**ORACLE SQL DÖKÜMAN VE NOTLARI**

Oracle komutlarının Türkçe açıklamaları ve örnekleri. Başucunuzda bulunması gereken Türkçe döküman.

**(+)**

(+), Birleştirme yapılan tablolardan ikinci tabloda birinci tablodaki her kaydın karşılığı olmazsa, karşılığı olmayan kayıtlar sql sonucunda sadece olmayan alanlar değil bilakis kayıt hiç gelmez. Bunun önlemi dış birleştirmedir. Dış birleştirme işlemi, kayıtları eksik olan tablonun şart tarafına "(+)" işareti konularak yapılır.

SELECT \* FROM PERSONEL, UNVAN WHERE PERSONEL.UNVANKEY(+) = UNVAN.UN\_KEY

Bu örnekte, 3 tane kayıt gelmektedir, yani UNVAN tablosunda sadece UN\_KEY=1 olan sadece bir kayıt var.

Diğer tablo da ise (PERSONEL) UNVANKEY'i birinci tabloda ki UN\_KEY=1' e eşit olan kayıtlar sorgu sonucu gelir. Not: Biz null değere sahip olanları da birleştirmek istersek eksik olan tablonun yanına (+) işareti eklenir.

PERSONEL.PR\_KEY PERSONEL.SICIL PERSONEL.UNVANKEY UNVAN.UN\_AD UNVAN.SERVISKEY

1 XI 1 6

50 1 1 1

54 1212541 1 1

**ABS**

ABS, N sayısının pozitif halini sonuç olarak döndürür.

SELECT ABS(-1), ABS(1) FROM HASTA

Bu örnekte olduğu gibi bir sayıyı (pozitif yada negatif) pozitif bir sayıya çevirir.

ABS(-1) ABS(1)

1 1

1 1

1 1

**COS**

COS(N), N sayısının kosinüsünü sonuç olarak döndürür.

SELECT COS(-1), COS(0), COS(1) FROM HASTA

Bu örnekte olduğu gibi bir sayıyı (pozitif yada negatif) kosinüsüne çevirir.

ACOS(1) ACOS(0) ACOS(-1)

1 1 1

1 1 1

1 1 1

**ACOS**

ACOS(N), N sayısının ark kosinüsünü sonuç olarak döndürür.

SELECT ACOS(-1), ACOS(0), ACOS(1) FROM HASTA

Bu örnekte olduğu gibi bir sayıyı (pozitif yada negatif) ark kosinüsüne çevirir.

ACOS(-1) ACOS(0) ACOS(1)

3 2

3 2

3 2

**ADD\_MONTHS**

ADD\_MONTHS, ADD\_MONTHS(t,n) t tarihini, n ay eklenmis olarak sonuçta döndürür.

SELECT ADD\_MONTHS(TO\_DATE('31.01.2001'),1) FROM HASTA

Bu örnekte olduğu gibi t (31.01.2001) tarihine n (1) ay ekleyerek sonuca yansıtır.

ADD\_MONTHS(TO\_DATE('31.01.2001')

28.02.2001

28.02.2001

28.02.2001

**ALL**

ALL, Tablolarda ki bütün alanların listeleneceğini (çift olsa dahi) gösterir.

SELECT HS\_KEY, HS\_AD, CINSIYET, ISLEMSAYI FROM HASTA WHERE ISLEMSAYI>ALL(SELECT DISTINCT HS\_KEY FROM HASTA WHERE TELEKOMBOLUM='BASIN YAYIN')

All komutu ile bu örnekte basın yayın (BASIN YAYIN) bölümünde çalışan her hastadan daha fazla işlem sayısı (ISLEMSAYI) olan hastanın hs\_ad, hs\_soyad, cinsiyet, işlemsayi alanları listelenir. Not: Bu örnekteki " her hasta " ifadesi bize ALL kullanılacağını gösterir.

HS\_KEY HS\_AD CINSIYET ISLEMSAYI

93 ORHAN E 13

42017 TÜRKAN K 4

42029 ÖZLEM K 5

**ALL\_OBJECTS**

ALL\_OBJECTS, Kullanıcının erişebileceği bütün nesneler hakkında bilgi içeren görüntüdür.

**ALTER**

ALTER, Tabloya sütün / tablo kısıtlaması ekleme komutudur. Tabloya yeni sütun ekleme komutudur. Tablonun kayıt parametrelerini değiştirme komutudur. Bir kısıtlamayı açma/kapama komutudur. Tablonun üzerinde ki bütün tetiklemeleri açma/kapama komutudur. Tablo ya kayıt girmeye müsaade etme/etmeme komutudur. Tablonun paralellik derecesini değiştirme komutudur.

ALTER TABLE HASTA ADD (ADI CHAR(20), SOYAD CHAR(20))

Bu örnekte, hasta tablosu içerisine ADI ve SOYADI karakter (char) tipteki alanlar eklenmiştir. Bu işlemden sonra SELECT HASTA.\*, ROWID FROM HASTA komutunu yazarak gelen tabloda ki ADI ve SOYADI alanlarına ADI alanına (ÖZLEM,HÜLYA,ZEYNEP) SOYADI alanına (YILDIZ,AK,MERT) verilerini alt alta girdikten sonra SELECT \* FROM HASTA dediğimizde aşağıdaki tablo karşımıza çıkacaktır.

HS\_AD HS\_SOYAD CINSIYET ADI SOYAD

ORHAN AYAZ E ÖZLEM YILDIZ

TÜRKAN AYAZ K HÜLYA AK

ÖZLEM AYAZ K ZEYNEP MERT

ALTER TABLE HASTA MODIFIY HS\_AD VARCHAR2(40)

Bu örnekte de hasta tablosunun bir alanının uzunluğu 40 olarak değiştiriliyor. Modify komutu bir alanın uzunluğunu (karakter sayısını) değiştirir. Hasta tablosunda ki HS\_AD alanının uzunluğunun VARCHAR2(40) olarak değiştiğini DESC HASTA komutunu çalıştırarak görebiliriz.

SIRA ALAN ADI ALAN TIPI BOŞ

1 HS\_KEY NUMBER NOT NULL

2 HS\_AD VARCHAR2(40)

3 HS\_SOYAD VARCHAR2(18)

4 ISLEMSAYI NUMBER

5 DOGUMTARIHI DATE

6 DOGUMYERI VARCHAR2(15)

**ANALYZE**

ANALYZE, Tablolar ve indekslerle ilgili istatistik toplayan DLL komutudur.

**AND**

AND, Where bölümünde birden fazla şart yazıldığı zaman, aralarındaki ilişkiye göre kullanılan ifadedir. And her iki şartın aynı anda doğruluğunun gerektiği durumlarda kullanılır.

SELECT \* FROM HASTA WHERE HS\_KEY>20 AND HS\_KEY<40

Bu örnek ile hasta numarası(HS\_KEY) 20 ile 40 arasında olan tüm alanlar listelenmektedir.

HS\_KEY HS\_AD HS\_SOYAD CINSIYET ISLEMSAYI DOGUMYER DOGUMTARIHI

21 ELAHATTIN İÇER E

25 MUHARREM BEŞIR E 1

29 VURAL CENGİZ E

33 ALI TURGAY ERKAN E 1

37 KAZIM GÜÇLÜ E 1

**ANY**

ANY, Alt sorgulamadan seçilen değerlerden en az biri karsılaştırılır.

SELECT \* FROM HASTA WHERE HS\_KEY=ANY(93,2005,14005)

Any komutu ile hasta tablosu içerisinde hasta numarası (HS\_KEY) olan ve any komutu içerisine yazılan değerlerden en az birini karşılaştırarak sonuca yansıtır.

HS\_KEY HS\_AD HS\_SOYAD CINSIYET ISLEMSAYI DOGUMYERI DOGUMTARIHI

93 AHMET AYAZ E 14

2005 SEMA SARIKAYA K 1

14005 VİLDAN TURHAL K

**AS**

AS, Bir tablodan, içindeki kayıtlarla birlikte sadece balirtilen alanları alarak bir başka tablo oluşturma komutudur. Bir sütünun adını değiştirme komutudur.

SELECT HS\_KEY AS HASTANUMARASI FROM HASTA

As, komutu ile hs\_key olan alanın adını HASTANUMARASI olarak değiştirilebilir.

