1.2. Yapısal Sorgulama Dili (Structured Query language,SQL)

İlişkisel Veritabanı Yönetim Sistemleri (Relational Database Management Systems -RDBMS) modeli ilk önce
1970 yılında Dr. E. F .Codd tarafından tarif edilmiştir. SQL veya Structured English Query Language (SEQUEL), IBM
firması tarafından Codd'un modelini kullanmak için geliştirilmiştir. SEQUEL sonraları SQL olmuştur. 1979 yılında,
Relational Software (şu an, Oracle Corporation), SQL'in ilk ticari uygulamasını geliştirmiştir. Bugün,SQL, ilişkisel
veritabanı yönetim sistemleri standardı olarak kabul edilmektedir.

SQL, ilişkisel veritabanlarındaki bilgileri orgulamak için kullanılan dildir. SQL, bütün kullanıcıların ve ygulamaların
veritabanına erişmek için kullandıkları komutlar bütünüdür. Uygulama programları ve veritabanı araçları kullanıcılara
çoğu durumda SQL kullanmadan veritabanına erişim imkanı sunmaktadırlar fakat bu uygulamalar da geri planda
SQL kullanmaktadırlar.

Oracle SQL'i, standartlara uygundur. Daha da ötesinde, Oracle, SQL standartlarının gelişmesinde motor güç olan
bir kurumdur. American National Standarts Institue (ANSI) ve International Standarts Organization (ISO) tarafından
belirlenen son SQL standardı, SQL-92'dir. SQL-92'de, üç aşamalı uygunluk vardır. Bunlar;

.ilk seviye (Entry Level)
.Orta seviye (Intermediate Level)
.ileri seviye'dir (Full Level)

Oracle7, ilk seviye uygunluğuna sahiptir. SQL, ilişkisel veritabanları ile uygulamaların diyaloğunu sağlamaktadır.
SQL, temelde verilerle mantıksal seviyede çalışmaktadır. Yani, bir tablodan bir kaç kayıt seçebilmek için, o kayıtları
seçebilecek bir şart belirtilir. Şarta uyan bütün kayıtlar bir basamakta gelir ve bunlar kullanıcıya gösterilebildiği gibi,
bir başka SQL'e veya bir uygulamaya da gönderilebilir. Kayıtların tek tek nasıl geldiği ve fiziksel olarak eritabanının
neresinde ve nasıl tutulduğu ile SQL ilgilenmemektedir.

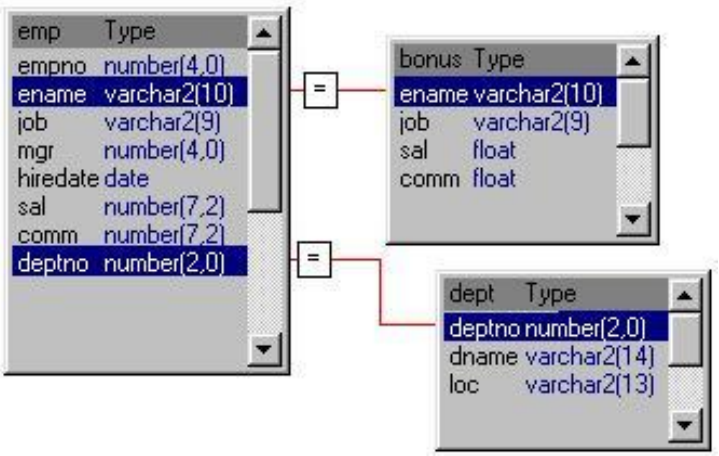
SQL komutları ile

.veri sorgulama
.bir tabloya kayıt ekleme, değiştirme ve silme
.veritabanı nesnelerini (database objects) yaratma, değiştirme ve silme
.veritabanına ve nesnelerine erişimi kontrol etme
.veritabanı bütünlüğünü ve tutarlılığını sağlama işlemleri yapılabilmektedir.

SQL komutları bir veya daha fazla satır olabilmektedir. SQL cümlelerinin sonuna noktalı virgül (;) konmaktadır.
irden fazla satır olan komutlarda en son satırın sonuna "/" işareti isteğe bağlı olarak konabilmektedir.
PL/SQL, Oracle'ın SQL komutlarına yapısal dillere ait özellikleri (begin, end, ioop, for, if, elsif, vb.) eklediği kendi standardı
olan bir dildir. ORACLE SQL, SQL *PLUS, PL/SQL komutlarının kullanılabildiği Oracle ürünüdür.

ORACLE SQL, SQL *PLUS, PL/SQL ve eklemiş olduğu bir dildir. PUSQL ile, veritabanı ile ilgili çok önemli
işlemler yapılabilmektedir. SQL bilinmeden, PL/SQL ile hiçbir işlem yapılamaz, bu yüzden önce SQL iyi bir şekilde
öğrenilmelidir.

SQL cümlelerinde kullanılan tabloların yapısı ve içeriği



Empno	Ename	Job	Mgr	Sal	Hiredate	Comm	Deptno
7369	SMITH	CLERK	7902	800	17.12.1980		20
7499	ALLEN	SALESMAN	7698	1600	20.02.1981	300	30
7521	WARD	SALESMAN	7698	1250	22.02.1981	500	30
7566	JONES	MANAGER	7839	2975	02.04.1981		20
7654	MARTIN	SALESMAN	7698	1250	28.09.1981	1400	30
7698	BLAKE	MANAGER	7839	2850	01.05.1981		30
7782	CLARK	MANAGER	7839	2450	09.06.1981		10
7788	SCOTT	ANALYST	7566	3000	19.04.1987		20
7839	KING	PRESIDENT		5000	17.11.1981		10
7844	TURNER	SALESMAN	7698	1500	08.09.1981	0	30
7876	ADAMS	CLERK	7788	1100	23.05.1987		20
7900	JAMES	CLERK	7698	950	03.12.1981		30
7902	FORD	ANALYST	7566	3000	03.12.1981		20
7934	MILLER	CLERK	7782	1300	23.01.1982		10

Deptno	Dname	Loc
10	ACCOUNTING	NEW YORK
20	RESEARCH	DALLAS
30	SALES	CHICAGO
40	OPERATIONS	BOSTON

İÇİNDEKİLER

SQL'e giriş
Fonksiyonlar
Karakter fonksiyonları
Sayısal fonksiyonlar
Tarih fonksiyonları
Dönüştürme (Conversion) fonksiyonları
Değişik türlerde parametre kabul edebilen fonksiyonlar
Grup fonksiyonlarr
Birden fazla tabloyu sorgulama (Join)
Equi-Joins
Non-Equi-Joins
Outher Joins
Tabloyu kendisi ile birleştirme
Sorguda küme işlemleri kullanma (Union, intersect, minus)
İç içe sorgu cümleleri
Nasıl işlenir ?
ANY/ALL işleci
HAVING işleci
Sıralı liste üretme
Dikkat edilmesi gereken noktalar
EXIST işleci

Veri tanımlama Dili (Data Definition Language)

Tablo yaratma komutu (CREATE TABLE)
Tablo kolon türleri
Tablo tanrmlarını değiştirme (ALTER TABLE)
Tablo silme (DROP TABLE)
Tablo ismini degiştirme (RENAME)
Tablo tanımnr görme (DESCRIBE)

