(T16)討論LinqToSql的CRUD(Create、Read、Update、Delete)  
CourseGUID: 5ba9a6fe-7475-4b0c-8b99-bbcf7f5e2e1c  
=======================================================================  
(T16)討論LinqToSql的CRUD(Create、Read、Update、Delete)  
=======================================================================  
0. Summary

-----------

1. Web Form Application - Linq Query

1.1. TSQL

1.2. Set up SQL Authentication

1.3. Create Web Application

1.4.Web.config

-----------

2. Linq to SQL

2.1. Add Connection

2.2. Sample.dbml

2.3. WebForm1.aspx

2.3.1. WebForm1.aspx

2.3.2. WebForm1.aspx.cs - PrintGeneratedSql

2.4. SQL Profiler  
=======================================================================

0. Summary

1.

1.1.

Language Integrated整體 Query (LINQ) is a component between the LINQ query and the actual data source which includes SQL Server, XML documents, Objects in memory etc.

E.g. Linq to SQL provider can convert a Linq query to TSQL.

1.2.

LINQ query can be written by any .NET supported programming language, and it provides compile time error checking.

2.

Linq to Sql

Reference:

<https://docs.microsoft.com/en-us/dotnet/framework/data/adonet/sql/linq/>

LINQ to SQL is an ORM (Object Relational Mapping) framework in .NET Framework

that provides a run-time infrastructure

for managing relational data as strongly typed .Net objects.

The Linq to Sql provider can convert Linq query to TSql for Sql Server Database.

Linq to Sql supports Transactions, Views, and Stored Procedures.

Because Linq to Sql use strongly typed .Net objects,

it has intellisense support, compile time error checking and debugging support

3.

