KENT BİLGİ SİSTEMİ OLUŞTURMADA KULLANILABİLECEK VERİ MODELLERİNİN KARŞILAŞTIRILMASI

GÜRSEL GÜZEL
HÜSEYİN ALI KÖKLÜ

ÖZET :

Bu araştırmada, Kent Bilgi Sistemi oluşturma amacı ile kullanılabilecek yazılımların grafik olmayan bilgileri için kullandıkları veritabanları incelenmiş olup, bu veritabanı yazılımlarının kullandıkları veri modellerinden olan "ilişkisel veri modeli" ve "ağ veri modeli" karşılaştırılmıştır. Bu iki veri modeli ile taşınmaz bilgilerinin nasıl kayıt edilebileceği ve nasıl ulaşılabileceği üzerinde durulmuş ve iki veri modeli arasındaki benzerlik ve farklılıklar ortaya çıkarılmaya çalışılmıştır. Konu ile ilgili iki ayrı yazılım örneği hazırlanmış ve denenmiştir.

1.GİRİŞ:

Teknolojik gelişmelere paralel olarak kartografik çalışmalar da süratle gelişmekte ve yeni boyutlar kazanmaktadır. Bilgisayar teknolojisinin devreye girmesi ile bu gelişme daha da süratlenmiş ve kartografya kavramı "Modern Kartografya" ve son olarakta "Sayısal Kartografya" olarak yeni boyutlar kazanmıştır.

Sayısal Kartografya kavramı incelendiği zaman; değişik ortamlardan bilgilerin bilgisayara aktarılması ile başlayıp Coğrafi Bilgi Sistemi(CBS), Kent Bilgi Sistemi (KBS) gibi bilgi sistemlerinin oluşturulması, haritaların bilgisayar destekli ortamlarda basılması ve çoğaltılması, sayısal haritalardan değişik hesaplamaların yapılması (alan ve hacim hesaplamaları vb.), istatistiksel sonuçların çıkarılması, değişik açılardan perspektif görüntülerin elde edilmesi gibi devam eden bir yelpaze karşımıza çıkmaktadır.

Bu çalışmada kartografyanın ilgi alanı içerisine de giren Kent Bilgi Sistemlerinin oluşturulması ve kullanılması ele alınmış, bu amaçla kullanılabilecek yazılımların veri yapıları üzerinde durulmuştur.

Kent Bilgı Sistemlerinin oluşturulabilmeleri için kullanılan yazılımlar üç ana bölümden oluşmaktadır. Bunlar Grafik bilgilerin derlenmesi, işlenmesi, depolanması, sorgulanması ve güncelleştirilmesi gibi işlemlerin yapılmasını gerçekleştirebilecek *grafik* bölüm, grafik olmayan bilgilerin sisteme aktarılmaları, sıralanmaları, sorgulanmaları, depolanmaları ve benzeri işlemleri gerçekleştirilebilecekleri *veritabanı* bölümü ve grafik ile grafik olmayan bilgileri bir arada inceleyip, analiz yapabilecek, değişik yorumlamalar ve raporlamalar için kullanılabilecek olan *özel uygulama ve analiz* yazılımlarıdır.

Grafik ve grafik olmayan verilerin girilmesi, işlenmesi analız edilmesi, raporların alınması ve benzeri işlemlerin yapılabilmesi için sisteme girilecek olan verilerin neler olduğunun ve bu verilerin yapılarının nasıl olduğunun iyi incelenmesi ve bu verilere uygun yazılımların seçilmesi halinde, oluşturulacak olan bilgi sistemleri sağlıklı bir şekilde çalışmaması için hiç bir neden yoktur. Ancak seçilecek olan yazılımların kullandıkları veri yapılarının, kurulması istenen bilgi sistemine girilecek olan verilerin yapısına uygun olmaması durumunda bilgiye ulaşım zorlaşır ve zaman, performans gibi kayıplar ortaya çıkar.

Kent Bilgi Sistemlerinin temelini oluşturan bilgiler, taşınmaz mal bilgileri (parsel, bina, kat mülkiyeti vs.), teknik altyapı (Elektrik, su, kanalizasyon, gaz ve şebekeler) ve insandır. Bu bilgilerin hepsi de birbiri ile ilişkilidirler. Bu bilgilerin sisteme girilebilmesi için herşeyden önce sağlıklı ve uygun ölçekte altlık haritalara ihtiyaç olduğu açıktır.

