

**LATIHAN MODUL 2**  
**LAPORAN PRAKTIKUM BASIS**  
**DATA Data Definition Language (DDL)**



Disusun oleh:

**NAMA: ANTONIUS MUNTHE**

**NIM: 121140032**

**KELAS: BASIS DATA RA**

**PROGRAM STUDI TEKNIK**  
**INFORMATIKA INSTITUT TEKNOLOGI**  
**SUMATERA LAMPUNG SELATAN**

**2022**

# BAB 1

## TEORI DASAR

### A. Data Definition Language (DDL)

DDL (Data Definition Language) adalah kumpulan perintah SQL yang dapat anda gunakan untuk mengelola, mengubah struktur datatype dari objek pada database seperti index, table, trigger, view dan lain sebagainya. MySQL itu sendiri adalah DBMS atau Database management system yang menggunakan bahasa SQL sebagai penghubung antara software dengan database server. DDL digunakan untuk membuat skema, tabel, indeks, dan lain sebagainya.

### B. Perintah Dasar DDL

#### 1. Create

- Create database => Untuk membuat database yang baru
- Create Table => Untuk membuat tabel yang baru
- Create Procedure => Untuk membuat prosedur yang baru
- Create index => Untuk membuat index yang baru
- Create function => Untuk membuat fungsi yang baru
- Create trigger => Untuk membuat reaksi tertentu atau trigger pada data base

#### 2. Alter

Alter merupakan perintah dalam DDL yang digunakan untuk mengubah struktur pada tabel yaitu Alter. Perintah ini digunakan untuk memodifikasi bentuk kolom, menambah dan juga mengganti tabel yang sudah ada

- Menambah Kolom Tabel  
Berikut perintah Alter yang digunakan untuk menambah kolom table.  
`ALTER TABLE nama_table ADD nama_field tipe_data`
- Modifikasi Kolom Tabel  
Berikut perintah Alter yang digunakan untuk memodifikasi kolom table. `ALTER TABLE nama_table MODIFY nama_field tipe_data`
- Menghapus Kolom Tabel  
Berikut perintah alter untuk menghapus kolom table.  
`ALTER TABLE nama_table DROP nama_field`

#### 3. Drop

Drop merupakan perintah dalam DDL yang digunakan untuk membuat perintah menghapus objek dalam database.

- Menghapus Database  
Berikut perintah pada drop yang digunakan untuk menghapus database. `DROP DATABASE nama_database`
- Menghapus tabel  
Berikut perintah pada drop yang digunakan untuk menghapus tabel dalam database. `DROP TABLE nama_table`

## BAB 2

### PEMBAHASAN DAN ANALISIS

#### 1. Membuat Database dengan nama “Bank\_Mino”

Sintaks yang digunakan untuk membuat database yang baru adalah “CREATE DATABASE nama\_database”. Sehingga, untuk membuat database yang baru sesuai dengan nama database di atas ialah dengan cara mengetikkan perintah “CREATE DATABASE Bank\_Mino” pada command prompt atau cmd. Perintah dan hasil tersebut dapat dilihat pada gambar sebagai berikut. Kemudian, ketikkan perintah SHOW DATABASES; untuk melihat list database yang sudah dibuat.

```
MariaDB [(none)]> create database Bank_Mino;  
Query OK, 1 row affected (0.002 sec)
```

```
MariaDB [(none)]> show databases;  
+-----+  
| Database |  
+-----+  
| bank_mino |  
| information_schema |  
| mysql |  
| performance_schema |  
| phpmyadmin |  
| test |  
| universitas |  
+-----+  
7 rows in set (0.002 sec)
```

