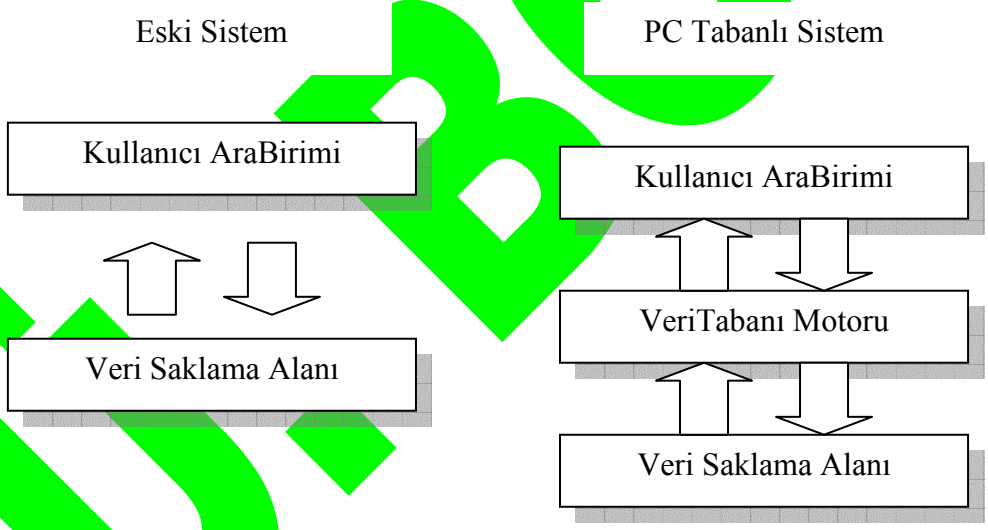
Dr. Halil Yurdugül
yurdugul@hacettepe.edu.tr

Veri Tabanı ve Visual Basic

Geçmişte, random dosya ve yapı değişkenleri ile oluşturulan kayıtlar bugünkü veri tabanı uygulamalarının temelini oluşturmaktadır. Random dosya ve yapı değişkenleri ile oluşturulan veri tabanı programlama geleneği daha sonra yerini Clipper, Dbase, Paradox gibi uygulamalara bırakmıştır. Ancak bu uygulamalar işletim sistemi desteğinden bağımsızdı ve karakter yönelimli (command oriented) idi. Daha sonraları, bilgisayarlardaki veri tabanı uygulamalarının önem kazanması ile daha üst düzey veri tabanı uygulamaları, veri tabanı ortam geliştiricileri ve işletim sistemi desteği ile veri tabanları oluşturulmaya başlandı. Bunlara en önemli örnekler ise Access, Oracle, MySql vb.

Veritabanlı ile işletim sistemlerinin etkileşimlerinde arabirim olan Veri Tabanı Motorları (DataBase Engine) geliştirildi ve ortaya çıkan model aşağıda verildiği gibi bir ortam geliştirildi.



Kullanıcı Arabirimi ve Uygulama Kodları:

Kullanıcının etkileşimde olduğu ortamdır. Geliştirme ortamı ya uygulama yazılımı şeklindedir ya da kod girişi şeklindedir.

VeriTabanı Motoru:

Kullanıcı arabirimi ile veriler arasında bir arabirim oluşturur. Aynı zamanda da işletim sisteminin bir parçası olarak sistem çağrısı şeklindedir.

Özellikle Microsoft, VeriTabanı uygulamalarında, VeriTabanı motoru olarak MS Jet VeriTabanı Motorunu (MS Jet DataBase Engine) kullanmaktadır. Bu motor, sistem

