

# VİSUAL STUDIO İLE LOCAL MSSQL DATABASE KULLANIMI

## 1. Başlangıç

İlk olarak sistemimizde kurulu olan sqllocaldb örneklerini listeleyelim. Herhangi bir örnek sistemimizde yüklü değilse local veritabanı işlemlerini yapamayız. Bunun için aşağıdaki komut satırını çalıştırın;

```
Komut İstemi
Microsoft Windows [Version 10.0.22621.1555]
(c) Microsoft Corporation. Tüm hakları saklıdır.

C:\Users\Administrator>sqllocaldb.exe i
MSSQLLocalDB

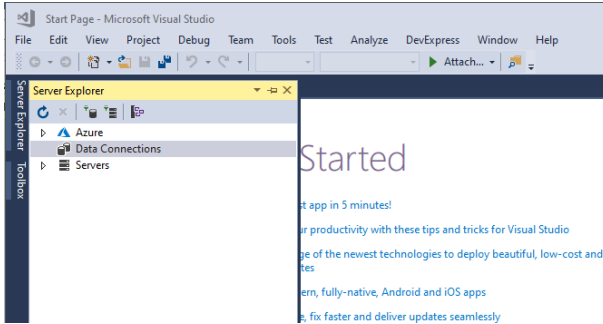
C:\Users\Administrator>
```

Yandaki çıktıya göre sistemimizde kurulu olan örneğin ismi **MSSQLLocalDB**'dir.

Eğer sisteminizde herhangi bir örnek kurulu değil ise “SQL Server Express LocalDB” şeklinde aratarak Microsoft resmî web sitesinden indirin ve kurun. (*Genellikle Visual Studio ile birlikte kurulur.*)

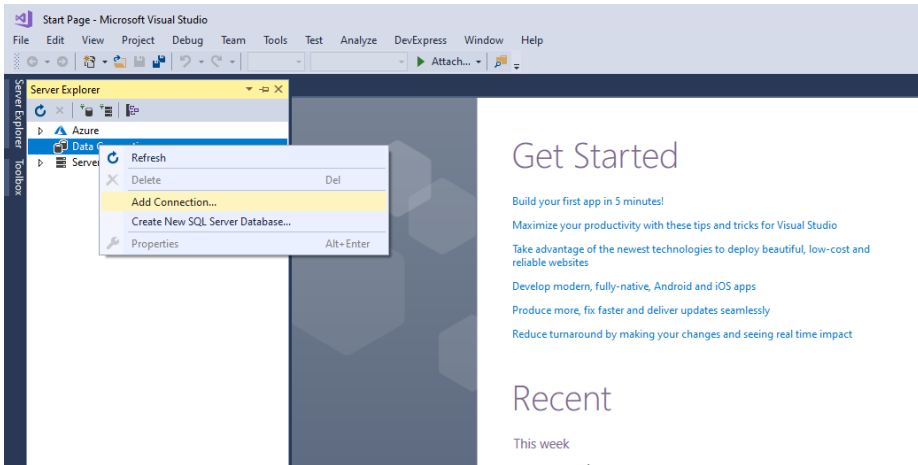
## 2. Veritabanına bağlanma ya da yeni bir veritabanı dosyası oluşturma

Visual Studio uygulamasını başlatın. Herhangi bir proje oluşturmanıza şimdilik gerek yoktur. View->Server Explorer yolunu kullanarak Server Explorer penceresini açın.



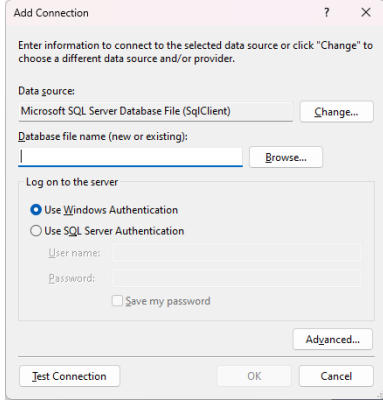
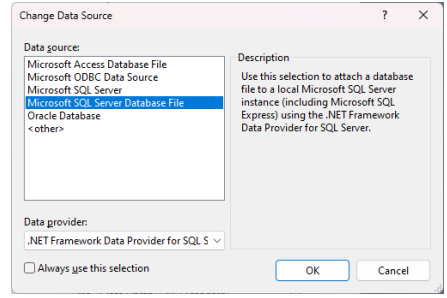
Server Explorer penceresi bir veritabanına bağlanmak ya da yeni bir veritabanı oluşturmak için gerekli arayüzü bize sunar. Bu pencerede **Data Connections** bağlantısını kullanarak bu işlemleri gerçekleştireceğiz.

Data Connections üzerinde sağ tıklayın ve Add Connection komutunu seçin.



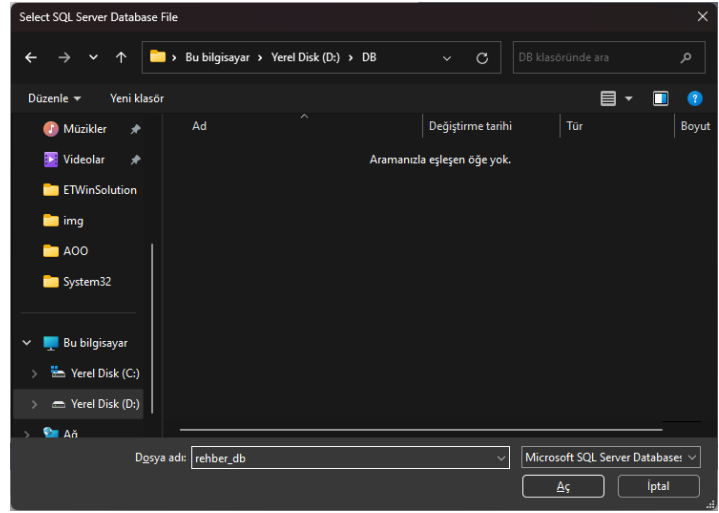
Eğer daha önce bir bağlantı oluşturulmadıysa ilk olarak veri kaynağı seçim penceresi gelecektir.

Farklı veri tabanı kaynakları kullanılabilir. Biz ise burada local veritabanı kullanacağımız için **Microsoft SQL Server Database File** seçeneği ile ilerleyeceğiz.

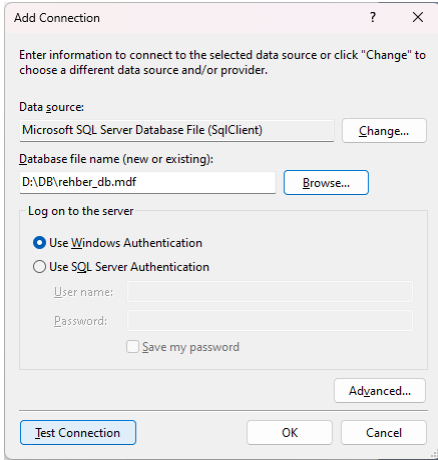


**Microsoft SQL Server Database File** seçeneği ile devam ettiğinizde yandaki pencere gelecektir. Burada birkaç seçim yapmamız gerekecektir. Bir önceki pencereye dönmek için **Change...** butonuna tıklayabilirsiniz.

