Modul II

Basisdata Relasional dan DDL

I. Tujuan

a. Mahasiswa mampu menjelaskan definisi dan fungsi data definition language (DDL)
b. Mahasiswa mampu membuat database dan tabel dengan menerapkan data definition language (DDL)
c. Mahasiswa mampu menjelaskan jenis-jenis tipe data

II. Dasar Teori

A. Basisdata Relasional

Model Data Relasional adalah suatu model basis data yang menggunakan tabel dua dimensi, yang terdiri atas baris dan kolom untuk menggambarkan sebuah berkas data. Model ini diperkenalkan pertama kali oleh E.F. Codd.

Atribut	Field	Property	Kolom
Data	Data	Data	Data
Data	Data	Data	Data
Data	Data	Data	Data
Data	Data	Data	Data
Data	Data	Data	Data

Gambar 1. Tabel

A.1. Istilah-Istilah dalam Model Data Relasional

1. Relasi yaitu sebuah tabel yang terdiri dari beberapa kolom dan beberapa baris.
2. Atribut yaitu kolom pada sebuah relasi.
3. Tupel yaitu baris pada sebuah relasi.
4. Domain yaitu kumpulan nilai yang valid untuk satu atau lebih atribut
5. Derajat yaitu jumlah atribut dalam sebuah relasi (jumlah field)
6. Cardinality yaitu jumlah tupel dalam sebuah relasi (jumlah record)

A.2. Relational Keys dalam Model Data Relasional

1. **Super key**

   Satu/kumpulan atribut yang secara unik mengidentifikasi sebuah tupel di dalam relasi (satu atau lebih field yang dapat dipilih untuk membedakan antara 1 record dengan record lainnya).

2. **Candidate key**

   Atribut di dalam relasi yang biasanya mempunyai nilai unik (super key dengan jumlah field yang paling sedikit)

3. **Primary key**

   Candidate key yang dipilih untuk mengidentifikasikan tupel secara unik dalam relasi

4. **Alternate key**

   Candidate key yang tidak dipilih sebagai primary key

5. **Foreign key**

   Atribut dengan domain yang sama yang menjadi kunci utama pada sebuah relasi tetapi pada relasi lain atribut tersebut hanya sebagai atribut biasa.

B. Data Definition Language (DDL)

DDL (Data Definition Languange) merupakan sekumpulan set perintah yang bertujuan untuk mendefinisikan atribut – atribut database, tabel, atribut kolom (field), maupun batasan – batasan terhadap suatu atribut dan relasi/hubungan antar tabel. Yang termasuk dalam kelompok perintah DDL adalah : CREATE, ALTER, dan DROP.

CREATE merupakan perintah DDL yang digunakan untuk membuat database maupun tabel. Nama database maupun tabel tidak boleh mengandung spasi (space). Nama database tidak boleh sama antar database.

ALTER merupakan perintah DDL yang digunakan untuk mengubah nama/struktur tabel.

DROP merupakan perintah DDL yang digunakan untuk menghapus database ataupun tabel.

1. *Basisdata*

Syntax untuk membuat database sebagai berikut :

- CREATE DATABASE namadatabase;

Contoh penulisan : CREATE DATABASE db_kampus;

Sedangkan syntax tambahan untuk menampilkan daftar nama database yang terdapat dalam database server pada MySQL menggunakan perintah :

- SHOW DATABASES;

Sebelum kita membuat suatu tabel yang digunakan untuk menyimpan data, terlebih dahulu harus memilih/mengaktifkan salah satu database sebagai database aktif yang akan digunakan untuk menyimpan beberapa tabel yang akan kita buat. Untuk memilih/mengaktifkan salah satu database menggunakan syntax :

- USE namadatabase;

Contoh penulisan : USE db_kampus;

Selain perintah - perintah di atas, terdapat perintah yang berfungsi untuk menghapus database maupun tabel. Perintah tersebut adalah sebagai berikut :

- DROP DATABASE namadatabase;

Contoh penulisan : DROP DATABASE db_kampus;

2. *Tabel*

- Membuat Tabel

Nama tabel tidak boleh mengandung spasi (space). Ketika membuat tabel ada beberapa yang harus dideklarasikan dalam pembuatannya, yaitu antara lain meliputi : nama tabel, nama Field (Kolom), type data dari field dan panjang data. Adapun syntax yang digunakan untuk membuat tabel secara umum adalah sebagai berikut:

