

SQL Syntax Basic

Wempi Satria

te_no_net@yahoo.com

http://www.geocities.com/te_no_net

Lisensi Dokumen:

Copyright © 2004 IlmuKomputer.Com

Seluruh dokumen di IlmuKomputer.Com dapat digunakan, dimodifikasi dan disebarkan secara bebas untuk tujuan bukan komersial (nonprofit), dengan syarat tidak menghapus atau merubah atribut penulis dan pernyataan copyright yang disertakan dalam setiap dokumen. Tidak diperbolehkan melakukan penulisan ulang, kecuali mendapatkan ijin terlebih dahulu dari IlmuKomputer.Com.

Database relational besar seperti Oracle, SQL Server, Informix, Sybase dan lain-lain biasanya mendukung SQL, dimana SQL merupakan bahasa standar sebagai interface bagi suatu aplikasi untuk berinteraksi dengan database relasional. Dalam tulisan ini penulis akan memaparkan dasar-dasar syntax SQL.

I. Data Defenition Language (DDL) / Pembentukan database

Membuat tabel (*Creating tables*)

Syntax

```
CREATE TABLE <nama_tabel> (  
  <nama_kolom> <tipe_data>(<panjang_data>  
  [UNIQUE] [NOT NULL] [PRIMARY KEY] [DEFAULT<nilai_default>]  
  [referential_constraint_defenition>] [CHECK<constraint_defenotion>],  
  <nama_kolom> <tipe_data>(<panjang_data>  
  [UNIQUE] [NOT NULL] [PRIMARY KEY] [DEFAULT<nilai>]  
  [referential_constraint_defenition>] [CHECK<constraint_defenition>],  
  ...  
);
```

keterangan

Unique; Pada kolom tersebut tidak boleh ada data yang sama.

Not Null; tidak boleh data pada kolom tersebut bernilai null

Unique dan *Not Null*; kolom tersebut dapat dijadikan *primary key*.

Default; nilai default yang secara otomatis akan mengisi kolom dengan data default tersebut setiap operasi insert dilakukan.

Referential_Constraint_Definition; Bila kolom tersebut merupakan *foreign key* terhadap tabel lain. Dengan syntax

FOREIGN KEY <nama_kolom> REFERENCES <nama_tabel>

Contoh :

```
CREATE TABLE Pelajar (  
  No_Induk CHAR(8),  
  Nama CHAR(20),  
  Tgl_Lahir DATE,  
  Kelas CHAR(2)  
);  
CREATE TABLE Mata_Pelajaran(  
  Kode CHAR(4),  
  Nama CHAR(20),  
  Kelas CHAR(2)  
);  
CREATE TABLE Nilai(  
  No_Induk CHAR(8),  
  Kode CHAR(4),  
  NI_Angka Number  
);
```

Membuat index (*Creating indices*)

Syntax

[<nama_kolom> <tipe_data> (<panjang_data>) REFERENCES <nama_tabel>(<nama_kolom>), ...]
CREATE INDEX <nama_index> ON <namatabel>(<nama_kolom>);

Contoh :

```
DROP TABLE Pelajar;  
CREATE TABLE Pelajar (  
  No_Induk CHAR(8) PRIMARY KEY,  
  Nama CHAR(20),  
  Tgl_Lahir DATE,  
  Kelas CHAR(2)  
);  
  
CREATE INDEX nm ON Pelajar(Nama);  
  
DROP TABLE Mata_Pelajaran;  
CREATE TABLE Mata_Pelajaran(  
  Kode CHAR(4) PRIMARY KEY,  
  Nama CHAR(20),  
  Kelas CHAR(2)  
);  
CREATE TABLE Nilai(  
  No_Induk CHAR(8) REFERENCES Pelajar(No_Induk),  
  Kode CHAR(4) REFERENCES Mata_Pelajaran(Kode),  
  Nilai Number  
);
```

Mengubah tabel (*Altering tables*)

Syntax

```
ALTER TABLE <nama_tabel>  
[ ADD (<nama_kolom> < tipe_data>(<panjang_data>),... ); ]  
[ MODIFY (<nama_kolom><tipe_data>(<panjang_data>),... ); ]
```

Keterangan

Add; Penambahan kolom baru.

Modify; Mengubah kolom yang sudah ada sebelumnya.