KBS nin oluşturulabilmesi için yukarıda sözü edilen bilgilerin yapılarının ve aralarındaki ilişkilerin iyi incelenmesi gerekmektedir. Bu çalışmada, parsel ve insan (malik) bilgileri örnek olarak alınmış ve aralarındaki ilişkiler incelenerek ilişkisel veri modeli ve ağ veri modelini kullanan yazılımlarda bu verilerin sisteme nasıl aktarılacağı ve sorgulama gerektiği zaman nasıl ulaşılabileceği üzerinde durulmuştur.

2. KBS AMACI İLE KULLANILABİLECEK VERİ TABANI YAZILIMLARINDAN BEKLENENLER

Bugün piyasaya sürülmüş olan çok sayıda veritabanı yazılımı mevcut olmakla birlikte bu yazılımların herbirinin diğerine göre farklılıkları ve üstünlükleri vardır. Bu farklılık ve üstünlükler, veritabanı yazılımlarının üretimi sırasında kullanılan veri modeli, kullanılan programlama dili, üretilecek olan yazılımdan beklenen amaç gibi daha bir

çok nedenden kaynaklanmaktadır. Bu yazılımlar arasından KBS amacı ile kullanılabilecek olanlarının seçilebilmesi ise sisteme girilecek olan verilerin yapısı ve sistemden beklenenlerin ortaya konulması ile daha kolay hale gelir. Bu amaç için kullanılabilecek yazılımlardan beklenenler aşağıdaki gibi özetlenebilir.

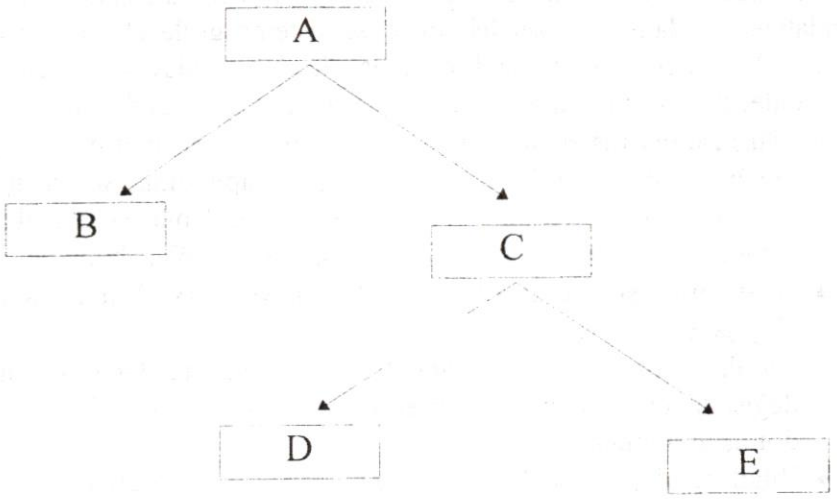
- Oluşturulması istenen bilgi sisteminin amacına hizmet edebilmelidir. Diğer bir deyimle sisteme girilecek olan bilgilerin yapılarının, sistem içersinde rahatça tasarlanması ve aralarındaki ilişkilerin oluşturulabilmesi olanaklı olmalıdır.
- Bilgilere çok hızlı ve güvenilir olarak erişim sağlanabilmelidir.
- Network sistemlerde (örneğin Nowel Network, Windows For Worksgrup vb.) çalışabilmelidir.
- İleriki zaman dilimi içersinde veri yapısında meydana gelebilecek olası bazı değişiklikler nedeni ile mevcut verilerin tümü kolayca yeni yapıya dönüştürebilmelidir.
- Bilgilere ulaşım esnek olmalı, kullanıcı değişik sorgulama ve yorumlamaları rahatlıkla yapabilmelidir. Bunun için de *yapılandırılmış sorgulama lisanı* (Structured Query Language) oluşturulabilmelidir.