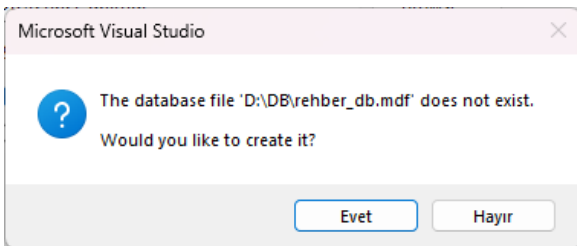
Biz burada veritabanı dosyamızı seçeceğiz ya da yeni bir tane oluşturacağız. **Browse...** butonuna tıklayarak açılan pencereden **veritabanı dosyanız var ise seçiniz ya da şimdi yapacağımız gibi seçtiğiniz bir klasöre yeni bir tane oluşturunuz.**



Yanda görüldüğü gibi Yerel Disk (D:) üzerinde DB isimli klasör altında dosya adı olarak **rehber\_db** adında bir veritabanı oluşturacağız. Aç diyerek devam ediyoruz.

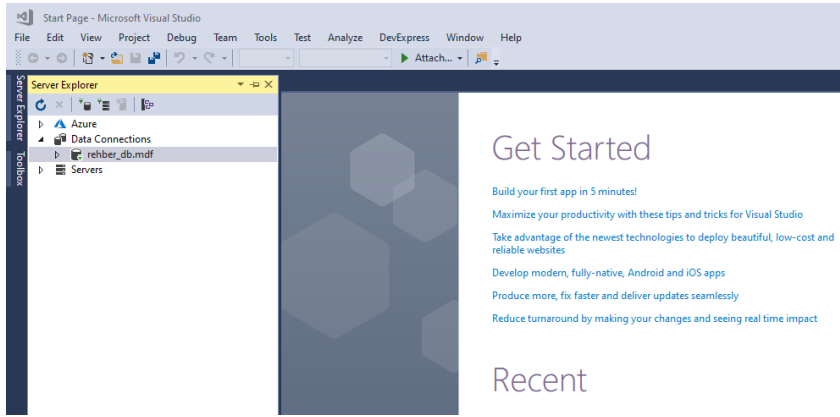


Burada Use Windows Authentication seçeneğini kullanacağız. Bu seçenek ile Yerel Windows kullanıcısı ile herhangi bir parolaya gerek kalmadan veritabanına bağlanabilirsiniz. OK diyerek ilerleyin.



*“Bu veritabanı dosyası yok, oluşturmak ister misiniz?”*  
Şeklinde bir uyarı ile karşılaşacağız. Evet dediğinizde veritabanı dosyasını oluşturacaktır.

**Harika! Artık bir veritabanı dosyanız var ve ona bağlısınız.**



### 3. Veritabanı tablolarını oluşturma

Yapacak olduğunuz iş ile alakalı tabloları oluşturmanız gerekmektedir. Bir veritabanında veriler tablolarda saklanır. Tablolar oluşturulurken veriler olabildiğince küçük parçalara bölünmelidir. Tekrarlayan kayıtlar olmamalı ve bir tabloda birbiri ile doğrudan ilişkili veriler olmalıdır. Örneğin bir e-okul projesini düşünelim. Bu projede varlıklarımız öğrenciler, öğretmenler, sınıf bilgileri ve dersler olabilir. Her bir varlık ayrı bir tablo olarak düşünülmelidir. Hatta bir varlığa ait uzun veriler ayrı bir tabloda saklanmalıdır. Örneğin öğrencilere ait veli bilgileri öğrenciler tablosunda değil ayrı olarak veli bilgileri tablosunda saklanmalıdır. Bir tablonun çok uzun olması veri sorgulamayı yavaşlatacaktır. Tabloların mükemmel şekilde tasarımı veritabanı normalizasyonu olarak adlandırılır. Bu ayrı bir konu olduğu için burada üzerinde durulmayacaktır.

Her tablo satır ve sütunlardan oluşmaktadır. Her sütun veri alanlarını içerir ve her satır ise kayıtları içerir. Yani bir sütundaki tüm veriler aynı türden farklı kayıtları içerir fakat her satır farklı bir kayıta aittir.

	Id	No	Ad	Soyad	Cinsiyet
1. Kayıt →	1	1	Ahmet	KURAL	1
	2	2	Ayza	KURAL	0
	NULL	NULL	NULL	NULL	NULL

Bir veritabanı tablosunu tasarlarken alanların isimleri, veri türleri ve özellikleri belirlenir.

#### a. Alanlara İsim Verilmesi

Alanlara isim verilirken ortak bir yaklaşım kullanmak ve tıpkı değişken isimlendirme de yaptığımız gibi anlamlı ve bazı kuralları olan isimlendirme yapmalıyız. Her kuruluşun isimlendirme kuralı farklı olabilir. Biz burada Upper Camel Case notasyonunu kullanacağız. Kelimelerin ilk harfleri büyük harf ve tüm kelimeler bitişik yazılır. Örneğin; **OğrenciNo**, **BolumKodu**, **StokSiparisAdedi** gibi.

- 1- Alanlara isim verirken Upper Camel Case notasyonunu kullananın.
- 2- Alanlara isim verirken İngiliz alfabesini kullanın. (ç, Ç, ğ, Ğ, ş, Ş, ü, Ü, ö, Ö, ı, İ ve özel karakter kullanmayın)
- 3- Alanlara isim verirken rakam ile başlamayın.
- 4- Alanlara isim verirken C# dilinde bulunan anahtar kelimeleri kullanmamanız karışıklığın önüne geçecektir.

#### b. Alanların Veri Türlerinin Belirlenmesi

Alanlarda saklanacak veriler farklı türlerde olabilir. Bir sayı, metinsel bir veri ya da tarih gibi. Bunun için önceden hangi türden verinin saklanacağı belirtilmelidir. SQL dili birçok veri türünü sağlar, fakat burada hepsine yer verilmeyecektir. En çok kullanılacak olan veri türleri aşağıda verilmiştir.

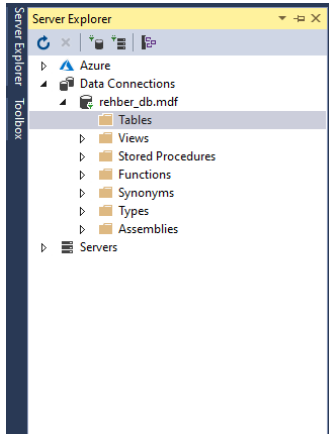
int	Tam sayı veriler için kullanılır.
float	Ondaklıkl sayılar için kullanılır.
nvarchar(N)	Metinsel Unicode veriler için değışken uzunluklu veriler için kullanılır. Burada N maksimum karakter uzunluğudur.
bit	Mantıksal veriler için kullanılır. (True = 1 ya da False = 0)
datetime	Tarih ve saat verisini saklamak için kullanılır.

**c. Alanların Özelliklerinin Belirlenmesi**

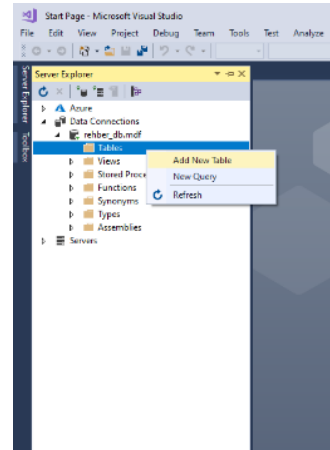
Alanların özellikleri aşağıdaki gibi olabilir.

