai balik berupa ekspresi 1 kalau ekspresi 1 tidak bernilai NULL. Untuk Keadaan sebaliknya, hasilnya berupa ekspresi2. Contoh :

```
mysql> SELECT id_person, city FROM person;
+-----------+----------+
| id_person | city     |
+-----------+----------+
|         1 | medan    |
|         2 | NULL     |
|         3 | medan    |
|         4 | padang   |
|         5 | makassar |
|         6 | karo     |
|         7 | karo     |
|         8 | karo     |
|         9 | karo     |
|        10 | medan    |
+-----------+----------+
10 rows in set (0.02 sec)

mysql> SELECT id_person,
    -> IFNULL(city, '--Kota Belum Diisi--')
    -> FROM person;
+-----------+---------------------------------------+
| id_person | IFNULL(city, '--Kota Belum Diisi--') |
+-----------+---------------------------------------+
|         1 | medan                                 |
|         2 | --Kota Belum Diisi--                  |
|         3 | medan                                 |
|         4 | padang                                |
|         5 | makassar                              |
|         6 | karo                                  |
|         7 | karo                                  |
|         8 | karo                                  |
|         9 | karo                                  |
|        10 | medan                                 |
+-----------+---------------------------------------+
10 rows in set (0.00 sec)
```

BAB 8
QUERY

8.1. Membuat Query

Query merupakan suatu proses pengolahan data yang digunakan untuk memberikan hasil dari database berdasarkan kriteria tertentu.

Ketiklah tabel-tabel berikut ini pada database pelanggan :

```
mysql> CREATE TABLE Pelanggan
    -> (KdPlg char(5) not null primary key,
    -> NamaPlg char(18) not null,
    -> Alamat char(25),
    -> Kota char(10),
    -> NoHP char(12));
Query OK, 0 rows affected (0.03 sec)
```

```
mysql> DESC pelanggan;
+----------+----------+------+-----+---------+-------+
| Field    | Type     | Null | Key | Default | Extra |
+----------+----------+------+-----+---------+-------+
| KdPlg    | char(5)  |      | PRI |         |       |
| NamaPlg  | char(18) |      |     |         |       |
| Alamat   | char(25) | YES  |     | NULL    |       |
| Kota     | char(10) | YES  |     | NULL    |       |
| NoHP     | char(12) | YES  |     | NULL    |       |
+----------+----------+------+-----+---------+-------+
5 rows in set (0.00 sec)
```

```
mysql> CREATE TABLE FakturJual
    -> (NoFaktur char(6) not null,
    -> Tanggal date,
    -> KdPlg char(5) not null,
    -> primary key (NoFaktur, KdPlg));
Query OK, 0 rows affected (0.03 sec)
```

```
mysql> DESC FakturJual;
+----------+----------+------+-----+---------+-------+
| Field    | Type     | Null | Key | Default | Extra |
+----------+----------+------+-----+---------+-------+
| NoFaktur | char(6)  |      | PRI |         |       |
| Tanggal  | date     | YES  |     | NULL    |       |
| KdPlg    | char(5)  |      | PRI |         |       |
+----------+----------+------+-----+---------+-------+
3 rows in set (0.00 sec)
```

```
mysql> CREATE TABLE Penjualan
    -> (NoFaktur char(6) not null,
    -> KodeBrg char(5) not null,
    -> JlhJual integer(4),
    -> primary key (NoFaktur, KodeBrg));
Query OK, 0 rows affected (0.03 sec)
```

```
mysql> DESC Penjualan;
+----------+---------+------+-----+---------+-------+
| Field    | Type    | Null | Key | Default | Extra |
+----------+---------+------+-----+---------+-------+
| NoFaktur | char(6) |      | PRI |         |       |
| KodeBrg  | char(5) |      | PRI |         |       |
| JlhJual  | int(4)  | YES  |     | NULL    |       |
+----------+---------+------+-----+---------+-------+
3 rows in set (0.02 sec)
```

Isilah data untuk setiap tabel di atas sesuai dengan data pada halaman 22. Lihat cara pengisian data pada modul-modul sebelumnya (gunakan perintah INSERT).

```
mysql> SELECT * FROM pelanggan;
+-------+----------------+-------------------------+------------+--------------+
| KdPlg | NamaPlg        | Alamat                  | Kota       | NoHP         |
+-------+----------------+-------------------------+------------+--------------+
| PL001 | Muhammad Yusuf | Jl. Cempaka Putih No. 7 | Medan      | 081534762647 |
| PL002 | Doni Siregar   | Jl. Tapilaya No. 80     | Siantar    | -            |
| PL003 | Alfiansyah     | Jl. Kemenangan No. 20   | Medan      | 081361787526 |
| PL004 | Deni Sebayang  | Jl. Sutomo no. 5        | Perbaungan | 081361652007 |
| PL005 | Indri Siregar  | Jl. Juanda no. 40       | Siantar    | 081361652009 |
| PL006 | Pipit Hariati  | Jl. Jumono no. 32       | Medan      | -            |
+-------+----------------+-------------------------+------------+--------------+
6 rows in set (0.02 sec)
```

```
mysql> SELECT * FROM fakturjual;
+----------+------------+-------+
| NoFaktur | Tanggal    | kdplg |
+----------+------------+-------+
| FKR001   | 2006-07-20 | PL001 |
| FKR002   | 2006-07-20 | PL002 |
| FKR003   | 2006-07-20 | PL004 |
| FKR004   | 2006-07-21 | PL001 |
+----------+------------+-------+
4 rows in set (0.00 sec)
```

```
mysql> SELECT * FROM Penjualan;
+----------+---------+---------+
| NoFaktur | KodeBrg | JlhJual |
+----------+---------+---------+
| FKR001   | BR001   |       5 |
| FKR001   | BR003   |       3 |
| FKR001   | BR004   |       1 |
| FKR001   | BR007   |      10 |
| FKR002   | BR001   |       7 |
| FKR002   | BR002   |       4 |
| FKR002   | BR004   |       2 |
```

```
| FKR003   | BR010   |       3 |
| FKR003   | BR009   |       6 |
| FKR003   | BR001   |       3 |
| FKR003   | BR003   |       6 |
| FKR003   | BR006   |       8 |
| FKR003   | BR008   |       2 |
+----------+---------+---------+
13 rows in set (0.02 sec)
```

8.2. Query Dengan Dua Tabel

Aturan dalam melakukan query antartabel :

1. Setiap field disebutkan bersama dengan nama tabelnya, dipisahkan tanda titik (**.**).

   Sintak : **namatabel.namafield**

   Contoh : **Pelanggan.KdPlg** artinya field **KdPlgn** dari tabel **Pelanggan**.

2. Setiap tabel yang terlibat dalam proses query harus disebutkan dalam klausa FROM, dengan pemisah koma (**,**)

   Contoh : **FROM FakturJual, Pelanggan**

3. Kondisi dalam klausa WHERE mempengaruhi jenis join yang tercipta.

Ketiklah perintah penggabungan dua tabel berikut ini :

```
mysql> SELECT fakturjual.nofaktur, fakturjual.tanggal, fakturjual.kdplg,
    -> pelanggan.namaplg, pelanggan.kota FROM fakturjual, pelanggan
    -> WHERE fakturjual.kdplg=pelanggan.kdplg;
+----------+------------+-------+----------------+------------+
| nofaktur | tanggal    | kdplg | namaplg        | kota       |
+----------+------------+-------+----------------+------------+
| FKR001   | 2006-07-20 | PL001 | Muhammad Yusuf | Medan      |
| FKR002   | 2006-07-20 | PL002 | Doni Siregar   | Siantar    |
| FKR003   | 2006-07-20 | PL004 | Deni Sebayang  | Perbaungan |
| FKR004   | 2006-07-21 | PL001 | Muhammad Yusuf | Medan      |
+----------+------------+-------+----------------+------------+
4 rows in set (0.00 sec)
```

cara lain :

```
mysql> SELECT f.nofaktur, f.tanggal, f.kdplg,
    -> p.namaplg, p.kota FROM fakturjual f, pelanggan p
    -> WHERE f.kdplg=p.kdplg;
+----------+------------+-------+----------------+------------+
| nofaktur | tanggal    | kdplg | namaplg        | kota       |
+----------+------------+-------+----------------+------------+
| FKR001   | 2006-07-20 | PL001 | Muhammad Yusuf | Medan      |
| FKR002   | 2006-07-20 | PL002 | Doni Siregar   | Siantar    |
| FKR003   | 2006-07-20 | PL004 | Deni Sebayang  | Perbaungan |
| FKR004   | 2006-07-21 | PL001 | Muhammad Yusuf | Medan      |
+----------+------------+-------+----------------+------------+
4 rows in set (0.00 sec)
```

```
mysql> SELECT penjualan.nofaktur, fakturjual.tanggal, pelanggan.namaplg,
    -> barang.nama_barang, barang.harga, penjualan.jlhjual
    -> FROM penjualan, fakturjual, pelanggan, barang
    -> WHERE penjualan.nofaktur=fakturjual.nofaktur and
    -> fakturjual.kdplg=pelanggan.kdplg and penjualan.kodebrg=barang.kodebrg;
+----------+------------+----------------+-------------------+-------+---------+
| nofaktur | tanggal    | namaplg        | nama_barang       | harga | jlhjual |
+----------+------------+----------------+-------------------+-------+---------+
| FKR001   | 2006-07-20 | Muhammad Yusuf | Sabun LUX         |  3000 |       5 |
| FKR001   | 2006-07-20 | Muhammad Yusuf | Sabun Cuci        |  3000 |       3 |
| FKR001   | 2006-07-20 | Muhammad Yusuf | Pengharum Ruangan | 10000 |       1 |
| FKR001   | 2006-07-20 | Muhammad Yusuf | Korek Api         |   500 |      10 |
| FKR002   | 2006-07-20 | Doni Siregar   | Sabun LUX         |  3000 |       7 |
| FKR002   | 2006-07-20 | Doni Siregar   | Pepsodent 25 g    |  1000 |       4 |
| FKR002   | 2006-07-20 | Doni Siregar   | Pengharum Ruangan | 10000 |       2 |
| FKR003   | 2006-07-20 | Deni Sebayang  | Sabun LUX         |  3000 |       3 |
| FKR003   | 2006-07-20 | Deni Sebayang  | Sabun Cuci        |  3000 |       6 |
| FKR003   | 2006-07-20 | Deni Sebayang  | Lilin             |  4000 |       8 |
| FKR003   | 2006-07-20 | Deni Sebayang  | Penyedap Rasa     |  1000 |       2 |
| FKR003   | 2006-07-20 | Deni Sebayang  | Ikan Kaleng       |  4500 |       6 |
| FKR003   | 2006-07-20 | Deni Sebayang  | Coca Cola         |  2500 |       3 |
+----------+------------+----------------+-------------------+-------+---------+
13 rows in set (0.00 sec)

mysql> SELECT penjualan.nofaktur, fakturjual.tanggal, pelanggan.namaplg,
    -> count(*) as 'Jlh Jenis Brg',
    -> sum(penjualan.jlhbeli*barang.harga) as Total
    -> FROM penjualan, fakturjual, pelanggan, barang
    -> WHERE penjualan.nofaktur=fakturjual.nofaktur and
    -> fakturjual.kdplg=pelanggan.kdplg
    -> and penjualan.kodebrg=barang.kodebrg group by penjualan.nofaktur;
+----------+------------+----------------+---------------+-------+
| nofaktur | tanggal    | namaplg        | Jlh Jenis Brg | Total |
+----------+------------+----------------+---------------+-------+
| FKR001   | 2006-07-20 | Muhammad Yusuf |             4 | 39000 |
| FKR002   | 2006-07-20 | Doni Siregar   |             3 | 45000 |
| FKR003   | 2006-07-20 | Deni Sebayang  |             6 | 95500 |
+----------+------------+----------------+---------------+-------+
3 rows in set (0.03 sec)
```