Linq to Sql

**.Net Application**

   |                                                                ↑

   |                                                                 |

   ↓                                                                |

from gamer in dbContext.Gamers                gamer objects

where gamer.Gender == "Female"                   ↑

orderby gamer.Score descending                      |

select gamer;                                                 |

   |                                                                |

   |                                                                |

   ↓

**Linq to Sql Provider**

   |                                                                ↑

   |                                                                |

   ↓                                                                |

exec sp\_executesql N'SELECT [T0].[Id]          Matching data rows

[T0].[Name]                                               are returned

[T0].[Gender]                                                  ↑

[T0].[Score]                                                    |

[T0].[Team]                                                    |

FROM [dbo].[Gamer]  AS [t0]                            |

WHERE [t0].[Gender]=@p0                              |

ORDER BY [t0].[Score DESC']',N'@p0=N'Male'    |

   |                                                                 |

   |                                                                 |

   ↓                                                                 |

**MS Sql Server**

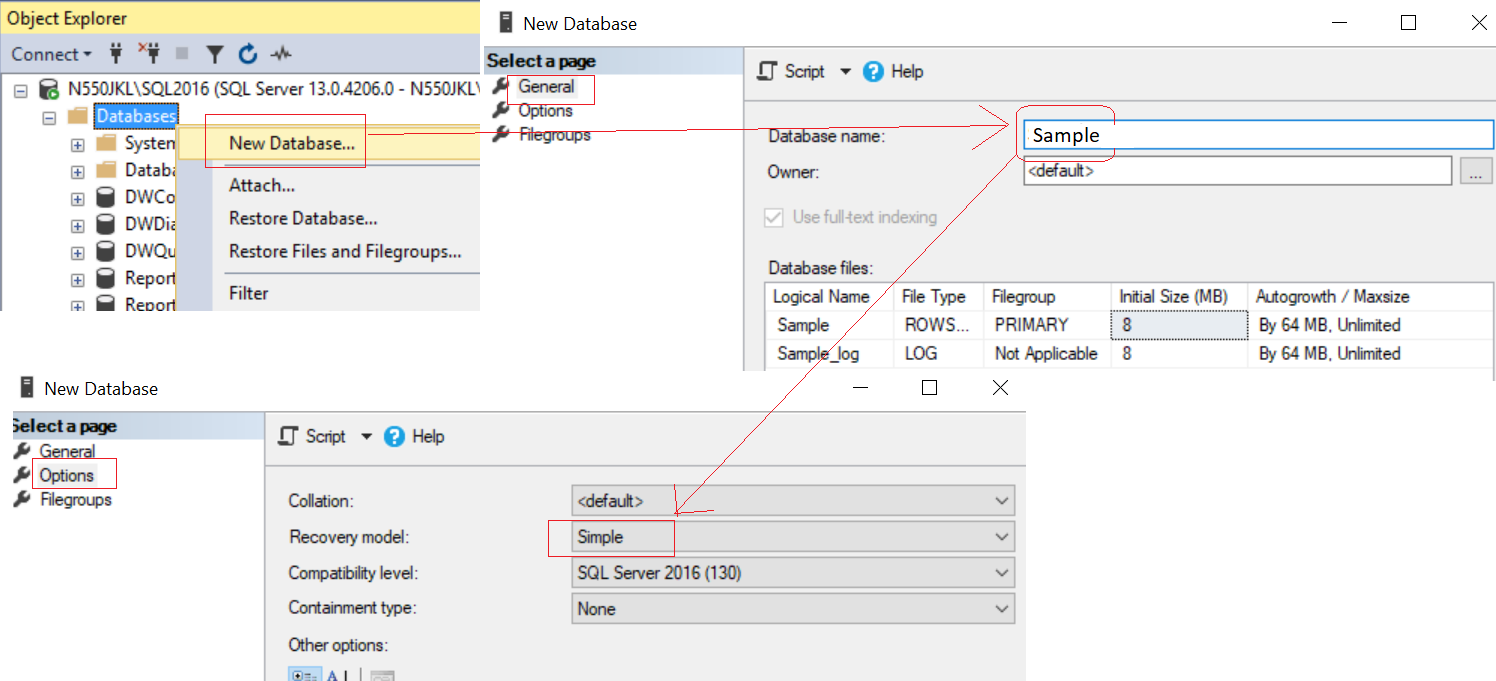
1. Web Form Application - Linq Query

1.1. TSQL

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.

\*/

--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

--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

1.2. 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.









1.3. Create Web Application

Open Visual Studio, I am currently using VS2017

If you don't have it, you may following the instruction here to download.

<http://ithandyguytutorial.blogspot.com/2017/10/ch00install-visual-studio-2017-offline.html>

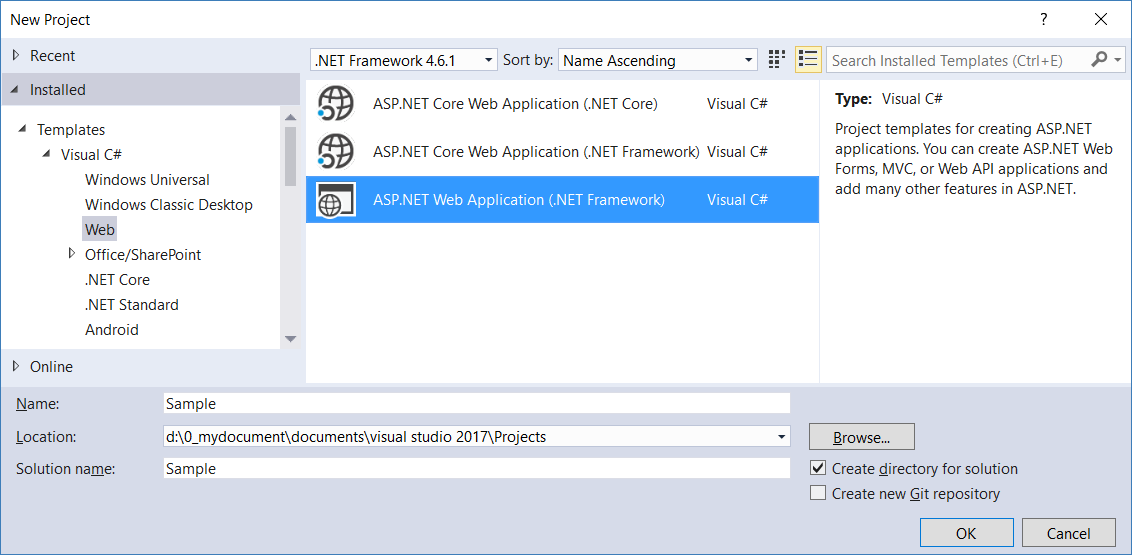
New Project --> Web -->[ASP.NET](http://asp.net/)**Web Application (.Net Framework)**

