

## MODUL II

### DDL (Data Definition Language)

#### 2.1. TUJUAN

1. Memahami konsep dasar Data Definition Language (DDL)
2. Memahami implementasi beberapa perintah Create dari DDL
3. Memahami implementasi beberapa perintah Alter dari DDL

#### 2.2. DASAR TEORI

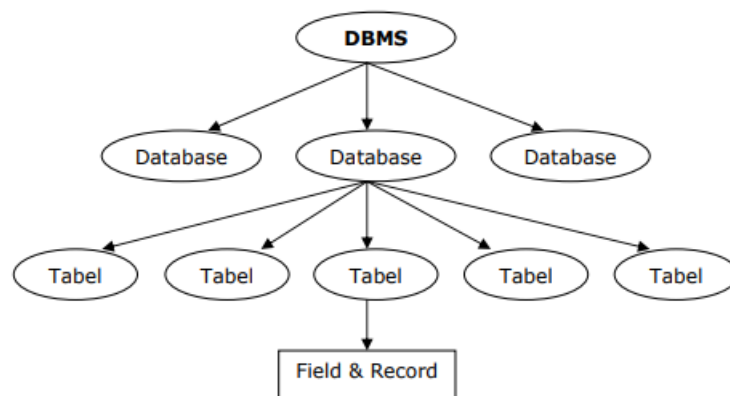
##### 2.2.1. Pengertian DDL

*Data Definition Language* (DDL) adalah kumpulan perintah SQL yang berkaitan dengan pembuatan, perubahan, dan penghapusan *database* maupun objek-objek yang terdapat di dalam *database*. Salah satu bentuk bahasa basis data yaitu *Data Definition Language* (DDL) yang digunakan untuk membuat, mengubah, serta menghapus basis data dan objek-objek yang diperlukan dalam basis data. Metode yang digunakan untuk menerjemahkan kalimat perintah dalam bahasa Indonesia ke *query* DDL adalah metode *Rule-Based*. Proses pada sistem yang dibangun memiliki dua tahap utama yaitu Preprocessing dan *Translasi*. Tahap Preprocessing terdiri dari *case folding*, *filtering*, *tokenizing* kata, *stemming*, dan *removing stopword*. Tahap *Translasi* terdiri dari deteksi kata kunci, *tokenizing* perintah, identifikasi perintah DDL, indentifikasi konten, dan penyusunan *query*

##### 2.2.2. Pengertian SQL

SQL atau *Structured Query Language* merupakan suatu bahasa (*language*) yang digunakan untuk mengakses *database*. SQL sering disebut juga sebagai *query*. Bahasa ini secara *de facto* merupakan bahasa standar yang digunakan dalam manajemen basis data relasional. Saat ini hampir semua server basis data yang ada mendukung bahasa ini untuk melakukan manajemen datanya. Sejarah SQL dimulai dari artikel seorang peneliti dari IBM bernama EF Codd yang membahas tentang ide pembuatan basis data relasional pada bulan Juni 1 1970. Artikel ini juga membahas

kemungkinan pembuatan bahasa standar untuk mengakses data dalam basis data tersebut. Bahasa tersebut kemudian diberi nama SEQUEL (*Structured English Query Language*). Setelah terbitnya artikel tersebut, IBM mengadakan proyek pembuatan basis data relasional berbasis bahasa SEQUEL. Akan tetapi, karena permasalahan hukum mengenai penamaan SEQUEL, IBM pun mengubahnya menjadi SQL. Implementasi basis data relasional dikenal dengan *System/R*. Di akhir tahun 1970-an, muncul perusahaan bernama *Oracle* yang membuat server basis data populer yang bernama sama dengan nama perusahaannya. Dengan naiknya kepopuleran *Oracle*, maka SQL juga ikut populer sehingga saat ini menjadi standar *de facto* bahasa dalam manajemen basis data.



