Entity Framework – Model First Approach – Part 6

In this tutorial we'll learn Model First approach. In this approach, we create a model (edmx) file first and then we generate SQL scripts from it to create our database objects (e.g. tables)

This tutorial is prepared with Visual Studio 2013 + SQL Server.

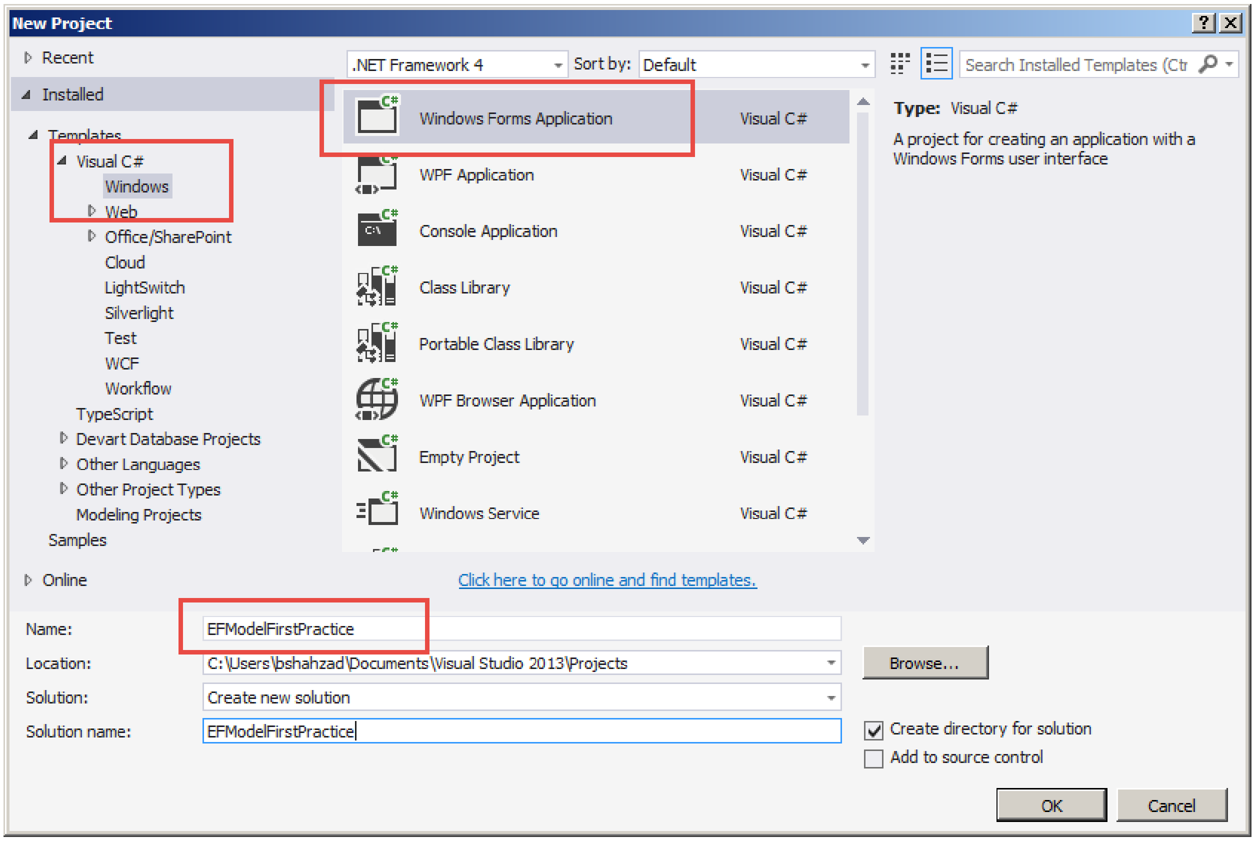
Version	**Last updated**	**Comments**	**Modified By**
V1.0	02-05-2016		Bilal Shahzad

Introduction

1- Here are quick steps to complete this exercise
   a. Create a new Windows Application.
   b. Adding reference of “Entity Framework” reference using “NuGet”.
   c. Add New Item => Data (ADO.NET Data Entity Model)
   d. Add New Entity in EDMX.
   e. Add a New Property in created entity.
   f. Right click on designer => Generate Database from Model.
   g. Provide Server Detail & Name of database (which we want to create).
   h. Ready to use

Step by Step Walkthrough

1- Create a New Windows Application in Visual Studio



.NET Framework 4

Sort by: Default

Type: Visual C#

A project for creating an application with a Windows Forms user interface

Name: EFModelFirstPractice

Location: C:\Users\bshahzad\Documents\Visual Studio 2013\Projects

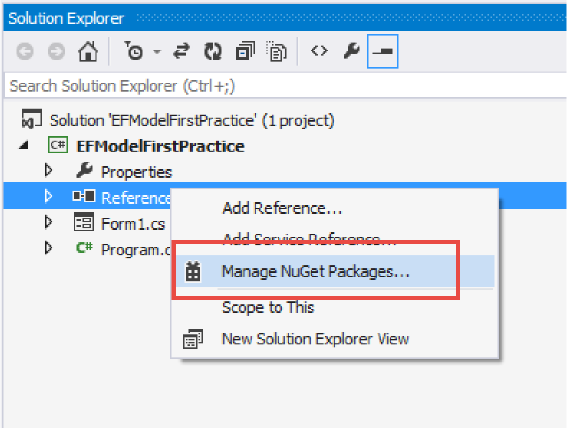
Solution: Create new solution

Solution name: EFModelFirstPractice

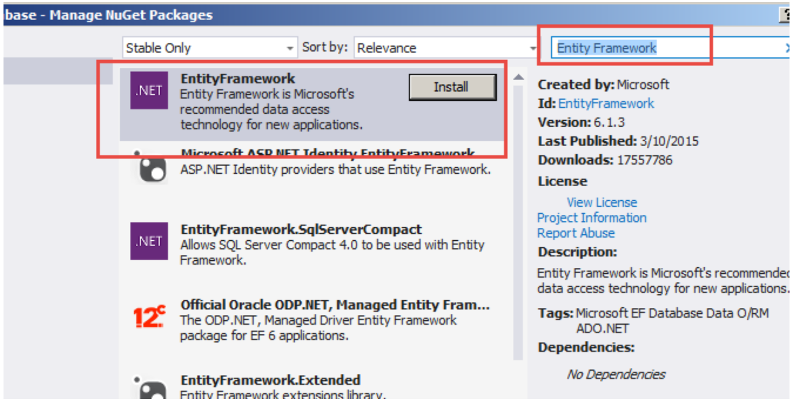
☑ Create directory for solution

□ Add to source control

2- Right click on “References” and select “Manage NuGet Packages”



3- Find “Entity Framework” and “Install” it.