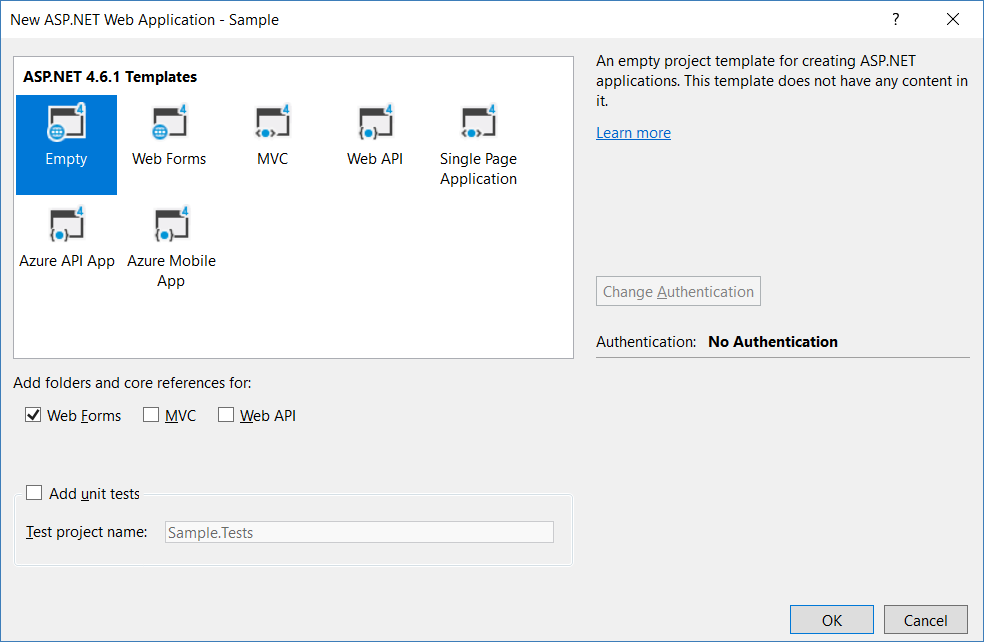
-->

Name:

**Sample**

--> **Empty** --> Select "**Web Forms**"  --> OK





1.4.Web.config

Add connection String

If you use Linq to Sql, you don't have to set this connection string.

I personally already get used to set it by my own.

