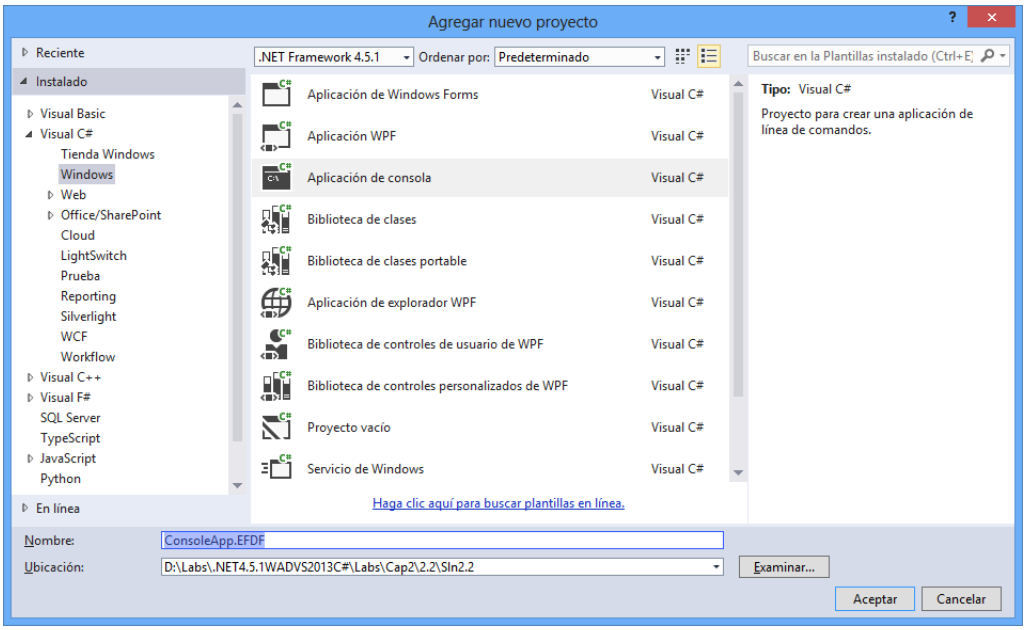
Guía Complementaria a Laboratorio 2.2

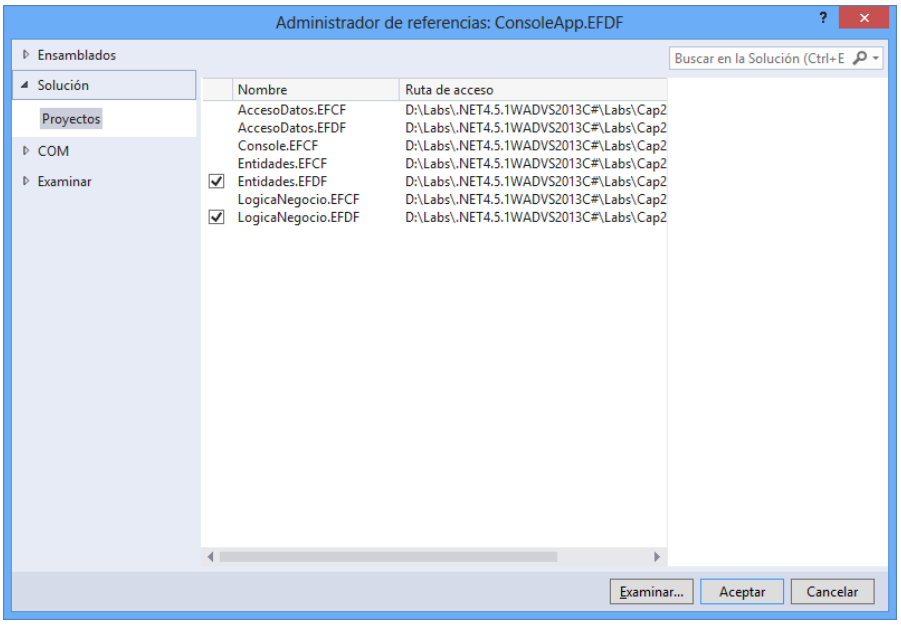
Profesor: Ing. Angel Ramos – Cibertec DAT

