**T.C.**

**BİLECİK ŞEYH EDEBALİ ÜNİVERSİTESİ**

**MÜHENDİSLİK FAKÜLTESİ**

**BİLGİSAYAR MÜHENDİSLİĞİ**

**VERİ TABANI YÖNETİM SİSTEMLERİ**

****

**Adı Soyadı : Sinan Yazıcı**

**Numarası : 37813301512**

**Bölümü : Bilgisayar Müh.**

**Sınıf : 2**

**Ödev : 1**

1.12. ÇALIŞMA SORULARI

* 1.Soru :Geleneksel dosyalama sistemlerinin çalışma şeklini açıklayarak bildiğiniz bir programlama dili yardımıyla txt dosyadan veri okumak ve veri yazmak için bir uygulama geliştiriniz
* 1.Cevap :

*Bir dosyaya veri yazma*

/\* 16prg01.c:

10 öğrenciye ait bilgileri 'ogrenci.txt' dosyasına kaydeder. \*/

#include <stdio.h>

#include <stdlib.h>

int main()

{

FILE \*dg; /\* dosya göstericisi \*/

const int n = 10; /\* öğrenci sayısı \*/

char ad[10];

int no, Not, i=0;

dg = fopen("ogrenci.txt", "w");

if( dg == NULL )

puts("ogrenci.txt dosyasi acilmadi. !\n"), exit(1);

puts("10 ogrenciye ait bilgileri girin:");

while( i++<n )

{

printf("%d. ogrencinin numarasi: ",i);

scanf("%d",&no);

printf("%d. ogrencinin adi : ",i);

scanf("%s",ad);

printf("%d. ogrencinin notu : ",i);

scanf("%d",&Not);

printf("\n");

fprintf(dg,"%5d %10s %3d\n",no,ad,Not); /\* verileri formatlı yaz! \*/

}

/\* dosyayı kapat \*/

fclose(dg);

puts("Bilgiler kaydedildi.\a");

return 0;

}

*Bir dosyadan veri okuma*

/\* 16prg02.c:

ogrenci.txt dosyasından no, isim ve not bilgileri okur ve

notlar arasından en büyüğü, en küçüğü ve sınıf ortlamasını (0'lar hariç) hesaplar. \*/

#include <stdio.h>

#include <stdlib.h>

int main()

{

FILE \*dg; /\* dosya işaretçisi \*/

char Ad[10];

int Not, No, eb, ek, n, top;

float ort;

if( (dg=fopen("ogrenci.txt","r")) == NULL )

puts("Dosya açılmadı !\n"), exit(1);

/\* başlangıç değerleri ata \*/

ek = 1000; /\* çok büyük \*/

eb = -1000; /\* çok küçük \*/

top = 0; /\* notların toplamı \*/

n = 0; /\* notu 0'dan farlı öğrencilerin toplamı \*/

while( !feof(dg) ) /\* dosyanın sonuna kadar \*/

{

fscanf(dg,"%d %s %d",&No,Ad,&Not); /\* verileri oku! \*/

if(Not>eb) eb = Not; /\* en büyük ve en küçük bulunuyor... \*/

if(Not<ek) ek = Not;

if(Not) n++; /\* Not 0'dan farklı mı? \*/

top += Not; /\* Notlarin toplamı \*/

}

fclose(dg); /\* dosyayı kapat \*/

ort = (float) top/n; /\* ortalama (0 lar hariç!) \*/

printf("En yuksek not = %2d\n",eb); /\* sonuçlar ekrana ... \*/

printf("En dusuk not = %2d\n",ek);

printf("Ortalama = %4.1f\n",ort);

return 0;

}

* 2.Soru :Geleneksel dosyalama sistemleriyle veritabanı yönetim sistemlerinin benzerlik ve farklılıklarını açıklayınız.
* 2.Cevap :

Geleneksel dosyalama sistemleri veritabanı yönetim sistemleri öncesinde veri depolamak için kullanılan sistemlerdir.

Zamanla geleneksel dosyalama sistemleri artan veri kapasitesi, veri işleme hızı ve kullanım alanı gibi birçok ihtiyaca cevap veremez duruma gelmiştir. Dolayısıyla artık yerini veritabanı yönetim sistemleri yazılımlarına bırakmıştır.

