T.C.
MİLLÎ EĞİTİM BAKANLIĞI

BİLİŞİM TEKNOLOJİLERİ

VERİ TABANINDA SORGULAR

481BB0037

Ankara, 2012

- Bu modül, mesleki ve teknik eğitim okul/kurumlarında uygulanan Çerçeve Öğretim Programlarında yer alan yeterlikleri kazandırmaya yönelik olarak öğrencilere rehberlik etmek amacıyla hazırlanmış bireysel öğrenme materyalidir.
- Millî Eğitim Bakanlığınca ücretsiz olarak verilmiştir.
- PARA İLE SATILMAZ.

İÇİNDEKİLER

AÇIKLAMALAR

KOD	**481BB0037**
ALAN	**Bilişim Teknolojileri**
DAL/MESLEK	**Veri Tabanı Programcılığı, Web Programcılığı**
MODÜLÜN ADI	**Veri Tabanında Sorgular**
MODÜLÜN TANIMI	Veri tabanı üzerinde sorguların oluşturulması ve çalıştırılması ile ilgili bilgilerin verildiği öğrenme metaryalidir.
SÜRE	40/24
ÖN KOŞUL	"Veri Tabanı Hazırlama" modülünü tamamlamış olmak
YETERLİK	
MODÜLÜN AMACI	**Genel Amaç** Bu modülü başarı ile tamamlayan öğrenci; İlişkisel Veri Tabanını, SQL (Structured Query Language-Yapısal Sorgulama Dili) ve DML (Data Manipulation Language) sorgularını verimli ve düzenli çalıştırarak Veri Tabanında sorgulama yapabilecektir. **Amaçlar** **1.** Sorgu oluşturabilecek ve çeşitlerini kullanabileceksiniz. **2.** İlişkili tablolar ile sorgu hazırlayabileceksiniz. **3.** Veri işleme (DML) sorgularını kullanabileceksiniz.
EĞİTİM ÖĞRETİM ORTAMLARI VE DONANIMLARI	**Ortam:** Bilgisayar Laboratuvarı **Donanım:**Bilgisayar, internet, projeksiyon
ÖLÇME VE DEĞERLENDİRME	Modül içinde yer alan her öğrenme faaliyetinden sonra verilen ölçme araçları ile kendinizi değerlendireceksiniz. Öğretmen modül sonunda ölçme aracı (çoktan seçmeli test, doğru-yanlış testi, boşluk doldurma, eşleştirme vb.) kullanarak modül uygulamaları ile kazandığınız bilgi ve becerileri ölçerek sizi değerlendirecektir.

GİRİŞ

Sevgili Öğrenci,

Bir önceki modül olan "Veri Tabanı Hazırlama" modülünde veri tabanı yazılımı kurulumunu yaparak, veri tabanının en önemli ögesi olan tabloları, tabloların özelliklerini ve tablolarla ilgili çeşitli işlemler yapmayı öğrenmiştiniz.

Veri tabanında tablo oluşturma işleminden sonra gelen en önemli işlem ise "Sorgular" oluşturmaktır.

Bu modül ile sorgular hazırlamayı öğrenecek, veri tabanı tablolarınız üzerinde ekleme, silme sıralama gibi işlemleri yapabilecek, kısacası veri tabanı tablolarınıza istediğiniz şekilde erişim sağlayabileceksiniz.

Her modülün sonunda kendinizi değerlendirebileceğiniz ölçme ve değerlendirme soruları, öğrenmiş olduğunuz konuyu pekiştirmeniz için ise uygulama faaliyetleri ve tüm modüllerin sonunda bu öğrenme faaliyeti kapsamında neler öğrendiğinize dair kendinizi test etmenizi sağlayacak "Modül Değerlendirme" soruları bulunmaktadır. Bunları başarı ile tamamladığınız takdirde sizler artık "Veri Tabanında Sorgu" yapmayı öğrenmiş olacaksınız. Tabii ki takıldığınız bir konuda da öğretmeninize baş vurmayı ihmal etmeyiniz.

ÖĞRENME FAALİYETİ–1

AMAÇ

Sorgu oluşturabilecek ve çeşitlerini kullanabileceksiniz.

ARAŞTIRMA

- Sorgu çeşitlerinin neler olduğunu araştırınız.
- SQL komutlarının kullanılan veri tabanı hazırlama programlarına göre ne gibi değişiklikleri olduğunu araştırınız.

1. SORGULAR VE ÇEŞİTLERİ

"Veri Tabanı Hazırlama" modülünde veri tabanının temelini oluşturan tablolar ve tablolarla ilgili özellikler anlatılmıştı. Veri tabanında tablo oluşturma işleminden sonra gelen en önemli işlem ise "Sorgular" oluşturmaktır.

Tablolardaki kayıtlarda silme, ekleme, sıralama, seçme, değiştirme gibi işlemlere ihtiyaç duyulacaksa sorgular hazırlanması gerekir.

Ayrıca, yapılan bir takım işlemleri otomatikleştirmek ve verilerde yapılan değişiklikleri kaydetmeden önce gözden geçirmek istendiği zaman da sorgular oluşturulur.

Tablonun yapısına ve verileri görüntüleme yöntemine göre değişik özelliklerde sorgular hazırlanmaktadır. Seçme sorguları, parametre sorguları, çapraz sorgular, eylem sorguları ve SQL sorguları kullanılan sorgu türleridir.

- **Seçme Sorguları:** En sık kullanılan sorgu türüdür. Seçme sorguları, bilgileri "veri sayfası görünümü"nde gösteren veri tabanı nesnesi türüdür. Sorgu, verileri bir veya birden fazla tablodan, mevcut sorgulardan veya bunların her ikisinden alabilmektedir.

- **Parametre Sorguları:** Parametre sorguları, çalıştırıldığı zaman bir ölçüt girilmesini sağlayan iletişim kutusunu açan sorgulardır. Örneğin bir okuldaki personelin ocak ayında sevk aldığı günleri görmek istiyorsak, açılan iletişim kutusuna ölçüt olarak istenilen tarih aralıkları girilir ve bu tarihler arasındaki veriler listelenir.

- **Çapraz Sorgular:** Bir tablodaki bilgileri analiz etmek, karşılaştırmak ve tablonun özetini hazırlamak için kullanılan sorgu türüdür**.** Belirtilen iki alana göre istenilen işlemi (toplama, ortalama, vs.) tablo şeklinde göstermekte kullanılır.

- **Eylem Sorguları:** Tek işlemle birçok kayıtta değişiklik yapan sorgulardır. Silme sorgusu, güncelleştirme sorgusu, ekleme sorgusu ve tablo yapma sorgusu olmak üzere dört çeşit eylem sorgusu vardır.

- **SQL Sorguları:** SQL deyimlerini kullanarak veri tabanını güncelleştirme ve yönetme ile ilgili oluşturulan sorgulardır**.**

1.1. SQL Dilinin Yapısı

SQL (Structured Query Language), veri tabanındaki verileri okumak, güncellemek, yeni veri eklemek, verileri silmek vb. gibi işlemleri yapan, program yazarken kolaylıklar sağlayan, satırlarca kodun yaptığı işlemi tek bir sorguda yapabilen, yapısal bir sorgulama dilidir.

1.1.1. Sorgu İle Tablo Oluşturma

Sorgu kullanarak tablo oluşturmak için **CREATE TABLE** ifadesi kullanılır.

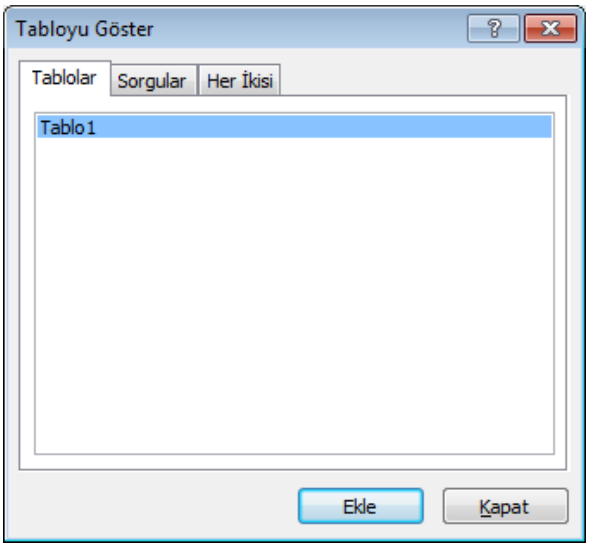
SQL kodu yazılarak tablo oluşturma aşağıdaki şekilde yapılmaktadır.

Kullanımı:

```
CREATE TABLE tablo_adı
(
Sütun1 veri tipi,
Sütun2 veri tipi,
.
.
sütunN veri tipi
)
```

Sorgu kullanarak tablo oluşturmak için öncelikle "Veri Tabanı Hazırlama" modülünde öğrendiğiniz şekilde boş bir veri tabanı oluşturulur ya da önceden hazırlamış olduğunuz veri tabanı dosyası açılır.

Veri tabanını oluşturduktan sonra **Oluştur** sekmesi altında yer alan **Diğer** grubundan Sorgu Tasarımı( **)** düğmesine tıklanır. Sorgu Tasarımı seçeneği **sorgu tasarımcısının** açılmasını sağlar ve Tabloyu Göster iletişim kutusu görüntülenir.



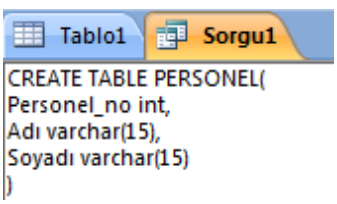
Şekil 1.1: Tabloyu göster iletişim kutusu

Tabloyu Göster İletişim kutusundan eklemek istenilen Tablo veya sorgular seçilip sırası ile Ekle ve Kapat düğmelerine tıklanır.

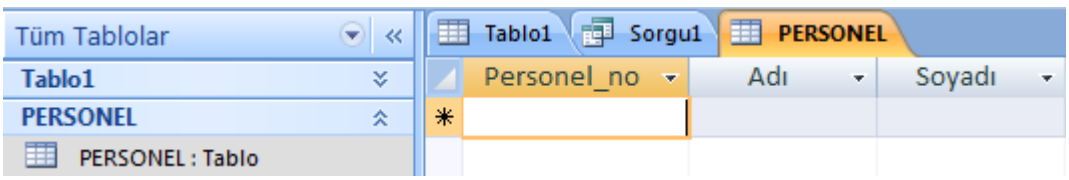
Sorgu1 sekmesi üzerinde farenin sağ tuşuna basılıp açılan menüden SQL Göster seçeneği seçilip istenilen SQL kodları yazılır.

Örnek:

Aşağıdaki SQL kodları Veri tabanı hazırlama programında yazılıp çalıştırıldığı zaman **PERSONEL** adında bir tablo oluşturulur ve bu tablonun sütunları **Personel_no, Adı, Soyadı** olarak tanımlanmış olur.



