第 3 章

EF 数据模型

本章导读

EF(Entity Framework)是模型开发中除 LINQ 外另一种常用设计模式,使用 EF 进行项目开发时不需要去学习 SQL 指令,程序员只要会应用 LINQ 就能很方便地操作 SQL Server 数据库。本章将学习基于 EF 的数据库优先、模型优先以及代码优先的三种设计模式进行模型的快速创建,以及如何调用相关方法对模型中数据进行增、删、改、查处理。

本章要点

- EF 数据模型
- EF 的三种设计模式
- EF 模型数据处理

3.1 Entity Framework 简介

视频讲解

EF(Entity Framework)是微软以 ADO.NET 为基础开发的对象关系映射(Object Relational Mapping,ORM)解决方案。作为一种 ORM 的数据访问框架,EF 将数据从对象自动映射到关系数据库,不需要编写大量的数据访问代码,只要会应用 LINQ 就可以如同 Object 对象一样方便地操作数据库,节省编写数据库访问代码的时间。

EF 框架具有良好的扩展性,除了可以访问 SQL Server 数据库,其他数据库只要按 EF 所提供的接口实现对应的 SQL 生成器与连接管理机制,就能在 EF 中得到支持,当前 EF 支持的主要数据库如表 3.1 所示。

表 3.1 EF 支持的主要数据库

数 据 库 名
Microsoft SQL Server(含 Express、Express LocalDb 及 Compact)
Oracle
MySql
IBM DB2、Informix 与 U2
Npgsql(PostgreSQL)
Sybase SQL Anywhere、Adaptive Server
SQLite
Synergy DBMS
Firebird
VistaDB

3.2 Entity Framework 设计模式


视频讲解

Entity Framework 在开发时主要有 Code First、Model First 以及 Database First 三种设计模式。对于初次使用 EF 的读者，建议从 Database First 模式开始学习，熟悉了 ObjectContext＜T＞和 LINQ to Entities 之后，再使用 Code First 模式和 Model First 模式进行实践。下面依次介绍这三种设计模式的基本用法。

3.2.1 Database First 模式


视频讲解

Database First 模式，即数据库优先设计模式，是指以数据库设计为基础，通过设计好的数据库自动生成实体数据模型，从而实现整个系统开发的设计流程，该模式设计较简单，适合对数据库有一定了解的初学者。

