



Entity Framework

Mål med lektionen!

- Lite mer LINQ
- Lite mer om diagrammen och lite allmänna tips.

Vad lektionen omfattar

- Titta lite på vilka varianter av LINQ vi har, och jobba lite mer med detta.
- Titta på lite tips och kolla på ett bra verktyg.

Vilka varianter av LINQ har vi?

- LINQ to Objects
- LINQ to XML
- LINQ to SQL
- LINQ to Entities

LINQ to SQL

Linq to sql är en OR/M (object relational mapper)
Fungerar med SQL Server 2000 -> (och frammåt)
Översätter query uttryck (våra query frågor) till T-SQL

Lite LINQ operatorer och extensions

Alla Queryable operatorerna som finns på ett LINQ objekt är inget annat än en hel hög med Extension metoder på vårt LINQ objekt så länge det är av typen IEnumerable så kan alla operatorerna "jobba" mot <T> (typen T).

Det ger oss en massa operatorer som Select, OrderBy Where och många fler.

Uppgift

Skapa upp en databas med i alla fall följande tabeller: Film och Kategori. Film skall innehålla följande kolumner: id, titel, längd, kategoriId, utgivningsår och omdöme. Kategori skall innehålla följande: id, Namn.

Skapa sedan upp ett webbforms projekt och gör EF-Database first koppling, med gränssnitt så att användaren kan lägga in filmer och skriva omdömen om dem och i övrigt uppdatera informationen om filmen.

Entity Framework

- LINQ to Entities and Entity SQL
- Change tracking and identity management
- Serialization and data binding
- Connection and transaction management

EF 6.0

Det finns en ny funktion i EF 6.0 som fungerar på ett sätt som påminner lite om delegates (funktionalliteten).

Med denna funktion så kan vi enkelt logga allt som EF gör genom *Context.Database.log*

Allt vi behöver göra är att koppla på vilken metod som helst som accepterar en sträng parameter och returnerar void.

```
using (var context = new SchoolDBEntities())
{
    context.Database.Log = Console.Write;
    var student = context.Students
        .Where(s => s.StudentName == "Student1").FirstOrDefault<Student>();

    student.StudentName = "Edited Name";
    context.SaveChanges();
}
```

Koden på föregående sida ger följande:

```
Opened connection at 14-05-2014 02:43:49 +05:30
SELECT TOP (1)
    [Extent1].[StudentID] AS [StudentID],
    [Extent1].[StudentName] AS [StudentName],
    [Extent1].[StandardId] AS [StandardId]
FROM [dbo].[Student] AS [Extent1]
WHERE 'Student1' = [Extent1].[StudentName]
-- Executing at 14-05-2014 02:43:50 +05:30
-- Completed in 1 ms with result: SqlDataReader

Closed connection at 14-05-2014 02:43:50 +05:30
Opened connection at 14-05-2014 02:43:50 +05:30
Started transaction at 14-05-2014 02:43:50 +05:30
UPDATE [dbo].[Student]
SET [StudentName] = @0
WHERE ([StudentID] = @1)
-- @0: 'Edited Name' (Type = AnsiString, Size = 50)
-- @1: '32' (Type = Int32)
-- Executing at 14-05-2014 02:43:50 +05:30
-- Completed in 1 ms with result: 1

Committed transaction at 14-05-2014 02:43:50 +05:30
Closed connection at 14-05-2014 02:43:50 +05:30
-
```

Tips

Om vi har en code-first lösning och skulle vilja se hur modellen ser ut baserat på vår "context" klass och våra domän klasser utan att för den sakens skull kompilera hela projektet och därmed skapa upp hela databasen så kan man göra det genom att installera ett verktyg ifrån Microsoft genom att klicka på "Tools" i menyn -> välj "Extensions and Updates" i sökrutan skriver du " Entity framework power tools beta 4"


Extensions and Updates


Installed


All
Controls
Samples
Templates
SDKs
Tools
Search Results


Online
Updates (8)

Sort by: Name: Ascending


**Application Insights Tools for Visual Studio**
Gain visibility into your application using Application Insights right from Visual Studio.


