/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* 作 者：

\* 创建日期：2011-3-14

\* 描 述：学生德育评分管理系统数据库创建脚本

\* 修改日期：

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

--使用系统默认的数据库

use master

go

create database IPQ

go

use IPQ

go

--学生信息表StudentList

CREATE TABLE [dbo].[StudentList] (

[StudentId] [varchar] (8) NOT NULL Primary Key,

[StudentName] [varchar] (20) NOT NULL ,

[StudentPassword] [varchar] (20) NOT NULL ,

[StudentSex] [varchar] (4) NOT NULL ,

[DormitoryType] [varchar] (8) NOT NULL ,

[FloorId] [int] NOT NULL ,

[DormitoryId] [varchar] (10) NOT NULL

)

GO

--IPQ扣分管理表（SubtractionIPQList）

CREATE TABLE [dbo].[SubtractionIPQList] (

[SubtractionId] [int] IDENTITY(1,1) NOT NULL Primary Key,

[StudentId] [varchar] (8) NOT NULL ,

[SubtractionReason] [varchar] (25) NOT NULL ,

[SubtractionScore] [int] NOT NULL

)

GO

--IPQ增加管理表（AddIPQList）

CREATE TABLE [dbo].[AddIPQList] (

[AddId] [int] IDENTITY(1,1) NOT NULL Primary Key,

[StudentId] [varchar] (8) NOT NULL ,

[AddReason] [varchar] (50) NOT NULL ,

[AddScore] [int] NOT NULL

)

GO

--每周考勤信息表（AttendanceList）

CREATE TABLE [dbo].[AttendanceList] (

[AttendanceId] [int] IDENTITY(1,1) NOT NULL Primary Key,

[Week] [varchar] (50) NOT NULL ,

[StudentId] [varchar] (8) NOT NULL ,

[Semester] [int] NOT NULL ,

[StayAway] [int] NOT NULL ,

[DesertExercises] [int] NOT NULL

)

GO

--每周卫生检查表（HealthList）

CREATE TABLE [dbo].[HealthList] (

[HealthId] [int] IDENTITY(1,1) NOT NULL Primary Key,

[Week] [int] NOT NULL ,

[DormitoryType] [varchar] (8) NOT NULL ,

[FloorId] [int] NOT NULL ,

[DormitoryId] [varchar] (10) NOT NULL ,

[HealthScore] [int] NOT NULL ,

[HealthGrade] [varchar] (4) NOT NULL

)

GO

--账户管理表（UserList）

CREATE TABLE [dbo].[UserList] (

[UserName] [varchar] (15) NOT NULL Primary Key,

[Password] [varchar] (25) NOT NULL ,

[Role] [varchar] (25) NOT NULL

)

GO

--学生处分信息表（PunishList）

CREATE TABLE [dbo].[PunishList] (

[PunishId] [int] IDENTITY(1,1) NOT NULL Primary Key,

[StudentId] [varchar] (8) NOT NULL ,

[PunishName] [varchar] (20) NOT NULL ,

[PunishReason] [varchar] (20) NOT NULL ,

[PunishDateTime] [datetime] NOT NULL

)

GO

--外键关联字段

--IPQ扣分管理表（SubtractionIPQList）

ALTER TABLE [dbo].[SubtractionIPQList] ADD

CONSTRAINT [SubtractionIPQList\_StudentList\_fk] FOREIGN KEY

(

[StudentId]

) REFERENCES [dbo].[StudentList] (

[StudentId]

)

GO

--IPQ增加管理表（AddIPQList）

ALTER TABLE [dbo].[AddIPQList] ADD

CONSTRAINT [AddIPQList\_StudentList\_fk] FOREIGN KEY

(

[StudentId]

) REFERENCES [dbo].[StudentList] (

[StudentId]

)

GO

--每周考勤信息表（AttendanceList）

ALTER TABLE [dbo].[AttendanceList] ADD

CONSTRAINT [AttendanceList\_StudentList\_fk] FOREIGN KEY

(

[StudentId]

) REFERENCES [dbo].[StudentList] (

[StudentId]

)

GO

--学生处分信息表（PunishList）

ALTER TABLE [dbo].[PunishList] ADD

CONSTRAINT [PunishList\_StudentList\_fk] FOREIGN KEY

(

[StudentId]

) REFERENCES [dbo].[StudentList] (

[StudentId]

