Chengzhu Jiang

cjiang@atu.edu

Final Project Report

Multi-database Querying Sustem

Table of Contents

[1. Final Project Summary 2](#_Toc417490761)

[2. System Techniques 2](#_Toc417490762)

[2.1. Login Page 2](#_Toc417490763)

[2.2. SQL Practice Page 3](#_Toc417490764)

[2.3. Change Password Page 8](#_Toc417490765)

[2.4. User Management Page 8](#_Toc417490766)

[2.5. Database Management Page 9](#_Toc417490767)

[2.6. Back-end Database Creation 9](#_Toc417490768)

[3. Conclusion and Future Work 10](#_Toc417490769)

[4. User Manual 10](#_Toc417490770)

[5. Appendix A: Source Code 11](#_Toc417490771)

1. Final Project Summary

The web-based multi-database querying system is basically a system for students’ on-line SQL practice who enrolled in the Database courses. It provides an interactive way for SQL learning which allows students to practice what they have learned. The students will be able to choose different databases in multiple DBMS platforms to perform SQL commands. They will receive immediate results after submitting queries. The accounts management and database management functions are also implemented for site administrators.

The system has been deployed on the IIS server hdbs.atu.edu. It uses multi-tier architecture to implement its functionality. The ASP.NET Web Forms pattern with a combination of HTML, JavaScript, and server controls are used to present information to the browser. The C# code in behind is used to handle the logic for the page and the database interaction. It uses multiple database platforms as the back-end such as Access, MySQL, SQL Server, SQLite, PostgreSQL, Oracle, and IBM DB2 to meet the demands for students in different levels or groups.

1. System Techniques

This website contains 5 pages, login page, SQL practice page, change password page, user management page, and database management page.

* 1. Login Page

I added ASP.NET login control to utilize login authentication functionality. It will compare the user name and password with the records in database to manage the users’ login. If the login is authorized, it will update the login date time in the database and redirect the users to different pages according to the users’ role.

Figure 1 Login Page

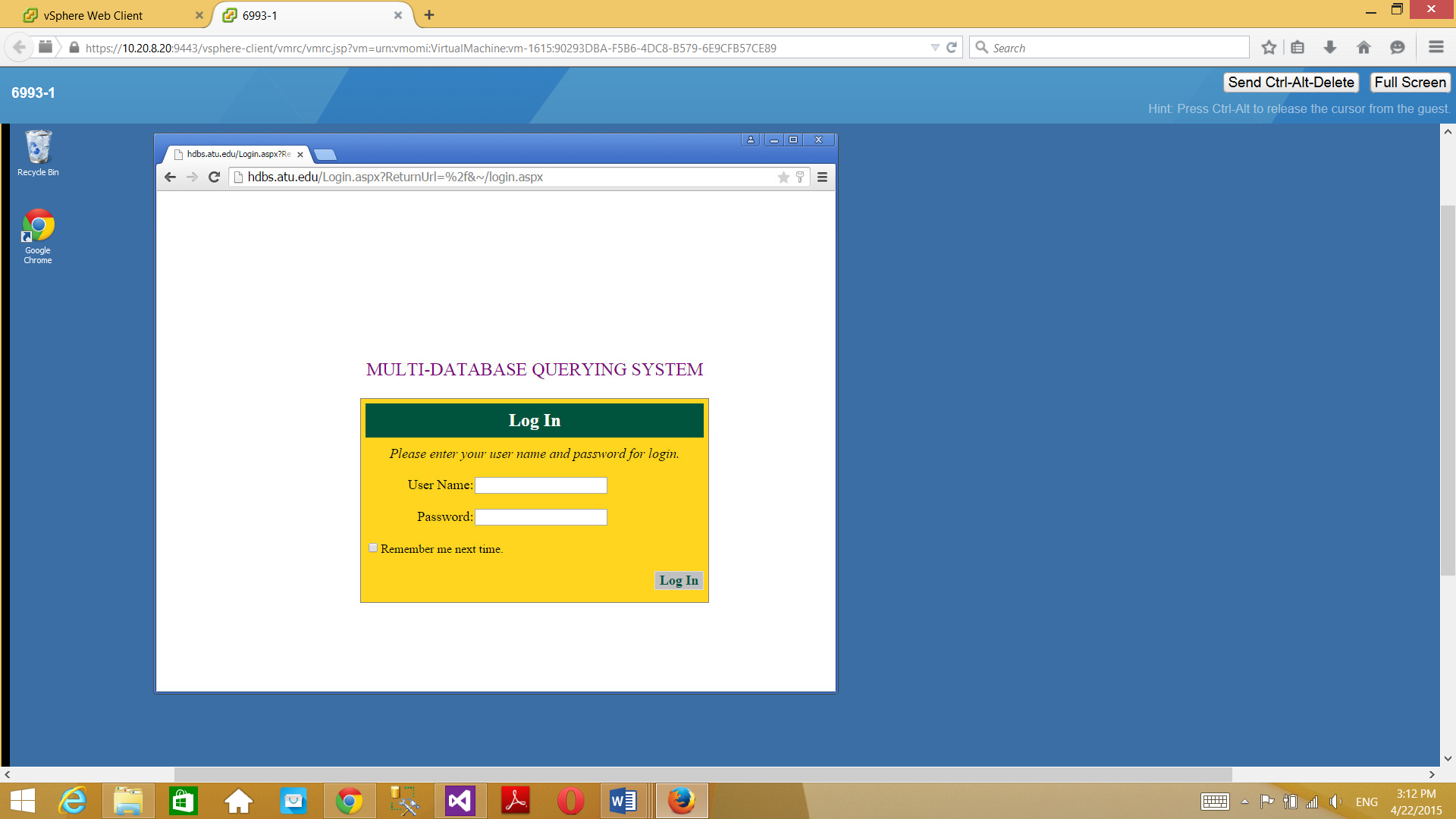


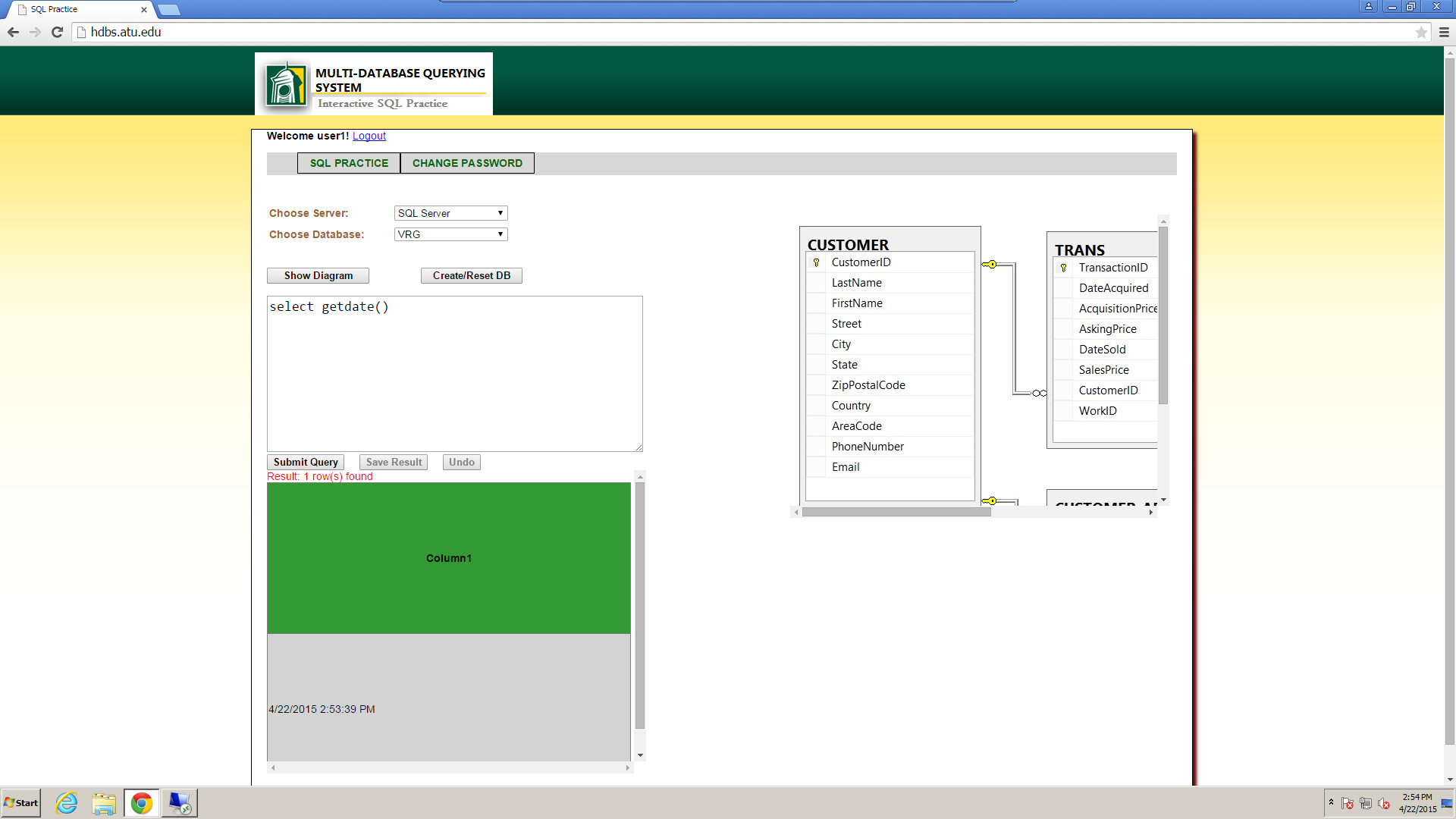
Figure 2 Management database diagram



* 1. SQL Practice Page

On the SQL Practice page, it offers multiple database servers and databases which associated with the selected server platform in the dropdown list. The user can view the database diagram and can create their own databases to practice on. It also contains a text editor which can accept the input as well as a Grid View to manipulate the query results.

Figure 3 SQL Practice Page



The flowchart describes the logic of the process in this page.

Login Page

Login

N

SQL Practice

Run Query

Show Diagram

Create Database

Database exists

Forward query to server

One statement

DB2 or Oracle

SQL Server

Non-select statement

Trans rollback

Trans commit

Error window

N

Y

N

Return result

Other DBMS

N

Privilege allowed

N

Input query

Show trans result

Save result

N

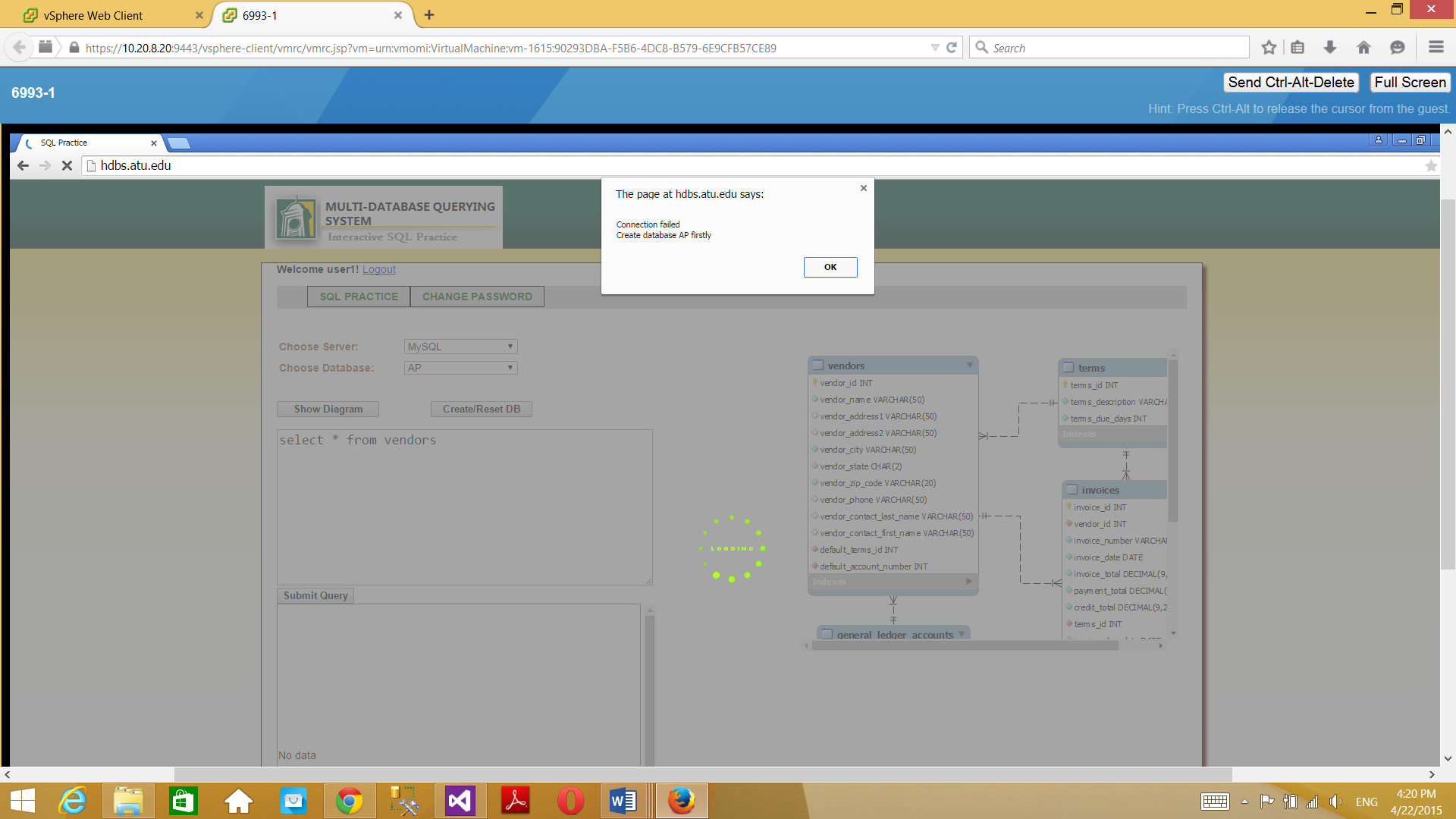
Y

Select Database

If it’s the first time a user practicing on the particular database, she has to create her own unique database firstly. The connection string is designed to connect to each user’s individual database.