Gambar 2.1 Hierarki Database

### 2.2.3. Membuat Data Base dan menghapus data base

#### a. Membuat Data Base

Sintaks umum SQL untuk membuat suatu database adalah sebagai berikut :

```
CREATE DATABASE [IF NOT EXISTS]
nama_database;
```

Bentuk perintah di atas akan membuat sebuah database baru dengan nama `nama_database`. Aturan penamaan sebuah database sama seperti aturan penamaan sebuah variabel, dimana secara umum nama database boleh terdiri dari huruf, angka dan underscore (`_`). Jika database yang akan dibuat sudah ada, maka akan muncul pesan error. Namun jika ingin otomatis menghapus

database yang lama jika sudah ada, aktifkan option IF NOT EXISTS.

Berikut ini contoh perintah untuk membuat database baru dengan nama “penjualan” :

```
CREATE DATABASE penjualan;
```

Jika query di atas berhasil dieksekusi dan database berhasil dibuat, maka akan ditampilkan pesan kurang lebih sebagai berikut :

```
Query OK, 1 row affected (0.02 sec)
```

#### b. Menghapus Data Base

Untuk menghapus suatu database, sintaks umumnya adalah sbb :

```
DROP DATABASE [IF EXISTS] nama_database;
```

Bentuk perintah di atas akan menghapus database dengan nama nama\_database. Jika databasenya ada maka database dan juga seluruh tabel di dalamnya akan dihapus. Jadi berhati-hatilah dengan perintah ini! Jika nama database yang akan dihapus tidak ditemukan, maka akan ditampilkan pesan error. Aktifkan option IF EXISTS untuk memastikan bahwa suatu database benar-benar ada

### 2.2.4. Stored Procedure

Stored Procedure merupakan suatu kumpulan perintah atau statement yang disimpan dan dieksekusi di server database MySQL. Dengan SP (Stored Procedure), kita dapat menyusun program sederhana berbasis sintaks SQL untuk menjalankan fungsi tertentu. Hal ini menjadikan aplikasi yang kita buat lebih efektif dan efisien.

Stored Procedure dari segi bentuk dan sifatnya terbagi menjadi 2 (dua), yaitu FUNCTION dan PROCEDURE. Perbedaan utama antara function dan Procedure adalah terletak pada nilai yang dikembalikannya (di-return). Function memiliki suatu nilai yang dikembalikan (di-return), sedangkan procedure tidak. Umumnya suatu procedure hanya berisi suatu kumpulan proses yang tidak menghasilkan value, biasanya hanya menampilkan saja.

### 2.2.5. Membuat dan Menghapus Tabel

#### a. Membuat Tabel

Bentuk umum SQL untuk membuat suatu table secara sederhana sebagai berikut :

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

Bentuk umum di atas merupakan bentuk umum pembuatan tabel yang sudah disederhanakan. Penamaan tabel dan field memiliki aturan yang sama dengan penamaan database.

**b. Menghapus table**

Untu`k menghapus sebuah tabel, bentuk umum dari perintah SQL adalah sebagai berikut :

```
DROP TABLE nama_tabel;
```

### 2.2.6. Merubah Struktur Tabel

Untuk mengubah struktur suatu tabel, bentuk umum perintah SQL-nya sebagai berikut :

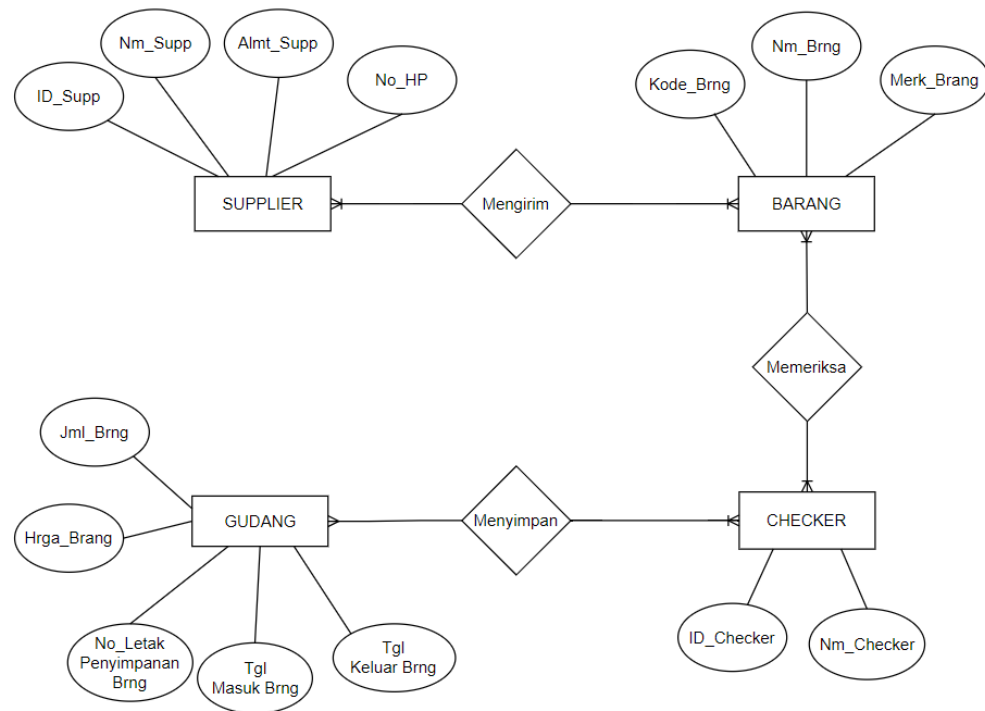
```
ALTER TABLE nama_tabel alter_options;
```

