**GÜN: 1 TARİH: 09.07.2007**

**SAYFA:1**

**STAJDA İLK GÜN…**

Stajıma başlamak için ‘İSKİ BİLGİ İŞLEM MERKEZİ’ ne geldim.Yapacağım staj ile ilgili kısa bilgiler verildi ve çalışacağım bilgisayar gösterildi.Stajyerlerin kullanacağı ayrı bir oturum vardı.Bilgisayarda stajla ilgili detaylı bilgilerin bulunduğu klasörlere baktım.Staj süresince yapılacak çalışmalarda takip edilecek sıra aşağıdaki gibi belirtilmiştir:

* **SQL İLE ALAKALI BİLGİLERİN ÖĞRENİLMESİ. (GEREKLİ DÖKÜMANLARIN BULUNDUĞU KLASÖRLERİN YERİ BELİRTİLMİŞ)**
* **VERİ MODELLEME. (DÖKÜMANLARIN BULUNDUĞU KLASÖRÜN YERİ BELİRTİLMİŞ)**
* **STAJ SÜRESİNCE ÖRNEK PROJE OLARAK SİPARİŞ TAKİP SİSTEMİ GELİŞTİRİLECEKTİR.**
* **GELİŞTİRME ARACI OLARAK ORACLE DEVELOPER 2000 KULLANILMASI. (FAYDALI DÖKÜMANLARIN BULUNDUĞU YERLER BELİRTİLMİŞ)**
* **MODELLEME İÇİN ERWİN KULLANILMASI.**

Belirtilen klasörlere kısaca baktım.İlk önce SQL çalışmaya başladım.

**SQL (STRUCTURED QUERY LANGUAGE) YAPISAL SORGULAMA DİLİ**

**SQL’İN TARİHÇESİ**

1970 IBM Laboratuarlarında geliştirilmeye başlandı.

Ocak 1970’te,Dr E.F.Codd’un “Relational Model of Data For Large Shared Data Banks” isimli ilişkisel veritabanı temel makalesi ile ortaya çıkmıştır.

1979 ORACLE ilk ilişkisel veritabanını piyasaya çıkardı.

**SQL’İN ÖZELLİKLERİ**

* SQL prosedürel olmayan ve set temelli dildir.

Set temelli çalışma, bir komutun veri tabanına gönderilip çalıştırılması, arkasından

2. bir komut gönderilmesi şeklindeki çalışmadır.

**BASİT BİR MODEL**

SQL cümlesi

gönderilir

Veritabanı

Veri geri alınır

* SQL data set üzerindeki operasyonlara izin verir.
* Fiziksel saklama ortamından bağımsız şekilde çalışır.Datanın DBMS sistemlerinden nasıl alınacağı ile ilgili fiziksel işlemlerden bağımsızdır.Burada kastedilen işlemler fiziksel dosya erişimi, verinin yazılacağı işletim sistemi dosyasının belirlenmesi şeklindeki işlemlerdir.SQL, kesin ve tam olarak hangi datanın alınacağını veya hangi data üzerinde işlem yapılacağını tanımlar.

**GÜN: 1 TARİH: 09.07.2007**

**SAYFA:2**

SQL ile ilgili endüstri standartları American National Standards Institute(ANSI) ve

The International Standards Organisation (ISO) tarafından oluşturulmuştur.ANSI-92 standardı en genel SQL standardıdır. Oracle PL/SQL ve Microsoft Transact –SQL (T-SQL) bazı eklemelerle geliştirilmiş farklı versiyonlardır.SQL in en önemli özelliklerinden biri tüm ilişkisel veri tabanları için geçerli bir standart olmasıdır.Access,Oracle,Sqlserver ,DB2 veri tabanlarında temel SQL komutları aynı şekilde kullanılabilir.

**UYGULAMA PROGRAMLAMASINDA SQL**

SQL için ilk ANSI standardı 1986’da çıkarıldı.SQL-89 olarak da bilinen 1989 da koyulan standart uygulama SQL arasında 3 farklı tipte ara yüz tanımlıyordu:

1. Modül dil:Program içinde prosedürler kullanır.Program içinden parametre geçirerek çağrılır.
2. Statik SQL(Embedded SQL):Program kodları arasına SQL ifadeler koymaya imkan verir.
3. Direct invocation

**STATİK SQL:** SQL deyimleri programla birlikte derlenir ve çalışma zamanı aşamasında değiştirilemezler. Bu tip SQL için performans yüksektir fakat çok değişken iş şartlarında uygulama esnek değildir.

**DİNAMİK SQL:** ANSI-92 standart (SQL-92) dile ilaveler yaparak uluslar arası bir standart haline getirdi. Aşağıdaki yeni özellikler eklendi.

🡪Database Connections

🡪Scrollable Cursor

🡪Dynamic SQL

🡪Outer Join

Dinamik SQL ifadelerinin program çalışırken hazırlamasına imkânı verir. Uygulama geliştiricilere ve kullanıcılara esneklik sağlar. Performansı statik SQL’den daha düşüktür. Ancak bu performans kaybı göz ardı edilebilir seviyelerdedir. SQL içerisindeki değişkenlere SQL çalıştırılırken değer atanabilmesine imkân verir.

**SQL İLE YAPILABİLECEKLER:** SQL ile şunları gerçekleştirmek mümkündür:

* Bir veri tabanı yapısının değiştirilmesi
* Sistem güvenlik ayarlarının yapılması ve değiştirilebilmesi
* Veri tabanı veya tablolara kullanıcı erişim haklarının düzenlenebilmesi
* Bir veri tabanını sorgulayarak bilgi elde edilmesi
* Bir veri tabanının güncellenmesi

**SQL TEMEL KOMUT TİPLERİ VE KOMUTLARI**

* **DML:** Veri üzerinde yapılan işlemler

**🡪 SELECT**

**🡪 INSERT**

**🡪 UPDATE**

**🡪 DELETE**

* **DDL:** Verinin yapısı üzerinde yapılan işlemler

**🡪CREATE**

**🡪ALTER**

**🡪DROP**

* **DCL:** Güvenlik işlemleri

**🡪GRANT**

**🡪REVOKE**

* **TRANSACTION KONTROL**

**🡪COMMIT**

**🡪ROOLBACK**

DML, DDL, DCL ve Transaction Kontrol komut tiplerinin bilinmesi özellikle önemlidir. Bu tipler çalışma, yetki verme vb. yapıların anlaşılmasını kolaylaştıracaktır.

**GÜN: 1 TARİH: 09.07.2007**

**SAYFA:3**

**BASİT SQL İFADELERİNİN YAZILMASI**

**SELECT(SORGULAMA)**

SELECT deyimi veri tabanından veriyi alarak kullanıcıya geri döndürür. En çok kullanılan SQL komutlarından biridir.

**SELECT’İN KULLANIMI:**

* Sorgu sonucu istenilen kayıtların seçiminin sağlanması
* İstenilen sütunların sorgulanmasının sağlanması
* Değişik tablolar arasında link oluşturarak veriyi bir araya getirip kullanıcıya sunması

**‘SELECT’ İFADESİNİN YAZIM KURALLARI**

SELECT [DISTINCT],[\*],(sütun listesi)

FROM (tablo listesi)

WHERE (birleştirme (join) şartları ve sorgu şartları)

GROUP BY (sütun listesi)

HAVING (sorgu koşulları)

ORDER BY (sütunlar, ifadeler)

* Köşeli parantezler seçime bağlı alanları belirtmektedir.
* Normal parantezler programcı tarafından belirlenen kısımlardır.

**EN BASİT SQL İFADESİ**

En basit biçimiyle SQL ifadesi,

1. İstenilen tablonun kolonlarının belirtildiği bir SELECT cümleciği,
2. Hangi tablodan veri seçileceğini ifade eden bir FROM cümleciği içermelidir.

SELECT [DISTINCT],[\*],(sütun listesi)

FROM (tablo listesi)

**SQL İFADE YAZIMI KURALLARI**

* SQL komutları büyük ya da küçük harflerle yazılabilir, fark etmez.
* Satırlara bölünebilir. Yani yazılan SQL cümlesi bir ya da daha fazla satır olabilir.
* Satıra bölünürken kelimeler bölünemez.
* Cümleciklerin ayrı satırlara yazımı tercih edilir.
* Kolay okunacak şekilde yazılmalıdır.

**DISTINCT ifadesi:**

Kelime anlamı ayrı, hariç, tek demektir. Deyim olarak aynı alandaki benzersiz olanları bulmayı ifade eder. Sorgu sonucu gelen değerler içerisindeki birden fazla aynı kayıt varsa onları eleyerek tek kayıt olarak gösterilmesini sağlar.

**\* SEÇİCİSİ:**

[\*] Tüm kolonların aynı anda seçilmesi için kullanılır.

SELECT \* FROM (tablo\_ismi)

Eğer bir sınırlayıcı ölçüt koymadan tablodaki tüm bilgilere (tüm kayıtlar ve her bir kayıttaki tüm alanlar) erişmek istendiği DBMS e bildirilirse bütün tablo içeriği kullanıcıya geri döndürülür. Bu sorgulamayı yapan \* yerine ALL parametresi de kullanılabilir.

Belli kolonları seçmek için o kolonların isimleri SELECT ve FROM arasına yazılmalıdır.

**WHERE ifadesi:**

Tablodaki belli kayıtlara erişmek için WHERE kısmında yazılan koşullar kullanılır. Kayıtlar arasından sadece istenilen koşulları taşıyanların seçilebilmesini sağlamak amacıyla WHERE ifadesi kullanılır.

Örneğin, İstanbul’daki departman bilgilerine erişmek için kullanıcı veri tabanı yönetim sistemine isim alanı(sütunu) 34 olan yerdeki kayıtlara erişmek istediğini bildirir.

SELECT \* FROM departman

WHERE il=34

**GÜN: 1 TARİH: 09.07.2007**

**SAYFA:4**

**EXPRESSIONS, CONDITIONS VE OPERATORS**

Sorgulama işleminde kullanılabilecek bazı işlemler

Expression (ifade) tanımı ve kullanımı

Condition (koşul) tanımı ve kullanımı

WHERE cümlesi ile koşullu sorgulama

Operatörler: Matematiksel, karşılaştırma, karakter dönüşümleri, mantıksal ve veri setleri oluşturan operatörler vardır.

**EXPRESSIONS – İFADELER:** SQL cümlesinin geriye değer döndüren, keyword u tamamlayan kısmına “expression” denir.

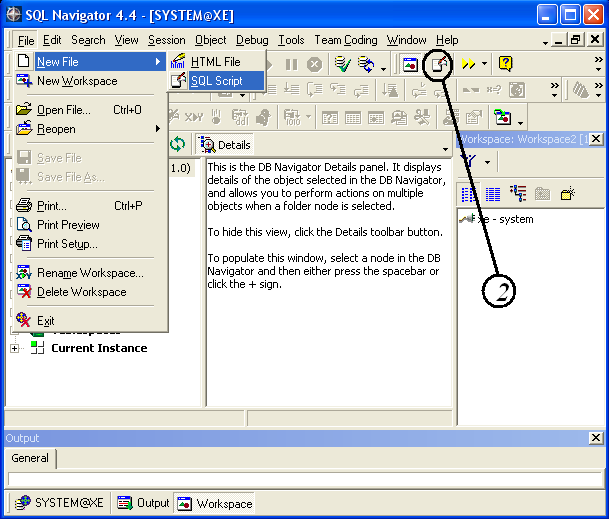
**CONDITIONS – KOŞULLAR:** Eğer database den özel bir grup veriyi elde etmek istiyorsak sorgulamada bununla ilgili koşul veya koşulları kullanmamız gerekir.