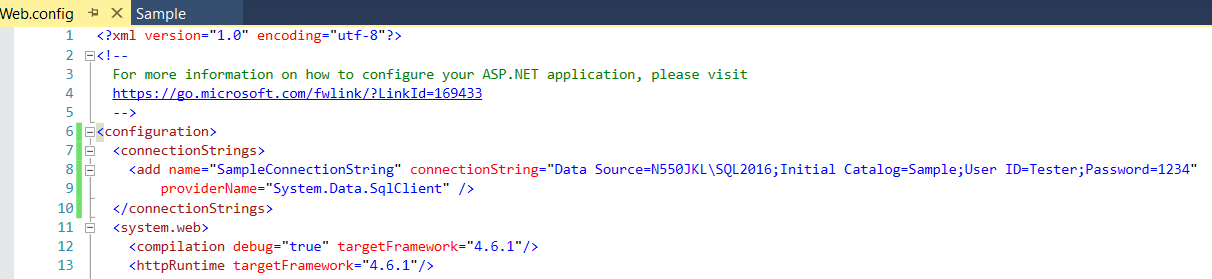
<configuration>

  <connectionStrings>

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

        providerName="System.Data.SqlClient" />

  </connectionStrings>



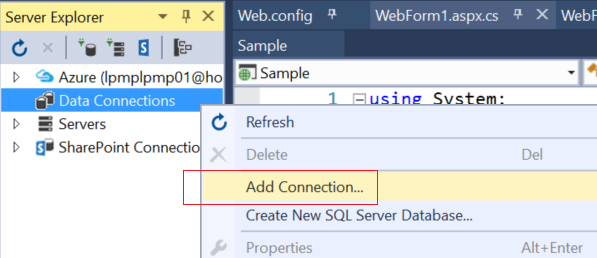
2. Linq to SQL

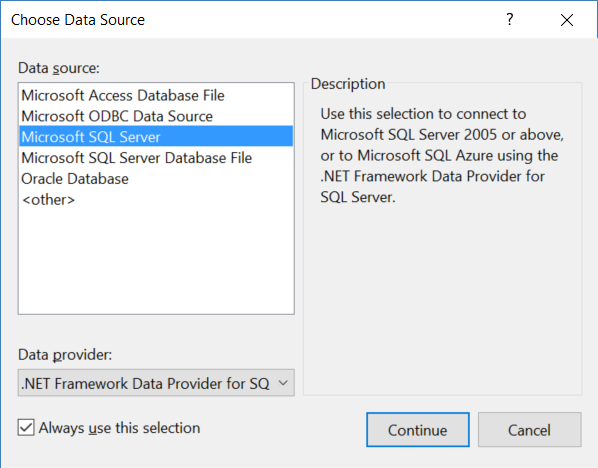
2.1. Add Connection

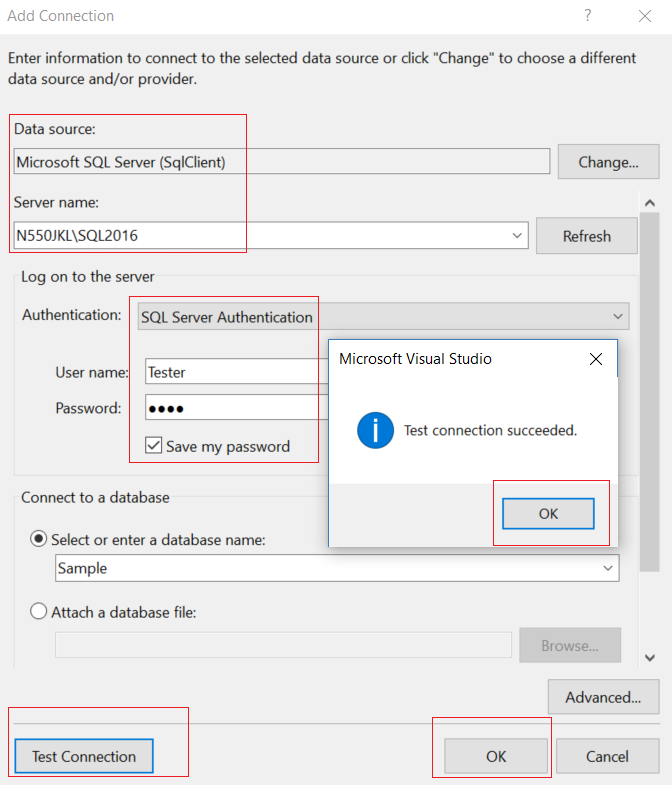
Server Explorer --> Data Connections --> Right click --> Add Connection...

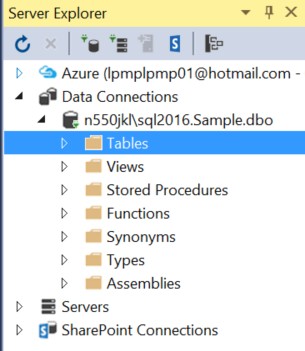
--> Microsoft SQL server -->

Enter your server and database details ....