【例 3-1】 创建控制台应用程序，使用 EF 框架中的 Database First 模式，基于 Demo 数据库在项目中创建实体类。

步骤 1：Visual Studio 2017 菜单栏中选择“文件”→“新建”→“项目”选项，如图 3.1 所示。

步骤 2：创建控制台应用程序，命名为 EF-DatabaseFirst，如图 3.2 所示。

步骤 3：在控制台应用程序上右击，选择“添加”→“新建项”选项，如图 3.3 所示。

步骤 4：在“添加新项”窗口，选择 ADO.NET 实体数据模型，单击“添加”按钮，如图 3.4 所示。

步骤 5：在“实体数据模型向导”窗口的“选择模型内容”项中，选择“来自数据库的 EF 设计器”，单击“下一步”按钮，如图 3.5 所示。

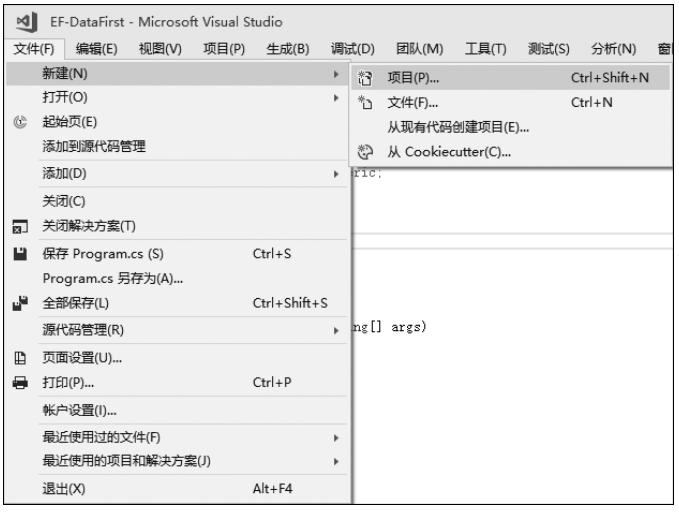

图 3.1　创建新项目

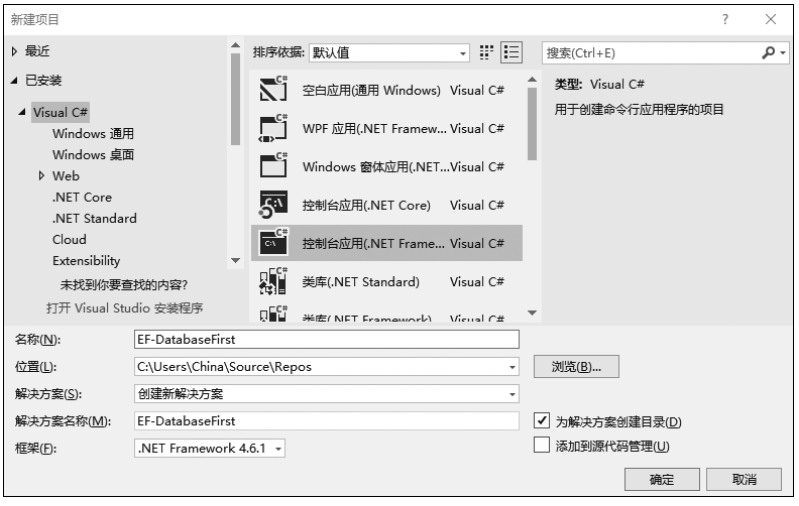

图 3.2　创建控制台应用程序

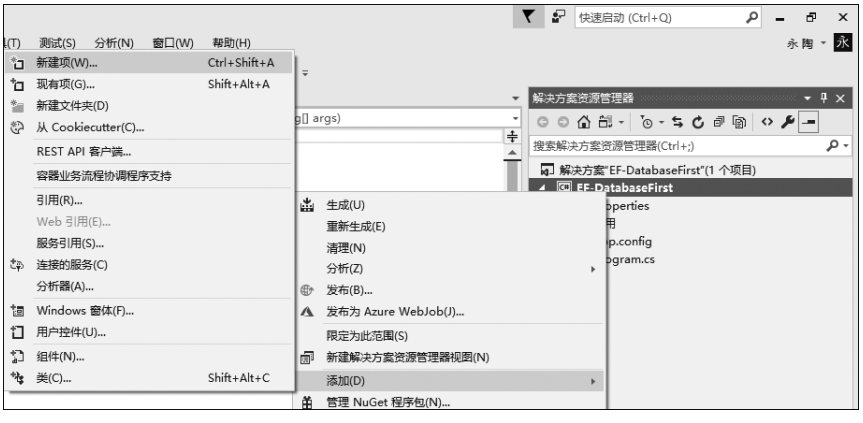

图 3.3　添加新项

图 3.4 添加实体数据模型

图 3.5 选择模型

步骤 6：在“实体数据模型向导”窗口的“选择您的数据连接”项中，单击“新建连接”按钮，弹出“连接属性”窗口，“数据源”项选择为 Microsoft SQL Server (SqlClient)，“服务器名”项设置为.\sqlexpress，选择数据库名称项设置为 demo，单击“确定”按钮，如图 3.6 所示。

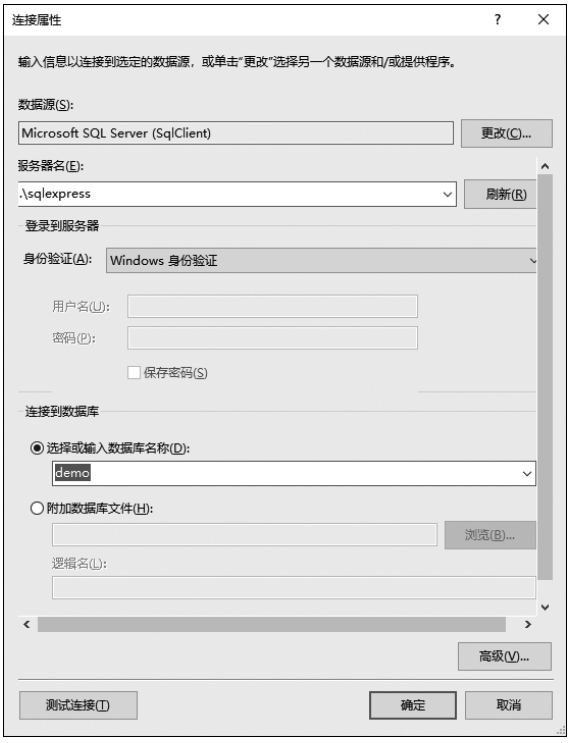

图 3.6　数据库连接设置

步骤 7：默认将连接字符串保存到 App.Config 文件的 demoEntities 标签内。单击“下一步”按钮，如图 3.7 所示。

图 3.7　保存数据库连接字符串

步骤 8：在“实体数据模型向导”窗口的“选择您的版本”项中，选择“实体框架 6.x”版本，单击“下一步”按钮，如图 3.8 所示。

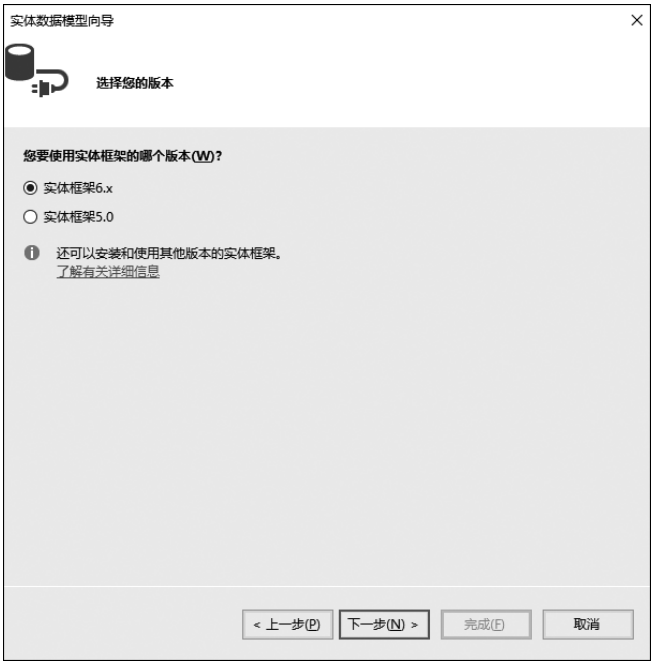

图 3.8 选择实体框架版本

步骤 9：在“实体数据模型向导”窗口的“选择您的数据库对象和设置”项中，选择数据库对象“表”，选中“确定所生成对象名称的单复数形式”项，单击“完成”按钮，如图 3.9 所示。