Veri Kullanma Dili

Tabloya yeni kayıt ekleme (INSERT)
Başka bir tablodan kayıt kopyalama
Kayıt değiştirme (UPDATE)
Kayıt silme (DELETE)
İşlemleri bir süreç içerisinde toplama
(Transaction processing and COMMIT/ROLLBACK)
Tutarlı kayıt okuma (Read consistency)
Eşzamanlı işlemler ve kaynakların paylaşımı (LOCK)
Kaynak kilitleme türleri
ROWID yapay kolonunun kullanrmı
INDEX kuilanımı
Index kullanma nedenleri
Index türleri (unique, non-unique, single/concatenated)
Index yaratma (CREATE INDEX)
Index silme (DROP INDEX)

1. SQL'E GiRiŞ

Onceleri SEQUEL (Structred English QUEry Language) olarak adlandırılan bu dilin adı, daha sonra ingilizce
söylenişine uygun olarak SQL olarak değiştirildi. SQL, bilgisayar uzmanı olmayan ve yüksek düzeyli, işlemsel
olmayan (non-procedural) bir dil aracılığıyla veri tabanı ile iletişim kurmak isteyen kullanıcı kitlelerine

yönelik bir
veri tabanı dilidir.

SQL Komut Kümesi

Komut	Acıklama
SELECT	Verileri getirmek için kullanılır.
INSERT	Yeni kayıt eklemede kullanılır.
UPDATE	Var olan bir kaydı değiştirmek üzere kullanılır.
DELETE	Var alan bir kaydı silmek için kullanılır.
CREATE	Veri tabanında yeni bir nesne yararmak için kullanılır. (Table, View, Index, Synonym vb.)
ALTER	Veri tabanındaki nesne üzerinde değişiklik yapar. (Omeğin yeni bir kolon eklemek gibi)
DROP	Veri tabanındaki bir nesneyi siler.
GRANT	Kullanıcılara erişim yetkilerinin verilmesinde kulanılır.
REVOKE	Verilen erişim yetkilerinin geri alınmasında kullanılır.

SQL komulan nasıl yazılır
.Yazılan SQL cümlesi bir yada daha fazla satır olabilir.
.SQL cümlesi içerisinde yer alan cümlecikler genelde farklı satırlara yazılır.
.Komut kelimeleri bölünerek alt satırdan devam edilemez.
.SQL komutları büyük yada küçük harflarle yazılabilir.
.Her SQL cümlesi noktalı virgül (;) ile bitirilir.

Basit SELECT ifadeleri

En basit biçimiyle bir SELECT ifadesi
1. Istenilen tablonun kolonlarının belirtildiği bir SELECT cümleciği,
2. Hangi tablodan veri seçileceğini ifade eden bir FROM cümleciği içermelidir.

```
SELECT DEPTNO,ENAME,MGR FROM EMP;
```

DEPT NO	ENAME	MNG
10	KING	
20	JONES	7839
30	BLAKE	7839
10	CLARK	7839
20	SCOTT	7566
20	FORD	7566
20	SMITH	7902
30	ALLEN	7698
30	WARD	7698
30	MARTIN	7698
30	TURNER	7698
20	ADAMS	7788
30	JAMES	7698
10	MILLER	7782

Eğer tablodaki bütün kolonlar seçilek istenseydi, * ile hepsi ifade edilebilirdi.

```
SELECT * FROM EMP
```

Aritmetik ifadeler Aritmetik ifadeler içerisinde kolon isimleri, sabit sayılar ve aritmetik işleçler

kullanılabilir.

~ Açıklama
+ toplama
- çıkrama
* çarpma
/ bölme

Aritmetik ifade içerisinde "*" ve "/" eşit öncelikli, "+" ve "-"ye göre daha önceliklidir. "+" ve "-" de eşit nceliklidirler. Cncelikleri tam ifade etmek için parantezler kullanılabilir.

SELECT SAL *12, COMM FROM EMP;

SAL kolonunun 12 ile çarpılmış biçimi getirilir.

Kolon isimlendirme Kolonların başlıkları SQl cümlesinde yazıldıkları biçimi ile getirilirler. Bunu değiştirmek mümkündür.

SELECT ENAME, SAL *12 ANNSAL, COMM FROM EMP;

Kolon başlığı olarak SAl *12 yerine ANNSAL kullanılacak.
Değiştirilen kolon başlıkları SQl cümlesi içerisinde artık yeni isimleri ile kullanılırlar.

Birleştirme işleci (||) Kolonlardan gelen değerleri yan yana yapıştırmak mümkündür. "11" Işleci ile kolonlar arasında boşluk olmadığına dikkat ediniz.

SELECT EMPNOIIENAME EMPLOYEE FROM EMP;

EMPLOYEE
7839KING
7566JONES
7698BLAKE
7782CLARK

Literaller Literailer veriler ile birlikte çıkmasını istediğimiz karakterlerdir.

SELECT EMPNOII'-'IIENAME CALISAN,'Calistigi bolum', DEPTNO BLM. FROM EMP;

CALISAN	Calistigi bolum	BLM
7839-KING	Calistigi bolum	10
7566-JONES	Calistigi bolum	20
7698-BLAKE	Calistigi bolum	30
7782-CLARK	Calistigi bolum	10
7788-SCOTT	Calistigi bolum	20
7902-FORD	Calistigi bolum	20
7369-SMITH	Calistigi bolum	20
7499-ALLEN	Calistigi bolum	30
7521-WARD	Calistigi bolum	30
7654-MARTIN	Calistigi bolum	30
7844-TURNER	Calistigi bolum	30
7876-ADAMS	Calistigi bolum	20
7900-JAMES	Calistigi bolum	30
7934-MILLER	Calistigi bolum	10

Fonksiyonlar Fonksiyonlar aldıkları kolonun yada sabit değişkenin değerini yeni bir biçime dönüştürürler. NULL değerlerin yakalanması NULL değeri olmayan anlamı taşır ve herhangibir işleme giren bu değer mutlaka NULL üretir ve istenmeyen bir sonuç alınmasına yol açabilir. Bunu engellemek için NVL fonksiyonu kullanılır.

SELECT SAL *12-NVL(COMM,O) YILLIK FROM EMP;

Eğer NVL fonksiyonu kullanılmamış olsa idi COMM değeri olmayan bütün kayırlar için SAL *12-COMM esaplamasından NULL geri döndürülecekti. NVL fonksiyonu içerisine yazılan kolonun değeri NULL ise NULL yerine ne kullanılacağını belirler. Omekte COMM kolonunun NULL değerleri için sıfır kullanılmaktadır.

DISTINCT ifadesi Eğer seçilen kolon aynı satırdan birden fazla getiriyorsa, bunlardan fazla olan satırları atmak mümkündür.

SELECT DISTICT DEPTNO FROM EMP;

Bu sorgu bir bölümde birden fazla çalışan olduğu halde çalışanlar tablosundan sadece bölüm numaralarını seçmekte ve tekkarlı satırları elemektedir.

ORDER BY ifadesi Seçilen kolonların hangi sırada geleceğini belirlemek amacıyla kullanılır.
SELECT ENAME,JOB FROM EMP ORDER BY ENAME;
Ornekte çalışanlar isimlerilerine göre sıralı olarak getirilmektedir.
ORDER BY ENAME ~ : Küçükten büyüğe sıralar
ORDER BY ENAME ~ : Büyükten küçüğe sıralar
Birden fazla kolon sıralanmak istendiğinde kolon isimleri virgül ile ayrılarak yazılır. (ODER BY DEPTNO.SAL DESC gibi). Sıralama sırasında NULL değerleri sıralama nasıl olursa olsun ilk sırada yer alırlar.

