

3 BAB 3 – DATA DEFINITION LANGUAGE (DDL)

3.1 IDENTITAS

Kompetensi

1. Praktikan memahami SQL dan perintah DDL pada SQL.
2. Praktikan dapat membuat table dengan benar beserta relationshipnya
3. Praktikan memahami type table InnoDB.

Topik

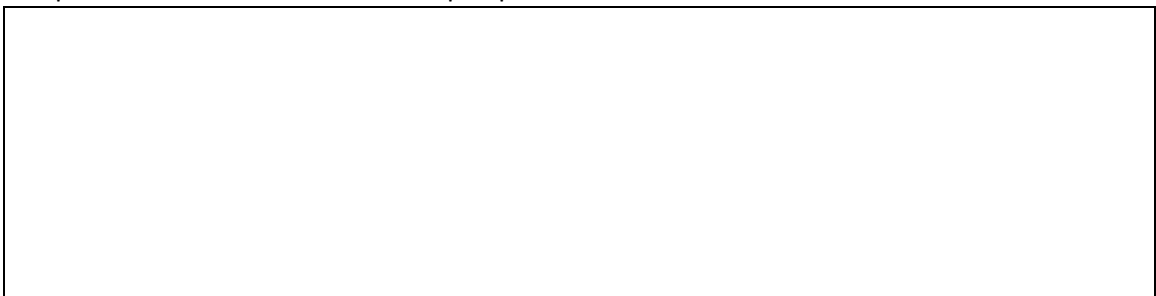
1. SQL
2. Membuat Table
3. Constraint Relasi
4. Nilai otomatis dan nilai default

3.2 TEST AWAL

1. Operasikan perintah SQL untuk :
 - a. Membuat database
 - b. Melihat seluruh database pada mysql server
 - c. Mengakses database/ menggunakan database
 - d. Menghapus database



2. Tampilkan hasil screen shoot dan simpan pada folder PrakDB-NIM



3.3 SQL

Secara umum perintah-perintah yang terdapat di dalam SQL, diklasifikasikan menjadi tiga bagian, antara lain yaitu :

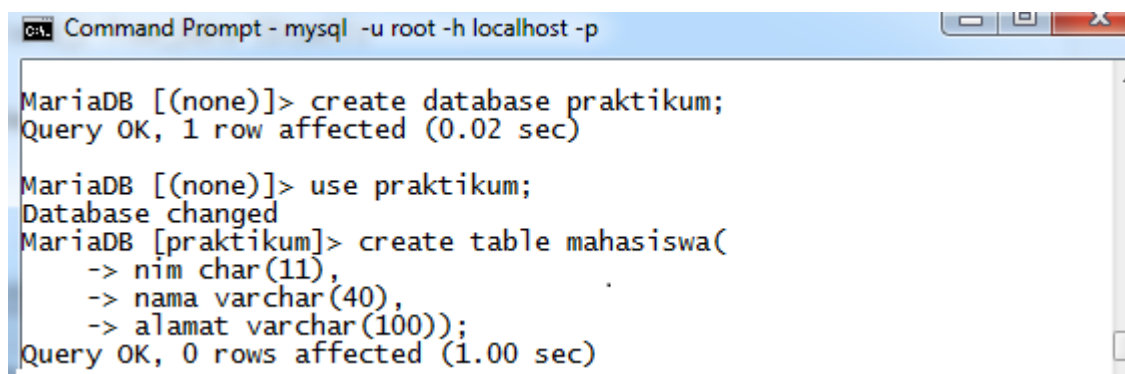
1. DDL (*Data Definition Language*)
 - Merupakan perintah SQL yang berkaitan dengan pendefinisian suatu struktur database, dalam hal ini database dan table.
 - Perintah DDL adalah: CREATE, ALTER, RENAME, DROP.
2. DML (*Data Manipulation Language*)
 - Merupakan perintah SQL yang berkaitan dengan manipulasi atau pengolahan data atau record dalam table.
 - Perintah DML antara lain: SELECT, INSERT, UPDATE, DELETE.
3. DCL (*Data Control Language*)
 - Merupakan perintah SQL yang berkaitan dengan manipulasi user dan hak akses (priviledges).
 - Perintah SQL yang termasuk dalam DCL antara lain: GRANT, REVOKE.

