

## **BAB IV**

### **STRUCTURED QUERY LANGUAGE (SQL)**

#### **Tujuan**

1. Mahasiswa dapat memahami dan menerapkan syntax SQL untuk Data Definition Language (DDL).
2. Mahasiswa dapat memahami dan menerapkan syntax SQL untuk Data Manipulation Language (DML).
3. Mahasiswa dapat memahami dan menerapkan syntax SQL untuk Data Control Language (DCL).

#### **4. 1. Data Definition Language (DDL)**

Structured Query Language (SQL) merupakan bahasa yang digunakan untuk mengakses data base. Hampir semua soft ware data base menggunakan SQL sebagai bahasa untuk pengelolaan data base. Query dikelompokkan menjadi 3 kelompok yaitu Data Definition Language (DDL), Data Manipulation Language (DML), Data Control Language (DCL).

DDL adalah query yang digunakan untuk mendefinisikan struktur data base dan table. Query yang termasuk dalam DDL adalah CREATE, ALTER dan DROP. Create digunakan untuk membuat data base dan table. Alter digunakan untuk mengubah data base dan table. Sedangkan Drop digunakan untuk menghapus Data base dan table.

##### **4. 1. 1. Membuat, Menggunakan dan Menghapus Data Base**

Sesuai dengan hirarki data maka yang harus didefinisikan pertama kali adalah database. Untuk membuat data base dapat menggunakan query dengan syntax:

```
CREATE DATABASE nama_database;
```

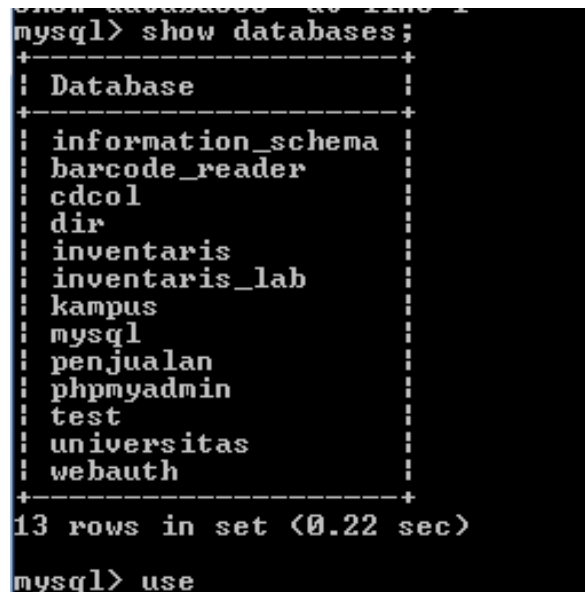
Penamaan database menggunakan aturan yang sama dengan penamaan variable yaitu diawali dengan huruf dan tidak boleh menggunakan spasi. Contoh berikut ini merupakan syntax query untuk membuat data base dengan nama “penjualan”.

```
CREATE DATABASE  Penjualan;
```

Jika ingin melihat data base yang telah dibuat, maka dapat menggunakan query sebagai berikut:

```
SHOW DATABASES;
```

Setelah dieksekusi maka akan tampil data base yang telah dibuat sebelumnya., seperti terlihat pada gambar 4.1



```
mysql> show databases;
+-----+
| Database |
+-----+
| information_schema |
| barcode_reader     |
| cdcol              |
| dir                |
| inventaris          |
| inventaris_lab      |
| kampus              |
| mysql               |
| penjualan           |
| phpmyadmin          |
| test                |
| universitas         |
| webauth             |
+-----+
13 rows in set (0.22 sec)

mysql> use
```

Gambar 4. 1 Daftar Data Base

Sedangkan jika ingin menggunakan / mengakses data base “penjualan” , query yang digunakan adalah sebagai berikut:

```
USE penjualan;
```

Jika perintah atau query di atas berhasil dieksekusi , maka akan ditampilkan pesan sebagai berikut :

```
Database changed;
```