Çalıştır( ) düğmesine basıldığında aşağıdaki sonuç elde edilir.



1.1.2. Tablo Silme

Daha önceden oluşturmuş olduğunuz bir tabloya ihtiyaç duymayıp veri tabanından silmek isteyebilirsiniz. Silme işlemini gerçekleştirmek için DROP deyimi kullanılmaktadır.

Kullanımı:

DROP TABLE tablo

Örnek:

DROP TABLE PERSONEL

Yeni bir sorgu oluşturup yukarıdaki komut satırını yazıp çalıştırdığınız zaman veri tabanı programı önceden oluşturmuş olduğunuz PERSONEL isimli tabloyu silecektir.

NOT: Tabloyu silmeden önce tabloyu kapatmanız gerekmektedir.

1.1.3. Sütun Ekleme

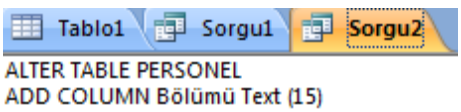
Bir tablo oluşturduktan sonra, isteğe göre oluşturulmuş olan tabloya yeni sütunlar ekleyebilirsiniz. Tabloya yeni bir sütun eklemek için **ADD COLUMN** deyimi kullanılır.

Kullanımı:

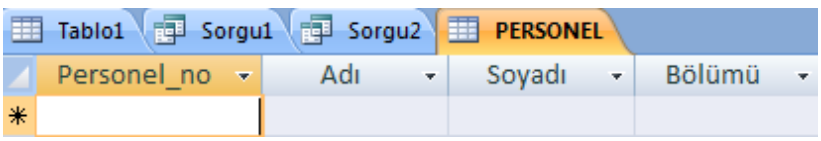
ALTER TABLE tablo_adı
ADD sütun_adı, veri_türü

Örnek:

Önceden oluşturulmuş olan **PERSONEL** isimli tabloya **"Bölümü"** sütununu eklemek istersek SQL ifadesi aşağıdaki şekilde olacaktır.



Çalıştır( ) düğmesine basıldığında aşağıdaki sonuç elde edilecektir.



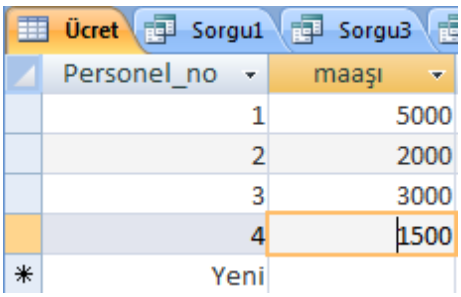
1.1.4. Tablo Güncelleme

Belirtilen tablodaki alanların değerlerini belirtilen ölçütlere göre değiştirmek için bir güncelleme sorgusu oluşturmak gerekir. Bunun için UPDATE deyimi kullanılır.

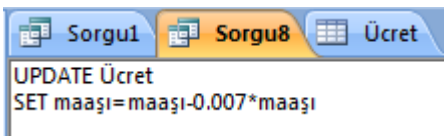
Kullanımı:

UPDATE *tablo*
SET *sütun_adı=yenideğer*
WHERE *ölçütler*;

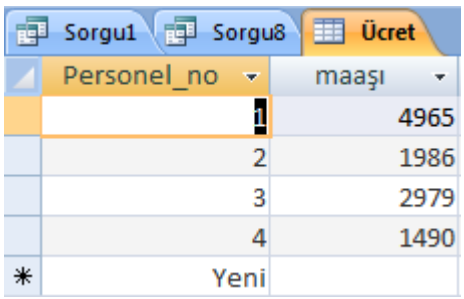
Örnek: *Ücret* tablosundaki kayıtlarda, *Maaşı* isimli alanda yer alan değerlerden % 0,07 kesinti yaparak sonucu yine aynı alan üzerinde kaydetmek istensin. Ücret tablosunun kesinti yapılmadan önceki hali aşağıdaki gibidir.



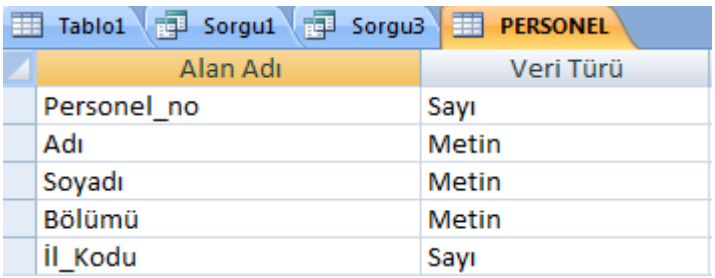
Maaş üzerinde % 0.07 kesinti yapmayı sağlayacak sorgu aşağıdaki gibidir.



Sorgu yukarıdaki gibi yazılıp çalıştır( ) düğmesine basıldığında sonuç aşağıdaki gibi olacaktır.

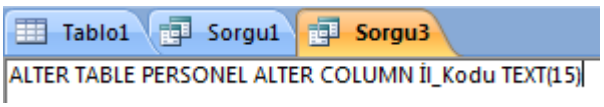


Bazı durumlarda tanımlamış olduğunuz sütunlarda değişiklik yapmak isteyebilirsiniz. Örneğin Personel tablosunda başlangıçta tam sayı olarak tanımlamış olduğunuz İl_Kodu alanının veri türünü 15 karakterden oluşan bir metin alanı olarak değiştirmek isteyebilirsiniz.

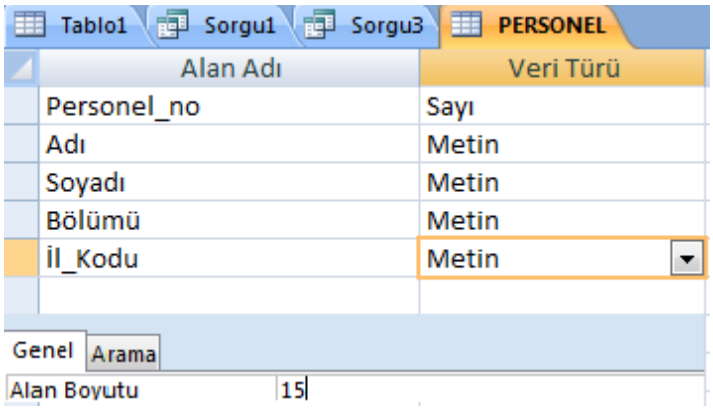


Şekil 1.2 : İl_kodu alanının veri türünün değiştirilmeden önceki hali

İl_kodu veri türünü metin veri türüne dönüştürmek için Sorgu penceresine yazılması gereken kod aşağıdaki gibi olacaktır.



Kod satırı yazılıp çalıştır( ) düğmesine basıldığı zaman aşağıdaki sonuç elde edilecektir.



Şekil 1.3: İl_kodu alanının veri türünün değiştirilmiş hali

1.1.5. SELECT Deyiminin Yapısı

Veri tabanında verilere erişebilmek için SELECT deyimi kullanılmaktadır. Select deyimi ile bir tabloda bulunan belli bir sütun, birden fazla sütun veya tüm sütunları çekebilirsiniz.

Bunun yanı sıra sorgulama işlemlerini gerçekleştirmek için de SELECT deyiminden yararlanılır.

Kullanımı:

SELECT [sütun_listesi]
FROM [tablo_listesi]

Örnek:

SELECT adı FROM PERSONEL

Bu satır ile Personel tablosunda bulunan "adı" alanını seçersiniz.

adı
ali
veli
ayşe
mehmet
*

Örnek:

SELECT * FROM PERSONEL

Bu satır ile Personel tablosunda bulunan tüm alanlar seçilmiş olur.

Personel_no	adı	Soyadı	Bölümü	İl_Kodu
1	ali	ak	matematik	34
2	veli	kara	matematik	34
30	ayşe	ay	ingilizce	35
44	mehmet	tek	müzik	07
*				

Örnek:

SELECT adı, Soyadı FROM PERSONEL

Bu satır ile Personel tablosunda bulunan adı ve soyadı alanlarını seçersiniz.

adı	Soyadı
ali	ak
veli	kara
ayşe	ay
mehmet	tek

Örnek:

SELECT adı, Soyadı, adı +" " + Soyadı FROM PERSONEL

Bu satır ile Personel tablosunda bulunan adı ve soyadı alanlarını birleştirmiş olursunuz. + operatörü sütun birleştirmek için kullanılmakla birlikte, eğer sütunlar rakamlardan oluşan veriler içeriyorlarsa bu operatör matematiksel toplama işlemi yapmış olur.

adı	Soyadı	Expr1002
ali	ak	ali ak
veli	kara	veli kara
ayşe	ay	ayşe ay
mehmet	tek	mehmet tek

1.1.5.1. Verilerin Sınırlandırılması

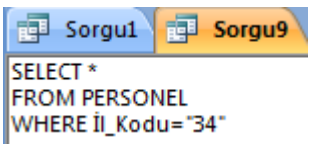
Veri tabanında veriyi alma işlemi sırasında satırlara birtakım sınırlamalar getirilerek tablonun tüm satırları yerine istenildiği kadarını elde etmek mümkündür. Tabloda belirli kısımları seçme işlemini gerçekleştirmek için WHERE sözcüğü kullanılmaktadır.

Kullanımı:

SELECT [sütunlar]
FROM [tablo]
WHERE [koşul]

Örnek:

PERSONEL tablosunda yer alan İl_kodu 34 olan kayıtları listelemek istersek komut satırı aşağıdaki şekilde olacaktır.



Yukarıdaki SQL kodunu yazıp çalıştırdığımız zaman aşağıda görüldüğü gibi İl_kodu 34 olan kayıtlar listelenecektir.

Personel_no	adı	Soyadı	Bölümü	İl_Kodu
1	ali	ak	matematik	34
2	veli	kara	matematik	34

Örnek:

Ücret tablosunda, Personel_no ve maaşı alanları görülmektedir.

Personel_no	maaşı	*Yeni alan ekleyin*
1	4965	
2	1986	
3	2979	
4	1490	

Aşağıdaki SQL deyimi, maaşları 2000 TL'den yüksek olan çalışanları seçmeye yarar.

```
SELECT Personel_No, Maaşı FROM Ücret WHERE Maaşı > 2000;
```

Bu komut satırı yazılıp çalıştırıldığında listelenen kayıtlar aşağıdaki gibi olacaktır.

Personel_No	Maaşı
1	4965
3	2979

Aşağıdaki tabloda Where komutuyla kullanılabilecek koşul operatörlerine örnekler verilmiştir.