BAB 9

SUBQUERY

Untuk menjalankan perintah subquery maka kita perlu membuat table-tabel seperti dibawah ini yakni :

- **Tabel infoprib**

```
+-------+---------------------+------------+------------+--------+
| nip   | nama                | kota       | tgl_lahir  | sex    |
+-------+---------------------+------------+------------+--------+
| 12345 | Dian Rahma          | Jakarta    | 1982-02-03 | Wanita |
| 12356 | Reza Najib          | Semarang   | 1981-11-23 | Pria   |
| 12367 | Mas Aditya          | Surabaya   | 1985-08-17 | Pria   |
| 12378 | Mhd Rowi            | Semarang   | 1978-05-17 | Pria   |
| 12389 | Andhini             | Bandung    | 1979-03-27 | Wanita |
| 12390 | Bagus Sigit Haryadi | Bandung    | 1987-09-09 | pria   |
| 12434 | Anggraini Anggun    | Yogyakarta | 1984-06-25 | wanita |
+-------+---------------------+------------+------------+--------+
```

- **Tabel Bagian**

```
+----------+-----------+
| kode_bag | nama_bag  |
+----------+-----------+
| 1        | Pemasaran |
| 2        | Produksi  |
| 3        | Akutansi  |
| 4        | SDM       |
| 5        | PDE       |
+----------+-----------+
```

- **Tabel Pekerjaan**

```
+-------+----------+---------+
| nip   | kode_bag | gaji    |
+-------+----------+---------+
| 12345 | 2        | 3000000 |
| 12356 | 1        | 1500000 |
| 12367 | 4        | 4500000 |
| 12378 | 3        | 3500000 |
| 12389 | 5        | 3000000 |
| 12390 | 1        | 1500000 |
| 12434 | 3        | 3500000 |
+-------+----------+---------+
```

9.1. Subquery Baris Tunggal

Subquery baris tunggal adalah subquery yang menghasilkan hanya satu baris. Biasanya subquery ini melibatkan fungsi-fungsi agregat. Contoh :

```
mysql> SELECT nip, gaji
    -> FROM pekerjaan
    -> WHERE gaji =
    -> (SELECT MIN(gaji) FROM pekerjaan);
+-------+---------+
| nip   | gaji    |
+-------+---------+
| 12356 | 1500000 |
| 12390 | 1500000 |
+-------+---------+
2 rows in set (0.00 sec)
mysql> SELECT nip, gaji
    -> FROM pekerjaan
```

```
    -> WHERE kode_bag =
    -> (SELECT kode_bag FROM bagian WHERE nama_bag='Akutansi');
+-------+---------+
| nip   | gaji    |
+-------+---------+
| 12378 | 3500000 |
| 12434 | 3500000 |
+-------+---------+
2 rows in set (0.00 sec)
```

9.2. Subquery Baris Berganda

Subquery baris beganda adalah subquery yang menghasilkan lebih dari satu baris. Hal seperti ini diimplementasikan dengan menggunakan operator IN, EXISTS, ANY dan ALL

Contoh :

```
mysql> SELECT nip, kode_bag
    -> FROM pekerjaan
    -> WHERE kode_bag IN
    -> (SELECT DISTINCT pekerjaan.kode_bag
    -> FROM pekerjaan, infoprib
    -> WHERE pekerjaan.nip = infoprib.nip AND infoprib.nama LIKE '%a');
+-------+----------+
| nip   | kode_bag |
+-------+----------+
| 12345 | 2        |
| 12367 | 4        |
+-------+----------+
2 rows in set (0.12 sec)

mysql> SELECT nip, kode_bag
    -> FROM pekerjaan
    -> WHERE kode_bag NOT IN
    -> (SELECT DISTINCT pekerjaan.kode_bag
    -> FROM pekerjaan, infoprib
    -> WHERE pekerjaan.nip = infoprib.nip AND infoprib.nama LIKE '%a');
+-------+----------+
| nip   | kode_bag |
+-------+----------+
| 12356 | 1        |
| 12378 | 3        |
| 12389 | 5        |
| 12390 | 1        |
| 12434 | 3        |
+-------+----------+
5 rows in set (0.00 sec)

mysql> SELECT nip, kode_bag FROM pekerjaan
    -> WHERE EXISTS
```

```
    -> (SELECT * FROM bagian
    -> WHERE kode_bag = pekerjaan.kode_bag);
+-------+----------+
| nip   | kode_bag |
+-------+----------+
| 12345 | 2        |
| 12356 | 1        |
| 12367 | 4        |
| 12378 | 3        |
| 12389 | 5        |
| 12390 | 1        |
| 12434 | 3        |
+-------+----------+
7 rows in set (0.01 sec)

mysql> SELECT nip, gaji
    -> FROM pekerjaan
    -> WHERE gaji > ANY
    -> (SELECT gaji FROM pekerjaan);
+-------+---------+
| nip   | gaji    |
+-------+---------+
| 12345 | 3000000 |
| 12367 | 4500000 |
| 12378 | 3500000 |
| 12389 | 3000000 |
| 12434 | 3500000 |
+-------+---------+
5 rows in set (0.00 sec)

mysql> SELECT nip, gaji
    -> FROM pekerjaan
    -> WHERE gaji < ALL
    -> (SELECT gaji FROM pekerjaan WHERE kode_bag='3');
+-------+---------+
| nip   | gaji    |
+-------+---------+
| 12345 | 3000000 |
| 12356 | 1500000 |
| 12389 | 3000000 |
| 12390 | 1500000 |
+-------+---------+
4 rows in set (0.00 sec)
```

9.3. Subquery Skalar

Subquery scalar adalah subquery yang menghasilkan satu nilai (satu kolom dan satu baris). Subquery seperti ini biasanya dapat diletakkan pada posisi kolom dalam suatu query ataupun sebagai ekspresi dan termasuk dalam CASE. Namun tidak dapat dikenakan pada klausa GROUP BY ataupun HAVING.

Contoh :

```
mysql> SELECT nip, kode_bag,
    -> CASE
    -> WHEN kode_bag = (SELECT kode_bag FROM bagian WHERE nama_bag =
'produksi')
    -> THEN 'produksi'
    -> ELSE 'non-produksi'
    -> END AS 'produksi?'
    -> FROM pekerjaan;
+-------+----------+--------------+
| nip   | kode_bag | produksi?    |
+-------+----------+--------------+
| 12345 | 2        | produksi     |
| 12356 | 1        | non-produksi |
| 12367 | 4        | non-produksi |
| 12378 | 3        | non-produksi |
| 12389 | 5        | non-produksi |
| 12390 | 1        | non-produksi |
| 12434 | 3        | non-produksi |
+-------+----------+--------------+
7 rows in set (0.00 sec)

mysql> SELECT nip,
    -> (SELECT nama_bag
    -> FROM bagian WHERE kode_bag = pekerjaan.kode_bag)
    -> AS 'nama bagian'
    -> FROM pekerjaan;
+-------+-------------+
| nip   | nama bagian |
+-------+-------------+
| 12345 | Produksi    |
| 12356 | Pemasaran   |
| 12367 | SDM         |
| 12378 | Akutansi    |
| 12389 | PDE         |
| 12390 | Pemasaran   |
| 12434 | Akutansi    |
+-------+-------------+
7 rows in set (0.00 sec)

mysql> SELECT nip,
    -> (SELECT nama_bag FROM bagian WHERE kode_bag = pekerjaan.kode_bag)
    -> AS 'nama bagian'
    -> FROM pekerjaan
    -> WHERE nip = ( SELECT nip FROM infoprib WHERE nip = pekerjaan.nip
AND sex = 'pria');
```

```
+-------+-------------+
| nip   | nama bagian |
+-------+-------------+
| 12356 | Pemasaran   |
| 12367 | SDM         |
| 12378 | Akutansi    |
| 12390 | Pemasaran   |
+-------+-------------+
4 rows in set (0.00 sec)
```

```
mysql> SELECT nip, nama FROM infoprib
    -> ORDER BY (SELECT kode_bag
    -> FROM pekerjaan
    -> WHERE nip = infoprib.nip);
```

```
+-------+--------------------+
| nip   | nama               |
+-------+--------------------+
| 12356 | Reza Najib         |
| 12390 | Bagus Sigit Haryadi |
| 12345 | Dian Rahma         |
| 12378 | Mhd Rowi           |
| 12434 | Anggraini Anggun   |
| 12367 | Mas Aditya         |
| 12389 | Andhini            |
+-------+--------------------+
7 rows in set (0.01 sec)
```

9.4. Subquery Pada Klausa HAVING

Subquery juga bisa diletakkan dalam klausa HAVING. Sebagai contoh, untuk menampilkan jumlah gaji pegawai per departemen khusus yang jumlah pegawainya lebih dari satu.

```
mysql> SELECT kode_bag, SUM(gaji)
    -> FROM pekerjaan p1
    -> GROUP BY kode_bag
    -> HAVING 1 < (SELECT COUNT(*) FROM pekerjaan p2 WHERE p1.kode_bag =
p2.kode_bag);
```

```
+----------+-----------+
| kode_bag | SUM(gaji) |
+----------+-----------+
| 1        |   3000000 |
| 3        |   7000000 |
+----------+-----------+
2 rows in set (0.07 sec)
```

BAB 10

PENGGABUNGAN DATA

10.1. Natural Join

Natural join adalah penggabungan data dari dua table yang didasarkan pada pada kolom dengan nama yang sama pada kedua table tersebut. Penggabungan ini mencerminkan hubungan antara foreign key dan primary key dalam dua table.