Probando EF Database First

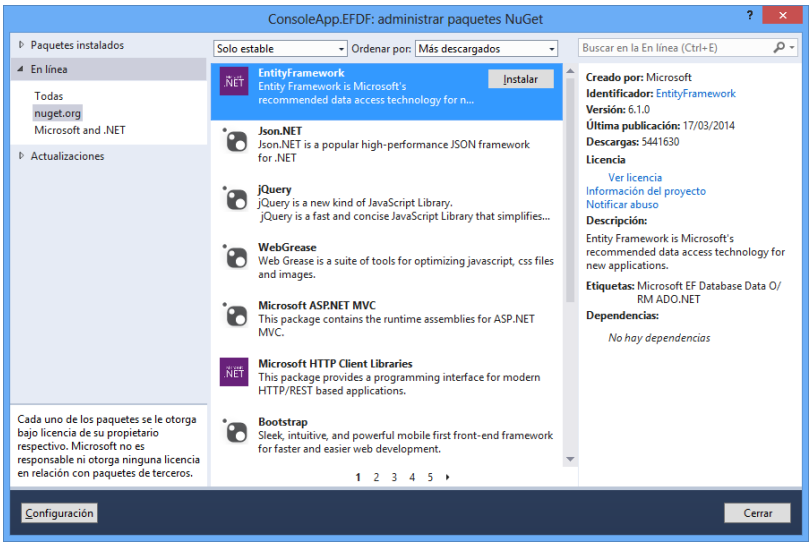
1. Clic derecho sobre la carpeta EF.DatabaseFirst y crear un proyecto del tipo **Aplicación de consola** con el nombre **ConsoleApp.EFDF**.



2. Agregue las referencias a los proyectos Entidades.EFDF y LogicaNegocio.EFDF



3. Agregue el paquete **EntityFramework** al proyecto **ConsoleApp.EFDF** desde la Administración de paquetes NuGet (clic en Instalar)



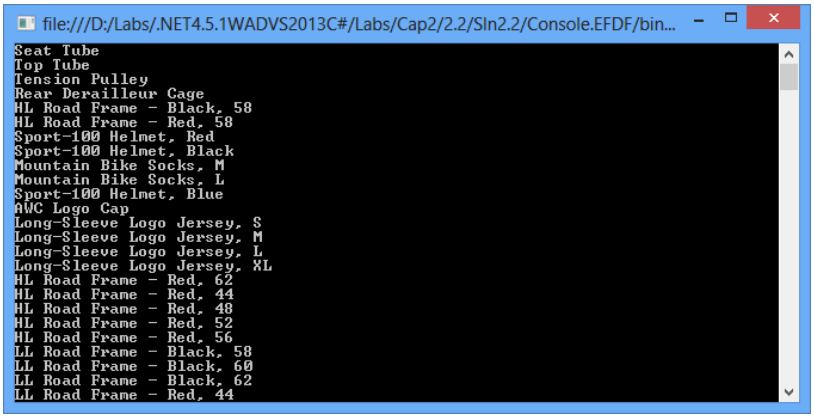
4. Desde el archivo **App.config** del proyecto **AccesoDatos.EFDF** copie la cadena de conexión `AdventureWorksContext` al archivo **App.Config** del proyecto **ConsoleApp.EFDF**.
5. Abra la clase **Program.cs** e ingrese lo siguiente:

```
using Entidades.EFDF;
using LogicaNegocio.EFDF;
```

6. En la misma clase **Program.cs** ingrese el siguiente código

```
class Program
 {
     static void Main(string[] args)
     {
         ProductLN productLN = new ProductLN();
         IEnumerable<Product> products = productLN.SelectAll();

         products.ToList().ForEach(p => Console.WriteLine(p.Name));

         Console.ReadLine();
     }
}
```