Sedangkan untuk menghapus data base “penjualan” dapat menggunakan query sebagai berikut:

```
DROP DATABASE penjualan;
```

#### **4. 1. 2. Membuat, Mengubah dan Menghapus Table**

Data base yang telah dibuat merupakan container bagi table – table yang menampung data, table dapat dibuat dengan menggunakan query :

```
CREATE TABLE nama_tabel (  
    field1 tipe(panjang),  
    field2 tipe(panjang),  
    ...  
    fieldn tipe(panjang),  
    PRIMARY KEY (field_key)  
);
```

Query tersebut merupakan bentuk umum untuk membuat table, sebagai contoh untuk membuat table pelanggan dengan struktur seperti terlihat pada table 4. 1.

Tabel 4. 1 Tabel Pelanggan

No	Nama Field	Tipe	Panjang
1	<b>id_pelanggan *</b>	Varchar	5
2	nm_pelanggan	Varchar	30
3	alamat	Text	-
4	telepon	Varchar	20
5	email	Varchar	50

Untuk membuat tabel tersebut di atas, query adalah sebagai berikut :

```
CREATE TABLE pelanggan (
  id_pelanggan varchar(5) NOT NULL,
  nm_pelanggan varchar(40) NOT NULL,
  alamat text,
  telepon varchar (20),
  email varchar (50),
  PRIMARY KEY(id_pelanggan)
);
```

Table pelanggan tersebut memiliki satu key (primary key) yaitu id\_pelanggan. id\_pelanggan, nm\_pelanggan, alamat, telepon dan email merupakan nama field. Varchar dan text merupakan tipe data dari field NOT NULL merupakan opsi untuk field yang tidak boleh kosong. Setelah membuat table tersebut maka struktur table dapat dilihat dengan menggunakan query :

```
DESC pelanggan;
```

Setelah dieksekusi maka akan tampil struktur dari table pelanggan.

```
mysql> desc pelanggan;
```

Field	Type	Null	Key	Default	Extra
id_pelanggan	varchar(5)	NO	PRI	NULL	
nm_pelanggan	varchar(40)	NO		NULL	
alamat	text	NO		NULL	
telepon	varchar(20)	NO		NULL	
email	varchar(50)	NO		NULL	

```
5 rows in set (0.06 sec)
```

Gambar 4. 2 Struktur table pelanggan

Bentuk umum query untuk mengubah struktur suatu tabel, adalah sebagai berikut :

```
ALTER TABLE nama_tabel alter_options;
```

ALTER TABLE merupakan perintah dasar untuk mengubah table nama\_tabel merupakan nama tabel yang akan diubah strukturnya. alter\_options merupakan pilihan perubahan tabel. Option yang bisa digunakan, beberapa di antaranya sebagai berikut :

- **ADD definisi\_field\_baru**  
Option ini digunakan untuk menambahkan field baru dengan “definisi\_field\_baru” (nama field, tipe dan option lain). Contoh:  

```
ALTER TABLE pelanggan ADD tgl_lahir date NOT NULL;
```
- **ADD INDEX nama\_index**  
Option ini digunakan untuk menambahkan index dengan nama “nama\_index” pada tabel.
- **ADD PRIMARY KEY (field\_kunci)**  
Option untuk menambahkan primary key pada tabel
- **CHANGE field\_yang\_diubah definisi\_field\_baru**  
Option untuk mengubah field\_yang\_diubah menjadi definisi\_field\_baru.
- **MODIFY definisi\_field**  
Option untuk mengubah suatu field menjadi definisi\_field. Contoh:

```
ALTER TABLE pelanggan MODIFY tgllahir varchar(8)
NOTNULL;
```

- **DROP nama\_field**

Option untuk menghapus field nama\_field. Contoh: ALTER TABLE pelanggan DROP tgllahir;

