MODUL BASIS DATA – DASAR BASIS DATA 2

A. MANIPULASI DAN RETRIEVE DATA (DML LANJUTAN)

SELECT merupakan instruksi yang berfungsi untuk memilih spesifik kolom dari satu atau beberapa tabel. Secara umum instruksi SELECT adalah sebagai berikut : `SELECT kolom1, kolom2, kolom-n,… FROM nama_tabel WHERE predikat/kondisi`

Jika klausa WHERE tidak digunakan atau dituliskan, maka record atau data yang dimunculkan dari hasil seleksi adalah seluruh Data dalam tabel. Predikat atau kondisi yang diberikan setelah klausa WHERE menyatakan syarat yang harus dipenuhi untuk menampilkan suatu data tertentu. Bentuk umum klausa WHERE adalah sebagai berikut : `WHERE kolom <operator> <nilai>`

Beberapa operator yang berlaku meliputi :

No.	Operator	Arti
1.	=	Sama dengan
2.	< >	Tidak sama, atau dapat juga !=
3.	<	Kurang dari
4.	<=	Kurang dari sama dengan
5.	>	Lebih dari
6.	>=	Lebih dari sama dengan

Selain menggunakan operator di atas, klausa WHERE juga dapat digabungkan dengan menggunakan logika AND, OR, dan NOT. Hal ini bertujuan untuk menggabungkan lebih dari satu kondisi maupun negasi. SELECT * merupakan karakter khusus yang menyatakan bahwa kolom yang akan dipilih adalah seluruh kolom yang terdapat pada tabel tersebut.

1) Memberi nama pada kolom

```
SELECT nama_kolom_lama AS nama_kolom_baru FROM namatabel
```

Contoh:

```
    select  asal_kota as kota from pegawai;
```

```
MariaDB [PEGW]> select asal_kota as kota from pegawai;
+----------+
| kota     |
+----------+
| Kendari  |
| Bandung  |
| Kendari  |
| Surabaya |
+----------+
4 rows in set (0.00 sec)
```

2) Menggunakan alias untuk nama tabel

```
SELECT nama_alias .namakolom1, nama_alias .namakolom2 FROM
namatabel nama_alias;
```

Contoh:

```
select p.nama_peg, p.alamat from pegawai p;
```

```
MariaDB [PEGW]> select p.nama_peg, p.alamat from pegawai p;
+----------+--------------------+
| nama_peg | alamat             |
+----------+--------------------+
| Alana    | Jl. Bunga Matahari |
| Andini   | Jl. Sao-sao        |
| Taqy     | Kampus Baru        |
| Emil     | Jl. Balaikota      |
+----------+--------------------+
4 rows in set (0.01 sec)
```

3) Nested quieries/subquery (IN, NOT IN, EXISTS, NOT EXISTS)

Subquery adalah query di dalam query. Dengan subquery, hasil dari query akan menjadi bagian dari query di atasnya. Subquery dapat diaplikasikan dengan menggunakan klausa WHERE atau HAVING. Klausa WHERE digunakan untuk memilih baris-baris tertentu, sedangkan klausa HAVING, subquery digunakan untuk menyeleksi kelompok baris yang akan digunakan/ditampilkan.

- Contoh IN:

```
select nama_peg,iddiv from pegawai where asal_kota IN
("Bandung","Surabaya");
```

```
MariaDB [PEGW]> select nama_peg,iddiv from pegawai where asal_kota IN ("Bandung","Surabaya");
+----------+-------+
| nama_peg | iddiv |
+----------+-------+
| Andini   | KEU   |
| Emil     | PEG   |
+----------+-------+
2 rows in set (0.00 sec)
```

- Contoh NOT IN:

```
select nama_peg,iddiv from pegawai where asal_kota NOT IN
("Bandung");
```

```
MariaDB [PEGW]> select nama_peg,iddiv from pegawai where asal_kota NOT IN ("Bandung");
+----------+-------+
| nama_peg | iddiv |
+----------+-------+
| Alana    | KEU   |
| Taqy     | HUM   |
| Emil     | PEG   |
+----------+-------+
3 rows in set (0.00 sec)
```

❖ Contoh EXISTS:

```
select * from pegawai as p where exists
    -> (select * from gaji as g where p.npp=g.npp);
```

```
MariaDB [PEGW]> select * from pegawai as p where exists
    -> (select * from gaji as g where p.npp=g.npp);
+------+----------+--------------------+-------+-----------+
| npp  | nama_peg | alamat             | iddiv | asal_kota |
+------+----------+--------------------+-------+-----------+
| 1101 | Alana    | Jl. Bunga Matahari | KEU   | Kendari   |
| 1102 | Andini   | Jl. Sao-sao        | KEU   | Bandung   |
| 1103 | Taqy     | Kampus Baru        | HUM   | Kendari   |
+------+----------+--------------------+-------+-----------+
3 rows in set (0.00 sec)
```