HASTANUMARASI

5

9

13

17

**ASC**

Sorgu sonucu dönen kayıtlarda sıralamayı küçükten büyüye doğru yapmak için kullanılan ifadedir.

SELECT \* FROM HASTA ORDER BY HS\_KEY ASC

Hasta tablosunda ki hasta numaralarını(HS\_KEY) küçükten büyüye doğru sıralamaktadır.

HS\_KEY HS\_AD HS\_SOYAD CINSIYET ISLEMSAYI DOGUMYER DOGUMTARIHI

5 I.HAKKI ALPTÜRK E 3

9 VEDAT KARAARSLAN E 2

13 ZAFER TEKBUDAK E 1

17 CELALETTİN DINÇER E 1

**ASCII**

ASCII, Bir karakteri yada cümleyi ascıı sayıya çevirir.

SELECT ASCII('COZUM'), ASCII('BILGISAYAR') FROM HASTA

Ascıı komutu ile yazılan karakter yada cümleyi ascıı karakterine çevirerek sonuca yansıtır.

ASCII(COZUM) ASCII(BILGISAYAR)

67 66

67 66

67 66

**ASIN**

ASIN(n) n sayısının ark sinüsünü sonuç olarak döndürür.

SELECT ASIN(-1), ASIN(0), ASIN(1) FROM HASTA

Bu örnekte olduğu gibi bir sayıyı (pozitif yada negatif) ark sinüsünü sonuç olarak yansıtır.

ASIN(-1) ASIN(0) ASIN(1)

-2 2

-2 2

-2 2

**ASSOCIATE**

**ATAN**

ATAN(n) n sayısının ark tanjantını sonuç olarak döndürür.

SELECT ATAN (.781285627), TAN(663225116) FROM HASTA

Bu örnekte olduğu gibi bir sayıyı (pozitif yada negatif) ark tanjantını sonuç olarak yansıtır.

ATAN (.781285627 TAN(663225116)

1 1

1 1

1 1

**ATAN2**

SELECT ATAN2(.8,1), ATAN(.9) FROM HASTA

ATAN2(.8,1) ATAN(.9)

1 1

1 1

1 1

**AUDIT**

AUDIT, Veritabanı nesneleri hakkında kontrol işlemleri yapar ve Sql' leri seçerek tanımlama komutudur.

**AVG**

AVG, bütün değerlerinin ortalamasını döndürür.

SELECT AVG(ISLEMSAYI) FROM HASTA

AVG komutu ile hasta tablosundaki bütün kayıtların ortalama işlem sayısını (ISLEMSAYI) hesaplar.

AVG(ISLEMSAYI)

2

**BETWEEN**

BETWEEN, İki değer arasındaki kayıtları belirler.

SELECT \* FROM HASTA WHERE HS\_KEY BETWEEN 1 AND 20

Between komutu ile hasta numarası (hs\_key) 1 ile 20 arasında olan kayıtları listeler.

HS\_KEY HS\_AD HS\_SOYAD CINSIYET ISLEMSAYI DOGUMYER DOGUMTARIHI

5 I.HAKKI ALPTÜRK E 3

9 VEDAT KARAARSLAN E 2

13 ZAFER TEKBUDAK E 1

17 CELALETTİN DINÇER E 1

**BFILENAME**

Declare admin\_photo bfile; Begin dmin\_photo:=BFILENAME('('/home/oracle','ADMINISTRATOR\_PHOTO.JPG'); end;

**BLOB**

BLOB, Oracle veritabanında temel kayıt ünitesidir.

**BY**

BY komutu, Order By ve Group By komutları ile birlikte kullanılmaktadır.

SELECT \* FROM HASTA ORDER BY HS\_KEY

HS\_KEY HS\_AD HS\_SOYAD CINSIYET ISLEMSAYI DOGUMYER DOGUMTARIHI

5 I.HAKKI ALPTÜRK E 3

9 VEDAT KARAARSLAN E 2

13 ZAFER TEKBUDAK E 1

17 CELALETTİN DINÇER E 1

**CALL**

CALL, Database de ki yazılan bir proceduru yada fonksiyonu çağırmak için kullanılır.

**CEIL**

CEIL(n) n sayısından büyük mevcut en küçük tam sayıyı sonuç olarak döndürür.

SELECT CEIL(10), CEIL(10.5), CEIL(-10.5) FROM HASTA

CEIL komutu verilen pozitif yada negatif sayıyı sonuç olarak en küçük sayıyı tam sayıya çevirir.

CEIL(10) CEIL(10.5) CEIL(-10.5

10 11 -11

10 11 -11

10 11 -11

**CHARTTOROWID**

CHARTTOROWID(n) Karkater olan n sayısını ROWID değere çevirir ve sonuç olarak da bu değeri döndürür.

SELECT \* FROM HASTA WHERE ROWID=CHARROWID(")

**CHR**

CHR, Bir sayının karakter halini sonuç olarak gösterir.

SELECT CHR(70), CHR(80), CHR(90), CHR(120) FROM HASTA

Bu örnekte olduğu gibi CHR komutu bir sayıyı karakter haline çevirerek sonuca yansıtılmıştır.

CHR(70) CHR(80) CHR(90) CHR(120)

F P Z X

F P Z X

F P Z X

F P Z X

**CLUSTER**

CLUSTER, Bir kaydın fiziksel yerinin kayıdın içindeki değere bağlı olarak değişen bir tablo yapısı çeşididir.

**COMMENT**

COMMENT, Tablo,sütun,görüntü ve snapshot hakkında veri sözlüğüne yorum yazma komutudur.

COMMENT on table hasta coloumn(hs\_ad) is 'ÖZLEM'

**COMMIT**

COMMIT, Bütün yapılan İşlemleri kesin olarak kalıcı olmasını sağlar. Böylece yapılan değişiklikleri varsa diğer kullanıcılarda görür.

**CONCAT**

CONCAT, Concat(d1,d2) ile d1 dizesini (string), devamına d2 dizesi eklenmiş olarak sonuçta döndürür.

SELECT CONCAT('ÇÖZÜM',' BİLGİSAYAR') AS SIRKET FROM HASTA

Bu örnekte olduğu gibi CONCAT komutu ile birinci string' in yanına ikinci stringi ekleyerek bu iki kelimeyi sonuca yansıtır. AS komutu ile de bu birleştirilmiş olan 2 stringin alan adı (SİRKET) belirlenmiş olur.

SIRKET

ÇÖZÜM BİLGİSAYAR

ÇÖZÜM BİLGİSAYAR

ÇÖZÜM BİLGİSAYAR

**CONSTRAINT**

CONSTRAINT, O kolon için daha açıklayıcı olması açısından ve yapısı açısından kullanılır. Oracle veritabanı tablolarında veri bütünlüğünün sağlanması için veritabanı düzeyinde konan sınırlamalardır. Null/Not Null , Unique, Primary Key, Froeign Key ve Check olmak üzere 5 çeşit kısıtlama vardır.

**CONTROLFILE**

**CONVERT**

CONVERT, Oracle versiyon 6 'ya ait veri sözlüğünün oracle versiyon 7'ye çevrilmesinin belirtildiği bölümdür.

SELECT CONVERT(CHR(194), 'US7ASCII', 'WE8EBCDIC37C') FROM HASTA

Bu örnekte CONVERT komutu ile Oracle versiyon 6'ya ait sayı yada karakteri Oracle 7' ye çevrilmesi sağlar.

CONVERT(CHR(194),US7ASCII','W

B

B

B

**COS**

COS(n) n sayısının kosinüsünü sonuç olarak döndürür.

SELECT COS(0), COS(-1), COS(1) FROM HASTA

Bu örnekte COS komutu ile bir sayıyı (pozitif yada negatif) kosinüsüne çevirerek sonuç olarak yansıtır.

COS(0) COS(-1) COS(1)

1 1 1

1 1 1

1 1 1

**COUNT**

COUNT(\*,sütün) sorgudaki şartlara uyan kayıt sayılarını döndürür.

SELECT COUNT(\*) FROM HASTA

Bu örnekte olduğu gibi COUNT komutu ile hasta tablosunda ki toplam kayıt sayısını bulabiliriz.

COUNT(\*)

20,997

**CREATE**

CREATE, Tablo, sütun, procedür, index yaratmak için kullanılır.

