DASAR – DASAR
SQL (Structured Query Language)

SQL adalah bahasa standar dalam basis data yang digunakan untuk melakukan manipulasi data. Standardisasi bahasa ini dilakukan oleh ANSI tahun 86, 89,92 dan 99, dimana tiap perubahan tahun dilakukan peningkatan kemampuan SQL. Pada perkembangan saat ini standar yang paling banyak digunakan adalah standar ANSI 92. Hampir semua DBMS menggunakan SQL sebagai fasilitas untuk memanipulasi data seperti Oracle, SQLServer, MySQL, PostgreSQL, Foxpro dsb.

Meskipun awalnya hanya merupakan bahasa untuk memanipulasi data, pada perkembangannya SQL juga dapat digunakan untuk melakukan definisi data maupun control (security) terhadap data. Sehingga bahasa Query ini dibagi menjadi 3 bagian :

- DDL (Data Definition Language)
- DML (Data Definition Language)
- DCL (Data Control Language)

DDL

DDL merupakan bahasa yang digunakan untuk membuat atau memodifikasi database dan tabel, Perintah DDL a.l:

CREATE → Untuk membuat

Syntak :

- membuat database

CREATE DATABASE *nama_database*

- membuat tabel

CREATE TABLE *nama_tabel* (field1 *type_data1* (*lebar_data1*), field2 *type_data2* (*lebar_data2*))

Apabila akan menambahkan konstrain integritas PRIMARY KEY maka syntaknya adalah sbb :

CREATE TABLE *nama_tabel* (field1 *type_data1* (*lebar_data1*) PRIMARY KEY, field2 *type_data2* (*lebar_data2*))

Catatan : yang akan dijadikan primary key adalah field1

Misal :

CREATE TABLE mahasiswa (nim char(10), nama char(30), jurusan char(2), ipk float(4,2))

DROP → Untuk menghapus :

Syntak :

- menghapus database

DROP DATABASE *nama_table*

- menghapus tabel

DROP TABLE *nama_table*

ALTER → Untuk memodifikasi tabel

Syntak :

- menambah field baru

ALTER TABLE *nama_tabel* ADD COLUMN *field_baru type_data*(*lebar_data)*

Misal : ALTER TABLE mahasiswa ADD COLUMN alamat varchar(80)

- menghapus field

ALTER TABLE *nama_tabel* DROP COLUMN *field_yang_dihapus*

Misal : ALTER TABLE mahasiswa DROP COLUMN alamat

- mengedit / mengganti field

ALTER TABLE *nama_tabel* CHANGE [COLUMN] *field_lama field_baru type_data*(*lebar_data)*

Misal : ALTER TABLE mahasiswa CHANGE nama nama_mhs char(40)

DML

DML merupakan bahasa untuk memanipulasi data (membaca/menampilkan, menambah, mengedit, menghapus)

Perintah DML a.l :

1. untuk membaca atau menampilkan data

SELECT *daftar_field_yang_akan_ditampilkan*

FROM *nama_tabel*

[WHERE *kriteria_data_yang_akan_ditampilkan*]

Operator Relasi

Beberapa operator relasi yang digunakan pada saat akan dibutuhakan suatu kriteria tertentu untuk menampilkan data adalah :

Operator	Arti
=	Sama dengan
>	Lebih Besar
>=	Lebih Besar atau sama dengan
<	Lebih Kecil
<=	Lebih Kecil atau sama dengan
<>	Tidak Sama Dengan
LIKE	Mengandung suatu kata/huruf tertentu
BETWEEN	Rentang antara dua nilai

Sedangkan operator logika yang sering digunakan adalah :
AND, OR dan NOT

Misal :
- menampilkan nim dan nama semua mahasiswa
SELECT nim, nama FROM mahasiswa
- menampilkan nim dan nama mahasiswa jurusan MI
SELECT nim, nama FROM mahasiswa WHERE jurusan = "MI"
- menampilkan semua field dari tabel mahasiswa jurusan MI
SELECT * FROM mahasiswa WHERE jurusan = "MI"
- menampilkan nama mahasiswa yang berawalan "PAR"
SELECT nama FROM mahasiswa WHERE nama LIKE "PAR%"
- Menampilkan nim,nama dan IPK mahasiswa yang mempunyai IPK 2,5 sampai 3,2
SELECT nim,nama,ipk FROM mahasiswa WHERE ipk BETWEEN 2.5 AND 3.2
- Menampilkan nim,nama dan alamat mahasiswa jurusan TI yang berjenis kelamin wanita
SELECT nim,nama,alamat FROM mahasiswa WHERE jurusan= "TI" AND jenis_kel = "WANITA"

