When you're finished, an **SqlDataSource Control** is placed on your Web form.

In Tutorial 11-5, you display the *Karate Members* table in a GridView.

Tutorial 11-5:
Displaying the *Karate Members* table in a GridView

In this tutorial, you will create a connection to the *Karate* database, and display the *Members* table in a GridView control. You will perform some basic configurations of the grid's appearance.

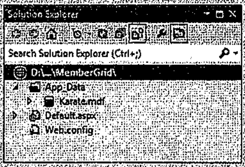
Step 1: Create a new emtpy Web site named *MemberGrid*. Add a new Web form named *Default.aspx* to the project.

Step 2: Right-click the project name in the *Solution Explorer* window, then click *New Folder* on the pop-up menu. Name the folder *App_Data*.

Step 3: Open a *Windows Explorer* window and copy the *Karate.mdf* file from the student sample programs folder named *Chap11* to your project's *App_Data* folder.

Step 4: In the *Solution Explorer* window, right-click the project name and select *Refresh Folder*. Then expand the entry under *App_Data* and look for the *Karate.mdf* filename, as shown in Figure 11-49.

Figure 11-49 Locating the *Karate.mdf* file under *App_Data* in the *Solution Explorer* window



Step 5: In the *Design* view window of the Default.aspx form, select *DOCUMENT* in the *Properties* window and set its Title property to *Karate Members*.

Step 6: On the first line of the Web page (in *Design* view) type *Members Table, Karate Database*. Set the block style to *Heading 2*, and press Enter at the end of the line.

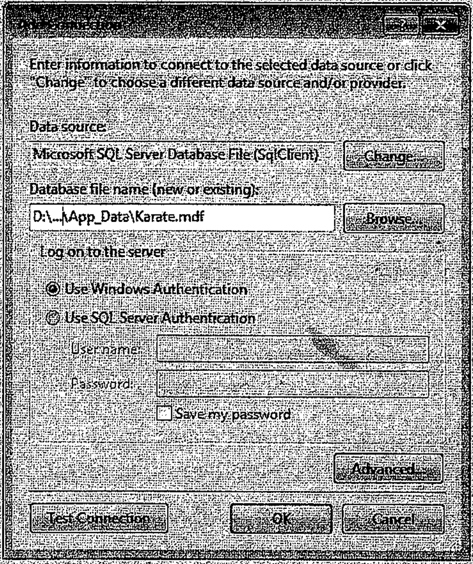
Step 7: Place a GridView control on the form. You can find it in the *Data* section of the *Toolbox* window. Drag its right handle until its Width property equals about 640 pixels. Its Height property should be blank.

Step 8: Click the grid's smart tag, opening the *GridView Tasks* dialog box. Select *<New data source . . .>* from the *Choose Data Source* DropDown list.

Step 9: In the *Data Source Configuration Wizard*, select *Database*, change the ID value to *KarateDataSource*, and click the *OK* button.

Step 10: The next step in the wizard is named *Choose Your Data Connection*. Click the *New Connection* button. When the *Add Connection* window appears, as shown in Figure 11-50, make sure the *Data Source* field is set to *Microsoft SQL Server Database File (SqlClient)*. If some other value appears in the field, change it.

Figure 11-50 Select the *Karate.mdf* database file



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 Database File (SqlClient) [Change...]

Database file name (new or existing):

D:\...\App_Data\Karate.mdf [Browse...]

Log on to the server

● Use Windows Authentication

○ Use SQL Server Authentication

User name:

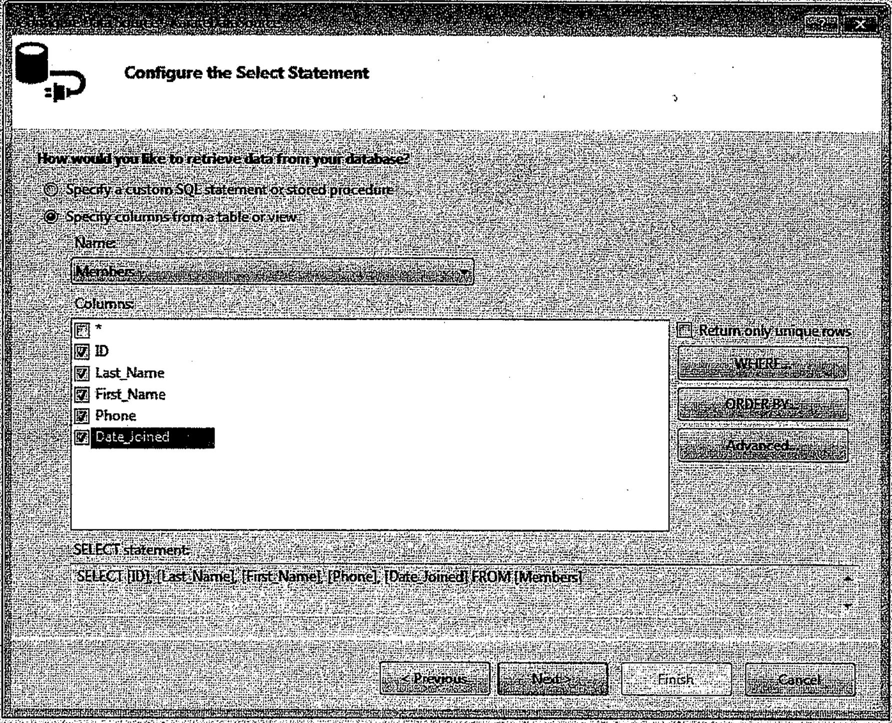
Password:

□ Save my password

Step 11: For the Database file name entry, click the *Browse* button, select your project's *App_Data* folder, select *Karate.mdf*, and click the *Open* button. Then, click the *OK* button to close the *Add Connection* dialog. When you return to the window that reads *Choose Your Data Connection*, click the *Next* button. When a window appears that reads *Save the Application Connection String to the Application Configuration File*, click the *Next* button.

Step 12: You will be asked to configure the SELECT statement that pulls rows and columns from the database. From the *Name* DropDown list, select the *Members* table, as shown in Figure 11-51.

Figure 11-51 Configuring the SELECT statement



Configure the Select Statement

How would you like to retrieve data from your database?

○ Specify a custom SQL statement or stored procedure

● Specify columns from a table or view

Name:

Members

Columns:

□ *
☑ ID
☑ Last_Name
☑ First_Name
☑ Phone
☑ Date_Joined

□ Return only unique rows

SELECT statement:

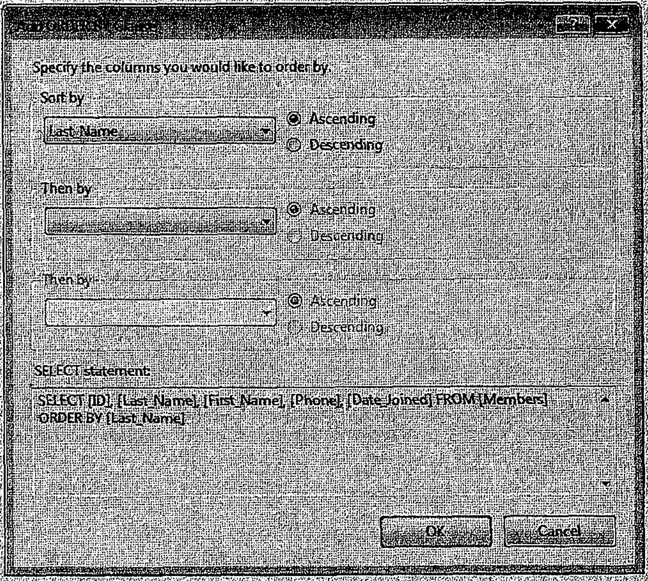
```
SELECT [ID], [Last_Name], [First_Name], [Phone], [Date_Joined] FROM [Members]
```

Step 13: Place check marks next to the following columns, in order: *ID*, *Last_Name*, *First_Name*, *Phone*, and *Date_Joined*.

Step 14: Click the *ORDER BY . . .* button. In the dialog box shown in Figure 11-52, sort by the *Last_Name* column. Click *OK* to close the dialog box.

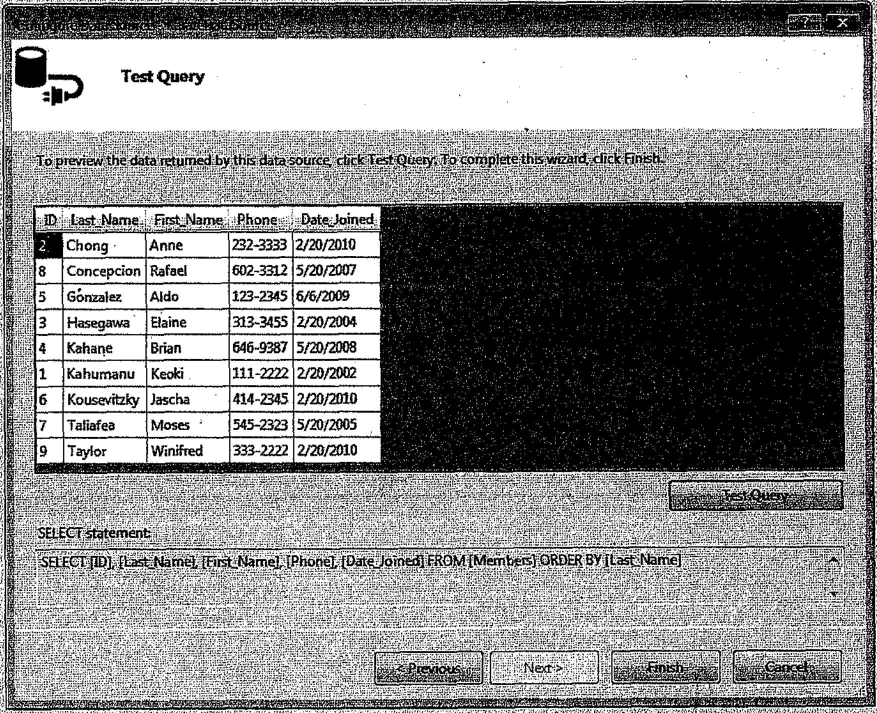
Step 15: Returning to the *Configure the Select Statement* dialog box, click the *Next* button, which takes you to the *Test Query* dialog box. Click the *Test Query* button. If the displayed columns match those shown in Figure 11-53, click the *Finish* button to close the window.

