

The background features a dark grey horizontal band across the middle. Above and below this band are light grey areas containing stylized circuit traces and circular gear-like patterns. Four solid black circles are positioned along the top circuit traces, and several more are along the bottom traces.

Entity Framework

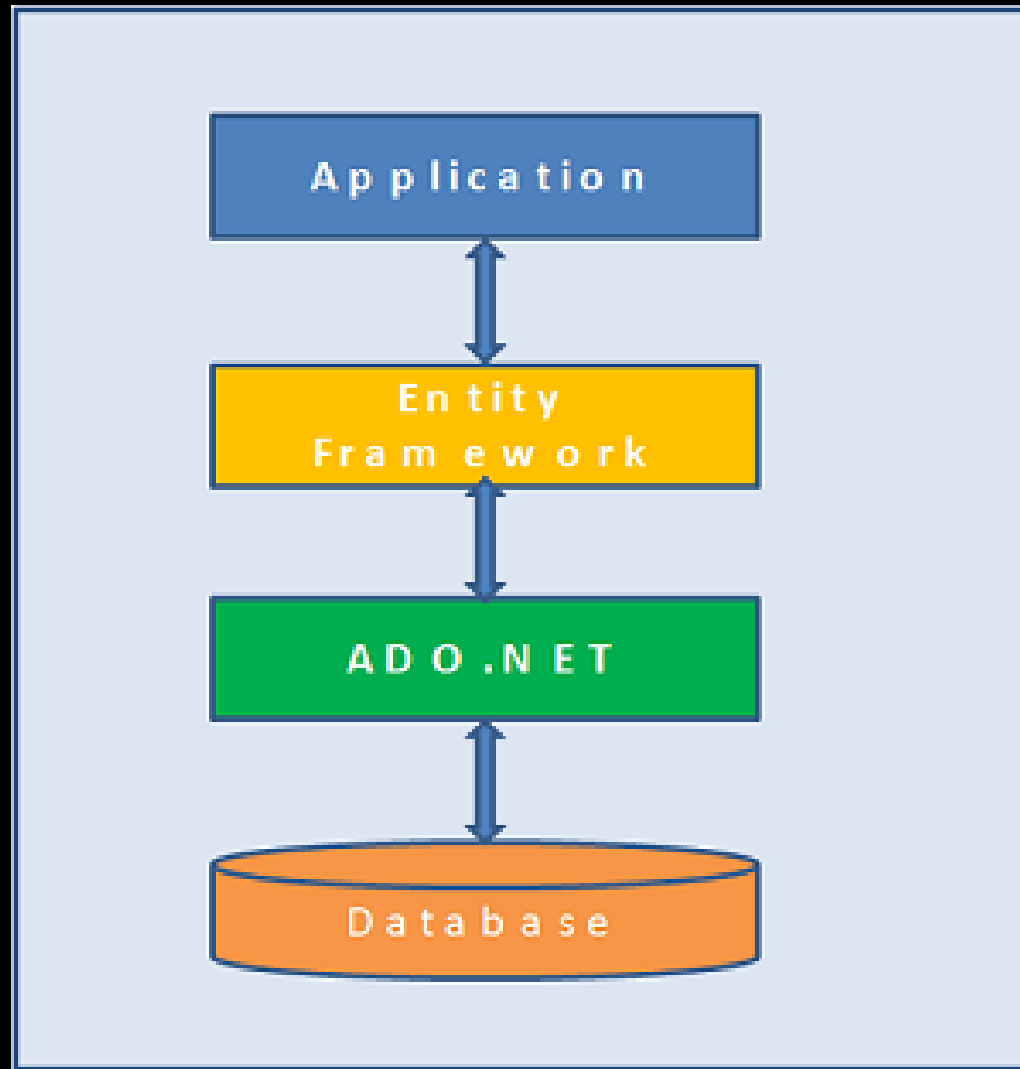
ASP.NET MVC

Entity Framework

- Is an ORM Framework (Object Relational Mapper): translates domain classes into rows and columns in the database tables.
- Sits between your application and the data store.
- The applications use the Entity framework API for the database related operations. The Entity Framework maps all the database related operations to the database.

Entity Framework

The EF
uses ADO.NET
to perform the
operations on
the database



Benefits

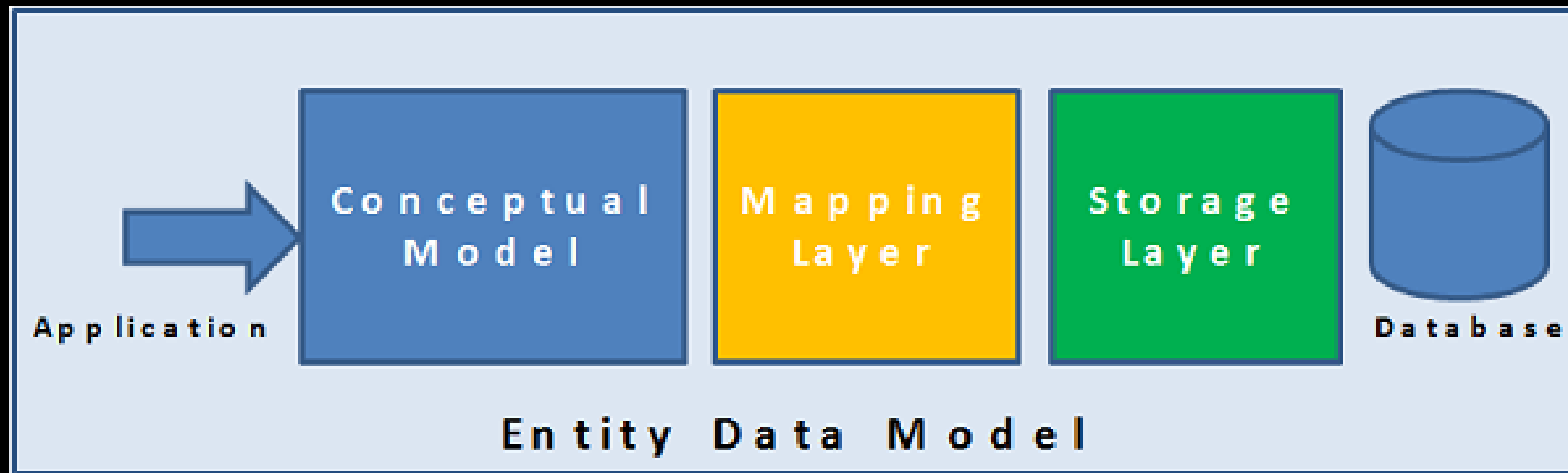
- Developers spend lots of time coding the plumbing required to save the data to the data store. EF **reduces** this time, hence the developers can spend time actually building the application.
- Developers can work against **domain specific objects** such as employee and employee address and need not to worry about how and where these data is stored.
- Applications are now easier to maintain as the amount of code is reduced. Reduced code also increase the productivity.
- You don't have to write SQL queries. EF will take care of that. It is smart enough to **know the associates between tables and generates the join queries.**
- Advanced relationships like inheritance, associations can be set up between domain models to suit the need of the application.

