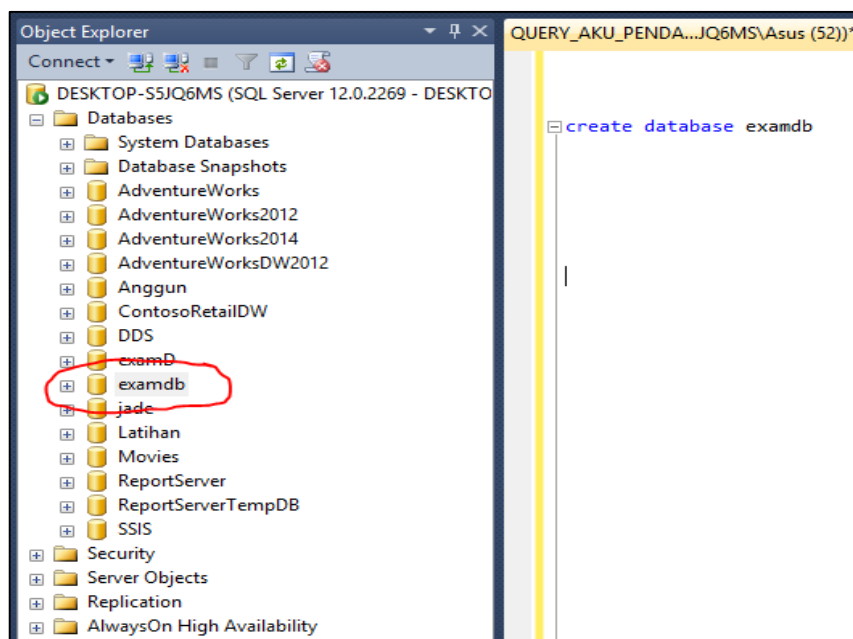


BAB VI HASIL DAN PEMBAHASAN

4.1 Implementasi Tabel pada *Database*

Setelah melakukan normalisasi maka langkah selanjutnya yaitu mengimplementasikannya dalam bentuk tabel pada *SQL Server*, berikut implementasinya:

4.1.1 Membuat *Database*

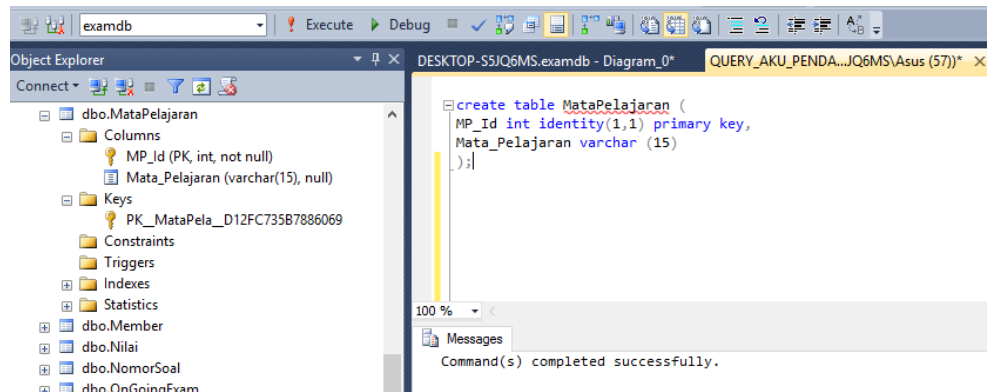


Gambar 4.1 Create Nama Database

Pada Gambar 4.1, di sisi kanan merupakan *query* untuk membuat nama *database*. Di sisi kiri merupakan hasil *execute query database*, yaitu database dengan nama “examdb” yang terdapat lingkaran merah pada gambar. Di *database* ini tabel-tabel akan diimplementasikan.

4.1.2 Membuat Tabel

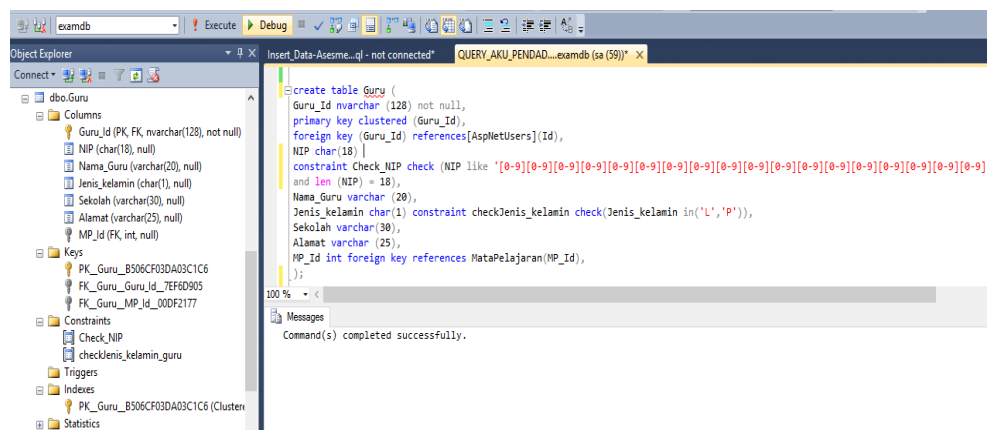
A. Tabel MataPelajaran



Gambar 4. 2 Create Tabel MataPelajaran

Pada Gambar 4.2, di sisi kanan merupakan *query* untuk membuat tabel MataPelajaran, pada sudut kiri atas merupakan nama *database* sebagai tempat tabel tersebut. Di sisi kiri merupakan hasil *execute query database*, yaitu tabel Matapelajaran yang memiliki kolom MP_Id dan Mata_Pelajaran.

B. Tabel Guru

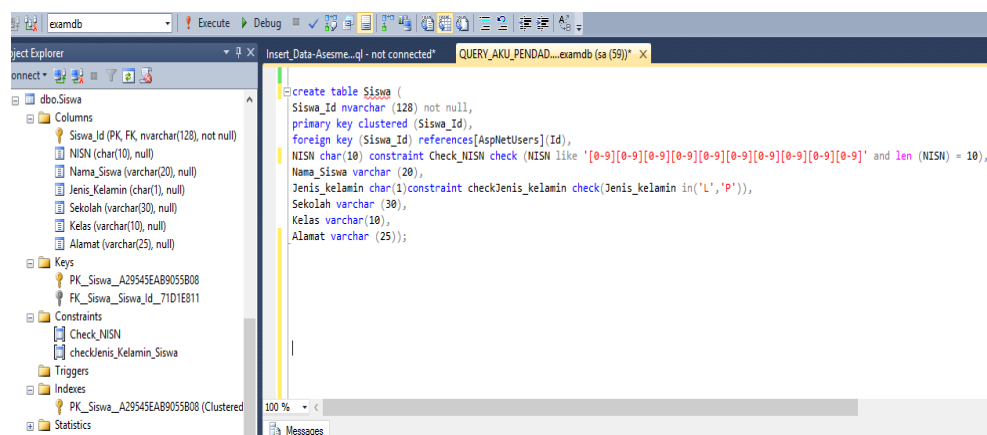


Gambar 4. 3 Create Tabel Guru

Pada Gambar 4.3, di sisi kanan merupakan *query* untuk membuat tabel Guru, pada sudut kiri atas merupakan nama *database* sebagai tempat tabel tersebut. Di sisi kiri merupakan hasil *execute query database*, yaitu tabel Guru yang memiliki kolom Guru_Id, NIP, Nama_Guru, Jenis_Kelamin, Sekolah, Alamat, dan MP_Id. Memiliki *keys* Guru_Id sebagai *primary key* dan *foreign*

key dari tabel ASPNetUsers, dan MP_Id sebagai *foreign key* dari tabel MataPelajaran. Memiliki *check constraint* pada kolom Jenis_Kelamin agar pengisian data pada kolom Jenis_Kelamin hanya singkatan yaitu “P” atau “L”. Dan Memiliki *check constraint* pada kolom NIP dengan pola pengisian hanya bisa diisi angka dengan panjang 18 karakter.

