***MEMANIPULASI DATA***

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

Tabel dalam database adalah komponen utama yang membentuk database itu sendiri. Baris(row) dan kolom (column) mengindikasikan data (record).Cara untuk mengatur dan memanipulasi data pada database dilakukan dengan bantuan instruksi SQL seperti

*SELECT, INSERT, UPDATE, DELETE.* Instruksi untuk membuat, mengubah, danmenghapus table dilakukan melalui DDL yaitu dengan *CREATE TABLE, ALTER* *TABLE, DROP TABLE*

Pada contoh dibawah ini adalah sebuah table Mahasiswa UNIROW tuban dengan bentuk sebagai berikut :

Tabel : Mahasiswa

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| NRP | Nama | Jenis kelamin | Alamat | Kota | Propinsi | Telpon |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |

Setiap kolom mempunyai nilai yang terkait dengan tipe data tertentu.

Himpunan yang berupa nilai kolom yang valid atau dapat diterima (acceptable) disebut sebagai *DOMAIN.*

Nilai kolom diluar domain tersebut dianggap sebagai pelangggaran atas integritas data. Usaha untuk mengelompokkkan nilai dalam himpunan yang berlaku disebut dengan *CONSTRAINT* atau batasan nilai.

**2. Tipe Data**

Adapun tipe yang banyak digunakan pada Ms.SQL Server adalah :

*char(n)* : Mendefinisikan string sepanjang n karakter. Bila n tidak didefinisikan makapanjang karakter adalah 1.

*varchar(n)* : *Mendefinisikan string sepanjang variabel n.*

*binnary(n)* : Untuk menyimpan bit pattern seperti heksadecimal. Contoh : 0x0fa9008e

*datetime* : Mendefinisikan tanggal, menyimpan tahun, bulan, hari, jam, menit, detik danseperseribu detik (milliseconds). Nilai tanggal sampai dengan 31 desember 9999.

*text* : Menyimpan teks sampai dengan 2 GB. Text disebut juga dengan binary largeobjects (BLOBs)

*image* : Mendefinisikan binary data untuk menyimpan image seperti GIF, JPG, TIFF, dll.

*money* : Bilangan pecahan dengan 4 angka dibelakang koma. Digunakan untukperhitungan moneter.

*smallint* : Sama dengan int, membutuhkan 50% memory yang ditempati int.

*int* : Mendefinisikan integer, bilangan bulat yang menampung angka sebanyak 4 byte.

*float(n)* : Mendefinisikan angka pecahan (floating point). Nilai n adalah jumlah angkayang dapat ditampung . sysname

*real(n)* : sama dengan float namun menempati memory 50% dari float.

*smalldatetime* : *sama dengan datetime hanya dengan presisi lebih kecil dimana satuan waktu terkecil adalah menit dan nilai tanggal sampai dengan 6 juni 2079.*

*numeric(n,p)* : Mendefinisikan angka pecahan baik fixed desimal ataupun floating point.

Nilai n adalah jumlah bytes total dan p adalah presisi angka dibelakang koma. Numeric analog dengan DECIMAL(n,p).

**3. Unicode Charakter String**

Unicode adalah karakter international yang menampung 16 bit per karakter. Unicode digunakan oleh bahasa non latin, misalnya Jepang, Jerman, Thai, dll.

*nchar* : Unicode karakter maksimum 4.000 karakter.

*nvarchar* : Unicode karakter dengan data variable maksimum 4.000 karakter.

*Sysname* adalah contoh tipe data yang

didefinisikan sebagai synonym dari nvarchar(128) dan digunakan untuk referensi ke nama object database.

*ntext* : Unicode texts dengan panjang dua pangkat tigapuluh (1.073.741.823)karakter.

