Modul 3
Data Definition Language

Data Definition Language (DDL) merupakan perintah SQL yang digunakan untuk mendefinisikan dan mendeklarasikan suatu objek database, menciptakan objek database atau menghapus objek database. Objek database tersebut dapat berupa database dan tabel.

DDL dapat juga digunakan untuk membuat koneksi antar tabel dalam database beserta batasannya dengan menentukan kolom indeks sebagai kuncinya. DDL yang biasa digunakan adalah CREATE, DROP dan ALTER.

3.1 Membuat database dan tabel

CREATE digunakan untuk membuat objek database baru. Sebagai contoh adalah untuk membuat database baru, tabel baru, dan lain-lain.

Membuat database baru

```
CREATE DATABASE Nama_database
```

Contoh:

```
CREATE DATABASE PBD
```

Sedangkan untuk membuat basis data menggunakan parameter querynya adalah sebagai berikut.

```
CREATE DATABASE Nama_Database
ON
(
NAME = 'Nama_File_Primer',
FILENAME = 'Lokasi_File_Primer',
SIZE = Ukuran,
MAXSIZE = Ukuran Maksimal,
FILEGROWTH = Pertambahan File
)
LOG ON
(
NAME = 'Nama_File_log',
FILENAME = 'Lokasi_File_Log',
SIZE = Ukuran,
MAXSIZE = Ukuran Maksimal,
FILEGROWTH = Pertambahan File
)
```

Contoh lain dengan tambahan parameter, membuat database Penjualan :

```
CREATE DATABASE PENJUALAN
ON
( NAME = PENJUALAN_dat,
   FILENAME = 'D:\PENJUALANdat.mdf',
   SIZE = 10,
   MAXSIZE = 50,
   FILEGROWTH = 5 )
LOG ON
( NAME = 'PENJUALAN_log',
   FILENAME = 'D:\PENJUALANlog.ldf',
   SIZE = 5MB,
   MAXSIZE = 25MB,
   FILEGROWTH = 5MB )
```

3.2 Mengaktifkan Basis Data

Sebelum membuat tabel maka anda harus mengaktifkan basis data yang akan dibuatkan tabelnya dengan menggunakan perintah **USE.**

Sintaks :

```
USE DatabaseName
```

Contoh :

```
USE Penjualan
```

3.3 Attach dan Detach pada Query Analyzer

Script untuk attach database adalah sebagai berikut :

```
EXEC sp_attach_db @dbname = N'Penjualan',
   @filename1 = N'c:\Program Files\Microsoft SQL Server\MSSQL\Data\penjualan.mdf',
   @filename2 = N'c:\Program Files\Microsoft SQL Server\MSSQL\Data\penjualan_log.ldf'
```

Keterangan :

- @dbname : adalah nama database yang akan di attach
- @filename1 : adalah letak file data yaitu yang berekstensi MDF
- @filename2 : adalah letak file log yaitu yang berekstensi LDF

Sedangakan Script untuk detach database adalah sebagai berikut :

```
EXEC sp_detach_db 'Penjualan', 'true'
```

Keterangan :

- Penjualan adalah nama database yang akan di detach

3.4 Membuat Tabel

Membuat tabel baru

```
CREATE TABLE nama_table(
Nama_field1 tipedata1 [ket1],
Nama_field2 tipedata2 [ket2],
.     .     .     .     .     .
Nama_fieldN tipedataN [ketN])
```

Catatan:

- Sebelum membuat tabel maka anda harus mengaktifkan database yang akan dibuat tabelnya dengan perintah USE.

  Contoh:

  ```
  USE PBD
  ```

- Tanda [ ] berarti bahwa perintah tersebut bersifat optional, artinya boleh diikutsertakan atau boleh juga diabaikan, tergantung dari keperluan. Misalnya, jika kita ingin membuat primary key pada table yang kita buat, maka kita harus menyertakan kata kunci primary key.

Contoh :

```
CREATE TABLE dokter(
kd_dokter varchar(5) primary key not null,
nama varchar(30),
gender char(1) check(gender='L' or gender='P'),
alamat varchar(30),
gaji numeric
)
```

Catatan :

- Null tidak sama dengan nol, tetapi memiliki arti bahwa kolom atau field tersebut tidak ada data yang dimasukkan.
- Not null menyatakan bahwa data pada kolom tersebut tidak boleh kosong dan harus diisi.

