(T40)討論DynamicSql。討論SqlInjection在Asp.NetWebForm的SearchBar  
CourseGUID: e48417fc-9db5-4e99-822c-706c5ccef6cc  
=======================================================================  
(T40)討論DynamicSql。討論SqlInjection在Asp.NetWebForm的SearchBar  
=======================================================================  
0. Summary

1. Create Sample Data

2. stored procedure spSearchGamer

3. Dynamic SQL stored procedure

4. Dynamic SQL stored procedure

5. Good dynamic sql queries

-----------

6. SQL Injection

-----------

7. Web Application - DynamicSQL, SearchWebPage

7.1. Set up SQL Authentication

7.2. Create Web Application

-----------

8. Code

8.1. Web.config

8.2. WebForm1.aspx

8.3. WebForm1.aspx.cs

8.4. WebForm2.aspx

8.5. WebForm2.aspx.cs

8.6. WebForm3.aspx

8.7. WebForm3.aspx.cs

-----------

9. Test it

9.1. WebForm1.aspx, WebForm1.aspx.cs, spSearchGamer

9.2. WebForm2.aspx, WebForm2.aspx.cs, spSearchGamer2

9.3. WebForm3.aspx, WebForm3.aspx.cs, spSearchGamer3

-----------

10. Clean up  
=======================================================================

0. Summary

1.

--CREATE PROCEDURE spSearchGamer

--    @FirstName NVARCHAR(100) = NULL ,

--    @LastName NVARCHAR(100) = NULL ,

--    @Gender NVARCHAR(50) = NULL ,

--    @GameScoreGreaterThanOrEqual INT = NULL

--AS

--    BEGIN

--        SELECT  \*

--        FROM    Gamer

--        WHERE   ( FirstName LIKE ( '%' + @FirstName + '%' )

--                  OR @FirstName IS NULL

--                )

--                AND ( LastName LIKE ( '%' + @LastName + '%' )

--                      OR @LastName IS NULL

--                    )

--                AND ( Gender = @Gender

--                      OR @Gender IS NULL

--                    )

--                AND ( GameScore >= @GameScoreGreaterThanOrEqual

--                      OR @GameScoreGreaterThanOrEqual IS NULL

--                    );

--    END;

If we set the default value for the parameter,

that will make the parameter become optional.

Without the parameter default value,

the parameter will become compulsory.

Thus, in where clause we need to add the IS NULL for each parameter

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

2.

2.0.

In Summary:

Building a dynamic sql queries by concatenating strings cause the vulnerability of SQL injection.

Using sp\_executesql parameters is always the best dynamic sql queries.

2.1.

sp\_executesql Syntax

--EXECUTE sp\_executesql @statement, @params, ...user-defined parameters...

sp\_executesql has 2 pre-defined parameters

and any number of user-defined parameters.

2.1.1.

@statement

is the SQL statements to execute

2.1.2.

@params

is a optional pre-defined parameter

and it is used to declare parameters specified in @statement.

2.2.

E.g.

--DECLARE @sql1 NVARCHAR(1000)

--= 'SELECT \*

--FROM Gamer

--WHERE FirstName LIKE ''%' + 'B' + '%'' AND ' + 'LastName LIKE ''%' + 'Y'

--    + '%''';

--EXECUTE sp\_executesql @sql1;

Building a dynamic sql queries by concatenating strings

is a bad dynamic sql queries and

it cause the vulnerability of SQL injection.

2.3.

E.g.

--DECLARE @sq2 NVARCHAR(1000)

--= 'SELECT \*

--FROM Gamer

--WHERE FirstName LIKE ''%''+@FirstName+''%''

--AND LastName LIKE ''%''+@LastName+''%''';

--DECLARE @params NVARCHAR(1000) = '@FirstName NVARCHAR(100), @LastName NVARCHAR(100)';

--EXECUTE sp\_executesql @sq2, @params, @FirstName = 'B', @LastName = 'Y';

Using sp\_executesql parameters is always the best for dynamic sql queries.

=====================================================================

1. Create Sample Data

--=====================================================================

--T040\_01\_Create Sample Data

--=====================================================================

IF ( EXISTS ( SELECT    \*

              FROM      INFORMATION\_SCHEMA.TABLES

              WHERE     TABLE\_NAME = 'Gamer' ) )

    BEGIN

        TRUNCATE TABLE dbo.Gamer;

        DROP TABLE Gamer;

    END;

GO -- Run the previous command and begins new batch

CREATE TABLE Gamer

(

  Id INT IDENTITY(1, 1)

         PRIMARY KEY ,

  FirstName NVARCHAR(50) ,

  LastName NVARCHAR(50) ,

  Gender NVARCHAR(50) ,

  GameScore INT

);

GO -- Run the previous command and begins new batch

INSERT  INTO Gamer

VALUES  ( 'AFirst01', 'XLast01', 'Female', 3500 );

INSERT  INTO Gamer

VALUES  ( 'AFirst02', 'YLast02', 'Female', 4000 );

INSERT  INTO Gamer

VALUES  ( 'BFirst03', 'YLast03', 'Male', 4600 );

INSERT  INTO Gamer

VALUES  ( 'BFirst04', 'YLast04', 'Male', 5400 );

INSERT  INTO Gamer

VALUES  ( 'BFirst05', 'ZLast05', 'Female', 2000 );

INSERT  INTO Gamer

VALUES  ( 'CFirst06', 'YLast06', 'Male', 4320 );

INSERT  INTO Gamer

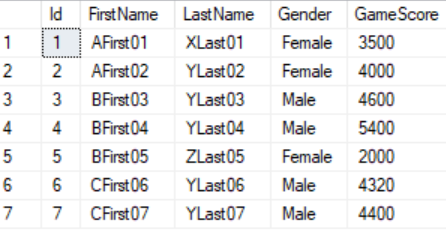
VALUES  ( 'CFirst07', 'YLast07', 'Male', 4400 );

GO -- Run the previous command and begins new batch

SELECT  \*

FROM    Gamer;

GO -- Run the previous command and begins new batch



=====================================================================

2. stored procedure spSearchGamer

--=====================================================================

--T040\_02\_stored procedure spSearchGamer

--=====================================================================

IF ( EXISTS ( SELECT    \*

              FROM      INFORMATION\_SCHEMA.ROUTINES

              WHERE     ROUTINE\_TYPE = 'PROCEDURE'

                        AND LEFT(ROUTINE\_NAME, 3) NOT IN ( 'sp\_', 'xp\_', 'ms\_' )

                        AND SPECIFIC\_NAME = 'spSearchGamer' ) )

    BEGIN

        DROP PROCEDURE spSearchGamer;

    END;

GO -- Run the previous command and begins new batch

CREATE PROCEDURE spSearchGamer

    @FirstName NVARCHAR(100) = NULL ,

    @LastName NVARCHAR(100) = NULL ,

    @Gender NVARCHAR(50) = NULL ,

    @GameScoreGreaterThanOrEqual INT = NULL

AS

    BEGIN

        SELECT  \*

        FROM    Gamer

        WHERE   ( FirstName LIKE ( '%' + @FirstName + '%' )