)

GO

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* 作 者：

\* 创建日期：2011-3-15

\* 描 述：学生德育评分管理系统数据库添加数据脚本

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

USE master

GO

USE IPQ

GO

--学生信息表StudentList

INSERT INTO

[dbo].[StudentList]([StudentId],[StudentName],[StudentPassword],[StudentSex],[DormitoryType],[FloorId],[DormitoryId])

VALUES('20072236','邢成玉','xiner','男','公寓',4,'215')

GO

INSERT INTO

[dbo].[StudentList]([StudentId],[StudentName],[StudentPassword],[StudentSex],[DormitoryType],[FloorId],[DormitoryId])

VALUES('20072423','任志芳','xiner','女','公寓',5,'219')

GO

INSERT INTO

[dbo].[StudentList]([StudentId],[StudentName],[StudentPassword],[StudentSex],[DormitoryType],[FloorId],[DormitoryId])

VALUES('20071681','邢晶晶','xiner','女','普宿',7,'526')

GO

--账户管理表（UserList）

INSERT INTO

[dbo].[UserList]([UserName],[Password],[Role])

VALUES('xcy','xiner','admin')

GO

INSERT INTO

[dbo].[UserList]([UserName],[Password],[Role])

VALUES('xiner','xiner','manager')

GO

--每周卫生检查表（HealthList）

INSERT INTO

[dbo].[HealthList]([Week],[DormitoryType],[FloorId],[DormitoryId],[HealthScore],[HealthGrade])

VALUES(1,'公寓',4,'215',99,'A')

GO

INSERT INTO

[dbo].[HealthList]([Week],[DormitoryType],[FloorId],[DormitoryId],[HealthScore],[HealthGrade])

VALUES(1,'公寓',4,'216',88,'B')

GO

INSERT INTO

[dbo].[HealthList]([Week],[DormitoryType],[FloorId],[DormitoryId],[HealthScore],[HealthGrade])

VALUES(1,'公寓',4,'214',66,'D')

GO

INSERT INTO

[dbo].[HealthList]([Week],[DormitoryType],[FloorId],[DormitoryId],[HealthScore],[HealthGrade])

VALUES(2,'公寓',5,'217',99,'A')

GO

INSERT INTO

[dbo].[HealthList]([Week],[DormitoryType],[FloorId],[DormitoryId],[HealthScore],[HealthGrade])

VALUES(1,'公寓',5,'218',88,'B')

GO

INSERT INTO

[dbo].[HealthList]([Week],[DormitoryType],[FloorId],[DormitoryId],[HealthScore],[HealthGrade])

VALUES(1,'公寓',5,'219',66,'D')

GO

INSERT INTO

[dbo].[HealthList]([Week],[DormitoryType],[FloorId],[DormitoryId],[HealthScore],[HealthGrade])

VALUES(2,'普宿',7,'217',99,'A')

GO

INSERT INTO

[dbo].[HealthList]([Week],[DormitoryType],[FloorId],[DormitoryId],[HealthScore],[HealthGrade])

VALUES(1,'普宿',7,'218',88,'B')

GO

INSERT INTO

[dbo].[HealthList]([Week],[DormitoryType],[FloorId],[DormitoryId],[HealthScore],[HealthGrade])

VALUES(2,'普宿',7,'219',66,'D')

GO

--学生处分信息表（PunishList）

INSERT INTO

[dbo].[PunishList]([StudentId],[PunishName],[PunishReason],[PunishDateTime])

VALUES('20072236','留校察看','考试作弊','2011-3-3')

GO

INSERT INTO

[dbo].[PunishList]([StudentId],[PunishName],[PunishReason],[PunishDateTime])

VALUES('20072423','留校察看','考试作弊','2011-3-3')

GO

--每周考勤信息表（AttendanceList）

INSERT INTO

[dbo].[AttendanceList]([Week],[StudentId],[Semester],[StayAway],[DesertExercises])

VALUES(1,'20072236',1,0,1)

GO

INSERT INTO

[dbo].[AttendanceList]([Week],[StudentId],[Semester],[StayAway],[DesertExercises])

VALUES(1,'20072423',1,2,10)

GO

INSERT INTO

[dbo].[AttendanceList]([Week],[StudentId],[Semester],[StayAway],[DesertExercises])

VALUES(1,'20071681',1,0,0)

