Översikt

- Arbeta med Entity Data Models.
- LINQ (Language Integrated Query).

.easec

Lektion 1: Arbeta med Entity Data Models

- Introduktion till ADO.NET Entity Framework.
- Stöd i ADO.NET Entity Framework.
- Versioner.
- Entity Data Model Tools.
- Skräddarsy genererade klasser.
- Exempel på klass som har generats av guide.
- Partial Class.
- Fördelar med Partial Class.
- Läsa och modifera information.
- Spara information.

.easec

Introduktion till ADO.NET Entity Framework

- ADO.NET Entity Framework tillhandahåller Entity Data Model (EDM).
- Entity Data Model:
  - Modeller som kan användas för att knyta databastabeller och frågor till .NET Framework objekt.
- Entity SQL:
  - Är ett lagringsoberoende frågespråk som ger möjlighet till att fråga och manipulera EDM construct.

.easec

Introduktion till ADO.NET Entity Framework (forts.)

- Object Service:
  - Tjänst som ger möjlighet att arbeta med Common Language Runtime (CLR) objekt.

.easec

Versioner

- Version 5.
- Version 6.
- Version 7.
- Core.

.easec

Stöd i ADO.NET Entity Framework

- ADO.NET Entity Framework har stöd för:
  - Skriva programkod mot konceptuell modell.
  - Lätt uppdatering av applikationen när ny datakälla skall användas.
  - Skriva programkod som är oberoende av lagring.
  - Skriva programkod som tillåter kontroller vid kompilering.

Entity Data Model Tools

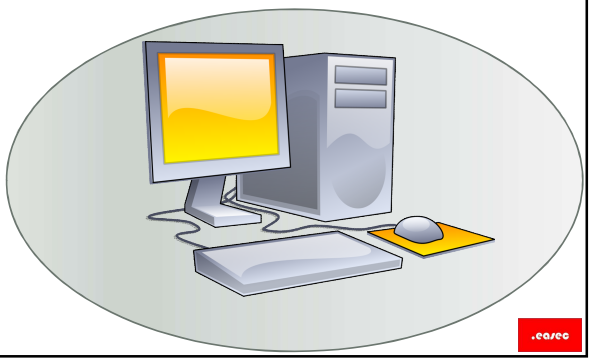
- Stöd finns för ett antal ingående verktyg:
  - Database-first design:
    - Du designer och skapar databas först innan modell genereras, begränsningar i flexibiltet.
  - Code-first design:
    - Designar poster för din applikation, skapar därefter databasstruktur runt dessa poster.
- Tillhandahåller möjlighet att visa, uppdatera och ta bort poster .
- Update Model Wizard:
  - Används för att uppdatera modell.

Entity Data Model Tools (forts.)

- Ett antal guider finns:

Guide	Beskriving
Entity Data Model Wizard	Ger möjlighet att generera ny konceptuell modell från existerande information, genom att använda design metoden database-first
Update Model Wizard	Ger möjlighet att uppdatera existernade konceptuell model med förändringar
Generated Database Wizard	Ger möjlighet att uppdatera existerande konceptuell modell, som har blivit designad i Entity Data Model Designer, genom att använda desigmetoden code-first

Övning Skapa Entity Data Model (10a)



.easec

Skräddarsy genererade klasser

- När du använder guiden Entity Data Model för att generera en modell, kommer klasser för att exponera poster i modell att automatiskt genereras.
- Dessa klasser innehåller egenskaper dom tillhandahåller tillgång till egenskaper för poster i databasen.
- Modifiera inte dessa klasser, använd partial class och partial methods för att lägga till specifik funktionalitet till de genererade klasserna.