2.2. Sample.dbml

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

--> Linq to SQL classes -->

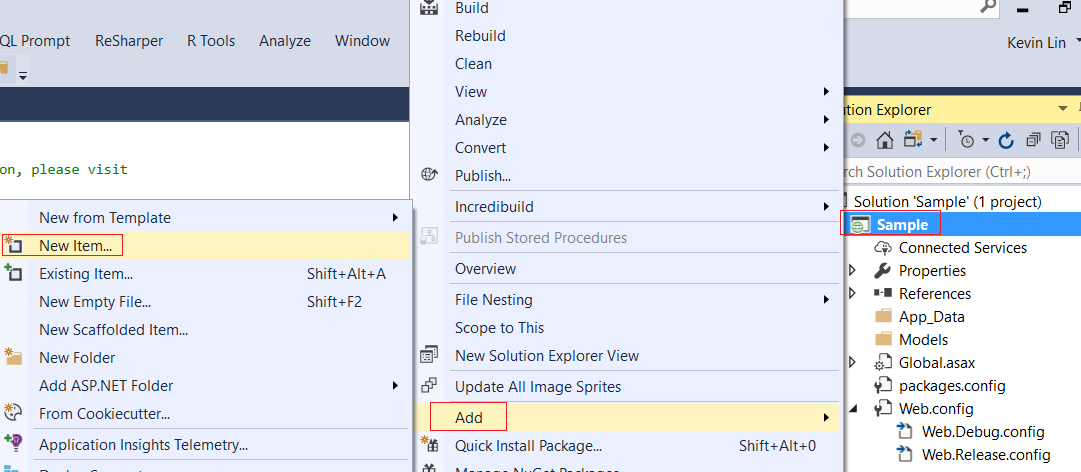
Name : **Sample.dbml**

I name it as "Sample.dbml",

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

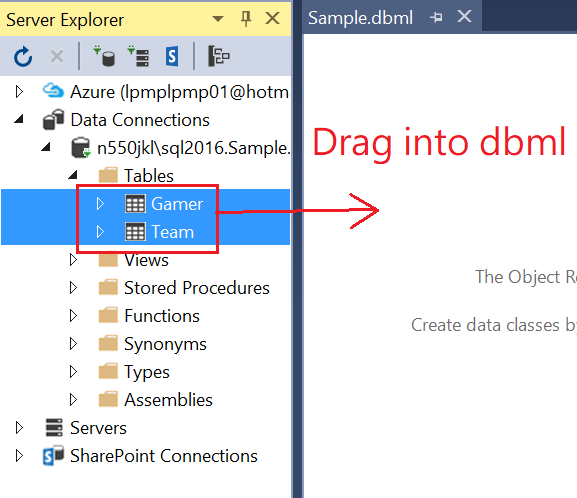
-->

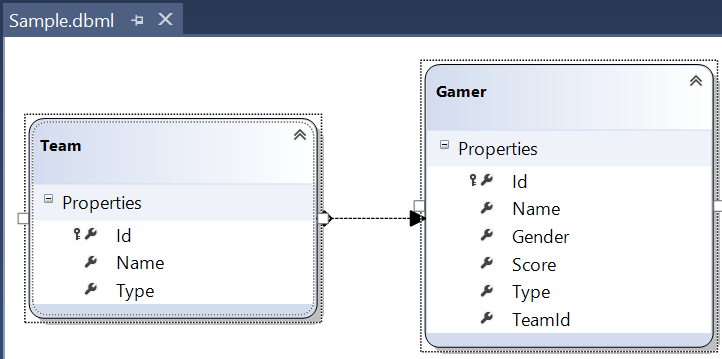
Drag Table from Server Explorer into DBML



Graphical user interface, application

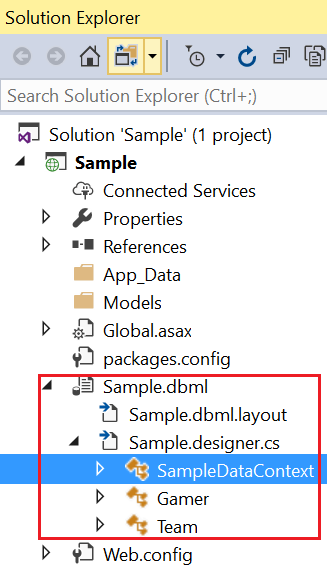
Description automatically generated





Save the dbml, it will generate the following files.