GO

INSERT INTO

[dbo].[AttendanceList]([Week],[StudentId],[Semester],[StayAway],[DesertExercises])

VALUES(2,'20071681',1,0,0)

GO

INSERT INTO

[dbo].[AttendanceList]([Week],[StudentId],[Semester],[StayAway],[DesertExercises])

VALUES(2,'20072236',1,0,1)

GO

INSERT INTO

[dbo].[AttendanceList]([Week],[StudentId],[Semester],[StayAway],[DesertExercises])

VALUES(2,'20072423',1,2,10)

GO

--IPQ扣分管理表（SubtractionIPQList）

INSERT INTO

[dbo].[SubtractionIPQList]([StudentId],[SubtractionReason],[SubtractionScore])

VALUES('20072236','思想作风不好',10)

GO

INSERT INTO

[dbo].[SubtractionIPQList]([StudentId],[SubtractionReason],[SubtractionScore])

VALUES('20072236','思想作风不好',10)

GO

INSERT INTO

[dbo].[SubtractionIPQList]([StudentId],[SubtractionReason],[SubtractionScore])

VALUES('20072236','思想作风不好',10)

GO

--IPQ增加管理表（AddIPQList）

INSERT INTO

[dbo].[AddIPQList]([StudentId],[AddReason],[AddScore])

VALUES('20072236','拾金不昧',5)

GO

INSERT INTO

[dbo].[AddIPQList]([StudentId],[AddReason],[AddScore])

VALUES('20072236','拾金不昧',5)

GO

INSERT INTO

[dbo].[AddIPQList]([StudentId],[AddReason],[AddScore])

VALUES('20072236','拾金不昧',5)

GO

INSERT INTO

[dbo].[AddIPQList]([StudentId],[AddReason],[AddScore])

VALUES('20071681','拾金不昧',5)

GO

INSERT INTO

[dbo].[AddIPQList]([StudentId],[AddReason],[AddScore])

VALUES('20072423','拾金不昧',5)

GO

select \* from [StudentList]

主要代码如下：

Imports Microsoft.VisualBasic

Imports System.Data

Imports System.Data.SqlClient

Imports System.Configuration

Imports System.Collections

Public Class SQLDatabase

Private \_conn As SqlConnection

Public Property conn() As SqlConnection

Get

Return \_conn

End Get

Set(ByVal value As SqlConnection)

\_conn = value

End Set

End Property

Private connectionstring As String

Sub New()

'下面这个是使用的服务器连接()

'connectionstring = ConfigurationManager.AppSettings("DBConnectionString")

'下面这个是使用的dypj.mdf连接()

connectionstring = ConfigurationManager.AppSettings("DBConnectionString2")

'下面这个是使用的服务器连接, windows验证

connectionstring = ConfigurationManager.AppSettings("DBConnectionString3")

End Sub

Protected Sub Open()

If conn Is Nothing = True Then

conn = New SqlConnection(connectionstring)

End If

If conn.State.Equals(ConnectionState.Closed) Then

conn.Open()

End If

End Sub

Protected Sub Close()

If conn Is Nothing = False Then

conn.Close()

End If

End Sub

Public Function ExecuteSQL(ByVal SQLString As String) As Boolean

Dim success As Boolean = True

Me.Open()

Try

Dim cmd As SqlCommand = New SqlCommand(SQLString, conn)

cmd.ExecuteNonQuery()

Catch ex As Exception

success = False

Finally

Me.Close()

End Try

Return success

End Function

Public Function ExecuteSQL(ByVal SqlStrings As ArrayList) As Boolean

Dim success As Boolean = True

Me.Open()

Dim cmd As SqlCommand = New SqlCommand

Dim trans As SqlTransaction = Me.conn.BeginTransaction

cmd.Connection = Me.conn

cmd.Transaction = trans

Try

For Each str As String In SqlStrings

cmd.CommandText = str

cmd.ExecuteNonQuery()

Next

trans.Commit()

Catch ex As Exception

success = False

trans.Rollback()

Finally

Me.Close()

End Try

Return success

End Function

Public Function GetDataReader(ByVal SqlString As String) As SqlDataReader

Me.Open()

Dim cmd As SqlCommand = New SqlCommand(SqlString, conn)

Return cmd.ExecuteReader(System.Data.CommandBehavior.CloseConnection)