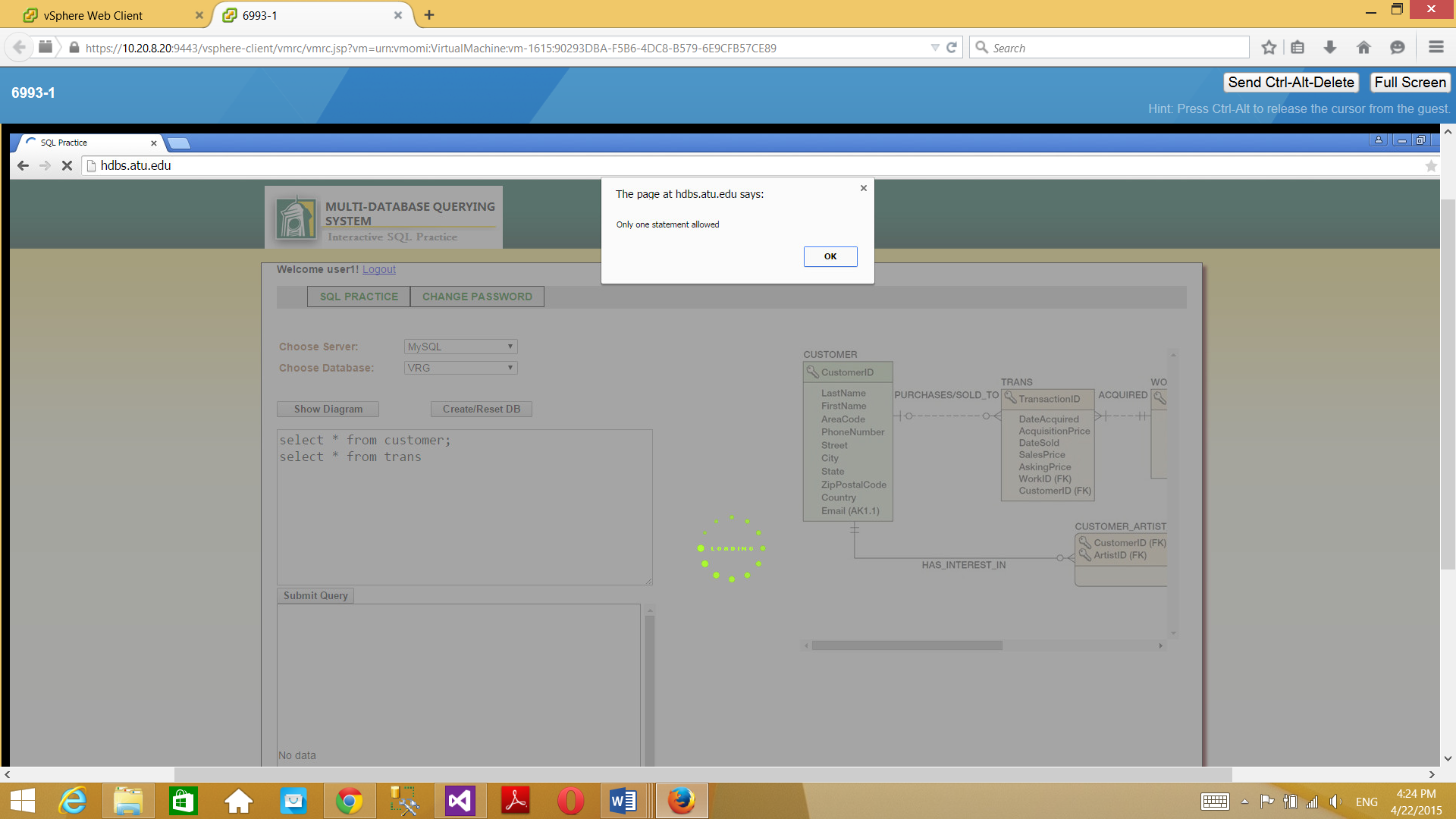
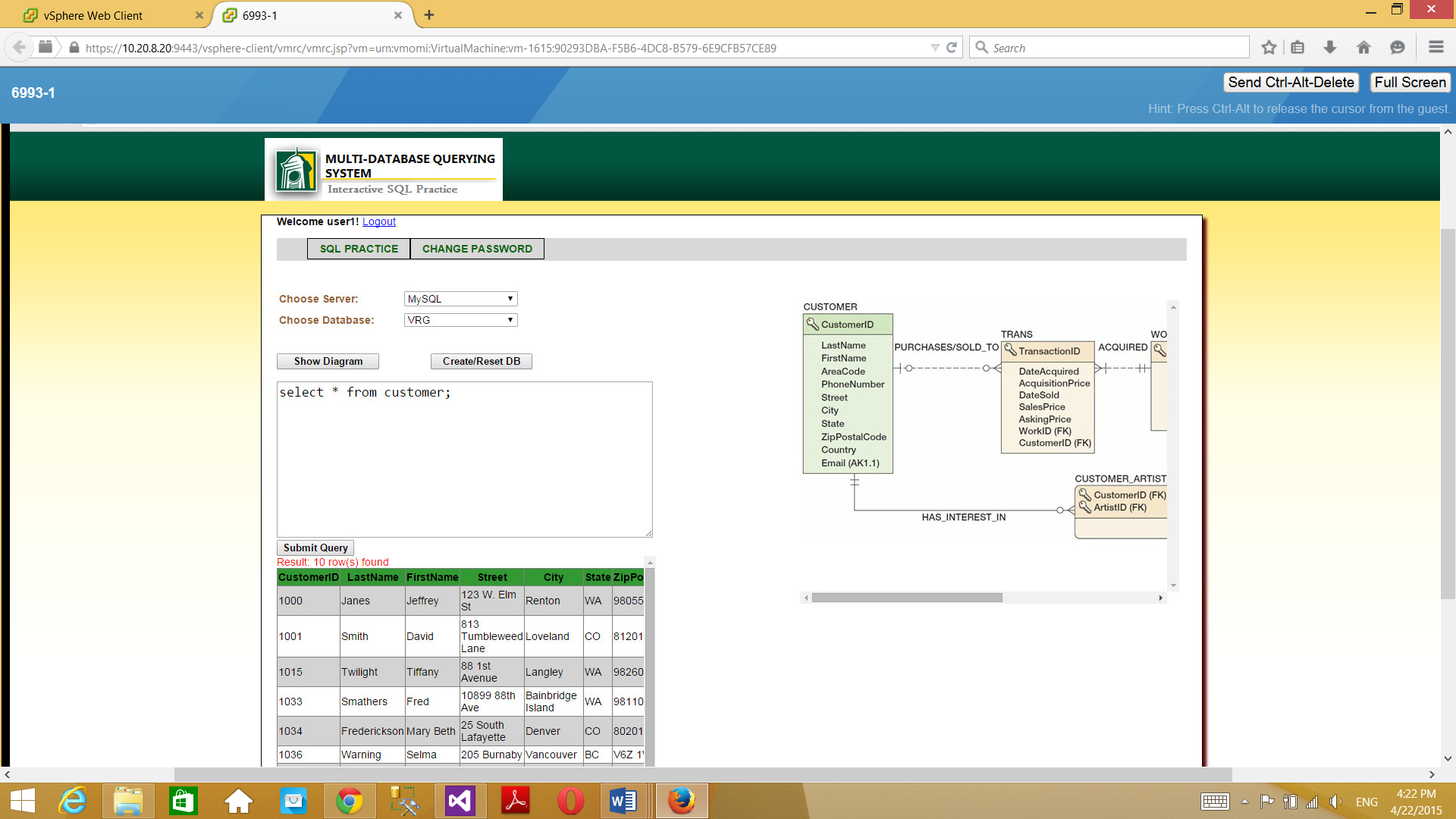
For the database creation function, I created a PowerShell Script to duplicate and rename the Access file and SQLite database file. I used a backup tool to backup and restore databases in MySQL. The package can be installed via NuGet. In SQL server, I created a stored procedure to restore databases from backup files. For PostgreSQL, I coded the commands to clone databases from standard system database template. And for IBM DB2 and Oracle, the system does not have this feature yet. This function requires the database file or backup file pre-created. Of course, the users can restore the database to its original contents at any time by clicking the Create/Reset DB button.

Figure 4 Database Creation



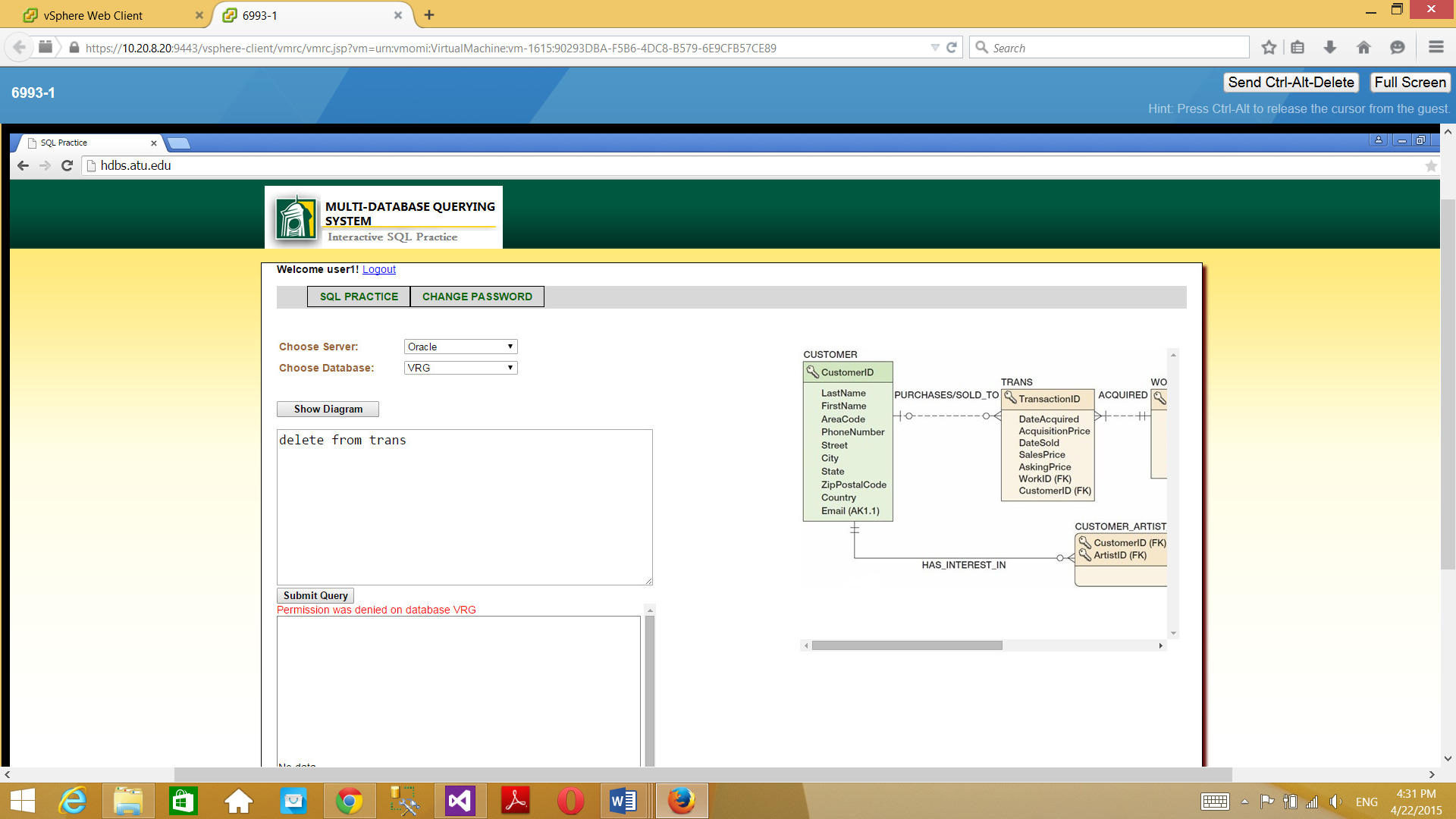
The editor is designed to allow only one statement to be executed at a time, if it contains zero or more than one statement, the error window will pop up to prevent further execution.

Figure 5 Text Editor Restriction

The users’ input can also be restricted to run specified SQL command according to the users’ group. For example, we can restrict the user run select-only queries in Oracle. The other statement will be denied to execute. So the users do not have the permission to change the data.

Figure 6 Text Editor Restriction



If the input contains syntax error or some errors occurred during the execution, it will return the error message instead.

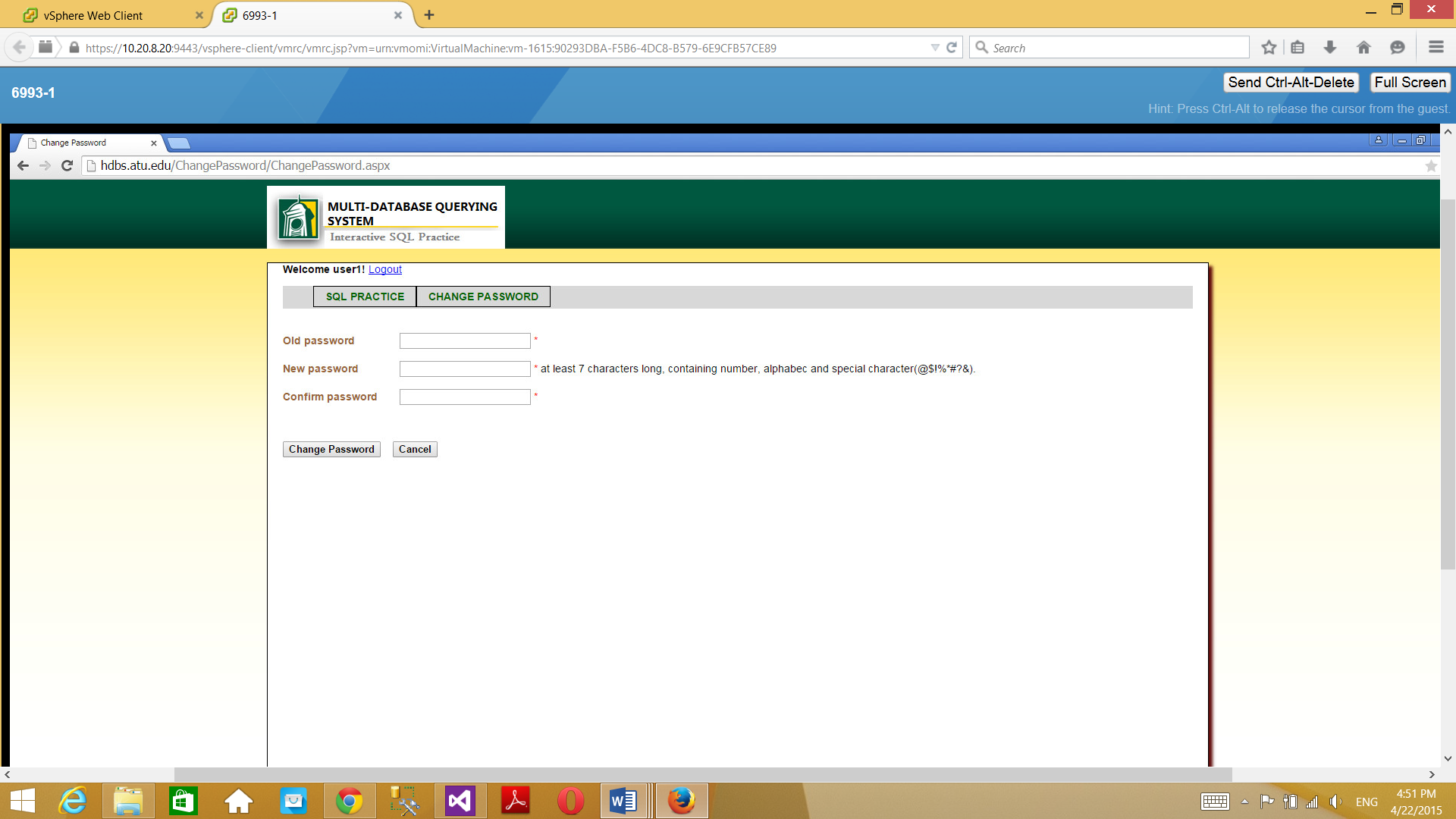
For SQL Server, there is a particular feature. I took the advantage of SQL Server’s transaction controls. When the user executing the non-select DML, the system will automatically insert the output statement into the query typed in the editor, then forward it to back-end server for execution. The results will be shown in the Grid View. The user can choose to commit or rollback the result. If the undo button is clicked, the result will rollback. If the save result button is clicked, the commit statement will be issued for this session.

