EXISTS ve NOT EXISTS fonksiyonu örneği

DECLARE @sipno int

Select @sipno = 1

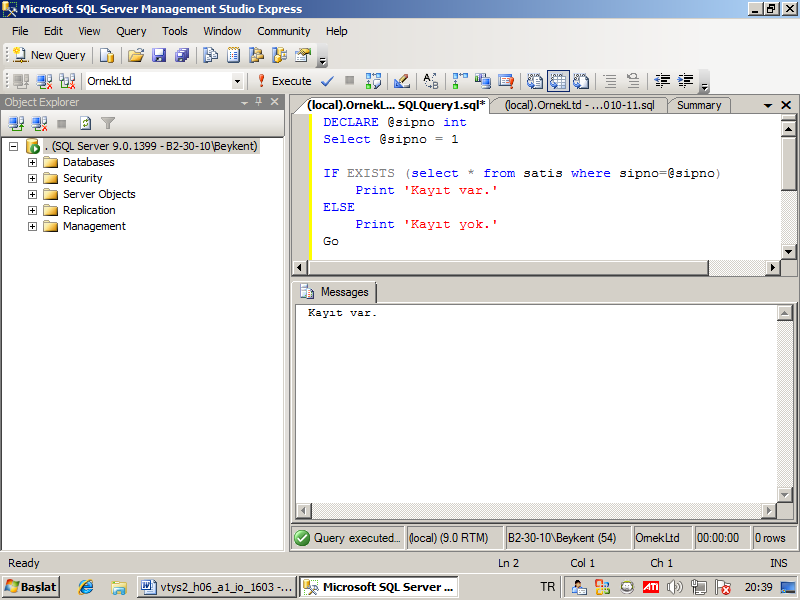
IF EXISTS (select \* from satis where sipno=@sipno)

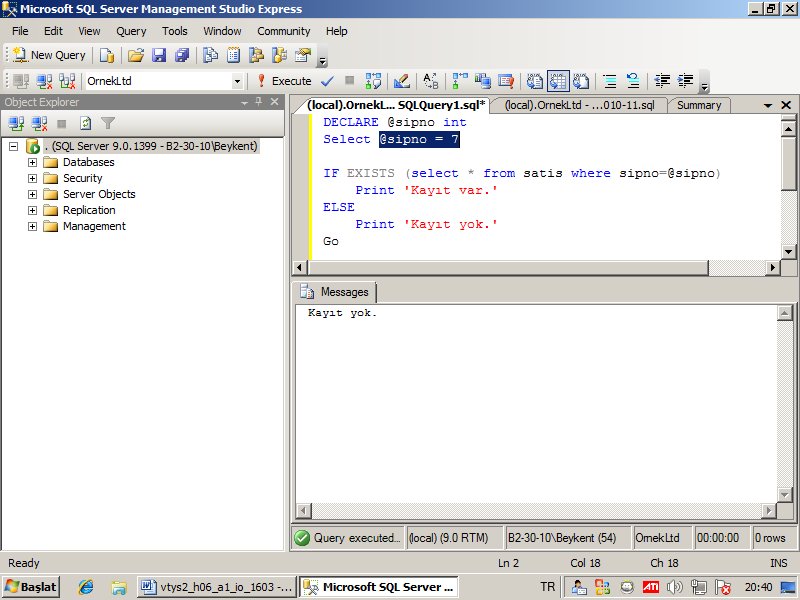
Print ‘Kayıt var.’

ELSE

Print ‘Kayıt yok.’

Go





DECLARE @sipno int

Select @sipno = 1

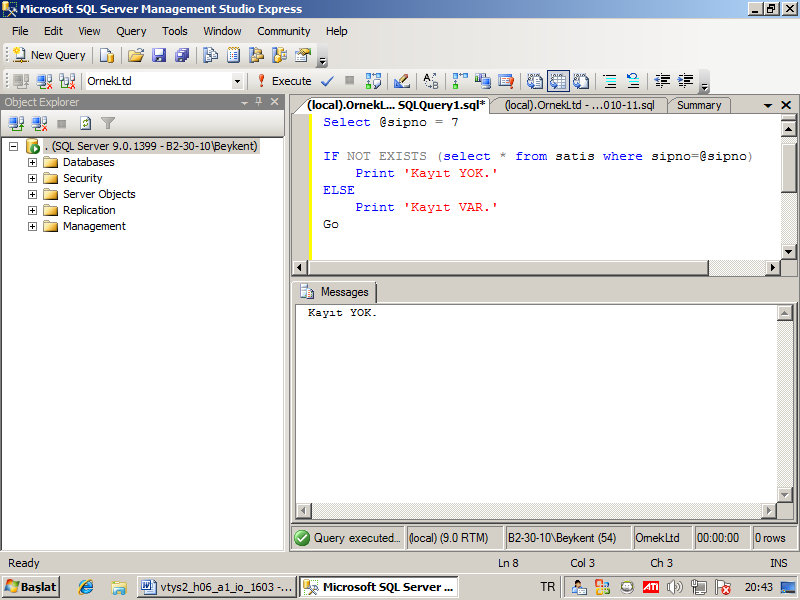
IF NOT EXISTS (select \* from satis where sipno=@sipno)

Print ‘Kayıt YOK.’

ELSE

Print ‘Kayıt VAR.’

Go



TRIGGER ÖRNEKLERİ

Örnek:

Satış tablosuna veri girildiğinde çalışan ve veri girildi mesajı veren bir trigger oluşturunuz. Satis tablosuna veri girerek trigger’ı test ediniz.