End Function

Public Function Insert(ByVal TableName As String, ByVal Cols As Hashtable) As Boolean

Dim Count As Integer = 0

If Cols.Count <= 0 Then

Return True

End If

Dim Fields As String = " ("

Dim Values As String = " Values("

For Each Item As DictionaryEntry In Cols

If Count <> 0 Then

Fields += ","

Values += ","

End If

Fields += Item.Key.ToString

Values += Item.Value.ToString

Count += 1

Next

Fields += ")"

Values += ")"

Dim SqlString As String = "Insert into " + TableName + Fields + Values

Return Convert.ToBoolean(ExecuteSQL(SqlString))

End Function

Public Function Update(ByVal TableName As String, ByVal Cols As Hashtable, ByVal Where As String) As Boolean

Dim Count As Integer = 0

If Cols.Count <= 0 Then

Return True

End If

Dim Fields As String = " "

For Each Item As DictionaryEntry In Cols

If Count <> 0 Then

Fields += ","

End If

Fields += Item.Key.ToString

Fields += " = "

Fields += Item.Value.ToString

Count += 1

Next

Fields += " "

Dim SqlString As String = "Update " + TableName + " Set " + Fields + Where

Return Convert.ToBoolean(ExecuteSQL(SqlString))

End Function

Public Function GetDataSet(ByVal SQLString As String) As DataSet

Me.Open()

Dim sda As SqlDataAdapter = New SqlDataAdapter(SQLString, conn)

Dim ds As DataSet = New DataSet

sda.Fill(ds)

Me.Close()

Return ds

End Function

Public Function GetDataTable(ByVal SQLString As String) As DataTable

Me.Open()

Dim sda As SqlDataAdapter = New SqlDataAdapter(SQLString, conn)

Dim ds As DataSet = New DataSet

sda.Fill(ds)

Me.Close()

Return ds.Tables.Item(0)

End Function

Public Function GetDataRow(ByVal SQLstring As String) As DataRow

Dim ds As DataSet = Me.GetDataSet(SQLstring)

ds.CaseSensitive = False

If ds.Tables.Item(0).Rows.Count > 0 Then

Return ds.Tables.Item(0).Rows.Item(0)

Else

Return Nothing

End If

End Function

End Class

Imports Microsoft.VisualBasic

Imports System.Data

Imports System.Data.SqlClient

Public Class GetSafeData

Public Shared Function ValidateDataRow\_S(ByVal row As DataRow, ByVal colname As String) As String

If row(colname).Equals(DBNull.Value) = False Then

Return row(colname).ToString

Else

Return System.String.Empty

End If

End Function

Public Shared Function ValidateDataRow\_N(ByVal row As DataRow, ByVal colname As String) As Integer

If row(colname.ToString).Equals(DBNull.Value) = False Then

Return row(colname.ToString).ToString

Else

Return System.Int32.MinValue

End If

End Function

Public Shared Function ValidateDataRow\_F(ByVal row As DataRow, ByVal colname As String) As Double

If row(colname.ToString).Equals(DBNull.Value) = False Then

Return row(colname.ToString).ToString

Else

Return System.Double.MinValue

End If

End Function

Public Shared Function ValidateDataRow\_T(ByVal row As DataRow, ByVal colname As String) As Date

If row(colname.ToString).Equals(DBNull.Value) = False Then

Return row(colname.ToString).ToString

Else

Return System.DateTime.MinValue

End If

End Function

Public Shared Function ValidateDataReader\_S(ByVal reader As SqlDataReader, ByVal colname As String) As String

If reader.GetValue(reader.GetOrdinal(colname)).Equals(DBNull.Value) = False Then

Return reader.GetString(reader.GetOrdinal(colname))

Else

Return System.String.Empty

End If

End Function

Public Shared Function ValidateDataReader\_N(ByVal reader As SqlDataReader, ByVal colname As String) As String

If reader.GetValue(reader.GetOrdinal(colname.ToString)).Equals(DBNull.Value) = False Then

Return reader.GetString(reader.GetOrdinal(colname.ToString))

Else

Return System.Int32.MinValue.ToString

End If

End Function

Public Shared Function ValidateDataReader\_F(ByVal reader As SqlDataReader, ByVal colname As String) As String

If reader.GetValue(reader.GetOrdinal(colname)).Equals(DBNull.Value) = False Then

Return reader.GetString(reader.GetOrdinal(colname.ToString))