Contoh penggunaan NATURAL JOIN dapat diliat seperti berikut :

```
Mysql > SELECT nip, nama_bag FROM bagian NATURAL JOIN pekerjaan;
```

```
+-------+-----------+
| nip   | nama_bag  |
+-------+-----------+
| 12356 | Pemasaran |
| 12390 | Pemasaran |
| 12345 | Produksi  |
| 12378 | Akutansi  |
| 12434 | Akutansi  |
| 12367 | SDM       |
| 12389 | PDE       |
+-------+-----------+
7 rows in set (0.00 sec)
```

NATURAL JOIN pada prinsipnya dapat digunakan beberapa kali dalam sebuah pernyataan SELECT. Contoh berikut menunjukkan cara untuk mendapatkan informasi mengenai nama pegawai, nama bagian dan gaji.

```
mysql> SELECT nama, nama_bag, gaji FROM infoprib
    -> NATURAL JOIN pekerjaan
    -> NATURAL JOIN bagian;
```

```
+--------------------+-----------+---------+
| nama               | nama_bag  | gaji    |
+--------------------+-----------+---------+
| Dian Rahma         | Produksi  | 3000000 |
| Reza Najib         | Pemasaran | 1500000 |
| Mas Aditya         | SDM       | 4500000 |
| Mhd Rowi           | Akutansi  | 3500000 |
| Andhini            | PDE       | 3000000 |
| Bagus Sigit Haryadi | Pemasaran | 1500000 |
| Anggraini Anggun   | Akutansi  | 3500000 |
+--------------------+-----------+---------+
7 rows in set (0.00 sec)
```

10.2. Cross Join (Perkalian Kartesian)

Bentuk sederhana dari penggabungan dua buah table akan membentuk perkalian kartesian (Cartesian product) atau yang biasa dikenal dengan sebutan CROSS JOIN atau FULL JOIN. Hal ini diperoleh jika klausa WHERE tidak disebutkan. Sebagai contoh terdapat dua table sebagai berikut :

A	B
α	1
β	2

r

C	D	E
α	10	a
β	10	a
β	20	b
γ	10	b

s

Maka jika dikenakan CROSS JOIN maka menjadi :

A	B	C	D	E
α	1	α	10	a
α	1	β	10	a
α	1	β	20	b
α	1	γ	10	b
β	2	α	10	a
β	2	β	10	a
β	2	β	20	b
β	2	γ	10	b

Hal ini sama dengan perintah SQL sebagai berikut *SELECT * FROM r, s;*

Perhatiakan bahwa setiap nama pegawai akan dipasangkan dengan setiap nama bagian maka perintah SQL adalah sebagai berikut :

```
mysql> SELECT i.nama, b.nama_bag FROM infoprib i, bagian b;
```

Juga dapat ditulis sebagai berikut :

```
mysql> SELECT nama, nama_bag FROM infoprib CROSS JOIN bagian;
```

```
+--------------------+-----------+
| nama               | nama_bag  |
+--------------------+-----------+
| Dian Rahma         | Pemasaran |
| Dian Rahma         | Produksi  |
| Dian Rahma         | Akutansi  |
| Dian Rahma         | SDM       |
| Dian Rahma         | PDE       |
| Reza Najib         | Pemasaran |
| Reza Najib         | Produksi  |
| Reza Najib         | Akutansi  |
| Reza Najib         | SDM       |
| Reza Najib         | PDE       |
| Mas Aditya         | Pemasaran |
| Mas Aditya         | Produksi  |
| Mas Aditya         | Akutansi  |
| Mas Aditya         | SDM       |
| Mas Aditya         | PDE       |
| Mhd Rowi           | Pemasaran |
| Mhd Rowi           | Produksi  |
| Mhd Rowi           | Akutansi  |
| Mhd Rowi           | SDM       |
| Mhd Rowi           | PDE       |
| Andhini            | Pemasaran |
| Andhini            | Produksi  |
| Andhini            | Akutansi  |
| Andhini            | SDM       |
| Andhini            | PDE       |
| Bagus Sigit Haryadi | Pemasaran |
```

```
| Bagus Sigit Haryadi | Produksi  |
| Bagus Sigit Haryadi | Akutansi  |
| Bagus Sigit Haryadi | SDM       |
| Bagus Sigit Haryadi | PDE       |
| Anggraini Anggun    | Pemasaran |
| Anggraini Anggun    | Produksi  |
| Anggraini Anggun    | Akutansi  |
| Anggraini Anggun    | SDM       |
| Anggraini Anggun    | PDE       |
+---------------------+-----------+
35 rows in set (0.00 sec)
```

Pada MySQL, CROSS JOIN dapat diikuti dengan suatu kondisi penggabungan berbentuk :

- ON *kondisi*
- USING *(daftar kolom)*

Contoh berikut memperlihatkan penggunaan ON pada CROSS JOIN.

```
mysql> SELECT nip, nama_bag FROM pekerjaan
    -> CROSS JOIN bagian
    -> ON bagian.kode_bag=pekerjaan.kode_bag;
+-------+-----------+
| nip   | nama_bag  |
+-------+-----------+
| 12356 | Pemasaran |
| 12390 | Pemasaran |
| 12345 | Produksi  |
| 12378 | Akutansi  |
| 12434 | Akutansi  |
| 12367 | SDM       |
| 12389 | PDE       |
+-------+-----------+
7 rows in set (0.00 sec)
```

Contoh berikut memperlihatkan penggunaan USING pada CROSS JOIN.

```
mysql> SELECT nip, nama_bag FROM pekerjaan
    -> CROSS JOIN bagian USING(kode_bag);
+-------+-----------+
| nip   | nama_bag  |
+-------+-----------+
| 12356 | Pemasaran |
| 12390 | Pemasaran |
| 12345 | Produksi  |
| 12378 | Akutansi  |
| 12434 | Akutansi  |
| 12367 | SDM       |
| 12389 | PDE       |
+-------+-----------+
7 rows in set (0.00 sec)
```

10.3. Inner Join dan Outer Join

Equijoin sering dibedakan menjadi dua kategori yakni inner join dan outer join. Untuk melihat perbedaannya maka kita perhatikan contoh berikut ini :

```
mysql> SELECT bagian.nama_bag, pekerjaan.gaji FROM pekerjaan
    -> INNER JOIN bagian ON pekerjaan.kode_bag = bagian.kode_bag;
```

```
+-----------+---------+
| nama_bag  | gaji    |
+-----------+---------+
| Pemasaran | 1500000 |
| Pemasaran | 1500000 |
| Produksi  | 3000000 |
| SDM       | 4500000 |
| PDE       | 3000000 |
+-----------+---------+
5 rows in set (0.00 sec)
```

Nama bagian akutansi tidak ditampilkan karena data tersebut sudah tidak ada ditemukan pada table bagian.

Penggunaan lebih lanjut bisa menggunakan NATURAL LEFT JOIN atau NATURAL RIGHT JOIN. Pada persoalan di atas kita dapat menggunakan NATURAL RIGHT JOIN, pernyataannya sebagai berikut :

```
mysql> SELECT bagian.nama_bag, pekerjaan.nip, pekerjaan.gaji FROM bagian
    -> NATURAL RIGHT JOIN pekerjaan;
+-----------+-------+---------+
| nama_bag  | nip   | gaji    |
+-----------+-------+---------+
| Produksi  | 12345 | 3000000 |
| Pemasaran | 12356 | 1500000 |
| SDM       | 12367 | 4500000 |
| NULL      | 12378 | 3500000 |
| PDE       | 12389 | 3000000 |
| Pemasaran | 12390 | 1500000 |
| NULL      | 12434 | 3500000 |
+-----------+-------+---------+
7 rows in set (0.00 sec)
```

Alternatif lain yakni dengan menggunakan LEFT JOIN atau RIGHT JOIN, akan tetapi mesti mempergunakan ON dan USING untuk memuat suatu kondisi seperti halnya pada CROSS JOIN diatas. Contoh :

```
mysql> SELECT bagian.nama_bag, pekerjaan.gaji FROM pekerjaan
    -> LEFT JOIN bagian ON pekerjaan.kode_bag = bagian.kode_bag;
+-----------+---------+
| nama_bag  | gaji    |
+-----------+---------+
| Produksi  | 3000000 |
| Pemasaran | 1500000 |
| SDM       | 4500000 |
| NULL      | 3500000 |
| PDE       | 3000000 |
| Pemasaran | 1500000 |
| NULL      | 3500000 |
+-----------+---------+
7 rows in set (0.00 sec)

mysql> SELECT bagian.nama_bag, pekerjaan.gaji FROM pekerjaan
    -> LEFT JOIN bagian USING(kode_bag);
+-----------+---------+
| nama_bag  | gaji    |
+-----------+---------+
| Produksi  | 3000000 |
| Pemasaran | 1500000 |
| SDM       | 4500000 |
| NULL      | 3500000 |
```

```
| PDE       | 3000000 |
| Pemasaran | 1500000 |
| NULL      | 3500000 |
+-----------+---------+
7 rows in set (0.00 sec)
```

10.4. Operator Union

Operator UNION berguna untuk menggabungkan hasil dari dua buah query tanpa ad baris yang kembar. Sebagai contoh :

```
mysql> SELECT bagian.nama_bag, pekerjaan.nip, pekerjaan.gaji
    -> FROM bagian NATURAL RIGHT JOIN pekerjaan
    -> UNION
    -> SELECT bagian.nama_bag, pekerjaan.nip, pekerjaan.gaji
    -> FROM bagian NATURAL LEFT JOIN pekerjaan;
+-----------+-------+---------+
| nama_bag  | nip   | gaji    |
+-----------+-------+---------+
| Produksi  | 12345 | 3000000 |
| Pemasaran | 12356 | 1500000 |
| SDM       | 12367 | 4500000 |
| NULL      | 12378 | 3500000 |
| PDE       | 12389 | 3000000 |
| Pemasaran | 12390 | 1500000 |
| NULL      | 12434 | 3500000 |
+-----------+-------+---------+
7 rows in set (0.00 sec)
```

BAB 11
APLIKASI VIEW

11.1. Konsep View

View merupakan suatu bentuk representasi data yang dapat dibuat dengan melibatkan data yang ada pada satu atau beberapa table. Dengan menggunakan view, dimungkinkan untuk membuat hanya bagian tertentu dalam suatu table yang akan muncul.

Secara internal, view akan disimpan sebagai kamus data. Data yang sesungguhnya tidak ikut dicatat secara eksplisit, melainkan tetap mengacu pada table basisnya.

11.2. Membuat View

Untuk keperluan membuat view, MySQL menyediakan perintah CREATE VIEW. Contoh berikut memperlihatkan cara membuat view yang hanya melibatkan sebuah table.

```
mysql> CREATE VIEW info_umum AS
    -> SELECT nip, nama, tgl_lahir
    -> FROM infoprib;
Query OK, 0 rows affected (0.06 sec)
```

Dalam hal ini, **info_umum** adalah nama view yang dibuat. Sedangkan data yang terkandung didalamnya mencakup **nip**, **nama** dan **tanggal lahir** yang berasal dari table **infoprib**.