CREATE TRIGGER trg\_satis\_gir

ON satis

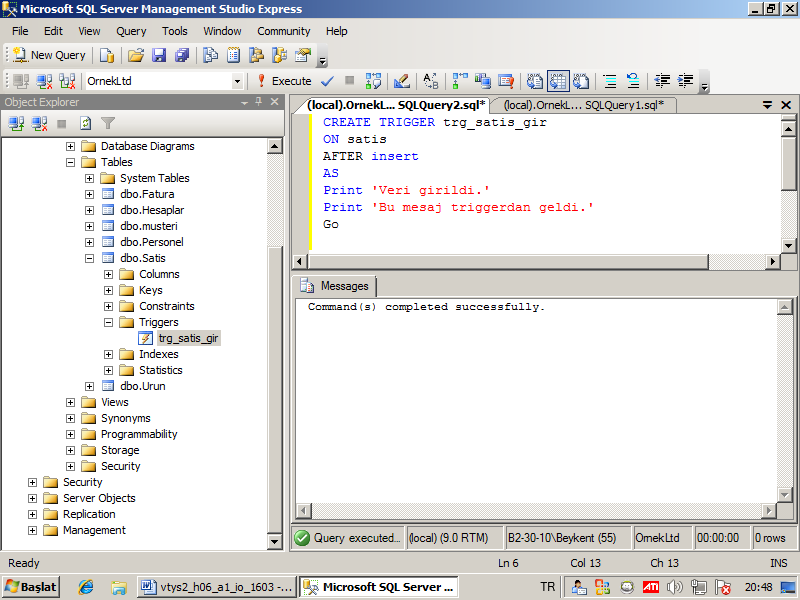
AFTER insert

AS

Print ‘Veri girildi.’

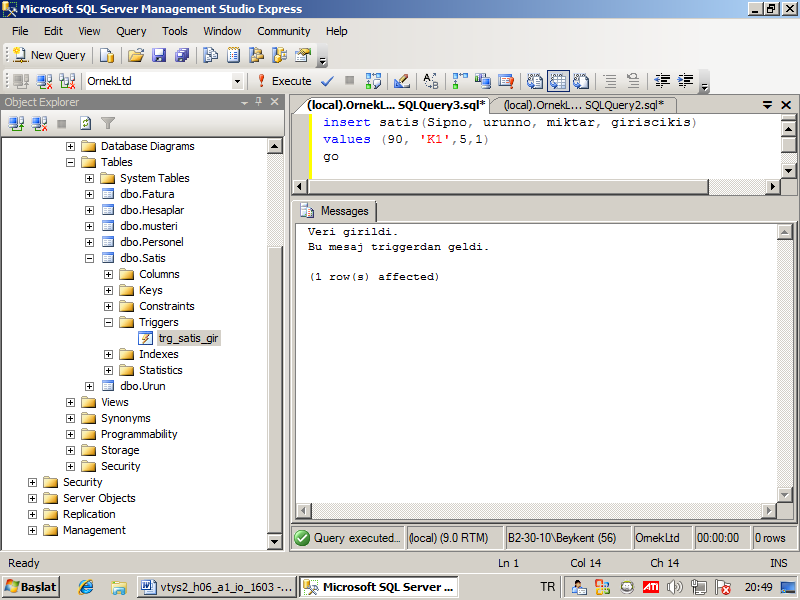
Print ‘Bu mesaj triggerdan geldi.’

Go



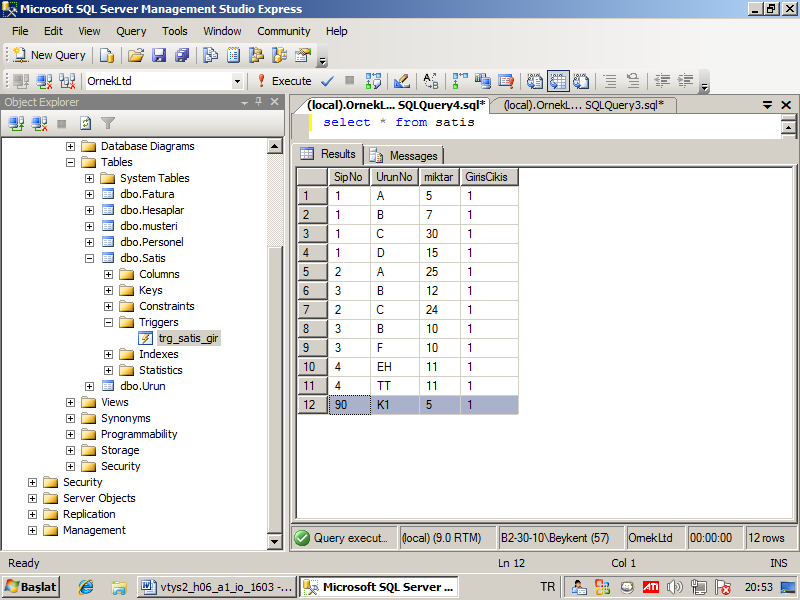
insert satis(Sipno, urunno, miktar, giriscikis) values (90, ‘K1’,5,1)

go



Örnek:

Yukarıdaki trigger’ı değiştirelim. İnsert deyimi ile girilen bilgileri değişkenlere atayalım. Bu değişkenlerin değerlerini yazdıralım.