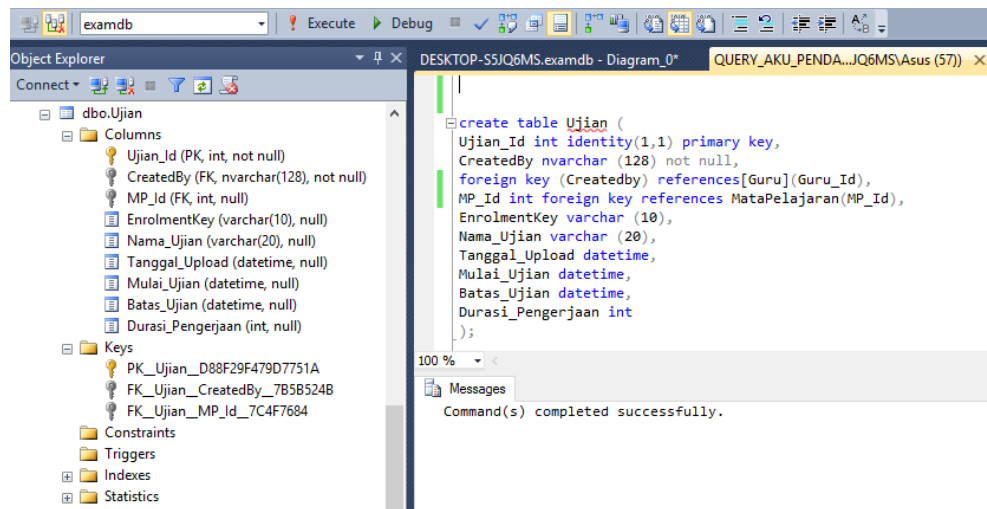
C. Tabel Siswa



Gambar 4. 4 Create Tabel Siswa

Pada Gambar 4.4, di sisi kanan merupakan *query* untuk membuat tabel Siswa, pada sudut kiri atas merupakan nama *database* sebagai tempat tabel tersebut. Di sisi kiri merupakan hasil *execute query database*, yaitu tabel Siswa yang memiliki kolom Siswa_Id, NISN, Nama_Siswa, Jenis_Kelamin, Sekolah, Kelas, dan Alamat. Memiliki *keys* Siswa_Id sebagai *primary key* dan *foreign key* dari tabel ASPNetUsers. Memiliki *check constraint* pada kolom Jenis_Kelamin agar pengisian data pada kolom Jenis_Kelamin hanya singkatan yaitu “P” atau “L”. Dan Memiliki *check constraint* pada kolom NISN dengan pola pengisian hanya bisa diisi angka dengan panjang 10 karakter.

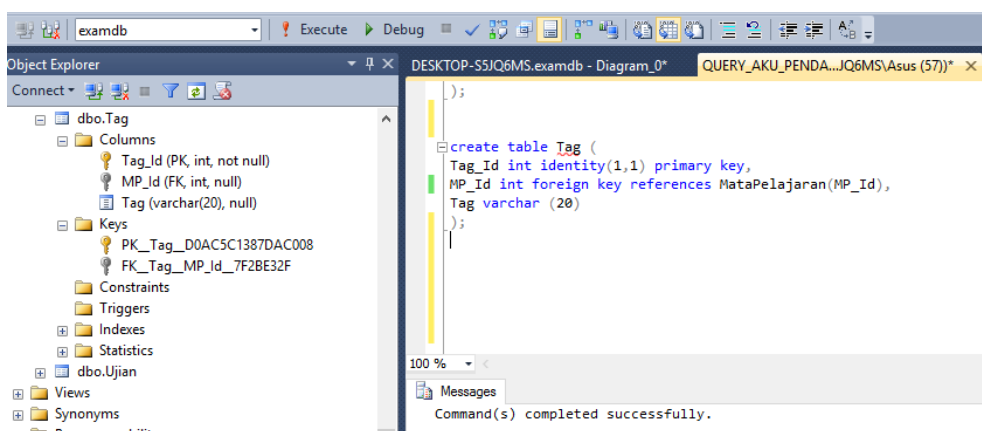
D. Tabel Ujian



Gambar 4. 5 Create Tabel Ujian

Pada Gambar 4.5, di sisi kanan merupakan *query* untuk membuat tabel Ujian, pada sudut kiri atas merupakan nama *database* sebagai tempat tabel tersebut. Di sisi kiri merupakan hasil *execute query database*, yaitu tabel Ujian yang memiliki kolom Ujian_Id, CreatedBy, MP_Id, Nama_Ujian, EnrolmentKey, Tanggal_Upload, Mulai_Ujian, Batas_Ujian, dan Durasi_Pengerjaan. Memiliki *keys* Ujian_Id sebagai *primary key*, CreatedBy (Guru_Id) sebagai *foreign key* dari tabel Guru, dan MP_Id sebagai *foreign key* dari tabel MataPelajaran.

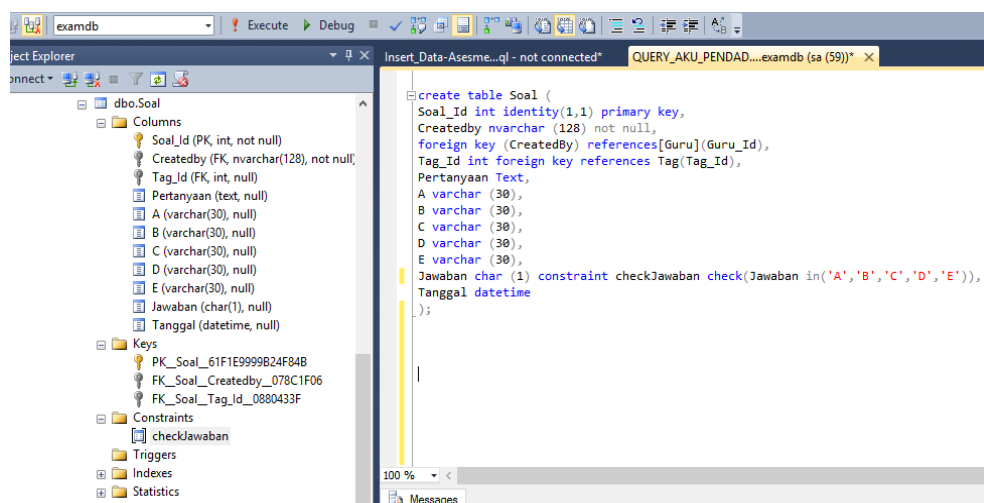
E. Tabel Tag



Gambar 4. 6 Create Tabel Tag

Pada Gambar 4.6, di sisi kanan merupakan *query* untuk membuat tabel Tag, pada sudut kiri atas merupakan nama *database* sebagai tempat tabel tersebut. Di sisi kiri merupakan hasil *execute query database*, yaitu tabel Tag yang memiliki kolom Tag_Id, MP_Id, dan Tag. Memiliki *keys* Tag_Id sebagai *primary key*, dan MP_Id sebagai *foreign key* dari tabel MataPelajaran.

