<https://www.hackerrank.com/challenges/weather-observation-station-12/problem>

<https://leetcode.com/denizp/>

<https://web.kamihq.com/web/viewer.html?state=%7B%22ids%22%3A%5B%221n_HFQunbolFGSuHykem7_ybIcJbbzFyU%22%5D%2C%22action%22%3A%22open%22%2C%22userId%22%3A%22107531782040487277606%22%2C%22resourceKeys%22%3A%7B%7D%7D&kami_user_id=12540471>

[**https://sqliteonline.com/**](https://sqliteonline.com/)

**MS SQL**

SQL(Structured Query Language), veritabanindaki verileri yonettigimiz yazilim dilleridir. MS SQL, Oracle, MySQL en cok kullanilan SQL dilleridir. Bu yazimda en temel SQL komutlarindan baslayip ileri seviye sorgular yazarak sizlere SQL hakkinda en gerekli bilgileri vermeyi hedeflemekteyim.

1. **SELECT**

* Select komutuyla veri tabanindaki veriyi cekebiliriz. Veri tabanindaki hangi tablodan bu verinin cekilecegini ise FROM’dan sonra belirtmeliyiz.



Resim 1

* Ornegin Resim 1’deki ‘CUSTOMER’ tablosundaki tum sehirleri goruntulemek istersek su komutu yazmaliyiz;

>> SELECT City FROM CUSTOMER

sonuc;



* Tablonun tamamini goruntulemek istersek;

>> SELECT \* FROM CUSTOMER

1. **SELECT INTO**

SELECT INTO yapisi ile daha cok tablolarin kopyasini olusturmak amacli kullaniriz.

* Eger ‘CUSTOMER’ tablosunun bir kopyasini cekmek istiyorsak;

>> SELECT \* INTO MusteriKopya FROM CUSTOMER

1. **WHERE**

Where komutu ile bir kosul belirleyerek sadece istedigimiz kosulu saglayan degerleri dondurebiliriz.

* Eger CUSTOMER tablosundaki musterilerden, dogum gunu ‘1944-09-12’ olan kisilerin isim ve sehir bilgilerini cekmek istersek;

>> SELECT CustomerName, City

FROM CUSTOMER

WHERE Birthdate = ‘1944-09-12’

sonuc;



1. **BETWEEN, AND, OR, NOT**

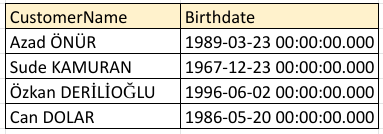
* Eger CUSTOMER tablosundaki kisilerden dogum gunu ‘1967-01-01’ ile ‘2000-01-03’ arasinda olan kisilerin isim ve dogum tarihi bilgilerini secmek istersek;

>> SELECT \*

FROM CUSTOMER

WHERE Birthdate BETWEEN ‘1967-01-01’ AND ‘2000-01-03’

sonuc;



1. **DISTINCT**

* Distinct komutu ile tabloda birden fazla olan veriler birer kez gosterilir. Ornegin CUSTOMER tablosunda ‘Kutahya’ sehri iki kere geciyorken biz distinct kullanarak secim yaparsak tekrarlayan verileri teke dusurecektir;

>> SELECT DISTINCT City

FROM CUSTOMER

sonuc;



1. **SELECT TOP**

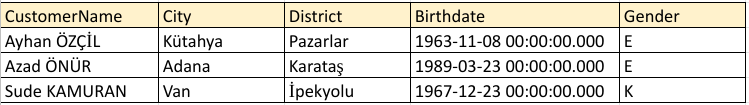
Select Top komutu ile secmek istedigimiz ilk n veriyi cekebiliriz.

* Eger CUSTOMER tablosundaki ilk 3 veriyi secmek istiyorsak;

>> SELECT TOP 3\*

FROM CUSTOMER

sonuc;



1. **INSERT INTO**

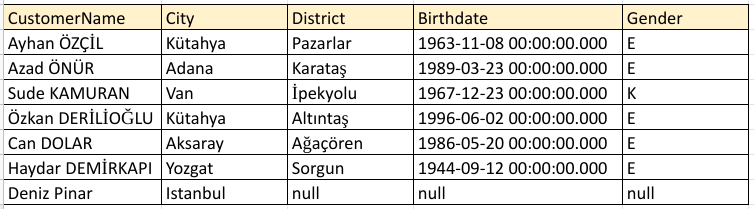
Tabloda var olan bir sutuna yeni bir veri eklemek icin INSERT INTO komutu kullanilir.