                  OR @FirstName IS NULL

                )

                AND ( LastName LIKE ( '%' + @LastName + '%' )

                      OR @LastName IS NULL

                    )

                AND ( Gender = @Gender

                      OR @Gender IS NULL

                    )

                AND ( GameScore >= @GameScoreGreaterThanOrEqual

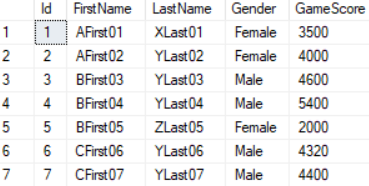
                      OR @GameScoreGreaterThanOrEqual IS NULL

                    );

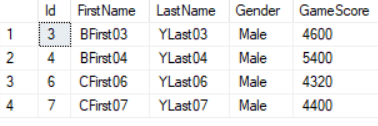
    END;

GO -- Run the previous command and begins new batch

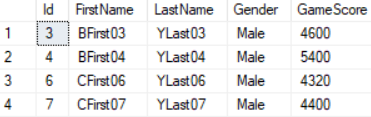
EXECUTE spSearchGamer;



EXECUTE spSearchGamer @Gender = 'Male';



EXECUTE spSearchGamer @Gender = 'Male', @LastName = 'Y';



EXECUTE spSearchGamer @Gender = 'Male', @FirstName = 'B', @LastName = 'Y';

A picture containing text

Description automatically generated

EXECUTE spSearchGamer @Gender = 'Male', @FirstName = 'B', @LastName = 'Y',

    @GameScoreGreaterThanOrEqual = 5000;



GO -- Run the previous command and begins new batch

/\*

1.

--@FirstName NVARCHAR(100) = NULL ,

...

--WHERE   ( FirstName LIKE ( '%' + @FirstName + '%' )

--    OR @FirstName IS NULL

--)

If we set the default value for the parameter,

that will make the parameter become optional.

Without the parameter default value,

the parameter will become compulsory.

Thus, in where clause we need to add the IS NULL for each parameter

2.

In this case, the stored procedure is easy to maintan,

because it only has 4 filters.

When it has more than 10 filters,

it will contain a lof of AND, OR ... in the filters

and this is too complex to maintain.

Thus, we need Dynamic SQL stored procedure, sp\_executesql.

\*/

=====================================================================

3. Dynamic SQL stored procedure

--=====================================================================

--T040\_03\_Dynamic SQL stored procedure

--=====================================================================

--Dynamic SQL stored procedure

--EXECUTE sp\_executesql @statement, @params, ...user-defined parameters...

--=====================================================================

--T040\_03\_01

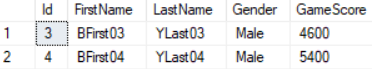
SELECT  \*

FROM    Gamer

WHERE   FirstName LIKE '%B%'

        AND LastName LIKE '%Y%';

GO -- Run the previous command and begins new batch



--=====================================================================

--T040\_03\_02

--Bad dynamic sql queries.

--Building a dynamic sql queries by concatenating strings cause the vulnerability of SQL injection.

DECLARE @sql1 NVARCHAR(1000)

= 'SELECT \*

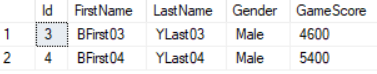
FROM Gamer

WHERE FirstName LIKE ''%' + 'B' + '%'' AND ' + 'LastName LIKE ''%' + 'Y'

    + '%''';

EXECUTE sp\_executesql @sql1;

GO -- Run the previous command and begins new batch



--=====================================================================

--T040\_03\_03

--Good dynamic sql queries.

--Using sp\_executesql parameters is always the best for dynamic sql queries.

DECLARE @sq2 NVARCHAR(1000)

= 'SELECT \*

FROM Gamer

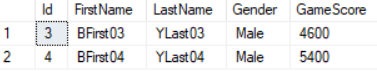
WHERE FirstName LIKE ''%''+@FirstName+''%''

AND LastName LIKE ''%''+@LastName+''%''';

DECLARE @params NVARCHAR(1000) = '@FirstName NVARCHAR(100), @LastName NVARCHAR(100)';

EXECUTE sp\_executesql @sq2, @params, @FirstName = 'B', @LastName = 'Y';

GO -- Run the previous command and begins new batch



/\*

1.

1.0.

In Summary:

Building a dynamic sql queries by concatenating strings cause the vulnerability of SQL injection.

Using sp\_executesql parameters is always the best dynamic sql queries.

1.1.

sp\_executesql Syntax

--EXECUTE sp\_executesql @statement, @params, ...user-defined parameters...

sp\_executesql has 2 pre-defined parameters

and any number of user-defined parameters.

1.1.1.

@statement

is the SQL statements to execute

1.1.2.

@params

is a optional pre-defined parameter

and it is used to declare parameters specified in @statement.

1.2.

E.g.

--DECLARE @sql1 NVARCHAR(1000)

--= 'SELECT \*

--FROM Gamer

--WHERE FirstName LIKE ''%' + 'B' + '%'' AND ' + 'LastName LIKE ''%' + 'Y'

--    + '%''';

--EXECUTE sp\_executesql @sql1;

Building a dynamic sql queries by concatenating strings

is a bad dynamic sql queries and

it cause the vulnerability of SQL injection.

1.3.

--DECLARE @sq2 NVARCHAR(1000)

--= 'SELECT \*

--FROM Gamer

--WHERE FirstName LIKE ''%''+@FirstName+''%''

--AND LastName LIKE ''%''+@LastName+''%''';

--DECLARE @params NVARCHAR(1000) = '@FirstName NVARCHAR(100), @LastName NVARCHAR(100)';

--EXECUTE sp\_executesql @sq2, @params, @FirstName = 'B', @LastName = 'Y';

Using sp\_executesql parameters is always the best for dynamic sql queries.

\*/

=====================================================================

4. Dynamic SQL stored procedure

--=====================================================================

--T040\_04\_Bad dynamic sql queries

--=====================================================================

--=====================================================================

--T040\_04\_01

--Drop Store Procedure if it exists then recreate.