Untuk mengubah nama suatu tabel, dapat menggunakan perintah SQL sebagai berikut:

```
RENAME TABLE pelanggan TO plg;
ALTER TABLE plg RENAME TO pelanggan;
```

Perintah di atas akan mengubah tabel pelanggan menjadi plg dan sebaliknya. Untuk menghapus sebuah tabel, bentuk umum dari perintah SQL adalah sebagai berikut :

```
DROP TABLE nama_tabel;
```

Contohnya untuk menghapus tabel dengan nama “pelanggan” maka perintah adalah :

```
DROP TABLE pelanggan;
```

## **4. 2. Data Manipulation Language (DML)**

DML merupakan perintah SQL yang digunakan untuk mengelola (menambah, menghapus, menampilkan dan mengubah ) data pada table. Untuk menambah data dalam table dapat menggunakan salah satu dari bentuk umum query berikut ini:

1. INSERT INTO nama\_tabel VALUES ( 'nilai1', 'nilai2', ... );  
untuk menambahkan data (record) dengan jumlah nilai sama dengan jumlah field dalam table .
2. INSERT INTO nama\_tabel (field1, field2, ...) VALUES  
( 'nilai1', 'nilai2', ... ); Untuk menambahkan data pada table, pada field tertentu pada table. Syntax ini adalah syntax yang lengkap.

3. `INSERT INTO nama_tabel SET field1='nilai1', field2='nilai2', ...;` pada syntax ini antara field dan nilai langsung dipasangkan.

Contoh query untuk menambahkan data pada table pelanggan:

```
INSERT INTO `penjualan`.`pelanggan` (  
  `id_pelanggan` ,  
  `nm_pelanggan` ,  
  `alamat` ,  
  `telepon` ,  
  `email`  
)  
VALUES (  
  'AK07', 'Reza  
P', 'Singkawang', '567890', 'reza@gmail.com');
```

Setelah dieksekusi maka pada table pelanggan akan bertambah satu data dengan rincian seperti pada table berikut:

Table 4. 2 Hasil eksekusi

			id_pelanggan	nm_pelanggan	alamat	telepon	email
<input type="checkbox"/>			AK01	Yeremias	Sinkawangng	123456	yermias@gmail.com
<input type="checkbox"/>			AK02	M. Hatta	Singkawang	234567	hatta@gmailcom
<input type="checkbox"/>			AK03	Ema	Singkawang	345678	ema@gmail.com
<input type="checkbox"/>			AK04	Purwo	Sigkawang	456789	purwo.gmai.com
<input type="checkbox"/>			AK05	Sulastina	Singkawang	567890	Sulastina@gamil.com
<input type="checkbox"/>			AK06	Bella	Pontianak	987654	bella@yahoo.com
<input type="checkbox"/>			AK07	Reza P	Singkawang	567890	reza@gmail.com

Saat diperlukan perubahan data, baik dikarenakan salah menambahnya data pada table atau karena telah terjadi perubahan. Misalkan pada table pelanggan terjadi perubahan data dengan id pelanggan AK07 pindah alamat menjadi Sambas. Hal ini dapat dilakukan dengan menggunakan syntax untuk melakukan update data dengan bentuk umum sebagai berikut:

```
UPDATE nama_tabel SET field1='nilaibaru' [WHERE kondisi];
```

Sesuai dengan keperluan tersebut maka query untuk mengubah alamat pada data dengan Id pelanggan AK07 adalah sebagai berikut:

```
UPDATE    pelanggan    SET    alamat='Sambas'    WHERE    id  
pelanggan='AK07' ;
```

Setelah dieksekusi maka nilai alamat pada pelanggan dengan id\_pelanggan AK07 yang sebelumnya Singkawang berubah menjadi Sambas. Untuk mengubah nilai dari beberapa field sekaligus, gunakan koma (,) untuk memisahkan masingmasing field.