**4. Mendefinisikan tipe data baru**

*Tipe data dapat dibangun dari tipe data yang sudah ada. Syntax secara umum untuk membuat tipe data baru :*

USE nama\_database

SP\_ADDTYPE nama\_tipe\_data\_baru,tipe\_data\_SQL

*Syntax secara umum untuk menghapus tipe data baru :*

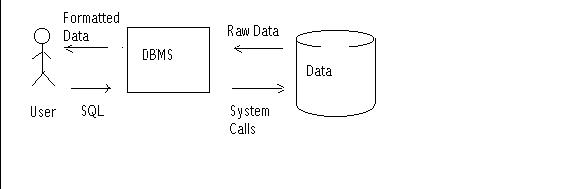
USE nama\_database

SP\_DROPTYPE nama\_tipe\_data\_baru

*Kunci kata NOT NULL atau NULL dapat ditambahkan dalam mendefinisikan tipe data.*

SP\_ADDTYPE alamat, 'varchar(9)', 'NOT NULL'

*Contoh :*



*Alur komunikasi data antara user dengan database menggunakan SQL*

Dalam membuat sebuah tabel dan kolom ada beberapa aturan :

1. Nama tabel dan kolom mempunyai panjang karakter 1-30 dan karakter pertama harus huruf, karakter berikutnya boleh gabungan huruf, garis bawah dan angka dan tidak diperbolehkan mengandung spasi.
2. Tidak mengandung *Reserved word* yang ada pada SQL (contoh:select, where, create, dll).
3. Dalam satu user tidak boleh mempunyai nama tabel yang sama, tetapi dalam tabel yang berbeda boleh memiliki nama kolom yang sama.
4. Tidak bersifat *case sensitive.*

Perintah perintah yang berhubungan dengan tabel :

**INSTRUKSI** **KETERANGAN**

**CREATE TABLE** : Membuat tabel

**SP\_COLUMN** : Memeriksa struktur suatu table

**ALTER TABLE** : Mengubah struktur tabel

**SELECT** : Memilih query tabel

**INSERT** : **UPDATE** :

Menyisipkan baris data ke tabel Mengubah data pada tabel

**DELETE** : Menghapus data pada suatu table

**5. Membuat Tabel**

Syntax:

*CREATE TABLE nama\_tabel(nama\_filed1 tipedata batasan, nama\_filed2 tipedata batasan,……..)*

*Batasan (constraint) = Yang mengikat atribut apakah sebagai primary Key, Foreign Key, Unique, Not Null, dll.*

**6. Melihat Struktur Tabel**

Syntax:

*sp\_columns Nama tabel;*

**7. Memodifikasi dan menambah Field pada Tabel**

Untuk memodifikasi sebuah field pada tabel , syntax:

*ALTER TABLE Nama\_tabel* ALTER COLUMN *Nama\_Field tipe\_data(panjang) Constraint*

Untuk menambah sebuah field pada tabel , syntax:

*ALTER TABLE Nama\_tabel ADD Nama\_Field\_baru tipe\_data(panjang) constraint*

**8. Menghapus Tabel**

Syntax:

*DROP TABLE Nama\_tabel;*

**9. INTEGRITAS DATA**

*Integritas data adalah konsistensi dan kebenaran data yang disimpan. Antara*

*kehidupan nyata dan nilai dari data harus dilindungi oleh system, agar mempunyai arti yang sesuai.*

*Integritas data tetap dijaga melalui beberapa cara, antara lain :*

* *Validasi field secara individual*
* *Verifikasi satu field melalui field yang lainnya*
* *Validasi data dari satu table ke table yang lainnya.*
* *Verifikasi bahwa transaksi berjalan secara sukses dari awal hingga akhir*

***Beberapa jenis integritas data***

***Integritas Entitas***

*Dalam tabel setiap baris harus mempunyai identitas yang unik yang disebut dengan primary key. Hubungan antara primary key dan foreign key menyatakan apakah sebuah baris table dapat diubah atau di hapus.*

*Batasan :*

* *PRIMARY KEY*
* *UNIQUE KEY*
* *UNIQUE INDEX*
* *IDENTITY*

***Integritas Domain***

*Menyatakan bahwa nilai data sesuai dengan kenyataannya. System dalam hal ini melakukan CHECK sesuai dengan aturan (rule)yang berlaku.*

