



SQL' e Giriş

Uzm. Murat YAZICI



SQL (Structured Query Language)

- SQL Türkçe'de Yapısal Sorgulama Dili anlamına gelmektedir ve ilişkisel veritabanlarında çok geniş bir kullanım alanına sahiptir.
- SQL ile kullanıcılar veritabanı sistemleri ile iletişim kurmaktadır. Veritabanı sistemlerinin hemen hemen tamamı bu dili kullandığı için bir standart haline gelmiştir.
- SQL bir programlama dili değildir. SQL komutları kullanarak veritabanına kayıt ekleme, kayıt silme, kayıt güncelleme, tablo oluşturma ve kayıt listeleme gibi bir çok işlem gerçekleştirilir.



SQL (Structured Query Language)

- Günümüzde kullanılan programlama dillerinin neredeyse tamamı SQL komutlarını desteklemektedir.
- Standart SQL ifadelerinde fonksiyon, döngü, karşılaştırma ifadeleri gibi programlamaya yönelik ifadeler kullanılamamaktadır.
- Bu sorunu çözmek için veritabanı sistemlerinde **PL/SQL** ve **T-SQL** sorgulama dilleri geliştirilmiştir. Ancak programcılıkta kullanılan *if, case, for* gibi ifadeler PL/SQL ve T-SQL' de farklı şekillerde kullanılmaktadır.



PL/SQL ve T-SQL

PL/SQL (Procedural Language/Structured Query Language)

- Oracle tarafından geliştirilen ve Oracle veritabanı sistemlerine özel dildir. Temel SQL komutlarının yanı sıra programlamada akış kontrollerini ve değişken kullanımına olanak sağlar.

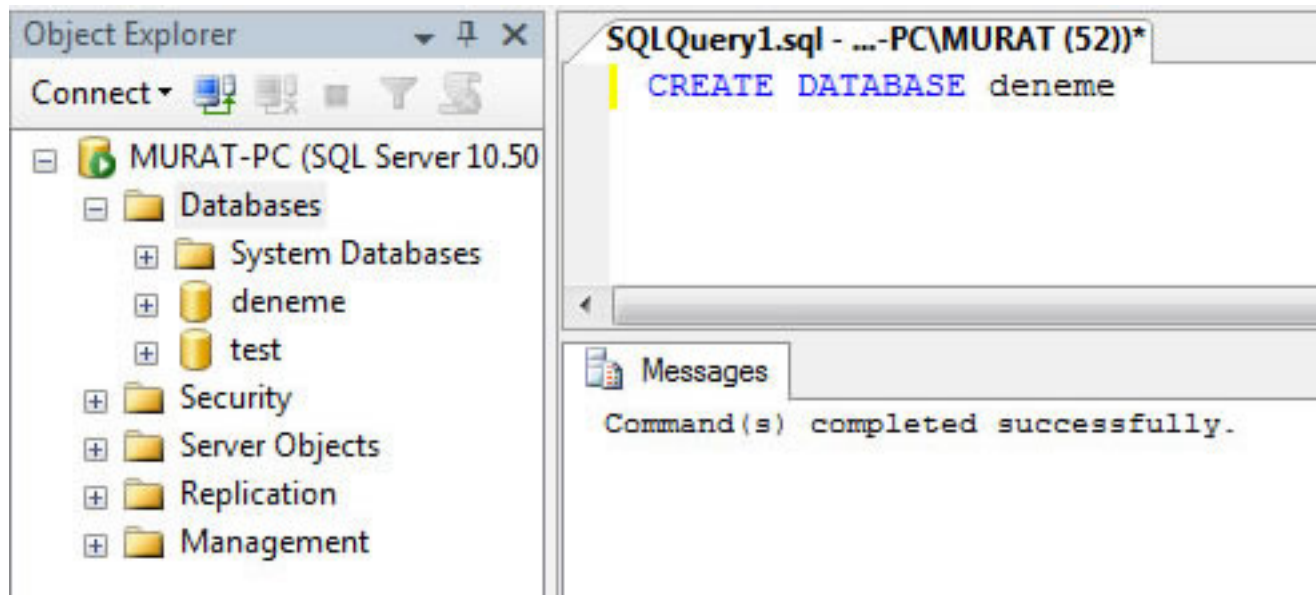
T-SQL (Transact-Structured Query Language)

- Microsoft ve Sysbase tarafından geliştirilmiştir. PL/SQL' de olduğu gibi temel SQL komutlarının yanı sıra akış kontrollerine ve değişken kullanımına olanak sağlar.

SQL Komutları

CREATE DATABASE isim

- Bu komut belirtilen isimde veritabanı (database) oluşturur.





Tablo Oluşturma

CREATE TABLE **tabloadı** (

sütunAdı1 **veriTipi** **diğerParametreler** ,

sütunAdı2 **veriTipi** **diğerParamatreler** ,

...