Terms

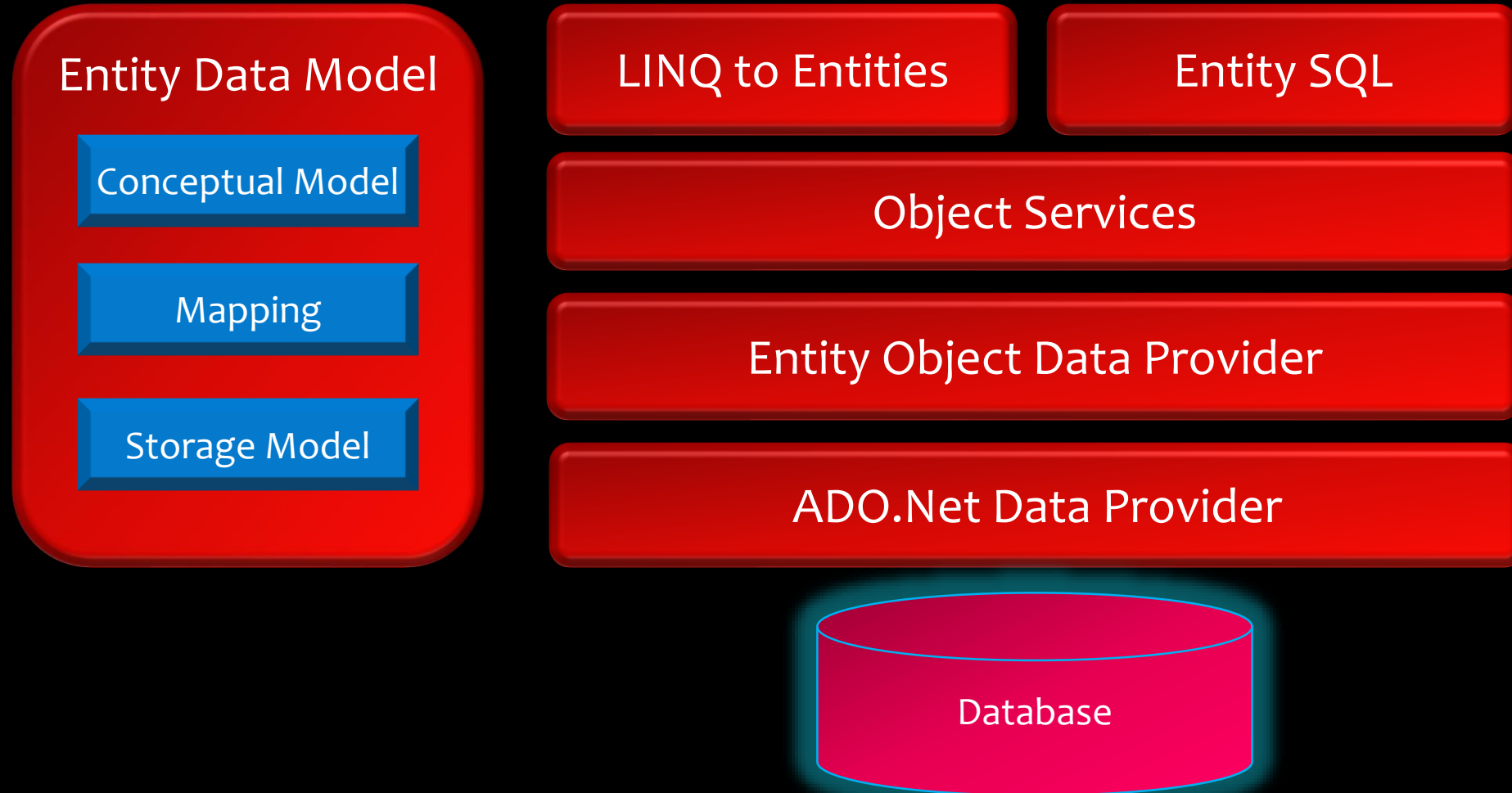
- **Entity:** The Entity is defined as an object with independent existence. For example, an employee in working for a company is an entity.
- **Properties:** Each Entity has properties or attributes similar to class. For example, in the case of employee entity, his name, joining date and address are Properties. The Properties can be int, string, collections or of a complex type.
- **Entity Type:** The Entity type is a collection of entities that share the common properties. It represents a unique object in your domain model. The Entity is a single instance of Entity Type.

Entity Data Model

Entity Data Model (EDM) is the heart of our Entity framework. The Entity Data model describes the Conceptual model of our domain objects. It enables you to create strongly typed domain classes (Entity types). It defines associations between these domain classes. It then maps these classes to the database Schema.

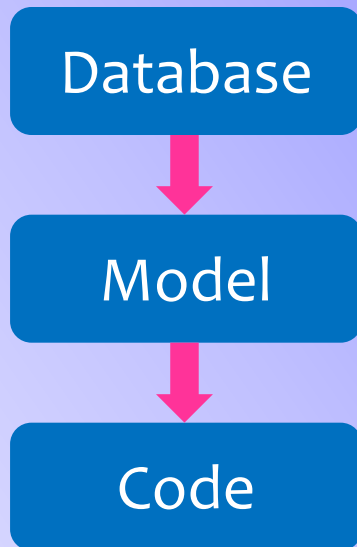


Entity Framework Architecture

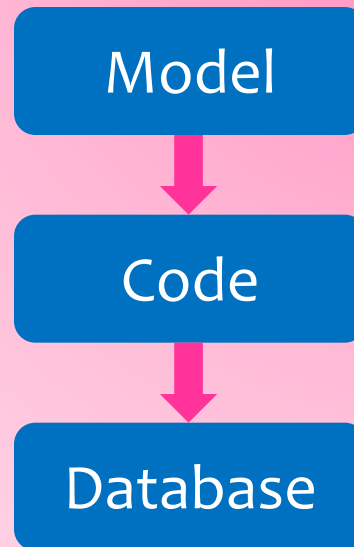


Entity Framework

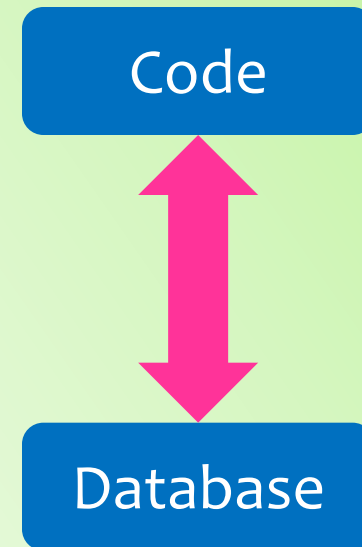
Database First

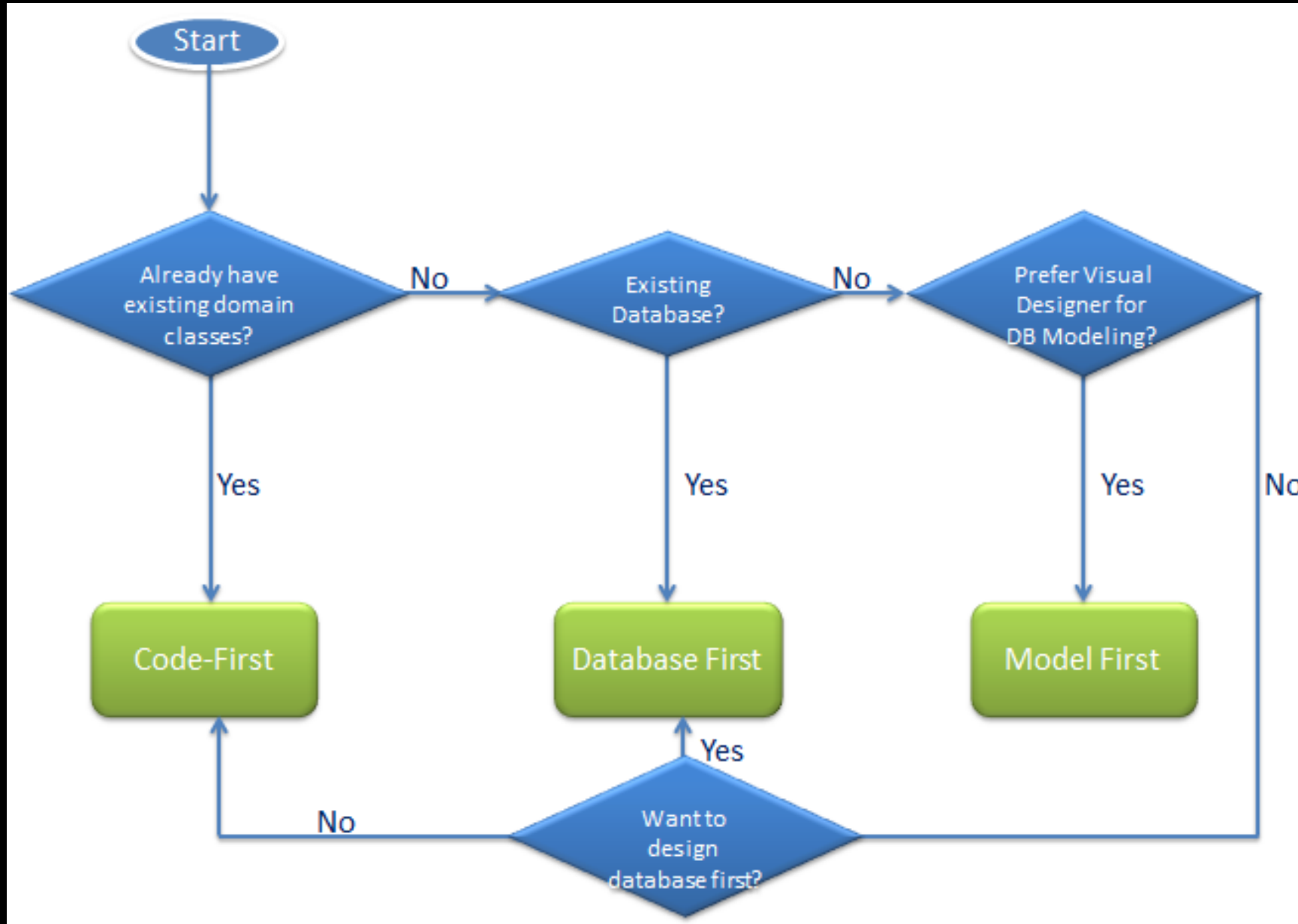


Model First



Code First





Database
First?

Model
First?

Code
First?

Model First

- In this approach database models are created First. The models are created using EDMX (Entity Data Model Designer) Designer.
- Entity Framework then uses the .EDMX, which is generated by the tool to generate the database from the model.

