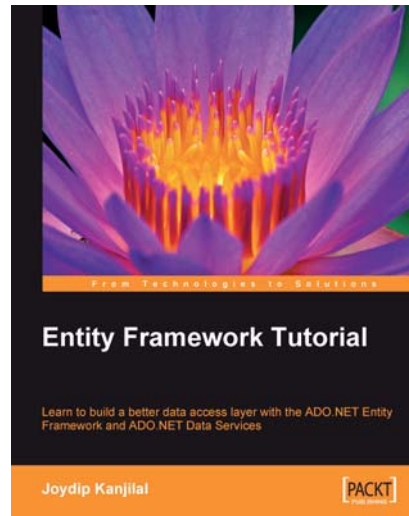




Entity Framework Tutorial

Joydip Kanjilal



Chapter No. 2 "Getting Started"

In this package, you will find:

A Biography of the author of the book

A preview chapter from the book, Chapter NO.2 "Getting Started"

A synopsis of the book's content

Information on where to buy this book

About the Author

Joydip Kanjilal is a Microsoft MVP in ASP.NET. He has over 12 years of industry experience in IT with more than 6 years in Microsoft .NET and its related technologies. He has authored many articles for some of the most reputable sites like, www.asptoday.com, www.devx.com, www.aspalliance.com, www.aspnetpro.com, www.sql-server-performance.com, www.sswug.com, etc. Several of these articles have been featured at www.asp.net—Microsoft's Official Site on ASP.NET. Joydip was also a community credit winner at www.community-credit.com a number of times.

He is currently working as a Senior Consultant in a reputable company in Hyderabad, INDIA. He has years of experience in designing and architecting solutions for various domains. His technical strengths include C, C++, VC++, Java, C#, Microsoft .NET, Ajax, Design Patterns, SQL Server, Operating Systems, and Computer Architecture. Joydip blogs at <http://aspadvice.com/blogs/joydip> and spends most of his time reading books, blogs, and writing books and articles. His hobbies include watching cricket and soccer and playing chess.

For More Information: www.packtpub.com/entity-framework-tutorial/book

Writing a book is always a rewarding experience. My special thanks to Douglas Paterson for providing me the opportunity to author this book—turning this idea into a reality. I am also thankful to the entire Packt team for their support.

I am also thankful to Abhishek Kant (Microsoft), Steve Smith (AspAlliance), Russell Jones (DevX), Steve Jones (SSWUG), Jude Kelly (SQL Server Performance), and Anand Narayaswamy (AspAlliance) for their inspiration and support. My heartiest thanks to my friends Tilak and Vinod for their continued encouragement.

My deepest respects and gratitude to my parents for their love, blessings, and encouragement. My thanks to my other family members too, for their support, and to little Jini in particular, for her continued inspiration and love.

Thank you all so much!

Entity Framework Tutorial

The ADO.NET Entity Framework, the next generation of Microsoft's data access technology, is an extended Object Relational Mapping (ORM) technology that makes it easy to tie together the data in your database with the objects in your applications. This is done by abstracting the object model of an application from its relational or logical model. It is an extended ORM in the sense that it provides many additional features over an ORM. Some of these features are:

- Entity Inheritance and Composition
- Identity Resolution and Change Tracking
- LINQ Support
- The Object Service Layer

This book is a clear and concise guide to the ADO.NET Entity Framework. Packed with plentiful code examples, this book helps you to learn the ADO.NET Entity Framework and ADO.NET Data Services and build a better data access layer for your application.

What This Book Covers

Chapter 1 is an introduction to the basics of the ADO.NET Entity Framework (EF), its usefulness, its features, and the benefits.

Chapter 2 discusses how you can get started with EF, create an Entity Data Model (EDM), and write a program to query data.

Chapter 3 gives a detailed explanation of entities, relationships, and each of the sections of the EDM.

Chapter 4 provides a sample application that illustrates how to perform CRUD operations against the EDM.

Chapter 5 discusses the Entity SQL query language and how to work with the Entity Client provider.

Chapter 6 includes a detailed discussion on LINQ to Entities with many code examples.

Chapter 7 provides a detailed discussion on the Object Services Layer and its helpful and useful features.

Chapter 8 provides an introduction to ADO.NET Data Services and how it can be used with the EDM to perform CRUD operations.

For More Information: www.packtpub.com/entity-framework-tutorial/book

2

Getting Started

In the previous chapter we took a look at the ADO.NET Entity Framework including its architecture and its features. We also designed our Payroll database that we will be using throughout this book to store and retrieve data. We will use the same database in this chapter to generate an Entity Data Model and then use it, along with the Entity Data Source control, to bind data to a GridView data control.

In this chapter, we will cover the following points:

- Creating an Entity Data Model
- Introducing the Entity Data Source Control
- Implementing our first application using the ADO.NET Entity Framework

We will start this chapter with a discussion on how we can create an Entity Data Model from our Payroll database.

Creating an Entity Data Model

You can create the ADO.NET Entity Data Model in one of two ways:

- Use the ADO.NET Entity Data Model Designer
- Use the command line Entity Data Model Designer called EdmGen.exe

We will first take a look at how we can design an Entity Data Model using the ADO.NET Entity Data Model Designer which is a Visual Studio wizard that is enabled after you install ADO.NET Entity Framework and its tools. It provides a graphical interface that you can use to generate an Entity Data Model.

For More Information: www.packtpub.com/entity-framework-tutorial/book

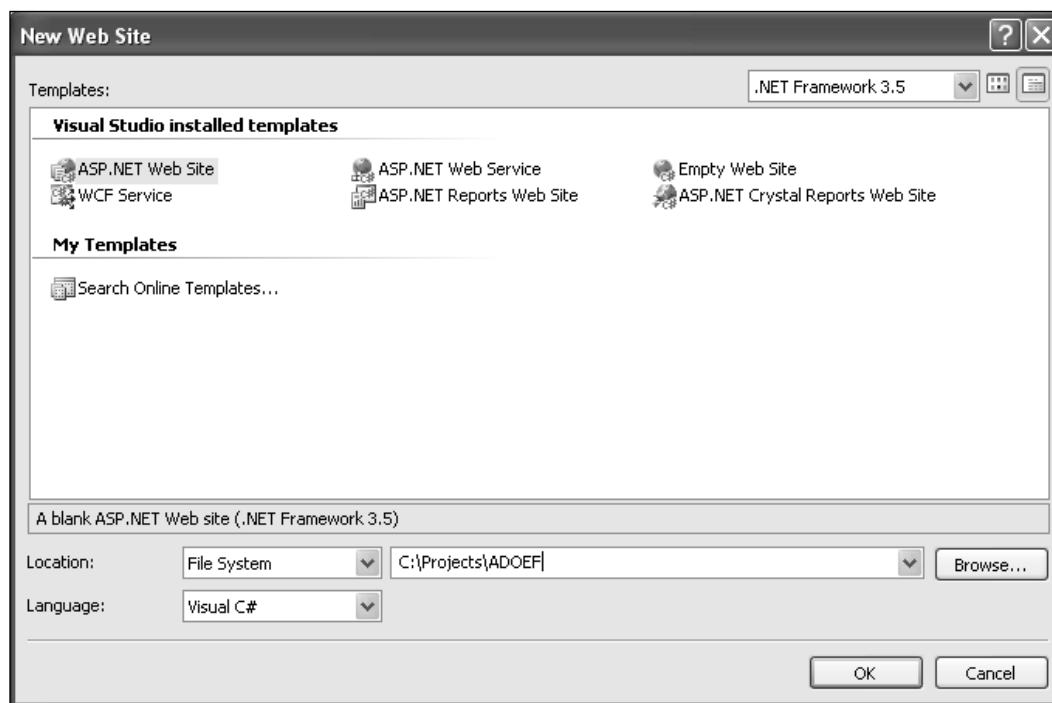
Creating the Payroll Entity Data Model Using the ADO.NET Entity Data Model Designer