dimana :

- ALTER TABLE merupakan perintah dasar untuk mengubah tabel.
- 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).
- 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
- DROP nama\_field  
Option untuk menghapus field nama\_field
- RENAME TO nama\_tabel\_baru  
Option untuk mengganti nama tabel

## 2.3. DATA HASIL

### 2.3.1. Entity Relationship Diagram



## 2.3.2. Jobsheet

No	Latihan		Query dan Hasil
1	Database Baru	Membuat Database baru	<pre>create database dataGudang;</pre> <pre>MariaDB [(none)]&gt; create database dataGudang; Query OK, 1 row affected (0.002 sec)</pre>
		Melihat database	<pre>show databases;</pre> <pre>MariaDB [(none)]&gt; show databases; +-----+   Database   +-----+   datagudang     information_schema     mysql     performance_schema     phpmyadmin     test   +-----+ 6 rows in set (0.002 sec)</pre>
		Menggunakan database yang telah dibuat	<pre>use dataGudang;</pre> <pre>MariaDB [(none)]&gt; use dataGudang; Database changed MariaDB [dataGudang]&gt; █</pre>
2	Membuat tabel	Membuat table	<pre>create table Supplier( idSupplier int, namaSupplier varchar(50), alamatSupplier varchar(50), noHP varchar(13), primary key (idSupplier));</pre> <pre>MariaDB [dataGudang]&gt; create table Supplier( idSupplier int, namaSupplier varchar(50), alamatSupplier varchar(50), noHP varchar(13), primary key(idSupplier)); Query OK, 0 rows affected (0.014 sec)</pre>
		Membuat tabel	<pre>create table Barang( kodeBarang int, namaBarang varchar(50), jumlahBarang int, hargaBarang int, noTataLetak int, tanggalMasuk int, tanggal keluar int, primary key (kodeBarang));</pre> <pre>MariaDB [dataGudang]&gt; create table Barang( kodeBarang int, namaBarang varchar(50), merekBarang varchar(50), jumlahBarang int, hargaBarang int, noTataLetak int, tanggalMasuk int, tanggalKeluar int, primary key(kodeBarang)); Query OK, 0 rows affected (0.014 sec)</pre>
			<pre>create table Checker( idChecker int, namaChecker varchar(50), primary key (idChecker));</pre>
		Membuat tabel	<pre>MariaDB [dataGudang]&gt; create table Checker( idChecker int, namaChecker varchar(50), primary key(idChecker)); Query OK, 0 rows affected (0.014 sec)</pre>
	Membuat	Membuat	<pre>create index `namaBarang` on `barang`(`namaBarang`);</pre>

