EXISTS ve NOT EXISTS

DECLARE @sipno int

SELECT @sipno=1

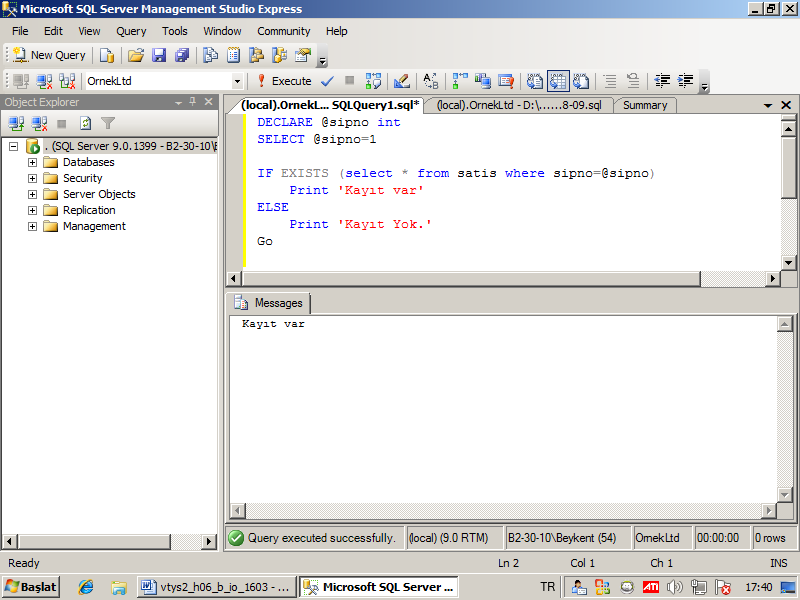
IF EXISTS (select \* from satis where sipno=@sipno)

Print ‘Kayıt var’

ELSE

Print ‘Kayıt Yok.’

Go



DECLARE @sipno int

SELECT @sipno=1

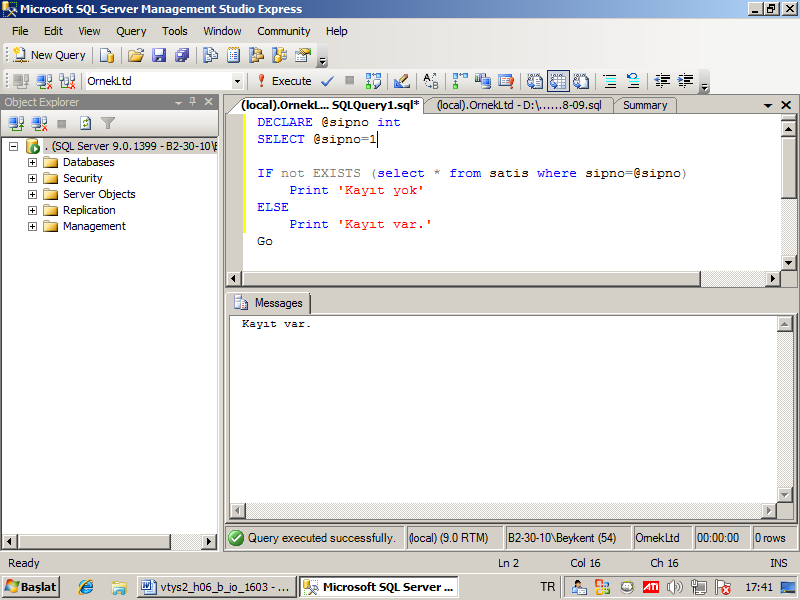
IF NOT EXISTS (select \* from satis where sipno=@sipno)

Print ‘Kayıt var’

ELSE

Print ‘Kayıt Yok.’

Go



TRIGGER ÖRNEKLERİ

Örnek:

Satış tablosuna veri girildiğinde çalışan ve veri girildi mesajı veren bir trigger yazınız. Satış tablosuna veri girerek trigger’ın çalıştığını görünüz.

CREATE TRIGGER trg\_satis\_gir

ON satis

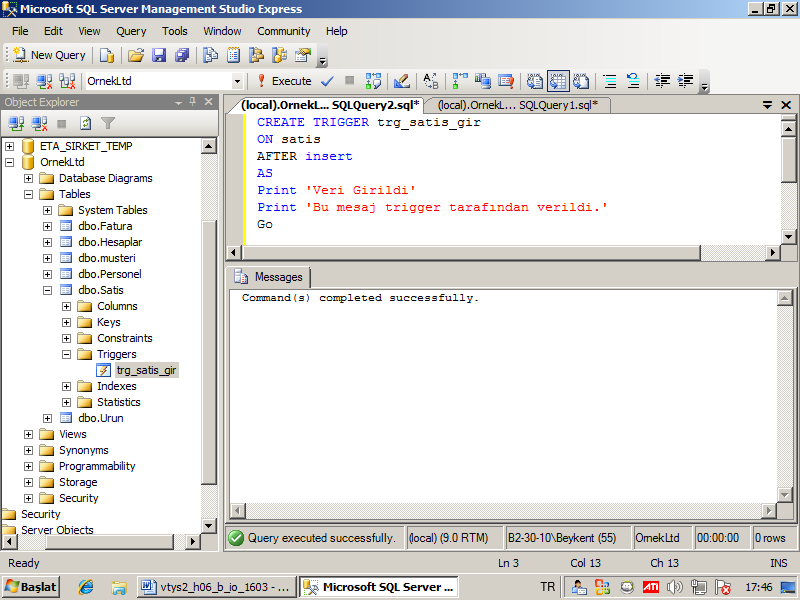
AFTER insert

AS

Print ‘Veri Girildi’

