Sql uyum Geliştirme

19.06.2018

DML (Veri İşleme) KOMUTLARI

- Select
- * where, and,or
- * like
- * null, isnull
- * between
- * in
- İnsert
- Delete
- Update

•SELECT

Tablodaki tüm alanları getirmek için:

```sql
SELECT * FROM table_name;
```

Tabloda istenilen getirmek için:

```sql
SELECT column1, column2, ...
FROM table_name;
```

- **WHERE**

Belli koşullara göre veri getirmek istediğimizde where ile şart bildiririz

```
SELECT column1, column2, ...
FROM table_name
WHERE condition;
```

Where Operatörleri

> Operatörü : Büyüktür operatörüdür. WHERE deyimi ile birlikte kullanılır, ilk değerin ikincisinden büyüklüğünü kontrol eder.

```
1  SELECT EmployeeID,FirstName, LastName
2  FROM Employees
3  WHERE EmployeeID > 5
```

< Operatörü : Küçüktür operatörüdür. WHERE deyimi ile birlikte kullanılır, ilk değerin ikincisinden küçüklüğünü kontrol eder.

```
1  SELECT EmployeeID,FirstName, LastName
2  FROM Employees
3  WHERE EmployeeID < 5
```

>= Operatörü : Büyük eşittir operatörüdür. WHERE deyimi ile birlikte kullanılır, ilk değerin ikincisinden büyüklüğünü ve ona eşitliğini kontrol eder.

```
1  SELECT EmployeeID,FirstName, LastName
2  FROM Employees
3  WHERE EmployeeID >= 5
```

<= Operatörü : Küçük eşittir operatörüdür. WHERE deyimi ile birlikte kullanılır, ilk değerin ikincisinden küçüklüğünü ve ona eşitliğini kontrol eder.

```
1  SELECT EmployeeID,FirstName, LastName
2  FROM Employees
3  WHERE EmployeeID <= 5
```

<>, != Operatörü : İki türde de kullanılabilir, <>, != gibi. Eşit değildir operatörüdür. Belirtilen ilk değerin ikincisine eşit olmadığını kontrol eder. Burada != yerine <> kullanırsak da aynı sonucu alırız

```
1  SELECT EmployeeID,FirstName, LastName FROM Employees
2  WHERE EmployeeID != 5
```

= Operatörü : Eşitlik operatörüdür.

```
1  SELECT EmployeeID,FirstName, LastName FROM Employees
2  WHERE EmployeeID = 5
```

AND Operatörü : Verilen şartlardan tümünün sağlanması istendiğinde kullanılır.

```
SELECT column1, column2, ...
FROM table_name
WHERE condition1 AND condition2 AND condition3 ...;
```

```
SELECT * FROM Customers
WHERE Country='Germany' AND City='Berlin';
```

OR Operatörü : Verilen şartlardan herhangi biri sağlanması istendiğinde kullanılır

```
SELECT column1, column2, ...
FROM table_name
WHERE condition1 OR condition2 OR condition3 ...;
```

```
SELECT * FROM Customers
WHERE City='Berlin' OR City='München';
```

IS NULL Operatörü : Tabloda verilen alanda null olan kayıtları getirir.

```
SELECT column_names
FROM table_name
WHERE column_name IS NULL;
```

```
SELECT LastName, FirstName, Address FROM Persons
WHERE Address IS NULL;
```

IS NOT NULL Operatörü : Tabloda verilen alanda null olmayan kayıtları getirir.

```
SELECT column_names
FROM table_name
WHERE column_name IS NOT NULL;
```

```
SELECT LastName, FirstName, Address FROM Persons
WHERE Address IS NULL;
```

LİKE Operatörü

Soyadı “Z” harfi ile başlayan üyeler

```
SELECT * FROM uyeler WHERE soyisim LIKE 'Z%’
```

Adının son harfi “t” olan üyeler

```
SELECT * FROM uyeler WHERE isim LIKE '%t'
```

İsminin içerisinde “er” ifadesi geçen üyeler

```
SELECT * FROM uyeler WHERE isim LIKE '%er%'
```

isminin içerisinde “er” ifadesi **geçmeyen** üyeler

```
SELECT * FROM uyeler WHERE isim NOT LIKE '%er%'
```

- uye_adi alanında hdzafer, hczafer, hazafer gibi kayıtlar
- LIKE ‘_a_’ : Üç harfli ortadaki harfi “a” olanlar.
- LIKE ‘ m_s_n’ : *mısın, musun, müsün* veya *muson* gibi bir çok kelime

```
SELECT * FROM uyeler WHERE uye_adi LIKE 'h_zafer'
```

- LIKE ‘c[ai]n’: Bu desen *can* ve *cin* kelimelerini kapsar.
- ismi erhan, ercan, erkan veya erman olan üyeler
- Adı E veya K harfi ile **başlamayan** üyeler

```
SELECT * FROM uyeler WHERE uye_adi LIKE 'er[hckm]an'
```

