-----------------------------------------------------------------------------------------------------------------------------------------------------------------

***Generating connection string:***

-----------------------------------------------------------------------------------------------------------------------------------------------------------------

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;***

public partial class Default2 : System.Web.UI.Page

{

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

{

**string cs = "data source=.; database=adocon; integrated security=SSPI"; //key value pair**

**SqlConnection con = new SqlConnection(cs);**

**try**

**{**

**SqlCommand cmd = new SqlCommand("select \* from tblStudents", con);**

**con.Open();**

**GridView1.DataSource = cmd.ExecuteReader();**

**GridView1.DataBind();**

**}**

**catch**

**{**

**}**

**finally**

**{**

**con.Close(); //garrenties to execute, connection will be closed if any eception occurs**

**}**

}

}

-----------------------------------------------------------------------------------------------------------------------------------------------------------------

***Without closing the connection:***

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.OleDb;

using System.Data.Odbc;

public partial class Default2 : System.Web.UI.Page

{

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

{

string cs = "data source=.; database=adocon; integrated security=SSPI"; //key value pair

using (SqlConnection con = new SqlConnection(cs))

{

SqlCommand cmd = new SqlCommand("select \* from tblStudents", con);

con.Open();

GridView1.DataSource = cmd.ExecuteReader();

GridView1.DataBind();

}

}

}

-----------------------------------------------------------------------------------------------------------------------------------------------------------------

***CREATING CONNECTION STRING AT WEBCONFIG FILE:***

<?xml version="1.0" encoding="utf-8"?>

<!--

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

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

-->

<configuration>

<configSections>

<!-- For more information on Entity Framework configuration, visit http://go.microsoft.com/fwlink/?LinkID=237468 -->

<section name="entityFramework"

type="System.Data.Entity.Internal.ConfigFile.EntityFrameworkSection, EntityFramework, Version=6.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089"

requirePermission="false"/>

</configSections>

***<connectionStrings>***

***<add name="DBCS"***

***connectionString="data source=.; database=adocon; integrated security=SSPI"***

***providerName="System.Data.SqlClient"/>***

***</connectionStrings>***

<system.web>

<authentication mode="None"/>

*Respective code at aspx;*

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.Configuration;***

public partial class Default2 : System.Web.UI.Page

{

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

{

**string cs=ConfigurationManager.ConnectionStrings["DBCS"].ConnectionString;**

using (SqlConnection con = new SqlConnection(cs))

{

SqlCommand cmd = new SqlCommand("select \* from tblStudents", con);

con.Open();

GridView1.DataSource = cmd.ExecuteReader();

GridView1.DataBind();

}

}

}

-----------------------------------------------------------------------------------------------------------------------------------------------------------------

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | Mark Hastings | Male | 900 |
| 2 | Pam Nicholas | Female | 760 |
| 3 | John Stenson | Male | 980 |
| 4 | Ram Gerald | Male | 990 |
| 5 | Ron Simpson | Male | 440 |
| 6 | Able Wicht | Male | 320 |
| 7 | Steve Thompson | Male | 983 |
| 8 | James Bynes | Male | 720 |
| 9 | Mary Ward | Female | 870 |
| 10 | Nick Niron | Male | 680 |
|  |  |  |  |

***The following are the most commonly used methods of the SqlCommand class.***

-----------------------------------------------------------------------------------------------------------------------------------------------------------------

**ExecuteReader** - Use when the T-SQL statement returns more than a single value. For example, if the query returns rows of data.

public partial class Default2 : System.Web.UI.Page

{

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

{

string cs=ConfigurationManager.ConnectionStrings["DBCS"].ConnectionString;

using (SqlConnection con = new SqlConnection(cs))

{

SqlCommand cmd = new SqlCommand("**select \* from tblStudents**", con);

con.Open(); //it should be opened before EXECUTEREADER, otherwise you’ll get an RUNTIME error, not COMPILATION TIME ERROR

GridView1.DataSource = cmd.ExecuteReader();

//an instance of SQLDATAREADER can not be constructed using NEW operator

GridView1.DataBind();

}

}

}

**ExecuteNonQuery** - Use when you want to perform an Insert, Update or Delete operation

[returns int]

public partial class Default2 : System.Web.UI.Page

{

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

{

string cs=ConfigurationManager.ConnectionStrings["DBCS"].ConnectionString;

using (SqlConnection con = new SqlConnection(cs))

{

SqlCommand cmd = new SqlCommand("Insert into tblStudents(Name, Gender, TotalMarks) values('Bull', 'Male', 220)", con);

con.Open();

int tot =cmd.ExecuteNonQuery();

Response.Write("total no of rows are inserted" + tot);

}

}

}

**ExecuteScalar** - Use when the query returns a single(scalar) value. For example, queries that return the total number of rows in a table.