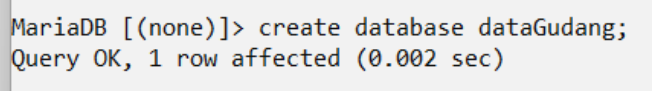
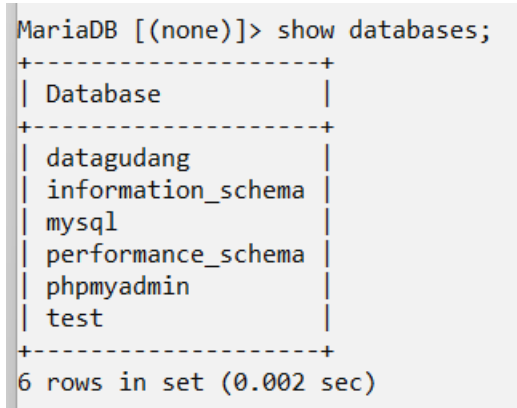
	t Index	t Index	<pre>MariaDB [dataGudang]&gt; create index `namaBarang` on `Barang`(`namaBarang`); Query OK, 0 rows affected (0.009 sec) Records: 0 Duplicates: 0 Warnings: 0  MariaDB [dataGudang]&gt; desc Barang;</pre> <table><tr><th>Field</th><th>Type</th><th>Null</th><th>Key</th><th>Default</th><th>Extra</th></tr><tr><td>kodeBarang</td><td>int(11)</td><td>NO</td><td>PRI</td><td>NULL</td><td></td></tr><tr><td>namaBarang</td><td>varchar(50)</td><td>YES</td><td>MUL</td><td>NULL</td><td></td></tr><tr><td>merekBarang</td><td>varchar(50)</td><td>YES</td><td></td><td>NULL</td><td></td></tr><tr><td>jumlahBarang</td><td>int(11)</td><td>YES</td><td></td><td>NULL</td><td></td></tr><tr><td>hargaBarang</td><td>int(11)</td><td>YES</td><td></td><td>NULL</td><td></td></tr><tr><td>noTataLetak</td><td>int(11)</td><td>YES</td><td></td><td>NULL</td><td></td></tr><tr><td>tanggalMasuk</td><td>int(11)</td><td>YES</td><td></td><td>NULL</td><td></td></tr><tr><td>tanggalKeluar</td><td>int(11)</td><td>YES</td><td></td><td>NULL</td><td></td></tr></table> <pre>8 rows in set (0.006 sec)</pre>	Field	Type	Null	Key	Default	Extra	kodeBarang	int(11)	NO	PRI	NULL		namaBarang	varchar(50)	YES	MUL	NULL		merekBarang	varchar(50)	YES		NULL		jumlahBarang	int(11)	YES		NULL		hargaBarang	int(11)	YES		NULL		noTataLetak	int(11)	YES		NULL		tanggalMasuk	int(11)	YES		NULL		tanggalKeluar	int(11)	YES		NULL	
Field	Type	Null	Key	Default	Extra																																																				
kodeBarang	int(11)	NO	PRI	NULL																																																					
namaBarang	varchar(50)	YES	MUL	NULL																																																					
merekBarang	varchar(50)	YES		NULL																																																					
jumlahBarang	int(11)	YES		NULL																																																					
hargaBarang	int(11)	YES		NULL																																																					
noTataLetak	int(11)	YES		NULL																																																					
tanggalMasuk	int(11)	YES		NULL																																																					
tanggalKeluar	int(11)	YES		NULL																																																					
4	Membuat stored procedure	Membuat	<pre>delimiter // create procedure lihat() begin select*from barang; end //</pre> <pre>MariaDB [dataGudang]&gt; delimiter // MariaDB [dataGudang]&gt; create procedure lihat() -&gt; begin -&gt; select*from Barang; -&gt; end // Query OK, 0 rows affected (0.010 sec)  MariaDB [dataGudang]&gt; delimiter;</pre> <pre>MariaDB [dataGudang]&gt; call lihat();</pre> <table><tr><th>kodeBarang</th><th>namaBarang</th><th>merekBarang</th><th>jumlahBarang</th><th>hargaBarang</th><th>noTataLetak</th><th>tanggalMasuk</th><th>tanggalKeluar</th></tr><tr><td>1</td><td>Stop Kontak</td><td>Broco</td><td>100</td><td>40000</td><td>5</td><td>220522</td><td>0</td></tr><tr><td>2</td><td>Stop Kontak</td><td>mitsubishi</td><td>100</td><td>40000</td><td>5</td><td>220522</td><td>0</td></tr></table> <pre>2 rows in set (0.001 sec)  Query OK, 0 rows affected (0.020 sec)</pre> <pre>delimiter // create procedure lihat (IN kodeInput int) begin select*from barang where kodeBarang=kodeInput; end //</pre> <pre>MariaDB [dataGudang]&gt; create procedure lihatBarang -&gt; (IN kodeInput int) -&gt; begin -&gt; select*from Barang where kodeBarang=kodeInput; -&gt; end // Query OK, 0 rows affected (0.007 sec)  MariaDB [dataGudang]&gt; call lihatBarang(2);</pre> <table><tr><th>kodeBarang</th><th>namaBarang</th><th>merekBarang</th><th>jumlahBarang</th><th>hargaBarang</th><th>noTataLetak</th><th>tanggalMasuk</th><th>tanggalKeluar</th></tr><tr><td>2</td><td>Stop Kontak</td><td>mitsubishi</td><td>100</td><td>40000</td><td>5</td><td>220522</td><td></td></tr></table> <pre>1 row in set (0.003 sec)  Query OK, 0 rows affected (0.015 sec)</pre> <pre>delimiter // create procedure lihat (IN inputKode int, IN inputNama varchar(50), IN inputMerek varchar(20), IN inputJumlah int, IN inputHarga int, IN inputLetak int, IN inputMasuk int, IN inputKeluar int) begin insert into Barang values (inputKode, inputNama, inputMerek, inputJumlah, inputHarga, inputLetak, inputMasuk, inputKeluar); end // delimiter ;</pre>	kodeBarang	namaBarang	merekBarang	jumlahBarang	hargaBarang	noTataLetak	tanggalMasuk	tanggalKeluar	1	Stop Kontak	Broco	100	40000	5	220522	0	2	Stop Kontak	mitsubishi	100	40000	5	220522	0	kodeBarang	namaBarang	merekBarang	jumlahBarang	hargaBarang	noTataLetak	tanggalMasuk	tanggalKeluar	2	Stop Kontak	mitsubishi	100	40000	5	220522															
kodeBarang	namaBarang	merekBarang	jumlahBarang	hargaBarang	noTataLetak	tanggalMasuk	tanggalKeluar																																																		
1	Stop Kontak	Broco	100	40000	5	220522	0																																																		
2	Stop Kontak	mitsubishi	100	40000	5	220522	0																																																		
kodeBarang	namaBarang	merekBarang	jumlahBarang	hargaBarang	noTataLetak	tanggalMasuk	tanggalKeluar																																																		
2	Stop Kontak	mitsubishi	100	40000	5	220522																																																			

