(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)

==

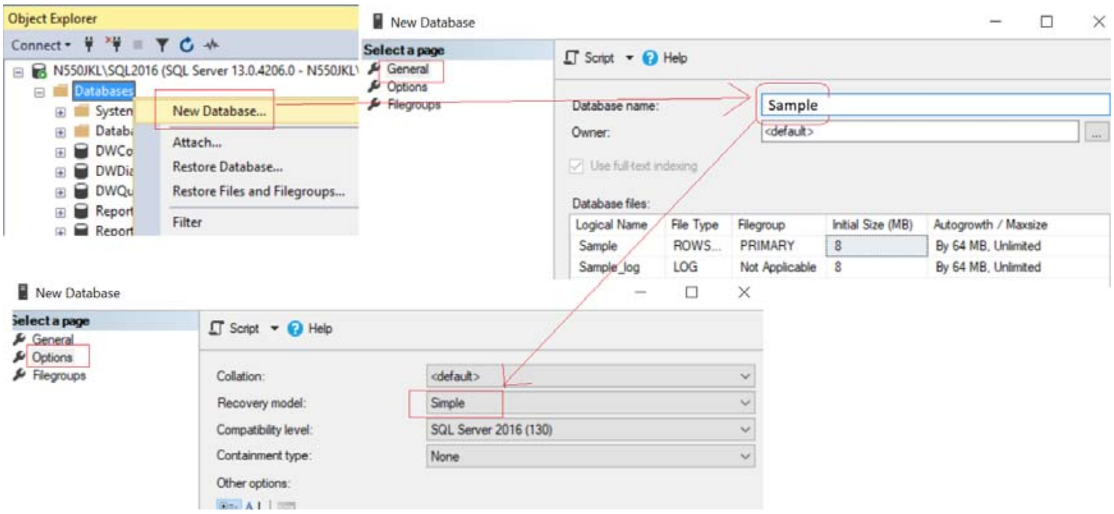
==

1. Web Form Application - Linq Query

1.1. TSQL

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



Database name: Sample
Owner: <default>
☑ Use full-text indexing

Database files:

Logical Name	File Type	Filegroup	Initial Size (MB)	Autogrowth / Maxsize
Sample	ROWS...	PRIMARY	8	By 64 MB, Unlimited
Sample_log	LOG	Not Applicable	8	By 64 MB, Unlimited

Collation: <default>
Recovery model: Simple
Compatibility level: SQL Server 2016 (130)
Containment type: None
Other options:

--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.

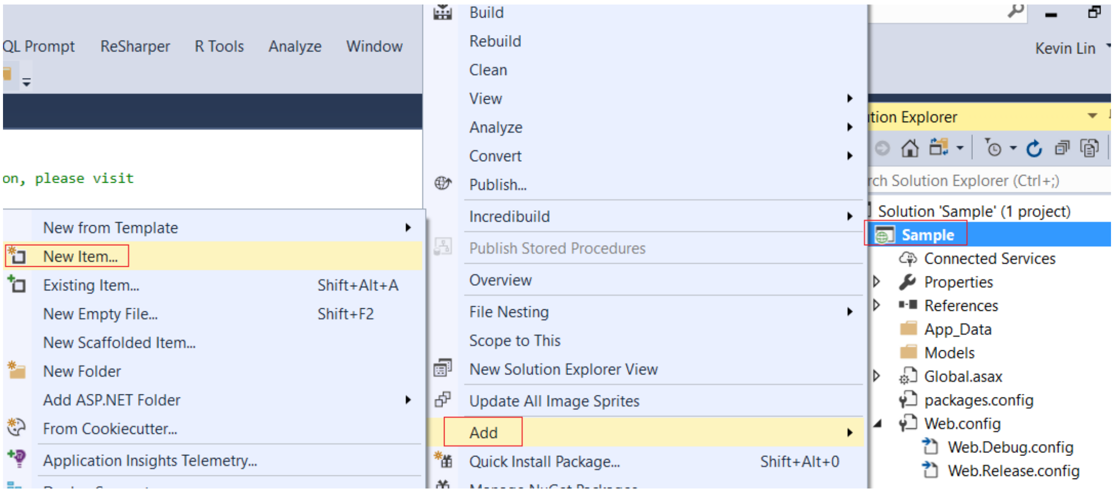
In previous tutorial, we created Sample.dbml.
We also created Team and Gamer Entity to dbml.
If you have no Sample.dbml, please following the following the instruction to create one.

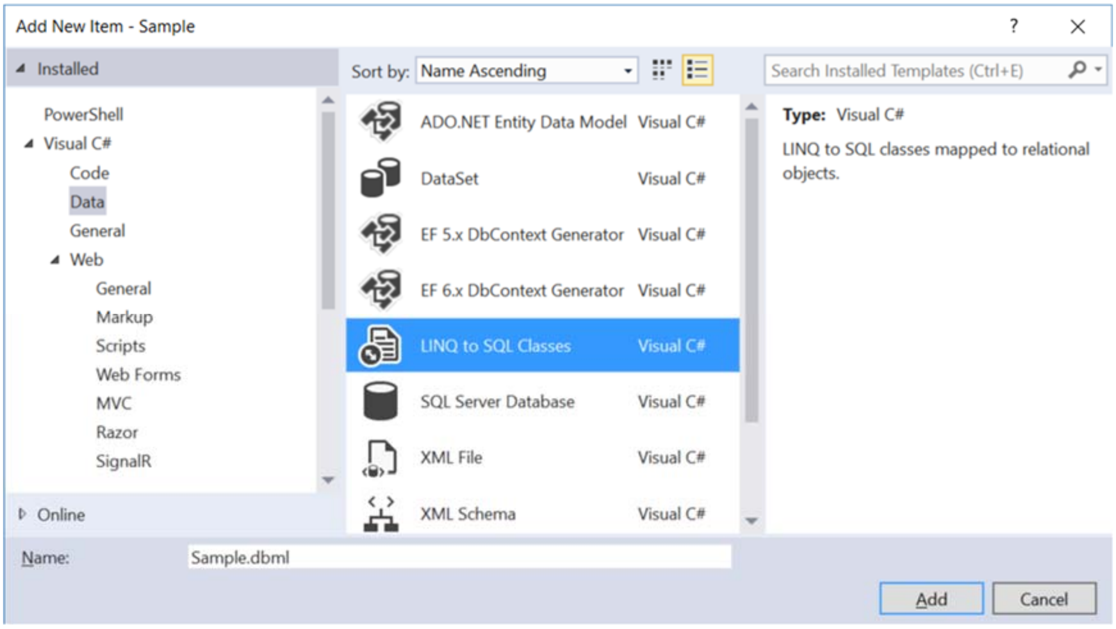
ProjectName --> Right Click --> Add --> New Item...
--> Linq to SQL classes -->
Name : **Sample.dbml**