Koşullar (conditions) WHERE clause içinde bulunur. Koşul içinde bir değişken (variable) bir sabit (constant) ve bir de karşılaştırma operatörü (comparison operator) bulunur.

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

SQL ile ilgili bilgilere çalışmaya devam ederken bu bilgilerimi uygulayacağım SQL Navigator programını da öğrenmeye başladım. Program aşağıdaki gibidir:

**SQL NAVIGATOR:**

****

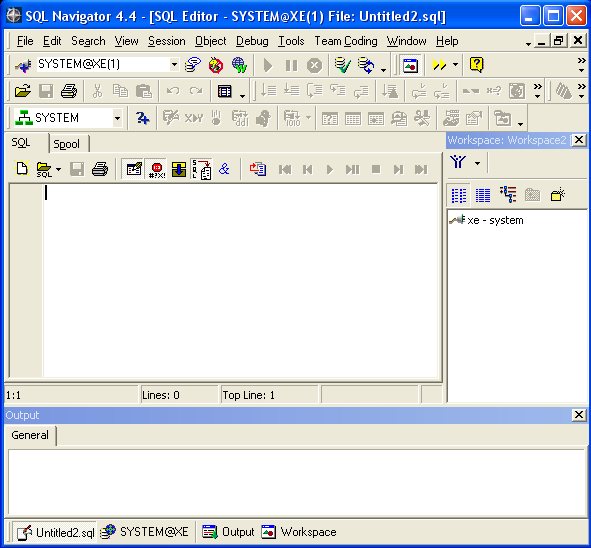
**SQL NAVIGATOR’**DEÇALIŞIRKEN YENI BIR SQL SAYFASI AÇMAK İÇİN ASAGIDAKI YOLLAR IZLENEBILIR:

1. FILE 🡪 NEW FILE 🡪 SQL SCRIPT
2. 2 NUMARA ILE GOSTERILEN ICON TIKLANARAK
3. CTRL+M YAPILARAK YENI SQL SAYFASI AÇILIR**.**

**GÜN: 1 TARİH: 09.07.2007**

**SAYFA:5**

Yeni açılan SQL sayfası aşağıdaki gibi olacaktır:



Kullanacağım programa baktıktan sonra konu çalışmaya devam ettim.

**WHERE Clause ile koşullu sorgulama:** WHERE cümlesinin yazımı şu şekildedir:

WHERE<SEARCH CONDITION>

SELECT, FROM ve WHERE cümleleri en sık kullanılan SQL keywordleridir. WHERE cümlesi ile daha ayırt edici sorgulamalar yapılabilir. WHERE cümlesi kullanılmadığı zaman belirtilen tablodaki sorgulama sonucunda bütün kayıtlar listelenir.

Veri tabanındaki personel isimli bir tablodan WHERE kullanmadan bütün kayıtları getirmek için:

SELECT \* FROM personel;

Ancak özel bir kişiyi arıyorsak WHERE ifadesini kullanmamız gerekir:

SELECT \* FROM personel

WHERE adi=’AYŞE’ ; (Sorgu sonucu adı AYŞE olan kayıtlar listelenecektir.)

**OPERATORS – OPERATÖRLER:** Operatörler istenilen özel koşulları oluşturmak için bir ifade içerisinde kullanılan elemanlardır. Aritmetik, Karşılaştırma,Karakter Operatörleri,

Mantıksal Operatörler,Set Operatörleri,Diğer Operatörler

**Aritmetik Operatörler:**

Toplama (+), Çıkarma (-), Bölme (/), Çarpma (\*), Modül

**GÜN: 2 TARİH: 10.07.2007**

**SAYFA:1**

SQL çalışmaya devam ettim.Ayrıca uygulama yapmaya başladım.

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

**Toplama(+) Operatörü:**Artı operatörünü değişik şekillerde kullanabiliriz.Örneğin,

SELECT adi,maas+1000

FROM personel;

Burada personel tablosunda olmayan yeni bir sütun türetildi.Dikkat edilmesi gereken bir nokta yeni üretilen sütunun başlığının ifadedeki maas+1000 olmasıdır.

Bu operatörler SELECT ifadelerinde temel matematiksel işlemlerin yapılmasını sağlarlar.

**Karşılaştırma Operatörleri**

Eşit(=), Büyük(>), Büyük eşit (>=), IS NULL, Küçük(<), Küçük eşit (<=),

Eşit değildir(<> ya da !=)

Karşılaştırma operatörleri bulundukları ifade içerisinde karşılaştırma işlemi yapar ve TRUE,FALSE ve UNKNOWN sonucu döndürürler.

NULL ARİTMETİĞİ:NULL değeri yok demektir.Veri tabanı açısından bir kaydın belli bir sütununda değer olmaması,değer atanamamış olması,veri bulunmaması gibi anlamlara gelir.Aritmetik ifadelerde NULL bulunması halinde ifadenin sonucu NULL olur.

IS NULL:IS NULL karşılaştırma operatörü NULL değerlerle karşılaştırma yapmak için kullanılan özel bir operatördür.

**Karakter Operatörleri**

**LIKE ve percent(%) ve wildcard(\_)**

String kalıbını tam olarak bilmediğimiz sorgulamalarda istediğimiz bilgilere ulaşamayabiliriz.Arayacağımız tam değeri bilmiyorsak:

* ( % )bir ya da daha fazla değer içeren bir kısım arıyorsak
* ( \_ ) herhangi tek bir karakter arıyorsak

**String birleştirme ( || ):** String leri birleştirir.

**Mantıksal Operatörler**.

AND OPERATÖRÜ: Koyulan iki koşulun da TRUE olması durumunda TRUE değerini,koşullarda birinin sağlanmaması durumunda FALSE değerini döndürür.

OR OPERATÖRÜ: İki koşuldan bir tanesinin TRUE olması durumunda TRUE değerini,iki koşulunda FALSE olması durumunda FALSE değerini döndürür.

NOT OPERATÖRÜ: İfadeyi mantıksal tersi yapar.İfadedeki koşul TRUE üretiyorsa NOT bunu FALSE yapar.

Operatörlerde öncelik sırası: Tüm karşılaştırma operatörleri 🡪 NOT 🡪 AND 🡪 OR

Sırayı değiştirmek için parantez kullanılmalıdır.

**IN Operatörleri:** IN bir dizi içerisinde belli bir değerin bulunması halinde TRUE döner.

**BETWEEN operatörü:** Bir aralıkta kıyaslama yapar.Aralığın küçük değeri önce yazılmalıdır.

**ORDER BY (VERİYİ SIRALAMA)**

SELECT sonucu gelen verinin sıralaması belirsizdir.Eğer veriyi sıralı bir şekilde almak istiyorsak ORDER BY kullanırız.ORDER BY SELECT ifadesinin en sonunda olmalıdır.

**ASC:**Kayıtları artan bir sıralama ile sıralar.

**DESC:**Kayıtları azalan bir sıralama ile sıralar.

**GROUP BY (VERİ GRUPLARI ÜZERİNDEKİ İŞLEMLER)**

Kayıt grupları üzerinde işlem yaparak her grup için bir sonuç üreten fonksiyonlara **grup** **fonksiyonları** denir.Bunlardan birkaçı:

**AVG:** ORTALAMA.Bir sütundaki değerlerin ortalamasını verir.

**COUNT:** TOPLAM SATIR SAYISI.Bir sütundaki değerlerin sayısını bulur.

**MIN:** MINIMUM.Bir sütundaki değerlerin en küçüğünü bulur.

**GÜN: 2 TARİH: 10.07.2007**

**SAYFA:2**

**MAX:** MAXİMUM.Bir sütundaki değerlerin en büyüğünü bulur.

**SUM:** SATIR DEĞERLERİ TOPLAMI.Bir sütundaki değerlerin toplamını verir.

* COUNT(\*) dışındaki tüm grup fonksiyonları NULL değerleri hesaba katmazlar.
* Gruplama sırasında aynı zamanda sıralama da yapılır.

BAZI KURALLAR:

* SELECT ifadesinde grup fonksiyonları kullanıldığında GROUP BY ifadesindekiler haricinde başka sütun seçilemez.Aksi halde hata oluşur.
* SELECT ifadesinde grup fonksiyonları kullanıldığında GROUP BY ifadesindeki tüm sütunlar seçilmiş olmalıdır.Aksi halde hata oluşur.
* WHERE şartı kullanıldığında önce kayıtlar kısıtlanır.Sonra gruplama yapılır.

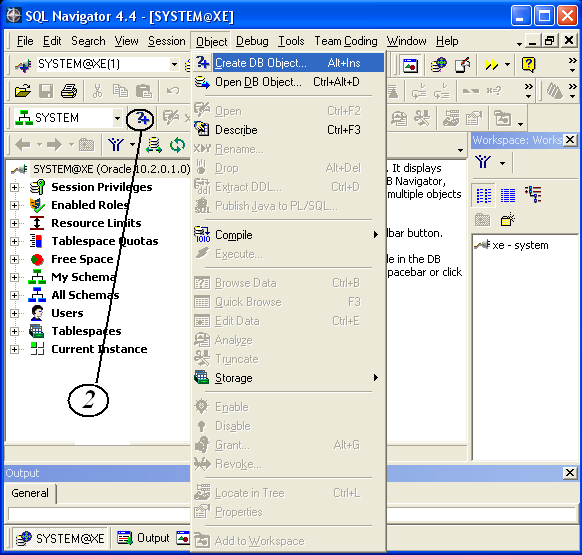
**HAVING:** HAVING ifadesi kaydın sorguca belirlenen değerlere sahip olup olmadığına bakar.GROUP BY ile birlikte kullanılır.Grup fonksiyonları yalnızca HAVING şartı içerisinde kullanılabilirler.WHERE şartında gelecek kayıtları belirlediğimiz gibi HAVING şartında da gelecek grupları belirleriz.

* Önce satırlar gruplanır.
* Sonra grup fonksiyonu uygulanır.
* En sonunda HAVING şartı uygulanarak hangi grupların seçileceği belirlenir.
* Önce WHERE şartındaki kısıtlamalar yapılır daha sonra HAVING teki şartlar çalışır.