Figure 11-52 Adding the `ORDER BY` clause



```
SELECT [ID], [Last_Name], [First_Name], [Phone], [Date_Joined] FROM [Members]
ORDER BY [Last_Name]
```

Figure 11-53 Testing the `SELECT` query



To preview the data returned by this data source, click Test Query. To complete this wizard, click Finish.

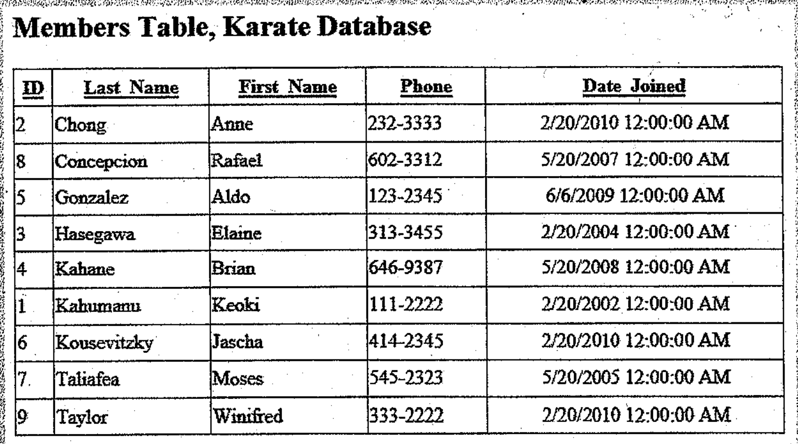
ID	Last_Name	First_Name	Phone	Date_Joined
2	Chong	Anne	232-3333	2/20/2010
8	Concepcion	Rafael	602-3312	5/20/2007
5	Gonzalez	Aldo	123-2345	6/6/2009
3	Hasegawa	Elaine	313-3455	2/20/2004
4	Kahane	Brian	646-9387	5/20/2008
1	Kahumanu	Keoki	111-2222	2/20/2002
6	Kousevitzky	Jascha	414-2345	2/20/2010
7	Taliafea	Moses	545-2323	5/20/2005
9	Taylor	Winifred	333-2222	2/20/2010

```
SELECT [ID], [Last_Name], [First_Name], [Phone], [Date_Joined] FROM [Members] ORDER BY [Last_Name]
```

Step 16: Click the *GridView*'s smart tag again and check the *Enable Sorting* check box.

Step 17: Save and run the Web application. The contents of your browser window should appear as shown in Figure 11-54, although some of the data in the rows may be different.

Figure 11-54 Running the Web application



Members Table, Karate Database

ID	Last Name	First Name	Phone	Date Joined
2	Chong	Anne	232-3333	2/20/2010 12:00:00 AM
8	Concepcion	Rafael	602-3312	5/20/2007 12:00:00 AM
5	Gonzalez	Aldo	123-2345	6/6/2009 12:00:00 AM
3	Hasegawa	Elaine	313-3455	2/20/2004 12:00:00 AM
4	Kahane	Brian	646-9387	5/20/2008 12:00:00 AM
1	Kahumanu	Keoki	111-2222	2/20/2002 12:00:00 AM
6	Kousevitzky	Jascha	414-2345	2/20/2010 12:00:00 AM
7	Taliafea	Moses	545-2323	5/20/2005 12:00:00 AM
9	Taylor	Winifred	333-2222	2/20/2010 12:00:00 AM

Step 18: Experiment with sorting columns by clicking each of the column headers. If you click the same column twice in a row, it reverses the sort order. Close the browser to end the application.