IF ( EXISTS ( SELECT    \*

              FROM      INFORMATION\_SCHEMA.ROUTINES

              WHERE     ROUTINE\_TYPE = 'PROCEDURE'

                        AND LEFT(ROUTINE\_NAME, 3) NOT IN ( 'sp\_', 'xp\_', 'ms\_' )

                        AND SPECIFIC\_NAME = 'spSearchGamer2' ) )

    BEGIN

        DROP PROCEDURE spSearchGamer2;

    END;

GO -- Run the previous command and begins new batch

CREATE PROCEDURE spSearchGamer2

(

  @FirstName NVARCHAR(100) = NULL ,

  @LastName NVARCHAR(100) = NULL ,

  @Gender NVARCHAR(50) = NULL ,

  @GameScoreGreaterThanOrEqual INT = NULL

)

AS

    BEGIN

        DECLARE @sql NVARCHAR(MAX);

        SET @sql = 'SELECT \* FROM Gamer WHERE 1 = 1';

        IF ( @FirstName IS NOT NULL )

            SET @sql = @sql + ' AND FirstName LIKE ''%' + @FirstName + '%''';

        IF ( @LastName IS NOT NULL )

            SET @sql = @sql + ' AND LastName LIKE ''%' + @LastName + '%''';

        IF ( @Gender IS NOT NULL )

            SET @sql = @sql + ' AND Gender=''' + @Gender + '''';

        IF ( @GameScoreGreaterThanOrEqual IS NOT NULL )

            SET @sql = @sql + ' AND GameScore>='''

                + CAST(@GameScoreGreaterThanOrEqual AS NVARCHAR(100)) + '''';

        EXECUTE sp\_executesql @sql;

    END;

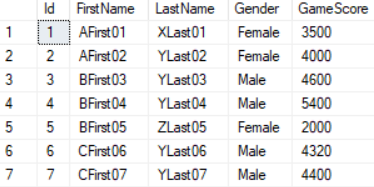
GO -- Run the previous command and begins new batch

--=====================================================================

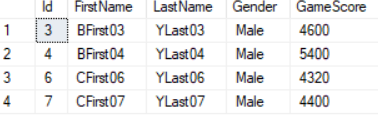
--T040\_04\_02

--EXECUTE spSearchGamer2

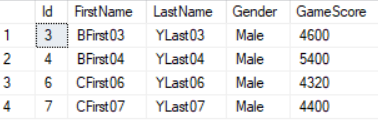
EXECUTE spSearchGamer2;



EXECUTE spSearchGamer2 @Gender = 'Male';



EXECUTE spSearchGamer2 @Gender = 'Male', @LastName = 'Y';



EXECUTE spSearchGamer2 @Gender = 'Male', @FirstName = 'B', @LastName = 'Y';



EXECUTE spSearchGamer2 @Gender = 'Male', @FirstName = 'B', @LastName = 'Y',

    @GameScoreGreaterThanOrEqual = 5000;



====================================================================

5. Good dynamic sql queries

--=====================================================================

--T040\_05\_Good dynamic sql queries

--=====================================================================

--=====================================================================

--T040\_05\_01

--Drop Store Procedure if it exists then recreate.

IF ( EXISTS ( SELECT    \*

              FROM      INFORMATION\_SCHEMA.ROUTINES

              WHERE     ROUTINE\_TYPE = 'PROCEDURE'

                        AND LEFT(ROUTINE\_NAME, 3) NOT IN ( 'sp\_', 'xp\_', 'ms\_' )

                        AND SPECIFIC\_NAME = 'spSearchGamer3' ) )

    BEGIN

        DROP PROCEDURE spSearchGamer3;

    END;

GO -- Run the previous command and begins new batch

CREATE PROCEDURE spSearchGamer3

    @FirstName NVARCHAR(100) = NULL ,

    @LastName NVARCHAR(100) = NULL ,

    @Gender NVARCHAR(50) = NULL ,

    @GameScoreGreaterThanOrEqual INT = NULL

AS

    BEGIN

        DECLARE @sqlParams NVARCHAR(MAX) = N'@FN NVARCHAR(100), @LN NVARCHAR(100), @Gen NVARCHAR(50), @Gsgtoe INT';

        DECLARE @sql NVARCHAR(MAX);

        SET @sql = 'SELECT \* FROM Gamer WHERE 1 = 1';

        IF ( @FirstName IS NOT NULL )

            SET @sql = @sql + ' AND FirstName LIKE ''%''+@FN+''%''';

        IF ( @LastName IS NOT NULL )

            SET @sql = @sql + ' AND LastName LIKE ''%''+@LN+''%''';

        IF ( @Gender IS NOT NULL )

            SET @sql = @sql + ' AND Gender=@Gen';

        IF ( @GameScoreGreaterThanOrEqual IS NOT NULL )

            SET @sql = @sql + ' AND GameScore>=@Gsgtoe';

        EXECUTE sp\_executesql @sql, @sqlParams, @FN = @FirstName,

            @LN = @LastName, @Gen = @Gender,

            @Gsgtoe = @GameScoreGreaterThanOrEqual;

    END;

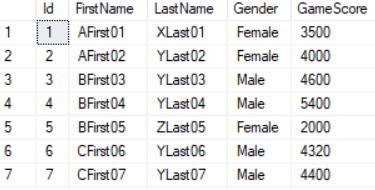
GO -- Run the previous command and begins new batch

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

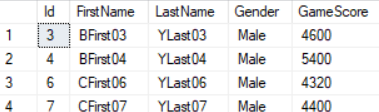
--T040\_05\_02

--EXECUTE spSearchGamer3

EXECUTE spSearchGamer3;



EXECUTE spSearchGamer3 @Gender = 'Male';

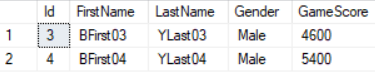


EXECUTE spSearchGamer3 @Gender = 'Male', @LastName = 'Y';

Table

Description automatically generated

EXECUTE spSearchGamer3 @Gender = 'Male', @FirstName = 'B', @LastName = 'Y';



EXECUTE spSearchGamer3 @Gender = 'Male', @FirstName = 'B', @LastName = 'Y',

    @GameScoreGreaterThanOrEqual = 5000;



GO -- Run the previous command and begins new batch

6. SQL Injection

--=====================================================================

--T040\_06\_SQL Injection

--=====================================================================

--=====================================================================

--T040\_06\_01

--Create Sample data

IF ( EXISTS ( SELECT    \*

              FROM      INFORMATION\_SCHEMA.TABLES

              WHERE     TABLE\_NAME = 'Table1' ) )

    BEGIN

        TRUNCATE TABLE dbo.Table1;

        DROP TABLE Table1;

    END;

GO -- Run the previous command and begins new batch

CREATE TABLE Table1

(

  Id INT IDENTITY(1, 1)

         PRIMARY KEY ,

  [Name] NVARCHAR(50)

);

GO -- Run the previous command and begins new batch

--=====================================================================

--T040\_06\_02

EXECUTE sp\_executesql N'SELECT \* FROM Gamer WHERE 1 = 1 AND FirstName=@FirstName',

    N'@FirstName NVARCHAR(26)', @FirstName = N'AFirst01';

SELECT  \*

FROM    Table1;

GO -- Run the previous command and begins new batch

/\*

Display the FirstName=N'AFirst01'

\*/

--=====================================================================

--T040\_06\_03

EXECUTE sp\_executesql N'SELECT \* FROM Gamer WHERE 1 = 1 AND FirstName=@FirstName',

    N'@FirstName NVARCHAR(26)', @FirstName = N'''; DROP TABLE dbo.Table1; --';

SELECT  \*

FROM    Table1;

GO -- Run the previous command and begins new batch

/\*

sp\_executesql with parameters is parameterised queries.