Birincil Anahtar (Primary Key)	Anahtar alan bir tablonun kayıtlarını tekilleştiren bir özelliktir. Aynı anahtar değeriine sahip iki kayıt olamaz. Genellikle tam sayı veri türü seçilerek uygulanır. Anahtar alanlar boş geçilemez.
Otomatik Sayı (Identity)	Tam sayı bir alanın otomatik olarak sql tarafından değeri verilmesi için kullanılır. Genellikle anahtar alanlar için kullanılır.
Boş Geçilebilir (Allow Null)	Bir alanın boş geçilip geçilemeyeceğini belirlemek için kullanılır. Önemli alanlar boş geçilmemelidir. Mesela bir öğrencinin numarası boş geçilmemeli fakat telefon numarası boş geçilebilir olmalıdır.

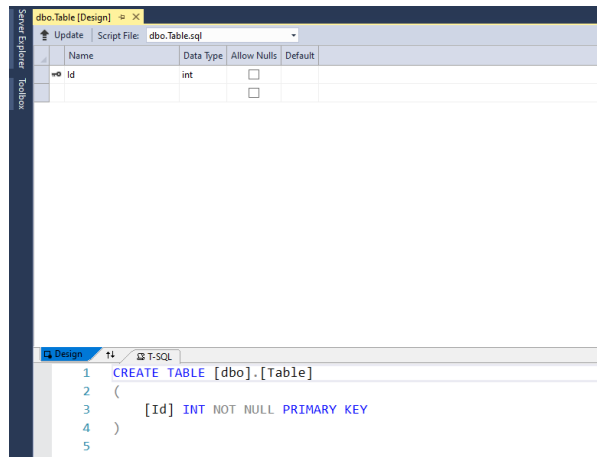
Şimdi isterseniz daha önce oluşturmuş olduğumuz rehber\_db isimli veritabanı için örnek bir tablo oluşturalım. Bunun için öncelikle Visual Studio Server Explorer penceresinde bulunan rehber\_db dalını genişletelim ve Tables klasörünü bulalım. Tables klasörünün altında mevcut tablolar görüntülenir.



Burada görüldüğü gibi herhangi bir tablo şu anda bulunmuyor. Şimdi Tables klasörü üzerinde sağ tıklayın ve Add New Table komutunu seçin.

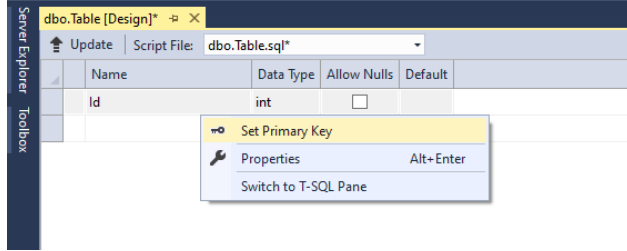


Bu aşamadan sonra karşımıza tablo oluşturmak için editör gelecektir. Bu editör yardımıyla tablonun alanları belirlenebilir.



Visual studio içinde bulunan bu editör varsayılan olarak Id adında bir alan oluşturur. Eğer böyle bir alan yok ise siz oluşturun. Bu alan tablonun birincil anahtarı olacaktır. Birincil anahtarı olmayan tablolar birbirleriyle ilişkilendirilemez. Dikkat ederseniz editörün üst kısmında belirlediğiniz alanların oluşturulması için gerekli SQL kodları editörün alt kısmında otomatik olarak oluşturulmaktadır. Siz yalnızca yukarıdan değişiklik yaparak devam edin.

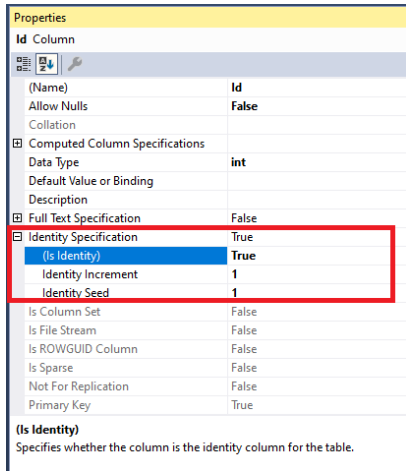
Eğer Id alanı otomatik olarak oluşturulmadıysa **Name** altına **Id** yazın ve **Data Type** olarak **int** seçin. Bu satırda sağ tıklayın ve menüden Set Primary Key seçin.



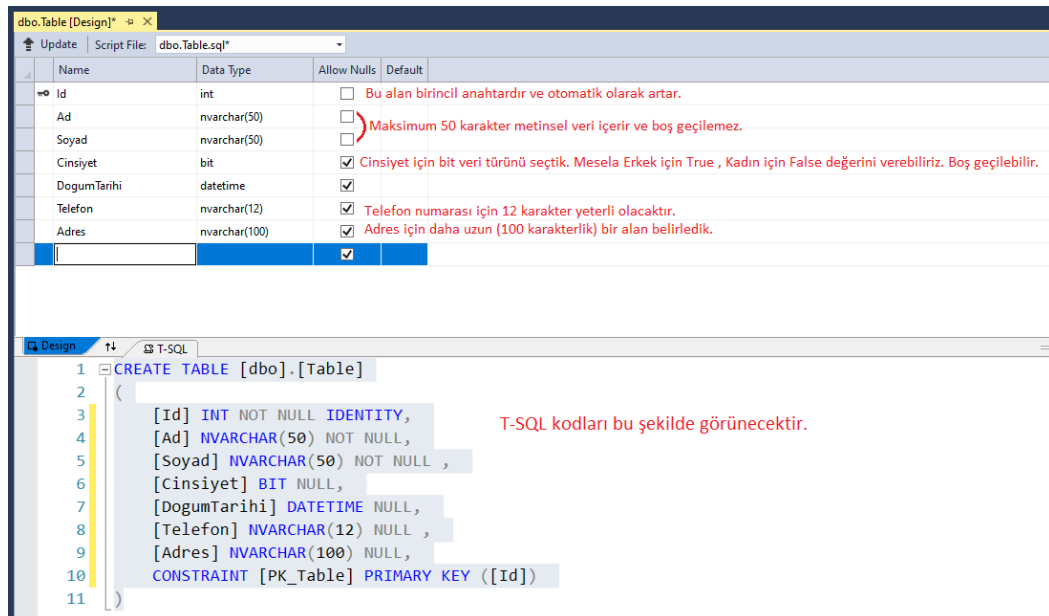
Sonrasında ise bu alanın sol tarafında bir anahtar simgesi gelecektir. Artık bu alan birincil anahtardır ve boş geçilemez.

Şimdi ise bu alanı **otomatik sayı** yapalım. Bunun için bu alanın properties penceresinden aşağıda kırmızı çerçeve içine alınmış özelliğini değiştirin.

Bu alan otomatik sayı olacak, 1'er artarak gidecek ve 1'den başlayacaktır.