Untuk melihat hasilnya maka ketikkan perintah berikut ini :

```
mysql> SELECT * FROM info_umum;
+-------+---------------------+------------+
| nip   | nama                | tgl_lahir  |
+-------+---------------------+------------+
| 12345 | Dian Rahma          | 1982-02-03 |
| 12356 | Reza Najib          | 1981-11-23 |
| 12367 | Mas Aditya          | 1985-08-17 |
| 12378 | Mhd Rowi            | 1978-05-17 |
| 12389 | Andhini             | 1979-03-27 |
| 12390 | Bagus Sigit Haryadi | 1987-09-09 |
| 12434 | Anggraini Anggun    | 1984-06-25 |
+-------+---------------------+------------+
7 rows in set (0.03 sec)
```

11.3. Mengubah View

Bila kita telah membuat suatu view dan pada waktu berikutnya kita ingin melakukan perubahan, Kita bisa mengubah dengan menggunakan ALTER VIEW. Contohnya sebagai berikut :

```
mysql> ALTER VIEW info_umum AS
    -> SELECT nip, nama, sex
    -> FROM infoprib;
Query OK, 0 rows affected (0.05 sec)
```

```
mysql> SELECT * FROM info_umum;
+-------+---------------------+--------+
| nip   | nama                | sex    |
+-------+---------------------+--------+
| 12345 | Dian Rahma          | Wanita |
| 12356 | Reza Najib          | Pria   |
| 12367 | Mas Aditya          | Pria   |
| 12378 | Mhd Rowi            | Pria   |
| 12389 | Andhini             | Wanita |
| 12390 | Bagus Sigit Haryadi | pria   |
| 12434 | Anggraini Anggun    | wanita |
+-------+---------------------+--------+
7 rows in set (0.00 sec)
```

11.4. Membuat View dari sejumlah Table

Sebuah view tidak harus tersumber dari satu table. Jika dikehendaki, kita juga bisa membentuk view yang asalnya dari beberapa table. Contohnya :

```
mysql> CREATE VIEW info_peg AS
    -> SELECT infoprib.nip, infoprib.nama, bagian.nama_bag
    -> FROM infoprib, pekerjaan, bagian
    -> WHERE infoprib.nip = pekerjaan.nip AND
    -> pekerjaan.kode_bag = bagian.kode_bag;
Query OK, 0 rows affected (0.06 sec)

mysql> SELECT * FROM info_peg;
+-------+---------------------+-----------+
| nip   | nama                | nama_bag  |
+-------+---------------------+-----------+
| 12345 | Dian Rahma          | Produksi  |
| 12356 | Reza Najib          | Pemasaran |
| 12367 | Mas Aditya          | SDM       |
| 12389 | Andhini             | PDE       |
| 12390 | Bagus Sigit Haryadi | Pemasaran |
+-------+---------------------+-----------+
5 rows in set (0.00 sec)
```

11.5. Menghapus View

Suatu view dapat dihapus dengan menggunakan perintah DROP VIEW. Sebagai contoh penulisan sintaksnya adalah :

```
mysql> DROP VIEW info_peg;
Query OK, 0 rows affected (0.00 sec)
```

BAB 12
PROSEDUR DAN FUNGSI TERSIMPAN

12.1. Prosedur dan Fungsi Tersimpan

Prosedur tersimpan adalah suatu modul yang berisi kumpulan pernyataan SQL yang ditujukan untuk melaksanakan tugas tertentu dan terletak pada server. Modul ini bisa dipanggil oleh klien, sedangkan pengeksekusiannya dilakukan di server.

Keuntungan penggunaan prosedur tersimpan adalah sebagai berikut :

- ❖ Meningkatkan kinerja karena mengurangi pengiriman kode dari klien ke server.
- ❖ Tingkat keamanan yang tinggi karena pengakses data tertentu ditangani dalam server.
- ❖ Integritas data tinggi saat sejumlah aplikasi memanggil prosedur tersimpan yang sama.

Sebaliknya kelemahan yang terjadi sebagai akibat dari keuntungan-keuntungan tersebut, server akan terbebani, karena banyak proses yang harus ditangangi oleh server. Dalam MySQL terdapat istilah prosedur tersimpan dan fungsi tersimpan. Perbedaanya adalah :

- ❖ Prosedur tersimpan tidak menghasilkan nilai ketika dipanggil
- ❖ Fungsi tersimpan menghasilkan nilai ketika dipanggil dan tentu saja seperti fungsi biasa dipanggil di dalam suatu pernyataan.

Hal ini terbagi menjadi dalam tiga bagian seperti berikut :

```
DECLARE
   -- berisi deklarasi variable, konstanta,
   -- prosedur, ataupun fungsi
BEGIN
   -- berisi statemen-statemen yang akan dieksekusi
EXCEPTION
   -- berisi perintah untuk mengatasi kesalahan
   -- yang mungkin muncul
END;
```

12.2. Menciptakan Prosedur Tersimpan

Prosedur tersimpan diciptakan dengan penyataan CREATE PROCEDURE. Kaidah dasarnya adalah sebagai berikut :

```
CREATE PROCEDURE nama_prosedurtersimpan ([parameter_prosedur [,…]])
bagian_kode
```

Dalam hal ini:

- nama_prosedur tersimpan menyatakan nama prosedur tersimpan
- parameter_prosedur menyatakan definisi untuk parameter prosedur tersimpan

- bagian_kode berupa pernyataan-pernyataan SQL

Contoh membuat prosedur :

```
mysql> DELIMITER !
mysql> CREATE PROCEDURE jumpeg()
    -> BEGIN
    -> SELECT COUNT(*) AS 'Jumlah Pegawai'
    -> FROM infoprib;
    -> END
    -> !
Query OK, 0 rows affected (0.27 sec)
```

Untuk menjalankan prosedur yang ada pergunakan pernyataan CALL. Pernyataan ini memiliki bentuk sebagai berikut :

```
mysql> CALL jumpeg();
    -> !
+----------------+
| Jumlah Pegawai |
+----------------+
|              7 |
+----------------+
1 row in set (0.12 sec)
Query OK, 0 rows affected (0.13 sec)
```

12.3. Menciptakan Fungsi Tersimpan

Fungsi tersimpan diciptakan dengan penyataan CREATE FUNCTION. Kaidah dasarnya adalah sebagai berikut :

```
CREATE FUNCTION nama_fungsitersimpan ([parameter_fungsi [,…]])
    RETURN tipe
    bagian_kode
```

Dalam hal ini:

- nama_fungsi tersimpan menyatakan nama fungsi tersimpan
- parameter_fungsi menyatakan definisi untuk parameter fungsi tersimpan
- RETURN tipe menyatakan tipe nilai balik
- bagian_kode berupa pernyataan-pernyataan SQL

Contoh membuat fungsi sebagai berikut :

```
mysql> CREATE FUNCTION hobi(hobi1 varchar(40), hobi2 varchar(40), hobi3
       varchar(40), hobi4 varchar(40))
    -> returns varchar(200)
    -> begin
    -> return concat('Saya memiliki hobi sebagai berikut ',hobi1,' ,
       ',hobi2,' , ',hobi3,' dan ',hobi4);
```

```
    -> end;
      -> !
Query OK, 0 rows affected (0.00 sec)
```

Untuk menjalankan fungsi yang ada pergunakan pernyataan **SELECT nama_fungsi**. Pernyataan ini memiliki bentuk sebagai berikut :

```
mysql> SELECT hobi('Sepeda','Travelling','Baca','Berenang')!
+-----------------------------------------------------------------------------+
| hobi('Sepeda','Travelling','Baca','Berenang')                               |
+-----------------------------------------------------------------------------+
| Saya memiliki hobi sebagai berikut Sepeda , Travelling , Baca dan Berenang |
+-----------------------------------------------------------------------------+
1 row in set (0.00 sec)
```

12.4. Memperoleh Informasi Prosedur dan Fungsi Tersimpan

MySQL menyediakan perintah SHOW PROCEDURE dan SHOW FUNCTION untuk memperoleh informasi tentang prosedur dan fungsi tersimpan. Kaidah penulisan :

```
SHOW { PROCEDURE | FUNCTION } STATUS [ LIKE 'pola']
```

Contoh untuk menampilkan informasi prosedur tersimpan :

```
mysql> SHOW PROCEDURE STATUS!
+------------+---------------+-----------+----------------+---------------------+---------
------------+---------------+---------+----------------------+------------------
----+--------------------+
| Db         | Name          | Type      | Definer        | Modified            | Created
| Security_type | Comment | character_set_client | collation_connect
ion | Database Collation |
+------------+---------------+-----------+----------------+---------------------+---------
------------+---------------+---------+----------------------+------------------
----+--------------------+
| data       | jumpeg        | PROCEDURE | root@localhost | 2011-11-09 21:15:12 | 2011-11-
09 21:15:12 | DEFINER       |         | latin1               | latin1_swedish_ci
    | latin1_swedish_ci  |
| pratikumcs | addproduct    | PROCEDURE | root@localhost | 2011-10-26 20:44:57 | 2011-10-
26 20:44:57 | DEFINER       |         | latin1               | latin1_swedish_ci
    | latin1_swedish_ci  |
| pratikumcs | deleteproduct | PROCEDURE | root@localhost | 2011-10-26 20:53:28 | 2011-10-
26 20:53:28 | DEFINER       |         | latin1               | latin1_swedish_ci
    | latin1_swedish_ci  |
| pratikumcs | editproduct   | PROCEDURE | root@localhost | 2011-10-26 20:50:45 | 2011-10-
26 20:50:45 | DEFINER       |         | latin1               | latin1_swedish_ci
    | latin1_swedish_ci  |
+------------+---------------+-----------+----------------+---------------------+---------
------------+---------------+---------+----------------------+------------------
----+--------------------+
4 rows in set (0.11 sec)
```

Contoh untuk menampilkan informasi prosedur tersimpan dengan pola :

```
mysql> SHOW PROCEDURE STATUS LIKE 'j%'!
+------+--------+-----------+----------------+---------------------+---------------------
+---------------+---------+----------------------+----------------------+--------
------------+
| Db   | Name   | Type      | Definer        | Modified            | Created             |
Security_type | Comment | character_set_client | collation_connection | Databas
e Collation |
+------+--------+-----------+----------------+---------------------+---------------------
+---------------+---------+----------------------+----------------------+--------
```

```
------------+
| data | jumpeg | PROCEDURE | root@localhost | 2011-11-09 21:15:12 | 2011-11-09 21:15:12 |
DEFINER       |         | latin1               | latin1_swedish_ci    | latin1_
swedish_ci  |
+------+--------+-----------+----------------+---------------------+---------------------
+---------------+---------+----------------------+----------------------+--------
------------+
1 row in set (0.01 sec)
```

Contoh untuk menampilkan informasi fungsi tersimpan :

```
mysql> SHOW FUNCTION STATUS!
```

```
+------------+------+----------+----------------+---------------------+------------------
--+---------------+---------+----------------------+----------------------+-----
---------------+
| Db           | Name | Type      | Definer          | Modified               | Created
| Security_type | Comment | character_set_client | collation_connection | Data
base Collation |
+------------+------+----------+----------------+---------------------+------------------
--+---------------+---------+----------------------+----------------------+-----
---------------+
| pratikumcs | hobi | FUNCTION | root@localhost | 2011-11-10 10:44:17 | 2011-11-10
10:44:17 | DEFINER       |         | latin1               | latin1_swedish_ci    | lati
n1_swedish_ci  |
+------------+------+----------+----------------+---------------------+------------------
--+---------------+---------+----------------------+----------------------+-----
---------------+
1 row in set (0.01 sec)
```

Contoh untuk menampilkan informasi fungsi tersimpan dengan pola :

```
mysql> SHOW FUNCTION STATUS LIKE '%i'!
```

```
+------------+------+----------+----------------+---------------------+------------------
--+---------------+---------+----------------------+----------------------+-----
---------------+
| Db           | Name | Type      | Definer          | Modified               | Created
| Security_type | Comment | character_set_client | collation_connection | Data
base Collation |
+------------+------+----------+----------------+---------------------+------------------
--+---------------+---------+----------------------+----------------------+-----
---------------+
| pratikumcs | hobi | FUNCTION | root@localhost | 2011-11-10 10:44:17 | 2011-11-10
10:44:17 | DEFINER       |         | latin1               | latin1_swedish_ci    | lati
n1_swedish_ci  |
+------------+------+----------+----------------+---------------------+------------------
--+---------------+---------+----------------------+----------------------+-----
---------------+
1 row in set (0.01 sec)
```

12.5. Menghapus Prosedur dan Fungsi Tersimpan

Bila dikehendaki menghapus suatu prosedur atau fungsi tersimpan, kita bisa menggunakan pernyataan DROP PROCEDURE atau DROP FUNCTION. Kaidah penulisan pernyataan tersebut :

```
DROP {PROCEDURE|FUNCTION} {IF EXISTS} {nama_prosedur | nama fungsi }
```

Tanda { | } berarti yang ada didalamnya bisa berupa salah satu. Bagian IF EXISTS bersifat opsional artinya jika ada. Bila IF EXISTS disertakan, pernyataan tidak akan menimbulkan kesalahan walaupun prosedur dan fungsi tersimpan nama tidak pernah didefinisikan.