WHERE ifadesi Kayıtlar arasından sadece istenilen koşulları taşıyanların seçilebilmesini sağlamak amacıyla WHERE ifadesi kullanılır. Koşulu vermek için mantıksal işleçler kullanılır.

~ Anlamı
'=' eşittir
'>' büyüktür
'<' küçüktür
'>=' büyük eşittir
'<=' küçük eşittir

SELECT DNAME,DEPTNO FROM DEPT DEPTNO WHERE DEPTNO>20;
20'den büyük olanları seç

SQL işleçleri
islec anlamı
BETWEEN ...AND ...iki değer arasında (değerler dahil)
IN (liste) listedeki değerler içinden en az biri
LIKE Benzer ise
IS NULL Boş değer ise

BETWEEN işleci
SELECT ENAME,SAL FROM EMP WHERE SAI BETWEEN 1 000 AND 2000;
Değeri 1000'e eşit ve 1000'den büyük ve 2000'e eşit ve 2000'den küçük olanları seçer.

IN İşleci
SELECT EMPNO,ENAME,SAI,MGR FROM EMP WHERE MGR IN (7902, 7566, 7788);
MGR 7902 yada 7566 yada 7788 olan kayıtlar seçilir.

LIKE İşleci
Bazı durumlarda tam karşılaştırma yapmak işimize yaramayabilir. Omeğin adı 'A' harfi ile başlayanları
seçmek iştediğimizde. Bu durumda llKE işleci kullanılır. Karşılaştırmada kullanılacak sabit içerisinde
özel semboller kullanılır.
Sembol Anlamı
% Sıfır yada daha fazla karakterler
Bir karakter
SELECT ENAME FROM EMP WHERE ENAME llKE 'S%';
Omekteki SQL cümlesi 'S' harfi ile başlayan isimleri getirir.
SELECT ENAME FROM EMP WHERE ENAME llKE '-';
Ikinci ömekteki SQl cümlesi ise dört harfli isimleri getirir.
/%' ve '-' sembolleri bir sabit içerisinde aynı anda değişik kombinasyonlarla kullanılabilir.

IS NULL işleci
Bir kolon değerinin NUll olup olmadığını anlamanın tek yolu bu işleci kullanmaktır. '=NUll' işleci kullanmak aynı şey değildir ve hiçbir sonuç vermez.
SELECT ENAME,MGR FROM EMP WHERE MGR IS NULL;
Yukarıdaki ömek yöneticisi olmayan çalışanların listesini üretir.

Negatif Test
Bir koşulu sağlayan değil de sağlamayanlar seçilmek isteniyor olabilir. Bu durumda işlecin anlamının tersine
çevrilmesi gerekir.
~ Anlamı

!= eşit değildir (V AX için)
,,= eşit değildir (IBM için)
<> eşit değildir

SQL isleci Anlamı
NOT BETWEEN Verilen değerler arasında olamyan değerler
NOT IN (liste) Listedeki değerler içerisinde olmayan
NOT LIKE Benzer olmayan
IS NOT NULL Boş olmayan

Sorguda birden fazla koşul kullanma AND (ve) ve OR (ya da) işleçleri birden fazla verilmiş olan koşul
ifadelerini birleştirmek amacıyla kullanılır. AND işleci arasında bulunduğu her iki koşul ifadesinin de doğru olmak
zorunda olduğunu, OR işleci arasında bulunduğu her iki koşul ifadesinden sadece birisinin doğru olmasının
yeterli olduğunu ifade etmek için kullanılır.

Aynı SQL cümlesinde hem AND hem de OR işleçleri koşullar arasında istenilen kombinasyonda kullanılabilir.
AND işleci her zaman OR işlecine göre daha önceliklidir ve daha önce işletilir. OR işlecini AND işlecine göre
öncelikli hale getirebilmek için parantezlerden "()" faydalanılır.

```
SELECT EMPNO,ENAME,JOB,SAL
FROM EMP
WHERE SAL BETWEEN 1 000 AND 2000
AND JOB='CLERK'.
```

FONKSİYONLAR

Fonksiyonlar aldıları değerleri, kendilerine yüklenmiş olan göreve göre yeni bir biçime dönüştürürler.

Fonksiyonlar aşağıdaki gibi sınıflandırılabilir.
.Karakter fonksiyonları
.Sayısal fonksiyonlar
.Tarih fonksiyonları
.Dönüştürme fonksiyonları
.Herhangibir tür veri üzerinde işlem yapabilen fonksiyonlar
.Grup fonksiyonları

Karakter Fonksiyonları
Karakter türü değişken ya da sabitleri alarak yüklenmiş olduğu göreve göre bunlar üzerinde işlem yapar ve
karakter ya da sayısal bir değer döndürür.

LOWER(kolon/sabit) : Giriş olarak okuduğu karakter dizisini küçük harflere önüştürür. 'kolonlsabit' şeklindeki gösterim bir tek parametre girileceğini ve bu parametrenin bir kolon adı ya da
bir sabit karakter dizisi olabileceğini gösterir.

```
SELECT LOWER(DNAME),LOWER('SQL') FROM DEPT;
```

UPPER(kolonlsabit) : Parametre olarak alınan karakterleri büyük harfe dönüştürür.
İfadenin büyük harf veya küçük harf olması sonucu değiştirmez, her durumda byük harfe çevirir.

INITCAP(kolonIsabit) : Parametre olarak verilen karakter dizisi içerindeki her kelimenin baş harfini büyük harfe diğer harflerini küçük harfe dönüştürür.

LPAD(kolon/sabit,n,'karakter dizisi') : Parametre olarak verilen kolon ya da sabiti yazmadan önce başına n adet istenilen karakter dizisinden basar. Eğer karakter dizisi verilmez ise boşluk ekler.

SELECT LPAD(DNAME,20,'*'),LPAD(DNAME, 15) FROM DEPT;

PAD(DNAME,20,'*')	LPAD(DNAME,15)
**********ACCOUNTING	ACCOUNTING
************RESEARCH	RESEARCH
******************SALES	SALES
**********OPERATIONS	OPERATIONS

RPAD(kolonIsabit,n,'karakter dizisi') : Parametre olarak verilen kolon ya da sabitin sonuna n adet istenilen karakter dizisinden basar. Eğer karakter dizisi verilmez ise boşluk ekler.

SELECT LPAD(DNAME,20,'*'),LPAD(DNAME, 15) FROM DEPT;

RPAD(DNAME,20,'*')	RPAD(DNAME,15)
ACCOUNTING**********	ACCOUNTING
RESEARCH************	RESEARCH
SALES******************	SALES
OPERATIONS**********	OPERATIONS

SUBSTR(kolonIsabit,poz,n) : Verilen kolon yada sabit karakter dizisinin istenilen pozisyonundan başlayarak n adet karakteri geritir.

SELECT SUBSTR('Cengiz Çakmak',4,3) FROM DUAL;

INSTR(kolonIsabit1'karakter dizisi') : Verilen kolon ya da sabit karakter dizisi içerisinde istenilen karakter ya da karakter dizisinin ilk geçtiği konumu döndürür.