图 3.9 选择数据库中对象

步骤 10：Visual Studio 将开始加入 EF 的程序包，以及自数据库中查询待导入的对象，并同步打开 EDM Designer 编辑页面。除此，还生成了包括数据集合类 Model1.Context.cs，以及各数据表对应的 student.cs、course.cs、sc.cs 等实体类，如图 3.10、图 3.11 所示。

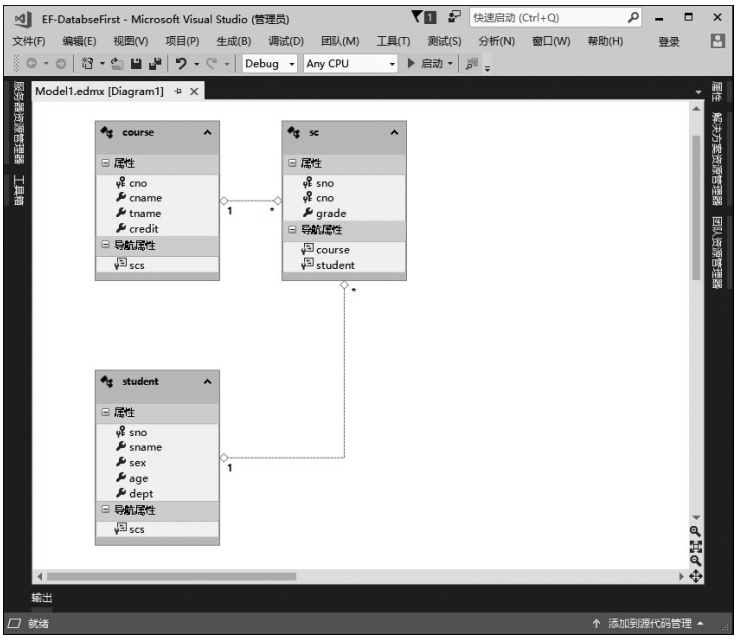

图 3.10　模型关系图

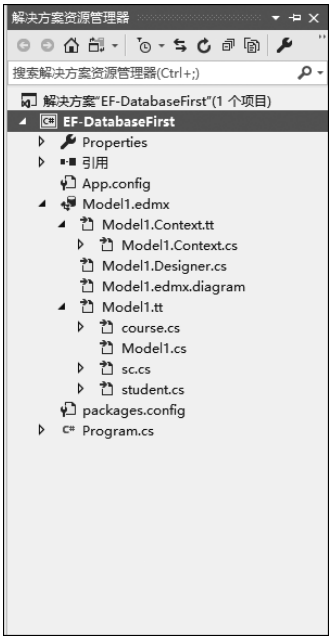

图 3.11　自动生成的文件

其中，数据集合类 Model1.Context.cs 主要代码如下。

```
public partial class demoEntities2 : DbContext
{
    public demoEntities2()
        : base("name=demoEntities2")
    {
    }

    protected override void OnModelCreating(DbModelBuilder modelBuilder)
    {
        throw new UnintentionalCodeFirstException();
    }
    public virtual DbSet<course>courses { get; set; }
    public virtual DbSet<sc>scs { get; set; }
    public virtual DbSet<student>students { get; set; }
}
```

实体类 sc.cs 中代码如下。

```
public partial class sc
```

```
{
    public int sno { get; set; }
    public int cno { get; set; }
    public Nullable<int>  grade { get; set; }
    public virtual course course { get; set; }
    public virtual student student { get; set; }
}
```

步骤 11：创建完基本的模型后，打开 Program.cs 文件，添加基本的数据访问程序，进行数据访问，编辑代码如下。

```
class Program
{
    static void Main(string[] args)
    {
        demoEntities2 context=new demoEntities2();
        var students = from item in context.students
                       select new
                       {
                           no = item.sno,
                           name = item.sname,
                           age = item.age
                       };
        foreach (var item in students)
        {
            Console.WriteLine("No:{0},Name:{1},Age:{2}", item.no, item.name.Trim(),
item.age);
        }
    }
}
```

步骤 12：运行网站结果如图 3.12 所示。

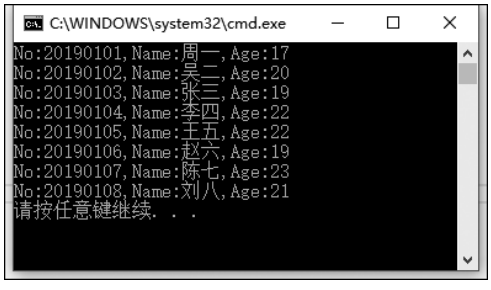

图 3.12 数据库中查询的数据

3.2.2 Model First 模式


视频讲解

Model First 模式，即模型优先设计模式，是从 EF 4 开始新增的功能，是指从实体数据模型入手，根据模型创建数据库的开发模式。Model First 模式是当前使用较多的一种开发模式，更符合面向对象的设计理念，只要在 Designer 内设计好模型的结构，就可以利用这个结构来生成数据库及对应代码。Model First 和 Database First 两种模式是可逆的，都可以得到数据库和实体数据模型。