Contoh menghapus sebuah prosedur :

```
mysql> DROP PROCEDURE IF EXISTS jumpeg !
Query OK, 0 rows affected (0.00 sec)
```

Contoh menghapus sebuah fungsi :

```
mysql> DROP FUNCTION IF EXISTS order !
Query OK, 0 rows affected (0.00 sec)
```

BAB 13
TRIGGER

13.1. Konsep Trigger

Trigger adalah kumpulan pernyataan SQL yang dimaksudkan untuk dieksekusi oleh pernyataan INSERT, UPDATE, atau DELETE dengan tujuan untuk menjaga konsistensi data. Sebagai contoh, ketika kita menghapus suatu baris di suatu table, trigger mampu diatur agar data yang terkait di table lainnya juga ikut terhapus.

13.2. Fungsi dan Kelebihan

Fungsi dan kelebihan penggunaan trigger antara lain:

- Memperbaiki integritas data dengan membuat integrity constraint yang kompleks yang mana tidak mungkin ditangani oleh sintaks pembutan table.
- Memvalidasi transaksi data.
- Memperbaiki keamanan database dengan menyediakan audit yang lebih kompleks mengenai informasi perubahan database dan user siapa yang melakukan perubahan.

13.3. Menciptakan Trigger

Trigger diciptakan dengan menggunakan pernyataan CREATE TRIGGER. Kaidah penulisannya sebagai berikut :

```
CREATE {OR REPLACE} TRIGGER nama_trigger
     BEFORE|AFTER {INSERT|DELETE|UPDATE} Event ON nama_tabel
FOR EACH ROW
DECLARE
-- berisi deklarasi variabel
BEGIN
-- berisi statemen-statemen yang akan dieksekusi
END;
```

Kemungkinan Event yang dapat mengaktivasi sebuah trigger :

EVENT	KETERANGAN
INSERT	Trigger dijalankan ketika terdapat operasi penambahan sebuah baris data ke table bersangkutan
UPDATE	Trigger dijalankan ketika terdapat operasi pengubahan sebuah baris data ke table bersangkutan
DELETE	Trigger dijalankan ketika terdapat operasi penghapusan sebuah baris data ke table bersangkutan

Terkait dengan trigger terdapat alias dengan nama OLD dan NEW. Penulisannya adalah sebagai berikut :

```
NEW.nama_kolom
OLD.nama_kolom
```

Dalam hal ini :

- OLD.nama_kolom

  Menyatakan nilai nama_kolom sebelum dihapus atau diubah.

- NEW.nama_kolom

  Menyatakan nilai nama_kolom setelah diubah atau setelah data baru dimasukkan.

Contoh trigger dengan event INSERT :

```
mysql> DELIMITER !
mysql> CREATE TRIGGER dt_pekerjaan
    -> AFTER INSERT on infoprib
    -> FOR EACH ROW
    -> BEGIN
    -> INSERT INTO pekerjaan VALUE (NEW.nip,'1',0);
    -> END
    -> !
 Query OK, 0 rows affected (0.06 sec)

mysql> INSERT INTO infoprib
    -> VALUES ('10192','Teza Yanda','Jakarta','11112011','Pria')!
 Query OK, 1 row affected, 1 warning (0.00 sec)

mysql> SELECT * FROM pekerjaan WHERE nip = '10192';
    -> !
```

```
+-------+----------+------+
| nip   | kode_bag | gaji |
+-------+----------+------+
| 10192 | 1        |    0 |
+-------+----------+------+
1 row in set (0.01 sec)
```

Contoh trigger dengan event UPDATE :

```
mysql> CREATE TRIGGER ubah_dt
    -> BEFORE UPDATE on infoprib
    -> FOR EACH ROW
    -> BEGIN
    -> IF NEW.nip <> OLD.nip THEN
    -> SET NEW.nip = OLD.nip;
    -> END IF
    -> ;
    -> END;
    -> !
Query OK, 0 rows affected (0.07 sec)

mysql> UPDATE infoprib SET nama = 'Indra Atmajaya',tgl_lahir='19900530',
sex ='Pria'
    -> WHERE nip = '10192';!
Query OK, 1 row affected (0.01 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> SELECT * FROM infoprib WHERE nip='10192';!
+-------+----------------+---------+------------+------+
| nip   | nama           | kota    | tgl_lahir  | sex  |
+-------+----------------+---------+------------+------+
| 10192 | Indra Atmajaya | Jakarta | 1990-05-30 | Pria |
+-------+----------------+---------+------------+------+
1 row in set (0.01 sec)
```

Contoh trigger dengan event DELETE :

```
mysql> CREATE TRIGGER hapus_dt
    -> AFTER DELETE on infoprib
    -> FOR EACH ROW
    -> BEGIN
    -> DELETE FROM pekerjaan WHERE nip=OLD.nip;
    -> END;
    -> !
Query OK, 0 rows affected (0.19 sec)
```

```
mysql> DELETE FROM infoprib WHERE nip = '10192';
    -> !
Query OK, 1 row affected (0.06 sec)

mysql> SELECT * FROM pekerjaan;!
+-------+----------+---------+
| nip   | kode_bag | gaji    |
+-------+----------+---------+
| 12345 | 2        | 3000000 |
| 12356 | 1        | 1500000 |
| 12367 | 4        | 4500000 |
| 12378 | 3        | 3500000 |
| 12389 | 5        | 3000000 |
| 12390 | 1        | 1500000 |
| 12434 | 3        | 3500000 |
+-------+----------+---------+
7 rows in set (0.00 sec)
```

13.4. Melihat Daftar Trigger

Untuk melihat daftar trigger yang tersedia, Kita bisa memberikan perintah sebagai berikut :

```
SHOW Triggers;
```

13.5. Menghapus Trigger

Untuk menghapus trigger, gunakan perintah DROP TRIGGER. Kaidah penulisan dalam SQL adalah sebagai berikut :

```
DROP TRIGGER [nama_database.]nama_trigger
```

13.6. Batasan Trigger

Batasan yang berlaku bagi trigger :

- Tidak dapat melibatkan store procedure
- Tidak dapat melibatkan pernyataan SQL dinamis PREPARE
- Tidak dapat melibatkan pernyataan yang terkait dengan transaksi (START TRANSACTION, COMMIT, atau ROLLBACK)
- Tidak dibenarkan perintah SELECT tanpa INTO

BAB 14
MANAJEMEN USER

15.1. Anonym User

Anonym user adalah user tanpa identitas dan tanpa password. Pada saat anda mengakses SQL pertama kali seperti yang telah dijelaskan pada Modul-1 halaman 1, berati anda sudah menggunakan Anonym User. Ketiklah perintah berikut ini pada prompt mysql untuk melihat user dan password yang ada saat ini di tabel user.

```
mysql> use mysql;
Database changed

mysql> select user, host, password from user;
+------+-----------+----------+
| user | host      | password |
+------+-----------+----------+
| root | localhost |          |
|      | %         |          |
|      | localhost |          |
| root | %         |          |
+------+-----------+----------+
4 rows in set (0.02 sec)
```

Untuk tindakan pengamanan awal, kita harus menghapus semua user tanpa identitas tersebut dengan perintah DELETE :

```
mysql> delete from user where user='';
Query OK, 2 rows affected (0.01 sec)
```

Sekerang coba lihat hasilnya :

```
mysql> select user, host, password from user;
+------+-----------+----------+
| user | host      | password |
+------+-----------+----------+
| root | localhost |          |
| root | %         |          |
+------+-----------+----------+
2 rows in set (0.00 sec)
```

15.2. Memberi Password root

Anda dapat memberi password root (administrator) dengan perintah berikut ini :

```
mysql> update user set password=password('dedy')
    -> where user='root';

Query OK, 0 rows affected (0.00 sec)
Rows matched: 2  Changed: 0  Warnings: 0
```

Lanjutkan dengan perintah FLUSH untuk merefresh MySQL sebagai berikut :

```
mysql> flush privileges;
Query OK, 0 rows affected (0.03 sec)
```

Kemudian periksa hasilnya di tabel user sebagai berikut :

```
mysql> select user, host, password from user;
+------+-----------+------------------+
| user | host      | password         |
+------+-----------+------------------+
| root | localhost | 6da7a38c664d4838 |
| root | %         | 6da7a38c664d4838 |
+------+-----------+------------------+
2 rows in set (0.00 sec)
```

Untuk melihat hasil dari pengisian password di atas keluarlah dari MySQL dengan mengetik perintah berikut ini :

```
mysql> quit
Bye
C:\xampp\mysql\bin>
```

Kemudian coba masuk kembali ke MySQL dengan perintah-perintah berikut :

```
C:\xampp\mysql\bin>mysql
ERROR 1045: Access denied for user: 'ODBC@localhost' (Using password: NO)
```

```
C:\xampp\mysql\bin>mysql -u root
ERROR 1045: Access denied for user: 'root@localhost' (Using password: NO)
```

Pesan kesalahan pada perintah di atas terjadi karena MySQL telah dipassword. Pengguna tidak bisa masuk tanpa mengisi password terlebih dahulu. Untuk dapat masuk ke dalam MySQL ketik perintah berikut ini :

```
C:\xampp\mysql\bin>mysql -u root -p
Enter password: ****
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 4 to server version: 3.23.32-debug
Type 'help;' or '\h' for help. Type '\c' to clear the buffer
mysql>
```

15.3. Membuat User Baru

Anda dapat membuat user baru beserta izin aksesnya menggunakan perintah GRANT.

Sintak : **GRANT** *jenis_akses(nama_kolom)* **ON** *nama_database*

TO *nama_user* **IDENTIFIED BY** "*nama_password*"

[**WITH GRANT** *pilihan akses*]

Untuk menghapus izin akses user, tetapi tidak mengahapus seorang user secara permanen dapat menggunakan perintah REVOKE.

Sintak : **REVOKE** *jenis_akses* **ON** *nama_database*

FROM *nama_user*

Bila seorang user telah dihapus izin aksesnya dengan perintah REVOKE, dia tetap dapat masuk (*login*) ke dalam MySQL walaupun tidak dapat berbuat apa-apa. Untuk menghapus user secara permanen, gunakan perintah DELETE.