L TRIM(kolonIsabit,'karakter dizisi') : Istenilen karakteri ya da istenilen karakterleri verilen kolon ya da sabit karakter dizisinin başından siler. Dizinin başında yer alan silinecek karakter ard arda yer aldığı sürece silinir. Eğer hiçbir karakter verilmez ise dizinin başındaki boşluklar silinir.

RTRIM(kolonIsabit,'karakter dizisi') : L TRIM ile aynı işlemi yapar; tek farkı islenilen karakteri (ya da karakterler) dizinin sonundan siler.

LENGTH(kolonIsabit) : Verilen karakter dizisinin toplam kaç karakter uzunlukta olduğunu döndürür.

TRANSLATE(kolonIsabit1eski,yeni) : Verilen karakter dizisi içerisinde 'eski' parametresi olarak girilecek karakterleri bularak 'yeni' olarak girilecek olan karakterler ile yer değiştirir.

Sayısal Fonksiyonlar
Sayısal fonksiyonlar sayısal değerleri parametre olarak alır ve yine sayısal değerler üretirler.

ROUND(kolonIsabit,n) : Verilen değeri virgülden sonra n basamağını dikkate alarak yuvarlar. Eğer n değeri negatif girilirse virgülün soluna kalan n adet rakam yuvarlanır.

ROUND(45.923,1) --> 45,9
ROUND(45.923) --> 46
ROUND(45.923,1) --> 45,3

ROUND(42.323,-1) --> 40

TRUNC(kolonIsabit,n) : Virgülden sonraki n basamak sıfırlanır. Eğer n değeri negatif girilirse virgülün solunda kalan n adet rakam sıfırlanır.

CEIL(kolonIsabit) : Verilen değerden büyük en yakın tamsayıyı döndürür.

FLOOR(kolonIsabit) : Verilen değerden küçük en yakın tamsayıyı döndürür.

POWER(kolonIsabit,n) : Verilen değerin n'inci kuvvetini alır.

SQRT(konIsabit) : Verilen değerin kare kökünü bulur.

SIGN(kolonIsabit) : Eğer verilen değer 0 ise 0, negatif ise -1 , pozitif ise +1 değeri döndürür.

ABS(kolonIsabit) : Verilen değerin mutlak değerini döndürür.

MOD(d1 ,d2) : d1 'in d2'ye bölümü sonucu oluşan kalan değerini döndürür .

Tarih Fonksiyonları Oracle DA TE türünde tanımlanmış alanlar içerisinde saniye düzeyine kadar tarih bilgisini saklayabilir. Fakat giriş ve çıkıŞ sırasında tarih formatı ayarlanarak bu detayda bilgi girişine gerek duyulmadan tarih bilgisi
üzerinde istenilen işlemler yapılabilir.

Tarih : Yüzyıl, Yıl, Ay, Gün, Saat, Dakika, Saniye,

Oracle veri tabanının başlanğıç olarak kullandığı tarih formatı (değiştirilmediği sürece) DD-MON-YY şeklindedir.
Eğer veri tabanı türkçe modunda çalıştırılıyorsa ayların kısaltmaları türkçe olarak yapılacaktır (MA YIS : MAY,
HAZIRAN : HAZ gibi).

Veri tabanından sistem tarihini de okumak aşağıdaki SQL sorgusu ile mümkündür.

SELECT SYSDATE FROM SYSTEM.DUAL;

Bu sorgu DUAL isimli sahte bir tablo kullanarak o andaki sistem tarihini okumamızı sağlamaktadır.

Tarih üzerinde aritmetik işlemler
tarih+sayı : Tarihe istenilen gün sayısı eklenir.
tarih-sayı : Tarihten istenilen gün sayısı çıkartılır.
tarih1-tarih2 : Iki tarih arasındaki gün sayısını bulunur
tarih+sayıl24 : Istenilen saat sayısını tarihe eklenir.

MONTHS_BETWEEN(tarih1,tarih2) : Iki tarih arasında kaç ay olduğunu hesaplar. Eğer tarih2 tarih1 'den daha büyükse sonuç negatif olarak üretilir.

ADD_MONTHS(tarih,n) : Istenilen tarihe n ay ekler.

NEXT_DAY(tarih,gün) : Verilen tarihten bir sonraki haftanın istenilen gününün tarihini döndürür. Girilen gün bir numara yada günün adı olabilir. 'FRIDAV' ve 6 aynı günü ifade eder. Günler pazar gününden itibaren sayılmaya başlanır. Omeğin pazartesi günü için 2 girilmelidir.

LAST_DAY(tarih) : Girilen tarihin ayının son gününün tarihini döndürür.

ROUND(tarih) : Girilen tarihin saat kısmını yuvarlar. Bu genelde saat bilgisi içeren tarihlerin arşılaştırılmasında faydalı olur.

ROUND(tarih,'MONTH') : Girilen tarihi ay bilgisine kadar yuvarlar. Ayın 15'şinden önceki günler ayın ilk gününün tarihini, sonraki günler için sonraki ayın ilk gününün tarihini getirir.

ROUND(tarih,'VEAR') : Girilen tarihi yıl bazında yuvarlar. Girilen tarih yılın ilk yarısını gösteriyor ise o yılın ilk gününün tarihi geri dönderilir, değilse bir sonraki yılın ilk gününün tarihi döndürülür.
TRUNC(tarih,'MONTH') : Verilen tarihin ayının ilk gününün tarihini bulur.
TRUNC(tarih,'VEAR') : Verilen tarihin yılının ilk gününün tarihini bulur.

Dönüştürme Fonksiyonlan

TO_CHAR(sayıltarih,['format]) : Verilen rakam ya da tarihi istenilen formatta karaktere dönüştürür.

TO_NUMBER('karakterler') : Karakter türünde verilmiş olan rakamları sayısal türe dönüştürür.

TO_DATE('karakter' ,'format') : Formatı belirli karakter halindeki bir tarihi tarih türüne dönüştürür.

Tarih Formatlan (TO_DATE ile kullanılabilen)

ELEMAN	Anlamı
yyyy	Dört Basamaklı Yıl
yyy	Yılın son üç basamağı
yy	Yılın son iki basamağı
Y	Yılın son basamağı
RR	Farklı Dil İçin yılın son iki basamağı
Q	Ocak-Mart i
MM	İki basamaklı ay
RM	Romen rakamı ile ay
MONTH	Ayın uzun ismi
MON	Ayın kısa ismi
ww	Yılın hangi haftası olduğu(1-53)
w	Ayın hangi haftası olduğu(1-5)
DDD	Yılın günü (1-366)
DD	Ayın günü (1-31)
D	Haftanın günü(1-7)
DAY	Günün uzun adı (Pazartesi)
DY	Günün kısa adı (Pzt)
HH	veya Günün saati (1-12)
HH12	Günün saati (1-12)
HH24	Günün Saati (0-23)
MI	Dakika(0-59)
SS	Saniye(0-59)
SSSSS	Gec nraki sagece yarısından sonra saniye sayısı

Ornek : TO-DATE('27-0CT -95','DD-MON-RR')

Sayı Formatlan

ELEMAN	ÖRNEK	Anlamı

9	9999	Yazılacak sayının uzunluğunu belirler
0	0999	Eğer sayı küçükse boşluk yerine sıfır basar
$	$999	Rakam başına dolar işareti ekler
B	B99999	Sıfır olan sayıları basmaz
S	S9999	Poz. sayıların başına + neg. ise - işareti ekler.
PR	99999PR	Negatif ise <...> şeklinde yazar
D	99D99	Ondalıklı sayıları bu şekilde ayırır
G	9G999	Grup ayrıcını G harfinin olduğu yere basar
C	99999C	Rakamın yanına parabirimi kısaltmasını yazar
L	9999L	Ülke için kullanılan parabirimi kısaltmasını yazar.
,	9,999	istenilen pozisyona virgül basılır
.	999.99	ondalıklı kısımlar nokta ile ayrılır
V	999V99	Gelen sayıyı 10'un n'inci kuvveti ile çarpar. n V harfinden sonraki 9'ların sayısıdır.
EEEE	9.999EEEE	8ilimsel olarak yazar
RN yada rn	RN rn	Girilen sayıyı büyük yada küçük romen rakamları ile 1 ile 3999 arası için yazar.