			<pre> MariaDB [dataGudang]&gt; delimiter // MariaDB [dataGudang]&gt; create procedure inputBarang -&gt; (IN inputKode int, IN inputNama varchar(50), IN inputMerek varchar(20), IN inputJumlah int, IN inputHarga int, IN inputLetak int, IN inputMasuk int, IN inputKeluar int) -&gt; begin -&gt; insert into Barang values(inputKode, inputNama, inputMerek, inputJumlah, inputHarga, inputLetak, inputMasuk, input tKeluar); -&gt; end // Query OK, 0 rows affected (0.008 sec)  MariaDB [dataGudang]&gt; delimiter ; MariaDB [dataGudang]&gt; call inputBarang(3, "Stop Kontak", "Panasonic", 500, 45000, 3, 250622, " "); Query OK, 1 row affected, 1 warning (0.005 sec)  MariaDB [dataGudang]&gt; call lihat(); +-----+-----+-----+-----+-----+-----+-----+   kodeBarang   namaBarang   merekBarang   jumlahBarang   hargaBarang   noTataLetak   tanggalMasuk   tanggalKeluar   +-----+-----+-----+-----+-----+-----+-----+   1   Stop Kontak   Broco   100   40000   5   220522   0     2   Stop Kontak   mitsubishi   100   40000   5   220522   0     3   Stop Kontak   Panasonic   500   45000   3   250622   0   +-----+-----+-----+-----+-----+-----+-----+ 3 rows in set (0.001 sec)  Query OK, 0 rows affected (0.015 sec) </pre>
5	Melihat tabel	Melihat table yang sudah di buat	<pre> show tables;  MariaDB [dataGudang]&gt; show tables; +-----+   Tables_in_datagudang   +-----+   barang                    checker                    supplier                 +-----+ 3 rows in set (0.000 sec) </pre>
6	Melihat deskripsi tabel	Melihat deskripsi dari tabel	<pre> select * from Supplier;  MariaDB [dataGudang]&gt; select * from Supplier; +-----+-----+-----+-----+   idSupplier   namaSupplier   alamatSupplier   noHP   +-----+-----+-----+-----+   1111   Ahmad Satriadi   Kopang   087728728561     1112   Wilad Surya Mandala   Narmada   087745655657   +-----+-----+-----+-----+ 2 rows in set (0.001 sec) </pre>
		Melihat deskripsi dari tabel	<pre> select * from Barang;  MariaDB [dataGudang]&gt; select * from Barang; +-----+-----+-----+-----+-----+-----+-----+   kodeBarang   namaBarang   merekBarang   jumlahBarang   hargaBarang   noTataLetak   tanggalMasuk   tanggalKeluar   +-----+-----+-----+-----+-----+-----+-----+   1   Stop Kontak   Broco   100   40000   5   220522   0     2   Stop Kontak   mitsubishi   100   40000   5   220522   0   +-----+-----+-----+-----+-----+-----+-----+ 2 rows in set (0.000 sec) </pre>
			<pre> select * from Checker; </pre>
		Melihat deskripsi dari tabel	<pre> MariaDB [dataGudang]&gt; select * from Checker; +-----+-----+   idChecker   namaChecker   +-----+-----+   1000   Otebbb     1001   Sarsana   +-----+-----+ 2 rows in set (0.001 sec) </pre>
7	Merubah tabel	Merubah nama tabel	<pre> alter table Supplier rename to Penyuplai;  MariaDB [dataGudang]&gt; alter table Supplier -&gt; rename to Penyuplai; Query OK, 0 rows affected (0.011 sec) </pre>
		Membuat kolom baru	<pre> alter table Penyuplai add namaSupplier varchar(100);  MariaDB [dataGudang]&gt; alter table Penyuplai -&gt; add namaSupplier varchar(100); Query OK, 0 rows affected (0.007 sec) Records: 0 Duplicates: 0 Warnings: 0 </pre>