*Batasan dan implementasi :*

* *DEFAULT*
* *CHECK*
* *FOREIGN KEY*
* *Tipe data, default, rules*

***Integritas Referensial***

*Menyatakan relasi antar table bahwa hubungan foreign key dengan primary key. Sebuah foreign key tidak dapat dimasukkan kedalam sebuah table bila primary key tidak atau belum ada. Batasan dan implementasi :*

* *CHECK*
* *FOREIGN KEY*
* *TRIGGERS DAN STORE PROCEDURE*

***Integritas melalui aturan yang didefinisikan sendiri***

*Sesuai dengan aturan logika dan bisnis yang ada maka pemakai dapat menetapkan batasan dan aturan dari data yang disimpan.*

*Batasan dan implementasi :*

* *Batasan melalui table dan kolom pada saat pembuatan table.*
* *FOREIGN KEY*
* *TRIGGERS DAN STORE PROCEDURE*

*Sekarang akan dibahas constraint yang ada dalam Ms.SQL Server beserta contoh penggunaannya.*

**10. PRIMARY KEY**

*Primary Key disebut sebagai constraint untuk menjaga integritas data, yaitu sebuah primary key tidak boleh mempunyai duplikat dan otomatis tidak boleh kosong. Secara umum syntax untuk membuat sebuah primary key adalah:*

*Atau jika primary key terdiri atas banyak kolom (multiple column), maka constraint dapat dibuat sebagai berikut:*

CREATE TABLE Nama\_Tabel (Nama\_Field Tipedata CONSTRAINT pk\_Namatabel PRIMARY KEY, … )

**11. NOT NULL**

*Jika sebuat kolom pada tabel tidak boleh kosong maka constraint* NOT NULL *harus diberikan.*

CREATE TABLE NamaTabel(NamaField1 TipeData1, NamaField2 TipeData2, NamaField3 TipeData3, NamaField4 TipeData4, NamaField5 TipeData5, NamaField6 TipeData6, NamaField7 TipeData7, CONSTRAINT pk\_NamaTabel PRIMARY KEY (NamaField3, NamaField2, NamaField7) )

**12. UNIQUE**

*UNIQUE digunakan untuk menjamin bahwa nilai kolom adalah tunggal tidak mempunyai duplikat. Primary key dan unique keduanya adalah kunci kata yang mempunyai arti sama.*

*Ms.SQL Server mengijinkan field dengan constraint unique untuk mempunyai nilai* NULL*. Bila satu kolom mempunyai nilai* NULL*, maka nilai* NULL *tidak boleh muncul lagi. Primary key sebaliknya tidak boleh kosong karena primary key secara implicit adalah* NOT NULL*.*

**13. CHECK**

*CHECK digunakan untuk menjamin bahwa nilai kolom dalah dalam ruang nilai tertentu. Misalnya nilai mata kuliah dibatasi tidak boleh melebihi nilai 100.*

*Operator logika seperti* AND, OR*, dan* NOT *dapat disertakan dalam pembuatan batasan ini.*

**14. FOREIGN KEY**

*FOREIGN KEY adalah kolom pada sebuah tabel yang menunjukkan bahwa kolom tersebut adalah primary key pada tabel lain*.

*Tabel Mahasiswa*

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **NPM** | **Nama** | | **JK** | **Alamat** |  | | | **Kota** | | | **Propinsi** | | **Telpon** | |
|  |  | |  |  |  | | |  | | |  | |  | |
| 02113216 | Bambang | |  | kebonsari |  | | | tuban | | | Jatim | | 03563221222 | |
|  |  | |  |  |  | | |  | | |  | |  | |
| *Tabel Mata\_kuliah* | |  | | | |  |  | |  |  | |  | |
| **Kode\_mata\_kuliah** | | **Nama\_Kuliah** | | | |  | **SKS** | |  |  | |  | |
|  | |  | | | |  |  | |  |  | |  | |
| TPL144 | | Sistem Database | | | |  | **3** | |  |  | |  | |
|  | |  | | | |  |  | |  |  | |  | |
| *Tabel ambil\_kuliah* | |  | | | |  |  | |  |  | |  | |
| **Kode\_mata\_kuliah** | | **NPM** | | | | **Nilai\_Uts** | | |  | **Nilai\_Uas** | | **Grade** | |
|  | |  | | | |  |  | |  |  | |  | |
| TPL144 | | 14120406003 | | | | **3** |  | |  | 90 | | A | |
|  | |  | | | |  |  | |  |  | |  | |