Here again are the tables of our Payroll database that we will use to generate the data model:

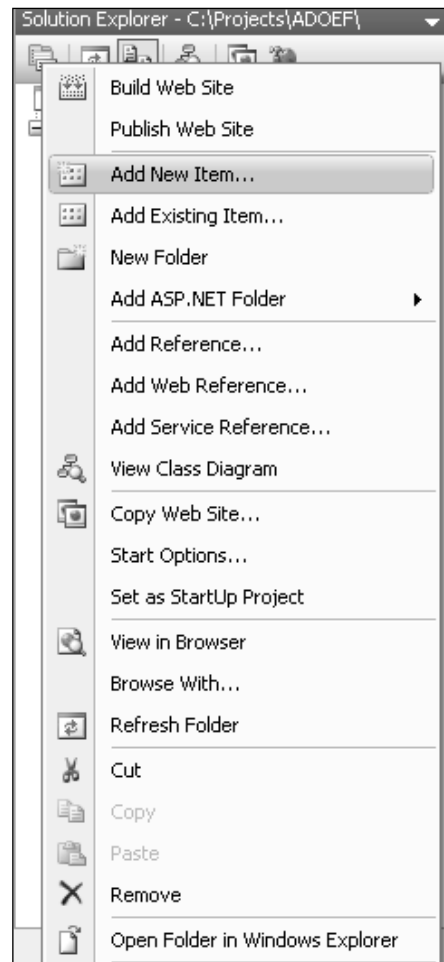
- Employee
- Designation
- Department
- Salary
- ProvidentFund

To create an entity data model using the ADO.NET Entity Data Model Designer, follow these simple steps:

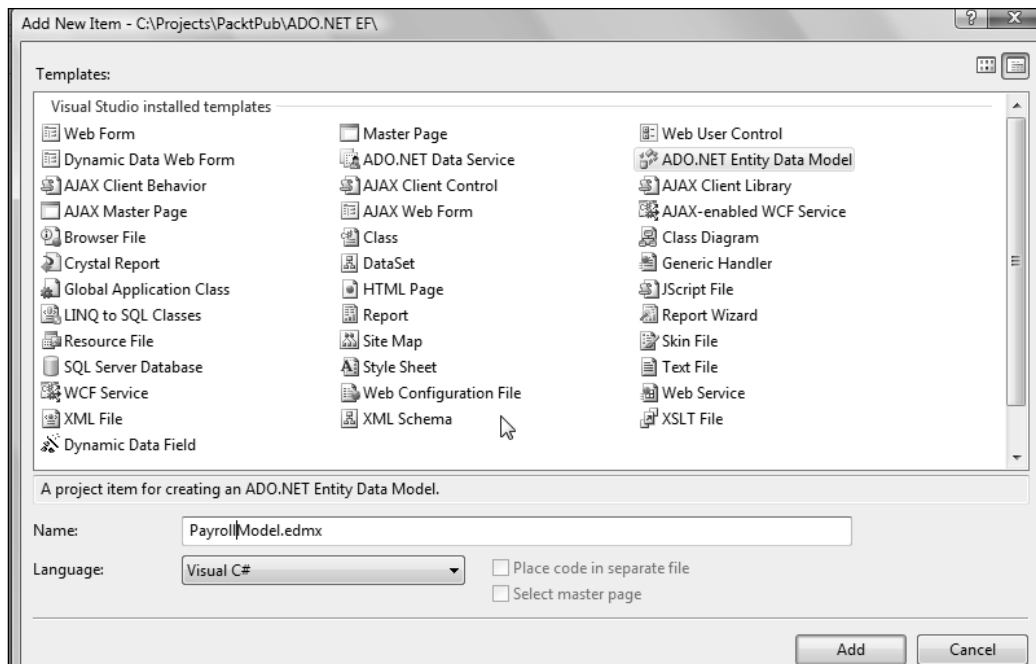
1. Open Visual Studio.NET and create a solution for a new web application project as seen below and save with a name.



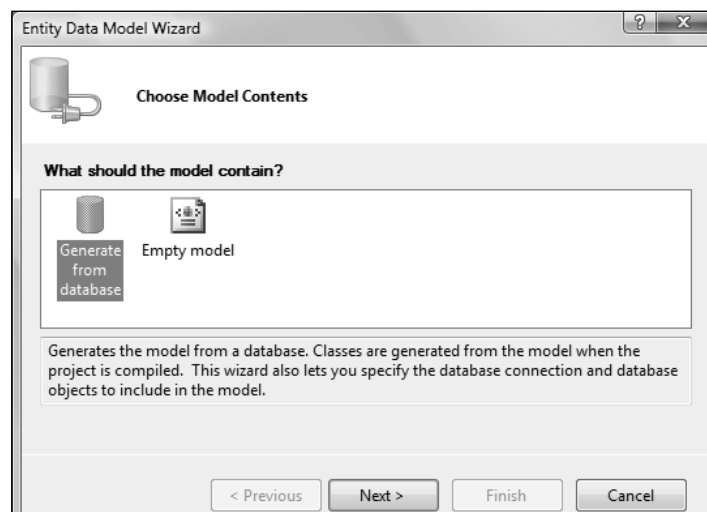
2. Switch to the Solution Explorer, right click and click on **Add New Item** as seen in the following screenshot:



- Next, select **ADO.NET Entity Data Model** from the list of the templates displayed as shown in the following screenshot:

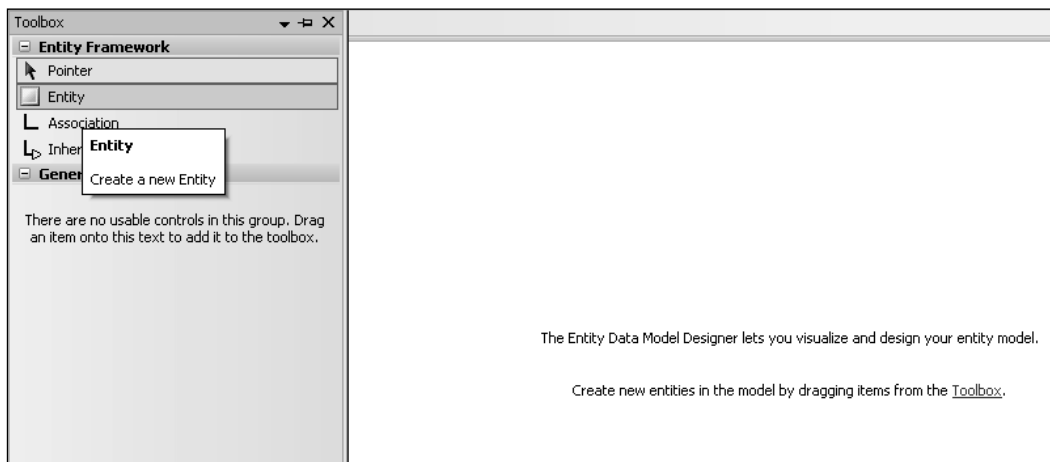


- Name the Entity Data Model **PayrollModel** and click on **Add**.
- Select **Generate from database** from the Entity Data Model Wizard as shown in the following screenshot:



Note that you can also use the Empty model template to create the Entity Data Model yourself.