* Eger CUSTOMER tablosuna ismi ‘Deniz Pinar’ ve yasadigi sehir ‘Istanbul’ olan yeni bir kisi eklenmek istenirse;

>> INSERT INTO CUSTOMER([CustomerName], [City])

VALUES (‘Deniz Pinar’, ‘Istanbul’)

sonuc;



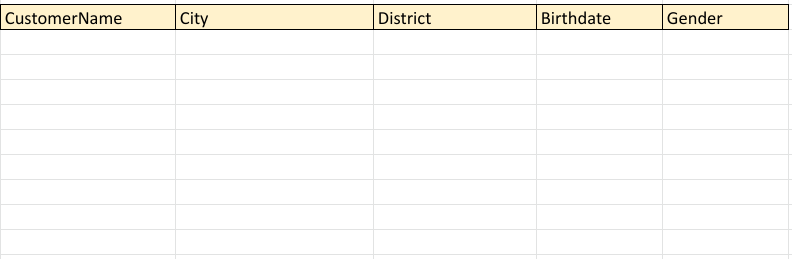
1. **TRUNCATE TABLE**

Tablonun ilk olusturuldugu halini bozmadan sadece icindeki verilerin bosaltilmasini saglar.

* Ornegin;

>> TRUNCATE TABLE CUSTOMER

sonuc;



1. **UPDATE, SET**

Veritabaninda bir duzenleme veya guncelleme yapilmasi istendiginde UPDATE komutu kullanilir.

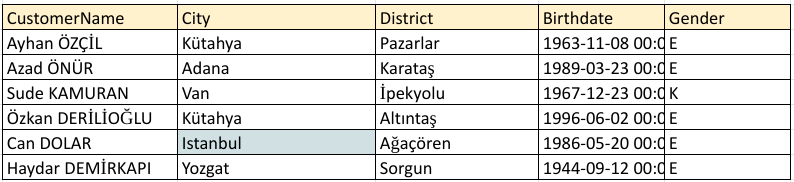
* Eger ‘Can Dolar’ in yasadigi sehri Istanbul olarak guncellemek istersek;

>> UPDATE CUSTOMER

SET City = ‘Istanbul’

WHERE CustomerName = ‘Can Dolar’

sonuc;



1. **DELETE**

Veri tabaninda bulunan belli alanlari silmek icin kullanilir. Truncate Table ile arasindaki fark ise ornegin tabloda verilen id otomatik artan bir veriyse veri delete komutuyla silinmesine ragmen aslinda tam silinmez, cunku id kaldigi yerden saymaya devam eder. Truncate table ile silindiginde ise tum veriler sifirlanir.

* Eger CUSTOMER tablosunu silmek istiyorsak;

>> DELETE FROM COSTMER

yazmamiz yeterlidir.

1. **ORDER BY**

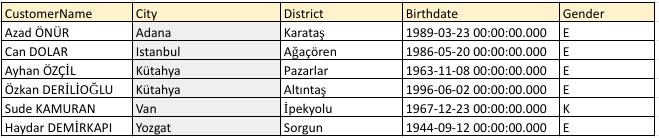
Order By ile veriyi istedigimiz sekilde siralayabiliriz.

>> SELECT \*

FROM CUSTOMER

ORDER BY City;

sonuc;



Goruldugu uzere City kendi icinde isim sirasina gore siralandi ve daha sonra bu siraya gore Customer tablosu siralandi.

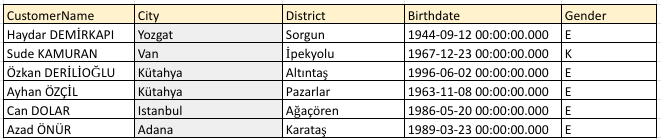
>> SELECT \*

FROM CUSTOMER

ORDER BY City DESC;

gibi yazmis olsaydik, azalan sekilde siralanirdi.

sonuc;



Eger artan sekilde siralamak isteseydik DESC yerine ASC yazardik. ORDER BY default olarak ASC calistirir.

1. **HAVING**

GROUP BY komutunu kullandigimizda genelde bir kosul belirtmek istedigimizde HAVING ifadesini kullaniriz. HAVING guplandirma islemi yapildiktan sonra kullanilirken WHERE daha gruplandirma yapilmadan calistirilmalidir. Ornegin;

>>SELECT BirthDate, Gender

FROM Customer

GROUP BY BirthDate, Gender

HAVING c4 = '1948-07-23 00:00:00.000'

sonuc;



Goruldugu uzere HAVING ifadesini GROUP BY’dan sonra yazdik. Peki HAVING yerine WHERE kullanmak isteseydik nasil yazmaliydik;

>>SELECT BirthDate, Gender

FROM Customer

WHERE c4 = '1948-07-23 00:00:00.000'

GROUP BY BirthDate, Gender

1. **SUBSTRING**

Eger bir string icerisinden belli karakterleri cekmek istersek substring komutunu kullaniriz. Bu fonksiyon 3 parametre alir, once string sonra da istenen ciktinin ilk ve son indexleri yazilir.