**CKS - Dev for Visual Studio 2013**
The CKS - Development Tools Edition for Visual Studio 2013 is a collection of Visual Studio templates, Server Explorer extensions and tools providin...

**EF4/EF5 Model/Database First View Gen .tt for C#**
C# T4 template for creating pre-generated views for applications using Entity Framework (EF4 or EF5) Model/Database First (Edmx)

**Entity Framework Power Tools Beta 4**
Preview of useful design-time features for DbContext.

Disable
Uninstall





**Microsoft ASP.NET and Web Tools**
Provides the latest Web Developer Tools for ASP.NET


**Microsoft Office 365 API Tools**
Integrate your applications with Office 365 services such as mail, calendar,

Entity framework power tools

Created by: Microsoft
Date Installed: 2016-10-11
Version: 0.9.0.0
[More Information](#)
[Getting Started](#)

Open
Open With...
Entity Framework

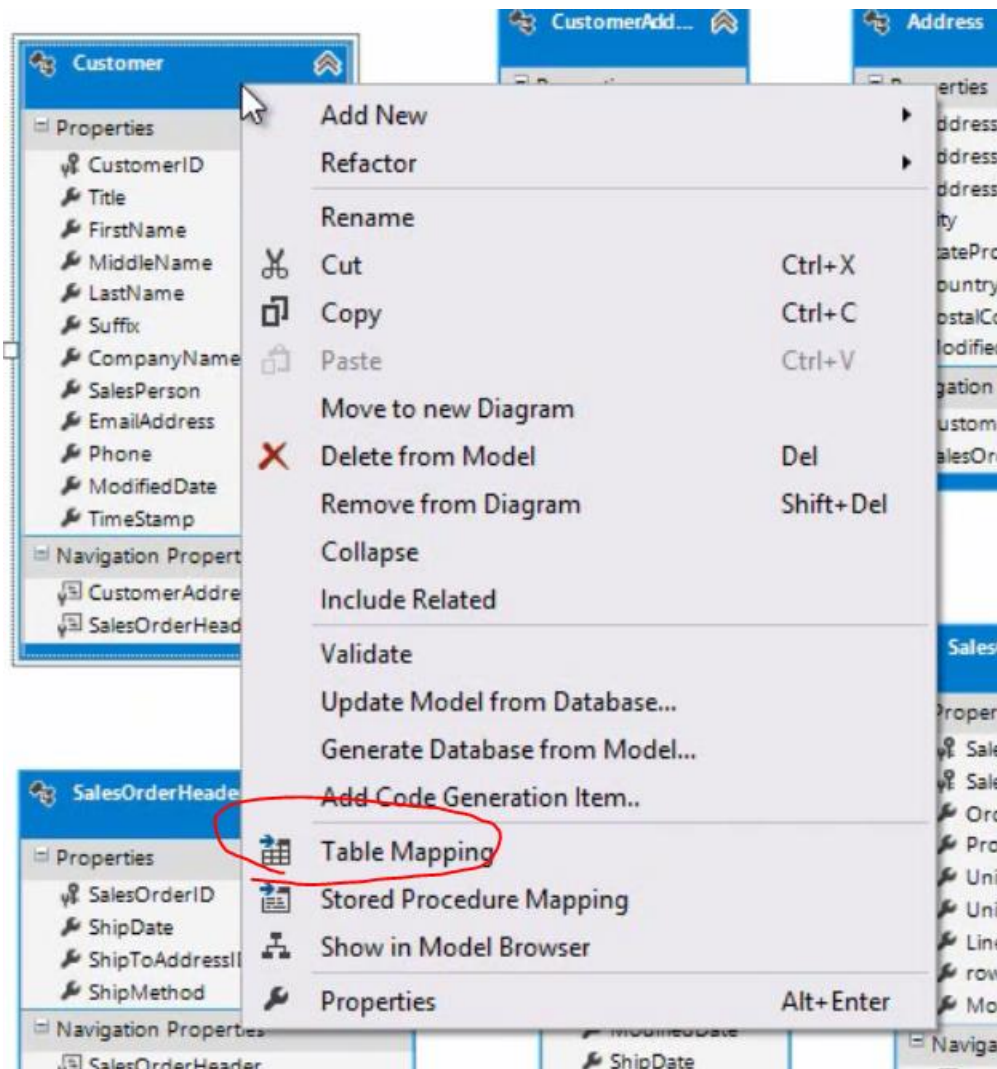
 View Entity Data Model (Read-only)
 View Entity Data Model XML
 View Entity Data Model DDL SQL
 Generate Views