CREATE TABLE namatabel (Field1, TypeData1, Field2, TypeData2);

Contoh berikut syntax untuk membuat tabel ***mahasiswa***:

```
CREATE TABLE mahasiswa (nim CHAR (20), nama_mhs CHAR (50), login CHAR(20), pass
    CHAR(20), umur INT, ipk real, PRIMARY KEY(nim));
```

- Menampilkan Tabel

Untuk menampilkan daftar nama tabel yang terdapat dalam database yang sedang aktif/digunakan menggunakan perintah :

SHOW TABLES;

- Menampilkan deskripsi atribut tabel

Untuk menampilkan deskripsi atribut – atribut yang terdapat pada suatu tabel dengan menggunakan perintah:

DESC namatabel;

misalkan DESC mahasiswa;

- Menghapus Tabel

Untuk menghapus Tabel perintahnya sama dengan untuk menghapus database yaitu dengan menggunakan perintah DROP. Syntax yang digunakan adalah:

DROP TABLE namaTabel;

Tabel yang akan dihapus harus sesuai dengan nama tabel. Misal kita akan menghapus tabel mahasiswa, maka syntax nya adalah:

DROP TABLE mahasiswa;

- Mendefinisikan Null/Not Null

Null ataupun Not Null merupakan pernyataan yang digunakan untuk membuat kolom yang kita buat boleh kosong (Null) atau tidak boleh kosong (Not Null). Ketika pada kolom tabel tidak diset, maka secara default akan bernilai Null (boleh kosong). Untuk mendefinisikannya maka perintah yang digunakan adalah:

CREATE TABLE mahasiswa (nim CHAR (20) NOT NULL, nama_mhs CHAR (50) NOT NULL, login CHAR(20) NOT NULL, pass CHAR(20) NOT NULL, umur INT, ipk real, PRIMARY KEY(nim));

- Mendefinisikan PRIMARY KEY pada Tabel

Suatu keharusan dalam suatu tabel adalah harus memiliki satu kolom yang dijadikan sebagai perwakilan dari tabel tersebut. Pembuatan perwakilan tabel ini berfungsi untuk melakukan hubungan/relasional dengan tabel lain. Bentuk perwakilan ini dalam database disebut sebagai PRIMARY KEY yang aturan pembuatannya adalah sebagai berikut:

- Satu tabel hanya diperbolehkan memiliki satu kolom kunci.
- Nama kolom kunci tidak digunakan pada kolom lain dalam satu tabel
- Nama kolom kunci tidak boleh sama dengan kolom kunci yang ada pada tabel lain
- Bentuk kolom kunci harus diset NOT NULL.

Terdapat tiga cara untuk mendefinisikan primary key. Berikut ini syntax yang digunakan:

CREATE TABLE mahasiswa (nim CHAR (20), nama_mhs CHAR (50), login CHAR(20), pass CHAR(20), umur INT, ipk real, PRIMARY KEY(nim));

Atau

```
CREATE TABLE mahasiswa ( nim CHAR (20) NOT NULL PRIMARY KEY, nama_mhs CHAR (50), login CHAR(20), pass CHAR(20), umur INT, ipk real);
```

atau

```
ALTER TABLE mahasiswa ADD CONSTRAINT namaconstraint PRIMARY KEY(namakolom);
```

- Menghapus PRIMARY KEY pada Tabel

Cara 1: Jika primary key dibuat menggunakan alter table:

```
ALTER TABLE namatabel DROP CONSTRAINT namaconstraint;
```

Cara 2: jika primary key dibuat melalui create table:

```
ALTER TABLE namatabel DROP PRIMARY KEY;
```

C. Tipe Data

Tipe data adalah kelompok (klasifikasi) data berdasarkan jenis-jenis tertentu. Terdapat 3 kelompok data yang sering dipakai dalam database MySQL yakni Karakter, Numerik, dan Waktu & Tanggal.