Geleneksel dosyalama sistemine göre verilere erişim hızlı.

Gereksiz veri tekrarını engeller.

Her ihtiyaca, büyük veya küçük bütün otomasyon sistemleri günümüz  artık veri tabanı kullanmaktadır.

* 3.Soru : Vtys'nin geleneksel sisteme göre üstün özelliklerini açıklayınız.
* 3.Cevap :
* **Veri Tekrarı (“Data Redundancy”) :**Aynı verinin sürekli tekrarlanmasını önler.Aynı tablo içerisinde, farklı bilgisayarlarda vb..; Veri Tekrarı (“Data Redundancy”) azaltılır ya da yok edilir.
* **Veri Tutarlılığı (“Data Consistency”):**Aynı verinin değişik yerlerde birkaç kopyasının bulunması “bakım” zorluğu getirir: bir yerde güncellenen bir adres bilgisi başka yerde güncellenmeden kalabilir ve bu durum veri tutarsızlığına (“Data Inconsistency”) yol açar.
* **Veri Paylaşımı / Eşzamanlılık (“Concurrency”):**Veri tabanı yönetim sistemi (VTYS) kullanılmadığı durumlarda veriye sıralı erişim yapılır. Yanı birden çok kullanıcı aynı anda aynı veriye erişemez. Bir VTYS’de ise aynı veritabanlarına saniyede yüzlerce, binlerce erişim yapılabilir.
* **Veri Bütünlüğü (“Data Integrity”):**Bir tablodan bir öğrenci kaydı silinirse, öğrenci var olduğu diğer tüm tablolardan silinmelidir.
* **Veri Güvenliği (“Data Security”):**Verinin isteyerek ya da yanlış kullanım sonucu bozulmasını önlemek için çok sıkı mekanizmalar mevcuttur. Veri tabanına girmek için kullanıcı adı ve şifreyle korumanın yanı sıra kişiler sadece kendilerini ilgilendiren tabloları ya da tablo içinde belirli kolonları görebilirler.
* **Veri Bağımsızlığı (“Data Independence”):**Programcı, kullandığı verilerin yapısı ve organizasyonu ile ilgilenmek durumunda değildir.
* 4.Soru : Veritabanının görevini açıklayarak kullanıldığı alanlara örnekler veriniz.
* 4.Cevap : En basit anlamda veritabanı, belirli bir amaca yönelik düzen verilmiş kayıt ve dosyaların tümüdür.

Veritabanı insanlar tarafından sadece bilgisayarlarda kullanıldığı düşünülmektedir. Fakat bu büyük bir yanılgıdır. Hayatımızın büyük bir alanında veritabanı kullanılmaktadır. Ancak en sık kullanıldığı yerler bilgisayarlardır. Veritabanı yazılımını diğer yazılımlardan ayıran en önemli özellik ise, aranan bir bilgiye hızlıca erişimi sağlamasıdır.

Veritabanı yazılımları, özellikle hastanelerde sıkça kullanılır. Çünkü hastenelerde gelen hasta kayıtlarının veritabanı yazılımı olmadan tutulması neredeyse imkansızdır. Ayrıca, birçok dev firma, stok durumu, üretim durumu gibi endüstriyel verilerini, veritabanı yazılımları ile kontrol ederler. Bu örnekler gibi birçok iş alanında veritabanı yazılımları kullanılır.

* Havayolları: Rezervasyonlar ve Bilet Satışı  
  Bankalar: Bankacılık işlem ve hareketleri  
  Üniversiteler: Öğrenci Takip sistemi ve Not girişleri  
  Satış: Müşteri ve Ürün bilgileri  
  Çevrimiçi Perakendeciliği: Sipariş Kayıtları ve sonuçları  
  İnsan Kaynakları: Çalışan Profilleri ve Maaşları  
  Üretim: Stoklar ve üretilen ürünlerin takibi
* 5.Soru :Tablo,satır ve sütun kavramlarını açıklayınız.
* 5.Cevap :