CREATE TABLE COZUM(ADI CHAR(20), SOYADI CHAR(20), CINSIYETI CHAR(5))

Bu komutu yazdığımızda COZUM adında ADI, SOYADI, CINSIYETI alanları yaratılmış olucaktır. Daha sonra INSERT INTO COZUM VALUES ('ÖMER', 'SISO', 'E') yazarak oluşturmuş olduğumuz alanlara bilgi girdikten sonra SELECT \* FROM COZUM yazdığımızda aşağıda ki tablo karşımıza çıkacaktır.

ADI SOYADI CINSIYETI

ÖMER SISO E

**DATABASE**

DATABASE, dataların tutulduğu taban.

**DATE**

DATE, Tarih tutan alanlar için kullanılır.

CREATE TABLE PERS (ADI CHAR(20), SOYADI CHAR(20), TARIH DATE)

Komutunu yazdıktan sonra içerisine 2 adet veri girişi yapalım. INSERT INTO PERS VALUES('HAKAN', 'MERT', '01.02.2000') bu komutu çalıştırdıktan sonra şimdi ikinci veri girişini yapalım. INSERT INTO PERS('CANAN', 'TERS', '02.11.2001') bu komutu da çalıştırdığımız da aşağıdaki tablo karşımıza çıkacaktır.

ADI SOYADI TARIH

HAKAN MERT 01.02.2000

CANAN TERS 02.11.2001

**DAY**

DAY , Gün tutan alanlar için kullanılır.

**DBA\_OBJECTS**

DBA\_OBJECTS, Veritabanındaki bütün nesleler hakkında bilgi içeren görüntüdür.

**DBA\_TRIGGERS**

DBA\_TRIGGERS, Veritabanındaki bütün tetiklemeler hakkında bilgi içeren görüntüdür.

**DECODE**

DECODE, Kodlanmış bilgileri açıklamalrıyla listelemek amacıyla kullanılır.

SELECT HS\_AD, HS\_SOYAD, DECODE(CINSIYET, 'E', 'ERKEK', 'K', 'KADIN') AS CINSIYET FROM HASTA

HS\_AD HS\_SOYAD CINSIYET

AHMET AKMAN ERKEK

VELI YILMAZ ERKEK

AYSE GÜN KADIN

**DELETE**

DELETE, Herhangi bir table yada sütünu silmek için kullanılır.

DELETE FROM ABIOTIK

Not: Lütfen bu örneği denemeyiniz. ABIOTIK tablosunu silersiniz , denediyseniz rollback komutunu yazarak silmiş olduğunuz tabloyu geri kurtarabilirsiniz.

**DESC**

DESC, Sorgu sonucu dönen kayıtları sıralamayı büyükten küçüğe doğru yapmak için kullanılan ifadedir. Tabloların sütünlarını listelemektedir.

SELECT \* FROM HASTA WHERE HS\_KEY DESC

HS\_KEY HS\_AD HS\_SOYAD CINSIYET ISLEMSAYI DOGUMYER DOGUMTARIHI

5 I.HAKKI ALPTÜRK E 3

9 VEDAT KARAARSLAN E 2

13 ZAFER TEKBUDAK E 1

17 CELALETTİN DINÇER E 1

**DESCRIBE**

DESCRIBE, Tabloların fieldlarınını ve yapısını listelemek amacıyla kullanılır.

DESCRIBE HASTA

SIRA ALAN ADI ALAN TIPI BOŞ

1 HS\_KEY NUMBER NOT NULL

2 HS\_AD VARCHAR2(40)

3 HS\_SOYAD VARCHAR2(18)

4 ISLEMSAYI NUMBER

5 DOGUMTARIHI DATE

6 DOGUMYERI VARCHAR2(15)

**DICT**

SQL\*PLUS' ta çalışırken, DICTINORY veye eş anlamı olan DICT görüntüsünden bütün veri sözlüğü görüntüleri hakkında bilgi alınabilmektedir.Aşağıdaki örnekte DICT görüntüsünün alanlarını (yapısını) listeleyelim.

DESC DICT

SIRA ALAN ADI ALAN TIPI BOS

1 TABLE\_NAME VARCHAR2(30)

2 COMMENTS VARCHAR2(4000)

DICT\_COLUMNS

**DICTIONARY**

SQL\*PLUS' ta çalışırken, DICTINARY veye eş anlamı olan DICT görüntüsünden bütün veri sözlüğü görüntüleri hakkında bilgi alınabilmektedir.Aşağıdaki örnekte DICT görüntüsünün alanlarını (yapısını) listeleyelim.

DESC DICTINARY

SIRA ALAN ADI ALAN TIPI BOS

1 TABLE\_NAME VARCHAR2(30)

2 COMMENTS VARCHAR2(4000)

**DIMENSION**

**DIRECTORY**

**DISASSOCIATE**

**DISTINCT**

DISTINCT, Kayıt yinelenmesini önlemek için kullanılan ifadedir. Hasta tablosu içerisinde aynı kayıt iki defa yada daha fazla girildiyse dahi DISTINCT komutu kayıt yinelenmesini önleyerek teke indirerek tabloda gösterilmesini sağlar.

SELECT DISTINCT HS\_KEY FROM HASTA

HS\_KEY HS\_AD HS\_SOYAD CINSIYET ISLEMSAYI DOGUMYER DOGUMTARIHI

5 I.HAKKI ALPTÜRK E 3

9 VEDAT KARAARSLAN E 2

13 ZAFER TEKBUDAK E 1

17 CELALETTİN DINÇER E 1

**DROP**

DROP, table yada sütun silme komutudur.

ALTER TABLE DROP ADRES

Not: Lütfen bu örneyi denemeyiniz. Hasta tablosu içerisinden adres fieldını silersiniz. Denediyseniz rollback komutu ile son yaptığınız işlemi geri alabilirsiniz.

**DUAL**

DUAL, Sql' de kullanılan fonksiyon veya değerler, eğer herhangi bir tablodan çağrılmıyacaksa, bu durumlarda SQL formatının yanlış yazılmış olmaması için FROM bölümünden sonra standart olarak DUAL yazılır.

**DUMP**

SELECT DUMP(SYSDATE) FROM HASTA

DUMP(SYSDATE)

Typ=13 Len=8: 209,7,11,6,13,57,3,0

Typ=13 Len=8: 209,7,11,6,13,57,3,0

Typ=13 Len=8: 209,7,11,6,13,57,3,0

**EMPTY\_BLOB**

CREATE TABLE HAK(ADI BLOB) Bu komut zinciri ile ilk önce HAK adında sadece ADI alanı blob olan bir table yarattık. Daha sonra bu tablo da ADI adlı alana bilgi girelim. INSERT INTO HAK(ADI) VALUES(EMPTY\_BLOB()) Şimdide SELECT \* FROM HAK komutunu çalıştırdığımızda aşağıda ki tabloyu elde ederiz.

ADI

(OraBlob)

**EMPTY\_CLOB**

CREATE TABLE HASTANE (ADI CLOB) Bu komut zinciri ile ilk önce HASTANE adında sadece ADI alanı Clob olan bir table yarattık. Daha sonra bu tablo da ADI adlı alana bilgi girelim. INSERT INTO HASTANE(ADI) VALUES(EMPTY\_CLOB()) Şimdide SELECT \* FROM HASTANE komutunu çalıştırdığımızda aşağıda ki tabloyu elde ederiz.

ADI

(OraClob)

**EXISTS**

"Var, mevcuttur" anlamındaki bu sözcük, SQL' de Boolean (lojik, mantıksal) operatörüdür. İçteki SELECT komutunun sorgulanması sonucunda, en az bir tablo satırı üretilmişse,EXISTS operatörü true (doğru) değerini, hiçbir tablo satırı üretilmemişse, EXISTS operatörü (yalnış) değerini üretir. Not: EXISTS operatörü, AND, OR ve NOT gibi diğer mantıksal ifadelelerle de kullanılır. Aşağıda ki örnekte HASTA tablosunda EXISTS içerisine yazılan ifadeyi sağladığı için true değerini döndürmüş ve aşağıdaki tabloyu oluşturmuştur.