The DataContext context is the entry point to database.



2.3. WebForm1.aspx

2.3.1. WebForm1.aspx

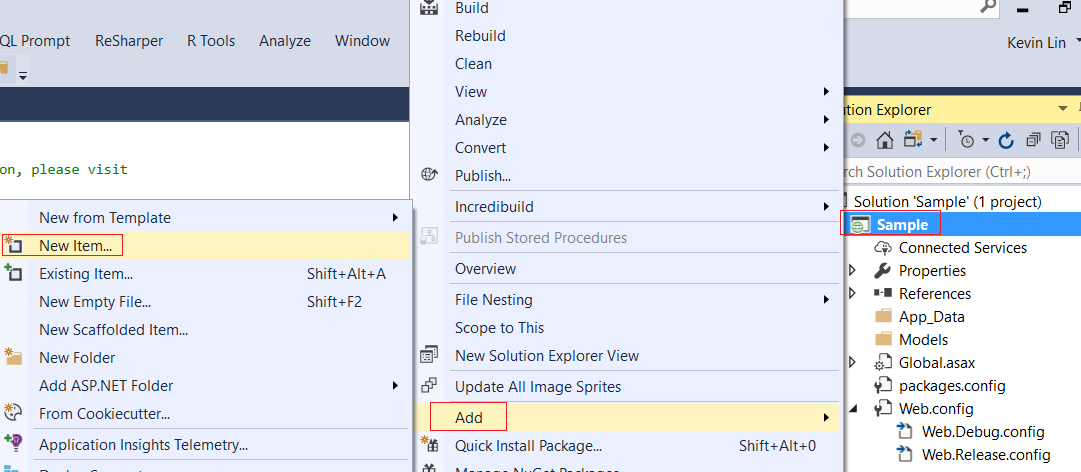
ProjectName --> Right Click --> Add --> New Item...

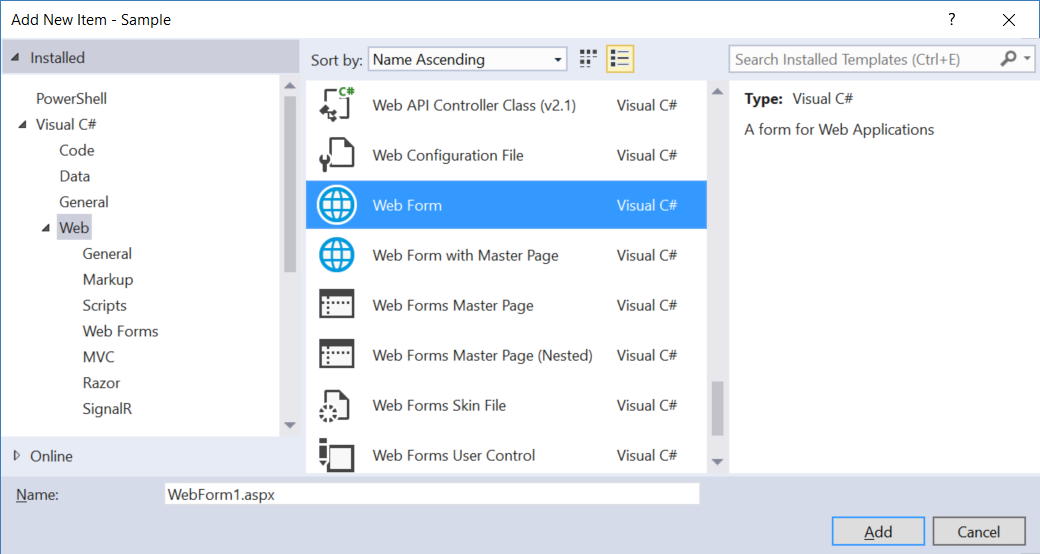
-->

**WebForm**

Name :