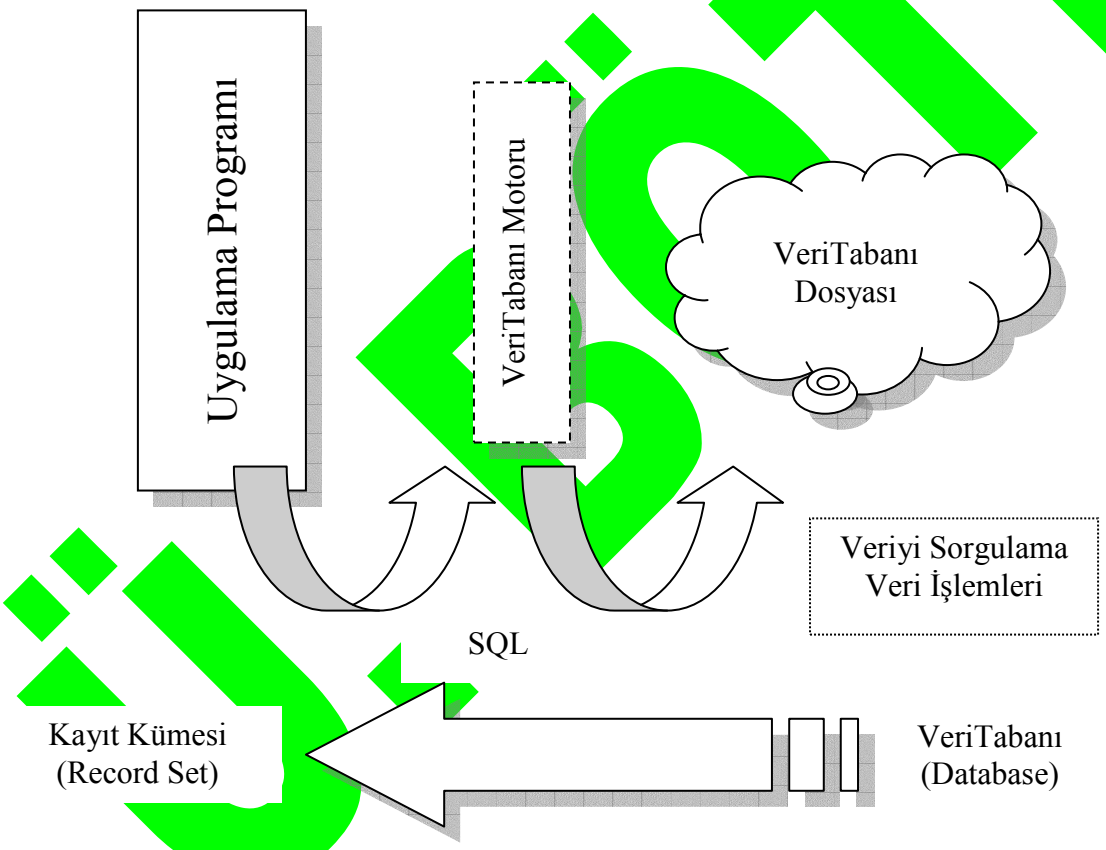
Dr. Halil Yurdugül
yurdugul@hacettepe.edu.tr

çağrılarından oluşan DLL dosyalarını içermektedir. Bu motorlar aynı zamanda **SQL** sorgulama diline uyumludur.

Veri Saklama Alanı:

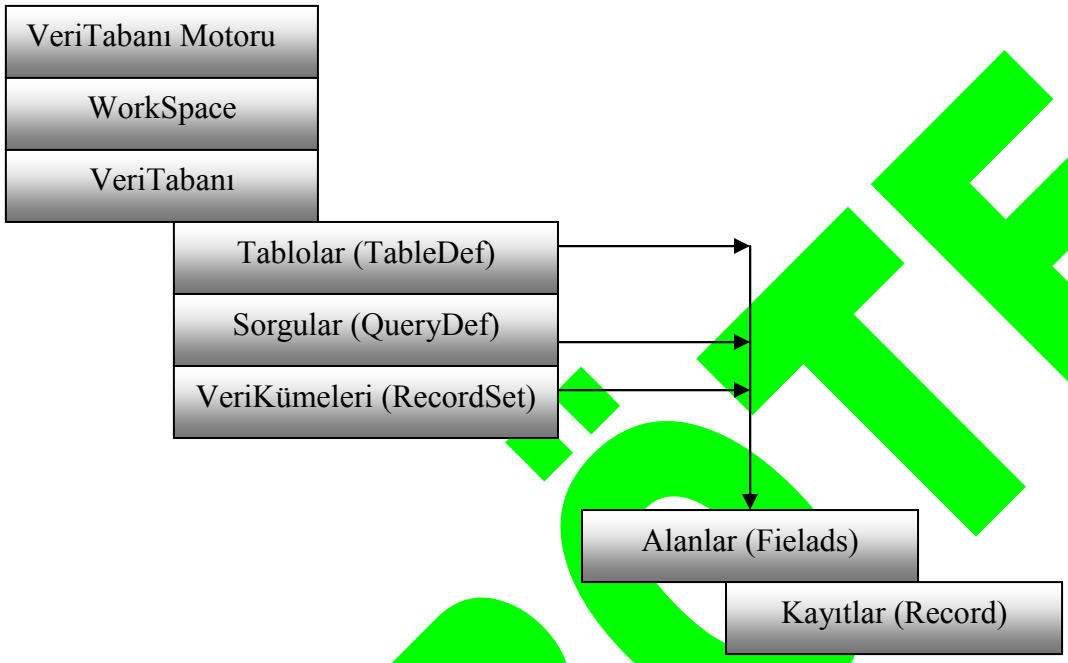
Veri saklama alanları, elektro-manyetik ortamlardır. Ancak bu modelde kastedilen ve ele alınan, genellikle veri tabanı dosyalarıdır.