3.VERİ MODELLERİ HAKKINDA GENEL BİLGİLER

Bilindiği üzere dünyada veri tabanlarının oluşturulmasında en yaygın olarak kullanılan üç veri modeli vardır. Bunlar Hiyerarşik veri modeli, İlişkisel veri modeli ve Ağ Veri modelidir. Bu çalışmada bu üç veri modeli üzerinde kısaca durulacaktır. Ayrıca bu veri modellerinden bazılarının (örneğin ilişkisel + ağ veri modelleri) bazı yazılımlarda bir arada kullanıldıkları bilinmektedir. Bu tür veri modelleri ise "karma" veri modellerini oluşturmaktadır.

3.1.HİYERARŞİK VERİ MODELİ

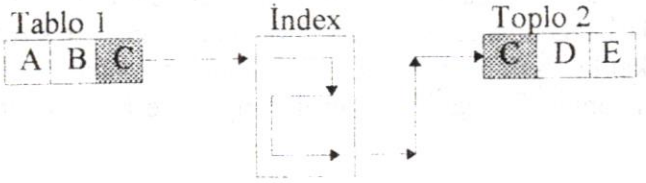
Bu veri modeli Ağ (Network) veri modelinin bir alt sistemi olarak ta tanımlanabilir. Bir kayıt tipinin yalnızca bir bağlantının üyesi (member) olmasına izin verilir. Ancak kayıt tipi birden fazla bağlantıya sahip (owner) olabilir. Yani, bir kayıtın yalnızca bir "sahip" i olabilir. Fakat bir "sahip" in birden fazla "üye" si olabilir. Aşağıdaki grafikte A bilgisi B ve C nin sahibi, C bilgisi A bilgisinin üyesi ve D,E bilgilerinin de sahibi konumundadır.



3.2.İLİŞKİSEL VERİ MODELİ

Bu veri modeli ise iki boyutlu tabloların (veya ilişkilerin) bağlantıları şeklinde yorumlanabilir. Tablolardaki kolonlar ve satırlar ("attributes" de denilir) arasında ilşkilerin kurulması ve değişikliklerin yapılması oldukça kolaydır. Bu modelde genellikle tablolar arasındaki ilişkilerin kurulması için ortak data alanları kullanılır.

Veri tabanlarının ilişkisel veri modelinden ağ veri modeline ve ağ veri modelinden ilişkisel veri modeline dönüştürülmeleri olanaklıdır.



3.3.AĞ (NETWORK) VERİ MODELİ

Bu veri modelinde kayıtlar arasındaki ilişkiler, bu kayıtlar arasına "set" ler yerleştirilerek kurulur. Bir set iki kayıt arasında bir "bir e çok (one-to-many)" ilişkiyi

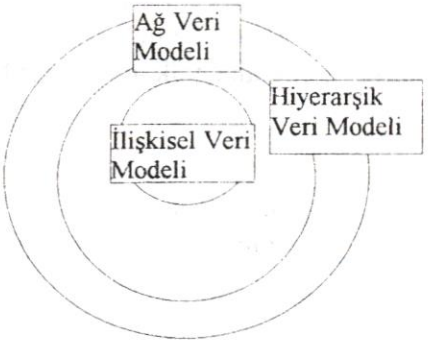
belirler. Yani bir kayıtın birden fazla başka kayıtlarla ilişkisinin olmasına olanak tanır. Örneğin bir futbol ligi birden fazla takıma sahiptir. (Lig burada "sahip = owner" takımlar ise "üye = member" durumundadır.) Öte yandan bir futbol takımı da birden fazla oyuncudan oluşmaktadır. (Bu kısımda ise takım "sahip" durumunda oyuncular ise "üye" durumundadır.)

3.4 KARMA VERİ MODELLERİ

Ağ veri modeli, veri tekrarını ve iki kayıt arasında doğrudan ilişki kurularak ortak dosya ve/veya kolon olayını ortadan kaldırmaktadır. Öte yandan ilişkisel veri modelinin kolay olması ve genellikle *ilk kayıtlara* daha kolay ulaşması nedeni ile bazı avantajlarının olduğu da bilinmektedir. Bu nedenle bu iki veri modelinin bir arada kullanılması oluşturulacak olan veritabanının maksimum esnekliğe sahip olmasını olanaklı kılar. Bu iki veri modelini bir arada kullanan yazılımların üretildikleri ve bir çok amaç için kullanıldıkları görülmektedir.

