SQL Server ve MySQL Server Destekli Yazılım Geliştirme

Kurumsal olarak geliştirilen yazılımların özelliklerine bakıldığında hemen hemen hepsinde en az iki veri tabanı sistemini desteklediği belirtilmiştir. Bu özellik ilk bakışta her ne kadar geliştirilmesi zor bir özellik olarak görünse de aslında sadece temel kalıtım özellikleri kullanılarak geliştirilebilir. Bu yazımızda biz de herhangi bir harici kütüphane kullanmadan bu işlemi nasıl yaptığımızı inceleyeceğiz.

Bu işle için öncelikle uygulamamızın destekleyeceği veri tabanı sistemlerinde eş yapılı bir veri tabanı yapısı kurulur. Biz örneğimizi tek tablo üzerinden götüreceğiz.

MySQL Tablo Yapısı:

```
CREATE TABLE `kisiler` (
  `id` int NOT NULL AUTO_INCREMENT,
  `isim` varchar(100) NOT NULL,
  `soyisim` varchar(100) NOT NULL,
  `telefon` char(10) NOT NULL,
  PRIMARY KEY (`id`)
)
```

SQL Server Tablo Yapısı

```
CREATE TABLE Kisiler2(
Id int PRIMARY KEY IDENTITY(1,1) NOT NULL,
Isim nvarchar(100) NOT NULL,
```

```
Soyisim nvarchar(100) NOT NULL,
Telefon char(10) NOT NULL
)
```

Veri tabanı yönetim sistemi üzerinde tablolar oluşturulduktan sonra bu tablolara karşılık gelen C# sınıfları yazılır. Bu sınıfların yazılasının sebebi veri tabanı yönetim sistemlerimiz ile uygulamamızı ortak bir yapı altında konuşturabilmektir.

Tablolara Karşılık C# Sınıfı

```
 class Kisi
    {
        public int Id { get; set; }
        public string Isim { get; set; }
        public string Soyisim { get; set; }
        public string Telefon { get; set; }
    }
```

Sınıfımızı da oluşturduktan sonra artık veri tabanı işlemleri için yapımızı kurmaya başlayabiliriz. Yapımızın temelinde interface (ara yüz) kavramı olduğundan burada interface kavramına değinmek lazım.

Interface

Interface tanımlanırken herhangi bir sınıf tanımlanır gibi tanımlanır ancak türüne class yerine interface yazılır.

Interface içerisinde neler bulunabilir neler bulunamaz:

- Interface içerisinde propertyler bulunabilir.
- Interface içerisinde tanımlanan metotların sadece başlık bildirimi yapılır, gövdesi tanımlanmaz.
- Interface içerisinde tanımlanan property ve metotlar public olduğundan ayrıca erişim belirteci belirtilmez.

Bu bilgilerden sonra şimdi tanımlanan tablomuz üzerinde yapılacak olan işlemlerin metotlarını barındıran bir interface tanımlayalım.

```
interface IKisi
{
   int KisiEkle(Kisi yeniKisi);
   List<Kisi> KisileriGetir();
}
```

İncelememizi iki metot üzerinden ilerleteceğiz. Tanımlanan interface içerisinde de göründüğü üzere metotların sadece başlık bilgileri girilmiş ancak gövdeleri tanımlanmamış. Gövdeleri yani bu metotların yapacakları işleri bu interfaceden türetilen sınıflar yapacak.

Interface tanımı da yaptığımıza göre şimdi de sıra SQL Server ve MySQL Server işlemlerini yapacak sınıfları tanımlamaya.

SQL Server Veri Tabanı İşlemleri

```
    class MsSqlKisiIslemleri : IKisi
    {
        public int KisiEkle(Kisi yeniKisi)
        {
                    SqlConnection sqlConnection = new
```

```
SqlConnection("Data        Source=.\\egitim;Initial
Catalog=OgretmenBilgi;Integrated Security=True");
             SqlCommand sqlCommand = new SqlCommand("INSERT
INTO Kisiler([Isim], [Soyisim], [Telefon]) values(@i, @s,
@t)", sqlConnection);

                sqlCommand.Parameters.AddWithValue("@i",
yeniKisi.Isim);
                sqlCommand.Parameters.AddWithValue("@s",
yeniKisi.Soyisim);
                sqlCommand.Parameters.AddWithValue("@t",
yeniKisi.Telefon);


            sqlConnection.Open();
            int i = sqlCommand.ExecuteNonQuery();
            sqlConnection.Close();

            return i;
        }

        public List<Kisi> KisileriGetir()
        {
            List<Kisi> kisiler  = new List<Kisi>();
                    SqlConnection sqlConnection = new
SqlConnection("Data        Source=.\\egitim;Initial
Catalog=OgretmenBilgi;Integrated Security=True");
            SqlCommand sqlCommand = new SqlCommand("select Id,
Isim, Soyisim, Telefon from Kisiler", sqlConnection);
            sqlConnection.Open();
                        SqlDataReader sqlDataReader =
sqlCommand.ExecuteReader();

            while (sqlDataReader.Read())
            {
                Kisi kisi = new Kisi();
                kisi.Id = sqlDataReader.GetInt32(0);
                kisi.Isim = sqlDataReader.GetString(1);
                kisi.Soyisim = sqlDataReader.GetString(2);
                kisi.Telefon = sqlDataReader.GetString(3);
                kisiler.Add(kisi);
```

```
            }


            sqlConnection.Close();
            return kisiler;
        }
    }
```