		Mengah pus kolom pada tabel	<pre>alter table Penyuplai drop namaSupplier;</pre>
			<pre>MariaDB [dataGudang]&gt; alter table Penyuplai -&gt; drop namaSupplier; Query OK, 0 rows affected (0.009 sec) Records: 0 Duplicates: 0 Warnings: 0</pre>

## 2.4 ANALISA DATA HASIL

No.	Latihan	Query dan Hasil	ANALISA
1	Membuat database	<pre>create database dataGudang;</pre> 	<p>Untuk membuat database menggunakan syntax <code>create database nama_database;</code> dan pada gambar di samping nama databasenya adalah <code>dataGudang;</code></p>
2	Melihat database	<pre>show databases;</pre> 	<p>Syntax untuk melihat database pada MySQL adalah <code>show databases;</code></p>

3	Menggunakan database	<pre>Use dataGudang;</pre> <pre>MariaDB [(none)]&gt; use dataGudang; Database changed MariaDB [dataGudang]&gt; █</pre>	<p>Setelah membuat database selanjutnya adalah menggunakan database dengan syntax <code>use nama_database</code>, dan pada gambar disamping nama databasenya adalah <code>dataGudang</code> sehingga penulisan syntaxnya <code>use dataGudang</code></p>
4	Membuat tabel	<pre>create table Supplier( idSupplier int, namaSupplier varchar(50), alamatSupplier varchar(50), noHP varchar(13), primary key (idSupplier));</pre> <pre>MariaDB [dataGudang]&gt; create table Supplier( idSupplier int, namaSupplier va rchar(50), alamatSupplier varchar (50), noHP varchar(13), primary key(idSupp lier)); Query OK, 0 rows affected (0.014 sec)</pre>	<p>Pada query dan gambar disamping digunakan untuk mebuat tabel diaman nama tabelnya adalah <code>Supplier</code> dan memiliki kolom <code>idSupplier</code> dengan tipe data <code>int</code>, selanjutnya kolom <code>namaSupplier</code> dengan tipe data <code>varchar(50)</code>, dan kolom-kolom selajutnya lalu tutup kurung dan diakhiri titik koma.</p>
5	Membuat index	<pre>create index `namaBarang` on `barang`(`namaBarang`);</pre>	<p>Pada query dan gambar disamping</p>