##### Tablo (table), saklanan verilerin düzenli olarak satırlar (rows), sütunlar (fields ) olarak düzenlenmesi düzenlenmesi ile olu şan yapılardır. Her satır, tabloda bulunan kayıtları (records) göstermektedir. Her sütun, tabloda bulunan her alanda saklanacak verilerin biçimini belirtir. Bir veritabanı tasarımına başlamadan önce veritabanında hangi tabloların olması gerektiği ve her bir tabloda bulunması gereken sütunların düşünülmesi gerekmektedir. Tablolar Tabloların oluşturulması her tabloda tabloda saklanacak saklanacak verilerin verilerin belirlenmesi belirlenmesi ve varsa tablolar arasındaki ilişkilerin düzenlenmesi “ İlişkili Veritabanı Modelinin” önemli adımlarıdır.

##### 6.Soru :Birincil anahtar ve yabancıl anahtar kavramlarını ve farklılıklarını açıklayınız.

* 6.Cevap :

**Birincil Anahtar (Primary Key)**

Birincil anahtar amacı benzersiz bir tabloda her bir kayıtı ayrı ayrı belirlemektir. Genel olarak, birincil anahtar tek bir sütun üzerinde tanımlıdır ve iki sütun üzerinde tanımlı olması mümkün değildir.

Üç temel kural birincil anahtarları tanımlayan özelliktir:

\*Her kayıtın birincil anahtarında bir değer olması gerekir. O boş olamaz.  
\*Birincil anahtar değerleri benzersiz olmalıdır.  
\*Birincil anahtar değerleri yenilenen olmamalıdır. Eğer yenileme olursa bir kayıt silindiğinde bu hata ve karışıklık yaratabilir, bu yüzden birincil anahtar başka bir kayıt yeniden atanınca yinelenen olmamalıdır.

**Yabancı (İkincil) Anahtar (Forein Key)**

Bir yabancı anahtar bir alanı (veya hedef olarak başvurulan bir tablodan) başka bir tabloda belirli bir anahtar yerine birincil anahtarı gösteren kolonlardır. Bu ilgili tabloları mantıklı bilgilerle ilişkilendiren bir yoludur.

Örneğin, ürün tablosunda  başvuran birincil anahtar alanı üreticiler tablosunda bir yabancı anahtar içerir. Bu şekilde, her ürün – dış kilit noktaları ilişkili bir üreticisi var olur. Unutmayın ki yabancı anahtar genellikle benzersiz değidir ancak başvurulan alanı başvurulan tabloda (birincil anahtar) gösterir.