Pengurutan

Untuk menampilkan suatu hasil query dengan urutan tertentu dapat dilakukan dengan tambahan perintah ORDER BY

Misal :

- tampilkan nim dan nama mahasiswa urut nama

SELECT nim, nama FROM mahasiswa ORDER BY nama

Field Function

Dalam SQL ada beberapa fungsi yang bisa langsung digunakan dalam perintah SELECT yaitu :

SUM → untuk menjumlahkan
COUNT → untuk mencacah
MAX → untuk menentukan nilai terbesar
MIN → untuk menentukan nilai terkecil
AVG → untuk menentukan nilai rata-rata

Contoh cara penggunaan :

- menampilkan total gaji pokok yang harus dibayarkan pada semua pegawai

  SELECT sum (gaji_pokok) FROM pegawai
- mengetahui jumlah pegawai yang bergaji pokok lebih besar dari 500000

  SELECT count (*) FROM pegawai where gaji_pokok > 500000
- menampilkan gaji_pokok terbesar

  SELECT max (gaji_pokok) FROM pegawai

Pengelompokan

Apabila diinginkan dapat ditampilkan suatu hasil query berdasar kelompok tertentu. Biasanya perintah ini digabungkan dengan suatu perintah field function. Statemen yang digunakan adalah GROUP BY

Misal :

- menampilkan total gaji pokok yang harus dibayarkan pada pegawai tiap departemen

  SELECT nama_dept, sum (gaji_pokok) FROM pegawai ORDER BY nama_dept GROUP BY nama_dept

  → ini akan menghasilkan data total gaji pokok tiap departemen

2. Untuk Menambah data baru

Syntak :

INSERT INTO *nama_tabel* (*field1, field2, …*) VALUES (*value1, value2,…*)

Field dan value harus berjumlah sama dan masing-masing berpasangan, artinya : value1 akan diisikan ke field1, value2 akan diisikan ke field2, dst.

Misal :

INSERT INTO mahasiswa (nim, nama) VALUES (“05023562”,”TOTOK”)

3. Untuk MengEdit Data

Syntak :

UPDATE *nama_tabel* SET *field1 = value1, field2 = value2,…*

[WHERE *kriteria*]

Misal :

Untuk mengganti nama mahasiwa menjadi ANDI untuk nim 05023562

UPDATE mahasiswa SET nama = “ANDI” WHERE nim=”05023562”

4. Untuk Menghapus Data

Syntax :

DELETE FROM nama_tabel

[WHERE *Kriteria*]

Misal :

Untuk menghapus data mahasiswa yang mempunyai nim 05023562

DELETE FROM mahasiswa WHERE nim = “05023562”

DCL

DCL merupakan bahasa yang digunakan untuk pengaturan akses seorang user terhadap data.

Catatan : user sudah harus terdaftar

1. Memberikan Hak akses kepada user

Syntak :

GRANT *hak_atau_privileges*

ON *Nama_Database.Nama_Tabel*

TO *Nama_User*

Misal :

(contoh berikut menggunakan MySQL)

Seorang user (totok dengan host : localhost) diperkenankan untuk membaca dan menambah data mahasiswa dalam database kampus tetapi tidak diperkenankan untuk melakukan hal yang lain seperti mengedit, menghapus dll

GRANT select, insert ON kampus.mahasiswa TO totok@localhost

2. Mencabut Hak akses dari user

Syntak :

REVOKE *hak_atau_privileges*

ON *Nama_Database.Nama_Tabel*

FROM *Nama_User*

Misal :

(contoh berikut menggunakan MySQL)

hak yang diberikan kepada totok di atas dicabut

REVOKE select, insert ON kampus.mahasiswa FROM totok@localhost

Tambahan Untuk SELECT:

Perintah untuk penggabungan 2 tabel atau lebih.