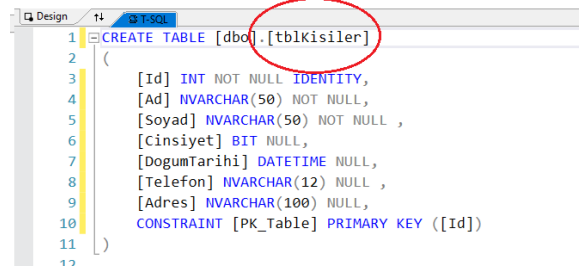


**Id adında bir otomatik sayı anahtar alan bizim tüm tablolarımızda ortak alandır.** Şimdi artık tablomuza ait diğer alanları belirleyebiliriz. Bu noktada tablonun ne ile ilgili olduğu alanlarımızı belirlememize yardımcı olacaktır. Örneğin bir rehber yapacağımız için kişiler ve iletişim bilgileri olabilir. Kişinin adı, soyadı, cinsiyeti, doğum tarihi, telefon numarası ve adresi bizim istediğimiz alanlar olacaktır. Aşağıdaki gibi alanları belirleyin.



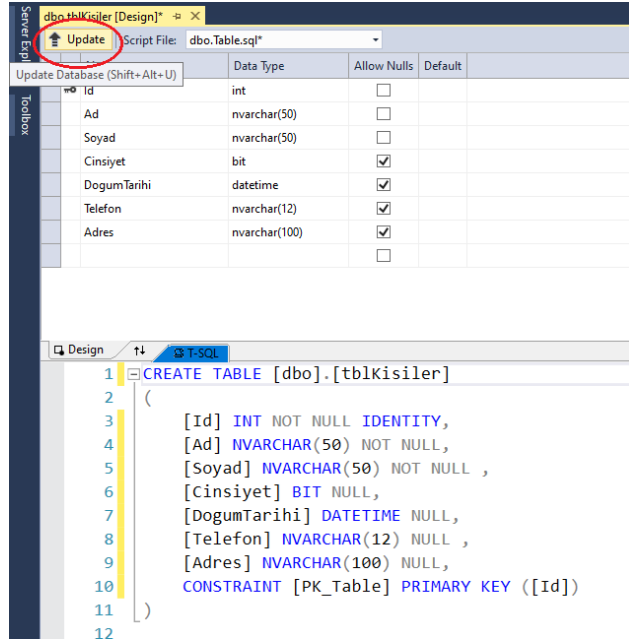
T-SQL kodları bu şekilde görünecektir.

Şimdi sıra geldi tablomuza bir isim vermeye. Aşağıda bulunan T-SQL bölümünden Table yazan yere tablonuza kendi vereceğiniz bir isim yazın. Biz genellikle tablolarımıza isim verirken macar notasyonunu kullanıyoruz ve ismin başına **tbl** ön ekini koyuyoruz. Buradaki örnekte bulunan tablonun ismi **tblKisiler** olacak.

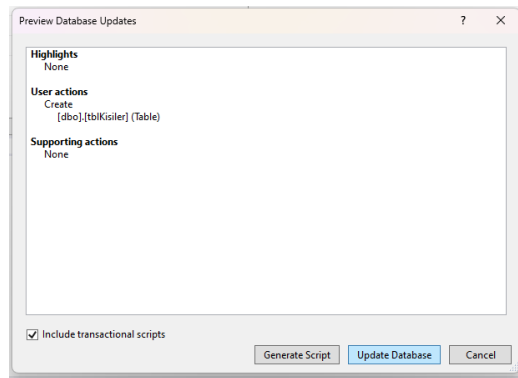


```
1 CREATE TABLE [dbo].[tblKisiler]
2 (
3     [Id] INT NOT NULL IDENTITY,
4     [Ad] NVARCHAR(50) NOT NULL,
5     [Soyad] NVARCHAR(50) NOT NULL ,
6     [Cinsiyet] BIT NULL,
7     [DogumTarihi] DATETIME NULL,
8     [Telefon] NVARCHAR(12) NULL ,
9     [Adres] NVARCHAR(100) NULL,
10    CONSTRAINT [PK_Table] PRIMARY KEY ([Id])
11 )
12
```

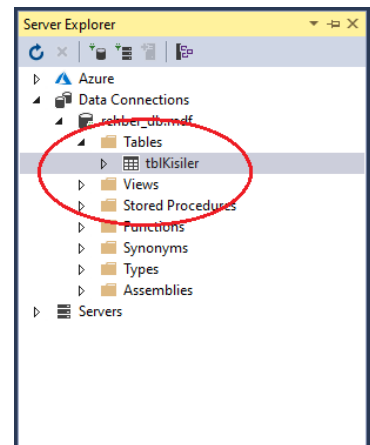
Tabloyu kayıt etmek için Editörün üstünde bulunan **Update** butonuna tıklayın.



Karşınıza gelecek olan ekrandan Update Database butonuna tıklayın.

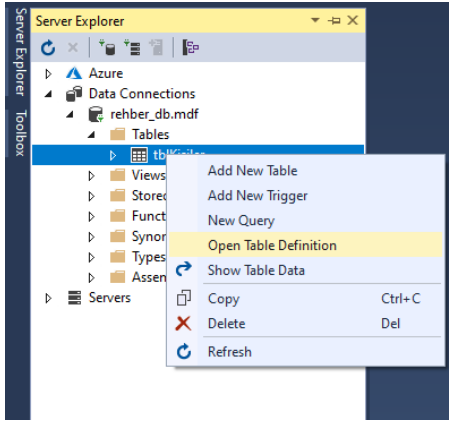


Server Explorer'da bulunan Tables klasörünü yenileyin ve tablonuzun geldiğini göreceksiniz.



#### 4. Veritabanındaki bir tablonun tasarımını deęiřtirme

Bunun için tablo üzerinde saę tıklayın ve **Open Table Definition** komutunu seçin.



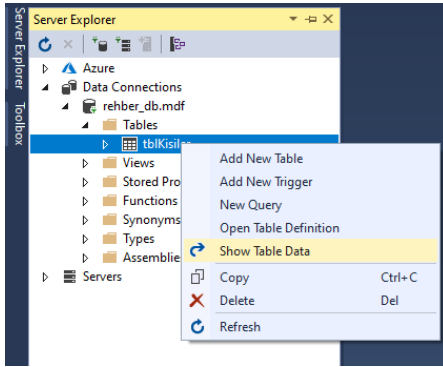
Bu komutu seçtikten sonra tablo tasarım ekranı karşınıza gelecektir.

Burada dikkat edilmesi gereken nokta tablodaki alanlarda kayıt edilmiş veriler var ise deęişiklik yaptığınızda uyumsuzluk olabileceğidir. Bu durumda tablo güncellenemez. Örneğin bir alanda metinsel veriler var ise bu alanın veri türü sayı olarak deęiřtirilemez ya da [Allow Null] seçeneęi seçili deęilse ve boş kayıtlar var ise bu alanın [Allow Null] özellięi deęiřtirilemez.

**Burada gerekli deęişiklikleri yapın ve tablonuzu Update butonunu kullanarak güncelleyin.**

#### 5. Tabloya kayıt girme

Bunun için tablo üzerinde saę tıklayın ve **Show Table Data** komutunu seçin.