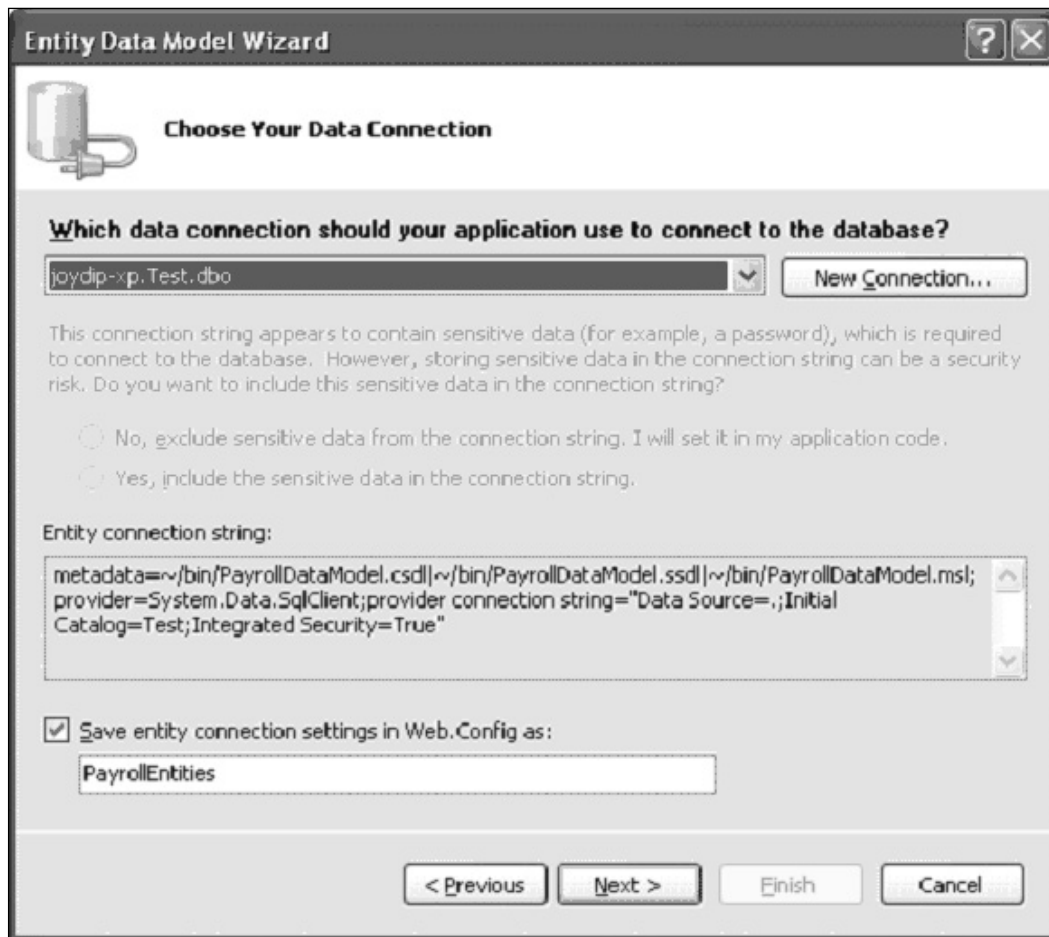
If you select the Empty Data Model template and click on next, the following screen appears:



As you can see from the above figure, you can use this template to create the Entity Data Model yourself. You can create the Entity Types and their relationships manually by dragging items from the toolbox. We will not use this template in our discussion here. So, let's get to the next step.

6. Click on **Next** in the Entity Data Model Wizard window shown earlier.

- The modal dialog box will now appear and prompts you to choose your connection as shown in the following figure:



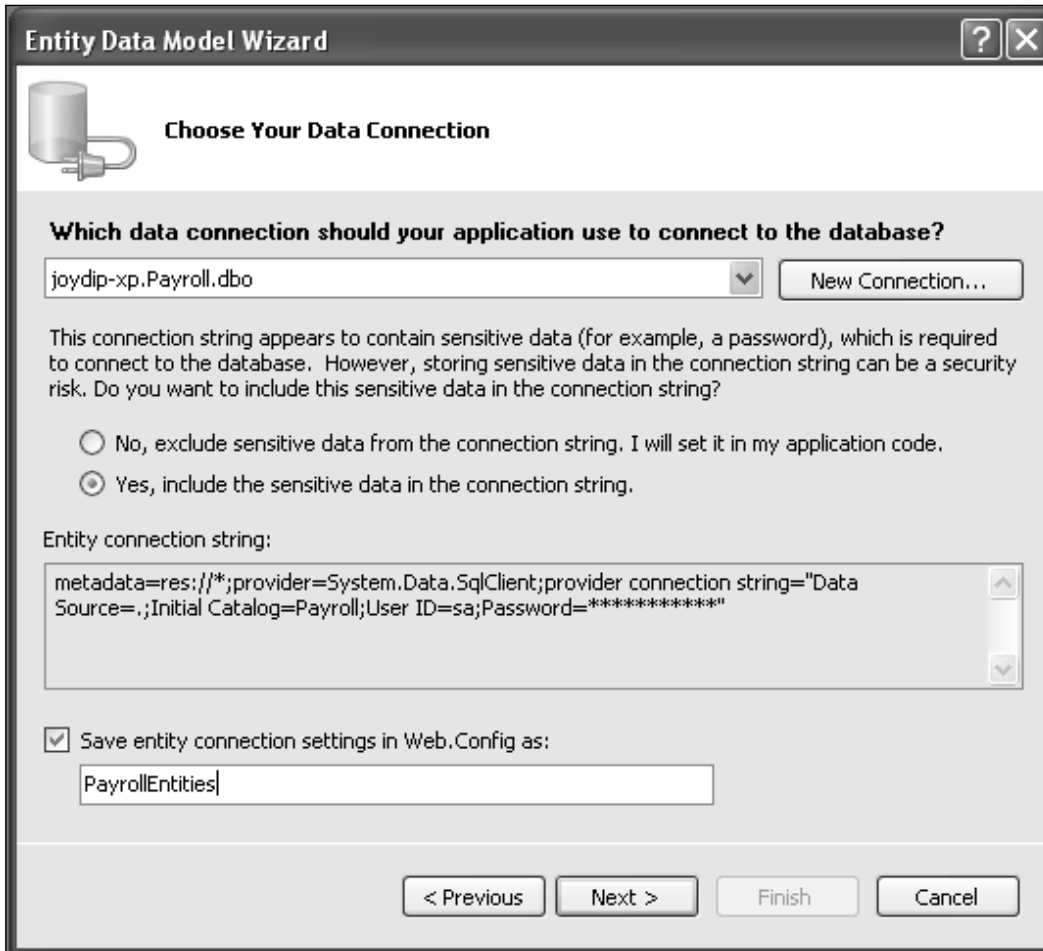
- Click on **New Connection** Now you will need to specify the connection properties and parameters as shown in the following figure:



We will use a dot to specify the database server name. This implies that we will be using the database server of the localhost, which is the current system in use.

9. After you specify the necessary user name, password, and the server name, you can test your connection using the **Test Connection** button. When you do so, the message **Test connection succeeded** gets displayed in the message box as shown in the previous figure.