*Pada sekenario di atas, record (*TPL144*) pada tabel ambil\_kuliah tidak dapat dihapus karena referensi dari* TPL144 *(filed kode\_mata\_kuliah ditabel mata\_kuliah),* 14120406003 *( field npm di tabelmahasiswa) masih ada.*

*Penulisan constraint foreignkey adalah :*

CONSTRAINT fk\_nama\_constraint FOREIGN KEY field REFERENCES nama\_tabel(field)

*Foreign key harus menunjuk ke primary key atau unique pada tabel lain*.

**15. DEFAULT**

*Nilai default sebuah kolom dapat disuplai oleh system, jika pemakai tidak memberikan nilai pada field tersebut. DEFAULT tidak berlaku untuk kolom dengan tipe data timestamp atau field yang mempunyai property* IDENTITY*.*

**16. IDENTITY**

*IDENTITY merupakan property sebuah field, yaitu system menciptakan nomor urut secara otomatis pada setiap memasukkan record baru.*

*Penggunaan:*

NamaKolom Tipe IDENTITY(No\_awal, incremen)

*Setiap terjadi INSERT, maka kolom akan mendapatkan nomor urut, yang dimulai dari No\_awal, dan setiap record berikutnya ditambahkan nilai inkremen (default adalah 1).*

CREATE TABLE pendaftar( no\_daftar int IDENTITY(100,2), nama char(20))

*Pada awalnya no\_daftar adalah 100. Untuk setiap record baru berikutnya, no\_daftar akan bernilai 102, 104, 106, dan seterusnya.*

**17. Drop Constraint**

*DROP digunakan untuk menghapus constraint yang ada. Contoh:*

ALTER TABLE nama\_tabel DROP fk\_nama\_constraint

**18. Mengisi data pada Tabel**

Syntax:

*INSERT INTO Nama\_tabel (Nama\_Field1, Nama\_Field2, Nama\_Field3)VALUES (data1, data2,data3)*

**19. Meng-copy data dari Tabel lain**

Suatu kolom pada sebuah tabel dapat diisi dengan data yang sama dengan kolom tabel lain dengan syarat type data dan panjang field kedua table tersebut

sama. Syntax:

*INSERT INTO Nama\_tabel\_tujuan (Field1, Field2, Field3) SELECT*

*field\_sumber1, field\_sumber2, field\_sumber3) FROM Nama\_tabel\_sumber*

Jika 2 buah tabel mempunyai struktur yang sama maka syntax di atas dapat dipersingkat menjadi :

INSERT INTO Tabel\_tujuan SELECT \* FROM Tabel\_asal

UPDATE Nama\_tabel

SET Field1=Data1, Field2=Data2, Field3=Data3

WHERE kondisi1, kondisi2

Contoh :

Merubah data mahasiswa dengan Npm ‘141203010026’ pada Mahasiswa\_XXX dengan data baru

UPDATE Mahasiswa\_XXX

SET Alamat= ‘Junrejo ’, kota=’Tuban’

WHERE npm= ’03563221422’

Pemanfaatan operasi aritmatika untuk merubah data. Misalnya Sebuah perusahaaan akan menaikkan gaji pegawainya sampai 20 %.

*UPDATE Data\_Gaji SET gaji= gaji\*1.2*

**2. Menghapus data pada Tabel**

Syntax:

DELETE FROM Nama\_tabel WHERE kondisi1

Dalam menghapus dan update bisa dilakukan untuk menghapus banyak b atau menghapus semua isi dari tabel tergantung dari kondisi yang diberikan.

Contoh :

Hapus Isi semua record pada Data mahasiswa yang telah dibuat dimana Mahasiswa dengan alamat yang tidak diisi<nilai=default> akan dihapus.