KOŞUL	SORGU SONUCU
200	Maaşın 200 TL olduğu kayıtları döndürür.
Not 2000	Maaşın 2000 TL olmadığı kayıtları döndürür.
< 200 <= 200	Maaşın 200 TL'den az (<200) olduğu kayıtları döndürür. İkinci ifade (<=200) birim fiyatın 200 TL'ye eşit veya daha az olduğu kayıtları görüntüler.
>100 >=100	Maaşın 100TL'den fazla olduğu kayıtları döndürür. İkinci ifade maaşın 100 TL'ye eşit veya daha fazla olduğu kayıtları görüntüler.
100 veya 125	Maaşın 100 TL veya 125 TL olduğu kayıtları döndürür.
>50 ve <100 -veya- Between 50 and 100	Maaşın 50 TL ile 100 TL arasında olduğu (ancak dahil değil) kayıtları döndürür.
<50 or >100	Maaşın 50 TL ile 100 TL arasında olmadığı kayıtları döndürür.
In(10, 22, 30)	Maaşın 10 TL, 22 TL veya 25 TL olduğu kayıtları döndürür.
Like "*4,99"	Maaşın sonunda "4,99" bulunan, 4,99 TL, 14,99 TL, 24,99 TL vb. kayıtları döndürür. NOT ***** ve **%** karakterleri bir ifadede kullanıldığı zaman, herhangi bir sayıda karakteri temsil eder (bunlara joker karakter de denir). **%**karakteri ***** karakteriyle birlikte bir ifadede kullanılamadığı gibi, **?** joker karakteriyle birlikte de bir ifadede kullanılamaz. **%** joker karakterini, **_** joker karakterini de içeren bir ifadede kullanabilirsiniz.
Is Null	Maaşı alanına değer girilmeyen kayıtları döndürür.
Is Not Null	Maaşı alanında değerin eksik olmadığı kayıtları döndürür.

Tablo 1.2: Sorgu ölçütleri örnek tablosu

Örnek:

Fiyatı 50 ile 100 TL arasında olan tüm ürünleri listelemek istersek SQL kodu aşağıdaki şekilde olacaktır.

SELECT * FROM URUN WHERE FİYAT BETWEEN 50 AND 100

Örnek:

İsmi "A" ile başlayan personeller listelenmek istenirse SQL kodu aşağıdaki şekilde olacaktır.

SELECT * FROM PERSONEL WHERE ADI LIKE "A%"

1.1.5.2. Sıralama İşlemleri

Tabloların satırlarının herhangi bir sütuna göre sıralanmasının istendiği durumlarda SELECT deyimi **ORDER BY** ile birlikte kullanılır.

Örnek:

```
SELECT Soyadı, adı
FROM PERSONEL
ORDER BY Soyadı;
```

Örnek:

```
SELECT Soyadı, adı
FROM PERSONEL
ORDER BY Soyadı ASC;
```

Soyadı	adı
ak	ali
ay	ayşe
kara	veli
tek	mehmet

Yukarıdaki her iki örnek de personelin adlarını soyadlarına göre sıralamaktadır. Varsayılan sıralama artan sıralamadır. Ve her ikisi de çalıştırıldığı zaman aynı sonucu vermektedir.

Azalan şekilde sıralama yapmak için (Z'den A'ya, 9'dan 0'a), azalan şekilde sıralamak istenilen her alanın sonuna DESC sözcüğünün eklenmesi gerekir.

Örnek:

```
SELECT Soyadı, İl_Kodu
FROM PERSONEL
ORDER BY İl_Kodu DESC, Soyadı
```

Bu kodları yazıp çalıştırdığımız zaman görüntü aşağıdaki şekilde olacaktır.

Soyadı	İl_Kodu
ay	35
ak	34
kara	34
tek	07

Yukarıdaki örnekte, İl_kodları seçilip azalan şekilde sıralama yapılmaktadır.

1.1.6. SQL Fonksiyonları

SQL fonksiyonları tek satır fonksiyonları ve çoklu satır fonksiyonları (grup fonksiyonları) olmak üzere iki türlüdür. Tek satır fonksiyonları genelde sadece Fonksiyon olarak adlandırılıp, tablonun her bir satırına uygulanabilen fonksiyonlardır.

Karakter, sayısal, tarih ve dönüştürme olmak üzere farklı türleri vardır.

Karakter fonksiyonları, girdi olarak karakter verilerini alıp, karakter veya sayısal değerler döndürebilen fonksiyonlardır. Karakter fonksiyonlarından en fazla kullanılanları Tablo 1.3'te gösterilmiştir.

LCASE	Büyük harfleri küçük harflere dönüştürmek için kullanılır.
UCASE	Küçük harfleri büyük harflere dönüştürmek için kullanılır.
MID	Verinin bir parçasını çekmek için kullanılır.
LEN	Bir string ifadesinin veya bir sütundaki verinin uzunluğunu döndürmek için kullanılır.

Tablo 1.3: Karakter fonksiyonları

Sayısal fonksiyonlar, sayısal veriler üzerinde birtakım işlemleri gerçekleştirmek için kullanılan fonksiyonlardır. Sayısal fonksiyonlardan en fazla kullanılanları Tablo 1.4'te gösterilmiştir.

ROUND	Sayısal değerleri yuvarlamak için kullanılır.
MOD	İki sayısal değerin, birbirine bölümünden elde edilen kalanı döndürmek için kullanılır.

Tablo 1.4: Sayısal fonksiyonlar

Dönüştürme fonksiyonları, karakter, sayısal ve tarih verilerinin birbirine dönüştürülmesi için kullanılan fonksiyonlardır. Dönüştürme fonksiyonlardan en fazla kullanılanları Tablo 1.5'te gösterilmiştir.

CSTR	Tarih ve sayısal bilginin istenildiği gibi biçimlendirilerek karakter dizisine dönüştürülmesini sağlar.
CINT	Bir dizenin içerdiği sayısal değerleri biçimlendirilerek sayısal veri türüne dönüştürmeye yarar.
CDATE	Bir dizenin içerdiği tarih verilerinin istenildiği gibi biçimlendirilerek tarih veri türüne dönüştürmek için kullanılır.

Tablo 1.5: Dönüştürme fonksiyonları

Tarih ve saat fonksiyonları, tarih ve saat ile bilgi almak ve tarih ve saat verilerini biçimlendirmek için kullanılan fonksiyonlardır. Tarih ve saat fonksiyonlardan en fazla kullanılanları Tablo 1.6'da gösterilmiştir.

NOW	Bilgisayarınızdaki geçerli sistem tarihini ve saatini bildirmek için kullanılır. Örnek: **SELECT Now();**
DAY	1 ve 31 dahil olmak üzere bu rakamlar arasında ayın gününü temsil eden bir tam sayı belirten **Variant** (**Tamsayı**) döndürmek için kullanılır. Örnek: **SELECT Day(#11/22/2003#);** Bu kod yazılıp çalıştırıldığında ayın gününü temsil eden "22" sayısı döndürülecektir.
MONTH	Yılın ayını gösteren 1 ile 12 arasında (bu sayılar dahil) bir tam sayı belirten **Variant** (**Integer**) türünde değer döndürmeye yarar. Örnek: **SELECT Month (#22/11/2009#);** Bu kod çalıştırıldığı zaman yılın ayını gösteren "11" değeri döndürülecektir.
YEAR	Yılı gösteren bir tam sayı içeren **Variant** (**Integer**) türünde bir değer döndürmeye yarar. Verilen tarihin yılını döndürür. Örnek: **SELECT Year (#22/11/2012#);**
DATEADD	Belirtilen bir zaman aralığı eklenmiş olan bir tarihi içeren **Variant** (**Tarih**) veri türünü döndürür. Örnek: **SELECT DateAdd("yyyy",3,#11/22/2009#);** 2009 yılına 3 yıl eklenmiştir
DATEDIFF	Belirtilen iki tarih arasındaki zaman aralıklarının sayısını belirten bir **Variant** (**Long**) döndürür. Örnek: **SELECT DateDiff('m',#11/17/2011#,#1/22/2012#);** Bu kod yazılıp çalıştırıldığı zaman bu zaman dilimleri arasında iki ay fark olduğundan "2" değeri dönecektir.
DATEPART	Verilen bir tarihin belirtilen kısmını içeren bir **Variant** (**Tam sayı**) döndürür. Örnek**: SELECT DATEPART("yyyy", "01/04/2012")**

Tablo 1.6: Tarih ve saat fonksiyonları

1.2. Verileri Gruplayarak Analiz Etme

Şu ana kadar incelemiş olduğumuz deyimler söz konusu tablonun tüm satırlarına uygulanmaktaydı. Bazı durumlarda bazı işlemlerin satırlar yerine gruplara uygulanması gerekmektedir. Veriler gruplara ayrılıp analiz edilir ve bu tür gruplama işlemleri için de grup fonksiyonları kullanılır.

1.2.1. Grup Fonksiyonları

Tek satır fonksiyonları tablonun bir satırına uygulanıp buna karşılık gelen bir sonuç satırı elde ediliyordu. Bir grup satıra bir fonksiyonun uygulanmasının söz konusu olduğu durumlara; **"çoklu satır"** veya **"grup fonksiyonları"** adı verilir. Grup fonksiyonları tablonun tüm satırlarına uygulanabilmektedir.

AVG()	Bu fonksiyon, herhangi bir sütunun içerdiği sayısal değerlerin aritmetik ortalamasını hesaplamak amacıyla kullanılır. Fonksiyonun uygulandığı sütunun veri türü sayısal olmalıdır. **Örnek**: **SELECT AVG(NOTLAR) AS ORTALAMA FROM OGRENCİ**
SUM()	Sütunların içerdiği sayısal değerleri toplamak amacıyla kullanılan fonksiyondur. **Örnek**: **SELECT SUM(NOTLAR) AS TOPLAM FROM OGRENCİ**
STDEV()	Standart sapma, sayısal verilerin aritmetik ortalamalardan farklarının kareli ortalaması olarak bilinir. Bu hesaplamayı yapan SQL fonksiyonu ise; **STDEV()** 'dir. **Örnek**: **SELECT STDEV(NOTLAR) AS "St SAPMA" FROM OGRENCİ**
VARP()	Sorgunun belirtilen alanında bulunan değerler kümesiyle temsil edilen bir grubun tahmini varyansını gösterir. Varyans, sayısal değerler arasındaki değişkenliği ölçen bir kavramdır. Standart sapmanın karesi olarak bilinmektedir. **Örnek: SELECT VARP(MAAŞ) AS VARYANS FROM PERSONEL**
MAX() ve MIN()	Tablodaki değerler arasında en büyük olanı bulmak için **MAX(),** en küçük olanını bulmak içinse **MIN()** ve fonksiyonları kullanılır. **Örnek: SELECT MAX(NOLAR) AS "EN YÜKSEK" FROM ÖĞRENCİ** **Örnek: SELECT MİN(NOTLAR) AS "EN DÜŞÜK" FROM ÖĞRENCİ**
COUNT()	Bir tablodaki kayıtların sayılması amacıyla kullanılan fonksiyondur. **COUNT(*)** fonksiyonu, **NULL** değerleri de içeren tüm kayıtların sayılmasını sağlar. **WHERE** ile birlikte kullanılırsa, bu koşula uygun tüm kayıtları sayar. **COUNT(sütun)** biçiminde kullanılırsa, söz konusu sütunda NULL değerler içermeyen tüm kayıtların sayılmasına neden olur. **Örnek: SELECT COUNT(*) AS "ÖĞRENCİ SAYISI" FROM ÖĞRENCİ**

Tablo 1.7: Grup fonksiyonları ve örnekleri

1.2.2. Birden Fazla Sütuna Göre Gruplama

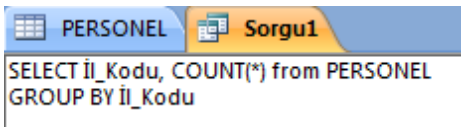
GROUP BY kullanarak belirtilen alan listesindeki benzer değerlere sahip kayıtları tek bir kayıt olarak birleştirebilirsiniz. Yani birden fazla sütun için de gruplama yapabilirsiniz.