Bu etkileşimler aşağıdaki gibi tanımlanabilir.



Dr. Halil Yurdugül
yurdugul@hacettepe.edu.tr

Yukarıda tanımlanan etkileşim, bir model gereği şu şekilde Visula Basic (Microsoft) geliştirme ortamında şu hiyerarşi ile ele alınırlar.



Hiyerarşi şu şekilde açıklanabilir:

Kayıtlar, alanlardaki kayıtlı verilerdir. Kayıtlar, alanların içindedir. Alanlar, tabloların ya da sorguların alanları olabilir. Tablo ve sorgular veritabanını oluşturur. Veritabanları ise bir araya gelerek çalışma alanlarını (WorkSpace) oluşturur. Bu hiyerarşik bütün, veri erişim nesnesi (Data Access Object-DAO) olarak ifade edilen bir nesne ile yönetilir.

Verilere yönelik bilgi uygulama programından veritabanına ya da veritabanından uygulama programına göredir. Ancak buradaki amaç, DAO ya da SQL ile belirlenir. DAO ile verilere dolaysız erişim olanaklı iken, SQL ile uygulama kodları yazmak suretiyle veriye erişim ve veri/veritabanı işlemleri olanaklıdır.

SQL (Yapısal Sorgulama Dili - Structured Query Language)

SQL, bir veritabanı programlama dilidir ve tablolar üzerinde sorgulama özelliğine sahiptir. Bu dilin iki özelliğinden söz etmek olanaklıdır. Bunlar sırasıyla; Veri Tanımlama Dili (Data Defination Language-DDL) ve Veri İşleme Dili (Data Manipulation Language-DML).

Dr. Halil Yurdugül
yurdugul@hacettepe.edu.tr

Veri Tanımlama Dili:

Bu veritabanı dilinin; varolan bir veritabanına yeni tablolar ya da tablolara alanlar eklemek (CREATE), bir veri tabanından tablo silmek (DROP) ve bir tabloya alan eklemek ya da alan özelliklerini değiştirmek (ALTER) gibi bir işlevselliği vardır.

Örnekler:

```
CREATE TABLE Tablo_Adi ([Alan_Adi] Alan_Ozellik
CREATE TABLE Kisiler ([Ad] TEXT (25), [SoyAd] TEXT (25))
ALTER TABLE Tablo_Adi {ADD {COLUMN field type[(size)] [NOT NULL]
                       DROP {COLUMN field I CONSTRAINT indexname} }
ALTER TABLE Kisiler ADD COLUMN Eposta TEXT (30)
DROP {TABLE Tablo_Adi }
DROP TABLE Kisiler
```

Veri İşleme Dili:

Bu veritabanı dilinin ise; bir tablodaki kayıtlardan belirli ölçütlere göre sorgulama yapmak (SELECT), tabloya yeni kayıtlar ekelemek (INSERT), varolan kayıtlar üzerinde değişiklikler yapmak (UPDATE) ve tablodan kayıt silmek (DELETE) gibi işlevselliği vardır.