I also created a progress bar for long time running operation by using ModalPopupExtender in AJAX toolkit.

* 1. Change Password Page

I added the required validator and password compare validator to accomplish the password change function.

Figure 7 Change Password Page



* 1. User Management Page

The user management page is for the site administrators or the people who are kind of controlling the site. They can manage user accounts and assign the users to different groups for permission management.

Figure 8 User Management Page

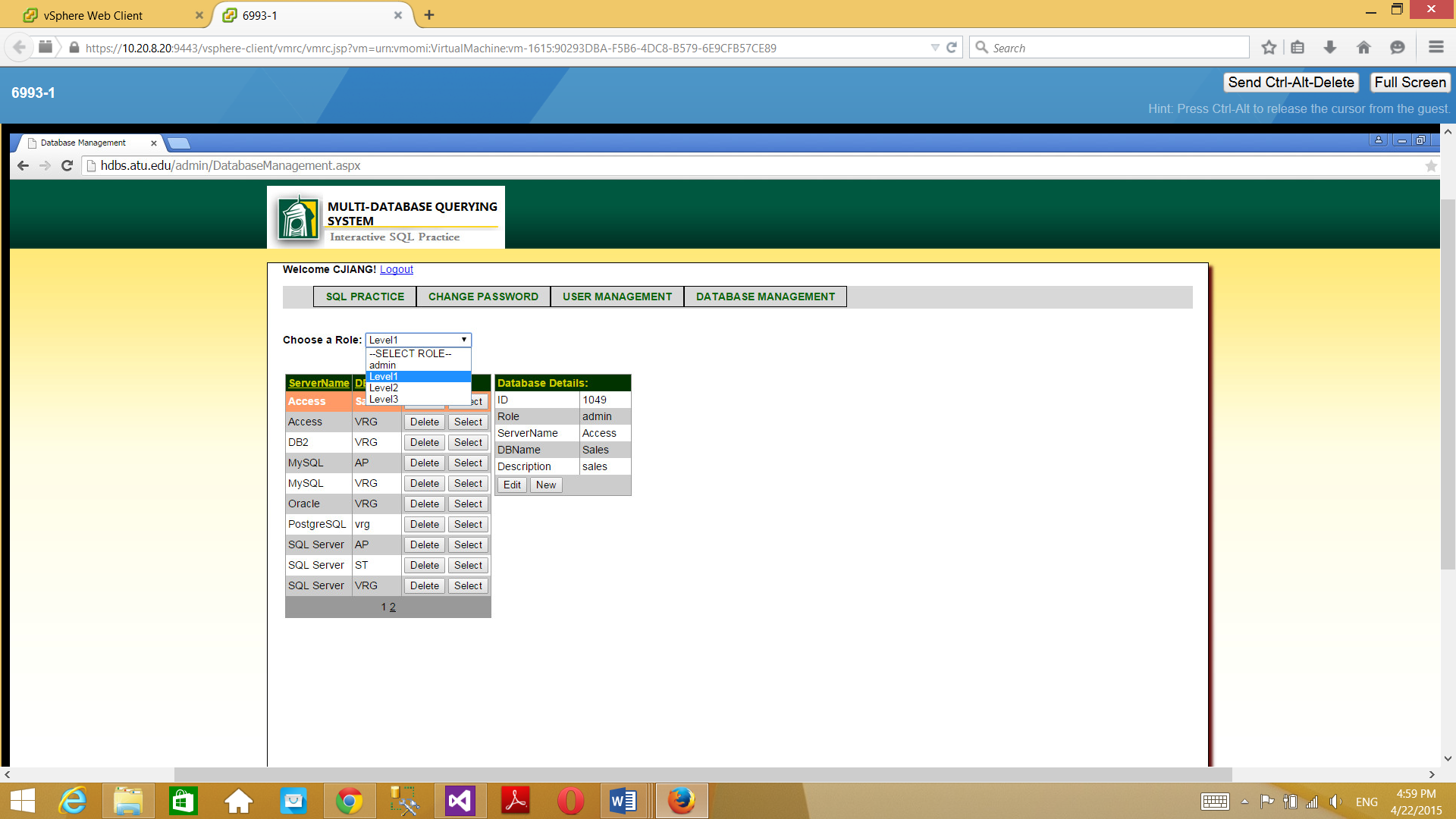


I used a Grid View control to load and store the users’ information and also use it to implement the inserting, deleting, paging, and sorting function.

* 1. Database Management Page

The group (role) name is associated with back-end database names. Different group can has different databases available to operate. The GridView-DetailsView control demonstrates a clearer way of inserting, editing, and deleting records.

Figure 9 Database Management Page



* 1. Back-end Database Creation

I implemented different databases in different DBMS for students’ practice which includes DBMS configuration, server implementation, user and schema creation.

1. Conclusion and Future Work

All that I have been done is I want to make something that takes the knowledge and skills I have learned such as C#, ASP.NET, HTML, JavaScript, and SQL. This website mainly focuses on the interaction between web application and back-end server, cares little about the security issue. Since this website requires constant access to the back-end database, even the real database name is kind of hidden to end user and the input can be restricted at the front-end, the user can still do something damage.

It is possible to work out a way to establish individual connection to the database for each user, executing stored procedure to grant proper access to the database or schema object and to control the ability to execute system operations. This can be the future work to improve the interactivity and the security of this system.

We can also do further development on this system to assess student learning. We can create test or quiz page and ask students to solve the given questions. The system will generate the feedback to correct the students’ input answer according to their actions. This may bring unexpected benefits and convenience for students learning.

1. User Manual

<https://youtu.be/W4oIbIbanUk>

1. Appendix A: Source Code

Web.config

<?xml version="1.0"?>

<!--

For more information on how to configure your ASP.NET application, please visit

http://go.microsoft.com/fwlink/?LinkId=169433

-->

<configuration>

<configSections>

</configSections>

<system.webServer>

<validation validateIntegratedModeConfiguration="false"/>

<handlers>

<remove name="ChartImageHandler"/>

<add name="ChartImageHandler" preCondition="integratedMode" verb="GET,HEAD,POST" path="ChartImg.axd" type="System.Web.UI.DataVisualization.Charting.ChartHttpHandler, System.Web.DataVisualization, Version=4.0.0.0, Culture=neutral, PublicKeyToken=31bf3856ad364e35"/>

</handlers>

</system.webServer>

<appSettings>

<add key="ValidationSettings:UnobtrusiveValidationMode" value="None"/>

<add key="ChartImageHandler" value="storage=file;timeout=20;dir=c:\TempImageFiles\;"/>

</appSettings>

<connectionStrings>

<add name="ManagementConnectionString" connectionString="Data Source=.;Initial Catalog=Management;Persist Security Info=True;User ID=sa;Password=\*\*\*\*\*\*" providerName="System.Data.SqlClient"/>

</connectionStrings>

<system.web>

<httpHandlers>

<add path="ChartImg.axd" verb="GET,HEAD,POST" type="System.Web.UI.DataVisualization.Charting.ChartHttpHandler, System.Web.DataVisualization, Version=4.0.0.0, Culture=neutral, PublicKeyToken=31bf3856ad364e35" validate="false"/>

</httpHandlers>

<authentication mode="Forms">

<forms loginUrl="~/Login.aspx" slidingExpiration="true" timeout="2880"/>

</authentication>

<compilation debug="true" targetFramework="4.5">

<assemblies>

<add assembly="System.Management, Version=4.0.0.0, Culture=neutral, PublicKeyToken=B03F5F7F11D50A3A"/>

<add assembly="System.Management.Automation, Version=3.0.0.0, Culture=neutral, PublicKeyToken=31BF3856AD364E35"/>

<add assembly="System.Data.Services.Client, Version=4.0.0.0, Culture=neutral, PublicKeyToken=B77A5C561934E089"/>

<add assembly="System.Web.DataVisualization, Version=4.0.0.0, Culture=neutral, PublicKeyToken=31BF3856AD364E35"/>

<add assembly="IBM.Data.DB2, Version=9.7.4.4, Culture=neutral, PublicKeyToken=7C307B91AA13D208"/>

</assemblies>

</compilation>

<httpRuntime targetFramework="4.5"/>

<pages>

<controls>

<add tagPrefix="ajaxToolkit" namespace="AjaxControlToolkit" assembly="AjaxControlToolkit"/>

<add tagPrefix="asp" namespace="System.Web.UI.DataVisualization.Charting" assembly="System.Web.DataVisualization, Version=4.0.0.0, Culture=neutral, PublicKeyToken=31bf3856ad364e35"/>

</controls>

</pages>

</system.web>

<location path="Admin">

<system.web>

<authorization>

<!-- Order and case are important below -->

<allow roles="admin"/>

<!--allow users="admin"/-->

<deny users="\*"/>

</authorization>

</system.web>

</location>

<system.data>

<!--

NOTE: The extra "remove" element below is to prevent the design-time

support components within EF6 from selecting the legacy ADO.NET

provider for SQLite (i.e. the one without any EF6 support). It

appears to only consider the first ADO.NET provider in the list

within the resulting "app.config" or "web.config" file.

-->

<DbProviderFactories>

<remove invariant="System.Data.SQLite"/>

<remove invariant="System.Data.SQLite.EF6"/>

<add name="SQLite Data Provider (Entity Framework 6)" invariant="System.Data.SQLite.EF6" description=".NET Framework Data Provider for SQLite (Entity Framework 6)" type="System.Data.SQLite.EF6.SQLiteProviderFactory, System.Data.SQLite.EF6"/>

<add name="SQLite Data Provider" invariant="System.Data.SQLite" description=".NET Framework Data Provider for SQLite" type="System.Data.SQLite.SQLiteFactory, System.Data.SQLite"/>

</DbProviderFactories>

</system.data>

</configuration>

Login Page Source Code

Login.aspx

<%@ Page Language="C#" AutoEventWireup="true" CodeFile="Login.aspx.cs" Inherits="Login" %>

<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml">

<head runat="server">

<title></title>

<style>

.modalBackground {

background-color:#808080;

filter:alpha(opacity=70);

opacity:0.7;}

</style>

</head>

<body>

<form id="form1" runat="server">

<div style="position:fixed; left:30%; top:30%">

<p style="font-family: 'Times New Roman', Times, serif; font-size: x-large; text-transform: capitalize; color: #800080; text-align: center">MULTI-DATABASE QUERYING SYSTEM</p>

<asp:Login ID="Login1" runat="server" style="margin:auto" BackColor="#FFD520" BorderPadding="4" BorderStyle="Solid" BorderWidth="1px" Font-Names="Times New Roman" Font-Size="Large" Height="260px" OnAuthenticate="Login1\_Authenticate" Width="450px" InstructionText="Please enter your user name and password for login." ValidateRequestMode="Enabled">

<CheckBoxStyle Font-Bold="False" Font-Italic="False" Font-Overline="False" Font-Size="Medium" Font-Strikeout="False" />

<InstructionTextStyle Font-Italic="True" ForeColor="Black" />

<LoginButtonStyle BackColor="Silver" BorderStyle="Solid" BorderWidth="1px" Font-Names="Times New Roman" Font-Size="Large" ForeColor="#00543D" Font-Bold="True" />

<TextBoxStyle Font-Size="0.8em" />

<TitleTextStyle BackColor="#00543D" Font-Bold="True" Font-Size="X-Large" ForeColor="White" />

</asp:Login>

<asp:ValidationSummary ID="ValidationSummary1" runat="server" Text="\*" ValidationGroup="Login1" ForeColor="Red" />

</div>

</form>

</body>

</html>

Login.aspx.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.UI;

using System.Web.UI.WebControls;

using System.Data.OleDb;

using System.Web.Security;

using System.Configuration;

using System.Data.SqlClient;

public partial class Login : System.Web.UI.Page

{

protected void Page\_Load(object sender, EventArgs e)

{

}

protected void Login1\_Authenticate(object sender, AuthenticateEventArgs e)

{

FormsAuthentication.Initialize();

string query = "Select Count(\*) From Users Where UserID = @ID And Password = @Pass COLLATE Latin1\_General\_CS\_AS";

int result = 0;

using (SqlConnection conn = new SqlConnection(ConfigurationManager.ConnectionStrings["ManagementConnectionString"].ConnectionString))

{

using (SqlCommand cmd = new SqlCommand(query, conn))

{

cmd.Parameters.AddWithValue("@ID", Login1.UserName);

cmd.Parameters.AddWithValue("@Pass", Login1.Password);

conn.Open();

result = (int)cmd.ExecuteScalar();

}

}

if (result > 0)

{

string checkrole = "update users set LastLoginDate=GETDATE() where userid='"

+ Login1.UserName.ToString()+ "';"

+ "select Role from users where userid = '" + Login1.UserName.ToString() + "';";

string usertype = "";

using (SqlConnection conn = new SqlConnection(ConfigurationManager.ConnectionStrings["ManagementConnectionString"].ConnectionString))

{

using (SqlCommand cmd = new SqlCommand(checkrole, conn))

{

conn.Open();

usertype = (string)cmd.ExecuteScalar();

}

}

string roles = "admin";

if (usertype == "admin")

{

// Create a new ticket used for authentication

FormsAuthenticationTicket ticket = new FormsAuthenticationTicket(

1, // Ticket version

Login1.UserName, // Username associated with ticket

DateTime.Now, // Date/time issued

DateTime.Now.AddMinutes(30), // Date/time to expire

true, // "true" for a persistent user cookie

roles, // User-data, in this case the roles

FormsAuthentication.FormsCookiePath);// Path cookie valid for

// Encrypt the cookie using the machine key for secure transport

string hash = FormsAuthentication.Encrypt(ticket);

HttpCookie cookie = new HttpCookie(

FormsAuthentication.FormsCookieName, // Name of auth cookie

hash); // Hashed ticket

// Set the cookie's expiration time to the tickets expiration time

if (ticket.IsPersistent) cookie.Expires = ticket.Expiration;

// Add the cookie to the list for outgoing response

Response.Cookies.Add(cookie);

Response.Redirect(@"~\admin\admin.aspx");

}

else

{

FormsAuthentication.RedirectFromLoginPage(Login1.UserName, Login1.RememberMeSet);

}

}

}

SQL Practice Page Source Code

Default.aspx

<%@ Page Language="C#" AutoEventWireup="true" CodeFile="Default.aspx.cs" Inherits="\_Default" %>

<%@ Register Assembly="AjaxControlToolkit" Namespace="AjaxControlToolkit" TagPrefix="cc1" %>

<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml">

<head runat="server">

<title>SQL Practice</title>

<link href="StyleSheet.css" rel="stylesheet" />

<script type="text/javascript">

function showPopup() {

var modalPopupBehavior = $find('ProgressBarModalPopupExtender');

modalPopupBehavior.show();