Kullanımı:
SELECT alanlistesi
FROM tablo
WHERE ölçütler
[GROUP BY grupalanlistesi]

Örnek:

Personel_no	adı	Soyadı	Bölümü	İl_Kodu
1	ali	ak	matematik	34
2	veli	kara	matematik	34
30	ayşe	ay	ingilizce	35
44	mehmet	tek	müzik	07
8	fatma	sam	bilgisayar	34
9	yeşim	yek	resim	35

Yukarıdaki PERSONEL tablosunda her ilde yaşayan personel sayısına göre gruplama yapmak istersek SQL kodu aşağıdaki gibi olacaktır.



Bu kodu çalıştırdığımız zaman PERSONEL tablosunda yer alan personellerin yaşadıkları illere göre sayısı listelenecektir.

İl_Kodu	Expr1001
07	1
34	3
35	2

Örnek:

SELECT Count(*) AS Personelsayısı FROM PERSONEL;

Yukarıdaki komut satırı personel tablosunda yer alan toplam personel sayısını gösterir.

Personelsayısı
6

1.2.3. Grup Koşullarının Kullanımı

Grup işlemlerinin uygulanması sırasında birtakım sınırlamalar gerekebilir. Grup koşulları belirlenirken **HAVING** sözcüğü kullanılır. HAVING sözcüğü gruplama yaptıktan sonra kullanılır. Gruplandırmak istenilmeyen satırları dışarıda tutmak için WHERE, gruplandırılan kayıtlara filtre uygulamak için ise HAVING kullanılır.

Group By ifadesi Where ifadesinden sonra, Having ifadesinden önce kullanılır. Order By ifadesi ise en son kullanılır. Kayıtlar GROUP BY ile gruplandırılır ve HAVING ile de hangi kayıtların görüntüleneceği gösterilir.

Örnek:

Personel_no	maaşı
1	4965
2	1986
3	2979
4	1490

Yukarıdaki Ücret tablosunda Toplam maaşı 2500'den büyük olan grupları listelemek istiyorsak SQL kodu aşağıdaki gibi olacaktır.

PERSONEL | Ücret | Sorgu1

```
SELECT Personel_no,
Sum(maaşı)
FROM Ücret
GROUP BY Personel_no
HAVING Sum(maaşı) > 2500;
```

Sorgu çalıştırıldığı zaman görüntü aşağıdaki gibi olacaktır.

Personel_no	Expr1001
1	4965
3	2979

UYGULAMA FAALİYETİ

Aşağıda verilen işlem basamaklarını takip ederek konuyu daha da pekiştirelim.

Öneriler kısmı, uygulama faaliyeti için yönlendirici olacaktır.

İşlem Basamakları	Öneriler
➢ Veri tabanınızda "Okul" adında bir tablo oluşturunuz. ➢ Tablonun alan adları ve veri türleri aşağıdaki gibi olsun: Öğrenci_no(int); Adı varchar(15); Soyadı varchar(15); Yaşı int;	➢ CREATE TABLE komutunu kullanarak tablonuzu oluşturunuz.
➢ Tabloya 4 kişinin bilgilerini içerecek şekilde kayıt girişi yapınız.	➢ Tabloyu veri sayfası görünümünde açıp veri girişini yapınız.
➢ "Dersler" isminde yeni bir tablo oluşturunuz.	➢ CREATE TABLE komutunu kullanınız.
➢ "Dersler" tablosunu siliniz	➢ DROP TABLE Dersler
➢ "Okul" tablosuna Doğum_tarihi sütununu ekleyiniz.	➢ ADD COLUMN deyimini kullanınız.
➢ "Okul" tablosundaki tüm öğrencilerin yaşını 2 yaş artırınız.	➢ UPDATE deyiminden yararlanınız.
➢ Yaşı 15'ten küçük olan öğrencileri seçiniz.	➢ SELECT deyimini kullanınız.
➢ "Notlar" isminde yeni bir tablo oluşturunuz. Ogr_Not int veri türünde bir alan ekleyiniz. Bu tabloya 4 tane not girişi yapınız.	➢ CREATE TABLO Notlar
➢ Notu 60 ile 90 arasında olan öğrencileri listeleyiniz.	➢ SELECT ve BETWEEN kullanarak listeleyiniz.
➢ "Okul" tablosundaki kayıtları soyadı alanına göre sıralayınız.	➢ ORDER BY deyiminden yararlanınız.
➢ "Okul" tablosundaki öğrencilerin yaşları toplamını bulunuz.	➢ SUM();
➢ "Okul" tablosunda en büyük yaşın kaç olduğunu bulunuz.	➢ MAX();
➢ Öğrencileri yaşlarına göre gruplayınız.	➢ GRUP BY();

KONTROL LİSTESİ

Bu faaliyet kapsamında aşağıda listelenen davranışlardan kazandığınız beceriler için **Evet**, kazanamadığınız beceriler için **Hayır** kutucuğuna (X) işareti koyarak kendinizi değerlendiriniz.

Değerlendirme Ölçütleri	Evet	Hayır
1. Sorgular ve çeşitlerinin neler olduğunu öğrendiniz mi?		
2. Sorgu ile veri tabanı oluşturdunuz mu?		
3. Sorgu ile tablo oluşturdunuz mu?		
4. Sorgu ile oluşturmuş olduğunuz tabloyu silebildiniz mi?		
5. Sorgu kullanarak tablonuza sütun eklediniz mi?		
6. Tablo güncelleme işlemini kullandınız mı?		
7. Select deyimi kullanarak tablodaki verilere erişebildiniz mi?		
8. Satırların ve sütunların sınırlandırılmasını yaptınız mı?		
9. Order By kullanarak sıralama yaptınız mı?		
10. SQL fonksiyonlarını kullandınız mı?		
11. Grup fonksiyonlarını kullandınız mı?		
12. Tablonuzdaki verileri SQL kullanarak birden fazla sütuna göre grupladınız mı?		
13. Having sözcüğü kullanarak grup koşullarını belirleyebildiniz mi?		

DEĞERLENDİRME

Değerlendirme sonunda **"Hayır"** şeklindeki cevaplarınızı bir daha gözden geçiriniz. Kendinizi yeterli görmüyorsanız, öğrenme faaliyetini tekrar ediniz. Bütün cevaplarınız **"Evet"** ise "Ölçme ve Değerlendirme"ye geçiniz.

ÖLÇME VE DEĞERLENDİRME

Aşağıdaki cümlelerin başında boş bırakılan parantezlere, cümlelerde verilen bilgiler doğru ise D, yanlış ise Y yazınız.

1. ( ) Tabloların satırlarının herhangi bir sütuna göre sıralanmasının istendiği durumlarda SELECT deyimi **ORDER BY** ile birlikte kullanılır.

2. ( ) MID() fonksiyonu bir karakter fonksiyonudur ve küçük harfleri büyük harfe dönüştürmek için kullanılır.

3. ( ) COUNT() fonksiyonu bir tablodaki kayıtların sayılması amacıyla kullanılan fonksiyondur.

Aşağıdaki soruları dikkatlice okuyunuz ve doğru seçeneği işaretleyiniz.

4. Sorgu kullanarak tablo oluşturmak için kullanılan SQL deyimi aşağıdakilerden hangisidir?

A) CREATE TABLE
B) INSERT TABLE
C) DELETE TABLE
D) DROP TABLE

5. ADD COLUMN deyiminin görevi aşağıdakilerden hangisidir?

A) Tabloya satır eklemek için kullanılır.
B) Tabloya yeni bir ad vermek için kullanılır.
C) Tabloyu silmek için kullanılır.
D) Tabloya sütun eklemek için kullanılır.

6. ÖĞRENCİLER tablosunda yer alan ADI ve BÖLÜMÜ alanlarını seçmek için kullanılan SQL kodu aşağıdakilerden hangisidir?

A) SELECT ADI, BÖLÜMÜ FROM ÖĞRENCİ
B) INSERT ADI, BÖLÜMÜ FROM ÖĞRENCİLER
C) SELECT AD, SOYAD, BÖLÜMÜ FROM ÖĞRENCİLER
D) SELECT ADI, BÖLÜMÜ FROM ÖĞRENCİLER

7. Büyük harfleri küçük harfe dönüştürmek için kullanılan fonksiyon aşağıdakilerden hangisidir?

A) UCASE()
B) LCASE
C) MOD( )
D) CINT()

8. Herhangi bir sütunun içerdiği sayısal değerlerin aritmetik ortalamasını hesaplamak amacıyla kullanılan fonksiyon aşağıdakilerden hangisidir?

A) SUM()
B) STDEV()
C) AVG()
D) MAX()

9. **SELECT DateAdd("yyyy",2,#11/22/2010#); komut satırı hangi sonucu üretir?**

A) 2010 yılına 2 yıl ekler.
B) 11/22/2010 tarihine 2 gün ekler.
C) 11/22/2010 tarihine 2 hafta ekler.
D) Hiçbiri

DEĞERLENDİRME

Cevaplarınızı cevap anahtarıyla karşılaştırınız. Yanlış cevap verdiğiniz ya da cevap verirken tereddüt ettiğiniz sorularla ilgili konuları faaliyete geri dönerek tekrarlayınız. Cevaplarınızın tümü doğru ise bir sonraki öğrenme faaliyetine geçiniz.

ÖĞRENME FAALİYETİ–2

AMAÇ

İlişkili tablolar ile sorgu hazırlayabileceksiniz.

ARAŞTIRMA

- Hangi durumlarda tabloların birleştirilmesi işlemine ihtiyaç duyulduğunu araştırınız.

2. İLİŞKİLİ TABLOLAR

İlişkisel veri tabanları, birbirleri ile mantıksal ilişkiler içinde olan tablolardan oluşmaktadır. Tabloları ortak olarak sahip oldukları alanlarda birleştirmek için ilişkiler kullanılır. İlişki ise bir sorguda birleştirmeyle gösterilmektedir.