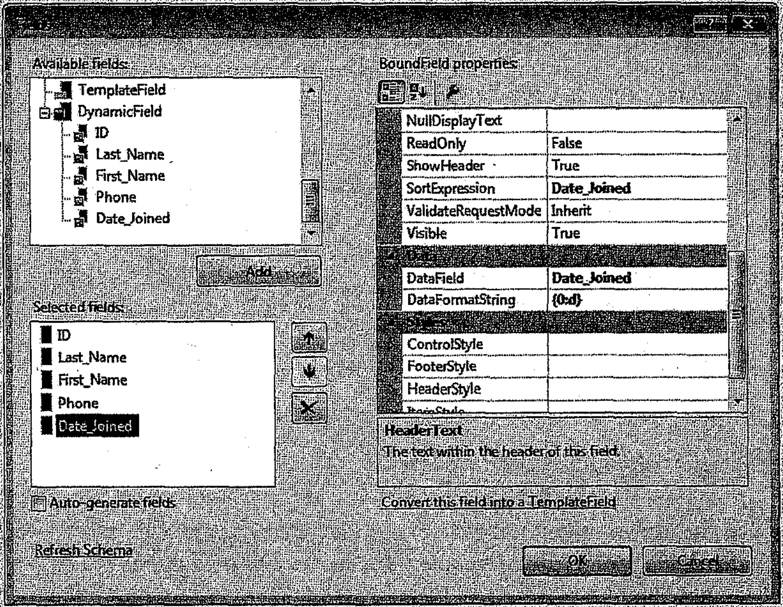
Step 19: Next, you will format the *Date_Joined* column. Select the grid with the mouse. In the *Properties* window, click the Columns property, which causes the *Fields* dialog box to display. In the lower left box, select *Date_Joined*. In the properties list for this column, enter `{0:d}` into the DataFormatString property, as shown in Figure 11-55. The `{0:d}` is called a *format specifier*. In this case, it says to use a *short date* format. Format specifiers are described in MSDN Help under the topic *Formatting overview*.

Step 20: Next, you will set a property that centers the values in the *Date_Joined* column. Expand the entries under the column's ItemStyle property (last in the list of *BoundField* properties in the *Fields* dialog window). Change the Horizontal-Align subproperty to *Center*. Click the *OK* button to close the dialog box.

Step 21: Save and run the application. Your output should be similar to that shown in Figure 11-56.

Step 22: Close the browser window to end the program.

Figure 11-55 Formatting the *Date_Joined* column



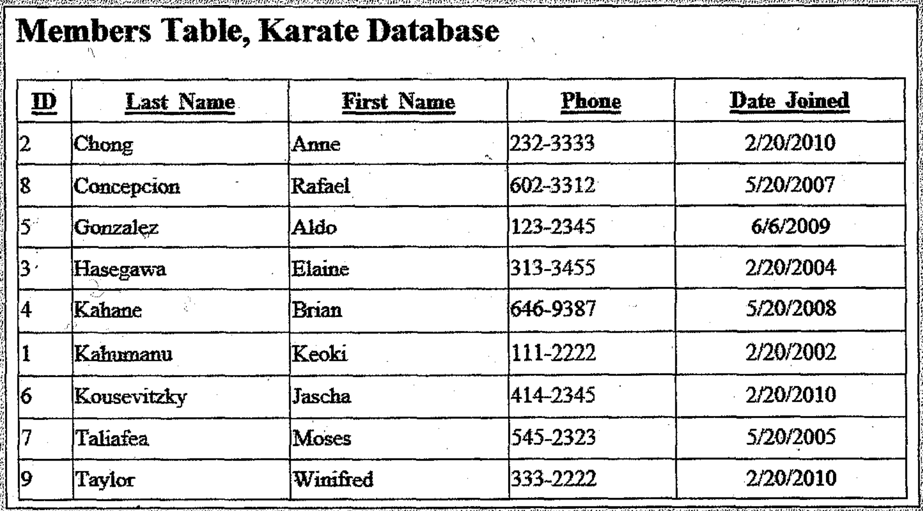
Selected fields:

- ID
- Last_Name
- First_Name
- Phone
- Date_Joined

NullDisplayText	
ReadOnly	False
ShowHeader	True
SortExpression	**Date_Joined**
ValidateRequestMode	Inherit
Visible	True
Data	
DataField	**Date_Joined**
DataFormatString	**{0:d}**
Styles	
ControlStyle	
FooterStyle	
HeaderStyle	

HeaderText
The text within the header of this field.

Figure 11-56 After formatting the *Date_Joined* column



Members Table, Karate Database

ID	Last Name	First Name	Phone	Date Joined
2	Chong	Anne	232-3333	2/20/2010
8	Concepcion	Rafael	602-3312	5/20/2007
5	Gonzalez	Aldo	123-2345	6/6/2009
3	Hasegawa	Elaine	313-3455	2/20/2004
4	Kahane	Brian	646-9387	5/20/2008
1	Kahumanu	Keoki	111-2222	2/20/2002
6	Kousevitzky	Jascha	414-2345	2/20/2010
7	Taliafea	Moses	545-2323	5/20/2005
9	Taylor	Winifred	333-2222	2/20/2010

In addition to the formats you modified just now, there are many detailed formatting changes you can make to a GridView control.

Using a DetailsView Control to Modify Table Rows

The **DetailsView control** makes it easy to view, edit, delete, or add rows to a database table. To use it, you must create a data source, as you did in Tutorial 11–5. When you connect the DetailsView to the data source, most of the work is done for you. Microsoft engineers have been working hard to automate as many menial tasks as they can, and database table editing is high on the list of tasks most programmers would prefer *not* to do repeatedly.