}

</script>

<style type="text/css">

#TextArea1 {

height: 54px;

width: 297px;

}

.auto-style1 {

width: 50%;

}

.auto-style2 {

width: 161px;

text-align: left;

}

.auto-style3 {

width: 196px;

text-align: left;

}

.auto-style4 {

width: 161px;

text-align: left;

height: 30px;

}

.auto-style5 {

width: 196px;

text-align: left;

height: 30px;

}

.modalBackground {

background-color:#808080;

filter:alpha(opacity=70);

opacity:0.7;}

</style>

</head>

<body>

<div id="header">

<div class="left">

<a href="/"><img src="../images/logo.png" alt="Logo" /></a>

</div>

</div>

<div id="wrapper">

<form id="form1" runat="server">

<asp:LoginView ID="LoginView1" Runat="server" Visible="true">

<LoggedInTemplate>

<asp:LoginName ID="LoginName1" runat="server" Font-Bold = "true" FormatString="Welcome {0}!"/>

</LoggedInTemplate>

<AnonymousTemplate>

You are not logged in

</AnonymousTemplate>

</asp:LoginView>

<asp:LoginStatus ID="LoginStatus1" runat="server" />

<div class="nav">

<ul>

<li><asp:HyperLink ID="MainPg" runat="server" NavigateUrl="~/Default.aspx">SQL Practice</asp:HyperLink></li>

<li><asp:HyperLink ID="ChangPwd" runat="server" NavigateUrl="~/ChangePassword/ChangePassword.aspx">Change Password</asp:HyperLink></li>

<li><asp:HyperLink ID="AdminPg" runat="server" NavigateUrl="~/Admin/Admin.aspx" Visible="False">User Management</asp:HyperLink></li>

<li><asp:HyperLink ID="DBPg" runat="server" NavigateUrl="~/Admin/DatabaseManagement.aspx" Visible="False">Database Management</asp:HyperLink></li>

</ul>

</div>

<br /><br />

<asp:SqlDataSource ID="ChooseServer" runat="server" ConnectionString="<%$ ConnectionStrings:ManagementConnectionString %>" SelectCommand="SELECT DISTINCT p.ServerName FROM Users AS u LEFT OUTER JOIN Permission AS p ON u.Role = p.Role WHERE (u.UserID = @UserID)" >

<SelectParameters>

<asp:SessionParameter DefaultValue="" Name="UserID" SessionField="LoginID" />

</SelectParameters>

</asp:SqlDataSource>

<div runat="server" id="imagediv" >

<asp:Image ID="Image1" runat="server" style="display:block; " />

</div>

<table class="auto-style1">

<tr>

<td class="auto-style4">

<asp:Label ID="server" runat="server" Text="Choose Server: " Font-Bold="True" ForeColor="#996633"></asp:Label>

</td>

<td class="auto-style5">

<asp:DropDownList ID="DropDownList1" AutoPostBack="true" runat="server" DataSourceID="ChooseServer" DataTextField="ServerName" DataValueField="ServerName" Width="150px" Height="20px" OnSelectedIndexChanged="DropDownList1\_SelectedIndexChanged" style="margin-left: 0px" AppendDataBoundItems="True">

<asp:ListItem Selected="True">--SELECT SERVER--</asp:ListItem>

</asp:DropDownList>

</td>

</tr>

<tr>

<td class="auto-style2">

<asp:Label ID="ChooseDB" runat="server" Text="Choose Database: " Font-Bold="True" ForeColor="#996633"></asp:Label>

</td>

<td class="auto-style3">

<asp:DropDownList ID="DropDownList2" runat="server" Height="18px" Width="150px" DataSourceID="DataBase" DataTextField="DBName" DataValueField="DBName" style=" margin-left: 0px" AutoPostBack="True" OnSelectedIndexChanged="DropDownList2\_SelectedIndexChanged">

</asp:DropDownList>

</td>

<td>&nbsp;</td>

</tr>

</table>

<br />

<asp:SqlDataSource ID="DataBase" runat="server" ConnectionString="<%$ ConnectionStrings:ManagementConnectionString %>" SelectCommand="SELECT DISTINCT p.DBName FROM Users AS u LEFT OUTER JOIN Permission AS p ON u.Role = p.Role WHERE (u.UserID = @UserID) AND (p.ServerName = @ServerName)">

<SelectParameters>

<asp:SessionParameter Name="UserID" SessionField="LoginID" />

<asp:ControlParameter ControlID="DropDownList1" Name="ServerName" PropertyName="SelectedValue" Type="String" />

</SelectParameters>

</asp:SqlDataSource>

<asp:SqlDataSource ID="Role" runat="server" ConnectionString="<%$ ConnectionStrings:ManagementConnectionString %>" SelectCommand="SELECT [Role] FROM [Users] WHERE ([UserID] = @UserID)">

<SelectParameters>

<asp:SessionParameter Name="UserID" SessionField="LoginID" Type="String" />

</SelectParameters>

</asp:SqlDataSource>

<asp:Label ID="output" runat="server" Text="Output: "></asp:Label>

<br />

<asp:Button ID="Button2" runat="server" OnClick="Button2\_Click" Text="Show Diagram" Width="135px" Font-Bold="True"/>

&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;

<asp:Button ID="Button3" title="Restore the database back to its original content" runat="server" OnClick="Button3\_Click" OnClientClick="showPopup()" Text="Create/Reset DB" Width="134px" Font-Bold="True"/>

<br /><br />

<asp:TextBox ID="TextBox1" runat="server" Height="200px" Width="490px" font-size="18px" Wrap="True" TextMode="MultiLine"></asp:TextBox>

<br />

<asp:Button ID="Button1" runat="server" OnClick="Button1\_Click" OnClientClick="showPopup()" Text="Submit Query" Font-Bold="True"/>

&nbsp;&nbsp;&nbsp;

<asp:Button ID="commit" runat="server" OnClick="commit\_Click" Text="Save Result" Visible="false" Enabled="False" Font-Bold="True"/>

&nbsp;&nbsp;&nbsp;

<asp:Button ID="rollback" runat="server" OnClick="rollback\_Click" Text="Undo" Visible="false" Enabled="False" Font-Bold="True"/>

<br />

<div id="result" style="width: 500px; height: 400px; overflow: scroll">

<asp:label id="msg" runat="server" forecolor="Red" />

<asp:GridView ID="GridView1" runat="server" Height="400px" Width="480px" style="overflow:scroll" emptydatatext="No data">

<HeaderStyle BackColor="#339933" Font-Bold="True" />

<SelectedRowStyle BackColor="#FFFF99" />

</asp:GridView>

</div>

<br />

<!--end of div content-->

<br />

<br />

<cc1:ToolkitScriptManager runat="Server" ID="ToolkitScriptManager1" />

<asp:UpdatePanel ID="UpdatePanel1" runat="server">

<%--<Triggers>

<asp:AsyncPostBackTrigger ControlID="Button1" EventName="Click"/>

</Triggers>--%>

</asp:UpdatePanel>

<asp:HiddenField ID="hiddenField" runat="server" />

<cc1:ModalPopupExtender runat="server" ID="ProgressBarModalPopupExtender"

BehaviorID="ProgressBarModalPopupExtender"

TargetControlID="hiddenField"

PopupControlID="Panel1"

BackgroundCssClass="modalBackground"

RepositionMode="RepositionOnWindowScroll" >

</cc1:ModalPopupExtender>

<asp:Panel ID="Panel1" runat="server" Style="display: none; ">

<img src="images/loader.gif" />

</asp:Panel>

</form>

</div>

</body>

</html>

Default.aspx.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.UI;

using System.Web.UI.WebControls;

using System.Data.SqlClient;

using System.Data;

using MySql.Data.MySqlClient;

using System.Data.SQLite;

using System.Data.OleDb;

using System.Drawing;

using System.Web.Security;

using System.Configuration;

using System.Text.RegularExpressions;

using System.Management.Automation; //Powershell Reference

using System.Management.Automation.Runspaces;

using Npgsql;

using Oracle.ManagedDataAccess.Client;

using Oracle.ManagedDataAccess.Types;

using IBM.Data.DB2;

public partial class \_Default : System.Web.UI.Page

{

protected void Page\_Load(object sender, EventArgs e)

{

if (this.Page.User.Identity.IsAuthenticated)

{

Session["LoginID"] = User.Identity.Name;

GetRole();

if (User.IsInRole("admin"))

{

AdminPg.Visible = true;

DBPg.Visible = true;

}

else

{

AdminPg.Visible = false;

DBPg.Visible = false;

}

}

else

{

FormsAuthentication.RedirectToLoginPage("~/login.aspx");

}

}

protected void GetRole()

{

string query = "SELECT [Role] FROM [Users] WHERE [UserID] = '" + Session["LoginID"] + "'";

using (SqlConnection conn = new SqlConnection(ConfigurationManager.ConnectionStrings["ManagementConnectionString"].ConnectionString))

{

using (SqlCommand cmd = new SqlCommand(query, conn))

{

conn.Open();

using (SqlDataReader r = cmd.ExecuteReader())

{

if (r.HasRows)

{

while (r.Read())

{

Session["Role"] = r["role"];

}

}

}

}

}

}

protected void DropDownList1\_SelectedIndexChanged(object sender, EventArgs e)

{

DropDownList2.Style.Add("display", "block");

imagediv.Style.Add("display", "none");

TextBox1.Text = "";

msg.Text = "";

GridView1.DataSource = null;

GridView1.DataBind();

TextBox1.Attributes.Remove("readonly");

Button1.Enabled = true;

switch (DropDownList1.SelectedItem.Text)

{

case "SQL Server":

commit.Visible = true;

rollback.Visible = true;

Button3.Visible = true;

commit.Enabled = false;

rollback.Enabled = false;

break;

case "Oracle":

case "DB2":

Button3.Visible = false;

commit.Visible = false;

rollback.Visible = false;

break;

default:

Button3.Visible = true;

commit.Visible = false;

rollback.Visible = false;

break;

}

}

protected void DropDownList2\_SelectedIndexChanged(object sender, EventArgs e)

{

Image1.ImageUrl = "images/" + DropDownList1.SelectedItem.Text + DropDownList2.SelectedItem.Text + ".jpg";

TextBox1.Text = "";

msg.Text = "";

GridView1.DataSource = null;

GridView1.DataBind();

TextBox1.Attributes.Remove("readonly");

Button1.Enabled = true;

commit.Enabled = false;

rollback.Enabled = false;

}

protected void Button2\_Click(object sender, EventArgs e)

{

if (DropDownList2.SelectedItem != null)

{

Image1.ImageUrl = "images/" + DropDownList1.SelectedItem.Text + DropDownList2.SelectedItem.Text + ".jpg";

imagediv.Style.Add("display", "block");

}

else

{

Header.Controls.Add(new LiteralControl("<script type=\"text/javascript\">alert('Error\\nPlease choose a database');</script>"));

}

}

protected void Button3\_Click(object sender, EventArgs e)

{

Button1.Enabled = true;

commit.Enabled = false;

rollback.Enabled = false;

GridView1.DataSource = null;

GridView1.DataBind();

msg.Text = "";

switch (DropDownList1.SelectedItem.Text)

{

case "SQL Server":

string s = "Data Source=localhost;Initial Catalog=master" + "; User ID=sa; Password=\*\*\*\*\*\*;";

SqlConnection ProcConn = new SqlConnection(s);

try

{

ProcConn.Open();

SqlCommand cmd = new SqlCommand("spCreateUserDb", ProcConn);

cmd.CommandType = CommandType.StoredProcedure;

cmd.Parameters.Add(

new SqlParameter("@sourceDb", DropDownList2.SelectedItem.Text));

cmd.Parameters.Add(

new SqlParameter("@destDb", User.Identity.Name + DropDownList2.SelectedItem.Text));

cmd.ExecuteNonQuery();

Header.Controls.Add(new LiteralControl("<script type=\"text/javascript\">alert('Database " + DropDownList2.Text + " has been created');</script>"));

output.Text = "Database " + DropDownList2.SelectedItem.Text + " has been created";

}

catch (Exception)

{

Header.Controls.Add(new LiteralControl("<script type=\"text/javascript\">alert('Database "+ DropDownList2.Text + " creation failed');</script>"));

}

finally

{

if (ProcConn != null)

{

ProcConn.Close();

}

}

break;

case "Access":

RunspaceConfiguration runspaceConfiguration = RunspaceConfiguration.Create();

try

{

using (Runspace runspace = RunspaceFactory.CreateRunspace(runspaceConfiguration))

{

runspace.Open();

RunspaceInvoke scriptInvoker = new RunspaceInvoke(runspace);

Pipeline pipeline = runspace.CreatePipeline();

string scriptfile = "C:\\SourceDatabase\\Access\\powershellCreateDB.ps1";

//add a new script with arguments

Command myCommand = new Command(scriptfile);

CommandParameter SourceParam = new CommandParameter("source", DropDownList2.SelectedItem.Text);

CommandParameter TargetParam = new CommandParameter("target", User.Identity.Name);

myCommand.Parameters.Add(SourceParam);

myCommand.Parameters.Add(TargetParam);

pipeline.Commands.Add(myCommand);

pipeline.Invoke();

}

Header.Controls.Add(new LiteralControl("<script type=\"text/javascript\">alert('Database " + DropDownList2.Text + " has been created');</script>"));

output.Text = "Database " + DropDownList2.SelectedItem.Text + " has been created";

}

catch (Exception ex)

{

Header.Controls.Add(new LiteralControl("<script type=\"text/javascript\">alert('Database " + DropDownList2.Text + " creation failed');</script>"));

msg.Text = ex.Message;

}

break;

case "MySQL":

string str = "server=localhost;Database=" + DropDownList2.SelectedItem.Text + ";Uid=root;Pwd=\*\*\*\*\*\*;";

string file = "C:\\SourceDatabase\\MySQL\\" + DropDownList2.SelectedItem.Text + ".sql";

try

{

using (MySqlConnection conn = new MySqlConnection(str))

{

using (MySqlCommand cmd = new MySqlCommand())

{

cmd.CommandText = "DROP DATABASE IF EXISTS " + User.Identity.Name + DropDownList2.SelectedItem.Text;

cmd.Connection = conn;

conn.Open();

cmd.ExecuteNonQuery();

using (MySqlBackup mb = new MySqlBackup(cmd))

{

mb.ImportInfo.TargetDatabase = User.Identity.Name + DropDownList2.SelectedItem.Text;

mb.ImportFromFile(file);

}

}

}

Header.Controls.Add(new LiteralControl("<script type=\"text/javascript\">alert('Database " + DropDownList2.Text + " has been created');</script>"));

output.Text = "Database " + DropDownList2.SelectedItem.Text + " has been created";

}

catch (Exception ex)

{

Header.Controls.Add(new LiteralControl("<script type=\"text/javascript\">alert('Database " + DropDownList2.Text + " creation failed');</script>"));

output.Text = ex.Message;

}

break;

case "PostgreSQL":

string ps = "Server=127.0.0.1;Port=5432;Database=template1;User Id=postgres; Password=\*\*\*\*\*\*;Pooling=false;";

try

{

using (NpgsqlConnection conn = new NpgsqlConnection(ps))