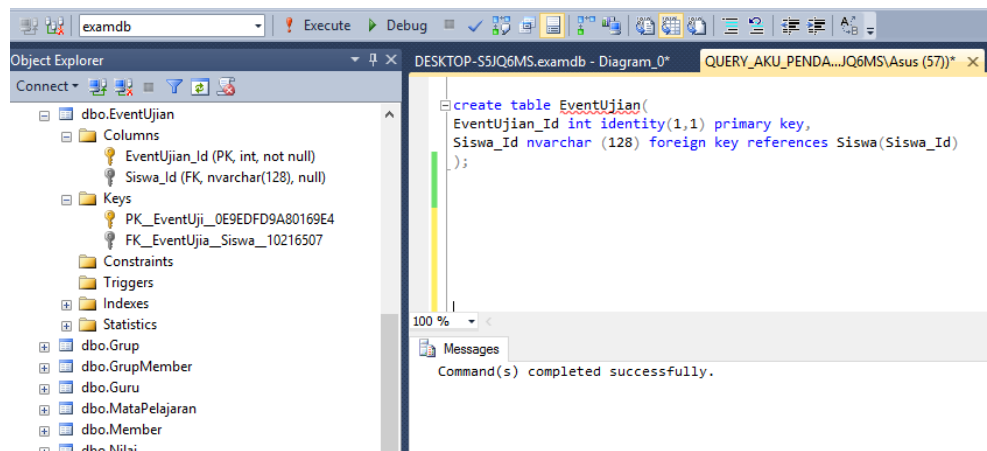
F. Tabel Soal



Gambar 4.7 Create Tabel Soal

Pada Gambar 4.7, di sisi kanan merupakan *query* untuk membuat tabel Soal, pada sudut kiri atas merupakan nama *database* sebagai tempat tabel tersebut. Di sisi kiri merupakan hasil *execute query database*, yaitu tabel Soal yang memiliki kolom Soal_Id, CreatedBy, Tag_Id, Pertanyaan, A, B, C, D, E, Jawaban, dan Tanggal. Memiliki *keys* Soal_Id sebagai *primary key*, CreatedBy (kolom Guru_Id) sebagai *foreign key* dari tabel Guru, dan Tag_Id sebagai *foreign key* dari tabel Tag. Pada kolom Jawaban memiliki *check constrain*, pada kolom tersebut hanya bisa diisi oleh huruf A atau B atau C atau D atau E.

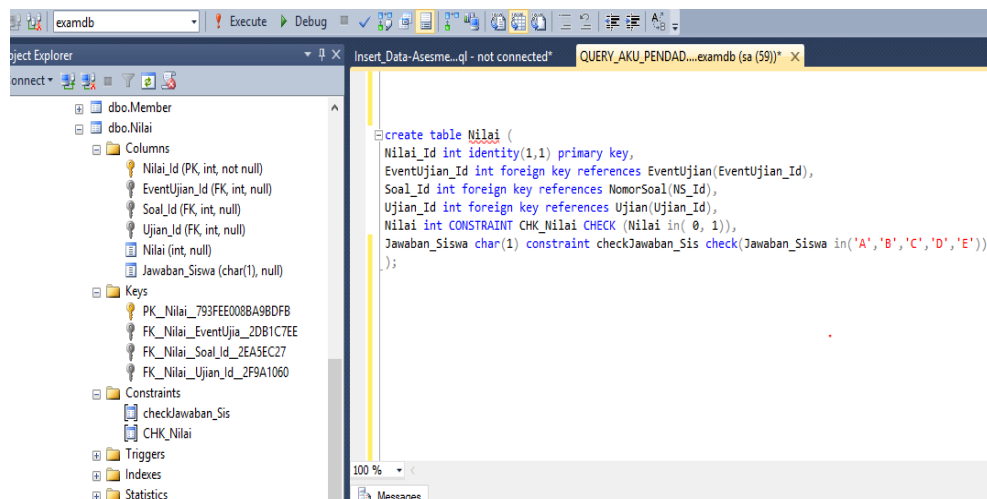
G. Tabel EventUjian



Gambar 4. 8 Create Tabel EventUjian

Pada Gambar 4.8, di sisi kanan merupakan *query* untuk membuat tabel EventUjian, pada sudut kiri atas merupakan nama *database* sebagai tempat tabel tersebut. Di sisi kiri merupakan hasil *execute query database*, yaitu tabel EventUjian yang memiliki kolom EventUjian_Id dan Siswa_Id. Memiliki *keys* EventUjian_Id sebagai *primary key* dan Siswa_Id sebagai *foreign key* dari tabel Guru.

H. Tabel Nilai

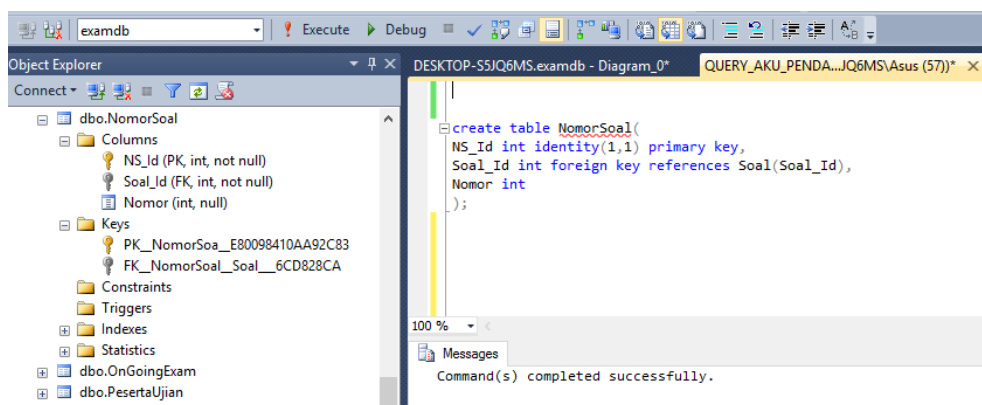


Gambar 4. 9 Create Tabel Nilai

Pada Gambar 4.9, di sisi kanan merupakan *query* untuk membuat tabel Nilai, pada sudut kiri atas merupakan nama *database* sebagai tempat tabel

tersebut. Di sisi kiri merupakan hasil *execute query database*, yaitu tabel Nilai yang memiliki kolom Nilai_Id, EventUjian_Id, Ujian_Id, Jawaban_Siswa, Nilai, dan Soal_Id. Memiliki *keys* Nilai_Id sebagai *primary key*, EventUjian_Id sebagai *foreign key* dari tabel EventUjian, Ujian_Id sebagai *foreign key* dari tabel Ujian dan Soal_Id sebagai *foreign key* dari tabel NomorSoal. Pada kolom Jawaban_Siswa memiliki *check constrain*, pada kolom tersebut hanya bisa diisi oleh huruf A atau B atau C atau D atau E.

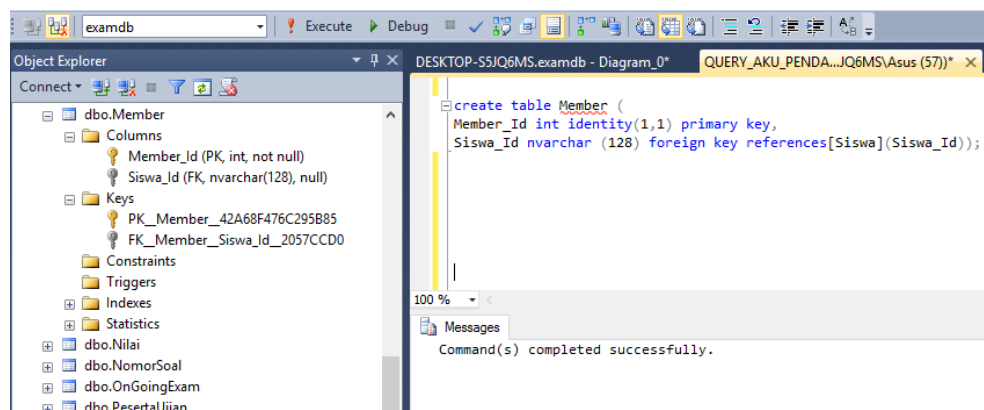
I. Tabel NomorSoal



Gambar 4.10 Create Tabel NomorSoal

Pada Gambar 4.10, di sisi kanan merupakan *query* untuk membuat tabel NomorSoal, pada sudut kiri atas merupakan nama *database* sebagai tempat tabel tersebut. Di sisi kiri merupakan hasil *execute query database*, yaitu tabel NomorSoal yang memiliki kolom NS_Id, Soal_Id, dan Nomor. Memiliki *keys* NS_Id sebagai *primary key*, EventUjian_Id sebagai *foreign key* dari tabel EventUjian, dan Soal_Id sebagai *foreign key* dari tabel Soal.