Else

Return System.Double.MinValue.ToString

End If

End Function

Public Shared Function ValidateDataReader\_T(ByVal reader As SqlDataReader, ByVal colname As String) As String

If reader.GetValue(reader.GetOrdinal(colname.ToString)).Equals(DBNull.Value) = False Then

Return reader.GetString(reader.GetOrdinal(colname.ToString))

Else

Return System.DateTime.MinValue

End If

End Function

End Class

Imports Microsoft.VisualBasic

Imports System.Data

Imports System.Data.SqlClient

Public Class SqlStringConstructor

Public Shared Function GetQuottedString(ByVal pstr As String) As String

Return ("'" + pstr.Replace("'", "''") + "'")

End Function

Public Shared Function GetConditionClause(ByVal queryItems As Hashtable) As String

Dim count As Integer = 0

Dim Where As String = ""

For Each Item As DictionaryEntry In queryItems

If count = 0 Then

Where = " Where "

Else

Where = " And "

End If

If Item.Value.GetType.ToString().Equals("System.String") = True OrElse Item.Value.GetType.ToString().Equals("System.DateTime") = True Then

Where += "[" + Item.Key.ToString + "]"

Where += " Like "

Where += SqlStringConstructor.GetQuottedString("%" + Item.Value.ToString + "%")

Else

Where += "[" + Item.Key.ToString + "]" + "=" + Item.Value.ToString

End If

count += 1

Next

Return Where

End Function

Public Shared Function GetAndConditionClause(ByVal queryItems As Hashtable, ByVal type As String) As String

Dim count As Integer = 0

Dim Where As String = ""

For Each Item As DictionaryEntry In queryItems

If count = 0 Then

Where = " Where "

Else

Where = " " + type + " "

End If

If Item.Value.GetType.ToString().Equals("System.String") = True OrElse Item.Value.GetType.ToString().Equals("System.DateTime") = True Then

Where += "[" + Item.Key.ToString + "]"

Where += +" Like "

Where += +SqlStringConstructor.GetQuottedString("%" + Item.Value.ToString + "%")

Else

Where += Item.Key.ToString + "=" + Item.Value.ToString

End If

count += 1

Next

Return Where

End Function

End Class

Imports Microsoft.VisualBasic

Imports System.Data

Imports System.Data.SqlClient

Public Class StudentLogic

Private \_StudentId As String

Public Property StudentId() As String

Get

Return \_StudentId

End Get

Set(ByVal value As String)

\_StudentId = value

End Set

End Property

Private \_StudentPassword As String

Public Property StudentPassword() As String

Get

Return \_StudentPassword

End Get

Set(ByVal value As String)

\_StudentPassword = value

End Set

End Property

Private \_StudentName As String

Public Property StudentName() As String

Get

Return \_StudentName

End Get

Set(ByVal value As String)

\_StudentName = value

End Set

End Property

Private \_exist As Boolean

Public Property Exist() As Boolean

Get

Return \_exist

End Get

Set(ByVal value As Boolean)

\_exist = value

End Set

End Property

Private \_StudentSex As String

Public Property StudentSex() As String

Get

Return \_StudentSex

End Get

Set(ByVal value As String)

\_StudentSex = value

End Set

End Property

Private \_DormitoryType As String

Public Property DormitoryType() As String

Get

Return \_DormitoryType

End Get

Set(ByVal value As String)

\_DormitoryType = value

End Set

End Property

Private \_FloorId As Integer

Public Property FloorId() As Integer

Get

Return \_FloorId

End Get

Set(ByVal value As Integer)

\_FloorId = value

End Set

End Property

Private \_DormitoryId As String

Public Property DormitoryId() As String

Get

Return \_DormitoryId

End Get

Set(ByVal value As String)

\_DormitoryId = value

End Set

End Property

Public Sub LoadData(ByVal StudentId As String)

Dim db As SQLDatabase = New SQLDatabase

Dim SQLString As String = "select \* from [StudentList] where StudentId=" + SqlStringConstructor.GetQuottedString(StudentId)

Dim dr As DataRow = db.GetDataRow(SQLString)

If dr Is Nothing = False Then

Me.\_StudentId = GetSafeData.ValidateDataRow\_S(dr, "StudentId")

Me.\_StudentName = GetSafeData.ValidateDataRow\_S(dr, "StudentName")