Thus, it prevent SQL Injection

\*/

--=====================================================================

--T040\_06\_04

DECLARE @sql2 NVARCHAR(1000) = N'SELECT \* FROM Gamer WHERE 1 = 1 AND FirstName='''

    + '''; DROP TABLE dbo.Table1; --' + ', AND LastName=@LastName';

EXECUTE sp\_executesql @sql2;

SELECT  \*

FROM    Table1;

GO -- Run the previous command and begins new batch

/\*

\*\*SQL Injection

The Table1 will be dropped.

\*/

--=====================================================================

--T040\_06\_05

--Create Sample data

IF ( EXISTS ( SELECT    \*

              FROM      INFORMATION\_SCHEMA.TABLES

              WHERE     TABLE\_NAME = 'Table1' ) )

    BEGIN

        TRUNCATE TABLE dbo.Table1;

        DROP TABLE Table1;

    END;

GO -- Run the previous command and begins new batch

CREATE TABLE Table1

(

  Id INT IDENTITY(1, 1)

         PRIMARY KEY ,

  [Name] NVARCHAR(50)

);

GO -- Run the previous command and begins new batch

--=====================================================================

--T040\_06\_06

EXECUTE spSearchGamer2 @Gender = 'Male';

SELECT  \*

FROM    Table1;

GO -- Run the previous command and begins new batch

--=====================================================================

--T040\_06\_07

EXECUTE spSearchGamer2 @Gender = N'''; DROP TABLE dbo.Table1; --';

SELECT  \*

FROM    Table1;

GO -- Run the previous command and begins new batch

/\*

\*\*SQL Injection

The Table1 will be dropped.

\*/

--=====================================================================

--T040\_06\_08

--Create Sample data

IF ( EXISTS ( SELECT    \*

              FROM      INFORMATION\_SCHEMA.TABLES

              WHERE     TABLE\_NAME = 'Table1' ) )

    BEGIN

        TRUNCATE TABLE dbo.Table1;

        DROP TABLE Table1;

    END;

GO -- Run the previous command and begins new batch

CREATE TABLE Table1

(

  Id INT IDENTITY(1, 1)

         PRIMARY KEY ,

  [Name] NVARCHAR(50)

);

GO -- Run the previous command and begins new batch

--=====================================================================

--T040\_06\_09

--Bad dynamic sql queries.

--Building a dynamic sql queries by concatenating strings cause the vulnerability of SQL injection.

DECLARE @sql1 NVARCHAR(1000)

= 'SELECT \*

FROM Gamer

WHERE FirstName LIKE ''%' + 'B' + '%'' AND ' + 'LastName LIKE ''%' + 'Y'

    + '%''';

EXECUTE sp\_executesql @sql1;

GO -- Run the previous command and begins new batch

/\*

Display the FirstName LIKE '%B%' AND LastName LIKE '%Y%'

\*/

--=====================================================================

--T040\_06\_10

--Bad dynamic sql queries.

--Building a dynamic sql queries by concatenating strings cause the vulnerability of SQL injection.

DECLARE @sql1 NVARCHAR(1000)

= 'SELECT \*

FROM Gamer

WHERE FirstName LIKE ''%' + N'''; DROP TABLE dbo.Table1; --' + '%'' AND ' + 'LastName LIKE ''%' + 'Y'

    + '%''';

EXECUTE sp\_executesql @sql1;

SELECT  \*

FROM    Table1;

GO -- Run the previous command and begins new batch

/\*

\*\*SQL Injection

The Table1 will be dropped.

\*/

--=====================================================================

--T040\_06\_11

--Create Sample data

IF ( EXISTS ( SELECT    \*

              FROM      INFORMATION\_SCHEMA.TABLES

              WHERE     TABLE\_NAME = 'Table1' ) )

    BEGIN

        TRUNCATE TABLE dbo.Table1;

        DROP TABLE Table1;

    END;

GO -- Run the previous command and begins new batch

CREATE TABLE Table1

(

  Id INT IDENTITY(1, 1)

         PRIMARY KEY ,

  [Name] NVARCHAR(50)

);

GO -- Run the previous command and begins new batch

--=====================================================================

--T040\_06\_12

--Good dynamic sql queries.

--Using sp\_executesql parameters is always the best for dynamic sql queries.

DECLARE @sq2 NVARCHAR(1000)

= 'SELECT \*

FROM Gamer

WHERE FirstName LIKE ''%''+@FirstName+''%''

AND LastName LIKE ''%''+@LastName+''%''';

DECLARE @params NVARCHAR(1000) = '@FirstName NVARCHAR(100), @LastName NVARCHAR(100)';

EXECUTE sp\_executesql @sq2, @params, @FirstName = 'B', @LastName = 'Y';

GO -- Run the previous command and begins new batch

/\*

Display the FirstName LIKE '%B%' AND LastName LIKE '%Y%'

\*/

--=====================================================================

--T040\_06\_13

--Good dynamic sql queries.

--Using sp\_executesql parameters is always the best for dynamic sql queries.

DECLARE @sq2 NVARCHAR(1000)

= 'SELECT \*

FROM Gamer

WHERE FirstName LIKE ''%''+@FirstName+''%''

AND LastName LIKE ''%''+@LastName+''%''';

DECLARE @params NVARCHAR(1000) = '@FirstName NVARCHAR(100), @LastName NVARCHAR(100)';

EXECUTE sp\_executesql @sq2, @params, @FirstName =  N'''; DROP TABLE dbo.Table1; --', @LastName = 'Y';

SELECT  \*

FROM    Table1;

GO -- Run the previous command and begins new batch

/\*

\*\*Prevent SQL Injection

The Table1 will NOT be dropped.

\*/

--=====================================================================

--T040\_06\_14

EXECUTE spSearchGamer3 @Gender = 'Male';

EXECUTE spSearchGamer3 @Gender = 'Male', @FirstName = 'B', @LastName = 'Y',

    @GameScoreGreaterThanOrEqual = 5000;

SELECT  \*

FROM    Table1;

GO -- Run the previous command and begins new batch

--=====================================================================

--T040\_06\_15

EXECUTE spSearchGamer3 @Gender = N'''; DROP TABLE dbo.Table1; --';

SELECT  \*

FROM    Table1;

EXECUTE spSearchGamer3 @Gender = 'Male', @FirstName = N'''; DROP TABLE dbo.Table1; --', @LastName = 'Y',

    @GameScoreGreaterThanOrEqual = 5000;

SELECT  \*

FROM    Table1;

GO -- Run the previous command and begins new batch

/\*

\*\*Prevent SQL Injection

The Table1 will NOT be dropped.