Bu bölümde birbirleri ile ilişkileri olan tablolar için kullanılabilecek birleştirme türlerinin neler olduğunu, bu türlerin hangi durumlarda kullanıldığını ve nasıl birleştirme oluşturulacağını öğreneceğiz.

2.1. Tabloların Birleştirilmesi

Birden fazla tablodan veri almak gerektiği durumlarda tablolar arasında ilişki kurulması gerekmektedir. Bu işleme **Join** (birleştirme) adı verilir. Join işlemi birden fazla tabloyu birbirine bağlayıp bu tablolar üzerinde işlem yapabilmemizi sağlamaktadır.

Birleştirme işlemi yapabilmek için tabloların aynı değerleri içeren sütunlarının kullanılması gerekir.

Tablo birleştirme işlemi yapılırken birleştirmek istediğiniz duruma göre, Kartezyen birleşim, eşiti olan birleştirme veya eşiti olmayan birleştirme türlerinden uygun olanını kullanabilirsiniz.

2.1.1. Kartezyen Çarpımı

İki tablo arasında birleştirme koşulunun tanımlanmadığı durumlarda Kartezyen çarpımından söz edilir. Soldaki tablonun her kaydı için, sağdaki tablodan bütün kayıtları çeker. Birleştirme koşulunun geçersiz olduğu ve birinci tablodaki tüm satırların ikinci tablodaki tüm satırlarla birleşmediği durumlarda da Kartezyen çarpım elde edilir.

Kullanımı:

SELECT sütun1, sütun2, sütun3,…, sütunN
FROM tablo1, tablo2, tablo3,…, tabloN

Örnek:

ÜRÜNLER TABLOSU:

Ürün_no	Ürün_adı	Grup_no
100	oys hazırlık	5
101	ales hazırlık	5
102	domates	10
103	kuzu eti	15

REYONLAR TABLOSU:

Grup_no	Reyon_adı
5	Kitap
10	Sebze
15	Et

Ürünler tablosundaki kayıtlar ile Reyonlar tablosundaki kayıtlara Kartezyen birleşimi uygulanmış örnek aşağıda verilmiştir.

SELECT *
FROM ÜRÜNLER, REYONLAR;

Ürünler tablosunda 4, Reyonlar tablosunda 3 kayıt bulunmaktadır. Bu iki tablonun birleşiminden 4x3=12 satırlık bir birleşim meydana gelecektir.

Şekil 2.1'de 1. Tablodaki her kayıt için 2. Tablodaki tüm kayıtlar listelenmiştir.

Ürün_no	Ürün_adı	ÜRÜNLER.Grup_no	REYONLAR.Grup_no	Reyon_adı
100	oys hazırlık	5	5	Kitap
100	oys hazırlık	5	10	Sebze
100	oys hazırlık	5	15	Et
101	ales hazırlık	5	5	Kitap
101	ales hazırlık	5	10	Sebze
101	ales hazırlık	5	15	Et
102	domates	10	5	Kitap
102	domates	10	10	Sebze
102	domates	10	15	Et
103	kuzu eti	15	5	Kitap
103	kuzu eti	15	10	Sebze
103	kuzu eti	15	15	Et

Şekil 2.1: Kartezyen çarpımı sonucu tabloların birleştirilmesi

2.1.2. Eşiti Olan Birleştirme

İç birleştirme olarak da adlandırılan birleştirme türüdür. İç birleştirme bir sorguya, birleştirilen tabloların birinde yer alan satırların, birleştirilen alanlardaki verileri temel alarak, diğer tablodaki satırlara karşılık geldiğini bildirir. İç birleştirme içeren bir sorgu çalıştırıldığında, sorgu işlemlerine yalnızca, birleştirilen tabloların her ikisinde de bulunan ortak değere sahip olan satırlar eklenir. Birleştirmede yer alan her iki tablodan sadece, birleştirme alanında eşleşen satırlar döndürülmek istenildiği zaman iç birleştirme kullanılır.

Eşiti olan birleştirme yapılırken **INNER JOIN** deyimi kullanılır.

Kullanımı:

FROM tablo1 **INNER JOIN** tablo2 **ON** tablo1.sütun1 karşılaştırma tablo2.sütun2

Birinci tablodaki tüm kayıtları çekip, bu kayıtlar ile eşleşen ikinci tablodaki kayıtlar listelenir.

Örnek:

```
SELECT *
FROM  ÜRÜNLER, REYONLAR
WHERE ÜRÜNLER.Grup_no=REYONLAR.Grup_no;
```

veya

```
SELECT *
FROM ÜRÜNLER INNER JOIN REYONLAR ON ÜRÜNLER.Grup_no=REYONLAR.Grup_no;
```

Yukarıdaki örnekteki her iki yazım türü de tabloları birleştirecek ve ortak değere sahip olan kayıtlar aşağıdaki gibi listelenecektir.

Ürün_no	Ürün_adı	ÜRÜNLER.Grup_nı	REYONLAR.Grup_no	Reyon_adı
100	oys hazırlık	5	5	Kitap
101	ales hazırlık	5	5	Kitap
102	domates	10	10	Sebze
103	kuzu eti	15	15	Et

2.1.3. Eşiti Olmayan Birleştirme

Eşiti olan birleştirme sırasında bir tablodaki bir sütunun içerdiği değerler diğer tablonun ilgili sütunu ile eşleştirilip sadece eşleşen değerler birleştiriliyordu. Eşleşmeyen satırlar ise birleştirilemiyordu.

Eşleşmeyen satırların da birleştirilip sonuca dâhil edilmesi istenilen durumlarda "Eşiti Olmayan Birleştirme" kullanılmaktadır.

Eşiti olmayan birleştirmeler (dış birleştirmeler), eşleşmeyen kayıtların hangi tabloda olduğuna bakarak sol dış birleştirme veya sağ dış birleştirme olmak üzere iki şekilde olabilmektedir.

Kullanımı:

FROM tablo1 [LEFT | RIGHT] JOIN tablo2
ON tablo1.sütun1 karşılaştırma tablo2.sütun2

Buradaki tablo1 ve tablo2 kayıtların birleştirileceği tabloların adını, sütun1 ve sütun2 birleştirilen sütunların adlarını, karşılaştırma ise ("=," "<," ">," "<=," ">=" veya "<>.") gibi işleçleri göstermektedir.

Sol dış birleştirme oluşturmak için LEFT JOIN kullanılır. Soldaki tablodan tüm kayıtlar alınır, sağdaki tabloda eşleşen kayıtlar yazılır ve eşleşmeyen kayıtlar için NULL değer döndürülür.

Sağ dış birleştirme oluşturmak için RIGHT JOIN kullanılır. Sağ dış birleşimler, ilk tablonun (soldaki tablo) kayıtlarında eşleşen değer olmasa bile, iki tablodan ikincisinin (sağdaki tablo) tüm kayıtlarını içerir.

Örneğin, Bölümler (sol) ve Personel (sağ) tablolarında bölüme atanmış personel olmasa bile tüm bölümleri seçmek için LEFT JOIN, herhangi bir bölüme atanmamış olanlar da dâhil, tüm personeli seçmek için ise RIGHT JOIN kullanılır.

Örnek 1:

ÜRÜNLER TABLOSU

Ürün_no	Ürün_adı	Grup_no
100	oys hazırlık	5
101	ales hazırlık	5
102	domates	10
103	kuzu eti	15
104	BALIK	

REYONLAR TABLOSU

Grup_no	Reyon_adı
5	Kitap
10	Sebze
15	Et
20	Çamaşır suyu

Yukarıdaki ÜRÜNLER ve REYONLAR tablosunu RIGHT JOIN kullanarak birleştirmek için aşağıdaki SQL komutu kullanılır.

```
SELECT *
FROM ÜRÜNLER  RIGHT  JOIN REYONLAR
ON ÜRÜNLER.Grup_no=REYONLAR.Grup_no;
```

Kod yazılıp çalıştırıldığı zaman görüntü aşağıdaki gibi olacaktır.

Ürün_no	Ürün_adı	ÜRÜNLER.Grup_	REYONLAR.Grup_no	Reyon_adı
101	ales hazırlık	5	5	Kitap
100	oys hazırlık	5	5	Kitap
102	domates	10	10	Sebze
103	kuzu eti	15	15	Et
			20	Çamaşır suyu

Örnek 2:

ÜRÜNLER ve REYONLAR tablosunu LEFT JOIN kullanarak birleştirmek için aşağıdaki SQL kodu yazılır.

```
SELECT *
FROM ÜRÜNLER  LEFT  JOIN REYONLAR
ON ÜRÜNLER.Grup_no=REYONLAR.Grup_no;
```

Kod yazılıp çalıştırıldığı zaman görüntü aşağıdaki gibi olacaktır.

Ürün_no	Ürün_adı	ÜRÜNLER.Grup_	REYONLAR.Grup_no	Reyon_adı
100	oys hazırlık	5	5	Kitap
101	ales hazırlık	5	5	Kitap
102	domates	10	10	Sebze
103	kuzu eti	15	15	Et
104	BALIK			

2.2. Alt Sorgular

Bazı durumlarda bir sorgudan elde edilen sonuç diğer başka bir sorgu içerisinde kullanılabilmektedir. Bu tür durumlarda iç içe sorgular oluşturulmaktadır. Kullanılan iç içe sorgularda yer alan içteki sorgulara "alt sorgular" adı verilir.

2.2.1. Alt Sorgu Düzenleme Kuralları

Alt sorgular düzenlenirken uyulması gereken birtakım kurallar bulunmaktadır. Bunlar;

- FORM sözcüğü içinde tanımlanan sorgular dışında, alt sorgu, ana sorgu içerisindeki karşılaştırma işlecinin sağ tarafında yer almalıdır.
- Alt sorgu, parantezler içerisinde yer almalıdır.
- Alt sorgu ORDER BY sözcüğünü içermemelidir. ORDER BY sadece ana sorgu içerisinde yer alabilmektedir.

2.2.2. Alt Sorgunun Tanımlanması

Alt sorgu bir SELECT, SELECT...INTO, INSERT...INTO, DELETE veya UPDATE deyimi içinde veya başka bir alt sorguda SELECT deyiminin kullanılması ile elde edilir.

Kullanımı:

SELECT liste
FROM tablo
WHERE ifade karşılaştırma
(SELECT liste
FROM tablo)

Bir alt sorgu aşağıdaki bölümlerden oluşmaktadır:

karşılaştırma [ANY | ALL | SOME] (*sqldeyimi*):Karılaştırma, ifadeyi, alt sorgunun sonuçları ile karşılaştırmaya yarayan bir karşılaştırma işlecidir.

ifade [NOT] IN (*sqldeyimi*): İfade, alt sorgu sonuç kümesinde aranan ifadeye verilen addır.