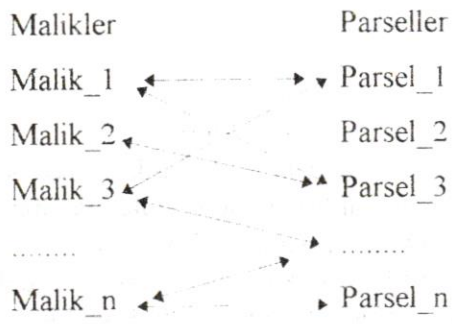
3.5 VERİ MODELLERİNİN KIYASLANMASI

Yukarıda özellikleri kısaca açıklanmaya çalışılan veri modellerinin aralarındaki ilişkiler aşağıdaki grafikte görülmektedir. Buna göre ilişkisel veri modeli hiyerarşik veri modelinin bir alt sistemi, hiyerarşik veri modeli ise ağ veri modelinin bir alt sistemidir.



4. KENT BİLGİ SİSTEMİNE GİRİLECEK OLAN VERİLER

Yukarıda da değinildiği gibi KBS ne girilecek olan verilere örnek olması açısından bu çalışmada yalnızca parsel ve malik bilgileri temel elemanları ile ele alınmıştır. Bir parsele ait pafta, ada, parsel no ve alanı parseli tanımlayan veriler ve bir malik ise adı, soyadı, baba adı, doğum tarihi ve cinsiyet bilgisi ile tanımlanmıştır. Bu verilere göre ağ veri modelini kullanan yazılımlara örnek olması açısından Raima Data Manager ve ilişkisel veri modellerine örnek olarak ta Dbase IV kullanılarak aşağıdaki örnek yazılımlar üretilmiştir. Bu yazılımlarda malik ile parsel arasındaki ilişki "hisse oranı" olarak düşünülmüştür. Ayrıca bir malikin n tane (n=1,2,3,...1000, ...) parselde hissesinin olabileceği ve bir parselin de n tane malikinin olabileceği düşünülmüştür. Diğer bir deyimle veriler arasında çok-çok (many to many) ilişkisi ele alınmıştır. Bununla birlikte herbir parselin ve mailikin yalnızca bir defa veritabanına kayıt edilmesi yani "tekil" olması esas alınmıştır.



4.1. RAIMA DATA MANAGER İLE VERİ TABANI OLUŞTURMA VE KULLANMA

Aşağıda verilen yazılım örneği Raima Data Manager (RDM) fonksiyonlarının Borland C++ ile kullanılması ile oluşturulmuştur.

```
database VT
{
   data file VTFILE1="vt.001"  contains maliktip;
   data file VTFILE2="vt.002"  contains parseltip;
   data file VTFILE3="vt.003"  contains conn;
   key  file VTFILE4="vt.004"  contains mlk,adi,soyadi,baba_adi,dogum_tar;
```

```
key  file VTFILE5="vt.005"  contains prs;

record maliktip
{
    char                cinsiyet;
    key char            adi[20];
    key char            soyadi[20];
    key char            baba_adi[20];
    key char            dogum_tar[12];
    compound unique key mlk
    {
     adi;
     soyadi;
     baba_adi;
     dogum_tar;
    }
}

record parseltip
{
    double      alan;
    long        ada_no;
    char        parsel_no[11];
    char        pafta_no[11];
    compound unique key prs
    {
        pafta_no;
        ada_no;
        parsel_no;
    }
}
record conn
{
    char  hisse[20]
}
set SET_MLTOCON
{
    order       last;
    owner       maliktip;
    member      conn;
}
set SET_PRSTOCON
```

```
    {
        order       last;
        owner       parseltip;
        member      conn;
    }
}
```

4.1.1.VERİ TABANINA KAYIT EKLEME :

Oluşturulan bu veri yapısına göre veri tabanına kayıt olark eklemek istediğimiz. bilgiler

