

Lab04.sql - Use this database table

```
-- Create the table to interact with
CREATE TABLE Products_Lab4(
ProductID      int IDENTITY PRIMARY KEY,
title          varchar(100),
authors        varchar(100),
copyrightDate  varchar(6),
edition        varchar(6),
isbn           varchar(25),
coverart       varchar(100),
description    varchar(100),
price          varchar(100)
);
```

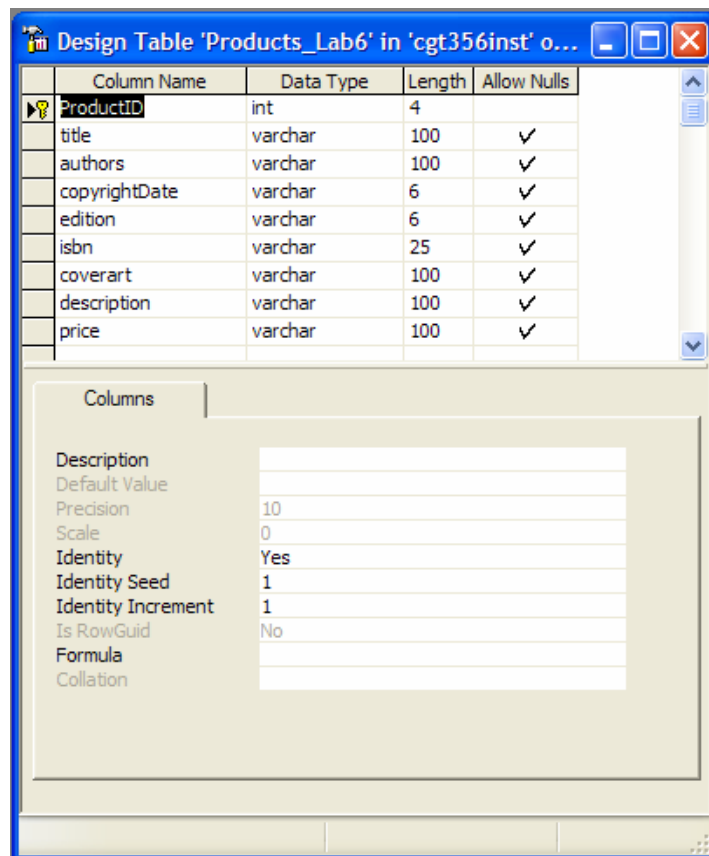


ProductID
title
authors
copyrightDate
edition
isbn
coverart
description
price

```
-- Insert some data to begin with
INSERT INTO Products_Lab4(title, authors, copyrightDate, edition, isbn, coverart, description,
price) VALUES('Visual Basic .NET How to Program: Second Edition', 'Harvey M. Deitel, Paul J. Deitel
& Tem R. Nieto', '2002', 2, '0-13-029363-6', 'vbnhttp2.png', 'Microsoft Visual Basic .NET',
76.00);
```

```
INSERT INTO Products_Lab4(title, authors, copyrightDate, edition, isbn, coverart, description,
price) VALUES('C++ How to Program: Fourth Edition', 'Harvey M. Deitel & Paul J. Deitel', '2002', 4,
'0-13-038474-7', 'cpphttp4.png', 'Introduces Web Programming with CGI', 76.00);
```

```
INSERT INTO Products_Lab4(title, authors, copyrightDate, edition, isbn, coverart, description,
price) VALUES('C# How to Program: First Edition', 'Harvey M. Deitel, Paul J. Deitel, Jeff
Listfield, Tem R. Nieto, Cheryl Yaeger & Marina Zlatkina', '2002', 1, '0-13-062221-4',
'csharphttp1.png', 'Introduces .NET and Web services', 76.00);
```



Column Name	Data Type	Length	Allow Nulls
ProductID	int	4	
title	varchar	100	✓
authors	varchar	100	✓
copyrightDate	varchar	6	✓
edition	varchar	6	✓
isbn	varchar	25	✓
coverart	varchar	100	✓
description	varchar	100	✓
price	varchar	100	✓