MySQL Server Veri Tabanı İşlemleri

```
    class MySqlKisiIslemleri : IKisi
    {
        public int KisiEkle(Kisi yeniKisi)
        {
                  MySqlConnection sqlConnection = new
MySqlConnection("server=localhost;user
id=root;database=egitim;pwd=123456qaZ.");
            MySqlCommand sqlCommand = new MySqlCommand("INSERT
INTO Kisiler(isim, soyisim, telefon) values(@i, @s, @t)",
sqlConnection);

                sqlCommand.Parameters.AddWithValue("@i",
yeniKisi.Isim);
                sqlCommand.Parameters.AddWithValue("@s",
yeniKisi.Soyisim);
                sqlCommand.Parameters.AddWithValue("@t",
yeniKisi.Telefon);


            sqlConnection.Open();
            int i = sqlCommand.ExecuteNonQuery();
            sqlConnection.Close();

            return i;
        }

        public List<Kisi> KisileriGetir()
        {
            List<Kisi> kisiler = new List<Kisi>();
```

```
                MySqlConnection sqlConnection = new
MySqlConnection("server=localhost;user
id=root;database=egitim;pwd=123456qaZ.");
            MySqlCommand sqlCommand = new MySqlCommand("select
id, isim, soyisim, telefon from Kisiler", sqlConnection);

            sqlConnection.Open();
                      MySqlDataReader sqlDataReader =
sqlCommand.ExecuteReader();

            while (sqlDataReader.Read())
            {
                Kisi kisi = new Kisi();
                kisi.Id = sqlDataReader.GetInt32(0);
                kisi.Isim = sqlDataReader.GetString(1);
                kisi.Soyisim = sqlDataReader.GetString(2);
                kisi.Telefon = sqlDataReader.GetString(3);
                kisiler.Add(kisi);
            }


            sqlConnection.Close();
            return kisiler;
        }
    }
```

İlgili işlem sınıflarımızı da tanımladıktan sonra şimdi de aralarında ki benzerlik ve farklılıklarına göz atalım.

Benzerlikler:

- Her iki sınıf da IKisi interfacesinden türetilmiştir ve interface özelliği gereği aynı metot tanımlarını barındırırlar
- Metotların aldıkları parametre türleri ve dönüş türleri aynıdır.

Farklılıklar:

- Metotların gövdelerinde yer alan veri tabanı işlemleri sınıfların sorumlu oldukları veri tabanı sisteminin kodlarıdır.

Ve geldik son işleme: Hem SQL Server hem MySQL Server için gerekli sınıflar ve kodlar yazıldığına göre uygulamamızda ilgili veri tabanı yönetim sistemine nasıl ulaşacağız?

Öncelikle uygulamamızın hangi veri tabanı sisteminde çalışacağını bir sistem parametresi olarak kaydetmek gerekiyor. Bu kayıt işlemi Properties -> Settings altında veya app.config içerinde olabilir. Yazımız içerisinde bu parametreye "dbType" ismini vereceğiz.

İlk işlemimizi olan kişi ekleme metodunu çağırmak için gerekli kodları yazalım.

```
IKisi iKisi = null;

if (dbType == SQLServer)
{
    iKisi = new MsSqlKisiIslemleri();
}

else if(dbType == MySQL)
{
    iKisi = new MySqlKisiIslemleri();
}

Kisi kisi = new Kisi
{
```

```
    Isim = textBoxAd.Text,
    Soyisim = textBoxSoyad.Text,
    Telefon = textBoxTelefon.Text
};

int kisiEkle = iKisi.KisiEkle(kisi);
```

Öncelikle veri tabanı türüne göre ilgili sınıfı barındıracak olan IKisi interfacesinden bir tanımlama yapıyoruz. Daha sonra dbType isimli sistem parametresinin değerine göre bu interfaceye ilgili sınıfın bir örneği atanıyor. Bu adımdan sonra metotlar interface üzerinden çağrıldığında artık kendisine atanan sınıfın içeriğini çalıştırır. Dolayısı ile bu adımdan sonra uygulamamız artık çoklu veri tabanı yönetim sistemi desteğine sahip olmuş oluyor.

Kodlamada son olarak kişileri listeleme işlemi metotlarını çağıralım.

```
 IKisi iKisi = null;

if (comboBoxDb.SelectedIndex == 0)
{
    iKisi = new MsSqlKisiIslemleri();
}

else if (comboBoxDb.SelectedIndex == 1)
     {
          iKisi = new MySqlKisiIslemleri();
      }

List<Kisi> kisiler = iKisi.KisileriGetir();
```

Kişi ekleme işleminde olduğu gibi burada da oluşturulan interface ve ona atanan sınıflar üzerinden işlemler yapılıyor.

Yazımızın sonucunu mini bir öz eleştiri ile kapatacak olursak; Eğitimlerde, okulda gördüğümüz her yeni bilginin muhakkak surette pratik hayatta profesyonel kullanımda nasıl kullanıldSQL Server ve MySQL Server Destekli Yazılım Geliştirmeığını sorgulayalım. Kendi adıma ilk olarak çoklu veri tabanı destekli yazılım geliştirme konusunu gördüğümde "çok zor bir işlem" demiştim ancak işlemi yaptığımda gördüm ki aslında çok iyi bildiğim bir konuymuş.

Hepinize bol ve bugsız bir kodlama yaşamı diliyorum