

SQL (*Structured Query Language*)

Bahasa *database* adalah bahasa khusus yang ditetapkan pembuat DMBS (*database management system*), tentang cara berinteraksi/berkomunikasi antara pemakai dengan *database*. Bahasa ini terdiri atas sejumlah perintah yang diformulasikan untuk dapat diberikan oleh pengguna dan dikenali oleh DBMS. Salah satu bahasa *database* yang populer adalah SQL.

SQL (dibaca "ess-que-el") singkatan dari *Structured Query Language*. SQL digunakan untuk berkomunikasi dengan *database*. Menurut ANSI (*American National Standards Institute*), SQL merupakan bahasa standar untuk sistem manajemen *database* relasional. Perintah SQL digunakan untuk melakukan tugas-tugas seperti *update* data, atau mengambil data dari *database*. Beberapa sistem manajemen *database* relasional umum yang menggunakan SQL adalah: Oracle, Sybase, Microsoft SQL Server, Access, Ingres, dan lain-lain. Meskipun sebagian besar sistem *database* menggunakan SQL, kebanyakan dari mereka juga memiliki ekstensi tambahan milik mereka sendiri yang biasanya hanya digunakan pada sistem mereka. Meskipun demikian, perintah-perintah SQL standar seperti "Select", "Insert", "Update", "Delete", "Create", dan "Drop" dapat digunakan pada seluruh DBMS tersebut. Tutorial ini akan memberikan petunjuk pada Anda dasar-dasar perintah SQL serta penggunaannya dalam operasi *database*.

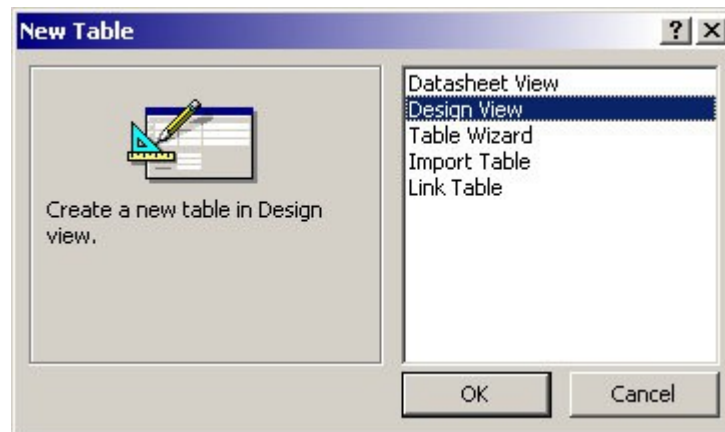
Pendahuluan

Secara umum, terdapat 2 jenis bahasa *database*, yaitu: DDL (*data definition language*) dan DML (*data manipulation language*). DDL merupakan perintah-perintah yang biasa digunakan administrator *database* untuk mendefinisikan skema dan sub-skema *database* (Contoh: CREATE, ALTER, MODIFY). Sedangkan, DML merupakan perintah-perintah yang memungkinkan pengguna melakukan akses dan manipulasi data sebagaimana yang telah diorganisasikan sebelumnya dalam model data yang tepat (Contoh: INSERT, UPDATE, DELETE).

DML terdiri dari 2 macam, yaitu: Prosedural dan Non-Prosedural. Prosedural berarti perintah yang memungkinkan pengguna menentukan data apa yang dibutuhkan serta bagaimana cara mendapatkannya (Contoh: dBASE III, FoxBASE). Sedangkan, Non-Prosedural berarti perintah yang memungkinkan pengguna menentukan data apa

saja yang diinginkan tanpa menyebutkan bagaimana cara mendapatkannya (Contoh: SQL, QBE).

Karena tutorial kali ini menggunakan **Microsoft Access**, pembahasan dibatasi pada penerapan perintah SQL untuk **DML**. Sebagai ganti perintah DDL, pembuatan tabel dalam *database* dapat menggunakan fitur **Design View** seperti yang ditunjukkan Gambar 1 berikut ini.



Gambar 1. Fitur **Design View** untuk membuat tabel

Berikut disajikan beberapa perintah SQL yang akan dibahas pada tutorial ini.