Database First

- In this approach EDMX is created from the existing database. This is the reverse of the model first approach where models are created first and database later. The Entity Data Model can be updated whenever database schema changes. Also, database-first approach supports stored procedure, view, etc.
- Database first is used if you have Database is developed separately or if you have an existing DB. Any changes made to the database can be easily updated in the model from the database.

Code First

- Code first gives full control over the code to the developer. There is no auto generated code as in the case of the database first or mode first.
- In code first you are more responsible for the database schema. You have to configure the relationships, constraints, and associations between entities.
- Adding a custom field in auto generated models, you may need to extend the model class. This is not a problem in the code first as you can add it to the existing class.
- Code first can be used in both existing or new databases. In case of an existing database, you may have to manually code model classes to match the database. The database first approach will create all the domain classes for you.

- Database first/Model first is going to be retired in Entity framework Version 7.
- In the EF 7 .EDMX file is discarded. Hence code first workflow is the only way going forward.

Conclusion

The Entity framework allows us to create the model directly using the code, which is called Code First.

DEMO

ASP.NET MVC Entity Framework

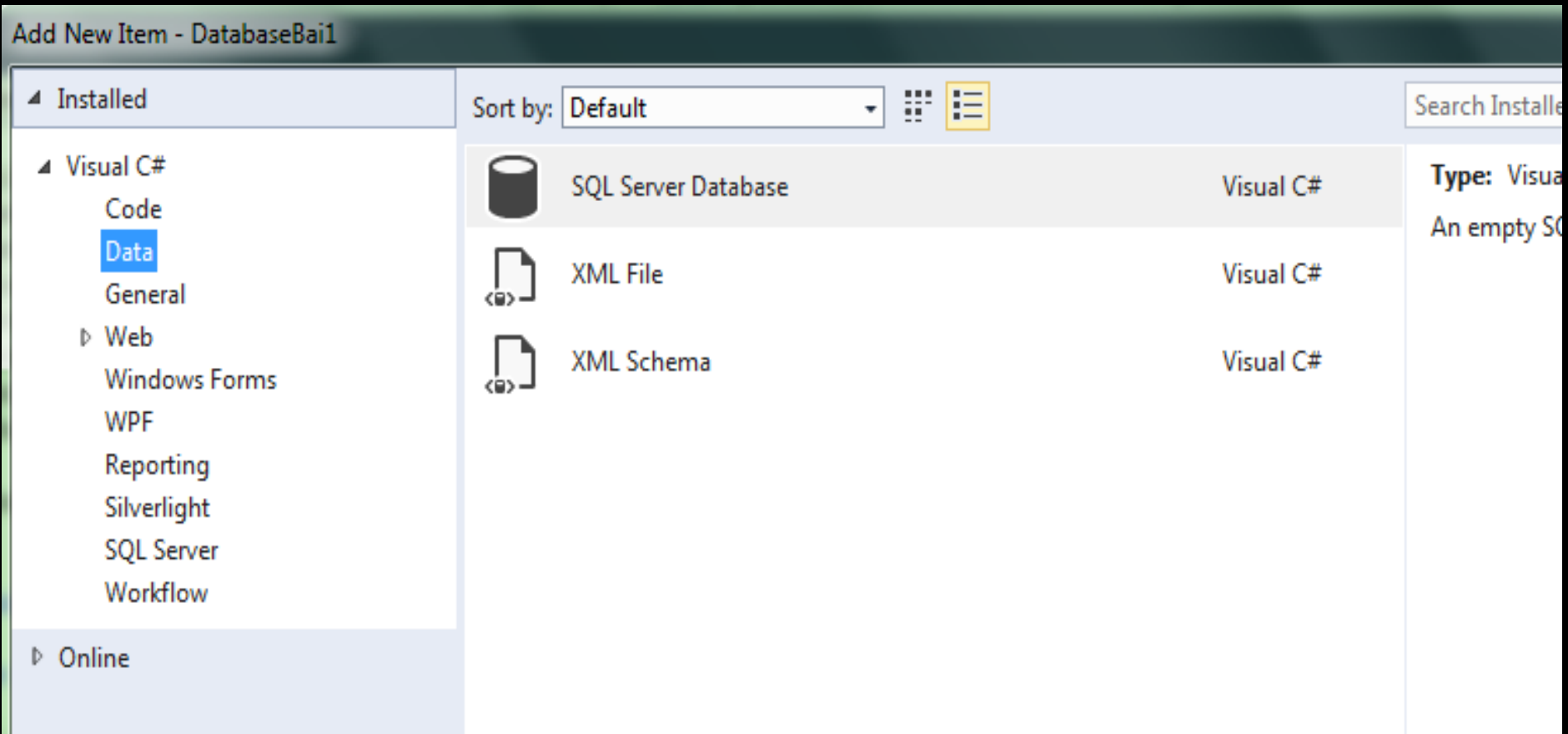
Database First

Code First

Model First

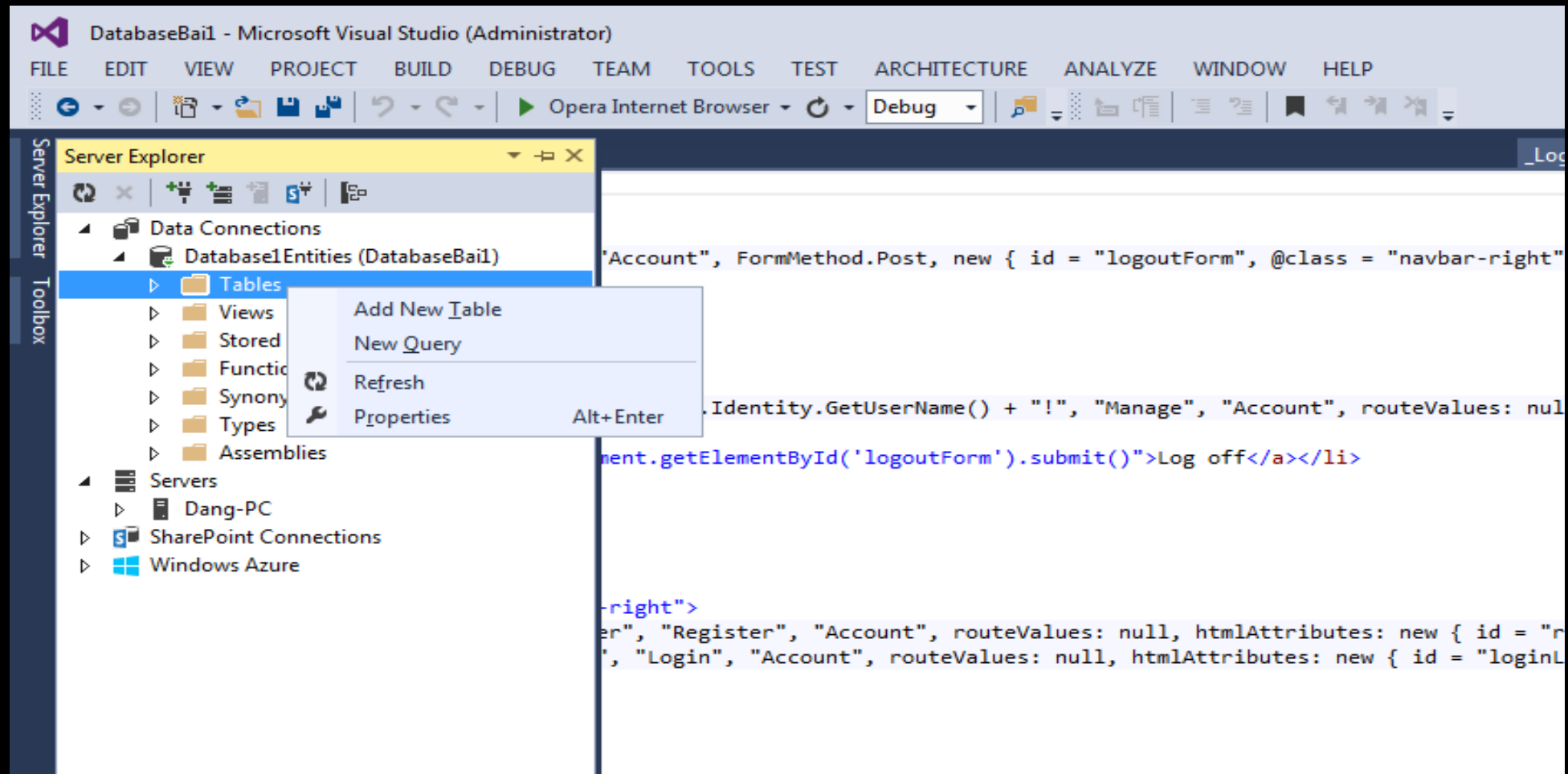
DB first

Bước 1: Tạo cơ sở dữ liệu : Data => Add New Item => Sql Sever Database.