Me.\_StudentPassword = GetSafeData.ValidateDataRow\_S(dr, "StudentPassword")

Me.\_StudentSex = GetSafeData.ValidateDataRow\_S(dr, "StudentSex")

Me.\_DormitoryType = GetSafeData.ValidateDataRow\_S(dr, "DormitoryType")

Me.\_FloorId = GetSafeData.ValidateDataRow\_N(dr, "FloorId")

Me.\_DormitoryId = GetSafeData.ValidateDataRow\_S(dr, "DormitoryId")

Me.\_exist = True

Else

Me.\_exist = False

End If

End Sub

Public Shared Sub Add(ByVal userInfo As Hashtable)

Dim db As SQLDatabase = New SQLDatabase

db.Insert("[StudentList]", userInfo)

End Sub

Public Shared Function HasUser(ByVal StudentId As String) As Boolean

Dim db As SQLDatabase = New SQLDatabase

Dim SQLString As String = "select \* from [StudentList] where StudentId=" + SqlStringConstructor.GetQuottedString(StudentId)

Dim dr As DataRow = db.GetDataRow(SQLString)

If dr Is Nothing = False Then

Return True

Else

Return False

End If

End Function

Public Shared Sub Update(ByVal newUserInfo As Hashtable, ByVal strCond As String)

Dim db As SQLDatabase = New SQLDatabase

'strCond = "Where UserName=" + SqlStringConstructor.GetQuottedString(Me.\_StudentId)

db.Update("[StudentList]", newUserInfo, strCond)

End Sub

Public Shared Sub Delete(ByVal StudentId As String)

Dim sqls As ArrayList = New ArrayList

Dim SQLString As String

SQLString = "Delete from [SubtractionIPQList] where StudentId=" + SqlStringConstructor.GetQuottedString(StudentId)

sqls.Add(SQLString)

SQLString = "Delete from [AddIPQList] where StudentId=" + SqlStringConstructor.GetQuottedString(StudentId)

sqls.Add(SQLString)

SQLString = "Delete from [AttendanceList] where StudentId=" + SqlStringConstructor.GetQuottedString(StudentId)

sqls.Add(SQLString)

SQLString = "Delete from [PunishList] where StudentId=" + SqlStringConstructor.GetQuottedString(StudentId)

sqls.Add(SQLString)

SQLString = "Delete from [StudentList] where StudentId=" + SqlStringConstructor.GetQuottedString(StudentId)

sqls.Add(SQLString)

Dim db As SQLDatabase = New SQLDatabase

db.ExecuteSQL(sqls)

End Sub

Public Shared Function Query() As DataTable

Dim SQL As String = "select \* from [StudentList]"

Dim db As SQLDatabase = New SQLDatabase

Return db.GetDataTable(SQL)

End Function

Public Shared Function QueryUsers(ByVal queryItems As Hashtable) As DataTable

Dim where As String = SqlStringConstructor.GetConditionClause(queryItems)

Dim sql As String = "Select \* from [StudentList] " + where

Dim db As SQLDatabase = New SQLDatabase

Return db.GetDataTable(sql)

End Function

End Class

Imports System.Data

Imports System.Data.SqlClient

Partial Class StudentManager

Inherits System.Web.UI.Page

Protected Sub Page\_Load(ByVal sender As Object, ByVal e As System.EventArgs) Handles Me.Load

If Not Page.IsPostBack Then

QueryALL()

End If

Me.btnDelete.Attributes.Add("onclick", "javascript:return confirm('确定删除?');")

End Sub

Private Sub QueryALL()

Me.TextBox1.Text = ""

StudentManager.username = ""

Dim dt As DataTable = StudentLogic.Query

GV.DataSource = dt

GV.DataBind()

End Sub

Protected Sub btnUpdate\_Click(ByVal sender As Object, ByVal e As System.EventArgs) Handles btnUpdate.Click

Dim selectedUsers As ArrayList = GetSelected()

If selectedUsers.Count <> 1 Then

Response.Write("<script language=javascript>alert('请选择一个用户!');</script>")

Exit Sub

End If

Dim loginName As String = selectedUsers(0).ToString

Response.Redirect("StudentUpdate.aspx?login\_name=" + loginName)