Contoh:

```
CREATE TABLE resep(
kd_resep int identity(1,1) primary key not null,
hari   varchar(10)   CHECK   (hari   IN   ('senin','selasa','rabu','kamis',
'jumat','sabtu')),
tanggal datetime,
kd_pasien varchar(5) foreign key references pasien(kd_pasien),
kd_penyakit varchar(5) foreign key references penyakit(kd_penyakit),
kd_obat varchar(5) foreign key references obat(kd_obat),
kd_dokter varchar(5) foreign key references dokter(kd_dokter)
)
```

Identity atau yang biasa disebut dengan istilah *autonumber* adalah nilai yang dihasilkan oleh sistem secara otomatis dan terurut sesuai dengan nilai urutan yang dimasukkan. Contoh di atas kolom kode resep (**`kd_resep int identity(1,1)`**) akan terurut mulai dari angka 1 dengan pertambahan pengurutan 1 nilai.

Bentuk umum dari identity adalah:

```
Nama_kolom tipe_data identity (m,n) [ket]
```

Dengan m adalah nilai awal sedangkan n adalah pertambahan nilainya.

3.5 Menghapus Obyek

DROP merupakan perintah SQL yang digunakan untuk menghapus objek database.

```
DROP Nama_objek
```

Contoh:

```
DROP DATABASE PBD
DROP TABLE dokter
```

3.6 Mengubah Obyek

ALTER digunakan untuk menambah kolom, mengubah kolom atau menghapus kolom dari sebuah tabel. ALTER juga bisa digunakan untuk mengubah sebuah database. Sintaks untuk mengubah tabel menggunakan perintah ALTER adalah:

```
ALTER TABLE nama_table <ACTION>
```

<ACTION> bisa berupa :

1. Manambah field

   `ADD nama_field tipe_data`

2. Menghapus field

   `DROP nama_field`

3. Memodifikasi field

   `ALTER nama_field tipe_data`

4. Menambah constraint

   `ADD CONSTRAINT nama_constraint definisi_constraint`

5. Menghapus constraint

   `DROP CONSTRAINT nama_constraint`

Contoh:

Menambah kolom agama pada tabel dokter dengan type data `varchar(10)`

```
ALTER TABLE dokter ADD agama varchar(10)
```

Jika hanya ada 6 agama yang boleh dimasukkan ke dalam kolom agama, maka pada tabel dokter maka kita bisa memakai perintah :

```
ALTER TABLE dokter
ADD CONSTRAINT cek_agama
CHECK (agama IN ('islam','protestan','katolik','hindu','budha','konghuchu'))
```

Perintah IN mempunyai arti semua data dalam kurung adalah benar. IN merupakan alternatif untuk mengganti perintah OR yang terlalu banyak. Seperti contoh yang sama di bawah ini :

```
ALTER TABLE dokter
ADD CONSTRAINT cek_agama
CHECK (agama ='islam'
OR agama='protestan'
OR agama='katolik'
OR agama='hindu'
OR agama='budha'
OR agama='konghuchu')
```

Menghapus constraint cek_agama

```
ALTER TABLE dokter
DROP CONSTRAINT cek_agama
```

3.7 Integritas Data

Suatu RDBMS mempunyai kemampuan data yang konsisten dengan menerapkan suatu aturan pada saat perancangan database. Aturan ini sering disebut sebagai *constraint* yang dibuat untuk membuat suatu integritas data. Constraint bisa dibuat pada saat pembuatan table atau setelah tabel dibuat. Constraint yang sering digunakan adalah *primary key* dan *foreign key*.

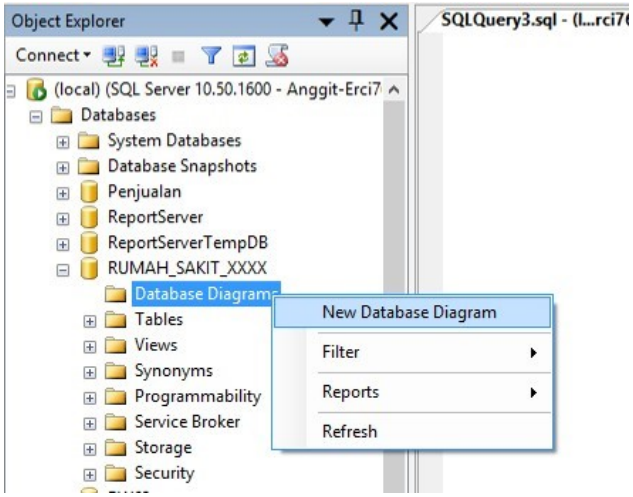
Sebuah tabel mempunyai sebuah kolom atau kombinasi beberapa kolom yang nilainya secara urut mengidentifikasi sebuah baris dalam tabel. Kolom-kolom seperti ini disebut dangan primary key. Sebuah tabel bisa terdiri dari beberapa primary key.