 **Tahoe**
SOLUTIONS

Utforska edmx (diagrammet)

Edmx – filen som vi har tittat på tidigare innehåller även information över hur databas strukturen skall vara, tabeller kolumner, vyer och stored procedures (om dessa behövs och finns) (detta är sektionen för StorageModel)

Sedan har vi sektionen över mapping informationen som talar om hur vi överför databas strukturen till våra domän klasser, vad det motsvararas i datatyper och vilka dessa är. EF använder dessa när vi kör våra queries, dvs jobbar med databasen.



CustomerAdd... Address

Customer

Properties

CustomerID

Properties

CustomerID

AddressID

AddressLine1

Mapping Details - Customer

Column	Operat...	Value / Property
Tables		
Maps to Customer		
Add a Condition		
Column Mappings		
CustomerID : int	↔	CustomerID : Int32
Title : nvarchar	↔	Title : String
FirstName : nvarchar	↔	FirstName : String
MiddleName : nvarchar	↔	MiddleName : String
LastName : nvarchar	↔	LastName : String
Suffix : nvarchar	↔	Suffix : String
CompanyName : nvarchar	↔	CompanyName : String
SalesPerson : nvarchar	↔	SalesPerson : String
EmailAddress : nvarchar	↔	EmailAddress : String
Phone : nvarchar	↔	Phone : String
ModifiedDate : datetime	↔	ModifiedDate : DateTime
TimeStamp : timestamp	↔	TimeStamp : Binary
<Add a Table or View>		

Product

Uppgift

- I den första labben gjorde vi en DataBase first koppling, ta fram detta projektet och ta upp modellen.
- Därefter skall ni gruppera alla entiteterna med färg vilka som "hör ihop".
- Skapa nya diagram som är för varje "gruppering" (kom ihåg att sätta dit "rätt" färg).

Kustomisera vår Entity Framework

Om man vill ändra Entity Framework relaterade inställningar så tidigare var man tvungen att göra detta via App.Config filen tidigare.

Men nu i och med att Entity Framework 6.0 har kommit (och funnits ett tag) så kan vi faktiskt göra detta via kod numera. Kravet är att vi måste ha en klass som ärver DbConfiguration.

Vilken finns i:

`System.Data.Entity.DbConfiguration` namespace.

Kustomisera vår Entity Framework(forts).

Vi behöver antingen lägga till följande i App.config:

```
<entityFramework codeConfigurationType="MyNamespace.MyDbConfiguration, MyAssembly">  
  ...Your EF config...  
</entityFramework>
```

Eller så sätter vi det så här i koden (på vår context klass):

```
[DbConfigurationType(typeof(MyDbConfiguration))]  
public class MyContextContext : DbContext  
{  
}
```

```
public class MyConfiguration : DbConfiguration  
{  
  public MyConfiguration()  
  {  
    SetDefaultConnectionFactory(new LocalDbConnectionFactory("v11.0"));  
    SetProviderServices("My.New.Provider", new MyProviderServices());  
  }  
}
```

Lab 5