Contoh :

```
ALTER TABLE Pelajar  
ADD (Jenis_Kelamin CHAR(10));
```

Menghapus tabel (*Dropping tables*)

Syntax

```
DROP TABLE <nama_tabel>  
DROP INDEX <nama_index>
```

Contoh :

```
DROP TABLE Pelajar;  
DROP INDEX nm;
```

II. Data Manipulation Language (DML) / Manipulasi Data

Penyisipan data (*Inserting*)

Syntax

```
INSERT INTO <nama_tabel> [(<nama_kolom1,nama_kolom2,... <nama_kolomN>)]  
VALUES  
(<nilai_kolom1>,<nilai_kolom2>,... <nilai_kolomN>);
```

Contoh :

```
DROP TABLE Pelajar CASCADE CONSTRAINTS;  
CREATE TABLE Pelajar (  
No_Induk CHAR(8) PRIMARY KEY,  
Nama CHAR(20),  
Tgl_Lahir DATE,  
Kelas CHAR(2)  
);  
INSERT INTO Pelajar  
VALUES ('00311217','Wempi Satria','02-JAN-1982','1','Laki-laki');  
INSERT INTO Pelajar  
VALUES ('00311211','Wempi','03-MAR -1982','1','Laki-laki');  
INSERT INTO Pelajar  
VALUES ('00311210','Satria','12-DEC -1982','1','Perempuan');
```

Mengubah data (*Updating*)

Syntax

```
UPDATE <nama_tabel>  
SET <nama_kolom1= 'nilai_kolom1'>,  
<nama_kolom2= 'nilai_kolom2'>,  
... ,  
<nama_kolomN= 'nilai_kolomN'>  
[WHERE <kondisi>];
```

Contoh :

```
UPDATE Pelajar  
SET No_Induk = '00311216' ,Nama = 'Wati'  
WHERE No_Induk ='00311210' and Nama = 'Satria';
```

Menghapus data (*Deletion*)

Syntax

```
DELETE FROM <nama_tabel>  
WHERE <kondisi>;
```

Contoh :

```
DELETE FROM Pelajar  
WHERE No_Induk = '00311211';
```

Seleksi data (*Selection*)

Syntax

```
SELECT [*] [<kolom1>, <kolom2>, . . . , <kolomN>]  
[<alias.kolom1>, <alias.kolom2>, . . . , <alias.kolomN>]  
FROM <nama_tabel>  
WHERE <kondisi>  
[AND <kondisi>]  
[AND MONTH_BETWEEN (<kondisi>);
```

Contoh :

```
SELECT * FROM Pelajar;  
SELECT a.No_Induk, a>Nama, b.Kode, b>Nama, c.Nl_Angka  
FROM Pelajar a, Mata_Pelajaran b, Nilai c;  
WHERE a.No_Induk=c.No_Induk and b.Kode=c.kode;
```

Membuat tabel maya (*Creating views*)

Syntax

```
CREATE VIEW <nama_view>  
AS SELECT <kolom1, kolom2, . . . , kolomN>  
FROM <nama_tabel>  
WHERE <kondisi>;
```

III. Data Control Language (DCL) / Kontrol Data
Konfirmasi menyimpan data di *memory* ke *database* (*Commit*)

Syntax

```
COMMIT [WORK];
```

Contoh :

```
INSERT INTO Pelajar  
VALUES ('00311210','Satria','15-DEC -1982','1','Perempuan');  
COMMIT;
```

Mengembalikan status transaksi sebelum penyimpanan (*Rollback*)

Syntax

ROLLBACK [WORK];

Pemberian hak dari satu *user* ke *user* lain (*Grant*)

Syntax

```
GRANT <spesifikasi_akses>  
ON <nama_tabel/nama_view> TO <nama_user>  
[WITH GRAN OPTION];
```

Penghapusan hak yang diberikan (*Revoke*)

Syntax

```

REVOKE <spesifikasi_akses>
      FROM <nama_user>;

```

Spesifikasi akses

All Privileges; Semua hak diberikan.

