(T17)LinqToSql的StoredProcedure的CRUD(Create、Read、Update、Delete)  
CourseGUID: 5ba9a6fe-7475-4b0c-8b99-bbcf7f5e2e1c  
=======================================================================  
(T17)LinqToSql的StoredProcedure的CRUD(Create、Read、Update、Delete)  
=======================================================================  
1. Web Form Application - Linq Query

1.1. TSQL

-----------

2.2. Sample.dbml

2.3. Sample.dbml - Store Procedure Return Type

2.4. Sample.dbml - Stored procedure

2.5. WebForm2.aspx

2.5.1. WebForm2.aspx

2.5.2. WebForm2.aspx.cs  
=======================================================================

1. Web Form Application - Linq Query

1.1. TSQL

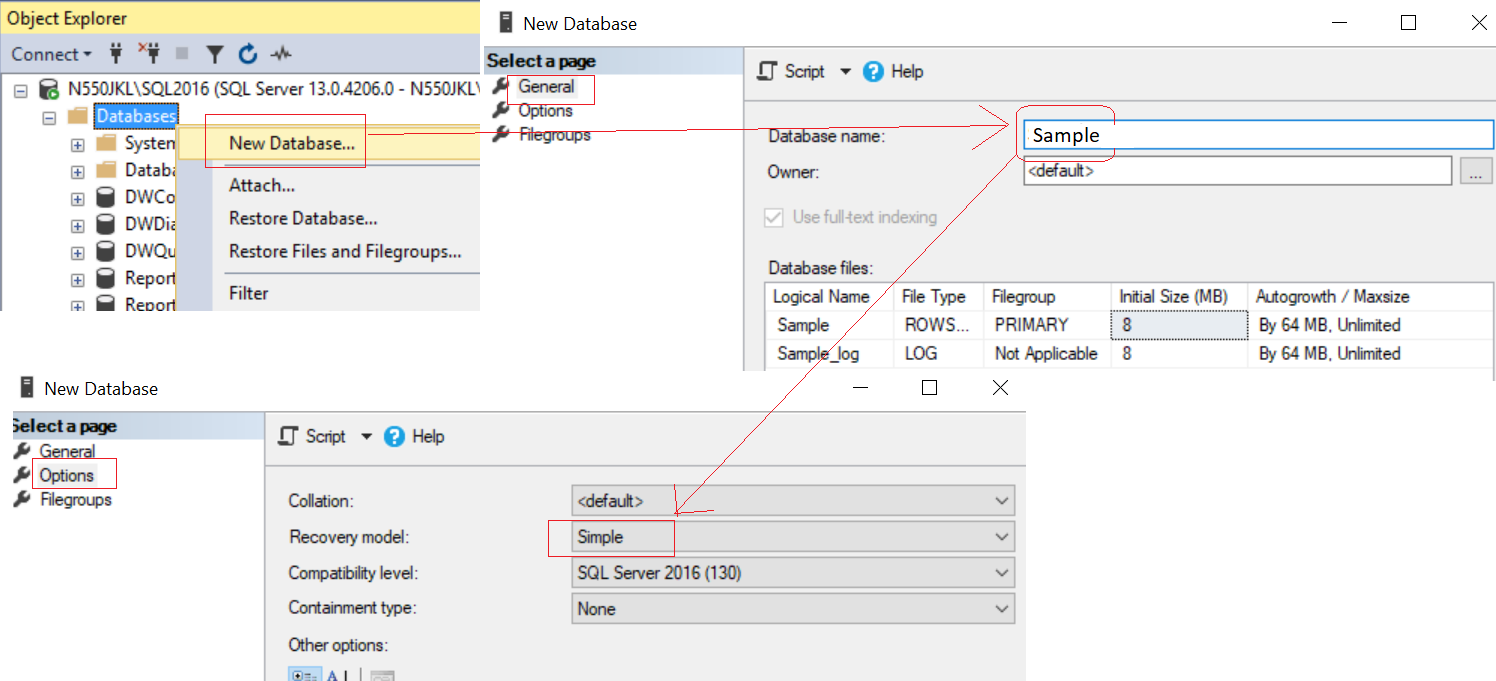
In previous tutorial, we created an Sample Database.

If you do not have the Sample Database, please following the instruction to create one.

Database --> Right Click --> New Database -->

Database Name : Sample

Options --> Recovery Model : Simple



--Create an Sample DataBase and Run the following TSQL

/\*

1.

One Team can have many Gamers

One Gamer can have One Team.

This is One to Many Relationship.

2.

Team Id==4 has no Gamer.

Gamer Id==7 has no Team.

\*/

--1 ----------------------------------------------------------

--Drop Table if it exists.

--IF OBJECT\_ID('Gamer') IS NOT NULL

IF ( EXISTS ( SELECT    \*

              FROM      INFORMATION\_SCHEMA.TABLES

              WHERE     TABLE\_NAME = 'Gamer' ) )

    BEGIN

        TRUNCATE TABLE Gamer;

        DROP TABLE Gamer;

    END;

GO -- Run the previous command and begins new batch

--Drop Table if it exists.

