Accessing Databases

Continued
The Easy Way with ASP.NET

- New web controls make working with data much easier than programming ADO.NET directly.
- The SqlDataSource control is central.
- Drag one to the web form and configure the data source using the associated dialogs.
Save the Connection String to the Application Configuration File

Storing connection strings in the application configuration file simplifies maintenance and deployment. To save the connection string in application configuration file, enter a name in the text box and then click Next. If you choose not to do this, the connection string is saved in the page as a property of the data source control.

Do you want to save the connection in the application configuration file?

Yes, save this connection as:

ConnectionString
Configure Data Source - SqIDataSource1

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:
main

Columns:

- *
- name

Select statement:

SELECT [name] FROM [main]
To preview the data returned by this data source, click Test Query. To complete this wizard, click Finish.

SELECT statement:

```
SELECT [name] FROM [main]
```
Binding Controls

• Many controls can be bound to a data source object.
• Some special controls are specifically designed to display data effectively.
• Add a list box.
Set the Properties

![Properties window](image)

DataTextField
The field in the data source which provides the item text.
The Output
Adding an Item
Open the InsertQuery for the Data Source

Click data1
Add an Insert with Parameter

![Query Builder dialog box](image-url)

**Query Builder**

```
INSERT INTO main
(name)
VALUES (@newdata)
```
Bind the Parameter

Use tb1

INSERT INTO main(name) VALUES (@newdata)
The Button Event Handler

using System;
using System.Data;
using System.Configuration;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Web.UI.WebControls.WebParts;
using System.Web.UI.HtmlControls;

public partial class _Default : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {
    }

    protected void Button1_Click(object sender, EventArgs e)
    {
        if (tb1.Text != "") SqlDataSource1.Insert();
    }
}
Add a Delete Button
Add Delete Query
DELETE command:

DELETE FROM main WHERE (name = @name)

Parameters:

<table>
<thead>
<tr>
<th>Name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td></td>
</tr>
</tbody>
</table>

Parameter source:

- None

Default Value:

Show advanced properties
DELETE command:
DELETE FROM main WHERE (name = @name)
The Button Handler

protected void Button1_Click(object sender, EventArgs e)
{
    if (tb1.Text != "") SqlDataSource1.Insert();
}

protected void Delete_Click(object sender, EventArgs e)
{
    if (ListBox1.SelectedItem != null)
        SqlDataSource1.Delete();
}
Using a DataView

• You can use a DataView object to get a table from the select query of an SQL data source control.
• You can’t use this class to modify the table.
• It is useful to obtain data that is not bound to a control.
• Here is an example:
DataView dv = (DataView)SqlDataSource1.Select(DataSourceSelectArguments.Empty);
    if (dv == null)
    {
        //handle error
    }

    //The select query has returned the desired row
    DataRow row = dv.Table.Rows[0];
    string s = (string)row["name"];  
    //obtain data from other columns

    namebox.Text = s;  //display the name
    //display or use data from other columns