C.1 Tipe Data Karakter

Tipe Karakter	Penjelasan
CHAR(size)	A FIXED length string (can contain letters, numbers, and special characters). The *size* parameter specifies the column length in characters - can be from 0 to 255. Default is 1
VARCHAR(size)	A VARIABLE length string (can contain letters, numbers, and special characters). The *size*parameter specifies the maximum column length in characters - can be from 0 to 65535
BINARY(size)	Equal to CHAR(), but stores binary byte strings. The *size*parameter specifies the column length in bytes. Default is 1
VARBINARY(size)	Equal to VARCHAR(), but stores binary byte strings. The *size*parameter specifies the maximum column length in bytes.
TINYBLOB	For BLOBs (Binary Large OBjects). Max length: 255 bytes
TINYTEXT	Holds a string with a maximum length of 255 characters
TEXT(size)	Holds a string with a maximum length of 65,535 bytes
BLOB(size)	For BLOBs (Binary Large OBjects). Holds up to 65,535 bytes of data

MEDIUMTEXT	Holds a string with a maximum length of 16,777,215 characters
MEDIUMBLOB	For BLOBs (Binary Large OBjects). Holds up to 16,777,215 bytes of data
LONGTEXT	Holds a string with a maximum length of 4,294,967,295 characters
LONGBLOB	For BLOBs (Binary Large OBjects). Holds up to 4,294,967,295 bytes of data
ENUM(val1, val2, val3, ...)	A string object that can have only one value, chosen from a list of possible values. You can list up to 65535 values in an ENUM list. If a value is inserted that is not in the list, a blank value will be inserted. The values are sorted in the order you enter them
SET(val1, val2, val 3, ...)	A string object that can have 0 or more values, chosen from a list of possible values. You can list up to 64 values in a SET list

C.2 Tipe Data Numerik

Data Numerik	Penjelasan
BIT(*size*)	A bit-value type. The number of bits per value is specified in *size*. The *size* parameter can hold a value from 1 to 64. The default value for*size*is 1.
TINYINT(*size*)	A very small integer. Signed range is from - 128 to 127. Unsigned range is from 0 to 255. The *size*parameter specifies the maxi mum display width (which is 255)
BOOL	Zero is considered as false, nonzero values are considered as true.
BOOLEAN	Equal to BOOL
SMALLINT(*size*)	A small integer. Signed range is from - 32768 to 32767. Unsigned range is from 0 to 65535. The *size*parameter specifies th e maximum display width (which is 255)
MEDIUMINT(*size*)	A medium integer. Signed range is from -8388608 to 8388607. Unsigned range is from 0 to 16777215. The *size*parameter specifies the maximum display width (which is 255)
INT(*size*)	A medium integer. Signed range is from -2147483648 to 2147483647. Unsigned range is from 0 to 4294967295. The *size*parameter specifies the maximum display width (which is 255)
INTEGER(*size*)	Equal to INT(size)
BIGINT(*size*)	A large integer. Signed range is from - 9223372036854775808 to 9223372036854775807. Unsigned range is from 0 to 184 46744073709551615. The *size* parameter specifies the maximum display width (which is 255)
FLOAT(*size*,*d*)	A floating point number. The total number of digits is specified in *size*. The number of digits after the decimal point is specified in the *d* parameter. This syntax is deprecated in MySQL 8.0.17, and it will be removed in future MySQL versions

FLOAT(*p*)	A floating point number. MySQL uses the *p*value to determine whether to use FLOAT or DOUBLE for the resulting data type. If*p*is from 0 to 24, the data type becomes FLOAT(). If*p*is from 25 to 53, the data type becomes DOUBLE()
DOUBLE(*size*,*d*)	A normal-size floating point number. The total number of digits is specified in *size*. The number of digits after the decimal point is specified in the *d* parameter
DECIMAL(*size*,*d*)	An exact fixed-point number. The total number of digits is specified in *size*. The number of digits after the decimal point is specified in the *d* parameter. The maximum number for*size*is 65. The maximum number for*d*is 30. The default value for*size*is 10. The default value for*d*is 0.
DEC(*size*,*d*)	Equal to DECIMAL(size,d)

