MODUL IV

SQL

Tujuan :

Mampu memahami dan membuat perintah SQL lebih dalam dengan berbagai kondisi, Ekpresi, pengurutan dan mampu mengabungkan beberapa argument / perintah pada beberapa tabel (join) dalam basis data.

A. Dasar Teori

a. SQL

Secara umum perintah-perintah yang terdapat di dalam SQL, diklasifikasikan menjadi tiga bagian, antara lain yaitu :

1. DDL (Data Definition Language)
   - Merupakan perintah SQL yang berkaitan dengan pendefinisian suatu struktur database, dalam hal ini database dan table.
   - Perintah DDL adalah: CREATE, ALTER, RENAME, DROP
2. DML (Data Manipulation Language)
   - Merupakan perintah SQL yang berkaitan dengan manipulasi atau pengolahan data atau record dalam table.
   - Perintah DML antara lain: SELECT, INSERT, UPDATE, DELETE
3. DCL (Data Control Language)
   - Merupakan perintah SQL yang berkaitan dengan manipulasi user dan hak akses (priviledges).
   - Perintah SQL yang termasuk dalam DCL antara lain: GRANT, REVOKE.

b. SELECT

SELECT merupakan salah satu pondasi dalam SQL Programming. SELECT digunakan untuk menampilkan data, terlebih untuk mencari informasi dalam kumpulan data.

Sintak

SELECT dibagi kedalam 6 komponen, antara lain:

a) SELECT. Diikuti oleh **<select_list>,** dapat berupa literal_value atau column_list atau asterisk (*).

b) FROM. Diikuti oleh **<table_name>** sesuai dengan column_list. Jadi jika ada

data yang diambil dari kolom tertentu, harus diketahui kolom tersebut diambil dari tabel mana. Tabel pada FROM dapat diikuti dengan alias untuk mempermudah penulisan khususnya ketika join dan subquery.

c) WHERE. Diikuti oleh kondisi secara umum.

d) GROUP BY. Diikuti oleh **<select_list>.** Bagian ini muncul ketika ada fungsi-fungsi agregasi.

e) HAVING. Diikuti oleh kondisi hanya untuk fungsi -fungsi agregasi.

f) ORDER BY. Diikuti oleh <select_list>

Perintahnya :

```
SELECT     <select_list>
[FROM      <table_name>]
[WHERE     <kondisi1> [AND/OR <kondisi2>]]
[GROUP BY  <select_list>]
[HAVING    <kondisi1> [AND/OR <kondisi2>]]
[ORDER BY  <select_list>]
```

Keterangan :

- SELECT, INTO, FROM, WHERE, GROUP BY, HAVING DAN ORDER BY → kata kunci (keyword) yang harus disertakan jika kita membutuhkannya di dalam pengolahan data.
- select_list, table_source, search_condition, group_by_expression, order_expression → isian yang bisa kita ubah berdasarkan kebutuhan kita
- Kurung kotak [ ] → bagian tersebut boleh disertakan atau tidak, tergantung dari kebutuhan

c. Menyaring data

Tidak semua data yang ada pada tabel, ingin ditampilkan. Terlebih ketika tabel terbagi kedalam banyak kolom dengan jumlah data yang sangat besar.

Operator Pembanding :

Operator	Keterangan
=	Sama dengan
>	Lebih besar dari
>=	Lebih besar sama dengan
<	Kurang dari
<=	Kurang dari sama dengan
<> atau !=	Tidak sama dengan
BETWEEN .. AND ..	Diantara 2 nilai
IN (set)	Cocok dengan salah satu diantara daftar nilai
LIKE	Cocok dengan pola karakter
IS NULL	Sama dengan NULL

Perintahnya :

Select * From Nama_Table Where Nama_Field [Operator Relasional] Ketentuan;

contoh :

select * from customer where cust_id = '10003' or cust_name ='wascals;

d. Pengurutan data (ACS, DESC, ORDER BY)

1. Mengurutkan Data

Untuk mengurutkan tampilan data dari suatu table, digunakan klausa Order By. Klausa Order By, dapat digunakan untuk mengurutkan data :

- Asc (Ascending) : Untuk mengurutkan data dari kecil ke besar
- Desc (Descending) : Untuk mengurutkan data dari besar ke kecil Perintahnya :

Select * From Nama_Table Order By Nama_Field_Key Asc/Desc;

Contoh :

Select * From products Order By prod_name Asc;

e. OPERATOR BETWEEN, IN,

LIKE 1. Operator Between

Operator Between merupakan operator yang digunakan untuk menangani operasi jangkauan.

Perintahnya :

Select * From Nama_Table Where Nama_Field_ketentuan Between 'Ketentuan_1' And 'Ketentuan_2';

Contoh :

Select * From orderitems Where quantity Between '1' And '10';

2. **Operator In**

Operator In merupakan operator yang digunakan untuk mencocokan suatu nilai.