Değişik türlerde parametre kabul edebilen fonksiyonlar

DECODE Bu komut ile kolon isimlerini koşullara bağlayarak verilerin durumuna gôre değişik kolon isimlerini seçmek
mümkündür. Yapısal dillerdeki 'if-then-else' yapısının ilkel bir ômeğidir.

```
DECODE(kolon/ifade,ara1,sonuc1,[ara2,sonuc2, ...,]default)
SELECT ENAME,JOB,
DECODE(JOB,'CLERK','WORKER','MANAGER','BOSS','UNDIFIENED')
DECODE_JOB FROM EMP
```

ENAME	JOB	DECODE_JOB
SMITH	CLERK	WORKER
ALLEN	SALESMAN	UNDIFIENED
WARD	SALESMAN	UNDIFIENED
JONES	MANAGER	BOSS
MARTIN	SALESMAN	UNDIFIENED
BLAKE	MANAGER	BOSS
CLARK	MANAGER	BOSS
SCOTT	ANALYST	UNDIFIENED
KING	PRESIDENT	UNDIFIENED
TURNER	SALESMAN	UNDIFIENED
ADAMS	CLERK	WORKER
JAMES	CLERK	WORKER
FORD	ANALYST	UNDIFIENED
MILLER	CLERK	WORKER

14 satırları seçildi.

Grup Fonksiyonlan Grup fonksiyonları veri tabanından seçilen bir dizi satır üzerinde işlem yapar ve sonuç olarak kendisine
yüklenmiş olan göreve göre özet bir bilgi üretir. AVG([DISTINCTIALL] kolon) : Verilen kolon değerlerinin
ortalamasını bulur.

COUNT([DISTINCTIALL] kolonl*): Verilen kolonun NULL olmayanlarının sayısını getirir. Eğer * kullanılırsa toplam kaç satır sorgulandığı bulunur.

MAX([DISTINCTIALL] kolon) : Maximum değeri getirir.

MIN((DISTINCTIALL] kolon) : Minimum değeri getirir.

SUM([DISTINCTIALL] kolon) : Verilen kolon toplamını bulur.

```
SELECT COUNT(*) FROM EMP WHERE DEPTNO = 20;
```

20 numaralı bölümde kaç kişi olduğunu bulur.

GROUP BY ifadesi GROUP BY ifadesi ile sorgulanan satırlar belirli guruplara ayrılarak bu gruplar üzerinde grup fonksiyonları kullanılır.

```
SELECT JOB,AVG(SAL) FROM EMP
GROUP BY JOB;
```

J

JOB	AVG(SAL)
ANALYST	3000
CLERK	1037.5
MANAGER	2758.3333
PRESIDENT	5000
SALESMAN	1400

Her iş için ayrı ayrı ne kadar ortalama maaş verildiğini hesaplar.

```
SELECT DEPTNO,JOB,AVG(SAL) FROM EMP
GROUP BY DEPTNO,JOB;
```

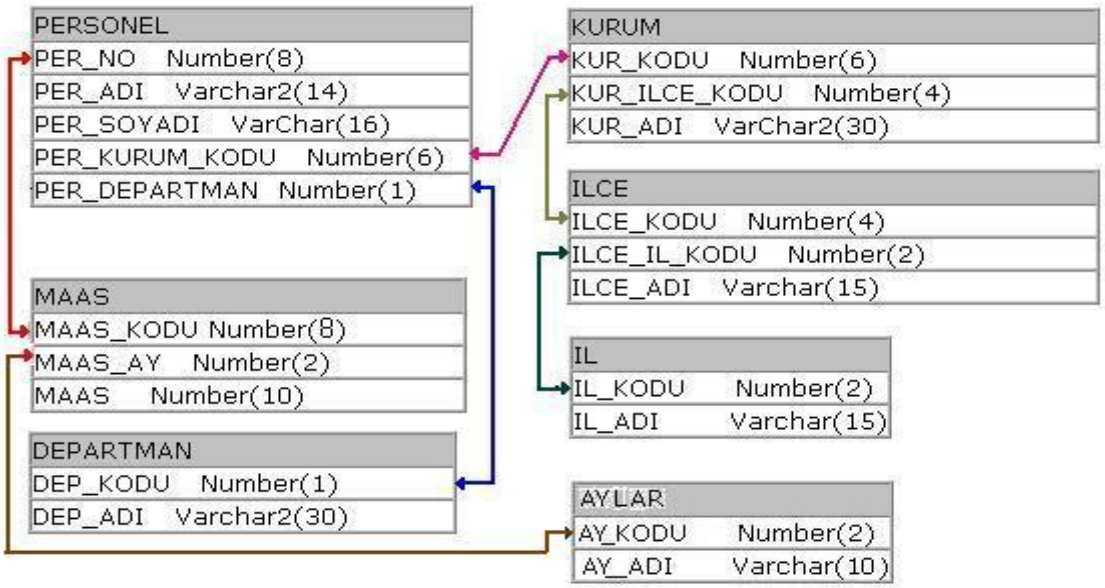
DEPT NO	JOB	AVG(SAL)
10	CLERK	1300
10	MANAGER	2450
10	PRESIDENT	5000
20	ANALYST	3000
20	CLERK	950
20	MANAGER	2975
30	CLERK	950
30	MANAGER	2850
30	SALESMAN	1400

9 satırları seçildi.

Her bölümdeki farklı işler için ne kadar ortalama maaş verildiğini hesaplar. Grup fonksiyonları ile birlikte kullanılan kolon isimleri mutlaka GROUP BY ifadesi içinde yer almalıdır. Tersi durumda hatalı bir SQL cümlesi yazılmış olur.

Tabloların Oluşturulması

Oluşturulması gereken tablolar ve aralarındaki ilişkiler aşağıda verilmiştir. Ayrıca tabloların üzerine tıklayarak örnek verileri de görebilirsiniz. Tabii ki veriler örnektir. Gerçek bir veritabanında kurum ve personel tabloları yüzbinlerce olabilir. Ayrıca SQL komutunun hızlı çalışabilmesi için birbirleriyle ilişki kurulmuş tablolar indexlenmelidir.



Burada örnek bir veritabanı tasarlanmıştır. Bir kuruluşun ülke çapında yayılmış şuıbeleri olsun. Bu şubelerde çalışan elemanların genel bilgileri ,maaş bilgileri ve hangi bölümde çalıştıklarına ilişkin bilgileri tutmaya çalışalım.
Primary Key Alanlar:
Per_No,Kur_Kodu,Ilce_Kodu,Il_KoduDep_Kodu (Bu alanlara girilen değerlerin tekrarlanmaması için tanımlanır.)

