BUGÜN 3 KONU VAR
1-Tabloya veri girme
2-Tablo tipi değişkenler
3-Kullanıcı tanımlı fonksiyonlar

TABLOYA VERİ GİRME
3 farklı şekilde tabloya veri girilebilir.
1- insert deyimi
2- insert into deyimi
3- select into deyimi

1.metot : insert deyimi

INSERT tabloAdi(Kolon isimleri)
VALUES (değerler)

INSERT Personel(ad, soyad) VALUES
('Kenan', 'KILIÇASLAN')
Personel tablosuna veri eklemiş oluyoruz.

2.Metot: insert into deyimi
Bu metot var olan(mevcut) tabloya, başka bir tablodan veri aktarmak için kullanılır.

Tablodaki verinin kopyalanıp, başka bir tabloya aktarılma gibi bir işlem.

Örnek:
Personel tablosundaki 10 nolu departmandakileri ad ve soyadlarını test isimli tabloya aktaralım.

Çözüm
INSERT INTO test(ad, soyad) SELECT ad, soyad FROM Personel WHERE Departman=10

Kodunu çalıştırdıktan sonra aşağıdaki sorguyu çalıştıralım. Verinin eklendiğini görürüz.

SELECT * FROM test
Mavi renkli sorgunun çıktısı, kırmızı renkte yazılı olan tabloya eklenir.

Eğer sorgudaki, kolon isimleri eklenecek tablonun tüm kolonlarını içeriyorsa kırmızı

tarafa kolon isimleri yazmaya gerek yoktur.

INSERT INTO test SELECT ad, soyad FROM Personel WHERE Departman=5

3.Metot: select into deyim

Bu metod mevcut olmayan tabloya veri ekler, veri eklenirken yeni bir tablo oluşturur.

Örnek:

SELECT SicilNo,Ad,Soyad INTO test2 FROM Personel WHERE departman=12

Yukarıdaki kod, test2 isimli bir tablo oluşturur. Bu tabloda SicilNo, Ad, Soyad kolonları bulunur. Departman=12 olan verileri bu tabloya ekler.

Örnek:

Personel tablosunda 2053 nolu personeli bir kez daha ekleyen sorguyu yazınız.

Çözüm:
Tüm kolon isimleri sorguda bulunursa, aktarılacak tablonun kolon isimleri yazma zorunluluğu yok.

```
INSERT INTO  Personel SELECT *
FROM Personel WHERE SicilNo='2053'
```

Bu kodu aşağıdaki gibi de yazabilir.

```
INSERT INTO Personel(Departman,
SicilNo,ad,soyad,cinsiyet,maas) SELECT
Departman, SicilNo,ad,soyad,cinsiyet,maas
FROM Personel WHERE SicilNo='2053'
```

TABLO TİPİ DEĞİŞKENLER

Tablo gibi davranan değişkenlerdir. Tablolar gibi içinde kayıtlar bulunur. Bu değişkenlere veri eklenebilir, değiştirilebilir, silinebilir. Tablo gibi

sorgulanabilir. Tabloların tüm özelliklerine sahiptir.

DECLARE @ad varchar(20)

@ad isminde varcha(20) veritipnde bir değişken tanımlamış olurum. Bu değişkende 20 karakterlik metin bulunur.

Tablo tipi değişken tanımı
DECLARE @isim TABLE (kolon tanımları)

Örnek
DECLARE @personel TABLE (ad varchar(20), soyad varchar(20), maas Money, giristarihi datetime)

@personel tablo yapısında bir değişkendir. Bu değişkenin ad, soyad, maas, giritarihi isimli kolonları vardır.

Bu değişkene insert ve insert into deyimleri veri eklenebilir.

Örnek:
Tüm personel tablosunun içeriğinin bulunduğu bir değişken tanımlayınız ve bu değişkeni sorgulayınız.
Çözüm

```
-- değişken tanımı
DECLARE @Personel TABLE (SicilNo
char(10), ad varchar(20), soyad
varchar(20), maas Money, cinsiyet tinyint)
--@personel değişkenine değerleri aktar
--@personel tablo gibi davrandığı için
buna veri ekleyelim.
INSERT INTO @Personel SELECT
SicilNo, ad, soyad, maas, cinsiyet FROM
Personel
--@personel değişkenini sorgulayalım
SELECT * FROM @Personel
```

Örnek:

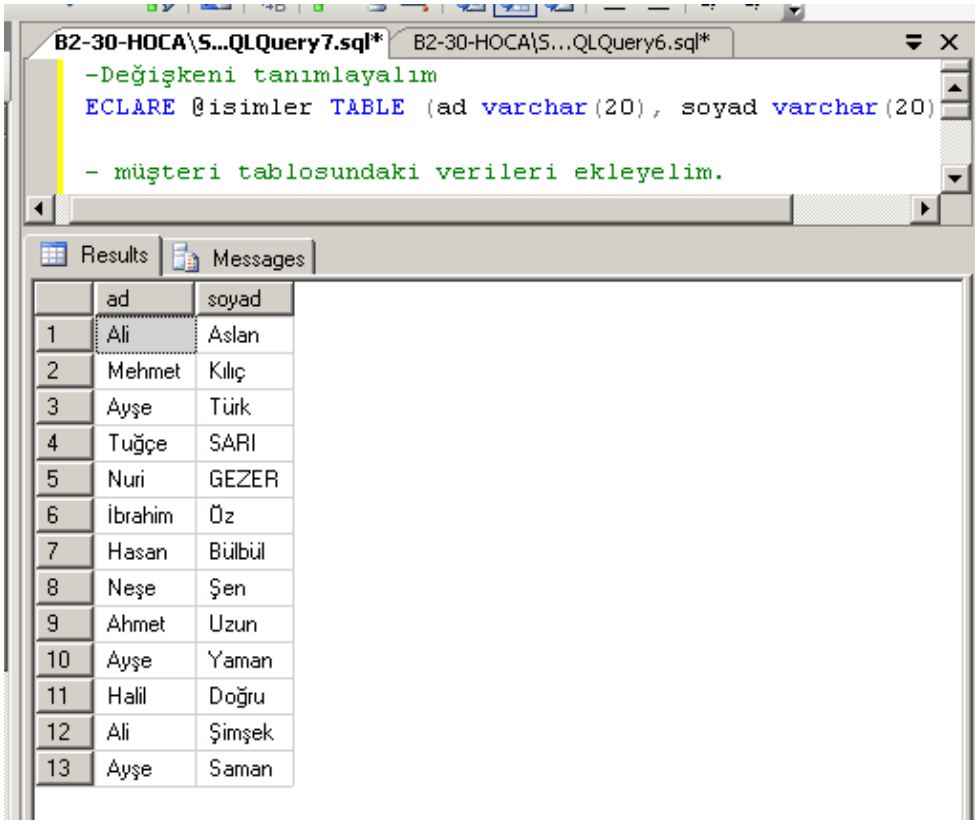
Ad ve soyad kolonları olan bir tablo tipi değişken tanımlayınız. Bu değişkene, musteri ve personel tablosundaki verileri ekleyiniz ve bu değişkeni sorgulayınız.

```
--Değişkeni tanımlayalım
DECLARE @isimler TABLE (ad
varchar(20), soyad varchar(20))

-- müşteri tablosundaki verileri ekleyelim.
INSERT INTO @isimler SELECT adi,
soyadi from musteri
-- Personel tablosundaki verileri ekleyelim
INSERT INTO @isimler SELECT ad,
soyad FROM Personel

--@isimler  değişkenini sorgulayalım
SELECT * FROM @isimler
```

Çalıştırdığımızda aşağıdaki görüntü oluşur



KULLANICI TANIMLI FONKSİYONLAR

Kullanıcı tanımlı fonksiyonlar SQL2000 ile gelen özelliklerden biridir. Fonksiyonlar tek bir değer veya tablo döndürmek için kullanılır.

Fonksiyonları veritabanı programlamada nereye koyacağımızı anlamak biraz zor olabilir.

Bu amaçla fonksiyonları, view ve stored procedure ile karşılaştıralım.
* dışarıdan parametre alan bir view tanımlanamaz. Bu türden ihtiyaçlar için fonksiyon kullanılır.
* stored procedürleri bir sorgunun parçası olarak kullanamayız. Yani stored prosedürleri sorgulayamayız. Ancak view ve fonksiyonlar sorgulanabilir.
* bir tek select ifadesi ile view oluşturamadığımız zaman fonksiyonlar kullanılır.
* sql server'da tanımlı olmayan REPLACE() gibi fonksiyonları tanımlamak için fonksiyonlar kullanılır.

3 çeşit fonksiyon vardır.
1- Skaler tanımlı (tek değer döndüren) kullanıcı tanımlı fonksiyonlar.

Skaler değerli fonksiyonlar tek bir değer döndürür. Örnek getdate() fonksiyonu, bir skaler fonksiyondur., çünkü tek bir değer döndürür.
Bazı durumlarda buna benzer fonksiyonlara ihtiyaç duyulur. Örneğin,bir müşterinin sepetinde kaç ürünün bulunduğu, kaç sipariş verdiğini döndüren kullanıcı tanımlı fonksiyonlar tanımlanabilir.