10. When you click on **OK on the Test connection dialog box**, the following screen appears:

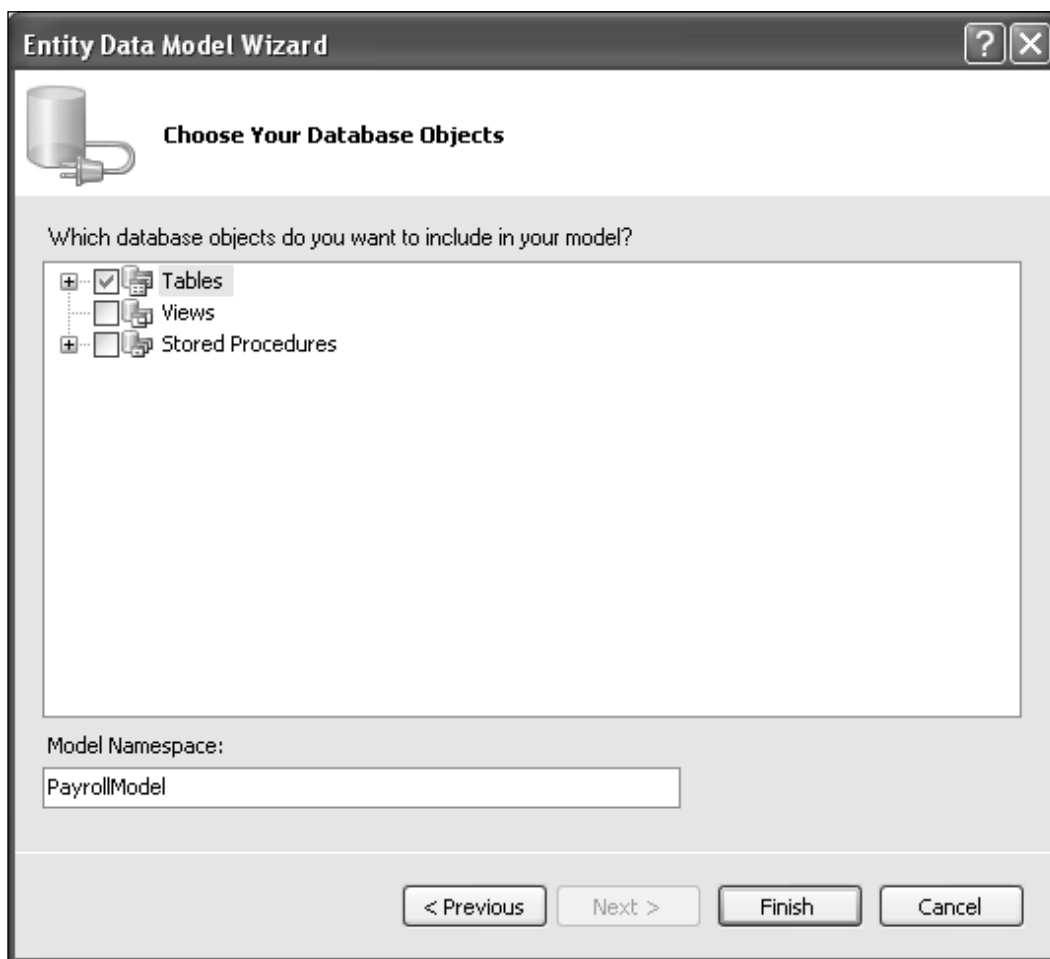


The image shows the 'Entity Data Model Wizard' dialog box, specifically the 'Choose Your Data Connection' step. The title bar reads 'Entity Data Model Wizard'. Below the title bar is a cylinder icon and the text 'Choose Your Data Connection'. The main question is 'Which data connection should your application use to connect to the database?'. A dropdown menu shows 'joydip-xp.Payroll.dbo' and a 'New Connection...' button is to its right. Below this, a warning message states: 'This connection string appears to contain sensitive data (for example, a password), which is required to connect to the database. However, 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.' The 'Yes' option is selected. Below the radio buttons, the text 'Entity connection string:' is followed by a text box containing the connection string: 'metadata=res://*;provider=System.Data.SqlClient;provider connection string="Data Source=.;Initial Catalog=Payroll;User ID=sa;Password=*****"'. At the bottom, there is a checkbox 'Save entity connection settings in Web.Config as:' which is checked, followed by a text box containing 'PayrollEntities'. At the very bottom are four buttons: '< Previous', 'Next >', 'Finish', and 'Cancel'.

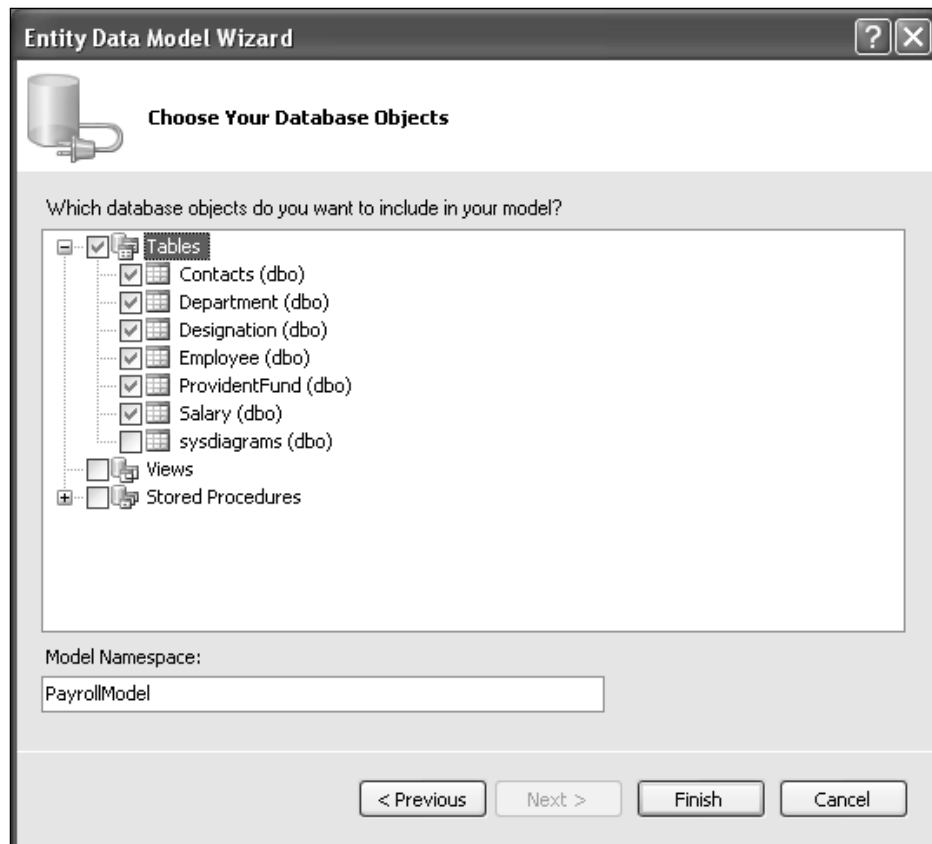
Note the Entity Connection String generated automatically. This connection string will be saved in the ConnectionStrings section of your application's web.config file. This is how it will look like:

```
<connectionStrings>
  <add name="PayrollEntities" connectionString="metadata=res://
    *;provider=System.Data.SqlClient;provider connection
    string=&quot;Data Source=.;Initial Catalog=Payroll;User
    ID=sa;Password=joydip1@3;MultipleActiveResultSets=True&quot;;"
    providerName="System.Data.EntityClient" />
</connectionStrings>
```