Untuk melakukan penghapusan dapat menggunakan perintah DELETE. Sebelum melakukan penghapusan data harus dilakukan dengan hati – hati karena jika terjadi kesalahan query maka data yang telah dihapus tidak bisa kembali lagi (tidak bisa *undo*). Berikut ini adalah bentuk umum dari syntax *delete* :

```
DELETE FROM nama_tabel [WHERE kondisi];
```

Kondisi pada klausa where harus ada jika tidak ada maka semua data pada table akan hilang. Sebagai contoh jika akan menghapus data pada table pelanggan dengan id\_pelanggan = AK07, maka query untuk menghapus data tersebut adalah sebagai berikut:

```
DELETE FROM pelanggan WHERE id_pelanggan = 'AK07';
```

Semua proses pengelolaan data baik penambahan, penghapusan, dan perubahan data hasil eksekusinya dapat dilihat dengan menampilkan data pada table. Hal ini dapat dilakukan menggunakan query dengan bentuk umum sebagai berikut:



```
SELECT [field | *] FROM nama_tabel [WHERE kondisi];
```

Jika ingin menampilkan semua data (field) maka pada opsi field diganti dengan \* sedangkan jika hanya ingin menampilkan field tertentu maka pada opsi field tersebut diisi dengan field yang ingin di tampilkan.

```
SELECT * FROM pelanggan;
```

Setelah dieksekusi maka akan tampil semua data pada table pelanggan, seperti table 4.3.

Tabel 4. 3 Tabel Pelanggan.

id_pelanggan	nm_pelanggan	alamat	telepon	email
AK01	Yeremias	Sinkawanng	123456	yermias@gmail.com
AK02	M. Hatta	Singkawang	234567	hatta@gmailcom
AK03	Ema	Singkawang	345678	ema@gmail.com
AK04	Purwo	Sigkawang	456789	purwo.gmai.com
AK05	Sulastina	Singkawang	567890	Sulastina@gamil.com
AK06	Bella	Pontianak	987654	bella@yahoo.com
AK07	Reza P	Sambas	567890	reza@gmail.com

Untuk menampilkan field tertentu dapat dilakukan dengan merubah query yaitu mengganti \* dengan nama field. Jika terdapat lebih dari satu field maka antara field satu dan yang lainnya dipisahkan dengan tanda koma (,).

```
SELECT id_pelanggan, nm_pelanggan FROM pelanggan;
```

Hasil eksekusi dari query tersebut akan menampilkan field id\_pelanggan dan nm\_pelanggan, seperti pada table berikut.

Tabel 4. 4. Tabel pelanggan dengan field id pelanggan dan nm\_pelanggan

id_pelanggan	nm_pelanggan
AK01	Yeremias
AK02	M. Hatta
AK03	Ema
AK04	Purwo
AK05	Sulastina
AK06	Bella
AK07	Reza P

Klausula where dapat digunakan untuk memberikan batasan atau kondisi tertentu dalam menampilkan data pada table. Query adalah contoh untuk menampilkan data pada table pelanggan dengan alamat Pontianak.

```
SELECT * FROM `pelanggan` WHERE alamat = 'pontianak';
```

Hasil eksekusinya adalah sebagai berikut:

AK06	Bella	Pontianak	987654	bella@yahoo.com
------	-------	-----------	--------	-----------------

Klausula where juga dapat digunakan untuk memberikan batasan atau kondisi data yang mirip.

Contoh:

```
SELECT * FROM `pelanggan` WHERE email LIKE '%gmail%';
```

Tabel 4. 5 Pelanggan dengan email gmail

id_pelanggan	nm_pelanggan	alamat	telepon	email
AK01	Yeremias	Sinkawanng	123456	yermias@gmail.com
AK02	M. Hatta	Singkawang	234567	hatta@gmailcom
AK03	Ema	Singkawang	345678	ema@gmail.com
AK07	Reza P	Sambas	567890	reza@gmail.com

Kondisi pada klausa where juga bisa lebih dari 2 dengan menghubungkan antara 2 kondisi dengan operator logika.