\*/

7. Web Application - DynamicSQL, SearchWebPage

7.1. Set up SQL Authentication

In SQL server

Object Explorer --> Security --> Logins --> New Logins

-->

General Tab

Login Name :

Tester

Password:

1234

Default Database:

Sample

-->

Server Roles Tab

Select

**sysadmin**

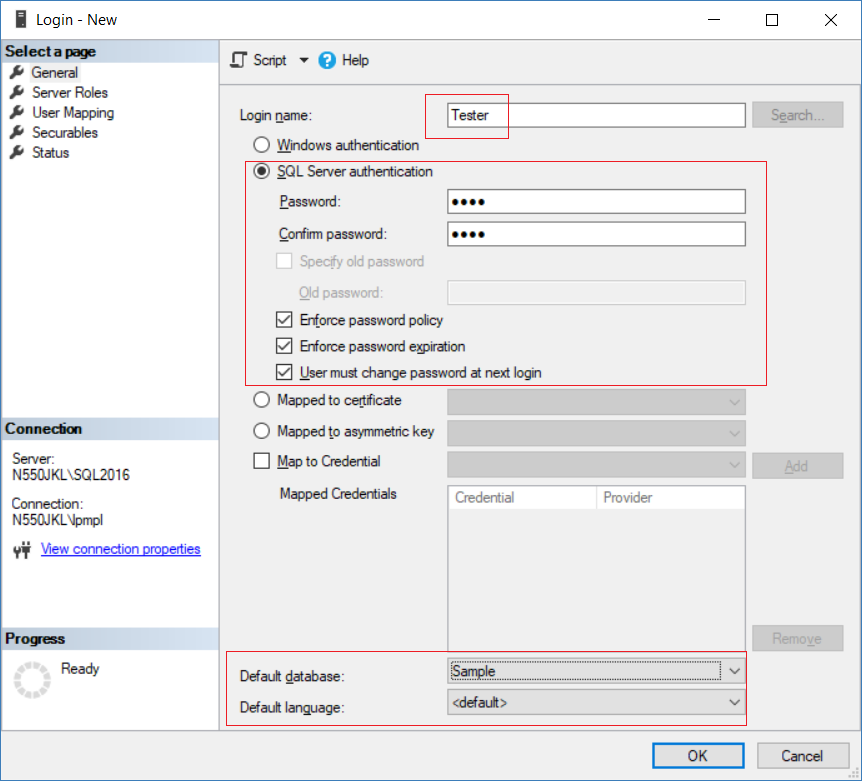
-->

User Mapping Tab

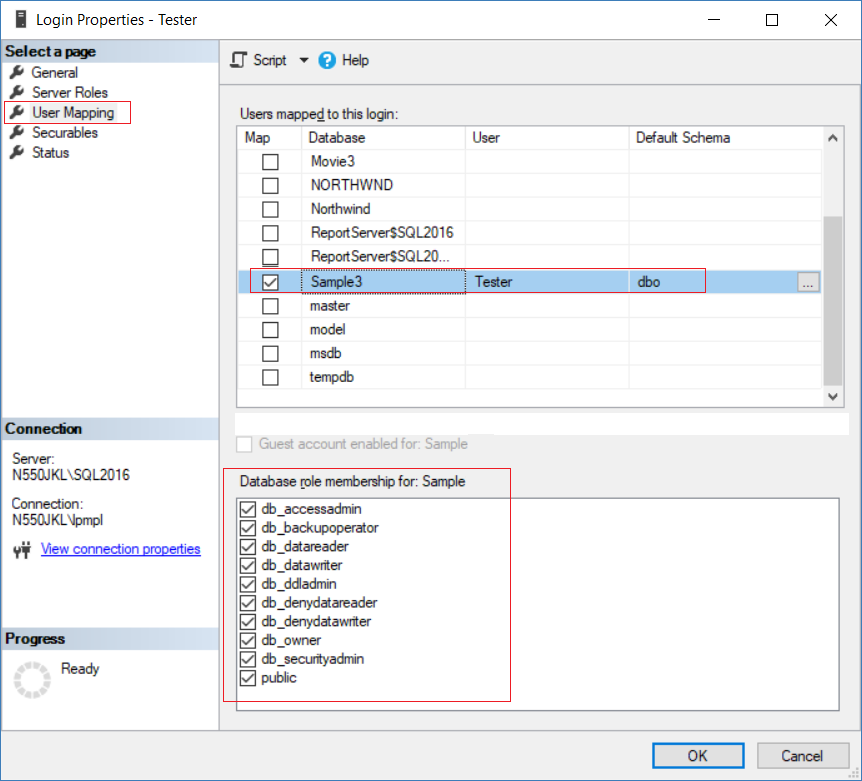
Select Sample

Select every Roles.









7.2. Create Web Application

Do **not** Execute

--Clean up

in previous section

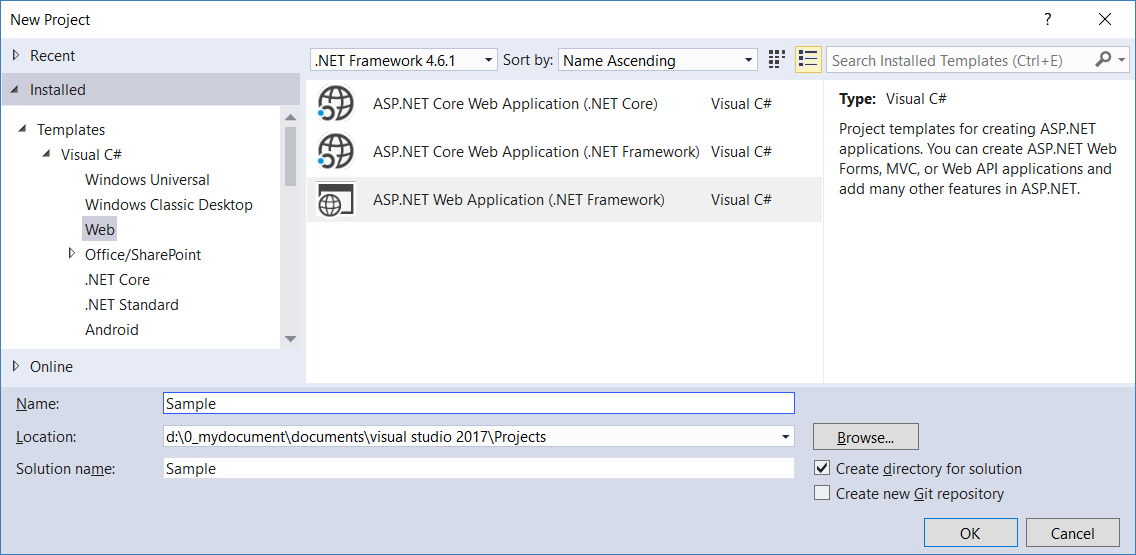
New Project --> Web --> ASP.NET Web Application (.Net Framework)