Columns	
Description	
Default Value	
Precision	10
Scale	0
Identity	Yes
Identity Seed	1
Identity Increment	1
Is RowGuid	No
Formula	
Collation	

Lab04.aspx Browser View

Select	Insert	Update	Delete						
ProductID	title	authors	copyrightDate	edition	isbn	coverart	description	price	
7	Microsoft Visual	Jon Jagger	2003	2003	0735619093	Microsoft	Teach yourself	26.39	
8	Microsoft Visual	Jon Jagger	2003	2003	0735619093	Microsoft	Teach yourself	26.39	
9	New Title Value	Jon Jagger	2003	2003	0735619093	Microsoft	Teach yourself	26.39	

Lab04.aspx

```
<%@ Page language="c#" Codebehind="Lab04.aspx.cs" AutoEventWireup="false" Inherits="Lab04.Lab04" %>
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN" >
<HTML>
  <HEAD>
    <title>Lab04</title>
    <meta name="CODE_LANGUAGE" Content="C#">
    <meta name="vs_defaultClientScript" content="JavaScript">
    <meta name="vs_targetSchema" content="http://schemas.microsoft.com/intellisense/ie5">
  </HEAD>
  <body MS_POSITIONING="GridLayout">
    <form id="Form1" method="post" runat="server">
      <asp:Button Runat="server" ID="select" Text="Select" OnClick="Select_Click" />
      <asp:Button Runat="server" ID="insert" Text="Insert" OnClick="Insert_Click" />
      <asp:Button Runat="server" ID="update" Text="Update" OnClick="Update_Click" />
      <asp:Button Runat="server" ID="delete" Text="Delete" OnClick="Delete_Click" />

      <!-- A DataGrid is a data bound list control that displays the items from a
      data source in a table. Use the DataGrid control to display the fields of a
      data source as columns in a table. Each row in the DataGrid control represents
      a record in the data source. The DataGrid control supports selection, editing,
      deleting, paging, and sorting. -->
      <asp:DataGrid Runat="server" ID="results" />
    </form>
  </body>
</HTML>
```