Ders 1

[Il tablosuna gözat](#) (sorgularda kullanılan tablo)

il_kodu	il_adi
1	Adana
2	Adıyaman
3	Afyon
4	Ağrı
5	Amasya
6	Ankara
7	Antalya
8	Artvin
9	Aydın
10	Balıkesir

Basit SQL Cümleleri

Mevcut bir tablodan istenilen şartları sağlayan kayıtların seçilmesi için SELECT kullanılabilir.
Genel Kullanım Şekli:
SELECT [Tablodan istenen alan isimlerinin yazılacağı bölüm]
FROM [Verilerin bulunduğu tablo isimlerinin yeraldığı bölüm]
WHERE [Koşul Yazılacak bölüm]
GROUP BY [Kayıtları gruplayıp getirmek için kulanılacak cümlecik]
HAVING [Grup bulunan SQL cümleciklerinde grup içinde şart kullanmak için]
ORDER BY Kayıtlarının sıralanması için kullanılacak bölüm]

Bu cümleciğe şu soruları sormamız mümkündür;
SELECT (Hangi alanları ?)
FROM (Hangi tablo ya da tablolardan)
WHERE (Hangi şartlarda ?)
GROUP BY (Nasıl bir gruplama ?)
ORDER BY (Hangi sırada ?)

Öncelikle bir tablonun yapısını görmek için;
DESC il;

```
İsim      Null?      Tip
----      ---------- ----
IL_KODU              NUMBER(2)
IL_ADI               VARCHAR2(14)
```

Bu bilgiler ışığında basit bir SQL cümlesiyle işe başlayabiliriz.
SELECT * FROM IL;
IL tablosundan tüm alanları getir..

```
IL_KODU IL_ADI
---------- --------------
1 Adana
2 Adıyaman
3 Afyon
4 Ağrı
5 Amasya
6 Ankara
7 Antalya
8 Artvin
9 Aydın
10 Balıkesir

10 satırları seçildi.
```

SELECT IL_ADI FROM IL;
Personel tablosundan tüm kayıtların perno,adı ve soyadı alanlarını listele..

IL_ADI

Adana

Adıyaman

Afyon

Ağrı

Amasya

Ankara

Antalya

Artvin

Aydın

Balıkesir

10 satırları seçildi.

Sorguda koşul vermek gerektiğinde bir takım karşılaştırmalar yapmamız gerekir. Karşılaştırma işlemine geçmeden önce karşılaştırma operatörlerini bir inceleyelim. Sorguda istediğiniz koşulu aşağıda görülen operatörlerle yapabiliriz.

=	:eşit
>	:Büyük
>=	:Büyük eşit
<	:Küçük
<=	:Küçük eşit
!=	:Farklı

Şimdi de IL tablosundan belli koşullara uyan kayıtları listelemeye çalışalım;
SELECT * FROM IL WHERE IL_KODU=6;

IL_KODU IL_ADI

---------- ------

6 Ankara

SELECT * FROM IL WHERE IL_KODU<5; Il tablosundan il_kodu 5 ten küçük kayıtlar seçildi

IL_KODU IL_ADI

--------- --------

1 Adana

2 Adıyaman

3 Afyon

4 Ağrı

SELECT * FROM IL WHERE IL_KODU<5; Il tablosundan il_kodu 5 veya 5 ten küçük kayıtlar seçildi

IL_KODU IL_ADI

---------- --------

1 Adana

2 Adıyaman

3 Afyon

4 Ağrı

5 Amasya

SELECT * FROM IL WHERE IL_KODU!=5; Il tablosundan il_kodu 5 ten farklı olan kayıtları listele..

IL_KODU IL_ADI

--------- ----------

1 Adana

2 Adıyaman

3 Afyon

4 Ağrı

6 Ankara

7 Antalya

8 Artvin

9 Aydın

10 Balıkesir

Where cümleceğinden sonra koşul bir tane olmak zorunda değildir. Koşul sayısını artırabiliriz. Ancak kullanılacak koşullar arasında mantıksal bir bağlaç olması gerekir. Bunlar;
AND : Koşulun sağlanması için şartlardan ikisinin de sağlanması gerekir.
OR : Koşulun sağlanması için şartlardan yalnız birinin sağlanması yeterlidir.

SELECT * FROM il WHERE Il_kodu=5 AND il_kodu=7;

Bu sorguda ben Il_kodu 5 ve 7 olan kayıtları listele dedim ama sonuçte hiç bir kayıt dönmeyecektir. Çünkü Il_kodu hem 5 hem de 7 olan hiç bir kayıt yoktur.

SELECT per_no,per_adi,per_soyadi FROM personel WHERE per_adi='ALİ' AND per_departman='SATIŞ';

Burada ise kişinin adı ALİ olacak ve departmanı SATIŞ olacak. Yani Satış departmanında çalışan ALİ 'leri listeleyen bir sorgu.

SELECT * FROM il WHERE il_kodu=5 OR il_kodu=7;

IL_KODU IL_ADI

---------- -------

7 Antalya

5 Amasya

Il_kodu 5 olan VEYA il_kodu 7 olan kayıtları listele ifadesini kullandığımda ise hem 5 hem 7 olan kayıtları listeledi.

Oracle SQL Plus'ta koşul hanesinde eğer string bir alan karşılaştırılacaksa ; sabit ifadeleri tek tırnak içinde belirtmeliyiz.

SELECT * FROM il WHERE il_adi='ANKARA'; gibi..

WHERE şart cümleciği içinde yukarıdaki karşılaştırma operatörlerini kullanabildiğimiz gibi ayrıca operatörlerden de söz edebiliriz

BETWEEN a1 AND a2 :a1 ve a2 arasındaki kayıtlar
NOT BETWEEN a1 AND a2 :a1 ve a2 arasında olmayan kayıtlar
IN(a1,a2,a3,...) : Listede belirtilen herhangi bir değeri içeren kayıtlar
NOT IN(liste) : Listede belirtilen herhangi bir değeri içermeyen kayıtlar
LIKE :Karakter ifadelerde alan içeriğinin bir bölümünü sağlayan kayıtlar
NOT LIKE :Karakter ifadelerde alan içeriğinin bir bölümünü sağlamayan kayıtlar
IS NULL :Boş olan kayıtlar
IS NOT NULL :Boş olmayan kayıtlar

SELECT * FROM il WHERE il_kodu BETWEEN 3 AND 7;

IL_KODU IL_ADI

---------- --------

3 Afyon

4 Ağrı

5 Amasya

6 Ankara

7 Antalya

SELECT * FROM il WHERE il_kodu NOT BETWEEN 3 AND 7;

IL_KODU IL_ADI

--------- ----------

1 Adana

2 Adıyaman

8 Artvin

9 Aydın

10 Balıkesir

SELECT * FROM il WHERE il_kodu IN(3,5,8,105);

IL_KODU IL_ADI

---------- -------

3 Afyon

5 Amasya

8 Artvin

SELECT * FROM il WHERE il_kodu NOT IN(3,5,8,105);

IL_KODU IL_ADI

---------- ---------------

1 Adana

2 Adıyaman

4 Ağrı

6 Ankara

7 Antalya

9 Aydın

10 Balıkesir

LIKE Karakter alan içinde bulunan değerin yalnızca bir kısmını sorgulamak için kullanılır. Burada % ifadesi de genelde kullanılır. Büyük küçük harf ayrımı yapılacağından sorgu yazılırken buna dikkat edilmesi gerekir. Eğer Access kullanılıyorsanız % yerine * kullanmalısınız. Ayrıca string ifadeleri tek tırnak içine değil çift tırnak içine almalısınız.