SELECT \* FROM HASTA WHERE EXISTS(SELECT HS\_AD FROM HASTA)

HS\_KEY HS\_AD HS\_SOYAD CINSIYET ISLEMSAYI DOGUMYER DOGUMTARIHI

5 I.HAKKI ALPTÜRK E 3

9 VEDAT KARAARSLAN E 2

13 ZAFER TEKBUDAK E 1

17 CELALETTİN DINÇER E 1

**EXP**

EXP(n) e' nin n' inci üssünü sonuç olarak döndürür. (e=2.718)

SELECT EXP(3) FROM HASTA

EXP(3)

20

20

20

**EXPLAIN PLAN**

EXPLAIN PLAN, Tablosunu ya utlxpin.sql'ini çalıştırarak ya da create table cümlesi ile kendiniz yaratabilirsiniz.

**EXTRACT**

**FLOOR**

FLOOR, FLOOR(n) n sayısından küçük veya n sayısına eşit mevcut en büyük tam sayıyı sonuç olarak döndürür. Bu örnekte 9.9 ve -9.9 sayılarını FLOOR komutu ile 9.9' u 9'a ve -9.9'u da -10' a (kendisinden küçük en yakın tam sayıya) çevirir.

SELECT FLOOR(9.9), FLOOR(-9.9) FROM HASTA

FLOOR(9.9) FLOOR(-9.9)

9 -10

9 -10

9 -10

**FOR**

For döngüleri özellikle belirli bir sayıda yapılan döngüler için kullanılan döngü tipidir. Döngünün kaç defa olacağını baş tarafında ki sayılar belirlemektedir.

**FROM**

FROM, Kullanılacak tabloların yazıldığı bölümdür.

SELECT \* FROM HASTA

HS\_KEY HS\_AD HS\_SOYAD CINSIYET ISLEMSAYI DOGUMYER DOGUMTARIHI

5 I.HAKKI ALPTÜRK E 3

9 VEDAT KARAARSLAN E 2

13 ZAFER TEKBUDAK E 1

17 CELALETTİN DINÇER E 1

**FUNCTION**

FUNCTION, Bir fonksiyon bir değer hesaplayan alt programdır. Fonksiyon ve prosedür yapıları RETURN anahtar kelimesi haricinde benzerdir.

**GRANT**

GRANT, Sistem hakları ,rolleri ve nesne haklarını bir kullanıcı veya rolden hak alarak geri alma komutudur.

**GREATEST**

GREATEST, Belirtilen sayı yada gün yada tarih içerisinden en büyüyünü bulur.

SELECT GREATEST(93,42017,2005) AS ENBUYUKSAYI FROM HASTA

GREATEST içerisine yazılan sayılardan en büyük rakkamı, tarihi yada mantıksal ifadeyi bularak sonuca yazdırır.Burada ki AS komutu ile de yzdığımız bu sayılar arasında ki en büyük sayıyı bularak bu alanın adını ENBUYUKSAYI olarak değiştirir.

ENBUYUKSAYI

42017

42017

42017

SELECT GREATEST('ONE','TWO') AS SAYI FROM HASTA

SAYI

TWO

TWO

SELECT GREATEST(TO\_DATE('11/11/2000','mm/dd/yyyy'), TO\_DATE('12/12/2001')) AS ENBUTUKTARIH FROM HASTA

ENBUYUKTARIH

12/12/2001

12/12/2001

12/12/2001

**GROUP**

GROUP, Sorgu sonucu dönen kayıtları belli özelliklerine göre gruplama işleminin yapıldığı bölümdür.

SELECT AVG(HS\_KEY) FROM HASTA GROUP BY HS\_KEY

Bu örnek ile Hasta tablosu içerisinde hasta numaralarına(HS\_KEY) göre gruplama yaparak, herbir gurubun ortalamasını (AVG) bulur.

HS\_KEY HS\_AD HS\_SOYAD CINSIYET ISLEMSAYI DOGUMYER DOGUMTARIHI

5 I.HAKKI ALPTÜRK E 3

9 VEDAT KARAARSLAN E 2

13 ZAFER TEKBUDAK E 1

17 CELALETTİN DINÇER E 1

**GROUPING**

GROUPING, Group by dan sonra kullanılır.

SELECT HS\_KEY, HS\_AD, COUNT(\*), GROUPING(HS\_KEY) FROM HASTA GROUP BY ROLLUP(HS\_AD, HS\_KEY)

HS\_KEY HS\_AD COUNT(\*) GROUPING(HS\_KEY)

87625 AYŞE 1

AYŞE 1

125757 BİRSEN 1

BİRSEN 1

**HAVING**

GROUP BY kullanıldığı SQL' lerdeki grubu ilgilendiren şartların yazıldığı bölümdür. Burada grup içerisinde ki şartlar belirlenir. Bu örnekte hasta numarasına(HS\_KEY) göre guruplama yapılmış. Ve HAVING komutu ile her bir group içerisinde ki hasta numarası(HS\_KEY) 8' den küçük olan numaraların ortalaması alınmıştır.

SELECT AVG(HS\_KEY) FROM HASTA GROUP BY HS\_KEY HAVING AVG(HS\_KEY)<8

AVG(HS\_KEY)

5

**HEXTORAW**

SELECT RAWTOHEX(HEXTORAW('12341312314151')) FROM HASTA

RAWTOHEX(HEXTORAW('19'))

19

19

**IN**

IN(liste) Liste' nin içindeki herhangi bir değeri sağlayan kayıtlar.

SELECT \* FROM HASTA WHERE HS\_KEY IN(SELECT MAX(HS\_KEY) FROM HASTA)

Hasta tablosu içerisinde hasta numarası (HS\_KEY) en büyük olan kaydın tüm alanlarını(bilgilerini) getirir.

HS\_KEY HS\_AD HS\_SOYAD CINSIYET ISLEMSAYI DOGUMYER DOGUMTARIHI

125853 SELMA ZENGİN K 1 ANKARA 11.11.1976

**INDEX**

Tablolarda ki kayıtlara daha hızlı erişebilmek için kullanılan nesnelerdir.

**INITCAP**

INITCAP(d) d dizesinin ilk harfi büyük diğer harfleri küçük olarak sonuçta döndürür.

SELECT INITCAP(HS\_AD) FROM HASTA

Bu örnek ile hastanın adı (HS\_AD) alanında ki bilgilerin baş harfini büyük diğer harflerini küçük olarak değiştirir.

INITCAP(HS\_AD)

Ahmet

Kaan

Yiğit

**INSERT**

Tablo'ya bilgi girişi yapmak için kullanılan bir komuttur.

CREATE TABLE COZUM(ADI CHAR(20), SOYADI CHAR(20))

Yukarıda COZUM adında ADI ve SOYADI alanları olan bir tablo yarattık.Şimdide yaratmış olduğumuz bu tabloya 1 satır bilgi girişi yapalım.

INSERT INTO COZUM VALUES('ÖZLEM','YILDIZ')

Daha sonrada SELECT \* FROM diyerek yaratmış olduğumuz tabloyu görelim.

ADI SOYADI

ÖZLEM YILDIZ

**INSTR**

INSTR(d1,d2,[m,n]) d1 dizesinin içerisinde ,d2 dizesini d12in m'inci karakterden başlamak üzere n'inci defaki tekrarını arar ve sonuçta bulduğu karakter sayısını döndürür eğer belirtilmezse m ve n start olarak 1' dir.

SELECT HS\_AD, INSTR(HS\_AD, 'A') FROM HASTA

Hasta tablosu içerisinde hastanın adı(HS\_AD) adlı alanda A harfini arar ve A harfi kaçıncı karakterden itibaren başlarsa INSTR ile A harfinin başladığı konumunu yazar.

HS\_AD INSTR(HS\_AD,'A')

AYSE 1

OZLEM

KAAN 2

OSMAN 4

OMER

**INSTRB**

SELECT INSTRB(HS\_AD,'A') FROM HASTA

HS\_AD INSTR(HS\_AD,'A')

AYSE 1

OZLEM

KAAN 2

OSMAN 4

OMER

**INTERSECT**

İki tane ayrı sql sonucu dönen kayıtların kesişim kümesini sonuca yansıtır.

**INTO**

INTO kelimesi INSERT kelimesinden sonra kullanılır. Tabloda ki bir satıra bilgi girişi yapabilmek için kullanılır. İlk önce bir tablo yaratalım daha sonrada INTO kelimesini kullanarak yaratmış olduğumuz bu tablo ya bir satır veri girişi yapalım.