Contoh:

```
SELECT * FROM `pelanggan` WHERE alamat = 'singkawang' &&
email LIKE '%gmail%;
```

Table 4. 6 Hasil eksekusi dengan 2 kondisi

id_pelanggan	nm_pelanggan	alamat	telepon	email
AK02	M. Hatta	Singkawang	234567	hatta@gmailcom
AK03	Ema	Singkawang	345678	ema@gmail.com

Dalam menampilkan data, untuk kasus tertentu diperlukan tampilan data yang bersifat ringkasan. Hal ini dapat dilakukan dengan menggunakan fungsi agregasi pada mysql. Untuk pembahasan tentang fungsi agregasi ini digunakan data pada table produk seperti yang terlihat pada table 4.7

Table 4. 7. Tabel Produk

id_produk	nm_produk	satuan	harga	stock
PAK01	Buku Tulis	Pak	40000	20
PAK02	Pencil 2B	Pak	60000	5
PAK03	Penggaris Besi	buah	50000	12
PAK04	Penghapus	buah	2000	15

Fungsi agregasi yang berkaitan dengan perhitungan data adalah seperti pada table 4.8.

Tabel 4. 8. Fungsi Agregasi

Syntax / Bentuk Umum	Keterangan
SELECT SUM ( kolom) FROM tabel	Menghitung total nilai ekspresi yang tidak NULL.
SELECT AVG( kolom) FROM tabel	Menghitung rata-rata nilai ekspresi yang tidak NULL.
SELECT COUNT( kolom) FROM tabel	Menghitung banyaknya baris yang dipilih secara query.
SELECT MAX( kolom) FROM tabel	Mencari nilai maksimal dari suatu kolom.
SELECT MIN( kolom) FROM tabel	Mencari nilai minimal dari suatu kolom
SELECT FIRST(kolom) AS [ekspresi] FROM tabel	Menampilkan nilai dari record yang pertama dari suatu kolom/field.
SELECT LAST(kolom) AS [ekspresi] FROM tabel	Menampilkan nilai dari record yang terakhir dari suatu kolom/field.

Contoh:

<b>SELECT</b> SUM( stock ) <b>FROM</b> produk;	→hasil :52
<b>SELECT</b> AVG( stock ) <b>FROM</b> produk;	→hasil :13
<b>SELECT</b> MAX( stock ) <b>FROM</b> produk;	→hasil :20
<b>SELECT</b> MIN( stock ) <b>FROM</b> produk;	→hasil :5

Fungsi ORDER BY. Dengan fungsi ini data pada table juga bisa diurutkan sesuai dengan field tertentu dengan urutan naik atau urutan turun

Contoh:

```
SELECT * FROM `pelanggan` ORDER BY nm_pelanggan ASC;
```

```
SELECT * FROM `pelanggan` ORDER BY nm_pelanggan DESC;
```

Tabel 4. 9. Hasil eksekusi pengurutan Ascending

id_pelanggan	nm_pelanggan	alamat	telepon	email
AK06	Bella	Pontianak	987654	bella@yahoo.com
AK03	Ema	Singkawang	345678	ema@gmail.com
AK02	M. Hatta	Singkawang	234567	hatta@gmailcom
AK04	Purwo	Sigkawang	456789	purwo.gmai.com
AK07	Reza P	Sambas	567890	reza@gmail.com
AK05	Sulastina	Singkawang	567890	Sulastina@gamil.com
AK01	Yeremias	Sinkawanng	123456	yermias@gmail.com

Tabel 4. 10. Hasil eksekusi pengurutan Descending

id_pelanggan	nm_pelanggan	alamat	telepon	email
AK01	Yeremias	Sinkawanng	123456	yermias@gmail.com
AK05	Sulastina	Singkawang	567890	Sulastina@gamil.com
AK07	Reza P	Sambas	567890	reza@gmail.com
AK04	Purwo	Sigkawang	456789	purwo.gmai.com
AK02	M. Hatta	Singkawang	234567	hatta@gmailcom
AK03	Ema	Singkawang	345678	ema@gmail.com
AK06	Bella	Pontianak	987654	bella@yahoo.com

Fungsi group by digunakan untuk membentuk group/kelompok dari tupel-tupel yang memiliki nilai yang sama. Hasil pengelompokan dirutkan secara ascending.