11. When you click on **Next** in the previous figure, the following screen appears:

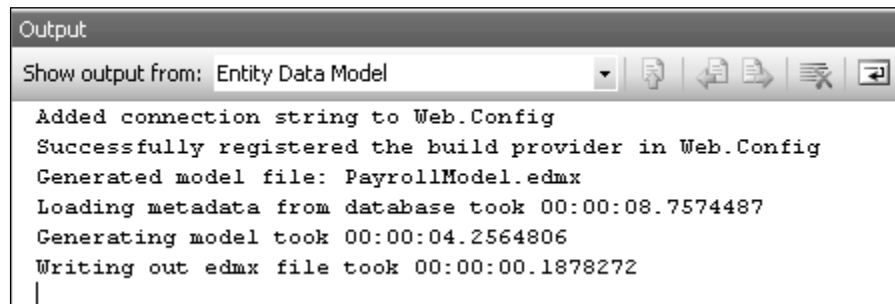


12. Expand the **Tables** node and specify the database objects that you require in the Entity Data Model to be generated as shown in the following figure:



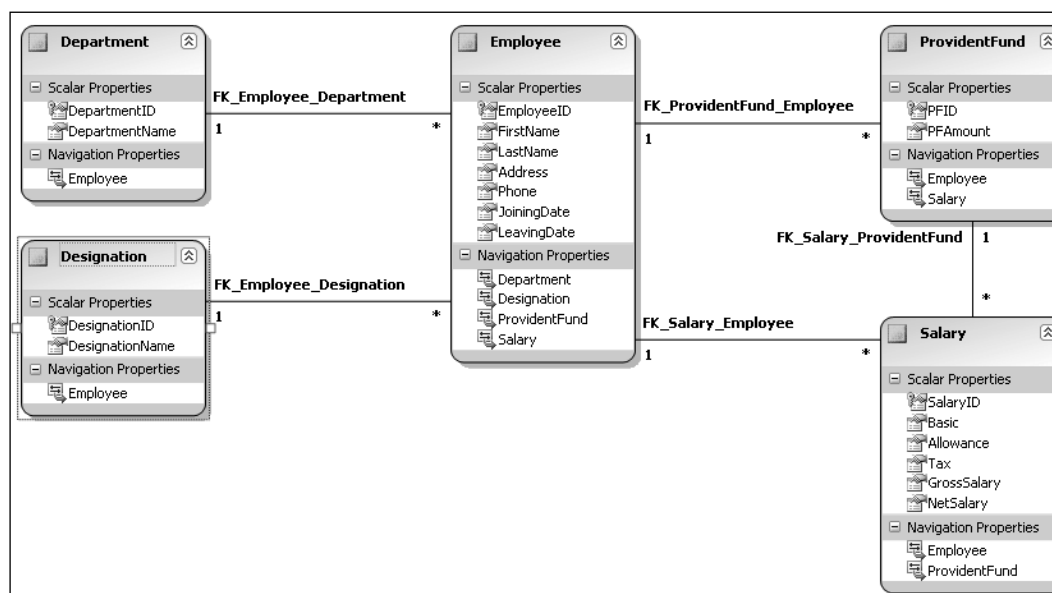
13. Click on **Finish** to generate the Entity Data Model.

Here is the output displayed in the Output Window while the Entity Data Model is being generated:



Your Entity Data Model has been generated and saved in a file named `PayrollModel.edmx`. We are done creating our first Entity Data Model using the ADO.NET Entity Data Model Designer tool.

When you open the Payroll Entity Data Model that we just created in the designer view, it will appear as shown in the following figure:



Note how the Entity Types in the above model are related to one another. These relationships have been generated automatically by the Entity Data Model Designer based on the relationships between the tables of the Payroll database we created in the previous chapter.

In the next section, we will learn how we can create an Entity Data Model using the `EdmGen.exe` command line tool.

Creating the Payroll Data Model Using the EdmGen Tool

We will now take a look at how to create a data model using the Entity Data Model generation tool called **EdmGen**.

The EdmGen.exe command line tool can be used to do one or more of the following:

- Generate the .csdl, .msl, and .ssdl files as part of the Entity Data Model
- Generate object classes from a .csdl file
- Validate an Entity Data Model

The EdmGen.exe command line tool generates the Entity Data Model as a set of three files: .csdl, .msl, and .ssdl. If you have used the ADO.NET Entity Data Model Designer to generate your Entity Data Model, the .edmx file generated will contain the CSDL, MSL, and the SSDL sections. You will have a single .edmx file that bundles all of these sections into it. On the other hand, if you use the EdmGen.exe tool to generate the Entity Data Model, you would find three distinctly separate files with .csdl, .msl or .ssdl extensions.

Here is a list of the major options of the EdmGen.exe command line tool:

Option	Description
/help	Use this option to display help on all the possible options of this tool. The short form is /?
/language:CSharp	Use this option to generate code using C# language
/language:VB	Use this option to generate code using VB language
/provider:<string>	Use this option to specify the name of the ADO.NET data provider that you would like to use.
/connectionstring:<connection string>	Use this option to specify the connection string to be used to connect to the database
/namespace:<string>	Use this option to specify the name of the namespace
/mode:FullGeneration	Use this option to generate your CSDL, MSL, and SSDL objects from the database schema
/mode:EntityClassGeneration	Use this option to generate your entity classes from a given CSDL file
/mode:FromSsdGeneration	Use this option to generate MSL, CSDL, and Entity Classes from a given SSDL file
/mode:ValidateArtifacts	Use this option to validate the CSDL, SSDL, and MSL files
/mode:ViewGeneration	Use this option to generate mapping views from the CSDL, SSDL, and MSL files

Option	Description
/entitycontainer:<string>	Use this option to specify the name of the Entity Container to be used in the conceptual model
/project:<string>	Use this option to specify the base name to be used for all the artifact files (.csdl, .msl, .ssdl) to be generated. The short form of this option is /p

Note that you basically need to pass the connection string, specify the mode, and also the project name of the artifact files (.csdl, .msl, and the .ssdl files) to be created. To create the Entity Data Model for our database, open a command window and type in the following:

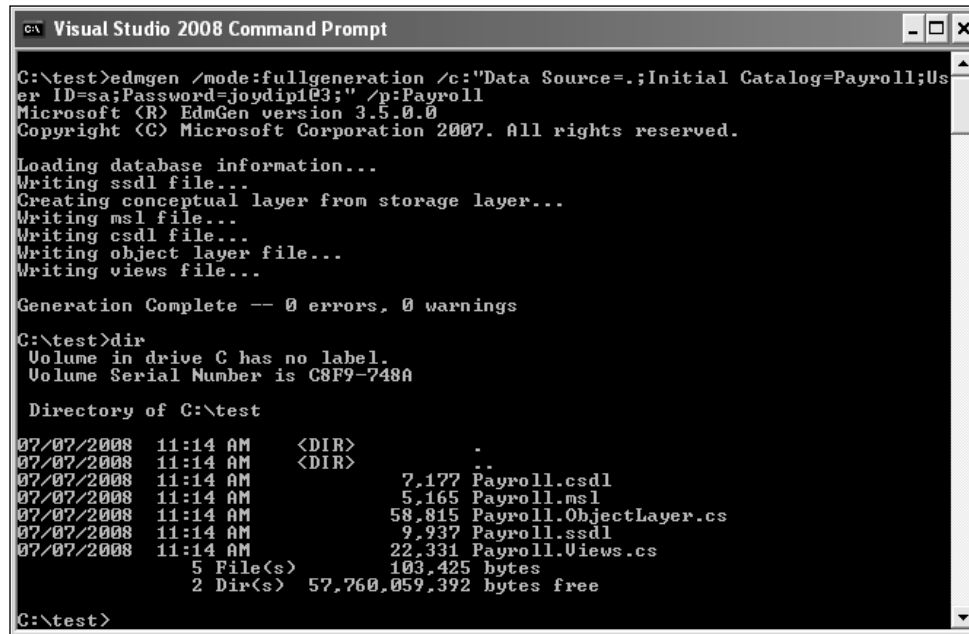
```
edmgen /mode:fullgeneration /c:"Data Source=.;Initial
Catalog=Payroll;User ID=sa;Password=joydip1@3;" /p:Payroll
```