#### 2. Membuat tabel rekening

Membuat tabel dengan nama rekening pada database Bank\_Mino. Pada Tabel tersebut berisi atribut no rekening, kode cabang, pin, saldo. Atribut no rekening dijadikan sebagai *primary key*. Masing-masing atribut tersebut juga terdapat tipe datanya. Untuk memilih database yang ingin dimanipulasi ialah dengan cara mengetikkan perintah “USE bank\_mino”. Terdapat output “Database changed” yang menyatakan bahwa database tersebut telah berubah atau sedang diakses. Selanjutnya, ketikkan perintah “CREATE table rekening” sesuai sintaks berikut yang diikuti dengan atribut serta tipe data dan juga *primary key* nya. Hal itu dapat dilihat pada gambar sebagai berikut.

```
MariaDB [Bank_Mino]> create table rekening(  
-> no_rekening int(11),  
-> kode_cabang char(5),  
-> pin char(6),  
-> saldo int(11),  
-> PRIMARY KEY (no_rekening) );  
Query OK, 0 rows affected (0.008 sec)
```

```
MariaDB [Bank_Mino]> desc rekening;  
+-----+-----+-----+-----+-----+-----+  
| Field | Type | Null | Key | Default | Extra |  
+-----+-----+-----+-----+-----+-----+  
| no_rekening | int(11) | NO | PRI | NULL | |  
| kode_cabang | char(5) | YES | | NULL | |  
| pin | char(6) | YES | | NULL | |  
| saldo | int(11) | YES | | NULL | |  
+-----+-----+-----+-----+-----+-----+  
4 rows in set (0.026 sec)
```

### 3. Mengubah Struktur Tabel

Berikut cara dan sintaks aturan perubahan struktur pada table rekening

#### a. Tambahkan kolom “nama” setelah kolom kode cabang

Pada tabel rekening, tambahkan atribut nama yang dimana atribut tersebut bertipe char. Kolom atribut tersebut diletakkan setelah kolom kode cabang. Sintaks yang digunakan adalah “ALTER TABLE nama\_table ADD new\_column new-data\_type after name\_column;”. Sehingga, Hal itu dilakukan dengan mengetikkan perintah “ALTER TABLE rekening ADD nama(char) after kode\_cabang;”. Hal itu dapat dilihat pada gambar sebagai berikut.

```
MariaDB [bank_mino]> alter table rekening add nama char(14) after kode_cabang;
Query OK, 0 rows affected (0.030 sec)
Records: 0 Duplicates: 0 Warnings: 0

MariaDB [bank_mino]> desc rekening
-> ;
+-----+-----+-----+-----+-----+-----+
| Field      | Type      | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| no_rekening | int(11)    | NO   | PRI | NULL    |       |
| kode_cabang | char(5)    | YES  |     | NULL    |       |
| nama        | char(14)   | YES  |     | NULL    |       |
| pin         | char(6)    | YES  |     | NULL    |       |
| saldo       | int(11)    | YES  |     | NULL    |       |
+-----+-----+-----+-----+-----+-----+
5 rows in set (0.014 sec)
```

#### b. Tambahkan kolom “keterangan” pada posisi akhir kolom

Pada tabel rekening, tambahkan atribut keterangan yang dimana atribut tersebut bertipe varchar. Kolom tersebut diletakkan pada akhir kolom. Sintaks yang digunakan untuk menambahkan kolom adalah “ALTER TABLE nama\_table add new\_column new-data\_type. Sehingga, hal itu dilakukan dengan mengetikkan perintah “ALTER TABLE rekening add keterangan varchar(20);”. Hal itu dapat dilihat pada gambar sebagai berikut.

```
MariaDB [bank_mino]> alter table rekening add keterangan varchar(20) after saldo;
Query OK, 0 rows affected (0.024 sec)
Records: 0 Duplicates: 0 Warnings: 0

MariaDB [bank_mino]> desc rekening
-> ;
+-----+-----+-----+-----+-----+-----+
| Field      | Type      | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| no_rekening | int(11)    | NO   | PRI | NULL    |       |
| kode_cabang | char(5)    | YES  |     | NULL    |       |
| nama        | char(14)   | YES  |     | NULL    |       |
| pin         | char(6)    | YES  |     | NULL    |       |
| saldo       | int(11)    | YES  |     | NULL    |       |
| keterangan  | varchar(20) | YES  |     | NULL    |       |
+-----+-----+-----+-----+-----+-----+
6 rows in set (0.027 sec)
```