{

using (NpgsqlCommand cmd = new NpgsqlCommand())

{

cmd.CommandText = "DROP DATABASE IF EXISTS " + User.Identity.Name + DropDownList2.SelectedItem.Text;

cmd.Connection = conn;

conn.Open();

cmd.ExecuteNonQuery();

cmd.CommandText = "CREATE DATABASE " + User.Identity.Name + DropDownList2.SelectedItem.Text + " TEMPLATE " + DropDownList2.SelectedItem.Text;

cmd.ExecuteNonQuery();

}

}

Header.Controls.Add(new LiteralControl("<script type=\"text/javascript\">alert('Database " + DropDownList2.Text + " has been created');</script>"));

output.Text = "Database " + DropDownList2.SelectedItem.Text + " has been created";

}

catch (Exception ex)

{

Header.Controls.Add(new LiteralControl("<script type=\"text/javascript\">alert('Database " + DropDownList2.Text + " creation failed');</script>"));

output.Text = ex.Message;

}

break;

case "SQLite":

RunspaceConfiguration SQLiterunspaceConfiguration = RunspaceConfiguration.Create();

try

{

using (Runspace runspace = RunspaceFactory.CreateRunspace(SQLiterunspaceConfiguration))

{

runspace.Open();

RunspaceInvoke scriptInvoker = new RunspaceInvoke(runspace);

Pipeline pipeline = runspace.CreatePipeline();

string scriptfile = "C:\\SourceDatabase\\SQLite\\powershellCreateDB.ps1";

//add a new script with arguments

Command myCommand = new Command(scriptfile);

CommandParameter SourceParam = new CommandParameter("source", DropDownList2.SelectedItem.Text);

CommandParameter TargetParam = new CommandParameter("target", User.Identity.Name);

myCommand.Parameters.Add(SourceParam);

myCommand.Parameters.Add(TargetParam);

pipeline.Commands.Add(myCommand);

pipeline.Invoke();

}

Header.Controls.Add(new LiteralControl("<script type=\"text/javascript\">alert('Database " + DropDownList2.Text + " has been created');</script>"));

output.Text = "Database " + DropDownList2.SelectedItem.Text + " has been created";

}

catch (Exception ex)

{

Header.Controls.Add(new LiteralControl("<script type=\"text/javascript\">alert('Database " + DropDownList2.Text + " creation failed');</script>"));

msg.Text = ex.Message;

}

break;

case "Oracle":

case "DB2":

Header.Controls.Add(new LiteralControl("<script type=\"text/javascript\">alert('Error\\nDataBase already exists, no need to create.');</script>"));

break;

default:

Header.Controls.Add(new LiteralControl("<script type=\"text/javascript\">alert('Error\\nPlease choose a database');</script>"));

break;

}

}

protected void Button1\_Click(object sender, EventArgs e)

{

GridView1.DataSource = null;

GridView1.DataBind();

msg.Text = "";

commit.Enabled = false;

rollback.Enabled = false;

switch (DropDownList1.SelectedItem.Text)

{

case "SQL Server":

string inputPattern = @"^[\s\;]\*(?i)\bdelete\b|\binsert\b|\bupdate\b|\bdrop\b";

Regex input = new Regex(inputPattern);

Match inputmatch = input.Match(TextBox1.Text);

if (inputmatch.Success)

{

commit.Enabled = true;

rollback.Enabled = true;

Button1.Enabled = false;

TextBox1.Attributes.Add("readonly", "true");

SqlConnection(" rollback");

}

else

{

SqlConnection(String.Empty);

}

break;

case "MySQL":

MySqlConnection();

break;

case "SQLite":

SqliteConnection();

break;

case "Access":

AccessConnection();

break;

case "PostgreSQL":

PostgreConnection();

break;

case "Oracle":

OracleConnection();

break;

case "DB2":

DB2Connection();

break;

default:

Header.Controls.Add(new LiteralControl("<script type=\"text/javascript\">alert('Error!\\nPlease Choose a Database!');</script>"));

break;

}

}

protected void SqlConnection(string end)

{

msg.Text = "";

string Text1 = TextBox1.Text.ToString().ToLower();

int index = Text1.Length;

List<string> tablenames = new List<string>();

string tablename = "";

string inputPattern = @"^[\s\;]\*\w+[^\;]\*[\;\s]\*$";

Regex input = new Regex(inputPattern);

Match inputmatch = input.Match(TextBox1.Text);

if (inputmatch.Success)

{

string s = "Data Source=localhost;Initial Catalog=" + User.Identity.Name + DropDownList2.SelectedItem.Text + "; User ID=sa; Password=\*\*\*\*\*\*;Pooling=false;";

SqlConnection myConnection = new SqlConnection(s);

SqlCommand cmd = new SqlCommand();

SqlDataReader r = null;

try

{

myConnection.Open();

if (myConnection.State == ConnectionState.Open)

{

try

{

cmd.CommandText = "SELECT TABLE\_NAME FROM INFORMATION\_SCHEMA.TABLES WHERE TABLE\_TYPE = 'BASE TABLE'";

cmd.Connection = myConnection;

r = cmd.ExecuteReader();

while (r.Read())

{

tablenames.Add(r.GetString(0).ToLower());

}

r.Close();

string firstWord = Regex.Match(TextBox1.Text, @"\w+\b").ToString().ToLower();

var pattern = new Regex(@"\W");

bool isFound = pattern.Split(Text1).Any(w => tablenames.Contains(w));

if (isFound)

{

tablename = pattern.Split(Text1).First(w => tablenames.Contains(w));

index = Text1.IndexOf(tablename) + tablename.Length;

switch (firstWord)

{

case "delete":

cmd.CommandText = "begin tran " + Text1.Insert(index, " output deleted.\* ") + end;

break;

case "insert":

Regex containCol = new Regex(tablename + @"\s\*\(");

if (containCol.IsMatch(Text1))

{

index = Text1.IndexOf(")") + 1;

}

cmd.CommandText = "begin tran " + Text1.Insert(index, " output inserted.\* ") + end;

break;

case "update":

int a = Text1.IndexOf("from");

int b = Text1.IndexOf("where");

Text1 = Text1.Replace(";", ""); //for insert output clause at the end of the query except ";"

if (a >= 0 && b >= 0)

index = Math.Min(a, b);

else

if (a < 0 && b < 0)

index = Text1.Length;

else

index = Math.Max(a, b);

cmd.CommandText = "begin tran " + Text1.Insert(index, " output inserted.\*, deleted.\* ") + end;

break;

default:

cmd.CommandText = "begin tran " + Text1 + end; //in case input 'select \* from x delete y'

break;

}

}

else

{cmd.CommandText = Text1;}

cmd.Connection = myConnection;

DataTable dt = new DataTable();

using (SqlDataAdapter adapter = new SqlDataAdapter(cmd))

{

adapter.Fill(dt);

}

GridView1.DataSource = dt;

if (firstWord == "select")

{

r = cmd.ExecuteReader();

int rowCount = 0;

if (r.HasRows)

{

while (r.Read())

{

rowCount++;

}

msg.Text = "Result: " + rowCount + " row(s) found";

}

else

{

msg.Text = "Result: 0 row(s) found.<br /><span style='background-color:#339933; color:black; font-size: 15pt'>";

for (int i = 0; i < r.FieldCount; i++)

{

msg.Text += r.GetName(i) + " |";

}

msg.Text += "</span><br />";

}

r.Close();

}

else

{

int numberOfRecords = cmd.ExecuteNonQuery();

if (numberOfRecords > 0)

{

msg.Text = "Result: " + numberOfRecords + " row(s) affected.";

}

else

{

msg.Text = "Excuted sucessfully.";

}

}

}

catch (Exception ex)

{

if (ex.Message.ToLowerInvariant().Contains("output"))

{

// do whatever to generate the real error msg

msg.Text = ex.Message.Replace("output", tablename);

}

else

{

msg.Text = ex.Message.Replace(User.Identity.Name, "");

}

Button1.Enabled = true;

TextBox1.Attributes.Remove("readonly");

commit.Enabled = false;

rollback.Enabled = false;

}

finally

{

if (myConnection != null)

{

myConnection.Close();

}

GridView1.DataBind();

FormatView();

}

}

}

catch (SqlException) { Header.Controls.Add(new LiteralControl("<script type=\"text/javascript\">alert('Connection failed\\nCreate database "+ DropDownList2.Text + " firstly');</script>")); }

}

else

{

Header.Controls.Add(new LiteralControl("<script type=\"text/javascript\">alert('Only one statement allowed');</script>"));

Button1.Enabled = true;

TextBox1.Attributes.Remove("readonly");

commit.Enabled = false;

rollback.Enabled = false;

}

}

protected void MySqlConnection()

{

string Text1 = TextBox1.Text;

string inputPattern = @"^[\s\;]\*\w+[^\;]\*[\;\s]\*$";

Regex input = new Regex(inputPattern);

Match inputmatch = input.Match(TextBox1.Text);

if (inputmatch.Success)

{

string s = "Server=localhost;Database=" + User.Identity.Name + DropDownList2.SelectedItem.Text + ";Uid=root;Pwd=\*\*\*\*\*\*;";

MySqlConnection myConnection = new MySqlConnection(s);

MySqlCommand cmd = new MySqlCommand();

MySqlDataReader r = null;

MySqlDataAdapter adapter = new MySqlDataAdapter(Text1, myConnection);

try

{

myConnection.Open();

if (myConnection.State == ConnectionState.Open)

{

try

{

cmd.CommandText = TextBox1.Text;

cmd.Connection = myConnection;

//myConnection.Open();

string firstWord = Regex.Match(TextBox1.Text, @"\w+\b").ToString().ToLower();

if (firstWord == "select")

{

DataTable dt = new DataTable();

adapter.Fill(dt);

GridView1.DataSource = dt;

r = cmd.ExecuteReader();

int rowCount = 0;

if (r.HasRows) //results>0

{

while (r.Read())

{

rowCount++;

}

msg.Text = "Result: " + rowCount + " row(s) found";

}

else

{

msg.Text = "Result: 0 row(s) found.<br /><span style='background-color:#339933; color:black; font-size: 15pt'>";

for (int i = 0; i < r.FieldCount; i++)

{

msg.Text += r.GetName(i) + " |";

}

msg.Text += "</span><br />";

}

r.Close();

}

else

{

int numberOfRecords = cmd.ExecuteNonQuery();

msg.Text = "Result: " + numberOfRecords + " row(s) affected.";

}

}

catch (Exception ex)

{

msg.Text = ex.Message.Replace(User.Identity.Name, "");

}

finally

{

myConnection.Close();

GridView1.DataBind();

FormatView();

}

}

}

catch (Exception) { Header.Controls.Add(new LiteralControl("<script type=\"text/javascript\">alert('Connection failed\\nCreate database " + DropDownList2.Text + " firstly');</script>")); }

}

else

{

Header.Controls.Add(new LiteralControl("<script type=\"text/javascript\">alert('Only one statement allowed');</script>"));

}

}

protected void SqliteConnection()

{

string Text1 = TextBox1.Text;

string inputPattern = @"^[\s\;]\*\w+[^\;]\*[\;\s]\*$";

Regex input = new Regex(inputPattern);

Match inputmatch = input.Match(TextBox1.Text);

if (inputmatch.Success)

{

string s = @"Data Source=C:\TargetDatabase\SQLite\" + User.Identity.Name + DropDownList2.SelectedItem.Text + ".db;Version=3;";

SQLiteConnection myConnection = new SQLiteConnection(s);

SQLiteCommand cmd = new SQLiteCommand();

SQLiteDataReader r = null;

SQLiteDataAdapter adapter = new SQLiteDataAdapter(Text1, myConnection);

try

{

cmd.CommandText = TextBox1.Text;

cmd.Connection = myConnection;

myConnection.Open();

string firstWord = Regex.Match(TextBox1.Text, @"\w+\b").ToString().ToLower();

if (firstWord == "select")

{

DataTable dt = new DataTable();

adapter.Fill(dt);

GridView1.DataSource = dt;

r = cmd.ExecuteReader();

int rowCount = 0;

if (r.HasRows) //results>0

{

while (r.Read())

{

rowCount++;

}

msg.Text = "Result: " + rowCount + " row(s) found";

}

else

{

msg.Text = "Result: 0 row(s) found.<br /><span style='background-color:#339933; color:black; font-size: 15pt'>";

for (int i = 0; i < r.FieldCount; i++)

{

msg.Text += r.GetName(i) + " |";

}

msg.Text += "</span><br />";

}

r.Close();

}

else

{

int numberOfRecords = cmd.ExecuteNonQuery();

msg.Text = "Result: " + numberOfRecords + " row(s) affected.";

}

}

catch (Exception ex)

{

msg.Text = ex.Message;

}

finally

{

GridView1.DataBind();

FormatView();

}

}

else

{

Header.Controls.Add(new LiteralControl("<script type=\"text/javascript\">alert('Only one statement allowed');</script>"));

}

}

protected void AccessConnection()

{

string Text1 = TextBox1.Text;

string s = @"Provider=Microsoft.ACE.OLEDB.12.0;Data Source=C:\TargetDatabase\Access\" + User.Identity.Name + DropDownList2.SelectedItem.Text + ".accdb;Persist Security Info=False;";

OleDbConnection myConnection = new OleDbConnection(s);

OleDbCommand cmd = new OleDbCommand();

OleDbDataReader r = null;

OleDbDataAdapter adapter = new OleDbDataAdapter(Text1, myConnection);

try

{

myConnection.Open();

if (myConnection.State == ConnectionState.Open)

{

try

{

cmd.CommandText = TextBox1.Text;

cmd.Connection = myConnection;

string firstWord = Regex.Match(TextBox1.Text, @"\w+\b").ToString().ToLower();

if (firstWord == "select")

{

DataTable dt = new DataTable();

adapter.Fill(dt);

GridView1.DataSource = dt;

r = cmd.ExecuteReader();

int rowCount = 0;

if (r.HasRows) //results>0

{

while (r.Read())

{

rowCount++;

}

msg.Text = "Result: " + rowCount + " row(s) found";

}

else

{

msg.Text = "Result: 0 row(s) found.<br /><span style='background-color:#339933; color:black; font-size: 15pt'>";

for (int i = 0; i < r.FieldCount; i++)

{

msg.Text += r.GetName(i) + " |";

}

msg.Text += "</span><br />";

}

r.Close();

}

else

{

int numberOfRecords = cmd.ExecuteNonQuery();

msg.Text = "Result: " + numberOfRecords + " row(s) affected.";

}

}

catch (Exception ex)

{

msg.Text = ex.Message.Replace(User.Identity.Name,"");

}

finally

{

myConnection.Close();

GridView1.DataBind();

FormatView();

}

}

}

catch (Exception e)

{

Header.Controls.Add(new LiteralControl("<script type=\"text/javascript\">alert('Connection failed\\nCreate database " + DropDownList2.Text + " firstly');</script>"));

output.Text = e.Message;

}

}

protected void PostgreConnection()

{

string Text1 = TextBox1.Text;

string inputPattern = @"^[\s\;]\*\w+[^\;]\*[\;\s]\*$";

Regex input = new Regex(inputPattern);