>> SELECT SUBSTRING(‘deniz pinar’, 1, 3)

sonuc;

‘den’

1. **CHARINDEX**

Eger bir string icerisindeki karakterin indexini dondurmek istersek charindex komutunu kullanabiliriz. Bu fonksiyon da yine 3 parametre alir, once indexini dondurmek istedigimiz karakteri, sonra bu karakteri arayacagimiz stringi ve son olarak bu string icerisinde hangi indexten baslayarak arama yapacagimizi yaziyoruz.

>> SELECT CHARINDEX(‘p’, ‘deniz pinar’, 1)

sonuc;

7

1. **CONCAT**

Stringleri birlestirmek icin kullanilir.

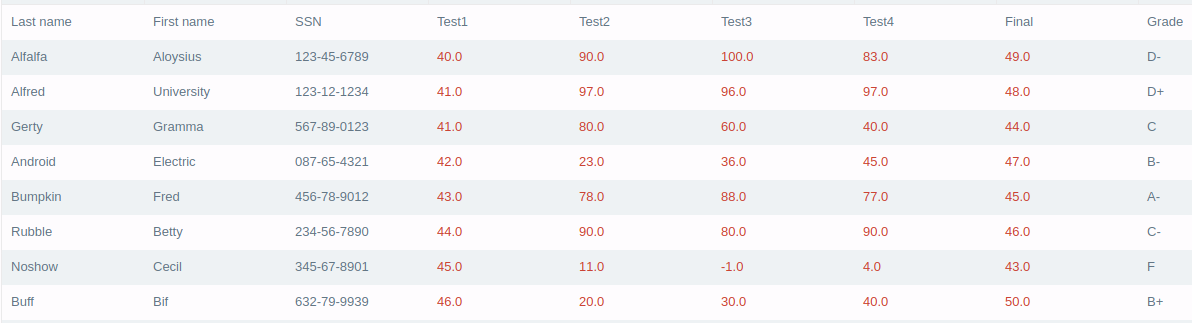
>> SELECT CONCAT(‘deniz’, ‘pinar’)

sonuc;

‘denizpinar’

1. **SUM, MIN, MAX, AVG, COUNT**

Bu fonksiyonlari asagida ornekler yaparak gosterecegim. Elimizde bir siniftaki ogrenciler ve notlarinin oldugu grades tablosu olsun.



grades tablosu

* Eger bu tablodaki tum ogrencileri aldiklari final notlarina gore siralamak istersek;

>> SELECT \*

FROM grades

ORDER BY Final

sonuc;



* Eger grades tablosundaki ogrenci sayisini bulmak istiyorsak;

>> SELECT COUNT(“First Name”)

FROM grades

sonuc;

8

* Eger Test2 sinavinda alinan en dusuk notu dondurmek isteseydik;

>> SELECT MIN(Test2)

FROM grades

sonuc;

11.0

Alinan en yuksek notu dondrumek isteseydik MIN yerine MAX yazmamiz yeterliydi.

* Eger butun ogrencilerin Test1 sinavindan aldigi notlarin ortalamasini dondurmek istiyorsak;

>>SELECT AVG(test1)

FROM grades

sonuc;

42.75

* Eger adi ‘Bif Buff’ olan ogrencinin Test1’den aldigi not ile Test2’den aldigi notun toplamini dondurmek istiyorsak;

>> SELECT SUM(test1) + SUM(test2)

FROM grades

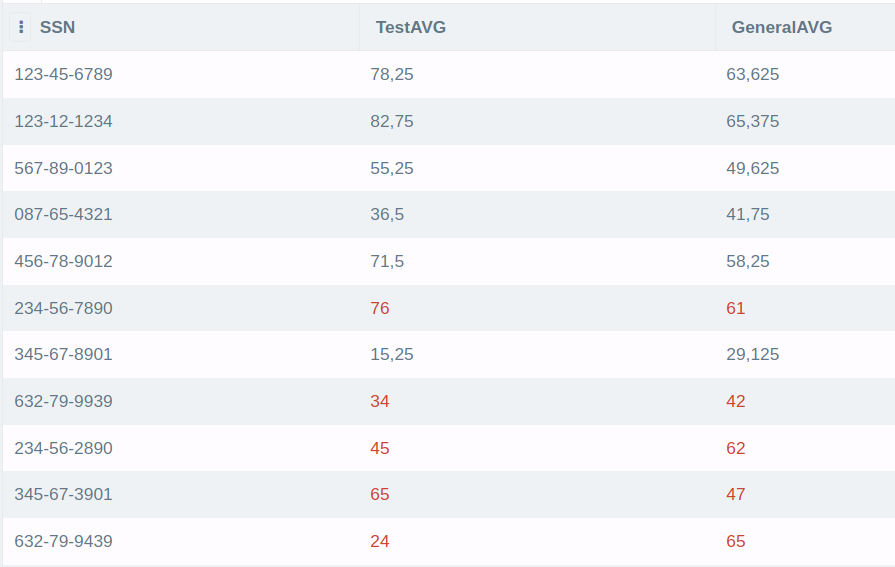
WHERE firstname = 'Bif'

sonuc;