--IF OBJECT\_ID('Team') IS NOT NULL

IF ( EXISTS ( SELECT    \*

              FROM      INFORMATION\_SCHEMA.TABLES

              WHERE     TABLE\_NAME = 'Team' ) )

    BEGIN

        TRUNCATE TABLE Team;

        DROP TABLE Team;

    END;

GO -- Run the previous command and begins new batch

--Create Tables

CREATE TABLE Team

    (

      Id INT PRIMARY KEY

             IDENTITY ,

      Name NVARCHAR(100) ,

      Type NVARCHAR(100)

    );

GO -- Run the previous command and begins new batch

CREATE TABLE Gamer

    (

      Id INT PRIMARY KEY

             IDENTITY ,

      Name NVARCHAR(50) ,

      Gender NVARCHAR(50) ,

      Score INT ,

      Type NVARCHAR(50) ,

      TeamId INT FOREIGN KEY REFERENCES Team ( Id )

    );

GO -- Run the previous command and begins new batch

--2 ----------------------------------------------------------

--Insert Data

INSERT  INTO Team

VALUES  ( 'Team1\_Guardian', 'Guardian' );

INSERT  INTO Team

VALUES  ( 'Team2\_Assassinator', 'Assassinator' );

INSERT  INTO Team

VALUES  ( 'Team3\_Soldier', 'Soldier' );

INSERT  INTO Team

VALUES  ( 'Team4\_Civilian', 'Civilian' );

GO -- Run the previous command and begins new batch

INSERT  INTO Gamer

VALUES  ( 'Name1 ABC', 'Male', 5000, 'Water', 1 );

INSERT  INTO Gamer

VALUES  ( 'Name2 ABCDE', 'Female', 4500, 'Fire', 3 );

INSERT  INTO Gamer

VALUES  ( 'Name3 EFGH', 'Male', 6500, 'Fire', 2 );

INSERT  INTO Gamer

VALUES  ( 'Name4 HIJKLMN', 'Female', 45000, 'Water', 2 );

INSERT  INTO Gamer

VALUES  ( 'Name5 NOP', 'Male', 3000, 'Wood', 1 );

INSERT  INTO Gamer

VALUES  ( 'Name6 PQRSTUVW', 'Male', 4000, 'Earth', 3 );

INSERT  INTO Gamer

VALUES  ( 'Name7 XYZ', 'Male', 4500, 'Metal', NULL );

GO -- Run the previous command and begins new batch

--3 ----------------------------------------------------------

--3.1. --------------------------------

--Drop Stored Procedure if it exists.

--IF OBJECT\_ID('spGetGamers') IS NOT NULL

IF ( EXISTS ( SELECT    \*

              FROM      INFORMATION\_SCHEMA.ROUTINES

              WHERE     ROUTINE\_TYPE = 'PROCEDURE'

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

                        AND SPECIFIC\_NAME = 'spGetGamers' ) )

    BEGIN

        DROP PROCEDURE spGetGamers;

    END;

GO -- Run the previous command and begins new batch

CREATE PROCEDURE spGetGamers

AS

    BEGIN

        SELECT  Id ,

                Name ,

                Gender ,

                Score ,

                Type ,

                TeamId

        FROM    Gamer;

    END;

GO -- Run the previous command and begins new batch

--3.2. --------------------------------

-- Update Stored Procedure

--Drop Stored Procedure if it exists.

--IF OBJECT\_ID('spInsertGamer') IS NOT NULL

IF ( EXISTS ( SELECT    \*

              FROM      INFORMATION\_SCHEMA.ROUTINES

              WHERE     ROUTINE\_TYPE = 'PROCEDURE'

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

                        AND SPECIFIC\_NAME = 'spInsertGamer' ) )

    BEGIN

        DROP PROCEDURE spInsertGamer;

    END;

GO -- Run the previous command and begins new batch

CREATE PROCEDURE spInsertGamer

    @name NVARCHAR(50) ,

    @gender NVARCHAR(50) ,

    @score INT ,

    @type NVARCHAR ,

    @teamId int

AS

    BEGIN

        INSERT  INTO Gamer

        VALUES  ( @name, @gender, @score, @type, @teamId );

    END;

GO

--3.3. --------------------------------

-- Update Stored Procedure

--Drop Stored Procedure if it exists.

--IF OBJECT\_ID('spUpdateGamer') IS NOT NULL

IF ( EXISTS ( SELECT    \*

              FROM      INFORMATION\_SCHEMA.ROUTINES

              WHERE     ROUTINE\_TYPE = 'PROCEDURE'

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

                        AND SPECIFIC\_NAME = 'spUpdateGamer' ) )

    BEGIN

        DROP PROCEDURE spUpdateGamer;

    END;

GO -- Run the previous command and begins new batch

CREATE PROCEDURE spUpdateGamer

    @Id INT ,

    @name NVARCHAR(50) ,

    @gender NVARCHAR(50) ,

    @score INT ,

    @type NVARCHAR ,

    @teamId int

AS

    BEGIN

        UPDATE  Gamer

        SET     Name = @name ,

                Gender = @gender ,

                Score = @score ,

                Type = @type ,

                TeamId = @teamId

        WHERE   Id = @Id;

    END;

GO -- Run the previous command and begins new batch

--3.4. --------------------------------

-- Delete Stored Procedure

--Drop Stored Procedure if it exists.