### 4. 3. Data Control Language (DCL)

DCL merupakan query yang digunakan untuk keamanan basis data, dengan membuat user dan memberikan (membatasi ) hak akses user sesuai dengan level user tersebut. Untuk membuat user dan memberikan hak akses dapat dilakukan dengan menggunakan perintah GRANT dan REVOKE. Bentuk umum dari query GRANT dan REVOKE adalah sebagai berikut:

```
GRANT priv_type ON {tbl_name | * | *.* | db_name.*} TO  
user_name [IDENTIFIED BY 'password'] [WITH GRANT OPTION]
```

```
REVOKE priv_type ON {tbl_name | * | *.* | db_name.*}  
FROM user_name.
```





Contoh :

Untuk membuat user dengan nama satriyo, dengan password tiyokrbg , hak akses ALL PREVILIGES untuk data base penjualan.

```
GRANT ALL PRIVILEGES ON penjualan.* TO satriyo@localhost  
IDENTIFIED BY 'tiyokrbg';
```

Hasil eksekusi query tersebut seperti terlihat pada table 4. 11.






Tabel 4. 11 Hasil Eksekusi

 Users having access to "penjualan"					
User	Host	Type	Privileges	Grant	Action
root	127.0.0.1	global	ALL PRIVILEGES	Yes	
root	localhost	global	ALL PRIVILEGES	Yes	
satriyo	localhost	database-specific	ALL PRIVILEGES	No	

Untuk membuat user dengan nama admin yang dapat mengakses semua data base dan dapat memberikan grand pada user lain dengan password admin1, sesuai dengan query berikut:

```
GRANT ALL PRIVILEGES ON *.* TO admin@localhost IDENTIFIED BY
'admin1' WITH GRANT OPTION;
```







Tabel 4.12 Hasil Eksekusi untuk user admin

 Users having access to "penjualan"					
User	Host	Type	Privileges	Grant	Action
admin	localhost	global	ALL PRIVILEGES	Yes	
root	127.0.0.1	global	ALL PRIVILEGES	Yes	
root	localhost	global	ALL PRIVILEGES	Yes	
satriyo	localhost	database-specific	ALL PRIVILEGES	No	

Untuk membuat user dengan nama alya yang hanya dapat create pada data penjualan dengan password admin2, query berikut ini:

```
GRANT CREATE ON penjualan.* TO alya@localhost IDENTIFIED
BY 'admin2';
```

Tabel 4. 13 Hasil eksekusi untuk user alya

 Users having access to "penjualan"					
User	Host	Type	Privileges	Grant	Action
admin	localhost	global	ALL PRIVILEGES	Yes	
alya	localhost	database-specific	CREATE	No	
root	127.0.0.1	global	ALL PRIVILEGES	Yes	
root	localhost	global	ALL PRIVILEGES	Yes	
satriyo	localhost	database-specific	ALL PRIVILEGES	No	

Untuk menghapus user alya, dapat menggunakan query berikut ini:

```
REVOKE CREATE ON penjualan.* FROM alya@localhost;
```

#### **4. 4. Soal Latihan**

1. Bagaimana query untuk membuat data base dengan nama Universitas?
2. Tulislah query untuk membuat table mahasiswa dena field nim type varchar (20), nama typr varchar (50) dan alamat type text.
3. Tambahkan data 1234567, Evi P, Pontianak; 345678, Eva A, Singkawang.
4. Bagaimana query untuk menambah data alamat menjadi Pontianak pada record dengan nama Eva A.
5. Hapuslah data dengan nama Evi P.