Perintahnya :

Select Nama_Field From Nama_Table Where Nama_Field_Pencocok In ('Isi_Field_1','Isi_Field_2');

Contoh :

Menampilkan nama customer, alamat dan email customer tertentu.

Select cust_name,cust_address,cust_email From customers Where cust_id In ('10002','10005');

3. **Operator Like**

Operator Like merupakan operator yang digunakan untuk mencari suatu data (*search)*.

Perintahnya :

Select * From Nama_Table Where Nama_Field_Dicari Like '%Key';

Contoh :

Select * From Products Where prod_name Like '%s';

Query yang pertama menampilkan produk dengan nama produk diawali huruf dan pada query yang kedua nama produk diakhiri huruf s.

f. Ekspresi Query

Ekspresi Query dapat digunakan untuk melakukan perubahan terhadap field kolom keluaran, menambah baris teks field keluaran.

1. **Mengganti Nama Field**

keluaran Perintahnya :

Select Nama_Field_Asal As 'Nama_Field_Pengganti' From Nama_Table; Contoh :

Select Kode_Mtkul As 'Kode Matakuliah', Nama_Mtkul As 'Matakuliah' From Mtkul;

2. **Menambahkan Baris Teks Field**

Keluaran Perintahnya :

Select 'Nama Field Tambahan', Nama_Field_Asal From Nama_Table; Contoh :

Select vend_name,'diproduksi di', vend_city From vendors;

3. Ekspresi Kondisi

Perintahnya :

Select Nama_Field_1 Case Nama_Field_2 When 'Nilai_field_2' Then 'Keterangan_1' Else 'Keterangan_2' End As Nilai_field_2 From Nama_Table;

Contoh :

Select Kode_Mtkul, Nama_Mtkul, Case Sks When '1' Then 'Praktikum' Else 'Matakuliah' End As Sks From Mtkul;

g. Agregasi dan Pengelompokan Data

Dalam pemrosesan data mentah menjadi data statistik, diperlukan fungsi-fungsi yang dapat meng-agregasi data-data tersebut. Fungsi-fungsi ini meliputi SUM, MIN, MAX, COUNT, dan AVG.

Fungsi	Keterangan
AVG	Menghitung rata-rata
COUNT	Menghitung cacah data /jumlah baris
MAX	Memperoleh nilai terbesar
MIN	Memperoleh nilai terkecil
SUM	Memperoleh jumlahan data

h. Multiple-table Query

Data-data yang tersimpan dalam basis data, tersebar kedalam beberapa tabel. Tabel-tabel ini dihubungkan dengan yang namanya referential constraint, yaitu hubungan antara foreign key dan primary key.

Karena itulah, untuk mendapatkan informasi yang tersebar, dibutuhkan metode untuk menggabungkan property tabel-tabel tersebut. Metode yang digunakan ada 2 macam, yaitu join dan subquery.

Perbedaannya sederhana, join menggunakan satu SELECT, sedangkan subquery menggunakan dua atau lebih SELECT (umumnya dikatakan sebagai SELECT within a SELECT).

Join

Bentuk join pertama kali adalah menggunakan kata kunci WHERE untuk melakukan penggabungan tabel.

```
SELECT  <select_list>
FROM    <table1>, <table2> [, ...]
WHERE   <table1.PK = table2.FK> [AND ...]
```

Perkembangan SQL ANSI

```
SELECT  <select_list>
FROM    <table1> JOIN <table2>
        ON < table1.PK = table2.FK> [[AND ...]
        JOIN ...];
```

Tipe Join

Ada 2 tipe join, yaitu inner join yang lebih menekankan pada keberadaan data yang sama, dan outer join.

```
SELECT  <select_list>
FROM    <tabel1 sebagai kiri>
        <LEFT/RIGHT> [OUTER] JOIN
        <tabel2 sebagai kanan>
        ON <table1.PK = table2.FK> [AND ...];
```

Pada INNER JOIN atau CROSS JOIN *output*/hasil yang ditampilkan adalah data-data dari semua table yang terlibat dimana baris yang tampil hanya yang memiliki kondisi kesamaan data. Kesamaan data berdasarkan relasinya (kesamaan data *foreign key* dengan *primary key* tabel yang diacu). Berikut adalah bentuk umum INNER JOIN yang umumnya hanya disebut sebagai JOIN:

SELECT nm_tabel1.nm_kolom1, nm_tabel1.nm_kolom2, nm_tabel2.nm_kolom1, nm_tabel2.nm_kolom2 FROM tabel1, tabel2 WHERE tabel1.nama_kolom1

(primary key)=tabel2.nama_kolom1(foreign key yg mengacu ke tabel1)

Contoh penggunaan Join, kita lihat kembali skema order entry dibawah ini.

Menampilkan prod_name, vend_name dari table vendors dan products.