Match inputmatch = input.Match(TextBox1.Text);

if (inputmatch.Success)

{

string s = @"Server=127.0.0.1;Port=5432;Database=" + User.Identity.Name + DropDownList2.SelectedItem.Text + ";User Id=postgres; Password=\*\*\*\*\*\*;Pooling=false;";

NpgsqlConnection myConnection = new NpgsqlConnection(s);

NpgsqlCommand cmd = new NpgsqlCommand();

NpgsqlDataReader r = null;

NpgsqlDataAdapter adapter = new NpgsqlDataAdapter(Text1, myConnection);

try

{

cmd.CommandText = TextBox1.Text;

cmd.Connection = myConnection;

myConnection.Open();

string firstWord = Regex.Match(TextBox1.Text, @"\w+\b").ToString().ToLower();

if (firstWord == "select")

{

DataTable dt = new DataTable();

adapter.Fill(dt);

GridView1.DataSource = dt;

r = cmd.ExecuteReader();

int rowCount = 0;

if (r.HasRows) //results>0

{

while (r.Read())

{

rowCount++;

}

msg.Text = "Result: " + rowCount + " row(s) found";

}

else

{

msg.Text = "Result: 0 row(s) found.<br /><span style='background-color:#339933; color:black; font-size: 15pt'>";

for (int i = 0; i < r.FieldCount; i++)

{

msg.Text += r.GetName(i) + " |";

}

msg.Text += "</span><br />";

}

r.Close();

}

else

{

int numberOfRecords = cmd.ExecuteNonQuery();

msg.Text = "Result: " + numberOfRecords + " row(s) affected.";

}

}

catch (Exception ex)

{

msg.Text = ex.Message;

}

finally

{

myConnection.Close();

GridView1.DataBind();

FormatView();

}

}

else

{

Header.Controls.Add(new LiteralControl("<script type=\"text/javascript\">alert('Only one statement allowed');</script>"));

}

}

protected void OracleConnection()

{

string Text1 = TextBox1.Text;

string inputPattern = @"^[\s\;]\*\w+[^\;]\*[\;\s]\*$";

Regex input = new Regex(inputPattern);

Match inputmatch = input.Match(TextBox1.Text);

if (inputmatch.Success)

{

string s = @"Data Source=(DESCRIPTION=

(ADDRESS\_LIST=(ADDRESS=(PROTOCOL=TCP)(HOST=localhost )(PORT=1521)))

(CONNECT\_DATA=(SERVER=DEDICATED)(SERVICE\_NAME=" + DropDownList2.SelectedItem.Text + "))); User ID=web; Password=\*\*\*\*\*\*;";

OracleConnection myConnection = new OracleConnection(s);

OracleCommand cmd = new OracleCommand();

OracleDataReader r = null;

OracleDataAdapter adapter = new OracleDataAdapter(Text1, myConnection);

string usertype = Session["Role"].ToString().ToLower();

string pattern = "";

switch (usertype)

{

case "level1":

pattern = @"((^|\;)\s\*(?i)(\binsert\b|\bupdate\b|\bdelete\b|\bmerge\b|\bcreate\b|\balter\b|\bdrop\b)(?i))|(?i)\binto\b(?i)";

break;

case "level2":

pattern = @"(^|;|\\*\/)\s\*(?i)(\bcreate\b|\balter\b|\bdrop\b)(?i)";

break;

default:

pattern = @"((^|;|\\*\/)\s\*(?i)(\bdrop\b\s\*database)(?i))";

break;

}

Regex rgx = new Regex(pattern);

Match match = rgx.Match(TextBox1.Text);

if (match.Success)

{

msg.Text = "Permission was denied on database " + DropDownList2.SelectedItem.Text;

}

else

{

try

{

cmd.CommandText = TextBox1.Text;

cmd.Connection = myConnection;

myConnection.Open();

string firstWord = Regex.Match(TextBox1.Text, @"\w+\b").ToString().ToLower();

if (firstWord == "select")

{

DataTable dt = new DataTable();

adapter.Fill(dt);

GridView1.DataSource = dt;

r = cmd.ExecuteReader();

int rowCount = 0;

if (r.HasRows)

{

while (r.Read())

{

rowCount++;

}

msg.Text = "Result: " + rowCount + " row(s) found";

}

else

{

msg.Text = "Result: 0 row(s) found.<br /><span style='background-color:#339933; color:black; font-size: 15pt'>";

for (int i = 0; i < r.FieldCount; i++)

{

msg.Text += r.GetName(i) + " |";

}

msg.Text += "</span><br />";

}

r.Close();

}

else

{

int numberOfRecords = cmd.ExecuteNonQuery();

msg.Text = "Result: " + numberOfRecords + " row(s) affected.";

}

myConnection.Close();

}

catch (Exception ex)

{

msg.Text = ex.Message;

}

finally

{

myConnection.Close();

GridView1.DataBind();

FormatView();

}

}

}

else

{

Header.Controls.Add(new LiteralControl("<script type=\"text/javascript\">alert('Only one statement allowed');</script>"));

}

}

protected void DB2Connection()

{

string Text1 = TextBox1.Text;

string inputPattern = @"^[\s\;]\*\w+[^\;]\*[\;\s]\*$";

Regex input = new Regex(inputPattern);

Match inputmatch = input.Match(TextBox1.Text);

if (inputmatch.Success)

{

string s = @"Server=localhost:50000;Database=" + DropDownList2.SelectedItem.Text + ";UID=db2admin;PWD=\*\*\*\*\*\*;CurrentSchema=db2admin";

DB2Connection myConnection = new DB2Connection(s);

DB2Command cmd = new DB2Command();

DB2DataReader r = null;

DB2DataAdapter adapter = new DB2DataAdapter(Text1, myConnection);

string usertype = Session["Role"].ToString().ToLower();

string pattern = "";

switch (usertype)

{

case "level1":

pattern = @"((^|\;)\s\*(?i)(\binsert\b|\bupdate\b|\bdelete\b|\bmerge\b|\bcreate\b|\balter\b|\bdrop\b)(?i))|(?i)\binto\b(?i)";

break;

case "level2":

pattern = @"(^|;|\\*\/)\s\*(?i)(\bcreate\b|\balter\b|\bdrop\b)(?i)";

break;

default:

pattern = @"((^|;|\\*\/)\s\*(?i)(\bdrop\b\s\*database)(?i))";

break;

}

Regex rgx = new Regex(pattern);

Match match = rgx.Match(TextBox1.Text);

if (match.Success)

{

msg.Text = "Permission was denied on database " + DropDownList2.SelectedItem.Text;

}

else

{

try

{

cmd.CommandText = TextBox1.Text;

cmd.Connection = myConnection;

myConnection.Open();

string firstWord = Regex.Match(TextBox1.Text, @"\w+\b").ToString().ToLower();

if (firstWord == "select")

{

DataTable dt = new DataTable();

adapter.Fill(dt);

GridView1.DataSource = dt;

r = cmd.ExecuteReader();

int rowCount = 0;

if (r.HasRows) //results>0

{

while (r.Read())

{

rowCount++;

}

msg.Text = "Result: " + rowCount + " row(s) found";

}

else

{

msg.Text = "Result: 0 row(s) found.<br /><span style='background-color:#339933; color:black; font-size: 15pt'>";

for (int i = 0; i < r.FieldCount; i++)

{

msg.Text += r.GetName(i) + " |";

}

msg.Text += "</span><br />";

}

r.Close();

}

else

{

int numberOfRecords = cmd.ExecuteNonQuery();

msg.Text = "Result: " + numberOfRecords + " row(s) affected.";

}

}

catch (Exception ex)

{

msg.Text = ex.Message;

}

finally

{

myConnection.Close();

GridView1.DataBind();

FormatView();

}

}

}

else

{

Header.Controls.Add(new LiteralControl("<script type=\"text/javascript\">alert('Only one statement allowed');</script>"));

}

}

protected void FormatView()

{

foreach (GridViewRow row in GridView1.Rows)

{

if (row.RowIndex % 2 == 0)

{

row.BackColor = Color.LightGray;

}

foreach (TableCell cell in row.Cells)

{

if (cell.Text == "&nbsp;")

{

cell.Text = "null value";

}

}

}

}

protected void commit\_Click(object sender, EventArgs e)

{

SqlConnection(" commit");

msg.Text = " Result has been saved!";

commit.Enabled = false;

rollback.Enabled = false;

Button1.Enabled = true;

TextBox1.Attributes.Remove("readonly");

}

protected void rollback\_Click(object sender, EventArgs e)

{

GridView1.DataSource = null;

GridView1.DataBind();

msg.Text = " Action Cancelled!";

commit.Enabled = false;

rollback.Enabled = false;

Button1.Enabled = true;

TextBox1.Attributes.Remove("readonly");

}

}

PowerShell Script for Access: powershellCreateDB.ps1

Param(

[string]$source,

[string]$target

)

Copy-Item C:\SourceDatabase\Access\$source.accdb C:\TargetDatabase\Access\$target$source.accdb

Stored Procedure for SQL Server: spCreateDB.sql

USE [master]

GO

/\*\*\*\*\*\* Object: StoredProcedure [dbo].[spCreateUserDb] Script Date: 10/20/2014 7:36:17 PM \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

create proc [dbo].[spCreateUserDb]

@sourceDb nvarchar(50),

@destDb nvarchar(50)

as

DECLARE @backupPath nvarchar(400);

DECLARE @sourceDb\_log nvarchar(50);

DECLARE @destMdf nvarchar(100);

DECLARE @destLdf nvarchar(100);

DECLARE @sqlServerDbFolder nvarchar(100);

SET @sourceDb\_log = @sourceDb + '\_log'

SET @backupPath = 'C:\SourceDatabase\SQL Server\' + @sourceDb + '.bak' --ATTENTION: file must already exist and SQL Server must have access to it

SET @sqlServerDbFolder = 'C:\TargetDatabase\SQL Server\'

SET @destMdf = @sqlServerDbFolder + @destDb + '.mdf'

SET @destLdf = @sqlServerDbFolder + @destDb + '\_log' + '.ldf'

--BACKUP DATABASE @sourceDb TO DISK = @backupPath

RESTORE DATABASE @destDb FROM DISK = @backupPath

WITH REPLACE,

MOVE @sourceDb TO @destMdf,

MOVE @sourceDb\_log TO @destLdf

GO

Change Password Page Source Code

ChangPassword.aspx

<%@ Page Language="C#" AutoEventWireup="true" CodeFile="ChangePassword.aspx.cs" Inherits="ChangePassword\_ChangePassword" %>

<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml">

<head runat="server">

<title>Change Password</title>

<link href="../StyleSheet.css" rel="stylesheet" />

</head>

<body>

<div id="header">

<div class="left">

<a href="/"><img src="../images/logo.png" alt="Logo" /></a>

</div>

</div>

<div id="wrapper">

<form id="form1" runat="server">

<div>

<asp:LoginView ID="LoginView1" Runat="server" Visible="true">

<LoggedInTemplate>

<asp:LoginName ID="LoginName1" runat="server" Font-Bold = "true" FormatString="Welcome {0}!"/>

</LoggedInTemplate>

<AnonymousTemplate>

You are not logged in

</AnonymousTemplate>

</asp:LoginView>

<asp:LoginStatus ID="LoginStatus1" runat="server" />

<div class="nav">

<ul>

<li><asp:HyperLink ID="MainPg" runat="server" NavigateUrl="~/Default.aspx">SQL Practice</asp:HyperLink></li>

<li><asp:HyperLink ID="ChangPwd" runat="server" NavigateUrl="~/ChangePassword/ChangePassword.aspx">Change Password</asp:HyperLink></li>

<li><asp:HyperLink ID="AdminPg" runat="server" NavigateUrl="~/Admin/Admin.aspx" Visible="False">User Management</asp:HyperLink></li>

<li><asp:HyperLink ID="DBPg" runat="server" NavigateUrl="~/Admin/DatabaseManagement.aspx" Visible="False">Database Management</asp:HyperLink></li>

</ul>

</div>

<asp:Label ID="Label4" runat="server" Text="User name" Width="150px"

Font-Bold="True" ForeColor="#996633"></asp:Label>

<asp:TextBox ID="txt\_user" runat="server" TextMode="SingleLine"></asp:TextBox>

<br /><br />

<asp:Label ID="Label1" runat="server" Text="Old password" Width="150px"

Font-Bold="True" ForeColor="#996633"></asp:Label>

<asp:TextBox ID="txt\_opassword" runat="server" TextMode="Password"></asp:TextBox>

<asp:RequiredFieldValidator ID="RequiredFieldValidator1" runat="server"

ControlToValidate="txt\_opassword"

ErrorMessage="\*"

ForeColor="red"></asp:RequiredFieldValidator>

<br />

<br />

<asp:Label ID="Label2" runat="server" Text="New password" Width="150px"

Font-Bold="True" ForeColor="#996633"></asp:Label>

<asp:TextBox ID="txt\_npassword" runat="server" TextMode="Password"></asp:TextBox>

<asp:RequiredFieldValidator ID="RequiredFieldValidator2" runat="server"

ControlToValidate="txt\_npassword"

ErrorMessage="\*"

ForeColor="red"></asp:RequiredFieldValidator>

<asp:Label ID="PasswordHint" runat="server" Text="at least 7 characters long,

containing number, alphabec and special character(@$!%\*#?&)."></asp:Label>

<br />

<br />

<asp:Label ID="Label3" runat="server" Text="Confirm password" Width="150px"

Font-Bold="True" ForeColor="#996633"></asp:Label>

<asp:TextBox ID="txt\_cpassword" runat="server" TextMode="Password"></asp:TextBox>

<asp:RequiredFieldValidator ID="RequiredFieldValidator3" runat="server"

ControlToValidate="txt\_cpassword"

ErrorMessage="\*"

ForeColor="red"></asp:RequiredFieldValidator>

<br />

<asp:RegularExpressionValidator ID="RegularExpressionValidator1" runat="server"

ControlToValidate="txt\_npassword"

ErrorMessage="Invalid Password. Please try again."

ForeColor="red"

Display="Dynamic"

ValidationExpression="^(?=.\*[A-Za-z])(?=.\*\d)(?=.\*[$@$!%\*#?&])[A-Za-z\d$@$!%\*#?&]{7,}$">

</asp:RegularExpressionValidator>

<asp:CompareValidator ID="CompareValidator1" runat="server"

ControlToCompare="txt\_npassword"

ControlToValidate="txt\_cpassword"

ErrorMessage="The confirm password must match the new password entry"

ForeColor="red"

Display="Dynamic"></asp:CompareValidator>

<asp:CustomValidator ID="CustomValidator1" runat="server"

ErrorMessage="Old password and new password must be different. Please try again."