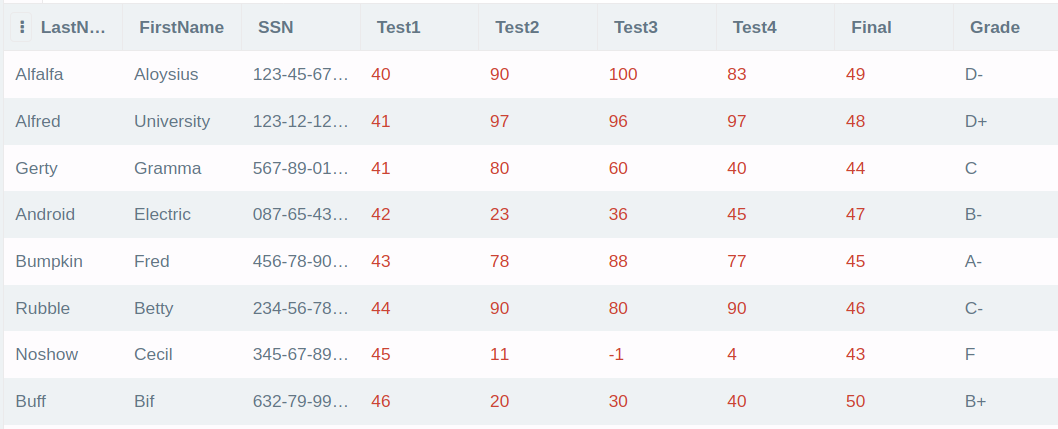
66

1. **JOIN**

Tablolari birlestirmek istedigimizde join fonksiyonlarini kullaniriz. Asagida grades ile AVG tablolari uzerinden join islemlerini gostermeye calisacagim.



AVG TABLOSU



GRADES TABLOSU

1. **INNER JOIN**

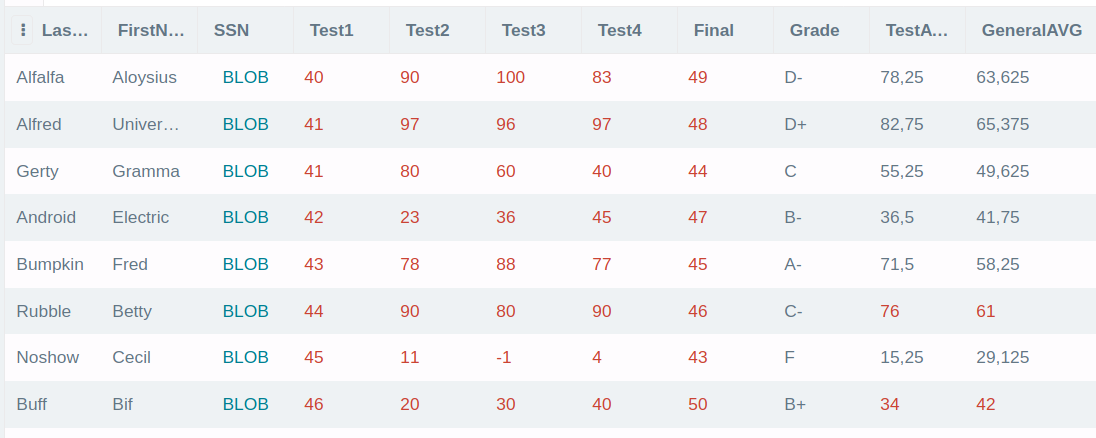
Iki tablonun kesisimi olan degerleri cekmek istedigimizde INNER JOIN kullaniriz. Ornegin ben grades tablosundaki SSN’lere gore AVG tablosundaki kesisen degerleri cekmek istersem;

>> SELECT \*

FROM GRADES

INNER JOIN AVG ON GRADES.SSN = AVG.SSN

sonuc;



Yukarida SSN’leri ortak olan tum degerler iki tablodan cekildi ve birlestirildi.