WHERE iki sorguyu birbirine bağlamak için kullanılmaktadır. ANY veya SOME, yapılan karşılaştırma sonucunda alt sorgu kayıtlarından herhangi bir tanesi ile eşleşen ana sorgu kayıtlarını almak için kullanılır. ALL, yapılan karşılaştırma sonucunda alt sorgu kayıtlarının tümüyle eşleşen ana sorgu kayıtlarını almak için kullanılmaktadır. IN, alt sorgudaki kayıtların değerine eşit olan ana sorgu kayıtlarını almak için kullanılır. NOT IN ise alt sorgudaki kayıtların değerine eşit olmayan ana sorgu kayıtlarını almak için kullanılır.

2.2.3. Çoklu Satır Alt Sorguları

Alt sorgudan bir satır yerine birden fazla satırın elde edildiği durumlar çoklu satır alt sorgusu olarak adlandırılır.

Bu tür sorgular IN, ANY, ALL gibi işleçler yardımıyla yapılabilmektedir.

">ANY" en azdan daha büyük**, "<ANY"** ise en çoktan daha az anlamına gelmektedir.

">ALL" en büyükten daha büyük, **"<ALL"** ise en küçükten daha küçük anlamına gelmektedir.

Örnek IN işleci:

İndirim oranı %30'dan daha fazla olan tüm ürünleri gösteren SQL kodu aşağıdaki gibidir.

```
SELECT *
FROM Ürünler
WHERE U_No IN
(SELECT U_No FROM Sipariş
WHERE İndirim = .30);
```

Örnek ANY işleci:

Birim fiyatı, % 30 veya daha fazla indirimle satılmış herhangi bir ürünün birim fiyatından yüksek olan tüm ürünleri gösteren SQL kodu aşağıdaki gibidir.

```
SELECT *
FROM Ürünler
WHERE B_Fiyat > ANY
(SELECT B_Fiyat FROM Sipariş
WHERE İndirim >= .30);
```

Örnek ALL işleci:

Birim fiyatı, % 30 veya daha fazla indirimle satılmış tüm ürünlerin birim fiyatından yüksek olan ürünleri listelemek için kullanılan SQL kodu aşağıdaki gibidir.

```
SELECT *
FROM Ürünler
WHERE B_Fiyat > ALL
(SELECT B_Fiyat FROM Sipariş
WHERE İndirim >= .30);
```

2.2.4. Tek Satır Alt Sorguları

Alt sorgudan tek satırın elde edildiği sorgulardır.

Örnek:

p_no	Adı	Bölüm_no
10	ali	200
11	veli	200
12	ahmet	50
14	mehmet	60

Yukarıdaki PERSONEL tablosuna göre Personel numarası "10" olan personelle aynı bölümde çalışan personelin isimlerini listeleyecek SQL kodu aşağıdaki gibi olacaktır.

```
SELECT Adı, Bölüm_no
FROM PERSONEL
WHERE Bölüm_no=
(SELECT Bölüm_no
FROM PERSONEL
WHERE p_no=10);
```

Alt sorgu sonucunda "10" numaralı personelin çalıştığı bölümün numarası, yani "200" değeri elde edilecektir. Ana sorguda ise çalıştığı bölüm numarası "200" olan kişileri sorgulamaktadır. Sorguyu yazıp çalıştırdığımızda aşağıdaki sonuç elde edilecektir.

Adı	Bölüm_no
ali	200
veli	200

UYGULAMA FAALİYETİ

PERSONEL ve BÖLÜM tabloları aşağıda verilmiştir.

PERSONEL

p_no	Adı	Bölüm_no
10	ali	200
11	veli	200
12	ahmet	50
14	mehmet	60

BÖLÜM

Bölüm_no	Bölüm_adı
200	Bilgi İşlem
50	Musasebe
60	Halkla İlişkiler
70	Satış

İşlem Basamakları	Öneriler
➢ Yukarıdaki alan ve verilerden oluşan PERSONEL ve BÖLÜM tablolarını oluşturunuz.	➢ Tabloları veri sayfası görünümünde açarak oluşturabilirsiniz.
➢ PERSONEL ve BÖLÜM tablolarını Kartezyen çarpımı kullanarak birleştiriniz.	
➢ PERSONEL ve BÖLÜM tablolarını eşiti olan birleştirme kullanarak birleştirecek SQL kodunu yazınız.	➢ INNER JOIN kullanınız.
➢ PERSONEL ve BÖLÜM tablolarını eşiti olmayan birleştirme kullanarak birleştirecek SQL kodunu yazınız.	➢ RIGHT JOIN kullanarak dış birleştirmeyi gerçekleştiriniz.
➢ PERSONEL tablosuna göre Personel numarası “12” olan personelle aynı bölümde çalışan personelin isimlerini listeleyecek SQL kodunu yazıp çalıştırınız.	➢ Tek satır alt sorgu oluşturunuz.

KONTROL LİSTESİ

Bu faaliyet kapsamında aşağıda listelenen davranışlardan kazandığınız beceriler için **Evet**, kazanamadığınız beceriler için **Hayır** kutucuğuna (X) işareti koyarak kendinizi değerlendiriniz.

Değerlendirme Ölçütleri	Evet	Hayır
1. SQL kullanarak istenilen tabloları birleştirebildiniz mi?		
2. Çoklu tabloları kullanabildiniz mi?		
3. Kartezyen çarpımı kullandınız mı?		
4. Eşiti olan birleştirme işlemini öğrendiniz mi?		
5. Eşiti olmayan birleştirme işlemini gerçekleştirebildiniz mi?		
6. SQL kodları kullanarak alt sorgu tanımlaması yapabildiniz mi?		
7. Alt sorgu düzenleme kurallarının neler olduğunu öğrendiniz mi?		
8. Tek ve çok sütunlu alt sorgular oluşturdunuz mu?		

DEĞERLENDİRME

Değerlendirme sonunda "**Hayır**" şeklindeki cevaplarınızı bir daha gözden geçiriniz. Kendinizi yeterli görmüyorsanız öğrenme faaliyetini tekrar ediniz. Bütün cevaplarınız "**Evet**" ise "Ölçme ve Değerlendirme" ye geçiniz

ÖLÇME VE DEĞERLENDİRME

Aşağıdaki cümlelerin başında boş bırakılan parantezlere, cümlelerde verilen bilgiler doğru ise D, yanlış ise Y yazınız.

1. ( ) Birinci tablodaki tüm satırların ikinci tablodaki tüm satırlarla birleşmediği durumlarda eşiti olan birleştirme türü kullanılır.

2. ( ) Tablolar birleştirilirken eşiti olmayan birleştirme yapmak için INNER JOIN deyimi kullanılır.

3. ( ) Tablo birleştirme sırasında eşleşmeyen satırların da birleştirilip sonuca dahil edilmesi istenilen durumlarda "Eşiti olmayan birleştirme" kullanılır.

4. ( ) Sağ dış birleştirme oluşturmak için RIGHT JOIN kullanılır.

5. ( ) Alt sorgu içerisinde ORDER BY sözcüğü kullanılamaz.

6. ( ) All işleci alt sorgudaki kayıtların değerine eşit olmayan ana sorgu kayıtlarını almak için kullanılır.

7. ( ) Alt sorgudan elde edilen sonuçta tek satırlık bir sonuç elde ediliyorsa bu işleme çoklu satır sorguları denir.

DEĞERLENDİRME

Cevaplarınızı cevap anahtarıyla karşılaştırınız. Yanlış cevap verdiğiniz ya da cevap verirken tereddüt ettiğiniz sorularla ilgili konuları faaliyete geri dönerek tekrarlayınız. Cevaplarınızın tümü doğru ise bir sonraki öğrenme faaliyetine geçiniz

ÖĞRENME FAALİYETİ–3

AMAÇ

Veri işleme (DML) sorgularını kullanabileceksiniz.

ARAŞTIRMA

- DML sorgularının neler olduğunu ve hangi amaçlar için kullanıldığını araştırınız.

3. DML SORGULARI

Bir tabloya yeni bir kayıt eklemek, mevcut kayıt üzerinde değişiklik yapmak veya bir kaydı silmek istediğimiz durumlarda, Select deyimi kullanarak yapmış olduğumuz sorgulamalar işe yaramaz. DML (Data Manipulation Language) dili kullanılarak, veri tabanı tablosuna veri ekleme, silme ve güncelleme işlemleri yapılabilmektedir.

3.1. Tabloya Satır Ekleme İşlemi

Bir tabloya bir veya daha çok satır (kayıt) eklemek için INSERT INTO deyimi kullanılır. Bu işlem "Ekleme Sorgusu" olarak adlandırılmaktadır.

3.1.1. INSERT Deyimi Yapısı

Bir tabloya yeni veri eklemek için **INSERT** deyimi kullanılır. INSERT deyimi, INTO ve VALUES ifadeleri ile birlikte kullanılır ve tabloya yeni veri eklenmesi sağlanır.

Veri ekleme sırasında ilk olarak INSERT INTO ile ekleme yapılacak olan tablo veya sütunlar belirlenir. Daha sonra VALUES ifadesi ile eklenecek olan değerler parantez içinde belirtilir. Verilerin sıralanışına dikkat etmek gerekmektedir. Örneğin (Personel_no, Adı, Soyadı, Bölümü) şeklinde sıralanmış olan alanlarda, eklenecek değerlerin de aynı sırada olması gerekmektedir.

Tablodaki tüm alanlara değer girilecekse, tablo isminden sonra sütun isimlerinin belirtilmesine gerek yoktur. Çünkü bu tür durumda tablodaki tüm alanlara veri girişi sağlanmış olacaktır.

- **Sütun ismi belirtmeden kullanımı:**

```
INSERT INTO tablo
VALUES (değer1, değer2…)
```

- **Sütun ismi belirterek kullanımı:**

INSERT INTO tablo (sütun1, sütun2…)
VALUES (değer1, değer2…)

Mevcut kayıtlarda bulunan tek alanlardaki verileri değiştirmek için ekleme sorgusu yerine güncelleştirme sorgusu kullanılmaktadır. Ekleme sorguları sadece veri satırları eklemek için kullanılır.

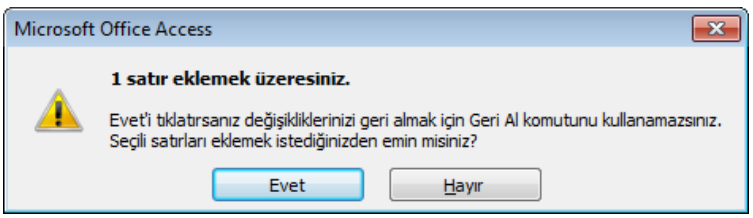
Örnek:

Personel_no	adı	Soyadı	Bölümü	İl_Kodu
1	ali	ak	matematik	34
2	veli	kara	matematik	34
30	ayşe	ay	ingilizce	35
44	mehmet	tek	müzik	07
8	fatma	sam	bilgisayar	34
9	yeşim	yek	resim	35

Yukarıdaki Personel tablosuna bir kayıt eklemek istiyorsak SQL kodu aşağıdaki şekilde olacaktır.

```
INSERT INTO PERSONEL
VALUES (11, 'ÖYKÜ', 'SER', 'TARİH', '6');
```

Kodu yazıp çalıştır düğmesine bastığımız zaman program bize yeni bir satır eklemek üzere olduğumuza dair aşağıdaki uyarı ekranını görüntüleyecektir.