ALTER TRIGGER trg\_satis\_gir

ON satis

AFTER insert

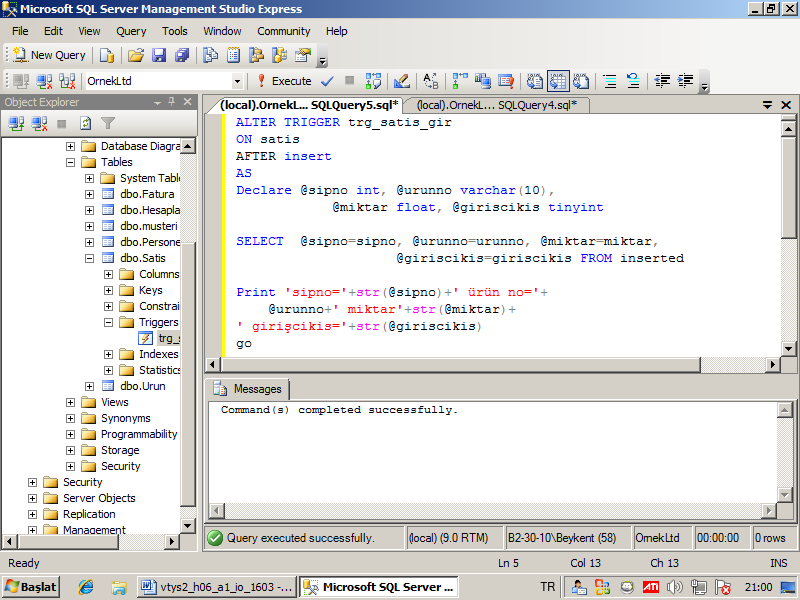
AS

Declare @sipno int, @urunno varchar(10), @miktar float, @giriscikis tinyint

SELECT @sipno=sipno, @urunno=urunno, @miktar=miktar, @giriscikis=giriscikis FROM inserted

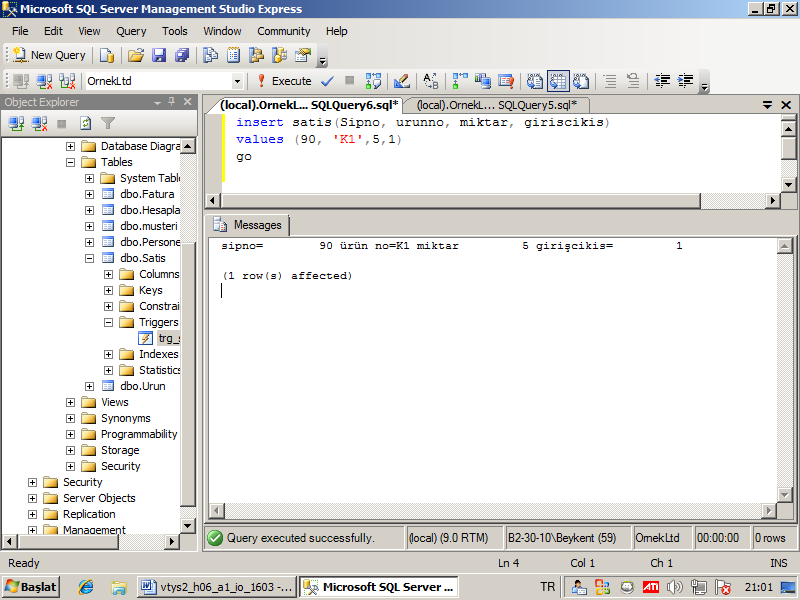
Print ‘sipno=’+str(@sipno)+’ ürün no=’+ @urunno+’ miktar‘+str(@miktar)+   
‘ girişcikis=’+str(@giriscikis)

Go



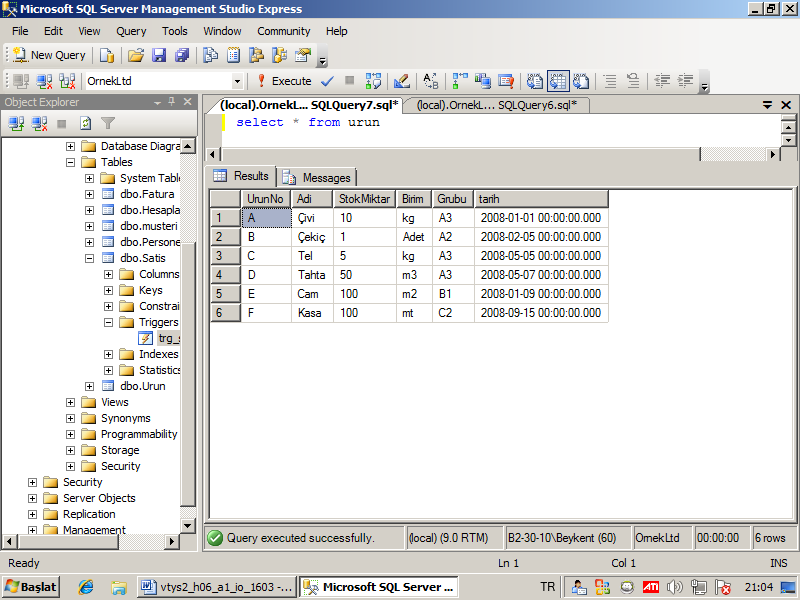
insert satis(Sipno, urunno, miktar, giriscikis) values (90, ‘K1’,5,1)

go



Örnek:

Yukarıdaki trigger’ı değiştirelim. Bu trigger girilen ürün, ürün tablosunda yoksa kaydı iptal etsin.



ALTER TRIGGER trg\_satis\_gir

ON satis

AFTER insert

AS

-- satış tablosuna girilen ürünün ürün numarasını bulalım.

-- değişken tanımla

Declare @urunno varchar(10)

-- girilen ürünno değerini değişkene atayalım

Select @urunno=urunno from inserted

IF NOT EXISTS (select \* from urun where urunno =@urunno)

Begin

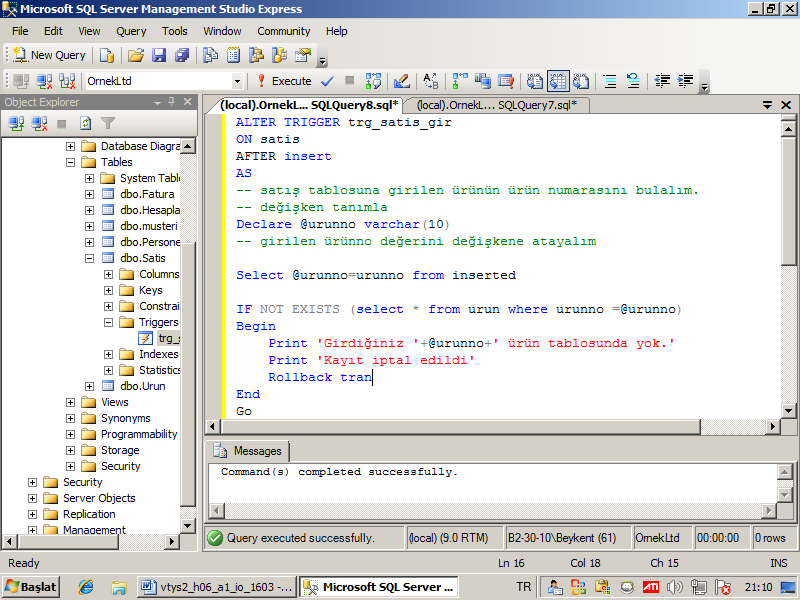
Print ‘Girdiğiniz ‘+@urunno+‘ ürün tablosunda yok.’