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

SQL ile ilgili çalıştığım konuları uygulamaya başladım.Önce **UYEE** adında bir tablo oluşturdum.Oluşturmak için aşağıdakileri yaptım:

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



Yeni bir tabloyu kod yazmadan oluşturmak için:

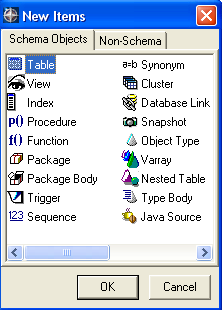
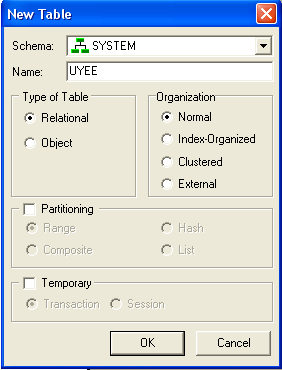
1. OBJECT 🡪 CREATE DB OBJECT
2. 2 numara ile gösterilen icon ı tıklarız.

Daha sonra gelen seçeneklerden TABLE seçilir ve OK denir.

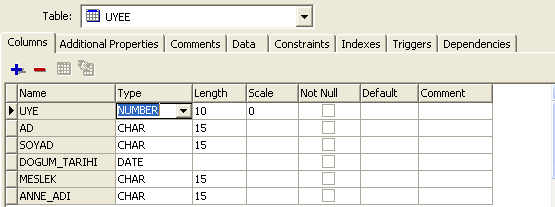
**GÜN: 2 TARİH: 10.07.2007**

**SAYFA:3**

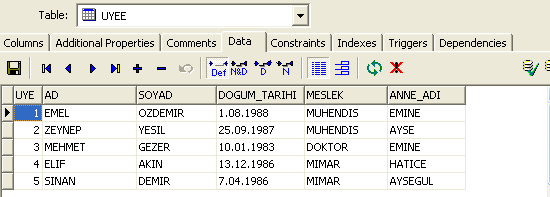
Seçenekler şu şekilde gösterilir:

****

New Items’ dan Table ı seçerek OK dedikten sonra yanındaki New Table gelecektir ve burada da tablonun ismini yazarak OK tuşunu tıklamamız gerekir.Tabloyu bu şekilde oluşturduktan sonra kolonları aşağıdaki gibi oluşturdum:



DATA kısmında verileri girdim ve commit i tıklayarak kayıt ettim.



**GÜN: 2 TARİH: 10.07.2007**

**SAYFA:4**

Tabloya bilgileri kayıt ettikten sonra uygulama yapmaya başladım.Yaptığım uygulamalar SQL de yazılış biçimiyle aşağıdakilerdir:

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

SELECT \* FROM UYEE; */\* Tablodaki bütün verileri getirir \*/*

SELECT UYE,AD,SOYAD

FROM UYEE; */\* Tablodan sadece uye,ad,soyad bilgilerini getirir \*/*

SELECT UYE,AD,SOYAD,DOGUM\_TARIHI

FROM UYEE

ORDER BY AD ASC; */\* AD lara göre alfabetik sırada önce gelenden sonra gelene doğru sıralama yapıyor,ekranda üye,ad,soyad,dogum\_tarihi bilgilerini gösteriyor \*/*

SELECT UYE,AD,SOYAD,DOGUM\_TARIHI

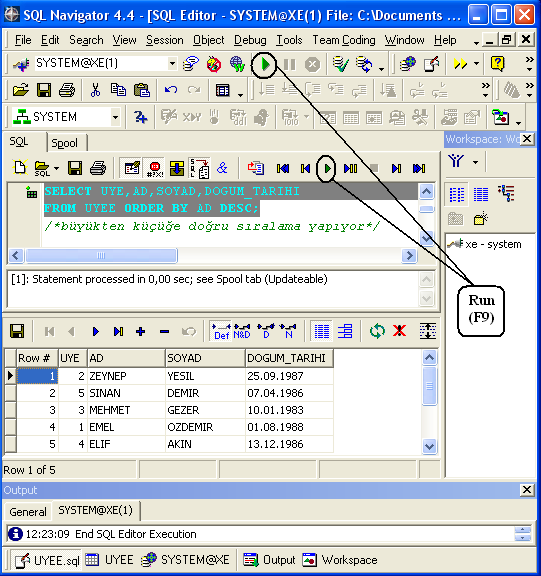
FROM UYEE

ORDER BY SOYAD ASC; */\* SOYAD lara göre alfabetik sırada önce gelenden sonra gelene doğru sıralama yapıyor,ekranda üye,ad,soyad,dogum\_tarihi bilgilerini gösteriyor \*/*

SELECT UYE,AD,SOYAD,DOGUM\_TARIHI

FROM UYEE ORDER BY AD DESC; */\* AD lara göre alfabetik sırada sonra gelenden önce gelene doğru sıralama yapıyor \*/*

**Bu kodun çalıştırılmasınınsonucuaşağıdagörülmektedir.Kodu çalıştırmak şekildeki iconlarla ve F9 ile olur**:

**

**GÜN: 2 TARİH: 10.07.2007**

**SAYFA:5**

SELECT UYE,AD,SOYAD,DOGUM\_TARIHI

FROM UYEE

ORDER BY UYE ASC; /\* UYE numarasına göre küçükten büyüğe doğru sıralama yapıyor \*/

SELECT UYE,AD,SOYAD,DOGUM\_TARIHI

FROM UYEE

ORDER BY DOGUM\_TARIHI DESC; /\* Doğum tarihine göre büyükten küçüğe doğru sıralama yapıyor \*/

SELECT AD,SOYAD,MESLEK

FROM UYEE

WHERE MESLEK='MUHENDIS'

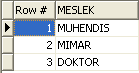
ORDER BY AD ASC; */\* Mesleği MUHENDIS olanlar AD a göre alfabetik sırada önce gelenden sonra gelene doğru sıralı gelir \*/*

SELECT DISTINCT MESLEK

FROM UYEE

ORDER BY MESLEK DESC; */\* Birbiri ile aynı olan mesleklerden bir tane görüntülenir ve mesleğe göre sıralı olarak gösterilir \*/*

Bu kod çalıştırılınca sonuç aşağıdaki gibi olur:

**

SELECT AD,SOYAD,MESLEK

FROM UYEE

WHERE MESLEK='MUHENDIS'

ORDER BY AD DESC; */\** *Mesleği MUHENDIS olanlar AD a göre alfabetik sırada sonra gelenden önce gelene doğru sıralı gelir \*/*

SELECT AD,SOYAD,MESLEK

FROM UYEE

WHERE AD='EMEL' AND MESLEK='MUHENDIS'

ORDER BY AD ASC; */\* Mesleği MUHENDIS ve adı EMEL olanlar AD a göre alfabetik sırada önce gelenden sonra gelene doğru sıralı gelir \*/*

SELECT AD,SOYAD,MESLEK

FROM UYEE

WHERE MESLEK='MIMAR' OR MESLEK='MUHENDIS'

ORDER BY AD ASC; */\* Mesleği MUHENDIS veya MIMAR olanlar AD a göre alfabetik sırada önce gelenden sonra gelene doğru sıralı gelir \*/*

SELECT \* FROM UYEE

WHERE NOT MESLEK IN('MIMAR','DOKTOR'); */\* Mesleği MIMAR veya DOKTOR olanların dışındakileri gösterir \*/*

SELECT \* FROM UYEE

WHERE MESLEK IN('MIMAR','DOKTOR'); */\* Mesleği MIMAR veya DOKTOR olanları gösterir \*/*

SELECT \* FROM UYEE

WHERE NOT MESLEK IN('MIMAR','MUHENDIS')

ORDER BY SOYAD DESC; */\* Mesleği MUHENDIS veya MIMAR olanların dışındakiler SOYADa göre alfabede sonra gelenden önce gelene doğru sıralı gelir \*/*

**GÜN: 3 TARİH: 11.07.2007**

**SAYFA:1**

SQL de uygulama yapmaya devam ettim,konu da çalıştım.Yaptığım uygulamalar ve çalıştığım konular(özet olarak) aşağıdakilerdir:

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

SELECT \* FROM UYEE

WHERE MESLEK IN('MIMAR','MUHENDIS') and ANNE\_ADI='EMINE'; */\* Mesleği MUHENDIS veya MIMAR olup anne adı EMINE olanları getirir \*/*

Bu kod çalıştırılınca sonuç aşağıdaki gibi olur:



SELECT \* FROM UYEE

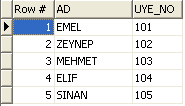
WHERE MESLEK IN('DOKTOR','MUHENDIS') and ANNE\_ADI='EMINE'

ORDER BY AD ASC; */\* Mesleği DOKTOR veya MUHENDIS olup Anne adı EMINE olanları AD a göre sıralı getirir \*/*

SELECT ad,uye+100 uye\_no

FROM UYEE; */\* uye kolonundaki numaralara 100 eklenir ve yeni kolonun adı uye\_no yapılır \*/*

Bu kod çalıştırılınca sonuç aşağıdaki gibi olur:

**

SELECT ad,soyad,meslek,uye-1

FROM UYEE ORDER BY AD ASC; */\* uye kolonundaki numaralardan 1 çıkarılır ve bilgiler AD a göre sıralı gelir \*/*

Bu kod çalıştırılınca sonuç aşağıdaki gibi olur:



SELECT ad||soyad,uye/2 uye\_no FROM UYEE; */\* uye kolonundaki numaraları 2 ye böler ve yeni kolonun adını UYE\_NO yapar ve ad ile soyad kolonlarını birleştirir \*/*

SELECT ad,soyad,uye\*5 FROM UYEE; */\* uye kolonundaki numaraları 5 ile çarparak gösterir \*/*

SELECT count(\*) FROM UYEE

WHERE meslek='DOKTOR'; */\* Mesleği DOKTOR olan kaç kişi olduğunu gösterir \*/*

SELECT sum(uye) FROM UYEE

WHERE meslek='MIMAR'; */\* Mesleği MIMAR olanların uye numaraların toplamını gösterir \*/*

**GÜN: 3 TARİH: 11.07.2007**

**SAYFA:2**

SELECT ad,soyad,meslek,anne\_adi FROM UYEE

WHERE ad LIKE '%NA%' or soyad like '%SIL%' or ANNE\_ADI like '%TI%'

ORDER BY AD ASC; */\*Adının içinde NA veya Soyadının içinde SIL veya Anne adının içinde TI geçenleri AD a göre sıralı gösterir \*/*

Bu kod çalıştırılınca sonuç aşağıdaki gibi olur:



SELECT count(\*) FROM UYEE; */\* Toplam kayıt sayısını gösterir \*/*

SELECT min(dogum\_tarihi)

FROM UYEE; */\* Minimum doğum tarihini gösterir \*/*

SELECT max(dogum\_tarihi)

FROM UYEE; */\* Maximum doğum tarihini gösterir \*/*

SELECT avg(uye) FROM UYEE; */\* uye numaralarının ortalamasını gösterir \*/*

SELECT meslek,max(dogum\_tarihi) FROM uyee

GROUP BY meslek; */\* Mesleklere göre gruplama yapar ve her gruptaki Maximum doğum tarihini gösterir \*/*

SELECT meslek,avg(uye) ort\_uye,max(dogum\_tarihi)

FROM uyee

GROUP BY meslek

HAVING avg(uye)<4; */\* Mesleğe göre gruplar ve ortalama uye numarası 4 ten küçük olan grupları gösterir \*/*

SELECT \* FROM UYEE

where ad like '\_\_E%'; */\* Adının baştan 3.karakteri E olanı gösteriyor\*/*

SELECT \* FROM uyee

WHERE uye BETWEEN 2 AND 4; */\*uye numarası 2 ile 4 arası olanları gösterir\*/*

SELECT \* FROM UYEE

WHERE uye+100>102 and ad like 'E%'; */\* uye numarasının 100 fazlası 102den büyük olanlar arasından isminin baş harfi E olanları buluyor \*/*

**----------------------------------------------------------------------------------------------**

SQL de kaldığım konulardan çalışmaya devam ettim.Konu çalışmalarını okuyarak ve yazarak yapıyorum.Daha sonra da çalıştıklarımı uyguladım.