【例 3-2】 创建控制台应用程序，按 Demo 数据库中数据表的结构设计模型，使用 EF 框架中的 Model First 模式，在项目中生成数据库以及代码。

步骤 1：创建控制台应用程序，命名为 EF-ModelFirst。

步骤 2：在控制台应用程序上右击，选择“添加”→“新建项”选项。在“添加新项”窗口，选择 ADO.NET 实体数据模型并命名为 Model，单击“添加”按钮。

步骤 3：在“实体数据模型向导”窗口的“选择模型内容”项中，选择“空 EF 设计器模型”选项，单击“完成”按钮，如图 3.13 所示。

图 3.13 实体数据模型向导

步骤 4：打开空白的 Designer 设计页面，工具箱会出现创建模型的控件，如同 Windows Forms 的窗体设计一样，可以从工具箱拖拽控件建立模型，如图 3.14 所示。

步骤 5：参考 Demo 数据库的模型，在工具箱中选中“实体”，按住鼠标左键拖放到 Designer 内，产生新的模型，将名称改为 student，在模型上右击，选择“新增”→“标量属性”命令，可为其重命名，如图 3.15 所示。

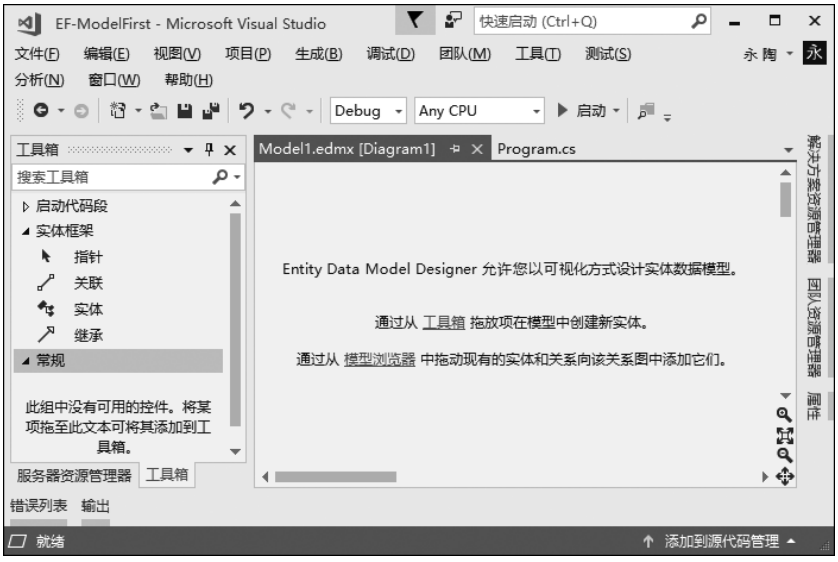

图 3.14 模型设计界面

步骤 6：新增并修改 student 实体的 sno、sname、sex、age 和 dept 五个属性，设计 student 实体模型如图 3.16 所示。

图 3.15 属性添加页面

图 3.16 student 实体模型

步骤 7：依次选中每个属性，在“属性”窗口的类型中按表 3.2 设置 sno、sname、sex、age 和 dept 五个属性的数据类型，如图 3.17 所示。

表 3.2 student 实体属性

属性名	字段描述	数据类型	主 键	可否为 NULL
sno	学号	Int32	True	False
sname	姓名	String	False	False
sex	性别	String	False	True
age	年龄	Int32	False	True
dept	部门	String	False	True

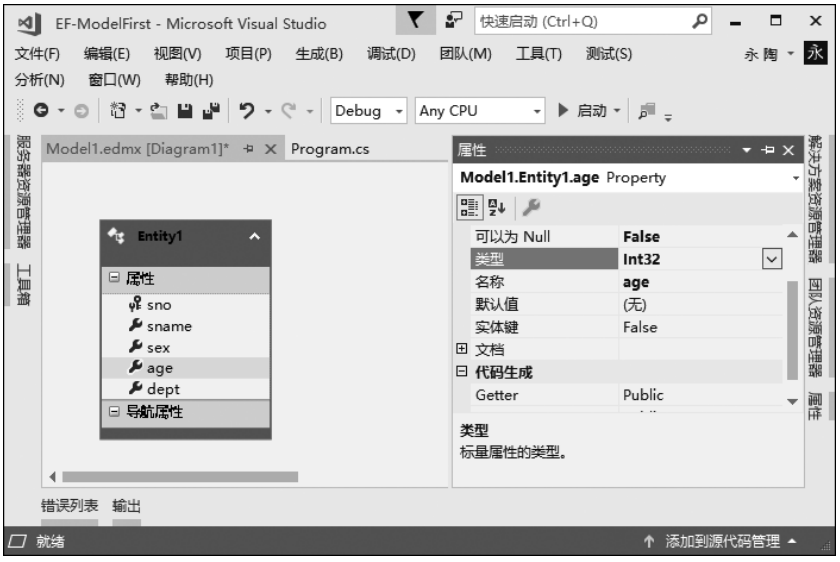

图 3.17　student 属性

步骤 8：创建另外两个新实体 course 和 sc，如图 3.18 所示。相关属性设置如表 3.3 和表 3.4 所示。

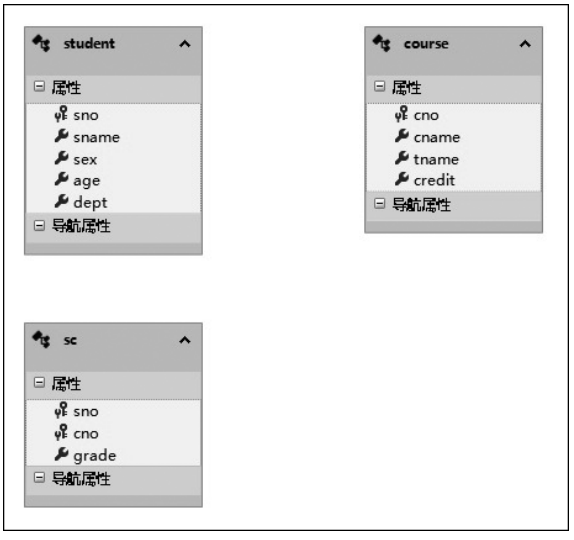

图 3.18　各实体属性

表 3.3　course 表实体属性

属性名	数据类型	主　　键	可否为 NULL
cno	Int32	True	False
cname	String	False	True
tname	String	False	False
credit	Int32	False	False