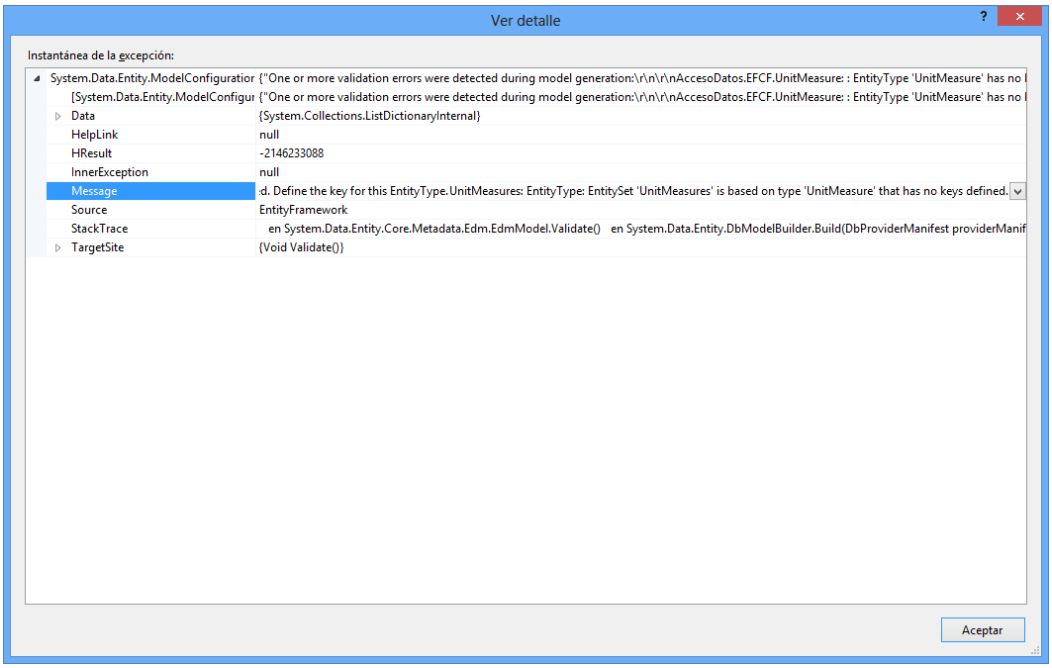
7. Establezca el proyecto **ConsoleApp.EFDF** como proyecto de inicio de la solución y ejecútelo