Print ‘Kayıt iptal edildi’

Rollback tran

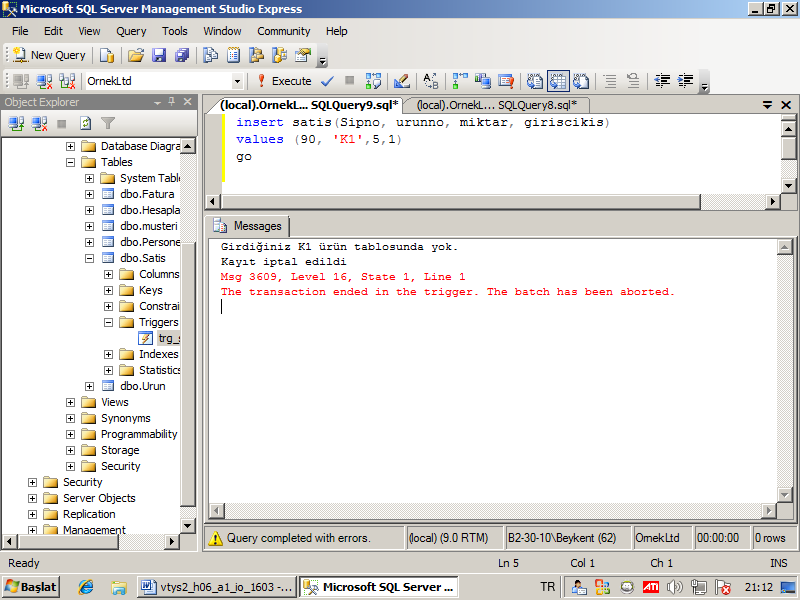
End

Go



insert satis(Sipno, urunno, miktar, giriscikis) values (90, ‘K1’,5,1)

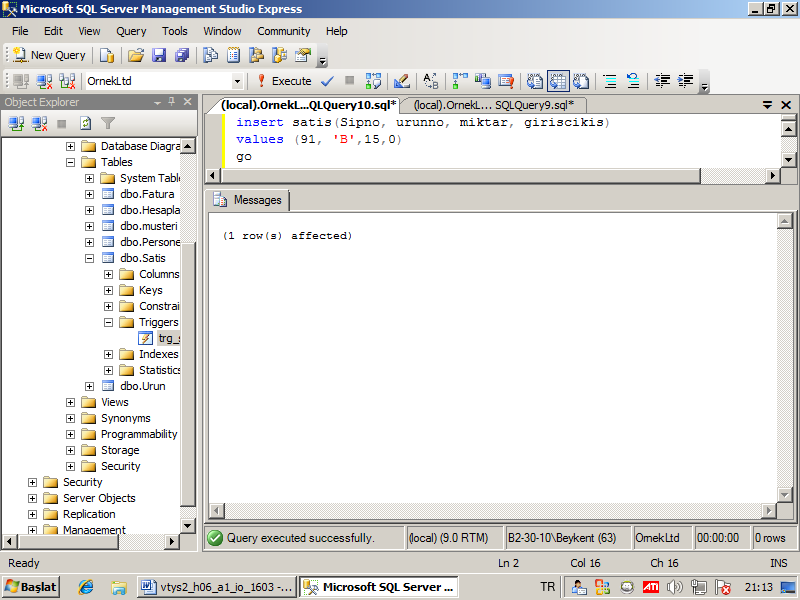
go



K1 nolu ürün, ürün tablosunda yok. Kayıt trigger tarafından iptal edildi.

insert satis(Sipno, urunno, miktar, giriscikis) values (91, ‘B’,15,0)

go



B nolu ürün, ürün tablosunda var. Kayıt gerçekleşti.

Örnek:

Yukarıdaki trigger’ı değiştirelim. Satış tablosuna girilen ürün, ürün tablosunda yoksa bu kaydı ürün tablosuna eklesin.

ALTER TRIGGER trg\_satis\_gir

ON satis

AFTER insert

AS

-- girilen ürünno ve miktarı değişkenlere aktaralım.

Declare @urunno varchar(10), @miktar int

Select @urunno=urunno, @miktar=miktar from inserted

IF NOT EXISTS (select \* from urun where urunno=@urun)

BEGIN

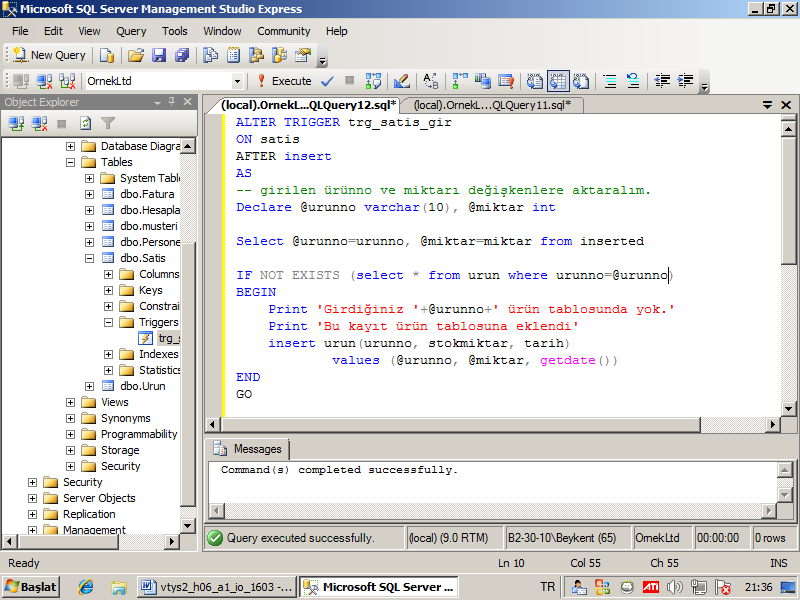
Print ‘Girdiğiniz ‘+@urunno+‘ ürün tablosunda yok.’