CREATE TABLE MUSTER(MUSTERIADI CHAR(20), MUSTERISOYADICHAR(20))

(Müşteri adında MUSTERI NO ve MUSTERI ADI adlı alanları olan bir tablo yarattık.)

INSERT INTO MUSTER VALUES('DERYA', 'KOÇ')

(Şimdi de yaratmış olduğumuz bu tablo ya bir satır bilgi girdik.Aşağıda yazmış olduğumuz kodu çalıştırarak sonuç tablosunu görebiliriz.

SELECT \* FROM MUSTER

MUSTERIADI MUSTERISOYADI

DERYA KOÇ

**LAST\_DAY**

LAST\_DAY(t) t tarihinin içerisinde bulunduğu ayın son gününü sonuç olarak döndürür.

SELECT LAST\_DAY(TO\_DATE('01.01.2001')) FROM HASTA

LAST\_DAY(TO\_DATE('01.01.2001')

31.01.2001

**LENGTH**

LENGTH(d1) d1 dizesinin boyutunu sonuç olarak döndürür.

SELECT LENGTH('COZUM') FROM HASTA

LENGTH('COZUM')

5

**LENGTHB**

SELECT LENGTHB('COZUM') FROM HASTA

LENGTH('COZUM')

5

**LIKE**

LIKE, Şarta bir bölümü uyan kayıtları listeler. Benzerlik vermek için kullanılır. '%' karakteri tüm karakterler yerine geçer. '\_' karakteri ise tek karakter yerine geçer. Sadece \_ işaretini çıkarmak için ' \_ ' işareti kullanılır. Küçük büyük harf ayrımı vardır.

HASTA tablosu içerisinde ki hastanın adı(HS\_AD) alanında ki adı A ile başlayanları listeleyiniz.

SELECT \* FROM HASTA WHERE HS\_AD LIKE 'A%'

HS\_KEY HS\_AD HS\_SOYAD CINSIYET ISLEMSAYI DOGUMYER DOGUMTARIHI

137 ATİLLA BAYBAŞ E 3

149 ALPER YASEMİN E 2

241 APDULRAHMAN BAY E 2

261 AYNUR KOÇ K 1

HASTA tablosu içerisinde ki hastanın adı(HS\_AD) alanında ki adının son harfi A olanları listeleyiniz.

SELECT \* FROM HASTA WHERE HS\_AD LIKE '%A'

HS\_KEY HS\_AD HS\_SOYAD CINSIYET ISLEMSAYI DOGUMYER DOGUMTARIHI

2005 SEMA SARIKAYA K 3

1005 DOĞA AMİKLİOĞLU K 2

20005 MEHLİKA ÖĞÜT K 2

8005 SÜREYYA TEKİNCAN K 1

HASTA tablosu içerisinde ki hastanın adı(HS\_AD) alanında başı ve sonu belli olmayan içerisinde A harfi bulunan kayıtları listeleyiniz.

SELECT \* FROM HASTA WHERE HS\_AD LIKE '%A%'

HS\_KEY HS\_AD HS\_SOYAD CINSIYET ISLEMSAYI DOGUMYER DOGUMTARIHI

2005 KAAN MERT E 3

1005 SEMA CAN K 1

20005 SİNAN YILDIZ E 1

8005 İHSAN TEKİNCAN E 1

**LINK**

**LN**

LN(n) n sayısının doğal logaritmasını sonuç olarak döndürür.

SELECT LN(120) FROM HASTA

LN(120)

5

**LOG**

LOG(m,n) n sayısının m tabanına göre logaritmasını sonuç olarak döndürür.

SELECT LOG(10,10000) FROM HASTA

LOG(10,10000)

4

**LOWER**

LOWER(d) d dizesinin bütün harflerini küçük olarak sonuçta döndürür. Bu örnekte HASTA tablosunda hastanın adı(HS\_AD) adlı alanda tüm bilgiler(satırlar) küçük harfe çevrilerek sonuca yazdırıldı.

SELECT LOWER(HS\_AD) FROM HASTA

HS\_KEY HS\_AD HS\_SOYAD CINSIYET ISLEMSAYI DOGUMYER DOGUMTARIHI

200 ali KANBUR E 3

1000 mert SERT K 1

2003 yasemin YILMAZ K 1

8005 seda TEKİNCAN K 1

**LPAD**

LPAD(d1,n,d2) d1 dizisini, n karakter oluncaya kadar soldan d2 eklenmiş olarak sonuçta döndürür.

SELECT LPAD('L',3,'P') FROM HASTA

LPAD('L',3,'P')

LPP

LPP

LPP

**LTRIM**

LTRIM(d1[d2])d1 dizisinin baş tarafından d2 dizesinde olan karakterler çıkartılmış olarak sonuçta döndürür, eğer belirtilmezse standart olarak d2 boşluktur. Bu örnekte HASTA tablosu içerisinde hastanın adı(HS\_AD) adlı alanda baş harfi A ile başlayan adların baş harfini çıkartarak sonuca yansıtılmıştır.

SELECT HS\_KEY, HS\_AD, CINSIYET, LTRIM(HS\_AD,'A') FROM HASTA

HS\_KEY HS\_AD CINSIYET LTRIM(HS\_AD,'A')

137 ATİLLA E TİLLA

149 ALPER E LPER

241 APDULRAHMAN E PDURRAHMAN

261 SERAP K ERAP

**MATERIALIZED**

**MAX**

MAX(Sütün) Sütün değerlerinin en büyüyünü döndürür. Burada DISTINCT komutu ile tekrarlanan kayıtlar teke düşürerek tüm kayıtlar arasında maximum kaydı bulur. ALL komutu ile de tekrarlanmış kayıtlar olsa dahi tüm kayıtlar arasında maximum en büyük hasta numarasını(HS\_KEY) bulur.

SELECT MAX(HS\_KEY), MAX(DISTINCT HS\_KEY), MAX(ALL HS\_KEY) FROM HASTA

MAX(HS\_KEY) MAX(DISTINCT HS\_KEY) MAX(ALL HS\_KEY)

125853 125853 125853

**MIN**

MIN(Sütün) Sütün değerlerinin en küçüğünü döndürür. Burada DISTINCT komutu ile tekrarlanan kayıtlar teke düşürerek tüm kayıtlar arasında minumum kaydı bulur. ALL komutu ile de tekrarlanmış kayıtlar olsa dahi tüm kayıtlar arasında minumum en küçük hasta numarasını(HS\_KEY) bulur.

SELECT MIN(HS\_KEY), MIN(DISTINCT HS\_KEY), MIN(ALL HS\_KEY) FROM HASTA

MIN(HS\_KEY) MIN(DISTINCT HS\_KEY) MIN(ALL HS\_KEY)

5 5 5

**MINUS**

MINUS, Birinci sql sonucu dönen kayıtlarla ikinci sql sonucu dönen kayıtlar arasında ki fark kümesi.

**MOD**

MOD(m,n) m sayısının n sayısına bölümünden kalanı sonuç olarak döndürür.

SELECT MOD(18,12) FROM HASTA

MOD(18,12)

6

6

**MONTH**

MONTH, tarih kullanımında ay anlamındadır.

**MONTHS\_BETWEEN**

MONTHS\_BETWEEN(t1,t2) t1 ve t2 tarihleri arasındaki ay sayısını sonuç olarak döndürür.

SELECT MONTHS\_BETWEEN(TO\_DATE('09-09-2001'), TO\_DATE('01-02-2001')) FROM HASTA

MONTHS\_BETWEEN(TO\_DATE('09-09-2001'), TO\_DATE('01-02-2001'))

7

7

7

**NEW\_TIME**

SELECT NEW\_TIME(TO\_DATE('8-MAY-2000 8:00 AM'), 'PST','EST') FROM HASTA

**NEXT\_DAY**

NEXT\_DAY(t,d) t tarihinden sonra ki d isimli ilk günü sonuç olarak döndürür. NEXT\_DAY(tarih,' gün' ) -->tarih 'den sonraki günün ilk tarihini verir.