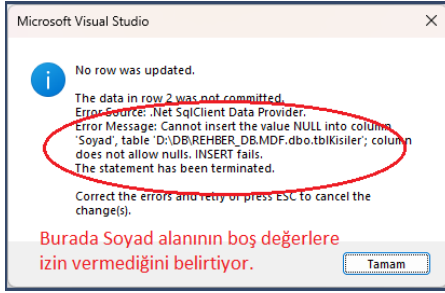


Bu komut seçildiğinde veri giriři için bir editör açılacaktır. İsteddiğiniz kayıtları uygun veri türleri girerek oluřturun. Bir satırdan ayrıldıktan sonra veriler tabloya eklenir. Yani kaydet demenize gerek yoktur.

Id	Ad	Soyad	Cinsiyet	DogumTarihi	Telefon	Adres
1	Kemal	CAN	True	1.01.2002 00:00:...	5991234567	Torbalı
NULL	NULL	NULL	NULL	NULL	NULL	NULL

Yukarıda resimde gördüğünüz gibi bir adet kayıt eklenmiştir. Doğru veriler girilmediğinde ünlem işareti ile uyarı verecektir ve düzeltilmenizi isteyecektir. Mesela Ad ve Soyad alanı boş geçilemez.

Id	Ad	Soyad	Cinsiyet	DogumTarihi	Telefon	Adres
1	Kemal	CAN	True	1.01.2002 00:00:...	5991234567	Torbalı
NULL	Sedef	NULL	False	14.02.1996 00:00:00	NULL	NULL
NULL	NULL	NULL	NULL	NULL	NULL	NULL

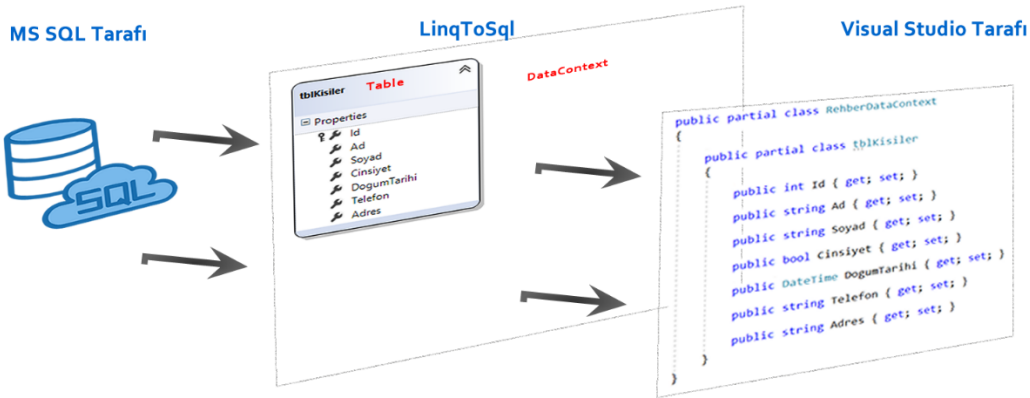


Böyle bir hatadan sonra Id sıranız değişebilir önemsemeyin.

Id	Ad	Soyad
1	Kemal	CAN
3	Sedef	SU
NULL	NULL	NULL

## 6. LinqToSql Framework İle Veritabanına Bağlanma

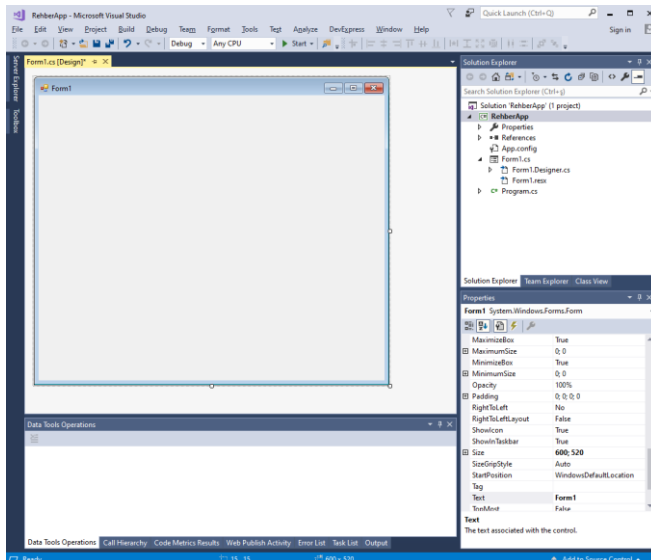
Visual Studio tarafında bir veritabanına bağlanmak ve üzerinde işlem yapmak için bize bir çatı sağlayan LinqToSql Framework burada açıklanacaktır. Normalde veritabanı işlemleri T-SQL dili ile yapılmaktadır. Fakat bizi T-SQL kullanımından kurtaran ve kayıtlar üzerinde ön bellekleme yapabilen kullanışlı frameworkler vardır. Bunlar biri olan LinqToSql MS Sql veritabanlarına bağlanmak için Microsoft tarafından geliştirilmiştir. Tüm frameworklerde veritabanındaki bir tabloyu kodlama ortamında bir sınıf olarak temsil etme ve işlemleri DataContext denilen bir sınıf üzerinden veritabanına yazmayı ve veritabanından okumayı gerçekleştirme temel fikirdir.



Veri tabanının LinqToSql ile Visual Studio tarafında Sınıflar ile Temsil Edilmesi

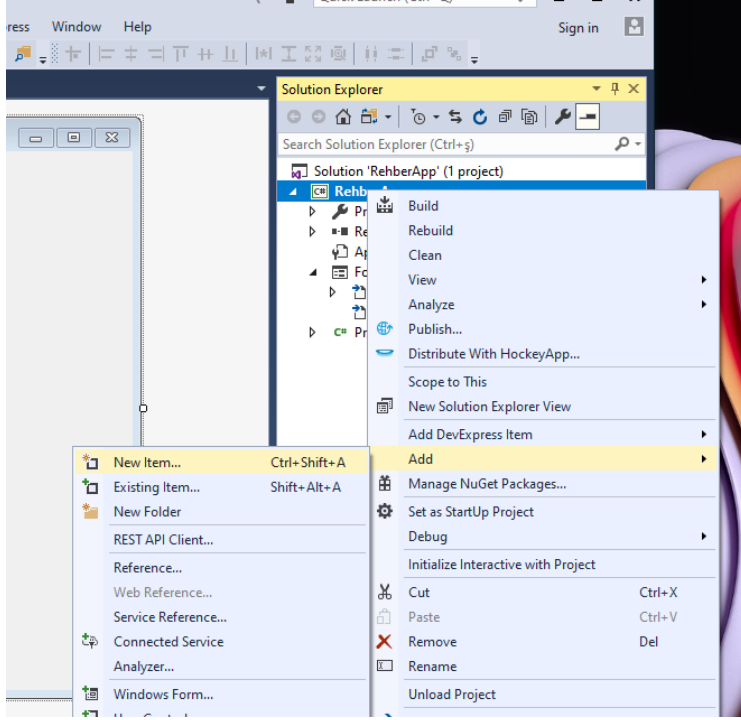
## 7. LinqToSql Sınıflarını Visual Studio Projesine Ekleme

Bu aşamada yeni bir Windows Forms Application oluşturalım. Projemizin adı **RehberApp** olabilir.