End Sub

Private Function GetSelected() As ArrayList

Dim selectedItems As ArrayList = New ArrayList

For Each row As GridViewRow In GV.Rows

If CType(row.FindControl("chkSelected"), CheckBox).Checked = True Then

selectedItems.Add(Convert.ToString(row.Cells.Item(1).Text))

End If

Next

Return selectedItems

End Function

Protected Sub btnDelete\_Click(ByVal sender As Object, ByVal e As System.EventArgs) Handles btnDelete.Click

Dim selectedUsers As ArrayList = GetSelected()

For Each loginName As String In selectedUsers

StudentLogic.Delete(loginName)

Next

QueryALL()

End Sub

Private Sub Query()

Dim queryItems As Hashtable = New Hashtable

queryItems.Add("StudentId", StudentManager.username)

Dim dt As DataTable = StudentLogic.QueryUsers(queryItems)

GV.DataSource = dt

GV.DataBind()

End Sub

Shared username As String

Protected Sub btnQuery\_Click(ByVal sender As Object, ByVal e As System.EventArgs) Handles btnQuery.Click

StudentManager.username = Me.TextBox1.Text.Trim

Query()

Me.TextBox1.Text = ""

End Sub

Protected Sub Button1\_Click(ByVal sender As Object, ByVal e As System.EventArgs) Handles Button1.Click

QueryALL()

End Sub

Protected Sub GV\_PageIndexChanging(ByVal sender As Object, ByVal e As System.Web.UI.WebControls.GridViewPageEventArgs) Handles GV.PageIndexChanging

GV.PageIndex = e.NewPageIndex

If StudentManager.username = "" Then

QueryALL()

Else

Query()

End If

End Sub

End Class

Imports System.Data

Imports System.Data.SqlClient

Imports System.Configuration

Partial Class \_Default

Inherits System.Web.UI.Page

Protected Sub btnLogin\_Click(ByVal sender As Object, ByVal e As System.EventArgs) Handles btnLogin.Click

Dim userName As String = Me.txtUserName.Text.ToString().Trim()

Dim userPwd As String = Me.txtPwd.Text.ToString().Trim()

Dim userRole As String = Me.rblClass.SelectedValue.Trim()

Dim selectStr As String = ""

Select Case userRole

Case "1"

Dim user As StudentLogic = New StudentLogic

user.LoadData(userName)

Session.Add("student", userName)

If user.Exist Then

If user.StudentPassword.Equals(userPwd) Then

Response.Redirect("Main.aspx")

Else

Session.Clear()

Response.Write("<script language=javascript>alert('" + "验证失败，请重新登录!" + "')</script>")

End If

Else

Session.Clear()

Response.Write("<script language=javascript>alert('" + "对不起，用户不存在!" + "')</script>")

End If

Exit Select

Case "2"

Dim user As UserLogic = New UserLogic

user.LoadData(userName)

If user.Exist Then

If user.Password.Equals(userPwd) Then

If user.Role.Equals("admin") Then

Session.Add("admin", userName)

Response.Redirect("Main.aspx")

Else

Session.Add("manager", userName)

Response.Redirect("Main.aspx")

End If

Else

Session.Clear()

Response.Write("<script language=javascript>alert('" + "验证失败，请重新登录!" + "')</script>")

End If

Else

Session.Clear()

Response.Write("<script language=javascript>alert('" + "对不起，用户不存在!" + "')</script>")

End If

Case "3"

Dim user As UserLogic = New UserLogic

user.LoadData(userName)

If user.Exist Then

If user.Password.Equals(userPwd) Then

If user.Role.Equals("admin") Then

Session.Add("admin", userName)

Response.Redirect("Main.aspx")

Else

Session.Add("manager", userName)

Response.Redirect("Main.aspx")

End If

Else

Session.Clear()

Response.Write("<script language=javascript>alert('" + "验证失败，请重新登录!" + "')</script>")

End If

Else

Session.Clear()

Response.Write("<script language=javascript>alert('" + "对不起，用户不存在!" + "')</script>")

End If

Exit Select

End Select

End Sub

Protected Sub Page\_Load(ByVal sender As Object, ByVal e As System.EventArgs) Handles Me.Load

Session.Clear()

End Sub

Protected Sub btnClose\_Click(ByVal sender As Object, ByVal e As System.EventArgs) Handles btnClose.Click

Me.txtUserName.Text = ""

Me.txtPwd.Text = ""

End Sub

End Class