Dökümanı forklayan adının yanına bir tik atıp PR atsın.

Emre Çalışkan x

Senem Canpolat x

Berkay Gülşen

Eda Kurnaz x

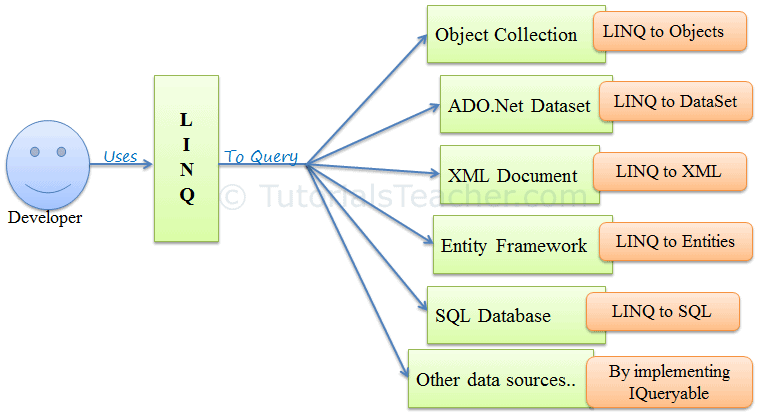
Rıdvan Öztürk x

**LİNQ NEDİR?**

LINQ; açılımı **Language Integrated Query** (Dile Entegre Edilmiş Sorgu) olan, Microsoft tarafından geliştirilen ve ilk olarak 2007'de .NET Framework 3.5'in büyük bir parçası olarak piyasaya sürülen .NET dillerine yerel veri sorgulama yetenekleri ekleyen bir Microsoft .NET Framework bileşenidir. C # ve VB.NET'teki tek tip sorgu sözdizimidir. C# veya VB'ye entegre edildiği için programlama dilleri ve veritabanları arasındaki uyuşmazlığı ortadan kaldırır.

LINQ, koleksiyonlar, ADO.Net DataSet, XML , SQL Server, Entity Framework ve diğer veritabanları gibi farklı veri kaynağı türlerinden veri almak için oluşturulmuş bir sorgu söz dizimidir.

**LINQ NESNELERİ**



**LINQ sorguları, sonuçları nesneler olarak döndürür.**



**LİNQ AVANTAJLARI**

1. Farklı kaynaklardaki veriler için ortak sözdizim.
2. Daha kısa ve okunabilir kod.
3. Derleme zamanı hata denetimi ve VS IntelliSense özelliği. Bu çalışma zamanı hatalarından kaçabilmemiz ve daha kolay kod yazabilmemiz için akıllı tahmin yeteneğinin desteklenmesi anlamına gelir.
4. Kolayca sıralama, filtreleme ve gruplama özellikleri sağlar.

**LİNQ DEZAVANTAJLARI**

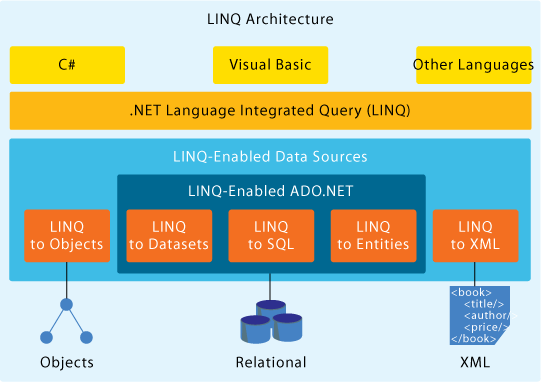
1. LINQ to SQL ile SQL le yazılan kadar detaylı ve karmaşık sorgulamalar yapılamaz.
2. SQL Server’da her sorgu için bir sorgu planı oluşturulur ve bu sorgu planı önbelleğe atılır daha sonra aynı sorgu çalıştırılacağı zaman tekrar sorgu planı hazırlamakla uğraşılmaz ve aynı sorgu planıyla tekrar çalıştırılır. LINQ bu özellikten yararlanmaz.
3. Doğru yazılmamış LINQ sorguları performans düşüşüne sebep olabilir.
4. Sorgularda değişiklik yapılırsa kod tekrar derlenmeli ve dll uzak bilgisayara tekrar atılmalı.

KISACA LINQ NEDIR?

LINQ açılımı dile entegre edilmiş sorgu. Hayatımıza C# ile girmiştir. Farklı veri kaynaklarından sorgulama yapma imkanı sunar.

LINQ Mimarisi 3 katmandan oluşmaktadır. En üstteki katman LINQ yapısını kullanacak olan programlama dilleridir. Ortadaki katman LINQ sorgularının işlendiği katmandır. En alttaki katman ise LINQ için çeşitli veri yapılarını bulundurur. Bu veri yapıları ile LINQ Query katmanı arasında LINQ provider (sağlayıcı) iletişim kurar. Sağlayıcı sayesinde sorgular anlaşılır hale gelir.

LINQ farklı veri kaynaklarına erişim sağlayabilir. Veri üzerindeki işlemleri kolaylaştırır.