--IF OBJECT\_ID('spDeleteGamer') IS NOT NULL

IF ( EXISTS ( SELECT    \*

              FROM      INFORMATION\_SCHEMA.ROUTINES

              WHERE     ROUTINE\_TYPE = 'PROCEDURE'

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

                        AND SPECIFIC\_NAME = 'spDeleteGamer' ) )

    BEGIN

        DROP PROCEDURE spDeleteGamer;

    END;

GO -- Run the previous command and begins new batch

CREATE PROCEDURE spDeleteGamer @Id int

AS

    BEGIN

        DELETE  FROM dbo.Gamer

        WHERE   Id = @Id;

    END;

GO

--3.5. --------------------------------

--3.5.1. --------------------------------

-- Delete Stored Procedure

--Drop Stored Procedure if it exists.

--IF OBJECT\_ID('spGetGamerByTeam') IS NOT NULL

IF ( EXISTS ( SELECT    \*

              FROM      INFORMATION\_SCHEMA.ROUTINES

              WHERE     ROUTINE\_TYPE = 'PROCEDURE'

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

                        AND SPECIFIC\_NAME = 'spGetGamerByTeam' ) )

    BEGIN

        DROP PROCEDURE spGetGamerByTeam;

    END;

GO -- Run the previous command and begins new batch

CREATE PROCEDURE spGetGamerByTeam

    @teamId INT ,

    @teamName NVARCHAR(100) OUT

AS

    BEGIN

        SELECT  @teamName = Name

        FROM    dbo.Team

        WHERE   Id = @teamId;

        SELECT  \*

        FROM    dbo.Gamer

        WHERE   TeamId = @teamId;

    END;

GO

--3.5.2. --------------------------------

--Test

DECLARE @teamName NVARCHAR(50);

EXECUTE spGetGamerByTeam 1, @teamName OUT;

SELECT  @teamName;

2.2. Sample.dbml

In previous tutorial, we added connection of Sample DB in Server Explore,

Please read the previous tutorial before you continue.

In previous tutorial, we created Sample.dbml.

We also created Team and Gamer Entity to dbml.

If you have no Sample.dbml, please following the following the instruction to create one.

ProjectName --> Right Click --> Add --> New Item...

--> Linq to SQL classes -->

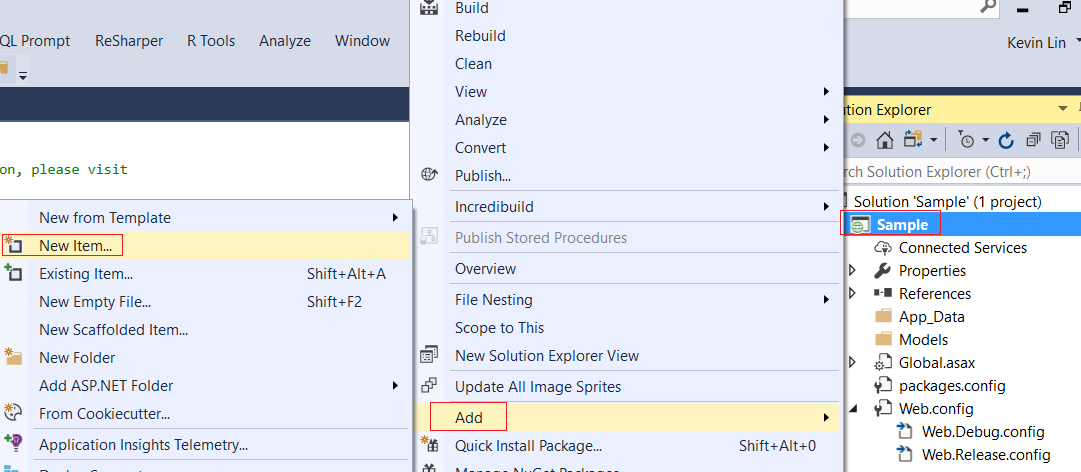
Name : **Sample.dbml**

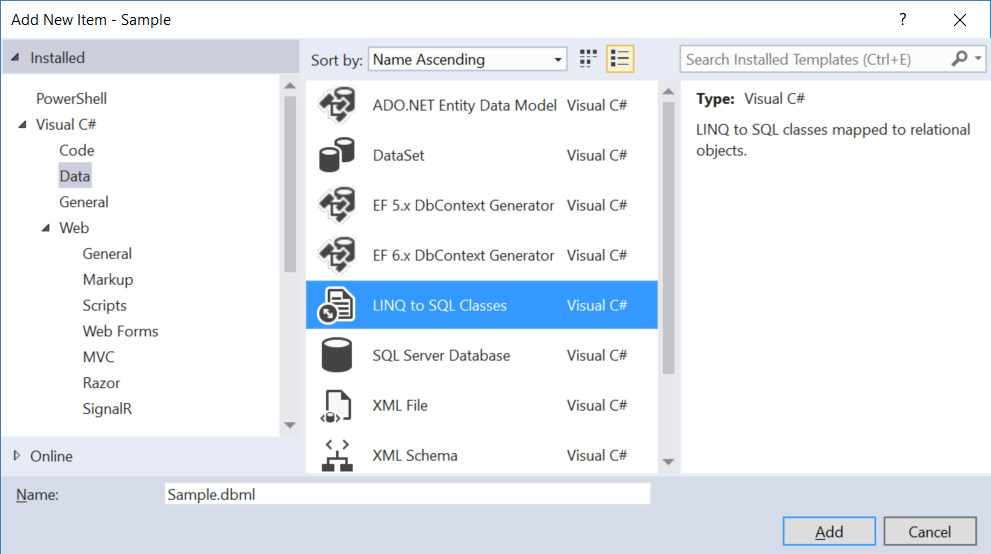
I name it as "Sample.dbml",

because I know this is for connection to "Sample" Database.