The DetailsView control is found in the Data section of the *Toolbox* window. When you place it on a Web form, use its smart tag (upper right-hand corner) to add a database connection and set various options. You did the same for the GridView control in Tutorial 11–5.

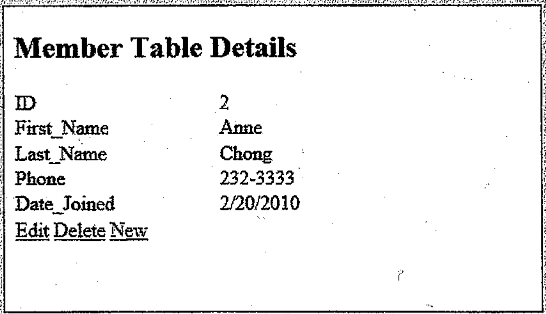
In Tutorial 11-6, you will update the *Karate Members* table using a DetailsView control.

Tutorial 11-6:
Updating the *Karate Members* table

In this tutorial, you will write an application that lets the user view, edit, insert, and delete individual rows in the *Members* table in the *Karate* database. You will create an SqlDataSource control and hook it up to a DetailsView control. You will not have to write any program code.

Figure 11-57 shows the finished program right after it starts, with rows sorted by last name. The underlined words *Edit*, *Delete*, and *New* are called *link buttons* (LinkButton controls). They look like HTML links, but function like ordinary button controls.

Figure 11-57 Adding a member at runtime



Member Table Details

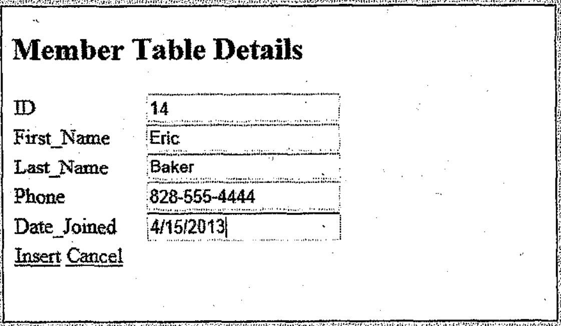
```
ID             2
First_Name     Anne
Last_Name      Chong
Phone          232-3333
Date_Joined    2/20/2010
```

In Figure 11-58, the user has clicked the *New* button and begun to enter data for a new member. The user will soon click the *Insert (link)* button, which will save the new row in the database.

Figure 11-59 shows the same form after the user has clicked the *Insert* button. The new member (Eric Baker) appears in the detail fields.

If the user tries to add a row having an ID number equal to an existing ID in the table, an error page displays, as shown in Figure 11-60. The user can click the browser's *Back* button, enter a different ID, and try again.

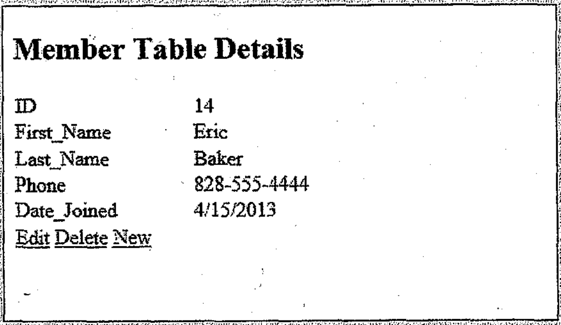
Figure 11-58 About to insert a new member



Member Table Details

ID 14
First_Name Eric
Last_Name Baker
Phone 828-555-4444
Date_Joined 4/15/2013

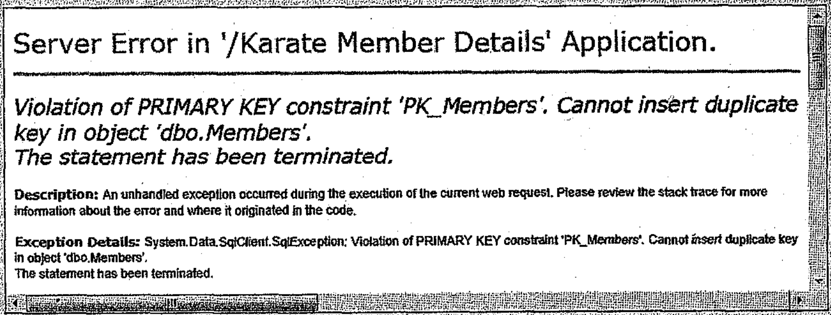
Figure 11-59 After clicking the *Insert* button



Member Table Details

ID 14
First_Name Eric
Last_Name Baker
Phone 828-555-4444
Date_Joined 4/15/2013

Figure 11-60 Error displayed when the user tries to add a row with a duplicate ID



Server Error in '/Karate Member Details' Application.