Örnekler:

```
SELECT Alan_Adi FROM Tablo_Adi WHERE Ölçütler
                               ORDER BY
                               HAVING
                               GROUP BY

INSERT INTO Tablo_Adi (Alan_Adi) VALUES (Deger)
INSERT INTO Kisiler (Ad, SoyAd) VALUES ('Ali','Güzel')
```

SQL her ne kadar programlama dili olarak ifade edilse de aslında değişkenleri bir tablonun alanları, verileri ise alanlara ilşkin kayıtlar olan ve en önemli özelliği ise birer sql cümleciğinden (sql clauses) oluşmasıdır. Bu dil yapısı, yukarıda belirtilen DDL ve DML komutlarına eklenerek oluşturulurlar. Örneğin;

SELECT *Alan_Adi* **FROM** *Tablo_Adi* **WHERE** *Alan_Adi*='Bilgisayar';

SELECT → DML komutu
FROM,WHERE → Sql Cümleciği
Alan/Tablo Adi → İşlem alanları

Bu sorgu cümlesi ise; belirtilen tablonun belirtilen alanında arama yaparak, Alan_Adi'nda kayıtlı "Bilgisayar" kayıtlı bilgileri seçmesidir (görüntülemesi).

Dr. Halil Yurdugül
yurdugul@hacettepe.edu.tr

SQL cümleciğinde kullanılan atomlar:

FROM: Tablodan seçilecek/görüntülenecek alanları belirler. Birden fazla alan görüntülenecek ise alan adları "," ila ayrılır.

WHERE: Koşul ifade eder. Seçilen alanda belirli bir ölçüte göre seçme yapılmasını sağlar.

GROUP BY: Özel/belirtilen gruplara göre kayıtların seçilmesini sağlar.

ORDER BY: Seçilen kayıtların görüntülenmesi sırasında sıralama ölçütünü belirlen.

SQL cümleciğinde kullanılan operatörler:

Mantıksal Opertörler:

Birden fazla koşul söz konusu olduğunda kullanılır ve koşulları birbirine bağlar. AND, OR, NOT bu operatörlere örnektir.

SELECT *Alan_Adi* **FROM** *Tablo_Adi* **WHERE** *Kosu11* **AND** *Kosul2*

Karşılaştırma Opertörler:

Ölçüt olarak kullanılan alan değerleri (kayıtlar) için karşılaştırma amaçlı kullanılır.

< → Daha Küçük
<= →Daha küçük ya da eşit
> → Daha Büyük
>= →Daha büyük ya da eşit
= → Eşit
<> → Eşit Değil
BETWEEN →İki değerin arasındaki değerler
LIKE →Benzerlik gösteren değerler
IN → Özel kayıtlar için kullanılır.

Örnekler:

SELECT LastName, FirstName FROM Employees;

Dr. Halil Yurdugül
yurdugul@hacettepe.edu.tr

1) Aşağıdaki yapıya göre bir access veritabanı oluşturunuz.

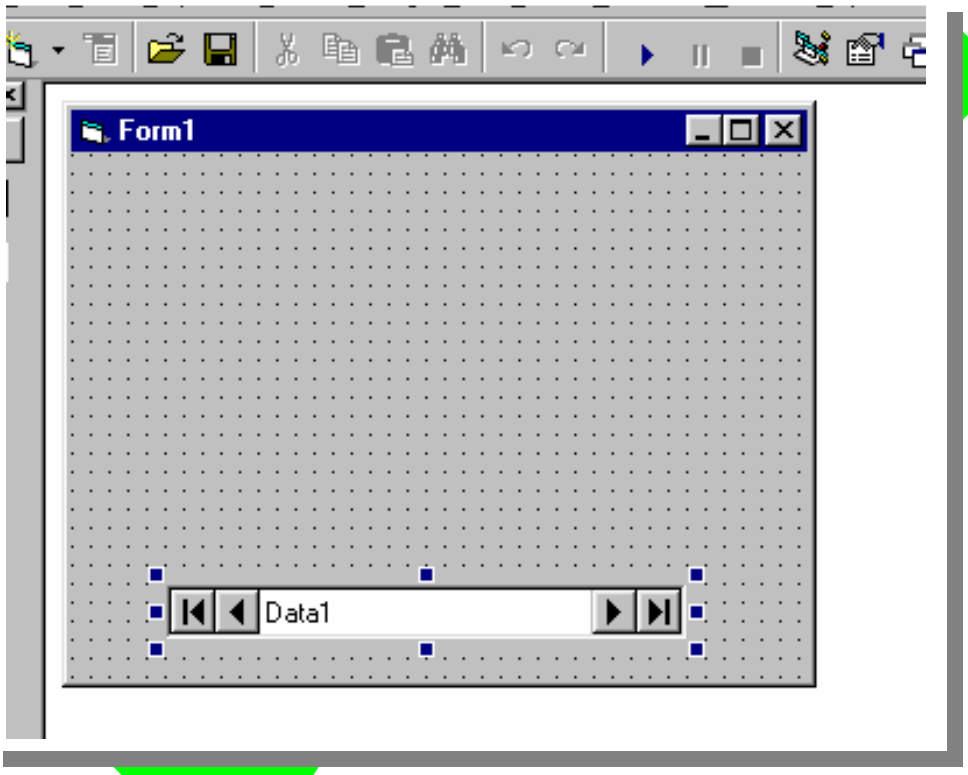
Tablo_Adi→ "Rehber"