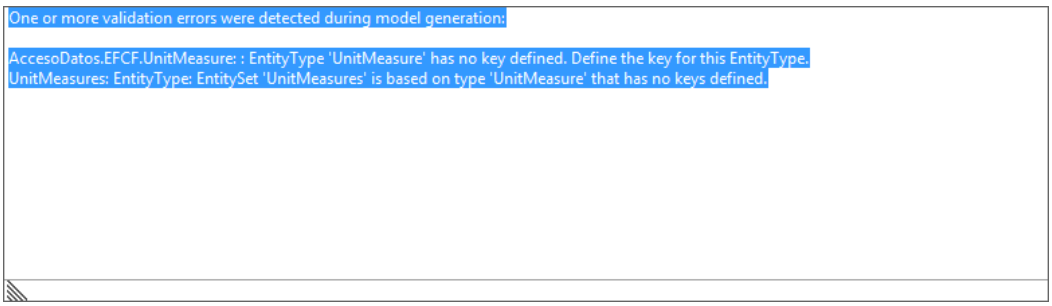


8. Para más información:
   http://msdn.microsoft.com/en-us/data/jj206878

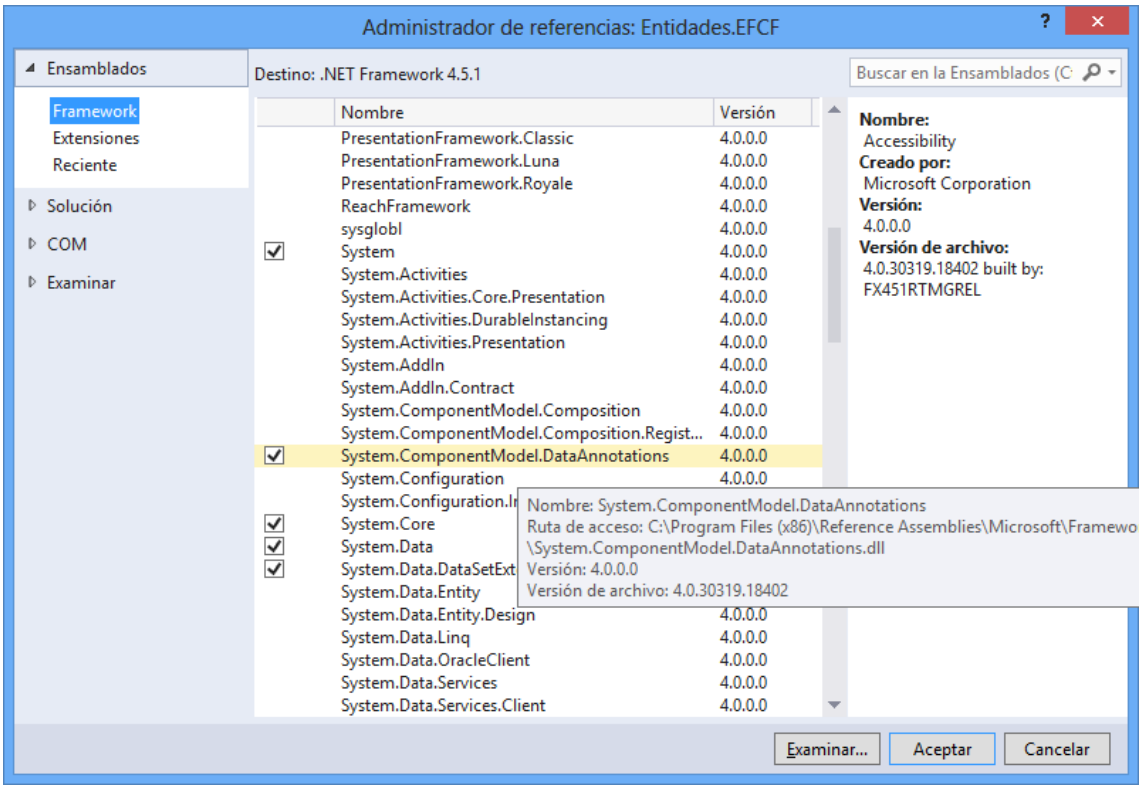
Probando EF Code First

1. Similar a los pasos realizados para EF Database First, agregar un proyecto a la carpeta **EF.CodeFirst** del tipo Aplicación de consola con el nombre **ConsoleApp.EFCF**
2. Agregar las referencias a los proyectos **Entidades.EFCF** y **LogicaNegocio.EFCF** al proyecto **ConsoleApp.EFCF**
3. Agregar **Entity Framework** desde la Administración de paquetes NuGet al proyecto **ConsoleApp.EFCF**
4. Desde el archivo **App.config** del proyecto **AccesoDatos.EFCF** copie la cadena de conexión `AdventureWorksContext` al archivo **App.Config** del proyecto **ConsoleApp.EFCF**.
5. Repita los pasos 5 y 6 del ejemplo anterior.
6. Establezca el proyecto **ConsoleApp.EFCF** como proyecto de inicio de la solución y ejecútelo
7. Al ejecutar la aplicación nos retorna un error indicándonos que la entidad UnitMeasure no tiene una llave (key) definida. Esto ocurre por convenciones de EF CodeFirst, normalmente reconoce una llave a la propiedad que se llama igual que la entidad con el sufijo Id. Ejm: Entidad Producto, Llave: ProductoId.