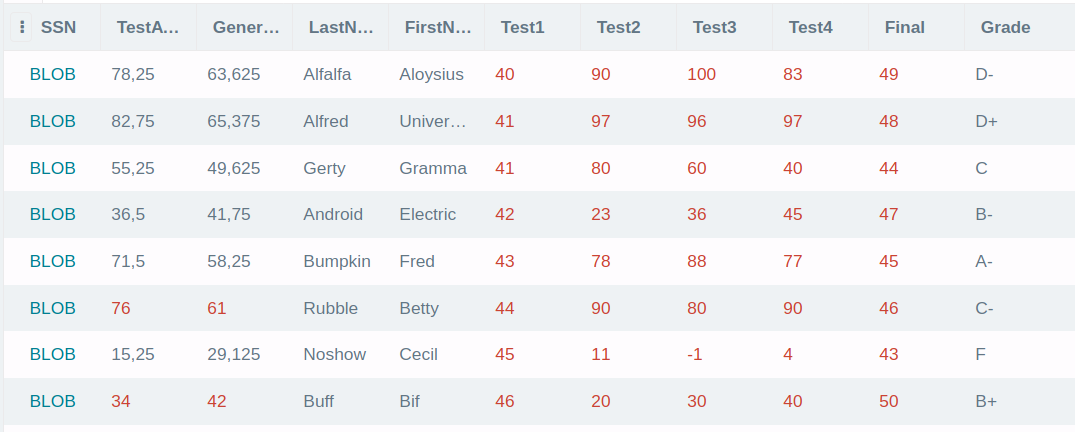
Simdi bir de AVG tablosuna SSN’leri ortak olan GRADES tablosunu ekleyelim;

>> SELECT \*

FROM AVG

INNER JOIN GRADES ON AVG.SSN = GRADES.SSN

sonuc;



Goruldugu uzere iki tabloda ortak olan SSN degerlerini kullanarak iki tabloyu birbirine birlestirdik. Iste bu ortak olan degerleri join etme terimi bize INNER JOIN’i isaret etmektedir. Kod kismina yazdigimiz INNER ifadesini kullanmadan da yine ayni islemleri yapabilirdik.

1. **LEFT JOIN**

Ornegin Grades tablosuna AVG tablosunu LEFT JOIN ile eklemek ve Test1 degerlerini dondurmek istersek, bize Grades’deki tum degerleri verirken AVG’deki sadece kesisen(Grades’de karsiligi olan) degerler eklenir.

>>SELECT AVG.TestAVG, GRADES.LastName

FROM AVG

LEFT JOIN GRADES ON GRADES.SSN = AVG.SSN

sonuc;



Goruldugu uzere AVG’deki tum degerleri aldi fakat Grades’deki karsiligi olanlari dondurdu. Karsiligi olmayanlar ise NULL seklinde geldi. Burada sol kume AVG, sag kume ise GRADES olmustur. Boylece soldaki tum degerler alinmisken, sagdaki kume icin sadece karsiligi olan degerler secilmistir.

1. **RIGHT JOIN**

Yukarida left join icin yaptigimiz islemin aynisini kumleleri yer degistirerek yaparsak o zaman da RIGHT JOIN yapmis oluruz. Yukaridaki ornegin ciktisiyla ayni ciktiyi dondurecek kodu bu sefer RIGHT JOIN ile yazalim;

>> SELECT AVG.TestAVG, GRADES.LastName

FROM GRADES

RIGHT JOIN AVG ON GRADES.SSN = AVG.SSN

Bu sefer sag kume AVG iken, sol kume GRADES oldu. Yine AVG’deki tum elemanlar geldi fakat GRADES’te karsiligi olan soyadlar cekildi. Karsiligi olmayan alanlar ise otomatikmen NULL ile dolduruldu.

1. **FULL JOIN**

Eger iki kumedeki tum degerlerin gelmesini istiyorsak yani aslinda iki kumenin birlesimini dondurmek istiyorsak FULL JOIN metodunu kullaniriz. Ornegin yukaridaki ornek ile ayni ciktiyi dondurmek icin asagidaki iki sorgu da uygundur;

>> SELECT AVG.TestAVG, GRADES.LastName

FROM GRADES

FULL JOIN AVG ON GRADES.SSN = AVG.SSN;

>> SELECT AVG.TestAVG, GRADES.LastName

FROM AVG

FULL JOIN GRADES ON GRADES.SSN = AVG.SSN;

Bu iki sorgu arasindaki tek fark sag ve sol kumlerin yerlerinin degismesidir. Fakat sonuc olarak iki sorgu da ayni sonucu vermektedir. FULL JOIN iki kume arasinda birlesim islemi yaptigi icin sag ve sol kumenin hangi kumeler oldugu onemini kaybeder ve sonuc ayni olur.