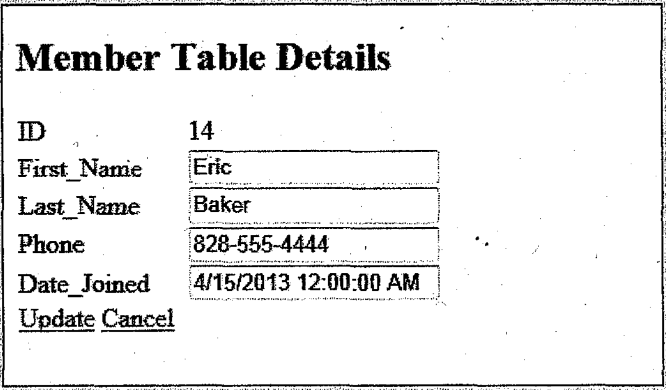
Violation of PRIMARY KEY constraint 'PK_Members'. Cannot insert duplicate key in object 'dbo.Members'.
The statement has been terminated.

Description: An unhandled exception occurred during the execution of the current web request. Please review the stack trace for more information about the error and where it originated in the code.

Exception Details: System.Data.SqlClient.SqlException: Violation of PRIMARY KEY constraint 'PK_Members'. Cannot insert duplicate key in object 'dbo.Members'.
The statement has been terminated.

When the user clicks the *Edit* button, he or she can modify any of the member fields, as shown in Figure 11-61. When the user clicks the *Update* button, changes to the record are saved in the database.

Figure 11-61 After clicking the *Edit* button



Member Table Details

ID 14
First_Name Eric
Last_Name Baker
Phone 828-555-4444
Date_Joined 4/15/2013 12:00:00 AM
Update Cancel

Now let's build the program.

Step 1: Create a new empty Web site named *Karate Member Details*. Add the following items to the project:

- a Web form named *Default.aspx*
- a folder named *App_Data*

Step 2: Copy the *Karate.mdf* database file into your project's *App_Data* folder.

Step 3: Right-click the project name in the *Solution Explorer* window and select *Refresh Folder*. Verify that *Karate.mdf* appears under the *App_Data* entry.

Step 4: Switch to the Design view of Default.aspx, select *DOCUMENT* in the *Properties* window, and set its Title property to *Members Table Details*.

Step 5: On the first line of the page, insert *Members Table Details*, and give it a *Heading 2* block style. Then, press Enter to go to the next line.

Step 6: Add a DetailsView control to the page, and set its ID property to `dvwAddMember`. Widen it to about 300 pixels. Make sure its Height property is blank.

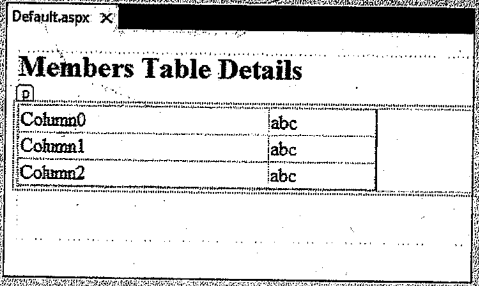
Step 7: Save the project. Figure 11-62 shows your work so far.

Next, you will add a data source to the project.

Step 8: Select the smart tag in the upper right corner of *dvwAddMember*. From the *Choose Data Source* DropDown list, select *<New data source . . .>*. Select *Database*, and name the data source *MembersDataSource*, as shown in Figure 11-63. Click the *OK* button to continue.

Step 9: As in the previous tutorial, create a connection to the *Karate.mdf* file in the project's *App_Data* folder.

Figure 11-62 Designing the *Karate Member Details* form

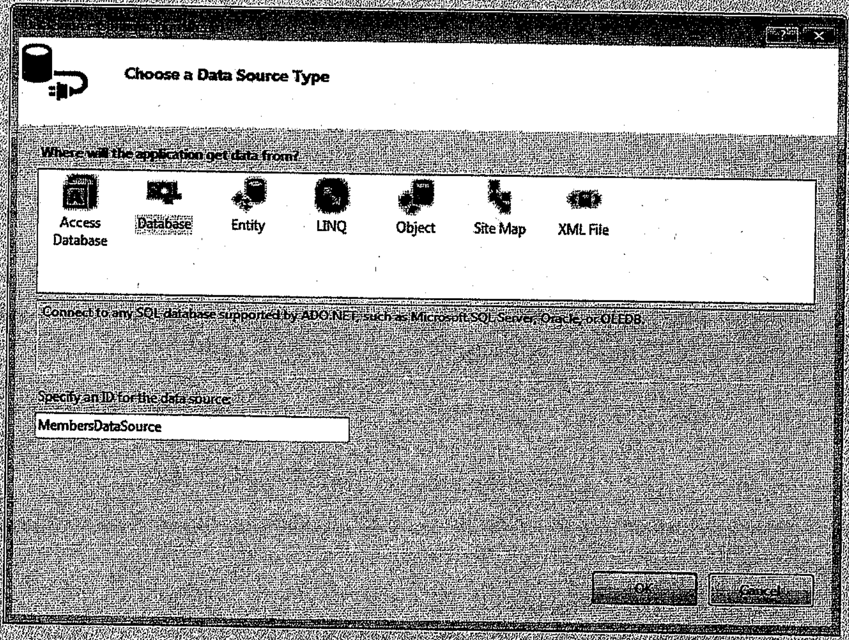


Default.aspx × [redacted]

Members Table Details

Column0	abc
Column1	abc
Column2	abc

Figure 11-63 Creating the data source



Choose a Data Source Type

Where will the application get data from?