Perintah SQL Dasar

- **SELECT**, digunakan untuk memfilter atribut-atribut dari relasi (tabel) berdasarkan kondisi yang mengikutinya.
- **FROM**, digunakan untuk menunjukkan dari relasi mana data yang akan difilter.
- **WHERE**, digunakan untuk membuat suatu kondisi.
- **GROUP BY**, digunakan untuk mengelompokkan data berdasarkan atribut tertentu
- **HAVING**, digunakan untuk mendukung klausa **GROUP BY**, yakni untuk menentukan kondisi bagi klausa **GROUP BY**.
- **AVG**, digunakan untuk menghitung rata-rata.
- **COUNT**, digunakan untuk menghitung cacah data.
- **MAX**, digunakan untuk memperoleh nilai terbesar
- **MIN**, digunakan untuk memperoleh nilai terkecil.
- **SUM**, digunakan untuk memperoleh jumlahan data.
- Dan berbagai perintah SQL lainnya.

Sub-Query

Subquery berarti *query* di dalam *query*. Dengan menggunakan *subquery*, hasil *query* akan menjadi bagian dari *query* lain.

Subquery terletak di dalam klausa **WHERE** atau **HAVING**. Pada klausa **WHERE**, *subquery* digunakan untuk memilih baris-baris tertentu, yang kemudian digunakan oleh *query*. Sedangkan pada klausa **HAVING**, *subquery* digunakan untuk memilih kelompok baris, yang kemudian digunakan oleh *query*.

- **EXISTS**, digunakan untuk memeriksa keadaan baris yang dihasilkan *query* terhadap yang dihasilkan oleh *subquery*.
- **ANY**, digunakan berkaitan dengan *subquery*, hampir mirip dengan memilih tetapi dengan operasi **OR** (lihat contoh penerapan perintah **ANY** di bagian pembahasan).

- ALL, digunakan untuk melakukan perbandingan dengan *subquery*. Kondisi dengan ALL menghasilkan nilai *true* jika *subquery* tidak menghasilkan apapun atau jika perbandingan menghasilkan *true* untuk setiap nilai *query* terhadap hasil *subquery*.

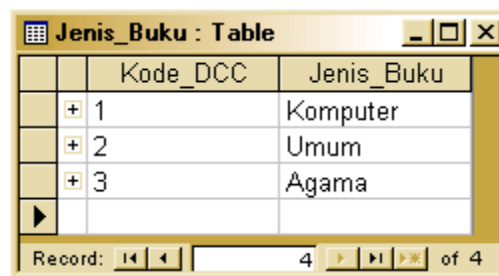
Perintah SQL untuk Banyak Tabel

- UNION, merupakan operator yang digunakan untuk menggabungkan hasil *query*.
- JOIN, digunakan untuk menggabungkan dua tabel atau lebih dengan hasil berupa gabungan dari kolom-kolom yang berasal dari tabel-tabel tersebut. Pada JOIN sederhana, tabel-tabel digabungkan dan didasarkan pada pencocokan antara kolom pada tabel yang berbeda. Ada beberapa perintah JOIN pada Access, yakni INNER JOIN, LEFT JOIN, dan RIGHT JOIN.

Persiapan Data

Untuk mempermudah memahami konsep dan cara kerja *query* (perintah SQL), berikut disajikan beberapa tabel lengkap dengan *record*-nya.

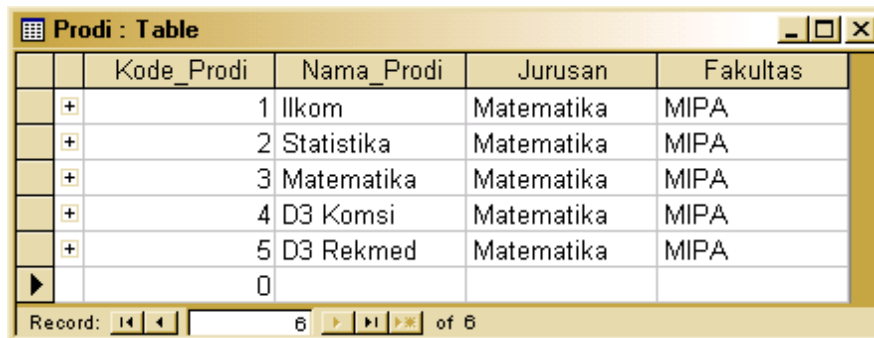
- Tabel **Jenis_Buku**.