-->

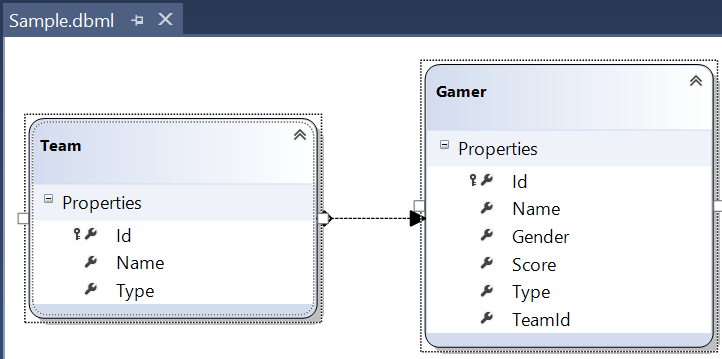
Drag Table from Server Explorer into DBML





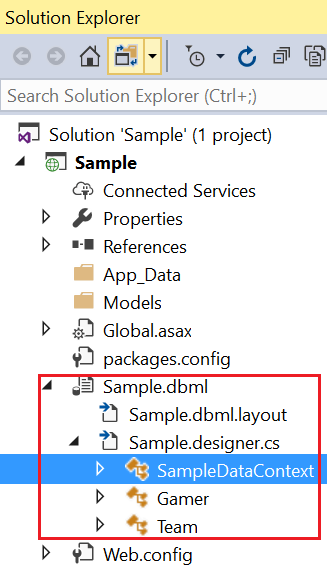
Graphical user interface, application

Description automatically generated



Save the dbml, it will generate the following files.

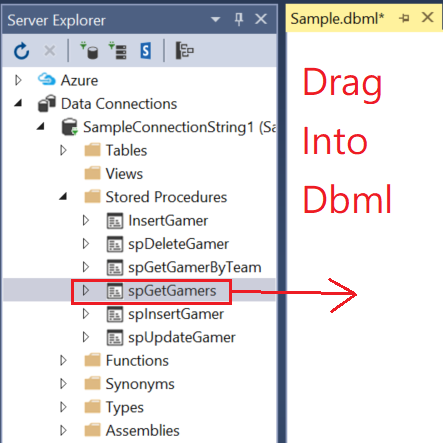
The DataContext context is the entry point to database.

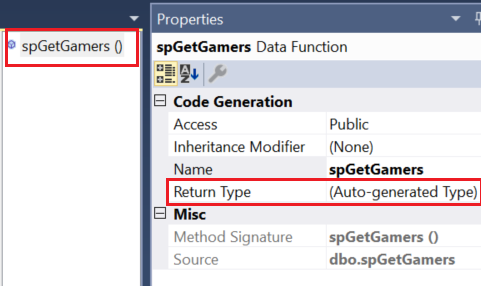


2.3. Sample.dbml - Store Procedure Return Type

We have just revised the previous tutorial.

We are going to add the store procedure into dbml.





By default, it will create the following method in **Sample.designer.cs**

The return type is ISingleResult<spGetGamersResult>

[global::System.Data.Linq.Mapping.FunctionAttribute(Name="dbo.spGetGamers")]

public ISingleResult<spGetGamersResult> spGetGamers()

{

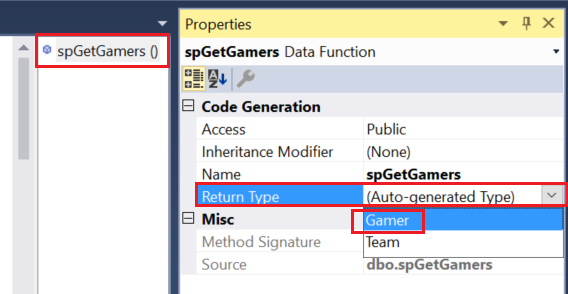
       IExecuteResult result = this.ExecuteMethodCall(this, ((MethodInfo)(MethodInfo.GetCurrentMethod())));

       return ((ISingleResult<spGetGamersResult>)(result.ReturnValue));

}

However, there are 2 popular ways to change the return type from

ISingleResult<spGetGamersResult>  to Gamer Type



Please note, change type can not be undo.

Graphical user interface, text, application

Description automatically generated

It will generate the following code in **Sample.designer.cs**

The return type is now Gamer.

[global::System.Data.Linq.Mapping.FunctionAttribute(Name="dbo.spGetGamers")]

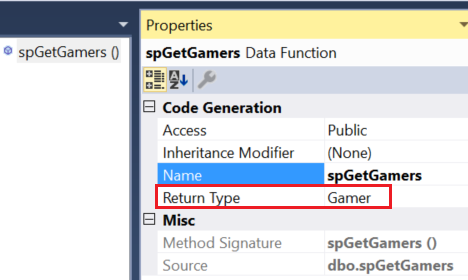
public ISingleResult<Gamer> spGetGamers()

{

       IExecuteResult result = this.ExecuteMethodCall(this, ((MethodInfo)(MethodInfo.GetCurrentMethod())));

       return ((ISingleResult<Gamer>)(result.ReturnValue));