8. Necesitamos indicarle al Entity Framework cual es la llave de la entidad UnitMeasure, para esto realizaremos lo siguiente:
   Agregar al proyecto **Entidades.EFCF** la referencia a **System.ComponentModel.DataAnnotations**



Abrir la clase **UnitMeasure.cs** y realizar las siguientes modificaciones:

```
using System;
using System.Collections.Generic;
using System.ComponentModel.DataAnnotations;

namespace Entidades.EFCF
{
        public partial class UnitMeasure
        {
                public UnitMeasure()
                {
                        this.ProductSizeList = new HashSet<Product>();
                        this.ProductWeightList = new HashSet<Product>();
                }

                [Key]
                public string UnitMeasureCode { get; set; }
               public string Name { get; set; }
        public DateTime ModifiedDate { get; set; }

                public virtual ICollection<Product> ProductSizeList { get; set; }
                public virtual ICollection<Product> ProductWeightList { get; set; }
        }
}
```

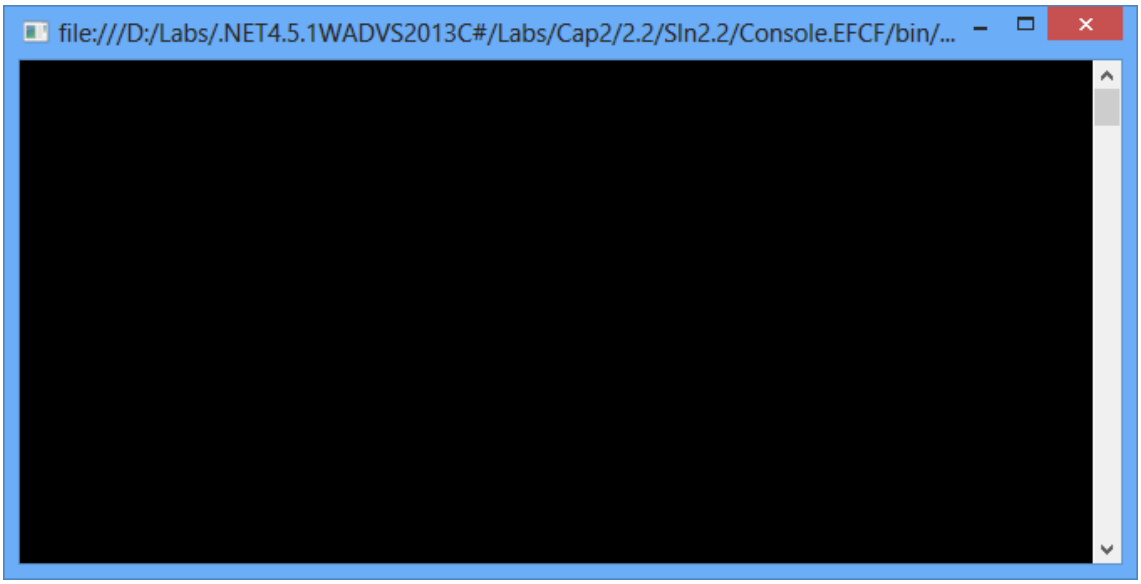
Para más información acerca de Code First Data Annotations:

http://msdn.microsoft.com/en-us/data/jj591583.aspx

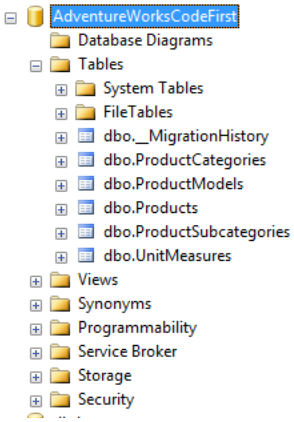
http://msdn.microsoft.com/en-us/data/jj679962.aspx

http://msdn.microsoft.com/en-us/data/jj819164

9. Establezca el proyecto **ConsoleApp.EFCF** como proyecto de inicio de la solución y ejecútelo



No aparece ningún texto pues nuestras tablas están vacías y no hay filas para recorrer en la tabla **Products**, puede verificar la creación de las tablas en su Consola de Administración de SQL Server en la base de datos especificada en el **App.Config**:



9. Un punto importante a considerar en este tipo de proyectos es que necesitaremos crear data de prueba a la base datos. EF Code-First cada vez que encuentre un cambio en nuestras clases de dominio re-creará la base de datos.
10. Empecemos agregando una clase al proyecto **AccesoDatos.EFCF** con el nombre **AdventureWorksCodeFirstInitializer.cs**
11. Ingrese el siguiente código en la clase recientemente creada para crear registros para las tablas creadas. Se ha configurado para que siempre que se ejecute la aplicación se borre y cree la base de datos (DropCreateDatabaseAlways).