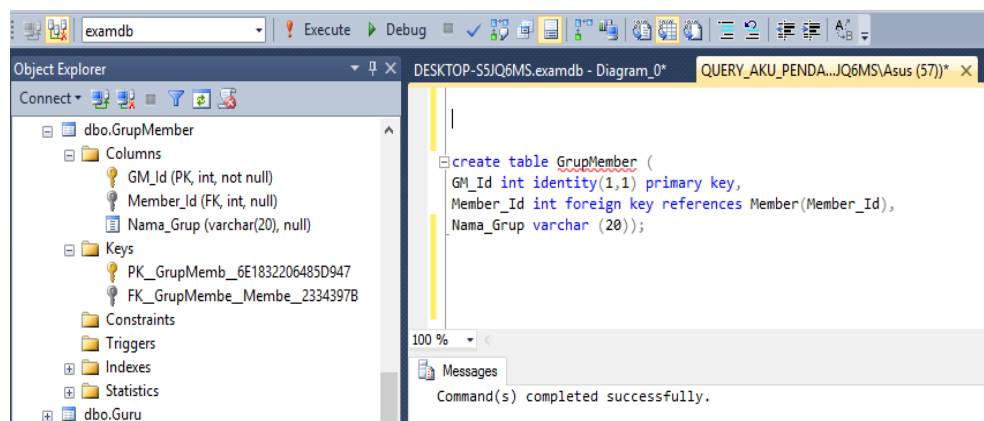
J. Tabel Member



Gambar 4. 11 Create Tabel Member

Pada Gambar 4.11, di sisi kanan merupakan *query* untuk membuat tabel Member, pada sudut kiri atas merupakan nama *database* sebagai tempat tabel tersebut. Di sisi kiri merupakan hasil *execute query database*, yaitu tabel Member yang memiliki kolom Member_Id, dan Siswa_Id. Memiliki *keys* Member_Id sebagai *primary key* dan Siswa_Id sebagai *foreign key* dari tabel EventUjian.

K. Tabel GrupMember

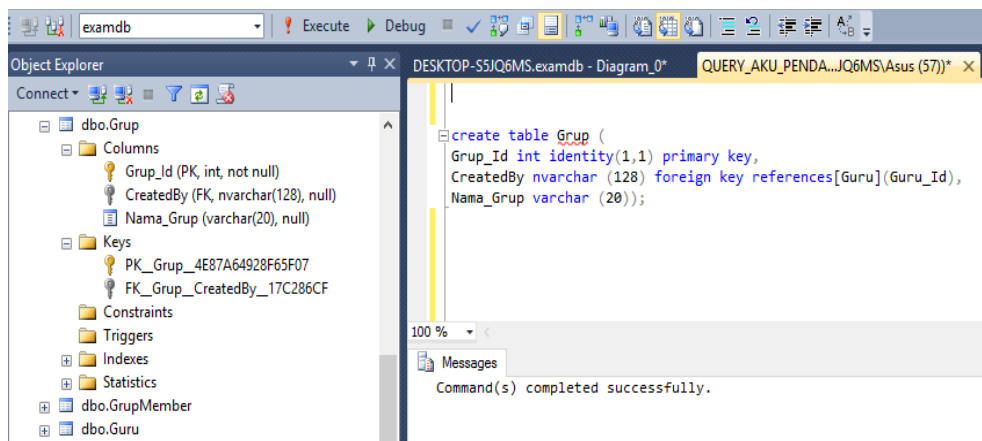


Gambar 4. 12 Create Tabel GrupMember

Pada Gambar 4.12, di sisi kanan merupakan *query* untuk membuat tabel GrupMember, pada sudut kiri atas merupakan nama *database* sebagai tempat tabel tersebut. Di sisi kiri merupakan hasil *execute query database*, yaitu tabel GrupMember yang memiliki kolom GM_Id, Member_Id, dan Nama_Grup.

Memiliki *keys* GM_Id sebagai *primary key*, Member_Id sebagai *foreign key* dari tabel Member.

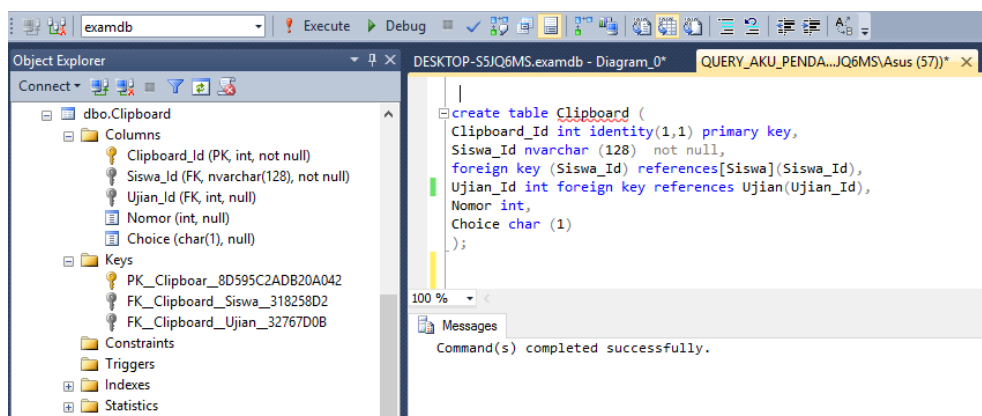
L. Tabel Grup



Gambar 4. 13 Create Tabel Grup

Pada Gambar 4.13, di sisi kanan merupakan *query* untuk membuat tabel Grup, pada sudut kiri atas merupakan nama *database* sebagai tempat tabel tersebut. Di sisi kiri merupakan hasil *execute query database*, yaitu tabel Grup yang memiliki kolom Grup_Id, CreatedBy, dan Nama_Grup. Memiliki *keys* Grup_Id sebagai *primary key* dan CreatedBy sebagai *foreign key* dari tabel Guru.

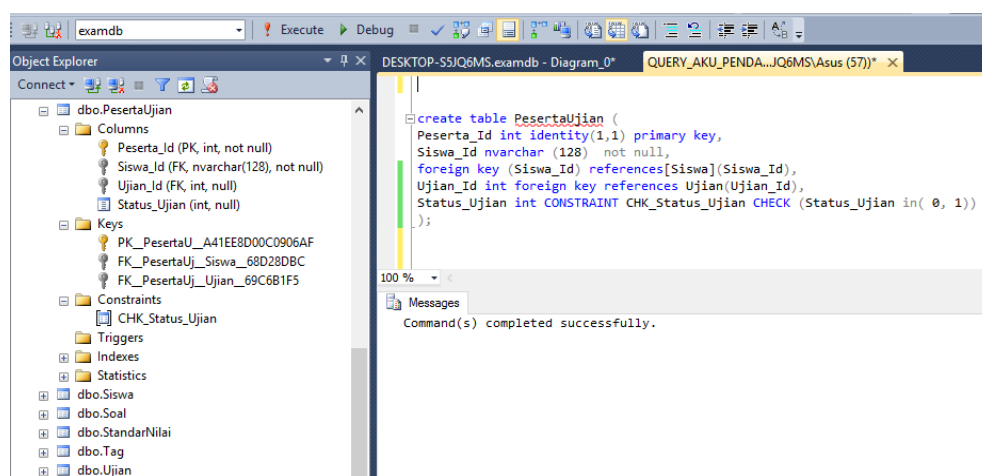
M. Tabel Clipboard



Gambar 4. 14 Create Tabel Clipboard

Pada Gambar 4.14, di sisi kanan merupakan *query* untuk membuat tabel Clipboard, pada sudut kiri atas merupakan nama *database* sebagai tempat tabel tersebut. Di sisi kiri merupakan hasil *execute query database*, yaitu tabel Clipboard yang memiliki kolom Clipboard_Id, Siswa_Id, Nomor, Choice dan Ujian_Id. Memiliki *keys* Clipboard_Id sebagai *primary key*, Siswa_Id sebagai *foreign key* dari tabel Siswa dan Ujian_Id sebagai *foreign key* dari tabel Ujian.