SELECT NEXT\_DAY('01-MAY-2001' ,'SATURDAY') FROM HASTA

NEXT\_DAY('01-MAY-2001' ,'SATURDAY')

07-MAY-2001

**NLS\_CHARSET\_DECL\_LEN**

SELECT NLS\_CHARSET\_DECL\_LEN (100,NLS\_CHARSET\_ID('US7ASCII')) FROM HASTA

NLS\_CHARSET\_DECL\_LEN (100,NLS\_CHARSET\_ID('US7ASCII'))

100

100

**NLS\_CHARSET\_ID**

SELECT NLS\_CHARSET\_ID('US7ASCII'), NLS\_CHARSET\_ID('WE8EBCDIC37C') FROM HASTA

NLS\_CHARSET\_ID('US7ASCII') NLS\_CHARSET\_ID('WE8EBCDIC37C')

1 90

1 90

**NLS\_CHARSET\_NAME**

SELECT NLS\_CHARSET\_NAME(1),NLS\_CHARSET\_NAME(90) FROM HASTA

NLS\_CHARSET\_NAME(1) NLS\_CHARSET\_NAME(90)

US7ASCII WE8EBCDIC37C

US7ASCII WE8EBCDIC37C

**NLS\_INITCAP**

SELECT NLS\_INITCAP('COZUM', 'NLS\_SORT=XFRENCH') FROM HASTA

NLS\_INITCAP('COZUM', 'NLS\_SORT=XFRENCH')

Cozum

Cozum

Cozum

**NLS\_LOWER**

SELECT NLS\_LOWER('COZUM', 'NLS\_SORT=XFRENCH') FROM HASTA

NLS\_LOWER('COZUM', 'NLS\_SORT=XFRENCH')

cozum

cozum

cozum

**NLS\_UPPER**

SELECT NLS\_UPPER('COZUM', 'NLS\_SORT=XFRENCH') FROM HASTA

NLS\_UPPER('COZUM', 'NLS\_SORT=XFRENCH')

COZUM

COZUM

COZUM

**NLSSORT**

SELECT NLSSORT('OZLEM','NLS\_SORT=XFRENCH') FROM HASTA

NLSSORT('OZLEM','NLS\_SORT=XFRENCH')

5A874B285000010101010100

5A874B285000010101010100

**NOAUDIT**

Kontrol işlemini geri alma komutudur.

**NOT**

NOT' ın listenin içerisinde ki herhangi bir değeri sağlayan kayıtlar haricindeki kayıtlar yada not like şarta bir bölümü uyan kayıtlar haricinde ki kayıtlar gerektiğine not' ın yada not like komutları ile kullanılır. Aşağıda ki örnekte HASTA tablosunda hastanın adı(HS\_AD) A ile başlamayan kayıtlar listelenecektir.

SELECT HS\_KEY, HS\_AD FROM HASTA WHERE HS\_AD NOT LIKE 'A%'

HS\_KEY HS\_AD

12 SEMA

138 MERT

152 YASEMİN

**NULL**

NULL, IS NULL(boş olan kayıtlar) yada IS NOT NULL (boş olmayan kayıtlar) için kullanılan bir komuttur. Bu örnekte HASTA tablosu içerisinde hastanın adı(HS\_AD) boş olan kayıtlar listelenmiştir.

SELECT HS\_KEY, HS\_AD, CINSIYET, ISLEMSAYI FROM HASTA WHERE HS\_AD IS NULL

HS\_KEY HS\_AD CINSIYET ISLEMSAYI

42017 E 1

42029 E 2

32005 K 1

**NUMBER**

NUMBER, Nümerik (sayısal) dataların tutulduğu alanlar için kullanılır. Number(m,n) : m kadar (max e38) Sayının n kadar ondalık alan için değer alır.

**NUMERIC**

**NVL**

NVL, Null değeri yerine yeni değer atar.Alan türü ne türde ise alacağı değerde o türdedir. Bu örnekte HASTA tablosu içerisinde ki null (boş) olan hastanın numarası(HS\_KEY) adlı alana 0 değeri atanmıştır.

SELECT NVL(HS\_KEY,0) FROM HASTA

NVL(HS\_KEY,0)

12

23

0

**OR**

OR, Where bölümünde birden fazla şart yazıldığı zaman, aralarındaki ilişkiye göre kullanılan ifadedir. Or her iki şartın en az birinin doğruluğunun yettiği durumlarda kullanılır.

SELECT HS\_KEY, HS\_AD, CINSIYET, ISLEMSAYI FROM HASTA WHERE HS\_KEY=8 OR HS\_KEY=12

HS\_KEY HS\_AD CINSIYET ISLEMSAYI

8 OZCAN E 2

12 KADİR E 1

**ORDER**

ORDER, Bir tablo da belli bir alana göre sıralama yapılmasını sağlar. Bu örnekte HASTA tablosunda hastanın numarasına (HS\_KEY) göre küçükten büyüye göre listeler.

SELECT \* FROM HASTA ORDER BY HS\_KEY

HS\_KEY HS\_AD HS\_SOYAD CINSIYET ISLEMSAYI DOGUMYER DOGUMTARIHI

1 ali KANBUR E 3

2 mert SERT K 1

3 yasemin YILMAZ K 1

4 seda TEKİNCAN K 1

**PACKAGE**

**POWER**

POWER(m,n) m üssü n sayısını sonuç olarak döndürür.

SELECT POWER(10,3), POWER(-10,-3), POWER(-10,3) FROM HASTA

POWER(10,3) POWER(-10,-3) POWER(-10,3)

1000 -1000

1000 -1000

1000 -1000

**PROCEDURE**

Bir prosedürün iki kısmı vardır: tanımlama ve gövde. Tanımlama kısmı PROCEDURE kelimesi ile başlar ve prosedür adı ya da parametre listesi ile biter. Parametre tanımlamaları zorunlu değildir. Parametre kullanılmayan prosedürler parantez kullanmadan yazılabilirler. Prosedürün gövde kısmı IS anahtar kelimesi ile başlar ve END anahtar kelimesi ile biter. Prosedürün gövdesi de üç kısma ayrılır: değişkenlerin tanımlandığı kısım, komut cümlelerinin yazıldığı kısım ve hata durumlarının kontrol edildiği kısım. Değişken tanımlama kısmı IS kelimesinden hemen sonra başlar. Burada DECLARE kelimesi kullanılmaz. Komut cümlelerinin yazıldığı kısım ise BEGIN anahtar kelimesi ile başlar ve EXCEPTION ya da END ile biter. Bu kısımda en az bir komut yazılmalıdır. Hata durumları kısmı zorunlu değildir. Prosedür END kelimesi ile son bulur. Bu anahtar kelimenin yanına prosedür ismi yazılabilir, zorunlu değildir.

**PROFILE**

**RAWTOHEX**

SELECT RAWTOHEX(HEXTORAW('19')) FROM HASTA

RAWTOHEX(HEXTORAW('19'))

19

19

19

**RENAME**

RENAME, Bir nesnenin ismini değiştirme komutudur.

RENAME HASTA TO HASTANE

**REPLACE**

REPLACE(d1,d2,d3) d1 dizesini ,içinde geçen d2 dizelerini d3 ile değişmiş olarak sonuçta döndürür.

SELECT REPLACE('COZUM BİLGİSAYAR HASTA TAKİP','HASTA','PERSONEL') FROM HASTA

REPLACE('COZUM BİLGİSAYAR HASTA TAKİP','HASTA','PERSONEL')

COZUM BİLGİSAYAR PERSONEL TAKİP

COZUM BİLGİSAYAR PERSONEL TAKİP

COZUM BİLGİSAYAR PERSONEL TAKİP

**RESOURCE COST**

**REVOKE**

REVOKE, Sistem hakları, rolleri ve nesne haklarını bir kullanıcı veya rolden hak olarak geri alma komutudur.

**ROLE**

ROLE, Veri tabanında ki hakların toplanmış haline denmektedir. Rollerle, DBA(Database Administrator,Veritabenı Yöneticisi) işini daha kolay yapabilmektedir. Roller kullanılarak veritabanının güvenliği de bir derece artırılmış olmaktadır.

**ROLLBACK**

ROLLBACK, Bütün yapılan İşlemleri kesin olarak iptal eder. Select, Update, İnsert, Delete. Vb işlemleri yedekler ROLLBACK komutunu çalıştırdığımızda ise tüm yapmış olduğu işlemleri(update, delete, vb) geri alır.