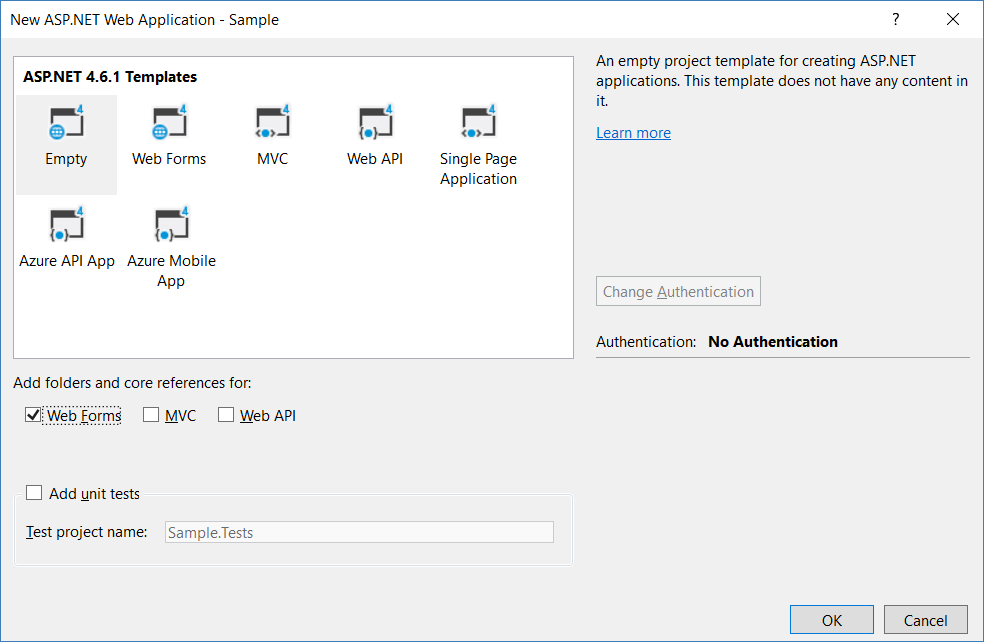
-->

Name:

Sample

--> Web Forms --> OK

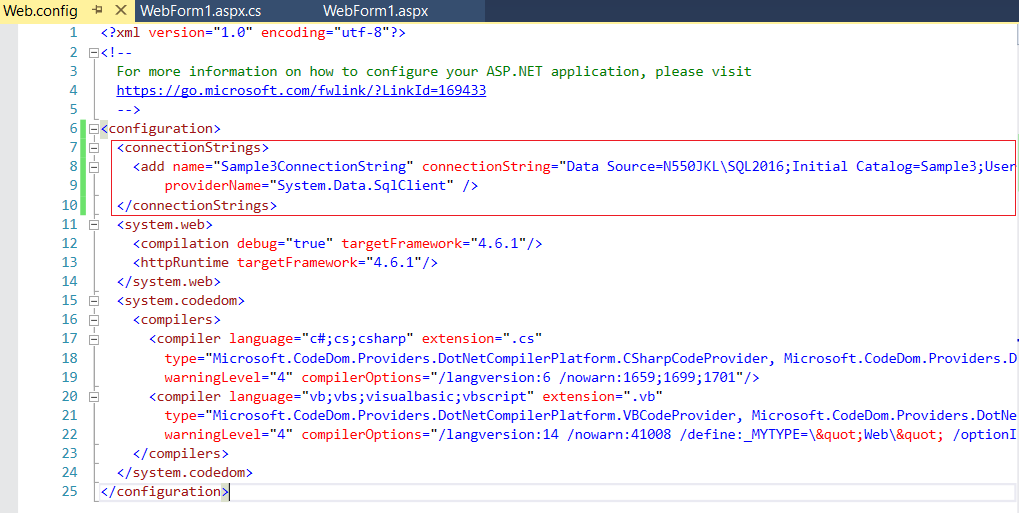




8. Code

8.1. Web.config

Add connection String



<configuration>

  <connectionStrings>

    <add name="SampleConnectionString" connectionString="Data Source=N550JKL\SQL2016;Initial Catalog=Sample;User ID=Tester;Password=1234"

        providerName="System.Data.SqlClient" />

  </connectionStrings>

.....

8.2. WebForm1.aspx

This form will use spSearchGamer

<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="WebForm1.aspx.cs" Inherits="Sample.WebForm1" %>

<!DOCTYPE html>

<html xmlns="[http://www.w3.org/1999/xhtml">](http://www.w3.org/1999/xhtml%22%3E);

<head runat="server">

    <title>Web Search</title>

</head>

<body>

    <form id="form1" runat="server">

        <div>

            <h3>Employee Search Form</h3>

            <div>

                <label for="inputFirstname">

                    Firstname

                </label>

                <input type="text" runat="server"

                    id="inputFirstname" />

            </div>

            <div>

                <label for="inputLastname">

                    Lastname

                </label>

                <input type="text" runat="server" id="inputLastname" />

            </div>

            <div>

                <label for="inputGender">

                    Gender

                </label>

                <input type="text" runat="server"

                    id="inputGender" />

            </div>

            <div>

                <label for="inputGameScoreGreaterThanOrEqual">

                    GameScore >=

                </label>

                <input type="number" runat="server"

                    id="inputGameScoreGreaterThanOrEqual" />

            </div>

            <div>

                <asp:Button ID="btnSearch" runat="server" Text="Search"

                    OnClick="btnSearch\_Click" />

            </div>

        </div>

        <div>

            <h3>Search Results</h3>

            <div>

                <asp:GridView

                    ID="gvResults" runat="server">

                </asp:GridView>

            </div>

        </div>

    </form>

</body>

</html>

8.3. WebForm1.aspx.cs

This form will use spSearchGamer

using System;

using System.Configuration;

using System.Data;

using System.Data.SqlClient;

using System.Web.UI;

namespace Sample

{

    public partial class WebForm1 : Page

    {

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

        {

        }

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

        {

            string connectionStr = ConfigurationManager

                .ConnectionStrings["SampleConnectionString"].ConnectionString;

            using (var con = new SqlConnection(connectionStr))

            {

                var cmd = new SqlCommand();

                cmd.Connection = con;

[cmd.CommandText](http://cmd.commandtext/) = "spSearchGamer";

[cmd.CommandType](http://cmd.commandtype/) = CommandType.StoredProcedure;

                if (inputFirstname.Value.Trim() != "")

                {

                    var param = new SqlParameter

                        ("@FirstName", inputFirstname.Value);

                    cmd.Parameters.Add(param);

                }

                if (inputLastname.Value.Trim() != "")

                {

                    var param = new SqlParameter

                        ("@LastName", inputLastname.Value);

                    cmd.Parameters.Add(param);

                }

                if (inputGender.Value.Trim() != "")

                {

                    var param = new SqlParameter

                        ("@Gender", inputGender.Value);

                    cmd.Parameters.Add(param);

                }

                if (inputGameScoreGreaterThanOrEqual.Value.Trim() != "")