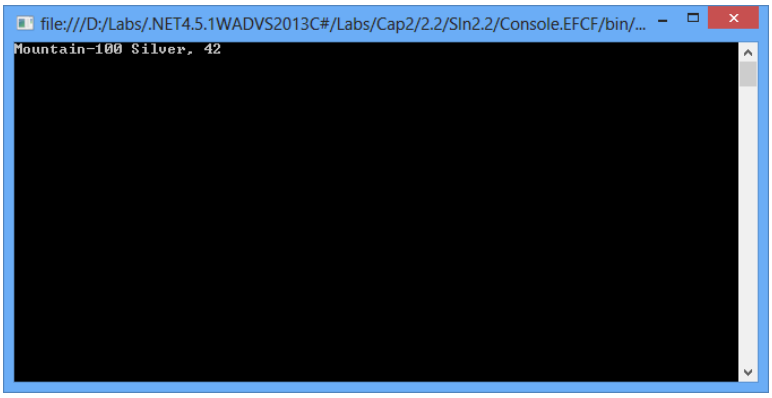
```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Data.Entity;
using Entidades.EFCF;

namespace AccesoDatos.EFCF
{
    public class AdventureWorksCodeFirstInitializer:
        DropCreateDatabaseAlways<AdventureWorksContext>
    {
        protected override void Seed(AdventureWorksContext context)
        {
```

```
            var productCategories = new List<ProductCategory>
            {
                new ProductCategory{Name = "Bikes", RowGuid = Guid.NewGuid(),
ModifiedDate = DateTime.Now}
            };
            productCategories.ForEach(pc => context.ProductCategories.Add(pc));
            context.SaveChanges();

            var productSubCategories = new List<ProductSubcategory>{
                new ProductSubcategory{ProductCategoryID = 1, Name = "Mountain Bikes",
                    RowGuid = Guid.NewGuid(), ModifiedDate = DateTime.Now},
                new ProductSubcategory{ProductCategoryID = 1, Name = "Road Bikes",
                    RowGuid = Guid.NewGuid(), ModifiedDate = DateTime.Now}
            };
            productSubCategories.ForEach(ps => context.ProductSubcategories.Add(ps));
            context.SaveChanges();

            var productModels = new List<ProductModel>{
                new ProductModel{Name = "Classic Vest", CatalogDescription = null,
Instructions = null,
                    RowGuid = Guid.NewGuid(), ModifiedDate = DateTime.Now}
            };
            productModels.ForEach(pm => context.ProductModels.Add(pm));
            context.SaveChanges();

            var unitMeasures = new List<UnitMeasure>(){
                new UnitMeasure{UnitMeasureCode = "BOX", Name = "Boxes", ModifiedDate =
DateTime.Now}
            };
            unitMeasures.ForEach(um => context.UnitMeasures.Add(um));
            context.SaveChanges();

            var products = new List<Product>
            {
                new Product{Name = "Mountain-100 Silver, 42", ProductNumber="BK-M82S-
42", MakeFlag = true, FinishedGoodsFlag = true,
                Color = "Silver", SafetyStockLevel = 1000, ReorderPoint = 75,
StandardCost = 1912.1544M, ListPrice = 3399.99M,
                Size = "42", SizeUnitMeasureCode = "CM", WeightUnitMeasureCode = "LB",
Weight = 20.77M, DaysToManufacture = 4,
                ProductLine = "M", Class = "H", Style = "U", ProductSubcategoryID = 1,
ProductModelID = 1, SellStartDate = DateTime.Now.AddDays(-3),
                SellEndDate = DateTime.Now.AddDays(4), DiscontinuedDate = null, RowGuid
= Guid.NewGuid(), ModifiedDate = DateTime.Now}
            };
            products.ForEach(p => context.Products.Add(p));
            context.SaveChanges();
        }
    }
}
```