EntityFramework
Entity Framework is Microsoft's recommended data access technology for new applications.

Microsoft.ASP.NET.Identity.EntityFramework
ASP.NET Identity providers that use Entity Framework.

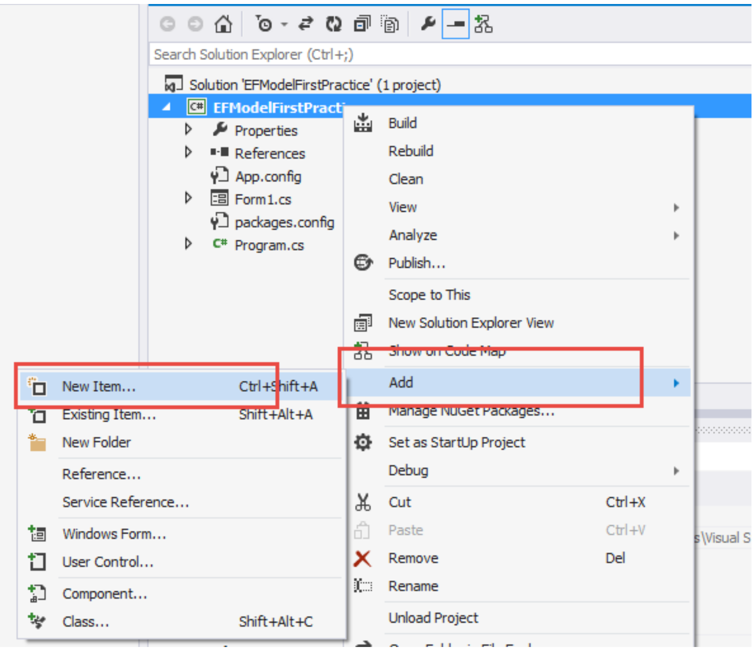
EntityFramework.SqlServerCompact
Allows SQL Server Compact 4.0 to be used with Entity Framework.

Official Oracle ODP.NET, Managed Entity Fram…
The ODP.NET, Managed Driver Entity Framework package for EF 6 applications.

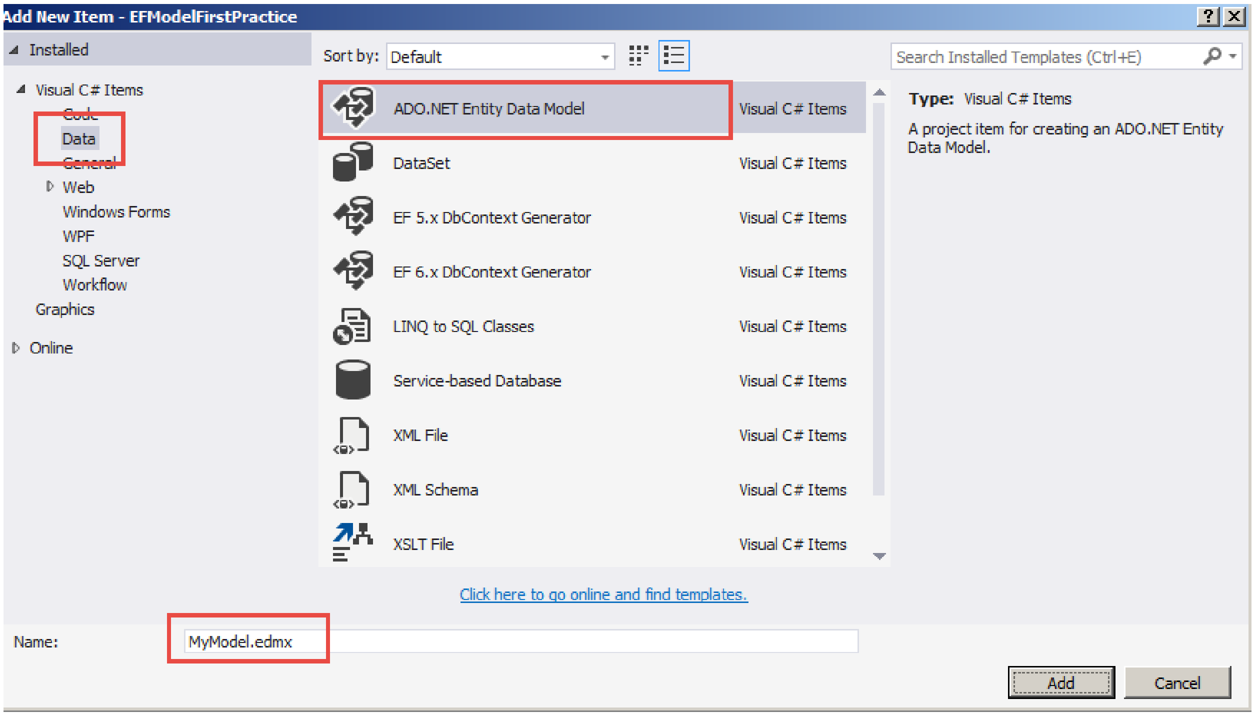
EntityFramework.Extended
Entity Framework extensions library.

Created by: Microsoft
Id: EntityFramework
Version: 6.1.3
Last Published: 3/10/2015
Downloads: 17557786
License
View License
Project Information
Report Abuse
Description:
Entity Framework is Microsoft's recommended data access technology for new applications.
Tags: Microsoft EF Database Data O/RM ADO.NET
Dependencies:
No Dependencies

4- Right click on your project => Add => New Item.



5- Choose “Data” from left panel. And then choose “ADO.NET Entity Data Model” from right panel. Give it meaningful name (e.g. MyModel.edmx) and click “Add”.