Malik :		***Parsel:***	
Adı	: Gürsel	Pafta No	: F22-a-04-b-1-c
Soyadı	: Güzel	Ada No	: 506
Baba Adı	: Nail	Parsel No	: 86
Doğum Tarihi	: 30.05.1966	Alanı	: 386.25
Cinsiyeti	: Erkek		

Malikin parseldeki hissesi 1/2 olsun. Bu iki kayıtın veri tabanına eklenmesi ve aralarında ilişkinin kurulabilmesi için aşağıdaki program parçası yeterlidir.

```
// değişkenlerin tanımlanması
maliktip        malik;
parseltip       parsel;
conn            con,

// malik ve parsel bilgilerinin ilgili değişkenlere atanması
strcpy(malik.adı, "Gürsel");
strcpy(malik.soyadı, "Güzel");
strcpy(malik.baba_adı, "Nail");
strcpy(malik.dogum_tar, "30.05.1966");
malik.cinsiyet = 'E';

parsel.ada_no  = 506;
parsel.alanı   = 386.25;
strcpy(parsel.parsel_no,"86");
strcpy(parsel.pafta_no, "F22-a-04-b-1-c");

strcpy(con.bilgi, "1/2");

// Veri tabanının açılması ve kayıt işlemi
d_open("vt","o");
if (d_fillnew ( MALIKTIP, &malik , 0 ) != S_OKAY)
```

```
{
        if (d_fillnew ( MALIKTIP, &malik , 0 ) == S_DUPLICATE)
                printf("Bu Malik veri tabanında mevcut....");
        else
                printf("Veri tabanına kayıt yapamıyorum...");
        d_close();
        return;
}

if (d_fillnew (PARSELTIP, &parsel, 0 ) != S_OKAY)
{
        if (d_fillnew (PARSELTIP, &parsel, 0 ) == S_DUPLICATE)
                printf("Bu Parsel veri tabanında mevcut....");
        else
                printf("Veri tabanına kayıt yapamıyorum.. ");
        d_close();
        return;
}

d_setor(SET_MLTOCON,0);
d_setor(SET_PRSTOCON,0);
d_fillnew(CONN, &con, 0);
d_connect(SET_MLTOCON,0);
d_connect(SET_PRSTOCON,0);
d_close();
```

4.1.2. VERİ TABANINDA BULUNAN KAYITLARA ULAŞIM :

Parsel bilgilerine ulaşılmak istenen malik:

Adı : Gürsel
Soyadı : Güzel
Baba adı : Nail
Doğum Tarihi : 30.05.1966

```
// değişkenlerin tanımlanması
int             i;
struct mlk      mlk;
conn            con;
maliktip        malik;
parseltip       parsel;

//değişkenlere data atama
```

```
strcpy(malik.adı, "Gürsel");
strcpy(malik.soyadı, "Güzel");
strcpy(malik.baba_adı, "Nail");
strcpy(malik.dogum_tar, "30.05.1966");

// Veri tabanının açılması ve kayıtlara ulaşım
d_open("vt","o");
if (d_keyfind(MLK,&mlk,0)==S_OKAY)
{
        d_recread(&malik,0);
        d_setor ( SET_MLTOCON , 0 ) ;
        if ( d_findfm ( SET_MLTOCON , 0 ) == S_OKAY )
        {
                i = 0;
                do
                {
                        i++;
                        d_recread(&con,0);
                        d_setmr ( SET_PRSTOCON , 0 ) ;
                        d_findco ( SET_PRSTOCON , 0 ) ;
                        d_recread ( &parsel, 0 ) ;
                        printf("Malikin %d.nci parselinün",i);
                        printf("Pafta No        : %s",  parsel.pafta_no);
                        printf("Ada No          : %ld", parsel.ada_no);
                        printf("Parsel No       : %s",  parsel.parsel_no);
                        printf("Alanı           : %.2lf",parsel.alan);
                        printf("Malikin Hissesi: %s",  con.bilgi);
                } while ( d_findnm ( SET_PRSTOCON , 0 ) == S_OKAY ) ;
                printf("Bu Malikin %d adet parselde hissesi var",i);
        }
        else
        {
                printf("Bu malike ait herhangi bir parsel kayıtlı değil değil....");
        }
}
else
{
        printf("Bu malik Veritabanında kayıtlı değil....");
}
d_close();
```

4.2. DBASE IV İLE VERİ TABANI OLUŞTURMA VE KULLANMA

Aşağıda verilen yazılım, Dbase IV yazılımı yardımı ile oluşturulan parsel ve malik tablolarının birbiri ile olan ilişkilerini düzenlemek amacı ile oluşturulmuş ve Dbase IV altında koşmaktadır.