I name it as "Sample.dbml",

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

-->
Drag Table from Server Explorer into DBML

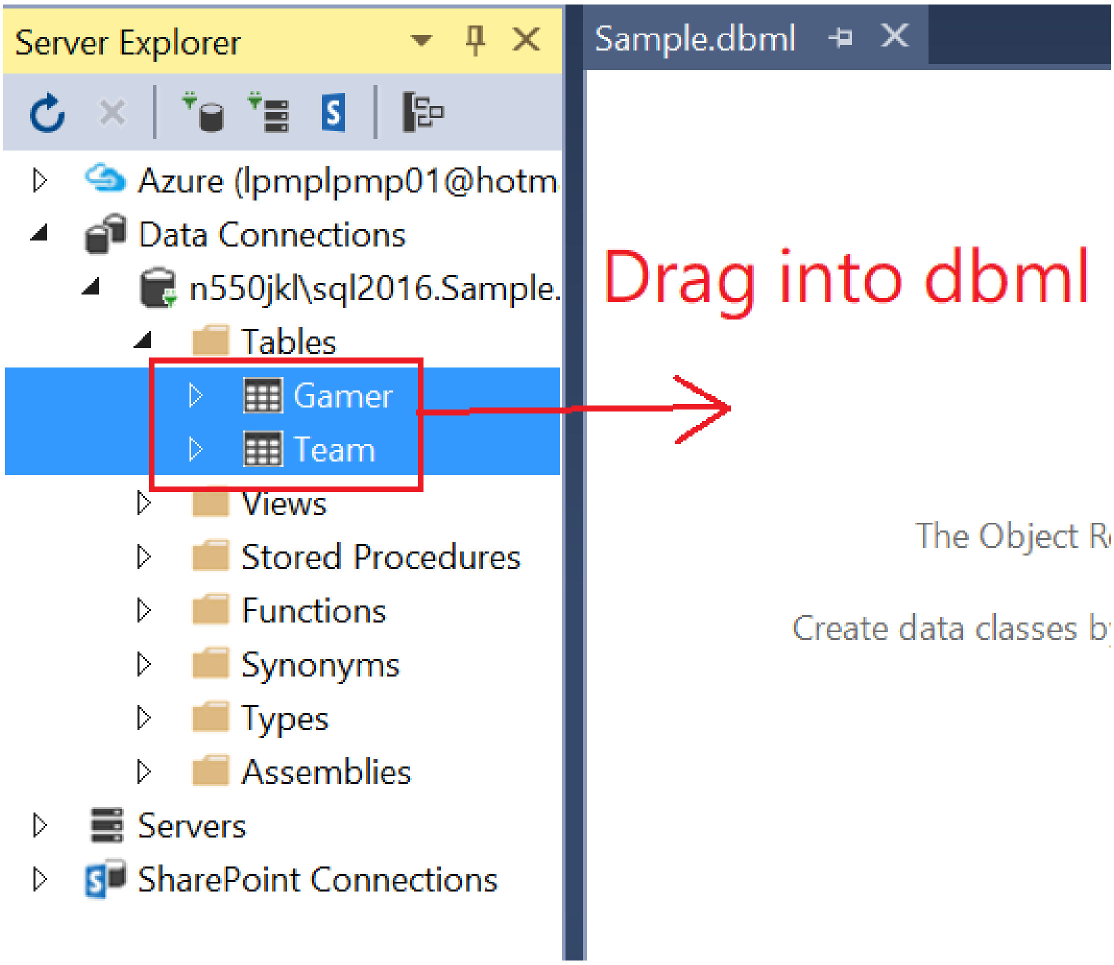


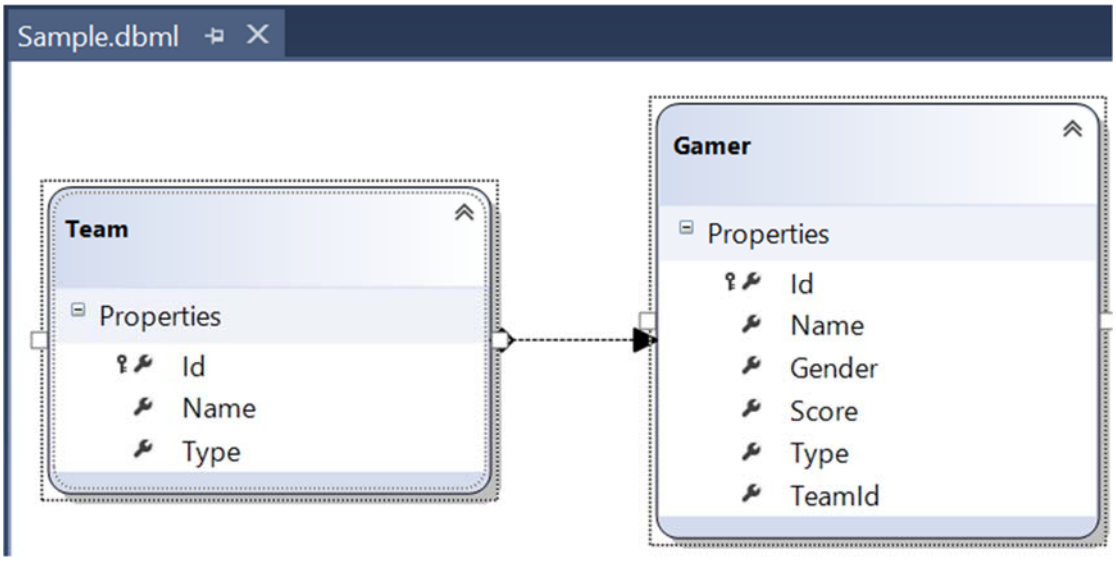


Type: Visual C#

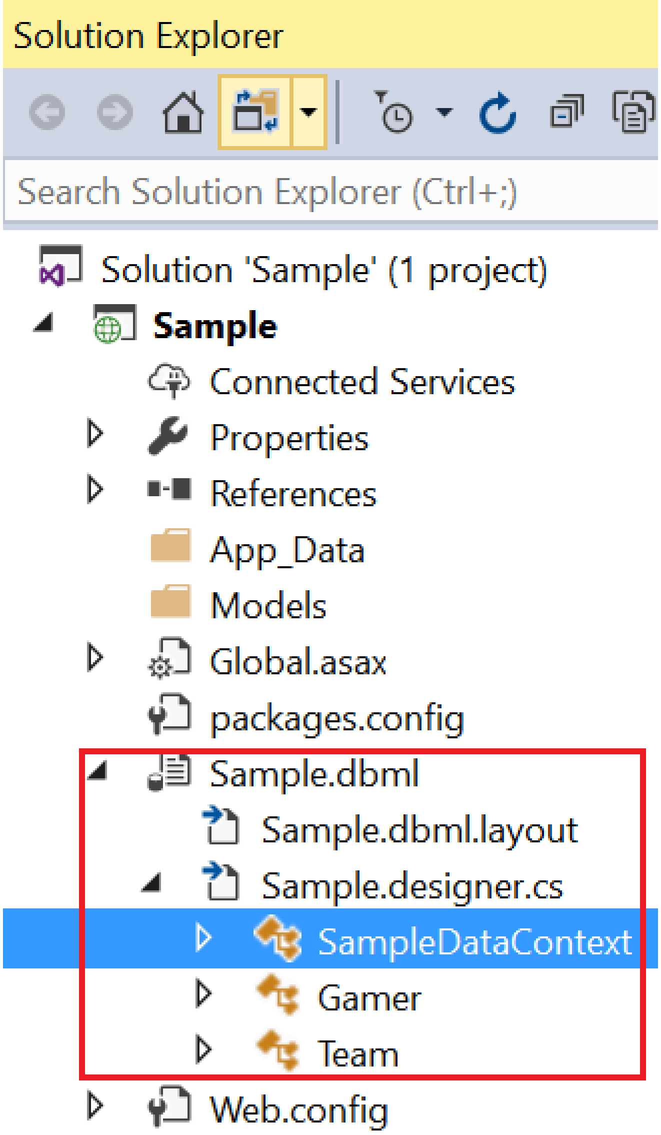
LINQ to SQL classes mapped to relational objects.

Name: Sample.dbml





Save the dbml, it will generate the following files.
The DataContext context is the entry point to database.