}

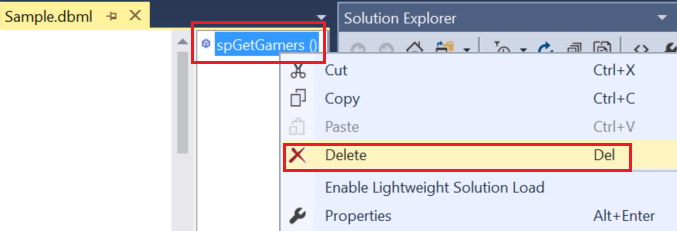


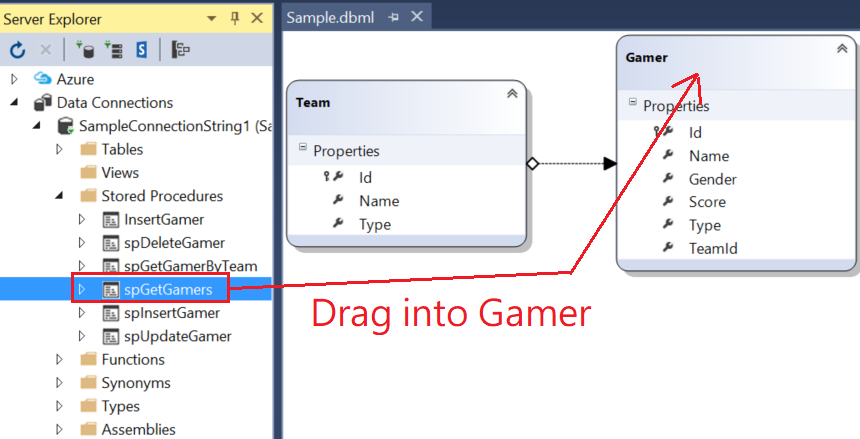
Let's try the second way to set the return type to Gamer

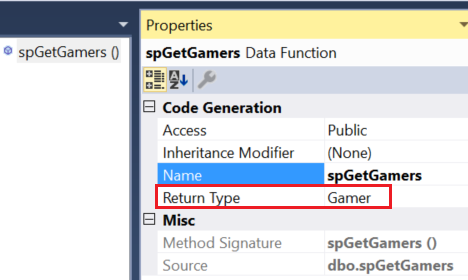
Firstly, you need to delete the method.

Then you may drag the stored procedure into Gamer.

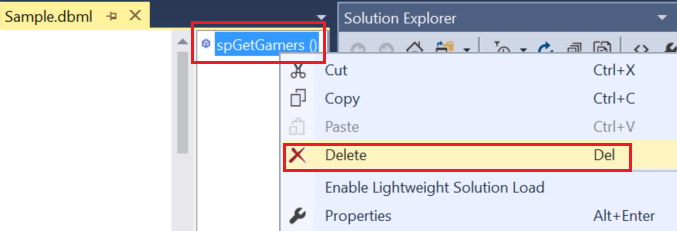
The return type will be Gamer.





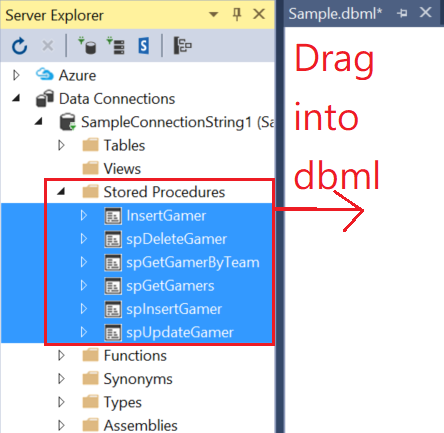


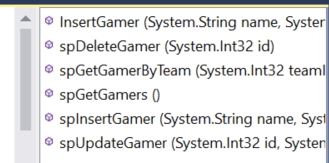
Please delete it before you continue this tutorial.



2.4. Sample.dbml - Stored procedure

Drag All stored procedures into dbml





In dbml

Gamer --> Right click --> Configure Behavior

-->

Behavior : Insert

Customize : spInsertGamer

Please confirm the mapping is right.

Gamer --> Right click --> Configure Behavior

-->

Behavior : Update

Customize : spUpdateGamer

Please confirm the mapping is right.

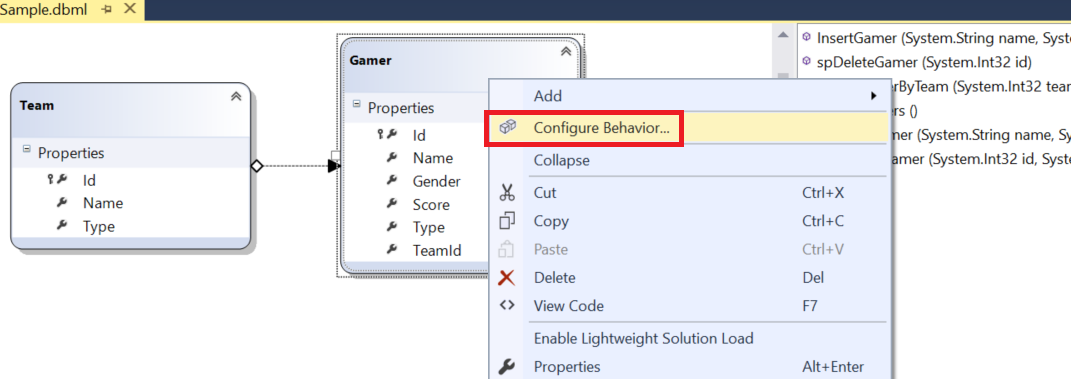
Gamer --> Right click --> Configure Behavior