Print ‘Bu kayıt ürün tablosuna eklendi’

insert urun(urunno, stokmiktar, tarih) values (@urunno, @miktar, getdate())

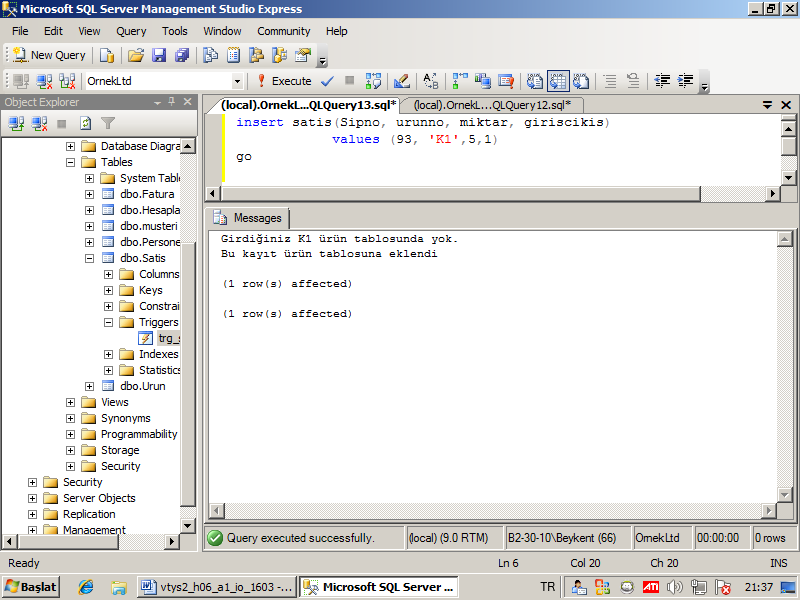
END

GO

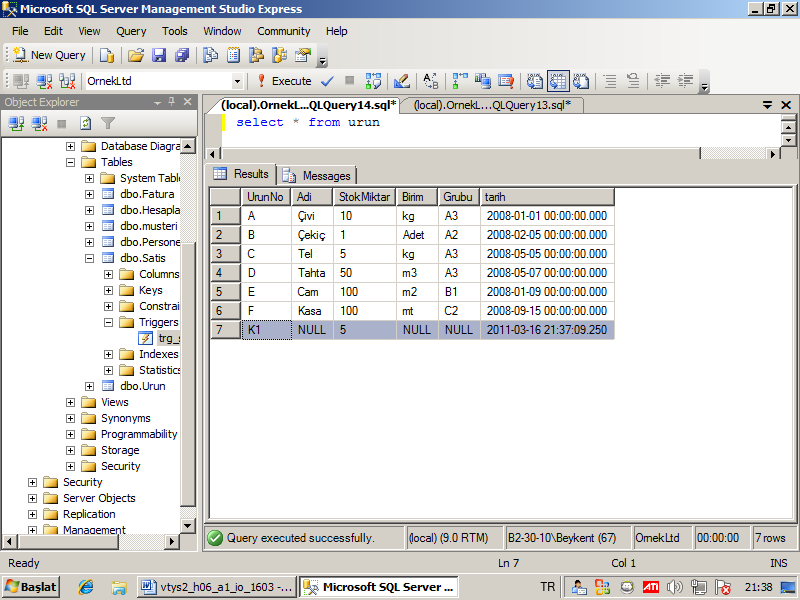


insert satis(Sipno, urunno, miktar, giriscikis) values (93, ‘K1’,5,1)

go



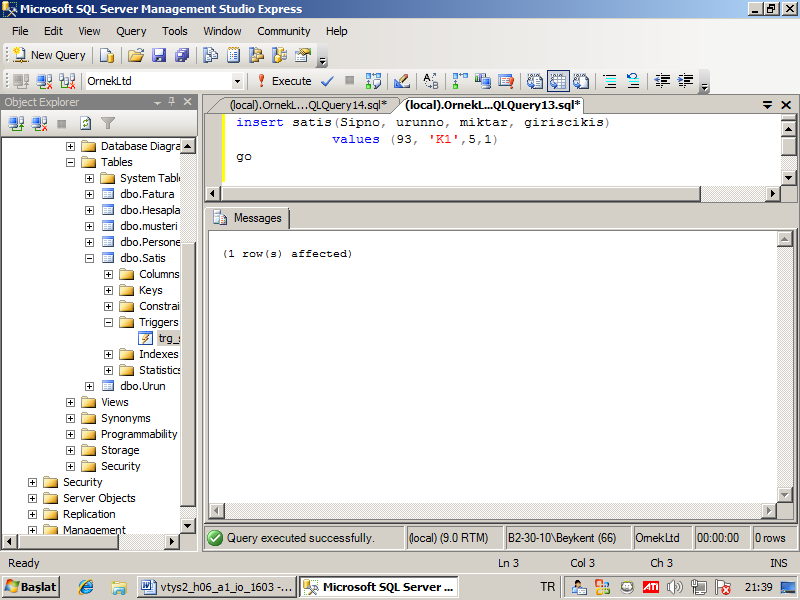
K1 nolu ürün, ürün tablosunda bulunmadığı için, trigger bu ürünle ilgili bilgileri ürün tablosuna ekledi. Bu aşağıdaki resimde görülmektedir.