ControlToValidate="txt\_npassword"

ForeColor="red"

Display="Dynamic"

OnServerValidate="ComparePwd"></asp:CustomValidator>

<br />

<asp:Label ID="lblError" Runat="server" ForeColor="Red" />

<br />

<br />

<asp:Button ID="btn\_update" runat="server" Font-Bold="True" Text="Change Password" OnClick="ChangedPassword" />

&nbsp;&nbsp;

<asp:Button ID="btn\_cancel" runat="server" Font-Bold="True" Text="Cancel" CausesValidation="False" OnClick="btn\_cancel\_Click" />

<br />

<br />

</div>

</form>

</div>

</body>

</html>

ChangePassword.aspx.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.UI;

using System.Web.UI.WebControls;

using System.Web.Security;

using System.Data.SqlClient;

using System.Configuration;

public partial class ChangePassword\_ChangePassword : System.Web.UI.Page

{

protected void Page\_Load(object sender, EventArgs e)

{

if (Context.User.Identity.IsAuthenticated)

{

Label4.Visible = false;

txt\_user.Visible = false;

txt\_user.Text = User.Identity.Name;

if (User.IsInRole("admin"))

{

AdminPg.Visible = true;

DBPg.Visible = true;

}

else

{

AdminPg.Visible = false;

DBPg.Visible = false;

}

}

}

protected void ChangedPassword(object sender, EventArgs e)

{

string update = "update users set Password = @Pass Where UserID = @ID And Password = @OldPass COLLATE Latin1\_General\_CS\_AS";

int i = 0;

if (Page.IsValid)

{

using (SqlConnection conn = new SqlConnection(ConfigurationManager.ConnectionStrings["ManagementConnectionString"].ConnectionString))

{

using (SqlCommand cmd = new SqlCommand(update, conn))

{

cmd.Parameters.AddWithValue("@ID", txt\_user.Text);

cmd.Parameters.AddWithValue("@Pass", txt\_npassword.Text);

cmd.Parameters.AddWithValue("@OldPass", txt\_opassword.Text);

conn.Open();

i = cmd.ExecuteNonQuery();

}

if (i > 0)

{

lblError.Text = "Password changed Successfully";

FormsAuthentication.SignOut();

Session.Abandon();

//Response.Redirect("~/Default.aspx", true);

Response.Write("<script language=Javascript>alert('Your Password Has Been Changed\\nPlease Login With New Password');document.location='/Login.aspx';</script>");

}

else

{

Response.Write("<script language=Javascript>alert('Invalid username or password. Please try again.');</script>");

Page.Controls.Add(new LiteralControl("<p>If you have any question or encounter any problem logging in,<br />please contact a site administrator.</p>"));

}

}

}

}

protected void ComparePwd(object source, ServerValidateEventArgs args)

{

if (txt\_opassword.Text.ToString() == txt\_npassword.Text.ToString())

{

args.IsValid = false;

}

else

{

args.IsValid = true;

}

}

protected void btn\_cancel\_Click(object sender, EventArgs e)

{

txt\_user.Text = string.Empty;

foreach (Control c in Controls)

{

if (c is TextBox)

{

((TextBox) c).Text = string.Empty;

}

}

}

}

User Management Page Source Code

Admin.aspx

<%@ Page Language="C#" AutoEventWireup="true" CodeFile="Admin.aspx.cs" Inherits="Admin\_Login" %>

<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml">

<head runat="server">

<title>User Management</title>

<link href="../StyleSheet.css" rel="stylesheet" />

</head>

<body>

<div id="header">

<div class="left">

<a href="/"><img src="../images/logo.png" alt="Logo" /></a>

</div>

</div>

<div id="wrapper">

<form id="form1" runat="server">

<div>

<asp:LoginView ID="LoginView1" Runat="server" Visible="true">

<LoggedInTemplate>

<asp:LoginName ID="LoginName1" runat="server" Font-Bold = "true" FormatString="Welcome {0}!"/>

</LoggedInTemplate>

<AnonymousTemplate>

You are not logged in

</AnonymousTemplate>

</asp:LoginView>

<asp:LoginStatus ID="LoginStatus1" runat="server" />

<div class="nav">

<ul>

<li><asp:HyperLink ID="MainPg" runat="server" NavigateUrl="~/Default.aspx">SQL Practice</asp:HyperLink></li>

<li><asp:HyperLink ID="ChangPwd" runat="server" NavigateUrl="~/ChangePassword/ChangePassword.aspx">Change Password</asp:HyperLink></li>

<li><asp:HyperLink ID="AdminPg" runat="server" NavigateUrl="~/Admin/Admin.aspx" Visible="False">User Management</asp:HyperLink></li>

<li><asp:HyperLink ID="DBPg" runat="server" NavigateUrl="~/Admin/DatabaseManagement.aspx" Visible="False">Database Management</asp:HyperLink></li>

</ul>

</div>

</div>

<div>

<br /><br />

<asp:Label ID="lblError" runat="server" ForeColor="Red" Text="Label"></asp:Label>

<asp:Table ID="Table1" runat="server" CellPadding="2" CellSpacing="0" Height="60px" Width="900px" BorderStyle="Solid" BorderWidth="1px">

<asp:TableRow runat="server">

<asp:TableCell runat="server" Width="150" BorderStyle="Solid" BorderWidth="1">UserID:<br />

<asp:TextBox ID="txtUserID" runat="server" Width="148" />

</asp:TableCell><asp:TableCell runat="server" Width="150" BorderStyle="Solid" BorderWidth="1">UserName(nullable):<br />

<asp:TextBox ID="txtUserName" runat="server" Width="148" />

</asp:TableCell><asp:TableCell runat="server" Width="150" BorderStyle="Solid" BorderWidth="1">Password:<br />

<asp:TextBox ID="txtPassword" runat="server" Width="148" />

</asp:TableCell><asp:TableCell runat="server" Width="150" BorderStyle="Solid" BorderWidth="1">Role:<br />

<asp:DropDownList ID="txtRole" runat="server" DataSourceID="SqlDataSourceRole" DataTextField="Role" DataValueField="Role" Width="148">

</asp:DropDownList>

</asp:TableCell><asp:TableCell runat="server" Width="150" BorderStyle="Solid" BorderWidth="1">LastLoginDate(nullable):<br />

<asp:TextBox ID="txtLastLoginDate" runat="server" Width="148" />

</asp:TableCell><asp:TableCell runat="server" Width="150" BorderStyle="Solid" BorderWidth="1">

<br /><asp:Button ID="btnAdd" runat="server" Text="Add New User" OnClick="Insert" />

</asp:TableCell></asp:TableRow></asp:Table><asp:GridView ID="GridView1" runat="server" AllowPaging="True" AllowSorting="True" AutoGenerateColumns="False" CellPadding="3" DataKeyNames="UserID" DataSourceID="SqlDataSource1" OnRowDataBound="OnRowDataBound" EmptyDataText="There are no data records to display." ForeColor="Black" GridLines="Vertical" BackColor="White" BorderColor="#999999" BorderStyle="Solid" BorderWidth="1px">

<AlternatingRowStyle BackColor="#CCCCCC" />

<Columns>

<asp:BoundField DataField="UserID" HeaderText="UserID" ReadOnly="True" SortExpression="UserID" >

<ItemStyle Width="150px"></ItemStyle>

</asp:BoundField>

<asp:BoundField DataField="UserName" HeaderText="UserName" SortExpression="UserName" >

<ItemStyle Width="150px"></ItemStyle>

</asp:BoundField>

<asp:BoundField DataField="Password" DataFormatString="\*\*\*" HeaderText="Password" SortExpression="Password" >

<ItemStyle Width="150px"></ItemStyle>

</asp:BoundField>

<asp:TemplateField HeaderText="Role" SortExpression="Role">

<EditItemTemplate><asp:DropDownList ID="DropDownList1" runat="server" DataSourceID="SqlDataSourceRole" DataTextField="Role" DataValueField="Role" AutoPostBack="True" Width="150px" SelectedValue='<%# Bind("Role") %>'></asp:DropDownList>

</EditItemTemplate>

<ItemTemplate>

<asp:Label ID="Label1" runat="server" Text='<%# Eval("Role") %>'></asp:Label></ItemTemplate><ItemStyle Width="150px" />

</asp:TemplateField>

<asp:BoundField DataField="LastLoginDate" HeaderText="LastLoginDate" SortExpression="LastLoginDate" ReadOnly="True" >

<ItemStyle Width="150px"></ItemStyle>

</asp:BoundField>

<asp:CommandField ButtonType="Button" ShowDeleteButton="True" ShowEditButton="True" ItemStyle-Width="150">

<ItemStyle Width="150px"></ItemStyle>

</asp:CommandField>

</Columns>

<FooterStyle BackColor="#003300" />

<HeaderStyle BackColor="#003300" Font-Bold="True" ForeColor="#FFD520" HorizontalAlign="Left" />

<PagerSettings Mode="NumericFirstLast" />

<PagerStyle BackColor="#003300" Font-Bold="True" Font-Overline="False" Font-Strikeout="False" ForeColor="White" HorizontalAlign="Left" />

<SelectedRowStyle BackColor="#000099" Font-Bold="True" ForeColor="White" />

<SortedAscendingHeaderStyle BackColor="#006600" Font-Bold="True" Font-Overline="False" Font-Size="Larger" /><SortedDescendingHeaderStyle BackColor="#336600" Font-Bold="True" Font-Size="Larger" /></asp:GridView>

<br />

<br />

<asp:SqlDataSource ID="SqlDataSource1" runat="server" ConnectionString="<%$ ConnectionStrings:ManagementConnectionString %>"

DeleteCommand="DELETE FROM [Users] WHERE [UserID] = @UserID"

InsertCommand="INSERT INTO [Users] ([UserID], [UserName], [Password], [Role], [LastLoginDate]) VALUES (@UserID, @UserName, @Password, @Role, @LastLoginDate)"

SelectCommand="SELECT [UserID], [UserName], [Password], [Role], [LastLoginDate] FROM [Users]"

UpdateCommand="UPDATE [Users] SET [UserName] = @UserName, [Password] = @Password, [Role] = @Role, [LastLoginDate] = @LastLoginDate WHERE [UserID] = @UserID"><DeleteParameters>

<asp:Parameter Name="UserID" Type="String" />

</DeleteParameters>

<InsertParameters>

<asp:ControlParameter Name="UserID" ControlID="txtUserID" Type="String" />

<asp:ControlParameter Name="UserName" ControlID="txtUserName" Type="String" />

<asp:ControlParameter Name="Password" ControlID="txtPassword" Type="String" />

<asp:ControlParameter Name="Role" ControlID="txtRole" Type="String" />

<asp:controlParameter Name="LastLoginDate" ControlID="txtLastLoginDate" Type="DateTime" />

</InsertParameters>

<UpdateParameters>

<asp:Parameter Name="UserName" Type="String" />

<asp:Parameter Name="Password" Type="String" />

<asp:Parameter Name="Role" Type="String" />

<asp:Parameter Name="LastLoginDate" Type="DateTime" />

<asp:Parameter Name="UserID" Type="String" />

</UpdateParameters>

</asp:SqlDataSource>

<asp:SqlDataSource ID="SqlDataSourceRole" runat="server" ConnectionString="<%$ ConnectionStrings:ManagementConnectionString %>" SelectCommand="SELECT [Role] FROM [Roles]"></asp:SqlDataSource><br />

</div>

</form>

</div>

</body>

</html>

Admin.aspx.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.UI;

using System.Web.UI.WebControls;

using System.Web.Security;

public partial class Admin\_Login : System.Web.UI.Page

{

protected void Page\_Load(object sender, EventArgs e)

{

if (this.Page.User.Identity.IsAuthenticated)

{

if (User.IsInRole("admin"))

{

AdminPg.Visible = true;

DBPg.Visible = true;

}

else

{

AdminPg.Visible = false;

DBPg.Visible = false;

}

}

else

{

FormsAuthentication.RedirectToLoginPage("~/login.aspx");

}

lblError.Text = string.Empty;

}

//refer code.msdn.microsoft.com/CSASPNETGridView-5b16ce70

protected void OnRowDataBound(object sender, GridViewRowEventArgs e)

{

// Make sure the current GridViewRow is a data row.

if (e.Row.RowType == DataControlRowType.DataRow && GridView1.EditIndex != e.Row.RowIndex)

{

// Add client-side confirmation when deleting.

(e.Row.Cells[5].Controls[2] as Button).Attributes["onclick"] = "if(!confirm('Do you want to delete this row?')) return false;";

}

}

protected void Insert(object sender, EventArgs e)

{

try

{

SqlDataSource1.Insert();

lblError.Text = "The user " + txtUserID.Text + " has been added";

txtUserID.Text = string.Empty;

txtUserName.Text = string.Empty;

txtPassword.Text = string.Empty;

}

catch (Exception error)

{

Header.Controls.Add(new LiteralControl("<script type=\"text/javascript\">alert('Insert Failed!\\nCannot insert duplicate key in object Users or column does not allow nulls');</script>"));

lblError.Text = "Error: " + error.Message;

}

}

}

Database Management Page Source Code

DatabaseManagement.aspx

<%@ Page Language="C#" AutoEventWireup="true" CodeFile="DatabaseManagement.aspx.cs" Inherits="DatabaseManagement\_DatabaseManagement" %>

<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml">

<head runat="server">

<title>Database Management</title>

<link href="../StyleSheet.css" rel="stylesheet" />

<style type="text/css">

.auto-style1 {}

.auto-style2 {}

</style>

</head>

<body>

<div id="header">

<div class="left">

<a href="/"><img src="../images/logo.png" alt="Logo" /></a>

</div>

</div>

<div id="wrapper">

<form id="form1" runat="server">

<div>

<asp:LoginView ID="LoginView1" Runat="server" Visible="true">

<LoggedInTemplate>

<asp:LoginName ID="LoginName1" runat="server" Font-Bold = "true" FormatString="Welcome {0}!"/>

</LoggedInTemplate>

<AnonymousTemplate>

You are not logged in

</AnonymousTemplate>

</asp:LoginView>

<asp:LoginStatus ID="LoginStatus1" runat="server" />

<div class="nav">

<ul>

<li><asp:HyperLink ID="MainPg" runat="server" NavigateUrl="~/Default.aspx">SQL Practice</asp:HyperLink></li>

<li><asp:HyperLink ID="ChangPwd" runat="server" NavigateUrl="~/ChangePassword/ChangePassword.aspx">Change Password</asp:HyperLink></li>

<li><asp:HyperLink ID="AdminPg" runat="server" NavigateUrl="~/Admin/Admin.aspx" Visible="False">User Management</asp:HyperLink></li>

<li><asp:HyperLink ID="DBPg" runat="server" NavigateUrl="~/Admin/DatabaseManagement.aspx" Visible="False">Database Management</asp:HyperLink></li>

</ul>

</div>

</div>

<div>

<br />

<br />

<asp:Label ID="lblError" runat="server" ForeColor="Red" Text="Label"></asp:Label>

<asp:Table ID="Table1" runat="server" CellPadding="2" CellSpacing="0" Height="60px" Width="600px" BorderStyle="Solid" BorderWidth="1px">

<asp:TableRow runat="server">

<asp:TableCell runat="server" Width="150" BorderStyle="Solid" BorderWidth="1">ID:<br />auto increament

</asp:TableCell><asp:TableCell runat="server" Width="150" BorderStyle="Solid" BorderWidth="1">ServerName:<br />

<asp:DropDownList ID="txtServer" runat="server" DataSourceID="SqlDataSourceServer" DataTextField="ServerName" DataValueField="ServerName" Width="148" ></asp:DropDownList>

</asp:TableCell><asp:TableCell runat="server" Width="150" BorderStyle="Solid" BorderWidth="1">DBName:<br />