3.4 MEMBUAT TABLE

Setelah menciptakan suatu database dan mengaktifkan database tersebut maka dapat dilakukan perintah pembuatan tabel.

3.4.1 Create Table

- Perintahnya :
Create Table Nama_Table (Nama_Field_1 Tipe_Data (Size),
Nama_Field_2 Tipe_Data (Size));
- Contoh :
mysql > **Create Table** Mahasiswa (NIM char(11),
Nama varchar(40),
Alamat varchar(100));



```
Command Prompt - mysql -u root -h localhost -p
MariaDB [(none)]> create database praktikum;
Query OK, 1 row affected (0.02 sec)

MariaDB [(none)]> use praktikum;
Database changed
MariaDB [praktikum]> create table mahasiswa(
-> nim char(11),
-> nama varchar(40),
-> alamat varchar(100));
Query OK, 0 rows affected (1.00 sec)
```

2.4.2 Melihat Table dan Struktur Table

Untuk melihat seluruh table yang telah dibuat sebelumnya, (Dengan syarat : sudah berada di database yang mempunyai table tersebut).

- perintahnya :
mysql > **Show Tables;**

```
MariaDB [praktikum]> show tables;
+-----+
| Tables_in_praktikum |
+-----+
| mahasiswa            |
+-----+
1 row in set (0.01 sec)
```

Sedangkan untuk melihat struktur dari masing-masing tabel,

- perintahnya :
Desc/Describe Nama_Table ;
- Contoh :
mysql > **Desc Mahasiswa;**

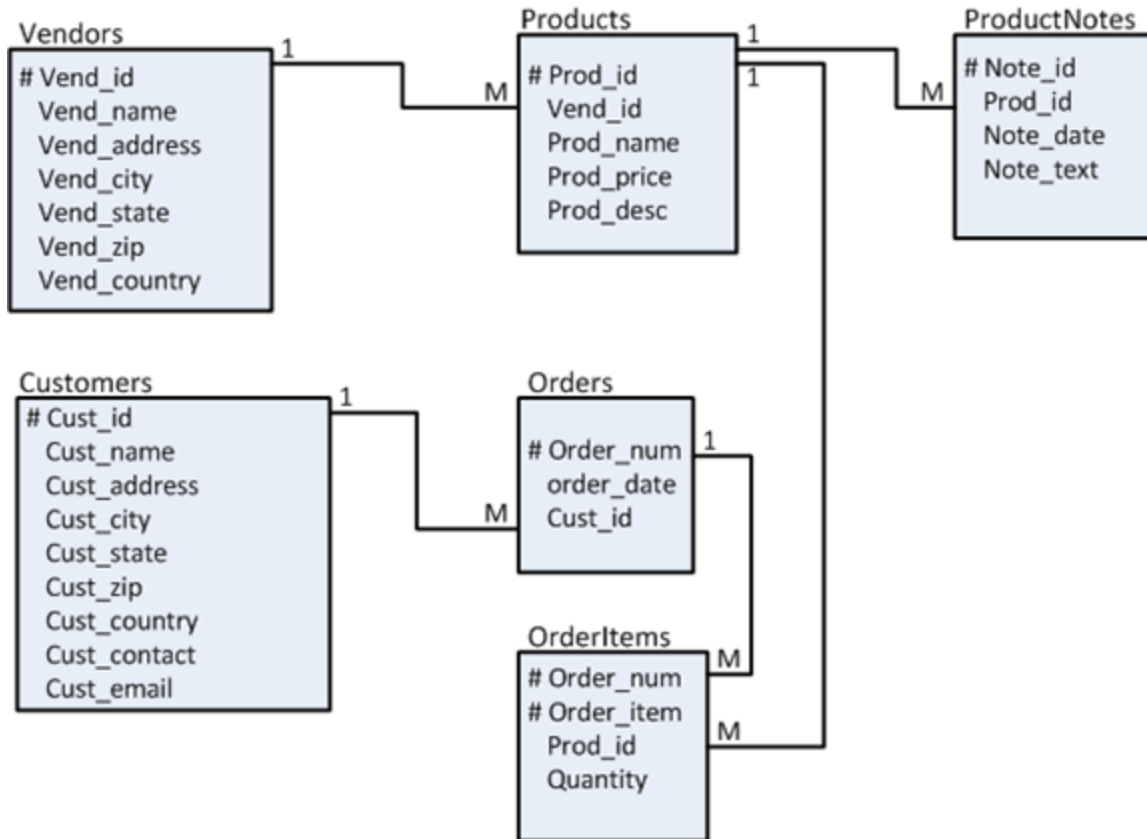
```
MariaDB [praktikum]> desc mahasiswa;
+-----+-----+-----+-----+-----+-----+
| Field | Type          | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| nim   | char(11)      | YES  |     | NULL    |       |
| nama  | varchar(40)   | YES  |     | NULL    |       |
| alamat| varchar(100)  | YES  |     | NULL    |       |
+-----+-----+-----+-----+-----+-----+
3 rows in set (0.03 sec)
```