		<pre> MariaDB [dataGudang]&gt; create index `namaBarang` on `Barang`(`namaBarang`); Query OK, 0 rows affected (0.009 sec) Records: 0 Duplicates: 0 Warnings: 0  MariaDB [dataGudang]&gt; desc Barang; +-----+-----+-----+-----+-----+-----+   Field        Type        Null   Key   Default   Extra   +-----+-----+-----+-----+-----+-----+   kodeBarang   int(11)     NO     PRI   NULL                namaBarang   varchar(50)   YES    MUL   NULL                merekBarang   varchar(50)   YES          NULL                jumlahBarang   int(11)     YES          NULL                hargaBarang   int(11)     YES          NULL                noTataLetak   int(11)     YES          NULL                tanggalMasuk   int(11)     YES          NULL                tanggalKeluar   int(11)     YES          NULL              +-----+-----+-----+-----+-----+-----+ 8 rows in set (0.006 sec) </pre>	<p>digunakan untuk membuat index dengan syntax <code>create index `nama_index` on `nama_tabel`(`nama_kolom`);</code> pada kasus disamping nama index yang dibuat adalah <code>namaBarang</code> pada tabel <code>Barang</code> dibaris <code>namaBarang</code>.</p>
6	Membuat stored procedure	<pre> delimiter // create procedure lihat (IN inputKode int, IN inputNama varchar(50), IN inputMerek varchar(20), IN inputJumlah int, IN inputHarga int, IN inputLetak int, IN inputMasuk int, IN inputKeluar int) begin insert into Barang values (inputKode, inputNama, inputMerek, inputJumlah, inputHarga, inputLetak, inputMasuk, inputKeluar); end // delimiter ; </pre>	<p>Pada query di samping digunakan untuk membuat stored procedure dengan syntax</p> <pre>create procedure nama_prosedure(value) begin perintah_yang_akan_dilakukan end //</pre> <p>pada kasus query disamping digunakan untuk membuat stored procedure menginputkan isi kolom.</p>

		<pre> MariaDB [dataGudang]&gt; delimiter // MariaDB [dataGudang]&gt; create procedure inputBarang -&gt; (IN inputKode int, IN inputNama varchar(50), IN inputMerek varchar(20), IN inputJumlah int, IN inputHarga int, IN inputLetak int, IN inputMasuk int, IN inputKeluar int) -&gt; begin -&gt; insert into Barang values(inputKode, inputNama, inputMerek, inputJumlah, inputHarga, inputLetak, inputMasuk, inputKeluar); -&gt; end // Query OK, 0 rows affected (0.008 sec)  MariaDB [dataGudang]&gt; delimiter ; MariaDB [dataGudang]&gt; call inputBarang(3, "Stop Kontak", "Panasonic", 500, 45000, 3, 250622, " "); Query OK, 1 row affected, 1 warning (0.005 sec)  MariaDB [dataGudang]&gt; call lihat(); +-----+-----+-----+-----+-----+-----+-----+   kodeBarang   namaBarang   merekBarang   jumlahBarang   hargaBarang   noTataLetak   tanggalMasuk   tanggalKeluar   +-----+-----+-----+-----+-----+-----+-----+   1   Stop Kontak   Broco   100   40000   5   220522   0     2   Stop Kontak   mitsubishi   100   40000   5   220522   0     3   Stop Kontak   Panasonic   500   45000   3   250622   0   +-----+-----+-----+-----+-----+-----+-----+ 3 rows in set (0.001 sec)  Query OK, 0 rows affected (0.015 sec) </pre>	
7	Melihat tabel	<pre> Show tables;  MariaDB [dataGudang]&gt; show tables; +-----+   Tables_in_datagudang   +-----+   barang                    checker                   supplier                +-----+ 3 rows in set (0.000 sec) </pre>	<p>Pada query di samping digunakan untuk melihat tabel dengan syntax Show tables; maka akan terlihat tabel yang telah dibuat pada database.</p>
8	Melihat deskripsi tabel	<pre> select*from supplier;  MariaDB [dataGudang]&gt; select*from Supplier; +-----+-----+-----+-----+   idSupplier   namaSupplier   alamatSupplier   noHP   +-----+-----+-----+-----+   1111   Ahmad Satriadi   Kopang   087728728561     1112   Wilad Surya Mandala   Narmada   087745655657   +-----+-----+-----+-----+ 2 rows in set (0.001 sec) </pre>	<p>Pada query disamping digunakn untuk melihat isi dari tabel secara keseluruhan dengan syntax select*from nama_tabel; dan pada gambar di samping terlihat nama tabel yang akan dilihat adalah supplier</p>