Evet düğmesine bastıktan sonra Personel tablomuzu çalıştırdığımızda yeni kayıt eklenmiş hali aşağıdaki gibi olacaktır.

Personel_no	adı	Soyadı	Bölümü	İl_Kodu
1	ali	ak	matematik	34
2	veli	kara	matematik	34
30	ayşe	ay	ingilizce	35
44	mehmet	tek	müzik	07
8	fatma	sam	bilgisayar	34
9	yeşim	yek	resim	35
10	ÖYKÜ	SER	TARİH	6

NOT: Veri eklenecek alan sayısal veri türü tanımlanmışsa eklenecek veri tek tırnaklar arasında yazılmaz. Metin ve tarih veri türünde tanımlanmış ise eklenecek veri tek tırnaklar içinde yazılır.

Örnek:

Personel tablosunda sadece Personel_no ve Bölümü alanlarına veri girişi yapmak istiyorsak SQL sorgu kodu aşağıdaki şekilde olacaktır.

```
INSERT INTO PERSONEL(Personel_no, Bölümü)
VALUES (18, 'Müzik');
```

Kodu çalıştırdığımız zaman Personel tablosunun yeni hali aşağıdaki şekilde olacaktır.

Personel_no	adı	Soyadı	Bölümü	İl_Kodu
1	ali	ak	matematik	34
2	veli	kara	matematik	34
30	ayşe	ay	ingilizce	35
44	mehmet	tek	müzik	07
8	fatma	sam	bilgisayar	34
9	yeşim	yek	resim	35
10	ÖYKÜ	SER	TARİH	6
18			Müzik	

3.1.2. NULL Değer Ekleme

Oluşturduğunuz tablodaki bir veya birden fazla alana Null değer girilmesi gerekebilir. Bunun için kullanılan iki yol mevcuttur. Birinci yöntem ile INSERT INTO deyimi içerisinde Null değer alacak olan alanlar yazılmaz.

```
INSERT INTO PERSONEL(Personel_no, Bölümü)
VALUES (18, 'Müzik');
```

Diğer yöntem ise aşağıdaki şekilde uygulanır;

```
INSERT INTO PERSONEL(Personel_no, adı, Soyadı, Bölümü, İl_Kodu)
VALUES (18, NULL, NULL, 'Müzik', NULL)
```

Bu yöntem ile boş bırakılmak istenilen alanlara *NULL* şeklinde bir atama yapılmıştır.

3.1.3. Fonksiyonların Kullanımı

INSERT INTO içerisinde fonksiyonlar da kullanılabilmektedir. Örneğin PERSONEL tablosuna yeni bir kayıt eklerken Giriş_tarihi isimli alana o günün tarih ve zaman bilgisini eklemek istiyorsak komut satırı aşağıdaki gibi olacaktır.

```
INSERT INTO PERSONEL(Personel_no, Bölümü, Giriş_tarihi)
VALUES( 55, 'Müzik', now);
```

Bu kodu çalıştırdığımız zaman Personel tablosunun yeni hali aşağıdaki gibi olacaktır.

Personel_no	adı	Soyadı	Bölümü	İl_Kodu	Giriş_tarihi
1	ali	ak	matematik	34	
2	veli	kara	matematik	34	
30	ayşe	ay	ingilizce	35	
44	mehmet	tek	müzik	07	
8	fatma	sam	bilgisayar	34	
9	yeşim	yek	resim	35	
10	ÖYKÜ	SER	TARİH	6	
55			Müzik		09.03.2012 18:36:08

Bu örnekte Now() fonksiyonu kullanılarak, sisteminizin tarih ve saat bilgisi 55 numaralı personelin işe giriş tarihi olarak eklenmiştir.

3.1.4. Bir Diğer Tablodan Satır Kopyalama

INSERT işlemini kullanarak bir tabloda bulunan kayıtları istediğiniz diğer bir tabloya kopyalayabilirsiniz. Bu işlem için yapılması gereken Values yerine Select ifadesini kullanmaktır.

Kullanımı:

INSERT INTO Tablo1(sütun1, sütun2,…)
SELECT sütun1, sütun2
FROM Tablo2

Örnek:

ÇALIŞANLAR tablosunda bulunan kayıtları PERSONEL tablosuna kopyalayalım.

Personel_no	adı	Soyadı	Bölümü	İl_Kodu
1	ali	ak	matematik	34
2	veli	kara	matematik	34
30	ayşe	ay	ingilizce	35
44	mehmet	tek	müzik	07
8	fatma	sam	bilgisayar	34
9	yeşim	yek	resim	35
10	ÖYKÜ	SER	TARİH	6

Şekil 3.1: Personel tablosu

Personel_no	adı	Soyadı	Bölümü	İl_Kodu
11	aylin	az	kimya	35

Şekil 3.2: Çalışanlar tablosu

Çalışanlar tablosunda tek bir kayıt bulunmaktadır. Bu kaydı Personel tablosuna kopyalamak için gerekli olan komut satırı aşağıdaki gibi olacaktır.

```
INSERT INTO PERSONEL(Personel_no, adı, Soyadı, Bölümü)
SELECT Personel_no, adı, Soyadı, Bölümü
FROM ÇALIŞANLAR
```

Yukarıdaki kod, sorgu penceresine yazılıp çalıştırıldığı zaman Personel tablosunun yeni hali aşağıdaki gibi olacaktır.

Personel_no	adı	Soyadı	Bölümü	İl_Kodu
1	ali	ak	matematik	34
2	veli	kara	matematik	34
30	ayşe	ay	ingilizce	35
44	mehmet	tek	müzik	07
8	fatma	sam	bilgisayar	34
9	yeşim	yek	resim	35
10	ÖYKÜ	SER	TARİH	6
11	aylin	az	kimya	

3.2. Tablodaki Verileri Güncelleme

Bir tabloda bulunan kayıt veya kayıtların istenildiği zaman değiştirilmesi mümkündür. Tablolarda güncelleme işlemini gerçekleştirmek için UPDATE komutu kullanılır. SET ifadesi ile güncellenecek alanlar ve bu alanların alacakları yeni değerler belirlenir. WHERE deyimi ile de verilerin güncelleştirilmesi için koşul belirlenir. Eğer WHERE ifadesi ile bir koşul belirlenmezse tablodaki tüm kayıtlar güncellenmiş olacağından WHERE kullanmaya dikkat edilmesi gerekmektedir.

Kullanımı:

UPDATE tablo
SET sütun1=değer1, sütun2=değer2,..
WHERE Koşul

Örnek:

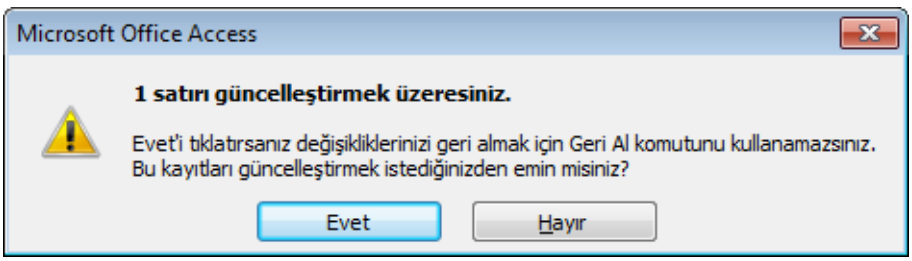
Personel tablosunda yer alan 30 numaralı personelin “35” olan İl_Kodu bilgisini “06” olarak değiştirmek isteyelim.

Personel_no	adı	Soyadı	Bölümü	İl_Kodu
1	ali	ak	matematik	34
2	veli	kara	matematik	34
30	ayşe	ay	ingilizce	35

Bu işlem için yazılması gereken SQL kodu aşağıdaki şekilde olacaktır.

```
UPDATE PERSONEL
SET İl_Kodu = '06'
WHERE Personel_no = 30
```

Yukarıdaki kodu yazıp çalıştırdığımızda tek bir satırın güncelleştirileceğini belirtip onaylamamız istenen aşağıdaki pencere görüntülenecektir.



Buradan Evet düğmesine tıklanır. Daha sonra Personel tablosu çalıştırıldığı zaman kayıt güncelleştirme aşağıdaki gibi olacaktır.

Personel_no	adı	Soyadı	Bölümü	İl_Kodu
1	ali	ak	matematik	34
2	veli	kara	matematik	34
30	ayşe	ay	ingilizce	06

Örnek:

PERSONEL tablosundaki tüm illerin kodunu "06" olarak değiştirmek istersek SQL kodları aşağıdaki gibi olmalıdır.

```
UPDATE PERSONEL
SET İl_Kodu = '06'
WHERE Personel_no
```

Kodu yazıp çalıştırdığımız zaman personel tablosu aşağıdaki gibi güncellenmiş olacaktır.

Personel_no	adı	Soyadı	Bölümü	İl_Kodu
1	ali	ak	matematik	06
2	veli	kara	matematik	06
30	ayşe	ay	ingilizce	06
44	mehmet	tek	müzik	06
8	fatma	sam	bilgisayar	06
9	yeşim	yek	resim	06
10	ÖYKÜ	SER	TARİH	06

3.3. Tablolardan Veri Silme

Bir tabloda bulunan kayıt veya kayıtların istenildiği zaman silinmesi mümkündür. Tablolarda silme işlemini gerçekleştirmek için DELETE komutu kullanılır. DELETE komutu kullanılırken FROM eki ile birlikte tablo ismi yazılarak hangi tablodan veri silinmesi istendiği belirtilebilir. WHERE deyimi ile de verilerin silinme koşulu belirlenir. Eğer WHERE ifadesi ile bir koşul belirlenmezse tablodaki tüm kayıtlar silineceğinden WHERE kullanmaya dikkat edilmesi gerekmektedir.

Kullanımı:

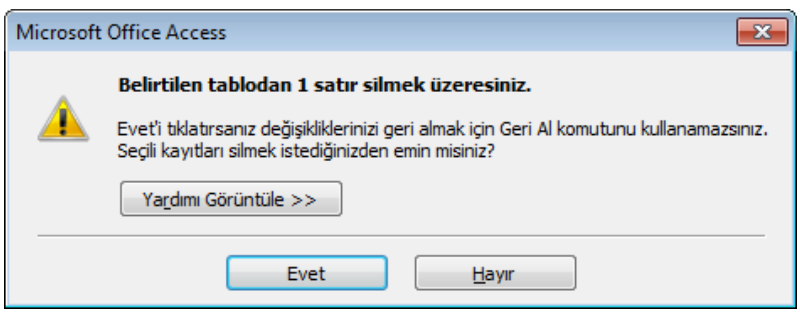
DELETE FROM Tablo
WHERE Koşul

Örnek:

Personel tablosunda yer alan 30 numaralı personeli silmek için yazılacak SQL komutu aşağıdaki gibi olacaktır.