Connect to any SQL database supported by ADO.NET, such as Microsoft SQL Server, Oracle, or OLEDB.

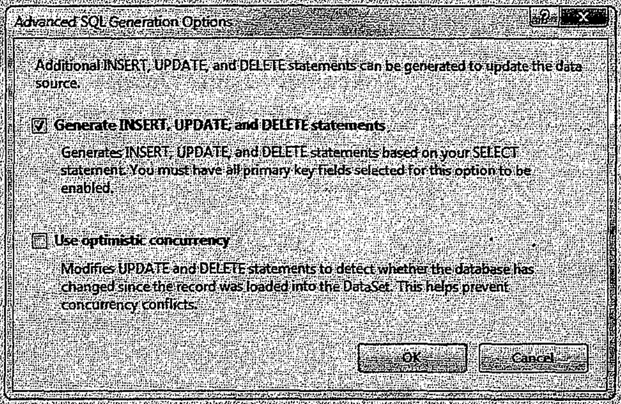
Specify an ID for the data source:

MembersDataSource

Step 10: When the window entitled *Save the Connection String to the Application Configuration File* appears, click the *Next* button.

Step 11: In the next window, select all columns in the *Members* table, and order the rows by *Last_Name* in ascending order. When you return to the window entitled *Configure the Select Statement*, click he *Advanced* b

Figure 11-64 Selecting *Advanced SQL Generation Options*



Additional INSERT, UPDATE, and DELETE statements can be generated to update the data source.

☑ **Generate INSERT, UPDATE, and DELETE statements**

Generates INSERT, UPDATE, and DELETE statements based on your SELECT statement. You must have all primary key fields selected for this option to be enabled.

☐ **Use optimistic concurrency**

Modifies UPDATE and DELETE statements to detect whether the database has changed since the record was loaded into the DataSet. This helps prevent concurrency conflicts.

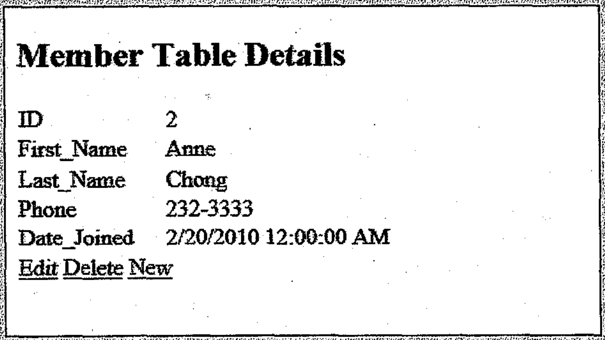
Step 14: Save and run the application. Sample output is shown in Figure 11-65.

Let's pause and reflect on what you have accomplished so far in this tutorial. You have created a connection to the Members database table, and you have created a useful Web form that lets the user do all of the following:

1. Display the Members table
2. Add new rows to the table
3. Modify (Edit) existing rows
4. Delete rows from the table

Behind this useful control, as you can imagine, are the same types of S queries that you used in Chapter 10. In fact, the SQL queries are embedded rectly into the HTML of your web page. When this tutorial is over, we will a closer look at the way queries are stored. But, now, let's test the DetailsVi control while the browser window is still open.

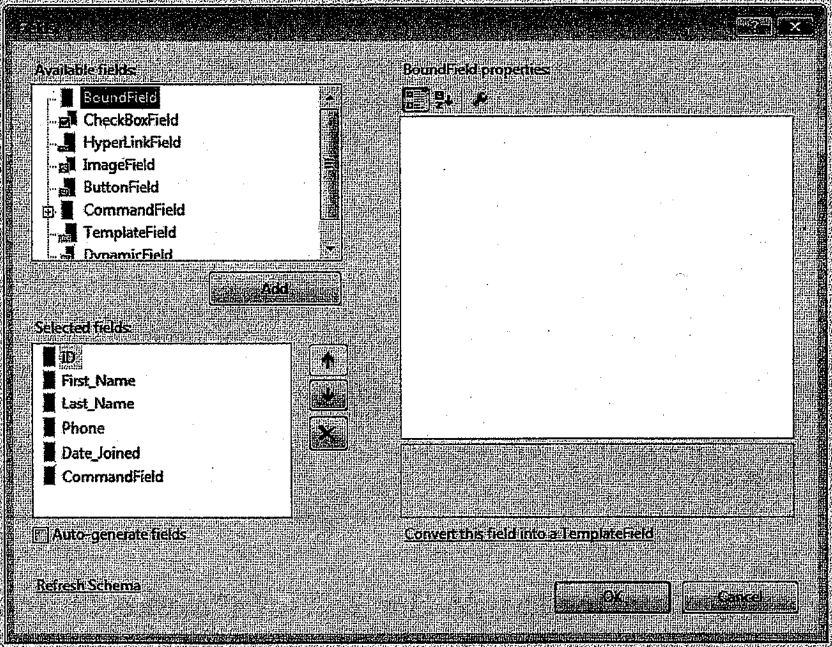
Figure 11-65 Running the *Karate Member Details* application for the first time