.easec

Exempel på klass som har genererats av guide

```
namespace FourthCoffee.Employees
{
        using System;
        using System.Collection.Generic;
        public partial class Employee
        {
                public int EmployeeID { get; set; }
                public string FirstName { get; set; }
                public string LastName { get; set; }
                public Nullable<System.DataTime> DateOfBirth { get;
set }
                public Nullable<int> Branch { get; set; }
                public Nullable<int> JobTitle { get; set; }
                public virtual Branch Branch1 { get; set; }
                public virtual JobTitle JobTitle1 { get; set; }
        }
}
```

.easec

Partial Class

- Klasser i C# finns i separata fysiska filer, med extension .cs.
- I C# finns det möjlighet att dela upp en klass i flera filer, nyckelordet *partial* används.
- Partial kan appliceras på klass, metod, interface eller structure.
- Måste ha samma access modifiers.

.eo/ee

Partial Class -exempel

- PartialClassFil1.cs:

```
public partial class MyPartialClass
{
        public MyPartialClass()
        {
        }

        public void Method1(int val)
        {
                Console.WriteLine(val);
        }
}
```

.eo/ee

Partial Class -exempel

- PartialClassFil2.cs:

```
public partial class MyPartialClass
{
        public void Method2(int val)
        {
                Console.WriteLine(val);
        }
}
```

- Kompilator kommer att kombinera ihop dessa till en klassfil.

.eo/ee

Fördelar med Partial class

- Flera utvecklar kan arbeta med klass i separata filer.
- Vanligt vid automatisk generad kod, kod kan läggas till utan att behöva skapa om källfil.

Läsa och modifiera information

- Den automatiskt genererade programkoden för en modell, innehåller också en partial class som ärver från klassen System.Data.Entity.DbContext.
- Tillhandahåller funktionalitet för att fråga och arbeta med poster som objekt.
- Innehåller en standard constructor som initierar klass genom att använda anslutningssträng som guiden genererar.

Läsa och modifiera information (forts.)

- Klassen FourthCoffeeEntities:

```
public partial class FourthCoffeeEntities : DbContext
{
        public FourthCoffeeEntities() :
base(”name=FourthCoffeeEntities”)
        {
        }
        public DbSet<Branch> Branches { get; set; }
        public DbSet<Employee> Employees { get; set; }
        public DbSet<JobTitle> JobTitles { get; set; }
}
```

Läsa och modifiera information (forts.)

- Läsa information:

```
FourthCoffeeEntities DBContext = new FourthCoffeeEntities();

// Print a list of employees.
foreach (FourthCoffee.Employees.Employee emp in DBContext.Employees)
{
    Console.WriteLine("{0} {1}", emp.FirstName, emp.LastName);
}
```

- Modifiera information:

```
var emp = DBContext.Employees.First(e => e.LastName == "Prescott");
if (emp != null)
{
    emp.LastName = "Forsyth";
    DBContext.SaveChanges();
}
```

.eo/ec

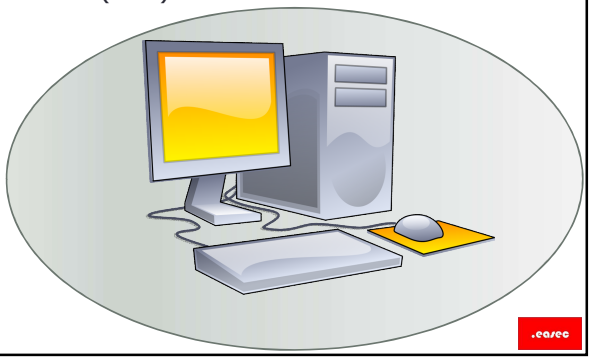
Spara förändringar

- När information har förändrats i modellen, måste den skrivas tillbaks.
- Detta görs genom att kalla på metoden SaveChanges i objektet ObjectContext.

```
DBContext.SaveChanges();
```

.eo/ec

Övning Läsa och modifiera information i EDM (10b)



.eo/ec

Lektion 2: LINQ (Language Integrated Query)

- Använda LINQ.
- Välja ut information.
- Filtrera information på kolumn.
- Använda Anonymous Types.
- Tvinga fram exekvering.