表 3.4 sc 表实体属性

属性名	数据类型	主 键	可否为 NULL
sno	Int32	True	False
cno	Int32	True	False
grade	Int32	False	False

步骤 9：建立实体之间的关联，为实体创建外键关联。在工具箱中选择“关联”选项，然后分别选中 student 和 sc 实体，添加关联线，一个 student 对象可以同时拥有多个 sc 对象，故为一对多的关系。同样为 course 实体和 sc 实体之间添加关联，如图 3.19 所示。

在建立关联的同时，会产生“导航属性”选项，通过导航属性，就能直接浏览关联好的对象。在 Designer 的空白处右击，在“属性”窗口将“以复数形式表示新对象”属性值设置为 True，如图 3.20 所示。该属性值为 True，student 产生数据表时对应的名称将设为 students，course 名称将设为 courses，sc 名称将设为 scs。

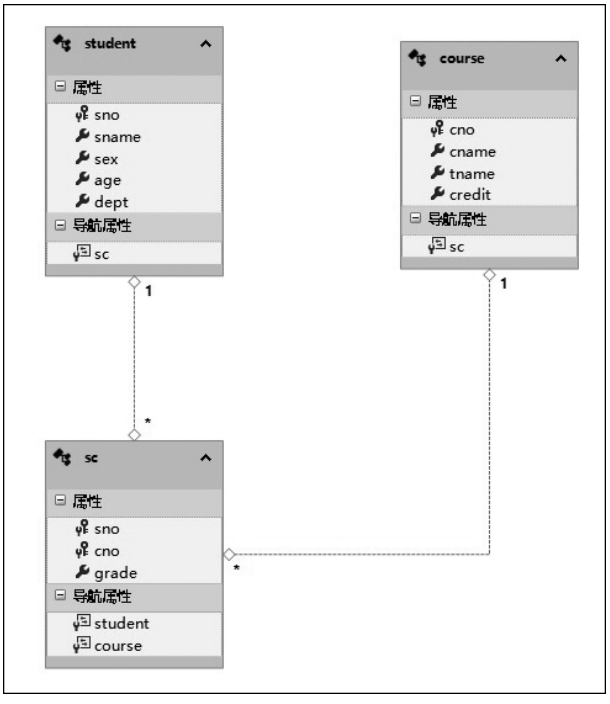

图 3.19 各实体关系图

图 3.20 模型属性设置

步骤 10：生成为数据库前，进行 DbContext 对象设置。在 Designer 上右击，选择“添加代码生成项”选项，如图 3.21 所示。

步骤 11：在“添加新项”窗口选择 DbContext 生成器，选择“EF 6.x DbContext 生成器”选项，单击“添加”按钮，如图 3.22 所示。