Ex: always reaturns an object…..

public partial class Default2 : System.Web.UI.Page

{

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

{

string cs=ConfigurationManager.ConnectionStrings["DBCS"].ConnectionString;

using (SqlConnection con = new SqlConnection(cs))

{

SqlCommand cmd = new SqlCommand("select count(Name) from tblStudents", con);

con.Open();

int tot =(int)cmd.ExecuteScalar();

Response.Write("total no of rows are" + tot);

}

}

}

-----------------------------------------------------------------------------------------------------------------------------------------------------------------

***SQL INJECTION:***

-----------------------------------------------------------------------------------------------------------------------------------------------------------------

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.UI;

using System.Web.UI.WebControls;

using System.Configuration;

using System.Data.SqlClient;

public partial class Sql\_injection : System.Web.UI.Page

{

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

{

string cs = ConfigurationManager.ConnectionStrings["DBCS"].ConnectionString;

using (SqlConnection con = new SqlConnection(cs))

{

SqlCommand cmd = new SqlCommand**("select \* from tblStudents where Name like '"+ TextBox1.Text + "%'", con**);

con.Open();

GridView1.DataSource = cmd.ExecuteReader();

GridView1.DataBind();

}

}

}

Write: R’; delete from tblstudents--

SqlCommand cmd = new SqlCommand**("select \* from tblStudents where Name like '"+ TextBox1.Text + "%'", con**);

SqlCommand cmd = new SqlCommand**("select \* from tblStudents where Name like '"+** R’; delete from tblstudents-- **+ "%'", con**);

“Select \* from tblstudents where Name like ‘R’; delete from tblstudents**--%’ 🡪became commant part**

**That winds up any database**

-----------------------------------------------------------------------------------------------------------------------------------------------------------------

**Prevention of SQL injection:**

-----------------------------------------------------------------------------------------------------------------------------------------------------------------

**Parameterized;**

public partial class Preventing\_SQL\_Injection : System.Web.UI.Page

{

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

{

string cs = ConfigurationManager.ConnectionStrings["DBCS"].ConnectionString;

using (SqlConnection con = new SqlConnection(cs))

{

string @vol = TextBox1.Text;

SqlCommand cmd = new SqlCommand("select \* from tblStudents where Name like '" + @vol +"%'", con);

con.Open();

GridView1.DataSource = cmd.ExecuteReader();

GridView1.DataBind();

}

}

}

**Stored procedure;**

public partial class Preventing\_SQL\_Injection : System.Web.UI.Page

{

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

{

string cs = ConfigurationManager.ConnectionStrings["DBCS"].ConnectionString;

using (SqlConnection con = new SqlConnection(cs))

{

string @vol = TextBox1.Text;

SqlCommand cmd = new SqlCommand("cat", con);

cmd.CommandType = System.Data.CommandType.StoredProcedure;

cmd.Parameters.AddWithValue("@nose" , TextBox1.Text + "%");

con.Open();

GridView1.DataSource = cmd.ExecuteReader();

GridView1.DataBind();

}

}

}

-----------------------------------------------------------------------------------------------------------------------------------------------------------------

***calling a stored procedure using an output parameter***

-----------------------------------------------------------------------------------------------------------------------------------------------------------------

table:

1 Mike Male 5000

2 Pam Female 3500

3 John Male 2350

4 Sara Female 5700

5 Steve Male 4890

6 Sana Female 4500

7 Micle Male 6000

8 Shawn Male 6080

9 Dean Male 5500

NULL NULL NULL NULL

**stored procedure:**

create proc holyshit

@Name nvarchar(50),

@gen nvarchar(10),

@sal int,

@id int out

as

begin

insert into tblEmployees values(@Name, @gen, @sal)

set @id=SCOPE\_IDENTITY()

print @id

end

//executing the sp

declare @id2 int

execute holyshit 'Shawn', 'Male', 6080, @id2 out

**adding ado.net code**

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.Configuration;

using System.Data;

public partial class calling\_sp\_with\_output\_parameter : System.Web.UI.Page

{

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

{

}

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

{

string cs = ConfigurationManager.ConnectionStrings["DBCS"].ConnectionString;

using (SqlConnection con = new SqlConnection(cs))

{

SqlCommand cmd = new SqlCommand("holyshit", con);

cmd.CommandType = System.Data.CommandType.StoredProcedure;