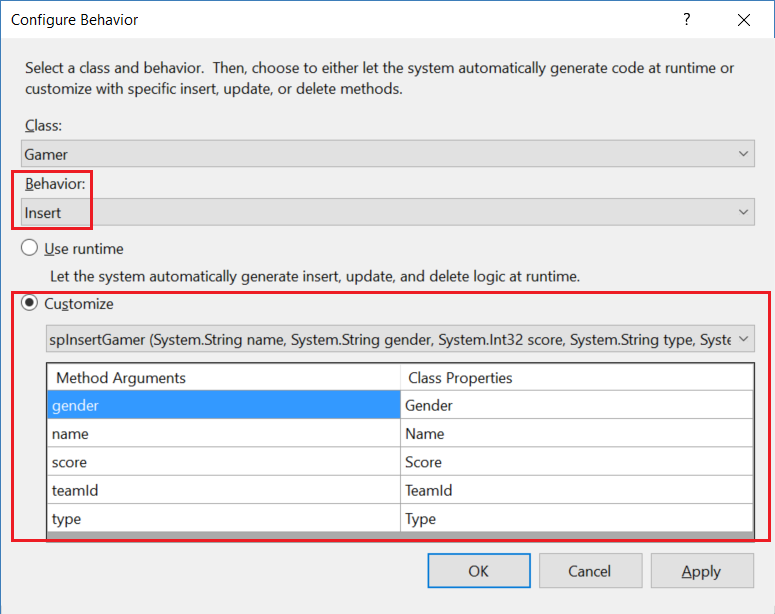
-->

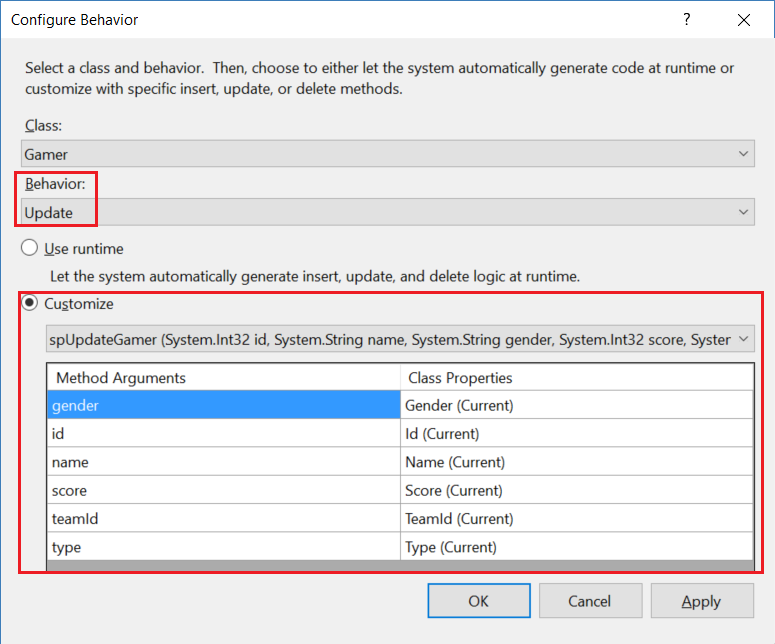
Behavior : Delete

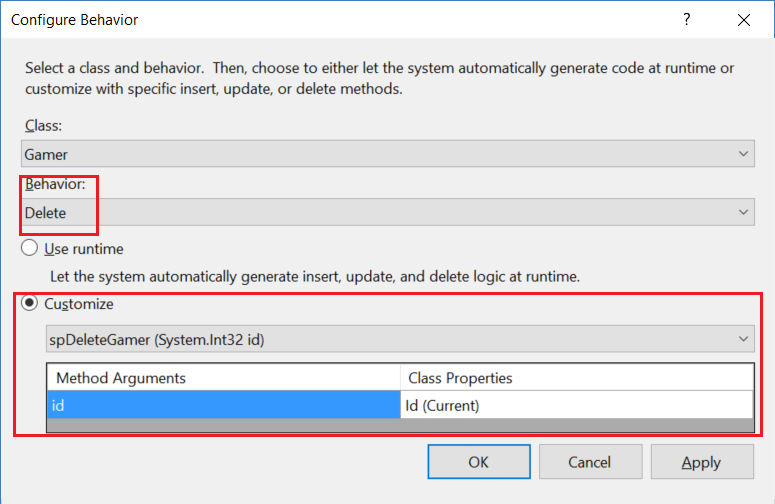
Customize : spDeleteGamer

Please confirm the mapping is right.