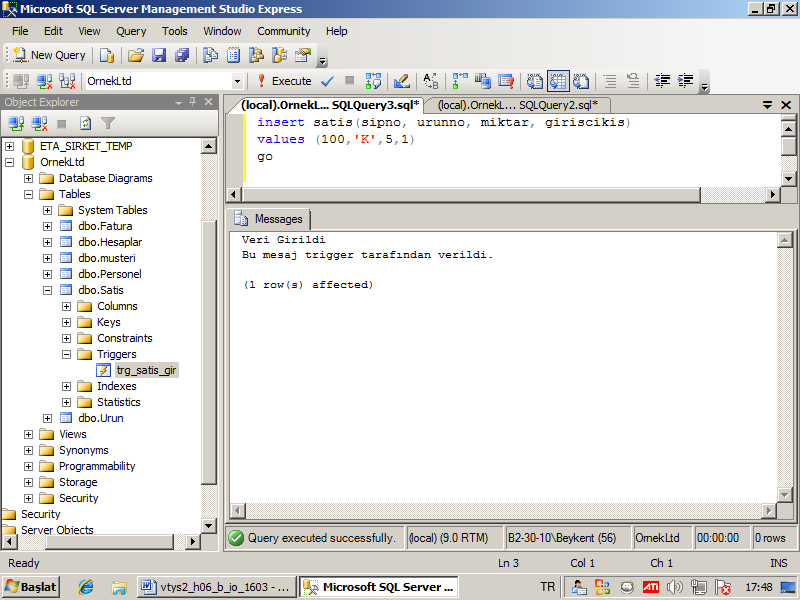
Print ‘Bu mesaj trigger tarafından verildi.’

Go



insert satis(sipno, urunno, miktar, giriscikis) values (100,’K’,5,1)

go



Örnek:

Yukarıdaki trigger’ı değiştirelim. Girilen verileri bir değişkene atayalım ve bu değişkenin değerini yazdıralım.

ALTER TRIGGER trg\_satis\_gir

ON satis

AFTER insert

AS

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

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

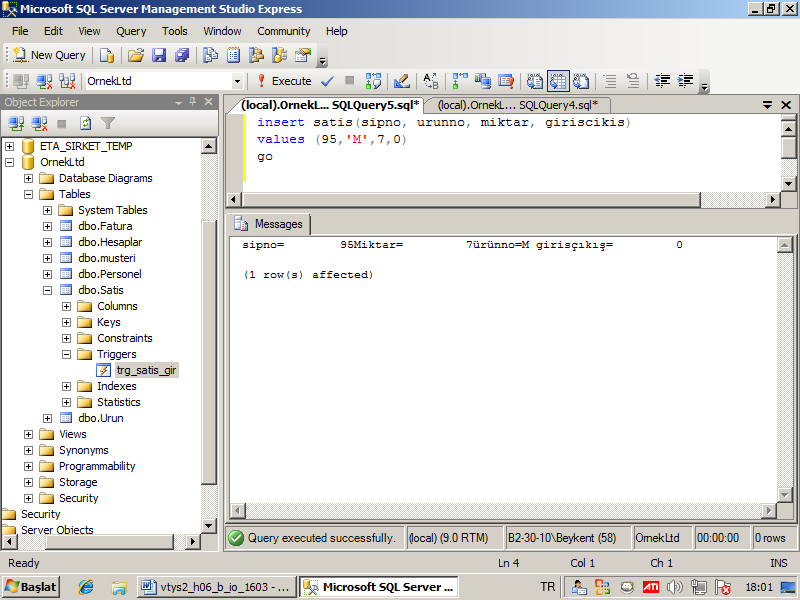
Print ‘sipno=’+str(@sipno)+ ‘Miktar=’+str(@miktar)+ ’ürünno=’+@urunno+’ girisçıkış=’+@giriscikis

Go



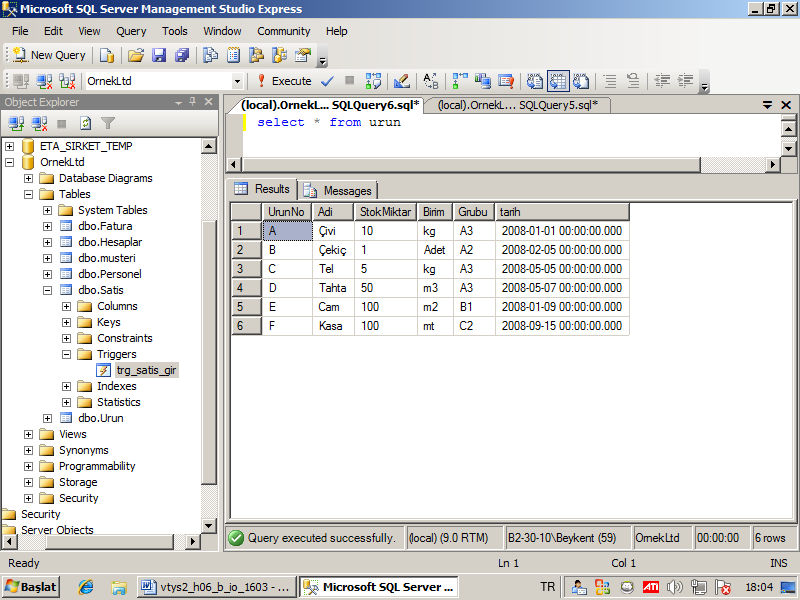
insert satis(sipno, urunno, miktar, giriscikis) values (95,’M’,7,0)

go



Örnek:

Yukarıdaki trigger’ı değiştireceğiz. Bu trigger girilen ürün, ürün tablosunda yok ise girilen satış kaydını iptal edecek.



ALTER TRIGGER trg\_satis\_gir

ON satis

AFTER insert

AS

-- girilen verideki ürün kodunu bulalım.

-- ürün kodu için değişken tanımlayalım.

Declare @urunno varchar(10)

-- girilen ürünno değerini değişkene atayalım

SELECT @urunno=urunno from inserted

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

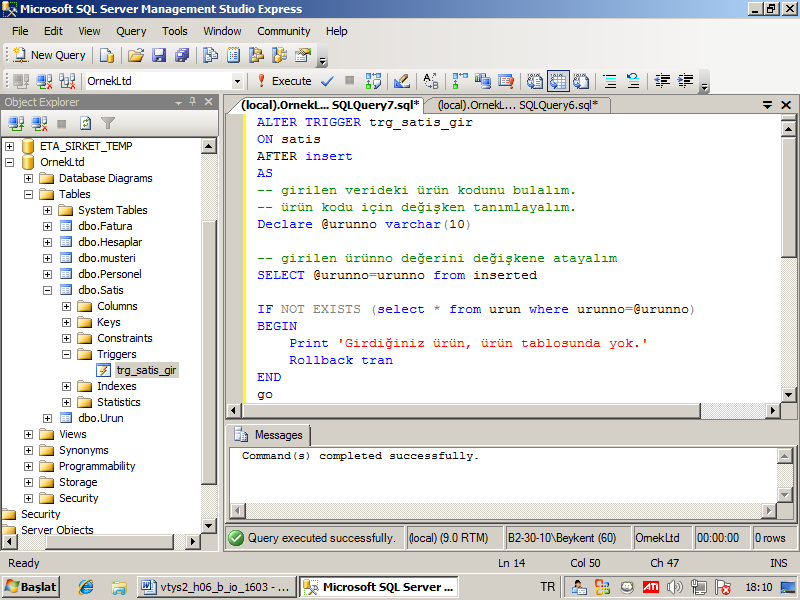
BEGIN

Print ‘Girdiğiniz ürün, ürün tablosunda yok.’