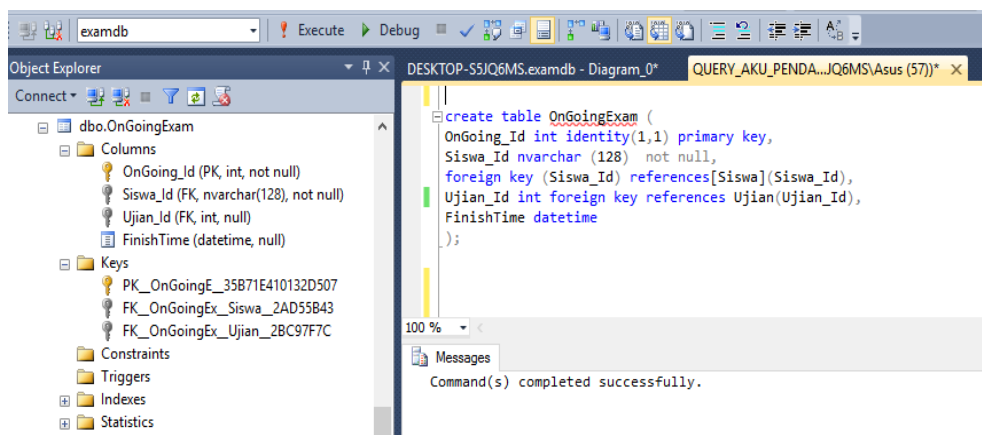
N. Tabel PesertaUjian



Gambar 4. 15 Create Tabel PesertaUjian

Pada Gambar 4.15, di sisi kanan merupakan *query* untuk membuat tabel PesertaUjian, pada sudut kiri atas merupakan nama *database* sebagai tempat tabel tersebut. Di sisi kiri merupakan hasil *execute query database*, yaitu tabel PesertaUjian yang memiliki kolom Peserta_Id, Siswa_Id, Ujian_Id, dan Status_Ujian. Memiliki *keys* Peserta_Id sebagai *primary key*, Siswa_Id sebagai *foreign key* dari tabel Siswa dan Ujian_Id sebagai *foreign key* dari tabel Ujian.

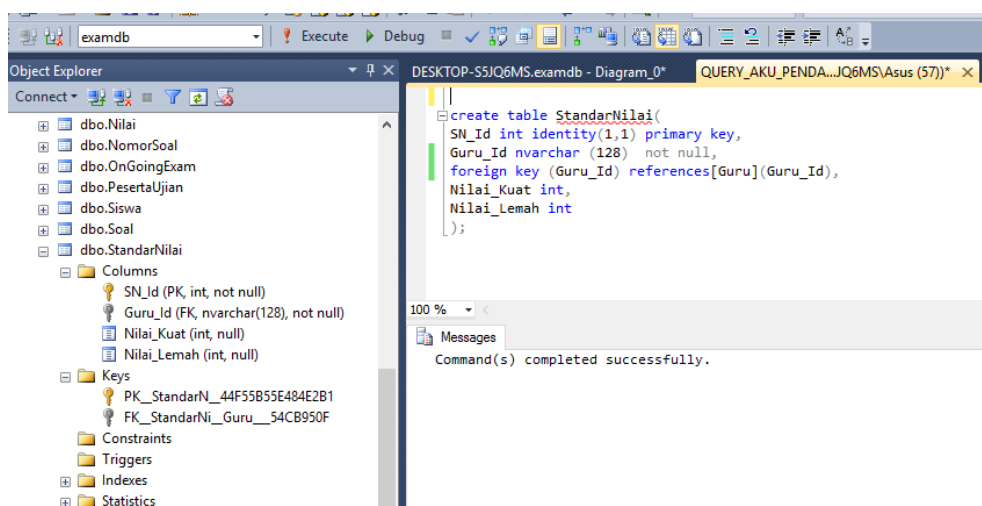
O. Tabel OnGoingExam



Gambar 4.16 Create Tabel OnGoingExam

Pada Gambar 4.16, di sisi kanan merupakan *query* untuk membuat tabel OnGoingExam, pada sudut kiri atas merupakan nama *database* sebagai tempat tabel tersebut. Di sisi kiri merupakan hasil *execute query database*, yaitu tabel OnGoingExam yang memiliki kolom OnGoing_Id, Siswa_Id, Ujian_Id, dan FinishTime. Memiliki *keys* OnGoing_Id sebagai *primary key*, Siswa_Id sebagai *foreign key* dari tabel Siswa dan Ujian_Id sebagai *foreign key* dari tabel Ujian.

P. Tabel StandarNilai



Gambar 4.17 Create Tabel StandarNilai

Pada Gambar 4.17, di sisi kanan merupakan *query* untuk membuat tabel StandarNilai, pada sudut kiri atas merupakan nama *database* sebagai tempat

tabel tersebut. Di sisi kiri merupakan hasil *execute query database*, yaitu tabel StandarNilai yang memiliki kolom SN_Id, Guru_Id, Nilai_Kuat, dan Nilai_Lemah. Memiliki *keys* SN_Id sebagai *primary key* dan Guru_Id sebagai *foreign key* dari tabel Guru.

4.2 Akses Database

Database tidak dapat diakses oleh *user* (admin, guru, dan siswa), database hanya bisa diakses oleh *developer* (pengembang), di sisi lain *user* memiliki otorisasi yang berbeda terhadap sistem sesuai dalam peran yang terdaftar dalam tabel ASPNETRoles yang terdapat dalam database.

4.3 Pengujian

Pengujian perangkat lunak dilakukan untuk memperoleh informasi serta mengevaluasi kualitas dari produk atau layanan yang sedang diuji. Tujuan pengujian dalam pengembangan basis data adalah untuk mengetahui apakah basis data yang diuji dapat memenuhi kebutuhan *admin* dan *user* dengan mendasari pada rancangan dan pengembangan perangkat lunak.

4.3.1 Metode Pengujian

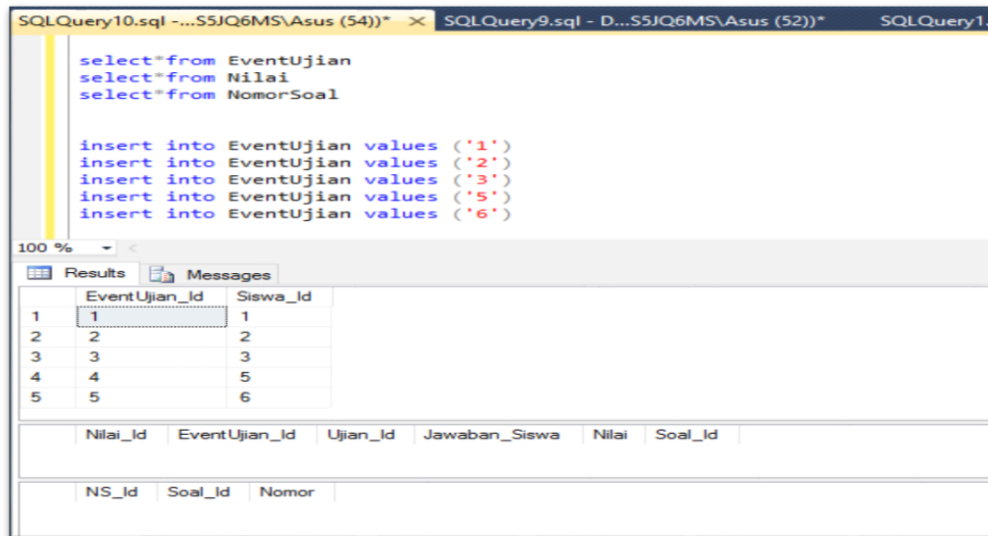
Metode pengujian yang dipakai dalam pengembangan database adalah *anomaly testing*. Pengujian *anomaly testing* dilakukan melalui page admin. *Anomaly testing* berfungsi untuk mengetahui apakah proses basis data yang memberikan efek samping yang tidak diharapkan (misalnya menyebabkan ketidakonsistenan data atau membuat suatu data menjadi hilang ketika data dihapus)