**** Bu program Malik.DBF ile Parsel.DBF kutukleri arasında His_hak.DBF kütüğü * vasıtasıyla ÇOK tan ÇOKa ilişki kurar Ana program ARA.PRG yardımcı *programlar ARA1.PRG ARA2.PRG dir.***

```
SET TALK OFF
SET ECHO OFF
SET STATUS OFF
SET SAFETY OFF
SET CONFIRM OFF
SET SCOREBOARD OFF
SET DATE BRITISH
SET DELETE ON

        CLOSE DATABASES
        !DEL *.NDX

        USE MALIK
        INDEX ON ADI+SOYADI+BABAADI TO MALIK

        USE PARSEL
        INDEX ON ADA+PARSEL TO PARSEL

        USE HIS_HAK
        INDEX ON MALIK_REC TO HIS_HAK
        INDEX ON parsel_REC TO HIS_HAK2

        CLOSE DATABASES

 DO WHILE .T.
         CLEAR
          @0,1 SAY "          MALIK & PARSEL   ILISKI PROGRAMI"
                @1,1 TO 3,78 DOUBLE

        YANIT=" "
        @2, 2 SAY " 1 Malike gore ARA    2- Parsele gore ARA    3-
CIKIS " GET YANIT
        READ
        YANIT=VAL(YANIT)
DO CASE
   CASE YANIT=1
        DO ara1
   CASE YANIT=2
        DO ara2
```

```
   CASE YANIT=3

        close databases
        clear
        QUIT

   OTHERWISE
                AA="H"
                @15,49 TO 17,78
                @16,50 SAY " ÇIKMAK ÿST. EMÿN. ?[ E/H ]"
                @16,77 GET AA
           READ
             AA=UPPER(AA)
                IF AA="E"
                      CLOSE DATABASE
                      QUIT
                ELSE
                LOOP
                ENDIF
        LOOP
ENDCASE
ENDDO

*  Bu program Malik.DBF ile Parsel.DBF kutukleri arasinda
*  His_hak.DBF kutugu vasitasiyla COKtan COKa iliski kuran
*  ARA.PRG nin bir alt yardimci programidir

SELECT 1
        USE MALIK INDEX MALIK
SELECT 2
        USE PARSEL INDEX PARSEL
SELECT 3
        USE HIS_HAK INDEX HIS_HAK

CLEAR
           XADI="                    "
        XSOYADI="                    "
            @ 4,10 SAY "MALIK ADINI GIRINIZ     : " GET XADI
            @ 5,10 SAY "MALIK SOYADINI GIRINIZ : " GET XSOYADI
            READ
                KAYIT=0
                XADI    = UPPER(XADI)
                XSOYADI = UPPER(XSOYADI)
        SELECT 1
          xadisoy=  XADI+XSOYADI
             SEEK xadisoy
                kayit=recno()
                IF RECNO()>RECCOUNT()
                   ? "BOYLE BIR ADA PARSEL  YOK"
                   WAIT " "
                   RETURN
                ENDIF
        SELECT 3
          SEEK KAYIT
              xparsel=parsel_rec
```

```
                xpay     =hissepay
                xpayda   =hissepayda
  ? "    ADI              SOYADI                    ADA      PARSEL    HISSE"
         DO WHILE MALIK_REC=KAYIT
                SELECT 2
                   GO xPARSEL
                   ? XADI,XSOYADI,ADA,PARSEL,xpay,xpayda
                  SELECT 3
                  SKIP +1
                xparsel=parsel_rec
                xpay     =hissepay
                xpayda   =hissepayda
         ENDDO
       WAIT " Dokum bitmistir bir tusa dokununuz....!    BEKLIYORUM"
         CLOSE DATABASES
RETURN

*  Bu program Malik.DBF ile Parsel.DBF kutukleri arasinda
*  His_hak.DBF kutugu vasitasiyla COKtan COKa iliski kuran
*  ARA.PRG nin bir alt yardimci programidir

SELECT 1
        USE MALIK
SELECT 2
        USE PARSEL INDEX PARSEL
SELECT 3
        USE HIS_HAK INDEX HIS_HAK2

 DO WHILE .T.
            XADA      =0
            XPARSEL   =0
            @ 4,40 SAY "ADA NUMARASINI  GIRINIZ     : " GET XADA
            @ 5,40 SAY "PARSEL NUMARASINI  GIRINIZ : " GET XPARSEL
               READ
    SELECT 2
            xadaparsel=xada+xparsel
            seek xadaparsel
            kayit=recno()
         IF RECNO()>RECCOUNT()
                ? "BOYLE BIR ADA PARSEL  YOK"
               WAIT " "
               RETURN
         ENDIF

    SELECT 3
                  SEEK KAYIT
                   XMALIK = MALIK_REC
                  xpay     = hissepay
                  xpayda   = hissepayda
@ 7,40
   ? "       ADI                SOYADI                HISSE",XADA,XPARSEL

              DO WHILE parsel_REC=KAYIT
                    SELECT 1
                       GO xMALIK
```

```
                    ? ADI,SOYADI,xpay,xpayda

                   SELECT 3
                   SKIP +1
                  XMALIK = MALIK_REC
                   xpay    =hissepay
                   xpayda  =hissepayda
            ENDDO
      WAIT " Dokum bitmistir bir tusa dokununuz....!    BEKLIYORUM"
ENDDO
 CLOSE DATABASES

RETURN
```