Rollback tran

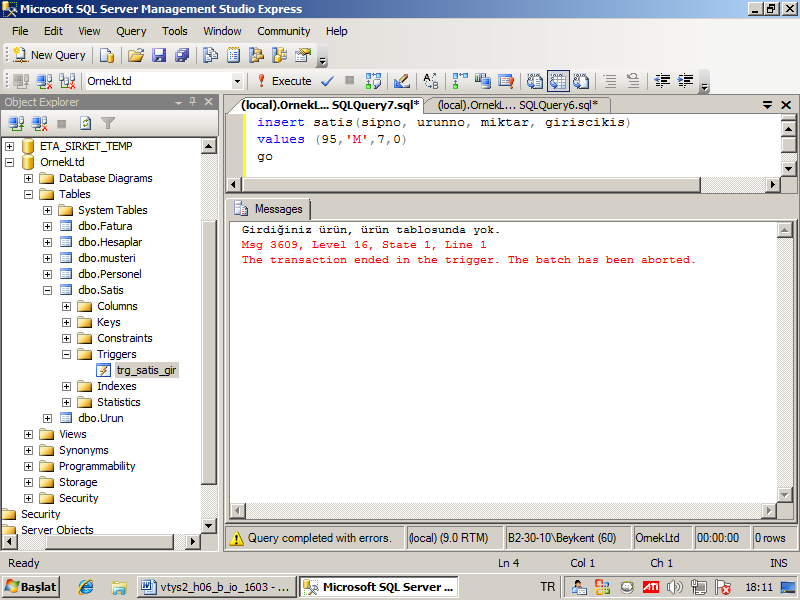
END

go



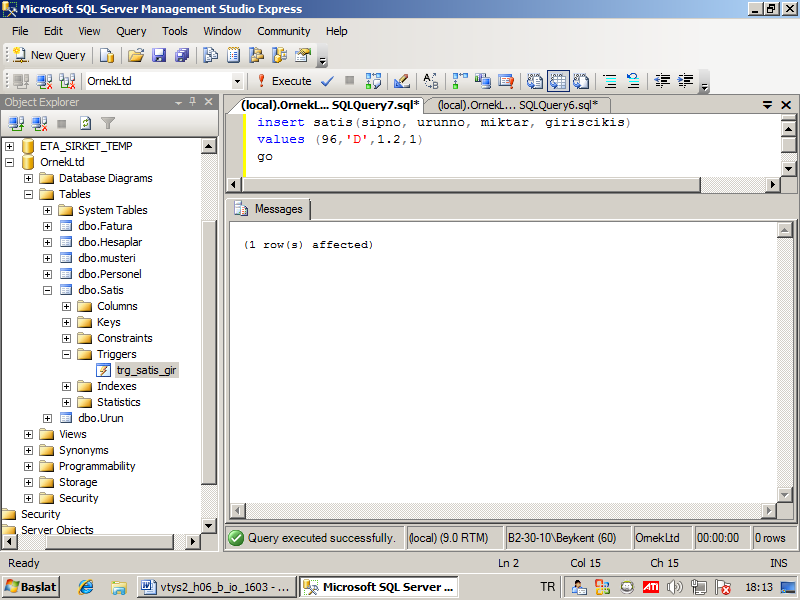
insert satis(sipno, urunno, miktar, giriscikis) values (95,’M’,7,0)

go



insert satis(sipno, urunno, miktar, giriscikis) values (96,’D’,1.2,1)

go



Örnek:

Yukarıdaki trigger’ı değiştirelim. Satis tablosuna girilen ürün, ürün tablosunda yoksa ürün tablosuna eklesin.

ALTER TRIGGER trg\_satis\_gir

ON satis

AFTER insert

AS

-- girilen verideki ürün kodunu bulalım.

-- ürün kodu için değişken tanımlayalım.

Declare @urunno varchar(10),  
@miktar float

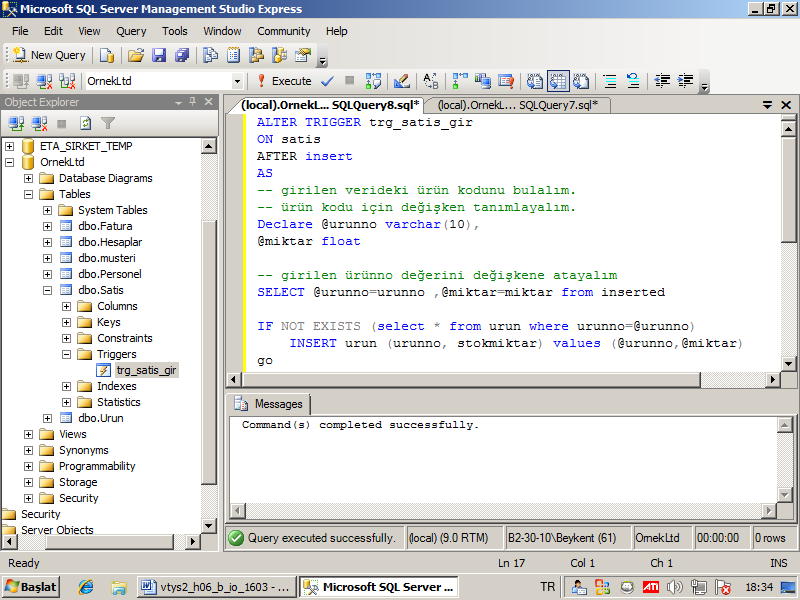
-- girilen ürünno değerini değişkene atayalım

