Merhaba! MVC projelerinde LINQ (Language Integrated Query), veri çekme, filtreleme, sıralama ve diğer veri işlemlerini yapmanın güçlü ve okunabilir bir yoludur. LINQ sorguları ile veritabanı işlemleri çok daha basit ve anlaşılır hale gelir. Şimdi sana LINQ sorgularını hem açıklayıp hem de örneklerle detaylandırayım.

---

1. LINQ ile Temel Veritabanı İşlemleri

Veri Getirme (Select)

Bir tablodan tüm verileri getirmek için ToList veya AsEnumerable kullanabilirsin.

Örnek:

csharp
Kodu kopyala
```
var kullanicilar = db.Kullanicilar.ToList();
```

Bu sorgu, Kullanicilar tablosundaki tüm verileri liste olarak döner.

Filtreleme (Where)

Where metodu ile belirli bir koşula uyan kayıtları çekebilirsin.

Örnek:

csharp
Kodu kopyala
```
var aktifKullanicilar = db.Kullanicilar
    .Where(k => k.AktifMi == true)
    .ToList();
```

Bu sorgu, yalnızca aktif kullanıcıları getirir.

---

2. LINQ ile Sıralama

Artan Sıralama (OrderBy)

OrderBy kullanarak verileri bir alana göre sıralayabilirsin.

Örnek:

csharp
Kodu kopyala
```
var siraliKullanicilar = db.Kullanicilar
    .OrderBy(k => k.Ad)
    .ToList();
```

Bu sorgu, kullanıcıları Ad alanına göre alfabetik olarak sıralar.

Azalan Sıralama (OrderByDescending)

Azalan sıralama yapmak için OrderByDescending kullanılır.

Örnek:

```
csharp
Kodu kopyala
var tersSiraliKullanicilar = db.Kullanicilar
    .OrderByDescending(k => k.Ad)
    .ToList();
```

---

3. LINQ ile Projeksiyon (Seçim Yapma)

Belirli sütunları seçmek için Select kullanılır.

Örnek:

```
csharp
Kodu kopyala
var kullaniciAdlari = db.Kullanicilar
    .Select(k => new { k.Ad, k.Soyad })
    .ToList();
```

Bu sorgu, sadece Ad ve Soyad alanlarını döner.

---

4. Gruplama (GroupBy)

Verileri bir alana göre gruplamak için kullanılır.

Örnek:

```
csharp
Kodu kopyala
var grupKullanicilar = db.Kullanicilar
    .GroupBy(k => k.Rol)
    .Select(grup => new
    {
        Rol = grup.Key,
        KullaniciSayisi = grup.Count()
    })
    .ToList();
```

Bu sorgu, kullanıcıları Rol alanına göre gruplar ve her grup için kullanıcı sayısını döner.

---

5. İlişkili Tablolar (Join) ile Çalışma

Birden fazla tabloyu birleştirmek için Join kullanılabilir.

Örnek:

csharp
Kodu kopyala
```
var kullaniciRoller = db.Kullanicilar
    .Join(db.Roller,
          k => k.RolId,
          r => r.Id,
          (k, r) => new
          {
              KullaniciAd = k.Ad,
              RolAd = r.Ad
          })
    .ToList();
```

Bu sorgu, kullanıcılar ile roller tablolarını birleştirir ve her kullanıcının rolünü döner.

---

6. LINQ ile Toplama İşlemleri

Toplam Hesaplama (Sum)

Örnek:

csharp
Kodu kopyala
```
var toplamMaas = db.Calisanlar.Sum(c => c.Maas);
```

Bu sorgu, Calisanlar tablosundaki tüm maaşların toplamını döner.

Ortalama (Average)

Örnek:

csharp
Kodu kopyala
```
var ortalamaMaas = db.Calisanlar.Average(c => c.Maas);
```

En Büyük Değer (Max)

Örnek:

csharp
Kodu kopyala
```
var enYuksekMaas = db.Calisanlar.Max(c => c.Maas);
```

---

7. Performans İçin Öneriler

- Veritabanına gereksiz yere büyük sorgular göndermemek için sorguları doğru şekilde filtrelemeye dikkat et.
- `ToList()` çağrısını yalnızca gerekli olduğunda kullan, çünkü bu metot sorguyu hemen çalıştırır.
- Karmaşık sorgularda önceden `Include` ile ilişkili verileri dahil etmeyi düşün (Eager Loading).

---

1. İçinde Geçenleri Bulma (Contains)

Bir alandaki değerin belirli bir metni içerip içermediğini kontrol etmek için kullanılır.

Örnek:

```
csharp
Kodu kopyala
var icindeABCGeçenler = db.Kullanicilar
    .Where(k => k.Ad.Contains("ABC"))
    .ToList();
```

Bu sorgu, `Ad` alanında "ABC" geçen tüm kullanıcıları getirir.

---

2. Başlangıç Kontrolü (StartsWith)

Bir alanın belirli bir metinle başlayıp başlamadığını kontrol eder.

Örnek:

```
csharp
Kodu kopyala
var adBaslayanA = db.Kullanicilar
    .Where(k => k.Ad.StartsWith("A"))
    .ToList();
```

Bu sorgu, adı "A" harfiyle başlayan kullanıcıları döner.