			sehingga penulisan querynya adalah select*from supplier;.
9	Merubah nama tabel	<pre>alter table Supplier rename to Penyuplai;</pre> <pre>MariaDB [dataGudang]&gt; alter table Supplier -&gt; rename to Penyuplai; Query OK, 0 rows affected (0.011 sec)</pre>	Untuk merubah nama tabel adalah dengan sintax alter table nama_tabel rename to nama_tabel_baru; pada query disamping nama tabel yang akan diubah adalah Supplier menjadi Penyuplai.
10	Membuat kolom baru	<pre>alter table Penyuplai add namaSupplier varchar(100);</pre> <pre>MariaDB [dataGudang]&gt; alter table Supplier -&gt; rename to Penyuplai; Query OK, 0 rows affected (0.011 sec)</pre>	Untuk membuat kolom baru menggunakan sintax alter table nama_tabel add nama_kolom type_data; pada query disamping nama kolom yang akan di tambahkan adalah namaSupplier dengan tipe data varchar (100) pada tabel Supplier.
11	Menghapus kolom	<pre>alter table Penyuplai drop namaSupplier;</pre> <pre>MariaDB [dataGudang]&gt; alter table Penyuplai -&gt; drop namaSupplier; Query OK, 0 rows affected (0.009 sec) Records: 0 Duplicates: 0 Warnings: 0</pre>	Untuk menghapus kolom adalah dengan sintax alter table nama_tabel drop nama_kolom; pada query disamping nama kolom yang akan dihapus adalah namaSupplier pada tabel Penyuplai.

## 2.5 KESIMPULAN

1. *Data Definition Language* (DDL) adalah kumpulan perintah SQL yang berkaitan dengan pembuatan, perubahan, dan penghapusan *database* maupun objek-objek yang terdapat di dalam database. Salah satu bentuk bahasa basis data yaitu *Data Definition Language* (DDL) yang digunakan untuk membuat, mengubah, serta menghapus basis data dan objek-objek yang diperlukan dalam basis data.

2. Membuat tabel dapat menggunakan perintah CREATE seperti pada sintaks berikut:

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

3. ALTER adalah perintah yang di gunakan untuk mengubah atau memodifikasi tabel yang sudah di buat, seperti pada sintaks berikut:

```
ALTER TABLE nama_tabel alter_options;
```

4. DROP adalah perintah yang digunakan untuk menghapus tabel yang tidak di inginkan, seperti pada sintaks berikut:

```
DROP TABLE nama_tabel;
```

### DAFTAR PUSTAKA

Anwar, Purnamasari. *PENERJEMAHAN TEKS BAHASA  
INDONESIAMENJADI DATA DEFINITION LANGUAGE (DDL)  
DENGAN PENANGANAN KALIMAT MAJEMUK*. Teknik  
Informatika Universitas Komputer Indonesia.

Achmad Solichin. 2010 . *MySQL 5 Dari Pemula hingga Mahir*. Universitas  
Budi Luhur , Jakarta

Muhammad Denny Prayoga. *Pengertian dan Komponen SQL*. Fakultas  
Komputer