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

**KARAKTER FONKSİYONLARI**

BÜYÜK KÜÇÜK HARF DEĞİŞTİRME:

**🡪 UPPER():** String i büyük harfe çevirir.

**🡪 LOWER():** String i küçük harfe çevirir.

**🡪 INITCAP():** String in ilk harfini büyük,diğerlerini küçük yapar.

KARAKTER DÖNÜŞÜM:

**🡪 CONCAT():** Birleştirme

**🡪 SUBSTR():** Bölüm ayırma

**🡪 LENGT():** Karakter uzunluğu

**🡪 INSTR():** Karakter yeri arama

**🡪 LPAD | RPAD :** Boşluk ekleme

**🡪 TRIM():** String in bazı kısımlarını alır

**🡪 REPLACE():** Karakter değiştirme

**GÜN: 3 TARİH: 11.07.2007**

**SAYFA:3**

**NUMARA FONKSİYONLARI:**

* **ROUND():** Yakın değere yuvarlama
* **TRUNC():** Belli bir ondalık kısma kadar kesme
* **MOD():** Kalan değer

**TARİH FONKSİYONLARI:**

* **SYSDATE:** Sistem tarihini döndürür.
* **MONTHS\_BETWEEN:** İki tarih arası ay sayısı
* **ADD\_MONTHS:** Ay ekleme
* **NEXT\_DAY:** Belirlenen bir sonraki gün
* **LAST\_DAY:** Ayın son günü
* **ROUND:** Formata göre yuvarlama
* **TRUNC:** Formata göre kesme

**DÖNÜŞÜM FONKSİYONLARI:** Karakter,tarih ve sayı arasında dönüşüm işlemleri:

İÇ DÖNÜŞÜM: ORACLE SERVER tarafından otomatik yapılan dönüşümler.

VARCHAR2 🡪 NUMBER , VARCHAR2 🡪 DATE

NUMBER 🡪 VARCHAR2 , DATE 🡪 VARCHAR2

HARİCİ DÖNÜŞÜM: Fonksiyonlar vasıtasıyla

* **TO\_CHAR() :** (DATE 🡪 KARAKTER) Veriyi özel formatta almak ve göstermek.

(NUMBER 🡪 KARAKTER) Veriyi özel formatta almak ve göstermek.

* **TO\_NUMBER() ve TO\_DATE():** Belli formattaki karakter veriyi numaraya ve tarihe dönüştürmek için kullanılır.

**ŞART FONKSİYONLARI:**

* **CASE:** Doğru olan şarta göre sonuç belirle
* **DECODE:** Eşit olan şarta göre sonuç belirle

**----------------------------------------------------------------------------------------------**

Bu konularla ilgili olarak yaptığım örnekler:

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

SELECT UPPER(ad||soyad)

FROM uyee; /\**AD ve SOYAD kolonlarını* *birleştirir büyük harflerle gösterir*\*/

SELECT LOWER(ad),LOWER(soyad)

FROM uyee; /\* *AD ve SOYAD ları küçük olarak gösterir* \*/

SELECT INITCAP(ad),UPPER(soyad)

FROM uyee; /\* *AD ı baş harfi büyük olarak,SOYAD ı büyük olarak gösterir* \*/

Bu kod çalıştırılınca sonuç aşağıdaki gibi olur:



SELECT TO\_CHAR(SYSDATE,'YYYY')

FROM DUAL; /\* *Sistemin tarih bilgisinden yıl bilgisini gösteriyor* \*/

Bu kod çalıştırılınca sonuç aşağıdaki gibi olur:

**

**GÜN: 3 TARİH: 11.07.2007**

**SAYFA:4**

SELECT ROUND (SYSDATE, 'year') - TRUNC (SYSDATE, 'year')

FROM DUAL; */\* İçinde bulunulan yılın gün sayısı \*/*

Bu kod çalıştırılınca sonuç aşağıdaki gibi olur:

**

SELECT ROUND (TO\_DATE (('16.' ||

TO\_CHAR (SYSDATE, 'mm.yyyy')),'dd.mm.yyyy'),'month')-

TRUNC (SYSDATE, 'month');

FROM DUAL; */\* İçinde bulunulan ayın gün sayısı \*/*

SELECT TO\_CHAR (SYSDATE, 'HH:MI:SS')

FROM DUAL; */\* Sistem tarihinden saat bilgisini gösteriyor \*/*

SELECT ad,soyad

FROM uyee

WHERE SUBSTR(ad,3,1)

IN('E','I'); */\* Adının 3.harfi E veya I olanları gösteriyor \*/*

Bu kod çalıştırılınca sonuç aşağıdaki gibi olur:

**

SELECT INITCAP(ad)||UPPER(soyad)

FROM uyee

WHERE SUBSTR(ad,3,2)in('EL','YN'); */\* Adının 3.harfinden itibaren 2*

*karakteri EL veya YN olanların adlarını baş harfi büyük ve soyadları büyük gösterir \*/*

Bu kod çalıştırılınca sonuç aşağıdaki gibi olur:

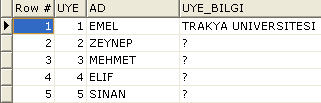
**

SELECT uye,ad,DECODE(uye,'1','TRAKYA UNIVERSITESI','?')uye\_bilgi

FROM uyee; */\* Uye numarası 1 olanın UYE\_BILGI kolonuna*

*Trakya Universitesi,diğerlerine de ? yazar\*/*

Bu kod çalıştırılınca sonuç aşağıdaki gibi olur:

**

*-------------------------------------------------------------------------*

**ALT SORGULAR :**

Alt sorgu kullanımı birinci sorgunun sonuçlarını ikinci sorguya değer olarak gönderen iki ayrı sorguyu ard arda çalıştırmakla eşdeğerdir.

**Alt sorgu tipleri :**

* **Tek satır**
* **Çoklu satır**
* **Çoklu sütun**

**GÜN: 3 TARİH: 11.07.2007**

**SAYFA:5**

**Tek Satır :** Alt sorgu yalnızca tek bir satır döndürür.Bu değerle üst sorgu değerleri karşılaştırılır.Bu işlemde kullanılabilecek operatörler (=,<,>) vb operatörlerdir.Grup oeratörleri de kullanılabilir.

ÖRNEK :

SELECT \*

FROM UYEE

WHERE UYE>(SELECT UYE

FROM UYEE

WHERE MESLEK='DOKTOR'); */\* Mesleği DOKTOR olan kişinin uye numarasından(3) büyük numaraya sahip kişilerin bilgilerini bulur \*/*

Bu kod çalıştırılınca sonuç aşağıdaki gibi olur:

**

**Çoklu Kayıt:** Birden fazla kayıt döndüren alt sorgulardır.IN operatörü kullanılır.

ÖRNEK:

SELECT \*

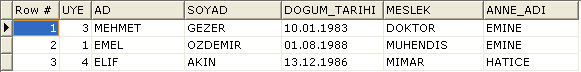
FROM UYEE

WHERE UYE IN(SELECT MIN(UYE)

FROM UYEE

GROUP BY MESLEK); */\* Meslek gruplarından her grubun min uye numarasını ve bu numaralara sahip kişilerin bilgilerini bulur \*/*

Bu kod çalıştırılınca sonuç aşağıdaki gibi olur:



**Çoklu Sütun:** Önceden tek sütunu tek sütunla karşılaştıran ifadeler yazdık.Birden fazla sütun arasında aynı anda karşılaştırma yapmak için Çoklu Sütun karşılaştırma teknikleri kullanılır.Yazılışı: SELECT sütun,sütun,…

FROM tablo

WHERE (sütun,sütun) IN (SELECT sütun,sütun

FROM tablo)

Bu durumda karşılaştırmadaki tüm sütunlar birleştirilerek alt sorgudaki sütunların birleşimi ile işlem yapılmış gibi olur.

**ALT SORGULARIN GELİŞMİŞ BAZI HALLERİ**

* Bir alt sorguyu SQL ifadesinin FROM kısmında yazabiliriz.

SELECT \*

FROM (SELECT sütun FROM tablo) a

Burada kullanılan alias’la çeşitli kolonlara referans edilebilir.

* Sorguda üstten alta veri geçişi:Alt sorguya üst sorgudan değer gönderip her üst sorgu satırı için farklı sonuç üretmek,satır satır karşılaştırma yapmak için kullanılır.

SELECT column1,column2

FROM tablo üst\_tablo

WHERE üst\_tablo\_kolonlar operatör (SELECT alt\_tablo\_kolonlar

FROM tablo alt\_tablo

WHERE alt\_tablo.değer=üst\_tablo.değer)

**GÜN: 4 TARİH: 12.07.2007**

**SAYFA:1**

SQL çalışmaya kaldığım yerden devam ettim ve SQL konu çalışmamı bugün bitirdim.Yine uygulamalar yaptım.Konular özet olarak aşağıdakilerdir:

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

**TABLO BİRLEŞTİRME (JOIN) İŞLEMİ**

Sorgulanan veriler eğer birden fazla tabloda yer alıyorsa JOIN işlemi yapılır.JOIN işlemi yapılabilmesi için iki tabloda da ortak alanlar olmalıdır.

JOIN İŞLEMİNİN YAZILIŞI:

* Ortak alanlar eşitlenir.
* Tekrar eden kolonlar önünde tablo isimleri konulmalıdır.

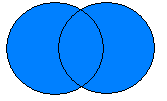
SELECT sütun1,sütun2,…

FROM tablo1,tablo2

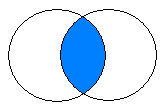
WHERE tablo1.değer=tablo2.değer

**SET OPERATÖRLERİ (ORACLE’ a özel ek fonksiyonlar)**

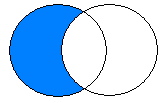
1. **UNION:** Sorguların mükerrer kayıtlar elenerek birleştirilmesidir.Tekrarlanan kayıtlar sadece bir defa alınır.
   * Seçilen kolon sayısı ve tipleri her sorgu ifadesinde aynı olmalıdır.
   * NULL değerler mükerrer kayıt kontrolünde dikkate alınırlar.
   * Kayıtlar SELECT ifadesindeki ilk kolona göre sıralanırlar.Burada ilave bir sıralama işlemi olduğuna dikkat edilmelidir.
2. **UNION ALL:** Sorguların mükerrer kayıtlar dikkate alınmaksızın direkt olarak birleştirilmesidir.Burada tekrarlanan kayıtlar da alınır.