步骤 12：设置完成后，在 Model1.edmx 项目下新增了 Model1.Context.tt 和 Model1.tt 文件，如图 3.23 所示。

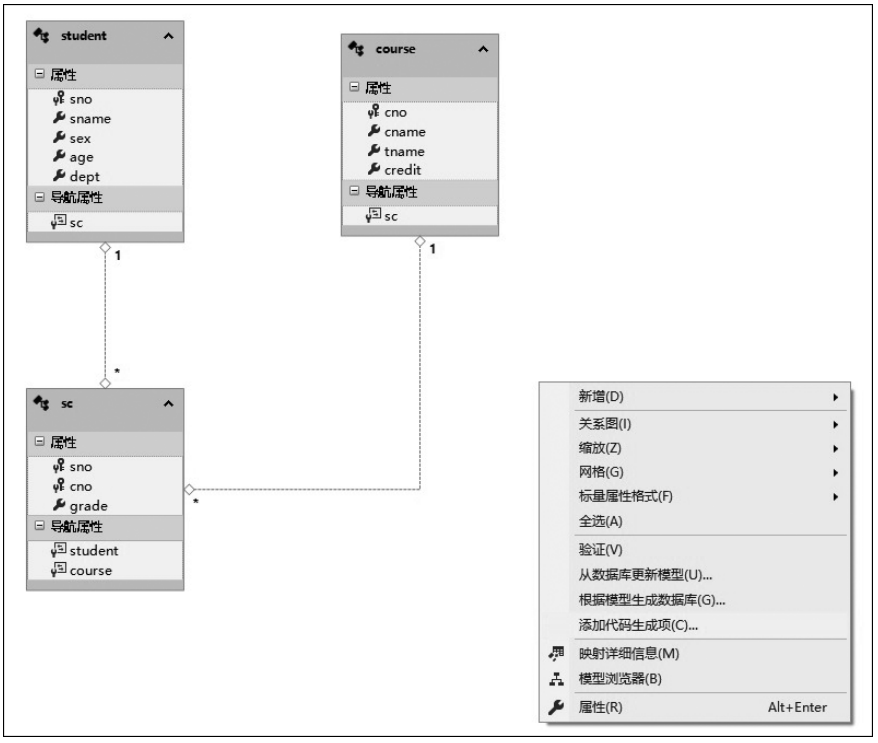

图 3.21　添加代码生成项

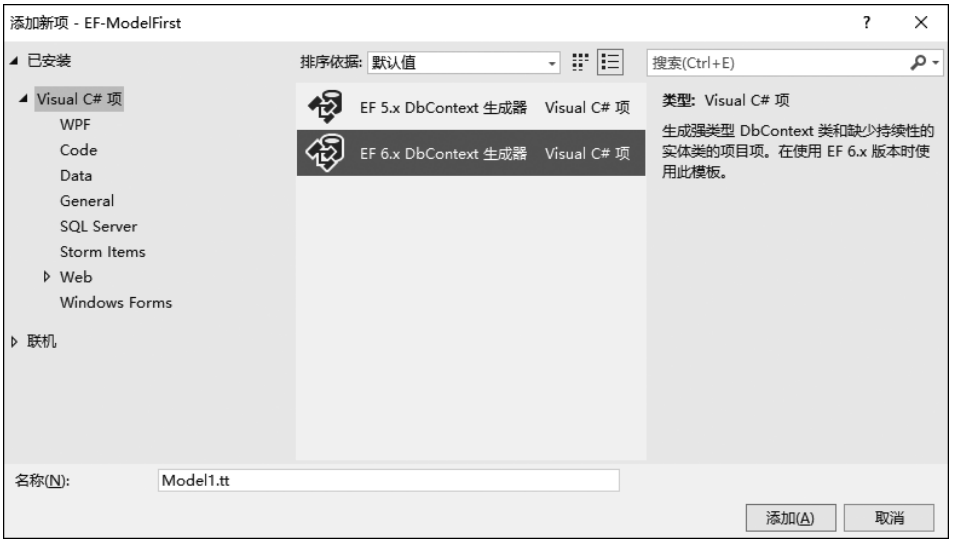

图 3.22　选择 EF 6.x DbContext 生成器

步骤 13：在 Designer 的空白处右击，选择“根据模型生成数据库”命令，如图 3.24 所示。

步骤 14：在“生成数据库向导”窗口单击“新建连接”按钮，按例 3-1 中数据库的连接步骤，连接到一个不含任何数据表的空白数据库 demo2(demo2 数据库需要读者自行创建)。

步骤 15：在“摘要和设置”选项中出现 DDL 项，显示数据库的定义语言命令画面，如图

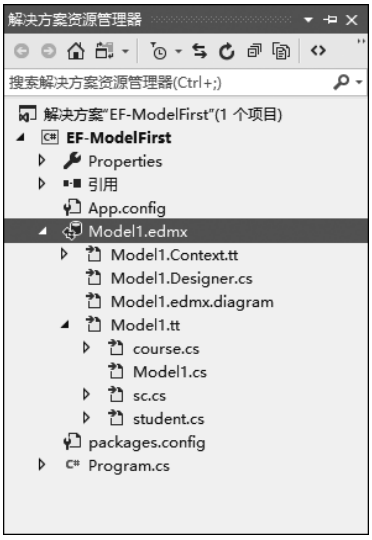

图 3.23 新增模型代码

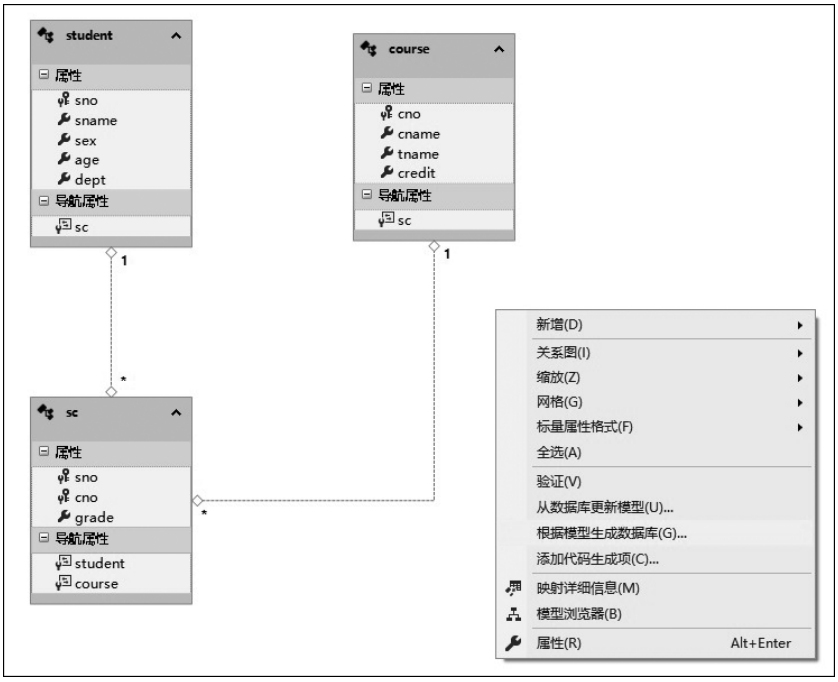

图 3.24 生成数据库操作

3.25 所示。若单击"上一步"按钮会回到"生成数据库"向导,单击"完成"按钮,Visual Studio 将自动生成 DDL 文件并打开,如图 3.26 所示。