LINQ VERİ SAĞLAYICILARI

LINQ to Object : List<T> tipindeki yapıların sorgulanması amacıyla geliştirilmiştir.

LINQ to SQL: Veriyi tutan ortamlardaki verilere erişmek ve erişilen verilerin üzerinde işlem yapmak amacıyla geliştirilimiştir.

LINQ to XML: Veri kaynağına parsel kullanılmadan hızlı bir biçimde erişilmek için geliştirilmiştir.

NEDEN LINQ TERCIH ETMELIYIZ?

-Sorguların yazımında üretkenlik düşmektedir.

-Çalışma esnasında veri tipi uyuşmazlığı ile karşılaşırız.

-Tip kontrolü yapılmaz. Bundan ötürü hatalar meydana gelebilir.

LINQ to Object

Hafızada bulunan ve Enumerable<T> arayüzünü uygulayan bütün koleksiyonlarda kullanılabilir. From where ve select gibi yapıları SQL ortamındaki gibi kullanarak işlem yapabiliriz.

LINQ to SQL

LINQ ifadeleri arka planda Queryable arayüzü ile uyumlu nesneler üretir. Bu arayüz içinde LINQ ifadeleri kollara ayrılır. Kollara ayıran nesnenin adı Expressiondur. Queryprovider isimli arayüz sayesinde ise dinamik sorgu cümleleri oluşturur.

Bu sorguda çalışma esnasında t-sql yapısına çevrilir. Çevirilen yapı veri tabanı üzerinde direkt olarak sorgulama yapabilir.

LINQ to XML

LINQ mimarisinin XML verilerini sorgulaması için üretilmiştir. Bu sınıfların tamamı Syste.Xml.Linq olarak adlandırılır.

**LINQ - EF İlişkisi**

**Entity Framework:** .Net platformunda ORM (Object Relational Mapping) araçlarından birisidir. ORM (Object Relational Mapping) ise veritabanı ile nesneye yönelik programlama (OOP) arasındaki ilişkiyi kuran teknolojidir. Basite indirgemek istersek, Nesnelerimizi ilişkisel veritabanındaki tablomuza bağlayan ve veri alış-verişini bizim için yapan bir tekniktir/metodtur. ORM tekniği belli bir programlama diline bağlı değildir ve her nesneye yönelik programlama dillerinde yazılabilir/kullanılabilir. Yani Entity Framework, nesne tabanlı programlamada veritabanındaki tablolara uygun nesneler oluşturma tekniğidir.

Entity Framework ‘ün amacı yazılım geliştiricileri katı SQL sorgularından kurtararak ORM (Object Relational Mapping) imkânı sağlar. LINQ sayesinde yazılım geliştirici katı, uzun ve karmaşık SQL sorgularını yazmaktan kurtulur. Bunun yanında Entity Framework CRUD (Create, Read, Update, Delete) işlemleri ile uygulama ve veritabanı arasındaki iletişimin yapılmasını sağlar.

* CRUD (Create, Read, Update, Delete) işlemleri ile uzun, karışık ve zahmet verici SQL kodlarından kurtulmamızı sağlar.
* Kodu veritabanından soyutlar, yani kod tarafında EF için oluşturulan entity’ler aslında veritabanı tarafındaki tabloları ve bu tablolar arasındaki bağlantıları (one to one, one to many ve many to many gibi) temsil ederler.
* Veritabanı işlemlerinde nesneye yönelik kod yazmamızı sağlar.
* Daha sade ve zahmetsiz SQL sorguları sayesinde veritabanı performansını artırır.
* Yazılım geliştiricinin data işlemleri ile haşır neşir olmadan sadece uygulama üzerinde odaklanmasına olanak sağlar.
* Kod yazma süresini kısaltarak daha az zamanda daha çok iş yapmayı sağlar.
* SQL tarafındaki bütün sorguları (query’leri) LINQ kullanarak kod tarafında yazmamızı sağlar

Entity Framework ile 3 farklı yöntem ile proje geliştirilebilir. Bu yöntemler;

1. Model First (New Database)

2. Database First (Existing Database)

3. Code First (New Database)

**Model First (Önce Model):** Bu yöntemde Visual Studio üzerinde boş bir model dosyası (.edmx) eklenerek veri tabanı bu model üzerinde tasarlanır. Derleme adımında verilen script dosyasi ile veri tabanı oluşturulur.

**Database First (Önce Veritabanı):** Bu yöntemde hali hazırda var olan veritabanı projeye model dosyası ile bağlanır ve gerekli class’lar EF tarafından üretilir.

**Code First (Önce Kod — Yeni Veritabanı):** Bu yöntemde classlar ve mapping kodları yazılımcı tarafından oluşturulur. Daha sonra veri tabanı bu class’lardan türetilir.