DB first

Bước 2 : tạo các table Student và Class từ Server Explorer



DB first

- Tạo table với các thuộc tính và ràng buộc như sau:

The screenshot displays the SQL Server Enterprise Designer interface. The top pane shows the table design for `dbo.tblStudent` with the following columns:

Name	Data Type	Allow Nulls	Default
StudenId	int	<input type="checkbox"/>	
Name	nvarchar(30)	<input type="checkbox"/>	
Gender	nvarchar(3)	<input type="checkbox"/>	
BOD	date	<input type="checkbox"/>	
ClassId	int	<input type="checkbox"/>	

The right pane shows the constraints for the table:

- Keys (1)**
 - <unnamed> (Primary Key, Clustered: StudenId)
- Check Constraints (0)**
- Indexes (0)**
- Foreign Keys (1)**
 - <unnamed> (tblClass: ClassId)
- Triggers (0)**

The bottom pane shows the T-SQL script for creating the table:

```
CREATE TABLE [dbo].[tblStudent] (  
    [StudenId] INT NOT NULL,  
    [Name] NVARCHAR (30) NOT NULL,  
    [Gender] NVARCHAR (3) NOT NULL,  
    [BOD] DATE NOT NULL,  
    [ClassId] INT NOT NULL,  
    PRIMARY KEY CLUSTERED ([StudenId] ASC),  
    FOREIGN KEY ([ClassId]) REFERENCES [dbo].[tblClass] ([ClassId])  
);
```

DB first

- Tạo table với các thuộc tính và ràng buộc như sau:

The screenshot displays the SQL Server Enterprise Designer interface. The top toolbar shows the 'Update' button and a 'Script File' dropdown set to 'dbo.tblClass.sql'. The main workspace is divided into two panes. The left pane shows the table design with columns 'ClassId' and 'Name'. The right pane shows the 'Keys' section with a primary key constraint on 'ClassId'. The bottom pane shows the T-SQL script for creating the table.

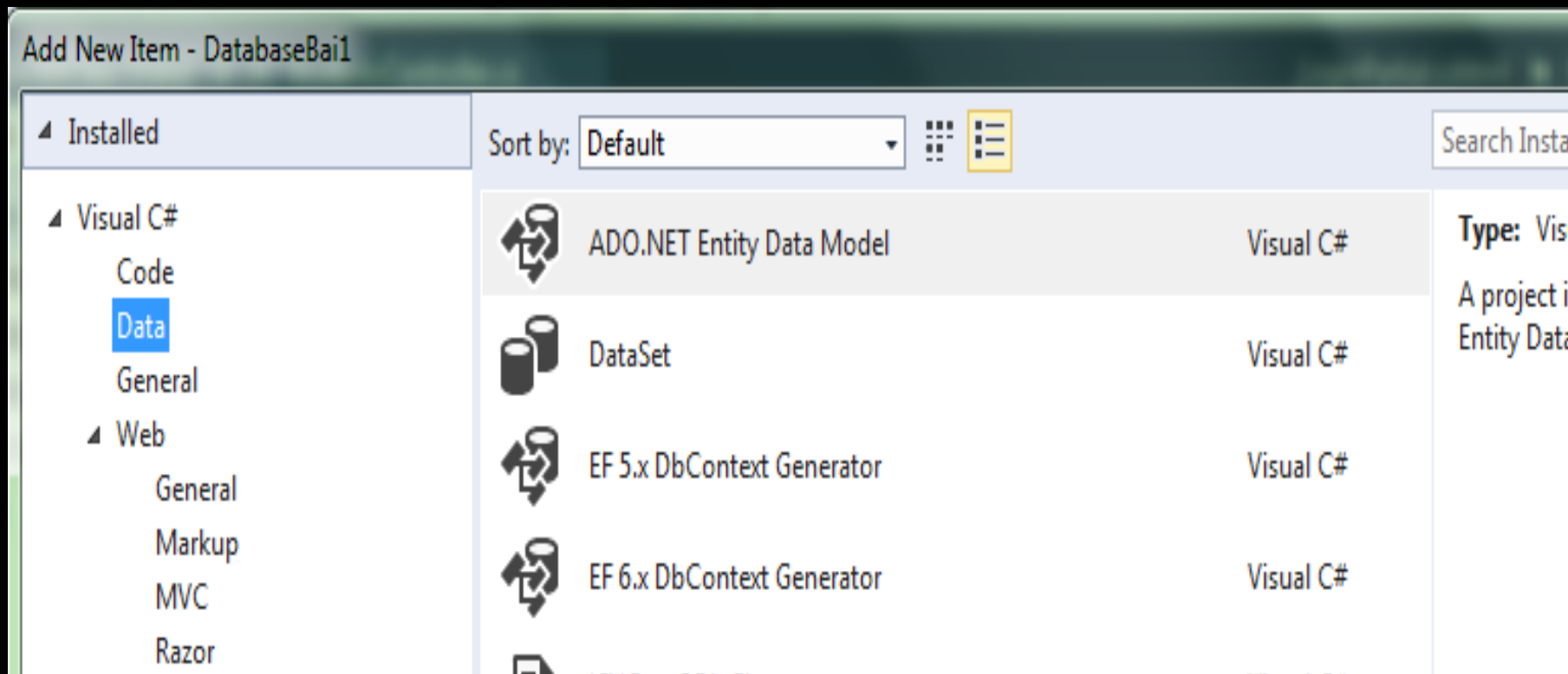
Name	Data Type	Allow Nulls	Default
ClassId	int	<input type="checkbox"/>	
Name	nvarchar(30)	<input type="checkbox"/>	

Keys (1)
<unnamed> (Primary Key, Clustered: ClassId)
Check Constraints (0)
Indexes (0)
Foreign Keys (0)
Triggers (0)

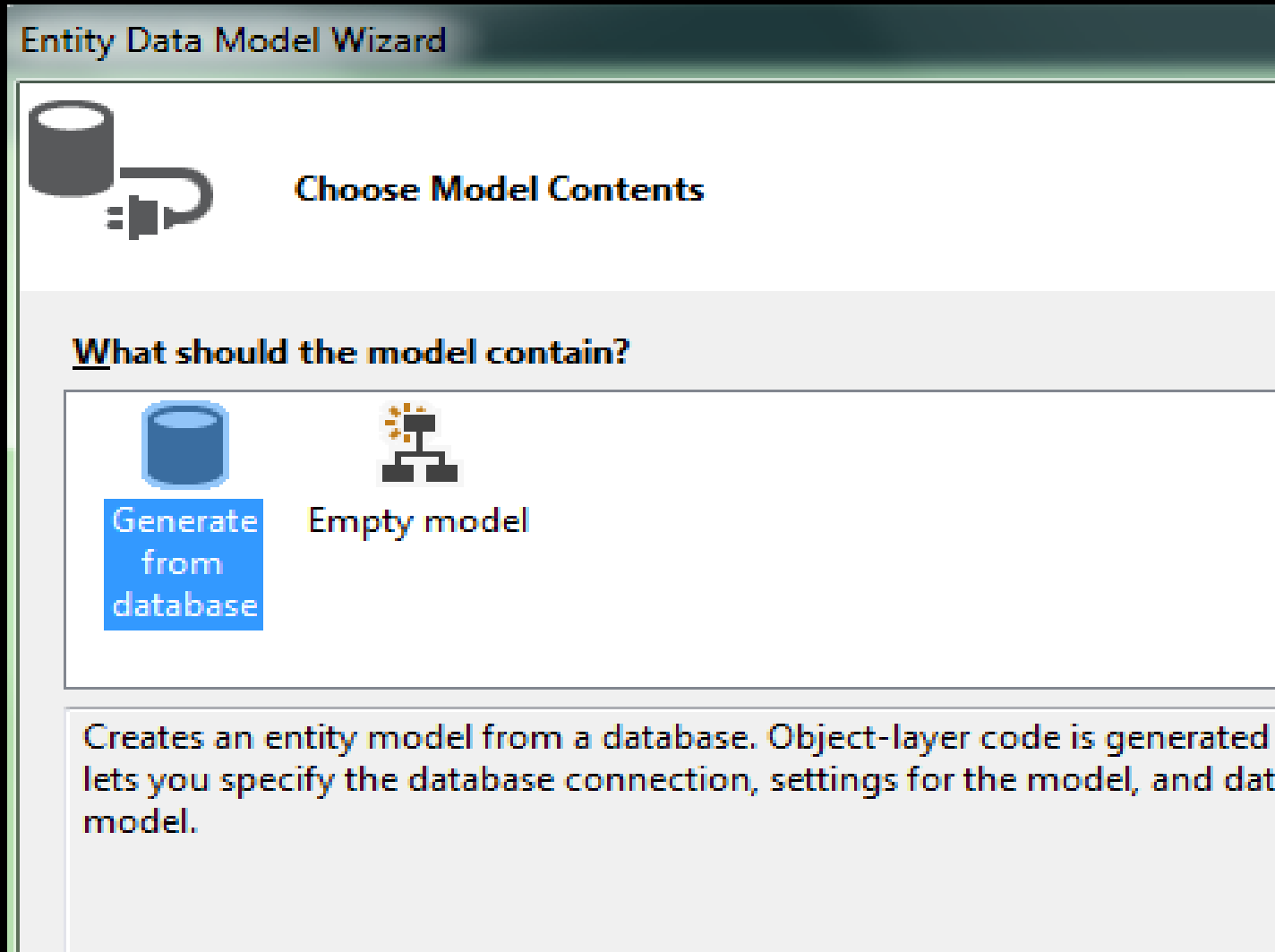
```
CREATE TABLE [dbo].[tblClass] (  
    [ClassId] INT NOT NULL,  
    [Name] NVARCHAR (30) NOT NULL,  
    PRIMARY KEY CLUSTERED ([ClassId] ASC)  
);
```

