



Veri Tabanı Yönetim Sistemleri

Bölüm 9 – Fonksiyon ve Saklı Yordam (Stored Procedure)

Dr. Öğr. Üyesi Elvan DUMAN

Bölüm 9

Fonksiyon ve Saklı Yordam (Stored Procedure)

Bölüm 9 - Amaçlar

- ▶ Fonksiyonlar ve Kullanımı
- ▶ Saklı Yordam ve Kullanımı



Fonksiyon (Function)



- ▶ Fonksiyon, bir veya daha fazla giriş (parametre) alıp tek bir değer döndüren, küçük ve bağımsız çalışan SQL yapılarıdır.
- ▶ Genellikle bir hesaplama, kontrol veya veri işleme işlevi yapar.
- ▶ Fonksiyonlar SELECT, WHERE, ORDER BY gibi SQL ifadeleri içinde doğrudan çağrılabilir.

Fonksiyon Özellikleri



Özellik	Açıklama
Girdi alır mı?	Evet (parametreler)
Çıktı döndürür mü?	Evet (tek bir değer döndürür)
SQL içinde çağrılır mı?	Evet, SELECT, WHERE gibi sorgularda kullanılabilir
Ne zaman çalışır?	Sorgu sırasında
Nerede tanımlanır?	Veritabanında saklanır
Kod tekrarı azaltır mı?	Evet

Fonksiyon Temel Yapısı



```
DELIMITER $$  
CREATE FUNCTION fonksiyon_adi(parametreler)  
RETURNS veri_tipi  
DETERMINISTIC  
BEGIN  
    RETURN hesaplama;  
END $$  
DELIMITER ;
```

CREATE FUNCTION: Yeni bir fonksiyon oluşturur.

parametreler: Girdi değerleri

RETURNS: Dönecek veri tipidir

DETERMINISTIC: Aynı giriş için her zaman aynı sonucu üretir.

RETURN: Dönecek değerdir.

Fonksiyon Örneği

İki Sayının Toplamını bulma



```
DELIMITER $$  
CREATE FUNCTION toplam(sayi1 INT, sayi2 INT)  
RETURNS INT  
DETERMINISTIC  
BEGIN  
    RETURN sayi1 + sayi2;  
END $$  
DELIMITER ;  
  
SELECT toplam(5, 8);
```

Fonksiyon Örneği

Vize ve Final Notu ile Ortalama Hesaplama



```
DELIMITER $$  
CREATE FUNCTION ortalama_hesapla(vize INT,  
final INT)  
RETURNS DECIMAL(5,2)  
DETERMINISTIC  
BEGIN  
    RETURN (vize * 0.4) + (final * 0.6);  
END $$  
DELIMITER ;
```

```
SELECT ad, vize, final, ortalama_hesapla(vize, final) AS ortalama  
FROM ogrenci_notlari;
```

Fonksiyon Örneği

VARCHAR Döndüren Fonksiyon



```
DELIMITER $$
CREATE FUNCTION not_durumu(ortalama
DECIMAL(5,2))
RETURNS VARCHAR(20)
DETERMINISTIC
BEGIN
    RETURN CASE
        WHEN ortalama >= 60 THEN 'Başarılı'
        ELSE 'Başarısız'
    END;
END $$
DELIMITER ;
```

Fonksiyon Kullanım Alanları



- ▶ Matematiksel işlemler (ortalama, toplam, fark)
- ▶ Tarih işlemleri (gün ekleme, hafta bulma)
- ▶ Karar verme (örneğin, bir puanın harf notunu belirlemek)
- ▶ Metin işlemleri (string birleştirme, kısaltma)
- ▶ Tekrarlayan hesaplamaları merkezi hale getirme

Saklı Yordam (Stored Procedure)



- ▶ Saklı Yordam ingilizce adıyla Stored Procedure önceden tanımlanmış ve saklanmış bir SQL komutları bloğudur.
 - ▶ Tek bir komutla çağrılarak çalıştırılır.
 - ▶ Tekrar tekrar kullanılan SQL işlemlerinin kolay, güvenli ve düzenli şekilde yönetilmesini sağlar.
 - ▶ "Saklı" ismi sunucuda depolanması ve doğrudan ismiyle çağrılmasından gelmektedir.
- Programla dilindeki hazır fonksiyonlar (built-in (internal) functions)

Saklı Yordam Kullanım Alanları



- ▶ Tekrar eden işlemleri otomatikleştirir. Örneğin, her ayın sonunda rapor oluşturmak, istatistik çıkarmak gibi.
- ▶ Veri Tabanı mantığını merkezi hale getirir. İş mantığını uygulama kodundan ayırır.
- ▶ Veri Güvenliğini Artırır. Uygulama kullanıcılarına doğrudan tabloya erişim vermek yerine prosedür aracılığıyla işlem yaptırır.
- ▶ Performans Artışı Sağlar: Sunucu tarafında çalışır, istemci-sunucu iletişim yükünü azaltır.

Yordamların Temel Yapısı



Yordam tanımlaması

```
DELIMITER $$  
  
CREATE PROCEDURE prosedur_adi (parametreler)  
BEGIN  
    -- SQL komutları  
END $$  
  
DELIMITER ;
```

Delimiter ';' kullanımlarının karışmaması için tanımlanır.

BEGIN...END bloğu içinde birden çok sql ifadesi kullanılabilir.

Yordam Parametreleri



Yordamların alabileceği üç tür parametre mevcuttur.