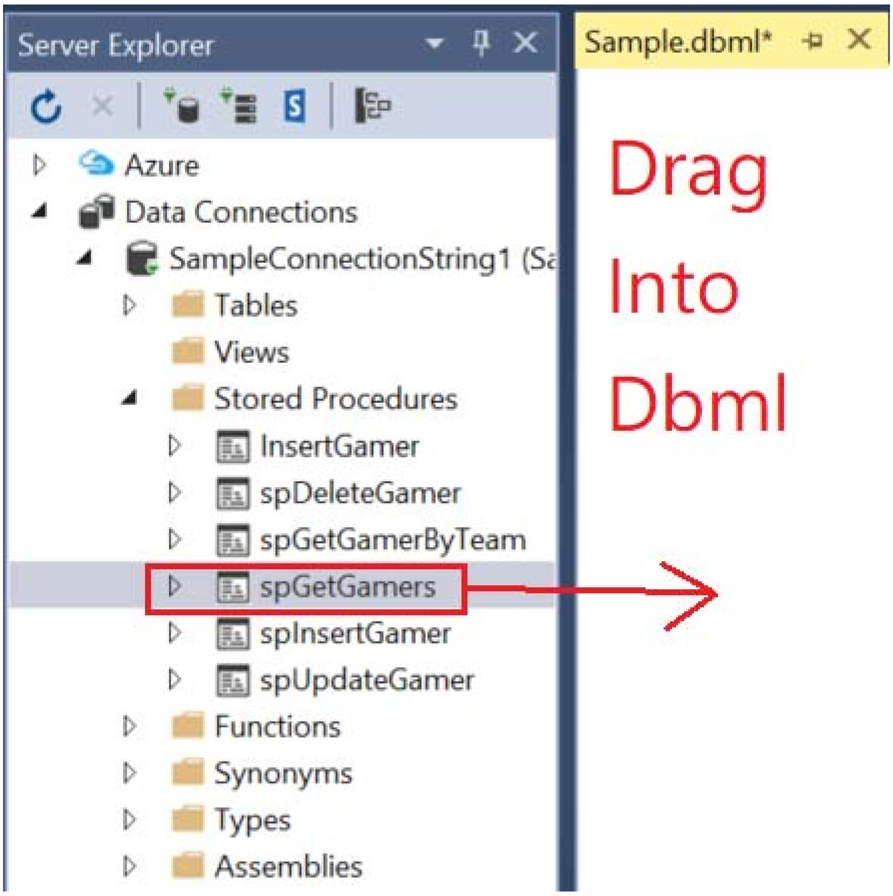
Solution Explorer

Search Solution Explorer (Ctrl+;)

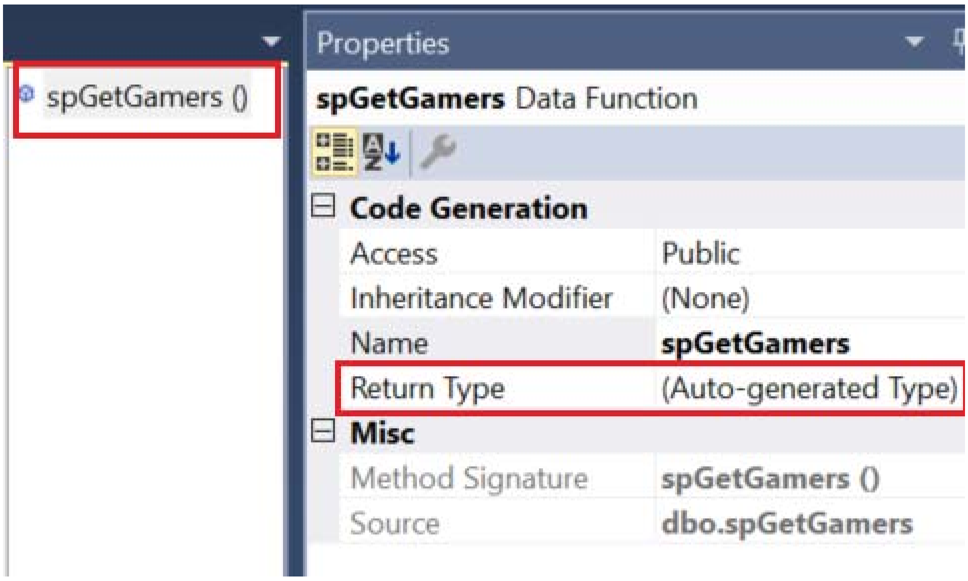
```
Solution 'Sample' (1 project)
◢ Sample
    Connected Services
  ▷ Properties
  ▷ References
    App_Data
    Models
  ▷ Global.asax
    packages.config
  ◢ Sample.dbml
      Sample.dbml.layout
    ◢ Sample.designer.cs
      ▷ SampleDataContext
      ▷ Gamer
      ▷ Team
  ▷ Web.config
```

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.



Drag
Into
Dbml



spGetGamers Data Function

Code Generation

Access	Public
Inheritance Modifier	(None)
Name	**spGetGamers**
Return Type	(Auto-generated Type)

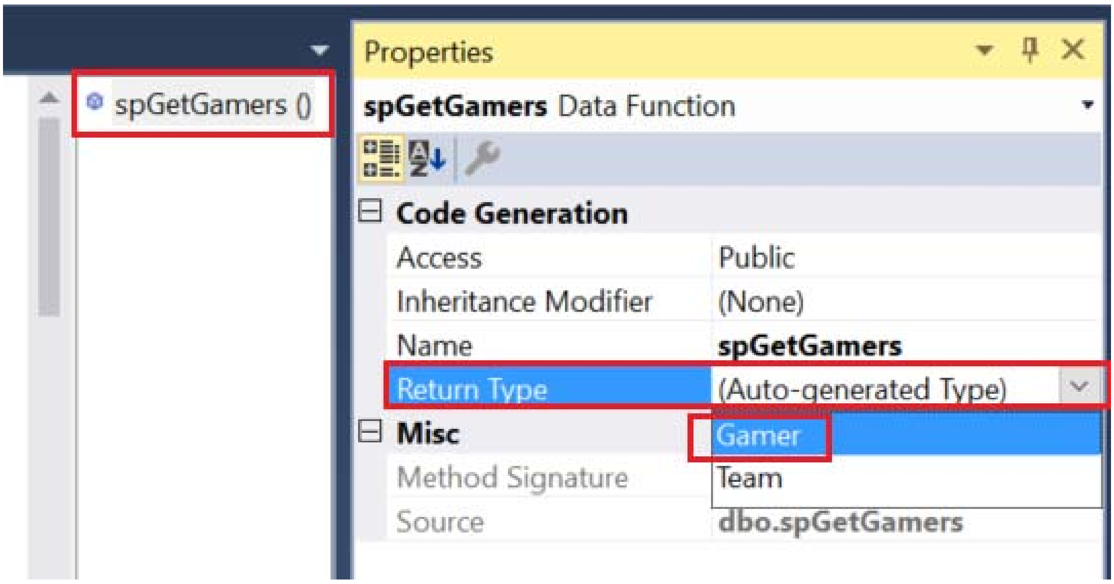
Misc

Method Signature	spGetGamers ()
Source	dbo.spGetGamers

By default, it will create the following method in **Sample.designer.cs**
The return type is `ISingleResult<spGetGamersResult>`

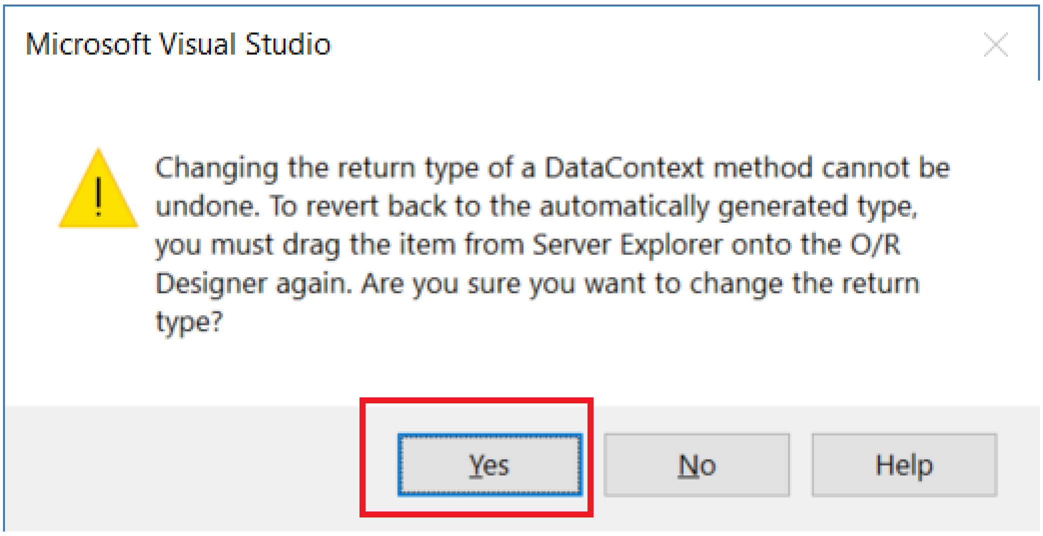
```
[global::System.Data.Linq.Mapping.FunctionAttribute(Name="dbo.spGetGamers")]
public ISingleResult<spGetGamersResult> spGetGamers()
{
        IExecuteResult result = this.ExecuteMethodCall(this,
((MethodInfo)(MethodInfo.GetCurrentMethod())));
        return ((ISingleResult<spGetGamersResult>)(result.ReturnValue));
}
```