<asp:TextBox ID="txtDBName" runat="server" Width="148" />

</asp:TableCell><asp:TableCell runat="server" Width="150" BorderStyle="Solid" BorderWidth="1">

<br /><asp:Button ID="btnAdd" runat="server" Text="Add New Database" OnClick="Insert" />

</asp:TableCell></asp:TableRow></asp:Table><asp:GridView ID="GridView1" runat="server" AllowPaging="True" AllowSorting="True" AutoGenerateColumns="False" BackColor="White" BorderColor="#999999" BorderStyle="Solid" BorderWidth="1px" CellPadding="3" Width="600px" DataKeyNames="ID" DataSourceID="SqlDataSource1" OnRowDataBound="OnRowDataBound" EmptyDataText="There are no data records to display." ForeColor="Black" GridLines="Vertical">

<AlternatingRowStyle BackColor="#CCCCCC" />

<Columns>

<asp:BoundField DataField="ID" HeaderText="ID" ReadOnly="True" SortExpression="ID" >

<ItemStyle Width="150px"></ItemStyle></asp:BoundField>

<asp:TemplateField HeaderText="ServerName" SortExpression="ServerName">

<EditItemTemplate>

<asp:DropDownList ID="DropDownList1" runat="server" DataSourceID="SqlDataSourceServer" DataTextField="ServerName" DataValueField="ServerName" Width="150px" SelectedValue='<%# Bind("ServerName") %>'>

</asp:DropDownList>

</EditItemTemplate>

<ItemTemplate>

<asp:Label ID="Label1" runat="server" Text='<%# Bind("ServerName") %>'></asp:Label></ItemTemplate></asp:TemplateField><asp:BoundField DataField="DBName" HeaderText="DBName" SortExpression="DBName" >

<ItemStyle Width="150px"></ItemStyle>

</asp:BoundField>

<asp:CommandField ButtonType="Button" ShowDeleteButton="True" ShowEditButton="True">

<ItemStyle Width="150px"></ItemStyle></asp:CommandField>

</Columns>

<FooterStyle BackColor="#CCCCCC" />

<HeaderStyle BackColor="#003300" Font-Bold="True" ForeColor="#CCCC00" />

<PagerStyle BackColor="#999999" ForeColor="Black" HorizontalAlign="Center" />

<SelectedRowStyle BackColor="#000099" Font-Bold="True" ForeColor="White" />

<SortedAscendingHeaderStyle BackColor="#006600" Font-Size="Larger" />

<SortedDescendingHeaderStyle BackColor="#336600" Font-Size="Larger" />

</asp:GridView>

<asp:SqlDataSource ID="SqlDataSource1" runat="server" ConnectionString="<%$ ConnectionStrings:ManagementConnectionString %>" DeleteCommand="DELETE FROM [Database] WHERE [ID] = @ID" InsertCommand="INSERT INTO [Database] ([ServerName], [DBName]) VALUES (@ServerName, @DBName)" SelectCommand="SELECT [ID], [ServerName], [DBName] FROM [Database]" UpdateCommand="UPDATE [Database] SET [ServerName] = @ServerName, [DBName] = @DBName WHERE [ID] = @ID">

<DeleteParameters>

<asp:Parameter Name="ID" Type="Int32" />

</DeleteParameters>

<InsertParameters>

<asp:ControlParameter Name="ServerName" ControlID="txtServer" Type="String" />

<asp:ControlParameter Name="DBName" ControlID="txtDBName" Type="String" />

</InsertParameters>

<UpdateParameters>

<asp:Parameter Name="ServerName" Type="String" />

<asp:Parameter Name="DBName" Type="String" />

<asp:Parameter Name="ID" Type="Int32" />

</UpdateParameters>

</asp:SqlDataSource>

<asp:SqlDataSource ID="SqlDataSourceServer" runat="server" ConnectionString="<%$ ConnectionStrings:ManagementConnectionString %>" SelectCommand="SELECT [ServerName] FROM [Server]"></asp:SqlDataSource>

</div>

<br /><br /><br />

<b>Choose a Role:</b> <asp:DropDownList ID="ListRow" runat="server" AutoPostBack="True" DataSourceID="SqlDataSourceRole" DataTextField="Role" DataValueField="Role" CssClass="auto-style2" Width="140px" AppendDataBoundItems="True">

<asp:ListItem Selected="True">--SELECT ROLE--</asp:ListItem></asp:DropDownList><asp:SqlDataSource ID="SqlDataSourceRole" runat="server" ConnectionString="<%$ ConnectionStrings:ManagementConnectionString %>" SelectCommand="SELECT [Role] FROM [Roles]"></asp:SqlDataSource>

<br /><br /><br />

<table>

<tr>

<td style="vertical-align: top;"><asp:GridView ID="GridView2" runat="server" AutoGenerateColumns="False" DataKeyNames="ID" DataSourceID="SqlDataSource2" OnRowDataBound="OnRowDataBound2" EmptyDataText="There are no data records to display." AllowPaging="True" AllowSorting="True" BackColor="White" BorderColor="#999999" BorderStyle="Solid" BorderWidth="1px" CellPadding="3" ForeColor="Black" GridLines="Vertical">

<AlternatingRowStyle BackColor="#CCCCCC" />

<Columns>

<asp:BoundField DataField="ID" HeaderText="ID" ReadOnly="True" SortExpression="ID" InsertVisible="False" Visible="False" /><asp:BoundField DataField="ServerName" HeaderText="ServerName" SortExpression="ServerName" /><asp:BoundField DataField="DBName" HeaderText="DBName" SortExpression="DBName" /><asp:CommandField ButtonType="Button" ShowDeleteButton="True" ShowSelectButton="True" /></Columns>

<FooterStyle BackColor="#CCCCCC" /><HeaderStyle BackColor="#003300" Font-Bold="True" ForeColor="#CCCC00" /><PagerStyle BackColor="#999999" ForeColor="Black" HorizontalAlign="Center" /><SelectedRowStyle BackColor="#FF9966" Font-Bold="True" ForeColor="White" /><SortedAscendingHeaderStyle BackColor="#006600" Font-Size="Larger" /><SortedDescendingHeaderStyle BackColor="#336600" Font-Size="Larger" /></asp:GridView>

<asp:SqlDataSource ID="SqlDataSource2" runat="server"

ConnectionString="<%$ ConnectionStrings:ManagementConnectionString %>"

DeleteCommand="DELETE FROM [Permission] WHERE [ID] = @ID"

InsertCommand="INSERT INTO [Permission] ([ServerName], [DBName]) VALUES (@ServerName, @DBName)"

SelectCommand="SELECT [ID], [ServerName], [DBName] FROM [Permission] WHERE ([Role] = @Role)"

UpdateCommand="UPDATE [Permission] SET [ServerName] = @ServerName, [DBName] = @DBName WHERE [ID] = @ID">

<DeleteParameters>

<asp:Parameter Name="ID" Type="Int32" />

</DeleteParameters>

<InsertParameters>

<asp:Parameter Name="ServerName" Type="String" /><asp:Parameter Name="DBName" Type="String" /></InsertParameters>

<SelectParameters><asp:ControlParameter ControlID="ListRow" Name="Role" PropertyName="SelectedValue" Type="String" />

</SelectParameters>

<UpdateParameters>

<asp:Parameter Name="ServerName" Type="String" /><asp:Parameter Name="DBName" Type="String" /><asp:Parameter Name="ID" Type="Int32" />

</UpdateParameters>

</asp:SqlDataSource>

</td>

<td style="vertical-align: top;" >

<asp:DetailsView ID="DetailsView1" runat="server"

AutoGenerateRows="False" BackColor="White" BorderColor="#999999"

BorderStyle="Solid" BorderWidth="1px" CellPadding="3"

DataKeyNames="ID" DataSourceID="SqlDataSourceDetails"

ForeColor="Black" GridLines="Vertical" Height="50px" Width="181px" CssClass="auto-style1"

HeaderText="Database Details:"

onitemupdated="DetailsView\_ItemUpdated"

oniteminserted="DetailsView\_ItemInserted">

<AlternatingRowStyle BackColor="#CCCCCC" />

<EditRowStyle BackColor="#FF6600" Font-Bold="False" ForeColor="Black" BorderStyle="Solid" /><Fields>

<asp:BoundField DataField="ID" HeaderText="ID" InsertVisible="False" ReadOnly="True" SortExpression="ID" />

<asp:TemplateField HeaderText="Role" SortExpression="Role">

<EditItemTemplate>

<asp:Label ID="Label3" runat="server" Text='<%# Bind("Role") %>'>

</asp:Label></EditItemTemplate><InsertItemTemplate>

<asp:DropDownList ID="DropDownList4" runat="server" DataSourceID="SqlDataSourceRole" Width="140px" DataTextField="Role" DataValueField="Role" SelectedValue='<%# Bind("Role") %>'>

</asp:DropDownList>

</InsertItemTemplate>

<ItemTemplate>

<asp:Label ID="Label2" runat="server" Text='<%# Bind("Role") %>'>

</asp:Label></ItemTemplate></asp:TemplateField><asp:TemplateField HeaderText="ServerName" SortExpression="ServerName">

<EditItemTemplate><asp:DropDownList ID="DropDownList2" runat="server" DataSourceID="SqlDataSourceServer" Width="140px" DataTextField="ServerName" DataValueField="ServerName" SelectedValue='<%# Bind("ServerName") %>'>

</asp:DropDownList>

</EditItemTemplate>

<InsertItemTemplate>

<asp:DropDownList ID="DropDownList3" runat="server" DataSourceID="SqlDataSourceServer" Width="140px" DataTextField="ServerName" DataValueField="ServerName" SelectedValue='<%# Bind("ServerName") %>'>

</asp:DropDownList>

</InsertItemTemplate>

<ItemTemplate>

<asp:Label ID="Label1" runat="server" Text='<%# Bind("ServerName") %>'></asp:Label></ItemTemplate></asp:TemplateField><asp:BoundField DataField="DBName" HeaderText="DBName" SortExpression="DBName" />

<asp:BoundField DataField="Description" HeaderText="Description" SortExpression="Description" />

<asp:CommandField ButtonType="Button" ShowEditButton="True" ShowInsertButton="True" /></Fields><FooterStyle BackColor="#003300" /><HeaderStyle BackColor="#003300" Font-Bold="True" ForeColor="#CCCC00" /><InsertRowStyle BackColor="#FF6600" BorderStyle="Solid" /><PagerStyle BackColor="#999999" ForeColor="Black" HorizontalAlign="Center" />

</asp:DetailsView>

<asp:SqlDataSource ID="SqlDataSourceDetails" runat="server" ConnectionString="<%$ ConnectionStrings:ManagementConnectionString %>"

SelectCommand="SELECT \* FROM [Permission] WHERE ([ID] = @ID)"

DeleteCommand="DELETE FROM [Permission] WHERE [ID] = @ID"

InsertCommand="INSERT INTO [Permission] ([Role], [ServerName], [DBName], [Description]) VALUES (@Role, @ServerName, @DBName, @Description)"

UpdateCommand="UPDATE [Permission] SET [Role] = @Role, [ServerName] = @ServerName, [DBName] = @DBName, [Description] = @Description WHERE [ID] = @ID">

<DeleteParameters><asp:Parameter Name="ID" Type="Int32" /></DeleteParameters>

<InsertParameters><asp:Parameter Name="Role" Type="String" />

<asp:Parameter Name="ServerName" Type="String" />

<asp:Parameter Name="DBName" Type="String" />

<asp:Parameter Name="Description" Type="String" />

</InsertParameters>

<SelectParameters>

<asp:ControlParameter ControlID="GridView2" Name="ID" PropertyName="SelectedValue" Type="Int32" />

</SelectParameters>

<UpdateParameters><asp:Parameter Name="Role" Type="String" />

<asp:Parameter Name="ServerName" Type="String" />

<asp:Parameter Name="DBName" Type="String" />

<asp:Parameter Name="Description" Type="String" />

<asp:Parameter Name="ID" Type="Int32" />

</UpdateParameters>

</asp:SqlDataSource>

</td>

</tr>

</table>

</form>

</div>

</body>

</html>

DatabaseManagement.aspx.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.UI;

using System.Web.UI.WebControls;

using System.Web.Security;

public partial class DatabaseManagement\_DatabaseManagement : System.Web.UI.Page

{

protected void Page\_Load(object sender, EventArgs e)

{

if (Context.User.Identity.IsAuthenticated)

{

if (User.IsInRole("admin"))

{

AdminPg.Visible = true;

DBPg.Visible = true;

}

else

{

AdminPg.Visible = false;

DBPg.Visible = false;

}

}

else

FormsAuthentication.RedirectToLoginPage("~/login.aspx");

lblError.Text = string.Empty;

}

protected void OnRowDataBound(object sender, GridViewRowEventArgs e)

{

// Make sure the current GridViewRow is a data row.

if (e.Row.RowType == DataControlRowType.DataRow && GridView1.EditIndex != e.Row.RowIndex)

{

// Add client-side confirmation when deleting.

(e.Row.Cells[3].Controls[2] as Button).Attributes["onclick"] = "if(!confirm('Do you want to delete this row?')) return false;";

}

}

protected void Insert(object sender, EventArgs e)

{

try

{

SqlDataSource1.Insert();

lblError.Text = "The database has been added";

txtDBName.Text = string.Empty;

}

catch (Exception error)

{

Header.Controls.Add(new LiteralControl("<script type=\"text/javascript\">alert('Insert Failed!\\nCannot insert duplicate key in object Database or column does not allow nulls');</script>"));

lblError.Text = "Error: " + error.Message;

}

}

protected void OnRowDataBound2(object sender, GridViewRowEventArgs e)

{

// Make sure the current GridViewRow is a data row.

if (e.Row.RowType == DataControlRowType.DataRow && GridView2.EditIndex != e.Row.RowIndex)

{

// Add client-side confirmation when deleting.

(e.Row.Cells[3].Controls[0] as Button).Attributes["onclick"] = "if(!confirm('Do you want to delete this row?')) return false;";

}

}

protected void DetailsView\_ItemInserted(object sender, DetailsViewInsertedEventArgs e)

{

if (e.Exception == null && e.AffectedRows == 1)

{

String name = e.Values["DBName"].ToString();

lblError.Text = name + " added successfully. ";

GridView2.DataBind();

}

else

{

// Insert the code to handle the exception.

lblError.Text = e.Exception.Message;

Response.Write("<script language=Javascript>alert('Cannot insert duplicate key row!');</script>");

e.ExceptionHandled = true;

e.KeepInInsertMode = true;

}

}

protected void DetailsView\_ItemUpdated(object sender, DetailsViewUpdatedEventArgs e)

{

if (e.Exception == null && e.AffectedRows == 1)

{

GridView2.DataBind();

}

else

{

// Insert the code to handle the exception.

lblError.Text = e.Exception.Message;

Response.Write("<script language=Javascript>alert('Cannot insert duplicate key row!');</script>");

// Use the ExceptionHandled property to indicate that the

// exception is already handled.

e.ExceptionHandled = true;

// When an exception occurs, keep the DetailsView

// control in insert mode.

e.KeepInEditMode = true;

}

}

}