Kode_DCC	Jenis_Buku
1	Komputer
2	Umum
3	Agama

Gambar 2. Tabel **Jenis_Buku**

- Tabel **Prodi**.



Kode_Prodi	Nama_Prodi	Jurusan	Fakultas
1	Ilkom	Matematika	MIPA
2	Statistika	Matematika	MIPA
3	Matematika	Matematika	MIPA
4	D3 Komsu	Matematika	MIPA
5	D3 Rekmed	Matematika	MIPA
0			

Gambar 3. Tabel **Prodi**

- Tabel **Pengarang**.



Kode_Pengarang	Nama_Pengarang
1	Inge Martina
2	Quraish Shihab
3	Antony Pranata
4	Onno W Purbo
5	Susilo Husodo
0	

Gambar 4. Tabel **Pengarang**

- Tabel Anggota.

	No_Anggota	NIM	Nama	Alamat	Jenis_Kelamin	Angkatan	Kode_Prodi
+	1	1390	Riyanto	Pati	0	2002	4
+	2	1366	Sismanto	Yogyakarta	0	2002	4
+	3	1392	Ningrum	Yogyakarta	1	2002	5
*	0					0	0

Record: 1 of 3

Gambar 5. Tabel Anggota

- Tabel Daftar_Buku.

	Kode_Buku	ISBN
	1	9998887776
	2	9998887777
	3	9998887778
	4	9998887779
▶	0	

Record: 5 of 5

Gambar 6. Tabel Daftar_Buku

- Tabel Penerbit.

	Kode_Penerbit	Nama_Penerbit	Kota_Penerbit
+	1	Elex Media Komputindo	Jakarta
+	2	Eksekutif	Bandung
+	3	Andi Offset	Yogyakarta
+	4	Gava Media	Yogyakarta
+	5	Toha Putra	Semarang
▶	0		

Record: 6 of 6

Gambar 7. Tabel Penerbit

- Tabel Sirkulasi.

	Kode_Buku	No_Anggota	Tgl_Pinjam	Tgl_Harus Kembali	Tgl_Kembali
	1	1	1/3/2004	1/18/2004	3/2/2004
	2	2	3/2/2004	3/17/2004	3/16/2004
	1	2	3/2/2004	3/17/2004	3/15/2004
▶	0	0			

Record: 4 of 4

Gambar 8. Tabel Sirkulasi

Penggunaan Perintah SQL

Sekarang saatnya masuk pembahasan, yaitu penerapan perintah SQL untuk operasi *database*. Operasi yang dimaksud adalah pengambilan data atau informasi dari *database* baik yang melibatkan 1 atau lebih tabel, maupun *query* dalam *query* (*nested query*).

Penerapan perintah SQL yang hanya melibatkan 1 tabel

- `SELECT * FROM Anggota order by NIM;`

Artinya:

Menampilkan **seluruh** atribut yang ada pada tabel **Anggota** dan diurutkan berdasarkan atribut **NIM**

Output:

No_Anggota	NIM	Nama	Alamat	Jenis_Kelamin	Angkatan	Kode_Prodi
2	1366	Sismanto	Yogyakarta	0	2002	4
1	1390	Riyanto	Pati	0	2002	4
3	1392	Ningrum	Yogyakarta	1	2002	5

- `SELECT NIM, Nama FROM Anggota where Jenis_Kelamin=0 order by NIM;`

Artinya:

Menampilkan atribut **NIM** dan **Nama** yang ada pada tabel **Anggota** dengan kriteria berjenis kelamin 0 (laki-laki) dan diurutkan berdasarkan atribut **NIM**.