#### c. Ubah kolom nama menjadi “nama\_lengkap”

Pada tabel rekening, ubah atribut nama menjadi nama\_lengkap yang dimana atribut tersebut bertipe varchar. Sintaks yang digunakan adalah ALTER TABLE table\_name CHANGE old\_name\_column new\_name\_new\_column new-data\_type. Sehingga, hal itu dilakukan dengan mengetikkan perintah “ALTER TABLE rekening CHANGE nama nama\_lengkap varchar(50); “. Hal itu dapat dilihat pada gambar sebagai berikut.

```
MariaDB [bank_mino]> ALTER TABLE rekening CHANGE nama nama_lengkap varchar (50);
Query OK, 0 rows affected (0.045 sec)
Records: 0 Duplicates: 0 Warnings: 0

MariaDB [bank_mino]> desc rekening
-> ;
+-----+-----+-----+-----+-----+-----+
| Field      | Type      | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| no_rekening | int(11)    | NO   | PRI | NULL    |       |
| kode_cabang | char(5)    | YES  |     | NULL    |       |
| nama_lengkap | varchar(50) | YES  |     | NULL    |       |
| pin         | char(6)    | YES  |     | NULL    |       |
| saldo       | int(11)    | YES  |     | NULL    |       |
| keterangan  | varchar(20) | YES  |     | NULL    |       |
+-----+-----+-----+-----+-----+-----+
6 rows in set (0.019 sec)
```

d. Hapus kolom keterangan

Pada tabel rekening, hapus atribut keterangan. Sintaks yang digunakan adalah “ALTER TABLE nama\_table drop nama\_kolom”. Hal itu dilakukan dengan mengetikkan perintah “ALTER TABLE rekening drop keterangan; “. Hal itu dapat dilihat pada gambar sebagai berikut.

```
MariaDB [bank_mino]> ALTER TABLE rekening drop keterangan;
Query OK, 0 rows affected (0.026 sec)
Records: 0 Duplicates: 0 Warnings: 0

MariaDB [bank_mino]> desc rekening
-> ;
+-----+-----+-----+-----+-----+-----+
| Field      | Type      | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| no_rekening | int(11)    | NO   | PRI | NULL    |       |
| kode_cabang | char(5)    | YES  |     | NULL    |       |
| nama_lengkap | varchar(50) | YES  |     | NULL    |       |
| pin         | char(6)    | YES  |     | NULL    |       |
| saldo       | int(11)    | YES  |     | NULL    |       |
+-----+-----+-----+-----+-----+-----+
5 rows in set (0.014 sec)
```

4. Buatlah tabel dengan nama “transaksi” dengan struktur tabel berikut. Membuat tabel dengan nama transaksi pada database Bank\_Mino. Pada Tabel tersebut berisi atribut no transaksi, no rekening, jenis transaksi, jumlah. Atribut no transaksi dijadikan sebagai *primary key*. Masing-masing atribut tersebut juga terdapat tipe datanya.