SELECT * FROM il WHERE il_adi LIKE "A*"; il_adi A ile başlayan kayıtlar. (Access için)

SELECT * FROM il WHERE il_adi LIKE 'A%'; il_adi A ile başlayan kayıtlar.

IL_KODU IL_ADI

---------- --------

1 Adana
2 Adıyaman
3 Afyon
4 Ağrı
5 Amasya
6 Ankara
7 Antalya
8 Artvin
9 Aydın

SELECT * FROM il WHERE il_adi LIKE '%a'; il_adi a ile biten kayıtlar.

IL_KODU IL_ADI
------- -------
1 Adana
5 Amasya
6 Ankara
7 Antalya

SELECT * FROM il WHERE il_adi LIKE '%s%'; il_adi içinde s harfi geçenler

IL_KODU IL_ADI
-------- ----------
5 Amasya
10 Balıkesir

Ders 2 personel, kurum,departman tablolarına gözat

personel

per_no	per_adi	per_soyadi	per_kurum_kodu
40651092	MEHMET	AKYÜREK	111943

46404046	FATMA	ŞAHİN-ONAY	111943
48693196	İHSAN	ATEŞ	111918
49691388	AKGÜL	YILMAZ	111918
49696089	ABDULLAH	ESER	111943
50643261	MEHMET İHSAN	PEKMEZOĞLU	111918
64697143	ARİF	EFE	111906
41941157	AYSEL	KAYGUN	268532
42691389	KOCA	ŞİRİN	111607
43241232	OSMAN	KARAPINAR	111715
43691338	BEKİR	AYDIN	280543
43695150	MUSTAFA	DÜLDÜL	374215
43911080	MEHMET EMİN	ÇOKAN	473392
44557104	MUSTAFA	ÖZLÜ	473487
44691441	MEHMET	KARAMAN	244170
44691443	MUSA KAZIM	ÖRNEK	111656
44731206	SEVGİ GÜLŞEN	ÖNDERLİ	111597
45696146	MEHMET	YILDIZ	111607
45833157	ŞAHMAN	YILDIRIM	444123
45836109	HALİL	YILDIZ	113680
46661279	ZEYNEL	BUTEV	443703
46697250	ABDİ	TOPAKTAŞ	443715
46833136	İMAM	ERDOĞDU	211562
47506028	GÜLTEN	TURHAN	111835
47696097	SADULLAH	KARTAL	111872
47835108	ALİ	AYATA	268532
47849033	OSMAN	DENİZ	443715
47922026	AHMET	BİNARDAN	246886
48566170	ZEKİ	TORAMAN	473152
48703024	OSMAN	YAPÇA	336860
49696114	TASİN	GÜNGÖR	342828
49697408	KEMAL	AKTAŞ	215822
49702033	MUSTAFA	DÜNDAR	113680
49844045	M.EMİN	ÖNER	270814
49882068	HAYDAR	TEKİN	111668
49997023	ÖMER	DEMİRCAN	473259
50580089	YUSUF	DOĞANER	270814
50691303	HAMZA	KURTOĞLU	111668
50691638	SALİH	OBA	444123
50694061	CEZMİ	KIZILKAYA	111872
50697508	OSMAN	ÇOŞKUNOĞLU	111752
50697512	MEHMET	GÖZÜKARA	111536
50697537	RAHİME	KARAKUŞ	443703
50896193	MAHMUT	DAYIKARACA	352006
51354054	HANİFE	ÇAPÇI	790157
51580060	BEKIR	KELEŞER	270814

51691471	HANDAN	TATLI	111656
51691597	ESİN	CANAK	374107
51691640	KEMAL	KALKAN	111536
51693200	OSMAN	OZAN	113680

Kurum

kur_kodu	kur_ilce_kodu	kur_adi
111918	100	İl Milli Eğitim Müdürlüğü
111906	100	İl Eğitim Araçları ve Donatım Merkezi(ASO)
111943	100	Sağlık Eğitim Merkezi
733321	101	Süreyya Nihat Oral İlköğretim Okulu
323158	101	Adasokağı Lisesi
816584	101	Oğuz Kağan Köksal Görme Engelliler ilköğ.O
817959	101	Bahçeşehir İlköğretim Okulu
818894	101	Cumhuriyet Anaokulu
846449	101	Seyhan İlköğretim Okulu
846425	101	Yenişehir İlköğretim Okulu
846437	101	2000 Evler İlköğretim Okulu
320703	101	Şakirpaşa Lisesi
348168	102	Öğretmen Evi
375149	102	Büyüksofulu İlköğretim Okulu
337147	102	Sinanpaşa İlköğretim Okulu
337302	102	Akören İlköğretim Okulu
214088	102	Halk Eğitim Merkezi
111955	102	İlçe Milli Eğitim Müdürlüğü
428712	104	Kurtkulağı Orhanekinci İlköğretim Okulu
428773	104	Besocak İlköğretim Okulu
428797	104	Ataturk İlköğretim Okulu
428819	104	Ayse Malaz İlköğretim Okulu
428832	104	Fevzı Cakmak İlköğretim Okulu
428820	104	Dumlupınar İlköğretim Okulu
442710	106	Tortulu Ziyelli İlköğretim Okulu
442709	106	Yerebakan İlköğretim Okulu
442663	106	Tokmanaklı İlköğretim Okulu
442651	106	Sahmuratlı İlköğretim Okulu
442161	106	Çandırlar-Bekirhacılı İlköğretim Okulu
442053	106	Akkaya İlköğretim Okulu
441993	106	Belenköy İlköğretim Okulu

442041	106	Akoluk İlköğretim Okulu
790001	107	OTLUK İLKÖĞRETİM OKULU
790050	107	Ü.ORTAEĞRİÇAM İLKÖĞRETİM OKULU
790108	107	ALAYBEYİ İLKÖĞRETİM OKULU
337255	107	Mehmet Akif İlköğretim Okulu
374264	107	Cumhuriyet İlköğretim Okulu
322883	107	İmamoğlu Çok Programlı Lisesi
826526	107	Hürriyet İlköğretim Okulu
392081	109	Örcün İlköğretim Okulu
392093	109	Çevlik İlköğretim Okulu

Depertman

Dep Kodu	Dep Adi
1	Muhasebe
2	Satış
3	İnsan Kaynakları

Alan (Sütun Birleştirmek ve sütunlara isim vermek)

Liste alırken bazı alanları birleşik olarak almanız gerekebilir. Personel Tablomuzda kişinin adı ve soyadı farklı alanlarda kayıtlı ve biz bu iki alanı birleşik almak istiyoruzbu durumda || (pipe) ifadesini kullanmalıyız.

```
SELECT per_no,per_adi||per_soyadi from personel where per_kurum_kodu=111918;
```

```
PER_NO     PER_ADI||PER_SOYADI
48693196İHSANATEŞ
49691388AKGÜLYILMAZ
```