)

CREATE TABLE komutu kullanılırken tablonun her alanı (sütunu) için veri tipi ve başka parametreler girilir. Bazı alanlar için bu parametrelerin girilmesi gereklidir. Bu parametreleri şu şekilde sayabiliriz :

Tablo Oluşturma

1. **NOT NULL** alan için bir değerin mutlaka girilmesi gerektiğini gösterir. (Bu parametre daha çok birincil anahtarlar için kullanılır)
2. **DEFAULT** alan için bir başlangıç değeri verilmesinde kullanılır.
3. **PRIMARY KEY** (alanı birincil anahtar olarak belirlemek için)
FOREIGN KEY (alanı yabancı anahtar olarak belirlemek için)
REFERENCES (başka bir tablonun alanıyla ilişki kurmak için)

Tablo Oluşturma (Örnek)

SQLQuery1.sql - ...-PC\MURAT (52))*

```
CREATE TABLE ogrenci (  
    ogrno INT NOT NULL , -- ogrno tamsayı türünde ve boş geçilemez.  
    ad varchar(40) NOT NULL , -- ad değişken uzunlukta karakter türünde ve boş geçilemez.  
    soyad varchar(40) NOT NULL , -- soyad değişk. uzunlukta karakter türünde ve boş geçilemez.  
    dogumtarihi DATE , -- dogumtarihi tarih türünde (boş geçilebilir)  
    ortalama REAL , -- ortalama ondalık sayı türünde (boş geçilebilir)  
    PRIMARY KEY (ogrno) -- ogrno alanı birincil anahtar olarak seçilmiş.  
)
```

Messages

Command(s) completed successfully.

Tabloya kayıt ekleme

INSERT INTO tabloadı (alan1, alan2, ..., alan(n))
VALUES (deger1, deger2, ..., deger(n))

SQLQuery1.sql - ...-PC\MURAT (52))*

```
INSERT INTO ogrenci(ogrno, ad, soyad, dogumtarihi, ortalama)  
VALUES (125210, 'HAKAN', 'ERKAN', '1990-07-23', 2.85)
```



MURAT-PC.deneme - dbo.ogrenci

	ogrno	ad	soyad	dogumtarihi	ortalama
▶	125210	HAKAN	ERKAN	1990-07-23	2,85
*					

SELECT Deyimi

SELECT [DISTINCT] {*, SÜTUN, ...} **FROM** tabloAdı

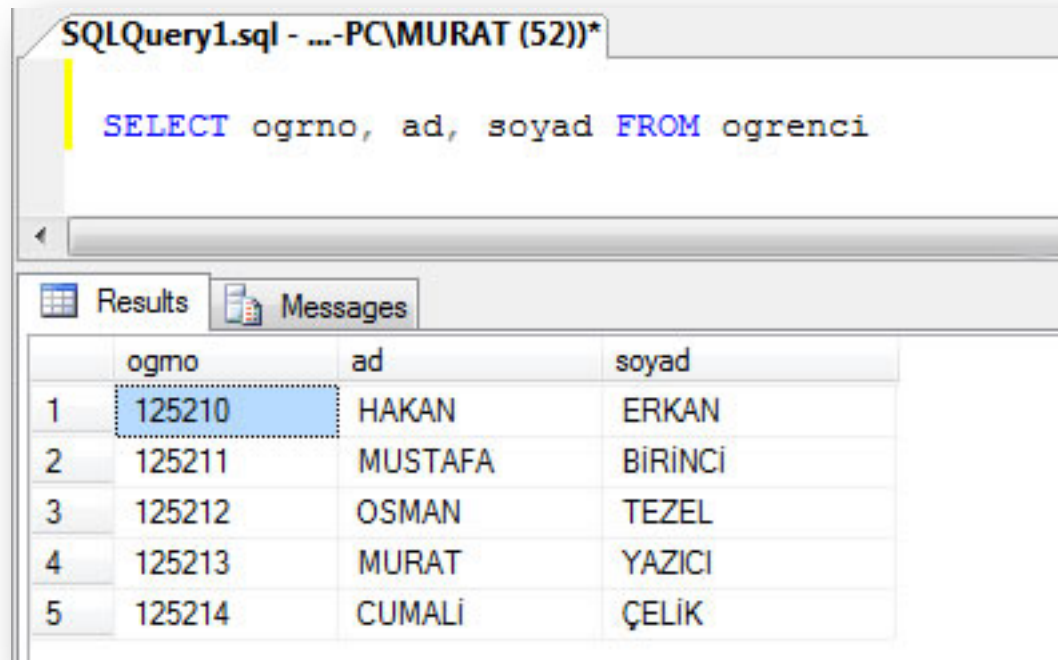
- SQL' de sorgulama işlemleri SELECT deyimi kullanılarak yerine getirilir.
- Tablonun tüm sütunlarını seçmek için * kullanılır.
- Belirli sütunların seçilmesi için sütunların isimleri belirtilmelidir.