DB first

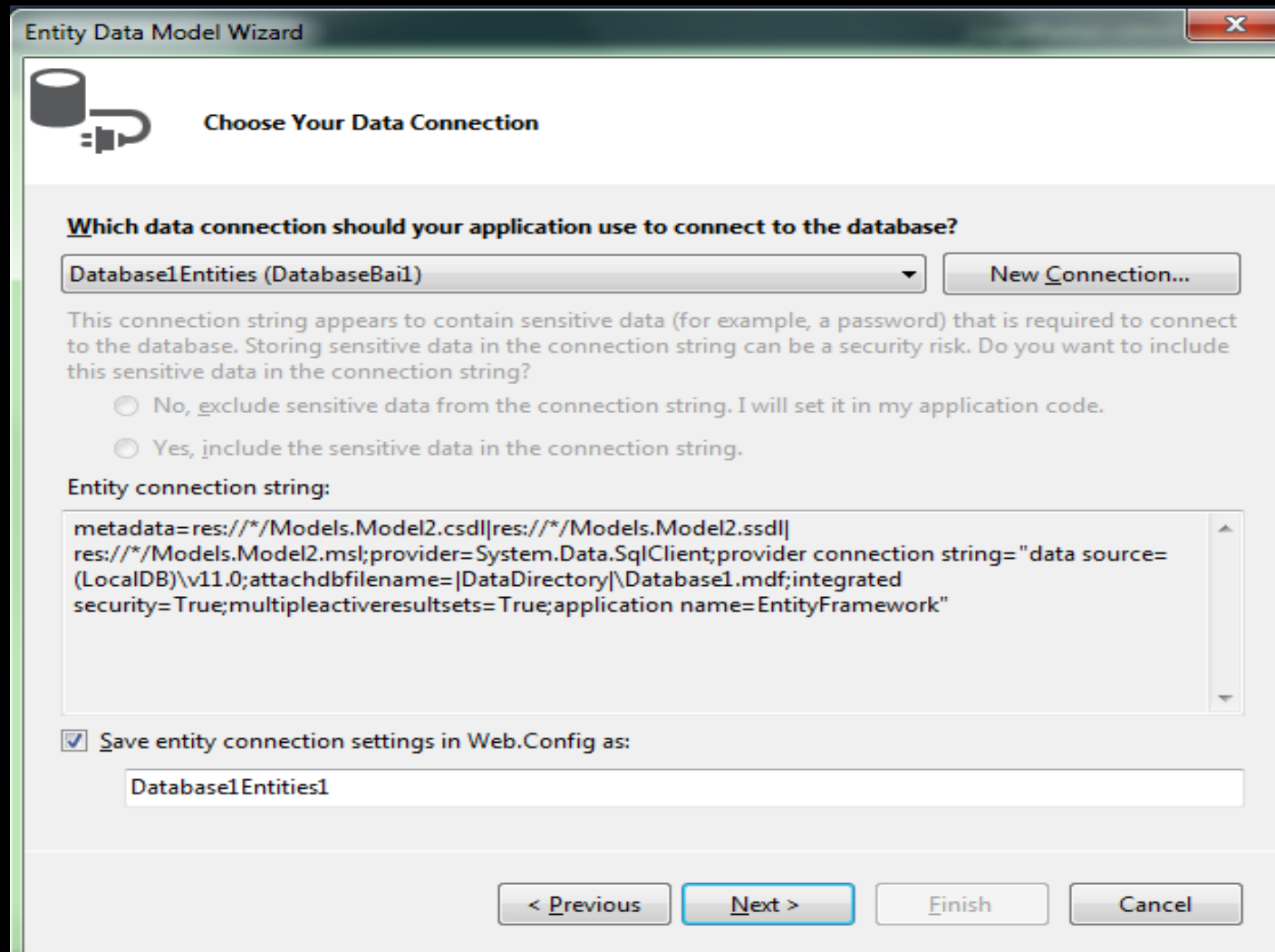
- **Bước 3 :** Thiết lập Entity Data Model dựa trên CSDL có sẵn: Model => Add=>New Item=> Data => ADO.NET Entity Data Model



DB first




DB first



The image shows a screenshot of the 'Entity Data Model Wizard' dialog box. The title bar reads 'Entity Data Model Wizard'. The main area has a header 'Choose Your Data Connection' with a database icon. Below this, a question asks 'Which data connection should your application use to connect to the database?'. A dropdown menu shows 'Database1Entities (DatabaseBai1)' and a 'New Connection...' button is next to it. A warning message states: '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?'. There are two radio buttons: 'No, exclude sensitive data from the connection string. I will set it in my application code.' and 'Yes, include the sensitive data in the connection string.'. Below this, the 'Entity connection string:' is displayed in a text box. At the bottom, there is a checkbox 'Save entity connection settings in Web.Config as:' which is checked, and a text box containing 'Database1Entities1'. Navigation buttons at the bottom are '< Previous', 'Next >', 'Finish', and 'Cancel'.

Entity Data Model Wizard

 Choose Your Data Connection

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

Database1Entities (DatabaseBai1) New Connection...

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:

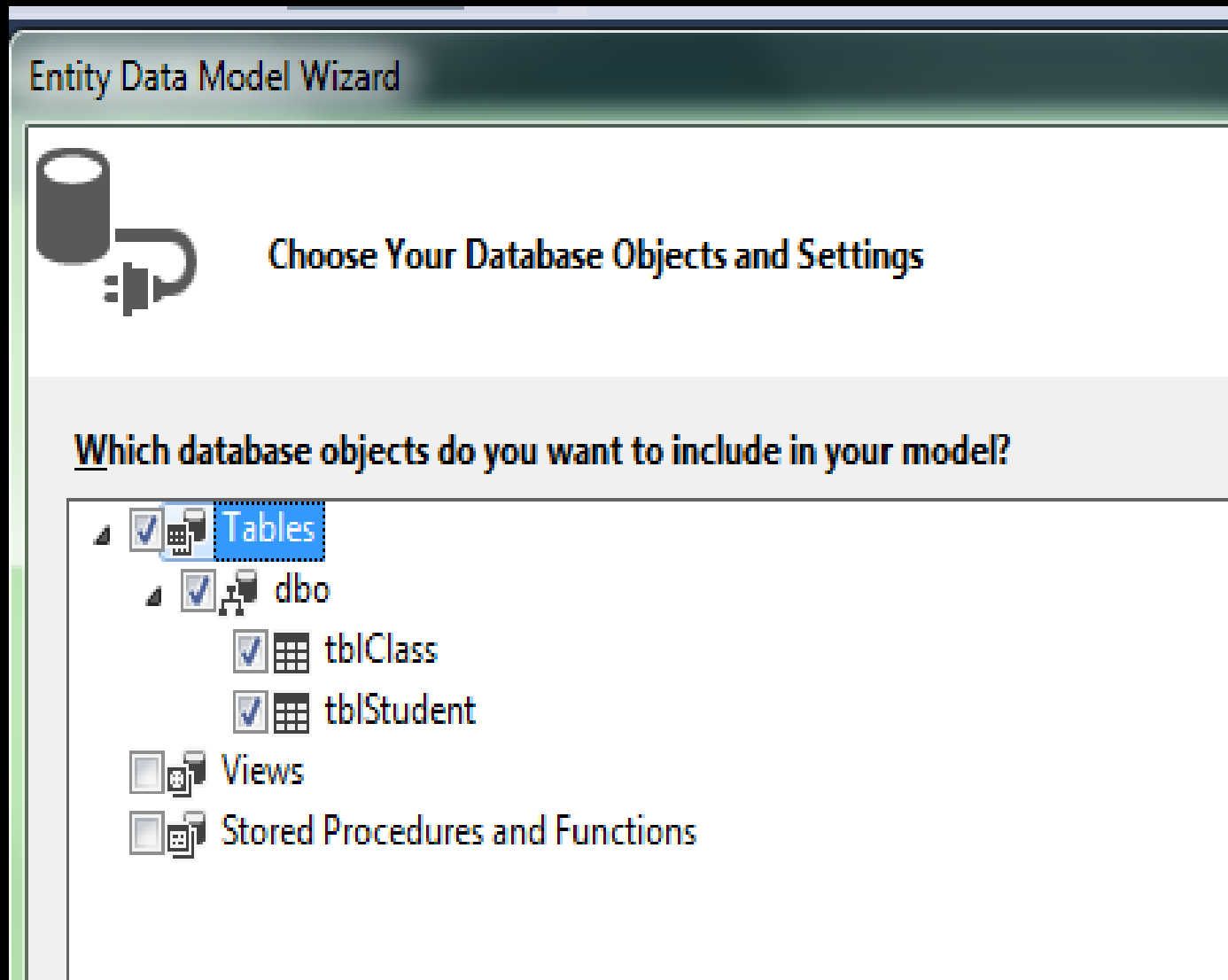
```
metadata=res://*/Models.Model2.csdl|res://*/Models.Model2.ssdl|
res://*/Models.Model2.msl;provider=System.Data.SqlClient;provider connection string="data source=
(LocalDB)\v11.0;attachdbfilename=|DataDirectory|\Database1.mdf;integrated
security=True;multipleactiveresultsets=True;application name=EntityFramework"
```