1. **INTERSECT:** Tüm sorgularda ortak olan kayıtların seçilmesini sağlar.



1. **MINUS (FARK):** İlk sorguda olup ikinci sorguda olmayan kayıtları netice olarak döndürür.



**VERİ DEĞİŞTİRME İŞLEMLERİ (DML)**

Veri tabanına veri eklemek,mevcut veri üzerinde değişiklik yapmak ve mevcut veriyi silmek için kullanılan SQL komutlarıdır.

**GÜN: 4 TARİH: 12.07.2007**

**SAYFA:2**

**INSERT**

Tabloya yeni kayıt ve kayıtlar eklemek için kullanılır.

Tabloya tek kayıt ekleme:

INSERT INTO (tablo\_ismi)

(sütun\_listesi)

VALUES (değer\_listesi)

**UPDATE:** Tablodaki kayıtların değiştirilmesini sağlar.

UPDATE (tablo\_ismi)

SET (sütun\_listesi)=(değer\_listesi)

WHERE (sorgu\_şartları)

**!** WHERE şartına hiç birşey yazmazsak tablodaki tüm kayıtlar değiştirilecektir.

**DELETE:** Tablodaki kayıtları silmemizi sağlar.

DELETE FROM (tablo\_ismi)

WHERE (sorgu\_şartları)

**!** WHERE şartına hiç birşey yazmazsak tablodaki tüm kayıtlar silinecektir.

**TRANSACTION YÖNTEMİ**

Veri tabanı üzerinde gerçekleştirilen kalıcı birim iş şeklinde tanımlanabilir.Data tutarlılığı bu sayede sağlanır.Birlikte başarı ve başarısızlık kavramı üzerine kuruludur.Yani işlemlerden hepsi başarılı olursa veri tabanına kalıcı olarak saklanırlar.Aksi halde tüm değişiklikler geri alınır.Bir transaction şu parçalardan oluşur:DML, DDL,DCL.

Transaction bir SQL DML komutu ile başlar.Aşağıdakilerden birinin oluşması ile son bulur.

* COMMIT
* ROLLBACK
* DDL ve DCL komutları
* Sistem çökmeleri
* Kullanıcının ortamdan çıkması

DDL ve DCL ifadeleri otomatik commit’ tir.Yapılan değişiklikleri hemen kaydedilir.

COMMIT: O an ki transaction ı değişiklikleri kaydederek sonlandırır.

ROLLBACK: O an ki transaction ı değişiklikleri geri alarak sonlandırır.

**VERİ TABANI ÜZERİNDE YAPISAL İŞLEMLER (DDL)**

Database objelerini oluşturmak,değiştirmek veya silmek amacıyla kullanılan SQL komutlarıdır.

* CREATE : Oluşturmak
* DROP : Silmek
* ALTER : Değiştirmek

DDL komutları:Oto-commit’ tir.Yani çalıştırıldıkları anda işlemler veri tabanında kalıcı olarak yapılır ve geri alınamazlar.

**CREATE TABLE**

CREATE TABLE tablo\_ismi

[(kolon\_ismi DATATYPE,

kolon\_ismi DATATYPE,…)]

AS alt\_sorgu

**VAROLAN BİR TABLONUN YAPISININ DEĞİŞTİRİLMESİ**

Tablo oluşturulduktan sonra yapısının değiştirilmesi gerekebilir.Örneğin, yeni kolon ekleme, varolan kolonu değiştirme, kolonun veri tipini ve varsayılan değerini değiştirme, kolon silme.

**ALTER TABLE:** Varolan bir tablo üzerinde yani kolon ekleme, varolan bir kolonun özelliklerinin değiştirilmesi vb işlemlerin yapılmasını sağlayan komuttur.

ALTER TABLE tablo\_ismi

ADD [(kolon\_ismi DATATYPE,kolon\_ismi DATATYPE,…)]

**DROP TABLE (TABLO SİLME):** Varolan bir tablonun kalıcı olarak silinmesi**.**Tablonun yapısı ve tüm verileri silinir.Tüm index ler silinir.

DROP TABLE tablo\_ismi

**GÜN: 4 TARİH: 12.07.2007**

**SAYFA:3**

**KOLONLARIN KULLANIM DIŞI YAPILMASI (ORACLE’ a özel bir işlem)**

ALTER TABLE tablo\_ismi

SET UNUSED [(column)]

* Bu kolonlar gerçekten silinmez.
* Sorguda gözükmezler.
* Aynı isimli başka bir kolon eklenebilir.
* Kontrollü bir şekilde kolon silmek için kullanılır.

ALTER TABLE tablo\_ismi

DROP SET UNUSED [(column)]

**OBJELERDE İSİM DEĞİŞİKLİĞİ**

Belli durumlarda kullanılan objelerin isimlerinin değiştirilmesi gerekmektedir.Tablo,view vb objelerin isimlerinin değiştirilmesi için:

* RENAME eski\_tablo\_ismi TO yeni\_tablo\_ismi

**TRUNCATE TABLE**

Varolan bir tablonun verilerinin kalıcı olarak silinmesi

* TRUNCATE TABLE tablo\_ismi

**NOT:** Dikkat edilirse aynı işlemi DELETE komutu ile yapmak ta mümkündür.Ancak DELETE komutu bir DML komutudur ve silinen veriler COMMIT edilmedikçe kaydolmaz.Bunu sağlamak için de DML komutlarının oluşturduğu değişimler veri tabanında ayrı bir yapı içerisinde saklanır.Bu işlem yavaşlamaya sebep olur.

TRUNCATE komutu ise bir DDL komutudur.Veriyi direkt olarak siler ve işlem hızı çok yüksektir.

**VIEWS:** View lar tabloların mantıksal alt veri setlerinden veya tabloların birleşmesi sonucu oluşturulan veri setlerinden oluşan mantıksal tablolardır.Kısaca sorgulardan oluşan mantıksal tablo tanımlarıdır.View lar veri içermezler.İçerdikleri verinin yapısını belirleyen SELECT ifadeleri olarak saklanırlar.View lar karmaşık sorguları basitleştirmek, veri erişimini kısıtlamak, programlamayı kolaylaştırmak, aynı verinin farklı görünümlerini oluşturmak amacıyla kullanılırlar.

CREATE VIEW view\_ismi

AS alt\_sorgu bu şekilde oluşturulur.

**ÇOK KULLANICILI ORTAMDA GÜVENLİK**

(KULLANICI ERİŞİM KONTROLLERİ)

Çok kullanıcılı ortamlarda veri tabanına erişimin güvenli olarak yapılması çok önemlidir.

Güvenlik temelde 3 seviyedir:

* SİSTEM: Sistem yani veri tabanı ve kullanıcı seviyesinde yapısal işlemler için verilen haklar.
* DATA: Objelere verilen haklar.
* PROGRAM: Projedeki gereksinimlere göre veri üzerinde tanımlı kurallar.

**HAKLAR:**

* **Sistem Hakları:** Veri tabanına erişim.
* **Object Hakları:** Veri tabanındaki obje yapılarına erişilmesi

**SİSTEM HAKLARI**

* Yeni kullanıcı oluşturma

CREATE USER user\_ismi

IDENTIFIED BY password

* Kullanıcı silme
* Tablo oluşturma
* Tablo silme
* Tablo yedekleme

**KULLANICIYA HAK VERME (GRANT)**

Bir kullanıcıya bir işlem için sistemde tanımlı haklardan verilmesi

GRANT sys\_hak\_ismi TO USER [user\_ismi/role\_ismi]

**GÜN: 4 TARİH: 12.07.2007**

**SAYFA:4**

**HAK GRUPLARI (ROLE)**

Role isimlendirilmiş bir sistem hakları grubudur.Bu sayede kullanıcılara hak verilmesi ve geri alınması çok kolaylaşır.Haklar tek tek kullanıcılara verilmek yerine bir grup halinde verilir ve alınır.

**ROLE OLUŞTURULMASI**

* Önce role oluşturulur.

CREATE ROLE role\_name

* Role haklar atanır.

GRANT hak TO role\_name

* ROLE kullanılarak kullanıcılara haklar verilir.

GRANT role\_name TO user\_name

**OBJE HAKLARI**

* Tablo,view gibi objeler üzerinde yapılabilecek SELECT,ALTER gibi haklardır.
* Objeden objeye değişebilir.
* Objenin sahibi oluşturan tüm haklara sahiptir.
* Objenin sahibi diğer kullanıcılara hak verme hakkı verebilir.(WITH GRANT OPTION)
* PUBLIC ile tüm kullanıcılara aynı anda hak verilebilir.
* Yazılışı:

GRANT hak ON obje TO (user/role)

[WITH GRANT OPTION] [PUBLIC]

* **HAK SİLME (REVOKE)**

REVOKE hak ON obje FROM (user/role)

**HİYERARŞİK VERİ (ORACLE VERİ TABANINA ÖZEL)**

Birbiri ile ana/detay , alt/üst hiyerarşisi olan verinin veritabanından sorgulanması.

Hiyerarşik veri sorgulama:

SELECT [LEVEL],column

FROM tablo\_ismi

[WHERE şartlar]

[START WITH (başlama şartları)]

[CONNECT BY PRIOR (bağlantı şartları)]

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

Çalıştığım bu konuları da uygulayarak SQL de konu çalışmalarımı bitirdim.Yaptığım uygulamalar:

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

UYEE isimli tabloyu kod ile aşağıdaki gibi oluştururdum:

CREATE TABLE UYEE(

UYE NUMBER(10),

AD CHAR(15),

SOYAD CHAR(15),

DOGUM\_TARIHI DATE,

MESLEK CHAR(15),

ANNE\_ADI CHAR(15) );

Ayrıca tabloya bilgileri aşağıdaki gibi ekleyebilirdim:

INSERT INTO UYEE(UYE,AD,SOYAD,DOGUM\_TARIHI,MESLEK,ANNE\_ADI)

VALUES('1','EMEL','OZDEMIR','01.08.1988','MUHENDIS','EMINE');

INSERT INTO UYEE(UYE,AD,SOYAD,DOGUM\_TARIHI,MESLEK,ANNE\_ADI)

VALUES('2','ZEYNEP','YESIL','25.09.1987','MUHENDIS','AYSE');

INSERT INTO UYEE(UYE,AD,SOYAD,DOGUM\_TARIHI,MESLEK,ANNE\_ADI)

VALUES('3','MEHMET','GEZER','10.01.1983','DOKTOR','EMINE');

**GÜN: 4 TARİH: 12.07.2007**

**SAYFA:5**

INSERT INTO UYEE(UYE,AD,SOYAD,DOGUM\_TARIHI,MESLEK,ANNE\_ADI)

VALUES('4','ELIF','AKIN','13.12.1986','MIMAR','HATICE');

INSERT INTO UYEE(UYE,AD,SOYAD,DOGUM\_TARIHI,MESLEK,ANNE\_ADI)

VALUES('5','SINAN','DEMIR','07.04.1986','MIMAR','AYSEGUL');