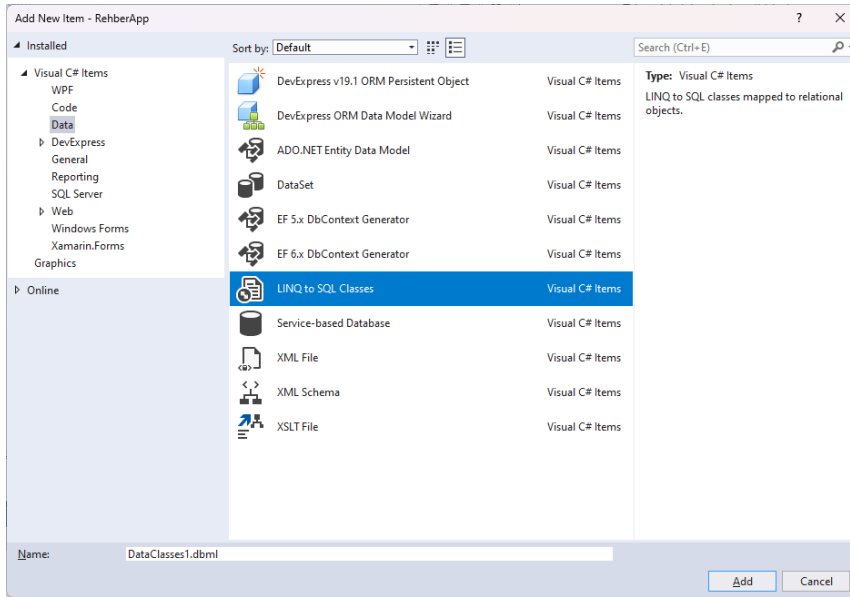




Şimdi Solution Explorer penceresinde proje adı üzerinde sağ tıklayalım ve **Add→New Item** komutunu seçelim.

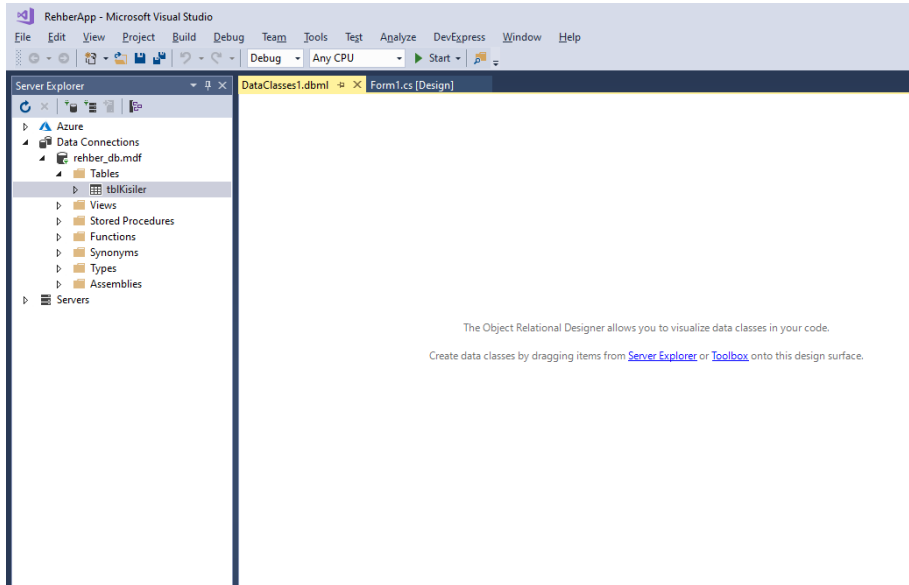


Sonrasında açılan pencereden sol taraftan **Data** bölümünü seçelim ve **Linq to SQL Classes** öğesini projemize ekleyelim.

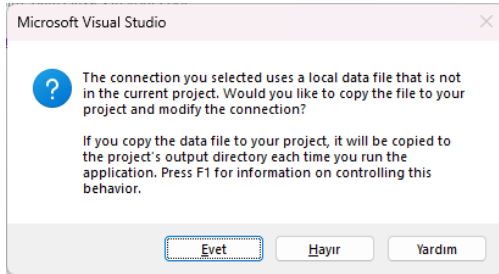


Bu işlemten sonra DataClasses1.dbml isimli öğe projemize eklenecektir. Bu öğenin ismini isterseniz değiştirebilirsiniz ama biz olduğu gibi bıraktık.

Şimdide Server Explorer penceresinden daha önce eklemiş olduğumuz rehber\_db bağlantısını açalım ve Tables klasörü altında bulunan tabloları görelim. Eğer veritabanını görmüyorsanız önceki adımlara göz atarak bağlantıyı ekleyebilirsiniz.

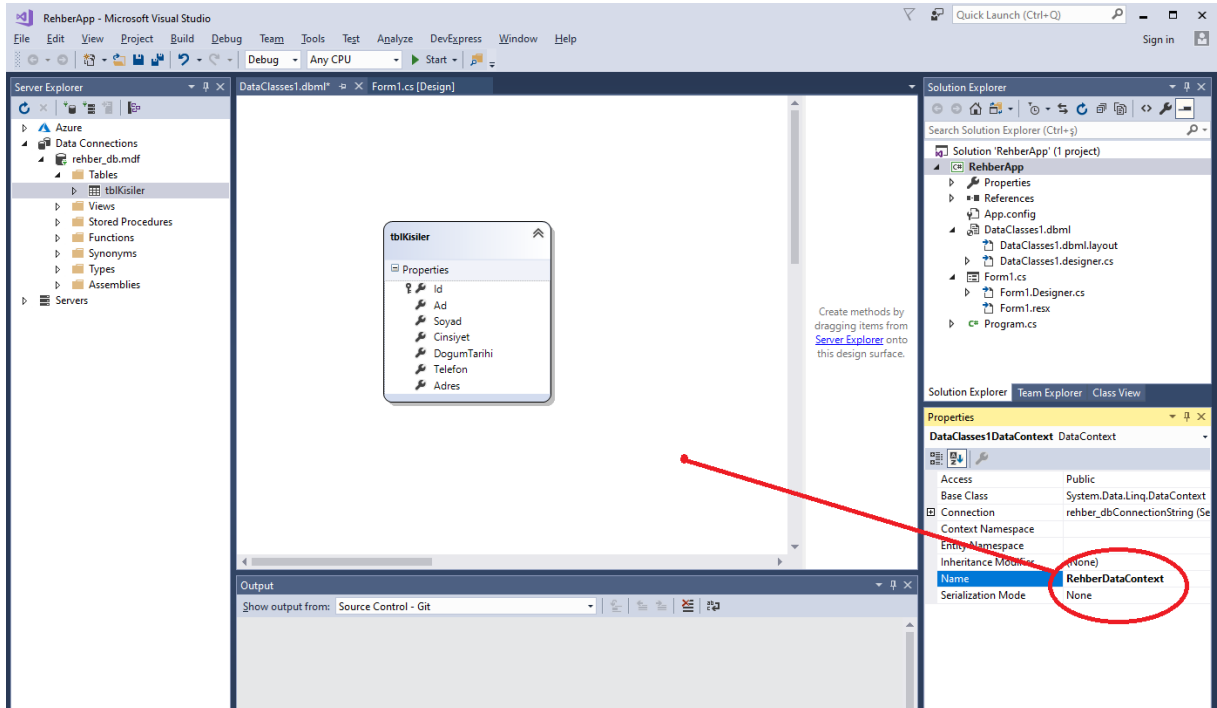


Server Explorer da gördüğünüz tabloları sürükleyerek DataClasses1.dbml penceresine bırakalım.