3.5 CONSTRAINT TABLE

Constraint adalah aturan atau batasan yang sengaja kita terapkan pada table untuk menjaga integritas dan konsistensi data. Ada 5 aturan constraint yang biasanya diterapkan pada table. Constraint ini biasanya diterapkan saat melakukan **create table** atau bisa juga saat **alter table** (dibahas di bab 4).

Berikut ini 5 aturan constraint pada mysql yaitu *primary key, foreign key, unique, not null dan check*.

Untuk lebih memahami penggunaan constraint tersebut kita coba terapkan pada skema order entry. Berikut ini adalah diagram relationship atau relasi antar table dari skema order entry.



Gambar Skema Order Entry

Untuk satu skema kita buat satu database. Untuk menerapkan skema tersebut dalam sebuah database berikut langkah – langkah yang kita lakukan :

1. Create database *order entry*
2. Buat table skema *order entry* dengan memperhatikan urutan, mulailah dari table kuat yaitu table hasil dari entitas tunggal dan tanpa foreign key kemudian dilanjutkan table hasil relasi. Dalam kasus *order entry* urutannya sbb :
 - Vendors, customers
 - Products, orders
 - ProductNotes, orderItems

3.5.1 Penerapan constraint pada Skema Order Entry

1. Terlebih dahulu kita buat database nya dan use databasenya

```

MariaDB [praktikum]> create database OrderEntry;
Query OK, 1 row affected (0.01 sec)

MariaDB [praktikum]> use orderEntry;
Database changed
MariaDB [orderEntry]>
    
```

2. Buat table vendor dan table customer

```

MariaDB [orderEntry]> CREATE TABLE vendors(
-> vend_id CHAR(4) NOT NULL PRIMARY KEY ,
-> vend_name VARCHAR(25) NOT NULL,
-> vend_address VARCHAR(30),
-> vend_city VARCHAR(20),
-> vend_state VARCHAR(5),
-> vend_zip VARCHAR(7),
-> vend_country VARCHAR(15));

```

Query OK, 0 rows affected (1.14 sec)

```

MariaDB [orderEntry]> CREATE TABLE customers(
-> cust_id CHAR(5) NOT NULL PRIMARY KEY,
-> cust_name VARCHAR(25) NOT NULL,
-> cust_address VARCHAR(30) NULL,
-> cust_city VARCHAR(25) NULL,
-> cust_state VARCHAR(5) NULL,
-> cust_zip VARCHAR(5) NULL,
-> cust_country VARCHAR(20) NULL,
-> cust_contact VARCHAR(25) NULL,
-> cust_email VARCHAR(30) NULL);

```

Query OK, 0 rows affected (0.49 sec)

```

MariaDB [orderEntry]> show tables;