**18. TRIM**

String uzerinde bazi silme ve duzenlemeler yapmamizi saglar. Ornegin verilen stringin hem sagindaki hem de solundaki bosluklarin silinmesini istiyorsak;

>> SELECT TRIM(' deniz pinar ')

sonuc;

|  |
| --- |
| deniz pinar |

Goruldugu uzere yukaridaki stringin sag ve solundaki bosluklari temizledik ancak ortadaki bosluga dokunmadik. Ayni zamanda TRIM ile istedigimiz herhangi bir seyin stringden silinmesini saglayabiliriz. Ornegin asagida verilen stringden ‘d’ harfini cikarmak icin TRIM fonksiyonun nasil kullanilmasi gerektigi aciklanmistir;

>> SELECT TRIM('d' FROM 'deniz pinar')

sonuc;

|  |
| --- |
| eniz pinar |

Bir ornek daha, eger ‘pinar’ ifadesinin silinmesini istersek;

>> SELECT TRIM(‘pinar’ FROM ‘deniz pinar’)

sonuc;

|  |
| --- |
| deniz |

**19. LTRIM**

Bu fonksiyonla dizinin sol tarafindaki bosluklari yok edebiliriz.

>> SELECT LTRIM(‘ deniz pinar ’)

sonuc;

|  |
| --- |
| deniz pinar |

**20. RTRIM**

Bu fonksiyonla dizinin sag tarafindaki bosluklari yok edebiliriz.

>> SELECT RTRIM(‘ deniz pinar ’)

sonuc;

|  |
| --- |
| deniz pinar |

**21. LOWER**

Bu fonksiyon dizinin tum harflerini kucuk harfe cevirmeyi saglar.

>> SELECT LOWER(‘DENIZ PINAR’)

sonuc;

|  |
| --- |
| deniz pinar |

**22. UPPER**

Bu fonksiyon dizinin tum elemanlarini buyuk harfe donusturur.

>> SELECT UPPER(‘deniz pinar’)

sonuc;

|  |
| --- |
| DENIZ PINAR |

**23. REVERSE**

Bu fonksiyon ile diziyi tersten yazdirabiliriz.

>> SELECT REVERSE(‘deniz’)

sonuc;

|  |
| --- |
| zined |

**24. REPLICATE**

Verilen ifadenin istenilen sayida tekrarlanmasini saglar.

>> SELECT REPLICATE(‘la’,3)

sonuc;

|  |
| --- |
| lalala |

**25. CONCAT**

Concat ile birden fazla diziyi birlestirebiliriz.

>> SELECT CONCAT(‘DENIZ’, ‘PINAR’, ‘SQL’ , ‘OGRENIYOR’)

sonuc;

|  |
| --- |
| DENIZ PINAR SQL OGRENIYOR |

**26. CONCAT\_WS**

Peki eger verdigimiz her parametreyi birlestirirken araya istedigimiz bir karakter veya dizi koymak istersek? Iste burada CONCAT\_WS fonksiyonu imdadimiza yetisiyor.

>> SELECT CONCAT\_WS(‘\*\*’ , ‘deniz’, ‘pinar’)

sonuc;

|  |
| --- |
| deniz\*\*pinar |

**27. FORMAT**

Format ile veriyi sayi veya tarih gibi degisik turlere donusturebiliriz. 3 parametre alir fakat 2 parametreyle de kullanilabilir. İlk parametre tarih değeridir. İkinci parametre formatlama şeklini belirttiğimiz parametredir. Son olarak üçüncü parametre olarak da culture parametresini giriyoruz.

Ornegin GETDATE() ile bugunun tarihini cekelim ve istedigimiz tarih formatinda goruntuleyelim;

>> SELECT FORMAT(GETDATE(), ‘d’, ‘en-us’)

sonuc;

|  |
| --- |
| 4/18/2021 |

Simdi FORMAT fonksiyonunu iki parametre ile kullanalim;

>> SELECT FORMAT(GETDATE(), ‘d/M/y’)

sonuc;

|  |
| --- |
| 18/4/21 |

Yukarida kullandigimiz bazi kisaltmalarin ne anlama geldigine bakalim;

d = day

M = month

m = minute

y = year

Ben burada FORMAT fonksiyonu ile ilgili ozet bilgi vermeye calistim ama aslinda FORMAT fonksiyonun bircok farkli kullanimlari vardir. Dilerseniz Microsoft SQL SERVER sitesinden daha detayli bilgi alabilirsiniz.

**28. LEFT**

Eger bir string icerisinde belli bir karakterin solunda kalan kismi dondurmek istiyorsak LEFT komutunu kullanabiliriz. LEFT komutu iki parametre alir. Ilk parametresi veriyi cekmek istedigimiz ana dizi, ikinci parametre ise index tir. Bu index’in solunda kalan kisim cekilecektir. Ornegin;

>> SELECT LEFT(‘deniz pinar’, 5)

sonuc;

|  |
| --- |
| deniz |

>> SELECT LEFT(‘deniz pinar’, CHARINDEX(‘ ‘, ‘deniz pinar’))

Yukaridaki kod ile de ayni sonucu dondurebiliriz. CHARINDEX ile aslinda ‘deniz’ ile ‘pinar’ arasindaki boslugun indexini dondurmus olduk. LEFT ile de bu indexin solundaki ifadeyi dondurmus olduk.