---

3. Bitiş Kontrolü (EndsWith)

Bir alanın belirli bir metinle bitip bitmediğini kontrol eder.

Örnek:

```
csharp
Kodu kopyala
var adBitenZ = db.Kullanicilar
    .Where(k => k.Ad.EndsWith("Z"))
    .ToList();
```

Bu sorgu, adı "Z" harfiyle biten kullanıcıları getirir.

---

4. Büyük, Küçük veya Eşitlik Kontrolleri

LINQ'de sayısal ve tarihsel karşılaştırmalar da yapılabilir.

Büyüktür (>)

Örnek:

csharp
Kodu kopyala
```
var maasiBuyuk5000 = db.Calisanlar
    .Where(c => c.Maas > 5000)
    .ToList();
```

Bu sorgu, maaşı 5000'den büyük olan çalışanları döner.

Küçüktür (<)

Örnek:

csharp
Kodu kopyala
```
var maasiKucuk2000 = db.Calisanlar
    .Where(c => c.Maas < 2000)
    .ToList();
```

Büyük Eşit (>=)

Örnek:

csharp
Kodu kopyala
```
var maasi5000veUzeri = db.Calisanlar
    .Where(c => c.Maas >= 5000)
    .ToList();
```

Küçük Eşit (<=)

Örnek:

csharp
Kodu kopyala
```
var maasi2000veAlti = db.Calisanlar
    .Where(c => c.Maas <= 2000)
    .ToList();
```

---

5. Mantıksal Operatörler

Birden fazla koşulu birleştirmek için mantıksal operatörler kullanılır.

VE Koşulu (&&)

Örnek:

csharp
Kodu kopyala
```
var aktifVeRolAdmin = db.Kullanicilar
    .Where(k => k.AktifMi && k.Rol == "Admin")
    .ToList();
```

Bu sorgu, hem aktif olan hem de rolü "Admin" olan kullanıcıları getirir.

VEYA Koşulu (||)

Örnek:

csharp
Kodu kopyala
```
var adminVeyaModerator = db.Kullanicilar
    .Where(k => k.Rol == "Admin" || k.Rol == "Moderator")
    .ToList();
```

Bu sorgu, rolü "Admin" veya "Moderator" olan kullanıcıları getirir.

---

6. Aralık Kontrolü

Belirli Bir Aralıkta (&& ile)

Örnek:

csharp
Kodu kopyala
```
var maasArasi = db.Calisanlar
    .Where(c => c.Maas >= 3000 && c.Maas <= 7000)
    .ToList();
```

Bu sorgu, maaşı 3000 ile 7000 arasında olan çalışanları döner.

---

7. Boş veya Null Kontrolü

Boş Kontrolü (string.IsNullOrEmpty)

Örnek:

csharp
Kodu kopyala
```
var bosAdliKullanicilar = db.Kullanicilar
```

```
    .Where(k => string.IsNullOrEmpty(k.Ad))
    .ToList();
```

Bu sorgu, adı boş olan veya null olan kullanıcıları döner.

Null Olmayanlar (`!= null`)

Örnek:

```
csharp
Kodu kopyala
var aktifKayitlar = db.Kullanicilar
    .Where(k => k.SilmeTarihi != null)
    .ToList();
```

Bu sorgu, silme tarihi olmayan (aktif) kayıtları getirir.

---

8. Negatif Kontroller

Belirli bir koşulun sağlanmadığını kontrol etmek için `!` operatörü kullanılır.

Örnek:

```
csharp
Kodu kopyala
var aktifOlmayanlar = db.Kullanicilar
    .Where(k => !k.AktifMi)
    .ToList();
```

Bu sorgu, aktif olmayan kullanıcıları getirir.

---

9. Özel Karşılaştırmalar (`Any`, `All`, `Count`)

Bir Kayıt Var Mı (`Any`)

Örnek:

```
csharp
Kodu kopyala
var adminVarMi = db.Kullanicilar
    .Any(k => k.Rol == "Admin");
```

Bu sorgu, rolü "Admin" olan en az bir kullanıcı olup olmadığını kontrol eder.

Tüm Kayıtlar Sağlıyor Mu (`All`)

Örnek:

```
csharp
Kodu kopyala
var hepsiAktifMi = db.Kullanicilar
    .All(k => k.AktifMi);
```

Bu sorgu, tüm kullanıcıların aktif olup olmadığını kontrol eder.

Kayıt Sayısı (Count)

Örnek:

```
csharp
Kodu kopyala
var toplamKullaniciSayisi = db.Kullanicilar
    .Count(k => k.AktifMi);
```

Bu sorgu, aktif kullanıcıların toplam sayısını döner.

---

10. Özelleştirilmiş Koşullar

Belirli Bir Liste İçinde Mi (Contains ile)

Örnek:

```
csharp
Kodu kopyala
var roller = new List<string> { "Admin", "Moderator" };
var belirliRoller = db.Kullanicilar
    .Where(k => roller.Contains(k.Rol))
    .ToList();
```

Bu sorgu, rolü "Admin" veya "Moderator" olan kullanıcıları getirir.