However, there are 2 popular ways to change the return type from `ISingleResult<spGetGamersResult>` to Gamer Type



Access: Public
Inheritance Modifier: (None)
Name: spGetGamers
Return Type: (Auto-generated Type)
Method Signature
Source: dbo.spGetGamers

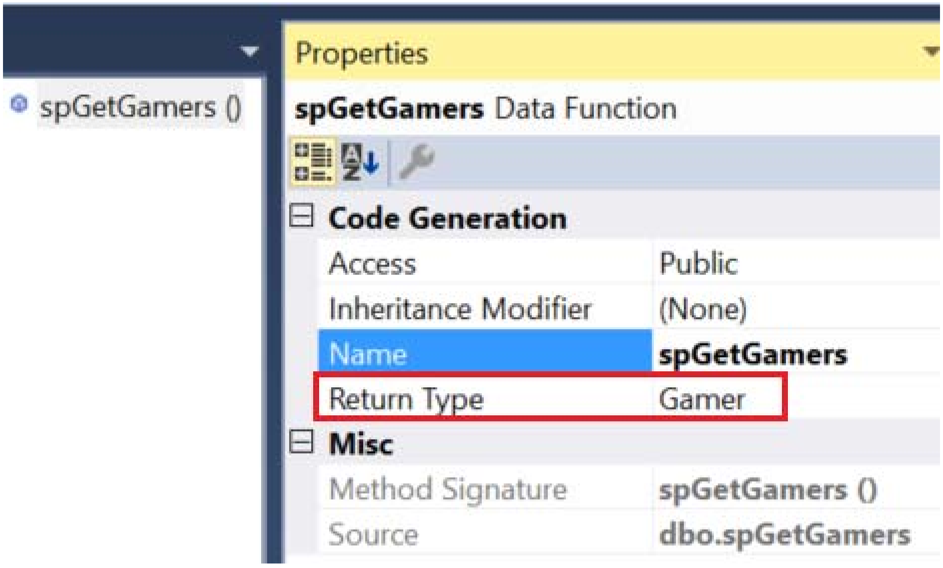
Please note, change type can not be undo.


Microsoft Visual Studio

Changing the return type of a DataContext method cannot be undone. To revert back to the automatically generated type, you must drag the item from Server Explorer onto the O/R Designer again. Are you sure you want to change the return type?