A. Pengujian Anomaly

Pengujian *anomaly* dilakukan terhadap tabel yang telah dinormalisasi, pengujian ini bertujuan untuk menguji apakah tabel tersebut masih memiliki ketergantungan atau ketidakkonsistenan data atau tidak.

a. Tabel Nilai, Tabel EventUjian, dan Tabel NomorUjian

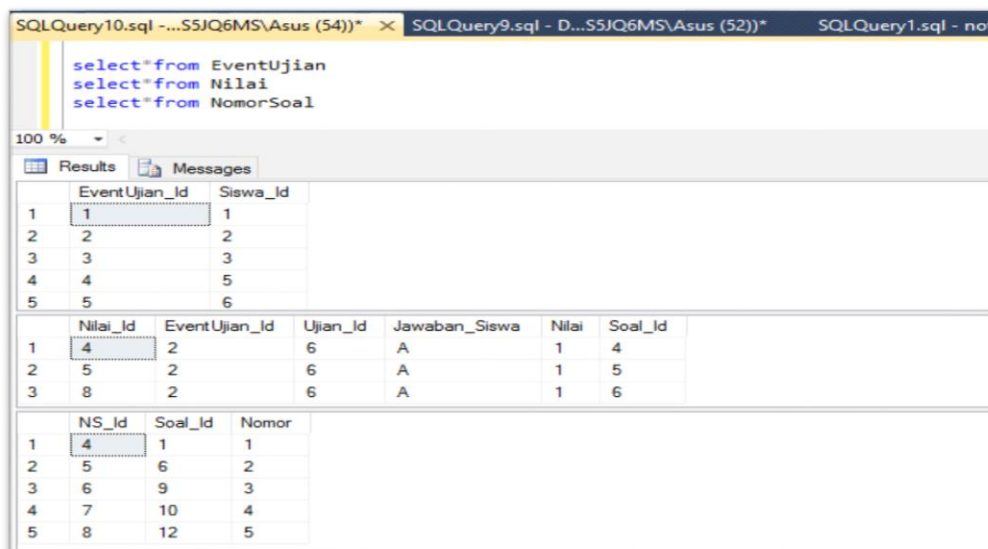
1) *Insert Anomaly*



Gambar 4. 18 Pengujian *Insert Anomaly*

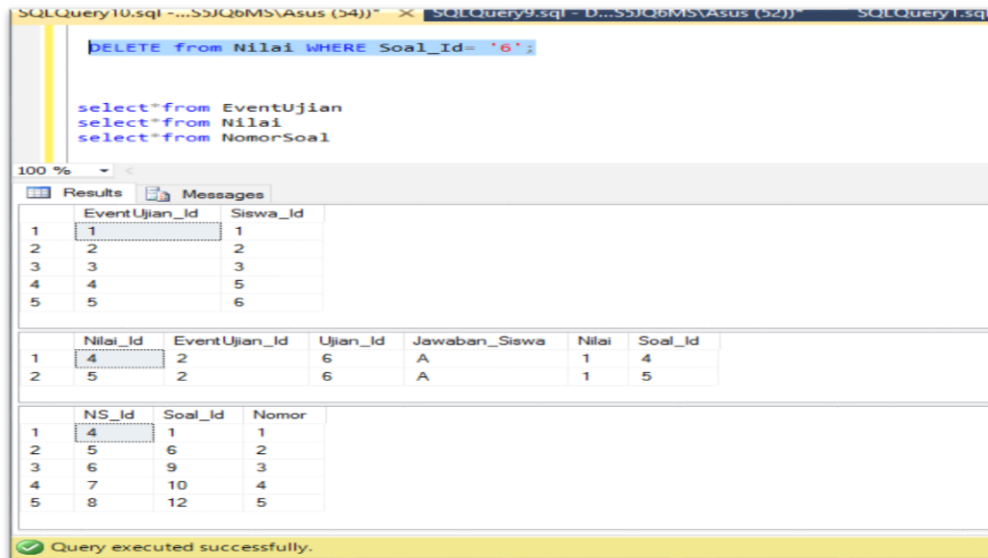
Pada Gambar 4.18, terlihat ketika *insert* data pada tabel EventUjian, maka tabel lain tidak ada perubahan, hanya pada tabel EventUjianr yang mengalami penambahan data, sehingga sudah tidak terdapat *anomaly*.

2) *Delete Anomaly*



Gambar 4. 19 Tabel Sebelum Pengujian

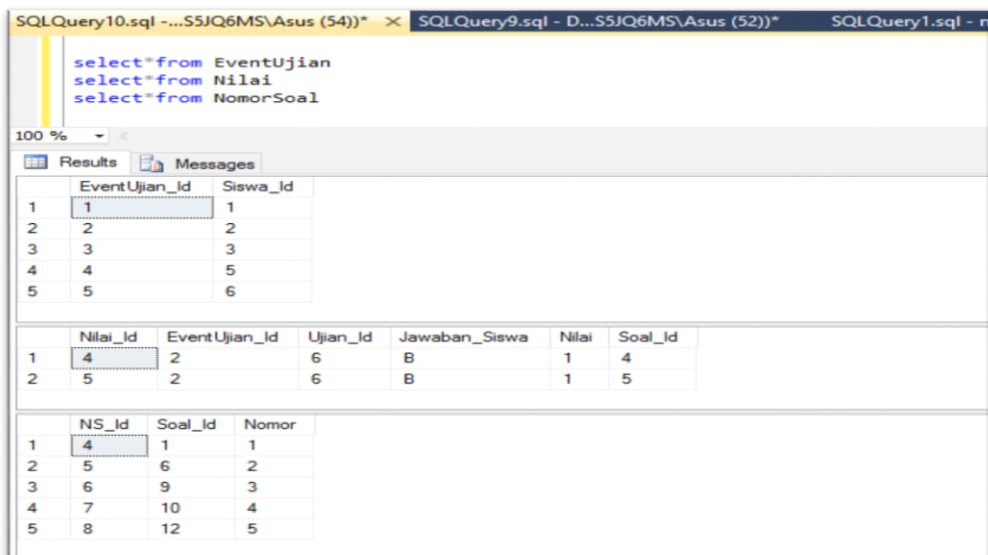
Gambar 4.19 merupakan keadaan awal sebelum dilakukan pengujian *delete anomaly*.



Gambar 4. 20 Pengujian *Delete Anomaly*

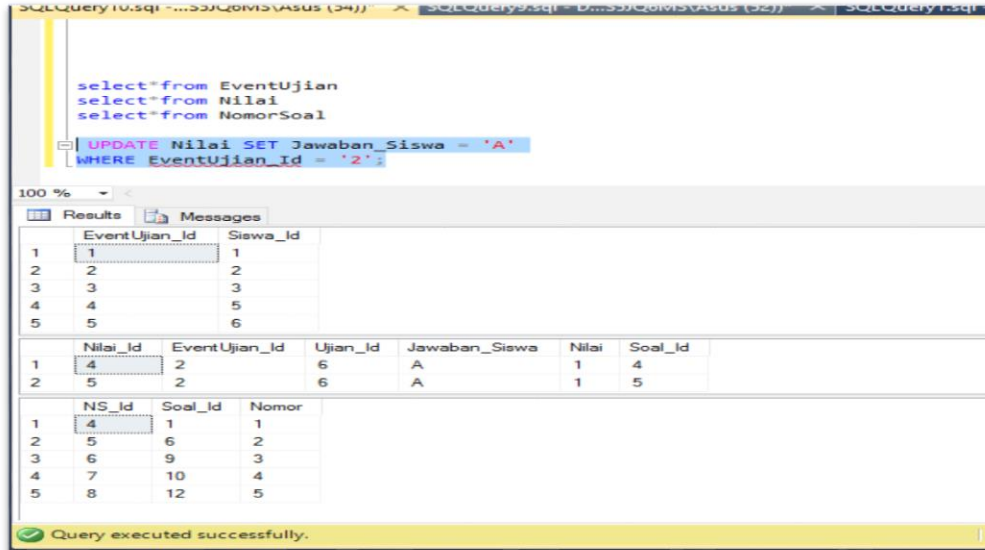
Dari Gambar 4.20 terlihat ketika dilakukan *delete* data pada tabel Nilai, maka tabel lain tidak ikut terhapus, hanya pada tabel Nilai yang terhapus, sehingga sudah tidak terdapat *anomaly*.