Yandaki şekilde bir uyarı ile karşılaşırsanız **Hayır** butonuna tıklayın. Bu veritabanı dosyasını proje klasörüne kopyalamak isteyip istemediğimizi soruyor. **Evet** tıkladığınızda dosyanın bir kopyasını proje klasörüne kopyalar ve o dosya üzerinden işlem yapar. Biz bunu istemiyoruz

Artık tablomuz eklendi. **DataClasses1.dbml** üzerinde boş bir yere tıklayın ve **Properties** penceresinden Name alanını **RehberDataContext** olarak değiştirin.



Buraya eklemiş olduğumuz tblKisiler bizim tablomuzu ve RehberDataContext ise veritabanımızı temsil ediyor. Artık verilere ulaşmak için her şey hazır gibi görünüyor. **Kaydet** butonuna tıklayın ve değişiklikleri kayıt edin.

## 8. Verileri Göstermek için Formu Tasarlama

Bu örnekte verileri göstereceğimiz formu aşağıdaki gibi tasarlayın. Resimde kontrollere verilen isimler kırmızı renk ile belirtilmiştir. Kontrollere ait özellikle değiştirilmesi gereken özellikler ise siyah kutu içerisinde belirtilmiştir.

Rehber Uygulaması

# Rehber Uygulamam

Kişilerim Kişi Detayları

lbKisiler

Ad txtAd

Soyad txtSoyad

Cinsiyet cbCinsiyet

Doğum Tarihi 8.05.2023 dtDogumTarihi

Telefon txtTel

Adres txtAdres

btnYeni Yeni

btnSil Sil

btnGuncelle Güncelle

DropDownStyle = DropDownList  
Items = [Erkek, Kadın]

Format = Short

Multiline = True

Şimdi kod ekranına geçin ve aşağıdaki gibi formun global alanında **db** isimli (*isim önemli değil*) veri tabanı bağlantısını oluşturun.

```
3 references
public partial class Form1 : Form
{
    RehberDataContext db = new RehberDataContext();

    1 reference
    public Form1()
    {
        InitializeComponent();
    }
}
```

Hatırlarsanız **DataClasses1.dbml** dosyasını eklediğimizde **RehberDataContext** ismini vermiştik. İşte bu bizim veri tabanına ulaşmamızı sağlayacak nesne olacak.

Şimdide veritabanından kayıtları çekelim ve listBox içerisinde gösterelim.

```
List<tblKisiler> kisiler; veritabanından aldığım kişileri bu listede
geçici olarak saklayacağım.

1 reference
public Form1()
{
    InitializeComponent();

    KayitlariGoster(); form ilk açıldığında kayıtları göstereceğiz
    bu metodu çağır

1 reference
void KayitlariGoster()
{
    kisiler = db.tblKisilers.ToList(); bu satırda veritabanından tblKisilers
    tablosundan tüm kayıtları getirir.

    lbKisiler.Items.Clear(); liste kutusunu önce temizlemek istiyorum.

    foreach(var kisi in kisiler) burada tüm kişileri döngü ile dolaş ve her
    birinin Ad ve Soyadını liste kutusuna ekle
}
```

Hatırlarsanız **DataClasses1.dbml** dosyası içerisinde bizim tablomuzun adı **tblKisiler** olarak belirlenmişti. LinqToSql tabloların sonuna bir (s) harfi ekler. Sonuç olarak tablo **DataContext** içerisinde **tblKisilers** olarak yer alır.

Şimdi de liste kutusundan bir kayıt tıklandığında kişi detaylarında gösterelim. Bunun için tıklanan kişiyi formun global alanında saklayacağım. Çünkü Sil ya da Güncelle butonlarına tıklandığında ekranda kim olduğunu bilmek istiyorum.

```
3 references
public partial class Form1 : Form
{
    RehberDataContext db = new RehberDataContext();
    List<tblKisiler> kisiler;
    tblKisiler goruntulenenKisi;
```

Liste kutusunun **SelectedIndexChanged** olayına kod yazarak seçili kişiyi detay alanında gösterelim.

```
private void lbKisiler_SelectedIndexChanged(object sender, EventArgs e)
{
    if(lbKisiler.SelectedIndex>=0) seçili kişi var ise gösterelim
    {
        goruntulenenKisi = kisiler[lbKisiler.SelectedIndex]; seçili indexte bulunan kişiyi al

        txtAd.Text = goruntulenenKisi.Ad; buradan sonra tüm alanlarda seçili olan kişinin
        txtSoyad.Text = goruntulenenKisi.Soyad; bilgilerini gösteriyorum.

        if (goruntulenenKisi.Cinsiyet.GetValueOrDefault()) GetValueOrDefault, Allow Null
            cbCinsiyet.SelectedIndex = 0; türlerde veri yok ise varsayılan
        else cbCinsiyet.SelectedIndex = 1; değerini okuyabilirsiniz. Zaten veri
        dtDogumTarihi.Value = goruntulenenKisi.DogumTarihi.GetValueOrDefault(); var ise onu getirecektir

        txtTel.Text = goruntulenenKisi.Telefon;
        txtAdres.Text = goruntulenenKisi.Adres;
    }
    else
    {
        goruntulenenKisi = null;
        txtAd.Text = ""; Bir kişi bulunamadı ise
        txtSoyad.Text = ""; tüm alanları temizle
        cbCinsiyet.SelectedIndex = 0;
        dtDogumTarihi.Value = DateTime.Now;
        txtTel.Text = "";
        txtAdres.Text = "";
    }
}
```


Artık gördüğümüz gibi veritabanından kayıtları getiriyorum ve detaylarda gösterebiliyorum.

Şimdi de detayda bulunan kişiyi güncelleyelim. Hatırlarsanız görüntülenen kişiyi formun global alanında saklamıştım. Şimdi onun alanlarını değiştireceğim ve veritabanıma kayıt edeceğim.

```
private void btnGuncelle_Click(object sender, EventArgs e)
{
    if(goruntulenenKisi!=null) görüntülenen bir kişi var ise işlem yap
    {
        goruntulenenKisi.Ad = txtAd.Text;           burada formdaki tüm
        goruntulenenKisi.Soyad = txtSoyad.Text;      verileri kişiye aktarıyorum
        goruntulenenKisi.Cinsiyet = cbCinsiyet.SelectedIndex == 0;
        goruntulenenKisi.DogumTarihi = dtDogumTarihi.Value;
        goruntulenenKisi.Telefon = txtTel.Text;
        goruntulenenKisi.Adres = txtAdres.Text;
    }

    db.SubmitChanges(); değişikliklerin geçerli olabilmesi için
                        veritabanına kayıt ediyorum
}
```

