In Chapter 1, you were introduced to modern data access paradigms. To help you remember these, there are three primary strategies for designing a data access layer in your application.

**Database First:** Start by designing the database (or use a preexisting database as source) and then create an entity container from the database using a conceptual model generator tool. The entity container models will represent the various tables of the relational database and associations representing the relationships. You can then query and manipulate the models and let the tool perform the CRUD operations behind the scenes. This approach is of increasing importance if your application has an existing database with which it can connect and from which it can fetch information.

**Model First:** You start by designing the conceptual model using the available Designer tools, and then you can either generate the database from the conceptual models or map the data context generated by the tool to a preexisting database.

**Code First:** This is a developer-friendly approach where you create the conceptual model in code and then use proprietary APIs to map the model to a physical database.

These paradigms are well supported by the ADO.NET Entity Framework, and this strong support is one of the main reasons behind the growing popularity of Entity Framework in creating robust data access layers in a wide variety of applications.

In the previous chapter, we focused on learning the nuances of Entity Framework, and for the most part you used the Database-First paradigm. This chapter serves as an extension of what you learned previously. Beyond that, you will learn more about the other two paradigms: Model First and Code First.

Specifically, we are going to cover the following in this chapter:

- How to program data access using the Model-First approach.
- How to program data access using the Code-First approach.
- How to perform validations using Data Annotations.

Although you learned about these paradigms in Chapter 1, now you will learn to implement them practically using Entity Framework.

---

**Note** At the time of this writing, the latest version of Entity Framework was EF 5.0. We have leveraged EF 5.0 for executing the exercises in this chapter.
Model-First Approach

In the previous chapter, you learned how to leverage Entity Framework to build a data access layer in your application quickly and effectively. In the examples in that chapter, you created the Entity Data Model (EDM) by generating it from the existing Store database. The tables Contacts and Orders showed up in the EDM Designer (the edmx file) as entities. The entity associations were also generated automatically using the foreign key relationship between the tables.

This is a reasonable approach if you are working against a database that already exists, and you just need to build an application around it. However, if you are creating a completely new stack from application to database, it is quite likely that you would want to create a visual model and then create the database based on the visual model.

Wouldn’t it be great if you could generate the database automatically from the visual model? The Model-First approach lets you do just that. You could model the database in your application and then evolve the model as the application grows. Behind the scenes, Entity Framework will map your model to a physical database based on the EDM conventions. Application developers who are not well versed in relational databases find this approach very useful. They often embrace it for applications that are built from the ground up, along with the storage schema. The benefit of using Entity Framework is that it is capable of translating the model into the database schema using the underlying providers, and it generates the database with one script. Let’s try performing the steps designed to create an application data access layer using the Entity Framework Model-First approach.

**TRY IT OUT: CREATING A DATA-DRIVEN WEB APPLICATION WITH THE ENTITY FRAMEWORK MODEL-FIRST PARADIGM**

You will use the most recent version of the very popular development tool Visual Studio to do this exercise. You will learn to model the Store database that we have been discussing throughout the book using the EDM Designer, and you will then use it to create the database tables and relationships. Follow these steps:

1. Open Visual Studio 2012, and create a Visual C# ASP.NET MVC 4 Web Application project. The project template wizard is displayed. Choose the Internet Application template, as shown in Figure 8-1.