Selanjutnya, ketikkan perintah “CREATE table transaksi” sesuai sintaks berikut yang diikuti dengan atribut serta tipe data dan juga primary keynya. Hal itu dapat dilihat pada gambar sebagai berikut.

```
MariaDB [bank_mino]> create table transaksi (
-> no_transaksi int(11),
-> no_rekening int(11),
-> jenis_transaksi char(10),
-> jumlah int(11),
-> primary key(no_transaksi),
-> foreign key(no_rekening) references rekening(no_rekening) );
Query OK, 0 rows affected (0.048 sec)

MariaDB [bank_mino]> desc transaksi
-> ;
+-----+-----+-----+-----+-----+-----+
| Field      | Type      | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| no_transaksi | int(11)    | NO   | PRI | NULL    |       |
| no_rekening  | int(11)    | YES  | MUL | NULL    |       |
| jenis_transaksi | char(10)   | YES  |     | NULL    |       |
| jumlah       | int(11)    | YES  |     | NULL    |       |
+-----+-----+-----+-----+-----+-----+
4 rows in set (0.009 sec)
```

Selanjutnya, pada tabel transaksi tersebut, kolom no rekening direlasikan dengan tabel rekening melalui kolom no rekening. Sintaks yang digunakan untuk merelasikan hal tersebut ialah “ALTER TABLE table\_name ADD FOREIGN KEY(column\_name\_of\_foreign\_key) kemudian diikuti dengan REFERENCES table\_name(column\_name\_as\_references) ON UPDATE RESTRICT, ON UPDATE CASCADE; “.

Perintah ON UPDATE RESTRICT itu berfungsi sebagai apabila terdapat penghapusan data pada suatu tabel, maka hal itu tidak akan diperbolehkan jika pada suatu tabel lainnya masih terdapat relasi datanya.

Perintah ON UPDATE CASCADE itu berfungsi sebagai apabila terdapat penghapusan data pada suatu tabel, maka akan secara otomatis akan dapat menghapus data yang sesuai dalam

tabel lainnya.

Kemudian, perintah untuk merelasikan data dapat dilihat pada gambar sebagai berikut.

```
MariaDB [bank_mino]> create table transaksi (
-> no_transaksi int(11),
-> no_rekening int(11),
-> jenis_transaksi char(10),
-> jumlah int(11),
-> primary key(no_transaksi),
-> foreign key (no_rekening) references rekening(no_rekening) );
Query OK, 0 rows affected (0.048 sec)

MariaDB [bank_mino]> desc transaksi
-> ;
+-----+-----+-----+-----+-----+-----+
| Field      | Type      | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| no_transaksi | int(11)   | NO   | PRI | NULL    |       |
| no_rekening  | int(11)   | YES  | MUL | NULL    |       |
| jenis_transaksi | char(10)  | YES  |     | NULL    |       |
| jumlah       | int(11)   | YES  |     | NULL    |       |
+-----+-----+-----+-----+-----+-----+
4 rows in set (0.009 sec)
```

## 5. Ubah struktur tabel “transaksi”

### a. Tambahkan kolom “tanggal” setelah kolom jenis\_transaksi.

Pada tabel transaksi, tambahkan atribut tanggal; yang dimana atribut tersebut bertipe varchar. Kolom atribut tersebut diletakkan setelah kolom jenis\_transaksi. Sintaks yang digunakan untuk menambahkan kolom adalah “ALTER TABLE nama\_table ADD new\_column new-data\_type after column\_name”. Sehingga, Hal itu dilakukan dengan mengetikkan perintah “ALTER TABLE transaksi ADD tanggal varchar(20) after jenis\_transaksi;”. Hal itu dapat dilihat pada gambar sebagai berikut.

```
MariaDB [bank_mino]> alter table transaksi add tanggal varchar(20) after jenis_transaksi;
Query OK, 0 rows affected (0.020 sec)
Records: 0 Duplicates: 0 Warnings: 0

MariaDB [bank_mino]> desc transaksi
-> ;
+-----+-----+-----+-----+-----+-----+
| Field      | Type      | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| no_transaksi | int(11)   | NO   | PRI | NULL    |       |
| no_rekening  | int(11)   | YES  | MUL | NULL    |       |
| jenis_transaksi | char(10)  | YES  |     | NULL    |       |
| tanggal     | varchar(20) | YES  |     | NULL    |       |
| jumlah       | int(11)   | YES  |     | NULL    |       |
+-----+-----+-----+-----+-----+-----+
5 rows in set (0.024 sec)
```