Output:

NIM	Nama
1366	Sismanto
1390	Riyanto

- `SELECT NIM, Nama, Angkatan, Alamat FROM Anggota Where Angkatan=2002 and Alamat='Yogyakarta';`

Artinya:

Menampilkan atribut **NIM**, **Nama**, **Angkatan** dan **Alamat** yang ada pada tabel **Anggota** yang memenuhi 2 kondisi, yaitu Angkatan 2002 dan beralamat di “Yogyakarta”.

Output:

NIM	Nama	Angkatan	Alamat
1366	Sismanto	2002	Yogyakarta
1392	Ningrum	2002	Yogyakarta

- `SELECT Jenis_kelamin, Count(*) As Jumlah FROM Anggota Group by Jenis_Kelamin;`

Artinya:

Menampilkan **Jumlah record** pada tabel **Anggota** yang dikelompokkan berdasarkan atribut **Jenis_Kelamin**.

Output:

Jenis_kelamin	Jumlah
0	2
1	1

- `SELECT Count(*) AS Jumlah_Pria FROM Anggota Where Jenis_Kelamin=0;`

Artinya:

Menampilkan **Jumlah record** pada tabel **Anggota** yang berjenis kelamin 0 (laki-laki) dan ditampilkan dalam atribut **Jumlah_Pria**.

Output:

Jumlah_Pria
2

- `SELECT Alamat, Count(*) AS Banyak FROM Anggota GROUP BY Alamat Having Count(*)>1;`

Artinya:

Menampilkan **Alamat** pada tabel **Anggota** beserta jumlahnya dengan kriteria yang jumlah alamatnya lebih dari 1 (>1).

Output:

Alamat	Banyak
Yogyakarta	2

- `SELECT Nama,Alamat FROM Anggota Where Alamat='Yogyakarta';`

Artinya:

Menampilkan isi atribut **Nama** dan **Alamat** dari data **Anggota** yang beralamat di “Yogyakarta”.

Output:

Nama	Alamat
Sismanto	Yogyakarta
Ningrum	Yogyakarta

- `SELECT distinct Kode_Buku FROM sirkulasi;`

Artinya:

Menampilkan seluruh isi dari atribut **Kode_Buku**, tetapi jika ada **Kode_Buku** yang sama, maka hanya akan ditulis satu.

Output:

Kode_Buku
1
2

- `SELECT * FROM Info_buku where judul like "*Delphi*";`

Artinya:

Menampilkan seluruh informasi dari tabel **Info_Buku** yang judulnya mengandung kata “delphi”.

Output:

ISBN	DCC	Judul	Kode_Pengaran	Kode_Penerbit	Th_Terbit
9998887776	1	Tip & Trik Delphi	3	1	2002

- `SELECT * FROM Info_buku where judul like "Tip*";`

Artinya:

Menampilkan seluruh informasi dari tabel **Info_Buku** yang judulnya diawali kata "Tip".

Output:

ISBN	DCC	Judul	Kode_Pengaran	Kode_Penerbit	Th_Terbit
9998887776	1	Tip & Trik Delphi	3	1	2002

- `SELECT TOP 40 PERCENT * FROM Info_buku;`

Artinya:

Menampilkan 40 persen teratas dari seluruh informasi yang ada pada tabel **Info_Buku**.

Output:

ISBN	DCC	Judul	Kode_Pengaran	Kode_Penerbit	Th_Terbit
9998887776	1	Tip & Trik Delphi	3	1	2002
9998887777	1	Step by step Access 2000	1	1	2000

- `SELECT * into BackUp_Buku FROM Info_buku;`

Artinya:

Menyimpan seluruh isi tabel **Info_Buku** dengan nama yang lain (akan tercipta tabel baru dengan nama **BackUp_Buku**), biasanya digunakan untuk membuat *temporary table*.

Output:

ISBN	DCC	Judul	Kode_Pengaran	Kode_Penerbit	Th_Terbit
9998887775	1	Kamus Internet	4	1	2002
9998887776	1	Tip & Trik Delphi	3	1	2002
9998887777	1	Step by step Access 2000	1	1	2000
9998887778	2	Manajemen Umum	5	2	1999
9998887779	3	Jalan Menuju Surga	2	1	2003

Penerapan Perintah SubQuery

- `SELECT Judul, Th_terbit FROM Info_buku where Th_terbit>=all(select Th_terbit from Info_buku);`

Artinya:

Menampilkan informasi buku terbaru (dalam hal ini diwakili oleh **Judul** buku), dengan membandingkan **tahun terbit** dengan seluruh **tahun terbit** yang lain, kemudian yang paling besar yang ditampilkan.