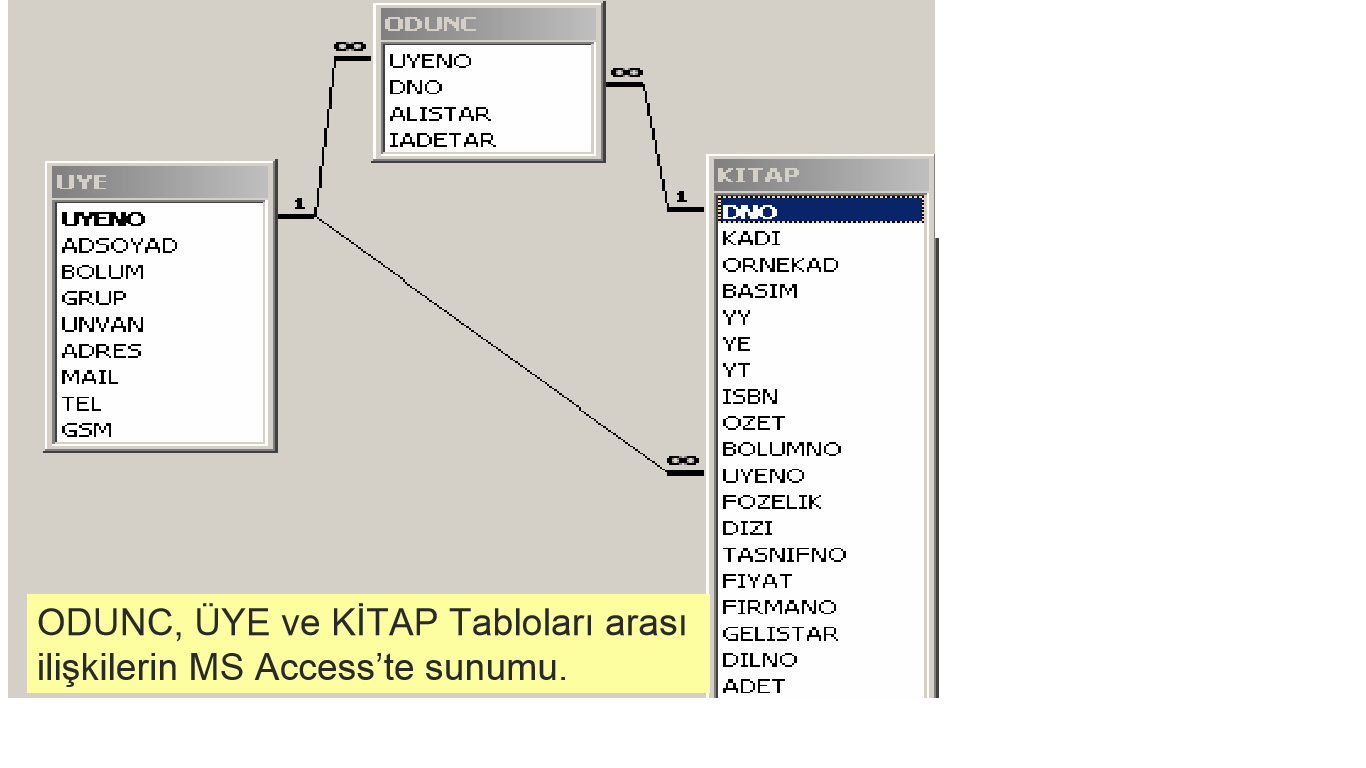
* 7.Soru :Veritabanı kullanıcı türleri nelerdir?
* 7.Cevap :

Bir veritabanının kullanıcılarını belirtirken, onları 3 genel sınıf halinde tanımlayabiliriz:

* Uygulama programcısı : COBOL, C++ gibi bazı yüksek seviyeli dillerde programlar yazmakla sorumludur.
* Son-kullanıcı : Sorgu dili ile veritabanına erişen kişidir.
* Veritabanı yöneticisi (DBA) : Veritabanı üzerindeki tüm işlemleri kontrol eder.

8.Soru :Örnek bir veritabanı için kullanıcılar belirleyerek yetkilendirmelerini şematik olarak gösteriniz.

* 8.Cevap :



* 9.Soru :Veri tabanı ile VTYS'nin farkını açıklayınız.
* 9.Cevap : Veritabanı ile VTYS’ nin farkları nelerdir?

**Veri tabanları** birbirleriyle ilişkili bilgilerin depolandığı alanlardır. Bilgi artışıyla birlikte bilgisayarda bilgi depolama ve bilgiye erişim konularında yeni yöntemlere ihtiyaç duyulmuştur. Veri tabanları; büyük miktardaki bilgileri depolamada geleneksel yöntem olan ‘’dosya-işlem sistemine’’ alternatif olarak geliştirilmiştir.

**Veri tabanı yönetim sistemi** (VTYS, İngilizce: Database Management System, kısaca DBMS), veri tabanlarını tanımlamak, yaratmak, kullanmak, değiştirmek ve veri tabanı sistemleri ile ilgili her türlü işletimsel gereksinimleri karşılamak için tasarlanmış sistem ve yazılımdır.

10.Soru :İlişkisel veri modelini açıklayınız.