ROLLBACK

**ROUND**

ROUND, (tarih[,' fmt '] ) -->tarih 'i belirtilen formata göre aya veya yıla göre yuvarlar.

SELECT ROUND(123.45), ROUND(123.45,1), ROUND(123.45,-1) FROM HASTA

ROUND(123.45) ROUND(123.45,1) ROUND(123.45,-1)

123 124 120

123 124 120

123 124 120

**ROWID**

ROWID, Bir kayıdın tekil (unigue) adresini tutan alanlar için kullanılır. Hasta tablosu içerisine alanlara direk bilgi girişi yapılmasını sağlar.

SELECT HASTA.\*, ROWID FROM HASTA

HS\_KEY HS\_AD HS\_SOYAD CINSIYET ISLEMSAYI DOGUMYER DOGUMTARIHI

5 I.HAKKI ALPTÜRK E 3

9 VEDAT KARAARSLAN E 2

13 ZAFER TEKBUDAK E 1

17 CELALETTİN DINÇER E 1

**ROWIDTOCHAR**

SELECT ROWID,ROWIDTOCHAR(ROWID) from hasta

**RPAD**

RPAD(d1,n,d2) d1 dizesini ,n karakter oluncaya kadar sağdan d2 eklenmiş olarak sonuçta döndürür.

SELECT RPAD('A',3), RPAD('B',3,'S'), RPAD('C',8,'K') FROM HASTA

RPAD('A',3) RPAD('B',3,'S') RPAD('C',8,'K')

A BSS CKKKKKKK

A BSS CKKKKKKK

A BSS CKKKKKKK

**RTRIM**

RTRIM, RTRIM(d1,[d2] ) d1 dizesini son tarafından d2 dizesinde olan karakterler eklenmiş olarak sonuçta döndürür.

SELECT RTRIM ('COZUM BİLGİSAYAR') FROM HASTA

RTRIM ('COZUM BİLGİSAYAR')

COZUM BİLGİSAYAR

COZUM BİLGİSAYAR

COZUM BİLGİSAYAR

**SAVEPOINT**

SAVEPOINT, İşlemi belirli bir yere yönlendirmek için kullanılır. X ile belirtilen alan için işaret konularak istenildiğinde bu işarete kadar işlemler yapılabilir.

**SCHEMA**

**SEGMENT**

SEGMENT, Belirli bir mantıksal yapı için ayrılmış extent 'lerin kümesidir. (data segment, index segment, rollback segment, temporary segment gibi).

**SELECT**

SELECT, En azından bir sütun isminin yazıldığı ve seçildiği bölümdür.

SELECT \* FROM HASTA

HS\_KEY HS\_AD HS\_SOYAD CINSIYET ISLEMSAYI DOGUMYER DOGUMTARIHI

5 I.HAKKI ALPTÜRK E 3

9 VEDAT KARAARSLAN E 2

13 ZAFER TEKBUDAK E 1

17 CELALETTİN DINÇER E 1

**SEQUENCE**

SEQUENCE, Yaptığı is unique sayılar üretmektir. Belli oranlarda arttırmalar yapılmakta kullanılır. Her çağrıldığında yeni bir sayı üretir. Ekstra bir hesaplama yapılmadan, seri olarak tanımlandığı şekilde rakamlar üretir. Sayıları cacheden okuduğu için çok hızlı sonuç üretir.

**SESSION**

**SET**

SET, Trusted oracle 7'de çalışmaktadır. DBHIGH, DBLOW, DBMAC ON, DBMS MAC, DBMAC OFF, MAC ile ilgili ayarlamalar yapılmaktadır.

**SIGN**

SIGN(n) Signum fonksiyonu (0 ' dan küçük sayılar için -1, 0 sayısı için 0 ve sıfırdan byük sayılar için de 1 değerini sonuç olarak döndürür.

SELECT SIGN(76),SIGN(0),SIGN(-76.17) FROM HASTA

SIGN(92) SIGN(0) SIGN(-92.15)

1 0 -1

1 0 -1

1 0 -1

**SIN**

SIN(n) n sayısının sinüsünü sonuç olarak döndürür.

SELECT SIN(180),SIN(90)FROM HASTA

SIN(180) SIN(90)

-1 1

-1 1

-1 1

**SNAPSHOT**

SNAPSHOT, veritabanında ki kayıtla ilgili parametrelerin belirtildiği bölümdür.

**SOME**

Some komutu ile işlem sayısı(ISLEMSAYI) 2 olan "herhangi bir" hastadan daha büyük olan hastanın HS\_KEY, HS\_AD, CINSIYET alanlarının listelenmesi aşağıdaki örnekte sağlanmıştır.

SELECT HS\_KEY, HS\_AD, CINSIYET FROM HASTA WHERE HS\_KEY> SOME(SELECT DISTINCT HS\_KEY FROM HASTA WHERE ISLEMSAYI=2)

HS\_KEY HS\_AD CINSIYET

5 I.HAKKI E

9 VEDAT E

13 ZAFER E

17 CELALETTİN E

**SOUNDEX**

**SQRT**

SQRT(n) n sayısının karakökünü sonuç olarak döndürür.

SELECT SQRT(25) AS KARAKOKU FROM HASTA

KARAKOKU

5

**STATISTICS**

STATISTICS, alınan ihracın ithal edilirken gerekli optimizasyonları yapmasının belirtildiği bölümdür.

**STDDEV**

SELECT STDDEV(HS\_KEY) FROM HASTA

STDDEV(HS\_KEY)

30362

**SBSTR**

SUBSTR(d1,m,n) d1 dizesinin m'inci karakterden başlayarak n karakterlik bölümünü sonuç olarak döndürür.

SELECT SUBSTR ('CozumBilgisayar',6) FROM HASTA

SUBSTR ('CozumBilgisayar',6)

Bilgisayar

Bilgisayar

**SUBSTRB**

SELECT SUBSTR('cozum',4) FROM HASTA

SUBSTR('cozum',4)

um

um

**SUM**

SUM(sütün) sütün değerlerinin toplamını döndürür.

SELECT SUM(HS\_KEY) FROM HASTA

SUM(HS\_KEY)

1574896781

**SYNONYM**

Bir objenin aynısınin kopyasının alarak kendi üzerinde yaratır.

**SYS\_CONTEXT**

SELECT SYS\_CONTEXT('USERENV', 'NLS\_SORT') FROM HASTA

SYS\_CONTEXT('USERENV', 'NLS\_SORT')

TURKISH

TURKISH

**SYS\_GUID**

SELECT SYS\_GUID() FROM HASTA

SYS\_GUID()

D6B89F3822624FFCB6213049FAC41DDB

AE90B8067D044A348D8D8F2E3D5819ED

**SYSDATE**

O anki tarih ve saati sonuç olarak döndürür.

SELECT HS\_KEY, HS\_AD, SYSDATE FROM HASTA

HS\_KEY HS\_AD SYSDATE

5 I.HAKKI 08.11.2001 13:08:29

9 VEDAT 08.11.2001 13:08:29

13 ZAFER 08.11.2001 13:08:29

17 CELALETTİN 08.11.2001 13:08:29

**SYSTEM**

Sistem kullanıcısıdır.Veri sözlüğünün hepsini kullanma hakkına sahiptir.Önemli nesneleri (tablespace,user,role.vb.) yaratma hakkına standart olarak (yani sistem hakları ile sonradan verilmeden) sahiptir.

**TABLE**

Oracle veritabanında verileri saklamak için kullanılan temel birimdir. Çizelgeler satırlar (rows) ve sütunlar (columns) olarak tutulurlar. Her çizelge, adı ve sütun kümesi (nitelik) ile tanımlanır. Her sütunun bir adı, türü ve genişlik ya da duyarlılığı verilir. Çizelge bir kez yaratıldıktan sonra içine geçerli satırlar konulabilir ve daha sonra da bu çizelgenin satırları sorgulanabilir, silinebilir ya da günlenebilir.

**TABLESPACE**

Oracle 'ın tabloyu yaratacağı tablo boşluğunu belirler. Eğer tanımlanmazsa çalışılmakta olan kullanıcının tablo boşluğunda yaratılır.

**TAN**

TAN(n) n sayısının tanjantını sonuç olarak döndürür.