Türü	Açıklama
IN	Dışarıdan değer alır (sadece okuma)
OUT	İşlem sonucu dışarıya değer gönderir
INOUT	Hem değer alır, hem de dışarıya değer gönderir.

Ad ve soyad bilgisiyle ogrenci tablosuna veri ekleyen bir yordam örneği

```
CREATE PROCEDURE ogrenci_ekle(IN ad VARCHAR(50), IN soyad  
VARCHAR(50))  
BEGIN  
    INSERT INTO ogrenci (ad, soyad) VALUES (ad, soyad);  
END;
```

Yordam Çağırma (CALL)



Bir yordamı çağırmak için CALL ifadesi kullanılır.

```
CALL SakliYordam();
```

```
CALL SakliYordam(parametre1, parametre2, parametreN);
```

```
CALL ogrenci_ekle('Gaye', 'Yılmaz');
```

Yordam Örneği



```
DELIMITER $$
CREATE PROCEDURE kitap_odunc_ver(
    IN p_kullanici_id INT,
    IN p_kitap_id INT
)
BEGIN
    INSERT INTO odunc (kullanici_id, kitap_id, odunc_tarihi,
teslim_tarihi)
    VALUES (
        p_kullanici_id,
        p_kitap_id,
        CURDATE(),
        DATE_ADD(CURDATE(), INTERVAL 1 MONTH)
    );
END $$
DELIMITER ;

CALL kitap_odunc_ver(1, 3);
```

Yordam Örneği

Out Kullanım Örneği



```
DELIMITER $$  
CREATE PROCEDURE kullanıcı_odunc_sayisi(  
    IN p_kullanici_id INT,  
    OUT p_odunc_sayisi INT  
)  
BEGIN  
    SELECT COUNT(*) INTO p_odunc_sayisi  
    FROM odunc  
    WHERE kullanıcı_id = p_kullanici_id;  
END $$  
DELIMITER ;
```

```
SET @sayac = 0;  
CALL kullanıcı_odunc_sayisi(5, @sayac);  
SELECT @sayac;
```

Yordam Örneği

INOUT Kullanım Örneği



```
DELIMITER $$  
CREATE PROCEDURE kullanıcı_odunc_arttir(  
    INOUT p_odunc_sayisi INT  
)  
BEGIN  
    SET p_odunc_sayisi = p_odunc_sayisi + 5;  
END $$  
DELIMITER ;
```

```
SET @odunc = 3;  
CALL kullanıcı_odunc_arttir(@odunc);  
SELECT @sayac;
```

IN, OUT, INOUT Parametrelerinin Farkları



Özellik	IN	OUT	INOUT
Kullanım amacı	Procedure'e veri göndermek	Procedure'den veri almak	Veri gönderip, işlem yapıp geri almak
Başlangıçta değer verir mi?	Evet	Hayır	Evet
Procedure içinde değiştirilebilir mi?	Evet	Evet	Evet
Procedure dışında değişikliği görür mü?	Hayır	Evet	Evet
Değeri procedure değiştirebilir mi?	Evet	Evet	Evet
Tipik kullanım örneği	Kayıt eklemek, veri işlemek	Hesaplama sonucu döndürmek	İşlem yapıp yeni değer döndürmek

Fonksiyon ve Saklı Yordam Farkı



Özellik	FUNCTION	PROCEDURE
Değer döndürür mü?	Evet (tek bir değer)	Hayır (işlem yapar veya OUT parametre kullanır)
SQL sorgusu içinde kullanılır mı?	Evet (SELECT, WHERE)	Hayır (CALL ile çalışır)
Bağımsız çalışabilir mi?	Hayır (sorguya gömülü çalışır)	Evet (CALL ile çalışır)
Ne için kullanılır?	Hesaplama, veri üretme	İş akışı, toplu işlem yapma
Dönen sonuç	Tek bir değer	Return yapmaz ancak out ile birden çok değeri yordam dışına taşıyabilir.

Uygulama Örnekleri



- Bir kitabın ödünç alınma sayısını döndüren bir OUT parametrelili procedure yazın ve çağırarak bir kitap için sonucu gösterin.

Uygulama Örnekleri



- Kullanıcıları ve ödünç aldığı kitap sayılarını azalan sırayla listeleyiniz.

```
SELECT K.ad, COUNT(O.id) AS odunc_sayisi  
FROM odunc AS O  
INNER JOIN kullanici AS K ON K.id=O.kullanici_id  
GROUP BY O.kullanici_id  
ORDER BY odunc_sayisi DESC;
```

Uygulama Örnekleri



- Ödünç tarihi '2025-04-01' ile '2025-04-15' arasında olan kitapların adını ve ödünç alan kullanıcı adını listeleyin.

Uygulama Örnekleri



- ▶ Hiç ödünç alınmamış kitapların adlarını bulun.

- ▶ SELECT *

- ▶ FROM kitap AS K

- ▶ WHERE id NOT IN (SELECT DISTINCT kitap_id FROM odunc)

- ▶ SELECT *

- ▶ FROM kitap AS K

- ▶ WHERE NOT EXISTS (SELECT * FROM odunc AS O WHERE O.kitap_id=K.id);

Uygulama Örnekleri



- Aynı kitabı birden fazla kullanıcı tarafından ödünç alınan kitapların adlarını listeleyin.

Bölüm sonu- Teşekkürler.

Dr. Öğr. Üyesi Elvan DUMAN – eduman@mehmetakif.edu.tr