Stored Procedure'ler
(Saklı Yordamlar)

Eskiden yazılımlar, sadece prosedür denilen kod parçalarından oluşurdu. Her prosedür, belli bir işlevi yerine getirmek için yazılmış kod parçalarıdır.

Mesela, 2 sayı alıp, bunların toplamını hesaplayan bir kod parçasını "toplayıcı" adında bir prosedür içinde paketleyebiliriz.

Bir prosedür, başka bir prosedür içinde çağrılabilir. Bu da sık yapılan işlemler için yazılmış kodların bir defa yazılıp, çok defa kullanılmasını, böylelikle programlamayı kolaylaştırmayı amaçlar.

Birden fazla işlemi, paketlenmiş bir halde, bir tek komut ile çalıştırmamı gerektiğinde Stored Procedure'leri kullanılır.
İşlemlerden kastedilen Transact SQL ile yapılan herşeydir.

Bir Stored Procedure oluşturulduktan sonra veritabanı sunucusunda saklanır. Her ihtiyaç duyulduğunda defalarca çağrılabilir.

Stored Procedure'le network bazlı çalışmalarda ağ trafiğini azaltır. Sistem performansını arttırır.

Neler yapılabilir? Yapısı nasıldır?

örnek c++ prosedürü

```
void a (int b, int c)
{
}
```

a fonksiyonun adı,
b ve c de parametre

- parametreli olabilir
- parametresiz olabilir
- istendiğinde parametreden veri geri dönebilir.

- İstendiğinde parametreye varsayılan değer atanabilir.
- Liste çıktısı alınabilir.
- Her türlü sql deyimi çalıştırılabilir.
- İçinde başka bir prosedür çalıştırılabilir.
- Sunucu tarafında çalışılması itenen uygulamalar yapılabilir.

Genel oluşturma ifadesi

CREATE : oluşturma
ALTER : yapı değiştirme
DROP : silme

```
CREATE PROCEDURE prosedürAdı
varsaParametreler
AS
BEGIN
…..
SQL deyimleri (çalıştırmak istediğimiz)
….
```

END

Parametre yazım şekilleri

Parametre isimlerinin başında, değişkenler gibi @ sembolü vardır. Aynen değişkenler gibi tanımlanır.

@parametreAdi veritipi, @parametre2 veritipi, ….

Örnek parametre
@ad varchar(20), @soyad varchar(20), @para money

Eğer parametrede varsayılan değer var ise varsayılanlı olarak aşağıdaki gibi yazılır

@parametre veritipi = varsayılanDeğer

Örnek:
@ad varchar(20) = 'Kenan'

Prosedürü çalıştırırken eğer @ad için bir değer girmez isem, @ad parametresinin değeri 'Kenan' dır.

Eğer parametreden veri geri dönmesi istenirse OUTPUT kelimesi kullanılır.

@Basarili int OUTPUT

@Basarili parametresi ile amacımız, veri göndermek değil, veri almak yada öğrenmek içindir. Prosedürden bilgi almak istiyoruz.

Örnek:
Tablo : Hesaplar

HesapNo	Adi	Soyadi	Bakiye
2053	İbrahim	Öz	12500
6804	Hasan	Bülbül	5000
2543	Neşe	Şen	4150
0415	Ahmet	Uzun	-1000
2749	Ayşe	Yaman	14000
0324	Halil	Doğru	12500

Soru: Hesaplar arası para transferi yapan bir prosedür oluşturunuz. Ancak bakiyenin sıfırın atına düşmesine izin vermesin.
Çözümü
Kullanacağımız parametreler,
AlacaklıHesapNo,
BorçluHesapNo,
ParaMiktarı

Prosedür adı : spParaTransfer

```
CREATE PROCEDURE spParaTransafer
@AlacakliHesapNo char(10),
@BorcluHesapNo char(10),
@ParaMiktari money
AS
BEGIN
….. SQL deyimleri
END
```

Procedür içinde çalışacak sql kodlarını yazalım.

Borçlu hesabın parasının sıfırın altına düşüp düşmediğinin belirlenmesi . Bunu için IF deyimi içine bir alt sorgu yazacağım.

```
IF (SELECT Bakiye FROM Hesaplar
WHERE HesapNo=@BorcluHesapNo)  >=
@ParaMiktari
BEGIN
      …..  Para Tranferini gerçekleştir.
END
ELSE
  … Para transferinin olmadığını bildir.
```

Mavi renkli olan alt sorgu, @BorcluHesapNo numarasının bakiyesini verir. Bu bakiyenin @ParaMiktari rakamından büyük olması gerekir.

Tümünü tek bir kod altında toplayalım.

```
CREATE PROCEDURE spParaTranfer
@AlacakliHesapNo char(10),
@BorcluHesapNo char(10),
@ParaMiktari money
AS
BEGIN

IF (SELECT Bakiye FROM Hesaplar
WHERE HesapNo=@BorcluHesapNo)  >=
@ParaMiktari
BEGIN
     UPDATE Hesaplar SET Bakiye=
Bakiye+@ParaMiktari WHERE
HesapNo=@AlacakliHesapNo
     UPDATE Hesaplar SET Bakiye=
Bakiye - @ParaMiktari WHERE
HesapNo=@BorcluHesapNo

END
ELSE
  PRINT 'Transfer Yapılamadı'

END
```