IN Operatörü

Verilen alan , verilen listedeki alanlardan biriyse sorguya dahil olur

IN Syntax

```
SELECT column_name(s)
FROM table_name
WHERE column_name IN (value1, value2, ...);
```

or:

```
SELECT column_name(s)
FROM table_name
WHERE column_name IN (SELECT STATEMENT);
```

BETWEEN Operatörü

Bir alanın değerinin iki değer arasında olup olmadığına dair koşul tanımlamak istediğimizde kullanılır. Örneğin *“yaşı 20 ile 30 arasında olanlar”*

```
SELECT column_name(s)
FROM table_name
WHERE column_name BETWEEN value1 AND value2;
```

Sorgu: Yaşı 20 ile 26 arasında olan personellerin isim, soyisim ve yaş bilgileri.

```
SELECT isim, soyisim, yas FROM personel WHERE yas BETWEEN 20 AND 26
```

Sorgu: İsimleri Ahmet ile Fatma arasında olanlar (alfabetik sıralama.)

```
SELECT isim, soyisim, yas FROM personel WHERE isim BETWEEN 'Ahmet' AND 'Fatma'
```

Model 2010 olan turu devlet olan vasitaları getirin

Son kullanam tarihi bugun biten yolcular

Son kullanma tarihi 2018-03-01 ile 2018-03-10 yolcular

Son kullanma tarihi 2018-03-01 ile 2018-03-10 arasında olmayan yolcular

İstanbulda olan suruculerin iletişim bilgilerini getiren sorgu

İstanbulda olmayan suruculerin iletişim bilgilerini getiren sorgu

Hat kodu 3a 8a 8 den biri olanları getiren sorgu

Adresi İstanbul yada ankara olan surucu iletişim bilgilerini getirin

Yolculardan tc si **10** ile biten yada adı ‘**de**’ ile başlayan yolcuları getirin

Sikayet tablosunda sikayet olarak kırmızı isikta geçtiği için şikayet edenin adı soyadını getiren sql

Gecen yıl Bugun şikayet edenleri getiren sql komutu

mahalle tablosunda köy olan mahalle isimlerini getirin

yolculardan son kullanım tarihi **’2017-02-06’** den önce yolcu isim soyisim bilgilerini listeleyin

vardiyasi biten ve 19:00 ile 23:50 arasındaki vardiyaları getiriniz

skt tarihi geçmiş olup bakiyesinde 40 tl olan yolcuları listeleyin

içerinde **Mah**. İfadesi gecen mahalleleri listeleyin

mahalle adların da **haci** yada **koca** ifadesi gecen mahalleleri listeleyin

şikayet tarihi **2016-04-08 ile 2017-04-07** arasında olmayan şikayetleri listeleyin

- **DELETE**

Tablodan silmemiz gereken bir veri(data) olduğunda kullanılır. Bir personel işten ayrıldı ve artık personel tablosunda olmaması gerekiyor:

```
delete from personnel where personnelid=50;
```

```
select * from tbl_sikayet where surucu_id=10
--delete from tbl_sikayet where surucu_id=10
```

•UPDATE

Mevcut olan bir datayı değiştirmek için kullanılır. Örneğin, bir personelin bölümü değişti. Mevcut datayı silmeden güncelleyelim:

```
update personnel set departmentid=1 where personnelid=904;
```

```
update tbl_yolcu set bakiye=bakiye+10  (mevcut bakiyeye 10 ekle)
where limit between 10 and 20

update tbl_yolcu set bakiye=10 (mevcut bakiyeyi 10 yap)
where limit between 10 and 20
```

10 yıl önce bugun doğan yolcuların limitini 200 tl olarak güncelleyin

şikayetlarde kavga kelimesi gecen şikayetleri «surucu yolcu ile tartıştı olarak »güncelleyin

22222222236 tc li yolcunun bakiyesi 54 yüklenecekken yanlışlıkla 24 yüklenmiş.yeni bakiyesini güncelleyin

22222222502 tc li yolcu bakiyesinde 10 tl varken aylık 200 binişlik abonmen yaptırmış. Sistemsel bir hata nedeniyle bakiyesi sıfırlanmış. Lütfen 10 tl sini bakiyesine yükleyin

- **INSERT**

Veritabanındaki bir tabloya yeni veriler eklemek için kullanılır. Örneğin, yeni bir kişi "İnsan Kaynakları"nda işe başladı. Bu kişinin verilerini "personel" tablosuna ekleyelim:

```
insert into personnel values (567, 'Bengi', 'Akın',1);
```

**Karakter tipli(char, varchar) veriler tek tırnak içine alınır, number tipli veriler tırnaksız girilir.

```
insert into tbl_vardiya values('2018-02-
01','09:00','15:00',1)

insert into tbl_vardiya(tarih,surucu_id)
values('2018-02-01',1)

insert into tbl_vardiya values('2018-02-01','','',1)

select * from tbl_vardiya order by vardiya_id desc
```