```

```

+-----+
| Tables_in_orderentry |
+-----+
| customers             |
| vendors               |
+-----+
2 rows in set (0.27 sec)

```

3. Buat table products dan orders

```

MariaDB [orderEntry]> CREATE TABLE products(
-> prod_id VARCHAR(10) NOT NULL PRIMARY KEY,
-> vend_id CHAR(4) NOT NULL ,
-> prod_name VARCHAR(25) NOT NULL ,
-> prod_price INT NOT NULL ,
-> prod_desc VARCHAR(255) NULL);

```

Query OK, 0 rows affected (0.90 sec)

```

MariaDB [orderEntry]> CREATE TABLE orders(
-> order_num INT NOT NULL ,
-> order_date DATE NOT NULL,
-> cust_id CHAR(5) NOT NULL,
-> PRIMARY KEY(order_num));

```

Query OK, 0 rows affected (0.33 sec)

4. Buat table productnotes dan table orderItems

```

MariaDB [orderEntry]> CREATE TABLE productnotes(
-> note_id CHAR(3) NOT NULL,
-> prod_id VARCHAR(10) NOT NULL,
-> note_date DATE NOT NULL,
-> note_text VARCHAR(200) NULL,
-> PRIMARY KEY (note_id),
-> FOREIGN KEY (prod_id) REFERENCES products (prod_id));

```

Query OK, 0 rows affected (0.36 sec)

Perhatikan table orderItem memiliki containt primary key dua kolom sekaligus karena menerapkan surrogate/kunci pengganti .

```

MariaDB [orderentry]> CREATE TABLE orderitems(
-> order_num INT NOT NULL ,
-> order_item INT NOT NULL ,
-> prod_id VARCHAR(10) NOT NULL ,
-> quantity INT NOT NULL,
-> PRIMARY KEY (order_num, order_item));
Query OK, 0 rows affected (0.45 sec)

MariaDB [orderentry]> Show tables;
+-----+
| Tables_in_orderentry |
+-----+
| customers            |
| orderitems           |
| orders               |
| productnotes         |
| products             |
| vendors              |
+-----+
6 rows in set (0.00 sec)

```

3.6 NILAI OTOMATIS DAN NILAI DEFAULT

3.6.1 Nilai otomatis / Auto Increment

Suatu nilai otomatis merupakan suatu field yang diisi secara otomatis oleh sistem. Biasanya paling banyak digunakan pada *primary key*. Tipe data kolom yang akan diset nilai autoincrement adalah **int**.

Perintahnya :

Auto_Increment

Contoh :

```

mysql> Create Table Mahasiswa2 (
          ID int(5) not null primary key auto_increment,
          NIM char(8) not null,
          Nama_Mhs varchar(50),
          Jurusan varchar(200),
          Fakultas varchar(30));

```

```

MariaDB [praktikum]> Create Table Mahasiswa2 (
-> ID int(5) not null primary key auto_increment,
-> NIM char(8) not null,
-> Nama_Mhs varchar(50),
-> Jurusan varchar(200),
-> Fakultas varchar(30));
Query OK, 0 rows affected (1.00 sec)

MariaDB [praktikum]> show tables;
+-----+
| Tables_in_praktikum |
+-----+
| mahasiswa           |
| mahasiswa2          |
+-----+
2 rows in set (0.03 sec)

MariaDB [praktikum]> desc mahasiswa2;
+-----+-----+-----+-----+-----+-----+
| Field | Type          | Null | Key | Default | Extra          |
+-----+-----+-----+-----+-----+-----+
| ID    | int(5)        | NO   | PRI | NULL    | auto_increment |
| NIM   | char(8)       | NO   |     | NULL    |                 |
| Nama_Mhs | varchar(50)  | YES  |     | NULL    |                 |
| Jurusan | varchar(200) | YES  |     | NULL    |                 |
| Fakultas | varchar(30)  | YES  |     | NULL    |                 |
+-----+-----+-----+-----+-----+-----+
5 rows in set (0.03 sec)

```

3.6.2 Nilai default

Suatu nilai default merupakan pemberian nilai secara otomatis oleh system terhadap suatu field tertentu dengan nilai NULL.