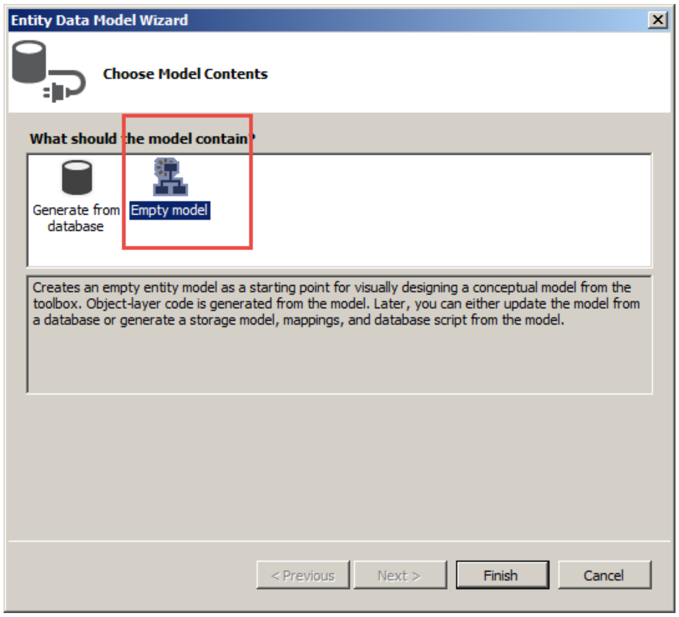

Add New Item - EFModelFirstPractice
Installed
Visual C# Items
Code
Data
General
Web
Windows Forms
WPF
SQL Server
Workflow
Graphics
Online
Sort by: Default
Search Installed Templates (Ctrl+E)
ADO.NET Entity Data Model — Visual C# Items
DataSet — Visual C# Items
EF 5.x DbContext Generator — Visual C# Items
EF 6.x DbContext Generator — Visual C# Items
LINQ to SQL Classes — Visual C# Items
Service-based Database — Visual C# Items
XML File — Visual C# Items
XML Schema — Visual C# Items
XSLT File — Visual C# Items
Click here to go online and find templates.
Add
Cancel

Type: Visual C# Items

A project item for creating an ADO.NET Entity Data Model.

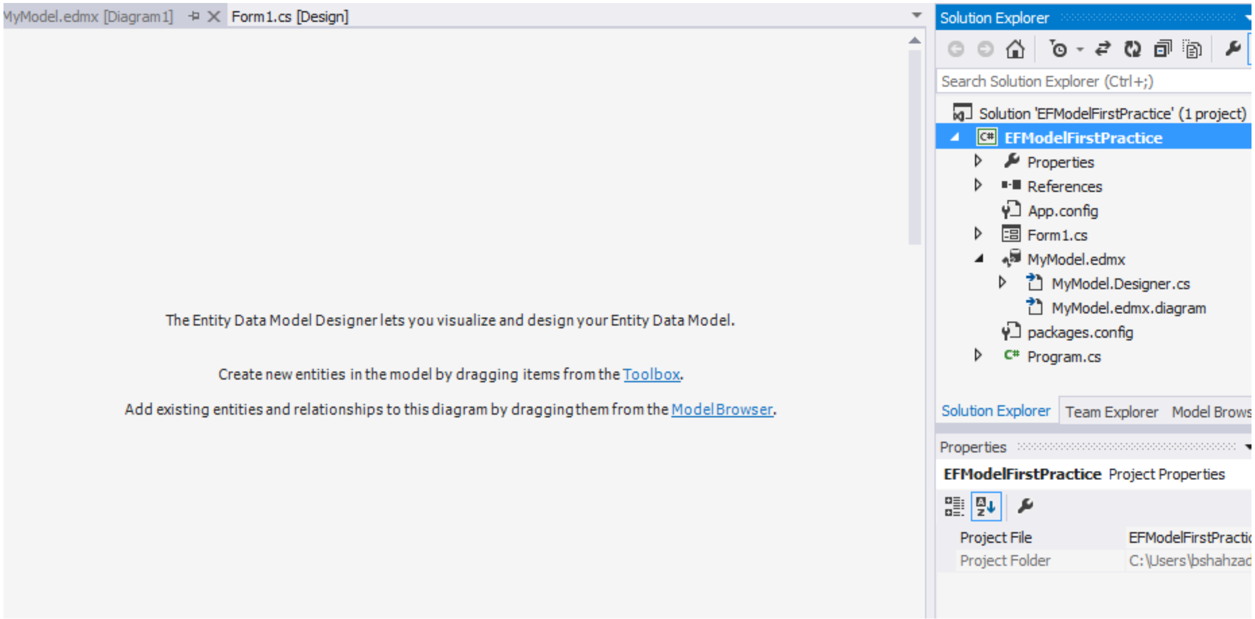
Name: MyModel.edmx

6- Choose “Empty Model”.



Creates an empty entity model as a starting point for visually designing a conceptual model from the toolbox. Object-layer code is generated from the model. Later, you can either update the model from a database or generate a storage model, mappings, and database script from the model.

7- We can see that a file name “MyModel.emx” is added. If you double click on it, you will see a designer window on left side.

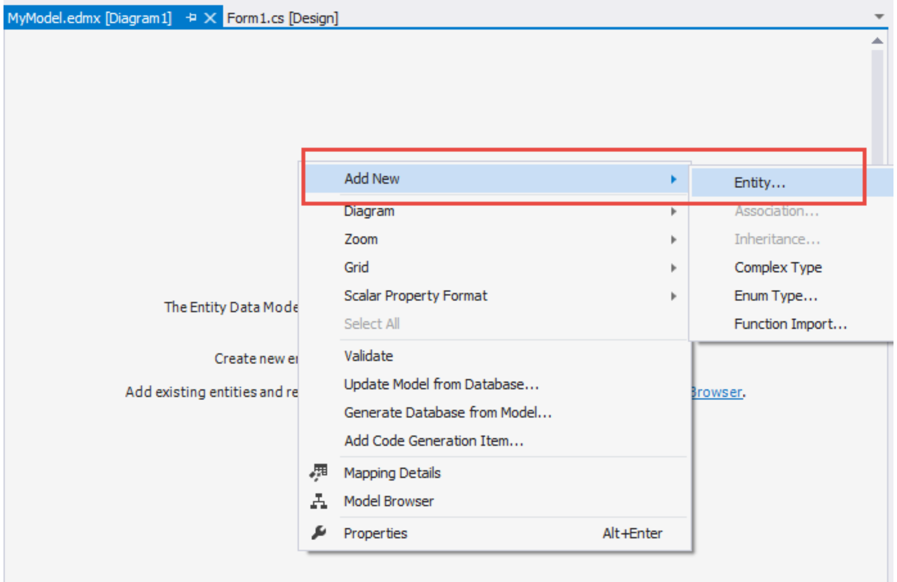


The Entity Data Model Designer lets you visualize and design your Entity Data Model.

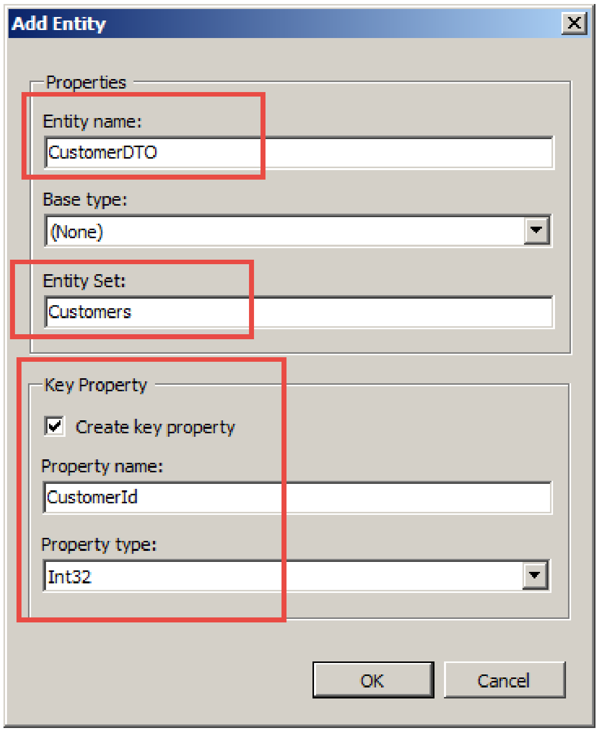
Create new entities in the model by dragging items from the Toolbox.