SELECT @urunno=urunno ,@miktar=miktar from inserted

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

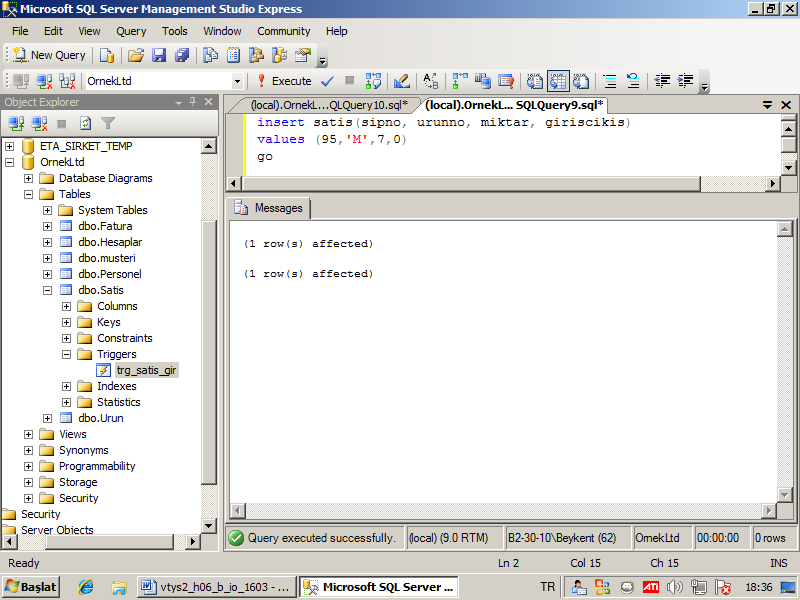
INSERT urun (urunno, stokmiktar) values (@urunno,@miktar)

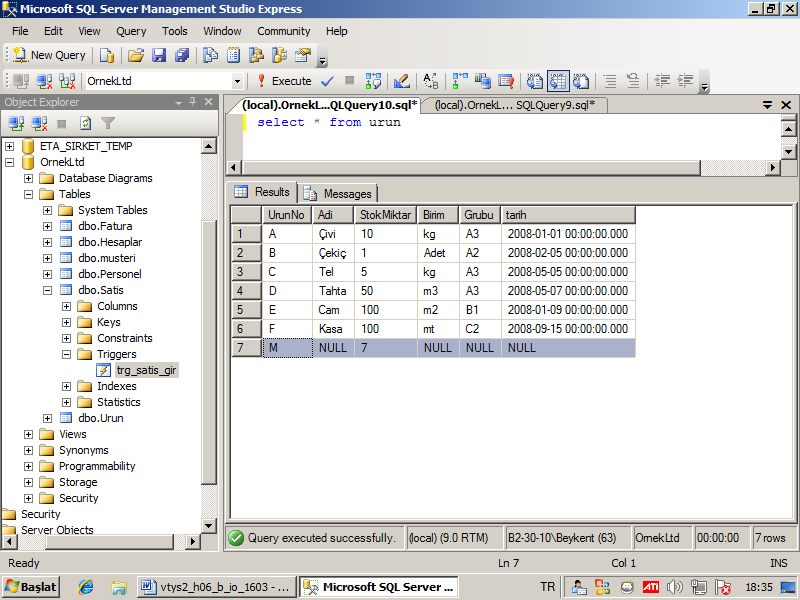
go



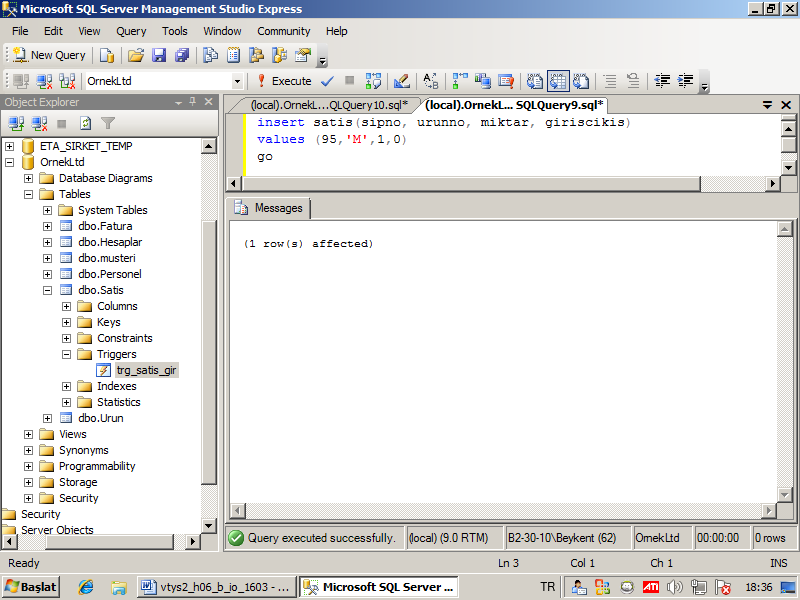
insert satis(sipno, urunno, miktar, giriscikis) values (95,’M’,7,0)

go





Yukarıdaki insert deyimini tekrar çalıştıralım.



Örnek:

Yukarıdaki trigger’ı değiştirelim. Satış tablosunda Girişçıkış değerine bağlı olarak, ürün tablosundaki miktarı artırıp azaltsın. Giriscikis değeri 1 ise azalsın, değilse artsın.