Perintah SQL nya adalah :

DELETE FROM Mahasiswa\_XXX WHERE Alamat= ’Tuban’

**3. Mencari Data / Retrieval (SELECT)**

*Select adalah instruksi yang paling banyak digunakan dalam SQL. SELECT memilih spesifik kolom dari satu atau beberapa tabel.*

*Secara umum bentuk SELECT adalah:*

*SELECT Kolom1, kolom2, … FROM NamaTabel WHERE predikat*

*Bila kunci kata “WHERE” tidak diberikan, maka yang diseleksi adalah seluruh record dalam tabel. Predikat menyatakan kualifikasi dari record yang ditemukan,*

*bila memenuhi syarat, maka record tersebut akan dipilih.*

*Select \* adalah karakter khusus (wild character) yang menyatakan bahwa kolom yang dipilih adalah seluruh kolom yang ada pada tabel tersebut.*

*SELECT \* FROM mata\_kuliah\_XXX*

*mempunyai arti sama dengan:*

*SELECT Kode\_mata\_kuliah, Nama\_Kuliah, Sks FROM Mata\_Kuliah\_XXX*

*Contoh berikut menampilkan Nama\_kuliah dan Sks:*

SELECT Nama\_kuliah, Sks FROM Mata\_kuliah\_XXX

**4. Predikat dengan WHERE**

*Dengan menggunakan WHERE, maka seleksi dilakukan tidak pada seluruh record, melainkan hanya pada record yang memenuhi syarat.*

*Bentuk klausa WHERE adalah:*

*WHERE Kolom <operator> <nilai>*

*Beberapa operator yang berlaku yaitu:*

*= sama dengan*

*<> tidak sama dengan < lebih kecil*

*<= lebih kecil atau sama > lebih besar*

*>= lebih besar atau sama*

*Contoh :*

*Tampilkan data kuliah yang jumlah sks nya dibawah 3*

SELECT \* FROM Mata\_kuliah WHERE sks < 3

**5. AND, OR, dan NOT**

*Untuk menggabungkan lebih dari satu kondisi, diperlukan logical AND dan OR. Sedangkan untuk negasi, diperlukan NOT*

*Contoh :*

*Tampilkan Mahasiswa perempuan yang berasal dari Tuban*

SELECT Nama, Alamat, Kota FROM Mahasiswa\_XXX WHERE kota=‘Tuban’ AND jk=‘P’

*Prioritas AND lebih kuat dari OR, sehingga kondisi OR harus dipisahkan dalam kurung sebelum AND.*

**6. BETWEEN dan NOT BETWEEN**

*BETWEEN menyederhanakan pencarian “antara” (range) seperti berikut:*

*Cari Nama\_kuliah yang mempunyai sks antara 1 sampai 3. Dengan menggunakan BETWEEN maka bentuk query adalah :*

*SELECT Kolom1, Kolom2 FROM tabel*

*WHERE kolom BETWEEN nilai1 AND nilai2*

*Contoh :*

*Cari Nama\_kuliah yang jumlah Sks nya tidak diantara 1 sampai 3 jam*

SELECT nama\_kuliah FROM Mata\_kuliah\_XXX

WHERE SKS NOT BETWEEN 1 AND 3

**7. LIKE & NOT LIKE**

*Umumnya LIKE digunakan untuk permintaan yang mencari suatu teks berdasarkan prefix (kata depan), sufix (kata akhir) atau kata tengah.*

*SELECT Kolom1, Kolom2, .. .. FROM NamaTabel WHERE Kolom LIKE nilai*

*Contoh :*

*Cari Mahasiswa yang bertempat tinggal di kota “Jakarta”*

SELECT nama, alamat, kota FROM Mahasiswa\_XXX

WHERE kota LIKE ‘Malang’

*Cari Mahasiswa yang tempat tinggalnya dimulai dengan ‘B’*

SELECT nama, alamat, kota FROM Mahasiswa\_XXX

WHERE kota LIKE ‘B%’