Select; Untuk seleksi

Update; Untuk mengubah data

Insert; Untuk menyisipkan data

Delete; Untuk menghapus data

IV. EKSPRESI

FROM

Untuk mendefinisikan tabel yang menjadi sumber data dari suatu perintah seleksi

Contoh : SELECT * FROM Pelajar

WHERE

Untuk mendefenisikan kondisi pengambilan data dari suatu perintah seleksi

Contoh : `SELECT * FROM Pelajar
WHERE No Induk = '00311217';`

GROUP BY

Untuk Mengelompokkan data berdasarkan ekspresi group

Syntax : SELECT <kolom1, kolom2, ... , kolomN>
 FROM <nama_tabel>
 WHERE <kondisi>
 GROUP BY <group_kolom>;

```
Contoh : SELECT a.No_Induk, b.Nama, c.Nl_Angka
FROM Pelajar.a, Nilai b
WHERE a.No_Induk=c.No_Induk and b.kode=c.kode
GROUP BY a.No_Induk, b.Nama, c.Nl_Angka;
```

ORDER BY

Untuk mengurutkan data hasil seleksi

Syntax : SELECT <kolom1, kolom2, ... , kolomN>
 FROM <nama_tabel>
 WHERE <kondisi>
 ORDER BY <nama_kolom> [DESC];

Contoh : SELECT * FROM Pelajar
 ORDER BY No_Induk;

HAVING

Untuk mendefenisikan batasan seleksi berdasarkan GROUP BY

Syntax : SELECT <kolom1, kolom2, ... , kolomN>
 FROM <nama_tabel>
 WHERE <kondisi>
 GROUP BY <group_kolom>
 HAVING <batasan_group>;

Contoh : SELECT a.No_Induk, b>Nama, c.Nl_Angka
 FROM Pelajar.a, Nilai b
 WHERE a.No_Induk=c.No_Induk and b.kode=c.kode
 GROUP BY a.No_Induk, b>Nama, c.Nl_Angka
 HAVING Nilai>80;

V. PREDIKAT COMPARISON

Pembandingan dua nilai dengan syarat type data yang dibandingkan harus sama

Sama dengan	=
Tidak sama dengan	<>
Lebih kecil	<
Lebih besar	>
Lebih kecil dan sama dengan	>=
Lebih besar dan sama dengan	<=

BETWEEN

Pembandingan untuk mengecek apakah suatu nilai berada dalam range tertentu atau tidak

Syntax : ... BETWEEN ... AND ...
 ... NOT BETWEEN ... AND ...

Contoh : Menampilkan data nilai pada range 80 dan 100

```
SELECT * FROM Nilai  
WHERE Nl_Angka BETWEEN 80 AND 100;
```

IN

Untuk melakukan pengecekan apakah suatu nilai terdapat dalam suatu himpunan

Syntax : IN (...)
 IN SELECT ...

Contoh : Select * FROM Pelajar a
 WHERE a.No_Induk IN (SELECT b.No_Induk FROM Nilai b);

LIKE / NOT LIKE

Untuk membandingkan data dengan pola / struktur tertentu, untuk satu karakter dipakai (_)
dan string (%)

Syntax : ... <kolom> LIKE <struktur>
 ... <kolom> NOT LIKE <struktur>
Contoh : SELECT * FROM Pelajar
 WHERE Nama LIKE 'We%';

IS NULL / IS NOT NULL

Untuk membandingkan suatu nilai dengan NULL

Syntax : ... <kolom> IS NULL
 ... <kolom> IS NOT NULL

Contoh : SELECT * FROM Pelajar
 WHERE Kelas IS NULL

EXIST

Untuk pengecekan apakah suatu query memiliki hasil atau tidak

Syntax : ... WHERE EXIST (SELECT ...)
Contoh : SELECT * FROM Pelajar a
 WHERE EXIST (
SELECT b.No_Induk FROM Nilai b
WHERE a.No_Induk=b.No_Induk);

Catatan

- Keyword dari program SQL tidak selalu sama sehingga perlu sedikit modifikasi sesuai dengan standar SQL yang digunakan perusahaan pembuatnya
- Syntax diatas dapat dikembangkan sesuai kebutuhan tergantung kreatifitas dalam pemrograman seperti penambahan sekuens looping, function, procedure, trigger dan lain – lain
- Syntax di atas juga dapat diselipkan pada aplikasi lain seperti web programming, visual programming dan relational programming.
- Tanda [] merupakan optional

Biografi Penulis



Wempi Satria. Lahir di Pasaman pada tanggal 2 Januari 1982, sekarang sedang menyelesaikan studi S1 pada jurusan Sistem Informasi Universitas Putra Indonesia.

Email : te_no_net@yahoo.com