3) *Update Anomaly*



Gambar 4. 21 Tabel Sebelum Pengujian

Gambar 4.21 merupakan keadaan awal sebelum dilakukan pengujian *update anomaly*.

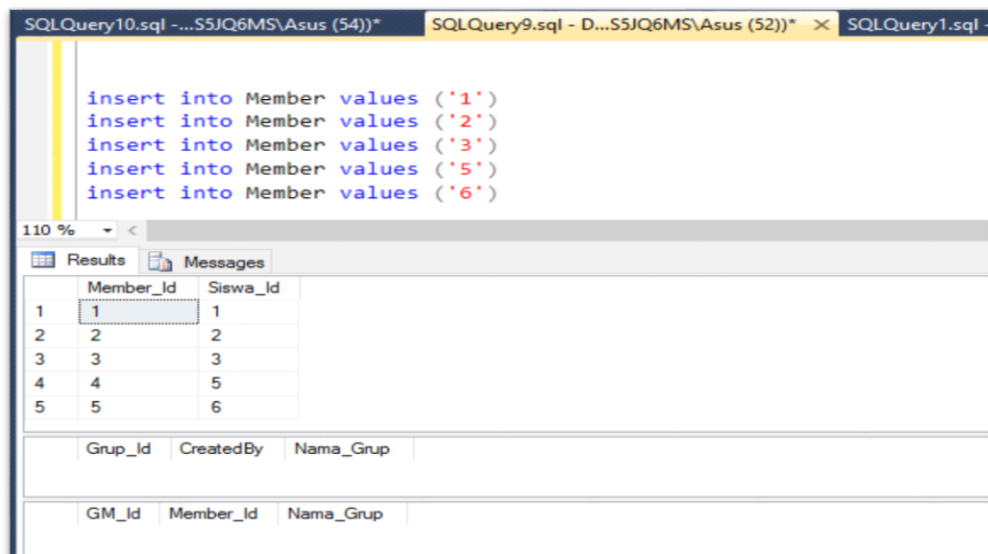


Gambar 4. 22 Pengujian *Update Anomaly*

Dari Gambar 4.22 terlihat ketika dilakukan *update* data pada tabel Nilai, tabel lain tidak ada perubahan pada tabel lain, hanya pada tabel Nilai yang mengalami perubahan, sehingga sudah tidak terdapat *anomaly*.

b. Tabel GrupMember, Tabel Member, dan Tabel Grup

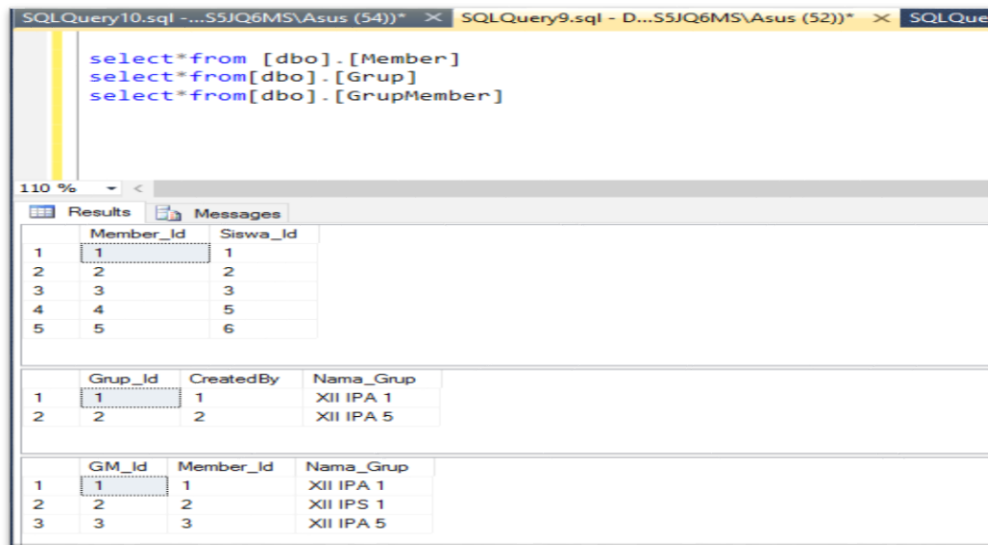
1) *Insert Anomaly*



Gambar 4. 23 Pengujian *Insert Anomaly*

Pada Gambar 4.23, terlihat ketika dilakukan *insert* data pada tabel Member, tabel lain tidak ada perubahan, hanya pada tabel Member yang mengalami penambahan data, sehingga sudah tidak terdapat *anomaly*.

2) Delete Anomaly



```

select*from [dbo].[Member]
select*from[dbo].[Grup]
select*from[dbo].[GrupMember]

```

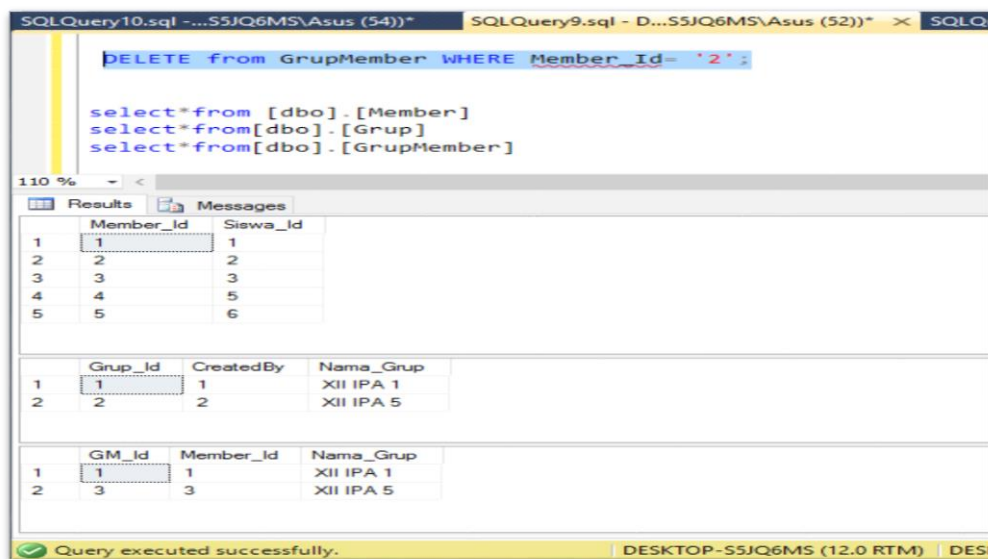
	Member_Id	Siswa_Id
1	1	1
2	2	2
3	3	3
4	4	5
5	5	6

	Grup_Id	CreatedBy	Nama_Grup
1	1	1	XII IPA 1
2	2	2	XII IPA 5

	GM_Id	Member_Id	Nama_Grup
1	1	1	XII IPA 1
2	2	2	XII IPS 1
3	3	3	XII IPA 5

Gambar 4. 24 Tabel Sebelum Pengujian

Gambar 4.24 merupakan keadaan awal sebelum dilakukan pengujian *delete anomaly*.



```

DELETE from GrupMember WHERE Member_Id= '2';

select*from [dbo].[Member]
select*from[dbo].[Grup]
select*from[dbo].[GrupMember]

```

	Member_Id	Siswa_Id
1	1	1
2	2	2
3	3	3
4	4	5
5	5	6

	Grup_Id	CreatedBy	Nama_Grup
1	1	1	XII IPA 1
2	2	2	XII IPA 5

	GM_Id	Member_Id	Nama_Grup
1	1	1	XII IPA 1
2	3	3	XII IPA 5

Query executed successfully.