                {

                    var param = new SqlParameter

                        ("@GameScoreGreaterThanOrEqual", inputGameScoreGreaterThanOrEqual.Value);

                    cmd.Parameters.Add(param);

                }

                con.Open();

                SqlDataReader rdr = cmd.ExecuteReader();

                gvResults.DataSource = rdr;

                gvResults.DataBind();

            }

        }

    }

}

8.4. WebForm2.aspx

This form will use spSearchGamer2

<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="WebForm2.aspx.cs" Inherits="Sample.WebForm2" %>

<!DOCTYPE html>

<html xmlns="[http://www.w3.org/1999/xhtml">](http://www.w3.org/1999/xhtml%22%3E);

<head runat="server">

    <title>Web Search</title>

</head>

<body>

    <form id="form1" runat="server">

        <div>

            <h3>Employee Search Form</h3>

            <div>

                <label for="inputFirstname">

                    Firstname

                </label>

                <input type="text" runat="server"

                    id="inputFirstname" />

            </div>

            <div>

                <label for="inputLastname">

                    Lastname

                </label>

                <input type="text" runat="server" id="inputLastname" />

            </div>

            <div>

                <label for="inputGender">

                    Gender

                </label>

                <input type="text" runat="server"

                    id="inputGender" />

            </div>

            <div>

                <label for="inputGameScoreGreaterThanOrEqual">

                    GameScore >=

                </label>

                <input type="number" runat="server"

                    id="inputGameScoreGreaterThanOrEqual" />

            </div>

            <div>

                <asp:Button ID="btnSearch" runat="server" Text="Search"

                    OnClick="btnSearch\_Click" />

            </div>

        </div>

        <div>

            <h3>Search Results</h3>

            <div>

                <asp:GridView

                    ID="gvResults" runat="server">

                </asp:GridView>

            </div>

        </div>

    </form>

</body>

</html>

8.5. WebForm2.aspx.cs

This form will use spSearchGamer2

using System;

using System.Configuration;

using System.Data;

using System.Data.SqlClient;

using System.Web.UI;

namespace Sample

{

    public partial class WebForm2 : Page

    {

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

        {

        }

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

        {

            string connectionStr = ConfigurationManager

                .ConnectionStrings["SampleConnectionString"].ConnectionString;

            using (var con = new SqlConnection(connectionStr))

            {

                var cmd = new SqlCommand();

                cmd.Connection = con;

[cmd.CommandText](http://cmd.commandtext/) = "spSearchGamer2";

[cmd.CommandType](http://cmd.commandtype/) = CommandType.StoredProcedure;

                if (inputFirstname.Value.Trim() != "")

                {

                    var param = new SqlParameter

                        ("@FirstName", inputFirstname.Value);

                    cmd.Parameters.Add(param);

                }

                if (inputLastname.Value.Trim() != "")

                {

                    var param = new SqlParameter

                        ("@LastName", inputLastname.Value);

                    cmd.Parameters.Add(param);

                }

                if (inputGender.Value.Trim() != "")

                {

                    var param = new SqlParameter

                        ("@Gender", inputGender.Value);

                    cmd.Parameters.Add(param);

                }

                if (inputGameScoreGreaterThanOrEqual.Value.Trim() != "")

                {

                    var param = new SqlParameter

                        ("@GameScoreGreaterThanOrEqual", inputGameScoreGreaterThanOrEqual.Value);

                    cmd.Parameters.Add(param);

                }

                con.Open();

                SqlDataReader rdr = cmd.ExecuteReader();

                gvResults.DataSource = rdr;

                gvResults.DataBind();

            }

        }

    }

}

8.6. WebForm3.aspx

This form will use spSearchGamer3

<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="WebForm3.aspx.cs" Inherits="Sample.WebForm3" %>

<!DOCTYPE html>

<html xmlns="[http://www.w3.org/1999/xhtml">](http://www.w3.org/1999/xhtml%22%3E);

<head runat="server">

    <title>Web Search</title>

</head>

<body>

    <form id="form1" runat="server">

        <div>

            <h3>Employee Search Form</h3>

            <div>

                <label for="inputFirstname">

                    Firstname

                </label>

                <input type="text" runat="server"

                    id="inputFirstname" />

            </div>

            <div>

                <label for="inputLastname">

                    Lastname

                </label>

                <input type="text" runat="server" id="inputLastname" />

            </div>

            <div>

                <label for="inputGender">

                    Gender

                </label>

                <input type="text" runat="server"

                    id="inputGender" />

            </div>

            <div>

                <label for="inputGameScoreGreaterThanOrEqual">

                    GameScore >=

                </label>

                <input type="number" runat="server"

                    id="inputGameScoreGreaterThanOrEqual" />

            </div>

            <div>

                <asp:Button ID="btnSearch" runat="server" Text="Search"

                    OnClick="btnSearch\_Click" />

            </div>

        </div>

        <div>

            <h3>Search Results</h3>

            <div>

                <asp:GridView

                    ID="gvResults" runat="server">

                </asp:GridView>

            </div>

        </div>

    </form>

</body>

</html>

8.7. WebForm3.aspx.cs

This form will use spSearchGamer3

using System;

using System.Configuration;

using System.Data;

using System.Data.SqlClient;

using System.Web.UI;

namespace Sample

{

    public partial class WebForm2 : Page

    {

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

        {

        }

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

        {

            string connectionStr = ConfigurationManager

                .ConnectionStrings["SampleConnectionString"].ConnectionString;

            using (var con = new SqlConnection(connectionStr))

            {

                var cmd = new SqlCommand();

                cmd.Connection = con;

[cmd.CommandText](http://cmd.commandtext/) = "spSearchGamer3";

[cmd.CommandType](http://cmd.commandtype/) = CommandType.StoredProcedure;

                if (inputFirstname.Value.Trim() != "")

                {

                    var param = new SqlParameter

                        ("@FirstName", inputFirstname.Value);

                    cmd.Parameters.Add(param);

                }

                if (inputLastname.Value.Trim() != "")

                {

                    var param = new SqlParameter

                        ("@LastName", inputLastname.Value);

                    cmd.Parameters.Add(param);

                }

                if (inputGender.Value.Trim() != "")

                {

                    var param = new SqlParameter

                        ("@Gender", inputGender.Value);

                    cmd.Parameters.Add(param);

                }

                if (inputGameScoreGreaterThanOrEqual.Value.Trim() != "")

                {

                    var param = new SqlParameter

                        ("@GameScoreGreaterThanOrEqual", inputGameScoreGreaterThanOrEqual.Value);

                    cmd.Parameters.Add(param);

                }

                con.Open();

                SqlDataReader rdr = cmd.ExecuteReader();

                gvResults.DataSource = rdr;

                gvResults.DataBind();