//providing value to input parameters

cmd.Parameters.AddWithValue("@Name", TextBox1.Text);

cmd.Parameters.AddWithValue("@gen", DropDownList1.SelectedValue);

cmd.Parameters.AddWithValue("@sal", Convert.ToInt32(TextBox2.Text));

//supplying value into output parameters

SqlParameter op = new SqlParameter(); //initialize

op.ParameterName = "@id"; //parameter name

op.SqlDbType = SqlDbType.Int; //parameter datatype

op.Direction = ParameterDirection.Output; //if it is input, output or both

//passing the Output Parameter into the cmd

cmd.Parameters.Add(op);

//execute the command

con.Open();

cmd.ExecuteNonQuery(); //because it is insert type

//take the value form the output parameter

int empid = Convert.ToInt32(op.Value);

//ExecuteNonQuery always returns an object, thats why it

//is needed to be converted at integer as the stored pros returns an integer

Label1.Text = empid.ToString();

}

}

}

-------------------------------------------------------------------------------------------------->

output:

--------------------------------------------------------------------------------------------------->

Employee Name: \_\_\_\_\_\_\_\_ (tb1)

Gender: \_\_\_\_\_\_\_\_\_\_\_ (dropdown)

Salary: \_\_\_\_\_\_\_\_\_\_\_ (tb2)

Submit --> button1

New employee id is: 9 --shows as a result of the output parameter

-----------------------------------------------------------------------------------------------------------------------------------------------------------------

***looping through each row in ado***

----------------------------------------------------------------------------------------------------------------------->

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.Configuration;

using System.Data;

public partial class Sql\_datareader : System.Web.UI.Page

{

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

{

string cs = ConfigurationManager.ConnectionStrings["DBCS"].ConnectionString;

using (SqlConnection con = new SqlConnection(cs))

{

SqlCommand cmd = new SqlCommand("select \* from tblEmployees", con);

con.Open();

//Sqldatareader object should also be closed as soon as the scope expeirs

//to maintain the scalibility and performance of an application.

//try() catch() finally() block can also be used in both cases of RDR or CON

using (SqlDataReader rdr = cmd.ExecuteReader())

{

DataTable table = new DataTable(); //creates another table in order to store processed data

//generating new cols for table

table.Columns.Add("Name");

table.Columns.Add("Gender");

table.Columns.Add("Salary");

table.Columns.Add("da");

//row by row process

while (rdr.Read()) //returns true or false (Boolian) it checks each rows if there is something then it returns true, else false;

{

DataRow dr = table.NewRow(); //

int original = Convert.ToInt32(rdr["Salary"]);

double da = original \* .1; //multiply makes an integer to double

dr["Name"] = rdr["Name"];//1st dr["Name"] comes from table::: rdr["Name"] comes from database table

dr["Gender"] = rdr["Gender"];

dr["Salary"] = Convert.ToInt32(rdr["Salary"]);

dr["da"] = da;

table.Rows.Add(dr);

}

GridView1.DataSource = table;

GridView1.DataBind();

}

}

}

}

|  |  |
| --- | --- |
| 1 Mike Male 5000  2 Pam Female 3500  3 John Male 2350  4 Sara Female 5700  5 Steve Male 4890  6 Sana Female 4500  7 Micle Male 6000  8 Shawn Male 6080 | Name Gender Salary da  Mike Male 5000 500  Pam Female 3500 350  John Male 2350 235  Sara Female 5700 570  Steve Male 4890 489  Sana Female 4500 450  Micle Male 6000 600  Shawn Male 6080 608  Dean Male 5500 550 |

------------------------------------------------------------------------------------------->

***//looping through result sets***

------------------------------------------------------------------------------------------->

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.Configuration;

using System.Data;

public partial class SQLdatareaders\_object\_s\_next\_result\_method : System.Web.UI.Page

{

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

{

string cs = ConfigurationManager.ConnectionStrings["DBCS"].ConnectionString;

using (SqlConnection con = new SqlConnection(cs))

{

//looping through result sets

SqlCommand cmd = new SqlCommand("select \* from tblEmployees;select \* from tblStudents", con);

con.Open();

using (SqlDataReader rdr = cmd.ExecuteReader())

{

GridView1.DataSource = rdr;

GridView1.DataBind();

//if you dont use rdr.nextresult the cursor will stop journey while looping through one result set.

while (rdr.NextResult()) //it also returns true or false.