Dosya Adı→"Deneme.mdb"

Rehber:

Ad →	String
SoyAd→	String
E-Posta→	String
Dogum_Tarihi→	String

2) Formunuza bir DAO nesnesi ekleyiniz.



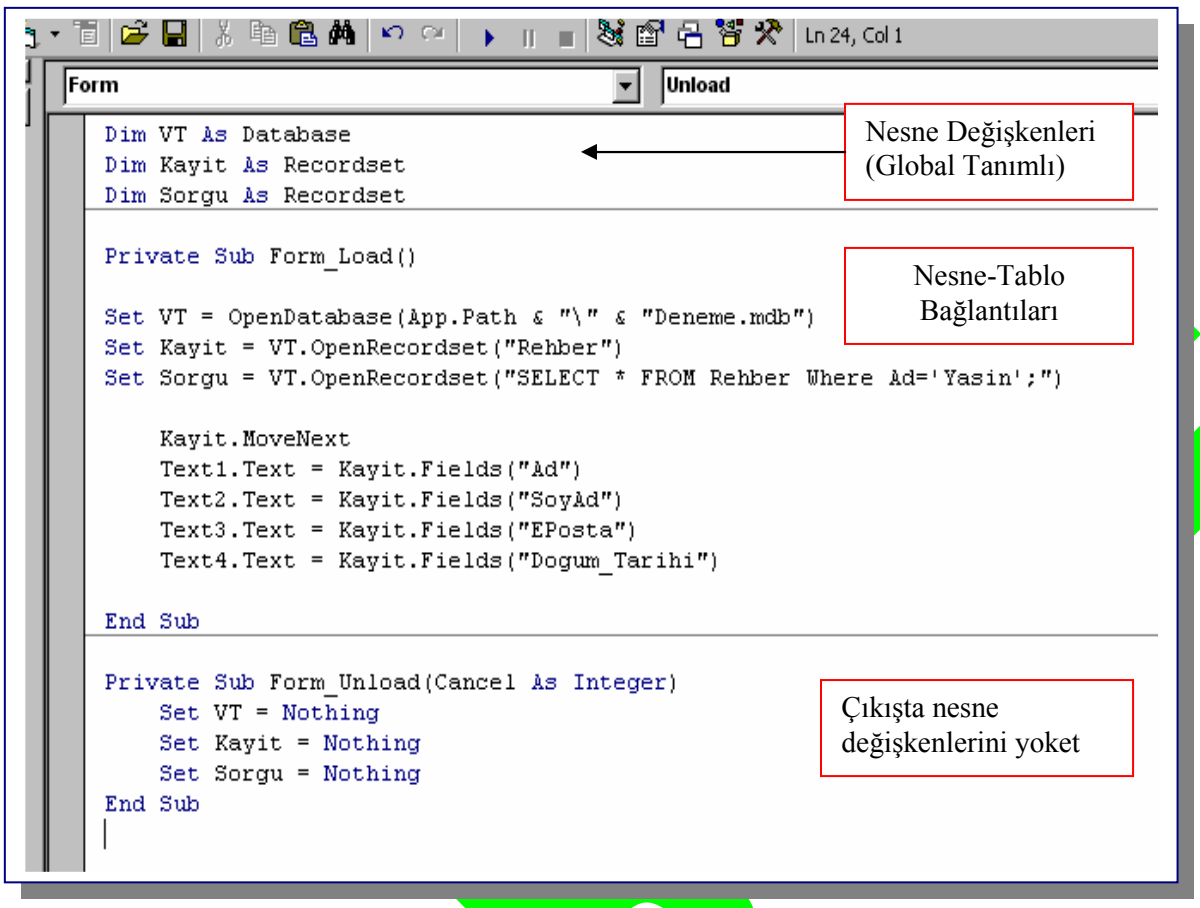
3)