❖ Contoh NOT EXISTS:

```
select * from pegawai as p where  not exists
    -> (select * from gaji as g where p.npp=g.npp);
```

```
Database changed
MariaDB [pegw]> select * from pegawai as p where  not exists
    -> (select * from gaji as g where p.npp=g.npp);
+------+----------+---------------+-------+-----------+
| npp  | nama_peg | alamat        | iddiv | asal_kota |
+------+----------+---------------+-------+-----------+
| 1104 | Emil     | Jl. Balaikota | PEG   | Surabaya  |
+------+----------+---------------+-------+-----------+
1 row in set (0.00 sec)
```

4) Penyusunan (Sort) menggunakan ORDER BY

ORDER BY digunakan untuk mengurutkan data berdasarkan kolom tertentu sesuai dengan tipe yang dimiliki.

Contoh ORDER BY:

```
select nama_peg, asal_kota from pegawai order by nama_peg;
```

```
MariaDB [pegw]> select nama_peg, asal_kota from pegawai order by nama_peg;
+----------+-----------+
| nama_peg | asal_kota |
+----------+-----------+
| Alana    | Kendari   |
| Andini   | Bandung   |
| Emil     | Surabaya  |
| Taqy     | Kendari   |
+----------+-----------+
4 rows in set (0.00 sec)
```

Dapat juga digunakan ASC dan DESC

❖ Contoh ASC:

```
select nama from divisi order by nama ASC;
```

```
MariaDB [pegw]> select nama from divisi order by nama ASC;
+------------+
| nama       |
+------------+
| HUMAS      |
| KEPEGAWAIA |
| KEUANGAN   |
+------------+
3 rows in set (0.00 sec)
```

❖ Contoh DESC:

```
SELECT NAMA_PEG FROM PEGAWAI ORDER BY NAMA_PEG DESC;
```

```
MariaDB [pegw]> SELECT NAMA_PEG FROM PEGAWAI ORDER BY NAMA_PEG DESC;
+----------+
| NAMA_PEG |
+----------+
| Taqy     |
| Emil     |
| Andini   |
| Alana    |
+----------+
4 rows in set (0.00 sec)
```

5) DISTINCT

Distinct digunakan untuk menampilkan data yang tunggal (tidak berulang).

Contoh DISTINCT:

```
select distinct iddiv from pegawai;
```

```
MariaDB [pegw]> select distinct iddiv from pegawai;
+-------+
| iddiv |
+-------+
| KEU   |
| HUM   |
| PEG   |
+-------+
3 rows in set (0.00 sec)
```

6) BETWEEN dan NOT BETWEEN

Digunakan untuk menyeleksi data berdasarkan range nilai tertentu.

Contoh:

```
select npp, total_gaji from gaji where total_gaji
    -> between 1400000 and 1500000;
```

```
MariaDB [pegw]> select npp, total_gaji from gaji where total_gaji
    -> between 1400000 and 1500000;
+------+------------+
| npp  | total_gaji |
+------+------------+
| 1101 |    1500000 |
| 1102 |    1500000 |
| 1105 |    1400000 |
+------+------------+
3 rows in set (0.00 sec)
```

7) LIKE dan NOT LIKE

Digunakan untuk mencari teks berdasarkan prefix (kata depan), sufix (kata akhir) atau kata tengah.

Contoh:

```
select npp, nama_peg from pegawai
    -> where asal_kota like 'surabaya';
```

```
MariaDB [pegw]> select npp, nama_peg from pegawai
    -> where asal_kota like 'surabaya';
+------+----------+
| npp  | nama_peg |
+------+----------+
| 1104 | Emil     |
+------+----------+
1 row in set (0.00 sec)
```

```
select npp, nama_peg, asal_kota from pegawai
    -> where asal_kota LIKE 'B%';
```

```
MariaDB [pegw]> select npp, nama_peg, asal_kota from pegawai
    -> where asal_kota LIKE 'B%';
+------+----------+-----------+
| npp  | nama_peg | asal_kota |
+------+----------+-----------+
| 1102 | Andini   | Bandung   |
+------+----------+-----------+
1 row in set (0.00 sec)
```

```
select npp, nama_peg, asal_kota from pegawai
    -> where asal_kota LIKE '%DA%';
```

```
MariaDB [pegw]> select npp, nama_peg, asal_kota from pegawai
    -> where asal_kota LIKE '%DA%';
+------+----------+-----------+
| npp  | nama_peg | asal_kota |
+------+----------+-----------+
| 1101 | Alana    | Kendari   |
| 1103 | Taqy     | Kendari   |
+------+----------+-----------+
2 rows in set (0.00 sec)
```

B. MANIPULASI DAN RETRIEVE DATA LANJUTAN (DML)

1) UNION

UNION merupakan operator atau perintah yang digunakan untuk ***menggabungkan*** hasil query atau isi data dari 2 (dua) tabel atau lebih dengan ketentuan jumlah, nama dan tipe data atau kolom dari masing – masing tabel yang akan ditampilkan datanya harus sama.