Lab04.aspx.cs

```
using System;
using System.Data;
using System.Data.SqlClient;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;

namespace Lab04
{
    /// <summary>
    /// //////////////////////////////////////
    ///     Author: Ronald J. Glotzbach
    ///
    ///     Date: February 16, 2005
    ///
    ///     Project: CGT 456 Lab04 Solution A
    /// //////////////////////////////////////
    /// Summary description for Lab04 class.
    /// Lab04 uses ExecuteReader and ExecuteNonQuery to execute SELECT,
    /// INSERT, UPDATE, and DELETE statements.
    /// </summary>
    public class Lab04 : System.Web.UI.Page
    {
        // Declare global variables
        // Pull controls over from aspx
        protected System.Web.UI.WebControls.DataGrid results;
        protected System.Web.UI.WebControls.Button select;
        protected System.Web.UI.WebControls.Button insert;
        protected System.Web.UI.WebControls.Button update;
        protected System.Web.UI.WebControls.Button delete;
        // Declare global objects to be used
        public String ConnStr = "server=sotdev4.tech.purdue.edu;uid=xxx;pwd=yyy;database=zzz";
        public SqlConnection oConn;
        public SqlCommand myCommand;
        public SqlDataReader myReader;
        public SqlDataAdapter sda;
        public DataSet ds;
        public DataTable dt;
        public String Sql;

        /// <summary>
        /// function Select_Click()
        /// When the Select All Button is clicked, all of the records will be
        /// selected and binded to the DataGrid.
        /// </summary>
        /// <param name="Source"></param>
        /// <param name="E"></param>
        protected void Select_Click(Object Source, EventArgs E)
        {
            // Create the SQL statement
            Sql= "SELECT * FROM Products_Lab4";
            // Create a new connection object using the connection string above.
            // oConn represents an open connection to a SQL Server database.
            oConn = new SqlConnection(ConnStr);
            // Create a new SQL command using the SQL command and connection object
            myCommand = new SqlCommand(Sql, oConn);

            // Open the connection to the database
            oConn.Open();

            // set the data source
            results.DataSource = myCommand.ExecuteReader();
        }
    }
}
```

```

    // bind the data source to the datagrid
    results.DataBind();

    // Close the connection to the database
    oConn.Close();
}

/// <summary>
/// function Insert_Click()
/// When the Insert Button is clicked, a new record will be inserted
/// into the database. The Primary Key is set to IDENTITY, so it will
/// auto-increment.
/// </summary>
/// <param name="Source"></param>
/// <param name="E"></param>
protected void Insert_Click(Object Source, EventArgs E)
{
    // Create the SQL statement
    Sql = "INSERT INTO Products_Lab4 (title, authors, copyrightDate, edition, isbn, ";
    Sql += "coverart, description, price) VALUES('Microsoft Visual', 'Jon Jagger', ";
    Sql += "'2003', '2003', '0735619093', 'Microsoft', 'Teach yourself', '26.39')";
    // Create a new connection object using the connection string above.
    // oConn represents an open connection to a SQL Server database.
    oConn = new SqlConnection(ConnStr);
    // Create a new SQL command using the SQL command and connection object.
    // myCommand represents a SQL statement or stored procedure to execute
    // against a SQL Server database.
    myCommand = new SqlCommand(Sql, oConn);

    // Open the connection to the database
    oConn.Open();
    // Execute the SQL statement against the connection. ExecuteNonQuery will
    // return the number of rows affected, but does not return a recordset.
    myCommand.ExecuteNonQuery();
    // Close the connection to the database
    oConn.Close();

    // Call the Select_Click() function to redisplay the data in the browser.
    Select_Click(Source, E);
}

/// <summary>
/// function Update_Click()
/// When the Update Button is clicked, the record with the maximum
/// ProductID is updated with a different Title.
/// </summary>
/// <param name="Source"></param>
/// <param name="E"></param>
protected void Update_Click(Object Source, EventArgs E)
{
    // Get the max id
    // Create the SQL statement
    Sql= "SELECT Max(ProductID) AS MaxID FROM Products_Lab4";
    // Create a new connection object using the connection string above.
    // oConn represents an open connection to a SQL Server database.
    oConn = new SqlConnection(ConnStr);
    // Create a new SQL command using the SQL command and connection object.
    // myCommand represents a SQL statement or stored procedure to execute
    // against a SQL Server database.
    myCommand = new SqlCommand(Sql, oConn);

    // Open the connection to the database
    oConn.Open();

    // Instantiate an instance of the SqlDataAdapter class
    // The SqlDataAdapter, serves as a bridge between a DataSet and SQL Server