**29. RIGHT**

Eger bir string icerisinde belli bir karakterin saginda kalan kismi dondurmek istiyorsak RIGHT komutunu kullanabiliriz. RIGHT komutu iki parametre alir. Ilk parametresi veriyi cekmek istedigimiz ana dizi, ikinci parametre ise index tir. Bu index’in saginda kalan kisim cekilecektir. Ornegin;

>> SELECT RIGTH(‘deniz pinar’, 6)

sonuc;

|  |
| --- |
| pinar |

>> SELECT RIGHT(‘deniz pinar’, LEN(‘deniz pinar’) - CHARINDEX(‘ ‘, ‘deniz pinar’))

Yukaridaki kod ile de ayni sonucu dondurebilirdik. Eger indexi bilmiyor ve programin bulmasini istiyorsak bu yontem isi cozmektedir.

**30. UPDATE**

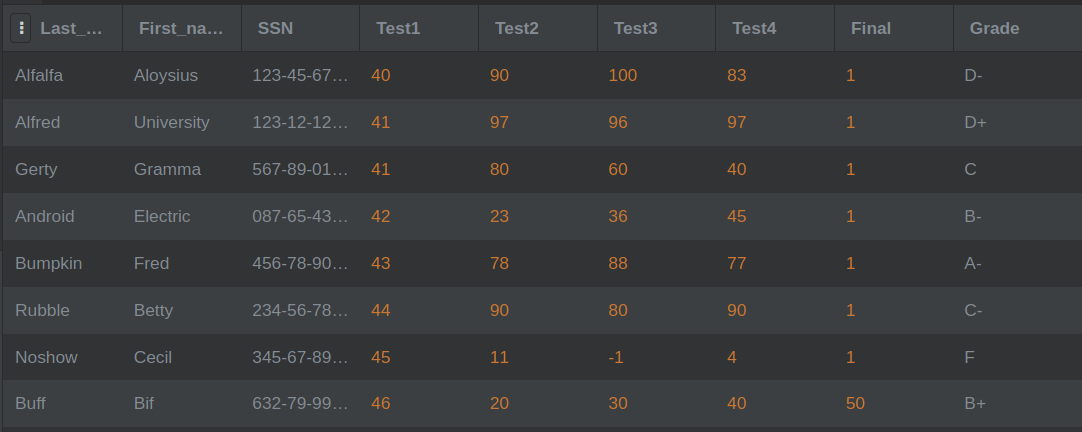
Veri tabanindaki herhangi bir veriyi guncellemek veya degistirmek icin update komutunu kullaniriz. Eger butun tabloyu guncellemek istemiyorsak UPDATE komutunu genelde UPDATE ... SET ... WHERE ... seklinde kullaniriz. UPDATE’ ten sonra guncelleme yapmak istedigimiz tablo ismini, SET’ten sonra tablo icerisinde nasil bir guncelleme yapmak istedigimizi, WHERE’den sonra ise hangi kosullarda bu guncellemeyi yapmak istedigimizi belirtiyoruz.

Ornegin GRADES tablosundaki final notlarinin 50’nin altinda olanlari 1 olacak sekilde guncelleyelim;

>> UPDATE GRADES SET Final = 1 WHERE Final < 50;

SELECT \* FROM GRADES

sonuc;



**31. DECLARE / SET**

Declare ile degisken tanimlayabilir, SET ile de bu degiskene bir deger atayabiliriz. Declare’den sonra basina ‘@’ koyarak degisken ismini, sonra da bu degiskenin tipini tanimlariz. MSSQL’de degiskenler baslarina ‘@’ isareti konarak ifade edilirler. Ornegin ‘cumle’ diye bir degisken olusturup bunu ‘SQL Ogreniyorum’ seklinde bir dizi olarak tanimlayalim;

>> DECLARE @cumle AS nvarchar(11)