**WebForm1.aspx**





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

<!DOCTYPE html>

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

<head runat="server">

    <title></title>

</head>

<body>

    <form id="form1" runat="server">

        <div>

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

            <asp:Button ID="btnGetMaleData" runat="server" Text="Get Male Data" OnClick="btnGetMaleData\_Click" />

            <asp:Button ID="btnGetData" runat="server" Text="Get Data" OnClick="btnGetData\_Click" />

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

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

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

        </div>

    </form>

</body>

</html>

2.3.2. WebForm1.aspx.cs - PrintGeneratedSql

using System;

using System.Linq;

namespace Sample

{

    public partial class WebForm1 : System.Web.UI.Page

    {

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

        {

            //GetData();

        }

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

        {

            using (SampleDataContext dbContext = new SampleDataContext())

            {

                // Write the generated sql query to the webform

                dbContext.Log = Response.Output;

                //// Write the generated sql query to the Console window

                //dbContext.Log = Console.Out;

                //dbContext.Log = Response.Output;  and

                //dbContext.Log = Console.Out;

                //You may only choose one to use.

                IOrderedQueryable<Gamer> gamerQueryable = from gamer in dbContext.Gamers

                    where gamer.Gender == "Male"

                    orderby gamer.Score descending

                    select gamer;

                GridView1.DataSource =

                    gamerQueryable;

                Response.Write($"<br/>gamerQueryable.ToString()<br/>{gamerQueryable.ToString()}<br/><br/>");

                Response.Write($"<br/>dbContext.GetCommand(gamerQueryable).CommandText<br/>{dbContext.GetCommand(gamerQueryable).CommandText}<br/><br/>");

                Response.Write($"<br/>dbContext.GetCommand(gamerQueryable).CommandType<br/>{dbContext.GetCommand(gamerQueryable).CommandType}<br/><br/>");

                GridView1.DataBind();

            }

        }

        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 btnGetData\_Click(object sender, EventArgs e)

        {

            GetData();

        }

        protected void btnInsert\_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 btnUpdate\_Click(object sender, EventArgs e)

        {

            using (SampleDataContext dbContext = new SampleDataContext())

            {

                //Get the last gamer

                int lastId = dbContext.Gamers.Count();

                Gamer gamer = dbContext.Gamers.SingleOrDefault(

                    x => x.Id == lastId);

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

                dbContext.SubmitChanges();

            }

            GetData();

        }

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

        {

            using (SampleDataContext dbContext = new SampleDataContext())

            {

                //Get the last gamer

                int lastId = dbContext.Gamers.Count();

                Gamer gamer = dbContext.Gamers.SingleOrDefault(

                    x => x.Id == lastId);

                //delete the last gamer from dbContext

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

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

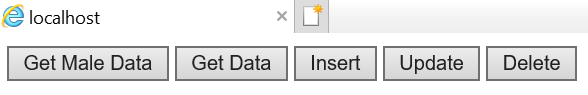
            }

            GetData();