**---------------------------------------------------------------------------**

UPDATE uyee SET uye=1312751

WHERE uye=2; */\*Uye numarası 2 olanın numarasını 1312751 olarak değiştirir\*/*

DELETE FROM uyee

WHERE ad='SINAN'; */\* Adı SINAN olan kişiyi siler \*/*

INSERT INTO UYEE(UYE,AD,SOYAD,DOGUM\_TARIHI,MESLEK,ANNE\_ADI)

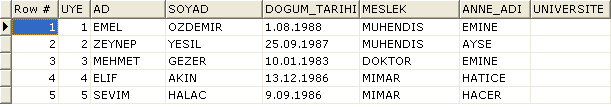
VALUES('5','SEVIM','HALAC','09.09.1986','MIMAR','HACER');

*/\* Yeni bir kişi ekler \*/*

ALTER TABLE UYEE

ADD (UNIVERSITE VARCHAR2 (30)); */\* UYEE tablosuna UNIVERSITE adında yeni kolon ekler \*/*

UNIVERSITE kolonu eklendikten sonra tablonun görünümü aşağıdaki gibidir :



UPDATE UYEE SET UNIVERSITE='TRAKYA'

WHERE UYE=1; */\** *UYE=1 olanın üniversitesini TRAKYA olarak değiştirir* *\*/*

UPDATE UYEE SET UNIVERSITE='ISTANBUL'

WHERE UYE=2; */\** *UYE=2 olanın üniversitesini ISTANBUL olarak değiştirir* *\*/*

UPDATE UYEE SET UNIVERSITE='HACETTEPE'

WHERE UYE=3; */\** *UYE=3 olanın üniversitesini HACETTEPE olarak değiştirir \*/*

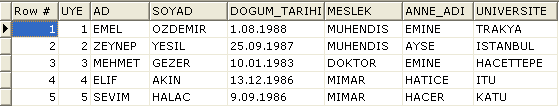
UPDATE UYEE SET UNIVERSITE='ITU'

WHERE UYE=4; */\** *UYE=4 olanın üniversitesini ITU olarak değiştirir* *\*/*

UPDATE UYEE SET UNIVERSITE='KATU'

WHERE UYE=5; /\* *UYE=5 olanın üniversitesini KATU olarak değiştirir \*/*

Yukarıdaki kodlar çalıştırıldıktan sonra tablonun görünümü aşağıdaki gibi olur:



TRUNCATE TABLE UYEE; /\* *UYEE tablosundaki verileri kalıcı olarak siler \*/*

**GÜN: 5 TARİH: 13.07.2007**

**SAYFA:1**

DEPARTMAN ve PERSONEL isimli 2 tablo oluşturdum ve tablolara bilgiler ekledim.Bu tablolarla ilgili biraz uygulama yaptım.Daha sonra bu iki tablo ile ilgili müdürün hazırladığı 38 tane soruyu cevaplamaya başladım.

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

CREATE TABLE DEPARTMAN(

DEPARTMAN\_ADI VARCHAR2 (20),

SEHIR VARCHAR2 (30),

DEPARTMAN\_NO NUMBER (10) ); */\*DEPARTMAN tablosunun oluşturulması \*/*

CREATE TABLE PERSONEL(

ID NUMBER (5),

ADI VARCHAR2 (50),

SOYADI VARCHAR2(20),

DEPARTMAN\_NO NUMBER (3),

MESLEK CHAR(15),

UNVAN CHAR(15),

MAAS NUMBER (15),

PRIM NUMBER (15),

ISE\_GIRIS\_TARIHI DATE,

ISTEN\_CIKIS\_TARIHI DATE ); */\*PERSONEL tablosunun oluşturulması \*/*

DEPARTMAN tablosuna aşağıdaki bilgileri ekledim:

INSERT INTO DEPARTMAN (DEPARTMAN\_ADI,SEHIR,DEPARTMAN\_NO)

VALUES ('MUHASEBE','ISTANBUL','1');

INSERT INTO DEPARTMAN (DEPARTMAN\_ADI,SEHIR,DEPARTMAN\_NO)

VALUES ('PLAN\_PROJE','ISTANBUL','2');

INSERT INTO DEPARTMAN (DEPARTMAN\_ADI,SEHIR,DEPARTMAN\_NO)

VALUES ('BILGI\_ISLEM','ISTANBUL','3');

INSERT INTO DEPARTMAN (DEPARTMAN\_ADI,SEHIR,DEPARTMAN\_NO)

VALUES ('EGITIM','IZMIR','4');

DEPARTMAN tablosu aşağıdaki gibidir:



PERSONEL tablosuna aşağıdaki bilgileri ekledim:

INSERT INTO PERSONEL (ID,ADI,SOYADI,DEPARTMAN\_NO,MESLEK,UNVAN,

MAAS,PRIM,ISE\_GIRIS\_TARIHI,ISTEN\_CIKIS\_TARIHI)

VALUES ('12345','HILAL','KARA','2','MIMAR','AMIR',

'10000','10','04.03.2005','16.07.2007');

INSERT INTO PERSONEL (ID,ADI,SOYADI,DEPARTMAN\_NO,MESLEK,UNVAN,

MAAS,PRIM,ISE\_GIRIS\_TARIHI,ISTEN\_CIKIS\_TARIHI)

VALUES ('12340','EMEL','OZDEMIR','3','MUHENDIS','MUDUR\_YARD',

'8000','20','22.03.2003','05.04.2007');

**GÜN: 5 TARİH: 13.07.2007**

**SAYFA:2**

INSERT INTO PERSONEL (ID,ADI,SOYADI,DEPARTMAN\_NO,MESLEK,UNVAN,

MAAS,PRIM,ISE\_GIRIS\_TARIHI,ISTEN\_CIKIS\_TARIHI)

VALUES ('12311','ZEYNEP','BILEN','3','MUHENDIS','AMIR',

'5000','15','20.03.1999','17.07.2006');

INSERT INTO PERSONEL (ID,ADI,SOYADI,DEPARTMAN\_NO,MESLEK,UNVAN,

MAAS,PRIM,ISE\_GIRIS\_TARIHI,ISTEN\_CIKIS\_TARIHI)

VALUES ('12319','AYSE','GUZEL','1','MUHASEBECI','MUHASEBECI',

'7000','20','08.03.1995','25.10.2004');

INSERT INTO PERSONEL (ID,ADI,SOYADI,DEPARTMAN\_NO,MESLEK,UNVAN,

MAAS,PRIM,ISE\_GIRIS\_TARIHI,ISTEN\_CIKIS\_TARIHI)

VALUES ('12330','EMINE','CALIS','4','OGRETMEN','MUDUR',

'4000','20','08.05.1995','27.10.2005');

INSERT INTO PERSONEL (ID,ADI,SOYADI,DEPARTMAN\_NO,MESLEK,UNVAN,

MAAS,PRIM,ISE\_GIRIS\_TARIHI,ISTEN\_CIKIS\_TARIHI)

VALUES ('12325','MEHMET','OZTURK','4','OGRETMEN','MUDUR\_YARD',

'4000','10','10.11.2000','05.08.2007');

INSERT INTO PERSONEL (ID,ADI,SOYADI,DEPARTMAN\_NO,MESLEK,UNVAN,

MAAS,PRIM,ISE\_GIRIS\_TARIHI,ISTEN\_CIKIS\_TARIHI)

VALUES ('12411','ALI','KAYIP','4','OGRETMEN','OGRETMEN',

'3500','10','15.01.1999','27.12.2007');

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

PERSONEL tablosuna da kişileri ekledikten sonra uygulama yaptım.

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

SELECT ADI,SOYADI,MAAS

FROM PERSONEL

WHERE MAAS<10000; */\* Maaşı 10000 den küçük olanları gösterir \*/*

SELECT ADI,SOYADI,MAAS

FROM PERSONEL

ORDER BY MAAS ASC; */\*Maaşlara göre küçükten büyüğe sıralama yapar\*/*

SELECT ADI,SOYADI,MAAS

FROM PERSONEL

ORDER BY MAAS DESC; */\*Maaşlara göre büyükten küçüğe sıralama yapar\*/*

SELECT ADI,SOYADI,ISE\_GIRIS\_TARIHI

FROM personel

ORDER BY ISE\_GIRIS\_TARIHI ASC; */\*işe giriş tarihine göre sıralama yapıyor\*/*

SELECT COUNT(\*)

FROM DEPARTMAN; */\*Kaç tane departman olduğunu gösterir\*/*

SELECT SUM(MAAS)

FROM PERSONEL; */\*Bütün personellere ödenen toplam maaşı gösterir\*/*

SELECT MIN(MAAS) FROM PERSONEL; */\*En düşük maaşı gösterir\*/*

SELECT MAX(MAAS)

FROM PERSONEL; */\*En yüksek maaşı gösterir\*/*

SELECT DEPARTMAN\_NO,DECODE(DEPARTMAN\_NO,'','BOŞ','DOLU')

FROM PERSONEL; */\*Departman no su boş olanlara 'BOŞ' diğerlerine 'DOLU' yazar \*/*

**GÜN:5 TARİH: 13.07.2007**

**SAYFA:3**

SELECT DEPARTMAN\_NO,DECODE(DEPARTMAN\_NO,'','BOŞ','-') EK\_BİLGİ

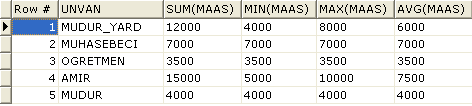
FROM PERSONEL; */\*Departman no su boş olanlara 'BOŞ' diğerlerine '-' yazar\*/*

SELECT UNVAN,SUM(MAAS),MIN(MAAS),MAX(MAAS),AVG(MAAS)

FROM PERSONEL GROUP BY UNVAN; */\* Personelleri unvanlarına göre gruplar ve*

*min,max,toplam ve ortalama maaşlarını bulur\*/*

Bu kod çalıştırılınca sonuç aşağıdaki gibi olur:

**

ALTER TABLE PERSONEL

ADD(AD\_SOYAD VARCHAR2(30)); */\*Personel tablosuna AD\_SOYAD isimli kolon ekler\*/*

UPDATE PERSONEL SET AD\_SOYAD='HILAL KARA'

WHERE ADI='HILAL'; */\* AD=HILAL ise AD\_SOYAD=HILAL KARA yapılır \*/*

UPDATE PERSONEL SET AD\_SOYAD='EMEL OZDEMİR'

WHERE ADI='EMEL'; */\* AD=EMEL ise AD\_SOYAD=EMEL OZDEMIR yapılır \*/*

UPDATE PERSONEL SET AD\_SOYAD='ZEYNEP BILEN'

WHERE ADI='ZEYNEP'; */\* AD=ZEYNEP ise AD\_SOYAD=ZEYNEP BILEN yapılır \*/*

UPDATE PERSONEL SET AD\_SOYAD='AYSE GUZEL'

WHERE ADI='AYSE'; */\* AD=AYSE ise AD\_SOYAD=AYSE GUZEL yapılır \*/*

UPDATE PERSONEL SET AD\_SOYAD='EMINE CALIS'

WHERE ADI='EMINE'; */\* AD=EMINE ise AD\_SOYAD=EMINE CALIS yapılır \*/*

UPDATE PERSONEL SET AD\_SOYAD='MEHMET OZTURK'

WHERE ADI='MEHMET'; */\* AD=MEHMET ise AD\_SOYAD=MEHMET OZTURK yapılır \*/*

UPDATE PERSONEL SET AD\_SOYAD='ALI KAYIP'

WHERE ADI='ALI'; */\* AD=ALI ise AD\_SOYAD=ALI KAYIP yapılır \*/*

Yukarıdaki kodlar çalıştırıldıktan sonra tablonun görünümü aşağıdaki kod ile gösterilirse:

SELECT AD\_SOYAD,MESLEK FROM PERSONEL;

 bu şekilde olur.