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));
}
```



spGetGamers Data Function

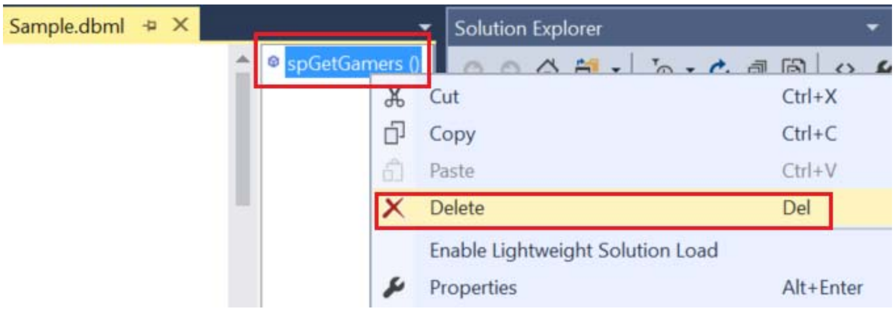
Code Generation

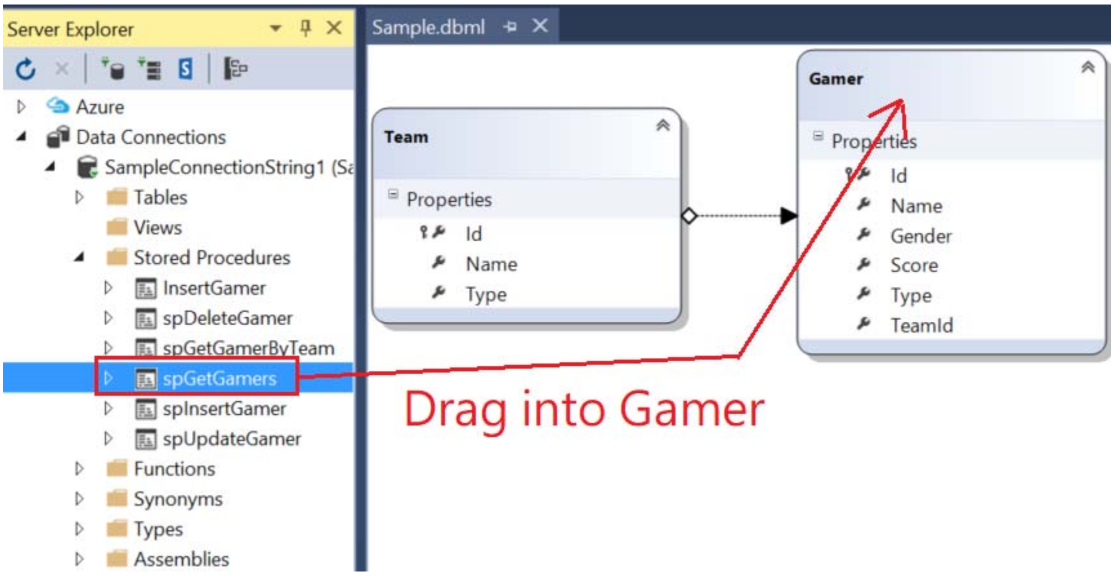
Access	Public
Inheritance Modifier	(None)
Name	spGetGamers
Return Type	Gamer

Misc

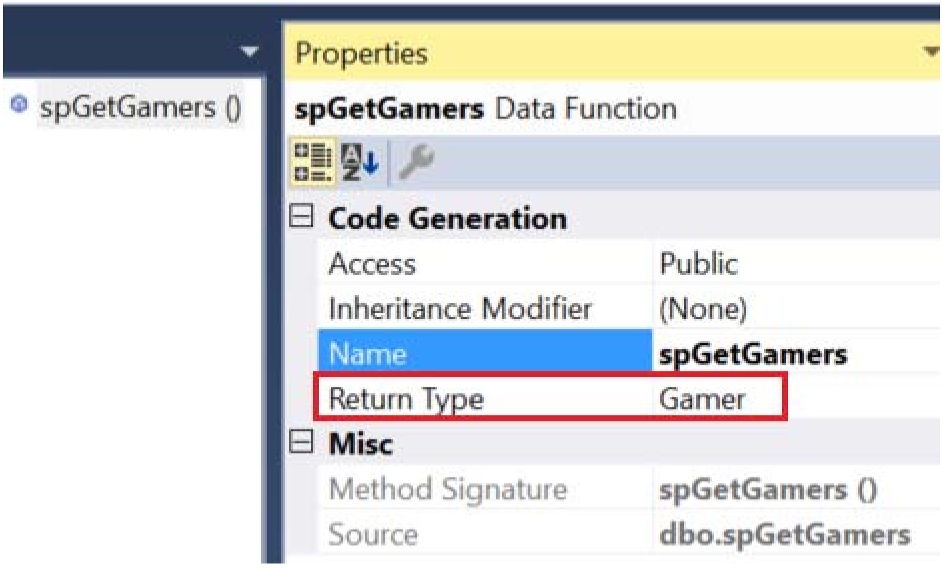
Method Signature	spGetGamers ()
Source	dbo.spGetGamers

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.

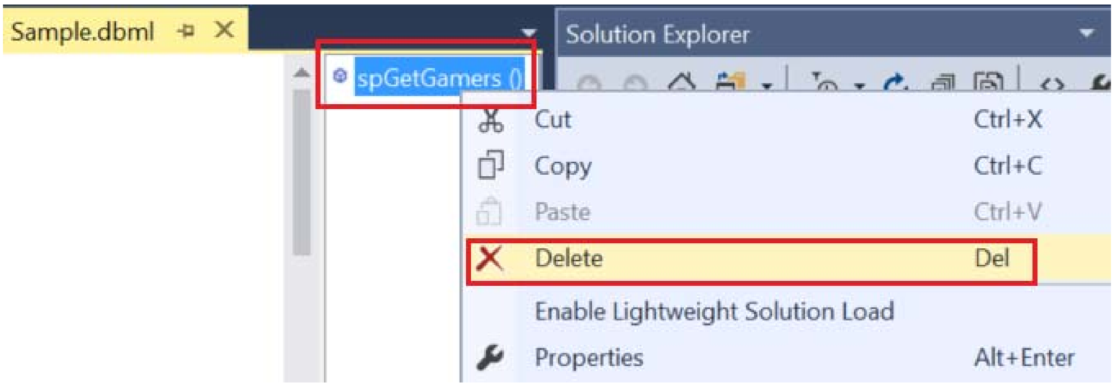




Drag into Gamer

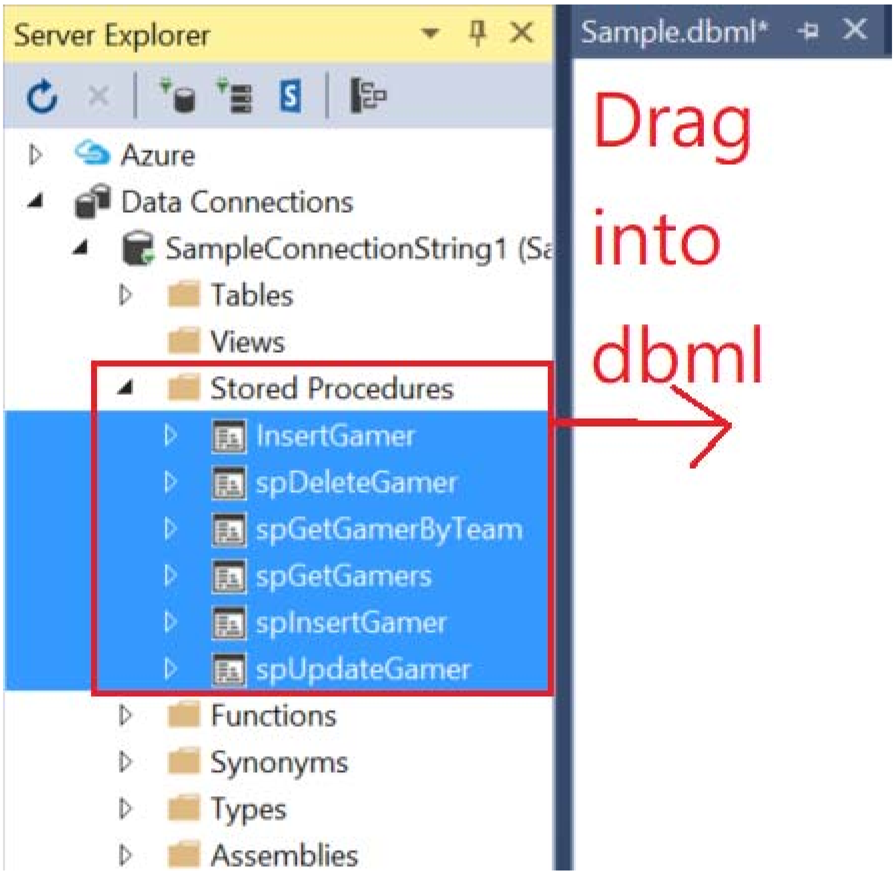


Please delete it before you continue this tutorial.

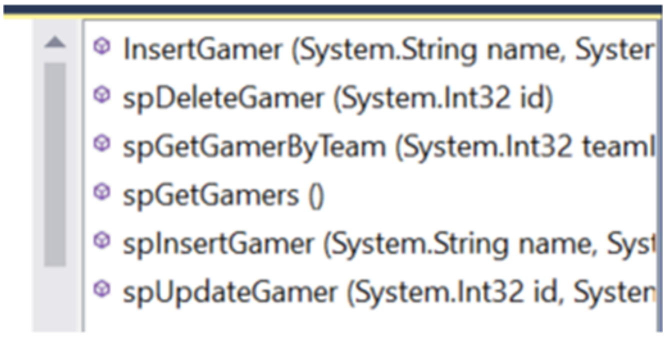


2.4. Sample.dbml - Stored procedure

Drag All stored procedures into dbml



Drag into dbml