DELETE FROM PERSONEL WHERE Personel_no = 30

Bu kodu yazıp Tasarım sekmesindeki Çalıştır düğmesine bastığımızda aşağıdaki uyarı ekranı karşımıza çıkacaktır. Buradan Evet düğmesine tıkladığımız zaman Personel tablosundaki 30 numaralı kayıt silinmiş olacaktır.



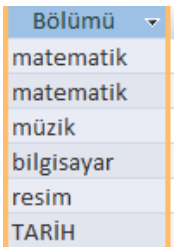
Örnek:

DELETE FROM ÇALIŞANLAR kodu ile ÇALIŞANLAR tablosunda bulunan tüm kayıtlar silinecektir.

Örnek:

Personel tablosunda Bölümü “müzik” ve “resim” olanlar dışındaki diğer kayıtların silinmesini istersek;

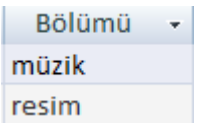
Personel tablosunda silme işlemi uygulamadan önce Bölümü alanında bulunan kayıtlar aşağıda gösterilmiştir



Bölümü
matematik
matematik
müzik
bilgisayar
resim
TARİH

Aşağıdaki kodları yazıp çalıştırdığımızda personel tablosundaki Bölümü müzik ve resim olan kayıtlar dışındaki tüm kayıtlar silinecektir.

```
DELETE FROM PERSONEL
WHERE Bölümü NOT IN ("müzik", "resim")
```



Bölümü
müzik
resim

UYGULAMA FAALİYETİ

Aşağıdaki şekilde ÖĞRENCİ tablosu görülmektedir.

kimlik_no	Adı	Soyadı	Doğum_yeri	Doğum_tarihi	Okulu	Bölümü
11111111111	azra	ay	Lefkoşa	12.07.1988	Ankara Ün.	Kimya
12341225874	aylin	ak	Ankara	11.08.1990	İstanbul İn.	Hukuk
22222222222	hasan	öz	İzmir	20.01.1991	Ege Ün.	İşletme
22224444444	nuri	kal	Manisa	17.01.1990	Dicle Ün.	Fizik

İşlem Basamakları	Öneriler
➢ Yukarıdaki şekilde görüldüğü gibi alan ve kayıtları içeren ÖĞRENCİ tablosunu oluşturunuz.	
➢ Yukarıdaki tabloya aşağıdaki verileri içeren bir satır eklemeyi sağlayacak sql kodunu yazınız. Kimlik_no:33332222255 Adı: Özge Soyadı: Şen Doğum_yeri: Mersin Doğum_tarihi:23.03.1991 Okulu: Marmara Ün. Bölümü: Matematik ➢ Sorguyu çalıştırınız.	➢ INSERT INTO ÖĞRENCİ ile kayıt eklemek istediğiniz tabloyu belirtip values ile de değerleri giriniz.
➢ Adı: Ayşe, Soyadı:Yaz ve diğer değerler Null olacak şekilde yeni bir satır eklemeyi sağlayacak SQL kodunu yazınız.	➢ VALUES(NULL,'Ayşe', 'Yaz', NULL, NULL, NULL, NULL)
➢ ÖĞRENCİ tablosunda yer alan "11111111111" kimlik nolu öğrencinin Kimya olan bölümünü İngilizce olarak değiştirmeyi sağlayacak SQL kodunu yazınız.	➢ UPDATE komutu kullanarak güncelleme yapınız.
➢ ÖĞRENCİ tablosundaki tüm öğrencilerin bölümünü "Resim" olarak değiştiriniz.	➢ UPDATE komutunu kullanınız.
➢ "11111111111" kimlik numaralı öğrenciyi ÖĞRENCİ tablosundan silmek için gerekli sql kodunu yazınız.	➢ DELETE komutunu kullanınız.
➢ ÖĞRENCİ tablosunda bulunan tüm kayıtları siliniz.	➢ DELETE FROM ÖĞRENCİ

KONTROL LİSTESİ

Bu faaliyet kapsamında aşağıda listelenen davranışlardan kazandığınız beceriler için **Evet**, kazanamadığınız beceriler için **Hayır** kutucuğuna (X) işareti koyarak kendinizi değerlendiriniz.

Değerlendirme Ölçütleri	Evet	Hayır
1. DML sorgusu kullanarak tablonuza satır eklediniz mi?		
2. DML sorgusu kullanarak tablonuza null değer eklediniz mi?		
3. Fonksiyonları kullanabildiniz mi?		
4. Bir tablodan diğer bir tabloya istediğiniz satırı kopyalayabildiniz mi?		
5. Tablonuzdaki verileri güncelleyecek sorguyu oluşturdunuz mu?		
6. DML sorgusu kullanarak tablonuzdan veri sildiniz mi?		

DEĞERLENDİRME

Değerlendirme sonunda "**Hayır**" şeklindeki cevaplarınızı bir daha gözden geçiriniz. Kendinizi yeterli görmüyorsanız öğrenme faaliyetini tekrar ediniz. Bütün cevaplarınız "**Evet**" ise "Ölçme ve Değerlendirme" ye geçiniz

ÖLÇME VE DEĞERLENDİRME

Aşağıdaki cümlelerin başında boş bırakılan parantezlere, cümlelerde verilen bilgiler doğru ise D, yanlış ise Y yazınız.

Aşağıdaki soruları dikkatlice okuyunuz ve doğru seçeneği işaretleyiniz.

1. ( ) Bir tabloya satır eklemek için INSERT INTO deyimi kullanılır.

2. ( ) Values ifadesi ile ekleme yapılacak olan tablo ismi belirtilir.

3. ( ) Insert Into içinde fonksiyonlar kullanılamaz.

4. ( ) Tablolarda güncelleme işlemini gerçekleştirmek için Delete komutu kullanılır.

5. ( ) Update komutu ile kullanılan SET ifadesi güncellenecek alanlar ve bu alanların alacakları yeni değerleri belirlemek için kullanılır.

DEĞERLENDİRME

Cevaplarınızı cevap anahtarıyla karşılaştırınız. Yanlış cevap verdiğiniz ya da cevap verirken tereddüt ettiğiniz sorularla ilgili konuları faaliyete geri dönerek tekrarlayınız. Cevaplarınızın tümü doğru ise "Modül Değerlendirme"ye geçiniz.

MODÜL DEĞERLENDİRME

Aşağıdaki cümlelerin başında boş bırakılan parantezlere, cümlelerde verilen bilgiler doğru ise D, yanlış ise Y yazınız.

1. ( ) Belirtilen tablodaki alanların değerlerini belirtilen ölçütlere göre değiştirecek bir güncelleme sorgusu oluşturmak için UPDATE deyimi kullanılır.

2. ( ) Veri tabanında verilere erişebilmek için SELECT deyimi kullanılmaktadır.

Aşağıdaki cümlelerde boş bırakılan yerlere doğru sözcüğü yazınız.

3. Tabloların satırlarının herhangi bir sütuna göre sıralanmasının istendiği durumlarda SELECT deyimi **……** ile birlikte kullanılır.

4. Bir dizenin içerdiği sayısal değerleri biçimlendirilerek sayısal veri türüne dönüştürmek için …….fonksiyonu kullanılır.

Aşağıdaki soruları dikkatlice okuyunuz ve doğru seçeneği işaretleyiniz.

5. Bilgisayardaki geçerli sistem tarih ve saatini bildirmek için aşağıdakilerden hangisi kullanılır?
A) SELECT Now()
B) SELECT Day()
C) SELECT Tarih()
D) SELECT Zaman()

6. Sütunların içerdiği sayısal değerleri toplamak amacıyla kullanılan fonksiyon aşağıdakilerden hangisidir?
A) AVG()
B) STDEV()
C) SUM()
D) VARP()

7. Birden fazla sütun içinde gruplama yapmak için aşağıdakilerden hangisi kullanılır?
A) ORDER BY
B) GROUP BY
C) SELECT BY
D) FROM BY

8. Tabloların birleştirilmesi sırasında eşiti olan birleştirme yapılırken kullanılan deyim aşağıdakilerden hangisidir?

A) INNER JOIN
B) LEFT JOIN
C) RIGHT JOIN
D) INSERT JOIN

9. Alt sorgudaki kayıtların değerine eşit olmayan ana sorgu kayıtlarını almak için kullanılan işleç aşağıdakilerden hangisidir?

A) SOME
B) ANY
C) ALL
D) IN

10. Bir tabloda bulunan kayıtların istenildiği zaman silinebilmesi için kullanılan komut aşağıdakilerden hangisidir?

A) UPDATE
B) INSERT
C) DELETE
D) WHERE

DEĞERLENDİRME

Değerlendirme sonunda "**Hayır**" şeklindeki cevaplarınızı bir daha gözden geçiriniz. Kendinizi yeterli görmüyorsanız öğrenme faaliyetini tekrar ediniz. Bütün cevaplarınız "**Evet**" ise bir sonraki modüle geçmek için öğretmeninize başvurunuz.

CEVAP ANAHTARLARI

ÖĞRENME FAALİYETİ-1'İN CEVAP ANAHTARI

1	Doğru
2	Yanlış
3	Doğru
4	A
5	D
6	D
7	B
8	C
9	A

ÖĞRENME FAALİYETİ-2'NİN CEVAP ANAHTARI

1	Yanlış
2	Yanlış
3	Doğru
4	Doğru
5	Doğru
6	Yanlış
7	Yanlış

ÖĞRENME FAALİYETİ-3'ÜN CEVAP ANAHTARI

1	Doğru
2	Yanlış
3	Yanlış
4	Yanlış
5	Doğru

MODÜL DEĞERLENDİRMENİN CEVAP ANAHTARI

1	Doğru
2	Doğru
3	Ordey By
4	Cint
5	A
6	C
7	B
8	A
9	D
10	C

KAYNAKÇA

- YALÇIN Özkan, **Veri Tabanı Sistemleri**, Alfa Yayınları, İstanbul, 2009.
- ÇİÇEK Musa, **Veritabanı Tasarımı ve SQL Sorgulama Dili**, Nirvana Yayınları, Ankara, 2010.
- GÜRKAN Osman, **Microsoft Office 2010**, Nirvana Yayınları, Ankara, 2010.
- http://altanmesut.trakya.edu.tr/vt/Verileri_gruplayarak_analiz.ppt (14.03.2012,14:00)
- http://altanmesut.trakya.edu.tr/vt/Sql_Fonksiyonları.ppt (08.03.2012,15:30)
- http://moodle.midas.baskent.edu.tr/mod/resource/view.php?inpopup=true&id=66 (15.03.2012,16:30)