This will create a full ADO.NET Entity Data Model for our database. The output is shown in the following figure:



```
C:\test>Visual Studio 2008 Command Prompt
C:\test>edmgen /mode:fullgeneration /c:"Data Source=.;Initial Catalog=Payroll;User ID=sa;Password=joydip1@3;" /p:Payroll
Microsoft (R) EdmGen version 3.5.0.0
Copyright (C) Microsoft Corporation 2007. All rights reserved.
Loading database information...
Writing ssdl file...
Creating conceptual layer from storage layer...
Writing msl file...
Writing csdl file...
Writing object layer file...
Writing views file...
Generation Complete -- 0 errors, 0 warnings
C:\test>_
```

You can now see the list of the files that have been generated:



```
C:\test>edmgen /mode:fullgeneration /c:"Data Source=.;Initial Catalog=Payroll;User ID=sa;Password=joydip1@3;" /p:Payroll
Microsoft (R) EdmGen version 3.5.0.0
Copyright (C) Microsoft Corporation 2007. All rights reserved.

Loading database information...
Writing ssdl file...
Creating conceptual layer from storage layer...
Writing msl file...
Writing csdl file...
Writing object layer file...
Writing views file...

Generation Complete -- 0 errors, 0 warnings

C:\test>dir
Volume in drive C has no label.
Volume Serial Number is C8F9-748A

Directory of C:\test

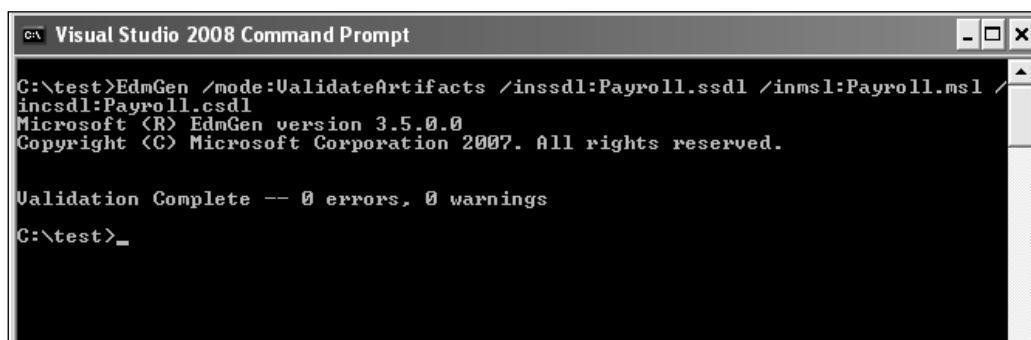
07/07/2008  11:14 AM    <DIR>          .
07/07/2008  11:14 AM    <DIR>          ..
07/07/2008  11:14 AM                7,177 Payroll.csdl
07/07/2008  11:14 AM                5,165 Payroll.msl
07/07/2008  11:14 AM            58,815 Payroll.ObjectLayer.cs
07/07/2008  11:14 AM                9,937 Payroll.ssdl
07/07/2008  11:14 AM            22,331 Payroll.Views.cs
               5 File(s)          103,425 bytes
               2 Dir(s)  57,760,059,392 bytes free

C:\test>
```

You can validate the Payroll Entity Data Model that was just created, using the `ValidateArtifacts` option of the EdmGen command line tool as shown below:

```
EdmGen /mode:ValidateArtifacts /inssdl:Payroll.ssdl /inmsl:Payroll.msl /
incsd1:Payroll.csdl
```

When you execute the above command, the output will be similar to what is shown in the following figure:



```
C:\test>EdmGen /mode:ValidateArtifacts /inssdl:Payroll.ssdl /inmsl:Payroll.msl /
incsd1:Payroll.csdl
Microsoft (R) EdmGen version 3.5.0.0
Copyright (C) Microsoft Corporation 2007. All rights reserved.

Validation Complete -- 0 errors, 0 warnings

C:\test>_
```

As you can see in the previous figure, there are no warnings or errors displayed. So, our Entity Data Model is perfect.

The section that follows discusses the new Entity Data Source control which was introduced as part of the Visual Studio.NET 2008 SP1 release.

The ADO.NET Entity Data Source Control

Data controls are those that can be bound to data from external data sources. These data sources may include databases, XML files, or even flat files. ASP.NET 2.0 introduced some data source controls with a powerful data binding technique so the need for writing lengthy code for binding data to data controls has been eliminated.

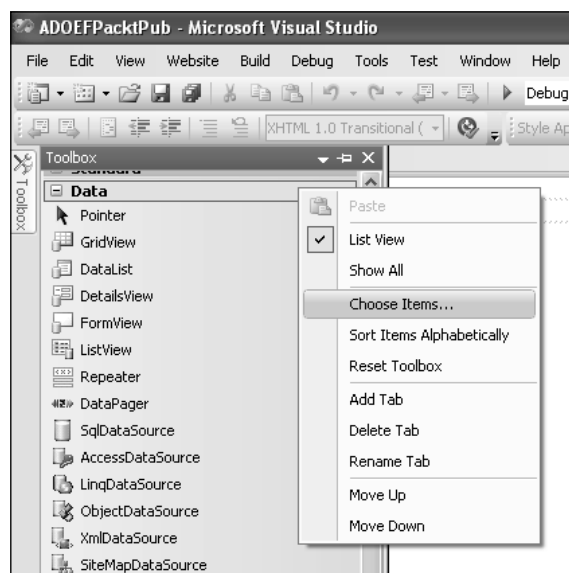


In ASP.NET, the term Data Binding implies binding the controls to data retrieved from a data source and providing a read or write connectivity between these controls and the data that they are bound to.