Select vendors.vend_name,products.prod_name from vendors, products
Where vendors.vend_id = products.vend-id

a. Clausa Join On Alias

SELECT a.nm_kolom1, b.nm_kolom2, a.nm_kolom3

FROM tabel1 a
JOIN tabel2 b
ON a.nama_kolom1(primary key)=b.nama_kolom1(foreign key yg mengacu ke tabel1)
WHERE kondisi;

b. JOIN 3 TABLE ATAU LEBIH

Pada prinsipnya sama, hanya jumlah tabel ditambah dan sintaks disesuaikan. Contoh penerapan join dua tabel atau lebih untuk menampilkan nama customer, tgl order dan total jumlah order.

select a.cust_name,b.order_date,c.quantity from customers a join orders b on a.cust_id=b.cust_id join orderitems c on b.order_num=c.order_num;

c. OUTER JOIN

Pada OUTER JOIN hasil yang ditampilkan adalah data-data dari semua tabel yang terlibat baik yang hanya yang memiliki kondisi kesamaan data berdasarkan relasinya (kesamaan data foreign key dengan primary key tabel yang diacu) maupun data yang tidak memiliki kesamaan data berdasarkan relasinya dari salah satu tabel. Terdapat dua tipe OUTER JOIN, yaitu:

1. LEFT OUTER JOIN atau biasa disebut left join
2. RIGHT OUTER JOIN atau biasa disebut righ join

a) LEFT JOIN

Pada LEFT JOIN output/hasil yang ditampilkan adalah data-data dari semua tabel yang terlibat baik yang hanya yang memiliki kondisi kesamaan data berdasarkan relasinya (kesamaan data foreign key dengan primary key tabel yang diacu) maupun data-data yang tidak memiliki kesamaan data berdasarkan relasinya dari tabel sebelah kiri dari klausa LEFT JOIN. Berikut adalah bentuk umum:

SELECT nm_tabel1.nm_kolom1, nm_tabel1.nm_kolom2, nm_tabel2.nm_kolom1, nm_tabel2.nm_kolom2 FROM tabel1

LEFT JOIN tabel2
ON tabel1.nama_kolom1(primary key)=tabel2.nama_kolom1(foreign key yg mengacu ke tabel1)
WHERE kondisi;

Contoh :

```
select a.cust_name,b.order_date
from customers a left join orders b on a.cust_id=b.cust_id RIGHT JOIN
```

Pada RIGHT JOIN output/hasil yang ditampilkan adalah data-data dari semua tabel yang terlibat baik yang hanya yang memiliki kondisi kesamaan data berdasarkan relasinya (kesamaan data foreign key dengan primary key tabel yang diacu) maupun data-data yang tidak memiliki kesamaan data berdasarkan relasinya dari tabel sebelah kanan dari klausa RIGHT JOIN.

```
SELECT nm_tabel1.nm_kolom1, nm_tabel1.nm_kolom2,
nm_tabel2.nm_kolom1, nm_tabel2.nm_kolom2
FROM tabel1
RIGHT JOIN tabel2
ON tabel1.nama_kolom1(primary key)=tabel2.nama_kolom1(foreign
key yg mengacu ke tabel1)
WHERE kondisi;
```

Contoh :

```
select a.cust_name,b.order_date from customers a right join orders b
on a.cust_id=b.cust_id
```

b) SELF JOIN

Self join adalah melakukan join dengan dirinya sendiri. Atau join dengan table yang sama.

Sintak nya sbb :

```
select nama alias1_table.kolom1, nama alias2_table.kolom2, from
table alias1 inner join table alias2 on alias1.kolom3=alias2.kolom3
```

contoh

```
select a.vend_name,b.vend_state, 'negaranya' ,b.vend_country from
vendors a inner join vendors b on a.vend_id=b.vend_id
```

i. Subquery

Subquery merupakan query didalam query. Umumnya, subquery ini dipakai untuk mencari data yang belum diketahui. Penggunaan query didalam query ini umumnya menjadi bagian dari kondisi.

Sintak subquery adalah sebagai berikut:

```
SELECT  <select_list>
FROM    <tabel>
WHERE   <column> =
           (SELECT <single_column>
           FROM    <tabel>
           WHERE   <kondisi>);
```

Tugas Pratikum

1. Buat sebuah database dengan nama coba !
2. Buat sebuah tabel dengan nama mahasiswa di dalam database coba, lalu
   - Tambahkan sebuah kolom : keterangan (varchar 15), sebagai kolom terakhir !
   - Tambahkan kolom nim (int 11) di awal (sebagai kolom pertama) !
   - Sisipkan sebuah kolom dengan nama phone (varchar 15) setelah kolom alamat varchar(15) !
   - Ubah atribut kolom nim menjadi char(11) !
   - Ubah nama kolom phone menjadi telepon (varchar 20) !
   - Hapus kolom keterangan dari tabel !
   - Ganti nama tabel menjadi student!
   - Jadikan nim sebagai primary key !
3. Screnshoot semua perintah-perintah SQL percobaan di atas beserta outputnya !
4. Apa maksud dari int (11) ?
5. Ketika kita melihat struktur tabel dengan perintah desc, ada kolom Null yang berisi Yes dan No. Apa maksudnya ?