Statemen yang digunakan adalah **JOIN**

Syntax :

SELECT field1, field2, … FROM *nama_tabel1* [INNER] [LEFT] [RIGHT] JOIN *nama_tabel2* ON *nama_tabel2.field_penghubung = nama_tabel1.field_penghubung*

Misal diketahui data :

Mahasiswa

NIM	Nama_mahasiswa	Kode_jurusan	IPK
001	TOTOK	TI	3
002	TITIK	TI	2.5
003	TATAK	MI	2.0
004	TUTUK	TK	3.2

Jurusan

Kode_jurusan	Nama_Jurusan
TI	Teknik Informatika
MI	Manajemen Informatika
SI	Sistem Informasi
KA	Komputer Akuntansi

- Menampilkan Nim, nama mahasiwa, kode_jurusan dan nama jurusan

INNER JOIN

Catatan : kedua tabel tersebut dihubungkan oleh field kode_jurusan

SELECT nim, nama_mahasiwa, **mahasiswa.kode_jurusan**, jurusan.nama_jurusan FROM mahasiswa INNER JOIN jurusan ON *jurusan.kode_jurusan = mahasiswa.kode_jurusan*

Hasil :

NIM	Nama_mahasiswa	Kode_jurusan	Nama_jurusan
001	TOTOK	TI	Teknik Informatika
002	TITIK	TI	Teknik Informatika
003	TATAK	MI	Manajemen Informatika

Beberapa hal yang perlu diperhatikan :

- Yang dicetak miring adalah field yang saling berelasi (penghubung)
- Data TUTUK tidak muncul karena TUTUK terdaftar di kode jurusan TK yang tidak ada di tabel Jurusan. → Perintah INNER JOIN hanya menampilkan data yang cocok pada kedua tabel.
- Pada SQL diatas terlihat bahwa untuk nama_mahasiswa tidak disertai nama tabelnya sedangkan kode_jurusan (yang dicetak tebal) ditulis dengan didahului nama tabelnya, ini disebabkan untuk nama_mahasiswa, nim maupun nama_jurusan hanya terdapat di salah satu tabel yang terhubung (nama_mahasiswa dan nim hanya terletak di tabel mahasiswa sedangkan nama_jurusan hanya terletak di tabel jurusan), sedangkan kode_jurusan terletak di kedua tabel yang terhubung sehingga diperlukan penulisan nama tabel pemilik field tersebut secara eksplisit.

Perintah yang menggunakan INNER JOIN di atas dapat diganti dengan perintah :

SELECT nim, nama_mahasiwa, **mahasiswa.kode_jurusan**, jurusan.nama_jurusan FROM mahasiswa, jurusan WHERE *jurusan.kode_jurusan = mahasiswa.kode_jurusan*

LEFT JOIN

SELECT nim, nama_mahasiwa, **mahasiswa.kode_jurusan**, jurusan.nama_jurusan FROM mahasiswa LEFT JOIN jurusan ON *jurusan.kode_jurusan = mahasiswa.kode_jurusan*

NIM	Nama_mahasiswa	Kode_jurusan	Nama_jurusan
001	TOTOK	TI	Teknik Informatika
002	TITIK	TI	Teknik Informatika
003	TATAK	MI	Manajemen Informatika
004	TUTUK	TK	

Dengan menggunakan LEFT JOIN semua data pada tabel yang disebut pertama (dianggap tabel kiri / left) akan ditampilkan semua, apabila tidak ada data yang cocok pada tabel kedua (yang kanan / right) maka akan ditampilkan NULL atau kosong.

RIGHT JOIN

SELECT nim, nama_mahasiwa, **mahasiswa.kode_jurusan**, jurusan.nama_jurusan FROM mahasiswa RIGHT JOIN jurusan ON *jurusan.kode_jurusan = mahasiswa.kode_jurusan*

NIM	Nama_mahasiswa	Kode_jurusan	Nama_jurusan
001	TOTOK	TI	Teknik Informatika
002	TITIK	TI	Teknik Informatika
003	TATAK	MI	Manajemen Informatika
			Sistem Informasi
			Komputer Akuntansi

Dengan menggunakan LEFT JOIN semua data pada tabel yang disebut kedua (dianggap tabel kanan / right) akan ditampilkan semua, apabila tidak ada data yang cocok pada tabel pertama (yang kiri / left) maka akan ditampilkan NULL atau kosong.

Perintah Penggabungan 2 tabel ini juga bisa digabung dengan kriteria yang lain, misal :

- menampilkan Nim, nama mahasiwa, kode_jurusan dan nama jurusan untuk mahasiswa yang mempunyai IPK lebih besar dari 2.75

Perintahnya :

```
SELECT nim, nama_mahasiwa, mahasiswa.kode_jurusan,
jurusan.nama_jurusan FROM mahasiswa INNER JOIN jurusan ON
jurusan.kode_jurusan = mahasiswa.kode_jurusan
WHERE ipk > 2.75
```

Atau :

```
SELECT nim, nama_mahasiwa, mahasiswa.kode_jurusan,
jurusan.nama_jurusan FROM mahasiswa, jurusan
WHERE jurusan.kode_jurusan = mahasiswa.kode_jurusan AND ipk > 2.75
```