```

```

// for retrieving and saving data. The SqlDataAdapter provides this bridge
// by mapping Fill, which changes the data in the DataSet to match the data
// in the data source, and Update, which changes the data in the data source
// to match the data in the DataSet, using the appropriate Transact-SQL
// statements against the data source.
sda = new SqlDataAdapter();
// Set the value of the SDA select command equal to the command object.
// SelectCommand gets or sets a SQL statement used to select records in the data source.
sda.SelectCommand = myCommand;

// Instantiate an instance of the DataSet class which represents
// an in-memory cache of data.
ds = new DataSet();
// When the SqlDataAdapter fills a DataSet, it will create the necessary
// tables and columns for the returned data if they do not already exist.
sda.Fill(ds);

// Instantiate an instance of the DataTable class which represents
// one table of in-memory data. Additional Info: To add tables to the
// collection, use Add method of the DataTableCollection. To remove
// tables, use the Remove method.
dt = new DataTable();
// Get the first array index of the collection of tables contained in the DataSet
dt = ds.Tables[0];

// If a record exists, Do the update
if(dt.Rows[0][0].ToString() != "")
{
    // Create the SQL statement
    Sql = "UPDATE Products_Lab4 SET Title='New Title Value' WHERE ProductID=";
    Sql += dt.Rows[0][0].ToString();
    // Create a new SQL command using the SQL command and connection object
    myCommand = new SqlCommand(Sql, oConn);

    // Execute the SQL statement against the connection. ExecuteNonQuery will
    // return the number of rows affected, but does not return a recordset.
    myCommand.ExecuteNonQuery();
}
// Close the connection to the database
oConn.Close();

// Call the Select_Click() function to redisplay the data in the browser.
Select_Click(Source, E);
}

/// <summary>
/// function Delete_Click()
/// When the Delete Button is clicked, the record with the maximum
/// ProductID is deleted from the database.
/// </summary>
/// <param name="Source"></param>
/// <param name="E"></param>
protected void Delete_Click(Object Source, EventArgs E)
{
    // Get the max id
    // Create the SQL statement
    Sql= "SELECT Max(ProductID) AS MaxID FROM Products_Lab4";
    // Create a new connection object using the connection string above.
    // oConn represents an open connection to a SQL Server database.
    oConn = new SqlConnection(ConnStr);
    // Create a new SQL command using the SQL command and connection object.
    // myCommand represents a SQL statement or stored procedure to execute
    // against a SQL Server database.
    myCommand = new SqlCommand(Sql, oConn);

    // Open the connection to the database

```

```

oConn.Open();

// Instantiate an instance of the SqlDataAdapter class
// The SqlDataAdapter, serves as a bridge between a DataSet and SQL Server
// for retrieving and saving data. The SqlDataAdapter provides this bridge
// by mapping Fill, which changes the data in the DataSet to match the data
// in the data source, and Update, which changes the data in the data source
// to match the data in the DataSet, using the appropriate Transact-SQL
// statements against the data source.
sda = new SqlDataAdapter();
// Set the value of the SDA select command equal to the command object.
// SelectCommand gets or sets a SQL statement used to select records in the data source.
sda.SelectCommand = myCommand;

// Instantiate an instance of the DataSet class which represents
// an in-memory cache of data.
ds = new DataSet();
// When the SqlDataAdapter fills a DataSet, it will create the necessary
// tables and columns for the returned data if they do not already exist.
sda.Fill(ds);

// Instantiate an instance of the DataTable class which represents
// one table of in-memory data. Additional Info: To add tables to the
// collection, use Add method of the DataTableCollection. To remove
// tables, use the Remove method.
dt = new DataTable();
// Get the first array index of the collection of tables contained in the DataSet
dt = ds.Tables[0];

// If a record exists, Do the delete
if(dt.Rows[0][0].ToString() != "")
{
    // Create the SQL statement
    Sql = "DELETE FROM Products_Lab4 WHERE ProductID=" + dt.Rows[0][0].ToString();
    // Create a new SQL command using the SQL command and connection object
    myCommand = new SqlCommand(Sql, oConn);

    // Execute the SQL statement against the connection. ExecuteNonQuery will
    // return the number of rows affected, but does not return a recordset.
    myCommand.ExecuteNonQuery();
}
// Close the connection to the database
oConn.Close();

// Call the Select_Click() function to redisplay the data in the browser.
Select_Click(Source, E);
}

private void Page_Load(object sender, System.EventArgs e)
{
    // Put user code to initialize the page here
    if(!Page.IsPostBack)
    {
        //do nothing
    }
}

#region Web Form Designer generated code
override protected void OnInit(EventArgs e)
{
    //
    // CODEGEN: This call is required by the ASP.NET Web Form Designer.
    //
    InitializeComponent();
    base.OnInit(e);
}

```

```
/// <summary>
/// Required method for Designer support - do not modify
/// the contents of this method with the code editor.
/// </summary>
private void InitializeComponent()
{
    this.Load += new System.EventHandler(this.Page_Load);
}
#endregion
}
```