Genel ifade

```
CREATE FUNCTION
fonksiyonAdi(varsaParametre tanımı)
RETURNS  geriDonustipi
AS
BEGIN
    Sql deyimleri
    RETURN geriDonusDeğeri
END
```

Topla fonksiyonu

```
CREATE FUNCTION topla (a int, b int)
RETURNS int
BEGIN
    RETURN (a+b)
END
```

Örnek:
Bir müşteri no girildiğinde, bu müşterinin sepetinde kaç ürün olduğunu bulan bir kullanıcı tanımlı fonksiyon yazınız ve fonksiyonu çalıştırınız.
Çözüm:

```
CREATE FUNCTION sepetUrunSayi
(@mno int)
RETURNS float
BEGIN
    RETURN ( select sum(miktar) from
sepet where mno=@mno )
END
```

Çalıştırma.
Not: skaler fonksiyonları çalıştırmak için dbo. şema ismi ile çalıştırmalıyız.

SELECT dbo.sepetUrunSayi(4)

Örnek:
Müşteri tablosunu, ürün sayıları ile birlikte listeyen sorguyu yazınız (sepetUrunSayi fonksiyonu kullanılacak)

SELECT *, dbo.sepetUrunSayi(mno) as UrunSayisi FROM musteri

Örnek :
5 den fazla ürün olan müşterileri, ürün miktarları ile listeleyen sorguyu yazınız.

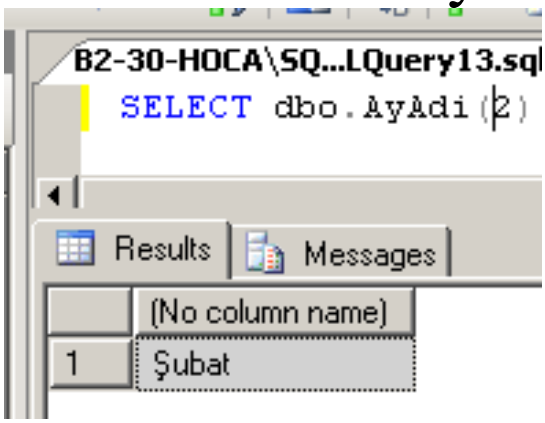
SELECT *, dbo.sepetUrunSayi(mno) as UrunSayisi FROM musteri WHERE dbo.sepetUrunSayi(mno)>5

Örnek:
1,2 gibi ay numarası gireceğiz, Ocak, Şubat gibi ay adını veren fonksiyon

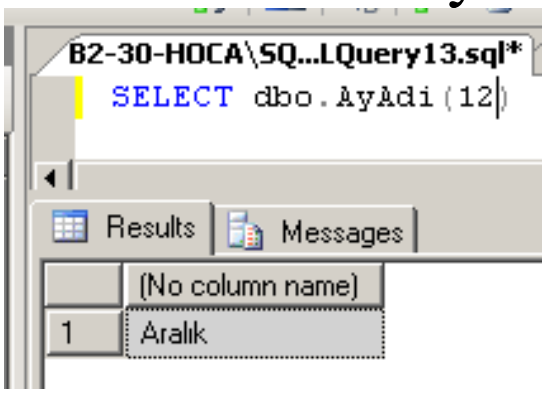
oluşturunuz . ay değeri 12 den büyük ise tanımsız değeri versin.

```
CREATE FUNCTION AyAdi(@ay
tinyint)
RETURNS varchar(20)
BEGIN
    RETURN (SELECT CASE @Ay
        WHEN 1 THEN ‘Ocak’
        WHEN 2 THEN ‘Şubat’
        WHEN 3 THEN ‘Mart’
        WHEN 4 THEN ‘Nisan’
        WHEN 5 THEN ‘Mayıs’
        WHEN 6 THEN ‘Haziran’
        WHEN 7 THEN ‘Temmuz’
        WHEN 8 THEN ‘Ağustos’
        WHEN 9 THEN ‘Eylül’
        WHEN 10 THEN ‘Ekim’
        WHEN 11 THEN ‘Kasım’
        WHEN 12 THEN ‘Aralık’
        ELSE ‘Tanımsız’
    END)
END
```

```
-- ikinci ayı öğrenelim
SELECT dbo.AyAdi(2)
```




```
-- 12.ay
SELECT dbo.AyAdi(12)
```




```
-- 13.ay
SELECT dbo.AyAdi(13)
```