Yukarıdaki insert deyimini tekrar çalıştırdık. Bir önceki insert deyimi K1 ürününü ürün tablosuna eklemişti. Bu nedenle artık ürün tablosunda K1 var. Aşağıdakini çalıştırdığımızda verdiği mesaj farklıdır.

insert satis(Sipno, urunno, miktar, giriscikis) values (93, ‘K1’,5,1)

go



Örnek:

Yukarıdaki triggerı tekrar değiştirelim. Giriş çıkış değerine bağlı olarak, ürün tablosundaki stok miktarını artırıp azaltsın. Girişçıkış değeri 0 ise azalsın, değilse artsın.

ALTER TRIGGER trg\_satis\_gir

ON satis

AFTER insert

AS

-- ihtiyacım olan değişkenleri tanımlayalım

Declare @urunno varchar(10), @miktar float, @giriscikis tinyint

-- girilen değerleri değişkenlere atayalım.

Select @urunno=urunno, @miktar=miktar, @giriscikis=giriscikis from inserted

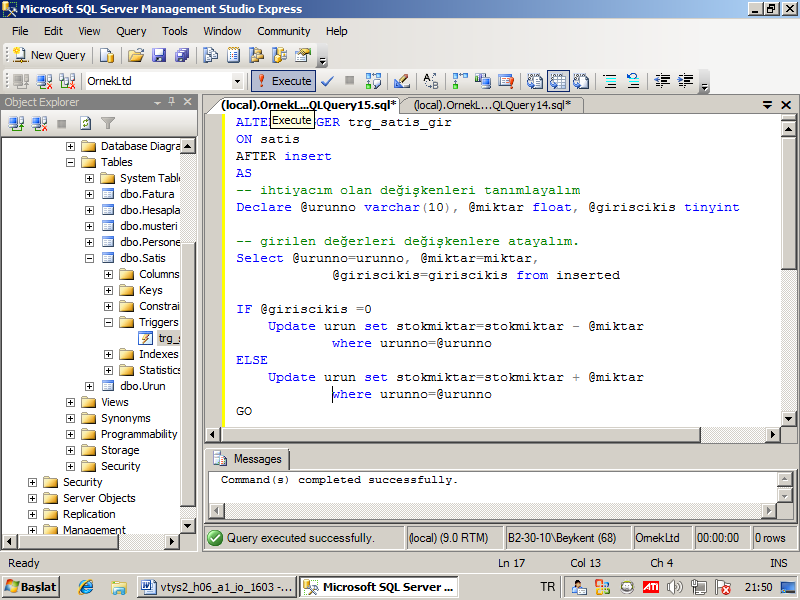
IF @giriscikis =0

Update urun set stokmiktar=stokmiktar - @miktar where urunno=@urunno

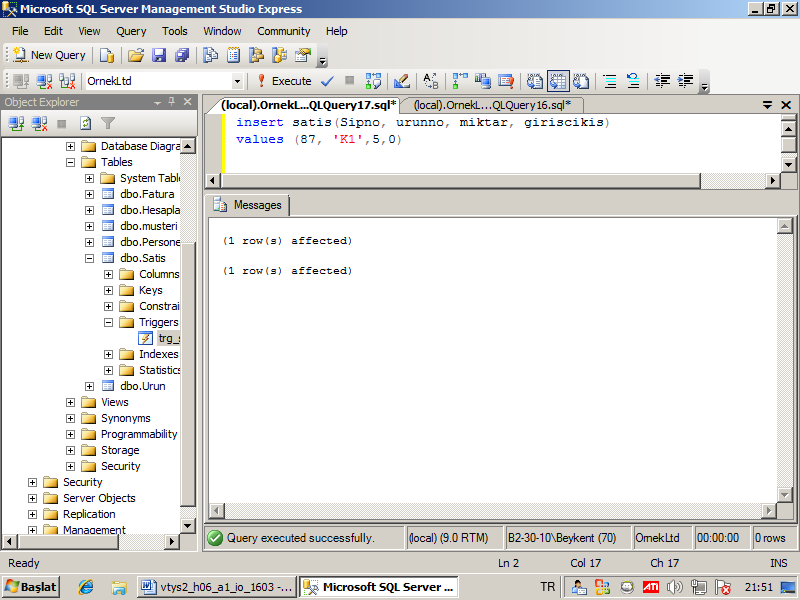
ELSE

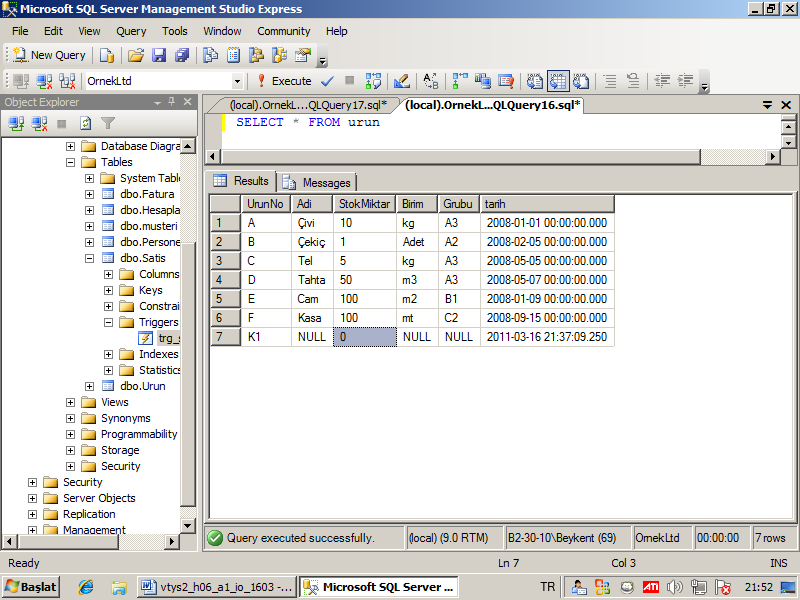
Update urun set stokmiktar=stokmiktar + @miktar where urunno=@urunno