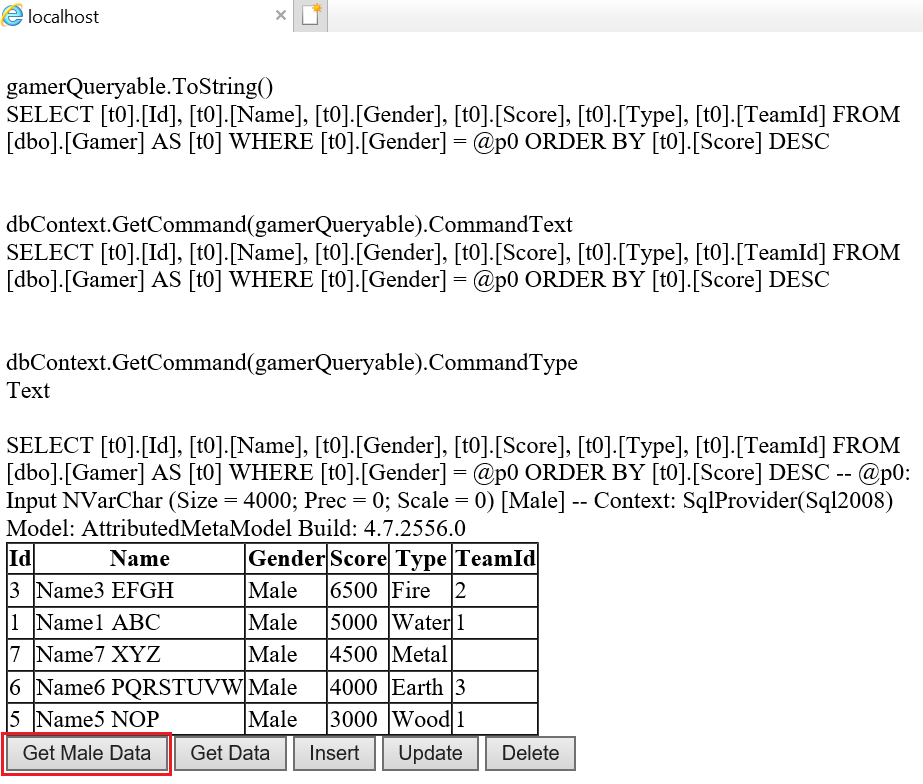
        }

    }

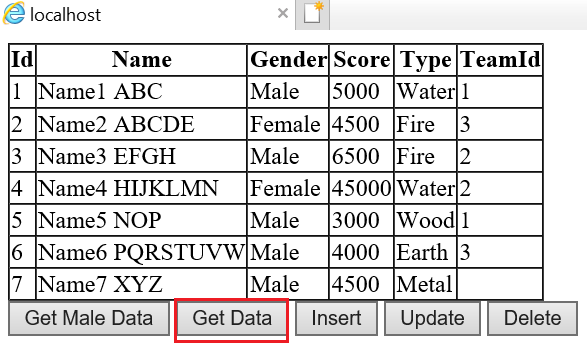
}



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



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



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

Table

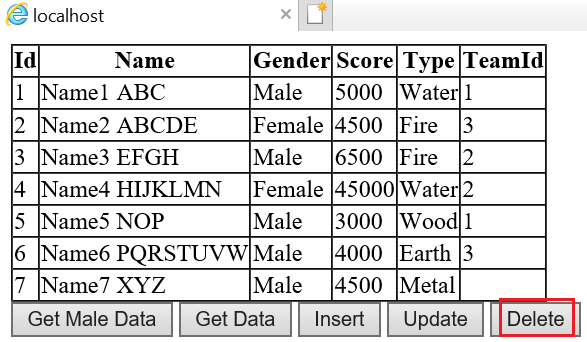
Description automatically generated

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

Table

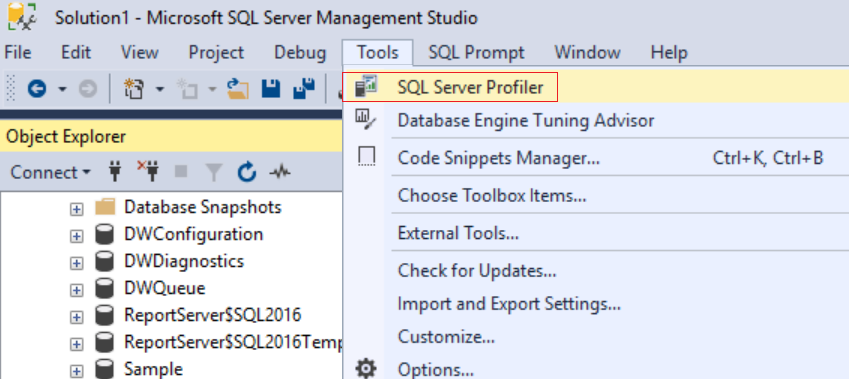
Description automatically generated

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



2.4. SQL Profiler

Tools --> SQL Server Profiler



Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

Now, go back to VS2017, and run WebForm2.aspx again

You will see Linq to SQL provider convert Linq to TSQL.