            }

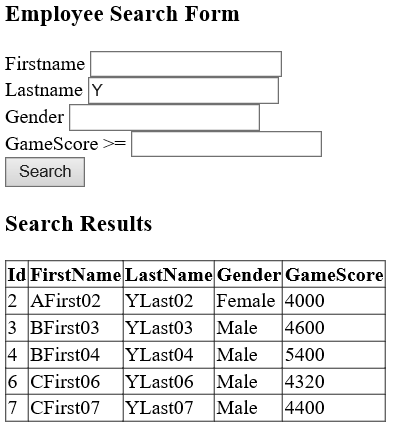
        }

    }

}

9. Test it

9.1. WebForm1.aspx, WebForm1.aspx.cs, spSearchGamer

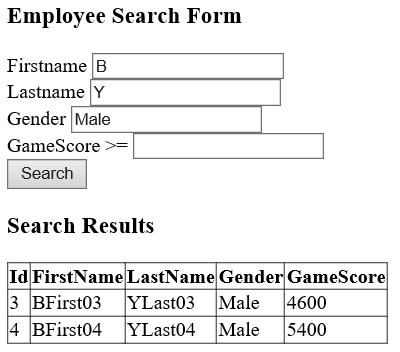


------------------------------------------------------------------------------------

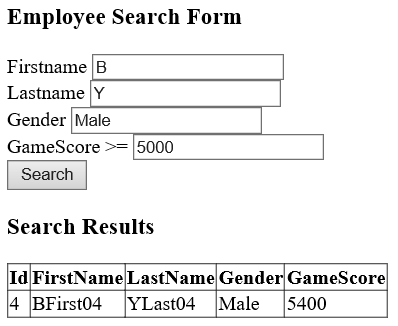
Table

Description automatically generated

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



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



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

Test SQL Injection

In any field, type in

**';DROP TABLE dbo.Table1; --**

This form is using spSearchGamer which is not using dynamic sql query.

Thus, the form has no sql injection issue.

9.2. WebForm2.aspx, WebForm2.aspx.cs, spSearchGamer2

Table

Description automatically generated

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

Table

Description automatically generated

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

Table

Description automatically generated

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

Table

Description automatically generated with low confidence

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

Test SQL Injection

In any field, type in

**';DROP TABLE dbo.Table1; --**

This form is using **spSearchGamer2** which is using dynamic sql query.

EXECUTE spSearchGamer2 @Gender = N'''; DROP TABLE dbo.Table1; --';

SELECT  \*

FROM    Table1;

GO -- Run the previous command and begins new batch

/\*

\*\*SQL Injection

The Table1 will be dropped.

\*/

**spSearchGamer2** has sql injection issue during the test in SQL server

But when we are using web form application

In any field, type in

**';DROP TABLE dbo.Table1; --**

The **dbo.Table1**will be dropped.

Table

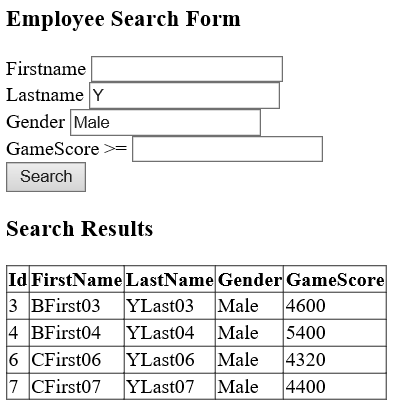
Description automatically generated

9.3. WebForm3.aspx, WebForm3.aspx.cs, spSearchGamer3

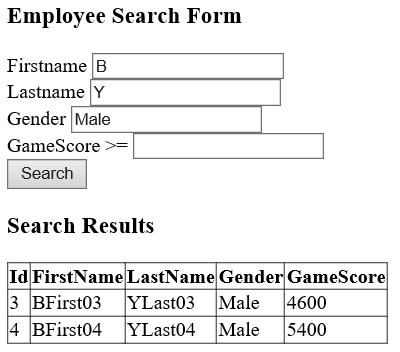
Table

Description automatically generated

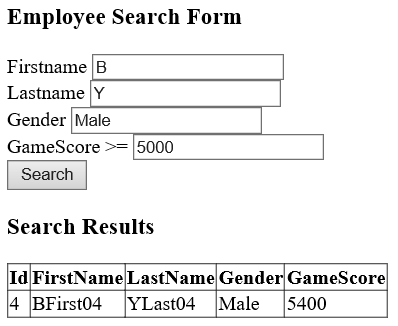
------------------------------------------------------------------------------------



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



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



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

Test SQL Injection

In any field, type in

**';DROP TABLE dbo.Table1; --**

This form is using **spSearchGamer3** which is using **dynamic sql** query **with parameters**.

Thus, the form has no sql injection issue.

====================================================================

10. Clean up

--=====================================================================

--T040\_07\_Clean up

--=====================================================================

IF ( EXISTS ( SELECT    \*

              FROM      INFORMATION\_SCHEMA.TABLES

              WHERE     TABLE\_NAME = 'Gamer' ) )

    BEGIN

        TRUNCATE TABLE dbo.Gamer;

        DROP TABLE Gamer;

    END;

GO -- Run the previous command and begins new batch

IF ( EXISTS ( SELECT    \*

              FROM      INFORMATION\_SCHEMA.TABLES

              WHERE     TABLE\_NAME = 'Table1' ) )

    BEGIN

        TRUNCATE TABLE dbo.Table1;

        DROP TABLE Table1;

    END;

GO -- Run the previous command and begins new batch

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

IF ( EXISTS ( SELECT    \*

              FROM      INFORMATION\_SCHEMA.ROUTINES

              WHERE     ROUTINE\_TYPE = 'PROCEDURE'

                        AND LEFT(ROUTINE\_NAME, 3) NOT IN ( 'sp\_', 'xp\_', 'ms\_' )

                        AND SPECIFIC\_NAME = 'spSearchGamer' ) )

    BEGIN

        DROP PROCEDURE spSearchGamer;

    END;

GO -- Run the previous command and begins new batch

IF ( EXISTS ( SELECT    \*

              FROM      INFORMATION\_SCHEMA.ROUTINES

              WHERE     ROUTINE\_TYPE = 'PROCEDURE'

                        AND LEFT(ROUTINE\_NAME, 3) NOT IN ( 'sp\_', 'xp\_', 'ms\_' )

                        AND SPECIFIC\_NAME = 'spSearchGamer2' ) )

    BEGIN

        DROP PROCEDURE spSearchGamer2;

    END;

GO -- Run the previous command and begins new batch

IF ( EXISTS ( SELECT    \*

              FROM      INFORMATION\_SCHEMA.ROUTINES

              WHERE     ROUTINE\_TYPE = 'PROCEDURE'

                        AND LEFT(ROUTINE\_NAME, 3) NOT IN ( 'sp\_', 'xp\_', 'ms\_' )

                        AND SPECIFIC\_NAME = 'spSearchGamer3' ) )

    BEGIN

        DROP PROCEDURE spSearchGamer3;

    END;

GO -- Run the previous command and begins new batch