12. Ahora necesitamos indicarle a EF que use la clase creada para crear data inicial a la BD creada. Dentro del archivo App.Config del proyecto ConsoleApp.EFCF agregar lo siguiente dentro de la sección entityFramework

```
<contexts>
      <context type="AccesoDatos.EFCF.AdventureWorksContext, AccesoDatos.EFCF">
        <databaseInitializer type="AccesoDatos.EFCF.AdventureWorksCodeFirstInitializer,
AccesoDatos.EFCF"></databaseInitializer>
      </context>
</contexts>
```

13. Ejecutar nuevamente el proyecto ConsoleApp.EFCF



14. Lo anterior indica que se ha creado la data inicial configurada, puede verificarlo por el Administrador de SQL Server
15. Si solo queremos que la base de datos se cree cuando el modelo haya cambiado, debemos modificar el archivo **AdventureWorksCodeFirstInitializer.cs**

```
public class AdventureWorksCodeFirstInitializer:
        DropCreateDatabaseIfModelChanges<AdventureWorksContext>
```

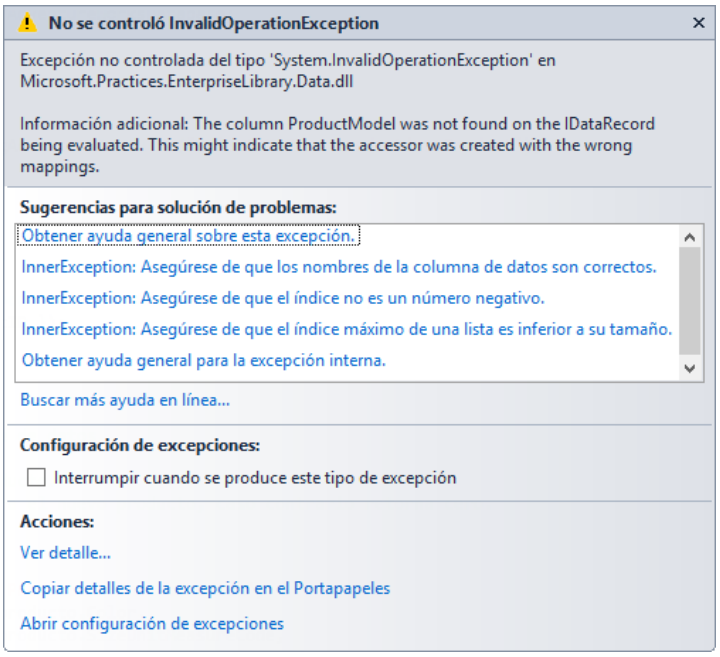
16. Para más información al respecto puede consultar http://www.asp.net/mvc/tutorials/getting-started-with-ef-using-mvc/creating-an-entity-framework-data-model-for-an-asp-net-mvc-application

Probando Enterprise Library Data Access Application Block

1. Similar a los pasos realizados para EF Database First y EF Code First, agregar un proyecto a la carpeta **DataAccessApplicationBlock** del tipo Aplicación de consola con el nombre **ConsoleApp.DAAB**
2. Agregar las referencias a los proyectos **Entidades.DAAB** y **LogicaNegocio.DAAB** al proyecto **ConsoleApp.DAAB**
3. En el archivo **App.config** del proyecto **AccesoDatos.DAAB** agregue la cadena de conexión `AdventureWorksConnection`.

```
<connectionStrings>
        <add name="AdventureWorksConnection" connectionString="data
        source=.\SQLEXPRESS2012;initial catalog=AdventureWorks2012;Integrated
        Security=True;" providerName="System.Data.SqlClient" />
</connectionStrings>
```