Sedangkan foreign key merupakan kolom atau kombiansi kolom yang dipakai unutk menghubungkan sebuah table dengan table lainnya. Kolom yang didefinisikan sebuah foreign key harus selalu merujuk kepada kolom primary key dari tabel asal baik tipe data maupun isi datanya.

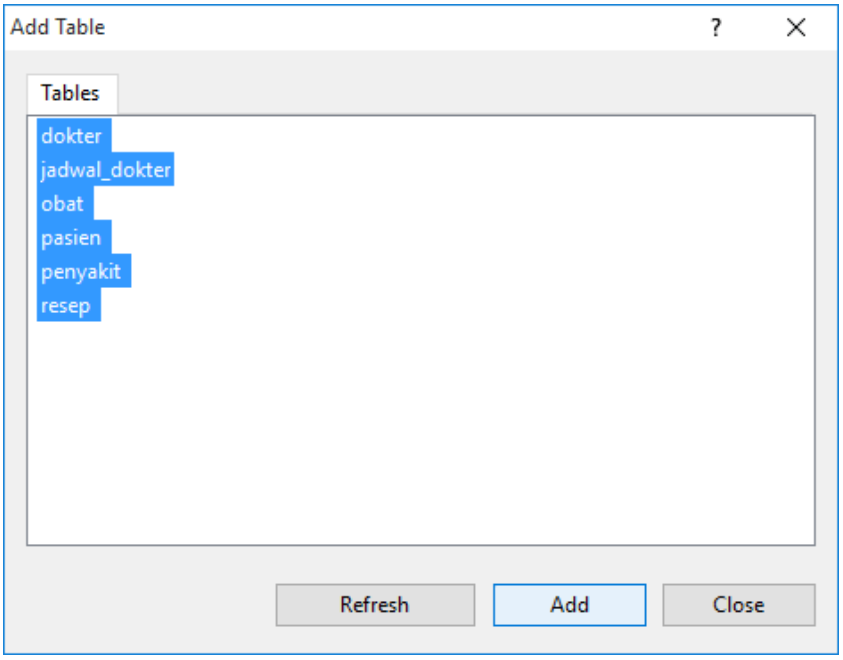
4. Membuat Diagram Relasi

Setelah Semua Tabel Terbuat, Selanjutnya akan dibuat sebuah diagram relasi dari beberapa tabel yang telah dibuat melalui fasilitas yang ad. Fungsi dari diagram relasi ini adalah untuk melihat relasi antar tabel dengan jelas. Adapaun caranya sebagai berikut :

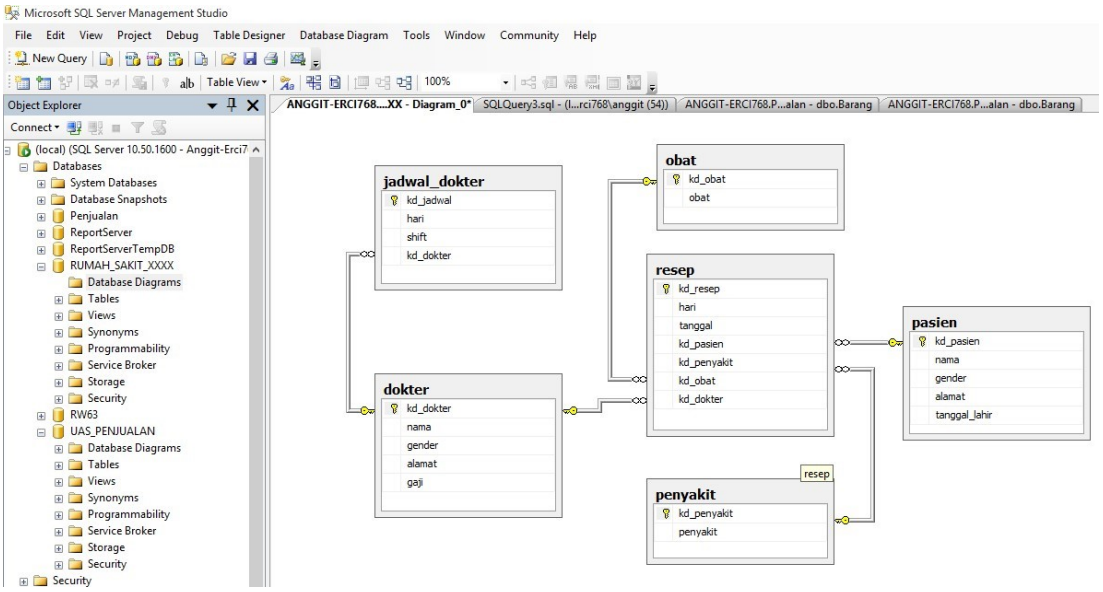
a) Klik Kanan Mouse pada Database Diagram kemudian pilih New Database Diagram