Örnek: öğrenci tablosundaki tüm sütunları seçmek için,

```
SELECT * FROM öğrenci
```

SELECT Deyimi

ogrenci tablosundan sadece **öğrenci no**, **ad** ve **soyad** bilgilerini seçmek için,



The screenshot shows a SQL Server Enterprise Manager window titled "SQLQuery1.sql - ...-PC\MURAT (52))*". The query editor contains the following SQL statement:

```
SELECT ogrno, ad, soyad FROM ogrenci
```

Below the query editor, the "Results" tab is active, displaying the following data:

	ogrno	ad	soyad
1	125210	HAKAN	ERKAN
2	125211	MUSTAFA	BİRİNCİ
3	125212	OSMAN	TEZEL
4	125213	MURAT	YAZICI
5	125214	CUMALİ	ÇELİK

SELECT Deyimi

Özellikle birden fazla tablo ile çalışıldığı durumlarda,
- **sütun isimleri tablo ismiyle birlikte kullanılır.**

```
SELECT ogrenci.ogrno, ogrenci.ad, ogrenci.soyad, bolum.fakulteAdi
FROM ogrenci, bolum
WHERE ogrenci.fakulte = bolum.id
```

	ogrno	ad	soyad	fakulteAdi
1	125213	MURAT	YAZICI	MÜHENDİSLİK FAKÜLTESİ
2	125214	HAKAN	ERKAN	MÜHENDİSLİK FAKÜLTESİ
3	125220	CUMALİ	ÇELİK	TIP FAKÜLTESİ
4	125224	HÜSEYİN	ERTEKİN	FEN-EDEBİYAT FAKÜLTESİ

Sütunlar için Takma İsim Kullanılması

Başlıkların alan adları dışında bir isimle görüntülenmesi için **AS** anahtar kelimesi kullanılır.

```
SELECT ogrno AS "ÖĞRENCİ NO", ad AS "ADI", soyad AS "SOYADI"  
FROM ogrenci
```

	ÖĞRENCİ NO	ADI	SOYADI
1	125213	MURAT	YAZICI
2	125214	HAKAN	ERKAN
3	125220	CUMALİ	ÇELİK
4	125224	HÜSEYİN	ERTEKİN



Tekrarlı Satırların Engellenmesi

Tekrarlı satırların yalnızca bir tanesi listelenmek isteniyorsa **SELECT** deyimi **DISTINCT** anahtar kelimesi ile kullanılır.

Örnek; Aynı bölüm isimlerinden sadece birer tanesi listelenmek istenirse,

SELECT DISTINCT BOLUM FROM OGRENCI

sorgusu yazılmalıdır.



WHERE Şart ifadesi

Belirli bir koşulu sağlayan kayıtların süzülmesi için **WHERE** Şart ifadesi kullanılır.

Örnek; Bölüm sütunu **Bilgisayar** olanları listelemek için,

```
SELECT * FROM ogrenci WHERE bolum='Bilgisayar'
```

SQL sorgusu yazılmalıdır.

Karşılaştırma Operatörleri

İlişki Operatörleri

= eşit

```
Select ogrno, ad, soyad from ogrenci  
Where yas<18
```

> büyük

< küçük

```
Select * from personel  
Where maas>=2500
```

>= büyük eşit

<= küçük eşit

```
Select * from ogrenci  
Where bolum <> 'Bilgisayar'
```

<> eşit değil

Karşılaştırma Operatörleri

Mantıksal Operatörler

AND (ve) : Her iki şartı da sağlıyorsa

OR (veya) : Şartlardan herhangi birini sağlıyorsa

```
Select * from ogrenci
```

```
Where bolum='Bilgisayar' and ortalama>=3.0
```

```
Select * from personel
```

```
Where gorev='Mühendis' or maas>=3000
```



Karşılaştırma Operatörleri

(Between ... And ...) Operatörü

İki değer arasında kalanları seçme işleminde kullanılır.

Örnek, Maaşı 1000 ile 1500 arasında olanları listelemek için

```
Select ad, soyad, maas from bordro  
Where maas BETWEEN 1000 AND 1500
```

SQL sorgusu yazılmalıdır.

Karşılaştırma Operatörleri

IN Operatörü

Liste içindeki değerle karşılaştırma yapmak için kullanılır.

Örnek, Bölümü **Bilgisayar, Elektrik ve İnşaat** olan öğrencileri seçmek için,

```
Select ogrno, ad, soyad, bolum from ogrenci  
Where bolum IN ('Bilgisayar', 'Elektrik', 'İnşaat')
```

SQL sorgusu yazılmalıdır.



Karşılaştırma Operatörleri

LIKE Operatörü

Karakter grubu ile karşılaştırma yapmak için kullanılır.

M%	M harfi ile başlayanlar
%A	A harfi ile sona erenler
%U%	İçerisinde U harfi bulunanlar

```
Select ogrno, ad, soyad, bolum from ogrenci  
Where ad LIKE 'M%'
```



Karşılaştırma Operatörleri

NOT Operatörü

NOT operatörü, yapılan işlemlerin tersini kontrol etme amacıyla LIKE, IN, BETWEEN gibi deyimlerle kullanılır.

Örneğin; Doğum yeri Artvin, Rize ve Trabzon olmayan kayıtları listelemek için :

```
Select * from personel  
Where dogumyeri NOT IN ('Artvin', 'Rize', 'Trabzon')
```