2.5. WebForm2.aspx

2.5.1. WebForm2.aspx

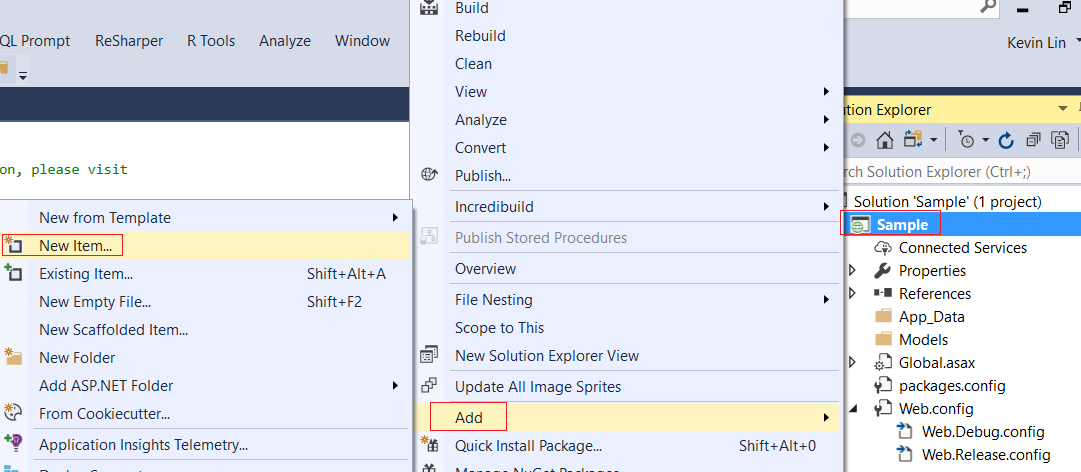
ProjectName --> Right Click --> Add --> New Item...

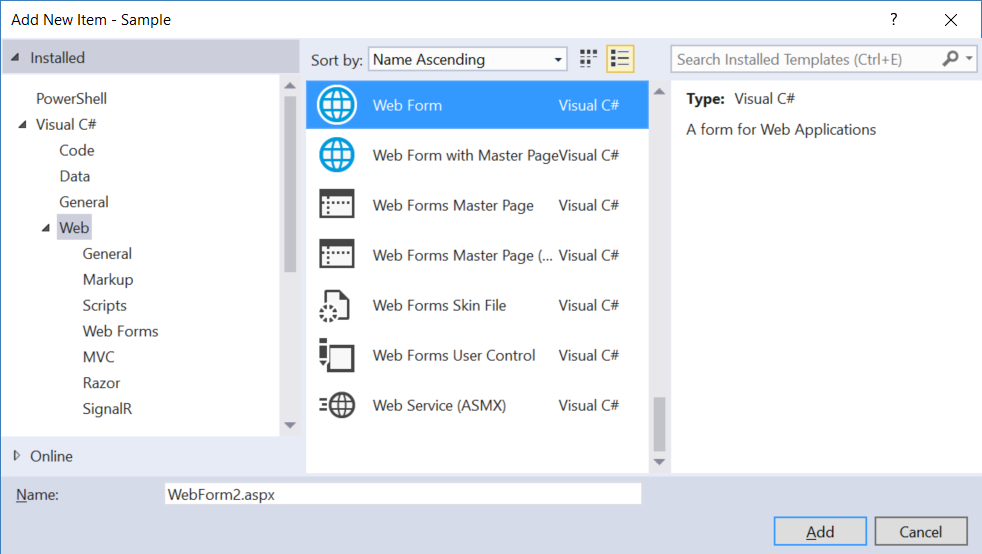
-->

**WebForm**

Name :

**WebForm2.aspx**





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

<!DOCTYPE html>

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

<head runat="server">

    <title></title>

</head>

<body>

    <form id="form1" runat="server">

        <div>

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

            <asp:GridView ID="GridView1" runat="server"></asp:GridView>

            <asp:Button ID="btnGetAllGamer" runat="server" Text="Get All Gamers" OnClick="btnGetAllGamer\_Click" />

            <asp:Button ID="btnInsertGamer" runat="server" Text="Insert Gamer" OnClick="btnInsertGamer\_Click" />

            <asp:Button ID="btnUpdateLastGamer" runat="server" Text="Update Last Gamer" OnClick="btnUpdateLastGamer\_Click" />

            <asp:Button ID="btnDeleteLastGamer" runat="server" Text="Delete Last Gamer" OnClick="btnDeleteLastGamer\_Click" />

            <asp:Button ID="btnGamersByTeam1" runat="server" Text="Gamer Gamers by Team1" OnClick="btnGamersByTeam1\_Click" />

        </div>

    </form>

</body>

</html>

2.5.2. WebForm2.aspx.cs

using System;

using System.Linq;

namespace Sample

{

    public partial class WebForm2 : System.Web.UI.Page

    {

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

        {

        }

        private void GetData()

        {

            using (SampleDataContext dbContext = new SampleDataContext())

            {

                IQueryable<Gamer> gamerQueryable =

                    from gamer in dbContext.Gamers

                    select gamer;

                GridView1.DataSource = gamerQueryable;

                GridView1.DataBind();

            }

        }

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

        {

            GetData();

        }

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

        {

            using (SampleDataContext dbContext = new SampleDataContext())

            {

                Gamer newGamer = new Gamer

                {

                    Name = "newGamer",

                    Gender = "Male",

                    Score = 4000,

                    Type = "Fire",

                    TeamId = 1

                };

                dbContext.Gamers.InsertOnSubmit(newGamer);  //insert into dbContext

                dbContext.SubmitChanges();  //Submit dbContext into Database

            }

            GetData();

        }

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

        {

            using (SampleDataContext dbContext = new SampleDataContext())

            {

                //Get the last gamer

                //1.

                //Gamer gamer = dbContext.Gamers.Last();

                //// Error! Last is not support..

                //2.