*Perhatikan bahwa tanda % menyerupai “wild character” yang bisa di variasi dalam menggunakannya. tanda % adalah terdiri dari 0,1 atau beberapa karakter, maka tanda \_ (under score/garis bawah) merepresentasikan persis satu karakter.*

*NOT LIKE digunakan sebagai negasi misalnya dengan pertanyaan*

*Contoh :*

*Cari Mahasiswa yang alamat rumahnya tidak mengandung kata “Jl.”*

SELECT nama, alamat, kota FROM Mahasiswa\_XXX

WHERE alamat NOT LIKE ‘%Jl%’

**8. IN & NOT IN**

*IN digunakan untuk mencari nilai dalam satu kelompok nilai (daftar/list). Nilai dalam daftar terdiri atas satu atau lebih.*

*SELECT Kolom1,Kolom2,… FROM namatabel WHERE Kolom IN*

*Contoh :*

*Cari mahasiswa yang bertempat tinggal di kota “Tuban” dan “banjarmasin”*

SELECT nama, alamat, kota FROM Mahasiswa\_XXX

WHERE kota IN ('Tuban','Bamjarmasin')

*Bila di negatifkan pertanyaan tersebut, maka hasilnya adalah :*

SELECT nama, alamat, kota FROM Mahasiswa\_XXX

WHERE kota NOT IN ('Tuban','Banjarmasin')

**9. IS NULL & IS NOT NULL**

*NULL adalah nilai yang belum diisi. NULL tidak identik dengan spasi(blank).*

**10. Tampilan Tersusun (Sortir)**

*Tampilan dari hasil SELECT dapat di susun (sort) menurut satu atau beberapa kolom.*

*SELECT Kolom1,Kolom2,… FROM namatabel WHERE predikat ORDER BY Kolom*

*Bila kunci kata where tidak diberikan, maka seluruh record dalam tabel akan ditampilkan. Contoh : Tampilkan seluruh mahasiswa urut berdasarkan alamat*

SELECT \* FROM Mahasiswa\_XXX ORDER BY alamat

*Susunan dapat diatur dari A ke Z (ASCending)atau dari Z ke A (DESCending). Defaultnya adalah ASC.*

*Untuk menyususn secara terbalik adalah :*

SELECT \* FROM Mahasiswa\_XXX ORDER BY inisial DESC

**11. DISTINCT**

*DISTINCT menghilangkan nilai ganda, sehinggga membuatnya menjadi tunggal. Hal ini dilakukan dengan mengeliminasi nilai yang sama.*

*Contoh : Berasal dari kota mana saja mahasiswa yang ada ?*

SELECT kota FROM mahasiswa\_XXX

*untuk menghilangkan duplikasi pada tampilan, maka digunakan kunci kata DISTINCT*

SELECT DISTINCT kota FROM mahasiswa\_XXX

**TUGAS PRAKTIKUM**

**TUGAS 1**

*1. Buat tabel berikut:*

*Mahasiswa\_XXX(Npm, nama, jk, alamat, kota, propinsi, telpon) Npm char 8*

*Nama varchar 50 Jk char 1*

*Alamat varchar 150 Kota varchar 50 Propinsi varchar 50 Telpon varchar 20 Perhatikan:*

* *Npm merupakan primary key*
* *Nama, Alamat tidak boleh kosong*
* *Kota bila tidak disisi maka akan otomatis diisi dengan ‘TUBAN ‘*
* *Jk mempunyai nilai ‘L’ atau ‘P’*

2. *Buat tabel Mata\_kuliah\_XXX*

*Mata\_Kuliah (kode\_mata\_kuliah, Nama\_kuliah, Sks) kode\_mata\_kuliah char 8*

*Nama\_kuliah varchar 100 Sks int*

*Perhatikan:*

* *Kode\_mata\_kuliah merupakan primary key*
* *Nama\_kuliah dan Sks tidak boleh kosong*

*3. Buat tabel ambil\_kuliah\_XXX*

*Ambil\_Kuliah (kode\_mata\_kuliah, Npm, Nilai\_Uts, Nilai\_Uts, Grade) kode\_mata\_kuliah char 8*