**GÜN:5 TARİH: 13.07.2007**

**SAYFA:4**

Daha sonra SQL sorularını cevaplamaya başladım.

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

**SQL SORULARI VE CEVAPLARI**

1) Departman tablosundaki departman adı kolonu ile şehir kolonunu birleştiriniz

**SELECT** DEPARTMAN\_ADI||SEHIR

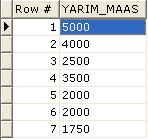
**FROM** DEPARTMAN;

2) Personel maaşlarını 2’ye bölüp yeni kolonu ‘yarım maaş’ olarak adlandırınız

**SELECT** MAAS/2 YARIM\_MAAS

**FROM** PERSONEL;

Bu kod çalıştırılınca sonuç aşağıdaki gibi olur:



3) Departman adının üçüncü satırı ‘A’ olanları getiriniz.

**SELECT** DEPARTMAN\_ADI

**FROM** DEPARTMAN

**WHERE** DEPARTMAN\_ADI **LIKE** '\_\_A%';

Bu kod çalıştırılınca sonuç aşağıdaki gibi olur:

****

4) Departman tablosunda departman numaralarına göre sıralama yapınız.

**SELECT** \*

**FROM** DEPARTMAN

**ORDER BY** DEPARTMAN\_NO;

5) Maaşın 100 fazlası 1200’den büyük olan kişiler içinde adı ‘E’ harfi ile başlayanları

getirin .

**SELECT** ADI,MAAS

**FROM** PERSONEL

**WHERE** MAAS+100>1200

**AND** ADI **LIKE** 'E%';

Bu kod çalıştırılınca sonuç aşağıdaki gibi olur:

****

6) Sistem tarihinde saat bilgisini alınız.

**SELECT TO\_CHAR** (**SYSDATE**, 'DD/MM/YYYY HH:MI:SS')

**FROM DUAL**;

**GÜN:6 TARİH: 16.07.2007**

**SAYFA:1**

Stajda 2.haftaya başladım.İlk hafta kullandığım bilgisayar izinli olan personelindi.Personel geldiği için masa değiştirdim ancak kullanacağım bilgisayar arızalıydı.Bunu bildiğim için kendi laptop ımı getirdim.Programları yüklediler.SQL Navigator ve ORACLE yüklendi.Yükleme işlemi uzun sürdü.Programların yüklenmesini bekledim.Bende PLATINUM ERwin 3.5.2 yi yükledim.

**ORACLE 10g veritabanı kurulması:**

Oracle 10g veritabanı yazılımı 1 CD dir.Bilgisayar en az  512 MB RAM e sahip olmalıdır.İşletim sistemi Windows 2000,2003,XP.Kuruluma başlamadan işletim sitemine "Administrator" olarak girilir.Kurulum yapmadan önce işletim sisteminin dil ayarları İngilizce yapılır.Diğer durumda kurulumda sorun çıkar.Kurulum bittikten sonra ayarlar tekrar Türkçe yapılır.

Programlar kurulduktan sonra SQL sorularını cevaplamaya kaldığım yerden devam ettim.

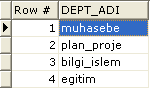
**----------------------------------------------------------------------------------------------**

7) Departman tablosunda departman isimlerini küçük harf olarak gösteriniz

**SELECT LOWER(DEPARTMAN\_ADI) DEPT\_ADI**

**FROM DEPARTMAN**;

Bu kod çalıştırılınca sonuç aşağıdaki gibi olur:

****

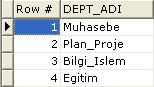
8) Departman tablosunda departman isimlerinin ilk harfini büyük diğerlerini küçük

yazdırınız.

**SELECT INITCAP(DEPARTMAN\_ADI) DEPT\_ADI**

**FROM DEPARTMAN**;

Bu kod çalıştırılınca sonuç aşağıdaki gibi olur:

****

1. Substr fonksiyonu kullanarak 3. karakteri ‘H’ ve ya ‘L’ olan departman isimlerini gösteriniz.

**SELECT DEPARTMAN\_ADI**

**FROM DEPARTMAN**

**WHERE SUBSTR(DEPARTMAN\_ADI,3,1)IN('H','L');**

Bu kod çalıştırılınca sonuç aşağıdaki gibi olur:

****

**GÜN:6 TARİH: 16.07.2007**

**SAYFA:2**

10) Personel tablosunda departman numarası boş olan kişiler için decode kullanarak boş

yazınız.

**SELECT DEPARTMAN\_NO,DECODE(DEPARTMAN\_NO,'','BOŞ')**

**FROM PERSONEL**;

11) 1.1.2003 ile günümüz arasındaki ay ve gün sayısını bulun.

**/\*GÜN SAYISI\*/**

**SELECT TO\_DATE('16/07/2007','DD/MM/YYYY')-**

**TO\_DATE('01/01/2003','DD/MM/YYYY')GÜN\_SAYISI**

**FROM DUAL;**

**/\*AY SAYISI\*/**

**SELECT MONTHS\_BETWEEN(TO\_DATE('16-07-2007','DD/MM/YYYY'),**

**TO\_DATE('01-01-2003','DD/MM/YYYY'))AY\_SAYISI**

**FROM DUAL;**

12) Departman adı ‘BILGI\_ISLEM’ olan personelleri bulun.

**SELECT** PERSONEL.adi,PERSONEL.soyadi

**FROM** DEPARTMAN,PERSONEL

**WHERE** DEPARTMAN.departman\_adi='BILGI\_ISLEM'

**AND** PERSONEL.departman\_no=DEPARTMAN.departman\_no**;**

Bu kod çalıştırılınca sonuç aşağıdaki gibi olur:

****

13) Personellerin amirlerinin çalıştığı departmanları bulunuz.

**SELECT** DEPARTMAN.departman\_no,DEPARTMAN.departman\_adi,PERSONEL.unvan

**FROM** DEPARTMAN,PERSONEL

**WHERE** UNVAN='AMIR'

**AND** DEPARTMAN.departman\_no=PERSONEL.departman\_no**;**

14) Departman numaralarına göre o departmanlarda çalışanlara ödenen toplam maaşı bulunuz

**SELECT** DEPARTMAN\_NO,SUM(MAAS)

**FROM** PERSONEL

**GROUP BY** DEPARTMAN\_NO;

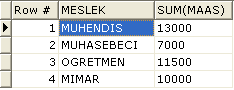
15) Departmanlarda çalışan meslek gruplarına ödenen toplam maaşı hesaplatınız.

**SELECT** MESLEK,SUM(MAAS)

**FROM** PERSONEL

**GROUP BY** MESLEK;

Bu kod çalıştırılınca sonuç aşağıdaki gibi olur:

****

16) 6000’den büyük maaş alanları departman numaralarına göre hesaplatınız.

**SELECT** DEPARTMAN\_NO **FROM** PERSONEL

**WHERE** MAAS>6000

**GROUP BY** DEPARTMAN\_NO;

**GÜN:6 TARİH: 16.07.2007**

**SAYFA:3**

17) Unvana göre max, min, sum ve ort maaş alanları hesaplatınız

**SELECT** UNVAN**,SUM**(MAAS)**,MIN**(MAAS)**,MAX**(MAAS),**AVG**(MAAS)

**FROM** PERSONEL

**GROUP BY** UNVAN;

18) Max ve min maaş arasındaki ücret farkı 2000 den büyük olan departman veya departmanlarda ödenen toplam maaşı bulunuz.

**SELECT** DEPARTMAN\_NO**,SUM**(MAAS)

**FROM** PERSONEL

**GROUP BY** DEPARTMAN\_NO **HAVING MAX**(MAAS)- **MIN**(MAAS)**>**(2000);

19) En çok kişi çalıştıran departman ismi.

**SELECT** DEPARTMAN\_ADI

**FROM** DEPARTMAN

**WHERE** DEPARTMAN\_NO=(**SELECT** DEPARTMAN\_NO

**FROM** PERSONEL

**GROUP BY** DEPARTMAN\_NO

**HAVING COUNT**(\*)=(**SELECT MAX**(**COUNT**(\*))

**FROM** PERSONEL

**GROUP BY** DEPARTMAN\_NO));

Bu kod çalıştırılınca sonuç aşağıdaki gibi olur:



20) 5 kişiden fazla çalışanı olan departmanlar.

**SELECT** DEPARTMAN\_NO**,COUNT**(\*)

**FROM** PERSONEL

**GROUP BY** DEPARTMAN\_NO

**HAVING COUNT**(\*)>5;

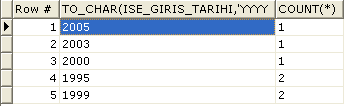
21) Personel tablosunda yıllara göre işe giren personel sayısı.

**SELECT TO\_CHAR**(ISE\_GIRIS\_TARIHI,'YYYY'),**COUNT**(\*)

**FROM** PERSONEL

**GROUP BY TO\_CHAR**(ISE\_GIRIS\_TARIHI,'YYYY');

Bu kod çalıştırılınca sonuç aşağıdaki gibi olur:



22) İsim ve soy ismi birbirinden ayırınız.İsim kısmına ismi, diğer kısmını da soy\_ismi olarak

**SELECT SUBSTR**(AD\_SOYAD,1**,INSTR**(AD\_SOYAD,' '))ISMI,

**SUBSTR**(AD\_SOYAD,**INSTR**(AD\_SOYAD,' '))SOY\_ISMI

**FROM** PERSONEL;

**GÜN:7 TARİH: 17.07.2007**

**SAYFA:1**

SQL sorularını cevaplamaya devam ettim.Soruları bugün bitirdim.Bugün cevapladıklarım aşağıdakilerdir:

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

23) 12311 no’lu çalışanın maaşından çok alanları bulun.

**SELECT** ADI,SOYADI,MAAS

**FROM** PERSONEL

**WHERE** MAAS>(**SELECT SUM**(MAAS)

**FROM** PERSONEL **WHERE** ID='12311');

24) 12330 no’lu çalışan ile aynı departmanda çalışan kişileri bulun

**SELECT** ADI,SOYADI

**FROM** PERSONEL

**WHERE** DEPARTMAN\_NO **IN**(**SELECT** DEPARTMAN\_NO

**FROM** PERSONEL **WHERE** ID=12330);

25) Ortalama maaştan küçük maaş alanları bulun.

**SELECT** ADI,SOYADI,MAAS

**FROM** PERSONEL

**WHERE** MAAS<(**SELECT AVG**(MAAS)

**FROM** PERSONEL);

Bu kod çalıştırılınca sonuç aşağıdaki gibi olur:

****

26) Mimar çalışmayan departmanda çalışan kişileri bulunuz.