Add existing entities and relationships to this diagram by dragging them from the Model Browser.

8- Right click on empty designer and choose “Add New => Entity” to add an entity (i.e. DTO)



9- Provide the detail of your entity (as shown in screenshot below). “Entity Set” is the name of your DbSet (which you normally provide in context class.



Add Entity

Properties

Entity name: CustomerDTO

Base type: (None)

Entity Set: Customers

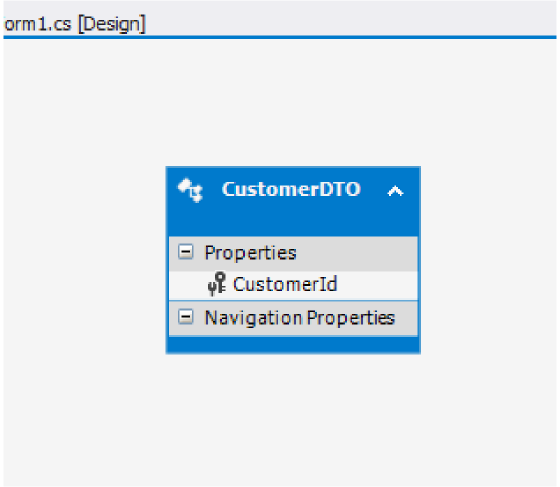
Key Property

☑ Create key property

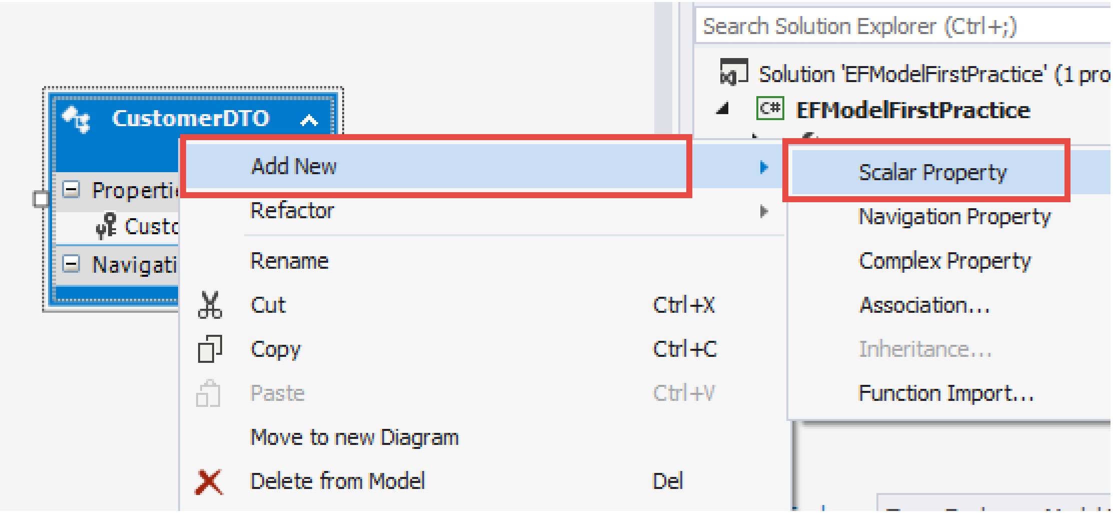
Property name: CustomerId

Property type: Int32

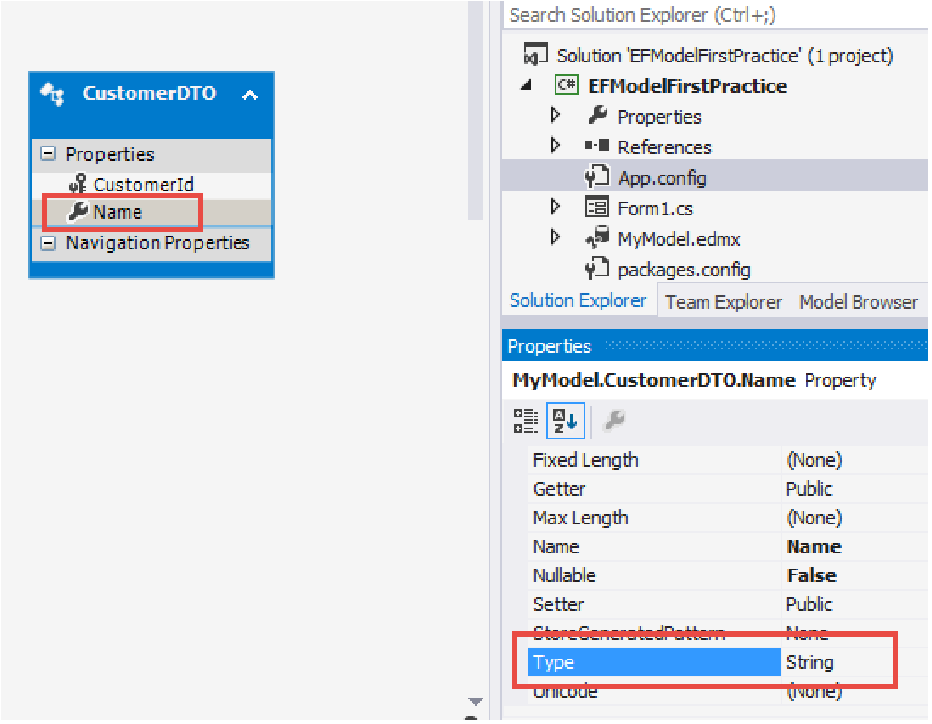
10- You will see an entity added on the designer. You may add more properties to it.