ALTER TRIGGER trg\_satis\_gir

ON satis

AFTER insert

AS

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

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

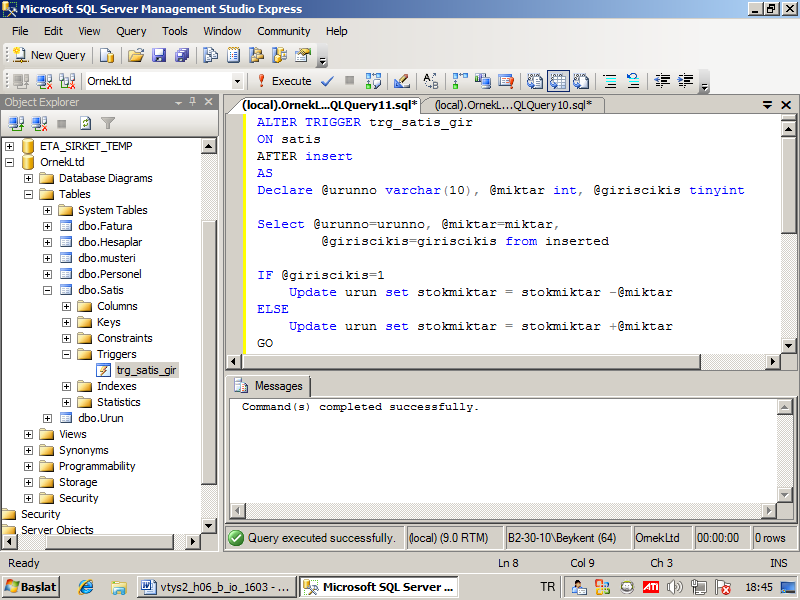
IF @giriscikis=1

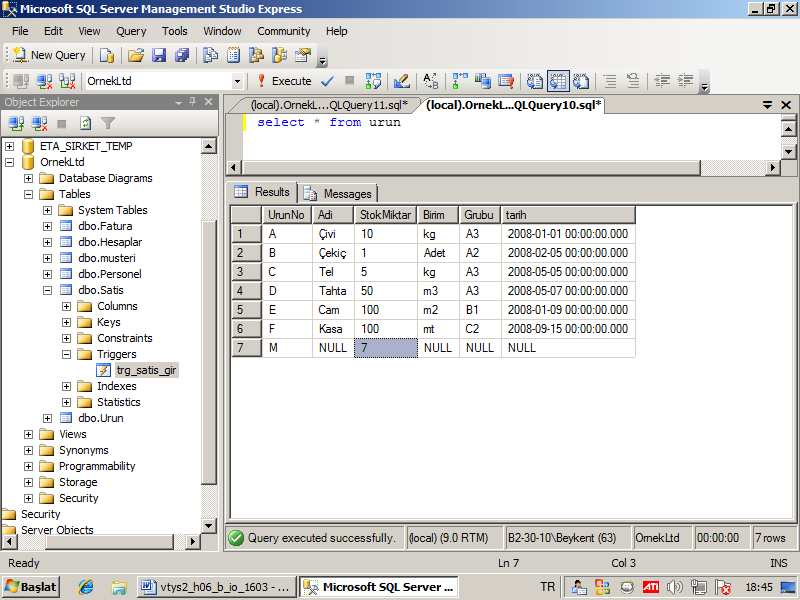
Update urun set stokmiktar = stokmiktar -@miktar

ELSE

Update urun set stokmiktar = stokmiktar +@miktar

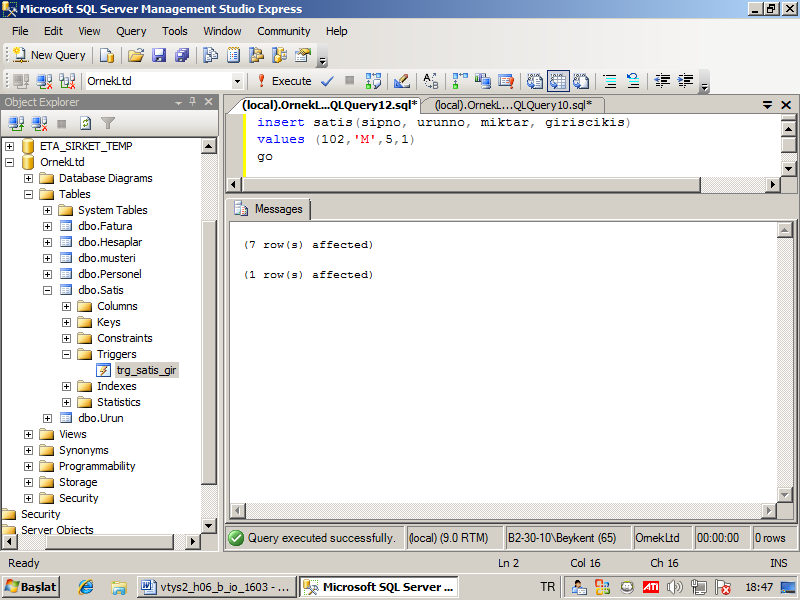
GO

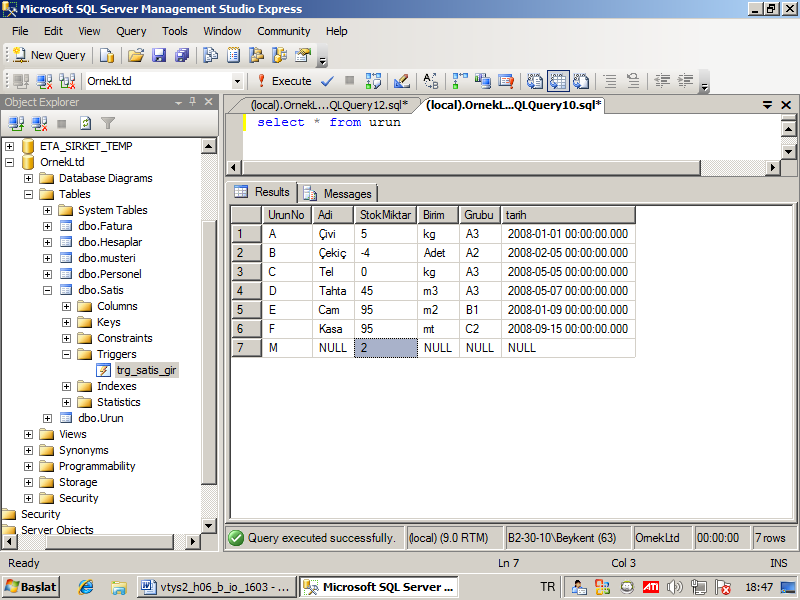




insert satis(sipno, urunno, miktar, giriscikis) values (102,’M’,5,1)