Untuk membuat beberapa user baru anda harus login sebagai root :

```
C:\xampp\mysql\bin>mysql -u root -h localhost -p
Enter password: ****
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 5 to server version: 3.23.32-debug

Type 'help;' or '\h' for help. Type '\c' to clear the buffer

mysql> grant all privileges on *.* to fadiyah@localhost
    -> identified by 'diah'
    -> with grant option;
Query OK, 0 rows affected (0.05 sec)
```

Perintah di atas membuat user baru dengan nama fadiyah dengan akses penuh.

```
mysql> grant all privileges on *.* to fadiyah@'%'
    -> identified by 'diah'
    -> with grant option;
Query OK, 0 rows affected (0.00 sec)
```

Perintah di atas sama seperti perintah berikut ini :

```
mysql> grant all privileges on *.* to fadiyah
    -> identified by 'diah'
    -> with grant option;
Query OK, 0 rows affected (0.00 sec)
```

Sekarang buat user baru dengan nama jono :

```
mysql> grant usage on *.* to jono@localhost
    -> identified by 'jono001';
Query OK, 0 rows affected (0.00 sec)
```

Si jono bisa masuk ke dalam MySQL dan bisa melihat database yang ada tetapi tidak bisa menggunakan database-database tersebut.

Ketik perintah berikut untuk melihat hak akses user di atas :

```
mysql> use mysql;
Database changed
```

```
mysql> select user, host, password from user;
+---------+-----------+-------------------+
| user    | host      | password          |
+---------+-----------+-------------------+
| root    | localhost | 6da7a38c664d4838 |
| fadiyah | %         | 6d67143458b40f7b |
| fadiyah | localhost | 6d67143458b40f7b |
| root    | %         | 6da7a38c664d4838 |
| jono    | localhost | 1408a83d497d0d9f |
+---------+-----------+-------------------+
5 rows in set (0.00 sec)
```

```
mysql> select user, select_priv, insert_priv, update_priv, delete_priv,
    -> create_priv, drop_priv from user;
+---------+-------------+-------------+-------------+-------------+-------------+-----------+
| user    | select_priv | insert_priv | update_priv | delete_priv | create_priv | drop_priv |
+---------+-------------+-------------+-------------+-------------+-------------+-----------+
| root    | Y           | Y           | Y           | Y           | Y           | Y         |
| fadiyah | Y           | Y           | Y           | Y           | Y           | Y         |
| fadiyah | Y           | Y           | Y           | Y           | Y           | Y         |
| root    | Y           | Y           | Y           | Y           | Y           | Y         |
| jono    | N           | N           | N           | N           | N           | N         |
+---------+-------------+-------------+-------------+-------------+-------------+-----------+
5 rows in set (0.00 sec)
```

```
mysql> select user, reload_priv, shutdown_priv, process_priv,
    -> file_priv, grant_priv from user;
+---------+-------------+---------------+--------------+-----------+------------
| user    | reload_priv | shutdown_priv | process_priv | file_priv | grant_priv
+---------+-------------+---------------+--------------+-----------+------------
| root    | Y           | Y             | Y            | Y         | Y
| fadiyah | Y           | Y             | Y            | Y         | Y
| fadiyah | Y           | Y             | Y            | Y         | Y
```

```
| root    | Y           | Y             | Y             | Y          | Y
| jono    | N           | N             | N             | N          | N
+---------+-------------+---------------+---------------+------------+------------
5 rows in set (0.00 sec)
```

```
mysql> select user, references_priv, index_priv, alter_priv
    -> from user;
+---------+-----------------+------------+------------+
| user    | references_priv | index_priv | alter_priv |
+---------+-----------------+------------+------------+
| root    | Y               | Y          | Y          |
| fadiyah | Y               | Y          | Y          |
| fadiyah | Y               | Y          | Y          |
| root    | Y               | Y          | Y          |
| jono    | N               | N          | N          |
+---------+-----------------+------------+------------+
5 rows in set (0.01 sec)
```

Ketiklah perintah-perintah dibawah ini untuk menguji hak akses user-user di atas :

```
mysql> quit
Bye
C:\xampp\mysql\bin>mysql -u fadiyah -h localhost -p
Enter password: ****
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 6 to server version: 3.23.32-debug

Type 'help;' or '\h' for help. Type '\c' to clear the buffer

mysql> use mysql;
Database changed
mysql> use dbpenjualan;
Database changed
```

Sekarang coba pakai user dengan nama jono :

```
mysql> exit
Bye
C:\xampp\mysql\bin>mysql -u jono -h localhost -p
Enter password: *******
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 7 to server version: 3.23.32-debug

Type 'help;' or '\h' for help. Type '\c' to clear the buffer
```

```
mysql> use mysql;
ERROR 1044: Access denied for user: 'jono@localhost' to database 'mysql'

mysql> use dbpenjualan;
ERROR 1044: Access denied for user: 'jono@localhost' to database 'dbpenjualan'

mysql> show databases;
 +-------------+
 | Database    |
 +-------------+
 | DBPenjualan |
 | Persediaan  |
 | mysql       |
 | test        |
 +-------------+
 4 rows in set (0.00 sec)

mysql> create database punyajono;
ERROR 1044: Access denied for user: 'jono@localhost' to database 'punyajono'
```

Dari perintah di atas tampak jelas perbedaan user yang memiliki akses penuh dengan user yang memiliki akses terbatas.

15.4. Memberi Izin Akses Tertentu

Untuk memberi izin akses SELECT, INSERT, UPDATE, dan DELETE kepada jono, yang dapat digunakan di dalam DBJono. **Masuk terlebih dahulu ke MySQL sebagai root** kemudian ketik perintah berikut :

```
C:\xampp\mysql\bin>mysql -u root -h localhost -p
Enter password: ****
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 9 to server version: 3.23.32-debug

Type 'help;' or '\h' for help. Type '\c' to clear the buffer

mysql> create database DBJono;
Query OK, 1 row affected (0.22 sec)
```

```
mysql> grant select, insert, update, delete
    -> on dbjono.*
    -> to jono@localhost;
Query OK, 0 rows affected (0.02 sec)

mysql> use mysql;
Database changed

mysql> select user, select_priv, update_priv, insert_priv,
    -> delete_priv, create_priv from db where user='jono';
+------+-------------+-------------+-------------+-------------+-------------+
| user | select_priv | update_priv | insert_priv | delete_priv | create_priv |
+------+-------------+-------------+-------------+-------------+-------------+
| jono | Y           | Y           | Y           | Y           | N           |
+------+-------------+-------------+-------------+-------------+-------------+
1 row in set (0.02 sec)

mysql> quit
Bye
```

Untuk melihat apakah jono sudah bisa menggunakan dbjono ketik perintah berikut :

```
C:\xampp\mysql\bin>mysql -u jono -h localhost -p
Enter password: *******
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 10 to server version: 3.23.32-debug

Type 'help;' or '\h' for help. Type '\c' to clear the buffer

mysql> use dbpenjualan;
ERROR 1044: Access denied for user: 'jono@localhost' to database 'dbpenjualan'

mysql> use dbjono;
Database changed

mysql> create table biodata
    -> (nama varchar(25) not null,
    -> alamat varchar(30),
    -> handphone char(12));
ERROR 1044: Access denied for user: 'jono@localhost' to database 'dbjono'
```

Pada contoh di atas jono masih belum bisa mengcreate sebuah tabel sebelum akses untuk itu dibuka oleh administrator (root). Ketik perintah berikut untuk keluar dari user jono dan masuk ke root :

```
mysql> exit
Bye

C:\xampp\mysql\bin>mysql -u root -h localhost -p
Enter password: ****
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 11 to server version: 3.23.32-debug

Type 'help;' or '\h' for help. Type '\c' to clear the buffer

mysql> grant create, drop
    -> on dbjono.*
    -> to jono@localhost;
Query OK, 0 rows affected (0.00 sec)

mysql> quit
Bye

C:\xampp\mysql\bin>mysql -u jono -h localhost -p
Enter password: *******
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 12 to server version: 3.23.32-debug

Type 'help;' or '\h' for help. Type '\c' to clear the buffer

mysql> use dbjono;
Database changed

mysql> create table biodata
    -> (nama varchar(25) not null,
    -> alamat varchar(30),
    -> handphone char(12));
Query OK, 0 rows affected (0.05 sec)

mysql> desc biodata;
```

```
+-----------+-------------+------+-----+---------+-------+
| Field     | Type        | Null | Key | Default | Extra |
+-----------+-------------+------+-----+---------+-------+
| nama      | varchar(25) |      |     |         |       |
| alamat    | varchar(30) | YES  |     | NULL    |       |
| handphone | varchar(12) | YES  |     | NULL    |       |
+-----------+-------------+------+-----+---------+-------+
3 rows in set (0.03 sec)
```

mysql> create table cobajono

-> (keterangan varchar(20));

```
Query OK, 0 rows affected (0.00 sec)
```

mysql> show tables;

```
+------------------+
| Tables_in_dbjono |
+------------------+
| biodata          |
| cobajono         |
+------------------+
2 rows in set (0.01 sec)
```

mysql> drop table cobajono;

```
Query OK, 0 rows affected (0.00 sec)
```

mysql> show tables;

```
+------------------+
| Tables_in_dbjono |
+------------------+
| biodata          |
+------------------+
1 row in set (0.00 sec)
```

mysql> insert into biodata values

-> ('Muhammad Rizki', 'Jl. Jermal X No. 48 Medan','-'),

-> ('Hardiansyah', 'Jl. Panglima Denai Medan', '081361787524');

```
Query OK, 2 rows affected (0.03 sec)
Records: 2  Duplicates: 0  Warnings: 0
```

mysql> select * from biodata;

```
+----------------+---------------------------+--------------+
| nama           | alamat                    | handphone    |
+----------------+---------------------------+--------------+
| Muhammad Rizki | Jl. Jermal X No. 48 Medan | -            |
| Hardiansyah    | Jl. Panglima Denai Medan  | 081361787524 |
+----------------+---------------------------+--------------+
2 rows in set (0.00 sec)
```

15.5. Menghapus Izin Akses User

Untuk menghapus izin akses seorang user dapat dilakukan dengan perintah-perintah berikut :

```
C:\xampp\mysql\bin>mysql -u root -h localhost -p
Enter password: ****
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 13 to server version: 3.23.32-debug
Type 'help;' or '\h' for help. Type '\c' to clear the buffer

mysql> revoke insert, select on dbjono.*
    -> from jono@localhost;
Query OK, 0 rows affected (0.00 sec)

mysql> exit
Bye
```