Output:

Judul	Th_terbit
Jalan Menuju Surga	2003

- `SELECT Judul, Th_terbit FROM Info_buku where Th_terbit > any(select Th_terbit from Info_buku);`

Artinya:

Menampilkan informasi buku beserta tahun terbit, dengan membandingkan **tahun terbit** dengan seluruh **tahun terbit** yang lain, kemudian yang **lebih besar atau sama** dengan yang ditampilkan (karena menggunakan ANY).

Output:

Judul	Th_terbit
Tip & Trik Delphi	2002
Step by step Access 2000	2000
Jalan Menuju Surga	2003
Kamus Internet	2002

- `SELECT Kode_Pengarang, Nama_Pengarang FROM Pengarang where exists(select * from Info_buku where Kode_Pengarang = Info_buku.Kode_Pengarang);`

Artinya:

Menampilkan informasi **Kode_Pengarang** dan **Nama_Pengarang** dari tabel **Pengarang** yang **Kode_Pengarang** tersebut ada di tabel **Info_Buku**.

Output:

Kode_Pengarang	Nama_Pengarang
1	Inge Martina
2	Quraish Shihab
3	Antony Pranata
4	Onno W Purbo
5	Susilo Husodo

Penerapan Perintah yang melibatkan Lebih dari Satu Tabel :

- `SELECT Info_buku.Judul, Penerbit>Nama_Penerbit FROM Penerbit INNER JOIN Info_buku ON Penerbit.Kode_Penerbit = Info_buku.Kode_Penerbit;`

Artinya:

Menampilkan informasi **Judul** beserta **Nama_Penerbit** yang berasal dari penggabungan dua tabel (tabel **Penerbit** dan **Info_Buku**) dengan kondisi **Kode_Penerbit** dari kedua tabel tersebut adalah sama.

Output:

Judul	Nama_Penerbit
Tip & Trik Delphi	Elex Media Komputindo
Step by step Access 2000	Elex Media Komputindo
Manajemen Umum	Eksekutif
Jalan Menuju Syurga	Elex Media Komputindo
Kamus Internet	Elex Media Komputindo

- ```
SELECT Info_buku.Judul, Pengarang>Nama_Pengarang,
Penerbit>Nama_Penerbit, Info_Buku.Th_Terbit FROM Pengarang
INNER JOIN (Penerbit INNER JOIN Info_buku ON
Penerbit.Kode_Penerbit = Info_buku.Kode_Penerbit) ON
Pengarang.Kode_Pengarang = Info_buku.Kode_Pengarang Order by
Th_Terbit;
```

*Artinya:*

Menampilkan informasi **Judul** beserta **Pengarang**, **Penerbit**, dan **Th\_Terbit**-nya yang berasal dari penggabungan 3 tabel (tabel **Penerbit**, **Pengarang** dan **Info\_Buku**) dengan kondisi **Kode\_Penerbit** pada tabel **Penerbit** sama dengan pada tabel **Info\_Buku**, dan **Kode\_Pengarang** pada tabel **Info\_Buku** sama dengan pada tabel **Pengarang**.

*Output:*

| Judul                    | Nama_Pengarang | Nama_Penerbit         | Th_Terbit |
|--------------------------|----------------|-----------------------|-----------|
| Manajemen Umum           | Susilo Husodo  | Eksekutif             | 1999      |
| Step by step Access 2000 | Inge Martina   | Elex Media Komputindo | 2000      |
| Kamus Internet           | Onno W Purbo   | Elex Media Komputindo | 2002      |
| Tip & Trik Delphi        | Antony Pranata | Elex Media Komputindo | 2002      |
| Jalan Menuju Syurga      | Quraish Shihab | Elex Media Komputindo | 2003      |

- ```
SELECT Judul, Th_Terbit FROM Penerbit INNER JOIN Info_buku ON
Penerbit.Kode_Penerbit = Info_buku.Kode_Penerbit where
Th_terbit='2002';
```

Artinya:

Menampilkan informasi **Judul** beserta **Th_Terbit**-nya yang berasal dari penggabungan 2 tabel (tabel **Penerbit** dan **Info_Buku**) dengan kondisi tahun terbit adalah 2002.