11- To add a new property, Right click on “CustomerDTO” entity and choose “Add New => Scalar Property”



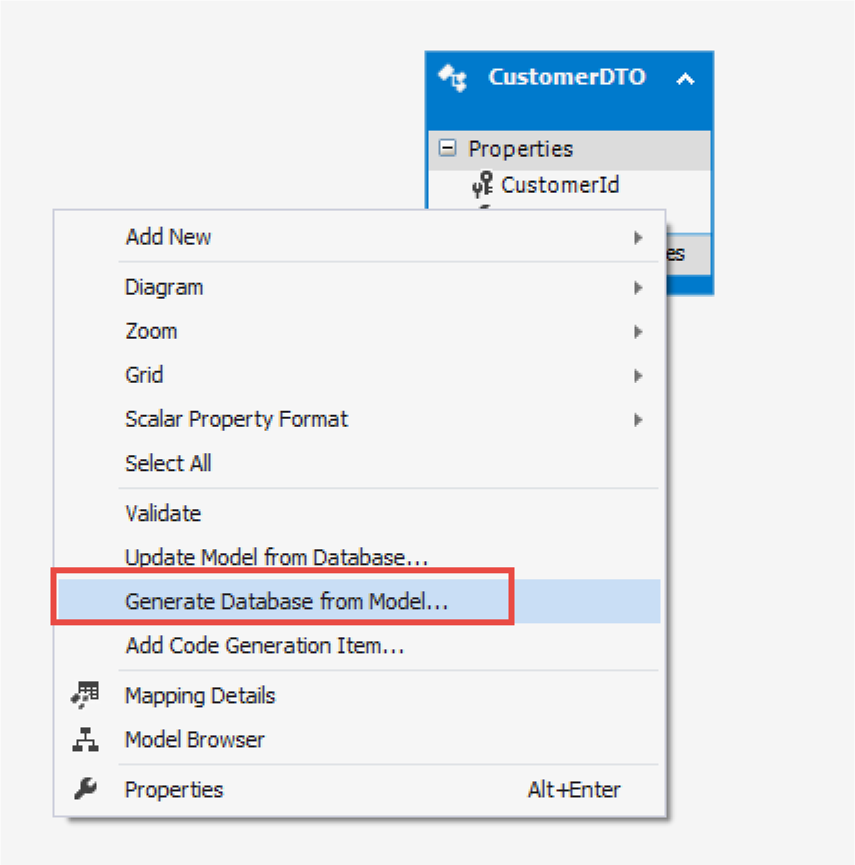
12- Set its value to “Name”. In “Properties Window”, you can change its data type. It should be string for our new property.



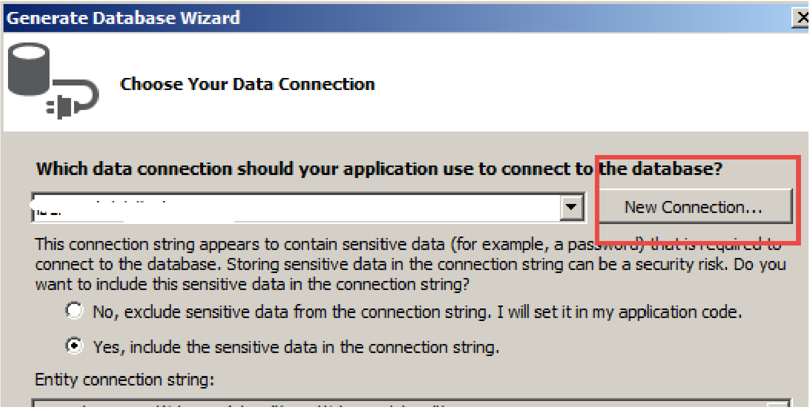
MyModel.CustomerDTO.Name Property

Fixed Length	(None)
Getter	Public
Max Length	(None)
Name	**Name**
Nullable	**False**
Setter	Public
StoreGeneratedPattern	None
Type	String
Unicode	(None)

13- Once your entities are created and your EDMX file is ready. You can create database from your model. To do so, Right click on empty space on your “edmx” file and choose “Generate Database from Model”.



14- It will ask for a database connection, Choose “New Connection” option.



Which data connection should your application use to connect to the database?

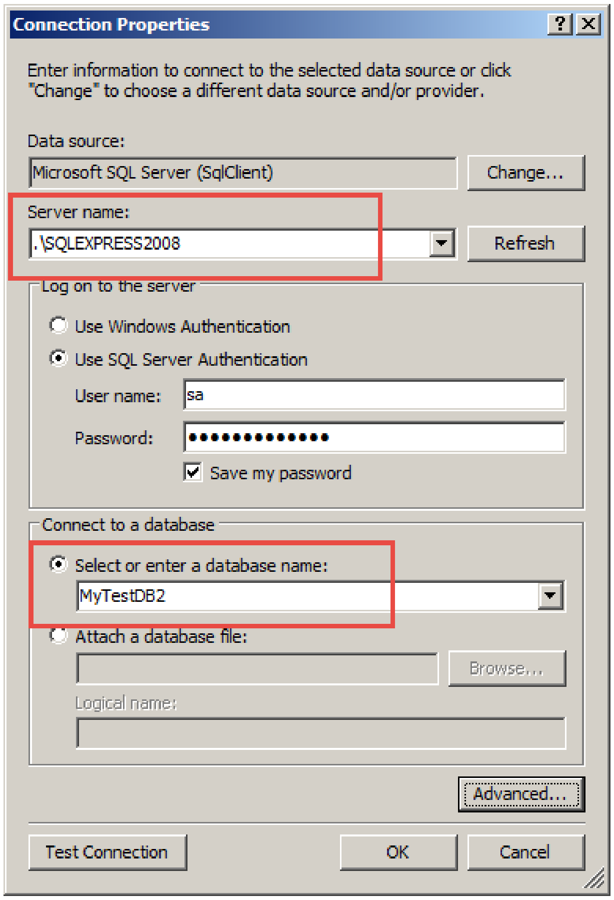
This connection string appears to contain sensitive data (for example, a password) that is required to connect to the database. Storing sensitive data in the connection string can be a security risk. Do you want to include this sensitive data in the connection string?

○ No, exclude sensitive data from the connection string. I will set it in my application code.

● Yes, include the sensitive data in the connection string.

Entity connection string:

15- Provide Server Name, Authentication Mechanism and the database name (if this database doesn’t exist, it will be created).



Connection Properties

Enter information to connect to the selected data source or click "Change" to choose a different data source and/or provider.