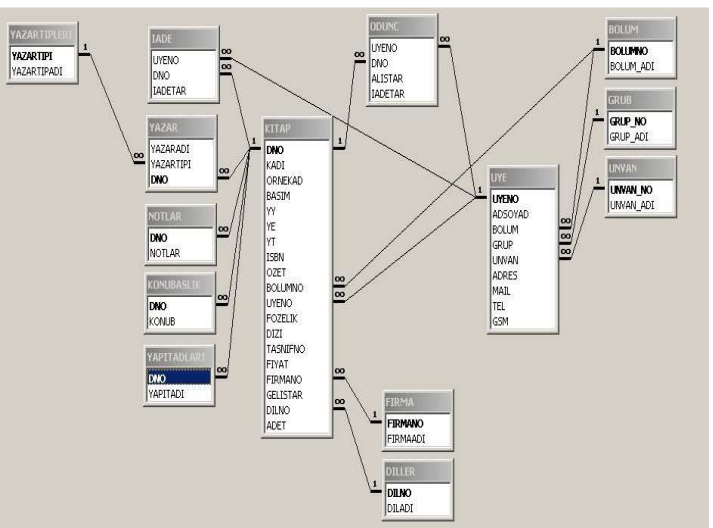
* 10.Cevap : Veritabanında asıl önemli kavram, kayıt yığını ya da bilgi parçalarının tanımlanmasıdır. Bu tanıma Şema adı verilir. Şema veri tabanında kullanılacak bilgi tanımlarının nasıl modelleneceğini gösterir. Buna Veri Modeli (*Data Model*) yapılan işleme de [Veri modelleme](http://tr.wikipedia.org/wiki/Veri_modelleme) denir. En yaygın olanı, [İlişkisel Model](http://tr.wikipedia.org/w/index.php?title=%C4%B0li%C5%9Fkisel_Model&action=edit&redlink=1)' dir.(*relational model*), bu modelde veriler tablolarda saklanır. Tablolarda bulunan satırlar (*row*) kayıtların kendisini, sütunlar (*column*) ise bu kayıtları oluşturan bilgi parçalarının ne türden olduklarını belirtir. Başka modeller (Sistem Modeli ya da Ağ Modeli gibi) daha belirgin ilişkiler kurarlar.
* 11.Soru :Bir öğrenci bilgi sistemi kullanılacak veritabanı, tablolar ve tablolar arasındaki ilişkiyi belirleyiniz.

**3. Bölüm Özeti**

**Örnek Kütüphane Veritabanı;**

Veritabanı Şeması;

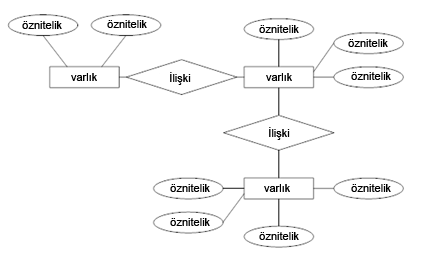
YAZARTIPLERI(YAZARTIPI, YAZARTIPADI), YAZAR(YAZARNO, YAZARADI, YAZARTIPI, DNO),   
NOTLAR(NOTNO, NOTLAR, DNO),  
KONUBASLIK(KONUBNO, KONUB, DNO),   
YAPITADI(YAPITNO, YAPITADI, DNO),  
FIRMA(FIRMANO, FIRMAADI),  
DILLER(DILNO, DILADI),   
KITAP(DNO, KADI, ORNEKAD, BASIM, YY, YE, YT, ISBN, OZET, BOLUMNO, UYENO, FOZELIK, DİZİ, TASNIFNO, FIYAT, FIRMANO, ELISTAR, DILNO, ADET),  
BOLUM(BOLUMNO, BOLUMADI), GRUP(GRUPNO, GRUPADI),   
UNVAN(UNVANNO, UNVANADI),   
UYE(UYENO, ADSOYAD, BOLUM, GRUP, UNVAN, ADRES, MAIL, TEL, GSM)¸   
ODUNC(UYENO, DNO, ALISTAR, IADETAR),   
IADE(UYENO, DNO, GELISTAR)



**Tablolarla çalışma**

İlişkisel veritabanlarının temel elemanları iki boyutlu tablolardır. Tablolarla çalışmanın çok çeşitli yöntemleri vardır, başta her veritabanı yönetim sistemi kendi içinde tablolarla çalışabilecek bir arabirime sahiptir. Diğer bir yöntem SQL Explorer veya Database Desktop gibi bir uygulama kullanarak tablolarla çalışmadır. Bunlar içinde ve bunlardan bağımsız kullanılabilen en yaygın yöntem ise doğrudan SQL cümleleri kullanmaktır. Çünkü SQL hem veri tanımlama hem de veri kullanım dilidir.