### b. Ubah kolom “tanggal” menjadi kolom “tanggal\_transaksi”.

Pada tabel transaksi, ubah atribut tanggal menjadi tanggal\_transaksi yang dimana atribut tersebut bertipe varchar. Sintaks yang digunakan adalah ALTER TABLE table\_name CHANGE old\_name\_column new\_name\_new\_column new-data\_type. Sehingga, hal itu dilakukan dengan mengetikkan perintah “ALTER TABLE transaksi CHANGE tanggal tgl\_transaksi varchar(20); “. Hal itu dapat dilihat pada gambar sebagai berikut.

```

MariaDB [bank_mino]> ALTER TABLE transaksi CHANGE tanggal tgl_transaksi varchar (20);
Query OK, 0 rows affected (0.014 sec)
Records: 0 Duplicates: 0 Warnings: 0

MariaDB [bank_mino]> desc transaksi;
+-----+-----+-----+-----+-----+-----+
| Field          | Type          | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| no_transaksi  | int(11)       | NO   | PRI | NULL    |       |
| no_rekening   | int(11)       | YES  | MUL | NULL    |       |
| jenis_transaksi | char(10)      | YES  |     | NULL    |       |
| tgl_transaksi | varchar(20)   | YES  |     | NULL    |       |
| jumlah       | int(11)       | YES  |     | NULL    |       |
+-----+-----+-----+-----+-----+-----+
5 rows in set (0.014 sec)

```

### BAB III

#### KESIMPULAN

1. Terdapat penerapan dan cara-cara untuk membuat database yang baru pada MySQL. Sintaks yang digunakan ialah `CREATE DATABASE nama_database`.
2. Terdapat penerapan dan cara-cara untuk membuat tabel beserta atribut dan tipe datanya masing-masing. Sintaks yang digunakan untuk membuat tabel ialah `CREATE TABLE nama_table` kemudian didalamnya terdapat database yang berisi atribut dan tipe datanya serta atribut yang dijadikan sebagai *primary key*.
3. Terdapat penerapan dan cara-cara untuk mengubah struktur tabel pada database. Sintaks yang digunakan ialah `ALTER TABLE table_name CHANGE old_name_column new_name_new_column new-data_type`.
4. Terdapat penerapan dan cara-cara untuk menambahkan kolom struktur tabel pada database. Sintaks yang digunakan ialah `ALTER TABLE nama_table add new_column new data_type`.
5. Terdapat penerapan dan cara-cara untuk menghubungkan suatu tabel dengan tabel lainnya dengan melalui *foreign key*. Sintaks yang digunakan ialah `“ALTER TABLE table_name ADD FOREIGN KEY(column_name_of_foreign_key) kemudian diikuti dengan REFERENCES table_name(column_name_as_references) ON UPDATE RESTRICT, ON UPDATE CASCADE; “`.
6. Perintah `ON UPDATE RESTRICT` itu berfungsi sebagai apabila terdapat penghapusan data pada suatu tabel, maka hal itu tidak akan diperbolehkan jika pada suatu tabel lainnya masih terdapat relasi datanya.
7. Perintah `ON UPDATE CASCADE` itu berfungsi sebagai apabila terdapat penghapusan data pada suatu tabel, maka akan secara otomatis akan dapat menghapus data yang sesuai dalam tabel lainnya.



## **PUSTAKA**

<http://teknik-informatika-s1.stekom.ac.id/informasi/baca/Pengertian-DDL-dan-DML-Beserta-Contoh-Perintahnya-dalam-Database/764d4cc8477ab0f54654ba0e80a71e687a3f987c#:~:text=Untuk%20pengertian%20DDL%20dan%20DML,trigger%2C%20view%20dan%20lain%20sebagainya>

Modul Praktikum Basis Data Pertemuan 2