步骤 16:直接单击左上角的▶按钮,可以将这个 DLL 送到 demo2 数据库执行,当执行完时,就可以看到数据库中创建了 studentSet、courseSet 以及 scSet 三张数据表,它们之间也设置了 Foreign Key 的关联。

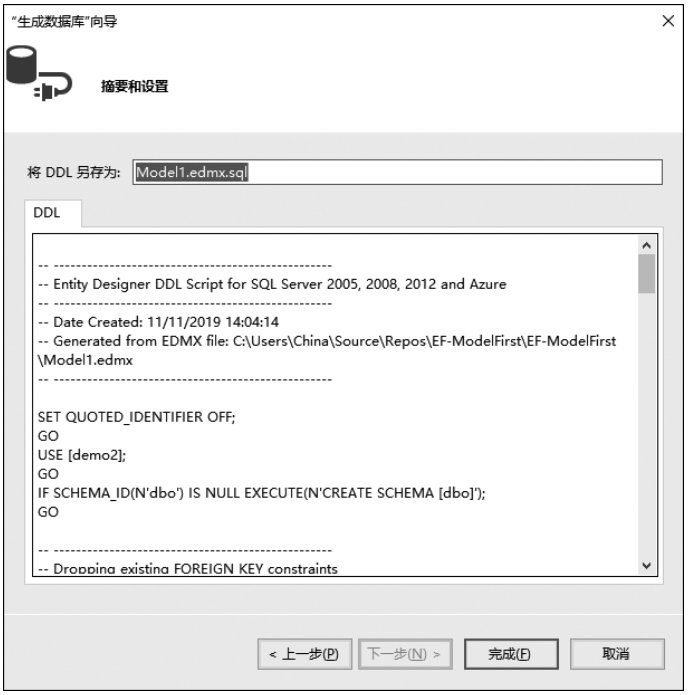

图 3.25 “摘要和设置”显示

```
45  -- Creating table 'studentSet'
46  CREATE TABLE [dbo].[studentSet] (
47      [sno] int IDENTITY(1,1) NOT NULL,
48      [sname] nvarchar(max)  NOT NULL,
49      [sex] nvarchar(max)  NOT NULL,
50      [age] int  NOT NULL,
51      [dept] nvarchar(max)  NOT NULL
52  );
53  GO
54
55  -- Creating table 'courseSet'
56  CREATE TABLE [dbo].[courseSet] (
57      [cno] int IDENTITY(1,1) NOT NULL,
58      [cname] nvarchar(max)  NOT NULL,
59      [tname] nvarchar(max)  NOT NULL,
60      [credit] int  NOT NULL
61  );
62  GO
63
64  -- Creating table 'scSet'
65  CREATE TABLE [dbo].[scSet] (
66      [sno] int IDENTITY(1,1) NOT NULL,
67      [cno] int  NOT NULL,
68      [grade] int  NOT NULL,
69      [student_sno] int  NOT NULL,
70      [course_cno] int  NOT NULL
71  );
72  GO
100 %
```

图 3.26 自动生成 DDL 命令

如果使用的 Visual Studio 中未安装 SQL Server Data Tools，则无法在 Visual Studio 中直接执行命令。此时，可以将 DLL 复制到 SQL Server Management Studio 中执行，同样可以完成数据库架构更新的操作。

3.2.3 Code First 模式


视频讲解

Code First 模式，即代码优先设计模式，是通过编写程序代码的方式来定义数据的结构，根据项目需求，撰写数据上下文类，程序运行时依据类创建数据表及相关属性，并转换成实体模型。在开发中没有特别的 GUI 工具，使用数据库的相关概念创建相关模型，适合熟悉传统 ADO.NET 开发的技术人员使用。

【例 3-3】 创建控制台应用程序，按 Demo 数据库中的数据表的结构设计模型，使用 EF 框架中的 Code First 模式，对应的在项目中生成模型及数据库文件。

步骤 1：创建控制台应用程序，命名为 EF-CodeFirst。

步骤 2：控制台应用程序上右击，选择“添加”→“新建项”选项。在“添加新项”窗口，选择 ADO.NET 实体数据模型并命名为 Model1，单击“添加”按钮。

步骤 3：在“实体数据模型向导”窗口的“选择模型内容”项中，选择空 Code First 模型，单击“完成”按钮，如图 3.27 所示。

图 3.27 模型选择

步骤 4：添加完成后，Visual Studio 会打开生成好的 Code First 程序代码 Model1。该类继承自 DbContext，相关的定义规则在类中均以注释代码的形式给出，简单修改后即可使用。

Model1.cs 源代码如下。

```
public class Model1 : DbContext
    {
        //你的上下文已配置为从你的应用程序的配置文件(App.config 或 Web.config)
        //使用 Model1 连接字符串。默认情况下,此连接字符串针对你的 LocalDb 实例上的
        //EF_CodeFirst.Model1 数据库。
        //
        //如果想要针对其他数据库和/或数据库提供程序,请在应用程序配置文件中修改 Model1
        //连接字符串。
        public Model1()
            : base("name=Model1")
        {
        }
        //为你要在模型中包含的每种实体类型都添加 DbSet。有关配置和使用 Code First 模型
        //的详细信息,请参阅 http://go.microsoft.com/fwlink/?LinkId=390109。
        // public virtual DbSet<MyEntity>  MyEntities { get; set; }
    }
    //public class MyEntity
    //{
    //    public int Id { get; set; }
    //    public string Name { get; set; }
    //}
```