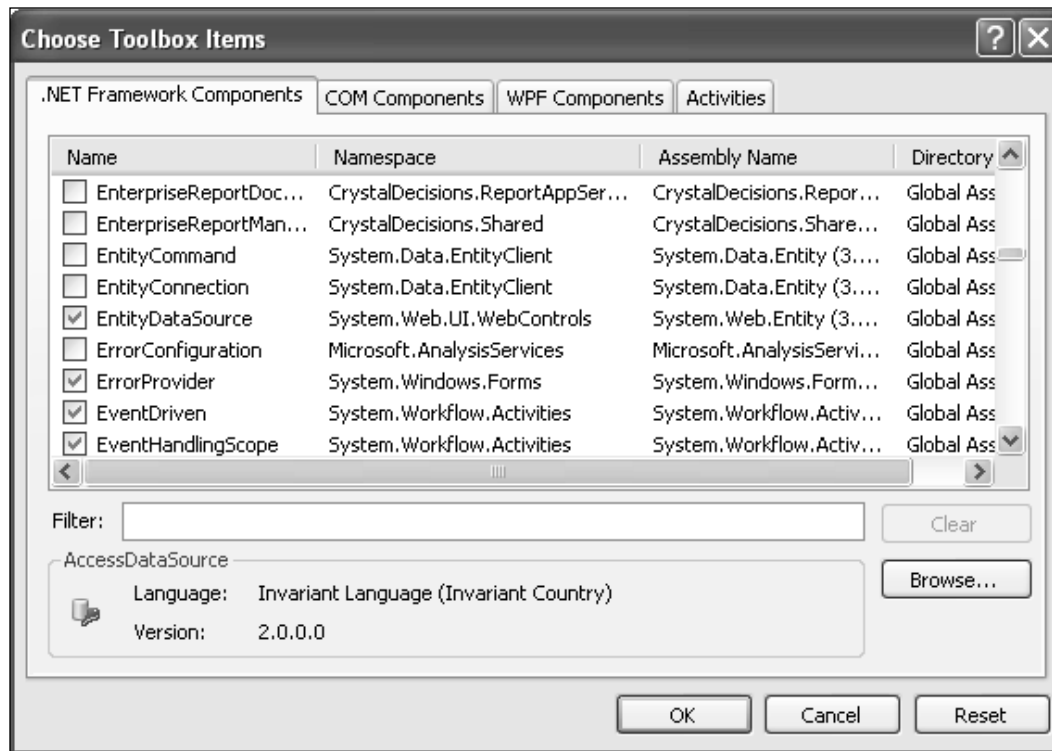
The Entity Data Source control is an example of a data control that is included as part of the Visual Studio 2008 SP1 release and can be used to bind data retrieved from an Entity Data Model to the data bound controls of ASP.NET. If you have installed Visual Studio 2008 SP1, you can see the EntityDataSource control listed in the Data section of your toolbox.

If you cannot locate the EntityDataSource control in the toolbox, follow these steps:

1. Right-click on the **Toolbox** and select the **Choose Items** option as shown in the following figure:

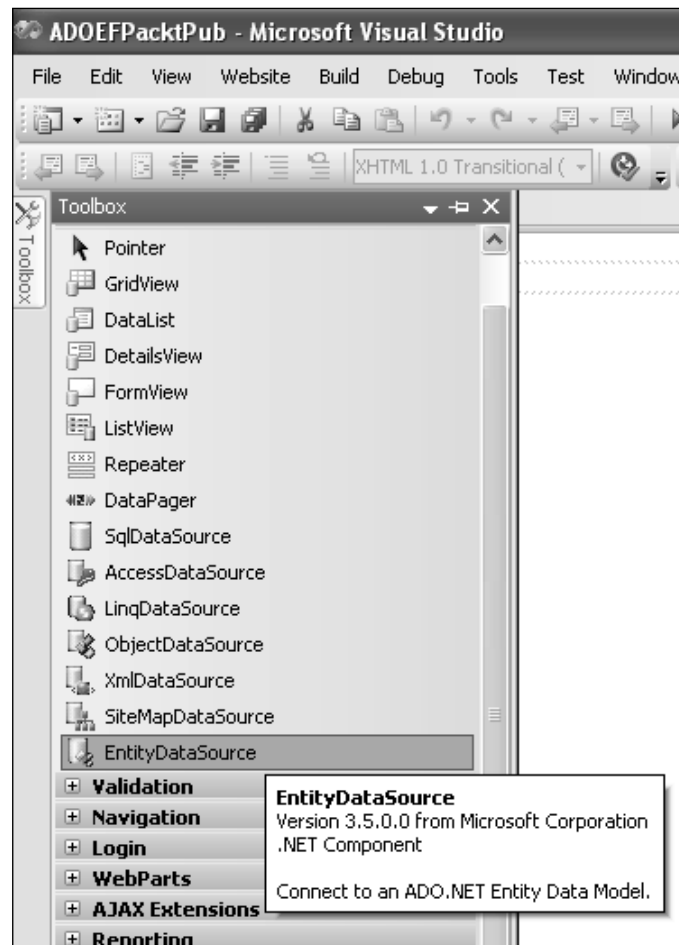


- From the list of the components displayed, scroll down to locate the EntityDataSource in the .NET Framework Components tab. Refer to the following figure:



- Now, check the checkbox next to the EntityDataSource component and click on OK.

The ADO.NET Entity Data Source control is now added to your toolbox as shown in the following figure:



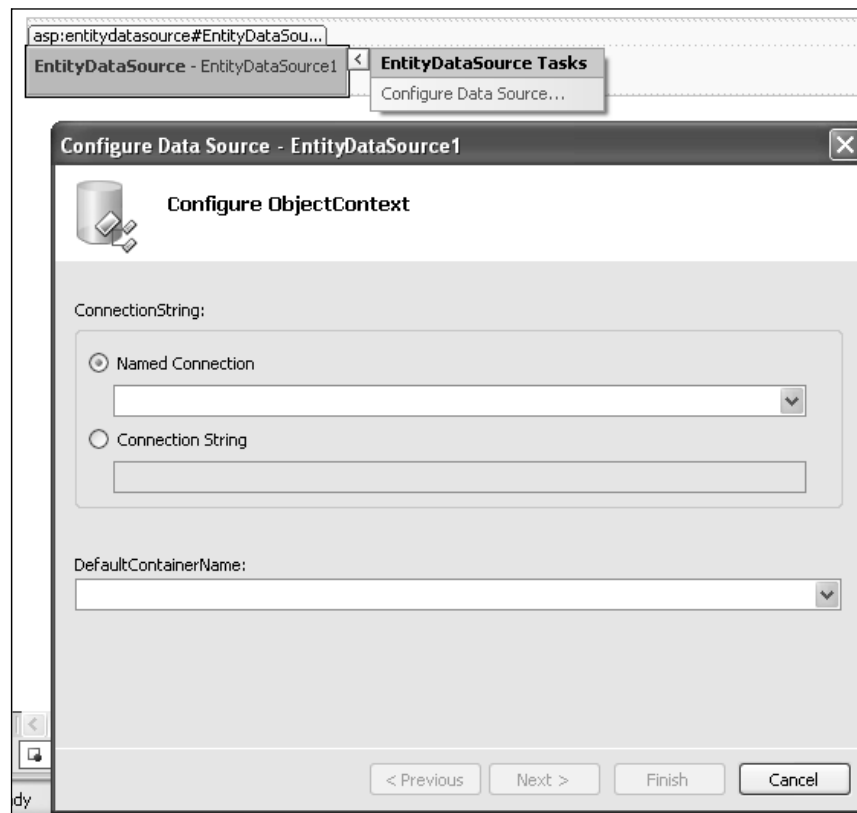
If the EntityDataSource component is not listed in the list of the components displayed in the Choose Toolbox Items window, you will have to add it manually. To do this, click on the **Browse** button in the Choose Toolbox Items window, locate the System.Web.Entity.dll in the folder in your system where Microsoft .NET Framework 3.5 has been installed and click on **OK**.

Implementing Our First Application Using the Entity Framework

In this section, we will learn how to use the Entity Data Model and the Entity Data Source Control to implement our first program using the Entity Framework. We will use a GridView control to display bound data.