Output:

Judul	Th_Terbit
Tip & Trik Delphi	2002
Kamus Internet	2002

- ```
SELECT Kode_Penerbit, Nama_Penerbit FROM Penerbit Where not
Kode_Penerbit in(select distinct Kode_Penerbit From
Info_Buku);
```

*Artinya:*

Menampilkan informasi **Kode\_Penerbit** beserta **Nama\_Penerbit**-nya dari tabel **Penerbit**, dimana **Kode\_Penerbit** yang dimaksud tidak ada pada tabel **Info\_Buku**.

Output:

| Kode_Penerbit | Nama_Penerbit |
|---------------|---------------|
| 3             | Andi Offset   |
| 4             | Gava Media    |
| 5             | Toha Putra    |

- `SELECT Info_buku.Judul, Penerbit>Nama_Penerbit FROM Penerbit left JOIN Info_buku ON Penerbit.Kode_Penerbit = Info_buku.Kode_Penerbit;`

Artinya:

Menampilkan informasi **Judul** beserta **Nama\_Penerbit**-nya yang berasal dari penggabungan 2 tabel (tabel **Penerbit** dan **Info\_Buku**) dengan kondisi **Kode\_Penerbit** dari kedua tabel tadi sama, dan **seluruh** penerbit pada *left table* (tabel **Penerbit**) meskipun **Kode\_Penerbit**-nya tidak ada pada *right table* (tabel **Info\_Buku**).

Output:

| Judul                    | Nama_Penerbit         |
|--------------------------|-----------------------|
| Kamus Internet           | Elex Media Komputindo |
| Jalan Menuju Syurga      | Elex Media Komputindo |
| Step by step Access 2000 | Elex Media Komputindo |
| Tip & Trik Delphi        | Elex Media Komputindo |
| Manajemen Umum           | Eksekutif             |
|                          | Andi Offset           |
|                          | Gava Media            |
|                          | Toha Putra            |

- `SELECT Info_buku.Judul, Penerbit>Nama_Penerbit FROM Penerbit Right JOIN Info_buku ON Penerbit.Kode_Penerbit = Info_buku.Kode_Penerbit;`

Artinya:

Menampilkan informasi **Judul** beserta **Nama\_Penerbit**-nya yang berasal dari penggabungan 2 tabel (tabel **Penerbit** dan **Info\_Buku**) dengan kondisi **Kode\_Penerbit** dari kedua tabel tadi sama, dan **seluruh** judul buku pada *right table* (tabel **Info\_Buku**) meskipun **Kode\_Penerbit**-nya tidak ada pada *left table* (tabel **Penerbit**).

Output:

| Judul                    | Nama_Penerbit         |
|--------------------------|-----------------------|
| Tip & Trik Delphi        | Elex Media Komputindo |
| Step by step Access 2000 | Elex Media Komputindo |
| Manajemen Umum           | Eksekutif             |
| Jalan Menuju Syurga      | Elex Media Komputindo |
| Kamus Internet           | Elex Media Komputindo |

## Kesimpulan

Berdasarkan tutorial yang disajikan dapat ditarik kesimpulan sebagai berikut:

- ✓ Terdapat 2 jenis bahasa *database*, DDL (*data definition language*) dan DML (*data manipulation language*). SQL merupakan salah satu contoh bahasa *database* yang populer.
- ✓ Terdapat 2 jenis DML, prosedural dan non-prosedural. SQL merupakan bahasa *database* yang non-prosedural.
- ✓ Perintah SQL dapat digunakan untuk perintah DDL dan DML. Dalam penerapannya untuk DML, SQL dapat digunakan untuk operasi *database* yang melibatkan 1 atau lebih tabel dan sub-query (*nested query*).