Gambar 4. 25 Pengujian *Delete Anomaly*

Dari Gambar 4.25 terlihat ketika dilakukan *delete* data pada tabel GrupMember, tabel lain tidak ikut terhapus, hanya pada tabel GrupMember yang terhapus, sehingga sudah tidak terdapat *anomaly*.

3) Update Anomaly

The screenshot shows a SQL query window with the following code:

```
select * from [dbo].[Member]
select * from [dbo].[Grup]
select * from [dbo].[GrupMember]
```

The results are displayed in three tables:

Member_Id	Siswa_Id
1	1
2	2
3	3
4	5
5	6

Grup_Id	CreatedBy	Nama_Grup
1	1	XII IPA 1
2	2	XII IPA 5

GM_Id	Member_Id	Nama_Grup
1	1	XII IPA 1
2	2	XII IPA 5
3	3	XII IPA 5

Gambar 4. 26 Tabel Sebelum Pengujian

Gambar 4.26 merupakan keadaan awal sebelum dilakukan pengujian *update anomaly*.

The screenshot shows a SQL query window with the following code:

```
insert into Member values ('5')
insert into Member values ('6')
UPDATE GrupMember SET Nama_Grup = 'XII IPS 1'
WHERE Member_Id = 2;
```

The results are displayed in three tables:

Member_Id	Siswa_Id
1	1
2	2
3	3
4	5
5	6

Grup_Id	CreatedBy	Nama_Grup
1	1	XII IPA 1
2	2	XII IPA 5

GM_Id	Member_Id	Nama_Grup
1	1	XII IPA 1
2	2	XII IPS 1
3	3	XII IPA 5

The status bar at the bottom indicates: Query executed successfully.

Gambar 4. 27 Pengujian Update Anomaly

Dari Gambar 4.27 terlihat ketika dilakukan *update* data pada tabel GrupMember, maka tabel lain tidak ada perubahan, hanya pada tabel GrupMember yang mengalami perubahan, sehingga sudah tidak terdapat *anomaly*.

B. Pengujian *Constraint*

Constraint basis data merupakan struktur yang dibuat oleh pengguna atau perancang basis data yang mencerminkan perilaku dari suatu tabel dan kolom. *Constraint* dirancang pertama pada saat mendefinisikan basis data dengan tujuan utama memproteksi validasi data.

a. *Check Constraint* pada Kolom Status_Ujian

```
create table PesertaUjian (
  Peserta_Id int identity(1,1) primary key,
  Siswa_Id nvarchar (128) not null,
  foreign key (Siswa_Id) references[Siswa](Siswa_Id),
  Ujian_Id int foreign key references Ujian(Ujian_Id),
  Status_Ujian int CONSTRAINT CHK_Status_Ujian CHECK (Status_Ujian in( 0, 1))
);
```

Gambar 4. 28 Gambar *Query Constraint* pada Kolom Status Ujian

Pada Gambar 4.28, *check constraint* diimplementasikan pada tabel pesertaUjian kolom Status_Ujian, pada kolom ini hanya diisi angka 0 atau 1.



Gambar 4. 29 Validasi *Check constraint* pada Kolom

Pada Gambar 4.29, saat pengujian *insert* data yang tidak sesuai dengan pembatasan data pada kolom Status_Ujian, maka data ketika *execute query* muncul validasi. Hal ini berarti *constraint* telah berhasil diimplementasikan.

b. *Check constraint* pada Kolom Nilai

```

create table Nilai (
Nilai_Id int identity(1,1) primary key,
EventUjian_Id int foreign key references EventUjian(EventUjian_Id),
Soal_Id int foreign key references NomorSoal(NS_Id),
Ujian_Id int foreign key references Ujian(Ujian_Id),
Nilai int CONSTRAINT CHK_Nilai CHECK (Nilai in( 0, 1)),
Jawaban_Siswa char(1)
);

```

Gambar 4. 30 Query Constraint pada kolom Nilai

Pada Gambar 4.30, *check constraint* diimplementasikan pada tabel Nilai kolom Nilai, pada kolom ini hanya diisi angka 0 atau 1.



Gambar 4. 31 Validasi Check Constraint pada kolom Nilai

Pada Gambar 4.31, saat pengujian *insert* data yang tidak sesuai dengan pembatasan data pada kolom Nilai, maka data ketika *execute query* muncul validasi. Hal ini berarti *constraint* telah berhasil diimplementasikan.

c. Check constraint pada kolom Jenis_Kelamin Guru

```

create table Guru (
Guru_Id nvarchar (128) not null,
primary key clustered (Guru_Id),
foreign key (Guru_Id) references[AspNetUsers](Id),
NIP char(18),
Nama_Guru varchar (20),
Jenis_kelamin char(1) constraint checkJenis_kelamin check(Jenis_kelamin in('L','P')),
Sekolah varchar(30),
Alamat varchar (25),
MP_Id int foreign key references MataPelajaran(MP_Id),
);

```

Gambar 4. 32 Query Constraint pada Kolom Jenis_Kelamin Guru

Pada Gambar 4.32, *check constraint* diimplementasikan pada tabel Guru kolom Jenis_Kelamin, pada kolom ini hanya diisi huruf “P” atau “L”.

```

insert into Guru values ('6','123456789012345679','Reihan','K','SMA 1 Batang','Batang',1);

```

Msg 547, Level 16, State 0, Line 141
The INSERT statement conflicted with the CHECK constraint "Jenis_kelamin_Guru". The conflict occurred in database "examdb", table "dbo.Guru", column 'Jenis_kelamin'.
The statement has been terminated.

Gambar 4.33 Validasi *Check Constraint* pada Kolom *Jenis_Kelamin* Guru

Pada Gambar 4.33, saat pengujian *insert* data yang tidak sesuai dengan pembatasan data pada kolom *Jenis_Kelamin*, maka data ketika *execute query* muncul validasi. Hal ini berarti *constraint* telah berhasil diimplementasikan.

d. *Check constraint* pada kolom *Jenis_Kelamin* Siswa

```

create table Siswa (
Siswa_Id nvarchar (128) not null,
primary key clustered (Siswa_Id),
foreign key (Siswa_Id) references[AspNetUsers](Id),
NISN char(10),
Nama_Siswa varchar (20),
Jenis_kelamin char(1)constraint checkJenis_kelamin check(Jenis_kelamin in('L','P')),
Sekolah varchar (30),
Kelas varchar(10),
Alamat varchar (25));

```

Gambar 4.34 *Query Constraint* pada Kolom *Jenis_Kelamin* Siswa

Pada Gambar 4.34, *check constraint* diimplementasikan pada tabel *Siswa* kolom *Jenis_Kelamin*, pada kolom ini hanya diisi huruf “P” atau “L”.

```

insert into Siswa values ('8', 1234567890, 'Anggun', 'G','SMA 1 Batang', 'XII IPA 4', 'Jl.Gatak Kasihan Bantul')

```

Msg 547, Level 16, State 0, Line 30
The INSERT statement conflicted with the CHECK constraint "checkJenis_kelamin_Siswa". The conflict occurred in database "examdb", table "dbo.Siswa", column 'Jenis_kelamin'.
The statement has been terminated.

Gambar 4.35 Validasi *Check Constraint* pada Kolom *Jenis_Kelamin* Siswa

Pada Gambar 4.35, saat pengujian *insert* data yang tidak sesuai dengan pembatasan data pada kolom *Jenis_Kelamin*, maka data ketika *execute query* muncul validasi. Hal ini berarti *constraint* telah berhasil diimplementasikan.