☒ Save entity connection settings in Web.Config as:

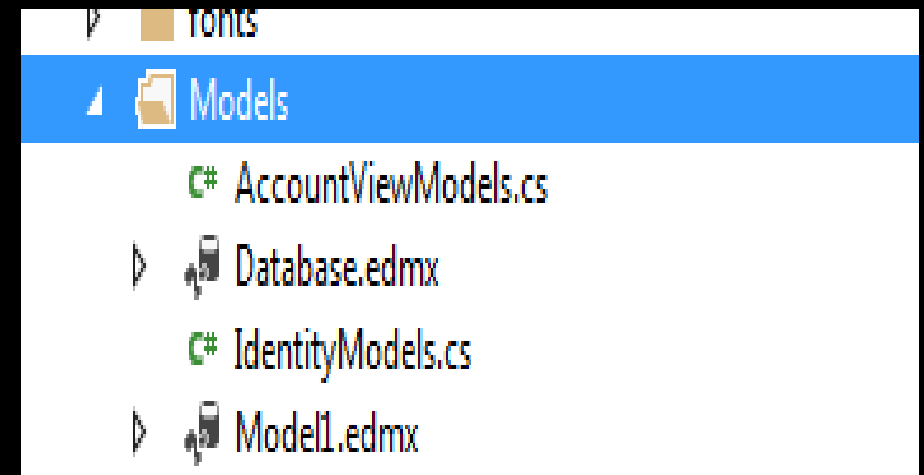
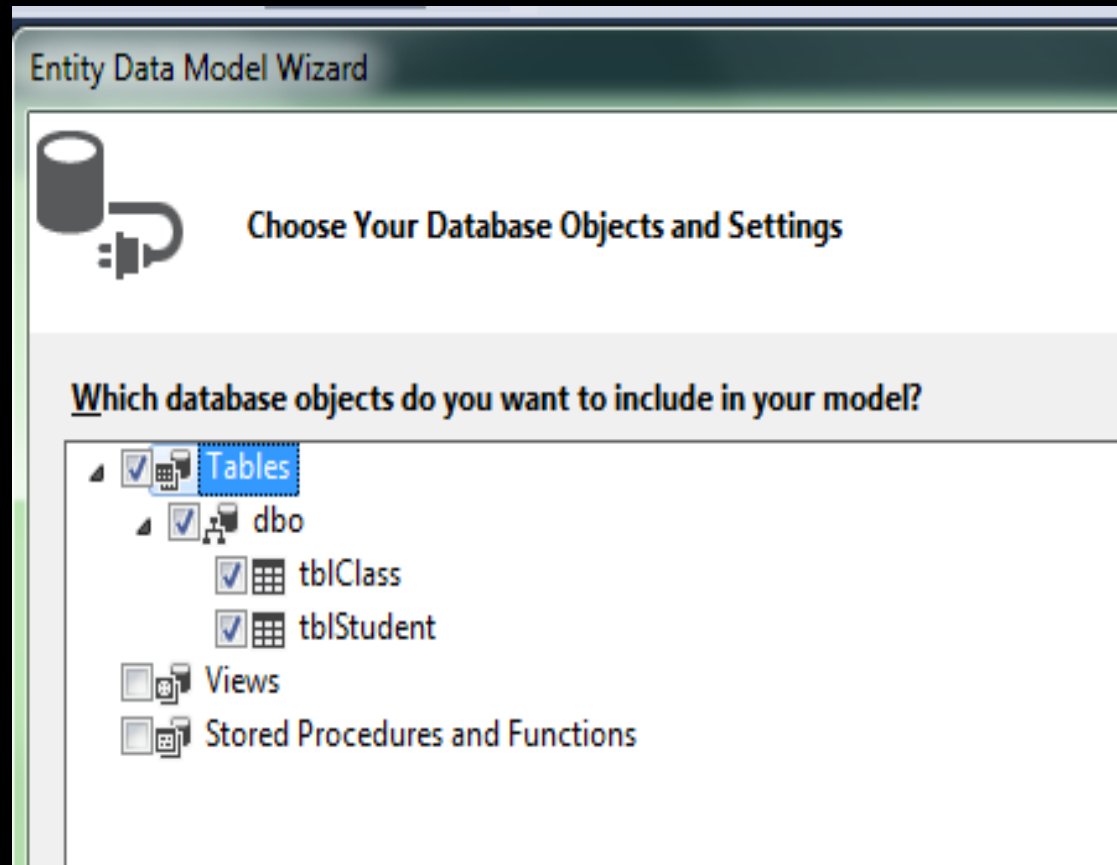
Database1Entities1