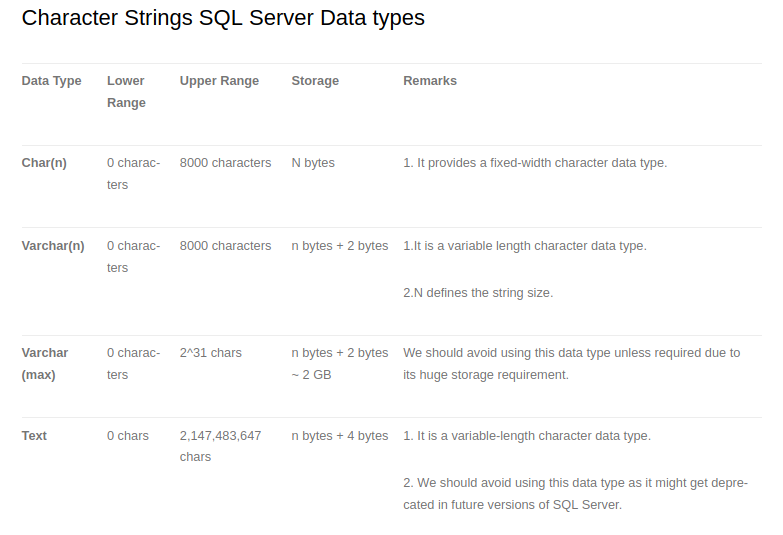
SET @cumle='deniz pinar'

SELECT @cumle

**sonuc;**

|  |
| --- |
| deniz pinar |

Yukarida nvarchar veri tipini kullanarak islem yaptik. nvarchar, girilen veri kadar yer kaplayan, en fazla 4000 karakter veri tutan, max depolama boyutu ise 2 GB olan bir veri tipidir. SQL dilinde kullanilan bir cok veri turu vardir. Asagida dizi veri tipleri ornek olarak verilmistir;



kaynak= <https://www.sqlshack.com/an-overview-of-sql-server-data-types/>

**32. CREATE TABLE**

Veri tabani icerisinde yeni bir tablo olusturmak istersek bu komutu kullaniriz. Ornegin ‘ogrenciler’ diye bir tablo olusturalim ve bu tablonun icerisinde ogrencilerin okul numaralarinin bulundugu ‘ogrenciNo’ ve ogrencilerin isimlerinin bulundugu ‘ogrenciAdi’ sutunlari olsun.

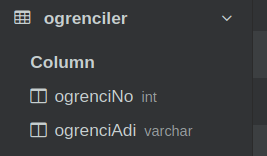
>> CREATE TABLE ogrenciler(

ogrenciNo int,

ogrenciAdi varchar(50)

)

sonuc;



CREATE TABLE komutundan sonra olusturmak istedigimiz tablo adini girdik, daha sonra parantez acarak eklemek istedigimiz sutun isimlerini ve veri tiplerini girdik.

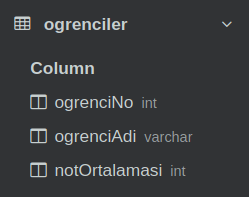
Goruldugu uzere veri tabaninda tablo olusturuldu ama ici bos bir tablo olusturuldu. Eger icerisine veri eklemek isterseniz daha once madde 7’de anlattigim kismi incelemenizi tavsiye ederim. INSERT INTO fonksiyonunu kullanarak tabloya veri ekleyebilirsiniz.

**33. ALTER TABLE**

Veri tabaninda bulunan herhangi bir tablonun ozelliklerini degistirmek istersek bu fonksiyonu kullaniriz. Ornegin yukarida olusturdugumuz ogrenciler tablosuna ‘notOrtalamasi’ adinda bir sutun daha eklemek istersek;

>> ALTER TABLE ogrenciler ADD notOrtalamasi int;

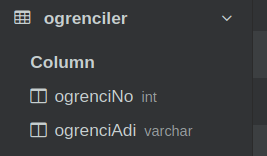
sonuc;



ALTER TABLE ile tablolarda istedigimiz degisikligi yapabilecegimizi soylemistik. Ornegin herhangi bir sutunu nasil sileriz?

>> ALTER TABLE ogrenciler DROP COLUMN notOrtalamasi;

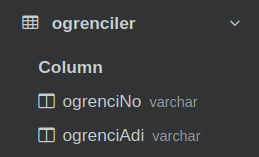
sonuc;



Peki var olan bir sutunun veri tipini degistirmek istersek;

>> ALTER TABLE ogrenciler ALTER COLUMN ogrenciNo varchar;

sonuc;



Peki ‘ogrenciNo’ nun primary key oldugunu varsayalim ve buraya mutlak bir degerin girilmesi gerektigi sartini koyalim;

>> ALTER TABLE ogrenciler ALTER COLUMN ogrenciNo int IS NOT NULL;

**34. DROP TABLE**

Tablo silmek icin bu fonksiyonu kullaniriz.

>> DROP TABLE Grades

seklinde yazarsak Grades tablosunu tamamen kaldirmis oluruz.

>> DROP TABLE IF EXISTS Grades

seklinde yazarsak Grades tablosu yoksa bile hata almadan eger Grades tablosu varsa bu komutla silinir.

**35. CREATE INDEX**