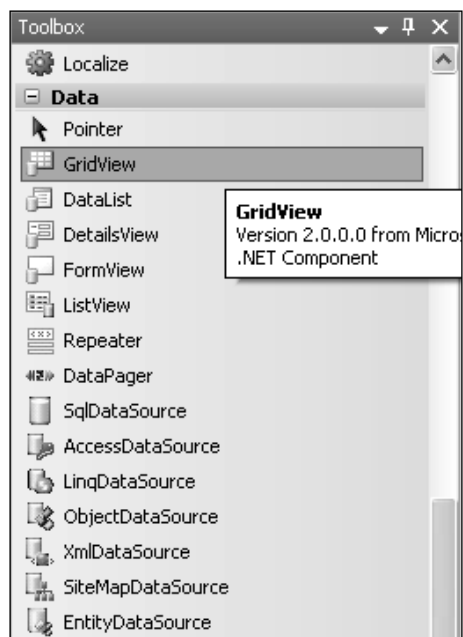
Refer to the solution we created earlier using the Entity Data Model Designer. Now, follow these steps:

1. Drag and drop an Entity Data Source control from the toolbox onto your `Default.aspx` web form.
2. Now, click on the **Configure Data Source** option to specify the data source. Refer to the following figure:

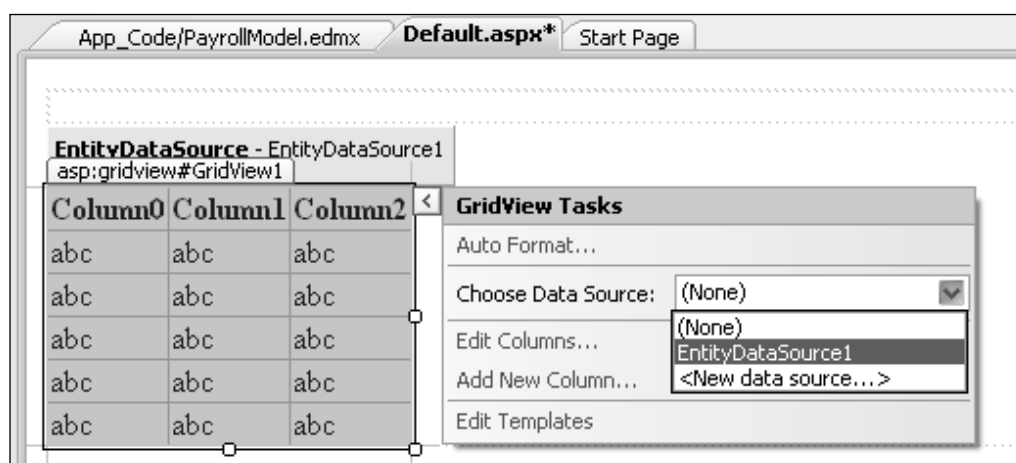


3. Specify the Connection String and DefaultContainerName and then click on **Next**.

4. Specify the fields you would want to retrieve from the database table and click on **Finish** when done.
5. Now, drag and drop a **GridView** control from the toolbox onto the `Default.aspx` web form as seen in the following figure:



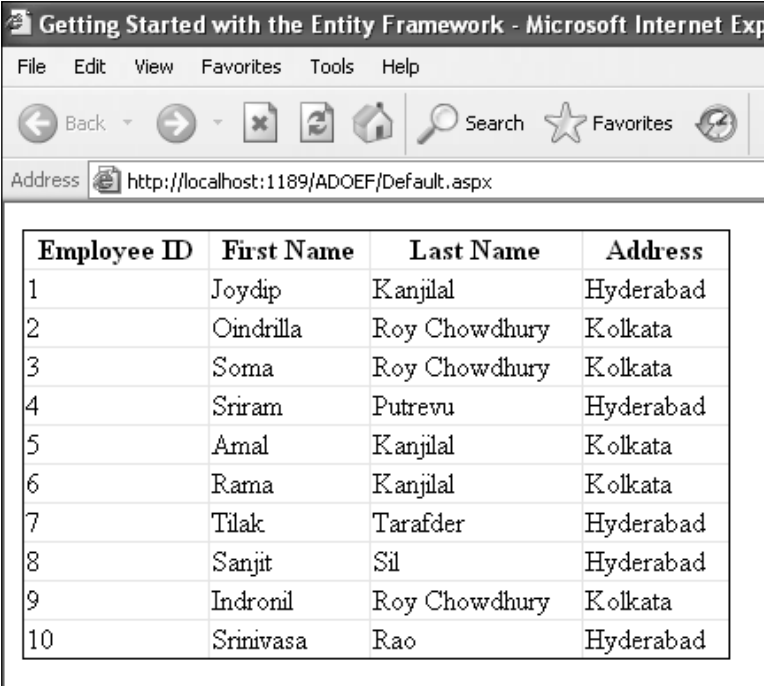
6. Next, use the **Choose Data Source** option of the GridView control to associate its data source with the Entity Data Source control we created earlier. Refer to the following figure:



Here is how the markup code of the GridView control looks with its templates defined. Note how the DataSourceID of the GridView control has been associated with the Entity Data Source control we created earlier.

```
<asp:GridView ID="GridView1" runat="server" AutoGenerateColumns=
"False" DataKeyNames="EmployeeID"
    DataSourceID="SqlDataSource1" BorderColor="Black"
    BorderStyle="Solid" Width="400px">
    <Columns>
        <asp:BoundField DataField="EmployeeID"
HeaderText="Employee ID" ReadOnly="True" SortExpression="EmployeeID"
/>
        <asp:BoundField DataField="FirstName" HeaderText=
"First Name" SortExpression="FirstName" />
        <asp:BoundField DataField="LastName" HeaderText=
"Last Name" SortExpression="LastName" />
        <asp:BoundField DataField="Address" HeaderText="Address"
SortExpression="Address" />
    </Columns>
</asp:GridView>
```

We are done! When you execute the application, your output should be similar to what is shown in the following figure:



The screenshot shows a web browser window titled "Getting Started with the Entity Framework - Microsoft Internet Exp". The address bar shows "http://localhost:1189/ADOEF/Default.aspx". The main content area displays a table with 4 columns: Employee ID, First Name, Last Name, and Address. The table contains 10 rows of data.

Employee ID	First Name	Last Name	Address
1	Joydip	Kanjilal	Hyderabad
2	Oindrilla	Roy Chowdhury	Kolkata
3	Soma	Roy Chowdhury	Kolkata
4	Sriram	Putrevu	Hyderabad
5	Amal	Kanjilal	Kolkata
6	Rama	Kanjilal	Kolkata
7	Tilak	Tarafder	Hyderabad
8	Sanjit	Sil	Hyderabad
9	Indronil	Roy Chowdhury	Kolkata
10	Srinivasa	Rao	Hyderabad

Summary

In this chapter, we have discussed how we can get started with the ADO.NET Entity Framework. We have learned how to create an Entity Data Model and use it along with the Entity Data Source control to bind data to a GridView data control. In the next chapter, we will continue to explore the Entity Data Model including each of its sections, and how they are related to each other.

Where to buy this book

You can buy Entity Framework Tutorial from the Packt Publishing website:
<http://www.packtpub.com/entity-framework-tutorial/book>

Free shipping to the US, UK, Europe and selected Asian countries. For more information, please read our [shipping policy](#).

Alternatively, you can buy the book from Amazon, BN.com, Computer Manuals and most internet book retailers.



www.PacktPub.com

For More Information: www.packtpub.com/entity-framework-tutorial/book