b) Pilih semua tabel dan klik add



b) Tampilan relasi tabel seperti gambar berikut.



2.8 Praktikum

Membuat database dengan ketentuan sebagai berikut:

Tabel dokter

Kolom	Tipe Data	Keterangan
Kd_dokter	Char(5)	Primary key not null
Nama	Varchar(30)	Not null
Alamat	Varchar(30)	
Gender	Char(1)	Check(L/P)
Gaji	Numeric	

Tabel pasien

Kolom	Tipe Data	Keterangan
Kd_pasien	Char(5)	Primary key not null
Nama	Varchar(30)	Not null
Tgl_Lahir	Datetime	
Gender	Char(1)	Check(L/P)
Alamat	Varchar(30)	
Periksa	Numeric	

Tabel penyakit

Kolom	Tipe Data	Keterangan
Kd_penyakit	Char(5)	Primary key not null
Penyakit	Varchar(30)	Not null

Tabel obat

Kolom	Tipe Data	Keterangan
Kd_obat	Char(5)	Primary key not null
Obat	Varchar(30)	Not null

Tabel jadwal_dokter

Kolom	Tipe data	Keterangan
Kd_jadwal	Char(5)	Primary key not null
Hari	Varchar(10)	Check (senin,selasa,rabu,kamis,jumat,sabtu)
Shift	Varchar(10)	Check(pagi/sore)
Kd_dokter	Char(5)	Foreign key (dokter.kd_dokter)

Tabel resep

Kolom	Tipe Data	Keterangan
Kd_resep	Int	Identity (1,1) Primary key not null
Hari	Varchar(10)	Check (senin,selasa,rabu,kamis,jumat,sabtu)
Tanggal	Datetime	
Kd_pasien	Char(5)	Foreign key (pasien.kd_pasien)
Kd_penyakit	Char(5)	Foreign key (penyakit.kd_penyakit)
Kd_obat	Char(5)	Foreign key (obat.kd_obat)
Kd_dokter	Char(5)	Foreign key (dokter.kd_dokter)

```
CREATE TABLE dokter(
kd_dokter char(5) primary key not null,
nama varchar(30) not null,
gender char(1) check(gender='L' or gender='P'),
alamat varchar(30),
gaji numeric
)

CREATE TABLE pasien(
kd_pasien char(5) primary key not null,
nama varchar(30) not null,
tgl_lahir datetime,
gender char(1) check(gender='L' or gender='P'),
alamat varchar(30),
periksa numeric
)

CREATE TABLE penyakit(
kd_penyakit char(5) primary key not null,
penyakit varchar(30) not null
)
```

```
CREATE TABLE obat(
kd_obat char(5) primary key not null,
obat varchar(30) not null
)

CREATE TABLE jadwal_dokter(
kd_jadwal char(5) primary key not null,
hari varchar(10) CHECK (hari IN ('senin', 'selasa', 'rabu', 'kamis',
'jumat','sabtu')),
shift varchar(10) CHECK (shift='pagi' or shift='sore'),
kd_dokter char(5) foreign key references dokter(kd_dokter)
)

CREATE TABLE resep(
kd_resep int identity(1,1) primary key not null,
hari varchar(10) CHECK (hari IN ('senin', 'selasa', 'rabu', 'kamis',
'jumat','sabtu')),
tanggal datetime,
kd_pasien char(5) foreign key references pasien(kd_pasien),
kd_penyakit char(5) foreign key references penyakit(kd_penyakit),
kd_obat char(5) foreign key references obat(kd_obat),
kd_dokter char(5) foreign key references dokter(kd_dokter)
)
```