< Previous Next > Finish Cancel

DB first

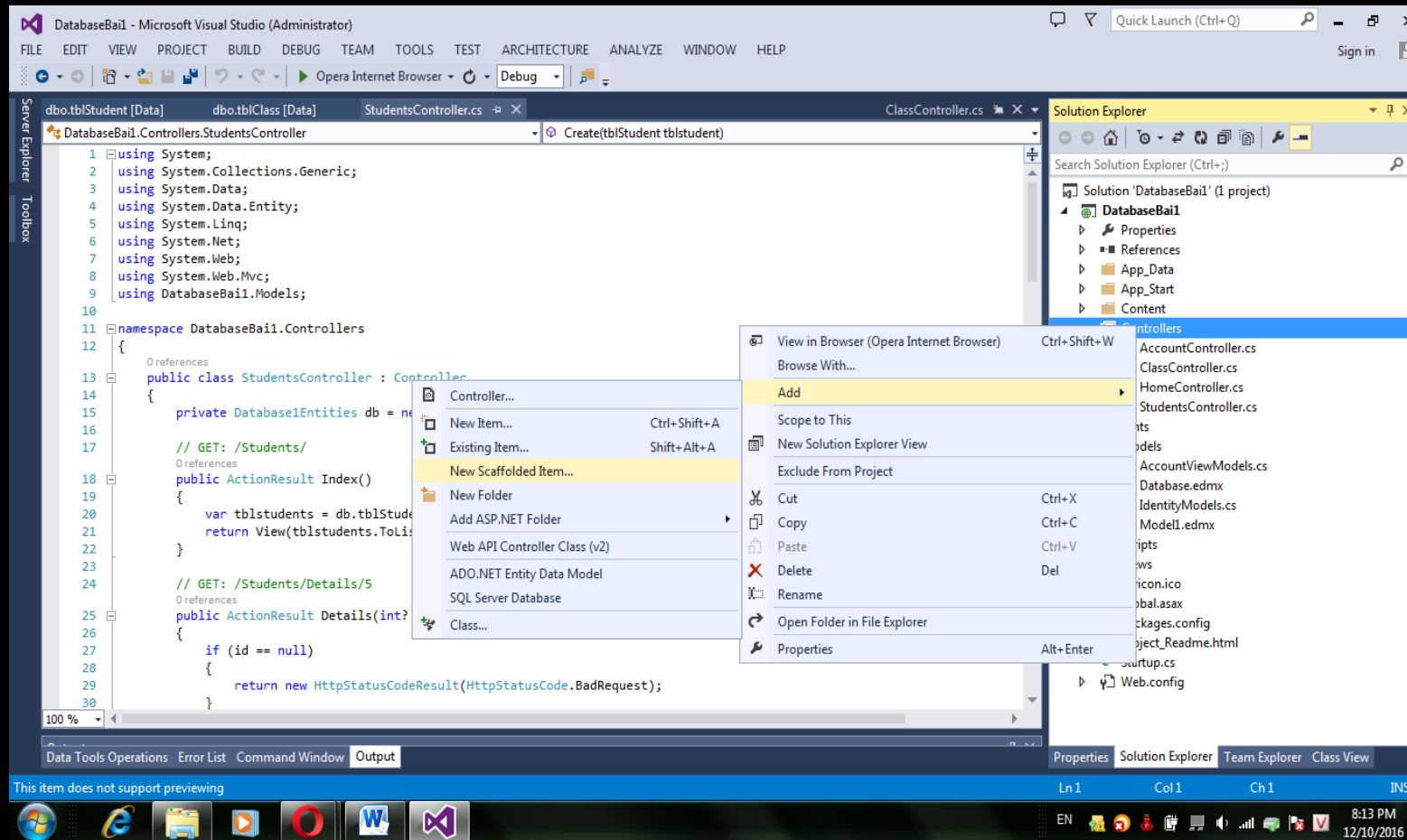


DB first



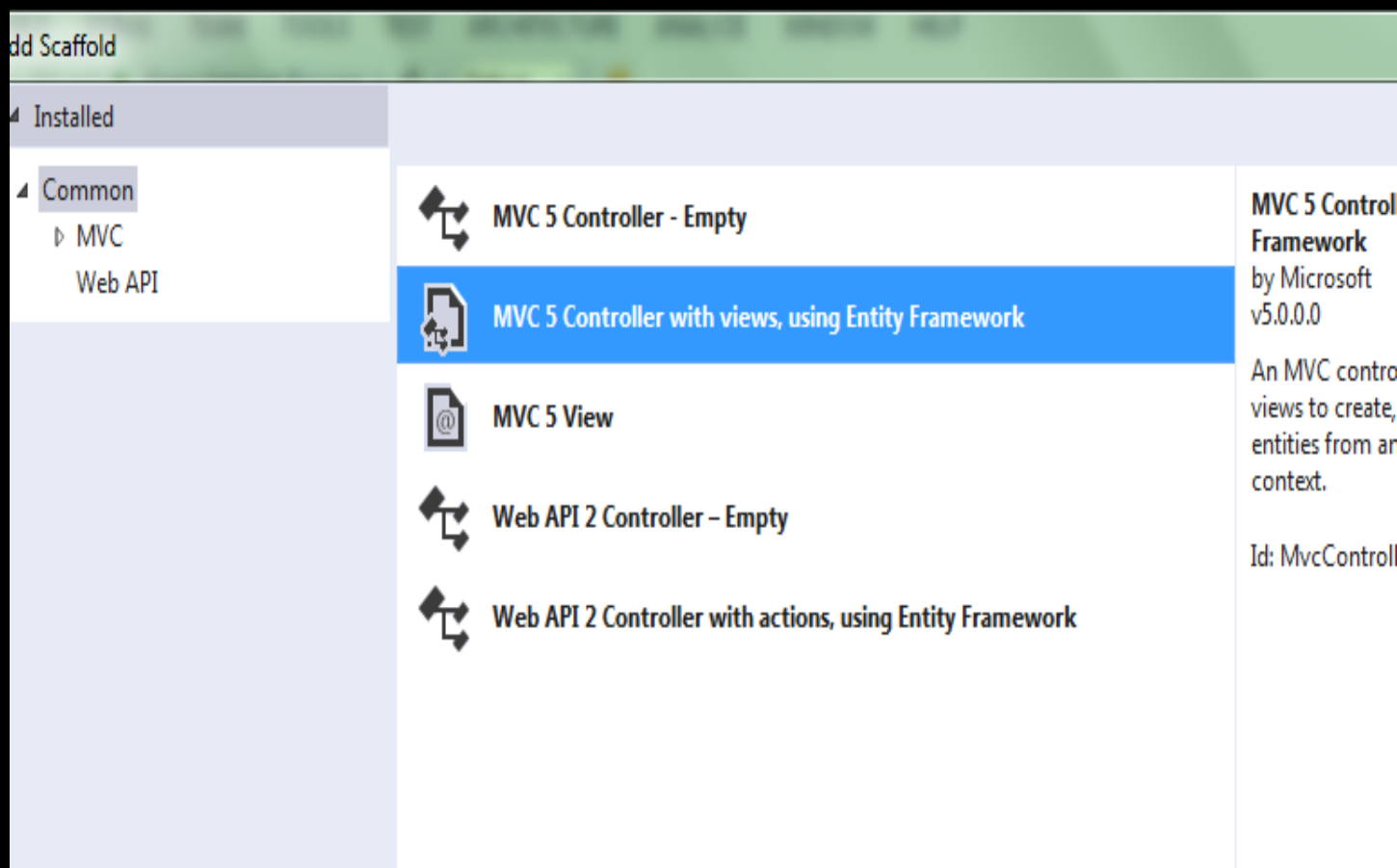
DB first

- **Bước 4:** Tạo Controller xử lý cho dữ liệu Student và Class



DB first

- **Bước 4:** Tạo Controller xử lý cho dữ liệu Student và Class



DB first

- Controller cho Student

Add Controller

Controller name:
StudentController

☐ Use async controller actions

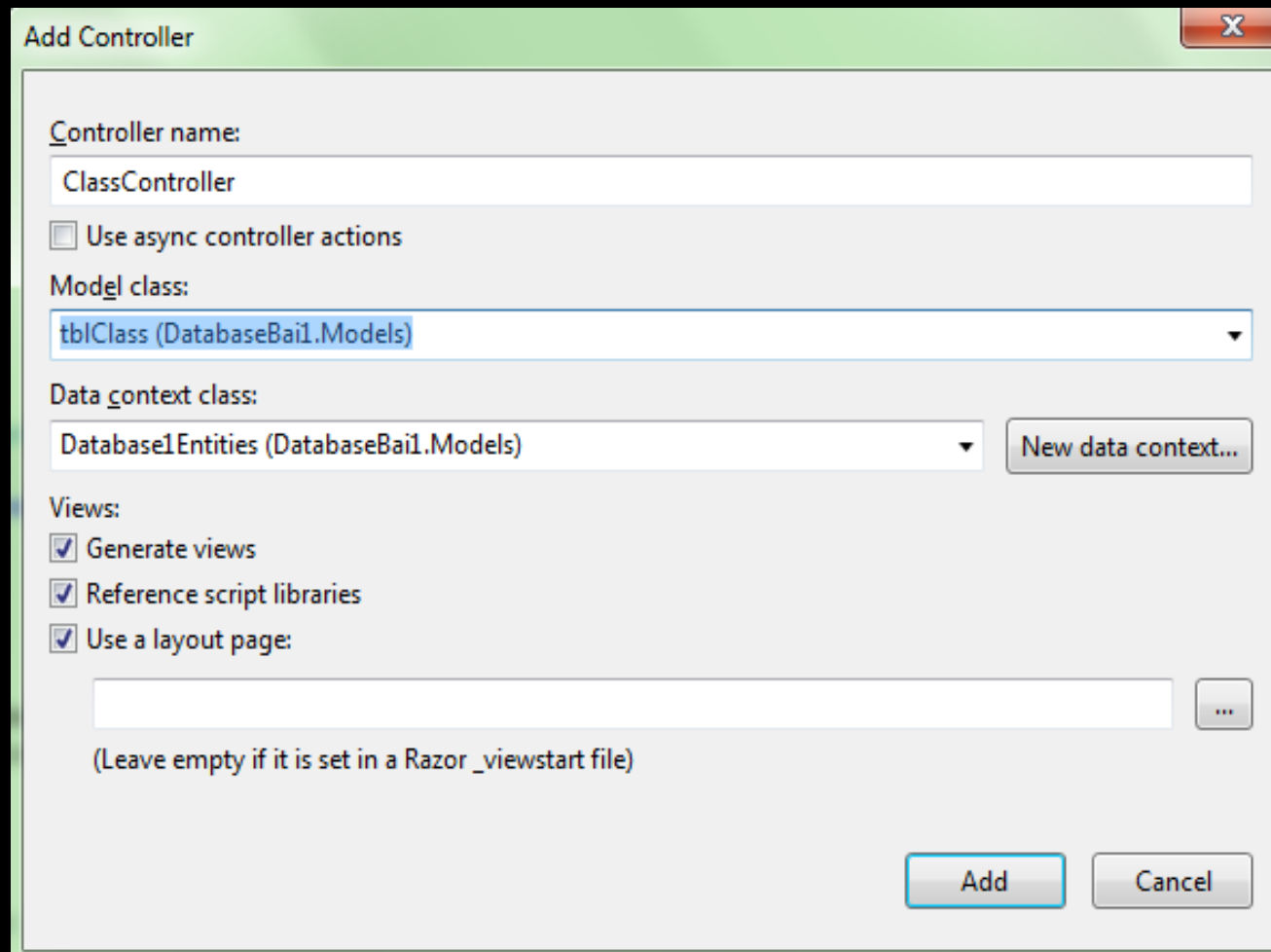
Model class:
tblStudent (DatabaseBai1.Models)

Data context class:
Database1Entities (DatabaseBai1.Models) ▼ New

Views:
☒ Generate views
☒ Reference script libraries
☒ Use a layout page:

DB first

- Controller cho Class



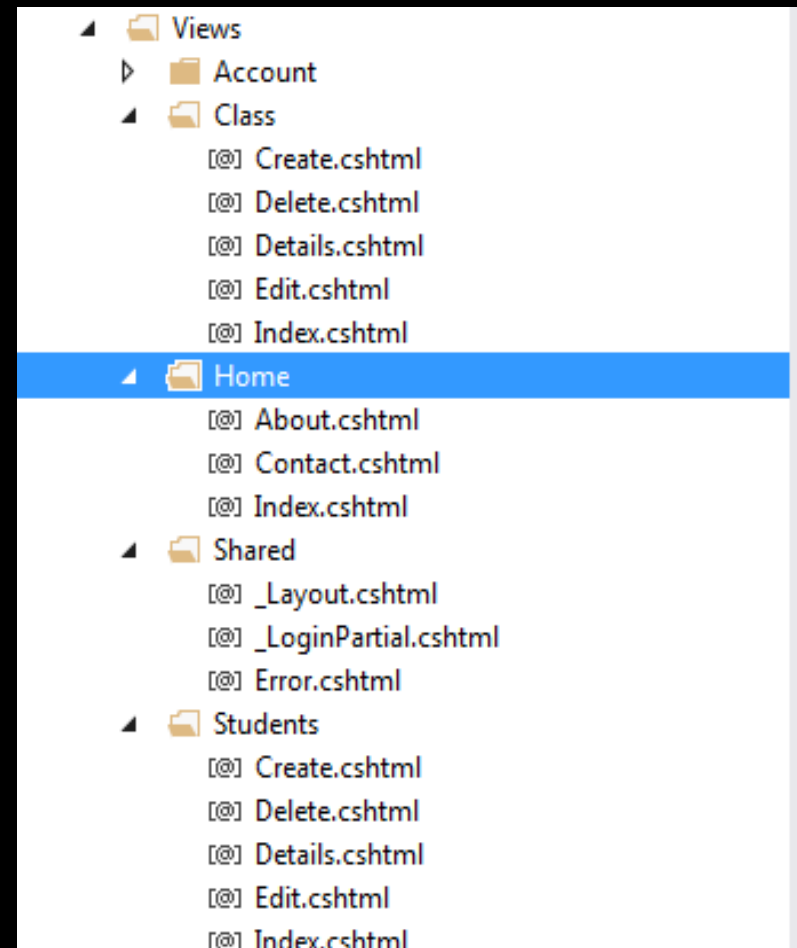
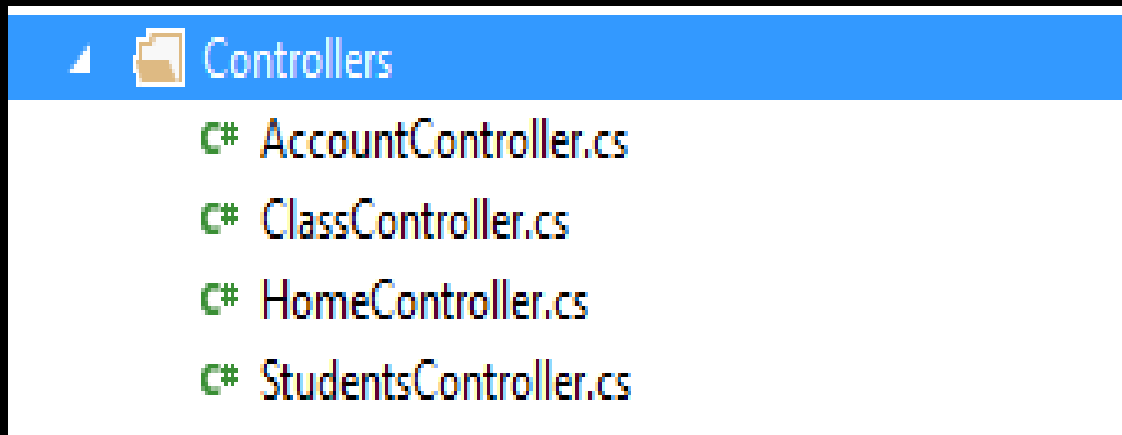
The screenshot shows the 'Add Controller' dialog box with the following configuration:

- Controller name:** ClassController
- ☐ Use async controller actions
- Model class:** tblClass (DatabaseBai1.Models)
- Data context class:** Database1Entities (DatabaseBai1.Models) [New data context... button]
- Views:**
 - ☒ Generate views
 - ☒ Reference script libraries
 - ☒ Use a layout page:

At the bottom, there is an empty text field for a layout page, followed by the instruction: (Leave empty if it is set in a Razor _viewstart file). The 'Add' and 'Cancel' buttons are at the bottom right.

DB first

- Sau khi add xong ta sẽ thêm 2 Controller như sau và các Views tương ứng



Kết quả

CHAUHAI DANG Add Student Add class Student View

Add Student

tblStudent

StudentId

Name

Gender

BOD

ClassName

▼

Create

[Back to List](#)

© 2016 - My ASP.NET Application

Kết quả

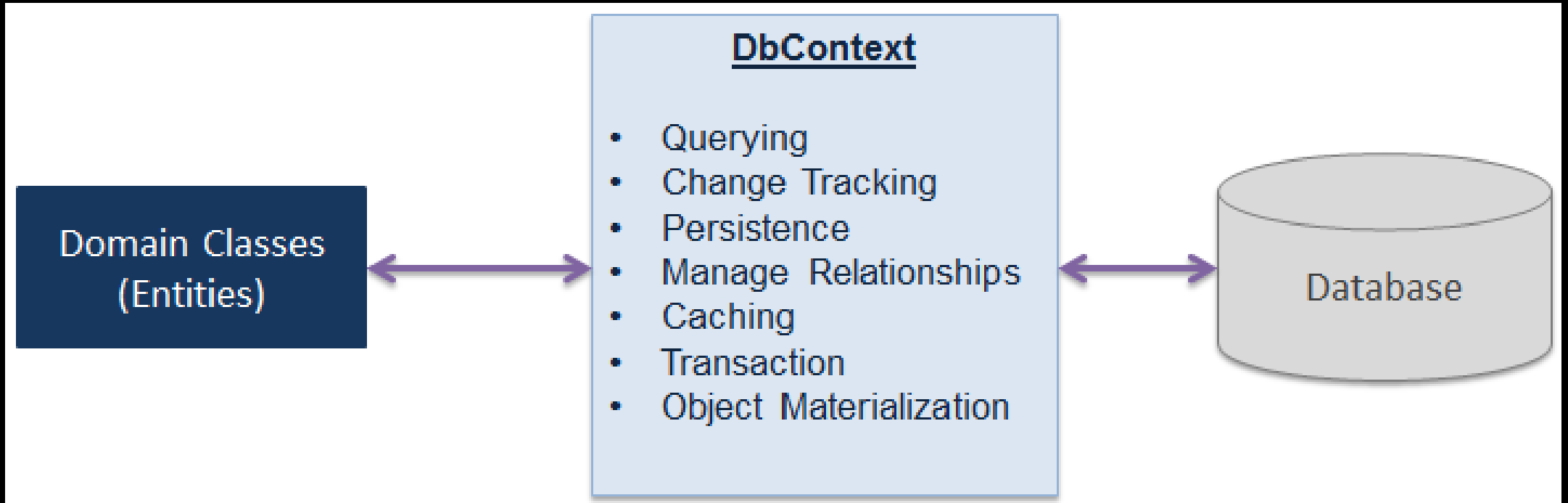
Student Information

Create New

Name	Gender	BOD	Name
Ga	Nu	9/9/1995 12:00:00 AM	PM1301-1 Edit Details Delete
Chau Hai Dang	Nam	9/9/1995 12:00:00 AM	PM1301-2 Edit Details Delete
Ngo Van Dat	Nam	1/1/1995 12:00:00 AM	MV1301-2 Edit Details Delete
Pham Yen Duyen	Nu	12/1/1995 12:00:00 AM	PM1301-2 Edit Details Delete
Dang Hoang Uyen Phuong	Nu	1/12/1995 12:00:00 AM	PM1301-2 Edit Details Delete

Question #1

- DbContext là gì?
-



Question #2

- The DbSet class là gì?

```
EFBasicTutorials.SchoolDBEntities  SchoolDBEntities()
namespace EFBasicTutorials
{
    using System;
    using System.Data.Entity;
    using System.Data.Entity.Infrastructure;
    using System.Data.Entity.Core.Objects;
    using System.Linq;

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

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

        public virtual DbSet<Course> Courses { get; set; }
        public virtual DbSet<Standard> Standards { get; set; }
        public virtual DbSet<Student> Students { get; set; }
        public virtual DbSet<StudentAddress> StudentAddresses { get; set; }
        public virtual DbSet<Teacher> Teachers { get; set; }
        public virtual DbSet<View_StudentCourse> View_StudentCourse { get; set; }
```