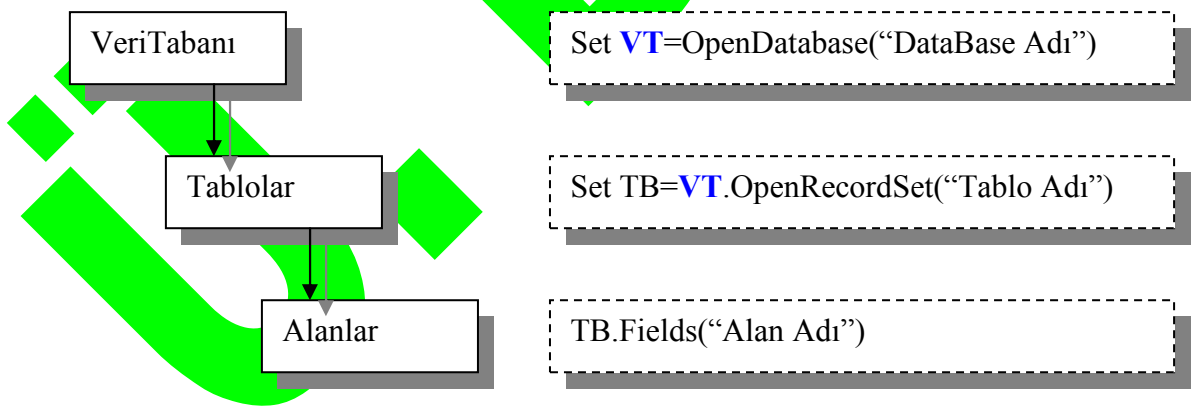
DAO nesnesi, "Deneme.mdb" veritabanı dosyasındaki bir tabloyu temsil edebilir. Bu tablomuzun adı: "Tablo" dur. Tablo içerisindeki verileri program ortamına ancak, kayıt kümesi (RecordSet) şeklinde alacağımızdan dolayı öncelikle programımızın içerisinde, DAO'yu öncelikle tanımlamamız gerekmektedir.

VeriTabanı → Deneme.mdb

KayıtKümesi→ Rehber

Dr. Halil Yurdugül
yurdugul@hacettepe.edu.tr





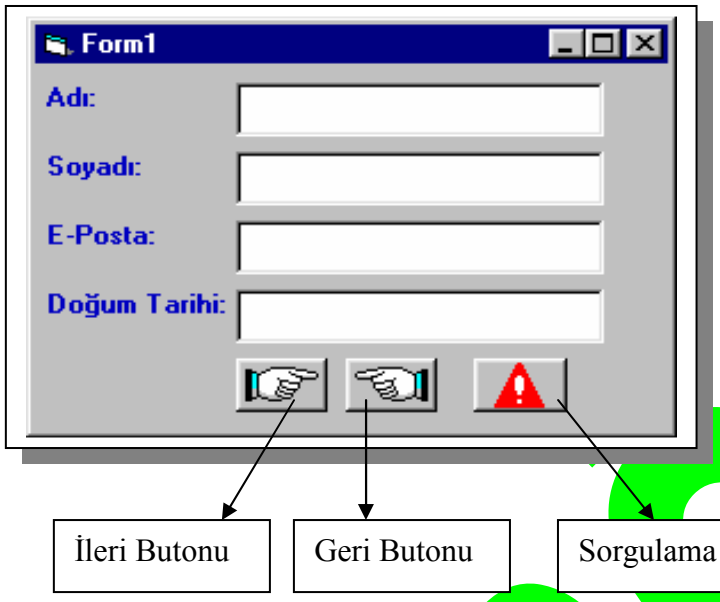
Böylelikle Tablodaki tüm kayıtlar TB adlı nesne değişkenindedir. Aşağıdaki örnekte, Tablo verilerini temsil eden TB değişkenini baştan sona okutup kontrol edebilirsiniz.

```
Do Until TB.EOF
      If TB.Field("Alan_Adi")="Ahmet" Then Sayac=Sayac+1
      TB.MoveNext
Loop
```

Dr. Halil Yurdugül
yurdugul@hacettepe.edu.tr

Projeyi geliştirelim:

Data1 (DAO) nesnesinin visible özelliğini “False” yapalım ve aşağıdaki alanları ekleyelim.