```
InsertGamer (System.String name, Syster
spDeleteGamer (System.Int32 id)
spGetGamerByTeam (System.Int32 teaml
spGetGamers ()
spInsertGamer (System.String name, Syst
spUpdateGamer (System.Int32 id, Systen
```

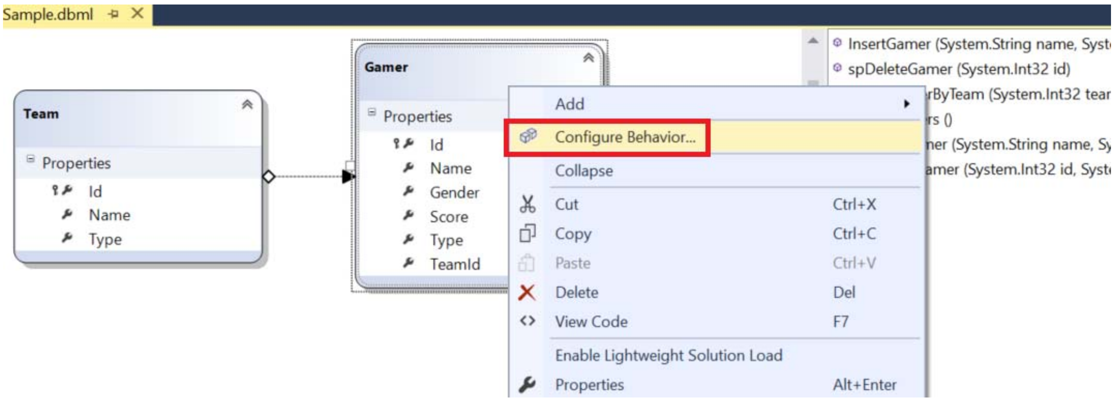
In dbml

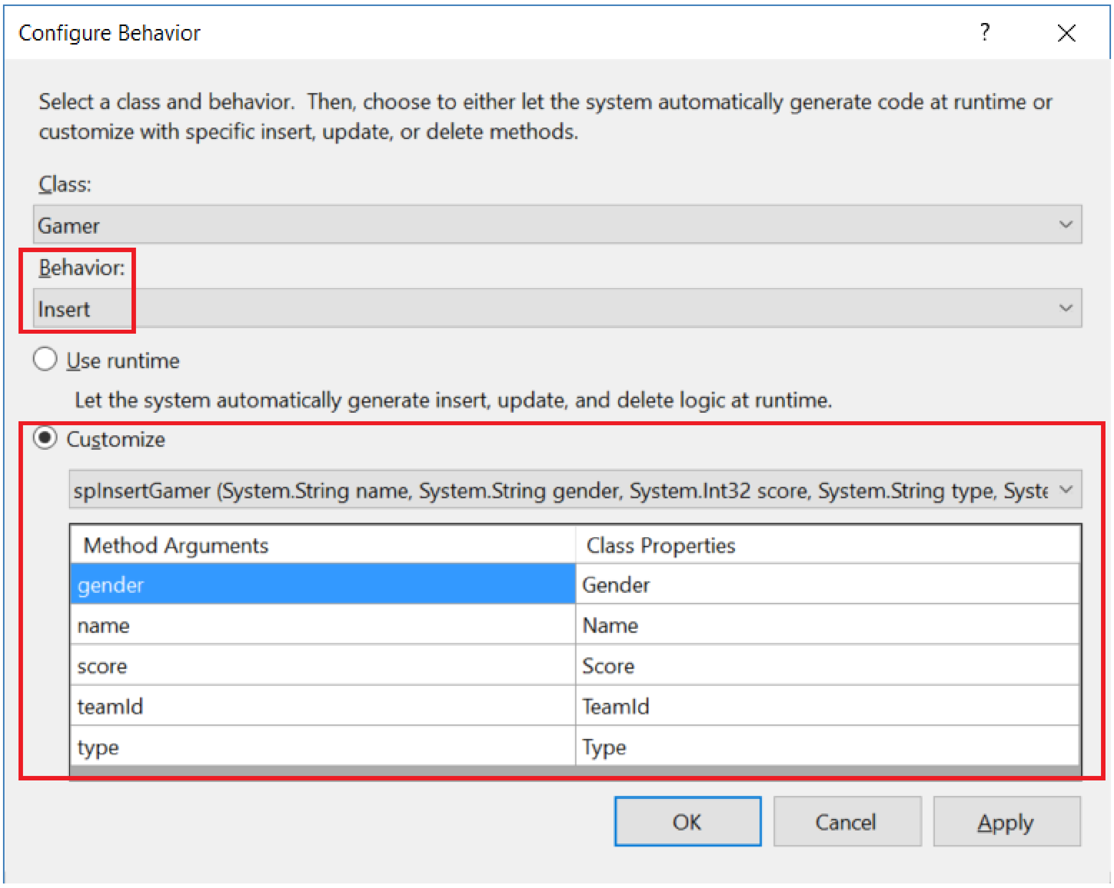
Gamer --> Right click --> Configure Behavior
-->
Behavior : Insert
Customize : spInsertGamer
Please confirm the mapping is right.

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.





Configure Behavior ? ✕

Select a class and behavior. Then, choose to either let the system automatically generate code at runtime or customize with specific insert, update, or delete methods.

Class:

Gamer

Behavior:

Insert

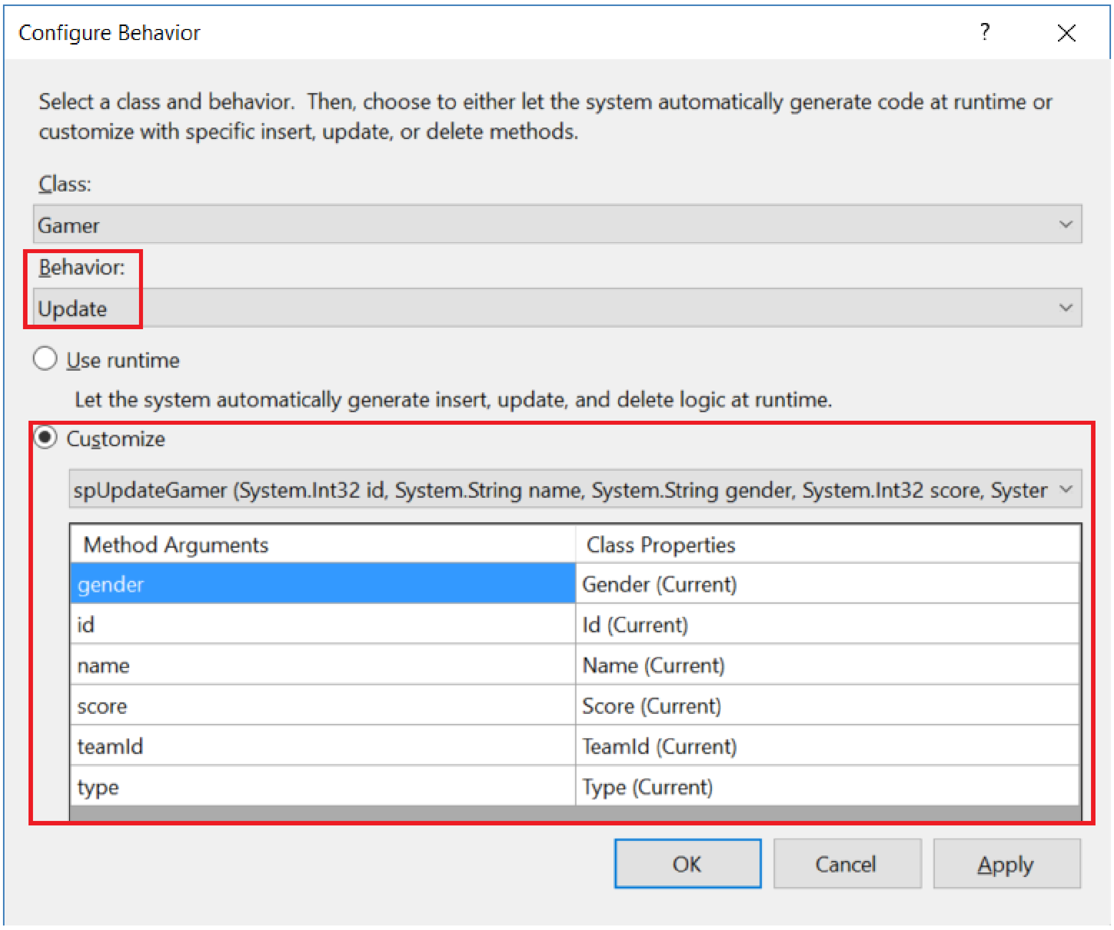
○ Use runtime

Let the system automatically generate insert, update, and delete logic at runtime.

● Customize

spInsertGamer (System.String name, System.String gender, System.Int32 score, System.String type, Syste

Method Arguments	Class Properties
gender	Gender
name	Name
score	Score
teamId	TeamId
type	Type



Configure Behavior

Select a class and behavior. Then, choose to either let the system automatically generate code at runtime or customize with specific insert, update, or delete methods.

Class:

Gamer

Behavior:

Update

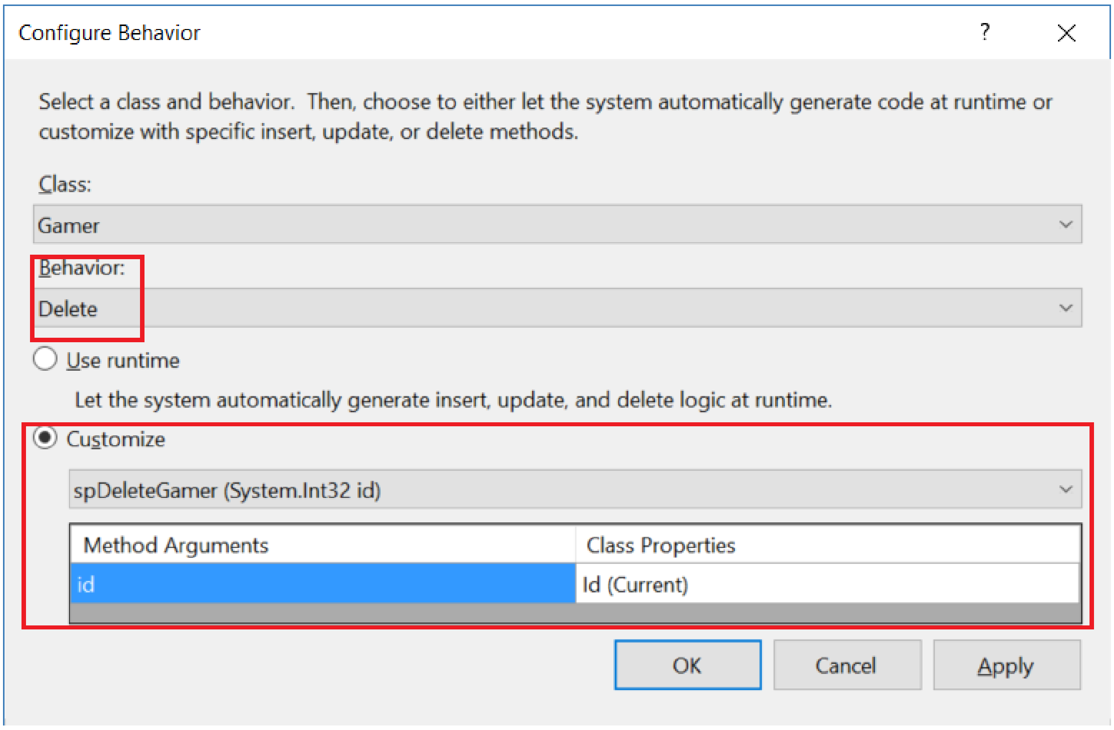
○ Use runtime

Let the system automatically generate insert, update, and delete logic at runtime.

● Customize

spUpdateGamer (System.Int32 id, System.String name, System.String gender, System.Int32 score, Syster

Method Arguments	Class Properties
gender	Gender (Current)
id	Id (Current)
name	Name (Current)
score	Score (Current)
teamId	TeamId (Current)
type	Type (Current)



Configure Behavior

Select a class and behavior. Then, choose to either let the system automatically generate code at runtime or customize with specific insert, update, or delete methods.

Class:

Gamer

Behavior:

Delete

○ Use runtime

Let the system automatically generate insert, update, and delete logic at runtime.

● Customize

spDeleteGamer (System.Int32 id)

Method Arguments	Class Properties
id	Id (Current)

2.5. WebForm2.aspx

2.5.1. WebForm2.aspx

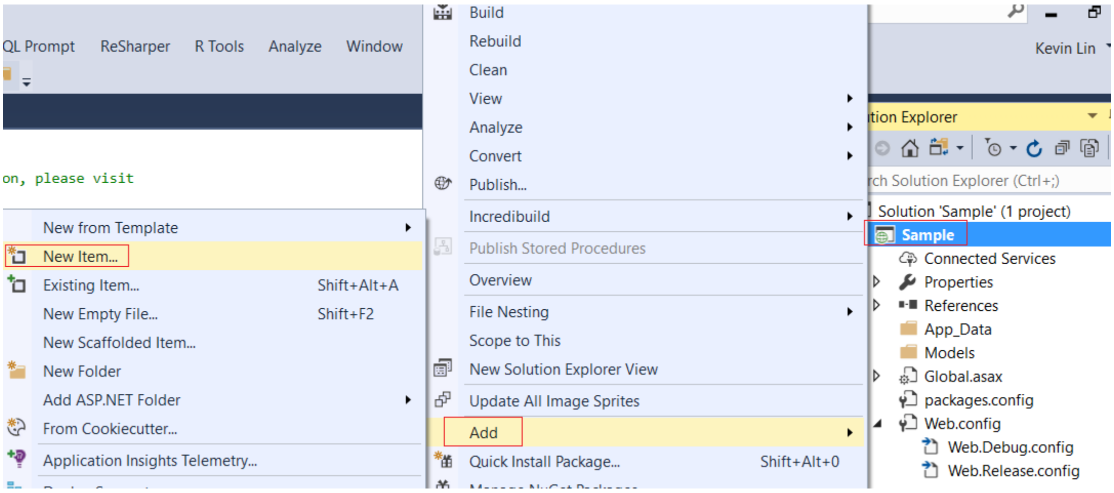
ProjectName --> Right Click --> Add --> New Item...

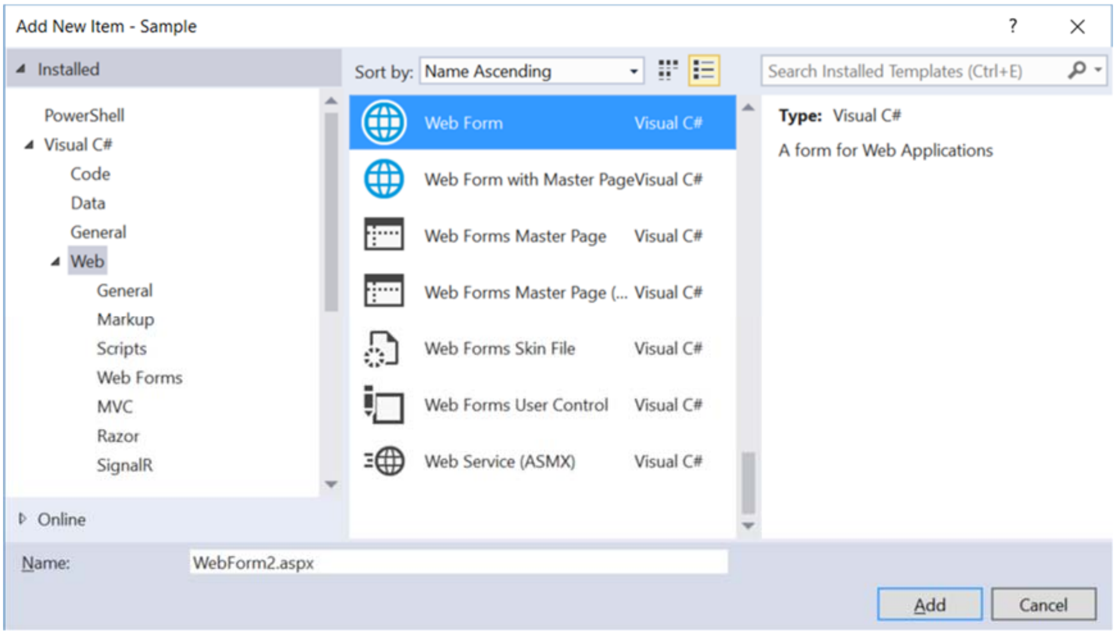
-->

WebForm

Name :

WebForm2.aspx




Add New Item - Sample

Sort by: Name Ascending

Type: Visual C#

A form for Web Applications

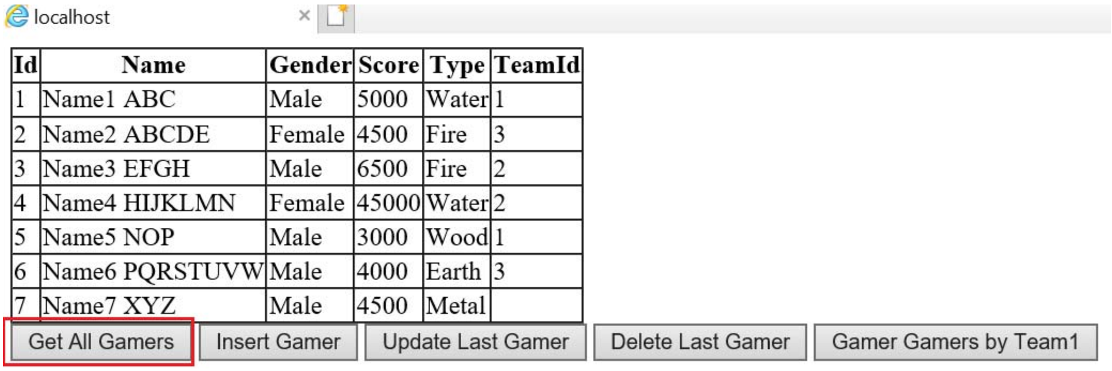
Name: WebForm2.aspx

Installed
PowerShell
Visual C#
Code
Data
General
Web
General
Markup
Scripts
Web Forms
MVC
Razor
SignalR
Online
Search Installed Templates (Ctrl+E)
Web Form Visual C#
Web Form with Master PageVisual C#
Web Forms Master Page Visual C#
Web Forms Master Page (... Visual C#
Web Forms Skin File Visual C#
Web Forms User Control Visual C#
Web Service (ASMX) Visual C#
Add
Cancel

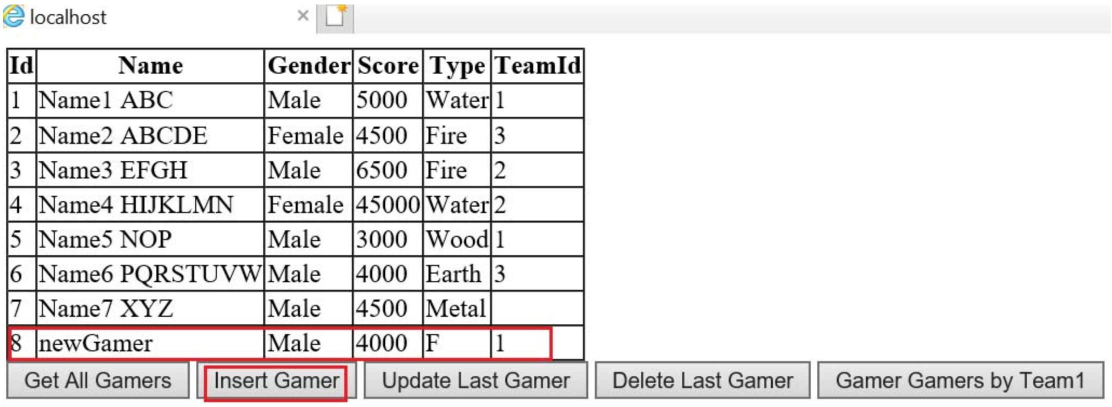
```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="WebForm2.aspx.cs" Inherits="Sample.WebForm2" %>
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title></title>
</head>
<body>
    <form id="form1" runat="server">
        <div>
            <asp:Label 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