**ER Modeli;**



**4. Bölüm Özeti**

İlişkisel cebir;

* İlişkisel veritabanının nasıl çalışacağına dair kurallara uygun tanımdır.
* Veritabanında verinin nasıl saklandığına dair bir ara yüzdür.
* SQL işlemlerini destekleyen matematiktir.

İlişkisel cebirdeki operatörler SQL operatörleri ile aynı isimlere sahip olsalar bile, aynı olmak zorunda değildirler. Mesela SELECT ifadesi SQL’de de ilişkisel cebirde de vardır. Fakat kullanımları aynı değildir. Veri tabanı yönetim sistemi kullanıcı her ne yazarsa, yazılanı veritabanına uygulamadan önce alıp, ilişkisel cebir işlemlerine çevirmelidir.

* İlişkisel cebirde temel işlemler:
  + *Seçim (Selection)(σ):* İlişkinin satırlarının bir alt kümesini seçer.
  + *Yansıtma (Projection)(π):* İlişkinin istenmeyen sütunlarını siler.
  + *Kartezyen çarpım (Cross-product)(χ):* İki ilişkiyi birleştirmeye yarar.
  + *Küme farkı (Set-difference)(-):* Birinci ilişkide bulunup ikinci ilişkide bulunmayan çoklular.
  + *Birleşim (Union)(∪):* Birinci ve ikinci ilişkide bulunan çoklular.
* Diğer işlemler:
  + Kesişim (intersection), bitişme (join), bölme(division), yeniden adlandırma (renaming): gerekli değil fakat çok kullanışlıdır.
  + Her işlem bir ilişki geri verdiğinden, işlemler birleşik olabilir. Dolayısıyla cebir kapalılık özelliğine sahiptir.

## Seçim (Selection) (σ)

* Seçim şartını gerçekleyen satırları seçer.
* Sonuçta tekrarlar yoktur.

## Sonucun şeması girdi ilişkisinin şeması ile aynıdır.

* Sonuç başka bir ilişkisel cebir işlemi için girdi olarak kullanılabilir.

## Yansıtma (Projection) (π)

* Yansıtma listesinde olmayan özellikleri siler.
* Sonuç şeması tam olarak yansıtma listesinde olan özellikleri, girdi olan ilişkideki isimlerle, içerir.
* Yansıtma işlemi tekrarları elemelidir.
  + Not: gerçek hayatta tekrarlar kullanıcı açıkça istemedikten sonra elenmez.

## Birleşim (Union), Kesişim (Intersection), Küme Farkı (Set-Difference)

* Bütün bu işlemler girdi olarak iki ilişki alır, ki bu iki ilişki birleşime uyumlu olmalıdır, bu da demek oluyor ki:
  + Aynı sayıda alana sahip olmalı.
  + Karşılıklı alanlar aynı tipte olmalılar.

## Kartezyen çarpım ( Cross-product) (χ)

* S1 χ R1 : S1 in her satırı R1 in her bir satırıyla eşleşir.
* Sonucun şeması S1 ve R1 deki her bir alana karşılık bir alan içerir.

## Şartlı Bitiştirme (Conditional Join) (⋈c)

R ⋈cS = σc(R χ S)

* Sonuç şeması kartezyen çarpımdaki gibidir.
* Kartezyen çarpımdan daha az çoklu içermesi daha verimli bir biçimde hesaplamayı sağlar.
* Teta bitiştirmesi diye de adlandırılır.

## Eşit Bitiştirme (Equijoin) (⋈e)

* e’nin herhangi bir eşitliğe karşılık geldiği özel bir şartlı bitiştirmedir.
* Sonucun şeması kartezyen çarpımdakine benzerdir, sadece eşitliğin belirlendiği alanın tek kopyası vardır.

## Bölme (Division)

Birincil işlem olarak kabul edilmez, fakat aşağıdaki gibi bir sorguyu açıklamak için kullanılır:

“Tüm elbiselerden almış bayanları bulun.”

A’nın 2 x ve yisimli alanı, B’nin yalnız y isimli bir alanı olsun.

* Eğer B deki y kümesi A’daki bir xile ilişkilendirilmişse ve A x ile ilişkili bir şekilde B deki tüm y’leri kapsıyorsa, bu xA/B kümesi içersindedir.
* Genellikle, x ve y herhangi iki alan listesi olabilir; y B’deki, x vey ise A’daki alanlardır.