{

GridView2.DataSource = rdr;

GridView2.DataBind();

}

}

}

}

}

captures both grid view at a time. without using NEXTRESULT cursor will only show the value of 1st query

----------------------------------------------------------------------------------------------------------->

***in case of data adopter***

------------------------------------------------------------------------------------------------------------>

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.Configuration;

using System.Data;

public partial class DATAADAPTER\_MULTIPLE\_TAB : System.Web.UI.Page

{

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

{

string cs = ConfigurationManager.ConnectionStrings["DBCS"].ConnectionString;

using (SqlConnection con = new SqlConnection(cs))

{

SqlDataAdapter cmd = new SqlDataAdapter("PANT", con);

cmd.SelectCommand.CommandType = CommandType.StoredProcedure;

//cmd.SelectCommand.Parameters.AddWithValue("@id", TextBox1.Text); //@id comes from databse

DataSet ds = new DataSet();

cmd.Fill(ds); //opens connection, execute sp, load data=> ds, close connection

GridView1.DataSource = ds;

GridView1.DataBind();

GridView2.DataSource = ds;

GridView2.DataBind();

}

}

}

-------------------------------------------------------------------------------------------------------------->

DATA ADAPTER ALWAYS gives two gridviews if you dontUSE ANY THING. [HAVING SAME VALUE]

SQLCOMMAND never gives 2nd gridview. THE COURSOR LEFT CONTROL AS SOON AS THE first query execution is done.

--------------------------------------------------------------------------------------------------------------->

public partial class DATAADAPTER\_MULTIPLE\_TAB : System.Web.UI.Page

{

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

{

string cs = ConfigurationManager.ConnectionStrings["DBCS"].ConnectionString;

using (SqlConnection con = new SqlConnection(cs))

{

SqlDataAdapter cmd = new SqlDataAdapter("PANT", con);

cmd.SelectCommand.CommandType = CommandType.StoredProcedure;

//cmd.SelectCommand.Parameters.AddWithValue("@id", TextBox1.Text); //@id comes from databse

DataSet ds = new DataSet();

cmd.Fill(ds); //opens connection, execute sp, load data=> ds, close connection

GridView1.DataSource = ds.Tables[0]; //this are called table collection 0 are the index of 1st atble and 1 means 2nd table

GridView1.DataBind();

GridView2.DataSource = ds.Tables[1];

GridView2.DataBind();

}

}

}

------------------------------------------------------------------------------------------------------------>

***SQL DATA ADAPTER***

------------------------------------------------------------------------------------------------------------->

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.Configuration;

using System.Data;

public partial class SQL\_data\_adapter : System.Web.UI.Page

{

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

{

string cs = ConfigurationManager.ConnectionStrings["DBCS"].ConnectionString;

using (SqlConnection con = new SqlConnection(cs))

{

SqlDataAdapter cmd = new SqlDataAdapter("emp", con); //an do all DML operations

cmd.SelectCommand.CommandType = CommandType.StoredProcedure; //not necessory when parameters are not necessory

DataSet ds = new DataSet(); //a dataset is a without connection representation of database

//stores table and relations

//SQL stores tables at HARD DRIVE, but DATASET stores data at SERVER STORAGE

cmd.Fill(ds);

GridView1.DataSource = ds;

GridView1.DataBind();

}

}

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

{

string cs = ConfigurationManager.ConnectionStrings["DBCS"].ConnectionString;

using (SqlConnection con = new SqlConnection(cs))

{

SqlDataAdapter cmd = new SqlDataAdapter("egg", con);

cmd.SelectCommand.CommandType = CommandType.StoredProcedure;

//providing values to input parameter. CHECK "calling a stored procedure using an output parameter" DOCUMENTATION

cmd.SelectCommand.Parameters.AddWithValue("@id", TextBox1.Text); //@id comes from database

DataSet ds = new DataSet();

cmd.Fill(ds);

GridView1.DataSource = ds;

GridView1.DataBind();

}

}

}

|  |  |
| --- | --- |
| create proc egg  @id int  as  begin  select \* from tblEmployees where EmployeeId=@id  end | create proc emp  as  begin  select \* from tblEmployees  end |

-----------------------------------------------------------------------------------------------------------------------------------------------------------------

***CACHING DATA***

-----------------------------------------------------------------------------------------------------------------------------------------------------------------

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.Configuration;

using System.Data;

public partial class Caching\_a\_dataset : System.Web.UI.Page

{

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

{

//if you run cache test once and dont clear it. cache will bip even after the 2nd time retun.