İleri Butonu | Geri Butonu | Sorgulama Butonu

```
General Declaration Kısmı:
      Dim VT As Database
      Dim Kayit As Recordset
      Dim Sorgu As Recordset
```

```
Private Sub Form_Load()

Set VT = OpenDatabase(App.Path & "\" & "Deneme.mdb")
Set Kayit = VT.OpenRecordset("Rehber")
Set Sorgu = VT.OpenRecordset("SELECT * FROM Rehber';")

    Text1.Text = Kayit.Fields("Ad")
    Text2.Text = Kayit.Fields("SoyAd")
    Text3.Text = Kayit.Fields("EPosta")
    Text4.Text = Kayit.Fields("Dogum_Tarihi")

End Sub
```

```
Private Sub Form_Unload(Cancel As Integer)
    Set VT = Nothing
    Set Kayit = Nothing
    Set Sorgu = Nothing
End Sub
```

Dr. Halil Yurdugül
yurdugul@hacettepe.edu.tr

Tablo verileri Kayit adlı nesne değişkeninde tutulacaktır. “Select * FROM Rehber;” SQL cümleciğinin sonuçları ise Sorgu nesne değişkeninde saklanacaktır.

```
Private Sub ileri_Click()
On Error GoTo Hata
    Kayit.MoveNext
    Text1.Text = Kayit.Fields("Ad")
    Text2.Text = Kayit.Fields("SoyAd")
    Text3.Text = Kayit.Fields("EPosta")
    Text4.Text = Kayit.Fields("Dogum_Tarihi")
Hata:
    Exit Sub
End Sub
```

```
Private Sub geri_Click()
On Error GoTo Hata
    Kayit.MovePrevious
    Text1.Text = Kayit.Fields("Ad")
    Text2.Text = Kayit.Fields("SoyAd")
    Text3.Text = Kayit.Fields("EPosta")
    Text4.Text = Kayit.Fields("Dogum_Tarihi")
Hata:
    Exit Sub
End Sub
```

```
Private Sub sorgulama_Click()
'Bugün Doğanlar
    Text1.Text = ""
    Text2.Text = ""
    Text3.Text = ""
    Text4.Text = ""

Do Until Sorgu.EOF
 If Sorgu.Fields("Dogum_Tarihi") = Format(Now, "d.m.y") Then
    Text1.Text = Sorgu.Fields("Ad")
    Text2.Text = Sorgu.Fields("SoyAd")
    Text3.Text = Sorgu.Fields("EPosta")
    Text4.Text = Sorgu.Fields("Dogum_Tarihi")
    Exit Sub
 End If
    Sorgu.MoveNext
Loop
End Sub
```

Dr. Halil Yurdugül
yurdugul@hacettepe.edu.tr

Uygulama 2 Ödevi

Öğrencinin

Adı	→ String	
Soyad	→ String	
Numarası	→ String	
Cinsiyeti	→ String*1	
Dersin Adı	→ String	
Dersin Kodu	→ String*6	(BTÖ304 gibi)
Ders Yılı	→ String	(2005 gibi)
Dönemi	→ String	(Güz/Bahar)
Notu	→ Single	(50 ya da 78 gibi)

Veri Girişleri

- Öğrenci Bilgi Girişi
- Öğrenci Bilgi Değişikliği
- Not Girişi

Veri Sorgulama

- Ders Kodu-Yılı-Dönemi ne göre öğrenci sayıları (BTÖ304-2005-Bahar)
  - ‘2005 Bahar Döneminde BTÖ 304 dersini alan öğrenci listesi
- Derse ilişkin kız ve erkek öğrencilerin başarı ortalamaları (BTÖ304-2005-Bahar-“E”-Notu)
  - ‘2005 Bahar Döneminde BTÖ 304 dersini alan erkek ve kız öğrencilerin ortalaması