                //int lastId = dbContext.Gamers.Count();

                //Gamer gamer = dbContext.Gamers.SingleOrDefault(

                //    x => x.Id == lastId);

                //// Wrong logic, sometimes Id=1,2,4.. (3 is missing, because someone delete it)

                Gamer gamer =

                    dbContext.Gamers

                    .OrderByDescending(g => g.Id)

                    .FirstOrDefault();

                if (gamer != null) gamer.Score = 5555;

                dbContext.SubmitChanges();

            }

            GetData();

        }

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

        {

            using (SampleDataContext dbContext = new SampleDataContext())

            {

                //Get the last gamer

                //1.

                //Gamer gamer = dbContext.Gamers.Last();

                //// Error! Last is not support..

                //2.

                //int lastId = dbContext.Gamers.Count();

                //Gamer gamer = dbContext.Gamers.SingleOrDefault(

                //    x => x.Id == lastId);

                //// Wrong logic, sometimes Id=1,2,4.. (3 is missing, because someone delete it)

                Gamer gamer =

                    dbContext.Gamers

                    .OrderByDescending(g => g.Id)

                    .FirstOrDefault();

                //delete the last gamer from dbContext

                if (gamer != null) dbContext.Gamers.DeleteOnSubmit(gamer);

                dbContext.SubmitChanges();  // Save dbContext into Database.

            }

            GetData();

        }

        protected void btnGamersByTeam1\_Click(object sender, EventArgs e)

        {

            using (SampleDataContext dbContext = new SampleDataContext())

            {

                string teamName = string.Empty;

                GridView1.DataSource = dbContext.spGetGamerByTeam(1, ref teamName);

                GridView1.DataBind();

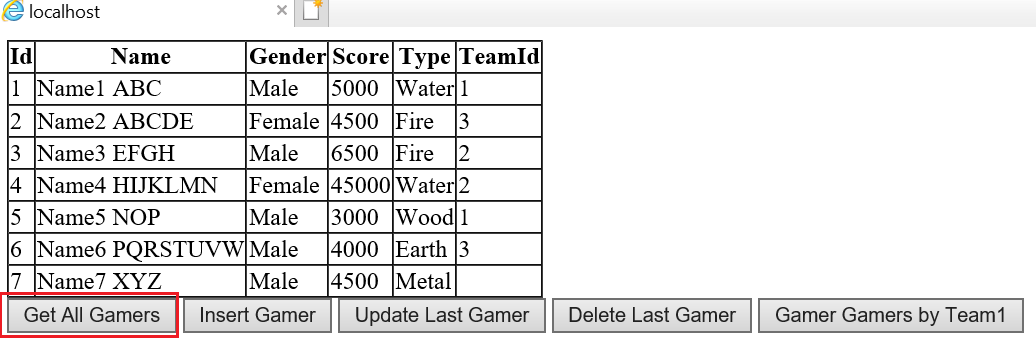
                lblTeamName.Text = $"TeamName=={teamName}";

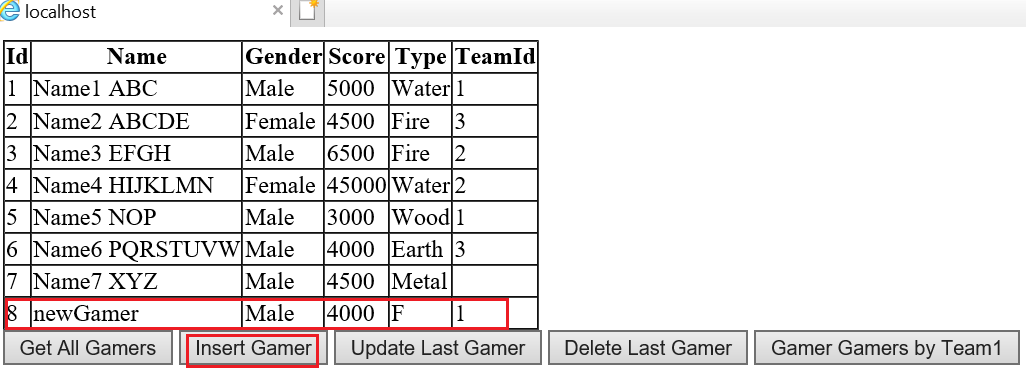
            }

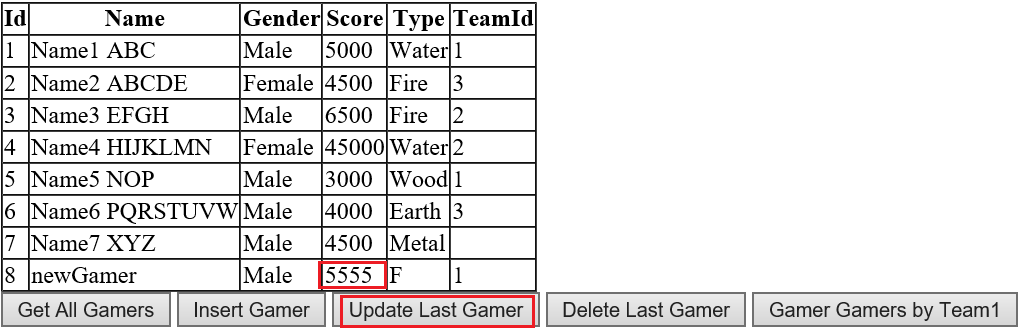
        }

    }

}







Table

Description automatically generated

Graphical user interface

Description automatically generated with medium confidence