Masuk ke MySQL dengan user jono dan ketik perintah berikut untuk melihat apakah jono diperbolehkan atau tidak untuk menambah data dan menampilkannya :

```
C:\xampp\mysql\bin>mysql -u jono -h localhost -p
Enter password: *******
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 14 to server version: 3.23.32-debug
Type 'help;' or '\h' for help. Type '\c' to clear the buffer

mysql> use dbjono;
Database changed

mysql> select * from biodata;
ERROR 1044: Access denied for user: 'jono@localhost' to database 'dbjono'

mysql> insert into biodata values
    -> ('Jini','Jl. Janda No. 1 Medan','-');
ERROR 1044: Access denied for user: 'jono@localhost' to database 'dbjono'

mysql> update biodata set handphone='081361652341'
    -> where nama='Muhammad Rizki';
Query OK, 1 row affected (0.00 sec)
Rows matched: 1  Changed: 1  Warnings: 0
```

```
mysql> select * from biodata;
ERROR 1044: Access denied for user: 'jono@localhost' to database 'dbjono'

mysql> quit
Bye
```

Masuklah ke MySQL sebagai root kemudian hapus semua akases untuk user jono dan hapus user tersebut dengan mengetik perintah-perintah berikut ini:

```
C:\xampp\mysql\bin>mysql -u root -h localhost -p
Enter password: ****
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 15 to server version: 3.23.32-debug
Type 'help;' or '\h' for help. Type '\c' to clear the buffer

mysql> revoke create, drop on dbjono.*
    -> from jono@localhost;
Query OK, 0 rows affected (0.00 sec)

mysql> use mysql
Database changed

mysql> select user, select_priv, update_priv, insert_priv,
    -> delete_priv, create_priv from db where user='jono';
+------+-------------+-------------+-------------+-------------+-------------+
| user | select_priv | update_priv | insert_priv | delete_priv | create_priv |
+------+-------------+-------------+-------------+-------------+-------------+
| jono | N           | Y           | N           | Y           | N           |
+------+-------------+-------------+-------------+-------------+-------------+
1 row in set (0.00 sec)
```

Menghapus semua hak ases atas user jono :

```
mysql> revoke all privileges on dbjono.*
    -> from jono@localhost;
Query OK, 0 rows affected (0.00 sec)

mysql> select user, select_priv, update_priv, insert_priv,
    -> delete_priv, create_priv from db where user='jono';
Empty set (0.00 sec)
```

Menghapus user dengan nama jono :

```
mysql> delete from user where user='jono';
Query OK, 1 row affected (0.00 sec)

mysql> flush privileges;
Query OK, 0 rows affected (0.00 sec)

mysql> exit
Bye

C:\xampp\mysql\bin>mysql -u jono -h localhost -p
Enter password: *******
ERROR 1045: Access denied for user: 'jono@localhost' (Using password: YES)
```

User dengan nama jono sudah dihapus secara permanen di dalam tabel user. Bila jono berniat masuk ke MySQL maka aksesnya akan dibatalkan oleh MySQL.

BAB 15
MENGEKSPOR dan MENGIMPOR DATA

15.1. Mengekspor Data

Data yang ada didalam database dapat diambil dan disimpan dalam bentuk sebuah file. Ada dua cara yang dapat digunakan untuk mengekpor data tersebut. Pertama dengan menggunakan utilitas mysqldump dan kedua menggunakan pernyataan INSERT INTO OUTFILE.

a. Utilitas mysqldump

Utilitas ini bermanfaat untuk mengambil struktur di dalam suatu database beserta isinya. Kaidah penulisannya adalah sebagai berikut :

```
Mysqldump [OPSI] database [tabel]
```

Pernyataan yang berada di dalam [] bersifat opsional, artinya kaidah mysqldump tetap dipanggil walau tidak diikuti dengan argument sama sekali. Seperti contoh berikut ini :

```
c:\xampp\mysql\bin>mysqldump
Usage: mysqldump [OPTIONS] database [tables]
OR     mysqldump [OPTIONS] --databases [OPTIONS] DB1 [DB2 DB3...]
OR     mysqldump [OPTIONS] --all-databases [OPTIONS]
For more options, use mysqldump --help
```

Contoh pemakaian mysqldump dengan menyertakan argument adalah sebagai berikut ini :

```
c:\xampp\mysql\bin>mysqldump -u root -p data infoprib
Enter password:
```

Pada perintah di atas bermakna :

- **`-u root`** menyatakan bahwa user yang digunakan untuk koneksi ke database adalah root
- **`-p`** menyatakan bahwa ada password dan password akan dimasukkan ketika perintah mysqldump dieksekusi
- **`Data`** adalah nama dari database
- **`Infoprib`** adalah table yang akan diproses

Menghilangkan Pernyataan CREATE TABLE

Utilitas mysqldump yang menyertakan **`–t`** atau **`--no-create-info`** berguna untuk menghilangkan bagian pernyataan ALTER TABLE. Contoh :

```
c:\xampp\mysql\bin>mysqldump --no-create-info –u root –p data infoprib
Enter password:
```

Menghilangkan Pernyataan INSERT

Utilitas mysqldump yang menyertakan **–d** atau **––no-data** berguna untuk menghilangkan bagian pernyataan INSERT. Contoh :

```
c:\xampp\mysql\bin>mysqldump --no-data –u root –p data infoprib
Enter password:
```

Mengekspor Sebuah Database

Mengekpor isi sebuah database, hanya cukup menyebutkan database-nya saja. Contoh :

```
c:\xampp\mysql\bin>mysqldump –u root –p data infoprib
Enter password:
```

Dengan cara seperti ini, isi seluruh table akan diproses yang hasilnya akan diproses untuk pemakaian selanjutnya ke server MySQL, melalui perintah mysqlimport.

Menyimpan Hasil ke File

Semua contoh di atas membuat hasil mysqldump ditampilkan pada layar. Agar dapat disimpan ke dalam sebuah file, gunakan fitur redirection yang telah tersedia pada sistem operasi. Kaidah penulisannya adalah sebagai berikut :

```
Mysqldump –u root –p nama_database > letak_file/nama_file.sql
```

Contoh :

```
c:\xampp\mysql\bin>mysqldump –u root –p data > D:\data.sql
Enter password:
```

Contoh di atas menyatakan database dengan nama data akan diekspor ke direktori D:\.

b. SELECT INTO OUTFILE

Pernyataan SELECT INTO OUTFILE berguna untuk mengambil isi suatu table dan menyimpan ke file. Perintah ini membuat antarkolom dipisahkan oleh tab dan diakhiri dengan tanda akhir baris. Kaidah penulisannya adalah sebagai berikut :

```
SELECT * FROM nama_tabel ORDER BY parameter
INTO OUTFILE 'letak_file/nama_file'
```

Contoh :

```
mysql> SELECT * FROM bagian INTO OUTFILE '../../tmp/bagian.txt';
Query OK, 4 rows affected (0.02 sec)
```

Hasilnya adalah sebagai berikut :

```
C:\xampp\tmp>type bagian.txt
1       Pemasaran
2       Produksi
```

```
4       SDM
5       PDE
```

SELECT INTO OUTFILE dapat mengatur pemisah antar data, pemberian tanda kutip data, dan bahkan penentu pemisah antar baris. Contoh penulisannya :

```
mysql> SELECT * FROM bagian
    -> INTO OUTFILE '../../tmp/bagian1.txt'
    -> FIELDS TERMINATED BY ','
    -> ENCLOSED BY '"'
    -> LINES TERMINATED BY '\n';
Query OK, 4 rows affected (0.00 sec)
```

Penulisan sintak di atas memiliki keterangan sebagai berikut :

- **FIELDS TERMINATED BY ','** Menyatakan bahwa setiap data kolom diakhiri dengan koma
- **ENCLOSED BY '"'** Menyatakan bahwa data setiap kolom diberi tanda petik ganda
- **LINES TERMINATED BY '\n'** Menyatakan bahwa setiap baris diakhiri dengan '\n' yang berarti tanda akhir baris.

Hasilnya adalah sebagai berikut :

```
C:\xampp\tmp>type bagian1.txt
"1","Pemasaran"
"2","Produksi"
"4","SDM"
"5","PDE"
```

15.2. Mengimpor Data

Untuk melakukan impor data ke dalam suatu database, maka dapat menggunakan perintah dengan kaidah sebagai berikut :

```
Mysqldump [OPSI] database < nama_file
```

Contoh :

```
c:\xampp\mysql\bin>mysqldump –u root –p data < D:\bagian.sql
Enter password:
```

a. Impor Data dengan LOAD DATA INFILE

Pernyataan LOAD DATA INFILE berguna untuk membaca data dari suatu file dan meletakkannya ke dalam sebuah table. Kaidah penulisannya :

```
LOAD DATA INFILE nama_file INTO TABLE nama_tabel
```

Contoh 1 :

Terlebih dahulu pastikan pada table bagian tidak ada data yang tersimpan. Kemudian ketikkan pernyataan berikut ini :

```
mysql> LOAD DATA INFILE '../../tmp/bagian.txt'
    -> INTO TABLE bagian;
Query OK, 5 rows affected (0.00 sec)
Records: 5  Deleted: 0  Skipped: 0  Warnings: 0
```

```
mysql> SELECT * FROM bagian;
+----------+-----------+
| kode_bag | nam_bag   |
+----------+-----------+
| 1        | Pemasaran |
| 2        | Produksi  |
| 3        | Keuangan  |
| 4        | SDM       |
| 5        | PDE       |
+----------+-----------+
5 rows in set (0.00 sec)
```

Contoh 2 :

Buatlah file baru dengan nama bagbaru.txt. Isikan data tersebut seperti berikut :

```
mysql> LOAD DATA INFILE '../../tmp/bagbaru.txt'
    -> INTO TABLE bagian
    -> FIELDS TERMINATED BY ','
    -> ENCLOSED BY '"'
    -> LINES TERMINATED BY '\n';
Query OK, 5 rows affected (0.00 sec)
Records: 5  Deleted: 0  Skipped: 0  Warnings: 0
```

```
mysql> SELECT * FROM bagian;
+----------+-------------------+
| kode_bag | nam_bag           |
+----------+-------------------+
| 1        | Pemasaran         |
| 2        | Produksi          |
| 3        | Keuangan          |
| 4        | SDM               |
| 5        | PDE               |
| 9        | Corporate Service |
| 8        | Layanan Khusus    |
| 7        | Layanan Umum      |
| 6        | Administrasi      |
| 10       | Jaringan          |
+----------+-------------------+
10 rows in set (0.00 sec)
```

b. Impor Data dengan mysqlimport

Alternatif yang lain untuk mengimpor adalah dengan menggunakan utilitas mysqlimport. Kaidah penulisannya adalah sebagai berikut :

Mysqlimport [OPSI] database nama_file

Pada utilitas mysqlimpor ada beberapa opsi pilihan diantaranya adalah sebagai berikut :

OPSI	Keterangan
-u nama_pemakai	Nama user untuk koneksi ke database
-p	Menyatakan password untuk user
--delete atau -d	Table target akan dikosongkan terlebih dahulu
--ignore atau -i	Kalau ada kunci yang kembar, baris tersebut tidak diproses, tetapi baris berikutnya diproses
-local atau -L	Mysqlimport dilakukan dimesin local (bukan server)
-replace atau -r	Kalau ada kunci kembar, maka data digantikan dengan data dari file
--fields-terminated-by=karakter	Menentukan pemisah antar data
--force atau -f	Memaksa untuk terus menerus melakukan proses impor walau ada error
--lines-terminated-by=karakter	Menentukan pemisah baris
--fields-enclosed-by=karakter	Menentukan karakter yang mengapit data