GO

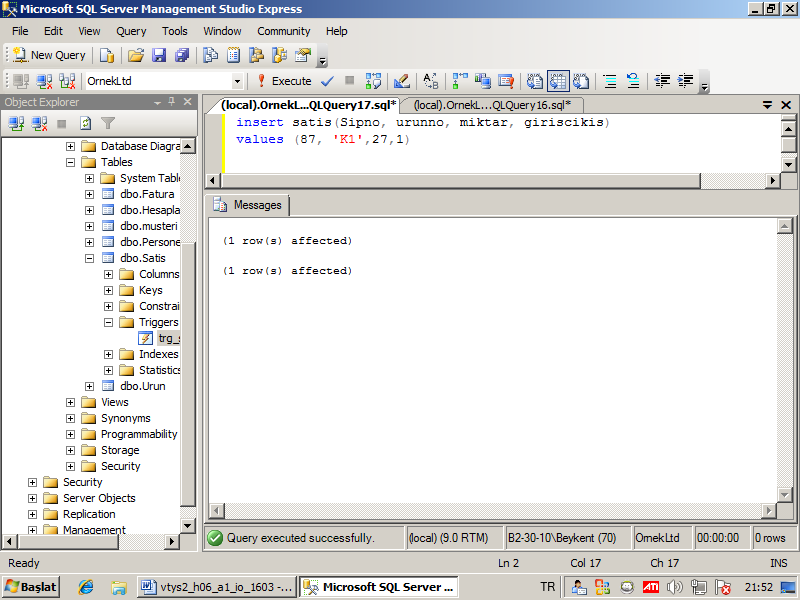


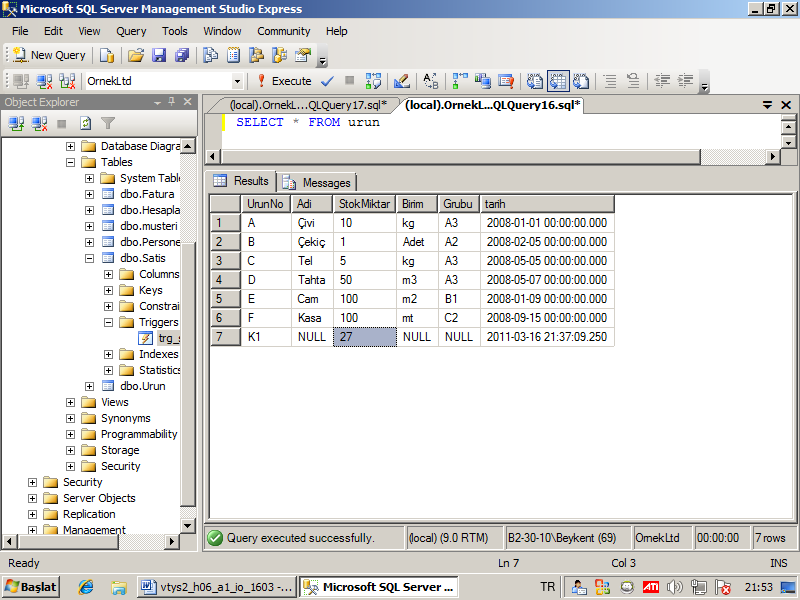
insert satis(Sipno, urunno, miktar, giriscikis) values (87, ‘K1’,5,0)





insert satis(Sipno, urunno, miktar, giriscikis) values (87, ‘K1’,27,1)





STORED PROCEDURE ÖRNEKLERİ

Örnek:

Fatura tablosunda veri değiştiren bir stored prosedür yazınız. Bu prosedür kaç kaydın güncellendiğini parametreden geri döndürsün.