C.3 Tipe Data Waktu & Tanggal

Waktu & Tanggal	Penjelasan
DATE	A date. Format: YYYY-MM-DD. The supported range is from '1000-01-01' to '9999-12-31'
DATETIME(*fsp*)	A date and time combination. Format: YYYY-MM-DD hh:mm:ss. The supported range is from '1000-01-01 00:00:00' to '9999-12-31 23:59:59'. Adding DEFAULT and ON UPDATE in the column definition to get automatic initialization and updating to the current date and time
TIMESTAMP(*fsp*)	A timestamp. TIMESTAMP values are stored as the number of seconds since the Unix epoch ('1970-01-01 00:00:00' UTC). Format: YYYY-MM-DD hh:mm:ss. The supported range is from '1970-01-01 00:00:01' UTC to '2038-01-09 03:14:07' UTC. Automatic initialization and updating to the current date and time can be specified using DEFAULT CURRENT_TIMESTAMP and ON UPDATE CURRENT_TIMESTAMP in the column definition
TIME(*fsp*)	A time. Format: hh:mm:ss. The supported range is from '-838:59:59' to '838:59:59'
YEAR	A year in four-digit format. Values allowed in four-digit format: 1901 to 2155, and 0000. MySQL 8.0 does not support year in two-digit format.

III. Guided

1. Buatlah sebuah database dengan nama **db_kampus**
2. Buatlah beberapa tabel dalam database tersebut sesuai dengan kriteria berikut :

a. Tabel ***mahasiswa***

Field	Type Data
nim	Int (8) Primary Key, Not Null
nama_mhs	Char (50)
sex	Enum ('L','P')
alamat	Varchar (50)
kota	Varchar (20)
asal_sma	Char (30)
nohp	Varchar (12)
Login	Char (20)
Pass	Char (20)
Umur	Integer
Kode_prodi	Char (6) foreign key fk0 (kode_prodi) references prodi (kode_prodi)

b. Tabel ***prodi***

Field	Type Data
kode_prodi	Char (6) Primary Key, Not Null
nama_prodi	Char (30)

c. Tabel ***mata_kuliah***

Field	Type Data
mk_id	Char (10) Primary Key, Not Null
nama_mk	Char (50)

jumlah_jam	Float (4,2)
Sks	Integer

*1 SKS = 13,33 jam; 2 SKS = 26,66 jam; 3 SKS = 40 jam; 4 SKS = 53,33 jam

d. Tabel ***ruang***

Field	Type Data
ruang_id	Char (3) Primary Key, Not Null
nama_ruang	Char (20)
Kapasitas	Integer

e. Tabel ***dosen***

Field	Type Data
Nik	Int (11) Auto Increment Primary Key, Not Null
Inisial	Char (3) UNIQUE KEY
nama_dosen	Char (50)
Status	Enum (‘T’,’LB’)
Sex	enum (‘L’,’P’)
Agama	Char (20)
Login	Char (20)
Pass	Char (20)
Alamat	Varchar (50)
Kota	Varchar (20)
Email	Varchar (50)
Nohp	Varchar (12)
Salary	Int

f. Tabel ***mengajar***

Field	Type Data
Id_mengajar	Int Auto Increment Primary Key , Not Null
jam_ke	Integer
Hari	Varchar (10)
mk_id	Char (10) foreign key fk1 (mk_id) references mata_kuliah (mk_id)
Inisial	Char (3) foreign key fk2 (inisial) references dosen (inisial)
kode_prodi	Char (6) foreign key fk3 (kode_prodi) references prodi (kode_prodi)
ruang_id	Char (3)foreign key fk4 (ruang_id) references ruang (ruang)id)

g. Tabel ***nilai***

Field	Type Data
Nim	Int foreign key fk5 (nim) references mahasiswa (nim)
mk_id	Char (10) foreign key fk6 (mk_id) references mata_kuliah (mk_id)
kode_prodi	Char (6) foreign key fk7 (kode_prodi) references prodi(kode_prodi)
Inisial	Char (3) foreign key fk8 (inisial) references dosen (inisial)
Nilai_UTS	Integer
Nilai_UAS	Integer
Nilai_akhir	Integer

IV. Unguided

1. Buatlah database perpustakaan dengan nama db_perpustakaan yang terdiri dari tabel-tabel dibawah.
2. Tentukan tipe data yang sesuai untuk setiap atribut/field/kolom/property.
3. Perhatikan relasi antar tabel (primary key dan foreign key)