**Yazım Kuralları**

Linq’in üretilme sebebi gereği görece esnek kurallara sahip yazım kuralları vardır. Her veri kaynağı için ayrı bir sorgu dili, yazım dili öğrenmek yerine tek bir sorgu dili öğrenerek hepsine aynı sorguyu atabilirsiniz. Linq sorguları iki farklı şekilde yazılabilir. Bunlardan biri, SQL sorgularına benzer yapıdadır. Örneğin SQL’deki gibi “where” ve “select” gibi komutların C# içinde kullanıldığını görürüz. (Başka dillerde de Linq kullanılabilmektedir. Ancak hiçbirinde C#’da olduğu gibi dile dahil değildir. Üçüncü parti olarak eklenebilen kütüphane şeklinde yer alır. Örn: PHP, Javascript, TypeScript, ActionScript) Örnek vermek gerekirse:

…

**int**[] sayilar = **new** **int**[] { 5, 4, 1, 3, 9, 8, 6, 7, 2, 0 };

**var kucukSayilar1 = from sayi in sayilar  
where sayi < 5  
select sayi;**

…

Diğer sorgu tipi ise metod kullanılarak yazılan ve daha kısa olan yazım biçimidir. Döngülenebilir veri tiplerinden olan bir değişkene property olarak “where()” geçersek, aynı sql sorgusu atar gibi bir koşul sağlayarak istediğimiz işlemi kısa bir zamanda yapabiliriz. Örneğin yukarıda yaptığımız işlemi:

**var** kucukSayilar2 = sayilar.Where(x => x < 5);

şeklinde yazabiliriz. Buna Lambda Expression denir. “Where” den return edilen değer x parametresi ok işaretinden sağ tarafta sıralanan koşullara uyan elemanları döndürür. Buna örnek olarak şunu da verebiliriz.

Bir tane sınıf yaratıp onun propertyleri üzerinde koşul kontrol ederek yaptığımız işlemi tam olarak bir SQL sorgusu işlevine getirebiliriz. Örneğin:

class Araba {

public int ArabaId { get; set; }

public string Marka { get; set; }

public double MotorHacmi { get; set; }

public int Fiyat

}

Araba adında bir sınıfımız olsun ve biz bu sınıfı kullanarak 3 tane arabadan oluşan bir List yaratalım:

List<Araba> arabalar = new List<Araba> {

new Araba{ArabaId=1, Marka=”Volkswagen”, MotorHacmi=2.0, Fiyat=500000}

new Araba{ArabaId=2, Marka=”Renault”, MotorHacmi=1.6, Fiyat=250000}

new Araba{ArabaId=3, Marka=”Mercedes”, MotorHacmi=3.0, Fiyat=2000000}

}

Artık “arabalar” listesinden istediğimiz arabayı Linq kullanarak çekebiliriz. Bunun için

var ucuzArabalar = arabalar.Where(araba => araba.Fiyat < 300000)

yazarak almak istediğimiz değeri yeni değişkene atayabiliriz. Sonunda ise bu son listede foreach döngüsü kullanarak istediğimiz işlemi yaptırabilir, ya da bunu ihtiyacımıza göre return edebiliriz. Yani aslında LINQ hem yazım anlamında okunabilirliği kolaylaştırırken hem de daha az karakterle ve daha hızlı sorgularla bize performans ve rahatlık getirir.

Aynı zamanda SQL’de yaptığımız JOIN, FIRST, LAST gibi komutları Linq de destekler. Buna örnek olarak iste yine

var sonuc = arabalar.Join(insanlar, araba => araba.ArabaId, insan => insan.AracID, (araba, insan) => new { AracSahibi = insan.Ad, SahibiOlduguArac = araba.Marka })

şeklinde yazılabilir. SQL kullanan biri için bu oldukça akıcı şekilde yazılabilir bir kod bloğudur.

**Üretilme Sebebi**

LINQ’in üretilme sebebi aslında her türlü veri kaynaklarından veri kullanırken o kaynak için gereken sorgu diline ya da farklı sorgu şekillerine ayak uydurmak zorunda olmamızdı. LINQ yokken veritabanı için ayrı, XML için ayrı sorgu yazma ve bunun gibi daha bir sürü sorgulama ile ilgili zamandan ve geliştirici rahatlığından ödün veren yöntemler kullanılıyordu. LINQ Microsoft tarafından getirildikten sonra ise veri kaynağınız ne olursa olsun, sorgunuzu sadece LINQ yazım kurallarına göre yazmanız yeterli olmuştu. Artık sorgular elle yazılmadığı için hem daha standardize(compiler için daha performanslı); hem de aynı sebepten yazımı daha rahat hale geldi.

**REFERANSLAR**

* <http://hurkanseyhan.blogspot.com/2014/04/linq-nedir.html>
* <https://medium.com/t%C3%BCrkiye/entity-framework-nedir-bff3943bec72>
* <https://www.mehmetaltunel.com/csharp/entity-framework-nedir-ne-ise-yarar/>
* <https://www.yusufsezer.com.tr/entity-framework/>
* <https://coskunkurtuldu.medium.com/net-core-entity-framework-code-first-yakla%C5%9F%C4%B1m%C4%B1-6ac02867b3e5>
* <https://hakanyalitekin.medium.com/linq-nedir-f55957d5bf52>