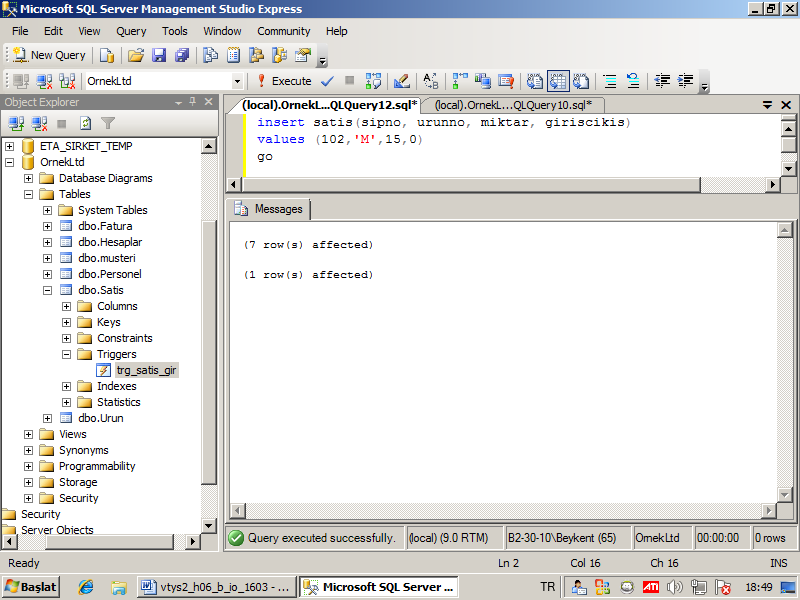
go

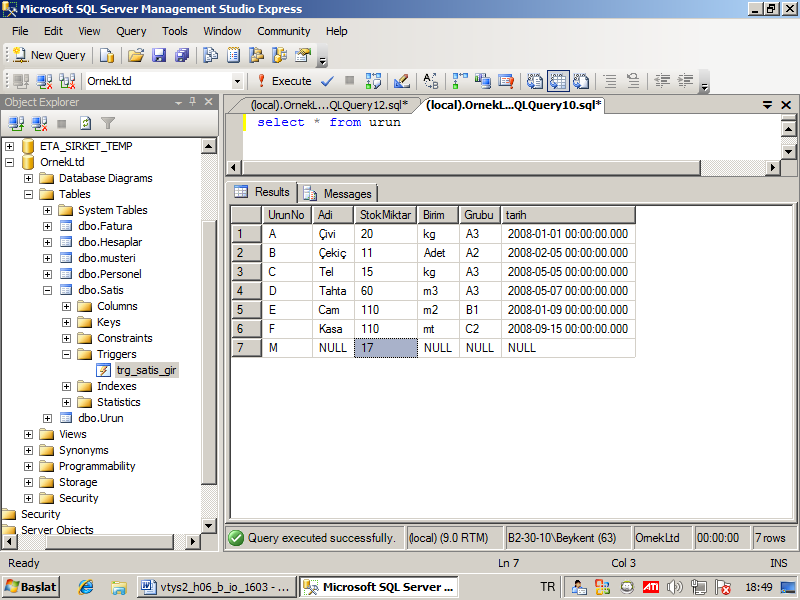




insert satis(sipno, urunno, miktar, giriscikis) values (102,’M’,15,0)

go





Stored Prosedure Örnekleri

Örnek:

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

CREATE PROC sp\_fatura\_guncelle

-- giriş parametreleri

@mno int,

@urunno varchar(10),

@miktar float,

@fiyat Money,

-- çıkış parametresi

@kayitsayisi int OUTPUT

AS

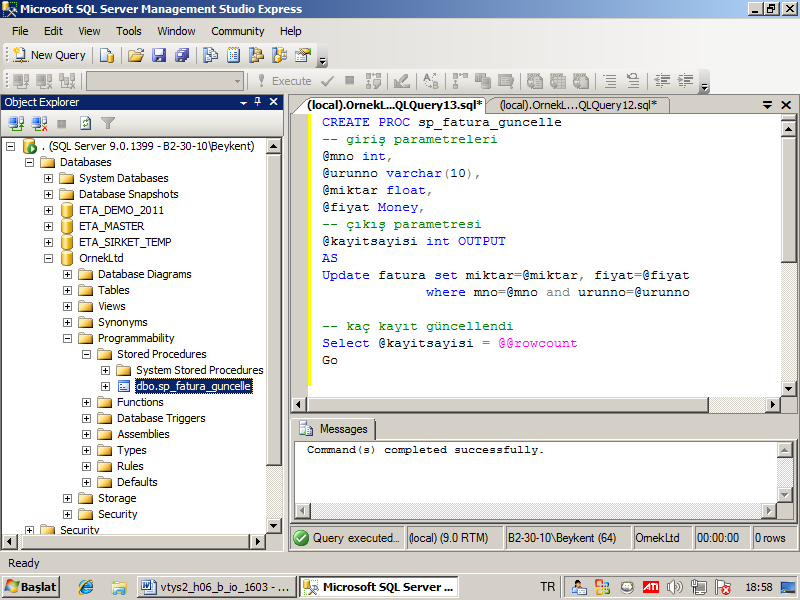
Update fatura set miktar=@miktar, fiyat=@fiyat where mno=@mno and urunno=@urunno

-- kaç kayıt güncellendi

Select @kayitsayisi = @@rowcount

Go

@@rowcount : etkilenen kayıt sayısı



Yukarıdaki prosedürü çalıştıralım.

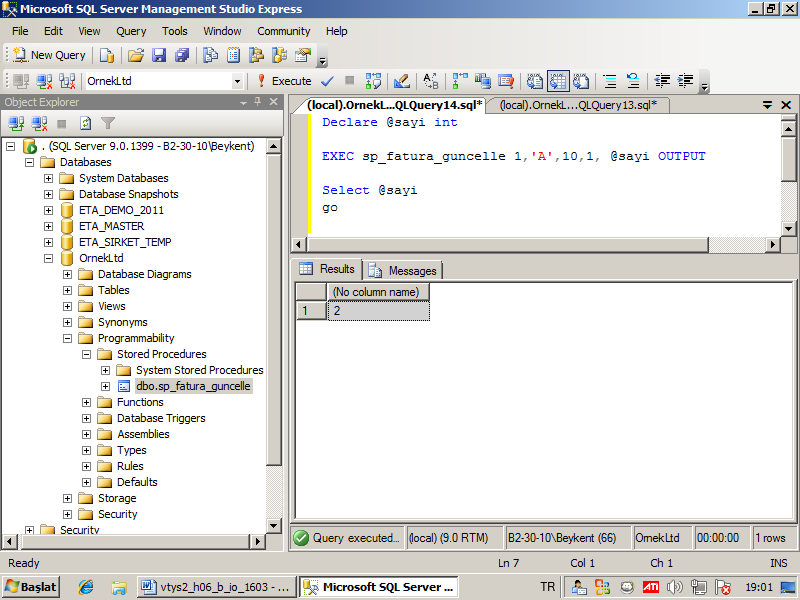
OUTPUT parametresinden geri dönen değeri okumak için değişken tanımı yapacağız.