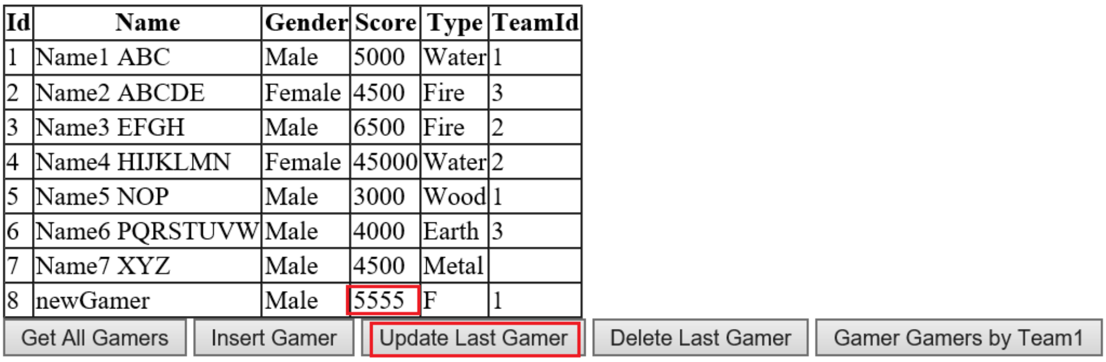
```
using System;
using System.Linq;
namespace Sample
{
    public partial class WebForm2 : System.Web.UI.Page
    {
        protected void Page_Load(object sender, EventArgs e)
        {
        }
        private void GetData()
        {
            using (SampleDataContext dbContext = new SampleDataContext())
            {
                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}";
            }
        }
    }
}
```