//if sql server is disconnected cached memory can still retrive data

}

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

{

if (Cache["data"] == null)

{

string cs = ConfigurationManager.ConnectionStrings["DBCS"].ConnectionString;

using (SqlConnection con = new SqlConnection(cs))

{

SqlDataAdapter cmd = new SqlDataAdapter("select \* from tblEmployees", con);

DataSet ds = new DataSet();

cmd.Fill(ds);

Cache["data"] = ds; //Cache has a global access, even from web page 1 to web page 5. it provides a global access on the solution

GridView1.DataSource = ds;

GridView1.DataBind();

}

Label1.Text = "Load from db";

}

else

{

GridView1.DataSource = (DataSet)Cache["data"]; //ds had a data set, so Cache["data"] should return a DATASET. but it returns a string. So we got to type cast it.

GridView1.DataBind();

Label1.Text = "Load from cache";

}

}

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

{

if(Cache["data"]!=null)

{

Cache.Remove("data");

Label1.Text = "data set is removed";

}

else

{

Label1.Text = "nothing to removed";

}

}

}

-----------------------------------------------------------------------------------------------------------------------------------------------------------------

***Sqlcommand builder***

-----------------------------------------------------------------------------------------------------------------------------------------------------------------

public partial class Sql\_command\_builder : System.Web.UI.Page

{

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

{

}

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

{

string cs = ConfigurationManager.ConnectionStrings["DBCS"].ConnectionString;

SqlConnection con = new SqlConnection(cs);

string sp = "select \* from tblEmployees where EmployeeId=" + TextBox1.Text; //PK mandatory

SqlDataAdapter da = new SqlDataAdapter(sp, con);

DataSet ds = new DataSet(); //contains the entire row at a time

da.Fill(ds, "Employee"); //feeling the New Employee virtual table from getting data from main table

ViewState["SQ"] = sp; //contains the query

ViewState["dataset"] = ds; //holds the value of virtual table

if(ds.Tables["Employee"].Rows.Count>0) //if it brings at least one item then value will be >0

{

DataRow dr = ds.Tables["Employee"].Rows[0]; //retriving row at index 0

TextBox2.Text = dr["Name"].ToString();

TextBox3.Text = dr["Salary"].ToString();

DropDownList1.SelectedValue = dr["Gender"].ToString();

}

else

{

Label1.Text = "No fucking match";

}

}

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

{

string cs = ConfigurationManager.ConnectionStrings["DBCS"].ConnectionString;

SqlConnection con = new SqlConnection(cs);

SqlDataAdapter da = new SqlDataAdapter(ViewState["SQ"].ToString(), con);

//contains the table associated with the quiey

//sqlCommandBuilder generates automatically Insert, Update, Delete statement

SqlCommandBuilder build = new SqlCommandBuilder(da);

//build.DataAdapter = da;

DataSet ds =(DataSet)ViewState["dataset"]; //by default returns OBJECT

if (ds.Tables["Employee"].Rows.Count > 0)

{

DataRow dr = ds.Tables["Employee"].Rows[0];

//taking updated value

dr["Name"] = TextBox2.Text;

dr["Gender"] = DropDownList1.SelectedValue;

dr["Salary"] = TextBox3.Text;

}

int no\_of\_rows= da.Update(ds, "Employee"); //updates dataset from Employee table(generated before) to main table; that is just getting updated

//returns an integer, the no of rows that has been updated

if(no\_of\_rows>0)

{

Label1.Text = "one row has been updated";

}

}

}

---------------------------------------------------------------------------------------------------------------

inspecting auto generated queries by SQL COMMAND BUILDER OBJECT

---------------------------------------------------------------------------------------------------------------

Label2.Text = build.GetDeleteCommand().CommandText;

Label3.Text = build.GetUpdateCommand().CommandText;

Label4.Text = build.GetInsertCommand().CommandText;

OUTPUT🡪

DELETE FROM [tblEmployees] WHERE (([EmployeeId] = @p1) AND ((@p2 = 1 AND [Name] IS NULL) OR ([Name] = @p3)) AND ((@p4 = 1 AND [Gender] IS NULL) OR ([Gender] = @p5)) AND ((@p6 = 1 AND [Salary] IS NULL) OR ([Salary] = @p7)))   
  
UPDATE [tblEmployees] SET [Name] = @p1, [Gender] = @p2, [Salary] = @p3 WHERE (([EmployeeId] = @p4) AND ((@p5 = 1 AND [Name] IS NULL) OR ([Name] = @p6)) AND ((@p7 = 1 AND [Gender] IS NULL) OR ([Gender] = @p8)) AND ((@p9 = 1 AND [Salary] IS NULL) OR ([Salary] = @p10)))   
  
INSERT INTO [tblEmployees] ([Name], [Gender], [Salary]) VALUES (@p1, @p2, @p3)