步骤 5：修改 Model1.cs 文件，向 demo3 数据库创建一个 MyStudents 学生表，结构与 student 表一致，编辑 Model1.cs 文件的 Model1 类代码如下。

```
public class Model1 : DbContext
{
    public Model1()
        : base("name=Model1")
    {
    }
    public virtual DbSet<Student>  MyStudents { get; set; }
}
```

新建 Student 类，编辑代码如下。

```
public class Student
{
    public int Id{ get; set; }          //int 类型的 Id 属性默认为实体表的主键
    public string Sname { get; set; }
    public string Sex { get; set;}
    public int Age { get; set; }
    public string Dept { set; get; }
}
```

使用编辑代码的方式编写 Code First 模型时，如果未使用系统自动创建的类，则需要在 App.config 或者 Web.config 中加入 EF 相关的配置，否则程序无法自动实现数据库的访问。

步骤 6：在 Code First 模式生成模型时，默认会使用 SQL Server Express LocalDb 作为目标的数据库服务器，如果要使用 demo3 数据库，需要对 App.config 中的属性进行设置。

原设置如下。

```
<defaultConnectionFactory type="System.Data.Entity.Infrastructure.
SqlConnectionFactory, EntityFramework" />
<connectionStrings>
<add name = " Model1 " connectionString = " data source = (LocalDb) \ MSSQLLocalDB;
initial catalog=EF_CodeFirst.Model1;integrated security=True;
MultipleActiveResultSets=True; App = EntityFramework " providerName = " System.
Data.SqlClient" />
</connectionStrings>
```

修改 App.config 文件部分设置如下。

```
<defaultConnectionFactory type="System.Data.Infastructure.
SqlConnectionFactory,EntityFramework">
<parameters>
< parameter value = " Data Srouce = . \ sqlexpress; Intergrated Security = True;
MultipleActiveResultSets=True" />
</parameters>
</defaultConnectionFactory>
<connectionStrings>
<add name="Model1" connectionString="data source=.\sqlexpress;initial catalog=
demo3;integrated security=True;MultipleActiveResultSets=True;
App=EntityFramework" providerName="System.Data.SqlClient" />
</connectionStrings>
```

步骤 7：在 Program.cs 文件的主程序中编写程序代码如下。

```
class Program
{
    static void Main(string[] args)
    {
        Model1 model = new Model1();
        model.Database.CreateIfNotExists();
    }
}
```

步骤 8：执行应用程序，成功后查询数据库中数据表，如图 3.28 所示。

3.2.4 App.config 的相关设置

在 3.2.3 节中介绍了 Code First 模式生成数据库时，需要在 App.config 中进行相关的

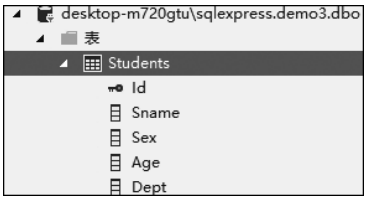

图 3.28　创建生成 student 表

设置。当需要变更数据库的类型时，需要对＜deafultConnectionFactory＞中的属性进行相关设置。当需要变更数据库名称或连接账户时，需要对＜connectionStrings＞内的连接字符串进行设置，常用的设置如下。

1. 设置数据库类型为 SQL Server（非 LocalDb）

```
< defaultConnectionFactory type = " System. Data. Infastructure. SqlConnectionFactory,
EntityFramework">
<parameters>
< parameter value = " Data Srouce = MyDatabaseServer; Intergrated Security = True;
MultipleActiveResultSets=True" />
</parameters>
</defaultConnectionFactory>
```

2. 设置数据库类型为 SQL Server Compact

```
<defaultConnectionFactory type="System.Data.Entity.Infastructure.
SqlCeConnectionFactory,EntityFramework">
<parameters>
<parameter value="System.Data.SqlServerCe.4.0" />
</parameters>
</defaultConnectionFactory>
```

3. 设置数据库类型为 SQL Server Express LocalDb

```
<defaultConnectionFactory type="System.Data.Entity.Infastructure.
LocalDbConnectionFactory,EntityFramework">
<parameters>
<parameter value="v12.0" />
</parameters>
</defaultConnectionFactory>
```

4. 设置 Code First 的连接字符串

```
<connectionStrings>
<add name="BlogContext" providerName="System.Data.SqlClient" connectionString=
"Server=(local);Database=Blogs;IntegratedSecurity=True;" />
</connectionStrings>
```

5. 设置 Database First/Model First 的连接字符串

```
<connectionStrings>
<add name="BlogContext" connectionString="metadata=res://*/BloggingModel.
csdl|res://*/BloggingModel.ssdl|res://*/BloggingModel.msl;provider=System.
Data.SqlClient
provider connection string=&quot;data source=(localdb)\v11.0;initial catalog=
Blogs;integratedsecurity=True;multipleactiveresultsets=True;&quot;
"providerName="System.Data.EntityClient" />
</connectionStrings>
```

3.2.5 由数据库生成模型

对已经存在的数据库也可以直接使用 Code First 模型，基本方法是在 ADO.NET 实体数据模型向导中选择“来自数据库的 Code First”选项，如图 3.29 所示。其他的操作和 Database First 模式的设计步骤基本相同。只是在 Code First 模式的设计对象中不支持存储过程和函数，如图 3.30 所示。

图 3.29 实体模型选择