Data source: Microsoft SQL Server (SqlClient)

Server name: .\SQLEXPRESS2008

Log on to the server

○ Use Windows Authentication

● Use SQL Server Authentication

User name: sa

Password: ••••••••••••••

☑ Save my password

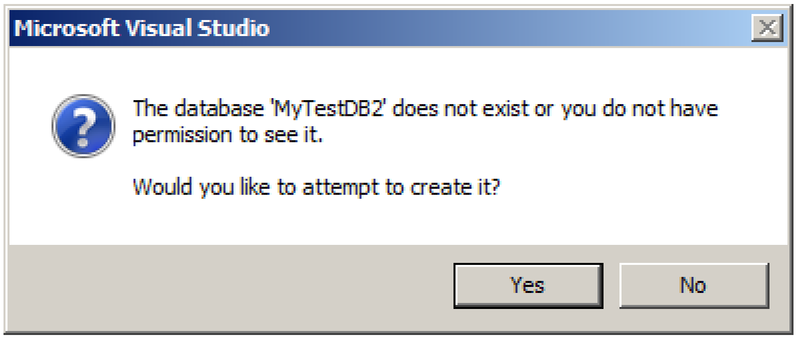
Connect to a database

● Select or enter a database name: MyTestDB2

○ Attach a database file:

Logical name:

16- When you will click on “OK”, it will verify if database with the name entered exists or not.

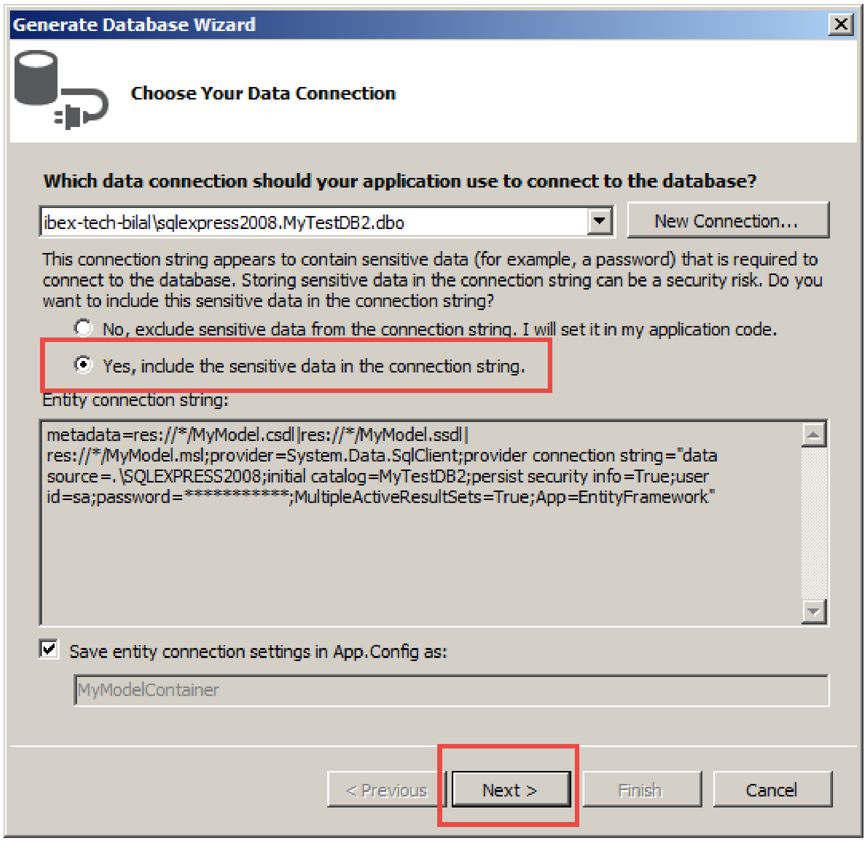


Microsoft Visual Studio

The database 'MyTestDB2' does not exist or you do not have permission to see it.

Would you like to attempt to create it?

17- It will show you the detail. Choose “Yes”. Here at the bottom, we can see the name of our connection string (which will be also the name of our Context class). Click “Next”.



Which data connection should your application use to connect to the database?

ibex-tech-bilal\sqlexpress2008.MyTestDB2.dbo

This connection string appears to contain sensitive data (for example, a password) that is required to connect to the database. Storing sensitive data in the connection string can be a security risk. Do you want to include this sensitive data in the connection string?

○ No, exclude sensitive data from the connection string. I will set it in my application code.

● Yes, include the sensitive data in the connection string.

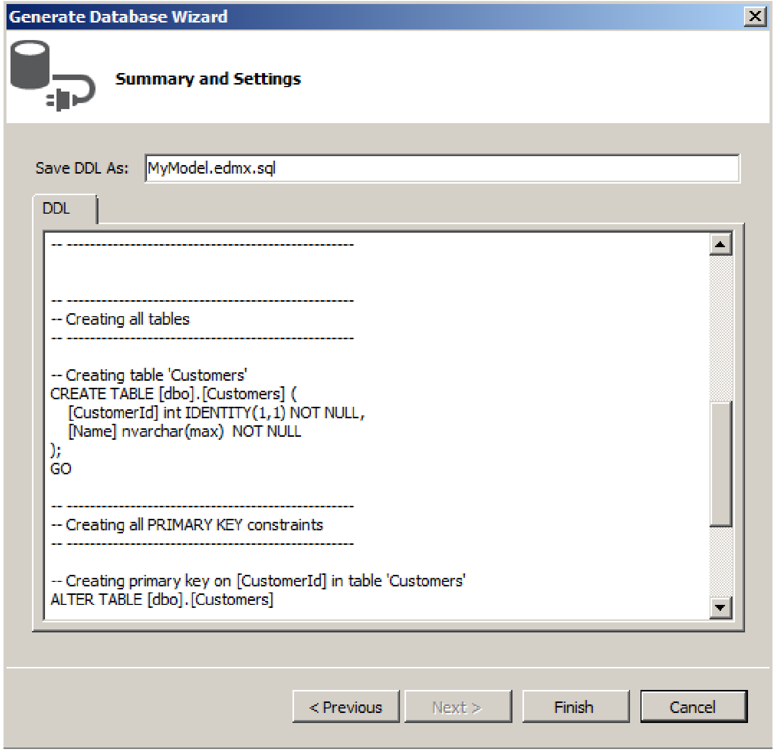
Entity connection string:

```
metadata=res://*/MyModel.csdl|res://*/MyModel.ssdl|
res://*/MyModel.msl;provider=System.Data.SqlClient;provider connection string="data
source=.\SQLEXPRESS2008;initial catalog=MyTestDB2;persist security info=True;user
id=sa;password=***********;MultipleActiveResultSets=True;App=EntityFramework"
```

☑ Save entity connection settings in App.Config as:

MyModelContainer

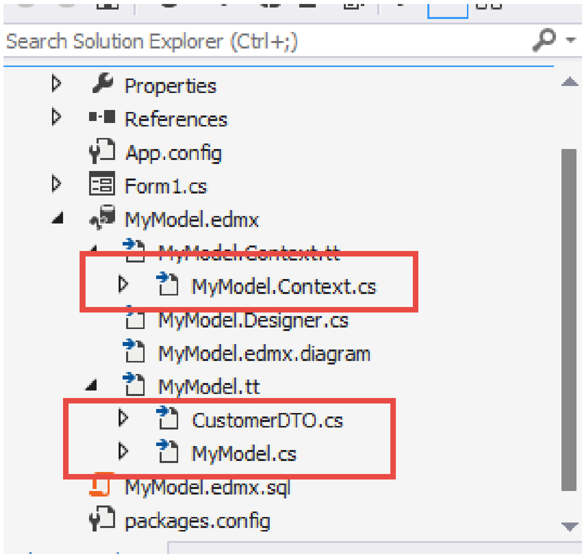
18- It will generate the required scripts to create the database + tables. You can copy the script from it and paste it in SQL Server query window.