.easec

Använda LINQ

- Ett alternativ till Entity Framework är att använda LINQ.
- LINQ kan användas för att ställa frågor till ett stort antal källor:
  - .NET Framework collections, SQL Server databaser, ADO.NET data sets och XML-dokument.
  - Alla datorkällor som implementerar Ienumerable.
- Används för att:
  - Välja ut information.
  - Filtrera information på rad eller kolumnnivå.
  - Kan även användas för beräkningar, t ex räkna antalet poster.

.easec

Välja ut information

- Använda Select för att hämta all information för en post:

```
IQuerable<Employee> emps = from e in DBContext.Employees
                                                    select e;
```

- Det som kommer tillbaks är IQueryable<Employee>, vilket ger möjlighet att läsa igenom information som kommer tillbaks.
- Filtrera information på rad:

```
string _LastName = ”Prescott”;
IQuerable<Employee> emps = from e in DBContext.Employees
                                    where e.LastName == _LastName
                                    select e;
```

- Exemplet använder nyckelordet *where* för att returnera radinformation där användare har efternamn Prescott.

.easec

Filtrera information på kolumn

- Exemplet visar hur ny typ deklareras, för att lagra information från kolumn som frågan kommer att returerna:

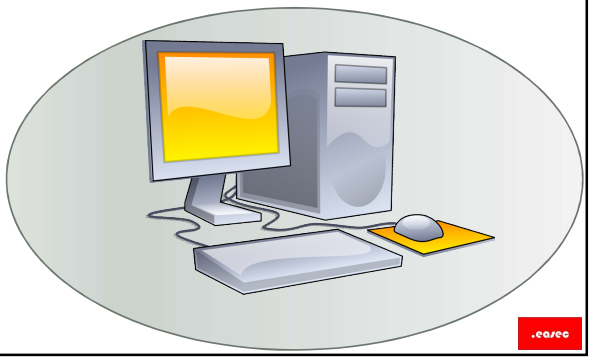
```
private class FullName
{
        public string Forename { get; set; }
        public string Surname { get; set; }
}
Private void FilteringDataByColumn()
{
        IQuerable<FullName> names = from e in DBContext.Employees
                                    select new FullName { Forename
= e.FirstName, SurName = e.LastName };
}
```


Filtrera information på kolumn (forts.)

- Få tillgång till resultatet:

```
foreach (var emp in emps)
{
        Console.WriteLine({0} {1}", emp.FirstName, emp.LastName);
```


Övning Använda LINQ to Entities (10c)



Använda Anonymous Types

- I tidigare exempel, returneras information till varabel IQuerable<Type>.
- Anonymous Types kan användas för att returnera värde.
- Nyckelordet *new* används för att skapa instans för variabel. Inget namn används för typ.
- Detta öppnar upp möjlighet att göra mer avancerade frågor med LINQ:
  - Gruppera resultatet.
    - Räkna fram t ex antalet poster, med aggregating.
    - Navigera i informationen.

.easec

Övning Använda Anonymous Types (10d)

.easec

Tvinga fram exekvering

- När LINQ-fråga definieras, körs den inte förrän du försöker använda det returnerade värdet.
- I exemplet kommer inte fråga att köras förrän foreach-blocket starts:

```
IQueryable<Employee> emps = from e in DBContext.Employees
                            select e;
foreach (var emp in emps)
{
       Console.WriteLine("{0} {1}", emp.FirstName, emp LastName);
```

- Om du däremot definierar LINQ query som returnerar ett singeltone värde, Average, Count eller Max, kommer query att köras direkt.
- Nödvändigt, för sekvens av resultat behövs.

.easec

Tvinga fram exekvering (forts.)

- Standardbeteende kan förändras genom att kalla på metod:
  - ToArray.
  - ToDictionary.
  - ToList.

```
IList<Employee> emps = (from e in DBContext.Employees
                                    select e;).ToList()
foreach (var emp in emps)
{
       Console.WriteLine(”{0} {1}”, emp.FirstName, emp LastName);
```

.easec

Repetitionsfrågor

.easec