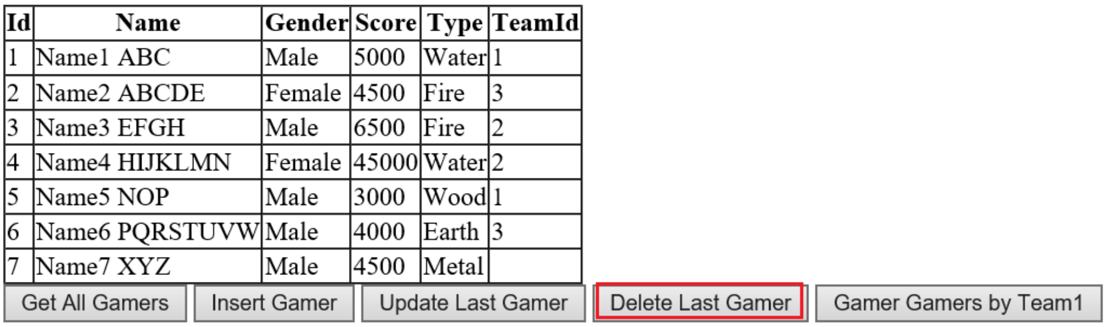
Id	Name	Gender	Score	Type	TeamId
1	Name1 ABC	Male	5000	Water	1
2	Name2 ABCDE	Female	4500	Fire	3
3	Name3 EFGH	Male	6500	Fire	2
4	Name4 HIJKLMN	Female	45000	Water	2
5	Name5 NOP	Male	3000	Wood	1
6	Name6 PQRSTUVW	Male	4000	Earth	3
7	Name7 XYZ	Male	4500	Metal	



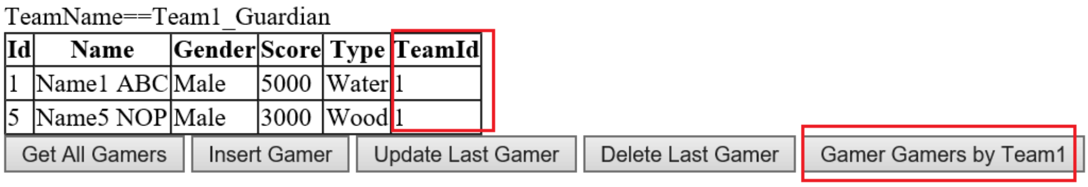
Id	Name	Gender	Score	Type	TeamId
1	Name1 ABC	Male	5000	Water	1
2	Name2 ABCDE	Female	4500	Fire	3
3	Name3 EFGH	Male	6500	Fire	2
4	Name4 HIJKLMN	Female	45000	Water	2
5	Name5 NOP	Male	3000	Wood	1
6	Name6 PQRSTUVW	Male	4000	Earth	3
7	Name7 XYZ	Male	4500	Metal	
8	newGamer	Male	4000	F	1



Id	Name	Gender	Score	Type	TeamId
1	Name1 ABC	Male	5000	Water	1
2	Name2 ABCDE	Female	4500	Fire	3
3	Name3 EFGH	Male	6500	Fire	2
4	Name4 HIJKLMN	Female	45000	Water	2
5	Name5 NOP	Male	3000	Wood	1
6	Name6 PQRSTUVW	Male	4000	Earth	3
7	Name7 XYZ	Male	4500	Metal	
8	newGamer	Male	5555	F	1



Id	Name	Gender	Score	Type	TeamId
1	Name1 ABC	Male	5000	Water	1
2	Name2 ABCDE	Female	4500	Fire	3
3	Name3 EFGH	Male	6500	Fire	2
4	Name4 HIJKLMN	Female	45000	Water	2
5	Name5 NOP	Male	3000	Wood	1
6	Name6 PQRSTUVW	Male	4000	Earth	3
7	Name7 XYZ	Male	4500	Metal	



TeamName==Team1_Guardian

Id	Name	Gender	Score	Type	TeamId
1	Name1 ABC	Male	5000	Water	1
5	Name5 NOP	Male	3000	Wood	1