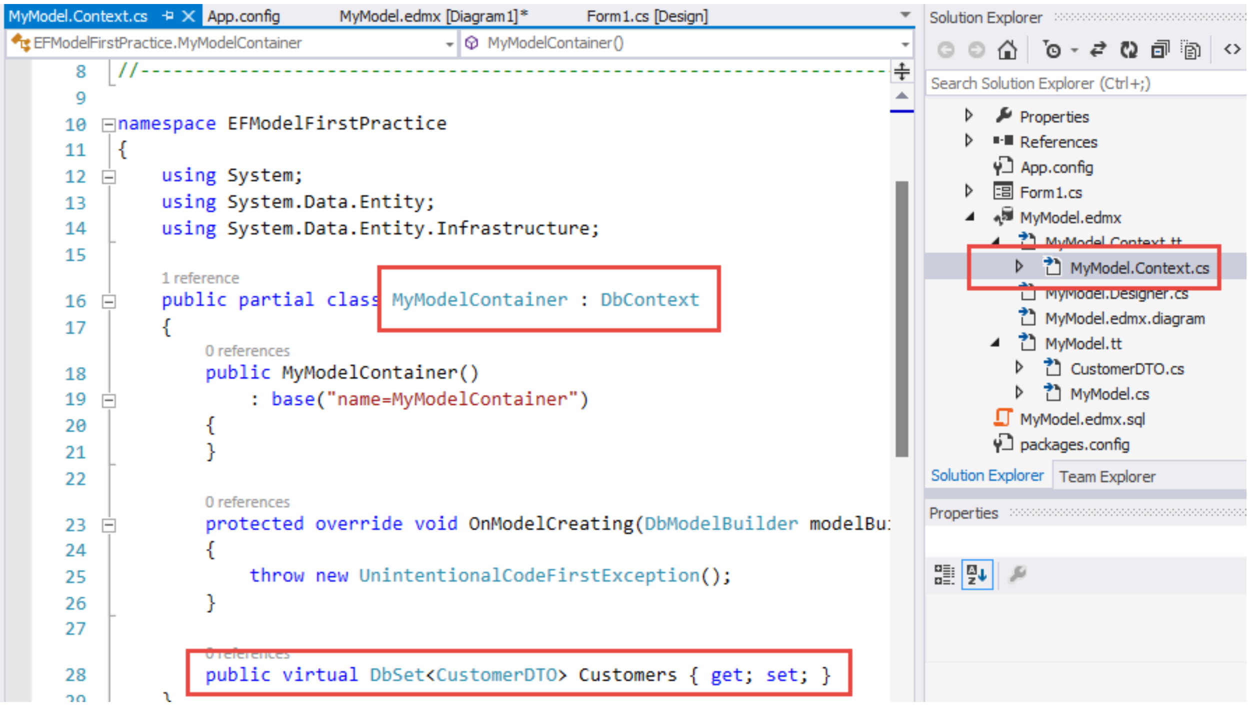
Save DDL As: MyModel.edmx.sql

```
-- --------------------------------------------------



-- --------------------------------------------------
-- Creating all tables
-- --------------------------------------------------

-- Creating table 'Customers'
CREATE TABLE [dbo].[Customers] (
    [CustomerId] int IDENTITY(1,1) NOT NULL,
    [Name] nvarchar(max)  NOT NULL
);
GO

-- --------------------------------------------------
-- Creating all PRIMARY KEY constraints
-- --------------------------------------------------

-- Creating primary key on [CustomerId] in table 'Customers'
ALTER TABLE [dbo].[Customers]
```

19- When you will click “Finish” in above step, you will see some changes in your solution explorer. You can see that a new context file is created. Also CutomerDTO file is created. In Code first approach, we’ve created these classes manually.

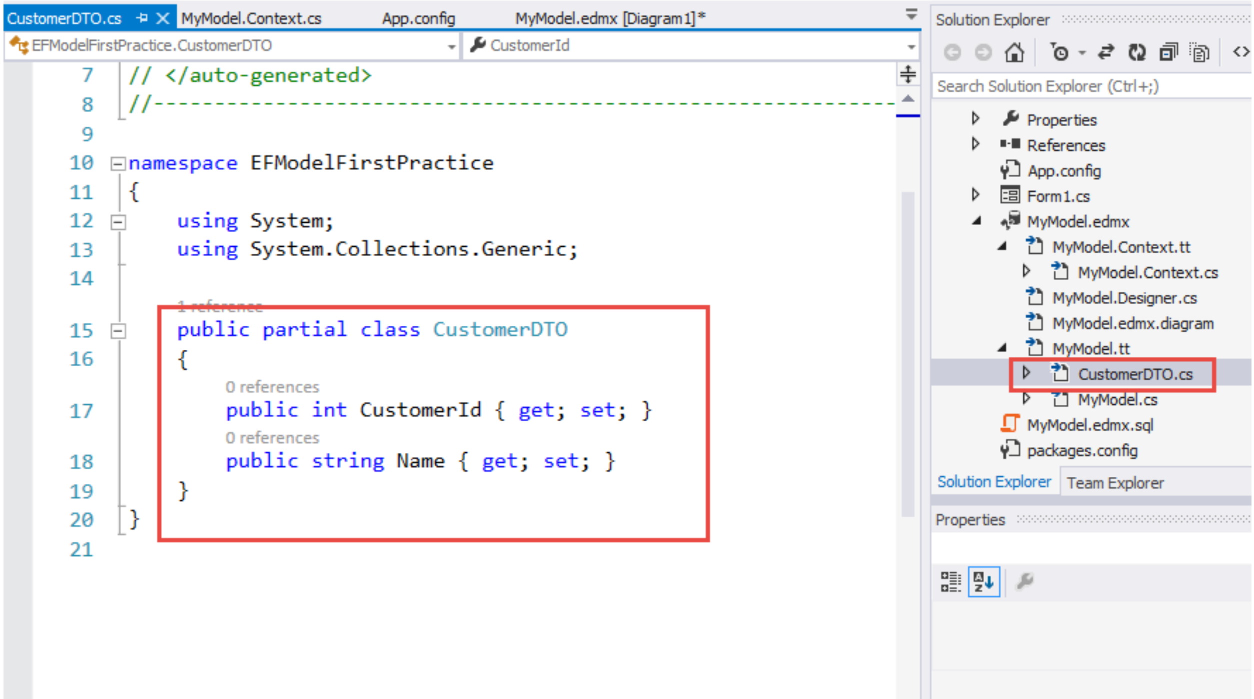


20- If you open “MyModel.Context.cs” file, you can see a context class and DbSet.