4. Repita los pasos 5 y 6 del ejemplo EF Database First.
5. Establezca el proyecto **ConsoleApp.DAAB** como proyecto de inicio de la solución y ejecútelo
6. Al ejecutar el proyecto aparecerá una excepción, pues no se están cargando ciertas propiedades de la clase **Product** en el método **SelectAll()** del **ProductDAO**

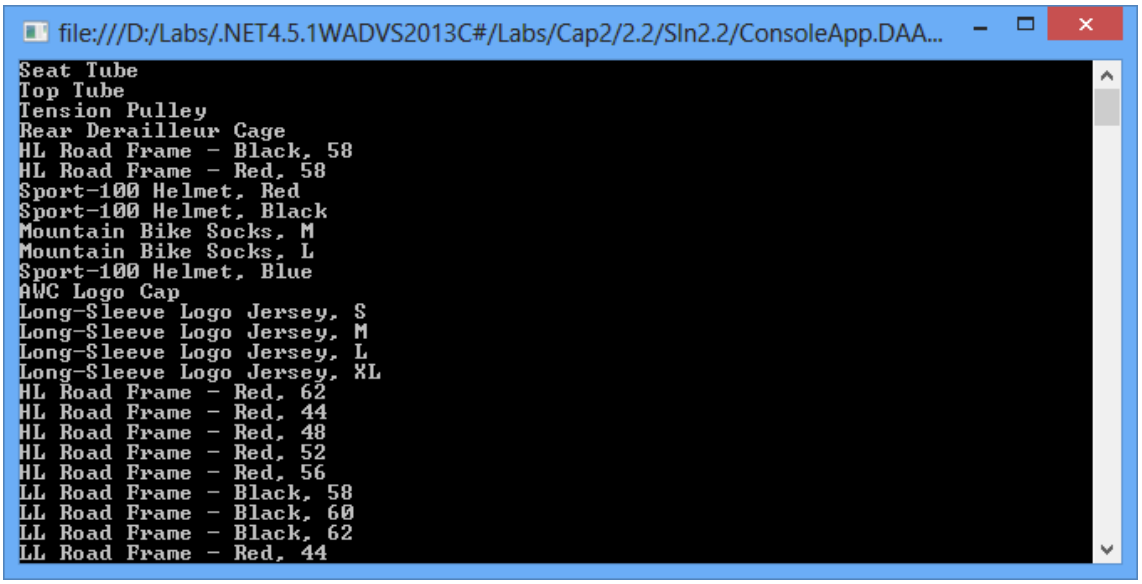


Esto se refiere a que las siguientes propiedades de la entidad **Product** no están siendo cargadas:

```
public ProductModel ProductModel { get; set; }
public ProductSubcategory ProductSubcategory { get; set; }
public UnitMeasure SizeUnitMeasure { get; set; }
public UnitMeasure WeightUnitMeasure { get; set; }
```

Esto es porque se ha copiado desde las clases generadas por Entity Framework, en este caso comentemos las 4 líneas de código.

7. Ejecute nuevamente la aplicación


file:///D:/Labs/.NET4.5.1WADVS2013C#/Labs/Cap2/2.2/Sln2.2/ConsoleApp.DAA...
Seat Tube
Top Tube
Tension Pulley
Rear Derailleur Cage
HL Road Frame - Black, 58
HL Road Frame - Red, 58
Sport-100 Helmet, Red
Sport-100 Helmet, Black
Mountain Bike Socks, M
Mountain Bike Socks, L
Sport-100 Helmet, Blue
AWC Logo Cap
Long-Sleeve Logo Jersey, S
Long-Sleeve Logo Jersey, M
Long-Sleeve Logo Jersey, L
Long-Sleeve Logo Jersey, XL
HL Road Frame - Red, 62
HL Road Frame - Red, 44
HL Road Frame - Red, 48
HL Road Frame - Red, 52
HL Road Frame - Red, 56
LL Road Frame - Black, 58
LL Road Frame - Black, 60
LL Road Frame - Black, 62
LL Road Frame - Red, 44