**4.13. ÇALIŞMA SORULARI;**

**1.** Aşağıda oto galeriye ait arac, satis, alim ve müşteri tabloları verilmiştir.

* Create database kisa\_sinav
* use kisa\_sinav
* create table arac(aracno int primary key,model varchar(30) ,marka varchar(30),plaka varchar(10),fiyat money)
* create table musteri(mno int primary key,madi varchar(30),msoyadi varchar(30),madres varchar(30),mtel varchar(30))
* create table satis(stno int primary key,mno int foreign key references musteri(mno),
  + - aracno int foreign key (aracno) references arac(aracno),strh varchar(30),sfiyat money)
* create table alim(alno int primary key,mno int foreign key references musteri(mno),aracno int foreign key references arac(aracno),altrh varchar(15),afiyat money)

1. 2004 ve daha aşağı modeldeki araçların model, marka ve fiyat bilgilerini sorgulatınız.  
   select \* from arac where model<2004
2. 2000 ve 2009 arası modele sahip araçların model, marka ve fiyat bilgilerini sorgulatınız.  
   select \* from arac where model>2000 and model<2009
3. Galeriden hem araç alan hem de galeriye araç satan müşterilerin ad ve soyad bilgilerini sorgulatınız.  
   select madi, msoyadi from musteri,satis,alim
4. Ford marka araçların satış ve alım tarihini sorgulatınız.  
   select altrh, strh from arac,satis,alim where marka=’Ford’
5. Ford veya Opel marka araç alan müşterilerin ad, soyad ve aldığı aracın plaka kodunu sorgulatınız.  
   select madi,msoyadi,plaka from musteri,arac where marka=’Ford’ or marka=’Opel’
6. Fiat veya Opel marka ve modeli 2000 den fazla olan araçları alan ve satan müşterileri sorgulatınız.  
   select madi,msoyadi, from musteri,arac, alim , satis where marka=’Opel’ or marka=’Fiat’ and model>2000
7. Ahmed isimli müşterinin aldığı araçların plaka bilgilerini sorgulatınız.  
   select plaka from arac,alim,musteri where madi='Ahmed'
8. Fiyatı 10000 den fazla olan araçları alan müşterilerin tüm bilgilerini sorgulatınız.  
   select madi,msoyadi,madres,mtel from musteri,arac,alim where fiyat>10000
9. 5000 ile 50000 TL arasında satılan araçların marka, model ve satıldığı müşterilerin ad ve soyad bilgilerini sorgulatınız.  
   select madi, msoyadi, marka, model from arac,musteri,satis where fiyat>5000 and fiyat<50000
10. Alış fiyatı satış fiyatından yüksek olan araçların satıldığı müşterilerin ad, soyad ve satılan aracın marka ve model bilgilerini sorgulatınız.  
    select madi,msoyadi,marka,model from arac,musteri,satis,alim where afiyat>sfiyat