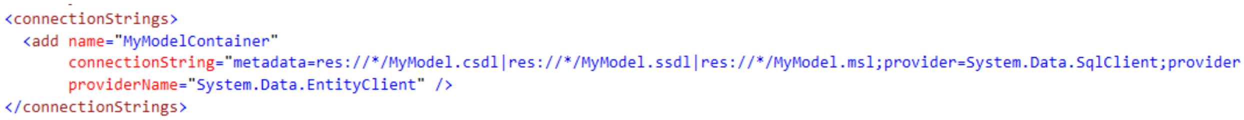
```
//------------------------------------------------------------------------------

namespace EFModelFirstPractice
{
    using System;
    using System.Data.Entity;
    using System.Data.Entity.Infrastructure;

    public partial class MyModelContainer : DbContext
    {
        public MyModelContainer()
            : base("name=MyModelContainer")
        {
        }

        protected override void OnModelCreating(DbModelBuilder modelBu
        {
            throw new UnintentionalCodeFirstException();
        }

        public virtual DbSet<CustomerDTO> Customers { get; set; }
    }
```

21- If you open “CustomerDTO.cs” file, you can see a DTO class is created.



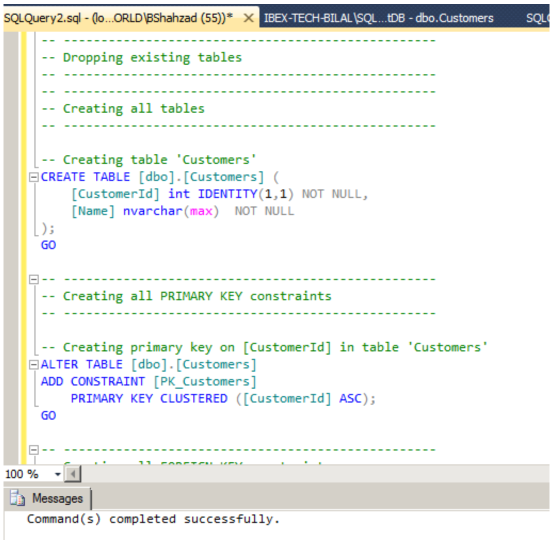
```
// </auto-generated>
//------------------------------------------------------------------------------

namespace EFModelFirstPractice
{
    using System;
    using System.Collections.Generic;

    1 reference
    public partial class CustomerDTO
    {
        0 references
        public int CustomerId { get; set; }
        0 references
        public string Name { get; set; }
    }
}
```

22- If you check “app.config”, you can see a connection string is added. This connection string also contains more data than normal connection string. We know that EDMX file is collection of three major components (CSDL, SSDL, MSL). You don’t need to make any change in this connection string (as it is generated by using the information you provided in Step-15 above.



```
<connectionStrings>
  <add name="MyModelContainer"
       connectionString="metadata=res://*/MyModel.csdl|res://*/MyModel.ssdl|res://*/MyModel.msl;provider=System.Data.SqlClient;provider
       providerName="System.Data.EntityClient" />
</connectionStrings>
```

23- Now if you run the queries (generated in step 18) in SQL Server, you will see that database + tables will be created.

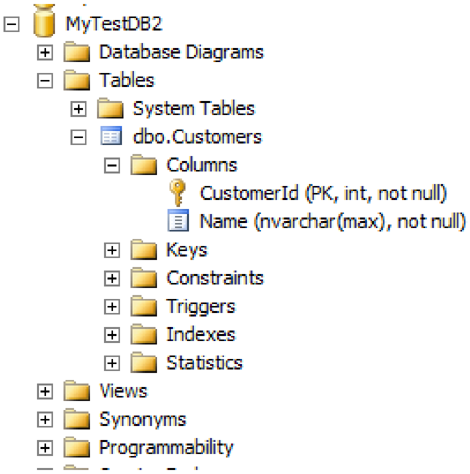


SQLQuery2.sql - (lo...ORLD\BShahzad (55))*
IBEX-TECH-BILAL\SQL...tDB - dbo.Customers
100 %
Messages

```
-- --------------------------------------------------
-- Dropping existing tables
-- --------------------------------------------------
-- --------------------------------------------------
-- Creating all tables
-- --------------------------------------------------

-- Creating table 'Customers'
CREATE TABLE [dbo].[Customers] (
    [CustomerId] int IDENTITY(1,1) NOT NULL,
    [Name] nvarchar(max)  NOT NULL
);
GO

-- --------------------------------------------------
-- Creating all PRIMARY KEY constraints
-- --------------------------------------------------

-- Creating primary key on [CustomerId] in table 'Customers'
ALTER TABLE [dbo].[Customers]
ADD CONSTRAINT [PK_Customers]
    PRIMARY KEY CLUSTERED ([CustomerId] ASC);
GO

-- --------------------------------------------------
```

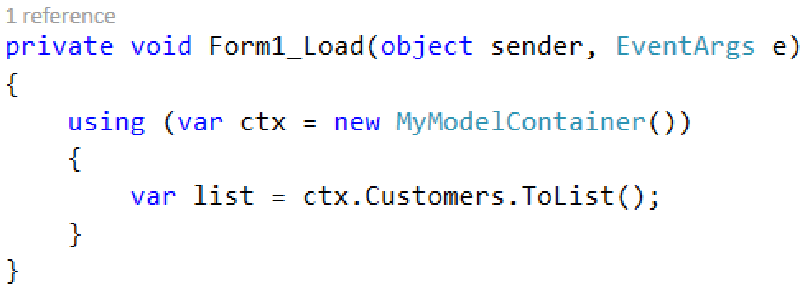
```
Command(s) completed successfully.
```

24- Here is the new database + table.



- MyTestDB2
  - Database Diagrams
  - Tables
    - System Tables
    - dbo.Customers
      - Columns
        - CustomerId (PK, int, not null)
        - Name (nvarchar(max), not null)
      - Keys
      - Constraints
      - Triggers
      - Indexes
      - Statistics
  - Views
  - Synonyms
  - Programmability

25- Now use of “Context” class & “Entities” classes is same as we’ve seen with “Code First” Approach.



```
1 reference
private void Form1_Load(object sender, EventArgs e)
{
    using (var ctx = new MyModelContainer())
    {
        var list = ctx.Customers.ToList();
    }
}
```