Perintah ini terdiri dari dua jenis yaitu UNION dan UNION ALL. Untuk menghasilkan suatu data set perintah UNION harus disisipkan diantara perintah SELECT. Untuk lebih jelasnya kita ambil contoh sederhana dari sekumpulan data sebagai berikut :

❖ Contoh UNION:

```
select iddiv from divisi union select iddiv from pegawai;
```

```
MariaDB [pegw]> select iddiv from divisi union select iddiv from pegawai;
+-------+
| iddiv |
+-------+
| HUM   |
| KEU   |
| PEG   |
+-------+
3 rows in set (0.25 sec)
```

❖ Contoh UNION ALL:

Berbeda dengan perintah UNION, perintah UNION ALL menggabungkan keseluruhan data walaupun data tersebut memiliki isi yang sama.

```
select iddiv from divisi union all select iddiv from pegawai;
```

```
MariaDB [pegw]> select iddiv from divisi union all select iddiv from pegawai;
+-------+
| iddiv |
+-------+
| HUM   |
| KEU   |
| PEG   |
| KEU   |
| KEU   |
| HUM   |
| PEG   |
+-------+
7 rows in set (0.00 sec)
```

2) ARITMATIKA dan FUNGSI AGREGAT

a. ARITMATIKA

Perhitungan aritmatika pada suatu database dapat dilakukan dengan menggunakan * / + - seperti pada umumnya.

Contoh, menghitung gaji dosen dengan menaikkan gaji sebesar 1.5 kali dari gaji:

```
select npp,total_gaji * 1.5 AS "Salary up" from gaji;
```

```
MariaDB [pegw]> select npp,total_gaji * 1.5 AS "Salary up" from gaji;
+------+-----------+
| npp  | Salary up |
+------+-----------+
| 1101 | 2250000.0 |
| 1102 | 2250000.0 |
| 1103 | 1950000.0 |
| 1105 | 2100000.0 |
+------+-----------+
4 rows in set (0.06 sec)

MariaDB [pegw]> select * from gaji;
+------+------------+
| npp  | total_gaji |
+------+------------+
| 1101 |    1500000 |
| 1102 |    1500000 |
| 1103 |    1300000 |
| 1105 |    1400000 |
+------+------------+
4 rows in set (0.00 sec)
```

b. Fungsi AGREGAT

- Count

Merupakan perintah atau fungsi yang digunakan untuk menghitung jumlah baris suatu kolom (record) pada tabel. Contoh:

```
select count(nama_peg) as 'Jumlah Pegawai' from pegawai;
```

```
MariaDB [pegw]> select count(nama_peg) as 'Jumlah Pegawai' from pegawai;
+-----------------+
| Jumlah Pegawai |
+-----------------+
|               4 |
+-----------------+
1 row in set (0.04 sec)
```

- SUM

Merupakan fungsi yang digunakan untuk menghitung jumlah nilai suatu kolom pada tabel. Perintah umumnya adalah sebagai berikut:

```
SELECT SUM(namakolom) FROM namatabel;
```

```
select sum(total_gaji) from gaji;
```

```
MariaDB [pegw]> select sum(total_gaji) from gaji;
+-----------------+
| sum(total_gaji) |
+-----------------+
|         5700000 |
+-----------------+
1 row in set (0.00 sec)
```

- AVG

Merupkan fungsi yang digunakan untuk menghitung nilai rata – rata suatu kolom pada tabel. Perintah umum untuk menghitung rata – rata adalah sebagai berikut :

```
SELECT AVG(namakolom) FROM namatabel;
```

```
select AVG(total_gaji) from gaji;
```

```
MariaDB [pegw]> select AVG(total_gaji) from gaji;
+-----------------+
| AVG(total_gaji) |
+-----------------+
|    1425000.0000 |
+-----------------+
1 row in set (0.00 sec)
```

- MIN

Merupakan fungsi untuk menampilkan nilai terkecil dari suatu kolom pada tabel. Syntax umumnya adalah sebagai berikut:

```
SELECT MIN(namakolom) FROM namatabel;
```

```
select min(total_gaji) from gaji;
```

```
MariaDB [pegw]> select min(total_gaji) from gaji;
+-----------------+
| min(total_gaji) |
+-----------------+
|         1300000 |
+-----------------+
1 row in set (0.02 sec)
```

- MAX

Merupakan fungsi untuk menampilkan nilai terbesar dari suatu kolom pada tabel. Syntax umumnya adalah sebagai berikut:

```
SELECT MAX(namakolom) FROM namatabel;
```

```
select MAX(total_gaji) from gaji;
```

```
MariaDB [pegw]> select MAX(total_gaji) from gaji;
+-----------------+
| MAX(total_gaji) |
+-----------------+
|         1500000 |
+-----------------+
1 row in set (0.00 sec)
```

Daftar Pustaka

MODUL PRAKTIKUM BASIS DATA STT TELKOM, 2014, Didi Supriyadi, ST., M.Kom