5.SONUÇ :

Yukarıda verilen yazılım örneklerinde de görüldüğü gibi :

- Ağ veri modelinde bilgilerin tekil (unique) olmalarının sağlanması oldukça kolaydır. Veri tekrarı yoktur. Buna karşılık ilişkisel veri modelinde veri tekrarı vardır. Bu veri tekrarını önlemek için ara işlemlere ihtiyaç durulur. Ayrıca verilerin tekil olmalarını sağlamak güçtür. Yukarıdaki örnekte de görüldüğü gibi veri tekrarını önlemek için ara bir tabloya gerek vardır. Bu ara tablo çoktan-çoka ilişki için kullanılmıştır. Ancak Malik.DBF veya Parsel.DBF tablolarından herhangi bir kaydın silinmesi veya dosyaların sıralanmaları halinde bütün kayıtların karışacağı açıktır. Buna karşılık ağ veri modeli çok daha güvenlidir.
- Ağ veri modelinde güncelleştirme daha kolaydır. Bunun nedeni ise veri tekrarı olmamasıdır.
- Ağ veri modeline göre sistem tasarlamak, kurmak ve kullanmak deneyimli personel gerektirmektedir. Buna karşılık ilişkisel veri modeline göre sistem tasarlamak daha kolaydır.
- Ağ veri modelinde verilere ulaşım çok hızlıdır. Buna karşılık ilişkisel veri modelinde veri sayısı arttıkça performans düşmektedir.

- Ağ veri modelinden ilişkisel veri modeline veya ilişkisel veri modelinden ağ veri modeline veri aktarmak olanaklıdır. Çünkü her iki modelde de çoktan-çoka ilişki kurmak (ilişkisel veri modelinde oldukça zor da olsa) olanaklıdır.

- Ağ veri modelini kullanacak olan grafik yazılımın sağlıklı bir yapıya sahip olması ve bu yazılımları kullacak olan personele iyi eğitim verilmesi halinde ağ veri modeli diğerlerine göre oldukça fazla üstünlükler sağlamaktadır.

Kaynaklar :

Borland C++ 4.0 Programmer's Guide	Borland international, inc 100 Borland Way, Scotts Valley , CA 95067-3249
Database Management	Fred R. McFadden Department of Business Information Systems, University Of Colorado Jeffrey A. Hoffer School Of Business, India University
Raima Data Manager user's guide	Raima Corporation 1605 NW sammamish Rd. Suite 200 Issaquah, Wa 98027 USA
An Introduction to Database Systems	Addison-Wesley System Programming Series IBM Corporation