SELECT TAN(90), TAN(180) FROM HASTA

TAN(90) TAN(180)

-2 1

-2 1

**TANH**

TANH(n) n sayısının hiperbolik tanjantını sonuç olarak döndürür.

SELECT TANH(0), TANH(180) FROM HASTA

TANH(90) TANH(180)

1 1

1 1

**TEMPORARY**

Tablespace'i geçici (içindeki bilgiler kalıcı olmayan ) hale getirme bölümüdür.

**TO\_CHAR**

TO\_CHAR(n) nümerik bir değere sahip n sayısını karakter tipe çevirir ve sonuç olarak da bu diziye döndürür.

SELECT TO\_CHAR(sysdate,' DD-MM-YYYY') FROM HASTA

TO\_CHAR(sysdate,' DD-MM-YYYY')

08-11-2001

08-11-2001

**TO\_DATE**

TO\_DATE(d,f) Krakter tipindeki d dizesini f formatında ki bir tarih değerine çevirir ve sonuç olarak da bu tarihi döndürür.

SELECT HS\_KEY, HS\_AD FROM HASTA WHERE DOGUMTARIH=TO\_DATE('EYLÜL 1, 2001', 'Month dd,YYYY')

HS\_KEY HS\_AD

125377 E.BURÇİN

**TO\_LOB**

CREATE table lob\_table(x clob); create table long\_table(x long); insert into long\_table(x) values('COZUM') ; insert into lob\_table(x) select to\_lob(x) from long\_table

**TO\_MULTI\_BYTE**

SELECT TO\_MULTI\_BYTE('COZUM') FROM HASTA

TO\_MULTI\_BYTE('COZUM')

COZUM

COZUM

**TO\_NUMBER**

TO\_NUMBER(d) Karkter tipinde ki d dizesini numerik bir değere çevirir ve sonuç olarak bu değeri döndürür.

**TO\_SINGLE\_BYTE**

SELECT TO\_SINGLE\_BYTE('COZUM BILGISAYAR') FROM HASTA

TO\_SINGLE\_BYTE('COZUM BILGISAYAR')

COZUM BILGISAYAR

COZUM BILGISAYAR

**TRANSACTION**

Sonucunda toplam olarak başarılı olan veya başarısız olan bir gurup dml işlemlerine denmektedir.

**TRANSLATE**

Verilen karakter dizisi içerisinde 'eski' parametresi olarak girilecek karakterleri bularak 'yeni' olarak girilecek olan karakterler ile yer değiştirir.

SELECT TRANSLATE('123.45','0123456789','COZUM') FROM HASTA

TRANSLATE('123.45','0123456789','COZUM')

OZU.M

OZU.M

**TRIGGER**

Tetikleme(triger) bir tabloda belirtilen işlem olduğunda veri tabanı tarafından otomatik olarak çalıştırılan kaydedilmiş yordamdır.

**TRIM**

SELECT TRIM('COZUM BILGISAYAR') FROM HASTA

TRIM('COZUM BILGISAYAR')

COZUM BILGISAYAR

COZUM BILGISAYAR

**TRUNC**

Sayi alanina girilen rakam, m olarak belirtilen ondalık kadar sondan keser. TRUNC(54.923,2) -->45.92 TRUNC(54.923,-1) -->40 TRUNC (54.929,2) -->45.92

SELECT TRUNC(54. 923,2), TRUNC(54.923,-1), TRUNC(54.929,2) FROM HASTA

TRUNC(54. 923,2) TRUNC(54.923,-1) TRUNC(54.929,2)

55 50 55

55 50 55

**TRUNCATE**

Tablodaki tüm kayıtları siler. Delete komutu gibi olmasına karşın o komuttan çok daha hızlı silme işlemi yapar. Rollback komutu ile silinen kayıtlar geri getirilemez. Otomatik olarak commit olur. Tabloyu ancak yetkisi olan kullanıcı silebilir.

**ORACLE SQL DÖKÜMAN VE NOTLARI - 2**

1. Dökümanın devamı.Oracle komutlarının Türkçe açıklamaları ve örnekleri. Başucunuzda bulunması gereken Türkçe döküman.

**UID**

SELECT UID FROM HASTA

UID

63

63

**UNION**

İki tane sql sonucu dönen kayıtların birleşim kümesi.

**UPDATE**

Tablodaki kolonlarin degerlerini degistirmek için kullanilir. NOT:Kesinlikle bu örneği denemeyiniz. Var olan bilgileriniz değişecektir. Hasta tablosu içinde ki hastanın numarası adlı alan otomatikmen 3 artacaktır. Eğer bu örneği denediğseniz. ROLLBACK yazarak yapmış olduğunuz bu değişikliği geri alabilirsiniz.

UPDATE HASTA SET HS\_KEY=HS\_KEY-3

HS\_KEY HS\_AD HS\_SOYAD CINSIYET ISLEMSAYI DOGUMYER DOGUMTARIHI

2 I.HAKKI ALPTÜRK E 3

7 VEDAT KARAARSLAN E 2

10 ZAFER TEKBUDAK E 1

14 CELALETTİN DINÇER E 1

**UPPER**

UPPER(d1) d1 dizesinin bütün karakterleri büyük olarak sonuçta döndürür.

SELECT UPPER ('Cozum Bilgisayar') FROM HASTA

UPPER ('Cozum Bilgisayar')

COZUM BILGISAYAR

COZUM BILGISAYAR

**USER**

Veritabanındaki nesnelerin sahiplerine kullanıcı (user) denmektedir.

SELECT USER FROM HASTA

USER

TELEKOM

TELEKOM

**USER\_OBJECTS**

Kullanıcının sahip olduğu nesneler hakkında bilgi içeren görüntüdür.

**USERENV**

SELECT USERENV('instance'), USERENV('ISDBA') FROM HASTA

USERENV('instance' USERENV('ISDBA')

1 FALSE

1 FALSE

**USING**

**V$CONTROLFILE**

**V$DATABASE**

Veritabanı ile ilgili kontrol dosyalarında ki bilgileri içeren dosyadır.

**V$DATAFILE**

Kontrol dosyalarındaki veri dosyaları hakkında bilgi içeren tablodur.

**V$DATAFILE\_HEADER**

**V$FIXED\_TABLE**

**V$INSTANCE**

Veritabanının anlık durumu hakkında bilgi içeren tablodur.

**V$LOGFILE**

Bütün log dosyalar hakkın da bilgi içeren dosyadır.

**V$OPTION**

**V$PARAMETER**

Bütün parametre degerleri hakkında bilgi içeren tablodur.

**V$PROCESS**

Anlık olarak aktif olan işlemler hakkında bilgi içeren tablodur.

**V$PWFILE\_USERS**

**V$SESSION**

Veritabanındaki o anki baglantılar hakkinda bilgi içeren tablodur.

**V$SGA**

'System global area' hakkında bilgi içeren tablodur.

**V$SYSTEM\_PARAMETER**

**V$THEREAD**

Kontrol dosyasındaki thread değerleri hakkında bilgi içeren tablodur.

**VARCHAR**

**VARCHAR2**

Değişken uzunluktaki alfanümerik (karakter) dataların tutulabildiği alanlar için kullanılır.

**VARIANCE**

Belirtilen kolondaki kayıtların değerlerinin matematiksel varyansını bulur.

**VIEW**

VIEW, Bir ya da daha fazla çizelgedeki verilerin özel bir gösterimidir. Bir görüntü, saklanmış bir sorgu (stored query) olarak da düşünülebilir. Görüntüler, gerçekte veri içermezler. Verilerini, temel çizelgeler (base tables) olarak adlandırılan çizelgelerden ya da başka görüntülerden türetirler. Görüntüler, çizelgeler gibi, üzerlerinde bazı sınırlamalarla ekleme, silme, günleme ve sorgulama işlemlerine izin verirler. Görüntü üzerinde gerçekleştirilen tüm işlemler, görüntünün temel tablolarını da etkiler.

**VIEW LOG**

**WHERE**

Belli şartları sağlayan kayıtları eleme işleminin yapıldığı (şartların yazıldığı) bölümdür.

SELECT \* FROM HASTA WHERE HS\_KEY=93

HS\_KEY HS\_AD HS\_SOYAD CINSIYET ISLEMSAYI DOGUMYER DOGUMTARIHI

90 AHMET AYAZ E 2 ANKARA 12.11.1970

**YEAR**