50643261MEHMETPEKMEZOĞLU

Yalnız burada isim ve soyad arasında boşluk eklemediğimiz için isim soyadı birleşik yazdı. Aynı mantıkla boşluğu da ekleyebiliriz.(tek tırnak içerisinde iki adet boşluk)

```
SELECT per_no,per_adi||'  '||per_soyadi from personel where per_kurum_kodu=111918;
```

```
PER_NO    PER_ADI||''||PER_SOYADI
48693196İHSAN ATEŞ
49691388AKGÜL YILMAZ
50643261MEHMET  PEKMEZOĞLU
```

Şimdi oldu yalnız bir problemimiz daha var liste başlığının kötü görüntüsü onu da aşağıdaki şekilde çözebiliriz. (Sütunlara isim tanımlama)

```
SELECT per_no,per_adi||'  '||per_soyadi AdSoyad from personel where
per_kurum_kodu=111918;
```

```
PER_NO    ADSOYAD
48693196İHSAN ATEŞ
49691388AKGÜL YILMAZ
50643261MEHMET  PEKMEZOĞLU
```

İki alanı birleştirmek zorunda değiliz sabit bir ifadeyi de ekleyebiliriz.

```
SELECT il_kodu,il_adi||'  '||'Kenti' FROM il WHERE il_kodu<6;
```

```
IL_KODU IL_ADI||''||'KENTI'
1       Adana Kenti
2       Adıyaman Kenti
3       Afyon Kenti
4       Ağrı Kenti
5       Amasya Kenti
```

Birden fazla tablo ile Sorgu Yazmak

İlişkisel veritabanlarında, tablolara en hızlı ulaşım ve verilerin az yer kaplaması düşünülerek tasarım yapılır. Bu anlamda biz de tablo tasarımlarımızı bu ölçülere dikkat ederek yapmaya çalıştık. Verilen örnek veri tablolarında personel ile kurum dikkate alınacak olursa;
Personel tablosunda per_no,per_adi,per_soyadi,per_kuurum_kodu yer alıyor. Kurum adı değil kurum kodu yer alıyor. Peki neden ?

PER_NO	PER_ADI	PER_SOYADI	PER_KURUM_KODU
48693196	İHSAN	ATEŞ	111918
49691388	AKGÜL	YILMAZ	111918
50643261	MEHMET	PEKMEZOĞLU	111918

111918 koduna sahip kurumda üç kişi çalışıyor. Bu sayı tabii ki gerçek bir veritabanında yüzlerce olabilir. Eğer biz personel tablosuna kurum kodu değil de kurum adını girseydik her personel için bir kurum adı hanesi yer alacaktı ve aşağıda görüldüğü gibi personel başına kurum adi için 'İl Milli Eğitim Müdürlüğü' veri girilecekti ve toplam 25 Byte 'lık bir yer gerekecekti. Oysa biz personel tablosuna kurum adı yerine kurum kodu girerek 4 byte ile bu veriyi ifade edebiliyoruz.

KUR_KODU	KUR_ILCE_KODU	KUR_ADI
111918	100	İl Milli Eğitim Müdürlüğü
111906	100	İl Eğitim Araçları ve Donatım Merkezi(ASO)
111943	100	Sağlık Eğitim Merkezi

Peki bu durumda biz personelin çalıştığı kurumu görüntülemek istersek ne yapmalıyız ? İşte bu durumda SQL yazarken birden fazla tablodan yararlanmak zorundayız. Yukarıda görülen personel tablosundaki per_kurum_kodu ile kurum tablosundaki kur_kodu alanları bağlantı alanlarıdır. SQL yazarken bu iki alanı join etmeliyiz;

```
SELECT per_no,per_adi,per_soyadi,kur_adi
FROM personel,kurum
WHERE per_kurum_kodu=kur_kodu and kur_kodu=111918;
```

PER_NO	PER_ADI	PER_SOYADI	KUR_ADI
48693196	İHSAN	ATEŞ	İl Milli Eğitim Müdürlüğü
49691388	AKGÜL	YILMAZ	İl Milli Eğitim Müdürlüğü
50643261	MEHMET	İHSAN PEKMEZOĞLU	İl Milli Eğitim Müdürlüğü

Select :Personel tablosundan per_no,per_adi,per_soyadi ve kurum tablosundan kur_adi listelenecek alanlar..
From :personel,kurum tablosundan alınacak..
Where: per_kurum_kodu=kur_kodu na eşitse ve per_kurum_kodu=111918 ise..

Burada ilk bağlantılı bir SQL komutunu gerçekleştirmiş olduk.

Birleştirilecek tablo sayısı ikiden çok (örneğin onlarca) olabilir. Yukarıda personelin çalıştığı kurumu görüntülrmiştik. Departman tablomuzla da bağlayarak kişilerin çalıştığı bölümü de görüntüleyelim.

```
SELECT
per_no,per_adi,per_soyadi,dep_adi,kur_adi
```

```
FROM
personel,departman,kurum
WHERE
per_departman=dep_kodu and
per_kurum_kodu=kur_kodu and
kur_kodu=111918;
```

PER_NO	PER_ADI	PER_SOYADI	DEP_ADI	KUR_ADI
48693196	İHSAN	ATEŞ	Satış	İl Milli Eğitim Müdürlüğü
50643261	MEHMET	PEKMEZOĞLU	Satış	İl Milli Eğitim Müdürlüğü
49691388	AKGÜL	YILMAZ	İnsan Kaynakları	İl Milli Eğitim Müdürlüğü

Personellerimizin maaşları da MAAS isimli bir tabloda tutuluyor. Bu tabloda primary key olmayacak çünkü bir personelin her hay aldığı maaş bu tabloya yazılacak. Yani bir yıl için düşünülecek olursa bir kişiye ait 12 adet kayıt bu tabloda yer alacatır. Biz bu tür ilişkilere master-detail diyoruz. Burada master personel tablosu detail ise maas tablosudur. Bu durumda bir kişiye ait maaş dökümlerini almak istediğimizde personel ve maaş tablolarını kullanmak suretiyle aşağıdaki SQL cümleciğini yazabiliriz.

```
SELECT per_no,per_adi,per_soyadi,maas_ay,maas
FROM personel,maas
WHERE per_no=maas_kodu and per_no=48693196;
```

PER_NO	PER_ADI	PER_SOYADI	MAAS_AY	MAAS
48693196	İHSAN	ATEŞ	1	26000
48693196	İHSAN	ATEŞ	2	27000
48693196	İHSAN	ATEŞ	3	33000
48693196	İHSAN	ATEŞ	4	2000

Yukarıda görüldüğü gibi maaş tablosunda ilgili kişiye ait 4 adet kayıt olduğundan 4 kayıt görüntülendi. Maaş tablosunda kişinin hangi ayda ne kadar ücret aldığı belli ancak ayların ismi değil kaçıncı ay olduğu görünüyor. Ağer ayların ismini de görüntülemek için aylar isimli tabloyu da kullanmalıyız.

```
SELECT per_no,per_adi,per_soyadi,ay_adi,maas
FROM personel,aylar,maas
WHERE per_no=maas_kodu and
maas_ay=ay_kodu and
per_no=48693196;
```

PER_NO	PER_ADI	PER_SOYADI	AY_ADI	MAAS
48693196	İHSAN	ATEŞ	Ocak	26000
48693196	İHSAN	ATEŞ	Şubat	27000
48693196	İHSAN	ATEŞ	Mart	33000
48693196	İHSAN	ATEŞ	Nisan	2000

devam edecek...