Perintahnya :

```
Default Nilai_Default
```

Contoh :

```
mysql> Create Table Mtkul (  
        Kode_Mtkul int(5) not null primary key,  
        Nama_Mtkul Varchar (30),  
        Sks int(1) default 0,  
        Semester int(1) default 0);
```

```
MariaDB [praktikum]> Create Table Mtkul (  
-> Kode_Mtkul int(5) not null primary key,  
-> Nama_Mtkul Varchar(30),  
-> Sks int(1) default 0,  
-> Semester int(1) default 0);  
Query OK, 0 rows affected (0.23 sec)
```

```
MariaDB [praktikum]> desc mtkul;
```

Field	Type	Null	Key	Default	Extra
Kode_Mtkul	int(5)	NO	PRI	NULL	
Nama_Mtkul	varchar(30)	YES		NULL	
Sks	int(1)	YES		0	
Semester	int(1)	YES		0	

4 rows in set (0.03 sec)

3.7 TYPE : INNODB DAN XTRADB

InnoDB Merupakan storage engine yang sering dipakai di website. MySQL memberikan pilihan beberapa *table engine* untuk setiap tabel yang ada. Sebelum versi 5.5, table yang dibuat dengan CREATE TABLE tanpa menyertakan *table engine* yang akan dipakai secara otomatis akan menggunakan engine MyISAM. Pada versi 5.5, *default table engine* diganti dari MyISAM menjadi InnoDB Storage engine ini sering dikenal karena mempunyai fitur transaksi, seperti commit, rollback dan crash recovery layaknya oracle. Disamping itu juga mempunyai fitur tabel relasi dan integritas – foreignkey. Kekurangan InnoDB adalah membutuhkan resource memori yang besar.

Pada mariaDB InnoDB digantikan dengan XtraDB yang lebih stabil.

Apabila pada satu kondisi anda memerlukan type table dengan type engine InnoDB maka dapat menambahkan sintak berikut pada akhir create table.

```
Engine = InnoDB;
```

Contoh :

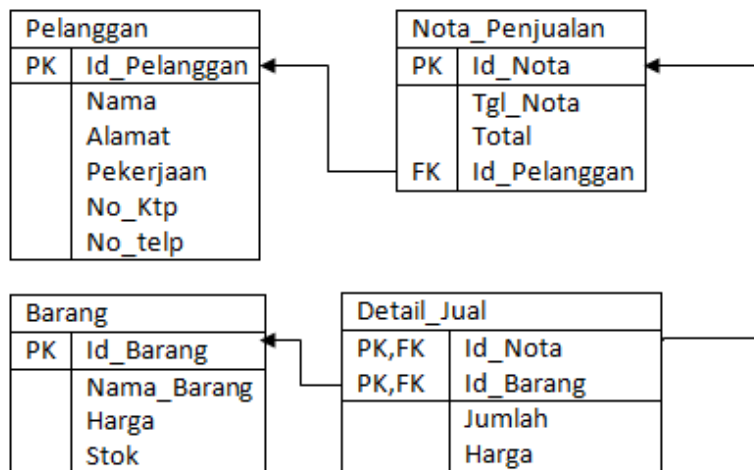
```

MariaDB [praktikum]> create table compress(
  -> x int primary key not null,
  -> y varchar(50) null) engine=InnoDB;
Query OK, 0 rows affected (0.33 sec)

```

3.8 TEST AKHIR

1. Buatlah sebuah database baru dengan nama Penjualan_Barang
2. Gunakan database penjualan_barang kemudian buatlah tabel tabcel hasil dari diagram relationship penjualan barang.
3. Buatlah laporan praktikum dengan ketentuan sbb :
 - a. Nama file laporan : PrakDB_Bab3_NIM.odt
 - b. Isi file laporan :
 - i. Souce sql
 - ii. Screenshot CMD
 - c. Simpan di direktory "PrakDB-NIM"



ERD Penjualan Barang