*Npm char 8 Nilai\_Uts smallint Nilai\_Uas smallint Grade Char 1*

*Perhatikan:*

* *Kode\_mata\_kuliah merupakan Foreign key yang mengacu pada tabel Mata\_Kuliah\_XXX (field kode\_ mata\_kuliah).*
* *Npm merupakan Foreign key yang mengacu pada tabel Mahasiswa\_XXX (field Npm).*

- *Nilai\_Uts, Nilai\_Uas tidak boleh melebihi 100*

1. Modifikasi field Telpon pada tabel Mahasiswa\_XXX sehingga type-nya menjadi char.
2. Isi satu record pada Data mahasiswa yang telah dibuat. Perintah SQL nya adalah :

**INSERT INTO** Mahasiswa\_XXX (NPM, Nama, Jk, Alamat, Kota, Propinsi, Telpon)

**VALUES** (‘02113216’, ’Bambang’, ’L’, ’Jl. Bumi Handayani 4’, ’ Banjarmasin’, ’Kalsel’,

’0356322133’)

5. Tambahkan data di bawah ini:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **NRP** | **NAMA** | **JK** | **ALAMAT** | **KOTA** | **PROPINSI** | **TELFON** |
|  |  |  |  |  |  |  |
| 141203010006 | Rita | P | JL.JAYA NO.20 | TUBAN | JATIM | 0356322133 |
|  | Sugiarti |  |  |  |  |  |
| 141203010007 | Bambang | L | Jl. Bumi | Banjarmasin | Kalsel | 081803803202 |
|  | Widiono |  | Handayani 4 |  |  |  |
|  |  |  |  |  |  |  |
| 14120406003 | Doni | L | Cendrawasih | Banyuwangi | Jatim | 08561234678 |
|  | armandho |  | 27 |  |  |  |
|  |  |  |  |  |  |  |
| 141203010026 | Adhis | P | Asrama | Malang | Jatim | 03417677993 |
|  | Mahaswi |  | PUSDIK |  |  |  |
|  |  |  | ARHANUD |  |  |  |
|  |  |  | no 114 |  |  |  |
|  |  |  |  |  |  |  |

6. Tampilkan data dari tabel Mahasiswa\_XXX Perintah SQL:

*SELECT \* FROM* Mahasiswa*\_XXX;*

**Tugas 2**

1. Buat tabel-tabel dari sebuah perpustakaan dengan tipe data dan aturan-aturannya:

**Tabel Anggota**

|  |  |  |  |
| --- | --- | --- | --- |
| **Nama Field** | **Type** | **Panjang** | **Aturan** |
|  |  |  |  |
| No\_Anggota | Char | **8** | Not Null, Unique, Primary Key |
| Nama | Varchar | 20 | Not Null |
|  |  |  |  |
| Alamat | Varchar | 30 | Can Be Null, Default = |
|  |  |  | ‘Malang’ |
|  |  |  |  |
| Telepon | Varchar | 15 | Can Be Null |
|  |  |  |  |
| **Tabel Buku** |  |  |  |
|  |  |  |  |
| **Nama Field** | **Type** | **Panjang** | **Aturan** |
|  |  |  |  |
| No\_Buku | Char | **8** | Not Null, Unique, Primary Key |
| Judul | Varchar | 20 | Not Null |
|  |  |  |  |
| Pengarang | Varchar | 30 | Can Be Null |
|  |  |  |  |
| Penerbit | Varchar | 20 | Can Be Null |
|  |  |  |  |
| Jumlah | Int |  | Not Null |
|  |  |  |  |

1. Isi data minimal 3 data. (Sesuaikan dengan soal no. 5 dan 6)
2. Jelaskan apa yang dimaksud dengan primary key dan foreign key. Jelaskan hubungan keduanya.
3. Tampilkan semua data dari tabel anggota dan tabel buku.
4. Tampilkan data anggota yang beralamat di Banyuwangi.
5. Tampilkan data anggota yang ada di kota Malang **dan** Banjarmasin.