Declare @sayi int

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

Select @sayi

Go



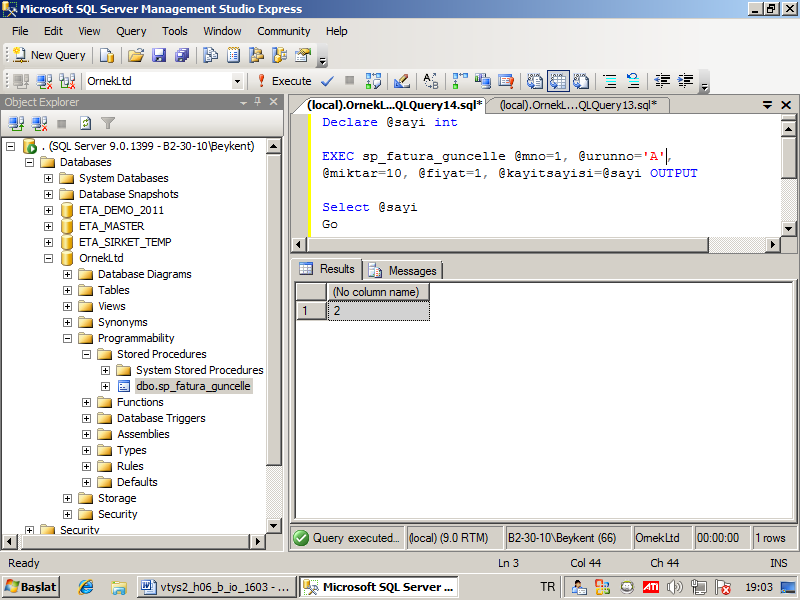
Yukarıdaki exec deyimi, parametre isimleri ile aşağıdaki gibi de yazılabilirdi.

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 yazınız.

CREATE PROC sp\_fatura\_verigir

@mno int=null,

@urunno varchar(10)=null,

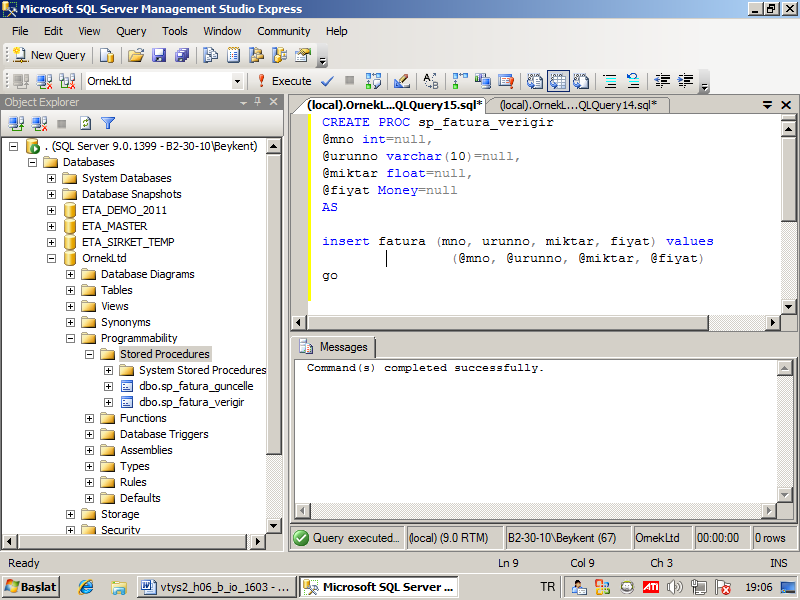
@miktar float=null,

@fiyat Money=null,

AS

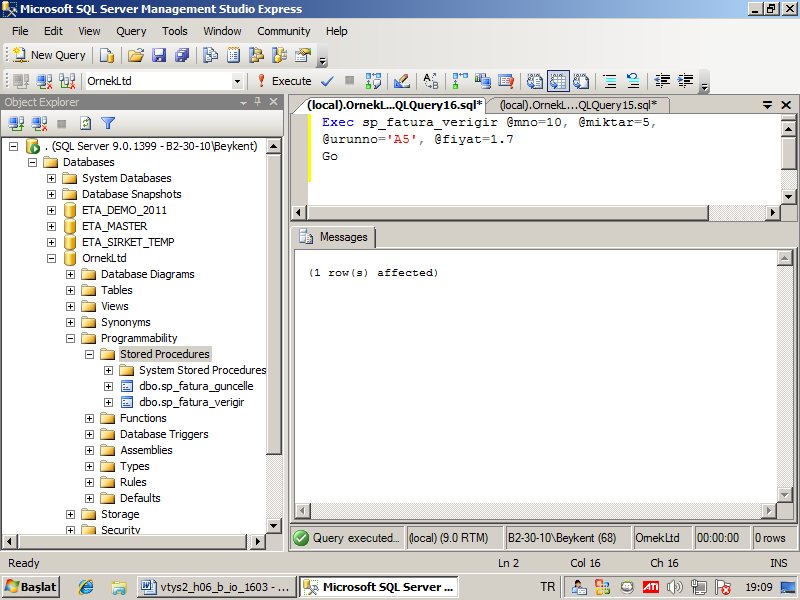
insert fatura (mno, urunno, miktar, fiyat) values (@mno, @urunno, @miktar, @fiyat)