Contoh :

```
c:\xampp\mysql\bin>mysqlimport -u root -p -d data bagian.txt
Enter password:
data.bagian: Records: 5  Deleted: 0  Skipped: 0  Warnings: 0
```

hasilnya sebagai berikut :

```
mysql> SELECT * FROM bagian;
+----------+-----------+
| kode_bag | nam_bag   |
+----------+-----------+
| 1        | Pemasaran |
| 2        | Produksi  |
| 3        | Keuangan  |
| 4        | SDM       |
| 5        | PDE       |
+----------+-----------+
5 rows in set (0.00 sec)
```

BAB 16

SOAL-SOAL LATIHAN

I. Latihan 1

1. Buatlah database dengan nama DBIndonesia
2. Buatlah table berikut ini :

TABEL PERSON

```
+------------+---------------+------+-----+---------+-------+
| Field      | Type          | Null | Key | Default | Extra |
+------------+---------------+------+-----+---------+-------+
| person_id  | varchar(10)   | NO   | PRI | NULL    |       |
| f_name     | varchar(35)   | YES  |     | NULL    |       |
| l_name     | varchar(40)   | YES  |     | NULL    |       |
| gender     | enum('M','F') | YES  |     | NULL    |       |
| birth_date | date          | YES  |     | NULL    |       |
| address    | varchar(50)   | YES  |     | NULL    |       |
| city_id    | smallint(11)  | YES  |     | NULL    |       |
| state_id   | tinyint(5)    | YES  |     | NULL    |       |
+------------+---------------+------+-----+---------+-------+
```

TABEL FAVORITE FOOD

```
+-------------+-------------+------+-----+---------+-------+
| Field       | Type        | Null | Key | Default | Extra |
+-------------+-------------+------+-----+---------+-------+
| person_id   | varchar(10) | NO   | PRI |         |       |
| food_id     | varchar(10) | NO   | PRI |         |       |
+-------------+-------------+------+-----+---------+-------+
```

TABEL FOOD

```
+---------+-------------+------+-----+---------+-------+
| Field   | Type        | Null | Key | Default | Extra |
+---------+-------------+------+-----+---------+-------+
| food_id | varchar(10) | YES  | PRI | NULL    |       |
| name    | varchar(25) | YES  |     | NULL    |       |
| price   | int(15)     | YES  |     | NULL    |       |
+---------+-------------+------+-----+---------+-------+
```

3. Isikan data-data tersebut sebanyak 5 record dengan cara satu per satu (data diberikan saat pratikum)
4. Tambahkan lagi data sebanyak 5 record dengan menginputkan data sekaligus (data diberikan saat pratikum)
5. Isikan semua data secara banyak ke table lain dengan nama table dt_person_copy, dt_favorite_copy, dan dt_food_copy

II. Latihan 2

1. Ubah struktur table person dengan menambahkan sebuah field dengan nama postal_code integer(10) yang diletakkan setelah state_id
2. Ubah field dengan nama gender menjadi sex dimanatipe datanya tetap
3. Hapuslah field postal_code dari table person kemudian tampilkan hasilnya
4. Hapuslah table dt_person_copy dan tampilkan hasilnya dilayar
5. Ubah beberapa baris pada table food kemudian tampilkan pada layar. Data yang diubah sebagai berikut :
   - ubah nama makanan Nasi Uduk menjadi Nasi Kuning
   - ubah harga soto medan menjadi 6000
6. Hapus data yang memiliki nama makanan ayam goring

III. Latihan 3

1. Tampilkan semua data siapa saja yang lahir tanggal 21-02-1990
2. Tampilkan nama depan dan nama belakang siapa saja yang lahir antara tanggal 01-01-1991 dan 02-02-1994
3. Tampilkan makanan apa saja yang harganya 10000
4. Tampilkan makanan apa saja yang harganya di atas 10000
5. Tampilkan makanan apa saja yang harganya di bawah 8000
6. Tampilkan makanan apa saja yang harganya di antara 13000 dan 20000
7. Tampilkan makanan yang harganya tidak sama dengan 10000
8. Tampilkan nama depan yang memiliki huruf pertama ‘i’
9. Tampilkan nama depan yang memiliki huruf pertama ‘a’ dan nama belakang yang memiliki huruf pertama ‘r’
10. Tampilkan nama makanan yang berakhiran dengan huruf ‘r’
11. Tampilkan makanan apa saja yang memiliki kandungan huruf ‘a’
12. Tampilkan makanan apa saja yang huruf pertamanya ‘m’ dengan harga di bawah 10000
13. Tampilkan makanan apa saja yang memiliki akhiran ‘g’ dengan harga di atas 7000

IV. Latihan 4

1. Tampilkan
   - harga rata-rata dari semua makanan yang ada
   - berapa orang yang memiliki makanan kesukaan dengan food_id=A11
   - makanan apa yang harganya paling murah
   - makanan apa yang hargany paling mahal
   - kalimat “Saya sedang makan mie aceh”
   - kalimat pada no 5 dengan huruf besar semua
   - semua nama makanan pada table food dengan huruf besar semua
   - semua nama makanan pada table food dengan huruf kecil semua
   - nama makanan dan panjang string dari table food
2. Ada sebuah password yakni “bangsaini33provinsi”, tampilkan enkripsi dari password tersebut
3. Tampilkan
   - pengulangan string “ini” sebanyak 10 kali
   - ubah karakter ‘a’ menjadi p dari nama makanan
   - string “Kaki Kuku Kakiku Kaku” dalam bentuk kebalikannya
   - nilai dari akar 81, 256, dan 3964081
   - nilai 92^{3}, 12^{17}, 82^{310}, dan 34^{81}
4. Tampilkan
   - Siapa saja yang memiliki nama depan dengan panjang karakter lebih dari 3
   - Kota mana saja yang memiliki panjang karakter lebih kurang dari 8
   - nama makanan yang huruf keduanya ‘o’

V. Latihan 5

1. Urutkan data person pada table person berdasarkan pada :
    - Nama depan
    - Nama belakang
    - Tanggal lahir
    - Nama belakang dan jenis kelamin
    - Nama depan dan tanggal lahir
    - Nama depan dengan susunan menaik
    - Tanggal lahir dengan susunan menurun
2. Tampilkan data makanan pada table food berdasarkan pada :
    - Nama makanan
    - Harga
    - Nama makanan tersususn menurun
    - Harga tersusun menaik
    - Nama makanan dan harga tersusun secara menaik
3. Tampilkan data makanan dengan syarat :
    - Hanya 3 data saja
    - Hanya 2 data saja dan dimulai dari baris ke 6

VI. Latihan 6

1. Tampilkan semua data orang dengan mengubah gender M menjadi P dan F menjadi W dengan memanfaatkan sintaks CASE
2. Tampilkan semua data makanan dengan menambah sebuah field keterangan untuk ditampilkan bersama data makanan tersebut. Dengan syarat Jika harga makanan di bawah 10000 maka tampilkan keterangan pada field keterangan “makanan ini murah dan terjangkau”. Sedangkan jika harga makanannya di atas 25000 maka tampilkan keterangan pada field keterangan “makanan ini agak mahal dan terjangkau”
3. Tampilkan nama depan dan gender yang memiliki nama alias gender_if_kondisi. Pada field gender jika nilainya P tampilkan dengan M dan jika bukan P tampilkan dengan W.
4. Carilah kemungkinan sintaks CASE dan IF yang dapat terbentuk dari table-tabel di atas selain kasus pada no 1 s/d no 3

VII. Latihan 7

1. Tampilkan semua data person yang tinggal di provinsi :
    - Sumatera Utara
    - Jawa Tengah dan Jawa Barat
    - DKI Jakarta
    - Nangroe Aceh Darussalam
2. Tampilkan nama depan dan tanggal lahir orang yang tinggal di kota :
    - Banda Aceh dan Medan
    - Semarang
    - Bandung dan Menyukai soto medan
    - Samarinda dan menyukai ayam goreng
    - Semarang dan menyukai soto makasar
3. Tampilkan nama-nama orang yang menyukai makanan :
    - Gado-gado
    - Ayam Goreng
    - Bakso dan Mie Ayam
    - Mie aceh atau Sroto Sokaraja
    - Tidak menyukai Sate Madura

VIII. Latihan 8

1. Buatlah sebuah prosedur tersimpan untuk melakukan :
    - Input data dalam table person, table favorite_food, dan table food
    - Pengubahan data yang mungkin dalam table person, table favorite_food dan table food
    - Penghapusan data dalam dalam table person, table favorite_food, dan table food
2. Merujuk dari soal no 1, jalankan fungsi pemanggilan procedure tersimpan tersebut
3. Buatlah sebuah prosedur tersimpan yang jika dijalankan fungsi pemanggilan prosedur tersimpan menghasilkan jumlah orang pada table person
4. Buatlah sebuah fungsi tersimpan yang menampilkan “saya menyukai makanan1, makanan2, makanan3 dan makanan4”. Sehingga ketika fungsi tersimpan dijalankan akan keluar seperti “Saya menyukai Bakso, Mie Aceh, Martabak, dan Soto”
5. Buatlah sebuah prosedur tersimpan dengan nama coba_if dimana pemanggilan prosedur tersimpan akan dimasukkan dua parameter x dan y. Jika x lebih kecil dari y maka akan tampil pesan dilayar bahwa y adalah nilai besar. Sedangkan jika y lebih besar dari x makan akan tampil pesan x adalah nilai besar.
6. Buatlah sebuah prosedur untuk menentukan nilai kelulusan mahasiswa dengan ketentuan :
    - Predikat A nilai lebih besar sama dengan 90
    - Predikat B nilai antara 70 dan lebih kecil 90
    - Predikat C nilai antara 60 dan lebih kecil 70
    - Predikat D nilai antara 50 dan lebih kecil 60
    - Predikat E nilai antara 0 dan lebih kecil 50
7. Hapuslah salah satu prosedur tersimpan yang telah Anda buat
8. Lakukan pemanggilan prosedur yang telah Anda buat

IX. Latihan 9

1. Buatlah sebuah trigger yang mungkin terbentuk dari 5 table di atas dengan memanfaatkan event trigger berikut :
   - Setelah input data person maka person_id akan terinput juga pada table favorite_food
   - Sebelum mengubah person_id di table person, lakukan pengecekan terhadap person_id agar tidak dapat diubah
   - Setelah menghapus data person maka hapuslah data-data di table lain yang memiliki keterkaitan dengan table person
2. Tampilkan trigger apa saja yang telah anda buat
3. Hapuslah salah satu trigger yang telah anda buat
4. Carilah kemungkinan trigger yang dapat terbentuk dari table-table di atas selain trigger pada soal no 1

DAFTAR PUSTAKA

Kadir,Abdul. 2008. *Tuntunan Praktis Belajar Database menggunakan MySQL*. Yogyakarta : Penerbit Andi

Beaulieu, Alan. *Learning SQL, 2nd edition*. O'Reilly Media,2009

Andy Oppel and Robert Sheldon. *SQL A beginer's Guide, 3rd edition*. Mc Graw-Hill, 2009

Politeknik Telkom.TT. *Modul Sistem Manajemen Basis Data.* Bandung : Politeknik Telkom