Veri Tabanı Yönetim Sistemleri
Bölüm - 7

İçerik

- Alt Sorgular
- Çoklu Tablolar (Tabloların Birleştirilmesi)
- Görünümler

Alt Sorgular

Uygulamada, bir sorgudan elde edilen sonuç, bir diğer sorguyu ilgilendirebilir. Bu gibi durumlarda alt sorgular ya da bir başka deyişle iç sorgular kullanılır.

Alt sorgu, SELECT deyimi içerisinde ikinci bir SELECT deyiminin kullanılması ile oluşturulur. İkinci select deyimi parantez içinde yer almalıdır.

Alt sorgunun kullanım şekli aşağıda yer almaktadır:

```
SELECT liste
FROM tablo
WHERE ifade işleç
    (SELECT liste
     FROM tablo);
```

Örnek Alt Sorgu

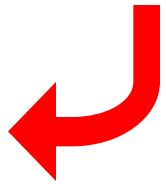
adsoyad	notu	sinif
Orçun Madran	70	4
Ahmet Kemali	80	3
Osman Musa	60	2
Hatice Kurt	90	1
Veli Faydalı	90	1
Ayşe Şahmeran	67	1
Mustafa Atlı	56	2
Zehra Kamuran	34	1
Oktay Bağcı	69	4
Demir Bülbül	24	4
Fatma Yerlisu	38	3
Hulusi Ay	86	3
Mahmut Şahin	100	4
Duygu Mutlu	90	4

Örnek tablomuzda (veritest) yer alan öğrencilerden notu en yüksek olan öğrenci ya da öğrencileri döndürmek istiyoruz

SQL cümleciği aşağıdaki gibi olmalıdır.

```
SELECT adsoyad
FROM veritest
WHERE notu = (SELECT MAX(notu)
FROM veritest)
```

adsoyad
Mahmut Şahin



Çoklu Tablolar (Tabloların Birleştirilmesi)

İlişkisel veri tabanı yönetim sistemlerinde bazen ilişkili veriler birden fazla tabloya bölünebilir ve bu tablolarda bulunan verilere aynı anda ihtiyacımız olabilir. Bu gibi durumlarda ilgili tabloları birleştirerek sorgulama yapmak gerekir.

Birden fazla tablonun ele alınarak, birleştirilmesi ve tek bir sonucun üretilmesi söz konusu olabilir. Bu işleme **"birleştirme"** denir.

Birleştirme işlemi, tabloların aynı değerler içeren sütunları kullanılmak suretiyle yapılır. Birleştirme işlemi şu şekilde tanımlanır;

```
SELECT tablo1.sütun1, tablo2.sütun2
FROM tablo1, tablo2
WHERE tablo1.sütun1 = tablo2.sütun2
```

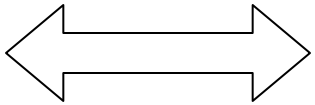
ması yararlı olacaktır. Özellikle her iki tablodaki sütun isimleri aynı ise, söz konusu sütunların hangi tablodan geldiğini belirtmek mümkün olamayacak ve bu durumda **SELECT** deyimi çalışmayacaktır.

Örnek Uygulama

Örnek uygulamamızda iki farklı tablomuz bulunmakta. “ogrenciler” isimli tabloda öğrenci bilgileri, “danismanlar” isimli tabloda danışman bilgileri yer almaktadır.

Her iki tabloda da birleştirme işleminde kullanılabilecek bir sütuna ihtiyaç duyulmaktadır. Bu sütun “sinif” sütunudur.

adsoyad	notu	sinif
Orçun Madran	70	4
Ahmet Kemali	80	3
Osman Musa	60	2
Hatice Kurt	90	1
Veli Faydalı	90	1
Ayşe Şahmeran	67	1
Mustafa Atlı	56	2
Zehra Kamuran	34	1



sinif	kimlik
1	Hakan Ars
2	Savaş Özbek
3	Bora Ünal
4	Tolga Ergin

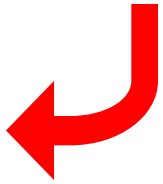
Örnek Tablo Birleştirme

Öğrencilerin danışmanlarının kimler olduğunu döndürecek bir sorguya ihtiyacımız var.