Member Table Details

ID	2
First_Name	Anne
Last_Name	Chong
Phone	232-3333
Date_Joined	2/20/2010 12:00:00 AM

Figure 11-66 *Fields* dialog box for the DetailsView control



to *0px* (zero pixels wide). Next, select its Fields property, causing the *Fields* dialog box to display (see Figure 11-66).

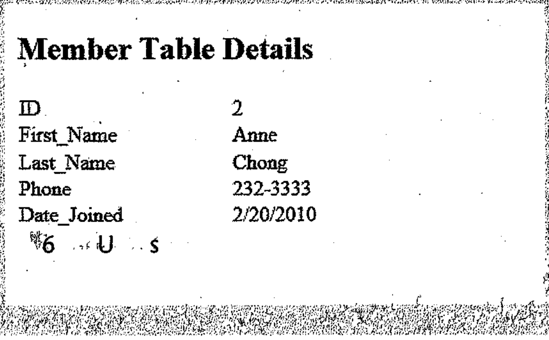
Step 17: In the *Fields* dialog box, select the *Date_Joined* field in the lower left list box. Then in the right-hand list box, set its DataFormatString property to *{0:d}*. You may recall this is the same short date format specifier we used in the GridView control.

Step 18: Next, you will change the field order slightly. Select the *First_Name* field in the lower left list box, and use the arrow on the right side of the box to move the *First_Name* field above the *Last_Name* field.

Step 19: Click *OK* to close the *Fields* dialog box. Next, find the GridLines property and set it equal to *None*.

Step 20: Save and run the application. It should now appear as shown in Figure 11-67.

Figure 11-67 After modifying the field display in the DetailsView control



Member Table Details

ID	2
First_Name	Anne
Last_Name	Chong
Phone	232-3333
Date_Joined	2/20/2010

Step 21: Click the *New* button, and notice that all the text boxes become empty. Enter the following data: *14, Eric, Baker, 654–3210, 3/1/2013*. Then click the *Insert* button. The display should now show the record you inserted.

Step 22: Again, try to insert a new record, using the same ID number (14). You should see a detailed error message that refers to a Violation of a primary key constraint. Because the ID field values must be unique, you cannot add two members to the table who have the same ID number. Click the browser's Back button, change the ID to *15*, and click the *Insert* button. This time, the insert operation should work.

Step 23: Click the *Delete* button. The last member you inserted should disappear. Then the previous member you inserted (Eric Baker) should display. Click the Delete button again to delete this member.

Step 24: Close the Web browser window.

You're done. You created a fully functional update program without writing a single line of code.

SQL Queries inside the SqlDataSource Control

In Tutorial 11–6, you created a simple web application that used the SqlDataSource control to populate a DetailsView control. The user was able to display, modify, insert, and delete rows from the Karate Members table. You might be interested to see how a SqlDataSource is represented in your web form's HTML code. You can click the *Source* tab for the Default.aspx page in the *Karate Member Details* application and then look for the SqlDataSource control named *MembersDataSource*. This is the beginning of the code that defines it:

```
<asp:SqlDataSource ID="MembersDataSource" runat="server"
```

The *asp:SqlDataSource* tag identifies the type of control and assigns it an ID (*MembersDataSource*). The *runat* property indicates that this control executes on the Web server. Next, you will see the name of the connection string that ties this control to the database:

```
ConnectionString="<%$ ConnectionStrings:karateConnectionString %>"
```

Next, you will find the *DeleteCommand* property, which contains an SQL query that deletes rows from the Members table. It uses a query parameter (named *@ID*) to identify exactly which member is to be deleted:

```
DeleteCommand="DELETE FROM [Members] WHERE [ID] = @ID"
```

Next is the *InsertCommand* property, which contains the query used to insert new rows into the Members table. It has a parameter for each column in the table:

```
InsertCommand="INSERT INTO [Members] ([ID], [Last_Name],
[First_Name], [Phone], [Date_Joined]) VALUES (@ID, @Last_Name,
@First_Name, @Phone, @Date_Joined)"
```

Although we do not show them here, the control also contains *SelectCommand* and *UpdateCommand* properties, with their respective queries. With practice, you can edit the properties of ASP.NET controls directly in Source mode. Expert web programmers do just that, because they feel that HTML editing gives them more precise control over a Web form than they could get by using Visual Studio's *Design* view.