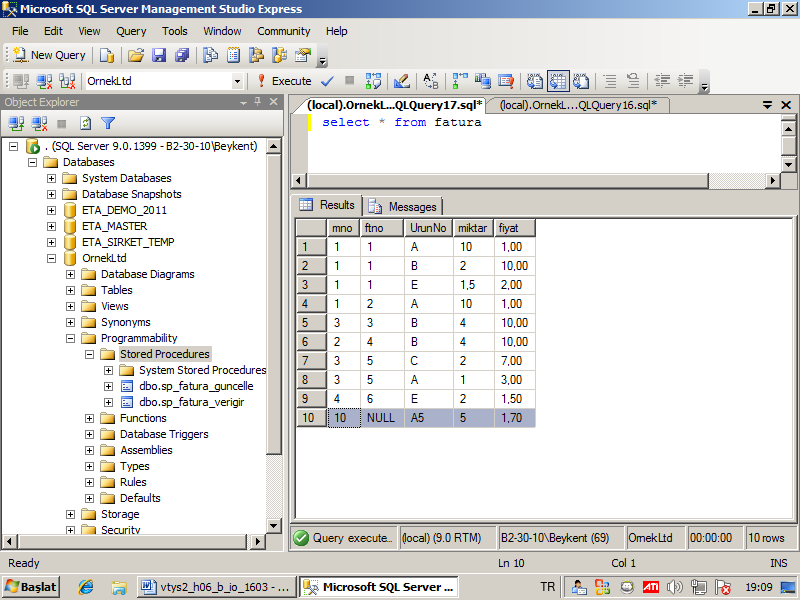
go



Exec sp\_fatura\_verigir @mno=10, @miktar=5, @urunno=’A5’, fiyat=1.7

Go





Örnek:

Fatura tablosundan verilen müşterinin tüm kayıtlarını silen bir prosedür yazınız.

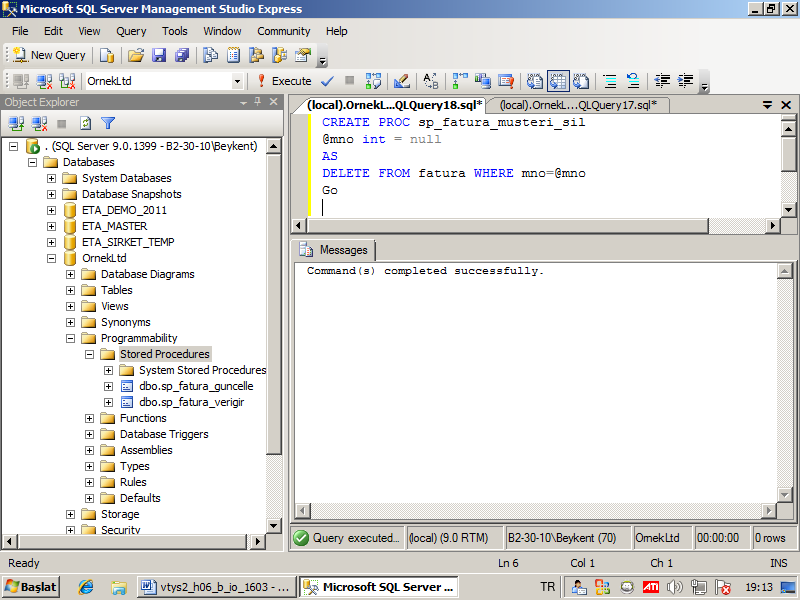
CREATE PROC sp\_fatura\_musteri\_sil

@mno int = null

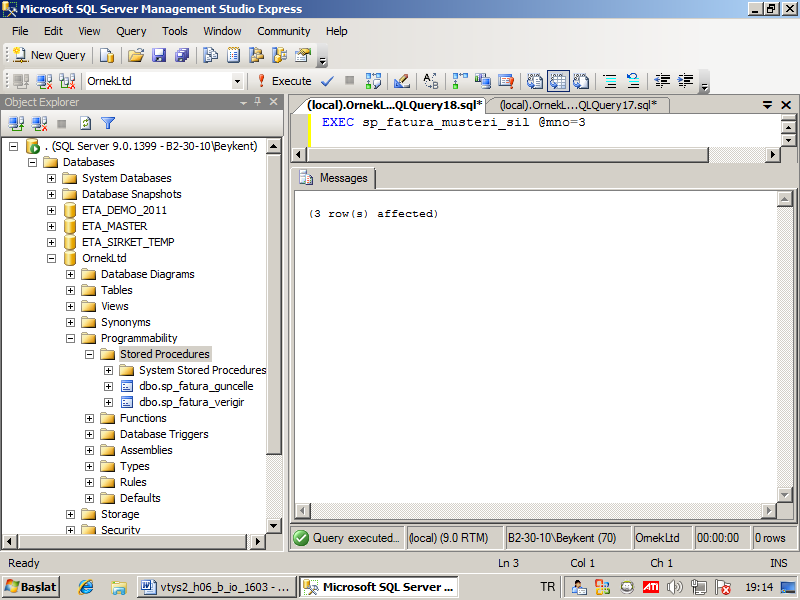
AS

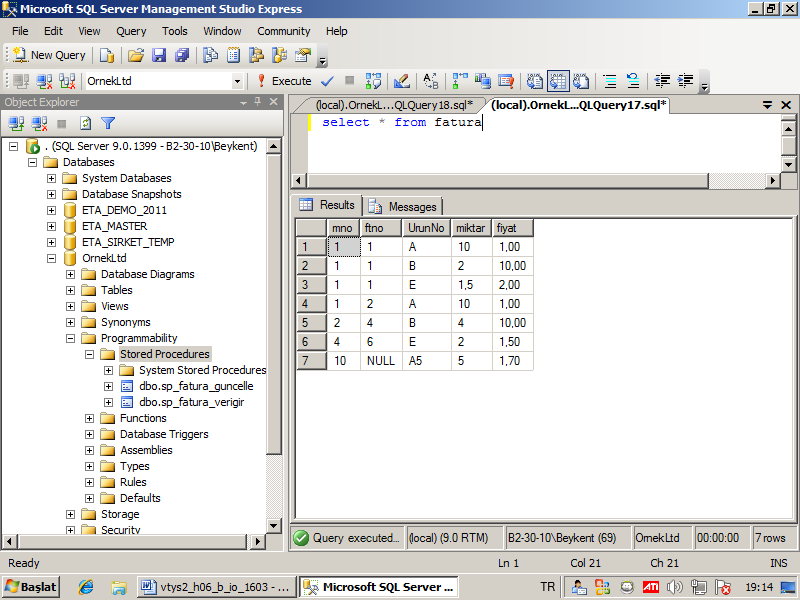
DELETE FROM fatura WHERE mno=@mno

Go



EXEC sp\_fatura\_musteri\_sil @mno=3





vtys2\_h06\_2010-11

Bilgisayarım adres çubuğundan aşağıdaki linki yazınız.

[\\B2-30-10\kenan](file:///\\B2-30-10\kenan)