**5.5. ÇALIŞMA SORULARI**

**1.** Aşağıda bir oto galeriye ait arac, satis, alim ve musteri tablolarını sql kodlarıyla oluşturunuz.

* create table arac(aracno int primary key,model varchar(30) ,marka varchar(30),plaka varchar(10),fiyat money)
* create table musteri(mno int primary key,madi varchar(30),msoyadi varchar(30),madres varchar(30),mtel varchar(30))
* create table satis(stno int primary key,mno int foreign key references musteri(mno), aracno int foreign key (aracno) references arac(aracno),strh varchar(30),sfiyat money)
* create table alim(alno int primary key,mno int foreign key references musteri(mno),aracno int foreign key references arac(aracno), altrh varchar(15), afiyat money)
* **2.** insert into arac(aracno,model,marka,plaka,fiyat) values(3,2004,'Fiat Marea','60tt6060',16000)
* insert into arac(aracno,model,marka,plaka,fiyat) values(4,2000,'Renault Megane','60tt6061',14000)
* insert into arac(aracno,model,marka,plaka,fiyat) values(5,2007,'Ford Focus','60tt6062',28000)
* insert into arac(aracno,model,marka,plaka,fiyat) values(6,2005,'Volkswagen Golf','60tt6063',26000)
* insert into arac(aracno,model,marka,plaka,fiyat) values(7,1998,'Opel Astra','60tt6064',9000)
* insert into musteri(mno,madi,msoyadi,madres,mtel) values(3,'Turgut','Özseven','Turhal Tokat','0531654849')
* insert into musteri(mno,madi,msoyadi,madres,mtel) values(4,'Mustafa','Çağlayan','Meran Konya','025164854')
* insert into musteri(mno,madi,msoyadi,madres,mtel) values(5,'Ahmet','Karaal','Zile Tokat','014846531')
* insert into musteri(mno,madi,msoyadi,madres,mtel) values(6,'Murat','Beyaz','Turmahal Tokat','5464123198')
* insert into musteri(mno,madi,msoyadi,madres,mtel) values(7,'Bülent','Ayar','Turhal Tokat','897984564')
* insert into satis(stno,mno,aracno,strh,sfiyat) values(3,1,1,'04052010',17000)
* insert into satis(stno,mno,aracno,strh,sfiyat) values(4,4,5,'01062010',11500)
* insert into satis(stno,mno,aracno,strh,sfiyat) values(5,7,4,'15062010',27000)
* insert into satis(stno,mno,aracno,strh,sfiyat) values(6,2,1,'02072010',17500)
* insert into alim(alno,mno,aracno,altrh,afiyat) values(3,3,1,'08022010',15000)
* insert into alim(alno,mno,aracno,altrh,afiyat) values(4,6,1,'12042010',15500)
* insert into alim(alno,mno,aracno,altrh,afiyat) values(5,2,5,'15042010',9500)
* insert into alim(alno,mno,aracno,altrh,afiyat) values(6,1,2,'15052010',14000)
* insert into alim(alno,mno,aracno,altrh,afiyat) values(7,5,3,'22082010',26000)

**3.** Aşağıda verilen işlemleri gerçekleştirmek için gerekli sql ifadelerini yazınız.

a.) 2004 ve daha aşağı modeldeki araçların model, marka ve fiyat bilgilerini sorgulatınız.

select \* from arac where model<2004

b.) 2000 ve 2009 arası modele sahip araçların model, marka ve fiyatlarını sorgulatınız.

select \* from arac where model>2000 and model<2009

c.) 01.05.2010 tarihinden sonra 10000 den fazla fiyata yapılan satışları listeleyin.

Select \* from satis where fiyat>10000 and strh>’01.05.2010’

d.) Adının içerisinde y karakteri geçen ve adresi turhan olan müşterileri sorgulatınız.

Select \* from musteri where madi LIKE (‘%y%’) and madres like (’%turhan%’)

e.) Fiat veya Opel marka ve modeli 2000’den fazla olan araçların aracno,model,plaka ve fiyat bilgilerini sorgulatınız.

Select aracno, model, plaka, fiyat from arac where marka=’Opel’ or marka=’Fiat’ and model>2000

**4.** Aşağıdaki kullanıcıların şekilde erişim yetkileri verilmiştir. Bu yetkilendirmeyi yapmak için gerekli sql kodlarını yazınız.

grant select, insert, update, delete on musteri to A

grant select, insert, update, delete on satis to A

deny select, insert, update, delete on arac to A

deny select, insert, update, delete on alim to A

grant select, insert on musteri to B

deny update, delete on musteri to B

grant select, insert, update, delete on arac to B

grant select, delete on satis to B

deny update, select on satis to B

grant select, insert on alim to B

deny update,delete on alim to B

deny select, insert, update, delete on musteri to C

grant select, insert, update on arac to C

deny delete on arac to C

deny select, insert, update, delete on satis to C

grant select, insert, update, delete on alim to C

deny select, insert, update, delete on musteri to D

grant select on arac to D

deny insert, update, delete on arac to D

grant select, insert, update, delete on satis to D

deny select, insert, update, delete on alim to D

grant select, insert, update, delete on musteri to E

grant select, insert, update, delete on arac to E

grant select, insert, update, delete on satis to E

grant select, insert, update, delete on alim to E