SQL cümleciği aşağıdaki gibi olmalıdır:

```
SELECT OGR.adsoyad AS Öğrenci, DAN.kimlik AS Danışman
FROM ogrenciler OGR, danismanlar DAN
WHERE OGR.sinif = DAN.sinif
```

Öğrenci	Danışman
Orçun Madran	Tolga Ergin
Ahmet Kemali	Bora Ünal
Osman Musa	Savaş Özbek
Hatice Kurt	Hakan Ars
Veli Faydalı	Hakan Ars
Ayşe Şahmeran	Hakan Ars
Mustafa Atlı	Savaş Özbek
Zehra Kamuran	Hakan Ars



Görünümler

Bir ya da daha fazla tablonun mantıksal alt kümelerini oluşturmak için **görünümlerden** yararlanılır.

Görünümler, bir tabloya dayalı mantıksal bir tablo olarak değerlendirilir.

Görünüm, tablolar gibi veriyi fiziksel olarak saklamaz. Görünümler, **saklanmış (depolanmış) SELECT** deyimi olarak değerlendirilir. Bir **SELECT** deyiminin defalarca kullanılması söz konusu ise, onu bir görünüm biçiminde tanımlayarak, bu görünümün çalıştırılması mümkündür.

Görünümler aşağıda sıralanan nedenlerle tercih edilir:

- Görünümler, veri tabanına erişimi sınırlayan olanaklardır çünkü görünüm, tabloların sadece seçilen bir kısmını görüntüleyebilir.
- Karmaşık sorguların kolayca yapılmasını sağlar.
- Aynı veriyi kullanan çok sayıda görünüm tanımlanabilmektedir.

Görünüm Komutları

Bir görünümün yaratılabilmesi için **CREATE VIEW** deyimi kullanılır. Bu deyimin kullanımı aşağıda gösterildiği gibidir:

CREATE VIEW *görünüm*
AS *altsorgu*;

Bir görünümün yaratılması esnasında kullanılacak alt sorgu içinde **ORDER BY** sözcüğü yer alamaz.

Var olan bir görünümü yok etmek amacıyla **DROP VIEW** deyimi kullanılır. Bu deyimin kullanımı aşağıda gösterildiği gibidir:

DROP VIEW *görünüm*;

Var olan bir görünümün güncelleştirilmesi amacıyla, **CREATE OR REPLACE VIEW** deyimi kullanılır. Bu deyimin kullanımı aşağıda gösterildiği gibidir:

CREATE OR RELACE VIEW *görünüm* [(*görünüm sütunları*)]
AS *altsorgu*;

Görünüm Oluşturma

Öğrencilerin danışmanlarının kimler olduğunu döndürecek bir sorguya ihtiyacımız var. Bu sorguya sık sık ihtiyaç duyacağımızı varsayarak SELECT deyimini bir görünüme dönüştürmek istiyoruz.

SQL cümleciği aşağıdaki gibi olmalıdır:

```
CREATE VIEW ogda AS
SELECT OGR.adsoyad AS Öğrenci, DAN.kimlik AS Danışman
FROM ogrenciler OGR, danismanlar DAN
WHERE OGR.sinif = DAN.sinif;
```

Görünümü Güncelleştirme

Yarattığımız “ogda” sorgusu içerisinde bir güncelleştirmeye ihtiyacımız var. Sorgu sonrası dönen listenin öğrencinin adına göre alfabetik sıralanmasını istiyoruz.

SQL cümleciği aşağıdaki gibi olmalıdır:

```
CREATE OR REPLACE VIEW ogda AS
SELECT OGR.adsoyad AS Öğrenci, DAN.kimlik AS Danışman
FROM ogrenciler OGR, danismanlar DAN
WHERE OGR.sinif = DAN.sinif
ORDER BY OGR.adsoyad
```

Görünümü Kaldırma

“ogda” isimli görümünü veritabanımızda daha fazla kullanmak istemiyoruz.

SQL cümleciği aşağıdaki gibi olmalıdır:

```
DROP VIEW ogda;
```