Peki görüntülenen kişi yok ise ne yapacağız! Evet. Görüntülenen kişi yok ise yeni bir kayıt olarak veritabanına ekleyebiliriz. Bunun için yukardaki koduma biraz ekleme yapıyorum.

```
private void btnGuncelle_Click(object sender, EventArgs e)
{
    int yenidenGoster;  bunu kullanmak zorundaydım. Çünkü KayitlariGoster
    if(goruntulenenKisi!=null) metodunu çağırınca liste kutusu temizleniyor ve seçili olan
    {                               elemanda kayboluyordu.
        goruntulenenKisi.Ad = txtAd.Text;
        goruntulenenKisi.Soyad = txtSoyad.Text;
        goruntulenenKisi.Cinsiyet = cbCinsiyet.SelectedIndex == 0;
        goruntulenenKisi.DogumTarihi = dtDogumTarihi.Value;
        goruntulenenKisi.Telefon = txtTel.Text;
        goruntulenenKisi.Adres = txtAdres.Text;

        yenidenGoster = lbKisiler.SelectedIndex; güncelleme yaptıysam aynı kayıt
    }                                           yenilemeden sonra görüntülensin
    else
    {
        tblKisiler yeniKisi = new tblKisiler(); burada yeni bir kişi oluştuyorum
        yeniKisi.Ad = txtAd.Text;
        yeniKisi.Soyad = txtSoyad.Text;
        yeniKisi.Cinsiyet = cbCinsiyet.SelectedIndex == 0; oluşturduğum kişinin tüm bilgilerini formdan alıyorum
        yeniKisi.DogumTarihi = dtDogumTarihi.Value; ve yeniKisi nesneme aktarıyorum.
        yeniKisi.Telefon = txtTel.Text;
        yeniKisi.Adres = txtAdres.Text;

        db.tblKisilers.InsertOnSubmit(yeniKisi); bu yeni oluşturduğum kişiyi veritabanı
        yeniGoster = lbKisiler.Items.Count; yeni kayıt eklediysen muhtemelen bu sona eklenecektir ve
        db.SubmitChanges(); değişiklikleri veritabanına kayıt et
    }

    KayitlariGoster(); kayıtları yenile
    lbKisiler.SelectedIndex = yenidenGoster; ve liste kutusunda yukarıda işlem
}                                           göre kayıtları seçili yap
```

Şu anda uygulamam çok güzel çalışıyor. Fakat liste kutusundan bir eleman seçilince yeni kayıt ekleyemiyorum. Bunun için görüntülenen kaydı temizlemek istiyorum. Yeni butonuna birkaç satır kod yazacağım.

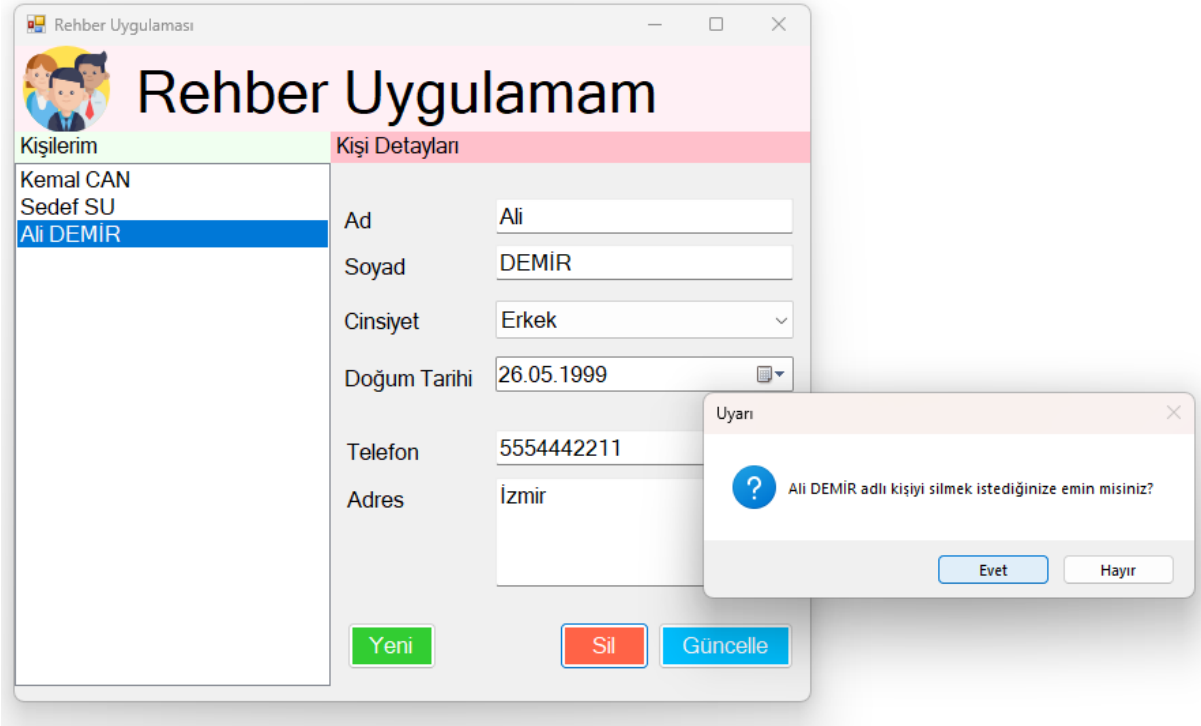
```
private void btnYeni_Click(object sender, EventArgs e)
{
    lbKisiler.SelectedIndex = -1; seçili elemanı kaldır. Dolayısıyla
    txtAd.Focus(); goruntulenenKisi null olacaktır.
} burada veri girişi için txtAd metin kutusuna
imleçi taşı
```

Evet şimdi sırada ne var? Sil butonuna uygun kodları yazarak artık görmek istemediğim kişileri silmem gerekiyor. Ama önce emin olup olmadığını soralım. Haydi yapalım.

```
private void btnSil_Click(object sender, EventArgs e)
{
    if(goruntulenenKisi!=null) ancak seçili kişi var ise silebilirim
    {
        DialogResult cevap = MessageBox.Show(goruntulenenKisi.Ad + " " + goruntulenenKisi.Soyad +
        " adlı kişiyi silmek istediğinize emin misiniz?", "Uyarı",
        MessageBoxButtons.YesNo, MessageBoxIcon.Question); burada seçili kişinin
        adını ve soyadını içeren
        bir uyarı mesajı
        gösteriyorum.

        if (cevap == DialogResult.Yes) soruya Yes butonuna tıklayarak
        cevap verdiyse
        {
            db.tblKisilers.DeleteOnSubmit(goruntulenenKisi);
            db.SubmitChanges(); kişiyi tablodan sil ve
            veritabanını kaydet

            KayitlariGoster();
            lbKisiler.SelectedIndex = 0; listeyi yeniledikten
            sonra ilk kaydı aktif et
        }
    }
    else tabiki seçili bir kişi yok ise böyle bir mesaj vermek uygun olabilir
    {
        MessageBox.Show("Silmek için önce bir kişi seçmelisiniz!", "Uyarı",
        MessageBoxButtons.OK, MessageBoxIcon.Error);
    }
}
```



**Tebrikler! Veritabanına bağlanabilen bir rehber uygulaması yaptınız.**