CREATE PROC sp\_fatura\_guncelle

-- giriş parametreleri

@mno int,

@urunno varchar(10),

@miktar float,

@fiyat Money,

-- çıkış parametesi

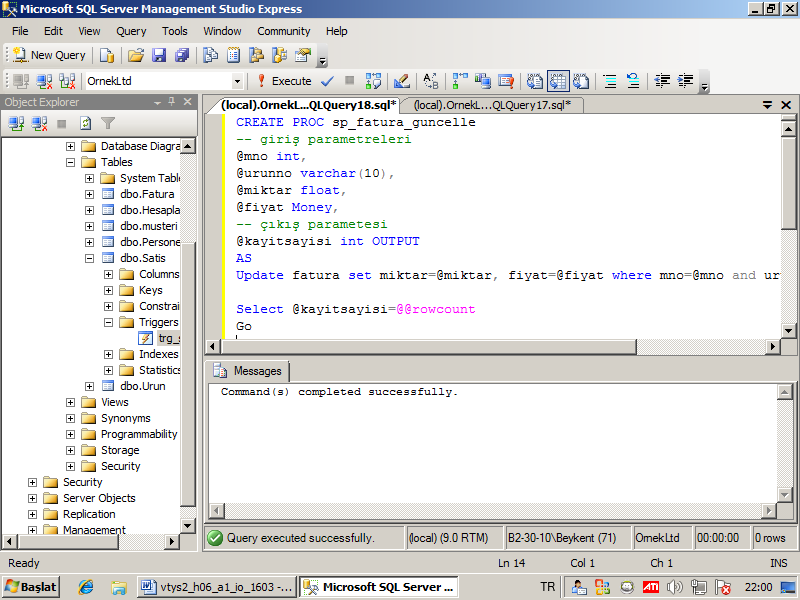
@kayitsayisi int OUTPUT

AS

Update fatura set miktar=@miktar, fiyat=@fiyat where mno=@mno and urunno=@urunno

Select @kayitsayisi=@@rowcount

Go

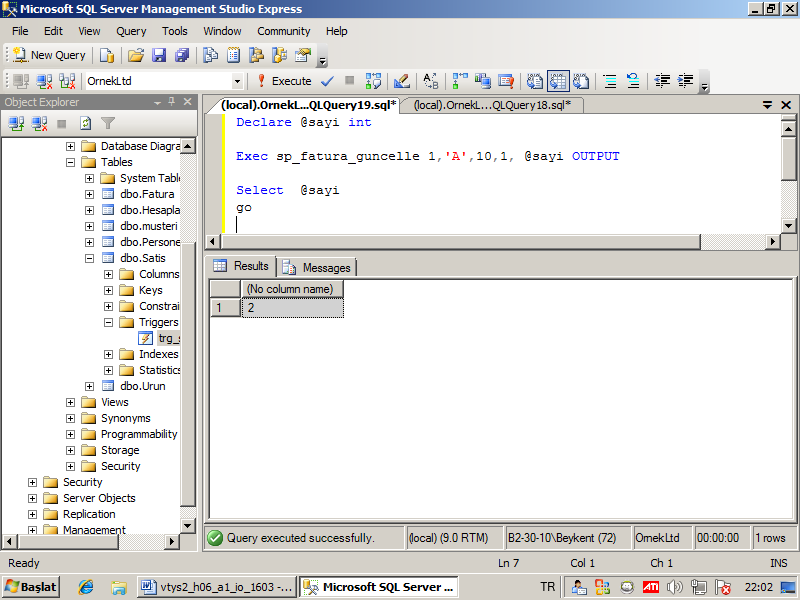


Declare @sayi int

Exec sp\_fatura\_guncelle 1,’A’,10,1, @sayi OUTPUT

Select @sayi

Go



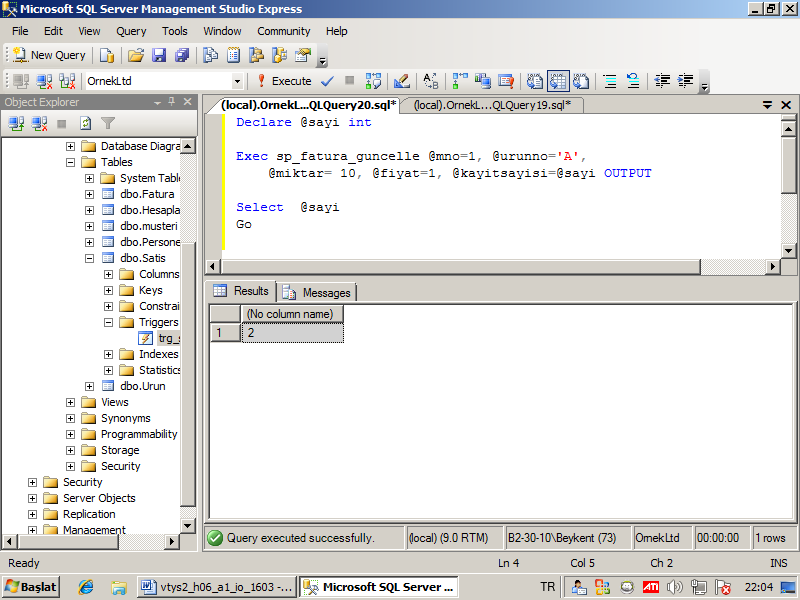
Yukarıdaki exec deyimini aşağıdaki gibi de yazabiliriz. Parametre ismi ile tekrar yazalım.

Declare @sayi int

Exec sp\_fatura\_guncelle @mno=1, @urunno=’A’, @miktar= 10, @fiyat=1, @kayitsayisi=@sayi OUTPUT

Select @sayi

Go



Örnek:

Fatura tablosuna veri giren bir prosedür oluşturunuz.

CREATE PROC sp\_fatura\_gir

@mno int=null,

@urunno varchar(10)=null,

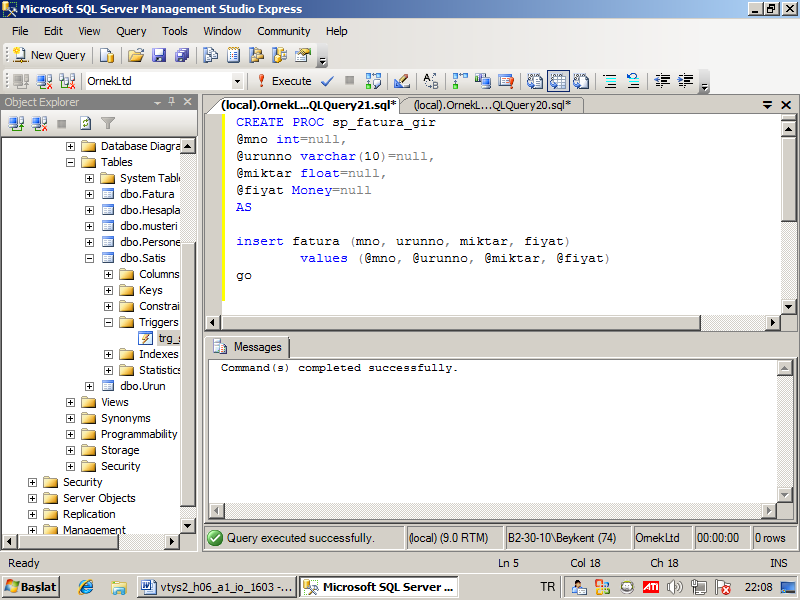
@miktar float=null,

@fiyat Money=null

AS

insert fatura (mno, urunno, miktar, fiyat) values (@mno, @urunno, @miktar, @fiyat)

go



Exec sp\_fatura\_gir @mno=21, @urunno=’B’, @miktar= 11, @fiyat=1.7