**SELECT** ADI

**FROM** PERSONEL

**WHERE** DEPARTMAN\_NO **IN**(**SELECT** DEPARTMAN\_NO

**FROM** PERSONEL

**WHERE** MESLEK!='MIMAR');

27) Unvanı Müdür olanın departmanında çalışan kişileri bulunuz

**SELECT** ADI||' '||SOYADI AD\_SOYAD

**FROM** PERSONEL

**WHERE** DEPARTMAN\_NO **IN**(**SELECT** DEPARTMAN\_NO

**FROM** PERSONEL

**WHERE** UNVAN='MUDUR');

28) 12311 no’lu personeli 3 no’lu departmana değiştiriniz

**UPDATE** PERSONEL

**SET** DEPARTMAN\_NO='3'

**WHERE** ID='12311';

29) Ortalama maaşın altında maaş alanların maaşını % 20 arttırın

**UPDATE** PERSONEL

**SET** MAAS=MAAS+MAAS\*(0.2)

**WHERE** MAAS<(**SELECT AVG**(MAAS)

**FROM** PERSONEL);

**GÜN:7 TARİH: 17.07.2007**

**SAYFA:2**

30) Mimarların sicillerini 1000 arttırıp işçi olarak insert edin.

**INSERT INTO** PERSONEL(ID,UNVAN)

**SELECT** ID+1000,'ISCI'

**FROM** PERSONEL

**WHERE** UNVAN='MIMAR';

31) Ortalama maastan düşük alıp 12330 no’lu personel ile aynı departmanda çalışanları bulunuz

**SELECT** ADI,SOYADI

**FROM** PERSONEL

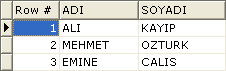
**WHERE** DEPARTMAN\_NO **IN**(**SELECT** DEPARTMAN\_NO

**FROM** PERSONEL

**WHERE** ID='12330'

**AND** MAAS<(**SELECT AVG**(MAAS) **FROM** PERSONEL));

Bu kod çalıştırılınca sonuç aşağıdaki gibi olur:

****

32) Personel tablosuna erişim için personel1 isimli SYNONYM tanımlayınız.

**CREATE PUBLIC SYNONYM** PERSONEL1 **FOR** PERSONEL;

33) Eğitim 9 isimli aynı şifreli bir kullanıcı oluşturunuz.

**CREATE USER** EGITIM9 **IDENTIFIED BY** EGITIM9;

34) Eğitim isimli bir role oluşturup bu role CREATE TABLE, DROP TABLE

haklarını atayınız. Bu rolude eğitim7 kullanıcısına veriniz.

**CREATE ROLE** EGITIM **GRANT CREATE TABLE,**

**DROP TABLE TO** EGITIM **GRANT** EGITIM **TO** EGITIM7;

35) Egitim10 kullanıcısından personel tablosu üzerindeki SELECT

hakkını kaldırınız

**REVOKE** HAK **ON SELECT TO** EGITIM10;

36) Fark Operatörü kullanarak Departman\_No=2’de çalışmayan unvanı 'İŞÇİ' olan kişileri bulunuz.

**SELECT** DEPARTMAN\_NO,ADI,SOYADI,UNVAN **FROM** PERSONEL

**WHERE** UNVAN='ISCI'

**MINUS**

**SELECT** DEPARTMAN\_NO,ADI,SOYADI,UNVAN **FROM** PERSONEL

**WHERE** DEPARTMAN\_NO=2;

37) Kendi departman ortalamasının üzerinde maaş alan kişileri bulunuz.

**SELECT** DEPARTMAN\_NO,ADI,MAAS

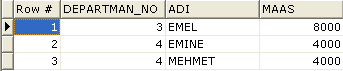
**FROM** PERSONEL PERS

**WHERE** MAAS>(**SELECT AVG**(MAAS)

**FROM** PERSONEL

**WHERE** DEPARTMAN\_NO=PERS.DEPARTMAN\_NO);

Bu kod çalıştırılınca sonuç aşağıdaki gibi olur:

****

**GÜN:7 TARİH: 17.07.2007**

**SAYFA:3**

38) Departmanlar hiyerarşisini gösteriniz.

**START WITH CONNECT BY**

**--------------------------------------------------------------------------------------------**

SQL soruları bittikten sonra veri modellemeye çalıştım.PLATINUM ERwin programını kullanmayı öğrendim ve basit bir veri modeli yaptım.

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

**VERİ MODELLEME**

Veri modelleme,bilginin nasıl saklanacağının mantıksal/fiziksel olarak tanımlanmasıdır.Bir işletmenin hatırlamaya değer bulduğu verilerin şekil ve metin olarak ifade edilmesidir.Gerçek dünya (olay ve işlemler) ile veritabanındaki fiziksel saklama yapısı arasında bir köprü oluşturur.Yazılım geliştirmenin en önemli süreçlerinden biri olan veri modelleme,bilişim ihtiyaçlarının keşfedilmesi ve herkesin anlayabileceği bir şekilde belgelenmesi işlemidir.

**DATA MODEL OLUŞTURMA BASAMAKLARI:** Data objelerinin tanımlanması,İlişkilerinin tanımlanması,Entity ve ilişkilerle birlikte geçici bir ER diyagram çizimi,ER diyagramın iyileştirilmesi,Diyagrama key alanlarının eklenmesi,Key olmayan alanların eklenmesi.

**ER Modelleme:**Bir analiz aşamasıdır.Kullanıcıdan veri ihtiyacı alınarak gerçek dünyadaki varlıkları ve onlar arasındaki ilişkilerin modeli oluşturulur.Bu model tasarlanırken ilişkisel modelin getireceği kısıtlamalar düşünülmez.Görsel tasarım araçları gibi kullanılır.

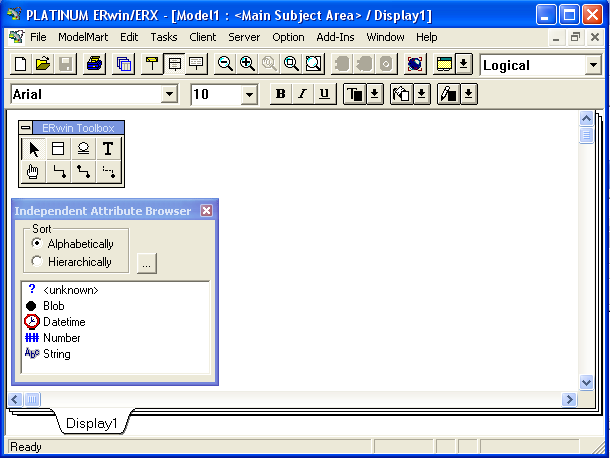
**ER Diyagram Objeleri:**

**Entity:** Datanın saklandığı temel objelerdir.Genellikle tanınabilir kavramlardır.Bir entity ilişkisel modelde genellikle bir tabloya karşı düşer.

**İlişkiler:** Bir ya da daha fazla entity arasındaki ilişkinin mahiyetini modeller.

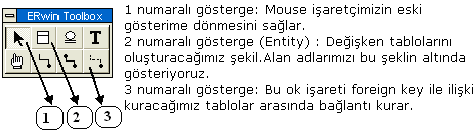
**Nitelikler:** Nitelikler bağlı bulundukları entity’yi tanımlayan veya belirleyen özelliklerdir.

Veri modeli oluşturmak için aşağıdaki ERwin programını kullandım:



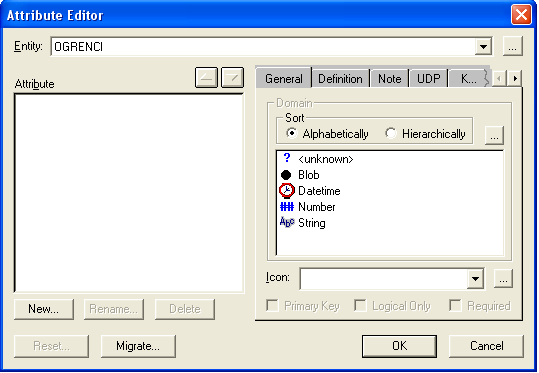
**GÜN:7 TARİH: 17.07.2007**

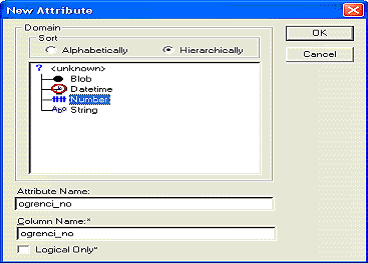
**SAYFA:4**

**ERwin Toolbox:** Veri modelimizin şeklini oluşturmak,entityler arasındaki ilişkileri göstermek için kullanırız.

Örnek olarak 2 entity oluşturalım.Birinin adı OGRENCI,diğerinin DERSLER olsun.

OGRENCI isimli entity’ ye çift tıkladığımız zaman verilerimizi girmemiz için aşağıdaki şekil gelir:





New butonunu tıkladığımızda

yandaki gibi gelen şekilde

verimizin adını Name

kısımlarına yazarız:

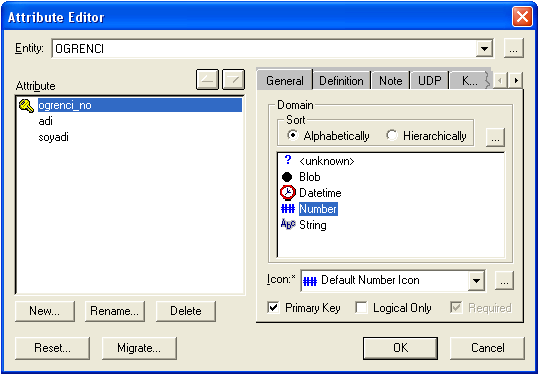
**GÜN:8 TARİH: 18.07.2007**

**SAYFA:1**

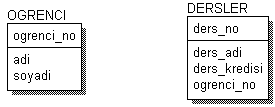
Veri modelini oluşturmada dün kaldığım yerden devam ettim.Bugün proje ile ilgili bilgilere baktım.Daha sonra veri modelimi oluşturdum.

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

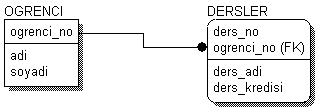
New Attribute de Name kısmına ogrenci\_no yazdıktan sonra Number ı da seçeriz ve OK butonunu tıklarız.Bu şekilde adi (string) ve soyadi (string) bilgilerini de ekledikten sonra aşağıdaki gibi ogrenci\_no yu Primary Key olarak seçeriz.



Daha sonra OK deriz ve aynı şekilde DERSLER entity sine de verileri ekleriz.DERSLER entity sinde ders\_no primary key yapılır.Eklemeler yapıldıktan sonra entity ler aşağıdaki gibi olur:



Öğrencilerin aldıkları dersler olacağı için 2 entity bağlantılıdır.Bağlantıyı yaptıktan sonra görünümü aşağıdaki gibi olur:



Veri modeli bu şekilde oluşturulduktan sonra kaydedilir ve programdan çıkılır.